Installation guide

Bachelor E2205

This installation guides provides a detailed explanation on how to setup the controller package, as well as how to effectively use the parameter file to change settings.

May 31, 2022

Content

Connect to BlueROV2 (SKIP IF NOT USING MODIFIED BLUEROV2)

- Open a new terminal to setup communication between PC and BlueROV2:

```
sudo netplan generate
sudo netplan apply
export ROS_DOMAIN_ID=1
sudo ifconfig lo multicast
```

Connect to BlueROV2 (SKIP IF NOT USING MODIFIED BLUEROV2)

- Make sure the Phantom Spool thether is connected to the Fanthom-X Tehther Interface and the Interface is connected to the PC via USB
- Open four terminals and SSH into the BlueROV2 in every one:

```
ssh ubuntu@192.168.2.2
password: ***
```

- In the first terminal, launch the actuator driver:

```
ros2 launch bluerov_actuator_driver actuator_driver.launch.py
```

- In the second terminal:

```
ros2 lifecycle set /bluerov_actuator_driver configure
ros2 lifecycle set /bluerov_actuator_driver activate
```

Connect to BlueROV2 (SKIP IF NOT USING MODIFIED BLUEROV2)

- In the third terminal, launch the Kalmanfilter:

```
ros2 run brov2_qekf brov2_qekf_exe
```

- In the fourth terminal, launch the rest of the sensors:

```
ros2 launch Navigation-brov2/launch/brov2_launch.py
```

- To verify that ROS on the BlueROV and on the PC is communicating, go to the first terminal where the communication between the PC and BlueROV2 was initiated. Check if any topics from the BlueROV2 are available:

```
ros2 topic list
```

  if not, reboot the pc with the USB cable connected and start from the top.

Setting up ROS2

- Clone repository

```
cd ~
mkdir -p bs_ws/src && cd bs_ws/src
git clone https://github.com/ErikRowe/Bachelor-E2205
```

- Build the controller package

```
cd ~/bs_ws
colcon build --packages-select controller_package && source install/setup.bash
```

- If using Bassos drivers, build bluerov interfaces and bridge. The interface package must be built and sourced before the bridge package

```
colcon build --packages-select bluerov_interfaces && source install/setup.bash
colcon build --packages-select bluerov2_interface_bridge
```

Setting up ROS2 2

- ROS2 nodes are now ready to be launched. To reduce chances of unexpected behavior, open a new terminal for every node and source the install.

```
    cd ~/bs_ws && source install/setup.bash
```

- Launch joy if using teleoperation:

```
    ros2 run joy joy_node
```

- Launch bluerov2 bridge if using Basso drivers:

```
    ros2 run bluerov2_interface_bridge bridge_node
```

- Launch controller package:

```
    ros2 launch controller_package controller_launch.py
```

Setting up ROS2 2

- To change controller settings during runtime, make changes to the parameter file:

```
nano ~/bs_ws/src/Bachelor-E2205/controller_package/params/params.yaml
```

- Upload the changes using (note: params.yaml is not on a new line):

```
ros2 param load /Control_Node ~/bs_ws/src/Bachelor-E2205/controller_package/params/
params.yaml
```

Teleoperation

- Teleoperation, or manual control, may work separately or alongside the controller. For pure teleoperation, set *Enable_controller* : *false* in the parameter file
- Joystick input is always available, as long as a joy node is running. If *Load_setpoint_from_topic* : *true*, the joystick is allowed to move the vehicle but not change the setpoint. When releasing the joystick, the controller (if enabled) will return to the setpoint loaded from ROS2 topic
- Teleoperation sensitivity may be adjusted using *Manual_control_scaling*
- If the vehicles body frame is represented in *NED*, turn *Compensate_NED* : *true*
- Button mapping may be changed in the code, but this requires rebuilding and sourcing the controller package. Change button mapping here:

```
nano ~/bs_ws/srs/Bachelor-E2205/controller_package/src/joy_to_action.cpp
```

Attitude control

- Make sure a state estimate is available and publishing as an odometry message to: *CSEI/observer/odom*
- Attitude control is active as long as *Enable_controller : true*.
- Pure attitude control is active when *Linear_control_xy : false* and *Linear_control_z : false*
- If *Load_setpoint_from_topic : false* the attitude setpoint is chosen to be where the vehicle is when a joystick applying rotation is released

Depth hold

- By turning *Linear_control_z* : *true*, the controller may control the depth (z) value. This requires a state estimate providing depth information
- As with attitude control, the setpoint is chosen either from a topic or when a joystick providing changes in z is released

Dynamic positioning

- By turning *Linear_control_xy* : *true* the controller may control the position (xy) values. This requires a state estimate providing position information
- As with attitude control, the setpoint is chosen either from a topic or when a joystick providing changes in x and/or y is released