A Multisensor Data Fusion Approach for Simultaneous Localization and Mapping*

Zhekai Jin[1], Yifei Shao[2], Minjoon So[3], Carl Sable[4], Neveen Shlayan [5] and Dirk Martin Luchtenburg[6]

***Abstract*— Simultaneous localization and mapping (SLAM) has been an emerging research topic in the fields of robotics, autonomous driving, and unmanned aerial vehicles over the past thirty years. State of the art SLAM research is often inaccessible for undergraduate student researchers due to expensive hardware and difficult software setup. We present a cost-friendly vehicle research platform and a robust implementation of SLAM. Our SLAM algorithm fuses visual stereo image and 2D light detection and ranging (Lidar) data and uses loop closure for accurate odometry estimation. Our algorithm is benchmarked against other popular SLAM algorithms using the publicly available KITTI dataset and shown to be very accurate. For educational purposes, we publicly share the models and code presented in this work*.**

I. INTRODUCTION

Simultaneous localization and mapping (SLAM) is the process by which a mobile robot can build a map of an environment and at the same time use this map to compute its own location [7]. In other words, it comprises the simultaneous estimation of the state of a robot equipped with on-board sensors, and the construction of a model (the map) of the environment that the sensors are perceiving. In simple instances, the robot state is described by its pose (position and orientation), although other quantities may be included in the state, such as the robot's velocity, sensor biases, and calibration parameters. The map, on the other hand, is a representation of aspects of interest (e.g., the position of landmarks, obstacles, etc.) describing the environment in which the robot operates [4].

There are many situations where a map is needed. For example, a map may be in need to support other tasks like informed path planning. However, most importantly, the map allows limiting the errors committed in estimating the state of the robot [4]. In the absence of a map, dead-reckoning would quickly drift over time; on the other hand, using a map, e.g., a set of distinguishable landmarks, the robot can reset its localization error by revisiting known areas, also referred to as loop closure. Therefore, SLAM finds applications in many scenarios in which a prior map is not available and needs to be built [4].

SLAM has been formulated and solved as a theoretical problem in a number of different forms [7]. It has also been implemented in a number of different domains including indoor robots, outdoor robots, and underwater and airborne systems. At a theoretical and conceptual level, SLAM can now be considered to be a solved problem pertaining to the estimation of the trajectory of a moving robot and building a map of its environment simultaneously [17]. However, in practice, substantial issues remain in realizing more general SLAM solutions and notably in building and using perceptually rich maps as part of a SLAM algorithm [7]. Even though the formulation of the SLAM problem has been well established and the robotics research community has seen tremendous progress over the past few decades, there are still a many open problems left unsolved including fail-safe SLAM algorithms, efficient map representations, and resource-aware SLAM systems [4]. Furthermore, a general SLAM solution that can run in real time and adapt to the available computing platforms has not yet been proposed. Also, many of the existing SLAM algorithms fail to identify previously visited locations and correct the corresponding odometry estimations, thus performing loop closure [6].

We introduce a robust and flexible multi-sensor data fusion architecture that leverages state-of-the-art Lidar algorithms. Our system provides custom configurations to allow further research, for example, in innovative image registration algorithms, frame matching algorithms, and backend nonlinear least-squares pose-graph solvers. We have also supplemented the multi-sensor data fusion model with the necessary hardware, control, and planning module to provide a cost-friendly autonomous driving platform. This platform, with the physical form of a differential drive robot, is capable of driving around in an unknown environment, creating a map of its surroundings and performing autonomous navigation to any targeted location in the self-created map.

The rest of this paper is organized as follows. In Section II, we provide a high-level overview of multi-sensor data fusion. Then, we present our multi-sensor data fusion pipeline in Section III. The system implementation, along with hardware and software dependencies, is then described in Section IV. Our experimental results are shown in Section V. Finally, we present our conclusions and explore possible future work.

