DHBW
Duale Hochschule
Baden-Württemberg
Ravensburg
Campus Friedrichshafen

Inbetriebnahme eines Transportroboters für eine neue Produktionslinie

Studienarbeit T3_3100 & T3_3200

Studiengang Elektrotechnik

Studienrichtung Fahrzeugelektronik

Duale Hochschule Baden-Württemberg Ravensburg, Campus Friedrichshafen

von

Daniel Schmid und Paul Straßburger

Abgabedatum:	15. Juli 2024
Bearbeitungszeitraum:	02.10.2023 - 22.12.2023 & 01.04.2024 - 28.06.2024
Matrikelnummer:	7749394, 4834384
Kurs:	TFE21-2
Betreuer:	M.Sc. Peter Buckel

Erklärung

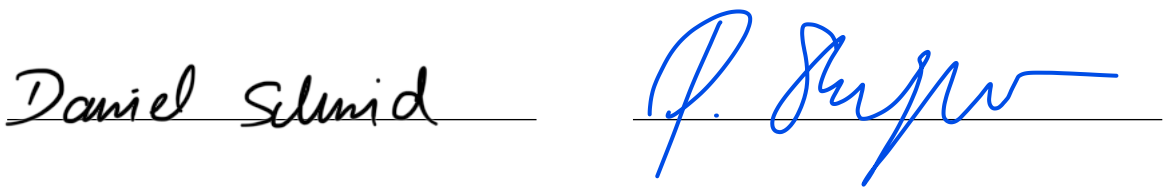
gemäß Ziffer 1.1.14 der Anlage 1 zu §§ 3, 4 und 5 der Studien- und Prüfungsordnung für die Bachelorstudiengänge im Studienbereich Technik der Dualen Hochschule Baden-Württemberg vom 29.09.2017 in der Fassung vom 24.07.2023.

Ich versichere hiermit, dass wir unsere Studienarbeit T3_3100 & T3_3200 mit dem Thema:

Inbetriebnahme eines Transportroboters für eine neue Produktionslinie

selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Wir versichern zudem, dass die eingereichte elektronische Fassung mit der gedruckten Fassung übereinstimmt.

Friedrichshafen, den 15. Juli 2024



Daniel Schmid und Paul Straßburger

Kurzfassung

Die vierte industrielle Revolution, auch Industrie 4.0 genannt, integriert viele moderne Aspekte in die Produktionstechnik und Fertigungsprozesse. Darunter zählen zum einen der Einsatz von Robotern und künstlicher Intelligenz, aber auch die Vernetzung und Datenanalyse über das Internet. Ziel dieser Arbeit ist die Inbetriebnahme eines Transportroboters, welcher eine solche moderne Produktionslinie über zwei Gebäude hinweg miteinander verbinden soll. Dieser soll mit Hilfe von LIDAR-Sensordaten und dem SLAM-Algorithmus eine detaillierte Karte seiner Umgebung erstellen und anhand dieser autonom die Werkstücke vom einen in das andere Gebäude transportieren. Der Fokus in dieser Studienarbeit liegt hauptsächlich auf der technischen Inbetriebnahme, also vor allem die Einrichtung einer Benutzeroberfläche, das Auslesen und Anzeigen von Sensordaten und die Inbetriebnahme und Positionierung eines neuen 3D-LIDAR Sensors. Schlussendlich konnte eine stationäre Arbeitsoberfläche eingerichtet werden, sowie eine kabellose Verbindung zwischen Transportroboter und einem separaten Laptop. Darüber können diverse Informationen ausgelesen und Befehle, wie zum Beispiel der Neustart des Roboters, ausgeführt werden. Des Weiteren konnten die Sensordaten des 2D- und 3D-LIDAR als Punktwolke in einem Visualisierungstool angezeigt werden. Zusätzlich wurde die Positionierung des 3D-LIDAR Sensors untersucht und evaluiert, um ein optimales Sichtfeld zu ermöglichen. Eine dafür vorgesehene Halterung wurde geplant, konstruiert und montiert. Des Weiteren wurde der 3D-LIDAR an das interne Stromnetz des Roboters angeschlossen. Um die Arbeit am Roboter unhabhägig von einem stationären Arbeitsplatz im Labor machen, wurden erste Ideen für ein mobiles Arbeitsplatzkonzept ausgearbeitet. Dafür wurde bereits ein USB-Hub angeschafft und montiert, sowie ein mobiler Monitor mit Halterung ausgewählt. Abschließend wurde ein ”Quick Start Guide” für die nachfolgende Gruppe erstellt, um dieser den Start in das Projekt zu vereinfachen.

Abstract

The fourth industrial revolution, also called Industry 4.0, integrates many modern aspects into production technology and manufacturing processes. These include the use of robots and artificial intelligence, but also networking and data analysis via the Internet. The aim of this work is to put a transport robot into operation, which will connect such a modern production line across two buildings. This will use LIDAR sensor data and the SLAM algorithm to create a detailed map of its surroundings and autonomously transport the workpieces from one building to the other. The focus of this study work is mainly on the technical commissioning, i.e. setting up a user interface, reading out and displaying sensor data and commissioning and positioning a new 3D-LIDAR. Ultimately, a stationary work surface could be set up, as well as a wireless connection between the transport robot and a separate laptop. Various information can be read out and commands, such as restarting the robot, can be executed. Furthermore, the sensor data from the 2D- and 3D-LIDAR could be displayed as a point cloud in a visualization tool. In addition, the positioning of the 3D-LIDAR sensor was examined and evaluated to enable an optimal field of view. A dedicated mount was planned, constructed and installed. Furthermore, the 3D-LIDAR was connected to the robot's internal power network. In order to make work on the robot independent of a stationary workplace in the laboratory, initial ideas for a mobile workplace concept were developed. A USB-hub has already been purchased and installed and a mobile monitor with a mount has been selected. Finally, a "Quick Start Guide" was created for the following group to make it easier for them to start the project.

Inhaltsverzeichnis

1. Einleitung **1**

2. Grundlagen **3**

2.1. Zukunftsfabrik Bodensee 3
2.2. Transportroboter Husky 5
2.3. Programmierung 8
2.3.1. Linux 8
2.3.2. Ubuntu 9
2.3.3. ROS 9
2.4. SLAM 10
2.5. LIDAR 13

3. Vorgehen **16**

4. Umsetzung und Ergebnisse **18**

4.1. Einrichten der Arbeitsoberfläche 18
4.2. RViz-Sensordaten (2D-LIDAR) 21
4.3. Integration 3D-LIDAR 24
4.3.1. Technische Inbetriebnahme 25
4.3.1.1. Physikalische Verbindung 25
4.3.1.2. Computer IP-Adresse konfigurieren 25
4.3.1.3. LIDAR mit Computer verbinden 26
4.3.1.4. Dateien und Treiber installieren 28
4.3.1.5. Daten in RVIZ anzeigen 29
4.3.2. Berechnung der Positionierung auf Roboter 31
4.3.3. Evaluierung der Halterungshöhe 37

4.3.4. Planung und Konstruktion der Halterung für den 3D-LIDAR . . 39
4.3.5. Elektrische Integration des LIDARs in das Robotersystem . . . 43
4.4. Planung und erste Umsetzungen der mobilen Arbeitsstation 47

5. Zusammenfassung **50**

6. Review der Studienarbeiten **52**

Literaturverzeichnis **54**

Abkürzungsverzeichnis **56**

Symbolverzeichnis **57**

Abbildungsverzeichnis **58**

A. Datenblätter **60**

B. Quick Start Guide **64**
B.1. Husky 101 64
B.2. Laden des Husky 66
B.3. Steuern des Roboters mithilfe des Controllers 66
B.4. Visualisierung der Sensordaten und Steuern des Roboters mittels GUI . 68
B.5. Support bei Softwarefragen 70
B.6. Nützliche Links 70
B.7. Next ToDos 71

1. Einleitung

Die industrielle Produktionstechnik hat sich im Laufe der letzten Jahrhunderte wesentlich weiterentwickelt. So wurde die erste industrielle Revolution Ende des 18. Jahrhunderts durch den Einsatz von Wasser- und Dampfkraft eingeleitet. Ende des 19. Jahrhunderts wurden erste Produkte in Massenfertigung hergestellt, was ein Merkmal der zweiten industriellen Revolution darstellt. Der Einsatz von Elektronik und IT in der Industrie gegen Mitte des 20. Jahrhunderts wird auch als dritte oder digitale Revolution bezeichnet. Mit der vermehrten Verwendung von Robotern, künstlicher Intelligenz, Datenanalyse und Verwendung des Internets (Internet of Things, IoT) wurde die vierte industrielle Revolution, auch Industrie 4.0 genannt, eingeleitet [1]. Da sich unsere heutige Gesellschaft bzw. Industrie genau in dieser Epoche aufhält, sind die Bemühungen groß Teil dieser Entwicklung zu sein, um weiterhin wirtschaftlich und modern zu bleiben. Um verschiedene Ansätze einer solchen Industrie 4.0 Produktionslinie zu untersuchen, wird im Moment in der Zukunftsfabrik Bodensee eine solche aufgebaut. Teil dieser Produktionslinie soll ein autonom fahrender Transportroboter sein, welcher die zwei in unterschiedlichen Gebäuden befindlichen Stationen der Produktionslinie miteinander verbinden soll. Im folgenden Kapitel wird auf die Grundlagen zum Projektumfeld (Zukunftsfabrik Bodensee), des Transportroboters (Husky) und das verwendete Betriebssystem Linux eingegangen. Zudem wird der zu benutzende Mapping-Algorithmus SLAM erläutert und die Funktionsweise der dafür genutzten LIDAR-Sensoren. In Kapitel 3 wird das Vorgehen im Projekt genauer beschrieben. Ziel dieser Studienarbeit ist die Inbetriebnahme dieses Transportroboters, was das Einrichten der Arbeitsoberfläche, das Auslesen der LIDAR-Sensordaten und die Inbetriebnahme eines 3D-LIDARs beinhaltet. Des Weiteren wird die optimale Höhe des 3D-LIDARs berrechnet und evaluiert, sowie eine dafür passende Halterung ausgewählt und montiert. Diese Themen bilden den Inhalt des Umsetzungs-Teils. Ab-

schließend wird die Arbeit zusammengefasst und ein kurzes Review formuliert. Im Anhang finden sich dann noch einige Datenblätter und ein "Quick Start Guide", welcher den nächsten Gruppen dabei helfen sollen schneller und einfacher in das Projekt einzusteigen.

2. Grundlagen

2.1. Zukunftsfabrik Bodensee

Die Zukunftsfabrik Bodensee am Campus Fallenbrunnen bildet einen zentralen strategischen Mittelpunkt für das Zentrum für Digitale Produktion (ZDP) und fungiert als Motor für innovative Forschungsaktivitäten. Die Idee hinter der Zukunftsfabrik Bodensee ist die Schaffung einer realen Wertschöpfungskette, die von vernetzten, adaptiven und selbstlernenden Systemen geprägt ist. Durch die Integration mit der bestehenden Lernfabrik des Instituts für Weiterbildung, Wissens- und Technologietransfer (IWT) entsteht ein einzigartiges Produktionsnetzwerk. In diesem Netzwerk wird ein 'Smart Product' in vielfältigen Varianten hergestellt und im Sinne der Kreislaufwirtschaft zurückgeführt. Die Zukunftsfabrik bildet somit die Grundlage für praxisorientierte Forschung und Lehre in den Bereichen Digitaler Zwilling, Künstlicher Intelligenz (KI) in der Produktion, Mensch-Maschine-Interaktion, Produktionsplattformen sowie Kreislaufwirtschaft. Die Forschungsvorhaben orientieren sich dabei stets an den aktuellen Bedürfnissen der Industrie, unter Einbeziehung kleiner und mittlerer Unternehmen (KMU) aus der Region.

Das Konzept der Zukunftsfabrik sieht ein Bandsystem mit Sensoren vor, das das Produkt durch vier Stationen führt. Die Auswahl der Variante erfolgt zu Beginn mittels eines QR-Readers, gefolgt von der Beladung des Werkstückträgers durch einen kollaborativen Roboter. An Station 2 werden Bolzen per Bin Picking kommissioniert und auf den Werkstückträger gelegt. Station 3 ist ein manueller Handarbeitsplatz mit ergonomischer Ausrichtung, der Vormontage und finale Montage vor der Endprüfung

im Produktionstechnischen Zentrum (PTZ) beinhaltet. Ein hochmodernes Werkerassistenzsystem begleitet den Arbeiter. Am Ende erfolgt die Endprüfung und Übergabe an das Transportsystem, das das Produkt zur Lernfabrik im Regionales Innovations- und Technologietransfer Zentrum (RITZ) transportiert. Dort findet sich eine automatisierte Schraubstation, gefolgt von einer Richtigkeitsprüfung an einem weiteren Endprüfstand, bevor das Produkt eingelagert und für die Demontage vorbereitet wird. Die Demontage erfolgt im gleichen Kreislauf wie die Montage.

Dieses Konzept bietet sowohl dem Campus als auch der regionalen Industrie einen faszinierenden Use Case, der gemeinsam die Implementierung und Erforschung von Industrie 4.0-Technologien ermöglicht.

Das Großprojekt am Campus Fallenbrunnen, welches im Oktober 2022 gestartet ist, setzt aktuell ein Projektteam aus 23 Studierenden, vier Professoren, vier Laboringenieuren, drei wissenschaftlichen Mitarbeitern sowie drei Doktoranden in 12 Teilprojekten in zwei Phasen um. Diese engagierte multidisziplinäre Gruppe arbeitet mit Hochdruck daran, die Konzepte der Zukunftsfabrik Bodensee in die Realität umzusetzen. Die Umsetzung erfolgt in enger Zusammenarbeit zwischen Forschern, Lehrenden und Studierenden, wobei der Fokus auf einer praxisorientierten Anwendung von Industrie 4.0-Technologien liegt. Durch diese dynamische Zusammenarbeit wird nicht nur die Forschung vorangetrieben, sondern es entstehen auch wertvolle Erfahrungen und Kompetenzen bei den Studierenden, die für ihre zukünftigen Karrieren von großem Nutzen sein werden. Die Zukunftsfabrik repräsentiert somit nicht nur einen Meilenstein in der Forschung und Lehre, sondern auch einen wichtigen Beitrag zur Förderung von Innovationen in der regionalen Industrielandschaft. [2]

2.2. Transportroboter Husky

Der Transportroboter muss einige Anforderungen erfüllen:

- Präzision: Der Roboter muss sich präzise durch seine Umgebung navigieren können, da unter anderem enge Passagen wie zum Beispiel Türen auf dem Weg sind. Zudem müssen Be- und Entladezonen der Produktionslinie präzise angefahren werden.

- Modularität: Um möglichst flexibel zu sein, sind viele Schnittstellen nötig, an denen vor allem diverse Sensoren angeschlossen werden können. Zudem muss die Plattform einfach bebaubar sein, damit Sensoren und Werkstückshalterungen montiert werden können.

- Performance: Der Roboter muss über eine ausreichend große Rechenkapazität verfügen um die Sensordaten zu analysieren, zu verarbeiten und sich je nach Position in die jeweilige Richtung auszurichten und fortbewegen zu können.

Der HUSKY Unmanned Ground Vehicle von Clearpath Robotics ist eine robuste und geländegängige Roboterplattform, die für vielfältige Anwendungen wie Forschung und Entwicklung, autonomes Fahren, Umweltüberwachung und Logistik eingesetzt wird. Diese Plattform ist darauf ausgelegt, autonom in verschiedenen Umgebungen zu operieren und dabei Hindernisse zu erkennen, Routen zu planen und sich in unbekannten Gebieten zu bewegen.

Die Mobilität des HUSKY wird durch Räder oder Raupen gewährleistet, und die Plattform verfügt über autonome Navigationsfunktionen, die es dem Roboter ermöglichen, sich sicher durch komplexe Umgebungen zu bewegen. Die Ausstattung mit einer Vielzahl von Sensoren wie Kameras, LIDAR und GPS ermöglicht dem HUSKY die präzise Erfassung seiner Umgebung.

Die Modularität ist ein weiteres Merkmal dieser Plattform, was es Benutzern ermöglicht, verschiedene Komponenten je nach Anforderung hinzuzufügen oder zu entfernen. Die drahtlose Konnektivität erleichtert die Fernsteuerung und Integration in bestehende Robotik-Systeme. [3]

Das Unternehmen MYBOTSHOP bietet diverse Erweiterungspakete für den Husky an. Der zu verwendete Transportroboter wurde direkt mit einigen Erweiterungen bestellt, wie zum Beispiel einem Industriecomputer auf welchem Linux installiert ist, sowie einem bereits eingebauten HOKUYO 2D-LIDAR. Diese zusätzlichen Komponenten bieten eine gute Grundlage für Navigation und Mapping Anwendungen. Die von MYBOTSHOP umgebaute Version des Husky ist in Abbildung 2.1 zu sehen.



Abbildung 2.1.: Roboter 'Husky' der Firma Clearpath Robotics [4]

Der Husky wurde durch ein Vorgängerprojekt erweitert durch ein Gestell, wie es in Abbildung 2.2 zu sehen ist. Die orangenen Plastikteile sind die Platzierungen bzw. Halterungen für die späteren beladenen Werkstücke. Durch das Gestell erreicht der Husky eine Gesamthöhe von 0,74 m. (Der zu sehende 3D-LIDAR ist hier nur provisorisch angebracht und entspricht nicht der endgültigen Höhe).



Abbildung 2.2.: Husky mit Erweiterung aus vorheriger Studienarbeit

2.3. Programmierung

2.3.1. Linux

Linux ist ein Open-Source-Betriebssystem, das von Tausenden Mitarbeitern entwickelt wurde. Im Gegensatz zu anderen Betriebssystemen ist Linux zuverlässiger und sicherer. Ein Linux-Betriebssystem bildet die Basis eines eigenen Betriebssystems, da es quelloffen ist und somit ein eigenständiges Erstellen eines Betriebssystems ermöglicht. Andere Betriebssysteme wie Windows oder MacOS enthalten ebenfalls Teile eines Linux-Systems, wodurch es allgegenwärtig wird. Linux bietet einige Vorteile, welche im Folgenden aufgezeigt werden sollen:

- Die Installation und die Verwendung von Linux ist im Gegensatz zu Windows vollständig **kostenlos**, was die Entwicklungskosten bei der Implementierung eines Roboters deutlich senkt.

- Dadurch, dass Linux von einer deutlich geringeren Anzahl an Benutzern (gegenüber Windows z.B.) bedient wird, lohnt es sich für Hacker nicht, Linux-Systeme anzugreifen, was Linux zu einem **sicheren Betriebssystem** macht.

- Durch die erhöhte Sicherheit durch ein Linux-System sind keine Antivirenprogramme notwendig, was den Speicher weniger belastet und das System damit zu einem **System mit hoher Geschwindigkeit** macht.

- Linux ist **programmierfreundlich**, da das System mit Compilern, Bashes, Shells und weiteren nützlichen Paketen geliefert wird, was es Programmierern erleichtert das Betriebssystem an ihre Bedürfnisse anzupassen.

- Eine **große und aktive Community** erleichtert die Fehlerbehebung.

Linux wird in der Roboterprogrammierung bevorzugt, weil es OpenSource, anpassbar, echtzeitfähig, günstig und mit umfangreichen Entwicklerwerkzeugen ist. [5]

2.3.2. Ubuntu

Genau genommen ist Linux gar kein Betriebssystem, sondern ein Kernel, also der Kern des Betriebssystems. Das Betriebssystem, das auf dem Linux-Kernel aufbaut, kann unterschiedliche Distributionen haben. Ubuntu ist eine solche Linux-Distribution. Ubuntu ist also ein Betriebssystem, das auf dem Linux-Kernel basiert. Ubuntu ist eines der bekanntesten Distributionen und bietet Vorteile wie der hohen Benutzerfreundlichkeit, einfache Installation und Konfiguration, große Auswahl für Software, moderner Look und hoher Kompatibilität[6] [7].

