An MICP Algorithm for 3D Map Reconstruction Based on 3D Cloud Information of Landmarks

Ba-Viet Ngo, Thanh-Hai Nguyen, and Duc-Dung Vo

***Abstract* — This paper aims to reconstruct 3D map based on environmental 3D cloud information of landmarks. A Modified Iterative Closest Point (MICP) algorithm is proposed to apply for merging point clouds through a transformation matrix with values updated using the robot's position. In particular, reconstruction of 3D map is performed based on the location information of landmarks in an indoor environment. In addition, the transformation matrix obtained using the MICP algorithm will be up-dated again whenever the error among the point clouds is greater than one setup threshold. The result is that the environmental 3D map is reconstructed more accurately compared using the MICP. The experimental results showed that the effectiveness of the proposed method in improving the quality of reconstructing 3D cloud map.**

***Index Terms* — RGB-D cameras, transformation matrix, 3D point clouds, MICP algorithm, 3D map reconstruction.**

I. Introduction

The problem of reconstructing 3D maps based on RGB-D cameras has gradually attracted researchers in recent years [1], [2]. Compared with 2D maps, 3D maps contain more valuable environmental information, especially there is information of route planning [3]-[5]. To obtain the whole information of a 3D map, the most important problem is to compute the variability of point clouds with different perspectives. The procedure for calculating the coordinate system transformation parameters is called Motion Estimation (ME). To solve this problem, researchers have proposed different solutions [6]-[8] and one of the most commonly used solutions is the ICP algorithm [9]. This algorithm effectively gathers more precise point clouds, but still has some limitations. In addition, the ICP algorithm is a kind of point-to-point registration method, in which if there is an increase in the number of points, the efficiency of this algorithm decreases obviously. Moreover, the initial position of the point plays an important role in the registration process, particularly the initial position of the point is inconsistent, it will cause the registration result related to calculation of local optimization and seriously affect the accuracy of this algorithm.

According to the characteristics of the ICP algorithm, researchers applied to develop researches and proposed various innovative algorithms. In particular, Izadi presented a real-time 3D reconstruction, in which an interaction system combined with GPU technology to improve operational efficiency [10]. However, this method is highly demanded about hardware performance such as big memory, and so it is impossible to complete the reconstruction of large-scale scene. In [11], authors proposed an innovative ICP algorithm based on the point-to-plane, in which computation speed increased. On the other hand, this method is difficult to converge and to be stable when the surface curvature has a large change. Mitra proposed an improved ICP algorithm based on feature point connection [12]. This method significantly improved the operational performance compared with the original ICP algorithm. However, due to the presence of exceptions, the registration accuracy of this algorithm is poor. In another research, Gibson Hu represented the RANSAC algorithm to solve the ICP model based on the corresponding points and features for improving the accuracy and robustness of the ICP algorithm [13]. However, the similarities randomly selected in the ICP model and this made the local cloud merging results, and so it did not meet the global map's requirements. Article [14] proposed to optimize the global map using a General Graph Optimization (G2O) framework combined with key frames for reconstructing the 3D map. Authors tested the performance of their proposed algorithm in six public datasets. The results demonstrate that the algorithm is feasible and effectively.

In recent years, approaches to RGB-D SLAM have been entirely focused on using raw depth [15] or combining depth and RGB information [16], [17] to predict camera movement. Combination techniques have generally been shown to be more robust due to including both visual and geometric information for motion estimation [18]. However, with processing complex indoor environments, two scenarios can happen and it makes the traditional RGB-D algorithm to be able to have errors. In particular, where there are expansive spatial scenes, little information about depth, flat scenes a lot of geometrical structure. This is why these techniques have limitations for reconstructing 3D images of office-style environments.

Kinect RGB-D systems has been used to replace stereo camera systems for robot localization [19]. This type of the RGB-D not only produces 3D images with the high precision but also enables calculation of fast machining. In addition, it is much cheaper than a 3D sensor with the same function and easy to install for use. That is why algorithms based on the RGB-D have been applied for determining spaces in motion

Submitted on April 05, 2021.
Published on April 26, 2021.
Ba-Viet Ngo, Department of Industrial Electronic, Biomedical Engineering, Faculty of Electrical, Electronics Engineering, HCMC University of Technology and Education, Vietnam.
Thanh-Hai Nguyen, Department of Industrial Electronic, Biomedical Engineering, Faculty of Electrical, Electronics Engineering, HCMC University of Technology and Education, Vietnam.
(e-mail: nthai@hcmute.edu.vn).
Duc-Dung Vo, Department of Industrial Electronic, Biomedical Engineering, Faculty of Electrical, Electronics Engineering, HCMC University of Technology and Education, Vietnam.

as well as locations of autonomous robots in recent years [20], [21]. One problem of this RGB-D is that its depth information is often noisy. Therefore, if a robot is equipped with the Kinect for moving a long distance, the cumulative error with robot localizations during moving increases over time [22]. There have been many proposed methods such as the consideration of noise characteristics, updating the distance dependence to reduce this error as well as improving the accuracy of 3D mapping [23]-[25].

In this paper, we propose an MICP algorithm based on the calculation of the robot's position and 3D data from the point clouds. Robot position in 3D space will be calculated based on landmarks in the indoor environment. These landmarks are identified in the previous coordinates through the process of identifying and collecting landmarks. In addition, the 3D data from the point clouds helps to calculate the deviation angle between 2 frames, from which it is possible to calculate the deviation angle of the robot in space. Finally, a transformation matrix will be computed based on this information and make the cloud merging process more accurate.

