

实验报告

姓名：廖嘉辉

时间： 7.10-7.23

【实验目的】

ROS 学习

【实验过程】

1. 创建 subscriber 节点

新建一个软件包与节点，再编写 subscriber 订阅者节点

```

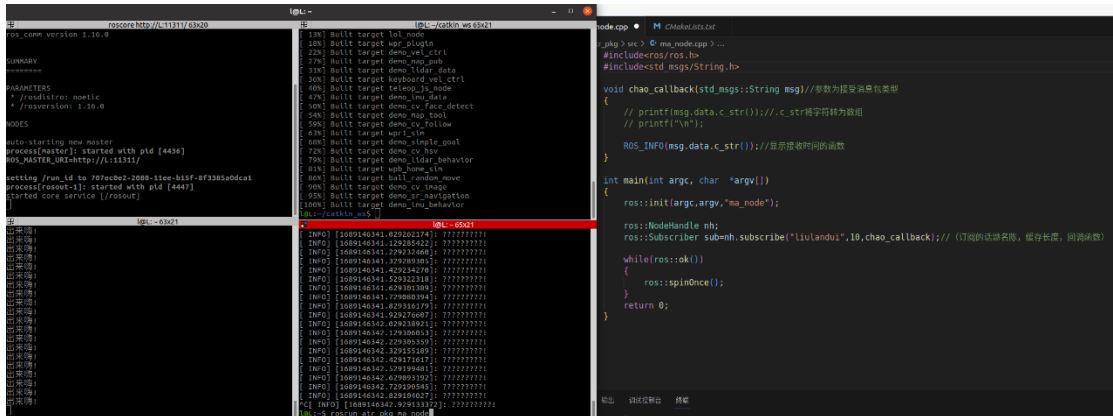
3 > atr_pkg > src > ma_node.cpp > main(int, char * [])
1 #include<ros/ros.h>
2 #include<std_msgs/String.h>
3
4 void chao_callback(std_msgs::String msg)//参数为接受消息包类型
5 {
6     printf(msg.data.c_str()); //.c_str将字符转为数组
7     printf("\n");
8 }
9
10 int main(int argc, char *argv[])
11 {
12     ros::init(argc,argv,"ma_node");
13
14     ros::NodeHandle nh;
15     ros::Subscriber sub=nh.subscribe("liulandui",10,chao_callback);// (订阅的话题名陈, 缓存长度, 回调函数)
16
17     while(ros::ok())
18     {
19         ros::spinOnce();
20     }
21     return 0;
22 }
23

```

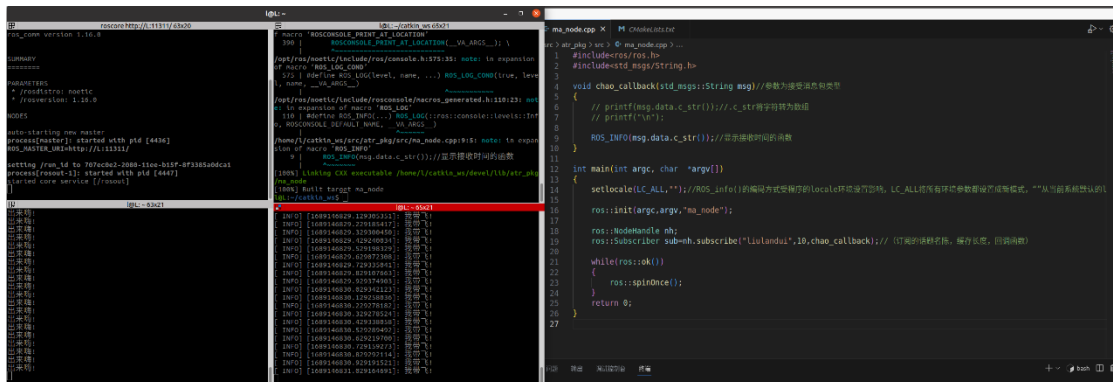
运行 publisher 节点后, 再运行 subscriber 节点

[illegible]

