Patrol-Bot self driving for map making based on laser and asus depth sensors

Alfredo Chávez Plascencia

April 23, 2018

Abstract

Autonomous driving for map making of a clutter environments is a complex task due to the robot has to deal with precise measurements of the environment, in other words the robot must have a good field of view of the environment in turn. This technical report mainly presents the functioning and description of the ROS package called patbot_self_driving that has to do with the navigation of a PatrolBot in the empty area of a indoor environment while making a 2D map by means of a laser hokuyo and a 3D depth asus sensors.

Contents

I Preliminares 2

1 Introduction 3

1.1 Introduction. 3

II Case Study 4

2 System Description 5

2.1 System Hardware 5

2.1.1 PatrolBot 5

2.1.2 Asus Xtion Pro Live 5

2.1.3 Hokuyo 6

2.2 System Software 6

3 System and Simulation Results 7

3.1 Hardware User Guide 7

3.2 Software User Guide 7

3.3 Technical Instructions 8

3.4 Simulation Results 9

4 Conclusion 13

Part I

Preliminares

Chapter 1

Introduction

1.1 Introduction.

In order for a mobile robot (MR) to navigate autonomously in an unknown environment, either it must know a prior the empty and occupied area of the environment or it must build a map while travailing and exploring the empty space using the well known simultaneous localization and mapping (SLAM) algorithm [1].

Generally, environments can be highly dynamic where the robot may face people walking around. To this end, autonomous navigation is a complex task and it still faces a great challenge. Thus, navigation requires synchronization of modules for localization, path planning, navigation, control, sensor fusion, among others. Moreover, autonomous navigation plays an important part in autonomous self driving cars, intelligent wheelchairs, warehouse robots, assistant robots, among others, [2].

Furthermore, the laser is a common sensor for making maps in 2D however a combination of two lasers can be set up for making maps in 3D. For instance, [3] uses an expectation–maximization (EM) algorithm and a combination of 2D laser range finders to build up a 3D map of the environment. Also, the kinect sensor has faced a great research interest in recently years for navigation, localization and 2D and 3D map making. . For instance [4] applies the fast sampling plane filtering (FSPF) algorithm to reduce the volume of the 3D point cloud by sampling points from the depth image. So, the reduced volume can be used to build up either 2D or 3D maps.

It is a common practice for navigation to use a map that has been built in advance by the robot in joystick mode M_j, e.g, the user controls the MR by a joystick while traveling in the indoor environment using its sensors to make the map. The objective of our approach focuses mainly in letting the robot to navigate autonomously in the free space of the indoor environment. Meanwhile, the MR uses the hokuyo laser range finder and the asus xtion pro depth sensors to interact with the environment to build up a 2D map. It is worth to mention that more sensors can be placed in the MR in order for the robot to both have a better field of view or build up a 3D map.

This technical report is divided as follows: Chapter 2 gives a brief description of the PatrolBot and the sensors use in this work. Whereas, Chapter 3 describes the ROS-package, the system as well as the results of this application, and finally Chapter 4 gives the conclusion.

Part II

Case Study

Chapter 2

System Description

2.1 System Hardware

2.1.1 PatrolBot

The PatrolBot from adept Mobile Robots is a differential mobile robot designed specially for research [5]. By the proper devices, it can accomplish different tasks such as mapping, teleoperation, localization, monitoring, reconnaissance, vision, manipulation among others. Figure 2.1 shows the robot whereas the Table 2.1 shows the technical specifications.

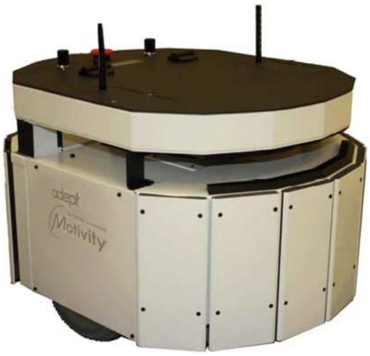


Figure 2.1: PatrolBot research

Manufacturer	Adept Mobile Robots
Model	PatrolBot research
Robot Weight	45 kg
Power	24 V @ 4 A
Max. Forward/Backward Speed	1.8 m/s
Dimensions	59 × 48 × 38 cm
Nominal Voltage	24 V
Interface	Serial

Table 2.1: Technical specifications.

