|  |
| --- |
| Laboratory 3:  Sensor Data Retrieval SEG4145 March 12, 2015 Martin Moncion 6146324 Jeremy Sivaneswaran 3116318 |
| Objective The objectives of this lab are to process the distance and temperature data on the robot in real-time and to also implement collision avoidance algorithms based on real-time data. Design and Algorithms The algorithm we used in run through nested loops. We use the sonar sensor to calculate how close our robot is to an object in front of it. Within the inner loop the robot will repeat the sequence of moving forward by one tile and then calculating the distance to nearby objects. If an object is < 10 cm away the robot will exit the inner loop end perform the outer loop operations which cause the robot to take the temperature, reverse by a tile, rotate clockwise 90 degrees, move forward 2 tiles, and rotate counter-clockwise 90 degrees. After the outer loop sequence the loops repeat and enter the inner loop sequence. See our flowchart section for a figure of the algorithm.  The algorithm used to ensure our robot is self-correcting itself is a bit more complex. We used simple arithmetic to determine how far a wheel must return to reach certain distances. For example when moving parallel to lab tiles the wheels need to rotate through 113 sensor changes, diagonally 160 changes and to rotate 45 degrees they need to move through 26 changes.  By constantly reading from each sensor and maintaining a synchronized count we shut down or restart a corresponding wheel motor that is out of sync. Flowchart  Discussion One problem we had with this lab is that the servo motor attached to the sonar sensor would not be straight when we start up our program. Due to this our robot would be moving straight but not looking straight to sense objects in front of it. We implemented a digital write signal to the center motor in order for the robot to be looking directly in front. Another issue is the imprecise control over the motors. We are only able to stop and start the motors there is no real way to slow them down which causes shaky movement. With we would be able to have even more precise results and much smoother motion.  Overall we were able to field test our implementation and our robot is able to execute our algorithm flawlessly. |

# Source Code

/\*

Names: Jeremy Sivaneswaran 3116318, Martin Monicon 6146324

Course Code: SEG 4145

Lab Number: 3

File name: lab\_3

Date: March 10, 2015

Description: Instructions for the robot to complete lab 3

\*/

#include <SoftwareSerial.h>

#include <Servo.h>

#include <Wire.h>

//Define constants

#define TEMPSENSOR 0x68

#define SONAR\_SENSOR 22

#define BOARD\_LED 13

#define LCD\_PIN 18

#define LEFT\_MOTOR 45

#define RIGHT\_MOTOR 8

#define RIGHT\_SENSOR 49

#define LEFT\_SENSOR 48

#define CENTER\_MOTOR 7

SoftwareSerial LCD(0, LCD\_PIN);

int reg = 0x01; // Ambient temperature = 0x01

/\*

Name: setup

Description: Initialize the robot

Parameters: none

Return: none

\*/

void setup() {

Wire.begin(); // Join I2C bus

pinMode(RIGHT\_SENSOR, INPUT);

pinMode(LEFT\_SENSOR, INPUT);

pinMode(SONAR\_SENSOR, OUTPUT);

pinMode(BOARD\_LED, OUTPUT);

pinMode(LCD\_PIN, OUTPUT);

pinMode(CENTER\_MOTOR, OUTPUT);

Serial.begin(9600);

LCD.begin(9600);

Servo Center ;

Center.attach(CENTER\_MOTOR);

Center.write(82);

}

/\*

Name: loop

Description: The main program

Parameters: none

Return: none

\*/

void loop() {

int dist;

clearScreen();

delay(1000);

displayStudentNumbers();

delay(1000);

while(1)

{

dist = getSonarDistance();

while( dist > 41){

moveForward(1);

dist = getSonarDistance();

}

displayTemperature();

moveBackward(1);

rotateRight(90);

moveForward(2);

rotateLeft(90);

}

}

/\*

Name: getSonarDistance

Description: find the distance to how close the robot is to objects in fron of it

Parameters: none

Return: the distance away from any obstruction that the sonar has found

\*/

int getSonarDistance(){

pinMode(SONAR\_SENSOR, OUTPUT);

digitalWrite(SONAR\_SENSOR, LOW);

delay(2);

