

# 3D Laserscanner für mobilen Roboter

**Industriearbeit PAIND+E1**

im Auftrag des Industriepartners

**RUAG AG**

an der

Hochschule Luzern Technik & Architektur

im Studiengang Elektrotechnik

## **Schwerpunkt**

Signalverarbeitung & Kommunikation,  
Automation & Embedded Systems

**Dozent:** Björn Jensen

**Experte:** Markus Thalmann

**Eingereicht von:** Daniel Zimmermann

**Matrikelnummer:** 15-465-271

**Datum der Abgabe:** 22.12.2017

**Klassifikation:** Rücksprache

# Eigenständigkeitserklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig angefertigt und keine anderen als die angegebenen Hilfsmittel verwendet habe. Sämtliche verwendeten Textausschnitte, Zitate oder Inhalte anderer Verfasser wurden ausdrücklich als solche gekennzeichnet.

Wolfenschiessen, den 22.12.2017

Daniel Zimmermann

# Abstract

This documentation is the result of the project Modul PAIND+E1 at the Lucerne School of Engineering and Architecture for the industry partner RUAG AG.

The following chapters contain the experiences and results during the project from September to December 2017. It describes the realisation of a 3D laser module prototype, which can map the environment. The main part is subdivided in different phases and reflects the timeline of the Project.

The first part is a summary of the results during the research phase. It contains the knowledge about the available 3D sensors, the potential hardware components and the necessary software to implement the solution after the functional specifications.

A total of three concepts were elaborated, which have different approaches. The first concept called "platform", which can turn the 3D sensor in a wide range of angle, while using servo motors. This concept was no longer pursued, since the two other concepts were more suitable.

The two other concepts are based on an endlessly rotating "tower". The idea behind it, are state-of-the-art projects. The difference between the two concepts is the position of the signal processing unit. In the unrotated version, the unit is below in a static case. Only the 3D-sensor is rotating for mapping. In the other version, the unit in the case is also rotating. Only the interface is static.

The main content is about the realised concept, which is similar to the unrotated version before. The realisation phase describes the process, how the case and the electronic parts are assembled. In a separate topic, it describes, how the Software is implemented and how it works together with the hardware.

After the realisation the prototype is tested. Because of a few faults and problems, the realised prototype can't satisfy the full requirements.

At the end, a short reflection summarises the biggest challenges during the project and how to solve them. It also reflects the project management and give a little outlook.

# Abstract

Diese Dokumentation ist das Ergebnis des Projektmoduls PAIND+E1 an der Hochschule Luzern Technik und Architektur für den Industriepartner RUAG AG.

Die nachfolgenden Kapitel beinhalten Erfahrungen und Resultate des Projekts, die zwischen September bis Dezember 2017 erarbeitet wurden. Die Dokumentation beschreibt die Realisierung eines Prototyps, mit welchem die Umgebung vermessen und modelliert werden kann. Der Hauptteil wurde in verschiedene Projektphasen unterteilt und gibt den Verlauf des Projekts wieder.

Der erste Teil ist eine Zusammenfassung der Ergebnisse aus der Informationsbeschaffung. Es beinhaltet die wesentlichsten Spezifikationen des vorhandenen 3D Sensor, die erforderlichen Hardwarekomponenten, sowie die nötige Software, um eine Lösung nach den Vorgaben zu entwickeln.

Insgesamt wurden drei Konzepte erstellt. Das erste Konzept "Plattform" kann den 3D Sensor mittels Servomotoren in alle Richtungen neigen. Da sich nachfolgende Konzepte besser eignen, wurde dieses Konzept verworfen. Die zwei anderen Konzepte basieren auf einem endlos drehenden "Turm". Die Idee dazu liefern bestehende State-Of-The-Art Projekte. In der unrotierenden Version dreht sich der 3D Sensor mit dem Turm. Die Signalverarbeitung wird in einem feststehenden Gehäuse unterhalb des Sensors platziert. In der rotierenden Version dreht die gesamte Elektronik mit und nur die Schnittstellen werden unrotierend herausgeführt.

Es handelt sich hierbei um eine angepasste unrotierende Version. Dabei wird der Bau des Gehäuses und der Einsatz der elektrischen Komponenten, sowie die Implementation der Software erläutert.

Danach wurden die Funktionen des Prototyps getestet. Der erarbeitete Prototyp konnte wegen einiger Fehler und Problemen die gewünschten Vorgaben nicht erfüllen. Zuletzt gibt eine Reflektion Auskunft über bedeutende Schwierigkeiten und wie sie gelöst wurden.

# Inhaltsverzeichnis

<b>Glossar</b>	<b>VII</b>
<b>Abbildungsverzeichnis</b>	<b>X</b>
<b>Tabellenverzeichnis</b>	<b>XI</b>
<b>Literaturverzeichnis</b>	<b>XII</b>
<b>1 Einleitung</b>	<b>1</b>
<b>2 Informationsbeschaffung</b>	<b>3</b>
2.1 Entfernungsmessung . . . . .	3
2.1.1 Velodyne VLP-16 Puck . . . . .	3
2.2 Stand der Technik . . . . .	6
2.2.1 IMM MSAS Team MSS Warschau . . . . .	7
2.2.2 Hector Tracker Team Hector Darmstadt . . . . .	8
2.2.3 Schlussfolgerung . . . . .	10
2.3 Software . . . . .	10
2.3.1 ROS Robot Operating System . . . . .	11
2.3.2 Velodyne Package . . . . .	11
2.3.3 Schlussfolgerung . . . . .	13
2.4 Datenverarbeitung . . . . .	13
2.4.1 Raspberry Pi 2 & 3 . . . . .	14
2.4.2 Banana Pi M3 . . . . .	14
2.4.3 Odroid C2 & XU4 . . . . .	15
2.4.4 Up Board Squared . . . . .	15
2.4.5 Schlussfolgerung . . . . .	15
2.5 Antriebsmöglichkeiten . . . . .	16
2.5.1 Schrittmotor . . . . .	16
2.5.2 Gleichstrommotor . . . . .	17