2.3.3. ROS

Roboter sind aus der heutigen Zeit schon lange nicht mehr wegzudenken. Die Tätigkeiten eines Roboters werden durch die Software festgelegt, welche von einem Hersteller abhängigen Programmiersprache geschrieben wird. Ein Unternehmen, welches verschiedene Roboter unterschiedlicher Hersteller in Betrieb nehmen will, müsste also für jeden Hersteller einen Experten einstellen, der die spezifische Programmiersprache kennt. Da dies ineffizient und kostenintensiv wäre, wurde 2007 ROS (Robot Operating System) entwickelt, um diese Sprachbarriere zu lösen. Grundgedanke ist das Erstellen eines quelloffenen Frameworks für die Programmierung von Robotern. ROS setzt auf ein vorhandenes Betriebssystem, wie z.B. Linux, Windows oder MacOS , auf und wird daher auch als Meta-Betriebssystem inklusive Middleware bezeichnet. Betriebssystem-Features wie Paketmanagement, Hardware-Abstraktion, Prozessmanagement und Geräte-Treiber, durch die der Roboter die Computerprogramme übertragen bekommt, werden jedoch von ROS selbst angeboten. Eine Softwarebibliothek ermöglicht es dem Anwender, Programme einfacher zu erstellen und diese flexibler anzupassen. ROS wird, wie es für eine Open Source üblich ist, von einer großen Community angetrieben. Durch die Bereitstellung von verschiedenen Paketen mit bestimmten, entwickelten Funktionen können Anwender ihren Roboter programmieren. ROS wird seit 2012 von einem Konsortium, bestehend aus Forschungseinrichtungen, Roboterhersteller, Technikkonzerne, Unternehmen aus der Automobilbranche und IT-Anbieter, weiterentwickelt, um das Framework auch für Industrieanwendungen auszu-

weiten. Dieses Konsortium stellt Pakete bereit, die industriellen Standards entsprechen und die Qualität sichern. ROS ermöglicht das Programmieren mit modernen Skriptsprachen wie Python oder C++. Ein Benutzer kann sich aus einer Vielzahl an Paketen bedienen, um die gewünschte Funktion mit seinem Roboter zu realisieren. Dabei ist es egal, von welchem Hersteller der Roboter stammt, da dieser ein mit ROS erstelltes Programm verstehen kann. ROS lebt also von seiner großen Community und deren Bereitstellung von Paketen, um bestimmte Funktionen herstellerunabhängig zu realisieren. Mit der Einführung von ROS 2 im Jahr 2017 erfüllt ROS nun auch Echtzeitanforderungen, was im industriellen Bereich oft erforderlich ist. Außerdem ist durch ROS 2 der koordinierte Einsatz mehrerer Roboter möglich. Das Marktforschungshaus ABI Research prognostizierte das Nutzen von mindestens einem ROS-Paket bei ca. 55% aller kommerziellen Robotern.[8]

2.4. SLAM

Der Begriff SLAM (**S**imultaneous **L**ocalization **a**nd **M**apping / Simultane Positionsbestimmung und Kartierung) entstand in den 80er-Jahren und beschreibt eine Technologie, die es Robotern ermöglicht, sich autonom und ohne Karte durch neue Umgebungen zu navigieren. Einerseits muss der Roboter eine Karte der Umgebung erstellen können. Andererseits muss er seine eigene Position ermitteln können. Die Schwierigkeit liegt darin, dass das Eine nicht ohne das Andere möglich ist. Um eine Karte der Umgebung erstellen zu können benötigt der Roboter eine Angabe zur eigenen Position. Die eigene Position wiederum ergibt sich aus der Kenntnis der Umgebung.[9]
Die Funktionsweise des SLAM lässt sich durch das Verhalten eines menschlichen Wesens in einem für ihn unbekannten Raum erklären. Um sich einigermaßen zurecht finden zu können werden nach markanten, stationären und leicht wieder erkennbaren Objekten gesucht. Wird der Abstand zu diesem Objekt abgeschätzt, kann die eigene Position in Relation zum Objekt ermittelt werden. Es gibt nun also einen Anhaltspunkt, um sich in diesem unbekannten Raum besser zurecht finden zu können. Lässt sich das betrachtete Objekt nicht wieder erkennen, so müssen einfach noch mehr Objekte der Umgebung eingeprägt werden, um sich durch die Kenntnis des Abstands zu

den Objekten im Raum zurechtfinden zu können. SLAM funktioniert auf die gleiche Art und Weise. Durch Sensoren werden sogenannte „Landmarks “identifiziert, anhand deren die eigene Position in Relation zu diesen Landmarks identifiziert werden kann. Die Umgebung wird weiter entdeckt und neue Landmarks identifiziert, bis sich eine Karte der Umgebung erstellen lässt. SLAM-fähige Geräte können sowohl eine Karte der Umgebung erstellen, als auch die eigene Position in dieser Karte erkennen [10]. Die Sensoren können sowohl visuelle (z.B. Kamera), als auch nicht sichtbare (z.B. LIDAR oder RADAR) Daten liefern, welche der SLAM-Algorithmus verwendet. SLAM ist ein iterativer Prozess, das heißt, je öfter diese Schritte (Landmarks identifizieren und eigene Position daraus ermitteln) wiederholt werden, desto genauer kann die Position des Roboters und die Strukturen der Umgebung ermittelt werden [11].

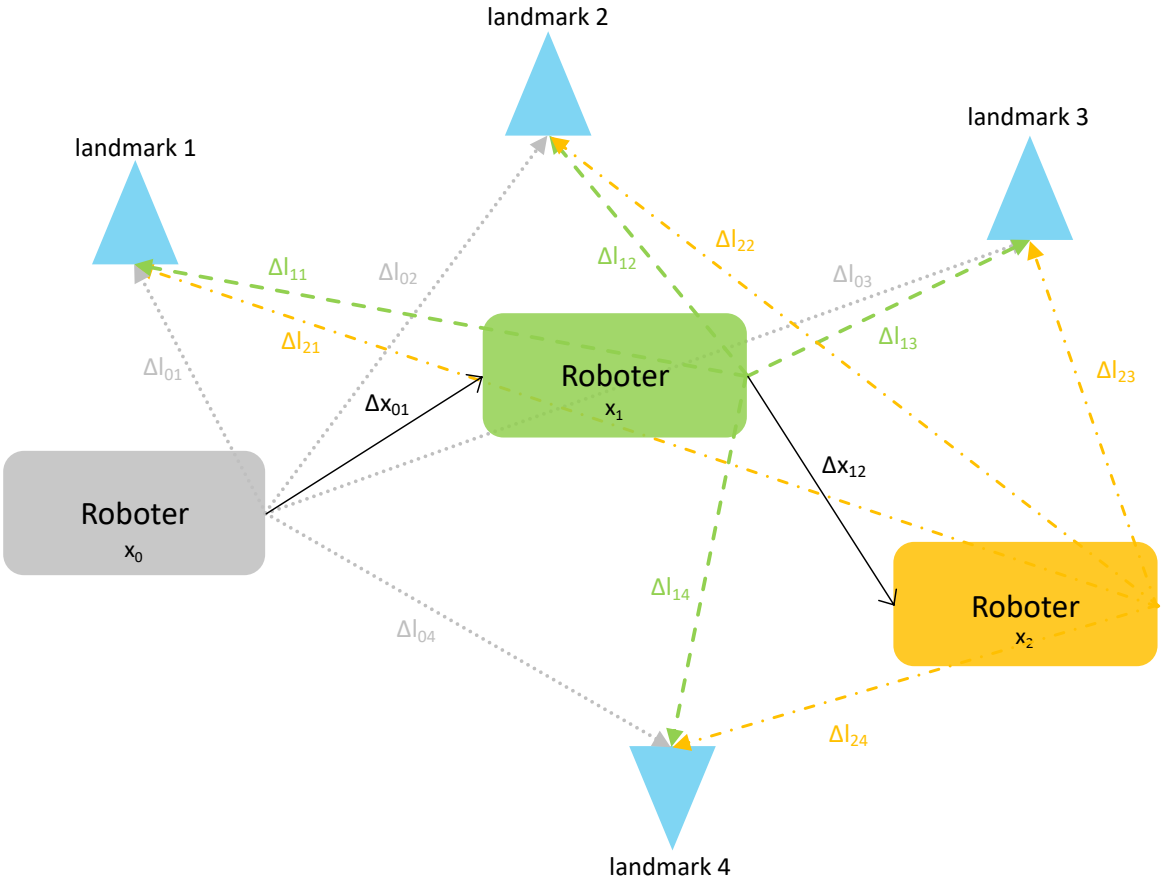


Abbildung 2.3.: Grundprinzip der Positionserkennung des Roboters durch Bestimmung des Abstands zu identifizierten Landmarks, eigene Darstellung

Abbildung 2.3 zeigt beispielhaft, wie sich ein Roboter in einem für sich unbekannten Raum an verschiedenen Positionen zurechtfinden kann. Durch die Identifikation der Landmarks und die Berechnung der Abständen zu den Landmarks, abhängig von der jeweiligen Position (x_0 bis x_2) kann sich der Roboter im Raum orientieren. Bei Position x_1 kennt der Roboter dank der Sensordaten die Abstände zu den Landmarks Δl_{11}, Δl_{12}, Δl_{13} und Δl_{14}. Durch diese Abstände kann er seine eigene Position im Raum abschätzen. Durch das Hinzufügen mehrere Landmarks entsteht eine Karte des Raums und der Roboter kann sich im gesamten Raum zurecht finden. Die SLAM-Technologie bietet im Gegensatz zu herkömmlichen Navigationstechniken wie das GPS (Global Positioning System) mehrere Vorteile. Während GPS abhängig von Satelliten arbeitet und ständig eine Verbindung zu ihnen benötigt, kann SLAM selbstständig und in unbekannten Umgebungen (vor allem auch in Fabriken mit schlechtem GPS-Empfang) eingesetzt werden. Da bei jeder Iteration die Karte aktualisiert wird, kann der Roboter ziemlich genau seine eigene Position bestimmen. Herausforderungen beim Einsatz von SLAM ergeben sich vor allem bei der Datenzuordnung, also dem Abgleich von Daten verschiedener Sensoren. Diese Daten sind oft verrauscht oder unvollständig. Außerdem können SLAM-Algorithmen rechenintensiv sein und damit den Einsatz bei Echtzeitanwendungen erschweren [12].

2.5. LIDAR

Ähnlich zu Radar und Sonar ist auch ein LIDAR (light detection and ranging) ein Abstandsmessungssensor. Ein LIDAR sendet Laserimpulse aus, die von Objekten reflektiert werden und durch die Lichtenergie eine Bestimmung von Abständen zu Objekten ermöglicht. Dadurch lassen sich Strukturen der Umgebung wahrnehmen und in einer 2D bzw. 3D-Map darstellen.



Abbildung 2.4.: 2D-LIDAR von HOKUYO [13]



Abbildung 2.5.: 3D-LIDAR von Velodyne [14]

Ein 2D-LIDAR, wie eines in Abbildung 2.4 abgebildet ist, kann zwei Dimensionen (Abstand und Azimut-Winkel) messen. Dadurch ergeben sich Sensordaten in einer Ebene. In Abbildung 2.5 ist ein 3D-LIDAR dargestellt, welches zusätzlich zum Abstand und dem Azimut-Winkel noch den Elevations-Winkel messen kann. Dadurch können die Sensordaten in einem ganzen dreidimensionalen Raum dargestellt werden. In Kombination mit einem Lokalisierungsmodul ist eine simultane Standortbestimmung und Kartierung (SLAM) möglich. Abbildung 2.6 zeigt beispielhaft eine LIDAR-Punktwolke einer Straßenszene, welche mit einem 3D-LIDAR aufgenommen wurde. Der Arbeitsbereich eines LIDAR liegt im Bereich der Infrarotstrahlung bzw. im Bereich des für den Menschen sichtbaren Lichts. Im in Abbildung 2.7 aufgezeigten elektromagnetischen Spektrum liegt der Arbeitsbereich des LIDARs demnach links von dem des Radars, das heißt der Arbeitsbereich eines LIDAR ist hochfrequenter bzw. kurzwelliger (höhere Frequenz f und kleinere Wellenlänge λ). LIDAR-Sensoren können gegenüber anderen Sensoren wie Kamera oder Radar hochgradig genaue, strukturelle und dreidimensionale Informationen zu ihrer Umgebung liefern.

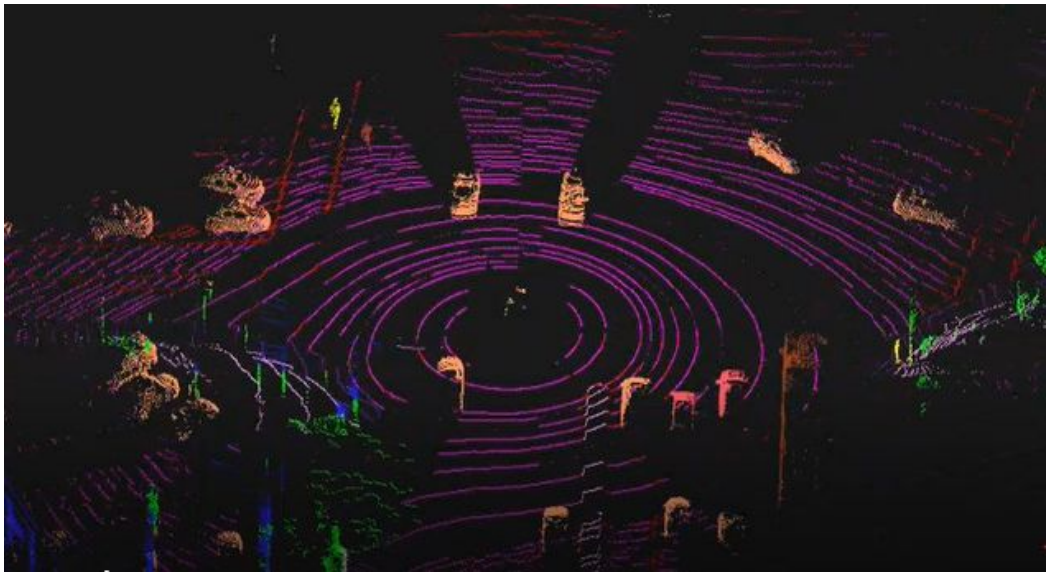


Abbildung 2.6.: LIDAR-Punktwolke einer Straßenszene [15]

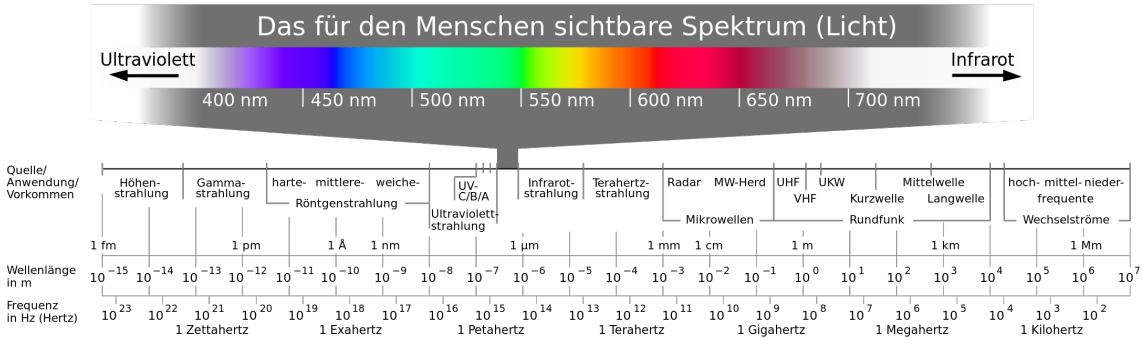


Abbildung 2.7.: Übersicht des elektromagnetischen Spektrums [16]

Es gibt drei wichtige Gründe, die zur Akzeptanz von LIDAR-Sensoren auf dem Markt beitragen:

- Die Verfügbarkeit für vergleichsweise günstigere Anwedungen hat aufgrund der **kostengünstigeren** LIDARs mit verbesserten Eigenschaften hinsichtlich Reichweite, Größe und Fehlerresistenz stark zugenommen.

- Gegenüber Radare oder Sonare weisen LIDARs eine **erhöhte Genauigkeit** durch die hochdichten 3D-Informationen und der daraus entstehenden Punktwolke auf.

- Mithilfe von Tools können **LIDAR-Verarbeitungsalgorithmen** entwickelt und angewendet werden, da es der Industrie durch LIDAR-Verarbeitungs-Workflow, wie z.B. SLAM ermöglicht wurde, LIDARs in den Entwicklungs-Workflow mit einzubeziehen.

Es gibt verschiedenste Anwendungsbereiche eines LIDARs, welche sich grob in drei Gruppen einteilen lassen. Zum einen gibt es die Luft-LIDARs, die wie der Name schon vermuten lässt auf einem fliegenden Objekt montiert sind. Dadurch kann eine 3D-Punktwolke eines großen Gebietes für die Kartenerstellung, Merkmalsextraktion, Geländeklassifizierung etc. genutzt werden. Anwendungen finden sich beispielsweise in der Landwirtschaft zur Kartierung der Vegetationsfläche, in der Städteplanung, der geologischen Kartenerstellung für den Bergbau oder die Forstwirtschaft, oder für die autonome Navigation im Luftraum. Zum anderen gibt es die Boden-LIDARs, welche sich weiter in die stationäre terrestrische und die mobilen LIDARs trennen lassen. Für Landvermessungen, Straßenvermessungen, topologische Kartierungen etc. werden hauptsächlich stationäre terrestrische LIDARs eingesetzt, da sie sich für eine detaillierte und engmaschige Datenerfassung anbieten. Das wichtigste Einsatzgebiet mobiler LIDARs hingegen ist das autonome Fahren, bei dem der Sensor auf einer stationären Plattform angebracht wird und dort für die Realisierung verschiedenster Funktionen verantwortlich ist. Als letztes seien noch die für diese Studienarbeit relevantesten LIDARs zu erwähnen, die sogenannten Innen-LIDARs. Haupteinsatzgebiet ist die Robotik-Anwendung, bei der die Sensoren auf dem mobilen Roboter montiert werden. Es werden sowohl 3D-LIDARs als auch 2D-LIDARs eingesetzt, die nützlich für mehrere Anwendungen sind. Mithilfe der 2D- bzw. 3D-Sensoren lassen sich durch SLAM-Karten erstellen, die Voraussetzung für das autonome Fahren eines mobilen Roboters sind. Vor Allem 2D-LIDARs werden zudem für die Hinderniserkennung und Kollisionswarnung bzw. -vermeidung eingesetzt und gewähren somit die Sicherheit zum einen des Roboters und zum anderen allen anderen sich in der Nähe befindlichen Objekte oder Lebewesen.[15]

3. Vorgehen

Zu Beginn des Projekts wurde die Ausgangssituation betrachtet. Der Transportroboter war verfügbar und kann mit einem Controller manuell gesteuert werden. Zusätzlich ist bereits ein 2D-LIDAR angebracht, sowie eine zweite Plattform, auf welcher die Werkstücke be- und entladen werden. Des Weiteren wurde ein 3D-LIDAR bestellt.

Um ein Überblick über Ziele, Aufgaben und Fortschritt zu bekommen hat das studentische Organisationsteam das Programm Trello vorgestellt und zusätzlich zwei Whiteboards im PTZ organisiert. Dort konnten verschiedene Themen in die Abschnitte 'To-Do', 'In Progress', 'In Review' und 'Done' eingeteilt werden. Mit Hilfe von Trello und den Whiteboards haben auch die verschiedenen Studentengruppen untereinander einen guten Überblick, wie weit die Teams aktuell sind und ab wann Schnittstellen hergestellt werden können. Um verschiedene Ziele und Meilensteine zu definieren wurde ein Top-Down Ansatz gewählt, bei welchem das Hauptziele in verschiedene Zwischenziele aufgeteilt wird. Durch eine solche Analyse kann sichergestellt werden, dass die verschiedenen Hauptziele bzw. Anforderungen nicht vergessen oder verfehlt werden. Zudem gibt es eine klare Hierarchie und Zuteilung, wodurch es wenig Missverständnisse im Bezug auf Verantwortlichkeiten und Zuständigkeiten gibt.

- Inbetriebnahme Transportroboter
  - Implementierung SLAM-Algorithmus
    - Sensordaten des 2D-LIDAR und 3D-LIDAR anzeigen lassen
    - Erste Karte aus Sensordaten erstellen
    - Roboter zu festgelegten Koordinaten auf Karte fahren lassen
  - Integration Benutzerschnittstelle (HMI)
    - Programmierung an festem Arbeitsplatz im PTZ mit Monitor, Tastatur und Maus
    - Programmierung mit separatem Laptop, verbunden über LAN (Kabel)
    - Programmierung mit separatem Laptop, verbunden über WLAN (Kabellos)
    - Programmierung über direkte Schnittstelle am Roboter (mobiler Monitor)
  - Integration 3D-LIDAR
    - Technische Inbetriebnahme
    - Evaluation Positionierung auf Roboter
- Schnittstelle zwischen Produktionslinie und Roboter herstellen

4. Umsetzung und Ergebnisse

Wichtig! Damit der Roboter voll funktionsfähig eingesetzt werden kann, muss dieser mithilfe der Remote Verbindung, wie sie in Kapitel 4.1 beschrieben ist, neu gestartet werden.

4.1. Einrichten der Arbeitsoberfläche

Um den Husky zu programmieren, muss eine Benutzeroberfläche aufgebaut werden. Dafür werden vorerst ein Bildschirm über ein HDMI-Kabel mit dem im Husky verbauten Computer verbunden. Zusätzlich werden eine Maus und eine Tastatur angeschlossen. Da der Roboter nur eine USB-Schnittstelle nach außen verfügt, wurde ein USB-Hub angeschafft, welcher einen USB-Port in mehrere aufteilt. Somit können gleichzeitig Maus, Tastatur sowie der USB-Dongle für die kabellose Verbindung zum Kontroller zur manuellen Steuerung des Husky verwendet werden.

