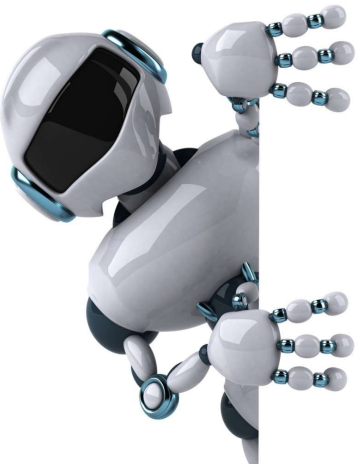
天之博特陆空协同训练营

DAY2：算法案例与路径跟踪演练

主 讲 人：王晋泰

天之博特



目录

01 abc_swarm功能包介绍

02 实例分析：领航跟随-多车

03 教学练习：路径跟踪-车车

04 陆空协同套装网络配置

05 无人车跟随无人机任务分解

abc_swarm功能包介绍

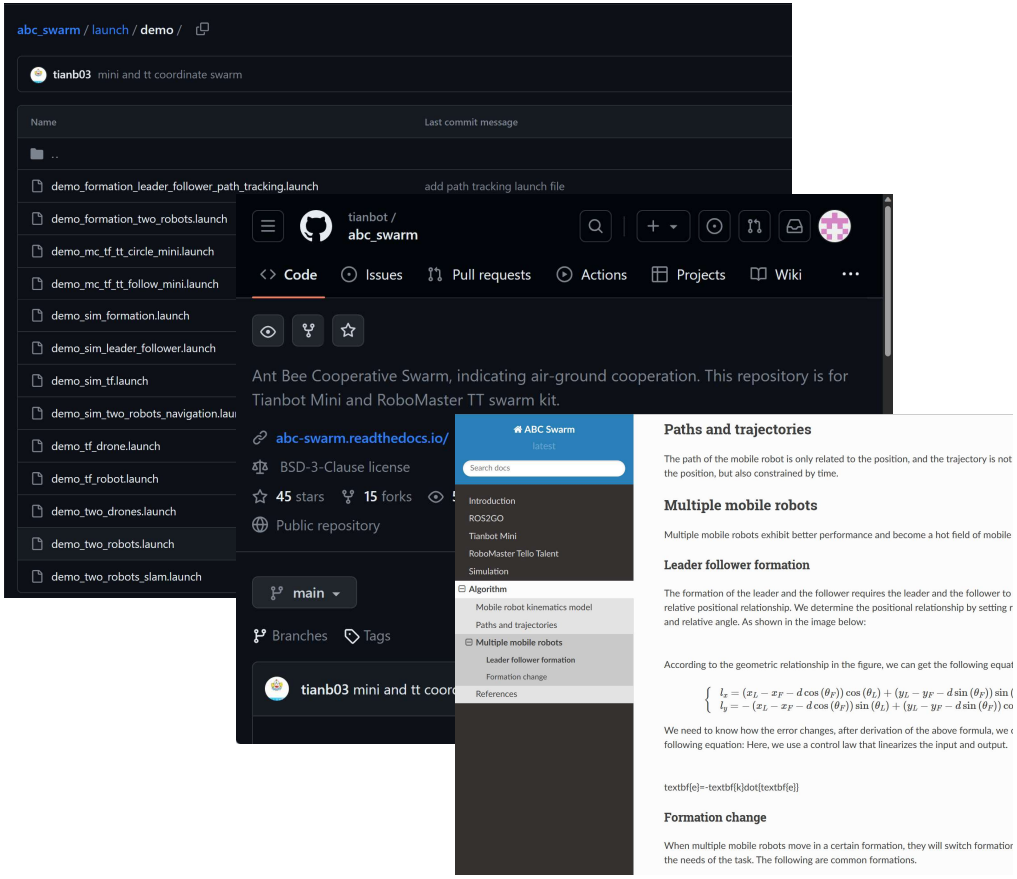
陆空协同套装配套功能包

1

功能介绍

abc_swarm 功能包的设计目标是帮助用户快速搭建和调试陆空协同控制系统，提供了一套工具和示例，支持在真实环境和仿真环境中进行测试和应用。

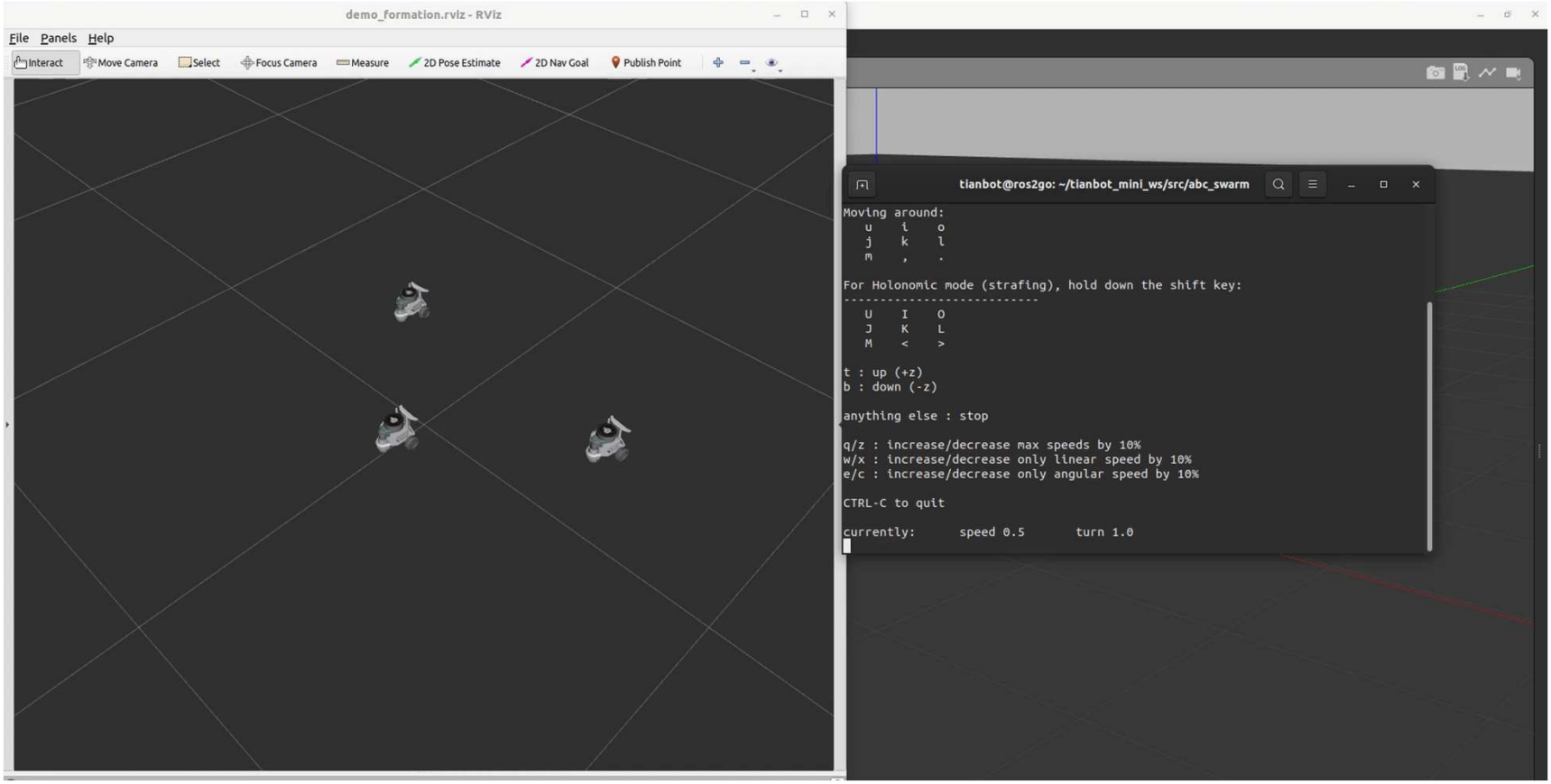
它对于机器人或无人机领域的研究人员和开发者来说，是一个有价值的资源和参考。通过使用abc_swarm功能包，用户可以更轻松地实现多机器人或多无人机的协同控制，以及教学场景的使用。



