DHBW
Duale Hochschule
Baden-Württemberg
Ravensburg
Campus Friedrichshafen

Intralogistics with the autonomous transport robot

T3_3100/3200

Course of study: Electrical engineering

Field of study: Automation

Duale Hochschule Baden-Württemberg Ravensburg, Campus Friedrichshafen

from

Alexander Ostermann & Felix Wirl

Submission date:	January 11, 2025
Processing period:	01.10.2024 - 13.01.2025
Student number:	5047110, 4800628
Course:	TEA22
Reviewer:	Pranav Tej Gangavarapu

Declaration

according to 1.1.14 appendix 1 to §§ 3, 4 and 5 of the study and examination regulations for bachelor's degree programs in the field of technology at the Dualen Hochschule Baden-Württemberg from 29.09.2017 in the version from 24.07.2023.

I hereby certify that i have written my T3_3100/3200 on the topic:

Intralogistics with the autonomous transport robot

independently and did not use any sources or resources other than those specified. I also certify that the submitted electronic version corresponds to the printed version.

Friedrichshafen, January 11, 2025



Alexander Ostermann & Felix Wirl

Abstract

The improvement to the Husky robot is intended to make it easier and faster to work with the robot. The first part of the work is about optimizing the Husky. Here, the given sensors are connected in a compliant manner at both hardware and software level. This also includes implementation in the robot operating system. It is important to pay attention to the linking of the coordinate transformations. Furthermore, the network construct of the Husky is optimized by placing the different subnets in a single subnet. Ultimately, the optimization serves as preparation for autonomous product transportation between A and B.

Kurzfassung

Die Verbesserung des Husky-Roboters soll dafür sorgen, einfacher und schneller mit dem Roboter arbeiten zu können. Im ersten Teil der Arbeit geht es um die Optimierung des Huskys. Hierbei werden die gegebenen Sensoren konform angeschlossen, sowohl auf Hardware- als auch auf Softwareebene. Dies schließt auch die Implementierung in das Robot operating system mit ein. Dabei ist es wichtig, auf die Verlinkung der Koordinatentransformationen zu achten. Des Weiteren wird das Netzwerkkonstrukt des Huskys optimiert, indem die unterschiedlichen Subnetze in ein einziges Subnetz gelegt wurden. Letztendlich dient die Optimierung als Vorbereitung für den autonomen Bauteiltransport von A nach B.

Contents

1 Introduction **1**

2 Theoretical background **3**

2.1 RITZ - Regionales Innovations- und Technologietransfer Zentrum GmbH 3

2.2 Zukunftsfabrik - Bodensee 4

2.3 Linux - Ubuntu 20.04 5

2.4 Robot operating System (ROS) 6

2.5 Light detection and ranging (LIDAR) 7

2.6 Networking 8

2.7 The Husky 10

2.8 Communication Husky - LIDAR 11

2.9 Transformation tree Husky 12

3 Problem definition **15**

3.1 Actual state 15

3.2 Problems 15

3.3 Possible solutions 16

4 Implementation **17**

4.1 Networking 17

4.2 Coordinate transformation 19

5 Summary **21**

5.1 Problems during Implementation 21

5.2 Future actions 21

Bibilography **22**

List of Figures **24**

Example code **25**

List of abbreviations 26

List of Tables 27

Use of Artificial Intelligence-based Tools 28

1 Introduction

The use of robots is becoming increasingly popular these days. The reason for this is the opportunity to increase competitiveness in business and society. This can be illustrated by the following figure:

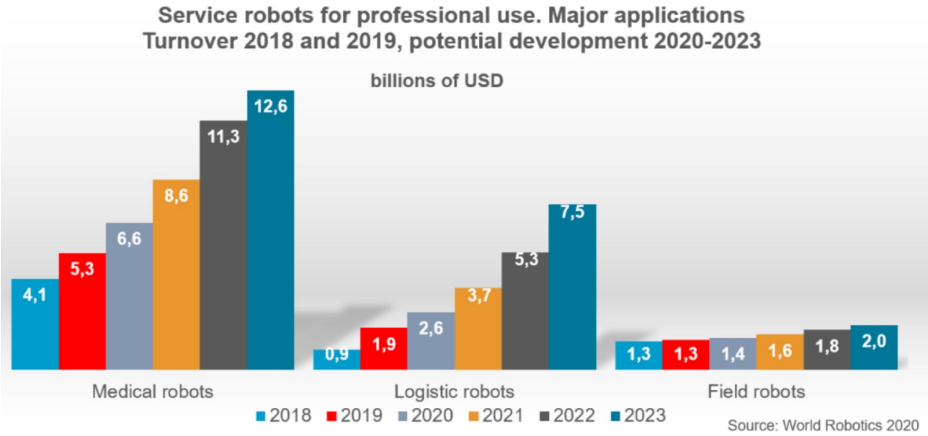


Figure 1.1: Service robots for professional use (cf. Schatton, 2024, p. 187)

Each robot is unique thanks to its sensors and actuators. For example, the sensors provide the necessary information about the system's environment and the actuators ultimately carry out the actions. This allows work steps to be carried out at a higher speed with consistent quality without the system becoming fatigued. The initial high acquisition costs are compensated for in the long term by the work performance of the robots, which means that human labour is increasingly being taken over by robots. Robots are therefore cheaper than human labour in the long term. (cf. Schatton, 2024)

The word robot comes from the Czech word "robota", which means forced labor, or from the late middle high German "robater", which means worker in forced labor. The term has its origins in science fiction stories. It was first used by the Czech writer Karel Capek in his play "R.U.R." (Rossumovi Univerzalni Roboti) in 1920. It

describes how artificial human-like automatons are created to take over human labor, but revolt against it. (cf. Dillmann and Huck, 1991)

Most robots used in industry are manipulators that are used at a fixed workstation and are not mobile. These industrial robots only work in a structured, controlled environment.

