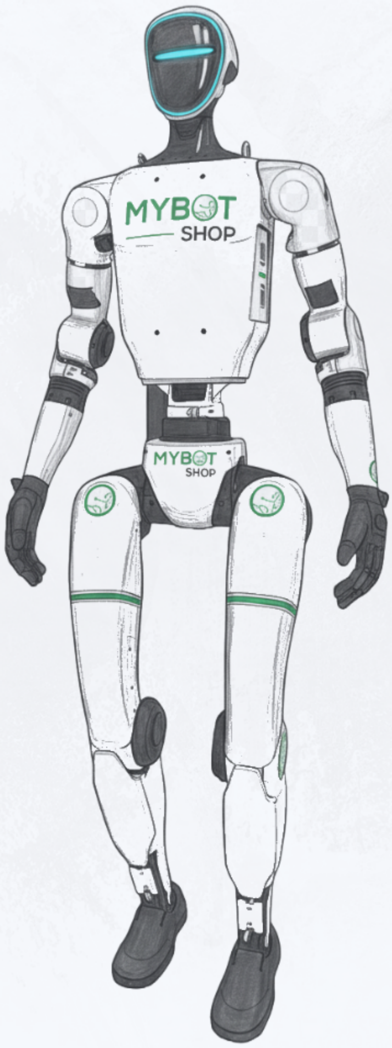
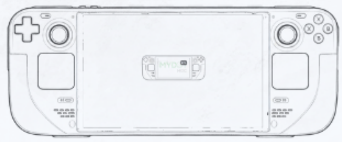
QUADRUPED ROBOTICS

QRE G1 | Steamdeck

USER MANUAL





info@mybotshop.de

support@mybotshop.de

forum.mybotshop.de

docs.mybotshop.de

info@quadruped.de

support@quadruped.de

forum.mybotshop.de

docs.quadruped.de

Contents

1 | Attention

Read this document carefully. In case of doubt, it is essential to consult specialists, experts, or manufacturers of the assemblies used! The robot must not be put into operation before clarification.

In case of doubt, please consult the guides below or contact a specialist, expert, or manufacturer of the assemblies used! The robot must not be put into operation before clarification. Contact information is available in the forums of the Guideline Section 11.

Do not allow persons who are not familiar with the robotics or these instructions to operate the robot. Robots are dangerous in the hands of untrained users.

2 | Disclaimer

The provided robot, sold by *MYBOTSHOP GmbH*, is an R&D device and does not have CE Marking and/or Certificate of Incorporation. Basic ROS understanding is required. If you are not familiar with ROS, we recommend checking the ROS2 Wiki first.

Please note that the quoted robot is considered a partly completed machine under the *Machinery Directive2006/42/EC*, and it does not carry a CE marking.

The client acknowledges and agrees that any information or materials provided by *MYBOTSHOP GmbH* are for R&D purposes only. Any services are provided "AS IS" and without any representation or warranty of any kind, express or implied, including but not limited to any warranty of merchantability, fitness for a particular purpose, non-infringement, or any other warranty.

MYBOTSHOP GmbH shall not be liable for any damages, including but not limited to direct, indirect, special, incidental, or consequential damages, arising out of or in connection with the use or inability to use the information or materials provided. This limitation on liability shall apply regardless of the form of action, whether in contract, tort, or otherwise.

3 | Quick-Start

3.1 | Powering-on

- Power on the robot by pressing the battery once and then again for 3 seconds.
- Ensure the battery is fully charged before powering on the robot.
- If a unitree remote is available, you can also power on the remote by pressing the power once and then for 3 seconds.

3.2 | Webserver

This module should come pre-installed for heavy integration projects. It should be accessible directly at http://192.168.123.164:9000/ or the WiFi ip to which the robot is connected. The G1 webserver can be configured via the config file in **g1_webserver** ros2 package located in **/opt/mybotshop/src/mybotshop/g1_webserver/config/robot_webserver.yaml**

