Sensors and Materials, Vol. 34, No. 3 (2022) 1091–1104
MYU Tokyo

S & M 2875

Optimization and Path Planning of Simultaneous Localization and Mapping Construction Based on Binocular Stereo Vision

Neng-Sheng Pai,* Wei-Zhe Huang, Pi-Yun Chen, and Shih-An Chen

Department of Electrical Engineering, National Chin-Yi University of Technology, Taichung 41170, Taiwan (ROC)

(Received June 19, 2021; accepted August 23, 2021)

Keywords: simultaneous localization and mapping (SLAM), random sample consensus (RANSAC), D*Lite, closed-loop detection

The aim of this study is to help individuals easily reach their destinations independently whenever they are situated in an unfamiliar environment. After performing pre-processing optimization, a map constructed by simultaneous localization and mapping (SLAM) is inputted to a route planning algorithm to find the most suitable path. The proposed method utilizes a stereo camera sensor to take images of the environment, after which it conducts 2D/3D mapping through the SLAM framework before converting the constructed map into images. Then, obstacles are identified using an image segmentation method, and pseudo-obstacles are filtered out through optimization. Finally, route planning is conducted using the D*Lite algorithm. Experimental results revealed that most of the pseudo-obstacles can be filtered out through image optimization, thereby increasing the accuracy of the 2D map.

1. Introduction

For the planning of ideal routes, route planning is conducted through the analysis of a map that is constructed using simultaneous localization and mapping (SLAM).[1] Casarrubias-Vargas described the integration of information obtained in extended Kalman filter SLAM (EKF-SLAM) through machine learning.[2] Although the SLAM algorithm has been continuously updated and optimized, there is always the possibility of obtaining inaccurate information. For example, obstacles incorrectly generated by SLAM may not actually exist, and these pseudo-obstacles can lead to poor route planning and increased computation. Therefore, in this study, we apply some optimization methods and conduct pre-processing on a map constructed by SLAM and use the pre-processed map as an input of a route planning algorithm. The commonly used route planning methods include the probabilistic roadmap (PRM),[3] rapidly exploring random trees (RRT),[4] Dijkstra's algorithm,[5] and the A* search algorithm.[6]

The PRM, which is a method based on an image search, randomly labels nodes (the number of nodes is self-defined) in a given image. Then, it uses a search algorithm, such as Dijkstra's algorithm, to find the route on a road map. Although it has the advantage of using only a small number of nodes to search for the path, it yields poor results if there are too few nodes or if their distribution is not rational. Meanwhile, RRT uses an initial node as the root node and generates a

*Corresponding author: e-mail: pai@ncut.edu.tw
https://doi.org/10.18494/SAM3474

ISSN 0914-4935
https://myukk.org/

random tree through random sampling to produce leaf nodes. When the leaf nodes of the random tree enter the target area or contain the target node, a route from the initial node to the target node can be found in the random tree. Its advantage is that it can efficiently and rapidly search a high-dimensional space in which the search is directed to a blank area through the randomly sampled nodes in the state space. However, it has the same disadvantages as the PRM. In comparison, Dijkstra's algorithm takes a certain node as the starting node and calculates the shortest route from that node to all other nodes. Its advantage is that it can obtain the best solution, i.e., the shortest route. However, the shortest distance for all nodes must be calculated, thereby requiring a large amount of computation, which results in low efficiency.

The A* search algorithm is a heuristic search algorithm that is an improvement of Dijkstra's algorithm. It increases the search speed and searches for the best route by calculating the minimum cost of each node to the target node. On the one hand, it can avoid the unnecessary calculation of the cost to irrelevant nodes. On the other hand, it cannot identify the best route when there are multiple minimum costs. The D*Lite algorithm is similar to the A* algorithm, and the most significant feature of the former is that it starts from the target node and searches reversely to the starting node.[7] Its advantage is that it has a very effective route search process within a dynamic environment. Thus, in this study, we adopt the D*Lite algorithm for route planning.

We aim to optimize a map constructed by SLAM, so that route planning can be used to obtain the ideal route more accurately. We identify the obstacles through image segmentation and effectively filter out the pseudo-obstacles to reduce the calculation time for path planning. In this way, we can also reduce the error rate of path planning. In this paper, we discuss SLAM, map optimization, and route planning. Furthermore, the ideal route is achieved using the 2D map generated by SLAM, which is then inputted to the route planning algorithm through optimization.

2. Hardware and System Environment

In this study, we designed an optimized SLAM and applied it to route planning using the hardware structure shown in Fig. 1. It includes two parts, the SLAM end and the route planning end. The structure of the system is shown in Fig. 2.

