轮 趣 科 技

多机器人编队使用教程

推荐关注我们的公众号获取更新资料

版本说明：

版本	日期	内容说明
V1.0	2021/02/02	第一次发布
V2.0	2022/04/20	第二次发布
V2.5	2022/10/10	适配部分车型，增加编队模式 2
V2.6	2022/12/21	适配鲁班猫主控
V3.0	2023/03/15	优化通讯方式，适配更多数量编队

网址:www.wheeltec.net

序言

本文档主要讲解了多机器人编队功能包（wheeltec_multi）的使用方法，文档分为三个部分：第一部分主要是多机编队方法的相关简介；第二部分主要讲述了 ros 环境的设置，包括 udp 通信的搭建和搭建过程中需要注意的问题；第三部分阐述了多机编队功能包的具体使用方法。本文档目的是让用户能快速上手实现多机器人编队。

目录

1. 多机编队算法介绍

1.1 编队算法

本功能包就多机器人的编队控制问题作为协同控制中的一个典型问题进行研究，为后续对此课题进行研发的同学奠定了基础。编队控制算法是指控制多个机器人组成特定的某种队形来执行任务的算法，协作是指多个机器人之间利用某种约束关系合作来完成一项任务，以多机器人编队问题为例，协作就是指多个机器人共同组成期望的队形，其本质为各机器人的位置之间满足某种数学关系。编队方法主要分为集中式编队控制与分布式编队控制两种，集中式编队控制方法中主要有虚拟结构法、图论法和模型预测控制，分布式编队方法主要有跟随领航者法、 基于行为法和虚拟结构法。

本功能包使用分布式编队控制中的领航-跟随法实现多机器人编队，指定队形中的某一个机器人作为领航者(Leader)，其他的机器人作为 Slave 跟随领航者运动。该算法先利用 Leader 机器人的运动轨迹来分离出坐标，将此作为 Follower 机器人的期望跟踪位姿，让 Slave 跟随者以一定的方向、速度跟踪领航者，并通过与期望跟踪位姿的位置偏差修正 Slave 跟随者的方向、速度，最终使 Slave 跟随者与期望跟踪位姿位置偏差为零，从而达到编队目的。算法复杂度较低。

1.2 避障算法

常见的避障算法为人工势场法。将机器人在环境中的运动视为一种机器人在虚拟的人工受力场的运动，通过雷达识别最近的障碍物，该障碍物提供斥力场对机器人产生斥力，目标点提供引力场对机器人产生引力，通过斥力与引力的共同作用下控制机器人运动。

本功能包基于人工势场法作出了改进。先利用编队算法求出 Slave 跟随者的线速度与角速度，然后根据避障需求，对其进行增减变化，从而达到避障要求。当 Slave 跟随者与障碍物距离越近，障碍物对 Slave 跟随者斥力越大，线速度与角速度增减量变化的越大。当障碍物越靠近 Slave 跟随者的前方时，障碍物对 Slave 跟随者斥力越大（正前方斥力最大，正侧方斥力最小），线速度与角速度增减量变化的越大。通过改进人工势场法，优化改进了人工势场法对于障碍物在

机器人正前方，机器人会停止不动的缺点，从而更好了达到避障的目的。

2. ROS 环境设置

位姿未知的情况下，机器人之间需要通过通信分享彼此的信息，方便建立联系。

ROS 多机通讯中只允许存在一个 Master 节点管理器，运行 Master 节点管理器的一端称为主机，其他的一端称为从机，主从机都利用 Master 节点管理器进行通讯。然而编队数量越多，Master 节点管理器需要处理的数据量就会越大，就越容易出现因为网络拥堵而造成的通讯超时等问题，所以本功能不采用 ROS 多机通讯方式进行通讯。

本功能优化通信方式，每辆小车分别以自己作为主机运行 Master 节点管理器，保证定位、避障、编队等节点顺利运行。此时，主车（领航者 Leader）通过 udp 广播（一种网络通讯协议）不断广播其位姿等信息给从车（跟随者 Slave），从车接收到相关信息，发送给编队节点进行编队行为。利用 udp 广播的方式选择一些重要信息进行通讯，大大减轻了网络通讯的压力，使得编队数量不再容易受网络通讯的影响。

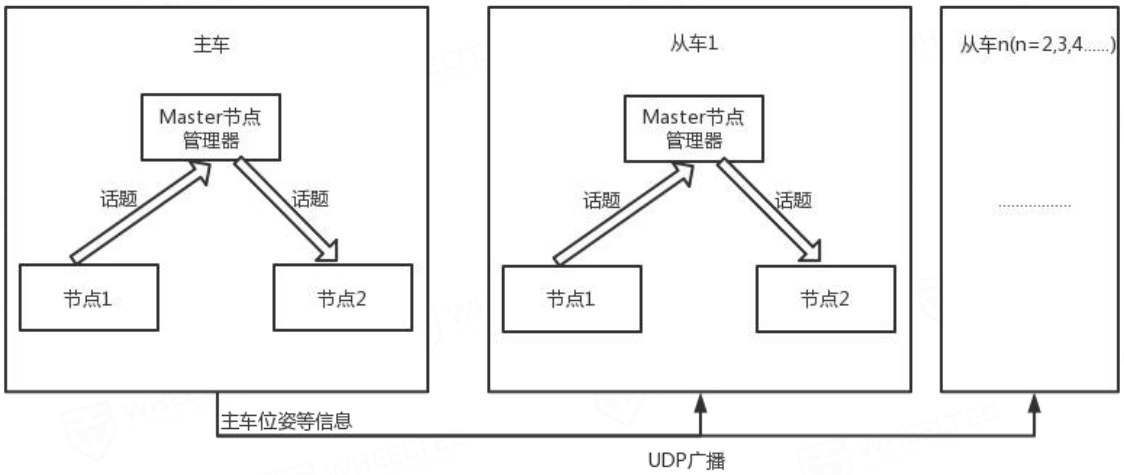

图 2-0-1　通信框架

2.1 实现 UDP 广播的步骤

① 将 ROS 主控处于同一个网络

将主车和从车的 ROS 主控均处于同一个网络下有两种方案。

一种是由主车的主控发射 wifi 提供网络。一般由其中一台车作为主车发出 wifi，其他车作为从车连接到这个 wifi。

另一种是由第三方路由器提供网络作为信息中转，所有车辆连接同一路由器。该路由器不联网也可以使用。

方案可以根据实际的需求来选择，如果进行通信的机器人数量较少，则建议使用方案一，可以节约成本，设置更简便；而当需要进行多机通信的机器人数量较多时，推荐使用方案二，因为 ROS 主车的板载 wifi 性能有限，使用路由器可以增强网络的稳定性，更大的带宽更加保证的通讯的质量。

