TẠP CHÍ KHOA HỌC ĐẠI HỌC TÂN TRÀO
ISSN: 2354 - 1431
http://tckh.daihoctantrao.edu.vn/

AN EXAMPLE OF 3D RECONSTRUCTION ENVIRONMENT FROM RGB-D CAMERA

Trung-Minh Bui [1], Hai-Yen Tran[2], Thi-Loan Pham[3], Van-Hung Le[1,*]
[1]*Tan Trao University, Vietnam*
[2]*Vietnam Academy of Dance, Vietnam*
[3]*Hai Duong College, Vietnam*
[*]*Correspondence: Van-Hung Le (Van-hung.le@mica.edu.vn)*

https://doi.org/10.51453/2354-1431/2021/692

Article Info

Received: 12/10/2021
Accepted: 1/12/2021
Online:

Keywords:
3D environment reconstruction
RGB-D camera
Point cloud data

Abstract

3D environment reconstruction is a very important research direction in robotics and computer vision. This helps the robot to locate and find directions in a real environment or to help build support systems for the blind and visually impaired people. In this paper, we introduce a simple and real-time approach for 3D environment reconstruction from data obtained from cheap cameras. The implementation is detailed step by step and illustrated with source code. Simultaneously, cameras that support reconstructing 3D environments in this approach are also presented and introduced. The unorganized point cloud data is also presented and visualized in the available figures .

TẠP CHÍ KHOA HỌC ĐẠI HỌC TÂN TRÀO
ISSN: 2354 - 1431
http://tckh.daihoctantrao.edu.vn/

XÂY DỰNG LẠI MÔI TRƯỜNG 3D TỪ DỮ LIỆU THU ĐƯỢC CỦA CẢM BIẾN RGB-D

Bùi Trung Minh [1], Trần Hải Yến[2], Phạm Thị Loan[3], Lê Văn Hùng[1,*]
[1]*Đại học Tân Trào, Việt Nam*
[2]*Học viện Múa, Việt Nam*
[3]*Cao đẳng Hải Dương, Việt Nam*
[*]*Tác giả liên hệ:Lê Văn Hùng (Van-hung.le@mica.edu.vn)*

https://doi.org/10.51453/2354-1431/2021/692

Thông tin bài báo

Lịch sử:
Ngày nhận bài:
12/10/2021
Ngày duyệt đăng:
1/12/2021

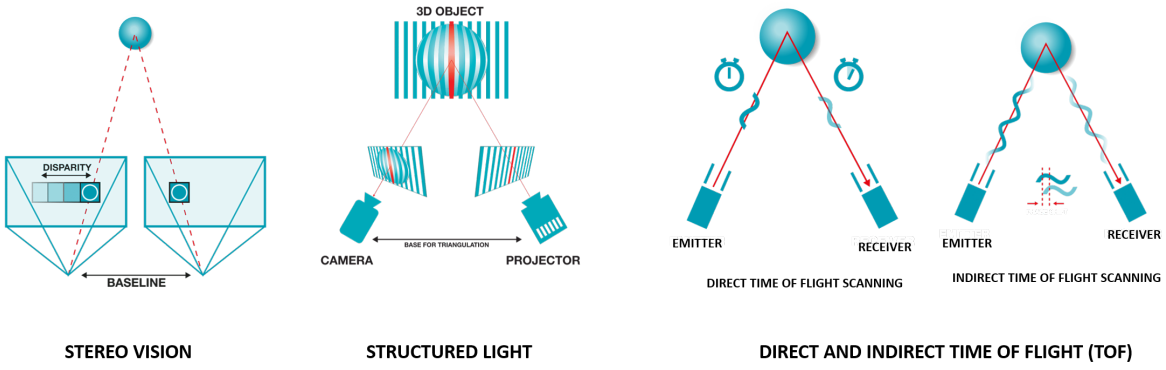
Từ khóa:
Dựng lại môi trường 3D
RGB-D camera
Đám mây điểm

Tóm tắt

Tái tạo môi trường 3D là một hướng nghiên cứu rất quan trọng trong lĩnh vực công nghệ Robot và thị giác máy tính. Hướng nghiên cứu này giúp Robot xác định vị trí và tìm đường đi trong môi trường thực tế hoặc giúp xây dựng hệ thống hỗ trợ dành cho người mù và người khiếm thị. Trong bài báo này, chúng tôi giới thiệu một cách tiếp cận đơn giản và được thực hiện trong thời gian thực để tái tạo môi trường 3D từ dữ liệu thu được từ cảm biến rẻ tiền. Quá trình thực hiện là được trình bày chi tiết từng bước và được minh họa bằng mã nguồn. Đồng thời, các loại cảm biến thu thập dữ liệu hình ảnh từ môi trường hỗ trợ tái tạo môi trường 3D theo cách tiếp cận này cũng được trình bày và giới thiệu. Dữ liệu được tạo ra là dữ liệu đám mây điểm không có cấu trúc cũng được trình bày và minh họa trong các số liệu có sẵn. Đồng thời các hình ảnh về môi trường cũng được thể hiện trực quan

