Hardware Control course-Restricting

Tip:

In order to avoid sunlight affecting the infrared sensor, the experiment needs to be carried out indoors.

1. Learning target

In this course, we will learn how to use tracking module to make car completes restricting experiment.

2. Principle of experimental

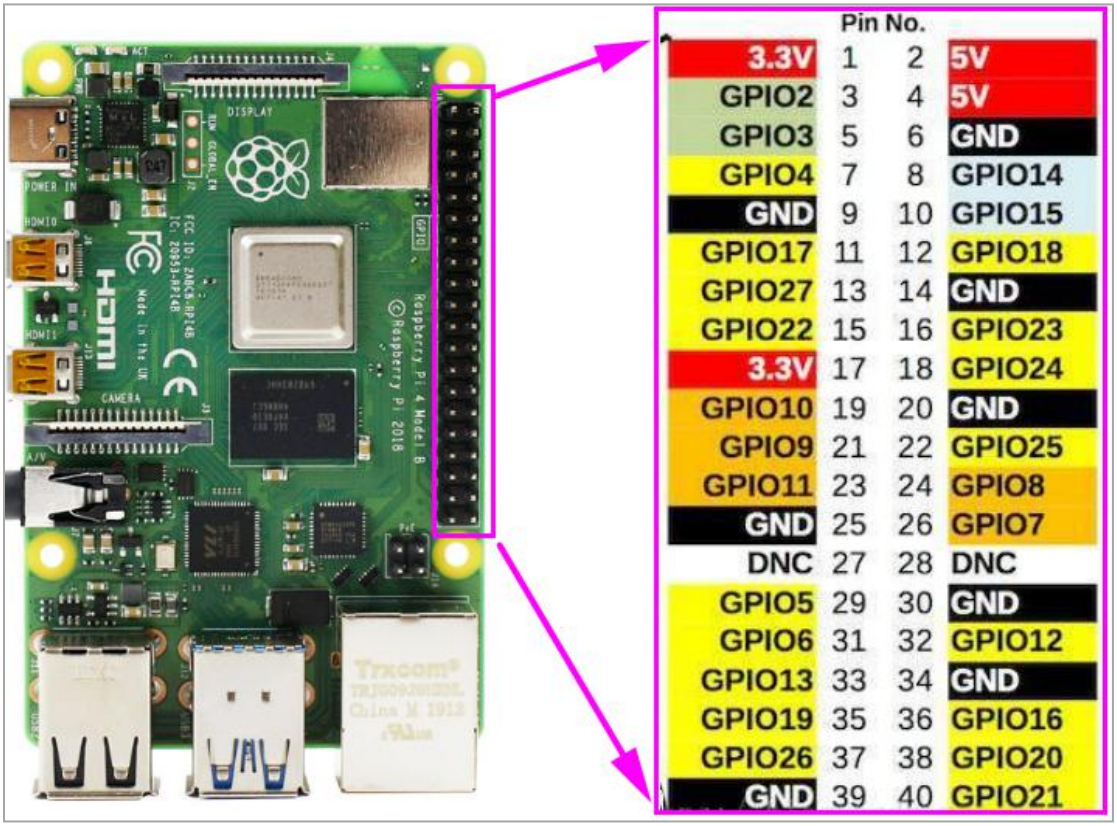
The basic principle of the tracking sensor is to use the reflective nature of the object. Our experiment is to tracking the black line. When the infrared light is emitted to the black line, it will be absorbed by the black line. When the infrared light is emitted to the other color line, it will reflected to the infrared receiver tube.

For the Raspbot car, we use 4 TT DC gear motors. They are driven by the TB6612 chip. The driver chip is not directly connected to the Raspberry Pi pins.
Raspberry Pi communicates with STM8 MUC through IIC, and then STM8 MCU drives TB6612 chip to drive the motor.

3. Coding method

In this course, we use BOARD coding method.
According to the hardware interface manual, we see that the tracking module pins are connected to the 11,7,13,15 pin of the Raspberry Pi board.
STM8 is connected to SDA.1, SCL.1 on the Raspberry Pi board.
The pin comparison table of Raspberry Pi as shown below.

