TURUN
YLIOPISTO
UNIVERSITY
OF TURKU

LIDAR BASED MULTI-SENSOR FUSION FOR LOCALIZATION, MAPPING, AND TRACKING

Qingqing Li

TURUN YLIOPISTON JULKAISUJA – ANNALES UNIVERSITATIS TURKUENSIS
SARJA – SER. F OSA – TOM. 40 | TECHNICA – INFORMATICA | TURKU 2024

TURUN
YLIOPISTO
UNIVERSITY
OF TURKU

LIDAR BASED MULTI-SENSOR FUSION FOR LOCALIZATION, MAPPING, AND TRACKING

Qingqing Li

TURUN YLIOPISTON JULKAISUJA – ANNALES UNIVERSITATIS TURKUENSIS
SARJA – SER. F OSA – TOM. 40 | TECHNICA – INFORMATICA| TURKU 2024

University of Turku

Faculty of Technology
Department of Computing
Information and Communication Technology
Doctoral Programme in Technology

Supervised by

Professor, Tomi Westerlund
University of Turku

Professor, Zhuo Zou
Fudan University

Reviewed by

Professor, George Nikolakopoulos
Luleå University of Technology

Research Professor, Antero Kukko
Finnish Geospatial Research Institute

Opponent

Associate Professor, Evangelos Boukas
Technical University of Denmark

Serial F 40
ISBN 978-951-29-9735-0 (PRINT)
ISBN 978-951-29-9736-7 (PDF)
ISSN 2736-9390 (PRINT)
ISSN 2736-9684 (ONLINE)
Painosalama Oy, Turku, Finland, 2024

To my wife Qianqian, my parents, and my brother
...and to my nephew Yiyi

UNIVERSITY OF TURKU
Faculty of Technology
Department of Computing
Information and Communication Technology
LI, QINGQING: LiDAR Based Multi-Sensor Fusion for Localization, Mapping, and Tracking
Doctoral dissertation, 122 pp.
Doctoral Programme in Technology
May 2024

ABSTRACT

The development of fully autonomous driving vehicles has become a key focus for both industry and academia over the past decade, fostering significant progress in situational awareness abilities and sensor technology. Among various types of sensors, the LiDAR sensor has emerged as a pivotal component in many perception systems due to its long-range detection capabilities, precise 3D range information, and reliable performance in diverse environments. With advancements in LiDAR technology, more reliable and cost-effective sensors have shown great potential for improving situational awareness abilities in widely used consumer products. By leveraging these novel LiDAR sensors, researchers now have a diverse set of powerful tools to effectively tackle the persistent challenges in localization, mapping, and tracking within existing perception systems.

This thesis explores LiDAR-based sensor fusion algorithms to address perception challenges in autonomous systems, with a primary focus on dense mapping and global localization using diverse LiDAR sensors. The research involves the integration of novel LiDARs, IMU, and camera sensors to create a comprehensive dataset essential for developing advanced sensor fusion and general-purpose localization and mapping algorithms. Innovative methodologies for global localization across varied environments are introduced. These methodologies include a robust multi-modal LiDAR inertial odometry and a dense mapping framework, which enhance mapping precision and situational awareness. The study also integrates solid-state LiDARs with camera-based deep-learning techniques for object tracking, refining mapping accuracy in dynamic environments. These advancements significantly enhance the reliability and efficiency of autonomous systems in real-world scenarios.

The thesis commences with an introduction to innovative sensors and a data collection platform. It proceeds by presenting an open-source dataset designed for the evaluation of advanced SLAM algorithms, utilizing a unique ground-truth generation method. Subsequently, the study tackles two localization challenges in forest and urban environments. Furthermore, it highlights the MM-LOAM dense mapping framework. Additionally, the research explores object-tracking tasks, employing both camera and LiDAR technologies for human and micro UAV tracking.

KEYWORDS : SLAM, LiDAR, sensor fusion, robotics, localization, autonomous driving, autonomous system, harvester, object tracking.

TURUN YLIOPISTO
Faculty of Technology
Department of Computing
Information and Communication Technology
LI, QINGQING: LiDAR Based Multi-Sensor Fusion for Localization, Mapping, and Tracking
Väitöskirja, 122 s.
Doctoral Programme in Technology
May 2024

TIIVISTELMÄ

Viimeisen vuosikymmenen aikana täysin itseohjautuvien ajoneuvojen kehitys on herättänyt laajaa kiinnostusta niin teollisuudessa kuin tiedemaailmassakin, mikä on merkittävästi edistänyt tilannetietoisuuden ja anturiteknologian kehitystä. Erityisesti LiDAR-anturit ovat nousseet keskeiseen rooliin monissa havainnointijärjestelmissä niiden tarjoaman pitkän kantaman havaintokyvyn, tarkan 3D-etäisyystiedon ja luotettavan suorituskyvyn ansiosta. LiDAR-teknologian kehittyminen on mahdollistanut entistä luotettavampien ja kustannustehokkaampien antureiden käytön, mikä puolestaan on osoittanut suurta potentiaalia parantaa laajasti käytettyjen kuluttajatuotteiden tilannetietoisuutta. Uusien LiDAR-antureiden hyödyntäminen tarjoaa tutkijoille monipuolisen valikoiman tehokkaita työkaluja, joiden avulla voidaan ratkaista paikannuksen, kartoituksen ja seurannan haasteita nykyisissä havaintojärjestelmissä.

Tässä väitöskirjassa tutkitaan LiDAR-pohjaisia sensorifuusioalgoritmeja. Tutkimuksen pääpaino on tiheässä kartoituksessa ja globaalissa paikan-nuksessa erilaisten LiDAR-anturien avulla. Tutkimuksessa luodaan kattava tietokanta uusien LiDAR-, IMU- ja kamera-antureiden tuottamasta datasta. Tietokanta on välttämätön kehittyneiden anturifuusioalgoritmien ja yleiskäyttöisten paikannus- ja kartoitusalgoritmien kehittämiseksi. Tämän lisäksi väitöskirjassa esitellään innovatiivisia menetelmiä globaaliin paikannukseen erilaisissa ympäristöissä. Esitellyt menetelmät kartoituksen tarkkuuden ja tilannetietoisuuden parantamiseksi ovat muun muassa modulaarinen monen LiDAR-anturin odometria ja kartoitus, toimintavarma multimodaalinen LiDAR-inertiamittau-sjärjestelmä ja tiheä kartoituskehys. Tutkimus integroi myös kiinteät LiDAR -anturit kamerapohjaisiin syväoppimismenetelmiin kohteiden seurantaa varten parantaen kartoituksen tarkkuutta dynaamisissa ympäristöissä. Näiden edistysaskeleiden avulla autonomisten järjestelmien luotettavuutta ja tehokkuutta voidaan merkittävästi parantaa todellisissa käyttöympäristöissä.

Väitöskirja alkaa esittelemällä innovatiiviset anturit ja tiedonkeruualustan. Tämän jälkeen esitellään avoin tietokanta, jonka avulla voidaan arvioida kehittyneitä paikannus- ja kartoitusalgoritmeja hyödyntäen ainutlaatuista perustotuuden kehittämismenetelmää. Työssä käsitellään myös kahta haastavaa paikannusympäristöä: metsä- ja kaupunkiympäristöä. Lisäksi tarkastellaan kohteen seurantatehtäviä sekä kameraettä LiDAR-tekniikoilla ihmisten ja pienten droonien seurannassa.

Avainsanat: SLAM, LiDAR, anturifuusio, robotiikka, lokalisointi, autonominen ajo, autonominen järjestelmä, harvesteri, objektien seuranta.

Acknowledgements

I am profoundly grateful to all colleagues and friends who have supported me throughout this incredible journey. I still vividly remember when I first arrived in Finland—the serene, welcoming atmosphere, the kindness of the people, and the world-class research environment—all of which inspired me to explore further and pursue my Ph.D. dream here. Today, I feel incredibly fortunate to have made such a wonderful decision and to have met so many amazing people who have had a profound impact on my life.

First and foremost, I would like to express my sincere appreciation to my supervisor and friend, Prof. Tomi Westerlund. His unwavering support, invaluable mentorship, and critical thinking have been indispensable to me throughout this journey. His profound expertise, insightful guidance, and genuine encouragement have fueled my passion for research and enabled me to explore a diverse range of engaging topics. I am deeply grateful for his continuous support and guidance, both in academia and in life.

Secondly, I also want to extend my heartfelt thanks to Prof. Zhuo Zou, whose support has been invaluable during this journey. Prof. Zhuo's deep understanding of the subject matter, combined with his insightful feedback and constructive criticism, have greatly enriched my research endeavors. His continuous encouragement and guidance have not only helped me navigate through various challenges but have also played a significant role in shaping my academic and professional growth.

Additionally, I want to extend my deepest gratitude to my wife, Qianqian, whose trust and constant support have been a source of strength for me during this journey. The decision to study abroad thousands of miles away from my home country was not an easy one, but time has proven that our relationship is strong and resilient. Your support, optimistic attitude, and beautiful smile have encouraged me every step along the way. I am forever grateful to my parents and my brother for their steadfast support and encouragement. Their constant support and sacrifices throughout my life have motivated me to keep pushing forward. Despite being physically apart in recent years, their encouragement has always inspired me to strive for more. I am truly thankful to have them as my family.

Moreover, I want to express my gratitude to all my colleagues and friends at TIERS lab. Special thanks to my dear friend, Jorge Peña Queralta, whose constant motivation, and curiosity for science, along with his innovative ideas, have made

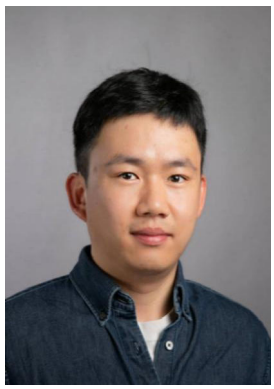
this journey more exciting. I will never forget the countless days spent preparing experiments, lectures, and papers before deadlines, as well as the movie nights and skiing trips we shared. Without your help and wisdom, this journey would have been far less fulfilling. I also want to thank Xianjia Yu, whose invaluable assistance has made this journey truly remarkable. My life here would be much less colorful without your help. I am also grateful to Tuan and Paavo for their passion for research, stimulating discussions, and constant encouragement. I extend my thanks to Ha Sier, Paola, Salma, Yuki, Daniel, Marius, Iacopo, and all the others who have walked this path with me, contributing to the formation of an exceptional team at TIERS.

Furthermore, I am immensely thankful to Henri Riihimäki, Peter Leander, Markus Karppinen, and my colleagues at Prefor Oy, who have provided me with excellent opportunities to apply my research in real-world applications. Your support has given me a unique perspective on my research journey and has greatly contributed to this thesis.

In the end, I would like to express my gratitude to the reviewer and pre-examiner whose critical feedback has greatly contributed to the quality of this thesis. Additionally, I want to acknowledge the support provided by the University of Turku Graduate School, the HPY Research Foundation, and the Finnish Foundation for Technology Promotion (TES). Their support has been instrumental in facilitating my research endeavors and ensuring the successful completion of this thesis.

Once again, I want to express my deepest gratitude to everyone mentioned above and to all those who have been part of my academic and personal journey. I am forever thankful for your presence in my life.

Helsinki, May 6th,2024
Qingqing Li



QINGQING LI
Qingqing Li received a B.Sc. in Electronic Engineering in 2016 and an M.Eng. in Electronics and Communication Engineering in 2018, both from Fudan University, China. Since 2019, he has been a researcher at the Turku Intelligent Embedded and Robotic Systems (TIERS) Lab, Faculty of Technology, University of Turku. His research interests include sensor fusion, SLAM, multi-robot cooperation, robotic perception,and deep learning.

Table of Contents

Abbreviations

ALS	Airborne Laser Scanning
CPU	Central processing unit
DARPA	Defense Advanced Research Projects Agency
DBH	Diameter at the breast height
DL	Deep learning
DOF	Degree of freedom
DT	Delaunay triangulation
EKF	Extended Kalman filter
FN	False negatives
FP	False positive
FoV	Field of view
GICP	Generalized Iterated Closest Point
GNSS	Global navigation satellite system
GPS	Global Positioning system
GPU	Graphics processing unit
IMU	Inertial measurement unit
IoU	Intersection over Union
ICP	Iterative closest point
LCD	Loop Closure Detection
LiDAR	Light Detection and Ranging
LOAM	Lidar odometry and mapping

MAV	Micro aerial vehicle
MINS	multi-sensor aided inertial navigation system
MLS	Mobile laser scanning
MM	Multiple model
MOCAP	Motion capture
MSE	mean square distance
MSCKF	Multi-State Constraint Kalman Filter
KD-tree	K Dimensional tree
KITTI	Karlsruhe Institute of Technology and Toyota Technological Institute
NDT	Normal distribution transforms
PCL	Point cloud library
PTP	Precise timestamp protocol
RAM	Random-access memory
RL	Reinforcement learning
ROS	Robot Operating System
RTK	Real-Time Kinematic position
SAR	search and rescue
SDP	Sequential Decision Process
SEIFs	Sparse extended information filters
SLAM	Simultaneous localization and mapping
SOTA	State of the art
TLS	Terrestrial laser scan
TP	True positive
UAV	Unmanned aerial vehicle
UGV	Unmanned ground vehicle

UWB	Ultra-wideband
VINS	Visual-inertial navigation systems
VIO	Visual-inertial odometry
VRPN	Virtual-Reality Peripheral Network
WGS	World Geodetic System
YOLO	You only look once

List of Original Publications

This dissertation is based on the following original publications:

I **Qingqing, L.**, Queralta, J.P., Gia, T.N., Zou, Z. and Westerlund, T., 2020. Multi-sensor fusion for navigation and mapping in autonomous vehicles: Accurate localization in urban environments. Unmanned Systems, 8(03), pp.229-237.

II **Qingqing, L.**, Nevalainen, P., Peña Queralta, J., Heikkonen, J. and Westerlund, T., 2020. Localization in unstructured environments: Towards autonomous robots in forests with delaunay triangulation. Remote Sensing, 12(11), p.1870.

III **Qingqing, L.**, Xianjia, Y., Queralta, J.P. and Westerlund, T., 2021, December. Adaptive lidar scan frame integration: Tracking known mavs in 3d point clouds. In 2021 20th International Conference on Advanced Robotics (ICAR) (pp. 1079-1086). IEEE.

IV **Qingqing, L.**, Xianjia, Y., Queralta, J.P. and Westerlund, T., 2022, October. Multi-modal lidar dataset for benchmarking general-purpose localization and mapping algorithms. In 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (pp. 3837-3844). IEEE.

V **Qingqing, L.**, Taipalmaa, J., Queralta, J.P., Gia, T.N., Gabbouj, M., Tenhunen, H., Raitoharju, J. and Westerlund, T., 2020, November. Towards active vision with UAVs in marine search and rescue: Analyzing human detection at variable altitudes. In 2020 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR) (pp. 65-70). IEEE.

VI **Qingqing, L.**, Queralta, J.P., Gia, T.N., Zou, Z., Tenhunen, H. and Westerlund, T., 2019, September. Detecting water reflection symmetries in point clouds for camera position calibration in unmanned surface vehicles. In 2019 19th International Symposium on Communications and Information Technologies (ISCIT) (pp. 507-512). IEEE.

VII Sier, H.,**Qingqing, L.**, Yu, X., Peña Queralta, J., Zou, Z. and Westerlund, T., 2023. A benchmark for multi-modal lidar slam with ground truth in gnss-denied environments. Remote Sensing, 15(13), p.3314.

VIII Nevalainen, P.,**Qingqing, L.**, Melkas, T., Riekki, K., Westerlund, T. and Heikkonen, J., 2020. Navigation and mapping in forest environment using sparse point clouds. Remote Sensing, 12(24), p.4088.

The following related publications are not directly included in this thesis:

I **Qingqing, L.**, Queralta, J.P., Gia, T.N. and Westerlund, T., 2019, December. Offloading monocular visual odometry with edge computing: Optimizing image quality in multi-robot systems. In Proceedings of the 2019 5th International Conference on Systems, Control and Communications (pp. 22-26).

II Queralta, J.P., **Qingqing, L.**, Ferrer, E.C. and Westerlund, T., 2022. Secure encoded instruction graphs for end-to-end data validation in autonomous robots. IEEE Internet of Things Journal, 9(18), pp.18028-18040.

III Gia, T.N.,**Qingqing, L.**, Queralta, J.P., Zou, Z., Tenhunen, H. and Westerlund, T., 2019, September. Edge AI in smart farming IoT: CNNs at the edge and fog computing with LoRa. In 2019 IEEE AFRICON (pp. 1-6). IEEE.

IV Queralta, J.P.,**Qingqing, L.**, Zou, Z. and Westerlund, T., 2020, April. Enhancing autonomy with blockchain and multi-access edge computing in distributed robotic systems. In 2020 Fifth International Conference on Fog and Mobile Edge Computing (FMEC) (pp. 180-187). IEEE.

V Queralta, J.P.,**Qingqing, L.**, Schiano, F. and Westerlund, T., 2022, July. VIO-UWB-based collaborative localization and dense scene reconstruction within heterogeneous multi-robot systems. In 2022 International Conference on Advanced Robotics and Mechatronics (ICARM) (pp. 87-94). IEEE.

VI Queralta, J.P., **Qingqing, L.**, Gia, T.N., Truong, H.L. and Westerlund, T., 2020, May. End-to-end design for self-reconfigurable heterogeneous robotic swarms. In 2020 16th International Conference on Distributed Computing in Sensor Systems (DCOSS) (pp. 281-287). IEEE.

VII Sarker, V.K., **Qingqing, L.** and Westerlund, T., 2020, June. 3D Perception with Low-cost 2D LIDAR and Edge Computing for Enhanced Obstacle Detection. In 2020 IEEE Conference on Industrial Cyberphysical Systems (ICPS) (Vol. 1, pp. 49-54). IEEE.

VIII Xianjia, Y., **Qingqing, L.**, Queralta, J.P., Heikkonen, J. and Westerlund, T., 2021, June. Applications of uwb networks and positioning to autonomous robots and industrial systems. In 2021 10th Mediterranean Conference on Embedded Computing (MECO) (pp. 1-6). IEEE.

IX Xianjia, Y., **Qingqing, L.**, Queralta, J.P., Heikkonen, J. and Westerlund, T., 2021, August. Cooperative uwb-based localization for outdoors positioning and navigation of uavs aided by ground robots. In 2021 IEEE International Conference on Autonomous Systems (ICAS) (pp. 1-5). IEEE.

X Zhao, W., Queralta, J.P.,**Qingqing, L.** and Westerlund, T., 2020, November. Towards closing the sim-to-real gap in collaborative multi-robot deep reinforcement learning. In 2020 5th International conference on robotics and automation engineering (ICRAE) (pp. 7-12). IEEE.

1 Introduction

Over the past few decades, the landscape of autonomous systems has undergone a profound evolution, propelled by significant advancements in sensor technologies. Among these, LiDAR sensors have emerged as crucial components in the realm of autonomous systems, with applications spanning from household vacuum cleaning robots [1] to state-of-the-art self-driving platforms [2]. Their ubiquitous integration in these systems can be credited to several pivotal factors, notably their exceptional long-range sensing abilities, precise object detection accuracy, and reliable performance across diverse environmental conditions [3].

LiDAR technology has seen significant advancements over the last decade, leading to improvements in density, accuracy, and higher frequency of measurements. The concept of integrating multiple 2D scanners along the Z-axis to attain 3D perception has inspired the development and widespread adoption of multiple channels spinning LiDAR [4]. During the 2007 DARPA Urban Challenge, five out of the six automated vehicles that successfully completed the race were equipped with the Velodyne 64 channels spinning LiDAR [5]. In the context of 3D LiDAR systems, the term channel refers to the individual laser beams or detection elements used to capture data points in the environment. Each channel corresponds to a specific laser beam or detector that emits or receives light pulses [6]. Nowadays, the 3D spinning LiDAR systems offer a range of channels, typically ranging from 16 to 32, and even up to 256 channels. For example, the 128 channels spinning LiDAR OS-0 in our data acquisition system shows in Figure 1 can generate 2.6 million points per second, with a maximum range of 50 meters. These advancements in multi-channel spinning LiDAR have enabled high degrees of situational awareness in mobile robotic applications. This innovative technology has garnered considerable interest across diverse fields, proving valuable in practical applications such as autonomous driving vehicles [7], unmanned aerial vehicles [8; 9], and forest inventory [10].

1.1 Research Background

The advancements in 3D LiDAR technology have become essential tool for fundamental perception tasks such as odometry, global localization, mapping, and tracking. In this section, we introduce the primary tasks addressed in this thesis.

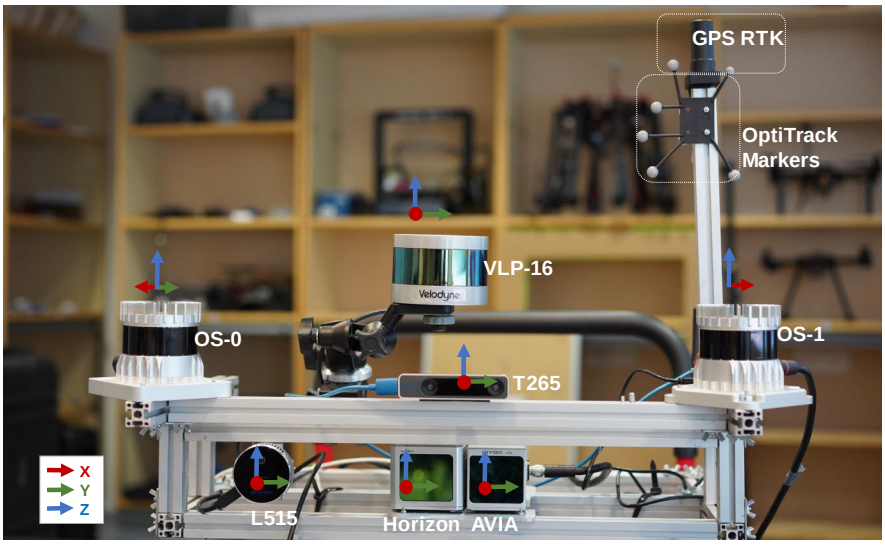


Figure 1. Front view of the proposed multi-modal multi-LiDAR data acquisition system. The setup comprises three spinning LiDARs: a 16-channel Velodyne LiDAR (VLP-16), a 64-channel Ouster LiDAR (OS1), and a 128-channel Ouster LiDAR (OS0), along with two solid-state LiDARs, Horizon and Avia.

1.1.1 LiDAR-Based Odometry

LiDAR odometry is a technique that utilizes LiDAR sensor data to estimate the movement and position of a mobile robot or vehicle in real-time.LiDAR odometry generally employs scan-matching techniques including ICP, KISS-ICP [11], GICP [12], and others to determine the relative transformation between two successive frames. Feature-based matching approaches have gained popularity as a computationally efficient alternative to full point cloud matching. For example, in [13], Zhang et al. propose the registration of edge and plane features for real-time LiDAR odometry. This type of operation assumes that the LiDAR moves within a structured environment, with edge and plane points clearly identifiable from the point clouds. The matching of consecutive scans is then performed by solving a least-squares optimization problem. Due to the high performance achieved by the proposed method, several iterations of LOAM have emerged. For instance, LeGO-LOAM [14] optimizes ground points separately, resulting in better odometry accuracy and computational efficiency.

Low-cost and high-performance solid-state LiDAR have attracted significant interest among researchers in both academia and industry in recent years. Compared to spinning LiDAR, the different sensing characteristics of solid-state LiDAR, such as non-repetitive scanning patterns, introduce new challenges in point cloud registration and mapping. In [15], Lin et al. address several fundamental challenges with

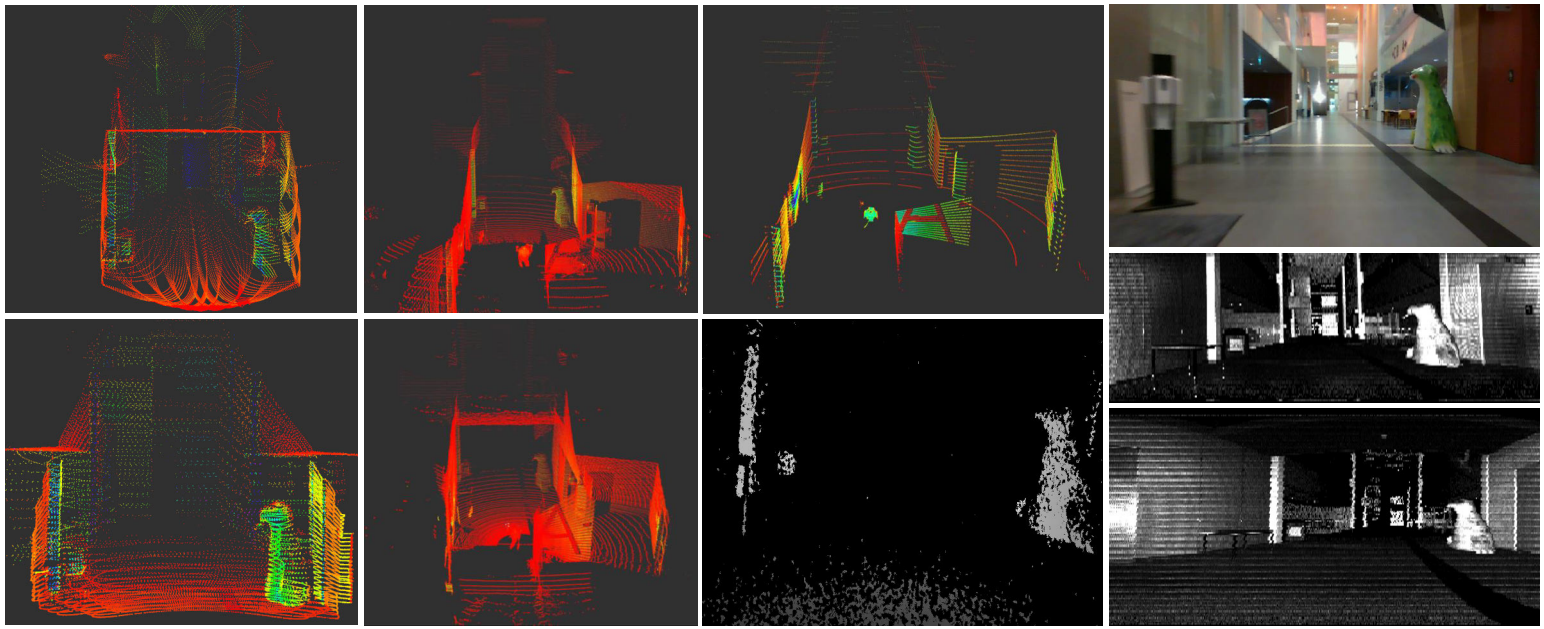


Figure 2. LiDAR pointcloud and camera data samples from the proposed dataset in the *Indoor04* sequence are visualized here. The leftmost column shows the LiDAR data from Avia and Horizon; the second column shows the LiDAR data from OS1 and OS0; the third column shows the data from the VLP-16 and the depth image from L515. The rightmost column shows the RGB image from L515 and the range images from OS1 and OS0.

Table 1. Sensor specification for the data collection platform. Angular resolution is configurable in the OS1-64 (varying the vertical FoV). Livox LiDARs have a non-repetitive scan pattern that delivers higher angular resolution with longer integration times. For LiDARs, range is based on manufacturer information, with values corresponding to 80% Lambertian reflectivity and 100 klx sunlight, except for the L515 LiDAR camera.

Sensor	IMU	Type	Channels	FoV	Resolution	Range	Freq.	Points (pts/s)
VLP-16	N/A	spinning	16	360°×30°	V:2.0°, H:0.4°	100 m	10 Hz	300,000
OS1-64	ICM-20948	spinning	64	360°×45°	V:0.7°, H:0.18°	120 m	10 Hz	1,310,720
OS0-128	ICM-20948	spinning	128	360°×90°	V:0.7°, H:0.18°	50 m	10 Hz	2,621,440
Horizon	BS-BMI088	solid-state	N/A	81.7°×25.1°	N/A	260 m	10 Hz	240,000
Avia	BS-BMI088	solid-state	N/A	70.4°×77.2°	N/A	450 m	10 Hz	240,000
L515	BS-BMI085	LiDAR camera	N/A	70°×43°(±3°)	N/A	9 m	30 Hz	-
T265	BS-BMI055	fisheye cameras	N/A	163±5°	N/A	N/A	30 Hz	-

a robust, real-time LiDAR odometry and mapping algorithm for solid-state LiDAR (LOAM Livox). The proposed method addresses the challenges of reduced field of view (FoV) and non-repetitive sampling patterns by focusing on feature extraction and selection, robust outlier rejection, filtering of moving objects, and compensation for motion distortion. Other recent results have also presented tightly-coupled LiDAR-inertial odometry and mapping schemes for both solid-state and mechanical LiDARs [16; 17]. Regarding the inherently limited FoV of a single solid-state LiDAR, a decentralized approach for simultaneous calibration, localization, and mapping utilizing multiple solid-state LiDARs was introduced in [18] to enhance system resilience. However, increasing the number of LiDARs can expand the field of view (FoV), but it also leads to higher costs and increased system complexity. The solid-state-based odometry and mapping algorithm has made significant progress. However, its limited field of view (FoV) hinders its application in cluttered environments where the entire FoV might be blocked. Therefore, more robust sensor fusion methods need to be investigated to ensure its robust performance in diverse environments.

1.1.2 Global localization

Global localization in robotics refers to the process of accurately estimating a robot's position within a global coordinate system or a given map. This determination is vital for enabling effective mapping, precise navigation, and facilitating cooperative tasks among multiple robots in diverse robotic applications. Accurate global localization also plays a pivotal role in Loop Closure Detection (LCD) within the SLAM framework, crucial for recognizing revisited locations and mitigating pose estimation drift

through graph optimization [19; 20; 21]. Extensive studies have addressed global localization and Loop Closure Detection in structured environments [22; 23; 24; 25]. However, research focusing on structure-poor environments like forests or orchards is limited [26; 27], mainly due to the complexity, scarcity of landmarks, and the large scale of these environments. Robust and efficient global localization methodologies tailored to unstructured environments remain an under-explored area of study.

1.1.3 Multi-Sensor Fusion for SLAM

Relying solely on LiDAR for pose estimation often results in substantial drift due to registering skewed point clouds. Modern LiDAR-based SLAM systems frequently integrate data from multiple sensors, such as IMU, GNSS, and cameras, to bolster accuracy, robustness, and situational awareness.

IMU sensors play a crucial role in modern SLAM framework by providing essential motion and orientation information at high frequency, leading to the fusion of LiDAR and IMU, which has received a lot attention among researchers and is widely employed in various SLAM algorithms [21; 17]. LIO-SAM was introduced as a solution to mitigate accumulated drift in LiDAR-inertial odometry using a graph optimization strategy. This method tightly couples LiDAR and the inertial measurement unit (IMU), optionally incorporating GNSS sensors, and utilizes smoothing and mapping techniques [21]. Other studies, such as LIO-Mapping [28] and Fast-LIO implementing an iterated Kalman filter [17], also leverage LiDAR and IMU fusion for SLAM. Furthermore, integrating LiDAR-inertial odometry with visual-inertial odometry (VIO) significantly enhances accuracy and robustness, especially in texture-less or feature-less environments [29; 30].

Integrating more sensors into the system provides a wealth of environmental information. However, this expansion increases system complexity, necessitating robust calibration and maintenance, as well as more powerful computers to process the combined data. These approach may not be viable for computing resource-limited platforms, such as micro UAVs.

1.1.4 SLAM with Solid-State-LiDAR

One of the key limitations of LiDAR technology preventing more widespread adoption for localization and mapping in mobile robots is the high cost of the sensors, specially compared to vision sensors. However, with lower-cost models becoming available, mainly solid-state LiDARs, multiple research efforts have been directed towards optimizing existing algorithms for the new sensing modalities and scanning patterns.

A robust, real-time LOAM algorithm for solid-state-LiDAR with small FoV and irregular samplings has been presented in [31] to address several fundamental chal-

lenges arising from solid-state-LiDARs. In another work, Lin *et al.* proposed a decentralized framework for SLAM tasks with multiple solid-state-LiDARs to increase the FoV and improve overall system robustness [18]. Inspired by local Bundle Adjustment (BA) techniques utilized in visual SLAM, a BA approach with an adaptive voxelization method to search feature correspondence and solve the problem of sparse features points in three-dimensional LiDAR data was presented in [32]. More recent works have also worked towards improving LiDAR-based SLAM system robustness, with tightly-coupled LiDAR-inertial odometry and mapping schemes for both solid-state and mechanical LiDARs presented in [16; 33]. Additionally, a camera and solid-state LiDAR fusion SLAM framework have also been proposed in [34].

1.1.5 Multi-LiDAR-Based Sensor Fusion

More recently, the research focus has also shifted towards the fusion of data from multiple LiDARs. For example, [35] addresses multi-LiDAR online extrinsic calibration, odometry, and mapping, where extracted edge and planar features are utilized and data uncertainty is modeled with Gaussian distribution. A tightly coupled LiDAR-inertial odometry and mapping approach with low drift is proposed in [36], which utilizes features extracted from multiple time-synchronized LiDARs with complementary FoV. Rather than utilizing spinning LiDARs, a decentralized extended Kalman filter (EKF) approach for simultaneous calibration, localization, and mapping for multiple solid-state LiDARs was introduced to improve upon the limited FoV of a single solid-state LiDAR [18].

Achieving denser 3D geometry measurements of the surrounding environment is crucial for enhancing 3D environment understanding. Unfortunately, spinning LiDAR with higher resolution can be costly due to its more complicated architecture. Low-resolution spinning LiDAR, while more affordable, produces sparse point clouds with limited features, making the problem difficult to tackle [37] and leading to inevitable inherent alignment errors [38]. Despite the clear advantages of solid-state LiDARs, the naturally narrow horizontal FoV leads, in a similar way to monocular pinhole camera systems, to the sensing volume being blocked by objects, or even entirely occupied by a near wall, resulting in an insufficient number of feature points to estimate a 6-degree-of-freedom (6-DoF) pose. This limitation has made current solid-state-based SLAM methods challenging to apply in outdoor or large indoor scenarios [16; 39] as shown in Figure 3. However, we find a lack of work in the literature in terms of fusing multiple LiDARs with different scanning modalities, which has potential to improve map quality and odometry accuracy by leveraging the advantages of different model of LiDAR sensors.

Multiple studies have focused on improving LiDAR maps by integrating point clouds from multiple LiDAR sensors [35; 40]. However, the low frame publishing frequency of typical LiDARs (e.g., 10 Hz) can hinder an accurate 6-DOF pose

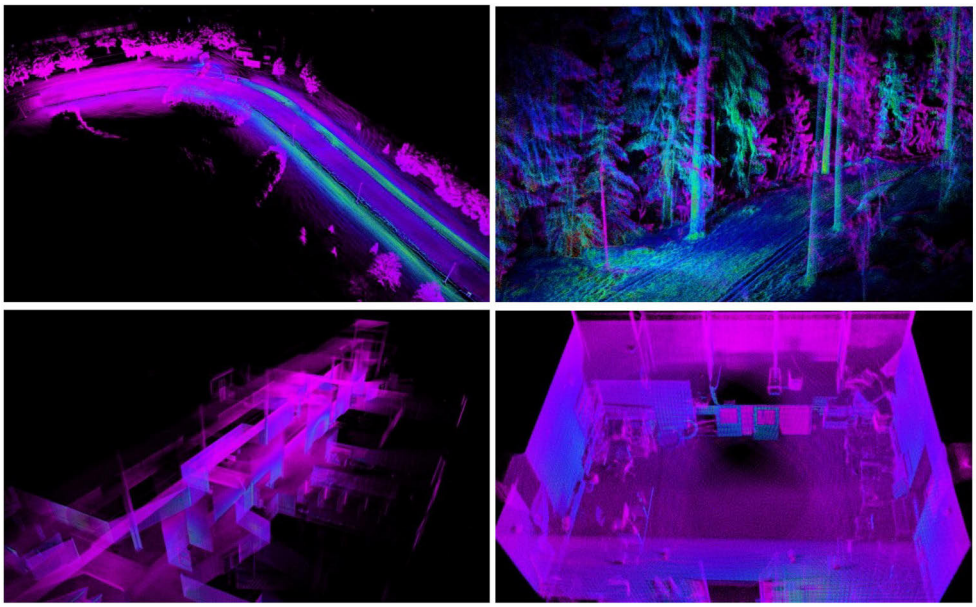


Figure 3. The dense mapping results with solid-state LiDAR by LIO-Mapping are showcased from different environments, positioned from top to bottom, left to right, depicting urban road, forest, hall, and office room environments, respectively.

estimation in multi-LiDAR systems. In contrast, IMUs have been widely used in state-of-the-art SLAM systems [41; 16], due to their ability to measure acceleration and angular velocity at a high frequency (e.g., 200 Hz) in three-dimensional space. Nonetheless, there remains a lack of methods that can effectively exploit multi-LiDAR inertial systems for odometry estimations.

1.2 Motivation and Objectives

LiDAR technology has witnessed significant advancements, especially with the emergence of higher-resolution spinning LiDAR and low-cost solid-state LiDAR. However, achieving a fully affordable, reliable, and precise localization, mapping, and object tracking system using LiDAR technology still presents challenges. This thesis aims to address these issues and unlock the technology's full potential for practical applications across various industries.

The following are the key research questions that this thesis aims to answer throughout the different chapters, ranging from more general to more specific concept:

1. Datasets are crucial for SLAM algorithm development as they provide real-world sensor measurements and ground truth information, enabling the train-

ing, validation, and improvement of SLAM algorithms, which are essential for accurate and reliable spatial mapping and navigation of autonomous systems. Among existed SLAM dataset, the KITTI benchmark [42] is the most significant dataset with capabilities of evaluating several tasks including odometry, SLAM, objects detection, tracking with spinning LiDAR. Solid-state LiDAR overcome some of the challenges of spinning LiDAR in terms of cost and resolution, but introduce some new limitations in terms of a relatively small field of view (FoV) [31; 16]. Despite solid-state LiDAR's increasing popularity, few works have benchmarked the performance of both spinning LiDAR and solid-state LiDAR in diverse environments, which limits the development of more general-purpose LiDAR-based SLAM algorithms [43]. In this thesis, we explored the limitations and advantages of each LiDAR sensor in various environmental conditions. We aim to answer the following questions: How does the mapping ability between different LiDAR sensors at the same time in diverse environments? What kind of sensor combination of these sensors is able to achieve a more robust and accurate mapping system?

2. In the context of autonomous driving vehicles, accurate mapping and localization are paramount, particularly in dense urban environments where precise localization is indispensable [44]. The dynamic surroundings and narrower pathways in such areas pose significant technical challenges [45]. In this thesis, we aim to address the following questions: Which LiDAR-based localization method is optimal for enhancing the navigation accuracy of delivery robots in dense urban environments? Additionally, what strategies can be employed to effectively update or repair maps in response to changes in the environment or map degradation in dense urban settings?

3. Most SLAM algorithms are primarily optimized for structured urban environments, leading to reduced performance in unstructured environments such as forests. Firstly, in forests, GNSS signals often suffer from the multipath effect, resulting in poor location accuracy, particularly under the canopy [46]. Secondly, due to environmental similarities, general point cloud ICP matching algorithms may encounter issues with local minima during loop closure detection tasks, leading to an inability to correct odometry drift [13]. Moreover, in forest harvesting operations, the movement of harvesters and forwarders along undefined paths, coupled with potential changes in the environment due to logging activities, further complicates the localization process. In this thesis, we aim to address the localization challenges in GNSS-denied environments and develop solutions for accurate localization in forest environment.

4. Multiple studies have focused on improving LiDAR maps in terms of accuracy and density by integrating point clouds from multiple LiDAR sensors [35; 40].

However, the low frame publishing frequency of typical spinning LiDARs (e.g., 10 Hz) can hinder an accurate 6-DOF pose estimation in multi-LiDAR systems. In contrast, IMUs have been widely used in state-of-the-art SLAM systems [41; 16], due to their ability to measure acceleration and angular velocity at a high frequency (e.g., 200 Hz) in three-dimensional space. Nonetheless, there remains a lack of methods that can effectively exploit multi-LiDAR inertial systems for odometry estimations. In this thesis, after studying multiple LiDAR systems and their limitations and advantages, our aim is to answer the following question: What is the most affordable, robust, and accurate mapping system with different sensor systems? Additionally, how can sensor fusion with different modalities of LiDARs and IMUs systems be implemented to achieve state-of-the-art mapping performance?

5. Most SLAM methods operate under the assumption that the environment is static. However, the presence of dynamic objects in the real world can significantly impact the accuracy of scan matching [47]. To counteract the influence of such dynamic objects, many researchers have concentrated on identifying and eliminating feature points originating from moving objects, particularly in autonomous driving systems involving humans and cars [48]. While considerable attention has been devoted to detecting large moving objects, such as vehicles and pedestrians, there has been relatively little emphasis on micro object detection. One primary reason for this neglect is the limited capability of traditional spinning LiDARs to capture sufficient points from small objects, posing challenges in tracking them effectively. However, the utilization of solid-state LiDARs with non-repetitive scanning patterns enables the generation of dense point clouds within the field of view (FoV), facilitating the tracking of smaller objects like micro objects. In this thesis, our goal is to further explore the tracking capabilities of solid-state LiDAR, specifically focusing on its ability to track smaller objects.At last, we aim to answer the following questions: How can small objects be effectively detected from the environment? What is the detection ability using a camera with deep learning methods from a UAV view at different attitudes? How can challenging objects be tracked using LiDAR sensors with small field of view (FoV) and non-repetitive scanning features? Answering these questions can contribute to the development of a more robust SLAM system.

1.3 Main Contributions

The thesis concentrates on the development of a cost-effective, robust 3D LiDAR-based localization, mapping, and tracking system. This system leverages multimodal sensors, encompassing various types of LiDARs, IMUs, and GNSS, particularly in

challenging environments.

The contribution can be categorized as following:

- An open-source multi-modal LiDAR dataset for general-purpose localization and mapping. The dataset includes a benchmark that compares different modality LiDARs (spinning, solid-state) in diverse environments, like indoor offices, long corridors, halls, forests, and open roads. To allow for more accurate and fair comparison, we introduce a new method for ground truth generation in larger indoor spaces. This enhanced ground truth enables significantly higher degree of quantitative benchmarking and comparison with respect to our previous work [43]. The dataset and ground truth labels, as well as more detailed data, provides a performance reference for multi-modal LiDAR sensors in both structured and unstructured environments to both academia and industry. The datasets are fully open sourced at [49] and [50].

- A segmentation and graph search-based global localization method for forest environments. Forest enviornments are characterized by limited distinctive features and visual similarity across the surroundings, specifically targeting autonomous harvesters. The method achieved a robust, efficient, and accurate localization ability in forest environment.

