2022 13th International Congress on Advanced Applied Informatics Winter (IIAI-AAI-Winter) | 979-8-3503-0992-8/22/$31.00 ©2022 IEEE | DOI: 10.1109/IIAI-AAI-WINTER58034.2022.00046

Obstacles Detection and Motion Estimation by Using Multiple Lidar Sensors Data

Shiying Liu
Graduate school of Data Science
Musashino University
Tokyo, Japan
g2250003@stu.musashino-u.ac.jp

Takafumi Nakanishi
Graduate school of Data Science
Musashino University
Tokyo, Japan
takafumi.nakanishi@ds.musashino-u.ac.jp

***Abstract*— This paper presents a Lidar-based multiple-sensor system that is used to estimate the motion of the observer via surrounding information. As a basic solution for surrounding sensing, SLAM is a system that combines simultaneous localization and mapping. It supports several types of sensors, including Lidar sensor, camera, odometer, and IMU (Inertial Measurement Unit). We focus on applying the SLAM system to a multiple linear laser Lidar sensor group by using only their sensing data stream of length and angle. For this sensor group, we propose a new method that combines an image phase correlation algorithm and pose-graph optimized algorithm to implement obstacles detection and motion estimation. We also design a filter applying the combination method to the Lidar sensor.**

Keywords—Lidar sensor, SLAM, dynamic pose estimation, pose-graph correlation, image registration, phase correlation

I. INTRODUCTION

Obstacle detection is one of the basic topics in data sensing, which is benefit for accuracy control and making a response or reaction by complex surrounding information. With the development of the sensing system in robotics, sensor data extraction plays an important role in the interaction between machines and the real-world. The utilities of surrounding information are dedicated to autonomous navigation systems such as mapping and path planning.

For an autonomous navigation system, to make a reliable mapping from the surroundings, the sensing data will be established based on the real-time location where the observer standing by. Thus, one of the necessary functions for obstacle detection is to obtain its own dynamic status and reliable measurement at the same time. IMU or odometer data is always used to collect real-time pose data and estimate the next pose of the observers. However, those pose data do not include any surrounding information, so that the errors will propagate after several loops. Therefore, we should consider the correlation between the differences in surroundings to reduce propagation errors.

The field of autonomous navigation consists of Lidar-based, vision-based, and a combination method [1]. The Lidar-based solution is beneficial for accuracy, dynamic movement, and memory efficiency but is limited in a priori knowledge and surroundings distinction. The vision-based solution is beneficial for object detection and feature recognition but is limited in processing efficiency and losing features [1]. The recent solution is to combine those two methods by using vision approaches such as camera intrinsic calibration and camera-Lidar-pair extrinsic calibration [2]. The combination method allows extracting landmarks of features to extend information into three-dimensional space. However, the combination method still has some limitations such as in some areas with featureless surroundings or repeating structures, it is difficult to find a true match between the recognitions from Lidar-based and vision-based [3].

In this paper, we propose a multiple Lidar-based obstacle detection and motion estimation system. Our system mainly consists of the pose estimation part and obstacle mapping part. The pose estimation is performed by using present and past period of data sets from different Lidar sensors. At the same time, the pose information is used for real-time mapping of obstacles and updating the historic surroundings. We discuss the rate of the pose measurement in a different method and the error after times rounds.

The main contribution of this paper is as follows.

- We design and propose a prototype system that uses multiple Lidar sensors to improve a priori knowledge instead of a vision-based solution.
- We develop the system by using phase correlation algorithm to solve the Lidar-based pose estimation.
- We conduct some experiments to improve the vision algorithm.

This paper is organized as follows. In section 2, we will discuss the related works. In section 3, we will introduce the system designed for multiple sensor associations and the preparation for simulation. In section 4, we will represent the simulation results and discussion. Finally, we will make a conclusion.

II. RELATED WORKS

At the beginning of the research, we should consider the four fields of solution as fellows, dynamic status, SLAM system, phase correlation, and loop-closing solution.

979-8-3503-0992-8/22/$31.00 ©2022 IEEE
DOI 10.1109/IIAI-AAI-Winter58034.2022.00046

A. Dynamic Status

The dynamic status is the way to describe the movement of the automobile, the related research is beneficial for pose-estimation on a system based on the sensor group.

