MYBOTSHOP

QUADRUPED ROBOTICS

Unitree H1

User Manual

CONTACT

info@quadruped.de

support@quadruped.de

quadruped.de

Contents

1 Attention

Attention: Read this document carefully. In case of doubt, it is essential to consult specialists, experts, or the manufacturer (Unitree Robotics) or distributor (MYBOTSHOP / QUADRUPED Robotics) of the assemblies used. The robot must not be put into operation before clarification. Contact information is available in the Getting Help & RMA chapter.

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

Warning: H1 is a full-size, human-weight bipedal humanoid. Both the Hang & Power On (gantry) and Sit & Power On (chair) startup procedures require an operator to physically support the robot — via the protective frame rope, or the rear holding bracket — throughout the stand-up transition to prevent it tipping. Never attempt a startup rig alone without the required support in place.

Caution: Monitor both battery levels continuously once powered on — if either battery depletes, the robot falls. H1-2 does not support the Sit & Power On/Off rig or Dance Mode; do not attempt these actions on an H1-2 unit. H1 is not dustproof or waterproof.

Starting the ROS 2 SDK before H1 has confirmed entry into Develop Mode (verify with L2+A) can cause command conflicts and make the robot jitter unexpectedly — always confirm Develop Mode first.

2 Disclaimer

The H1 robot documented here is sold and supported by *MYBOTSHOP / QUADRUPED Robotics* as the German distributor for Unitree Robotics. Basic ROS 2 understanding is required to operate the software stack covered in this manual — if you are not familiar with ROS 2, we recommend checking the ROS 2 documentation first.

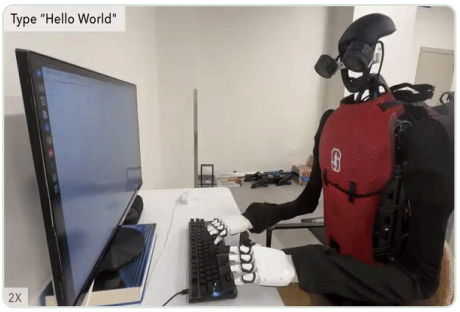
The client acknowledges and agrees that any information or materials provided by MYBOTSHOP / QUADRUPED Robotics are for R&D and development purposes only. Any services are provided "AS IS" and without any representation or warranty of any kind, express or implied, including but not limited to any warranty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

This product is intended for civilian use only. Users should avoid making dangerous modifications or using the robot in hazardous ways. This manual documents both the base H1 (19-DOF body) and the upgraded H1-2 (27-DOF body) — always confirm which variant you have before following a variant-specific procedure. It also documents both ROS 2 Humble (the primary, currently-shipping install path) and legacy references to ROS 2 Foxy on Ubuntu 20.04 found in the source documentation — confirm which distribution is installed on your unit before following distro-specific steps. For terms, policies and compliance with local laws and regulations, refer to the Unitree Robotics website and the RMA/support process in the Getting Help chapter.

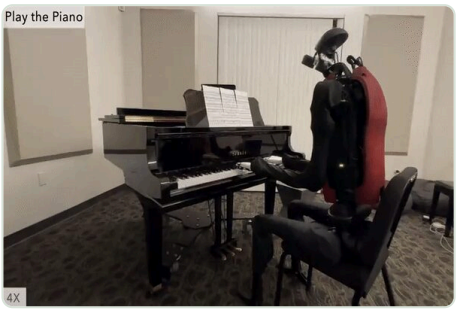
MYBOTSHOP / QUADRUPED Robotics shall not be liable for any damages, including but not limited to direct, indirect, special, incidental, or consequential damages, arising out of or in connection with the use or inability to use the information or materials in this document. This limitation on liability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

3 About the H1

H1 Robot Tutorial

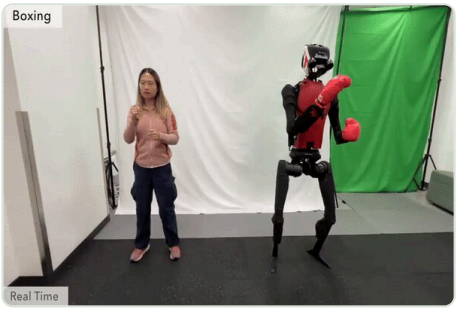


Humanoid Robot H1 Typing



Humanoid Robot H1 Playing Piano

Humanoid Robot H1 Playing Tennis



Humanoid Robot H1 Boxing

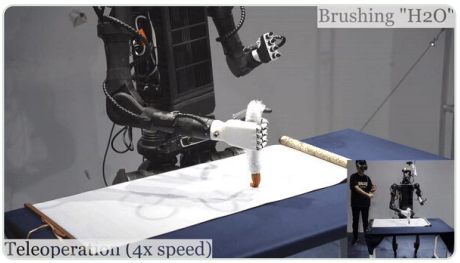
Important: Please note that the content showcased here represents the individual achievements of Humanoid AI and Omni Human-to-Humanoid , and there has been no collaboration between them and our organization. The videos and gifs included on this page are intended strictly for educational and demonstrative purposes.

The provided tutorial will assist you with setting up and operating your H1 bi-pedal robot. The tutorial topics are listed in the left column and presented in the suggested reading order.

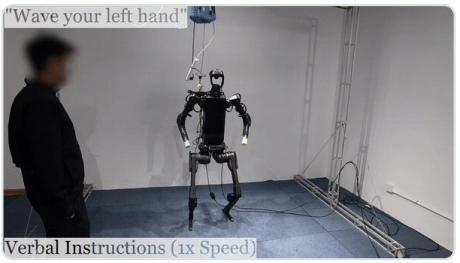
Insights from Leading Researchers

We are excited to highlight the innovative work presented on Humanoid AI and Omni Human-to-Humanoid . The researchers on Humanoid AI have developed a comprehensive system enabling humanoid robots to learn motion and autonomous skills from human data. Their approach involves training low-level policies in simulation through reinforcement learning using extensive human motion datasets. These policies allow humanoid robots to shadow human movements in real-time using only an RGB camera. The collected data is then used for supervised behavior cloning, enabling robots to autonomously perform various tasks such as wearing shoes, unloading objects, and interacting with other robots with impressive success rates. On the other hand, the work on Omni Human-to-Humanoid has resulted in OmniH2O, a versatile system for whole-body humanoid teleoperation and autonomy. By using kinematic poses as a universal control interface, OmniH2O allows humans to control humanoids through VR, verbal instructions, and RGB

cameras. They have also integrated advanced models like GPT-4 to enable full autonomy. Their RL-based sim-to-real pipeline and the release of the OmniH2O-6 dataset highlight their contributions to humanoid skill learning.

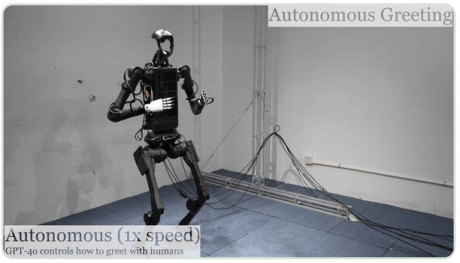


Humanoid Robot H1 Brush Teleoperation



Humanoid Robot H1 Verbal Instructions

Humanoid Robot H1 Squat & Pick Basket



Humanoid Robot H1 GPT4o-Autonomous Greeting

Warm Tips for Unitree Humanoid Robot Development and Usage

To align with the public's preference for natural human-like movements, it's important to adhere to these guidelines, particularly when filming humanoid robots:

- Strive for upright or nearly upright knee joint positions when developing leg movement programs.
- Reduce step frequency and minimize stationary stepping.
- Keep the feet slightly close together during walking to maintain natural movement.

These tips are intended to enhance the authenticity of humanoid robot movements, especially in video documentation.

4 Powering-On & Network Setup

Powering-on

_images/h1_power_on.jpg

Battery Power Button : Use the buttons on both batteries to power on the robot.

To power on the H1 robot, press the power button on both batteries simultaneously once, and then press and hold for 3 seconds to start the robot.

The battery compartments are located beneath the left arm of the robot.

Similarly, to turn off the robot, press the power button on both batteries simultaneously once, and then press and hold for 3 seconds to turn off the robot.

Note: Please be cautious with battery consumption. Continuously monitor the battery levels. If any battery is depleted, the robot will fall.

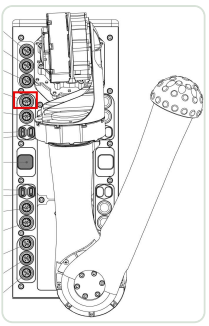
Network Interface

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

This procedure should be followed after successfully setting up and pairing with the H1 robot. Additionally, all of the H1's functionality should be verified through the accompanying app before proceeding.

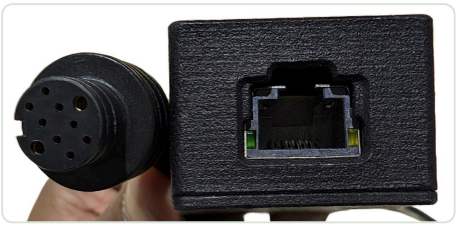
The Ethernet port , located beneath the right arm of the H1 (the fourth port from the top), can be used to establish communication via LAN. However, a special adapter cable is required, as a standard Ethernet cable cannot be inserted directly into the port.

Port Location on H1 (Real Robot)



Port Location (Layout Diagram)

Ethernet Adapter



Ethernet Adapter Cable Ends

Network Setup

To configure the robot's network for the first time, you need to connect using a LAN cable.

