Autonomous Navigation of Mobile Robot Using Kinect Sensor

N.A. Zainuddin., Y.M. Mustafah, Y.A.M. Shawgi., N.K.A. M. Rashid
Department of Mechatronic Engineering,
International Islamic University Malaysia,
Kuala Lumpur, Malaysia
e-mail: yasir@iium.edu.my

***Abstract*—The problem of achieving real time process in depth camera application, in particular when used for indoor mobile robot localization and navigation is far from being solved. Thus, this paper presents autonomous navigation of the mobile robot by using Kinect sensor. By using Microsoft Kinect XBOX 360 as the main sensor, the robot is expected to navigate and avoid obstacles safely. By using depth data, 3D point clouds, filtering and clustering process, the Kinect sensor is expected to be able to differentiate the obstacles and the path in order to navigate safely. Therefore, this research requirement to propose a creation of low-cost autonomous mobile robot that can be navigated safely.**

Keywords-Kinect sensor, Mobile Robot, Navigation, OpenCV, point cloud.

I. INTRODUCTION

The autonomous mobile robot is widely used in many applications. The sizes are different depending on the application. It can be found in various scenes, either harmful places or safe places, such as in store and warehouse, in earthquake scenes or in the corridor and the library. In recent years, the researchers seem has found interest in autonomous mobile robot. Thus, allowing the development of mobile robot into many different applications [1][2][3][4]. Despite the fact that the technology of robotics evolved rapidly, most of the existing vigilant robots are not fully automatic and unreliable for surveillance purposes [1][2]. To make the situation worse, they are not stable thus not available in the market.

Therefore, this paper proposes a creation of low-cost autonomous mobile robot that can be navigated safely. Thus, it can be used in many applications such as security robot, surveillance of dangerous or inaccessible place to human and executing a routine job like monitoring jobs that human find it so tedious.There are four sections in this paper. The structure of the paper is as follows. In section two, design and method of the research are described. Meanwhile, in section three the experimental results are presented. Finally, the conclusion is drawn in section five.

II. METHODOLOGY

In this paper, an autonomoud navigation of mobile robot is presented. Like any other robot, the design of the mobile robot is divided into two part which are hardware and software. For hardware, the mobile is designed to be able to carry a on-board PC and have suitable shape to move freely. Plus, the design must be able to carry another important components such as sensor, encoder and microprocessor. For the sensor, in this paper, we used Kinect sensor. Kinet sensor provides a low-cost solution to autonomous robot. It successfully replaced the expensive laser-based sensors that have been used so far.

For software, the essential component is the control and the navigation of the mobile robot [5]. Thus, the data is extracted by the help of Microsoft Visual Studio 2010, OpenCV library, OpenNI library and Point Cloud library for the control and navigation algorithms. After the robot platform is built, the algorithm implementation is done. The algorithm involves the differentiation of path and obstacle in order to make sure the mobile robot can navigate safely. The research ends with test and evaluation of the system.

A. Robot Implementation

The mobile robot platform is built in this stage. The design of the mobile robot consists of two decks, which are base deck and upper deck. The upper deck is used to carry the on-board PC, meanwhile the base deck is used to place the drive system, sensors and electronics components. Figure 1 shows the design of the mobile robot platform using Computer Aided Design (CAD) software.

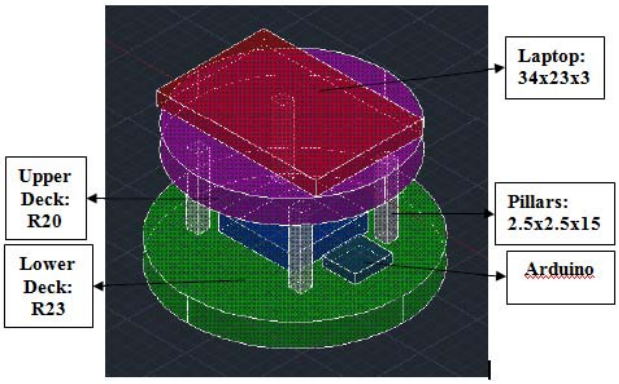

Figure 1. CAD design of the mobile robot platform.

