UWB_DR模式通信驱动手册

快速上手

1.目标点发送话题：target_topic，在launch文件修改 默认为/uwbx/target_position，话题类型geometry_msgs/PoseStamped

2.里程计odom发送话题：odomx，在launch文件修改 默认为 /odomx，话题类型默认nav_msgs/Odometry

3.目标点订阅话题：/uwb3/target_matrix，里面为扁平化的矩阵,一行为一个id的数据

```
---
layout:
  dim:
    -
      label: "rows"
      size: 4
      stride: 16
    -
      label: "cols"
      size: 4
      stride: 4
  data_offset: 0
data: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 2.0, 3.0
, 0.0]
---
```

分别为x,y,z，yaw(可选，默认为0)

为n*4矩阵；n个设备

第一行为id0的目标点，第二行为id1的目标点，以此类推

4.里程计pose订阅话题：/uwb3/pose_matrix，里面为扁平化的矩阵，一行为一个id的数据

```
  dim:
    -
      label: "rows"
      size: 4
      stride: 16
    -
      label: "cols"
      size: 4
      stride: 4
  data_offset: 0
data: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1111111.0, 2.
440000057220459, 111.0, 1.5700000524520874]
```

分别为x,y,z，yaw

为n*4矩阵；n个设备

第一行为id0的odom，第二行为id1的odom，以此类推

5.距离矩阵话题：/uwbx/distance_matrix，里面为扁平化的矩阵

发布节点间 UWB 距离矩阵（对称或非对称，单位：米）,与id一致，已进行z轴补偿

6.解析位置发布话题：/uwb3/target_position；geometry_msgs/PoseStamped

7.位置矩阵转发话题：/uwb3/matrix_from_uwb2 /uwb3/matrix_from_uwb1 /uwb3/matrix_from_uwb0

快速启动(仅通信测距+部分信息)

UWB节点频率为100Hz

1. **启动UWB硬件驱动**
   连接UWB模块后执行：

   ```
   roslaunch nlink_parser linktrack_single.launch
   ```

   参数说明：

   node_id：配置为uwbx的id

   port_name：连接UWB的串口

   baud_rate：串口使用的波特率

2. **启动节点**

   ```
   roslaunch distance uwb_distance_single_data.launch
   ```

   参数说明：

   node_id：配置为uwbx的id

   total_nodes：系统节点数量

   required_nodes：启动矩阵发布，最少节点数量

   matrix_print_rate：打印日志频率

   matrix_publish_rate：发布的频率

   distance_diff_threshold：最短滤波距离

   odom_topic：订阅的odom话题

   target_topic：订阅的位置点话题

3. **可视化定位结果（测试）**

   ```
   roslaunch distance MDS_display.launch  # 自动加载预配置的RViz界面
   ```

关键话题

话题名称	消息类型	说明
/odom3	nav_msgs/Odometry	接收本节点的位姿数据（位置+方向），用于更新自身 pose_matrix。
/rosout	rosgraph_msgs/Log	ROS 系统日志输出（标准话题）。
/rosout_agg	rosgraph_msgs/Log	聚合所有节点的日志信息（ROS 内部使用）。
/uwb3/custom_matrix	std_msgs/Float32MultiArray	接收外部 4×2 自定义矩阵数据，更新 `custom_matrix_`。
/uwb3/data_transmission	std_msgs/String	发布本节点封装后的完整数据字符串，供其他节点解析。
/uwb3/distance_matrix	std_msgs/Float32MultiArray	发布节点间 UWB 距离矩阵（对称或非对称，单位：米）。
/uwb3/matrix_from_uwb0	std_msgs/Float32MultiArray	接收并转发来自 UWB 节点 0 的 custom_matrix（4×2）矩阵。
/uwb3/matrix_from_uwb1	std_msgs/Float32MultiArray	接收并转发来自 UWB 节点 1 的 custom_matrix（4×2）矩阵。
/uwb3/matrix_from_uwb2	std_msgs/Float32MultiArray	接收并转发来自 UWB 节点 2 的 custom_matrix（4×2）矩阵。
/uwb3/nodeframe0	nlink_parser/LinktrackNodeframe0	接收其他节点广播的封装数据字符串（包含 pose、目标、距离信息）。
/uwb3/nodeframe2	nlink_parser/LinktrackNodeframe2	接收本节点 UWB 硬件输出的距离数据（目标节点 ID + 距离）。
/uwb3/pose_matrix	std_msgs/Float32MultiArray	发布所有已知节点的位姿（4×4 矩阵：x, y, z, yaw）。
/uwb3/target_matrix	std_msgs/Float32MultiArray	发布所有节点接收到的目标点位置（4×4 矩阵：x, y, z, yaw）。
/uwb3/target_position	geometry_msgs/PoseStamped	接收目标点（goal）位置输入，更新 target_matrix。

