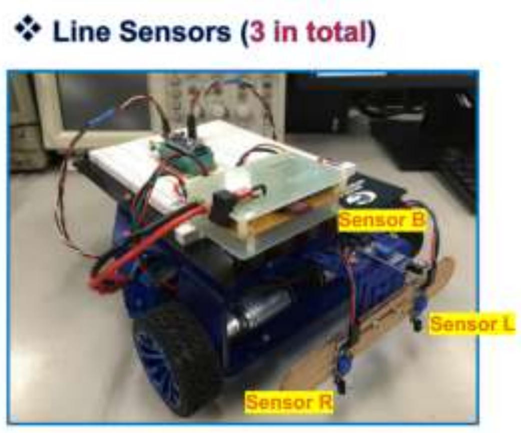
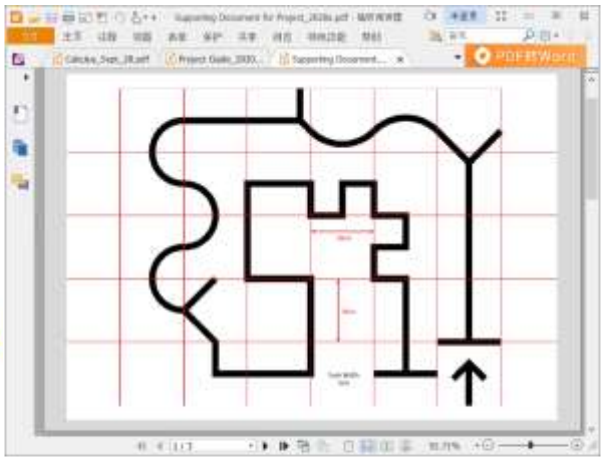
Project report

Introduction:

This project focuses on designing a robot car which should be able to run following a track (as shown below) and stop at the end. It will complete the tasks including going along the track, choosing the correct split to go, turning in the 90 degrees angles and stop at the end. Due to the special situation of this semester, I was only required to do the logic and program designing part.





The car is equipped with three sensors, the left one and the right one are for detecting the track, and the bumper sensor in the middle is for detecting whether the car should start up or stop. On the black mat the line will be white and the sensors will give out low voltage only when they detect the line

Code design (All the details can be checked from the code in appendix):

1. Start up and stop (function "StartUp()" and "Stop()")

When the bumper sensor is triggered for the first time, the car will start up, and it will stop for the second trigger. Unfortunately, it is impossible to draw a truth table for this part since what matters more is how many times the bumper sensor is triggered rather than whether it is triggered or not. As a result, I use a global variable "start", which is initialized to 0, and it will be set from 0 to 1 or from 1 to 0 for every time the bumper sensor is triggered. All the code in the loop function will be executed only when the start is 1. For security purpose, the "start" will only be set to 1 when both left and right sensor are triggered in case it may wrongly start up when it is not in at the start line.

2. Following the track(function "Adjust()"):

When the car goes along the track, the direction it goes is impossible to be perfectly parallel with the track, especially when the track is a curve or having a series of sharp turns, so the car must be able to fix its direction back. The sensors will help the car to know whether it should adjust a little bit to right or left. Here is the truth table. (L.S. represent left sensor, R.S. represent right sensor, "+" and "-" represent the direction of motor)

Left motor:

	1	0	R.S.
1	+	+	
0	-	x	
L.S.			

Right motor:

	1	0	R.S.
1	+	-	
0	+	x	
L.S.			

When both of the sensor are triggered, that indicates the car has run into a split, we will talk about it later. When only the left sensor is triggered, that indicates the car is going a bit right, so we will fixed it back by turning a bit to left. The situation that right sensor is triggered is symmetrical. To turning car around, we can control the two motors to run in different direction.

