Auto Navigating Rover

Group 8


Aline Cynthia Yiagnigni


Cristobal Gallardo

Emilio Antoine Altamirano


Gustavo Moura Scarenci


Keshav Kumar


Matheus Dias Cirillo

Smit Nareshbhai Kotadiya

Sumit Mor

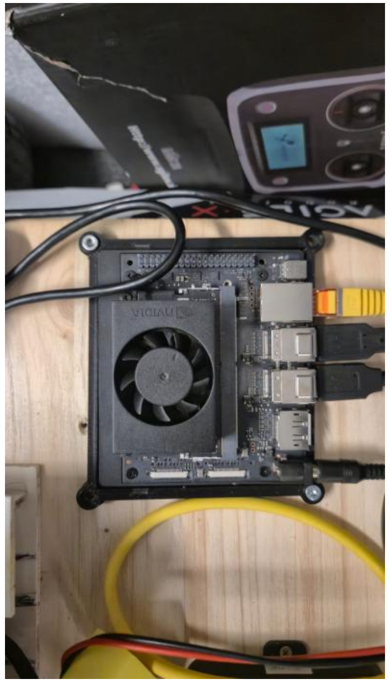
Goal

- Mapping a Physical Environment and Identifying Obstacles, in order to autonomously traverse the terrain safely
- Inspired from *Perseverance* Rover

The rover

AgileX Scout Mini


SCOUT MINI

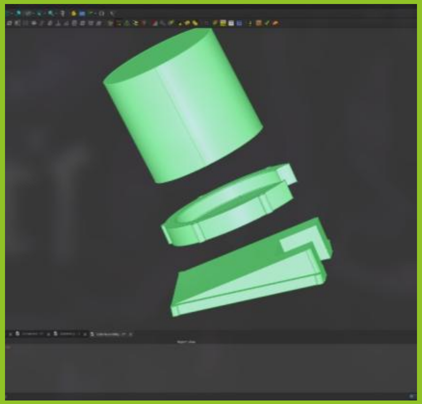



LeiShen Intelligent System
镭神智能
C16-151A-20110001

Hardware

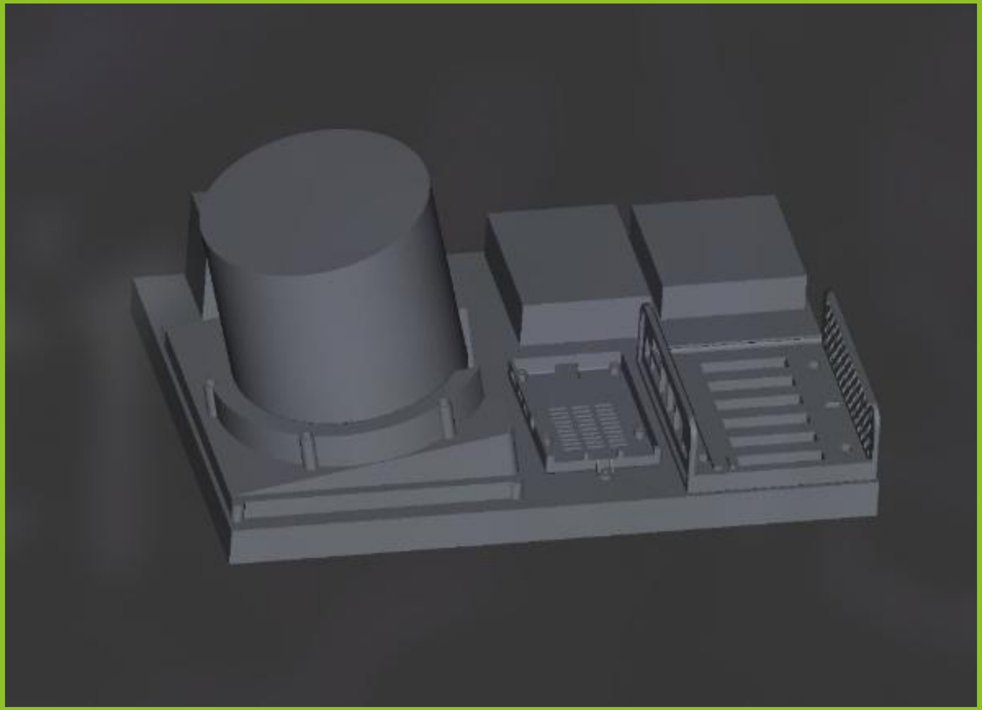
Mechanical Design & CAD Modeling

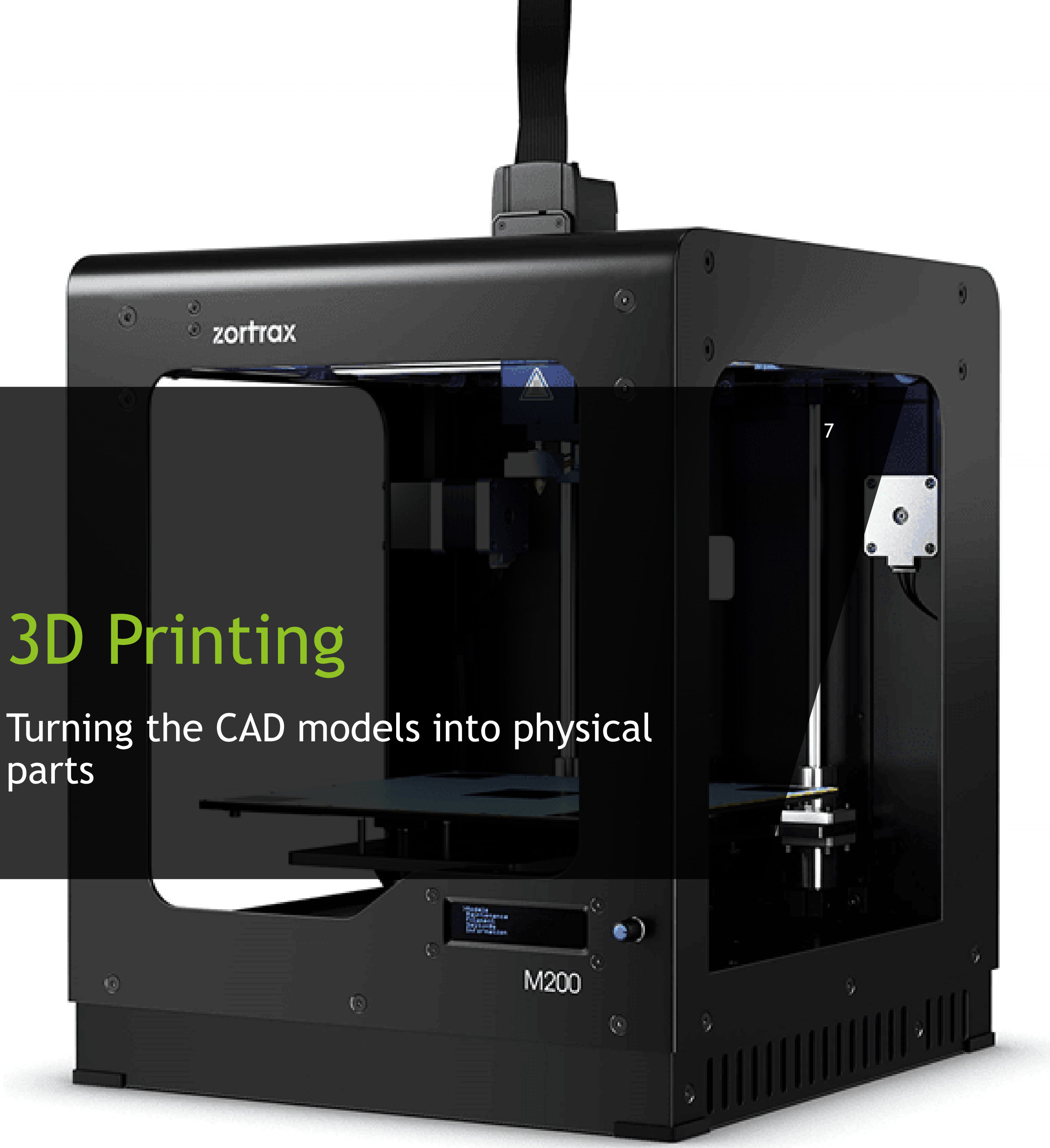
- The planning and digital creation phase.



