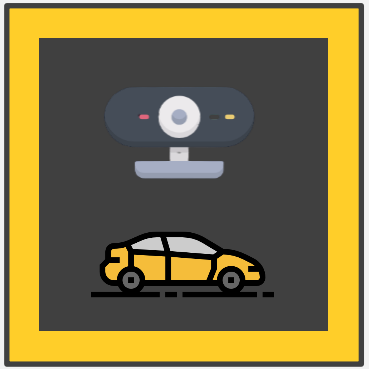

# 오토믹 자동차

Android 앱을 이용한 주행

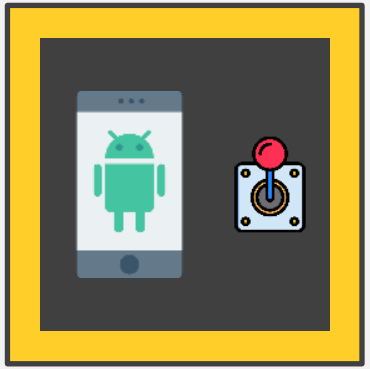
서강원
김기영
김채진
옥진해
전고은

# Contents

# 목표



Line Tracking



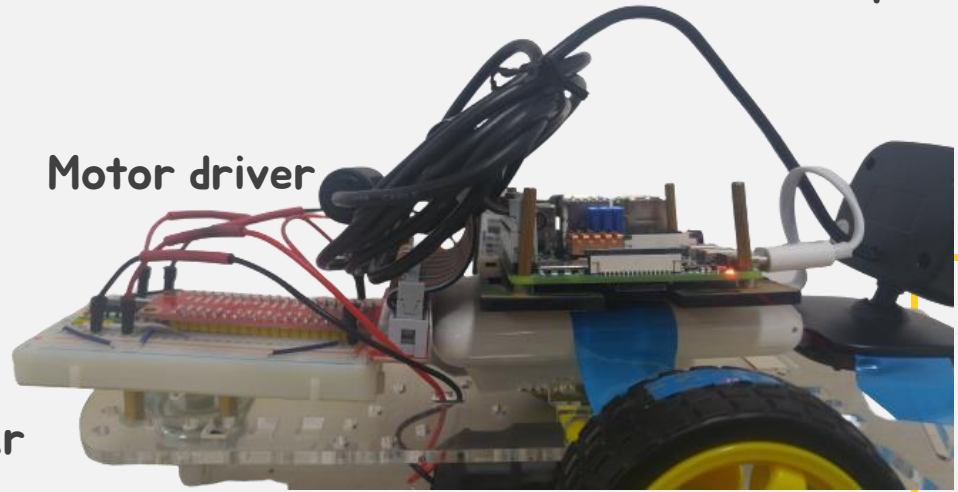
Joystick Controller

# 개발 환경 및 부품

배터리 (5V 1A)

Raspberry Pi 3

Raspbian OS

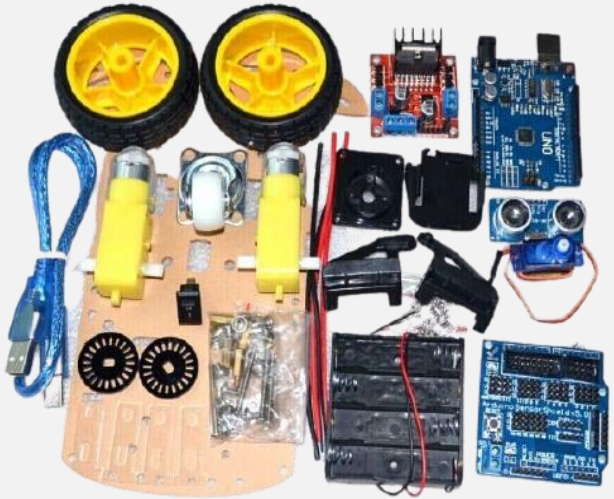


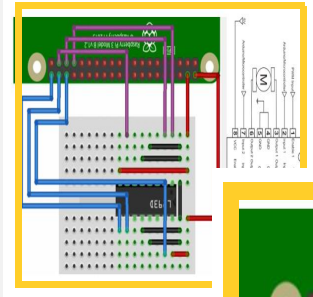
Motor driver

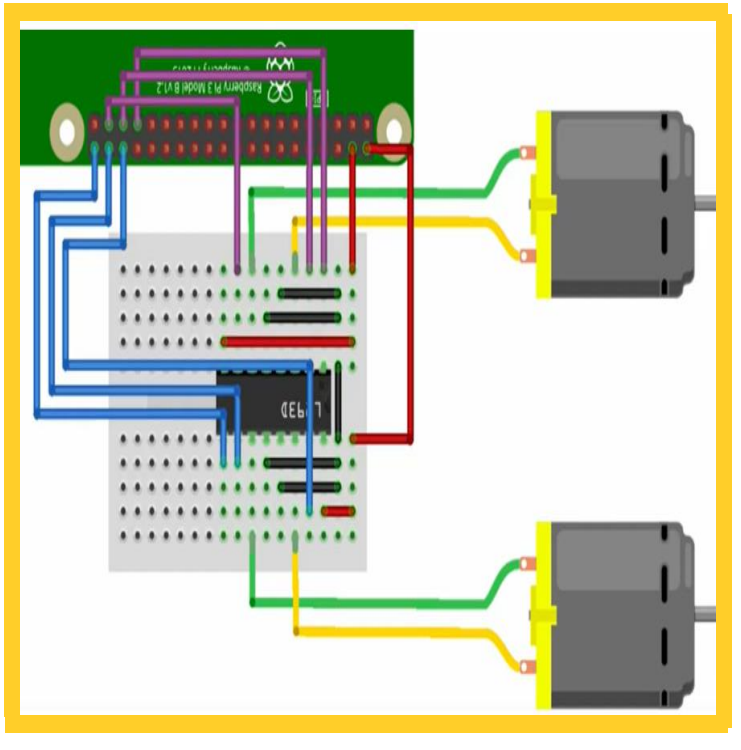
WebCam

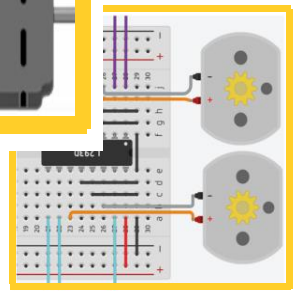
RC Car

# 회로 연결 과정



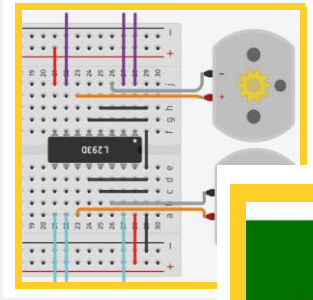



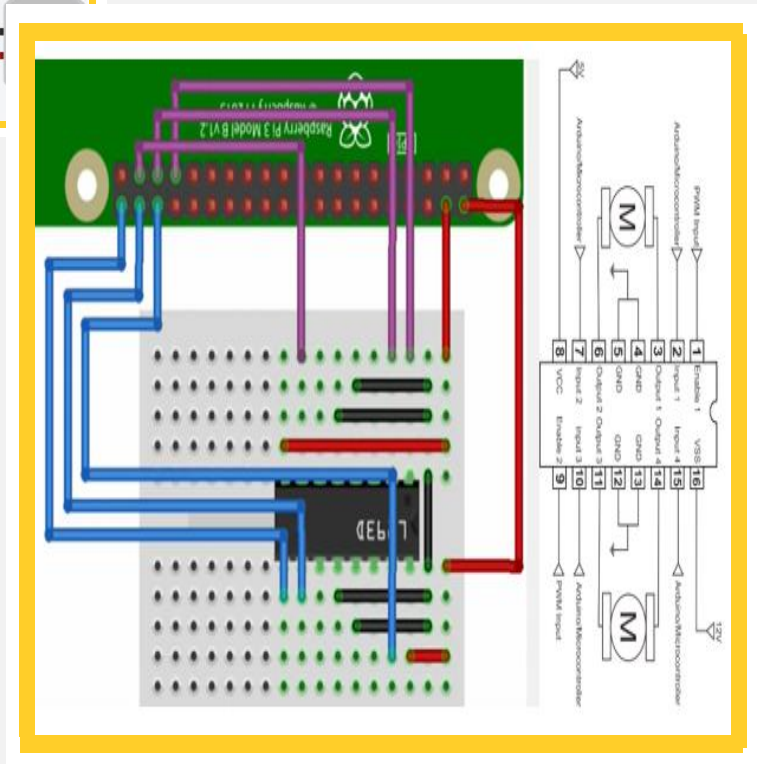


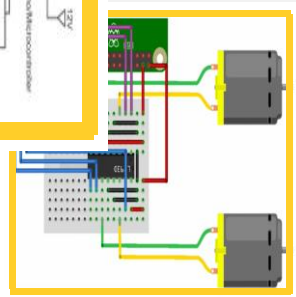



# 회로 연결 과정

| Rpi | L293D | Direction |
|---|---|---|
| 5V | 8,16 | |
| GND | 4,5,12,13 | |
| GPIO13 | 1 | LEFT_PWM |
| GPIO19 | 2 | LEFT_FORWARD |
| GPIO26 | 7 | LEFT_BACKWARD |
| GPIO21 | 9 | RIGHT_PWM |
| GPIO16 | 15 | RIGHT_FORWARD |
| GPIO20 | 10 | RIGHT_BACKWARD |