1 Introduction

Reconstructing the 3D environment is a hot topic of research in computer vision. In particular, this problem is widely applied in robotics technology and the design of assisting systems for the blind and visually impaired people to move and interact with the environment in daily life. In the past, when computer hardware had many limitations, reconstruction of 3D environments often used a sequence of RGB images. In which the most used technique is the Simultaneous Localization And Mapping technique (SLAM) [1], [2], [3]. SLAM uses image information obtained from cameras to recreate the outside environment by putting environmental information into a map (2D or 3D), from which equipment (robots, cameras, vehicles) can locate. (localization) themselves. Its state and position in the map are to automatically set up the path (path planning) in the current environment.



Hình 1: Illustrating three kinds of technology of depth sensing [4].

However, with the fast advancement of computer hardware over the last decade, 3D reconstruction has become simple and precise. Particularly the development of 3D depth sensing technology. It enables devices and machines to sense and respond to their environment. Depth sensing enables the collection of data on depth measurement and three-dimensional perception, and it is classified into three categories: stereo vision, structured light, and time of flight (ToF).. Figure 1 illustrates three kinds of technology of depth-sensing [4], [5]. The most commonly used depth sensors today are shown in Tab. 1 [6].

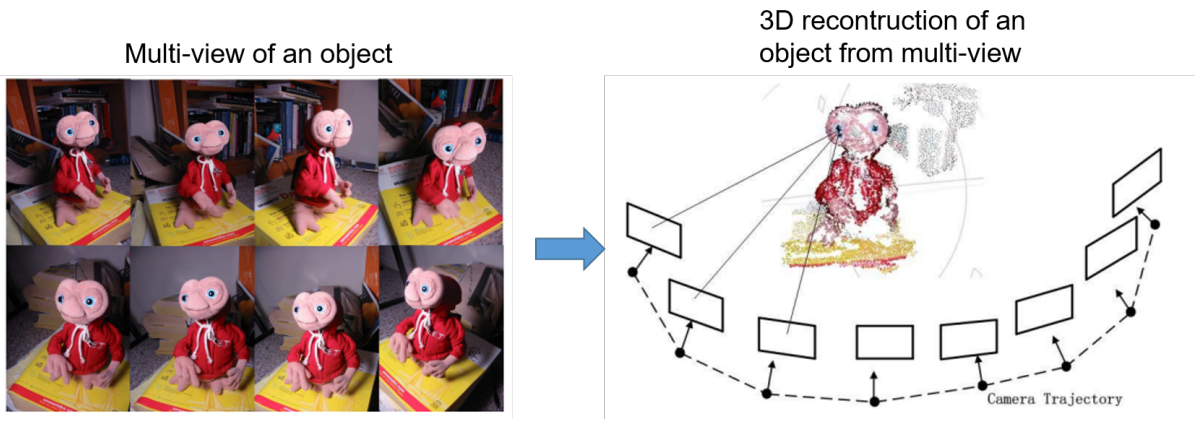
In this paper, we present an approach to reconstruct the 3D environment from the data obtained from the Microsoft (MS) Kinect v1. This is a cheap depth sensor and is frequently used in gaming and human-machine interaction. Simultaneously, integration with Windows becomes simple and straightforward. The environment's 3D data is accurately rebuilt and closely resembles the real one. Although Kramer et al. 's [7] tutorial has been studied in the past, the implementation process remains very abstract. Thus, we conduct and present our research in the form of steps to describe in detail the process of the installation, connection to the computer, data collection from the environment, reconstruction the 3D data of the environment, and some related problems.
The remaining of this paper will be presented as follows. In section 2, several related studies are presented; our method and experimental results analysis are described in section 3. Finally, the conclusion and some future ideas are presented in section 5.

2 Related Works

Simultaneous Localization and Mapping is a mapping and positioning technology that operates simultaneously. SLAM is used in a wide variety of automation control applications and was a prominent technology for recreating 3D environments from RGB picture sequences between 1985 and 2010. [8], [2], [9], [10]. Li et al. [11] have developed a meta-study of 3D environment reconstruction techniques and 3D object reconstruction with multiple approaches, in which the approach of using the SLAM technique to combine image sequences is important approach. Figure 2 illustrates the reconstruction of a 3D object from a sequence of images obtained from different views of the object. Davison et al. [12] proposed a MonoSLAM system for real-time localization and mapping with a single freely moving camera of mobile robotics. The MonoSLAM is a probabilistic feature-based map from a snapshot of the current estimates of the camera by the Extended Kalman Filter algorithm. The system is integrated and suitable for robot HRP-2 and has a processing capacity of 30Hz. Mitra et al. [13] computed the complexity and memory requirements required for the reconstruction of the 3D environment based on the number of cameras and the number of points on the point cloud data. Zhang et al. [14] proposed a motion estimation algorithm for strengthening based on a sliding window of images to process long image

Bảng 1: List of common depth sensors [6].

