

Automatic Navigation for A Mobile Robot with Real Time Depth Kinect Sensor and Map Database

Ali Uroidhi, Ronny Mardiyanto, Djoko Purwanto

Department of Electrical Engineering
Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia

ali.bafagih@hotmail.com, ronny@elect-eng.its.ac.id, djoko@ee.its.ac.id

Abstract – The ability to view and remember the surrounding conditions is important in navigation. Kinect sensor has some limitations. The viewing angle of Kinect sensor is too narrow to detect objects on its surrounding and distance measurement of Kinect sensor is limited to objects that can reflect laser light. This research proposes a navigation system which is using data of Kinect depth and databases map. The experiment uses a mini computer (Raspberry PI 2) to process data from Kinect. From the experiment can be concluded that the average time needed to process data from each frame is 379.73 ms. The process speed is not fast, but it's fast enough to navigate the mobile robot to avoid obstacle.

Keywords – Kinect, Map, Mobile Robot, Navigation, Raspberry PI

I. INTRODUCTION

Basically, the robot can be placed in a unique and dangerous place. Motion planning is the basic of mobile robot navigation. There are various types of navigation system for mobile robots. One of this navigation systems is using Kinect sensor. In this paper, the navigation system of the mobile robot is using online and offline data. The online data are taken directly from the Kinect sensor and the offline data are taken from the results of processed Kinect sensor data that has been saved.

Kinect is a motion sensing input device by Microsoft and relatively low price [1]. Kinect sensor is not only used in computer games, but also in robot researches [2] [3]. In the navigation system, Kinect sensor is used to recognize the surrounding environment. The results from the recognition process will be used for motion planning. Kinect sensor utilizes laser light to generate three dimensional data from its surrounding, but not all areas can reflect light well, so there will be some pixel errors. Kinect sensor also have viewing angle about 57°. It covers only the area in front of the mobile robot, so the areas aside the mobile robot cannot be captured.

Data from Kinect can be used to create a map [4] [5]. Map is used to remember the obstacle and to correct the error pixels from Kinect sensor. Map can also be used to determine the direction of navigation. There are several methods that can be used for navigation systems, one of

them is Vector Field Histogram (VFH) [6] [7] [8]. In this research VFH applied to the online and offline data.

II. NAVIGATION METHOD

This paper proposes a method of navigation that is using Kinect and maps. Kinect uses laser sensors. The reflected light is used to measure the distance, but not all areas can reflect laser light well. Kinect sensor has about 57° of viewing angle. It is only used to monitor obstacle in front of Kinect. Because of these weaknesses from Kinect sensor, the map data is used to help navigate the mobile robot.

A. Differential drive kinematics

Based on the design, mobile robot has its own movement character. This is illustrated in Fig. 1 and Fig. 2. Fig. 2 shows the relationship between the velocity of wheels and the direction of mobile robot. The difference from the velocity of wheels from motion will form the radius (R) of the Instantaneous Center of Curvature (ICC)

- If $V_l = V_r$, then mobile robot will move straight forward, so value of R is infinity
- If $V_l > V_r$ then mobile robot will move to the left
- If $V_r = 0$, then mobile robot will rotate on the right wheel, so value of R is half-length robot
- If $V_r = -V_l$ then mobile robot will spin at the midpoint of the robot, so value of R is 0

Relationship of R to the wheel speed is shown in Equation 1 and Equation 2, where ω is the rotation speed of the ICC

