Proceedings of the 23rd International Conference on Automation & Computing, University of Huddersfield, Huddersfield, UK, 7-8 September 2017

An Asus_xtion_probased indoor MAPPING using a Raspberry Pi with Turtlebot Robot Turtlebot Robot

Hamza Aagela IEEE studet Member, Maha Al-Nesf, Violeta Holmes
Computing and Engineering
University of Huddersfield
United Kingdom
Hamza.aagela@hud.ac.uk, U1672789@unimail.hud.ac.uk,V.Holmes@hud.ac.uk

Abstract—**The developers of path planning algorithms and localization have significantly improved the usability of the robot those days. By using software such as a Gmapping, the robot will be able to create a map of the surrounding area. This research utilizes the 3D sensor Asus_xtion_ pro to create an indoor map using SLAM and create 3D models for surrounding objects with a Turtlebot robot. In the first case, we used the Turtlebot to generate an indoor map of the robotic lab room using the Gmapping ROS packet. In the second case, we used the robot to create 3D models for the surrounding objects in the room. We used the Raspberry Pi 3 as a replacement of the laptop that was used to control the Turtlebot. The same implementation of the first and second tasks have been repeated to compare the performance. The Raspberry Pi accomplishes the given tasks successfully; however, there is some delay due to the different on the CPU power. Finally, the low cost proposed solution is capable of running ROS based SLAM algorithm and using the point on cloud to create 3D models. In addition, the use of Raspberry Pi allows the robot save considerable amount of power in contrast with the use of a normal laptop.**

Keywords-component; Raspberry Pi; Asus_Xtion_pro; mapping; 3D; Turtlebot II; ROS; Robotic.

I. INTRODUCTION

Nowadays, the mobile robots become more popular that perceives a great improvement in technology. However, many obstacles prevent the robots to be used widely in our daily activity, such as robot cost and hardware and software performance in operating tasks, such as mapping, localization and path planning. The mapping is the area of robotics research, which focus on giving the robot's ability to learn about the surrounding area and it is essential for mobile robots in order to be able to perform the localization and the path planning [1] [3]. Therefore, creating a virtual environment for the surrounding area of the robot that could be used as a map would be vital task, which is needed to be used by another application such as, localization and path planning that could be useful in operations like military, search and rescue, smart cities and so forth.

In relation to this study, various researchers [16][17][18] have used a Raspberry Pi to control low cost robots for various purposes.

Sending the robot into rough missions and exploring unknown places that could be dangerous for humans to explore means that the robot itself will be in danger, therefore, the cost of the robot that could perform such an operation will be a vital factor [1]. Thus, decreasing the cost of the hardware and the software for the robot is desirable. Mapping an Indoor environment will require a laser scan, a mobile robot base and compute unit with wireless communication. So, the robot will gather the depth information by the **Asus_Xtion_pro** 3D sensor as stated in various implementations [3], for instance, creating a map, or making a 3D model of an indoor environment. Localizing and mapping within indoor environments is actually vital for remote-controlled robots. There are several approaches for the mapping that have been researched, including utilizing laser scanners, 3D vision and wireless strength, [3] [8].

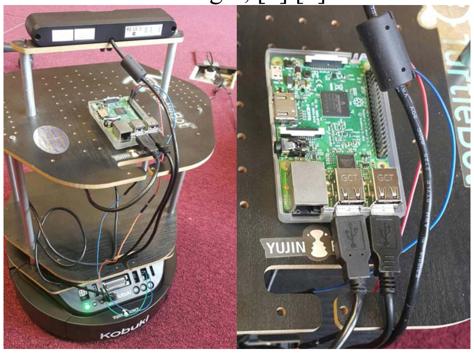
Fig 1. Utilizing a Raspberry Pi as a main computer unit for Turtlebot II

The objective of this paper is to integrate the Raspberry Pi 3 with the Turtlebot II robot as an alternative of the laptop. As shown in Fig 1, in order to establish a cost-effective robot platform, which could create a map for a new environment by utilizing low cost equipment while maintaining the performance in accomplishing the tasks, whilst in addition reducing the power consumption for the robot platform.

This paper is structured as follows. Section II discusses the system specification, section III gives information about the SLAM procedure, and section IV describes the experimental setup and the results. Finally, Section VI gives conclusions and suggests future work.