2.1.2 Asus Xtion Pro Live

The Asus Xtion Pro Live [6], has been designed for motion-sensing development solution. It comes with an RGB image, an infrared and audio sensors that can interact to produce depth images, gesture detection, whole body detection among others. Figure 2.2 shows the sensor whereas the Table 2.2 shows the technical specifications.

Figure 2.2: The Asus Xtion Pro Live

Manufacturer	Asus
Model	Xtion Pro Live
Depth Image Size	VGA (640x480) : 30 fps
Power	Below 2.5W
Distance of Use	Between 0.8m and 3.5m
Dimensions	18 × 3.5 × 5 cm
Operation Environment	OpenNI
Interface	USB2.0/ 3.0
Software	OpenNI
Field of View	58^o H, 45^o V, 70^o D

Table 2.2: Technical specifications.

2.1.3 Hokuyo

The laser hokuyo UTM-30LX [7] is an accurate, high-speed device for robotic detection applications. It has a wide field of view of 270^o and a housing and due to the size makes the sensor suitable for indoor and outdoor application. Figure 2.3 shows the sensor whereas the Table 2.3 shows the technical specifications.

Figure 2.3: The Hokuyo UTM-30LX Scanning Laser Rangefinder.

Manufacturer	Hokuyo
Model	UTM-30LX
Range	0.1 → 30.0 m
Power	8.4 Watts (0.7A at 12V)
scanning time	25msec
Dimensions	6 × 6 × 8.7 cm
scanning range area	270^o
Interface	USB2.0
accuracy	± 0.3m

Table 2.3: Technical specifications.

2.2 System Software

The robot operating system (ROS) [8] is selected to achieve the different issues in this case study.

Chapter 3

System and Simulation Results

3.1 Hardware User Guide

The system in turn is presented in figure 3.1 and mainly consists of the following:

- Lenovo ThinkPad L540 + Intel(R) Core(TM) i7-4712MQ CPU @ 2.30GHz running Ubuntu 14.04.5 LTS.
- PatrolBot research [5].
- Hokuyo UTM-30LX Scanning Laser Rangefinder [6].
- Asus Xtion Pro Live [7].

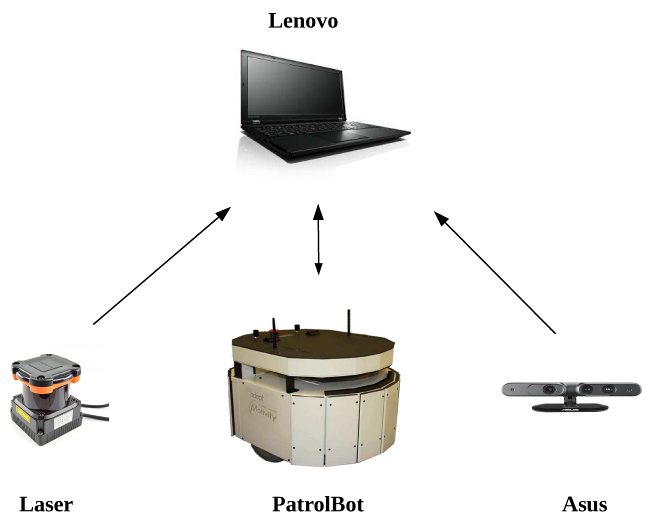


Figure 3.1: The Lenovo laptop communicates over a USB2 cable to the laser and the asus sensors respectively. The PatrolBot in turn communicates with the Lenovo over a serial cable.

During the functioning, the laptop communicates to the PatrolBot over a USB to Serial cable and viceversa. Moreover, the laser and asus readings are send to the laptop over a USB cable.