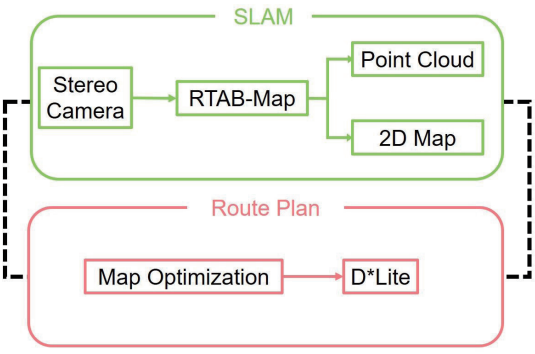

Fig. 1. (Color online) Structure of the hardware.

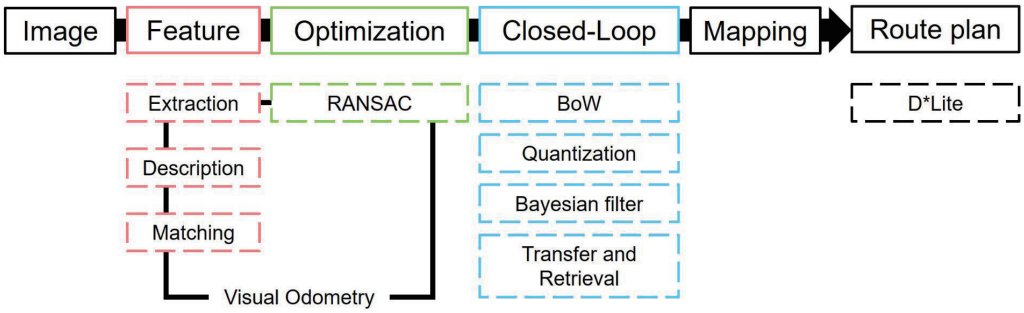

Fig. 2. (Color online) Structure of the system.

The images of the stereo camera and the data from the visual odometer are inputted to the RTAB-Map algorithm to obtain a 3D point cloud and a 2D map,[8] after which the 2D map is sent to the route planning end. The route planning end first optimizes the 2D map and then plans the route through the D*Lite algorithm.

3. Map Construction by SLAM

In SLAM, a visual odometer is used to determine the direction and position of camera movement. The feature points are extracted and matched through the speeded up robust features (SURF) characteristic algorithm, after which the moving direction can be calculated from these matched points, as shown in Fig. 3.[9]

3.1 Feature extraction

The process of feature extraction includes three main parts: feature extraction, feature description, and feature matching. In feature extraction, the Hessian matrix in Eq. (1) is used to generate the key points.[10]

$$f(x, y) = \begin{bmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial f}{\partial x}\frac{\partial f}{\partial y} \\ \frac{\partial f}{\partial x}\frac{\partial f}{\partial y} & \frac{\partial^2 f}{\partial y^2} \end{bmatrix} \tag{1}$$

In SURF, a box filter is used to replace the Gaussian filter, which can greatly improve the computational speed. The equation for determining the generated key points is given by

$$\det(H) = D_{xx} \times D_{yy} - 0.9 \times D_{xy} - D_{xy}. \tag{2}$$

The reason for multiplying by 0.9 is that the box filter assumes $\sigma = 1.2$ and a template size of 9×9 as the smallest scale space, where $\sigma = 1.2$ is selected from the authors' experience when inventing the box filter. In addition, $(L_{xx}/D_{xx})(L_{yy}/D_{yy})$ is a constant and does not affect the

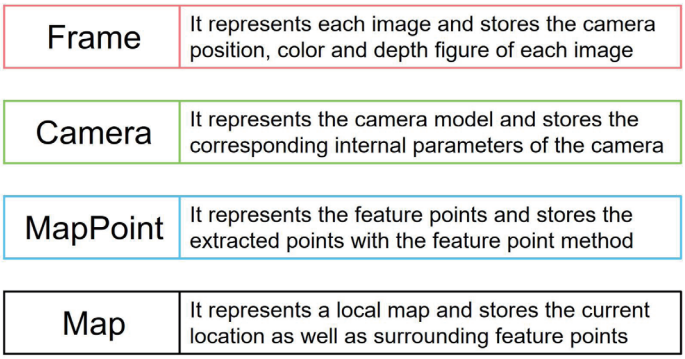

Fig. 3. (Color online) Structure of the visual odometer.

comparison of the extreme points. Thus, we calculated $\frac{|L_{xy}(1.2)|}{|D_{xy}(9)|}\frac{|D_{xx}(9)|}{|L_{xx}(1.2)|}$ to be $0.912 \approx 0.9$. Finally, we used non-maximum suppression to compare all key points with the surrounding 26 points, and the one with the largest value is determined as a feature point.

