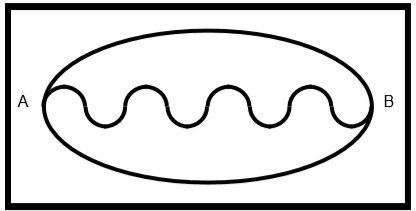
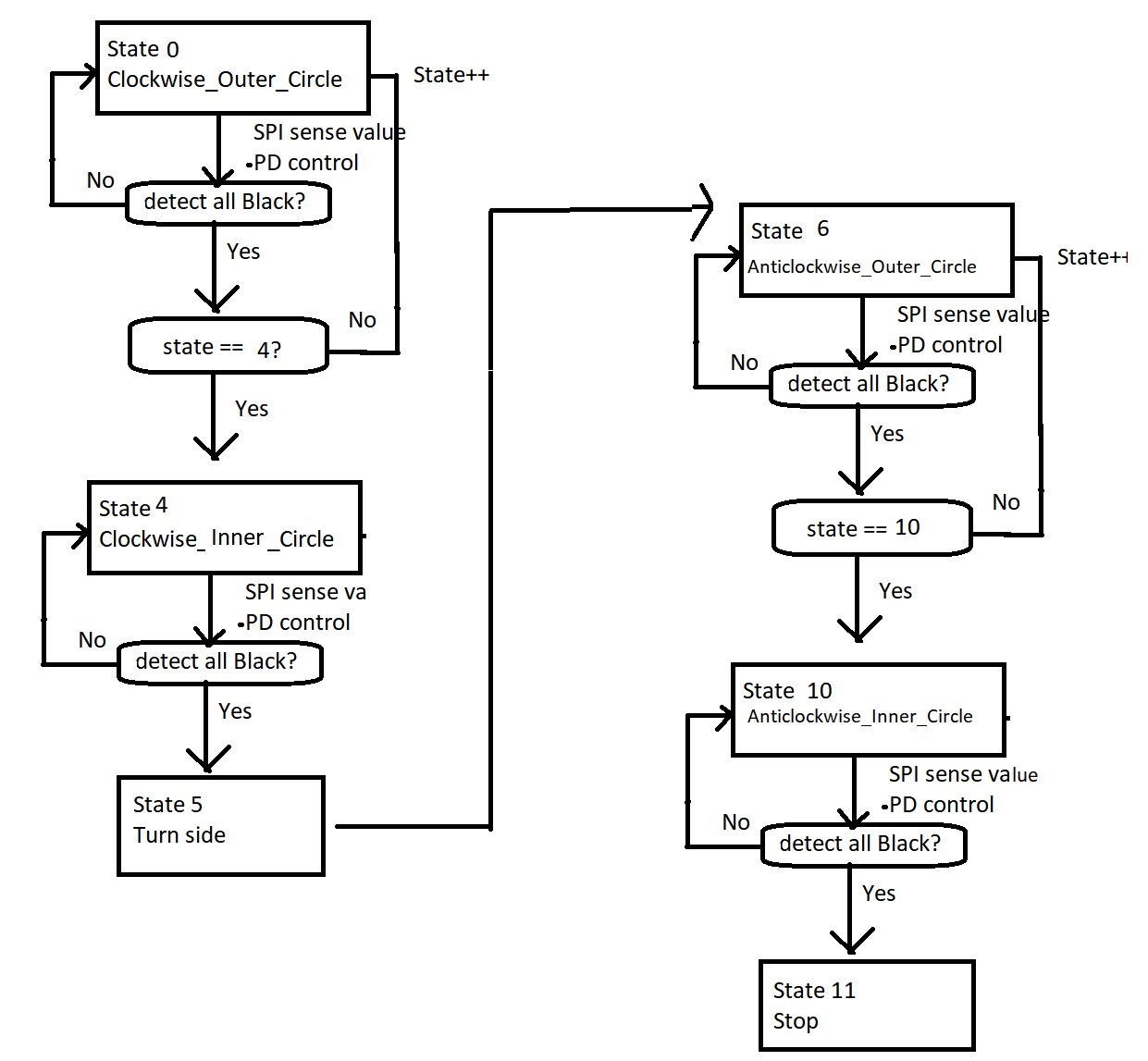
Instruction Manual

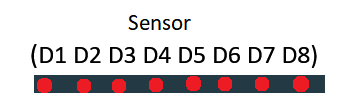
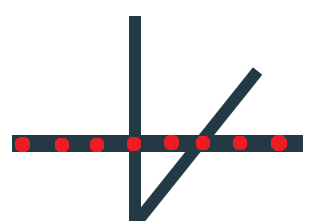
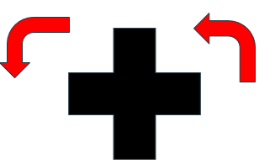
**Demonstration 2**