2.6	Positionsbestimmung . . . . .	17
2.6.1	LED und Photodiode . . . . .	18
2.6.2	Sensor QRE 1113 . . . . .	18
2.7	Speisung und Verkabelung . . . . .	19
2.7.1	Abwärtswandler . . . . .	19
2.7.2	Ethernet Schleifring . . . . .	19
2.7.3	Ethernet Switch . . . . .	20
2.8	Zwischenfazit . . . . .	20
<b>3</b>	<b>Konzeption</b>	<b>21</b>
3.1	Konzeptionsgrundlage . . . . .	21
3.2	Variante 1: Plattform . . . . .	21
3.3	Variante 2: Turm stationär . . . . .	23
3.4	Variante 3: Turm rotierend . . . . .	25
3.5	Konzeptfazit . . . . .	27
3.6	Evaluation der Komponenten . . . . .	27
3.6.1	Datenverarbeitung: Raspberry Pi 3 . . . . .	28
3.6.2	Antrieb: Pololu 25Dx56L . . . . .	28
3.6.3	Positionsbestimmung: Fairchild QR1113 . . . . .	28
3.6.4	Drehübertager: Senring SNE01-24GV . . . . .	29
3.7	Zwischenfazit . . . . .	29
<b>4</b>	<b>Realisierung</b>	<b>30</b>
4.1	Hardware . . . . .	31
4.1.1	mechanische Komponenten & Gehäuse . . . . .	31
4.1.2	elektrische Komponenten . . . . .	32
4.2	Software . . . . .	34
4.2.1	Ansteuerung mittels GPIO . . . . .	34
4.2.2	Softwareaufbau . . . . .	35
4.2.3	Phase 1: Empfangen und Weitersenden . . . . .	36
4.2.4	Phase 2: Positionstransformation . . . . .	37
4.2.5	Phase 3: Punktwolke zusammenfügen . . . . .	39
4.2.6	Anwendung der Software . . . . .	41
4.3	Zwischenfazit . . . . .	42

<b>5 Tests</b>	<b>43</b>
5.1 Testspezifikationen . . . . .	43
5.2 Testergebnisse . . . . .	44
5.2.1 Speisung . . . . .	44
5.2.2 Mechanik . . . . .	44
5.2.3 Datenkommunikation . . . . .	44
5.2.4 Datenverarbeitung . . . . .	45
5.2.5 Positionsbestimmung . . . . .	45
5.2.6 Nullpunktterkennung . . . . .	45
5.2.7 Software . . . . .	45
5.3 Zwischenfazit . . . . .	45
<b>6 Reflektion</b>	<b>46</b>
6.1 Fazit . . . . .	46
6.2 Erläuterungen Projektmanagement . . . . .	46
6.3 Ausblick . . . . .	47
6.4 Schlusswort . . . . .	47
6.5 Danksagung . . . . .	48
<b>A Pflichtenheft</b>	<b>I</b>
<b>B Projektmanagement</b>	<b>XI</b>
<b>C Risikoanalyse</b>	<b>XIII</b>
<b>D Aufgabenstellung</b>	<b>XV</b>

# Glossar

ARM	Acorn Risc Machine verbreitetes Mikroprozessor-Design, das grösstenteils auf 32Bit-Prozessoren beruht(ARMv7)
AMD64	X64 Prozessor-Architektur der handelsüblichen 64-Bit-PCs und Server
BSD	Berkley Software Distribution frei verwendbare Lizenz, auch als Vorlage für kommerzielle Produkte
CPR	Counts Per Revolution
eMMC	embedded Multimedia Card Digitales Speichermedium, basiert auf dem Prinzip der Flash Speicherung arbeitet
EnRich	European Robotic Hackathon weltweit erste und einzige Robotik Wettbewerb mit realen Szenarios
GPS	Global Positioning System globales Navigationssatellitensystem zur Positionsbestimmung
GPU	Grafikprozessor spezialisierter Prozessor für Grafikanwendungen
IMU	Inertiale Messeinheit Kombination mehrerer Trägheitssensoren
LIDAR	Light Detection And Ranging Verfahren zur optischen Distanzmessung
Mikrostepping	Schrittteilung Unterteilung der zu tätigenden Schritten bei Schrittmotoren um feinere Abstufungen zu erhalten