- LiDAR-based sensor fusion algorithms for autonomous delivery vehicles in GNSS-denied and corrupted map environments. By integrating data from LiDAR sensors with other complementary sensors such as cameras, IMUs, and odometry, the proposed method enhance localization accuracy, perception capabilities, and dynamic path planning. The LiDAR-based sensor fusion algorithms, which won the first award in the US area and achieved the second position globally in the JDD competition, effectively address the last mile delivery problem in GNSS-denied and corrupted map environments for autonomous vehicles.

- A cost-effective and robust simultaneous localization and dense mapping system that take advantage of different modalities of LiDAR sensors. The objective was to achieve enhanced situational awareness and mapping accuracy by effectively fusing data from low-resolution spinning LiDAR and low-cost solid-state LiDAR with limited field of view (FoV). The research successfully tackled calibration, synchronization, data processing, and feature extraction challenges to seamlessly integrate diverse data sources.

- A novel method for object detection and tracking. The method leverages the dense measurement capability of solid-state LiDAR sensors. Additionally, the research delved into camera-based human detection and tracking using data-

driven based method at variable altitudes, specifically for search and rescue operations.

1.4 Thesis Content Summary

The thesis is organized as follows: The introduction chapter offers an overview of the research background and the addressed problem. Chapter 2 analyzes various sensor modalities and datasets while benchmarking SLAM algorithms across diverse environments. In Chapter 3, a graph search-based localization strategy for autonomous harvesters in dense forests is proposed. Chapter 4 focuses on accurate localization for autonomous delivery vehicles in urban environments with scenarios involving GNSS-denied and map-corrupted situations. Chapter 5 introduces a multi-modal multi-LiDAR inertial dense mapping framework that harnesses the benefits of wide FoV spinning LiDAR and high-resolution solid-state LiDAR. In Chapter 6, the research delves into dynamic object detection and tracking, specifically focusing on its application in search and rescue missions and multi-robot cooperation tasks. Finally, Chapter 7 summarizes research findings, presents conclusions, and outlines future research directions.

2 Multi-modal Multi-LiDAR Sensors Dataset

Datasets have had a significant impact on research efforts in the development of robust LiDAR odometry, localization, and mapping algorithms. By facilitating collaboration and comparison between different research groups, open-access datasets have accelerated algorithm development. Researchers can evaluate their algorithms on standardized datasets, enabling fair comparisons and benchmarking of different approaches. This common ground for testing and validation has led to the adoption of best practices and the sharing of knowledge within the research community, ultimately resulting in faster progress and innovation in the field.

One of the pioneers and perhaps the most significant dataset to date is arguably the KITTI benchmark[42]. The KITTI dataset includes a 64-beam 3D laser scanner, four gray-scale and color cameras, and a GNSS/IMU navigation system within a single data-gathering platform. The KITTI benchmark has become an essential tool to evaluate the performance of algorithms in multiple tasks such as odometry, SLAM, object detection, or tracking, among others, in both academia and industry. Several similar datasets have also been published with a system composed of multiple cameras and spinning LiDARs, providing images and the corresponding point clouds in urban environment. Some relevant examples include the Oxford Robocar dataset [51], nuScences [52], or the EU long-term dataset [53]. Multiple spinning LiDARs are often employed in these data collecting platform, albeit mostly sharing the same sensing modality or technology.

The aim of this chapter is to investigate state of the art LiDAR based odometry and mapping algorithms' performance in diverse environment on proposed datasets. One of the main limitations of existing datasets that we aim to overcome is the lack of availability of data from solid-state LiDARs. Solid-state LiDARs often present limited FoV, owing to the lack of mechanical rotation. The limited FoV together with the different scan patterns pose significant differences for many of the popular LiDAR odometry, localization and mapping algorithms [33]. As a result, we believe that more data is necessary to advance research towards general-purpose and sensor-agnostic LiDAR data processing algorithms. To bridge this gap, we present a novel multi-LiDAR dataset with spinning LiDARs of three different resolutions, two solid-state LiDARs with different FoV and scan pattern, and a LiDAR-camera.

The primary characteristics of the presented dataset are the following:

1. A dataset with data from five different LiDAR sensors and one LiDAR camera in a variety of environments. This is, to our knowledge, the most diverse dataset in terms of LiDAR sensors for these environments. The dataset includes spinning LiDARs with 16 (Velodyne VLP-16), 64 (Ouster OS1-64) and 128 (Ouster OS1-128) channels and different vertical FoVs. Two different solid-state LiDARs (Livox Horizon and Livox Avia) with different scanning patterns and FoVs are also included. A LiDAR camera (RealSense L515) provides RGB images and LiDAR-aided depth images. Low-resolution images with depth, near-infrared and laser reflectivity data from the Ouster sensors complete the dataset. These are illustrated in Figure 4.

2. The dataset includes sequences with MOCAP (motion capture) based ground truth in both indoors and outdoors environments. This is, to the best of our knowledge, the first LiDAR dataset to provide such accurate ground truth in forest environments in addition to indoor areas, albeit the limited trajectory length (see for samples Figure 5).

3. In addition to the MOCAP labeled data, the dataset includes other sequences in large indoor halls, roads, and forest paths. The wide variety of sensors enables comparison between LiDAR odometry and mapping algorithms to an extent that was not possible before, with both general-purpose and sensor-specific approaches.

4. Proposed a new multi-modal multi-LiDAR SLAM-assisted and ICP-based sensor fusion method for generating ground truth maps. With these maps, we then match real-time point cloud data using a normal distributions transform (NDT) method to obtain the ground truth with full six degrees of freedom (DOF) pose estimation. This novel ground truth data leverages high-resolution spinning and solid-state LiDARs.

5. Based on the presented dataset, we provide a baseline comparison of the state-of-the-art in LiDAR odometry, localization and mapping. We compare the odometry drift as well as the quality of the maps obtained with different sensors and different algorithms.

Based on the above characteristics of the presented dataset, this work provides a timely and complimentary addition to existing datasets which are mostly focused towards mobile robots indoors or autonomous cars outdoors. This chapter also show how the performance of the state-of-the-art LiDAR SLAM algorithms varies significantly based on the environment and the type of LiDAR sensor. The high degree of multi-modality in the sensor suite opens the door to research a variety of new challenges for the research community. This dataset can aid in the design and de-

velopment of future algorithms that better adapt to both structured and unstructured environments, to limited FoV of the sensors, and to different scanning modalities.

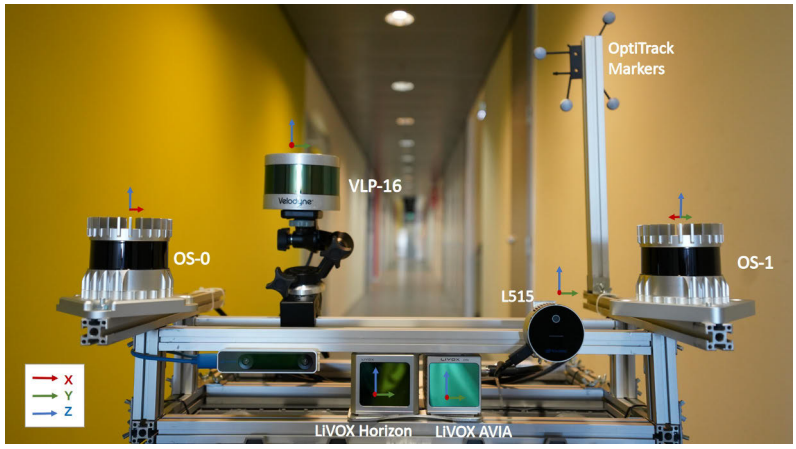


Figure 4. Front view of the multi-modal data acquisition system. Next to each sensor, we show the individual coordinate frames for the generated point clouds.

2.1 LiDAR-Based SLAM Datasets

There are a number of datasets available that are relevant to this work, mostly gathered within the autonomous driving community. In Table 3, we compare the most relevant related datasets from the literature with ours. In the rest of this section, we address the key differences between the proposed dataset and existing ones.

In addition to the myriad of datasets captured in urban road environments and focused towards research in autonomous driving, the literature also showcases efforts in off-road environments. For example, the NCLT dataset provides a large-scale indoor and outdoor dataset with multi-modal sensors, including spinning LiDARs, cameras and IMU attached on a wheeled robot [54]. In another work, a handheld device comprised of one spinning LiDAR and depth camera was utilized to collect data from urban outdoor and vegetated environments [55]. A multi-sensor SLAM benchmark, encompassing diverse indoor and outdoor environments, has been introduced in [56]. In relation to these works, we provide a wider variety of sensor data as well as more accurate ground truth in a selection of sequences.

There is also a number of datasets available in unstructured environments. For instance, the Robot Unstructured Ground Driving (RUGD) dataset captured from a small, unmanned mobile robot traversing in unstructured environments has been introduced in [57]. The RUGD dataset contains different terrain types focusing on visual perception tasks like semantic segmentation. Several similar datasets in unstructured environments have been presented for tasks such as scene depth prediction [58],

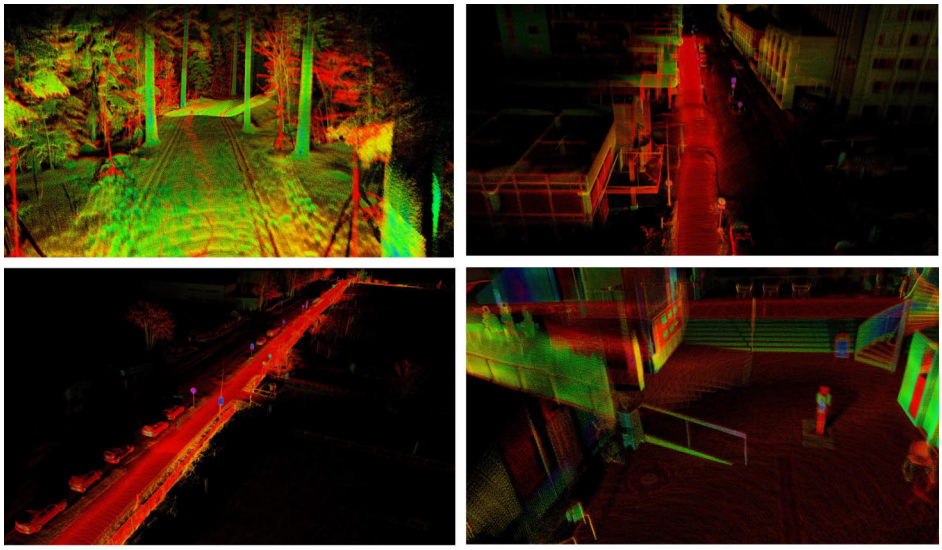


Figure 5. Samples of map data form different dataset sequences. From left to right and top to down, we display maps generated from a forest, an urban area, an open road and a large indoors hall, respectively.

terrain roughness understanding [59], off-road pedestrian detection [60]. Compared to these datasets, we provide MOCAP-based ground truth in a forest environment, while also including a wider variety of sensors.

In general, the number of publicly available datasets with solid-state LiDAR data is scarce. Among them, the PandaSet collects driving scenarios in urban environments with data from a forward-facing solid-state LiDAR and a 64-channels spinning LiDAR [61]. Additionally, Lin *et al.* presented an outdoor and indoor dataset with a solid-state LiDAR in college environment to test a novel LiDAR odometry and mapping (LOAM) algorithm tailored to solid-state LiDAR sensors [31]. In the present dataset, we provide a significantly higher number of sensors as well as ground truth both indoors and outdoors.

The research in the adaptation and tuning of algorithms for new LiDAR sensors lacks the support of a dataset for benchmarking and comparing the different approaches. Moreover, the lack of a truly heterogeneous and multi-modal dataset with various types of LiDAR sensors is preventing further comparisons between the methods to advance towards general-purpose LiDAR-based SLAM algorithms. To bridge these gaps, this chapter focus on providing a dataset that can serve as an initial benchmark for odometry, localization and mapping in diverse environments and with different types of LiDAR sensors.

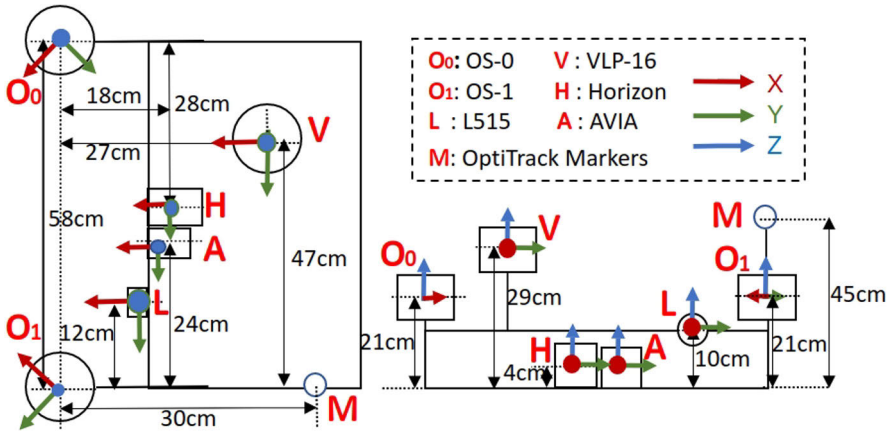


Figure 6. Our data collecting platform, top view (left) and front view (right)

2.2 Multi-LiDAR Data Collection Platform

The sensor configuration of our data collection platform is shown in Figure 4, and more specific information of each sensor is available in Table 1. Owing to the variety of environments where the platform has been used, it has been mounted on different types of mobile platforms. In road-like environments and large indoor halls, a Clearpath Husky mobile robot has been used. In forests outdoors with snow, it has been handheld. In small indoor spaces, it has been mounted on a mobile wheeled platform, manually pushed. In order to increase the usability of the dataset for benchmarking general-purpose algorithms, pitch and roll rotations have been applied in different configurations when handheld, in addition to standard horizontal settings where only the yaw angles varies if the surface where it is moving is horizontal.

2.2.1 Hardware System

The core objective of the sensor system is to provide data from various LiDAR sensors with different characteristics, from novel low-cost solid-state LiDAR to 3D spinning LiDARs with different resolutions and vertical FoV, and LiDAR cameras as well. To this end, our data collecting platform includes three spinning LiDARs: 16-channels Velodyne LiDAR (VLP-16), 64-channels Ouster LiDAR (OS1), and 128-channels Ouster LiDAR (OS0). On the side of solid-state LiDARs, two units from Livox are installed: Horizon, with a FoV close to a rectangle, and Avia, with an almost-circular FoV. An Intel RealSense L515 LiDAR camera completes the setup. Regarding the physical configuration, the Horizon and Avia LiDARs were installed in the center of the frame facing forward. The L515 camera was attached to the front left of the platform. On the sides, the OS0 and OS1 sensors were mounted at a bit higher level, where the OS1 is turned 45 degrees clockwise, and the OS0 is

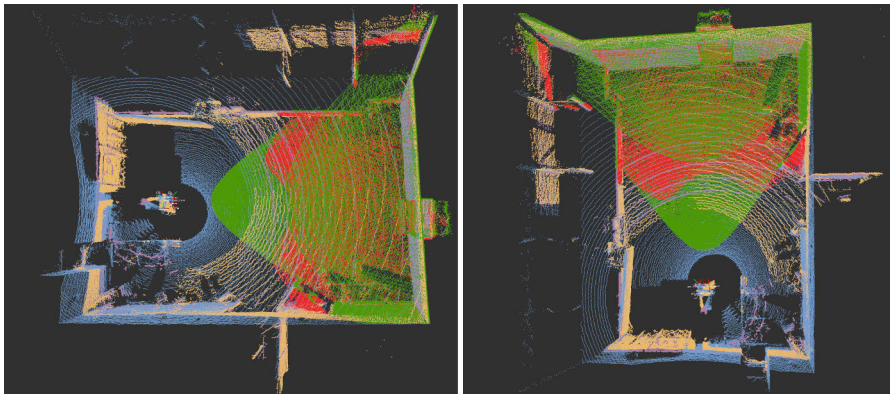


Figure 7. Top view of point cloud data generated for the calibration process with multiple LiDARs. The red and green point clouds represent data obtained from the Livox Horizon and Avia, respectively. The purple, yellow, and blue clouds are from the VLP-16, OS1, and OS0 sensors.

turned 45 degrees anticlockwise. The Velodyne LiDAR is at the top-most position with the x-axis facing forward as well. Please refer to the top view of Figure 6 for the detailed distances, positions and orientations. The Optitrack marker set for the MOCAP-based ground truth are fixed on the top of the aluminum stick to maximize its visibility and detection range.

To ensure a low-latency and high-speed transmission of all data, the LiDARs are connected to a Gigabit Ethernet router and a computer onboard the platform featuring an Intel i7-10750h processor, 64 GB of DDR4 RAM memory and 1 TB SSD storage. The Optitrack system is also physically connected via Ethernet to the onboard computer on a separate interface to the LiDARs. Finally, the RealSense L515 camera is connected using a USB 3.0 port.

2.2.2 Software System

Our software system is based entirely on ROS Melodic under Ubuntu 18.04. The set of ROS drivers and the publishing frequency of the different sensors is shown visually in Figure 8. Owing to the lack of hardware signals to synchronize the sensor data, as in other datasets in the literature [53], we approach the minimization of the data synchronization problem by running all the sensor drivers, data recording programs locally on a high performance computer. This, together with the networking equipment, aid in reducing the latency of data transmission at the hardware and software level (timestamped at the ROS drivers). In order to support a potentially wider use of the data, the dataset also includes the time stamp from built-in internal oscillators for both Livox and Ouster LiDARs, and for both point cloud and IMU data, in addition to the timestamp included in the header of all ROS messages. We have

Table 2. List of data sequences in proposed dataset (V: Velodyne VLP-16, H:Livox Horizon, A:Livox Avia, O_0: Ouster OS0, O_1: Ouster OS1, L:Realsense L515, T: Realsense T265, G: RTK/GNSS)

Sequence	Description	Ground Truth	Sensors
Forest01	Forest(Winter,Square)	Mocap	V,H,A,O_0,O_1
Forest02	Forest(Winter,Straight)	Mocap	V,H,A,O_0,O_1
Forest03	Forest (long path)	SLAM	V,H
Indoor01	Office room(easy)	Mocap	V,H,A,O_0,O_1,L
Indoor02	Office room(middle)	Mocap	V,H,A,O_0,O_1,L
Indoor03	Office room(hard)	Mocap	V,H,A,O_0,O_1,L
Indoor04	Large Hall	SLAM	V,H,A,O_0,O_1,L
Indoor05	Long Corridor	SLAM	V,H,A,O_0,O_1,L
Indoor06	Lab space (easy)	MOCAP	V,H,A,O_0,O_1,L
Indoor07	Lab space (hard)	MOCAP	V,H,A,O_0,O_1,L,T
Indoor08	Classroom space	SLAM+ICP	V,H,A,O_0,O_1,L,T
Indoor09	Corridor (short)	SLAM+ICP	V,H,A,O_0,O_1,L,T
Indoor10	Corridor (long)	SLAM+ICP	V,H,A,O_0,O_1,L,T
Indoor11	Hall (large)	SLAM+ICP	V,H,A,O_0,O_1,L,T
Road01	Open road(short)	SLAM	V,H,A,O_0,O_1
Road02	Open road(long)	SLAM	V,H,A,O_0,O_1
Road03	Open road	GNSS RTK	V,H,A,O_0,O_1,L,T,G

also compared the angular velocities of IMUs together with data from the MOCAP system to conclude that the latency of our system is less than 10 ms.

2.2.3 Sensors Calibration

The extrinsic parameters of the LiDARs are calculated based on optimization methods similar to those presented in [62]. We calculate the extrinsic parameters in an indoor office environment, while the sensor platform was stationary. The coordinate system of the Horizon LiDAR sensor is treated as the reference frame during the calibration process. Ten consecutive frames of point cloud data are integrated from the solid-state LiDARs to accumulate a higher degree of detail from the environment. The point cloud data from each different LiDAR is then transformed to the reference frame based on manual measurements of a set of features in the environment. Then, a Generalized Iterative Closest Point (GICP) method is employed to optimize the relative transformation between the reference frame and LiDARs iteratively [12]. For reference, in Figure 7 we show sample sensor data from one of the indoor environments after calibration. The ROS package $livox\_camera\_LiDAR\_calibration$ was utilized to calibrate the extrinsic parameter between the Horizon sensor and the

Table 3. Comparison of related open-access LiDAR-based datasets with the proposed dataset.

Dataset	Year	Environment	Ground Truth	LiDARs	Other
KITTI[42]	2013	Urban road	RTK_GPS/INS	3D-Velodyne HDL-64E @10 Hz	4× cameras , accel/gyro
NCLT[54]	2017	Urban Indoor Outdoor	GPS/INS	3D-Velodyne HDL-64E@10 Hz 2× 2D-Hokuyo @10/40 Hz	camera
Oxford RobotCar[51]	2017	Urban Road	GPS/INS	2× 2D-SICK @50 Hz 3D-SICK @12.5Hz	4 Camera; accel/gyro
RUGB Dataset[57]	2019	Unstructured outdoor	-	3D-Velodyne HDL-32E @10 Hz	GPS&IMU ; 3× cameras
nuScences[52]	2020	Urban Road	-	3D-32-Beams LiDAR @20 Hz	6x Camera (RGB);GPS&IMU; 5x Radar@13Hz
Newer College[55]	2020	Urban outdoor Vegetated	6DOF ICP	3D-Ouster-64 @10 Hz	D435i (Infrared); accel/gyro
PandaSet[61]	2021	Urban road	-	3D-Hesai-Pandar64 @10 Hz 3D solid-state LiDAR@10 Hz	6x Cameras. GNSS&IMU
M2DGR [56]	2022	Urban In/Outdoors	Laser 3D tracker RTK_GPS/INS	3D VLP-32C @10 Hz	3 Cameras. GNSS&IMU
Our Dataset	2022	Urban indoor Urban road Forest	6DOF MoCAP SLAM	3x 3D-Spinning LiDAR(16,64,128) @10 Hz 2x 3D-Solid-State-LiDAR @10 Hz LiDAR-Camera @30 Hz	2x accel/gyro @200 Hz 2x accel/gyro @100 Hz

L515 LiDAR camera. The intrinsic parameters of LiDARs and the LiDAR camera are given based on factory settings and manufacturer information. A specific rosbag containing raw data recorded in stationary settings at the room shown in Figure 7 is provided for end-user re-calibration and potential application of different methods.

2.3 Generating Ground Truth Data

Generating accurate ground truth data in complex environments is a challenging task, as has been identified in multiple existing datasets. Many vehicular benchmarks utilize the pose generated from GNSS/INS fusion method as ground truth. However, multi-path effect can affect the accuracy of the pose estimated by GNSS sensors in forest and urban environments. For indoor environments, GNSS signals are unavailable.

2.3.1 MOCAP Avaliable Ground Truth

MOCAP systems have been widely adopted in indoor environments owing to their ability to provide millimeter-level accuracy in positioning data. However, the utilization of MOCAP systems is limited mainly by the range of the cameras, usually in the 10 to 20 m range. The need for relatively complex setup of the system has also prevented the adoption of such systems for outdoors environments, and specially in unstructured environment such as forests.

To meet the demands of reliable ground-truth data for diverse environments,

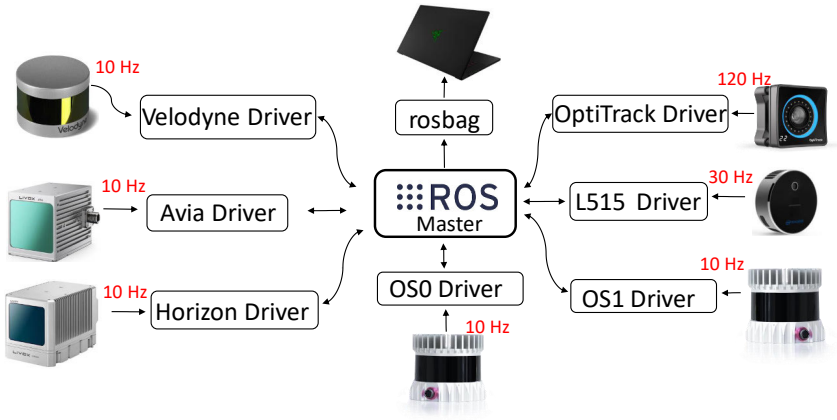


Figure 8. ROS drivers and data gathering frequency for the different LiDAR sensors used in data collection platform.

the present dataset includes MOCAP-based ground-truth data in both a subset of indoor and forest environments. This enables millimeter level pose estimation as ground truth for odometry algorithms in both structured and unstructured environments, which can aid in researching low-drift odometry algorithms, accurate feature tracking, and reduction of motion-induced distortions in the data. For large-scale environment, where the MOCAP system is unavailable, we also provide location information as a reference from SLAM methods [33]. In these settings, the higher-resolution LiDAR OS0-128 can be used as a baseline for the other sensors. We evaluate the SLAM algorithms in diverse environments, with a sample of environments shown in Figure 11. From the different SLAM methods further characterized in the next section, those that use data from the OS0 sensor showcase the most robust performance in a series of sampled sequences.

2.3.2 SLAM Assisted Ground Truth

To provide accurate ground truth for large-scale indoor and outdoor environments, where the MOCAP system is unavailable or GNSS/RTK positioning result becomes unreliable due to the multi-path effect, we propose a SLAM-assisted solid-state LiDAR-based ground map generation framework.

Inspired by the prior map generation methods in [55], where a survey-grade 3D imaging laser scanner Leica BLK360 scanner is utilized to obtain static pointclouds of the target environment, we employed a low-cost solid-state LiDAR Livox Avia and high resolution spinning LiDAR to collect undistorted pointclouds from environments. According to the Livox Avia datasheet, the range accuracy of the Avia sensor is 2 cm with a maximum detection range of 480 m. Due to the non-repetitive scanning pattern, the environment coverage of the pointcloud within the FoV increases with

time. Therefore, we integrated multiple frames when the platform was stationary to get more detailed undistorted environmental sampling. Each integrated pointcloud contains more than 240,000 points. The Livox built-in IMU is used to detect the stationary state of the platform when the acceleration values are smaller than 0.01 m/s^2 along all axes.

After gathering multiple undistorted pointcloud submaps from the target environment, the next step is to match and merge all submap into a global map by ICP. As the ICP process requires a good initial guess, we employ a high resolution spinning LiDAR OS0 with a 360-degree horizontal FOV to provide raw position by performing real-time SLAM algorithms. This process is outlined in Algorithm 1. A dense and high-definition ground truth map can be obtained by denoising the map generated by the algorithm described above to remove noise.

Let $\mathcal{P}_{sk}$ be the pointcloud produced by the spinning LiDAR, $\mathcal{P}_{dk}$ be the pointcloud generated by solid-state LiDAR, and $\mathcal{I}_k$ be the IMU data from the built-in IMU. Our previous work has shown high resolution spinning LiDAR has the most robust performance in diverse environments. Therefore, LeGo-LOAM [14] is performed with a high resolution spinning LiDAR (OS0-128) and outputs the estimated pose for each submap.

The cached data $\mathcal{S}_{cache}$ stores submaps and the related poses. Let $\mathcal{P}_i$ be the pointcloud and related pose $\mathbf{p}_i$ in $\mathcal{S}_{cache}[i]$. The submap $\mathcal{P}_i$ will be first transformed to map coordinate as $\mathcal{P}_i^m$ based on estimated pose $\mathbf{p}_i$; then GICP methods are employed on $\mathcal{P}_i^m$ to minimize the Euclidean distance between closest points against pointcloud $\mathcal{M}_{ap}$ iteratively; $\mathcal{P}_i^m$ will be transformed by the transformation matrix generated from GICP process, then merged to the map $\mathcal{M}_{ap}$. The result map $\mathcal{M}_{ap}$ is treated as ground truth map.

After the ground truth map generated, we employ normal NDT method in [63] to match the real-time pointcloud data from spinning LiDAR against the HD (High-definition) map as the Figure 9 shows to get the platform position in ground truth map. The matching result from the NDT localizer is treated as the ground truth.

The evaluation of the accuracy of the proposed ground truth prior map method is challenging for some scenes in the dataset, as both GNSS and MOCAP systems are not available in indoor environments such as long corridors. To evaluate the generated ground truth, we adhere to the methodological approach delineated in the referenced study [55]. We evaluated the standard deviations of the ground truth generated by the proposed method during the first ten seconds when the device is stationary from sequence *Indoors09*. The standard deviations along the X, Y, and Z axes are 2.2 cm, 4.1 cm, and 2.5 cm, respectively, or about 4.8 cm overall. However, evaluating localization performance when the device is in motion is more difficult. To better understand the order of magnitude of the accuracy, we compare the NDT-based ground truth Z values with the MOCAP-based ground truth Z values in the sequence *Indoor06* environment.

Algorithm 1: SLAM-assisted ICP-based prior map generation for ground truth data.

Input:

Spinning LiDAR pointcloud: $\mathcal{P}_{sk}$

Solid-state LiDAR pointcloud: $\mathcal{P}_{sk}$

IMU data: $\mathcal{I}_k$

Output:

Platform state: $\mathbf{p}_k$

Prior map: $\mathcal{M}_{ap}$

while *new* $\mathcal{P}_{sk}$ **do**

  $\mathbf{p}^k \leftarrow SLAM(\mathcal{P}_{sk})$;

`// Cached still clouds and raw pose`

$\mathcal{S}_{cache} = \{\}$;

`// Cached still cloud`

$\mathcal{P}_{cache} = []$;

while *new* $\mathcal{P}_{dk}$ **do**

  if $\mathcal{I}_k.V_{angular} < th_a$, $\boldsymbol{p}_k.V_{linear} < th_v$ **then**

    $s = True$;

    $\mathcal{P}_m = \mathcal{P}_m + \mathcal{P}_{dk}$;

  else

    $s = False$;

    $\mathcal{P}_{cache}.clear()$;

    $\mathcal{S}_{cache} \leftarrow (\mathcal{P}_m, \mathbf{p}_k)$;

while $\mathcal{S}_{cache}.size() > 0$ **do**

  $\mathcal{M}_{ap} \leftarrow ICP(\mathcal{S}_{cache}, \mathbf{p}_k, \mathcal{M}_{ap})$;

  $\mathcal{S}_{cache}.clear()$;

2.3.3 Data Sequences

The different subsets of proposed dataset are divided into three categories based on the environment: forest, indoor, urban outdoor. Table 2 lists all the sequences in our dataset.

Three sequences are provided for the forest environment. The forest data is collected at a forest in Turku, Finland $(60°28'14.3"N 22°19'54.8"E)$. The sequences Forest01 and Forest02 are collected in winter time with snow-covered ground. *Forest01* includes a square-shaped trajectory, while in *Forest02* the system is moved in a straight trajectory. Both of these sets include MOCAP data. A larger-scale forest recording is also provided in the *Forest03* sequence, with Horizon and VLP-16 LiDARs mounted on a smaller, handheld device. These sequences can support research in areas from tree-counting to tree stem diameter estimation. The vast difference in environment structure from urban settings to forest settings can also support LiDAR-based general-purpose odometry, localization and mapping algorithms.

The indoor environment then adds another dimension to the dataset with five data sequences. The data is collected in rooms and open halls of ICT-City in Turku, Finland. Three sequences are collected in a large experiment room where data from the MOCAP system is available. From these, *Indoor03* contains faster rotations and sudden movements, while positioning the sensors closer to objects in front and around. In consequence, most of the solid-state LiDAR view is blocked by objects or walls, presenting a significantly more challenging situation for odometry estimation algorithms based primarily on scan matching methods. The data in *Indoor01* is recorded while maintaining a longer distance ($\approx 50\,cm$) with objects and following a square-shaped trajectory with a reduced number of rotations. The *Indoor02* sequence then features a circular trajectory with more rotation but again maintaining an even larger distance to objects than in *Indoor03*. Sequences *Indoor04* and *Indoor05* correspond to recordings in a large hall and long corridor environment, respectively.

Finally, two sequences of open-road environment around the ICT-City building in Turku, Finland, are also included in this dataset. The length of *Road01* is over 50 m, while the traversed length of the trajectory in *Road02* is about 500 m.

2.3.4 Data Format

The data is collected in ROS and saved with the rosbag format, which has become a standard in the robotics research community. Sampled data frames from a subset of the sensors is shown in 2. The detailed data format for each type included in the dataset is listed as follow:

1. **Point cloud from spinning LiDARs** from the three spinning LiDARs, namely VLP-16, OS0-128 and OS1-64. The sensor message type from spinning LiDARs is recorded as $sensor\_msgs :: PointCloud$. Each point in the point

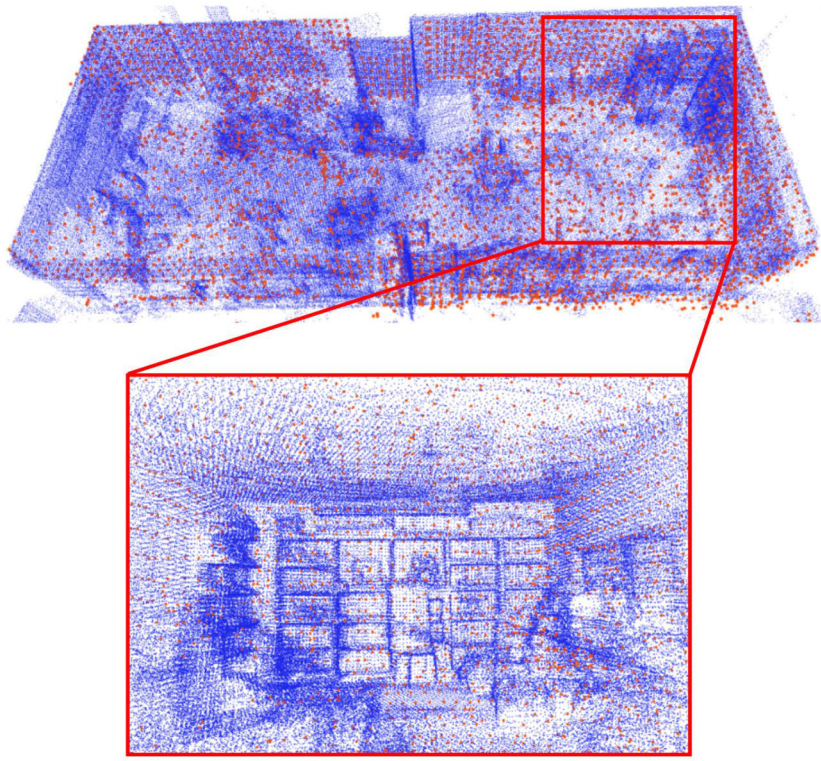


Figure 9. NDT localization with ground truth map. External view and Internal view when the current laser scan (orange) is aligned with the Ground truth map (blue).

cloud holds four values (x, y, z, I), where x, y, z represent the local Cartesian coordinates, and I is the laser reflectance of the point measured.

2. **Point cloud from solid-state LiDAR** from the two solid-state LiDARs, namely Avia and Horizon. The message type of these solid-state LiDARs in the rosbags is Livox's custom data format named $livox\_ros\_driver/CustomMsg$. The customized message keeps the first point's timestamp of each frame as the base time and then provides an offset time relative to the base time for each point. This is needed as the non-repetitive pattern does not allowed for a posteriori estimation, unlike the spinning LiDARs, in which we can estimate the time difference between points based on the settings of the mechanical parts. With this information, the de-skew process can then be conducted on the data to compensate for the distortion in the point cloud data caused by the sensor's egomotion [31]. We have maintained this message type that contains time information for each point for algorithms that include in the processing flow the de-skew of point cloud data and other related research. However, standard ROS messages simplify the visualization of the point cloud with tools such

as Rviz, and provide a format that many other LiDAR processing algorithms relying on standard ROS messages use [16]. Therefore, we provide format conversion tools to transform the Livox custom message data to the ROS standard message type $sensor\_msgs :: PointCloud$. Each point is then converted to a new one that holds five values (x, y, z, I, C), where x, y, z is the local Cartesian coordinate set, I is the intensity of the point, and where the integer part of C represents the line number and the decimal part the point timestamp.

3. **Images from LiDAR camera**. The RealSense L515 LiDAR camera is configured to publish RGB images with a size of 1920×1080, and depth images with a size of 1024×768. The message type is $sensor\_msgs :: Image$ at frequency 10 Hz. The depth estimations are aided by the built-in LiDAR sensor.

4. **Images from high-resolution spinning LiDAR**. The two high resolution LiDAR from Ouster, OS0-128 and OS1-64, can output fixed-resolution range images, near-infrared images captured by the laser sensor, and signal images. In these, each pixel represents the distance from the sensor origin to the point, the strength of the light captured, and the object's reflectivity, respectively. The images are published at frequency of 10 Hz. The image data is spatially correlated, with 16 bits per pixel and a linear photo response. The message type in the rosbags is the standard $sensor\_msgs/Image$.

5. **Inertial data from spinning and solid-state LiDARs**. There are in total four built-in 6-axis IMU sensors with 3-axis gyroscope and a 3-axis accelerometer, one in each of the Ouster and Livox LiDARs. They publish data at a frequency of 100 Hz in the former and 200 Hz in the latter. The data type of IMU data in the rosbags is again ROS' standard $sensor\_msgs :: Imu$.

6. **Ground truth data**. The ground truth data from the MOCAP system is included in rosbags as $geometry\_msgs :: PoseStamped$ messages. They are obtained from the computer driving the set of OptiTrack cameras through a VRPN (Virtual-Reality Peripheral Network) connection.

2.4 SOTA Algorithms Benchmarking

As a part of the dataset, we have evaluated several state-of-the-art SLAM algorithms on the different sequences. Through the rest of this section we discuss the best methods for different types of LiDAR data and environments.

Table 4. ATE (μ/σ) of selected SLAM methods (N/A when odometry estimations diverge). Best results for each sequence are in bold.

Sequence	FLIO_OS0	FLIO_OS1	FLIO_Avia	FLIO_Hori	LeGo_Velo	LIOL_Hori
Indoor01	**0.11 / 0.07**	0.12 / 0.04	0.58 / 0.3	0.65 / 0.24	0.22 / 0.19	N/A
Indoor02	**0.17 / 0.12**	0.34 / 0.21	0.70 / 0.20	N/A	0.48 / 0.17	N/A
Indoor03	**0.16 / 0.09**	0.21 / 0.08	N/A	N/A	0.38 / 0.23	N/A
Forest01	0.14 / 0.05	0.13 / 0.04	0.10 / 0.03	0.09 / 0.03	0.12 / 0.05	**0.04/ 0.01**
Forest02	0.13 / 0.07	0.12 / 0.06	0.09 / 0.03	0.11 / 0.05	0.31 / 0.05	**0.07/ 0.04**

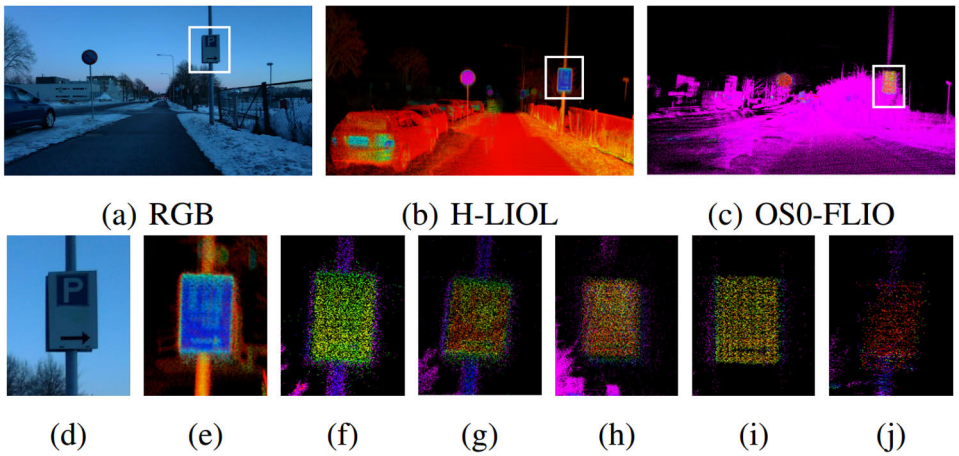


Figure 10. Qualitative comparison of the mapping quality using different LiDAR and SLAM algorithms. Bottom row shows in (e) to (j) the Horizon-based LIOL, Horizon, Avia, OS0, and OS1-based FLIO, and Velodyne's LeGo-LOAM maps, respectively.

2.4.1 LiDAR Odometry

Different LiDAR SLAM methods have been employed in our experiments. The FAST-LIO (FLIO) algorithm [33], a LiDAR-inertial odometry system that works for both spinning LiDAR and solid-state LiDAR, has been applied on Ouster LiDARs and Livox LiDARs leveraging the built-in IMUs. The objective here is to compare the performance of the same SLAM method applied to data from different types of LiDARs. In addition to FLIO, LeGo-LOAM[1] has been applied to data from the Velodyne LiDAR [14]. Another SLAM system, LIO-LIVOX (LIOL)[2], a tightly coupled SLAM that specifically developed for Horizon LiDAR, has also been tested on Horizon LiDAR data.

The estimated trajectories are visualized in Figure 11. The plots are two-dimensional

[1] https://github.com/RobustFieldAutonomyLab/LeGO-LOAM

[2] https://github.com/Livox-SDK/LIO-Livox

to improve readability as the changes in the vertical coordinate are minimal. Full data of the reconstructed paths is available in the dataset repository. From the results, one of the first conclusions is that the solid-state LiDAR-based SLAM system performs as well as or even better than the spinning LiDARs with the appropriate algorithms in the outdoors environments, but perform significantly more poorly in the indoor environments.

For the outdoor sequences, *Forest01*, *Forest02*, and *Road02*, all SLAM methods perform well, and the trajectories are completed without major disruptions. For the indoor sequence *Indoor01*, Avia- and Horizon-based FLIO are able to reconstruct the sensor trajectory but show that significant drift accumulates. In the same sequence, Horizon-based LIOL fails to reconstruct even the first loop in the trajectory. Similar behaviour is observed in the *Indoor02* sequence, with all the solid-state LiDARs failing completely in *Indoor03*. In all of these sequences, all the methods applied to spinning LiDARs perform satisfactorily. This result can be expected as they have full view of the environment, which has a clear geometry.

For the sequence *Indoor04* showcasing a long corridor, all the spinning LiDARs can again reconstruct a complete trajectory. The best performance is obtained from OS0-based FLIO and Velodyne-based LeGo-LOAM, with correct alignment between the first and last location. However, angular drifts accumulates with OS1-based FLIO, while Horizon- and Avia-based algorithms result in diverging odometry estimations.

In addition to the qualitative trajectory analysis, we also provide a quantitative analysis of the odometry error based on the MOCAP-based ground truth data in Table 4. Absolute trajectory errors (ATE) [64] are employed as an evaluation metric. All trajectories are transformed to the local coordinate reference of the MOCAP markers, and aligned with global ground truth data reference. Then, we calculate ATE using the EVO toolset [3]. Methods based on spinning LiDAR data clearly show performance indoors, with naturally lower error as the vertical resolution increases. However, in the forest environment solid-state LiDARs demonstrate superior performance, with LIOL featuring an ATE error as low as 4 cm mean error.