Mobile robots are fundamentally different and characterized by the following features:

- Adaptation to changes in its environment
- Learning from experience and corresponding behavioral change
- Development of an internal world view
- Independent planning and execution of complex tasks in an unknown environment (navigation planning)
- Manipulation of physical objects in the real world

The field of robotics has links to various disciplines, including computer science, electrical- (sensor technology) as well as mechanical engineering (kinematics and mechanical design), mathematics (statistical evaluation and error calculation), AI (artificial intelligence) and cognitive sciences. Mobile transport robots also include the Husky robot from Clearpath Robotics (cf. "Digitalisierung der Baustelle: Clever bauen mit mobilem Roboter" 2020).

This robot is at the center of the work, focusing on the connectivity of the unit, for example, how a simple connection between user and robot can be created without many individual steps for registration. The aim of this work is to automatically register the robot in order to simplify the start up process after the start button is pressed.

2 Theoretical background

2.1 RITZ - Regionales Innovations- und Technologietransfer Zentrum GmbH

The Regional Innovation and Technology Transfer Center (RITZ) is a competence center for digitalization and mobility in Friedrichshafen. It provides the innovation structure and network in which business and science work together on the future of digitalization and mobility. This bundling of knowledge combined with the active networking of the various players increases the innovative strength. The main goal of the RITZ is to create a vibrant innovation community. The regional proximity to internationally well-known companies such as ZF Friedrichshafen or Rolls-Royce Power Systems, as well as the close proximity to Zeppelin University and the Baden-Württemberg Cooperative State University, provides the RITZ with an innovative and knowledge-promoting environment. In summary, the RITZ is a platform for companies, research and educational institutions to work together on forward-looking projects in the field of digitalization and mobility. With a modern infrastructure and a broad network, it offers comprehensive support for innovative projects and future-oriented cooperation in the region. (cf. Technologietransfer Zentrum GmbH, 2025)



Figure 2.1: Illustrative representation of RITZ (cf. *Ein Leuchtturm mit überregionaler Strahlkraft* 2024; *ZPD Zentrum für Digitalisierung in Produktion und Produktentwicklung* 2023)

2.2 Zukunftsfabrik - Bodensee

The Zukunftsfabrik Bodensee on the Fallenbrunnen campus in Friedrichshafen is a key research location for the Center for Digitalization in Production and Product Development (ZDP). Here, different research projects work together to create new ideas and make progress. The core of the factory is a conveyor system with sensors that deliver the product right on time to the four designated stations. (cf. *ZPD Zentrum für Digitalisierung in Produktion und Produktentwicklung* 2023)

To start off, our system gets the particular product version through a QR reader. A collaborative robot retrieves the product from a designated spot and places it on a workpiece carrier that runs through the conveyor system. At the second station, the right bolts for each product are loaded onto the workpiece carrier using bin picking (automatic parts removal from a container). The third station is an ergonomically optimized workstation for manual work. This is where the final assembly of the pre-product takes place. These listed work steps take place in the Produktions Technologie

Zentrum (PTZ). A mobile robot transports the product to the RITZ, where the final assembly is carried out by an automated screwing station. (cf. *ZPD Zentrum für Digitalisierung in Produktion und Produktentwicklung* 2023)

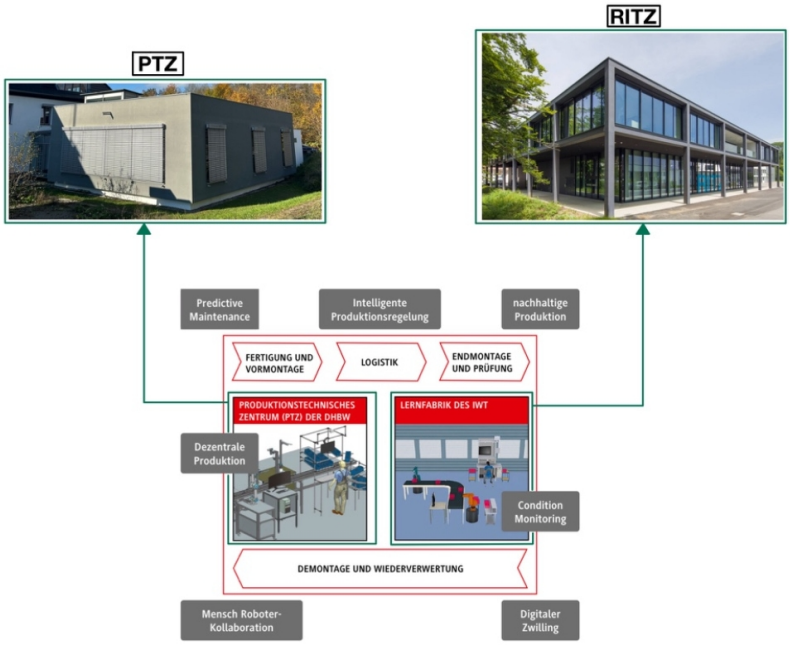


Figure 2.2: Illustrative representation of Zukunftsfabrik Bodensee (cf. *Ein Leuchtturm mit überregionaler Strahlkraft* 2024; *ZPD Zentrum für Digitalisierung in Produktion und Produktentwicklung* 2023)