$$R = \frac{l (V_l + V_r)}{2 (V_r - V_l)} \quad (1)$$

$$\omega = \frac{V_r - V_l}{l} \quad (2)$$

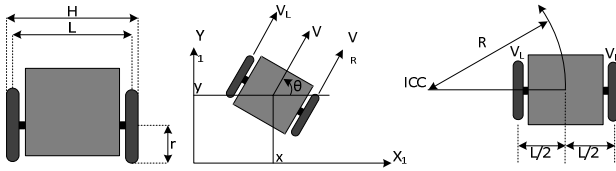


Fig. 1. A Description of The Variable and The Position

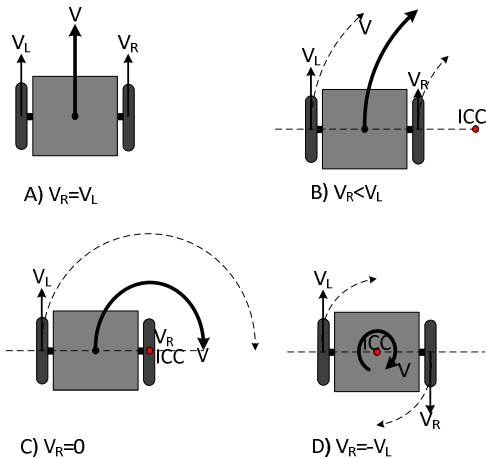


Fig. 2. The Direction of Movement of Differential Drive Kinematics

Based on Equation 1 and Equation 2, when V_r and V_l is known, we can find the ICC location use Equation 3. ICC location can determine the robot location via computed by Equation 4

$$ICC = [x - R * \sin(\theta), y + R * \cos(\theta)] \quad (3)$$

$$\begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = \begin{bmatrix} \cos(\omega) & -\sin(\omega) & 0 \\ \sin(\omega) & \cos(\omega) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x - ICC_x \\ y - ICC_y \\ \theta \end{bmatrix} + \begin{bmatrix} ICC_x \\ ICC_y \\ \omega \end{bmatrix} \quad (4)$$

B. Movement system using fuzzy logic control

Because navigation use map, the robot knows its location, then movement system based on the location is needed. This movement system research uses fuzzy logic control. It uses multiple membership functions like membership function of the input variable and the output variable. Input variable is value of the distance (d) and the angle (θ), which includes:

- Distance (d) is divided into 3 fuzzy sets, namely: Near (N), Medium (M), and Far (F)
- Angle (θ) is divided into 3 fuzzy sets, namely: Left (L), Center (C), and Right (R)

Output variable is the value of the right motor speed (V_r) and the left motor (V_l), which includes:

- V_r is divided into 5 fuzzy sets, namely: fast backward (FB), medium backward (MB), stop (S), medium forward (MF), and fast forward (FF)

- V_l is divided into 5 fuzzy sets, namely: fast backward (FB), medium backward (MB), stop (S), medium forward (MF), and fast forward (FF)

Fig. 3, Fig. 4 shows the membership function of input variable (distance and angle) and Fig. 5, Fig. 6 shows the membership function of output variable (motor speed right and motor speed left). Fuzzy logic rules for motor right and motor left are shown in Table I and Table II

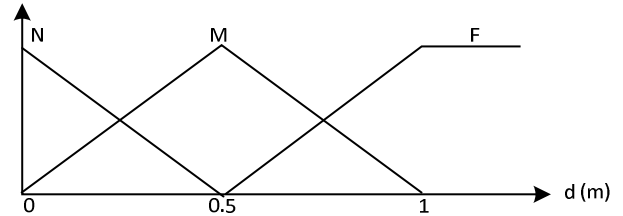


Fig. 3. Membership Functions Fuzzy For distance Input

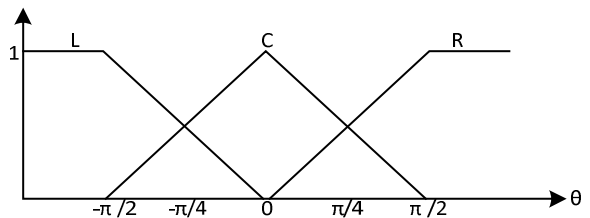


Fig. 4. Membership Functions Fuzzy for Angle Input

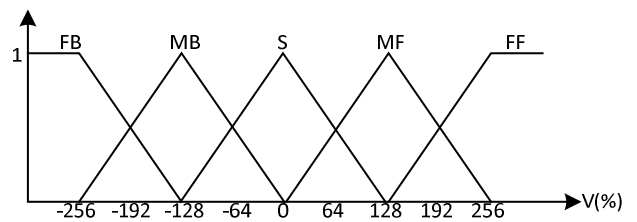


Fig. 5. Membership Functions for Output Right Motor Velocity(v_r)