Camera name	Release date	Discontinued	Depth technology	Range	Max depth speed (fps)
Microsoft Kinect Version (V1)	2010	Yes	Structured light	500–4500 mm	30
Microsoft Kinect V2	2014	Yes	ToF	500–4500 mm	30
ASUS Xtion PRO LIVE	2012	Yes	Structured light	800–3500 mm	60
ASUS Xtion 2	2017	Yes	Structured light	800–3500 mm	30
Leap Motion (new 2018)	2013	No	Dual IR stereo vision	30–600 mm	200
Intel RealSense F200	2014	Yes	Structured light	200–1200 mm	60
Intel RealSense R200	2015	No	Structured light	500–3500 mm	60
Intel RealSense LR200	2016	Yes	Structured light	500–3500 mm	60
Intel RealSense SR300	2016	No	Structured light	300–2000 mm	30
Intel RealSense ZR300	2017	Yes	Structured light	500–3500 mm	60
Intel RealSense D415	2018	No	Structured light	160–10000 mm	90
Intel RealSense D435	2018	No	Structured light	110–10000 mm	90
SoftKinetic DS311	2011	Yes	ToF	150–4500 mm	60
SoftKinetic DS325	2012	Yes	ToF	150–1000 mm	60
SoftKinetic DS525	2013	Yes	ToF	150–1000 mm	60
SoftKinetic DS536A	2015	Yes	ToF	100–5000 mm	60
SoftKinetic DS541A	2016	Yes	ToF	100–5000 mm	60
Creative Interactive Gesture	2012	Yes	ToF	150–1000 mm	60
Structure Sensor (new 2018)	2013	No	Structured light	400–3500 mm	60



Hình 2: 3D object reconstruction from RGB image sequence [11].

sequences. This study reconstructed 3D environment from cubicle dataset (148 cameras, 31,910 3D points and 164,358 image observations) and outdoor dataset (308 cameras, 74,070 3D points and 316,696 image observations). Clemente et al. [15] used the EKF-SLAM algorithm to reconstruct the outdoor complex environment from the captured images. The Hierarchical Map technique is used in the algorithm to improve its robustness in dynamic and complex environments. The mapping process has been tested to run with a speed is at 30Hz with maps up to 60 point features. Strasdat et al. [16] proposed near real-time visual SLAM system for a 3D reconstruction environment, this method used the keyframe-based in the large images, the frames with different resolutions.

3 3D Environment Reconstruction from RGB-D Camera

3.1 RGB-D camera

From 2010 the present, several types of RGB-D sensors have been developed; these sensors are shown in Tab. 1. In this article, we only introduce the cheapest and most popular sensor, MS Kinect v1/ Xbox 360. Figure 3 illustrate the structure of MS Kinect v1/ Xbox 360. The components inside MS Kinect v1 include: RAM, a Prime Sense PS1080-A2 sensor, a cooling fan, a motorized Tilt, a three-axis accelerometer, four microphones (Multi - Array Mic)) and three cameras: RGB camera, depth sensor (3D Depth Sensors).

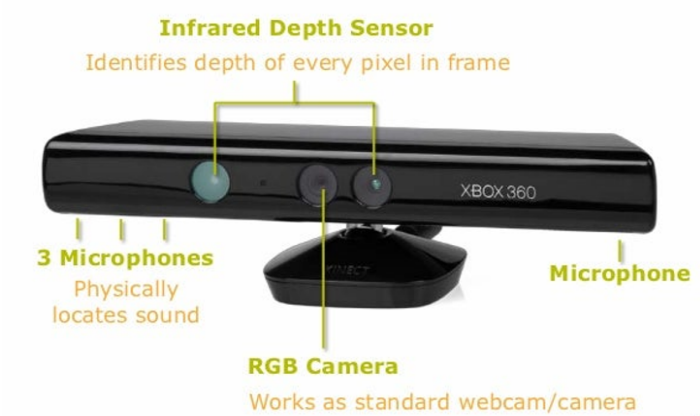
MS Kinect v1 is widely applied in gaming and human-machine interaction applications, so there are many libraries to support connecting to computers such as Libfreenect, Code Laboratories Kinect, OpenNI, and Kinect SDK.