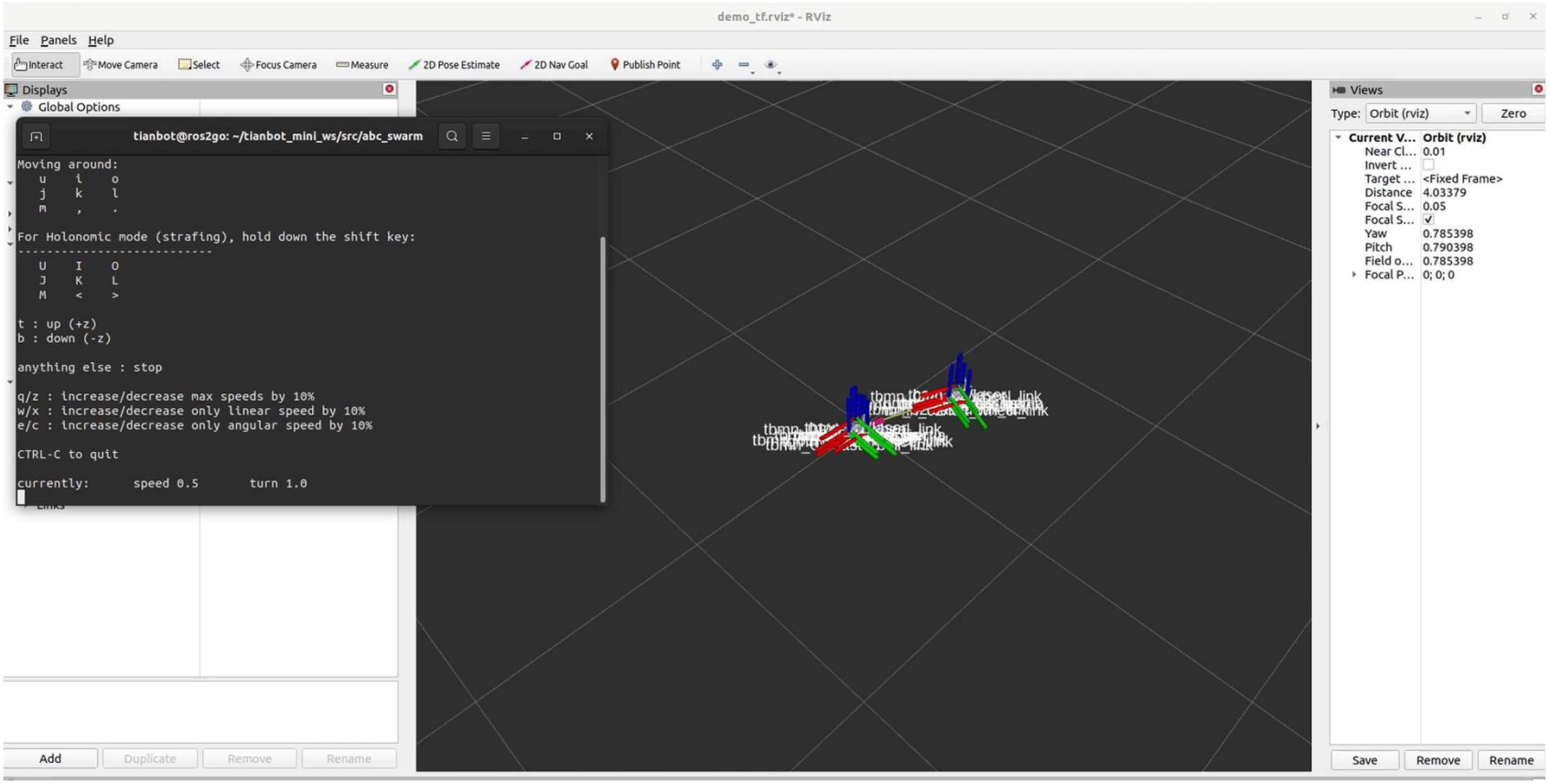
功能介绍

【开源】ROS机器人集群的仿真与实践详解一：多机协同项目abc_swarm讲解支持无人机 多移动机器人轻松组队集群，田博主讲_哔哩哔哩_bilibili

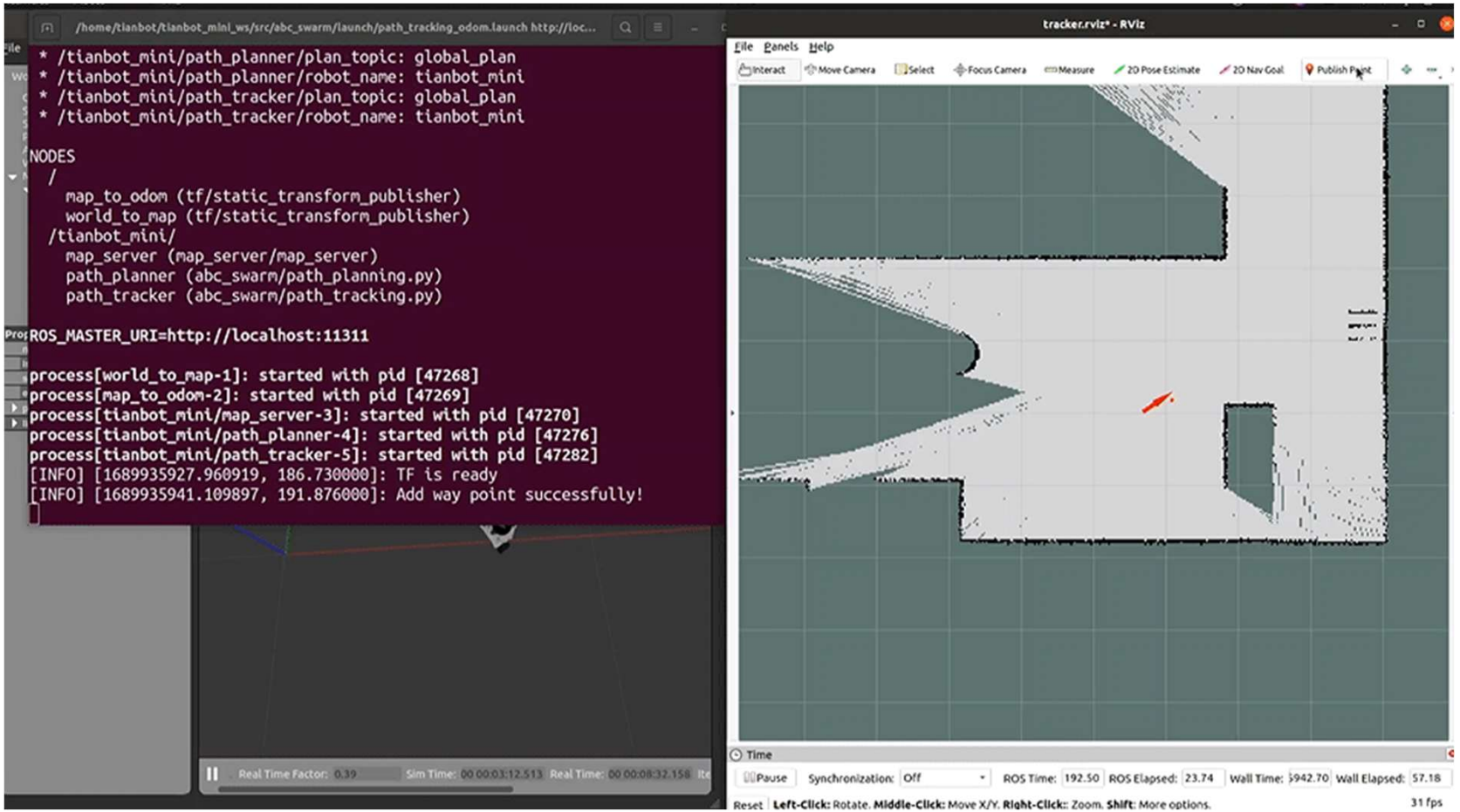
功能介绍


demo_formation.rviz - RViz
File Panels Help
Interact Move Camera Select Focus Camera Measure 2D Pose Estimate 2D Nav Goal Publish Point
tianbot@ros2go: ~/tianbot_mini_ws/src/abc_swarm
Moving around:
u i o
j k l
m , .
For Holonomic mode (strafing), hold down the shift key:
U I O
J K L
M < >
t : up (+z)
b : down (-z)
anything else : stop
q/z : increase/decrease max speeds by 10%
w/x : increase/decrease only linear speed by 10%
e/c : increase/decrease only angular speed by 10%
CTRL-C to quit
currently: speed 0.5 turn 1.0