3.2 Software User Guide

The issue about the software is tackled directly by explaining the launch files that runs the application in the PatrolBot. They are called **patbotsd_navigation.launch** and **patbotsd_directions_navigation.launch**. These launch files are found under **patbot_self_driving/patbotsd/launch/**. And, in order to launch the application, one must type the following commands in a terminator terminal **roslaunch patbotsd patbotsd_navigation.launch** or

roslaunch patbotsd patbotsd_directions_navigation.launch. These launch files run the following packages and launch files:

- **patbotsd_slam_gmapping.launch**, a launch file that runs the gmapping package which in turn launches the laser based on SLAM package that creates a 2D occupancy grid map, [9].
- **patbotsd_asus.launch**, a launch file that launches the **openni.launch** file that runs the asus driver. Also, it launches the **pointcloud_to_laserscan** file that converts the **point_cloud** asus readings to **laserscan** data, [10, 11].
- **rosaria**, a package that runs the **RosAria** node that provides a ROS interface for PatrolBot, [12].
- **hokuyo_node**, a package that runs the **hokuyo_node** that provides a ROS interface for the hokuyo laser sensor, [13].
- **patbotsd**, a package that runs the following nodes:
  - **patbotsd_filter_hokuyo** node that has to do with the filtering off some laser readings that are produced by some obstacles in the robot.
  - **patbotsd_laser_asus** node that has to do with the combination of 2D laser and 2D depth asus readings.
  - **patbotsd_map_maker_laser_asus_rotation** node that has to do with the self driving of the PatBot in the empty space of the indoor environment while making a 2D map based on the laser and the camera_depth_points readings.
  - **patbotsd_map_maker_laser_asus_directions** node that has to do also with the safe driving of the robot on the empty space of the area.
  - **patbotsd_map_maker_laser_asus_range_cones** node that has to do also with the safe driving of the robot on the empty space of the area.
  - **patbotsd_asus_groundfloor_remove** node that has to do with the removing of the ground_floor of the 3D "*/camera/depth/points*" PointCloud2 asus topic.

Once the launch file **roslaunch patbotsd patbotsd_navigation.launch** is launched, it runs two packages **joint_state_publisher** and **robot_state_publisher** which in turn runs the **joint_state_publisher** and **robot_state_publisher** nodes respectively. These nodes have to do with the publishing of all the join states and their transform tree (tf) of the states of the robot. Then, it establishes communication between the laptop-PatrolBot, laptop-laser and laptop-asus by running the **RosAria**, **hokuyo_node** nodes and the **openni.launch** launch file. Moreover, the **patbotsd_filter_hokuyo** node takes as an input the /**scan** topic from the hokuyo laser and gives as an output the /**scan_filter** topic which contain the filtered /**scan** ranges. Furthermore, the **patbotsd_asus_groundfloor_remove** node takes as an input the the /**camera**/**depth**/**points** topic that contains the PointCloud2 from the asus sensor and gives as output the /**PC2_asus_cut_image** topic which contains the PointCloud2 with the ground floor removed. Then, the **pointcloud_to_laserscan** node takes as an input the /**PC2_asus_cut_image** topic and delivers the /**scan_xtion** topic which contain a LaserScan message.

The **patbotsd_laser_asus** node has as an input the /**scan_filter** and the /**scan_xtion** topics which are synchronized and combined in a single LaserScan message and published in the /**scan_laser_asus** topic. Then, the **patbotsd_map_maker** node takes as an input the /**scan_laser_asus** topic and drives the PatrolBot in the empty space of the indoor environment while the **gmapping_node** is making a map on a 2D grid. Figure 3.2 shows a screenshot where the yellow marks are the combined 2D-laser with the 2D-asus-depth readings, the 3D blue points represent the asus PointCloud2 message, the red cubes represent the free space in front of the robot, the green cube represents the middle point of the free space in front of the robot and the black surface represents the obstacles created by the **gmapping_node**.

3.3 Technical Instructions

- Turn on the PatrolBot by pressing the green 'PWR ON' button as it is shown in Figure 3.3(a).