3.2 Calibration

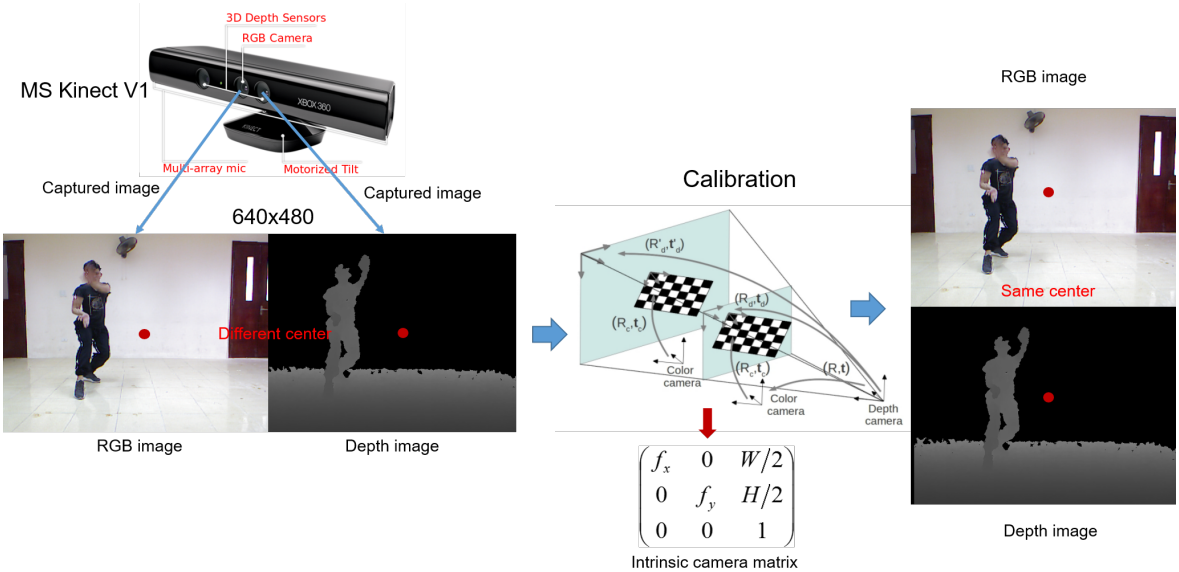
Ms Kinect v1 sensor captures data from the environment using the following methods: RGB sensors collects RGB pictures, infrared lamps projected infrared rays onto the surface of objects, and an infrared depth sensor acquired depth map data of the environment. Two sensors are not in the same position, there is a distance between them, as shown in Fig. 3. Therefore, to combine RGB and depth images into a coordinate, an image calibration procedure is required. Some researchers in the computer vision community proposed techniques for calibrating RGB and depth images collected from a MS Kinect sensor. There are many studies on this problem. The result of the calibration process is the camera's intrinsic matrix H_m for projecting pixels in 2-D space to 3-D space.

It is illustrated in Fig. 4.

Where the calibration process is the process of finding the calibration matrix, which has the form



Hình 3: The structure of the MS Kinect v1 sensor.



Hình 4: Camera calibration model of MS Kinect v1.

of the Eq. 1.

$$H_m = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \quad (1)$$

where (c_x, c_y) is the principle point (usually the image center), f_x and f_y are the focal lengths. The result of this process is that the color and depth image are corrected to the same center by the calibration matrix, as shown in Fig. 4. In figure 4 and equation 1, $c_x = \frac{W}{2}; c_y = \frac{H}{2}$, where W is the width of the image and H is the height of the image.

In Nicolas et al.'s research [17], the matrix H_m is published as Eq. 2.

$$H_m = \begin{bmatrix} 594.214 & 0 & 339.307 \\ 0 & 591.040 & 242.739 \\ 0 & 0 & 1 \end{bmatrix} \quad (2)$$

In Jason et al.'s research [18], the intrinsic parameters of RGB camera is computed and published as Eq. 3.

$$H_m = \begin{bmatrix} 589.322 & 0 & 321.1408 \\ 0 & 589.849 & 235.563 \\ 0 & 0 & 1 \end{bmatrix} \quad (3)$$

The intrinsic parameters of depth camera [18] is computed and published as Eq. 4.

$$H_m = \begin{bmatrix} 458.455 & 0 & 343.645 \\ 0 & 458.199 & 229.8059 \\ 0 & 0 & 1 \end{bmatrix} \quad (4)$$

3.3 Point Cloud Data

We re-introduce the definition of point cloud data "Point clouds are datasets that represent objects or space. These points represent the X, Y, and Z geometric coordinates of a single point on an underlying sampled surface. Point clouds are a means of collating a large number of single spatial measurements into a dataset that can then represent a whole. When colour information is present, the point cloud becomes 4D." [19].