Da man mit dieser Benutzeroberfläche sehr standortgebunden (Arbeitsplatz im PTZ) ist, aber der Roboter auf der ganzen Strecke zwischen PTZ und RITZ getestet werden muss, ist eine flexiblere Methode notwendig. Zum einen kann eine Remote Verbindung zwischen dem Husky und einem separaten Laptop hergestellt werden. Dafür müssen sowohl Husky, als auch der Laptop im gleichen WLAN Netzwerk sein. Dieses muss vorher an der stationären Arbeitsoberfläche des Husky über die Linux-Netzwerkeinstellungen ausgewählt werden. Über das vorhandene DHBW-Netzwerk konnte keine direkte Verbindung zwischen Husky und Laptop aufgebaut werden. Vermutlich werden hier die IP-Adressen stark verändert und es bedarf eines speziellen

Routings. Auch vorhandene Sicherheitszertifikate könnten die Verbindung blockieren. Daher wurde vorerst auf einen einfachen mobilen Hotspot eines Handys zurückgegriffen. Nachdem beide Endgeräte mit dem WLAN verbunden sind, muss über ein Terminal auf dem Husky-Computer folgender Befehl eingegeben werden:

```
ifconfig
```

Mit diesem Befehl werden alle Netzwerkschnittstellen angezeigt. Die Eigenschaften der WLAN-Verbindung sind unter der Bezeichnung 'wlp2s0' aufgelistet und in Abbildung 4.1 dargestellt. Anschließend kann unter dem Punkt 'inet' die IP-Adresse (blau

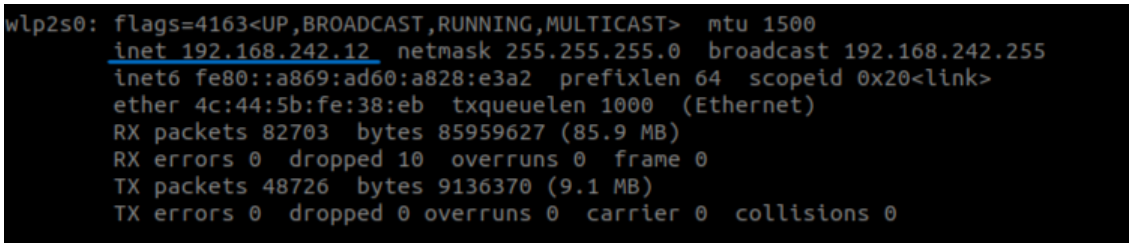


Abbildung 4.1.: Eigenschaften der Netzwerkschnittstelle 'wlp2s0'

unterstrichen) des Huskys innerhalb des WLAN-Netzwerks abgelesen werden. Wichtig hierbei ist, dass die IP-Adresse nicht immer gleichbleibt. Durch zum Beispiel das Neustarten des mobilen Hotspots kann sich die IP-Adresse ändern. Daher muss diese, bevor die Remote Verbindung hergestellt werden kann, immer erst ausgelesen werden. In diesem Beispiel lautet die IP-Adresse '192.168.242.12'. Mit dieser IP-Adresse kann man jetzt am Laptop die Remote Verbindung einrichten. Dafür öffnet man auf dem separaten Laptop (Windows) die Eingabeaufforderung (cmd) oder Powershell. Dort wird folgender Befehl eingegeben:

```
ssh administrator@192.168.242.12
```

Hier wird die oben ermittelte IP-Adresse beispielhaft verwendet, wenn diese anders ist, lautet der Befehl auch anders. Anschließend sollte eine Eingabeaufforderung für das Passwort erscheinen. Dieses lautet standardmäßig 'clearpath'. Beachte, dass das Passwort beim Eingeben nicht angezeigt wird, jedoch wird dieses 'unsichtbar' geschrieben.

Um das Passwort zu bestätigen, muss die 'Enter'-Taste gedrückt werden. Nach erfolgreicher Eingabe sollte eine von MYBOTSHOP konfigurierte Übersicht erscheinen, wie sie in Abbildung 4.2 dargestellt ist. In dieser Übersicht können einige Informationen

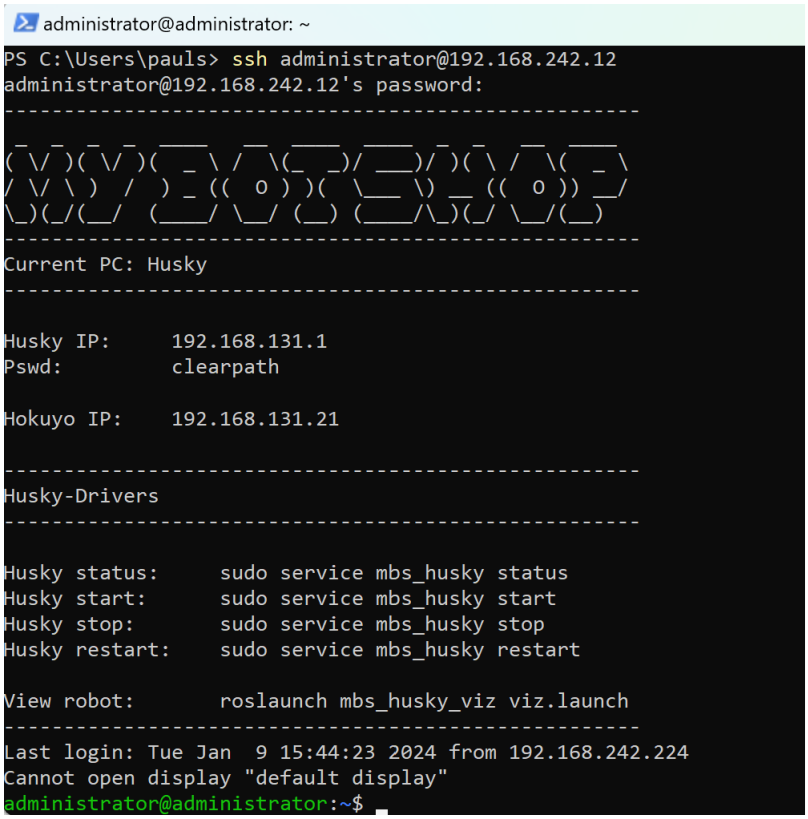


Abbildung 4.2.: Konfigurierte Übersicht von MYBOTSHOP

und mögliche Befehle abgelesen werden. So kann der Roboter über folgenden Befehl neu gestartet werden:

```
sudo service mbs_husky restart
```

Dieser Befehl muss durch eine erneute Eingabe des Passworts 'clearpath' bestätigt werden. Wie am Anfang dieses Kapitels erwähnt, muss dieser Befehl ausgeführt werden um den Husky vollumfänglich nutzen zu können. Die oben durchgeführten Schritte haben sich an folgender Anleitung orientiert [17].

Mit diesem Remote Setup kann im Moment der Husky nur gestoppt und neu gestartet

werden, sowie einige weitere interne Befehle eingegeben werden. Um zum Beispiel über RViz-Sensordaten anzeigen zu lassen wird auf dem separaten Laptop ein Linux Betriebssystem benötigt.

Da dieser Schritt ein zu großer Sprung wäre, war die nächste Überlegung ein mobiles Setup direkt auf dem Roboter, um zum Beispiel die Sensordaten anzeigen zu lassen. Dafür wurden sogenannte mobile Monitore in Betracht gezogen. Diese sind meist kleiner und leichter als normale Monitore, brauchen daher auch weniger Energie. Diese können meist über einen USB-Port betrieben werden, welcher entweder direkt vom Husky kommt oder durch eine externe Batterie bereitgestellt werden kann. Es gibt auch einige mobile Monitore, die bereits integrierte Akkus haben und einige Stunden ohne Laden betrieben werden können. Eine genaue Auswahl und Anschaffung eines solchen Monitors wird Teil der nächsten Studienarbeit sein. Zudem muss die Halterung bzw. Befestigung des Monitors am Roboter selbst berücksichtigt werden. Die Eingabe über Text und Maus kann zum Beispiel über eine Bluetooth-Tastatur und Maus umgesetzt werden, damit auch hier die nötige Flexibilität gewährleistet ist. Zusätzlich gibt es mobile Monitore mit Touch-Eingabe, wodurch im reduzierten Umfang auch Benutzereingaben gemacht werden können.

4.2. RViz-Sensordaten (2D-LIDAR)

Bereits beim Kauf war ein 2D-LIDAR (Hokyuo UTM-30LX-EQ mit der Artikelnummer MBS-SES-206) auf dem Husky montiert.
Folgend die wichtigsten Spezifikationen aufgelistet [13]:

- Reichweite: 30 m
- Toleranz ±3cm
- Öffnungswinkel Azimut (horizontal): 270°

RViz ist ein leistungsstarkes 3D-Visualisierungstool, das in der Robotik für die Darstellung von Robotersensordaten, Navigation und anderen wichtigen Informationen verwendet wird. Es ist ein Teil des ROS und ermöglicht die visuelle Überwachung und Analyse von Robotersystemen.
Um RViz zu starten, muss folgender Befehl in der Konsole ausgeführt werden:

```
roslaunch mbs_husky_viz viz.launch
```

Danach öffnet sich die GUI von RViz und je nachdem, ob bereits Sensorik mit ROS verknüpft wurde, zeigt es die aktuellen Daten des Sensors an. In diesem Fall wurde in RViz ein Bild nach Abbildung 4.3 angezeigt.

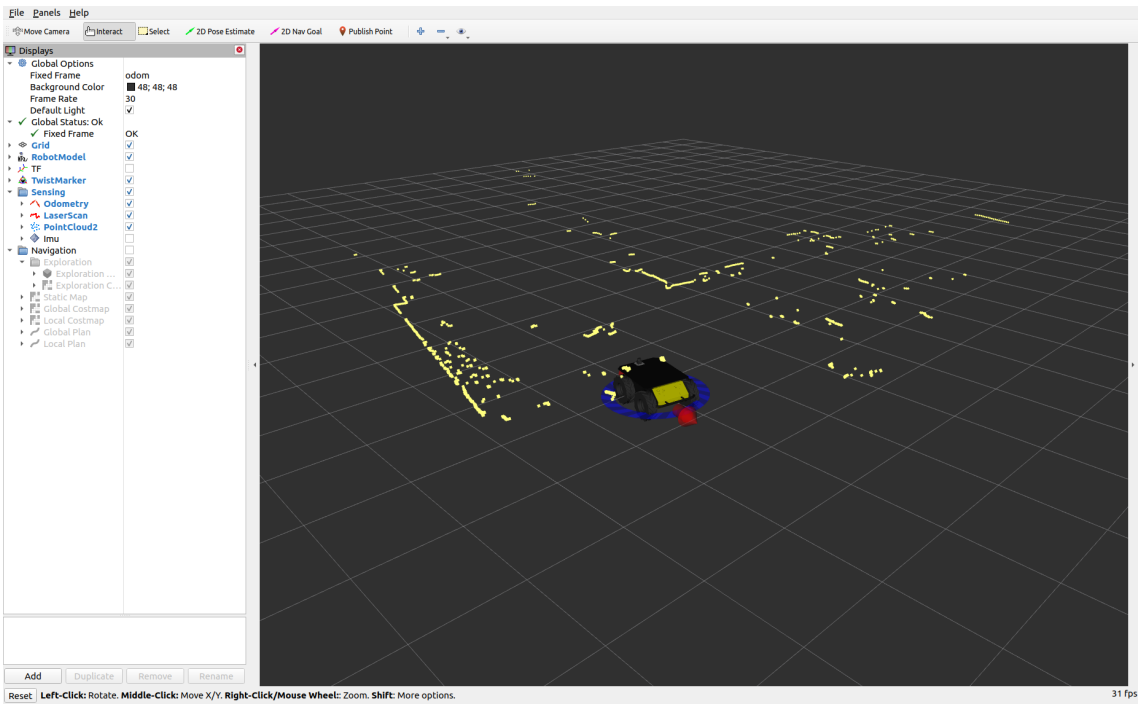


Abbildung 4.3.: Benutzeroberfläche von RViz mit bereits sichtbaren 2D-LIDAR Daten

Durch die roten Pfeile lässt sich der Roboter nach vorne und nach hinten fahren, über den blauen Kreis lässt er sich um die eigene Achse rotieren. Demnach könnte theoretisch auf einen externen Controller verzichtet werden. (Aufgrund der angeschlossenen Kabel lässt sich der Roboter natürlich nur in einem begrenzten Raum bewegen). Bei

der Bewegung des Roboters bewegt sich das Sensorbild in RViz mit und ermöglicht somit eine nahezu echtzeitfähige Übertragung der LIDAR-Daten. Eine aktualisierte Punktwolke (nach hauptsächlich rotatorischer Bewegung des Roboters) ist in Abbildung 4.4 zu sehen. Zu erkennen ist, dass sich die Punktwolke um etwa 180° gedreht hat. Ebenfalls deutlich zu erkennen ist, dass die Punktwolke in einer Ebene liegt, also zweidimensional ist. Das liegt daran, dass das verwendete LIDAR nur zweidimensional ist und kein Elevationswinkel gemessen wird, wordurch die Höhe von Objekten bestimmt werden könnten.

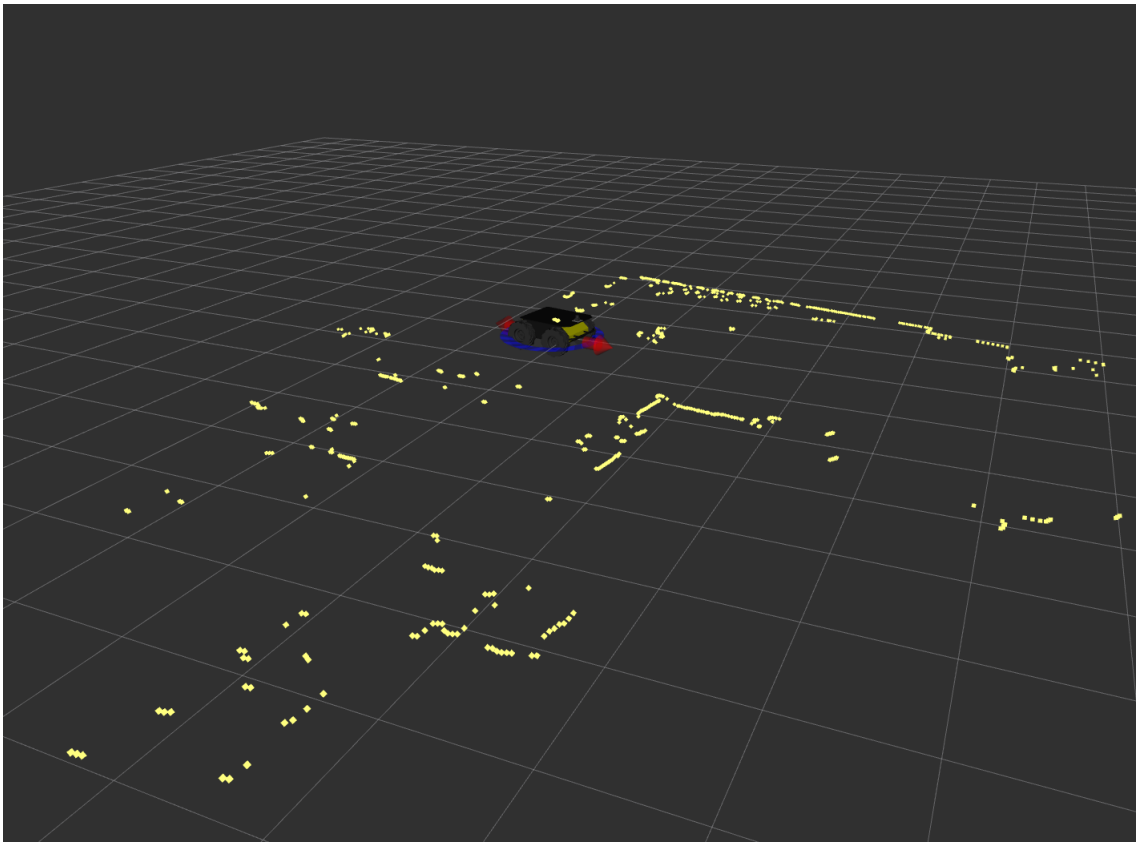


Abbildung 4.4.: Punktwolke der LIDAR-Daten nach rotatorischer Bewegung gegenüber Abbildung 4.3

4.3. Integration 3D-LIDAR

Damit die Karte, welche mithilfe des SLAM-Algorithmus erstellt wird, detaillierter ist, wird der 3D-LIDAR Sensor 'Puck' (VLP-16) der Firma Velodyne integriert [14]. Dieser misst im Vergleich zu einem 2D-LIDAR nicht nur den Azimut-Winkel und Abstand, sondern auch den Elevations-Winkel. Dadurch ist es möglich Hindernisse auf verschiedenen Höhen wahrzunehmen und mehrere Referenzpunkte für die Karte zu generieren. Die wichtigsten Spezifikationen sind hier aufgelistet:

- Reichweite: 100 m
- Anzahl Kanäle (Azimut): 16
- Reichweitengenauigkeit: ±3 cm
- Field of View / Blickfeld Elevation (Vertikal): +15° bis -15° (30°)
- Winkelauflösung Elevation: 2°
- Field of View / Blickfeld Azimut (Horizontal): 360°
- Winkelauflösung Azimut: 0,1° - 0,4°

In den weiteren Unterkapiteln wird zum einen die technische Inbetriebnahme genauer dokumentiert, sowie die Positionierung des Sensors auf dem Roboter selbst.

4.3.1. Technische Inbetriebnahme

Die folgenden Prozessschritte orientieren sich an der Anleitung unter [18].

4.3.1.1. Physikalische Verbindung

Der 3D-LIDAR Sensor ist, wie in Abbildung 4.5 dargestellt, direkt mit der Interface Box verbunden. Diese wiederum wird über Ethernet mit dem Husky verbunden, um den Datentransfer zu ermöglichen. Zuletzt muss die Interface Box noch mit Strom versorgt werden, was im Moment noch über das Netz des PTZ läuft. Später wird die Interface Box dann mit Strom von der Batterie des Husky versorgt.

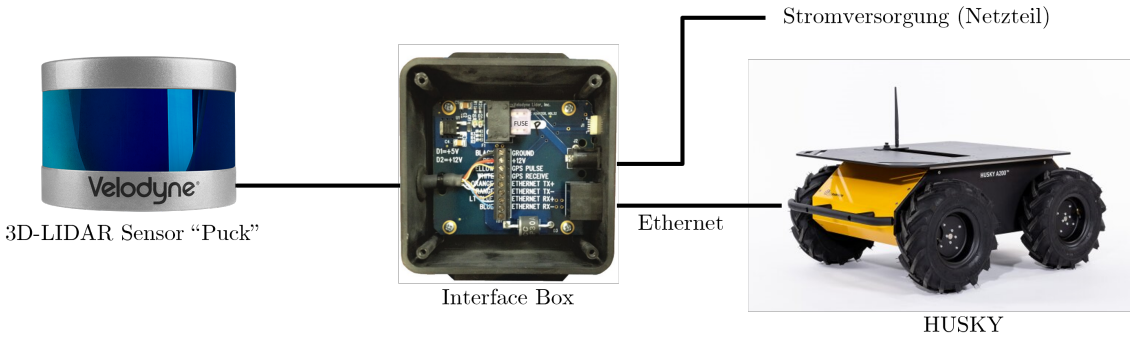


Abbildung 4.5.: Aufbau der physikalischen Verbindung des 3D-LIDAR [14] [3]

4.3.1.2. Computer IP-Adresse konfigurieren

Um die IP-Adresse des Computers zu konfigurieren, muss zunächst das 'Gnome Menu' geöffnet werden, was den gewöhnlichen Einstellungen entspricht. Unter 'Networks Connection' kann nun manuell eine Verbindung erstellt werden. Zunächst wird in den Einstellungen lediglich der Name der Ethernet-Verbindung angezeigt, im Falle des 3D-LIDARs heißt diese 'eno1'. Unter einem '+' bzw. 'add' kann nun manuell eine IP-Adresse konfiguriert werden. Im Falle des 3D-LIDAR wurde die Konfiguration entsprechend Abbildung 4.6 durchgeführt. Die IP-Adresse muss die Form '192.168.1.XXX'

aufweisen. Die letzte Zahl der IP-Adresse kann hierbei eine Zahl zwischen 1 und 254 sein (jedoch nicht 201, da diese vom 3D-LIDAR belegt ist). In diesem Fall wurde die IP-Adresse zu '192.168.1.100' gewählt. Das Gateway wurde im Gegensatz zur Anleitung, die sich unter [18] findet, zu '192.168.1.1' konfiguriert, da mit dem vorgeschlagenen Gateway keine erfolgreiche Verbindung hergestellt werden konnte.

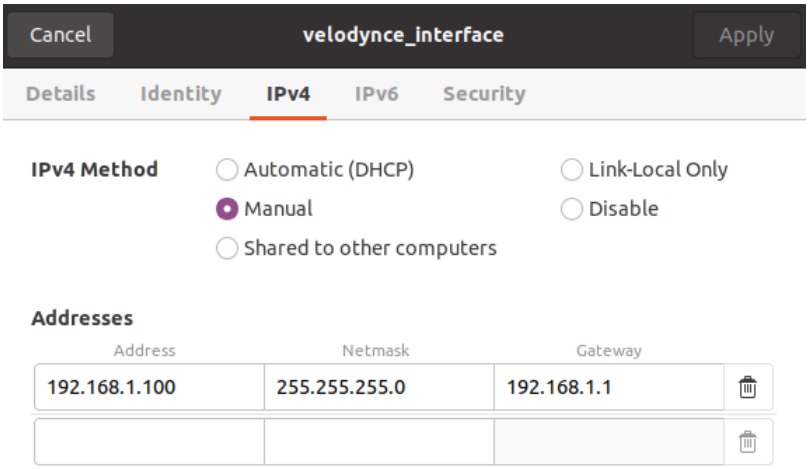


Abbildung 4.6.: Einstellungen für die Konfiguration der IP-Adresse des Computers

4.3.1.3. LIDAR mit Computer verbinden

Um den LIDAR nun mit dem Computer zu verbinden, muss folgender Befehl im Terminal eingegeben werden:

```
sudo ifconfig eno1 192.168.3.100
```

Mit diesem Befehl wird eine statische IP-Adresse hinzugefügt. In diesem Fall gab es zwei Ethernet-Netzwerkverbindungen: einmal 'eno1' und einmal 'enp1s0'. Um herauszufinden, welche zum 3D-LIDAR und welche zum 2D-LIDAR gehört, wurde das Ethernet-Kabel des 3D-LIDARs vom Husky getrennt. Anschließend war nur noch die Verbindung 'enp1s0' zu sehen, somit gehört diese zum 2D-LIDAR und folglich 'eno1'

zum 3D-LIDAR. Danach muss eine statische Route zur IP-Adresse des LIDARs durch folgenden Befehl hinzugefügt werden

```
sudo route add 192.168.1.201 eno1
```

Die IP-Adresse des LIDARs sollte standardmäßig '192.168.1.201' betragen, dies wurde jedoch noch einmal durch Anschließen an einen Laptop und der Software Wireshark überprüft. Nun sollte die Verbindung zwischen Computer und LIDAR erfolgreich abgeschlossen sein. Dies lässt sich überprüfen, indem die IP-Adresse des LIDARs im Web-Browser des Computers eingeben wird. Erscheinen sollte ein Webseite, wie in Abbildung 4.7 dargestellt. Hier kann beispielsweise die Drehzahl des LIDAR-Motors unter 'Motor RPM' angepasst werden. Setzt man diesen Wert auf einen niedrigeren Wert ist deutlich hörbar, dass sich die Drehzahl verringert hat. Ebenfalls kann unter 'FOV Start' und 'End' der Winkel angepasst werden. Dies lässt sich dann auch nach Einrichtung des RViz-Bildes nachweisen. Erreicht man diese Webseite kann von einer erfolgreichen Verbindung zwischen Computer und LIDAR ausgegangen werden.