For most automobiles, knowing their own dynamic status is one of the prerequisites in processing any interaction and reaction with surroundings. There are various dimensions to describe current status, basically as velocity and acceleration. By using those statuses, we can establish reliable estimation models to predict the following status. We also can reduce the error from the measurement of dynamic status due to the parameters of the uncertain surrounding. For example, to train a 4-DOF (degree of freedom) vehicle estimation model [4], acceleration, wheel speed, velocity, and yaw rate sensor configurations are needed. When the yaw rate data is covered, it requires an expert sensor to distinguish them from the most capable IMU sensors.

In our research, we focus on only multiple Lidar sensors. Thus, we prefer to use linear velocity and angular velocity to describe the dynamic status. The status of yaw rate and acceleration changes relies on using a priori knowledge of surroundings to be resolved.

B. SLAM algorism

For storing sensing data, simultaneous localization and mapping (SLAM) systems are designed to collect and estimate their own location and compose a map of the surrounding information at the same time [5]. It is one of the main parts of solutions in autonomous navigation and surrounding detection.

SLAM algorithm has been developed a lot by researchers, and there are mainly three groups of methods. The first group is using probability theory to fill the pixel boxes like RBPF (region-based pressure force) [6]. The second group is using pose transformation to match the real surroundings to estimate the pose; this method is not using any odometers like Hector-SLAM [7]. The last group is the most common one updated by Google groups [8]. It has been updated to vision 3 [8], which provides developed surroundings for several types of sensors embracing Lidar, RGB-D cameras, or IMU sensors. It also offers multi-map SLAM and survives for a longer period in case of featureless surroundings.

Our research is mainly based on the solving steps of a SLAM algorithm; thus, the components of the system consist of data association, estimation, and loop-closing, except the requirement from visual components.

C. Image Phase correlation

Phase correlation is an image algorithm used to find the translation between two images. It is a light method to take the place of traditional SLAM algorithm due to the complexation of SLAM algorithm in calculation and facility.

Image phase correlation algorithm [9] is an efficient process in the image registration domain based on the Fourier shift theory [10]. The process has been widely used in the field of computer vision [10]. According to the reference [11], the limitation of the phase correlation is the registration speed and the occurrence of unpredictable errors.

In our research, we consider calculating the transformations between surroundings measurements to estimate the present status.

D. Loop closing

Loop-closing is one type of control system in which control processes are applied from the outputs they produce. In the SLAM system, we use this control system to reduce the error by using a series of pose estimations. Besides, the Loop-closing solution is the methodology to improve system reliability and accuracy.

Overlap Net [12] is designed to solve the loop-closing problem by using a deep neural network to find the controlling candidate from range images and relative yaw between two scans. ORB-SLAM3 [8] uses types of sensor data association to build a map of surroundings and compute in a real-time pose of an agent on the map. The data association has three occasions: the short-term data association matches elements during the last few seconds for continuous drift estimation; the mid-term data association matches elements close to the observer for loop detection; the long-term data association matches observations with elements in previously visited to reset the drift and correct observation. Based on DBoW2 [13] (Bags of Binary Words for FAST Recognition in Image Sequence), reference [14] applies dictionary tree clustering to extract feature points instead of using k-means clustering to improve a real-time SLAM loop-closing detection algorithm. LAGO [15] applies formulation optimization to solve the matrix created by five related pose-graph nodes. Reference [16] proposed a robust pose-graph optimization formulation to cope with heavy-tailed noise when they estimate pose from relative pose measurement.

In our research, we propose a new style of pose-graph optimization by four neighbor nodes of the pose-graph in similar surroundings.

III. Method for Simulation

We design a simulation to test and optimize our system. The overview of the system is shown in Fig.1. The components of the system include an initialization unit, a Lidar simulation unit, a pose estimation unit, and an error reducing unit. Besides, we need a database to store two types of execution results: sensor dataset as the result of simulation and trace dataset as the result of the pose estimation. The procedure of the system is to create a ground of world map and trace by initialization unit first. Then the Lidar simulation unit uses the world map to create sensing data set via setting different traces. The pose estimation unit regards those data flow as real-time sensing data and executes various settings of functions to estimate real-time poses. Error reducing unit is used to reduce errors in pose estimation.