The point cloud data is divided into two types: organized point cloud data and unorganized point cloud data [7]. The organized point cloud data is organized points like an image, the image that makes up the point cloud is $(W \times H)$ pixels then the organized point cloud data also has the size of $(W \times H)$ points and sort by rows and columns of the matrix, as illustrated in Fig. 5(top-right). The unorganized point cloud data is organized by the size of $(W \times H)$ points, the matrix that sorts the points is $1 \times (W \times H)$, as illustrated in Fig. 5(bottom-right).

The process of converting to point cloud data is done [17]. Each 3D point (P_3D) is created from a pixel with coordinates (x, y) on the depth image and a corresponding pixel on the color image that has a color value $C(r, g, b)$. P_3D includes the following information: coordinates (P_3D_x, P_3D_y, P_3D_z) in 3D space, the color value of that point (P_3D_r, P_3D_g, P_3D_b), where the depth value (D_v) of point $P(x, y)$ must be greater than 0. P_3D RGB (a color point) is computed according to Eq. 5, P_3D (a no color point) is computed according to Eq. 6.

$$\begin{aligned} P\_3D_x &= \frac{(x - c_x) * D}{f_x} \\ P\_3D_y &= \frac{(y - c_y) * D}{f_y} \\ P\_3D_z &= D_v \\ P\_3D_r &= C_r \\ P\_3D_g &= C_g \\ P\_3D_b &= C_b \end{aligned} \quad (5)$$

$$\begin{aligned} P\_3D_x &= \frac{(x - c_x) * D_v}{f_x} \\ P\_3D_y &= \frac{(y - c_y) * D_v}{f_y} \\ P\_3D_z &= D_v \end{aligned} \quad (6)$$

where (f_x, f_y—focal length), (c_x, c_y—center of the images) are intrinsics of the depth camera.

To inverse project a point point (P_3D) of the cloud data to a pixel (P_2D_{rgb}) of the image data (3D to 2D space), the formula (7) is used.

$$\begin{aligned} P\_2D_{rgb}.x &= \frac{(P\_3D.x * f_x)}{P\_3D.z} + c_x \\ P\_2D_{rgb}.y &= \frac{(P\_3D.y * f_y)}{P\_3D.z} + c_y \end{aligned} \quad (7)$$

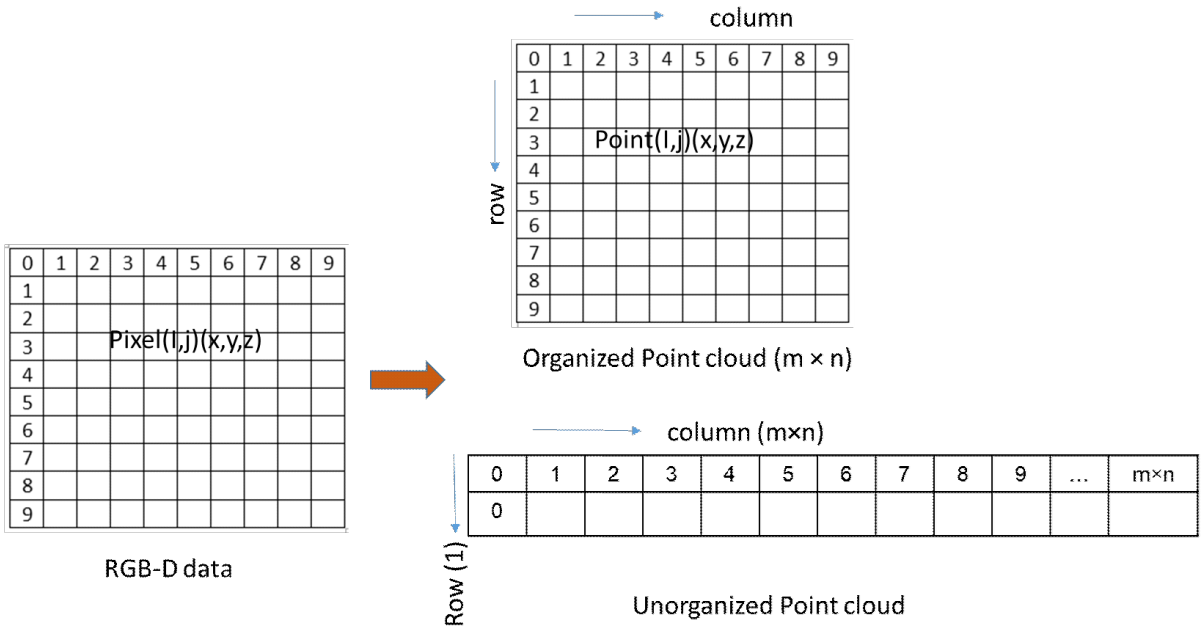
Figure 6 illustrates the result of color point cloud data generated from color data and depth data obtained from MS Kinect v1.