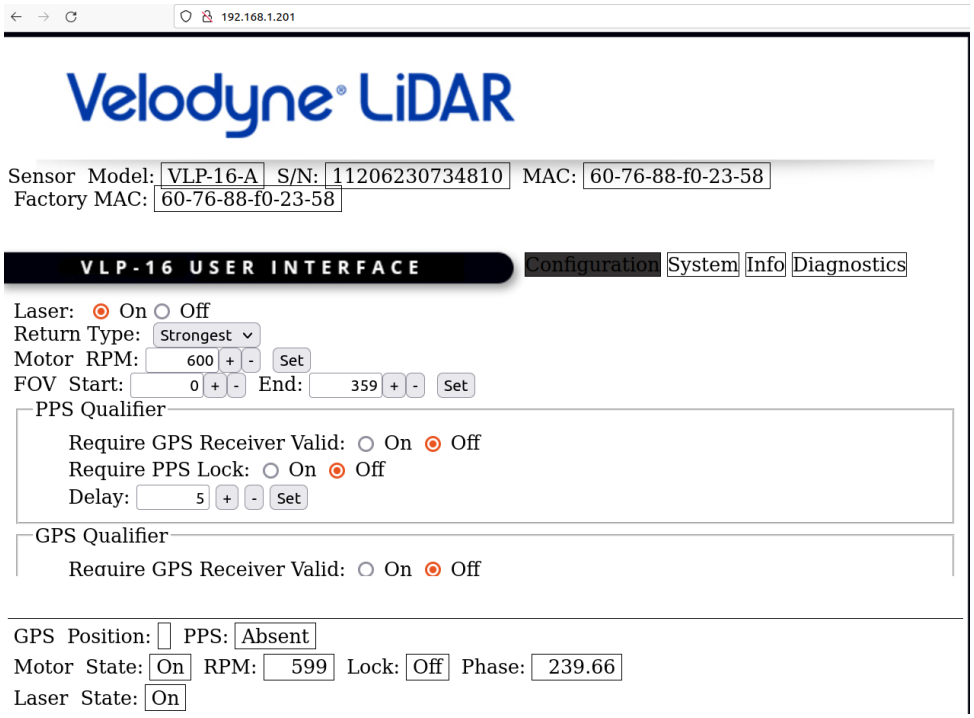


Abbildung 4.7.: IP-Adresse des LIDARs zum Nachweise einer erfolgreichen Verbindung zwischen Computer und Sensor

4.3.1.4. Dateien und Treiber installieren

Als nächsten Schritt müssen die 'ROS dependencies' installiert werden. Abhängig von der ROS-Version muss folgender Befehl im Terminal eingeben werden

```
sudo apt-get install ros-VERSION-velodyne
```

Da auf dem Husky standardmäßig die Version 'noetic' installiert wurde konkretisiert sich der Befehl zu

```
sudo apt-get install ros-noetic-velodyne
```

Die Version kann mit folgendem Befehl noch einmal nachgeprüft werden:

```
lsb_release -a
```

Als nächsten Schritt muss im Terminal in den 'src'-Ordner des entsprechenden Workspace gewechselt werden. Der bereits installierte Workspace 'ros_ws' und der darin enthaltene Ordner 'src' lässt sich durch folgenden Befehl erreichen:

```
cd ~/ros_ws/src
```

Folglich muss das von Velodyne bereitgestellte Git-repository in den 'src'-Ordner geklont werden

```
git clone https://github.com/ros-drivers/velodyne.git
```

Die nächsten beiden Schritte, die in der Anleitung angeben sind können übersprungen werden, da sie nur Fehlermeldungen ausspucken und keinen Einfluss auf die Darstellung der LIDAR-Daten in RViz haben.

4.3.1.5. Daten in RVIZ anzeigen

Um die Daten des 3D-LIDAR in RViz anzeigen zu lassen muss zunächst folgender Befehl ausgeführt werden:

```
roslaunch velodyne_pointcloud VLP16_points.launch
```

Durch diesen Befehl laufen die notwendigen 'nodes'. Dieses Terminal muss im Hintergrund dauernd geöffnet bleiben. Um zu sehen, welche 'nodes' aktuell laufen kann folgender Befehl eingegeben werden

```
rosnode list
```

Um RViz nun zu starten, muss ein neues Terminal geöffnet werden und dieser Befehl eingegeben werden

```
rosrun rviz rviz -f velodyne
```

Es öffnet sich die RViz Benutzeroberfläche, wie in Abbildung 4.8 dargestellt. Zunächst sollten keine Sensordaten angezeigt werden. Unter 'Add' muss nun 'PointCloud2' und im dann erscheinenden Feld 'Topic' '/velodyne_points' ausgewählt werden. Danach sollten die Daten des 3D-LIDARs in RViz angezeigt werden. Je nachdem, wo sich der Roboter befindet ergibt sich ein Bild nach Abbildung 4.9.

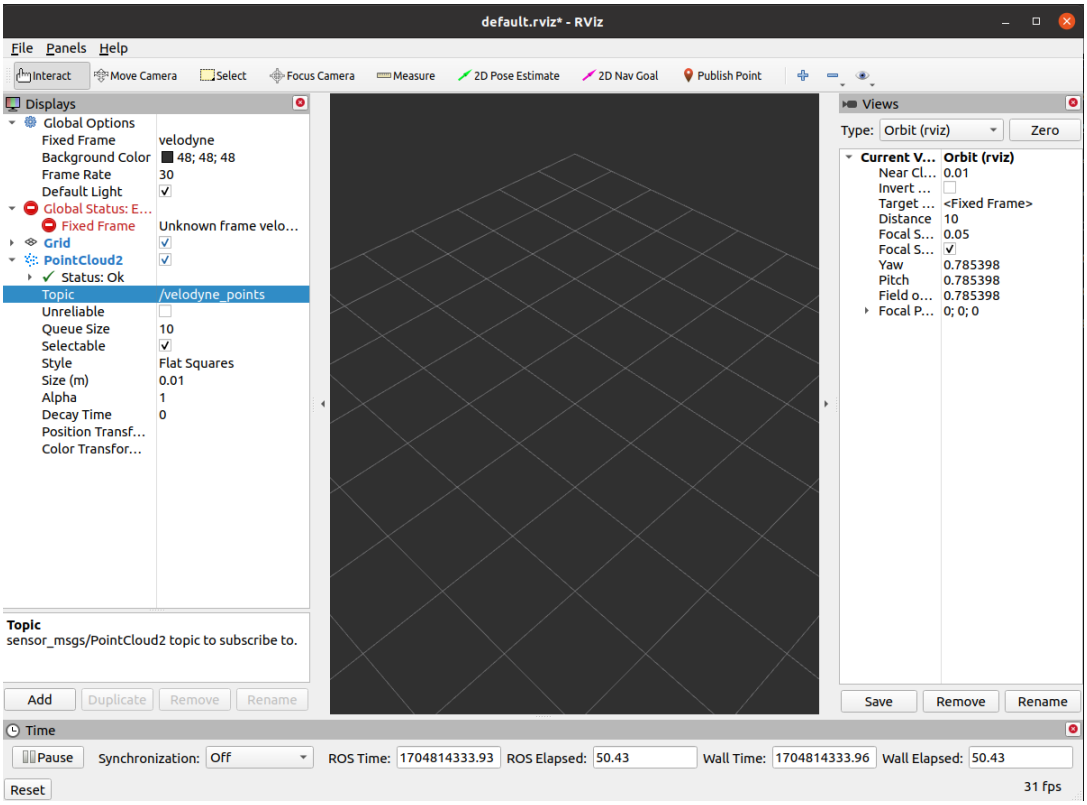


Abbildung 4.8.: Benutzeroberfläche RViz zur Darstellung der 3D-LIDAR Daten

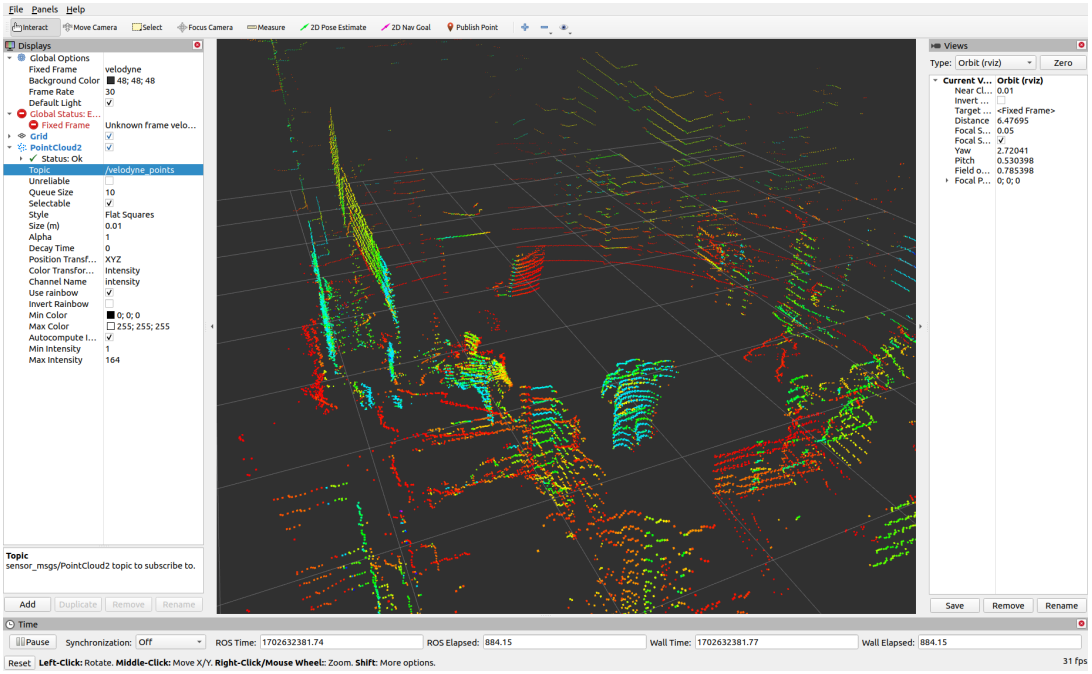


Abbildung 4.9.: Echtzeitdaten des 3D-LIDARs dargestellt in RViz

4.3.2. Berechnung der Positionierung auf Roboter

Um die bestmögliche Position für den Sensor zu finden, wurden verschiedene Möglichkeiten betrachtet. Zum einen könnte dieser auf der gleichen Ebene wie der 2D-LIDAR auf der anderen Seite des Roboters angebracht werden. Auf dieser Position wird jedoch die Sicht stark eingeschränkt, zum Beispiel durch die zweite Ebene (Werkstückträgerplattform) oder durch die Stützen der zweiten Ebene. Etwas besser wäre die Möglichkeit den Sensor an die Unterseite der zweiten Ebene anzubringen. Auch hier schränken die Stützen und die Plattform die Sicht ein, jedoch ist der Winkel etwas besser, da der Sensor erhöhter platziert ist. Damit man das volle Potential des Sensors ausschöpfen kann, muss dieser auf der zweiten Ebene bzw. der Werkstückträgerplattform positioniert werden. Dort ist die Sicht nicht durch Stützen eingeschränkt. Da die Werkstücke in der Mitte des Roboters platziert werden, kann der Sensor nur an der Vorder- oder Rückkante der Plattform angebracht werden. Um eine möglichst gute Abdeckung zu erreichen, wurde sich für die Seite entschieden, die entgegengesetzt zu der Seite mit dem 2D-LIDAR liegt. Durch die Positionierung am Rand ergibt sich wieder eine Sichteinschränkung nach unten in Richtung der 'langen' Seite, also in Richtung 2D-LIDAR. Damit diese Einschränkung behoben wird, muss der 3D-LIDAR Sensor erhöht platziert werden. Die minimale Höhe h_{LIDAR}, auf welcher der Sensor angebracht werden muss, kann mithilfe der Bemessungen des Roboters und des LIDARs aus Anhang A berechnet werden. Relevant für die Berechnungen sind folgende Informationen:

- Länge Werkstückträgerplattform: $l_{\text{Plattform}} = 600$ mm
- Durchmesser 3D-LIDAR Sensor: $d_{\text{LIDAR}} = 103{,}3$ mm
- Höhe des optischen Zentrums des LIDAR: $h_{\text{LIDAR,Z}} = 37{,}7$ mm
- Abstand opt. Zentrum (horizontal, ab Sensormitte): $d_{\text{LIDAR,Z}} = 41{,}91$ mm
- Öffnungswinkel LIDAR (Vertikal, ab optischem Zentrum): $\alpha_{\text{LIDAR}} = 15^{\circ}$

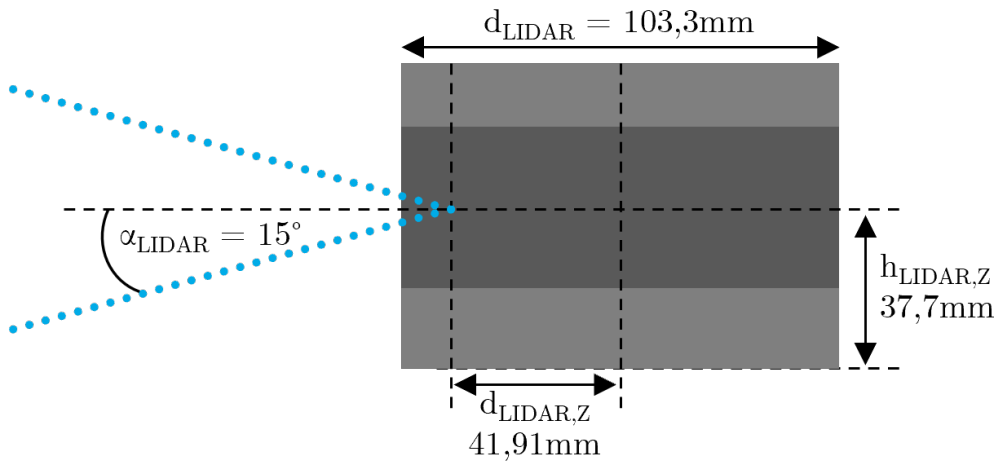


Abbildung 4.10.: Abmessungen Velodyne 3D-LIDAR [14]

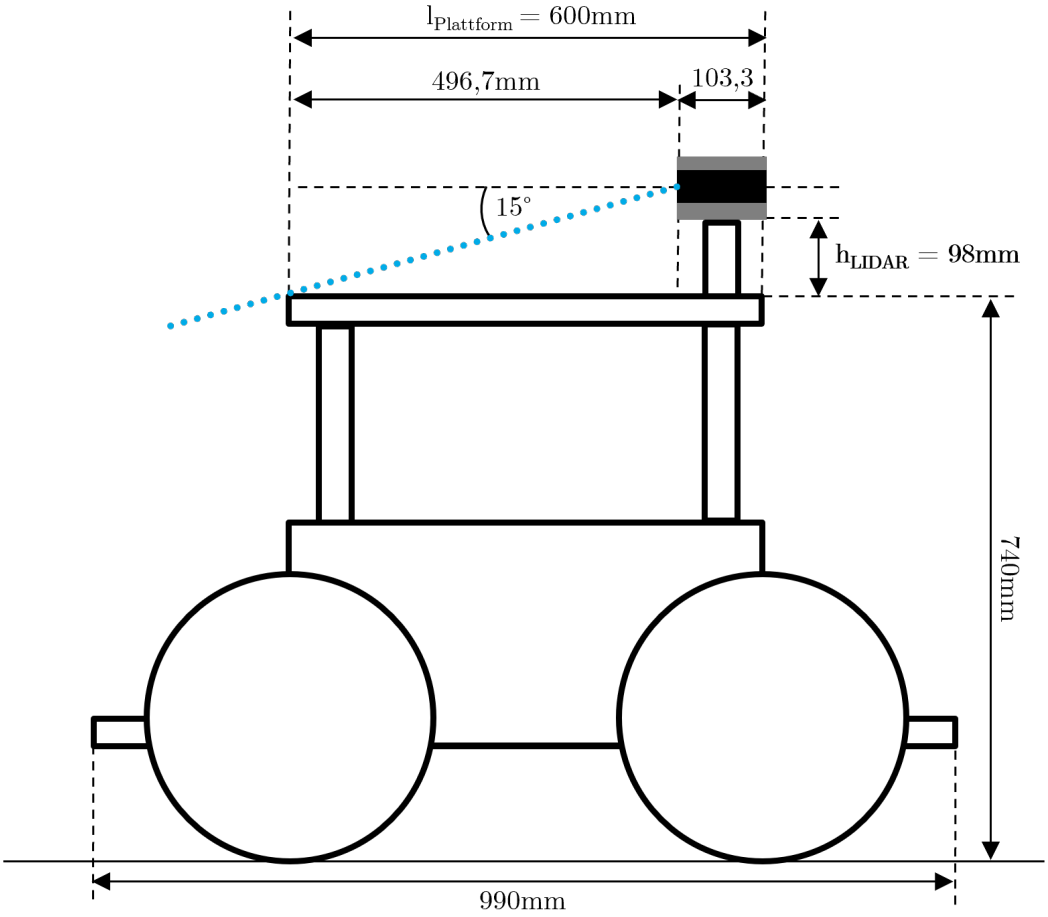


Abbildung 4.11.: Abmessungen Husky Roboter mit 3D-LIDAR [3]

Mithilfe der Zeichnung aus Abbildung 4.10 und 4.11 und trigonometrischer Berechnungen kann nun die minimale Sensorhöhe h_{LIDAR} berechnet werden:

$$\tan(\alpha_{\text{LIDAR}}) = \frac{h_{\text{LIDAR}} + h_{\text{LIDAR,Z}}}{l_{\text{Plattform}} - (\frac{d_{\text{LIDAR}}}{2} + d_{\text{LIDAR,Z}})} \tag{4.1}$$

$$h_{\text{LIDAR}} = \tan(\alpha_{\text{LIDAR}}) \cdot [l_{\text{Plattform}} - (\frac{d_{\text{LIDAR}}}{2} + d_{\text{LIDAR,Z}})] - h_{\text{LIDAR,Z}} \tag{4.2}$$

$$\rightarrow h_{\text{LIDAR}} \approx 98 \text{ mm} \tag{4.3}$$

Damit ergibt sich eine Höhe von mindestens 98 mm auf welcher der 3D-LIDAR Sensor platziert werden muss, um keine Sichteinschränkungen zu haben. Um Toleranzen mit zu berücksichtigen und einen Puffer einzubauen wird die Höhe des Sensor-Trägers auf 100 mm festgelegt.

In folgender Rechnung wird der mit Werkstücken beladene Zustand untersucht und berücksichtigt. Dafür wurden folgende für die Berechnung relevanten Größen nachgemessen:

- Abstand zum letzten Werkstückträger: $d_{\text{Träger}} = 445$ mm
- Werkstückhöhe: $h_{\text{Werkstück}} = 80$ mm

Eine reduzierte Zeichnung, welche die Werkstücke mit betrachtet, ist in Abbildung 4.12 dargestellt. Bei dieser Berechnung kann die bereits verwendete Formel 4.2 mit ein paar kleinen Anpassungen benutzt werden. So wird zuerst anstelle der Plattformlänge $l_{\text{Plattform}}$ der Abstand zum letzten Werkstück $d_{\text{Träger}}$ verwendet. Abschließend wird auf das Ergebniss noch die Werkstückhöhe $h_{\text{Werkstück}}$ hinzuaddiert:

$$h_{\text{LIDAR}} = \tan(\alpha_{\text{LIDAR}}) \cdot [d_{\text{Träger}} - (\frac{d_{\text{LIDAR}}}{2} + d_{\text{LIDAR,Z}})] - h_{\text{LIDAR,Z}} + h_{\text{Werkstück}} \tag{4.4}$$

$$\rightarrow h_{\text{LIDAR}} \approx 136{,}5 \text{ mm} \tag{4.5}$$

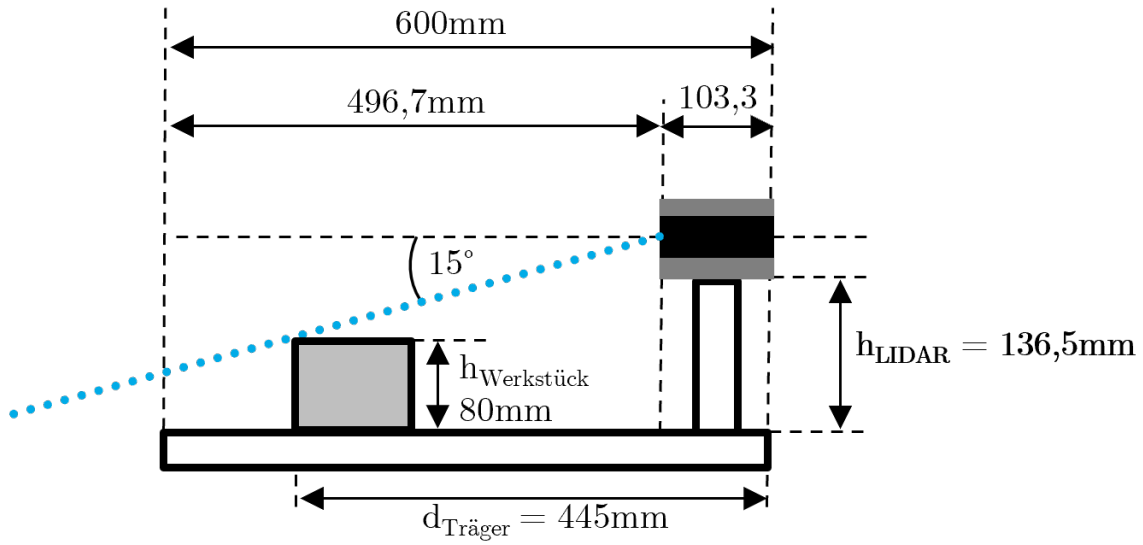


Abbildung 4.12.: Abmessungen Husky Roboter mit 3D-LIDAR, Werkstücke berücksichtigt

Damit ergibt sich eine Höhe von mindestens 136,5 mm auf welcher der 3D-LIDAR Sensor platziert werden muss, um keine Sichteinschränkungen auch mit beladenem Werkstück zu haben. Um Toleranzen mit zu berücksichtigen und einen Puffer einzubauen wird die Höhe des Sensor-Trägers auf 140 mm festgelegt. Diese Höhe kann / muss gegebenenfalls bei Veränderung der Werkstückträgerposition oder der Werkstückshöhe angepasst werden. Die theoretisch berrechnete Höhe des LIDAR muss praktisch noch getestet werden, bevor dieses fest auf dem Roboter montiert wird.