测试用例

话题发送实例：

1.发送goal点

```
rostopic pub -r 180 /uwb1/target_position geometry_msgs/PoseStamped "header:
  frame_id: 'map'
  stamp:
    secs: 0
    nsecs: 0
pose:
  position:
    x: 1.0
    y: -0.8
    z: 4.0
  orientation:
    x: 0.0
    y: 0.0
    z: 0.707
    w: 0.707"
```

订阅话题：rostopic echo /uwb3/pose_matrix

2.发送odom点

```
rostopic pub /odom2 nav_msgs/Odometry \
'{header: {stamp: now, frame_id: "map"}, child_frame_id: "base_link", pose:
{pose: {position: {x: 1111111.0, y: 2.44, z: 111}, orientation: {x: 0.0, y: 0.0,
z: 0.707, w: 0.707}}}, twist: {twist: {linear: {x: 0.0, y: 0.0, z: 0.0}, angular:
{x: 0.0, y: 0.0, z: 0.0}}}}'
```

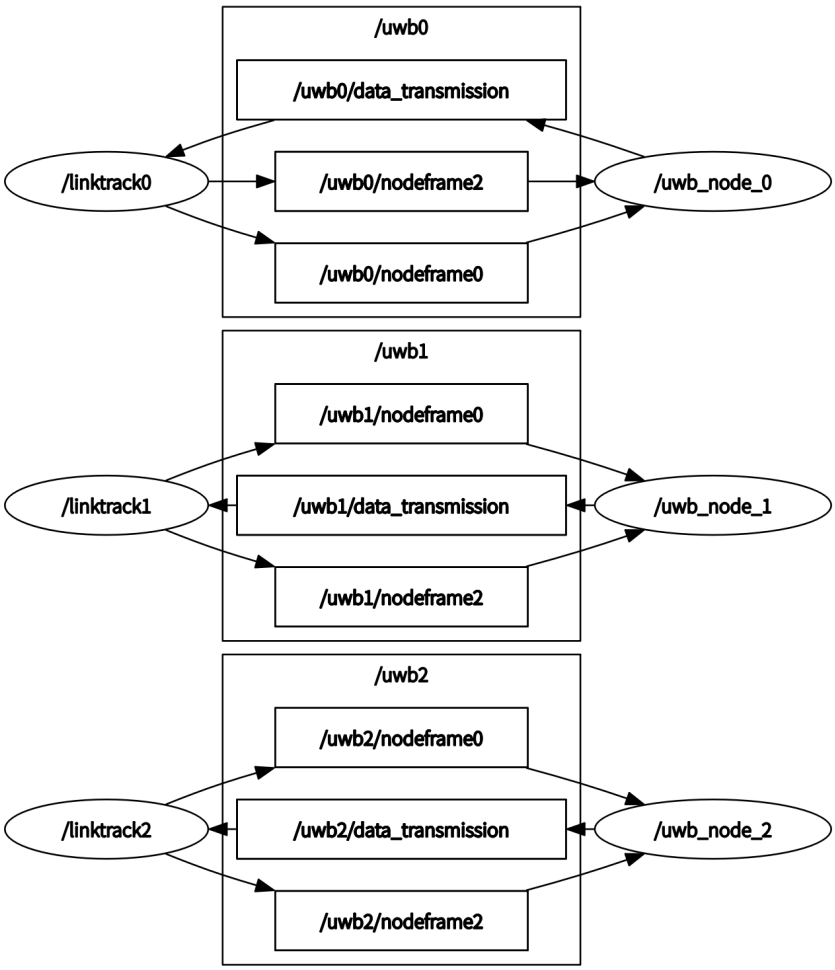
订阅话题：rostopic echo /uwb3/target_matrix

3.发送坐标矩阵

```
rostopic pub /uwb2/custom_matrix std_msgs/Float32MultiArray "layout:
  dim:
    - label: 'rows'
      size: 4
      stride: 8
    - label: 'cols'
      size: 2
      stride: 2
  data_offset: 0
data: [1.1, 1.2, 2.1, 2.2, 3.1, 3.2, 4.1, 422.2]"
```

订阅话题：rostopic echo /uwb2/matrix_from_uwb3

消息机制


/uwb0
/uwb0/data_transmission
/linktrack0
/uwb0/nodeframe2
/uwb_node_0
/uwb0/nodeframe0
/uwb1
/uwb1/nodeframe0
/linktrack1
/uwb1/data_transmission
/uwb_node_1
/uwb1/nodeframe2
/uwb2
/uwb2/nodeframe0
/linktrack2
/uwb2/data_transmission
/uwb_node_2
/uwb2/nodeframe2