*This work was not supported by any organization. The code and models presented in this work are publicly available at https://github.com/ZhekaiJin/the-Cooper-Mapper

[1]Zhekai Jin is a student of Electrical and Computer Engineering, The Cooper Union, 41 Cooper Sq, New York, NY. thisisscott24@gmail.com

[2]Yifei is a student of Mechanical Engineering, The Cooper Union, 41 Cooper Sq, New York, NY. simonshao96@gmail.com

[3]Minjoon So is a student of Mechanical Engineering, The Cooper Union, 41 Cooper Sq, New York, NY. somj94@gmail.com

[4]Carl Sable is an Associate Professor of Computer Engineering, The Cooper Union, 41 Cooper Sq, New York, NY. carlsable.cooper@gmail.com

[5]Neveen Shlayan is an Assistant Professor of Electrical Engineering, The Cooper Union, 41 Cooper Sq, New York, NY. neveenshlayan@gmail.com

[6]Dirk Luchtenburg is an assistant Professor of Mechanical Engineering, The Cooper Union, 41 Cooper Sq, New York, NY. dluchten@cooper.edu

II. BACKGROUND

An autonomous mobile robot operates by processing information from its surroundings and then making intelligent and accurate driving decisions. This means that the perception system, the very first module to acquire peripheral information on which other parts of the platform depend, needs to be as robust and accurate as possible to safeguard the performance of the whole system. A system operating with a single sensor often fails to capture the rich physical attributes of the environment. The camera, a typical visual perception sensor, is likely to fail in environments where the lighting intensity is dramatically changed or the lighting intensity is particularly low. On the other hand, a radar sensor has a longer sensing distance and lower computational demands but is less accurate than the light imaging detection and ranging (Lidar) in terms of angular accuracy. Due to the inherent vulnerability of the single-sensor system, multi-sensor data fusion has become an overarching paradigm for avoiding single-point failure and enhancing the system with reduction in ambiguity and uncertainty, increase in accuracy, robustness against interference, etc [15]. For instance, Tesla's Autopilot leverages a hardware suit of eight cameras, a forward-looking radar, and twelve ultrasonic sensors to ensure 360 degrees of visibility for its perception system.

In the last decade, significant progress has been made in the field of multisensor data fusion to solve problems related to combining multimodel data efficiently and support intelligent robots in decision making [2], [3], [12]. The diversity offered by multiple sensors can positively contribute to the perception task of the intelligent robot. Overcoming heterogeneity of different sensors through robust fusion algorithms lead to effective utilization of the redundancy across the sensors [5]. However, data coming from different sources are typically in different formats and also propagate different sensing uncertainties. Multisensor data fusion research is typically focused on the effective alignment of different sensor streams which could be either partially, geometrically, or temporally aligned [13]. The introduction of the multi-sensor data fusion model, though effective in theory, does lead to some practical challenges including how to handle noise in the operation, data imputation, the determination of where in the processing pipeline to perform the fusion algorithm, and how and when to keep or drop the previously acquired information. Moreover, due to inevitable sensor manufacturing variations, extreme external calibration efforts across the sensors is often needed to ensure the performance of the fusion architecture.

III. PROJECT DESCRIPTION

To address the problems mentioned in Section II and to maximize the cost efficiency along with the mapping accuracy, we introduced a loosely coupled time-stamp-based multi-sensor data fusion architecture which leverages camera and Lidar data as default. On top of the default fusion setup, the system provides custom configuration freedom for researchers to add additional sensor modality and experiment with various fusion algorithms. The default fusion architecture, which leverages the state-of-the-art Lidar SLAM pipeline and multiple visual place recognition algorithms, ensures the basic functionality for accurate mapping with global closure and reduces the unnecessary exterior calibration effort [19].

In Figure 1, the features of a few typical implementations of the SLAM algorithm are summarized. As can be seen, most existing solutions do not incorprorate 2D Lidar, a camera module, and global loop closure. For example, V-LOAM typically does not implement loop closure even though it uses both a 3D Lidar and a camera; however, some form of loop closure is introduced in some relevant work [16]. Another issue is that all of them except Cartographer require a 3D Lidar component, which is costly. The proposed solution herein can be used with either 2D or 3D Lidar sensors in tandem with a camera, and at the same time offers both global loop closure and online operation.