Mit dieser Festlegung kann schließlich auch die Distanz berechnet werden, bei welcher der Sensor den Boden frühestens erkennt. Dafür wird die Höhe des optischen Zentrum des LIDAR in Relation zum Boden bestimmt, indem die Höhe der Werkstückträgerplattform, die Höhe des LIDAR und die Höhe des optischen Zentrums des LIDAR addiert werden. Es folgt ein Wert von 917,7 mm. Mit dieser Höhe und Trigonometrie kann die Strecke d_{Boden} berechnet werden, bis der Sensor den Boden erkennt:

$$d_{\text{Boden}} = \frac{917,7mm}{\tan(\alpha_{\text{LIDAR}})} \approx 3425 \text{ mm} = 3,425 \text{ m} \tag{4.6}$$

Da der Sensor nicht mittig auf dem Roboter platziert wird, ist der Abstand zwischen Vorder- bzw. Rückseite und dem Punkt, an welchem der 3D-LIDAR den Boden er-

kennt, unterschiedlich. So ergibt sich für die Seite, auf welcher der 2D-LIDAR befestigt ist eine Strecke von etwa 2,7 m und auf der anderen Seite 3,2 m. Damit wird auch die Seite, an welcher der 2D-LIDAR befestigt ist zur Vorderseite bestimmt, da dort der Abstand bis der Sensor den Boden erkennt kleiner ist. Da diese Distanz relativ weit ist, können Objekte die direkt (1-2 m) vor dem Roboter sind nicht erkannt werden. Gegebenenfalls müssen hier andere Sensoren, wie zum Beispiel RADAR-Sensoren, nachgerüstet werden, um die Funktionsfähigkeit und Sicherheit (vor allem gegenüber anderen Personen oder deren Eigentum) zu gewährleisten.

ENDE STUDIENARBEIT 1

Ein wichtiger Teil des ganzen Projekts Zukunftsfabrik Bodensee ist die Kommunikation und Zusammenarbeit der verschiedenen Teams. So war es ein großer Teil der zweiten Studienarbeit sich Gedanken zu machen, wie schlussendlich das ganze Konzept aussehen soll. Eine wichtige Schnittstelle für den Husky ist selbstverständlich die Werkstückbeladung am Ende der Produktionslinie. In Absprache mit der dazugehörigen Gruppe wurde festgestellt, dass der Roboter, welcher die Werkstücke vom Band auf den Husky laden soll, in der aktuellen Konfiguration die Werkstückträger nicht erreichen kann. Folglich müssen die Werkstückträger, welche aktuell mittig auf der Plattform befestigt sind (siehe Abbildung 4.12) weiter zum Rand geschoben werden. Da auf der einen Seite der LIDAR Sensor befestigt werden soll, werden die Werkstückträger auf der anderen Seite der Plattform montiert. Durch diese Änderung muss die Höhe der Befestigung des LIDARs erneut berechnet werden, da die Werkstücke nun weiter weg vom Sensor sind und somit die Sicht versperren. Mit Hilfe der abgeänderten Skizze in Abbildung 4.13 kann die neue Befestigungshöhe mit Formel 4.7 berechnet werden. Die einzige Änderung zum vorherigen Fall ist das $d_{\text{Träger, neu}}$ nun 595mm beträgt, anstelle von 445mm.

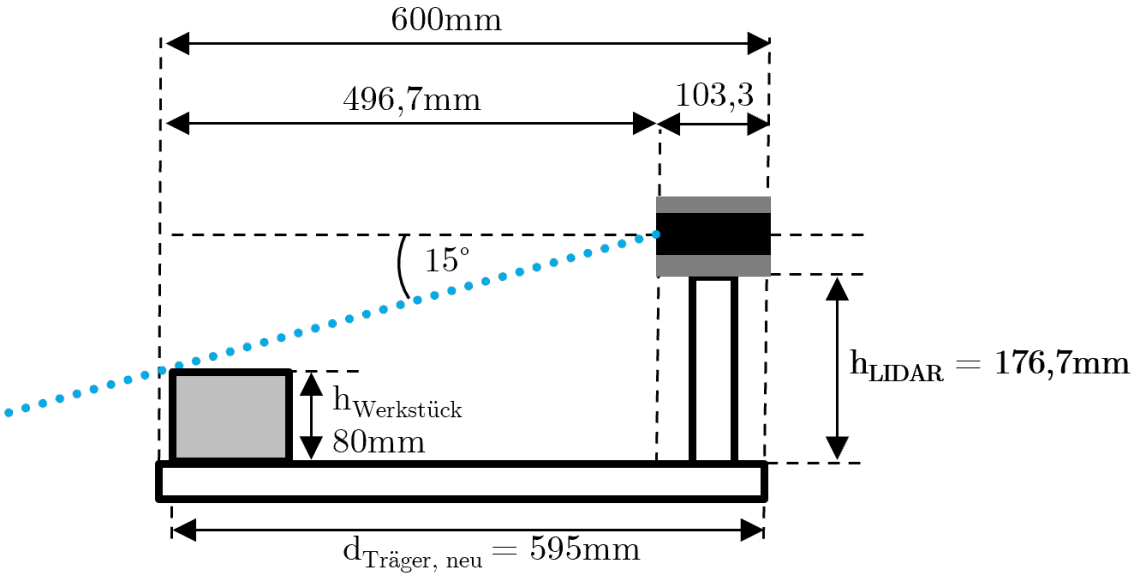


Abbildung 4.13.: Abmessungen Husky Roboter mit 3D-LIDAR, Werkstücke berücksichtigt + Umbau

$$h_{\text{LIDAR}} = \tan(\alpha_{\text{LIDAR}}) \cdot [d_{\text{Träger, neu}} - (\frac{d_{\text{LIDAR}}}{2} + d_{\text{LIDAR,Z}})] - h_{\text{LIDAR,Z}} + h_{\text{Werkstück}} \quad (4.7)$$

$$\rightarrow h_{\text{LIDAR}} \approx 176{,}7 \text{ mm} \quad (4.8)$$

Durch die Verschiebung muss die Halterung etwa 4cm erhöht werden. Mit Toleranzen wurde schlussendlich eine Halterungshöhe von 180mm bzw. 18cm gewählt.

4.3.3. Evaluierung der Halterungshöhe

Bevor die Halterung gebaut wird, muss die in der Theorie berechnete Höhe praktisch überprüft werden. Dafür wurde der LIDAR Sensor provisorisch auf eine Höhe von 18cm gestellt und die Sensordaten in RVIZ betrachtet. Wie aus Anhang A entnommen und in Abbildung 4.14 (links) zu sehen, sendet der Velodyne LIDAR in insgesamt 16 Elevationsebenen innerhalb von +15° und -15°. Um nun zu überprüfen, ob der LIDAR durch die Werkstücke verdeckt ist muss untersucht werden, ob die unterste Elevationsebene (1) den Boden detektiert oder das Werkstück.

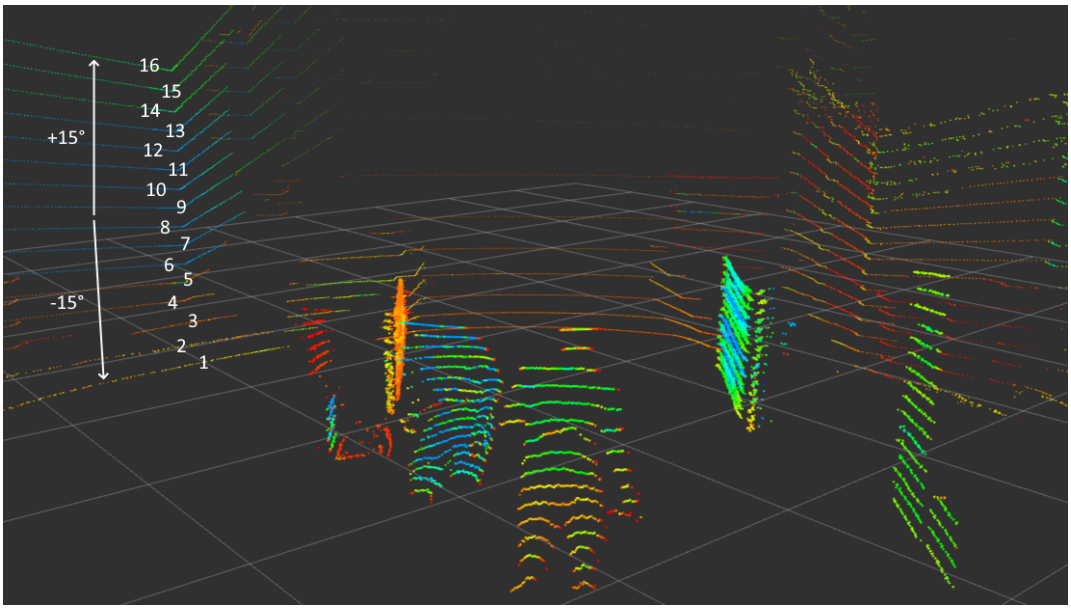


Abbildung 4.14.: 3D-LIDAR Daten in RVIZ - Erklärung Elevationsebenen

Dafür wurde in RVIZ die Kameraperspektive so geändert, dass man von oben zentriert auf den Roboter blickt und vorne die Seite mit den Werkstückträgern ist. In Abbildung 4.15 erkennt man, wie die unteren Elevationsebenen in einem Bogen geformt sind und dann seitlich in eine Linie verlaufen. Diese seitlichen Linien Stellen den Gang dar, in welchem das Experiment durchgeführt wurde. Gut erkennbar ist hier, dass die unterste Elevationsebene (1) vollständig abgebildet ist, was bedeutet, dass der LIDAR nicht von den Werkstücken verdeckt ist. Daraus folgt, dass eine Sensorhöhe von 18cm ausreichend ist, um keine blinden Flecke zu erzeugen. Um nun zu überprüfen, ob nicht eine zu große Sensorhöhe berechnet wurde, wird der Sensor auf eine Höhe von 17cm gestellt. Die dazugehörigen Sensordaten sind in Abbildung 4.16 dargestellt. Im Gegensatz zu Abbildung 4.15 erkennt man nun, dass die unterste Elevationsebene (1) an zwei Stellen unterbrochen wurde.



Abbildung 4.15.: 3D-LIDAR Daten in RVIZ - Sensorhöhe >= 18 cm



Abbildung 4.16.: 3D-LIDAR Daten in RVIZ - Sensorhöhe < 18cm

Dies ist auf die zwei Werkstücke zurückzuführen, welche den Sensor blockieren. Aufgrund der minimalen Sichtweite von rund einem Meter, werden die Werkstücke im Bild nicht detektiert. Aus dieser Untersuchung folgt, dass eine Sensorhöhe von 18cm ausreichend ist, um ein volles Sichtfeld zu haben, jedoch auch nicht zu hoch ist, damit der tote Winkel unnötig größer wird. Der tote Winkel beläuft sich auf die Distanz bis der Sensor den Boden detektiert, was in dieser Konfiguration etwa 3,57m beträgt. Im Vergleich zur vorherigen Konfiguration sind das etwa 15cm mehr. Mit der berechneten und evaluierten Sensorhöhe kann die Konstruktion der Halterung begonnen werden.

4.3.4. Planung und Konstruktion der Halterung für den 3D-LIDAR

In vorherigem Kapitel wurde die Frage geklärt, wo auf dem Roboter und vor allem wie hoch der Sensor platziert werden muss, um das Field of View (FOV) des LIDARs optimal zu nutzen. In diesem Kapitel wird nun der Frage "Wie" nachgegangen. Für diesen Prozess wurden im Vorfeld einige Requirements festgelegt:

- Die Halterung soll auf der Oberseite der Platte montiert werden
- Die Halterungshöhe soll 18cm betragen
- Die Halterung soll dafür sorgen, dass der Sensor stabil angebracht ist und im Wasser liegt
- Der Sensor soll recht einfach abnehmbar von der Halterung sein
- Auch die Halterung selbst soll einfach von der Oberseite des Huskys getrennt werden können, falls eine Anpassung der Position in Zukunft notwendig sein sollte

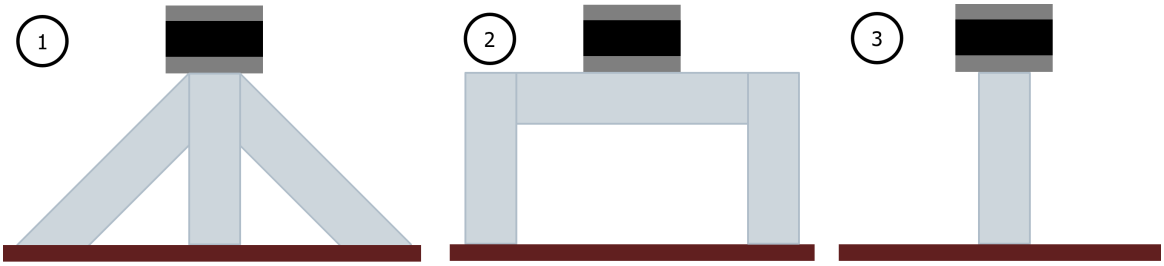


Abbildung 4.17.: Verschiedene Halterungsoptionen für LIDAR-Sensor

Um die Konstruktion der Halterung möglichst einfach und günstig zu halten, wurde versucht mit den Materialien, welche im PTZ zur Verfügung stehen, auszukommen. Deshalb wurden Aluprofile und passende Winkel als Grundmaterialien gewählt. Im weiteren Schritt wurden drei Konzepte ausgearbeitet, welche in Abbildung 4.17 dargestellt sind. Option 1 und 3 sehen vor den Sensor auf der Oberseite des Aluprofils zu befestigen, wobei Option 1 noch zusätzlich durch Aluprofile gestützt wird. Bei Option 2 hingegen, bildet ein Bogen die Grundlage der Halterung. Dabei soll der Sensor auf die lange Seite des Aluprofils befestigt werden. Die Wahl hing schlussendlich davon ab, ob es einfacher ist den Sensor auf der Oberseite oder auf der langen Seite zu montieren und ob eine zusätzliche Stütze zur Stabilisierung notwendig ist. Der Vorteil der Aluprofile ist, dass auf der Oberseite in die Mitte ein M12-Gewinde eingedreht werden kann. Dafür wurde ein manueller Gewindedreher verwendet. Da der LIDAR-Sensor

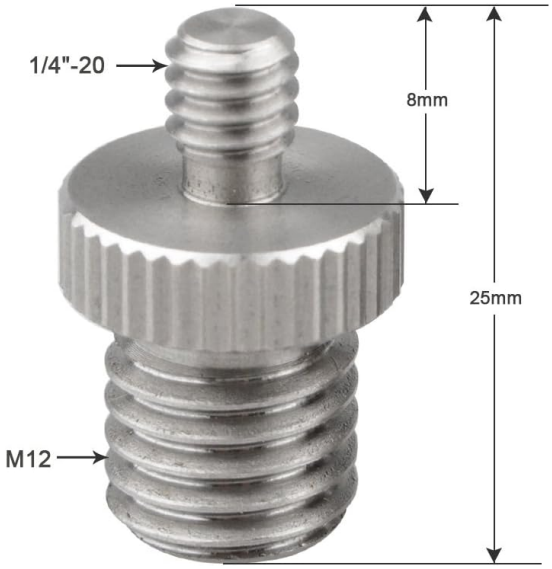


Abbildung 4.18.: Male-Male 1/4″ auf M12 Schraubadapter

ein 1/4″-Gewinde auf der Unterseite besitzt, kann dieser mithilfe eines Male-Male-Adapters mit dem Aluprofil verbunden werden. Der dafür verwendete Schraubadapter ist in Abbildung 4.18 dargestellt. Mit diesem Adapter können Option 1 und 3 realisiert werden. Option 2 wurde folglich aufgrund des Bauteilaufwands und Komplexität verworfen. Um die Entscheidung zwischen Option 1 und 3 zu treffen wurde beschlossen Option 3 zu bauen und ggf. falls diese nicht stabil genug sein sollte, um Option 1 zu erweitern. Für die Halterung wurde ein Stück Aluprofil auf 18cm zugesägt und auf eine Seite ein M12 Gewinde eingedreht. Mithilfe von zwei Winkeln, konnte die Halterung auf die Grundplatte des Roboters geschraubt werden. Damit die Halterung wirklich stabil angebracht ist, wurde diese mit Muttern von der Unterseite der Grundplatte angezogen. Eine Skizze des Halterungsaufbaus ist in Abbildung 4.19 dargestellt. In

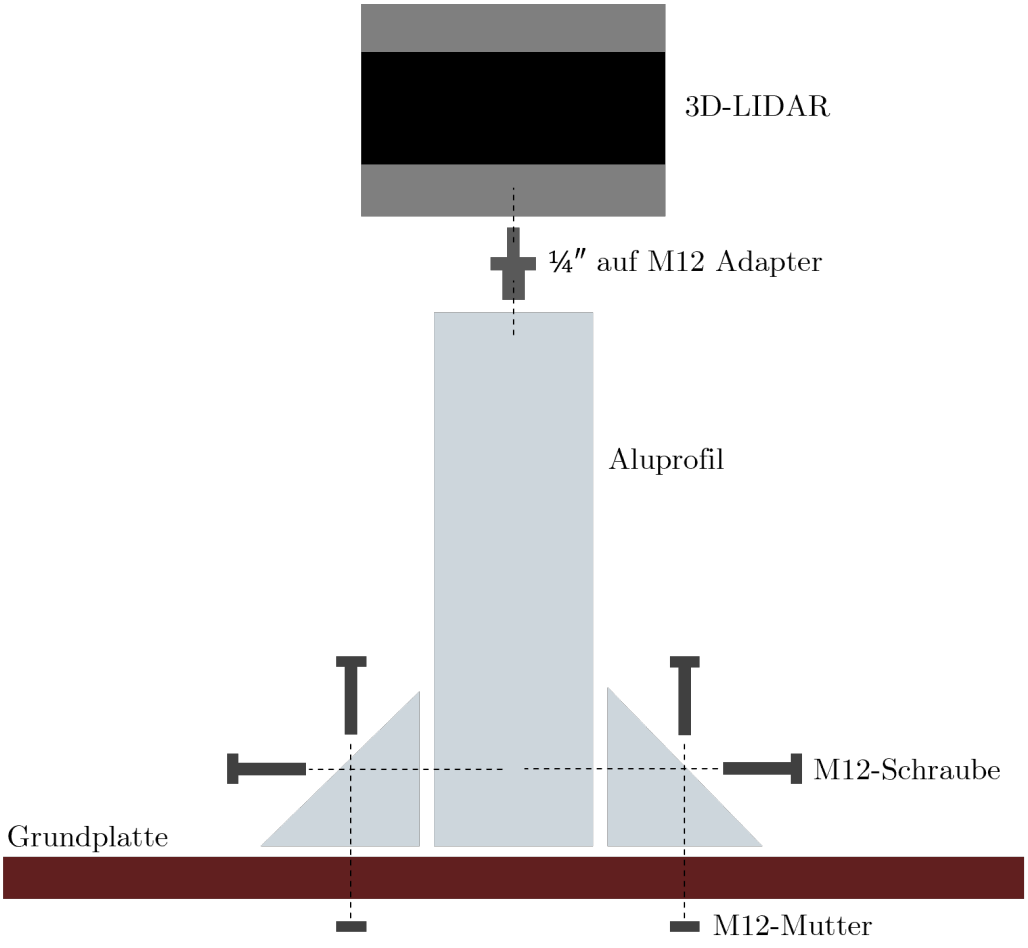


Abbildung 4.19.: Konstruktionsskizze der LIDAR-Halterung

Abbildungen 4.20 und 4.21 ist zu sehen, wie die Halterung Option 3 ("Rakete") auf der Grundplatte des Roboters befestigt ist. Um zu überprüfen, ob die Halterung fest genug wurde an dieser gewackelt, sowie mit dem Husky über unebenes Gelände gefahren. Abschließend konnte festgestellt werden, dass Option 3 ausreichend fest und stabil ist und keine zusätzlichen Stützen benötigt werden. Abschließend kann man sagen, dass die Halterung trotz ihres simplen Aufbaus sehr stabil und ausreichend für die Befestigung des LIDARs ist. Einziger (optischer) Nachteil ist, dass der Anschluss für das Datenkabel bzw. der Velodyne-Schriftzug nicht mittig ist, was dem Gewinde geschuldet ist. Falls man dies beheben wollen würde, müsste man das Gewinde im Aluprofil an einer bestimmten Position anfangen zu drehen, damit der Schraubadapter an der Position fest ist, an welcher der Schriftzug genau nach vorne zeigt.

