1 | Robotic Platform for Mobile Manipulation

This chapter will describe the robotic platform used for the project, specifically the hardware components. The platform is composed of a mobile robot base, a robotic arm manipulator, sensors and perception systems, a soft gripper actuator, 3D-printed mounts, batteries, power management systems, and other electronic devices. The chapter will also discuss the issues faced during the development of the mobile manipulation platform.

1.1. Mobile Robot Platform

The mobile robot used for the Thesis project is an *AgileX Scout 2.0* robot. This robot is a skid-steering robot, suitable for outdoor and indoor environments. Designed for robotics research and development, the Scout 2.0 is an unmanned ground vehicle (UGV). This autonomous mobile robot is CE-certified and offers a robust mechanical design along with capable mobility performance. Built to endure diverse conditions, Scout 2.0 features rugged materials and protective casings, ensuring longevity and reliability during missions. SCOUT 2.0 offers aluminum T-slot rails for secure mounting of external sensors or kits. On these rails, a variety of sensors, computers or other devices are mounted, allowing the creation of a mobile robotic platform for a wide range of applications, and without relying on power cables to the wall, thanks to its **onboard battery system**.

It supports CAN bus protocol for connections and provides open-source SDK and software resources for expanded capabilities. Its maximum speed is $1.5 m/s$, and it can carry a payload of 50 kg. The robot is powered by a 24V battery, which provides a range of 15 km maximum. The robot is controlled by a ROS-based software system, which allows the control of the robot's speed using a ROS topic and receiving odometry data from the robot's encoders.

On the mobile robot, an *Intel NUC 12* computer is mounted. This **computer** is used for running all the control software and the perception algorithms. The computer is connected

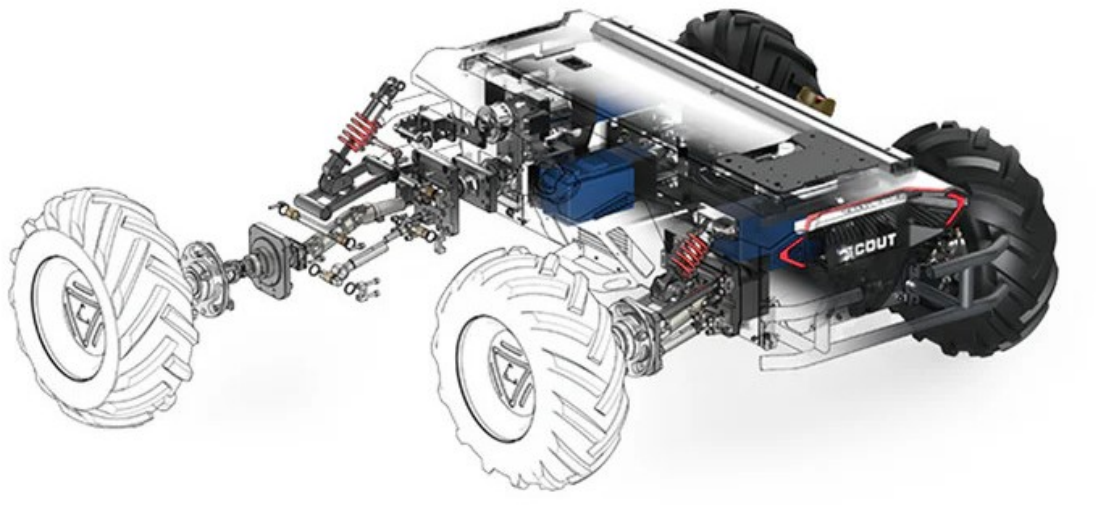


Figure 1.1: SCOUT employs 200W brushless servo motors to drive each wheel independently. Its double-wishbone suspension with shock absorbers ensures stability on rough terrain, enabling it to tackle obstacles up to 10cm tall effortlessly.

.

to a switch, which is used to connect the computer to the robot's control system and all the sensors and electronic devices mounted on the robot. The computer is also connected to a router, which is used to connect the computer to the laboratory's network and the internet. This allows the robot to be controlled remotely via a remote desktop connection. This computer has the following technical specifications:

- Intel Core i7-12700H CPU
- 32GB DDR4 RAM
- 1TB NVMe SSD
- Intel Iris Xe Graphics
- Kubuntu 22.04 operating system

The robot is equipped with an *TP-Link Archer MR200* router and a *Netgear GS108* switch. The router is used to connect the robot to the laboratory's network and the internet. The router was necessary to establish a **remote connection** from a personal laptop to the robot's computer, allowing the control and monitoring of the robot from a remote location. This was essential for the development of the project, as it allowed me to work safely with the robot, ensuring that everything was working smoothly and stopping the system in case of any unexpected behavior or software crashes and malfunctions.

Figure 1.2: Intel NUC 12 computer mounted on the robot

.

The switch is used to connect the robot's computer to the robot's control system and all the sensors and electronic devices mounted on the robot. The switch is connected to the router, allowing the robot's computer to communicate with the laboratory's network and the internet. The robotic arm manipulator is also connected to the switch, allowing the robot's computer to control the manipulator and receive data from its motors' encoders. The LiDAR is connected to the switch, allowing the robot's computer to receive pointcloud data from the LiDAR sensor, at a fast transmission rate.