To create a static connection on your PC (not on the robot), follow these steps in Ubuntu :

1. Go to Settings → Network , then click on the + button to create a new connection.
2. In the IPv4 settings, change the connection type to Manual .
3. Set the Address to 192.168.123.51 and the Netmask to 24 .
4. Click Save , then restart your network.

After successfully connecting, check the host's local IP by typing the following command in the host PC's terminal:

```
ifconfig
```

Now, ping the robot by entering:

```
ping 192.168.123.162
```

```
ping 192.168.123.164
```

To access the robot via SSH, use the following command:

```
ssh -X unitree@192.168.123.162
```

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

The default password is:

```
Unitree0408
```

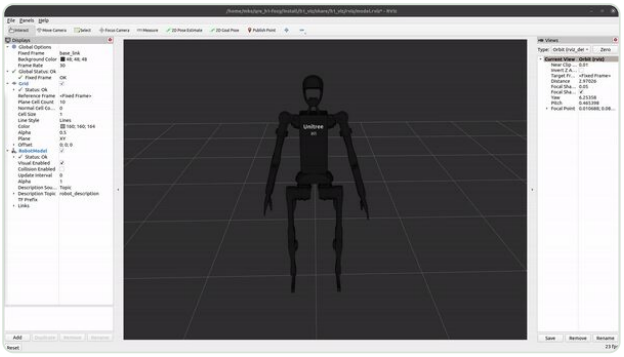
IP Addresses

IP Address	Device	Username	Password
192.168.123.161	H1 MCU	x	x
192.168.123.164	H1 Auxiliary PC 2	unitree	Unitree0408
192.168.123.164:9000	H1 Webserver MBS	admin	mybotshop
192.168.123.165	H1 Nvidia BagPack	administrator	mybotshop
192.168.123.120	Mid360 Lidar	x	x
192.168.123.20	Ouster Lidar	x	x

Note: Sometimes other networks can cause disruptions when connecting to the H1. It is best to have only your connection to the robot active and disable all others.

5 Manuals & Operation Guide

Manuals



_images/h1_hello.gif

The following text contains hyperlinks. Click on them to be redirected to the websites.

- Unitree Documentation
- Isaac Lab documentation
- Issac Lab H1 Deployment

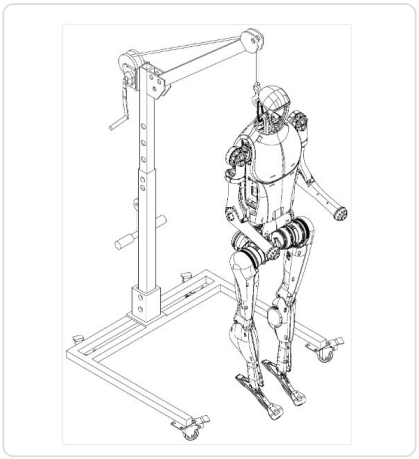
Mobile Apps

- H1 iOS
- H1 Android

Boot Process

Hang & Power On

1. Place H1 on the ground and secure it with a rope tied in a dead knot at H1's shoulder latch.
2. Hang the rope on the clasp of the protective frame to support H1.
3. Raise H1 along with the protective frame to prepare for startup.

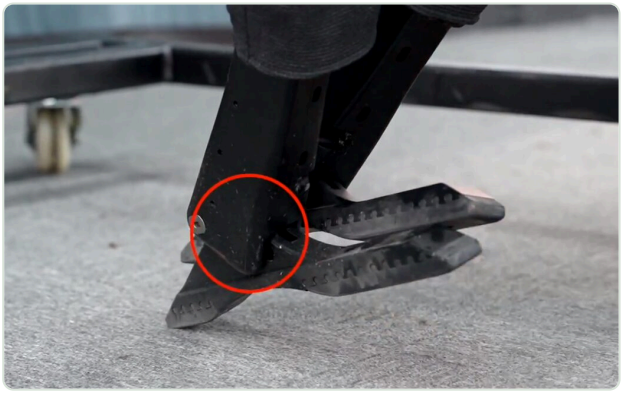


_images/h1_setup_start.png

1. Insert two batteries into the battery compartment with the key-forward orientation.
2. Turn the arm inward until it reaches its limit and then lower it vertically.

_images/h1_arm_position.gif

1. Ensure that the feet are cocked up to their limit.



_images/h1_feet_up.png

1. Maintain the position as depicted in the image below:

_images/h1_damp_position.png

1. Short press and long press both batteries simultaneously to power them up.
2. Wait until you hear the sound indicating that the ankle has reached its limit, and then wait an additional 30 seconds.
3. Press L2+B to unlock damping, then press L2+UP to enter ready mode. At this point, the feet will spread out, and the arms will be at the sides of the waist.

_images/h1_startup_finish.gif

1. Lower the H1 with the protective frame until H1's feet touch the ground and the fuselage remains stable.
2. Press R2+X again to activate Sport Mode, and H1 will start marching in place.
3. Press START to switch to standing mode, then lower the hook and remove the rope.

H1 Quick Start Video

Hang & Power Off

1. Reattach the rope to the protection hook once H1 remains standing.
2. Raise the hook until there is tension on the rope supporting H1.
3. Press L2+B to enter damping mode.
4. Simultaneously short press and long press both batteries to power off the double battery simultaneously, completing the shutdown process.

Attention: The H1-2 currently does not support sitting down . Do not attempt these actions.

Sit & Power On

> **Note:** The H1 has a comparatively wide hip structure. Ensure the seating surface is wide and stable before performing the sit-down action.

Step 1: Power On

Place the H1 securely on a chair and install the battery. Adjust the arm and ankle joints to a natural position, then turn the power on.

When the ankle joints reach their mechanical limit and produce an audible click, the initialization sequence is complete.

Step 2: Assisted Stand-Up

Press L2 + B on the remote control to switch the robot into damping mode , then press L2 + UP to command the stand-up motion.

> **Important:** The H1 requires active assistance while standing up . Use the rear holding bracket to support the robot from behind and prevent backward tipping throughout the motion.

Once fully upright and stable, press R2 + X to enter walking mode . After confirming balance, you may gradually release support.

Sit & Power Off

1. Position the robot in front of the chair.
2. Press L2 + LEFT to initiate the sit-down movement.
3. The robot will step backward and lower into a seated position. Continue holding and supporting the robot using the rear bracket to ensure a controlled descent and prevent loss of balance.

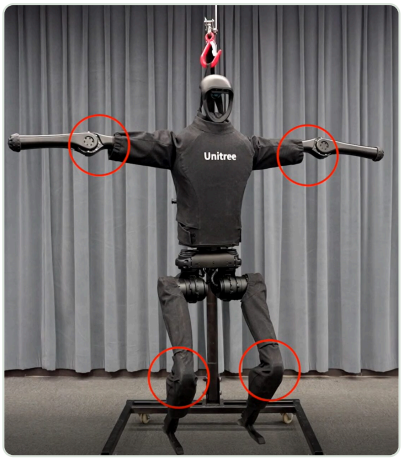
After seated, the robot can be powered off or commanded to stand again if needed.

> **Note:** Emergency Stop: If the H1 enters an abnormal or unsafe state, press L2 + B . The robot will switch to damping mode and lower itself to the ground slowly.

Develop Mode

To enter Develop Mode , ensure that H1 is in the suspension and damping state . Then, press the L2 + R2 key combination on the remote control simultaneously, and H1 enters the Develop mode.. Once activated:

- Press L2 + A → H1 enters position mode and assumes a specific diagnostic posture .



_images/h1_diagnostic_pose.png

- Press L2 + B → H1 returns to the damping state .

_images/h1_damp_pose.png

This process helps confirm whether H1 has successfully entered Develop Mode and can also be used for hardware troubleshooting . Once in Develop Mode , you can begin using the SDK for development and debugging.

> **Note:** In the current system version, the built-in motion control program starts automatically when H1 is powered on. Even if the remote control is not operated, the system continuously sends periodic instructions with a speed of 0 . If you attempt SDK-based development without first entering Develop Mode , instruction conflicts may occur, potentially causing H1 to jitter . To prevent this, always ensure that H1 is in Develop Mode before using the SDK to stop the motion control program from sending commands. You can verify this by pressing L2 + A . If the robot's response does not match the expected behavior shown in the instructional video, press L2 + R2 multiple times to ensure entry into Develop Mode .

Basic Operation

Once The H1 is Powered On and in Sports Mode:

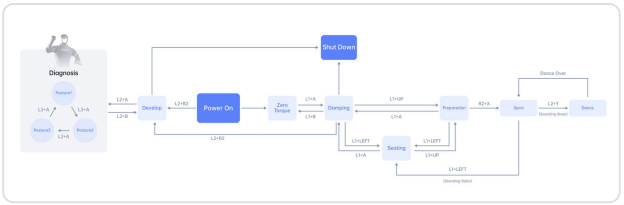
1. Press START to switch to continuous walking mode

2. Push the left joystick forward to move H1 forward.
3. Press START again to stop H1 and make it stand in place.
4. Press START once more to return H1 to continuous walking mode.
5. Push the left joystick back to make H1 walk backward.
6. Push the right stick to the right to rotate H1 in that direction.
7. Push the right stick to the left to rotate H1 in the opposite direction.
8. Press B to increase the height of the leg lift, and press A to lower the leg lift.
9. Reattach the rope to the protection hook once H1 remains standing.
10. Raise the hook until there is tension on the rope supporting H1.
11. Press L2+B to enter damping mode; at this point, you can safely shut down or enter debug mode.

