Proceedings of The 2016 IEEE International
Conference on Real-time Computing and Robotics
June 6-9, 2016, Angkor Wat, Cambodia

A Localization and Navigation Method with ORB-SLAM for Indoor Service Mobile Robots

Shirong Wang, Yuan Li, Yue Sun, Xiaobin Li, Ning Sun, Xuebo Zhang, Ningbo Yu*

Abstract— **Autonomous mobile robots need to acquire environment information for localization and navigation, and thus are usually equipped with various sensors. Consequently, the system is complex and expensive, bringing obstacles for general home applications. In this paper, we present an efficient, yet economic and simple solution for indoor autonomous robots, consisting of a basic mobile platform, a Kinect V2 sensor and a computing unit running Linux. Within the ROS environment, the ORB-SLAM algorithm, pointcloud processing methods and a feedback controller have been developed and implemented respectively for localization, obstacle detection and avoidance, and navigation. Experimental results showed robust localization, safe and smooth navigation, good motion control accuracy and repeatability, demonstrating the efficacy of the system architecture and algorithms.**

I. Introduction

In recent years, along with the increasing number of aging people and decrease of labor population, the demand for service robots is sharply growing and autonomous mobile robots for services at home, care-giving agencies, office buildings, etc, are remarkably popular. Autonomous mobile robots can assist human to complete various tasks, and they need to acquire the current pose and environment information for localization, motion planning and control [1], [2].

Autonomous mobile robots are usually equipped with a variety of sensors, such as range sensors of laser or ultrasound or infrared principles, GPS, IMU, camera, high precision encoder and so on, to acquire information from the environment in which they are deployed [3], [4], [5]. With these information, the mobile robot can localize itself and make motion planning and control. More sensors can improve the system performance, but will increase the complication and expense of the system, making it difficult for use by average people and hard for large-scale deployments.

As technologies are advanced, RGB-D sensors have been developed, such as Microsoft Kinect, ASUS Xtion Pro Live, Intel RealSense, etc. RGB-D sensors have the advantages of rich environmental information, non-contact measurement and low cost. With one RGB-D sensor, color and depth information can be obtained at the same time, and they have been widely used in the field of robotics. Extensive research has been focused on RGB-D based SLAM methods [6], [7], [8]. Felix Endres et. al. from Freiburg University proposed the RGB-D SLAM algorithm, using the Kinect sensor in the ROS platform to realize autonomous localization and mapping [9]. Daniel Maier et. al. used the NAO humanoid robot together with ASUS Xtion Pro Live sensor to realize autonomous localization, obstacle avoidance and motion planning in indoor environment [10]. Washington University and Microsoft lab developed a real-time visual SLAM system based on graph optimization to build 3D maps [11]. Carnegie Mellon University has developed a polygon reconstruction and fusion algorithm to extract the planar features in the original data [12]. Since the pointcloud map is not directly used, the real-time performance of the algorithm can be improved. Raul Mur-Artal et. al. proposed the ORB-SLAM algorithm [13], taking advantages of the ORB features, and the system operates well in real time even without GPU. These work have significantly improved the accuracy and real-time performance of RGB-D based SLAM techniques, and thus promoted their application in mobile robots.

When autonomous mobile robots move in the environment, they need the ability to detect and avoid obstacle, in addition to localization. Thus, various sensor technologies have been taken to obtain the information of the obstacles in the environment. Liu et. al. used raw point cloud from 3D laser to solve the 2.5D navigation problem [14]. In recent years, with the growing application of RGB-D sensors, studies on RGB-D based obstacle detection and avoidance arose [15], [16].

In this work, we propose and establish a low-cost autonomous mobile robot system, with a basic mobile platform equipped with a general RGB-D sensor and a computing unit running Linux. This mobile robot can autonomously localize itself in the map, detect and avoid obstacles, and navigate in the environment with its built in motion planning and control algorithms. The robot is able to perform various service tasks at different indoor environment such as home, care-giving agencies, office buildings, etc, with its simple architecture and minimized complexity. Furthermore, functionalities of the system can be conveniently extended by software development within the open ROS environment.

