The Ciência & Engenharia - Science & Engineering Journal
ISSN: 0103-944X
Volume 9 Issue 1, 2021
pp: 19 - 24

Autonomous Voice Controlled Robot

[1]Ch. Sai Divya, [2]D. Likhitha, [3]G. Sumanjali, [4]G. Kavya Renuka
[1,2,3,4] UG Student, Department of ECE,
Dr K V Subba Reddy College Of Engineering For Women, Kurnool, Andhra Pradesh, India

Abstract
TurtleBot is a low-cost personal robot kit with open-source software that is the focus of this project. Its goal is to implement a variety of autonomous navigation algorithms. You'll be able to build a robot that can drive around your house, see in three dimensions, and have enough power to make exciting applications with TurtleBot. In accordance with the REP 119 specification, the TurtleBot2 is the second generation. The definitions of compatibility for TurtleBot-compatible platforms are outlined in this REP. The TurtleBot is a specific product that is available under the FreeBSD Documentation License and is an Open Hardware Design. This document aims to distinguish between design elements that can be altered without affecting the core design's functionality and critical design elements that are required for compatibility. The purpose of this document is to serve as a blueprint for the creation of new, more affordable robots with more features that can still benefit from the TurtleBot community's development efforts. TurtleBot is a personal robotics kit that combines open-source software and hardware at a low cost. A mobile base, a grid sensor, on-board computation, wireless communication, and expansion space for adding new hardware are the TurtleBot's primary hardware features

1. Introduction

In the past, investigation into the development of unmanned air, underwater and land vehicles has been fundamentally the domain of military related organizations. Nowadays, the technological context, availability of precise sensors, the spread of open- source software and the increasing of computation power, has led the largest companies to take an interest on the concept of automation and robotization and as a result autonomous navigation has become also one of hottest topics in the research‟s field.
In this thesis, we study the problem of autonomous navigation through an environment that is initially unknown, with the objective of reaching the farthest point in which the robot can move avoiding the obstacles. Without prior knowledge of the map, a moving robot must recognize its surroundings through onboard sensors and make instantaneous decisions to react to obstacles as they come into view. This problem lies at the intersection of several areas of robotics, including motion planning, perception, and exploration.
Different techniques could be used to implement the navigation that is generally separated into global motion planning and local motion control. The algorithms introduced in this work are linked to the local motion planning; therefore, using the sensor mounted on it, the robot is capable of avoiding the obstacles by moving toward the free area.

This document explains three possible algorithm solutions, based on Obstacle Avoidance, that address a complete autonomous navigation in an unstructured indoor environment.

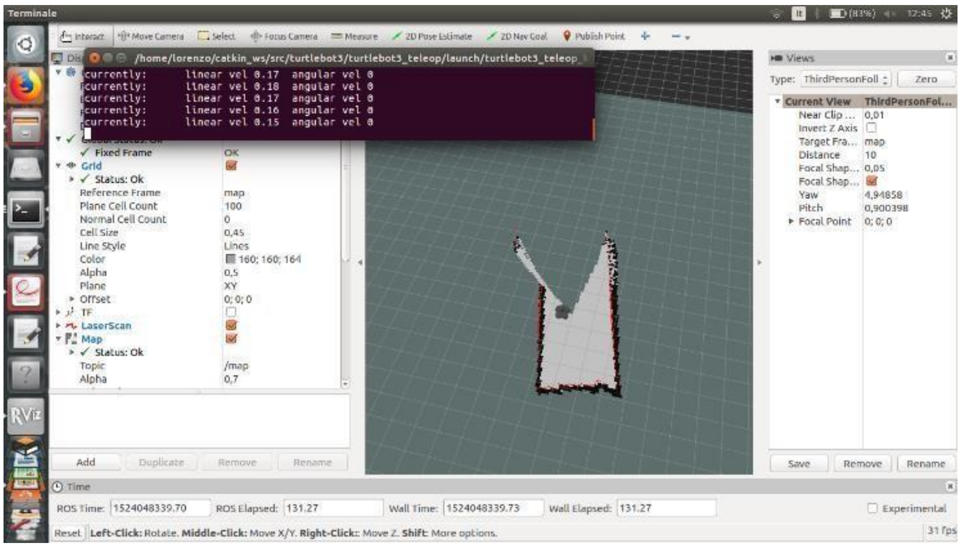
The algorithms raise in complexity taking into consideration the evolution and the possible changed in which the robot will have to move, and all are tested on the TurtleBot3 robot (Waffle and Burger), where only LiDAR was used as sensor.

The implemented techniques necessitate the robot to select actions based on the construction of the environment that it has perceived. As we will observe in this thesis, standard motion planning techniques often limit performance to be conservative when deployed in unknown environments, where every unexplored region of the map may, in the worst case, poses hazard

The trajectory and the speed of the robot depend on many factors such as the type of floor, the limitations of the hardware, the size and the material of the wheels and the type of algorithm that manages the movement of the robot.