1.2. Robotic Arm Manipulator

The robotic arm manipulator used for the Thesis project is a *Igus ReBeL 6-DoF* **cobot**. Cobot is a term used to describe a collaborative robot, which is a robot designed to work alongside humans in a shared workspace. This cobot is a lightweight, compact, and affordable robotic arm, suitable for research and development in robotics. It is produced by the German company Igus, which specializes in the production of robotic components for low-cost automation. The robotic arm is composed of six joints, each driven by a DC motor with an integrated encoder. The outer contour and mechanical components of the ReBeL utilize Igus® plastic polymers, making it particularly inexpensive and the lightest cobot on the market. Its lightweight and compact design makes it suitable for mounting on top of mobile robot platforms, such as the Scout robot. The maximum payload of the arm is 2 kg, which is more than enough for the project's requirements. The weight is 8.2 kg, which is light enough to be carried around by the mobile robot base, without affecting the robot's mobility and stability.

The **advantages** of the ReBeL cobot are:

- Lightweight, sleek and compact design

Figure 1.3: Igus ReBeL 6-DoF robotic arm (cobot)

.

- Plastic arm, inexpensive and cost-effective
- Easy to install and operate
- Plug and play proprietary control system, or open-source control option

The **disadvantages** of the ReBeL cobot are:

- Limited reach and workspace due to the joints' design and physical constraints
- Limited precision and repeatability
- The plastic gear components are not as durable and reliable as metal components

This robotic arm was the ideal choice for the project since the open-source control option allowed me to develop the control software for the arm, based on ROS2 software packages. The lightweight and compact design made it suitable for mounting on top of the SCOUT 2.0 robot, without affecting the robot's mobility and performance. The arm's easy installation and operation allowed me to quickly set up the arm and start developing the control software and perception algorithms for the project.

Issues with Igus Rebel motors' encoders and calibration

Throughout the development of the project, I encountered several issues with the Igus ReBeL cobot. The issues arose when the arm's motors' encoders were not providing accurate position data to the control software. This caused the arm not to move to

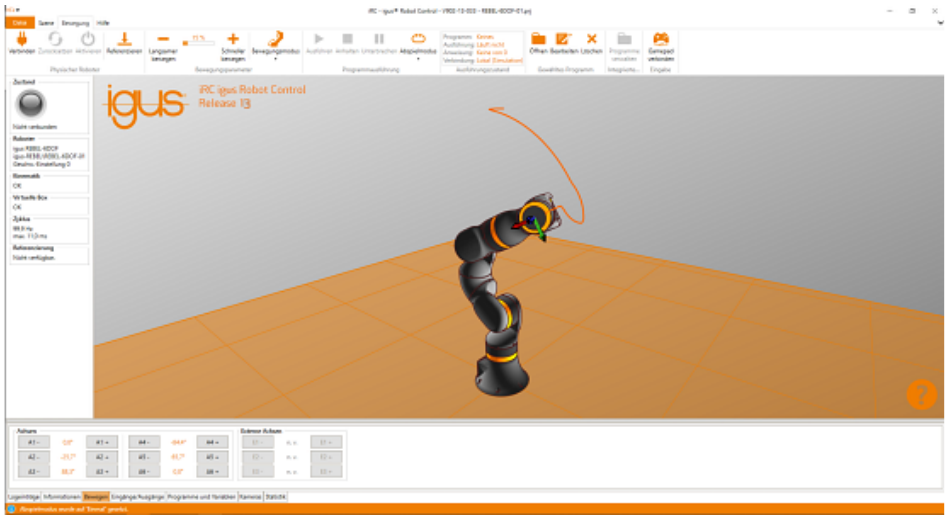


Figure 1.4: Igus ReBeL proprietary control software interface

.

the defined joint positions with an accuracy of the end effector of $\pm 0.1mm$, as specified by the manufacturer in the technical documentation. This was mainly due to both the calibration of the motors' encoders and the plastic gear components' backlash and play. The backlash and play in the gear components caused the arm's joints to have a certain amount of free movement before the motor started to move the joint. This free movement caused the arm's joints to be in an incorrect position, which was not detected by the motor's encoders. This resulted in the arm jiggling and vibrating when moving to a new joint position, and not reaching the desired position with a smooth velocity profile.

Furthermore, the main issue encountered was the **calibration of the motors' encoders**. This problem slowed down the development of the project since the software solutions for the calibration of the encoders and the digital zeroing of the joints were not successful in providing a working solution for the arm's accurate positioning. The calibration of the encoders was necessary to provide accurate position data to the control software, allowing the arm to move to the desired joint positions with high precision. The problem encountered was that the software for the automatic calibration of the motor controllers did not work as expected, resulting in errors and faulty calibration of the encoders. The only solution was to replace the entire robot arm with a new one, correctly calibrated by the manufacturer.