Operation Guide

The following table describes the different operational modes of the robot:

Concept	Description
Zero Torque Mode	All motors of the robot stop active motion, and there is no damping feeling when swinging.
Damping Mode	All motors of the robot stop active motion, but there is a clear damping feeling when swinging. This mode allows the robot to enter Ready Mode .
Seating Mode	The robot will slowly transition into a seated position within 5 seconds .
Ready Mode	The robot will slowly swing into the preparatory posture before entering Motion Mode , within 5 seconds .
Motion Mode	A mode in which the robot can be controlled to move via a remote control .
Standing Mode	When the joystick input is zero , the robot stops stepping and remains in the standing state . When the joystick input is non-zero or the robot is disturbed and cannot maintain balance, it will start taking steps to regain stability.
Dance Mode	A mode for whole-body dynamic coordinated dancing .
Debug Mode	This mode is used for low-level development . When using the SDK for development and debugging , ensure that the robot is in Debug Mode to stop the motion control program from sending commands. This prevents potential command conflicts . You can press L2 + A to confirm whether the robot has successfully entered Debug Mode .



_images/h1_mode_switch.jpg

Attention: When entering Seating Mode , it is necessary to press against the robot's back . When exiting this mode, push the robot back to transition out of it. Note: The H1-2 model has only one diagnostic mode and does not support Seating Mode .

The following table provides an overview of button commands for different modes:

Mode	Button	Description
General	L2 + B	Enter Damping Mode
L2 + R2	Enter Debug Mode	
Damping Mode	L2 + UP	Enter Ready Mode
L2 + Y	Enter Zero Torque Mode	
Motion Mode	R2 + X	Enter Motion Mode
Left Joystick Right Joystick X Y A B START L1 + Y SELECT + A SELECT + Y L2 + LEFT	Translational velocity command (v_x, v_y) Yaw angular velocity command ($omega_{yaw}$) Decrease standing height Increase standing height Decrease leg lifting height Increase leg lifting height Adjust posture, take a few steps in place, then stop Enter Dance Mode and return to Motion Mode Handshake Wave Hand Sit Down	
Debug Mode	L2 + A	Perform position control for diagnostics
L2 + B	Stop diagnostics, enter Damping State	

Attention: The H1-2 currently does not support sitting down or dancing . Do not attempt these actions.

6 ROS 2 Quick Reference (H1-2 E)

H1-2 E

H1 Overview

ROS2 drivers for the H1 platform include:

- Domain Bridge
- Joint State publisher
- Robot Description
- Arm and Leg High-Level Control
- IMU Publisher
- Sensor Fusion
- Twist Mux
- Depth Camera driver (D435i)
- Lidar drivers (Livox / Ouster)

Drivers are managed via startup services or the webserver interface.

Auto Drivers Startup

Check driver status:

```
sudo service h1-platform status
sudo service h1-webserver status
```

Start drivers manually:

```
sudo service h1-platform start
```

Restart if required:

```
sudo service h1-platform restart
```

Modify startup jobs in:

h1_bringup/scripts/startup_installer.py

Apply changes:

```
ros2 run h1_bringup startup_installer.py
```

RViz2

Visualize robot state:

```
ros2 launch h1_viz view_robot.launch.py
```

Tele-Operation

Keyboard teleop:

```
ROS_DOMAIN_ID=10 ros2 run teleop_twist_keyboard teleop_twist_keyboard \
--ros-args --remap cmd_vel:=/h1_unit_0001/hardware/cmd_vel
```

Core Launch Files

Useful launch files:

```
ros2 launch h1_platform highlevel_ros.launch.py
ros2 launch h1_platform domain_bridge.launch.py
ros2 launch h1_platform h1_2_state_publisher.launch.py
ros2 launch h1_webserver webserver.launch.py
ros2 launch h1_control twistmux.launch.py
ros2 launch h1_lidar ouster.launch.py
ros2 launch h1_depth_camera realsense.launch.py
```

Arms (7-DoF):

```
ros2 launch h1_description h1_2_description.launch.py
ros2 launch h1_platform h1_2_arm_record.launch.py
```

SDK examples:

```
./h1_loco_client --network_interface=eth0 --shake_hand
```

Modes Selection

Posture & State Commands

- damp — all motors damping mode
- start — begin locomotion
- squat — transition to squat
- sit — sit down
- stand_up — stand up
- zero_torque — disable torque
- stop_move — stop motion

- high_stand — high posture
- low_stand — low posture
- balance_stand — active balance stand
- shake_hand — perform handshake

Setter Commands

- set_fsm_id=<id>
- set_balance_mode=<0/1>
- set_swing_height=<m>
- set_stand_height=<m>
- set_velocity="vx vy w dur"
- move="vx vy w"
- set_task_id=<id>
- set_speed_mode=<mode>

Getter Commands

- get_fsm_id
- get_fsm_mode
- get_balance_mode
- get_swing_height
- get_stand_height
- get_phase

Example: Damp Mode

```
ROS_DOMAIN_ID=10 ros2 service call \
/h1_unit_0001/hardware_modes h1_interface/srv/H1Modes \
'{"request_data": "damp"}'
```

Dual Arm Control (H1-2)

Launch server:

```
ros2 launch h1_platform h1_2_arm_server.launch.py
```

Configuration file:

h1_platform/config/h1_2_dual_arm_controller.yaml

Example T-pose (trajectory):

```
ros2 action send_goal /h1_unit_1789/dual_arm/follow_joint_trajectory \
control_msgs/action/FollowJointTrajectory "{trajectory: {...}}"
```

(Full trajectories preserved in source document.)

Dual Arm Record (Experimental)

Launch:

```
ros2 launch h1_platform h1_2_arm_record.launch.py
```

Start recording:

```
ros2 service call /h1_unit_284/dual_arm/record std_srvs/srv/SetBool "data: true"
```

Stop:

```
ros2 service call /h1_unit_284/dual_arm/record std_srvs/srv/SetBool "data: false"
```

Playback latest:

```
ros2 service call /h1_unit_284/dual_arm/playback h1_interface/srv/H1Modes \
"request_data: ''"
```

Playback specific:

```
ros2 service call /h1_unit_284/dual_arm/playback h1_interface/srv/H1Modes \
"request_data: '/opt/mybotshop/h1_2_trajectories/<file>.csv'"
```

Additional ROS2 Packages

Located under:

/opt/mybotshop/src/mybotshop/

Navigation (In-Development)

SLAM

Launch SLAM:

```
ros2 launch h1_navigation slam.launch.py
```

Save map:

```
ROS_DOMAIN_ID=10 ros2 run nav2_map_server map_saver_cli \
-f /opt/mybotshop/src/mybotshop/h1_navigation/maps/custom_map
```

Rebuild:

```
colcon build --symlink-install
source install/setup.bash
```

Odometric Navigation

```
ros2 launch h1_navigation odom_navi.launch.py
```

Map Navigation

```
ros2 launch h1_navigation map_navi.launch.py
```

Auxiliary Sensors

Intel Realsense D435i

```
ros2 launch h1_depth_camera realsense.launch.py
```

Config file: h1_depth_camera/launch/realsense.launch.py

ZED2i

```
ros2 launch h1_depth_camera zed.launch.py
```

Config: params_zed_stereo.yaml

Ouster Lidar

```
ros2 launch h1_lidar ouster.launch.py
```

Config: ouster_params.yaml

GPS – Emlid M2

```
ros2 launch h1_gps emlid_m2.launch.py
```

Config: h1_gps/config/emlid_m2.yaml

Simulation

Gazebo Fortress

```
ros2 launch h1_gazebo h1_2_fortress_simulation.launch.py
```

Isaac Sim

See official Unitree / MYBOTSHOP guides.

7 ROS 2 Pre-requisites & Full Reference

Pre-requisites

- Power On Powering-on .
- Network Interface Network Interface .
- Installation Robot Installation .

Attention: Read this document carefully! In case of doubt, consult specialists, experts, or manufacturers of the assemblies used. The robot must not be operated before all uncertainties are clarified. For further assistance, refer to the guides below or contact a specialist. Important : Contact information is available in the RMA/ Support Section. Warning : Do not allow untrained individuals or those unfamiliar with robotics or these instructions to operate the robot. Improper use of robots is dangerous.

This guide will walk you through the process of setting up and configuring the Unitree H1 robot, including network configurations, SSH access, internet setup, software installation and various operational controls. Follow the steps below carefully to ensure proper operation.

Autostart

All ROS2 drivers for the H1 developed by MYBOTSHOP operate with the environment variable ROS_DOMAIN_ID=10 . To start the custom Mybotshop ROS2 drivers for the H1, run the following command:

Note: Please ensure that the robot's arms are straight down when you power on the robot. If the arms are in a different orientation, it may affect the ROS2 control operations.

```
ros2 launch h1_bringup system.launch.py
```

This command starts the driver for operating the H1 in high-level mode for the legs and arms.

Important: The arms will initialize and move so that the forearms are facing forward.