In summary, the above results show that spinning LiDARs are more stable across different environments, while the solid-state LiDARs show significantly better cost-performance ratio in some outdoors environments.

2.4.2 Mapping Quality

In the last part of our analysis, we compare the mapping quality generated from different LiDARs in urban open road environments as shown in Figure 10. From the figure, we can observe that the LIOL method applied to solid-state LiDAR presents

[3] https://github.com/MichaelGrupp/evo.git

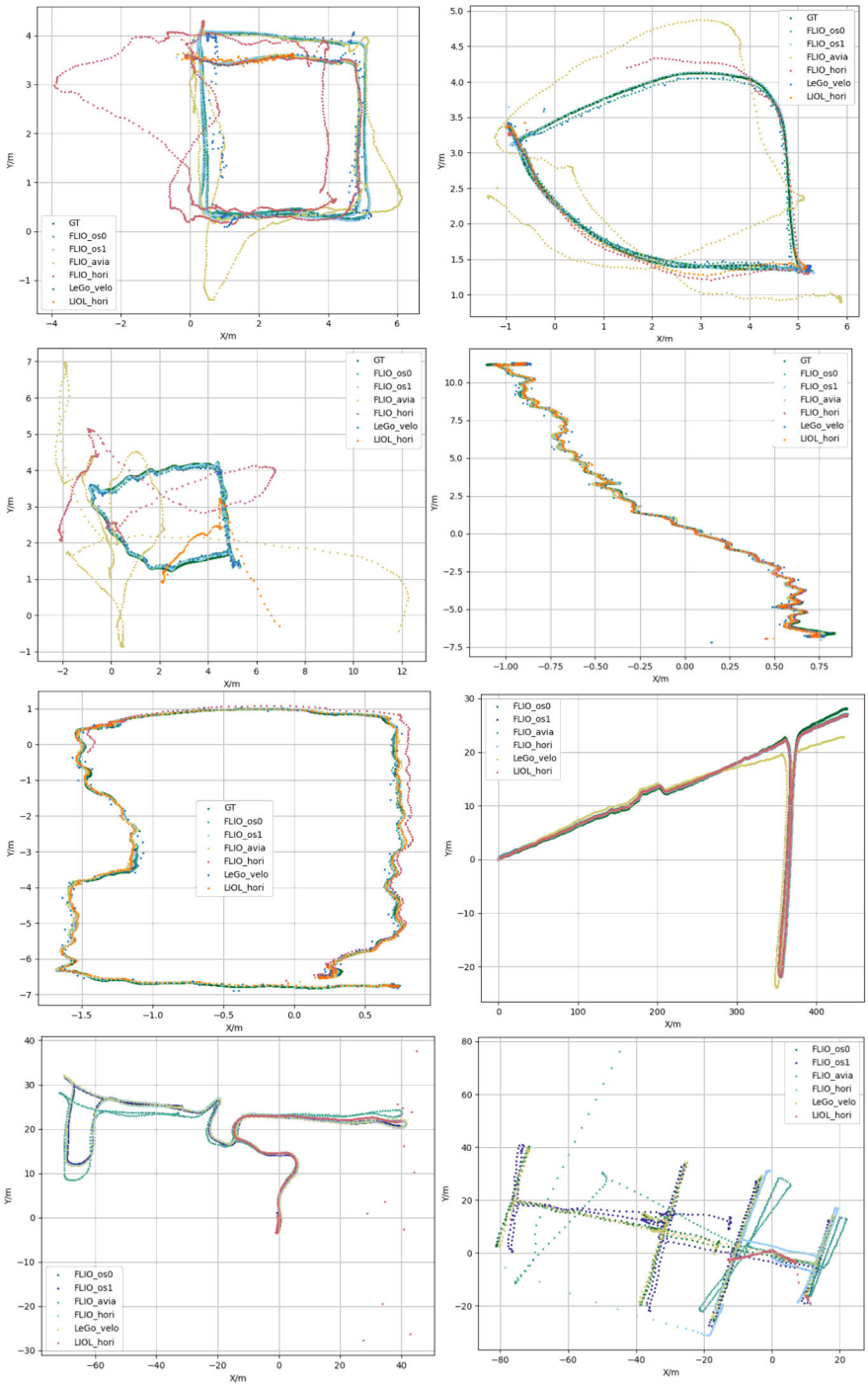


Figure 11. Estimated trajectory results using different LiDAR and SLAM algorithms. First row: *Indoor01*, *Indoor02*. Second row: *Indoor03*, *Forest02*. Third row: *Forest01*, *Road02*. Bottom row: *Indoor04*, *Indoor05*.

the most detailed and clear map. It is worth noting that these maps have been generated with default configuration of the methods and without changing parameters such as the map update frequency. This result matches the quantitative results obtained with the same sensors and algorihtms in the forest environment. More results are available in our project page.

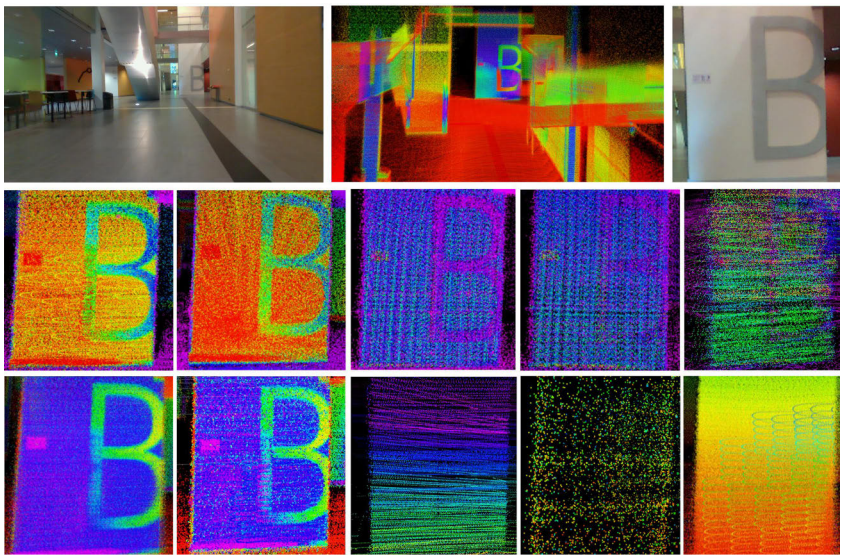


Figure 12. Qualitative comparison of the mapping quality using different LiDAR and SLAM algorithms. The first row from left to right shows RGB full view image, full view Horizon-based LIOL and close view RGB image. The second row from left to right shows OS0, OS1, Velodyne, Avia and Horizon-based FLIO. The bottom row from left to right shows the Horizon-based LIOL, Horizon, OS1-based LLOM and LLOMR, Velodyne's LeGo-LOAM maps and Horizon-based LVXM, respectively.

From Figure 10,we can observe that the LIOL method applied to solid-state LiDAR presents the most detailed and clear map. It is worth noting that these maps have been generated with the default configuration of the methods and without changing parameters such as the map update frequency. This result matches the quantitative results obtained with the same sensors and algorithms in the forest environment.

As shown in Figure 12, Horizon-based LIOL has the best mapping ability, but if the environment (such as sequence *indoors 06-09*) is complex, LIOL will fail to map due to drift. In addition, OS0 and OS1-Based FLIO also have good mapping ability, thanks to the wide FOV and excellent resolution of OS0 and OS1. Compared to OS0 and OS1, Velodyne has poorer mapping ability due to its lower vertical resolution, and it has almost failed to reconstruct the letter B sign in Figure 10. LVMX, LLOM, and LLOMR focus on calculating the mobile platform's pose estimation rather than point cloud mapping ability, so the point cloud maps they reconstructed are relatively poor.

2.4.3 Runtime Evaluation

We conducted this experiment on 4 different platforms. The first platform (1) was a Lenovo Legion Y7000P with 16 GB RAM, a 6-core Intel i5-9300H (2.40 GHz) and an Nvidia GTX 1660Ti (1536 CUDA cores, 6 GB VRAM). The second (2) platform was the Jetson Xavier AGX, a popular computing platform for mobile robots, which has an 8-core ARMv8.2 64-bit CPU (2.25 GHz), 16 GB RAM and 512-core Volta GPU. From its 7 power modes, we chose MAX and 30 W (6 core only) modes. The third (3) platform was the Nvidia Xavier NX which is a common embedded computing platform with a 6-core ARM v8.2 64-bit CPU, 8 GB RAM, and 384-core Volta GPU with 48 Tensor cores. We chose the 15 W power mode (all 6 cores) for the NX. The fourth (4) platform was the UP Xtreme board featuring an 8-core Intel i7-8665UE (1.70 GHz) and 16 GB RAM.

Table 5. Average run-time resource (CPU/RAM) utilization and performance (pose calculation speed) comparison of selected SLAM methods across multiple platforms. The data is played at 15 times the real speed for the pose publishing frequency. CPU utilization of 100% equals one full processor core.

	(CPU utilization (%), RAM utilization (MB), Pose publication rate (Hz))				
	Intel PC	**AGX MAX**	**AGX 30 W**	**UP Xtreme**	**NX 15 W**
FLIO_OS0	(79.4, 384.5, 74.0)	(40.9, 385.3, 13.6)	(55.1, 398.8, 13.2)	(90.9, 401.8, 47.3)	(53.7, 371.1, 14.3)
FLIO_OS1	(73.7, 437.4, 67.5)	(54.5, 397.5, 21.2)	(73.9, 409.2, 15.4)	(125.9, 416.2, 58.0)	(73.3, 360.4, 14.2)
FLIO_Velo	(69.9, 385.2, 98.6)	(44.4, 369.7, 29.1)	(58.3, 367.6, 21.4)	(110.5, 380.5, 89.6)	(57, 331.5, 19.5)
FLIO_Avia	(65.0, 423.8, 98.3)	(40.8, 391.5, 32.3)	(47.4, 413.4, 24.5)	(113.2, 401.2, 90.7)	(51.2, 344.8, 21.9)
FLIO_Hori	(65.7, 423.8, 103.7)	(37.6, 408.4, 34.7)	(50.5, 387.9, 26.8)	(109.7, 422.8, 91.0)	(47.5, 370.7, 23.4)
LLOM_Hori	(126.2, 461.6, 14.5)	(128.5, 545.4, 9.1)	(168.5, 658.5, 1.5)	(130.1, 461.1, 12.8)	(N / A)
LLOMR_OS1	(112.3, 281.5, 25.8)	(70.8, 282.3, 9.6)	(107.1, 272.2, 6.5)	(109.0, 253.5, 13.6)	(N / A)
LIOL_Hori	(186.1, 508.7, 19.1)	(247.2, 590.3, 9.6)	(188.1, 846.0, 4.1)	(298.2, 571.8, 14.0)	(239.0, 750.5, 4.54)
LVXM_Hori	(135.4, 713.7, 14.7)	(162.3, 619.0, 10.5)	(185.86, 555.81, 5.0)	(189.6, 610.4, 7.9)	(198.0, 456.7, 5.5)
LEGO_Velo	(28.7, 455.4, 9.8)	(42.4, 227.8, 7.0)	(62.8, 233.4, 3.5)	(39.7, 256.6, 9.1)	(36.9, 331.4, 3.7)

These platforms all run ROS Melodic on Ubuntu 18.04. The CPU and memory utilization is measured with a ROS resource monitor tool [4]. Additionally, for minimizing the difference of the operating environment, we unified the dependencies used in each SLAM system into same version, and each hyperparameter in the SLAM system is configured with the default values. The results are shown in Table 5.

The memory utilization of each selected SLAM approach among the two proces-

[4] https://github.com/alspitz/cpu_monitor

sor architectures platforms is roughly equivalent. However, the CPU utilization of the same SLAM algorithm running on Intel processors is generally higher than the other algorithms, and also the highest publishing frequency is obtained. LeGO_LOAM has the lowest CPU utilization but its accuracy is towards the low end (see Table 5), and has a very low pose publishing frequency. Fast-LIO performs well, especially on embedded computing platforms, with good accuracy, low resource utilization, and high pose publishing frequency. In contrast, LIO_LIVOX has the highest CPU utilization due to the computational complexity of the frame-to-model registration method applied to estimate the pose.

A final takeaway is in the generalization of the studied methods. Many state-of-the-art methods are only applicable to a single LiDAR modality. In addition, those that have higher flexibility (e.g., FLIO) still lack the ability to support a point cloud resulting from the fusion of both types of LiDARs.

2.5 Summary and Conclusions

This chapter presented a novel dataset collected with a multi-modal LiDAR sensor system in diverse environments. The dataset includes data from LiDARs of different types (spinning and solid-state), resolution (16, 64 and 128 channels for spinning LiDARs) and scan patterns (for two different solid-state LiDARs), in addition to a LiDAR camera. This opens the door to future research in general-purpose algorithms, as our analysis shows that different algorithms clearly perform better in one or another type of LiDAR, if they are able to process the data at all. There is therefore a significant gap to be filled in more robust LiDAR odometry, localization and mapping algorithms. To aid in analyzing the drift of the algorithms, we have provided ground truth data both indoors and in a forest environment. For comparison of the mapping quality mainly, we also provide data from larger indoor halls and urban roads. The experiments have covered nine sequences across four computing platforms.

Overall, we found that in both indoor and outdoor environments, the spinning LiDAR-based FLIO exhibited good performance with low power consumption, which we believe is due to the ability of the spinning LiDAR to obtain a full view of the environment. However, in the forest environment, the LIOL algorithm based on solid-state LiDAR It has the best accuracy and mapping quality performance, although it has the highest power consumption due to sliding window optimization. These findings led to the proposal of a novel multi-modal multi-LiDAR-Inertial SLAM framework in Chapter 5, leveraging the strengths of various LiDAR sensor modalities.

3 Accurate Localization in Large-scale Forest Environments

Accurate global localization on a given map is critical for autonomous systems as it ensures safe navigation and robust performance in various conditions by taking advantage of prior knowledge from the map. While LiDAR-based SLAM algorithms excel in simultaneously creating maps and estimating poses relative to their original position, there is an additional need to accurately locate the robot globally on an existing map for long-term navigation, mapping, and multi-robot cooperation tasks [65]. In structured environments like urban roads, where perception systems can detect abundant identifiable edge and plane features from ground structures, buildings, and traffic signs, global localization becomes more feasible. However, unstructured environments such as forests pose unique challenges, where irregular terrain and visually similar surroundings render conventional ICP-based algorithms ineffective in such settings [13; 12]. Moreover, GNSS accuracy is significantly hindered by the multipath effect occurring under forest canopies.

In this chapter, our objective is to enable fast, real-time global localization on prior map in forest environment for autonomous driving vehicle in forest, such as forest harvesters and forwarders. In a forest harvesting operation, harvesters and forwarders travel through undefined paths. Moreover, the detection of potential features (ground and tree stems) is considerately affected by irregularities (branches and foliage in the trees, and small plants, rocks or fallen trees on the ground). In this work, we pursue the design and development of a localization method which is reliable, accurate and of low computational cost. The computation is aimed to occur in real time on-board of harvesters and forwarders in the forest with canopy cover. The final hardware to be utilized in a real application would have similar capabilities to the hardware we have used in the experiments reported in this work. Relatively randomly located but ubiquitous tree stems are a natural basis for localization efforts. We also assume that a prior map of the forest in the form of a pointcloud is available before the mission starts.

In this chapter, we proposed a lightweight and graph-based localization method for pose estimation within a global map of a harvester in dense and unstructured forest environments. The proposed method is lightweight and matches a triangularization of a single LiDAR scan with a subgraph of the triangularization of the global forest map. Compared to matching raw LiDAR data point-by-point, we reduce the

problem to a triangularization created based on the positions of tree stems. In our experiment setting where a 16-channel 3D LiDAR with a field of view of about 210°and situated at a height of 1.5 m., individual trees get an average of 100 laser hits. The proposed triangularization process is able to reduce the number of points to be matched to approximately 1% of the size of a single LiDAR scan, significantly improving computation efficiency.To the best of our knowledge, this is the first work to utilize Delaunay triangulation for the purpose of localizing a vehicle in a forest environment. Moreover, we provide a fully experimental approach with a realistic use case in forest harvesting.

3.1 Autonomous Driving Meet Forest

3.1.1 Autonomous Vehicles in Forest

Over the last ten years, autonomous harvesting and transportation have become the long-term goal of the future development of the forest industry and attracted the interest of the research community. Naturally, there are several intermediate goals such as the organization of the public data [66], and the gradual increase of autonomy in varying degrees, starting with short-range transport in forests [67]. Incremental advances in forest autonomy include driver assistance platforms and function-specific automation. For example, these include the automated selection of tree stems to be processed, micro-tasks such as sequencing the processing of individual trees, local route planning, or semi-autonomous transportation and quality assurance. In all these tasks, one key element is the availability of tree maps, together with methods enabling the identification and selection of individual trees. In addition, local route planning requires accurate updates on the position of the harvester inside the forest in real-time, which can not be relied on global navigation satellite system (GNSS) sensors [68; 69]. An autonomous path tracking based on GNSS only was demonstrated in [70] using an industrial vehicle in a scene with clear-cut forest. To actually operate autonomously, and to locate and collect logs, more complex environments need to be analyzed. A summary of autonomous robots in cross-country terrain is [71], which includes also military applications. There seems to be not many attempts to have autonomous forwarders under full canopy.

There are multiple well-established frameworks and algorithms for autonomous driving in urban environments [72], as well as localization and mapping in roads and buildings [73]. Many of the solutions proposed in these areas have a strong dependency on LiDAR scanners [74], among other sensors. In the field of forest mapping and navigation, several researchers have utilized terrestrial laser scan (TLS) to build point-cloud maps [75; 76; 77; 78; 79]. In some of these works [75; 77], data are collected from fixed-positioned tripods to gain an understanding of how well individual trees can be detected and their diameter at the breast height (DBH) can

be registered. Other studies rely on mobile laser scanning (MLS) [76; 78; 79] to simulate the operation of the harvester.

Navigating in a forest presents several inherent challenges owing to the complexity and lack of structure in the environment [80]. A realistic forest harvesting process involves constant obstructions from cut or fell branches and trunks and irregular microrelief and terrain. Furthermore, the movement of a harvester tends to be rotational most of the time, with constant changes in orientation along the work cycle and very slow translational motion or even with no movement at all while trees are being cut. These particularities of forest navigation bring both advantages and disadvantages to standard development of autonomous mobile robots: idle periods and slow motion aid at realizing real-time operation and data processing, while fast rotations hinder accurate matching of scan data (odometry) owing to a large variability in the distance between near and far objects, and multiple similar objects in different directions. A small error in the orientation estimation can significantly affect the mapping of trees that are farther away [76; 81].

This work focuses on the real-time localization of an autonomous forwarder unit, a forestry vehicle that collects felled logs and hauls them to a local loading area. The required transport distance is usually short, in the range of 100–400 m, and happens along a rough-terrain track a forest harvester has previously made while performing logging. To autonomously navigate in the forest, visual sensors present significant challenges owing to the lack of a stable background from which contours could be extracted, as well as demanding low-light and harsh weather conditions [82]. Forest environments present difficult light conditions during winter and at night, therefore we considered only the LiDAR technology. Nonetheless, it is worth noting that visual-inertial odometry [41], and other methods for estimating structure from motion [83], have seen significant advances over the past few years. For instance, in [83] the authors report improvements in day-night recognition stability. This work required long range up to 60 m for location and wide field of view, and we have not tested any approach based on visual sensors.

3.1.2 LiDAR-Based Localization

Autonomous mobile robots meant to operate outdoors often rely on GNSS sensors as the basic source of global localization data, and then integrate other sensor data through sensor fusion techniques for local position estimation [74]. GNSS sensors alone do not provide enough accuracy in urban or dense environments, with accuracy often in the order of meters [84]. While GNSS sensors have increased their accuracy in recent years, multiple challenges remain in environments with structures affecting the signal path. In particular, GNSS signals are weak in forests, owing to the irregular land contours and the coverage that tree foliage provides [85]. Therefore, we focus on the design and development of localization methods that rely solely on LiDAR

data, with GNSS providing only the initial position before entering the forest or starting the autonomous operation. The following are the main approaches enabling the estimation of the movement of a mobile robot based on LiDAR scans by either comparing consecutive scans (hereinafter also called point clouds) or a given scan with a global point cloud. We refer to comparing scans as scan matching:

Full scan matching. In SLAM algorithms, a necessary step previous to update the map is to estimate the relative movement of the robot with respect to its last known position. This is equivalent to finding a geometrical transformation between the corresponding point clouds. One of the most widely utilized methods for calculating such transformation is the iterative closes point (ICP) algorithm [86; 87]. In ICP, pairs of points between the two point clouds being compared are iteratively selected until a transformation with acceptable pairing error distance is found. In this case the acceptable error is chosen to be 0.6 m, which is 30% of the mean nearest neighbor tree distance. The mean nearest neighbor distance is a dynamical entity (2.0 m in this case), which can be observed from the global map in each case.

Several variants of the ICP algorithm are available through the open-source point cloud library (PCL) [88]. Other popular methods include the perfect match (PM) [89], or normal distribution transforms (NDT) [90] algorithms. Most of these algorithms can be extended and utilized for map-based localization, where a global point cloud is available and a given LiDAR scan is matched with only a subset of the global map. However, in these cases, an estimation of the previous position is needed, since otherwise a standard ICP routine could enter a global search phase, which is computationally much more costly due to convergence safeguards, etc. See e.g., [91] about the intricacies of the guaranteed global convergence. Since our objective is to enable global localization without complete dependence on previous states, we develop a novel method that takes into account the geometry of the environment instead of using the full point cloud.

Feature-based matching. Generic point cloud matching algorithms do not take into account the structure of any potential features within the scans that are being compared. When the environment where robots operate has known structures, feature-based scan matching can significantly enhance the accuracy of the matching process [92; 93; 94]. Moreover, owing to the preprocessing step in which raw data is transformed into features, the computational time required to calculate the transformation can be reduced. In this direction, Zhang et al. presented the LiDAR odometry and mapping (LOAM) method [13], which assumes a structured environment where planes and edges can be detected. Then, the transformation between two point clouds can be estimated by aligning the planar and edge feature points from each of them. Multiple algorithms have extended the original LOAM method for other types of features. Among them, Shan et al. presented the ground-optimized Lego-Loam [14], which delivers optimized real-time three-dimensional point cloud matching in small scale embedded devices. While feature-based matching algorithms yield accurate re-

sults, their applicability is limited in forests, with most methods focusing on urban or indoor environments. In forests or other unstructured environments, edge and planar features are either noisy or missing. In this work, we still rely on the state-of-the-art Lego-Loam algorithm for building a map of the forest prior to the operation of the robots. This process is done offline. However, we develop an alternative method for localization because of our needs for real-time operation and global localization.

Geometrical matching methods. Geometry-based localization algorithms have the potential to recognize the position in constant time, which is an attractive property towards real-time localization with embedded processors. Geometrical methods are often applied to graph-like structures to find geometric transformations that match either the complete graph or a set of subgraphs. Thrun et al. developed the notions of sparse extended information filters (SEIFs) which exploit the structure inherent through the local web-like networks of features maps [95; 96]. In a similar direction, Ulrich et al. proposed a global topological representation of places with object graphs serving as local maps for place classification and place recognition [97; 98; 99]. Among the most relevant approaches to our work are place-recognition algorithms, such as Bosse's work on a keypoints similarities voting method in 2D and 3D LiDAR data [100; 101; 102].

3.2 Delaunay Triangulation Based Localization

In this section we introduced the proposed localization algorithm. Figure 13 shows an overview of our proposed method. The complete system takes as a sole input the point cloud data from the 3D LiDAR, and outputs the position and orientation estimation in the reference of the given global map. The system can be divided into six steps as illustrated in Figure 14. The former two steps are done offline by processing the point cloud data defining the global forest map. These steps have a computational complexity that grows linearly with the number of LiDAR points (first step) and with the number of trees in the map (second step). The latter four steps are then done online on-board the harvester to estimate its position in real time. The steps are:

1. Point-cloud trunk segmentation from the global map (offline).
2. Delaunay triangulation of the global map from the segmented trunk points (offline).
3. Aggregation of 3D LiDAR scans into a local point cloud defining the robot's position (online).
4. Segmentation of trunks from the local point cloud (online).
5. Delaunay triangulation from the local segmented trunk points (online).

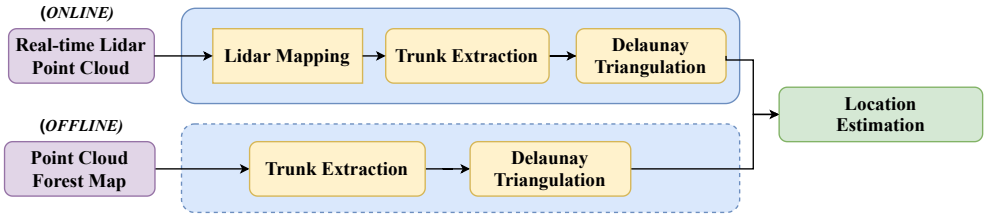


Figure 13. System overview: sensor data is matched with a global map to produce a global position estimation. Consecutive runs are independent.

6. Estimation of a geometrical transformation that matches the local Delaunay triangulation with a subset.

A 2D Delaunay triangulation (DT) process [103] takes in a set of planar points $V \subset \mathbb{R}^2$ and produces a triplet $G = (V, E, T)$, where $E \subset V^2$ is a set of edges and $T \subset V^3$ is a set of triangles with the so called Delaunay property i.e., the circumscribed triangle associated with each triangle $t \in T$ contains no points $v \in V$ others than the three vertex points of the triangle t.

The first step in the proposed method takes a global point cloud $PC_{map} = \{P_i\}_{i=1...n_m} \subset \mathbb{R}^3$, representing the map of the forest, and produces a robust set of trunk positions $PC_{trunks} = \{P_i\}_{i=1...n_t} \subset \mathbb{R}^2$. The accuracy of these positions depends on the accuracy of the LiDAR being utilized and the size of the trees, as they are calculated as the Euclidean average of all LiDAR points defining a tree trunk. Note that PC_{trunks} is not a subset of PC_{map}. The second step in our method then subjects the horizontal plane projection PC_{map} to the Delaunay triangularization graph $G_{map} = (V_{map}, E_{map}, T_{map})$ for matching. As we mentioned earlier, both of these steps happen offline before the robot is deployed, or whenever the robot gets a global map update. A global map update only happens when the robot changes to a new location, or enters an area from which it previously had no information.

The third, fourth and fifth steps, which happen online, cover the generation of a local, real-time Delaunay triangulation of the trees within the field of view of the robot. First, we accumulate and aggregate several raw point cloud scans from the 3D LiDAR and generate a local point cloud $PC_{local} \subset \mathbb{R}^3$. The aggregation relies on real-time LiDAR odometry from LOAM [13]. Then, following the same procedure as with the global map, we generate a robust two-dimensional set of trunk positions $L_{trunks} \subset \mathbb{R}^2$. From the set of trunks, we can define the local DT graph $G_{local} = (V_{local}, E_{local}, T_{local})$.

Finally, in the sixth step of the process we calculate a rigid body transformation (also called Euclidean transformation or Euclidean isometry), defined by a rotation θ and a translation p_t, between the local DT graph G_{local} and a subset of the global DT graph $G_{match} \subset G_{map}$. The transformation relates directly to the robot position and orientation in the global map. Instead of matching large quantities of the

3D point cloud, the proposed system seeks to match a triangularized representation of the local data against a similar precomputed triangularized global representation. The resulting process is computationally efficient and with a low memory demand.

The approach allows noise in the trunk detection in that the set of triangles that we consider for matching are selected to be best matches (different sets at each location), and the final transformation is averaged by this set.

In the rest of this section, we further describe the process outlined above and delve into the details of the most critical steps. In general, the key idea is building a unique 2D graph topology from the 3D point cloud map, and finding the best match between local and global topology.

3.2.1 Trunk Point Cloud Segmentation

With a given map PC_{map} of the forest, an essential step in the proposed method is to extract the trunks points that define PC_{trunks} as a set of landmarks from PC_{map}. Compared to the other environmental features such as branch structures, forest floor vegetation and the bushes in the forest, tree trunks are a natural choice as geometric features.

To segment the trunk points, the first step is extracting the trunk point cloud PC_{trunks} from the map point cloud PC_{map}. We employ the Kd-Tree space partitioning structure to accelerate the neighborhood search [104], together with a Euclidean Cluster to find the trunk point cloud [105]. Instead of focusing on finding every trunk in the PC_{map}, we focus on extracting the most significant ones. We define the most significant trunks in this work as those that show observational stability, i.e., they can be assumed to be easy to observe by the robot in the near future locations. Thus, we do not consider small trunks or bushes, which are not a worthwhile computational investment to locate and give them a landmark status. As a typical trunk is an approximately vertical object, we assume that the point from a stable trunk has a neighboring point located $2m$ above it. This naturally limits the detected trunks, but allows the rest of the algorithm to proceed, since it is enough to get a number of matches from the most prominent objects for the further triangle matches to succeed. Another point is that the trunk detection scheme can be altered in a modular fashion in the future.

As outlined in Algorithm 2, our objective extract all the tree points which are above the ground from PC_{map}. We traverse all points in the PC_{map}, and for each point $p_i = (x_i, y_i, z_i) \in PC_{map}$ we find the nearest neighbor of a hypothetical point $p_i' = p_i + e_3 t_h$, where e_3 represents the vertical unit vector and $t_h = 2\,\mathrm{m}$ above the point p_i. For all such points p_i which do have other points above them, we set the z-axis value to zero and add them to the set P_t. This latter set thus contains the projections of all the points belong to the tree into two dimensions. Following this, we remove the branch points in P_t. To do this, we traverse all points in P_t,

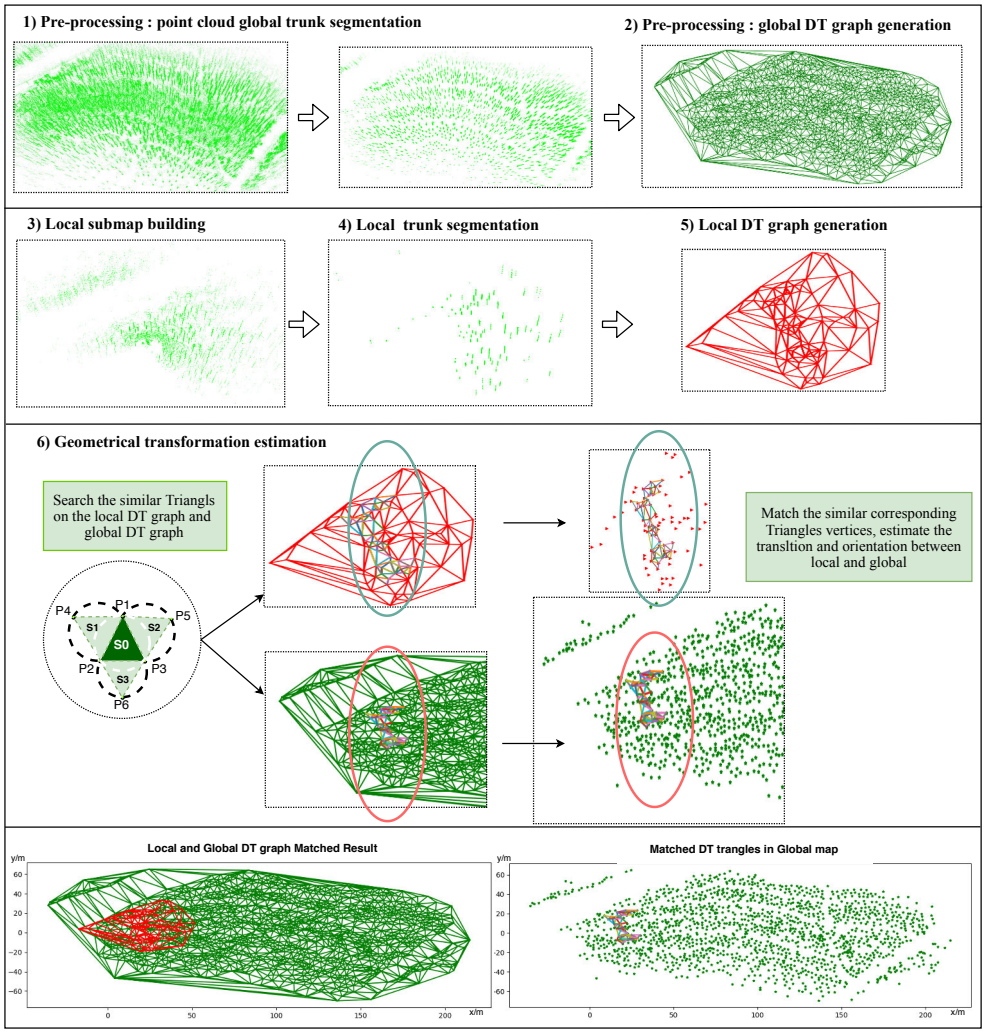


Figure 14. Schematic representation of the proposed method to predict the transformation between a local observation sub-map and the global map.

and find all the points p that have at least one other neighbor point at a distance less than $t_H < 5\,\text{cm}$. All points meeting the aforementioned condition are added to the set PC_t. Then, we form Euclidean clusters (formed by close points) and drop the clusters that have a size in points smaller than a threshold value $cluster\_threshold$. This value has been defined manually taking into account the existing knowledge on the type of trees in the objective forest area. If larger or smaller trees of interest are present in the forest, then this must be taken into account and the value of $cluster\_threshold$ adjusted accordingly. Finally, we compute the mean $\bar{p} \in \mathbb{R}^2$ of the horizontal projections of the cluster points to represent a landmark trunk. These

landmarks then compose the 2D trunk map M_{trunks}. Figure 15 show the proposed trunk point cloud segmentation result. Figure 15a shows the map to be processed, Figure 15b shows the extracted point cloud after removing the ground point cloud and branch point cloud from the input map. Figure 15c shows the final extracted stable trunk point cloud after cluster processing.

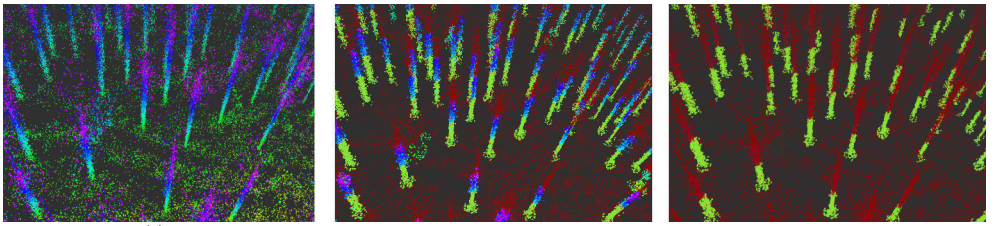


Figure 15. Trunk point cloud segmentation approach. (**a**) The original point cloud map. (**b**) The extracted trunk point cloud (blue and green) after removing ground and branch point cloud (red). (**c**) The final extracted trunk point cloud (green) after clustering process.

3.2.2 Global Map DT Graph Generation

A DT has multiple beneficial properties for our problem setting [106], including the fact that it maximizes the minimal angle at all the vertices of the triangulation. This means that the noise in angular values is minimal in a noisy point set. A DT also defines the so called natural neighborhood around a point (the point set connected to a given point along the triangle edges), which solves the problem of setting the local point cloud feature scope or number of neighbors in an intuitive way [103].

Our objective is to match the local trunk graph G_{local} against a subset of the global trunk graph map G_{map}, and the prerequisite is that there exist some sets of geometric structures that can be uniquely identified in both graphs, i.e., geometrically similar structures such as triangles or sets of triangles that are defined utilizing the trunks as vertices. From the trunk landmarks M_{trunks} of the PC_{map} obtained as described in Section 3.2.1, we get the Delaunay triangulation $M_{trunks} \rightarrow G_{map} = (V_{map}, E_{map}, T_{map})$.

3.2.3 Local Map and Point Cloud aggregation

In a dense forest environment, nearby trunks may be blocked from the LiDAR view. This may result in not enough trunks which can be reliably observed also in the future locations. Thus, in our case, we will employ the LOAM method, which builds a local map from aggregating several consecutive observations. However, we will also explore the direct construction using only one frame observation.

Algorithm 2: Extracting trunks from the point cloud map.

Data: A 3D point cloud map $PC_{map} = \{p_i\}_{i=1..n_m}$
Result: Trunk segmentation point cloud $PC_{trunks} = \{P_j\}_{j=1...N_t}$, 2D Trunk map $M_{trunks} = \{M_k\}_{k=1...n_t}$
foreach $p \in PC_{map}$ **do**
  $SearchPoint : p_{sp} = (p.x, p.y, p.z + h_{th})$;
  $NearestPointSearch : p_{sn} = nearestKSearch(P_m, p_{sp})$;
  $ComputeDistance : distance(p_{sp}, p_{sn})$;
  if $distance < d\_threshold$ **then**
    $P_t \leftarrow p_i$ // Add point to trunks point cloud;
  end
end
foreach $cluster \in Clusters$ **do**
  if $Cluster.points.size() < cluster\_threshold$ **then**
    Delete $cluster$ from $Clusters$
  else
    $\bar{p} = \text{mean}_{p \in Cluster}\, p$;
    $M_{trunks} \leftarrow \bar{p}$;
  end
end

Both the construction of the ground truth map and the aggregation of consecutive LiDAR frames rely on the LOAM method [13]. It operates as follows. Let $P_t = \{p_i\}_{i=1...n}$ represent a raw scanned point cloud received at time t. All these raw point cloud's are processed with the LOAM algorithm to build a local map based on one scan or the aggregation of several consecutive scans. The LOAM algorithm is a state-of-the-art feature-based LiDAR odometry algorithm. LOAM receives the 3D point cloud from the LiDAR, and projects the point cloud onto a range image for feature extraction. By calculating the curvature and some features from each row of the range image, the registration process selects subsets P_e and P_p (edge and plane points, respectively). Instead of comparing all the points, LOAM utilizes only those two subsets to find a transformation between scans, and then a two-step Levenberg–Marquardt optimization method is employed to optimize the six-degree-of-freedom transformation across consecutive scans. The complete LiDAR odometry problem gets solved with a speed of 1 Hz resulting the local map PC_{local} with the computing platform utilized in our experiments.

In this study, the local point cloud PC_{local} was generated with one or several scans by utilizing the estimated position from the LOAM method. The raw LiDAR output frequency is 30 Hz, but not every scan from the LiDAR stream needs to be used to calculate the robot odometry. To reduce the accumulation error and balance

the computation burden, LOAM only considers those scans that the Euclidean distance between two observation positions is longer than a certain threshold distance (e.g., 30 cm) or the angular change is larger than a certain threshold angle (e.g. 30°). As the nearby trunks or objects may block a sector of the view, aggregating more frames observation from consecutive positions helps to increase the number of the stable trunks registered into the map PC_{local}.

After we have obtained the local map PC_{local}, the next step is to generate the DT graph $G_{local} = \{V_{local}, E_{local}, T_{local}\}$, just as with the global map. The methodology explained in Section 3.2.1 applies in this case as well, but this time producing a local map L_{local}, its 2D projected subset L_{trunks} and a local DT graph G_{local}.

3.2.4 Local and Global DT Graph Matching

The DT graph obtained from the global point cloud map, $G_{map} = (V, E, T)$, defines a set V of vertices, a set $E \subseteq V^2$ of edges and a set $T \subseteq V^3$. Each vertex represents a trunk in the point cloud map. Edges establish a relation between a point and its natural neighbors it is connected within the graph. For each local DT graph that we obtain by aggregating consecutive LiDAR frames in real-time, our objective is to find a matching subgraph for G_{local} in the graph G_{map}. We proceed as follows in order to obtain such subgraphs.

Triangle search based on dissimilarity. A dissimilarity $d(.,.)$ of two triangles t_1 and t_2 has two properties: it is always positive, $0 \leq d(t_1, t_2)$, and zero in case of identity, $d(t_1, t_1) \equiv 0$. Typical triangle dissimilarities use an intermediate vector of two descriptors, and some sort of norm between these vectors. For instance, in [107] the authors utilize the ratio of the lengths of the shortest and longest edges, and the angle between those edges. We utilize the intermediate vector (A, l) of a triangle t with an area A and $l = L^2$, where L is the perimeter length. The vector components have the dimensionality of area (in square meters). Then, the dissimilarity $d(.,.)$ is defined as: $d(t_1, t_2) = |A_2 - A_1| + |l_2 - l_1|$.

To speed up the real-time matching process, all triangle perimeters and areas of the global DT graph have been computed offline prior to the real-time matching process. The information utilized by our algorithm while operating for real-time localization is therefore not the raw global map but instead the global DT graph that has been precomputed. In terms of comparing triangles based on the magnitude of the difference of (A, l) vectors, the triangles composed by peripheral points in a graph are usually different because of fewer observed points, so only triangles $T_{selected}$ which not include a peripheral point in the local graph G_{local} are selected. Therefore, a set of candidate triangles $\{T_{selected}\}$ are selected and used to find a matching subset in G_{map}. From $\{T_{selected}\}$, we build the corresponding graph.

There may exist multiple closely similar triangles $T_{candidate}$ in G_{map} to any specific selected triangle $T \in \{T_{selected}\}$ in G_{local}. Thus, the next step is to compare the

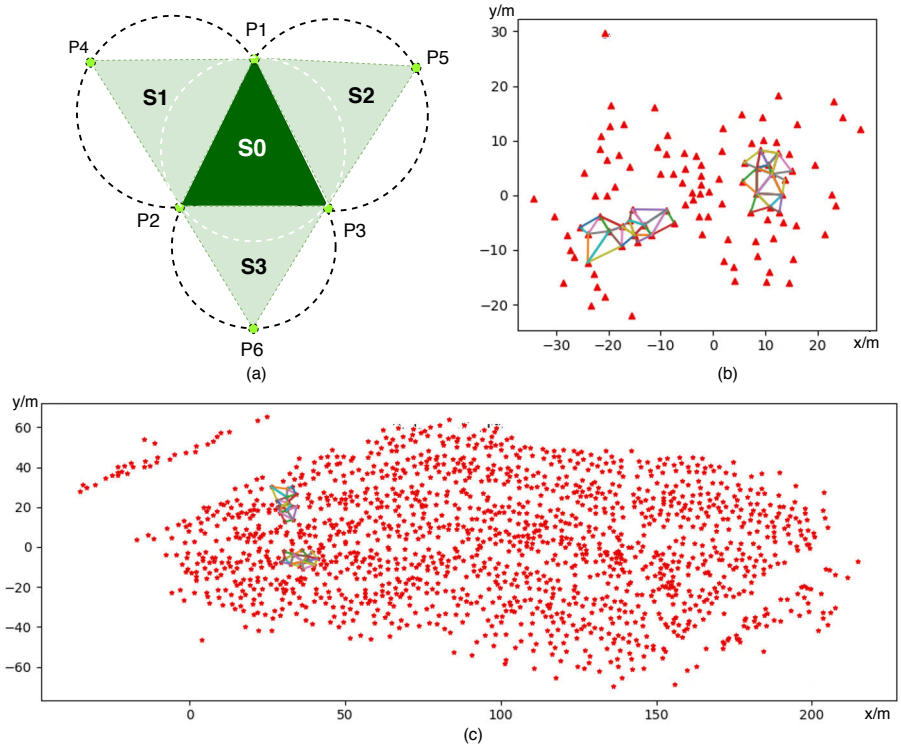


Figure 16. Semantic trunk triangulation on local and global map. (**a**) The target triangle S_0 with its neighbors S_i, $i = 1, 2, 3$. (**b**) All the similar triangles found in the local trunk map, (**c**) the corresponding triangles found in global trunk map.

neighboring triangles too. As we have excluded the peripheral triangles, the global subset $G_{selected}$ that will be the match of $\{T_{selected}\}$ should also have three neighbor triangles. The triangle neighborhood vote is performed using the same dissimilarity measure. As we show in Figure 16, the triangle S_0 is one of the selected triangles $S_0 \in T_{selected}$ in the local graph G_{local} and the three triangles S_1, S_2, S_3 are its neighbors. A feature vector of the triangle S_0 used in the match process is combined from the intermediary dissimilarity vectors of the triangle S_0 and its neighbors (called a triangle star) according to Equation (1).

$$features(S_0) = [A_0, l_0, A_1, l_1, A_2, l_2, A_3, l_3] \tag{1}$$

By comparing the $features(S_0)$ of a triangle S_0 to feature vectors in $T_{selected}$, a set of matching triangles $T_{candidate}$ meeting the error tolerance may be found. After a pair of similar triangles $(S_0, S_0') \in T_{selected} \times T_{candidate} \subset G_{local} \times G_{map}$ are found, the next step would be to estimate the position p_t and the orientation θ, which matches $T_{selected}$ with G_{map}.

