无线手柄功能使用与讲解

1. 功能简介

将无线手柄的 usb 适配器接入虚拟机/树莓派后，打开手柄控制节点以及 ROS 端接收手柄数据节点，再通过相关功能包订阅数据话题和发布速度话题，从而实现通过无线手柄控制小海龟以及 ROS 小车运动。

2. 使用方法

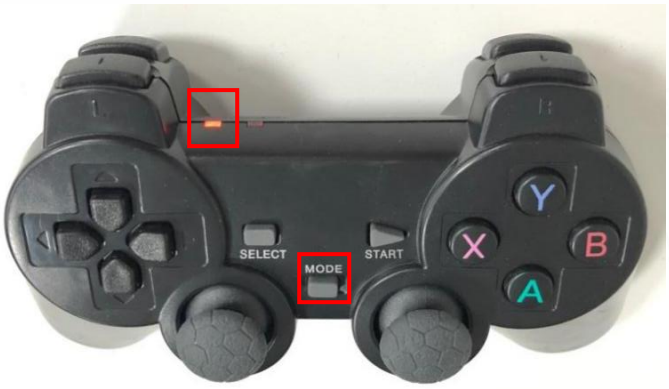
① 安装配置

在 linux 环境下使用 joy 手柄，需要先安装相关驱动。前期准备如：虚拟机操作系统 Ubuntu18.04，ROS 版本为 melodic，无线手柄，使用 WHEELTEC 树莓派版本的阿克曼小车进行测试。手柄及其适配器如图所示。

无线手柄以及 usb 适配器

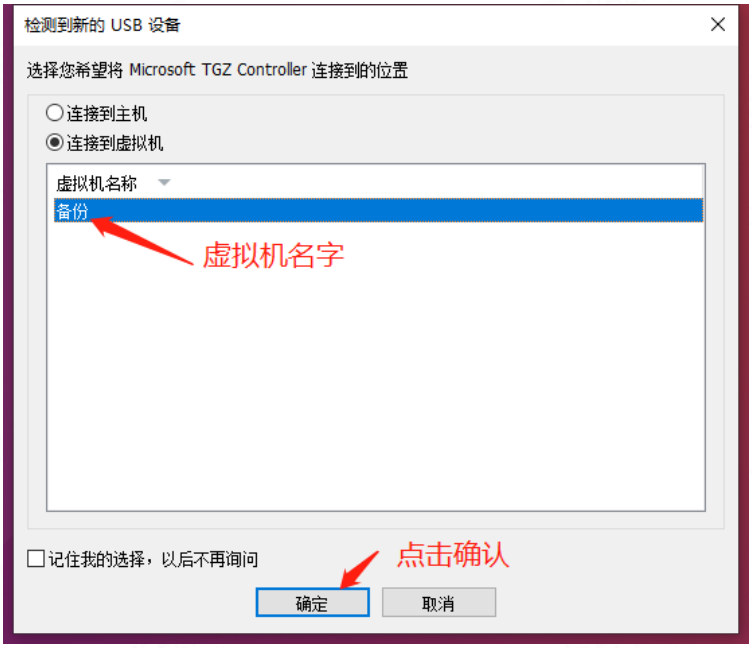
1) 在虚拟机/树莓派中连接手柄：

将手柄的 usb 适配器接入虚拟机/树莓派中，此时手柄会自动与适配器进行连接，无需用户再次配对。配对完成后，手柄的红灯会常亮，若为绿灯，则表示是其它控制模式，需要长按[MODE]按键 5 秒，切换模式。如图所示。需要注意的是，手柄的自动锁定时间为 5 分钟，即如果用户超过 5 分钟未唤醒手柄，则手柄进入睡眠状态，此时长按 5 秒[MODE]按键即可唤醒手柄。