3.2.1 | Login

- Username: **admin**
- Password: **mybotshop**

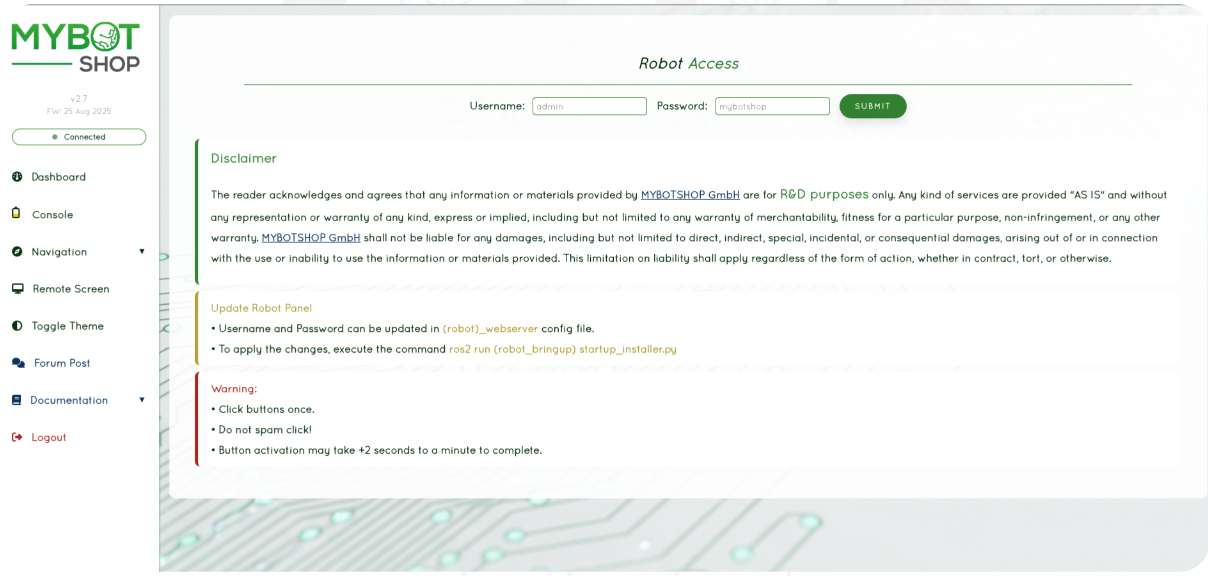


Figure 3.1: Web Login

3.2.2 | Dashboard

- View IP Address of the G1
- View System load

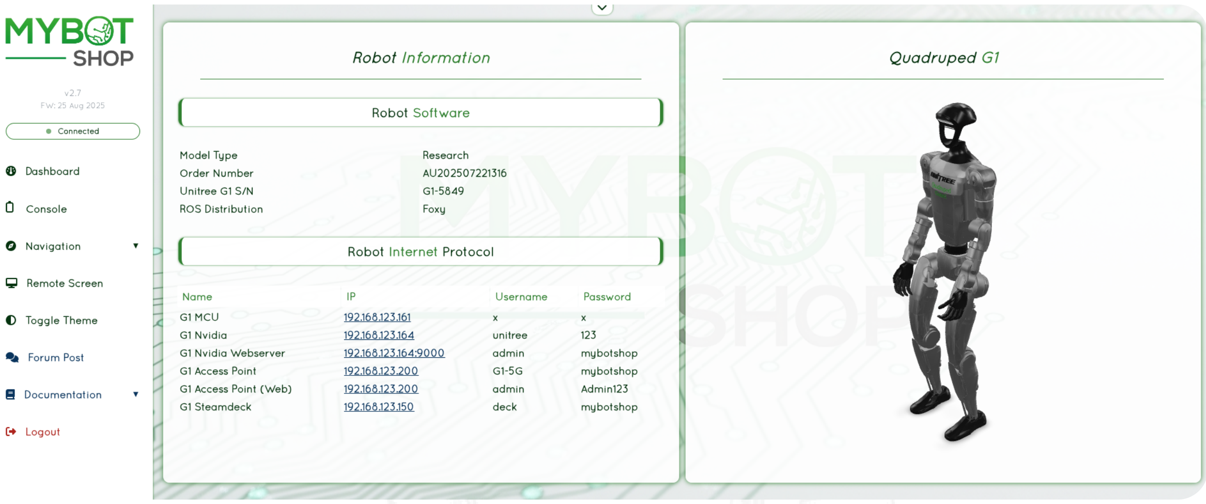


Figure 3.2: Web Dashboard

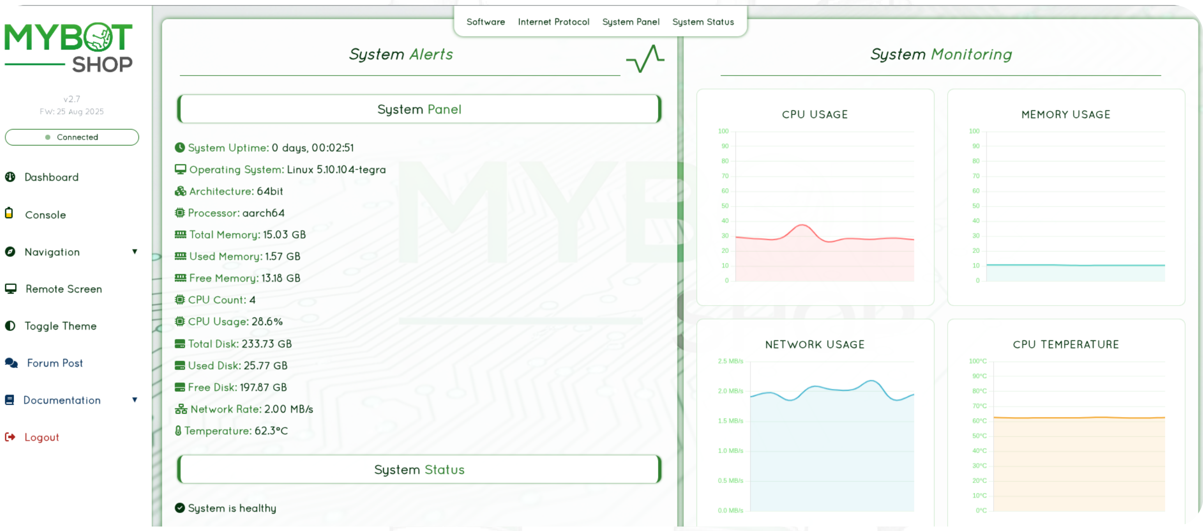


Figure 3.3: Web System

3.2.3 | Console

- Enable the G1 ROS2 Services
- Disable the G1 ROS2 Services
- Record System G1 logs
- Battery status
- Pre-configured action buttons
  - Switch Gaits
  - Stand up
  - Sit down
  - Adapatable to new ros2 services

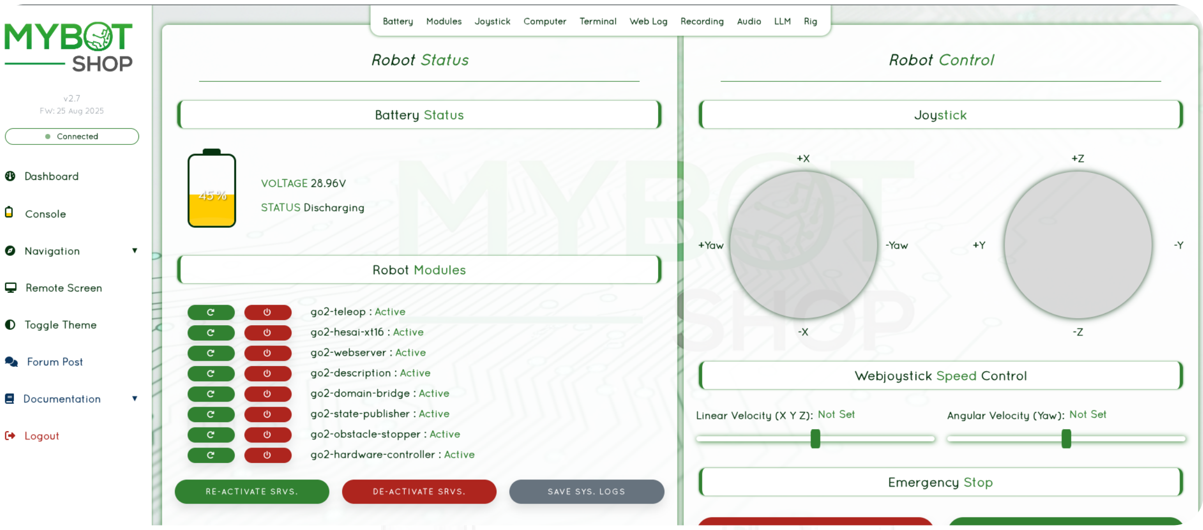


Figure 3.4: Web Console

- Web Joystick
- Online image stream
- STT to A.I to TTS (Disabled)

3.2.4 | Remote Desktop

- On-board screen of the Unitree G1's computer

Figure 3.5: Web Remote Desktop

3.2.5 | Navi Indoor

- Disabled

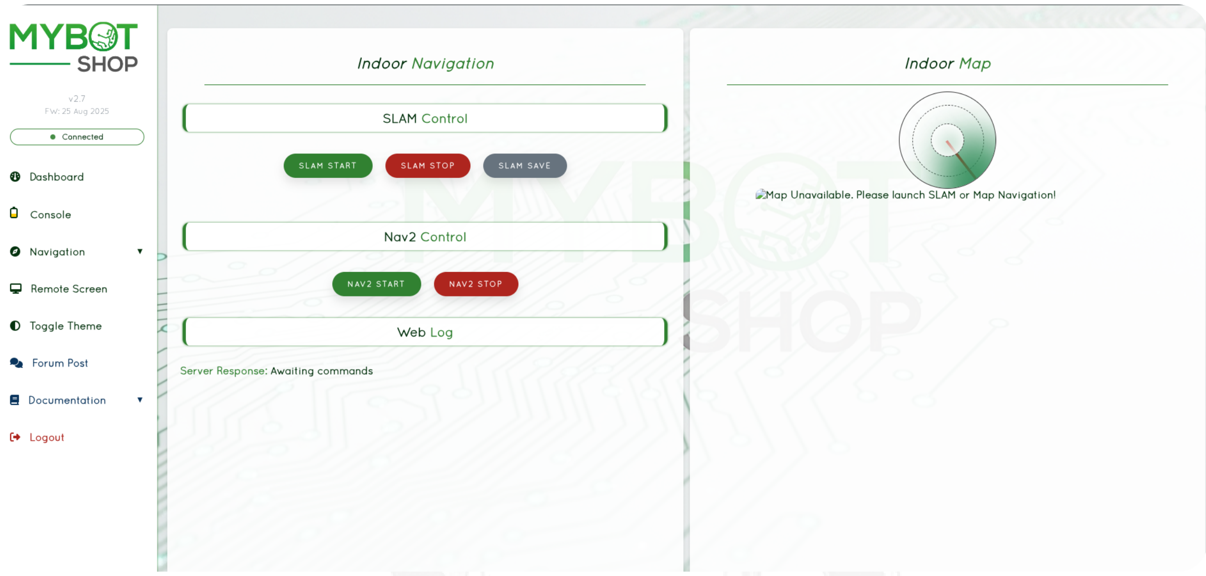


Figure 3.6: Web Navigation Indoor

3.2.6 | Navi Outdoor

- Disabled

3.2.7 | LLM Interface

- Disabled

3.3 | Robot Interface

Instructions for interfacing with the robot using **Ubuntu 20.04** and **ROS2 Foxy**.