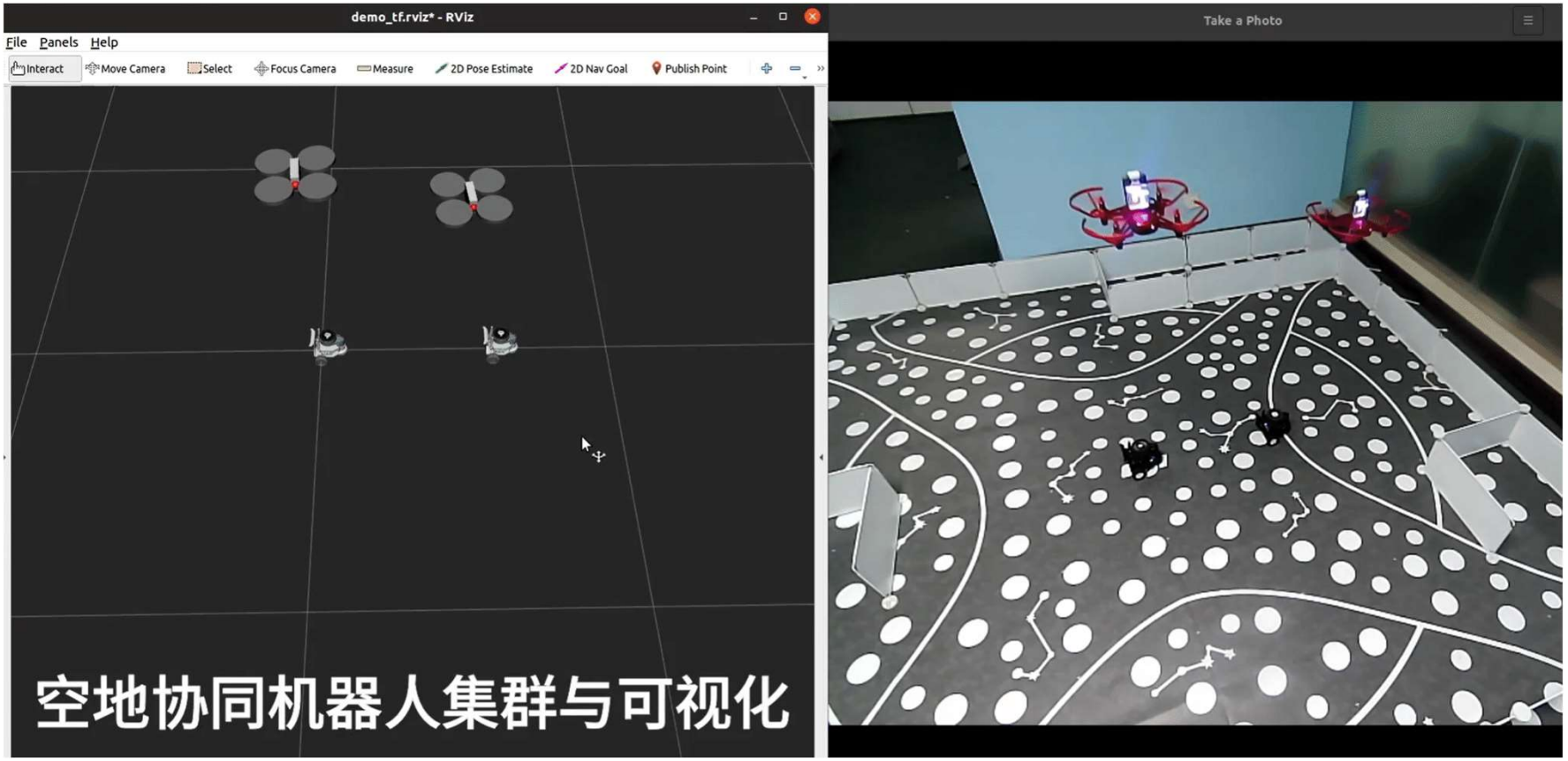
功能介绍


demo_tf.rviz* - RViz
File Panels Help
Interact Move Camera Select Focus Camera Measure 2D Pose Estimate 2D Nav Goal Publish Point
Displays
Global Options
tianbot@ros2go: ~/tianbot_mini_ws/src/abc_swarm
Moving around:
u i o
j k l
m , .
For Holonomic mode (strafing), hold down the shift key:
U I O
J K L
M < >
t : up (+z)
b : down (-z)
anything else : stop
q/z : increase/decrease max speeds by 10%
w/x : increase/decrease only linear speed by 10%
e/c : increase/decrease only angular speed by 10%
CTRL-C to quit
currently: speed 0.5 turn 1.0
Add Duplicate Remove Rename
Views
Type: Orbit (rviz) Zero
Current V... Orbit (rviz)
Near Cl... 0.01
Invert ...
Target ... <Fixed Frame>
Distance 4.03379
Focal S... 0.05
Focal S...
Yaw 0.785398
Pitch 0.790398
Field o... 0.785398
Focal P... 0; 0; 0
Save Remove Rename

功能介绍


/home/tianbot/tianbot_mini_ws/src/abc_swarm/launch/path_tracking_odom.launch http://loc...
* /tianbot_mini/path_planner/plan_topic: global_plan
* /tianbot_mini/path_planner/robot_name: tianbot_mini
* /tianbot_mini/path_tracker/plan_topic: global_plan
* /tianbot_mini/path_tracker/robot_name: tianbot_mini
NODES
/
map_to_odom (tf/static_transform_publisher)
world_to_map (tf/static_transform_publisher)
/tianbot_mini/
map_server (map_server/map_server)
path_planner (abc_swarm/path_planning.py)
path_tracker (abc_swarm/path_tracking.py)
ROS_MASTER_URI=http://localhost:11311
process[world_to_map-1]: started with pid [47268]
process[map_to_odom-2]: started with pid [47269]
process[tianbot_mini/map_server-3]: started with pid [47270]
process[tianbot_mini/path_planner-4]: started with pid [47276]
process[tianbot_mini/path_tracker-5]: started with pid [47282]
[INFO] [1689935927.960919, 186.730000]: TF is ready
[INFO] [1689935941.109897, 191.876000]: Add way point successfully!
tracker.rviz* - RViz
File Panels Help
Interact Move Camera Select Focus Camera Measure 2D Pose Estimate 2D Nav Goal Publish Point
Time
Pause Synchronization: Off ROS Time: 192.50 ROS Elapsed: 23.74 Wall Time: 1942.70 Wall Elapsed: 57.18
Reset Left-Click: Rotate. Middle-Click: Move X/Y. Right-Click: Zoom. Shift: More options.
31 fps

功能介绍

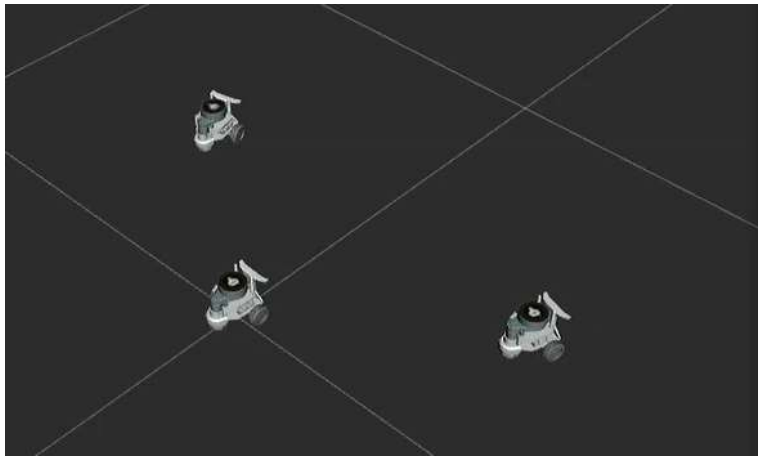

demo_tf.rviz* - RViz
File Panels Help
Interact
Move Camera
Select
Focus Camera
Measure
2D Pose Estimate
2D Nav Goal
Publish Point
空地协同机器人集群与可视化
Take a Photo