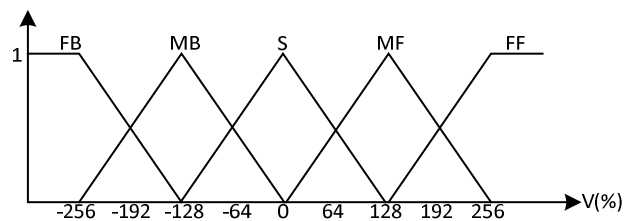


Fig. 6. Membership Functions Fuzzy for Output Right Motor Velocity(v_r)

TABLE I. DESIGN OF FUZZY LOGIC RULES FOR MOTOR RIGHT

		Distance		
		N	M	F
Angle	L	S	MB	MB
	C	S	FF	FF
	R	S	MF	MF

TABLE II. DESIGN OF FUZZY LOGIC RULES FOR MOTOR LEFT

		Distance		
		N	M	F
Angle	L	S	MF	MF
	C	S	FF	FF
	R	S	MB	MB

C. Navigation use map

The navigation system utilizes the data from the depth data of Kinect sensor. Fig. 7 is an example of depth data from sensors Kinect. Kinect depth is 11 bits each pixel, so each pixel has a value of 0 - 2048. Each pixel is presented with color. Least Significant Bit (LSB) color scheme are shown in the Fig. 7. Depth data generates data in Cartesian coordinates configuration by using Equation 5 for the Z axis, the equation 6 for the X axis, and Equation 7 for the Y axis,

$$Z_k(i, j) = 0.1236 * \tan\left(\frac{dept(i, j)}{2842.5} + 1.1863\right) \quad (5)$$

$$X_k(i, j) = \left(i - \frac{w}{2}\right) * 1.12032 * \frac{Z_k(i, j)}{1000} \quad (6)$$

$$Y_k(i, j) = \left(j - \frac{h}{2}\right) * 0.84024 * \frac{Z_k(i, j)}{1000} \quad (7)$$

Fig. 8 is illustrated relations between Cartesian coordinates of the depth data and Cartesian coordinates of the robot. Equation 8 to Equation 10 is show a relationship of coordinate-depth with coordinate-robot

$$Dkin = \sqrt{Xk^2 + Zk^2} \quad (8)$$

$$X' = x + (Dkin * \sin(\theta' + \theta)) \quad (9)$$

$$Y' = y + (Dkin * \cos(\theta' + \theta)) \quad (10)$$

The calculation results of depth data at j = 200, use Equation 5 and Equation 6, generating correlation X and Z are shown in Fig. 9.

