主要工作：运用LDS激光雷达模块实现探测以及相应的行驶功能（在视场中找到目标物然后走至其附近）

1.数据获取

查看节点的相关话题，如图1所示：

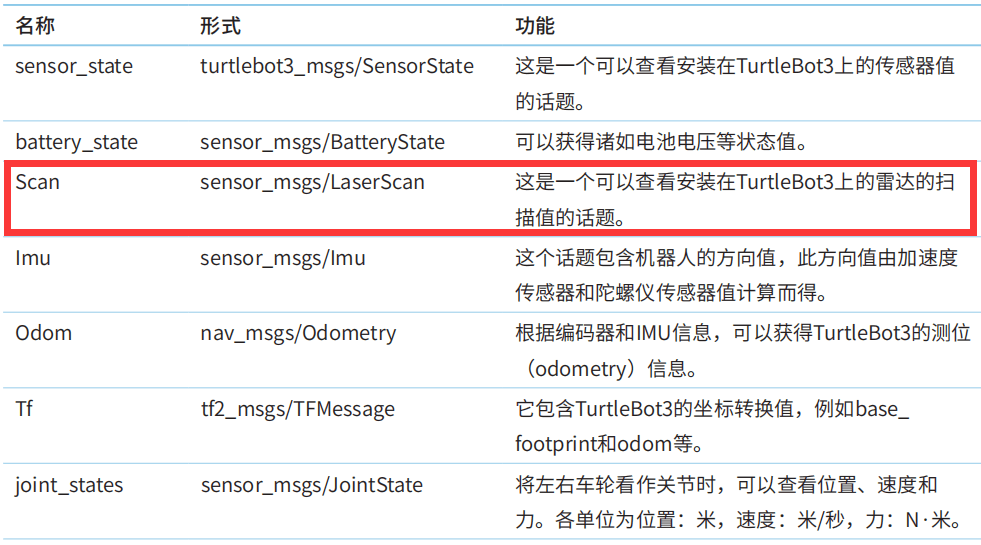


图1：有关雷达的相关话题

查看该话题发布的消息：



图2：查看Scan话题发布的消息

查看该消息发布的内容：

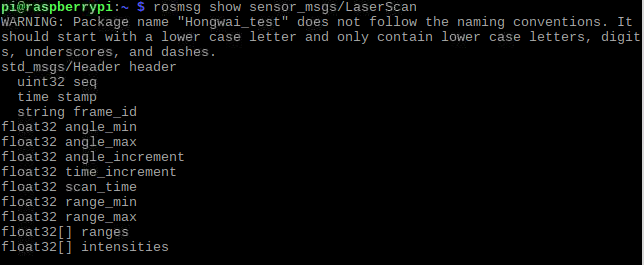


图3：查看LaserScan话题发布的内容

通过官方文档查询内容中具体数值是什么：

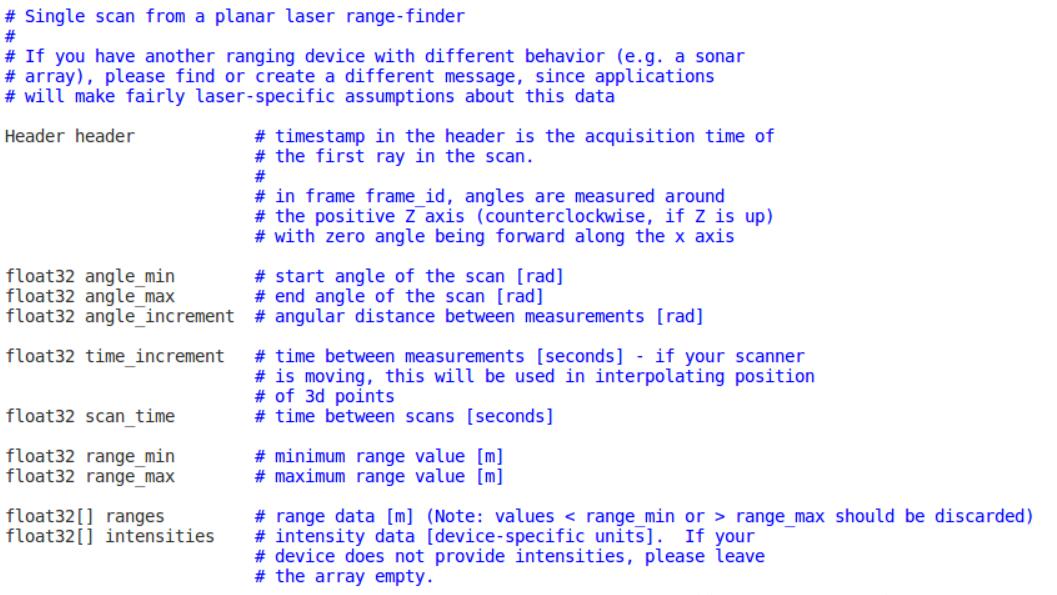


图4：LaserScan话题中对应的具体数值

从图中可以看出，在改组数据中，ranges为每个角度上对应的距离，intensities为每个角度上对应的强度（色彩强度）。

2.编写相关代码

获取雷达数据的代码：

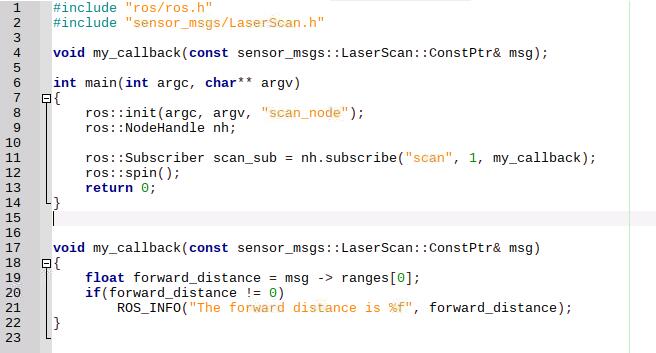


图5：编写代码获取雷达的数据

用XML的格式编写.launch文件，然后运行roslaunch，这样做的好处是将所有按照顺序运行的节点写到.launch文件中，避免了一个一个运行：

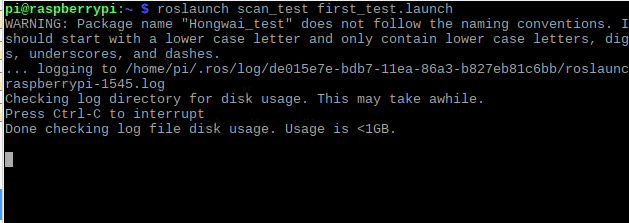


图6：运行.launch文件

结果如下：

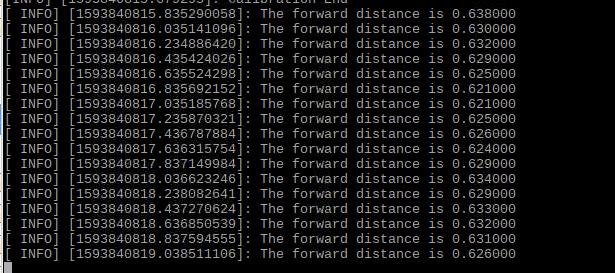


图7：代码运行结果

编写运用雷达获取的信号控制小车行走的程序：

1. #include <ros/ros.h>
2. #include <sensor\_msgs/LaserScan.h>
3. #include <geometry\_msgs/Twist.h>
4. #include <wiringPi.h>
5. #include <math.h>
7. ros::Publisher cmdVelPub;
8. geometry\_msgs::Twist speed;
10. **void** run\_forward();
11. **void** run\_stop();
12. **void** turn\_angle(**float** angle);
13. **void** my\_callback(**const** sensor\_msgs::LaserScan::ConstPtr& msg);
15. **int** main(**int** argc, **char**\*\* argv)
16. {
17. wiringPiSetup();
18. ros::init(argc, argv, "find\_bench");
19. ros::NodeHandle nh;
20. cmdVelPub = nh.advertise<geometry\_msgs::Twist>("cmd\_vel", 1);
22. ros::Subscriber scan\_sub = nh.subscribe("scan", 1, my\_callback);
23. ros::spin();
24. **return** 0;
25. }