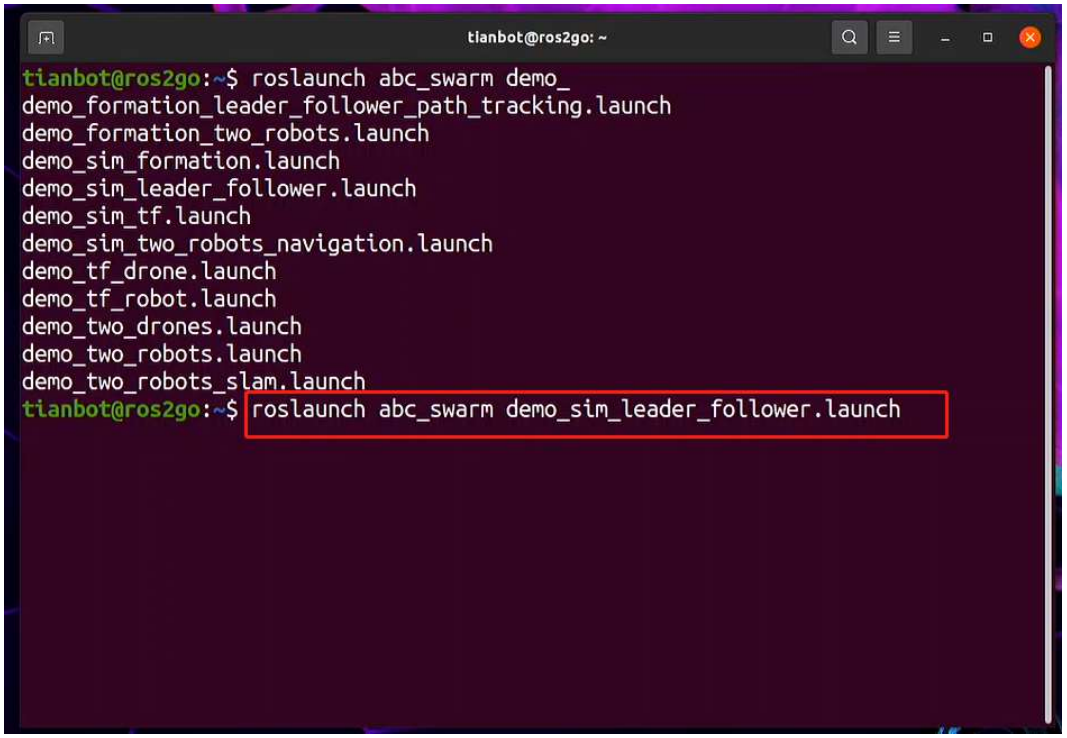
实例分析

leader follower 领航跟随

1

实例分析：领航跟随





运行领航跟随仿真示例：
roslaunch demo_sim_leader_follower.launch

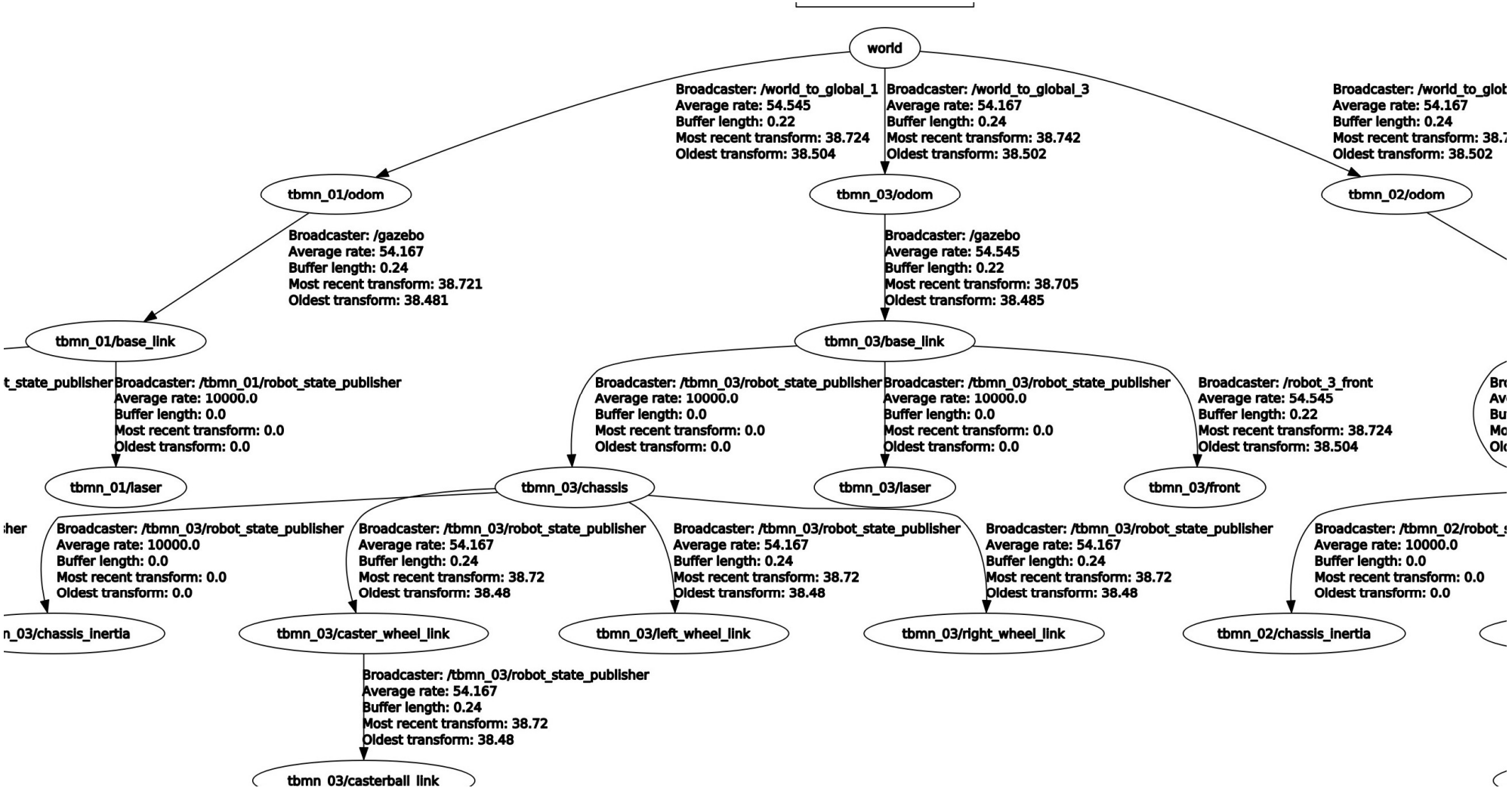
实例分析：领航跟随

分析TF树

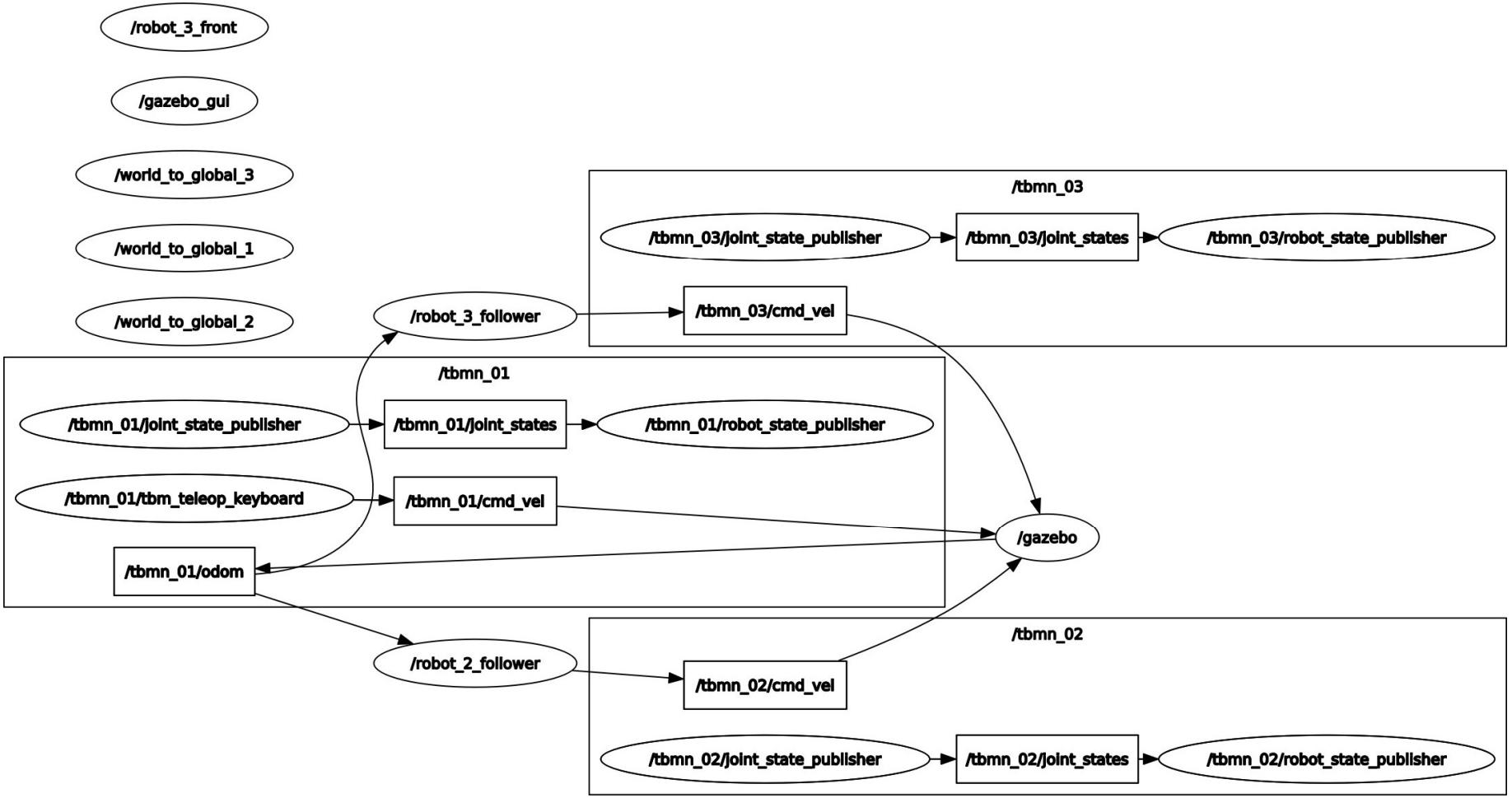
分析节点关系

代码解析

实例分析：领航跟随


world
Broadcaster: /world_to_global_1
Average rate: 54.545
Buffer length: 0.22
Most recent transform: 38.724
Oldest transform: 38.504
Broadcaster: /world_to_global_3
Average rate: 54.167
Buffer length: 0.24
Most recent transform: 38.742
Oldest transform: 38.502
Broadcaster: /world_to_glob
Average rate: 54.167
Buffer length: 0.24
Most recent transform: 38.7
Oldest transform: 38.502
tbmn_01/odom
tbmn_03/odom
tbmn_02/odom
Broadcaster: /gazebo
Average rate: 54.167
Buffer length: 0.24
Most recent transform: 38.721
Oldest transform: 38.481
Broadcaster: /gazebo
Average rate: 54.545
Buffer length: 0.22
Most recent transform: 38.705
Oldest transform: 38.485
tbmn_01/base_link
tbmn_03/base_link
t_state_publisher
Broadcaster: /tbmn_01/robot_state_publisher
Average rate: 10000.0
Buffer length: 0.0
Most recent transform: 0.0
Oldest transform: 0.0
Broadcaster: /tbmn_03/robot_state_publisher
Average rate: 10000.0
Buffer length: 0.0
Most recent transform: 0.0
Oldest transform: 0.0
Broadcaster: /tbmn_03/robot_state_publisher
Average rate: 10000.0
Buffer length: 0.0
Most recent transform: 0.0
Oldest transform: 0.0
Broadcaster: /robot_3_front
Average rate: 54.545
Buffer length: 0.22
Most recent transform: 38.724
Oldest transform: 38.504
tbmn_01/laser
tbmn_03/chassis
tbmn_03/laser
tbmn_03/front
her
Broadcaster: /tbmn_03/robot_state_publisher
Average rate: 10000.0
Buffer length: 0.0
Most recent transform: 0.0
Oldest transform: 0.0
Broadcaster: /tbmn_03/robot_state_publisher
Average rate: 54.167
Buffer length: 0.24
Most recent transform: 38.72
Oldest transform: 38.48
Broadcaster: /tbmn_03/robot_state_publisher
Average rate: 54.167
Buffer length: 0.24
Most recent transform: 38.72
Oldest transform: 38.48
Broadcaster: /tbmn_03/robot_state_publisher
Average rate: 54.167
Buffer length: 0.24
Most recent transform: 38.72
Oldest transform: 38.48
Broadcaster: /tbmn_02/robot_s
Average rate: 10000.0
Buffer length: 0.0
Most recent transform: 0.0
Oldest transform: 0.0
n_03/chassis_inertia
tbmn_03/caster_wheel_link
tbmn_03/left_wheel_link
tbmn_03/right_wheel_link
tbmn_02/chassis_inertia
Broadcaster: /tbmn_03/robot_state_publisher
Average rate: 54.167
Buffer length: 0.24
Most recent transform: 38.72
Oldest transform: 38.48
tbmn_03/casterball_link

实例分析：领航跟随


/robot_3_front
/gazebo_gui
/world_to_global_3
/world_to_global_1
/world_to_global_2
/tbmn_03
/tbmn_03/joint_state_publisher
/tbmn_03/joint_states
/tbmn_03/robot_state_publisher
/robot_3_follower
/tbmn_03/cmd_vel
/tbmn_01
/tbmn_01/joint_state_publisher
/tbmn_01/joint_states
/tbmn_01/robot_state_publisher
/tbmn_01/tbm_teleop_keyboard
/tbmn_01/cmd_vel
/gazebo
/tbmn_01/odom
/tbmn_02
/robot_2_follower
/tbmn_02/cmd_vel
/tbmn_02/joint_state_publisher
/tbmn_02/joint_states
/tbmn_02/robot_state_publisher

实例分析：领航跟随

```
<launch>
  <arg name="robot_name_1" default="tbmn_01" />
  <arg name="robot_name_2" default="tbmn_02" />
  <arg name="robot_name_3" default="tbmn_03" />
  <arg name="front_distance" default = "0.1" />
  <param name="use_sim_time" value="true"/>

  <include file="$(find tianbot_mini)/launch/simulation.launch" >
    <arg name="robot_name" value="$(arg robot_name_1)" />
    <arg name="world_name" value="$(find gazebo_ros)/launch/empty_world.launch" />
  </include>
  <include file="$(find abc_swarm)/launch/spawn_robot.launch">
    <arg name="robot_name" value="$(arg robot_name_2)" />
    <!--the position is controlled by the static transform-->
    <arg name="initial_pose_x" value="-0.732"/>
    <arg name="initial_pose_y" value="0.5"/>
  </include>
  <include file="$(find abc_swarm)/launch/spawn_robot.launch">
    <arg name="robot_name" value="$(arg robot_name_3)" />
    <!--the position is controlled by the static transform-->
    <arg name="initial_pose_x" value="-0.732"/>
    <arg name="initial_pose_y" value="-0.5"/>