The map is built with two-dimensional Cartesian histogram grid based on the RViz software that is the official software used in ROS environment, which is updated continuously with range data sampled by onboard sensors.

In order to make this work more complete a different solution to the automatic creation of the map, has been proposed; this map will be analyzed and compared with the one created by the ROS tool. In the field of robotics, platforms are of increasing importance. A platform is divided into a software platform and hardware platform. A robot software platform contains tools that are used to build robot application programs such as low-level device control, SLAM (Simultaneous Localization and Mapping), navigation, manipulation, recognition of objects or humans, sensing and package management, debugging and development tools especially in the industry, within which they are nowadays mostly used. Robot hardware platforms not only study platforms such as mobile robots, drones, and humanoids, but also commercial products.



```
/home/lorenzo/catkin_ws/src/turtlebot3/turtlebot3_teleop/launch/turtlebot3_teleop_
currently:      linear vel 0.17  angular vel 0
currently:      linear vel 0.18  angular vel 0
currently:      linear vel 0.17  angular vel 0
currently:      linear vel 0.16  angular vel 0
currently:      linear vel 0.15  angular vel 0
```

```
  ✓ Fixed Frame           OK
▾ Grid                    ✓
  ▸ ✓ Status: Ok
    Reference Frame       map
    Plane Cell Count      100
    Normal Cell Count     0
    Cell Size             0,45
    Line Style            Lines
    Color                 160; 160; 164
    Alpha                 0,5
    Plane                 XY
  ▸ Offset                0; 0; 0
▸ TF                      ☐
▸ LaserScan               ✓
▾ Map                     ✓
  ▸ ✓ Status: Ok
    Topic                 /map
    Alpha                 0,7
```

```
Type: ThirdPersonFoll
▾ Current View      ThirdPersonFol...
    Near Clip ...   0,01
    Invert Z Axis   ☐
    Target Fra...   map
    Distance        10
    Focal Shap...   0,05
    Focal Shap...   ✓
    Yaw             4,94858
    Pitch           0,900398
  ▸ Focal Point     0; 0; 0
```

```
ROS Time: 1524048339.70   ROS Elapsed: 131.27   Wall Time: 1524048339.73   Wall Elapsed: 131.27   ☐ Experimental
Left-Click: Rotate. Middle-Click: Move X/Y. Right-Click:: Move Z. Shift: More options.
```

Fig.1 Illustration of the Burger robot

2. Literature Review

Real robots need logistics including laboratory space, refreshing of batteries and operational quirks that often-become part of the institutional knowledge of the organization operating the robot. In a real case of work, even the best robots break periodically due to various combinations of operator errors, environmental conditions, manufacturing or design defects. These problems can be avoided by using simulated robots that move in a simulated environment.

Software robots are extraordinarily useful, in simulation we can model as much or as little reality as we desire. Sensors and actuators can be modeled as ideal devices, or they can incorporate various levels of distortion, errors and unpredicted faults.

The simulated robots and environment represent the ultimate low-cost platforms. The two-dimensional simultaneous localization and mapping (SLAM) problem was one of the greatest researched topics in the robotics community. Several 2D simulators were developed in response to the necessity for repeatable experiments as „Stage‟. Canonical laser range-finders and differential-drive robots were modeled, often using simple

kinematic models. These 2D simulators are very fast computationally and they are generally quite simple to interact with.Gazebo is a 3D simulator that provides robots, sensors, environment models for 3D simulation required for robot development, and offers realistic simulation with its physics engine. Gazebo is one of the most popular simulators for open source robotics in recent years and has been widely used in the field of robotics because of its high performance and reliability.

Gazebo uses OGRE (Open-source Graphics Rendering Engines) for the 3D Graphics, which is often used in games, not only for the robot model but also for the light that can be realistically drawn on the screen.

A lot of sensors are already supported Laser range finder (LRF), 2D/3D camera, depth camera, a contact sensor, force-torque sensor; noise can be considered as added to the sensor data like in real environment.

Some robot models are already available in gazebo: PR2, Pioneer2 DX, iRobot Create, and TurtleBot are already supported in the form of SDF, a Gazebo model file, and users can add their own robots with an SDF file.

Both GUI and CUI tools are supported to verify and control the simulation status.

The latest version of Gazebo is 8.0, and just five years ago, it was 1.9.

