Large Area Inspection Using 3D Point Cloud Data in a Disaster Response Robot

Hiroaki Ogawa¹, Keito Shishiki¹, Uda A. Manawadu², and Keitaro Naruse^{1*}

¹Department of Computer Science and Engineering, University of Aizu, Aizu-Wakamatsu, Japan.

²Graduate School of Computer Science and Engineering, University of Aizu, Aizu-Wakamatsu, Japan.

Abstract.

Large area inspection using a robot is critical in a disastrous situation; especially when humans are inhabiting the catastrophic environment. Unlike natural environments, such environments lack details. Thus, creating 3D maps and identifying objects has become a challenge. This research suggests a 3D Point Cloud Data (PCD) merging algorithm for the less textured environment, aiming World Robot Summit Standard Disaster Robotics Challenge 2021 (WRS). Spider2020, a robotic system designed by the Robot Engineering Laboratory, University of Aizu, was used in this research. Detecting QR codes in a wall and merging PCD, and generating a wall map are the two main tasks in the competition. The Zxing library was used to detect and decode QR codes, and the results were quite accurate. Since the 3D mapping environment has fewer textures, decoded QR code locations are used as the PCD mapping markers. The position of the PCD file was taken from the location given by the robotic arm in Spider2020. The accuracy of merging PCD was improved by including the position of PCD files in the merging algorithm. The robotic system can be used for Large area Inspections in a disastrous situation.

1 Introduction

Disaster response robots were among the most valuable robots during the past few decades due to notable disasters that humans faced. When it comes to human-made disasters, it is too dangerous for humans to reach the disaster environment. In nuclear disaster incidents such as Chernobyl and Fukushima Daiichi, it was too risky for humans to enter the buildings to inspect the damages because radioactive materials were being released [1]. For these kinds of emergencies, it is necessary to use large area inspections remotely. A mobile rescue robot called Quince was used for surveillance missions at Fukushima Daiichi nuclear power plant [2]. To inspect the inside of the Fukushima Daiichi power plant, the research team considered several requirements from the robots that have been used. They considered mounting sensors and cameras for 3D mapping, which was identified as an excellent tool for understanding the disaster environment [1].

For addressing the issue of building a fully-equipped disaster response robot, World Robot Summit (WRS) introduced a task under Standard Disaster Robot Challenge. Those tasks were designed for assessing standard performance levels (e.g. mobility, sensing, information collection, wireless communication, remote control, on-site deployment and, durability) required in disaster prevention and responses of robots [3]. Large area inspection (Code name: Task EXP1) was designed to inspect a large area and recognize targets that spread widely over flat and curved

spaces, then generate a map of targets quickly and accurately. QR codes gave the targets/ markers. By detecting the positions of the QR codes accurately, a complete 3D map should be generated. According to the outline of rules provided by WRS, the following instructions were given [4].

1. Scanning should be started from the white box (see figure 1).
2. As in figure 1 robot should investigate the wall area and create a large map of the QR codes within the targets.
3. The submitted map's quality and accuracy will be thoroughly considered for the competition's final points.

As in figure 1 the width of the QR code may vary from 140 mm to 7 mm, and the mapping coordinates of the start zone should be $(x, y, z) = (0, 0, 0)$. Recognizing QR codes correctly and generating a map of the given wall with QR codes accurately are the basic facts that are checked in the competition.

As remote sensing technologies have rapidly developed in the last few years, 3D Point Cloud Data (PCD) is used to analyze textures in a wide area of fields [5]. PCD files can be used for creating large 3D maps. Rusu *et al.* have researched 3D Point cloud-based object maps for household environments [6]. They have mainly focused on large item detection, including cupboards, tables, drawers, and shelves, which are of particular importance for a

*e-mail: naruse@u-aizu.ac.jp

Figure 1. A sample curved surface is given in the WRS competition for mapping. Targets (QR codes) are placed at heights of less than 2.5m, over an area of approximately 4m² to 40m² [4].

household robotic assistant. Point cloud 3D mapping is famous for collecting 3D Data in urban areas and outside environments since the objects are large and easy to distinguish [7, 8]. Merging PCD files using visual similarities was used in the above research since they can be easily distinguished. Unlike in a natural environment, there are fewer textures in an artificial environment. So it has become an enormous challenge to identify objects using cameras inside such an environment, like inside a power plant or a factory.