II. The Proposed 3D Map Reconstruction Algorithm

A. RGB-D Mapping Framework

Environmental 3D mapping algorithm consists of two main and independent processes as shown in Fig. 1. In the first step, data is collected by the RGB-D camera system while the robot is in motion. The collected data includes the point clouds and the robot's locations is determined based on the locations of the landmarks in the environment. In the second step, the data is processed to reconstruct the 3D environment map.

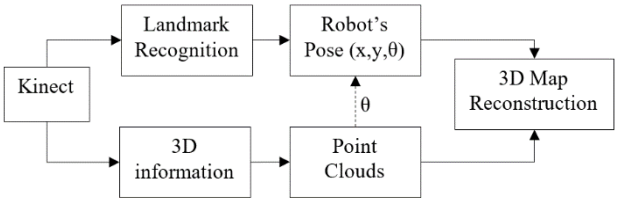


Fig. 1. Overview of the proposed 3D map reconstruction algorithm.

In order to reconstruct the 3D map, it is necessary to capture in depth images of an indoor environment and the locations where they were collected. Before starting to collect any data, the robot must be localized in the 2D map obtained earlier. This means that an initial exploration must be performed, including the identification of landmarks in a natural environment [30] and the positioning of these landmarks in a 2D environment. Once data collection has started, two different processes run in parallel. Figure 1 shows how the Kinect camera collects data and manages the robot's localization for providing the robot's location for each point cloud.

B. Cloud's Filtering

Once all data is saved, reconstructing 3D map is begun by processing these data to produce the final 3D map. This is done in two main processes: Firstly, every cloud is processed to reduce its size; Secondly, clouds are added to the final map by registering them. In practice, each cloud image is obtained with the Kinect has about 300000 points. The cloud with the large will take more time for calculation and a lot of memory for saving. For this reason, filtering the cloud image will reduce resource usage and computation time during processing 3D cloud images. The process of filtering consists of four steps: Filtering; Reducing pattern; Removing unnecessary points; and Reconstructing [26].

Pass Through filter: This filter is used to set a depth limit in the cloud. The Kinect has a working range of 3.5 meters. However, it is actually obtained points further than this distance threshold. We have found that in order to perform accurate reconstruction, it is necessary to use points farther than 3.5 meters, although the accuracy of the depth will be decreased as the distance between the points and the Kinect will be increased. For this reason, a Pass Through filter should be applied for removing any point that is more than 6 meters deep from the Kinect.

Down Sampling: Two points in a point cloud captured from the Kinect can have a minimum distance of a few millimeters. In the case of mapping environments, its area is about some square meters, it is unnecessary to obtain the minimum distance with such accuracy. It is acceptable to reduce the accuracy of the reconstructed 3D image by reducing the size of all point clouds and significantly both computation time and memory usage. In particular, the Voxel Grid [27] represented how to reduce the number of points by dividing a point cloud in the "voxels" boxes of 5 cm-sided cubes by the desired width. Then all points in a box are reduced into a single point corresponding to their centers. In this way, it is possible to set the minimum distance between points with the desired accuracy for reducing the number of points of a point cloud.

Remove Outliers: Point clouds from the Kinect can have measurement errors that can produce sparse outliers. Such points can lead to errors in the surface normalization of the local point cloud. Calculations of this type often require investigating a certain number of neighborhood points in an adjacent area of one point, so it is important to ensure that the neighborhood points are correct. Furthermore, removing some unnecessary points contributes to a reduction in processing time, although its effect in this algorithm is less important compared to the two processes as shown above. The method based on statistical analysis on the neighborhoods of each point was applied to eliminate outliers [28]. The average distance from each point to all of its neighborhoods is calculated. With the Gaussian distribution, mean and standard deviation, all points have the average distance outside a certain range are called outliers and they will be removed from the data set. In this study, 50 neighborhood points were used for each point to analyze its state and it will remove all points with distances greater than the standard deviation of the average distance to the analyzed point.

Surface Reconstruction: Surface reconstructing is used to improve the elimination of data anomalies. It is based on the Moving Least Squares (MLS) algorithm [28]. In particular, the MLS provides a reconstructed surface for a given set of points by interpolating higher-order polynomials among rounded local neighborhoods. Smoothing and resampling a

noisy cloud allows to more accurately estimate of surfaces and curvature. Such estimates are further used to merge point clouds together.

C. Modified ICP Algorithm for 3D Point Cloud Reconstruction

In theory, positioning the robot would allow to reconstruct the entire map by applying translations and rotations to all point clouds. After the map initialization for each new cloud, we merge it with the previous map and decide if such merging is good enough to insert it into the map. This process will allow to consider the quality of the newly merged cloud based on the Fitness Score (FC) value.

The MICP algorithm is a well-known process that aligns two sets of point clouds by minimizing the Euclidean distance between their corresponding points [9]. It finds the closest pairs of 3D points in the source and target which are identifies as objects if their distance is less than a specified distance. Therefore, it estimates a transformation that minimizes the distance between them and the iterations until the difference between consecutive transformations is less than the defined limit or the maximum iterations is reached. However, the MICP can have some problems due to its convergence is at the local minimum. Therefore, it can produce poor merging results and in order to improve this one, it may need large iterations. The MICP also returns a parameter, called the fitness score, that provides information about the quality of adjusting. The fitness score corresponds to the error of distance between clouds adjusted after the merging process. This parameter will be useful to decide under which cases of adjustment is good enough for inserting it into the map.