3. Split (function "turn_in_branch ()"):

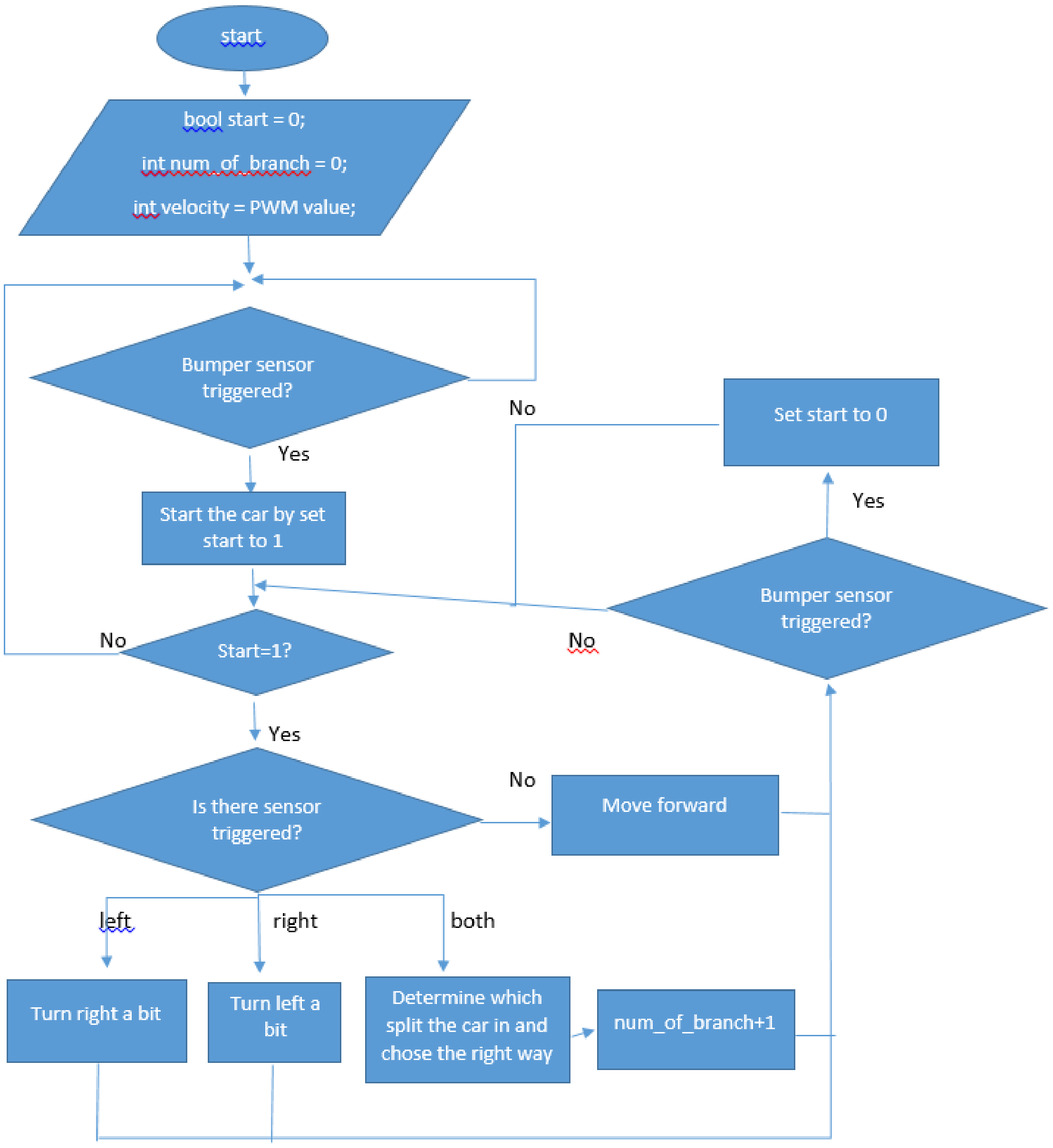
When both of the sensor are triggered, the function "branch()" will be executed to make the car pass the split. Since there is no advanced equipment to detect where each split led to, the car will not able to know which way it should go automatically, the only solution is hard code. The global variable "num_of_branch" is to record how many split the car has passed. It will turn left at the first two splits and right in the third splits, with "num_of_branch" increment after each split. Since there is no more splits after the third one, the function will be disabled (just do nothing) when num_of_branch>=3, in case there is any accident trigger both sensors unexpectedly.

4. PWM

My PWM is a global constant since there is no need to change it. It should not be too high or too low, or the car will either run out of the track due to the high speed or can`t move due to the poor power. Since I don`t have an Arduino car and the actual speed is not linear to the PWM, it was hard to predict the suitable PWM and I could only choose one mainly by guess for the first time. In the first code test I tried 50, but it turned out that the power was too low to move the car. Afterward I tried to discuss it with other classmates, taking their PWM values as reference to help me set my own value. With those additional information, I estimated that 80 might be a suitable value, which is actually the PWM in my final version of code.

5. The logic flow

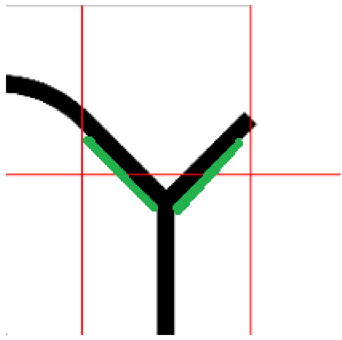
This is the logic for the robot car and the steps just show what the car should do. You can see how I realize those functions in the code pasted in appendix (there is no end of the program since the loop function in Arduino will never end)



Evaluation:

Since my code has performed a perfect run, I think it is good in general. The part I am proud of most is the coding logic. I divided the code in to 4 functions and each of them matched with a one required task. That makes the code easy to read and arrange, which saves me a lot of time on debugging and finding out the logic error. Apart from that, by simulating on Tinkercad, I found the priority of passing the split should be higher than following the track, so I carefully ordered those functions to make them perform properly. As a result, my car can perform more stable and able to overcome more unexpected situation.