> **Note:** This procedure should be followed after setting up and pairing with the G1 Edu. Furthermore, all of the G1's functionality should be verified via the app. Instructions for set up can be found at G1 Docs also available in section 11. This guide builds upon the information from the docs. The QRE G1 operates in ROS DOMAIN ID 10. Ensure that your ROS2 environment is set to the same domain ID when interfacing with the G1's QRE packages.

3.3.1 | Static Network Connection

For the first time, one needs to connect through a **LAN** cable to configure the robot's network.

To create a static connection in your PC (not the robots), in Ubuntu go to Settings → **Network** then click on + and create a new connection.

1. Change the connection to **Manual** in the **IPv4** settings.

2. Set the **Address** IP as **192.168.123.51** and the **Netmask** as **24**.

3. Click save and restart your network.

After a successful connection, check the host's local IP by typing in the Host PC's terminal:

```
ifconfig
```

Now, ping the robot:

```
ping 192.168.123.164
```

Access the robot via SSH:

```
ssh -X unitree@192.168.123.164
```

The default password is:

```
123
```

3.3.2 | G1 IP Addresses

Robot	IP Addresses	Username	Password
G1 MCU	192.168.123.161	-	-
G1 External	192.168.123.164	unitree	123
G1 External Web	http://192.168.123.164:9000	admin	mybotshop
Livox Mid360	192.168.123.20	-	-
Router SSID	192.168.123.200	G1XXXXX-5G	mybotshop
Router Web	192.168.123.200	-	Admin123
Steamdeck	192.168.123.150	deck	mybotshop

> **Note:** Sometimes other networks can cause disruptions when connecting to the G1. It is best to have only your connection to the robot active and all others inactive.

3.3.3 | Interfacing

- Firstly, connect to the G1 as described in the Network section 3.3.
- Secondly, open several ssh sessions into the g1 via

  ```
  ssh -X unitree@192.168.123.164
  ```

  - Verify that the robot is processing information by typing the command in an ssh session.

```
ros2 topic echo /sportmodestate
```

- □ If data is displayed, the drivers are running.

3.4 | Visualization

You can view the G1's current state by typing in one of the ssh sessions:

```
ros2 launch g1_viz view_robot.launch.py
```

3.5 | Teleop

This requires installation of the ROS2 Modules on G1. If not done please follow installation instructions.

```
ROS_DOMAIN_ID=10 ros2 run teleop_twist_keyboard \
teleop_twist_keyboard \
--ros-args --remap cmd_vel:=/g1_unit_0001/cmd_vel
```

4 | ROS2 Modules

4.1 | G1 Overview

> **Note:** Please ensure that the arms are straight down when you power on the robot, if the orientation is different, it will effect the ros2 control operation.

Starts driver for operating G1 in high-level mode for the legs and arms:

> **Important:** The arms will initialize and move such that the forearms will be facing forward.

Launch the g1 ros2 driver which includes

- Domain Bridge
- Joint States
- Robot Description
- Arm Control
- Leg Control (High-level)
- Inertial Measurement Unit Publisher
- Sensor Fusion

- Twist Mux
- D435i depth camera driver
- Livox Mid360 lidar driver

These files can be viewed in the webserver and activated and deactivated from there.

4.2 | Auto Drivers Startup

Note: The robot system can be verified by typing in the terminal. Generally, all drivers are off except the webserver and can be activated via web server or by the command:

```
sudo service g1-platform start
```

The list of all drivers can be found in the g1_bringup startup_installer.py

```
sudo service g1-webserver status
sudo service g1-platform status
```

If it is available and green, then do not launch the **g1_bringup** as it is running in the background.

Launch the g1 ros driver the communicates and publishes the state of the robot joints.

```
ros2 launch g1_bringup system.launch.py
```

The G1 ordinarily does not have a startup job unless otherwise specified (In this case the G1 has a startup job). The G1 launches the **g1_bringup system.launch.py**. To verify if the startup job is available in G1. Run the command:

```
sudo service g1-platform status
```

Note: If an error such as **Unit ros2.service could not be found.**, then it means that there is no startup installed.

1. The red marker in the service indicates that the startup job has failed.
2. Green marker indicates everything is working correctly.
3. Grey marker indicates that the service has not started yet.

In case of red or grey marker, you may restart the service via:

```
sudo service g1-platform restart
```

If you want to modify the upstart job or add other ROS launch files to it then it is recommended to add your changes to the main upstart file i.e. **system.launch.py** located in the **g1_bringup** package. Once done, save the file and run the following command to update startup job.

```
ros2 run g1_bringup startup_installer.py
```

4.3 | RViz

Launch the g1 rviz

```
ros2 launch g1_viz view_robot.launch.py
```

4.4 | Tele-Operation

Teleop g1

```
ROS_DOMAIN_ID=10 ros2 run teleop_twist_keyboard \
teleop_twist_keyboard \
--ros-args --remap cmd_vel:=/g1_unit_0001/hardware/cmd_vel
```

4.5 | Core Launch Files

These files can be launched/activated from the webserver as well. The webserver should be on by default. The installation procedure should be followed first.

```
ros2 launch g1_platform highlevel_ros.launch.py
```

```
ros2 launch g1_platform domain_bridge.launch.py
```

```
ros2 launch g1_platform state_publisher.launch.py
```

```
ros2 launch g1_webserver webserver.launch.py
```

```
ros2 launch g1_control twistmux.launch.py
```

```
ros2 launch g1_lidar livox_mid360.launch.py
```

```
ros2 launch g1_platform audio.launch.py
```

```
ros2 launch g1_platform led.launch.py
```

```
ros2 launch g1_platform videostream.launch.py
```

4.5.1 | Arms 7Dof Core Launch Files

```
ros2 launch g1_description g1_29_description.launch.py
```

```
ros2 launch g1_platform arm_7dof.launch.py
```

4.5.2 | Arms 5Dof Core Launch Files

```
ros2 launch g1_description g1_23_description.launch.py
```

```
ros2 launch g1_platform arm_5dof.launch.py
```

4.5.3 | SDK Examples

```
cd /opt/mybotshop/04Aug2025_unitree_sdk2/build/
cmake .. && make && sudo make install
```

```
cd /opt/mybotshop/04Aug2025_unitree_sdk2/build/bin
./g1_loco_client --network_interface=eth0 --shake_hand
./g1_audio_client_example eth0
```

4.6 | Modes Selection

4.6.1 | Posture & State Commands

```
|Command              |Description                       |
|---------------------|----------------------------------|
|‘damp‘               |Set all motors to damping mode.   |
|‘start‘              |Start locomotion control.         |
|‘squat‘              |Transition to squat posture.      |
|‘sit‘                |Sit down.                         |
|‘stand_up‘           |Stand up from sitting/squatting.  |
|‘zero_torque‘        |Disable torque on all motors.     |
|‘stop_move‘          |Stop all motion immediately.      |
|‘high_stand‘         |Stand at high position.           |
|‘low_stand‘          |Stand at low position.            |
|‘balance_stand‘      |Stand with balance mode active.   |
|‘shake_hand‘         |Perform handshake sequence        |
|‘wave_hand‘          |Perform wave motion.              |
|‘wave_hand_with_turn‘|Wave hand and turn simultaneously.|
```

