CSCI 458: Autonomous Mobile Robotics

Assignment Name: Boundary Bot
Assignment Number: 4
Group Members: Nisha Patel & Ricky Hempel

In this assignment, we used light/color sensors for a robot that respects a boundary. The robot will be set on a practice board on an unmarked region, and it should explore again until it reaches a boundary by a black marker. The region will be entirely enclosed by the black marker and can be in any shape. The basic functionality of a robot is that it can detect a boundary and do some sort of movement to stay in the region.

This is what worked. With the hardware, we mounted the light sensor close to the ground where it points directly downwards. We also chose to go with a simple design this time without making it too complicated. This being that the sensor would be as close to the robot as possible and still be ahead of the wheels so that the robot would then detect the line before the wheels got to the line.

With the software, we used a threshold for the black marker that can be found by taking a reading on both a dark surface (the marker) and a light surface (the board), adding them together, then dividing by 2. After we calculated the threshold, we then used getColorReflected() on it. Our robot is capable of detecting a marked region by exploring its way around. It won't move further when it detects a black marked boundary. Then, it will stop and reverses before choosing a random direction to continue the exploration this done by using the random() function. It chooses a random number between zero and ten, and if the number is less than 5 turn right, if the number is 5 or greater turn left.

This is what did not work. First, we tried the color/light sensor further away from the robot and that did not work because we felt that the sensor was too far from it. After the first time, we moved it closer to the robot the sensor was at an angle that would not read the colors right so we moved it to the position that ended up being our final position.

The software that did not work was first choosing which function to use with the sensor based on outside reading we first thought getColorAmbient() would work but we soon realized that this was not the way to go. We then tried getColorRGB() to get the RGB values, but we read about thresholds, and we were not sure how to get thresholds from RGB values. Lastly, we tried getColorReflected() and realized that was the way to go. The last major aspect to figure was the threshold we tried testing different numbers to see if we could guess a bound but that did not work.

We learned a lot about how the light/color sensor worked. We light the difference between getColorAmbient(), getColorReflected(),getColorRGB(). We also how positioning works with respect to the sensor and the ground. Lastly, we learn how to find the light and dark values and how to calculate them to find the threshold.