1.3. Sensors and Perception

The mobile robot platform is equipped with several sensors and perception systems, which are essential for the project's objectives. The sensors and perception systems are used

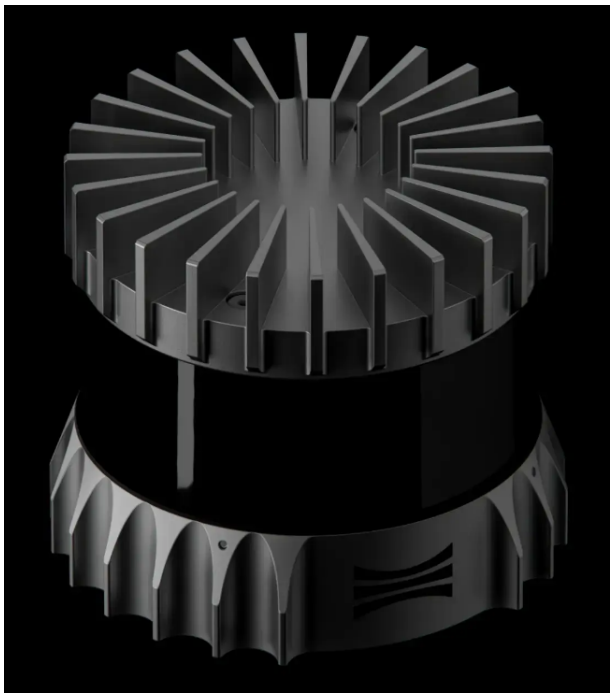


Figure 1.5: Ouster OS1-64 LiDAR sensor

.

for mapping and localization, obstacle avoidance, object detection and recognition, and manipulation tasks. Two main sensors are installed on the robot: a 3D LiDAR sensor and an RGB-D stereo camera sensor. The LiDAR sensor is in a fixed position, on top of the robot to have a 360-degree field of view, while the camera sensor is mounted on the robotic arm's wrist, allowing the camera to move with the arm's end effector.

1.3.1. 3D LiDAR Sensor

The main sensor for environment perception for localization and navigation is mounted on the robot's T-slot rails. The sensor is a *Ouster OS1-64* LiDAR sensor. The OS1 offers clean, dense data across its entire field of view for accurate perception and crisp detail in industrial, automotive, robotics, and mapping applications. This sensor is a **64-plane LiDAR sensor**, capable of providing a 360-degree field of view with a range of 120 meters. The sensor has a resolution of $\pm 0.1cm$ and a stable scan rate of $10Hz$ at 1024 points resolution. The vertical and horizontal scan resolution are $\pm 0.01°$. Its minimum range of 0.5 meters makes it suitable for indoor environments, while its maximum range of 120 meters makes it suitable also for outdoor environments. This sensor was employed to create maps of the environments and also to localize the robot within the environment. It proved also useful for dynamic obstacle avoidance, thanks to its high resolution and scan rate.

1.3.2. Issues with LiDAR low frequency

Throughout the development of the project, I encountered several issues with the autonomous navigation stack. Sometimes the software crashed at random times, unpredictably, and without any apparent reason. Some other times the dynamic obstacle avoidance algorithm did not work as expected, causing the robot to collide with obstacles that were not perceived by the LiDAR sensor in the surrounding environment. Overall, the robot's navigation stack was not reliable and robust, which was a significant issue for the project's development, especially for the autonomous navigation tasks in the laboratory's environment (cluttered and dynamic).

After many trials and tests, my supervisor and I discovered that the LiDAR pointcloud data's low and unreliable frequency was the main cause of all these problems. The LiDAR sensor works at a nominal frequency of 10Hz. The LiDAR sensor was not providing a stable scan rate of 10Hz, but it was fluctuating between 3Hz and 8Hz, and only in a few cases, it was able to provide the nominal 10Hz scan rate. Once we discovered the source of the problem and ensured that the LiDAR sensor was working correctly (meaning that the sensor was not malfunctioning or working at a lower but valid data rate), we tried to find the culprit of the low and unreliable frequency of the LiDAR sensor.

After many attempts, I found out the cause of the problem: the culprit was the **router** used to connect the robot's computer to the laboratory's network and the internet and the DDS (*Data Distribution Service*) middleware used internally by ROS2 for the communication between the nodes in a local network. Basically, the DDS middleware was configured to broadcast all ROS2 packets and data streams to the laboratory's network, causing delays in the transmission and packet loss, as the network was not able to handle the high-frequency and heavy-weight data streams of the LiDAR sensor. In fact, by just unplugging the router from the robot's computer, the LiDAR sensor was able to provide a stable scan rate of 10Hz, without any fluctuations or drops in the frequency, since no broadcast packets were sent to the network. Since the router was necessary for the remote control and monitoring, it was simply just not an option to unplug it from the robot's computer.

For the correct operation of the robot's localization and obstacle avoidance algorithms, it is fundamental that the LiDAR sensor works at a **stable frequency**, without any fluctuations or drops in the scan rate. This requirement is essential because these algorithms rely on the coherent and continuous data stream for time interpolation and filtering of the pointcloud data. This is why the LiDAR sensor's low and unreliable frequency was causing the software to crash and malfunction.