According to previous researches, building an accurate and detailed 3D map in a less textured was never achieved. If it is achieved, it would be easily identified the objects in a less textured environment from a point cloud 3D map. A robot can automatically identify small buttons, levers, or valves using this kind of accurate 3D map, especially inside a factory. This research was mainly focused on generating an accurate 3D map in a less textured environment.

2 Methodology

The main objective of this research was to build an accurate point cloud from a less-textured environment. Since WRS task EXP1 is focused on the same goal, the research team mainly focused on the instructions and the guidelines given by WRS when building the robot system. When selecting the hardware components, the research team primarily focused on the ease of use, flexibility, equipment with a good Software Development Kit (SDK), and the hardware's size.

2.1 Spider2020 Robotic System

A robotic system called "Spider2020" designed by REL was used as the hardware interface (Figure 2). Spider2020 is made by combining several hardware components such as a robotic crawler, a robotic arm, several cameras, and an integrated system. The base of Spider2020 is a disaster response robot called "Giraffe", a crawler-type robot with two main crawlers and four sub crawlers. It also consists of a KINOVA JACO arm with 6 degrees of freedom curved wrist. The arm has a maximum height of 90 cm,

Figure 2. Spider2020 system. The red base is Giraffe robot and the Giraffe base was modified to put an external arm, external cameras, and external box PC.

which enables capturing PCD files of higher targets. Kinova SDK gives access to different parameters of the arm. Intel Realsense camera was attached to a stable position of a JACO arm (Figure 2). Intel Realsense D435 was used as the depth camera for this research. D435 has a maximum of 90 frames per second and that helped to easily identify QR codes quickly [9]. Functionalities that Realsense SDK gives to the RGB-D cameras were used for capturing PCD files.

2.2 Software Development

For the task EXP1 in the WRS challenge, QR decoding and its location estimation, merging the PCD files, and generating an accurate map are needed. For achieving those two tasks, two software systems were developed. One was for QR decoding using Zxing and the other for PCD merging, using Realsense SDK, and JACO arm SDK.

2.2.1 QR Decoding Methodology

A library called Zxing was used for QR decoding. This library is said to be more tolerant of poor environments than other QR libraries. Zxing is very popular and also used in Android's Barcode Scanner app [10]. The lighting of the competition environment cannot be guaranteed, so this QR library was chosen to prepare for the worst-case scenario. The algorithm used for Zxing first binarizes the image and emphasizes the black and white, then detects QR from the finder pattern embedded in the three corners. Finally, the decoding area is checked, and the QR code is decoded (As in figure 3).

The ratio of white cells to black cells is always 1: 1: 3: 1: 1 from any position of A, B, and C (refer figure 3), and even if the image is rotated, it is possible to detect the position and recognize the rotation angle from the positional

Figure 3. ZXing's QR finder pattern.

relationship. This is the factor that makes QR Codes resistant to rotation [11].

To decode multiple images from a single photo, the decoded QR is masked with a black area. In this way, the decoding process can be performed efficiently. If a QR code can be detected but cannot be decoded, the noise is removed from the QR detection pattern position, and the trapezoid is corrected and decoded again. This method successfully expands QR decoding's width (As in figure 4).

Figure 4. Expanded functions of QR decoding. Left is the whole image, and right is the QR after noise removal and trapezoidal correction. Below is the console screen at the time of execution.

For the position estimation and to find the real-world coordinates of the QR code, finding the center point and projecting its coordinates onto the points obtained from the Realsense is needed. For the given coordinates, the amount of movement of the JACO arm is applied. By applying it to all QRs, their positioning can be ascertained, and the better its accuracy, the better it will be evaluated in the competition.

2.2.2 Capturing and Merging PCD Files.

After a QR code is decoded, it is required to take a PCD file of the QR code location. After taking all the PCD files, two major problems arose when creating the 3D map of the wall.

