HOWTO BLUEROV REV4 HEAVY

Revision nov 2024, MIR intake 4.

INTRODUCTION

The new Bluerovs now embed a Navigator board as a hat board directly connected to a RPI 4, instead of a Pixhawk board linked to a RPI3 through USB cable. In addition, the RPI4 is now running a linux OS with the BlueOS layer from BlueRobotics that allows configuring the system more easily. On the old bluerovs with RPI3/Pixhawk, a linux/ros1 system was embedded, with the mavros node being run onboard to deal with MAVlink communications with the hardware.
Now, there is no more ros onboard (it could be possible to embed one, but it is cumbersome due to blueOS running at the same time, and it is no more necessary).
As a consequence, the mavros node (ros2) will have to be run on your laptop (see launch files below).
BlueOS is configured to run a MAVlink endpoint client onboard, where the mavros node and the QGC (QGroundControl) on your laptop can connect to, the MAVlink endpoint acting as a relay of sensor readings/commands, etc.
It is now possible to run the QGControl interface to do the calibration steps, without any change inside the RPI. However, QGControl appears to be not very stable and may crash some time after the calibration. In addition, QGControl cannot be run with mavros at the same time, if one wants to control the robot with the gamepad (QGControl uses a value of 255 for SYS_MYCGS whereas mavros uses a value of 1).

INSTALLATION OF LINUX/ROS2 ON LAPTOP

1) install Ubuntu 22.04 LTS, follow instructions on website. It is recommended to make space for a linux partition and install a dual boot.
2) install ros humble or ros iron (both are ros2 versions)
3) install mavros and mavlink
sudo apt-get install ros-iron-mavros ros-iron-mavlink ros-iron-mavros-extras ros-iron-mavros-msgs
install joy package :
sudo apt-get install ros-iron-joy
4) install python3 stuff if not already done:
sudo apt install -y python3-vcstool python3-rosinstall-generator python3-osrf-pycommon
5) Install GeographicLib datasets:
wget https://raw.githubusercontent.com/mavlink/mavros/ros2/mavros/scripts/install_geographiclib_datasets.sh
chmod u+x install_geographiclib_datasets.sh
sudo ./install_geographiclib_datasets.sh
6) install additional stuff for 3D transformations:
sudo apt-get install ros-iron-tf-transformations
sudo pip3 install transform3d
7) Install packages for the video stream:
sudo usermod -a -G dialout $USER
sudo apt-get remove modemmanager -y
sudo apt install gstreamer1.0-plugins-bad gstreamer1.0-libav gstreamer1.0-gl -y
sudo apt install libfuse2 -y
sudo apt install libxcb-xinerama0 libxkbcommon-x11-0 libxcb-cursor0 -y
8) rviz2 can be installed, but it will not be necessary:
sudo apt-get install ros-iron-rviz2
9) Create workspace:
mkdir -p ~/ros2_ws/src
cd ~/ros2_ws/src
10) Download QGroundControl.*AppImage*
give execution rights to user : ***chmod u+x ./QGroundControl.AppImage***
run it : ***./QGroundControl.AppImage***
11) install jstest-gtk for joystick calibration : ***sudo apt-get install jstest-gtk***
12) Install pingviewer on the laptop (download the AppImage for linux): **https://github.com/bluerobotics/ping-**

viewer

INSTALLATION OF ROS2 PACKAGES

- Get ros2 **packs_marine_mech.tgz** from moodle and copy it into ros2_ws/src folder.
- Untar the file :
  tar xzvf packs_marine_mech.tgz
- Compile in workspace:
  cd ~/ros2_ws
  colcon build
- install:
  source install/setup.bash

N.B. the source command can be put in the **.bashrc** file located in the connection directory. However, it must be executed each time colcon has been run.

NETWORK CONFIGURATION AND CONNECTION TO THE ROBOT

- Plug the yellow cable of the robot into the bluebox, and connect the USB cable of the bluebox to the laptop.
- Create a network connection (USB/Ethernet) with the blue box :
  - Give a name to the connection, e.g. BR2
  - Select Manual mode
  - Set local IP (of the laptop) 192.168.2.1, and netmask 255.255.255.0
- Connect the battery to the robot
- Wait a little bit (~1min) until a second sound is emitted, which means that the Navigator board has finished to boot.
- Open a browser and enter the default IP address of the robot – 192.168.2.2 - to get access to the interface of BlueOS. With this interface, you can check the detection of the camera, sonar pinger, ardupilot hardware, etc. It is possible to enable the pirate mode to get more information (icon on top right).

CALIBRATIONS

1. **Sensors, especially IMU**

- Click on the Accelerometers, choose “Roll90” for the Autopilot Orientation and click Ok.
- Follow the instructions to calibrate the accelerometer. After the calibration process, you will be prompted to reboot the autopilot. Click on reboot and wait for the ROV to reconnect.
- Click on Compass, make sure “Roll90” is still selected, and click Ok.
- Follow the instructions to calibrate the compass.
- The robot must be moved vigorously in all directions.
- After the calibration process, you will be prompted to reboot the autopilot. Click on reboot and wait for the ROV to reconnect.
- Click on Calibrate Pressure and wait for the calibration to complete.
- When completed, the Sensors tab will no longer be red.

