zxcar: 机器人入口配置

zxcar_nav工程参考: https://github.com/pirobot/ros-by-example/tree/master/rbx_vol_1/rbx1_nav

zxcar_driver : 主要负责与底盘进行通信

imu_calib: 校准imu安装误差和软件误差 ,参考 https://github.com/dpkoch/imu_calib 和 https://www.cnblogs.com/buxiaoyi/p/7541974.html

imu_filter_madgwick : 滤波器对IMU数据进行滤波,并将角速度信息转成四元素.参考 http://wiki.ros.org/imu_filter_madgwick

robot_localization : 非线性状态估计节点的程序包 参考http://docs.ros.org/melodic/api/robot_localization/html/index.html

move_base: 最重要的包 http://wiki.ros.org/move_base/

里程计信息获取参考:http://wiki.ros.org/navigation/Tutorials/RobotSetup/Odom

gmapping: http://wiki.ros.org/gmapping 和 https://github.com/ros-autom/slam_gmapping

# 搭建调试环境

执行依赖包的安装脚本

```
./install_melodic18.04.sh
./installPackages.sh
```

# 操控小车

## 校准imu误差

执行如下命令,会校准每一个轴的正负方向数据

```
rosrun imu_calib do_calib
```

## 校准角速度

```
rosrun heimarobot_nav calibrate_angular.py
```

执行完成之后,根据提示执行如下命令:

```
rosrun rqt_reconfigure rqt_reconfigure
```

## 校准线速度

```
rosrun heimarobot_nav calibrate_linear.py
```

执行完成之后,根据提示执行如下命令:

```
rosrun rqt_reconfigure rqt_reconfigure
```

# SLAM建图与导航

## 创建地图

当前面所有步骤执行都没有问题之后,下一个步骤我们就是来创建地图

我们这里默认使用的是gmapping建图算法

```
rosluanch heimarobot lidar_slam.launch
```

然后执行rviz我们可以查看建图相关的数据

```
roslaunch heimarobot lidar_slam_rviz.launch
```

在开启一个命令行窗口,打开我们的键盘控制程序,让小车随意溜达一圈

```
rosrun teleop_twist_keyboard teleop_twist_keyboard.py
```

当所有的地图边缘都已经闭合之后,我们就可以执行程序来保存地图,

```
roscd heimarobot/maps
./map.sh
```

## 自动避障导航

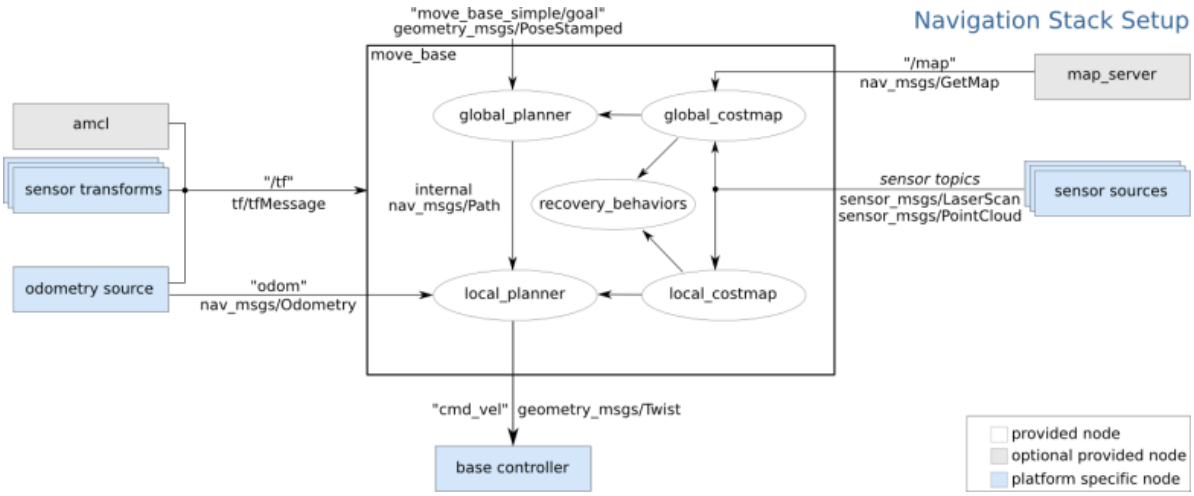
执行下面这条命令,打开自动导航程序

```
roslaunch heimarobot navigate.launch
```

执行下面这条命令,打开导航rviz窗口

```
roslaunch heimarobot navigate_rviz.launch
```

# move_base参数说明




move_base依赖4个重要的配置文件：

2. costmap_common_params.yaml： 全局代价地图和局部代价地图共有的配置
3. global_costmap_params.yaml： 全局代价地图参数配置
4. local_costmap_params.yaml： 局部代价地图参数配置
5. base_local_planner_params.yaml： 路径规划算法需要的配置