2.3 Linux - Ubuntu 20.04

Linux for itself is an open-source operating system (OS) based on the Linux kernel. There are many Linux distributions available that uses different tools to suit the requirements of the given use case. Furthermore the installation and usage of Linux is completely free. (cf. *Linux* 2024)

As mentioned the Husky uses the Linux system with the Ubuntu distribution due to its pricing, simplicity regarding robotic development and its extensibility with open source software. (cf. Ubuntu, 2024)

2.4 Robot operating System (ROS)

ROS is an open source robot operating system. The whole idea is to let people develop new robotic applications without the hustle to do everything by there own. Therefore it is necessary to share all the code, algorithms, libraries as well as the ideas to develop highly complex robotic applications. There are many algorithms and drivers ready to use to simply create an autonomous robot like the Husky. ROS gathers data from connected sensors and uses a wrapper to publish data packaged in nodes. These nodes are mend to do just one thing reliable. For example there is just one node that is capable of visual object detection and one just for the algorithms. These nodes are published by the ROS master. The ROS master is also needed to find other nodes in the network. (cf. p. 419f Blubaugh et al., 2022)

In the following illustration (Figure 2.3) the construct of ROS is described.

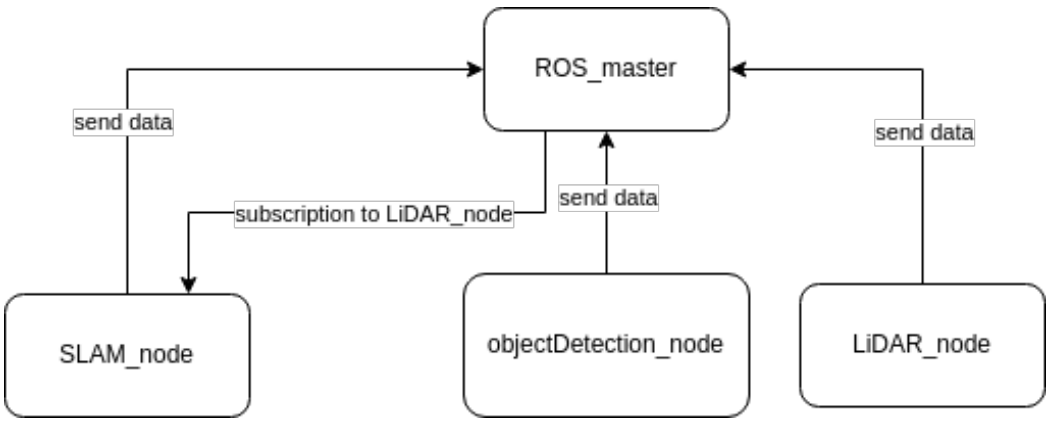


Figure 2.3: Illustrative representation of ROS

Furthermore the ROS system consists of executable packages that can either be developed by the user or can be downloaded as open source software. These packages are a collection of python source code, launch files, world models or robotic description files. To execute these, ROS uses the „ROS rosrun“ command. To find available packages or topics it is possible to query them with „ROS rostopic“ command. Then the ROS system will list all available topics. (cf. p. 426f. Blubaugh et al., 2022)

Based on given information ROS has the advantage to save time and money during development based on the open source infrastructure.

2.5 Light detection and ranging (LIDAR)

A LIDAR is a measurement device that is capable of measure the distance, reflection and scattering of objects by sending electromagnetic (EM) waves at every direction. It is possible due to the reflection and scattering characteristics of objects which will be hit by the EM waves. Furthermore LIDAR technology uses the infrared as well as ultraviolet EM waves to detect the objects. In the case of automotive driving the measurement of the distance is commonly used to get a three dimensional cloud of dots. Each dot has the information about its signal strength which can be used in combination with enough points to mimic the shape of an object. With the help of these data it is possible to label the objects and create a part of a 3D map with different categories. (cf. Beuth, Parl, and Gotzig, 2024)

The Husky has two different LIDAR sensors to ensure 360 degree detection of objects. At the front top there is a three dimensional (3D) LIDAR from Velodyne. One 3D LIDAR would be enough but it is not possible to cover the complete surrounding of the Husky caused by its physical structure. In other words the construct of the robot would block the LIDAR measurement system in front of itself. Therefore a second LIDAR system is installed at the opposite site as well as at a different hight as the 3D LIDAR to ensure 360 degree object detection. Furthermore the second LIDAR is a two dimensional (2D) measuring system from Hokuyo. A 2D image is generated by moving or rotating measurement beam at one level. Thereby it is possible to calculate the angle and distance to objects (cf. left LIDAR Figure 2.4). On the other hand it is possible to generate a 3D image by moving and sensing more than one measurement beam in order to calculate the x- y- and z-axis (cf. right LIDAR Figure 2.4). On the illustration Figure 2.4 the differences of both system functionalities are briefly described.