PCL	Point Cloud Library umfangreiche Softwarebibliothek für 3D Visualisierungen
KO	Kathodenoszilloskop Messgerät zur Analyse von Signalen
PWM	Pulsweitenmodulation Modulationsverfahren mit varierenden Rechteckimpulsen
Quaternionen	Erweiterung der komplexen Zahlen Beschreibung von Drehbewegungen, bietet den Vorteil von eindeutigen 3D Positionen
RAM	Random Access Memory direkt ansprechbare Speicherbausteine
Refactoring	Strukturverbesserung strukturelle Anpassung, verbessert Lesbarkeit, Wartbarkeit und Erweiterbarkeit
ROS	Robot Operating System Software Framework für Robotikanwendungen
Scope	Sichtbarkeitsbereich Sichtbarkeitsbereich einer Variable in einem Softwareprojekt
SD	Secure Disk Memory Card digitales Speichermedium, das nach dem Prinzip der Flash-Speicherung arbeitet
SOC	Sysem-on-a-Chip Integration der Funktionen eines programmierbaren elektronischen Systems auf einem Chip
State-Machine	sequenzielle Ablauffolge Verhaltensmodell einer Software, dass Aktionen, Zuständen und Zusandsänderungen beschreibt

UDP	User Data Protocol verbindungsloses Netzwerkprotokoll zur Versendung von Datagrammen von IP-basierten Rechnernetzen
XPC	Barebone PC kompakte unvollständige Computerdesign, meist mit vielen Schnittstellen und guter Erweiterbarkeit

# Abbildungsverzeichnis

2.1	Laserstrahlen des Velodyne VLP-16 . . . . .	4
2.2	Ansicht auf die Interfacebox . . . . .	5
2.3	Aufbau Datenpaket . . . . .	6
2.4	Roboter des Team IMM EnRich . . . . .	7
2.5	Roboter des Team Hector EnRich . . . . .	9
2.6	Velodyne Treiber aus rqt-graph . . . . .	12
2.7	PointCloud2 Datenstream mit Rviz . . . . .	12
2.8	Quadraturencoder mit 4 Sensoren . . . . .	18
2.9	Wirkungsgrad D24V50F5 . . . . .	19
3.1	Skizze Variante 1 . . . . .	22
3.2	Ball on a Plate CAD . . . . .	23
3.3	Skizze Variante 2 . . . . .	24
3.4	Skizze Variante 3 . . . . .	26
4.1	realisiertes Konzept . . . . .	30
4.2	gelayoutete Komponenten in OnShape . . . . .	32
4.3	Nullpunktterkennung mittels QRE 1113 . . . . .	33
4.4	Ablaufdiagramm der GPIO-Ansteuerung . . . . .	34
4.5	Software Ablaufstruktur . . . . .	35
4.6	Softwareerweiterung mit Laser_3D . . . . .	36
4.7	Roll-Pitch-Yaw mit X-Y-Z-Achsen . . . . .	37
4.8	dynamische Koordinatentransformation . . . . .	39
4.9	UML Klassendiagramm SuListener . . . . .	39
4.10	Punktwolke einer Person bei Distanz 1m . . . . .	41
4.11	zusammengefügte Punktwolken . . . . .	42

# Tabellenverzeichnis

3.1 Morphologischer Kasten . . . . .	27
5.1 Funktionstest im Überblick . . . . .	43

# Literaturverzeichnis

- [Bed17] Janusz Bedkowski. *Enrich Team Informationen*. Juli 2017. URL: <http://enrich.european-robotics.eu/> (besucht am 13.11.2017).
- [Ben15] Benedict. *Student's 3D printed balancing device is a whole new ball game*. Dez. 2015. URL: [www.3ders.org/articles/20151213-students-3d-printed-balancing-device-is-a-whole-new-ball-game.html](http://www.3ders.org/articles/20151213-students-3d-printed-balancing-device-is-a-whole-new-ball-game.html) (besucht am 06.10.2017).
- [Bit17] Prof. Dr. O. Bittel. *Mobile Roboter - Position und Orientierung*. 2017. URL: [http://www-home.htwg-konstanz.de/~bittel/ain\\_robo/Vorlesung/02\\_PositionUndOrientierung.pdf](http://www-home.htwg-konstanz.de/~bittel/ain_robo/Vorlesung/02_PositionUndOrientierung.pdf) (besucht am 28.11.2017).
- [Cor17] Pololu Corporation. *25D mm Metal Gearmotors*. Jan. 2017. URL: <https://www.pololu.com/product/3264> (besucht am 29.09.2017).
- [Koh17] Stefan Kohlbrecher. *Enrich Team Informationen*. Juli 2017. URL: <http://enrich.european-robotics.eu/> (besucht am 13.11.2017).
- [LiD16] Velodyne LiDAR. *VLP-16 Manual*. März 2016. URL: <http://velodynelidar.com/vlp-16.html> (besucht am 28.09.2017).
- [Mou17] Mouser. *Suchergebnisse Mouser stepper motor encoder*. Jan. 2017. URL: <https://www.mouser.ch/Search/Refine.aspx?Keyword=stepper+motor+encoder> (besucht am 29.09.2017).
- [Pol17a] "Pololu". *DRV8825 Stepper Motor Driver Carrier, High Current*. Jan. 2017. URL: <https://www.pololu.com/product/2133> (besucht am 29.10.2017).
- [Pol17b] Pololu. *Pololu 5V, 5A Step-Down Voltage Regulator D24V50F5*. Jan. 2017. URL: <https://www.pololu.com/product/2851> (besucht am 11.10.2017).
- [ros16a] ros.org. *Ros Tutorials*. Mai 2016. URL: [http://wiki.ros.org/velodyne\\_pointcloud?distro=kinetic](http://wiki.ros.org/velodyne_pointcloud?distro=kinetic) (besucht am 19.10.2017).
- [ros16b] ros.org. *Writing a Simple Publisher and Subscriber*. Sep. 2016. URL: <http://wiki.ros.org/ROS/Tutorials/WritingPublisherSubscriber%28c%2B%2B%29> (besucht am 12.10.2017).

