Study on the Use of Q-R Codes as Landmarks for Indoor Positioning: Preliminary Results

Zheqi Li
Electrical Engineering Department
California State University, Fullerton
Fullerton, USA
novusli@csu.fullerton.edu

Jidong Huang
Electrical Engineering Department
California State University, Fullerton
Fullerton, USA
jhuang@fullerton.edu

Abstract— **Indoor positioning, as often needed by mobile robots operating indoor and in other indoor navigation applications, usually has to be obtained without access to GNSS (Global Navigation Satellite Systems) signals. Many indoor localization techniques, such as RF-beacon-based ones and SLAM (Simultaneous Localization and Mapping) techniques, are either susceptible to tampering, environmental noise, or are costly. Alternatively, indoor landmarks, either permanent ones such as doors, windows and light fixtures or human-introduced ones such as pictures and QR codes, can be identified through image processing techniques and used to determine a user's location based on known landmark locations.**

This paper explored how to use the high-definition RGB camera and the depth sensor of the Kinect to detect the QR-code-based landmarks and measure the distance to the QR codes, so that it can assist a robot or other user to navigate in indoor environment. The RGB camera of the Kinect sensor was used to capture the images of the QR codes attached on the wall in different locations in the room. Each one of this QR codes has unique identifier and was coded to contain the indoor reference location information. Then the ZBar open-source barcode reading algorithm was applied to decode the QR codes and extract the reference coordinate information within the QR codes. After that, the depth sensor of the Kinect was initiated to measure the distance from the Kinect sensor to these QR codes. These raw data collected from the RGB camera and the depth sensor were calibrated and then used to locate the position of the Kinect within the indoor coordinate frame. In this paper, the preliminary results from testing the algorithm to locate the stationary position of the Kinect sensor using QR codes were presented.

Keywords—Indoor Positioning, Kinect, QR codes, Depth Sensing.

I. Introduction

Indoor navigation is of great importance to unmanned vehicles and robots. Since the GNSS (Global Navigation Satellite System) reception is normally unavailable inside buildings, a variety of approaches to indoor positioning, such as RF-beacon-based ones and SLAM (Simultaneous Localization and Mapping) techniques are developed. However, many of these techniques are either susceptible to tampering, environmental noise, or are costly to implement. Alternatively, indoor landmarks, either permanent ones such as doors, windows and light fixtures or human-introduced ones such as pictures and QR codes, can be identified through image processing techniques and used to determine a user's location based on known landmark locations. In this paper, an approach using Kinect sensor reading the QR codes attached on the wall to position in indoor environment is presented.

Kinect is a motion sensing device that includes a high-definition RGB camera, a depth sensor and a multi-array microphone. The RGB camera of the Kinect can be used to capture the images of the QR codes attached on the wall and the depth sensor measures the distance between the Kinect camera and the QR codes.

QR code has been widely used for manufacturing, business, etc. due to its capability of storing great capacity of information in one code. It is cheap and easy to generate. Using QR codes to assist indoor navigation would be stable, easy-to-use and cheap, thus having great potential to be used widely in indoor positioning and localization applications.

In this paper, we explored the feasibility and accuracy of using the Kinect sensor to read QR codes for indoor positioning. Variable software and libraries were used to develop the application and the results were demonstrated at the end.

II. Kinect Specifications and Development Tools

A. Introduction to the Kinect

There are two generations of Kinect sensors developed by Microsoft. The Kinect v2 has a significant improvement compared with the Kinect v1. Taking into consideration of the compatibility of the Kinect to the applications and the accuracy of the experiment, we decided to use the Kinect v2 as our experimental apparatus.

The Kinect v2, as shown in Fig. 1 [1], includes a high-definition RGB camera and a depth sensor. The RGB camera can capture high-definition image and the video captured in high-definition can be displayed on the screen in the same resolution. The video communication has been improving due to the fast data streaming. The depth sensor of the Kinect captures the IR light dots emitted by the IR emitters to detect the depth of the objects. Using the RGB camera and the depth sensor of the Kinect to capture and decode the QR codes and detect the distance between the QR codes and the Kinect sensor, we can obtain the raw data measured by the Kinect to calculate the exact position of the Kinect sensor.