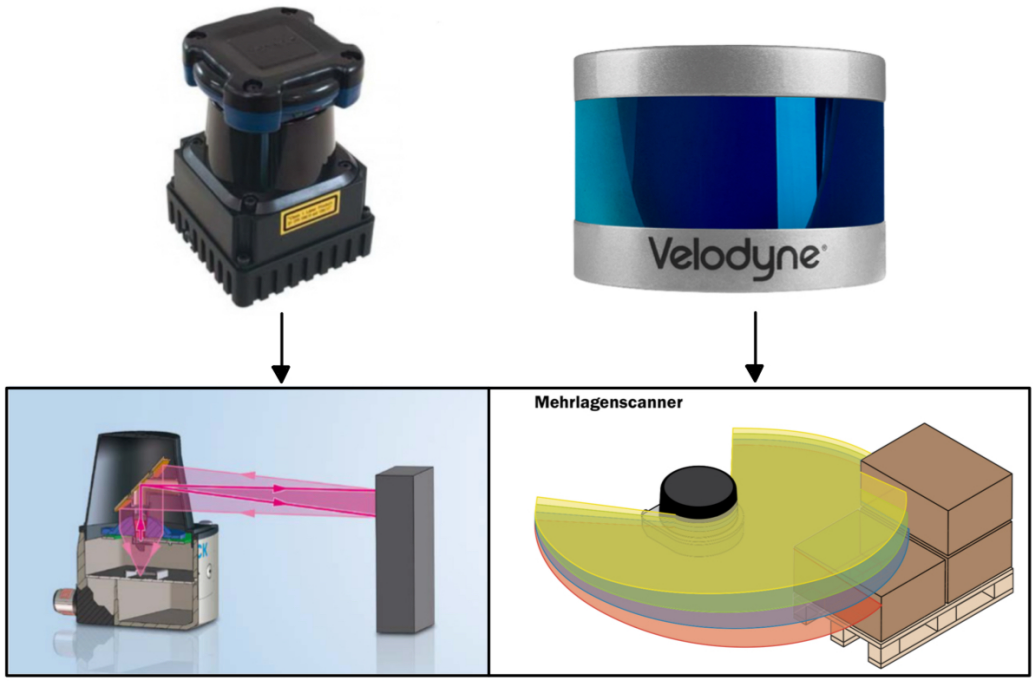


Figure 2.4: Illustrative representation of LIDAR systems (cf. elektronik, 2024; Hexagon, 2024; SICK, 2018)

2.6 Networking

Due to the Husky pc has two LIDAR sensors that are connected via ethernet to the pc it is necessary to configure the networking right in order to get the data from the sensors. Therefore the sensors as well as the ports of the pc must configured with a static ipv4 (internet protocol version four) address. Based on the number of devices the use of only ipv4 addresses is more than sufficient. In the matter of fact the ipv4 address has a length of 32 bit as outlined below. Therefore it is possible to connect nearly 2^{32} devices together.



Figure 2.5: Husky PC

A ipv4 address is a tuple with the length of 32 bit which contains the number of the network as well as the number of the host. Each number is written in decimal system between 0 and 255. The numbers are separated by a dot like „192.168.131.1“. Thereby many hosts are able to have the same network number but only one host number in one network is allowed. (cf. p. 43 Mandl, 2018)

It is possible to separate one network in multiple subnets to stop the waste of ip addresses in one network. To configure multiple subnets the network- and host number is split into subnet- and host number. To define a subnet in a network a subnet mask is needed to restrict the subnet into preferred range. The subnet mask is similar to the ipv4 address, four blocks of decimal numbers between 0 and 255. In order to get the range of the subnets a bitwise AND operation is needed. As a result of the operation the third block of the ip address describes the actual subnet. Because the Husky has the ip address of „192.168.131.1“ the subnet is „131“. (cf. p. 48f. Mandl, 2018)

From the information above it is more easy to configure all devices into the same subnet rather than working with multiple subnets.

2.7 The Husky

The Husky A200 transport robot from Clearpath Robotics is a development platform for demanding outdoor applications. It is characterized by modern and simple design, as well as a robust construction. Thanks to autonomous programming, this platform is used to autonomously perceive environments and recognize obstacles to plan routes and explore unauthorized areas. The Husky's mobility is ensured by the four wheels, whereby it can carry a load capacity of 20-75 kg and an incline of 45°, it can reach a speed of up to 1 m/s. The precise detection of the unknown environment is given through a multitude of sensors such as LIDAR and GPS. (cf. MYBOTSHOP, 2025)

The following features demonstrate the uniqueness of the robot:

- Compact ROS development platform
- Modularly expandable
- Optimized for outdoor applications

The optimization and expandability of the transport robot is made possible by the company MYBOTSHOP. The system has already been equipped with sensors such as 2D LIDAR from HOKUYO and 3D LIDAR from Velodyne to provide highly accurate information about the surroundings. This equipment provides a good prerequisites for navigation and mapping applications. By extending the transport robot with a frame, the total height is 0,74 m. The orange holders on the woden plate of the frame allow components to be placed and removed. The extended version of the Husky can be seen in the following illustration. (cf. MYBOTSHOP, 2025)