The Haar wavelet transformation is adopted in the feature description.[11] With the feature point as the center, we can thus calculate the area within a radius of $6s$ (where s is the scale value of the feature point). The sum of all points within the 60° sector for the Haar wavelet response in the x and y directions is calculated, and the values obtained after Gaussian weighting are the horizontal and vertical components, respectively, as shown in Fig. 4. The blue arrow in Fig. 4(c) shows the main direction.

After finding the main direction, the feature point is surrounded by a square frame with a subarea of size 4 × 4. The sum of the Haar features of 25 pixels in the horizontal and vertical directions is calculated for each subarea, thereby yielding a $4 \times 4 \times 4 = 64$-dimensional vector as the feature description, as shown in Fig. 5.

Finally, the matching degree is determined by the Euclidean distance through the two feature points in the feature matching, wherein a shorter distance represents a better matching degree. Furthermore, SURF includes an additional determination. If the signs of the Hessian matrices of the two feature points are the same, this signifies that the two features have the same direction. In contrast, different signs indicate that the directions are opposite and can thus be directly excluded.

3.2 Optimization

In visual SLAM, image optimization is used as the main method. Here, we adopt the random sample consensus (RANSAC) algorithm for optimization, after which we estimate the parameters of the mathematical model from the observed data by iteration. Firstly, we assume that some of the data are internal points and then we use these points to design a model. Next, we input all of the data and apply this model to determine whether the number of internal points is sufficient. The model is adjusted through the internal points if their number is sufficient; otherwise, the original model is discarded, and a new one is redefined. The process of the RANSAC algorithm is shown in Fig. 6.[12]

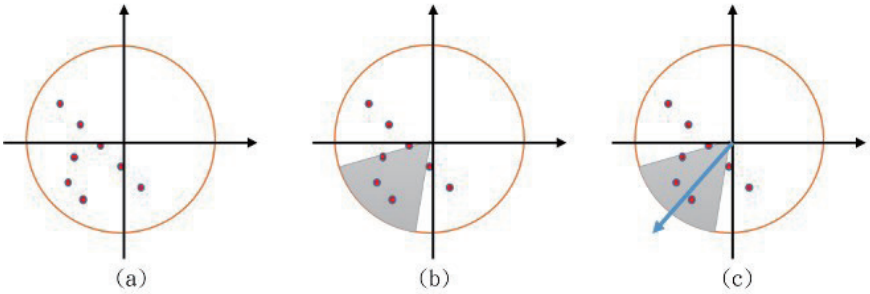

Fig. 4. (Color online) Main direction of the feature.

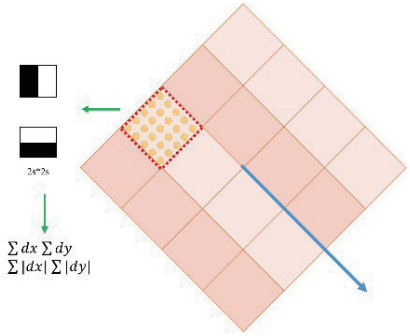

Fig. 5. (Color online) Feature description.

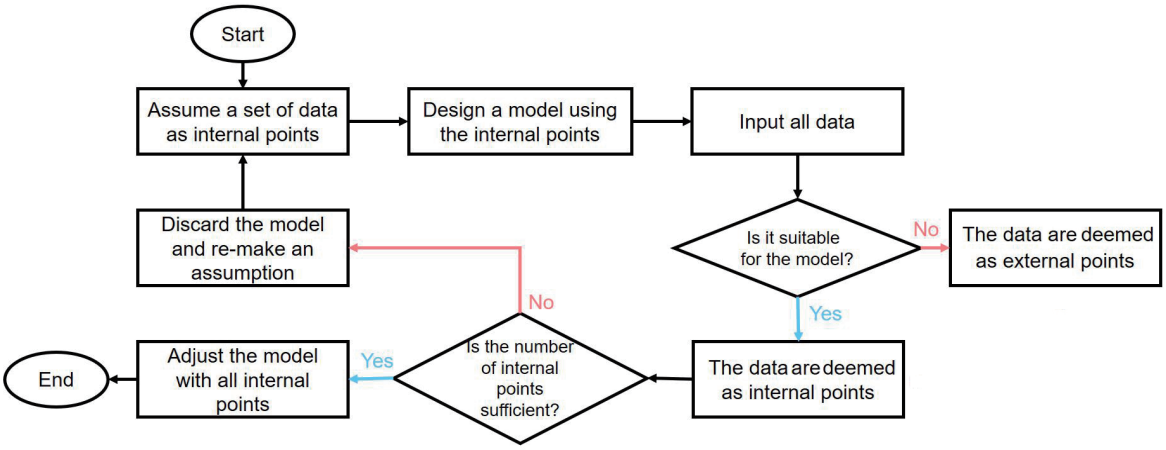

Fig. 6. (Color online) Flowchart of the RANSAC algorithm.