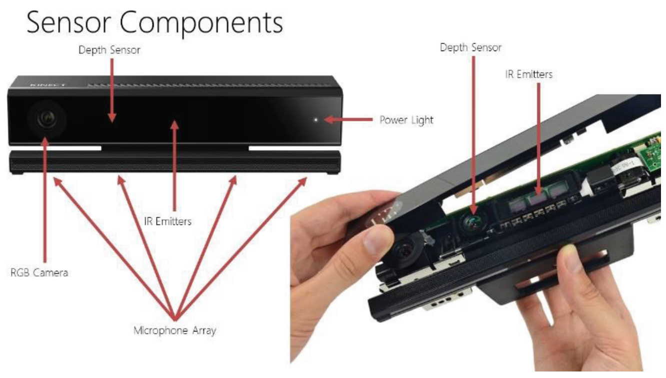

Fig. 1. Kinect v2 sensor components.

B. Specifications of the Kinect sensor

The TABLE I [2] below shows the differences between the Kinect v1 and the Kinect v2.

TABLE I. SUMMARY AND COMPARISON OF THE KINECT V1 AND KINECT V2

Feature	**Kinect for Windows v1**	**Kinect for Windows v2**
Color Camera	640 x 480 @30fps	1920 x 1080 @30fps
Depth Camera	320 x 240	512 x 424
Max Depth Distance	4.5 M	8 M
Min Depth Distance	40 cm in near mode	50 cm
Depth Horizontal Field of View	57 degrees	70 degrees
Depth Vertical Field of View	43 degrees	60 degrees
Tilt Motor	Yes	No
Skeleton Joints Defined	20 joints	25 joints
Full Skeletons Tracked	2	6
USB Standard	2.0	3.0
Supported OS	Win 7, Win 8	Win 8, Win 8.1, Win 10

Taking into consideration of the compatibility of the Kinect sensor to the applications and the accuracy of the experiment, we decided to use the Kinect v2 as our experimental apparatus. With a high-definition camera, capturing a clear image would be much more efficient using Kinect v2.

C. Software and Libraries

There are various open source development kits for the Kinect sensor, such as LibFreenet2 [3], OpenKinect [4], Processing [5], etc. In this project, we used Microsoft Visual Studio [6] to program the Kinect sensor due to its high compatibility to various libraries. Visual Studio is an integrated development environment (IDE) on Windows. It can be used to develop various applications and it supports 36 different programming languages, such as C/C++, C#, VB.NET, HTML/XHTML, JavaScript, etc. [7]

Since the images captured by the color sensor need to be processed, we used an image processing library named Open Source Computer Vision Library (OpenCV) [8]. OpenCV is a library of programming functions that focus on computer image processing. OpenCV is based on C++, with additional bindings in Python, Java and MATLAB [9]. This library is cross-platform and free to use. In order to find the QR codes within the images taken by the Kinect RGB camera, we use the OpenCV to scan the contours within the image to find out and highlight the QR codes so that we can see clearly if the camera has captured the QR codes.

Zbar [10] is an open-source C-based barcode reading library. It can be used to read any barcodes from various sources, such as image, video, etc. It supports real-time scanning of video streams and can recognize EAN-13, UPC-A, UPC-E, EAN-8, Code 128, Code 39, Interleaved 2 of 5 and QR code symbologies[11]. In this study, ZBar library is applied to decode the QR codes within the image taken by the color sensor.

Kinect for Windows SDK is a non-commercial Kinect software development kit (SDK) for Windows users by Microsoft. The SDK includes Windows compatible PC drivers for Kinect devices and it supports C++, C# and Visual Basic in Microsoft Visual Studio [12]. This is a required software for Windows users to develop applications or program for the Kinect device.

III. The Methodology and Algorithm

Trilateration-based positioning was used in this paper to determine the exact position of the Kinect sensor by collecting the depth data from the Kinect sensor to the QR codes and the decoded coordinate information from the QR codes. In order to obtain the 3D position of the Kinect sensor, at least 3 QR codes need to be tracked.