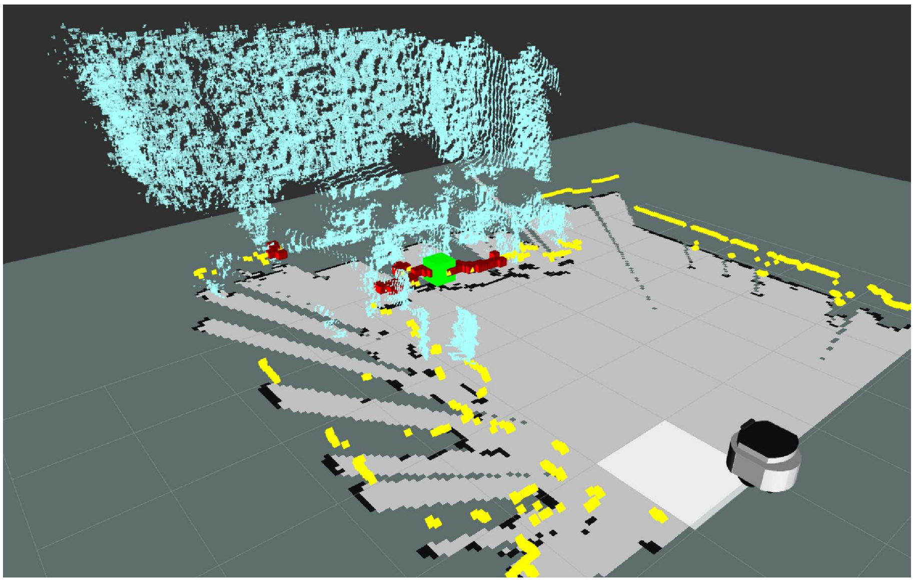


Figure 3.2: The RVIZ screenshot.

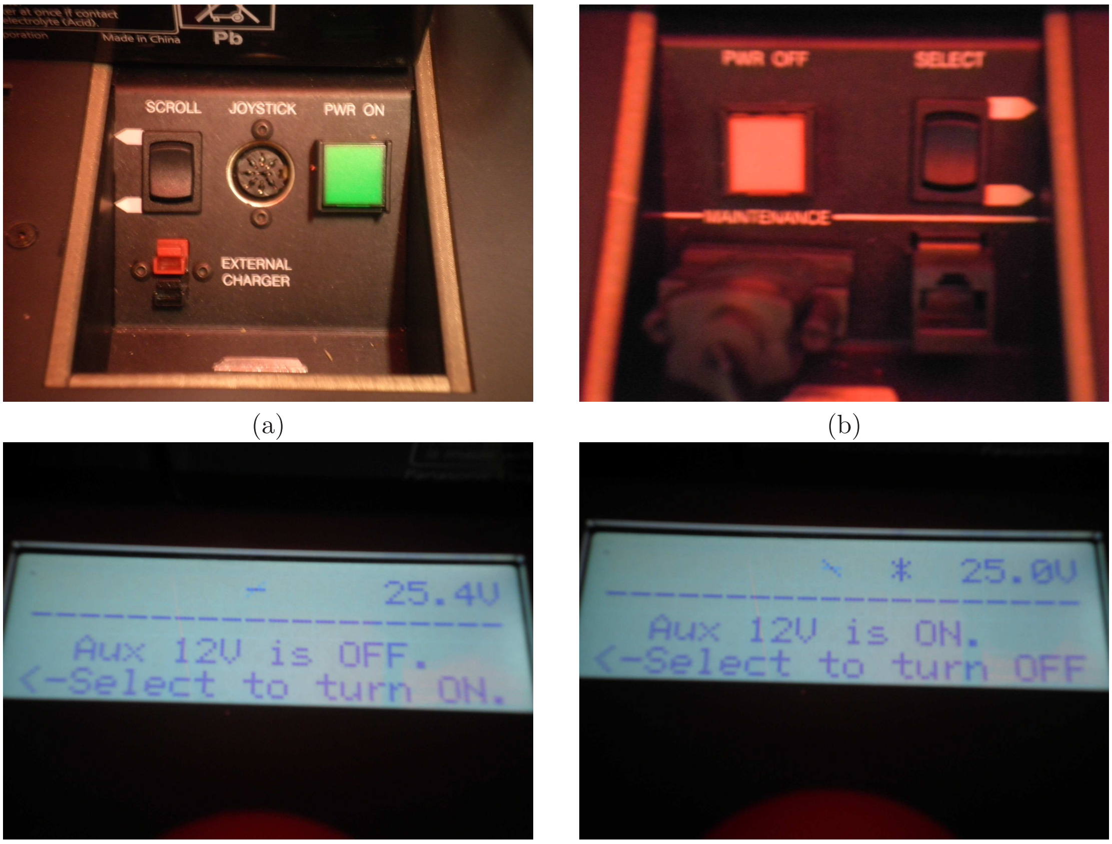
- To power the laser hokuyo utm-30lx which is situated inside the PatrolBot, scroll up or down the SCROLL rocker switch to get the "Aux 12V is OFF" status message in the interactive LCD window as it is shown in Figure 3.3(a) and 3.3(c) respectively. Then, press the SELECT rocker switch to power the laser as depicted in Figure 3.3(b), then the status message in the LCD window will change to "Aux 12V is ON" as it is shown in Figure 3.3(d).
- Connect the hokuyo laser to a USB port on the laptop's side. Then, make sure that the laser is working properly by launching the command **rosrun hokuyo_node hokuyo_node**. Sometimes, the previous rosrun command does not launch the ROS-interface at the first attempt, so one must try few times till the ROS-interface is launched.
- Connect the asus xtion pro depth sensor to a USB port on the laptop's side. Then, make sure that the asus is working properly by launching the command **roslaunch openni_launch openni.launch**.
- Connect the PatrolBot to a USB port on the laptop's side by using a serial to USB converter cable.
- A good practice is to check whether the Patrolbot serial port and the USB laser have the right permissions. One can give the right permissions by the following commands **sudo chmod a+wr /dev/ttyUB0** and **sudo chomod a+wr /dev/ttyACM0**.
- Once all the communications between the hokuyo laser, the asus depth sensor and the PatrolBot are established with the laptop, either the commands **roslaunch patbotsd patbotsd_navigation.launch** or **roslaunch patbotsd patbotsd_directions_navigation.launch** can be launched.