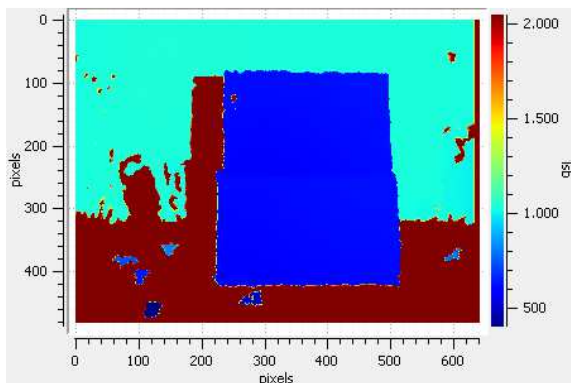


Fig. 7. Depth of Sensor Kinect

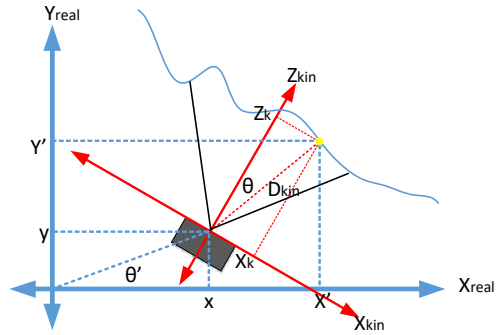


Fig. 8. Relationship of Coordinate Robot With Coordinate Kinect

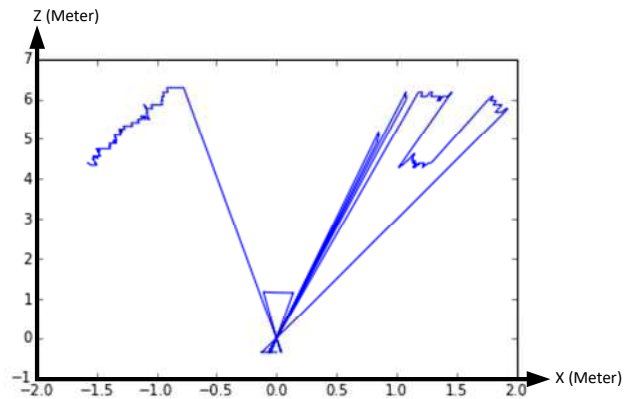


Fig. 9. Relationship The z-Axis With The x-Axis at j = 200

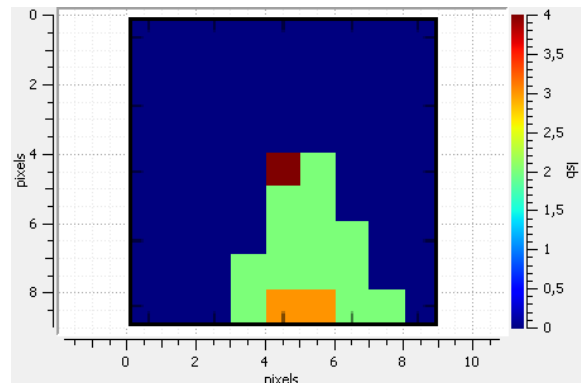


Fig. 10. Map Databases With Radius 1.25 Meter

The depth data is processed into robot's location data. The robot's location data is saved to a map database. Data results from the database is used for mobile robot navigation. These data will be segmented into several parts according robot position. The segmentation of data shown in Fig. 10. Map databases contain level of object. Lower level is obscure road. It is saved with number 1. Second level is the surest road. It is saved with number 2. Third level is obstacle. It is saved with number 3. Fourth level is the position of the robot. It is marked with number 4, it is not saved to databases map, but saved in databases location.

Polar histogram based on depth data from Kinect is shown in Fig. 11. Polar histogram based on map segment is shown in Fig. 12. Both polar histogram is combined for mobile robot motion planning. Its result is shown in Fig. 13.

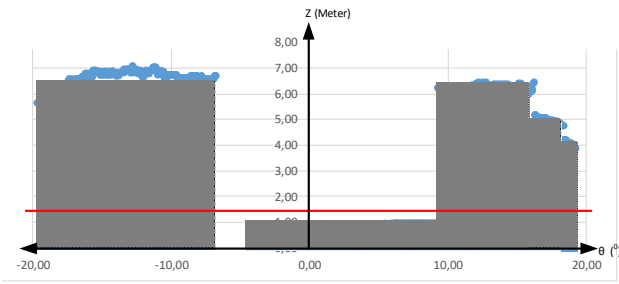


Fig. 11. Resouting Polar Histogram of Depth Kinect

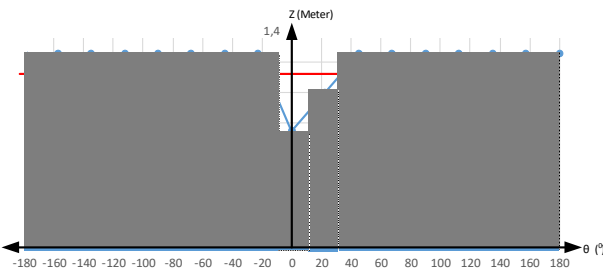


Fig. 12. Resouting Polar Histogram of Map Databases

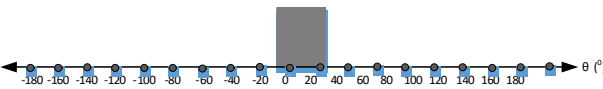


Fig. 13. Resouting Polar Histogram of combination Map Databases with Depth Kinect

