**COMP 4500, Mobile Robotics I**

**Spring 2018, Prof. Yanco**

**Daniel MacMillan**

**Dangnhi Ngo**

**Lab 4: Wall Following**

**Lab Report**

1. **How do the two programs compare?**

In the 2-state program, we simply make the robot turn when its side sensor goes too far or too close to the wall, and also turns left when it approaches a corner.

In the 3-state program, we wrote it in such a way that it can go straight if it is within a certain range of distance from the wall.

1. **Which one does a better job following the wall?**

The 3 state program did better going straight and rounding corners because it wasn’t constantly making turns when going straight, and it also did a better job staying a steady distance away from the wall.

1. **How far can your robot travel down the hall?**

We think it’s possible for the robot to go down the hall indefinitely, barring the occasional open door.

1. **What does it do when there are doorways with open doors?**

If the robot is heading towards the gap between the door and the wall, it will just run into the open door and get stuck. If the robot is just heading through the open doorway, it won’t have problems.

1. **What does it do with recessed doors?**

The robot, when approaching a recessed door, will just treat the door like a wall and maneuver around it.

**/\* Daniel MacMillan and Dangnhi Ngo**

**\***

**\* Lab 4**

**\* Build the robot to follow the wall using 3 states: left, straight, right**

**\*/**

#include <kipr/botball.h>

#define VEL 30

int main()

{

while(!c\_button\_clicked())

{

//printf("%d %d\n", analog(0), analog(1));

int analog0 = analog(0); // front sensor

int analog1 = analog(1); // back sensor

// The robot is at a corner or too close to the wall, turn left

if ((analog1 > 2000 && analog0 > 2200 && analog0 < 2800) || (analog1 > 2500))

{

motor(0, VEL);

motor(1, -VEL);

printf("turn left\n");

}

//following the wall, go straight

else if (analog1 > 1900 && analog1 < 2500 && analog0 < 2000)

{

motor(0, VEL);

motor(1, VEL);

printf("go straight\n");

}

//too far to the wall, turn right

else if (analog0 < 1400 && analog1 < 1800)

{

motor(0, -VEL);

motor(1, VEL\*1.5);

printf("turn right\n");

}

msleep(20);

}

ao();

return 0;

}

**/\* Daniel MacMillan and Dangnhi Ngo**

**\***

**\* Lab 4**

**\* Build the robot to follow the wall using 2 states: left and right**

**\*/**

#include <kipr/botball.h>

#define VEL 20

int main()

{

while(!c\_button\_clicked())

{

//printf("%d %d\n", analog(0), analog(1));

int analog0 = analog(0); // front sensor

int analog1 = analog(1); // side sensor

//too far the wall, turn right

if (analog1 < 1800 && !(analog0 > 2300))

{

motor(0, VEL);

motor(1, VEL\*1.7);

printf("right\n");

}

// too close to the wall or there's an obstacle

else if (analog1 > 2300 || analog0 > 2300)

{

motor(0, VEL\*2);

motor(1, VEL);

printf("left\n");

}

msleep(20);

}

ao();

return 0;

}