Although the coding is my strength, that doesn`t means there is no error at all. I would like the share the hardest problem I met during the project period and how I fixed it. In the first version of my code, I made the car turn by spin the two motor in different direction in both the part of following track and split part. However, during the first code test lab, I found my car got stuck when it came to the split. With thoroughly thinking and analysis, I found the reason was the both sensors will be triggered at the edge of the split (as the green mark). When the car turned a small angle, for example, turned to the left, the sensors would spin with the car, and the left sensor would go out of the track while the right sensor would go into the middle of the white line. That means the left sensor couldn`t sense the white line anymore but the right sensor still could, so the car would consider it needed to adjust to right side. But when it spin back, the both sensor were triggered again. It would run into an infinite loop and got stuck. The solution I came up with is turning while the car moves forward. To do so, I just set one of the motor`s PWM lower than another and the difference of speed will make the car turn. That`s what I did with my second version code, which finally performed a perfect run.



To be honest, the car is not perfect. The speed is slow and still need to be improved. If we were going to have a race like what we usually do in the other semesters, it would not able to make it. If I could access to the physical lab, I would have more chance to test different PWM value and chose the most ideal one. Besides, I observed some strange behave of the car in the vertical turns. It always stopped a while at the turns to make adjustment, and sometimes it would even go backward a bit, although I never made any attempt to make it do so. Firstly, there is a risk that the car might got stuck at the turns, although it passed all the turns in the rehearsal demo. Secondly, that going backward unexpectedly may indicates some hidden error which I haven`t noticed yet. For safty there should be some improvement to make it pass the turns more smoothly, and find out why the car will go backward.

Conclusion:

This is the first time for me the actually build a robot and run it. Building something is a complex thing and very different from doing an exam or quiz, there are always some problems that can never be predicted in advance. You have to try, find out what is wrong, and try again to fix it. The knowledge we have learnt in lecture is not everything here, experience also plays an important role. Sometimes when the knowledge can`t help, we have to figure out the problem by experience, and even intuition. From this project I got to know the outline of how to design a robot. It is not an easy experience, but really worthy.

Appendix:

1. First version code:

```
bool start = 0;
int num_of_branch = 0;
int velocity = 50;// may be changed
void setup() {
  // put your setup code here, to run once:
  pinMode(12,OUTPUT);//R_dir
  pinMode(11,OUTPUT);//L_dir
  pinMode(10,OUTPUT);//L_PWM
  pinMode(9,OUTPUT);//R_PWM
  pinMode(A3,INPUT);//R_sensor
  pinMode(A4,INPUT);//B_sensor
  pinMode(A5,INPUT);//L_sensor
}

void turn_in_branch(){
  int R_sensor_value = digitalRead(A3);
  int L_sensor_value = digitalRead(A5);
  if(R_sensor_value==0&&L_sensor_value==0){
  while(R_sensor_value==0&&L_sensor_value==0){
    if(num_of_branch == 0){
      digitalWrite(11,LOW);
      digitalWrite(12,HIGH);
      //analogWrite(9,velocity);
      }
     else if(num_of_branch == 1){
      digitalWrite(11,LOW);
      digitalWrite(12,HIGH);
      //analogWrite(9,velocity);
      }
     else if(num_of_branch == 2){
      digitalWrite(11,HIGH);
      digitalWrite(12,LOW);
      //analogWrite(10,velocity);
      }
     else{
      //analogWrite(9,velocity);
      //analogWrite(10,velocity);
      }
    R_sensor_value = digitalRead(A3);
    L_sensor_value = digitalRead(A5);
    }
  num_of_branch++;
}
  }

void Adjust(){
  int R_sensor_value = digitalRead(A3);
  int L_sensor_value = digitalRead(A5);
  while(R_sensor_value==0||L_sensor_value==0){
    R_sensor_value = digitalRead(A3);
    L_sensor_value = digitalRead(A5);
  if(R_sensor_value==0&&L_sensor_value!=0){
    digitalWrite(12,LOW);
    //analogWrite(9,0);


    }
  else if(L_sensor_value==0&&R_sensor_value!=0){
    digitalWrite(11,LOW);
    //analogWrite(10,0);
    }
    else if(L_sensor_value==0&&R_sensor_value==0){
    turn_in_branch();
    }
  }
}

void StartUp(){
  int B_sensor_value = digitalRead(A4);
  int R_sensor_value = digitalRead(A3);
  int L_sensor_value = digitalRead(A5);
  if(B_sensor_value==0&&R_sensor_value==0&&L_sensor_value==0){
    start = 1;
    digitalWrite(12,HIGH);
    digitalWrite(11,HIGH);
    analogWrite(9,velocity);
    analogWrite(10,velocity);
    delay(1500);// may be changed
    }
}

void Stop(){
  int B_sensor_value = digitalRead(A4);
  if(B_sensor_value==0){
    start = 0;
    digitalWrite(12,LOW);
    digitalWrite(11,LOW);
    bool stop = 0;
    while(stop==0){
    int R_sensor_value = digitalRead(A3);
    int L_sensor_value = digitalRead(A5);
    if(R_sensor_value==0||L_sensor_value==0){
      analogWrite(9,0);
      analogWrite(10,0);
      stop = 1;
    }
    }
    }
  }
void loop() {
  // put your main code here, to run repeatedly:
  StartUp();
  if(start==1){
    digitalWrite(12,HIGH);
    digitalWrite(11,HIGH);
    analogWrite(9,velocity);
    analogWrite(10,velocity);
    Adjust();
    Stop();
  }


}
```

