CSCE 274 Section 1 Fall 2016 – Project 03 – Boyd Compton, Jose Tadeo, Timothy Senn

Project Description:

The wall following program uses a program structure similar to the second project. The program divides the task into three major threads that can be characterized by their goals. These threads are the high level controller, the actuator controller, and the reading of sensor data. The high level controller is responsible for initializing shared resources, maintaining a handle on shared resources, updating the robot's state, and controlling the state of spawned threads. The actuator controller decides which actuator command to execute at a given time. For example, this thread is where the PID controller's output will be used to determine the velocity of each wheel. The thread that reads sensor data ensures synchronized sensor data for all other threads.

On top of the separation of goals in the main program, commonly used objects such as the robot or a PID controller were defined as classes in an interface. For this project, we had to modify the robot interface and sensor interface from project two to include the polling of infrared sensors on the iRobot Create 2. We also constructed a definition for a generic discrete time PID controller in the robot interface with changeable gains and a modifiable set point. The defined PID controller was then used as a uniform way to generate an output of a system no matter the type of controller. For instance, whether it be a PD or PID controller, the output of the controller can be generated from the same method.

Within the actuator controller, the obstacle avoidance was performed by checking the bump sensor values. If either bump was pressed, the robot would then rotate in place in the opposite direction. Currently, when both bumps are pressed, the robot will always rotate clockwise. The amount of rotation caused by a bump sensor is tailored to the detectable range of the infrared sensor. This choice was made to reduce the likelihood of the wall following behavior losing the wall on an accidental wall collision. For the wall following behavior, a PID controller's output is fused with the base wheel velocity for forward movement to generate the actuator command. This PID controller used the right infrared sensor as the input. For this behavior, the controller's gain was tuned for maximum stability and minimal oscillations. The right turn behavior was handled by the wall following PID controller. Although, the gain had to be adjusted to decrease the turn radius. so the robot did not lose the wall during a right turn. The left turn behavior used a separate PD controller whose input was the center right infrared sensor. When the output of the PD controller exceeded a set threshold the robot would rotate in place counter-clockwise. This threshold was chosen according to reaction time. This choice would help the robot maintain a similar set point during left turns so that the initial oscillation is reduced when the wall following behavior regains actuator control.

The set point for any PID controller was selected according to following method. First, a program that displays all the infrared sensors' value every second would be run. Next, the robot would be placed in a clear environment. We would then analyze the program's output in order to identify the base reading and noise for each sensor. After noting the observations, a wall like

object was placed around the robot. The new sensors' reading was then observed and noted. The wall like object was moved to a new position in order to see how the sensors responded to the change. The sensors' response was then noted. This was repeated until a rough positon and range for infrared sensor was found. With this information, we picked a distance where the robot had enough space to react and a change in both directions could be seen in the closest infrared sensor.

The general method used for tuning the PID controller was to start by zeroing all the gains. Next, we would steadily increase the proportional gain until a stable oscillation was achieved. Ideally, we would try to settle on a proportional gain with the largest stability range and a minimal amount of oscillation around the set point. After a suitable proportional gain was found, we would increase the derivative gain until no noticeable oscillations were present. Finally, we would adjust the integral gain to a point where the response time is acceptable with minimal oscillation around the set point. During each tuning test, the PID controller's response would be tested in three different situations which are too close to the wall, too far from the wall, and optimal distance from the wall.

Project Evaluation:

Our wall following program will follow walls containing left and right turns with minimal oscillation. A relative amount of stability can be guaranteed while the wall stays within the infrared sensors detectable range. When the wall remains outside the infrared sensors range for an extended amount of time, the robot will then continually drive forward until a new wall is located. Thus, the robot will not endlessly rotate in a circle when no wall is present. However, there is still a possibility that the robot could oscillate between left and right rotations. The method used to evaluate the program's response to various walls was to construct a course made of wall like objects. The robot was then placed at some point in the course and started. We would then monitor the robot's response to the course. After the robot followed the course reasonably well, we dynamically modified the objects in the course to test the reliability of the wall following behavior.

The only major problem found during this project was that the left infrared sensors were not positioned like the right infrared sensors. Assuming the right infrared sensors were positioned correctly, the left infrared sensor seemed to act like a front left infrared sensor. With the positional problem, it was impossible to distinguish between travelling away from the wall and being a reasonable set point. If we were to use a distinguishable set point, the robot would always be on a trajectory to collide with the wall. Thus, the left infrared sensor could not be used to accomplish the project's goals. This complication led us to only use the right infrared sensor for the wall following behavior which resulted in the program only being able to follow walls in a single direction.

Overall, the PID controller was an integral part in the success of this task, and we would recommend a PID controller to someone only if they had a systematic way of determining the goal and gains. To begin, the PID controller is an excellent interface to unify the implementation of various types of controllers such as a PD and PID controller. Additionally, various properties of a controller's response such as stability and response time can be altered by simply changing the gains for the PID controller. However, without a systematically method to experimentally tune the controller, a PID controller's tuning can take an extensive amount of time. The amount of time taken to initially tune a controller can be compounded if they do not know how each gain effects the controller's response. Another downside is the fact that a PID controller is theoretically and initially unstable until the gains have been tuned. Thus, the PID controller is a very useful control system to a knowledgeable person, so we would recommend a PID controller to someone trying to complete this task only if he was willing to research this type of controller beforehand.

Allocation of Effort:

Boyd Compton:

- Augmented the interfaces to include the infrared sensors
- Created the generic PID controller definition
- Helped tune the PID controller for the wall following behavior
- Helped plan the right and left turn behaviors
- Wrote the project write up

Jose Tadeo:

- Helped tune the PID controller for the wall following behavior
- Helped plan the wall following behavior
- Helped plan the right and left turn behaviors
- Determined the set point and gains for the PID controller
- Documented the methods used to tune and test the robot

Timothy Senn:

- Helped tune the PID controller for the wall following behavior
- Helped plan the wall following behavior
- Aided in maintaining a clear and understandable code structure
- Wrote the README.txt