- [ros17] ros.org. *Ros Tutorials*. Nov. 2017. URL: <http://wiki.ros.org/ROS/Tutorials> (besucht am 12.10.2017).
- [Sen15] Senring. *SNE01 series Mini 1000M Ethernet slip ring*. Dez. 2015. URL: <http://www.senring.com/ethernet-slip-rings/SNE01.html> (besucht am 03.10.2017).
- [TEA16] BPI TEAM. *Banana Pi M3 Ubuntu Image*. Apr. 2016. URL: <http://forum.banana-pi.org/t/bpi-m3-new-image-bpi-m3-ubuntu-16-04-beta-mate/1440> (besucht am 29.09.2017).
- [Tea16] sinvoip bpi Team. *Banana pi BPI-M64 vs Raspberry pi 3 vs odroid vs Pine64*. Aug. 2016. URL: <http://forum.banana-pi.org/t/banana-pi-bpi-m64-vs-raspberry-pi-3-vs-odroid-vs-pine64/2070> (besucht am 01.10.2017).
- [tfo16] tfoote. *libpcl-dev missing on Ubuntu armhf, Accelerometers, and Machine Learning*. Mai 2016. URL: <https://github.com/ros/rosdistro/issues/11583> (besucht am 30.09.2017).

# Kapitel 1

## Einleitung

Im Forschungszweig der Robotik entstehen aktuell ständig neue und revolutionäre Technologien. Dabei steht die Transformation des weitgehend selbstständigen Roboters im Vordergrund. Diverse Pilotprojekte beweisen bereits heute, dass durch eine komplexe Abstimmung hoch präziser Sensoren die kognitiven und sensorischen Fähigkeiten des Menschen nachgeahmt, wenn nicht sogar übertroffen werden können. Ein gutes Beispiel für diese Transformation sind mobile Roboter wie der iRobot Packbot. Durch entsprechende Logik und Sensorik können die geländegängigen Roboter dem Menschen einen enormen Dienst erweisen. Für Menschen unzugängliche oder nur unter hohem Gefahrenpotential begehbarer Orte wie Kriegsgebiete, von Naturkatastrophen geschädigte oder radioaktiv verstrahlte Umgebungen, können sie Aufgaben bewältigen, welche dem Menschen alleine unmöglich erscheinen.

Durch die zunehmende Rechenleistung von Computern und den daraus resultierenden Datenmengen entsteht nun auch die Möglichkeit, mittels Robotern detaillierte Modelle der erwähnten Einsatzgebieten zu erstellen. An diesem Punkt setzt nun die Aufgabenstellung des PAIND+E1 an. Es soll ein low-cost Prototyp eines 3D-Laser-Modul entwickelt werden, mit dem eine dreidimensionales Modell der Umgebung möglichst detailliert visualisiert werden kann. Dabei soll einerseits die Frage geklärt werden, welche Konfiguration eine bestmögliche Modellierung der Umgebung bietet, und anderseits mit welchen Mitteln eine Realisierung möglich ist.

Nach Erhalt der Aufgabenstellung galt es anfänglich, ein 3D-Laser-Modul mit einem bestehenden 2D-Laser zu realisieren. Beim Projektbeginn im September 2017 wurde dies von Dr. Björn Jensen abgeändert, da nun ein 3D-Laserscanner für diese Aufgabe zur Verfügung stand. Der 3D-Laserscanner besitzt gegenüber dem 2D-Laser den Vorteil, dass bereits räumliche Messdaten zur Echtzeit übermittelt werden können. Zudem besitzt dieser auch

einen bedeutend grösseren Messbereich, welcher für die Erstellung von Umgebungsmodellen nötig ist. Beim zu erarbeitende Projekt handelt es sich um eine Realisierung eines funktionsfähigen Prototypen. Der Prototyp soll sich um eine Achse drehen und die Daten dem mobilen Roboter einmal pro Umdrehung zur Verfügung stellen. Das entwickelte Laser-Modul soll im Rahmen der Arbeit auf dem Packbot-Roboter getestet werden.

Ziel des Projektes ist die Realisierung eines 3D-Laser Moduls. Dabei wird die gesamte Hardware mit den gewählten Komponenten zusammengebaut. Die Software wird durch bestehende Codepakete und eigenen Erweiterungen auf die Aufgabenstellung angepasst. In erster Priorität soll damit 3D Mapping betrieben werden können. Das Modul wird mit dem bestehenden 3D-Laserscanner der Marke Velodyne des Typs VLP-16 realisiert. Dabei soll eine möglichst grosse räumliche Abdeckung der Umgebung erreicht werden. Diese wird in einer möglichst detaillierten Punktwolke modelliert. Zweite Priorität ist die Hinderniserkennung in Frontrichtung. Dazu muss in Frontrichtung eine detaillierte Punktwolke ermittelt werden können. Das Modul soll einerseits auf dem Packbot nutzbar, sowie auch eigenständig einsetzbar sein. Das Pflichtenheft im Anhang A, grenzt die Aufgabenstellung auf weitere Punkte ein. Alle Anhänge wurden in digitaler Form am Ende dieser Dokumentation auf einer CD hinterlegt.