4.6.2 | Setter Commands (Change Robot State)

```
|Command                      | Parameters                               |
|-----------------------------| -----------------------------------------|
|‘set_fsm_id=<id>‘            | Integer FSM ID to switch to.             |
|‘set_balance_mode=<0/1>‘     | Enable (1) or disable (0) balance mode. |
|‘set_swing_height=<value>‘   | Set swing height in meters.              |
|‘set_stand_height=<value>‘   | Set stand height in meters.              |
|‘set_velocity="vx vy w dur"‘| Set velocity (m/s, m/s, rad/s, dur in s)|
|‘move="vx vy w"‘             | Command motion without duration.         |
|‘set_task_id=<id>‘           | Set active task ID.                      |
|‘set_speed_mode=<mode>‘      | Change locomotion speed mode.            |
```

4.6.3 | Getter Commands (Query Current Robot State)

```
|Command            |Returns                         |
|-------------------|--------------------------------|
|‘get_fsm_id‘       |Current finite state machine ID|
|‘get_fsm_mode‘     |Current FSM mode.               |
|‘get_balance_mode‘|Current balance mode (0/1).     |
|‘get_swing_height‘|Current swing foot height.      |
|‘get_stand_height‘|Current standing height.        |
|‘get_phase‘        |Current gait phase vector.      |
```

4.6.4 | Toggle Commands

```
|Command                       | Params|Description                       |
|------------------------------| ------|----------------------------------|
|continous_gait=<true/false>   | bool  |Enable/disable continuous gait.   |
|switch_move_mode=<true/false>| bool  |Switch b/w movement and standing|
```

4.6.5 | Examples

Robot Startup Sequence

- Damp Mode

```
ROS_DOMAIN_ID=10 ros2 service call /g1_unit_0001/hardware_modes \
g1_interface/srv/G1Modes \
'{"request_data": "damp"}'
```

- Locked standing Mode

```
ROS_DOMAIN_ID=10 ros2 service call /g1_unit_0001/hardware_modes \
g1_interface/srv/G1Modes \
'{"request_data": "stand_up"}'
```

- Lift the robot to its feet and then put in Locomotion Mode

```
ROS_DOMAIN_ID=10 ros2 service call /g1_unit_0001/hardware_modes \
g1_interface/srv/G1Modes \
'{"request_data": "start"}'
```

Chair Sitdown Sequence

- Prepare a chair behind the robot and be ready to hold it as it goes into Sitdown Mode

```
ROS_DOMAIN_ID=10 ros2 service call /g1_unit_0001/hardware_modes \
g1_interface/srv/G1Modes \
'{"request_data": "sit"}'
```

Arm and Leg Commands

- Shake hand:

```
ROS_DOMAIN_ID=10 ros2 service call /g1_unit_0001/hardware_modes \
g1_interface/srv/G1Modes \
'{"request_data": "shake_hand"}'
```

- Wave hand:

```
ROS_DOMAIN_ID=10 ros2 service call /g1_unit_0001/hardware_modes \
g1_interface/srv/G1Modes \
'{"request_data": "wave_hand_with_turn"}'
```

- Set swing height to 0.12 m:

```
ROS_DOMAIN_ID=10 ros2 service call /g1_unit_0001/hardware_modes \
g1_interface/srv/G1Modes \
'{"request_data": "set_swing_height=0.12"}'
```

- Move forward at 0.3 m/s for 2 seconds:

```
ROS_DOMAIN_ID=10 ros2 service call /g1_unit_0001/hardware_modes \
g1_interface/srv/G1Modes \
'{"request_data": "set_velocity=0.3 0.0 0.0 2.0"}'
```

- Query current FSM mode:

```
ROS_DOMAIN_ID=10 ros2 service call /g1_unit_0001/hardware_modes \
g1_interface/srv/G1Modes \
'{"request_data": "get_fsm_mode"}'
```

- Enable continuous gait:

```
ROS_DOMAIN_ID=10 ros2 service call /g1_unit_0001/hardware_modes \
g1_interface/srv/G1Modes \
'{"request_data": "continous_gait=true"}'
```

4.7 | Led Selection

- set color to orange

```
ros2 service call /g1_unit_0001/hardware/led \
g1_interface/srv/G1Modes "{request_data: 'color=green'}"
ros2 service call /g1_unit_0001/hardware/led \
g1_interface/srv/G1Modes "{request_data: 'color=red'}"
ros2 service call /g1_unit_0001/hardware/led \
g1_interface/srv/G1Modes "{request_data: 'color=white'}"
ros2 service call /g1_unit_0001/hardware/led \
g1_interface/srv/G1Modes "{request_data: 'color=blue'}"
```

- set color with hex and 60

```
ros2 service call /g1_unit_0001/hardware/led \
g1_interface/srv/G1Modes "{request_data: \
'color=#00a0a0;brightness=60'}"
```

- set color with rgb()

```
ros2 service call /g1_unit_0001/hardware/led \
g1_interface/srv/G1Modes "{request_data: \
'color=rgb(128,32,200)'}"
```

- query state (local)

```
ros2 service call /g1_unit_0001/hardware/led \
g1_interface/srv/G1Modes "{request_data: 'get_brightness'}"
ros2 service call /g1_unit_0001/hardware/led \
g1_interface/srv/G1Modes "{request_data: 'get_color'}"
```

4.8 | Audio Selection

- Set Volume

```
ROS_DOMAIN_ID=10 ros2 service call /g1_unit_0001/hardware/audio\
 g1_interface/srv/G1Modes "{request_data: 'volume=80'}"
```

- Speak English

```
ROS_DOMAIN_ID=10 ros2 service call /g1_unit_0001/hardware/audio\
 g1_interface/srv/G1Modes "{request_data: 'speak=I am Bender with\
 Good Muney'}"
```

- Volume + Text

```
ROS_DOMAIN_ID=10 ros2 service call /g1_unit_0001/hardware/audio\
 g1_interface/srv/G1Modes "{request_data: 'volume=100;speak= Hello from \
 my bot shop. My name is Danny'}"
```

- Get current volume

```
ROS_DOMAIN_ID=10 ros2 service call /g1_unit_0001/hardware/audio\
 g1_interface/srv/G1Modes "{request_data: 'get_volume'}"
```

- Self Intro

```
ROS_DOMAIN_ID=10 ros2 service call /g1_unit_0001/hardware/audio\
 g1_interface/srv/G1Modes "{request_data: 'volume=100;speak= Hello from \
 my bot shop.  I am Danny, a compact humanoid robot for research and \
 education. My height is 1.32 meters  , and my weight is 35 kg. I walk\
  up to 2 meters per second, climb stairs, and keep balance. \
 Hehehe climbing stairs not really. My base model has 23 degrees of \
 freedom where as my educational + version has 43 degrees of freedom.\
  I have 3D lie daar specifically livox mid360 \
 and depth camera intel real sense d435i. \
 Battery lasts about 2 hours +'}"
```

¿ More info on these are available on the Unitree documentation.

