Final Report: Parallel Parking using TurtleBot

Dylan Tey and Tian Wu

December 18, 2018

Abstract

TurtleBot2 is being used to complete a parallel parking task. After overcoming challenges in the initial setup of getting the robot, ros and computer up and running, we are able to achieve the parallel parking tasks. The parallel parking movement is achieved by completing two arcs using the radius from TurtleBot to the wall obtained through vision.

1 Introduction

The goal of this project is to achieve the parallel parking tasks through Robot Navigation and Robot Vision by using a robot. TurtleBot2, a robot produced at Willow Garage [1], serves as the intelligent robot car in the project. This project seeks to achieve a limited functionality of Tesla's parallel parking where the car would go straight along a row of parked cars and stop when it finds a suitable parking spot. After aligning itself with the car in front of the spot, the TurtleBot would then parallel park into the open spot.

To achieve this, TurtleBot needs to be able to map its environment using its camera. TurtleBot will move along a straight path until it finds a spot on its right where it is able to parallel park.

2 The Equipment

As mentioned in the Introduction, we will using the TurtleBot2 in this project. The TurtleBot2 mainly consists of three parts - the Kobuki base, Orbbec Astra Pro camera and the computer. Kobuki is the base of the TurtleBot. It enables the TurtleBot to navigate according to the user input value. Unlike a car, the Kobuki is also able to spin in circle. The Astra Pro camera is a 3D camera that includes a microphone, IR Sensor, projector, and a RGB sensor. The camera has a range of 0.6m - 8.0m and a 73°field of view. The computer is an Acer netbook running Ubuntu 14.04 which serves as the workstation for the TurtleBot.

3 Methodology

There are certain limitation that we put on this project that will be outlined at a later section. We studied onilne the step-by-step process of the parallel parking process [2], as well as how the parallel parking algorithms are being implemented in other robots [3] The TurtleBot runs using a few operations to achieve its task.

1. Moving Forward
   We chose to run this in a controlled environment where the TurtleBot will only move in a straight line from where its starting location is. It will not stop if it sees an obstacle in front of it. This can be easily achieved by monitoring the sensor in front of it and set a certain threshold for it to stop. Our project focuses on the aspect of the parallel parking motion and thus is not in our scope. However, it would be a great addition to the project nonetheless.

2. Scanning For The First Open Space

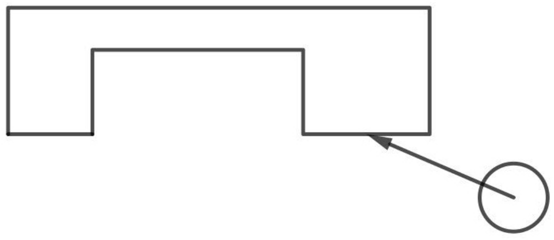

Figure 1: The Start of Searching for empty spot

As seen in Figure 1, the sensor, as part of the limitation, has to see the box. It stores this distance and uses it to compare to future distances that's measured to find the first sighting of an empty spot.

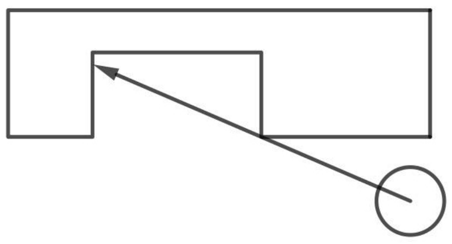
Figure 2: Detected the Start of the Parking Spot

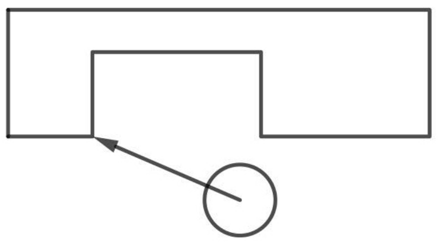
Figure 3: Detected the End of the Parking Spot

When the sensor detects a drop in distance (Figure 2), it stores this value. It then continue to take the measurement until it "sees" (decrease in distance between object and robot). This is illustrated in Figure 3 It stores this value as the end of the spot.

