交叉项目综合训练 A 第二次实验

自动化系机器人实验室

due on 2020-03-09

1 实验目的

1. 掌握从零新建 ROS Package 的能力
2. 熟悉面向对象的 ROS 程序的编写
3. 熟练使用基本的 ROS 调试工具

2 实验环境

虚拟机/双系统 + Ubuntu 16.04/18.04 + ROS + gazebo

3 新建一个 package 并编写小车控制节点

3.1 命令行新建一个 package

根据 PPT2-6 的提示，在工作空间的 src 目录下利用 catkin_create_pkg 新建一个叫 husky_rcl_controller 的包，依赖项为 std_msgs, roscpp, nav_msgs, geometry_msgs 等。或者根据 [1] 使用 catkin 工具的 catkin creat pkg 命令新建 package。

3.2 面向对象的 ROS 编程

参考 [2] 以及 PPT2-18，使用面向对象的方法创建**算法类**和 **ROS 操作类**，其中算法类的对象实例作为 ROS 类的成员之一，使 ROS 交换得到的数据能通过算法实例进行计算。main 函数定义在单独的 cpp 中，使用定义好的类创建实例对象。

1. 使用实验一中的命令行调试工具，得到启动 husky_empty_world.launch 时 /odometry/filtered 消息的接收格式。（ROS 类中声明 subscribe 实例）

2. 类似实验一中下载的 teleop_twist_keyboard 所发布的消息，在算法类中编写小车的控制方法，并在 ROS 类中将控制的输出发送到话题：/cmd_vel。（ROS 类中声明 publisher 实例）

3. 参考 PPT2-15，在 ROS 类中编写接收 service 的方法，在收到小车的目标位置的 Request 后，调用控制器发布新的速度，来控制小车的里程计到目标位置 (x,y)，成功或者超时后返回 Service 的结果到 Response 中。自定义服务消息（.srv）的格式可以是

   float64 targetX

   float64 targetY

   float64 targetOrientationZ.

   —

   bool result

   float64 timeCost

4. 使用命令行语句，手动输入参数，调用编写好的 Service 接口，完成让小车移动到指定位置的任务。

   尤其要注意一些边界情况，例如在两个 terminal 中分别给不同的位置指令，需要进行异常的处理（选择抛弃新的指令等待上一个指令完成，或者抛弃旧指令开始执行新的指令）；再比如，需要对完成、无法完成、完成超时等进行区分。

   挑战：对路径规划有兴趣的可以尝试将指令换成 x,y 和目标位置的朝向角，而小车的速度指令仍然是 x 速度和 z 转速。

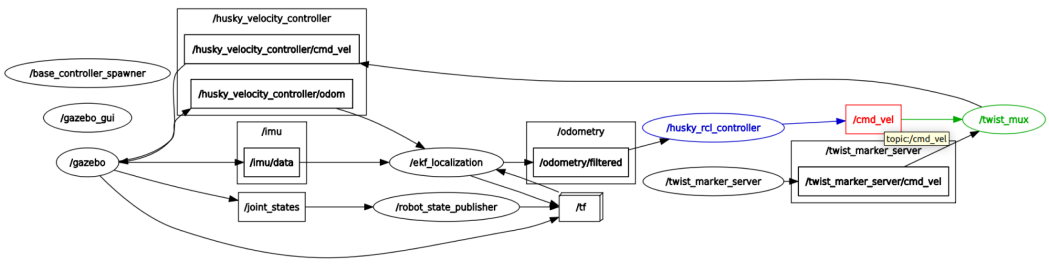

图 1: rqt_graph 观察自定义控制器的作用

提示：subscribe 和 service 的 callback 都有类似的回调和队列机制。在不使用多线程的情况下，由 RosHandler 负责分配当前的执行权限，如果多个回调函数被依次触发，上一个未执行完的回调函数会暂时保存在函数栈中，等待最新的回调函数执行完毕，并且回调函数内定义的局部变量会在每次回调时在新开辟的栈内初始化，但回调函数能够存取全局或者类内等非栈空间的变量。

关于 ros::spin() 和 ros::spinOnce() 这两个唤醒回调的函数区别，可以参考 [3]，本次作业在合理分配回调时机的情况下不需要用多线程完成。