This paper is organized as following. The next section describes in details the hardware and algorithms of the mobile robot system. Section III presents the experiments and results. Finally, section IV concludes the paper.

This work is supported by the National Natural Science Foundation of China (61403215), the Natural Science Foundation of Tianjin (13JCYBJC36600) and the Fundamental Research Funds for the Central Universities.

Corresponding author Assoc. Prof. Dr. Ningbo Yu is with the Institute of Robotics and Automatic Information Systems, Nankai University, and Tianjin Key Laboratory of Intelligent Robotics, Nankai University, Haihe Education Park, Tianjin 300353, China. Phone: +86 (0)22 2350 3960 ext. 801, Email: nyu@nankai.edu.cn.

Mr. Shirong Wang, Mr. Yuan Li, Ms. Yue Sun, Mr. Xiaobin Li, Dr. Ning Sun, and Assoc. Prof. Dr. Xuebo Zhang are with the Institute of Robotics and Automatic Information Systems, Nankai University, and Tianjin Key Laboratory of Intelligent Robotics, Nankai University, Haihe Education Park, Tianjin 300353, China.

978-1-4673-8959-4/16/$31.00 © 2016 IEEE

II. METHODS

A. Hardware Platform and System Architecture

The hardware platform in this paper is as shown in Fig. 1. It is mainly composed of three parts: Yujin Kobuki mobile platform, Kinect V2 RGB-D sensor and a laptop as the mobile computing unit.

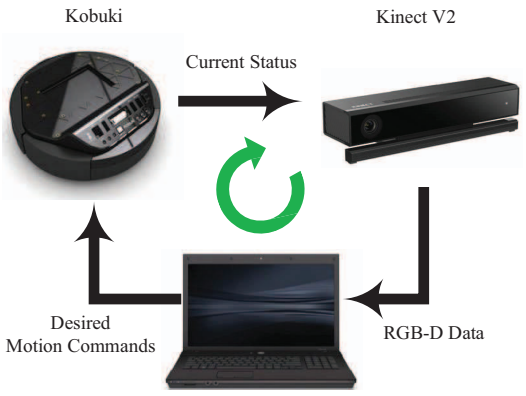

Fig. 1. The hardware platform

The Yujin Kobuki mobile platform is a low cost mobile platform, and it supports a variety of operating systems such as Windows, Linux, etc. Its maximum translational velocity can go up to 70 cm/s, maximal rotational velocity reaches 180 deg/s, and its load capacity is up to 5kg. Furthermore, this platform can also provide power supply for the Kinect v2 RGB-D sensors and the computing laptop. This mobile platform also has the advantages of supporting multiple programming languages, low cost, and so on. It has been widely used in the field of mobile robotics research and can meet the needs of various service tasks.

TABLE I
COMPARISON OF KEY PARAMETERS OF THE RGB-D SENSORS KINECT V1 AND V2

Feature	Kinect V1	Kinect V2	Comparison
Color Camera	640×480	1920×1080	↑
Depth Camera	320×240	512×424	↑
Frequency	30fps	30fps	–
Horizontal Field of View	58.5°	70.6°	↑
Vertical Field of View	46.6°	60°	↑
Max Depth Distance	4.5m	8m	↑
Min Depth Distance	0.4m	0.5m	↓
Depth precision	4cm	2cm	↑

The Kinect V2 sensor is released in 2014, from which both color image and depth image of the environment can be obtained. Compared with the Kinect V1 sensor, the new Kinect V2 sensor has improved a lot with respect to resolution, field of view, depth distance, and depth resolution, as shown in Table I.

The Ubuntu 14.04 LTS operating system equipped with ROS Indigo is installed on the mobile computing platform. We also used other tools including libfreenect2 and PCL. In this software environment, we implemented the ORB-SLAM algorithm for localization, together with algorithms for obstacle detection and motion planning and control. With these algorithms, the mobile robot are empowered with the capabilities of localization and navigation in cluttered environments, as shown in Fig. 2.