Although Stage and other 2D simulators are computationally efficient and excel at simulating planar navigation in office-like environments, it is important to note that planar navigation is only one aspect of robotics. Nonplanar motion, ranging from outdoor ground vehicles to underwater and space robotics is another aspect too. Three- dimensional simulation is necessary for software development in these environments

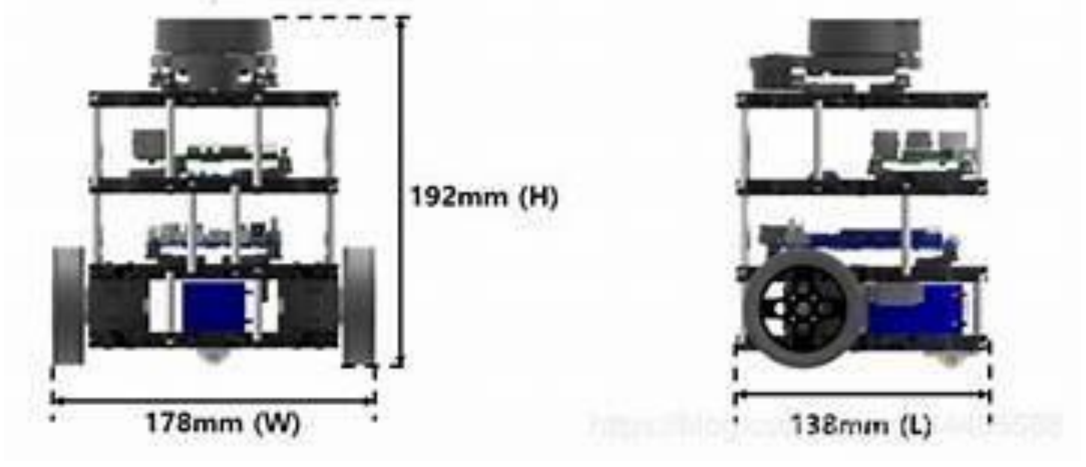


Fig.2 3D image of turtle bot3 Burger

3. Proposed System

Light Detection and Ranging (LiDAR), Laser Range Finder (LRF), and Laser Scanner are all components of the Laser Distance Sensors (LDS) category. The LDS sensor uses a laser as its source to determine the distance to an object. Since the LDS guarantees high performance, high speed, and real-time data acquisition, it is frequently used in all systems that require distance measurement and a wide range of robotics fields. It is one of the most important sensors used in robotics to recognize people and objects from a distance.

If an obstacle is found, the LDS calculates the wavelength difference when the laser source is reflected by it. Control issues that can't be fixed by another sensor are the biggest issues that could arise with this kind of sensor because, for the low price, only one LDS is typically mounted on one device. A single laser source, a motor, and a reflective mirror make up a typical LDS. While scanning with the laser, the inner mirror is rotated by the motor. The LDS has a range of 180° to 360°.

The manner in which the laser is reflected on the environment by the tilted mirror is depicted in the first image from the left of Figure 3.2. While the motor rotates the mirror and scans the entire environment in the second and third images, the sensor records the returned laser and saves the return time (calculates the wavelength difference). The LDS sensor scans objects horizontally, so the accuracy decreases as the distance increases. However, there are some drawbacks with LDS. Closer objects are easier to identify. First, the objects must properly reflect the laser source because it measures the roundtrip wavelength difference between the two waveforms. A lot of obstacles, like transparent glass, mirrors, plastic bottles, and other objects, have a tendency to scatter or reflect the laser in a different direction, changing the origin wavelength and leading to an incorrect and inaccurate measurement.

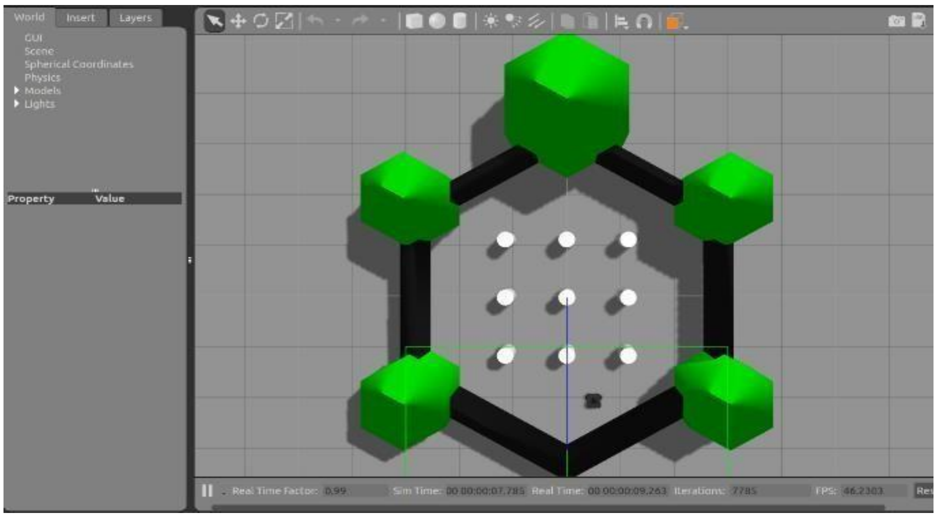
The fact that this sensor only scans horizontal objects and acquires 2D data makes the second issue appear to be a limitation. The last one is related to the possibility of eye damage because the LDS uses lasers that range in class from 1 to 4 (the higher the class, the more damage will occur).