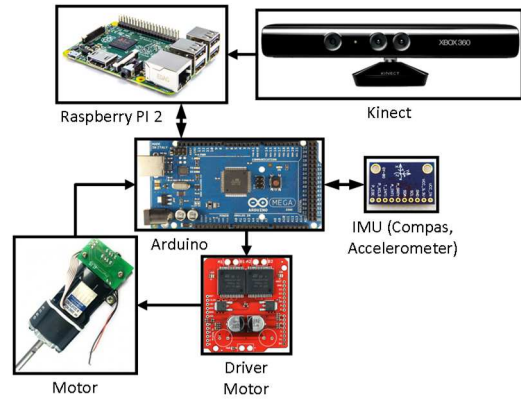


Fig. 14. Mobile Robot System Controller

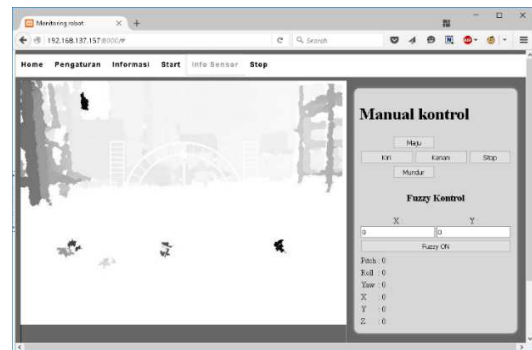


Fig. 15. Web Interface for Controller

III. EXPERIMENTS AND ANALYSIS

A. Experiment system

The design of the mobile robot to experiments the system is tank like mobile robot. The main system comprises six components, which are mini-computer (Raspberry PI 2), Arduino, motor driver, motor, Kinect, and compass sensor. Minicomputer is used to process data from Kinect sensor, generates map databases, and plans the robot's motion. Arduino is used to process data from the robot's motion data to the motor, and is used to read compass sensor and rotary encoder. The sensors (rotary and compass) and motion motor is controlled by Arduino, so it can produce smoother motor's motion, because motor's motion is not disturbed by navigation process. The block diagram of the process is shown in Fig. 14.

The interface of the controller is web-based interface made with python language. It is used to display information from the sensors (Kinect, compass, and position rotary), give manual controls such as forward, backward, turn right and turn left, and add the target points that will be tracked. The design of the control interface are shown in Fig. 15

B. Navigation experiment

Navigation experiment is performed by placing obstacle on the route. The system detects the presence of obstacle. Obstacle is identified by the system in Raspberry PI via depth data of Kinect sensor and map databases. Raspberry PI send movement signals to the Arduino, which will be forwarded to the motor. Map databases help the ability to remember the position of obstacle. Its data is needed when viewing angle of the sensor does not reach obstacle. The results of the experiment is shown in Fig. 16.

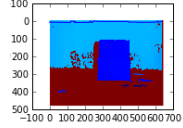
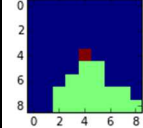
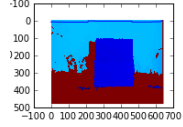
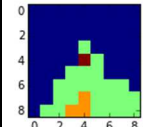
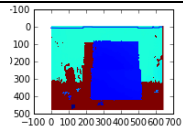
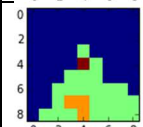
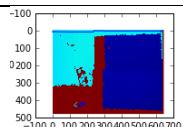
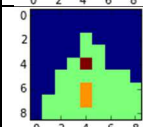
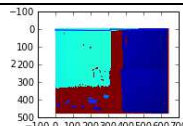
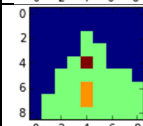
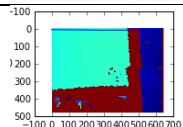
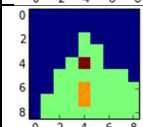
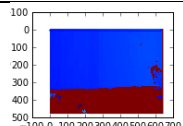
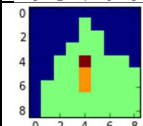
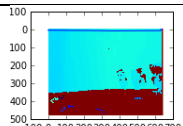
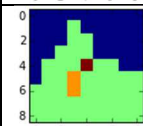
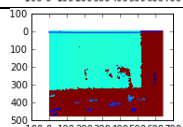
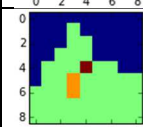
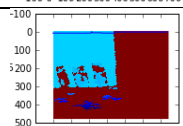
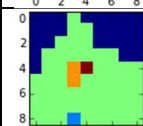
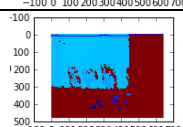
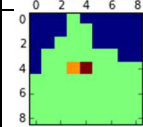
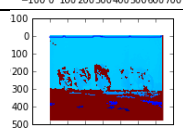
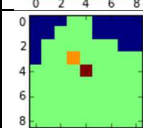
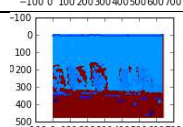
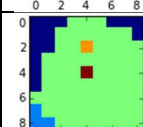