When estimating the model parameters, p represents the probability that all internal points are real internal points and w represents the probability that a real internal point is chosen by random selection. This is expressed by

$$w = \frac{\text{Number of internal points}}{\text{Total number of points}}. \tag{3}$$

In addition, k is the number of iterations, n is the number of points required for the model estimation, and w^n is the probability that all n points are internal points. Hence, the following equation is satisfied:

$$1 - p = (1 - w^n)^k \tag{4}$$

The advantage of RANSAC is that it can stably estimate the model parameters and the parameters in big data with high precision. However, because it adopts an iterative method, the results may not be the optimal solution, especially if the number of iterations is insufficient. At the same time, it requires a large amount of computation if the number of iterations is too high.

The RANSAC parameter matrix is defined as

$$s\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}, \tag{5}$$

where (x, y) is the position of the target image, (x', y') is the position of the scene image, s is the scale, and $h_{33} = 1$. These variables are used to calculate the homography matrix, after which the model can be used to test all data, compute the number of data points that satisfy this model, and determine the projection error e, where

$$e = \sum_{i=0}^{n}\left(x' - \frac{h_{11}x_i + h_{12}y_i + h_{13}}{h_{31}x_i + h_{32}y_i + h_{33}} \right)^2 + \left(y' - \frac{h_{21}x_i + h_{22}y_i + h_{23}}{h_{31}x_i + h_{32}y_i + h_{33}} \right)^2 . \tag{6}$$

Here, a smaller e value indicates a better model.

3.3 Closed-loop detection

The objective of closed-loop detection is to correct errors through a closed loop. In calculating each frame, some errors inevitably occur, and these will increase with the consecutive computations of several frames. Therefore, a closed loop is used to determine and correct any error when returning to places that were visited previously. The flowchart of the detection process is shown in Fig. 7. As can be seen, the detection process uses the bag-of-words model (BoW) to preliminarily determine whether this frame of the places is a closed loop and whether the closed loop is achieved using Bayesian filtering.[13,14]

The BoW is a type of document representation in the field of information retrieval. It creates a dictionary and uses it to generate a model with a high degree of differences. After inputting an image, the number of feature points within each word is calculated through the dictionary, as a result of which the feature histogram of an image is obtained and stored. Finally, the new feature histogram is compared with all the old feature histograms, and if their similarity is greater than a threshold, then the closed-loop detection is considered to be successful.

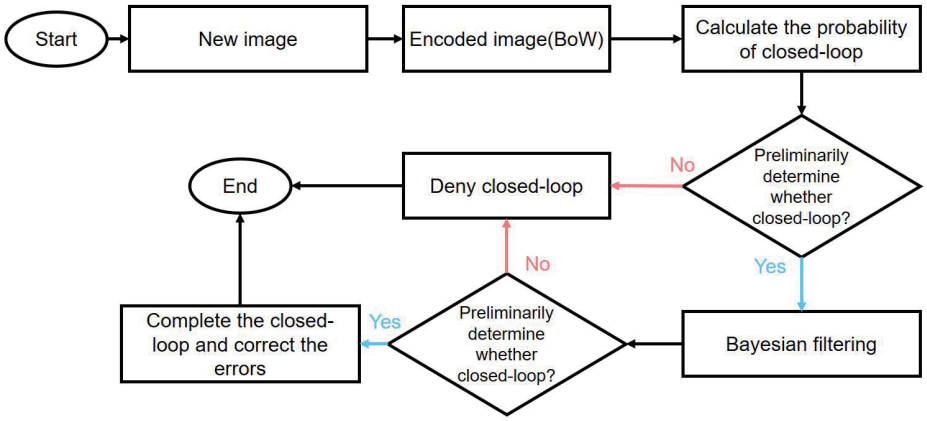

Fig. 7. (Color online) Closed-loop detection process.