  </include>
```

实例分析：领航跟随

```
  <node pkg="tf" type="static_transform_publisher" name="world_to_global_1" args="0 0 0 0 0 0
world $(arg robot_name_1)/odom 20" />

  <node pkg="tf" type="static_transform_publisher" name="world_to_global_2" args="0 0 0 0 0 0
world $(arg robot_name_2)/odom 20" />

  <node pkg="tf" type="static_transform_publisher" name="world_to_global_3" args="0 0 0 0 0 0
world $(arg robot_name_3)/odom 20" />


  <node pkg="tf" type="static_transform_publisher" name="robot_2_front" args="$(arg
front_distance) 0 0 0 0 0 $(arg robot_name_2)/base_link $(arg robot_name_2)/front 20" />

  <node pkg="tf" type="static_transform_publisher" name="robot_3_front" args="$(arg
front_distance) 0 0 0 0 0 $(arg robot_name_3)/base_link $(arg robot_name_3)/front 20" />

  <!-- RVIZ可视化调试工具 -->
  <node name="rviz" pkg="rviz" type="rviz" args="-d $(find abc_swarm)/rviz/demo_formation.rviz"
/>
</launch>
```

实例分析：领航跟随

```
<launch>
  <arg name="robot_name_1" default="tbmn_01" />
  <arg name="robot_name_2" default="tbmn_02" />
  <arg name="robot_name_3" default="tbmn_03" />
  <arg name="front_distance" default = "0.1" />
  <param name="use_sim_time" value="true"/>

  <node pkg="abc_swarm" type="leader_follower.py" name="robot_2_follower" output="screen">
    <param name="leader_robot_name" value="$(arg robot_name_1)" />
    <param name="follower_robot_name" value="$(arg robot_name_2)" />
    <param name="expected_distance" value="0.5" />
    <param name="expected_theta" value="$(eval 3.14159 * 2 /3)" />
    <param name="front_distance" value="$(arg front_distance)" />
  </node>

  <node pkg="abc_swarm" type="leader_follower.py" name="robot_3_follower" output="screen">
    <param name="leader_robot_name" value="$(arg robot_name_1)" />
    <param name="follower_robot_name" value="$(arg robot_name_3)" />
    <param name="expected_distance" value="0.5" />
    <param name="expected_theta" value="$(eval -3.14159 * 2 /3)" />
    <param name="front_distance" value="$(arg front_distance)" />
  </node>
```

教学练习

绘制一条路径，机器人跟踪路径移动

1

教学练习：路径跟踪

在rviz中绘制一个路径(点拟合曲线)，机器人跟踪路径（PID）

挑战：移植到真实的机器人上面

教学练习：路径跟踪

需要用到的源代码(abc_swarm)：

path_tracker.launch 组织下面两个程序

path_planning.py　　路径生成

path_tracking.py　　路径跟踪

教学练习：路径跟踪

它们的输入和输出是什么？

path_planning.py　　路径生成

path_tracking.py　　路径跟踪

教学练习：路径跟踪

path_tracker.launch(abc_swarm)：

```
<launch>
  <arg name="robot_name" default="tianbot_mini" />
  <arg name="plan_topic" default="global_plan" />
  <group ns="$(arg robot_name)">
    <node pkg="abc_swarm" type="path_planning.py" name="path_planner" output="screen">
      <param name="robot_name" value="$(arg robot_name)" />
      <param name="plan_topic" value="$(arg plan_topic)" />
    </node>

    <node pkg="abc_swarm" type="path_tracking.py" name="path_tracker" output="screen">
      <param name="robot_name" value="$(arg robot_name)" />
      <param name="plan_topic" value="$(arg plan_topic)" />
    </node>
  </group>
</launch>
```

教学练习：路径跟踪

路径生成：path_planning.py

使用了三次样条插值算法生成平滑的路径，并将路径发布为ROS消息。

输入：
ROS话题：路径点是通过ROS中的/clicked_point话题发布的geometry_msgs/PointStamped类型消息。路径点消息中包含目标点的三维坐标信息。

输出：
ROS话题：生成的平滑路径是通过ROS中的global_plan话题发布的nav_msgs/Path类型消息。路径消息中包含一系列平滑的位姿点，用于描述路径。

教学练习：路径跟踪

ROS中的/clicked_point话题发布的geometry_msgs/PointStamped类型消息。

RVIZ中可以通过PublishPoint插件来手动发布这个消息。(只能点到map上面)

教学练习：路径跟踪

路径生成：path_planning.py

```
class PathPlanning():
    def wpts_cb(self, msg):
        # 处理收到的路径点
        ...

    def cubic_spline(self):
        # 三次样条插值生成平滑路径
        ...

    def __init__(self):
        # 初始化ROS节点、订阅和发布话题等
        ...
```

教学练习：路径跟踪

路径跟踪：path_tracking.py

使用纯追踪算法实现机器人跟随给定路径运动的控制。

输入：
ROS话题：路径信息是通过ROS中的/global_plan话题发布的nav_msgs/Path类型消息。路径消息中包含一系列位姿点，描述了要追踪的路径。

输出：
ROS话题：机器人的速度指令是通过ROS中的/cmd_vel话题发布的geometry_msgs/Twist类型消息。速度指令包含线性速度和角度速度的信息，用于控制机器人的运动。

教学练习：路径跟踪

路径跟踪：path_tracking.py

```
class PathTracking():
    def path_cb(self, path):
        # 处理收到的路径消息
        ...

    def pure_pursuit(self):
        # 纯追踪算法实现路径追踪
        ...

    def __init__(self):
        # 初始化ROS节点、订阅和发布话题等
        ...
```