需要注意的是，虚拟机想要获取其中一辆车的话题节点，则必须要通过多机通讯方式，其网络模式需要设置成桥接模式。设置步骤为：

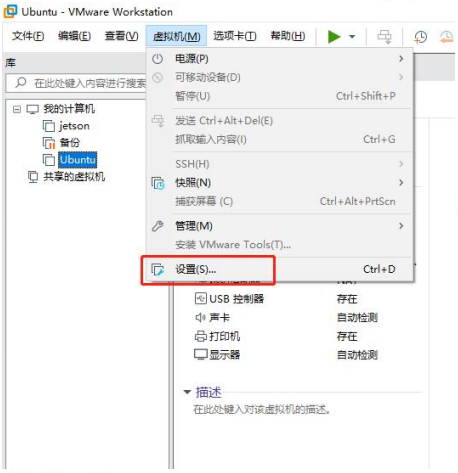

图 2-1-1　虚拟机网络设置桥接模式步骤 1

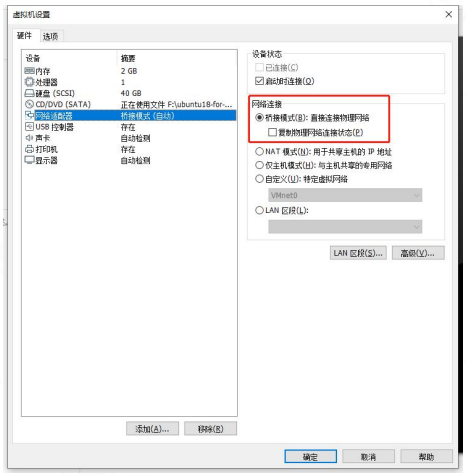

图 2-1-2　虚拟机网络设置桥接模式步骤 2

登录虚拟机后，点击右上方电源按钮，在“有线连接”中除了不能选择“Static IP”外其他两个都行。保证虚拟机 ip 地址与小车在同一网段即可。

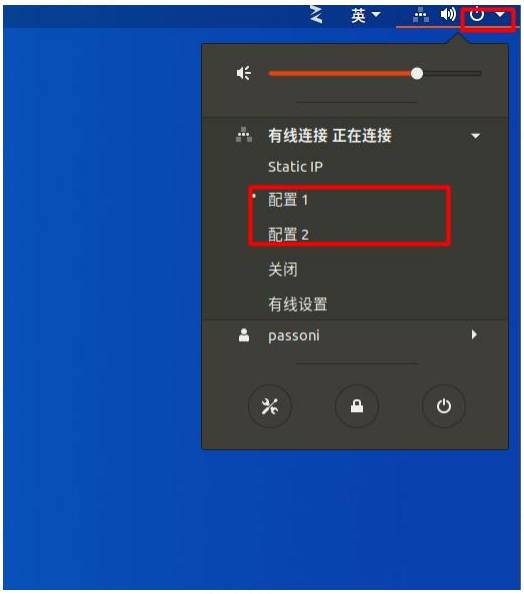

图 2-1-3　虚拟机网络连接方式选择

② 设置 ROS 环境变量

ROS 主控连接同一个网络，同时要保证各个主控的 ip 地址不一致。

ROS 通信的环境变量的设置在主目录下的.bashrc 文件，运行 gedit ~/.bashrc 命令打开它，需要更改 ROS_MASTER_URI 和 ROS_HOSTNAME，两者需要设置成小车端连接 wifi 后的 ip 地址，即 2.2 节自动连接 wifi 操作步骤八中设置的 ip 地址（见图 2-2-7）。

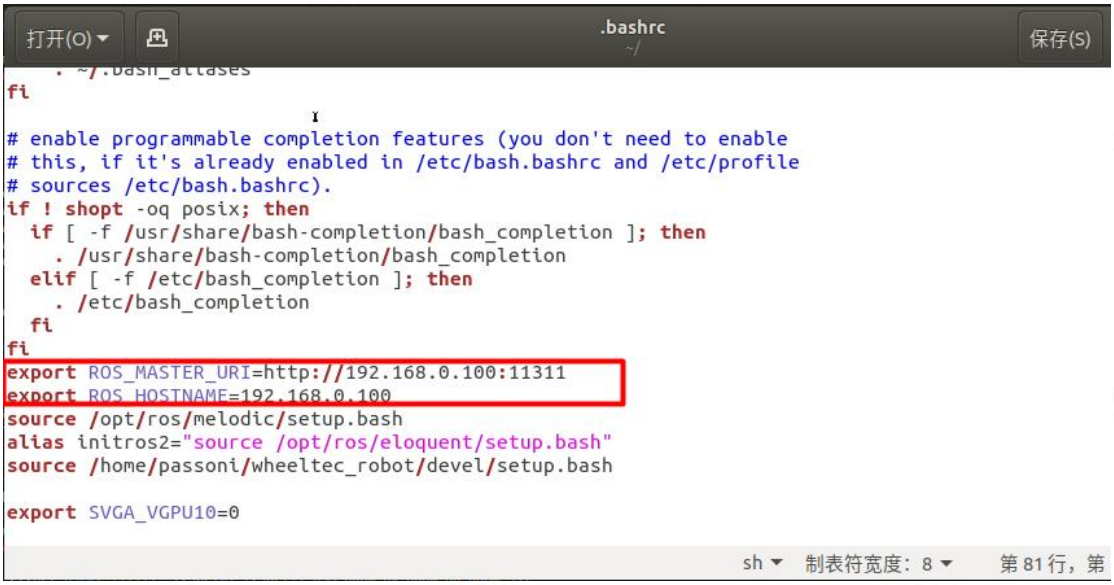

图 2-1-4　环境配置文件.bashrc

本功能采用的 udp 通信是不受 ROS 系统版本限制的，在通信的过程中，有以下这些注意事项：

1) 通信里的所有主控 ip 地址需要处于同一个网段，即 ip 地址和子网掩码相与得到相同的网络地址。
2) 各个小车的 ROS_MASTER_URI 和 ROS_HOSTNAME 的 ip 地址与 2.2 节自动连接 wifi 操作步骤八中设置的 ip 地址一致。
3) 因为各个小车不采用 ROS 多机通讯方式，各个小车无法知道其余小车的话题、节点等信息。
4) 如果是虚拟机想要获取其中一辆小车的话题信息，建议先 ssh 登录这辆小车，然后再获取。

2.2 ROS 开机自动连接 wifi 设置

此步骤是针对需要自动连接主车网络或者路由器网络的小车进行网络配置。

① Jetson nano 自动连接 wifi 设置

首先使用 VNC 远程工具或将 Jetson nano 连接至屏幕。点击右上角的 wifi 图标，单击”Edit Connections..”

图 2-2-1 Jetson nano 连接 wifi 步骤（一、二）

在网络连接中点击+按钮。

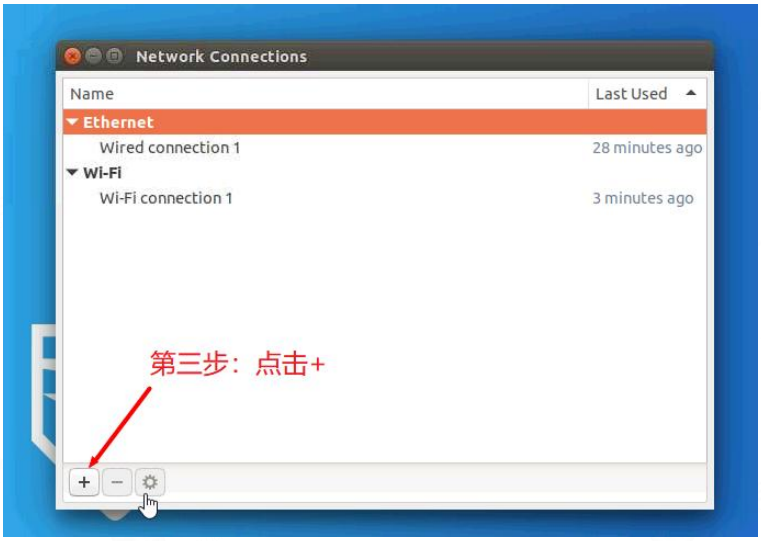

图 2-2-2　Jetson nano 连接 wifi 步骤（三）

在“Choose a Connection Type”弹窗中，选择“Wi-Fi”选项，然后点击“Create...”