Assuming there are three readings from the QR codes and that the (x_i, y_i, z_i), where i = 1, 2, 3 are the exact positions of the QR codes and the d_i, where i = 1, 2, 3 are the distance between the Kinect sensor to the QR codes, the system equations are:

$$\sqrt{(x-x_1)^2 + (y-y_1)^2 + (z-z_1)^2} = d_1$$
$$\sqrt{(x-x_2)^2 + (y-y_2)^2 + (z-z_2)^2} = d_2$$
$$\sqrt{(x-x_3)^2 + (y-y_3)^2 + (z-z_3)^2} = d_3$$

By solving this equation set, the position of the Kinect sensor (x, y, z) will be located. To solve this equation set, the Newton- Raphson method was applied and used to calculate the position of the Kinect sensor.

IV. The Experimental Results

The experiments conducted in this study shows that using the Kinect sensor for in indoor positioning can be efficient and economical. The experiments have been conducted as follows:

A. QR Code Detection with Kinect Color Sensor

B. Indoor Depth Sensor Measurement

C. Calibration of the RGB camera and the depth sensor

D. QR Codes Decoding

E. Positioning Calculation

F. Error Analysis

A. QR Code Detection with Kinect Color Sensor

Experiments have been conducted to detect the QR codes with the color sensor of the Kinect. The Fig. 2 shows that the image taken by the color sensor is a mirrored image of a QR code, therefore it needs to be further processed using OpenCV.

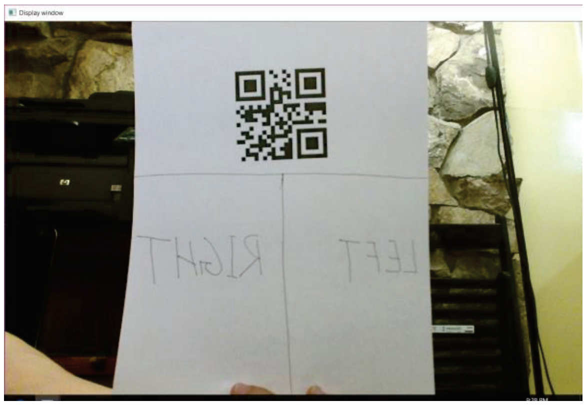

Fig. 2. The mirrored image taken by the Kinect color sensor.

Since the image taken by the camera of the Kinect is a mirrored image, OpenCV was applied to process the image so that the QR code within the image can be read correctly. OpenCV has provided a lot of programming functions to process the image. In this case, we used the function “flip” to flip this image around the y-axis to make it into a normal image, as shown below in Fig. 3.

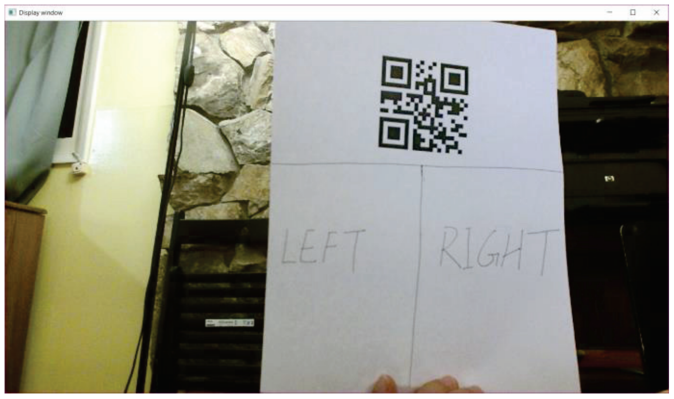

Fig. 3. The flipped image using the OpenCV library.

In order to find the region of interest within the image for further processing, we converted the image into grayscale using OpenCV as shown in Fig. 4.

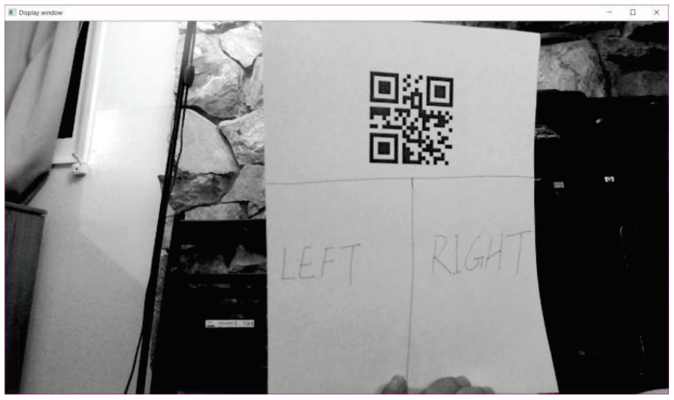

Fig. 4. The grayscale image.