The H1 ROS2 driver includes the following components:

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

- Sensor Fusion
- Twist Mux
- Logitech F710 Control
- D435i Depth Camera Driver
- Livox Mid360 LiDAR Driver
- 2D Scanner

All ROS2 drivers for the H1 developed by MYBOTSHOP operate with the environment variable ROS_DOMAIN_ID=10 . To start the custom Mybotshop ROS2 drivers for the H1, run the following command:

> **Note:** Please ensure that the robot's arms are straight down when you power on the robot. If the arms are in a different orientation, it may affect the ROS2 control operations.

```
ros2 launch h1_bringup system.launch.py
```

This command starts the driver for operating the H1 in high-level mode for the legs and arms.

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

The H1 ROS2 driver includes the following components:

- Domain Bridge
- Joint States
- Robot Description
- Arm Control
- Leg Control (High-level)
- Inertial Measurement Unit Publisher
- Sensor Fusion
- Twist Mux
- Logitech F710 Control
- D435i Depth Camera Driver
- Livox Mid360 LiDAR Driver
- 2D Scanner

Webserver

Coming Soon

Not Supported

Teleoperation

Command Line Interface

This requires the installation of the ROS2 Modules on H1. If not done please follow installation on H1 ROS2 Modules section. To teleoperate the H1, run the following command in one of the SSH sessions: .. code-block:: bash

ROS_DOMAIN_ID=10 ros2 run teleop_twist_keyboard teleop_twist_keyboard –ros-args –remap cmd_vel:=/h1_unit_238/cmd_vel

Command Line Interface

To teleoperate the H1, run the following command in one of the SSH sessions:

```
ROS_DOMAIN_ID=10 ros2 run teleop_twist_keyboard teleop_twist_keyboard
```

Steamdeck Integration

Important: The Lidar data and camera stream are dependent on the DDS configuration and the WiFi dongle being used.

Setup Instructions:

1. Before launching the H1 bringup, run the following command on the H1: ros2_wifi
2. Launch the system: ros2 launch h1_bringup system.launch.py
3. Click on the joystick controller icon on the Steamdeck to access controls.

Controls:

Leg Control:

- Slow Movement: L1 + Left Joystick
- Right Movement: R1 + Left Joystick

Arm Control:

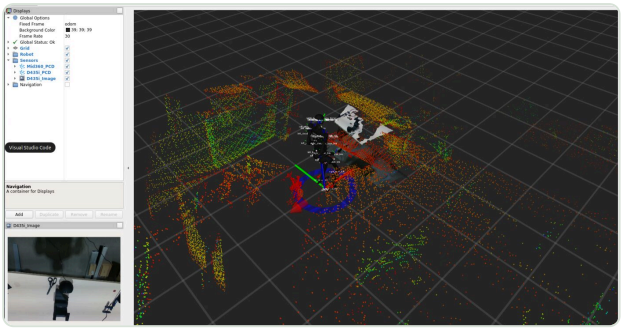
- To be announced (TBA). This can be modified using the h1_steamdeck libjoy .

Visualization

RViz2

You can view the H1's current state by entering the following command in one of the SSH sessions:

```
ros2 launch h1_viz view_robot.launch.py
```



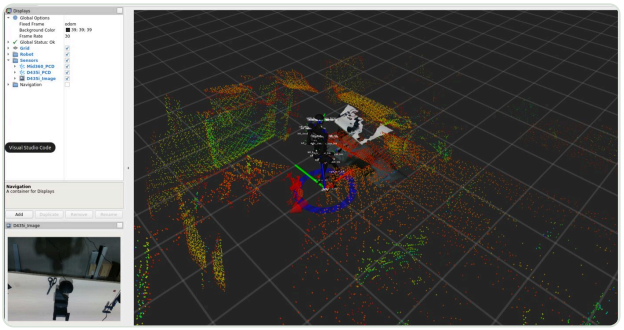
_images/h1_viz.png

H1 Robot Visualization in RViz

RViz2

You can view the H1's current state by entering the following command in one of the SSH sessions:

```
ros2 launch h1_viz view_robot.launch.py
```



_images/h1_viz.png

H1 Robot Visualization in RViz

Rigs

Not Supported

Not Supported

Manipulation

H1 Robot Control

Modes Selection

Important: Most functionalities are disable for Unitree H1. Startup requires remote.

Posture & State Commands

Command	Description

damp	Set all motors to damping mode.
start	Start locomotion control.
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.

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 ω [duration]"	Set velocity (m/s, m/s, rad/s, [duration in s]).
move="vx vy ω"	Command motion without duration.
set_task_id=<id>	Set active task ID.
set_speed_mode=<mode>	Change locomotion speed mode.

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.

Toggle Commands

Command	Parameters	Description

continous_gait=<true/false>	bool	Enable or disable continuous gait mode.
switch_move_mode=<true/false>	bool	Switch between movement mode and standing mode.

Examples

Damp State:

```
ROS_DOMAIN_ID=10 ros2 service call /h1_unit_238/hardware_modes h1_interface/srv/H1Modes \
'{"request_data": "damp"}'
```

Dual Arm Record H1-2

Warning: When launch the arm driver the arm violently snaps to the H1-2 initial position. Ensure the arms are near to the start position to prevent damage to the robot and keep surrounding area clear.

```
ros2 launch h1_platform h1_2_arm_record.launch.py
```

Configuration

Update to requirements the /opt/mybotshop/src/mybotshop/h1_platform/config/h1_2_dual_arm_controller.yaml file

Start recording

Important: Keep in mind the arms will start moving from this area so try to keep away from the body to avoid scratches. In the end of the recording be sure to bring the arms in a location close to the initial position for the next recording session.

```
ROS_DOMAIN_ID=10 ros2 service call /h1_unit_238/dual_arm/record std_srvs/srv/SetBool "data:
true"
```

Stop recording and save

```
ROS_DOMAIN_ID=10 ros2 service call /h1_unit_238/dual_arm/record std_srvs/srv/SetBool "data:
false"
```

Play last recorded trajectory

```
ROS_DOMAIN_ID=10 ros2 service call /h1_unit_238/dual_arm/playback h1_interface/srv/H1Modes
"request_data: ''"
```

Play specific file

```
ROS_DOMAIN_ID=10 ros2 service call /h1_unit_238/dual_arm/playback h1_interface/srv/H1Modes
```

```
"request_data: '/opt/mybotshop/h1_2_trajectories/trajectory_20250925_160052.csv'"
```

Dual Arm Control H1-2

```
ros2 launch h1_platform h1_2_arm_server.launch.py
```

Configuration

Update to requirements the /opt/mybotshop/src/mybotshop/h1_platform/config/h1_2_dual_arm_controller.yaml file

Poses

T pose

```
ROS_DOMAIN_ID=10 ros2 action send_goal /h1_unit_1789/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, 0.3, 0.0, 1.57, 0.0, 0.0, 0.0,   0.0,-0.3, 0.0, 1.57, 0.0,
0.0, 0.0],
            time_from_start: {sec: 2, nanosec: 0}},
            {positions: [0.0, 1.57, 0.0, 1.57, 0.0, 0.0, 0.0,  0.0,-1.57,0.0, 1.57, 0.0,
0.0, 0.0],
            time_from_start: {sec: 5, nanosec: 0}}
        ]}}"
```

- Ready pose

```
ROS_DOMAIN_ID=10 ros2 action send_goal /h1_unit_1789/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, 0.3, 0.0, 0.7, 0.0, 0.0, 0.0,
                     0.0,-0.3, 0.0, 0.7, 0.0, 0.0, 0.0],
        time_from_start: {sec: 2, nanosec: 0}},
    ]}}"
```

Leg Control Mode

The following H1 modes are available via ROS2 services:

```
- damp
- stand_switch
- increase_swing_height
- decrease_swing_height
- enable_ctrl
```

Note: Pressing the Tab key on the keyboard will autocomplete the service request.

Example of Activating a Mode

To activate a specific mode, use the following command:

```
ROS_DOMAIN_ID=10 ros2 service call /h1/leg_modes h1_srvs/srv/H1Modes '{"request_data":
"stand_switch"}'
```

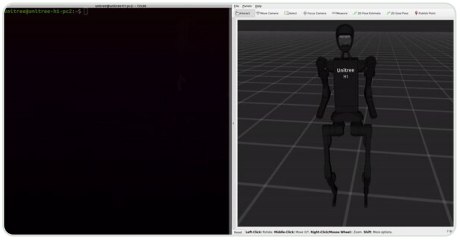
Arm Low-Level Control

The predefined arm positions do not include obstacle avoidance for either the robot or its environment.

Important: Ensure that the arms are straight down when powering on the robot. If the orientation is different, it will affect the ROS2 control operation. The arms will initialize and move so that the forearms are facing forward.

To set the arms to their default position, use the following command:

```
h1_arm_default
```



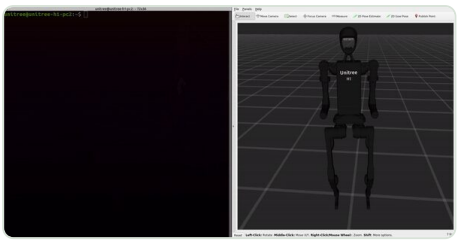
Executing arm default command

H1 arm moving to default position

Warning: Using the h1_arm_zero command brings the arms to a position that prevents backlash when restarting the ROS2 control driver. This should be done before closing the ROS2 driver if you plan to run the driver again later.