# costmap_common_params

| | |
|---|---|
| **obstacle_range** | **用来设置机器人检测障碍物的最大范围<br>设置为2.5意为在2.5米范围内检测到的障碍信息，才会在地图中进行更新** |
| raytrace_range | 用于清除指定范围外的空间。将其设置为3.0米，这意味着机器人将尝试清除3米外的空间，在代价地图中清除3米外的障碍物。 |
| footprint | 如果机器人外形是四边形，则需要设置机器人的四个角 |
| robot_radius | 如果机器人外形是圆形，则需要设置机器人的外形半径 |
| inflation_radius | 用来设置障碍物的膨胀参数，也就是机器人应该与障碍物保持的最小安全距离 |
| max_obstacle_height | 传感器读数的最大有效高度，单位为 m最大有效高度通常设置为略高于机器人的实际伸展的最大高度 |
| min_obstacle_height | 传感器读数的最小有效高度，单位为 m最大有效高度通常设置为略高于机器人的实际伸展的最大高度 |
| observation_sources | 代价地图需要关注的所有传感器信息 |
| sensor_frame | 标识传感器的参考系名称 |
| data_type | 表示激光数据或者点云数据使用的消息类型 |
| topic_name | 表示传感器发布的话题名称 |
| marking&clearing | 表示是否需要使用传感器的实时信息来添加或清楚代价地图中的障碍物信息 |

将膨胀半径设定为0.55米，意味着机器人针对相同的障碍物获取的所有路径都保持距离障碍物0.55米或更远。如果你的机器人不能很好地通过窄门或其它狭窄的地方，则稍微减小这个值，相反地，如果机器人不断地撞到东西，则尝试增大这个值。

# global_costmap_params.yaml

| **global_frame** | **用来表示全局代价地图需要在哪个参考系下运行** |
| --- | --- |
| robot_base_frame | 用来表示代价地图参考的机器本体参考系 |
| update_frequency | 用来决定全局地图信息更新的频率，单位是Hz |
| static_map | 用来设置代价地图是否需要根据map_server提供的地图信息进行初始化<br>如果不需要使用已有的地图或者map_server，最好将static_map设置为false |
| rolling_window | 用来设置在机器人移动过程中是否需要滚动窗口，以保持机器人处于中心位置 |
| resolution | 用来设置分辨率 |
| map_type | 用来设置地图类型，“voxel”表示3D视图，“costmap”表示2D视图 |

# local_costmap_params.yaml

| **publish_frequency** | **用来设置代价地图发布可视化信息的频率，单位是hz** |
| --- | --- |
| width&height | 用来设置代价地图宽高，单位是米 |
| resolution | 分辨率可以设置的与静态地图不同，但是一般情况下两者是相同的 |

# base_local_planner_params.yaml

| 参数 | 描述 |
| --- | --- |
| controller_frequency | 更新规划进程的频率，单位Hz |
| TrajectoryPlannerROS | 声明了机器的本地规划采用Trajectory Rollout算法 |
| max_vel_x | 机器x方向最大线速度，单位m/s |
| min_vel_x | 机器x方向最小线速度，单位m/s |
| max_vel_y & min_vel_y | 对于两轮差分驱动机器人是无效的 |
| max_vel_theta | 机器最大转动速度，单位rad/s |
| min_vel_theta | 机器最小转动速度，单位rad/s |
| min_in_place_vel_theta | 机器最小原地旋转速度 |
| escape_vel | 机器的逃离速度，即背向相反方向行走速度，单位是m/s |
| acc_lim_x | 在x方向上最大的线加速度，单位是m/s2 |
| acc_lim_theta | 最大角加速度，单位是rad/s2 |
| holonomic_robot | 除非拥有一个全方位的机器人，否则设置为false |
| yaw_goal_tolerance | 接近目标方向允许的误差（rad），此值太小会造成机器在目标附近震荡 |
| xy_goal_tolerance | 接近目标允许的误差（m），此值太小会造成机器在目标附近做小调整，而且此值不能小于地图的分辨率 |
| pdist_scale | 紧贴全局路径比到达目标的相对重要性，如果此值比gdist_scale大，那么机器会更紧靠全局路径行走 |
| gdist_scale | 到达目标比紧靠全局路径的相对重要性，如果此值比pdist_scale大，那么机器会采取任意路径优先到达目标 |
| meter_scoring | 表示gdist_scale和pdist_scale参数是否假设goal_distance和path_distance以米或者单元来表达 |
| occdist_scale | 控制器尝试避开障碍物的权重 |
| sim_time | 规划器能看到未来的时间，单位s |
| dwa | 当模拟轨迹走向未来，是否使用动态窗口法 |

# 参考

global path planner: 全局路径规划算法http://wiki.ros.org/global_planner

local path planner: 局部路径规划算法,并且实时避开动态的障碍物http://wiki.ros.org/base_local_planner

costmap_2d:代价地图http://wiki.ros.org/costmap_2d

move_base: http://wiki.ros.org/move_base?distro=kinetic

# 多机网络配置

树莓派端配置

```
export ROS_IP=`hostname -I | awk '{print $1}'`
export ROS_HOSTNAME=`hostname -I | awk '{print $1}'`
export ROS_MASTER_URI=http://`hostname -I | awk '{print
$1}'`:11311
```

PC端配置

```
export ROS_IP=`hostname -I | awk '{print $1}'`
export ROS_HOSTNAME=`hostname -I | awk '{print $1}'`
export ROS_MASTER_URI=http://树莓派的IP地址:11311
```