教学练习：路径跟踪

通过ROS是如何联系起来的？

RVIZ 手动给点->/click_point -> path_planning

-> gloal_plan -> path_tracking -> /cmd_vel

-> ROS机器人动起来

教学练习：路径跟踪

启动无人车仿真环境:

Roslaunch tianbot_mini simulation.launch

运行path_tracking.launch

教学练习：路径跟踪

为什么无法正常工作？

教学练习：路径跟踪

为什么无法正常工作？

1.RVIZ给点需要一个地图

2.忽略了消息中隐含的TF坐标系

教学练习：路径跟踪

1.RVIZ给点需要一个地图

map_server load map.yaml

2.忽略了消息中隐含的TF坐标系

/world -> /odom

教学练习：路径跟踪

使用gmapping建立一张地图，在仿真环境中：

roslaunch tianbot_mini simulation.launch

roslaunch tianbot_mini slam.launch

roslaunch tianbot_mini map_save.launch

教学练习：路径跟踪

my_path_tracker.launch(abc_swarm)：

```
<launch>
  <arg name="robot_name" default="tianbot_mini" />
  <arg name="plan_topic" default="global_plan" />

    <!-- map file -->
  <arg name="map" default="$(find tianbot_mini)/maps/map.yaml" />

  <node pkg="tf" type="static_transform_publisher" name="world_to_map" args="0 0 0 0 0 0 world $(arg
robot_name)/map 20" />
  <node pkg="tf" type="static_transform_publisher" name="map_to_odom" args="0 0 0 0 0 0 $(arg robot_name)/map
$(arg robot_name)/odom 20" />

  <group ns="$(arg robot_name)">

      <!-- Map服务器，加载*.yaml地图 -->
    <node pkg="map_server" type="map_server" name="map_server" args="$(arg map)">
      <param name="frame_id" value="$(arg robot_name)/map" />
    </node>
    ...
  </group>
</launch>
```

教学练习：路径跟踪

my_path_tracker.launch(abc_swarm)：

```
<launch>
  <arg name="robot_name" default="tianbot_mini" />
  <arg name="plan_topic" default="global_plan" />

    <!-- map file -->
  <arg name="map" default="$(find tianbot_mini)/maps/map.yaml" />

  <node pkg="tf" type="static_transform_publisher" name="world_to_map" args="0 0 0 0 0 0 world $(arg robot_name)/map 20" />
  <node pkg="tf" type="static_transform_publisher" name="map_to_odom" args="0 0 0 0 0 0 $(arg robot_name)/map $(arg robot_name)/odom 20" />

  <group ns="$(arg robot_name)">

      <!-- Map服务器，加载*.yaml地图 -->
    <node pkg="map_server" type="map_server" name="map_server" args="$(arg map)">
      <param name="frame_id" value="$(arg robot_name)/map" />
    </node>

    <node pkg="abc_swarm" type="path_planning.py" name="path_planner" output="screen">
      <param name="robot_name" value="$(arg robot_name)" />
      <param name="plan_topic" value="$(arg plan_topic)" />
    </node>

    <node pkg="abc_swarm" type="path_tracking.py" name="path_tracker" output="screen">
      <param name="robot_name" value="$(arg robot_name)" />
      <param name="plan_topic" value="$(arg plan_topic)" />
    </node>
  </group>
</launch>
```

教学练习：路径跟踪

通过ROS的launch文件进行配置和启动。

<arg> 标签定义了两个参数，robot_name 和 plan_topic，用于设置机器人名称和路径规划的话题名称。
可以通过在launch文件中设置这些参数来自定义机器人和话题名称。

<node> 标签使用了 tf 软件包中的 static_transform_publisher 程序，用于定义静态的坐标系变换关系。

这里定义了两个变换关系：
world_to_map 将世界坐标系 (world) 转换到地图坐标系 ($(arg robot_name)/map)。
map_to_odom 将地图坐标系 ($(arg robot_name)/map) 转换到机器人的里程计坐标系 ($(arg robot_name)/odom)。
这些坐标系变换关系是为了确保地图和机器人坐标系之间的一致性。
<group> 标签定义了一个ROS节点组，命名空间为 $(arg robot_name)，将下面的节点放在了同一个命名空间下。

<node> 标签使用了 map_server 软件包的 map_server 程序，用于加载地图文件。

pkg="map_server" 指定了程序所在的软件包为 map_server。
type="map_server" 指定了程序的类型为 map_server。
name="map_server" 指定了节点的名称为 map_server。
args="$(arg map)" 通过参数 map 指定了地图文件的路径。
<node> 标签使用了轨迹生成程序 path_planning.py，用于生成路径点

教学练习：路径跟踪

pkg="abc_swarm" 指定了程序所在的软件包为 abc_swarm。
type="path_planning.py" 指定了程序的类型为 path_planning.py。
name="path_planner" 指定了节点的名称为 path_planner。
output="screen" 指定了将节点的输出打印到屏幕上。
param name="robot_name" value="$(arg robot_name)"
设置了参数 robot_name 的值为 $(arg robot_name)，即上述定义的机器人名称参数。
param name="plan_topic" value="$(arg plan_topic)"
设置了参数 plan_topic 的值为 $(arg plan_topic)，即上述定义的路径规划话题名称参数。
<node> 标签使用了轨迹跟踪程序 path_tracking.py，用于实现路径追踪控制。

pkg="abc_swarm" 指定了程序所在的软件包为 abc_swarm。
type="path_tracking.py" 指定了程序的类型为 path_tracking.py。
name="path_tracker" 指定了节点的名称为 path_tracker。
output="screen" 指定了将节点的输出打印到屏幕上。
param name="robot_name" value="$(arg robot_name)"
设置了参数 robot_name 的值为 $(arg robot_name)，即上述定义的机器人名称参数。
param name="plan_topic" value="$(arg plan_topic)"
设置了参数 plan_topic 的值为 $(arg plan_topic)，即上述定义的路径规划话题名称参数。

通过运行这个launch文件，可以将路径生成和路径跟踪节点一起启动，实现机器人的路径规划和路径追踪功能。
同时，通过定义的参数，可以方便地配置机器人名称和话题名称，以适应不同的场景和要求。

教学练习：路径跟踪

已有代码分析：path_planer, path_tracker
分析运行流程
梳理输入输出
添加TF变换，连接 world -> map -> odom
使用map_server加载地图
运行路径跟踪，指定点，机器人运动