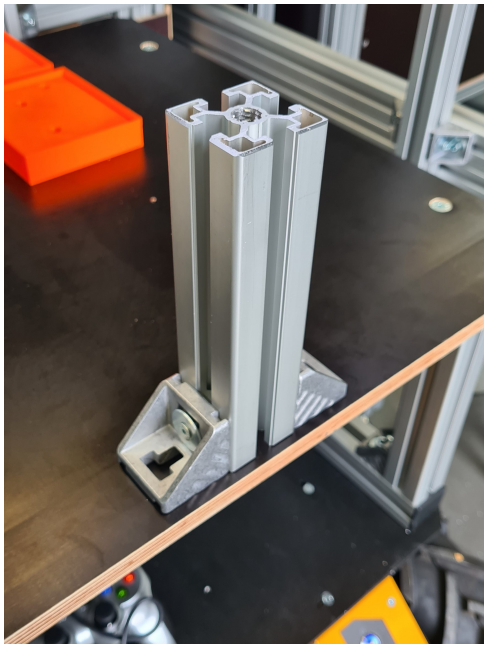


Abbildung 4.20.: Halterung Option 3

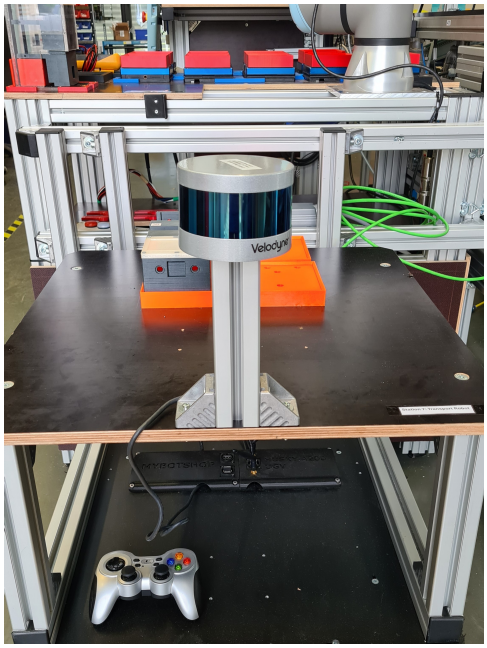


Abbildung 4.21.: Halterung Option 3 mit LIDAR

4.3.5. Elektrische Integration des LIDARs in das Robotersystem

Um den 3D-LIDAR unabhängig von einer externen Stromversorgung zu machen, musste der Sensor an die Batterie des Huskys angeschlossen werden. Da es von vorhanden Dokumenten und Skizzen nicht ersichtlich war, was der Husky für Anschlüsse bezüglich Stromversorgung besitzt, musste dies erst einmal herausgefunden werden. Dazu wurde zuerst das komplette obere Teil des Roboters abmontiert. In Abbildung 4.22 ist ein Bild im auseinandergebauten Zustand dargestellt.

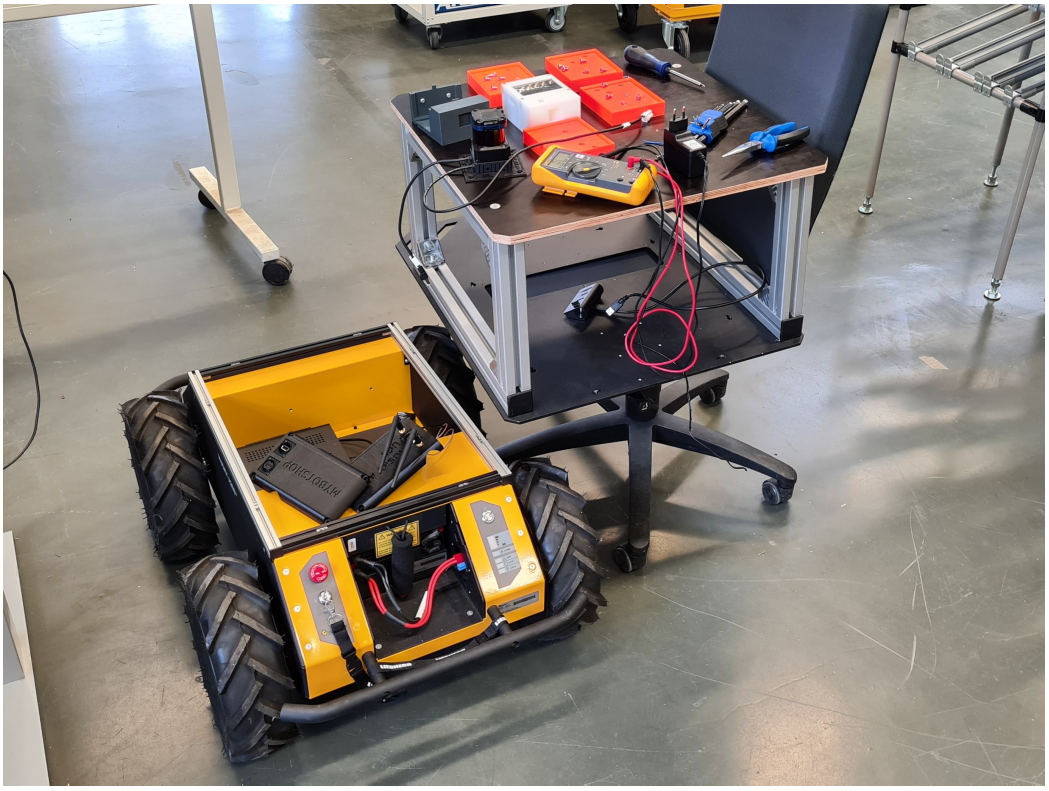


Abbildung 4.22.: Husky mit demontierter Oberseite

Innerhalb des Huskys befindet sich, wie in Abbildung 4.23 erkennbar, der Computer, auf welchem das Betriebssystem und sämtliche Programme z.B. zur Steuerung des Huskys laufen. Die Anschlüsse des Computers sind in Abbildung 4.24 dargestellt. Links wird der Computer mit Strom von der Roboterbatterie versorgt. Dann gibt es einige USB-Anschlüsse worüber z.B. der Kompass und der USB-Hub angeschlossen

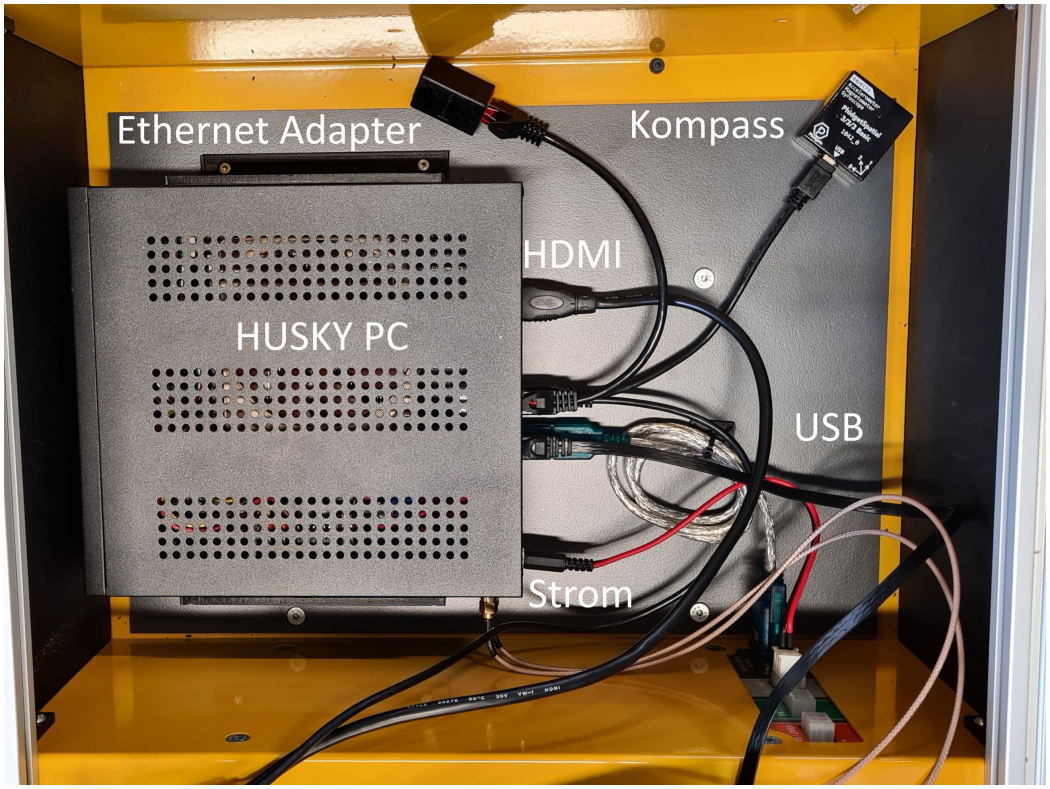


Abbildung 4.23.: Innenraum des Husky

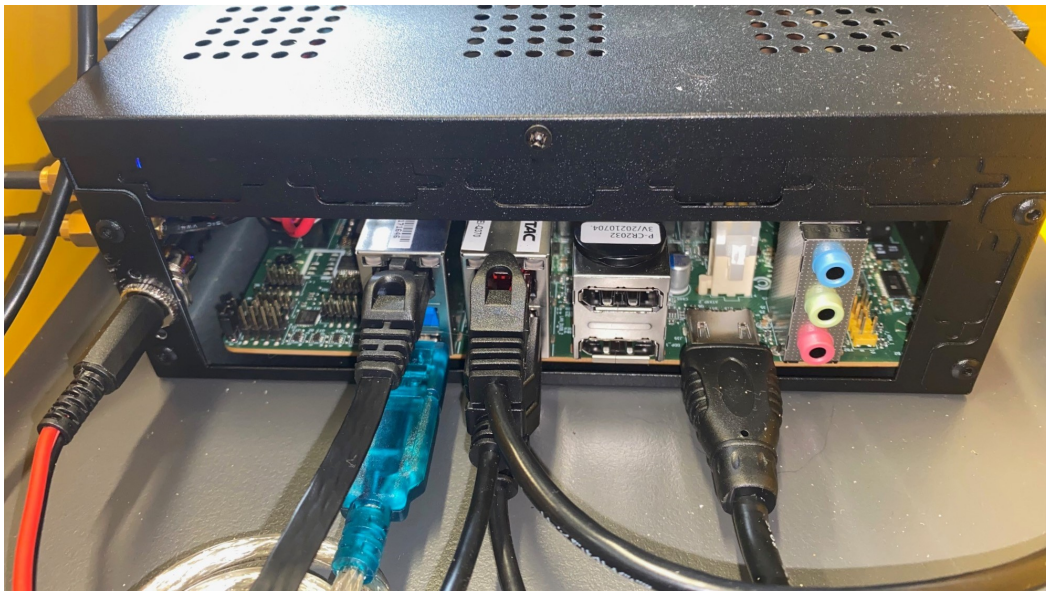


Abbildung 4.24.: Anschlüsse des Husky-Computers

ist. Zudem gibt es zwei Ethernet Anschlüsse, von welchen einer an den 2D-LIDAR Sensor angeschlossen ist. Zuletzt ist der Computer auf der rechten Seite mit einem HDMI-Kabel verbunden, welches zum Monitor führt. Um die Daten des 3D-LIDARs an den Computer zu übertragen, müssen diese über den freien Ethernet-Port miteinander verbunden werden. Für die Stromversorgung des LIDARs sind die Stromanschlüsse des Huskys interessant, welche in Abbildung 4.25 dargestellt ist. Für die bessere Anschauung wurde in Abbildung 4.26 eine Übersicht der Pinbelegung erstellt. Es gibt

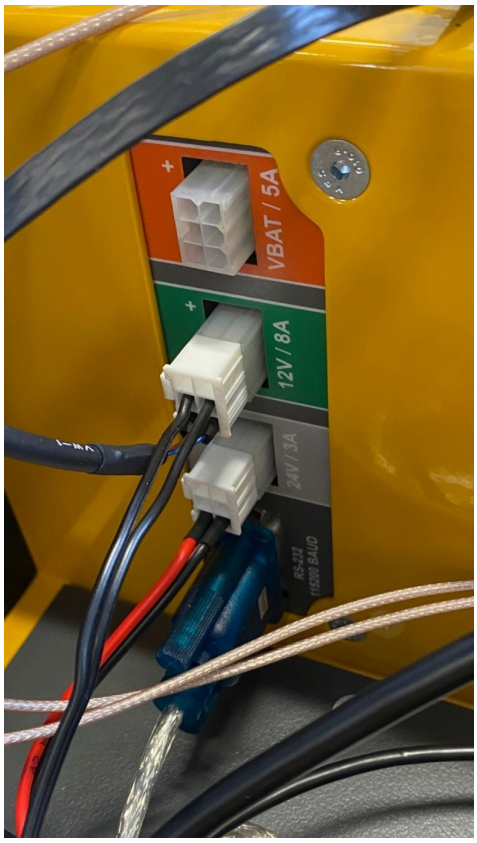


Abbildung 4.25.: Bild der Pinbelegung

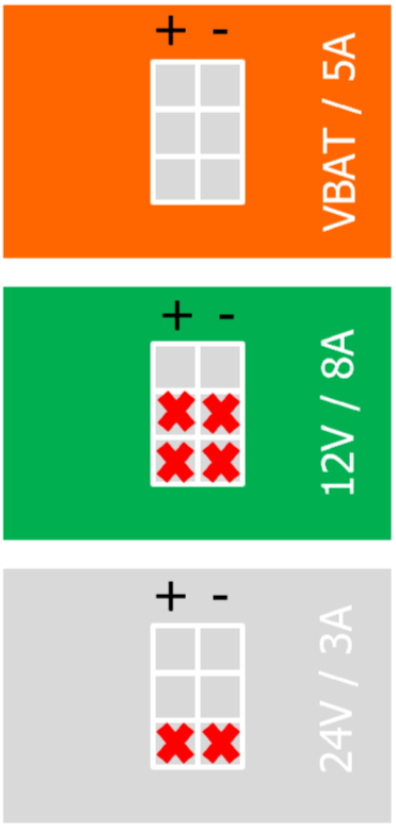


Abbildung 4.26.: Pinbelegung der Roboterbatterie

drei verschiedene Anschlüsse mit verschiedenen Spannungs- und Strombereichen. An jedem Anschlüssen können bis zu drei Geräte angeschlossen werden. Am unteren Anschluss, welcher mit 24V versorgt ist, ist der Husky Computer angeschlossen. Am mittleren Grünen Anschluss, welcher mit 12V versorgt ist, ist der 2D-LIDAR Sensor angeschlossen. Da der 3D-LIDAR von Velodyne auch mit 12V betrieben werden muss, konnten die dazugehörigen Kabel abisoliert, gecrimpt und in den bereits vorhandenen

Stecker eingesteckt werden. Abschließend wurde die dazugehörige Velodyne Interface Box noch im Innenraum verstaut und festgeklebt, damit diese nicht beschädigt wird (siehe 4.27). Somit führt nur das Datenkabel aus der Interface Box von Innenraum nach außen zum 3D-LIDAR auf der Oberseite. In Zukunft kann am 12V Anschluss noch ein weiter 2D-LIDAR versorgt werden, welcher bereits angeschafft wurde und sich auf dem stationären Arbeitsplatz in einem Paket befindet.

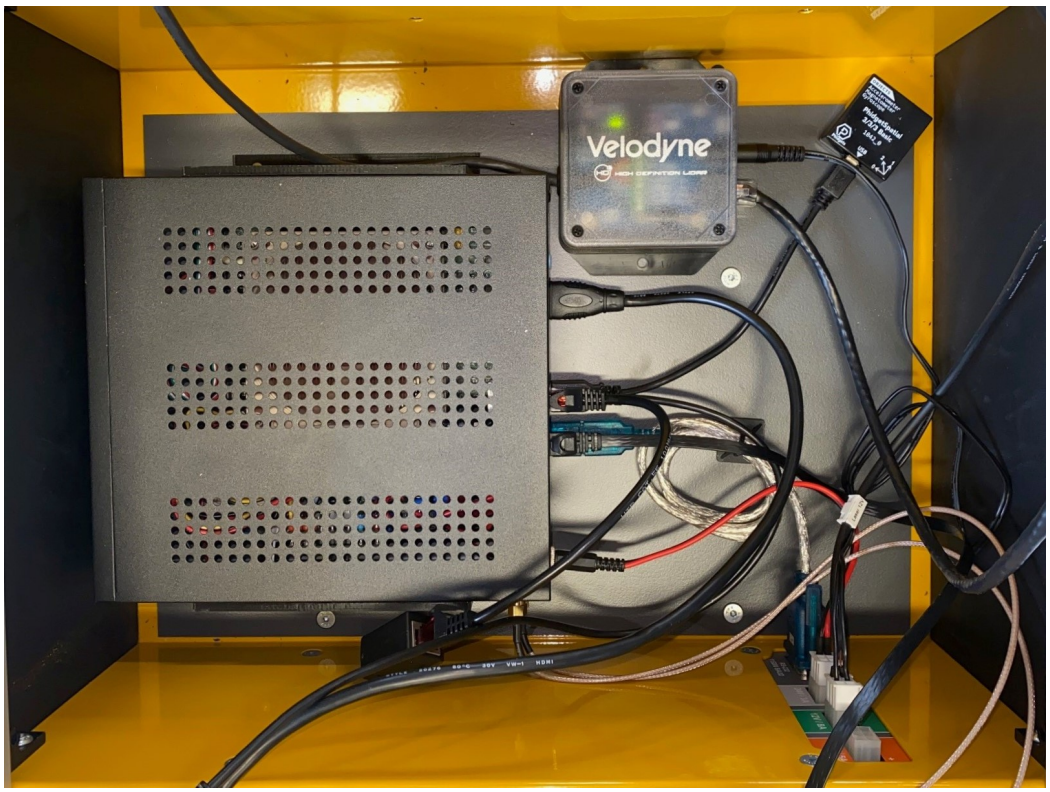


Abbildung 4.27.: Innenraum des Husky mit Velodyne Interface Box

4.4. Planung und erste Umsetzungen der mobilen Arbeitsstation

Ziel der mobilen Arbeitsstation ist die Unabhängigkeit von externen Stromversorgungen bzw. ortsfesten Geräten. Ein Monitor zum Anzeigen der Sensordaten ist essenziell, um mit dem Husky im Außenbereich zu interagieren. Um den Monitor unabhängig von der Steckdose zu machen, gibt es prinzipiell zwei Möglichkeiten:

1. Der Monitor besitzt einen eigenen integrierten Akku, welcher nach einer bestimmten Entladezeit an der Steckdose wieder aufgeladen werden muss

2. Der Monitor wird von der Batterie des Huskys versorgt

Der große Vorteil der zweiten Möglichkeit ist, dass durch die Stromversorgung des Monitors über die Roboterbatterie lediglich ein Ladezustand zu überwachen ist. Da es relativ einfach ist einen weiteren Verbraucher an die Batteriespannung zu klemmen empfiehlt sich Möglichkeit zwei. Die Batterie des Huskys hält lang genug, um eine Weile mit ihm zu arbeiten und das Anschließen eines Monitors sollte diese Zeit nicht wesentlich verkürzen. Ein Monitor, der diesen Anforderungen entspricht findet sich im Anhang B "Quick Start Guide" im Kapitel B.6. Es handelt sich hierbei um einen industriellen Monitor von dem deutschen Unternehmen beetronics und hat folgende Vorteile für diese Anwendung:

- Verschiedene Monitorgrößen zwischen 7 und 27 Zoll. Dies ist von Vorteil, da so eine passende Größe für die gewählte Position gefunden werden kann

- Einfache und flexible Montage entweder über integrierten Standfuß oder über 100mm-VESA-Halterung

- Schutzart nach IP65 also Spritzwassergeschützt, das heißt die Anwendung bei leichtem Regen ist kein Problem (generell empfiehlt sich es nicht den Husky bei Starkregen zu betreiben)

- Variabler Eingangsspannungsbereich zwischen 9-33V, kann also sehr gut an den freien 24V-Anschluss angesteckt werden

- Kompatibel mit Linux Systemen

- Videosignal kann über HDMI oder DisplayPort empfangen werden, welche beide am Husky Computer vorhanden sind. Auch USB-C und VGA wären möglich

- Touchscreen-Variante möglich. Damit lässt sich schneller Arbeiten und es sind keine Tastatur und Maus erforderlich. Für längeres Arbeiten empfiehlt sich natürlich eine Tastatur und eine Maus, welche als Bluetooth Option gewählt werden und über den USB-Hub verbunden werden können

- Gehärtetes Glas und Metallgehäuse, um sicher im Outdoorbereich verwendet zu werden

Des Weiteren soll der Monitor stabil an eine Halterung befestigt werden und flexibel sein, sodass er nicht sperrig bzw. im Weg ist und je nach Bedarf rotiert werden kann. Bei der Halterung gibt es viele verschiedene Möglichkeiten, je nach Position des Monitors oder Befestigungsart. Eine Möglichkeit wäre zum Beispiel direkt eine Halterung zu finden, welche an den Alu-Profilen festgeschraubt werden kann. Eine solche Halterung findet man auch im Anhang B "Quick Start Guide" im Kapitel B.6. Die Halterungen unterscheiden sich meist in den möglichen Freiheitsgraden, also sollte man sich zuvor Gedanken, wo der Monitor später einmal positioniert werden soll. Die sinnvollste Variante, bei welcher wenig Platz verwendet wird und auch die Sensoren nicht blockiert werden, ist den Monitor im Zwischenbereich zu verstauen. Mit einer flexiblen Halterung kann der Monitor dann ausgefahren und höher gestellt werden, damit angenehmeres Arbeiten möglich ist. Am besten sollten die Halterungen 100mm-VESA kompatibel sein (ist ein bekannter Standard, sollte also kein Problem sein) damit der Bildschirm von beetronics ohne Probleme befestigt werden kann. Letztlich bedarf es noch an einer Möglichkeit zum Anschließen einer Maus, Tastatur und dem Bluetooth-Dongle für den Kontroller. Der Husky hat nach außen nur einen freien USB-Port, wodurch ein USB-Hub notwendig wurde. Die Anforderungen an diesen waren recht simpel: mindestens drei USB-Ports und eine Möglichkeit den USB-Hub stabil auf dem

Husky zu befestigen. Für diese Anwendung wurde der "ORICO USB-Hub Clip-Typ" gewählt. Dieser kann mithilfe einer Schraubklemme auf jeder Oberfläche fest montiert werden und besitzt 4 USB-Ports. Der USB-Hub wurde vorerst auf der Seite der Grundplatte montiert, kann aber bei Bedarf an eine andere Position gesetzt werden. Der montierte USB-Hub ist in Abbildung 4.28 dargestellt.