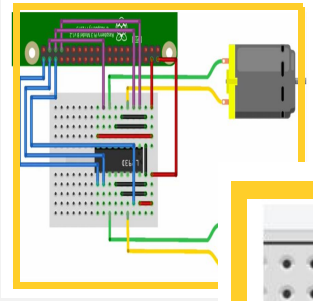


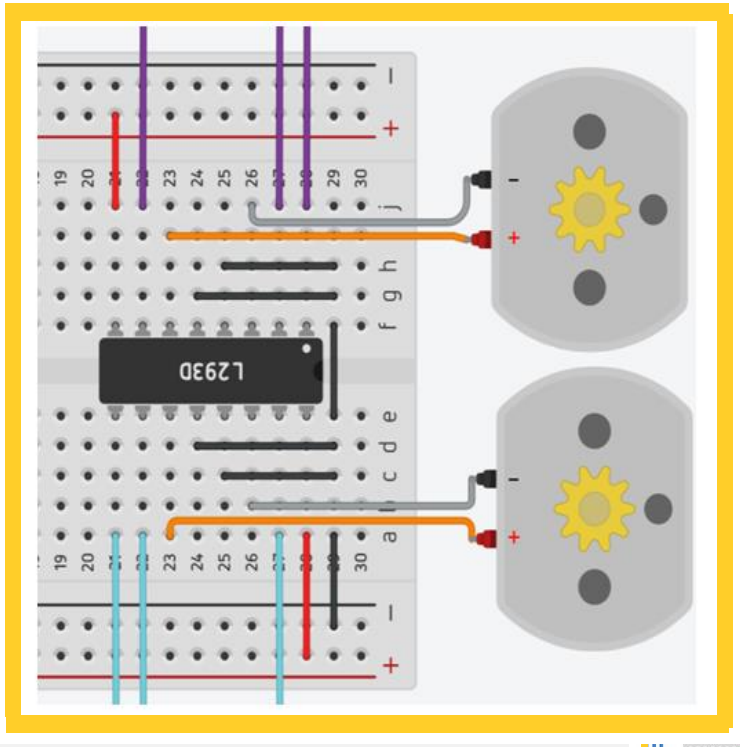


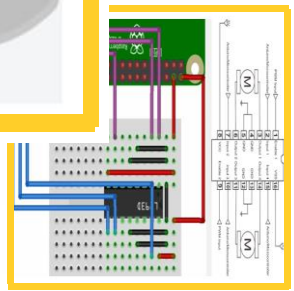


# 회로 연결 과정

| Rpi | L293D | Motor |
|---|---|---|
| 5V | 6 | 5V(Orange) |
| GND | 3 | GND(Gray) |
| 5V | 11 | 5V(Orange) |
| GND | 14 | GND(Gray) |

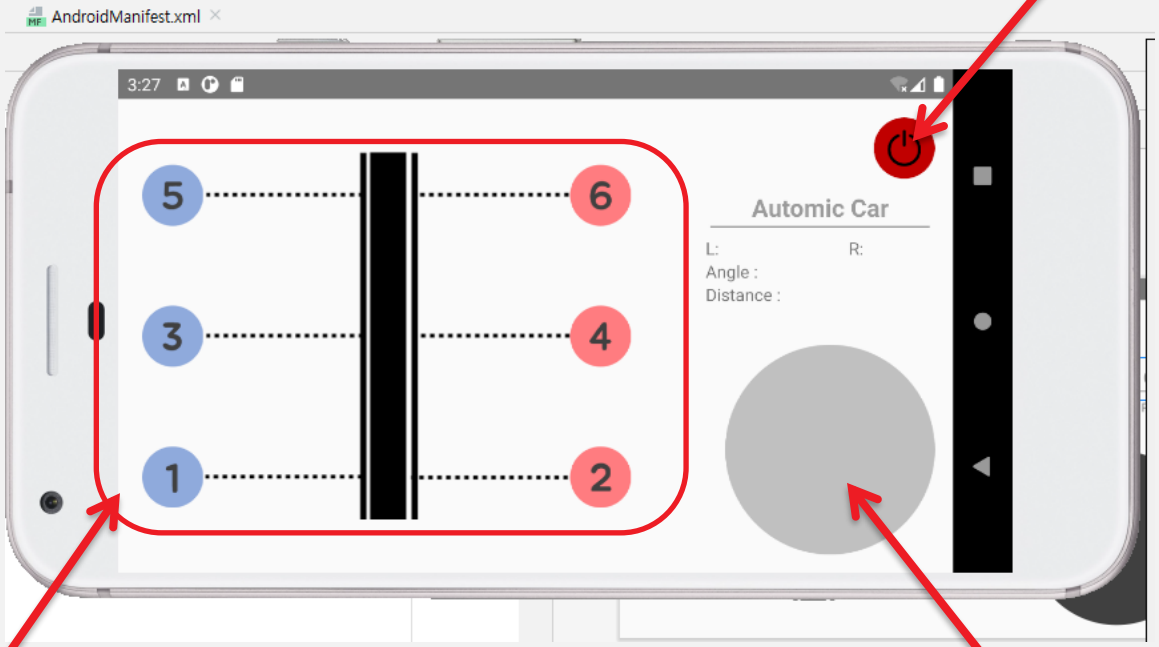




# Application

종료 버튼 (실행 종료)