After adjusting the new cloud using the MICP as shown in Fig. 2, this c merged loud will be performed the second adjustment, if the result is not accepted. In this case, an MICP is applied to get better first prediction than that of the ICP. After applying the second MICP, the result will be used to decide whether or not to use the new cloud on the map. Fig. 2 shows the structure of cloud merging process using the MICP method.

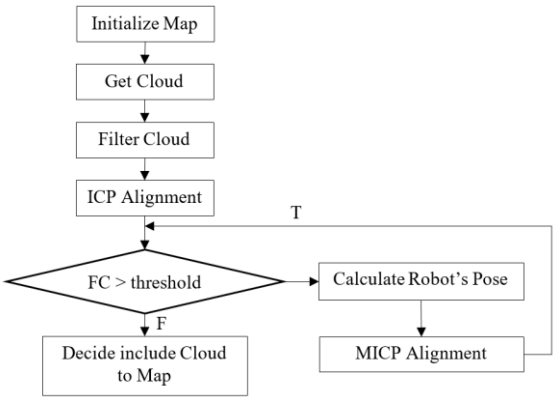


Fig. 2. The process of the 3D map rescontruction based on merging point clouds.

1) Map Initialization

The first point cloud and its location are used to initialize the first map. This process applies a transformation matrix to shift the cloud from its local reference position to its global reference position. P^{K}_{i} is the i^{th} point cloud in its local reference space and P^{G}_{i} is the corresponding point cloud in the global reference space. At this time, the cloud is ready to be set as the first map so that any accepted new cloud will be inserted into. Fig. 3 shows steps for performing according to this method.

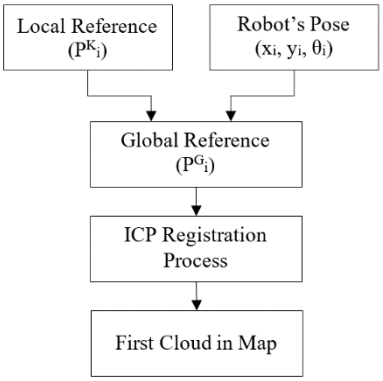


Fig. 3. Representation of the block diagram for the initial process of the first map.

2) Building a new cloud

For a new cloud merging process, a new cloud is obtained after filtering and the corresponding global position of the camera system is collected. With using this information, this cloud can be transformed into a global reference. In order to combine different clouds and create a common map, all of them must be in the same reference.

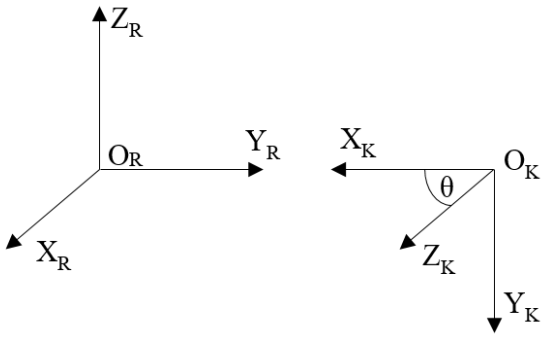


Fig. 4. The coordinates of the Kinect and Robot.

The obtained point clouds are referenced to the Kinect camera's coordinate R_K. Its original point O_K is at the camera location and its X_K, Y_K and Z_K axes are determined as shown in Fig. 4. In particular, it corresponds to a moving coordinate system during the robot movement for acquisition of different clouds. In addition, the robot's reference system R_R, as shown in Fig. 4, is a mobile system. Therefore, to adjust the cloud with the robot's reference system, the transformation $\mathbf{T^R_K}$ is applied to transform the cloud with the Kinect coordinate system into the Robot coordinate system using the following formula:

$$\mathbf{T}_{\mathbf{K}}^{\mathbf{R}} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1)$$

When the cloud is in the appropriate local reference system, it is necessary to transform the cloud into the global reference system similar to other clouds. R_G is the fixed coordinate system and has the origin and direction as R_R when the position of the robot is (X_0, Y_0, θ_0) = (0,0,0). In practice, the robot moves around the floor, so the robot has the unchanged coordinate Z_G and the plane (X_G - Y_G) with the camera system is considered as not to move to the R_R. Fig. 5 shows the relationship between the coordinates of R_G and R_R.

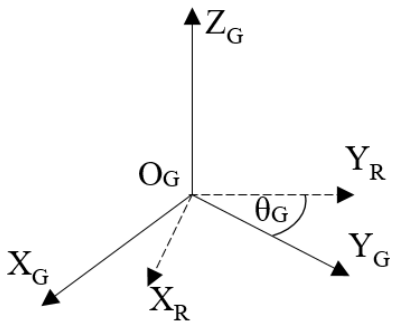


Fig. 5. Representation of the coordinates of R_G and R_R.

