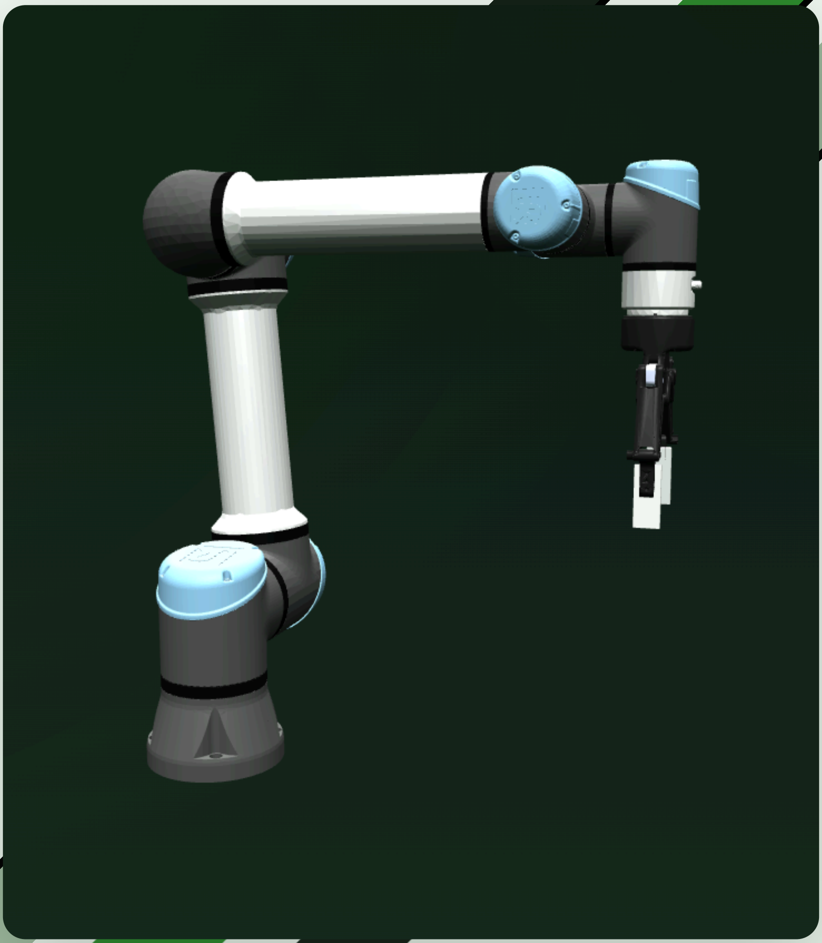
MYBOTSHOP

MOBILE MANIPULATION

Husky Mobile Manipulator

User Manual



CONTACT

info@mybotshop.de

support@mybotshop.de

mybotshop.de

Contents

1 Attention

Attention: Read this document carefully. In case of doubt, it is essential to consult specialists, experts, or the manufacturer (Clearpath Robotics for the mobile base; Universal Robots / UFactory for the manipulator) or distributor (MYBOTSHOP GmbH) of the assemblies used. The robot must not be put into operation before clarification. Contact information is available in Chapter 17, Getting Help & RMA.

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.

Do not operate the robot if you are intoxicated, under the influence of drugs, or unable to concentrate. Familiarize yourself with the emergency stop procedure (release the twist-switch and verify the key is in the unlock position) before switching the robot on.

Caution: The manipulator arm (UR5e or UFactory xArm6, depending on unit) can cause severe injury or equipment damage if operated by untrained personnel. Always clear the workspace before commanding arm motion, use slow speeds while testing new trajectories, and keep the emergency stop within reach at all times.

2 Disclaimer

The Husky Mobile Manipulator documented here is built and supported by *MYBOTSHOP GmbH* on the Clearpath Robotics Husky A200 platform. Basic ROS 2 (and, on several legacy customer units, ROS 1 Noetic) 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 Humble documentation first.

The client acknowledges and agrees that any information or materials provided by MYBOTSHOP GmbH 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 the general MMP Husky software/hardware stack as well as six distinct customer/university deployment units (ATC, TUM, BE, FH, FRA, TUHH), each with its own sensor and manipulator loadout — always confirm which configuration and firmware/ROS distribution is installed on your specific unit before following a startup or shutdown procedure. For terms, policies and compliance with local laws and regulations, refer to the Clearpath Robotics, Universal Robots and UFactory websites and the RMA/support process in Chapter 17.

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 in this document. This limitation on liability shall apply regardless of the form of action, whether in contract, tort, or otherwise.

3 About the Husky Mobile Manipulator

Mobile Manipulation Platform Husky

Note: This package supplies Sphinx-based tutorial content to assist you with setting up and using MBS Husky-xARM6-Robotiq robot. The tutorials topics are listed in the left column, and are presented in the suggested reading order.

Husky ATC	Husky TUM	Husky FH
Husky FRA	Husky BE	

Attention: These tutorials assume that you are comfortable working with ROS. We recommend starting with ros2 tutorials . if you are not familiar with ROS already.

Clearpath Husky

Welcome to the Clearpath Husky Mobile Manipulation Platform hub, your gateway to exploring a range of innovative projects and configurations built on the robust and versatile Clearpath Husky platform. The Clearpath Husky serves as a reliable foundation for diverse research and development endeavors, offering rugged all-terrain mobility, precise control, and customizable capabilities. The Husky is shipped in a medium wooden crate, ensuring safe transportation. At MyBotShop GmbH , we go the extra mile by providing the Husky with pre-installed and configured base software packages, including motor drivers and IMU sensors, in the Husky's ros2_ws .

From autonomous navigation and environmental sensing to manipulation and interaction, the Husky platform empowers researchers and developers to tackle a wide array of challenges in robotics. Explore the projects below to discover how the Husky is being utilized in various domains, from academic research and industrial automation to field applications and beyond.

Robot Upgrade

Manipulators

In terms of manipulators that can be installed and integrated with your mobile platform, several recommended arms are available. Each has its own perks and limitations. The table below illustrates some of the key features of the recommended arms that can be installed with the mobile platform.

Comparative Analysis

	xARM6	PANDA	UR10E

Model			
Company	UFACTORY	FRANKA EMIKA	UNIVERSAL ROBOTS
Operation	Host PC	Host PC	Host PC
			Teach Pendant
Software Development	Plug & play	Config. Req.	Config. Req.
I/O Ports	Multi	None	Multi
Sensor Integration	Available	None	Available
ROS Control	Immediate	Config. Req.	Config. Req.
Pricing	Low	High	High

Mobile Platforms

For mobile platform, several recommended mobile platforms are available. A new and interesting mobile platform that can be combined with your robot arm is the ROVO2 . The table below illustrates some of the key features of the recommended mobile platforms that can be utilized.

Comparative Analysis

	ROVO2	RIDGEBACK	HUSKY	WARTHOG
Robot Weight	290 kg	135 kg	50 kg	280 kg
Payload	500 kg	100 kg	75 kg	272 kg
Operation time	8 hrs	8 hrs	8 hrs	2.5 hrs
Top Speed	20 kmhr	3.96 kmhr	3.6 kmhr	18 kmhr
Terrain Adaptability	Extreme	None	Good	Very good
Pricing	Medium	Medium	Low	High

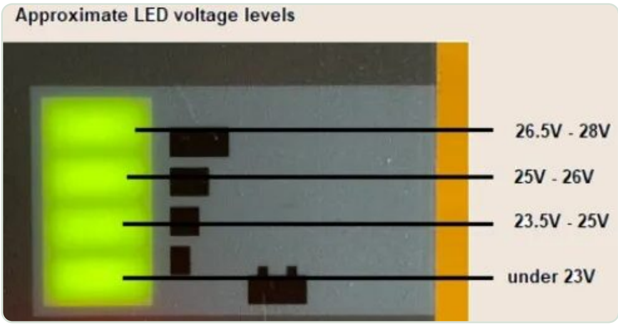
4 Quick Start – Powering On, Charging & Manipulator Activation

Powering on the Husky

Power Button	Emergency Switch Release	Charging Port

- Ensure the emergency button is released as well as the key is turned to the unlock position. ◦ To check if the emergency switch is released, twist the switch in the indicated direction on it. If it pops up, then the switch has successfully been released. ◦ If the robot light is yellow , it means that either the emergency switch is not released or the key is not turned.
- Press the power button on the front of the Husky. It has the power symbol as displayed.

Charging the Husky



_images/charging_indicator.webp

LED Voltage Indicators

The power port can be used to charge the Husky. The indicator shows the voltage levels of the Husky and does not necessarily indicate the remaining power available.

Manipulator Activation

To power-on the UR5e and Robotiq 2F-140:

```
ros2 service call /a200_1025/ur5e_activation/hsky_nano/srv/SerialBoolean "{data: true}"
```

Launch ROS2 driver for UR5e as well as Robotiq 2F-140:

```
ros2 launch hsky_manipulation ur5e_robotiq.launch.py
```

To test the UR5e and gripper you can use Moveit2 :

```
ros2 launch hsky_ur_moveit ur_moveit.launch.py
```

To power-off the UR5e and Robotiq 2F-140 completely, run:

```
ros2 run hsky_manipulation remote_shutdown.py
```

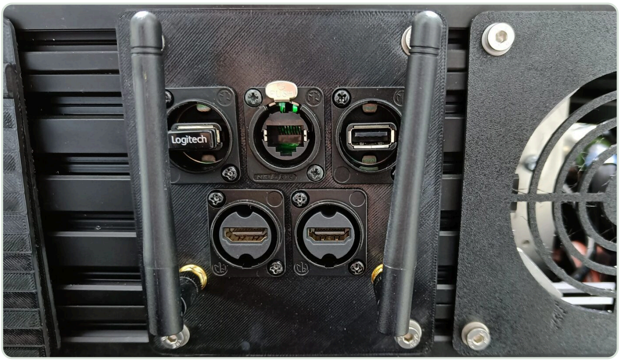
Important: The hsky_manipulation driver should be running when executing the shutdown procedure.

5 Robot Interface – Network Setup & SSH

Robot Interface

Instructions for interfacing with the robot using Ubuntu 22.04 and ROS2 Humble .

Important: This procedure should be followed after reading the manuals for the Clearpath Husky as well as the Universal Robots UR5e. Furthermore, all of the Husky's functionality can be verified via the tutorials by Clearpath. Instructions for set up can be found at Husky Docs .



_images/interface.webp

The Ethernet port at the back of the Husky can be utilized for setting up communication via LAN.

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 192.168.131.51 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 192.168.131.1
```

Access the robot via SSH:

```
ssh -X administrator@192.168.131.1
```

The default password is:

mybotshop

Husky IP Addresses

Device	IP Address	Password
Husky MCU	192.168.131.1	mybotshop
Husky Nvidia	192.168.131.4	mybotshop
Router	192.168.131.100	mybotshop
UR5e	192.168.131.5	X
Husky Hokuyo	192.168.131.21	X

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

6 Controllers – Logitech Gamepad & Steamdeck

Controllers

Logitech Controller

1. Logitech button is to power on the controller.
2. L1 is the dead man's switch for slow speed.
3. R1 is the dead man's switch for high speed.
4. LJ is to move in x and y direction.

Steamdeck Controller

_images/steamdeck.webp

Steamdeck

Note: The Steamdeck usage is in experimental phase. 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!

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. Hold the three dashes (located on the top right of the Steamdeck next to the Y button) for 3 seconds to enter gamepadmode and hold for 3 seconds again to return to desktop mode .

Initialization with Robot

Before usage:

1. Ensure the Steamdeck is connected to the robot's WiFi access point.
2. Ensure the robot is powered on and operational.
3. Double 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.)
4. Double tap the Rviz2 Launcher on the screen to visualize the robot.

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 is to move in x and y direction.
4. R5 powers ON the UR5e.
5. L5 powers OFF the UR5e only if the UR5e driver is running.

7 ROS 2 Software – Manipulator, Visualization, Lidar & Navigation

MMP Husky UR5e + Robotiq 2F-140

To power-on the UR5e and Robotiq 2F-140 completely, run:

```
ros2 service call /a200_1025/ur5e/activation/hsky_nano/srv/SerialBoolean "{data: true}"
```

Launch ROS2 driver for UR5e as well as Robotiq 2F-140:

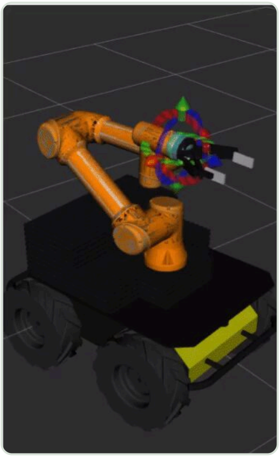
```
ros2 launch hsky_manipulation ur5e_robotiq.launch.py
```

To test the UR5e and gripper you can use Moveit2 :

```
ros2 launch hsky_ur_moveit ur_moveit.launch.py
```

To power-off the UR5e and Robotiq 2F-140 completely, run:

```
ros2 run hsky_manipulation remote_shutdown.py
```



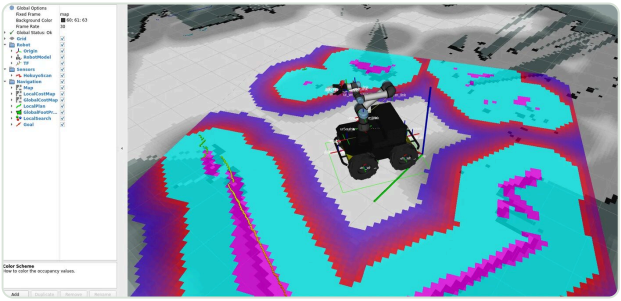
_images/usage_ur5e.gif

Important: The hsky_manipulation driver should be running when executing the shutdown procedure.

MMP Husky Viz

To visualize the Husky, you can run:

```
ros2 launch hsky_viz view_robot.launch
```



_images/rviz2.webp

MMP Husky Hokuyo

The Hokuyo is on by default from the startup job. To launch the Hokuyo only:

```
ros2 launch hsky_lidars system.launch.py
```

_images/hokuyo_mbs.webp

Hokuyo

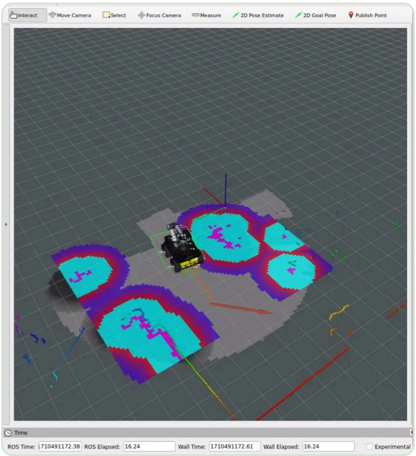
Warning: Do not run this command, as this is running in the background on startup via the hsky startup job.

MMP Husky Navigation

Navi2 — Odom Navigation

Navigate in the global frame of map

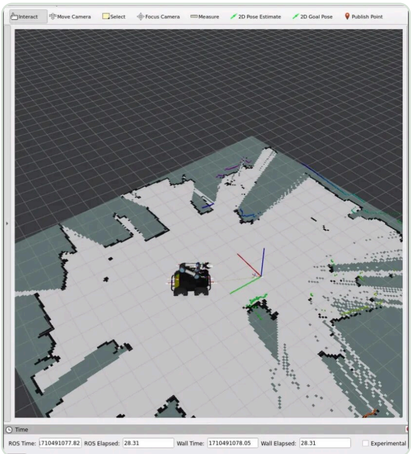
```
ros2 launch hsky_nav2 odom_navi.launch.py
```



_images/odom_based_navi.webp

Odom Based Navigation: After launching the drivers and commands, providing a 2D navigation goal in RViz initiates the navigation process. The brown arrow indicates the 2D navigation goal, while the small orange line represents the global planner and the green arrows depict the local planner. The green box surrounding the robot signifies the bounding box. In the map, blue areas represent the local cost map, and the grey area corresponds to the global cost map

Navi2 — Mapping



_images/maping_area.webp

Mapping

1. Mapping can begin using the joystick controller :

```
ros2 launch hsky_nav2 slam.launch.py
```

1. Once you are satisfied with your map you can export it by running the following command in one of the Huskys terminals:

```
ros2 run nav2_map_server map_saver_cli -f
/home/administrator/ros2_ws/src/mybotshop/hsky_nav2/maps/custom_map
```

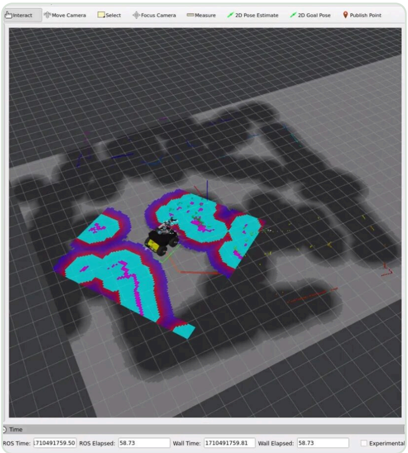
1. Rebuild the husky nav2 so that the maps can be installed in the install folder:

```
colcon build --symlink-install --cmake-args -DCMAKE_BUILD_TYPE=Release
```

Note: The rebuilding step is not required if the filename is not changed.

Navi2 — Map Navigation

```
ros2 launch hsky_nav2 map_navi.launch.py
```



_images/map_based_navi.webp

Map Based Navigation: After launching the drivers and commands, providing a 2D pose estimate in RViz in the Map frame initiates map navigation

8 ROS 2 Humble Installation & Resources

Resources

Data Sheets

- Husky Data Sheet
- Zed2i Data Sheet
- MP1-11TGS (Main PC) Data Sheet
- Telefonica RUT240 (WiFi Router) Data Sheet
- Fixposition Data Sheet
- Ouster Data Sheets
- Nvidia Jetson Orin 2GB
- RealSense Data Sheet
- Flir Data Sheet
- Ufactory xarm6 Data Sheet
- Robotiq Force Torque sensor Data Sheet
- Robotiq gripper Data Sheet
- Universal Robots UR5e Data Sheet

Documentations

- ZED2 ROS Docs
- ZED2 ROS2 Docs
- Husky Noetic ROS Docs
- Husky Foxy ROS2 Docs
- Ouster ROS Docs
- Ouster ROS2 Docs
- Nvidia Jetson Orin SDK Resources
- Fixposition ROS & ROS2 Docs
- RealSense SDK Docs
- Ufactory xarm6 Documentation
- Ufactory xarm6 ROS Docs
- Ufactory xarm6 ROS2 Docs
- Robotiq Force Torque sensor Docs
- Robotiq gripper Docs
- Universal Robots UR5e Docs

ROS2 Installation Humble

_images/humble.webp

The ROS driver for the robots is initially supported for the ROS2 Humble . All of ROS related software should be installed in a remote Husky-PC as well as the Remote-PC . The steps for shown below are for Remote-PC as the Husky is pre-installed with the required drivers and ROS distribution.

Warning: Care should be taken when installing ROS so that it matches with your CPU's architecture (i.e. armhf, amd64, arm6, etc.)

Note: All the ROS related software should be installed/run on remote PC/Rasberry Pi/Nvidia board, nothing needed to be installed on robots control board if bought with the ROS pkg.

PC Setup

ROS2 Humble installation requires Ubutnu 22.04.

1. Download Ubuntu 22.04
2. Follow the Ubuntu 22.04 installation guide

ROS Humble Installation

The ROS driver provided can be run either on a remote PC or on board robot computer. Typically, the on-board computer already has a pre-installed ROS distribution, so the instructions below will be applicable to a remote computer. You may execute the following commands to install ROS Humble, or you can simply follow the instructions from the ROS Wiki.

Set Locale

Ensure that your system supports UTF-8 locale. Verify your locale settings with the following command:

```
locale
```

If necessary, set up a UTF-8 locale:

```
sudo apt update && sudo apt install locales
sudo locale-gen en_US en_US.UTF-8
sudo update-locale LC_ALL=en_US.UTF-8 LANG=en_US.UTF-8
export LANG=en_US.UTF-8
```

Setup Sourcess

Add the ROS2 apt repository to your system. First, make sure that the Ubuntu Universe repository Ubuntu Universe repository is enabled:

```
sudo apt install software-properties-common
sudo add-apt-repository universe
```

Next, add the ROS2 GPG key with apt:

```
sudo apt update && sudo apt install curl -y
sudo curl -sSL https://raw.githubusercontent.com/ros/rosdistro/master/ros.key -o
/usr/share/keyrings/ros-archive-keyring.gpg
```

Then, add the repository to your sources list:

```
echo "deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/ros-archive-
keyring.gpg] http://packages.ros.org/ros2/ubuntu $(. /etc/os-release && echo
$UBUNTU_CODENAME) main" | sudo tee /etc/apt/sources.list.d/ros2.list > /dev/null
```

Install ROS2 Packages

Update your apt repository caches after setting up the repositories:

```
sudo apt update
```

Ensure your system is up to date before installing new packages:

```
sudo apt upgrade
```

Warning: Due to early updates in Ubuntu 22.04 it is important that systemd and udev -related packages are updated before installing ROS2. The installation of ROS2's dependencies on a freshly installed system without upgrading can trigger the removal of critical system packages . Please refer to ros2/ros2#1272 and Launchpad #1974196 for more information.

Desktop Install (Recommended)

Install ROS, RViz, demos, and tutorials:

```
sudo apt install ros-humble-desktop
```

ROS-Base Install (Bare Bones)

Install communication libraries, message packages, and command line tools without GUI tools:

```
sudo apt install ros-humble-ros-base
```

Development Tools

Install compilers and other tools to build ROS packages:

```
sudo apt install ros-dev-tools
```

Environment Setup

Source the setup script to set up your environment:

```
# Replace ".bash" with your shell if you're not using bash
# Possible values are: setup.bash, setup.sh, setup.zsh
source /opt/ros/humble/setup.bash
```

Try Some Examples

If you installed Desktop Install (Recommended) above, try some examples:

Run a C++ talker in one terminal:

```
source /opt/ros/humble/setup.bash
ros2 run demo_nodes_cpp talker
```

Run a Python listener in another terminal:

```
source /opt/ros/humble/setup.bash
ros2 run demo_nodes_py listener
```

This verifies both the C++ and Python APIs are working properly.

Next Steps After Installing

Continue with the tutorials and demos to configure your environment, create your own workspace and packages, and learn ROS2 core concepts.

Using the ROS1 Bridge

Connect topics from ROS1 to ROS2 and vice-versa using the ROS1 bridge. See the dedicated documentation on how to build and use the ROS1 bridge.

Additional RMW Implementations (Optional)

ROS2 uses Fast DDS as the default middleware (RMW), but you can replace it at runtime. See the guide on how to work with multiple RMWs.

Troubleshooting

Refer to the troubleshooting techniques found here if you encounter any issues.

Uninstall

If you need to uninstall ROS2 or switch to a source-based install once you have already installed from binaries, run the following command:

```
sudo apt remove ~nros-humble-* && sudo apt autoremove
```

You may also want to remove the repository:

```
sudo rm /etc/apt/sources.list.d/ros2.list
sudo apt update
sudo apt autoremove
# Consider upgrading for packages previously shadowed.
sudo apt upgrade
```

9 Field Deployment: Husky ATC (Ouster + Blickfeld + ZED2i + Qoocam + xArm6)

MMP Husky ATC

_images/husky_atc_side.webp

This specifies the configuration for Husky ATC. It has an Ouster lidar, Nvidia Orin, Zed2i depth camera, Cradlepoint Router, Qoocam, xARM6, Robotiq 2F-140 gripper, and Blickfeld Lidar.

Getting Started

To start-up the Husky, the instructional video playlist can be used which details the opening and use of the Husky.

Data Sheets

- Husky Data Sheet
- Phidgets Data Sheet
- Zed2i Data Sheet
- Ouster Data Sheets
- Nvidia Orin AGX Datasheet

Documentations

- Husky Noetic ROS Docs
- ZED2 ROS Docs
- xARM6 User Manual
- xARM6 Developer Manual
- xARM6 Github
- Ouster ROS Docs

- Cradlepoint R1900 Router Resources
- Blickfeld Cube 1 Resources
- Qoocam 8k Resources

Auxiliary Sensor Startup

On husky startup the following drivers are started:

- Husky base driver
- Ouster
- Blickfeld

The xARM6, Robotiq 2F-140, Ouster, Zed2, and Blickfeld are powered on when the Husky robot starts. However, only the software drivers for Ouster and Blickfeld are activated on start-up.

For Qoocam, the camera has to manually be turned on and be set to live mode for the camera to provide a livestream. More on this in the Qoocam section.

Network-Setup

The Husky ATC has multiple networks and vary slightly, the following table shows the networks.

Device	Network Address	Users	Password
Husky	192.168.131.1	administrator	clearpath
Nvidia	192.168.131.2	administrator	clearpath
Blickfeld	192.168.131.3	N/A	N/A
xARM6	192.168.131.4	N/A	N/A
xARM6 Web	192.168.131.4:18333	N/A	N/A
Qoocam	192.168.131.5:8554	N/A	N/A
Ouster	192.168.132.1	N/A	N/A
Router	192.168.131.101	root	clearpath

Operation

To use the connect and use the MMP Husky, the following steps must be followed:

1. Connect to the Husky WiFi (Password is clearpath).
2. ssh into the husky via:

```
$ ssh -X administrator@192.168.131.1
$ clearpath
```

The following message will show up with quick instructions for startup.

```
--------------------------------------------------------------------------------
__  ____     ______   ____ _______ _____ _    _  ____  _____            _____
|  \/  \ \   / /  _ \ / __ \__   __/ ____| |  | |/ __ \|  __ \          / ____|
| \  / |\ \_/ /| |_) | |  | | | | | (___ | |__| | |  | | |__) |  _   _| |  __
| |\/| | \   / |  _ <| |  | | | |  \___ \|  __  | |  | |  ___/  | | | | | |_ |
| |  | |  | |  | |_) | |__| | | |  ____) | |  | | |__| | |      | |_| | |__| |
|_|  |_|  |_|  |____/ \____/  |_| |_____/|_|  |_|\____/|_|       \__,_|\_____|

Current PC: Husky
--------------------------------------------------------------------------------

Husky IP:     192.168.131.1 (ssh -X administrator@192.168.131.1)
Pswd:         clearpath

Nvidia IP:    192.168.131.2 (ssh -X administrator@192.168.131.2)
Pswd:         clearpath

Blick IP:     192.168.131.3

xARM IP:      192.168.131.4
xARM IP Web:  192.168.131.4:18333

Qoocam IP:     192.168.131.5:8554 (RTSP)

Ouster IP:    192.168.132.1

WiFi IP:      192.168.131.101
Pswd:         clearpath
Router Name:  root

--------------------------------------------------------------------------------
Husky-Drivers
--------------------------------------------------------------------------------

Husky status:     sudo service mbs_husky status
Husky start:      sudo service mbs_husky start
Husky stop:       sudo service mbs_husky stop
Husky restart:    sudo service mbs_husky restart

View robot:       roslaunch mbs_husky_viz viz.launch

--------------------------------------------------------------------------------
Auxiliary driver:
--------------------------------------------------------------------------------

Ouster driver:   roslaunch mbs_husky_lidars top_laser.launch
                (Launches on startup)

Blick driver:    roslaunch mbs_husky_lidars top_laser.launch
                (Launches on startup)

xARM driver:     roslaunch mbs_husky_xarm_interface real_arm_moveit.launch
                (Doesnt launch on startup)
```

```
---------------------------------------------------------------------------------
```

Custom Startup Job

The launch file mbs_husky.launch in the catkin package of mbs_husky_startup includes everything which is being launched at the startup. Startup job and its content can be verified in the directory /etc/ros/noetic , here the original launch file will be copied.

> **Warning:** Do not create multiple startup jobs with multiple launch files this will break the system. Only one launch file should be created and everything else should be included in this.

If the systems breaks at startup you can start debugging by:

```
roslaunch mbs_husky_startup mbs_husky.launch
```

This will show which part of the launch file is having errors or you can start running everything in the launch file one by one in separate terminals.

Adding ROS-Nodes & Launch Files to Robot Startup

1. Navigate and open mbs_husky_startup/launch/mbs_husky.launch .