After the image was converted into grayscale, the canny algorithm was applied to further process the image to extract useful structural information and reduce the amount of data to be processed. The image processed using canny algorithm was shown in Fig. 5.

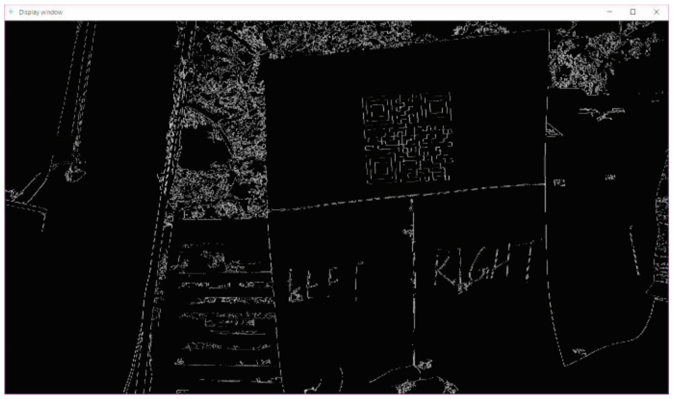

Fig. 5. The image processed using the canny algorithm.

Next, we have processed the image with contour hierarchy and highlight the position detection pattern of the QR codes in the image. In order to test if multiple QR codes can be detected at the same time within a single image, four QR codes were printed out on a single paper. The experiment conducted shows that the position detection patterns of the four QR codes have been detected and highlighted as shown in the Fig. 6.

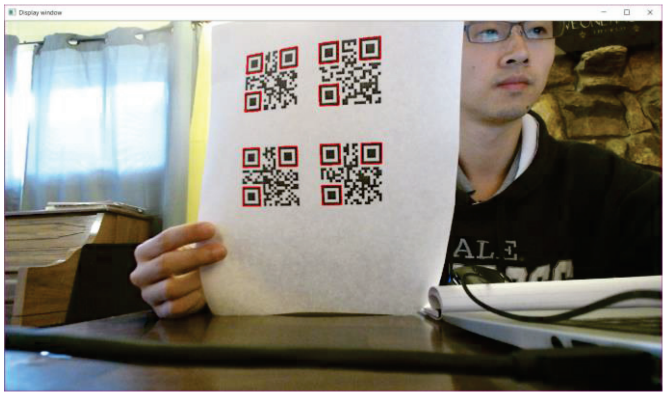

Fig. 6. Four QR codes in a paper.

B. Indoor Depth Sensor Measurement

The next experiment was conducted to test the accuracy of the Kinect depth sensor. In order to make the experiment more precise, a laser distance meter was used to analyze the accuracy of Kinect depth measurements. The results were shown below in the Fig. 7.

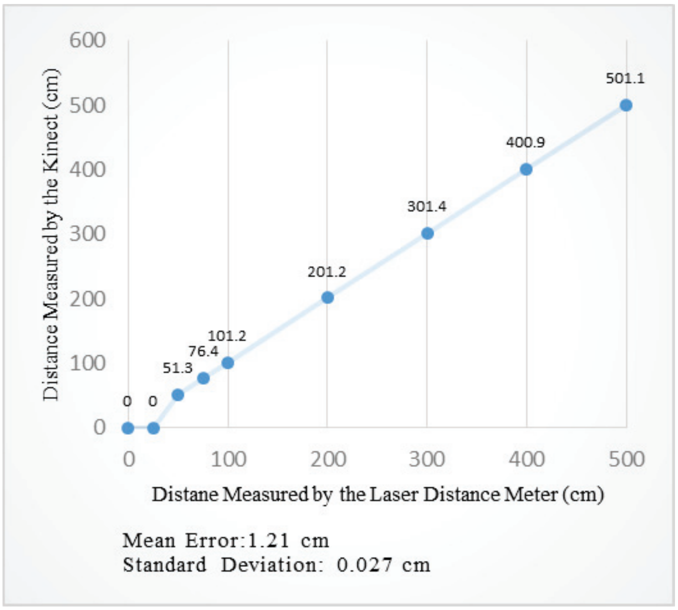

Fig. 7. The indoor distance measurement.