	2D Lidar	3D Lidar	Camera	Global Loop Closure	Online Operation
BLAM		●			●
Laser SLAM		●		●	●
Cartographer	●	●		●	●
LOAM		●			●
V-LOAM		●	●		●
Our Solution	●	●	●	●	●

Fig. 1: Typical Implementations of SLAM

While state-of-the-art visual and Lidar SLAM algorithms are equivalent in terms of accuracy, visual pipelines are more robust for dynamic scenes and less expensive computationally. Lidar SLAM systems, on the other hand, are more consistent and less sensitive to changes in illumination and appearance due to their heavy dependence on the geometric structure of the surrounding. Even though most of modern Lidar SLAM algorithms have shown impressive results [8], they failed to address the drift problem over time with the assumption that the world is an "infinite corridor" [4]. Therefore, we propose a fusion mechanism that supplements the Lidar SLAM algorithm with visual stereo image data for place recognition and drift correction. In order to implement Lidar SLAM, we tested our vehicle with various open-source libraries including Hector SLAM, Fast SLAM, and Gmapping. In this section, we discuss the various Lidar-SLAM libraries tested. Then, we introduce our perception system by explaining the selection of the sensors, the base SLAM module, and the data fusion model proposed for the heterogeneous sensors involved.

A. *Lidar-SLAM Libraries*

Hector SLAM was developed for a system capable of autonomous exploration in Urban Research and Rescue environments [11]. It serves as a general open-source algorithm, which only needs minor modifications to operate on a given platform. A remarkable feature of this algorithm is that it does not necessarily need the odometry data to support its operation. Another feature of Hector SLAM is its elevation and cost mapping. The Hector-elevation-mapping module allows us to fuse the point cloud measurements produced by a stereo camera into an elevation map, resulting in a 2D grid with another variable height stored in a corresponding variance for each cell. Odometry, however, is notoriously known to be unreliable in an environment where there are many altitude changes (such as an uneven floor). Therefore, we decided to test Hector SLAM on our data. Even though it was able to create a map, the drift was large. This meant that the odometry data had to be fed to the system so that the algorithm can make more informed estimations of its pose and create a more accurate map.

Instead of entirely relying on fast Lidar data feature selection and scan-matching, Fast SLAM uses a particle filter method which uses numerous small particles to perceive a submap and then creates a complete map by stitching those submaps together. The particles are generated randomly, and submaps are then compared with each other to test for agreement about the perceived environment given their poses. In other words, a particle's correctness is evaluated by consensus and inference based on the other submaps. Faulty particles are immediately discarded. Eventually, only the particles that can make sense of each other's submaps are kept and used to stitch together the whole map. However, the particle-filter-based method is relatively memory intensive since each particle needs to be kept in a joint state matrix and updated every frame. Also the comparison process consumes a tremendous computing power. The problem could be largely simplified by providing a prior map. The particles' submaps could then be compared with the prior map and discarded if the difference is too large. Thus, the particle number would quickly decrease and converge to allow the construction of a complete map. This is especially useful in the re-localization problem for self-driving automobiles where a prior high definition (HD) map is available.

Gmapping has been implemented as described in Grisetti et al. [9] and is then improved using the Rao-Blackwellized particle filters (RBPF) method, which shares a similar idea with the particle filter method introduced above. The key idea behind RBPF is to estimate a posterior of potential trajectories of the robot given its observation and its odometry measurements. The posterior is then used to compute a posterior over maps and trajectories and thus gives a relative pose estimation. To do so, RBPF uses a particle filter in which an individual map is associated with every sample. The robot's trajectory changes over the robot's motion, therefore the proposal distribution is chosen to be the same as the probabilistic odometry motion model. One of the most common particle filtering algorithms is the sampling importance resampling (SIR) filter. An SIR filter incrementally processes the observations and the odometry readings as they become available. This is done by updating a set of samples representing the posterior about the map and the trajectory of the vehicle. The algorithm for RBPF is then applied by computing an improved proposal distribution on every particle so that information obtained from the sensors can be used while generating the particles. This algorithm has two main advantages: First, the algorithm draws the particles more effectively; computing accurate proposal distribution handles not only the movement of the robot but also the most recent observation, which causes the uncertainty in the prediction of the robot's pose to decrease. Second, the highly accurate proposal distribution allows the system to utilize the number of effective particles as a robust indicator to decide whether or not a resampling has to be carried out. This effect further reduces the particle depletion problem, which refers to the scenario where no particle is valid at all. Therefore, we decided to use the Gmapping algorithm to implement an improved version of the visual Lidar [9].