```
<?xml version="1.0"?>
<launch>

  <!-- Launch standard Husky base drivers -->
  <include file="$(find mbs_husky_startup)/launch/include/husky_base.launch" />
  <!-- <include file="$(find husky_base)/launch/base.launch" /> -->

  <!-- UM7 IMU Driver -->
  <include file="$(find mbs_husky_startup)/launch/include/um7_imu_driver.launch" />

  <!-- Phidgets IMU Driver -->
  <!-- <include file="$(find mbs_husky_startup)/launch/include/phidget_imu_driver.launch" /> -->

  <!-- Odom Localization Driver -->
  <!-- <include file="$(find mbs_control)/launch/wheel_ekf.launch" />  -->

  <!-- Launch Power Relay -->
  <include file="$(find mbs_husky_startup)/launch/include/power_relay.launch" />

</launch>
```

_images/mbs_husky_launch.webp

1. Add your custom node or launch file.
2. Run the following script:

```
rosrun mbs_husky_startup startup_script.sh
```

1. Verify if it is working correctly via:

```
sudo service mbs_husky status
```

> **Important:** Once you add your custom node or launch files in the mbs_husky.launch , you will need to run rosrun mbs_husky_startup startup_script.sh so that it updates the start-up job on the Husky's PC otherwise it will not be updated.

Ouster

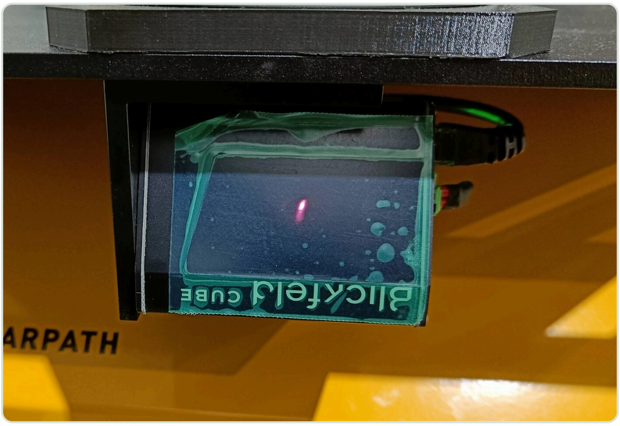
_images/husky_atc_ouster.webp

To manually launch the ouster driver, the following command can be used:

```
roslaunch mbs_lidars top_laser.launch
```

Important: Please note that the lidars launch on start-up. Running it again will shut them down. They can be disabled by following the instuctions in the Custom Startup Job .

Blickfeld



_images/husky_atc_blickfeld.webp

To manually launch the blickfeld driver, the following command can be used:

```
roslaunch mbs_lidars front_laser.launch
```

Important: Please note that the lidars launch on start-up. Running it again will shut them down. They can be disabled by following the instuctions in the Custom Startup Job

ZED2i

_images/husky_atc_zed2i.webp

On the robot start-up the Nvidia Orin starts, however, the Zed2i has to be manually launched.

- ssh into the Nvidia Orin via:

```
ssh -X administrator@192.168.131.2
clearpath
```

- To launch the ZED2i run the command.

```
roslaunch zed_wrapper zed2i.launch
```

Customization of the camera position can be done here via the launch file.

> **Note:** It may take upto a minute for all the drivers of the Nvidia to start. The stream for the ZED2 is transferred over to the main Husky PC via ROS multi machine. An example stream can be viewed rostopic echo /zed2/zed2_node/rgb/image_rect_color .

The information can be verified via:

```
rostopic list
```

> **Important:** The ZED2i requires USB 3.0 port and in the installed Nvidia board there is one available USB 3.0 port. So incase of removal please plug in the ZED2i into the same initial port it was in.

Qoocam

_images/husky_atc_qoocam.webp

To start the Qoocam driver.

1. First you must power on the Qoocam by pressing the power button.
2. Once Qoocam is powered on, swipe left on its touch screen and launch live mode.
3. Ensure that the ethernet cable is connected. The router will automatically assign it the ip of 192.168.131.5 .
4. Click the middle button of the Qoocam to start the RTSP stream.
5. Now launch the qoocam driver in the Husky via:

```
roslaunch mbs_qoocam qoocam_to_ros.launch
```

The driver will fail if it does not receive the camera stream.

Note: Please note that the Qoocam has limited battery life and must be charged fully for optimal use.

xARM6

Warning: Please disconnect the xARM6 when removing the batter to ensure that the connector is not damaged. Also ensure that there is sufficient charge in the Husky when using the xARM6.

- The xARM6 firstly requires its power connector to be connected. It is located next to the battery of the Husky.
- Once connected, launch the xARM6 hardware driver, via the command:

```
roslaunch mbs_husky_xarm_interface real_arm_moveit.launch
```

An emergency power cut-off for the xARM6 is available in the auxiliary frame of the Husky.

_images/husky_atc_emergencystop.webp

Robotiq 2F-140

- To launch the Robotiq 2f-140 hardware driver, run the command:

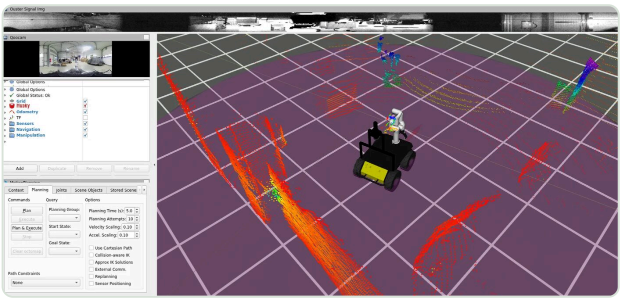
```
roslaunch mbs_xarm_robotiq robotiq_hardware_driver.launch
```

> **Important:** The robtiq is controlled via ros action client. Examples are provided in the mbs_xarm_robotiq/nodes/test_robotiq_gripper.py file.

Visualization

- The visualize the robot model.

```
roslaunch mbs_viz viz.launch
```



_images/husky_atc_viz.webp

MMP Husky xARM6 URDF

10 Field Deployment: Husky TUM (xArm6 + Robotiq 2F-140)

Note: The Robotic Manipulator and Autonomous Mobile Robot safety guideline sections that originally appeared at the end of this unit's documentation are identical across the TUM, BE and FRA units and have been consolidated into a single shared chapter — see Chapter 15, Robotic Manipulator & AMR Safety Guidelines.

MMP Husky TUM

_images/husky_tum_1.webp

Table of Contents

Getting Started

To start-up the Husky, the instructional video playlist can be used which details the opening and use of the Husky.

Platform Overview

The Husky TUM project is equipped with a state-of-the-art configuration tailored to meet its diverse operational requirements. At its core, it features the Clearpath Husky (IP44) platform, powered by a Mitac MP1-11TGS-1185G7E processor with 16GB of RAM and a 500GB SSD, ensuring efficient data processing and ample storage capabilities.

The robotic arm utilized in the system is the UFactory xArm6, controlled via the UFactory xArm DC Control Box, offering precise manipulation and versatility in performing various tasks. Additionally, the system integrates the ROBOTIQ 2-Finger-Gripper 2F-140 CB3, which provides dexterous and reliable object handling. The Phidgets Spatial 3/3/3 Basic IMU enhances the robot's spatial awareness, enabling improved navigation and stability.

Sensory input is enriched by two cutting-edge cameras: the Intel® RealSense Depth Camera D405, mounted as a wrist camera on xArm, and the Intel® RealSense Depth Camera D455f, mounted on an aluminum frame for enhanced environmental perception. Accurate environmental mapping and obstacle detection are facilitated by the HOKUYO UST-20LX LiDAR.

This advanced setup ensures robust performance across a wide range of tasks, making the Husky TUM a versatile and powerful platform for modern robotic applications.

Data Sheets

- Husky Data Sheet
- MP1-11TGS (Main PC) Data Sheet
- Ouster Data Sheets
- RealSense Data Sheet
- Robotiq gripper Data Sheet
- Ufactory xarm6 Data Sheet

Documentations

- Husky Foxy ROS2 Docs
- Ouster ROS Docs
- Ouster ROS2 Docs
- RealSense SDK Docs
- Robotiq gripper Docs
- Ufactory xarm6 Documentation
- Ufactory xarm6 ROS Docs
- Ufactory xarm6 ROS2 Docs

Robot Interface

Instructions for interfacing with the robot using Ubuntu 22.04 and ROS2 Humble .

Important: This procedure should be followed after reading the manuals for the Clearpath Husky as well as the Ufactory xArm6. Furthermore, all of the Husky's functionality can be verified via the tutorials by Clearpath. Instructions for set up can be found at Husky Docs .

_images/hsky_tum_interface.webp

The Ethernet port at the top of the Husky can be utilized for setting up communication via LAN.

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 192.168.131.51 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 192.168.131.1
```

Access the robot via SSH:

```
ssh -X administrator@192.168.131.1
```

The default password is:

mybotshop

Husky IP Addresses

Device	IP Address	Password
Husky MCU	192.168.131.1	mybotshop
Router	192.168.131.100	mybotshop
xArm	192.168.131.5	X
Husky Hokuyo	192.168.131.20	X
Husky Ouster	192.168.131.21	X

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

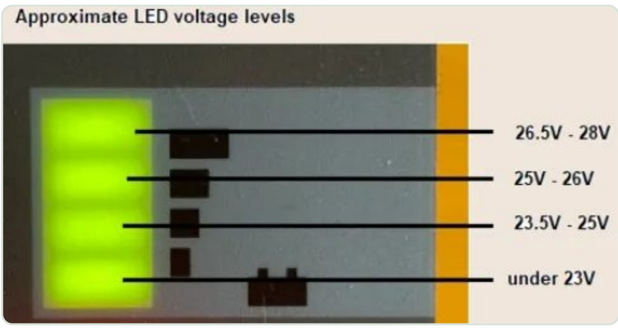
Quick-Start

Powering on the Husky

Power Button	Emergency Switch Release	Charging Port

- Ensure the emergency button is released as well as the key is turned to the unlock position. ◦ To check if the emergency switch is released, twist the switch in the indicated direction on it. If it pops up, then the switch has successfully been released. ◦ If the robot light is yellow , it means that either the emergency switch is not released or the key is not turned.
- Press the power button on the front of the Husky. It has the power symbol as displayed.

Charging the Husky



_images/charging_indicator.webp

LED Voltage Indicators

The power port can be used to charge the Husky. The indicator shows the voltage levels of the Husky and does not necessarily indicate the remaining power available.

Manipulator Activation

_images/hsky_tum_side.webp

To power on the xARM6 and Robotiq 2F-140 :

```
ros2 launch hsky_xarm system.launch.py
```

This command launches the driver for the xARM6 , integrates with MoveIt2 , and activates the Robotiq 2F-140 gripper for manipulation tasks.

Gripper Control

To control the Robotiq 2F-140 gripper:

- Open Gripper : ros2 action send_goal /a200_1058/GripperPosition hsky_msgs/action/GripperPosition final_position: \ 0
- Close Gripper : ros2 action send_goal /a200_1058/GripperPosition hsky_msgs/action/GripperPosition final_position: \ 255

Important: Ensure that the hsky_xarm driver is running when using the gripper control commands.

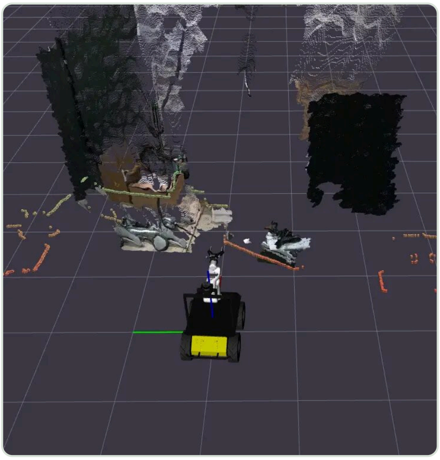
Logitech Controller

1. Logitech button is to power on the controller.
2. L1 is the dead man's switch for slow speed.
3. R1 is the dead man's switch for high speed.
4. LJ is to move in x and y direction.

Digital Twin (RVIZ2)

To visualize the Husky, you can run:

```
ros2 launch hsky_viz view_robot.launch
```



_images/husky_tum_rviz.webp

ROS2 Drivers | Start-up

The ROS2 drivers mentioned below are all automatically started when launching the system launch file. These drivers further expand on the functionality for interfacing as well as for independent component checking.

- clearpath-robot — Manages all Clearpath Husky-related services.
- clearpath-platform — Handles Husky's hardware-related drivers.
- mbs-hokuyo — Activates the Hokuyo UST-20LX LiDAR for environmental perception.
- mbs-realsense-i — Activates the Intel® RealSense Depth Camera D455f (mounted on an aluminum frame).
- mbs-realsense-ii — Activates the Intel® RealSense Depth Camera D405 (mounted on the arm's wrist).
- mbs-localization — Manages EKF localization and spatial awareness using the PhidgetSpatial IMU.

These services can be queried, started, stopped, or restarted via the following commands:

```
sudo service {service_name} status
sudo service {service_name} start
sudo service {service_name} stop
sudo service {service_name} restart
```

Example

To check the status of the clearpath-robot service:

```
sudo service clearpath-robot status
```

Uninstalling a Service

To remove a service:

```
ros2 run robot_upstart uninstall {service_name}
```

If you encounter an error such as:

```
Unit {service_name}.service could not be found.
```

This indicates that the startup job has not been installed.

Red marker : The startup job has failed.

Green marker : The service is functioning correctly.

Grey marker : The service has not started yet.

In case of a red or grey marker, you can restart the service using:

```
sudo service {service_name} restart
```

Modifying the Upstart Job

To modify the upstart job or add other ROS launch files, it is recommended to edit the main upstart file, system.launch.py , located in the hsky_bringup package.