LiDAR Mounting Assembly - Exploded View

Full Hardware Layout / Digital System Assembly

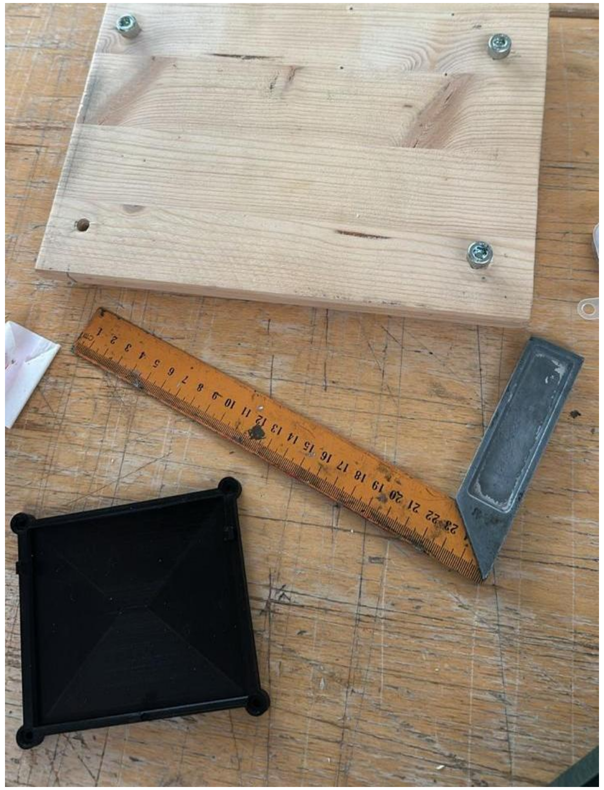


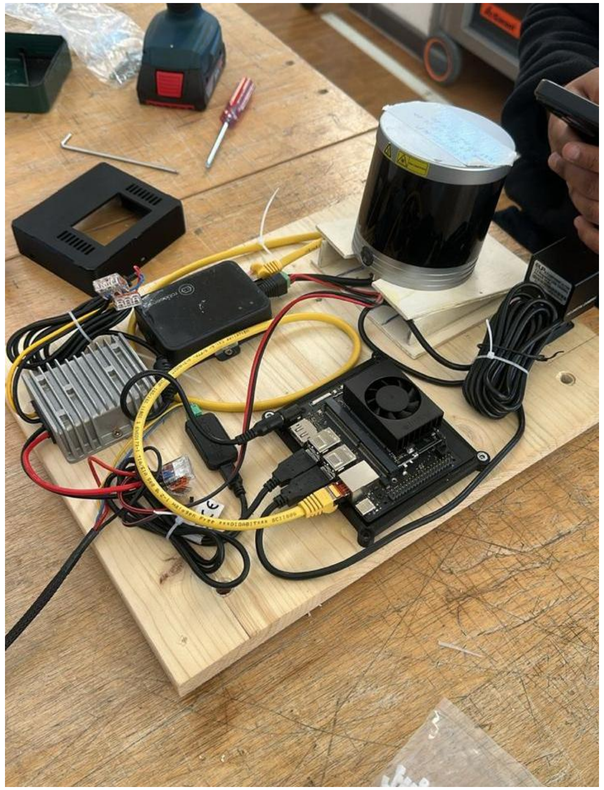


3D Printing

Turning the CAD models into physical parts

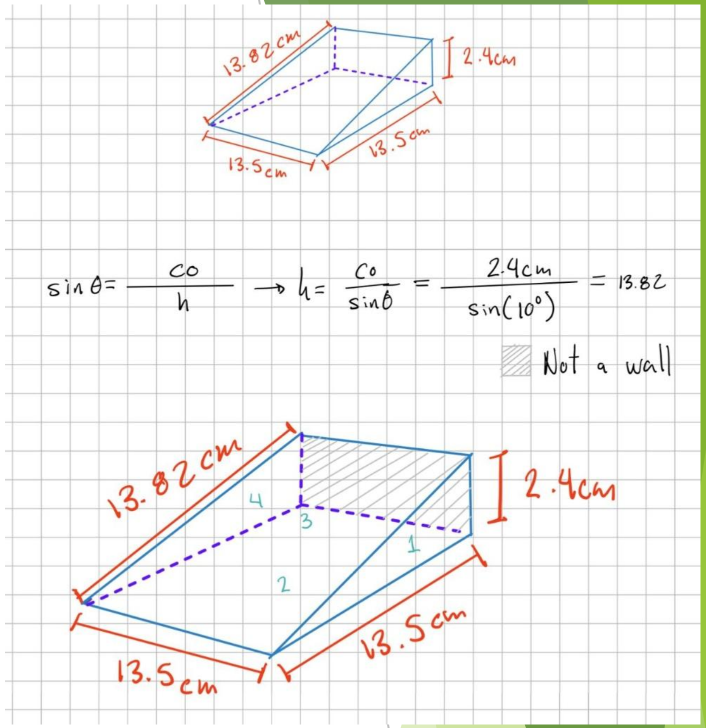
Hardware Mount Integration & Assembly

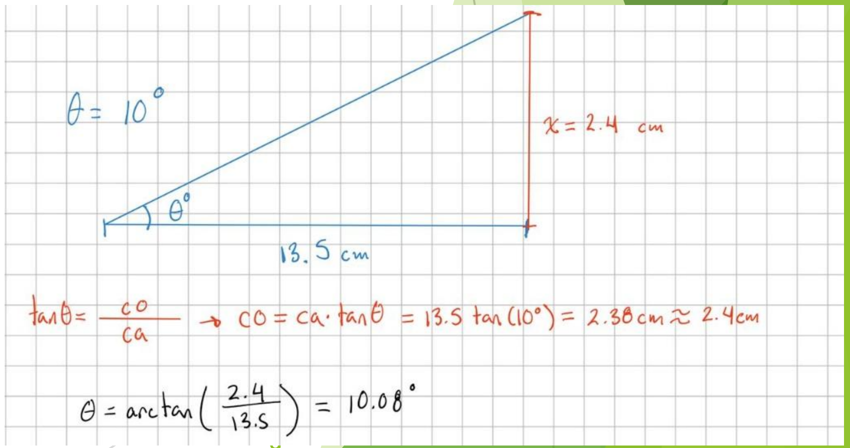




Geometric Design Parameters & Calculations

- **Sensor Angle Definition:** Defined a required inclination angle to optimize the sensor's scanning area.
- **Dimensional Analysis:** Calculated the vertical offset based on the available base length using trigonometric function
- **Slope Calculation:** Determined the hypotenuse length to ensure the 3D printed part fit the chassis perfectly.



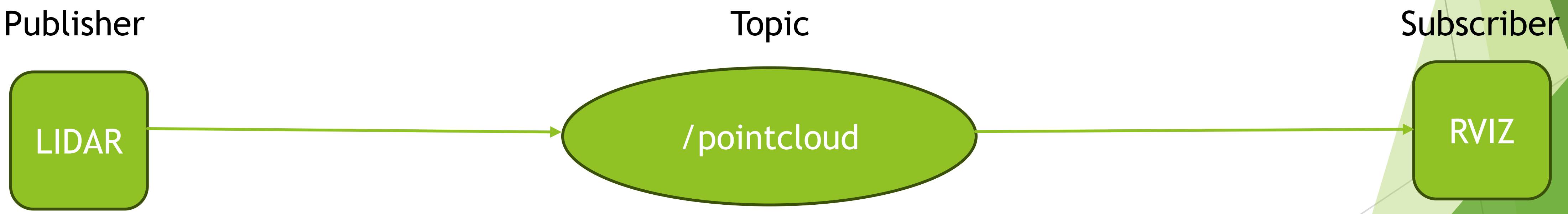


System Validation & Field Testing

- the integrated system works under real-world conditions.
  - Structural Integrity Check
  - Vibration Assessment

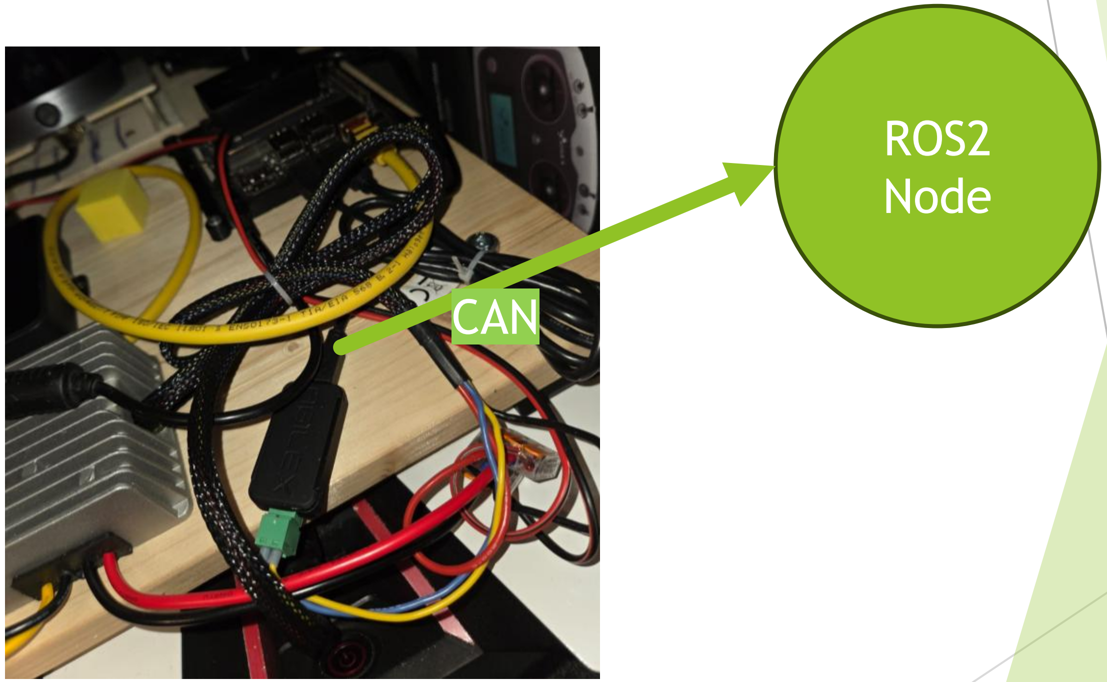
ROS

- Based Pub-Sub Model
- ROS2 Humble Hawkbills
- Why?: Due to multiple Pieces of Hardware