4.9 | Arm ROS2 Control

- To start the arms (Mandatory):

```
ROS_DOMAIN_ID=10 ros2 service call\
 /g1_unit_0001/hardware/control/arm_dof7/start \
std_srvs/srv/Trigger {}
```

- To stop the arms (Mandatory):

```
ROS_DOMAIN_ID=10 ros2 service call\
 /g1_unit_0001/hardware/control/arm_dof7/stop \
std_srvs/srv/Trigger {}
```

- Wave dual arms (Will Wave on spot clear the area around the G1)

```
ROS_DOMAIN_ID=10 ros2 action send_goal\
 /g1_unit_0001/dual_arm/follow_joint_trajectory \
control_msgs/action/FollowJointTrajectory "{
  trajectory: {
    joint_names: [
      \"left_shoulder_pitch\",\"left_shoulder_roll\",
      \"left_shoulder_yaw\",
      \"left_elbow\",\"left_wrist_roll\",\"left_wrist_pitch\",
      \"left_wrist_yaw\",
      \"right_shoulder_pitch\",\"right_shoulder_roll\",
      \"right_shoulder_yaw\",
      \"right_elbow\",\"right_wrist_roll\",\"right_wrist_pitch\",
      \"right_wrist_yaw\"
    ],
    points:[
    {positions:[0.0,0.2,0.0,1.57,0.0,0.0,
    0.0,0.0,-0.2,0.0,1.57,0.0,0.0,0.0],
    time_from_start:{sec:2,nanosec:0}},
    {positions:[0.0,1.57,1.57,-1.57,0.0,
    0.0,0.0,0.0,-1.57,-1.57,-1.57,0.0,0.0,0.0],
    time_from_start:{sec:4}},
    {positions:[0.0,1.57,1.57,-1.0,0.0,
    0.0,0.0,0.0,-1.57,-1.57,-1.0,0.0,0.0,0.0],
    time_from_start:{sec:6}},
    {positions:[0.0,1.57,1.57,-1.57,0.0,
    0.0,0.0,0.0,-1.57,-1.57,-1.57,0.0,0.0,0.0],
    time_from_start:{sec:8}},
    {positions:[0.0,1.57,1.57,-1.0,0.0,
    0.0,0.0,0.0,-1.57,-1.57,-1.0,0.0,0.0,0.0],
    time_from_start:{sec:10}},
    {positions:[0.0,1.57,1.57,-1.57,0.0,
    0.0,0.0,0.0,-1.57,-1.57,-1.57,0.0,0.0,0.0],
    time_from_start:{sec:12}},
```

```
    {positions:[0.0,1.57,1.57,-1.0,0.0,
    0.0,0.0,0.0,-1.57,-1.57,-1.0,0.0,0.0,0.0],
    time_from_start:{sec:14}},
    {positions:[0.0,0.2,0.0,1.57,0.0,
    0.0,0.0,0.0,-0.2,0.0,1.57,0.0,0.0,0.0],
    time_from_start:{sec:16,nanosec:0}}
    ]


  }
}"
```

- Cross dual arms (Will Wave on spot clear the area around the G1)

```
ROS_DOMAIN_ID=10 ros2 action send_goal\
 /g1_unit_0001/dual_arm/follow_joint_trajectory \
  control_msgs/action/FollowJointTrajectory "{
  trajectory: {
    joint_names: [
      left_shoulder_pitch, left_shoulder_roll,
      left_shoulder_yaw,
      left_elbow, left_wrist_roll, left_wrist_pitch,
      left_wrist_yaw,
      right_shoulder_pitch, right_shoulder_roll,
      right_shoulder_yaw,
      right_elbow, right_wrist_roll, right_wrist_pitch,
      right_wrist_yaw
    ],
    points: [
      { positions: [-0.2,  0.2, 0.0, 0.7, 0.0, 0.0, 0.0,
                    -0.2, -0.2, 0.0, 0.7, 0.0, 0.0, 0.0],
        time_from_start: {sec: 2, nanosec: 0} },

      { positions: [-0.5, 0.2, -1.57, 0.6, 0.0, 0.0, 0.0,
                    -0.2, -0.2, 0.0,  0.7, 0.0, 0.0, 0.0],
        time_from_start: {sec: 4, nanosec: 0} },

      { positions: [-0.5, 0.2, -1.57, 0.6, 0.0, 0.0, 0.0,
                    -0.7, -0.2, 1.57, 0.6, 0.0, 0.0, 0.0],
        time_from_start: {sec: 8, nanosec: 0} },
    ]
```

```
  }
}"
```

- Remove Cross dual arms (Will Wave on spot clear the area around the G1)

```
ROS_DOMAIN_ID=10 ros2 action send_goal\
 /g1_unit_0001/dual_arm/follow_joint_trajectory \
  control_msgs/action/FollowJointTrajectory "{
  trajectory: {
    joint_names: [
      left_shoulder_pitch, left_shoulder_roll, left_shoulder_yaw,
      left_elbow, left_wrist_roll, left_wrist_pitch, left_wrist_yaw,
      right_shoulder_pitch, right_shoulder_roll, right_shoulder_yaw,
      right_elbow, right_wrist_roll, right_wrist_pitch, right_wrist_yaw
    ],
    points: [
      { positions: [-0.2,  0.2, 0.0, 0.7, 0.0, 0.0, 0.0,
                    -0.2, -0.2, 0.0, 0.7, 0.0, 0.0, 0.0],
        time_from_start: {sec: 2, nanosec: 0} },

      { positions: [0.0, 0.2, 0.0, 1.57, 0.0, 0.0, 0.0,
                    0.0, -0.2, 0.0, 1.57, 0.0, 0.0, 0.0],
        time_from_start: {sec: 4, nanosec: 0} }
    ]
  }
}"
```

- Move left arm

```
ROS_DOMAIN_ID=10 ros2 action send_goal\
 /g1_unit_0001/left_arm/follow_joint_trajectory \
  control_msgs/action/FollowJointTrajectory \
  "{trajectory: {
      joint_names: [
        left_shoulder_pitch, left_shoulder_roll, left_shoulder_yaw,
        left_elbow, left_wrist_roll, left_wrist_pitch, left_wrist_yaw
      ],
      points: [
        {positions: [-0.2, 0.2, 0.0, 0.7, 0.0, 0.0, 0.0],
         time_from_start: {sec: 2, nanosec: 0} },
      ]}}"
```

- Move right arm

```
ROS_DOMAIN_ID=10 ros2 action send_goal\
 /g1_unit_0001/right_arm/follow_joint_trajectory \
  control_msgs/action/FollowJointTrajectory \
  "{trajectory: {
      joint_names: [
        right_shoulder_pitch, right_shoulder_roll, right_shoulder_yaw,
        right_elbow, right_wrist_roll, right_wrist_pitch,
        right_wrist_yaw],
      points: [
        {positions: [0.0, -0.2, 0.0, 1.57, 0.0, 0.0, 0.0],
         time_from_start: {sec: 2, nanosec: 0}}
      ]}}"
```

5 | Navigation

5.1 | SLAM (Simultaneous Localization and Mapping)

To create a map for map-based navigation, the first step is to create a map of the environment.

- Launch

```
ros2 launch g1_navigation slam.launch.py
```

- You can begin mapping using the *teleop* at **0.2m/s** with the keyboard and/or the provided Logitech controller. Once you are satisfied with your map you can export it by running the following command:

```
ROS_DOMAIN_ID=10 ros2 run nav2_map_server map_saver_cli\
 -f /opt/mybotshop/src/mybotshop/g1_navigation/maps/custom_map
```

- Rebuild so that the maps can be found (This is required if the map name is not **map_** otherwise it will directly work)

```
cd /opt/mybotshop/ && colcon build --symlink-install
source /opt/mybotshop/install/setup.bash
```

5.2 | Odometric Navigation

To navigate without creating a map, you can simply launch the odom navi and give goals in RVIZ.

```
ros2 launch g1_navigation odom_navi.launch.py
```

5.3 | Map Navigation

This will automatically take in the map created and save from the SLAM being **map_**. If the map has been named something other, then in **g1_navigation**, the parameters have to be updated.