Calculation of corresponding vertex pairs. In order to describe the matching process, we use the following notation. A triangle $S_0 = ABC$ consists of vertices A, B and C, and an edge vector $\vec{e} = \vec{AB}$ of an edge $e = AB$ is oriented and signified with its end points.

To solve the transformation parameters p_t and θ, we first need to find the correspondence between vertices of triangle stars $\mathcal{S} = \{S_i\}_{i=0..3} \in T_{selected}$ and $\mathcal{S}' = \{S'_i\}_{i=0..3} \in T_{candidate}$. The definition of a triangle star is easily seen from Figure 17a. As the figure shows, the vertex match between $T_{selected}$ and $T_{candidate}$ can be divided into three steps. The first step is illustrated in detail in Figure 17b. We then find the first matching pair $S_0 = ABC \in T_{selected}$ and $S'_0 = MHN \in T_{candidate}$. Through comparing the side lengths between two triangles, an edge BC of the ABC can be selected so that there is an edge $|NH|$ with similar length in the MHN, so the remaining vertices A and M are a pair of corresponding vertices. For the second step, the goal is to find the other corresponding vertices in ABC and MHN. As Figure 17c shows, selecting one edge which has the vertex A as already known, and one edge contains the vertex M in the $\triangle MHN$, then we compute which side of the edge the remaining vertices C and H are located. This happens by inspecting the value $a \in \mathbb{R}$ of the following formula:

$$a = (\vec{CA} \times \vec{CB}) \cdot (\vec{HM} \times \vec{HN}) \tag{2}$$

If $a > 0$, the vertices C and H are on the same side of the edge AB and edge MN, otherwise the two vertices are on the opposite. In the case shown in Figure 17c, the result is $a < 0$, so the vertex pairs are (C, M) and (B, H). The last step is to match vertices in the triangle neighbors. Since we already know the correspondence between triangles ABC and MHN, we just need to get which two vertices are included in the neighboring triangles. For example, as we get the corresponding pairs of vertices (A, M) and (B, H) from last two steps, the last vertices E and Q are a match, too.

The match between vertices of two triangle sets is now complete. The match relation is one-to-one, so it can be recorded as a permutation function: $f : [1, 6] \subset \mathbb{N} \to [1, 6] \subset \mathbb{N}$. This way each vertex $V_i \in T_{selected}$ has a matching vertex $V_{f(i)} \in T_{candidate}$. Next we will use the permutation f while estimating the best possible translation and rotation.

Rotation and translation estimation

By comparing the vertex orientations in the found two triangle sets $\mathcal{S} = \{S_i\}_{i=0..3} \in T_{selected}$ and $\mathcal{S}' = \{S'_i\}_{i=0..3} \in T_{candidate}$, we can find a unifying 2D rigid body transformation $T[p_t, \theta]$ between the local sample and the global map. The transformation T consists of a rotation of the angle θ followed up with a translation p_t.

To calculate the transformation parameter p_t we utilize the definitions in Equation (3), where the translation t_p is the best possible one estimate, since it equates the mean of two patterns. The angle θ is defined by measuring how large a rotation is needed to transform a vectors $V_i - \overline{V}$ to $V'_{f(i)} - \overline{V}'$, where the $\overline{V}$ represents the mean of $T_{selected}$ and $\overline{V'}$ is the mean of the $T_{candidate}$.

There are six pairs $i = 1...6$ of vertices coupling $T_{selected}$, T_{local} and $T_{candidate}$ from G_{map}, and six possible candidates β_i to be chosen as the final local rotation θ. We choose the value of β_i which fits the triangle patterns $T_{selected}$ and T_{local} with the least error. One important detail is arranging a signed angular measure between two vectors. We use a rectangular rotation matrix $P = \left(\begin{smallmatrix} 0 & -1 \\ 1 & 0 \end{smallmatrix}\right)$ to get a positive $sign(Pa \cdot b)$ for rotation angles, which move a vector a to a vector b. We omit possible fit minimizing values in between the angles $\{\beta_i\}_{i=1..6}$ because the objective at this stage is to only have a close estimate and not the final position estimation.

Using a zeroth power as a shorthand when producing unit vectors $a^0 = a/\|a\|$, we define counter-clockwise oriented angles β_i, which rotate vectors V_i parallel to vectors $V_{f(i)}$. Finally, we define approximate values for the translation p_t and the rotation θ according to Equation (3).

$$
\begin{cases}
\overline{V} = \underset{V \in T_{selected}}{\text{mean}} \; V \\
\overline{V}' = \underset{V' \in T_{candidate}}{\text{mean}} \; V' \\
p_t = \overline{V}' - \overline{V} \\
\beta_i = sign\left(PV_i \cdot V_{f(i)}\right) \arccos\left(V_i^0 \cdot V_{f(i)}^0\right), \;\; i = 1..6 \\
\theta = \underset{\beta_i,\, i=1...6}{\arg\min} \sum_{j=1...6} \left\| \begin{bmatrix} \cos\beta_i & -\sin\beta_i \\ \sin\beta_i & \cos\beta_i \end{bmatrix} \left[V_j - \overline{V}\right]^T - \left[V_{f(j)} - \overline{V'}\right]^T \right\|
\end{cases}
\tag{3}
$$

Geometric verification, final translation and rotation estimation

From the G_{local} match against G_{map}, usually multiple matches $\{(\mathcal{S}_k, \mathcal{S}'_k)\}_{k=1...m}$ between $T_{selected}$ and $T_{candidate}$ can be found. Therefore, we need to select one among m transformation candidates with rotation and translation parameters θ_k and t_k. This final step is to find the best estimation between G_{local} and G_{map}. Let the corresponding points of a match candidate be (V_{local}, V_{map}). Starting from the previously computed initial guesses θ and p_t, we select the final solution from Equation (4). Note that this does not ensure global convergence, and unexpected results might be obtained. For instance, if the global map is too homogeneous, then different regions

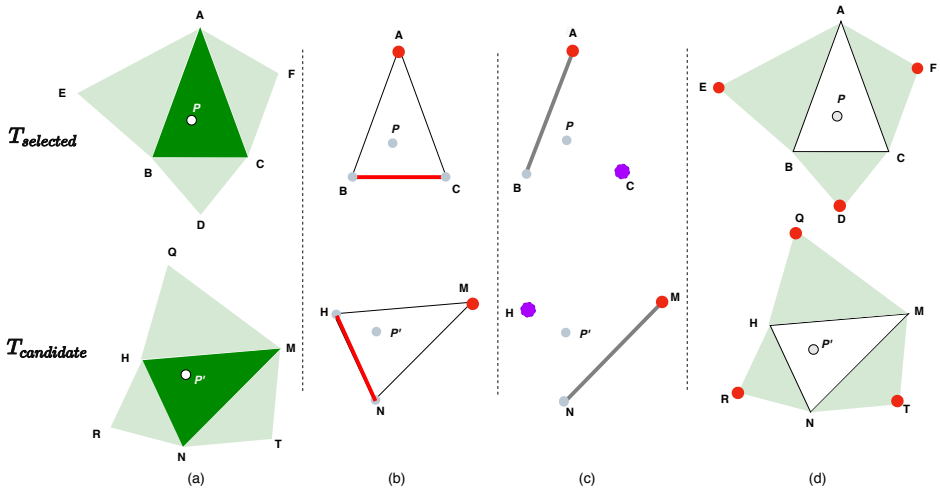


Figure 17. Semantic triangle matching process between local and global maps. (**a**) The selected triangle $T_{selected}$ in G_{local} with candidate matched triangle $T_{candidate}$ in G_{map}. (**b**) Find the first corresponding point A in $T_{selected}$ with its M in $T_{candidate}$ by finding the similar featured side length. (**c**) Find the rest points B, C and its corresponding points H, N. (**d**) Find the corresponding neighbor points E, D, F and its corresponding points Q, R, T.

might have sets of trees that yield similar local maps. The risk for this can be reduced by taking into account the previous locations, or accumulating larger local maps until the matching gives a single matched subgraph from the global map with low error. Another risk is that not enough trees might be detected, which could again result in multiple global subgraphs matching with similar error. Finally, if the map is too large then multiple positions could also yield similar error. This latter scenario can be avoided by creating a set of smaller (non-disjoint) maps from the global map. Through our experiments, nonetheless, we have been able to confirm stable position estimation when enough consecutive LiDAR scans were aggregated.

$$(\theta, t_x, t_y) = \arg_{\beta, x_t, y_t} \min_{\substack{\beta \in B \\ x \in D_x \\ y \in D_y}} \sum_{j=1...m} \left\| \begin{bmatrix} \cos\beta & -\sin\beta & x \\ \sin\beta & \cos\beta & y \\ 0 & 0 & 1 \end{bmatrix} \left[V_j^T - V_{f(j)}^T \right] \right\|, \quad (4)$$

where m is the number of local matches, $\beta \in B = [\min \theta_{j\,j=1...m}, \max \theta_{j\,j=1...m}] \subset \mathbb{R}$ goes through the scope occurring in the local rotation angles and x goes through a similar scope of D_x occurring in translations along the x axis. D_y is defined correspondingly on the y axis. Note that both Equations (3) and (4) require the vertices to be in a homogeneous form $V \rightarrow [V\ 1]$. The local start point is (x_0, y_0) with an orientation of θ_0, and the final estimation of the robot position is $(x_0 + t_x, y_0 + t_y)$ with an orientation θ in the global map.

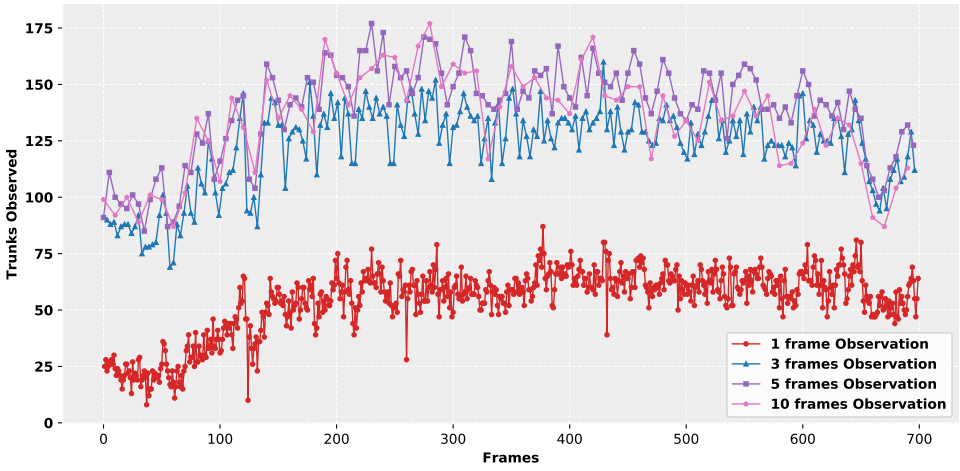


Figure 18. Number of trunks observed for different number of consecutive aggregated frames of LiDAR data.

3.3 Experimental Evaluation

This section presents our experimental results. We analyze the performance of our method from the point of view of the trunk segmentation as well as the DT matching between the local graph and a subset of the global graph.

3.3.1 Study Area & Hardware

The study area covers one mature pine stand ready for second thinning, which represents a typical second thinning Finnish forest. This second thinning increased the mean distance of trees from 4.4 m to approximately 8 m judging by a before and after point cloud maps. The location of the study area is illustrated in Figure 20. The terrain profile was mostly flat, the maximum height difference over the site was 8 m on the covered area of approximately 200 m by 300 m. The overall trail length is approximately 200 m covering roughly 2200 trees.

The mobile platform was a Komatsu Forest 931.1 forest harvester with a GNSS unit and the LiDAR unit attached to the top of the cabin. The harvester has physical dimensions (length, width, height) of 7.6 m by 2.9 m by 3.9 m, and its mass is 19,610 kg.The maximum driving speed is 8 km/h off-road and 25 km/h on road.

The on-board LiDAR is Velodyne-16, a 16 channel LiDAR with 3 cm distance accuracy, 360°horizontal field of view, 30°vertical field of view with $\pm 15°$ up and down, 5–20 Hz scanning frequency and 100 m scan range. As Figure 21 shows, the LiDAR unit is fixed on the front of the harvester with a 210°horizontal view. The harvester has a folding frame, which means the point cloud frames scanned have

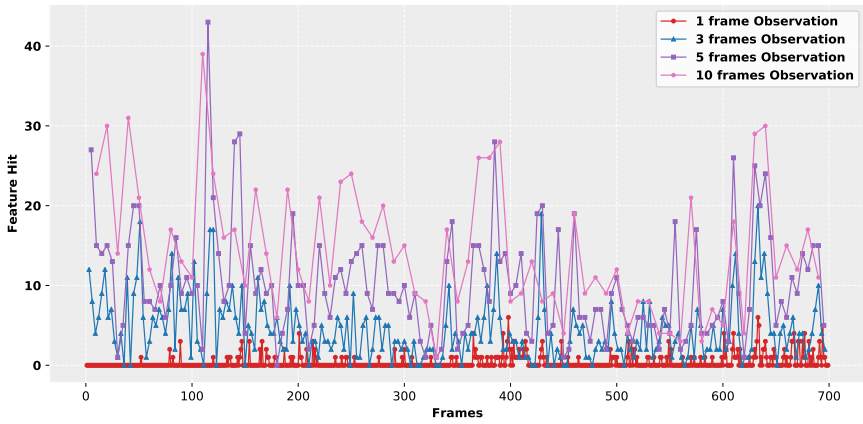


Figure 19. Number of trunks matched between G_{local} and G_{map} for different number of consecutive aggregated frames of LiDAR data.

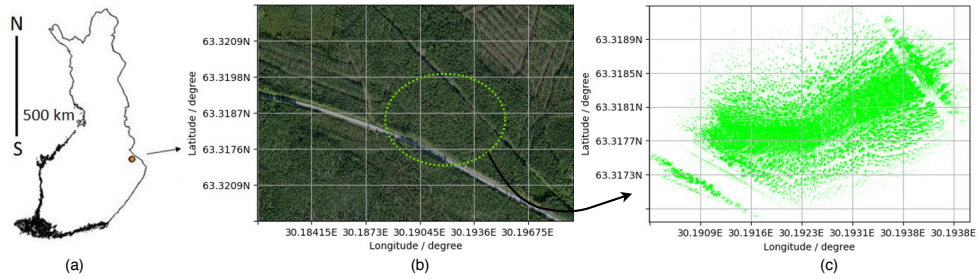


Figure 20. (**a**) The test site in Lieksa, Eastern Finland. (**b**) The forest canopy map from Google maps. (**c**) Point cloud map of the study area.

constantly alternating horizontal orientation during the work cycle.

(a) (b)

Figure 21. Sensor and harvester platform. (**a**) Komatsu Forest 931.1 forest harvester. The LiDAR scanner is marked by a red circle in the front of the windshield of the cabin. (**b**) A close-up of the LiDAR scanner.

The GNSS data was recorded by Spectra SP60 GNSS unit. This unit fully utilized all six GNSS systems: GPS, GLONASS, BeiDou, Galileo, QZSS and SBAS. In SBAS (WAAS/EGNOS/MSAS/GAGAN) mode, the horizontal position error smaller than 50 cm, and the vertical error is smaller than 85 cm. In differential GNSS mode, the accuracy is able to reach 25 cm in horizontal and 50 cm in vertical accuracy. In Real-Time Kinematic position (RTK) mode, the accuracy can reach 8 mm in horizontal and 15 mm in vertical accuracy. Nonetheless, these values are not achievable under dense foliage in a forest environment. Regarding the computing platform, the proposed methods have been tested on an Intel Core i7-9700K CPU (8 cores, up to 3.60 GHz), and 16 GB of RAM. The GPU was not utilized for the implementation of any of the algorithms involved in the localization process introduced in this work.

This study concerns the development of a localization algorithm for a forwarder unit to use a tree map during autonomous navigation under potentially heavy canopy. For that purpose, the forwarder scans were simulated from the existing LiDAR produced by a forest harvester. The harvester fells trees and a sector of the view is constantly obstructed by a tree being processed. Effects of this disappear when the initial tree map is being constructed using normal SLAM procedures. This study concerns the utility of the local scans done by a forwarder to be used in comparison to the existing map.

3.3.2 Trunk Segmentation Performance

Figure 18 shows the number of trunks that can be observed from a single LiDAR scan, and when aggregating three, five or ten consecutive frames. Figrue 19 shows

Table 6. Number of trunks detected and success match for different number of consecutive aggregated frames (average number of trunks matched per scan and total ratio of successful matches).

#Frames	#Trunks	Average #Matching Triangles	Success Rate
1	54.09	1.71	22.89%
3	122.98	4.97	88.79%
5	139.54	10.22	99.28%
10	133.96	14.37	100%

the number of features matched between G_{local} and G_{map} for different number of consecutive aggregated frames. In Table 6 we show the average number of trunks observed, as well as the average number of trunks that can be matched with the global map after generating the DT graph. We also include the overall matching success rate, which is defined as the number of local DT graphs that have been successfully matched with the corresponding subsets of the global DT graph. From individual LiDAR frames, we were only able to obtain an average of 54.09 valid trunks in total. When aggregating consecutive scans, we were able to obtain approximately a twofold increase in the number of detected trunks, and a tenfold increase in the number of matched trunks. The number of aggregated frames thus has a significant quantitative impact on the performance of our algorithm with a difference of up to an order of magnitude in the number of matched trunks. As the number of trunks and their positions determines the appearance of the local DT graph, this will directly affect the number of features detected by our descriptor. The number of trunks observed is the key factor influencing the overall results of our algorithm. From Table 6 we can see the relationship between the DT matching success and the number of trunks. With a single frame observation, only an average 1.71 trunks are successfully matched; with three aggregated frames, there are in average 4.97 trunks matched; with 5 and 10 aggregated frames, there are more than 10 trunks in average that our algorithm can match with the global map, taking the overall DT match success rate to 100% in the latter case.

3.3.3 Computational Time and Accuracy

The key matching process is based on the calculation and matching of the Delaunay triangulations, which is in turn based on the aggregation of 1, 3, 5 or 10 LiDAR frames. These four possibilities with a varying number of aggregated frames are labeled as C1, C3, C5 and C10 in Figure 22. The aggregation of frames adds complexity and requires computing time both in the local map creation phase (black line) and the similarity matching time (blue line). We can see that the rotational location error significantly improves when aggregating more frames. The computing time of the match process grows approximately in terms of $O(\Delta_r^{-2})$, where Δ_r is the angular ac-

curacy achieved by different choices C1, C3, C5, C10. In summary, the computation time is about 0.1 seconds for a single frame (C1) and 0.6 seconds when aggregating 10 frames (C10).

To test our pose estimation accuracy, we used the original LOAM position as the ground truth. Figure 22 depicts only the rotational error, since the translation error was approx. 0.2 m with a standard deviation of 0.14 m in all cases. It is worth noting that the rotational error is not cumulative, since the match between the local and global DT graphs is being done separately and independently at each location. The previous location is only used as the initial state. The maximum occurred translation error was approx. 0.5 m and the maximum rotation error 3.23°, both for the C1 scenario.

We can also see that the match time is relatively stable in the C1-C5 scenarios. The local map creation time (aggregation of frames) stabilizes as the number of frames increases, owing to the stabilization in the number of trees that can be observed. The segmentation algorithm is implemented in C++, while the matching process is implemented in Python.

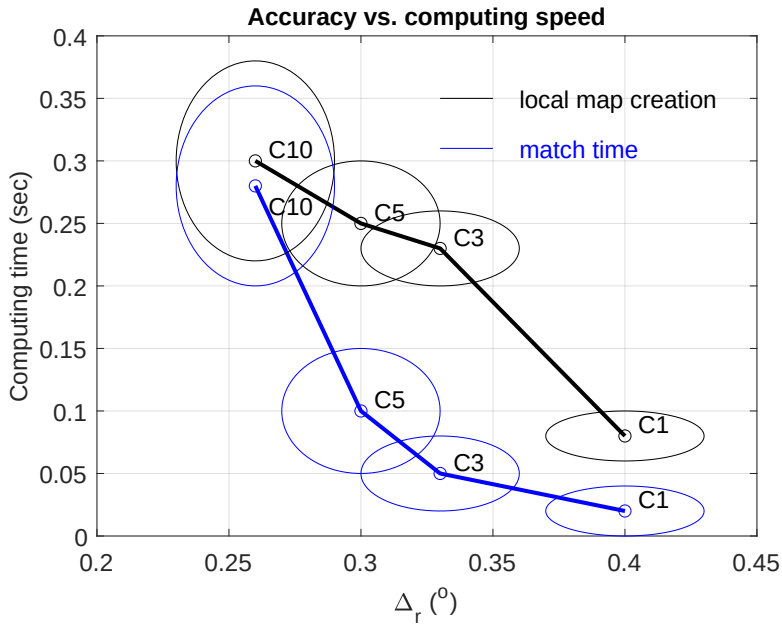


Figure 22. Dependency of the rotational accuracy and computation time on the choice of the number of aggregated frames. Both the local map creation time (black) and the similarity matching time (blue) are depicted. Number of frames ranges from 1 to 10 (C1, C3, C5, C10). Ellipses represent the 2 s.t.d. of the measurements. Δ_r is the angular accuracy achieved by different choices C1, C3, C5, C10.

3.4 Discussion

An application topic of the proposed navigation system is an automated transport robot, which moves logs from the forest to the collection road. Since work happens in two phases, where a forest harvester produces stripping roads and fells trees, and the collection phase, a robot can assume to have the global map built by the harvester, which is not necessarily autonomous. This way its task is just to orientate itself along the strip road. A challenge arises from the high utility load of the transfer robot, which can be up to 8000 kg. A transfer robot may not have an as big capacity, but the ability to locate itself, and to detect possible stability risks is a requirement.

In addition, a transfer robot moves faster than the harvester going through its work cycle. A good estimate in rough terrain is 1.0 m/s. Thus, a fast and accurate odometric computation is essential.

3.4.1 Topology Mapping

In this work, the forest trunk topology map is generated from the previous LiDAR data by the harvester. There are also two alternative ways to generate the global map: a large-scale (e.g., nationwide) digital forest inventory, or an aerial laser scan (ALS) from a drone working as an autonomous team member of the harvester and the robot carrier. Compared to the point cloud map, the topology map has the potential to do fast localization and dynamic update without much computation burden and makes a three actor teams (harvester-drone-robot) an interesting target of further study.

Even the data utilized in our experiments were gathered in a relatively flat forest stand (approximately 8 m height difference across the site), the proposed approach does not make any assumptions regarding the inclination of the terrain, but instead we assume that trees grow relatively upright. This is because the tree stem detection is aware of the vertical orientation independent of the dominant terrain orientation. There are only few relatively simple modifications to the presented algorithm, where the triangle matches and triangle stars would be defined in actual 3D. The actual effect of topographic features has to do with the reduced scope of the LiDAR scanning. This could be alleviated by an aided UAV unit, but our first approach is studying the usability of the tree maps in relatively benign conditions.

3.4.2 Potential Accuracy Improvements

As illustrated in Figure 13, each location operation is independent, except naturally each step will be initialized by the previous location to gain advantage in the corresponding search operation. In addition, one can reduce the number of localizations e.g., by assuming an interpolated movement in between sampled positions. Since this is a registration task, there is no actual error accumulation per se. The location

accuracy and computation time are tightly related as seen in Figure 22.

From the experiments, we can see that the number of trunks detected in each observation significantly influence the result of the success matching ratio. In our method, due to the LiDAR sensor characteristic, far trunks are unable to be recognized because of too few LiDAR points sampled from the appearance. Therefore, we can employ other sensors producing photogrammetric point clouds which can obtain more detail of the trunk appearance, which lets the system get more capability to recognize the trunks from PC.

Another significant source of error can be found in the trunk position estimation process. In our method, the G_{map} and G_{local} are generated from the segmented trunks points in PC_{map} and PC_{local} by calculating the average coordinate of points. However, it is impossible to get each individual trunk whole appearance observation with several consecutive LiDAR observations. As the observation in different positions will get different point cloud data of different sides of the trunk, the estimation pose of the trunk will be different in each local graph. However, the global graph has a more accurate trunks position as more details of each trunk collect from different sensors and different views, so the more accurate trunk position estimation in the local map can increase the chance of successful matching between local graphs and global graph. To reach a more robust system, we also can utilize other sensors like a camera through sensor fusion to get more details about each individual trunk.

3.4.3 Potential Computational Improvements

We proposed an efficient, robust framework to locate the robot harvester in large area forest and we tested it on a real harvester. In our case, the GNSS/GPS info is not taken into account. The location accuracy is approx. 0.3 m from consequential field measurements and approx. 2 m when comparing to general odometric result. This accuracy is enough to give a robot an initial position, and this can accelerate the matching process by conveniently initializing the search with a close match. For a real application, the GNSS also can help decide which global point cloud map it will access from the cloud server for the localization.

The local Delaunay triangularization generating the local trunk map T_{local} can be implemented in a radius-limited way by using the S-Hull method [108], which limits the size n of the computational task with the well-established complexity $O(n \log n)$. A selection of an active subset of triangles can be made in the global map T_{local} based on the previous localization result. Together these improvements have the potential to make the matching process much faster.

There is a possibility to speed the convergence of the search of the final transformation in Equation (4) by using a branch-and-bound algorithm [109] with properly set estimates for local extreme. This possibility is left for future research.

All in all, the C1-C5 cases are applicable to the intended situation where the

robot moves at approx. 1 m/s and has a sampling rate 2 scans/sec. The case C10 is within the reach after implementing the above-mentioned improvements.

3.5 Summary and conclusions

In this chapter, we proposed a simple yet effective segmentation-based approach to detect trunk position and Delaunay triangulation (DT) graph-based localization method for autonomous robots navigating in a forest environment. The proposed methods can provide accurate positioning based only on real-time LiDAR data processing in the unstructured and relatively complex environments that forests represent. The proposed method can be utilized for harvesters or other autonomous robots enabling fast global localization and recognizing individual trunks in real-time.

The experiments show the proposed method reach accurate global localization precision without a good initial pose or GNSS signal. The proposed method is simple and efficient, and it is a sensible solution to meet localization needs of harvesting operations in the forest. In future work, we plan to explore the forest localization algorithm in the context of significantly larger forests and to apply the proposed method at a system level for map updating or within the SLAM stack. Gathering more data, we will also be able to further analyze the performance of our algorithm when the sensors have different points of view, or when the global map is gathered in different conditions than the real-time LiDAR data of the forwarder, such as different vehicles or sensor settings.

4 Multi-Sensor Fusion for Accurate Localization in Urban Environment

In dense urban environments, accurate localization is a paramount aspect of a robot's autonomous operation [110; 44]. However, smaller pathways restrict robot movement, while dynamic environments require continuous adaptation to unpredictable changes, posing significant technical challenges for autonomous navigation and mapping [45]. In addition, the robot mission often adds accuracy requirements, such as in autonomous post delivery [111]. Camera and LiDAR-based perception systems are extensively used in autonomous navigation and operation in urban environments. While camera data captures visual details such as colors, textures, and shapes, offering more semantic and qualitative information [112; 113], LiDAR measurements are more accurate and provide precise geometric descriptions of objects, including their shape, size, and distance from the sensor [114].

The objective of this chapter is to analyze and compare different approaches for vehicle localization estimation while focusing on developing a sensor fusion technique for precise localization in dense urban environment. Additionally, this research aims to propose a strategy for reconstructing a section of a local map in scenarios where data corruption or substantial environmental modifications have occurred.

4.1 SLAM and Autonomous Driving Vehicle

The past decade has seen a boost in the development of autonomous vehicles for civilian use. Google started the development of its self-driving technology for cars in 2009 [115], and since then a myriad of industry leaders [116; 117], start-ups [118], and academic researchers [119] have joined the race in the technology sector, a race to make everything autonomous. In any mobile robot or vehicle, SLAM algorithms are an essential and crucial aspect of autonomous navigation [120; 44].

Autonomous robots or self-driving cars will potentially disrupt the logistics industry worldwide [121]. Autonomous trucks or autonomous cargo vessels are already in advanced stages of development and might be seen in operation within the next five or ten years [122]. However, both technological and legal challenges remain within the so called "last-mile" delivery [123]. Last-mile refers to the last step in the delivery of goods from a local logistics or supply center to the clients' door. In this chapter, we utilize data gathered using a small delivery robot from Jingdong, one of

the top two e-commerce platform and B2C online retailer in China.

The development of simultaneous localization and mapping (SLAM) algorithms has seen a rapid evolution over the past two decades [13; 124]. In SLAM algorithms, information from a wide range of sensors can be used to map the environment and localize the vehicle or robot in real time. These include inertial measurement units, monocular or binocular cameras, GNSS sensors, LiDARs, ultrasonic sensors or radars, or wheel encoders [125]. Detailed 3D maps in the form of point clouds can be generated, for instance, from 3D LiDARs or with stereo vision [114]. We focus on the study and comparison of different localization methods for a small delivery robot in dense urban environments. In these scenarios, an existing map of the operating environment is obtained in advance and either pre-loaded or accessible by the autonomous robot. The map is used in order to obtain more accurate localization by matching each scan with a certain area of the map in real-time [126; 44].

The local motion of a robot or vehicle can be estimated directly by integrating data from inertial measurement units, including accelerometers, gyroscopes and compasses. Alternatively, different odometry methods can be applied based on non-inertial sensors. Visual odometry algorithms utilize feature extraction and tracking, while LiDAR-based odometry uses mostly geometric information [13; 124]. Inertial measurement units can be easily combined with wheel encoders. Differential GNSS measurements also provide accurate local motion estimation [127]. Global localization can be estimated either with GNSS data, or by comparing sensor data with predefined maps or information gathered a priori. For instance, different methods exist to match a LiDAR scan with section of a 3D point cloud that defines a map of the operational area [126; 44]. Over the past decades, researchers from both industry and academia have been exploring the utilization of these methods and their combinations to obtain accurate mapping and localization. More concretely, scholars often refer to the combination of different sensor data as sensor fusion or data fusion. In this chapter, we compare different techniques and provide arguments on the best sensor fusion techniques for a small delivery robot for last-mile delivery.

The algorithms, analysis and results presented in this chapter were mostly developed during the JD Digital (JDD) Globalization Challenge Competition in "Multi-sensor fusion localization for autonomous driving". The challenge was a global competition, with 4 classification divisions depending on the geographical location of the team. Our team ranked first in the US division semi-final and qualified for the global final in Beijing, China, where the four semi-final winners competed for the first prize. The available sensor data during the competition was GNSS and gyroscope data, wheel odometry and the output from a 16-channel 3D LiDAR. A map of the area was given as a 3D point cloud. Multiple datasets exist to test and benchmark different localization algorithms. However, the most accurate algorithms are obtained through fine tuning and parameters specific for the dataset, with different parameters being potentially necessary to achieve the optimal accuracy in a different

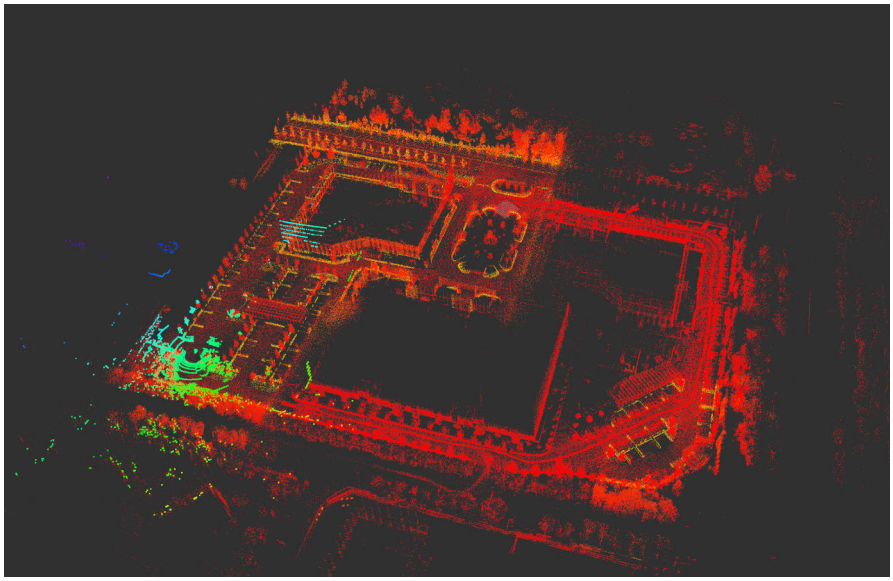


Figure 23. Illustration of the matching process between the pre-acquired map (red) and current LiDAR scan (green-blue).

sensor or environment setup [128]. Therefore, in this chapter we have utilized the data provided as part of the JDD competition as it was gathered from the sensors on-board the vehicle in order to compare a variety of methods. This ensures that our algorithms can be implemented on the same robot without a significant impact to performance.

4.2 JDD Dataset and Challenge

In this section, we describe the dataset that we have utilized and the different localization approaches. We also introduce a strategy for situations where the existing map data is corrupted, outdated, or part of the data is missing.

The data utilized in this chapter was provided as part of the JD Discovery Global Digitalization Challenge from December 2018 to January 2019. The data was gathered using JD's autonomous last-mile delivery robot. The data includes: (1) GNSS directional and positional data referenced in the World Geodetic System (WGS84) format; (2) LiDAR data as a 3D point cloud; (3) raw accelerometer and gyroscope data; and (4) wheel speed meter. Ground truth data is provided as well. The output of the 3D LiDAR is given at 10 Hz, IMU data is acquired at 100 Hz and GNSS data is updated at 5 Hz. In addition, a map of the objective operation area is given. The map represented as a point cloud is shown in red in Fig. 23. The dataset contains sensor data recorded in an 800-second long closed loop movement.

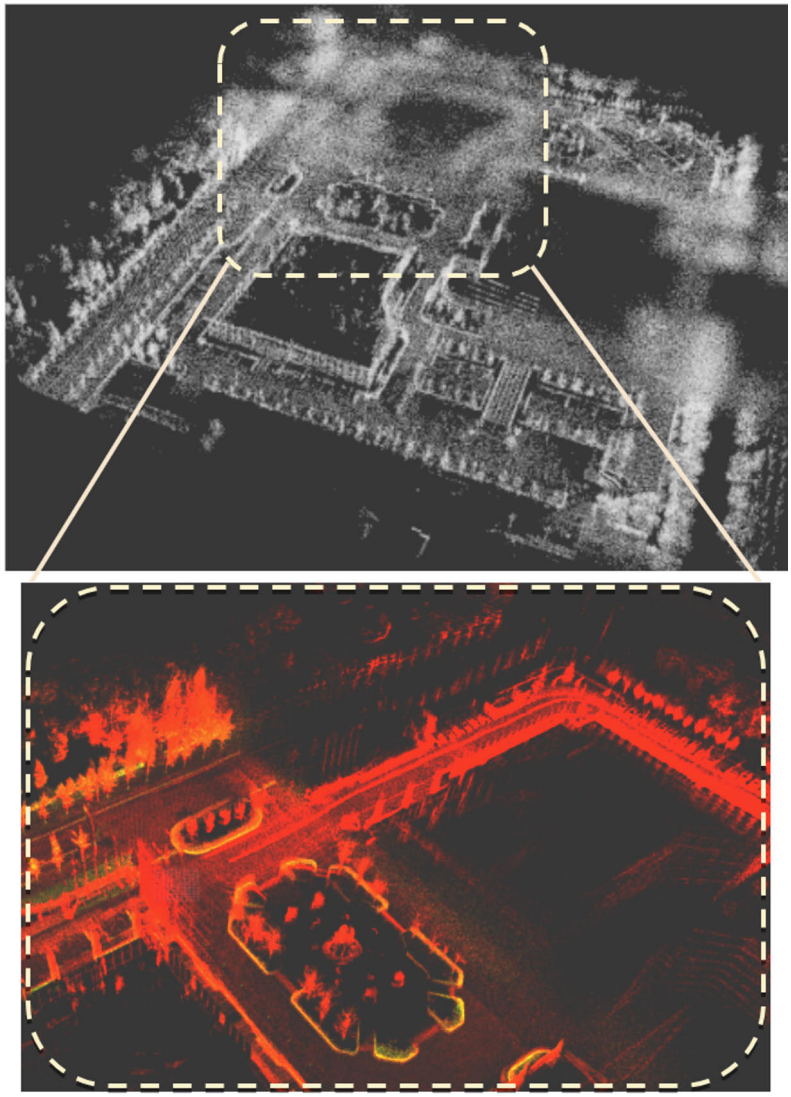


Figure 24. Corrupted map reconstruction: On the top, a map with noise added in some areas to simulate corrupted data. In the bottom, one of the two corrupted sections of the map has been restored using GNSS, IMU, and LiDAR odometry.

In order to both read and process data, ROS has been utilized. ROS (Robot Operating system) is an open source operating system for robots, which provides a publish-subscribe communication framework that allows for rapid development of distributed robotic systems [129]. ROS provides algorithm reuse support for robot research and development, as well as abstraction of data models for easier integration of different modules. PCL (Point Cloud Library) is a cross-platform open source C++ library, which implements common algorithms and data structures of point

clouds [130]. It can realize point cloud acquisition, filtering, segmentation, registration, retrieval, feature extraction, recognition, tracking, surface reconstruction, visualization and so on. If OpenCV is the crystallization of 2D visual data acquisition and processing, PCL has the same position in the 3D geometrical data domain.

4.3 Localization Methods

Based on the available sensor data, we have utilized five different approaches to estimate the vehicle's localization. In each approach, we use a different combination of sensors and describe how the robot position is calculated based on their data.

4.3.1 GNSS-based localization

One of the most traditional methods for outdoor robot localization is to use a global navigation satellite system. In this case, data from multiple satellite constellations was available and used for increase accuracy. GNSS data error are mostly caused by the atmospheric conditions and multi-path interference. The effect of the environment in a larger scale and the atmospheric conditions can be minimized using differential GNSS readings, and assuming that the real-time error is equivalent to the error obtained in a near known location with which the system is synchronized. However, in this work we have not relied on differential GNSS.

4.3.2 GNSS+IMU Localization

We can easily combine GNSS data with inertial data, including both accelerometer and accuracy. As differential GNSS has not been implemented in this case, instead, the results labelled as "IMU" utilize the IMU readings for local motion estimation, and the GNSS reading for an initial global estimation and estimations when the robot movement is almost zero for a prolonged period of time.

4.3.3 LiDAR Odometry

Zhang *et al.* introduced LiDAR odometry as an alternative to the more classical visual odometry techniques [13]. As with many odometry approaches, features are extracted from data and compared within consecutive frames, or scans in the case of a LiDAR. Features extracted from LiDAR data are usually based on geometrical aspects. These include corners and surfaces, for instance. Because LiDARs are able to provide high accuracy distance measurements even for objects far away from the sensors, LiDAR-based odometry is able to provide higher accuracy than visual-based odometry in open space situations with clearly differentiated objects. An implementation based on Zhang's algorithm has been used in this case.

4.3.4 NDT-Based Localization (NDT+)

The NDT algorithm is a kind of registration algorithm that uses the existing high-precision point cloud map and real-time 3D LiDAR point cloud data to achieve high-precision localization.

NDT algorithm does not directly compare the distance between points in point clouds map and points in LiDAR point clouds. First, the NDT algorithm will transform the point cloud map into the normal distribution in three-dimensional space.

If a variable X is normal distribution $X \sim (\mu, \delta)$, then it can be described as:

$$f(x) = \frac{1}{\delta\sqrt{2\pi}} e^{\frac{-(x-\mu)^2}{2\delta^2}} \tag{5}$$

where μ is the mean of the variable distribution and δ^2 is the variance. For a multivariate normal distribution, its probability density function can be expressed as:

$$f(\vec{x}) = \frac{1}{\sqrt{(2\pi)^k \left|\sum\right|}} e^{-(\vec{x}-\vec{\mu})^T \sum^{-1} (\vec{x}-\vec{\mu})} \tag{6}$$

Where $\vec{x}$ represents the mean vector, k is the variable amount, and $\sum$ represents the covariance matrix.

The first step of the NDT algorithm is to divide the point cloud into a 3D grid coordinate. For each cell, the probability distribution function(PDF) is calculated based on the points distribution density in the grid.

$$\vec{\mu} = \frac{1}{m} \sum_{k=1}^{m} \vec{y_k} \tag{7}$$

$$\sigma = \frac{1}{m} \sum_{k=1} m(\vec{y_k} - \vec{\mu})(\vec{y_k} - \vec{\mu})^T \tag{8}$$

Where $\vec{y_k} = y_1, y_2, y_3, ..., y_m$ denotes all LiDAR points in a grid. $\vec{\mu}$ and σ represent the average position and covariance matrix of the points in a grid.Then the PDF can be expressed as:

$$f(\vec{x}) = \frac{1}{(2\pi)^{\frac{3}{2}} \sqrt{|\sigma|}} e^{-(\vec{x}-\vec{\mu})^T \sum^{-1} (\vec{x}-\vec{\mu})} \tag{9}$$

We use the normal distribution to represent the discrete points of each grid. Each probability density function can be considered as an approximation of a local surface. It not only describes the location of the surface in space but also contains information about the direction and smoothness of the surface.

