|  |  |
| --- | --- |
|  | Rose-Hulman  Institute of Technology |

Memo

To: Dr. Carlotta Berry

From: Ander A Solorzano \_\_\_\_\_\_\_\_\_\_\_\_\_\_ and Ruffin White \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Class: ECE425 – Mobile Robotics

Date:

Title: Lab05 – Homing: Hybrid Control

**PURPOSE**

The purpose of this lab is to use a type of locomotion called **homing** or **docking** with hybrid control to move the CEENBoT toward a light **beacon**. With a beacon placed within our robot’s environment, our robot will search for the beacon while it is wall following. Once it finds the light beacon using the photoresistors, the robot will turn towards the light and start tracking the light using the *light lover* behavior of Braintenberg’s vehicles. Once it is in front of the light, the robot will dock then retreat to the last place it left the wall and resume the wall following behavior.

**PROCEDURES AND STRATEGY**

One of the key sub-functions we developed to keep track of our states was *moveBehavior()*. This function will first run the *moveAway()* behavior to check if there is an obstacle in the vicinity of the robot. If there is no obstacle, the robot will then proceed to detect if there is light around it by using the *moveTrackLight()*. This function will first check to see if the robot is right in front of a light by checking its light thresholds. If it is in front of a light, it will then set some global light and retreat flags that will inhibit a docking and retreat function but also suppress the light tracking behavior. Otherwise if the robot is not in front of the light, it will track the light and find the source. After the *moveTrackLight()* behavior is checked and completed, the robot will then move to wall following behavior. Finally if no wall or light is detected (i.e. the robot is in the middle of nowhere) the robot will fall back to a wander behavior.

For convenience purposes, we also set the light thresholds really high so that the robot will be able to operate under normal lighting conditions until a bright light source is detected. However, a problem we have detected with this algorithm is that the robot might get too confused if a random and powerful light source (e.g. sunlight through a window) is detected by the robot.

**QUESTIONS**

1. What does the hybrid control architecture for your design look like? What was on the planning layer? Middle layer? Reactive layer?

Our architecture comprises of a deliberative part and a reactive part. The deliberative component serves as the planning part of the robot. It is made up of a finite state machine that checks, decides what functions to run, and sets global flags to suppress behaviors. In this layer, the robot checks for each state of the robot and makes the appropiate decisions to execute the desired behavior. For our reactive layer, the robot ran the obstacle detection, wall following, and light tracking behaviors. However, only the planning layer would be able to turn them ON and/or OFF.

1. What was your general strategy for planning the path back to the wall from the beacon?

For planning our path back to the wall from the beacon, we allowed the robot to turn to the beacon directly (once a light was found) and travel towards it. Thus the path from the light to the wall is only a straight line and not a curved path. Once the robot docked, it retreated back from the light in the same straight path back to the wall. The contact sensors would serve as our proximity sensor to tell the robot that its near the wall it left.

1. How reliable was the photoresistor at detecting different objects at various shapes, sizes and distances. Compare and contrast sensor data.

The photoresistors were great for detecting light. We tuned the threshold values so that only bright light would set the tracking light behavior in motion. Compared to the temperature sensor, the photoresistors were more reliable since they were located on the extreme sides of the robot instead on the middle front of the robot. This allowed the robot to detect light a lot easier. The only drawback from using the photoresistors was that detecting a light source right in front of the robot required a bright and wide light source.

1. How significant was the difference in photoresistor voltages for the left and right sides. How did you use this difference to extract directional information to move the robot toward the beacon?

Our photoresistor values differed by 0.5 volts ± 0.2 volts. We used the Braitenberg vehicle light lover behavior described and implemented in the last lab to track the light beacons.

1. How significant was the difference in sensor data based upon distance from the source? How did you use this difference to extract distance information to move the robot toward the beacon?

When a bright light source was directly in front of the robot, the photoresistor voltages on either side read 4.7 volts ± 0.1 volt. We also made the robot have normal light thresholds of 4.19 volts. When the robot detected a light above 4.19 volts, the robot would track the light until both light sensors read 4.7 volts. This would indicate that the robot is right in front of the light source.

1. How did the architecture respond to differences in robot start position or beacon location?

The architecture behaved similarly in both cases whether the light switched positions or whether the robot track the light from a different position. The only noticable difference is the accuracy of returning back to the same spot it left the wall. This is due mostly to systematic errors of the robot coming from the differences in photoresistor readings.

1. How did the robot’s hybrid controller respond to dynamic changes in the environment (i.e. other robots and people) and compare this to purely deliberative control.

Our most primitive behavior introduced was the obstacle detection and avoidance reactive behavior. If an object or person was randomly introduced and the distance of object was within the obstacle detection threshold (not the wall following threshold) the robot would move away from the object. If the obstacle was within the wall following threshold, it would treat the random obstacle as wall and track it.

1. Were there any challenges in implementing the homing routine?

There were no significant challenges for the homing routine since we used and inhibitted the *light lover* behavior from our last lab, and retuning the PID controller for the new and different sampling time for the controller due to the added code..

1. What could you do to improve the robot homing?

Add more photoresistors to the front and sides of the robot as well as tune the light parameters so that light is detected earlier. Additional future improvements could include a sensor aggregation; have the photoresistors continue to steer the robot towards the light, and use the temperature sensor to detect the proximit. Tthe area of visible light cast is generous, but the thermal heat radiated from the light source is very intense yet small in radial proximity.

1. How did docking the robot modify the control architecture or algorithm?

We just introduced global variables that were set HIGH by our planning and middle layers to indicate that the robot was directly in front of the light and that the robot should proceed to a docking behavior. We made the robot print a status statement to the LCD screen to indicate that the robot reached this point and that it is preparing to retreat. The robot would then disable the light finding behavior in order to resume wall following and avoid tripping the docking procedure a second time.

**CONCLUSION**

. The purpose of this lab is to use a type of locomotion called **homing** or **docking** with hybrid control to move the CEENBoT toward a light **beacon**. We used a deliberative portion that would sense, think, and act accordingly, set global flags to inhibit and suppress primitive or current behaviors. For our reactive side, we constantly checked for obstacles, walls, and light sources were these primitive behaviors were either set by global thresholds, suppressed by the planning portion, or inhibited by our planning portion. To keep track of our states we used a finite state machine that would run, check, and trigger behaviors.