II. SYSTEM SPECIFICATION

This approach was planned to offer an affordable operational indoor mapping system that can be used over the network, particularly in environments that could be considered as a danger for humans. Alhough there are several existing systems that are able to do the mapping and planning task indoors, our solution becomes very handy when the risk of losing the robot is very high.

A. Robot Operating System (ROS)

It is an open-source platform that provides a number of packages and software that provides the robot functions [9]. In addition, it controls the commutation between the robots and the controller, where each device becomes a node within the ROS environment. The robot sensor data that are called topics, all the communications are managed by a master node, which is responsible for the interconnection protocol between the nodes, by using the sensors and the actuator as topics that would be available to subscribe or to publish, as Fig. 2 shows. The messages contain various kinds of data such as video and odometry data that's transferred between the nodes [6].

ROS support manly Linux Operating system such as Ubuntu and Debian, however, ROS hydro works with several other operating systems such as, Android, Windows and so on. ROS supports a wide range of hardware, sensors and actuators. It defines the hardware architecture as well as a low level of control to manage the linked devices, whether it be a robot or an individual sensor linked to a computer or smartphone.

The time between the ROS nodes is a key factor in the mapping process, which needs to be constantly synchronized between the nodes (robots and computers). Whereas the sensor data and the control command move between the nodes that need to be coordinated, ROS uses software called Chrony to manage the Network Time Protocol (NTP) [1].

ROS has flexibility to work with existing algorithms that can be utilized for improving the robots performing complex tasks. SLAM is an example that is used for robot mapping and navigation [3].

In order to run the ROS packages, at least one ROS master should be active in the network as Fig. 1 shows. The simple ROS connection system allows the ROS node to send and receive messages from other nodes. Once the node has been registered on the master the message could be sent directly between the nodes [6].

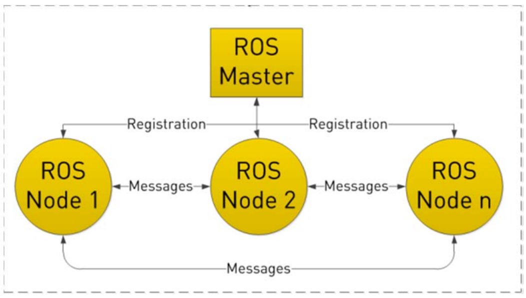

Fig 2. The ROS system [6].

Since 2010, ROS has developed and grown to be the main robot software manager for a number of robots, including the latest version named ROS Lunar. However, in this study, we used ROS Indigo, because it has all essential libraries to support the Turtlebot II..

B. Turtlebot II

Turtlebot II is a development of the initial Turtlebot, which is an open hardware developed by Willow Garage [5] and the robotic platform runs on a base of motorized wheels. The Odometry can be obtained by using a single sensor called Asus_xtion_pro sensor that is mounted on the robot in order to produce 3D images and depth image information.

Figure. 3. The Turtlebot 2 robots.

There are several features associated with this robot, such as low-cost wheel encoders, an integrated gyroscope, big batteries, bump sensors, cliff sensors as well as wheel drop sensors. It possesses an impressive speed of about 50cm/s, and is powered by a lithium battery, which also powers the Asus_xtion_pro sensor connected to it.

C. Raspberry Pi

A single board computer has most of the computer components built-in a small unit. The Raspberry Pi uses an ARM processor 1.2 GHz for the Raspberry Pi 3 [4], which is used in this research. The Raspberry Pi 3 runs on Ubuntu 16.04 Mate Operating System that has been installed in a 32 GB Micro SD card. Moreover, ROS has been installed with its basic packages. The Raspberry Pi 3 connects to the Turtlebot base and the Asus_xtion_pro Sensor via USB cables, the Asus_xtion_pro being mounted on the middle of the robot. The Raspberry Pi is responsible for collecting the sensor's data namely:

1. Depth image received from the Asus_xtion_pro sensor.
2. Images and video stream received from the Asus xtion pro.
3. Odometry information received from the robot.