图 2-2-3　Jetson nano 连接 wifi 步骤（四）

在设置面板中点击“Wi-Fi”选项，然后按照下面的图片设置填写相关内容：

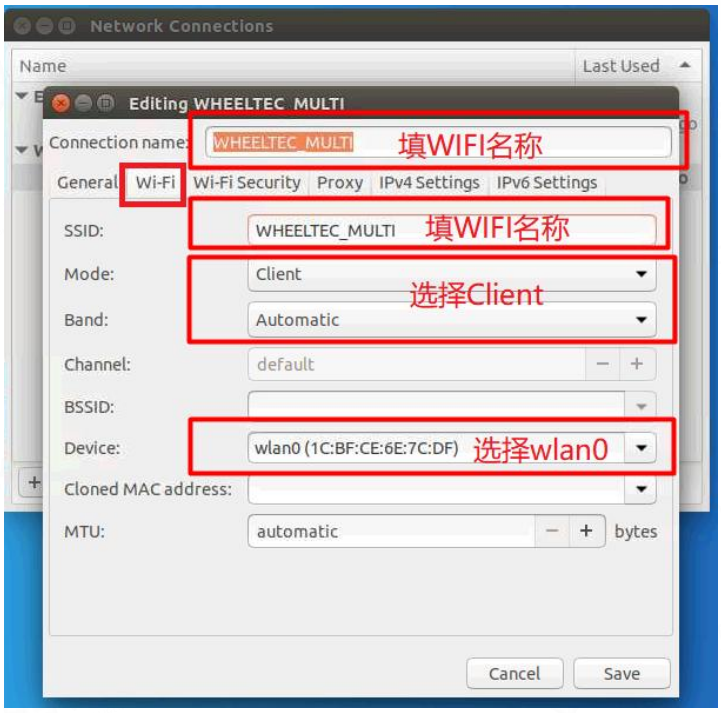

图 2-2-4　Jetson nano 连接 wifi 步骤（五）

在设置面板中点击”General”选项，勾选“Automatically connect to this network...”之后，在“Connetion priority for auto-activation”这一选项中设置连接优先级为 1，勾选“All users may connect to this network”栏。其他的 wifi 在“Connetion priority for auto-activation”这一选项设置为 0，表示在历史连接上的 wifi 中优先连接至此 wifi。

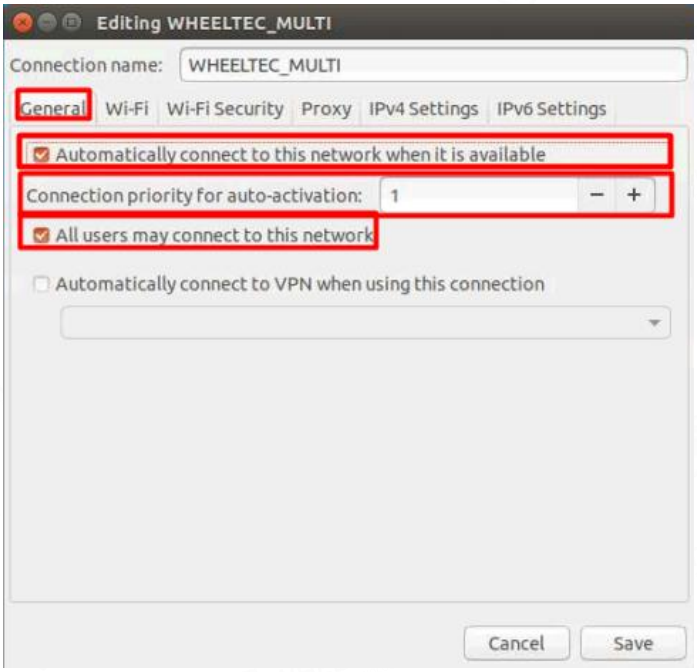

图 2-2-5　Jetson nano 连接 wifi 步骤（六）

在设置面板中点击”Wi-Fi Security”选项，在“Security”选项中选择“WPA & WPA2 Personal”，然后在“Password”中填入 WIFI 密码。

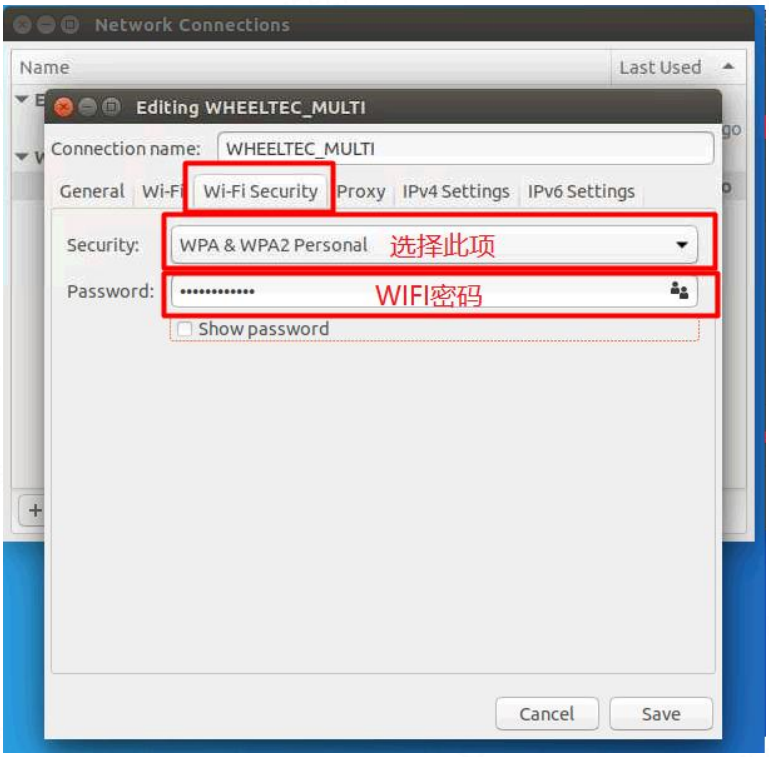

图 2-2-6　Jetson nano 连接 wifi 步骤（七）

这里需要注意的是：如果设置了优先级为 0 的 wifi，在开机之后无法自动连接，可能是 wifi 信号强度的问题，为了避免这一问题，可以选择将历史连接过的 wifi 选项全部删除，只留下主车或路由器发出的 wifi。

在网络设置面板中点击”IPv4 Settings”选项，在“Method”中选择“Manual”选项。然后点击“Add”，在“Address”中填入从机的 IP 地址，在“Netmask”填入“24”，在“Gateway”中填入 IP 网段，IP 网段即把 IP 地址后三位数字改成“1”即可。该步骤主要目的为固定 IP 地址，第一次设置完成，后续连接同一 WIFI 时 IP 地址会保持不变。