Requirement

Program the robot car to use PID control technique so that the car can follow the track automatically. The arena of the track is shown:

1. Initial Setting
   1. LED: Always on. When detected all black, turn off.
   2. SysTick: set as 1/10000 s= 0.1ms
   3. SPI2: get 16bits data and extract into 8bits sensor value
   4. Tim3: duty cycle around 40%
2. Program Flow

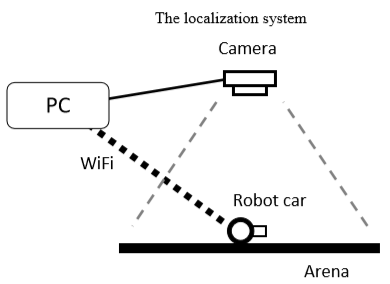
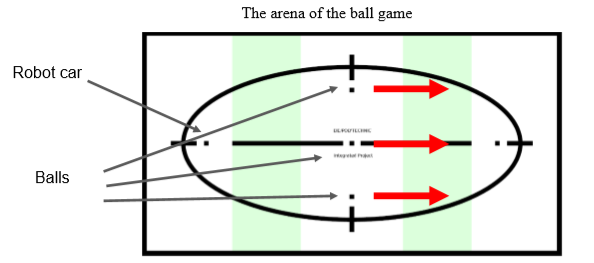


1. Implementation
   1. Error ratio
      1. error is given by the photo transistors reading
      2. (-8 -3 -2 -1 1 2 3 8) as the sensor degree error ratio for State Outer\_Circle
      3. (-8 -4 -2 -1 1 2 4 8) as the sensor degree error ratio for State Inner\_Circle
      4. Higher ratio for larger turn, i.e. outermost sensor has higher weighting of its reading
   2. Stability control
      1. Higher absolute value of error gives a more powerful turn due to the adjustment of PWM which depended on error
      2. Use P control to minimize response time
      3. Use D control to minimize overflow and vacillate
   3. Turning control
      1. To allow the car enters the inner track, priority of the sensor is changed
      2. E.g. at state 4, right sensor gives higher priority to right first
      3. If right sensor is detected, the program will not calculate others sensor error value
      4. the error is only given by the right sensor value, hence the car turns right
      5. Outer circle-> middle sensor higher priority -> detect the middle first
      6. Inner circle-> left & right sensor higher priority -> detect the left or right first
2. Difficulties Encounter
   1. Getting Sensor value at the beginning
      1. the sensor bit is shifted when using SPI2 8bit data size
      2. changed to 16bit data size and extract 8bit useful bit
   2. Difficulty in State 5: Turn side (Clockwise->Anticlockwise)
      1. time and speed are unknown, hard to control a 180-degree phase turn
      2. make use middle sensor to check the black line two times in state 5
      3. turn 90-degree first, then turn another-90 degree