// Write a HIGH value to sonar

digitalWrite(SONAR\_SENSOR, HIGH);

delay(5);

digitalWrite(SONAR\_SENSOR, LOW);

pinMode(SONAR\_SENSOR, INPUT);

unsigned long duration = pulseIn(SONAR\_SENSOR, HIGH);

return duration / (29 \* 2) ;

}

/\*

Name: rotateLeft

Description: rotate the robot by x degress

Parameters: input is the degrees of the rotation

Return: none

\*/

void rotateRight(int x){

clearScreen();

String d = "Right ";

d.concat(x);

printMessage("Rotating", 4, 0);

printMessage(d, 4, 1);

flashLED(3);

Servo Left\_Motor, Right\_Motor;

int left\_sensor, right\_sensor, old\_left\_sensor, old\_right\_sensor, left\_count, right\_count, count;

boolean right\_active, left\_active;

delay(1000);

left\_count = 0;

right\_count = 0;

left\_active = false;

right\_active = false;

old\_left\_sensor = digitalRead(LEFT\_SENSOR);

old\_right\_sensor = digitalRead(RIGHT\_SENSOR);

Right\_Motor.write(120);

Left\_Motor.write(120);

while ((left\_count <= (x/45)\*24) || (right\_count <= (x/45)\*24)){

//Serial.print(left\_count);

//Serial.print(" ");

//Serial.println(right\_count);

left\_sensor = digitalRead(LEFT\_SENSOR);

right\_sensor = digitalRead(RIGHT\_SENSOR);

if(old\_left\_sensor != left\_sensor){

left\_count++;

}

if(old\_right\_sensor != right\_sensor){

right\_count++;

}

old\_left\_sensor = left\_sensor;

old\_right\_sensor = right\_sensor;

if (left\_active){

if(left\_count > right\_count){

Left\_Motor.detach();

left\_active = false;

}

}else{

if(! (left\_count > right\_count)){

Left\_Motor.attach(LEFT\_MOTOR);

Left\_Motor.write(120);

left\_active = true;

}

}

if (right\_active){

if(right\_count>left\_count){

Right\_Motor.detach();

right\_active = false;

}

}else{

if(!(right\_count > left\_count)){

Right\_Motor.attach(RIGHT\_MOTOR);

Right\_Motor.write(120);

right\_active = true;

}

}

delay(1);

}

Right\_Motor.detach();

Left\_Motor.detach();

delay(1000);

}

/\*

Name: rotateLeft

Description: rotate the robot by x degress

Parameters: input is the degrees of the rotation

Return: none

\*/

void rotateLeft(int x){

clearScreen();

String d = "Left ";

d.concat(x);

printMessage("Rotating", 4, 0);

printMessage(d, 4, 1);

flashLED(3);

Servo Left\_Motor, Right\_Motor;

int left\_sensor, right\_sensor, old\_left\_sensor, old\_right\_sensor, left\_count, right\_count, count;

boolean right\_active, left\_active;

delay(1000);

left\_count = 0;

right\_count = 0;

left\_active = false;

right\_active = false;

old\_left\_sensor = digitalRead(LEFT\_SENSOR);

old\_right\_sensor = digitalRead(RIGHT\_SENSOR);

Right\_Motor.write(0);

Left\_Motor.write(0);

while ((left\_count <= (x/45)\*25) || (right\_count <= (x/45)\*25)){

//Serial.print(left\_count);

//Serial.print(" ");

//Serial.println(right\_count);

left\_sensor = digitalRead(LEFT\_SENSOR);

right\_sensor = digitalRead(RIGHT\_SENSOR);

if(old\_left\_sensor != left\_sensor){

left\_count++;

}

if(old\_right\_sensor != right\_sensor){

right\_count++;

}

old\_left\_sensor = left\_sensor;

old\_right\_sensor = right\_sensor;

if (left\_active){

if(left\_count > right\_count){

Left\_Motor.detach();

left\_active = false;

}

}else{

if(! (left\_count > right\_count)){

Left\_Motor.attach(LEFT\_MOTOR);

Left\_Motor.write(0);

left\_active = true;

}

}

if (right\_active){

if(right\_count>left\_count){

Right\_Motor.detach();

right\_active = false;

}

}else{

if(!(right\_count > left\_count)){

Right\_Motor.attach(RIGHT\_MOTOR);