We have provided a library text dedicated to driving motors and servos --YB_Pcb_Car.py.
It is located in the same directory as the motor driver.



wiringPi	BCM	Function	BOARD		Function	BCM	wiringPi
		3.3V	1	2	5V		
8	2	SDA.1	3	4	5V		
9	3	SCL.1	5	6	GND		
7	4	GPIO.7	7	8	TXD	14	15
		GND	9	10	RXD	15	16
0	17	GPIO.0	11	12	GPIO.1	18	1
2	27	GPIO.2	13	14	GND		
3	22	GPIO.3	15	16	GPIO.4	23	4
		3.3V	17	18	GPIO.5	24	5
12	10	MOSI	19	20	GND		
13	9	MISO	21	22	GPIO.6	25	6
14	11	SCLK	23	24	CE0	8	10
		GND	25	26	CE1	7	11
30	0	SDA.0	27	28	SCL.0	1	31
21	5	GPIO.21	29	30	GND		
22	6	GPIO.22	31	32	GPIO.26	12	26
23	13	GPIO.23	33	34	GND		
24	19	GPIO.24	35	36	GPIO.27	16	27
25	26	GPIO.25	37	38	GPIO.28	20	28
		GND	39	40	GPIO.29	21	29

4. About code

Path: /home/pi/Yahboom_project/Raspbot/2.Hardware Control course/7.Ultrasonic avoid/Tracking test.ipynb

1) Import time, GPIO and YB_Pcb_Car library

```
#-*- coding:UTF-8 -*-
import RPi.GPIO as GPIO
import time
import YB_Pcb_Car
```

2)Set the GPIO coding mode, define tracking module pin and define the car class is used to drive motors or servos.

```
car = YB_Pcb_Car.YB_Pcb_Car()

Tracking_Right1 = 11
Tracking_Right2 = 7
Tracking_Left1 = 13
Tracking_Left2 = 15

GPIO.setmode(GPIO.BOARD)

GPIO.setwarnings(False)

GPIO.setup(Tracking_Left1,GPIO.IN)
GPIO.setup(Tracking_Left2,GPIO.IN)
GPIO.setup(Tracking_Right1,GPIO.IN)
GPIO.setup(Tracking_Right2,GPIO.IN)
```

3) Define the restricting function

```
#画地为牢功能函数
def restricting():
    Tracking_Left1Value = GPIO.input(Tracking_Left1);
    Tracking_Left2Value = GPIO.input(Tracking_Left2);
    Tracking_Right1Value = GPIO.input(Tracking_Right1);
    Tracking_Right2Value = GPIO.input(Tracking_Right2);

        # 0000
    if Tracking_Left1Value == False and Tracking_Left2Value == False and Tracking_Right1Value == False and Tracking_Right2Value == False :
        car.Car_Spin_Right(75,75)
        time.sleep(1)
        #1X00
    elif Tracking_Left1Value == True and Tracking_Right1Value == False and Tracking_Right2Value == False :
        car.Car_Spin_Left(70,70)
        time.sleep(1)
        #00X1
    elif Tracking_Left1Value == False and Tracking_Left2Value == False and Tracking_Right2Value == True :
        car.Car_Spin_Right(70,70)
        time.sleep(1)
        #10X1
    elif Tracking_Left1Value == True and Tracking_Left2Value == False and Tracking_Right2Value == True :
        car.Car_Spin_Right(70,70)
        time.sleep(1)
        #1X01
    elif Tracking_Left1Value == True and Tracking_Right1Value == False and Tracking_Right2Value == True :
        car.Car_Spin_Right(70,70)
        time.sleep(1)
        #0110
    elif Tracking_Left1Value == False and Tracking_Left2Value == True and Tracking_Right1Value == True and Tracking_Right2Value == False :
        car.Car_Spin_Right(70,70)
        time.sleep(1)
        #0111
    elif Tracking_Left1Value == False and Tracking_Left2Value == True and Tracking_Right1Value == True and Tracking_Right2Value == True :
        car.Car_Spin_Right(50,50)
        time.sleep(1)
        #1110
    elif Tracking_Left1Value == True and Tracking_Left2Value == True and Tracking_Right1Value == True and Tracking_Right2Value == False :
        car.Car_Spin_Left(50,50)
        time.sleep(1)
    elif Tracking_Left1Value == True and Tracking_Left2Value == True and Tracking_Right1Value == True and Tracking_Right2Value == True :
        car.Car_Run(70,70)
```

4)Call the restricting function in loop to complete the obstacle avoidance experiment.
When we click the stop button, we exit the loop, stop the car, and clear the car class and GPIO.

```
try:
    while True:
        restricting()
except KeyboardInterrupt:
    pass
car.Car_Stop()
del car
print("Ending")
GPIO.cleanup()
```

6. Experimental phenomenon

Before running the tracking experiment.
We need to adjust the sensitivity of the tracking module. The indicator light is off when white is detected, and the indicator light is on when black is detected.
After the program runs, place the car within the range of the black line on the white background. When the car encounters the black line, it will turn around and always keep moving within the range of the black line.