B. Sensors

By using the multisensor data fusion pipeline, our algorithm can perform SLAM using a a stereo camera with a 2D or 3D Lidar as shown in Figure 2. With this specific combination, we can avoid disadvantages of each sensor and make the system more robust. For example, the camera will not perform as well as the Lidar in dark environments. However, each kind of Lidar has its own problems: 3D Lidars are costly and 2D Lidars alone do not offer enough resolution. To solve this issue, we supplement the 2D Lidar with a stereo camera so that we can extract more information from the images. This way, researchers can perform accurate SLAM algorithms with a cheaper 2D Lidar

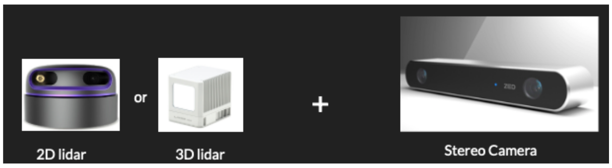

Fig. 2: Sensors Used

RPLiDAR A2: Whereas Hokuyo UST-20LX scanners are now the standard 2D Laser scanners for SLAM research, we found the RPLIDAR A2 scanners to be a cheaper option. Though 3D Laser scanners have the advantages of high resolution and a 360 degree range for 3D SLAM algorithm research, their high cost made the actuated 2D Lidar more suitable for our purpose. With a reasonable cost, the RPLIDAR A2 can perform 360 degree scans within a range of 12 meters or 18 meters and generate 8000 points per second with a 15 Hz sampling rate. Also DJI has released Livox Mid-40, a 3D Lidar with a reasonable price, which future researchers with a generous budget could consider for dense mapping purposes.

ZED stereo camera: ZED is the best-suited camera for our platform due to its detailed API documentation and its smooth integration with the Robotics Operating System (ROS) which most of robotics research uses. With its high resolution and frame rate, ZED can serve multiple applications such as depth perception, positional tracking and 3D mapping.

C. Base SLAM Module

We present a SLAM module which extends the state-of-the-art Lidar odometry estimator, LOAM [20], with back-end pose-graph optimization to correct drift and a place recognition system to allow global loop closure [10]. LOAM, with its high accuracy, robustness and real-time operation, takes in raw 3D point clouds, calculates the rigid transformation due to the corresponding sensor motion and outputs the global pose estimation, a local representation of the map, and the registered point clouds. The original work has been refactored, optimized, and made modular in this work to support custom configuration and allow smooth adaption to other SLAM backend solutions.

Due to the inherent drifting error in incremental odometry estimators like LOAM, an online pose estimation back-end is needed in the system to build the pose graph based on the LOAM odometry estimation and correct LOAM odometry estimation from drifting error by performing re-localization based on the visual data. Re-localization takes place if the system identifies previously visited places. To identify previously visited places in the existing internal map, the system has to periodically query the place recognition module, which relies on the visual stereo data; this is explained in Section III-D.

D. *Data Fusion Mechanism*

We propose a modular multi-sensor data fusion pipeline, as summarized in Figure 3, where Lidar is set as the default sensor for odometry estimation and visual stereo data is leveraged to perform place recognition. The Lidar-based SLAM backend keeps a set of keyframes to represent the sensor trajectory, each having an associated time stamp. With the stereo camera running constantly, the system registers stereo image data to the latest keyframe and performs cloud matching with all previously registered keyframes to find potentially matched frames. A pose graph optimization backend is running constantly to manage the environment mapping and correct odometry estimation by querying the visual place recognition system. We provide multiple state-of-the-art visual frame matching algorithms such as visual bag of words and SegMatch. Additionally, the system is able to incorporate other real-time matching algorithms and fuse with the result of existing matching algorithms [1].

When the system detects a matched frame, it calculates the transformation between the clouds of the associated keyframes using the iterative closest point (ICP) algorithm and adds a new edge to the pose graph representation of the existing map. Then, the estimated pose is fed back to the incremental odometry estimator to correct its internal motion estimation and perform the re-localization functionality.

IV. SYSTEM IMPLEMENTATION

To make our vehicle reasonably affordable and easier to assemble, we designed our vehicle to be an educational and cost-friendly research platform with minimal software setup on which versatile applications could be run. The hardware and computing platform are introduced in this section with key design features including affordability and versatility. It is worth noting that the components used for the vehicle are resources that are inexpensive, with a total cost of approximately $1200.