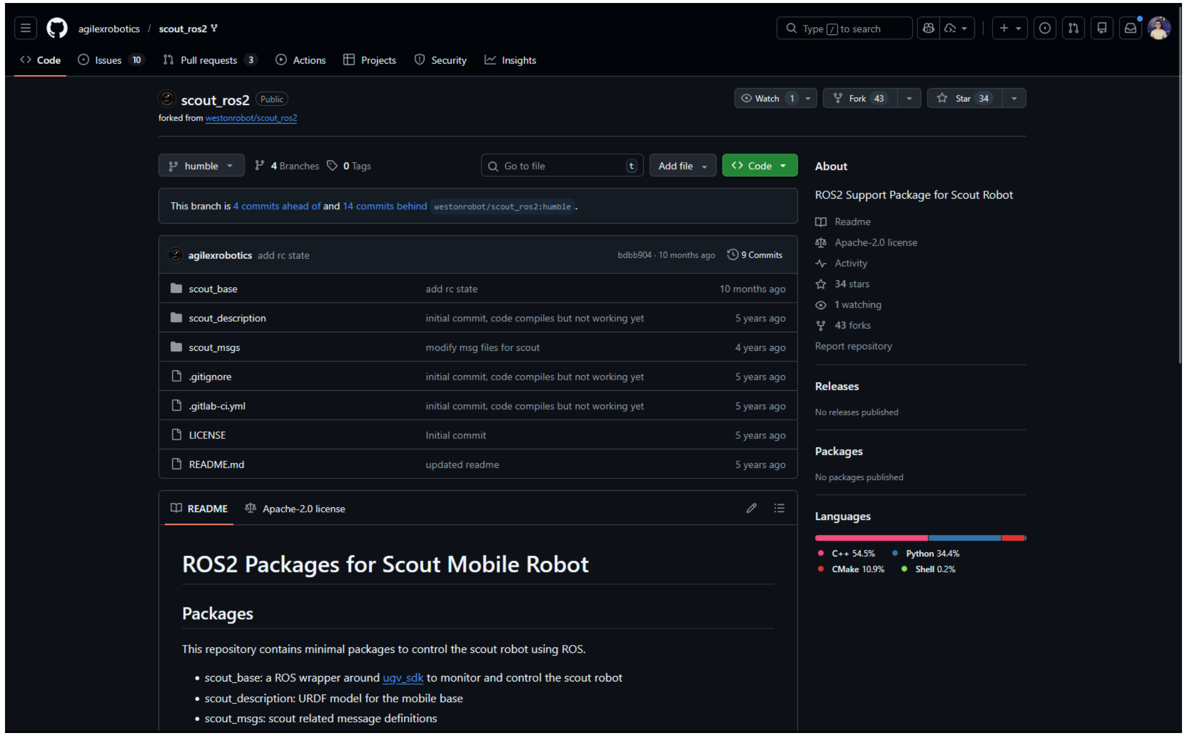


11

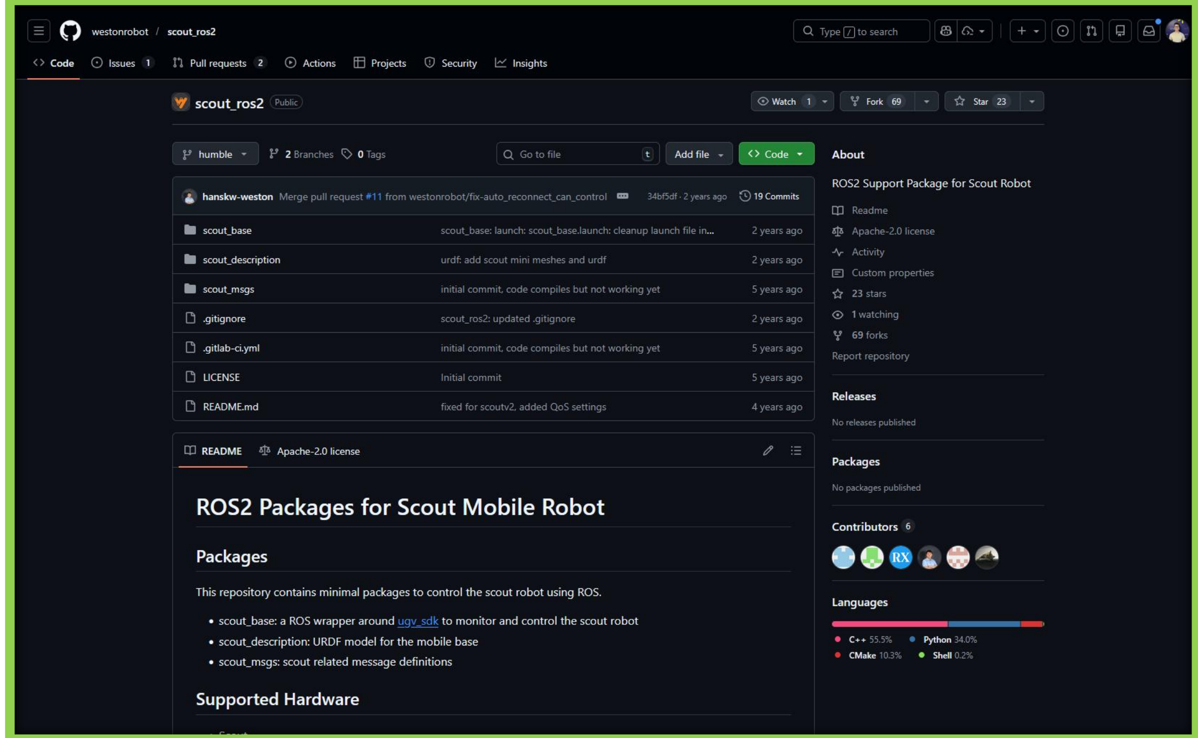
Rover Communication



ROS2 Node

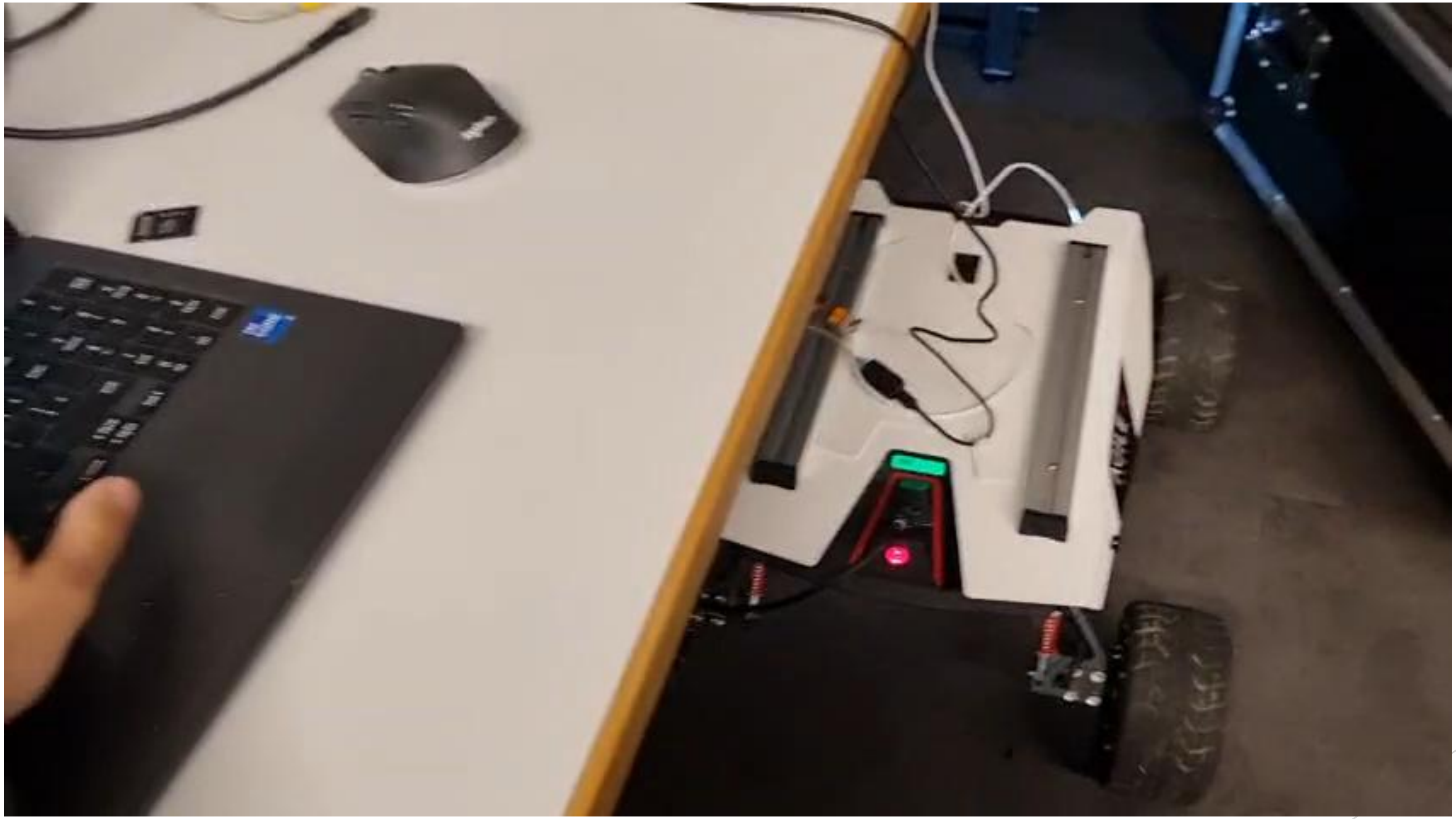


agilexrobotics / scout_ros2
Code
Issues 10
Pull requests 3
Actions
Projects
Security
Insights
scout_ros2 Public
forked from westonrobot/scout_ros2
Watch 1
Fork 43
Star 34
humble
4 Branches
0 Tags
Go to file
Add file
Code
This branch is 4 commits ahead of and 14 commits behind westonrobot/scout_ros2:humble .
agilexrobotics add rc state
bdbb904 · 10 months ago
9 Commits
scout_base add rc state 10 months ago
scout_description initial commit, code compiles but not working yet 5 years ago
scout_msgs modify msg files for scout 4 years ago
.gitignore initial commit, code compiles but not working yet 5 years ago
.gitlab-ci.yml initial commit, code compiles but not working yet 5 years ago
LICENSE Initial commit 5 years ago
README.md updated readme 5 years ago
README
Apache-2.0 license
ROS2 Packages for Scout Mobile Robot
Packages
This repository contains minimal packages to control the scout robot using ROS.
scout_base: a ROS wrapper around ugv_sdk to monitor and control the scout robot
scout_description: URDF model for the mobile base
scout_msgs: scout related message definitions
About
ROS2 Support Package for Scout Robot
Readme
Apache-2.0 license
Activity
34 stars
1 watching
43 forks
Report repository
Releases
No releases published
Packages
No packages published
Languages
C++ 54.5%
Python 34.4%
CMake 10.9%
Shell 0.2%