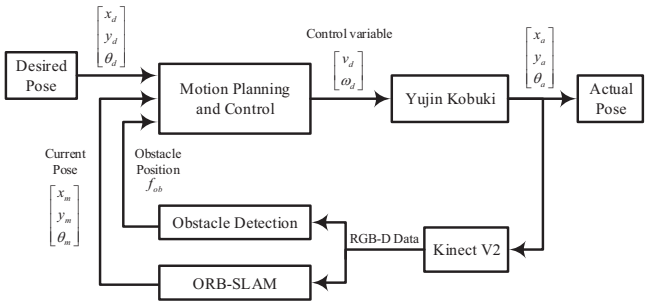

Fig. 2. The software architecture

- **Localization:** With the equipped Kinect V2 RGB-D sensor, the mobile robot can acquire both the color and also the depth data. Taking advantages of the ORB-SLAM algorithm, we can calculate the current pose $[x_m, y_m, \theta_m]^T$ for the mobile robot.
- **Obstacle detection:** With the point cloud data from the Kinect V2 sensor, the obstacle detection algorithm calculates the position of the obstacle relative to the mobile robot: $[d_{obs}, \theta_{obs}]^T$.
- **Motion planning:** According to the desired pose $[x_d, y_d, \theta_d]^T$, current pose $[x_m, y_m, \theta_m]^T$, and obstacle information f_{obs}, the motion planning algorithm gives the path to avoid the obstacle and pursue the target.
- **Motion control and execution:** The Yujin Kobuki mobile platform receives the control commands, drives it toward the targeting pose, and produces the actual pose $[x_a, y_a, \theta_a]^T$. This pose, calculated by the localization algorithm, together with the obstacle information, will be fed back to the motion control algorithm, which in turn produces the desired motion commands $[v_d, \omega_d]^T$.

B. Real-time Localization based on ORB-SLAM

Among the SLAM research based on RGB-D sensors, ORB-SLAM is an excellent algorithm that is accurate, fast, and stable [13]. Furthermore, the source code of this algorithm has been put in github and RGB-D interface of ORB-SLAM has been provided in [17].