The platform is designed to exhibit some important features like mobility, autonomy, and interactivity. The base with the largest diameter is set to be round. This is due to round-shaped platform allows the robot to rotate on the spot without hitting anything. The bottom deck is designed to be larger than upper deck in order to allow enough space for the Arduino, Motor driver, LIPO battery and sensors. As stability is also an important measure, the deck thickness is also crucial, the base must be stable when the robot turn, run or stop at full speed and upper deck must able to hold the weight of the on-board PC. Thus, acrylic board of 8 mm thickness was used for bottom deck, while 5 mm thickness for the upper deck.

The platform is set to be about the same width as a human so that it can navigate through doorways. This means that the diameter of the decks must typically be less than 50-60 cm. Most doors are 75 cm or greater in width, thus the radius for bottom deck is 23 cm and the rectangle shape of the upper deck is 20 cm x 20 cm. The size of the upper deck is finalized after

978-1-4799-7635-5/14 $31.00 © 2014 IEEE
DOI 10.1109/ICCCE.2014.21

considering the size of the on-board PC. The normal onboard PC is measured in order to estimate the size of the upper deck.

B. Obstacle Detection Algorithm

As the main sensor used in this research is the Kinect sensor, thus, the data from the Kinect sensor is used in the obstacle detection algorithm. The obstacle detection algorithm starts with the data acquisition by the live video from the Kinect sensor camera. Then, the data is converted in the depth data map and further, is converted into 3D point clouds. The 3D point cloud data carry the X, Y, Z coordinates data of the environment in order to reduce the processing the time, the point clouds are reduced by using the voxel Filtering. At this stage, the mobile robot still does not recognize the wall, the obstacles and the floor, thus the clustering process is done in order to organize the point clouds to understandable parameters of the mobile robot. After the clustering, the mobile robot is expected to recognize the obstacles and able to avoid them.

C. Navigation Algorithm

After the obstacles are detected, now, the mobile robot needs to avoid the obstacles and navigate to the free path. The last stage in the obstacle detection algorithm is the clustering. The data from the clusters are used as the inputs of the navigation system. If the current coordinate system is equal to the target (final location) coordinate, the mobile robot will rotate 90 degree and then suspend the program. However, if the target is not reached, the obstacles in the environment are scanned. If there are any close obstacles, the motor will avoid the obstacle and navigate in the free path. If there are no obstacles (free path), the motor is set to move in straight forward direction. The navigation algorithm is illustrated in the flowchart in Figure 2.

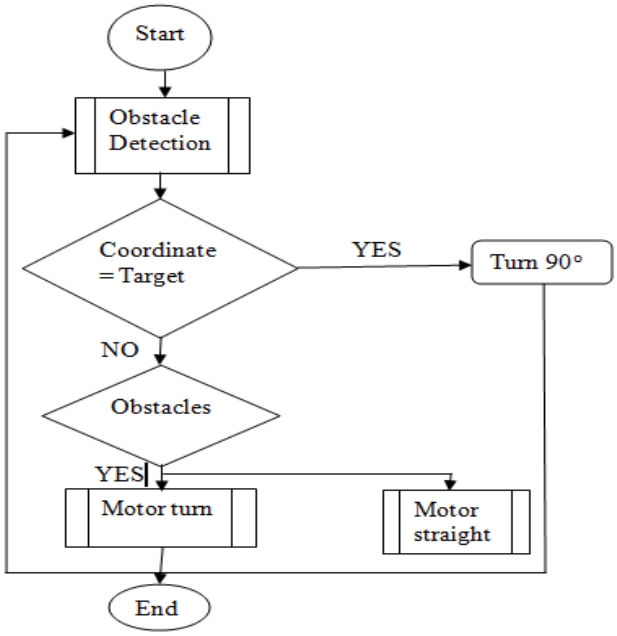

Figure 2. Navigation algorithm

III. Results and Discussions

As overall the system involving many stages, the outputs of each stage are discussed below. It starts with mobile robot construction and then the algorithm implementation by using Microsoft Visual Studio 2010.