2. Second version code (the perfect run one):

```
bool start = 0;
int num_of_branch = 0;
int velocity = 80;// may be changed
int velocity_in_branch = 0;// may be changed
void setup() {
  // put your setup code here, to run once:
  pinMode(12,OUTPUT);//R_dir
  pinMode(11,OUTPUT);//L_dir
  pinMode(10,OUTPUT);//L_PWM
  pinMode(9,OUTPUT);//R_PWM
  pinMode(A3,INPUT);//R_sensor
  pinMode(A4,INPUT);//B_sensor
  pinMode(A5,INPUT);//L_sensor
  // hepler output pins( to run the code in my car )
}

void turn_in_branch(){
  int R_sensor_value = digitalRead(A3);
  int L_sensor_value = digitalRead(A5);
```

```
if(R_sensor_value==0&&L_sensor_value==0){
while(R_sensor_value==0&&L_sensor_value==0){
  if(num_of_branch == 0){
    digitalWrite(12,HIGH);
    digitalWrite(11,HIGH);
    analogWrite(10,velocity_in_branch);
    }
   else if(num_of_branch == 1){
    digitalWrite(12,HIGH);
 digitalWrite(11,HIGH);
    analogWrite(10,velocity_in_branch);
    }
   else if(num_of_branch == 2){
    digitalWrite(12,HIGH);
 digitalWrite(11,HIGH);
    analogWrite(9,velocity_in_branch);
    }
   else{
    //analogWrite(9,velocity);
    //analogWrite(10,velocity);
    }
  R_sensor_value = digitalRead(A3);
  L_sensor_value = digitalRead(A5);
  }
num_of_branch++;
analogWrite(9,velocity);
analogWrite(10,velocity);
}
  }

void Adjust(){
  int R_sensor_value = digitalRead(A3);
  int L_sensor_value = digitalRead(A5);
  while(R_sensor_value==0||L_sensor_value==0){
    R_sensor_value = digitalRead(A3);
    L_sensor_value = digitalRead(A5);
  if(R_sensor_value==0&&L_sensor_value!=0){
    digitalWrite(12,LOW);
    //analogWrite(9,0);

    }
  else if(L_sensor_value==0&&R_sensor_value!=0){
    digitalWrite(11,LOW);
    //analogWrite(10,0);
    }
    else if(L_sensor_value==0&&R_sensor_value==0){
    turn_in_branch();
    }
  }
}

void StartUp(){
  int B_sensor_value = digitalRead(A4);
  int R_sensor_value = digitalRead(A3);
  int L_sensor_value = digitalRead(A5);
  if(B_sensor_value==0&&R_sensor_value==0&&L_sensor_value==0){
    start = 1;
    digitalWrite(12,HIGH);
    digitalWrite(11,HIGH);
    analogWrite(9,velocity);
    analogWrite(10,velocity);
    delay(1500);// may be changed
    }
}

void Stop(){
  int B_sensor_value = digitalRead(A4);
  if(B_sensor_value==0){
    start = 0;
    digitalWrite(12,LOW);
    digitalWrite(11,LOW);
    bool stop = 0;
    while(stop==0){
    int R_sensor_value = digitalRead(A3);
    int L_sensor_value = digitalRead(A5);
    if(R_sensor_value==0||L_sensor_value==0){
      analogWrite(9,0);
      analogWrite(10,0);
      stop = 1;
    }
    }
    }
  }
void loop() {
  // put your main code here, to run repeatedly:
  StartUp();
  if(start==1){
    digitalWrite(12,HIGH);
    digitalWrite(11,HIGH);
    analogWrite(9,velocity);
    analogWrite(10,velocity);
    Adjust();
    Stop();
  }

}
```