A. *Hardware*

We designed a custom differential drive chassis on which any electronics and hardware can be installed. We relied on easily accessible computer aided design (CAD) software and prototyping tools including SolidWorks and AutoCAD. Two Pololu 12V gear motors are used to drive the rear wheels with a 2000-count-per-rev encoder mounted on each motor. Having two independently-driven rear wheels gives the platform two degrees of freedom for intuitive manipulation and control. In addition, the built-in encoders enable wheel speed control and could provide inaccurate odometry information to the system for reference. Also, a custom PCB board is used to connect electronic components and divide an electrical power feed from the batteries into subsidiary systems. Two 12V Lithium-ion batteries are used to supply power to the computing hardware and motors separately, which prevents the motor's transient voltage from interfering with the computing hardware.

B. *Computing Platform*

The Nvidia Jetson TX2 is a fast, power-efficient embedded computing device which is used as the on-board computing processor. Jetson supports Ubuntu naively for ROS integration and provides the necessary processing power for online 3D mapping algorithms. The Arduino Uno is used along with the Jetson computer as an expansion to the GPIO and interrupt pins of the Jetson. Acting as a middleman, it exchanges messages between the hardware and the Jetson board. A photograph of our device with components labeled can be seen in Figure 4.

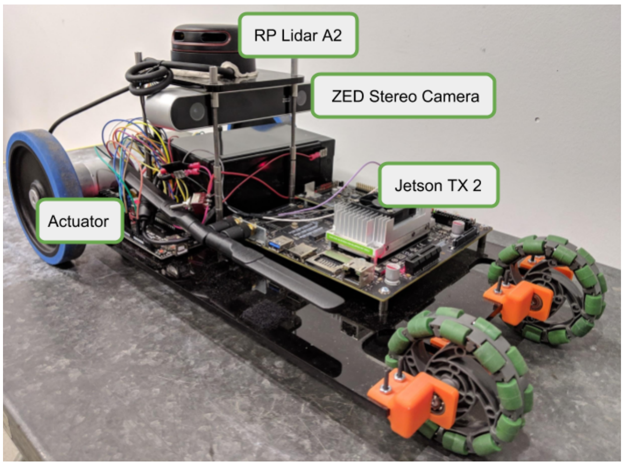

Fig. 4: Platform Overview

V. EXPERIMENTAL EVALUATION

To evaluate the accuracy of the odometry estimation of our proposed multisensor data fusion architecture, we fully tested our algorithm against the publicly available KITTI odometry benchmark dataset [8]. The result was evaluated by the metrics employed by KITTI and compared with the LOAM module's result. Our architecture, with the default setup, has shown equivalent results with LOAM for KITTI sequence 00 and