To transform the reference cloud from the coordinate system R_R to the global coordinate system R_G, the $\mathbf{T^G_R}$ transformation is described as follows:

$$\mathbf{T}_{\mathbf{R}}^{\mathbf{G}} = \begin{bmatrix} \cos(\theta_G) & -\sin(\theta_G) & 0 & X_G \\ \sin(\theta_G) & \cos(\theta_G) & 0 & Y_G \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2)$$

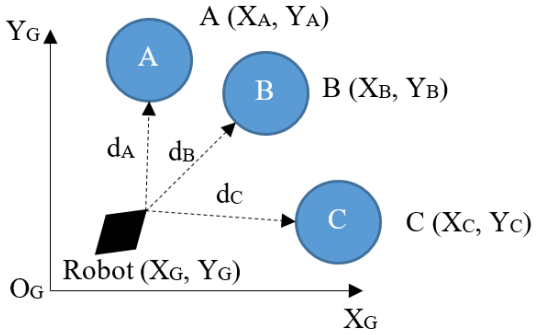


Fig. 6. The coordinates of the robot and surface landmarks.

The coordinates (X_G, Y_G) of the robot in the plane are determined by the position of 3 landmarks in that plane [29]. In Fig. 6, assuming that the robot moves in the flat space OX_GY_G with the undefined coordinate (X_G, Y_G). Moving space is assumed to be an absolute plane for obstacles chosen to be landmarks. These obstacles have distinctive features from other landmarks [30] and their positions are determined in space A(X_A, Y_A), B(X_B, Y_B), and C(X_C, Y_C), in which the distances from the robot to the landmarks are d_A, d_B, d_C, respectively.

To find two coordinate components X_G and Y_G, it is necessary to find the solution of the system, including 3 equations (3), (4), (5). To solve this system of the equations, it is necessary to determine two important parameters, the coordinates of the landmarks and the distance from the robot to these landmarks. In addition, the SURF landmark recognition algorithm is employed [31] to identify landmarks on the way of the robot's movement. This algorithm allows to find feature points on the landmark appearing on the image frame collected from the camera installed with the robot and matches the feature points of the landmarks stored in the library. When the robot wants to locate its position through landmarks, the closest landmarks will be detected and then they are used as the positioning reference. After finding the landmarks in the image frame, the distances from the robot to the corresponding landmarks d_A, d_B, d_C are determined based on the depth image obtained from the Kinect camera. The coordinates of the landmarks and the distance from the robot to the corresponding landmark will be used to determine the current coordinates of the robot using the following equations:

$$(x_A - x)^2 + (y_A - y)^2 = d_A^2 \quad (3)$$

$$(x_B - x)^2 + (y_B - y)^2 = d_B^2 \quad (4)$$

$$(x_C - x)^2 + (y_C - y)^2 = d_C^2 \quad (5)$$

In addition, the rotation angle θ_G of the robot around the O_GY_G axis is determined based on the calculation of the rotation angle θ between the point clouds. Assume that there are 2 point clouds **A** and **B**, the covariance matrix, **H** is calculated using the following equation:

$$\mathbf{H} = \left(\mathbf{A} - \frac{1}{N}\sum_{i=1}^{N}\mathbf{A}^i\right)\left(\mathbf{B} - \frac{1}{N}\sum_{i=1}^{N}\mathbf{B}^i\right)^T \quad (6)$$

There are some ways to find the optimal rotation between point clouds. The Singular Value Decomposition (SVD) [32] method is considered as "a powerful magic wand" in linear algebra to determine the rotation matrix R as follows:

$$[\mathbf{U},\mathbf{S},\mathbf{V}] = SVD(\mathbf{H}) \quad (7)$$

$$\mathbf{R} = \mathbf{V}\mathbf{U}^{\mathbf{T}} \quad (8)$$

Therefore, the rotation angle can be extracted from the matrix R as follows:

$$\theta = a\tan(\mathbf{R_{21}}, \mathbf{R_{11}}) \quad (9)$$

The rotation angle of the robot in the global space at the i^{th} is calculated as follows:

$$\theta_G(i) = \theta_G(i-1) + \theta(i) \quad (10)$$

Finally, the cloud transformation matrix obtained from the Kinect in the global reference space will be calculated using the following equation:

$$\mathbf{T}_{\mathbf{K}}^{\mathbf{G}} = \mathbf{T}_{\mathbf{R}}^{\mathbf{G}} . \mathbf{T}_{\mathbf{K}}^{\mathbf{R}} \quad (11)$$

3) Decision of the inclusion

In this final step, the cloud merging process will decide whether or not to insert the converted cloud into the 3D map. To perform this one, the fitness score parameter given by the MICP is used. From experience, we recognize that clouds with the fitness score less than 0.01 are adequate for the map to be merged.

During a cloud merging process, overlapping areas between point clouds for adjusting are used and such overlap areas need to appear at all times. However, this is impossible to perform any time due to appearing accepted clouds and the common area of the previous clouds for reconstruction of 3D cloud map. Therefore, for improving this one, we decided to add to the 3D cloud map a cloud with an unsatisfactory fitness score if the last 3 clouds fail instead of using the cloud transformed by using the MICP. In particular, this cloud is inserted into the map by adding its points to the 3D map.

If a new cloud is added, all of its points will be added to the previous map. In this case, the overlapping areas will be denser due to containing two points at the estimated same location. In practice, it is unnecessary to have such density in the 3D cloud map and it can also decrease the accuracy of the map. Hence, a filtering process is necessary to be able to maintain the appropriate density in the map. The map was re-sampled using the Voxel grid, in which setting its size to 2 cm. Therefore, the map is filtered with the parameters similar to filtering the new cloud before added.