WHEELTEC joy 手柄连接上虚拟机/树莓派时的亮灯状况

首先把手柄插入虚拟机/树莓派，如果用户是使用虚拟机进行测试，需要把 usb 适配器接入电脑主机，会弹出一个交互弹窗，如图所示。



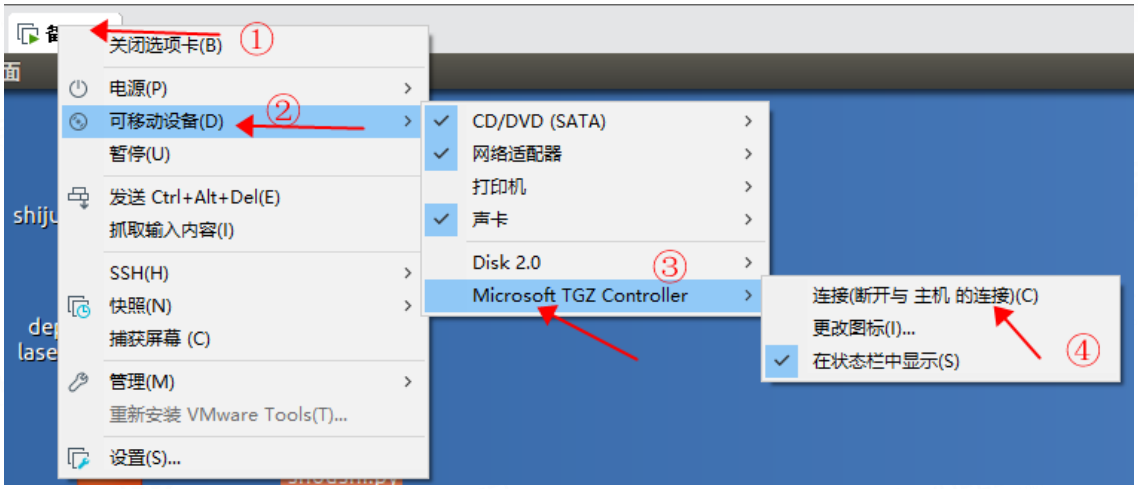
WHEELTEC joy 手柄连接上虚拟机时的交互弹窗

接入之后可以通过检查输入以下命令查看：

```
ls /dev/input
```

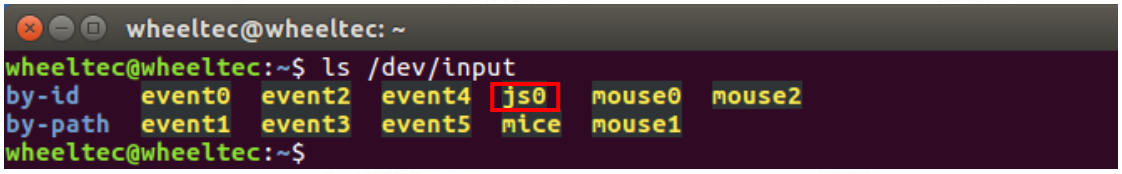
在终端输出中，带有 js#字样的设备一般就是手柄了。若在虚拟机中点击确认后，输入查看设备的命令找不到手柄设备，则需要再确认一遍 usb 适配器是否真正连接至虚拟机中。具体步骤为：

1. 右击虚拟机名字，弹出选项卡；
2. 选择[可移动设备]—>[Microsoft TGZ Controller]—>[连接]。



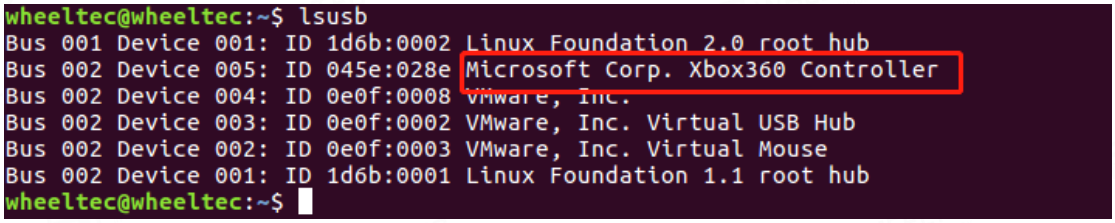
WHEELTEC joy 手柄连接上虚拟机的检查步骤

如图所示，终端显示的“js0”即为无线手柄的接口。注意：有的电脑会出现 js0，js1，此时只需要拔出手柄的 usb 接收器后再看设备，缺少哪个设备就是手柄的接口。一般默认的无线手柄接口是 js0。



WHEELTEC joy 手柄设备号

或使用”lsusb”指令查看虚拟机/树莓派是否正确识别手柄的 usb 适配器。



虚拟机识别到 WHEELTEC joy 手柄

2) 检查无线手柄是否工作：

使用 jstest 命令来检查它是否工作，及测试无线手柄每个键所对应的数字，终端中输入以下命令进行查看：

```
sudo jstest /dev/input/js0

若没有安装 jstest，则运行以下指令：
sudo apt-get install joystick
```

```
wheeltec@wheeltec: ~
wheeltec@wheeltec:~$ sudo jstest /dev/input/js0
Driver version is 2.1.0.
Joystick (Microsoft X-Box 360 pad) has 8 axes (X, Y, Z, Rx, Ry, Rz, Hat0X, Hat0Y)
and 11 buttons (BtnX, BtnY, BtnTL, BtnTR, BtnTR2, BtnSelect, BtnThumbL, BtnThumbR, ?, ?
, ?).
Testing ... (interrupt to exit)
Axes:  0:     0  1:     0  2:     0  3:     0  4:     0  5:     0  6:     0  7:     0 B
Axes:  0:     0  1:     0  2:     0  3:     0  4:     0  5:     0  6:     0  7:     0 B
Axes:  0:     0  1:     0  2:     0  3:     0  4:     0  5:     0  6:     0  7:     0 B
Axes:  0:     0  1:     0  2:     0  3:     0  4:     0  5:     0  6:     0  7:     0 B
Axes:  0:     0  1:     0  2:     0  3:     0  4:     0  5:     0  6:     0  7:     0 B
Axes:  0:     0  1:     0  2:     0  3:     0  4:     0  5:     0  6:     0  7:     0 B
Axes:  0:     0  1:     0  2:     0  3:     0  4:     0  5:     0  6:     0  7:     0 B
Axes:  0:     0  1:     0  2:     0  3:     0  4:     0  5:     0  6:     0  7:     0 B
Axes:  0:     0  1:     0  2:     0  3:     0  4:     0  5:     0  6:     0  7:     0 B
Axes:  0:     0  1:     0  2:     0  3:     0  4:     0  5:     0  6:     0  7:     0 B
Axes:  0:     0  1:     0  2:     0  3:     0  4:     0  5:     0  6:     0  7:     0 B
Axes:  0:     0  1:     0  2:     0  3:     0  4:     0  5:     0  6:     0  7:     0 B
Axes:  0:     0  1:     0  2:     0  3:     0  4:     0  5:     0  6:     0  7:     0 B
Axes:  0:     0  1:     0  2:-32767  3:     0  4:     0  5:     0  6:     0  7:     0 B
Axes:  0:     0  1:     0  2:-32767  3:     0  4:     0  5:     0  6:     0  7:     0 B
Axes:  0:     0  1:     0  2:-32767  3:     0  4:     0  5:     0  6:     0  7:     0 B
Axes:  0:     0  1:     0  2:-32767  3:     0  4:     0  5:-32767  6:     0  7:     0 B
Axes:  0:     0  1:     0  2:-32767  3:     0  4:     0  5:-32767  6:     0  7:     0 B
Axes:  0:     0  1:     0  2:-32767  3:     0  4:     0  5:-32767  6:     0  7:     0 B
uttons:  0:off  1:off  2:off  3:off  4:off  5:off  6:off  7:off  8:off  9:off 10:off
```