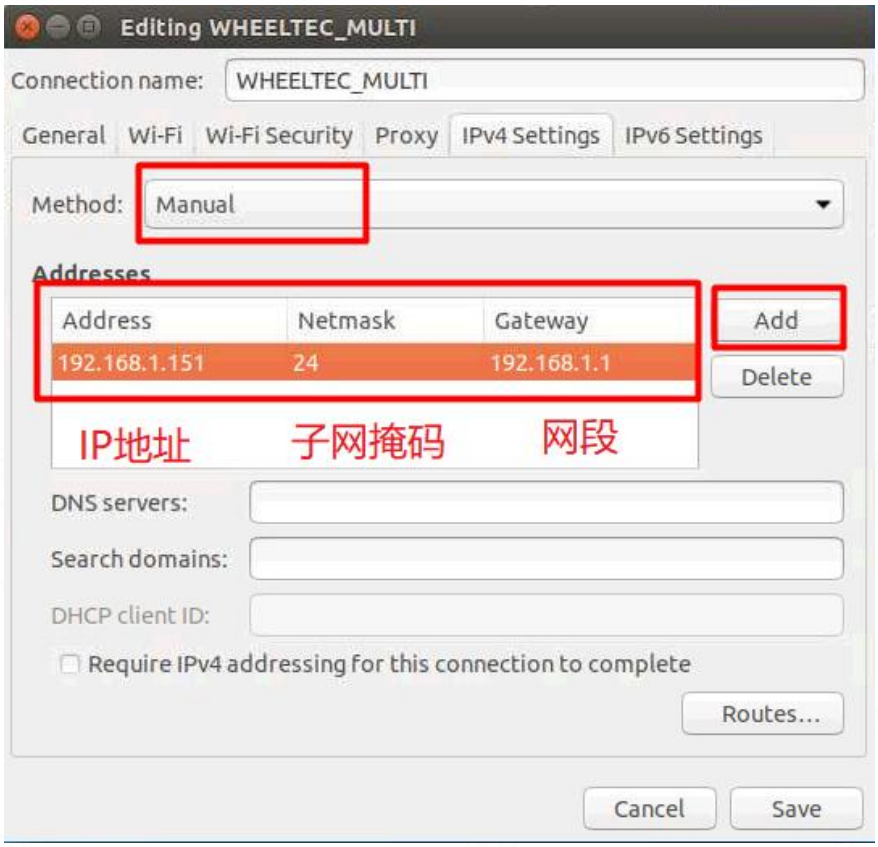

图 2-2-7　Jetson nano 连接 wifi 步骤（八）

全部设置完成后，点击“save”保存设置。保存成功后，后续小车上电就会自动连接主车或者路由器的网络。

这里需要注意的是：

1) 这里设置的 IP 需要和本地端.bashrc 文件中 ROS_MASTER_URI 和 ROS_HOSTNAME 设置的 IP 地址一致。
2) 不同小车设置的 IP 地址必须保持不一致。
3) 主从机 ip 地址需要处于同一个网段。
4) 主车或者路由器先开机，待其发出 WiFi 后，从车才能开机自动连接 WiFi。
5) 如图 2-2-7，网段处必须填写完整，否则无法进行 UDP 通讯。

② 树莓派自动连接 wifi 设置

在树莓派中连接 wifi 的设置与在 jetson nano 的设置方法一致。

③ jetson TX1 自动连接 wifi 设置

在 jetson TX1 中连接 wifi 的设置与在 jetson nano 的设置方法大致一致，有一点与 jetson nano 不同的是，jetson TX1 在网络设置面板中，“Device”应选择

“wlan1”的设备。

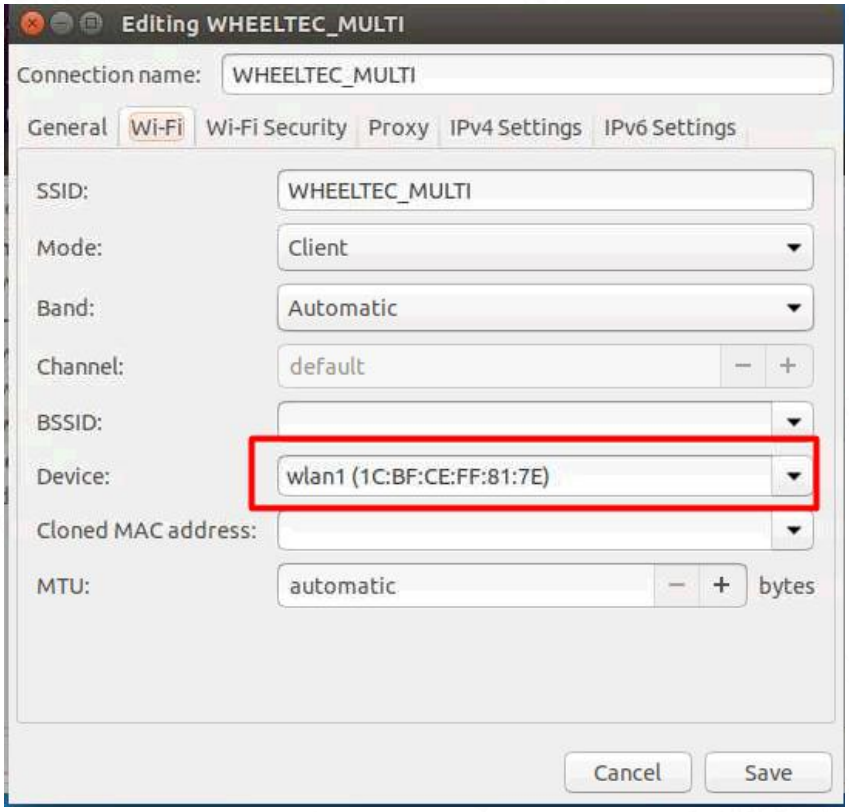

图 2-2-8　Jetson TX1 连接 wifi 设置

④ 鲁班猫自动连接 wifi 设置

在鲁班猫中连接 wifi 的设置与在 jetson nano 的设置方法一致，但需要增加的一步操作是：取消勾选其他 wifi 的“Automatically connect to this network when .....”选项，让其他 wifi 开机时不能自动连接上，确保当前自动连接的只有之前设置好的 wifi。

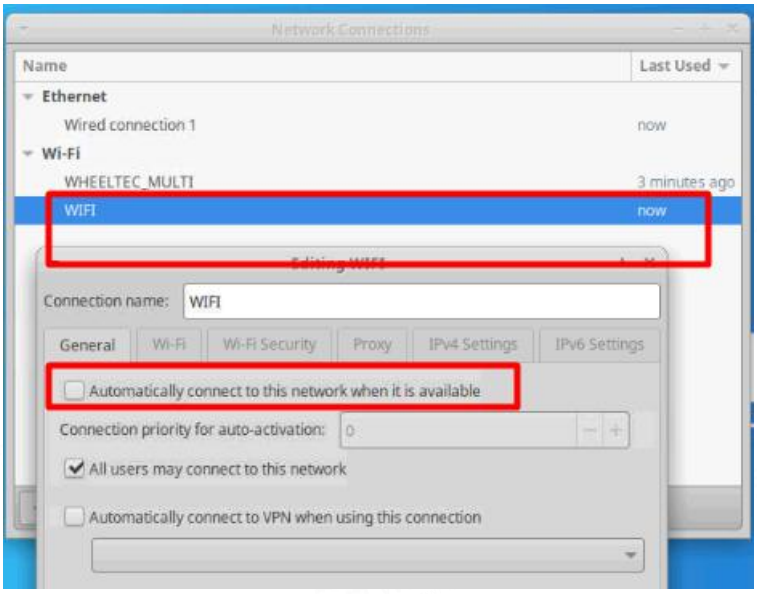

图 2-2-9　鲁班猫连接 wifi 设置

2.3 功能包编译

功能包默认已编译安装在小车中。

需要安装的，先联网运行下面的命令安装相关依赖

```
pip install paramiko
```

然后把功能包 wheeltec_multi 拷贝到工作空间中，编译即可

```
catkin_make -DCATKIN_WHITELIST_PACKAGES=wheeltec_multi
```