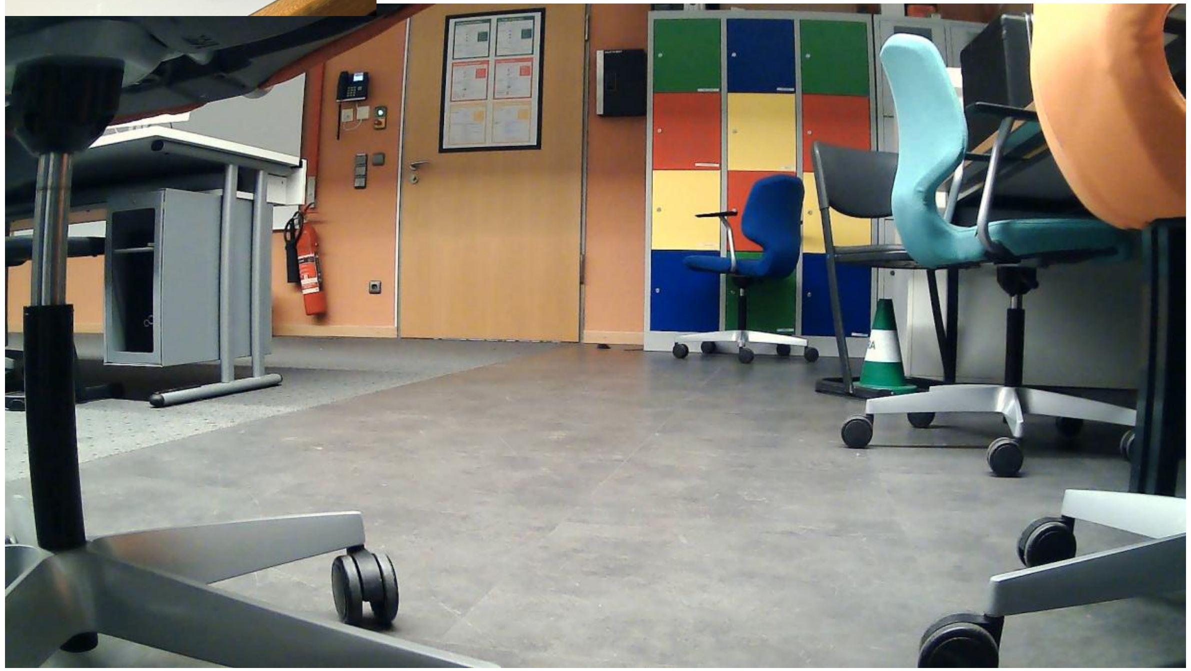
westonrobot / scout_ros2
Code
Issues 1
Pull requests 2
Actions
Projects
Security
Insights
scout_ros2 Public
Watch 1
Fork 69
Star 23
humble
2 Branches
0 Tags
Go to file
Add file
Code
hanskw-weston Merge pull request #11 from westonrobot/fix-auto_reconnect_can_control
34bf5df · 2 years ago
19 Commits
scout_base scout_base: launch: scout_base.launch: cleanup launch file in... 2 years ago
scout_description urdf: add scout mini meshes and urdf 2 years ago
scout_msgs initial commit, code compiles but not working yet 5 years ago
.gitignore scout_ros2: updated .gitignore 2 years ago
.gitlab-ci.yml initial commit, code compiles but not working yet 5 years ago
LICENSE Initial commit 5 years ago
README.md fixed for scoutv2, added QoS settings 4 years ago
README
Apache-2.0 license
ROS2 Packages for Scout Mobile Robot
Packages
This repository contains minimal packages to control the scout robot using ROS.
scout_base: a ROS wrapper around ugv_sdk to monitor and control the scout robot
scout_description: URDF model for the mobile base
scout_msgs: scout related message definitions
Supported Hardware
About
ROS2 Support Package for Scout Robot
Readme
Apache-2.0 license
Activity
Custom properties
23 stars
1 watching
69 forks
Report repository
Releases
No releases published
Packages
No packages published
Contributors 6
Languages
C++ 55.5%
Python 34.0%
CMake 10.3%
Shell 0.2%