Line Tracking 버튼

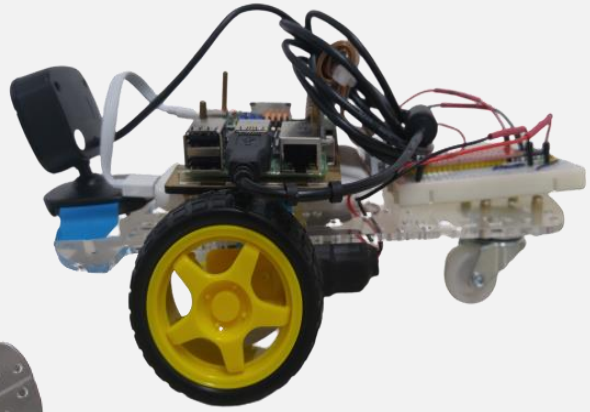
Joystick Controller

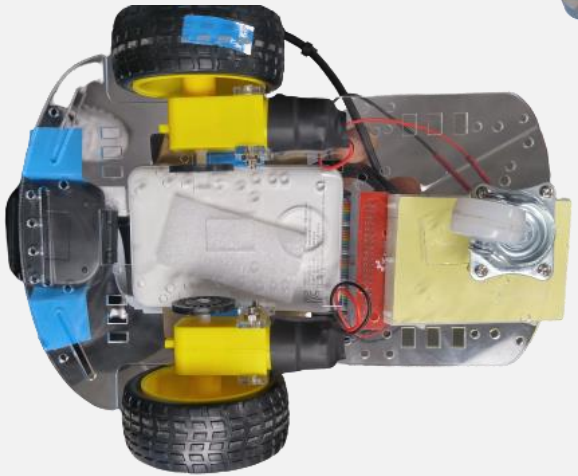
# H/W의 중요성

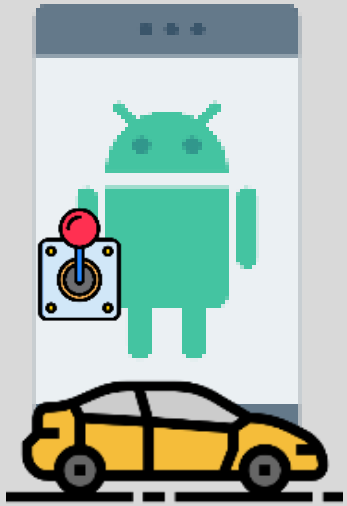
i) 모터드라이버
- 성능에 따른 불균형

ii) 무게중심
- 뒷 바퀴 제어

iii) 카메라 영상 처리
- 카운트







# Joystick

# Joystick

```
layout_joystick.setOnTouchListener(new View.OnTouchListener() {
            @Override
            public boolean onTouch(View v, MotionEvent event) {
                js.drawStick(event);
                textAngle.setText("Angle : " + String.valueOf(js.getAngle()));
                textDistance.setText("Distance : " +
String.valueOf(js.getDistance()));
                int valueX =js.getX(), valueY = js.getY();
                Float angle = js.getAngle();
                //int distance = 0; //js.getDistance();
                float distance = js.getDistance();
                String L = "", R = "";
                double numL, numR;
```

# Joystick

```
if (event.getAction() == MotionEvent.ACTION_MOVE) {
                String direction = js.get2Direction();
                textDirection.setText("valueX : " + String.valueOf(valueX));
                if (valueY <= 0) {
                    if (valueX >= 0) { //F_R
                        numL = (distance / 3);
                        L = String.valueOf(Math.round(numL));
                        numR = ((360 - angle) * 1.11 * (distance / 280));
                        R = String.valueOf(Math.round(numR));
                    } else { //F_L
                        numL = ((angle - 180) * 1.11 * (distance / 280));
                        L = String.valueOf(Math.round(numL));
                        numR = (distance / 3);
                        R = String.valueOf(Math.round(numR));
                    }
                }
```

# Joystick

```
else{
                    if (valueX >= 0) { //Back_R
                        numL = (distance / 3);
                        L = String.valueOf(Math.round(numL));
                        numR = (angle * 1.11 * (distance / 280));
                        R = String.valueOf(Math.round(numR));
                    } else { //Back_L
                        numL = ((180-angle) * 1.11 * (distance / 280));
                        L = String.valueOf(Math.round(numL));
                        numR =  (distance / 3);
                        R = String.valueOf(Math.round(numR));
                    }
                }
```

# Joystick

```
textSend.setText("L:  "+L + " R: " + R);
                    mSendData = new SendData();
                    CMD = direction+","+L+","+R;
                                        //보내기 시작
                    mSendData.start();
                }
                if (valueX == 0 && valueY ==0){
                    mSendData = new SendData();
                    textSend.setText("L: 0" + " R: 0");
                    CMD = "S"+",0, 0";
                                        //보내기 시작
                    mSendData.start();
                }
                return true;
            }
        });
```

# Joystick

CARServer_UDP.py

```
# CARServer_UDP.py
import CARControl_UDP
from socket import *
from time import ctime
import RPi.GPIO as GPIO
import sys

ctrCmd = ["F","B","S","P"]
cmd = []

HOST = ''
PORT = 8011
BUFSIZE = 1024
ADDR = (HOST,PORT)

# 소켓 생성 , 포트 설정
udpSerSock = socket(AF_INET, SOCK_DGRAM)
udpSerSock.bind(ADDR)
```

# Joystick

CARServer_UDP.py


```
while True:
        data, addr = udpSerSock.recvfrom(BUFSIZE)
        print('...connected from :', addr)
        try:
                while True:
                    data = “”
                    data = udpSerSock.recv(BUFSIZE);
                    data = data.decode('utf-8')
                    print(data)
                    cmd = data.split(',’)
```

# Joystick

CARServer_UDP.py

```
            if not data:
                break
            if cmd[0] == ctrCmd[0]:
                CARControl_UDP.Forward(int(cmd[1]), int(cmd[2]))
                print("Forward")
            if cmd[0] == ctrCmd[1]:
                CARControl_UDP.Backward(int(cmd[1]), int(cmd[2]))
                print("Backward")
            if cmd[0] == ctrCmd[2]:
                CARControl_UDP.Stop()
                print("Stop")
            if cmd[0] == ctrCmd[3]:
                print("Poweroff")
                CARControl_UDP.Poweroff()
                break
```

# Joystick


CARControl_UDP.py

```
def Forward(speed_L, speed_R):
    rightMotor(1 ,0, speed_R)
    leftMotor(1 ,0, speed_L)
    cmd_text = "FORWARD"

def Backward(speed_L, speed_R):
    rightMotor(0 ,1, speed_R)
    leftMotor(0 ,1, speed_L)
    cmd_text = "BACKWARD"

def Stop():
    rightMotor(0 ,0, 0)
    leftMotor(0 ,0, 0)
```

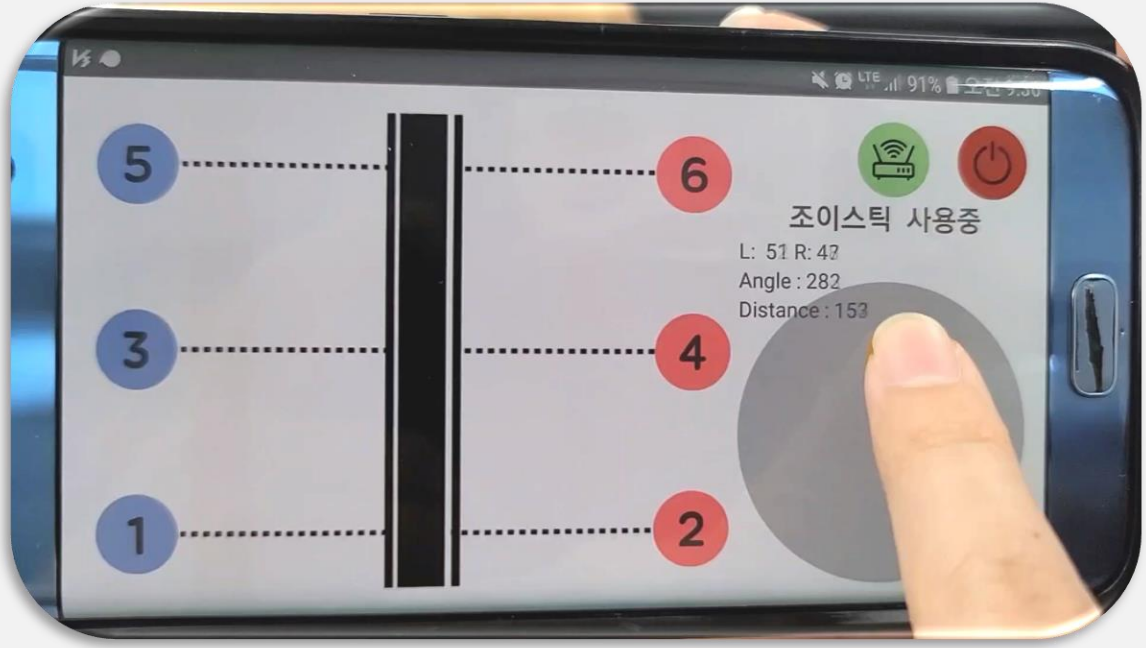
# Joystick

CARControl_UDP.py

```
def Forward(speed_L, speed_R):
    rightMotor(1 ,0, speed_R)
    leftMotor(1 ,0, speed_L)
    cmd_text = "FORWARD"

def Backward(speed_L, speed_R):
    rightMo
    leftMot
    cmd_te

def Stop(
    rightMo
    leftMot
```