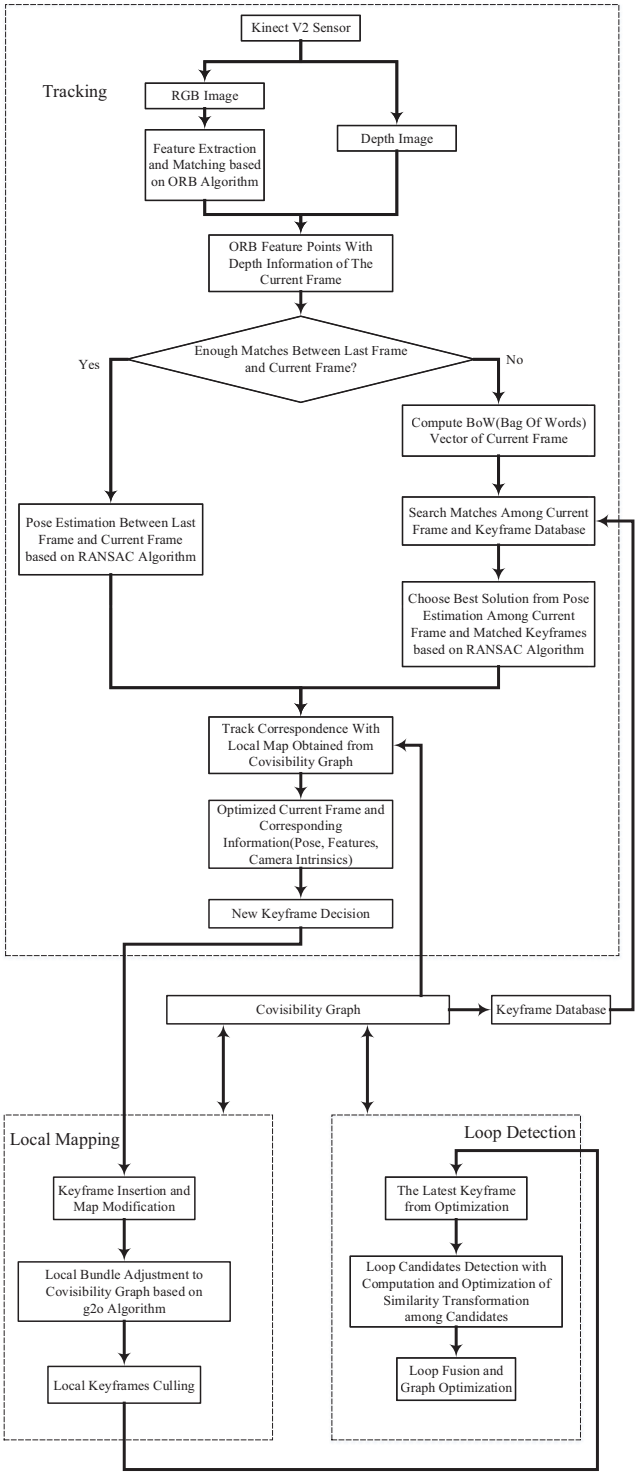

Fig. 3. The workflow of the slam method based on ORB features

The work flow of the ORB-SLAM algorithm is shown in Fig. 3. RGB-D information is used as ORB-SLAM system's input. Firstly, we extract ORB features from current frame and perform matching with the previous frame. An initial pose estimation based on RANSAC algorithm is processed from enough matches between current frame and last frame. If the matching fails, a visual vocabulary will be generated from ORB features of current frame using DBoW2 to perform global relocalization among keyframe database extracted from covisibility graph for a credible pose estimation. Covisibility graph is constructed from local mapping and loop closing threads. We use g2o algorithm to optimize the transformation of covisibility graph from correspondent pairs of frames for global consistency.

C. Obstacle Detection based on Pointcloud Segmentation

The mobile robot can get the color and depth information from environment with the RGB-D sensor. We can get the depth information of environment, and detect the possible obstacles with the pointcloud processing techniques, without asking for additional sensors.

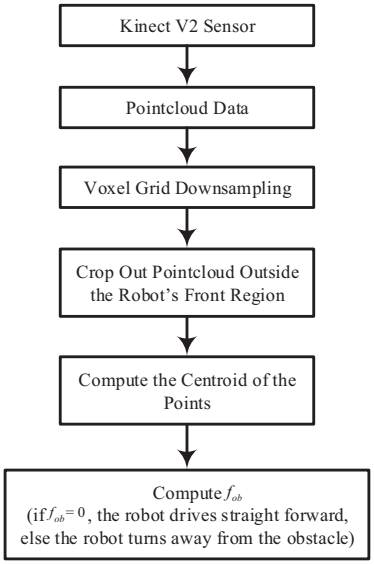

Fig. 4. The obstacle detection work flow

The obstacle detection algorithm that has been realized in this work is as shown in Fig. 4. First of all, in order to improve the efficiency of the algorithm and increase the computation speed, the pointcloud from Kinect V2 sensor is downsampled using the voxel grid. Then, points that go beyond the robot height and width or too far away from the current position, are cropped out. After that, the centroid of the cropped pointcloud can be calculated. Information about obstacles is sent to the motion planning and control algorithm, to ensure that the mobile robot can avoid obstacles in the subsequent movements.

D. Robot Navigation based on Motion Planning and Control

In this paper, we consider a differential-drive mobile robot. The kinematics of the robot described in inertial frame is given by

$$\begin{cases} \dot{x} = v\cos\theta \\ \dot{y} = v\sin\theta \\ \dot{\theta} = \omega \end{cases} \tag{1}$$

The system state $X = [x, y, \theta]^T \in R^3$ represents the robot position and orientation.

The control command is $u = [v, \omega]^T \in R^2$ and v, ω represent the linear and angular velocities respectively. Without loss of generality, the desired pose of the mobile robot can be set as $\left[0, 0, \frac{\pi}{2}\right]^T$, after necessary coordinate transformation. With the current pose from ORB-SLAM, the motion plan and controller will calculate the motion path and control input $[v, \omega]^T$.