Right\_Motor.write(0);

right\_active = true;

}

}

delay(1);

}

Right\_Motor.detach();

Left\_Motor.detach();

delay(1000);

}

/\*

Name: movediagonal

Description: move the robot Dagonal across a certain number of tiles diagonal lengths

Parameters: input is number of tiles to traverse

Return: none

\*/

void moveDiagonal(int x){

Servo Left\_Motor, Right\_Motor;

int left\_sensor, right\_sensor, old\_left\_sensor, old\_right\_sensor, left\_count, right\_count, count;

boolean right\_active, left\_active;

delay(1000);

left\_count = 0;

right\_count = 0;

left\_active = false;

right\_active = false;

old\_left\_sensor = digitalRead(LEFT\_SENSOR);

old\_right\_sensor = digitalRead(RIGHT\_SENSOR);

Right\_Motor.write(0);

Left\_Motor.write(90);

while ((left\_count <= x\*160) || (right\_count <= x\*160)){

//Serial.print(left\_count);

//Serial.print(" ");

//Serial.println(right\_count);

left\_sensor = digitalRead(LEFT\_SENSOR);

right\_sensor = digitalRead(RIGHT\_SENSOR);

if(old\_left\_sensor != left\_sensor){

left\_count++;

}

if(old\_right\_sensor != right\_sensor){

right\_count++;

}

old\_left\_sensor = left\_sensor;

old\_right\_sensor = right\_sensor;

if (left\_active){

if(left\_count > right\_count){

Left\_Motor.detach();

left\_active = false;

}

}else{

if(! (left\_count > right\_count)){

Left\_Motor.attach(LEFT\_MOTOR);

Left\_Motor.write(90);

left\_active = true;

}

}

if (right\_active){

if(right\_count>left\_count){

Right\_Motor.detach();

right\_active = false;

}

}else{

if(!(right\_count > left\_count)){

Right\_Motor.attach(RIGHT\_MOTOR);

Right\_Motor.write(0);

right\_active = true;

}

}

delay(1);

}

Right\_Motor.detach();

Left\_Motor.detach();

delay(1000);

}

/\*

Name: moveforward

Description: move the robot forward across a certain number of tiles

Parameters: input is number of tiles to traverse

Return: none

\*/

void moveForward(int x){

clearScreen();

String d="Forward ";

d.concat(x);

printMessage("Moving", 5, 0);

printMessage(d, 4, 1);

flashLED(3);

Servo Left\_Motor, Right\_Motor;

int left\_sensor, right\_sensor, old\_left\_sensor, old\_right\_sensor, left\_count, right\_count, count;

boolean right\_active, left\_active;

delay(1000);

left\_count = 0;

right\_count = 0;

left\_active = false;

right\_active = false;

old\_left\_sensor = digitalRead(LEFT\_SENSOR);

old\_right\_sensor = digitalRead(RIGHT\_SENSOR);

Right\_Motor.write(0);

Left\_Motor.write(90);

while ((left\_count <= x\*111) || (right\_count <= x\*111)){

//Serial.print(left\_count);

//Serial.print(" ");

//Serial.println(right\_count);

left\_sensor = digitalRead(LEFT\_SENSOR);

right\_sensor = digitalRead(RIGHT\_SENSOR);

if(old\_left\_sensor != left\_sensor){

left\_count++;

}

if(old\_right\_sensor != right\_sensor){

right\_count++;

}

old\_left\_sensor = left\_sensor;

old\_right\_sensor = right\_sensor;

if (left\_active){

if(left\_count > right\_count){

Left\_Motor.detach();

left\_active = false;

}

}else{

if(! (left\_count > right\_count)){

Left\_Motor.attach(LEFT\_MOTOR);

Left\_Motor.write(90);

left\_active = true;

}

}

if (right\_active){

if(right\_count>left\_count){

Right\_Motor.detach();

right\_active = false;

}

}else{

if(!(right\_count > left\_count)){

Right\_Motor.attach(RIGHT\_MOTOR);

Right\_Motor.write(0);

right\_active = true;

}

}

delay(1);

}

Right\_Motor.detach();

Left\_Motor.detach();

delay(1000);

}

/\*

Name: diagonal

Description: move the robot Dagonal across a certain number of tiles diagonal lengths