For a Raspberry Pi 3 that has ROS installed on it, we give a description of the application or the handlers that are running on it:

1. The Turtlebot handler, also referred to as the hardware driver would be main driver that is used in the connection of the various components that are found within the Turtlebot, and this connection is done by the use of a USB cable.
2. The OpenNI handler is the driver that is used in the connection of the Asus_xtion_pro to the Raspberry Pi 3, therefore, as to be able to collect video data, in-depth information or even streaming audio files.
3. The Depth_to_laserscan application refers to the application that has access to every bit of the Asus_xtion_pro's depth information, which occurs in a horizontal line of sight. It also exploits the use of laser scan data, which is later used within the application that is the main area for detecting and for the building of the grid map.
4. The Gmapping application is referred to as the main ROS application that has been tasked with the

processing of all the information within Odmetry or laser scan data, in order to build a grid map on the map sever [11].

D. Master PC

This personal computer runs on the Ubuntu 16.04 operating system and ROS desktop installed on it, which comes with various software that makes it possible to view the maps, as well as be able to view the video streams by the use of the following components:

1. The main application runs on the ROS master node. This ROS core Application communicates with the ROS master node, in order to be able to access all of the nodes that are available in the ROS environment. This makes it main database where all of the nodes are listed as well as the path to access them.
2. The application that is applied to the connection of the ROS system and the sending of the control commands to the robot to be able to control its movements is known as the tele-operating application. Its operation is guided by the use of a keyboard or a joystick that is connected to the corresponding PC.
3. The 3D visualizer that is used for displaying both the sensor data and the state information from the ROS is what is referred to as the RVIZ. This is software developed by ROS community and works as a visualizer [7]. It is used to view the robot model, the video stream, the 3D model and the created map. In addition, the software is able to deal directly with robot sensor topics such as infrared distance measurements, camera data, and sonar data, amongst others as shown in Fig. 3.
4. The Map server refers to the application that is used in the saving of the grid maps that have been currently built into a local file within the workstation.

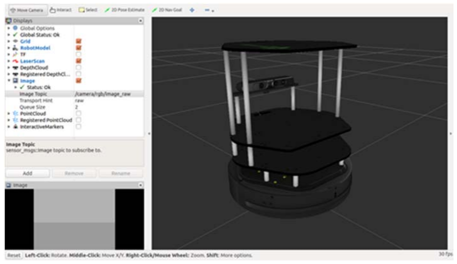

Fig 3. The RVIZ visualizer.

E. The Network:

Fig. 4 shows the network architecture used in the experiment. The robot is connected to the master node via a local wireless-G network with a bandwidth of 54 Mbit/s in Wi-Fi connection [2].

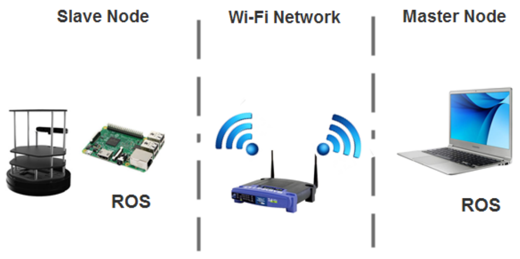

Fig 4. Workflow for the mapping process

In order to configure a ROS environment over our network, we provided the master node PC and the Raspberry Pi with IP addresses as well as the hostnames that are considered very important for ROS communication systems.

To make the Raspberry Pi mobile it needs to connect to a wireless network, which uses a router that connects it to the main master node. This connection is kept intact as long as the router is able to reach the Raspberry Pi.

The other important factor when it comes to a network would be latency. Given that the operation of the robot is done on a remote base, the failure for it to receive the right commands and on time might mean that the robot could be faced with obstacles since there is the possibility of the control commands reaching the Raspberry Pi late. Latencies of up to 400ms are typically found to be acceptable.

III. SIMULTANEOUS LOCALIZATION AND MAPPING

The simultaneous localization and the mapping (SLAM) within the Robots entails complex computational problems that are focused on dealing with building a map of an unknown environment, which is able to track down the location of the robot in the same map [3].

Normally, the depth information that is normally received through the 3D sensor, in our case the Asus Question Pro sensor is converted into laser 12-format data on a real time basis, which is deemed as the initial data that is needed by the entire process. This allows the generation of the first local grid map as well as the first pose of the robot, where the middle of the map gives the provision for the initial location of the robot. An algorithm for the Gmapping process that is considered main begins its generation of the local grid maps at a very opportune moment within a limited region.