After calculated the PDF of each grid, then our goal is to find the best transformation. The LiDAR point cloud set is $X = \vec{x_1}, \vec{x_2}, ..., \vec{x_n}$, and the parameter

of transformation is $\vec{p}$. The spatial transformation function $T(\vec{p}, \vec{x_k})$ represents using transformation $\vec{p}$ to move point $\vec{x_k}$, combined with the previous calculated state density function(the PDF of each grid), so the best transformation $\vec{p}$ should be the transformation of maximum likelihood function:

$$Likelihood : \theta = \prod_{k=1}^{n} f(T(\vec{p}, \vec{x_k}))$$

And the maximum likelihood is equivalent to minimum negative logarithmic likelihood:

$$-\log\theta = -\sum_{k=1}^{n} \log f(T(\vec{p}, \vec{x_k}))$$

The task now becomes to minimize the negative logarithmic likelihood by using an optimization algorithm to adjust the transformation parameter $\vec{p}$. We can use the Gaussian Newton method to optimize the parameters.

The main problem of the NDT approach is its stability when used standalone. As indicated by the authors of previous works, NDT alone has the disadvantage of being unstable depending on the scenario [131]. Therefore, we utilize GNSS data for setting the initial position as well as resetting the NDT localization method when a sudden change in position or orientation estimation is detected. In the results, we refer to this method as NDT+, and a close implementation to the one provided in the existing NDT method has been used [131].

4.3.5 NDT+IMU Localization (NDT++)

The final localization utilized in our experiments consisted on integrating the IMU data into the NDT+ method described above that uses LiDAR and GNSS data. With this approach, we have been able to eradicate the instabilities of the NDT+ method and increase its accuracy.

The algorithm workflow is as follows: first, on system start-up or reset, GNSS data is used in order to obtain an initial estimation of the robot's location. This estimation can be utilized in order to reduce the area of the map in which the NDT matching will be looked for. Second, when the robot starts moving, an unscented Kalman filter that uses IMU data as input serves as an estimation between LiDAR scans. The Kalman filter output is then feeded to the NDT algorithm for scan matching with the predefined map. The GNSS data is still used to avoid instabilities, even though we have not detected any in the dataset utilized.

The NDT+ and NDT++ approaches have an additional benefit over the LiDAR odometry method. In autonomous robots moving in an urban environment, it is essential to react on time to obstacles and to have localization information as frequently as possible. A LiDAR-only approach has the disadvantage of receiving sensor updates at only 10Hz in this case. With IMU readings having a refresh rate of 100Hz,

Table 7. Localization error mean and standard deviation

	$\mu_{rot.}$	σ_{rot}	μ_x	σ_x	μ_y	σ_y
GNSS	1.52	0.79	-0.46	0.22	10^{-4}	0.18
IMU	-1.24	0.62	N/A	N/A	N/A	N/A
LOAM	-0.50	1.88	0.40	0.49	0.10	0.49
NDT+	0.03	0.87	-0.02	1.51	**-0.05**	1.13
NDT++	10^{-3}	**0.31**	**-0.01**	0.10	**-0.05**	**0.10**

the IMU can be utilized to obtain local movement estimation between LiDAR scan matches using the NDT approach. This minimizes the possibilities of instabilities in the NDT algorithm as the matching possibilities are reduced and the goal of the algorithm partly shifts from coarse localization to increasing the accuracy of IMU-based movement estimation.

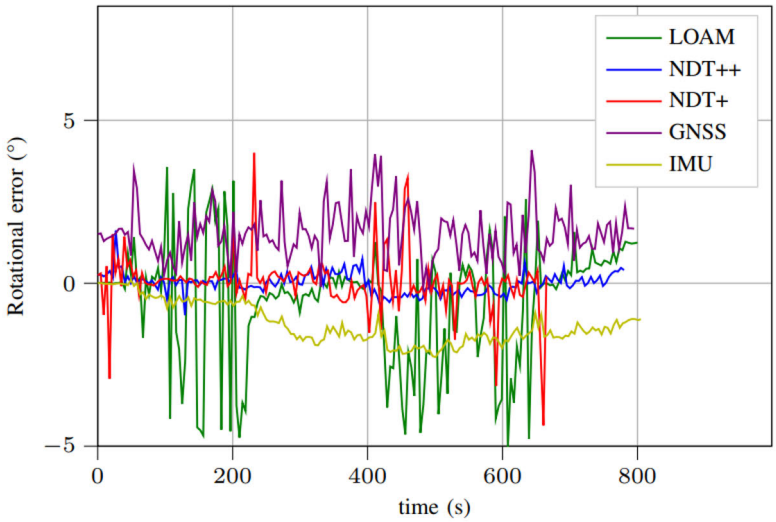
4.4 Corrupted Map Reconstruction

In an urban scenario, it is impractical to propose a localization method that has a high dependency on the existence of an accurate map of the operational area without a strategy for operating in case the map data is corrupted or outdated. In Fig. 24, we show the map of the operation area (on the left, in black and white), with two areas where the data has either been removed completely or noise has been added to render the NDT algorithm unusable. When the robot approaches these areas, we are able to detect them by monitoring the difference between the NDT localization and GNSS and IMU positioning. When part of the map cannot be matched with current scans, we utilize LiDAR odometry and mapping in order to rebuild the corrupted or missing data. The result of this process is shown on the right side of Fig. 24, where one of the corrupted map areas has been restored in real-time while the robot was travelling through it using LiDAR odometry and mapping. Even though it is not visible in the image, there is a relatively small mismatch in the map in the area where the robot emerges again into a mapped environment.

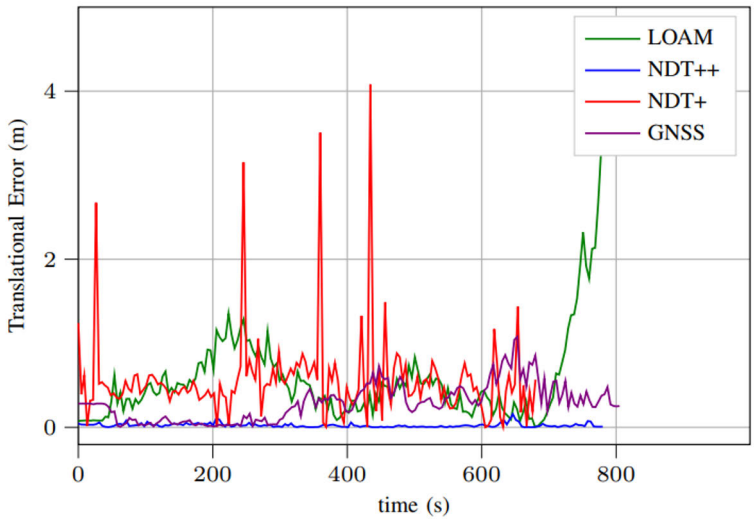
4.5 Experiment and Results

We have applied the five approaches proposed to the given dataset. The results showing the translational and rotational localization and orientation error are shown in Fig. 25b and Fig. 25a, respectively. In these figures, the NDT++ method shows a stable and very small error though time, both in position and rotation estimation. The NDT+ without taking into account inertial data shows a larger error but, more importantly, shows several instabilities that are corrected from the GNSS data.

In order to be able to compare in more detail the different methods, and to see whether there exist some background error or drifting, we show the variability of



(a) Rotational errors of the proposed approaches over time.



(b) Translation errors of the proposed approaches over time.

Figure 25. Rotational and Translation errors of the proposed approaches.

the localization error though two sets of boxplots in Fig. 26a and Fig. 26b. The specific values are also listed in Table 7. We have omitted the location errors of standalone IMU motion estimation as the error is significantly higher than the proposed approaches. However, inertial data is still valuable for local movement estimation and for orientation estimation.

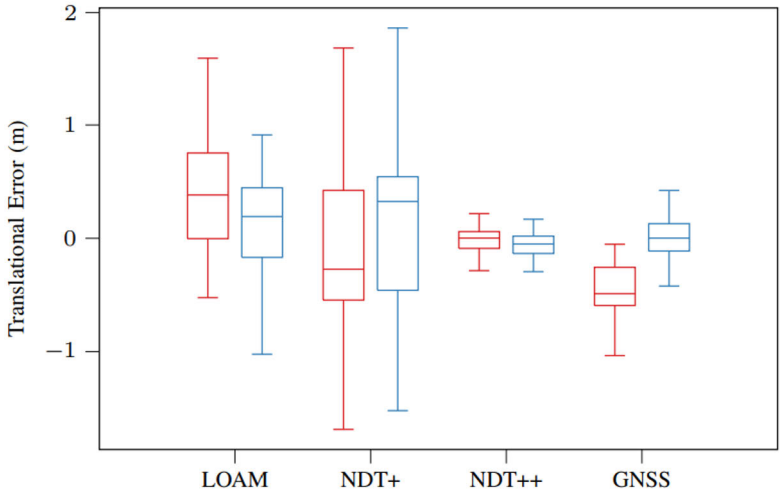
The translational errors are shown in Fig. 26a, where the red boxes show the error in the x coordinate, and the blue boxes refer to the y coordinate error. We have separated the error because in some cases the error mean differs between them. That is the case of the GNSS data, which due to atmospheric or environmental conditions shows a steady negative drift in the x direction. If consistent through large periods of time, it can be assumed that it is due to a sensing error in the device itself, or to environment conditions such a specific multi-path occurring in the operating area. Therefore, this value can be utilized to decrease the sensing error in real time during operation. In order to have a deeper understanding of the distribution of the GNSS error, we show the histogram of the three errors (two translational, one rotational) in Fig. 27. Only the error in the y direction has a mean of 0, while the distribution of the x error is symmetrical and narrow. Therefore, it is possible to fix the drift while keeping the same variance for both components of the translational error. In the case of the rotational error, it is more complex to correct even though the distribution is still symmetric.

From Fig. 26a and Fig. 26b we can see that the most stable methods are NDT++ and GNSS, with the NDT+ method providing accurate results for rotation estimation. However, the NDT+ is highly unstable for position estimation, with the highest variance of all presented approaches. In position estimation, all approaches have a relatively small error after 800 seconds of movement, except for the LOAM method, which error drifts away from 0 towards the end of the available data set. Similarly, the IMU constantly drifts in terms of orientation estimation but it provides a more stable measurement than LOAM, NDT+ or GNSS.

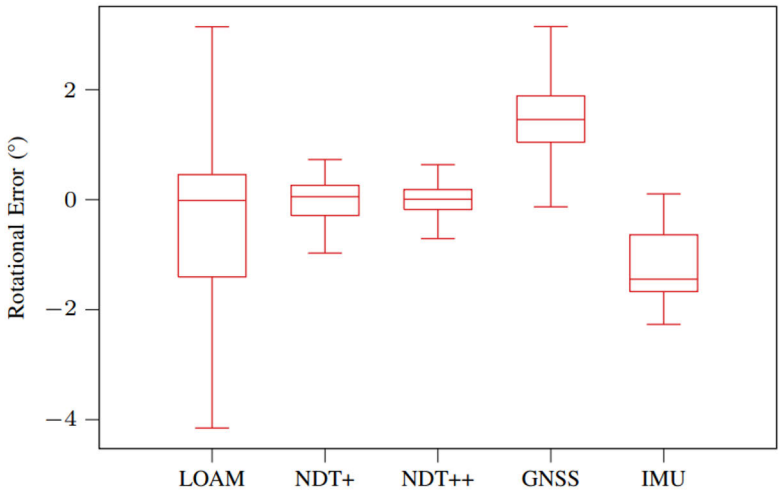
In summary, LiDAR scan matching with a 3D map provides the highest accuracy for localization, both in terms of position and orientation. Nonetheless, it is essential to take into account other sensor data in order to implement a more robust approach that is less prone to instabilities and depends less on the operational environment. GNSS and inertial data are essential for increasing the localization accuracy but also for minimizing the possibilities of unexpected behaviour in the algorithm.

4.6 Summary and Conclusion

Accurate localization in dense urban areas is paramount in order to solve the autonomous last-mile delivery problem. Nonetheless, it still presents important challenges. We have explored the possibilities for localization in a city environment using 3D LiDAR data complemented with GNSS and inertial data using a delivery



(a) Boxplots of translational errors for the different approaches: x-error (red) and y-error (blue).



(b) Boxplots of rotational errors for the different approaches in degrees.

Figure 26. Boxplots of rotational and translational errors with selected localization methods.

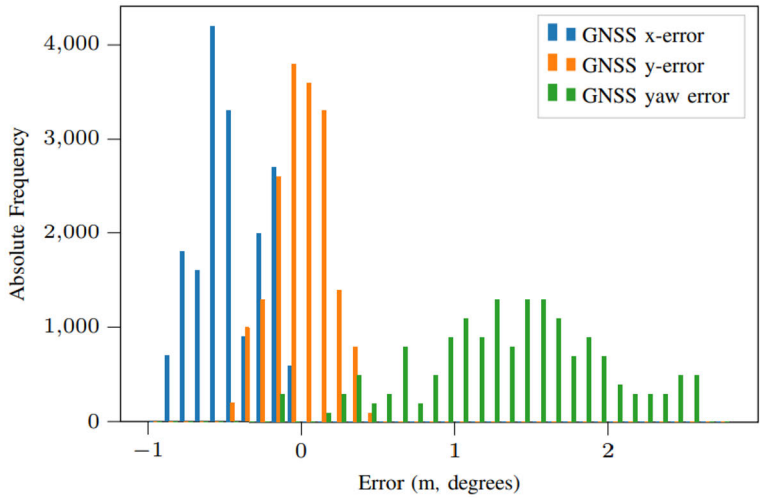


Figure 27. Distribution of GNSS signal errors in position (X, Y) and orientation (Yaw).

robot from JD. We have shown the accuracy of different approaches, assuming that a map of the operation area is given in the form of a point cloud. In addition, we have presented a strategy for situations where the map might be corrupted or the scenario might have undergone significant changes that rendered the map outdated. We have shown that 3D scan matching is the best approach for localization when properly complimented with IMU data within an unscented Kalman filter, and GNSS data.

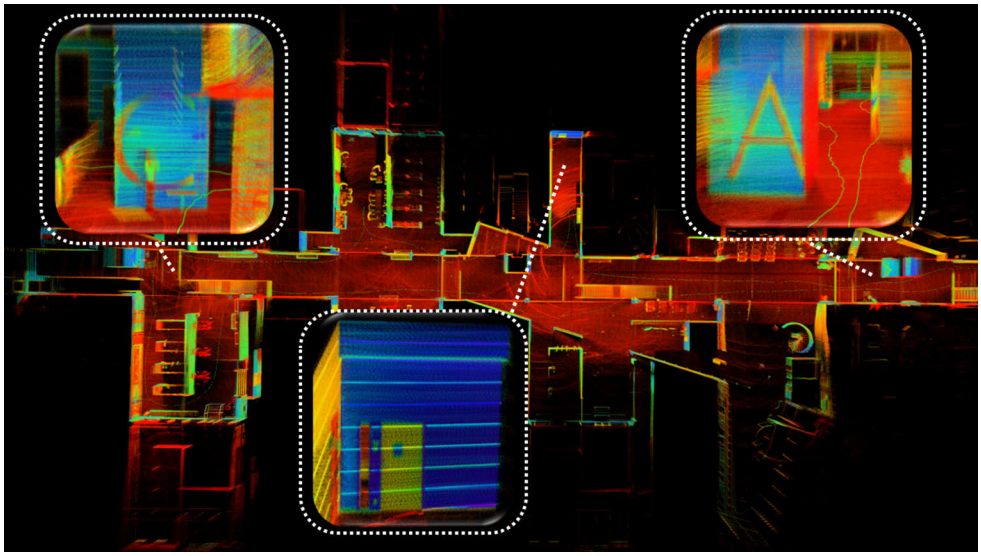
5 Robust Multi-Modal Multi-LiDAR-Inertial Odometry and Mapping

Multiple studies in the literature have concentrated on enhancing LiDAR maps by integrating point clouds from multiple LiDAR sensors, aiming to improve the situational awareness capability [35; 40; 8; 10]. However, the low frame publishing frequency of typical LiDARs (e.g., 10 Hz) can hinder an accurate 6-DOF pose estimation in multi-LiDAR systems. Therefore, IMUs have been widely used in state-of-the-art SLAM systems [41; 16], due to their ability to measure acceleration and angular velocity at a high frequency (e.g., 200 Hz) in three-dimensional space. Nonetheless, there remains a lack of methods that can effectively exploit multi-LiDAR inertial systems for odometry estimations.

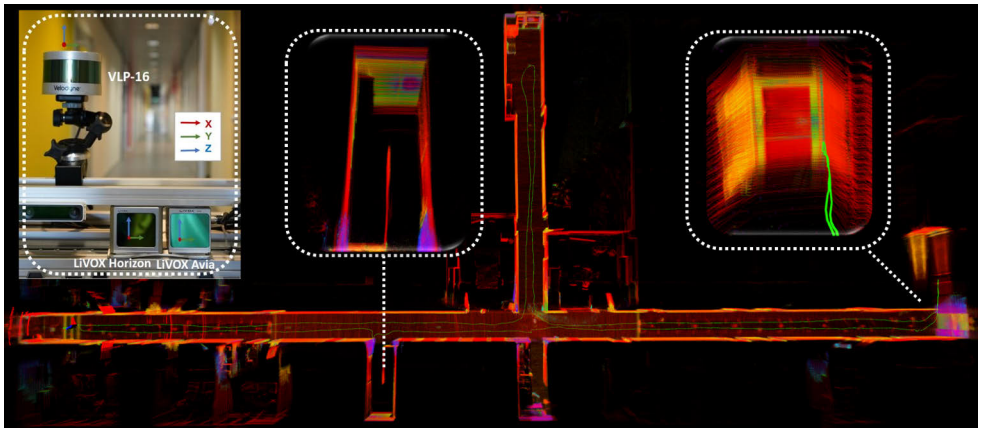
This chapter aim to improve the robustness of the SLAM system and introduce a novel tightly-coupled multi-modal multi-LiDAR-inertial odometry and mapping system, which takes advantage of both the large horizontal FoV from a spinning LiDAR and the dense measurements from a solid-state LiDAR as Table 8 shows. The proposed system first performs spatial-temporal calibration to align the timestamp and calibrate the extrinsic parameters between sensors. Then, we extract two group feature points, edge and planar points, from LiDAR data. Next, with pre-integrated IMU data, an un-distortion module is employed on LiDAR point cloud data. Finally, the un-distorted point cloud is merged into one point cloud and sent to sliding window based optimization module.

This proposed method, to the best of our knowledge, is the first multi-LiDAR-inertial SLAM system able to effectively integrate multiple modal LiDAR sensors with heterogeneous scan modalities within a single estimation and optimization framework. This work is inspired by the limitations found in state-of-the-art algorithms for different LiDAR sensors in our previous works [132; 39], where we show that low-cost solid-state LiDARs outperform high-resolution spinning LiDAR in an outdoor environment, while at the same time perform poorly in indoor environments. The unique characteristics and main contributions of our work can be summarized as follows:

1. Present a complete solution for multi-modal LiDAR spatio-temporal calibration and feature extraction. The method adopts an ICP-based scan-matching



(a) Mapping result with the proposed system at a hall environment. Thanks to the high resolution of solid-state LiDAR with a non-repetitive scan pattern, the mapping result is able to show clear detail of object's surface.



(b) Hardware and Mapping result in long corridor environment. Our proposed methods show robust performance in long corridor environments and survive in narrow spaces where 180° U-turn occurs.

Figure 28. Our proposed methods show high-resolution mapping results and robust performance in challenging environment

approach to obtain the extrinsic parameters, split-and-merge based timestamp alignment, and unified channel based feature extraction for both spinning and solid-state-LiDAR.

2. Design and implementation of a novel tightly-coupled multi-modal multi-LiDAR-inertial mapping framework that is able to combine LiDARs with different

Table 8. Characterization of off-the-shelf LiDAR sensors based on horizontal resolution (H. Res.), vertical resolution (V. Res.), and cost.

LiDAR Types	High H. Res.	High V. Res.	Low-cost
Spinning, 64+ channels	✓	✓	✗
Spinning, 16-32 channels	✓	✗	✓
Solid-State	✗	✓	✓
Ours (solid-state + spinning-16)	✓	✓	✓

scanning modalities and IMU for odometry estimations.

3. The demonstration of a SLAM method for taking advantage of low-cost spinning LiDARs and solid-state LiDARs that outperform the state-of-the-art in high-resolution mapping with high levels of detail.

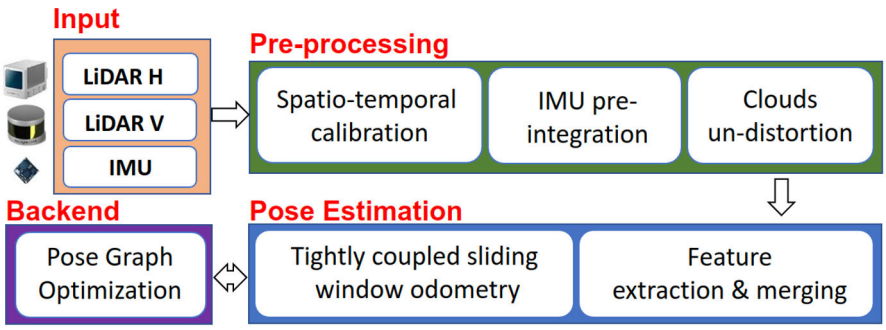


Figure 29. The pipeline of proposed multi-modal LiDAR-inertial odometry and mapping framework. The system starts with preprocessing module which takes the input from sensors and performs IMU pre-integration, calibrations, and un-distortions. The scan registration module extracts features and sent the features to a tightly coupled sliding window odometry. Finally, a pose graph is built to maintain global consistency.

5.1 Multi modal LiDARs and IMU platform

To simplify the system design, we have made follow assumptions:1) The LiDARs are synchronized at the software level. The time offset between sensors is not considered in our system. 2) The extrinsic parameters between IMU and at least one LiDAR are known. In our case, we use LiDAR's built-in IMU and use the extrinsic parameters from factory settings.

The design of our system is based on by our previous work in Chapter 2 where solid-state LiDAR shows significant performance outdoors but failed all tests in-

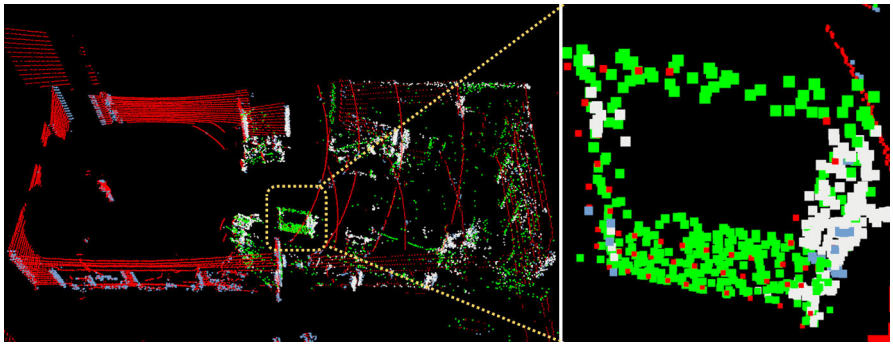


Figure 30. Extracted features points in office room environment(left) and zoom-in view of one gate object (right). Plane and edge feature points from Velodyne are in red and blue, and from Horizon are in green and white separately.

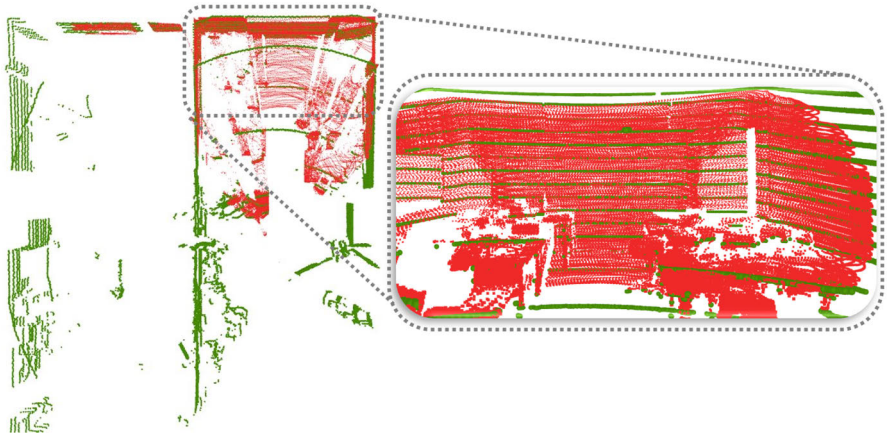


Figure 31. Multi-LiDAR extrinsic parameters calibration result in an office room environment. Red points come from Horizon and green points from VLP-16. Top view (left), and detail of the matching (right).

doors [132]. To combine the high situation awareness ability and robust performance, here we proposed multi-modal LiDAR-inertial odometry and mapping scheme. In this chapter, we consider a perception system consisting of multiple modal LiDARs and IMU, and LiDAR sensors are not triggered by hardware based external clock. The pipeline of the proposed method is illustrated in Figure 29 and the hardware system is shown in Figure 1. The hardware is composed of a spinning LiDAR Velodyne VLP-16, a low-cost solid-state LiDAR Livox Horizon, and its built-in IMU.

5.1.1 System pipeline

The system starts with a data pre-processing module, in which IMU data are pre-integrated, extrinsic parameters between sensors are calibrated, timestamps of clouds with different starting times are aligned, the clouds from multiple LiDARs are undistorted with IMU pre-integrated results. After pre-processing, feature point clouds representing plane and edge points are extracted and merged into one cloud. The merged feature cloud will be sent to the sliding window odometry module where the feature cloud will be matched against the local map. Together with pre-integrated IMU, fixed size of feature clouds, and local map, six-DoF egomotions and IMU parameters are estimated and optimized by keyframe-based sliding window optimization. At the backend, the system maintains a global pose graph with selected keyframes. Loop closure is detected in a keyframe basis graph using ICP, and a global graph optimization is invoked to guarantee the reconstructed map is globally consistent.

5.1.2 Problem Formulation

We treat IMU coordinate as the base local coordinate indicated as $()^b/()^I$. The merged cloud will be transformed to $()^b/()^I$. We use the first keyframe received by the system as the origin of the world coordinates denoted as $()^w$. The coordinate of spinning LiDAR is denoted as $()^v$, and the coordinates of solid-state LiDAR are denoted as $()^h$. We use $\mathcal{P}^v_{t_1} = \{\mathbf{p}^v_1, \mathbf{p}^v_2, ...\mathbf{p}^v_n\}$ be the point cloud acquired at time t_1 with spinning LiDAR, and $\mathcal{P}^h_{t_2} = \{\mathbf{p}^h_1, \mathbf{p}^h_2, ...\mathbf{p}^h_n\}$ be the cloud acquired at time t_2 with solid state LiDAR, where $\mathbf{p}^v_i$ and $\mathbf{p}^h_j$ are a point in $\mathcal{P}^v_{t_1}$ and $\mathcal{P}^h_{t_2}$.

We denote $\mathbb{F}_{Ek}$ and $\mathbb{F}_{Pk}$ as the edge and plane feature point cloud extracted from the LiDARs' data at time k. The transformation matrix is denoted as $\mathbf{T}^b_a \in SE(3)$, which transforms a point from frame $()^a$ into the frame $()^b$. $\mathbf{R}^b_a \in SO(3)$ and $\mathbf{t} \in R^3$ are the rotation matrix and the translation vector of $\mathbf{T}^b_a$ respectively. The quaternion $\mathbf{q}^b_a$ under Hamilton notation [133] is used, which corresponds to $\mathbf{R}^b_a$. $\otimes$ is used for the multiplication of two quaternions. $\mathbf{q}^b_a$ and $\mathbf{R}^b_a$ can be converted by Rodrigues formula [134]. With a given point cloud from multi-modal LiDAR sensor and IMU info, the state needs to be optimized for keyframe k is defined as (10) where $\mathbf{t}_k$ is the translation vector, $\mathbf{q}_k$ represents orientation in quaternion, $\mathbf{v}_k$ is the velocity, $\mathbf{b}_{a_k}$ and $\mathbf{b}_{g_k}$ are the bias vector of the accelerator and gyroscope.

$$\mathbf{X}_k = [\mathbf{p}_k, \mathbf{q}_k, \mathbf{v}_k, \mathbf{b}_{a_k}, \mathbf{b}_{g_k}] \in \mathbb{R}^3 \times \mathbb{S}^3 \times \mathbb{R}^3 \times \mathbb{R}^3 \times \mathbb{R}^3 \tag{10}$$

5.1.3 Spatial-temporal Calibration and Initialization

As the extrinsic parameters between IMU and one LiDAR is known, so here we focus on calibrating the extrinsic parameters between two LiDARs. We assume the sensor

platform is stationary during the extrinsic calibration process. As solid-state LiDAR with the non-repetitive pattern is able to obtain more details from the environment within the FoV, therefore, we integrated n consecutive frames(e.g., ten) to new point cloud $\mathcal{P}^h_{t_1 \sim t_n}$ for the calibration process. Let $\mathcal{P}^v_{t_m}$ be one cloud data obtained at $t_m \in (t_1, t_n)$ from spinning LiDAR. Generalized Iterated Closest Point (GICP)[12] method is employed to caculate the relative transformation matrix $\mathbf{T}^h_v$ between $\mathcal{P}^h_{t_1 \sim t_n}$ and $\mathcal{P}^v_{t_m}$ from the overlapped region. The extrinsic calibration results with data collected from a classroom environment are shown in Figure 31. As the transformation matrix $\mathbf{T}^i_h$ between IMU and one LiDAR given by the factory, we can get $\mathbf{T}^i_v$ by $\mathbf{T}^i_v = \mathbf{T}^h_v * \mathbf{T}^i_h$.

We consider a system where LiDARs are not triggered with an external clock (e.g., GNSS) as discussed in [35]. Each cloud $\mathcal{P}^h$ and $\mathcal{P}^v$ is collected at different starting timestamps. To merge these clouds into one combined cloud $\mathcal{P}^m$, it is necessary to align the starting and ending timestamps. We adopt a split-and-merge method similar to [40]. The individual timestamp of $p^h_i \in \mathcal{P}^h$ and $p^v_i \in \mathcal{P}^v$ can be obtained from the sensor's driver. If the timestamp for a point $p^v_i \in \mathcal{P}^v$ is not available, it can be calculated using orientation difference as described in [13]. When a new cloud $\mathcal{P}^h_{t_k}$ is received at time t_k, we put all points $p^v_i \in \mathcal{P}^h_{t_k}$ into a queue $\mathbb{Q}^h$ ordered by timestamp. When a new cloud $\mathcal{P}^v_{t_m}$ is received at time t_m, we first obtain its start time t_{m_s} and end time t_{m_e}. Then, all points in $\mathbb{Q}^h$ with timestamp $t_i < t_{m_s}$ will be dropped, and points with timestamps $t_i \in (t_{m_s}, t_{m_e})$ will be moved to a new frame $\mathcal{P}^h_{t_m}$, which shares the same time domain with $\mathcal{P}^v_{t_m}$.

5.1.4 IMU Initialization and Preintegration

The IMU sensor output angular velocity and acceleration measurements are defined as $\tilde{\omega}_t$ and $\tilde{\mathbf{a}_t}$ using equations. 11 and 12:

$$\boldsymbol{\omega}_t = \boldsymbol{\omega}_t + \mathbf{b}^{\boldsymbol{\omega}}_t + \mathbf{n}^{\boldsymbol{\omega}}_t \tag{11}$$

$$\tilde{\mathbf{a}_t} = \mathbf{R}^{WL}_t(\mathbf{a}_t - \mathbf{g}) + \mathbf{b_t}^{\mathbf{a}} + \mathbf{n_t}^{\mathbf{a}} \tag{12}$$

Where $\mathbf{b}_t$ is the measurement bias and $\mathbf{n}_t$ is white noise. $\mathbf{R}^{WL}_t$ is the rotation matrix from World coordinate $()^W$ to local coordinate $()^L$, $\mathbf{g}$ is the gravity vector in world coordinate. Based on the raw measurement $\boldsymbol{\omega}_t$ and $\tilde{\mathbf{a}_t}$, we can infer the motion of the robot as follows:

$$\begin{aligned} \mathbf{p}_{t+\Delta t} = \mathbf{p}_t + \mathbf{v}_t \Delta t + \frac{1}{2}\mathbf{g}\Delta t^2 + \\ \frac{1}{2}\mathbf{R}_t(\hat{\mathbf{a}}_t - \mathbf{b}^{\mathbf{a}}_t - \mathbf{n}^{\mathbf{a}}_t)\Delta t^2 \end{aligned} \tag{13}$$

$$\mathbf{v}_{t+\Delta t} = \mathbf{v}_t + \mathbf{g}\Delta t + \mathbf{R}_t(\hat{\mathbf{a}}_t - \mathbf{b}^{\mathbf{a}}_t - \mathbf{n}^{\mathbf{a}}_t)\Delta t \tag{14}$$

$$\begin{aligned}\mathbf{q}_{t+\Delta t} &= \mathbf{q}_t \otimes \mathbf{q}_{\Delta t} \\ &= \mathbf{q}_t \otimes \begin{bmatrix} exp(\frac{1}{2}\Delta t(\tilde{\boldsymbol{\omega}}_t - \mathbf{b}_t^{\boldsymbol{\omega}} - \mathbf{n}_t^{\boldsymbol{\omega}})) \\ 1 \end{bmatrix}\end{aligned} \tag{15}$$

Where the $\mathbf{p}_t, \mathbf{v}_t$ and $\mathbf{q}_t$ are the estimated position, velocity and orientation in quaternion at time t, $\mathbf{p}_{t+\Delta t}, \mathbf{v}_{t+\Delta t}$ and $\mathbf{q}_{t+\Delta t}$ are the estimated state at time $t + \Delta t$. We apply the IMU preintegration method proposed in [41]. The relative motion between two timestamp $\Delta\mathbf{V}, \Delta\mathbf{P}, \Delta\mathbf{Q}$ can be calculated based on equations 13∼15 and will be used for initial guess in 5.2.3.

5.2 Multi-modal LiDAR Pose Estimation

5.2.1 Feature Extraction

For computing efficiency, feature extraction is essential for the SLAM system. We focus on extracting the general features that exist in different modal LiDARs and can be shared in the optimization process. Here we extract feature points based on [16] that selects a set of feature points from measurements according to their continuous and surface normal vector. We extend the method and adapt it to both spinning and solid-state LiDARs. The set of extracted features consists of two subsets: plane points $\mathbb{F}_P$ and edge points $\mathbb{F}_E$. By checking the continuity, the edge feature included two types of points $\mathbb{F}_{E_l}$ and $\mathbb{F}_{E_b}$ where $\mathbb{F}_{E_l}$ represents the line feature where two surface meets, and $\mathbb{F}_{E_b}$ represents breaking points where plane end.

Let $\mathcal{P}_t^v$ be the point cloud acquired at time t from spinning LiDAR, $\mathcal{P}_t^h$ be the point cloud acquired from solid-state LiDAR at the same time domain after temporal alignment. If the channel number of each point $p_v \in \mathcal{P}_t^v$ is unavailable, then first project points in $\mathcal{P}_t^v$ onto a range image based on the horizontal and vertical angle w.r.t. the origin. Each row represents data from one channel of the spinning LiDAR. Then the points are divided into N subsets $\{\mathbf{L}_i^v\}_{i\in N}$ where N is the total channel numbers of spinning LiDAR. For point cloud $\mathcal{P}_t^h$, we divide the point based on line number which can be obtained from Livox ROS driver [1]. Similarly, we first divided the points into M subsets $\{\mathbf{L}_i^h\}_{i\in M}$ where M is the total line numbers of solid-state LiDAR. The points in each $\mathbf{L}_{i\,i\in N}^v$ and $\mathbf{L}_{i\,i\in M}^h$ are ordered by timestamp. For each subset $\mathbf{L}_i^v$ in $\{\mathbf{L}_i^v\}_{i\in N}$ and $\mathbf{L}_j^h$ $\{\mathbf{L}_j^h\}_{j\in M}$, we first extract the continues points $\mathcal{P}_{i\,C}^v$ and $\mathcal{P}_{j\,C}^h$ by checking the depth difference with its neighbour points. If the depth difference between the point in $\mathbf{L}_i^v$ or $\mathbf{L}_j^h$ and nearest neighbor points within the same subset is smaller than the depth threshold d_{th}, then the point is added to continuous points subset $\mathcal{P}_{i\,C}^h$ or $\mathcal{P}_{j\,C}^v$. Then we follow the feature extraction methods in [40], where a scatter matrix $\boldsymbol{\Sigma}$ is calculated based on neighbor points. By analyzing the

[1] `https://github.com/Livox-SDK/livox_ros_driver`

two largest eigenvalues λ_1 and λ_2 of $\mathbf{\Sigma}$, the plane points are detected and labeled as a plane, the point where two plane meet is labeled as corner features, the points where one plane ends and neighboring with discontinuous points are labeled as break points. We merge the corner points and break points together and use them as edge feature points. After this process, plane feature points $\mathbb{F}^h_{Pt}$ and $\mathbb{F}^v_{Pt}$, edge feature points $\mathbb{F}^h_{Et}$ and $\mathbb{F}^v_{Et}$ are extracted. We show extracted feature points in the office room environment in Figure 30

5.2.2 Feature Clouds Merging

Instead of keeping two different system state $\mathbf{X}^w_t$ for clouds $\mathcal{P}^h_t$ and $\mathcal{P}^v_t$, we merge the same type of feature point cloud into one frame and use a single system state. As the cloud $\mathcal{P}^h_t$ from solid-state LiDAR can be more easily blocked by near objects, with extreme cases leading to all points reflecting on a single plane (e.g., a close wall) in an indoor environment, this means that there is a certain probability of most of the points in $\mathbb{F}^h_{Pt}$ belonging to a single plane. This means that insufficient $\mathbb{F}^h_{Et}$ points can be extracted. In this case, we treat the $\mathcal{P}^h_t$ as *bad frame*, with the corresponding feature cloud not being considered to the union feature cloud $\mathbb{F}^i_{Et}$ and $\mathbb{F}^i_{Pt}$. This ensures more consistent behavior and robust estimation across environments and over time. To detect such *bad frames*, We first remove the points in $\mathbb{F}^h_{Pt}$ and $\mathbb{F}^h_{Et}$ that close to the origin of the sensor (e.g., 2 m threshold), and then check the amount n_e of edge point in the feature cloud $\mathbb{F}^h_{Et}$. If n_e is smaller than the edge feature threshold τ_e (e.g., 100 in our experiments), then the cloud $\mathcal{P}^h_t$ is treated as a *bad frame*. If no such *bad frame* is detected, we transform the complete feature clouds $\mathbb{F}^h_{Pk}$, $\mathbb{F}^v_{Pt}$, $\mathbb{F}^h_{Et}$ and $\mathbb{F}^v_{Et}$ to the $()^i$ coordinate frame and merge them to fused, unified feature clouds $\mathbb{F}^i_{Ek}$ and $\mathbb{F}^i_{Pk}$ using the extrinsic transformation matrices $\mathbf{T}^i_v$ and $\mathbf{T}^i_h$, calculated as described in Section. 5.1.3 with Eq. (16).

$$\mathbb{F}^i_E = \mathbf{T}^i_v * \mathbb{F}^v_E + \mathbf{T}^i_h * \mathbb{F}^h_E, \mathbb{F}^i_P = \mathbf{T}^i_v * \mathbb{F}^v_P + \mathbf{T}^i_h * \mathbb{F}^h_P. \tag{16}$$

If $\mathbb{P}^h_t$ is "bad frame", then $\mathbb{F}^i_E = \mathbf{T}^i_v * \mathbb{F}^v_E$ and $\mathbb{F}^i_P = \mathbf{T}^i_v * \mathbb{F}^v_P$. The union feature clouds $\mathbb{F}^i_E$ and $\mathbb{F}^i_P$ will be down-sampled before sending them into sliding window optimization module.

5.2.3 Keyframe Selection & Undistortion

Given feature point cloud and preintegrated IMU within the same time domain $\mathbb{F}^i_{Ek}, \mathbb{F}^i_{Pk}, \mathbb{I}^i_{preg_k}$ and a point cloud feature map $\mathbb{M}^w_k$ in world coordinate, the registration problem can be formulated as solving a non-linear least square problem. The initial guess of the state $\mathbf{X}^w$ is estimated with Eq. (17):

$$\tilde{\mathbf{p}}_k = \bar{\mathbf{p}}_{k-1} + \bar{\mathbf{q}}_{k-1} * \Delta\mathbf{P}^k_{k-1}$$

$$\tilde{\mathbf{q}}_k = \bar{\mathbf{q}}_{k-1} * \Delta\mathbf{Q}_{k-1}^k, \tilde{\mathbf{b}_{\mathbf{g}_k}} = \bar{\mathbf{b}_{\mathbf{g}_{k-1}}} \tag{17}$$
$$\tilde{\mathbf{v}}_k = \bar{\mathbf{v}}_{k-1} + \bar{\mathbf{q}}_{k-1} * \Delta\mathbf{V}_{k-1}^k, \tilde{\mathbf{b}_{\mathbf{a}k}} = \bar{\mathbf{b}_{\mathbf{a}k-1}}$$

As sliding window optimization is a relatively heavy process, therefore, maintain the sparsity of the frames in the window can significantly affect the real-time performance. Here we check the IMU drift during the time interval of two consecutive keyframes, and select the frame as keyframe if the orientation difference is higher than a certain degree (e.g., 30°) or the time difference between a current frame and the last keyframe larger then certain time (e.g., 2 seconds). Then each point in the selected keyframe will be un-distorted with ΔQ and ΔP provided by IMU pre-integration. Each keyframe contains deskewed feature clouds $\mathbb{F}_{Ek}^i$ and $\mathbb{F}_{Pk}^i$, pre-integrated IMU $\mathbb{I}_{preg_k}^i$, and initial guess of state $\tilde{\mathbf{X}}_k^w \sim [\tilde{\mathbf{p}}_k, \tilde{\mathbf{q}}_k, \tilde{\mathbf{v}}_k, \mathbf{b}_{ak}, \tilde{\mathbf{b}_{g_k}}]$ which will be optimized by sliding window optimization.