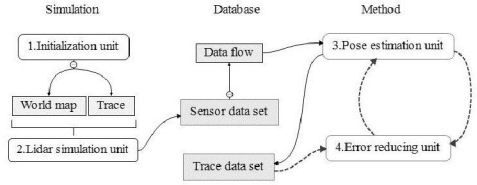

Fig. 1. Simulation system overview.

A. *Initialization unit*

Before the simulation launch, we should establish a world map as the ground in the system. The world map represents a two-dimensional matrix which is filled with grayscale values from 0 to 255. The value 0 stands for nothing, and the value over 0 stands for existing of obstacles. In this system, we presume a smart car has two Lidar sensors running on the world map. The distance between two Lidar sensors is settled by an unchangeable offset. The trace of smart car T will be filled by the locations in the world map sorted by the time flow t.

$$p_t = (x_t, y_t) \tag{1}$$

$$T = \{p_i \mid i = [0:t]\} \tag{2}$$

B. *Lidar simulation unit*

For the data in one location p_t, the Lidar sensor simulation and fragment surroundings calculation will be processed.

Lidar sensor simulation is the process to calculate one round of Laser distance (Fig.2) from the world map in location p_t. The result of process S will be filled by one round flow of angle α and length 1.

$$S = \{(\alpha, 1) \mid \alpha = [0:2\pi]\} \tag{3}$$

For each location data in trace T, we will execute two simulations to collect the sensor data set $S_{1|t}$, $S_{2|t}$ sorted by time step from two sensors.

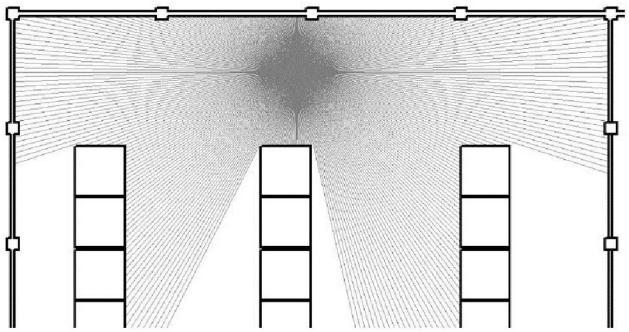

Fig. 2. One round of Lidar sensor simulation.

C. *Pose estimation unit*

The method for pose estimation consists of a filter and a phase correlation process.

For the input of a set of multiple sensor data $S_{1|t}$ and $S_{2|t}$, we create a filter Fl(S) to extend the data set. The filter contains coordinate decoding, and relationships of neighbors.

$$Fl(S_t) = \{(l_i \cos\alpha, l_i \sin\alpha, \frac{l_{i-1}{}^2 - l_{i+1}{}^2}{2l_i}, \mathrm{mean}(l_{i-1}, l_i, l_{i+1})\} \mid t \tag{4}$$

To estimate the movement between two locations, we prefer to execute the movement vector between them by using four sets of sensor data: present position $S_{1|t}$, present offset $S_{2|t}$, past location $S_{1|t-1}$ and its offset $S_{2|t-1}$. The movement vector between the present and the offset has been given so that it can be used it for correlation.

The main workflow is shown in Fig.3. We transfer the input pair of one-dimensional sensor data to two-dimensional image data firstly because of the requirement of phase correlation algorithm [14] [15]. Secondly we use Fourier transfer to get the phase. After using a normalize operator, we use Inverse Fourier transfer to optimize the corresponding from the coordinates to the amount, to fetch the result of pose movement estimation.

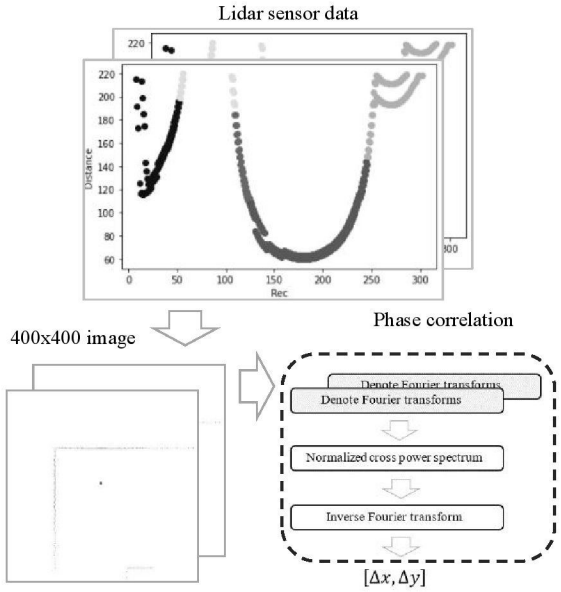

Fig. 3. Pose estimation workflow.