- Ensure the map is generated and available in the ros package (i.e. after the map is saved, you have placed and performed **colcon build**)

```
ros2 launch g1_navigation map_navi.launch.py
```

6 | Auxiliary Sensors

6.1 | Lidar - Livox Mid360

- To activate the Livox Mid360 Lidar, launch:

```
ros2 launch g1_lidar livox_mid360.launch.py
```

7 | Steamdeck

7.1 | Steamdeck - Basic Controls

1. **Power button** is on the top of the Steamdeck with the power symbol.
2. **R2** for mouse left-click
3. **L2** for mouse right-click
4. **Left trackpad** for mouse scrolling
5. **Right trackpad** for mouse movement
6. **L2+R2** for middle mouse button click
7. **- - -** (The 3 dashes is located on the top right of the Steamdeck next to the Y button) Hold for 3 seconds to *gamepadmode* and hold for 2 seconds again to go to *desktop mode*[1]

7.2 | Steamdeck Activation

> **Important:** When launching the **Joylauncher**, do not touch any of the buttons until the steamdeck displays the **robot is connected** or **not connected**.

[1]Switching to desktop mode activates the cursor for the trackpad as well

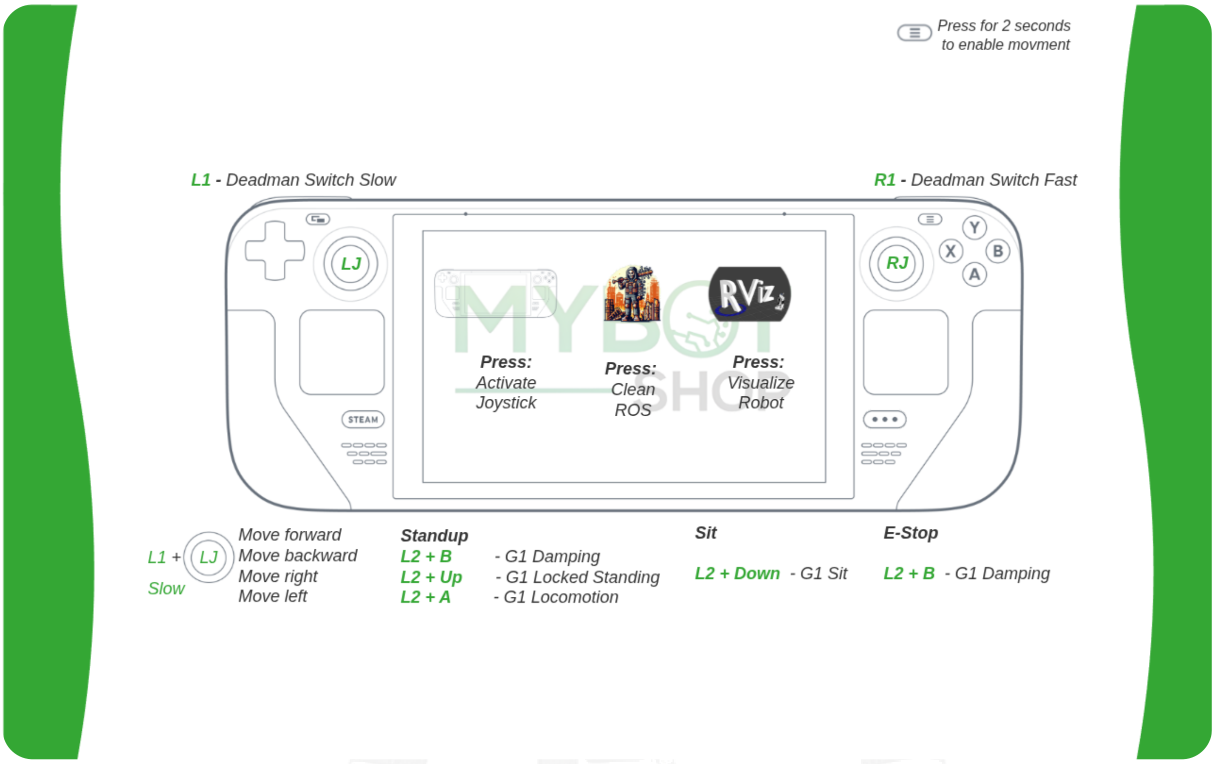


Figure 7.1: Steamdeck Shortcuts

Important: When using stand, sit and e-stop, please refer to Unitree manuals as these are immediately executed. Sit will cause the robot to immediately sit down on a chair while damping the motors and e-stop will immediately stop all motors and put them in damping mode.

1. Ensure the Steamdeck is connected to the robot's WiFi access point. Wait 20 seconds after powering up.
   - This is automatically done!
2. Ensure the robot is powered on and operational.
3. Single tap the **Joystick Launcher** on the screen to activate the controller for the robot.
   - Do not open multiple instances of the joystick launchers, as this can cause zombie nodes to spawn.
   - Hold the **- - -** for 3 seconds until you sense vibration from the controller and then hold **- - -** for 3 seconds again. This is required to re-enable the mouse in the steamdeck.
4. Robot stand up
   - Activate all the ros2 services from the webserver.
   - Once activated then press **L2+B** to put the robot in damping standing mode.
   - Then press **L2+Up** to put the robot in locked standing mode.

- Lift the robot to its feet.
- Then press **L2+A** to put the robot in locomotion standing mode.

5. Robot sit down
   - Prepare a chair behind the robot.
   - Press **L2+Down** to sit down the robot on a chair.
6. E-Stop
   - Press **L2+B** to activate e-stop.
7. Single tap the **Rviz2 Launcher** on the screen to visualize the robot.

7.3 | Steamdeck - Termination

- Single tap the **ROS2 Clean** to terminate all ROS2 instances **only** in the steamdeck (It does not effect the robot).

7.4 | Steamdeck - Robot Controls

1. **L1** is the dead man's switch for slow speed.
2. **R1** is the dead man's switch for high speed.
3. **LJ** (Left Joystick) is to move in x and y directions.

Notice: Loss of connection or (laggy connection) may lead to the robot moving and/or not stopping immediately. Caution should be taken when using the Steamdeck for the operation of the robot!

7.5 | Steamdeck - Rviz2 Controls

1. **Right trackpad** is used as a mouse
2. **Left trackpad** can be moved up and down to zoom in and zoom out
3. **L2+R2** should be held when wanting to move the orbit in the xy plane.
4. **R2** should be held to orbit around the cursors origin

8 | Installation

8.1 | NVIDIA Orin NX Developer Kit (Inbuilt G1 EDU)

Note: This repository should already be available and built on G1's Nividia board if it has been configured by the MYBOTSHOP team.

- Switch G1 id with ctrl+h

```
g1_unit_5049
```

- Switch username

```
sudo hostnamectl set-hostname g1-unit-5049
```

- Update date and time, ensure you have internet in the robot. The quickest way is using the USB WiFi stick

```
sudo timedatectl set-timezone Europe/Berlin
sudo date -s "$(wget --method=HEAD -qSO- --max-redirect=0 \
google.com 2>&1 | sed -n 's/^ *Date: *//p')"
```

- Do 'sudo apt-get update' if it fails usually removing the outdated packages resolves it. This is important

- Issues in latest g1's

```
sudo apt-key del F42ED6FBAB17C654
curl -sSL \
https://raw.githubusercontent.com/ros/rosdistro/master/ros.asc\
| sudo apt-key add -
sudo apt-get update
```

- Create directory for g1 Workspace

```
sudo mkdir /opt/mybotshop
```

```
sudo chown -R unitree:unitree /opt/mybotshop
```

- Clone the repository, copy over to the g1's PC and run the installer script

```
cd /opt/mybotshop/src/mybotshop/
sudo chmod +x g1_install.sh && ./g1_install.sh
cd /opt/mybotshop/src/mybotshop/g1_webserver
```

```
sudo chmod +x webserver_installer.sh
./webserver_installer.sh
```