WHEELTEC joy 手柄工作状态

结果如图所示，该无线手柄有 8 个轴向输入，11 个按键输入。通过操控手柄，可以看到数据源源不断地进行更新，分别按动按键来测试其所对应的数字。

3) 无线手柄驱动安装：

将手柄接入虚拟机/树莓派中之后，需要安装 ros 的 joystick_drivers 驱动功能包。该功能包包含了使用手柄操作机器人所需的所有节点和驱动程序，其主要作用是将手柄输入状态转换为 ROS 消息。该功能包通过以下指令进行安装：

```
sudo apt-get install ros-melodic-joystick-drivers
```

驱动功能包安装完成后，将【wheeltec_joy_control】这个功能包放入工作空间，使用”catkin_make”命令进行编译。编译完成后如下图所示。完成编译之后即可测试手柄数据信息。

```
wheeltec@wheeltec: ~/catkin_ws
Scanning dependencies of target rosgraph_msgs_generate_messages_nodejs
[  0%] Built target rosgraph_msgs_generate_messages_nodejs
Scanning dependencies of target roscpp_generate_messages_cpp
[  0%] Built target roscpp_generate_messages_cpp
Scanning dependencies of target rosgraph_msgs_generate_messages_cpp
[  0%] Built target rosgraph_msgs_generate_messages_cpp
Scanning dependencies of target std_msgs_generate_messages_cpp
[  0%] Built target std_msgs_generate_messages_cpp
Scanning dependencies of target roscpp_generate_messages_eus
[  0%] Built target roscpp_generate_messages_eus
Scanning dependencies of target rosgraph_msgs_generate_messages_py
[  0%] Built target rosgraph_msgs_generate_messages_py
Scanning dependencies of target rosgraph_msgs_generate_messages_lisp
[  0%] Built target rosgraph_msgs_generate_messages_lisp
Scanning dependencies of target roscpp_generate_messages_nodejs
[  0%] Built target roscpp_generate_messages_nodejs
Scanning dependencies of target std_msgs_generate_messages_nodejs
[  0%] Built target std_msgs_generate_messages_nodejs
Scanning dependencies of target roscpp_generate_messages_lisp
[  0%] Built target roscpp_generate_messages_lisp
Scanning dependencies of target std_msgs_generate_messages_eus
[  0%] Built target std_msgs_generate_messages_eus
Scanning dependencies of target joycontrol
[ 50%] Building CXX object wheeltec_joy_control/CMakeFiles/joycontrol.dir/src/
control.cpp.o
[100%] Linking CXX executable /home/wheeltec/catkin_ws/devel/lib/joy_control/j
oycontrol
[100%] Built target joycontrol
wheeltec@wheeltec:~/catkin_ws$
```

编译功能包

在启动 joy_node 节点之前要先设定一下这个节点所读取的手柄设备

```
rosparam set joy_node/dev "/dev/input/js0"
```

打开第二个终端，启动 joy_node 节点

```
roslaunch wheeltec_joy joy_data.launch
```

此时终端输出信息如下图所示，打开读取手柄数据的节点时，会出现一个警告，这个警告不影响使用，是可以忽略的。

```
wheeltec@wheeltec:~$ roslaunch wheeltec_joy joy_data.launch
... logging to /home/wheeltec/.ros/log/f2da04fe-0163-11ec-82fa-00044be71c46/roslaunch-wheel
tec-8537.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://192.168.0.100:46653/

SUMMARY
========

PARAMETERS
 * /rosdistro: melodic
 * /rosversion: 1.14.10
 * /wheeltec_joy/deadzone: 0.12
 * /wheeltec_joy/dev: /dev/input/js0

NODES
  /
    wheeltec_joy (wheeltec_joy/joy_node)

auto-starting new master
process[master]: started with pid [8548]
ROS_MASTER_URI=http://192.168.0.100:11311

setting /run_id to f2da04fe-0163-11ec-82fa-00044be71c46
process[rosout-1]: started with pid [8560]
started core service [/rosout]
process[wheeltec_joy-2]: started with pid [8566]
[ WARN] [1629428937.011852860]: Couldn't set gain on joystick force feedback: Bad file desc
riptor
```

启动 joy_node 节点的终端信息

测试用手柄向 ROS 端发送数据,可以通过如下命令来查看节点发布的/joy 话题数据：

```
^Cwheeltec@wheeltec:~$ rostopic echo /joy
header:
  seq: 154
  stamp:
    secs: 1629429091
    nsecs: 943382442
  frame_id: "/dev/input/js0"
axes: [0.0, 0.07927737385034561, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
buttons: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
---
header:
  seq: 155
  stamp:
    secs: 1629429091
    nsecs: 955480855
  frame_id: "/dev/input/js0"
axes: [0.0, 0.240956112742424, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
buttons: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
---
header:
  seq: 156
  stamp:
    secs: 1629429091
    nsecs: 963421307
```