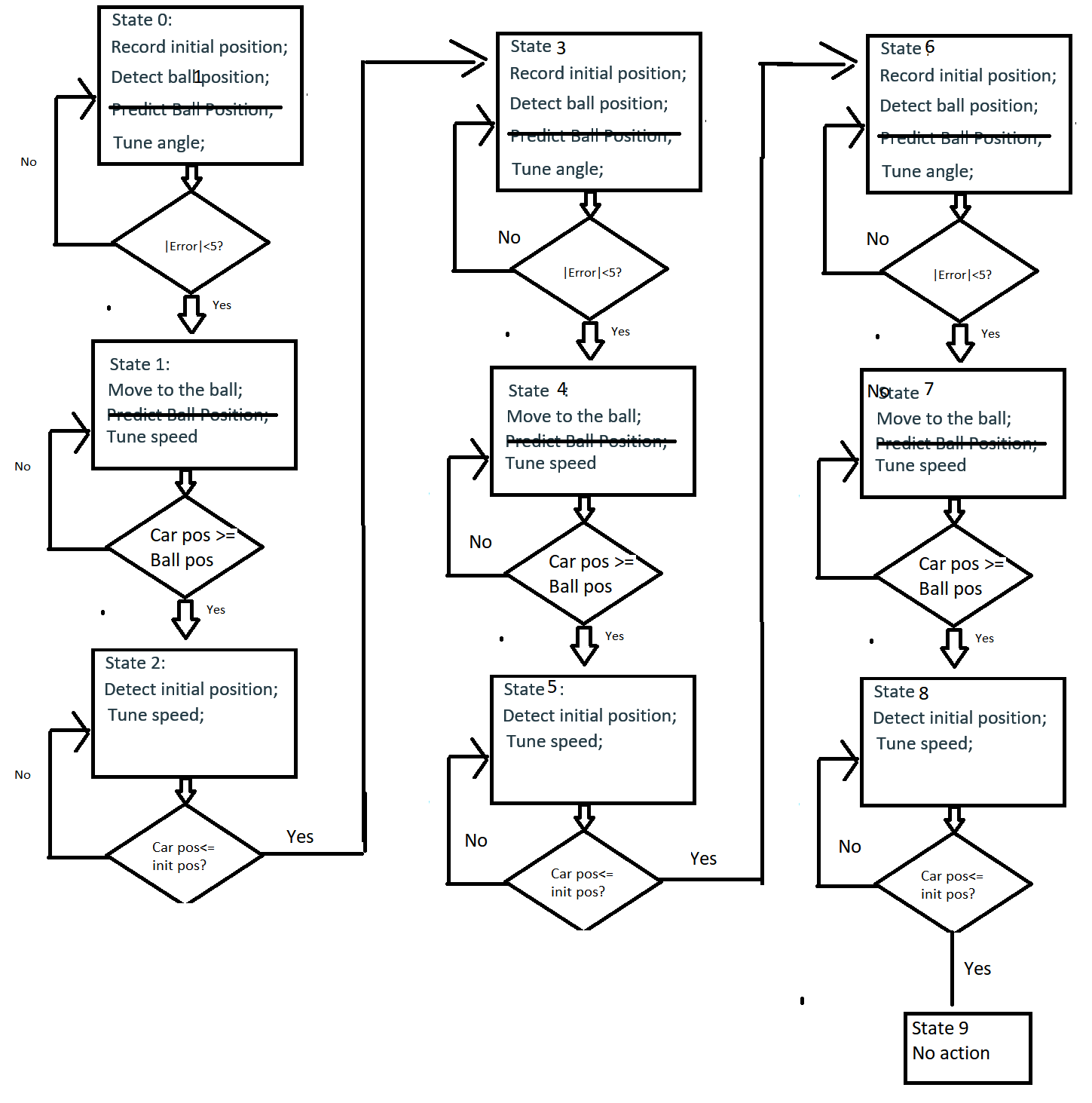
**Demonstration 3**

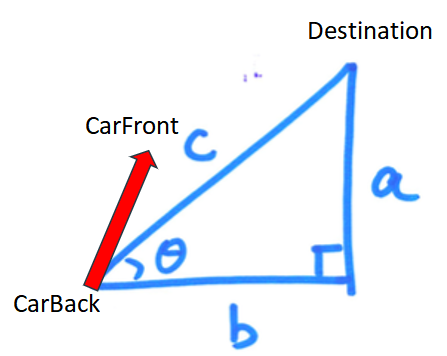
Requirement

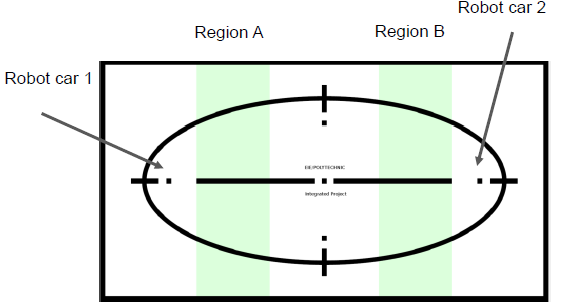
The robot car is required to use the information got from WiFi system to hit three balls automatically so that they can move to the same green zone. The positions of the robot car and the ball can be identified by a WiFi system. A camera is connected to a PC and an application program in the PC can send out the positions (a string) to each robot car through WiFi.