To zero the arms, execute:

```
h1_arm_zero
```

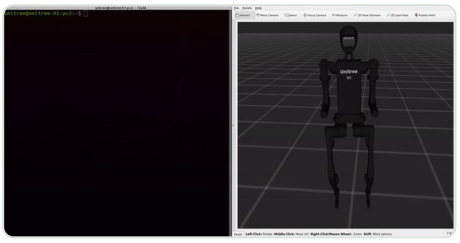


Executing arm zero command

H1 arm moving to zero position

To set the arms to a T-pose, use:

```
h1_arm_t
```

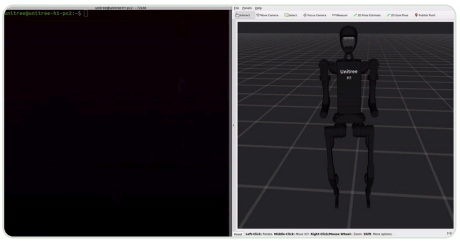


Executing T-pose command

H1 arm moving to T-pose

For a sample pose useful for picking objects, use:

```
h1_arm_pick
```



Executing pick pose command

H1 arm moving to pick pose

Examples of Trajectory Control

Bring Left Arm to Zero Position

To bring the left arm to the zero position, run:

```
ROS_DOMAIN_ID=10 ros2 action send_goal /left_arm_controller/follow_joint_trajectory
control_msgs/action/FollowJointTrajectory -f "{
  trajectory: {
    joint_names: [left_shoulder_pitch_joint,
    left_shoulder_roll_joint, left_shoulder_yaw_joint,
    left_elbow_joint],
    points: [
      { positions: [0.0, 0.0, 0.0, 0.0], time_from_start: { sec: 2 } },
      { positions: [0.5, 0.0, 0.0, 0.1], time_from_start: { sec: 4 } },
```

```
    ]
  }
}"
```

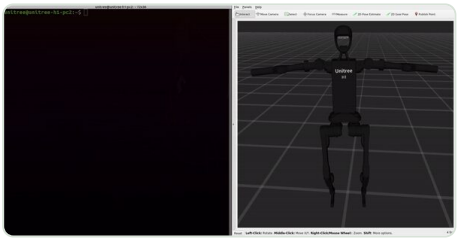
Bring Right Arm to Zero Position

To bring the right arm to the zero position, run:

```
ROS_DOMAIN_ID=10 ros2 action send_goal /right_arm_controller/follow_joint_trajectory
control_msgs/action/FollowJointTrajectory -f "{
  trajectory: {
    joint_names: [right_shoulder_pitch_joint,
    right_shoulder_roll_joint, right_shoulder_yaw_joint,
    right_elbow_joint],
    points: [
      { positions: [0.0, 0.0, 0.0, 0.0], time_from_start: { sec: 2 } },
      { positions: [0.5, 0.0, 0.0, 0.1], time_from_start: { sec: 4 } },
    ]
  }
}"
```

To perform a final power-off of the arms, use:

```
h1_arm_zero
```



Executing trajectory control to zero position

H1 arm transitioning from T-pose to zero position

MoveIt2 Integration

To launch the H1 MoveIt2 driver, run:

```
ros2 launch h1_moveit system.launch.py
```

This will open MoveIt2, which can be used for various manipulation tasks. Be sure to set a planner, such as TRRT, before planning and executing a motion plan.

Navigation

Simultaneous Localization and Modulation

Ensure the h1 services are running and then also launch

```
ros2 launch h1_navigation slam.launch.py
```

You can begin mapping using the teleop at 0.2m/s with the keyboard. 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/h1_navigation/maps/custom_map --ros-args --remap
map:=/h1_unit_001/map
ROS_DOMAIN_ID=10 ros2 run nav2_map_server map_saver_cli -f
/opt/mybotshop/src/mybotshop/h1_navigation/maps/custom_map_2 --ros-args --remap
map:=/h1_unit_001/map
```

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

```
colcon build --symlink-install
```

Then source the environment

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

Odometry Navigation

The odometry-based navigation is launched by executing:

```
ros2 launch h1_navigation odom_navi.launch.py
```

Map Navigation

The map-based navigation is launched by executing:

```
ros2 launch h1_navigation map_navi.launch.py
```

GPS Navigation

Not supported

Simultaneous Localization and Modulation

The SLAM navigation is launched by executing:

```
ros2 launch h1_navigation slam.launch.py
```

To save the map, run:

```
ros2 run nav2_map_server map_saver_cli -f ~/map
```

Odometry Navigation

The odometry-based navigation is launched by executing:

```
ros2 launch h1_navigation odom_navi.launch.py
```

Map Navigation

The map-based navigation is launched by executing:

```
ros2 launch h1_navigation map_navi.launch.py
```

GPS Navigation

Not supported

Sensors

Depth Cameras

Intel Realsense D435i

Test the native driver via:

```
ros2 launch realsense2_camera rs_launch.py depth_module.depth_profile:=1280x720x30
pointcloud.enable:=true
```

This should be running via services and can be controlled via the webserver or systemctl to activate or deactivate it. When turned off you can test via:

```
ros2 launch h1_depth_camera realsense.launch.py
```

Depth Camera ZED2i

This should be running via services in the nividia bagpack and can be controlled via the systemctl to activate or deactivate it. When turned off you can test via:

```
ros2 launch h1_depth_camera zed.launch.py
```

Lidars

Ouster

This should be running via services and can be controlled via the webserver or systemctl to activate or deactivate it. When turned off you can test via:

```
ros2 launch h1_lidar ouster.launch.py
```

GPS

Emlid M2

This should be running via services and can be controlled via the webserver or systemctl to activate or deactivate it. When turned off you can test via:

```
ros2 launch h1_gps emlid_m2.launch.py
```

For configuration, the webserver's VNC can be utilized. Open the browser and navigate to http://192.168.2.15 to configure the emlid M2's output, RTK settings, Base station connection, etc.

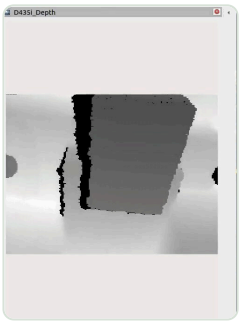
Auxiliary

Depth Cameras

Intel Realsense D435i

1. The D435i is launched by default with the H1 bringup. To launch the Realsense D435i separately, execute: ros2 launch h1_depth_camera realsense_d435i.launch.py
2. The launch file is configured to enable continuous depth stream information from the Realsense D435i without lag. To further change parameters, modify the configuration in: h1_depth_camera/launch/realsense_d435i.launch.py

Intel Realsense D435i Normal Output



Intel Realsense D435i Depth Output

Lidars

The Mid360 is launched by default with the H1 bringup. To activate the Mid360 Lidar, execute:

```
ros2 launch h1_livox system.launch.py
```

Mid360 Lidar Output Visualization in RViz

Mid360 Lidar Output Visualization in RViz

GPS

Auxiliary

Simulation

Gazebo

```
ros2 launch h1_gazebo h1_2_fortress_simulation.launch.py
```

Unitree RL Gym

For sim to real deployment please follow instructions at unitree_rl_gym

Isaac Lab

Please refer to Isaac Lab for training and deployment.

Gazebo

Available in Humble

Isaac Sim

Available in Isaac Lab

Packages

Robot Description

The H1 robot description package is named h1_description . It contains the URDF and mesh files necessary to visualize and simulate the H1 robot in various ROS2 tools.

Twist Mux

The twist mux package for the H1 is named h1_control . It manages multiple velocity command inputs and prioritizes them to ensure smooth and safe robot operation.

EKF Localization

The Extended Kalman Filter (EKF) localization package for the H1 is named h1_control . It fuses sensor data to provide accurate and reliable robot pose estimates.

Robot Description

The H1 robot description package is named h1_description . It contains the URDF and mesh files necessary to visualize and simulate the H1 robot in various ROS2 tools.

Twist Mux

The twist mux package for the H1 is named h1_control . It manages multiple velocity command inputs and prioritizes them to ensure smooth and safe robot operation.

EKF Localization

The Extended Kalman Filter (EKF) localization package for the H1 is named h1_control . It fuses sensor data to provide accurate and reliable robot pose estimates.

Debugging

RQT

```
ROS_DOMAIN_ID=10 ros2 run rqt_gui rqt_gui --ros-args --remap tf:=/h1_unit_238/tf --ros-args
--remap tf_static:=/h1_unit_238/tf_static
```

Dynamic Reqconfigure

- From configure select Dynamic Reconfigure

```
ROS_DOMAIN_ID=10 ros2 run rqt_gui rqt_gui --ros-args --remap tf:=/h1_unit_238/tf --ros-args
--remap tf_static:=/h1_unit_238/tf_static
```

TF

```
ROS_DOMAIN_ID=10 ros2 run rqt_tf_tree rqt_tf_tree --force-discover --ros-args --remap
tf:=/h1_unit_238/tf --ros-args --remap tf_static:=/h1_unit_238/tf_static
```

```
ROS_DOMAIN_ID=10 ros2 run tf2_tools view_frames.py --force-discover --ros-args --remap
tf:=/h1_unit_238/tf --ros-args --remap tf_static:=/h1_unit_238/tf_static
```

Debugging Tips

When running RQt GUI with multiple robots in the same network, it's important to set a unique ROS_DOMAIN_ID for each robot to avoid communication conflicts. You can do this by exporting the ROS_DOMAIN_ID environment variable before launching RQt GUI. For example, to set the domain ID to 10, use the following command:

```
ROS_DOMAIN_ID=10 ros2 run rqt_gui rqt_gui
```

This command ensures that RQt GUI communicates with the robot assigned to domain ID 10, preventing interference from other robots on the network.

Dynamic Reconfigure

To adjust parameters on-the-fly using dynamic reconfigure, you can use the following command:

```
ROS_DOMAIN_ID=10 ros2 run rqt_reconfigure rqt_reconfigure
```

TF Tree

To visualize the TF tree of the H1 robot, you can use the following command:

```
ROS_DOMAIN_ID=10 ros2 run rqt_tf_tree rqt_tf_tree --force-discover
```

Video stream

To view the video stream from the H1's camera, you can use the following command:

```
gst-launch-1.0 udpsrc address=230.1.1.1 port=1720 multicast-iface=eth0 ! \
 application/x-rtp, media=video, encoding-name=H264 ! \
 rtph264depay ! queue max-size-buffers=1 ! h264parse ! queue ! avdec_h264 ! \
 videoconvert ! autovideosink
```

Miscellanious

SDK Examples

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

```
cd /opt/mybotshop/04Aug2025_unitree_sdk2/build/bin
./h1_loco_client --network_interface=eth0 --shake_hand
```

Computer - Robot Synchronization

```
rsync -avP -t --delete -e ssh src unitree@192.168.123.164://opt/mybotshop
```

Update System Kernel Cyclone

```
printf
"net.core.rmem_max=33554432\nnet.core.rmem_default=33554432\nnet.core.wmem_max=33554432\nne
t.core.wmem_default=33554432\nnet.ipv4.udp_rmem_min=16384\nnet.ipv4.udp_wmem_min=16384\n" |
sudo tee /etc/sysctl.d/90-ros2-dds.conf
```

```
sudo sysctl --system
```

Enable Internet - LAN

This enables LAN via the ethernet cable. This has to be connected to a router with internet access

```
sudo ip link set eth0 down && sudo ip link set eth0 up
sudo dhclient eth0
```

Computer - Robot Synchronization

```
rsync -avP -t --delete -e ssh src unitree@192.168.123.164://opt/mybotshop
```

Low-Level Control

The low-level control for the H1 can be directly accessed via Unitree's provided examples in their documentation . Running the provided qre_h1 driver with the low-level commands from Unitree examples should work concurrently.

Installation

Installation Robot

- Enable Internet by connecting it to a switch with internet

```
sudo ip link set eth0 down && sudo ip link set eth0 up
sudo dhclient eth0
```

- Switch H1 id with ctrl+h

```
h1_unit_238
```

- Switch username

```
sudo hostnamectl set-hostname h1-unit-284-pc-4
```

- Disable redundant B2 slam

```
sudo systemctl stop unitree_slam.service
sudo systemctl disable unitree_slam.service
sudo rm /lib/systemd/system/unitree_slam.service
```

- 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')"
```

- Install ROS2 Humble dev tools

```
sudo apt update && sudo apt install curl -y
export ROS_APT_SOURCE_VERSION=$(curl -s https://api.github.com/repos/ros-
infrastructure/ros-apt-source/releases/latest | grep -F "tag_name" | awk -F\" '{print $4}')
curl -L -o /tmp/ros2-apt-source.deb "https://github.com/ros-infrastructure/ros-apt-
source/releases/download/${ROS_APT_SOURCE_VERSION}/ros2-apt-
source_${ROS_APT_SOURCE_VERSION}.$(. /etc/os-release && echo $VERSION_CODENAME)_all.deb" #
```

```
If using Ubuntu derivates use $UBUNTU_CODENAME
sudo dpkg -i /tmp/ros2-apt-source.deb
sudo apt install ros-dev-tools
```

- Create directory for h1 Workspace

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

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

- Clone the repository, copy over to the h1's PC and run the installer script - Press Y when installing Livox Mid360 - Select gdm3 when prompted in webserver installation

```
cd /opt/mybotshop/src/mybotshop/ && sudo chmod +x h1_install.sh && ./h1_install.sh
```

```
cd /opt/mybotshop/src/mybotshop/h1_webserver && sudo chmod +x webserver_installer.sh &&
./webserver_installer.sh
```

- 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
```

- Build Kiss ICP

```
python3 -m pip install --user --upgrade "cmake>=3.24"
export PATH="$HOME/.local/bin:$PATH"
cd /opt/mybotshop && colcon build --symlink-install --packages-select kiss_icp --cmake-args
-DCMAKE_POLICY_VERSION_MINIMUM=3.24
```

- Build ros2 workspace

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

- Add the following to .bashrc

```
# MYBOTSHOP
source /opt/mybotshop/install/setup.bash
alias h1_build='cd /opt/mybotshop && colcon build --symlink-install --cmake-args -
DCMAKE_BUILD_TYPE=Release && source install/setup.bash'
```

- Increase cyclone kernel

```
printf
"net.core.rmem_max=33554432\nnet.core.rmem_default=33554432\nnet.core.wmem_max=33554432\nne
```

```
t.core.wmem_default=33554432\nnet.ipv4.udp_rmem_min=16384\nnet.ipv4.udp_wmem_min=16384\n" |
sudo tee /etc/sysctl.d/90-ros2-dds.conf
```

```
sudo sysctl --system
```

- Run the startup installer - Ensure H1/H1-2 for description in the /opt/mybotshopt/src/mybotshop/h1_bringup/scripts/startup_installer.py

```
ros2 run h1_bringup startup_installer.py
```

Installation External

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

```
colcon build --symlink-install --packages-select h1_description h1_viz && source
install/setup.bash
```

- Source and run rviz2