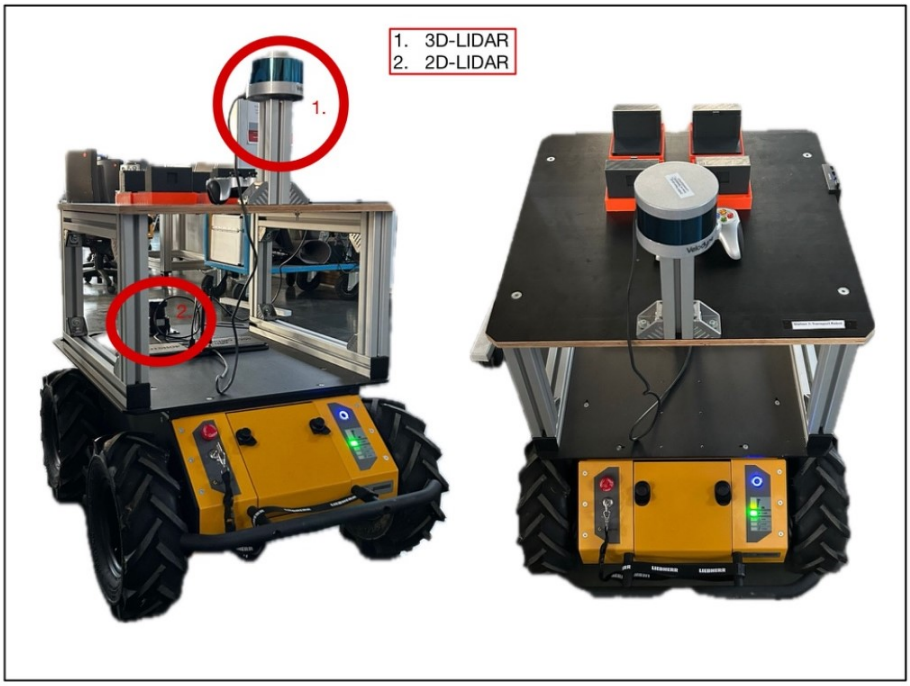


Figure 2.6: Representation of the Husky

2.8 Communication Husky - LIDAR

Both LIDAR sensors use the IP protocol to send their data to the PC. The Internet Protocol is what represents the network layer in communication technology. This protocol is responsible for translating the addresses into the IP addresses used by the network layer. This is important because devices only know each other's IP addresses, not their unique Media Access Control (MAC) addresses, which each network access receives. (cf. p. 28 Mandl, 2018)

The IP address is linked to the MAC address using the Address Resolution Protocol (ARP). When a host sends a request to a destination host with which it has not communicated for a period of time, an ARP request is sent over the LAN (Local Area Network). The ARP request basically asks everyone on the network if they know the IP address in the request. Other devices on the network that know the IP address

will respond with an ARP reply, passing the MAC address of the destination device. Alternatively, the destination device can send its MAC address itself. The fact that the MAC addresses are stored in an ARP cache for a period of time, eliminates the need to retrieve these again for each communication. (cf. p. 112 Mandl, 2018)

Once the IP and MAC addresses have been assigned and the devices are connected, a self-contained package can be sent over the network using Internet Protocol version 4 (IPv4). If the data is too large, the packages must be broken up and sent one at a time. Furthermore the IPv4 protocol does not guarantee package delivery, so data may be lost. (cf. p. 41 f. Mandl, 2018)

2.9 Transformation tree Husky

In order to locate objects as well as the Husky itself, everything needs a coordinate system. When humans want to locate a tree in front of them, they say „the tree is 50m in front of me“. They tell the axis by giving the direction and the distance by telling how far away the tree is in that particular direction. If another person is 10m to the right of the first person, the first person can only tell how far away the tree is from the second person if he knows the distance between them. In analogy to the robot, it is almost the same. The center of the robot is at its center and has the coordinates $\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$. To get the distance from the 3D LIDAR mounted on the front top of the Husky to the base or even at the front of the robot, it is necessary to tell the robot the transformation between the center point and the center of the LIDAR sensor.

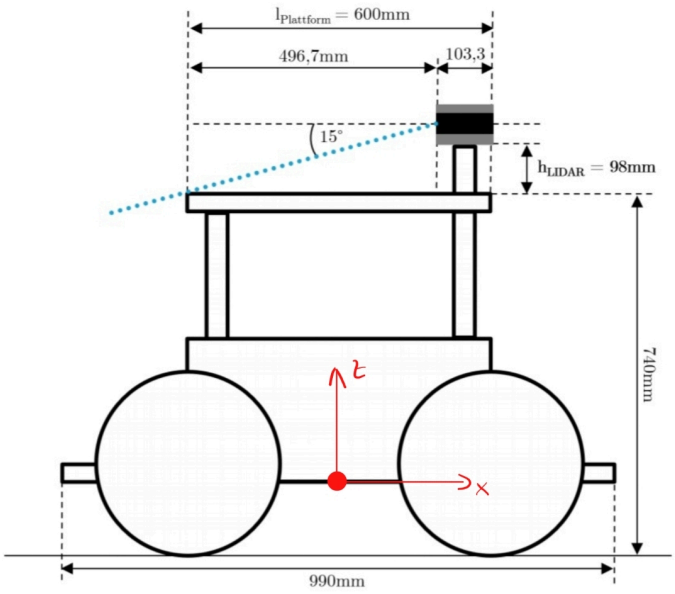


Figure 2.7: Illustrative representation of Husky (cf. Schmid and Straßburger, 2024)

From the Figure 2.7, the distance from the center point to the center of the LIDAR in x-direction can be calculated as „248.35“. Therefore the distance in z-direction would be „910.7“. The y-direction does not change, because the center of the LIDAR is at the center of the Husky in y-direction. Therefore the transformation matrix is $\begin{bmatrix} 248.35 \\ 0 \\ 910.7 \end{bmatrix}$ mm. (cf. Rieber, 2024, 4. Koordinatensysteme)

In general, the transformation tree in ROS looks like this:

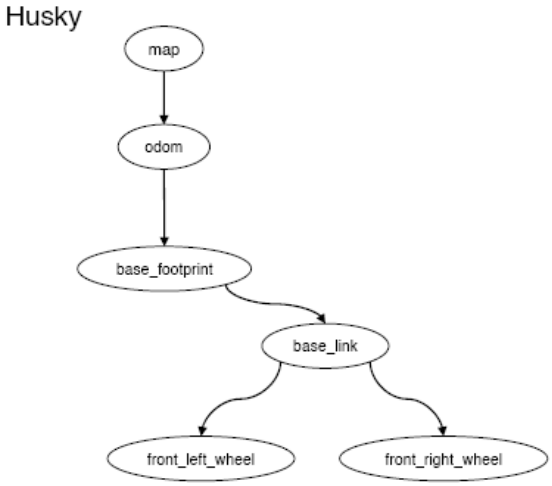


Figure 2.8: Part of Husky transformation tree

Each ellipse describes a frame and the arrows are the transformations between the frames.