Abbildung 4.28.: Montierter USB-Hub an der Oberseite des Husky

5. Zusammenfassung

Um auf die in der Einleitung gesetzten Ziele zurückzukommen, kann gesagt werden, dass diese erfolgreich umgesetzt werden konnten. Es wurde eine stationäre Arbeitsoberfläche errichtet, um mit dem Transportroboter interagieren zu können. Diese Arbeitsoberfläche besteht momentan noch aus einem Monitor mit externer Stromversorgung und Kabelverbindung zum Roboter. Zusätzlich können diverse Befehle (Husky stoppen und neu starten) über einen separaten Computer / Laptop kabellos ausgeführt werden, wenn sich dieser und der Husky im selben WLAN befinden. Um später im Outdoor-Bereich Sensordaten zu analysieren, wird ein mobiles Arbeitsplatzkonzept benötigt. Dafür wurde ein Konzept erstellt, welches einen mobilen Monitor und eine Halterung beinhaltet, die auf dem Roboter montiert werden. Zudem wurde bereits ein USB-Hub festmontiert, an welchem man z.B. eine Maus und Tastatur anschließen kann. Aus Budgetgründen wurde der Monitor und die Halterung noch nicht angeschafft, dies wird dann eine Aufgabe für kommende Studienarbeiten. Der neue Velodyne 3D-LIDAR Sensor wurde erfolgreich in Betrieb genommen. Darunter zählt das die elektrische Inbetriebnahme, was die Anbindung an das interne Stromnetztes des Huskys beinhaltet. Die softwaretechnische Inbetriebnahme konnte so weit umgesetzt werden, dass die Sensordaten im Visualisierungstools RVIZ als Punktwolke angezeigt werden konnten. In Zukunft müssen die 3D-LIDAR Daten in das bereits vorhandene Husky-Skript integriert werden, in welchem die 2D-LIDAR Daten im Verhältnis zu Sensor- bzw. Husky-Position angezeigt werden. Um das optimale Sichtfeld des 3D-LIDARs zu gewährleisten, muss der Sensor erhöht auf einer Halterung platziert werden. Um dessen Höhe zu ermitteln, wurden diverse Berechnungen durchgeführt. Diese Höhe wurde anschließend experimentell mit Hilfe von RVIZ überprüft. Zudem wurden verschiedene Halterungskonzepte ausgearbeitet und eine Halterungsoption gebaut, montiert und evaluiert. Abschließend wurde ein "Quick Start Guide" erstellt, mit

welchem zukünftige Studienarbeitsgruppen schneller in das Projekt einsteigen können. Dort wird auch ein kurzer Ausblick auf die zukünftigen Themen gegeben, die in diesem Projekt anstehen. Darunter zum Beispiel die Implementierung des SLAM-Algorithmus auf Basis der LIDAR Sensordaten. Mit diesem können anschließend erste Experimente zur autonomen Navigation durchgeführt werden. Ein weiterer großer Punkt wird die Kommunikation mit den anderen Gruppen der Zukunftsfabrik Bodensee sein, vor allem im Bereich Werkstückbeladung. Dort muss eine Kommunikationsschnittstelle hergestellt werden, welche dem Beladungsroboter sagt, wann der Husky bereitsteht, um beladen zu werden. Abschließend kann man sagen, dass in diesen beiden Studienarbeiten wichtige Grundlagen in vielen Bereichen gelegt worden ist, um das Projekt Transportroboter voranzubringen.

6. Review der Studienarbeiten

Dieses Kapitel soll eine kurze Einschätzung und Rückblick der beiden Studienarbeiten aufzeigen. Da die Möglichkeiten der Zeitinvestition aufgrund der vollen Vorlesungspläne stark eingeschränkt waren, war es schwierig im Vorhinein einschätzen zu können, was alles in der Zeit machbar zu erledigen ist. Grundlegend war das Ziel den Roboter in Betrieb zu nehmen und die internen Oberflächen kennen zu lernen. Die Ordnerstruktur der Software ist komplex, wodurch ein großer Zeitaufwand erforderlich ist, um hier wirklich Fortschritte zu machen. Zusätzlich zu der Zeit wäre es eine große Hilfe gewesen, sich von Beginn an mit MyBotShop zusammenzusetzen und an den Softwarethemen zu arbeiten. Unser Versuch über das Forum Kontakt aufzunehmen hat zwar funktioniert, jedoch war die Antwort eher kryptisch, wodurch weitere Erklärungen notwendig gewesen wären. Ein wichtiger, aber auch zeitintensiver Schritt war die Auseinandersetzung mit der GUI für die Darstellung der Sensorwerte. Dies war notwendig, um den 3D-LIDAR in Betrieb nehmen zu können. Hierbei war es entscheidend, an welcher Position bzw. Höhe dieser angebracht wird und wie er elektrisch versorgt wird. Hier war im Vorhinein wenig darüber bekannt, wie die interne Struktur des Huskys aussieht und welche Möglichkeiten dieser bietet. Des Weiteren lag ein besonderes Augenmerk darauf, den nachfolgenden Gruppen eine möglichst einfache Einarbeitung bzw. Einstieg zu ermöglichen, um ihnen unnötiges recherchieren zu ersparen. Mit Abbildung 6.1 wollen wir unsere Studienarbeit zusammenfassen, worauf der Husky mit fertig montiertem und integriertem 3D-LIDAR Sensor im Outdoorbetrieb zu sehen ist. Wir persönlich sind zufrieden mit dem, was wir mit gegebener Zeit erreichen konnten. Gerne hätten wir es geschafft, den Roboter das erste Mal autonom fahren zu lassen, jedoch haben wir schnell gemerkt, dass dies komplizierter ist als wir anfangs dachten. Generell würden wir sagen, dass die Studienarbeit Transportroboter einer der anspruchsvolleren Arbeiten ist, aber am Ende hat man auch ein sehr inter-

essantes und anschauliches Ergebnis. An dieser Stelle auch noch einmal vielen Dank an unseren Betreuer Peter Buckel, der uns mit regelmäßigen Meetings fachlich und organisatorisch unterstützt hat.



Abbildung 6.1.: Husky auf dem Gelände der DHBW

Literaturverzeichnis

[1] INRAY. *Der Weg von Industrie 1.0 zu Industrie 4.0*. URL: https://www.inray.de/aktuelles/der-weg-von-industrie-1-0-nach-industrie-4-0 (besucht am 05. 01. 2024).

[2] DHBW Ravensburg. *Zukunftsfabrik Bodensee*. URL: https://www.ravensburg.dhbw.de/forschung-transfer/kompetenzzentren/zentrum-fuer-digitalisierung-in-produktion-und-produktentwicklung#zukunftsfabrik-bodensee (besucht am 05. 01. 2024).

[3] Clearpath Robotics. *HUSKY - UNMANNED GROUND VEHICLE*. URL: https://clearpathrobotics.com/husky-unmanned-ground-vehicle-robot/ (besucht am 05. 01. 2024).

[4] MYBOTSHOP. *CLEARPATH HUSKY A200*. URL: https://www.mybotshop.de/Clearpath-Husky-A200 (besucht am 05. 01. 2024).

[5] Wakeup. *Linux: Was es ist und wozu es gut ist*. 2022. URL: https://www.grundlagen-computer.de/magazin/linux-was-es-ist-und-wozu-es-gut-ist (besucht am 08. 01. 2024).

[6] Redaktion Computerwissen. *Ubuntu: Funktionen, tipps und tricks Zur Linux-Distributionen*. 2020. URL: https://www.computerwissen.de/linux/distributionen/ubuntu/ (besucht am 08. 01. 2024).

[7] IONOS Redaktion. *Ubuntu vs. debian: Die beiden linux-distributionen im Vergleich*. 2023. URL: https://www.ionos.de/digitalguide/server/knowhow/ubuntu-vs-debian/ (besucht am 08. 01. 2024).

[8] Eske Gobes. *Ros - robot operating system*. 2023. URL: https://www.reichelt.de/magazin/ratgeber/wie-ros-die-robotik-vorantreibt/ (besucht am 08. 01. 2024).

[9] NavViz. *Alles, was Sie über SLAM und mobiles Mapping wissen müssen*. URL: https://www.navvis.com/de/technology/slam (besucht am 17. 11. 2023).

[10] FLYABILITY. *WHAT IS SIMULTANEOUS LOCALIZATION AND MAPPING (SLAM)?* URL: https://www.flyability.com/simultaneous-localization-and-mapping (besucht am 17. 11. 2023).

[11] Laserscanning Europe GmbH. *Was ist SLAM?* URL: https://www.laserscanning-europe.com/de/was-ist-slam (besucht am 17. 11. 2023).

[12] Datei.wiki. *SLAM erforschen: Ein umfassender Blick*. URL: https://datei.wiki/definition/slam-erforschen-ein-umfassender-blick/ (besucht am 17. 11. 2023).

[13] HOKUYO. *Product details - UTM-30LX-EW*. URL: https://www.hokuyo-aut.jp/search/single.php?serial=170 (besucht am 08. 01. 2024).

[14] Velodyne Lidar. *Puck*. URL: https://velodynelidar.com/products/puck/ (besucht am 08. 01. 2024).

[15] URL: https://de.mathworks.com/discovery/lidar.html (besucht am 08. 01. 2024).

[16] 2023. URL: https://de.wikipedia.org/wiki/Elektromagnetisches_Spektrum (besucht am 08. 01. 2024).

[17] Sohail. *[quick guide] husky: Hokuyo*. 2023. URL: https://forum.mybotshop.de/t/quick-guide-husky-hokuyo/622 (besucht am 08. 01. 2024).

[18] ROS.ORG. *Getting Started with the Velodyne VLP16*. URL: https://wiki.ros.org/velodyne/Tutorials/Getting%20Started%20with%20the%20Velodyne%20VLP16 (besucht am 08. 01. 2024).

Abkürzungsverzeichnis

2D / 3D	Zweidimensional / Dreidimensional
etc.	et cetera (und die übrigen Dinge)
FOV	Field of View
GPS	Global Positioning System
HMI	Benutzerschnittstelle (engl. Human Machine Interface)
IoT	Internet of Things
IWT	Institut für Weiterbildung, Wissens- und Technologietransfer
KI	Künstliche Intelligenz
KMU	Kleine und mittlere Unternehmen
LAN	Local Area Network
LIDAR	Light Imaging, Detection And Ranging
MacOS	Media Access Control Operating System
PTZ	Produktionstechnisches Zentrum
RADAR	RAdio Detection And Ranging
RITZ	Regionales Innovations- und Technologietransfer Zentrum
ROS	Robot Operating System
SLAM	Simultaneous Localization and Mapping
USB	Universal Serial Bus
WLAN	Wireless Local Area Network
z.B.	zum Beispiel
ZDP	Zentrum für Digitale Produktion

Symbolverzeichnis

Symbol	Beschreibung	Einheit
α_{LIDAR}	Öffnungswinkel 3D-LIDAR (Vertikal, ab optischem Zentrum)	°
d_{LIDAR}	Durchmesser 3D-LIDAR Sensor	mm
$d_{\text{LIDAR,Z}}$	Abstand optisches Zentrum (horizontal, ab Sensormitte)	mm
d_{Boden}	Abstand bis Sensor Boden erkennt	mm
$d_{\text{Träger}}$	Abstand zum letzten Werkstückträger	mm
h_{LIDAR}	(min.) Höhe des 3D-LIDAR Sensor	mm
$h_{\text{LIDAR,Z}}$	Höhe des optischen Zentrums des 3D-LIDAR	mm
$h_{\text{Werkstück}}$	Werkstückhöhe	mm
$l_{\text{Plattform}}$	Länge Werkstückträgerplattform	mm

Abbildungsverzeichnis

2.1. Roboter 'Husky' der Firma Clearpath Robotics [4] 6
2.2. Husky mit Erweiterung aus vorheriger Studienarbeit 7
2.3. Grundprinzip der Positionserkennung des Roboters durch Bestimmung des Abstands zu identifizierten Landmarks, eigene Darstellung 11
2.4. 2D-LIDAR von HOKUYO [13] 13
2.5. 3D-LIDAR von Velodyne [14] 13
2.6. LIDAR-Punktwolke einer Straßenszene [15] 14
2.7. Übersicht des elektromagnetischen Spektrums [16] 14

4.1. Eigenschaften der Netzwerkschnittstelle 'wlp2s0' 19
4.2. Konfigurierte Übersicht von MYBOTSHOP 20
4.3. Benutzeroberfläche von RViz mit bereits sichtbaren 2D-LIDAR Daten . 22
4.4. Punktwolke der LIDAR-Daten nach rotatorischer Bewegung gegenüber Abbildung 4.3 23
4.5. Aufbau der physikalischen Verbindung des 3D-LIDAR [14] [3] 25
4.6. Einstellungen für die Konfiguration der IP-Adresse des Computers . . . 26
4.7. IP-Adresse des LIDARs zum Nachweise einer erfolgreichen Verbindung zwischen Computer und Sensor 27
4.8. Benutzeroberfläche RViz zur Darstellung der 3D-LIDAR Daten 30
4.9. Echtzeitdaten des 3D-LIDARs dargestellt in RViz 30
4.10. Abmessungen Velodyne 3D-LIDAR [14] 32
4.11. Abmessungen Husky Roboter mit 3D-LIDAR [3] 32
4.12. Abmessungen Husky Roboter mit 3D-LIDAR, Werkstücke berücksichtigt 34
4.13. Abmessungen Husky Roboter mit 3D-LIDAR, Werkstücke berücksichtigt + Umbau 36
4.14. 3D-LIDAR Daten in RVIZ - Erklärung Elevationsebenen 37

4.15. 3D-LIDAR Daten in RVIZ - Sensorhöhe >= 18 cm 38
4.16. 3D-LIDAR Daten in RVIZ - Sensorhöhe < 18cm 38
4.17. Verschiedene Halterungsoptionen für LIDAR-Sensor 40
4.18. Male-Male 1/4″ auf M12 Schraubadapter 40
4.19. Konstruktionsskizze der LIDAR-Halterung 41
4.20. Halterung Option 3 42
4.21. Halterung Option 3 mit LIDAR 42
4.22. Husky mit demontierter Oberseite 43
4.23. Innenraum des Husky 44
4.24. Anschlüsse des Husky-Computers 44
4.25. Bild der Pinbelegung 45
4.26. Pinbelegung der Roboterbatterie 45
4.27. Innenraum des Husky mit Velodyne Interface Box 46
4.28. Montierter USB-Hub an der Oberseite des Husky 49

6.1. Husky auf dem Gelände der DHBW 53

B.1. Grundlegende Beschriftung des Huskys 65
B.2. An Batterie angeschlossen 66
B.3. An Netzteil angeschlossen 66
B.4. Kontrollerbelegung zum Steuern des Husky 68

A. Datenblätter

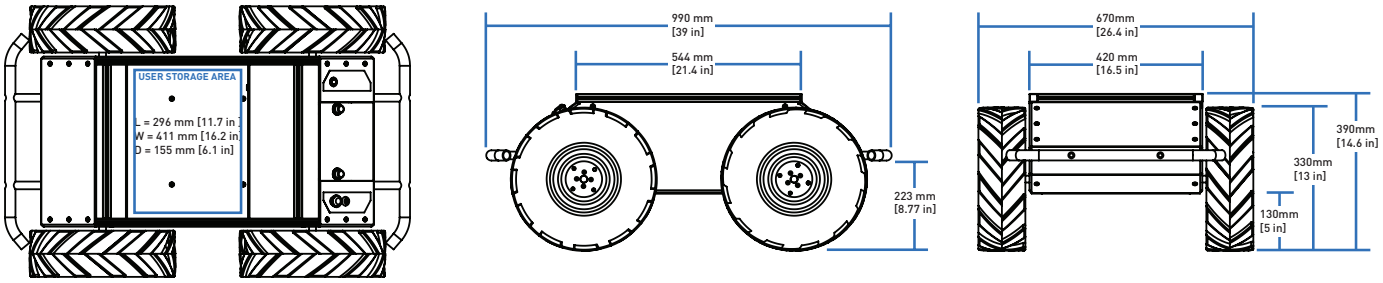
Übersicht

- Technische Spezifikationen HUSKY [3]
- Abmessungen Velodyne 3D-LIDAR (Gehäuse) [14]
- Abmessungen Velodyne 3D-LIDAR (Strahlenzentrum) [14]

HUSKY™

UNMANNED GROUND VEHICLE

CLEARPATH ROBOTICS™



TOP SIDE FRONT

TECHNICAL SPECIFICATIONS

SIZE AND WEIGHT	
EXTERNAL DIMENSIONS (L x W x H)	990 x 670 x 390 mm (39 x 26.4 x 14.6 in)
INTERNAL DIMENSIONS	296 x 411 x 155 mm (11.7 x 16.2 x 6.1 in)
WEIGHT	50 kg (110 lb)
WHEELS	330 mm (13 in) Lug Tread
GROUND CLEARANCE	130 mm (5 in)
SPEED AND PERFORMANCE	
MAX. PAYLOAD	75 Kg (165 lb)
ALL-TERRAIN PAYLOAD	20 Kg (44 lb)
MAX SPEED	1.0 m/s (2.3 mph)
DRIVETRAIN / DRIVE POWER	4x4 Zero-Maintenance
MAX CLIMB GRADE	45° (100% Slope)
MAX TRAVERSAL GRADE	30° (58% Slope)
BATTERY AND POWER SYSTEM	
BATTERY CHEMISTRY	Sealed Lead Acid
CAPACITY	24 V, 20 Ah
RUNTIME - STANDBY	8 Hours
RUNTIME - NOMINAL USAGE	3 Hours
CHARGE TIME	4 Hours
USER POWER	VBAT fused at 5A, 12V fused at 7.5A, 24V fused at 3A. 192W total power available. Upgrade to 480W optional.
INTERFACING AND COMMUNICATION	
CONTROL MODES	Direct voltage, wheel speed, and kinematic velocity.
FEEDBACK	Battery voltage, motor currents, wheel odometry, and control system output.
COMMUNICATION	RS232 @ 115200 baud
ENCODERS	Quadrature: 78,000 pulses/m
DRIVERS AND APIs	ROS, C++, and Python.
ENVIRONMENTAL	
OPERATING AMBIENT TEMPERATURE	-10 to 40 °C (14 to 104 °F) Not in direct sunlight
STORAGE TEMPERATURE	-40 to 60 °C (-40 to 140 °F)
RATING	IP 44 (upgrade to IP 55 available)

Contact us today for pricing and a free 30 minute technical assessment: 1-800-301-3863

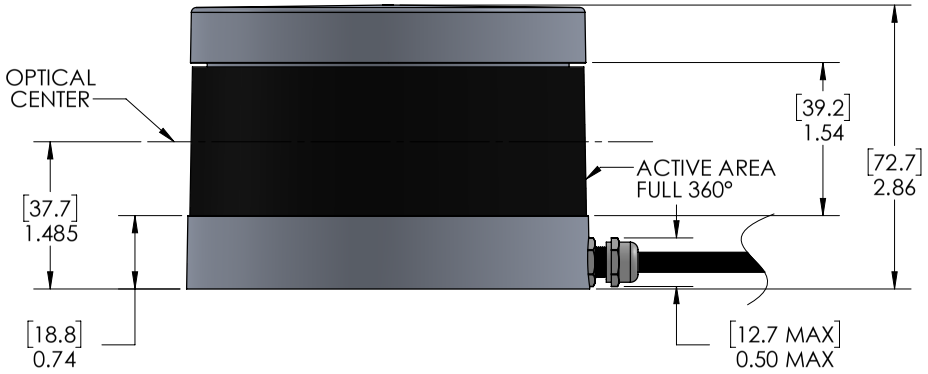
© 2016 Clearpath Robotics, Inc. All Rights Reserved. Clearpath Robotics, Husky, Husky A200, and clearpathrobotics.com are trademarks of Clearpath Robotics.
All other product and company names listed are trademarks or trade names of their respective companies.

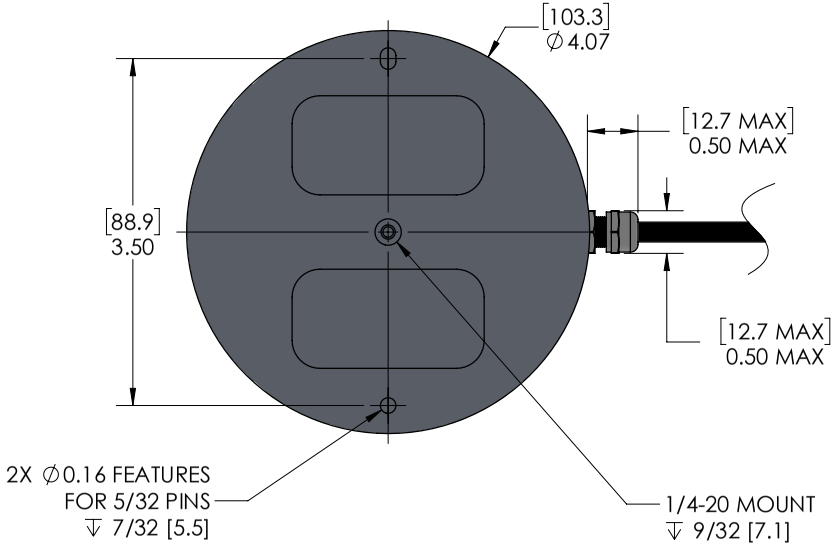
THE INFORMATION CONTAINED IN THIS DRAWING IS THE SOLE PROPERTY OF VELODYNE LIDAR, INC., ANY REPRODUCTION IN PART OR WHOLE WITHOUT THE WRITTEN PERMISSION OF VELODYNE LIDAR, INC., IS PROHIBITED.