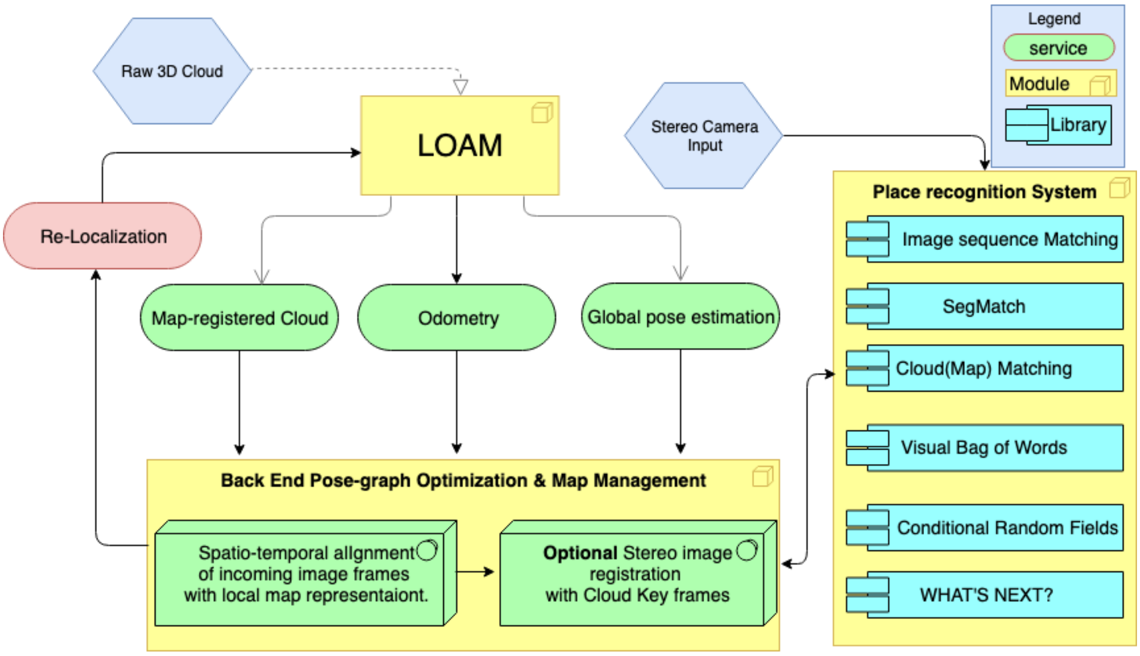

Fig. 3: Sensor Fusion Architecture

better performance than LOAM for KITTI sequence 05 by generating a trajectory map closer to the ground truth value in places where loop closure takes place. Figure 5 shows the results of our algorithm running on KITTI odometry dataset sequence 07. As depicted, our estimated trajectory is closer to the ground truth value than the popular SLAM module, ORB SLAM, a versatile and accurate monocular SLAM system where the loop closure happens [14].

We also ran our algorithm in a real-world indoor environment, the 6th floor of our academic building, by adapting the state-of-the-art 2D mapping algorithm, Cartographer [10], to our pipeline. Figure 6 shows the generated 2D occupancy grid map (bottom), and for comparison, the ground truth floor plan (top). As can be seen, the results are reasonable.

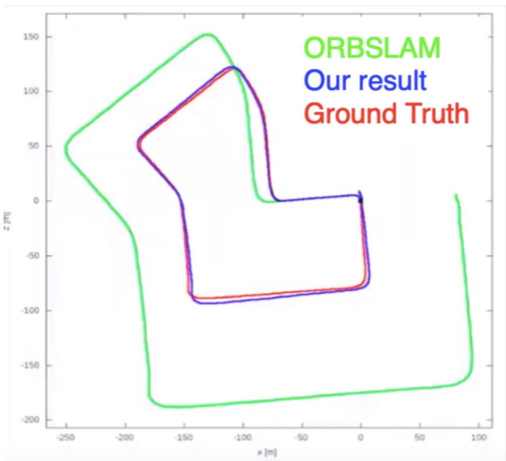

Fig. 5: Experimental results on KITTI Sequence 07

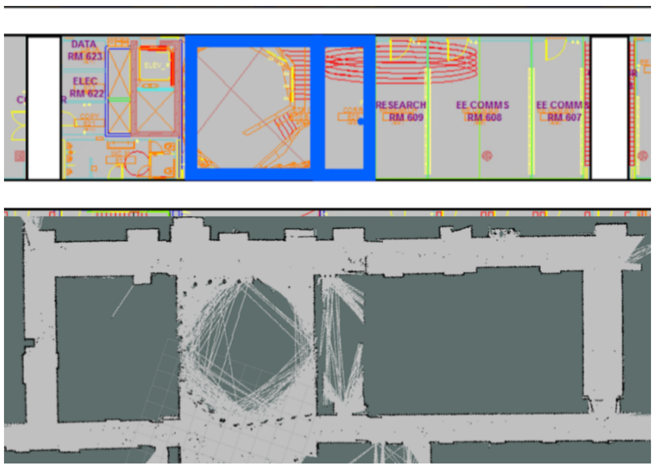

Fig. 6: Top: Floor Plan, Bottom: Mapping Result

VI. CONCLUSION

A new, integrated, and modular sensor fusion architecture has been developed and fully tested against a publicly available data set. Experiments have validated the hypothesis that by leveraging the redundancy across heterogeneous sensors, multi-sensor data fusion improves accuracy and robustness for applications such as mapping and motion estimation. In addition, the modular pipeline provides robotics researchers the freedom to adapt and experiment with related algorithms.

There are many directions in which this work can be expanded. For the multisensor data fusion model, pre-built models for sensors of different modalities can be developed. For example, a model can be built for the inertial measurement unit (IMU), which is often used in modern SLAM algorithms to improve the accuracy and robustness of mapping [18].