3.4 Simulation Results

Broadly speaking, the system operates in two different modes:

Mode_1: The main idea is to let the PatrolBot to navigate forward till an obstacle boundary is reached, then the mobile robot (MR) turns to the left with a small angle till it finds space to move forward. It keeps rotating and moving forward till a full rotation takes place. After that, it starts turning to the right and moving forward till a full rotation also takes place. This process repeats constantly and by doing so, the MR covers all the empty space of the area. The **mode_1** can be launched by the file **roslaunch patbotsd patbotsd_navigation.launch**. This mode is suitable for an open area where the robot can move freely in the middle of the space. Figure 3.4(a) shows the real laboratory environment whereas Figure 3.4(b) depicts the RVIZ mapping together with the PatrolBot.

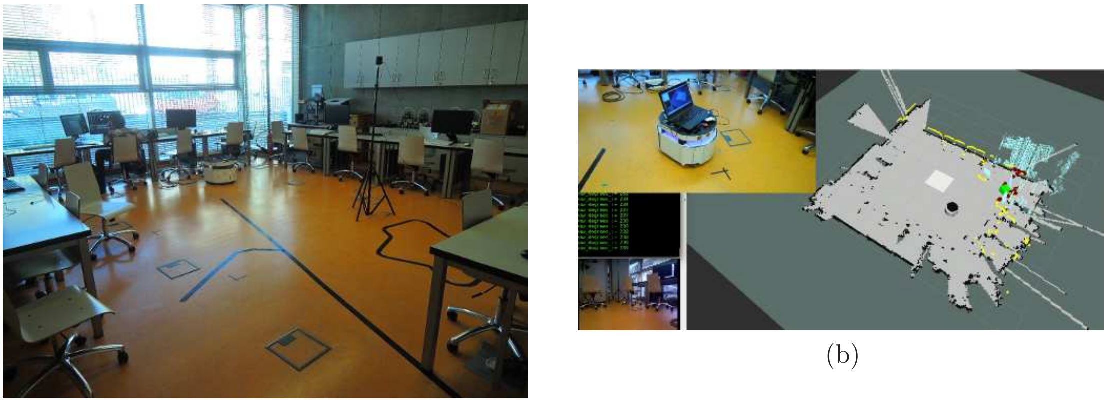
Mode_2: The main idea is to place range cones along the range laser which scans the front part of the robot by 180 degrees. To this end, a total of 5 range cones were placed and distributed in front of the



(a) (b)

(c) (d)

Figure 3.3: (a) The PWR ON bottom and the SCROLL rocker switch . (b) The SELECT rocker switch. (c) The LCD window showing Aux 12V is OFF. (d) The LDC window showing Aux 12V is ON.