Für die Aufgabenstellung eignet sich ein strukturierter Projektphasenablauf. Dabei werden nacheinander die Phasen Initialisierung, Informationsbeschaffung, Konzeption, Realisierung und die Testphase durchlaufen. Im Anhang A Pflichtenheft sind anfänglich abgeschätzter Aufwand, Arbeitsmittel und die zu erwartende Ergebnisse beschrieben. Des Weiteren beinhaltet es die Vorgaben der Aufgabe mit entsprechenden Kriterien. Der Inhalt der Dokumentation richtet sich nach den zu erarbeitenden Projektphasen. Da die Initialisierungsphase nur administrative Aufgaben beinhaltet wird diese Phase in dieser Dokumentation nicht näher erläutert. Die weiteren Phasen sind chronologisch mit entsprechenden Unterkapiteln im Inhaltsverzeichnis einsehbar.

Im Anhang B ist ein detaillierter Projektplan angefügt, welcher die einzelne Arbeitspakete und das Zeitmanagement aufzeigt. Im Kapitel 6 werden dazu noch Erläuterungen zu Abweichungen, Problemstellungen und Zeitplanänderungen getätigt. Zusätzlich werden persönliche Reflektionen über das gesamte Projekt getätigt.

# Kapitel 2

## Informationsbeschaffung

In einer ersten Phase wurde ein Zeitraum zur Informationsbeschaffung festgelegt. Dieser Abschnitt ist einerseits für die Themeneinarbeitung und anderseits für die Abgrenzung der Aufgabe und der Ziele erforderlich.

Nachfolgend werden die wichtigsten Erkenntnisse der Informationsbeschaffung erläutert, die massgebend für die Konzeption in Kapitel 3 und die Realisierung in Kapitel 4 sind. Dabei werden zu einzelnen Komponenten und Verfahren Stellung genommen und eruiert, ob diese sich für das Projekt eignen. Des Weiteren wird relevante Software erläutert, die für die Realisierung nötig ist. Ein weiterer Abschnitt behandelt bereits bestehende Lösungen, die den aktuellen Stand der Technik aufzeigen. Die einzelnen Unterkapitel wurden nach Funktionsblöcken unterteilt.

### 2.1 Entfernungsmeßung

In diesem Unterkapitel wird der bestehende 3D-Laserscanner Velodyne VLP-16 analysiert. Dabei werden wichtige Spezifikationen erläutert. Anhand der Spezifikationen und Schnittstellen des Velodyne werden in den nächsten Unterkapitel weitere Komponenten eruiert.

#### 2.1.1 Velodyne VLP-16 Puck

Beim Velodyne VLP-16 Puck handelt es sich um einen Echtzeit 3D-Laserscanner, der auf dem Light Detection And Ranging (LIDAR)-Verfahren basiert. Nachfolgende Angaben entstammen aus dem Datenblatt, wenn nicht anders referenziert. [LiD16]

Der VLP-16 bietet insgesamt 16 Laser-/Detektorpaare, die in Abbildung 2.1 ersichtlich sind. Mit diesen wird in horizontaler Lage ein Messbereich von  $360^\circ$  erreicht. Dies wird dadurch ermöglicht, dass der Laserscanner sich intern mit 5 - 20 Rotationen pro Sekunde (300 - 1200 RPM) um die eigene Achse dreht. Die Rotationsgeschwindigkeit ist softwaremässig einstellbar. Dabei wird mit einer horizontalen Auflösung von  $0.1^\circ$  –  $0.4^\circ$  gerechnet. Diese Auflösung begründet sich auf die nötige Zeit eines Messdurchgangs. Um alle 16 Laser abzufeuern und wieder zu entladen dauert es  $56 \mu\text{s}$ . Da die 16 Laser mit je  $2^\circ$  Unterschied ausgerichtet sind ergibt sich daraus ein vertikaler Messbereich von  $30^\circ$  mit einer vertikalen Auflösung von  $2^\circ$ .

Ein wichtiger Punkt ist somit, dass im stationären Zustand zwischen den Laserstrahlen keine Messpunkte ermittelt werden können. Um dies zu ermöglichen, muss der Laserscanner seine Position verändern.

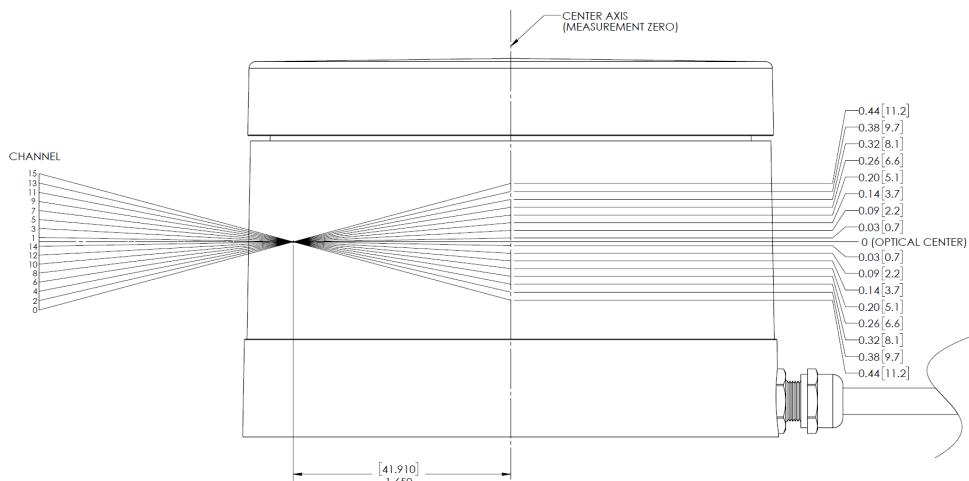


Abbildung 2.1: Laserstrahlen des Velodyne VLP-16  
[LiD16]

Eine relevante Eigenschaft dieses Laserscanners ist die grosse Messdistanz, die Distanzen zwischen 0.3 m bis 100 m ermöglichen. Dabei ist die typische Toleranz  $+/- 3 \text{ cm}$ . Der Reflektionsgrad wird in 256-bit Auflösung angegeben, d.h. dass der Sensor aus der zurückgesendeten Laserimpulsen die Intensität messen kann.

