Mobile Robotics - EE/CE 468

Functional Architecture

Instructor: Dr. Basit Memon

Lyeba Abid - la07309
Ali Muhammad Asad - aa07190
Sadiqah Mushtaq - sm07512
Syed Muhammad Ali Naqvi - sn07590

1 Functions

**Note: the functional architecture as a block diagram is attached at the end of the pdf*

1.1 Pose Estimation

This function takes the measurement input from the Lidar sensor. It also receives information about which objects in the field of view are landmarks and which ones are obstacles to avoid wrong estimation. It also takes the estimated pose through the wheel encoder about the pose of the robot and then uses the Kalman filter to combine this information to estimate the pose of the robot.

1.2 Odometry

The purpose of function is to calculate the next position and orientation of a robot given its current position and wheel speeds. It uses odometry calculations to update the robot's pose (position and orientation) based on the provided input parameters.

1.3 Obstacle & Landmark Recognition

This function takes input from the Camera sensor and using Image recognition decides whether the object is a landmark or an obstacle. In the simulation, we will use information from Gazebo directly to simulate the functioning of the camera and object recognition.

1.4 Set Wheel Speed

The purpose of this MATLAB function is to calculate the left and right wheel speeds for our robot based on its desired linear velocity (v) and angular velocity (omega), in order to achieve a desired path following behavior. It also considers waypoints, and the robot's current pose, and dynamically adjusts the velocity to slow down and stop the robot as it approaches the desired path endpoints.

1.5 Wheel Control

The purpose of the "wheel control" function is to determine the left and right motor torques required to achieve the desired left and right wheel speeds for our differential drive robot. It takes the target left and right wheel speeds as input, along with the current wheel speeds of the robot. The function calculates the necessary motor torques based on the difference between the target and current wheel speeds to control the robot's motion accurately. The output of this function is the left and right motor torques that need to be applied to the robot's wheels to reach the desired wheel speeds.

1.6 Distance form Person

This function receives measurements from the LiDar Sensor 2 which measures the distance of the walking cane (robot) to the person. It then calculates the distance of the person to the robot and sends this information to the control block to adjust the velocity of the cane according to the speed of the user.

1.7 PID and Obstacle Avoidance Controller

This controller uses PID to follow the path using the robot's pose and waypoints. The PID controller also takes into account the distance of the robot to the person to adjust its linear velocity. Additionally, this controller is also responsible for avoiding any obstacles that the robot might encounter in the way.

1.8 Calculate Waypoints

This function takes a map and desired destination coordinates as input. It uses this information to compute a series of intermediate waypoints that a robot should follow to navigate from its current location to the desired destination. These waypoints serve as a path or route for the robot to follow, guiding it through the map to reach the intended location.

2 Testing Plan

This environment comprises a single-story layout, completely devoid of stairs or multi-level elements. The indoor map will be pre-fed to the robot, with the landmarks having been marked. Given the final destination coordinates, the robot will calculate the waypoints it needs to follow. Thus effectively checking the navigation capabilities of the robot. The robot gets an input of the distances from the landmarks and the obstacles as odometric calculations of the pose of the robot through the wheel encodes, hence it needs to estimate its pose given this information. Hence, effectively testing the pose estimation.

In addition, the robot will also have to keep track of the objects and whether they are landmarks or obstacles (obstacles can also include people); in which case it would have to work around them. Furthermore, the robot will have to maintain its speed according to the speed of the person it is guiding, so that it doesn't lose track of the visually impaired. This will check the control module.

The above combined will ultimately be checking the PID Control as well, since that takes the robot's pose, the distance with the person, wheel speeds, and waypoints to follow the given path.

2.1 Obstacle and Landmark recognition

We will test this function by placing different objects in the field of view of the camera and checking whether the function can recognize the object and classify it as a landmark or an obstacle. We will also test the function by placing the robot in different positions and checking whether the function can recognize the object and classify it as a landmark or an obstacle. (We are assuming that the camera sensor works perfectly)

2.2 Pose Estimation

We will test this function by giving it various distances between objects and landmarks, along with different poses of the robot, and check whether it accurately estimates the correct pose of the robot in the following scenarios:
- camera sensor senses an object, and we have to get to a certain landmark where we have its distance from the lidar sensor (distances will be varied multiple times).
- camera sensor senses a landmark, and we have to get to a certain object where we have its distance from the lidar sensor (distances will be varied multiple times).

The above will be done on varying wheel speeds of the robot which would result in varying poses, therefore, we will be able to extensively test whether the robot can accurately estimate its pose or not.

2.3 Distance from Person

The second LiDaR sensor will be used to measure whether it is correctly measuring the distance between the visually impaired person and itself and whether it can correctly send this information to the PID Control block.

2.4 PID Control Block

This block will be tested on various waypoints, the pose of the robot, and the distance of the robot from the person. We will check whether the robot can follow the path correctly and whether it can adjust its velocity according to the distance of the robot from the person.

3 Revised Tentative Timeline

#	Milestone Description	Tentative Deadline
1	Define the Blocks within a Simulink Model	Week 12
2	Linear Motion	Week 13
3	Shortest Path	Week 14
4	Obstacle Avoiding	Week 14
5	Leading the Person	Week 15

Table 1: Tentative Project Timeline

Sanity?

Partially sane for now, slowly sinking into the abyss.

Functional Architecture