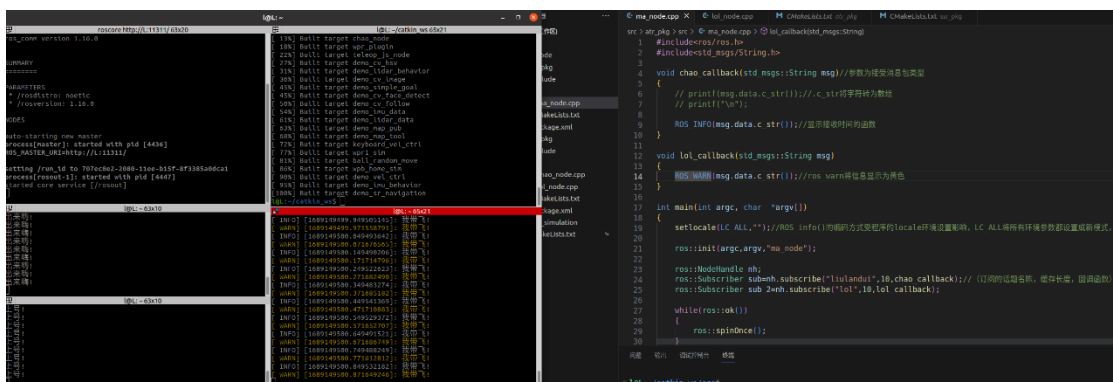
再加入显示接受时间的函数



显示内容为“?????”，因为 ROS_INFO() 函数收程序的 locale 环境设置影响，需要在主函数中加入“setlocale(LC_ALL, “”)”将所有环境参数设置为新模式



2. 使用 launch 文件启动多个节点
在 subscriber 节点同时添加需要订阅的话题，



再新建一个 launch 文件，包含所有节点

```
game.launch X
c > atr_pkg > launch > game.launch
1 <launch>
2
3 <node pkg="ssr_pkg" type="chao_node" name="chao_node"/>
4 <node pkg="ssr_pkg" type="lol_node" name="lol_node"/>
5 <node pkg="atr_pkg" type="ma_node" name="ma_node" output="screen"/>
6
7 </launch>
```

使用“roslaunch atr_node game.launch”一次启动多个节点

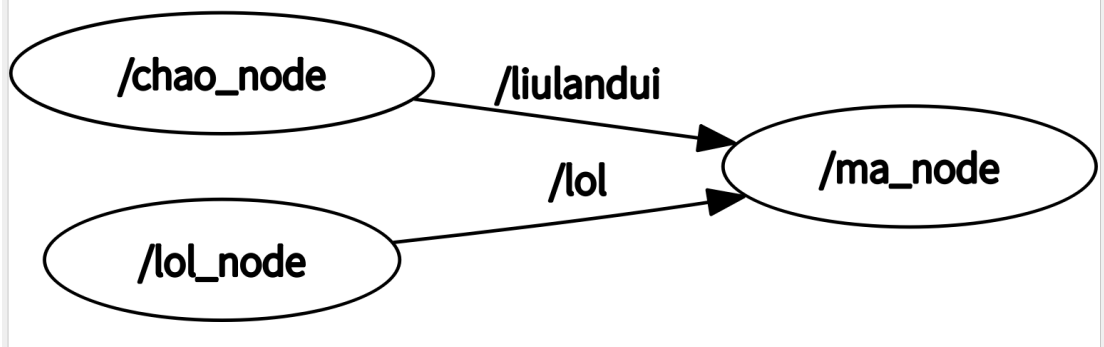
```
/home/j/catkin_ws/src/atr_pkg/launch/game.launch http://localhost:11311 -
/home/j/catkin_ws/src/atr_pkg/launch/game.launch http://localhost:11311 80x24
WARN [1689154596.279399729]: 我带飞!
INFO [1689154596.377141387]: 我带飞!
WARN [1689154596.379466523]: 我带飞!
INFO [1689154596.477065162]: 我带飞!
WARN [1689154596.479522724]: 我带飞!
INFO [1689154596.577128996]: 我带飞!
WARN [1689154596.579527064]: 我带飞!
INFO [1689154596.677072703]: 我带飞!
WARN [1689154596.679639436]: 我带飞!
INFO [1689154596.777186493]: 我带飞!
WARN [1689154596.779497316]: 我带飞!
INFO [1689154596.877134572]: 我带飞!
WARN [1689154596.879457992]: 我带飞!
INFO [1689154596.977167526]: 我带飞!
WARN [1689154596.979362028]: 我带飞!
INFO [1689154597.077126810]: 我带飞!
WARN [1689154597.079587942]: 我带飞!
INFO [1689154597.177065181]: 我带飞!
WARN [1689154597.179601766]: 我带飞!
INFO [1689154597.277180013]: 我带飞!
WARN [1689154597.279304724]: 我带飞!
INFO [1689154597.377193243]: 我带飞!
WARN [1689154597.379373953]: 我带飞!

game.launch X
c > atr_pkg > launch > game.launch
1 <launch>
2
3 <node pkg="ssr_pkg" type="chao_node" name="chao_node"/>
4 <node pkg="ssr_pkg" type="lol_node" name="lol_node"/>
5 <node pkg="atr_pkg" type="ma_node" name="ma_node" output="screen"/>
6
7 </launch>
```