查看 joy 话题数据

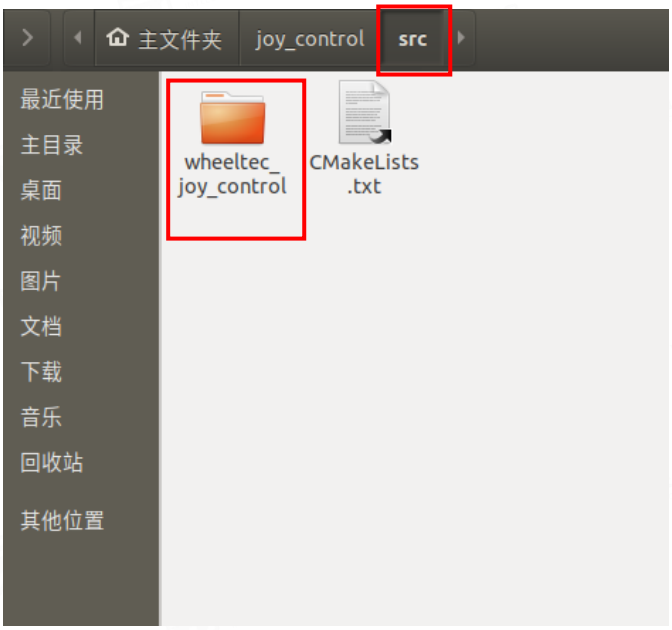
按下手柄的任一按键，若显示如上图所示的数据，则表示树莓派/虚拟机能成功收到手柄发送的消息。在应用中只要节点订阅该话题即可以通过手柄来向该节点发布命令。主要用到的手柄按键与/joy 话题中对应的变量关系如下图所示。

手柄控制主要按键

② 通过无线手柄在虚拟机端控制小海龟运动

1) 建立并编译工作空间：

安装手柄驱动功能包后，建立工作空间（如图建立了名为“joy_control”的工作空间），并把虚拟机端手柄控制小海龟运动的功能包【wheeltec_joy_control】放入工作空间目录下的 src 文件内。在工作空间目录下使用”catkin_make”命令进行编译。如下图所示：



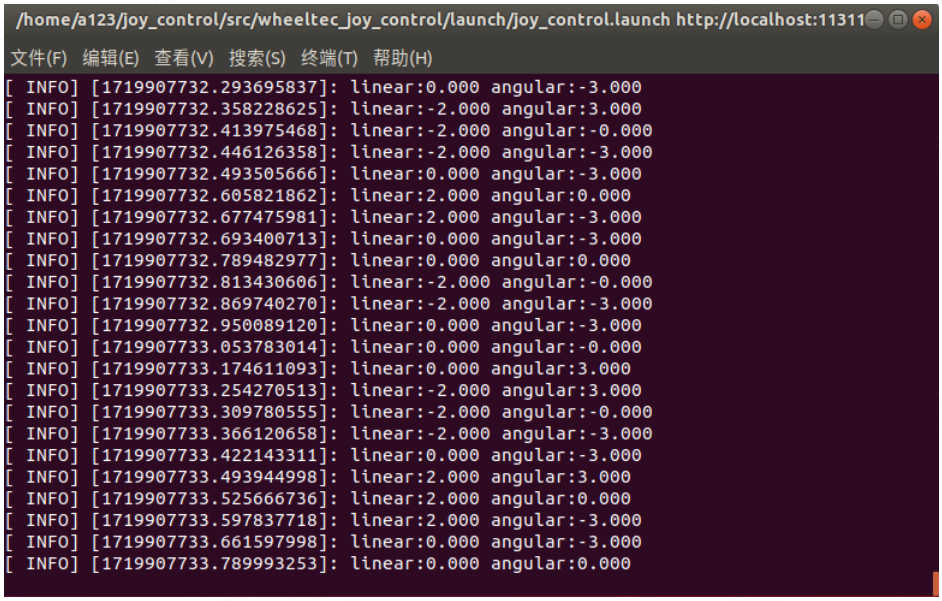
把功能包放入 src 目录下

```
a123@a123-virtual-machine:~/joy_control$ catkin_make
```

编译工作空间

2) 在虚拟机端运行 launch 文件启动手柄控制小海龟运动的功能，然后使用手柄控制小海龟运动，终端会一直更新消息。

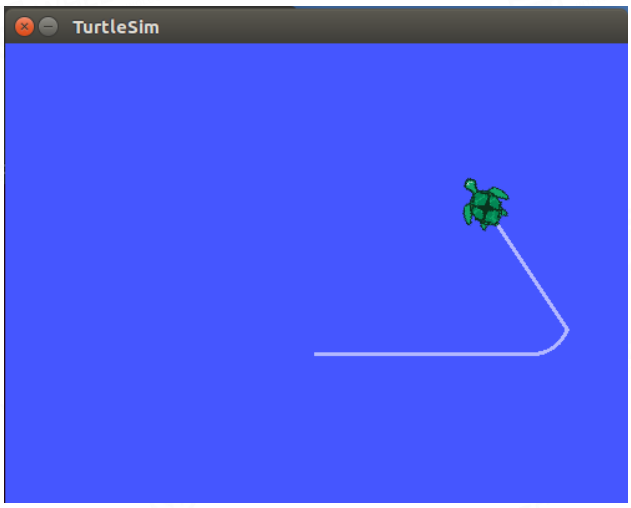
```
roslaunch wheeltec_joy joy_control.launch
```



手柄控制小海龟运动

控制操作为：左边摇杆的前后左右四个方向控制小车的前进和后退，以及左右转向，前后推动右边的摇杆实现小海龟的前进和后退的加减速，左右推动右摇杆实现小海龟的转向加减速。