III. Results and Discussion

A. Description of Mobile Platform

In this study, the hardware system architecture of the mobile robot platform consists of the Kinect RGB-D connected to a PC and other processing equipment for data processing and control of the robot. After navigating to control the differential robot, velocity signals from the PC through the Driver controller are sent to the left and right wheels as shown in Fig. 7. Finally, all the mappings and the localization process are displayed on the PC screen during the robot's movement.

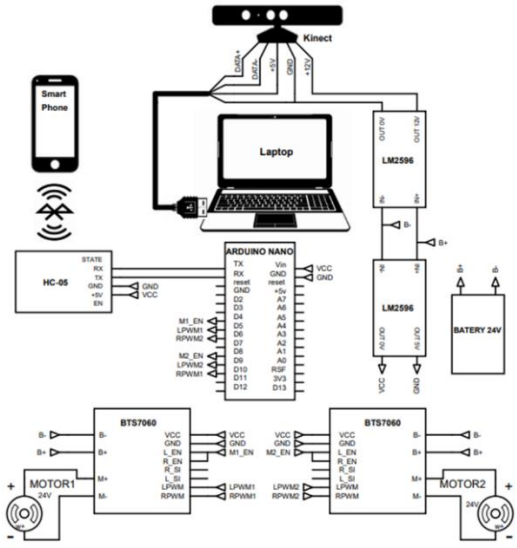


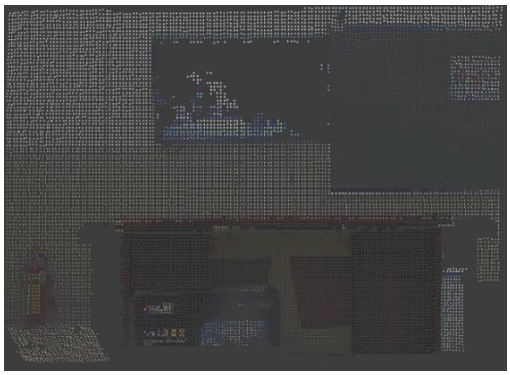
Fig. 7. Block diagram of the hardware system of the mobile platform.

Fig. 8. Robot model with the Kinect RGB-D and PC.

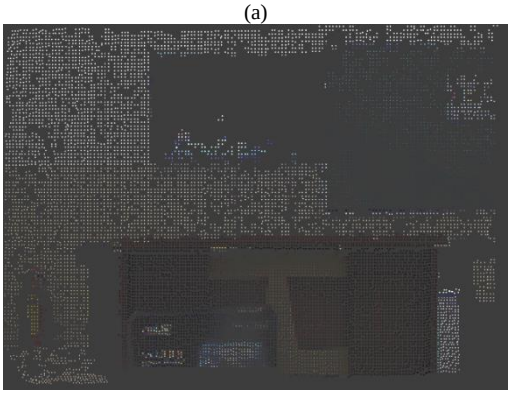
The robot with collecting environmental images is controlled via Bluetooth connection. The robot will move around a room to collect images for recognizing obstacles in the surrounding environment. The Kinect camera is installed with the front of the robot and it is setup a distance of 50 cm from the floor for collect almost obstacle images during the robot movement as shown in Fig. 8. In the hardware system, the rotor is installed with a laptop configured to a Core i3 processor, the CPU speed is 2.0GHz, the capacity of the integrated RAM in the laptop is 4 Gbyte. The Kinect camera and Arduino Nano are connected to the laptop by using a USB port for collect RGB images and corresponding depth images.

B. Results of Pre-processing Cloud Images

By preforming these steps, the number of points in the point cloud is significantly reduced. In this research, we propose the use of point clouds with less than 50000 points. This is usually obtained after filtering. However, in some cases, the point clouds still have a larger size compared to the setup size. In this case, the second filtering will be performed for reducing suitable points. This reduction will speed up the time of adjusting and this can avoid errors and obtain the adequate accuracy.



(a)



(b)

Fig. 9. Representation of processing clouds:
(a) Cloud image before filtering; (b) Cloud image after filtering.

C. Results of Determining the Robot Coordinate based on Landmarks

In order to determine the position of the robot, the most important step is to identify landmarks during movement. In particular, the landmarks are selected to calculate the robot position as shown in Fig. 10. In practice, the sample images may have different dimensions as well as details. In addition, each landmark with the coordinate is placed in a fixed position in the robot's movement space.

(a) Landmark image with two red fire extinguishers

(b) Landmark image with one black box

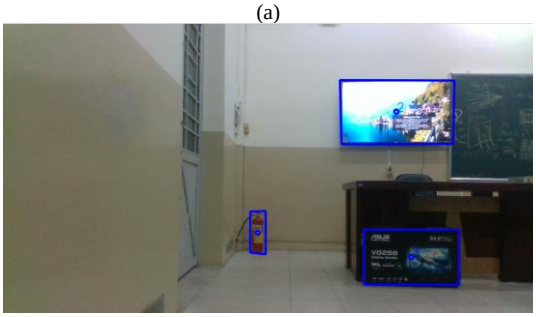
(c) Landmark image with blue box

(d) Landmark image with natural obstacles

Fig. 10. Landmark image chosen for identification.