We transfer the sensor data set $S_{n|t}$ to image. While the value of pixel $P = \{pixel_{i,j}\}$ in the image follows the rule (5). The return of the rule is a type of unit8, which is stronger when the pixel is near the center of the image or the pixel is suspected as an angle. The rest of the pixels will be filled by zeros, so the size of image can default use 400×400 pixels or even bigger than the traditional phase correlation process.

$$pixel_{l_i\cos\alpha, l_i\sin\alpha} = \mathrm{rule}(\frac{l_{i-1}{}^2 - l_{i+1}{}^2}{2l_i}, \mathrm{mean}(l_{i-1}, l_i, l_{i+1})) \tag{5}$$

Consider two images img1 and img2 which have been transferred from $S_{n|t}$, we use Fourier shift (6). While img2 is shifted by an amount $[\Delta x, \Delta y]$ relative to img1.

$$fs_2(x, y) = fs_1(x - \Delta x, y - \Delta y) \tag{6}$$

The boundary conditions as follows are in default settings:

$$fs(400 + x, 400 + y) = fs(x, y) \tag{7}$$

$$F_2(\omega_x, \omega_y) = e^{-j(\omega_x \Delta x + \omega_y \Delta y)} F_1(\omega_x, \omega_y) \tag{8}$$

The Fourier transforms of fs_1 and fs_2 as F_1 and F_2 while the linear phase term as $j(\omega_x \Delta x + \omega_y \Delta y)$.

This phase term can be calculated by normalized cross power spectrum (CPS). CPS is a function that includes a Schur product operator and complex conjugate operator.

$$e^{-j(\omega_x \Delta x + \omega_y \Delta y)} = \mathrm{CPS}(\omega_x, \omega_y) \quad (9)$$

Finally, we use the inverse Fourier transform of CPS to fetch the amount $[\Delta x, \Delta y]$ between given images. We obtain the vector between two sensor data sets.

As for the estimation of the trace data set, we assume the start location is (0, 0) and add the movement vector step by step to compute T_{sim} sort by time flow.

D. Error reducing unit

As for pose-graph correlation, we should design a loop for vectors from the pose estimation unit. Fig.4 shows four pose nodes P and four vectors ξ between nodes. We presume the past measurement node $P1$ as a ground truth. $P2$ is a past offset node, $P3$ and $P4$ are the present and present offset nodes. We presume the estimations are in a same error vector $[rx, ry]$, the estimations vector $\{\xi_{14}, \xi_{43}, \xi_{32}, \xi_{21}\}$ contains two ground truth factors, one loop-closing factor, and one pair of destination factor. We can optimize the problem to calculate the best solution to reduce the error vector $[rx, ry]$. The fixed estimation vector is as follows.

$$\xi_{12\to 34} = [\frac{\xi_{14} \mid x - \xi_{32} \mid x}{2} - rx, \frac{\xi_{14} \mid y - \xi_{32} \mid y}{2} - ry] \quad (10)$$

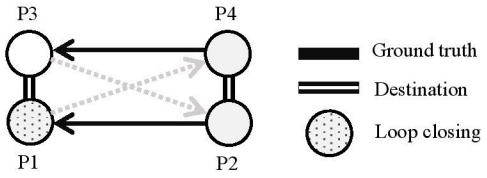

Fig. 4. Pose-graph correlation.

IV. Experiment

The overview of the experiment is shown in Fig.5. We have four steps in this experiment: build a surroundings, Lidar sensor simulation, sensor data input, and get results. We conduct a series of simulations to test the function we designed for the system: filter, Image phase correlation, and Pose-graph correlation.