The solution was to configure properly the **DDS middleware** to ensure that all ROS-related nodes and data streams were handled inside the robot's computer only, and not going through the router and the laboratory's network. Having the data streams in the local machine only, ensured that the LiDAR sensor's data was not distributed among the network, causing delays in the transmission and packet loss, which were the main causes of the low and unreliable frequency of the LiDAR sensor. Furthermore, we configured ROS2 to use the *Cyclone DDS* middleware, which is a lightweight and fast DDS implementation, suitable for real-time and high-frequency data streams. This DDS implementation was able to handle heavy-weight data packets at high frequencies, ensuring that the LiDAR sensor's data was transmitted and received correctly in its entirety, without any losses or delays in the transmission.

After this configuration, the LiDAR sensor was able to provide a stable scan rate of 10Hz, and the software stack was working correctly, without any crashes or malfunctions. Furthermore, I was also able to test the LiDAR sensor at a higher frequency of 20Hz, with lower resolution, and it was working fine and stable. This solution was essential also for the correct operation of the IMU sensor for the SLAM algorithm, which required a stable and continuous data stream of the LiDAR at the moment of startup of the sensor in the software stack. Before the fix of the LiDAR sensor's frequency, the IMU sensor was not able to start correctly, due to the lack of data from the LiDAR sensor, and the computed odometry data was drifting too much to be useful. Many other algorithms and software benefitted from this fix. The robot's navigation stack was now reliable and robust, and the autonomous navigation tasks were working correctly and smoothly.

1.3.3. RGB-D Stereo Camera Sensor

The robotic arm is equipped with a *Intel Realsense D435* RGB-D stereo camera sensor. This camera is mounted on the *wrist* of the robotic arm, allowing the camera to move with the arm's end effector. The camera is used for object detection and recognition, and also for Aruco markers detection and pose estimation. This is the camera of choice for robotic applications, as it provides both RGB and depth images, which are essential for perception tasks in robotics. The camera is also lightweight and compact, making it suitable for mounting on the cobot.

The Intel RealSense D435 is a stereo depth camera that is designed for capturing RGB and depth images. The camera is equipped with a global shutter and a rolling shutter, which allows it to capture images with a resolution of 1920×1080 pixels at 30 frames per second, even though the ROS2 driver provided by Intel reduces the resolution to 640×480

Figure 1.6: Intel RealSense D435 RGB-D stereo camera

.

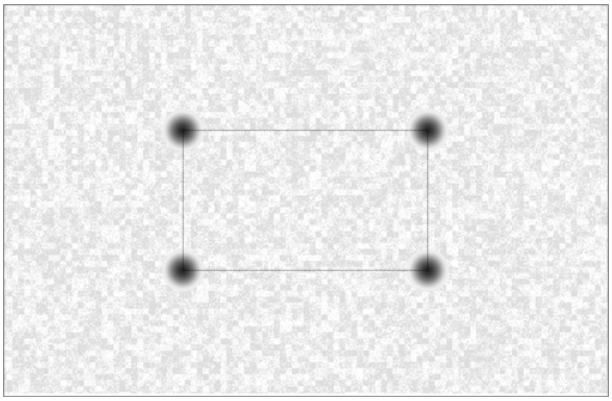


Figure 1.7: Calibration pattern used for the camera's depth image calibration

.

pixels at 30 frames per second. The camera has a field of view of 85.2° horizontal, 58° vertical, and 94° diagonal. The depth camera works within a range of 0.3 meters to 3 meters while maintaining high accuracy and precision.

1.3.4. Issues with Intel Realsense calibration

The camera provided by the laboratory was not calibrated correctly, especially the depth estimations were not accurate. In fact, the depth images provided depth estimations that were not consistent with the real-world distances of the objects in the environment. I was able to discover such issues when I started using the camera's depth sensor for the perception tasks of the project. I realized that the estimated distance from the camera and the Aruco markers (the distance estimated using geometrical calculations and the camera's intrinsic parameters) was **not consistent** with the estimated distance to the marker provided by the camera's depth sensor. This inconsistency was due to the camera's depth sensor not being calibrated correctly.

This issue was critical for the project, as the depth sensor was used for many perception algorithms. I managed to solve this issue by calibrating the camera's depth sensor using the proprietary software provided by Intel: *Interl Realsense Self-Calibration Tool*, an

application for **automatic on-chip calibration**. The calibration process was straightforward and required a few minutes to complete, even though many trials were needed to find the best possible calibration parameters. Two calibration procedures were carried out:

- **Intrinsic parameters calibration**: this calibration process was used to calibrate the camera's intrinsic parameters, such as the focal length, principal point, and distortion coefficients. This calibration was necessary to correct the distortion of the images and to provide accurate depth estimations from the RGB sensor. The calibration process required the camera to capture a series of images of a calibration pattern (checkerboard pattern) from different angles and distances. The calibration software then used these images to estimate the intrinsic parameters of the camera.
- **Depth sensor calibration**: this calibration process required the camera to capture a series of depth images of a calibration sheet 1.7, which was a flat surface with known distances between the points. The calibration software then used these depth images to estimate the depth sensor's parameters, such as the depth scale factor and the depth offset. These parameters were necessary to correct the depth estimations of the camera's depth sensor.