Der VLP-16 benötigt eine separate Interface Box, mit der die Speisung und die Datenschnittstellen zu einem 8-adrigen Kabel zusammengeführt werden. Die Adern 1-4 werden

für die Ethernet Datenübertragung benötigt. Die Adern 5 und 6 sind nur bei zugeschaltetem Global Positioning System (GPS) nötig, ansonsten sind diese unbenutzt. Die stabilisierte 12 Volt Spannung wird über die Adern 7 und 8 zugeführt. Die Interface Box ist in Abbildung 2.2 ersichtlich. Diese besitzt folgende Anschlüsse; eine 12 Volt Speisung, einen Ethernet RJ45-Anschluss und eine GPS Schnittstelle. Für die typische Leistungsaufnahme des Sensors wird 8 Watt angegeben. Aus einem Messaufbau konnte bei 12 Volt Speisespannung einen Strom von 0.6 Ampere ermittelt werden. Im Einschaltmoment kann der Strom kurzzeitig auf 0.9 Ampere ansteigen. Dies muss bei der Dimensionierung der Speisung beachtet werden.

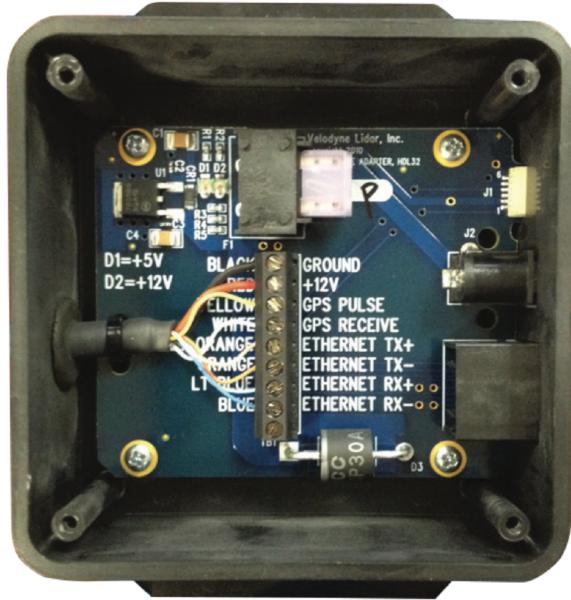


Abbildung 2.2: Ansicht auf die Interface Box

[LiD16]

Über die Ethernetverbindung werden die Daten- und Positionsdaten vom Velodyne an den Computer übermittelt. Dabei werden für die zwei verschiedenen User Data Protocol (UDP) Pakete die Ports 2368 und 8308 gebraucht. Nachfolgend wird in Abbildung 2.3 der Aufbau eines Datenpakets dargestellt.

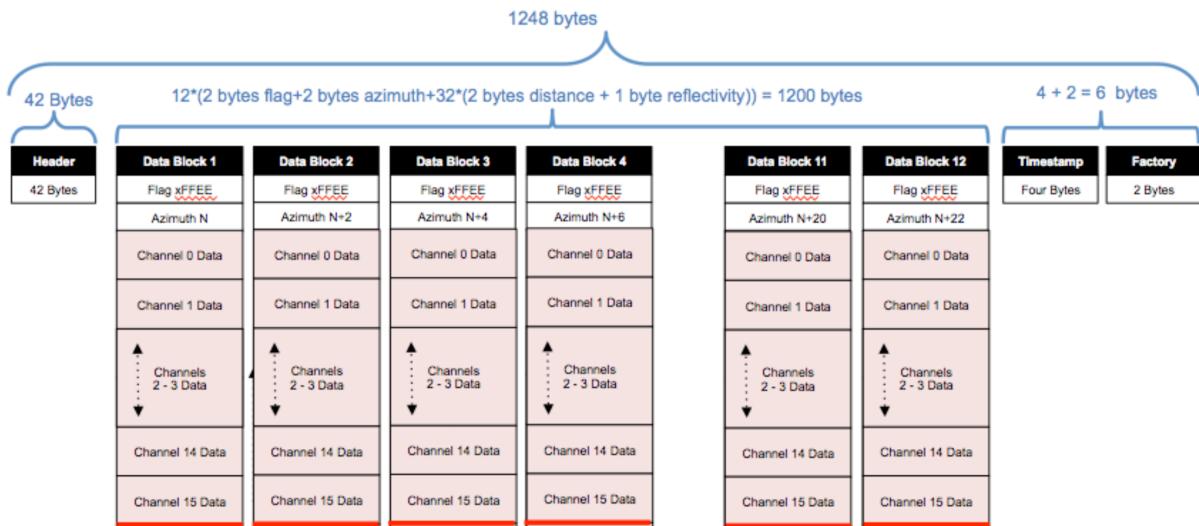


Abbildung 2.3: Aufbau Datenpaket

[LiD16]

Jedes Paket besitzt einen 42-Byte Header und einen Datenblock, der aus Laserrückgewert, kalibrierten Reflektionsgrad, Azimutwert und Zeitstempel besteht. Ein einzelnes Datenpaket besitzt die Grösse von 1248 Bytes und beinhaltet die Datensätze aller 16 Laserkanäle. Die typische Datenübertragungsrate wird angegeben mit 8 Mbit/s angegeben.