Although a grid map as may have been built previously, this is considered a continuous process even at times when the robot is steady.The laser scans received from the Asus_xtion_pro are produced at the rate of 30 depth images per second.

There is always a scan that matches the 14, 15 algorithm against the local grid map, which generates a set of what is referred to as the sampling points within an interval that is around the reported pose. The weight of the particle is normally calculated through the application of these points in the next interval.

In instances where the scans are not able to match for any reason, there is the substitution of the Odometry information as the main tool that estimates the pose of the robot. This normally leads to the same issues being inconsistent during the building of the map, as has been discussed under the Experimental results section.

We use the development system to run some experimental tests within our lab and compare the result with the same system that uses a laptop instead of the Raspberry Pi 3.

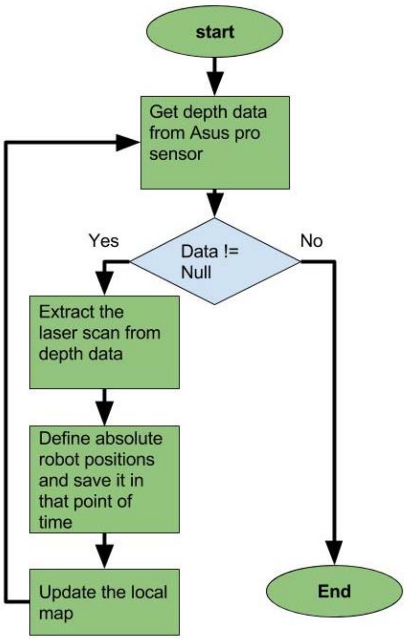
start
Get depth data from Asus pro sensor
Data != Null
Yes
No
Extract the laser scan from depth data
Define absolute robot positions and save it in that point of time
Update the local map
End

Fig 4. Workflow for the mapping

IV. Experiments

A. Case 1: Create a MAP for the lab

In this test, we created MAP by using Gmapping ROS software in the University of Huddersfield robotic lab as shown in Fig. 6., to map the floor layout of the lab. The robot moves around the room and scans it by using an Asus_xtion_pro. The scan is done while the robot is moving at a speed of 25cm/s forward, which is considered an average speed given that the Turtlebot robot is capable of a top speed of 50cm/s.

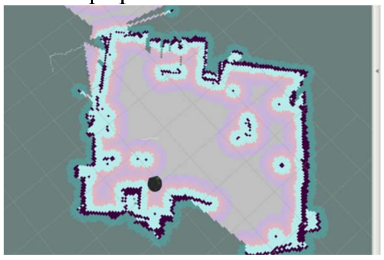

Fig 6. Map for a University of Huddersfield robotic lab using Raspberry Pi.

The Asus_xtion_pro has the ability to provide in-depth information using an IR beam. The beam is reflected from the facing wall. The power consumed by the Raspberry Pi 3 was negligible

In contrast, we repeated the same experiment using the laptop instead of the Raspberry Pi 3. As shown in Fig 7 the map was clearer in comparison to the map in Fig. 6.

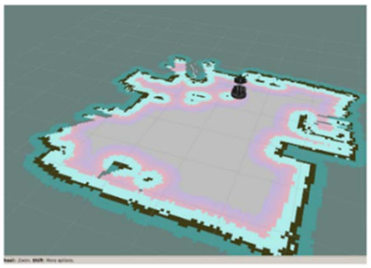

Fig 7. Map for a University of Huddersfield robotic lab using the laptop.

B. Case 2: Create a 3D model for the Objects

One of the processes that uses intensive processing power is the creation of a 3D model of the environment using a point cloud, which uses the depth image data collected by Asus_xtion_pro sensor. In this test, we tended to benchmark the network and processing performance for the Raspberry Pi in contrast to the laptop as shown in Fig 8.

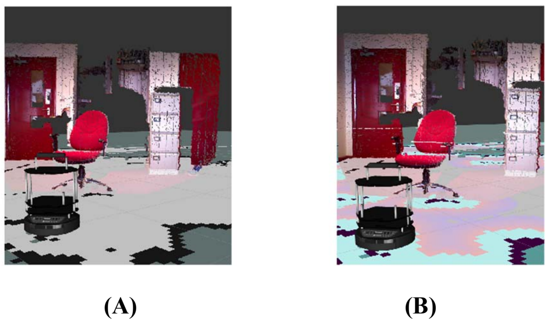

Fig 8. A) 3D model generated by Turtlebot with laptop B) 3D model generated by Turtlebot with Raspberry Pi 3.