After the calibration process, the camera's depth sensor was providing accurate and consistent depth estimations, which were consistent with the real-world distances of the objects in the environment.

1.4. Soft Gripper Actuator

The robotic arm is equipped with a Soft Gripper Pneumatic Actuator from *Soft Gripping*. This gripper is composed of 3 soft fingers, which are actuated by a pneumatic pump. The fingers are made of **silicone rubber**, which is soft and flexible, allowing the gripper to grasp and manipulate objects of different shapes and sizes. The silicone material of the fingers is also non-slip, which ensures a secure grip on the objects. It is also essential in food industries, where the gripper is used to handle delicate and fragile objects, such as fruits. The other advantage is that silicone is non-toxic and food-safe, making it suitable for handling food products.

The soft gripper is controlled by a pneumatic pump, which provides compressed air to the fingers, allowing them to open and close. The pneumatic system works at a pressure of 1 bar when the fingers are closed, and at a negative pressure of 0 bar when the fingers are opened. The pneumatic pump that controls the soft gripper allows the user to set the

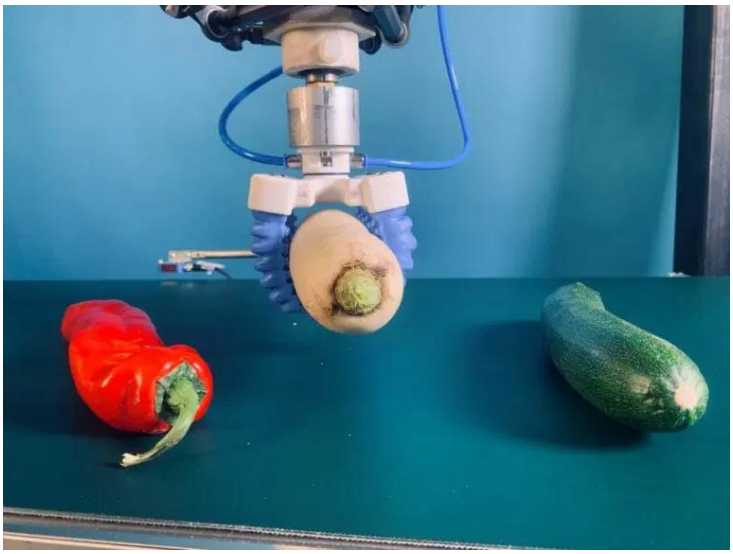


Figure 1.8: Soft Gripper Pneumatic Actuator handling vegetables

.

pressure value used when closing the fingers, but the pressure value cannot be changed dynamically via electronic control.

The pneumatic pump provided with the soft gripper has only one output tube used for providing positive pressure to the fingers. The negative pressure is provided by the environment, as the fingers are opened by the air pressure inside the fingers, which is lower than the air pressure outside the fingers. Other versions of the pneumatic pump provide also a negative pressure output tube, which allows the user to control the pressure value used when opening the fingers, but this feature requires an external vacuum compressor to work.

The gripper is mounted on the robotic arm's end effector, allowing the arm to grasp and manipulate objects in the environment. It is placed strategically close to the robotic arm's flange to ensure that the mobility and reach of the end effector are not affected by the gripper's size. Furthermore, the soft gripper is very lightweight and compact, making it suitable for mounting on the cobot.