3. 多机编队功能包使用

3.1 功能包简介

① 设置从车坐标（需要设置）

wheeltec_multi 功能包可实现自定义的编队队形，需要不同的队形时，只需要修改从车的期望坐标即可。slave_x、slave_y 是以主车为原点的从车 x、y 坐标，单位：m，以主车前方为 x 坐标正方向，左侧为 y 坐标正方向。

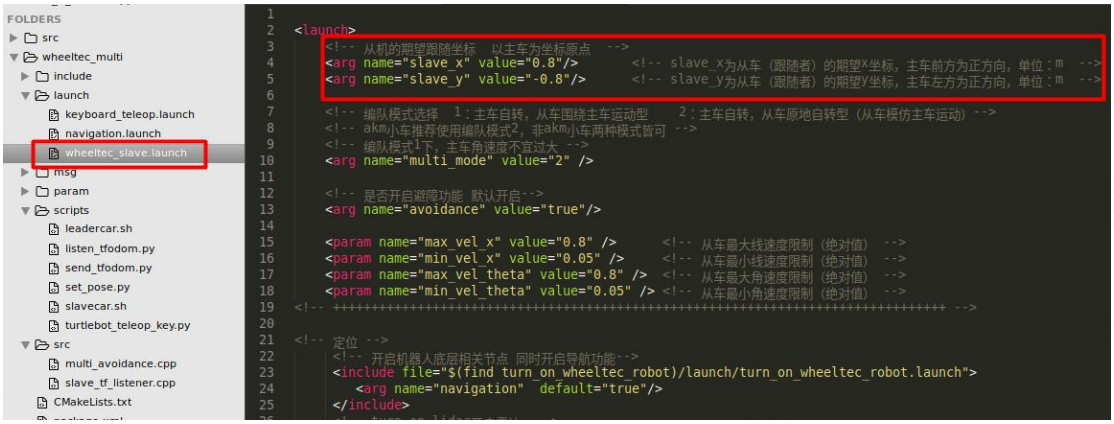

图 3-1-1 设置从车坐标

如有一辆主车，两辆从车的情况下，可设置如下队形：

1）横队队形：可将左侧从车坐标设置为：slave_x：0，slave_y：0.8，右侧从车坐标设置为：slave_x：0，slave_y：-0.8。

2）纵队队形：可将一从车坐标设置为：slave_x：-0.8，slave_y：0，另一从车坐标设置为：slave_x：-1.8，slave_y：0。

3）三角队形：可将一从车坐标设置为：slave_x：-0.8，slave_y：0.8，另一从车坐标设置为：slave_x：-0.8，slave_y：-0.8。

其他队形根据需要可自定义。

注意：如果编队模式选择编队模式 1，从车与主车距离尽量设置小点，推荐设置范围为 0.6~1.8，根据车辆大小进行相应的变化，距离主车越远，主车在做转弯运动时从车的线速度会越大，由于最大速度的限制，从车速度达不到要求就会偏离轨迹，编队队形变乱。

如果编队模式选择编队模式 2，则从车与主车的距离在 0.6 以上即可（0.6 为避障范围接线，在 0.6 范围内从车会有避障动作）。

② 从车位姿初始化设置（可选）

从车的初始位姿默认为从车的期望坐标位置。在运行程序前，只需要把小车放在从车期望坐标附近位置，便可完成从车位姿的初始化。

该功能具体是在 wheeltec_multi 功能包下面的 wheeltec_slave.launch 文件中的 pose_setter 节点中实现。

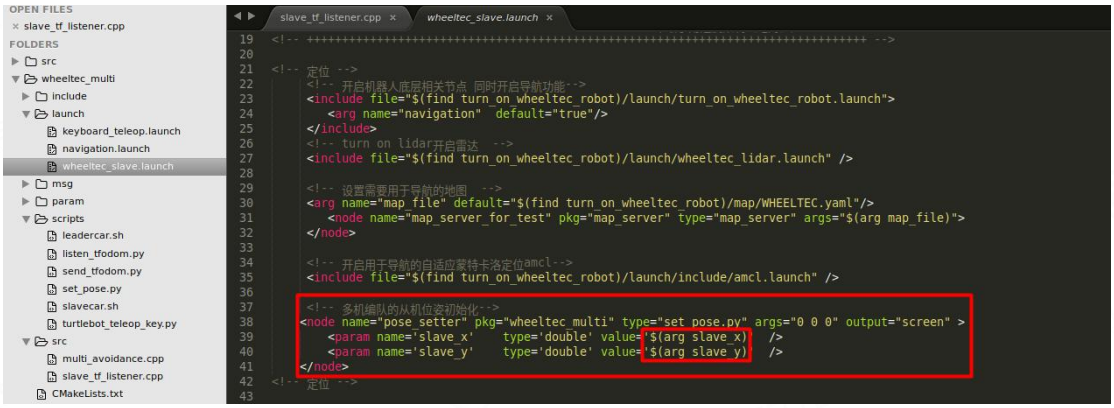

图 3-1-2　从车位置初始化节点

若用户想自定义从车的初始位置，只需要在 wheeltec_slave.launch 中设置 pose_setter 节点中的 slave_x、slave_y 值便可。在运行程序前，只要把小车放在自定义的位置便可。