5.2.4 Sliding Window Optimization

In this work, we follow keyframe based tightly coupled LiDAR-inertial sliding window optimization strategy in [16]. The merged feature points $\mathbb{F}_{Ek}^i$ and $\mathbb{F}_{Pk}^i$ of keyframe k are treated as feature clouds that are extracted from single LiDAR sensor as in [16]. We build a window with τ consecutive keyframes where the states that need to be optimized for each frame are $\tilde{\mathbf{X}^w} = [\tilde{\mathbf{X}}_1^w, \tilde{\mathbf{X}}_2^w, ..., \tilde{\mathbf{X}}_\tau^w]$. The optimal state can be obtained by minimizing the function:

$$\min_{\tilde{\mathbf{X}}} \{||\mathbb{D}_{prior}(\tilde{\mathbf{X}^w})||^2 + \sum_{k=1}^{\tau} \mathbb{D}_L(\tilde{\mathbf{X}_k^w}) + \sum_{k=1}^{\tau} \mathbb{D}_I(\tilde{\mathbf{X}_k^w})\} \tag{18}$$

Where $||\mathbb{D}_{prior}(\tilde{\mathbf{X}^w})||^2$ represents the prior residual term which is generated by marginalizing oldest frames before the current window via Schur-complement [41],$\mathbb{D}_I(\tilde{X^w})$ represents the pre-integrated IMU terms as defined in [16].$\mathbb{D}_L(\tilde{\mathbf{X}_k^w})$ is LiDAR term defined as (19).

$$\sum_{a=1}^{m} (\mathbb{D}_e(\mathbf{X}_k^w, \mathbf{p}_{k,a}^i, \mathbb{M}_k^w))^2 + \sum_{b=1}^{n} (\mathbb{D}_s(\mathbf{X}_k^w, \mathbf{p}_{k,b}^i, \mathbb{M}_k^w))^2 \tag{19}$$

$\mathbb{D}_e$ is the point-to-edge residual term defined as (20) and $\mathbb{D}_s$ point-to-plane residual term defined as (21).

$$\mathbb{D}_e(\mathbf{X}^w, \mathbf{p}^i, \mathbb{M}^w)) = \frac{||(\mathbf{p}^w - \grave{\boldsymbol{e}}^{\boldsymbol{w}}) \times (\mathbf{p}^w - \grave{\boldsymbol{e}}^{\boldsymbol{w}})||}{||\acute{\boldsymbol{e}}^{\boldsymbol{w}} - \grave{\boldsymbol{e}}^{\boldsymbol{w}}||} \tag{20}$$

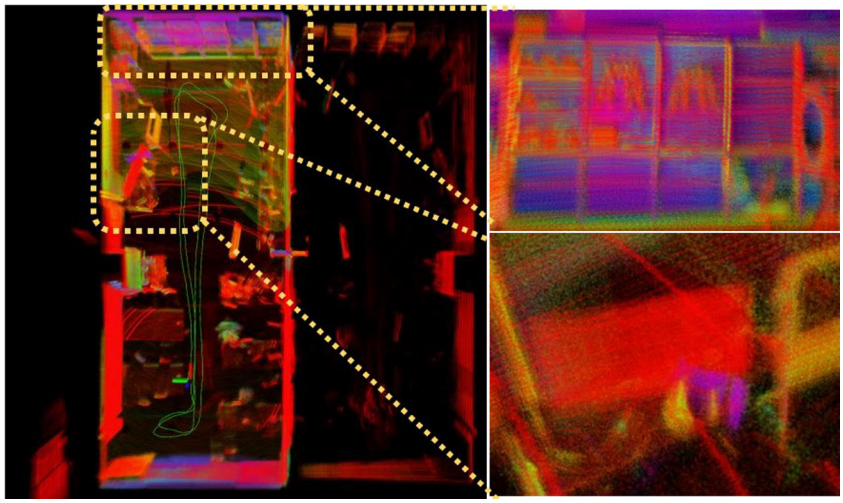
$$\mathbb{D}_s(\mathbf{X}^w, \mathbf{p}^i, \mathbb{M}^w)) = |\mathbf{n}_s^T \mathbf{p}^w + 1/||\mathbf{n}_s|||| \tag{21}$$

where $\mathbf{p}^i$ represents a feature point belonging to $\mathbb{F}_{Ek}, \mathbb{F}_{Pk}$. Then, $\mathbf{p}^w = \mathbf{R(q)p}^i + \mathbf{t}$ represents the scan point $\mathbf{p}^i$ at local frame $()^i$, which is transformed to world frame $()^w$ given the state estimation $[\mathbf{q}, \mathbf{t}]$ in $\mathbf{X}^w$. We denote by $\grave{\boldsymbol{e}}^{\boldsymbol{w}}$ and $\acute{\boldsymbol{e}}^{\boldsymbol{w}}$ the two closest corresponding edge feature points on the feature map $\mathbb{M}^w$, while $\mathbf{n}_s^w$ is the plane normal vector that is calculated by neighbor plane feature points in the $\mathbb{M}^w$ cloud. We solve the non-linear Eq. (18) using the Ceres Solver toolbox [135]. To ensure global consistency, we also maintain a pose-graph structure with optimized states $\mathbf{X}^w$ and pre-integrated IMU measurements as optimization constraints.

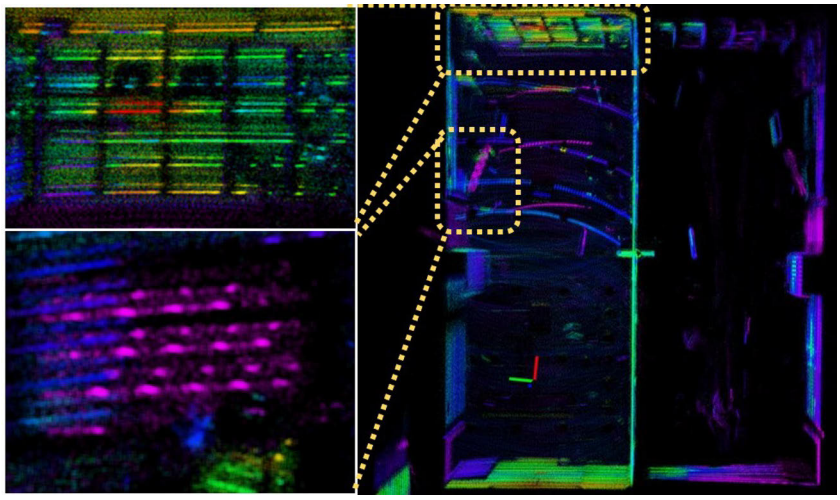
5.3 Experimental Evaluation

5.3.1 Sensor Configuration and Implementation

We implement the proposed multi-modal multi-LiDAR-inertial odometry and mapping system in C++ with ROS Melodic environment that can be shared within the robotic community. The system shown in Figure 29 is structured in four nodes: preprocessing, feature extraction, scan registration, and graph optimization. The factor graph optimization is maintained by GTSAM 4.0 [136], and non-linear optimization is performed by Ceres Solver 2.0 [135]. The framework proposed in this work is validated using datasets gathered by Velodyne VLP-16 (V), Livox Horizon (H) 3D LiDAR, and its built-in IMU (I). The VLP-16 measurement range is up to 100 m with an accuracy of ± 3 cm. It has a vertical FoV of 30°(± 15°) and a horizontal FoV of 360°. The 16-channel sensor provides a vertical angular resolution of 2°and The horizontal angular resolution varies from 0.1° to 0.4°. For solid-state LiDAR, we selected Livox Horizon, which is designed with an FoV of 81.7° × 25.1°. Horizon was scanning at 10 Hz and reaches a similar but more uniform FoV coverage compared with typical 64-line mechanical LiDARs. The extrinsic parameters between Horizon and its built-in IMU is provided by factory instruction. The sensors are connected to a laptop directly with Ethernet and synchronized with software-based precise timestamp protocol (PTP) [137]. We run ROS drivers for Velodyne and Horizon and recorded the data in rosbag format.



(a) Map results generated by our methods in HVI mode.



(b) Map results generated by Fast_LIO in VI mode.

Figure 32. Qualitative comparison of map details in the office room dataset sequence. The color of the points represents the reflectivity provided by raw sensor data. The point size is 1 cm^3, and transparency is set to 0.05. The middle two columns show the zoom-in view of the wall (top) and TV (bottom).

5.3.2 Qualitative Experiment

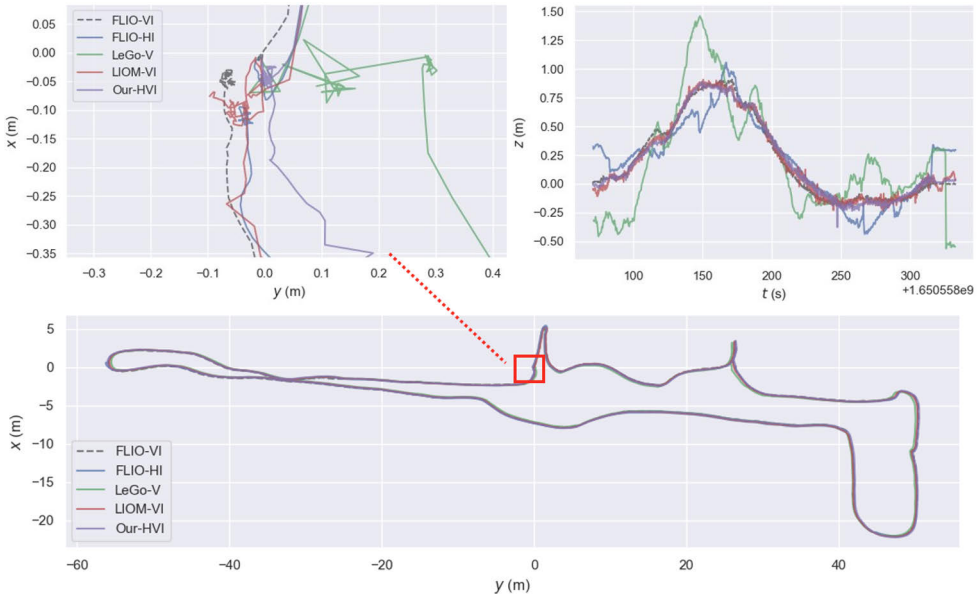


Figure 33. The trajectory result on dataset Hall. Our proposed methods show the smallest error when returning to the start point. The trajectory from different methods (bottom), the zoom-in view of starting and ending point(top left), the changes along Z-axis(top right)

From our previous research [132; 39], a tightly coupled solid-state LiDAR-inertial system shows competitive performance outdoors but performs poorly in indoor environments. Therefore, here we aim to compare our proposed system with a typical and challenging indoor environment: an office room, a long corridor, and a large hall. The data are gathered with the platform as Figure 1 shows at ICT-City in Turku, Finland.

We compare our proposed method with several state-of-the-art SLAM algorithms: LeGO-LOAM [14] [2], Fast-LIO [17] [3] and LILI-OM [16] [4]. LeGO-LOAM is a LiDAR-only odometry, while Fast-LIO and LILI-OM are tightly coupled LiDAR inertial odometry systems capable of working with both solid-state LiDARs and spinning LiDARs. Fast-LIO features a tightly-coupled iterated extended Kalman filter framework and an iKD-tree data structure, which demonstrate efficient and robust performance [17]. Similar to our proposed method, LILI-OM employs keyframe-based sliding window optimization but only fuses single LiDAR and pre-integrated IMU measurements. Like Fast-LIO, during the experiments, we utilize the default

[2] https://github.com/RobustFieldAutonomyLab/LeGO-LOAM

[3] https://github.com/hku-mars/FAST_LIO

[4] https://github.com/KIT-ISAS/lili-om

configurations from the official GitHub repository, and loop closure detection is turned off for each method. To compare the odometry accuracy, all three datasets started and ended at the same place. The mean square distance (MSE) between the starting and ending positions is treated as the error. The results generated by the selected methods for all datasets are presented in Table 9.

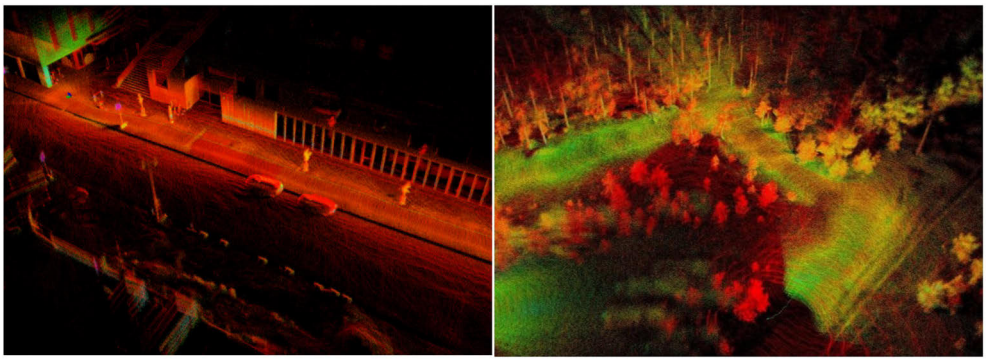


Figure 34. Mapping results with the proposed method outdoors: urban street (left) and forest environment (right).

Hall

The data was recorded in a hall environment measuring approximately 127 m $\times$ 35 m, aimed at comparing odometry and mapping performance in a relatively large indoor environment. The recording commenced in a narrow space and proceeded with a 180°U-turn where most of the field of view (FoV) of the solid-state LiDAR was covered by nearby walls. Consequently, solid-state LiDAR-only based methods may not receive enough features, leading to potential significant drift. The trajectory is depicted in Figure 33, and mapping results are shown in Figure 28a. According to the position error displayed in Table 9, our proposed methods exhibit the best performance, achieving a position error of 5.1 cm, with Fast-LIO (VI) showing the second-best performance.

Corridor

The corridor environment is another challenging environment as low-resolution LiDARs might not get enough feature points from the environment to perform robust localization. The data sequence was recorded at a 60 m long corridor. The mapping results are shown in Figure 28b. Fast-LIO (VI) performs the best, while our proposed method follows closely in performance.

Office

Another data sequence is recorded in a small office room measuring $12, m \times 3.7, m$. To make the mapping task more challenging in this environment, we performed several fast 180°U-turns during the data recording. From the results, we can see that LIOM (VI) exhibits the best performance, while our proposed method with HVI ranks second.

5.3.3 Outdoor Mapping

For the sake of completeness, we also test the proposed method with an outdoor data sequence on a city road and a forest environment. The resulting map is shown in Figure 34. A qualitative analysis of the map point cloud shows a high level of detail. However, the extrinsic calibration method has been designed for indoor environments with rich edge and planar features within the overlapped FoV between the two LiDAR sensors; therefore, the extrinsic parameters are not well calibrated in the forest environment, which leads to a decrease in sharpness in the final map. In any case, this demonstrates the potential for generalization to more unstructured environments and enables high-density mapping even with sub-optimal extrinsic parameters calibration

Our proposed multi-modal multi-LiDAR-inertial method demonstrates competitive and consistent performance compared to other selected methods. However, we observed that our method did not always outperform other approaches, despite utilizing more LiDAR data from the environment. One possible reason for this could be inaccurate time synchronization between the sensors, our time synchronization between LiDARs is on the software level with sub-microsecond accuracy. The inaccurate timestamp for each point will bring the error to the system during the cloud undistortion and cloud merging steps, which is hard to eliminate.

From the results, the VI-based method is capable of tracking the position of the sensor in all dataset sequences. However, the HI-based methods fail in most of the sequences, except for Fast_LIO(VI) in the large hall environment. To understand the difference between each LiDAR, we also tested our methods with Velodyne-Inertial odometry. The results indicate that the accuracy of the proposed VI method is less accurate than the HVI method, which suggests that horizon LiDAR can improve the system's performance. It is also worth noting that our focus is on integrating various multi-modal LiDAR sensors at the point cloud registration and feature extraction stages while aiming for a more consistent framework across environments.

Table 9. End-to-end position error in meters (N/A when odometry estimations diverge; V: Velodyne VLP-16, H: Livox Horizon, I: IMU). Numbers in bold indicate the best performance, while underscored numbers indicate the second best in each environment.

Dataset	Hall	Corridor	Office
LeGo (V)	0.567	0.336	0.127
FLIO (HI / VI)	0.109 / <u>0.069</u>	N/A / **0.062**	N/A / 0.188
LIOM (HI / VI)	N/A / 0.736	N/A / 1.951	NA / **0.102**
Ours (HI / VI)	N/A / 0.107	N/A / 0.132	NA / 0.165
Ours (HVI)	**0.051**	<u>0.085</u>	<u>0.124</u>

Table 10. Analysis of processing time (ms) for the different algorithm stages on an Intel i7-10875H CPU.

	Hall	**Corridor**	**Office**
Pre-processing stage	4.14	4.44	4.37
Multi-LiDAR feature extraction	80.32	86.74	94.21
Pose estimation, optimization	139.71	123.51	132.73

5.3.4 Mapping Quality Comparison

One of the key benefits of the multi-LiDAR system is its high perception awareness ability. Here we compare the mapping quality in terms of resolution. Part of the mapping results by our proposed methods has shown in Figure. 28a and 28b where the color represents intensity value. From the result, we can see many objects (e.g., door, wall letters). We compare the mapping result between our proposed method in Figure 32a and Fast_LIO (VI) in Figure 32b. Our method shows the most stable performance in experiments in an office room environment. By zooming in the same area, we can see a more uniform point cloud from the wall and a TV in the map generated with our method.

5.3.5 Runtime Analysis

Our evaluations were conducted on a laptop with an Intel Core i7-10875H CPU and 64 GB RAM on Unbuntu 18.04.6 LTS system. We show the average runtime per frame in Table 10 and feature numbers in Table 11. Preprocessing and feature extraction are lightweight. Runtime is dominated by the sliding-window-based pose estimation and optimization.

5.4 Summary and Conclusion

We present in this chapter a tightly coupled multi-modal multi-LiDAR-inertial odometry and mapping framework with sliding window optimization for pose estimation and dense mapping. To our knowledge, this is the first SLAM algorithm that har-

Table 11. Average number of feature points per frame extracted from the different LiDARs in the three tested environments. Only points in a range between 2 m, and 50 m are considered.

	V-raw	V-edge	V-plane	H-raw	H-edge	H-plane
Hall	17370	201	15680	21109	405	1934
Corridor	5965	101	5493	14623	302	4007
Office	20235	329	18430	17462	409	1029

nesses the benefits of both spinning LiDARs and solid-state LiDARs within a unified framework. Our primary focus is to demonstrate the feasibility of achieving high-robustness odometry and producing high-quality maps by utilizing an optimal combination of cost-effective sensors. The proposed system effectively integrates the advantages of dense point clouds generated by solid-state LiDAR for enhanced situational awareness and the larger Field of View (FoV) offered by spinning LiDARs. Although our odometry accuracy in some testing environments may similar to other methods, our approach excels in consistency across varying environments. Notably, it produces higher-quality maps capable of capturing intricate environmental details.

6 Dynamic Object Detection and Tracking

Object detection and tracking play a significant role in the field of robotic. The object detection and tracking can benefit for static mapping by removing moving objects in SLAM area [138], novel human-robot interaction algorithm [139; 140] and multi-robot coorperation tasks [141]. This enable robots to interact with the world at an object level rather than dealing with pixel-level information in images or individual points in point cloud data. In this chapter, we mainly discussed two problems in tracking. First, we explored human detection and tracking with deep learning methods in Search and Rescue (SAR) tasks with a camera mounted on UAV, enhancing the robot's ability to locate and monitor human subjects effectively. Second, we conducted experiments using novel LiDAR technology to track small challenging objects such as micro UAVs for multi-robot cooperation purpose, highlighting the versatility of LiDAR sensors in object tracking scenarios.

UAVs have been playing an increasingly active role in supporting SAR operations in recent years. The benefits are multiple such as enhanced situational awareness, status assessment, or mapping of the operational area through aerial imagery. Most of these application scenarios require the UAVs to cover a certain area. If the objective is to detect people or other objects, or analyze in detail the area, then there is a trade-off between speed (higher altitude coverage) and perception accuracy (lower altitude). An optimal point in between requires active perception on-board the UAV to dynamically adjust the flight altitude and path planning. As an initial step towards active vision in UAV search in maritime SAR scenarios, in this chapter we analyzed how the flight altitude affects the performance of object detection algorithms.

From the perspective of deployment within multi-robot systems, being able to track UAVs from ground robots, e.g., UGVs, enables miniaturization and higher degrees of flexibility relaxing the need for high-accuracy onboard localization. A recent and significant example of multi-robot system deployment in GNSS-denied environments is the DARPA Subterranean challenge [142; 143]. Reports from participating teams indicate that localization and collaborative sensing were among the key challenges, with UAVs being deployed from UGVs dynamically during the challenge. Since UAVs often rely on visual-inertial odometry (VIO) for self and relative estate estimation [144], relying on external LiDAR-based tracking can also extend the operability to low-visibility or other domains where VIO has inherent limitations [145; 146].

In this chapter, the focus is on two tracking scenarios: human tracking from a UAV equipped with a camera and UAV tracking by a ground robot equipped with novel solid-state LiDAR. It begins with an investigation into human detection and tracking using a UAV-mounted camera, primarily applied in search and rescue missions. The objective is to evaluate the detection performance at varying altitudes, specifically to enhance the efficiency of search and rescue operations. After demonstrating the UAV's tracking abilities, the discussion transitions to addressing relative spatial pose estimation for ground-aerial robot collaboration through novel adaptive tracking method with solid-state LiDAR.

6.1 Human Detection and Tracking with UAVs

Recent years have seen an increasingly wider adoption of unmanned aerial vehicles (UAVs) to support search and rescue (SAR) operations. Owing to their fast deployment, speed and aerial point of view, UAVs can aid quick response teams, but also in longer-term monitoring and surveillance [147]. Some of the main applications of UAVs in these scenarios are real-time mapping of the operational area or delivery of emergency supplies. In particular, UAVs can bring a significant increase of the response team's situational awareness and detect objects and people from the air, specially those in need of rescue [148]. An overview of recent research in this area is available in [149], where UAVs for SAR operations are characterized based on the operational environment, the type of robotic systems in use, and the onboard sensing capabilities of the UAVs.

We are interested in optimizing the support that UAVs can provide in maritime SAR operations (see Figure 35), but also for monitoring and surveillance in maritime environments, where they have already been widely utilized [150]. Maritime SAR operations might occur in both normal and harsh environments. For example, according to the Spanish national drowning report [151], in 2019 over 40% of drownings happened on a beach, around 60% of the incidents happened between 10:00 and 18:00, and in 20% of the cases lifeguards were present in the area. Therefore, there is still a need for better solutions for monitoring and supporting SAR operations in safeguarded beaches, lakes or rivers even with favorable weather conditions, which can then be extended towards rougher environments as the technology evolves. In this work, we study the detection of people in mostly still waters at different altitudes. In the future, we aim to utilize this information within an active vision algorithm that can dynamically adapt the flight plan of UAVs towards optimization of search speed and reliability.

In terms of UAV-based perception, deep learning (DL) methods have become the de-facto standards in object detection and image segmentation with great success across multiple domains [152; 153]. We utilize the YOLOv3 [152] architecture and characterize its performance for human detection on still water surfaces. Within the

Figure 35. Illustration of active-vision-based search in maritime environments with UAVs. A single UAV can first fly higher to cover larger areas and descend in the event of a positive detection to increase reliability. Search time can then be optimized by dynamically adjusting the altitude depending on the perception confidence.

machine perception field, active vision has been a topic of interest that has gained increasing research interest, owing to the multiple advances in DL and accessibility of UAV platforms for research. Active vision has been successfully applied for single and multi-agent tracking [154], but we have observed a gap in the literature in terms of active vision for search and area coverage. The most active research direction in active perception is currently reinforcement learning (RL) [155]. However, an RL approach can be challenging owing to the lack of realistic simulators to train models for sea SAR.

Deep learning for perception in maritime environments is limited by the lack of realistic training datasets openly available. Moreover, a key challenge for UAV-based person search and detection in these environments is the relatively small size of objects to be detected in comparatively large areas to be searched [156]. There is an evident trade-off between speed and area coverage, and reliability of both positive and negative detection. An additional challenge is that the view of people at sea from the air is only partial, as a significant portion of the body is immersed in the water. Water reflection and refraction effects might also distort the shape. In order to train YOLOv3 to adapt to this scenario, and owing to the lack of open data for detecting people in water, we collected over 450 high-resolution images to train, validate and test our model. The images have been taken at altitudes ranging from 20 m to 120 m.

This is, to the best of our knowledge, the first work to analyze the perception accuracy for UAVs with RGB cameras in maritime environments as a function of their altitude. The results can be generalized by accounting for the size in pixels of the persons to be detected assuming well-focused images. Moreover, the retrained YOLO model outperforms the state-of-the-art in object classification, as it has been trained to detect people even when only their head emerges above the water level. The retrained YOLO model can be applied for people swimming but also standing near the shore in a beach.

6.1.1 Human Detection for SAR Task

Multiple works have demonstrated the benefits of integrating UAVs to maritime SAR operations [157; 158]. Typical sensors onboard UAVs are RGB, RGB-D and thermal cameras, 3D LiDARs, and inertial/positional sensors for GNSS and altitude estimation [159; 160]. With these sensors, UAVs can aid in SAR operations by mapping the environment, locating victims and survivors, and recognising and classifying different objects [159]. From the perception point of view, DL methods have become the predominant solution for detecting humans or other objects [153; 161; 162].

Human detection is a sub-task of object detection that is of particular interest for SAR robotics [163]. Some of the most popular neural network architectures for object detection are R-CNN [164], Fast-RCNN [165], and YOLO [152]. In particular, YOLOv3 is the current state-of-the-art for real-time detection, able of fast inference and high accuracy [152]. In this work, we re-train the YOLOv3 network with a new dataset for detecting people in the water.

Active perception has been defined as:

> An agent is an active perceiver if it knows *why* it wishes to sense, and then chooses *what* to perceive, and determines *how*, *when*, and *where* to achieve that perception. [166]

In UAV-aided maritime SAR operations, algorithms for area coverage and human search incorporating active vision need to be aware that their main objective is to find humans (why), and need to be able to dynamically adjust their path planning and orientation to achieve higher-confidence results (what). This latter aspect can be achieved by, for instance, adjusting their height and camera pitch, or by moving around the person to get a better angle (how, where and when).

Active vision has been increasingly adopted in different object detection tasks. However, no previous research has, to the best of our knowledge, focused on active vision for detection of humans in SAR scenarios. We therefore list here some other relevant works in the area. Ammirato et al. presented a dataset for robotic vision tasks in indoor environments using RGBD cameras with the introduction of an active vision strategy using Deep RL to predict the next best move for object detec-

tion [167]. Juan et al. presented an autonomous Sequential Decision Process (SDP) for active perception of targets in uncertain and cluttered environments, with experiments conducted in a simulated SAR scenario [168]. Davide et al. applied active vision to a path planing algorithms that enabled quadrotor flight through narrow gaps in indoor complex environments [169]. Manuela et al. applied bio-inspired active vision for object avoidance with wheel robots in indoor environments [170]. In SAR operations, once a target has been identified, continuously updating the position of target is essential, so that path planning for the rescue teams can be adjusted. This can be achieved though active tracking [171].

In terms of detecting people in maritime environments, Eleftherios et al. presented a real-time human detection system using DL models that run on-board UAVs to detect open water swimmers [172]. The authors, however, do not study the accuracy of the perception for different altitudes or positions. In this work, we focus on analyzing human detection as a trade-off between larger area coverage (higher altitude) and higher amount of detail in the images (lower altitudes).

In general, we see a clear trend towards a more widespread utilization of UAVs in SAR missions and DL models for perception (either onboard or offloading computation). We have found, however, no previous works exploring the correlation between the altitude at which UAVs fly and the detection accuracy in maritime SAR scenarios.

6.1.2 Deep Learning-Based Huamn Detection

This section describes the data and details of the training process for the perception algorithm. We also outline the metrics that are analyzed in our experiments.

Table 12. mAP-scores for different IoU-thresholds.

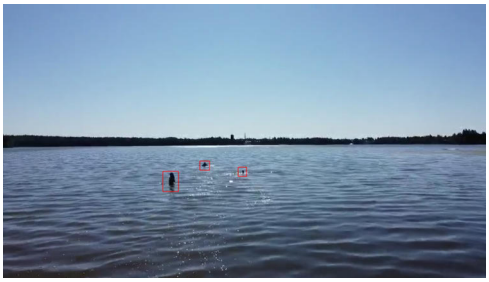
	IoU-threshold									
	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
TS model	0.698	0.698	0.697	0.695	0.693	0.688	0.670	0.638	0.542	0.000
YOLOv3	0.054	0.053	0.053	0.052	0.051	0.051	0.051	0.050	0.044	0.000

6.1.3 Data Acquisition

Owing to the lack of labeled open data showing people in water, and in particular data labeled with the flight altitude, we have collected data from people swimming and standing in a lake. The dataset contains 458 labeled photos that are taken by the camera mounted on the UAV. The camera has a fixed focal length of 24 mm (35 mm format equivalent) with a field of view of 83° and an aperture f/2.8. The images have a resolution of 9 MP (4000 by 2250 pixels), and were recorded near the beach area of Littoistenjärvi Lake (60.4582 N, 22.3766 E), shown in Figure 36 (a), Turku, Finland.

(a) Beach view

(b) Top view



(c) Low altitude

(d) Back light

Figure 36. (a) Example images of terrain at Littoinen Lake, Finland, (b) The top view of swimmer, (c) The far view of swimmers, (d) The close view of swimmer

Each photo captures one or more people that are either swimming or standing in the lake at different heights and angles. Some examples are shown in Figure 36 (b), (c) and (d). However, the majority of pictures were taken with a gimbal pitch of -90° (top-view images). The dataset contains 2D bounding boxes for two classes: *persons* and *other objects*, the latter one being used for animals in the water and other floating objects. In addition to the bounding boxes, each image contains information about the GPS position, relative altitude to the take-off point (just above the water level), and pitch angle of the camera gimbal (from horizontal images with 0° pitch to top-view images with -90° pitch). The relative altitude ranges from 0 m to 143 m. While the dataset has been acquired with good weather conditions and mostly still waters, variable light conditions are also introduced. This results in different colors for both water and people, as can be seen in Figure 36 (b) and (d). Some of the swimmers use swimming caps of different colors and wear different types of swimming suits.

6.1.4 Modal Training and Test Setup

Training and testing were done with the YOLOv3 real-time object detection model [152]. The YOLOv3 model pre-trained with ImageNet [173] was trained again with our dataset using transfer learning. Training is done in a way where all but the last three

layers are frozen for the first 50 epochs and then unfrozen and trained further for another 50 epochs with batch size of 32 and learning rate of 0.001.

Each image contains between 1 and 50 object instances. The objects are divided into two classes: *'person'*, containing 2454 instances, and *'something else'*, containing 238 instances, mostly birds but also some other objects floating in the water. All the images were labeled manually, using bounding boxes with the Labelbox annotation tool [174]. Training and testing were done using 4-fold cross-validation, randomly splitting the images using a 75/25 train/test split. We refer to the re-trained model as the task-specific model hereinafter.

6.1.5 Evaluation Metrics

Object detection performance was evaluated using PASCAL VOC challenge metrics [175] provided by [176]. We calculated average perception (*AP*) for both classes separately and mean average perception (*mAP*) over both classes using different intersection over union (*IoU*) thresholds. The comparison in performance was done between the pre-trained YOLOv3 model and the task-specific model with our data using transfer learning. Furthermore, since our objective is to analyze the correlation between the performance of the human detection and the altitude, we also analyze how the detection confidence and the ratio of false positives and false negatives changes as a function of the altitude.

6.1.6 Experimental Results

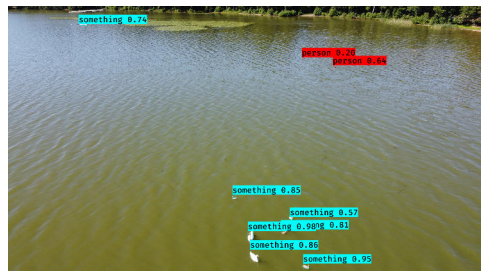
In this section, we assess the performance of the trained model as a classifier using the mean average precision for different IoU thresholds, but also its usability for active-vision-based control where the input to the algorithm is the confidence of the model on each of its detections.

Some representative example detections made by the task-specific model are illustrated in Figure 37. In Figure 37a, we observe how the network is able to pinpoint the location of people in the image, but the bounding box appears around the turbulent water rather than around the person itself. However, not all objects or turbulent areas are detected as people, as other objects are also properly identified (Figure 37b. In Figure 37b, we also observe that people can be located far away when the gimbal pitch is closer to 0°. Finally, we see that even at high altitudes, the confidence remains high and people are detected also when immersed (Figure 37c).

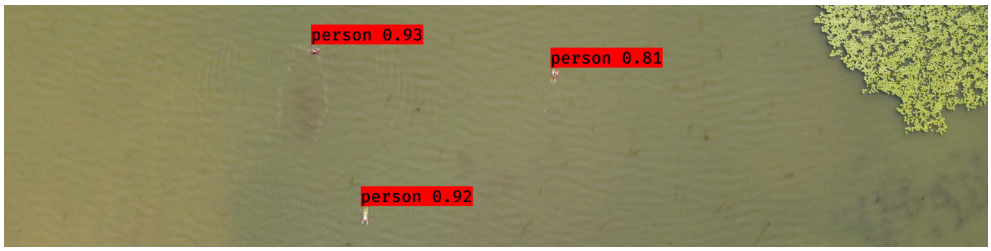
The performance of the task-specific model compared to the pre-trained YOLOv3 network is shown in Table 12, where we see that the task-specific model is clearly superior. In terms of the precision $\times$ recall curves, those corresponding to classes *'person'* and *'something else'* are provided in Figure 38a.

Next, we analyze performance at different altitudes. The significance of the al-

(a) Detection of one person (high confidence), and turbulent water next to another (lower confidence). Altitude: 37 m. Pitch: -80°.



(b) Detection of other objects but missing two persons in the distance. Altitude: 12 m. Pitch: -25°.



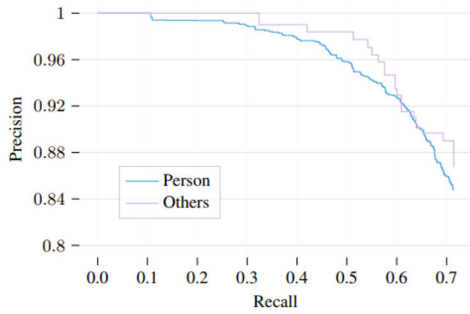
(c) Successful detection of three people at high altitude, one of them fully immersed in the water (only a portion of the original image is shown). Altitude: 86 m. Gimbal pitch: -90°.

Figure 37. Samples of detections made using the task-specific model.

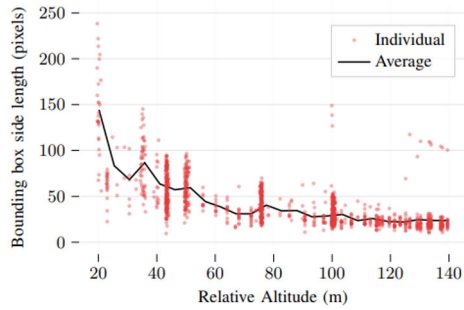
titude is, however, relative to the resolution of the camera and its ability to produce clear images. The camera pitch is also important as illustrated. In order to provide results that are more generalizable, Figure 38b shows the size in pixels of the ground truth bounding boxes.

Figure 39a(a) shows all the person detections plotted in terms of their confidence against the altitude, using $IoU = 0.1$ to consider true positives. We have set the IoU to 0.1 because we are only interested in pointing to the approximate location of persons but not their exact size and place. For altitudes under 60 m, over 98.8% of the detections with a confidence above 0.5 are correct. A clear threshold appears at an altitude of 90 m. Above 90 m, 83.3% of the detections are correct.

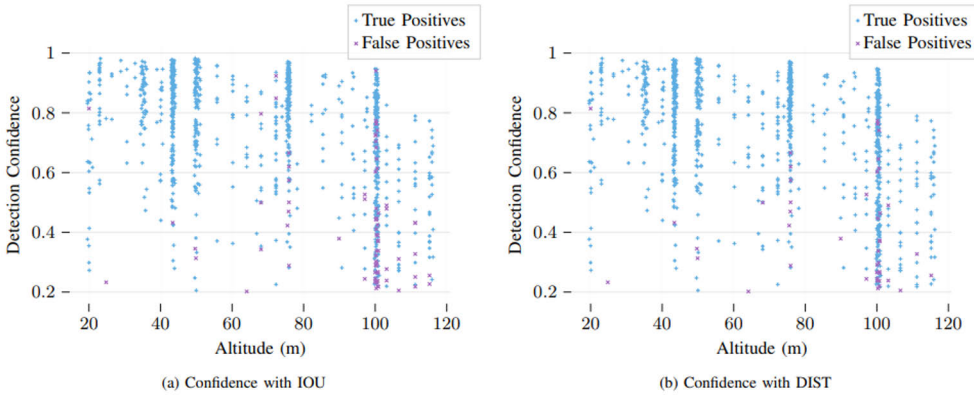
In some of the test images, we have noticed that the model detects turbulence in the water created by people as persons, and not the full bodies of the people themselves. Because we are not interested in analyzing how capable the task-specific model is of generating accurate bounding boxes, but instead on pointing to the approximate location of people at sea, we might also want to consider as correct detection boxes that are just adjacent to actual people. In Figure 39a(b), we have plotted the confidence as a function of the altitude, but now using a distance in pixels of less than 100 between the ground truth and the predicted box (DIST) to assume that a detection is correct. We now see that all except one of the positive detections with



(a) Precision x Recall curve for class 'person' and 'something else' using IoU-threshold 0.5 with the task-specific model.



(b) Side length of the ground truth bouding boxes, in pixels, based on the altitude.

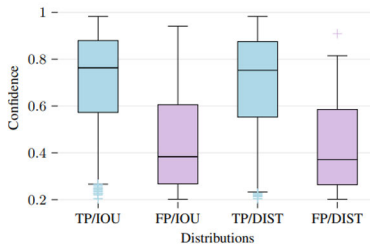


(a) Confidence with IOU (a) and Confidence with DIST (b).

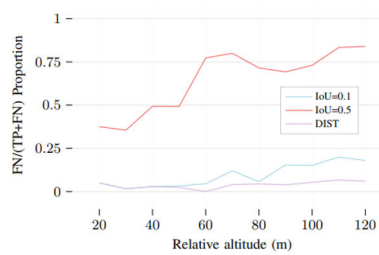
Figure 39. Confidence of individual detections as a function of the relative UAV altitude. We observe a clear difference between high-confidence and true positives under the threshold of 100 m, with lower confidence and higher rate of false positives above it.

a confidence of over 70% are correct for an altitude up to 100 m. For a confidence above 45%, all but one detections are correct up to an altitude of 70 m. The distributions of the true positives and false positives for each of the two metrics (IoU, DIST) are shown in Figure 40a. There is a clear threshold just under a confidence of 0.6, with almost 75% of true positive having a confidence over 0.6, and almost 75% of false positives having a lower confidence.

In order to evaluate this model within its context for SAR missions, we also need to take into account that false positives do not necessarily have a significant impact on the search performance, but false negatives do, as they mean that the UAV misses a person. We have therefore plotted in Figure 40b the proportion of false negatives over true positives. If we use the pixel distance to consider a detection as correct,



(a) Distributions for the confidence of true positive (TP) and false positive (FP) detections (DIST and $IoU = 0.1$).



(b) Proportion of false negatives (FN) over true positives (TP) and FN. This gives an idea of the probability of missing a person.

then the proportion remains under 10% for all altitudes. With $IoU = 0.5$, however, over 50% of the people in the water are undetected. However, we do not consider this an effective way of evaluating a detection in this scenario.

6.1.7 Summary and Conclusion

With UAVs increasingly penetrating multiple civil domains and, among them, search and rescue operations, more complex control mechanisms are required for more autonomous UAVs. To that end, active perception is one of the most promising research directions. In UAV search, active vision can be exploited to optimize the flight plan based on the confidence of the DL vision algorithms. We have presented preliminary work that studies the confidence of a re-trained YOLOv3 model for detecting people in the water for altitudes ranging from 20 m to 120 m. With a custom dataset, we have seen a major performance increase with respect to the pre-trained YOLOv3 network. Our results show a clear correlation between the altitude and the confidence of the detections and between the confidence and the correctness of the detections. When considering as true positives detections near actual people (e.g., over water turbulence created by people), we have seen that the proportion of false negatives remains low even for high altitudes, and the proportion of false positives over true positives drops significantly for all predictions with a confidence over 60%. Finally, we have observed a clear altitude threshold at around 100 m after which confidence and accuracy drop.

6.2 Micro UAV Detection and Tracking with Solid-state LiDAR

Micro-aerial vehicles (MAVs) have seen an increasing adoption across a variety of application domains in recent years [177]. Multiple works have been devoted to the navigation of MAVs in GNSS-denied environments [178], and state estimation in both single [179] and multi-MAV systems [180]. In this chapter, we are particularly

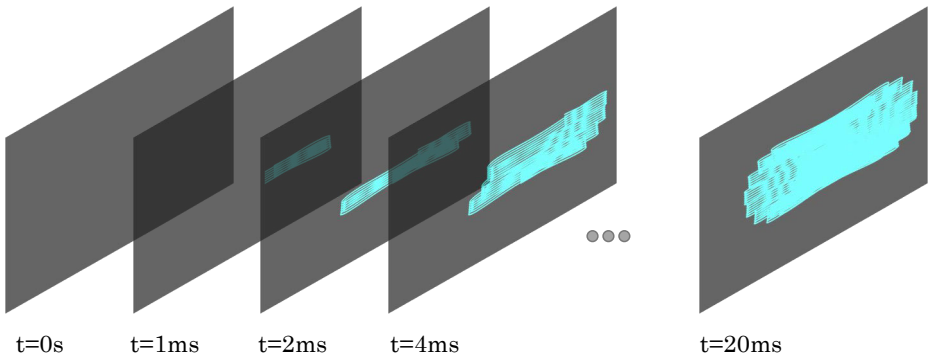


Figure 41. Illustration of the field of view (FoV) coverage with different point cloud integration times in a non-repetitive LiDAR scanning device.

interested in tracking and state estimation from an external system, for those applications where MAVs are deployed together with or from unmanned ground vehicles (UGVs) [181; 182].

Tracking and detecting UAVs has been a topic of interest for researchers in recent years. First, owing to the increasing need of identifying and detecting foreign objects or drones in areas with controlled airspace such as airports [183; 184]. Second, to optimize the utilization of UAVs as flexible mobile sensing platforms. This chapter focuses on the latter use. Compared to the existing literature, which relies mainly on vision-based techniques [185], we provide a LiDAR-based solution that can be utilized more independently of the environmental conditions. Until recently, most 3D LiDARs provided relatively sparse point clouds in terms of object recognition [186], with limited vertical resolution in inexpensive devices. However, solid-state LiDARs have recently emerged as state-of-the-art in terms of long-range scanners featuring high-density point clouds [16]. The main caveat is the limited field of view (FoV) in most of these devices [15], but solutions include utilizing multiple LiDARs or correspondingly adjusting the position and orientation of the robot base where the LiDAR is installed. In this section, we also put our focus on single and known micro UAV detection and present generic methods that can be extended to multi-MAV tracking.