First ROS2 operation

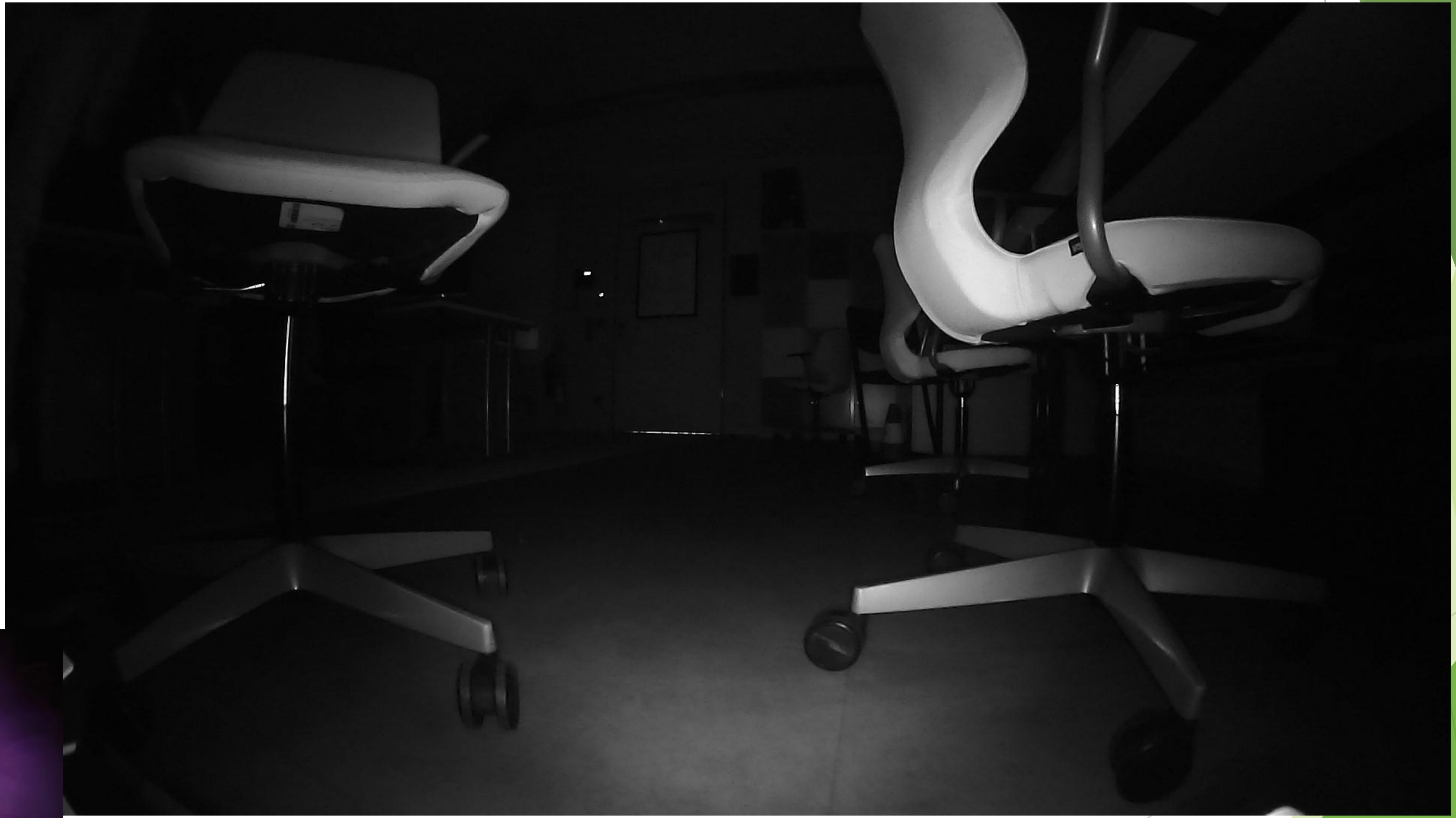


Camera

Camera

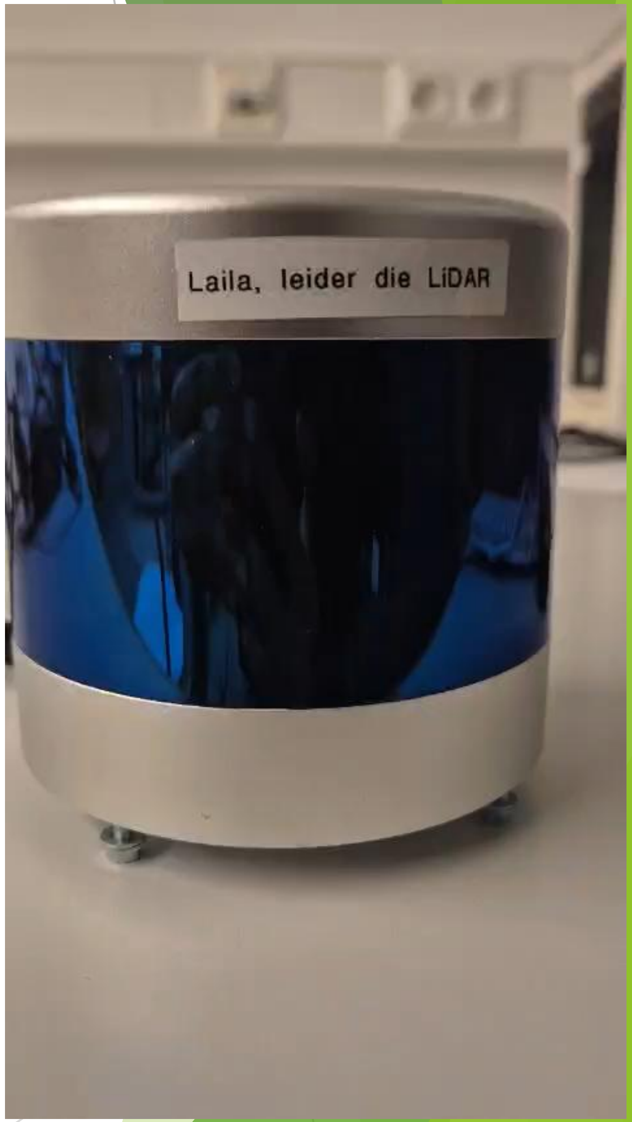


Camera



Leihsen C16 LIDAR

- Data Transmission was faulty
- Trial and Error: Check IP address, .config files, connection, Byte rate etc
- Configurations were right but problem remains unknown
- Conclusion: Change the LIDAR



RS Helios 16P

- LIDAR and Jetson are connected with Ethernet
- LIDAR send the data to a pre-configured IP address
- Sends data in UDP packets and publishes them in /rslidar_points

Source	Destination	Protocol	Length	Info
192.168.26.60	192.168.26.20	UDP	1290	6699 → 6699 Len=1248
192.168.26.60	192.168.26.20	UDP	1290	6699 → 6699 Len=1248
192.168.26.60	192.168.26.20	UDP	1290	6699 → 6699 Len=1248

In the Terminal:

1. Change the IP address of the Host Device to 192.168.26.20, LIDAR sends the data on this IP:

```
sudo ip addr add 192.168.26.20 dev eth0
sudo ip link set up dev eth0
```

2. Check if the IP address is up:

```
ip a                           # Should show IP of eth0 as set above
ip neigh                       # Should show eth0 as STALE or REACHABLE
```

3. ***OPTIONAL*** -> To check if the LIDAR is being detected and sending the UDP packets respectively:

```
sudo apt install wireshark     # Check if eth0 is STALE or REACHABLE
sudo tshark -i eth0 -f "udp"
```

4. Activate ROS and Launch:

```
source /opt/ros/humble/bash
cd rs_lidar
source install/setup.bash
ros2 launch rslidar_sdk start.py
```

5. Visualize the Point Cloud:

```
ros2 topic list               #You should see /rslidar_points
rviz2
```

In RIVZ2 -> In Global Options, set Fixed Frames to **rslidar** and Under PointCloud2 set Topic to **/rslidar_points**

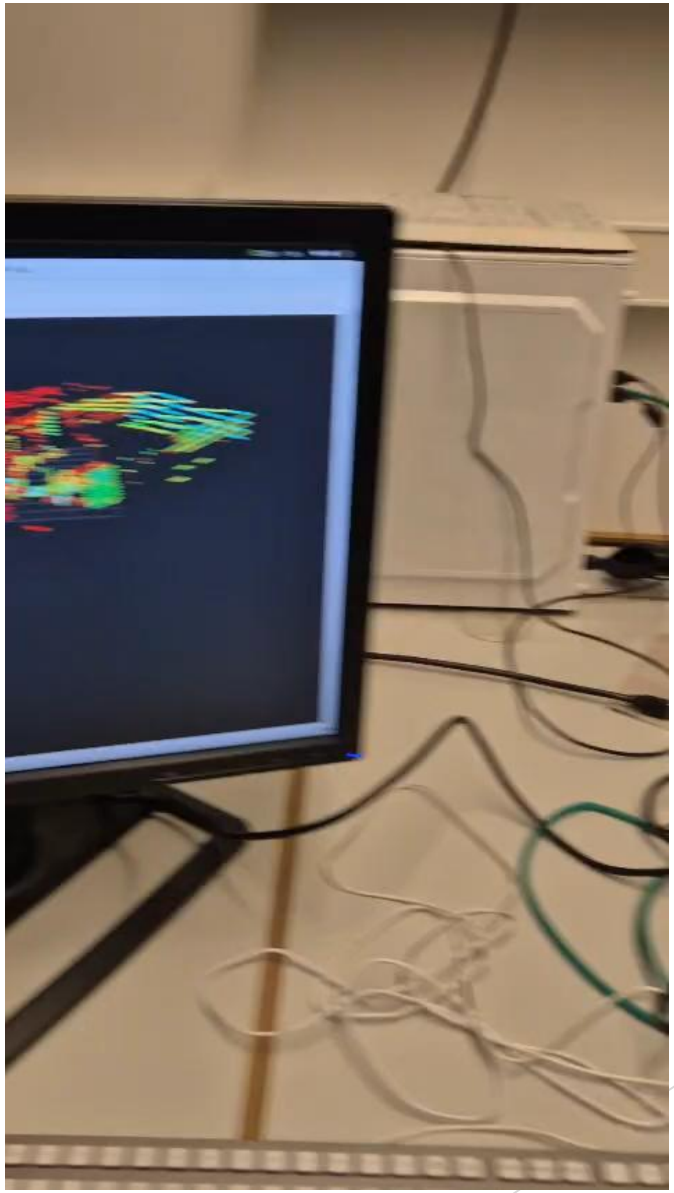
You should see the Points on the right Visualization Screen

Miscellaneous: LIDAR IP: 192.168.26.60

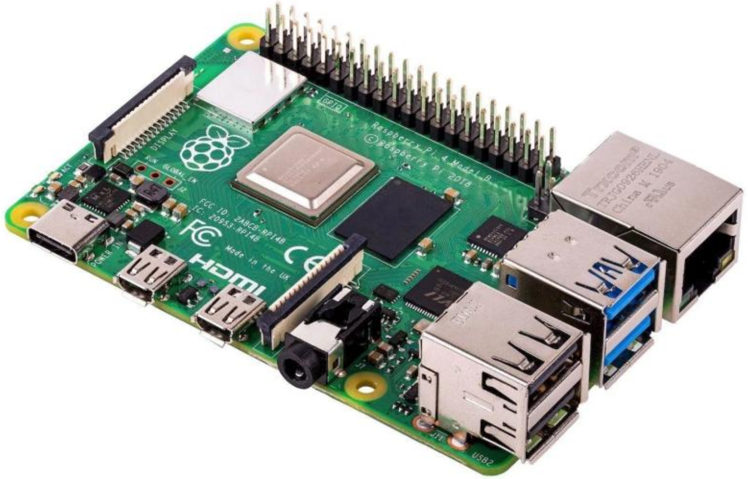
LIDAR Start-Up Script

RVIZ

- Visualizer
- Captures the Data from */rslidar_points*



CPUs and GPUs



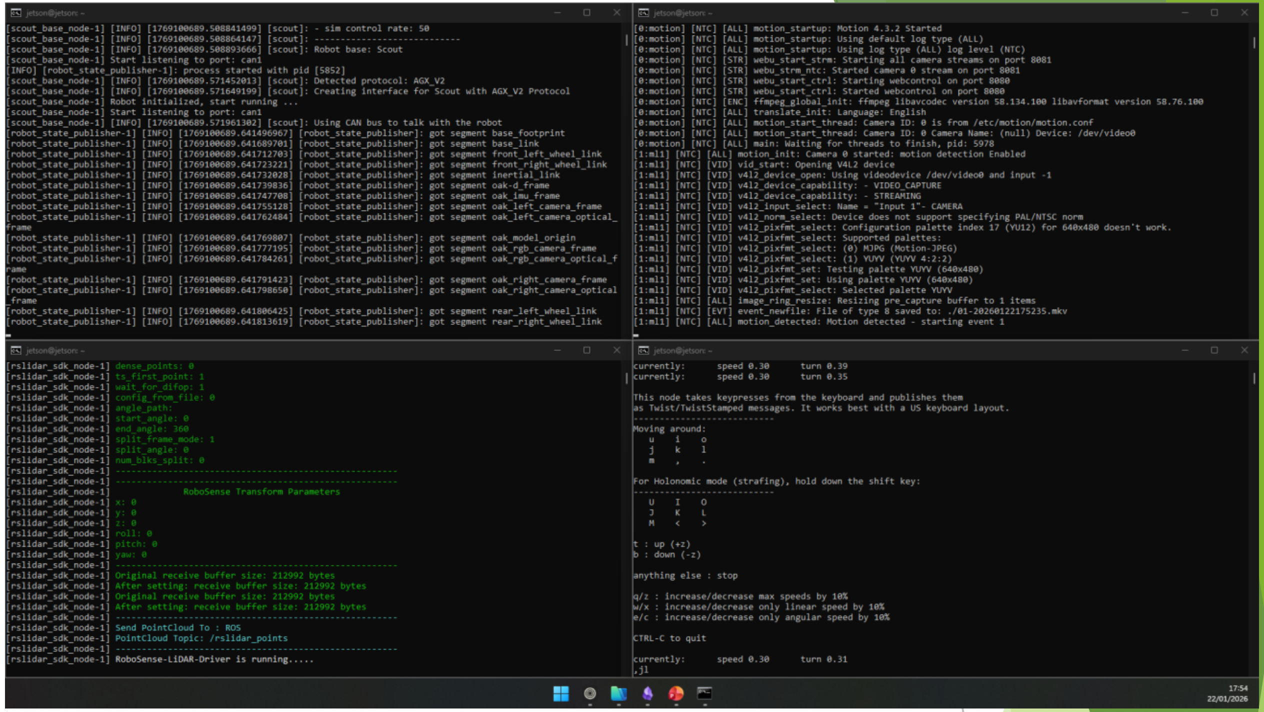
Cannot handle LiDAR (low processing power)