使用 rqt_graph 可查看当前活跃的各个节点的关系

```
/home/j/catkin_ws/src/atr_pkg/launch/game.launch http://localhost:11311 -
/home/j/catkin_ws/src/atr_pkg/launch/game.launch http://localhost:11311 80x11
INFO [1689154679.488480391]: 我带飞!
WARN [1689154679.490977298]: 我带飞!
INFO [1689154679.588477381]: 我带飞!
WARN [1689154679.590305022]: 我带飞!
INFO [1689154679.688541272]: 我带飞!
WARN [1689154679.691111589]: 我带飞!
INFO [1689154679.788525669]: 我带飞!
WARN [1689154679.791043448]: 我带飞!
INFO [1689154679.888456887]: 我带飞!
WARN [1689154679.891106706]: 我带飞!

[ati@ati ~]$ rqt_graph
rqt_graph: 未找到命令
[ati@ati ~]$ rqt_graph
[ati@ati ~]$ rqt_graph
```



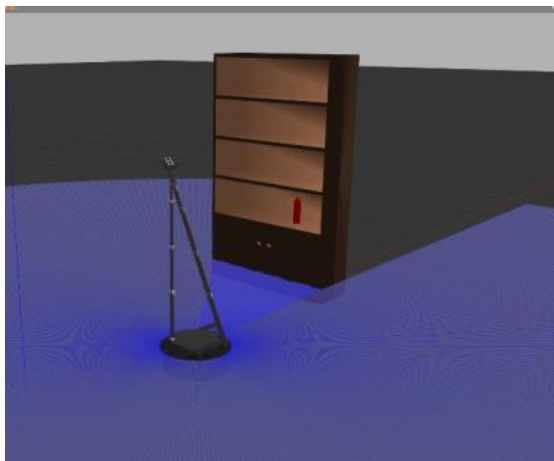
3. 实现机器人运动控制

```

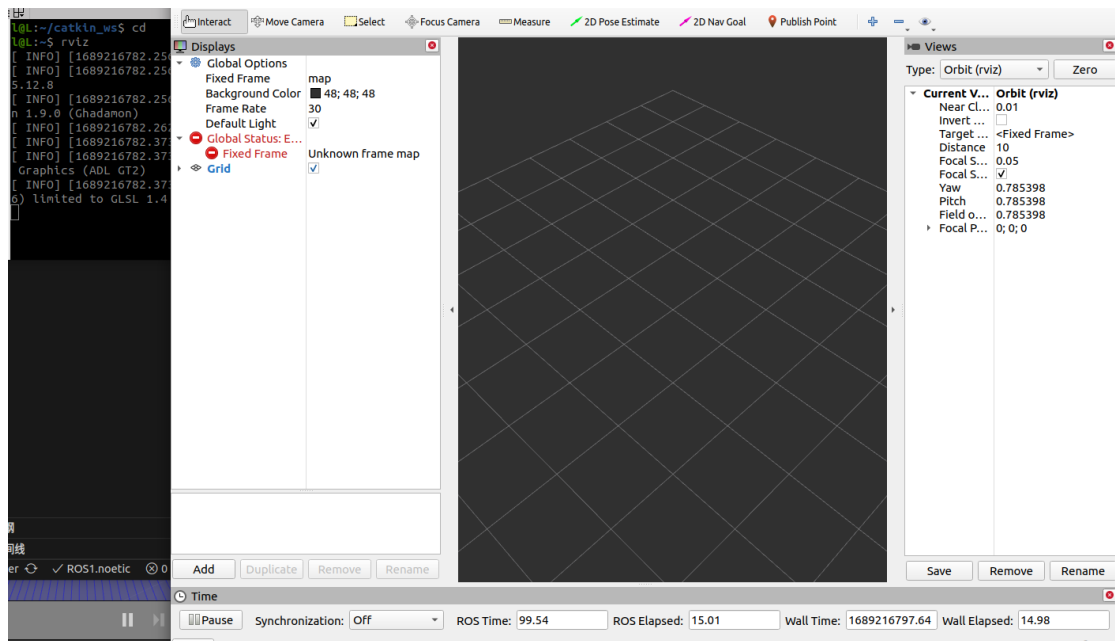
src / vel_pkg / src / vel_node.cpp / main(int, char *[])
1  #include<ros/ros.h>
2  #include<geometry_msgs/Twist.h>
3
4
5  int main(int argc, char *argv[])
6  {
7      ros::init(argc,argv,"vel");
8
9
10     ros::NodeHandle n;
11     ros::Publisher vel_pub = n.advertise<geometry_msgs::Twist>("/cmd_vel", 10);
12
13
14     geometry_msgs::Twist vel_msg;
15     vel_msg.linear.x=0.1;
16     vel_msg.linear.y=0.1;
17     vel_msg.linear.z=0.1;
18     vel_msg.angular.x=0;
19     vel_msg.angular.y=0;
20     vel_msg.angular.z=0;
21
22
23     ros::Rate r(30);
24
25     while(ros::ok())
26     {
27         vel_pub.publish(vel_msg);
28         r.sleep();