Here, WE are particularly interested in the problem of tracking a MAV that is deployed from a ground robot. We assume thus that the initial position of the MAV after take-off is known. We also assume that its shape and size are known a priori. We develop methods targeting solid-state LiDARs owing to the higher density of the resulting point cloud even with more limited FoV. Moreover, in these LiDARs, the concept of a frame or scan frequency changes considerably. Similarly, as in rotating 3D LiDARs, a frame in solid-state LiDARs can be naturally related to a single revolution. With non-repetitive scan patterns, LiDARs can output point clouds

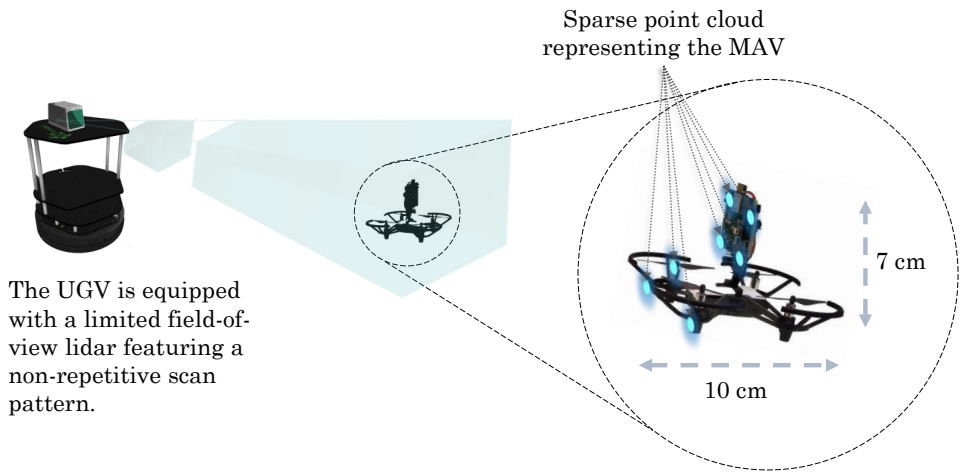


Figure 42. Illustration of a ground robot tracking a micro-aerial vehicle (MAV) using a limited-FoV solid-state LiDAR.

at adjustable frequencies with varying FoV coverage, as illustrated in Figure 41. This opens the door to new LiDAR perception methods that exploit the possibilities of adaptively adjust the frame integration time to better sense the objects. To the best of our knowledge, this approach has not been previously studied. We apply the proposed adaptive LiDAR scan integration methods within the problem of a UGV tracking a MAV for external state estimation, as conceptualized in Figure 42. While our focus is on MAVs, the proposed methods can also be easily adapted to detect foreign objects or intruder MAVs more accurately. We first put our focus on single and known MAV detection, but present generic methods that can be extended to multi-MAV tracking as long as FoV limitations are accounted for.

6.2.1 Problem Definition

We consider the problem of tracking a MAV from a ground robot. The objective is to improve the collaboration between the robots and the ability of the MAV to navigate in complex environments aided by the UGV. The rest of this work delves into the definition, design, and implementation of methods for tracking a single MAV. Nonetheless, these can be extended to multiple MAVs. The main limitation when tracking multiple units is the FoV of the LiDAR sensors onboard the ground vehicle, and therefore assumptions have to be made to the spatial distribution of the MAVs (always within the FoV of the ground robot). Alternatively, more LiDAR scanners can be installed to increase the FoV.

The majority of 3D laser scanners available to date are multi-channel, rotating LiDARs. While devices with 64 or 128 vertical channels can provide high angular

resolution in both horizontal and vertical dimensions, these high-end devices are not the most common. Moreover, the scanning pattern is in general repetitive, which has benefited from a geometric perspective in terms of data processing but does not enable a higher FoV coverage with longer exposure if the position of the sensor is fixed. New solid-state LiDARs featuring non-repetitive scan patterns, albeit having more limited FoV, can provide more dense point clouds and often feature longer detection ranges. In particular, we are interested in the possibilities of dynamically adjusting the FoV coverage and density in the point cloud to be processed for detection and tracking. Among the benefits of these new LiDARs and the possibilities of adaptive scanning rates is also higher resilience against one of the challenges in LiDAR-based perception: motion-induced distortion [187]. In general, the literature targeting tracking of MAVs using LiDAR scanners is scarce, and existing methods in point cloud object detection and tracking considering mainly static frames. We aim to define more optimal settings for generating point clouds based on the state (speed and distance to the sensor) of the MAV being tracked.

6.2.2 Tracking System Overview

We propose three simultaneous tracking modalities with three processes analyzing point cloud frames resulting in integration times ranging several orders of magnitude. A general view of the multi-modal tracking processes is shown in Figure 43. In more detail, the three modalities are described below:

1. Adaptive high-frequency tracking. In this first process, sparse point clouds are integrated at frequencies up to 100 Hz. The MAV is only trackable through a reduced number of points, but we are able to estimate its position and speed with high accuracy. In this process, the MAV is not necessarily recognizable in all processed frames.

2. Adaptive medium-frequency tracking. The second process operates at frequencies within the range of typical LiDAR scanners (i.e., 5 to 20 Hz). The frequency within that same range is dynamically adjusted to optimize the density of the point cloud. At these frequencies, the extracted point cloud representing the MAV is distorted by motion, and thus the localization and speed estimation accuracy is lower. However, this process enables more robust and persistent tracking as the MAV can be recognized in most if not all frames.

3. Low-frequency trajectory and object validation. The third and last process that runs in parallel to the previous two performs long-term tracking and validates the reconstructed trajectory of the MAV based on predefined dimensional constraints. An illustration of such trajectory reconstruction is shown in Figure 44

Let $\mathcal{P}_k(I_r^k)$ be the point cloud generated by the LiDAR with an integration time I_r^k. We also denote by $\mathbf{s}_{MAV}^k$=$\{\mathbf{p}_{MAV}^k$,$\dot{\mathbf{p}}_{MAV}^k\}$ the position and speed of the MAV. We use discrete steps represented by k owing to the discrete nature of the set of consecutive point clouds. The output of the main tracking algorithm is to extract from $\mathcal{P}_k(I_r^k)$ the set of points representing the MAV, which we denote by $\mathcal{P}_{MAV}^k$, and to adjust the integration time for the next point cloud, I_{HF}^k, I_{MF}^k.

Algorithm 3: MAV tracking with adaptive scan integration

Input:

High- and medium-freq int. rates:	$\{I_{HF}^{k-1}, I_{MF}^{k-1}\}$
3D lidar point clouds:	$\{\mathcal{P}_k(I_{HF}^{k-1}), \mathcal{P}_k(I_{MF}^{k-1})\}$
Last known MAV state:	$(\mathbf{p}_{MAV}^{k-1}, \dot{\mathbf{p}}_{MAV}^{k-1})$

Output:

MAV state:	$\{\mathbf{p}_{MAV}^k, \dot{\mathbf{p}}_{MAV}^k\}$
UGV control:	$\dot{\mathbf{q}}_{UGV}^k$
Int. rates:	$\{I_{HF}^k, I_{MF}^k\}$

Function $object\_extraction\left(\mathcal{P},\ I,\ \mathbf{p}_{MAV}^{k-1},\ \dot{\mathbf{p}}_{MAV}^{k-1}\right)$**:**

Ground removal:	$\mathcal{P}^{'} \leftarrow \mathcal{P}$;	
Generate KD Tree:	$kdtree \leftarrow \mathcal{P}^{'}$;	
MAV pos estimation:	$\hat{\mathbf{p}}_{MAV}^k \leftarrow \mathbf{p}_{MAV}^{k-1} + \frac{\dot{\mathbf{p}}_{MAV}^{k-1}}{I}$;	**return** $\mathbf{p}_{MAV}^k$;
MAV points:	$\mathcal{P}_{MAV}^k = KNN(kdtree,\ \hat{\mathbf{p}}_{MAV}^k)$;	
MAV state estimation:	$\mathbf{p}_{MAV}^k = \frac{1}{\lvert\mathcal{P}_{MAV}^k\rvert} \sum_{p \in \mathcal{P}_{MAV}^k} p$;	

`// Coarse but persistent tracking`

while *new* $\mathcal{P}_k(I_{MF}^k)$ **do**

  $\mathbf{p}_{MAV}^{k'} = object\_extraction(\mathcal{P}_k(I_{MF}^k), I_{MF}^k, \mathbf{p}_{MAV}^{k-1}, \dot{\mathbf{p}}_{MAV}^{k-1})$;

`// Fine-grained estimation`

while *new* $\mathcal{P}_k(I_{HF}^k)$ **do**

  $\mathbf{p}_{MAV}^{k''} = object\_extraction(\mathcal{P}_k(I_{HF}^k), I_{HF}^k, \mathbf{p}_{MAV}^{k-1}, \dot{\mathbf{p}}_{MAV}^{k-1})$;

  $\mathbf{p}_{MAV}^k,\ \dot{\mathbf{p}}_{MAV}^k \leftarrow estimate\left(\mathbf{p}_{MAV}^{k'},\ \mathbf{p}_{MAV}^{k''}\right)$;

$\{I_{HF}^k, I_{MF}^k\} \leftarrow adjust\_integration\_freqs\left(\mathbf{p}_{MAV}^k, \dot{\mathbf{p}}_{MAV}^k\right)$;

$\dot{\mathbf{q}}_{UGV}^k \leftarrow keep\_within\_FoV\left(\mathbf{p}_{MAV}^k, \dot{\mathbf{p}}_{MAV}^k\right)$;

6.2.3 Adaptive Scan Integration

Since we assume that the state of the MAV $(\mathbf{p}_{MAV}^{k-1}, \dot{\mathbf{p}}_{MAV}^{k-1})$ is initially known, the point cloud processing proceeds as follows. First, we perform ground removal based on the last-known altitude of the MAV. We then proceed with finding the nearest neighbor points to a predicted MAV position. This step is repeated for both the high and medium frequency scans, the former one providing a more accurate position

estimation while the latter is more persistent in time. Finally, these two estimations are combined, and the results are utilized to adjust the integration rates based on the point cloud density expected for the given distance and speed. This process is outlined in Algorithm 3.

Algorithm 4: Trajectory validation

Input:

Low-freq int. rate: I_{LF}^{k-1}

3D lidar point cloud: $\mathcal{P}_k\left(I_{LF}^{k-1}\right)$

MAV state history: $(\mathbf{p}_{MAV},\, \dot{\mathbf{p}}_{MAV})$

Output: Trajectory validation (*bool*)

while *new* $\mathcal{P}_k\left(I_{LF}^{k-1}\right)$ **do**

```
// Generate cubic splines
// with position and speed constraints
```
$\{B_i\} \leftarrow \{\mathbf{p}_{MAV},\, \dot{\mathbf{p}}_{MAV}\}$;
```
// Estimate expected point cloud from
// known density at given distance and speed
```
$\hat{\mathcal{P}}_k \leftarrow \{\{B_i\},\, \mathbf{p}_{MAV},\, \dot{\mathbf{p}}_{MAV}\}$;
```
// Calculate IoU
```
$IoU = calc\_IoU\left(\mathcal{P}_k\left(I_{LF}^{k-1}\right), \hat{\mathcal{P}}_k\right)$;

if $IoU > th$ **then**
    return True
else
    return False

The main purpose of the low-frequency scan stream is to validate the extracted MAV's trajectory. While the tracking with adaptive scan integration only takes into account the MAV size roughly in terms of distance within which nearest neighbors are looked for, the extracted point cloud is not validated against its known dimensions. This is done when enough points are accumulated into a reconstructed trajectory. As exposed in Algorithm 4, we first perform a cubic spline interpolation based on the history of estimated positions and speeds. To calculate the parameters of the cubic spline, we utilize constraints on the first derivative based on the speed, rather than forcing the first and second derivative to be continuous. Indeed, the acceleration of the MAV can suddenly change. Based on predetermined values of point cloud density as a function of the MAV's distance to the LiDAR and its speed, we then produce an expected point cloud. We validate the original point cloud given a threshold for the IoU measure with the generated estimate.

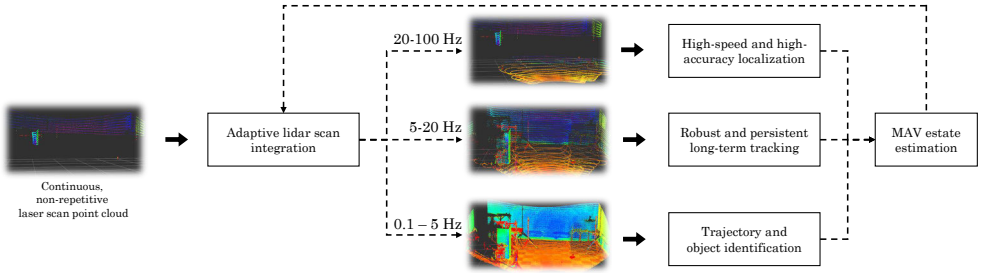


Figure 43. Overview of the proposed methods, where tracking is simultaneously performed at three different scan frequencies. Within each of these three threads, the scan frame integration is adjusted based on the distance to the target MAV and its speed.

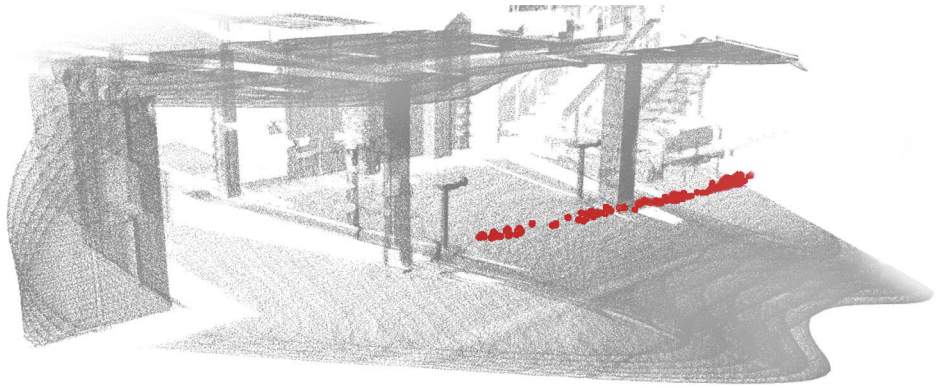


Figure 44. Integration trajectory recovery example

6.2.4 Experimental Evaluation

Experimental platforms

The experimental platforms consist on a single ground robot and a commercially available Ryze Tello MAV. The ground robot is an EAI Dashgo platform equipped with a Livox Horizon LiDAR ($81.7° \times 25.1°$ FoV). The LiDAR is able to output scanned pointcloud up to 100 Hz, featuring a non-repetitive pattern. A pair of ultra-wideband (UWB) transceivers is used to obtain a single range between the robot and the MAV at frequencies ranging from 10 Hz to 100 Hz. The UWB ranging is used in aiding the manual validation of the extracted trajectory in places where there was no external positioning system. In the future, it could be incorporated as part of the tracking algorithm as well, as is becoming increasing adopted in multi-robot systems [188; 189].

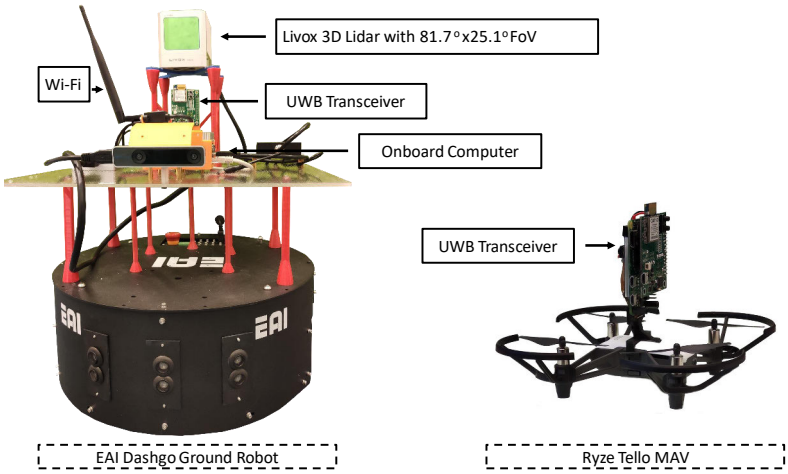


Figure 45. Ground robot and MAV utilized in the experiments.

Software

The system has been implemented using ROS Melodic under Ubuntu 18.04. The algorithms are running in the main computer onboard the ground robot. The computer runs the Tello MAV driver[1], the Livox LiDAR driver[2], and our open-source MAV tracking package[3]. The latter is a multi-threaded node able to process the different point clouds in real time. The point cloud library (PCL) [190] is utilized to extract the position of the MAV from the LiDAR's point cloud.

[1] `https://github.com/TIERS/tello-driver-ros`

[2] `https://github.com/Livox-SDK/livox_ros_driver`

[3] `https://github.com/TIERS/adaptive-LiDAR-tracking`

Metrics

Due to the lack of an accurate external positioning system, such as a motion capture system for large environments, our focus is instead on measuring the performance of the tracking at different scan integration rates and manually validating the overall trajectory. The experimental flights are carried out in large indoor halls with multiple columns and objects, as shown in Figure 44. Another set of experiments is carried out in a small flying area where an external UWB positioning system was available and used to fly the MAV over a predefined trajectory. A characterization of the accuracy of such a system can be found in [191].

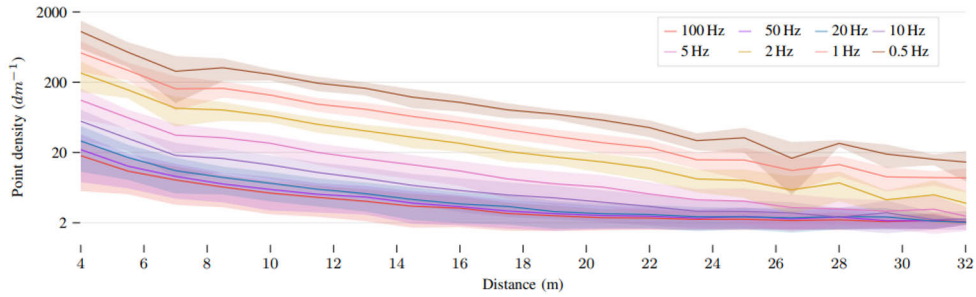


Figure 46. Density of the point cloud representing the MAV based on the distance to the LiDAR scanner and the scanning frequency.

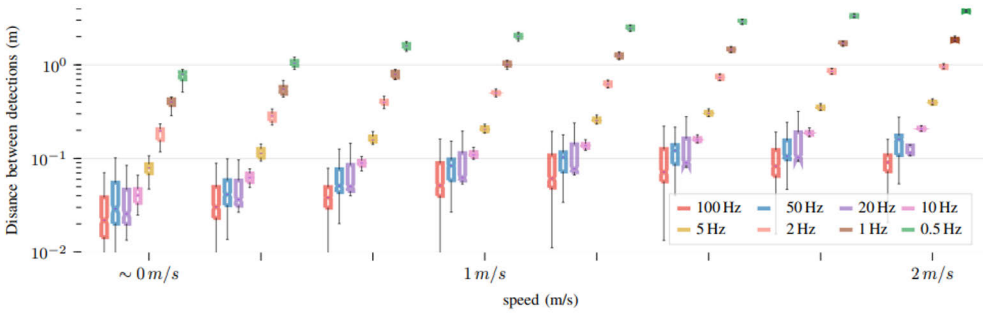


Figure 47. Distance between consecutive MAV detections based on its speed and the LiDAR's scanning frequency.

Adaptive Scan Integration Results

The first objective of our experiments was to assess the tracking performance at different scan frequencies in order to better model the adaptiveness of our algorithm. In order to adapt the scanning frequency to optimize the tracking performance, key

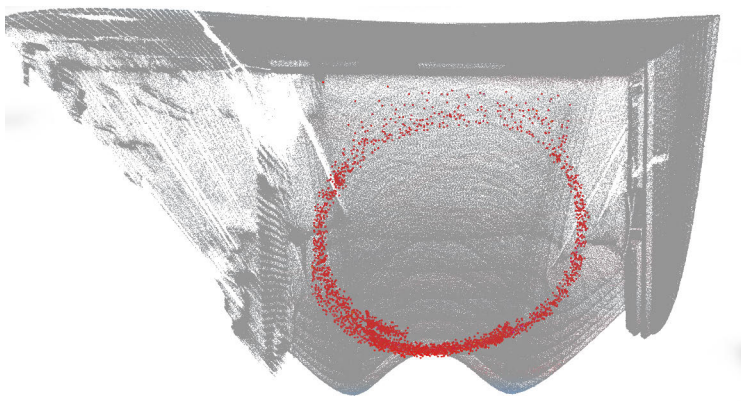


Figure 48. Accumulated point cloud for the circular trajectory.

parameters are the point cloud density at different distances and the reliability of the detections at different speeds.

The point cloud density for different scanning frequencies as a function of the distance between the LiDAR and the MAV is shown in Figure 46. This measure refers only to the density of the points representing the MAV and not the overall density including the rest of the scene. The darker lines represent the average point cloud density, while the band with higher transparency represents the values within the standard deviation. The size of the Tello MAV is about 500 cubic centimeters. Based on our experiments, reliable tracking at high speeds can be achieved with at least 4 points, while we require at least 20 points at medium scanning frequency. This, however, only applies in free space. As can be seen in Figure 48, significant noise appears in the point cloud between the MAV and walls in the environment when flying nearby. We discuss further this issue at the end of this section.

In terms of the tracking performance based on the speed, we plot in Figure 47 the distance between consecutive detections at different scanning frequencies. The results in this particular figure cannot be directly utilized to model the adaptive nature of our tracking algorithm. Nonetheless, they can be leveraged to better understand what are the speed limits under which given scanning frequencies do not provide the expected distance between detection that can be inferred from the MAV speed and the scan frequency.

The results included in Figure 46 and Figure 47 have been obtained flying the MAV in a long, straight corridor with a length of about 35 m. The MAV was flying mostly in straight lines and the speed was estimated using both visual odometry and the position history extracted from the LiDAR data in a partially manual manner.

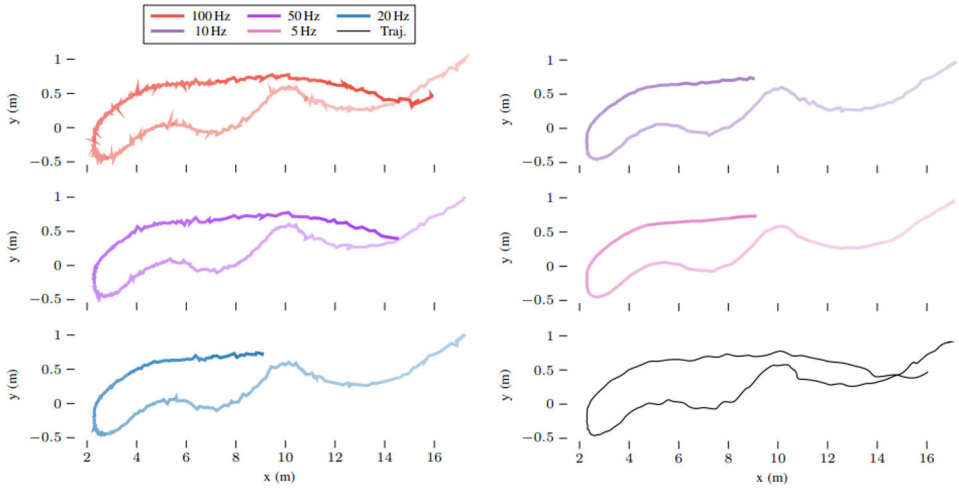


Figure 49. Estimated trajectories at different frequencies and with adaptive approach (top five plots), and trajectory estimated from our algorithm (bottom plot).

Qualitative Trajectory Validation

In order to validate the performance of the tracking algorithm and better understand the limitations of our tracking approach at different scanning frequencies, we compare two different types of trajectories. Owing to the lack of a system to obtain ground truth (e.g., a motion capture system), we provide qualitative analysis for one of the trajectories and compare it with a UWB positioning system in the other one.

First, we test the tracking algorithm through a trajectory where the MAV flies in a large open area at distances from 2 m to over 17 m far from the LiDAR scanner and variable speeds. In this scenario, the analysis is mostly qualitative, with the trajectories shown in Figure 49. However, the UWB ranging data and the LiDAR data has been both manually confirmed, so the maximum positioning error along the track is at worst around 20 cm. Qualitatively, the main results from this experiment are the ability of the tracking algorithm to keep track of the MAV over changes in speed, direction, and at longer distances. The figure only shows frequencies equal to or above 5 Hz because at lower scanning frequencies the speed estimation was highly inaccurate during the early stages of the flight. We can see that only at the highest frequency we are able to track the MAV along the completed trajectory, while the trajectory itself is noisier. The higher level of error when estimating the MAV position is due to a lower number of points being detected, which can correspond to different parts of the MAV in consecutive scans. The last subplot shows the overall estimated trajectory where our algorithm has combined the different scanning frequencies to obtain the smoothness of the medium frequencies and the performance of the higher frequencies. The trajectory also employs the cubic spline interpolation

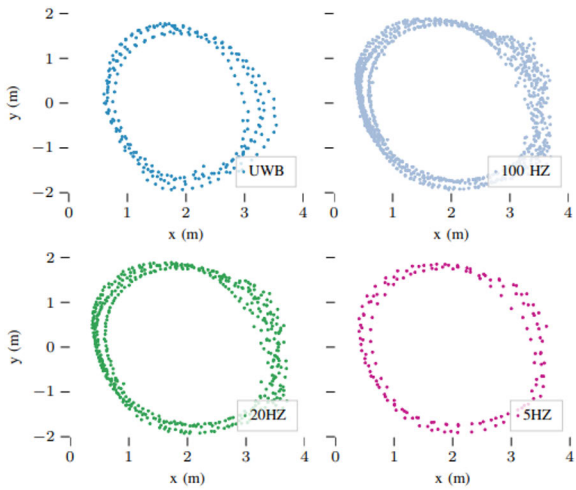


Figure 50. Reference trajectory (UWB) and estimated positions at different fixed frequencies.

from the validation algorithm.

Second, we perform a continuous flight with a predefined circular trajectory in a small flying arena where the UWB positioning system is available. The results for this flight are shown in Figure 50. The leftmost plot shows the reference position. However, it is worth noticing that the accuracy of the LiDAR, of around 2 cm for distances shorter than 20 m, is higher than the average accuracy of 10 to 15 cm in the UWB positioning system. Therefore, the trajectory is mere as a reference and only a qualitative discussion is possible with these results. In any case, owing to the continuous change in the speed of the MAV, which is a prior unknown to the tracking algorithm, again only at frequencies equal or over 5 Hz are we able to track the MAV. Nonetheless, at 5 Hz the tracking stops before the fourth revolution is completed, and persistent tracking is only possible when higher frequencies are taken into account.

6.2.5 Summary and Conclusion

We have shown in this section qualitative results that show the performance of the adaptive tracking algorithm and the same approach applied only to specific scanning frequencies. From both sets of experiments, the main conclusion is that the adaptive approach is able to accommodate a wider variety of scenarios. We have been able to put together the flexibility of high-speed tracking with the robustness of medium frequencies, avoiding the frequent errors of the former, and the lower tracking capacity of the latter is more challenging conditions.

One key limitation when tracking MAVs, as visualized in the circular trajectory

experiments, is the low density of the point cloud and the inability to tell the difference between the MAV's points and LiDAR noise. This is also due to the low reflectively of the MAV, and there is thus the potential for mitigation with more reflective surfaces that could aid in separating the sparse MAV point cloud from the LiDAR noise originated due to near objects. As we can see in Figure 48, the point cloud density near the rear wall is very sparse in some areas, therefore being unable to reconstruct a robust trajectory as there are multiple options available that would meet the dynamics and dimensional constraints of the MAV.

7 Conclusions

This thesis aimed to investigate robust localization, mapping, and tracking algorithms through the exploration of novel sensors and fusion methods. This encompassed creating a multi-LiDAR dataset to advance SLAM research, addressing mapping complexities in environments like large-scale forests and GNSS-denied urban areas. The work also involved devising methods for dynamic object detection and tracking and enhancing dense mapping capabilities using spinning LiDAR and solid-state LiDAR technologies.

7.1 Summary and Contributions

The thesis start by studying different modal LiDAR sensors and investigate the state-of-the-art SLAM algorithms with presented dataset in Chapter 2. The comprehensive LiDAR dataset that includes data from five LiDAR sensors, a LiDAR camera, and stereo fish-eye cameras, spanning various environments. The evaluation of nine sequences on two computing platforms focuses on LiDAR Odometry and power consumption. The dataset serves as a benchmark for different localization and mapping algorithms, facilitating the development of generalized methods for diverse LiDAR sensors and environments. From the experimental results, it was observed that the solid-state LiDAR-based SLAM algorithms present the most detailed and clear map. However, the spinning LiDAR-based method exhibits less odometry drift across different environments. This indicates that each LiDAR has its own advantages, and fusing these sensors into one LiDAR system can contribute to building a more robust mapping system

Chapter 3 addressed LiDAR-based global localization challenges in GNSS-denied unstructured forest environments. An effective segmentation-based approach was proposed for accurate trunk position detection and Delaunay triangulation-based localization. This method enables real-time estimation of precise 3D position and rotation in forest environments, aiding autonomous vehicle localization and trunk recognition. From the real experiments with data from spinning LiDAR attached to a working harvester, our semantic graph search approach demonstrated promising performance in accurately localizing vehicles in challenging forest environments without GNSS data. This can greatly benefit future fully autonomous harvesters operating in forest environments.

Chapter 4 delves into another global localization problem, focusing on urban environments for delivery robots, where pre-built maps are altered, and accurate GNSS data is unavailable. A sensor fusion method using 3D LiDAR data complemented with GNSS and inertial data was proposed to rectify corrupted maps. Algorithms and analysis were tested in the JD Digital Globalization Challenge, where our team excelled. From the results, LiDAR scan matching against a 3D map provides the highest accuracy for localization, both in terms of position and orientation. Therefore, it is essential to take into account different modality sensor data to implement a more robust approach that is less prone to instabilities and depends less on the operational environment. While GNSS and inertial data are essential for increasing localization accuracy, their performance can be affected by the environment. Hence, it is crucial to detect and minimize the possibilities of unexpected behavior of sensors for a robust localization algorithm.

Based on the findings from studying different modal LiDARs and IMU sensors in Chapter 2, and robust localization solutions in Chapters 3 and 4, Chapter 5 introduces a multi-modal multi-LiDAR sensor fusion framework utilizing novel LiDAR sensors for high-performance localization and mapping in challenging indoor environments. The proposed system effectively integrates the advantages of dense point clouds generated by solid-state LiDAR for enhanced situational awareness and the larger Field of View (FoV) offered by spinning LiDARs, achieving high-robustness odometry and producing high-quality maps by utilizing an optimal combination of cost-effective sensors. Additionally, the proposed framework adopts novel time synchronization, IMU pre-integration, as well as sliding window optimization methods for state-of-the-art performance. The methods are modularly designed and can be adapted to other LiDAR combinations systems.

Chapter 6 explores dynamic object detection and tracking techniques for accurate mapping and multi-robot cooperation, focusing on deep learning-based object detection and adaptive tracking algorithms in various scenarios, notably human detection during search and rescue operations and non-repetitive scanning patterns with solid-state LiDAR. Our experiments with camera-based human detection reveal correlations between altitude and detection confidence, as well as between confidence and detection correctness, particularly for detections near actual people and over water turbulence. Solid-state LiDAR with non-repetitive scanning patterns enables tracking of small objects, despite challenges arising from the continuous change in target speed. Integration at high frequencies results in higher error levels in MAV position estimation due to fewer detected points, while lower scanning frequencies lead to highly inaccurate speed estimation. In conclusion, persistent tracking is achievable only with consideration of higher frequencies.

In summary, this research addressed key challenges in localization, mapping, and tracking across different sensor modalities, providing solutions for challenging environments. Additionally, a large-scale multi-modal multi-sensor dataset was curated

to aid in the development of generalized localization and mapping algorithms.

7.2 Future Research

This thesis extensively investigates innovative methodologies employing advanced sensor fusion algorithms to significantly enhance various LiDAR perception tasks, including global localization, multi-sensor fusion, and object tracking. Despite addressing some key research problems, further efforts are required to develop a robust, accurate, and reliable localization and mapping system capable of operating effectively in diverse challenging environments. In the subsequent section, we primarily explore potential future research directions and address open questions based on our findings in these areas.

7.2.1 Adaptive Multi Sensor Online Calibration for Life Long Autonomous System

In Chapter 2, we manually measured the intrinsic and extrinsic parameters of proposed multi-sensor data collection platform. The research findings highlight the potential of integrating multiple sensors to significantly enhance a system's perception and environmental understanding. However, this integration presents challenges. The growing number of sensors increases system complexity, posing difficulties for long-term monitoring of intrinsic and extrinsic parameter, which are pivotal for achieving precision and robustness in state estimation within multi-sensor fusion algorithms.

Previous studies have introduced various offline approaches for estimating spatial and temporal sensor relationships [192; 193]. Recent efforts have delved into online extrinsic calibration for multi-LiDAR systems, as proposed by Jiao et al. [35]. Lee et al. presented an efficient real-time Multi-State Constraint Kalman Filter (MSCKF)-based multi-sensor aided inertial navigation system (MINS) [194], integrating data from cameras, wheel encoders, GPS, and 3D LiDAR with online calibration to ensure optimal performance. Online intrinsic calibration has also been addressed in [195], and camera-IMU extrinsic calibration discussed in [196] for visual-inertial navigation systems (VINS).

Despite these advancements, the escalating complexity of multi-sensor platforms underscores the need for further research in adaptive online sensor calibration. Addressing this challenge requires developing techniques that facilitate real-time and effective calibration of intrinsic, extrinsic, and temporal parameters across diverse sensor types and modalities.

7.2.2 Global Localization in Dynamic Environments

Chapter 3 of the thesis introduces the implementation of real-time, low-cost segmentation, and graph search-based global localization methods in challenging forest environments. The rapid progress in deep learning techniques has led to the emergence of more data-driven approaches utilizing semantic object-level global localization methods.

Multiple approaches involve the conversion of raw point cloud data into a unique graph representation by aggregating semantic information. These methods employ deep learning-based techniques to compare the similarity of semantic graphs for place recognition tasks [197; 23]. Several research studies tackle global localization challenges by computing discriminative 3D point cloud descriptors using advanced deep learning techniques [198; 199]. These methods leverage deep neural networks, such as PointNet, as an initial processing step to extract local features, which are then aggregated into powerful global descriptors [200; 201]. Additionally, beyond point-based descriptors, sparse voxelized point cloud descriptors and the 3D FPN architecture [202; 203]. These deep learning-based methods introduce powerful tools to extract both local and global features and have demonstrated promising results.

However, a fundamental requirement for creating a robust, lifelong autonomous system is the capacity to adapt to dynamic and changing surroundings. Chapter 3 challenges the assumption of a static environment, acknowledging real-world scenarios involving significant alterations, such as terrain changes due to harvesters navigating through forests or construction robots modifying their surroundings [204; 205]. Devising strategies to achieve robust global localization in such dynamic environments remains an open challenge that has not yet been adequately addressed.

7.2.3 Enhancing Human-robot Interaction Through Multi-Modal Multi-LiDAR Fusion

In chapters 5 and chapter 6, our research successfully implemented tightly coupled multi-LiDAR inertial dense mapping and dynamic object detection and tracking methods, utilizing dense measurements from solid-state LiDARs. Notably, solid-state LiDAR holds significant potential in the domain of human-robot interaction, offering heightened situational awareness capabilities.

Advancements in accurate and user-friendly technologies have progressed alongside sensor technology. Integrating hand gesture recognition into human-robot interaction has the potential to transform communication, streamlining collaboration. This innovation promises increased efficiency across applications and tackles challenges. Li et al. introduced pioneering work [206], presenting a LiDAR-captured human motion dataset, surpassing RGB-based methods with a deep learning-based approach [207; 208]. Similarly, Chamorro et al. showcased real-time robot tele-

portation using deep learning-based LiDAR gesture recognition [209]. They effectively predicted gestures using Long Short-Term Memory networks. Additionally, Moysiadis et al. proposed an integrated real-time hand gesture recognition framework for agricultural human-robot interaction [210]. This technology promises smarter and more user-friendly human-robot interactions.

Future research may emphasize solid-state LiDAR or multi-modal LiDAR-based gesture recognition to enhance human-computer interaction. Leveraging solid-state LiDAR capabilities can strengthen collaboration between humans and robots, fostering more intuitive interactions. This progress holds the potential to revolutionize human-robot collaboration, enabling seamless interactions in dynamic environments across industries.

List of References

[1] Diksha Singh, Esha Trivedi, Yukti Sharma, and Vandana Niranjan. Turtlebot: Design and hardware component selection. In *2018 International Conference on Computing, Power and Communication Technologies (GUCON)*, pages 805–809. IEEE, 2018.

[2] Shinpei Kato, Shota Tokunaga, Yuya Maruyama, Seiya Maeda, Manato Hirabayashi, Yuki Kitsukawa, Abraham Monrroy, Tomohito Ando, Yusuke Fujii, and Takuya Azumi. Autoware on board: Enabling autonomous vehicles with embedded systems. In *2018 ACM/IEEE 9th International Conference on Cyber-Physical Systems (ICCPS)*, pages 287–296. IEEE, 2018.

[3] Alexander Carballo, Jacob Lambert, Abraham Monrroy, David Wong, Patiphon Narksri, Yuki Kitsukawa, Eijiro Takeuchi, Shinpei Kato, and Kazuya Takeda. Libre: The multiple 3d lidar dataset. In *2020 IEEE Intelligent Vehicles Symposium (IV)*, pages 1094–1101. IEEE, 2020.

[4] Thinal Raj, Fazida Hanim Hashim, Aqilah Baseri Huddin, Mohd Faisal Ibrahim, and Aini Hussain. A survey on lidar scanning mechanisms. *Electronics*, 9(5):741, 2020.

[5] Martin Buehler, Karl Iagnemma, and Sanjiv Singh. *The DARPA urban challenge: autonomous vehicles in city traffic*, volume 56. springer, 2009.

[6] Hartmut Surmann, Andreas Nüchter, and Joachim Hertzberg. An autonomous mobile robot with a 3d laser range finder for 3d exploration and digitalization of indoor environments. *Robotics and Autonomous Systems*, 45(3-4):181–198, 2003.

[7] Qingqing Li, Jorge Peña Queralta, Tuan Nguyen Gia, Zhuo Zou, and Tomi Westerlund. Multi-sensor fusion for navigation and mapping in autonomous vehicles: Accurate localization in urban environments. *Unmanned Systems*, 8(03):229–237, 2020.

[8] Nina Varney, Vijayan K Asari, and Quinn Graehling. Dales: a large-scale aerial lidar data set for semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 186–187, 2020.

[9] Xu Liu, Guilherme V Nardari, Fernando Cladera Ojeda, Yuezhan Tao, Alex Zhou, Thomas Donnelly, Chao Qu, Steven W Chen, Roseli AF Romero, Camillo J Taylor, and Vijay Kumar. Large-scale autonomous flight with real-time semantic slam under dense forest canopy. *IEEE Robotics and Automation Letters (RA-L)*, 2022.

[10] Juntao Yang, Zhizhong Kang, Sai Cheng, Zhou Yang, and Perpetual Hope Akwensi. An individual tree segmentation method based on watershed algorithm and three-dimensional spatial distribution analysis from airborne lidar point clouds. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 13:1055–1067, 2020.

[11] Ignacio Vizzo, Tiziano Guadagnino, Benedikt Mersch, Louis Wiesmann, Jens Behley, and Cyrill Stachniss. Kiss-icp: In defense of point-to-point icp simple, accurate, and robust registration if done the right way. *IEEE Robotics and Automation Letters*, 2023.

[12] Aleksandr Segal, Dirk Haehnel, and Sebastian Thrun. Generalized-icp. In *Robotics: science and systems*, volume 2, page 435. Seattle, WA, 2009.

[13] Ji Zhang and Sanjiv Singh. Loam: Lidar odometry and mapping in real-time. In *Robotics: Science and systems*, volume 2, pages 1–9. Berkeley, CA, 2014.

[14] Tixiao Shan and Brendan Englot. Lego-loam: Lightweight and ground-optimized lidar odometry and mapping on variable terrain. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4758–4765. IEEE, 2018.

[15] Jiarong Lin and Fu Zhang. Loam livox: A fast, robust, high-precision LiDAR odometry and mapping package for LiDARs of small FoV. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3126–3131. IEEE, 2020.

[16] Kailai Li, Meng Li, and Uwe D Hanebeck. Towards high-performance solid-state-lidar-inertial odometry and mapping. *IEEE Robotics and Automation Letters*, 6(3):5167–5174, 2021.

[17] Wei Xu and Fu Zhang. Fast-lio: A fast, robust lidar-inertial odometry package by tightly-coupled iterated kalman filter. *IEEE Robotics and Automation Letters*, 6(2), 2021.

[18] Jiarong Lin, Xiyuan Liu, and Fu Zhang. A decentralized framework for simultaneous calibration, localization and mapping with multiple lidars. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4870–4877. IEEE, 2020.

[19] Huan Yin, Xuecheng Xu, Sha Lu, Xieyuanli Chen, Rong Xiong, Shaojie Shen, Cyrill Stachniss, and Yue Wang. A survey on global lidar localization: Challenges, advances and open problems. *arXiv preprint arXiv:2302.07433*, 2023.

[20] Rainer Kümmerle, Giorgio Grisetti, Hauke Strasdat, Kurt Konolige, and Wolfram Burgard. g 2 o: A general framework for graph optimization. In *2011 IEEE International Conference on Robotics and Automation*, pages 3607–3613. IEEE, 2011.

[21] Tixiao Shan, Brendan Englot, Drew Meyers, Wei Wang, Carlo Ratti, and Daniela Rus. Lio-sam: Tightly-coupled lidar inertial odometry via smoothing and mapping. In *2020 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 5135–5142. IEEE, 2020.

[22] Hao Dong, Xieyuanli Chen, Simo Särkkä, and Cyrill Stachniss. Online pole segmentation on range images for long-term lidar localization in urban environments. *Robotics and Autonomous Systems*, 159:104283, 2023.

[23] Xin Kong, Xuemeng Yang, Guangyao Zhai, Xiangrui Zhao, Xianfang Zeng, Mengmeng Wang, Yong Liu, Wanlong Li, and Feng Wen. Semantic graph based place recognition for 3d point clouds. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8216–8223. IEEE, 2020.

[24] Lin Li, Xin Kong, Xiangrui Zhao, Tianxin Huang, Wanlong Li, Feng Wen, Hongbo Zhang, and Yong Liu. Ssc: Semantic scan context for large-scale place recognition. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2092–2099. IEEE, 2021.

[25] Giseop Kim, Sunwook Choi, and Ayoung Kim. Scan context++: Structural place recognition robust to rotation and lateral variations in urban environments. *IEEE Transactions on Robotics*, 38(3):1856–1874, 2021.

[26] Marwan Hussein, Matthew Renner, and Karl Iagnemma. Global localization of autonomous robots in forest environments. *Photogrammetric Engineering & Remote Sensing*, 81(11):839–846, 2015.

[27] Georgi Tinchev, Simona Nobili, and Maurice Fallon. Seeing the wood for the trees: Reliable localization in urban and natural environments. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8239–8246. IEEE, 2018.