No official support for Ubuntu 22.04

Kernel not configured with CAN by default

Startup Script & Remote Operation

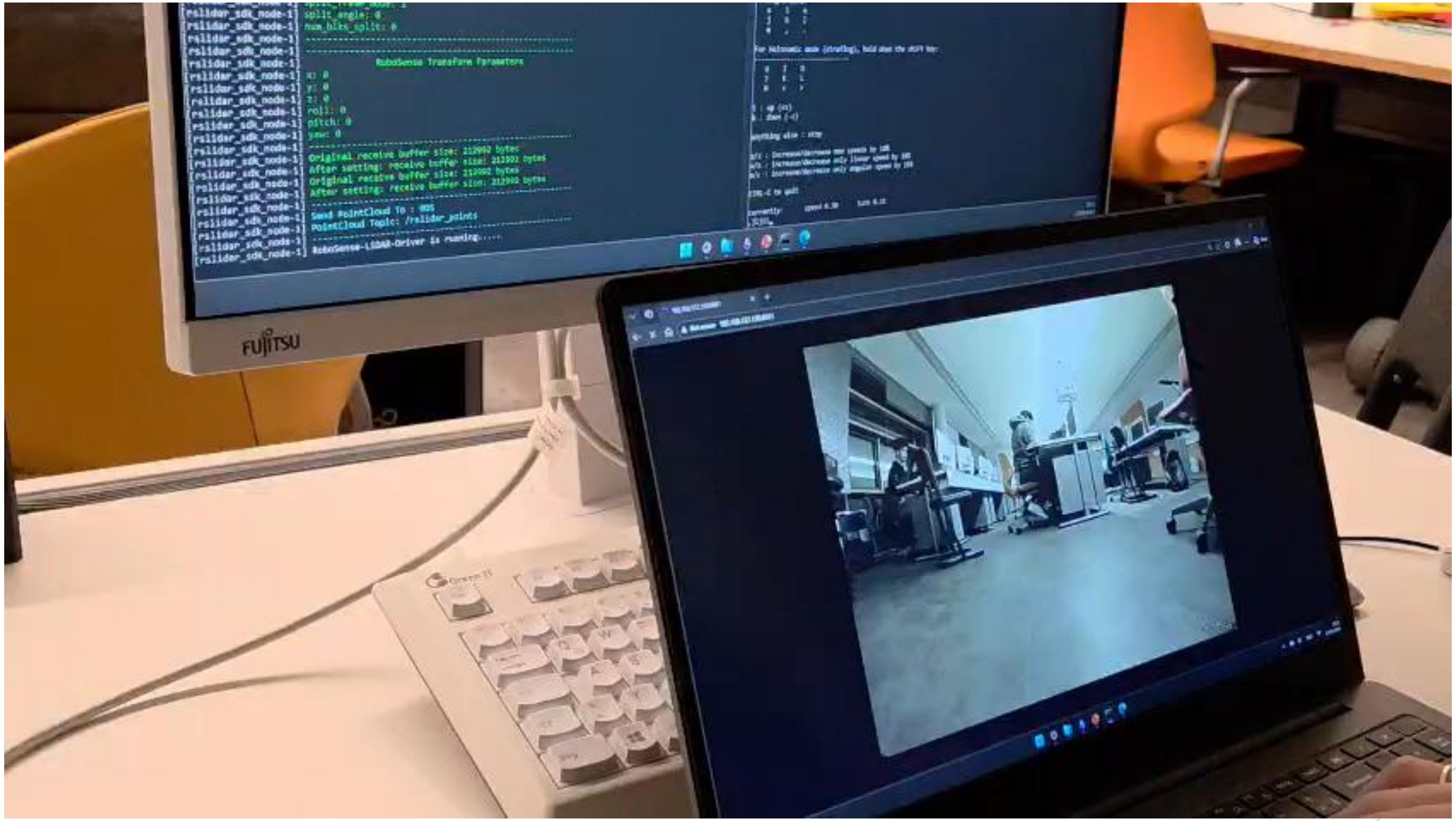
```bash
#!/bin/bash

# Installation Instructions:
# 1. Place the script at /usr/local/bin
#    (e.g., sudo cp scout_control.sh /usr/local/bin/rover)
# 2. This script name should be rover
# 3. Make it executable: sudo chmod +x /usr/local/bin/rover
#
# Usage:
# Always launch using: rover {base|teleop|lidar|camera}

# Define the workspace path - Change this if your workspace is elsewhere

ROVER_PATH="$HOME/ros2_ws"
LIDAR_WS_PATH="$HOME/rs_lidar"


if [ -z "$1" ]; then
    echo "Usage: $0 {base|teleop|lidar|camera}"
    exit 1
fi

setup_can() {
    echo "Checking CAN interface..."
    if ip link show can0 | grep -q "UP"; then
        echo "CAN0 is already up."
    else
        echo "Setting up CAN0..."
        sudo modprobe gs_usb
        sudo ip link set can0 up type can bitrate 500000

        if [ $? -eq 0 ]; then
            echo "CAN0 setup successful."
        else
            echo "Failed to setup CAN0. Check connections."
            exit 1
        fi
    fi
}

get_wifi_ip() {
    ip -4 addr show | grep -E 'wlan|wlp|wlx' | grep inet | awk '{print $2}' | cut -d/ -f1 | head -n 1
}

source /opt/ros/humble/setup.bash

case "$1" in
```



First remote operation

Final lab remote operation

Final lab remote operation

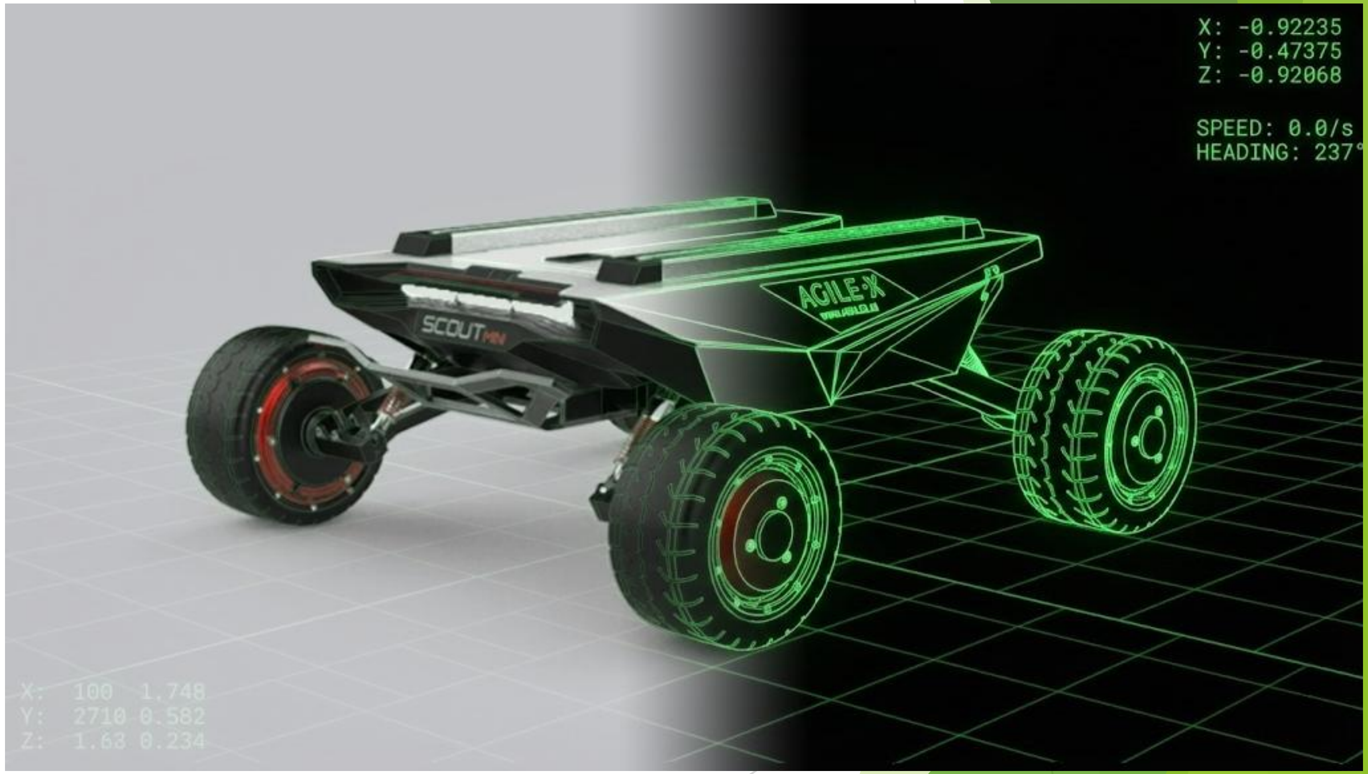


Field remote operation

Simulation

Objectives, roles and goals of the simulation in the project

- To familiarize the team with ROS 2 concepts;
- Develop a safe, reliable testing environment for the "Scout Mini" UGV;
- Isolate hardware problems from coding problems;
- Solving problems like the "Perception Disparity" (syncing movement in Gazebo vs. RViz);
- Finally achieve autonomous navigation and dynamic pathfinding.