A. Integration of the Robot

The main sensor used in this research is Kinect sensor. The Kinect sensor is designed to be mounted on the robot as shown in Figure 3.The Kinect sensor is programmed by using the Microsoft VisualStudio, OpenNI library and point cloud library. The command is sent to the microcontroller and motor driver board. The microcontroller will control the movement of the feedback from the robot will be sent back to the microcontroller and then the feedback coordinates will send back to the software algorithm.

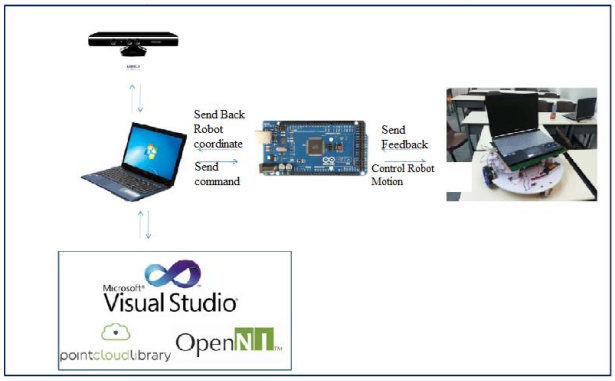

Figure 3. System setup

B. Capture video

By definition, video is a composition of many still images that produce a motion like image. The code starts by creating a Mat object that named as frame by executing the command cv::Mat frame. The command cv::Video Capture cap () is used to activate the default camera. After that, the code checking whether the camera is valid or not by executing if (!cap.isOpened()){printf("No Camera Detected")}, if the camera is not available, the message “No Camera Detected” will be appeared and on contrast if the camera is detected, a window named Webcam Video is created and the camera will start to capture the video by for(;;){cap>>frame;} and display it, cv::imshow("Webcam Video",frame). The waitKey() function in here serves the same role as before. So, Fig. 5 shows the output.

C. Depth Data

The bright color indicates that the object is near to the camera. However, being too near will cause the object not detected and it is indicated as black in color (Figure 4). The farthest object will be in dark color. The output is an accumulative histogram of the frequency of occurrence of each depth value. The histogram is built based on the depth values of each pixel. The pointer, *pDepth is used in order to access all the pixels in the updated image captured by Kinect sensor.

Figure 4. Depth map by kinect

D. Disparity Map

The basic way of writing disparity map is by using StereoBM and StereoBGM function. However, these functions require two input images. Thus, it is not suitable for Kinect sensor based disparity map. By using Kinect sensor, the algorithm of the code has started with the depth map acquisition. From the depth map, the disparity map is produced. OpenNI library is used for these purposes. The depth values produced is in mm. Note that all the objects, disparity_map, depth_map are declared as a Mat object so that they can be called out in the imshow () for the display purposes. After the program is executed, the output is as Figure 5.

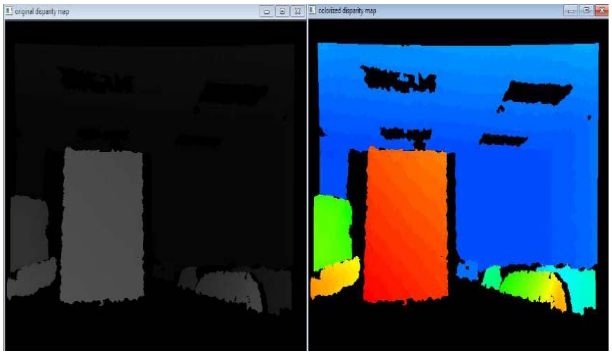

Figure 5. Disparity map by kinect

E. Point cloud

The depth data from the frames (video) captured by Kinect sensor is used to reconstruct the 3D point cloud. From the Kinect depth data, this point cloud class needs the width and height data. The width will specify the total number of points, pi, in the unorganized point cloud set, P. The width also be used in specifying the width (row) of the organized point cloud dataset. Meanwhile, for height, it is used to specify the height of the organized point cloud dataset and to check whether the dataset is organized or not by setting the unorganized dataset to one. An array called PointT also created in order to store all the points.

