Lawn-Mowing Robot

Group Name: *BeeRob*

Members:

Orhan KURTO – 150130904
Umut YAZGAN – 150130031
Ibrahim KARAHAN – 150160550

Date: 23.12.2018

Abstract

In this project, we have implemented a lawn-mowing-robot which is able to map the environment and mow the lawn in a garden given a path without human power and also avoid the possible obstacles. This robotic project is based on a 'Gazebo' simulation and uses the 'Turtlebot' as lawn-mowing machine. The code is written using Python language in an Intel I5 processor computer which uses Linux 16.04 as operating system.

The world we use in this simulation was created in Gazebo environment, and the code was written in Geany. For visualizing the sensor data from Turtlebot, we have used RVIZ.

Finally, we are able to follow the lawn-mowing process of the robot we have implemented in Gazebo environment as well as following the laser signals in RVIZ.

Contents

Introduction

This report has been prepared for the Fall2018 Robotics courses' semester project. The purpose of our project was to implement a lawn-mowing-robot as a Gazebo simulation using the Turtlebot robot as the lawn-mowing machine. We have implemented our idea by using several localization and mapping ROS algorithms as well as path following based on given coordinates which were obtained according to the map of the world we have created. The idea of creating a lawn-mowing-robot has been taken into consideration to ease the life of people who own houses with yards since the work of cutting grass is a continuous and long work which needs to be done about 6 months every year. By building such a project, the human power and the time this process takes can be reduced and a benefit from both financially and time wise can be gained.

This report will include the technical background, methods of our implementation as well as wide explanation of how the lawn-mowing-robot works.

Technical Background

ROS – Kinetic

ROS (Robot Operating System) is an open-source, meta-operating system for your robot. It provides services like hardware abstraction, low level device control etc. currently, ROS only runs in unix-based platforms. In our project, we have used the 10^{th} ROS distribution release called 'Kinetic' which is a release planned for Ubuntu 16.04. [1]

Gazebo

Gazebo is a free Robot-Simulation platform which makes it possible to test algorithms and robots design. It also gives the opportunity to creates new worlds (maps) using the already existing houses and tools as well as creating completely new ones. In our project, we have created a world using the tools gazebo contains. [2]

RVIZ

ROS Visualization (RVIZ) is a 3D visualizer for displaying the movements of the robot and the sensor data which comes from the sensors that are implemented in the robot. In our project, we have used the RGB camera implemented on the Turtlebot, and by using the RVIZ, we could see the generated map as well as the costmap limits. [3]

Gmap

Gmapping is a package we have used in our project which provides laser-based SLAM (Simultaneous Localization and Mapping). By the help of Gmapping, we have created a 2D map that shows the occupancy of a grid-like plane (in our case the map of the garden). [4]

AMCL

AMCL (Adaptive Monte Carlo Localization) is an algorithm which offers localization by a probabilistic approach. In our project, we have used amcl in order to keep track of the robot pose knowing the map of our world. [5]

Background Idea and Problem Review

In this project, we have aimed to build a lawn-mowing-robot which when given the map and the starting signal to it, it can cut the grass on an area following a path given by move base technology, and also avoid the possible obstacles in that area.

The idea and motivation of building such a robot was the videos we have watched of already existing lawn-mowing robots such as: Robomow RX12U Automatic Robotic Lawn Mower, John Deere Tango E5 Series II and Bosch Indego 400 Connect as well as the idea that our product can ease the human life by saving the time and power of the ones who own this product.

To solve our problem, we have used similar approach to the robots mentioned above but obviously more basic. Similarity to the other robots are that our robot also can detect obstacles and avoid them, and also has a map that shows the region it should progress. Differently from the other robots which are autonomous and make the plan of how to move in a given area, our robot follows a given path.

To solve the path following problem of our robot, we have given the move-base coordinates which were obtained according to the map our robot created using gmapping by giving the coordinates of the path we want the robot to follow in order of their appearance.

Problem Solving Design

In order to overcome the aim of this project and build a lawn-mowing robot, we tried to modify the turtlebot robot by implementing a grass cutting knife-like mechanism to the turtlebot, but unfortunately we did not succeed on that and considered to use the basic turtlebot with an rgb camera on it.

In our design, robot waits in a given base position until a command is given to it. When the command is given, it starts to mow the lawn of the mapped area according to the path we have given by avoiding obstacles in that area. When it finishes the mowing process, it returns to the base position.