After determining the success of closed-loop detection, Bayesian filtering (which applies Bayes' theorem) is performed for further closed-loop detection. On the basis of probabilistic statistics, we can determine whether a closed loop is formed by observation and prediction. Bayes' theorem is expressed as

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}. \tag{7}$$

The Bayesian filtering equation can be expressed as

$$bel(x_t) = p(x_t|u_{1:t}, z_{1:t}), \tag{8}$$

where x_t is the location at time t, $u_{1:t}$ represents all controls from time 1 to time t, and $z_{1:t}$ represents all observations from time 1 to time t. Substituting Eq. (8) into Eq. (7) results in the following:

$$bel(x_t) = \frac{p(z_t|x_t, z_{1:t-1}, u_{1:t}) \times p(x_t|z_{1:t-1}, u_{1:t})}{p(z_t|z_{1:t-1}, u_{1:t})}, \tag{9}$$

which can be simplified to

$$bel(x_t) = \eta p(z_t|x_t) \times \int p(x_t|u_t, x_{t-1}) bel(x_{t-1}) dx_{t-1}. \tag{10}$$

The first part, $\int p(x_t|z_{1:t-1}, u_{1:t}, x_{t-1}) bel(x_{t-1}) dx_{t-1}$, involves x_{t-1} and u_t and is used to predict the state of x_t. The second part, $\eta p(z_t|x_t)$, is used to update the state of x_t by observing z_t.

In addition, the RTAB-Map algorithm can enhance the computational speed without losing its precision by using a retrieval and transfer method. During the closed-loop detection, we compare this frame of the feature histogram with that of each frame in the working memory

(WM). When the closed-loop detection ends, if the feature histogram of the frame surrounding the location with the highest closed-loop probability is not in the WM, then it is retrieved from the long-term memory (LTM) and transferred to the WM. If the time for image computation is too long, closed-loop detection is used to find the feature histogram with the lowest weight and transfer it from the WM to the LTM.

4. Route Planning

There are two main types of route planning: global planning and regional planning. Global planning identifies a static mapping route, whereas regional planning finds a dynamic mapping route, wherein some of the maps are re-planned to avoid collisions when encountering sudden obstacles. Once the 2D map is constructed by SLAM, binarization is conducted on the output 2D map[15] to determine the positions of the obstacles. Then, the image is cut, and the pixel value of each block is summed. If the sum is greater than a threshold, then it is assumed that there is no obstacle in the block; otherwise, the block remains unchanged. This reduces the number of pseudo-obstacles generated by changes in light or fast movement during mapping.

4.1 D*Lite algorithm

As mentioned previously, the D*Lite algorithm divides a map into several nodes and searches for the best route through the evaluation of nodes. The key principle of the algorithm is that it assumes all of the unknown areas to be free spaces and conducts route planning on this basis. The shortest distance can be found by searching for the smallest value of $rhs(s)$, which is a provisional value based on $g(s)$. This provisional value is used to find the distances from s to all its surrounding nodes and the corresponding g values. The smallest value is used as the rhs value of the current node s, which is defined below.[7]

The D*Lite algorithm is derived from the LPA* algorithm, and it initially finds the optimum route from the starting node to the target node using the environment information provided by SLAM.[16] In this algorithm, each node has an initial $g(s)$ value that is infinite and is defined as the distance of the shortest route from the current node s to the goal. Whether this value needs to be updated is determined by $rhs(s)$.

$$rhs(s) = \begin{cases} 0, & \text{if } s = s_{goal} \\ \min\limits_{s' \in Succ(s)} \left(g(s') + c(s, s') \right), & \text{other nodes} \end{cases} \tag{11}$$

Here, the node s_{goal} is our goal node defined as 0; $s' \in Succ(s)$ is the successor node of node s, which means that the route can proceed from s to all other nodes; and $c(s', s)$ is the distance between nodes s' and s. When $g(s) = rhs(s)$, it is considered that there is local consistency; otherwise, there is local inconsistency. There are three types of consistency:

Local consistency (LC) occurs when $g(s) = rhs(s)$. When all nodes are locally consistent, the value of $g(s)$ is equal to the shortest distance from node s to the starting node. Hence, s' (the next node in the direction of the starting node) can be obtained from $\min(g(s') + c(s', s))$.
Local overconsistency (LOC) occurs when $g(s) > rhs(s)$, which means that the connected state of the node is better than before, i.e., the value of $c(s', s)$ can be decreased. This also indicates that an obstacle is cleared or a shorter route can be found, after which the value of g is updated to $rhs(s)$ to return to LC.
Local underconsistency (LUC) occurs when $g(s) < rhs(s)$, which indicates the sudden occurrence of an obstacle, which increases the length of the route from s' to the starting node. It is necessary to recalculate the node s and the related node after node s. When the g value of this node is taken out, its g value is set to infinity. Hence, the node will have LOC, after which LC will be achieved through the processing method of LOC to achieve the shortest route.