Technologies used

Operating System: Ubuntu 22.04

- Stable Linux distribution
- Long-term support (LTS)
- Official support for ROS 2 Humble

Robot Middleware : ROS 2 Humble

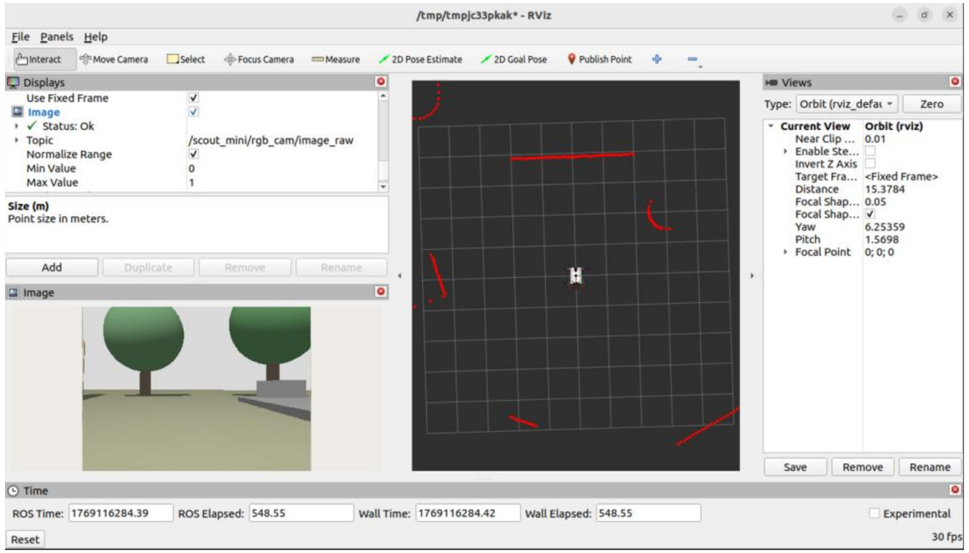
- Node-based architecture
- Topic-based communication
- Modular and scalable

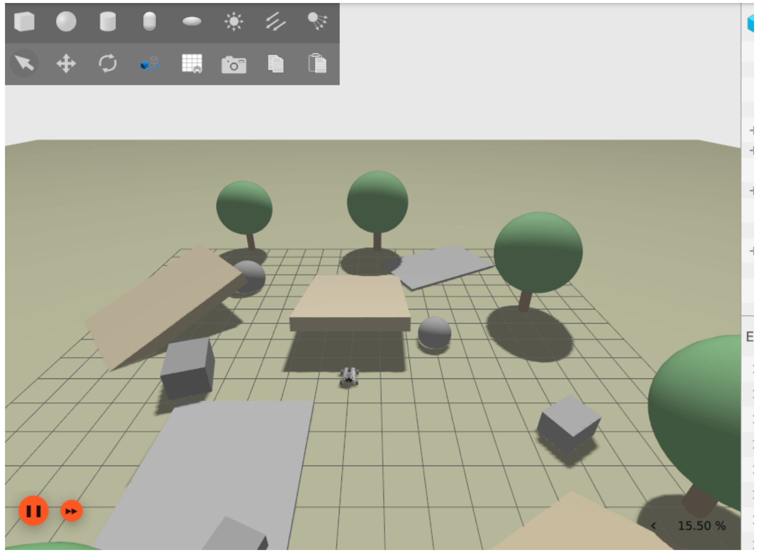
Simulation Engine: Gazebo Fortress

- Physics-based simulation
- 3D environment rendering
- Sensor simulation

Visualization Tools: RViz 2

- Robot visualization
- Sensor data display
- Map visualization


/tmp/tmpjc33pkak* - RViz
File Panels Help
Interact Move Camera Select Focus Camera Measure 2D Pose Estimate 2D Goal Pose Publish Point
Displays
Use Fixed Frame
Image
Status: Ok
Topic /scout_mini/rgb_cam/image_raw
Normalize Range
Min Value 0
Max Value 1
Size (m)
Point size in meters.
Add Duplicate Remove Rename
Image
Views
Type: Orbit (rviz_defau Zero
Current View Orbit (rviz)
Near Clip ... 0.01
Enable Ste...
Invert Z Axis
Target Fra... <Fixed Frame>
Distance 15.3784
Focal Shap... 0.05
Focal Shap...
Yaw 6.25359
Pitch 1.5698
Focal Point 0; 0; 0
Save Remove Rename
Time
ROS Time: 1769116284.39 ROS Elapsed: 548.55 Wall Time: 1769116284.42 Wall Elapsed: 548.55
Experimental
Reset
30 fps