Based on the observation from Fig 8 The quality of the 3D model generated by the Raspberry Pi was very close to the one that was generated by the laptop.

V. Result

The main outcome of this study is that utilizing a single board computer, such as the Raspberry Pi with the Turtlebot robot seems to be a successful model to create a low cost robot. In addition, it proved its ability to run such advanced robot tasks. The first task was an indoor map for an unknown environment. This task was a bit challenging for the Raspberry Pi, where the generated map had unclear edges for some of the room walls in contrast with the map generated by the laptop that had clear wall edges and object positions.

The mapping process depends on defining the location of the robot's previous position measured by the scanner sensor. This process can fail for a number of reasons, particularly when the robot starts rotating or moving with high speed, requiring high processing power to process the generated map grid. Therefore, the proposed solution will be sensitive to the robot speed and movement. It is suitable to run the SLAM process while moving the robot with slow speed to obtain a clear map as well as to create 3D models. Looking at the main components for the proposed system, we have the Raspberry Pi 3, the Turtlebot Robot, the Master PC and the network Modem Wi-Fi.

The Raspberry Pi has 40 GPIO pinout that could be utilized by the robot for several purposes such as send and receive data, control signal, and power control. This project utilized two power pins (5V and ground) to power up the Raspberry Pi.

The systems use a single charging point, with the Raspberry Pi powered from the kobuki base 5V socket, in contrast with the original system where the robot used a laptop that needed an additional 12 V charger.

The power that is consumed by the Raspberry Pi, contrasts with the power used by a laptop that has an Intel quad core I5 Processor. According to [10] the fact that such a computer requires an average of 48 ***Kilo***joule/day as in the figure below means that the laptop consumes about 20 times more than the Raspberry Pi as Fig 9, making the latter very cheap to maintain.

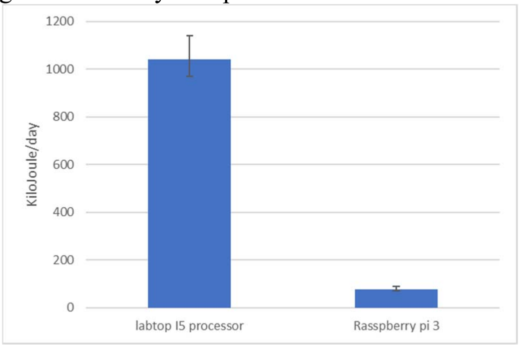

Fig 9. Power consumption in Kilo Joule for both Raspberry Pi 3 and Laptop Intel I5 Processor

The client node was using a Raspberry Pi 3 to control the Turtlebot II robot that has an ARM processor 1.2 with a 1GB RAM and a 32GB SD card. The Master node used is a PC with I5 processor with a 8 GB RAM and a 500GB hard disk. The Turtlebot robot has been utilized as mentioned earlier

VI. CONCLUSION

In conclusion, this research provides an affordable solution for utilizing the SLAM algorithm with ROS to generate 2D indoor MAPs as well as 3D models of the surrounding environment. The Raspberry Pi 3 proved its usability and compatibility with the Turtlebot II robot. In addition, the ROS provides most of the desirable robot functionality as ready-to-use packages and software. This concept will allow us to develop a cheap robot for a complex task, where the overall cost of the robot is a vital aspect that in the scanning and monitoring of unknown areas where the risk of losing the robot is high. The quality of the depth image received from the Asus_xtion_pro is accepted to create indoor maps, which can be shared or reused for motion planning. The new solution comes with the additional advantage that the power usage is far less than the normal system. Therefore, it will allow the robot to operate for a longer time. In addition, the robot now requires only one charging point. There are a number of limitations of this project as follows:

- The RAM on the Raspberry Pi was quite limited; therefore, we created a swappable RAM that uses the some of the SD card storage.
- The robot will require flat land to allow movement
- The performance of the robot with the Raspberry Pi 3 was slightly less than the same robot with the laptop when generating a MAP and creating the 3D model.