Transforms are very important, because otherwise the calculation has to be done with trigonometry, which becomes too complex. Therefore the relationship between coordinate frames are maintained by the tf-package in ROS. (cf. Wiki, 2025)

3 Problem definition

3.1 Actual state

Before making any changes to the robot, the previous student's paper was read and understood. This is essential to understand the actual physical as well as the software construct of the Husky. Actually, the 2D laser is implemented in software and physical. On the other hand, the 3D LIDAR is only physically and accessible through some routing commands like the students described in their thesis earlier. In addition, the Velodyne LIDAR is on a different subnet, which should not be the case. Right now, the robot is only controllable through a Logitech controller, and even that only works after connecting to the Husky remotely and rebooting it.

3.2 Problems

If the Husky is turned on, it needs to be restarted from a Linux bash („sudo service mbs_Husky restart“). It is necessary to ssh (secure shell) into the Husky pc to execute the given command. Only then it is possible to use Husky properly. Furthermore, the different subnets are not ideally configured regarding to the number of devices. In addition, the 3D LIDAR is only accessible after some routing commands inside the secure shell. Therefore, the Husky is not able to drive itself due to missing data from the configured components. It is also necessary to create a description file for the LIDAR to link it to the robot. This is critical so that the robot's position in relation to the objects can be calculated correctly. Otherwise, safe autonomous driving is not guaranteed.

3.3 Possible solutions

To solve the first problem, it is necessary to go through the startup files to find the script that is causing the error. After that, the robot should be able to boot without errors, so that it is possible to just press the start button and use it properly.

Furthermore, the networking should be solved by correctly configuring the IP addresses of both Ethernet ports as well as the addresses of both LIDAR sensors. Both ports need static ipv4 addresses on the same subnet as the robot. The LIDAR sensors also require static ip addresses in the same subnet as the ports in order to communicate with them without further commands.

After creating a description file, the physical distance between the center of the robot and the center of the 3D LIDAR must be calculated correctly. These calculated coordinates are added to the Husky transformation tree to link the base coordinate system to the LIDAR coordinate system.

4 Implementation

4.1 Networking

To set static IP addresses, changes can be made either graphically through the user interface or by editing the robot's netplan file. Editing the netplan file ensures that these network settings are applied at startup. It is therefore necessary to create a netplan file to manage the desired connections. To edit the netplan file, it must be opened with the following command:

```
sudo nano /etc/netplan/netconfig_husky.yaml
```

This command creates (if not existing) or opens the netplan file in a text editor. Now it is important to know the syntax of yaml in order to write a new netplan command. To simplify the editing, a code editor like „Visual Studio Code“ is more suitable. The command to open the file would then be:

```
code /etc/netplan/netconfig_husky.yaml
```

According to (ubuntuusers, 2024) it is necessary to give the netplan program the correct header as shown in Yaml code 4.1.

Yaml code 4.1: Netplan configuration file

```
1 network:
2   version: 2
3   ethernets:
4     eno1: # Velodyne 3D LIDAR
5       dhcp4: no
6       addresses:
7         - 192.168.131.30/24  # Static IP address for eno1 on your PC
8     en1sp: # Hokuyo 2D LIDAR
9       dhcp4: no
10      addresses:
11        - 192.168.131.20/24  # Static IP address for en1sp on your PC
12  routes:
13    - to: 192.168.131.32/24  # Velodyne IP address
14      via: 192.168.131.30
15    - to: 192.168.131.21/24  # Hokuyo IP address
16      via: 192.168.131.20
```

After the header (Yaml code 4.1, line 3) it is possible to define a custom netplan. The Husky has a default IP address of "192.168.131.1", which leads to the subnet of „131“ as said in section 2.6. Therefore, it was decided that all connected devices would be configured on the same subnet for simplicity. The 3D LIDAR was configured with an ip address of „192.168.131.31“ as described by the previous students in their paper (cf. Schmid and Straßburger, 2024). While the 2D LIDAR already had the ip address „192.168.131.21“, which is in the right subnet. Whereas the address of the 3D LIDAR had to be changed on the web interface to „192.168.131.32“. In order to get data from the sensors, the Ethernet ports of the PC must also be in the same subnet. So in Yaml code 4.1 the lines from 4 to 16 have to be added.

At first, the file sets static ip addresses for both ports (eno1 and en1sp), so the automatic ip address assignment (DHCP) must be turned off (line 5). Then it is possible to assign a static IP address to the port by adding the lines 6f. To establish the connection, it is necessary to add the routing commands from the devices to the ports, as shown in Yaml code 4.1. from the line 12 to the line 16.

After saving the file it is crucial to apply the netplan with the follwoing command:

```
sudo netplan apply
```

Otherwise the netplan configuration is not applied correctly.

4.2 Coordinate transformation

To ensure that the 3D LIDAR is implemented at startup, a new launch file named „custom_launch_velodyne“ was created in the same directory as the package „velodyne_pointcloud“. This file contains the calculated transformation of section 2.9 as well as a link to start the velodyne driver.

```
<launch>
    <node pkg="tf" type="static_transform_publisher" name="Velodyne_LIDAR"
    args="0.24835 0 0.9107 0 0 0 base_link Velodyne_LIDAR 90"/>

    <include file="$(find velodyne_pointcloud)/launch/VLP16_points.launch"/>
</launch>
```

It is necessary to start the Velodyne driver to create a point cloud. Therefore it was implemented in the „custom_launch_velodyne“ file.