In the robot position search algorithm, it is necessary to detect at least three landmarks. This means that when the three landmarks are identified, the center position of the landmark in the image frame captured from the camera system is determined. Therefore, this landmark information is used to determine the distance from the robot to the corresponding markers and the calculated results are compared with the actual measurement results. In particular, Fig. 11 depicts the landmarks obtained from the camera at different locations and the landmarks highlighted in blue using the SURF method. Therefore, according to the accurate calculation, the corresponding robot position will be returned close to the predicted result.

(a)



(b)

(c)

Fig. 11. Three landmarks captured from the camera system equipped with the robot at three different places:
(a) Three landmark positions at 1st place (Case-1);
(b) Three landmark positions determined at 2nd place (Case-2);
(c) Three landmark positions determined at 3rd place (Case-3).

Table I shows the different positions of the robot during its movement in 2D space and each position determined includes a standard value and another value is calculated using (3), (4), (5) for evaluating calculation errors under standard lighting conditions. While Table II shows the results of positioning the robot under low light conditions. Therefore, the SURF algorithm for landmark recognition is less dependent on the luminance change on the landmark image. With this main factor, when the landmarks are within the camera's visible area, the light intensity of the landmark images does not really affect the robot's positioning.

TABLE I: POSITIONS OF THE ROBOT DETERMINED USING THE LOCALIZATION ALGORITHM IN THE STANDARD LIGHT CONDITION

No.	(X_A,Y_A)	d_A	(X_B,Y_B)	d_B	(X_C,Y_C)	d_C	[Calculation position, standard position]
1	(450,1205)	2700 2030	(150,1065)	4050 4045	(300,903)	4400 4450	[294.781, 356.166] [297.513, 337.561]
2	(450,1205)	3900 3890	(150,1065)	4400 4420	(300,903)	4400 4442	[246.527, 504.365] [292.938, 506.473]
3	(600,1050)	3050 2920	(300, 900)	3000 2980	(450,1200)	3000 2985	[473.689, 244.784] [438.389, 298.139]

TABLE II: POSITIONS OF THE ROBOT DETERMINED USING THE LOCALIZATION ALGORITHM IN THE LOW LIGHT CONDITION

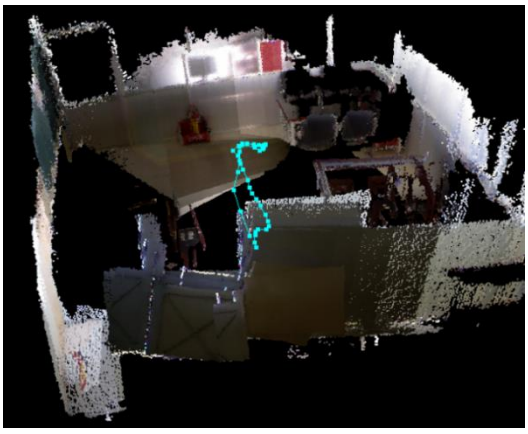
No.	(X_A,Y_A)	d_A	(X_B,Y_B)	d_B	(X_C,Y_C)	d_C	[Calculation position, standard position]
1	(450,1205)	2700 2030	(150,1065)	4050 4045	(300,903)	4400 4450	[294.781, 356.166] [297.513, 337.561]
2	(450,1205)	3900 3890	(150,1065)	4400 4420	(300,903)	4400 4442	[246.527, 504.365] [292.938, 506.473]
3	(600,1050)	3050 2920	(300, 900)	3000 2980	(450,1200)	3000 2985	[473.689, 244.784] [438.389, 298.139]

D. Experimental results of 3D mapping

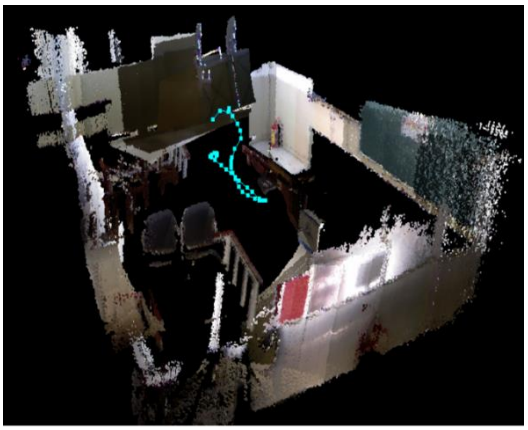
These Practical experiments we show how Kinect sensor can build 3D maps for made-up environment with dimensions 4800 mm x 4800 mm shown in Fig. 12. In this project Mobile robot is navigated manually by using Smartphone. This project is used Kinect with mobile robot for mapping by connecting Kinect to laptop. We control the mobile robot slowly on itself inside environment to start mapping.

Fig. 12. The environment for building maps.

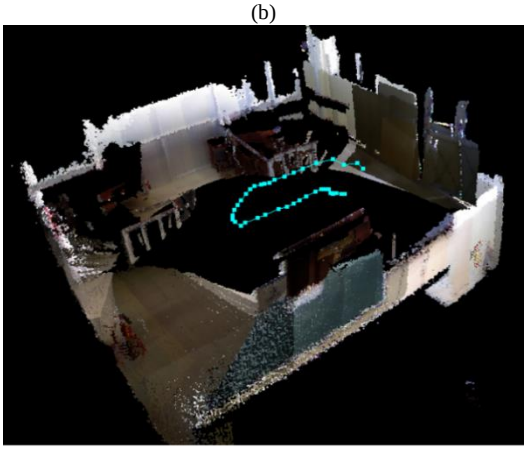
During mapping, if an error occurs, the map reconstruction system of the robot displays a warning and immediately stops the mapping. Therefore, the user should control the robot so that it can return back its correct original position. We have found that the error usually occurs when there are many areas without texture or blur motion as shown in Fig. 13. In particular, the blue line in the reconstructed map image shows the robot's path during data collection. The image of the experimental environment map from different angles shows that the 3D model of the indoor environment with good quality is successfully reconstructed.