运行 joy_control.launch 文件之后，终端会不断打印小海龟的线速度与角速度，此时小海龟运动如下图所示。

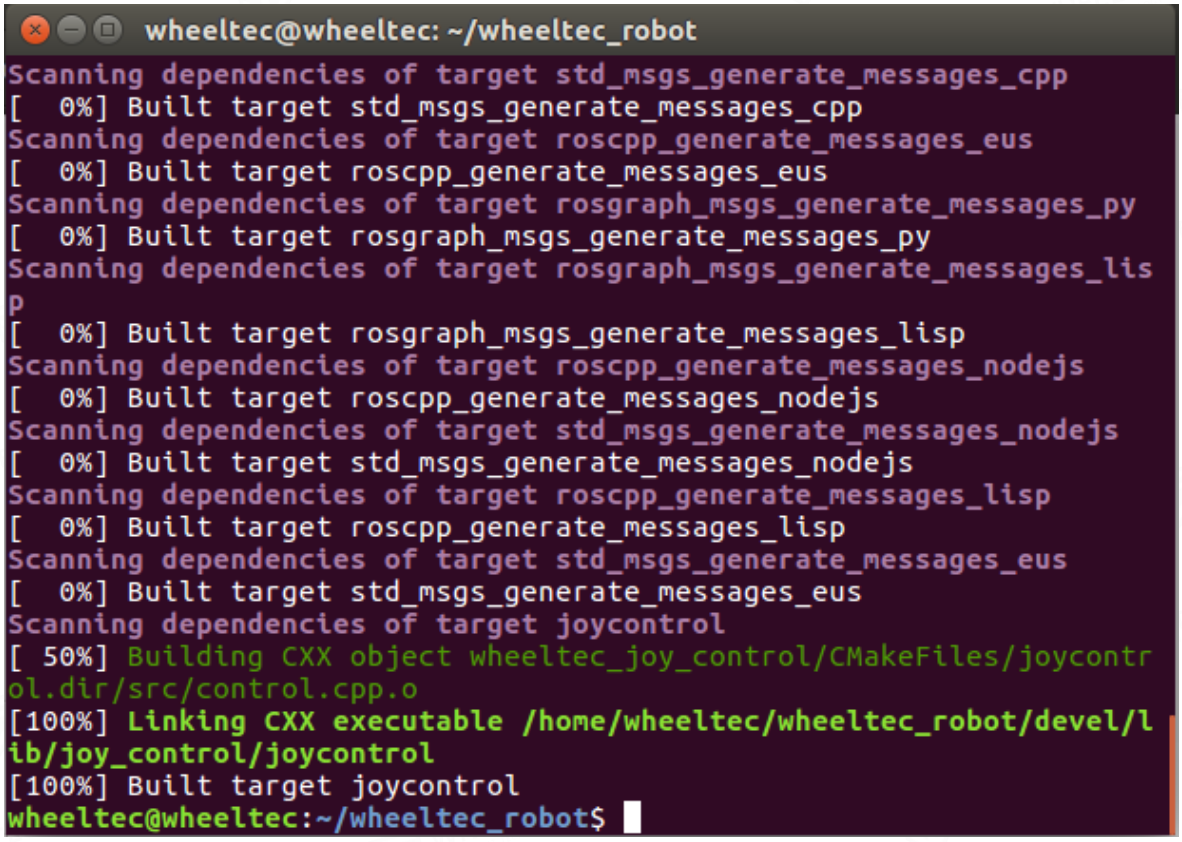


手柄控制小海龟运动

③ 通过无线手柄控制 ROS 小车运动

1) 编译工作空间：

将 ros 主控端使用手柄控制小车运动的功能包【wheeltec_joy_control】拖拽进树莓派的工作空间下使用”catkin_make”命令进行编译。编译完成后如图所示。



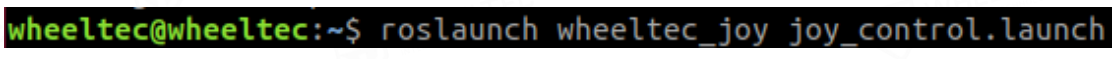
编译手柄控制功能包

2) 启动机器人底盘初始化节点：

```
roslaunch turn_on_wheeltec_robot turn_on_wheeltec_robot.launch
```

3) 在树莓派端运行 launch 文件启动手柄控制小车运动的功能，便可以使用手柄控制小车运动

```
roslaunch wheeltec_joy joy_control.launch
```



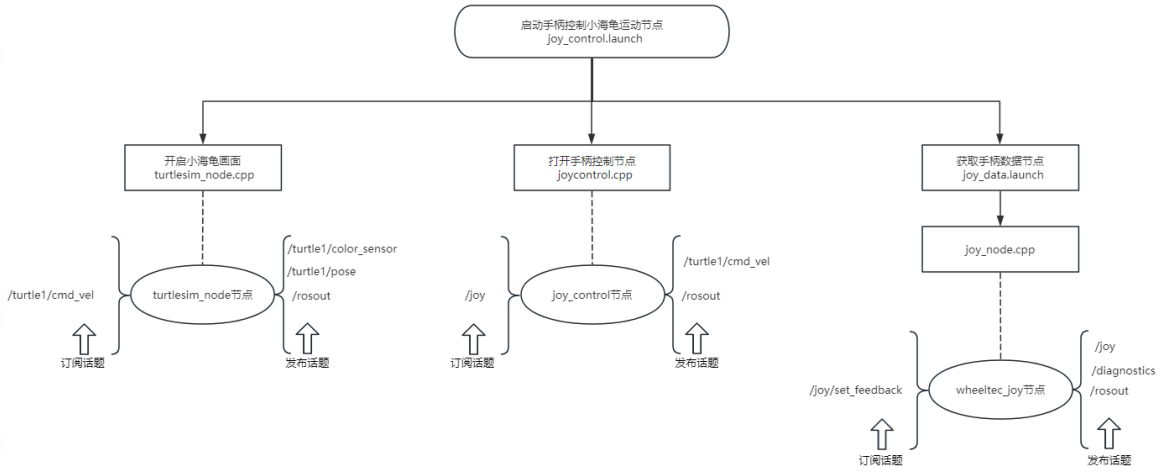
运行 launch 文件

控制操作为：左边摇杆的前后左右四个方向控制小车的前进和后退，以及左右转向，前后推动右边的摇杆实现小车的前进和后退的加减速，左右推动右摇杆实现小车的转向加减速。按下手柄中的【B】键后，切换为麦轮车控制模

