4.Infrared tracking+ ultrasonic avoid

1. Learning objectives

Previously, we learned about line patrol through ultrasonic obstacle avoidance and four-way line patrol module. Now we combine these two functions to realize line patrol, ultrasonic detection of obstacles and buzzer sounding.

2. Experimental preparation

1. The car wiring has been installed and installed correctly
2. Debug the four-way line patrol module. When it encounters a black line, the indicator light is on. When it is not a black line, the indicator light is off.

3. Implementation principle

Using ultrasound, we get the distance between the car and the obstacle. Through the four-way line patrol module, we let the car patrol the black line. When the car patrols the line, the ultrasonic wave detects that the obstacle is smaller than the distance value we set, the car stops, and the buzzer sounds a reminder.

4. Code analysis

Source code path:

/home/pi/project_demo/05.Comprehensive_gameplay/3.ultrasonic_followup.ipynb

```
import sys
sys.path.append('/home/pi/project_demo/lib')
#Import Mecanum Car Driver Library
from McLumk_Wheel_Sports import *

speed =30#30
try:
# Turn on the ultrasonic ranging function
bot.Ctrl_Ulatist_Switch(1)
while True:
# Read line sensor data from I2C
track_data = bot.read_data_array(0x0a, 1)
track = int(track_data[0])

# Analyze the status of the line patrol sensor
x1 = (track >> 3) & 0x01
x2 = (track >> 2) & 0x01
```

```
x3 = (track >> 1) & 0x01
x4 = track & 0x01
"""
X2 X1 X3 X4
| | | |
L1 L2 R1 R2
"""
lineL1=x2
lineL2=x1
lineR1=x3
lineR2=x4

# Reading distance from ultrasonic sensor
diss_H =bot.read_data_array(0x1b,1)[0]
diss_L =bot.read_data_array(0x1a,1)[0]
dis = diss_H << 8 | diss_L
if(dis <200):
stop_robot()
bot.Ctrl_BEEP_Switch(1) #Buzzer on
time.sleep(0.1)
bot.Ctrl_BEEP_Switch(0)
else:
if lineL1 == 0 and lineL2 == 0 and lineR1 == 0 and lineR2 == 0: # All black,
speed up
print("1")
print(lineL1,lineL2,lineR1,lineR2)
move_forward(int(speed))
elif( (lineL2 == 0 or lineL1 == 0) and lineR2 == 0):#Right acute angle: right
big bend, 0 means black line is detected
print("2")
print(lineL1,lineL2,lineR1,lineR2)
rotate_right(speed)
time.sleep(0.05)
elif lineL1 == 0 and (lineR2 == 0 or lineR1 == 0): # Left sharp angle or left
sharp bend
print("3")
print(lineL1,lineL2,lineR1,lineR2)
rotate_left(int(speed*1.5)) # Sharp left turn
time.sleep(0.15)
elif lineL1 == 0: # Left outermost detection
print("4")
print(lineL1,lineL2,lineR1,lineR2)
rotate_left(speed) # Sharp left turn
time.sleep(0.02)
elif lineR2 == 0: # Right outermost detection
print("5")
print(lineL1,lineL2,lineR1,lineR2)
rotate_right(speed)
time.sleep(0.01)
elif lineL2 == 0 and lineR1 == 1: # The sensor on the middle black line fine-
tunes the car to turn left
print("6")
print(lineL1,lineL2,lineR1,lineR2)
rotate_left(int(speed)) # Turn left
```

```
elif lineL2 == 1 and lineR1 == 0: # The sensor on the middle black line fine-
tunes the car to turn right
print("7")
print(lineL1,lineL2,lineR1,lineR2)
rotate_right(int(speed)) # Turn right
elif lineL2 == 0 and lineR1 == 0: # All black, speed up
print("8")
print(lineL1,lineL2,lineR1,lineR2)
move_forward(speed)

# Wait for a while before the next test
time.sleep(0.01)

except KeyboardInterrupt:
# When the user presses Ctrl+C When the user presses Ctrl+C, the ultrasonic
ranging function is turned off
bot.Ctrl_Ulatist_Switch(0)
# Ensure that all motors stop when the user interrupts the program
stop()
print("Ending")
```

5. Experimental results

We put the car on the map with black lines, make sure the car's four-way patrol module has been adjusted to the point where the indicator light turns on when it encounters a black line, and turns off when it is not a black line, and then run the program. After running the program, we can see that the car patrols the black line slowly. During the patrol process, when it encounters an obstacle in front, the car stops and the buzzer sounds. When we remove the obstacle, the car continues to patrol the line and the buzzer stops.