1. Accurate location of the PCD file was hard to achieve only from the QR code.
2. Generate an Accurate 3D map needs more markers. Since the environment was less textured and only markers were QR codes, the accuracy of the final map would be a problem.

To address the first problem, the research team used JACO arm SDK to get an accurate arm position other than the QR code. Using JACO SDK, it is possible to get x, y, z-axis

and roll, pitch, yaw angles of the hand's near tip (position of end-effector) [12]. When capturing a PCD file position of the robot hand and the robot hand's rotation matrix, it was saved in a text file alongside the PCD file. Since the camera is aligned with the robot hand's tip, those coordinates were used as the camera position when taking the PCD file. Some calculations should be done to convert the robot arm position to camera positions. A Two-Stage Frame Transformation was used to transfer PCD files in the camera frame to the base frame and merge them in the base frame.

Figure 5. Camera frame to base frame transformation. $x_7 - y_7 - z_7$ of the camera frame, first transformed to $x_6 - y_6 - z_6$ to the hand frame then $x_0 - y_0 - z_0$ base frame

As in figure 2, Intel Realsense D435 is fixed to the JACO arm. So, the frame transformation from camera frame to hand frame is a fixed transformation (figure 5). The relationship between $x_7 - y_7 - z_7$ and $x_6 - y_6 - z_6$ is given from the following matrix T_{67} (Equation 1). 0m, 0.055m and -0.065m are the differences between x, y and z values in camera and hand position.

$$T_{67} = \begin{pmatrix} -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0.055 \\ 0 & 0 & -1 & -0.065 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (1)$$

Frame transformation from hand frame to base frame can be done from the hand position and orientation given by JACO SDK. It can be given by the matrix T_{06} (Equation 2).

$$T_{06} = \begin{pmatrix} 0 & 1 & 0 & x_6 \\ 0 & 0 & 1 & y_6 \\ 1 & 0 & 0 & z_6 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (2)$$

$x_0 - y_0 - z_0$ can be given from the following equation.

$$\begin{pmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{pmatrix} = T_{06} T_{67} \begin{pmatrix} x_7 \\ y_7 \\ z_7 \\ 1 \end{pmatrix} \quad (3)$$

From $x_0 - y_0 - z_0$ the accurate location of the PCD files were taken. With the accurate locations merging PCD files would be easy. Not all PCD files can be taken horizontally, they are tilted in various directions. There are roll, pitch, and yaw in the coordinate data of the arm, and affine

transformation matrix was performed using those values to straighten the tilted data [13]. Suppose affine transformation matrix is given by "A",

$$A = \begin{pmatrix} R & \begin{matrix} x \\ y \\ z \end{matrix} \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (4)$$

A single rotation matrix can be formed by multiplying the yaw, pitch, and roll rotation matrices to obtain a conversion. Suppose roll, pitch, and yaw were given by α, β, γ .

$$R = R_z(\alpha)R_y(\beta)R_x(\gamma)$$

$$= \begin{bmatrix} \cos(\beta) \cos(\alpha) & \cos(\alpha) \sin(\beta) \sin(\gamma) - \cos(\gamma) \sin(\alpha) & \cos(\gamma) \cos(\alpha) \sin(\beta) + \sin(\gamma) \sin(\alpha) \\ \cos(\beta) \sin(\alpha) & \cos(\gamma) \cos(\alpha) + \sin(\beta) \sin(\gamma) \sin(\alpha) & -\cos(\alpha) \sin(\gamma) + \cos(\gamma) \sin(\beta) \sin(\alpha) \\ -\sin(\beta) & \cos(\beta) \sin(\gamma) & \cos(\beta) \cos(\gamma) \end{bmatrix} \quad (5)$$

To address the second problem, shooting PCD files from far and near viewpoints was used [14]. The smaller the number of data, the more accurate the merging of PCD can be. From the distance viewpoint the details were unclear, But from the near viewpoints the details were clear and less noisy. Figure 6 summarizes the methodology that was used to capture and merge PCD files.

Figure 6. Capturing far (point 1 in the picture) and merge them. The merged PCD files only from far points are noisy (point 2). Using near PCD files (point 3) the output 2 is fine-tuned. The final output (point 4) was more accurate than the point 2 output.