(a) (b)

Figure 3.4: (a) Real environment. (b) RVIZ mapping together with the PatrolBot.

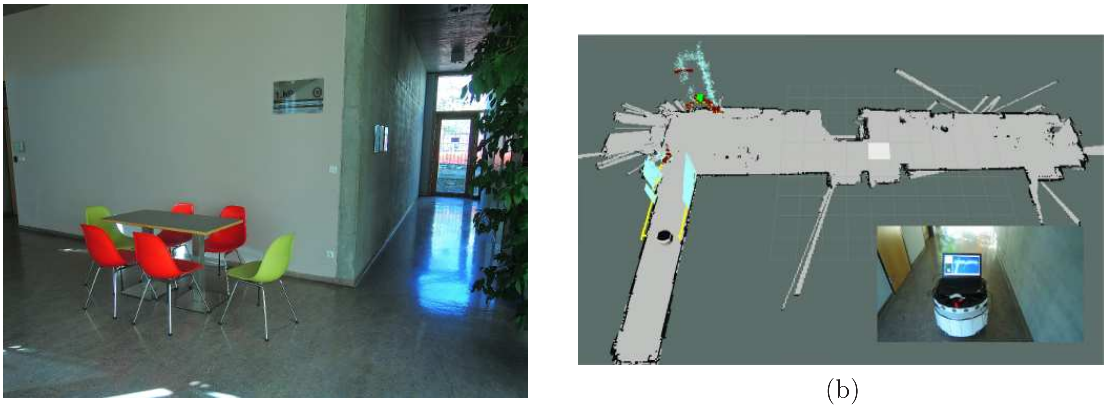
robot. Then the aim is to find whether they are empty to allow the robot to navigate. The **mode_2** is launched by the file **roslaunch patbotsd patbotsd_directions_navigation.launch**. This mode was tested in a bit cluttered corridor and with some people around. where the MR was navigating successfully in the empty area, just there were situations were the robot did not navigate in some areas. The result of this mode can be seen in the Figure 3.5.

The **Mode_2:** was modified by adding two more range cones along the range laser making a total of 7 ones. The source code can be found in the node **patbotsd_map_maker_laser_asus_range_cones**. The result of this simulation can be depicted in Figure 3.6. It can clearly be seen that the MR is able to navigate in areas wehere the **Mode_2:** with 5 range cones is not able to do it.

Mode_2: with 7 range cones was also tested on a more open clutter area with people walking ramdoly.

(a) (b)

Figure 3.5: (a) Real environment. (b) The figure shows the 2D map of an indoor environment after the PatrolBot has been navigating autonomously on the empty space of a clutter corridor.



(a) (b)

Figure 3.6: (a) Real environment. (b) The figure shows the 2D map of an indoor environment after the PatrolBot has been navigating autonomously on the empty space of a clutter corridor.

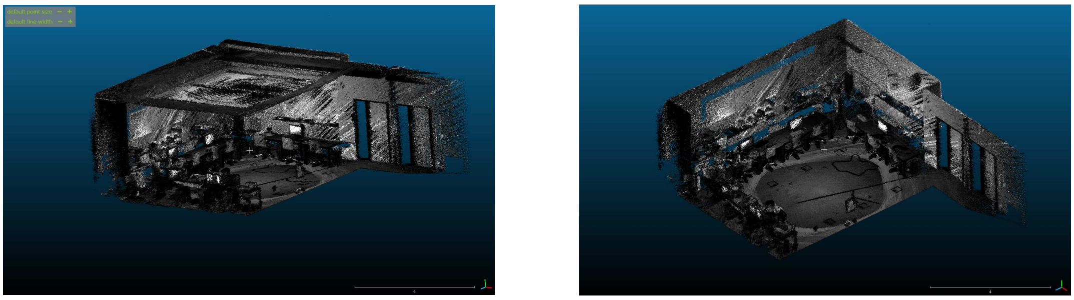
The results of the simulation are presented in Figure 3.7.

(a) (b)

Figure 3.7: (a) Real environment. (b) The figure shows the 2D map of an indoor environment after the PatrolBot has been navigating autonomously on the empty space of a clutter area.