```

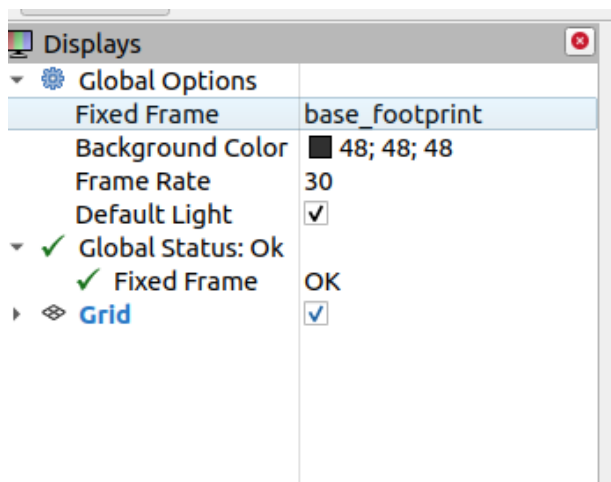
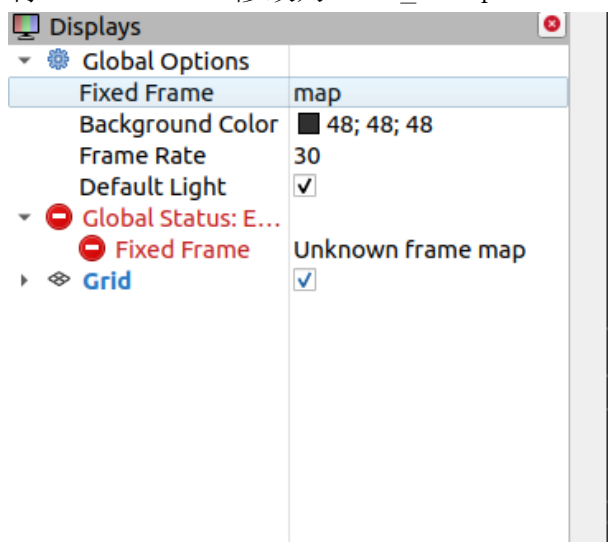
再启用仿真环境运行节点即可控制机器人运动