One of LDS's best uses is simultaneous localization and mapping, or SLAM. As we will see, SLAM successfully creates a map by estimating the robot's current position and recognizing obstacles in the area.

TurtleBot More than 200 robots are built on ROS, which is one of the most popular robot operating systems. The PR2 and TurtleBot, both developed by Willow Garage in collaboration with Open Robotics, are the most well-known examples.

As the name suggests, a turtle appears in the TurtleBot logo. TurtleBot is the most widely used ROS-based robot because it is simple to learn, comprehend, and control even if you are not familiar with ROS. It is a standard platform for ROS. The final version, TurtleBot3, tries to improve some characteristics, such as the presence of Dynamixel actuators, while retaining all of the features of the earlier versions. This new version (TurtleBot3) features three distinct types of robots: Burger, waffle, and waffle pie. They are all ROS-based and intended for use in research, education, and testing. They are quite small, simple to program, and relatively inexpensive (around $600 for a burger and $1400 for a waffle).

The TurtleBot3 can be modeled and altered to create various configurations; sensors can be added or removed depending on the goal and the sensors that are available; or to reassemble the mechanical components and make use of optional components like the computer to boost the calculations



Real Time Factor: 0.99 Sim Time: 00 00:00:07.785 Real Time: 00 00:00:09.263 Iterations: 7785 FPS: 46.2303

Fig.3 Waffle load

4. Conclusion

In this thesis, we have shown a possible solution to the problem of both the navigation, applying Autonomous Voice Controlled ROBOT , which allow a robot to move and perform in an unstructured and unknown indoor environment, and the realization of the map of the scanned area using RViz (the ROS tool) or an implemented method.

The final approach called “Autonomous Navigation” is based on the idea of combining different real time Autonomous Voice Controlled ROBOT to reach a goal position (the farthest distance the robot can reach) in an unknown environment. The mobile robot (Turtlebot3: Waffle and Burger), by analyzing the information of the LiDAR, generates a free collision motion to move around the detected obstacles from its position towards the goal. No prior knowledge about the environment is assumed in this approach, which makes use only of onboard sensors to acquire information during the motion

Satisfactory results have been obtained regarding the problem of autonomous navigation of a mobile robot in unknown environments (as shown in previous chapter), but some improvements could be brought using, for example, different sensors, such as cameras, LiDAR 3D or ultrasonic sensors. Using the information from the on-board stereo camera, it would be possible to improve the navigation quality of the mobile robot, to enable 3D SLAM and navigation, or objects recognition

References

1. YoonSeok Pyo, HanCheol Cho, RyuWoon Jung, TaeHoon Lim, ROS Robot Prog. Morgan Quigley, Brian Gerkey, and William D. Smart, Programming Robots with ROS, December 2015: First Edition, Published by O‟Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.
2. Masoud Nosrati * Ronak Karimi Hojat Allah Hasanvand, Investigation of the * (Star) Search Algorithms: Characteristics, Methods and Approaches, Vol (2), April 2012.
3. Nebot E., Bailey T., Guivant J., Navigation Algorithms for Autonomous Machines in Off-
4. Road Applications, The University of Sydney, NSW 2006
5. A. Oualid Djekoune, Karim Achour and Redouane Toumi, A Sensor Based Navigation Algorithm for a Mobile Robot using the DVFF Approach, International Journal of Advanced Robotic Systems, Vol. 6, No. 2 (2009)
6. Bruno Siciliano, Oussama Khatib, Eds., Springer Handbook of robotics,
7. Springer-Verlag Berlin Heidelberg2016
8. Chuang Ruan, Jianping Luo, Yu Wu, Map navigation system based on optimal dijkstra algorithm, Proceedings ojCCIS2014
9. Javier Minguez, Associate Member, IEEE, and Luis Montano, Member, IEEE, Nearness Diagram (ND) Navigation: Collision Avoidance in Troublesome Scenarios, ieee transactions on robotics and automation, vol. 20, no. 1, february 2004
10. Takahashi O, Schilling RJ (1989), Motion Planning in a Plane Using Generalized Voronoi Diagrams. IEEE Robotics and Automation.
11. Bhattacharya P, Gavrilova ML (2008), Roadmap-Based Path Planning-Using the Voronoi Diagram for a Clearance-Based Shortest Path. IEEE Robotics and Automation.