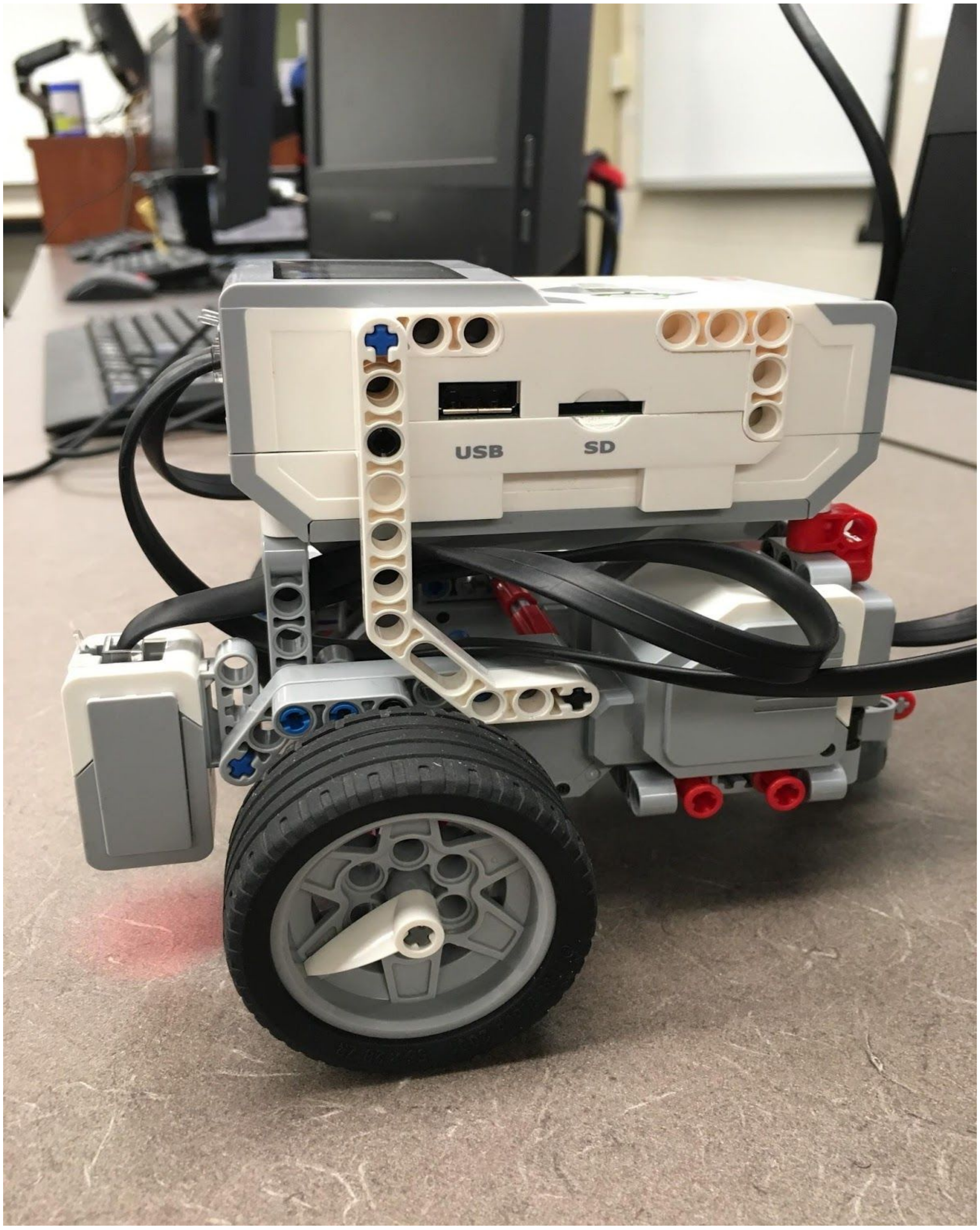

Figure 1: Side-view of a robot with the light sensor

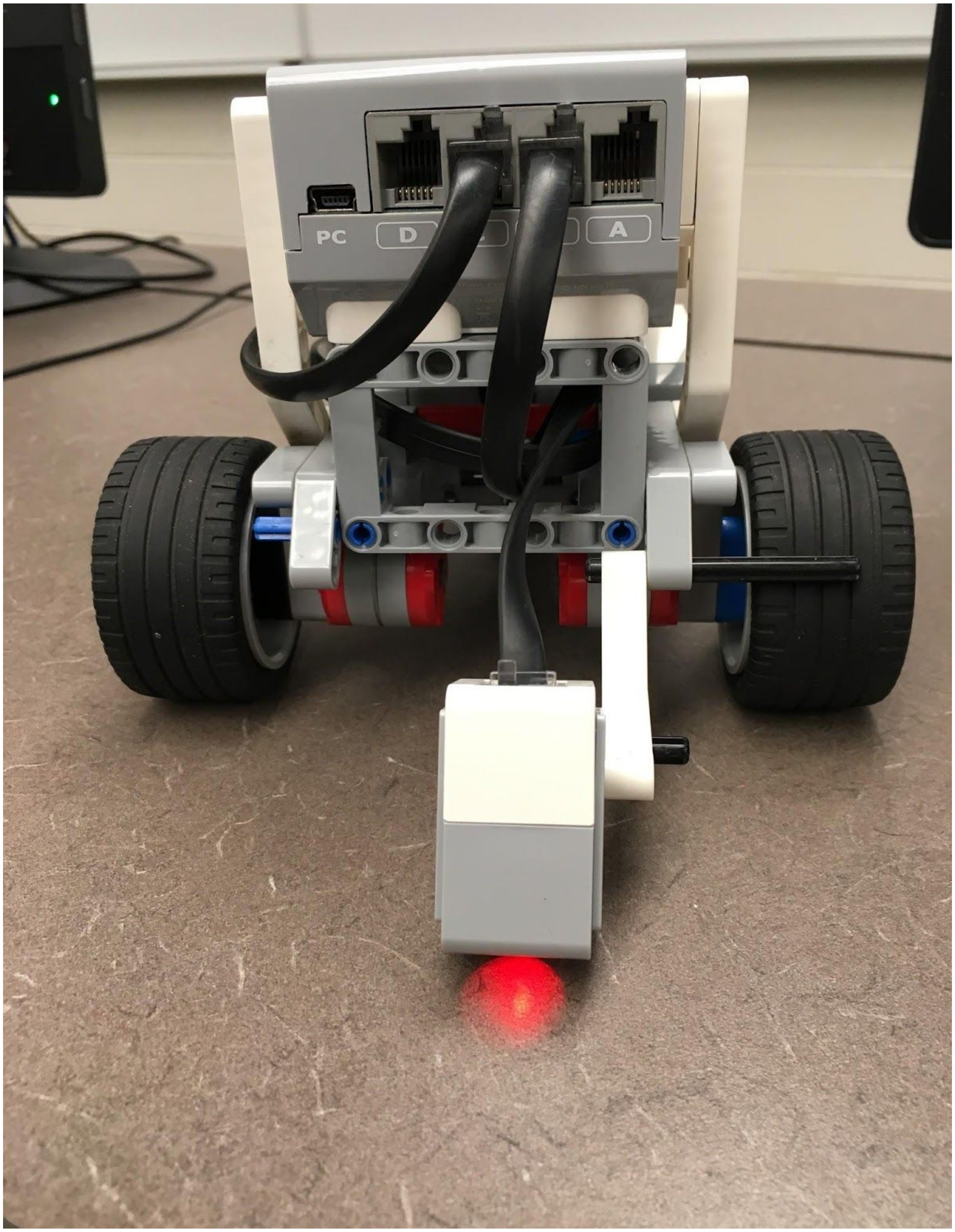

Figure 2: Old build of a robot with the light sensor.

```
/************************************
*Name: Nisha Patel and Ricky Hempel          *
*Description: purpose code for color sensor    *
*************************************/
//tells which port the sensor is in
#pragma config(StandardModel, "EV3_REMBOT")
#pragma config(Sensor, S3,sonar3, sensorEV3_Color)
//for motors
tMotor Left_Motor=motorB;
tMotor Right_Motor=motorC;
task main(){
//calculating threshold by taking a reading on both DARK and LIGHT surfaces,adding them together, then dividing by 2.
  int threshold = 11;
  while(true){
      //if threshold is greater than sensor value.
      if(getColorReflected(S3)<threshold){
            //stops when the line is detected

             setMotorSpeed(Left_Motor, 0);

             setMotorSpeed(Right_Motor, 0);
             sleep(500);
             //reverse
             setMotorSpeed(Left_Motor, -20);
             setMotorSpeed(Right_Motor, -20);
             sleep(2000);
           //randomly chooses a new direction
           //turn right
           if(random(10)<5){
               setMotorSpeed(Left_Motor, 10);
               setMotorSpeed(Right_Motor, -10);
               sleep(1875);
            }
             //turn left
           else {
              setMotorSpeed(Left_Motor, -10);
              setMotorSpeed(Right_Motor, 10);
             sleep(1875);
             }
      }
     //go forward
    setMotorSpeed(Left_Motor,50);
    setMotorSpeed(Right_Motor,50);
  }
}
```