F. Filtering using Voxel Grid

Through In terms of filtering, the voxel filtering is used to reduce the unwanted 3D point clouds. It is a built-in function in Point Cloud library. The VoxelGrid() function will produce a 3D voxel grid over the input point cloud data. In each of the voxels, or the 3D box, all of the point clouds will be filtered or downsampled with their centroid. In simpler words, an average with respect to the centroid is done. It starts with the reading of the point cloud data (.pcd) from the depth data of the Kinect sensor. When the input data is triggered, the computation is done and the output will be stored in the cloud_filtered variable.

G. Clustering

As the data from the point cloud is unorganized and not recognizable by the mobile robot, a clustering algorithm is needed. One of the clustering methods that can be used is Euclidean. The output points of the filtering will be the input in this clustering. The most important thing in this algorithm is creating the KdTree. KdTree is used in as the searching method in this algorithm. A vector of point indices, PointIndices is also needed to be created. The PointIndices is the vector that contains the actual index information. This vector will save the indices information of the detected cluster. As the point cloud is in 3D, PointXYZ is used. Note that there are values are needed to be set in this code, which are the tolerance, minimum cluster size, and the maximum cluster size.

IV. CONCLUSIONS

This paper focuses on autonomous navigation using Kinect sensor. Kinect sensor is chosen based on cost, effectiveness and real time processing. This paper sets out to develop an autonomous mobile robot navigation using Kinect sensor. At the same time, this research is carried out to explore existing algorithm used in the development of navigation. In addition, this research also investigates how to navigate the mobile robot successfully in an indoor environment as the biggest challenge of the indoor environment is the quality of images captured as the lighting condition is quite unsatisfactory. The software implementation comprises of the obstacle detection algorithm and navigation algorithm. As for now, the quality of 3D point cloud produced is quite unsatisfactory. Thus, the future work may include the optimization and noise reduction. To conclude, the current technology used in autonomous navigation is vision-based sensor. However, the high cost of the vision-based sensor always becomes the limiting factor. Therefore, the Kinect sensor provides the solution to these problems.

REFERENCES

[1] D.S. Correa, D.F. Sciotti and M.G. Prado. Mobile Robots Navigation in Indoor Environments Using Kinect Sensor. Second Brazilian Conference on Critical Embedded System, Carlos, Brazil: IEEE. 2012, pp. 36-41.

[2] P. Benavidez and M. Jamshidi. Mobile Robot Navigation and Target Tracking System. 6th International Conference on System Engineering, New Mexico , 2011, pp. 299-304.

[3] J. Cunha, E. Pedrosa, C. Cruz and N.Lau. Using a Depth Camera for Indoor Robot Localization and Navigation, 2011.

[4] L. Somlyai and Z. Vamossy. Map Building with RGB-D Camera for Mobile Robot. 16th International Conference on Intelligent Engineering Systems, Lisbon, Portugal, 2012, pp. 489-493.

[5] D.S. Correa, D.F Sciotti and M.G Prado. Mobile Robots Navigation in Indoor Environments Using Kinect Sensor. Second Brazilian Conference on Critical Embedded System Carlos, Brazil: IEEE, 2012, pp. 36-41.

[6] R. Palaniappan, P. Mirowski, T.K. Ho, H. Steck, P. Whiting and M. MacDonald. Autonomous RF Surveying Robot for Indoor Localization and Tracking. International Conference on Indoor Positioning and Indoor Navigation. Guimarales, Portugal, 2011.

[7] M. Castelnovi, M. Miozzo, A. Scalzo, M. Piaggio, A. Sgorbissa, and R. Zaccaria. Surveillance Robotics: analysing scenes by colours analysis and clustering, 2011.

[8] O. Hachour. Path Planning of Autonomous Mobile Robot. VOL. 2(4), 2008.

[9] P. Lester. A* Pathfinding for Beginners. Retrieved from: http://www.policyalmanac.org/games/aStarTutorial.htm/, 2005.

[10] K. Parnell and R. Bryner. Comparing and Contrasting FPGA and Microprocessor System Design and Development 2004.