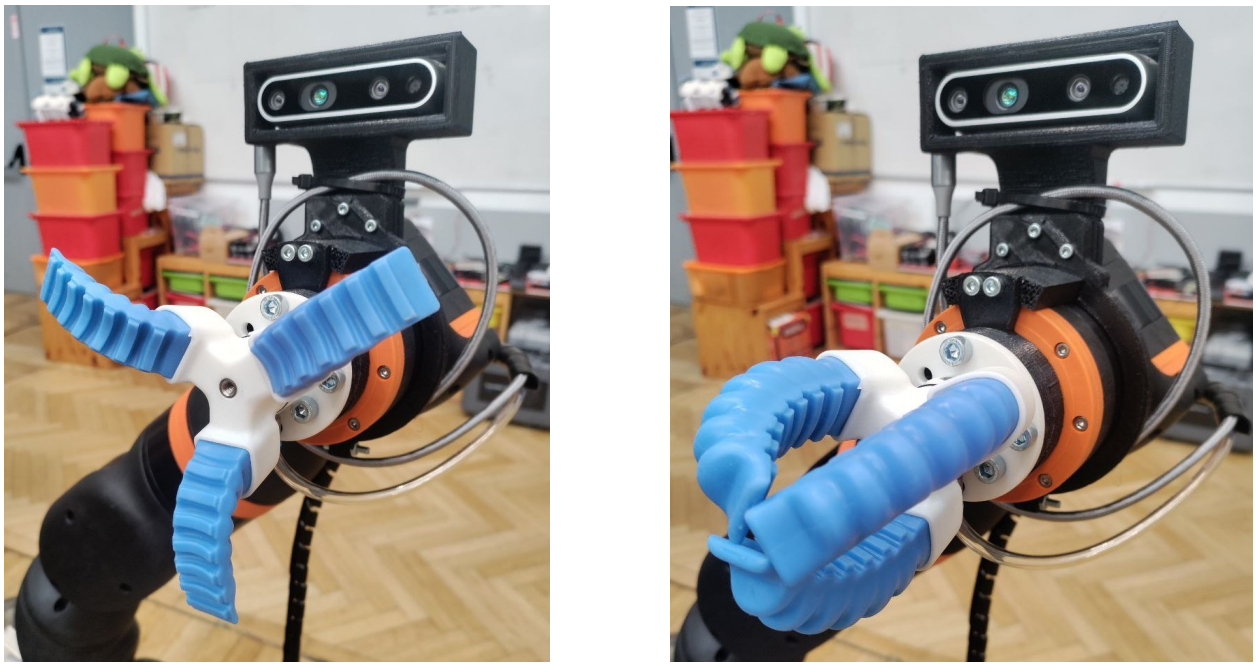
The pneumatic pump is provided without any power supply, so it was necessary to create a system to power the pump using the **onboard batteries**. The pump is powered by a 24V lead battery, which is the same battery powering the robotic arm. I created a system for powering both the robotic arm and the pneumatic pump with the same battery, using Molex cables and connectors. These Molex cables proved to be a reliable and optimal solution, as they allowed me to switch easily between the onboard batteries and the external cobot power supply. In fact, the cobot's power supply provides 24V at

Figure 1.9: Pneumatic Pump control box secured on top of the mobile robot

.

a maximum of 10A, which is enough to power the robotic arm and the pneumatic pump simultaneously, since the pneumatic pump doesn't require a high current to operate. The cable of the external power supply is also a Molex cable, so that's why I used Molex connectors and cables to connect the robotic arm and the pneumatic pump to the cobot's power supply.

The pneumatic pump must be controlled at 24V, as the pump's solenoid valve requires this voltage to operate. The pump provides 4 digital pins for controlling its operation, which are used to open and close the fingers. I created a simple system capable of controlling the pump's operation using an **Arduino UNO microcontroller**. The Arduino UNO is connected to the robot's computer via USB, allowing the computer to send commands to the Arduino via the serial port. The Arduino UNO is then connected to the pneumatic pump via a relay module, composed of 4 different relays, each controlling a different digital pin of the pump. The relays were necessary to provide an output voltage of 24V, while the Arduino UNO provides only 5V via its digital pinout. The relays are cheap and quick enough to switch between the digital pins of the pump, allowing an efficient control of the pump's operation, and the installation of a simple circuitry for the control system directly on the mobile robot platform.



(a) Soft Gripper opened

(b) Soft Gripper closed

Figure 1.10: Soft Gripper mounted on the end effector

1.5. 3D Printed Mounts Design

Mounting the sensors and electronic devices on the mobile robot platform required the design and 3D printing of custom mounts. The mounts were designed using the *Fusion 360* CAD software, which allowed me to create precise and accurate models of the mounts. The mounts were then 3D printed using the laboratory's 3D printer, which is a *Creality CR-10S* 3D printer. I designed and printed two versions of the mount for the cobot's flange:

- *Mount V1*: a mount for the Realsense camera and a digital button on top of a cylinder used for pressing buttons on a control panel. The mount was designed to be robust and easy to switch with other mount extensions. This mount was used for the first demos and tests of the project.
- *Mount V2*: a mount for the Realsense camera and the soft gripper as an end-effector. This mount was designed to be compact, robust and more effective for the final versions of the project.

I also designed and created another **mount for the GPS antenna**, which was used for outdoor localization and navigation tasks. The GPS antenna mount was designed to be placed on top of the LiDAR sensor, ensuring that the antenna had a clear view of the sky

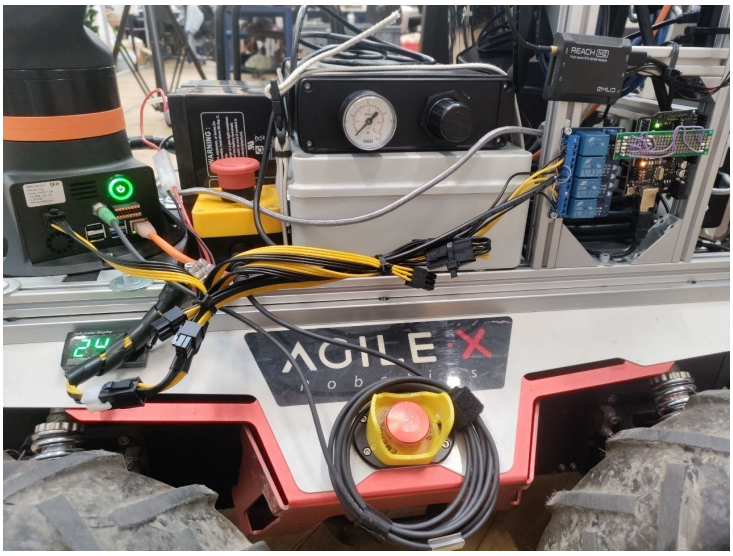


Figure 1.11: Molex connectors and power management for the cobot, pump, and relays
.

and the satellites. The mount was also designed to be lightweight and quick to install, without affecting the LiDAR sensor's field of view, and without using any screws or bolts. I never used the GPS in my project, but the mount that I created was used by other researchers in the laboratory for their projects.