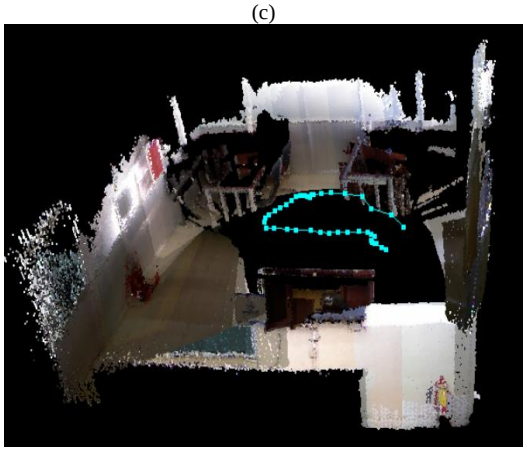
(a)



(b)



(c)



(d)

Fig. 13. 3D map of the indoor environment:
(a) Result of the 3D mapping of room at view 1;
(b) Result of the 3D mapping of room at view 2;
(c) Result of the 3D mapping of room at view 3;
(d) Result of the 3D mapping of room at view 4.

IV. CONCLUSION

In this paper, we have presented a MICP algorithm that improved the quality of reconstructing 3D maps using the Kinect cameras. Our approach used information from the 3D

point clouds and the position of the robot in its moving environment to compute the transformation matrix for the merging process of clouds. By evaluating the error of merging the point clouds during the 3D map reconstruction, we could eliminate point clouds which are not suitable for inserting into the 3D map. Finally, we reconstructed the 3D map with the high reliability of the indoor environment, and it can be used for robot localization, navigation and route planning.

ACKNOWLEDGMENT

This work is supported by Ho Chi Minh City University of Technology and Education (HCMUTE) under Grant T2020-02NCS. We would like to thank HCMUTE, students and colleagues for supports on this project.