To test the influence of different surroundings. We build a surrounding in step1 first. After step1, we can decide which location to start and which trace seed to process. As for surroundings, we consider testing the impact of the change from the width and the number of details in different areas. We create trace seeds that contain strict, right-turn, and zigzag paths. After we create another type of trace via settings of starting location and trace seed, we get into step2 to process the sensor dataset.

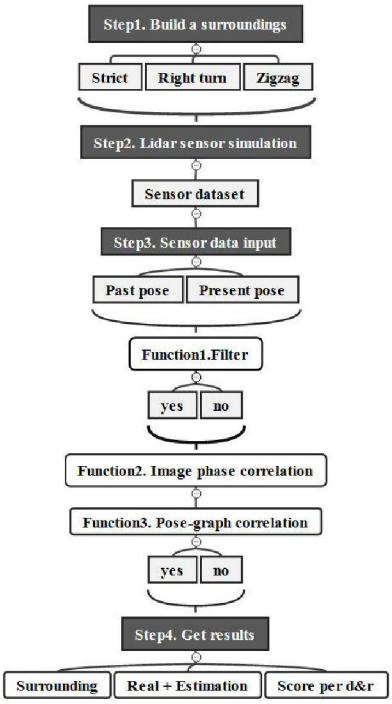

Fig. 5. Simulation overview.

In step3 we input the sensor dataset and process the three functions. In each function, we can choose a different method to process. Finally, in step4, we compare the estimation results with the ground truth and draw conclusions.

Thus we can create several groups for condition control experiments. The conditions are shown in Fig.6.

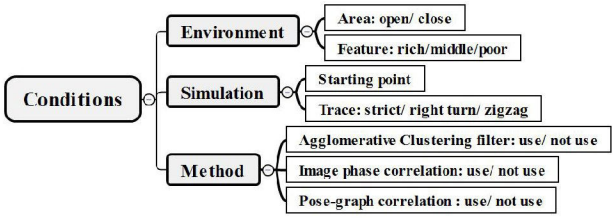

Fig. 6. Exprtiment conditions.

A. Surroundings of simulation

The simulation ground up to 2000 × 2000 pixels. The Lidar simulation calculates 360 pairs of length and angle data per round. The image size input of phase correlation is up to 1500 × 1500 pixels.

B. Pose estimation

In Fig.7, the black solid represents obstacles, and arrows show the destination of the trace. An open area means a huge room or outdoor. The closed area means indoor, especially the hallway. In group 1, we choose a right-turn trace seed and begin the simulation in an open area filled with several

individual cylinders. In group 2, we choose a zigzag trace seed to get through a closed area, the starting and the ending points are narrow and the middle of the path is broad. In group 3, we choose a zigzag trace seed to move from a closed area to an open area.

In each group, we process two individual experiments by using the same sensor dataset. The first experiment is using a combination of Lidar data flow to process function1, 2, and 3. The result is the line (Process with functions 1, 2, and 3) in Fig.8.The second experiment is using single Lidar data flow to process function1 and 2 after step3. The results contain two traces in Fig.8: green line (Process with function 1 and 2) and brown line (Process with function 1 and 2 offsets).

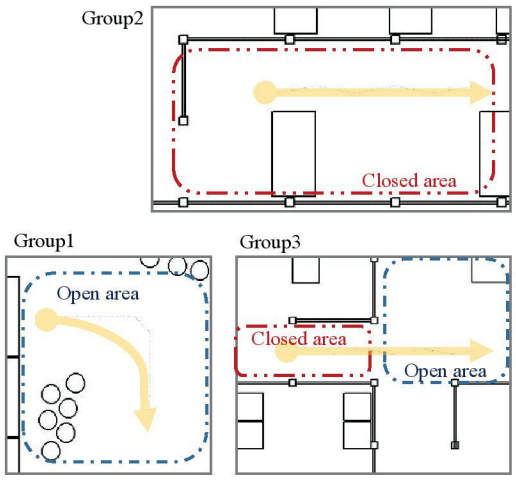

Fig. 7. Trace and surroundings.