We realized that the indoor environment distance measured by the depth sensor has a difference of about 1 centimeter from the laser measurement. This could be caused by sensor misalignment; the lighting condition of the environment, etc. Also, the objects with different color have impacts on the measurement accuracy. The reason could be objects with different color have different reflectance for infrared waves.

C. Calibration of the RGB Camera and the Depth Sensor

We have noticed that the RGB camera and the depth sensor are located in different parts of the Kinect sensor. So when we tried to measure the distance to a certain point of the image captured by the RGB camera, it was not the same place in the depth image as it was in the color image. The measured error becomes larger as the measured distance reaches further. Additionally, the RGB camera of the Kinect is a high-definition camera, the color image has a wider view of objects than the depth image horizontally. On the other hand, the depth image has a wider view vertically than the color image. In order to minimize errors in depth measurements, we calibrated the color image and the depth image by trimming the extra parts and make these two images match accordingly, as shown in Fig. 8.

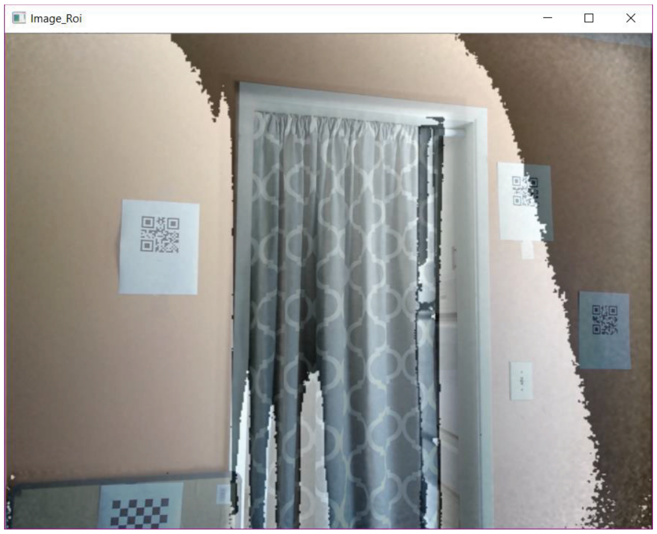

Fig. 8. Calibrated color image and depth image.

D. QR Code Decoding

When the position detection patterns of a QR code have been recognized, we can use ZBar library to decode the QR code and extract the information from it, which in this case is the indoor coordinates of the QR code. Using ZBar library to decode multiple QR codes can be quick and accurate. After the QR codes were detected, the ZBar library was applied to decode the QR codes and output the results on the screen. We can see from the TABLE II below that the coordinates information encrypted in the QR codes have been decoded and extracted.

TABLE II. THE DECODED INFORMATION

QR code	QR code decoded Msg X	QR code decoded Msg Y	QR code decoded Msg Z
#1 QR code	0	1200	700
#2 QR code	200	600	700
#3 QR code	600	600	1000
#4 QR code	700	600	300

E. Positioning Calculation

Experiments were conducted to capture and decode the QR codes attached on the wall (as shown in Fig. 9) and extract the coordinate information of the QR codes to calculate the position of the Kinect sensor. The image below shows that multiple QR codes were detected from the Kinect image and used to obtain the indoor coordinates of these QR-based landmarks.

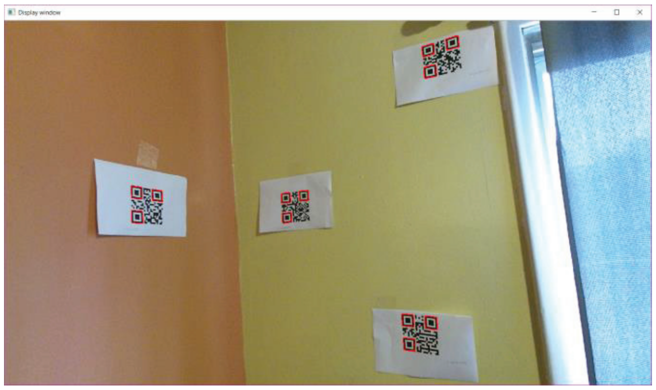

Fig. 9. Four QR codes detected by the Kinect sensor.