4 Experiment Results

4.1 Setup and Data collection

To collect data from the environment and objects, it is necessary to connect the RGB-D sensor to the computer. In this paper, we use MS Kinect v1 to connect to the computer by the USB port, as illustrated in Fig. 7.

To perform the connection and control, we use the Kinect for Windows SDK v1.8 (https://www.microsoft.com/en-us/download/details.aspx?id=40278 [accessed on 18 Dec 2021]) and the Kinect for Windows Developer Toolkit v1.8 (https://www.microsoft.com/

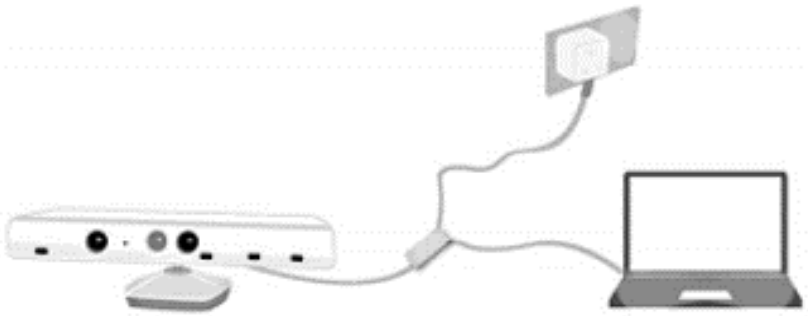


Hình 5: Two types of the point cloud data.

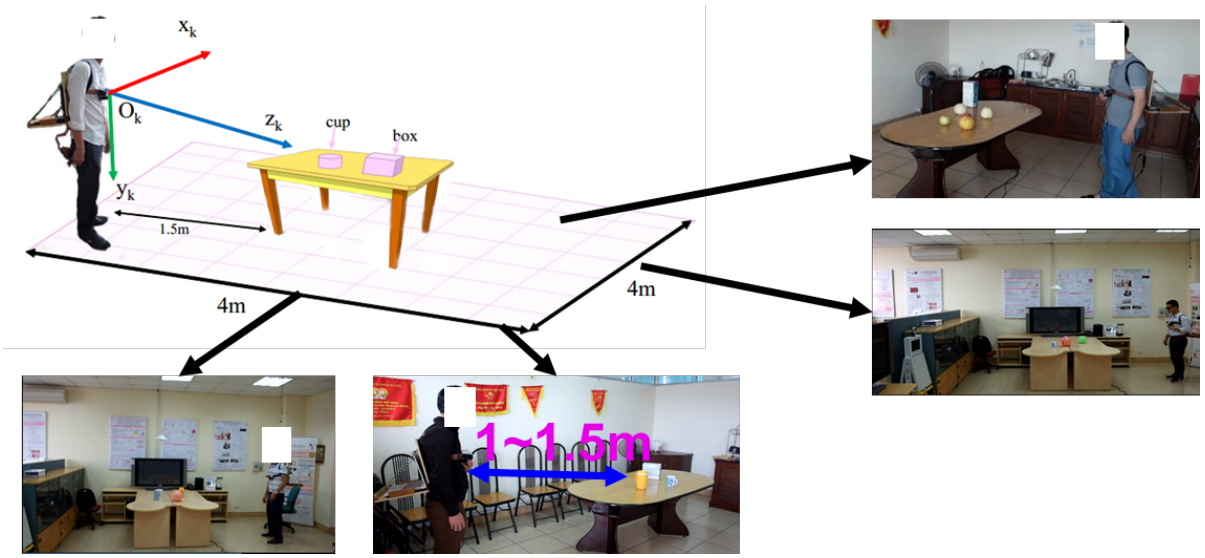
Hình 6: Camera calibration model of MS Kinect v1.

en-us/download/details.aspx?id=40276 [accessed on 18 Dec 2021]). Two libraries of MS Kinect v1 are standardized connected to Windows 7 operating system. The devices are set up as shown in Fig. 8. In figure 8, MS Kinect v1 is mounted on a person's chest, Laptop is worn on the person's back, we conduct our experiments on a Laptop with a CPU Core i5 processor (2540M) - RAM 8G. The collected data is the color image and depth image of the table, objects on the table, environment around the table in the receiving range of MS Kinect v1 (0.5-4.5m). The captured image has a resolution of 640 × 480 pixels.

