**Lab 8: Arduino Smart Car: the BOE-BOT**

Andrew Jiang, Sharanya Pulapura, Nabib Ahmed

1. a) Pins 5, 6, and 7 are connected to the infrared sensors.

2. a) It moves forward at a constant rate.

3. a) Change the method to maneuver(0, 200, 60), or just change speedLeft to 0.

b) Change speedRight to 80 so the method is maneuver(200, 80, 60).

4. a) This function sets the angle of the shaft.

b) One adds while the other subtracts because the angle set in the left wheel is symmetric and thus opposite to the angle set in the right wheel. Any parameter that we add to servoLeft thus must be negative to produce the same effect in servoRight.

c) x is the speed of the left wheel, or how much angle you’re adding to the left wheel. y is the same case, except for the right wheel. z acts as a variable of time that works as a delay for the entire maneuver function.

5. a) The total size of the BOE-BOT program is 3,358 bytes.

b) The total size of Arduino memory is 32,256 bytes.

c) 10% of the total memory is taken up by our program.

6. a) It moves straight on the black line and stops when it reaches a corner.

7. a)

/\*  
 \*Line Follower Robot  
 \*Adaptation of FastIrRoaming from Parallax Database    
 \*ES50 Lab 4  
 \*Jason Smith and Demetrio Anaya   
 \*/  
   
#include <Servo.h>                           // Include servo library  
   
Servo servoLeft;                             // Declare left and right servos  
Servo servoRight;  
   
void setup()                                 // Built-in initialization block  
  
{                               
  servoLeft.attach(11);                      // Attach left signal to pin 11  
  servoRight.attach(12);                     // Attach right signal to pin 12  
   Serial.begin(9600);  
}    
   
void loop()                                  // Main loop auto-repeats  
{  
  
  int irLeft   =  digitalRead(5);       // Set value of LEFT sensor  
  int irCenter =  digitalRead(6);       // Set value of CENTER sensor  
  int irRight  =  digitalRead(7);       // Set value of RIGHT sensor  
    
  //Serial.println(digitalRead(i));        // If enabled will display feedback of pin i in serial monitor  
  //delay(250);  
     
  if(irLeft == 0 && irCenter == 1 && irRight == 0) {    // Straddling Line (Ideal Case)  
    maneuver(200, 220, 0);  
  } else if(irLeft == 1 && irCenter == 0 && irRight == 0) {   // turn left  
    maneuver(250, 0, 0);  
  } else if(irLeft == 0 && irCenter == 0 && irRight == 1) {   // turn right  
    maneuver(0, 250, 0);  
  } else if(irLeft == 1 && irCenter == 1 && irRight == 0) {   // turn right  
    maneuver(250, 150, 0);  
  } else if(irLeft == 0 && irCenter == 1 && irRight == 1) {   // turn right  
    maneuver(150, 250, 0);                
  } else if(irLeft == 1 && irCenter == 1 && irRight == 1) {   // turn right  
    maneuver(200, 220, 0);   
  } else if(irLeft == 1 && irCenter == 0 && irRight == 1) {   // turn right  
    maneuver(20, 20, 0);                    
  } else {  
    maneuver(0, 0, 0);      // Stops  
  }   
  
}  
  
  
void maneuver(int speedLeft, int speedRight, int msTime) // Defines maneuver function  
{  
  // speedLeft, speedRight ranges: Backward  Linear  Stop  Linear   Forward  
  //                               -200      -100......0......100       200  
  servoLeft.writeMicroseconds(1435 + speedLeft);   // Set Left servo speed  
  servoRight.writeMicroseconds(1475 - speedRight); // Set right servo speed  
  if(msTime==-1)                                   // if msTime = -1  
  {                                    
    servoLeft.detach();                            // Stop servo signals  
    servoRight.detach();     
  }  
  delay(msTime);                                   // Delay for msTime  
}

b) Worked, verified by Joy.

8. a) The BOEBOT had lots of trouble turning left, despite the code being identical to the code for turning right. In addition, sometimes when going straight, the BOEBOT would stop for a few seconds before adjusting itself back into the straight direction.

b) The BOT needs to be fixed by changing hardware: the sensors simply aren’t precise enough to completely follow the black line path. The code makes sense, but the practical application of where the sensors detect is too big to be precise enough. Making the sensors smaller or spacing them out more, or simply increase the number of them will allow for a more precise following fo the line as our theoretical code should have followed.