The single board computer is constantly improving, therefore the same test case is going to be repeated with an upcoming single board computer with more advanced features in terms of processing power and RAM, in order to overcome the current limitations in the solution developed. Using the concept of cloud robotics with this low cost robot is essential to overcome the lack of processing and storage in this design.

REFERENCES

[1] Hamzeh, Osama, and Ashraf Elnagar. "A Kinect-based indoor mobile robot localization." Mechatronics and its Applications (ISMA), 2015 10th International Symposium on. IEEE, 2015.

[2] Hamza Aagela, Violeta Holems, Mahmud Dhimish and Dave Wilson. "Impact of Video Streaming Quality on Bandwidth in Humanoid Robot NAO Connected to the Cloud". Data and Cloud Computing, ICC conference, 2017.

[3] Ghani, Muhammad Fahmi Abdul, Khairul Salleh Mohamed Sahari, and Loo Chu Kiong. "Improvement of the 2D SLAM system using Kinect sensor for indoor mapping." Soft Computing and Intelligent Systems (SCIS), 2014 Joint 7th International Conference on and Advanced Intelligent Systems (ISIS), 15th International Symposium on. IEEE, 2014.

[4] "Raspberry Pi 3 Model B - Raspberry Pi". Raspberry Pi. N.p., 2017. Web. 31 Mar. 2017.

[5] Claessens, Rik, Yannick Müller, and Benjamin Schnieders. "Graph-based Simultaneous Localization and Mapping on the TurtleBot platform". 2013.

[6] Robotics, Clearpath et al. "ROS 101: Intro To The Robot Operating System | Robohub". Robohub.org. N.p., 2017. Web. 15 Mar. 2017.

[7] "Rviz - ROS Wiki". Wiki.ros.org. N.p., 2017. Web. 25 Mar. 2017.

[8] Rehman, Umair. Using Robots and SLAM for Indoor Wi-Fi Mapping in Indoor Geolocation. Diss. Worcester Polytechnic Institute, 2013.

[9] Quigley, Morgan, et al. "ROS: an open-source Robot Operating System." ICRA workshop on open source software. Vol. 3. No. 3.2. 2009.

[10] Anwaar, W. & Shah, M.A. Energy Efficient Computing: A Comparison of Raspberry PI with Modern Devices. International Journal of Computer and Information Technology , 04 (02), 2015.

[11] Afanasyev, Ilya, Artur Sagitov, and Evgeni Magid. "ROS-based SLAM for a Gazebo-simulated mobile robot in image-based 3D model of indoor environment." International Conference on Advanced Concepts for Intelligent Vision Systems. Springer International Publishing, 2015.

[12] Carraro, Marco, et al. "An Open Source Robotic Platform for Ambient Assisted Living." AIRO@ AI* IA. 2015.

[13] Voisan, Emil-Ioan, et al. "ROS-based robot navigation and human interaction in indoor environment." Applied Computational Intelligence and Informatics (SACI), 2015 IEEE 10th Jubilee International Symposium on. IEEE, 2015.

[14] "Xtion PRO | 3D Sensor | ASUS Global". ASUS Global. N.p., 2017. Web. 12 Mar. 2017.

[15] Wong, B. . Cooperation leads to smarter robots: ROS, the robot operating system, is enabling developers to combine the advantages of new sensors and algorithms for more interactive functionality Penton Media, Inc., Penton Business Media, Inc. and their subsidiaries. 2011

[16] Pereira, Viren, Vandyk Amsdem Fernandes, and Junieta Sequeira. "Low cost object sorting robotic arm using Raspberry Pi." Global Humanitarian Technology Conference-South Asia Satellite (GHTC-SAS), 2014 IEEE. IEEE, 2014.

[17] Premkumar, Keerthi, and K. Gerard Joe Nigel. "Smart phone based robotic arm control using Raspberry Pi, android and Wi-Fi." Innovations in Information, Embedded and Communication Systems (ICIIECS), 2015 International Conference on. IEEE, 2015.

[18] Horak, K., and L. Zalud. "Image Processing on Raspberry Pi for Mobile Robotics." International Journal of Signal Processing Systems 4.2 (2016): 1-5.