The C++ programming language, the OpenCV 2.4.9 library (https://opencv.org/ [accessed on 18 Nov 2021]), and the PCL 1.7.1 library (https://pointclouds.org/ [accessed on 18 Nov 2021]), and Visual studio 2010 (https://visualstudio.microsoft.com/fr/ [accessed on 18 Nov 2021]) are used to develop the program to connect, calibration images, generate point cloud data. In addition, the program also supports a number of other libraries in PCL such as Boost (https://www.boost.org/ [accessed on 18 Nov 2021]), VTK (https://vtk.org/ [accessed on 18 Nov 2021]), OpenNI (https://code.google.com/archive/p/simple-openni/ [accessed on 18



Hình 7: The connection of MS Kinect v1 and the computer.



Hình 8: Environment and the collection data.

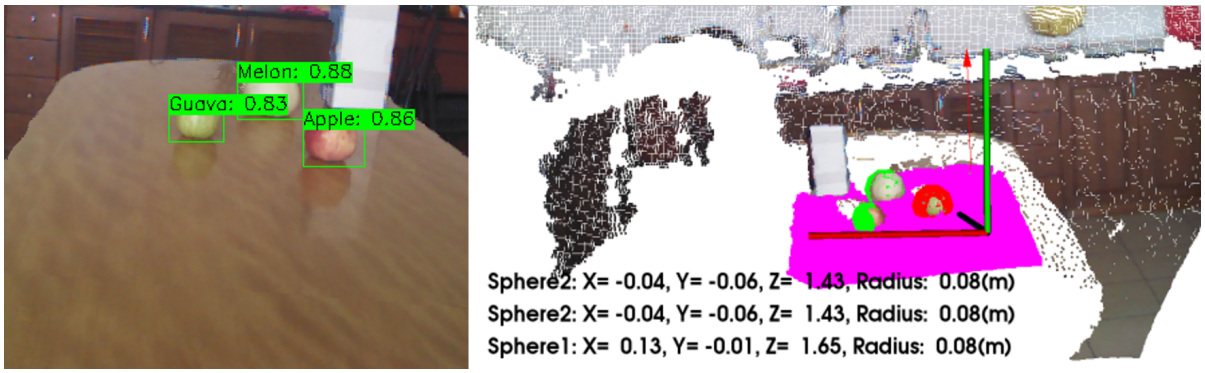
Nov 2021]), etc. All the source code we share in the link (https://drive.google.com/file/d/1KfXrGTDXGDxraMI9Cru4KrmBVOClLnrC/view?usp=sharing [accessed on 18 Nov 2021]).

4.2 Results and Discussions

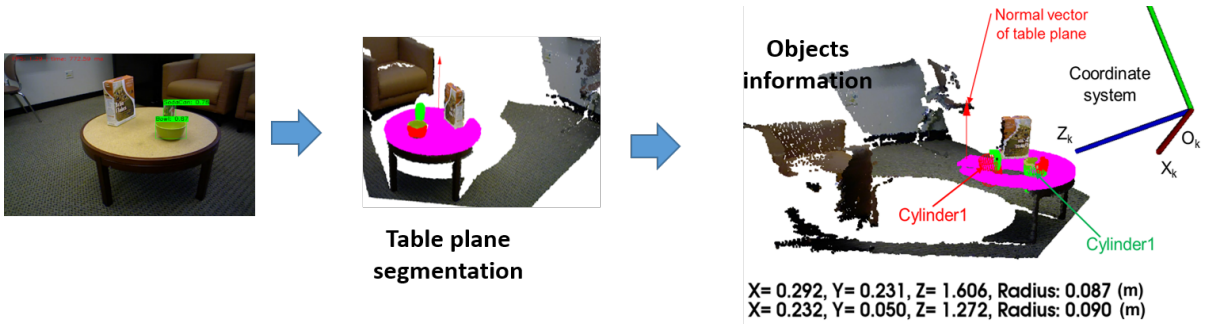
The point cloud data we generated is unorganized color point cloud data, which is 640 points, and included a lot of points with coordinates of (x=0,y=0,z=0). This problem occurs when objects, surfaces are outside the measuring range of MS Kinect v1 or their surface is the black color or their surface is glossy, so it absorbs infrared light from MS Kinect v1. Therefore, the depth value at these pixels is 0. Figure 9 illustrates some point cloud data obtained from point cloud acquisition and creation from the MS Kinect v1 sensor. Once point cloud data is generated, many issues need to be studied on this data. Like object segmentation problem on point cloud data, 3D object recognition detection problem needs to be studied, as illustrated in Fig. 10. The color point cloud data acquisition and data generation rate is 3 fps.

5 Conclusions and Future Works

Reconstructing a 3D environment from sensor/camera data is a classic computer vision research topic. It is very extensively adopted in robotics, industry, and self-driving cars. In this paper, we have detailed the setup, data collection,



Hình 9: The color point cloud data generated from RGB and depth image of MS Kinect v1.