REFERENCES

[1] J. Fuentes-Pacheco, "Visual simultaneous localization and mapping: A survey," *Springer Science - Business Media Dordrec*, pp. 55–81, 2015.
[2] X. Liu, B. Guo and C. Meng, "A method of simultaneous location and mapping based on RGB-D cameras," *in The 14th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, pp. 1-5, 2016.
[3] R. Z. M. Dr. Wael R. Abdulmajeed, "Comparison Between 2D and 3D Mapping for Indoor Environments," *International Journal of Engineering Research and Technology*, vol. 2, pp. 1-8, 2013.
[4] Nguyen Thanh Hai, N T Hung, "A Bayesian Recursive Algorithm for Freespace Estimation Using a Stereoscopic Camera System in an Autonomous Wheelchair," *American J. of Biomedical Eng*, vol. 1, pp. 44-54, 2011.
[5] N B Viet, N T Hai, N V Hung, "Tracking Landmarks for Control of an Electric Wheelchair Using a Stereoscopic Camera System," *in The Inter. Conf. on Advanced Tech for Communications*, pp. 12-17, 2013.
[6] Guanyuan Feng, Lin Ma, and Xuezhi Tan, "Visual Map Construction Using RGB-D Sensors for Image-Based Localization in Indoor Environments," *Journal of Sensors*, vol. 2017, pp. 1-18, 2017.
[7] B. Yuan and Y. Zhang, "A 3D Map Reconstruction Algorithm in Indoor Environment Based on RGB-D Information," *in The 15th International Symposium on Parallel and Distributed Computing (ISPDC)*, pp. 358-363, 2016.
[8] Thomas Whelan, Michael Kaess, Hordur Johannsson, Maurice Fallon, John J. Leonard, John McDonald, "Real-time large-scale dense RGB-D SLAM with volumetric fusion," *The International Journal of Robotics Research*, vol. 34, pp. 598 – 626, 2014.
[9] P. J. Besl and N. D. McKay, "A method for registration of 3-D shapes," *in IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, pp. 239-256, 1992.
[10] Izadi, et. all, "KinectFusion: Real-time 3D reconstruction and interaction using a moving depth camera", *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, pp. 559-568, 2011.
[11] Y Chen, G Medioni, "Object Modeling by Registration of Multiple Range Images", *Image and Vis. Computing*, vol. 10, pp. 145-155, 1992.
[12] Niloy J. Mitra, N. Gelfand, Helmut Pottmann, Leonidas J. Guibas, "Registration of Point Cloud Data from a Geometric Optimization Perspective", *ACM International Conference Proceeding Series*, vol.71, pp. 23-32, 2004.
[13] G. Hu, S. Huang, L. Zhao, A. Alempijevic and G. Dissanayake, "A robust RGB-D SLAM algorithm," *RSJ International Conference on Intelligent Robots and Systems*, pp. 1714-1719, 2012.
[14] B. Yuan and Y. Zhang, "A 3D Map Reconstruction Algorithm in Indoor Environment Based on RGB-D Information," *in 15th International Symposium on Parallel and Distributed Computing (ISPDC)*, pp. 358-363, 2016.
[15] S. Zhang and S. Qin, "An Approach to 3D SLAM for a Mobile Robot in Unknown Indoor Environment towards Service Operation," *Chinese Automation Congress (CAC)*, pp. 2101-2105, 2018.
[16] F. Endres, J. Hess, J. Sturm, D. Cremers and W. Burgard, "3-D Mapping With an RGB-D Camera," *in IEEE Transactions on Robotics*, vol. 30, pp. 177-187, 2014.
[17] G. Loianno, V. Lippiello and B. SicilianO, "Fast localization and 3D mapping using an RGB-D sensor," *in 16th International Conference on Advanced Robotics (ICAR)*, pp. 1-6, 2013.
[18] Huang A.S. et al, "Visual Odometry and Mapping for Autonomous Flight Using an RGB-D Camera," *in Christensen H., Khatib O. (eds) Robotics Research, Springer Tracts in Advanced Robotics*, vol. 100, pp. 235-252, 2017.
[19] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox, "RGB-D mapping: Using kinect-style depth cameras for dense 3d modeling of indoor environments," *The International Journal of Robotics Research*, vol. 31, pp. 647–663, 2012.
[20] H. Jo, S. Jo, H. M. Cho and E. Kim, "Efficient 3D mapping with RGB-D camera based on distance dependent update," *in 2016 16th International Conference on Control, Automation and Systems (ICCAS)*, pp. 873-875, 2016.
[21] Aguilar, Wilbert & Morales, Stephanie, "3D Environment Mapping Using the Kinect V2 and Path Planning Based on RRT Algorithms", *Electronics*, vol. 5, pp. 1-17, 2016.
[22] Hai T. Nguyen, Viet B. Ngo, Hai T. Quach, "Optimization of Transformation Matrix for 3D Cloud Mapping Using Sensor Fusion", *American Journal of Signal Processing*, vol. 8, pp. 9-19, 2018.
[23] C. Lim Chee, S. N. Basah, S. Yaacob, M. Y. Din, and Y. E. Juan, "Accuracy and reliability of optimum distance for high performance Kinect Sensor," *in Biomedical Engineering (ICoBE), 2nd International Conference on*, pp. 1-7, 2015.
[24] T. Yamaguchi, T. Emaru, Y. Kobayashi and A. A. Ravankar, "3D map-building from RGB-D data considering noise characteristics of Kinect," *in International Symposium on System Integration (SII)*, pp. 379-384, 2016.
[25] K. Lee, "Accurate Continuous Sweeping Framework in Indoor Spaces with Backpack Sensor System for Applications to 3D Mapping," *The IEEE Robotics and Automation Letters*, vol. 1, pp. 316-323, 2016.
[26] R. B. Rusu and S. Cousins, "3D is here: Point Cloud Library," *IEEE International Conference on Robotics and Automation*, pp. 1-4, 2011.
[27] Xian-Feng Han. et. all, "A review of algorithms for filtering the 3D point cloud", *Signal Processing: Image Communication*, vol. 57, pp. 103-112, 2017.
[28] Rusu, R.B, "Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments", *Künstl Intell*, vol. 24, pp. 345–348, 2010.
[29] Nguyen Tan Nhu, Nguyen Thanh Hai, "Landmark-Based Robot Localization Using a Stereo Camera System", *American Journal of Signal Processing*, vol. 5, pp. 40-50, 2015.
[30] Ba-Viet Ngo, Thanh-Hai Nguyen, "Dense Feature -based Landmark Identification for Mobile Platform Localization", *IJCSNS International Journal of Computer Science and Network Security*, vol.18, pp. 186-200, 2018.
[31] Bay H, Tuytelaars T, Van Gool L, "SURF: Speeded Up Robust Features", In: Leonardis A., Bischof H., Pinz A. (eds) Computer Vision – ECCV Lecture Notes in Comp. Science, Springer, vol. 3951, 2006.
[32] Liao Chengwang, "Singular Value Decomposition in Active Monitoring Data Analysis", *Handbook of Geophysical Exploration: Seismic Exploration*, vol. 40, pp. 421-430, 2010.

Ba-Viet Ngo received his M.Eng. in Electronics Engineering from HCMC University of Technology and Education in 2014. He is a Ph. D student in Electronics Engineering at HCM City University of Technology and Education. His research interests include smart wheelchair, artificial intelligence, image processing.

Thanh-Hai Nguyen received his BEng. degree with Electronics engineering from the HCMC University of Technology and Education, in Vietnam, 1995; MEng. One with Telecommunication and Electronics Engineering from HCMC University of Technology (UTE), in Vietnam, 2002; PhD. degree with Electronics Engineering from University of Technology, Sydney in Australia, 2010. Currently, he is a lecturer in the Department of Industrial Electronic – Biomedical Engineering, Faculty of Electrical - Electronics Engineering, the HCMCUTE, Vietnam. His research interests are Bio-signal and image processing, machine learning, smart wheelchairs and Artificial intelligence.

Duc-Dung Vo received his M.Eng. in Electronics Engineering from University of Transports and Communications in 2009. Currently, he is a lecturer in Electronics Engineering at HCM City University of Technology and Education. His research interests include electronics devices, artificial intelligence, image processing.