教学练习：路径跟踪

在rviz中绘制一个路径(点拟合曲线)，机器人跟踪路径（PID）

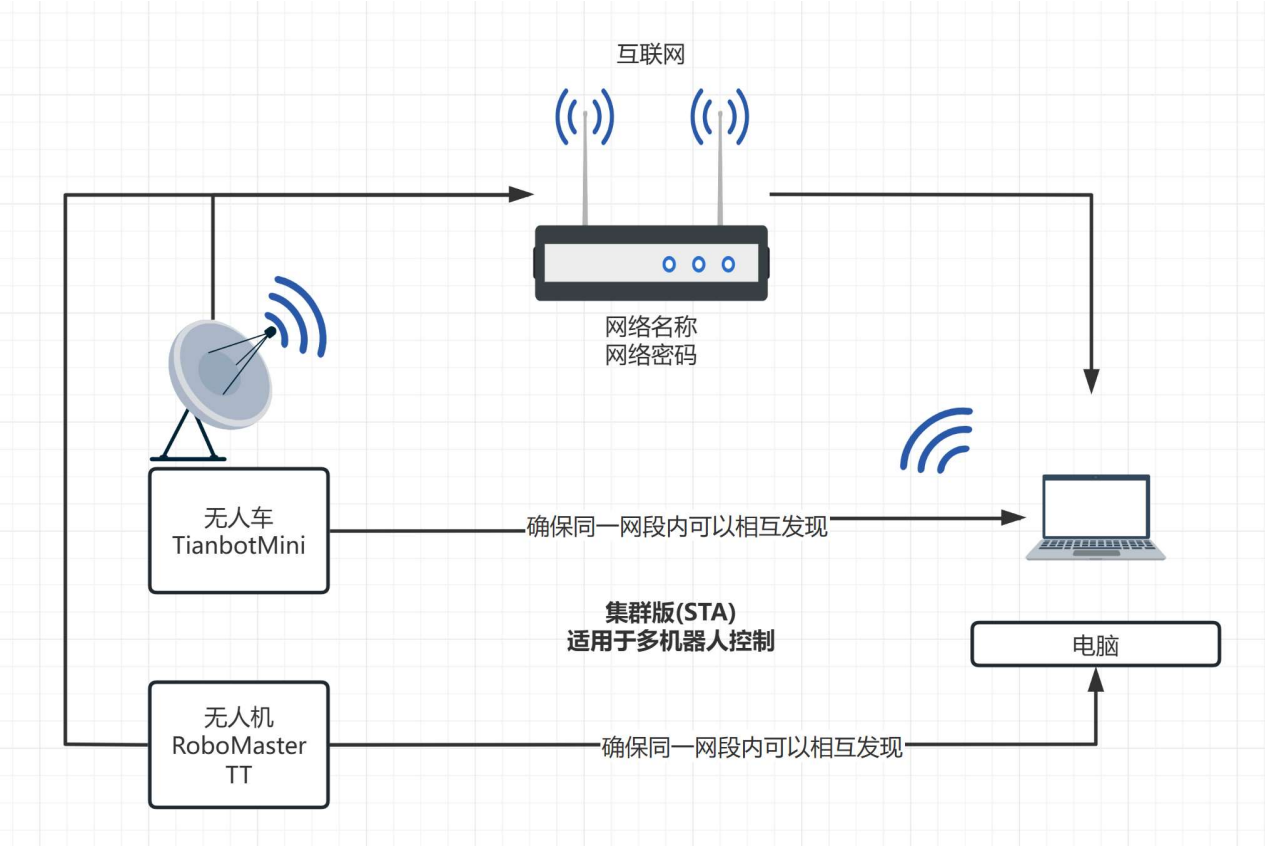
挑战：移植到真实的机器人上面

陆空协同套装网络配置

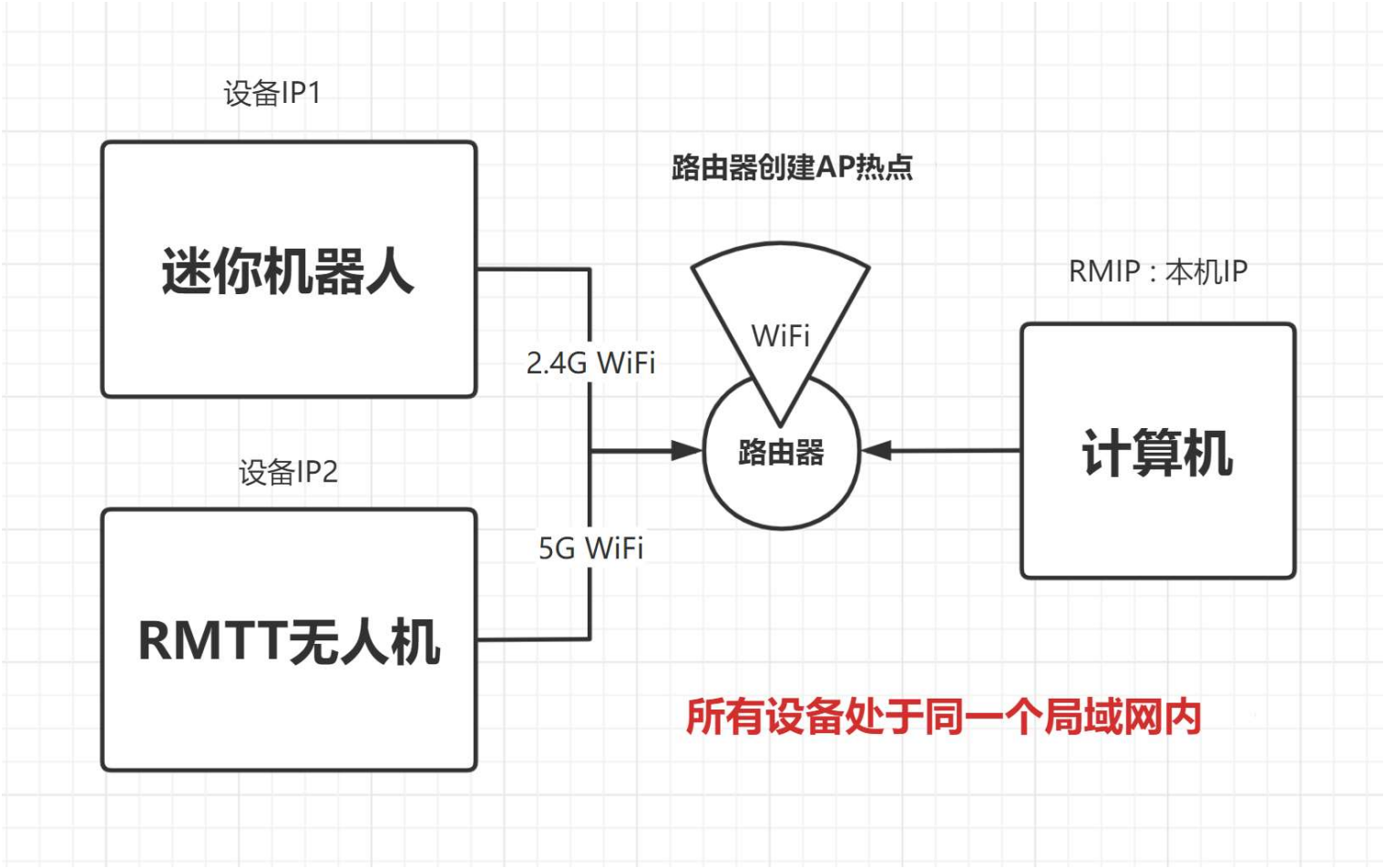
网络拓扑

1

网络拓扑


互联网
网络名称
网络密码
无人车
TianbotMini
确保同一网段内可以相互发现
集群版(STA)
适用于多机器人控制
电脑
无人机
RoboMaster
TT
确保同一网段内可以相互发现

网络架构异构多机


设备IP1
迷你机器人
路由器创建AP热点
WiFi
2.4G WiFi
RMIP：本机IP
计算机
路由器
设备IP2
5G WiFi
RMTT无人机
所有设备处于同一个局域网内

网络拓扑

在ROS中同时看到无人车和无人机的话题。

准备工作路由器

路由器上电，并通过手机或PC确认路由器WiFi包含2.4G和5G网络（下文将以TIANBOT-SWARM-xxxx，TIANBOT-SWARM-5G作为演示网络名称，具体请根据实际路由器网络做出调整）

在PC上启动ROS2GO随身系统，为TianbotMini做好ROS环境准备（说明文档参考：http://doc.tianbot.com/ros2go/1701321）

网络拓扑

网络信息:
TIANBOT-SWARM-5G　(飞机)
TIANBOT-SWARM-909C(小车)
网络密码: www.tianbot.com

配置信息:
无人车名称：tbmn_01
无人机名称：rmtt_01

网络拓扑

TianbotStudio 陆空协同无人车配置

连接无人车到计算机USB接口

打开TianbotStudio，选择串口

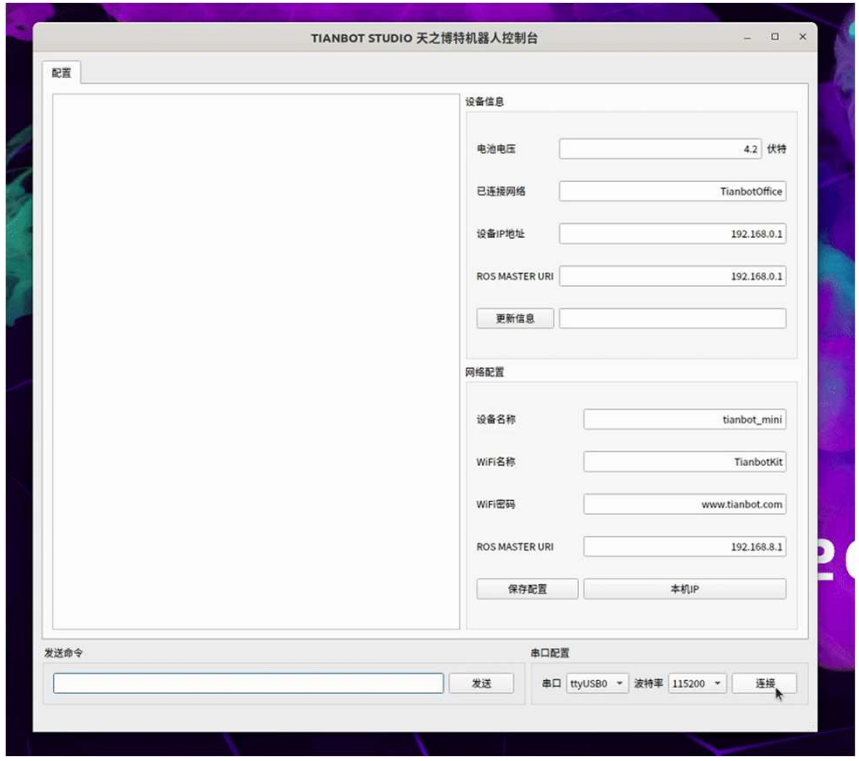
点击连接并更新信息(获取当前配置)