In Fig.8, the left is a sample of the Lidar map, which is used as an input of the phase correlation process. The right is the graph comparing estimation and real trace groups in different surroundings and settings. In group 1, the result without pose-graph correlation is continuous. But the estimations of distance and angle per movement are far from the truth. The result with pose-graph correlation gets almost currency in estimation per movement, but it has a disruption for one second different from related estimations. In group 2, all results have similar inflections. For estimation of the ending point, the result with pose-graph correlation is close to the truth, and the one without correlation is far from the endpoint. The main and offset sensing result without correlation have obvious distinctions. In group 3, all results are in similar estimation except the offset one occurs some disruption different to related estimations.

We use formula (11) to calculate the accuracy for each group. In this formula, $\Delta\text{real}|t$ is the real movement and vector $[rx, ry]$ is the estimation results.

$$\Delta\text{real}|t = \sqrt{(p_t|x - p_{t-1}|x)^2 + (p_t|y - p_{t-1}|y)^2}$$

$$\text{Accuracy} = \frac{1}{n-1}\sum_{t=2}^{n}\frac{\text{abs}|\sqrt{rx|t^2 + ry|t^2} - \Delta\text{real}|t|}{\Delta\text{real}|t} \times 100\% \quad (11)$$

In Table I, we show the settings and accuracy of the results. Case 2 get the highest accuracy, 91.10%, in an open area with rich features with pose-graph correlation. Case 1 get the worst accuracy, 31.25%, with the same settings in case 2 except pose-graph correlation. Cases 3, 4, 5 and 6 get the same accuracy between 60% and 65% in different settings, and the accuracy gets a little improved due to the richer feature of the surroundings.

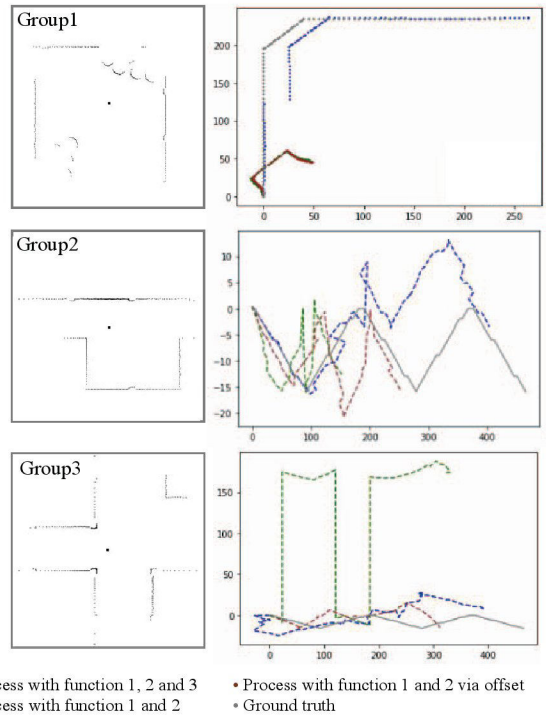

Fig. 8. Sample of Lidar map and trace estimation.

TABLE I. ESTIMATION CURRENCY IN GROUP

Case	Surroundings		Method		Experiment (Fig.8)		Result
	Area	*Feature*	*Phase correlation*	*Pose-graph*	*Group*	*Line*	*Accuracy*
1	open	rich	yes	no	1	brown	31.25%
2	open	rich	yes	yes		blue	91.10%
3	close	low	yes	no	2	brown	60.08%
4	close	low	yes	yes		blue	60.31%
5	close to open	medium	yes	no	3	brown	64.45%
6	close to open	medium	yes	yes		blue	63.54%

C. Discussion

According to the results, when we focus on long-term stability, the process with all functions activated performs better. When we focus on the influence of surroundings, the rich feature with all functions activated performs an accuracy

of 91.10% as best—the rich feature cause more reliable estimation. Our pose-graph correlation has three designs to avoid the errors: loop-closing design via ground truth; the fixed estimation vector formula design (10); avoiding using phase correlation algorithm to process the pair of objects which is too close to each other. Thus, pose-graph correlation does make a great improvement from phase correlation in the system. However, the results are unstable due to the unpredictable huge errors.