```
export H1_NS="h1_unit_238"
ros2 launch h1_viz view_robot.launch.py
```

Installation Robot

Note: This repository should already be available and built on the H1's Nvidia board if it has been configured by the MYBOTSHOP team.

1. Clone this repository into the H1's Nvidia board.
2. Run the bash install script.
3. Build the repository by executing the following command: colcon build --symlink-install && source install/setup.bash
4. After building, remember to source the install/setup.bash . Ensure you add the source command to your workspace in the .bashrc file.

Installation External

Note: Important: This process is necessary to enable Unitree SDK's DDS topics to appear. Without these steps, only the MYBOTSHOP package ROS2 topics will be visible. You can skip this procedure if you do not need the Unitree topics.

Part 1

1. Navigate to the unitree_sdk2 folder inside the thirdparty directory and execute the installation script: cd thirdparty/unitree_sdk2 sudo bash install.sh
2. Build the examples: rm -rf build && mkdir build && cd build && cmake .. && make

3. You can now directly run high-level and low-level examples as explained in the Unitree H1 SDK documentation.

Part 2

Requirements: ROS2 Foxy on Ubuntu 20.04 .

1. Install the necessary dependencies: sudo apt install ros- $ROS_DISTRO -rmw-cyclonedds-cpp ros- $ROS_DISTRO -rosidl-generator-dds-idl -y
2. Unsource ROS Foxy in the terminal (i.e., by commenting out the sourcing command in the .bashrc file).
3. Build Cyclone DDS: colcon build --packages-select cyclonedds
4. Install other dependencies by running the provided install script: sudo chmod +x h1_install.bash && ./h1_install.bash
5. Source the environment: source install/setup.bash source /opt/ros/foxy/setup.bash
6. Build the repository: colcon build --symlink-install

Network Setup

1. Connect to the 123 network using a static IP (No DHCP, as it disrupts the network).
2. Note down the network interface name (e.g., eth0).
3. Edit and source the h1_bringup/config/setup.bash file with the corrected network interface. For example: #!/bin/bash echo "Setup Unitree ROS2 environment" source /opt/ros/foxy/setup.bash source $HOME /ros2_ws/install/setup.bash export RMW_IMPLEMENTATION = rmw_cyclonedds_cpp export CYCLONEDDS_URI = '<CycloneDDS><Domain><General> <Interfaces> <NetworkInterface name="eth0" priority="default" multicast="default" /> </Interfaces></General></Domain></CycloneDDS>'
4. Add the above to your .bashrc file for persistence.

Verification

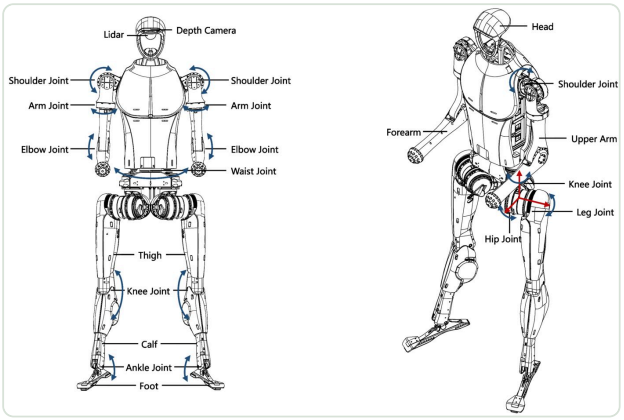
1. Check the available ROS2 topics: ros2 topic list
2. Echo a specific topic (e.g., /lowstate): ros2 topic echo /lowstate

8 H1 Hardware Overview & Joint Reference

H1 Overview

Component Overview

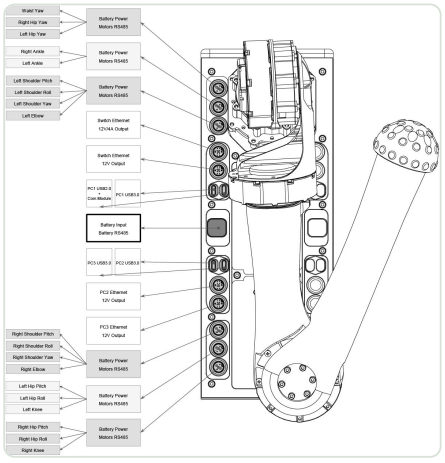
The H1 robot comprises upper and lower bodies, each featuring multiple degrees of freedom. The single arm includes joints for body-shoulder, shoulder, upper arm, and elbow movements, while the single leg features joints for hip, leg, knee, and ankle motions. Additionally, the waist section offers a waist joint. In total, the robot boasts 19 degrees of freedom, enabling precise motion and posture control through 19 joint motors.



H1 Robot

Electrical Interfaces

Located on the right side of the H1 robot, the electrical interfaces facilitate connections to various components such as body joint motors, sensor peripherals, and Ethernet ports. This design simplifies debugging, troubleshooting, and secondary development processes.



H1 Robot

On-board Computers

The H1 robot comes equipped with a “Motion Control Computing Unit” and optionally a “Development Computing Unit” (additional expansion unit available).

Component	Motion Control Computing Unit (PC 1)	Development Computing Unit (PC 2, PC 3)
Model	i5-1235U	i7-1255U / i7-1265U
Number of Cores	10	10
Number of Threads	12	12
Max Turbo Frequency	4.40 GHz	4.70 GHz & 4.80 GHz
Memory	8GB	16GB & 32GB
Memory Type	LPDDR5 5200 MT/s (dual-channel)	LPDDR5 5200 MT/s (dual-channel)
Cache	12 MB Intel® Smart Cache	12 MB Intel® Smart Cache
Storage	500GB or more	500GB or more
Intel® Graphics Processing Unit	None	6.0
GPU	Intel® Iris® Xe Graphics eligible	Intel® Iris® Xe Graphics eligible
Max Dynamic Frequency of GPU	1.20 GHz	1.20 GHz
Gauss and Neural Accelerator	3.0	3.0
Intel® Deep Learning Boost	Yes	Yes
Intel® Adaptix™ Technology	Yes	Yes
Intel® Hyper-Threading Technology	Yes	Yes
Instruction Set	64-bit	64-bit
OpenGL	4.6	4.6
OpenCL	3.0	3.0
DirectX	12.1	12.1
IP Address	Unaccessible	192.168.123.162, 192.168.123.163

Attention: The “Motion Control Computing Unit” is exclusively reserved for Unitree motion control programs and is not accessible for external use. Developers are limited to utilizing the “Development Computing Unit” for secondary development. For the initial username and password, please contact our technical support team. In the table above, the PC3 “Development Computing Unit” is an optional configuration, with its IP address set to 192.168.123.163. The CPU module may be upgraded to a more advanced version, provided that the performance meets or exceeds the specifications listed.

Lidar

The H1 robot's head incorporates the MID-360 Lidar (IP: 192.168.123.120), which offers superior environmental perception capabilities. Employing omnidirectional, full-angle scanning technology, it delivers real-time and precise environmental data, facilitating accurate recognition and measurement of surrounding objects.

Depth Camera

Equipped with the D435i depth camera, the H1 robot's head boasts exceptional visual perception capabilities. This enables more precise perception and understanding of the surrounding environment, allowing for accurate spatial perception and obstacle detection. Consequently, the robot can interact intelligently with its environment and adapt to various scenarios effectively.

M107 Joint Motor

The H1 robot utilizes the self-developed M107 joint motor by Unitree, known for its outstanding performance. Featuring a maximum torque of 360N.m and a maximum tension of 10000N (under equivalent conditions of a 3.5cm force arm), this motor's hollow axle design makes it lightweight and compact. Additionally, equipped with dual encoders, it provides more accurate position and velocity feedback, meeting the requirements of high-precision control.

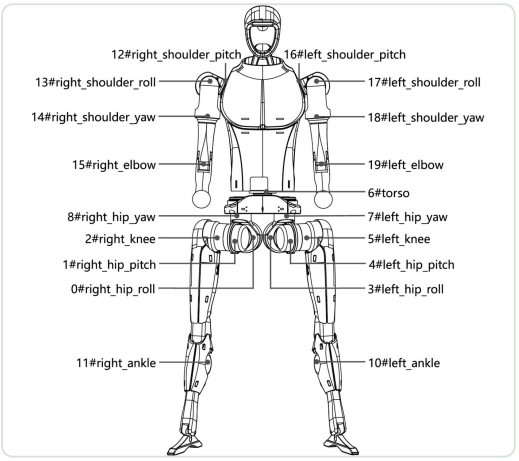
Joint Numbering and Joint Limits

The table below provides an overview of the joint numbering system and their respective movement limits. Each joint is identified by a unique number and name, detailing its specific range of motion in radians. These constraints define the permissible angular movement for each joint, ensuring safe and effective operation. The data serves as a reference for configuring, calibrating, or troubleshooting the system's kinematics.

Joint Number	Joint Name	Limits (radians)
8	right_hip_yaw_joint	-0.43~+0.43
0	right_hip_roll_joint	-0.43~+0.43
1	right_hip_pitch_joint	-3.14~+2.53
2	right_knee_joint	-0.26~+2.05
11	right_ankle_joint	-0.87~+0.52
7	left_hip_yaw_joint	-0.43~+0.43
3	left_hip_roll_joint	-0.43~+0.43
4	left_hip_pitch_joint	-3.14~+2.53
5	left_knee_joint	-0.26~+2.05
10	left_ankle_joint	-0.87~+0.52

6	torso_joint	-2.35~+2.35
12	right_shoulder_pitch_joint	-2.87~+2.87
13	right_shoulder_roll_joint	-3.11~+0.34
14	right_shoulder_yaw_joint	-4.45~+1.3
15	right_elbow_joint	-1.25~+2.61
16	left_shoulder_pitch_joint	-2.87~+2.87
17	left_shoulder_roll_joint	-0.34~+3.11
18	left_shoulder_yaw_joint	-1.3~+4.45
19	left_elbow_joint	-1.25~+2.61

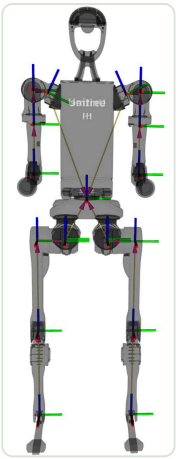
The image below displays the Unitree H1 humanoid robot with all joints labeled. It serves as a visual reference, correlating the joint numbers and names listed in the table to their positions on the robot.



H1 Robot

Reference Frame, Joint Axes, and Zero Position

In the zero-degree configuration of all joints, the coordinate system is depicted as shown in the figure below. The x-axis is represented by the red line, the y-axis by the green line, and the z-axis by the blue line.



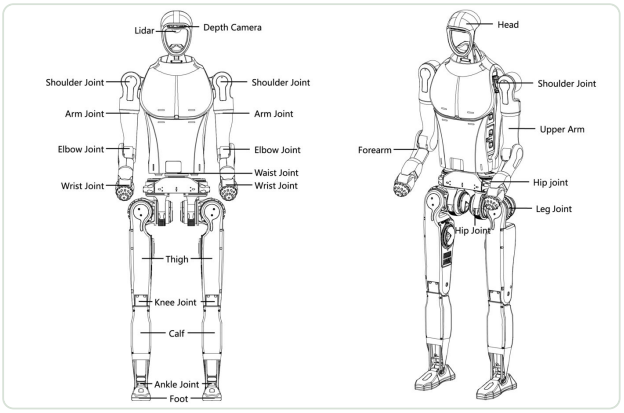
H1 Robot

9 H1-2 Hardware Overview & Joint Reference

H1-2 Overview

Component Overview

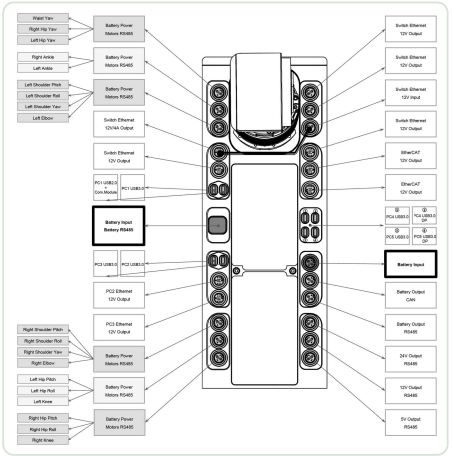
The H1-2 model is an advanced iteration of the H1, featuring improved motion capabilities and an expanded range of movement. Each arm is equipped with 7 degrees of freedom, each leg with 6, and the waist with 1 degree of freedom, resulting in a total of 27 degrees of freedom throughout the robot. With 27 joint motors, this configuration allows for highly precise control of motion and posture adjustment.



H1-2 Component Overview

Electrical Interfaces