1. Initial Setting
   1. LED: On when three position received (Ball, CarA, CarB); Off when buffer cleared.
   2. SysTick: set as 1/1000 s= 1ms
   3. Usart2: receive char from Wi-Fi, 115200 baud rate
   4. Tim3: duty cycle around 20%
2. Program Flow



1. Implementation
   1. Position Get
      1. Check ‘:’ to clear previous buffer value and ready to store new one
      2. Check ‘CBD’ (car’s filter name) to get data from Wi-Fi and store to string
      3. Extract the hexanal position from string, convert into decimal by lookup table
   2. Stability control
      1. Use P control to minimize response time
      2. Use I control to approach the offset value
      3. Use D control to minimize overflow and vacillate
   3. Error ratio
      1. Use Pythagorean Theorem to calculate hypotenuse (distance)
      2. calculate ratio of Δy / distance, which is the sinθ
2. Difficulties Encounter
   1. Calculate the angle
      1. We tried to use atan function from math.h to directly calculate the angle, but it has runtime error and the program run abnormally. As the function use Taylor series to simulate the angle, which takes time. (Solution: calculate sin(theta) as angle value)
   2. Adjust PID parameter
      1. We spent time to adjust the parameter to stable the system of car.
   3. Filter unstable data
      1. Failure to capture the coordinate due to brightness or coordinate of car close to side (Solution: adjust the parameter of filter by placing the object at different places)

**Demonstration 4**

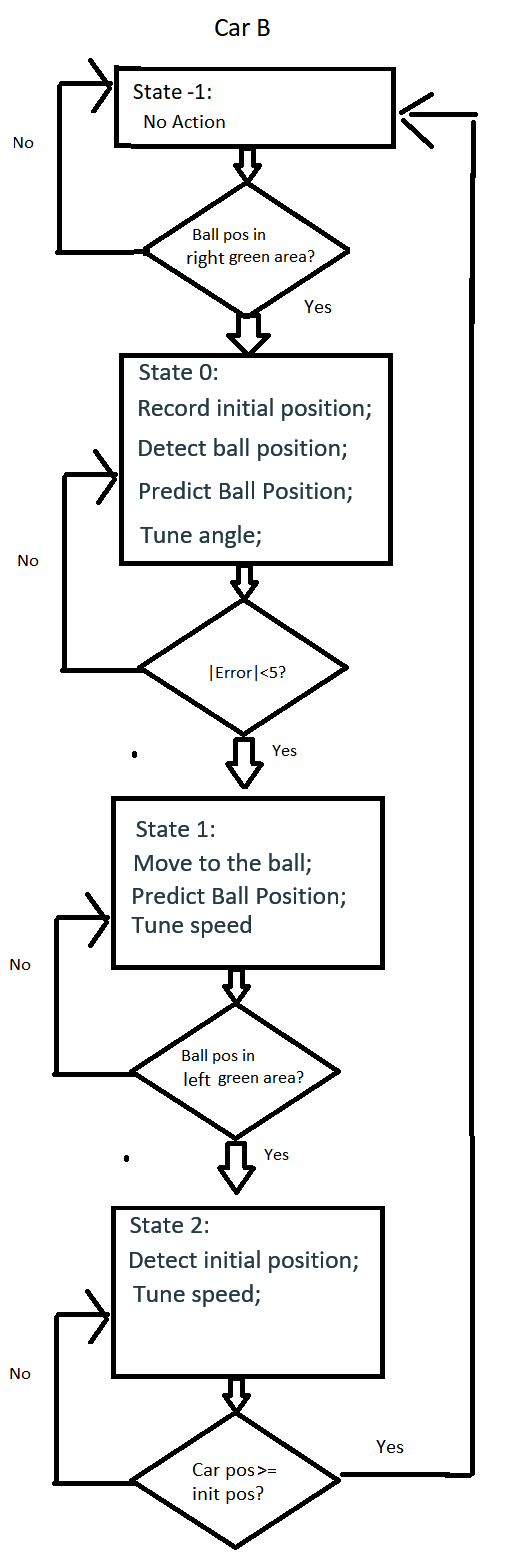
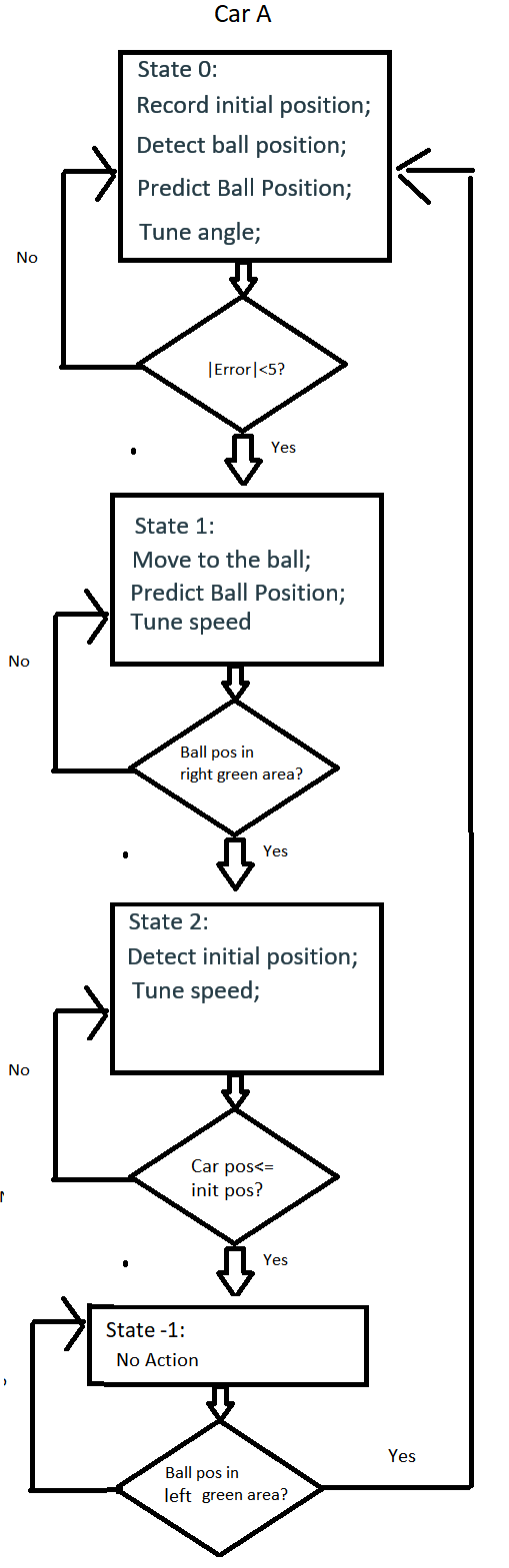
Requirement