V. Conclusion

We propose a multiple Lidar-based obstacle detection and motion estimation system. We use pure Lidar-base sensors with low-cost performance to prove the potential of our system. The pose-graph correlation is a utility method for solving the loop-closing problem, especially the totally blind system like automobiles which has no ground truth connection and mostly relies on the loop of measurement so that error may occur and make a no limited extension. Besides, the use of phase correlation is extended not just for image problems but also contributes to other sensor fields. As for the accuracy of our system, basically not less than 60.31% and up to 91.10% when the surroundings support it. However, unpredicted error and instability are also a problem.

For further research, it is possible to develop the exclusive phase correlation algorithm when we can avoid transferring the sensor data to an image to fit the input of phase correlation. The abundance of image transfer may decrease the cost or we can add more sensor data to improve the accuracy. In addition, the simulation has their limitation so it is also necessary to process those experiments on hard devices to get further research.

References

[1] S. Arshad, G.-W. Kim, "Role of Deep Learning in Loop Closure Detection for Visual and Lidar SLAM: A Survey," Sensors, 2021, vol.21, no.1243

[2] L. Zhou and Z. Deng, "A New Algorithm for the Establishing Data Association Between a Camera and a 2-D Lidar," Tsinghua Science and Technology, 2014, pp.314-322

[3] J. Aulinas, Y. Petillot, J. Salvi, X. Lladó, "The SLAM problem: a survey," vol.184: Artificial Intelligence Research and Development, 2008, pp.363-371

[4] H. Guo, D. Cao and H. Chen, "Vehicle dynamic state estimation: state of the art schemes and perspectives," IEEE/CAA Journal of Automatica Sinica, vol.5, no.2, pp.418-431, March 2018

[5] T. Taketomi,H. Uchiyama, S. Ikeda, "Visual SLAM algorithms: a survey from 2010 to 2016," IPSJ T Comput Vis Appl, 2017, vol.9, no.16

[6] M. P. Kevin, "Bayesian map learning in dynamic surroundings," Adv. Neural Inf. Process. Syst. 2000, vol.12, pp.1015-1021

[7] D. Talwar and S. Jung, "Particle Filter-based Localization of a Mobile Robot by Using a Single Lidar Sensor under SLAM in ROS Environment," 2019 19th International Conference on Control, Automation and Systems (ICCAS), 2019, pp. 1112-1115

[8] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. M. Montiel and J. D. Tardós, "ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual–Inertial, and Multimap SLAM," IEEE Transactions on Robotics, vol. 37, no. 6, pp. 1874-1890, Dec. 2021

[9] F. Wang, Z. Zhen, C. Liu, Z. Mi, Bri-Mathias Hodge, Miadreza Shafie-khah, João P.S. Catalão, "Image phase shift invariance based cloud motion displacement vector calculation method for ultra-short-term solar PV power forecasting," Energy Conversion and Management, 2018, vol.157, pp.123-135

[10] B. Jähne, H. of Computer Vision and Applications, 1999, Vol.2 Signal Processing and Pattern Recognition, pp.35-38

[11] Y, Jianli Wang, K. Yao, "Modified phase correlation algorithm for image registration based on pyramid," Alexandria Engineering Journal (2022) 61, pp.709–718

[12] X. Chen, T. Läbe, A. Milioto, To Röhling, Olga Vysotska, Alexandre Haag, Jens Behley, Cyrill Stachniss, "OverlapNet: Loop Closing for Lidar-based SLAM," Robotics: Science and Systems Foundation, 2020

[13] D. Galvez-Lopez and J. D. Tardos, "DBoW2 library for C++: Bag-of-word image database for image retrieval," IEEE Transactions on Robotics, Bags of Binary Words for Fast Place Recognition in Image Sequences, 2012, vol.28, pp.1188-1197

[14] Q. Zhang, G. Xu, Na Li, "Improved SLAM loop-closing detection algorithm based on DBoW2," Journal of Physics: Conference Series, 2019, vol.1345, issue4

[15] L. Carlone, R. Aragues, José A. Castellanos and B. Bona, "A fast and accurate approximation for planar pose-graph optimization," The International Journal of Robotics Research, 2014, vol.33, no.7 pp. 965-987

[16] L. Carlone and G. C. Calafiore, "Convex Relaxations for Pose Graph Optimization With Outliers," IEEE Robotics and Automation Letters, vol. 3, no. 2, pp. 1160-1167, April 2018