Meanwhile, to increase the efficiency of the algorithm, we set a *Key* value for each node and prioritize the nodes that may be found on the shortest path. The *Key* value is defined as follows:[7]

$$Key = \begin{bmatrix} K_1 \\ K_2 \end{bmatrix} = \begin{bmatrix} \min\left[g(s), rhs(s)\right] + h(s_{start}, s) \\ \min\left[g(s), rhs(s)\right] \end{bmatrix}, \tag{12}$$

where $h(s_{start}, s)$ is the estimated distance from node s to the starting node.

Next, the key values of each node are compared, and the value of K_1 for each node is compared first when the nodes are partially inconsistent ($g(s) \neq rhs(s)$). If the nodes have the same K_1, then this is compared with K_2, and the smaller value of K_1 and K_2 is selected.

5. Results of Experiment

In our experiment, we obtained the map information with a ZED stereo camera. We also used a handheld stabilizer during the mapping process, so that the 3D visual map and the 2D map used for route planning could be constructed stably and accurately. We also fabricated the yellow-green structure in Fig. 8 using a 3D printer, which was used to fix and balance the ZED camera on the handheld stabilizer.

Fig. 8. (Color online) Stereo camera attached to the handheld stabilizer.

5.1 Result of SLAM

The mapping in this experiment was conducted at Mingxiu Lake and on the second floor of the engineering hall of National Chin-Yi University of Technology. For the mapping conducted at Mingxiu Lake, shown in Fig. 9, the relevant data and parameters of the experiment are given in Table 1. The obtained 3D and 2D maps are shown in Figs. 10 and 11, respectively. In addition, for the mapping conducted on the second floor of the engineering hall, the relevant data and parameters are shown in Table 2. The obtained 3D and 2D maps in this case are shown in Figs. 12 and 13, respectively. Tables 3 and 4 show the time details for each part of the system in each mapping.

Fig. 9. (Color online) Photograph of Mingxiu Lake, National Chin-Yi University of Technology.

Table 1
Data of the experiment at Mingxiu Lake.

Type	
Construction time	12 min 19 s
Construction distance	337.73 m
Word count	294331
LTM nodes	713
WM nodes	692
Number of successful closed loops	3

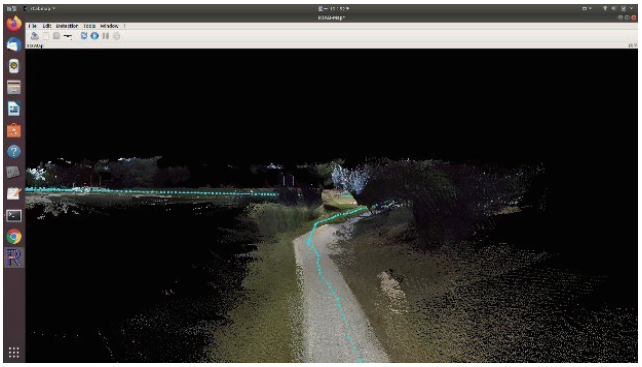

Fig. 10. (Color online) 3D map of Mingxiu Lake, National Chin-Yi University of Technology.

Fig. 11. 2D map of Mingxiu Lake, National Chin-Yi University of Technology.

Table 2
Data of the experiment at the second floor of the engineering hall.

Type	
Construction time	13 min 50 s
Construction distance	249.37 m
Word count	329440
LTM nodes	814
WM nodes	778
Number of successful closed loops	5

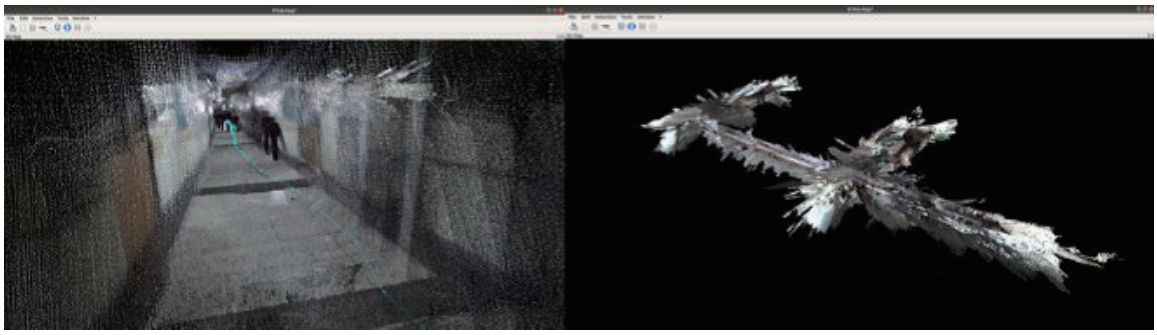

Fig. 12. (Color online) 3D map of the second floor of the engineering hall.

Fig. 13. 2D map of the second floor of the engineering hall.

Table 3
Time data of the experiment at Mingxiu Lake.