Parameters: input is number of tiles to traverse

Return: none

\*/

void diagonal(int x)

{

clearScreen();

String d= "Diagonal ";

d.concat(x);

printMessage("Moving", 5, 0);

printMessage(d, 3, 1);

flashLED(3);

Servo Right\_Motor;

Servo Left\_Motor;

Left\_Motor.attach(LEFT\_MOTOR);

Right\_Motor.attach(RIGHT\_MOTOR);

Right\_Motor.write(0);

Left\_Motor.write(90);

int count = 0;

int new\_sensor;

int old\_sensor = digitalRead(LEFT\_SENSOR);

//int old\_sensor = digitalRead(RIGHT\_SENSOR);

for(int i = 0; i<1950 \* x; i++)

{

new\_sensor = digitalRead(LEFT\_SENSOR);

//new\_sensor = digitalRead(RIGHT\_SENSOR);

//Serial.print(new\_sensor);

if (new\_sensor != old\_sensor){

count++;

}

old\_sensor = new\_sensor;

delay(1);

}

//Serial.print("\n");

Serial.println(count);

Left\_Motor.detach();

Right\_Motor.detach();

delay(2000);

}

/\*

Name: movebackward

Description: move the robot backards across a certain number of tiles

Parameters: input is number of tiles to traverse

Return: none

\*/

void moveBackward(int x){

clearScreen();

String d="Backward ";

d.concat(x);

printMessage("Moving", 5, 0);

printMessage(d, 4, 1);

flashLED(3);

Servo Left\_Motor, Right\_Motor;

int left\_sensor, right\_sensor, old\_left\_sensor, old\_right\_sensor, left\_count, right\_count, count;

boolean right\_active, left\_active;

delay(1000);

left\_count = 0;

right\_count = 0;

left\_active = false;

right\_active = false;

old\_left\_sensor = digitalRead(LEFT\_SENSOR);

old\_right\_sensor = digitalRead(RIGHT\_SENSOR);

Right\_Motor.write(90);

Left\_Motor.write(0);

while ((left\_count <= x\*111) || (right\_count <= x\*111)){

//Serial.print(left\_count);

//Serial.print(" ");

//Serial.println(right\_count);

left\_sensor = digitalRead(LEFT\_SENSOR);

right\_sensor = digitalRead(RIGHT\_SENSOR);

if(old\_left\_sensor != left\_sensor){

left\_count++;

}

if(old\_right\_sensor != right\_sensor){

right\_count++;

}

old\_left\_sensor = left\_sensor;

old\_right\_sensor = right\_sensor;

if (left\_active){

if(left\_count > right\_count){

Left\_Motor.detach();

left\_active = false;

}

}else{

if(! (left\_count > right\_count)){

Left\_Motor.attach(LEFT\_MOTOR);

Left\_Motor.write(0);

left\_active = true;

}

}

if (right\_active){

if(right\_count>left\_count){

Right\_Motor.detach();

right\_active = false;

}

}else{

if(!(right\_count > left\_count)){

Right\_Motor.attach(RIGHT\_MOTOR);

Right\_Motor.write(90);

right\_active = true;

}

}

delay(1);

}

Right\_Motor.detach();

Left\_Motor.detach();

delay(1000);

}

/\*

Name: forward

Description: move the robot forward across a certain number of tiles

Parameters: input is number of tiles to traverse

Return: none

\*/

void forward(int x)

{

clearScreen();

String d="Forward ";

d.concat(x);

printMessage("Moving", 5, 0);

printMessage(d, 4, 1);

flashLED(3);

Servo Right\_Motor;

Servo Left\_Motor;

Left\_Motor.attach(LEFT\_MOTOR);

Right\_Motor.attach(RIGHT\_MOTOR);

Right\_Motor.write(0);

Left\_Motor.write(90);

int count = 0;

int new\_sensor;

int old\_sensor = digitalRead(LEFT\_SENSOR);

//int old\_sensor = digitalRead(RIGHT\_SENSOR);

for(int i = 0; i<1375 \*x; i++)

{

new\_sensor = digitalRead(LEFT\_SENSOR);

//new\_sensor = digitalRead(RIGHT\_SENSOR);