The H1-2 robot incorporates electrical interfaces positioned on its right side. These interfaces facilitate the connection of body joint motors, sensor peripherals, Ethernet ports, and other components. This design enhances convenience during debugging, issue troubleshooting, and secondary development processes.



H1-2 Electrical Interface

On-board Computers

The H1-2 robot is equipped with one “Motion Control Computing Unit” and one “Development Computing Unit,” with an option to configure an additional expansion unit.

Component	Motion Control Computing Unit (PC 1)	Development Computing Unit (PC 2, PC 3)
Model	i5-1235U	i7-1255U / i7-1265U
Number of Cores	10	10
Number of Threads	12	12
Max Turbo Frequency	4.40 GHz	4.70 GHz & 4.80 GHz
Memory	8GB	16GB & 32GB
Memory Type	LPDDR5 5200 MT/s (dual-channel)	LPDDR5 5200 MT/s (dual-channel)
Cache	12 MB Intel® Smart Cache	12 MB Intel® Smart Cache
Storage	500GB or more	500GB or more
Intel® Graphics Processing Unit	None	6.0
GPU	Intel® Iris® Xe Graphics eligible	Intel® Iris® Xe Graphics eligible
Max Dynamic Frequency of GPU	1.20 GHz	1.20 GHz
Gauss and Neural Accelerator	3.0	3.0
Intel® Deep Learning Boost	Yes	Yes
Intel® Adaptix™ Technology	Yes	Yes
Intel® Hyper-Threading Technology	Yes	Yes
Instruction Set	64-bit	64-bit
OpenGL	4.6	4.6
OpenCL	3.0	3.0
DirectX	12.1	12.1
IP Address	Unaccessible	192.168.123.162, 192.168.123.163

Attention: The “Motion Control Computing Unit” is exclusively reserved for Unitree motion control programs and is not accessible for external use. Developers are limited to utilizing the “Development Computing Unit” for secondary development. For the initial username and password, please contact our technical support team. In the table above, the PC3 “Development Computing Unit” is an optional configuration, with its IP address set to 192.168.123.163. The CPU module may be upgraded to a more advanced version, provided that the performance meets or exceeds the specifications listed.

Lidar

The H1-2 robot's head is equipped with the MID-360 Lidar (IP: 192.168.123.120), providing exceptional environmental perception capabilities. Utilizing omnidirectional, full-angle scanning technology, the lidar captures real-time and precise environmental data. It enables the robot to efficiently detect and measure surrounding objects while delivering high-resolution point cloud data for enhanced operational accuracy.

Depth Camera

The H1-2 robot's head also features the D435 depth camera, which offers superior visual perception capabilities. This camera enhances the robot's ability to perceive and understand its environment accurately, facilitating precise spatial awareness and obstacle detection. These features allow the robot to intelligently interact with its surroundings and adapt flexibly to diverse scenarios.

M107 Joint Motor

The H1-2 robot integrates the self-developed M107 joint motor by Unitree, renowned for its exceptional performance and features. The motor delivers a maximum torque of 360 N·m and a maximum tension of 10,000 N under equivalent conditions with a 3.5 cm force arm. Designed with a hollow axle, the motor achieves a lightweight and compact structure. Additionally, it includes dual encoders for precise position and velocity feedback, meeting the demands of high-precision control applications.

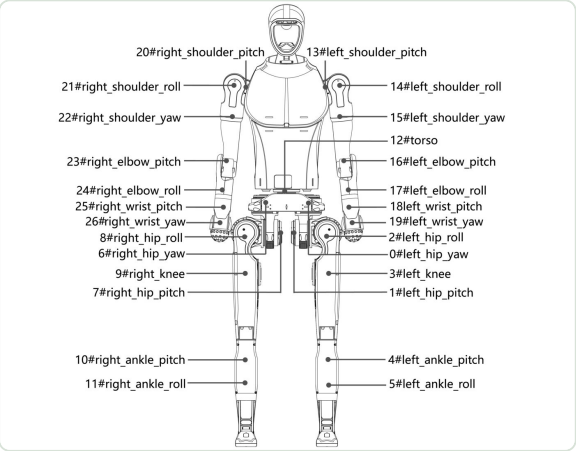
Joint Numbering and Joint Limits

The joint numbering and corresponding limits for the H1-2 robot are as follows:

Joint Number	Joint Name	Limits (radians)
20	right_shoulder_pitch	-1.57 ~ +3.14
21	right_shoulder_roll	-3.4 ~ +0.38
22	right_shoulder_yaw	-2.66 ~ +3.01
23	right_elbow_pitch	-1.6 ~ +2.53
24	right_elbow_roll	-2.967 ~ +2.967
25	right_wrist_pitch	-0.471 ~ +0.349
26	right_wrist_yaw	-1.012 ~ +1.012
13	left_shoulder_pitch	-3.14 ~ +1.57
14	left_shoulder_roll	-0.38 ~ +3.4
15	left_shoulder_yaw	-3.01 ~ +2.66

16	left_elbow_pitch	-2.53 ~ +1.6
17	left_elbow_roll	-2.967 ~ +2.967
18	left_wrist_pitch	-0.471 ~ +0.349
19	left_wrist_yaw	-1.012 ~ +1.012
12	torso	-3.14 ~ +1.57
6	right_hip_yaw	-0.43 ~ +0.43
7	right_hip_pitch	-3.14 ~ +2.5
8	right_hip_roll	-3.14 ~ +0.43
9	right_knee	-0.26 ~ +2.05
10	right_ankle_pitch	-0.897334 ~ +0.523598
11	right_ankle_roll	-0.261799 ~ +0.261799
0	left_hip_yaw	-0.43 ~ +0.43
1	left_hip_pitch	-3.14 ~ +2.5
2	left_hip_roll	-0.43 ~ +3.14
3	left_knee	-0.26 ~ +2.05
4	left_ankle_pitch	-0.897334 ~ +0.523598
5	left_ankle_roll	-0.261799 ~ +0.261799

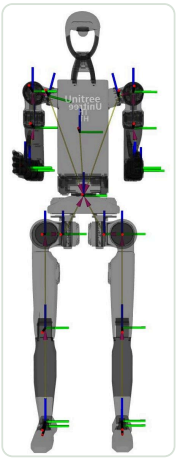
The image below displays the Unitree H1-2 humanoid robot with all joints labeled. It serves as a visual reference, correlating the joint numbers and names listed in the table to their positions on the robot.



H1-2 Joints

Reference Frame, Joint Axes, and Zero Position

In the zero-degree configuration of all joints, the coordinate system is depicted as shown in the figure below. The x-axis is represented by the red line, the y-axis by the green line, and the z-axis by the blue line.



H1-2 Tf

10 Getting Help & RMA

RMA / Support

How to Raise a Return Merchandise Authorization (RMA) Case

At QUADRUPED Robotics, we are committed to providing excellent support to ensure your issues are resolved 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 a RMA number along with a return address or a shipping label.

Important Shipping Information

Note: Please note: All goods must be sent back to the following address: QUADRUPED Robotics GmbH c / o MYBOTSHOP GmbH Willy - Messerschmitt - Strasse 12 50126 Bergheim Germany Do not ship goods without obtaining a RMA number. Parcels sent without a RMA number will be declined and returned to the sender. Do not ship goods to QUADRUPED Robotics GmbH Office in Leverkusen!

Help Resources

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

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

11 Getting Involved

Getting Involved

Contact support @ mybotshop . de for more information.

12 FAQ

Frequently Asked Questions

General Questions and Answers

- Question: Is it possible to power on the Unitree H1 using an external power supply? Answer: Not at the moment.
- Question: How can I get access to the ROS2-specific repository for the Unitree H1? Answer: The ROS2-specific repository for the Unitree H1 is available to customers and partner institutions of QUADRUPED ROBOTICS GmbH. If you have purchased the robot from us, please send your purchase ID along with your GitHub username to support @ mybotshop . de . This will allow us to grant you access to the repository.
- Question: How do I connect to the Unitree H1 and perform teleop and RViz tasks via Wi-Fi instead of Ethernet?

Answer: To connect via Wi-Fi, add a USB Wi-Fi dongle to the H1 and configure it according to the GitHub documentation. Once configured, you can connect via SSH and perform teleop and RViz tasks wirelessly. The recommended dongle is the TP-Link TL-WN722N.

- Question: Is the Unitree H1 dustproof and waterproof? Answer: No, the Unitree H1 does not have dustproof or waterproof features.
- Question: What should be done if the large remote controller for the Unitree H1 doesn't turn on? Answer: If the large remote controller doesn't turn on, locate the reset button at the back of the controller. Insert a pin into the hole and hold it for 3 seconds to reset the controller.
- Question: What sensors can be added to the top of the Unitree H1? Answer: Various sensors and accessories can be added to the top of the Unitree H1, including speakers, warning lights, thermal cameras, and high-power search lights, enabling multiple applications.
- Question: Is it possible to connect external accessories to the Unitree H1 using a USB-C port? Answer: Yes, the H1 model includes a USB-C port through a hub that supports all necessary connections for external accessories.
- Question: Is there a freely available Gazebo model for the Unitree H1 that can be used with ROS2? Answer: Yes, the Unitree H1 operates with ROS2 Foxy, and a Gazebo model for simulation is available. You can find the simulation resources on the Unitree GitHub repository.
- Question: How can the H1 remote control be calibrated? Answer: To calibrate the H1 remote control, follow these steps: Hold the remote without touching the joysticks. Press and release the F1 and F3 buttons simultaneously. The remote will emit a continuous "beep beep" sound, indicating it is in calibration mode. Move the left and right joysticks to their maximum positions and rotate them several times until the "beep beep" sound stops. Press the F3 button once to apply the calibration and complete the process.
- Question: The H1 remote is not making a beep sound. What should I do? Answer: Press the F1 button three times to mute/unmute the remote's beep sound.