[28] Haoyang Ye, Yuying Chen, and Ming Liu. Tightly coupled 3d lidar inertial odometry and mapping. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 3144–3150. IEEE, 2019.

[29] Tixiao Shan, Brendan Englot, Carlo Ratti, and Daniela Rus. Lvi-sam: Tightly-coupled lidar-visual-inertial odometry via smoothing and mapping. In *2021 IEEE international conference on robotics and automation (ICRA)*, pages 5692–5698. IEEE, 2021.

[30] Shibo Zhao, Hengrui Zhang, Peng Wang, Lucas Nogueira, and Sebastian Scherer. Super odometry: Imu-centric lidar-visual-inertial estimator for challenging environments. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8729–8736. IEEE, 2021.

[31] Jiarong Lin and Fu Zhang. Loam livox: A fast, robust, high-precision lidar odometry and mapping package for lidars of small fov. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3126–3131. IEEE, 2020.

[32] Zheng Liu and Fu Zhang. Balm: Bundle adjustment for lidar mapping. *IEEE Robotics and Automation Letters*, 6(2):3184–3191, 2021.

[33] Wei Xu and Fu Zhang. Fast-lio: A fast, robust lidar-inertial odometry package by tightly-coupled iterated kalman filter. *IEEE Robotics and Automation Letters*, 6(2):3317–3324, 2021.

[34] Yuewen Zhu, Chunran Zheng, Chongjian Yuan, Xu Huang, and Xiaoping Hong. Camvox: A low-cost and accurate lidar-assisted visual slam system. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5049–5055. IEEE, 2021.

[35] Jianhao Jiao, Haoyang Ye, Yilong Zhu, and Ming Liu. Robust odometry and mapping for multi-lidar systems with online extrinsic calibration. *IEEE Transactions on Robotics*, 2021.

[36] Thien-Minh Nguyen, Shenghai Yuan, Muqing Cao, Lyu Yang, Thien Hoang Nguyen, and Lihua Xie. Miliom: Tightly coupled multi-input lidar-inertia odometry and mapping. *IEEE Robotics and Automation Letters*, 6(3), 2021.

[37] Haoyang Ye, Yuying Chen, and Ming Liu. Tightly coupled 3d lidar inertial odometry and mapping. In *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019.

[38] Yan Xu, Junyi Lin, Jianping Shi, Guofeng Zhang, Xiaogang Wang, and Hongsheng Li. Robust self-supervised lidar odometry via representative structure discovery and 3d inherent error modeling. *IEEE Robotics and Automation Letters*, 2022.

[39] Ha Sier, Qingqing Li, Xianjia Yu, Jorge Peña Queralta, Zhuo Zou, and Tomi Westerlund. A benchmark for multi-modal lidar slam with ground truth in gnss-denied environments. *Remote Sensing*, 15(13):3314, 2023.

[40] Pengxin Chen, Wenzhong Shi, Sheng Bao, Muyang Wang, Wenzheng Fan, and Haodong Xiang. Low-drift odometry, mapping and ground segmentation using a backpack lidar system. *IEEE Robotics and Automation Letters*, 6(4), 2021.

[41] Tong Qin, Peiliang Li, and Shaojie Shen. Vins-mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Transactions on Robotics*, 34(4):1004–1020, 2018.

[42] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013.

[43] Li Qingqing, Yu Xianjia, Jorge Pena Queralta, and Tomi Westerlund. Multi-modal lidar dataset for benchmarking general-purpose localization and mapping algorithms. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3837–3844. IEEE, 2022.

[44] Naoki Akai, Luis Yoichi Morales, Eijiro Takeuchi, Yuki Yoshihara, and Yoshiki Ninomiya. Robust localization using 3d ndt scan matching with experimentally determined uncertainty and road marker matching. In *2017 IEEE Intelligent Vehicles Symposium (IV)*, pages 1356–1363. IEEE, 2017.

[45] Jesse Levinson, Michael Montemerlo, and Sebastian Thrun. Map-based precision vehicle localization in urban environments. In *Robotics: Science and Systems*, volume 4, page 1. Citeseer, 2007.

[46] Qingqing Li, Paavo Nevalainen, Jorge Peña Queralta, Jukka Heikkonen, and Tomi Westerlund. Localization in unstructured environments: Towards autonomous robots in forests with delaunay triangulation. *Remote Sensing*, 12(11):1870, 2020.

[47] Wei Wang, Jun Liu, Chenjie Wang, Bin Luo, and Cheng Zhang. Dv-loam: Direct visual lidar odometry and mapping. *Remote Sensing*, 13(16):3340, 2021.

[48] Junyi Ma, Jun Zhang, Jintao Xu, Rui Ai, Weihao Gu, and Xieyuanli Chen. Overlaptransformer: An efficient and yaw-angle-invariant transformer network for lidar-based place recognition. *IEEE Robotics and Automation Letters*, 7(3):6958–6965, 2022.

[49] Qingqing Li. Multi-modal lidars dataset for benchmarking general-purpose localization and mapping algorithms, 2022. URL `https://github.com/TIERS/tiers-lidars-dataset`.

[50] Qingqing Li. A benchmark for multi-modal lidar slam with ground truth in gnss-denied environments, 2023. URL `https://github.com/TIERS/tiers-lidars-dataset-enhanced`.

[51] Will Maddern, Geoffrey Pascoe, Chris Linegar, and Paul Newman. 1 year, 1000 km: The oxford robotcar dataset. *The International Journal of Robotics Research*, 36(1):3–15, 2017.

[52] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition(CVPR)*, pages 11621–11631, 2020.

[53] Zhi Yan, Li Sun, Tomáš Krajník, and Yassine Ruichek. Eu long-term dataset with multiple sensors for autonomous driving. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10697–10704. IEEE, 2020.

[54] Nicholas Carlevaris-Bianco, Arash K Ushani, and Ryan M Eustice. University of michigan north campus long-term vision and lidar dataset. *The International Journal of Robotics Research*, 35 (9):1023–1035, 2016.

[55] Milad Ramezani, Yiduo Wang, Marco Camurri, David Wisth, Matias Mattamala, and Maurice Fallon. The newer college dataset: Handheld lidar, inertial and vision with ground truth. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4353–4360. IEEE, 2020.

[56] Jie Yin, Ang Li, Tao Li, Wenxian Yu, and Danping Zou. M2dgr: A multi-sensor and multi-scenario slam dataset for ground robots. *IEEE Robotics and Automation Letters*, 7(2):2266–2273, 2021.

[57] Maggie Wigness, Sungmin Eum, John G Rogers, David Han, and Heesung Kwon. A rugd dataset for autonomous navigation and visual perception in unstructured outdoor environments. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5000–5007. IEEE, 2019.

[58] Chaoyue Niu, Danesh Tarapore, and Klaus-Peter Zauner. Low-viewpoint forest depth dataset for sparse rover swarms. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8035–8040. IEEE, 2020.

[59] Gabriela Gresenz, Jules White, and Douglas C Schmidt. An off-road terrain dataset including images labeled with measures of terrain roughness. In *2021 IEEE International Conference on Autonomous Systems (ICAS)*, pages 1–5. IEEE, 2021.

[60] Zachary Pezzementi, Trenton Tabor, Peiyun Hu, Jonathan K Chang, Deva Ramanan, Carl Wellington, Benzun P Wisely Babu, and Herman Herman. Comparing apples and oranges: Off-road pedestrian detection on the national robotics engineering center agricultural person-detection dataset. *Journal of Field Robotics*, 35(4):545–563, 2018.

[61] Pengchuan Xiao, Zhenlei Shao, Steven Hao, Zishuo Zhang, Xiaolin Chai, Judy Jiao, Zesong Li, Jian Wu, Kai Sun, Kun Jiang, et al. Pandaset: Advanced sensor suite dataset for autonomous driving. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pages 3095–3101. IEEE, 2021.

[62] Jinyong Jeong, Younggun Cho, Young-Sik Shin, Hyunchul Roh, and Ayoung Kim. Complex urban lidar data set. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6344–6351. IEEE, 2018.

[63] Peter Biber and Wolfgang Straer. The normal distributions transform: A new approach to laser scan matching. In *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)(Cat. No. 03CH37453)*, volume 3, pages 2743–2748. IEEE, 2003.

[64] Jürgen Sturm, Nikolas Engelhard, Felix Endres, Wolfram Burgard, and Daniel Cremers. A benchmark for the evaluation of rgb-d slam systems. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems(IROS)*, pages 573–580. IEEE, 2012.

[65] Yun Chang, Kamak Ebadi, Christopher E Denniston, Muhammad Fadhil Ginting, Antoni Rosinol, Andrzej Reinke, Matteo Palieri, Jingnan Shi, Arghya Chatterjee, Benjamin Morrell, et al. Lamp 2.0: A robust multi-robot slam system for operation in challenging large-scale underground environments. *IEEE Robotics and Automation Letters*, 7(4):9175–9182, 2022.

[66] Ville Kankare, Jari Vauhkonen, Topi Tanhuanpaa, Markus Holopainen, Mikko Vastaranta, Marianna Joensuu, Anssi Krooks, Juha Hyyppa, Hannu Hyyppa, Petteri Alho, and Risto Viitala. Accuracy in estimation of timber assortments and stem distribution - a comparison of airborne and

terrestrial laser scanning techniques. *ISPRS Journal of Photogrammetry and Remote Sensing*, 97:89–97, 2014. ISSN 0924-2716. doi: 10.1016/j.isprsjprs.2014.08.008.

[67] Thomas Hellström, P. Lärkeryd, Tomas Nordfjell, and Ola Ringdahl. Autonomous forest vehicles: Historic, envisioned, and state-of-the-art. *International Journal of Forest Engineering*, 20: 31–38, 01 2009.

[68] Fang Liao, Shupeng Lai, Yuchao Hu, Jinqiang Cui, Jian Liang Wang, Rodney Teo, and Feng Lin. 3d motion planning for uavs in gps-denied unknown forest environment. In *2016 IEEE Intelligent Vehicles Symposium (IV)*, pages 246–251. IEEE, 2016.

[69] Yulun Tian, Katherine Liu, Kyel Ok, Loc Tran, Danette Allen, Nicholas Roy, and Jonathan P How. Search and rescue under the forest canopy using multiple uas. In *International Symposium on Experimental Robotics*, pages 140–152. Springer, 2018.

[70] Ola Ringdahl, Ola Lindroos, Thomas Hellström, Dan Bergström, Dimitris Athanassiadis, and Tomas Nordfjell. Path tracking in forest terrain by an autonomous forwarder. *Scandinavian Journal of Forest Research*, 26(4):350–359, 2011. doi: 10.1080/02827581.2011.566889. URL `https://doi.org/10.1080/02827581.2011.566889`.

[71] Xiaorui Zhu, Youngshik Kim, Mark A. Minor, and Chunxin Qiu. *Autonomous Mobile Robots in Unknown Outdoor Environments*. CRC Press, Inc., USA, 1st edition, 2017. ISBN 1498740553.

[72] Claudine Badue, Rânik Guidolini, Raphael Vivacqua Carneiro, Pedro Azevedo, Vinicius B Cardoso, Avelino Forechi, Luan Jesus, Rodrigo Berriel, Thiago M Paixao, Filipe Mutz, et al. Self-driving cars: A survey. *Expert Systems with Applications*, 165:113816, 2021.

[73] Rajeev Thakur. Scanning lidar in advanced driver assistance systems and beyond: Building a road map for next-generation lidar technology. *IEEE Consumer Electronics Magazine*, 5(3): 48–54, 2016.

[74] Li Qingqing, Jorge Peña Queralta, Tuan Nguyen Gia, Zhuo Zou, and Tomi Westerlund. Multi sensor fusion for navigation and mapping in autonomous vehicles: Accurate localization in urban environments. *Unammned Systems (to appear). Accepted and presented at the 9th IEEE CIS-RAM conference.*, 2020.

[75] Wuming Zhang, Peng Wan, Tiejun Wang, Shangshu Cai, Yiming Chen, Xiuliang Jin, and Guangjian Yan. A novel approach for the detection of standing tree stems from plot-level terrestrial laser scanning data. *Remote sensing*, 11(2):211, 2019.

[76] Marek Pierzchała, Philippe Giguére, and Rasmus Astrup. Mapping forests using an unmanned ground vehicle with 3D LiDAR and graph-SLAM. *Computers and Electronics in Agriculture*, 145:217 – 225, 2018. ISSN 0168-1699. doi: https://doi.org/10.1016/j.compag.2017.12.034. URL `http://www.sciencedirect.com/science/article/pii/S0168169917301631`.

[77] Xinlian Liang, Ville Kankare, Juha Hyyppä, Yunsheng Wang, Antero Kukko, Henrik Haggrén, Xiaowei Yu, Harri Kaartinen, Anttoni Jaakkola, Fengying Guan, Markus Holopainen, and Mikko Vastaranta. Terrestrial laser scanning in forest inventories. *ISPRS Journal of Photogrammetry and Remote Sensing*, 115:63 – 77, 2016. ISSN 0924-2716. doi: https://doi.org/10.1016/j.isprsjprs.2016.01.006. URL `http://www.sciencedirect.com/science/article/pii/S0924271616000204`. Theme issue 'State-of-the-art in photogrammetry, remote sensing and spatial information science'.

[78] Mikko Miettinen, Matti Öhman, Arto Visala, and Pekka Forsman. Simultaneous localization and mapping for forest harvesters. In *IEEE International Conference on Robotics and Automation, Rooma April 2007*, pages 517–522, 2007.

[79] Jian Tang, Yuwei Chen, Antero Kukko, Harri Kaartinen, Anttoni Jaakkola, Ehsan Khoramshahi, Teemu Hakala, Juha Hyyppä, Markus Holopainen, and Hannu Hyyppä. Slam-aided stem mapping for forest inventory with small-footprint mobile lidar. *Forests*, 6(12):4588–4606, 2015.

[80] Ayumu Tominaga, Hayashi Eiji, and Abbe Mowshowitz. Development of navigation system in field robot for forest management. In *2018 Joint 10th International Conference on Soft Computing and Intelligent Systems (SCIS) and 19th International Symposium on Advanced Intelligent Systems (ISIS)*, pages 1142–1147. IEEE, 2018.

[81] Steven W Chen, Guilherme V Nardari, Elijah S Lee, Chao Qu, Xu Liu, Roseli Ap Francelin Romero, and Vijay Kumar. Sloam: Semantic lidar odometry and mapping for forest inventory. *IEEE Robotics and Automation Letters*, 5(2):612–619, 2020.

[82] Keisuke Yoneda, Naoki Suganuma, Ryo Yanase, and Mohammad Aldibaja. Automated driving recognition technologies for adverse weather conditions. *IATSS Research*, 2019.

[83] Aji Resindra Widya, Akihiko Torii, and Masatoshi Okutomi. Structure from motion using dense cnn features with keypoint relocalization. *IPSJ Transactions on Computer Vision and Applications*, 10:1–7, 2018.

[84] Tomastik et al. Horizontal accuracy and applicability of smartphone gnss positioning in forests. *Forestry: An International Journal of Forest Research*, 90(2):187–198, 2016.

[85] Eloise G. Zimbelman and Robert F. Keefe. Real-time positioning in logging: Effects of forest stand characteristics, topography, and line-of-sight obstructions on gnss-rf transponder accuracy and radio signal propagation. *PLOS ONE*, 13(1):1–17, 01 2018. doi: 10.1371/journal.pone.0191017. URL `https://doi.org/10.1371/journal.pone.0191017`.

[86] Szymon Rusinkiewicz and Marc Levoy. Efficient variants of the icp algorithm. In *Proceedings Third International Conference on 3-D Digital Imaging and Modeling*, pages 145–152. IEEE, 2001.

[87] Aleksandr Segal, Dirk Haehnel, and Sebastian Thrun. Generalized-icp. In *Robotics: science and systems*, volume 2, page 435. Seattle, WA, 2009.

[88] Dirk Holz, Alexandru E Ichim, Federico Tombari, Radu B Rusu, and Sven Behnke. Registration with the point cloud library: A modular framework for aligning in 3-d. *IEEE Robotics & Automation Magazine*, 22(4):110–124, 2015.

[89] Martin Lauer, Sascha Lange, and Martin Riedmiller. Calculating the perfect match: an efficient and accurate approach for robot self-localization. In *Robot Soccer World Cup*. Springer, 2005.

[90] Peter Biber and Wolfgang Straßer. The normal distributions transform: A new approach to laser scan matching. In *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)(Cat. No. 03CH37453)*, volume 3. IEEE, 2003.

[91] Jiaolong Yang, Hongdong Li, Dylan Campbell, and Yunde Jia. Go-icp: A globally optimal solution to 3d icp point-set registration. *IEEE Trans. Pattern Anal. Mach. Intell.*, 38 (11):2241–2254, 11 2016. ISSN 0162-8828. doi: 10.1109/TPAMI.2015.2513405. URL `https://doi.org/10.1109/TPAMI.2015.2513405`.

[92] Pedro Nunez, Ricardo Vazquez-Martin, Jose C del Toro, Antonio Bandera, and Francisco Sandoval. Feature extraction from laser scan data based on curvature estimation for mobile robotics. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pages 1167–1172. IEEE, 2006.

[93] Aparajithan Sampath and Jie Shan. Clustering based planar roof extraction from lidar data. In *American Society for Photogrammetry and Remote Sensing Annual Conference, Reno, Nevada, May*, pages 1–6, 2006.

[94] Jing Liang, Jixian Zhang, Kazhong Deng, Zhengjun Liu, and Qunshan Shi. A new power-line extraction method based on airborne lidar point cloud data. In *2011 International Symposium on Image and Data Fusion*, pages 1–4. IEEE, 2011.

[95] Sebastian Thrun, Daphne Koller, Zoubin Ghahmarani, and Hugh Durrant-Whyte. Slam updates require constant time. In *Workshop on the Algorithmic Foundations of Robotics*, pages 1–20. Citeseer, 2002.

[96] Yufeng Liu and Sebastian Thrun. Results for outdoor-slam using sparse extended information filters. In *2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422)*, volume 1, pages 1227–1233. IEEE, 2003.

[97] Iwan Ulrich and Illah Nourbakhsh. Appearance-based place recognition for topological localization. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, volume 2, pages 1023–1029. Ieee, 2000.

[98] Jun Chen, Chaomin Luo, Mohan Krishnan, Mark Paulik, and Yipeng Tang. An enhanced dynamic delaunay triangulation-based path planning algorithm for autonomous mobile robot navigation. In *Intelligent Robots and Computer Vision XXVII: Algorithms and Techniques*, volume 7539, page 75390P. International Society for Optics and Photonics, 2010.

[99] Marian Himstedt, Jan Frost, Sven Hellbach, Hans-Joachim Böhme, and Erik Maehle. Large scale place recognition in 2d lidar scans using geometrical landmark relations. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5030–5035. IEEE, 2014.

[100] Simon Lynen, Michael Bosse, and Roland Siegwart. Trajectory-based place-recognition for efficient large scale localization. *International Journal of Computer Vision*, 124(1):49–64, 2017.

[101] Michael Bosse and Robert Zlot. Place recognition using keypoint voting in large 3d lidar datasets. In *2013 IEEE International Conference on Robotics and Automation*, pages 2677–2684. IEEE, 2013.

[102] Michael Bosse and Robert Zlot. Keypoint design and evaluation for place recognition in 2d lidar maps. *Robotics and Autonomous Systems*, 57(12):1211–1224, 2009.

[103] Herbert Edelsbrunner. Triangulations and meshes in computational geometry. *Acta Numerica 2000*, 9:133 – 213, 01 2000. doi: 10.1017/S0962492900001331.

[104] Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, 1975.

[105] Radu Bogdan Rusu. Semantic 3d object maps for everyday manipulation in human living environments. *KI-Künstliche Intelligenz*, 24(4):345–348, 2010.

[106] Der-Tsai Lee and Bruce J Schachter. Two algorithms for constructing a delaunay triangulation. *International Journal of Computer & Information Sciences*, 9(3):219–242, 1980.

[107] Z. Arzoumanian, Jason Holmberg, and Brad Norman. An astronomical pattern-matching algorithm for computer-aided identification of whale sharks rhincodon typus. *Journal of Applied Ecology*, 42:999 – 1011, 09 2005. doi: 10.1111/j.1365-2664.2005.01117.x.

[108] David Sinclair. S-hull: a fast radial sweep-hull routine for Delaunay triangulation. *arXiv e-prints*, art. arXiv:1604.01428, 2016.

[109] Fani Boukouvala, Ruth Misener, and Christodoulos Floudas. Global optimization advances in mixed-integer nonlinear programming, minlp, and constrained derivative-free optimization, cdfo. *European Journal of Operational Research*, 252, 12 2015. doi: 10.1016/j.ejor.2015.12.018.

[110] Ryan W Wolcott and Ryan M Eustice. Visual localization within lidar maps for automated urban driving. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 176–183. IEEE, 2014.

[111] Ahti Heinla, Allan Martinson, Kalle-rasmus Volkov, Andrew Macks, Lindsay Roberts, Indrek Mandre, Märt Liivik, Tiit Liivik, and Ivo Liivik. Method and system for autonomous or semi-autonomous delivery, 11 2021. US Patent 11,164,273.

[112] Stephen Se, David G Lowe, and James J Little. Vision-based global localization and mapping for mobile robots. *IEEE Transactions on robotics*, 21(3):364–375, 2005.

[113] Emilio Garcia-Fidalgo and Alberto Ortiz. Vision-based topological mapping and localization methods: A survey. *Robotics and Autonomous Systems*, 64, 2015.

[114] Cristiano Premebida, Luis Garrote, Alireza Asvadi, A Pedro Ribeiro, and Urbano Nunes. High-resolution lidar-based depth mapping using bilateral filter. In *2016 IEEE 19th international conference on intelligent transportation systems (ITSC)*, pages 2469–2474. IEEE, 2016.

[115] M. John. Google cars drive themselves. *The New York Times. URL http://www. nytimes. com/2010/10/10/science/10google. html*, 2010.

[116] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*, 2016.

[117] Mark Harris. Documents confirm apple is building self-driving car. *The Guardian*, 14, 2015.

[118] Nathan A Greenblatt. Self-driving cars and the law. *IEEE spectrum*, 53(2):46–51, 2016.

[119] Shinpei Kato, Eijiro Takeuchi, Yoshio Ishiguro, Yoshiki Ninomiya, Kazuya Takeda, and Tsuyoshi Hamada. An open approach to autonomous vehicles. *IEEE Micro*, 35(6):60–68, 2015.

[120] Turn Zhang. Lidar-based road and road-edge detection. In *2010 IEEE Intelligent Vehicles Symposium*, pages 845–848, 06 2010. doi: 10.1109/IVS.2010.5548134.

[121] Erik Hofmann and Marco Rüsch. Industry 4.0 and the current status as well as future prospects on logistics. *Computers in Industry*, 89:23–34, 2017.

[122] Heike Flämig. Autonomous vehicles and autonomous driving in freight transport. In *Autonomous driving*, pages 365–385. Springer, 2016.

[123] Nils Boysen, Stefan Schwerdfeger, and Felix Weidinger. Scheduling last-mile deliveries with truck-based autonomous robots. *European Journal of Operational Research*, 271(3):1085–1099, 2018.

[124] Ji Zhang and Sanjiv Singh. Visual-lidar odometry and mapping: Low-drift, robust, and fast. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2174–2181. IEEE, 2015.

[125] Hyunggi Cho, Young-Woo Seo, BVK Vijaya Kumar, and Ragunathan Raj Rajkumar. A multi-sensor fusion system for moving object detection and tracking in urban driving environments. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1836–1843. IEEE, 2014.

[126] Pavel González Prieto, Fernando Martín, Luis Moreno, and Juan Carballeira. Dendt: 3d-ndt scan matching with differential evolution. In *2017 25th Mediterranean Conference on Control and Automation (MED)*, pages 719–724, 07 2017. doi: 10.1109/MED.2017.7984203.

[127] Chris Rizos, Volker Janssen, Craig Roberts, and Th Grinter. Precise point positioning: Is the era of differential gnss positioning drawing to an end? *University Of Tasmania*, 2012.

[128] The KITTI Vision Benchmark Suite. Visual odometry and slam evaluation, 2012. URL `http://www.cvlibs.net/datasets/kitti/eval_odometry.php`.

[129] Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y Ng. Ros: an open-source robot operating system. In *ICRA workshop on open source software*. Kobe, Japan, 2009.

[130] Radu Bogdan Rusu and Steve Cousins. 3D is here: Point cloud library (PCL). In *2011 IEEE International Conference on Robotics and Automation*, 05 2011. doi: 10.1109/ICRA.2011.5980567.

[131] Weisong Wen, Li-Ta Hsu, and Guohao Zhang. Performance analysis of ndt-based graph slam for autonomous vehicle in diverse typical driving scenarios of hong kong. *Sensors*, 18(11):3928, 2018.

[132] Li Qingqing, Yu Xianjia, Jorge Pena Queralta, and Tomi Westerlund. Multi-modal lidar dataset for benchmarking general-purpose localization and mapping algorithms. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022.

[133] TY Lam. Hamilton's quaternions. In *Handbook of algebra*, volume 3, pages 429–454. Elsevier, 2003.

[134] Jian S Dai. Euler–rodrigues formula variations, quaternion conjugation and intrinsic connections. *Mechanism and Machine Theory*, 92:144–152, 2015.

[135] Si Agarwal and Ki Mierle. Ceres solver, 2023. URL `http://ceres-solver.org`.

[136] Frank Dellaert. Factor graphs and gtsam: A hands-on introduction. *Georgia Institute of Technology, Tech. Rep*, 2:4, 2012.

[137] Marco Lixia, Andrea Benigni, Alessandra Flammini, Carlo Muscas, Ferdinanda Ponci, and Antonello Monti. A software-only ptp synchronization for power system state estimation with pmus. *IEEE Transactions on Instrumentation and Measurement*, 61(5):1476–1485, 2012.

[138] Hanjie Liu, Guoliang Liu, Guohui Tian, Shiqing Xin, and Ze Ji. Visual slam based on dynamic object removal. In *2019 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 596–601. IEEE, 2019.

[139] Li Hou, Wanggen Wan, Jenq-Neng Hwang, Rizwan Muhammad, Mingyang Yang, and Kang Han. Human tracking over camera networks: a review. *EURASIP Journal on Advances in Signal Processing*, 2017(1):1–20, 2017.

[140] Mahmudul Hasan, Junichi Hanawa, Riku Goto, Ryota Suzuki, Hisato Fukuda, Yoshinori Kuno, and Yoshinori Kobayashi. Lidar-based detection, tracking, and property estimation: A contemporary review. *Neurocomputing*, 2022.

[141] Li Qingqing, Yu Xianjia, Jorge Pena Queralta, and Tomi Westerlund. Adaptive lidar scan frame integration: Tracking known mavs in 3d point clouds. In *2021 20th International Conference on Advanced Robotics (ICAR)*, pages 1079–1086. IEEE, 2021.

[142] Tomáš Rouček, Martin Pecka, Petr Čížek, Tomáš Petříček, Jan Bayer, Vojtěch Šalanskỳ, Daniel Heřt, Matěj Petrlík, Tomáš Báča, Vojěch Spurnỳ, et al. Darpa subterranean challenge: Multi-robotic exploration of underground environments. In *Modelling and Simulation for Autonomous Systems: 6th International Conference, MESAS 2019, Palermo, Italy, October 29–31, 2019, Revised Selected Papers 6*, pages 274–290. Springer, 2020.

[143] Matěj Petrlík, Tomáš Báča, Daniel Heřt, Matouš Vrba, Tomáš Krajník, and Martin Saska. A robust uav system for operations in a constrained environment. *IEEE Robotics and Automation Letters*, 5(2):2169–2176, 2020.

[144] Ty Nguyen, Kartik Mohta, Camillo J Taylor, and Vijay Kumar. Vision-based multi-mav localization with anonymous relative measurements using coupled probabilistic data association filter. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3349–3355. IEEE, 2020.

[145] Jorge Peña Queralta, Qingqing Li, Fabrizio Schiano, and Tomi Westerlund. Vio-uwb-based collaborative localization and dense scene reconstruction within heterogeneous multi-robot systems. In *2022 International Conference on Advanced Robotics and Mechatronics (ICARM)*, pages 87–94. IEEE, 2022.

[146] Li Qingqing, Jorge Pena Queralta, Tuan Nguyen Gia, and Tomi Westerlund. Offloading monocular visual odometry with edge computing: Optimizing image quality in multi-robot systems. In *Proceedings of the 2019 5th International Conference on Systems, Control and Communications*, pages 22–26, 2019.

[147] Hazim Shakhatreh, Ahmad H Sawalmeh, Ala Al-Fuqaha, Zuochao Dou, Eyad Almaita, Issa Khalil, Noor Shamsiah Othman, Abdallah Khreishah, and Mohsen Guizani. Unmanned aerial vehicles (uavs): A survey on civil applications and key research challenges. *IEEE Access*, 7: 48572–48634, 2019.

[148] Jorge Peña Queralta, Jussi Taipalmaa, Bilge Can Pullinen, Victor Kathan Sarker, Tuan Nguyen Gia, Hannu Tenhunen, Moncef Gabbouj, Jenni Raitoharju, and Tomi Westerlund. Collaborative multi-robot search and rescue: Coordination and perception. *arXiv preprint arXiv:2008.12610 [cs.RO]*, 2020.

[149] Sean Grogan, R Pellerin, and M Gamache. The use of unmanned aerial vehicles and drones in search and rescue operations–a survey. *Proceedings of the PROLOG*, 2018.

[150] William Roberts, Kelly Griendling, Anthony Gray, and D Mavris. Unmanned vehicle collaboration research environment for maritime search and rescue. In *30th Congress of the International Council of the Aeronautical Sciences*. International Council of the Aeronautical Sciences (ICAS) Bonn, Germany, 2016.

[151] Royal Spanish Federation of First Aid and Rescue. National Drownings Report - Informe Nacional de Ahogamientos (INA), 2019.

[152] Joseph Redmon and Ali Farhadi. YOLOv3: An incremental improvement. *arXiv*, 2018.

[153] Suk-Ju Hong, Yunhyeok Han, Sang-Yeon Kim, Ah-Yeong Lee, and Ghiseok Kim. Application of deep-learning methods to bird detection using unmanned aerial vehicle imagery. *Sensors*, 19 (7):1651, 2019.

[154] Rahul Tallamraju, Eric Price, Roman Ludwig, Kamalakar Karlapalem, Heinrich H Bülthoff, Michael J Black, and Aamir Ahmad. Active perception based formation control for multiple aerial vehicles. *IEEE Robotics and Automation Letters*, 4(4):4491–4498, 2019.

[155] D. Gallos and F. Ferrie. Active vision in the era of convolutional neural networks. In *2019 16th Conference on Computer and Robot Vision (CRV)*, pages 81–88, 2019.

[156] Jorge Peña Queralta, Jenni Raitoharju, Tuan Nguyen Gia, Nikolaos Passalis, and Tomi Westerlund. Autosos: Towards multi-uav systems supporting maritime search and rescue with lightweight ai and edge computing. *arXiv preprint arXiv:2005.03409*, 2020.

[157] Anibal Matos, Alfredo Martins, Andre Dias, Bruno Ferreira, José Miguel Almeida, Hugo Ferreira, Guilherme Amaral, Andre Figueiredo, Rui Almeida, and Filipe Silva. Multiple robot operations for maritime search and rescue in eurathlon 2015 competition. In *OCEANS 2016-Shanghai*, pages 1–7. IEEE, 2016.

[158] Johannes Güldenring, Lucas Koring, Philipp Gorczak, and Christian Wietfeld. Heterogeneous multilink aggregation for reliable uav communication in maritime search and rescue missions. In *2019 International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, pages 215–220. IEEE, 2019.

[159] Rudin Konrad, Daniel Serrano, and Pascal Strupler. Unmanned aerial systems. *Search and Rescue Robotics—From Theory to Practice*, pages 37–52, 2017.

[160] Hartmut Surmann, Rainer Worst, Tim Buschmann, Artur Leinweber, Alexander Schmitz, Gerhard Senkowski, and Niklas Goddemeier. Integration of uavs in urban search and rescue missions. In *2019 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, pages 203–209. IEEE, 2019.

[161] Themistoklis Giitsidis, Evangelos G Karakasis, Antonios Gasteratos, and G Ch Sirakoulis. Human and fire detection from high altitude uav images. In *2015 23rd Euromicro International Conference on Parallel, Distributed, and Network-Based Processing*, pages 309–315. IEEE, 2015.

[162] S. Yong and Y. Yeong. Human object detection in forest with deep learning based on drone's vision. In *2018 4th International Conference on Computer and Information Sciences (ICCOINS)*, pages 1–5. IEEE, 2018.

[163] Sankula Likhith Krishna, Guduru Sai Rama Chaitanya, Abbasani Sree Hari Reddy, Arasada Manoj Naidu, SS Poorna, and K Anuraj. Autonomous human detection system mounted on a drone. In *2019 International Conference on Wireless Communications Signal Processing and Networking (WiSPNET)*, pages 335–338. IEEE, 2019.

[164] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Computer Vision and Pattern Recognition*, 2014.

[165] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

[166] R. Bajcsy, Y. Aloimonos, and J.K Tsotsos. Revisiting active perception. *Autonomous Robots*, 42:177—-196, 2018.

[167] P. Ammirato, P. Poirson, E. Park, J. Košecká, and A. C. Berg. A dataset for developing and benchmarking active vision. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1378–1385, 2017.

[168] Juan Sandino, Fernando Vanegas, Felipe González, and Frédéric Maire. Autonomous uav navigation for active perception of targets in uncertain and cluttered environments. In *2020 IEEE Aerospace Conference*, 2020.

[169] D. Falanga, E. Mueggler, M. Faessler, and D. Scaramuzza. Aggressive quadrotor flight through narrow gaps with onboard sensing and computing using active vision. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 5774–5781, 2017.

[170] M. Chessa, S. Murgia, L. Nardelli, S. P. Sabatini, and F. Solari. Bio-inspired active vision for obstacle avoidance. In *2014 International Conference on Computer Graphics Theory and Applications (GRAPP)*, pages 1–8, 2014.

[171] Fangwei Zhong, Peng Sun, Wenhan Luo, Tingyun Yan, and Yizhou Wang. AD-VAT: An asymmetric dueling mechanism for learning visual active tracking. In *International Conference on Learning Representations*, 2019.

[172] Eleftherios Lygouras, Nicholas Santavas, Anastasios Taitzoglou, Konstantinos Tarchanidis, Athanasios Mitropoulos, and Antonios Gasteratos. Unsupervised human detection with an em-

bedded vision system on a fully autonomous uav for search and rescue operations. *Sensors*, 19 (16):3542, 2019.

[173] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

[174] Labelbox. Labelbox, 2019. [Online]. Available: `https://labelbox.com`.

[175] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338, June 2010.

[176] R. Padilla, S. L. Netto, and E. A. B. da Silva. A survey on performance metrics for object-detection algorithms. In *2020 International Conference on Systems, Signals and Image Processing (IWSSIP)*, pages 237–242, 2020.

[177] Michael Blösch, Stephan Weiss, Davide Scaramuzza, and Roland Siegwart. Vision based mav navigation in unknown and unstructured environments. In *2010 IEEE International Conference on Robotics and Automation*, pages 21–28. IEEE, 2010.

[178] Matthias Nieuwenhuisen, David Droeschel, Marius Beul, and Sven Behnke. Autonomous navigation for micro aerial vehicles in complex gnss-denied environments. *Journal of Intelligent & Robotic Systems*, 84:199–216, 2016.

[179] Yu Song, Stephen Nuske, and Sebastian Scherer. A multi-sensor fusion mav state estimation from long-range stereo, imu, gps and barometric sensors. *Sensors*, 17(1):11, 2017.

[180] Hao Xu, Luqi Wang, Yichen Zhang, Kejie Qiu, and Shaojie Shen. Decentralized visual-inertial-uwb fusion for relative state estimation of aerial swarm. *arXiv preprint arXiv:2003.05138*, 2020.

[181] Abel Gawel, Yukai Lin, Théodore Koutros, Roland Siegwart, and Cesar Cadena. Aerial-ground collaborative sensing: Third-person view for teleoperation. In *IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, pages 1–7. IEEE, 2018.

[182] Jorge Pena Queralta, Jussi Taipalmaa, Bilge Can Pullinen, Victor Kathan Sarker, Tuan Nguyen Gia, Hannu Tenhunen, Moncef Gabbouj, Jenni Raitoharju, and Tomi Westerlund. Collaborative multi-robot search and rescue: Planning, coordination, perception, and active vision. *Ieee Access*, 8:191617–191643, 2020.

[183] Ismail Guvenc, Farshad Koohifar, Simran Singh, Mihail L Sichitiu, and David Matolak. Detection, tracking, and interdiction for amateur drones. *IEEE Communications Magazine*, 56(4): 75–81, 2018.

[184] Sebastien Hengy, Martin Laurenzis, Stéphane Schertzer, Alexander Hommes, Franck Kloeppel, Alex Shoykhetbrod, Thomas Geibig, Winfried Johannes, Oussama Rassy, and Frank Christnacher. Multimodal uav detection: Study of various intrusion scenarios. In *Electro-Optical Remote Sensing XI*, volume 10434, pages 203–212. SPIE, 2017.

[185] Matouš Vrba and Martin Saska. Marker-less micro aerial vehicle detection and localization using convolutional neural networks. *IEEE Robotics and Automation Letters (RA-L)*, 5(2):2459–2466, 2020.

[186] Jan Razlaw, Jan Quenzel, and Sven Behnke. Detection and tracking of small objects in sparse 3d laser range data. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2967–2973. IEEE, 2019.

[187] Frank Neuhaus, Tilman Koß, Robert Kohnen, and Dietrich Paulus. Mc2slam: Real-time inertial lidar odometry using two-scan motion compensation. In *Pattern Recognition: 40th German Conference, GCPR 2018, Stuttgart, Germany, October 9-12, 2018, Proceedings 40*, pages 60–72. Springer, 2019.

[188] Wang Shule, Carmen Martínez Almansa, Jorge Peña Queralta, Zhuo Zou, and Tomi Westerlund. Uwb-based localization for multi-uav systems and collaborative heterogeneous multi-robot systems. *Procedia Computer Science*, 175:357–364, 2020.

[189] Carmen Martínez Almansa, Wang Shule, Jorge Pena Queralta, and Tomi Westerlund. Autocalibration of a mobile uwb localization system for ad-hoc multi-robot deployments in gnss-denied environments. *arXiv preprint arXiv:2004.06762*, 2020.

[190] Radu Bogdan Rusu and Steve Cousins. 3d is here: Point cloud library (pcl). In *2011 IEEE international conference on robotics and automation*, pages 1–4. IEEE, 2011.

[191] Jorge Peña Queralta, Carmen Martínez Almansa, Fabrizio Schiano, Dario Floreano, and Tomi Westerlund. Uwb-based system for uav localization in gnss-denied environments: Characterization and dataset. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4521–4528. IEEE, 2020.

[192] Paul Furgale, Joern Rehder, and Roland Siegwart. Unified temporal and spatial calibration for multi-sensor systems. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1280–1286. IEEE, 2013.

[193] Guohang Yan, Zhuochun Liu, Chengjie Wang, Chunlei Shi, Pengjin Wei, Xinyu Cai, Tao Ma, Zhizheng Liu, Zebin Zhong, Yuqian Liu, et al. Opencalib: A multi-sensor calibration toolbox for autonomous driving. *Software Impacts*, 14:100393, 2022.

[194] Woosik Lee, Yulin Yang, and Guoquan Huang. Efficient multi-sensor aided inertial navigation with online calibration. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5706–5712. IEEE, 2021.

[195] Yulin Yang, Patrick Geneva, Xingxing Zuo, and Guoquan Huang. Online imu intrinsic calibration: Is it necessary? *2020 Robotics: Science and Systems*, 2020.

[196] Zhenfei Yang and Shaojie Shen. Monocular visual–inertial state estimation with online initialization and camera–imu extrinsic calibration. *IEEE Transactions on Automation Science and Engineering*, 14(1):39–51, 2016.

[197] Renaud Dubé, Daniel Dugas, Elena Stumm, Juan Nieto, Roland Siegwart, and Cesar Cadena. Segmatch: Segment based place recognition in 3d point clouds. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5266–5272. IEEE, 2017.

[198] Mikaela Angelina Uy and Gim Hee Lee. Pointnetvlad: Deep point cloud based retrieval for large-scale place recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4470–4479, 2018.

[199] Wenxiao Zhang and Chunxia Xiao. Pcan: 3d attention map learning using contextual information for point cloud based retrieval. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12436–12445, 2019.

[200] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017.

[201] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017.

[202] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017.

[203] Jacek Komorowski. Minkloc3d: Point cloud based large-scale place recognition. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1790–1799, 2021.

[204] Marco Hutter, Philipp Leemann, Stefan Stevsic, Andreas Michel, Dominic Jud, Mark Hoepflinger, Roland Siegwart, Ruedi Figi, Christian Caduff, Markus Loher, et al. Towards optimal force distribution for walking excavators. In *2015 international conference on advanced robotics (ICAR)*, pages 295–301. IEEE, 2015.

[205] Nived Chebrolu, Philipp Lottes, Thomas Läbe, and Cyrill Stachniss. Robot localization based on aerial images for precision agriculture tasks in crop fields. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 1787–1793. IEEE, 2019.

[206] Jialian Li, Jingyi Zhang, Zhiyong Wang, Siqi Shen, Chenglu Wen, Yuexin Ma, Lan Xu, Jingyi Yu, and Cheng Wang. Lidarcap: Long-range marker-less 3d human motion capture with lidar point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20502–20512, 2022.

[207] Lan Xu, Weipeng Xu, Vladislav Golyanik, Marc Habermann, Lu Fang, and Christian Theobalt. Eventcap: Monocular 3d capture of high-speed human motions using an event camera. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4968–4978, 2020.

[208] Ruilong Li, Yuliang Xiu, Shunsuke Saito, Zeng Huang, Kyle Olszewski, and Hao Li. Monocular real-time volumetric performance capture. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIII 16*, pages 49–67. Springer, 2020.

[209] Simon Chamorro, Jack Collier, and François Grondin. Neural network based lidar gesture recognition for realtime robot teleoperation. In *2021 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, pages 98–103. IEEE, 2021.

[210] Vasileios Moysiadis, Dimitrios Katikaridis, Lefteris Benos, Patrizia Busato, Athanasios Anagnostis, Dimitrios Kateris, Simon Pearson, and Dionysis Bochtis. An integrated real-time hand gesture recognition framework for human–robot interaction in agriculture. *Applied Sciences*, 12 (16):8160, 2022.

TURUN
YLIOPISTO
UNIVERSITY
OF TURKU

Painosalama, Turku, Finland 2024

ISBN 978-951-29-9735-0 (PRINT)
ISBN 978-951-29-9736-7 (PDF)
ISSN 2736-9390 (PRINT)
ISSN 2736-9684 (ONLINE)