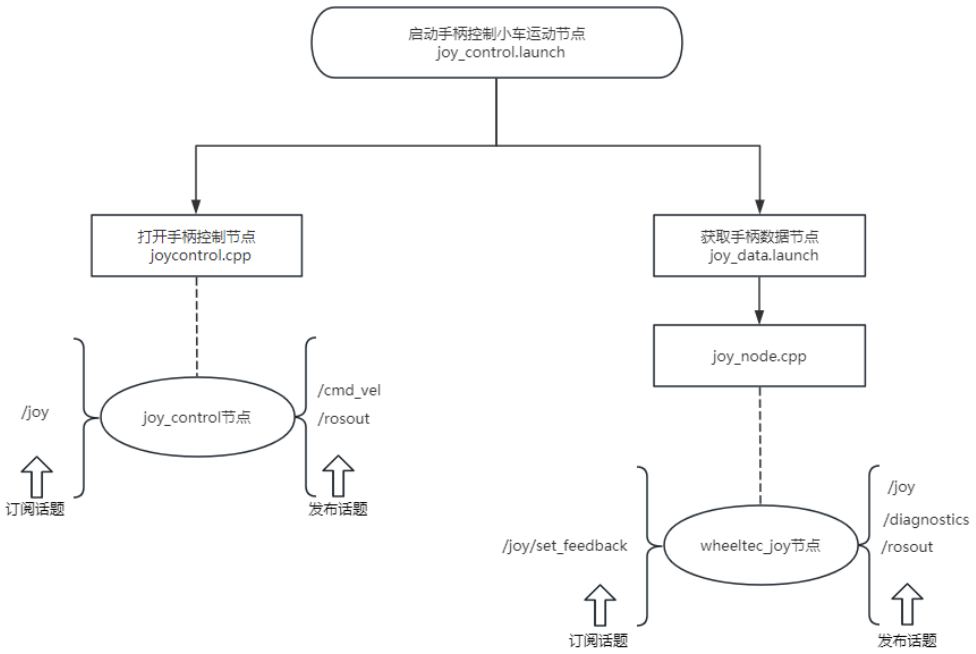
式，可左右平移，按下【A】键时，恢复正常转向模式。需注意：阿克曼小车无法实现自转运动。