Since the location and the proper rotation of the PCD files are available the research team used Matlab to merge 3-D PCDs [15].

3 Results

The system was tested in a robot demonstration held in Fukushima Robot Test Field on November 7-8, 2020. The QR code decoding algorithm that was used could identify every QR code that was given in WRS 2018. Apart from QR decoding, a 3D map was generated using the hand position as in figure 7. Twenty-five near and far PCD files were taken to create the 3D map.

Figure 7. Hand position of JACO arm was converted in to the base frame. x,y and z points of the each PCD files were represented as red, green and blue respectively.

The images of the final 3D map is given in figure 8. Using PCD file positions as in figure 7, PCD merging was successful and 3D maps can be constructed accurately. As in picture below of figure 8, the places without QR codes need to be improved.

Figure 8. Generated 3D map from the taken 25 PCD files as in figure 7.

4 Conclusion

For detecting and decoding QR codes, the results given by the Zxing library were quite accurate. Since there are fewer textures in the environment, merging only from visual points is not enough. The accuracy of merging PCD

can be improved by including PCD files' position into the merging algorithm. The final output can be improved if a higher number of PCD files are used for merging. As in figure 8 the points with low accuracy can be improved by taking more PCD files.

References

- [1] K. Nagatani, S. Kiribayashi, Y. Okada, K. Otake, K. Yoshida, S. Tadokoro, T. Nishimura, T. Yoshida, E. Koyanagi, M. Fukushima et al., *Journal of Field Robotics* **30**, 44 (2013)
- [2] K. Nagatani, S. Kiribayashi, Y. Okada, S. Tadokoro, T. Nishimura, T. Yoshida, E. Koyanagi, Y. Hada, *Redesign of rescue mobile robot Quince*, in *2011 IEEE International Symposium on Safety, Security, and Rescue Robotics* (IEEE, 2011), pp. 13–18
- [3] T. Kimura, M. Okugawa, K. Oogane, Y. Ohtsubo, M. Shimizu, T. Takahashi, S. Tadokoro, *Competition task development for response robot innovation in World Robot Summit*, in *2017 IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)* (2017), pp. 129–130
- [4] *World robot summit, standard disaster robotics challenge*. [online] . available:, <https://worldrobotsummit.org/en/wrs2020/challenge/disaster/standard.html>, accessed: 2020-12-03
- [5] M.E. Mohammadi, R.L. Wood, C.E. Wittich, *ISPRS International Journal of Geo-Information* **8**, 527 (2019)
- [6] R.B. Rusu, Z.C. Marton, N. Blodow, M. Dolha, M. Beetz, *Robotics and Autonomous Systems* **56**, 927 (2008)
- [7] N. Haala, M. Peter, J. Kremer, G. Hunter, *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci* **37**, 1119 (2008)
- [8] Y. Xu, V. John, S. Mita, H. Tehrani, K. Ishimaru, S. Nishino, *3D point cloud map based vehicle localization using stereo camera*, in *2017 IEEE Intelligent Vehicles Symposium (IV)* (IEEE, 2017), pp. 487–492
- [9] *Intel® realsense™ documentation*, [online] . available:, <https://dev.intelrealsense.com/docs/docs-get-started>, accessed: 2020-04-03
- [10] *Zxing: A barcode image processing library* [online] . available:, <https://opensource.google/projects/zxing>, accessed: 2020-12-03
- [11] *Qr code configuration*, [online] . available:, https://www.keyence.co.jp/ss/products/autoid/codereader/basic2d_qr.jsp, accessed: 2020-12-03
- [12] *Kinova™ ultra lightweight robotic arm, user guide* [online] . available:, <https://www.kinovarobotics.com/>, accessed: 2020-12-03
- [13] *Affine transformation to fully understand* [online] . available:, <https://qiita.com/>, accessed: 2020-12-03
- [14] T. KONNO, *The Journal of the Society for Art and Science* **5**, 80 (2006)
- [15] *Matlab pcmerge*, [online] . available:, <https://www.mathworks.com/help/vision/ref/pcmerge.html#bupmsru-1-ptCloudB>, accessed: 2020-04-03