15.50 %

Robot & Environment Description Formats

- URDF / Xacro → Robot model
- SDF → Simulation world

ROS–Gazebo Communication: ros_gz bridge

.Connects ROS 2 and Gazebo

.Sensor data → ROS

.Control commands → Gazebo

Navigation & Mapping Tools

.Slam toolboxs

. Nav 2

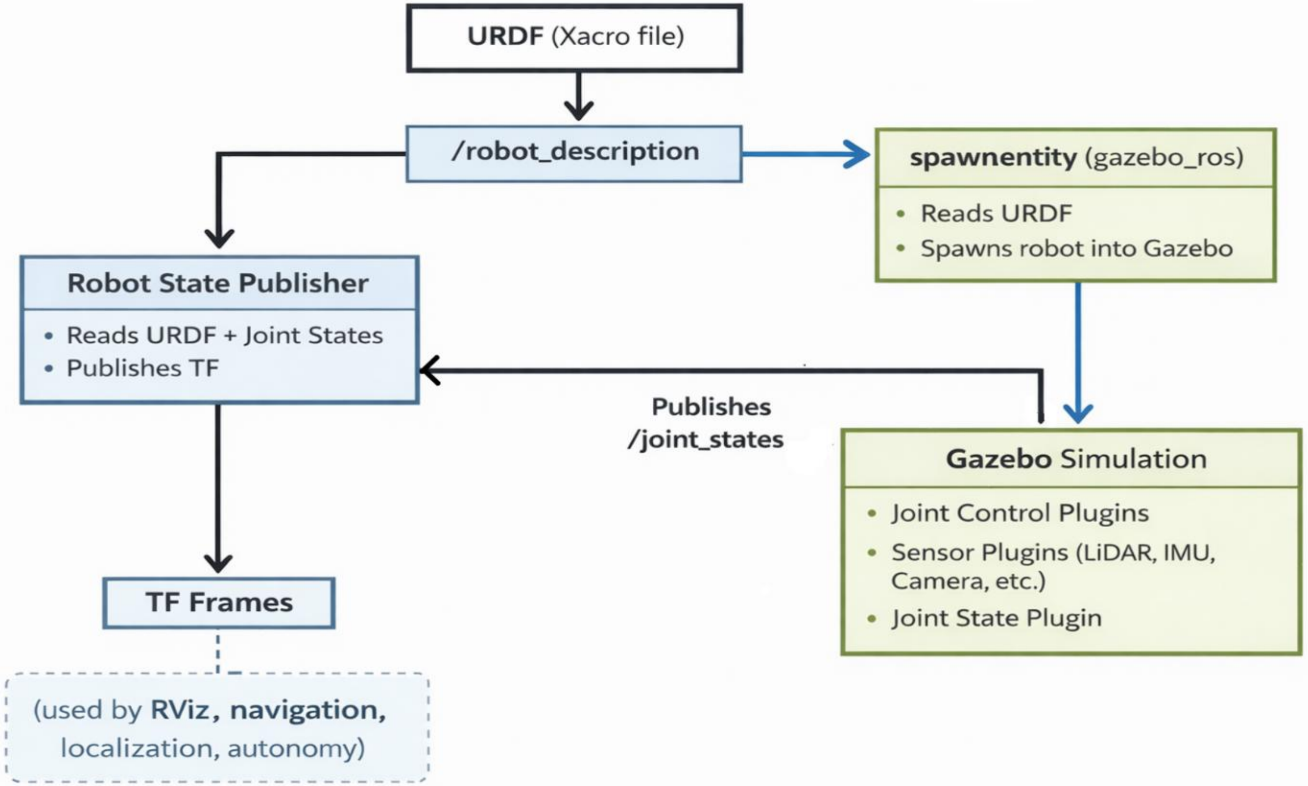
Programming & Build Tools

- python
- Colcon
- rosdep

languages

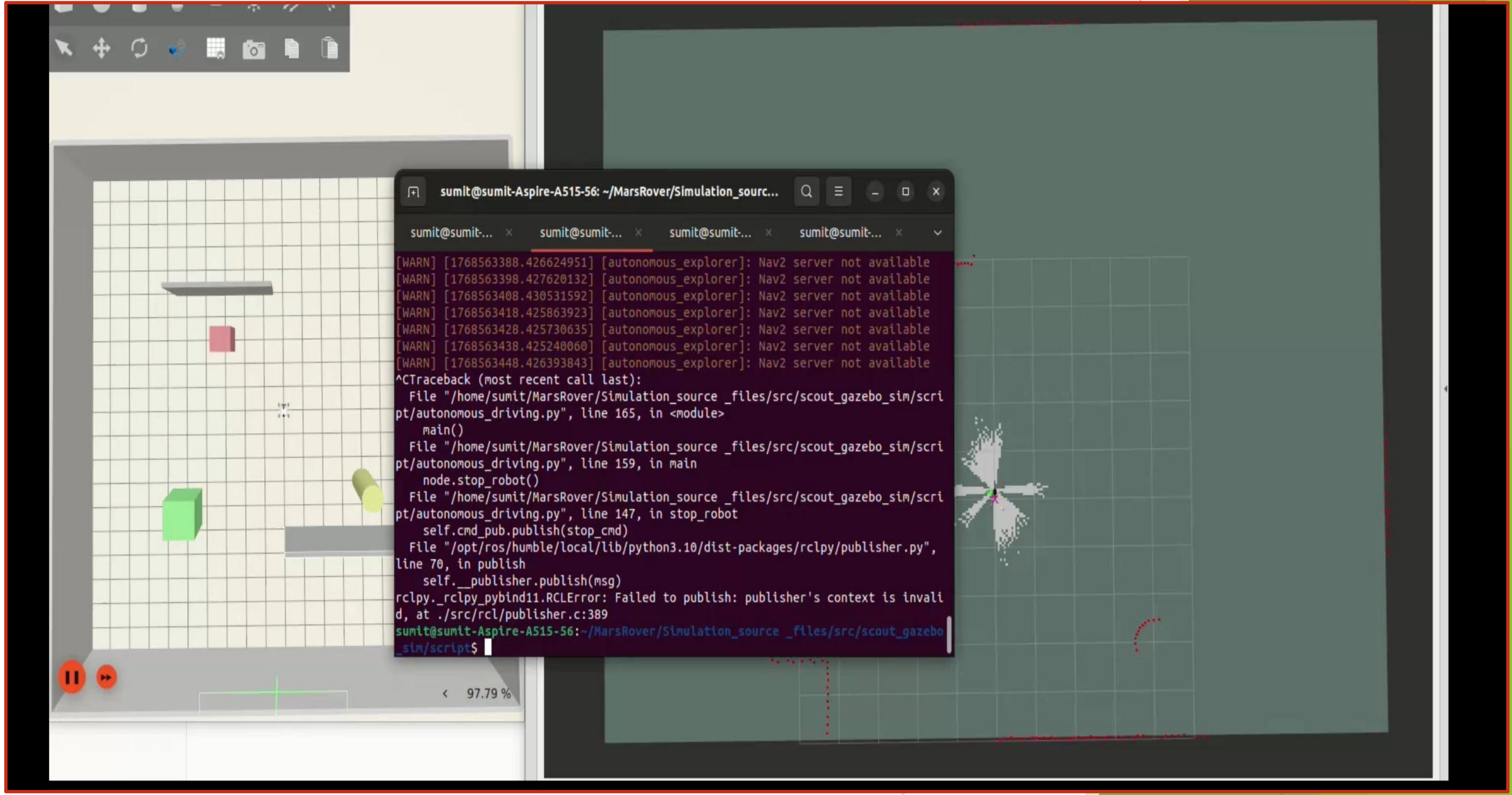
.XML
.python

Design of the simulation environment



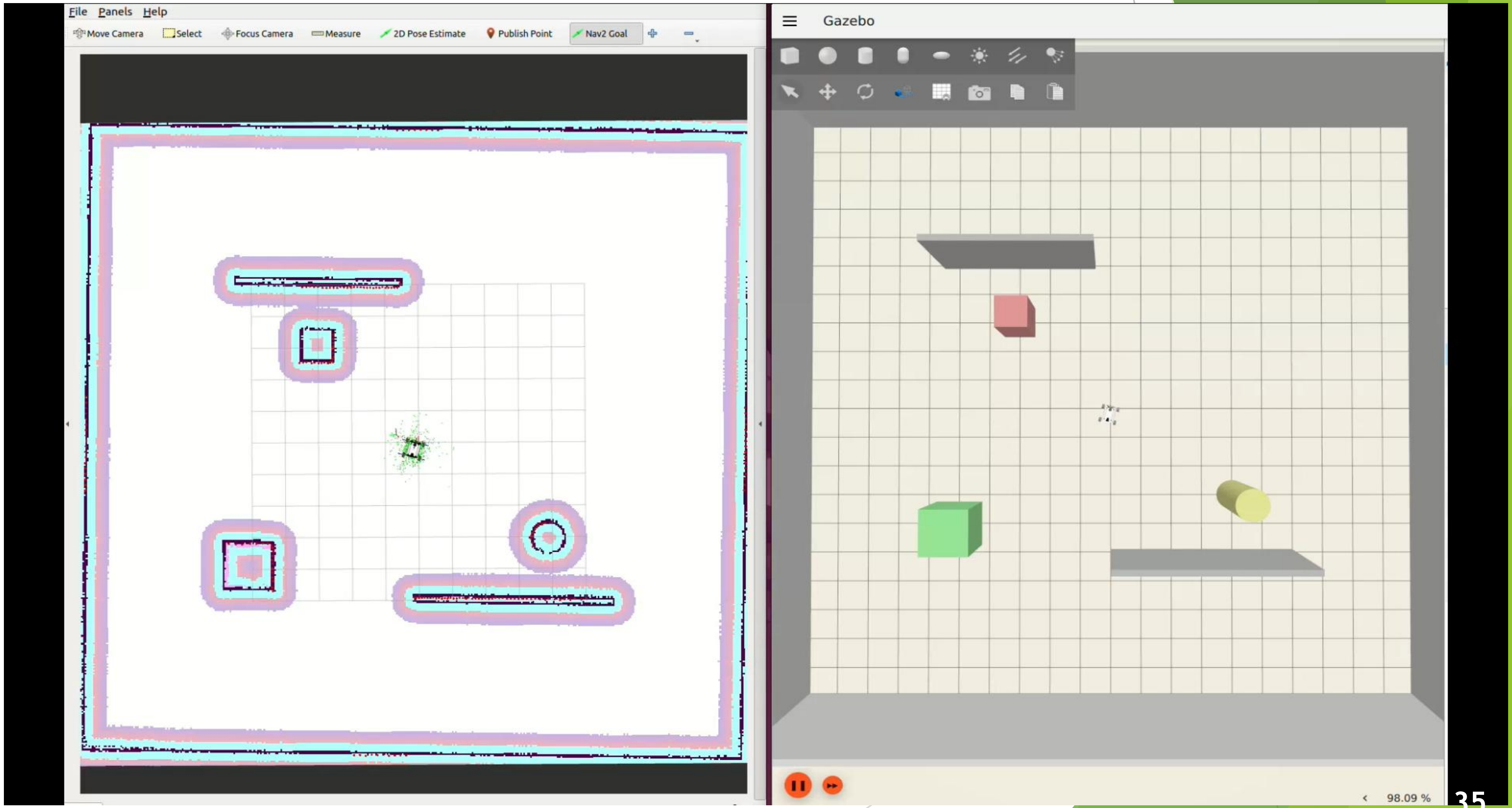
What it looks like

- SLAM



What it looks like

- Nav2



Challenges in the simulation

Challenge: Disparity between the Ground Truth (Gazebo) and the Robot's Perception (RViz).

Solution: Calibrated wheel radius parameter to sync physical movement with digital transforms (TF).

Challenge: Navigation in unmapped environments(no static global map).

Solution: integrated a custom Nav2 Action client with a dynamic goal generator.

Challenge: Latched Velocity Persistence (The rover won't stop).

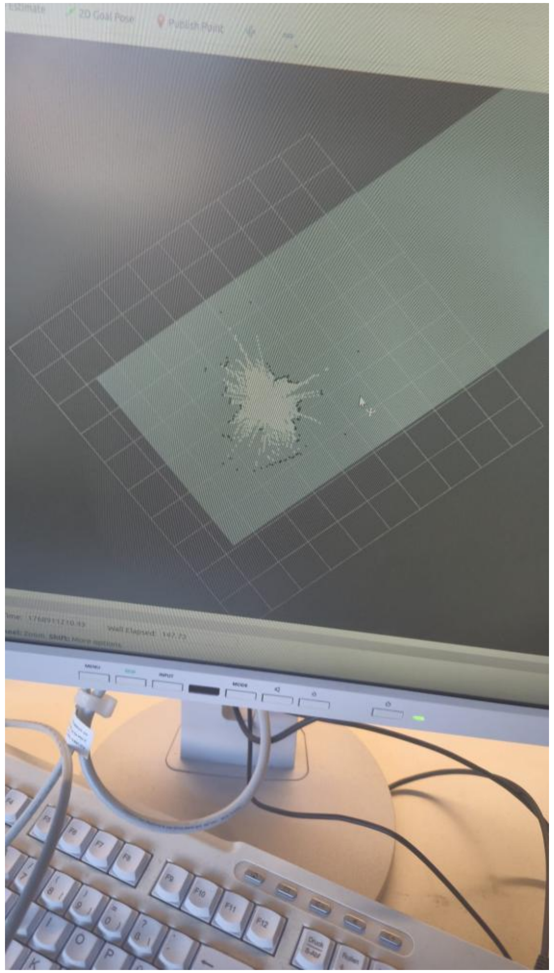
Solution: Implemented POSIX Signal Handlers and a velocity flush routine.

36

Integration and Final Results

Integration

- Launch all nodes together: Scout, LIDAR, SLAM
- For that there is a launch file: master_launch.py




Jan 20 13:15
RViz*
2D Pose Estimate
2D Goal Pose
Publish Point

References

- How this helicopter survived 1004 days on Mars, then disappeared..., Veritasium, available at https://www.youtube.com/watch?v=20vUNgRdB4o
- ROS2 Packages for Scout Mobile Robot, available at https://github.com/westonrobot/scout_ros2
- ROS2 Packages for RS Lidar, available at https://github.com/RoboSense-LiDAR/rslidar_sdk/tree/main
- Structure of the simulation environment for scout_mini available at https://github.com/agilexrobotics/ugv_gazebo_sim/tree/master/scout
- SLAM implementation in the simulation avialable at https://youtu.be/ZaiA3hWaRzE