鼓励使用面向对象的编程习惯，以下主节点程序参考 [2]：

```
#include <ros/ros.h>
#include "husky_rcl_controller/HuskyRosApplication.hpp"

int main(int argc, char** argv)
{
    ros::init(argc, argv, "husky_rcl_controller");
    ros::NodeHandle nodeHandle("~");

    husky_rcl_controller::HuskyRosApplication realRosApplication(nodeHandle);

    //Wait for topic/service callback.
    ros::spin();

    return 0;
}
```

在使用命令行工具创建新的 package 时，会自动生成 package.xml 和 CMakeLists.txt，然后我们需要根据需要修改这两个文件 (比如添加新的依赖)，可以参考 [2, 4]。

4 提交要求

1. 源代码包：包含控制器节点相关代码、自定义消息文件夹等，能独立重新构建。
2. README.txt：简要解释各个类以及各自的方法的作用。

3. pdf 报告：给出程序的算法流程图；给出程序的使用过程截图，以及实际控制效果（类似图4的数据曲线）。

4. 对 ROS 消息和服务机制的个人理解。

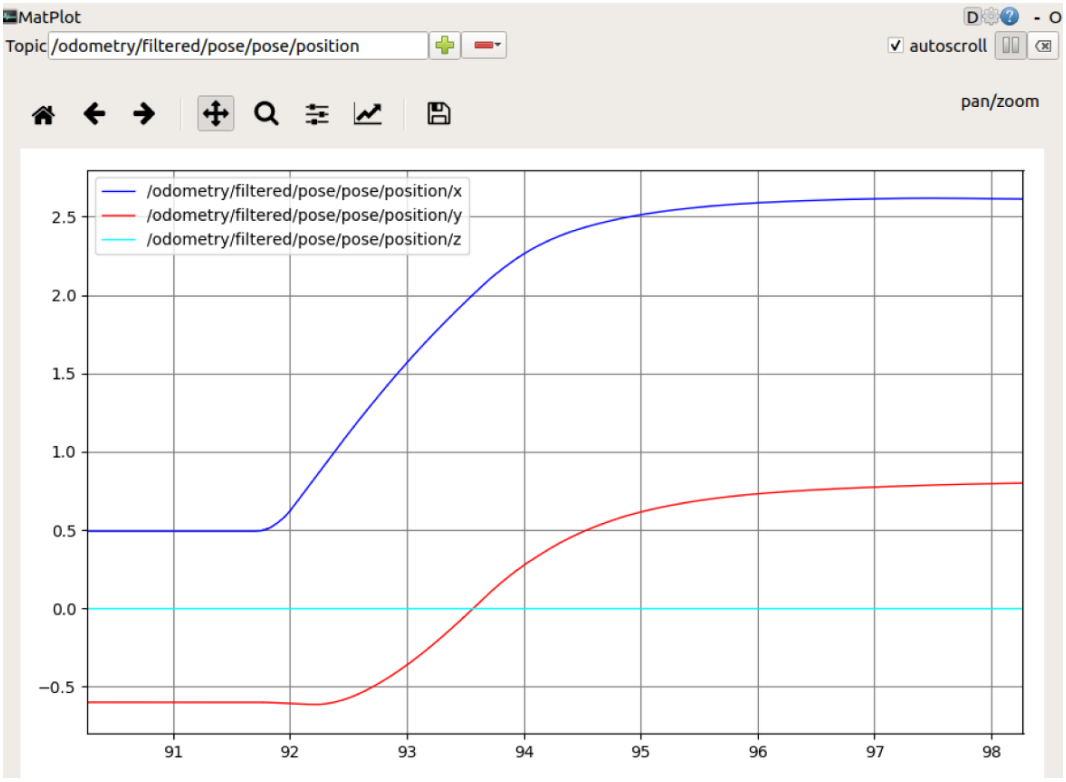

图 2: 控制 husky 到指定的位置 (x,y)

参考文献

[1] https://catkin-tools.readthedocs.io/en/latest/verbs/catkin_create.html

[2] https://github.com/leggedrobotics/ros_best_practices

[3] https://blog.csdn.net/hzy925/article/details/79373403

[4] http://wiki.ros.org/ROS/Tutorials/CreatingPackage