3. 使用无线手柄功能程序讲解

① 使用无线手柄功能启动框架：



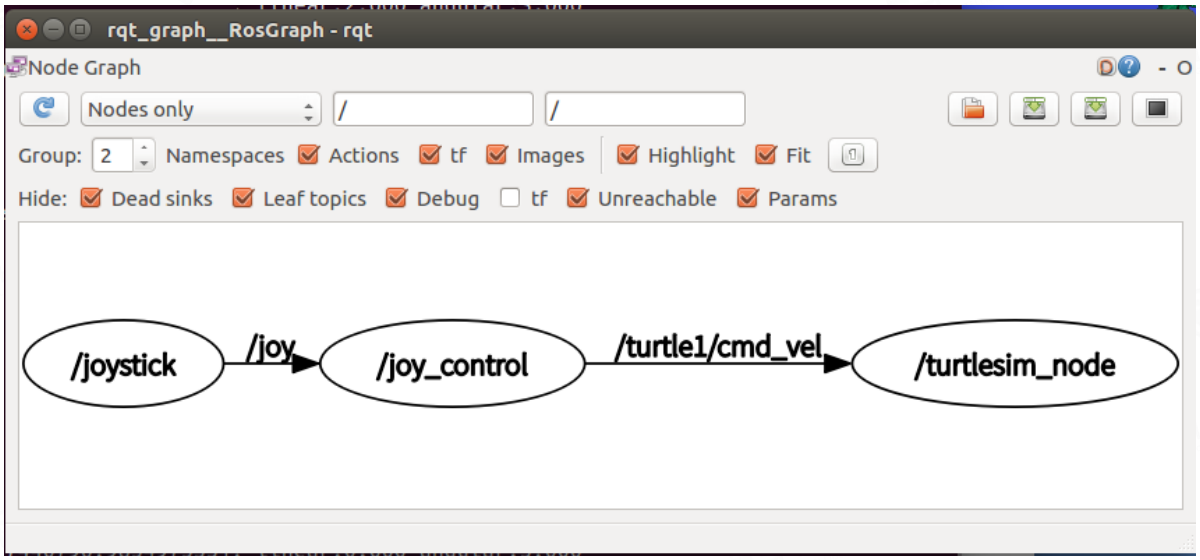
虚拟机端使用手柄控制小海龟运动的启动框架图



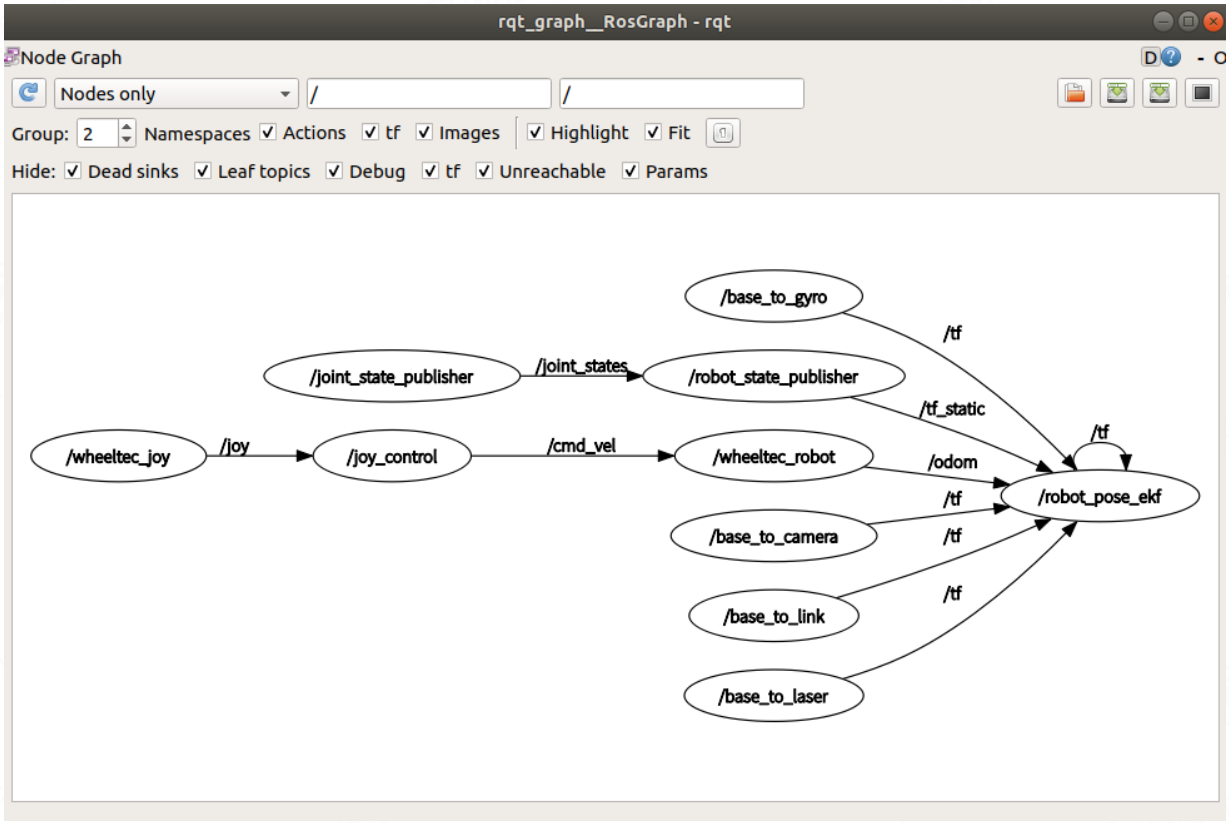
ros 主控端使用手柄控制小车运动的启动框架图

② 查看使用无线手柄功能的节点：

```
rqt_graph
```



虚拟机端使用手柄控制小海龟运动的节点关系图

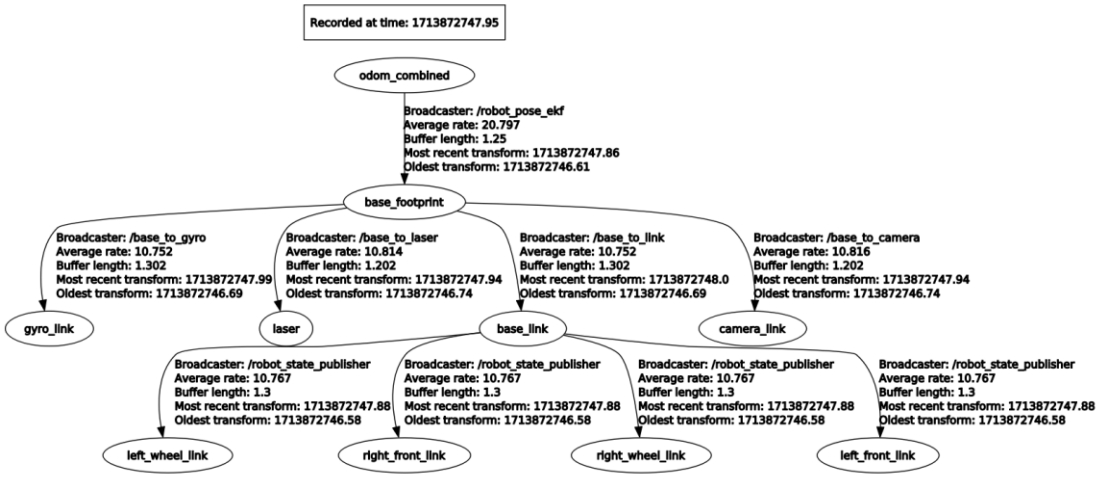


ros 主控端使用手柄控制小车运动的节点关系图

③ 查看使用无线手柄功能的 TF 树：

```
rosrun rqt_tf_tree rqt_tf_tree
```

虚拟机端使用手柄控制小海龟运动功能没有 TF 树。



ros 主控端使用手柄控制小车运动功能 TF 树

④ 使用无线手柄功能的 launch 文件解析：

1) 虚拟机端无线手柄控制小海龟运动的 launch 文件解析：

joy_control.launch 文件实现了使用无线手柄控制小海龟进行运动，首先打开了小海龟运动画面，再打开手柄控制节点以及 ROS 端接收手柄数据节点。文件内容如下图所示。

```
<launch>
    <!--打开小海龟画面-->
    <node pkg="turtlesim" type="turtlesim_node" name="turtlesim_node"/>
    <!--打开手柄控制节点-->
    <node pkg="wheeltec_joy" type="joycontrol" name="joy_control" output="screen">
       <param name="axis_linear"   type="int"      value="1" />
       <param name="axis_angular"  type="int"      value="3"/>
       <param name="vlinear"       type="double"   value="2" />
       <param name="vangular"      type="double"   value="3"/>
    </node>
    <!--获取手柄数据节点-->
   <include file="$(find wheeltec_joy)/launch/joy_data.launch" />
</launch>
```

joy_control.launch 文件内容

control.cpp 文件订阅了订阅手柄发来的数据话题/joy,发布了小海龟运动速度话题/turtle1/cmd_vel 。接收到/joy 话题数据之后，进入 wheeltec_joy() 回调函数。