In order to extend the functionality of the application, a mapping bagpack solution was mounted on the robot as it is depicted in Figure 3.8. This solution is novel a way of odometry estimation from velodyne lidar point cloud scans [14]. And, it is able to make a 3D map of the environment by means of two velodyne lidars which are placed on the top with a certain inclination to cover the whole area. Figures 3.9(a) and 3.9(b) shows the 3D reconstruction of the laboratory.

Figure 3.8: A bagpack solution for 3D reconstruction based on two velodyne lidars placed on the PatrolBot.



(a) (b)

Figure 3.9: (a) 3D reconstruction of the laboratory. (b) 3D reconstruction of the laboratory.

For more information about the ROS package of the 3D bagpack refer to [15]. Also, a Github repository of the previous patbotsd ROS package was created and can be seen on the site [16].

Chapter 4

Conclusion

The simulation results have shown the feasibility of combining 2D laser scan readings with projected 3D depth image points into 2D. This combination is used for constructing a 2D map of an indoor environment while the robot, in this case the PatrolBot, is navigating in autonomous mode on the empty space of the environment. The system has been run in two modes: The mode_1 has shown the viability of running the robot in open areas where the robot can freely ran in the middle of the empty space. Whereas, the mode_2 has shown the practicality of running the robot in more clutter spaces, like corridors or passages. A 3D pointcloud map was obtained from a bagpack solution which is based on two velodyne lidars. The bagpack has been placed on the PatrolBot while it has been navigating on the empty space of the indoor environment.

Bibliography

[1] Z. Riaz, A. Pervez, M. Ahmer, and J. Iqbal, "A fully autonomous indoor mobile robot using slam," in *Information and Emerging Technologies (ICIET), 2010 International Conference on Information and Emerging Technologies*, 2010.

[2] C. Wang, L. Meng, S. She, I. M. Mitchell, T. Li, F. Tung, W. Wan, M. Q. Meng, and C. W. de Silva, "Autonomous mobile robot navigation in uneven and unstructured indoor environments," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2017, Vancouver, BC, Canada, September 24-28, 2017*, 2017, pp. 109–116.

[3] S. Thrun, W. Burgard, and D. Fox, "A real-time algorithm for mobile robot mapping with applications to multi-robot and 3d mapping," in *IEEE International Conference on Robotics and Automation, San Francisco, April 2000*, 2000.

[4] J. Biswas and M. Veloso, "Depth camera based indoor mobile robot localization and navigation," in *2012 IEEE International Conference on Robotics and Automation RiverCentre, Saint Paul, Minnesota, USA May 14-18, 2012*, 2012.

[5] A. M. Robots. (2012) PatrolBot research. [Online]. Available: http://www.mobilerobots.com/ResearchRobots/ResearchPatrolBot.aspx

[6] Hokuyo. UTM-30LX. [Online]. Available: https://www.robotshop.com/en/hokuyo-utm-03lx-laser-scanning-rangefinder.html

[7] Asus. Xtion Pro Live. [Online]. Available: https://www.asus.com/3D-Sensor/Xtion_PRO_LIVE/

[8] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. B. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: an open-source robot operating system," in *ICRA Workshop on Open Source Software*, 2009.

[9] ROS. gmapping. [Online]. Available: http://wiki.ros.org/gmapping

[10] ——. openni_launch. [Online]. Available: http://wiki.ros.org/openni_launch

[11] ——. pointcloud_to_laserscan. [Online]. Available: http://wiki.ros.org/pointcloud_to_laserscan

[12] ——. RosAria_node. [Online]. Available: http://wiki.ros.org/ROSARIA

[13] ——. hokuyo_node. [Online]. Available: http://wiki.ros.org/hokuyo_node

[14] M. Velas, M. Spanel, and A. Herout, "Collar line segments for fast odometry estimation from velodyne point clouds," in *IEEE International Conference on Robotics and Automation (ICRA) Stockholm, Sweden*, 2016.

[15] Github. but_velodine_lib. [Online]. Available: https://github.com/robofit/but_velodyne_lib

[16] ——. PatBot_Self_Driving. [Online]. Available: https://github.com/robofit/patbot_self_driving

Acknowledgement