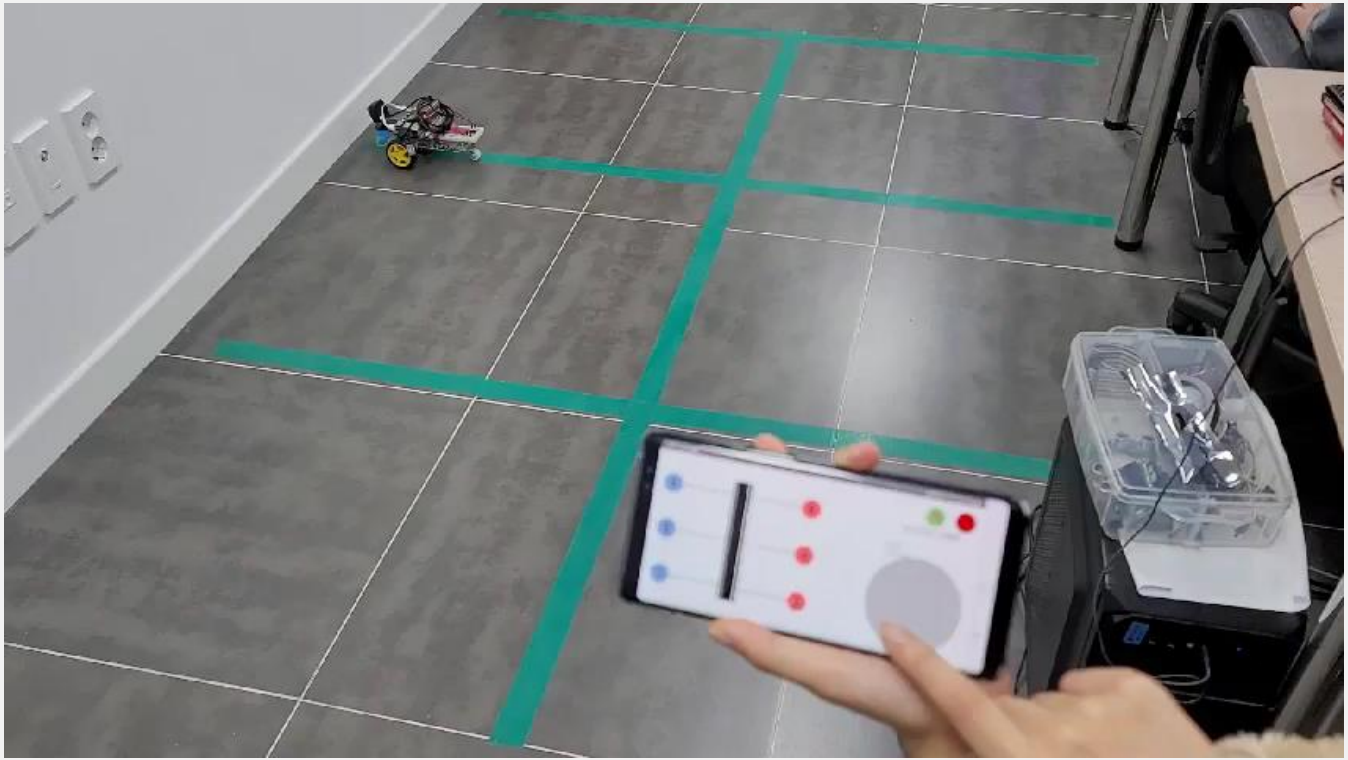
```
RIGHT_MOTOR =
GPIO.PWM(RIGHT_PWM,100

def rightMotor(forward, backward, pwm):
    GPIO.output(RIGHT_FORWARD,forward)
    GPIO.output(RIGHT_BACKWARD,backward)
    RIGHT_MOTOR.ChangeDutyCycle(pwm)
```

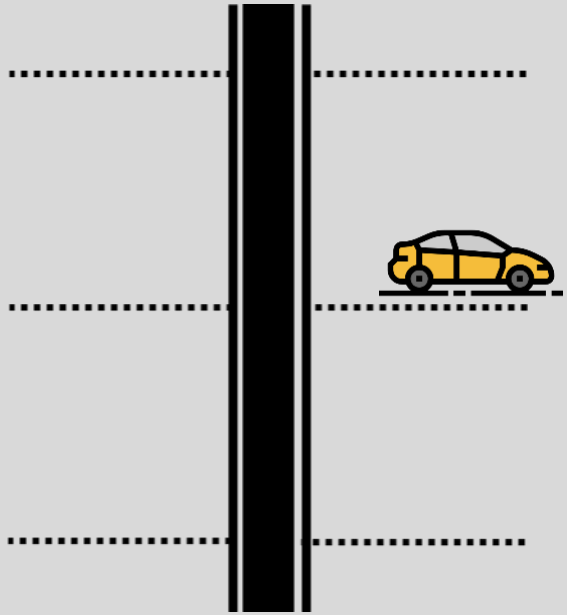
# Joystick



# Joystick

# 촬영 영상

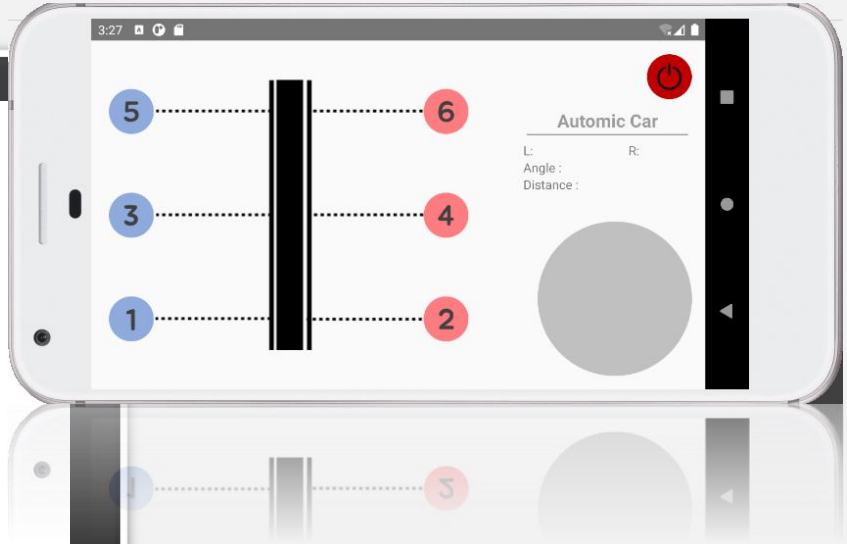




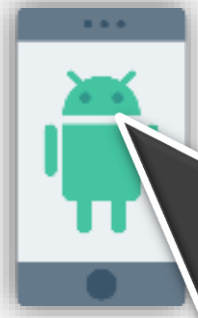
# Line Tracking

# Line Tracking

```
btn1.setOnClickListener(new
View.OnClickListener() {
    @Override
    public void onClick(View v) {
        mSendData = new SendData();
        CMD = "1,_";
        editText.setText("운행중");
        //보내기 시작
        mSendData.start();
        btnCancel.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View v) {
                dialog.dismiss();
            }
        });
        dialog.show();
    }
});
```

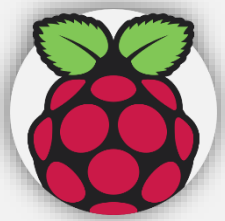


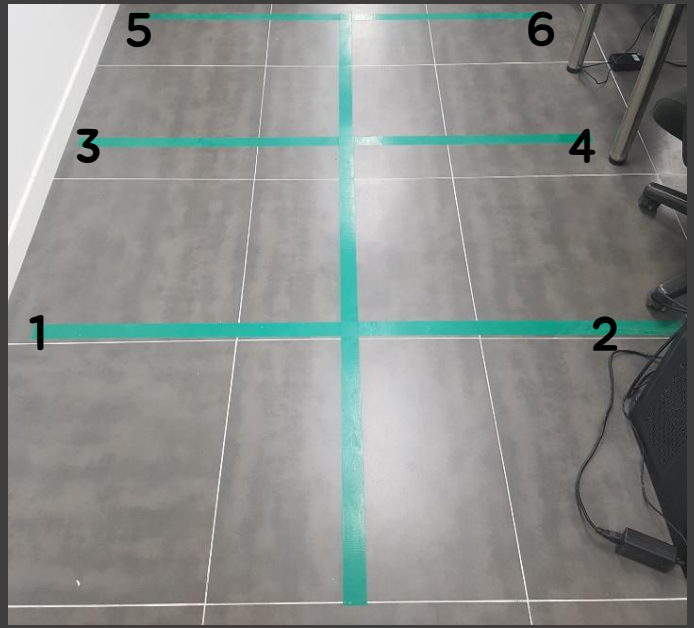
# Line Tracking



```
btn1.setOnClickListener(new
View.OnClickListener() {
    @Override
    public void onClick(View v) {
        mSendData = new SendData();
        CMD = "1,_";
        editText.setText("운행중");
        //보내기 시작
        mSendData.start();
        btnCancel.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View v) {
                dialog.dismiss();
            }
        });
        dialog.show();
    }
});
```