After making the changes, save the file and run the following command to update the startup job:

```
ros2 run hsky_bringup startup_installer.py
```

This ensures that all necessary services are configured and running optimally for the Husky TUM project.

ROS2 Driver — xArm6 + Robotiq 2F-140

To power on the xARM6 and Robotiq 2F-140 :

```
ros2 launch hsky_xarm system.launch.py
```

This command launches the driver for the xARM6 , integrates with MoveIt2 , and activates the Robotiq 2F-140 gripper for manipulation tasks.

Gripper Control

To control the Robotiq 2F-140 gripper:

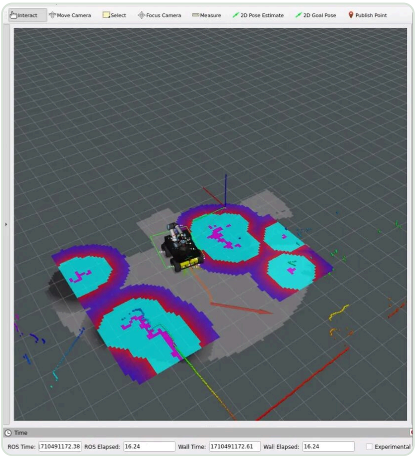
- Open Gripper : ros2 action send_goal /a200_1058/GripperPosition hsky_msgs/action/GripperPosition final_position: \ 0
- Close Gripper : ros2 action send_goal /a200_1058/GripperPosition hsky_msgs/action/GripperPosition final_position: \ 255

ROS2 Navigation

Navi2 — Odom Navigation

Navigate in the global frame of map

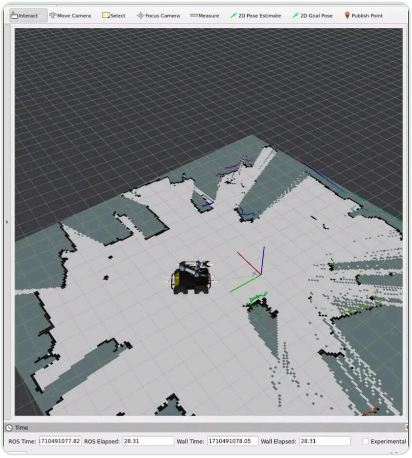
```
ros2 launch hsky_nav2 odom_navi.launch.py
```



_images/odom_based_navi.webp

Odom Based Navigation: After launching the drivers and commands, providing a 2D navigation goal in RViz initiates the navigation process. The brown arrow indicates the 2D navigation goal, while the small orange line represents the global planner and the green arrows depict the local planner. The green box surrounding the robot signifies the bounding box. In the map, blue areas represent the local cost map, and the grey area corresponds to the global cost map

Navi2 — Mapping



_images/maping_area.webp

Mapping

1. Mapping can begin using the joystick controller :

```
ros2 launch hsky_nav2 slam.launch.py
```

1. Once you are satisfied with your map you can export it by running the following command in one of the Huskys terminals:

```
ros2 run nav2_map_server map_saver_cli -f
/home/administrator/ros2_ws/src/mybotshop/hsky_nav2/maps/custom_map
```

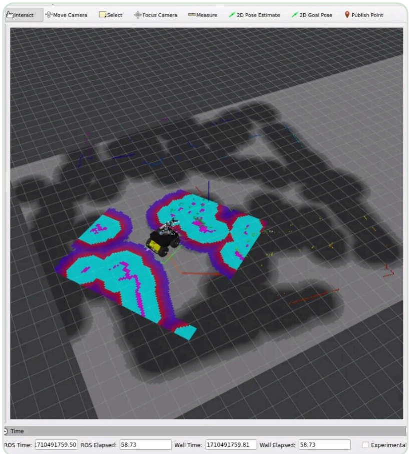
1. Rebuild the husky nav2 so that the maps can be installed in the install folder:

```
colcon build --symlink-install --cmake-args -DCMAKE_BUILD_TYPE=Release
```

Note: The rebuilding step is not required if the filename is not changed.

Navi2 — Map Navigation

```
ros2 launch hsky_nav2 map_navi.launch.py
```



_images/map_based_navi.webp

Map Based Navigation: After launching the drivers and commands, providing a 2D pose estimate in RViz in the Map frame initiates map navigation

ROS2 Information

1. ROS DOMAIN ID: 0
2. RMW Middleware: rmw fastrtps cpp

Caution

- Please read the instruction manuals pertaining to Clearpath Husky and Ufactory xAr6 before operating the robot.
- Always be vigilant and remain close to the emergency stop buttons when the robot is in operation.

Known Issues

1. Husky moving forward on startup This may occur due to time synchronization of the internet causing the robot's joint states to reverse. This is dependent on how the network is synchronized between the Husky's computer and the internet. By default we have disabled synchronization of time from the internet to prevent this issue, until Clearpath updates its drivers.

Frequently Asked Questions

1. Logitech controller not working Ensure that it is powered on. Ensure that the input mode is X, which is set on the backside of the controller.
2. Husky in standby mode (yellow color) Ensure that the emergency switch is twisted and released. Ensure that the key is in the unlock position.
3. xArm6 does not move with Moveit2 . The solution to this to simply restart the driver.
4. Unable to see some of the ros2 topic lists in remote PC. Ensure you are in the same network. Ensure that your ROS DOMAIN ID is 0. Ensure that your ROS LOCAL HOST is set to 0. Ensure that your network system (+router) allows for ROS2 communication and/or is configured. Restart the robot services: ◦ sudo service clearpath-robot restart ◦ sudo service hsky restart

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:

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.

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.

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.

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.

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.

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.

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 Field Deployment: Husky BE (UR5e + Robotiq 2F-140 + Hokuyo + Steamdeck)

Note: The Robotic Manipulator and Autonomous Mobile Robot safety guideline sections that originally appeared at the end of this unit's documentation are identical across the TUM, BE and FRA units and have been consolidated into a single shared chapter — see Chapter 15, Robotic Manipulator & AMR Safety Guidelines.

MMP Husky BE

_images/husky_ur5e.webp

Table of Contents

Getting Started

To start-up the Husky, the instructional video playlist can be used which details the opening and use of the Husky.

Platform Overview

The Husky BE project is equipped with a powerful configuration tailored to meet its diverse operational requirements. It features the robust Husky A200 platform, boasting an Mitac MPi11 i7 processor paired with 16GB of RAM and a 480GB SSD for efficient data processing and storage capabilities. Its manipulator arm consists of a UR5e CB5 DC arm, offering precise control and versatility in handling various tasks. The inclusion of a Phidgets IMU enhances the robot's spatial awareness and navigation capabilities. Furthermore, the ROBOITQ 2-Finger Gripper 140mm enables dexterous manipulation of objects with ease.

Sensory inputs are provided by the HOKUYO UTM30-LX-EW lidar for accurate environmental perception. The system is powered by the NVIDIA AGX ORIN 64GB board, ensuring robust computation for complex tasks. Additional sensors include an Arduino equipped with two 18B20 sensors housed within the Husky chassis. To facilitate wireless communication, a TP-Link WiFi Router is integrated into the setup, enabling seamless connectivity and control of the robot.

Data Sheets

- Husky Data Sheet
- Zed2i Data Sheet
- MP1-11TGS (Main PC) Data Sheet
- Telefonica RUT240 (WiFi Router) Data Sheet
- Fixposition Data Sheet
- Ouster Data Sheets
- Nvidia Jetson Orin 2GB
- RealSense Data Sheet
- Robotiq Force Torque sensor Data Sheet
- Robotiq gripper Data Sheet

Documentations

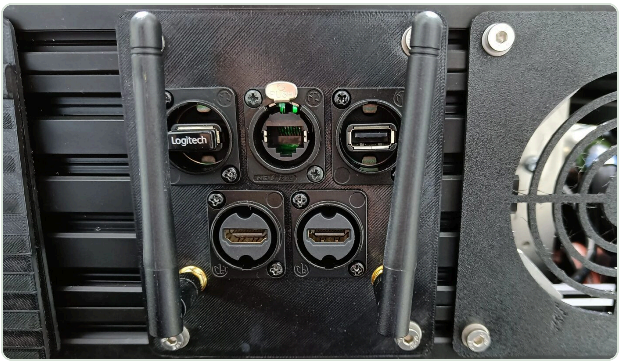
- ZED2 ROS Docs
- ZED2 ROS2 Docs
- Husky Noetic ROS Docs
- Husky Foxy ROS2 Docs

- Ouster ROS Docs
- Ouster ROS2 Docs
- Nvidia Jetson Orin SDK Resources
- Fixposition ROS & ROS2 Docs
- RealSense SDK Docs
- Robotiq Force Torque sensor Docs
- Robotiq gripper Docs

Robot Interface

Instructions for interfacing with the robot using Ubuntu 22.04 and ROS2 Humble .

Important: This procedure should be followed after reading the manuals for the Clearpath Husky as well as the Universal Robots UR5e. Furthermore, all of the Husky's functionality can be verified via the tutorials by Clearpath. Instructions for set up can be found at Husky Docs .



_images/interface.webp

The Ethernet port at the back of the Husky can be utilized for setting up communication via LAN.

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 192.168.131.51 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 192.168.131.1
```

Access the robot via SSH:

```
ssh -X administrator@192.168.131.1
```

The default password is:

mybotshop

Husky IP Addresses

Device	IP Address	Password
Husky MCU	192.168.131.1	mybotshop
Husky Nvidia	192.168.131.4	mybotshop
Router	192.168.131.100	mybotshop
UR5e	192.168.131.5	X
Husky Hokuyo	192.168.131.21	X

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

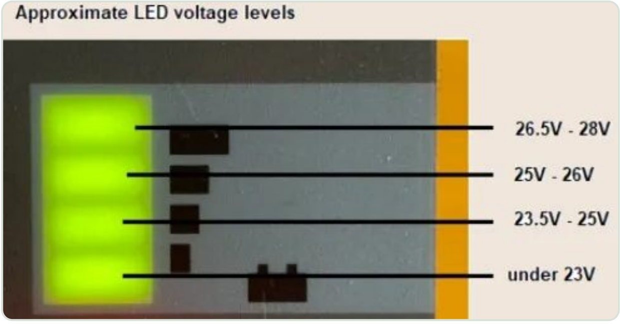
Quick-Start

Powering on the Husky

Power Button	Emergency Switch Release	Charging Port

- Ensure the emergency button is released as well as the key is turned to the unlock position. ◦ To check if the emergency switch is released, twist the switch in the indicated direction on it. If it pops up, then the switch has successfully been released. ◦ If the robot light is yellow , it means that either the emergency switch is not released or the key is not turned.
- Press the power button on the front of the Husky. It has the power symbol as displayed.

Charging the Husky



_images/charging_indicator.webp

LED Voltage Indicators

The power port can be used to charge the Husky. The indicator shows the voltage levels of the Husky and does not necessarily indicate the remaining power available.

Manipulator Activation

To power-on the UR5e and Robotiq 2F-140:

```
ros2 service call /a200_1025/ur5e_activation/hsky_nano/srv/SerialBoolean "{data: true}"
```

Launch ROS2 driver for UR5e as well as Robotiq 2F-140:

```
ros2 launch hsky_manipulation ur5e_robotiq.launch.py
```

To test the UR5e and gripper you can use Moveit2 :

```
ros2 launch hsky_ur_moveit ur_moveit.launch.py
```

To power-off the UR5e and Robotiq 2F-140 completely, run:

```
ros2 run hsky_manipulation remote_shutdown.py
```

Important: The hsky_manipulation driver should be running when executing the shutdown procedure.

Logitech Controller

1. Logitech button is to power on the controller.
2. L1 is the dead man's switch for slow speed.
3. R1 is the dead man's switch for high speed.
4. LJ is to move in x and y direction.

Steamdeck Controller

_images/steamdeck.webp

Steamdeck

Note: The Steamdeck usage is in experimental phase. 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!

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. Hold the three dashes (located on the top right of the Steamdeck next to the Y button) for 3 seconds to enter gamepadmode and hold for 3 seconds again to return to desktop mode .

Steamdeck - Initialization with Robot

Before usage:

1. Ensure the Steamdeck is connected to the robot's WiFi access point.
2. Ensure the robot is powered on and operational.
3. Double 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.)
4. Double tap the Rviz2 Launcher on the screen to visualize the robot.

_images/husky_ure5e.gif

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 is to move in x and y direction.
4. R5 powers ON the UR5e.
5. L5 powers OFF the UR5e only if the UR5e driver is running.

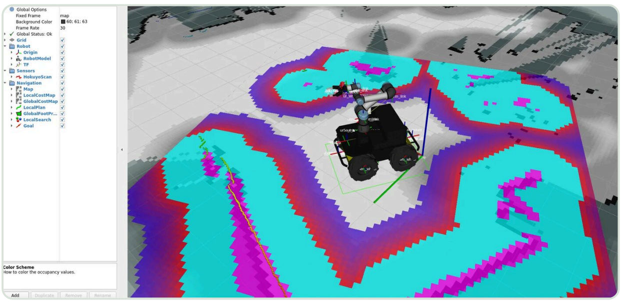
Steamdeck - Rviz2 Controls

1. L2+R2 should be held when wanting to move the orbit in the xy plane.
2. R2 should be held to orbit around the orbit's origin.
3. Left trackpad can be moved up and down to zoom in and zoom out.

Digital Twin (RVIZ2)

To visualize the Husky, you can run:

```
ros2 launch hsky_viz view_robot.launch
```



_images/rviz2.webp

ROS2 Drivers — MBS

The ROS2 drivers mentioned below are all automatically started when launching the system launch file. These drivers further expand on the functionality for interfacing as well as for independent component checking.

ROS2 Driver — UR5e + Robotiq 2F-140

To power-on the UR5e and Robotiq 2F-140 completely, run:

```
ros2 service call /a200_1025/ur5e/activation/hsky_nano/srv/SerialBoolean "{data: true}"
```

Launch ROS2 driver for UR5e as well as Robotiq 2F-140:

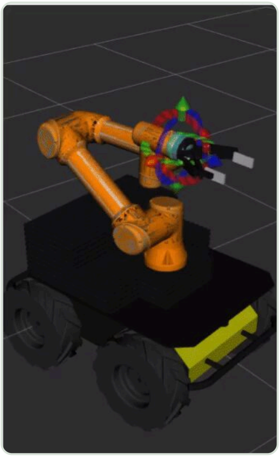
```
ros2 launch hsky_manipulation ur5e_robotiq.launch.py
```

To test the UR5e and gripper you can use Moveit2 :

```
ros2 launch hsky_ur_moveit ur_moveit.launch.py
```

To power-off the UR5e and Robotiq 2F-140 completely, run:

```
ros2 run hsky_manipulation remote_shutdown.py
```



_images/usage_ur5e.gif

Important: The hsky_manipulation driver should be running when executing the shutdown procedure.

ROS2 Driver — Hokuyo

The Hokuyo is on by default from the startup job. To launch the Hokuyo only:

```
ros2 launch hsky_lidars system.launch.py
```

_images/hokuyo_mbs.webp

Hokuyo

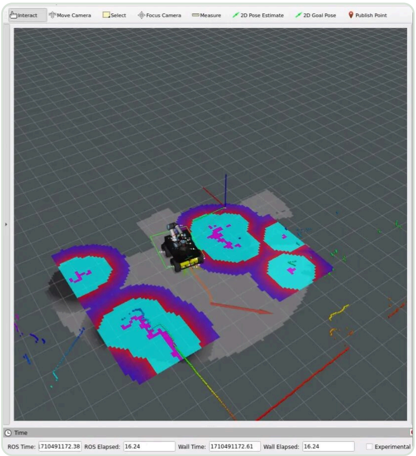
Warning: Do not run this command, as this is running in the background on startup via the hsky startup job.

ROS2 Navigation

Navi2 — Odom Navigation

Navigate in the global frame of map

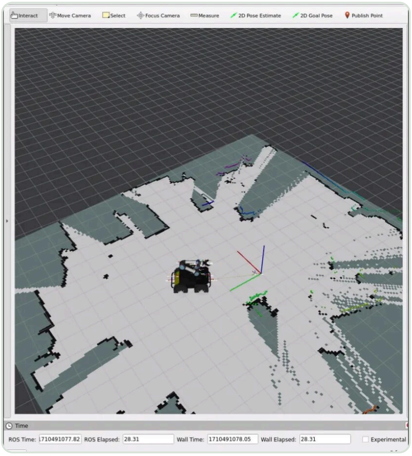
```
ros2 launch hsky_nav2 odom_navi.launch.py
```



_images/odom_based_navi.webp

Odom Based Navigation: After launching the drivers and commands, providing a 2D navigation goal in RViz initiates the navigation process. The brown arrow indicates the 2D navigation goal, while the small orange line represents the global planner and the green arrows depict the local planner. The green box surrounding the robot signifies the bounding box. In the map, blue areas represent the local cost map, and the grey area corresponds to the global cost map

Navi2 — Mapping



_images/maping_area.webp

Mapping

1. Mapping can begin using the joystick controller :

```
ros2 launch hsky_nav2 slam.launch.py
```

1. Once you are satisfied with your map you can export it by running the following command in one of the Huskys terminals:

```
ros2 run nav2_map_server map_saver_cli -f
/home/administrator/ros2_ws/src/mybotshop/hsky_nav2/maps/custom_map
```

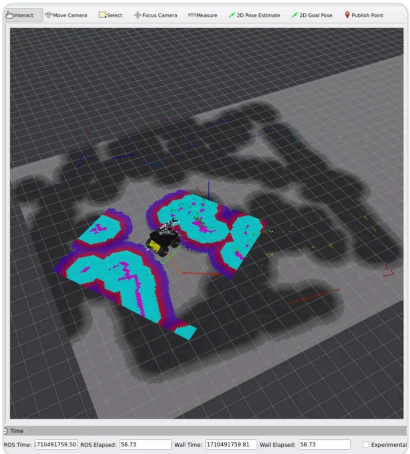
1. Rebuild the husky nav2 so that the maps can be installed in the install folder:

```
colcon build --symlink-install --cmake-args -DCMAKE_BUILD_TYPE=Release
```

Note: The rebuilding step is not required if the filename is not changed.

Navi2 — Map Navigation

```
ros2 launch hsky_nav2 map_navi.launch.py
```



_images/map_based_navi.webp

Map Based Navigation: After launching the drivers and commands, providing a 2D pose estimate in RViz in the Map frame initiates map navigation

Navi2 — Dependencies

```
sudo apt-get install ros-$ROS_DISTRO-twist-mux \
                     ros-$ROS_DISTRO-navigation2 \
                     ros-$ROS_DISTRO-nav2-bringup
```

ROS2 Information

1. ROS DOMAIN ID: 0
2. RMW Middleware: rmw fastrtps cpp

Caution

- Please read the instruction manuals pertaining to Clearpath Husky and Universal Robots UR5e before operating the robot.
- Always be vigilant and remain close to the emergency stop buttons when the robot is in operation.

Known Issues

1. Husky moving forward on startup This may occur due to time synchronization of the internet causing the robot's joint states to reverse. This is dependent on how the network is synchronized between the Husky's computer and the internet. By default we have disabled synchronization of time from the internet to prevent this issue, until Clearpath updates its drivers.

Frequently Asked Questions

1. Logitech controller not working Ensure that it is powered on. Ensure that the input mode is X, which is set on the backside of the controller.
2. Husky in standby mode (yellow color) Ensure that the emergency switch is twisted and released. Ensure that the key is in the unlock position.
3. UR5e does not move with Moveit2 . The solution to this to simply restart the driver.
4. UR5e driver not starting. This may happen if the emergency stop button is not connected to the UR5e control box. This causes the SD card in the control box to eject. The solution is to plug in back the emergency switch back and insert the SD card back into the UR5e. At times the ROS2 driver may not immediately detect the UR5e, hence, restarting the UR5e driver commonly resolves the issue.
5. UR5e driver not powering off. • Restart the UR5e driver.
6. Unable to see some of the ros2 topic lists in remote PC. Ensure you are in the same network. Ensure that your ROS DOMAIN ID is 0. Ensure that your ROS LOCAL HOST is set to 0. Ensure that your network system (+router) allows for ROS2 communication and/or is configured. Restart the robot services: ◦ sudo service clearpath-robot restart ◦ sudo service hsky restart
7. Steamdeck right trackpad not moving the mouse. Click the 3 dots below the right trackpad. If 3 dots does not work, then hold the 3 bars for 3 seconds next to the Y button. Do this twice.

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:

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.

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.

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.

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.

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.

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.

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.

12 Field Deployment: Husky FH (ROS1 Noetic + ROS2 Foxy, xArm6 + FT-300 + FLIR)

MMP Husky FH

_images/hxg.webp

This specifies the configuration for Husky FH. It has an ouster lidar, Fixposition gps, nvidia board, ZED2 depth camera, Flir thermal camera, Realsense depth camera, Ufactory xarm6,Robotiq gripper, Robotiq Force torque sensor and router to establish wireless connection with the robot via WiFi.

Getting Started

To start-up the Husky, the instructional video playlist can be used which details the opening and use of the Husky.

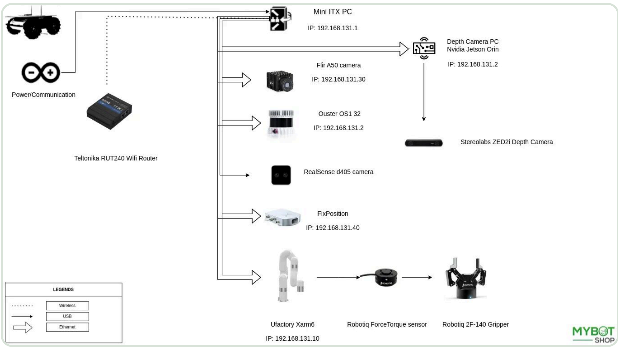
Data Sheets

- Husky Data Sheet
- Zed2i Data Sheet
- MP1-11TGS (Main PC) Data Sheet
- Telefonica RUT240 (WiFi Router) Data Sheet
- Fixposition Data Sheet
- Ouster Data Sheets
- Nvidia Jetson Orin 2GB
- RealSense Data Sheet
- Flir Data Sheet
- Ufactory xarm6 Data Sheet
- Robotiq Force Torque sensor Data Sheet
- Robotiq gripper Data Sheet

Documentations

- ZED2 ROS Docs
- ZED2 ROS2 Docs
- Husky Noetic ROS Docs
- Husky Foxy ROS2 Docs
- Ouster ROS Docs
- Ouster ROS2 Docs
- Nvidia Jetson Orin SDK Resources
- Fixposition ROS & ROS2 Docs
- RealSense SDK Docs
- Ufactory xarm6 Documentation
- Ufactory xarm6 ROS Docs
- Ufactory xarm6 ROS2 Docs
- Robotiq Force Torque sensor Docs
- Robotiq gripper Docs

Connection Scheme



_images/hxg_schematics.webp

Auxiliary Sensor Startup

On husky startup the following drivers are started:

- Husky base driver
- Phidgets IMU
- ZED2i(+Nvidia board)

The Ouster, ZED2i, Flir, Fixposition, xArm6, robotiq gripper & Force Torque sensor is powered on as well but the driver has to be manually started (can be automated as explained in Custom Startup Job) to ensure that the husky base driver starts up quickly and regardless of whether other components are attached or not.

ROS (Noetic)

The following commands are executed to activate the drivers for all the sensors when using ROS:

```
Robotiq 2F-240:  roslaunch hxg_robotiq robotiq_2f_140.launch
Robotiq Ft-300:  roslaunch hxg_robotiq robotiq_ft_300.launch
Flir:            roslaunch hxg_flir system.launch
Ouster:          roslaunch hxg_lidars lidars.launch
Realsense:       roslaunch hxg_realsense_cameras wrist_camera.launch
Fixposition:     roslaunch hxg_fixposition system.launch
```

ROS2 (Foxy)

The following commands are executed to activate the drivers for all the sensors when using ROS2:

```
Robotiq 2F-240:  ros2 run hxg_robotiq_2f_140 system_launch
Robotiq Ft-300:  ros2 launch hxg_robotiq_ft_300 system.launch.py
Flir:            ros2 launch hxg_flir system.launch.py
Ouster:          ros2 launch hxg_lidars system.launch.py
Realsense:       ros2 launch hxg_realsense system.launch.py
Fixposition:     ros2 launch hxg_fixposition system.launch.py
```

Network-Setup

The Husky FH has Multiple networks and vary slightly, the following table shows the networks.

Device	Network Address	Users	Password
Husky	192.168.131.1	administrator	mybotshop
Nvidia	192.168.131.3	administrator	mybotshop
Flir	192.168.131.30	admin	KirZ0G
Ouster	192.168.131.20	N/A	N/A
xArm6	192.168.131.10	N/A	N/A
xArm6 Web	192.168.131.10:18333	N/A	N/A
Fixposition	192.168.131.40	N/A	N/A
Router	192.168.131.100	N/A	Mybotshop23

Operation

To use the drivers, the following steps must be followed:

1. Connect to the Husky WiFi.
2. ssh into the husky via:

```
ssh -X administrator@192.168.131.1
```

```
mybotshop
```

The following message will show up with quick instructions for startup.

```
-----------------------------------------------------------------------

 _  _  _  _  ____   __  ____  ____  _  _   __  ____
( \/ )( \/ )(  _ \ /  \(_  _)/ ___)/ )( \ /  \(  _ \
/ \/ \ )  /  ) _ ((  O ) )(  \___ \) __ ((  O )) __/
\_)(_/(__/  (____/ \__/ (__) (____/\_)(_/ \__/(__)
-----------------------------------------------------------------------
Current PC: Husky
-----------------------------------------------------------------------
Husky IP:     192.168.131.1
Pswd:         mybotshop
SSH:          ssh -X administrator@192.168.131.1
Nvidia IP:    192.168.131.3
Pswd:         mybotshop
SSH:          ssh -X administrator@192.168.131.3
Router IP:    192.168.131.100  (INFO: Default off)
Router ID:    admin
Pswd:         Mybotshop23
xARM6 IP:     192.168.131.10  (INFO: Default off)
Web IP:       192.168.131.10:18333
Ouster IP:    192.168.131.20  (INFO: Default off)
Flir IP:      192.168.131.30  (INFO: Default off)
Web ID:       admin
Pswd:         KirZ0G
Fixpos IP:    192.168.131.40  (INFO: Default off)
--------------------------------------------------------------------------
HXG Noetic (Default)
--------------------------------------------------------------------------
Activate:        hxg_noetic
HXG status:      sudo service hxg status
HXG start:       sudo service hxg start
HXG stop:        sudo service hxg stop
HXG restart:     sudo service hxg restart
Update startup:  rosrun hxg_startup startup_installer.py
Startup:         roslaunch hxg_startup system_bringup.launch
                (INFO: Default on, Runs  Husky Driver
                                        Phidgets Driver)
Moveit:          roslaunch hxg_manipulation system_moveit.launch
                (INFO: Default off, Runs xARM6 Driver
                                        Robotiq-2F-140 Driver
                                        Robotiq-FT-300 Driver)
Robotiq 2F-240:  roslaunch hxg_robotiq robotiq_2f_140.launch
                (INFO: Default off)
Robotiq Ft-300:  roslaunch hxg_robotiq robotiq_ft_300.launch
Flir:            roslaunch hxg_flir system.launch
                (INFO: Default off)
Ouster:          roslaunch hxg_lidars lidars.launch
                (INFO: Default off)
Realsense:       roslaunch hxg_realsense_cameras wrist_camera.launch
                (INFO: Default off)
Fixposition:     roslaunch hxg_fixposition system.launch
```