//Serial.print(new\_sensor);

if (new\_sensor != old\_sensor){

count++;

}

old\_sensor = new\_sensor;

delay(1);

}

//Serial.print("\n");

Serial.println(count);

Left\_Motor.detach();

Right\_Motor.detach();

delay(2000);

}

void reversediagonal()

{

Servo Right\_Motor;

Servo Left\_Motor;

Left\_Motor.attach(LEFT\_MOTOR);

Right\_Motor.attach(RIGHT\_MOTOR);

Right\_Motor.write(120);

Left\_Motor.write(0);

int count = 0;

int new\_sensor;

int old\_sensor = digitalRead(LEFT\_SENSOR);

//int old\_sensor = digitalRead(RIGHT\_SENSOR);

for(int i = 0; i<1925; i++)

{

new\_sensor = digitalRead(LEFT\_SENSOR);

//new\_sensor = digitalRead(RIGHT\_SENSOR);

//Serial.print(new\_sensor);

if (new\_sensor != old\_sensor){

count++;

}

old\_sensor = new\_sensor;

delay(1);

}

//Serial.print("\n");

Serial.println(count);

Left\_Motor.detach();

Right\_Motor.detach();

delay(2000);

}

void reverseforward()

{

Servo Right\_Motor;

Servo Left\_Motor;

Left\_Motor.attach(LEFT\_MOTOR);

Right\_Motor.attach(RIGHT\_MOTOR);

Right\_Motor.write(120);

Left\_Motor.write(0);

int count = 0;

int new\_sensor;

int old\_sensor = digitalRead(LEFT\_SENSOR);

//int old\_sensor = digitalRead(RIGHT\_SENSOR);

for(int i = 0; i<1325; i++)

{

new\_sensor = digitalRead(LEFT\_SENSOR);

//new\_sensor = digitalRead(RIGHT\_SENSOR);

//Serial.print(new\_sensor);

if (new\_sensor != old\_sensor){

count++;

}

old\_sensor = new\_sensor;

delay(1);

}

//Serial.print("\n");

Serial.println(count);

Left\_Motor.detach();

Right\_Motor.detach();

delay(2000);

}

/\*

Name: printStopMessage

Description: Prints a stop message on the LCD display

Parameters: none

Return: none

\*/

void printStopMessage(){

clearScreen();

printMessage("Stopped", 5, 0);

flashLED(3);

delay(3000);

}

/\*

Name: printMessage

Description: Prints a given message to the LCD display

Parameters:

text - String - In - the message to print

col - int - In - the column of the display to start the message at

row - int - In - the row of the display used to display the message

Return: none

\*/

void printMessage(String text, int col, int row){

LCD.write(0xFE);

LCD.write(col + row \* 64 + 128);

LCD.print(text);

}

/\*

Name: flashLED

Description: Instructions to blink the LED light for a certain period of time

Parameters: x - int - In - the number of seconds to blink the LED

Return: none

\*/

void flashLED(int x) {

for (int a = x; a > 0; a--){

digitalWrite(BOARD\_LED, HIGH);

delay(500);

digitalWrite(BOARD\_LED, LOW);

delay(500);

}

}

/\*

Name: displayStudentNumbers

Description: Instructions to display the student numbers on the LCD screen

Parameters: none

Return: none

\*/

void displayStudentNumbers(){

clearScreen();

printMessage("3116318", 5, 0);

printMessage("6146324", 5, 1);

flashLED(5);

}

/\*

Name: displayTemperature

Description: Instructions to display the scurrent room temperature on the LCD screen

Parameters: none

Return: none

\*/

void displayTemperature(){

Wire.beginTransmission(TEMPSENSOR);

Wire.write(reg); // Indicate temperature value to read

Wire.endTransmission();

Wire.requestFrom(TEMPSENSOR, 1); // Request data

while(Wire.available() < 1); // Wait for data

byte temperatureData = Wire.read(); // Temp. value

clearScreen();

printMessage("Temperature", 2, 0);

printMessage(String(temperatureData) + " Degrees", 2, 1);

flashLED(5);

}

/\*

Name: clearScreen

Description: Instructions to clear the LCD display screen

Parameters: none

Return: none

\*/

void clearScreen (){

LCD.write(0xFE);

LCD.write(0x01);

delay(10);

}