After the QR codes have been detected and decoded, the coordinates of these QR codes were used to calculate the position of Kinect sensor through trilateration. The iterative Newton – Raphson method was applied in Visual Studio 2015 to solve the non-linear multivariable system of equations. The TABLE III below shows the coordinates of the four QR codes and the distance between each QR code and the Kinect sensor. The bottom result of the image shows the coordinate of the Kinect sensor.

TABLE III. THE DECODED INFORMATION AND THE COORDINATE OF THE KINECT SENSOR

QR code	X	Y	Z	The distance between the QR code and the Kinect sensor (mm)
#1 QR code	200	400	500	882
#2 QR code	600	400	200	895
#3 QR code	500	400	800	830
#4 QR code	0	700	500	862
The coordinate of the Kinect sensor	478.989	770.76	435.299	

F. Error Analysis

In order to test the accuracy of the experiment, we've conducted some experiments to calculate the errors and standard deviation of the indoor positioning results from Kinect depth sensor. From the following experiments, we collected 100 groups of the depth data, then compared with the actual distance measured by a laser distance meter and obtained the mean and the standard deviation from the data. Fig. 10 shows the depth data of a QR code. The TABLE IV shows the mean error and standard deviation of the distance between the depth sensor and the QR code when it was compared with the reading from the laser measurement.

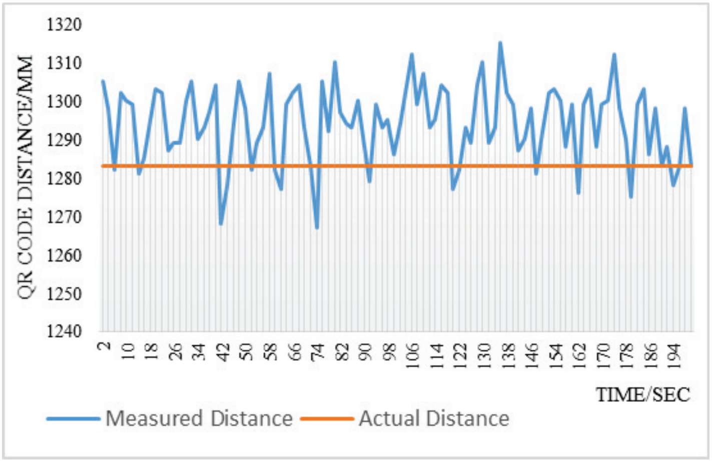

Fig. 10. Depth data collected by Kinect sensor over time.

TABLE IV. MEAN ERROR AND STANDARD DEVIATION OF THE RESULT

	Actual Distance (mm)	Mean Depth (mm)	Mean Error (mm)	Standard Deviation (mm)
QR code	1283	1293.98	10.98	11.00

After we have tested the accuracy of the depth sensor, we have conducted another experiment to see how the errors change by measuring multiple QR codes at the same time to calculate the exact location of the Kinect sensor.

The Kinect sensor was firstly set up in a fixed location with a certain coordinate measured by a laser distance meter and there were four QR codes attached on the wall in front of it. The Kinect sensor kept collecting the depth data of the four QR codes and calculating the exact position of itself and compared the calculated position with the actual fixed coordinates to obtain the difference. The Fig. 11 shows the mean of the xyz coordinates and the changes of the 100 groups of xyz data calculated.Then the mean errors and standard deviation of the differences were shown in the TABLE V.

The TABLE V shows the mean and standard deviation of the errors of 100 groups of data collected. The mean of the x-coordinate is relatively large, which might be caused by the distortion of the QR code image taken by the Kinect color sensor since there is always at least one of the QR codes having a certain angle towards the Kinect sensor.