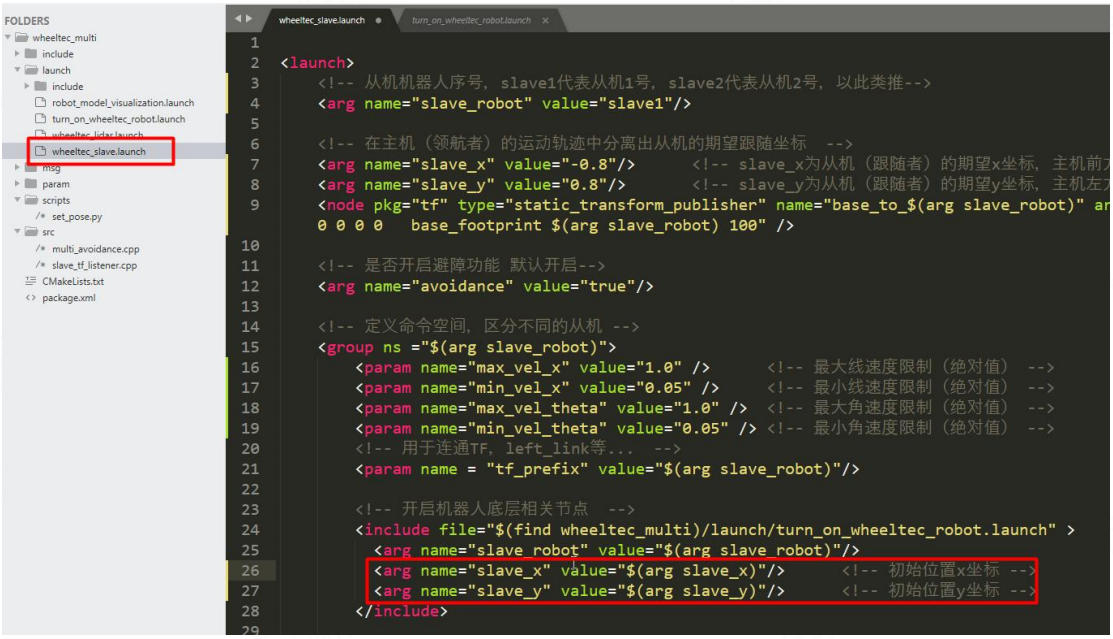

图 3-1-3　从车初始位置可自定义

③ 定位问题（需要设置）

在多机编队中，首先解决的是主从车定位问题，用来确定主从车之间的相对位置关系。

首先主从车都需要有 2D 地图，运行自适应蒙特卡洛定位（amcl 定位）功能，借助自适应蒙特卡洛定位（amcl 定位）来实现定位。要想确认主从车之间的相对位置关系，就需要两者的地图一致，主从车分别利用 amcl 定位知道自己相对于地图原点的坐标，因为两者的地图一致，地图原点位置重合，就可以求出两者

的相对位置关系，作为 slave_tf_listener 编队节点计算从车速度的基础信息。

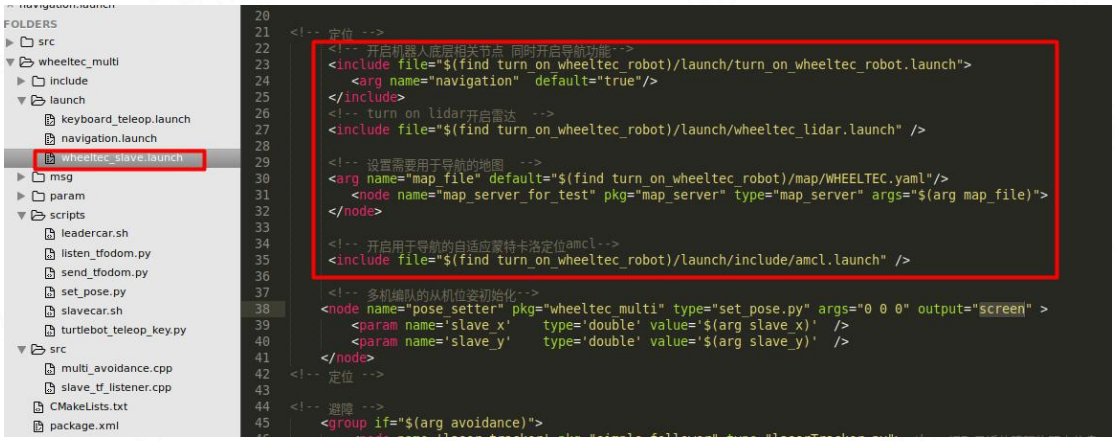

图 3-1-4　从车定位节点

操作步骤：

a. 主车运行 2D 建图功能进行建图，并保存地图（地图默认保存在 turn_on_wheeltec_robot 功能包下面的 map 文件夹里面）

b. 将主车的地图文件 WHEELTEC.yaml 与 WHEELTEC.pgm 文件复制到从车的 turn_on_wheeltec_robot 中的 map 文件夹里：可以拿 U 盘拷贝，也可以 ssh 远程到主车端，在主车端运行以下命令进行复制：

```
scp -r /home/wheeltec/wheeltec_robot/src/turn_on_wheeltec_robot/map/WHEELTEC*
wheeltec@192.168.0.101:/home/wheeltec/wheeltec_robot/src/turn_on_wheeltec_robot
/map/
```

格式为：“scp　-r 主车地图文件路径 从车用户名@从车 ip 地址：从车 map 文件夹路径”，逐一修改命令中的从车 ip 地址，运行命令复制给所有从车即可。（需要主从车在同一网络下，详情请参照第 2 章内容）

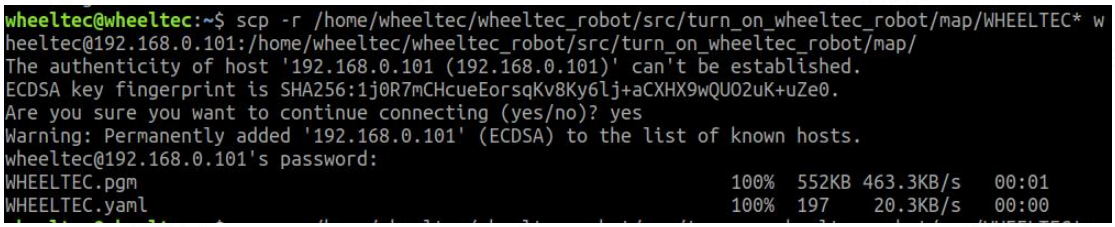

图 3-1-5　主车端 map 地图复制给从车端

④ 队形生成、队形保持问题（可选）

在编队运动的过程中，主车运动可通过 rviz、键盘控制、遥控器控制等多种方式控制。从车则通过 slave_tf_listener 节点计算得到从车的速度，从而控制从车的运动，达到编队的目的。

在 slave_tf_listener 节点中，对从车的运动速度进行限制，避免节点计算输出的从车速度过大，造成一系列影响。具体的数值可在 wheeltec_slave.launch 中修改。

图 3-1-6 速度限制

编队算法相关参数如下图：

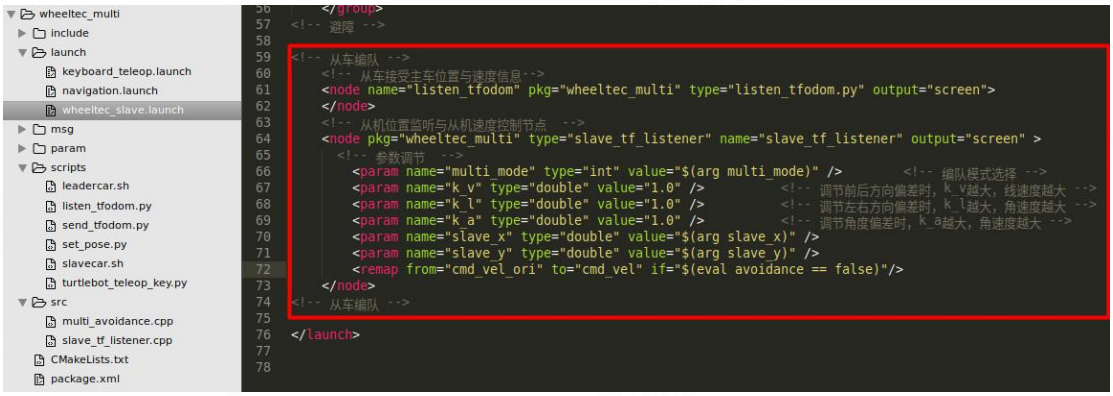

图 3-1-7 编队算法

在编队中，主车通过 send_tfodom 节点发布 UDP 广播信息，信息包含了主车的位置以及速度信息，从车通过 listen_tfodom 节点，接受从主车发出的 UDP 广播信息，接收主车的位置以及速度信息，并通过 multfodom 话题发布出来，编队节点 slave_tf_listener 订阅 multfodom 话题，然后进行编队。

⑤ 编队避障问题（可选）

在多机编队中，通过键盘控制主车运动，可以控制主车进行避障。但从车的速度由 slave_tf_listener 编队节点在不考虑障碍物作用下计算得到，有障碍物下直接输出给从车运动会导致从车与障碍物相撞。