3. Aligning Itself To The Car/Box/Obstacle In Front of The Parking Spot

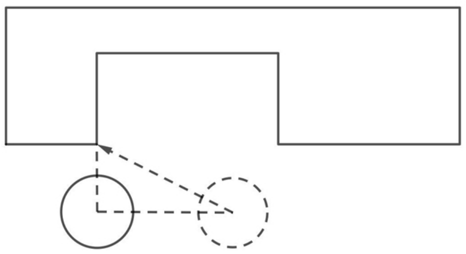

Figure 4: Aligning Itself to the Front Obstacle

After marking the end of the parking spot, the robot will determine if the space is too small. If it is, it will go back to step 2. If it isn't, it will execute the "warm up" procedure of lining up itself with the obstacle in front of the parking spot (Figure 3) for the preparation of the parallel parking process.

4. The Process of Parallel Parking

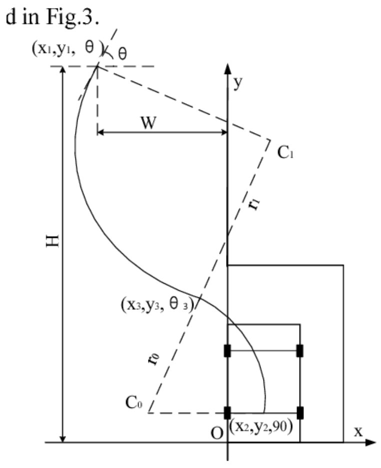

Figure 5: The Overview of Parallel Park Process

As seen in Figure 5[4], the TurtleBot first rotates around the radius between the robot and the obstacle. It would rotate until, as seen in the figure, the point where the arc and the doted line that intersects.

Then, it would rotate in the other direction of the arc to get into the parking spot. After getting into the spot, it will move forward for a predetermined time. This concludes the parallel parking process. It is important to note that this process does not use vision, but rather angles and distance traveled through mathematical equations.

4 Limitation of Our Parallel Parking Algorithm

To make our project simpler and achievable for the given time frame, we have decided to place a couple limitation to this projects. These are problems that we faced and decided not to address. In future iterations of this project, it would be reasonable to address these limitations to make the algorithm more robust.

1. Distance to Box
   The TurtleBot must be able to view the edge of the box when first ran. If it prints "nan", then the algorithm will fail since it is using the distance as a comparison.

2. TurtleBot Must Be Moving Parallel to the Parking Spot
   While moving straight, the TurtleBot must be moving parallel to the parking spot. If it veers off or close to the box or parking spot, the algorithm will no longer be accurate as it uses the radius obtained when we first engaged the robot.

5 Issues

We have issues with the Astra camera. The data we collect doesn't seem to match up with the documentation we read online. Our peers mentioned that the camera might be returning an imaginary number, so that will be something that we have to look into in the future if we were to future develop this project.

After we first programmed our first successful run of our algorithm, running it again on another TurtleBot yields a different result - minute, but results in our robot turning too much, or didn't properly aligning itself with the obstacle. Though it might be due to the TurtleBot, we think the Astra camera would be the one to blame. This resulted in our robot crashing to the box while reversing into the spot during the demonstration day.

We also faced issue with the availability of the TurtleBot in the lab. Different teams have different specification of their TurtleBot and thus we felt compelled to not "ruin" their design. Resulting in us testing our algorithm less. The scarcity and reliability of the TurtleBot laptop is also one issue we faced. One of the only three laptop isn't working, and other teams changing the configuration files. These issues though minute took up our precious time in developing our project.

6 Conclusion

In this project, we developed an algorithm for the TurtleBot to achieve the parallel parking process. Throughout this project, we faced some issues in

ROS, the Astra camera and the general lack of documentation in this field. There are many limitation and safety guards placed in this project in order to achieve the parallel parking tasks. For example, if the distance between the robot and the obstacle/parking spot is too large, it might not detect it (due to the configuration of the Astra camera we have chosen to use).

7 Future Work

In the future, we plan to develop a system that utilizes this code to complete a larger task. For example, the robot will search for parking spots along a square or rectangular pattern box on it's own, instead of the straight line limitation we currently have.

References

[1] Willow Garage. Turtlebot2: Overview.

[2] Alan Henry. The right way to parallel park, step-by-step.

[3] Jordan Van Duyne Kim Asenback, Jason Lan. Neato parking assistant.

[4] Yongyi He Lixue Wang, Liqin Guo. Path planning algorithm for automatic parallel parking from arbitrary initial angle.