```
                (INFO: Default off, Requires RTK Connection
                                    Requires Internet Connection)
Zed2i:           (INFO: Default off, ROS-Multimachine from Nvidia)
Visualization:   roslaunch hxg_viz view_robot.launch
---------------------------------------------------------------------------
HXG Foxy
---------------------------------------------------------------------------
Activate:        hxg_foxy
System Bringup:  ros2 launch hxg_bringup system.launch.py
                (Runs Husky Driver
                    Phidgets Driver
                    xARM6 Hardware Driver
                    xARM6 Joinstate Driver)
Moveit (HXG):    ros2 launch hxg_moveit system.launch.py
Moveit (xARM6):  ros2 launch hxg_manipulation xarm6_moveit.launch.py
Robotiq 2F-240:  ros2 run hxg_robotiq_2f_140 system_launch
Robotiq Ft-300:  ros2 launch hxg_robotiq_ft_300 system.launch.py
Flir:            ros2 launch hxg_flir system.launch.py
Ouster:          ros2 launch hxg_lidars system.launch.py
Realsense:       ros2 launch hxg_realsense system.launch.py
Fixposition:     ros2 launch hxg_fixposition system.launch.py
                (INFO: Default off, Requires RTK Connection
                                    Requires Internet Connection)
ZED2i:           (INFO: Default off, ROS-Multimachine from Nvidia)
Visualization:   ros2 launch hxg_viz view_robot.launch.py
---------------------------------------------------------------------------
```

Custom Startup Job

The launch file system_bringup.launch in the catkin package of hxg_startup includes everything which is being launched at the startup. Startup job and its content can be verified in the directory /etc/ros/noetic ; here the original launch file will be copied. If you are using ROS2 then in that case its content can be verified in the directory /etc/ros/foxy .

> **Warning:** Do not create Multiple startup jobs with Multiple launch files this will break the system. Only one launch file should be created and everything else should be included in this.

If the systems breaks at startup you can start debugging by:

For ROS-Noetic

```
roslaunch hxg_startup system_bringup.launch
```

For ROS2-Foxy

```
ros2 launch hxg_startup system_bringup.launch.py
```

This will show which part of the launch file is having errors or you can start running everything in the launch file one by one in separate terminals.

Adding ROS-Nodes & Launch Files to Robot Startup

1. Navigate and open hxg_startup/launch/system_bringup.launch .
2. Add your custom node or launch file.
3. Run the following script:

```
rosrun hxg_startup startup_installer.py
```

1. Verify if it is working correctly via:

```
sudo service hxg status
```

Important: Once you add your custom node or launch files in the system_bringup.launch , you will need to run rosrun hxg_startup startup_installer.py so that it updates the start-up job on the Husky's PC otherwise it will not be updated.

Note: Similar steps are available for ROS Foxy.

Ouster

_images/ouster.webp

To manually launch the ouster driver, the following command can be used when using ROS(Noetic):

```
roslaunch hxg_lidars lidars.launch
```

To manually launch the ouster driver, the following command can be used when using ROS2(Foxy):

```
ros2 launch hxg_lidars system.launch.py
```

Robotiq 2F-240

_images/robotiq-2f-140mm.webp

To manually launch the robotiq gripper driver, the following command can be used when using ROS(Noetic):

```
roslaunch hxg_robotiq robotiq_2f_140.launch
```

To manually launch the robotiq gripper driver, the following command can be used when using ROS2(Foxy):

```
ros2 run hxg_robotiq_2f_140 system_launch.py
```

Robotiq Ft-300

_images/robotiq_ft_300.webp

To manually launch the Robotiq Forcetorque driver, the following command can be used when using ROS(Noetic):

```
roslaunch hxg_robotiq robotiq_ft_300.launch
```

For force torque reset to zero ROS(Noetic).

```
rosservice call /robotiq_ft_sensor_acc "command_id: 8 command: ''"
```

To manually launch the Robotiq Forcetorque driver, the following command can be used when using ROS2(Foxy):

```
ros2 launch hxg_robotiq_ft_300 system.launch.py
```

For force torque reset to zero ROS2(Foxy).

```
ros2 service call /robotiq_ft_sensor_acc robotiq_ft_sensor/srv/Sensoraccessor command_id:\
8
```

Flir

_images/flir.webp

To manually launch the Flir driver, the following command can be used when using ROS(Noetic):

```
roslaunch hxg_flir system.launch
```

To manually launch the Flir driver, the following command can be used when using ROS2(Foxy):

```
ros2 launch hxg_flir system.launch.py
```

Realsense D405

_images/intel_rsd405.webp

To manually launch the Realsense driver, the following command can be used when using ROS(Noetic):

```
roslaunch hxg_realsense_cameras wrist_camera.launch
```

To manually launch the Realsense driver, the following command can be used when using ROS2(Foxy):

```
ros2 launch hxg_realsense system.launch.py
```

Fixposition

_images/fixposition.webp

To manually launch the Fixposition driver, the following command can be used when using ROS(Noetic):

```
roslaunch hxg_fixposition system.launch
```

To manually launch the Fixposition driver, the following command can be used when using ROS2(Foxy):

```
ros2 launch hxg_fixposition system.launch.py
```

ZED2i

_images/zed2.webp

On the robot start-up the ZED2i camera automatically starts and all its information is published to Husky's main PC. The information can be verified via:

```
rostopic list
```

when using ROS2 you can verify the information via:

```
ros2topic list
```

It may take upto a minute for all the drivers of the Nvidia to start. The main purpose of the Nvidia is to enable the CUDA for ZED2. The stream for the ZED2 is transferred over to the main Husky PC via ROS Multi machine. An example stream can be viewed rostopic echo /zed2/zed2_node/rgb/image_rect_color .

Important: The ZED2i requires USB 3.0 port and in the installed Nvidia board there is one available USB 3.0 port. So incase of removal please plug in the ZED2i into the same initial port it was in.

Moveit

If you want to run moveit:

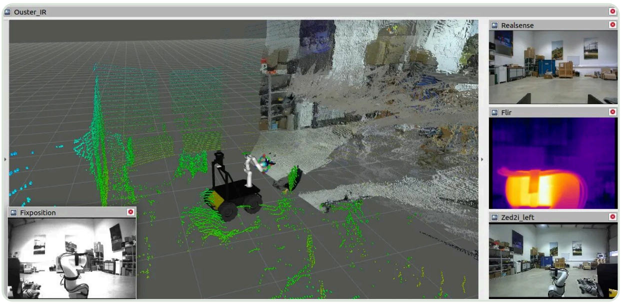
For ROS-Noetic:

```
roslaunch hxg_manipulation system_moveit.launch
```

For ROS2-Foxy:

```
ros2 launch hxg_moveit system.launch.py
```

Visualization



_images/hxg_viz2.webp

To visualize the robot in Rviz when using ROS-Noetic please run the following command:

```
roslaunch hxg_viz view_robot.launch
```

To visualize the robot in Rviz when using ROS2-Foxy please run the following command:

```
ros2 launch hxg_viz view_robot.launch.py
```

Nvidia AGX Orin

To verify the status of the Zed2i service and its connection with the main PC, please adhere to the following steps:

1. Connect to the Husky WiFi.
2. ssh into the Nvidia via:

```
ssh -X administrator@192.168.131.3
mybotshop
```

The following message will show up with quick instructions for startup.

```
-----------------------------------------------------------------------

     _  _  _  _  ____   __  ____  ____  _  _   __  ____
    ( \/ )( \/ )(  _ \ /  \(_  _)/ ___)/ )( \ /  \(  _ \
    / \/ \ )  /  ) _ ((  O ) )(  \___ \) __ ((  O )) __/
    \_)(_/(__/  (____/ \__/ (__) (____/\_)(_/ \__/(__)
-----------------------------------------------------------------------
Current PC: Nvidia
-----------------------------------------------------------------------
Husky IP:     192.168.131.1
Pswd:         mybotshop
SSH:          ssh -X administrator@192.168.131.1
Nvidia IP:    192.168.131.3
Pswd:         mybotshop
SSH:          ssh -X administrator@192.168.131.3
--------------------------------------------------------------------------
Nvidia Noetic (Default)
--------------------------------------------------------------------------
Activate:           hxg_noetic
Nvidia status:      sudo service nv_ros status
Nvidia start:       sudo service nv_ros start
Nvidia stop:        sudo service nv_ros stop
Nvidia restart:     sudo service nv_ros restart
Update startup:     rosrun nv_startup startup_installer.py
ZED2i:              roslaunch nv_startup startup.launch
                    (INFO: Default on)
--------------------------------------------------------------------------
Nvidia Foxy
--------------------------------------------------------------------------
Activate:        hxg_foxy
ZED2i:           ros2 launch nv_startup driver.launch.py
                (INFO: Default off)
Build package:   colcon build --symlink-install
                            --allow-overriding image_transport
                            --cmake-args=-DCMAKE_BUILD_TYPE=Release
--------------------------------------------------------------------------
```

Custom Startup Job Nvidia

The launch file startup.launch in the catkin package of nv_startup includes everything which is being launched at the startup. Startup job and its content can be verified in the directory /etc/ros/noetic ; here the original launch file will be copied. If you are using ROS2 then in that case its content can be verified in the directory /etc/ros/foxy .

> **Warning:** Do not create Multiple startup jobs with Multiple launch files this will break the system. Only one launch file should be created and everything else should be included in this.

If the systems breaks at startup you can start debugging by:

for ROS-Noetic .. code:: Bash

roslaunch nv_startup startup.launch

for ROS2-Foxy .. code:: Bash

ros2 launch nv_startup driver.launch.py

This will show which part of the launch file is having errors or you can start running everything in the launch file one by one in separate terminals.

Adding ROS-Nodes & Launch Files to Nvidia Startup

1. Navigate and open nv_startup/launch/startup_installer.launch .
2. Add your custom node or launch file.
3. Run the following script:

```
rosrun nv_startup startup_installer.py
```

1. Verify if it is working correctly via:

```
sudo service nv_ros status
```

> **Important:** Once you add your custom node or launch files in the startup.launch , you will need to run rosrun nv_startup startup_installer.py so that it updates the start-up job on the Husky's PC otherwise it will not be updated.

13 Field Deployment: Husky FRA (minimal – navigation only, no arm driver documented)

Note: The Robotic Manipulator and Autonomous Mobile Robot safety guideline sections that originally appeared at the end of this unit's documentation are identical across the TUM, BE and FRA units and have been consolidated into a single shared chapter — see Chapter 15, Robotic Manipulator & AMR Safety Guidelines. Note this unit ships without a dedicated manipulator driver section in its own source documentation.

MMP Husky FRA

_images/husky_fra.webp

Table of Contents

Getting Started

To start-up the Husky, the instructional video playlist can be used which details the opening and use of the Husky.

Platform Overview

The Husky Franka project is equipped with a state-of-the-art configuration tailored to meet its diverse operational requirements. At its core, it features the Clearpath Husky (IP44) platform, powered by a custom PC, ensuring efficient data processing and ample storage capabilities.

The robotic arm utilized in the system is the Franka Emika Research 3 , controlled via the Franka Emika DC Control Box, offering precise manipulation and versatility in performing various tasks. Additionally, the system integrates the Franka Emika 2-Finger-Gripper, which provides dexterous and reliable object handling.

This advanced setup ensures robust performance across a wide range of tasks, making the Husky Franka a versatile and powerful platform for modern robotic applications.

Data Sheets

- Husky Data Sheet
- Franka Emika Research 3 Data Sheet

Documentations

- Husky Foxy ROS2 Docs
- Franka Emika Quick-Start
- Franka Emika ROS Docs
- Franka Emika Control Interface Docs

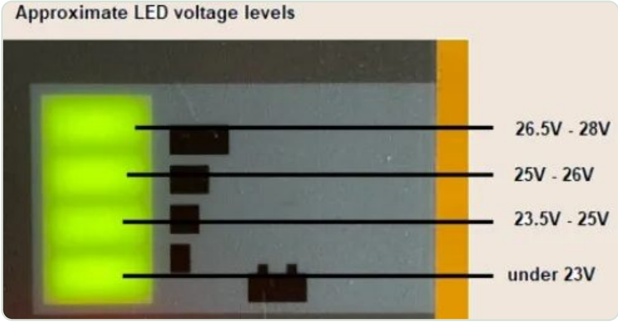
Quick-Start

Powering on the Husky

Power Button	Emergency Switch Release	Charging Port

- Ensure the emergency button is released as well as the key is turned to the unlock position. ◦ To check if the emergency switch is released, twist the switch in the indicated direction on it. If it pops up, then the switch has successfully been released. ◦ If the robot light is yellow , it means that either the emergency switch is not released or the key is not turned.
- Press the power button on the front of the Husky. It has the power symbol as displayed.

Charging the Husky



_images/charging_indicator.webp

LED Voltage Indicators

The power port can be used to charge the Husky. The indicator shows the voltage levels of the Husky and does not necessarily indicate the remaining power available.

ROS2 Navigation

Navi2 — Odom Navigation

Navigate in the global frame of map or odom

```
ros2 launch hsky_nav2 odom_navi.launch.py
```

Navi2 — Mapping

1. Mapping can begin using the joystick controller :

```
ros2 launch hsky_nav2 slam.launch.py
```

1. Once you are satisfied with your map you can export it by running the following command in one of the Huskys terminals:

```
ros2 run nav2_map_server map_saver_cli -f
/home/administrator/ros2_ws/src/mybotshop/hsky_nav2/maps/custom_map
```

1. Rebuild the husky nav2 so that the maps can be installed in the install folder:

```
colcon build --symlink-install --cmake-args -DCMAKE_BUILD_TYPE=Release
```

Note: The rebuilding step is not required if the filename is not changed.

Navi2 — Map Navigation