# Line Tracking

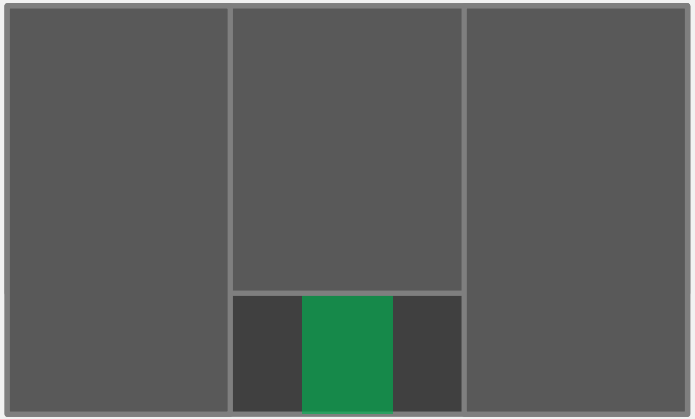




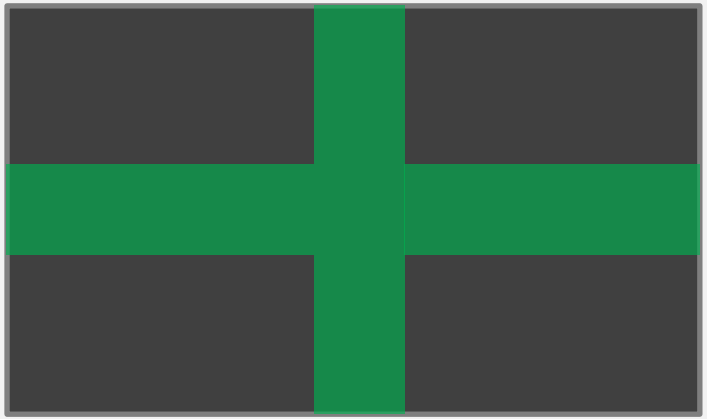
if i == 1 (좌측)

if i == 2 (우측)

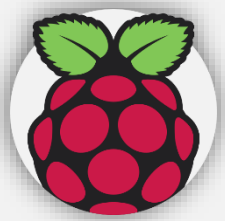
else: // i = 0 (중앙)



실제 화면



# Line Tracking

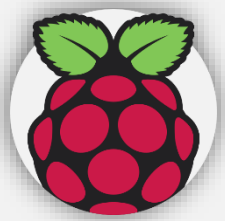


```
Color_Lower = (70,100,70)
Color_Upper = (92, 255, 255)
```

```
Frame_Width  = 320
Frame_Height = 240
```

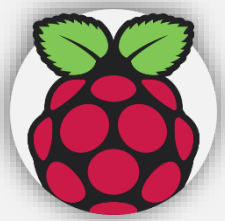
```
frame_list=[]# [center, left, right]
center = None
center_list=[None, None, None]
(_, frame) = camera.read()
for i in range(3):
    frame_list.append(frame)
    frame_list[i] = cv2.GaussianBlur(frame_list[i], (11, 11),1)
    hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)
    mask = cv2.inRange(hsv, Color_Lower, Color_Upper)
    if i==1:
        cv2.rectangle(mask, (40,0),(320,240),(0,0,0),-1)
    elif i==2:
        cv2.rectangle(mask, (0,0),(280,240),(0,0,0),-1)
    else:
        cv2.rectangle(mask, (0,0),(320,120),(0,0,0),-1)
        cv2.rectangle(mask, (0,0),(120,240),(0,0,0),-1)
        cv2.rectangle(mask, (240,0),(320,240),(0,0,0),-1)
```

# Line Tracking



```
frame_list=[]# [center, left, right]
center = None
center_list=[None, None, None]
(_, frame) = camera.read()
for i in range(3):
    frame_list.append(frame)
    frame_list[i] = cv2.GaussianBlur(frame_list[i], (11, 11),1)
    hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)
    mask = cv2.inRange(hsv, Color_Lower, Color_Upper)
    if i==1:
        cv2.rectangle(mask, (40,0),(320,240),(0,0,0),-1)
    elif i==2:
        cv2.rectangle(mask, (0,0),(280,240),(0,0,0),-1)
    else:
        cv2.rectangle(mask, (0,0),(320,120),(0,0,0),-1)
        cv2.rectangle(mask, (0,0),(120,240),(0,0,0),-1)
        cv2.rectangle(mask, (240,0),(320,240),(0,0,0),-1)
    contours,  = cv2.findContours(mask, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
```

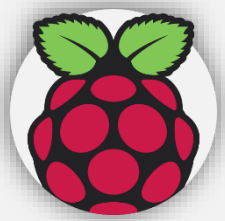
# Line Tracking

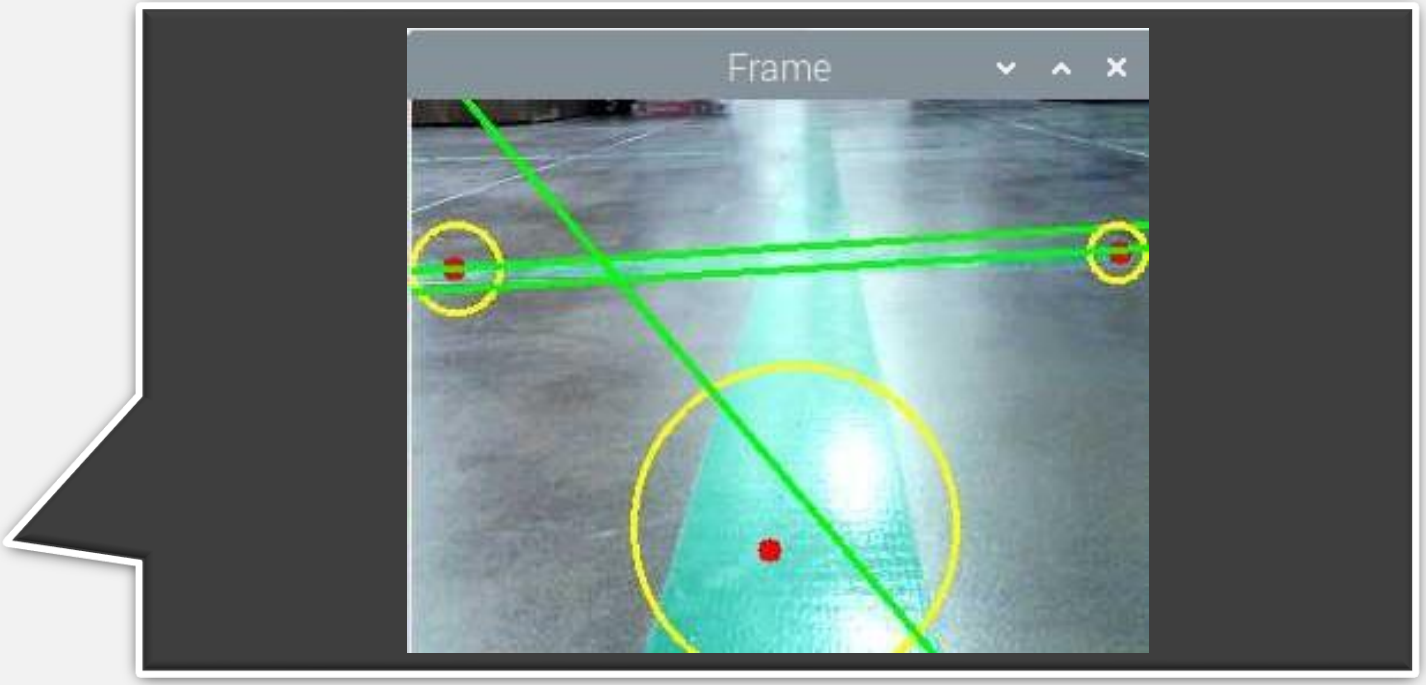


```
    draw(contours, center_list, frame)
```