从车实现避障功能，就需要 multi_avoidance 避障节点进一步处理编队节点

输出的从车原速度，结合障碍物信息输出从车最终的速度信息，从而实现从车避障。

在功能包中默认开启 multi_avoidance 避障功能，有需要的话可将 avoidance 参数设置为 false 关闭避障节点，从而关闭避障功能。

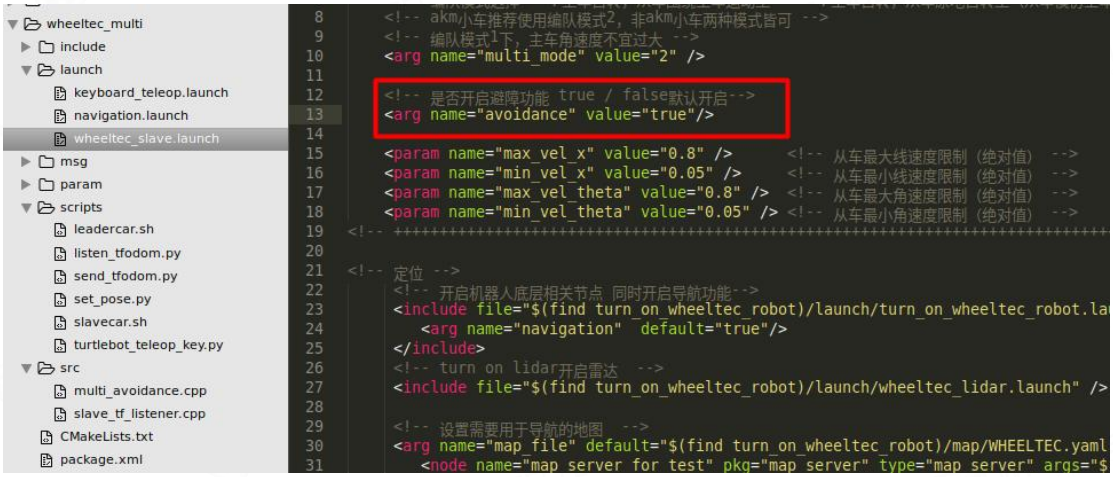

图 3-1-8　开启从车避障节点

避障节点的一些相关参数如下图，其中 safe_distance 为障碍物安全距离界限，danger_distance 为障碍物危险距离界限。当障碍物处于 safe_distance 与 danger_distance 之内时，从车调整位姿避让障碍物。当障碍物处于 danger_distance 之内时，从车以远离障碍物为主。

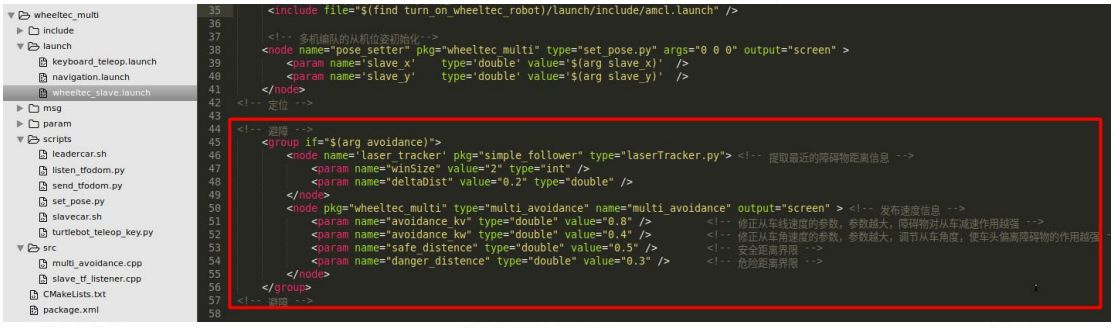

图 3-1-9　从车避障相关参数

⑥ 多机编队模式选择（可选）

多机编队设置两种编队模式：编队模式 1 和编队模式 2，用户在 wheeltec_slave.launch 中修改 multi_mode 参数即可，编队模式 1 设置 multi_mode 参数为 1，编队模式 2 设置 multi_mode 参数为 2。

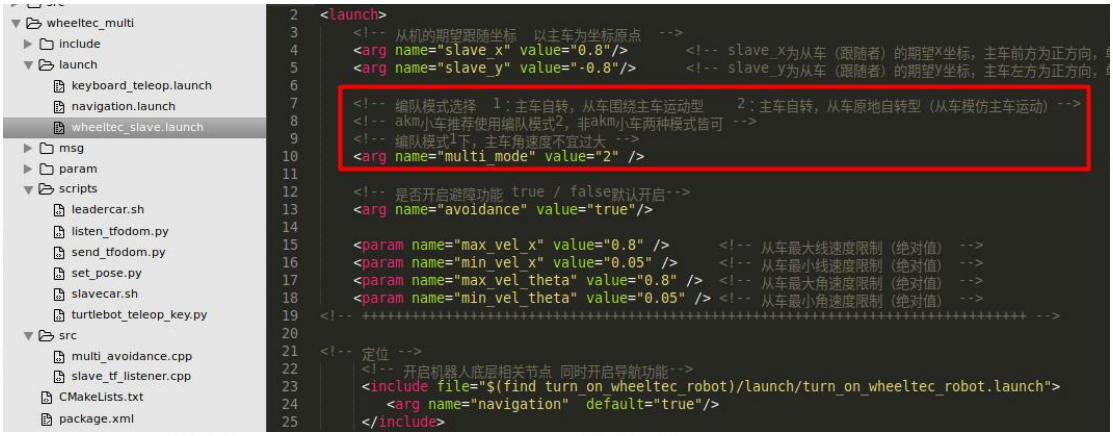

图 3-1-10 多机编队模式选择

阿克曼车推荐使用编队模式 2。非阿克曼车编队模式 1 或者编队模式 2 都可以。

编队模式 1 与编队模式 2 区别如下图所示。

模式 1：主车原地自旋转时，从车绕主车旋转，即队形整体旋转

模式 2：主车原地自旋转时，从车也原地自旋转，即队形内小车各自旋转，队形整体方向不变

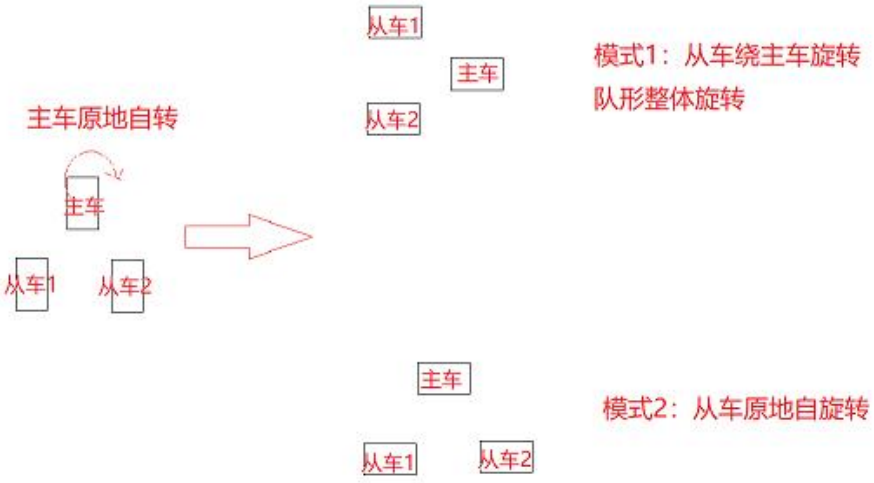

图 3-1-11 不同编队模式的区别

3.2 上手使用流程

① 输入执行指令

使用多机编队前准备：

✓ 主车从车连接同一网络并正确设置 ROS 环境

✓ 主车提前进行 2D 建图，并把地图复制给从车，保证主从车地图一致

✓ 主车放在建图起点，从车放在初始化位置（默认从车编队位置）附近

5 车编队以下两种执行命令方式皆可，6 车编队以上请使用执行命令方式二。

执行命令方式一：

第一步：在主车端中运行多机编队导航程序

```
roslaunch wheeltec_multi navigation.launch
```

第二步：在从机端中分别运行编队程序

```
roslaunch wheeltec_multi wheeltec_slave.launch
```

第三步：在主车端打开多机编队键盘控制节点控制主车运动

```
roslaunch wheeltec_multi keyboard_teleop.launch
```