Let α denote the angle between the X_R axis of the robots reference frame and the vector connecting the center of the desired position. ρ is the distance between the center of the robots reference and the goal position. Transform the coordinate into polar coordinates, when $\alpha \in \left(-\frac{\pi}{2}, \frac{\pi}{2}\right]$, the polar coordinates as

$$\begin{cases} \rho(x,y) = \sqrt{x^2 + y^2} \\ \alpha(x,y,\theta) = -\theta + \operatorname{atan2}(y,x) \\ \beta(x,y,\theta) = -\theta - \alpha(x,y,\theta) \end{cases} \tag{2}$$

Then, it can be derived in a matrix equation form:

$$\begin{bmatrix} \dot{\rho}(x,y) \\ \dot{\alpha}(x,y,\theta) \\ \dot{\beta}(x,y,\theta) \end{bmatrix} = \begin{bmatrix} -\cos\alpha(x,y,\theta) & 0 \\ \dfrac{\sin\alpha(x,y,\theta)}{\rho(x,y)} & -1 \\ -\dfrac{\sin\alpha(x,y,\theta)}{\rho(x,y)} & 0 \end{bmatrix} \begin{bmatrix} v \\ w \end{bmatrix} \tag{3}$$

When $\alpha \in \left(\frac{\pi}{2}, \frac{3\pi}{2}\right]$, setting $v = -v$ to redefine the forward direction of the robot, then

$$\begin{bmatrix} \dot{\rho}(x,y) \\ \dot{\alpha}(x,y,\theta) \\ \dot{\beta}(x,y,\theta) \end{bmatrix} = \begin{bmatrix} \cos\alpha(x,y,\theta) & 0 \\ -\dfrac{\sin\alpha(x,y,\theta)}{\rho(x,y)} & 1 \\ \dfrac{\sin\alpha(x,y,\theta)}{\rho(x,y)} & 0 \end{bmatrix} \begin{bmatrix} v \\ w \end{bmatrix} \tag{4}$$

The motion control algorithm can be designed as

$$\begin{cases} v(\rho) = k_\rho \rho \\ \omega(\alpha,\beta) = k_\alpha \alpha + k_\beta \beta \end{cases} \tag{5}$$

with $k_\rho, k_\alpha, k_\beta \in R$ being constant control gains. We can get a closed-loop system described by

$$\begin{bmatrix} \dot{\rho}(x,y) \\ \dot{\alpha}(x,y,\theta) \\ \dot{\beta}(x,y,\theta) \end{bmatrix} = \begin{bmatrix} -k_\rho \rho(x,y)\cos\alpha(x,y,\theta) \\ k_\rho \sin\alpha(x,y,\theta) - k_\alpha \alpha(x,y,\theta) - k_\beta \beta(x,y,\theta) \\ -k_\rho \rho(x,y)\sin\alpha(x,y,\theta) \end{bmatrix} \tag{6}$$

Thus, we can get the designed linear velocity and angular velocity as functions of the system state x, y, θ:

$$\begin{cases} v(x,y) = v(\rho(x,y)) \\ \omega(x,y,\theta) = \omega(\rho(x,y), \alpha(x,y,\theta), \beta(x,y,\theta)) \end{cases} \tag{7}$$

Substituting the linear velocity and angular velocity into the open-loop error dynamics, we can obtain the closed-loop error dynamics as

$$\begin{cases} \dot{x} = v(x,y)\cos\theta \\ \dot{y} = v(x,y)\sin\theta \\ \dot{\theta} = \omega(x,y,\theta) \end{cases} \tag{8}$$

By choosing appropriate control parameters $k_\rho, k_\alpha, k_\beta$ with the constraint of $k_\rho > 0$, $k_\beta < 0$, $k_\alpha - k_\rho > 0$, convergence of the closed-loop system can be ensured [18]. In this way, given the starting pose, the controller can calculate the control command according to the current pose and drive the mobile robot to the desired pose.

In order to achieve obstacle avoidance, the robot need to modify the control command with respect to the obstacle parameter. When $f_{ob} \neq 0$, the mobile robot will rotate to avoid the obstacle.