- Set VNC password

```
vncpasswd ~/.vnc/passwd
Password: mybotshop
Verify: mybotshop
Would you like to enter a view-only password (y/n)? y
Password: mybotshop
Verify: mybotshop
```

- Build unitree sdk

```
cp -r /opt/mybotshop/src/third_party/unitree/04Aug2025_unitree_sdk2/\
 /opt/mybotshop
cd /opt/mybotshop/04Aug2025_unitree_sdk2/build
cmake .. && make && sudo make install
```

- Fix old CMake error for Kiss icp

```
python3 -m pip install --user --upgrade "cmake>=3.24"
```

- Build ros2 workspace

```
cd /opt/mybotshop && colcon build --symlink-install
source install/setup.bash
```

- Change the .bashrc

```
# MYBOTSHOP
source ~/cyclonedds_ws/install/setup.bash
export RMW_IMPLEMENTATION=rmw_cyclonedds_cpp
export CYCLONEDDS_URI=~/cyclonedds_ws/cyclonedds.xml
export PATH=/usr/local/cuda-11.4/bin:$PATH
export LD_LIBRARY_PATH=/usr/local/cuda-11.4/lib64:$LD_LIBRARY_PATH

source /opt/ros/foxy/setup.bash
source /opt/mybotshop/install/setup.bash
```

- Run the startup installer

```
ros2 run g1_bringup startup_installer.py
```

- Enable robots WLAN for Nvidia

```
sudo ifconfig wlan0 up && sudo nmcli radio wifi on && nmcli radio wifi
nmcli device wifi connect wifi_name password wifi_password
```

8.2 | Host

- Clone the repository into a ros2_ws and build essential items only via:

```
colcon build --symlink-install --packages-select g1_description g1_viz
source install/setup.bash
```

- Source and run rviz2

```
export G1_NS="g1_unit_5049"
ros2 launch g1_viz view_robot.launch.py
```

9 | Robotic Manipulator Safety Guidelines

When deploying robotic manipulators, it is imperative to prioritize safety procedures to mitigate risks and ensure secure operations. The following guidelines delineate key safety measures when working with robotic manipulators:

9.1 | Work Area Safety

- Maintain a clean and well-organized work area. Cluttered or inadequately lit spaces can impede the proper functioning of sensors and hinder precise manipulation.
- Avoid operating robotic manipulators in hazardous environments, such as those containing corrosive substances, extreme temperatures, or sharp objects that may damage the manipulator.
- Ensure that only authorized personnel are present in the vicinity during manipulator operation to prevent interference and ensure a safe working environment.

9.2 | Electrical Safety

- Ensure the manipulator's power system adheres to electrical safety standards. Regularly inspect and maintain power components to prevent malfunctions.
- Implement safeguards to protect the manipulator from adverse environmental conditions, such as exposure to moisture or extreme temperatures.
- Regularly inspect power cables and connections, promptly replacing damaged components to minimize the risk of electrical issues.

9.3 | Manipulation Safety

- Implement collision detection systems to prevent unintended contact with objects, humans, or other equipment during manipulation tasks.
- Define and enforce safety zones around the manipulator's workspace to minimize the risk of unintended interactions with personnel or other objects.
- Regularly calibrate and test the manipulator's sensors and systems to ensure precise and reliable performance during manipulation tasks.

9.4 | Emergency Response

- Install an emergency stop mechanism to swiftly halt manipulator operation in unforeseen circumstances or emergencies.
- Clearly mark and communicate emergency stop locations within the manipulator's operational area.

- Conduct regular emergency response drills to ensure personnel are familiar with procedures for handling unexpected situations during manipulator operation.

9.5 | Data Security and Privacy

- Implement robust cybersecurity measures to safeguard the manipulator's control systems and data from unauthorized access or manipulation.
- Ensure compliance with privacy regulations when collecting, storing, or transmitting data captured by the manipulator's sensors.

9.6 | Human Interaction Safety

- Integrate sensors and communication systems to detect and respond to the presence of humans in the manipulator's vicinity.
- Clearly communicate the manipulator's operational status and intentions using visual and audible signals to alert nearby individuals.
- Establish protocols for safe human-robot collaboration, particularly in shared workspaces where manipulators are in operation.

9.7 | Residual Risks

Despite the implementation of safety measures, certain residual risks may persist. These include:

- Impairment of sensor functionality.
- Risk of collisions during complex manipulation tasks.
- Cybersecurity vulnerabilities.
- Unintended human interactions due to unforeseen circumstances.

Robotic manipulators are sophisticated technologies that demand correct usage to avoid accidents and ensure a secure environment. Please adhere to proper procedures diligently, prioritizing both precision and safety.

10 | Autonomous Mobile Robot Safety Guidelines

When deploying autonomous mobile robots, prioritizing safety procedures is essential to prevent accidents and ensure secure operations. The following guidelines outline key safety measures when working with an autonomous mobile robot:

10.1 | Work Area Safety

- Maintain a clean and well-lit work area. Cluttered or poorly illuminated spaces can impede the proper functioning of sensors and navigation systems.
- Avoid operating autonomous mobile robots in explosive atmospheres, such as areas with flammable liquids, gases, or dust. The robot's components may pose a risk in such environments.
- Keep bystanders and unauthorized personnel at a safe distance during robot operation to prevent interference with autonomous navigation.

10.2 | Electrical Safety

- Ensure the robot's power system adheres to electrical safety standards. Regularly inspect and maintain power components to prevent malfunctions.
- Implement mechanisms to protect the robot from adverse weather conditions, such as rain or wet environments.
- Regularly inspect power cables and connections and replace damaged components promptly to minimize the risk of electrical issues.

10.3 | Navigation Safety

- Implement obstacle detection and avoidance systems to prevent collisions with objects, people, or other robots.
- Define and enforce safety zones within the robot's operational area to minimize the risk of unintended interactions with personnel or other equipment.
- Regularly calibrate and test the robot's navigation sensors and systems to ensure accurate and reliable performance.

10.4 | Emergency Response

- Install an emergency stop mechanism to quickly halt the robot's operation in case of unforeseen circumstances or emergencies.
- Clearly mark and communicate emergency stop locations throughout the robot's operational area.

- Conduct regular emergency response drills to ensure personnel are familiar with procedures for handling unexpected situations.

10.5 | Data Security and Privacy

- Implement robust cybersecurity measures to protect the robot's control systems and data from unauthorized access or manipulation.
- Ensure compliance with privacy regulations when collecting, storing, or transmitting data captured by the robot's sensors.

10.6 | Human Interaction Safety

- Integrate sensors and communication systems to detect and respond to the presence of humans in the robot's vicinity.
- Clearly communicate the robot's operational status and intentions using visual and audible signals to alert nearby individuals.
- Establish protocols for safe human-robot collaboration, especially in shared workspaces.

10.7 | Residual Risks

Despite the implementation of safety measures, certain residual risks may persist. These include:

- Impairment of sensor functionality.
- Risk of collisions in crowded or dynamic environments.
- Cybersecurity vulnerabilities.
- Unintended human interactions due to unforeseen circumstances.

Autonomous Mobile Robots (AMR) are advanced technologies that require correct usage to avoid accidents and ensure a secure environment. Learn and follow the proper procedures diligently; prioritize both quality and safety.

11 | Support

MYBOTSHOP QUADRUPED

In case of any issues, help can be asked in:

1. MYBOTSHOP FORUMS (forum.mybotshop.de)
2. MYBOTSHOP SUPPORT (support@mybotshop.de)

Further information on different robots can be viewed in:

1. MYBOTSHOP DOCS (docs.mybotshop.de)
2. QUADRUPED DOCS (docs.quadruped.de)

12 | Tutorials

A collection of open-source training material to familiarize oneself with the ROS2 environment.