27. **void** run\_forward()
28. {
29. speed.linear.x = 0.01;
30. speed.angular.z = 0;
31. cmdVelPub.publish(speed);
32. }
34. **void** run\_stop()
35. {
36. speed.linear.x = 0;
37. speed.angular.z = 0;
38. cmdVelPub.publish(speed);
39. }
41. **void** turn\_angle(**float** angle)
42. {
43. **float** time = angle / 18.0;
44. speed.linear.x = 0;
45. speed.angular.z = 0.3;
46. cmdVelPub.publish(speed);
47. ros::Duration(time).sleep();
48. speed.angular.z = 0;
49. cmdVelPub.publish(speed);
50. ros::Duration(2).sleep();
51. }
53. **void** my\_callback(**const** sensor\_msgs::LaserScan::ConstPtr& msg)
54. {
55. **int**\* classify = **new** **int**[360];//存放聚类的类别
56. **int** num = 0;//存放类的总数
57. **float**\* class\_len = **new** **float**[360];//存放每一类的长度
58. **float** forward\_distance = msg -> ranges[0];//正前方的距离
59. **float** temp0;
60. **float** temp1 = msg -> ranges[0];
61. **float** temp\_len;
62. classify[0] = 1;
63. **int** i;
64. **for**(i = 1; i < 360; i++)
65. {
66. temp0 = temp1;
67. temp1 = msg -> ranges[i];
68. temp\_len = abs(temp1 - temp0);
69. **if**(temp\_len > 0.1)
70. num++;
71. classify[i] = num + 1;
72. }
73. ROS\_INFO("The num is %d", num);

76. /\*
77. float forward\_distance = msg -> ranges[0];
78. int i = 0;
79. if(forward\_distance != 0 && forward\_distance < 0.3)
80. {
81. if(ros::ok())
82. run\_forward();
83. }
84. else
85. {
86. if(ros::ok())
87. run\_stop();
88. }
89. \*/
90. }