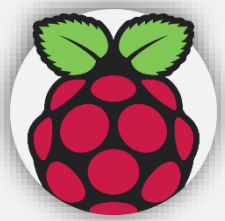
```
def draw(contours, center_list, frame):
    if len(contours) > 0:

        c = max(contours, key=cv2.contourArea)

        ((x, y), radius) = cv2.minEnclosingCircle(c)

        M = cv2.moments(c)
        try:

            center = (int(M["m10"] / M["m00"]), int(M["m01"] / M["m00"]))
            center_list[i] = center

            cv2.circle(frame, (int(x), int(y)), int(radius),(0, 255, 255), 2)
            cv2.circle(frame, center, 5, (0, 0, 255), -1)

            rows, cols = mask.shape[:2]
            [vx, vy, x,y] = cv2.fitLine(c, cv2.DIST_L2, 0, 0.01, 0.01)
            lefty = int((-x*vy/vx)+y)
            righty = int(((cols-x)*vy/vx)+y)
            cv2.line(frame, (cols-1,righty), (0, lefty), (0, 255,0),2)
        except:
            pass
```

# Line Tracking



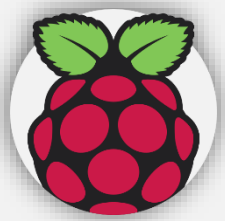


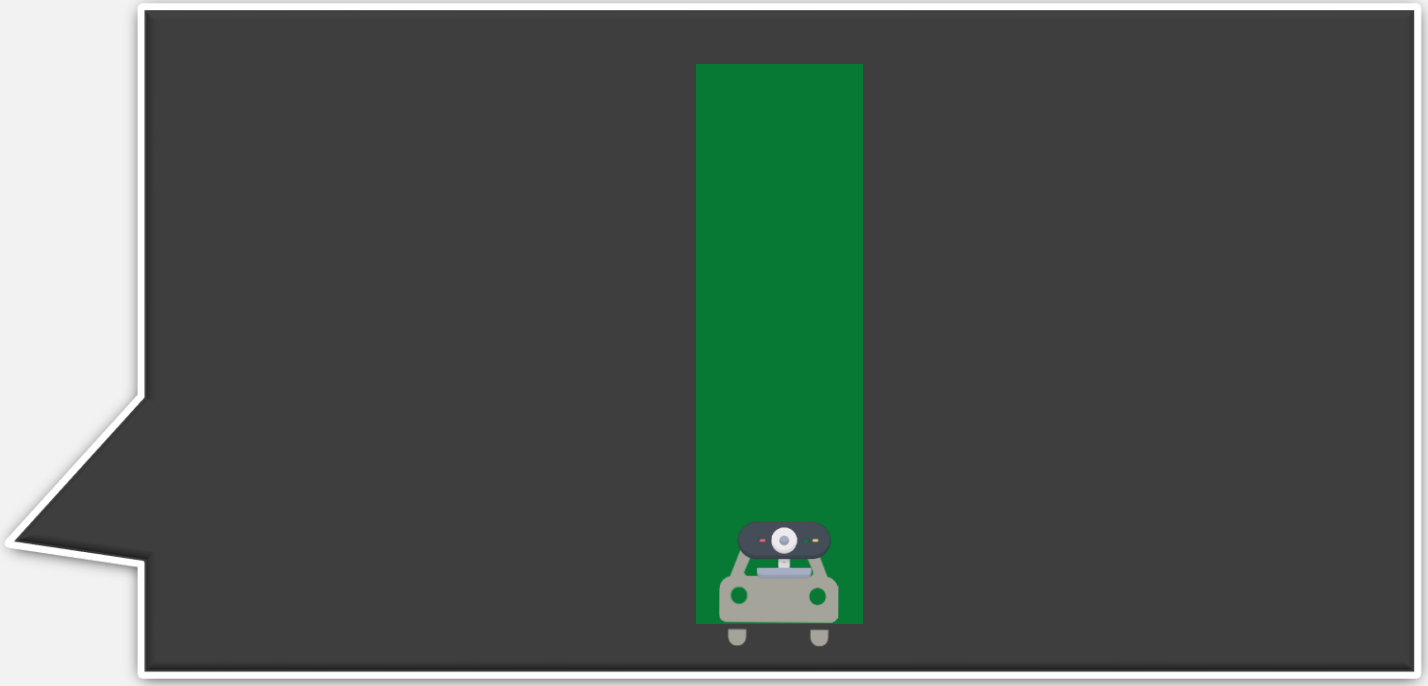
# Line Tracking



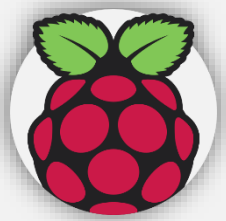
| center_list[0] | center_list[1] | center_list[2] |
|---|---|---|
| 중앙 | 좌측 | 우측 |
| (x,y) | (x,y) | (x,y) |

# Line Tracking

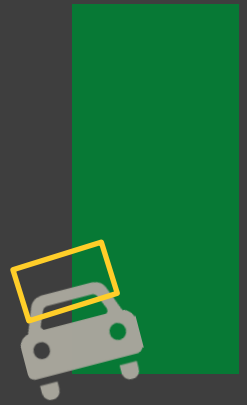




# Line Tracking



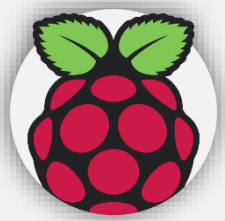
```
if center_list[0]!=None:
    if center_list[0][0] > Frame_Width/2 + 20 :  #turnRight_Area Set
        CARControl_UDP_ver2.TurnRight()
    elif center_list[0][0] < Frame_Width/2 -20 : #turnLeft_Area Set
        CARControl_UDP_ver2.TurnLeft()
```



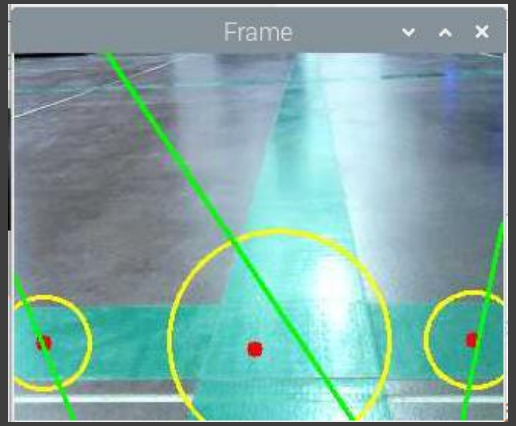
TurnRight()

```
def TurnRight(speed_L=100, speed_R=100):
    rightMotor(0 ,0, speed_R)
    leftMotor(1 ,0, speed_L)
```

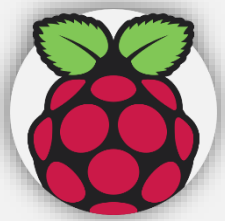
# Line Tracking



```
if center_list[1]!=None and center_list[2]!=None :
    if center_list[1][1] == Frame_Height/2
```

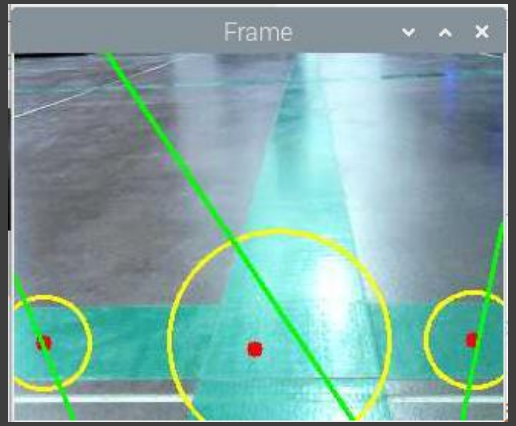


# Line Tracking

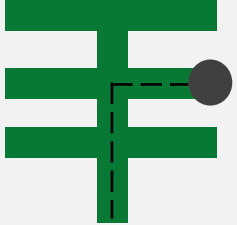


```
crossroad=0
flag=0
```

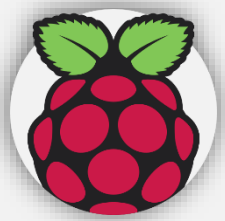
```
    else:
        if center_list[1]!=None and center_list[2]!=None :
            if center_list[1][1] > Frame_Height/2 and center_list[2][1] > Frame_Height/2 :
```