12.1 | Ubuntu Tutorials

1. Ubuntu Documentation - Official Ubuntu Documentation.
   - *help.ubuntu.com/*
2. Ubuntu ROS2 Documentation - Official Ubuntu ROS2 Documentation.
   - *ubuntu.com/tutorials/getting − started − with − ros − 2#1 − overview*

12.2 | ROS2 — Tutorials

1. ROS2 Tutorial List - ROS2 Tutorial List.
   - *docs.ros.org/en/foxy/Tutorials.html*
2. ROS2 Basics Training - ROS2 Basics Training.
   - *industrial−training−master.readthedocs.io/en/melodic/_source/session7/ROS2−Basics.html*
3. ROS2 foxy Wiki - Official ROS2 documentation and tutorials.
   - *docs.ros.org/en/foxy/index.html*
4. ROS Navigation2 Wiki - Official ROS navigation documentation and tutorials.
   - *navigation.ros.org/*
5. ROS2 Manipulation Wiki - Official ROS2 manipulation documentation and tutorials.
   - *moveit.picknik.ai/main/index.html*
6. ROS2 Discourse - Official ROS2 discussion platform.
   - *discourse.ros.org/c/ng − ros/25*
7. ROS2 Courses - Official ROS2 courses.

12.3 | ROS2 — Additional Resources

1. ROS2 Installation Guide - Documentation.
2. ROS2 Pertinent Repositories - GitHub Repository.
3. ROS2 Concepts - Documentation.
4. ROS2 How to guides - Documentation.

5. ROS2 for Beginners - The Construct - YouTube playlist.

6. ROS2 foxy Crash Course - YouTube playlist.

7. ROS2 Basic Learning - YouTube playlist.

8. ROS Tutorials - ETH Zurich - YouTube playlist. [2]

9. ROS Workshop - YouTube Video.

[2]It is in ROS1, however, provides basic understanding of the ROS paradigm.

13 | ROS — MYBOTSHOP/QUADRUPED GmbH Trainings

MYBOTSHOP/QUADRUPED GmbH offers paid training as well for basic ROS training as well as for transitional training. An example of the training schedule is provided below.

13.1 | ROS1 to ROS2 δ

The provided course would be a general concise transitional course from ROS1 to ROS2 foxy. The training would be geared towards basic practical implementation and deployment of ROS2 foxy with those who are **well** familiar with ROS1. The average time for the training is 2—3 days with a focus more on practicality and less on theory.

ROS1 to ROS2 δ - Pre-requisities

- ROS1 (Kinetic/Melodic/Noetic)
  - Working experience and concepts
  - Control concepts
  - Navigation stack working
  - URDF
  - Transforms (TFs)
  - Nodes — Nodelets
  - Topics —Parameters — Services — Msgs — Action Service
- Ubuntu 22.04 installed
- ROS2 Humble Installed
- Internet Connection (Package installation)

13.1.1 | ROS1 to ROS2 δ - List of Deliverables

1. Presentation slides
2. Simulation-based tasks
3. ROS2 Template Packages

ROS1 to ROS2 δ — Basics — Day 1

- **Introduction to ROS2 Humble**
  - Reason of ROS2 (T)
  - Key differences (T)

- □ New architecture (T)
- □ Why DDS? and its Robustness and longevity! (T)
- □ ROS2 Control (Brief Overview) (T)
- □ ROS2 Package Build (T)
- □ ROS Bridge (T)
- □ ROS2 Simulation (P)
- □ ROS2 Create Package (P)

■ **ROS2 Nodes — Lifecycle nodes — Components — Topics**

- □ ROS2 nodes (T)
- □ ROS2 launch files (T)
- □ Create ROS2 Node (P)
- □ Create ROS2 Launch file (P)
- □ ROS2 Sensor Fusion (P)

■ **ROS2 Parameters — Msgs — Servcices — Action-server**

- □ Item Differences (T)
- □ ROS2 Parameters (T)
- □ Create ROS2 Parameter Loading (P)

ROS1 to ROS2 δ — Autonomous Navigation — Day 2

■ **ROS2 Navstack**

- □ Differences (T)
- □ Setup (P)
- □ Mapping (P)
- □ Localization (P)
- □ Planner Tuning (P)
- □ Custom Robot Integration (P)
- □ Debugging (P)
- □ Tools Usage (P)

■ **Personal Experience**

- □ Issues faced
- □ Debugging
- □ Tools Usage

13.2 | Additional Trainings

In addition to the ROS1 to ROS2 δ trainings, we have trainings for:

1. Ubuntu 20 — ROS1 Noetic (1—2 days)
2. Ubuntu 22 — ROS2 Humble (1—2 days)
3. Robot operational training (Specialized Software Solution Projects) (1—2 days)

Resources and Contact Information

For your convenience, we have compiled the most important resources related to the robot platform. Each link can be accessed directly or by scanning the corresponding QR code.

Documentation

Our official documentation contains detailed setup instructions, usage guidelines, and troubleshooting information. Access the latest version here: https://www.docs.quadruped.de/index.html

Forum

Connect with other developers and researchers in our community forum. Here you can ask questions, share experiences, and stay up to date: https://forum.mybotshop.de/

Rentals

If you would like to test or temporarily use a robot, we also offer rental services. Browse our available systems here: https://www.mybotshop.de/Vermietung_3

Newsletter

Stay informed about product updates, events, and special offers by subscribing to our newsletter: https://www.mybotshop.de/Newsletter_1

Contact Information

- Vertrieb / Sales: +49 (0) 2271 588 93 10 — `sales@mybotshop.de`
- Support / After Sales: +49 (0) 2271 588 93 20 — `support@mybotshop.de`
- Buchhaltung / Accounting: +49 (0) 2271 588 93 80 — `accounting@mybotshop.de`
- Allgemeine Anfragen / General inquiries: +49 (0) 2271 588 93 0 — `info@mybotshop.de`

QUADRUPED ROBOTICS

RMA SUPPORT

How to Raise a Return Merchandise Authorization (RMA) Case

At QUADRUPED Robotics, we are committed to providing excellent support to resolve your issues promptly. If you encounter any problems with our products, please follow the steps below to raise a Return Merchandise Authorization (RMA) case:

Step 1: Open a Topic in Our Forum

1. Visit MYBOTSHOP Forum
2. Open a new topic describing your issue in detail. Our support team and community members will attempt to assist you in resolving the problem remotely.

Step 2: Contact Support for RMA Number

If the issue cannot be resolved remotely through our forum:

1. Email us at support@quadruped.de
2. Provide a detailed description of the issue and include any troubleshooting steps already taken.
3. Our support team will evaluate your case and, if necessary, issue an RMA number along with a return address or a shipping label.

Important Shipping Information

Please note:

- All goods must be sent back to the following address:
    - **QUADRUPED Robotics GmbH**
    - **c/o MYBOTSHOP GmbH**
    - **Willy-Messerschmitt-Strasse 1250126**
    - **Bergheim Germany**
- Do not ship goods without obtaining an RMA number. Parcels sent without an RMA number will be declined and returned to the sender.
- Do **not ship goods** to QUADRUPED Robotics GmbH's Office in Leverkusen!

Help Resources

In case of any issues, help can be taken from any of the following forums/resources:

- MYBOTSHOP Forum: The forum is actively monitored by support teams at MYBOTSHOP GmbH.
- Github Issues: Issues can be reported in the repository.
- Online QA Sites: Questions can be asked on any of the QA sites, like StackOverflow and Answers ROS.

MYBOT SHOP

QUADRUPED ROBOTICS