Type	Average time	Longest time
Feature extraction	46.53 ms	68.64 ms
Feature description	4.65 ms	11.86 ms
Similarity transformation matrix	0.63 ms	1.52 ms
BoW	131.25 ms	3.68 s
Time per frame	201.34 ms	3.81 s

Table 4
Time data of the experiment at the second floor of the engineering hall.

Type	Average time	Longest time
Feature extraction	68.52 ms	115.12 ms
Feature description	7.31 ms	14.68 ms
Similarity transformation matrix	0.42 ms	6.15 ms
BoW	253.71 ms	4.61 s
Time per frame	332.22 ms	4.78 s

5.2 Result of route planning

To reduce the calculation time, the 2D map constructed by SLAM was binarized before the D*Lite algorithm was used for route planning, as shown in Fig. 14. During the experiment, the map structure may be misjudged due to changes in light, object color, camera movement speed, and so forth. Therefore, we adopted image segmentation to segment each 4 × 4 pixels for the judgment. If the average value of these 16 pixels exceeds 150, these 16 pixels are defined as a pseudo-obstacle. The optimized map after all pixels are processed is shown in Fig. 15. The results show that many pseudo-obstacles leading to incorrect judgments have been removed. In addition, Fig. 16 shows the successful route planning realized by the D*Lite algorithm.

The experimental results confirmed that the proposed image optimization technique can indeed eliminate most of the incorrect obstacle nodes and that the 2D map can also obtain more accurate results. Finally, we used the D*Lite algorithm to calculate the best path in the environment, thus enabling users or a mobile robot to follow a certain path and reach the destination easily and autonomously.

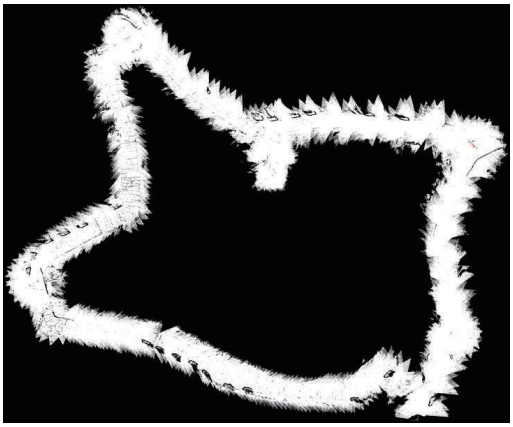

Fig. 14. Binary 2D map of Mingxiu Lake before optimization.

Fig. 15. Binary 2D map of Mingxiu Lake after optimization.

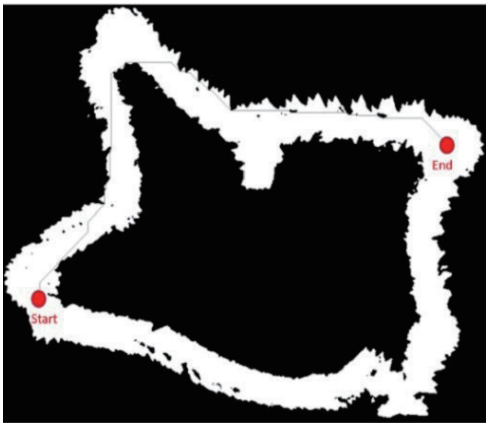

Fig. 16. (Color online) Result of route planning.

6. Conclusions

This study mainly included two parts: the SLAM system and the route planning system. To construct a map, a ZED stereo camera was used to generate the image input, and the feature of each frame of the image was searched for and matched through the feature point method to enable a visual odometer to determine the position and direction of the camera movement and optimize it through the RANSAC algorithm. In addition, closed-loop detection was used to correct the judgment errors of the odometer. BoW was used to calculate the image of each frame and create a feature histogram to preliminarily assess whether the closed loop is successful by comparing histograms in pairs. Then, we determined whether or not this frame is in a closed loop using Bayesian filtering. The points in the LTM were retrieved back to the WM when they were in a closed loop.

For route planning, pre-processing was conducted on the map to enhance the computational speed and reduce the number of pseudo-obstacles generated by the SLAM. In addition, obstacles were identified through image segmentation, after which we effectively filtered out the pseudo-obstacles to reduce the calculation time for path planning, thus reducing the error rate of path planning. Finally, the route was planned using the D*Lite algorithm.

Acknowledgments

This work was supported by the Ministry of Science and Technology, Taiwan, under contract number MOST 108–2221-E-167–025 (duration: August 1, 2019–July 31, 2020).