The 3D printer that I used in the AIRLab is a Fused Deposition Modeling (FDM) printer, which uses a thermoplastic filament to create the 3D models layer by layer. The material of choice for the 3D prints was *PETG* (Polyethylene Terephthalate Glycol), which is a strong and durable material, suitable for mechanical parts and mounts. The PETG material is also resistant to mechanical stress and heat, making it the ideal material for the mounts for these kinds of applications.

The 3D-printed mounts were designed to be lightweight and compact, to ensure that the robot's mobility and stability were not affected. The mounts were also designed to be very robust and durable, to withstand the vibrations and shocks of the mobile robot platform. The mounts are not very easy to install and remove, since they are designed to be mounted with several screws and bolts, ensuring that the sensors and electronic devices are securely attached to the robot. These mounts demonstrated to be very effective and reliable, as they withstood the vibrations while maintaining the sensors in their fixed position.

1.5.1. Mount V1

The first version of the mount, alias *Mount V1*, was designed to be robust and easy to switch with other mount extensions. The mount was designed to be lightweight and long,

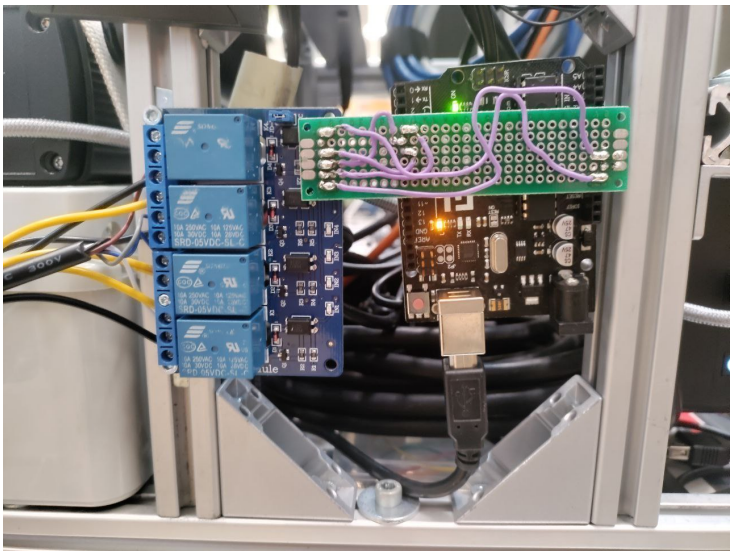


Figure 1.12: Arduino UNO microcontroller and relay module used to control the pneumatic pump

.

to ensure that the cobot's end effector would be able to reach objects not in the immediate vicinity of the robot. The mount was also designed to be compact, with the stereo camera mounted in front of the cobot's flange, and the digital button mounted on top of a cylinder used for pressing buttons on a control panel. The mount was also designed to be very robust and durable.

After many tests, the cylinder was then shrunk to a smaller length, to ensure that the cobot's end effector mobility and reach were not affected negatively. This allowed the cobot's end effector to reach objects in the vicinity with fewer limitations and constraints on its orientation. The digital button placed on the tip was initially used as a digital feedback mechanism for the pressure of the buttons on the control panel. This button was later removed, as it was not necessary for the project's objectives. The mount was then used for the first demos and tests of the project, and it proved to be very effective.

One of the main issues encountered with the mount was the **reduced field of view of the stereo camera**, due to its installation on the cobot's flange. The camera was placed at the right distance from the center of the cobot's flange, making it possible for the camera's field of view not to be obstructed by the cylinder. After many tests, I realized that the best configuration would be to mount the camera on top of the wrist, to enlarge the field of view. The Mount V1 was used for the "button presser demo", and replaced with its second improved version for the next demos.

Figure 1.13: GPS antenna 3d-printed mount on top of the LiDAR

.

1.5.2. Mount V2

The second version of the mount, alias *Mount V2*, was designed to be compact, robust, and more effective for the final versions of the project and the "soft grasping" demos. The mount comprises 3 parts: the flange connector, the camera mount, and the soft gripper mount. The flange connector is used to connect the mount to the cobot's flange, ensuring that the mount is securely attached to the cobot. The camera mount is used to mount the stereo camera on top of the cobot's wrist, ensuring that the camera has a wider field of view. The soft gripper mount is used to mount the soft gripper on the cobot's flange connector, allowing the cobot's end effector to grasp and manipulate objects in the environment.

