主要工作：通过写小车循迹来熟悉小车的操控

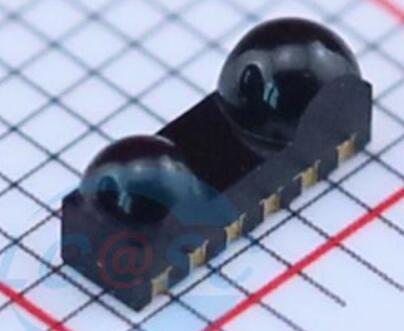
1.红外选择

图1：红外收发器

两边各有6个金属片，需要自己焊接电路。

图2：红外对管

分为红外探头和中控板，红外探头具有VCC，GND和OUT三个引脚，中控板有VCC，GNC和4个输出引脚。当红外接收管检测到有反射信号时对应的输出引脚输出低电平，无反射信号时输出高电平。

2.树莓派读取引脚

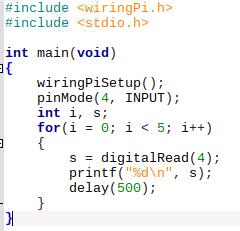
运用wiringPi库读取引脚：

图3：wiringPi库读取引脚

wiringPi库对应的引脚编号：

图4：树莓派不同编码格式下的引脚编号

3.ROS环境下编写程序

1. #include <ros/ros.h>
2. #include <signal.h>
3. #include <geometry\_msgs/Twist.h>
4. #include <wiringPi.h>
6. #define right\_senser 5
7. #define left\_senser 4
9. ros::Publisher cmdVelPub;
11. **void** turn\_left(geometry\_msgs::Twist speed);
13. **void** turn\_right(geometry\_msgs::Twist speed);
15. **void** straight\_go(geometry\_msgs::Twist speed);
17. **int** main(**int** argc, **char**\*\* argv)
18. {
19. wiringPiSetup();
20. ros::init(argc, argv, "car\_track");
21. ros::NodeHandle node;
22. cmdVelPub = node.advertise<geometry\_msgs::Twist>("cmd\_vel", 1);
23. geometry\_msgs::Twist speed;
24. pinMode(right\_senser, INPUT);
25. pinMode(left\_senser, INPUT);
26. **int** right\_read, left\_read;
27. **while**(ros::ok())
28. {
29. right\_read = digitalRead(right\_senser);
30. left\_read = digitalRead(left\_senser);
31. **if**(right\_read == 1)//黑线在右边，右转
32. {
33. turn\_right(speed);
34. }
35. **else** **if**(left\_read ==1)//黑线在左边，左转
36. {
37. turn\_left(speed);
38. }
39. **else**
40. {
41. straight\_go(speed);
42. }
43. delay(50);
44. }
45. **return** 0;
46. }
48. **void** turn\_left(geometry\_msgs::Twist speed)
49. {
50. speed.linear.x = -0.01;
51. speed.angular.z = -0.8;
52. cmdVelPub.publish(speed);
53. }
55. **void** turn\_right(geometry\_msgs::Twist speed)
56. {
57. speed.linear.x = -0.01;
58. speed.angular.z = 0.8;
59. cmdVelPub.publish(speed);
60. }
62. **void** straight\_go(geometry\_msgs::Twist speed)
63. {
64. speed.linear.x = -0.08;
65. speed.angular.z = 0;
66. cmdVelPub.publish(speed);
67. }

4.wiringPi库的导入

首先在代码中include需要使用的库：

图5：程序中直接include相关库

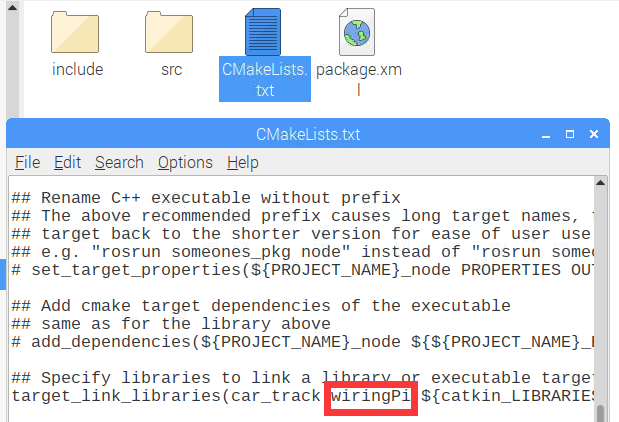
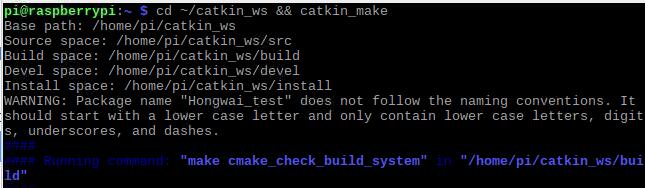
对于ROS环境之外的库，需要在CMakeLists文件中进行链接：

图6：CMakeLists中链接wiringPi库

5.编译

使用ROS环境中的编译工具catkin\_make在src文件目录下进行编译：

图7：catkin\_make进行编译

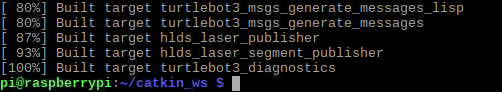
如果编译成功则会显示以下图片：

图8：编译成功

6.运行程序检测是否正常工作

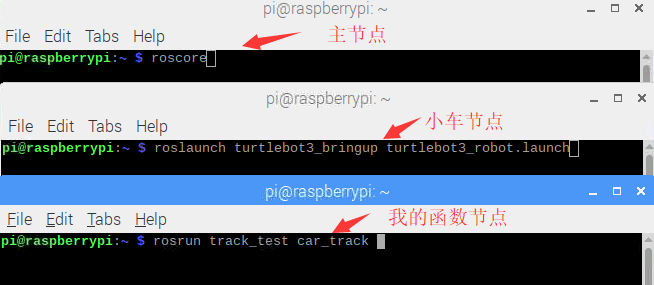
按顺序运行主节点、小车结点和我们的代码节点：

图9：代码运行

7.演示视频