References

1 S. H. Chan, P. T. Wu, and L. C. Fu: Proc. 2018 IEEE Int. Conf. Systems, Man, and Cybernetics (IEEE, 2018) 1263–1268.
2 H. Casarrubias-Vargas, A. Petrilli-Barceló, and E. Bayro-Corrochano: Proc. 2010 20th Int. Conf. Pattern Recognition (IEEE, 2010) 396–399.
3 L. E. Kavraki, P. Svestka, J. C. Latombe, and M. H. Overmars: IEEE Trans. Rob. Autom. **12** (1996) 566. https://doi.org/10.1109/70.508439
4 S. M. LaValle and J. J. Kuffner: Proc. 1999 IEEE Int. Conf. Robotics and Automation (IEEE, 2002) 473–479.
5 E. W. Dijkstra: Proc. 1959 Numerische Mathematik Conf. (Numerische Mathematik, 1959) 269–271.
6 P. E. Hart, N. J. Nilsson, and B. Raphael: IEEE Trans. Syst. Sci. Cybern. **4** (1968) 100. https://doi.org/10.1109/TSSC.1968.300136
7 S. Koenig and M. Likhachev: Proc. 18th National Conf. Artificial Intelligence (ACM, 2002) 476–483.
8 K. T. D. S. De Silva, B. P. A. Cooray, J. I. Chinthaka, P. P. Kumara, and S. J. Sooriyaarachchi: Proc. 2018 18th Int. Conf. Advances in ICT for Emerging Regions (IEEE, 2018) 433–438.
9 H. Bay, T. Tuytelaars, and L. Van Gool: Proc. 2006 European Conf. Computer Vision (ECCV, 2006) 404–417.
10 P. Liu, L. Xiao, J. Zhang, and B. Naz: IEEE Trans. Geosci. Remote Sens. **54** (2016) 2235. https://doi.org/10.1109/TGRS.2015.2497966
11 Q. Chen, N. D. Georganas, and E. M. Petriu: IEEE Trans. Instrum. Meas. **57** (2008) 1562. https://doi.org/10.1109/TIM.2008.922070
12 P. C. Niedfeldt, K. Ingersoll, and R. W. Beard: IEEE Trans. Aerosp. Electron. Syst. **53** (2017) 461. https://doi.org/10.1109/TAES.2017.2650818
13 F. X. Zeng, Y. Ji, and M. D. Levine: IEEE Trans. Image Process. **27** (2018) 1443. https://doi.org/10.1109/TIP.2017.2778561
14 B. T. Vo, B. N. Vo, and A. Cantoni: IEEE Trans. Signal Process. **56** (2008) 1313. https://doi.org/10.1109/TSP.2007.908968

15 Q. Chen and T. Yao: Proc. 2009 Int. Conf. Natural Language Processing and Knowledge Engineering (IEEE, 2009) 1–5.
16 S. Koenig, M. Likhachev, and D. Furcy: Artif. Intell. **155** (2004) 93. https://www.sciencedirect.com/science/article/pii/S000437020300225X

About the Authors

Neng-Sheng Pai received his B.S. and M.S. degrees from the Department of Automatic Control Engineering of Feng Chia University, Taichung, Taiwan, ROC, in 1983 and 1986, respectively. In 2002, he was awarded a Ph.D. degree from the Department of Electrical Engineering, National Cheng Kung University, Tainan, Taiwan, ROC. He is currently a professor in the Department of Electrical Engineering, National Chin-Yi University of Technology, Taichung, Taiwan, ROC. He was the chairman of the department from 2004 to 2007 and was also the chairman of the Computer Center of National Chin-Yi University of Technology from 2013 to 2017. His current research interests include fuzzy systems, artificial intelligence, image processing, advanced control systems, and microprocessor systems.
(pai@ncut.edu.tw)

Wei-Zhe Huang received his B.S. degree from the Department of Electrical Engineering, National Chin-Yi University of Technology, Taichung City, Taiwan, in 2020, where he has been pursuing his M.S. degree since 2020. His research interests include IoT, deep learning, and image processing.
(baydigh@gmail.com)

Pi-Yun Chen received her Ph.D. degree from the Graduate School of Engineering Science and Technology, National Yunlin University of Science & Technology, Yunlin, Taiwan, in 2011. She is currently an associate professor and also the chief of the Department of Electrical Engineering, National Chin-Yi University of Technology, Taichung City, Taiwan, where she has been since 2019. Her current research interests include neural network computing and its applications, fuzzy systems, and advanced control systems.
(chenby@ncut.edu.tw)

Shih-An Chen received his B.S. degree from the Department of Electrical Engineering, National Chin-Yi University of Technology, Taichung City, Taiwan, in 2018, where he has been pursuing his M.S. degree since 2018. His research interests include SLAM, deep learning, and image processing.
(pig01kq19@gmail.com)