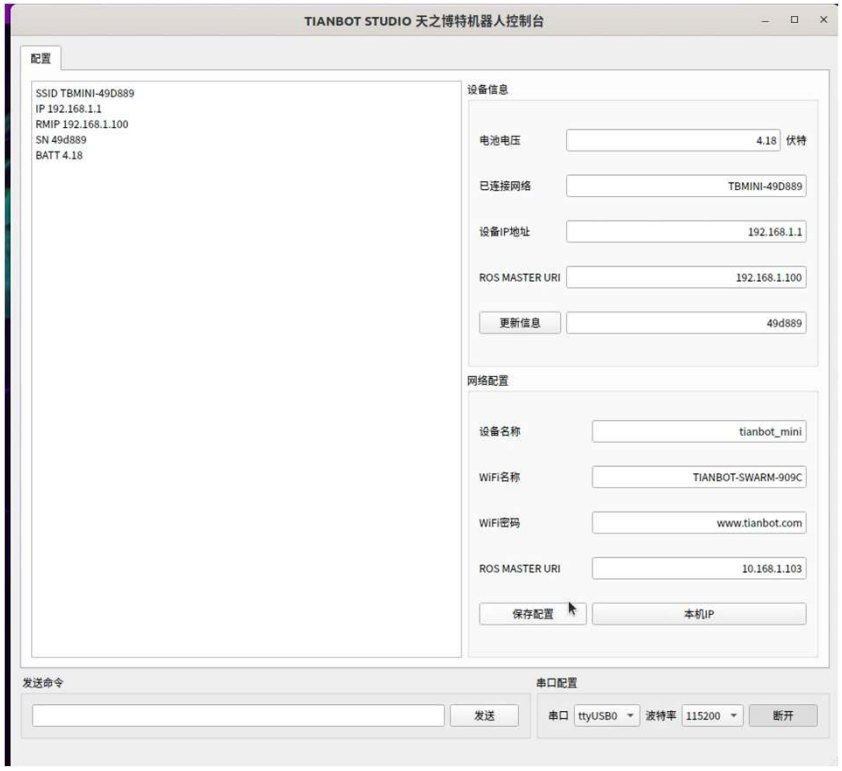
填写设备名称、ROS MASTER URI、网络信息

点击保存配置机器人将开始连接配置的网络(2.4G)

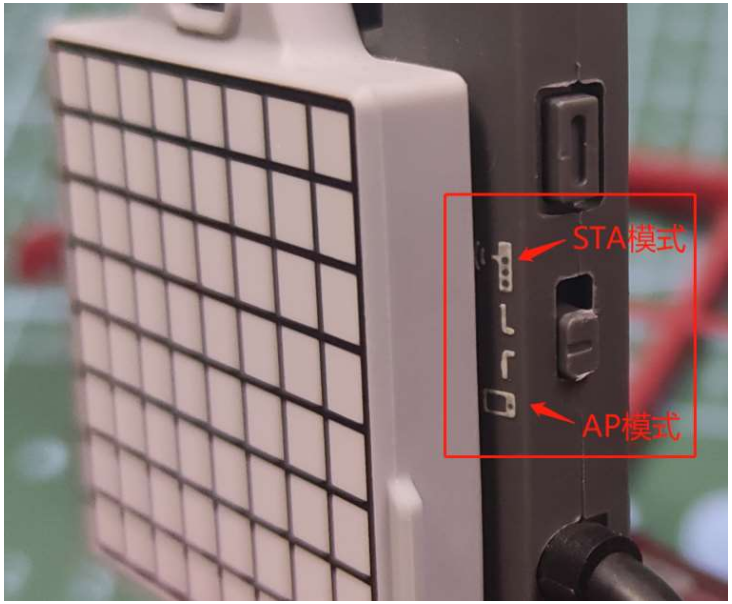
网络拓扑

TianbotStudio 陆空协同无人车配置





RoboMasterTT 陆空协同无人机配置



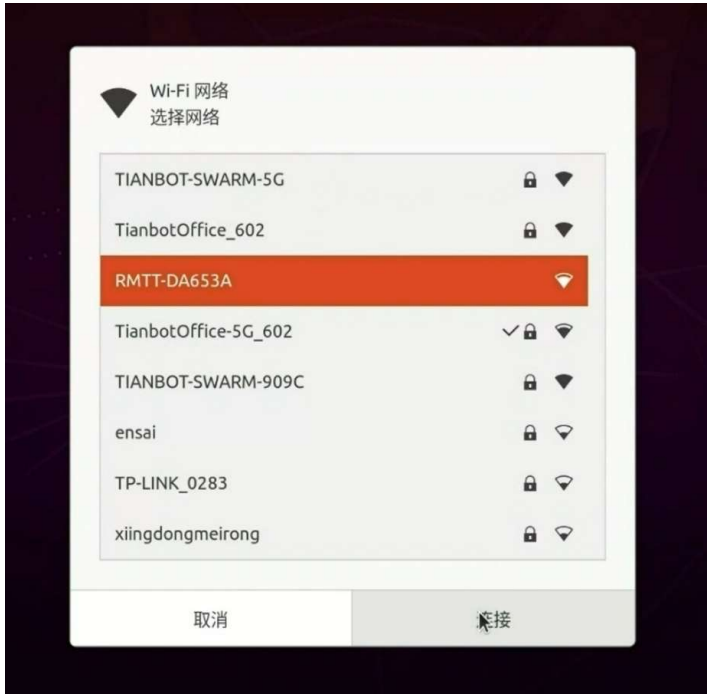
在首次无人机使用之前，我们需要将上面使用ROS2GO启动的电脑连接无人机的热点上即可使用。

首先将RMTT无人机转换至热点模式（开关切换至手机图样），短按电源按钮一下，此时设备进入单机模式（AP热点）。

无人机会创建RMTT-XXXX热点，该热点未加密，我们使用电脑连接该热点，即可使用。

网络拓扑

RoboMasterTT 陆空协同无人机配置



选择以RMTT开头的WIFI名称，点击连接。此时我们打开一个终端，输入命令：

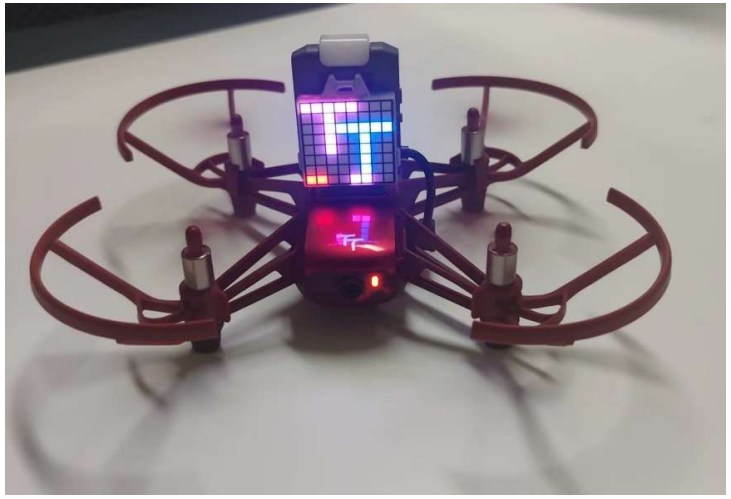
roscd rmtt_driver/scripts

./set_sta.py TIANBOT-SWARM-5G www.tianbot.com

执行之后，我们需要将RMTT无人机的开关拨至上方，切换为路由模式。

网络拓扑

RoboMasterTT 陆空协同无人机配置



如果配置成功的话，我们可以看到无人机桨叶开始旋转，以及指示灯变为闪烁的黄灯。

网络拓扑

RoboMasterTT 陆空协同无人机配置

如果配置成功完成RMTT的网络配置之后，把电脑连接到路由器，我们需要查询RMTT的IP地址，输入命令./rmtt_scan_ip.py。

```
tianbot@ros2go:~/tianbot_ws/src/rmtt_ros/rmtt_driver/scripts$ ./rmtt_scan_ip.py
Searching for one drone...
2023-06-26 16:11:47,371 INFO tool.py:221 [Start_Searching]Searching for 1 available Tello...

2023-06-26 16:11:47,377 INFO tool.py:234 [Still_Searching]Trying to find Tello in subnets...

2023-06-26 16:11:47,872 INFO tool.py:269 FoundTello: from ip ('10.168.1.100', 8889)_receive_task, recv msg: b'ok'
2023-06-26 16:11:47,872 INFO tool.py:272 FoundTello: Found Tello.The Tello ip is:10.168.1.100

2023-06-26 16:11:47,872 INFO tool.py:277 FoundTello: has finished, _scan_receive_task quit!
2023-06-26 16:11:47,872 INFO tool.py:301 recv thread start!
2023-06-26 16:11:50,897 INFO rmtt_scan_ip.py:65 send cmd
2023-06-26 16:11:51,008 INFO rmtt_scan_ip.py:78 get host
2023-06-26 16:11:51,014 INFO tool.py:305 recv thread quit!
scan result: sn:0TQZHBDCNT0CF7 host('10.168.1.100', 8889)
```

输出的scan result一行中，10.168.1.100就是查询到的IP地址。让我们输入命令：

roslaunch rmtt_driver rmtt_bringup.launch drone_ip:=10.168.1.100

网络拓扑

在ROS中查看无人车与无人机的话题

任务分解

无人车跟随无人机路径

1

任务分解

•预备知识（已经学到） 任务分解

- 驱动无人车(TianbotMini)与无人机(RoboMasterTT)
- 在ROS中同时获取到它们的话题
- 了解机器人位置与坐标系之间的联系
- 了解机器人的位置在ROS中的呈现形式以及获取方法
- 了解一个路径跟踪的例程是如何运行的（ROS层面）

- 驱动无人机与无人车，在同一个网络环境
- 手动对齐无人机与无人车的坐标系
- 获取无人机的位置
- 把无人机的位置以publish point的方式发出
- 调用路径跟踪的例程，实现拟合路径，并控制无人车跟踪

关注我们

公众号：TIANBOT

文字教程、新闻动态

B站:天之博特TIANBOT

视频教程、粉丝福利

天之博特商务洽谈

活动咨询、产品咨询、定制开发