The **flange connector** was designed from scratch because the flange provided by the cobot's manufacturer was not compatible with the soft gripper mount. The flange connector was designed to be lightweight and compact, to ensure that the 3d printing process would be fast and structurally sound. The flange connector was also designed to be robust and durable, to withstand the vibrations and shocks of the cobot's movements. The

Figure 1.14: 3d-printed mountV2 on the arm's wrist

.

camera mount is designed to be attached to the flange connector on the cobot's wrist with screws and bolts, ensuring that the camera is securely attached to the cobot while preventing vibrations that could offset the sensor position and orientation. The camera mount is designed to host the camera cable angled connector. This angled connector is magnetic, and it prevents the cable from being pulled out of the camera when the cobot's end effector moves around. This was a critical feature, set to avoid the camera's cable being broken or damaging the internal USB-C port of the stereo camera.

1.5.3. Issues with the laboratory's 3D printer

The 3D printer in the AIRLab is a printer that is used by many researchers and students for their projects. In the past, very little maintenance was done on the printer, and the printer was not calibrated correctly. This caused many issues with the 3D prints, such as **stringing, and under-extrusion**, which affected negatively the quality of my prints. Therefore I had to carry out maintenance on the printer to fix these issues. I calibrated the printer's bed and the extruder, to ensure that the prints were of high quality and accurate. I substituted the nozzle with a new one, and unclogged the hotend, to ensure that the filament was extruded correctly and in the right amount. I also cleaned the printer's bed and the extruder, to ensure that the prints were sticking correctly to the bed. After these maintenance operations, the printer was working correctly, and the prints were of higher quality, even though they were still far from being perfect. I was not able to clean the printer's extruder gears and the filament feeder, as they were not accessible to me due to some screws being stripped.

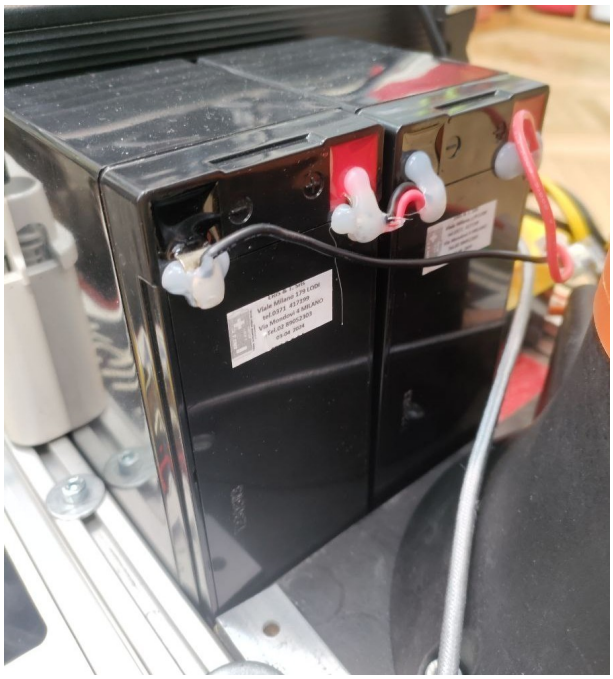


Figure 1.15: Lead batteries mounted onboard for the cobot and pneumatic pump
.

1.6. Batteries and Power Management

The mobile robot's internal battery is capable of powering all the sensors and computational units that are mounted. To meet the specific voltage and current requirements of these devices, two DC/DC converters are employed: - one DC/DC converter with an output voltage of 12V at a maximum of 15A for the on-board computer, router, and switch - one DC/DC converter with an output voltage of 24V at a maximum of 5A for the LiDAR sensor.

The mobile robot platform's internal batteries are not sufficient to power the robotic arm and the pneumatic pump mounted on board. To power these devices, an external power supply is used, which provides 24V at a maximum of 10A. The cobot is powered by two 12V **lead batteries** with 9Ah of power capacity, which provide the necessary power to the robotic arm motors and the pneumatic pump. The batteries are mounted and secured on the robot's base, ensuring that the robot is powered and operational during its missions. There are 2 batteries available in the laboratory, which can be switched easily when one of them is discharged.

Another tool used to power the robots is a *Santino*, a holy figure card presenting *Saint Staianet*, the patron saint and protector of the mobile manipulation robot. This figure, placed in the front, helped and protected the robot during the development and testing

Figure 1.16: DC/DC converter used to power the robot's sensors and computer

.

of the project, ensuring that everything was working fine and smoothly. The Santino was also used to provide a spiritual and religious touch to the project, ensuring that the robot was blessed and protected during its missions.

1.7. Complete Mobile Manipulation Setup

Figures 1.19 and 1.18 show the complete mobile manipulation setup, with the robotic arm mounted on top of the SCOUT 2.0 robot, and the soft gripper mounted on the robotic arm's end effector. The setup is ready for the "soft grasping" demos, where the robot is tasked with grasping and manipulating objects in simulated agricultural environments.

Figure 1.17: Santino

.

Figure 1.18: Front view of the robots

.

Figure 1.19: Lateral view of the robots

.