Now it is possible to start the 3D LIDAR with the custom coordinate transformation as well as the velodyne point cloud driver with following command:

```
roslaunch velodyne_pointcloud custom_launch_velodyne.launch
```

Only then the detected points of the LIDAR are visible in rviz (cf. Figure 4.1).

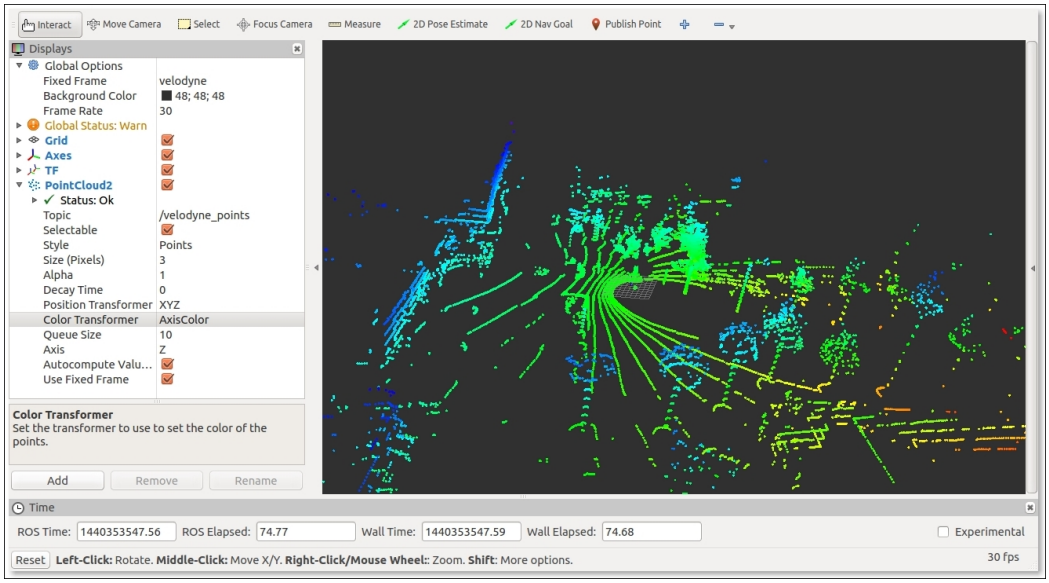


Figure 4.1: Point cloud of 3D LIDAR in RVIZ (cf. Joseph, 2017)

5 Summary

5.1 Problems during Implementation

Unfortunately, not all problems could be solved. The network problem took the most time, which meant that we could not work on the other sites. Most of the time, only one of the two LIDARs was able to send data, even though they were on the same subnet. The problems were the lack of knowledge about networks on the one hand and the insecurities with Linux on the other hand. However, the infrastructure of the Husky was adapted to ensure better utilization. Furthermore, the 3D LIDAR was linked to the base point of the robot using coordinate transformation to ensure efficient and easy calculation of the detected points. As a result, the robot is perfectly set up for the simulation part.

5.2 Future actions

In the next phase, the Husky's abilities will be transferred to a simulation. Autonomous driving can then be efficiently tested in this simulation. This will also make it easier to adjust parameters and optimize the navigation algorithm. The goal is to find out how the robot can drive autonomously from the PTZ to the RITZ. If the simulations are successful, the simulation will be tested in the real world.

Furthermore the communication between the Husky and the PTZ as well as the RITZ must be implemented. The Husky must know when it is allowed to start driving from PTZ to RITZ or vice versa. Hereby the PLC from the ZPD should be the superior system which issues the release.

Bibliography

Beuth, Thorsten, Christoph Parl, and Heinrich Gotzig (2024). "LiDAR". In: *Handbuch Assistiertes und Automatisiertes Fahren: Grundlagen, Komponenten und Systeme für assistiertes und automatisiertes Fahren*. Ed. by Hermann Winner et al. Wiesbaden: Springer Fachmedien Wiesbaden, pp. 357–377. ISBN: 978-3-658-38486-9. DOI: 10.1007/978-3-658-38486-9_16. URL: https://doi.org/10.1007/978-3-658-38486-9_16.

Blubaugh, David Allen et al. (2022). "Noetic ROS Further Examined and Explained". In: *Intelligent Autonomous Drones with Cognitive Deep Learning: Build AI-Enabled Land Drones with the Raspberry Pi 4*. Berkeley, CA: Apress, pp. 417–436. ISBN: 978-1-4842-6803-2. DOI: 10.1007/978-1-4842-6803-2_12. URL: https://doi.org/10.1007/978-1-4842-6803-2_12.

"Digitalisierung der Baustelle: Clever bauen mit mobilem Roboter" (June 2, 2020). In: *Forschung Kompakt*. URL: https://copilot.cloud.microsoft/?fromcode=bingchat&redirectid=F885CAA0A08D45FDB8D8AB2F96D9DB64&auth=2 (visited on 01/07/2025).

Dillmann, Rüdiger and Martin Huck (1991). *Informationsverarbeitung in der Robotik*. 1st ed. Springer-Lehrbuch. Springer-Verlag Berlin Heidelberg, pp. XI, 337. ISBN: 978-3-540-53036-7. DOI: https://doi.org/10.1007/978-3-642-50188-3.