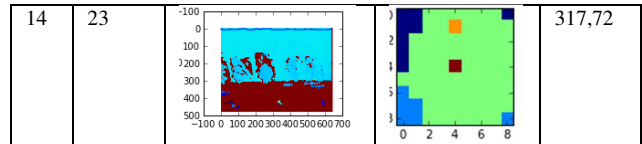
Fig. 16. Results of Experimental Navigation

C. Processing time experiment

Raspberry PI is used for motion planning. The processing time of each frame comprise four process, which are capture Kinect data, create maps, comparing the map with the data Kinect, and motion planning. Processing time can be seen in Table III.

TABLE III. TIME DATA PROCESSING BY THE RASPBERRY PI

No.	Frame	Depth	Map	Time (ms)
1	3			593,398
2	4			587,25
3	5			563,952
4	6			561,798
5	7			565,826
6	8			450,724
7	11			226,092
8	12			212,613
9	13			184,479
10	15			243,754
11	16			241,901
12	18			265,622
13	20			301,112



IV. CONCLUSION

This paper shows applications for motion planning for mobile robot navigation. System of motion planning contains two data readings, i.e. online data and offline data. Online data is the data directly from the sensor Kinect. Offline data is data results of Kinect sensor in map form. Polar histogram is produced by both readings (online data and offline data). It can determine motion planning of mobile robot. Basically all the process is performed by the Raspberry PI. The average time to process data from each frame is 379.73 ms. The processing speed is not fast, but the mobile robot navigation can be resolved

REFERENCES

- [1] E. Machida, M. Cao and T. Murao, "Human Motion Tracking of Mobile Robot with Kinect 3D Sensor," in *SICE Annual Conference*, Japan, 2012.
- [2] M. Cao and H. Hashimoto, "Specific Person Recognition and Tracking of Mobile Robot with Kinect 3D Sensor," *IEEE*, pp. 8323-8328, 2013.
- [3] A. Sgorbissa and D. Verda, "Structure-based object representation and classification in mobile robotics through a Microsoft Kinect," *Robotics and Autonomous Systems*, no. 61, pp. 1665-1679, 2013.
- [4] R. Mardiyanto, J. Anggoro and F. Budiman, "2D Map Creator for Robot Navigation by Utilizing Kinect and Rotary Encoder," in *Intelligent Technology and Its Applications*, 2015.
- [5] H. I. M. A. Omara and K. S. M. Sahari, "Indoor Mapping using Kinect and ROS," in *Multi-agent Systems and Robotics (ISAMSR)*, Agents, 2015.
- [6] J. Borenstein and Y. Koren, "The Vector Field Histogram -Fast Obstacle Avoidance for Mobile Robots," *IEEE TRANSACTIONS ON ROBOTICS AND AUTOMATION*, vol. 7, no. 3, pp. 278-288, 1991.
- [7] I. Ulrich and J. Borenstein, "VFH+: Reliable Obstacle Avoidance for Fast Mobile Robots," in *IEEE International Conference on Robotics & Automation*, Leuven, Belgium, 1998.
- [8] I. Ulrich and J. Borenstein, "VFH*: Local Obstacle Avoidance with Look-Ahead Verification," in *IEEE International Conference on Robotics & Automation*, San Francisco, 2000.