

Source code with detailed comments(1)

```
def setMotor():  
    motor.set_default_speed(100)  
    motorS.set_default_speed(100)  
  
def run(pw):  
    motor.start(pw)  
  
def stop():  
    motor.stop()  
    motorS.stop()
```

Necessary for robot
operation.forward, stop,
etc.lograms

Source code with detailed comments(2)

```
def corner():  
    motorS.run_to_position(-15, blocking=False)  
    time.sleep(0.2)  
    motor.run_for_degrees(300,-pw)  
    motorS.run_to_position(15, blocking=False)  
    time.sleep(0.2)  
    motor.run_for_degrees(100,pw)  
    motorS.run_to_position(15, blocking=False)  
    time.sleep(0.2)  
    motor.run_for_degrees(1000,pw)
```

```
def cornerB():  
    motorS.run_to_position(15, blocking=False)  
    time.sleep(0.2)  
    motor.run_for_degrees(300,-pw)  
    motorS.run_to_position(-15, blocking=False)  
    time.sleep(0.2)  
    motor.run_for_degrees(100,pw)  
    motorS.run_to_position(-15, blocking=False)  
    time.sleep(0.2)  
    motor.run_for_degrees(1000,pw)
```

90-degree bend along an
interior wallProgram for

Source code with detailed comments(3)

```
def correct_value(n):  
    if n < 150:  
        return 150  
    elif n > 450:  
        return 450  
    else:
```

Program to limit
ultrasonic sensor
values to a
minimum of 150 and
a maximum of 450

```
setMotor()  
cornerOn=False  
x=0  
segment_p = 0  
dis = -1  
count = 0  
countB = 0  
countC = 0  
non = 0  
CC = 0  
CCC = 0
```

Set initial values of
variables necessary
for executing the
main program

Source code with detailed comments(4)

```
while True:
    if x == 0:
        stop()
        print('setup')
        GPIO.wait_for_edge(18, GPIO.FALLING)

    if rotation == "":
        runStraight()
        if dist_L < 0 or 1200 < dist_L:
            rotation = "Left"
        elif dist_R < 0 or 1200 < dist_R:
            rotation = "Right"

    if shukai == 1:
        dis_p = dist.get_distance()
    else:
        dis_p = distB.get_distance()

    motorS.run_to_position(0, blocking=False)
    time.sleep(0.5)
    run(pw)
    x = 1

    if count >= 12:
        CCC += 1
        print(CCC)
        if CCC == 100 :
            stop()
            break
```

Main program Program to start the robot with a switch
A program that counts the number of bends and stops the robot above a certain value

Source code with detailed comments(5)

```
if shukai == 1:
    dis = dist.get_distance()
else:
    dis = distB.get_distance()

countB += 1
if dis == -1 or dis > 1100 :
    non += 1

if non >= 10 and countB >= 180 or non > 15:
    if non > 10:
        if shukai == 1:
            cornerC()
        else:
            cornerD()
    else:
        if shukai == 1:
            corner()
        else:
            cornerB()
if countB >= 120 or count == 0 and countB <= 119:
    count += 1

countC = 0
countB = 0
run(pw)
```

Program measures distance to wall and moves forward while maintaining distance. When the distance becomes much larger than the measured value Or when it becomes unmeasurable, The program turns 90 degrees.

Source code with detailed comments(6)

```
count +=1
```

```
countC = 0
```

```
countB = 0
```

```
run(pw)
```

```
while not 0 < dis < 340 and not countC > 60:
```

```
    if shukai == 1:
```

```
        dis = dist.get_distance()
```

```
        print(dis)
```

```
        countC += 1
```

```
        print(countC)
```

```
    else:
```

```
        dis = distB.get_distance()
```

```
        print(dis)
```

```
        countC += 1
```

```
        print(countC)
```

```
countC = 0
```

```
countD = 0
```

```
motorS.run_to_position(0, blocking=False)
```

```
motor.run_for_degrees(100,pw)
```

If the distance between the robot and the inner wall is too far after a 90-degree turn, program the robot to turn until a certain distance is reached.

Source code with detailed comments(7)

```
if dis == -1:
    if shukai == 1:
        segment = 10
    else:
        segment = -10

elif dis <= 200:
    if shukai == 1:
        segment = -10
    else:
        segment = 10

elif dis <= 100:
    if shukai == 1:
        segment = -15
    else:
        segment = 15

elif dis > 1000:
    segment = 0

elif -10 <= dis_p - dis <= 25:
    segment = 0

elif 25 < dis_p - dis:
    if shukai == 1:
        segment = -5
    else:
        segment = 5

elif dis_p - dis < 30:
    if shukai == 1:
        segment = 10
    else:
        segment = -10
```

Program to change
steering depending
on distance from wall

Pseudo code

Obstacle Challenge

Determine the color of the obstacle
with the greatest width in the
camera's view.



When the width of the obstacle
reaches the standard value, start
evasive action.



If the obstacle is red, avoid it to the
right. If the obstacle is green, avoid it
to the left.

Frow diagrams