Ein Leuchtturm mit überregionaler Strahlkraft (2024). Online. URL: https://www.ir-bauen.com/de/case-studies/ritz-friedrichshafen (visited on 11/29/2024).

elektronik, reichelt (2024). *UTM-30LX-EW HOKUYO UTM-30LX-EW LiDAR*. Online. URL: https://www.reichelt.de/de/de/hokuyo-utm-30lx-ew-lidar-utm-30lx-ew-p320261.html?PROVID=2788&gad_source=1&gbraid=0AAAADwnxtav2lqcyuIaaolot_Cm86R5n&gclid=CjwKCAiAxqC6BhBcEiwAlXp45xyyTOlAYlS1txuoMGBA1pw1q5e3l9IuYkuy7umi_wm3BfzzUVRPwRoCmeQQAvD_BwE&&r=1 (visited on 11/29/2024).

Hexagon (2024). *Velodyne Puck*. Online. URL: https://autonomoustuff.com/products/velodyne-puck-vlp-16 (visited on 11/29/2024).

Joseph, Lentin (2017). *ROS Robotics Projects: Make your robots see, sense, and interact with cool and engaging projects with Robotic Operating System*. English. 1st. Packt Publishing, p. 452. ISBN: 9781783554713.

Linux (Nov. 23, 2024). Online. URL: https://en.wikipedia.org/wiki/Linux (visited on 11/24/2024).

Mandl, Peter (Oct. 25, 2018). *Internet Internals. Vermittlungsschicht, Aufbau und Protokolle*. Springer Vieweg Wiesbaden. ISBN: 978-3-658-23535-2. DOI: https://doi.org/10.1007/978-3-658-23536-9.

MYBOTSHOP (2025). *Clearpath Husky A200*. Online. URL: https://www.mybotshop.de/Clearpath-Husky-A200 (visited on 01/03/2025).

Rieber, Prof. Dr.-Ing Jochen M. (2024). *Robotik. Vorlesung im Studium Elektrotechnik / Automation*. Script. (Visited on 11/29/2024).

Schatton, Dipl. Ing. Heike (2024). *Automatisierung*. Script. (Visited on 12/22/2024).

Schmid, Daniel and Paul Straßburger (July 15, 2024). "Inbetriebnahme eines Transportroboters für eine neue Produktionslinie". Studienarbeit T3 3100 & T3 3200. Studienarbeit. Duale Hochschule Baden-Württemberg Ravensburg, Campus Friedrichshafen.

SICK (July 2018). *Funktionsweise und varianten von LiDAR-Sensoren*. Online. URL: https://cdn.sick.com/media/docs/5/25/425/whitepaper_lidar_de_im0079425.pdf (visited on 11/29/2024).

Technologietransfer Zentrum GmbH, RITZ Regionales Innovations- und (Jan. 7, 2025). *RITZ Innovationszentrum*. URL: https://ritz-innovationszentrum.com.

Ubuntu (2024). *Powering robotics in every industry*. Online. URL: https://ubuntu.com/robotics (visited on 11/26/2024).

ubuntuusers (Nov. 23, 2024). *Netplan*. Online. URL: https://wiki.ubuntuusers.de/Netplan/ (visited on 12/10/2024).

Wiki, ROS (2025). *tf - ROS Wiki*. URL: http://wiki.ros.org/tf#view_frames (visited on 01/09/2025).

ZPD Zentrum für Digitalisierung in Produktion und Produktentwicklung (Dec. 2023). Imagebroschuere. URL: https://www.ravensburg.dhbw.de/fileadmin/Ravensburg/Dokumente_Bilder_Contentbereich/FIT/DHBW_RV_FN_Imagebroschuere_ZDP.pdf (visited on 11/28/2024).

List of Figures

1.1 Service robots for professional use (cf. Schatton, 2024, p. 187) 1

2.1 Illustrative representation of RITZ (cf. *Ein Leuchtturm mit überregionaler Strahlkraft* 2024; *ZPD Zentrum für Digitalisierung in Produktion und Produktentwicklung* 2023) 4
2.2 Illustrative representation of Zukunftsfabrik Bodensee (cf. *Ein Leuchtturm mit überregionaler Strahlkraft* 2024; *ZPD Zentrum für Digitalisierung in Produktion und Produktentwicklung* 2023) 5
2.3 Illustrative representation of ROS 6
2.4 Illustrative representation of LIDAR systems (cf. elektronik, 2024; Hexagon, 2024; SICK, 2018) 8
2.5 Husky PC 9
2.6 Representation of the Husky 11
2.7 Illustrative representation of Husky (cf. Schmid and Straßburger, 2024) 13
2.8 Part of Husky transformation tree 14

4.1 Point cloud of 3D LIDAR in RVIZ (cf. Joseph, 2017) 20

Example code

4.1 Netplan configuration file 18

List of abbreviations

2D	two dimensional
3D	three dimensional
AI	Artificial Intelligence
EM	electromagnetic
etc.	et cetera
ip	internet protocol
LIDAR	Light detection and ranging
OS	Operating system
PTZ	Produktionstechnisches Zentrum
RITZ	Regionales Innovations- und Technologietransfer Zentrum GmbH
ROS	Robot Operating system
ZPD	Zentrum für Digitalisierung in Produktion und Produktentwicklung

List of Tables

1.1 List of artificial intelligence-based tools used 28

Use of Artificial Intelligence-based Tools

This study utilized artificial intelligence (AI) tools. The table 1.1 offers a comprehensive overview of the applied tools and their respective purposes.

Table 1.1: List of AI-based tools used

Tool	**Description of use**
DeepL	• Help with the translation of individual sections