# Line Tracking



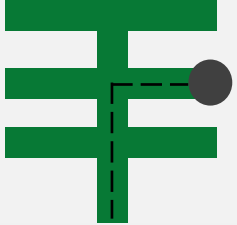
destination = 4



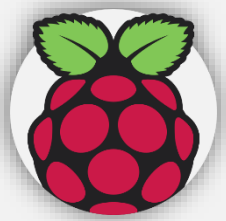
```
crossroad=0
flag=0
    else:
        if center_list[1]!=None and center_list[2]!=None :
            if center_list[1][1] > Frame_Height/2 and center_list[2][1] > Frame_Height/2 :
                if flag==0:
                    flag=1
                    crossroad+=1
                else:
                    if destination<3 and crossroad==1:
                        odd_even(int(cmd[0]))
                    elif destination<5 and crossroad==2:
                        odd_even(int(cmd[0]))
                    else:
                        if crossroad==3:
                            odd_even(int(cmd[0]))
            else:
                flag=0
        CARControl_UDP_ver2.Forward()
        print(crossroad)
        print(flag)
else:
    CARControl_UDP_ver2.Stop()
```

| crossroad | flag |
|---|---|
| 0 | 0 |

# Line Tracking



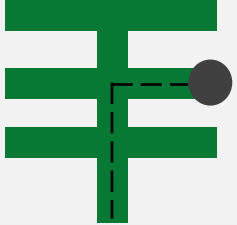
destination = 4



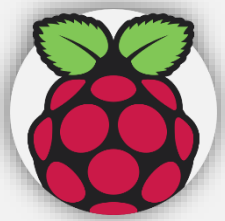
```
crossroad=0
flag=0
    else:
        if center_list[1]!=None and center_list[2]!=None :
            if center_list[1][1] > Frame_Height/2 and center_list[2][1] > Frame_Height/2 :
                if flag==0:
                    flag=1
                    crossroad+=1
                else:
                    if destination<3 and crossroad==1:
                        odd_even(int(cmd[0]))
                    elif destination<5 and crossroad==2:
                        odd_even(int(cmd[0]))
                    else:
                        if crossroad==3:
                            odd_even(int(cmd[0]))
            else:
                flag=0
        CARControl_UDP_ver2.Forward()
        print(crossroad)
        print(flag)
else:
    CARControl_UDP_ver2.Stop()
```

| crossroad | flag |
|---|---|
| 1 | 1 |

# Line Tracking



destination = 4

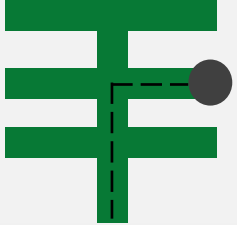


```
crossroad=0
flag=0
```

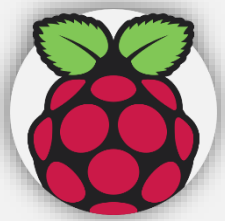
```
    else:
        if center_list[1]!=None and center_list[2]!=None :
            if center_list[1][1] > Frame_Height/2 and center_list[2][1] > Frame_Height/2 :
                if flag==0:
                    flag=1
                    crossroad+=1
                else:
                    if destination<3 and crossroad==1:
                        odd_even(int(cmd[0]))
                    elif destination<5 and crossroad==2:
                        odd_even(int(cmd[0]))
                    else:
                        if crossroad==3:
                            odd_even(int(cmd[0]))
            else:
                flag=0
        CARControl_UDP_ver2.Forward()
        print(crossroad)
        print(flag)
else:
    CARControl_UDP_ver2.Stop()
```

| crossroad | flag |
|---|---|
| 1 | 0 |

# Line Tracking



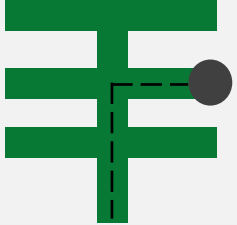
destination = 4



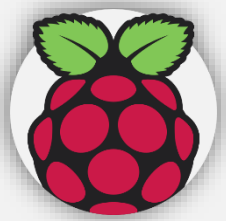
```
crossroad=0
flag=0
    else:
        if center_list[1]!=None and center_list[2]!=None :
            if center_list[1][1] > Frame_Height/2 and center_list[2][1] > Frame_Height/2 :
                if flag==0:
                    flag=1
                    crossroad+=1
                else:
                    if destination<3 and crossroad==1:
                        odd_even(int(cmd[0]))
                    elif destination<5 and crossroad==2:
                        odd_even(int(cmd[0]))
                    else:
                        if crossroad==3:
                            odd_even(int(cmd[0]))
            else:
                flag=0
        CARControl_UDP_ver2.Forward()
        print(crossroad)
        print(flag)
else:
    CARControl_UDP_ver2.Stop()
```

| crossroad | flag |
|---|---|
| 2 | 1 |

# Line Tracking



destination = 4



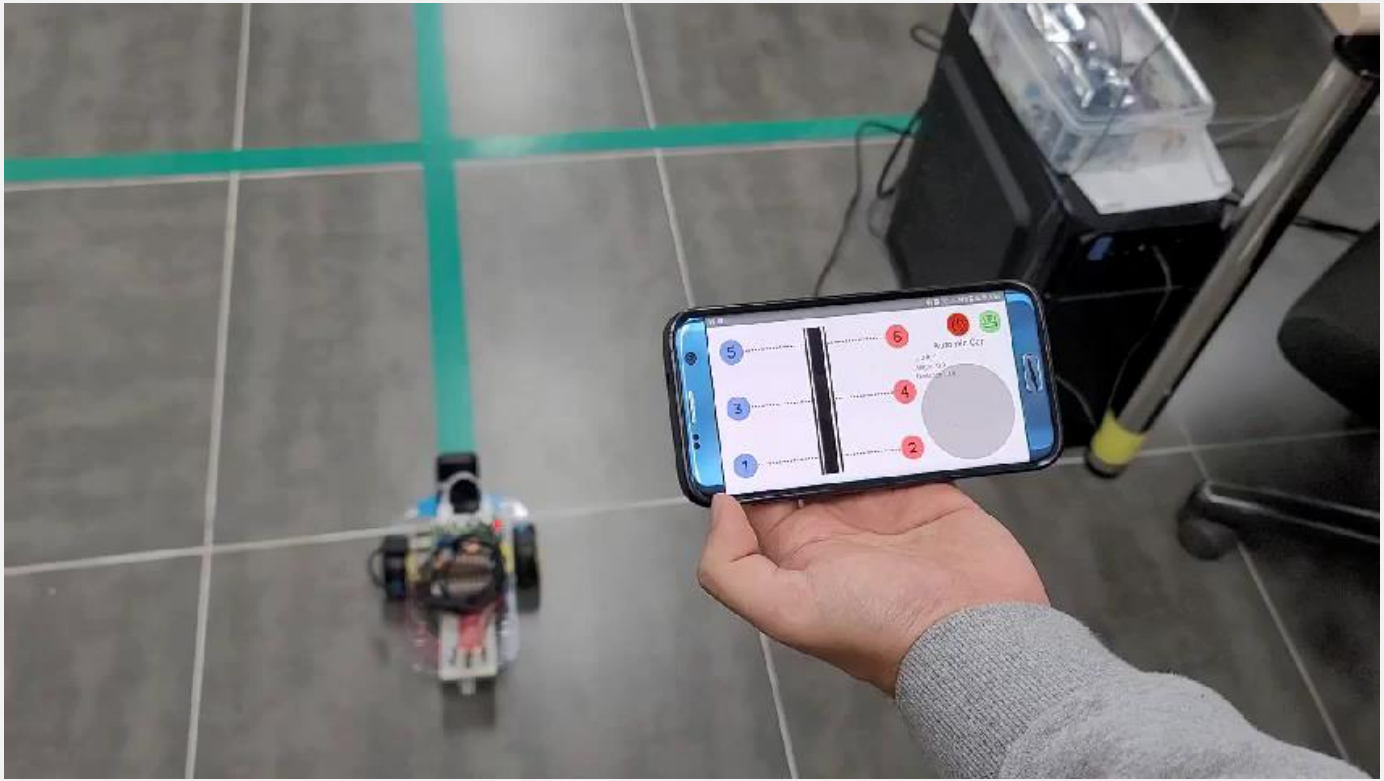
```
crossroad=0
flag=0
```

```
    else:
        if center_list[1]!=None and center_list[2]!=None :
            if center_list[1][1] > Frame_Height/2 and center_list[2][1] > Frame_Height/2 :
                if flag==0:
                    flag=1
                    crossroad+=1
                else:
                    if destination<3 and crossroad==1:
                        odd_even(int(cmd[0]))
                    elif destination<5 and crossroad==2:
                        odd_even(int(cmd[0]))
                    else:
                        if crossroad==3:
                            odd_even(int(cmd[0]))
            else:
                flag=0
        CARControl_UDP_ver2.Forward()
        print(crossroad)
        print(flag)
else:
    CARControl_UDP_ver2.Stop()
```