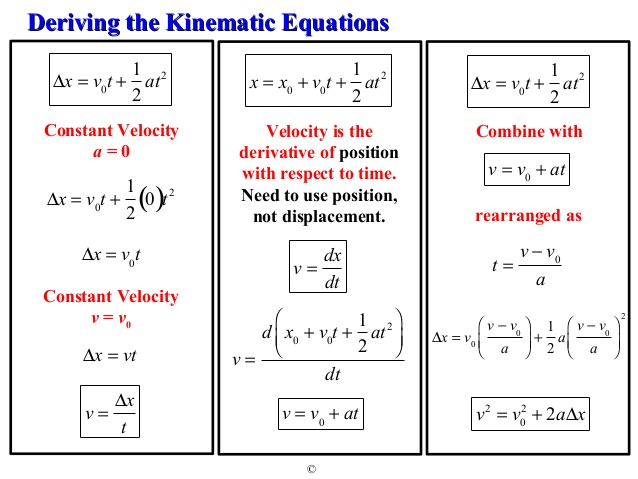
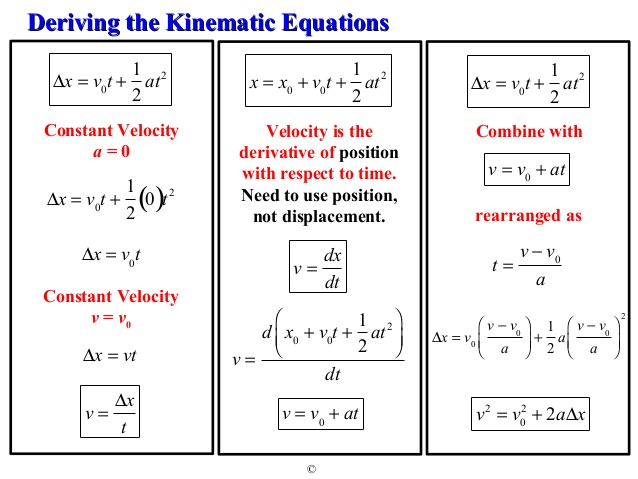
1. Hit the ball to move it from Region A to Region B by Robot Car 1.

2. Hit the ball to move it from Region B to Region A by Robot Car 2.

3. The ball moves from one green area to another green area four times.

1. Initial Setting
   1. Same as Demonstration 3
2. Program Flow



1. Implementation
   1. New: Prediction of Ball
   2. By using algorithm to predict ball after 1 time period
   3. 
2. Difficulties Encounter
   1. Mostly Hardware problem
   2. Some parts in the left gear are loosen, which stuck the gear movement
   3. Although PWM given to the car are the same, the torque is not enough to move one wheel, making the PID tuning very difficult
   4. To solve the problem, I add a small piece of paper in between the gear, to increase the friction of the other parts
   5. The two motors both need a strong PWM to drive, hence having a equal turning value
   6. The drawbacks of it is I cannot tune the car in high speed, as the friction to the gear is unstable

