Line Follower

Team 1

Lab 5

Daniel Bigelow, Ben Brown, and Brian Vees

Line Follower Robot

Problem Description:
In this lab, we had to write a program that used reflective sensors to follow a black line on the white background. During this lab, we developed and tested a program that followed the line using one sensor only, and a program that followed the line using two sensors. We then used the better of the two programs in the contest at the end of the lab.

Solutions:
Single sensor program:
We modified the simple bug by adding a single light sensor to the front of the robot, pointing at the ground.

We then designed a program that did the following:

- Once the start button was pressed, the robot would go forwards until the sensor saw black.
- Once the sensor saw black, the robot would continue to go forwards, while turning left gradually.
- Whenever the sensor saw white (i.e. the robot was leaving the line), the robot would turn to the right until the sensor found black again.
- It would continue the second and third steps until the stop button was pressed.

Two-sensor program:

Hardware:

We started with the simple bug, and added two light sensors to the front of the robot, spaced so that the gap between them was slightly wider than the black line. We also mounted the Lego RCX board onto the front of our robot, and made a face for it, but these were purely ornamental. The RCX board was used only to play music as our robot followed the line.

Software:

We designed a program that did the following:

- Once the start button was pressed, the robot would go forwards until the right-hand sensor saw black, then turn left.
- Once the line has been found (i.e. the right-hand sensor has seen black), the robot would do the following:
    - If both sensors read white, go forwards (because the robot's sensors are straddling the line.
    - Otherwise, turn towards the sensor that sees black (to get the sensors back to a position where they are straddling the line.)
- This would continue until the stop button was pressed.

We used the two-sensor program for the competition, because it worked much better than the one-sensor program. It was able to make a full circuit around the track much more quickly and smoothly. However, although the robot followed the line very closely, it was unable to go as quickly as we would have liked without losing the line. Our robot ended up taking about 30 seconds to traverse the course, which was much slower than many of the other teams'.

Problems and Solutions:

Single-Sensor:

We didn't have any problems implementing the program, but the robot couldn't move very fast at all. Because it was always turning either left or right, it was never able to go straight forwards at full speed. We were able to improve this a little by changing the sharpness of the turn, but it was never fully satisfactory.

Two-Sensor:

Originally, our control algorithm for the robot allowed it to go much faster, but would frequently lose the line and begin to spin around in circles until it found the line again, often ending up going in the opposite direction. The algorithm worked as follows:

- The robot would go forwards until it found the line.
- As long as both sensors saw the line, it would go straight.
- If one sensor saw white, the robot would turn away from that sensor to get back on the line.

This algorithm allowed the robot to go quite fast, but because the sensors were checked sequentially, the robot would sometimes check the wrong sensor first on a sharp corner, and would turn the wrong direction, thus losing the line.

We fixed this problem by changing our algorithm to the one described in the project solutions section of this report. Since the sensors were further apart, and were causing the robot to turn when one of them saw the line, the sequential checking of the sensors did not matter. However, a new problem was introduced by the change to this control algorithm. When we tried to make our robot go too fast, it would often lose the line on sharp corners, because it would not turn quickly enough, and the sensor that had seen black would cross to the other side of the line and see white again, causing the robot to go straight and lose the line. We ended up solving this problem by gearing down the robot so it would go more slowly, and have enough time to turn. However, given more time to test, we would have been able to adjust the amount of turn to a point where the robot wouldn't lose the line at higher speeds.

Conclusion:

Two-sensor line follower algorithms can work far better than one-sensor algorithms. However, sensor noise, spacing, and sampling rates can affect the results greatly. Another thing that is necessary when designing a good algorithm is a consistent power supply, so that the amounts of turn are uniform.