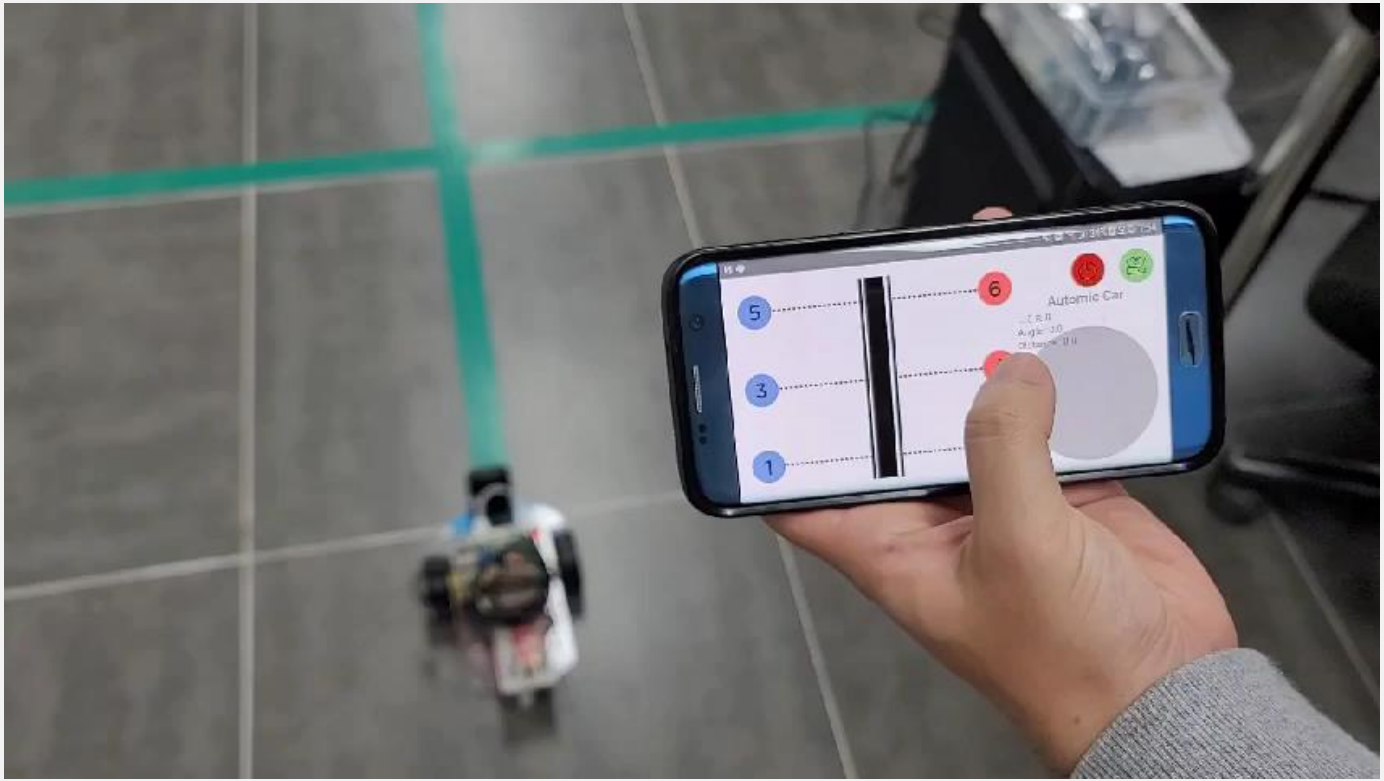
| crossroad | flag |
|---|---|
| 2 | 1 |

```
def odd_even(destination):
    if (destination%2)==1:
        CARControl_UDP_ver2.TurnLeft()
    else:
        CARControl_UDP_ver2.TurnRight()
```

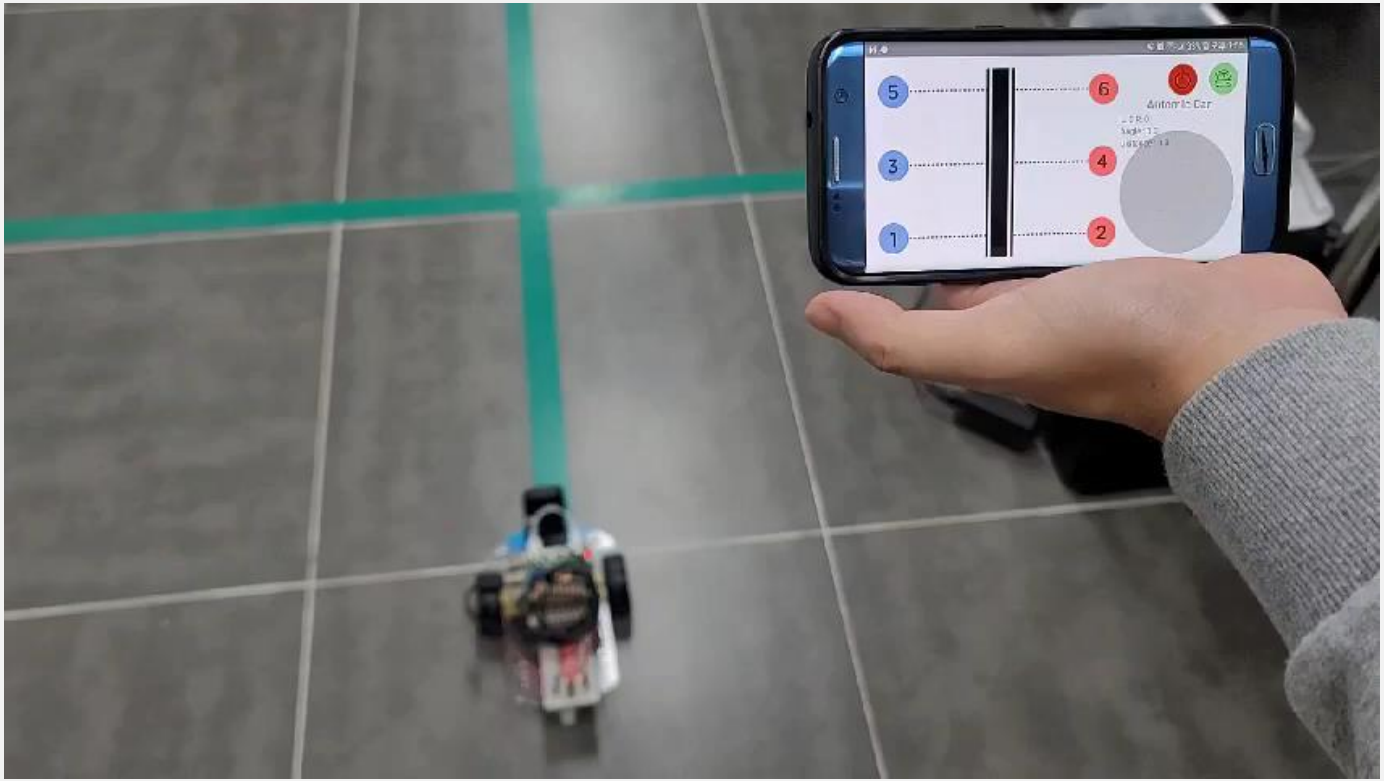
# 촬영 영상(1번)



# 촬영 영상(4번)



# 촬영 영상(5번)



# Thank you