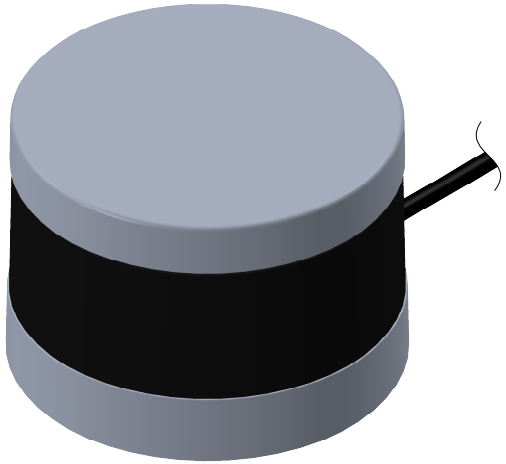
NOTES:
1. ALL DIMENSIONS ARE IN DECIMAL INCHES AND [MILLIMETERS].
2. ALL DIMENSIONS FOR REFERENCE ONLY

REVISIONS HISTORY

ECN	REV	DESCRIPTION	APPROVAL	DATE
1044	A	RELEASE TO PRODUCTION	BRP	12/05/14
1118	B	MOD. ENVELOPE DIMS (SHEET 1 ZONES A5, B5 AND C5). ADDED WINDOW CONFIGURATIONS (SHEET 2). ADDED OPTICAL KEEP OUT ZONE (SHEET 4).	SSN	08/26/15
0686	B1	MOD. COMPANY NAME IN TITLE AND COPYRIGHT BLOCK. MOD. DESCRIPTION ON SHEET 3 TO INCL. VERT. BEAM DIV.	SSN	12/13/16



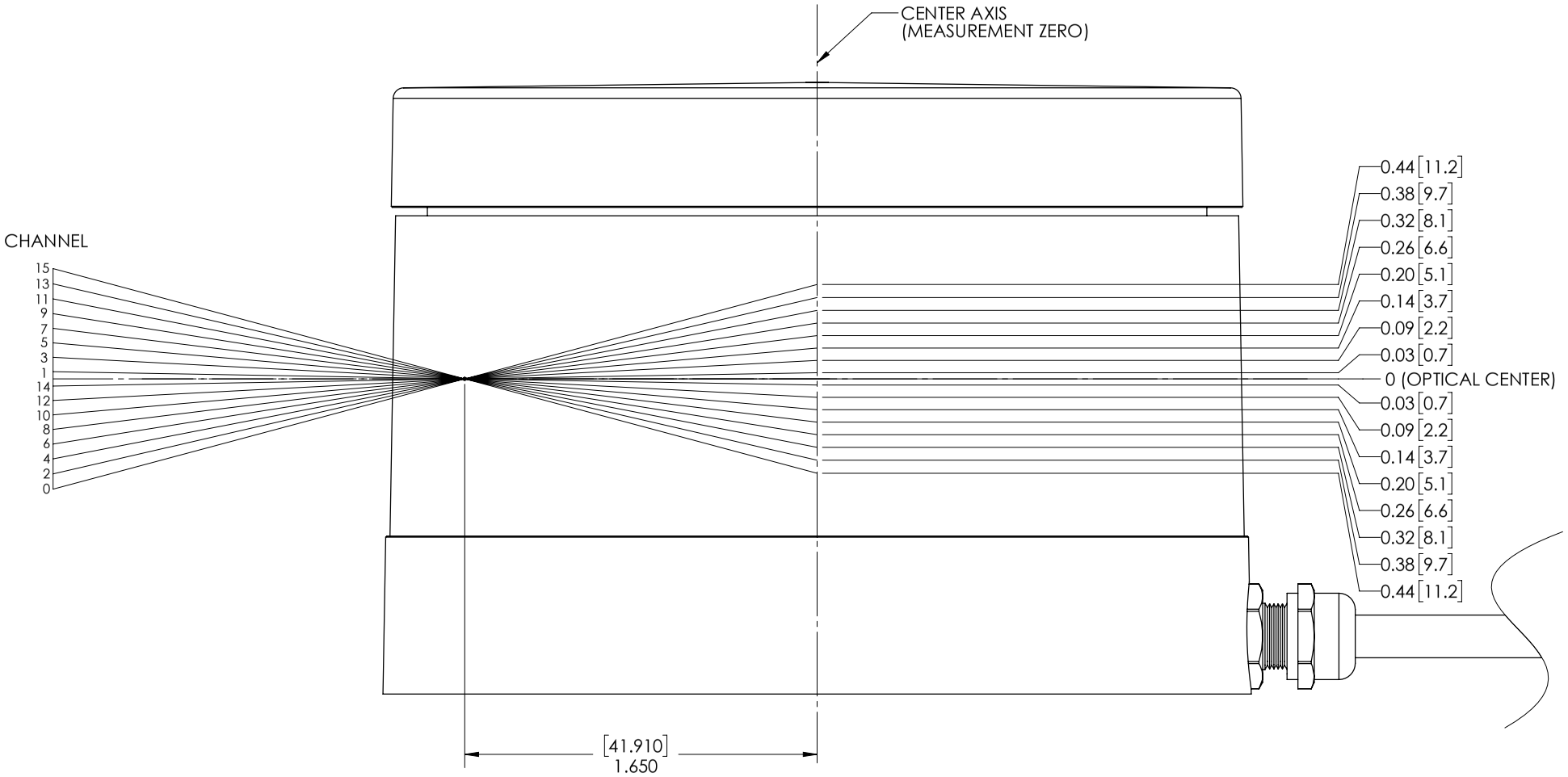




UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN DECIMAL INCHES AND [METRIC]. TOLERANCES ARE: DECIMALS .xx ± .01 [0.25] .xxx ± .005 [0.127] ANGLES ± N/A	SOLIDWORKS GENERATED DRAWING, DO NOT MANUALLY UPDATE	Velodyne LiDAR™
MATERIAL	APPROVALS / DATE	TITLE: ENVELOPE VLP-16
FINISH	DWN: BRP 12/05/14	
DO NOT SCALE DRAWING	CHKD / QC / ENG	SIZE: B DWG. NO.: 86-0101 REV.: B1
		SCALE: 3:4 SHEET 1 of 4

THE INFORMATION CONTAINED IN THIS DRAWING IS THE SOLE PROPERTY OF VELODYNE LIDAR, INC., ANY REPRODUCTION IN PART OR WHOLE WITHOUT THE WRITTEN PERMISSION OF VELODYNE LIDAR, INC., IS PROHIBITED.

REVISIONS HISTORY				
ECN	REV	DESCRIPTION	APPROVAL	DATE
		SEE SHEET 1		



APPROXIMATE BEAM CENTERS
DOES NOT REPRESENT BEAM WIDTHS
BEAMS ARE APPROXIMATELY 0.75 TALL AT FOCAL POINT
BEAM DIVERGENCE IS APPROXIMATELY 3 MILLIRADIANS HORIZONTALLY AND 1.5 MILLIRADIANS VERTICALLY
APPLY VERTICAL CORRECTION FOR GREATEST ACCURACY

SIZE	DWG. NO.	REV.
B	86-0101	B1
SCALE: NTS		SHEET 3 of 4

B. Quick Start Guide

B.1. Husky 101

Der Husky ist der Transportroboter, welcher die beiden Produktionslinien im PTZ und RITZ miteinander verbinden soll. Stationiert ist er am Ende der Produktionslinie im PTZ, an welcher er mit den Werkstücken beladen wird. Der Husky selbst kann manuell mit einem Kontroller gesteuert werden, soll später aber mithilfe des SLAM-Algorithmus und diversen Sensoren autonom fahren. Integriert sind bereits zwei LIDAR Sensoren, ein 2D-LIDAR auf der Vorderseite des Roboters, sowie ein 3D-LIDAR erhöht auf der Rückseite. Die vier orangenen Vierecke sind die Werkstückhalterungen, in welche die Werkstücke gelegt werden. Im Zwischenraum auf der schwarzen Abdeckung kann das HDMI-Kabel angeschlossen werden. Der Ethernet Anschluss funktioniert nicht, da bereits beide Ethernet-Anschlüsse des Computers für die LIDAR-Sensoren verwendet werden. Falls ein Ethernet-Port gebraucht wird, braucht man einen Ethernet-Splitter. Im USB-Hub, welcher auf der Grundplatte an der Seite montiert ist, können diverse Geräte wie Maus, Tastatur und Bluetooth-Dongle für den Kontroller eingesteckt werden. Alle genannten Beschriftungen finden sich in Abbildung B.1. Um mit dem Husky arbeiten zu können muss zunächst ein Bildschirm per HDMI-Kabel an den dafür vorgesehen Anschluss auf der Grundplatte des Huskys angeschlossen werden. Durch das Drücken des Power-Button startet automatisch auch der Husky-Computer. Maus und Tastatur können an den montierten USB-Hub angeschlossen werden. Mit dem Passwort 'clearpath' öffnet sich die Ubuntu-Oberfläche des Husky-Computers.

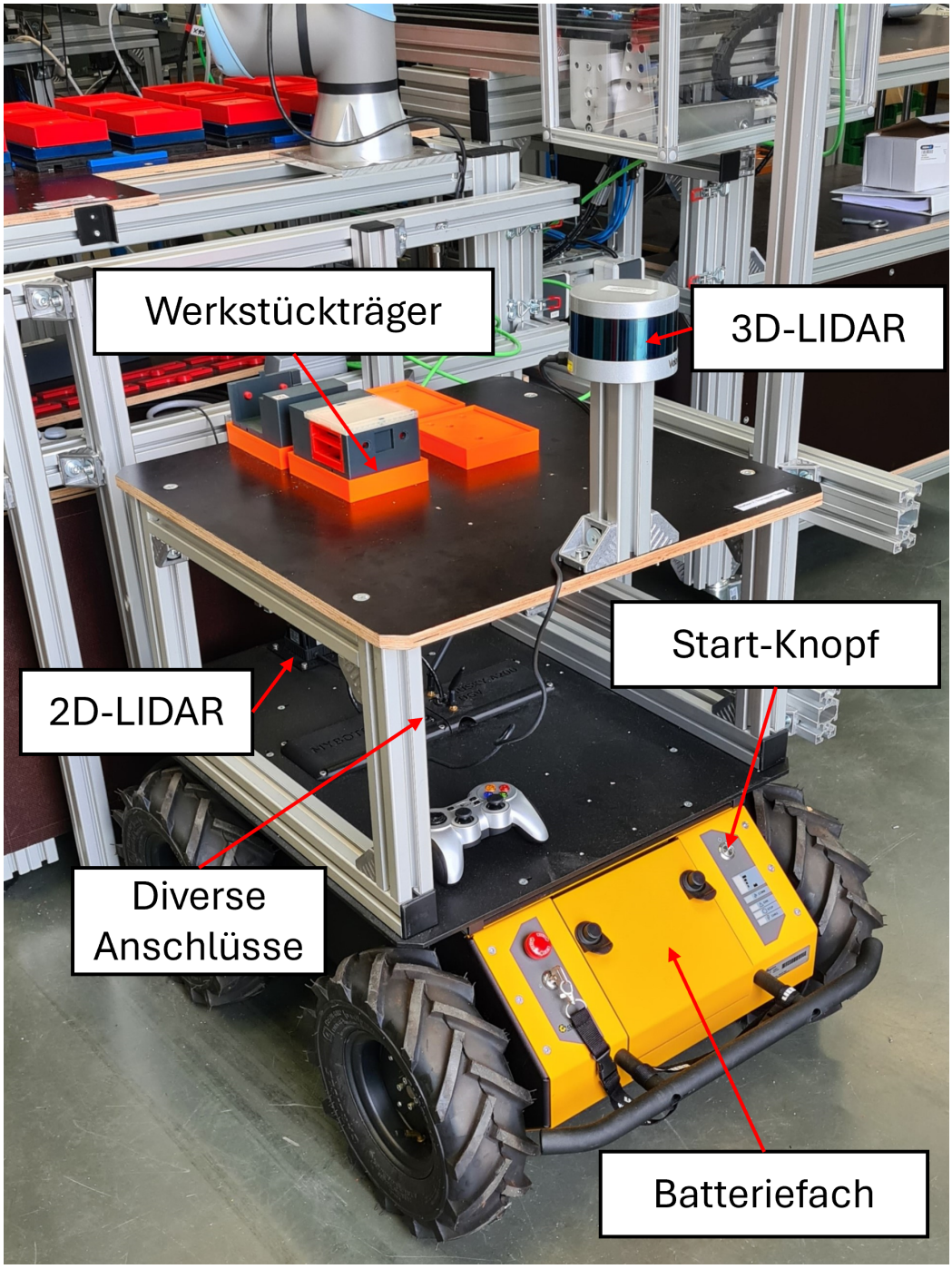


Abbildung B.1.: Grundlegende Beschriftung des Huskys

B.2. Laden des Husky

Zum Laden des Huskys muss zunächst die Abdeckung auf der Vorderseite entnommen werden. Die in Abbildung B.2 roten und schwarzen Kabel müssen aus der Batterie gezogen werden. Anschließend muss das Netzteil wie in Abbildung B.3 an die Kabel angeschlossen werden. Die Anzeige des Ladezustands der Batterie ist fehlerhaft, sie flackert immer im Bereich zwischen wenigen Strichen. Daher ist dieser Anzeige nicht zu trauen. Einen niedrigen State of Charge erkennt man daran, dass der Husky nicht mehr so stark beschleunigt bzw. rotiert oder an dem flackernden Bild des 2D-LIDARs in der GUI. Es empfiehlt sich den Husky nach eigener Schätzung ab und zu für mehrere Stunden zu laden.

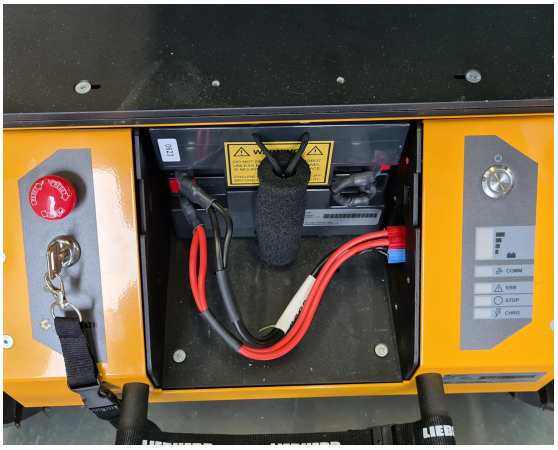


Abbildung B.2.: An Batterie angeschlossen



Abbildung B.3.: An Netzteil angeschlossen

B.3. Steuern des Roboters mithilfe des Controllers

Um den Husky mit dem Controller fernzusteuern, muss dieser nach dem Anschalten manuell neugestartet werden. Dies liegt vermutlich daran, dass der Prozess, welcher für die Controller-Verbindung und Fernsteuerung zuständig ist, beim normalen Startvorgang nicht ordnungsgemäß ausgeführt wird. Mit folgenden Schritten kann man eine Remote-Verbindung mit dem Husky aufbauen und das System neustarten:

1. Um eine Remote-Verbindung herzustellen ist ein gemeinsames Netzwerk notwendig, mit welchem sich sowohl Husky, als auch der Remote-Laptop verbinden. Da das DHBW-Netzwerk mit vielen Protokollen geschützt ist, verwendet man einfachheitshalber den Mobilen Hotspot eines Handys. Mit diesem müssen sich dann sowohl Husky als auch Remote Laptop verbinden.

2. Der nächste Schritt ist das Aufrufen der Kommandozeile auf dem Husky-Computer und die Eingabe des Befehls:

   ```
   ifconfig
   ```

   Danach öffnet sich ein Fenster, welches verschiedenste Verbindungen anzeigt. Die IP-Adresse der WLAN-Verbindung findet sich unter der Bezeichnung 'wlp2s0'. Zu beachten ist, dass sich diese Adresse geringfügig ändern kann, deshalb empfiehlt es sich vor jeder Remote-Verbindung die IP-Adresse zu überprüfen.

3. Als nächstes muss die Kommandozeile auf dem Remote-Computer aufgerufen werden und entsprechend der aus Schritt 2 ermittelten IP-Adresse folgender Befehl eingeben werden (hier ist eine beispielhafte IP-Adresse genommen worden):

   ```
   ssh administrator@192.168.242.12
   ```

4. Nun sollte eine Eingabeaufforderung für ein Passwort kommen. Das Passwort lautet standardmäßig 'clearpath'. Zu beachten ist, dass das Passwort beim Eingeben nicht durch ein '*' gekennzeichnet wird, sondern 'blind' geschrieben wird und durch 'ENTER' bestätigt werden muss.

5. Es sollte eine 'MYBOTSHOP' Schriftzug erscheinen, unter welchem eine Übersicht über das z.B. letzte Login oder die nun möglichen einzugebende Befehle gibt. Um den Roboter mit dem Controller fernzusteuern, muss dieser mit dem folgenden Befehl neu gestartet werden:

   ```
   sudo service mbs_husky restart
   ```

Durch die erneute Eingabe des Passworts sollte das Statuslämpchen auf Grün wechseln und der Husky sollte bereit sein, sich mit dem Controller steuern zu lassen. Falls dies nicht der Fall ist, dann Schritt 5 wiederholen oder den Husky ausschalten und die Anleitung von vorne beginnen.

6. Der Husky lässt sich nun entsprechend Abbildung B.4 steuern.



Abbildung B.4.: Kontrollerbelegung zum Steuern des Husky

B.4. Visualisierung der Sensordaten und Steuern des Roboters mittels GUI

Es kann vorkommen, dass die Daten (vor allem vom 2D-LIDAR) fehlerhaft angezeigt werden. Um dies zu beheben könnten die Schritte aus Kapitel B.3 hilfreich sein.

1. Durch das Öffnen des Terminals auf dem Husky-Computer und der Eingabe des folgenden Befehls öffnet sich die GUI 'RViz', welche die Sensordaten darstellt.

```
roslaunch mbs_husky_viz viz.launch
```

2. Durch das Anklicken der Schaltfläche 'Interact' erscheinen rote Pfeile bzw. ein blauer Kreis, mit dem sich der Husky über die GUI steuern lassen kann (Die ersten beiden Schritte beziehen sich lediglich auf den vorinstallierten 2D-LIDAR).

3. Um die Daten des 3D-LIDARs zu visualisieren, müssen zunächst zwei parallele Terminals geöffnet werden

4. Im ersten der beiden Terminals muss der Befehl

```
roslaunch velodyne_pointcloud VLP16_points.launch
```

   eingegeben werden. Dieses Fenster muss im Hintergrund dauernd geöffnet bleiben.

5. Im zweiten Terminal muss nun der Befehl zum Starten der GUI 'RViz' eingegeben werden

```
rosrun rviz rviz -f velodyne
```

6. Es öffnet sich die RViz Benutzeroberfläche ohne Sensordaten. Um die Sensordaten zu integrieren, muss zunächst unter der Schaltfläche 'Add' im linken unteren Bereich eine 'PointCloud2' hinzugefügt werden. Es erscheint ein Feld 'Topic' im linken Bereich. In dieser Schaltfläche muss nun noch '/velodyne_points' ausgewählt und mit 'ENTER' bestätigt werden. Nun sollten die 3D-LIDAR Daten erscheinen.

7. Der Roboter lässt sich über die 3D-Lidar GUI nicht steuern, da die Position des Roboters relativ zum Sensorbild in der Software nicht definiert ist.

B.5. Support bei Softwarefragen

Bei Fragen bzw. Unklarheiten bezüglich der Software und der Dateistruktur des Husky-Systems empfiehlt es sich direkt beim Hersteller nachzufragen. 'MYBOTSHOP' bietet hierzu ein Forum an, in dem es schon einige hilfreiche Beiträge gibt. Ansonsten gibt es die Möglichkeit ein eigenes Ticket zu eröffnen, welches im Normalfall binnen wenigen Tagen bearbeitet wird. Der Link zum Forum findet sich in Kapitel B.6. Bezüglich der Integration des 3D-LIDARs in das Robotergesamtsystem (gemeint ist die Kenntnis der Software über die Sensordaten relativ zur Roboterposition) wurde bereits ein Ticket eröffnet. Das eröffnete Ticket findet sich *hier*.

B.6. Nützliche Links

- Projektshare 23/24 (Dateien, Bilder, Literatur)
- Website Husky - clearpath
- Guides Husky - clearpath
- Website Husky - MyBotShop
- Forum - MyBotShop
- Website mobiler Monitor
- Website Halterung

B.7. Next ToDos

Der nächste Schritt stellt die Integration der 3D-LIDAR Daten in die Robotersoftware dar. Damit wäre eine Basis für die Implementierung des SLAM-Algorithmus geschaffen. Dafür ist auch ein aktualisiertes CAD-Modell notwendig. Es gibt bereits eine ältere Version, jedoch sind dort diverse Änderungen v.a. die 3D-LIDAR Sensor Halterung und Position noch nicht implementiert. Das Modell kann entweder selbst mit dem Programm "Siemens NX 1980" und den Dateien im Share bearbeitet werden oder man setzt sich in Verbindung mit der Gruppe Digitaler Zwilling. Durch die Einrichtung eines mobilen Arbeitsplatzes (siehe Kapitel 4.4) wird es möglich sein, unabhängig von einer externen Stromversorgung den Husky zu steuern und die Sensordaten im Outdoor-Bereich einzusehen. Zudem wäre es möglich einen zweiten 2D-LIDAR zu integrieren, um das Sichtfeld des Huskys noch weiter zu optimieren. Es soll lediglich darauf hingewiesen werden, dass noch ein weiterer Sensor vorhanden ist, mögliche Position und Anbringung wurde in dieser Studienarbeit jedoch nicht behandelt.