Hình 10: Table and objects segmentation on the point cloud data problem.

and point cloud generation from the MS Kinect v1 sensor, especially the steps of setting up, editing images, creating point cloud data are presented uniformly. The point cloud data generated from image data obtained from MS Kinect is 640×480 points, speed generation is 3 fps. This project will result in the development and publication of papers and tutorials on RGB-D sensors. In the near future, we will also conduct further studies on object recognition in point cloud data, especially using convolutional neural networks for 3D object recognition.

REFERENCES

[1] W. B. Gross, "Combined effects of deoxycorticosterone and furaltadone on Escherichia coli infection in chickens," American Journal of Veterinary Research, **45**(5), 963–966, 1984.

[2] H. Durrant-Whyte, T. Bailey, "Simultaneous localization and mapping: Part I," IEEE Robotics and Automation Magazine, **13**(2), 99–108, 2006, doi:10.1109/MRA.2006.1638022.

[3] P. Skrzypczyński, "Simultaneous localization and mapping: A feature-based probabilistic approach," International Journal of Applied Mathematics and Computer Science, **19**(4), 575–588, 2009, doi:10.2478/v10006-009-0045-z.

[4] "Depth Sensing Technologies," https://www.framos.com/en/products-solutions/3d-depth-sensing/depth-sensing-technologies, 2021, [Accessed 20 Nov 2021].

[5] "Depth Sensing Overview," https://www.stereolabs.com/docs/depth-sensing/, 2021, [Accessed 20 Nov 2021].

[6] R. Li, Z. Liu, J. Tan, "A survey on 3D hand pose estimation: Cameras, methods, and datasets," Pattern Recognition, **93**, 251–272, 2019, doi:10.1016/j.patcog.2019.04.026.

[7] J. Kramer, N. Burrus, F. Echtler, H. C. Daniel, M. Parker, Hacking the Kinect, 2012, doi:10.1007/978-1-4302-3868-3.

[8] R. Chatila, J. P. Laumond, "Position referencing and consistent world modeling for mobile robots," in Proceedings - IEEE International Conference on

Robotics and Automation, 138–145, 1985, doi:10.1109/ROBOT.1985.1087373.

[9] T. Bailey, H. Durrant-Whyte, "Simultaneous localization and mapping (SLAM): Part II," IEEE Robotics and Automation Magazine, **13**(3), 108–117, 2006, doi:10.1109/MRA.2006.1678144.

[10] J. Aulinas, Y. Petillot, J. Salvi, X. Lladó, "The SLAM problem: A survey," in Frontiers in Artificial Intelligence and Applications, volume 184, 363–371, 2008, doi:10.3233/978-1-58603-925-7-363.

[11] L. Ling, PHD Thesis - Dense Real-time 3D Reconstruction from Multiple Images, Ph.D. thesis, 2013.

[12] A. J. Davison, I. D. Reid, N. D. Molton, O. Stasse, "MonoSLAM: Real-Time Single Camera SLAM," IEEE Transactions on pattern analysis and machine intelligence, **29**(6), 2007.

[13] K. Mitra, R. Chellappa, "A scalable projective bundle adjustment algorithm using the L norm," in Proceedings - 6th Indian Conference on Computer Vision, Graphics and Image Processing, ICVGIP 2008, 79–86, 2008, doi:10.1109/ICVGIP.2008.51.

[14] Z. Zhang, Y. Shan, "Incremental Motion Estimation Through Local Bundle Adjustment," Technical Report MSR-TR-2001-54, 2001.

[15] L. A. Clemente, A. J. Davison, I. D. Reid, J. Neira, J. D. Tardós, "Mapping large loops with a single hand-held camera," in Robotics: Science and Systems, volume 3, 297–304, 2008, doi:10.15607/rss.2007.iii.038.

[16] H. Strasdat, J. M. Montiel, A. J. Davison, "Scale drift-aware large scale monocular SLAM," in Robotics: Science and Systems, volume 6, 73–80, 2011, doi:10.7551/mitpress/9123.003.0014.

[17] B. Nicolas, "Calibrating the depth and color camera," http://nicolas.burrus.name/index.php/Research/KinectCalibration, 2018, [Online; accessed 10-January-2018].

[18] C. Jason, "Kinect V1 Rgb and Depth Camera Calibration," https://jasonchu1313.github.io/2017/10/01/kinect-calibration/, 2017, [Online; accessed 10-Nov-2021].

[19] C. Thomson, "What are point clouds? 5 easy facts that explain point clouds," https://info.vercator.com/blog/what-are-point-clouds-5-easy-\facts-that-explain-point-clouds, 2019, [Online; accessed 10-Nov-2021].