在控制手柄动作时可以发现，操纵左摇杆前后动作时，发现 axes[1]从 -1 变化到 1，此时将它设置为线速度的方向，并将预先设置好的速度定值赋值给 linear.x；操纵左摇杆左右动作时，发现 axes[0]从 -1 变化到 1，此时将它设置为角速度的方向，并将预先设置好的速度定值赋值给 angualr.z；操纵右边遥感前

后动作时，axes[4]从-1 逐渐变换到 1，此时将它设置为 linear.x 的系数；操纵右边遥感左右动作时，axes[3]从-1 逐渐变换到 1，此时将它设置为 angualr.z 的系数，从而在回调函数中实现通过操作手柄的两个摇杆来控制小海龟运动和加减速，最后将速度数据发布至/turtle1/cmd_vel 话题中。回调函数内容如下图所示。

```
void wheeltec_joy::callback(const sensor_msgs::Joy::ConstPtr& Joy) //键值回调函数
 {
   double vangle_key,linera_key;
   double acce_x,acce_z;
   geometry_msgs::Twist v;
   vangle_key =Joy->axes[0];  //获取axes[0]的值
   linera_key =Joy->axes[1];  //获取axes[1]的值
   acce_x=Joy->axes[4]+1.0;   //读取右摇杆的值对海龟的线速度进行加减速处理
   acce_z=Joy->axes[3]+1.0;   //读取右摇杆的值对海龟的角速度进行加减速处理
   //判断前进后退
   if(linera_key>0)
   {
   dir=1;
   vlinear_x.data=vlinear;
   }
   else if(linera_key<0)
   {
   dir=-1;
   vlinear_x.data=-vlinear;
   }
   else  vlinear_x.data=0;
   //判断左转右转，大于0为左转，小于0为右转
   if(vangle_key>0)  vlinear_z.data=vangular;
   else if(vangle_key<0) vlinear_z.data=-vangular;
   else vlinear_z.data=0;
   //处理数据
   v.linear.x = vlinear_x.data*acce_x;
   v.angular.z = dir*vlinear_z.data*acce_z;
   //打印输出
   ROS_INFO("linear:%.3lf angular:%.3lf",vlinear_x.data,v.angular.z);
   pub.publish(v);
}
```

control.cpp 文件内容

2) ros 主控端使用无线手柄控制小车运动的 launch 文件解析：

joy_control.launch 文件实现了使用无线手柄控制树莓派小车进行运动，运行这个 launch 文件前需要先打开机器人底层控制节点，再打开手柄控制节点以及 ROS 端接收手柄数据节点。在参数服务器中添加了小车起始时的默认角速度与线速度。文件内容如下图所示。

```
<launch>
    <!--打开手柄控制节点-->
    <node pkg="wheeltec_joy" type="joycontrol" name="joy_control" output="screen">
       <param name="axis_linear"   type="int"      value="1" />
       <param name="axis_angular"  type="int"      value="0"/>
       <param name="vlinear"       type="double"   value="0.3" />
       <param name="vangular"      type="double"   value="1"/>
    </node>
    <!--获取手柄数据节点-->
   <include file="$(find wheeltec_joy)/launch/joy_data.launch" />
</launch>
```

joy_control.launch 文件内容

control.cpp 文件和在虚拟机端使用的 control.cpp 文件大致相同。改动地方为订阅了手柄发来的数据话题/joy,发布了控制小车运动的速度话题/cmd_vel 。接收到/joy 话题数据之后，进入 wheeltec_joy() 回调函数。文件内容如下图所示。

```
void wheeltec_joy::callback(const sensor_msgs::Joy::ConstPtr& Joy) //键值回调函数
 {
   double vangle_key,linera_key,acce_x,acce_z;
   geometry_msgs::Twist v;
   vangle_key =Joy->axes[0];  //获取axes[0]的值
   linera_key =Joy->axes[1];  //获取axes[1]的值
   acce_x=Joy->axes[4]+1.0;   //读取右摇杆的值对机器人的线速度进行加减速处理
   acce_z=Joy->axes[3]+1.0;   //读取右摇杆的值对机器人的角速度进行加减速处理
   //判断前进后退
   if(linera_key>0)
   {
       dir=1;
       vlinear_x.data=vlinear;}
   else if(linera_key<0)
   {
      dir=-1;
      vlinear_x.data=-vlinear;}
   else  vlinear_x.data=0;
   //判断左转右转，大于0为左转，小于0为右转
   if(vangle_key>0)      vlinear_z.data=vangular;
   else if(vangle_key<0) vlinear_z.data=-vangular;
   else vlinear_z.data=0;
   //处理数据
   v.linear.x = vlinear_x.data*acce_x;
   v.angular.z = dir*vlinear_z.data*acce_z;
   if(Joy->buttons[1]==1) //按下B键时，切换为麦轮车，可左右平移
     flag_mec=1;
   if(Joy->buttons[0]==1) //按下A键时，恢复正常转向模式
     flag_mec=0;
   if(flag_mec)
   {
     v.linear.y=0.2*vlinear_z.data*acce_z;
     v.angular.z=0;
   }
   //打印输出
   ROS_INFO("linear:%.3lf angular:%.3lf",vlinear_x.data,v.angular.z);
   pub.publish(v);
}
```

control.cpp 文件内容