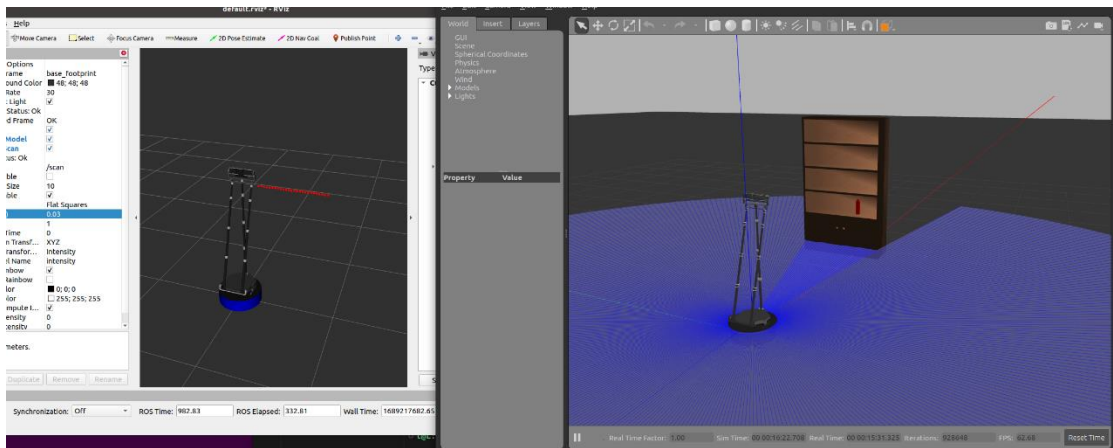
4. 使用 RViz 观测传感器数据
启动仿真环境后，再输入 RViz



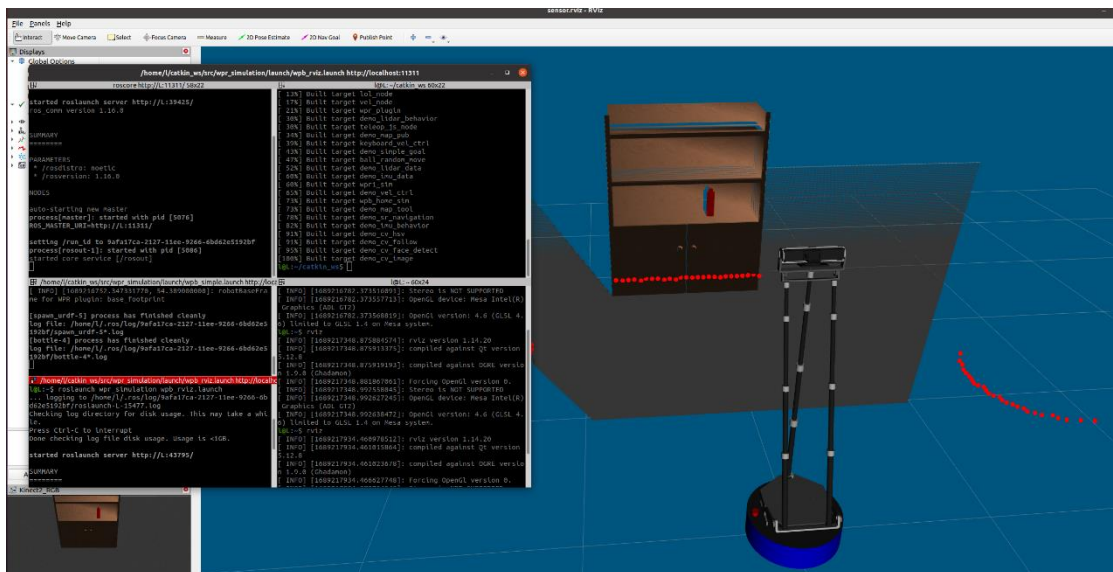
将 Fixed Frame 修改为 base_footprint



再 add 一栏选中 RobotModel 添加机器人模型，再选中 LaserScan 添加雷达，并在激光雷达话题名称选择 /scan，并修改 Size 数值



将 RViz 另存为文件，保存相关配置



【实验结果】

1. 学会 subscriber 订阅者节点编写
2. 会用 launch 文件一次启动多个节点
3. 初步了解机器人控制代码
4. 会使用 RViz 观测传感器数据