While we primarily focused on the perception system of the autonomous driving platform, the control and planning modules of the platform can be further developed to provide more research possibilities for the future users of our platform. Part of this work was originally presented as an IEEE undergraduate student paper at the 2019 IEEE Region 1 Annual Student Conference.

REFERENCES

[1] Mario Amoruso, Stefano Caiola, Giuseppe Doronzo, and Marino Difino. Altran and ni demonstrate adas hil with sensor fusion. Mar 2019.

[2] Minoru Asada, Yasuhito Fukui, and Saburo Tsuji. Representing global world of a mobile robot with relational local maps. *IEEE transactions on systems, man, and cybernetics*, 20(6):1456–1461, 1990.

[3] Dolores Blanco, Beatriz L Boada, Luis Moreno, and Miguel Angel Salichs. Local mapping from online laser voronoi extraction. In *Intelligent Robots and Systems, 2000.(IROS 2000). Proceedings. 2000 IEEE/RSJ International Conference on*, volume 1, pages 103–108. IEEE, 2000.

[4] Cesar Cadena, Luca Carlone, Henry Carrillo, Yasir Latif, Davide Scaramuzza, José Neira, Ian Reid, and John J Leonard. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on Robotics*, 32(6):1309–1332, 2016.

[5] Varuna De Silva, Jamie Roche, and Ahmet Kondoz. Robust fusion of lidar and wide-angle camera data for autonomous mobile robots. *Sensors*, 18(8):2730, 2018.

[6] Renaud Dubé, Daniel Dugas, Elena Stumm, Juan Nieto, Roland Siegwart, and Cesar Cadena. Segmatch: Segment based loop-closure for 3d point clouds. *arXiv preprint arXiv:1609.07720*, 2016.

[7] Hugh Durrant-Whyte and Tim Bailey. Simultaneous localization and mapping: part i. *IEEE robotics & automation magazine*, 13(2):99–110, 2006.

[8] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[9] Giorgio Grisetti, Cyrill Stachniss, and Wolfram Burgard. Improved techniques for grid mapping with rao-blackwellized particle filters. *IEEE transactions on Robotics*, 23(1):34–46, 2007.

[10] Wolfgang Hess, Damon Kohler, Holger Rapp, and Daniel Andor. Real-time loop closure in 2d lidar slam. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1271–1278. IEEE, 2016.

[11] Stefan Kohlbrecher, Johannes Meyer, Thorsten Graber, Karen Petersen, Uwe Klingauf, and Oskar von Stryk. Hector open source modules for autonomous mapping and navigation with rescue robots. In *Robot Soccer World Cup*, pages 624–631. Springer, 2013.

[12] John J Leonard, Hugh F Durrant-Whyte, and Ingemar J Cox. Dynamic map building for an autonomous mobile robot. *The International Journal of Robotics Research*, 11(4):286–298, 1992.

[13] Ren C Luo, Chih Chia Chang, and Chun Chi Lai. Multisensor fusion and integration: Theories, applications, and its perspectives. *IEEE Sensors Journal*, 11(12):3122–3138, 2011.

[14] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE transactions on robotics*, 31(5):1147–1163, 2015.

[15] KS Nagla, Moin Uddin, and Dilbag Singh. Multisensor data fusion and integration for mobile robots: A review. *IAES International Journal of Robotics and Automation*, 3(2):131, 2014.

[16] Yoshua Nava. *Visual-LiDAR SLAM with loop closure*. PhD thesis, Master's thesis, KTH Royal Institute of Technology, 2018.

[17] Paul Newman and Kin Ho. Slam-loop closing with visually salient features. In *proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pages 635–642. IEEE, 2005.

[18] Meixiang Quan, Songhao Piao, Minglang Tan, and Shi-Sheng Huang. Tightly-coupled monocular visual-odometric slam using wheels and a mems gyroscope. *arXiv preprint arXiv:1804.04854*, 2018.

[19] Jochen Sprickerhof, Andreas Nüchter, Kai Lingemann, and Joachim Hertzberg. An explicit loop closing technique for 6d slam. In *ECMR*, pages 229–234, 2009.

[20] Ji Zhang and Sanjiv Singh. Loam: Lidar odometry and mapping in real-time. In *Robotics: Science and Systems*, volume 2, page 9, 2014.