Implementation

Building World in Gazebo

The world was built in Gazebo editor. In order to make it visually better and attractive, we have implemented a real like environment. The image below is the world we have built.

Figure 1: Gazebo Wold

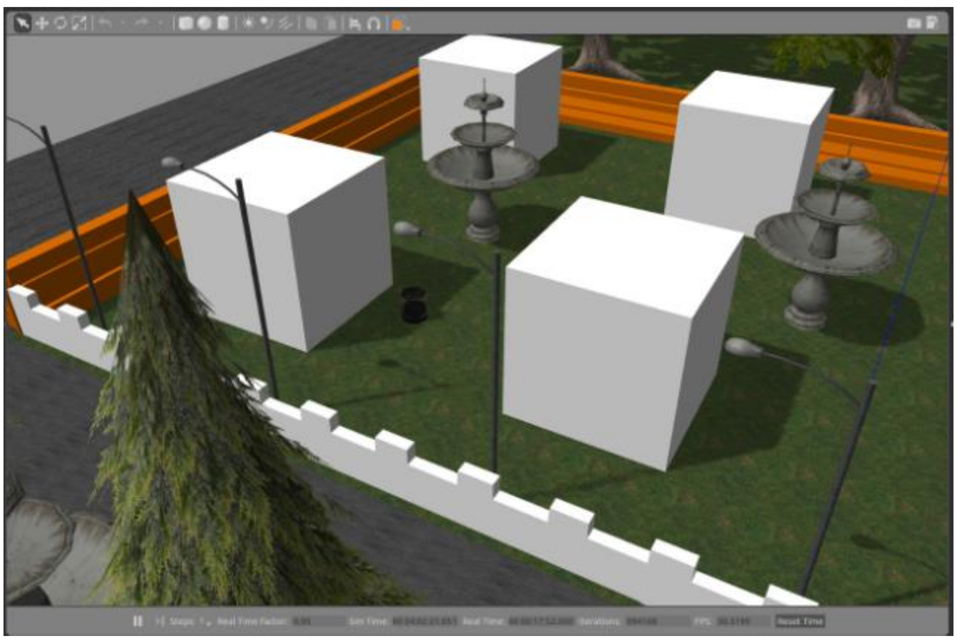

Figure 2: Gazebo World Yard and Turtlebot

Mapping

In our project, the mapping is done by using gmapping and for localization amcl. Since we have worked in a simulation, when mapping the area, we have used the laser which is already on the tutrlebot which has a 270 degrees angle of view. The created map which we created by using the teleop to move the robot in our world is as follows:

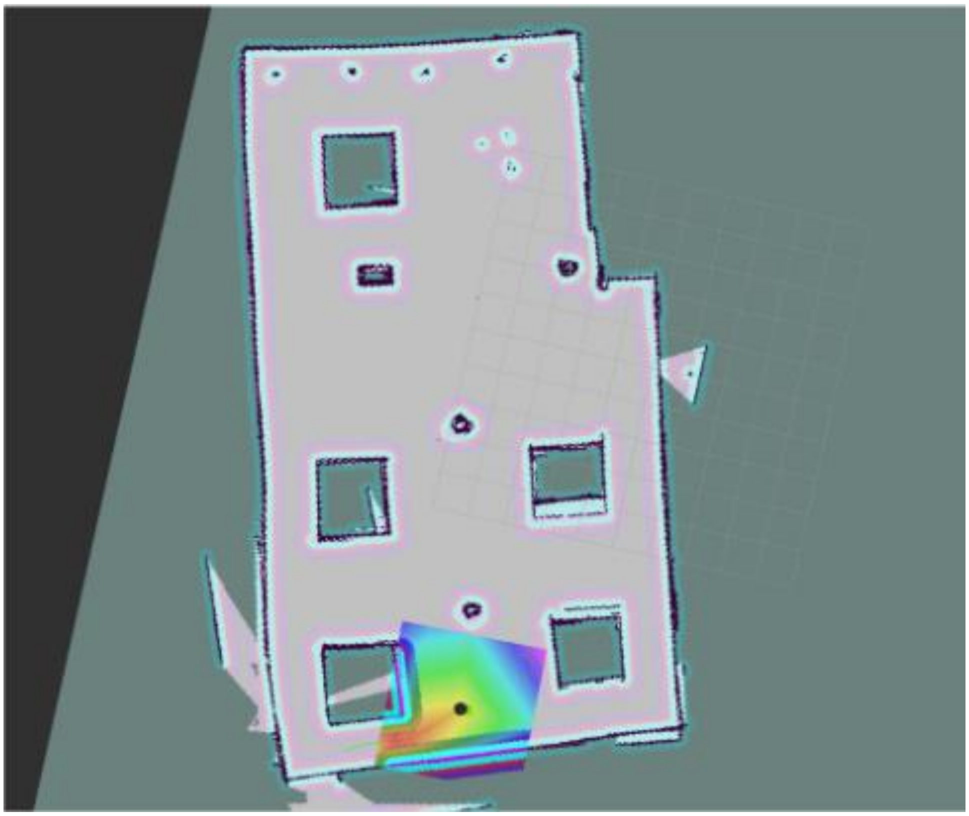

Figure 3: Map of the Gazebo World Yard

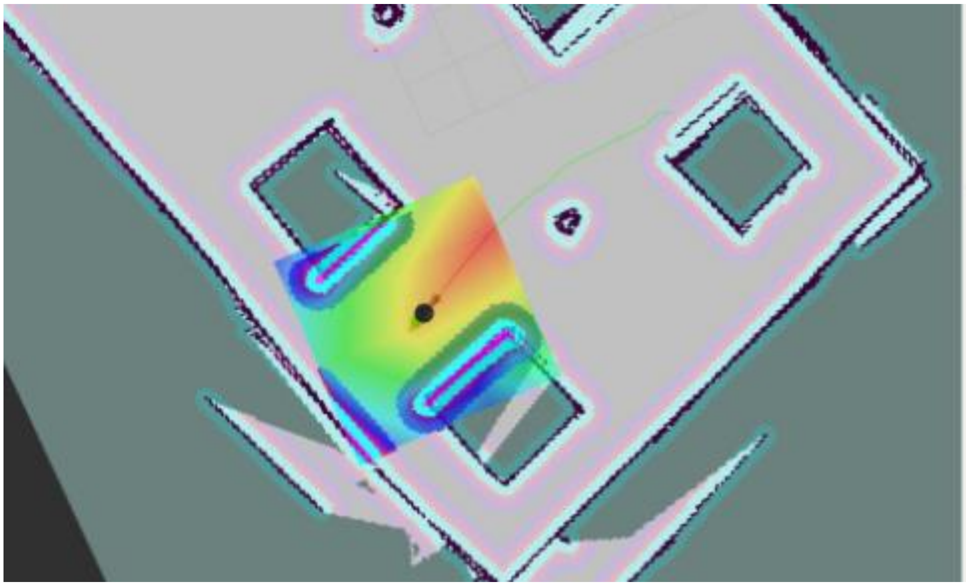

Figure 4: Costmap detail

Move-base path planning

To overcome the move-base path following problem, we have considered that robot has an initial position where it stays without moving. By using move-base, we gave the robot numerous coordinates (goals) to be reached in a planned order. We set a 20 seconds duration to robot in order to reach the given goal. In cases where the goal is not reachable, it continues with the other goals.

Results and Conclusion

To conclude, in our project, we have implemented the lawn-mowing-robot successfully. The Gazebo world we have created represents a real world instant. Mapping process worked smoothly as well as the path following part worked almost perfect with some exceptions of some given coordinates. To sum up, our model robot mapped the environment we have created, it followed the path we have given by avoiding possible obstacles in the path and returned to its base position.

After building with catkin_make, the project can be executed with the following commands:

```
$ roslaunch lawn_world lawn.launch
$ rosrun lawn_explorer go_to_goals.py
```

Video demo: https://youtu.be/5nnVwxw9Ukw

Github repository: https://github.com/umutyazgan/lawn-mowing-robot

For explanation in coding part, please check README.md and commands on the implemented code.

References

[1] Wiki. (n.d.). Retrieved from http://wiki.ros.org/kinetic

[2] Osrf. (n.d.). Why Gazebo? Retrieved from http://gazebosim.org/

[3] (n.d.). Retrieved from http://sdk.rethinkrobotics.com/wiki/Rviz

[4] Wiki. (n.d.). Retrieved from http://wiki.ros.org/gmapping

[5] Fernández, E., & Martinez, A. (n.d.). Learning ROS for Robotics Programming. Retrieved from https://www.oreilly.com/library/view/learning-ros-for/9781782161448/ch08s07.html