2. **Gampepad**

- Connect the gamepad to the laptop
- With QgroundControl, go to the Vehicle Settings page in QGroundControl, then click on the red Joystick tab in the sidebar on the left. “RC Mode” selection must be set to 3 for the Logitech F310 gamepad.
- In the Advanced settings tab, select « Center stick is zero throttle »/ »Allow negative Thrust »
- authorize deadbands, just move a little bit each stick, see calibration tab to check deadbands in red. Deadbands are necessary, otherwise, QGControl may send noisy commands to the robot while sticks are in zero position. **Check with the general tab that the sticks come back to the zero position when not activated.**
- **Proceed to the calibration with the calibration tab.**
- **Check with the general tab that the sticks work as expected.**

- If QGControl is not used, use *jstest-gtk to calibrate the joystick.*
  N.B. : it may not be sufficient for use with QGControl, due to the deadbands, see above.

3. **Motors, especially direction : BE CAREFUL**

BLUEROV inside WATER with slighlty positive buoyancy !!!
The direction that the thrusters will spin depends on how the vehicle and thrusters were assembled, so each thruster's forward/reverse direction must be configured in software.
To begin, navigate to the Vehicle Settings page in QGroundControl and select the Motors tab in the sidebar on the left, then proceed with Automatic Motor Direction Detection routine (recommended), or configure them manually.
Be sure to keep all body parts and clothing clear of thrusters while the BlueROV2 is armed.

4. **Echosounder pinger**

- Run the pingviewer app :
  chmod u+x pingviewer.AppImage
  ./pingviewer.AppImage
- Put the robot inside water
- check distance readings from the pingviewer
- Once distance appears to be correct for several measurements, get the parameters set by pingviewer on the output screen, e.g. :

```
#1_pingInterval: 66
#1_speedOfSound: 1500000
#2_automaticMode: 1
#3_gainIndex: 0
#3_lengthDistance: 5000
#3_startDistance: 0
```

- set the parameters inside file ping1_components.py from pinger_sonar_ros package :

```
### Declare ROS 2 Parameter
self.declare_parameter('speed', 1500000)
self.speed_:float = self.get_parameter('speed').value
self.declare_parameter('interval_num', 66)
self.interval_num_:float = self.get_parameter('interval_num').value
self.declare_parameter('gain_num', 0) # int 0 - 6
self.gain_num_:int = self.get_parameter('gain_num').value
self.declare_parameter('scan_start', 0) # default 100 [mm] range(30 to 200)
self.scan_start_:float = self.get_parameter('scan_start').value
self.declare_parameter('scan_length', 5000) # default 2000 [mm] range(2000 to 10000)
self.scan_length_:float = self.get_parameter('scan_length').value
self.declare_parameter('mode_auto', 1) # default 0: manual mode, 1: auto mode
self.mode_auto_:int = self.get_parameter('mode_auto').value
```

- Compile (cd ros2_ws, colcon build, source install/setup.bash)
- Relaunch run_pinger.launch
- Check with topic : **ros2 topic echo /bluerov2/ping1d/data**

5. **Camera**

Using BlueOS, it is possible to access the camera parameters to tune them (click on Configure). Just modify them and check impact on the image.

RUN ROS2 NODES

Run mavros on laptop terminal:
ros2 launch autonomous_rov run_mavros.launch

In another terminal, run listenenMIR and joystick nodes
ros2 launch autonomous_rov run_listener_MIR_joy.launch

N.B.

- if gamepad not detected, change the device path "joy_dev" inside the launch file
- QGControl uses a value of 255 for SYS_MYCGS whereas mavros uses a value of 1. **Be sure to have SYS_MYCGS set to 1 when using python programming with mavros**. SYS_MGCS can be set with BlueOS (see list of Ardupilot parameters) or with QGControl. QGControl cannot be run with mavros at the same time.

Camera :
In another terminal, run camera nodes :
ros2 launch autonomous_rov run_video.launch

Sonar pinger :
In another terminal, run pinger node:
ros2 launch ping_sonar_ros run_pinger.launch

Check thruster commands, sensor feedback, etc.
useful commands : ros2 topic list, ros2 topic hz topic_name, ros2 topic echo topic_name, ros2 node list, …
Normally, IMU sensor feedback is programmed to run at 25 Hz, the pinger feedback at 10 Hz.
The camera flux depends on the size of the image.

NB.
from the ros2_ws directory, each time you modify a launch or python file to implement algorithms, do not forget to run the commands:
***colcon build*, and**
source intall/setup.bash

TROUBLESHOOTING

- read BlueRobotics doc, **https://bluerobotics.com/learn/bluerov2-software-setup/**
- passwd for the RPI4 : raspberry (ssh pi@192.168.2.2)
- There may some conflict with conda (if you installed conda), if so just remove conda / fix path variables
- If chrome doesn't work with connecting to 192.168.2.2 for some reason, just use firefox instead.