## 2.2 Stand der Technik

Dieses Kapitel dient als Vorstudie über den Einsatz des Velodyne VLP-16 durch bereits bestehende Projekte. Dabei werden zwei verschiedene Konfigurationen betrachtet und dazu entsprechend Vor- und Nachteile erläutert. Es handelt sich hierbei um zwei Teams, welche an der European Robotic Hackathon (EnRich) 2017 teilgenommen haben und als State-of-the-Art Projekte betrachtet werden.

### 2.2.1 IMM MSAS Team MSS Warschau

Das Institute of Mathematical Machines (IMM) in Warschau hat den Velodyne VLP-16 an einer endlos drehenden Konstruktion befestigt. Dabei ist der Sensor nicht in der üblichen Lage (Ausrichtung XY-Ebene), sondern um  $90^\circ$  abgedreht (Ausrichtung YZ Ebene). In Abbildung 2.4 ist die entsprechende Konfiguration abgebildet.

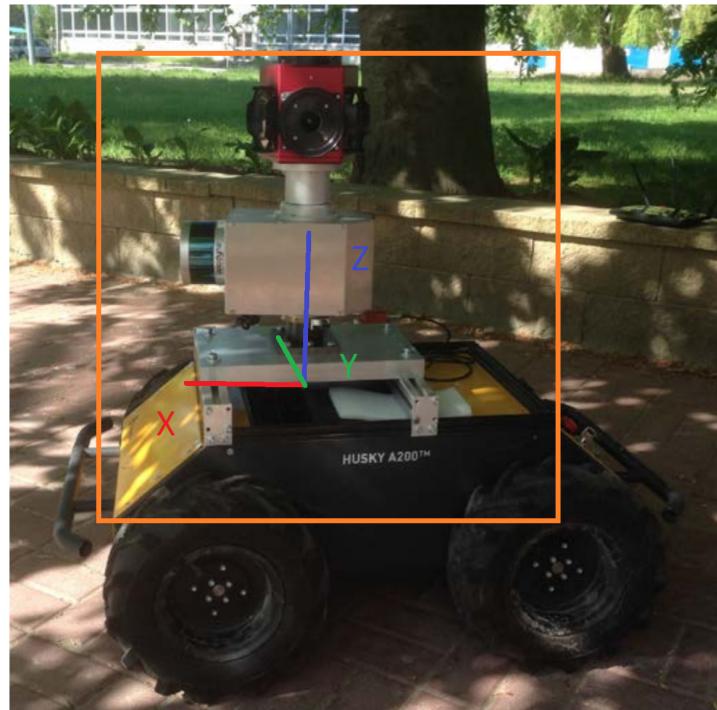


Abbildung 2.4: Roboter des Team IMM an der EnRich  
[Bed17]

In der nachfolgenden Betrachtung wird vom Koordinatensystem der Abbildung 2.4 ausgängen. Das Team nutzt bei dieser Konfiguration die begrenzte Auflösung des Sensors besser aus. Die vertikale und horizontale Auflösung wechseln dabei. Ist die Konstruktion nicht drehend, kann er in der Vertikalen den Bereich  $360^\circ$  mit der Auflösung von  $0.01^\circ$  -  $0.04^\circ$  messen. In der Horizontalen kann der Bereich  $30^\circ$  mit  $2^\circ$  aufgelöst werden.

Der Vorteil dieser Konfiguration wird jedoch erst durch die Rotation um die Z-Achse deutlich. Wird der Sensor nun kontinuierlich um die Z-Achse gedreht, verschieben sich die 16 horizontalen Laserstrahlen mit der Umdrehungsgeschwindigkeit der Konstruktion.

Einerseits bewegt sich der  $30^\circ$  grosse Messkegel durch den Raum und kann somit  $360^\circ$  in der Horizontalen vermessen. Anderseits kann durch die interne Rotation des Sensors  $360^\circ$  in der Vertikalen ausgemessen werden. Die schlechtere Auflösung von  $2^\circ$  kann somit kompensiert werden, da sich die Laserstrahlen kontinuierlich verschieben. Diese Konfiguration ermöglicht eine detaillierte Messung, da jeder Punkt im Raum von 16 Laserstrahlen mit der Auflösung zwischen  $0.1^\circ$  -  $0.4^\circ$  durchlaufen wird.

Die oben erläuterte Betrachtung gilt jedoch nur, wenn die Umdrehungsgeschwindigkeit der Konstruktion bedeutend langsamer als die interne Umdrehungsgeschwindigkeit des Sensors ist. Ansonsten besteht die Gefahr, dass die Reflektion des Laserstrahls nicht detektiert werden kann. Die Auflösung ist direkt von der Umdrehungsgeschwindigkeit der Konstruktion abhängig. Es wird zusätzlich davon ausgegangen, dass keine zusätzliche Translation, d.h. Bewegung des Roboters, stattfindet. Diese Konfiguration eignet sich somit, wenn das zu vermessende Gelände bekannt ist. Dadurch können vordefinierte Positionen angesteuert werden, an denen eine halbe Umdrehung ausreicht, um eine räumliche Messung an der definierten Position zu erstellen.