```
ros2 launch hsky_nav2 map_navi.launch.py
```

Navigate in the global frame of map . After launching the drivers and commands, providing a 2D pose estimate in RViz in the Map frame initiates map navigation

Caution

- Please read the instruction manuals pertaining to Clearpath Husky and Ufactory xAr6 before operating the robot.
- Always be vigilant and remain close to the emergency stop buttons when the robot is in operation.

Known Issues

1. Husky moving forward on startup This may occur due to time synchronization of the internet causing the robot's joint states to reverse. This is dependent on how the network is synchronized between the Husky's computer and the internet. By default we have disabled synchronization of time from the internet to prevent this issue, until Clearpath updates its drivers.

Frequently Asked Questions

1. Logitech controller not working Ensure that it is powered on. Ensure that the input mode is X, which is set on the backside of the controller.
2. Husky in standby mode (yellow color) Ensure that the emergency switch is twisted and released. Ensure that the key is in the unlock position.
3. xArm6 does not move with Moveit2 . The solution to this to simply restart the driver.
4. Unable to see some of the ros2 topic lists in remote PC. Ensure you are in the same network. Ensure that your ROS DOMAIN ID is 0. Ensure that your ROS LOCAL HOST is set to 0. Ensure that your network system (+router) allows for ROS2 communication and/or is configured. Restart the robot services: ◦ sudo service clearpath-robot restart ◦ sudo service hsky restart

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:

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.

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.

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.

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.

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.

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.

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.

14 Field Deployment: Husky TUHH (ROS1 Noetic, xArm6 + Robotiq, power-relay board)

MMP Husky TUHH

_images/husky_side_view3.webp

Getting Started

To start-up the Husky, the instructional video playlist can be used which details the opening and use of the Husky.

Data Sheets

- Husky Data Sheet
- Zed2i Data Sheet
- MP1-11TGS (Main PC) Data Sheet
- Telefonica RUT240 (WiFi Router) Data Sheet
- Fixposition Data Sheet
- Ouster Data Sheets
- Nvidia Jetson Orin 2GB
- RealSense Data Sheet
- Flir Data Sheet
- Ufactory xarm6 Data Sheet
- Robotiq Force Torque sensor Data Sheet
- Robotiq gripper Data Sheet

Documentations

- ZED2 ROS Docs

- ZED2 ROS2 Docs
- Husky Noetic ROS Docs
- Husky Foxy ROS2 Docs
- Ouster ROS Docs
- Ouster ROS2 Docs
- Nvidia Jetson Orin SDK Resources
- Fixposition ROS & ROS2 Docs
- RealSense SDK Docs
- Ufactory xarm6 Documentation
- Ufactory xarm6 ROS Docs
- Ufactory xarm6 ROS2 Docs
- Robotiq Force Torque sensor Docs
- Robotiq gripper Docs

Pre-requisite Software

_images/ros_noetic.webp

The ROS driver for the robots is initially supported for the ROS Noetic . All of ROS related software should be installed in a remote Husky-PC as well as the Remote-PC . The steps for shown below are for Remote-PC as the Husky is pre-installed with the required drivers and ROS distribution.

PC Setup

ROS Noetic installation requires Ubutnu 20.04.

1. Download Ubuntu 20.04
2. Follow the Ubuntu 20.04 Installation guide

ROS-noetic Installation

The ros driver provided can be run either on a remote PC or on board robot computer. Usually, the on-board computer already has pre-installed ROS distribution, so the instructions below will be applicable to a remote computer. You may run the following commands to install ROS noetic or you can simply follow the instructions from the roswiki.

1. Enter the Ubuntu terminal and typie in the following command:

```
cd
```

1. The next step is setting up the source list:

```
sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $(lsb_release -sc) main" >
/etc/apt/sources.list.d/ros-latest.list'
```

1. After which you will set up the keys:

```
sudo apt install curl
curl -s https://raw.githubusercontent.com/ros/rosdistro/master/ros.asc | sudo apt-key add -
```

1. Then, you will install ROS-noetic:

```
sudo apt update
sudo apt install ros-noetic-desktop-full
```

1. Finally, you will add ROS environment to your bash file, what this means is that each time you load up your terminal, it will automatically find the built-in ROS packages.

```
echo "source /opt/ros/noetic/setup.bash" >> ~/.bashrc
source ~/.bashrc
```

1. Install python dependencies and initialize repo:

```
sudo apt install python3-rosdep python3-rosinstall python3-rosinstall-generator python3-
wstool build-essential
sudo rosdep init
rosdep update
```

Once the ROS packages are set-up the next step would be to set up a workspace in which you can add pre-built packages to control your robot.

Catkin Workspace

1. The first step is to ensure that you have correctly setup you ROS noetic. It can be checked by:

```
cd ..
source /opt/ros/noetic/setup.bash
```

1. Next, install catkin_tools:

```
sudo apt install python3-catkin-tools
```

1. The next step is to create, your workspace. This is typically created in the home directory following the standard ROS convention, once built, you should see a build and devel folder next to your src folder:

```
mkdir -p ~/catkin_ws/src
cd ~/catkin_ws/
catkin build
```

1. To enable the workspace, you must type in:

```
source devel/setup.bash
```

1. To avoid typing this everytime you start a terminal, you'll add this into the last line of the bashrc file:

```
gedit ~/.bashrc
```

- Add the following line at the end of the bashrc file:

```
source /home/<your_computer_username>/catkin_ws/devel/setup.bash
```

- Everytime you do a catkin build , it is necessary to either restart all your terminals, or source them.
- To verify that everything is working correctly, type the following into the terminal:

```
echo $ROS_PACKAGE_PATH
```

Installation

The package is designed to work on the Husky. To install the package simply download it from the MBS Husky-xARM6-Robotiq github repo.

- Give root permission to the installation script via

```
sudo chmod +x Installation_script.bash
```

- Run the script from the terminal via

```
./Installation_script.bash
```

- Provide the workspace name as catkin_ws
- Provide the directory as /home/administrator .

> **Note:** In case the username is not administrator then manual changes will have to be made in the startup job.

Network Setup

To set up a connection to the Husky, please follow the provided guidelines:

1. Connect an Ethernet cable from your Host PC to the Ethernet port of the Husky.
2. Next in your Host PC, create a static connection from your network manager with an ip of 192.168.131.51 . For the netmask, use 255.255.255.0 .
3. Once the static connection is setup correctly, one can ssh into the on-board PC as described below.

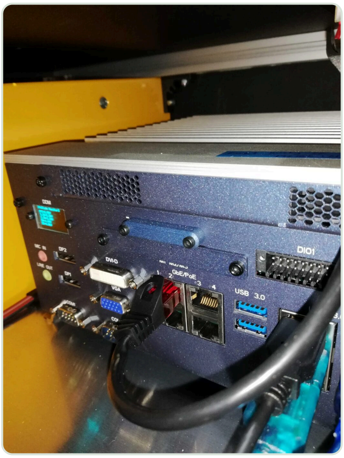
To ssh, enter the command terminal and enter the following:

```
ssh -X administrator@192.168.131.1
```

The password is:

```
clearpath
```

> **Attention:** The first port of the Husky is configured with the static ip 192.168.131.1 , others have been left dynamic for internet access.



Husky custom pc ethernet connection

Network tables

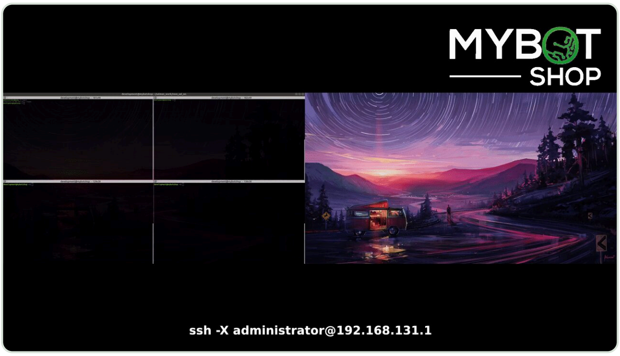
Device	Network Address
Husky	192.168.131.1
xARM6	192.168.132.1

Start-up

_images/husky_side_view3.webp

When the Husky is powered-on, the mbs_husky robot upstart job starts the ros hardware drivers for the Husky.

Startup Example



_images/husky_viz.gif

A 40 second video illustrating the activation of the visualization module, xARM6 hardware driver, and the Robotiq gripper driver.

Powering xARM6

- ssh into the husky.

```
ssh -X administrator@192.168.131.1
clearpath
```

- To power-on the xARM6

```
rostopic pub /pwr_relay std_msgs/Bool "data: true" --once
```

- To power-off the xARM6

```
rostopic pub /pwr_relay std_msgs/Bool "data: false" --once
```

Important: It takes approximately 60 seconds for the arm to power up. On start-up a beep sound can be heard. After 60 seconds two consecutive beep sounds can be heard indicating that the xARM6 has successfully started and can be accessed.

Hardware drivers xARM6

- To launch the xARM6 hardware driver, run the command:

```
roslaunch mbs_xarm_robotiq xarm_hardware_driver.launch
```

Hardware drivers Robotiq Gripper 2f-140

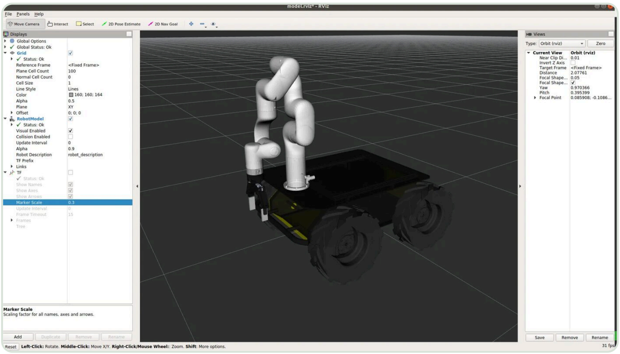
- To launch the Robotiq 2f-140 hardware driver, run the command:

```
roslaunch mbs_xarm_robotiq robotiq_hardware_driver.launch
```

Visualization

- The visualize the robot model.

```
roslaunch mbs_viz viz.launch
```



_images/husky_xarm_robotiq_viz1.webp

Husky xARM6 URDF

MBS Husky Startup

The launch file mbs_husky.launch includes everything which is being launched at the startup. This file can be updated to include all the nodes needed to run at the startup (e.g. your own packages). Startup job and its contents can be verified in the directory /etc/ros/noetic ; here the original launch file will be copied.

Important: Do not create multiple startup jobs with multiple launch files this will break the system. Only one launch file should be created and everything else should be included in this.

If the systems breaks at startup you can start debugging by:

```
roslaunch mbs_husky_startup mbs_husky.launch
```

This will show which part of the launch file is having errors or you can start running everything in the launch file one by one in separate terminals.

Note: Once you add your custom node or launch files in the mbs_husky.launch , you will need to run the following script so that it updates the start-up job on the Husky's PC; to run with the newly updated mbs_husky.launch .

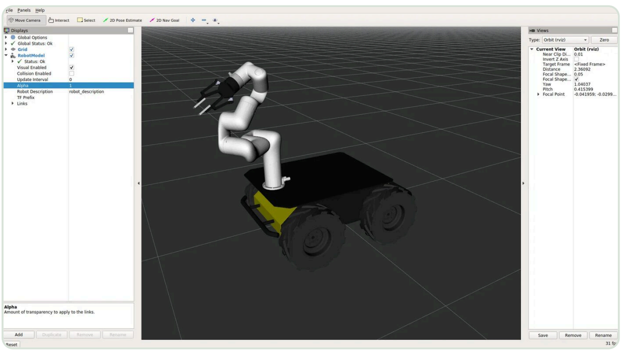
```
rosrun mbs_husky_startup startup_script.sh
```

MBS Viz

The mbs_viz package will be the primary viewing tool for the robot. It is launched via:

```
roslaunch mbs_viz viz.launch
```

Important: At start-up only the Husky and Phidgets IMU hardware drivers are running, whilst the Husky-xARM6-Robotiq and the Robotiq gripper are both powered-off, due to which only the Husky will be displayed. Once the xARM6 and the Robotiq hardware drivers are powered on then, they will show up in the rviz display.



_images/husky_xarm_robotiq_viz2.webp

RVIZ display for the Husky-xARM6-Robotiq

MBS Husky Description

The launch file description.launch includes a custom URDF that has the Husky, xARM6, Phidgets imu, and Robotiq 2f-140 gripper integrated into it. It simply uses the roswiki robot_state_publisher to publish the static joints of the robot. It is automatically launched via the startup job. To launch it manually, it can be done via:

```
roslaunch mbs_husky_description description.launch
```

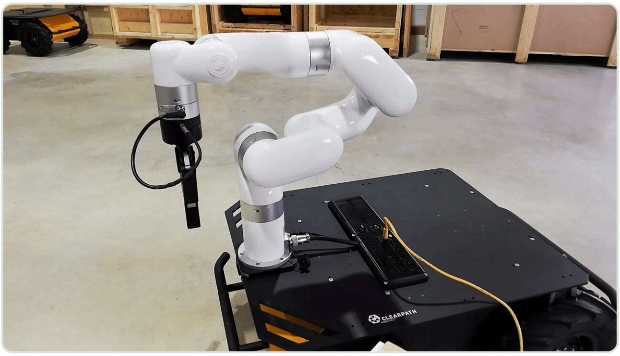
The bringup.launch launches the description launch file as well as the fake joint state publishers from roswiki joint_state_publisher_gui . It can be used for visualization when not connected to the robot or for debugging the URDF. It can be launched via:

```
roslaunch mbs_husky_description bringup.launch
```

MBS xARM Robotiq

The mbs_xarm_robotiq package is used to turn on the hardware drivers for the xARM6 and the Robotiq gripper.

xARM6



_images/xarm.gif

xARM6

Ensure that the xARM6 is powered-on by following the Start-up .

- To launch the xARM6 hardware driver, run the command:

```
roslaunch mbs_xarm_robotiq xarm_hardware_driver.launch
```

- To launch a stand-alone moveit planner for the xARM6:

```
roslaunch mbs_xarm_robotiq xarm_standalone_moveit.launch
```

> **Note:** Currently, for the xarm_standalone_moveit.launch the xARM6, is only valid. The gripper has not yet been configured to work with it in the moveit planner. For more information on how to use this, you may check the roswiki moveit_setup_assistant .

Robotiq Gripper 2f-140

_images/robotiq.gif

Robotiq Gripper 2f-140

Ensure that the xARM6 is powered-on by following the Start-up . * To launch the Robotiq 2f-140 hardware driver, run the command:

```
roslaunch mbs_xarm_robotiq robotiq_hardware_driver.launch
```

The hardware driver launches an action-client server as well as a joint-state publisher for the gripper. An example on how to communicate with the gripper is available in the mbs_xarm_robotiq/nodes/test_robotiq_gripper.py

Power-relay

Power-relay Status

When the robot starts, an Arduino starts.This Arduino controls a power relay via the main PC in the husky. A startup script starts the communication between the Arduino and the main PC in the husky. To check the status of this module, the following steps are to be performed:

1. Connecting to the robot via Ethernet as described:

```
ssh -X administrator@192.168.131.1
clearpath
```

1. Next, we can check the status of the power-relay node via:

```
sudo service mbs_husky status
```

> **Note:** If the icon is green, it means that the packages have started without any issues. In the event of an error, you may restart this startup job via:

```
sudo service mbs_husky restart
```

1. To power-on the relay, run the command:

```
rostopic pub /pwr_relay std_msgs/Bool "data: true"
```

Note: An audible click can be heard of the switching relay.One ROS message needs to go through after which you may cancel the publisher ctrl+c .

1. To power-off the relay, run the command:

```
rostopic pub /pwr_relay std_msgs/Bool "data: false"
```

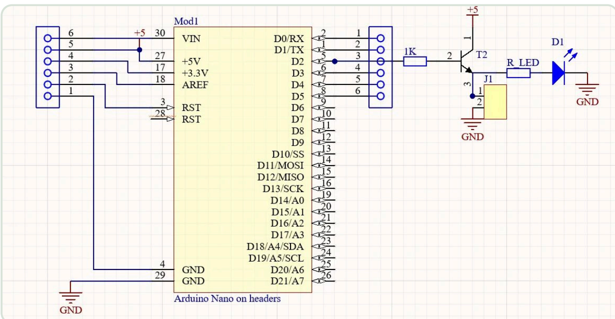
Attention: The rostopic that subscribes to this is the /pwr_relay and it accepts Boolean messages std_msgs/Bool .

Arduino Nano	Arduino Nano Connections

Power-relay Technical Specification

The Arduino is activating the 24V DC battery power on the XT60 connector. A 20amps fuse ensures that the cables and the installed electronics are not damaged. Furthermore, the voltage applied to the XT60 connector can vary depending on the battery capacity, since no voltage converter or something similar is integrated. The Arduino is connected to the /dev/ttyUSB0 port.

Warning: The Red/Black 2,5mm 2 cable inside the Husky, which is going to the 24V external power control box is always live as the battery power is continuously supplied.



_images/HUSKY_TUHH_24V_Power_Board_Schematic.webp

Power-board schematic

Phidget IMU

_images/phidgets_imu.webp

Phidgets IMU

Phidgets Start-up

When the robot starts, the Phidget's IMU starts via a startup script. To check the status of this module, the following steps are to be performed:

1. Connecting to the robot via Ethernet as described:

```
ssh -X administrator@192.168.131.1
clearpath
```

1. Next, we can check the status of the Phidget's IMU via:

```
sudo service mbs_husky status
```

> **Important:** If the icon is green, it means that the packages has started without any issues. In the event of an error, you may restart this startup job via:

```
sudo service mbs_husky restart
```

1. To check the linear accelerations and angular velocities:

```
rostopic echo /IMU/data_raw
```

> **Attention:** This topic does not include orientations. For more information on orientation please refer to the roswiki phidgets_imu .

1. To check magnetic orientation:

```
rostopic echo /IMU/mag
```

1. More information is available on roswiki phidgets_imu . To check the orientation on rviz, be sure to use use the IMU visualization plugin.

```
roslaunch mbs_husky_description description.launch
```

1. Calibration must be done as instructed in the wiki according to the location. The Phidgets is connected to the /dev/ttyUSB1 port.

Husky

_images/husky_nav.gif

> **Note:** As the MYBOTSHOP Husky-xARM6-Robotiq is a multi-robot system consisting of Clearpath's Husky, hence, the guide provided by Clearpath for Husky Navigation is applicable. For ease of view, the guide is available below as well.

Below are the example launch files for three different configurations for navigating Husky:

- Navigation in an odometric frame without a map, using only roswiki move_base .
- Generating a map using roswiki gmapping .
- Localization with a known map using roswiki amcl .

If you're working with a real Husky, it's suggested to connect via SSH and launch the roswiki husky_navigation launchfiles from on board the robot. You'll need to have bidirectional communication with the robot's roscore in order to launch roswiki rviz on your workstation.

> **Important:** It should be noted that without any auxiliary components such as a camera or lidar, the robot will not be able to perform obstacle avoidance or mapping.

Navigation Without a Map

In the odometry navigation demo Husky attempts to reach a given goal in the world within a user-specified tolerance. The 2D navigation, generated by move_base, takes in information from odometry, laser scanner, and a goal pose and outputs safe velocity commands. In this demo the configuration of move_base is set for navigation without a map in an odometric frame (that is, without reference to a map).

To get all Navigation related files for Husky, run:

```
sudo apt-get install ros-noetic-husky-navigation
```

To launch the navigation demo, run:

```
roslaunch husky_navigation odom_navigation_demo.launch
```

To visualize with the suggested rviz configuration launch:

```
roslaunch mbs_viz viz.launch
```

Note: Add the topic of /points to visualize the point clouds as well.

To send goals to the robot, select the 2D Nav Goal tool from the top toolbar, and then click anywhere in the rviz view to set the position. Alternatively, click and drag slightly to set the goal position and orientation.

If you wish to customize the parameters of move_base, local costmap, global costmap and base_local_planner, clone roswiki husky_navigation into your own workspace and modify the corresponding files in the params subfolder.

Making a Map

In this demonstration, Husky generates a map using gmapping. Begin by launch the gmapping launch file on the robot:

```
roslaunch husky_navigation gmapping_demo.launch
```

And on your workstation, launch rviz with the suggested configuration:

```
roslaunch husky_viz view_robot.launch config:=gmapping
```

You must slowly drive Husky around to build the map. As obstacles come into view of the laser scanner, they will be added to the map, which is shown in rviz. You can either drive manually using the interactive markers, or semi-autonomously by sending navigation goals.

When you're satisfied, you can save the produced map using map_saver :

```
rosrun map_server map_saver -f mymap
```

This will create a mymap.yaml and mymap.pgm file in your current directory.

Navigation With a Map

Using roswiki amcl , Husky is able to globally localize itself in a known map. AMCL takes in information from odometry, laser scanner and an existing map and estimates the robot's pose.

To start the AMCL demo:

```
roslaunch husky_navigation amcl_demo.launch [map_file:=/path/to/my/map.yaml]
```

If you don't specify map_file , it defaults to an included pre-made map of the default "Husky Race" environment which Husky's simulator spawns in. If you're using a real Husky in your own environment, you'll definitely want to override this with the map created using the gmapping demo.

Before navigating, you need to initialize the localization system by setting the pose of the robot in the map. This can be done using 2D Pose Estimate in rviz or by setting the amcl initial_pose parameters. To visualize with the suggested rviz configuration launch:

```
roslaunch husky_viz view_robot.launch config:=localization
```

When rviz appears, select the Set 2D Pose tool from the toolbar, and click on the map to indicate to the robot approximately where it is.

xARM6

xARM6-Guide

xARM6

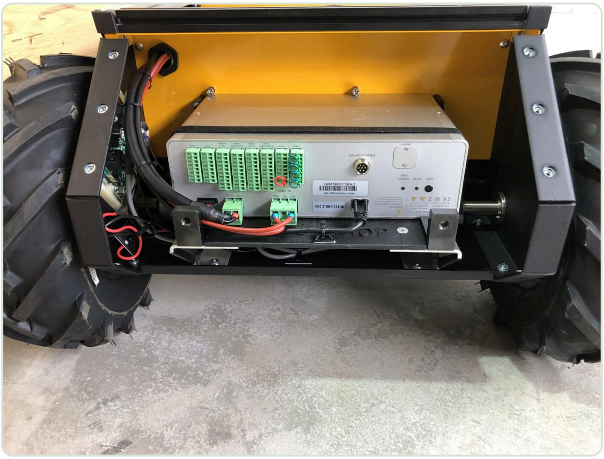
xARM6 integrated into Husky with a Robotiq 2f-140 gripper

> **Note:** As the Husky-xARM6-Robotiq is a multi-robot system consisting of xARM6 and robotiq gripper, hence, the xARM6 tutorials , xARM6 user manual , xARM6 developer manual and the Robotiq 2f-140 manual are applicable.

To power-on the xARM6 please follow the Start-up section.

xARM6 DC Control Box

The control box for the xARM6 is mounted in rear compartment of the Husky.



_images/DC_control_box.webp

xARM6-ROS

For the xARM6 ROS drivers please refer to the MBS xARM Robotiq section.

xARM6-Utilities

To operate the xARM6 from other means then ROS, the python and c++ sdks are available in the utils folder in the provided package. Moreover, direct control of the xARM6 is available via the app. To access the app, navigate into the /utils/app folder and run the command:

```
./xArmStudio-linux-0.6.0.AppImage
```

This should launch the xARM6. Connect to the IP address of 192.168.132.1 and you should have now complete access to the arm. Also the app has a built-in plugin to interface with robotiq gripper available in the control panels.

> **Important:** The app can also be accessed via ssh.

```
ssh -X administrator@192.168.131.1
clearpath
cd
cd catkin_ws/utils/app
./xArmStudio-linux-0.6.0.AppImage
```

Robotiq Gripper 2f-140

_images/robotiq.gif

Robotiq Gripper 2f-140

Attention: The robotiq gripper does all communication via the xARM6. Hence, the xARM6 must be powered on for the gripper to work.

Robotiq Gripper 2f-140 ROS

For the robotiq ROS drivers please refer to the MBS xARM Robotiq section.

15 Robotic Manipulator & AMR Safety Guidelines (TUM / BE / FRA units)

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:

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.

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.

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.

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.

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.

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.

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.

16 Frequently Asked Questions

Frequently Asked Questions

Husky moving forward on startup

1. Reason: This may occur due to time synchronization of the internet causing the robot's joint states to reverse. This is dependent on how the network is synchronized between the Husky's computer and the internet. By default we have disabled synchronization of time from the internet to prevent this issue, until Clearpath updates its drivers.

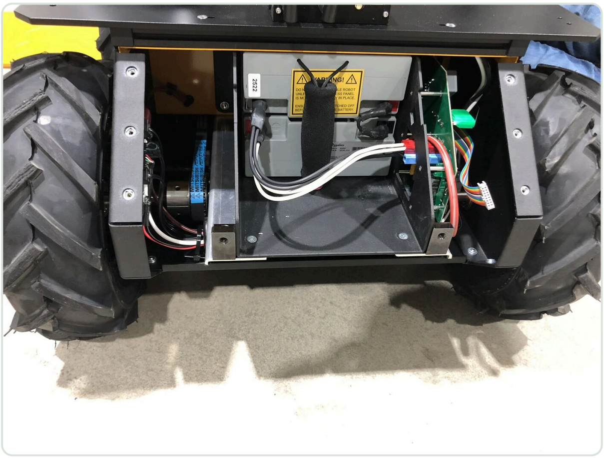
External Power Supply

The Husky cannot be moved when the external power supply is connected. It requires direct current power supply. The connection is the same as the one used for charging the Husky but only single connectors are required.

Important: 27 volts and 8 amps are required.

Noise from inside the Husky inner compartment

- Please remove the front cover of the Husky.



_images/IMG_7443.webp

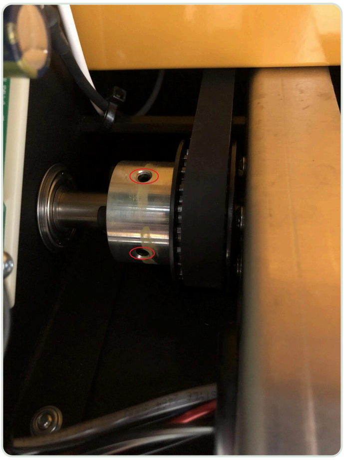
- Tighten the marked grub screws.

_images/IMG_7444.webp

- Please remove the back cover of the Husky.

_images/IMG_7441.webp

- Tighten the marked grub screws.



_images/IMG_7442.webp

- Additionally, you may have to tighten the Husky wheels grub screws.

_images/IMG_7445.webp

_images/IMG_7447.webp

General Precautions

1. Logitech controller not working Ensure that it is powered on. Ensure that the input mode is X, which is set on the backside of the controller.

1. Husky in standby mode (yellow color) Ensure that the emergency switch is twisted and released. Ensure that the key is in the unlock position.

1. UR5e does not move with Moveit2 . The solution to this to simply restart the driver.

1. UR5e driver not starting. This may happen if the emergency stop button is not connected to the UR5e control box. This causes the SD card in the control box to eject. The solution is to plug in back the emergency switch back and insert the SD card back into the UR5e. At times the ROS2 driver may not immediately detect the UR5e, hence, restarting the UR5e driver commonly resolves the issue.

1. UR5e driver not powering off. Restart the UR5e driver.

1. Unable to see some of the ros2 topic lists in remote PC. Ensure you are in the same network. Ensure that your ROS DOMAIN ID is 0. Ensure that your ROS LOCAL HOST is set to 0. Ensure that your network system (+router) allows for ROS2 communication and/or is configured. Restart the robot services: ◦ sudo service clearpath-robot restart ◦ sudo service hsky restart

2. Steamdeck right trackpad not moving the mouse. Click the 3 dots below the right trackpad. If 3 dots does not work, then hold the 3 bars for 3 seconds next to the Y button. Do this twice.

17 Getting Help & RMA

RMA / Support

How to Raise a Return Merchandise Authorization (RMA) Case

At MYBOTSHOP GmbH, 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 @ mybotshop . 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: 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.

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 .