轮式机器人技术综合实验1

石昊海 3180102707

# 实验内容与目的

实验内容：在仿真环境中完成各个节点的编写，让自己设计的小车在给定的低障碍环境中可以自行完成遥控运动、自动路径规划、简单跟踪路径点的功能。

实验目的：最终实现小车可以在给定目标后自主跟踪目标，并且对于算法和程序进行优化，使选择的路径的长度在保证可通过性的前提下越短越好、搜索路径的用时越短越好、路径跟踪越准确越好。

# 实验方法与步骤

1. 熟悉ubuntu环境和ros工具的使用方法。
   1. 熟练掌握命令行操作，和git、vim等工具
   2. 创建工作目录catkin\_ws，创建rospack
   3. 理解掌握ros中Node、message、topic、service等概念
   4. 会使用roslaunch、rqt\_console、 rosrun、rostopic等调试工具
   5. 分别写一个简单的publisher、subscriber、service、client的python程序
2. 使用urdf在gazebo中构建双轮差动机器人，使其能正常在gazebo中显示；编写ros节点使得可以通过键盘控制小车运动。
   1. 双轮差动机器人由助教配置好，在rviz里验证它joint位置正常，各个坐标系方向正确
   2. 在gazebo仿真环境中检查模型，通过rostopic验证它的运动能力
   3. 编写键盘控制节点keyboard\_velocity.py，负责将键盘命令转换为机器人本体坐标的速度和角速度命令
   4. 编写速度运动学分解节点kinematics.py，接收上一个节点的速度和角速度message，翻译为直接调整左右轮转速的控制信息并下发给轮子。
   5. 在gazebo中查看小车的运动状态，检验节点是否工作正常并改进。
3. 在gazebo中载入地图，在小车上添加sensor；添加tf使得rviz可以正常地确定小车与世界坐标系之间的转换关系。
   1. 在urdf里添加hokuyo激光传感器、imu加速度传感器。（已经由助教配置好）
   2. 写坐标系转换广播节点robot\_tf.py。订阅gazebo/model\_states的topic，获取agv的pose和position。根据pose和position信息定义map坐标系和robot\_base坐标系间简单的坐标变换关系，盖上时间戳rospy.Time.now()。
   3. 先打开gazebo（course\_agv\_world.launch），再运行rviz（course\_agv\_world\_rviz.launch）。上一步已经使得地图坐标系、机器人坐标系、传感器坐标系等形成了一棵统一而完整的坐标树。
   4. 在rviz中添加从/courses/imu 和/courses/laser两个topic订阅的加速度和激光传感数据。如果tf写对了，由红点表示的激光传感器障碍显示应该与地图的边界符合。以此为标准检测tf转换节点编写的正确性。
4. 学习掌握路径规划算法并实现；利用原始的路径跟踪程序使得小车可以自行追踪给定的目标点。
   1. 通过订阅/map 的topic获得全局的地图信息。将地图整理成栅格化的网格状，之后对障碍物边缘以适当半径进行膨胀，确保机器人运动过程中不会与障碍物发生碰撞。
   2. 通过tf获得当前的全局坐标。
   3. 编写路径规划global\_planning节点，通过订阅topic /course\_agv/goal获得rviz中通过交互式按钮设置的goal。
   4. 节点调用A\*算法函数（以当前坐标、目标点位置为参数）在网格图中搜索最佳的路径点序列，并发布给/course\_agv/global\_path topic，使得路径能在rviz中显示
   5. 运行stupid\_tracking原始路径跟踪节点。原始路径跟踪节点将自身与细分路径点之间的位置关系简单地映射到控制命令上，实现逐一到达路径点，最终到达目标的效果。

# 实验结果

实验中出现的问题：

Rospy.Time.now()时间不对应

Path里的posestamped的seq从0,1，不是升序会怎样

Pathplanner的map I,j是与x,y相反的，比较坑爹

# 讨论、心得

## 学到了什么？Question the answer

## 能不能做得更好？Answer the question

有很多地方和助教设置的解决方案不尽相同。

Map和global\_planning不是通过service来做而是通过调用内部的函数。可复用性有一定缺陷。

各个节点会发生各自有什么问题？

架构设计有没有缺陷？PathTracking能否要求重新规划路线？

小车的设计需要改进。如果将两轮放在质心中间，可能效果会好些。轮子转动惯量太大了，如果轮子质量比较小，角动量是不是就会比较好。减小轮子半径、调低底盘高度

Keyboard\_control是一个什么样的节点？如果要求指令更加平滑，应该在中间插入新节点还是应该修改keyboard\_control？抑或是在kinematics节点中调整？改进急刹的平稳性。PID控制？设置速度上限。

导航在这里由于mapserver的返回数据格式已经自动分好网格，直接用rrp就可以了，很省事。网格大小的选取影响大不大？路径规划的速度。。。膨胀半径的选取，多少合适？

障碍物更密一点，会不会有事？在路径更狭窄的时候？能用循环解决的事不要用递归。

Stupid\_tracking能不能做得更好，可以

右转没问题，左传跳动很厉害，还是

Subscriber/Publisher模式和service/client 模式有什么区别？适用场景上有无什么区分？能否相互替代？