执行命令方式二：

第一步：修改主车 wheeltec_multi 功能包下的 cmd.py 文件，将你当前所要操作所有编队小车的 ip 地址写进 ip 变量里面

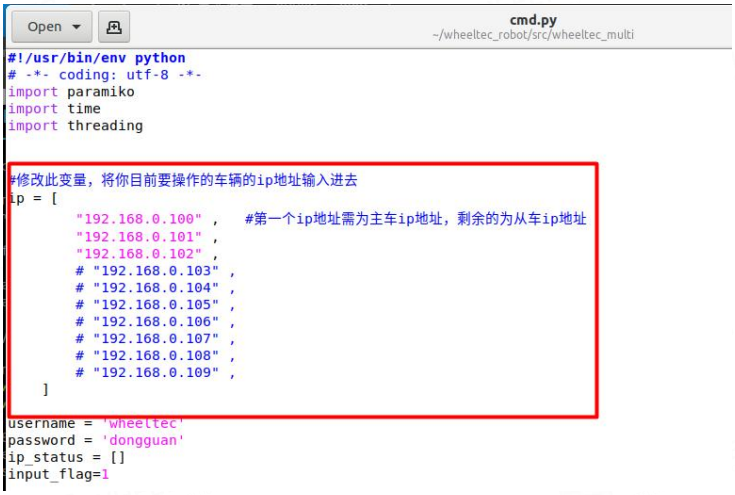

图 3-2-1 cmd.py 文件

第二步：ssh 远程主车，先运行 cd ~/wheeltec_robot/src/wheeltec_multi，进入 wheeltec_multi 文件夹，然后执行 cmd.py 文件

```
python cmd.py
```

图 3-2-2 cmd.py 运行操作步骤 1

回车键确认。等待小车初始化完成后，输入 0，然后再输入 open 开启所有

小车编队功能：

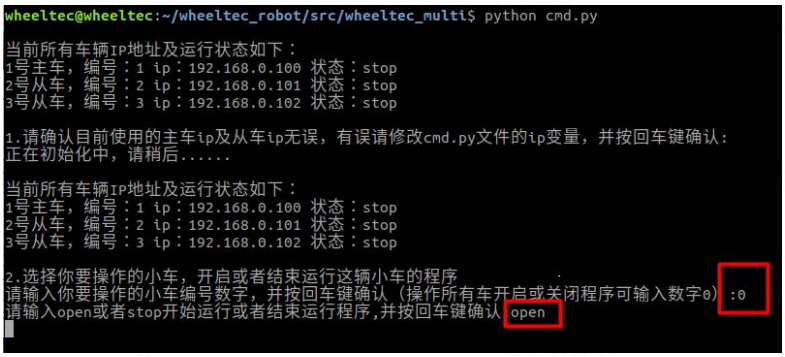

图 3-2-3　cmd.py 运行操作步骤 2

期间你可以根据 cmd.py 命令提示，控制某一辆小车开启或者关闭功能。

第三步：在主车端打开多机编队键盘控制节点控制主车运动

```
roslaunch wheeltec_multi keyboard_teleop.launch
```

② 使用注意事项

1. 执行程序前一定要确认主车从车连接同一网络。

2. 在多机编队选择编队模式 1 的时候，控制多机编队的主车时，角速度不宜过快。建议线速度 0.2m/s，角速度 0.3rad/s 以下。

3. 因为车与车之间话题数据不进行通讯连接，要获取某一车的话题数据，建议 ssh 远程该车然后在获取。

4. 主车的网络设置面板中，需要将网段填写进去，否则 UDP 无法通讯，从车无法发出/multfodom 话题。

5. 若从车没有动作，请检查从车的 multfodom 话题是否发出数据，能否单独运行 2D 导航功能，若无法正常发出 multfodom 话题，请检查第 4 点。

6. 当小车有两个 ip 地址时（如使用网口雷达时，一个 ip 地址为雷达目的 ip，一个 ip 地址为小车 wifi），主车的 send_tfodom.py 文件的第 34 行添加一句代码如下图，其中的 ip 地址填写主车 wifi 的 ip 地址。

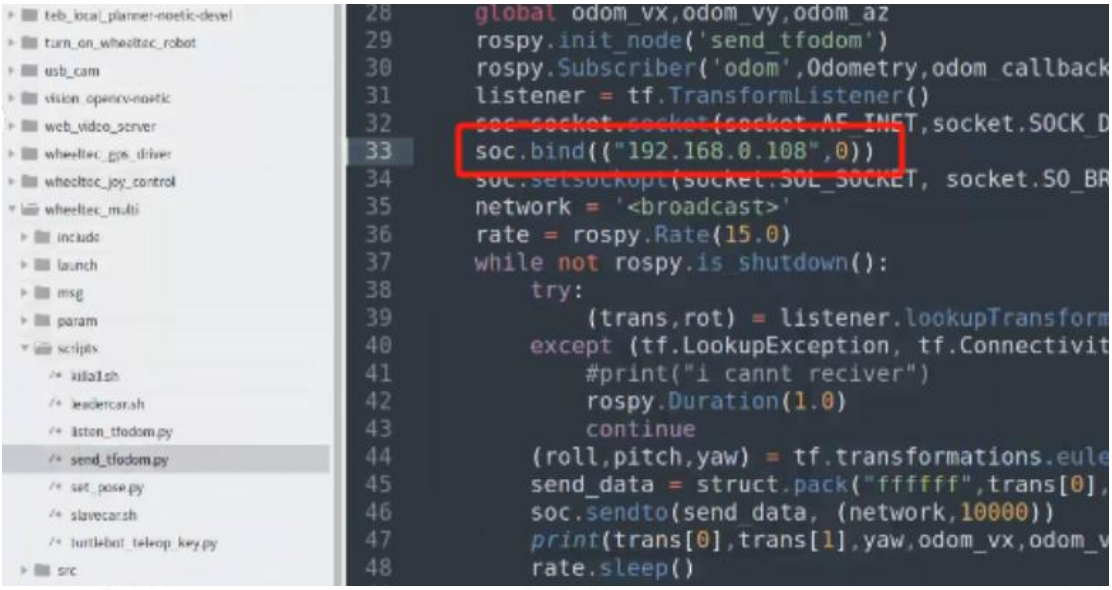

图 3-2-4 send_tfodom.py 文件修改