ECE452C_Fall-2017_Proj-01_Team-13

Video Link: https://youtu.be/OCblS2xzvso

Group Member	Responsibility
Dong, Zhenglin	1. Assembled the robot; 2. Made the environment for the robot; 3. Finished the codes for moving along the line; 4. Finished the codes for measure distance; 5. Finished codes for starting the robot with tapping on accelerometer.
Pan, Hongyi	1. Assemble the robot; 2. Adjusted line threshold value; 3. Finished the code for measure distance; 4. Combined the code of “line-following” and “drive in 50 inches”.
Zhang, Haopeng	1. Assembled the robot; 2. Made the environment for the robot; 3. Tested sensors and motors; 4. Made the video; 5. Finished code for starting the robot with tapping on accelerometer.

Difficulties

1. Wreck the wire of sensor for detecting distance by accident.
2. Robot rushes out the trace in sharp curve.
3. It is inconvenient to test the code with motor on.
4. Accelerometer is too sensitive. The car might start by mistake.

Solutions:

1. Buy a new sensor online for measuring distance.
2. Slow down the speed to move, and adjust the line threshold value. If the robot can keep going along the line, accelerate the robot.
3. Make use of two switches on the board: one is for motors, the other is for power.
4. Set a proper trigger value and use an average filter.

Steps to tune the sensors

1. Test the Line Follower Board
2. Test the Encoder Hall Effect Sensor (SIK)
3. Test the left, middle, right value of Line Follower Board when the board is above line and not above line. Get the line threshold value.
4. We use sample program to observe the reference value of the accelerometer. Then we get the trigger value which can not only guarantee that the car can start by tapping, but also prevent a wrong trigger. In consideration of the high sensibility of the accelerometer, we then use an average filter to ensure the stability.

Code list:

```
#include <RedBot.h>
RedBotMotors motors; // define motors
RedBotSensor left = RedBotSensor(A3);     // initialize a left sensor object on A3
RedBotSensor center = RedBotSensor(A6); // initialize a center sensor object on A6
RedBotSensor right = RedBotSensor(A7);    // initialize a right sensor object on A7
RedBotEncoder encoder = RedBotEncoder(A2, 10); // initialize left encouder object on A2, right encounder on 10
RedBotAccel accelerometer;
#define LINETHRESHOLD 800 //Line value
#define SPEED 60    // set the nominal speed. Set to any number from 0 - 255.
int leftSpeed;     // variable used to store the leftMotor speed
int rightSpeed;    // variable used to store the rightMotor speed

int buttonPin = 12; // set buttton as 12
int countsPerRev = 192;     // 4 pairs of N-S x 48:1 gearbox = 192 ticks per wheel rev
```

```
float wheelDiam = 2.56;    // diam = 65mm / 25.4 mm/in
float wheelCirc = PI*wheelDiam;    // Redbot wheel circumference = pi*D

void setup()
{
   pinMode(buttonPin, INPUT_PULLUP); // initialize button as input_pullup
   Serial.begin(9600); // initialize serial
//    while (digitalRead(buttonPin) == HIGH);
//    driveDistance(50);    //drive 50 inches
}

void loop(void)
{
   // drive on button press.
   accelerometer.read(); // updates the x, y, and z axis readings on the acceleromter

   int n;
   float X=accelerometer.x; //get the value of the accelerometer on x axis
   Serial.print( X,2 );
   Serial.print("\n");

   for( n=0;n<100;n++ )
   {
      X = X+accelerometer.x;
      }
     X=X/100;                    //average filtering

   if (    X < -3000 )      //      digitalRead(buttonPin) == LOW &&
   {
      driveDistance(50);    //drive 50 inches
   }
}

void driveDistance(float distance)
{
   long stopCount = 0;
   long lCount = 0; // left count value
   long rCount = 0; // right eount value
   float numRev; //goal encoder value
   numRev = distance / wheelCirc; //goal encoude value = goal distance / wheel circumference
   encoder.clearEnc(BOTH);    // clear the encoder count
   while (lCount+rCount < 2*numRev*countsPerRev) // while average count value < goal
   {
```

```
    if(center.read() > LINETHRESHOLD) // if center sensor is above the line, go straight
    {
      leftSpeed = -SPEED;
      rightSpeed = SPEED;
    }
    else if(right.read() > LINETHRESHOLD) // if right sensor is above the line, turn left
    {
      leftSpeed = -(SPEED + 50);
      rightSpeed = SPEED - 50;
    }
    else if(left.read() > LINETHRESHOLD) // if left sensor is above the line, urn right
    {
      leftSpeed = -(SPEED - 50);
      rightSpeed = SPEED + 50;
    }
    if((left.read() > LINETHRESHOLD) && (center.read() > LINETHRESHOLD) && (right.read() > LINETHRESHOLD)) // if all sensor is above the line, stop
    {
      if(stopCount > 500) motors.stop();
      else stopCount ++;
    }
    else //else, adjust speed
    {
      motors.leftMotor(-leftSpeed);
      motors.rightMotor(-rightSpeed);
      stopCount = 0;
    }
    lCount = abs(encoder.getTicks(LEFT)); // get left encoder value
    rCount = abs(encoder.getTicks(RIGHT)); //get right encoder value
    Serial.print("driveDistance "); // print drived distance
    Serial.print((lCount + rCount)*wheelCirc/2/countsPerRev);
    Serial.print("\"\n"); // print drived distance
  }
  motors.brake();//stop
  Serial.print("=====================================\n");
  Serial.print("driveDistance "); // print drived distance
  Serial.print((lCount + rCount)*wheelCirc/2/countsPerRev);
  Serial.print("\"\n"); // print drived distance
  delay(1000);
}
```