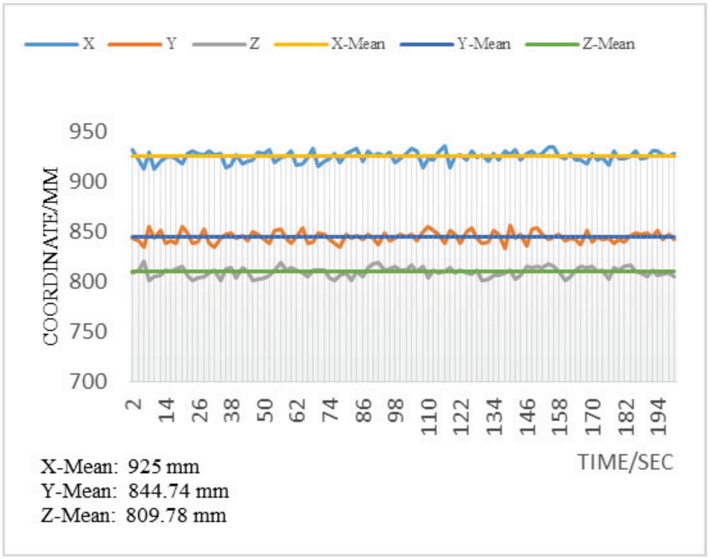

Fig. 11. 100 groups of kinect position solutions over time.

TABLE V. MEAN ERROR AND STANDARD DEVIATION OF THE XYZ – COORDINATE

Coordinate	Mean Error (mm)	Standard deviation (mm)
X	20.9099	8.1914
Y	-9.2563	8.1940
Z	0.7776	5.5730

V. CONCLUSION AND FUTURE STUDY

From the experiments conducted, we see that the high-definition RGB camera and the depth sensor of the Kinect can be used to detect the QR-code-based landmarks and measure the distance to the QR codes, so that it can assist a robot or other user to navigate in indoor environment. We realized that the measuring angle of the Kinect depth sensor had certain impact on the accuracy of depth data. The closer to the central region of the depth sensor, the more accurate data were collected. This could be caused by the attenuation of the infrared transmission since the central region of the depth image has shorter distance to the depth sensor than the four corners.

Secondly, even we have calibrated the RGB camera and the depth sensor, there are still errors of the collected data due to the different location of the RGB camera and the depth sensor in the Kinect device.

Nevertheless, the Kinect v2 is a very potential and comprehensive motion sensor. With the RGB camera and the depth sensor, it also can be applied to scan the environment and draw a 3D color image of the environment. The depth sensor can also help a user such as robots for indoor positioning and obstacle avoidance. The body tracking, facial recognition and gesture recognition applications will make robot capable of communicating with people easily. More applications can be developed from the Kinect sensor to be a useful sensor for robots or unmanned vehicles working indoors.

References

[1] B. Lower,et al., "Programming Kinect for Windows v2 jump start[PowerPoint slides]," retrieved from http://brightguo.com/k2-dev-video/, 2014.

[2] J. Ashley, "Quick reference: Kinect 1 vs Kinect 2," retrieved from http://www.imaginativeuniversal.com/blog/2014/03/05/Quick-Reference-Kinect-1-vs-Kinect-2/, 2014.

[3] "Libfreenect2 0.2", online, accessed Dec 2017, retrieved from https://openkinect.github.io/libfreenect2/

[4] "OpenKinect", online, accessed April 2017, retrieved from https://openkinect.org/wiki/Main_Page

[5] "Processing", online, accessed April 2017, retrieved from https://processing.org/

[6] "Microsoft", online, accessed April 2017, retrieved from https://www.visualstudio.com/

[7] "Microsoft Visual Studio", online, accessed April 2017, retrieved from https://en.wikipedia.org/wiki/Microsoft_Visual_Studio

[8] "OpenCV library", online, accessed April 2017, retrieved from https://opencv.org/

[9] "OpenCV", online, accessed April 2017, retrieved from https://en.wikipedia.org/wiki/OpenCV

[10] "Zbar", online, accessed April 2017, retrieved from https://en.wikipedia.org/wiki/ZBar

[11] "Zbar bar code reader", online, accessed April 2017, retrieved from http://zbar.sourceforge.net/

[12] "Kinect", online, accessed April 2017, retrieved from https://en.wikipedia.org/wiki/Kinect

[13] "The GPS", online, accessed April 2017, retrieved from http://www.math.tamu.edu/~dallen/physics/gps/gps.htm