III. Experiments and Results

A. Experiments on Localization and Motion Control

As shown in Fig. 5, the targeting pose is $[0, 0, 0]^T$. The mobile robot was placed at eight different locations around the targeting pose, and moved autonomously to it. The dashed lines are the theoretical moving paths, and the solid lines are the experimental paths. As can be clearly observed in Fig. 5, the robot successfully moved to the targeting pose from all initial poses.

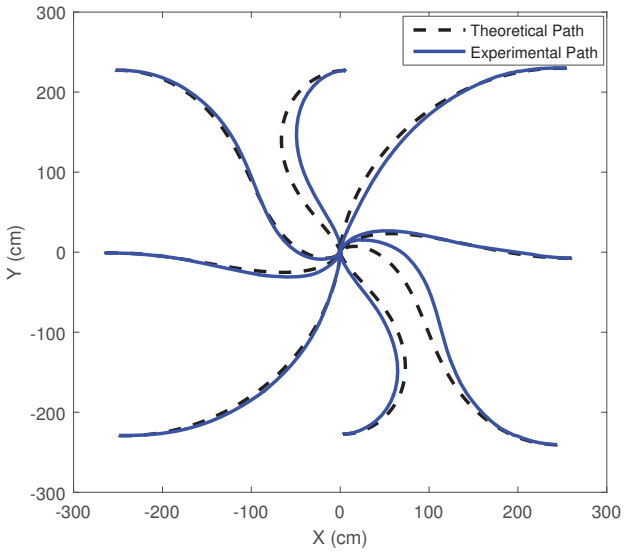

Fig. 5. Experimental results of traveled path

The final poses of the mobile robot in the experiment are shown in Fig. 6. The average deviation is 0.43cm in the X axis and 1.12cm in the Y axis. The absolute average deviation in angle is 2.16°. This also demonstrates the efficacy of the system architecture and localization and control algorithms.

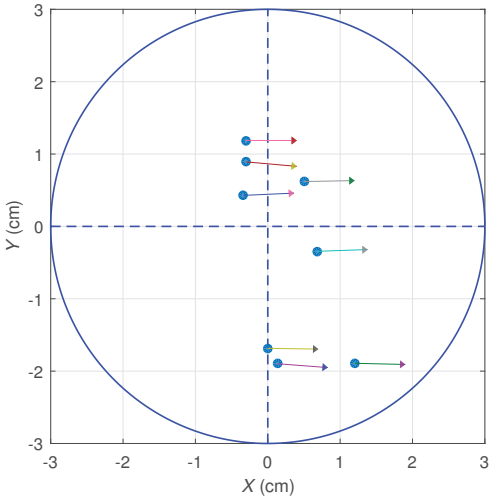

Fig. 6. Experimental results of the final poses

B. Experiments on Obstacle Avoidance

For the experiment of obstacle avoidance, an obstacle was placed in the pathway of the robot. The result is as shown in Fig. 7. From this plot we can find out that the robot can avoid the obstacle and reach the desired pose.

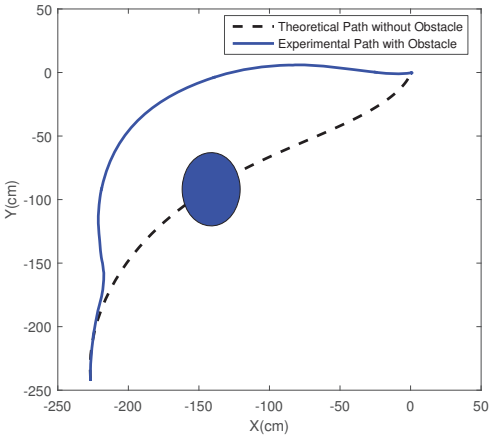

Fig. 7. Experimental result of obstacle avoidance

C. Results Analysis

The sources of error in the experiments mainly include the following aspects. First of all, the Kinect V2 sensor brings measurement inaccuracy in the obtained RGB-D data. Next, the Yujin Kobuki mobile platform is a low cost platform, and the motion control precision is limited. Furthermore, when the Kinect V2 sensor and the mobile platform are integrated together, the connection is not sufficiently rigid, which brings in inconsistency errors. Last but not the least, the libfreenect2 and ORB-SLAM algorithms demand heavy computing resources, slow down the control loop, and thus deteriorate the motion control performance. All these aspects can influence the precision of localization and control. Nevertheless, the integrated robotic system still achieves good control accuracy and repeatability.

IV. Conclusion

This paper has established a low cost autonomous mobile robot system, with a basic mobile platform equipped with a single RGB-D sensor and a computing platform, taking advantage of the ORB-SLAM algorithm for localization, pointcloud segmentation algorithm for obstacle detection, and a motion controller. Experiments have demonstrated the efficacy of the system structure and respective algorithms, and reasonable accuracy has been achieved. Besides, The system shows benefits of simple architecture, ease use, and open software platform. It can be conveniently expanded for specific service applications.

References

[1] S. Lee, S. Jung. "Novel design and control of a home service mobile robot for Korean floor-living life style: KOBOKER." *the IEEE International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)*, 2011.

[2] M. K. Habib, Y. Baudoin, F. Nagata. "Robotics for rescue and risky intervention." *the Annual Conference on IEEE Industrial Electronics Society (IECON)*, 2011.

[3] L. Yenilmez, H. Temeltas, "Real time multi-sensor fusion and navigation for mobile robots." *the 9th Mediterranean Electrotechnical Conference*, 1998.

[4] J. Tian, M. Gao, E. Lu, "Dynamic collision avoidance path planning for mobile robot based on multi-sensor data fusion by support vector machine." *the IEEE International Conference on Mechatronics and Automation (ICMA)*, 2007.

[5] F. Colas, S. Mahesh, F. Pomerleau, M. Liu, and R. Siegwart, "3D path planning and execution for search and rescue ground robots," *the IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2013.

[6] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox, "RGB-D mapping: Using Kinect-style depth cameras for dense 3D modeling of indoor environments." *the International Journal of Robotics Research*, vol. 31, pp. 647-663, 2012.

[7] J. Cunha, E. Pedrosa, C. Cruz, A. J. Neves, and N. Lau, "Using a depth camera for indoor robot localization and navigation." *DETI/IEETA-University of Aveiro*, Portugal, 2011.

[8] F. Endres, C. Sprunk, R. Kummerle, and W. Burgard, "A catadioptric extension for RGB-D cameras." *the IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2014.

[9] F. Endres, J. Hess, J. Sturm, D. Cremers, and W. Burgard, "3-D Mapping With an RGB-D Camera." *the IEEE Transactions on Robotics*, vol. 30, no. 1, 2014.

[10] D. Maier, A. Hornung, and M. Bennewitz, "Real-time navigation in 3d environments based on depth camera data." *the IEEE-RAS International Conference on Humanoid Robots*, 2012.

[11] G. Grisetti, S. Grzonka, C. Stachniss, P. Pfaff, W. Burgard, "Efficient estimation of accurate maximum likelihood maps in 3D." *the IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2007.

[12] J. Biswas, M. Veloso, "Depth camera based localization and navigation for indoor mobile robots." *Robotics: Science and Systems (RSS), RGB-D Workshop*, 2011.

[13] R. Mur-Artal, J. M. M. Montiel, J. D. Tardos. "ORB-SLAM: a versatile and accurate monocular SLAM system." *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147-1163, 2015.

[14] M. Liu, R. Siegwart, "Navigation on point-cloud a Riemannian metric approach." *the IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[15] D. S. O. Correa, D. F. Sciotti, M. G. Prado, D. O. Sales, D. F. Wolf, F. S. Osorio, "Mobile robots navigation in indoor environments using kinect sensor." *the Second Brazilian Conference on Critical Embedded Systems*, 2012.

[16] R. Mojtahedzadeh, "Robot obstacle avoidance using the Kinect." *Master Thesis*, KTH, Sweden, 2011.

[17] ORB-SLAM V2 for ROS Indigo, `https://github.com/raulmur/ORB_SLAM2`.

[18] R. Siegwart, R. N. Illah, S. Davide, *Introduction to autonomous mobile robots*, MIT press, 2011.