Weiter wird in der obigen Betrachtung die Abweichung zur Drehachse komplett vernachlässigt. Es entsteht ein Messfehler, der dem Abstand des Sensors zur Drehachse entspricht. Ein weiterer Messfehler kann durch Translation entstehen, wenn sich das Fahrzeug bewegt. Diese zwei Messfehler lassen sich jedoch durch Koordinatentransformationen in ein festes Koordinatensystem verschieben. Die Schwierigkeit dabei ist, die exakten Messewerte mittels einem Drehencoder und einer Inertiale Messeinheit (IMU) zum jeweiligen Zeitpunkt zu ermitteln.

Ein Nachteil für die Aufgabenstellung ist, dass bei dieser Konfiguration im freien Feld viele Messpunkte ins Leere messen, da himmelwärts (Richtung Z-Achse) keine Reflektion entsteht. Auch Messpunkte in Richtung negative Z-Achse müssen gefiltert werden, da ansonsten ständig die Oberfläche des Roboters selbst reflektiert wird.

### 2.2.2 Hector Tracker Team Hector Darmstadt

Das Team Hector der Universität Darmstadt besitzt auf dem Hector Tracker eine weitere Konfigurationsmöglichkeit. Auch dieses Team arbeitet mit einem endlos drehenden Konstruktion. Bei nachfolgenden Betrachtungen wird vom Koordinatensystem in Abbildung

2.5 ausgegangen. Der Velodyne VLP-16 befindet sich  $45^\circ$  abgeneigt zur XZ-Ebene. Dabei ist die Lage des Sensors zentral auf der Drehachse Z.

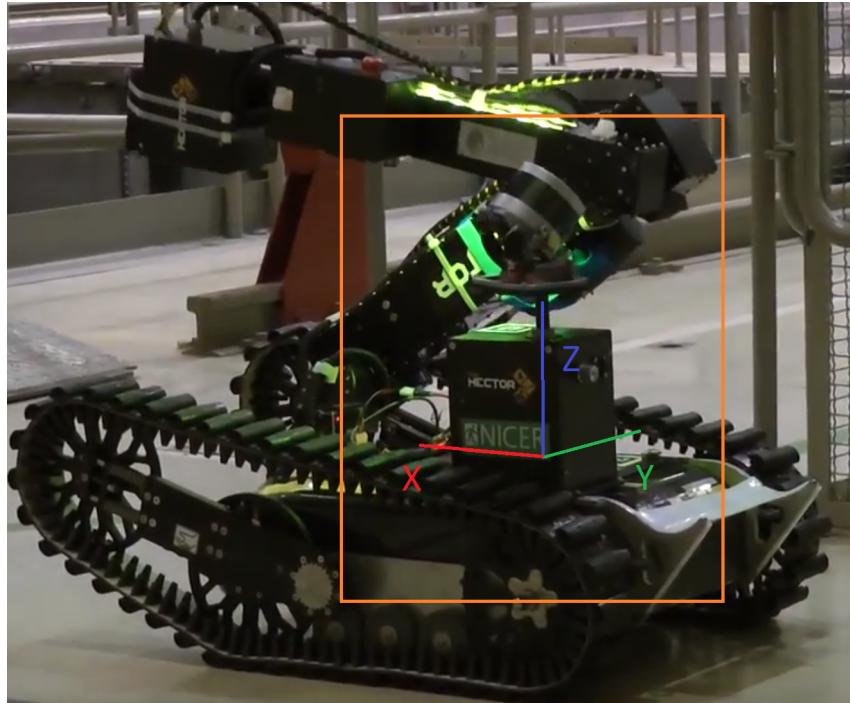


Abbildung 2.5: Roboter des Team Hector an der EnRich  
[Koh17]

Im stationären Zustand kann durch diese Konfiguration keine Verbesserung der Auflösung erreicht werden. Die Auflösungen bleiben erhalten, sind jedoch nun entsprechend der Neigung um  $45^\circ$  verschoben. Durch die Neigung des Sensors wird der Raum nicht gleichmäßig vermessen. Die räumliche Vermessung wird auch hier erst mit der Rotation um die Z-Achse ermöglicht. Gegenüber einer festen nicht drehenden Lösung besitzt diese Konfiguration somit den Vorteil, dass Messpunkte, welche sich zwischen den Laserstrahlen befinden, durch Drehen erreichen lassen. Dabei wird der  $30^\circ$  Messkegel durch die Rotation ständig im vertikalen Bereich hin- und hergeschoben.

Es kann während einer Umdrehung somit durch die Neigung von  $45^\circ$  und dem  $30^\circ$  Kegel des Sensors eine Abdeckung in der Vertikalen von  $240^\circ$  erreicht werden. Die horizontale Abdeckung bleibt hierbei weiterhin  $360^\circ$ , da der Sensor die interne Rotation vollführt.

Auch bei dieser Betrachtung muss die Umdrehungsgeschwindigkeit der Konstruktion bedeutend langsamer als die interne Umdrehungsgeschwindigkeit des Sensor sein. Die Auflösung bei dieser Konstruktion ist direkt zur Umdrehungsgeschwindigkeit abhängig. Des Weiteren muss bei dieser Konfiguration die Translation, d.h. die Bewegung des Roboters, noch separat korrigiert werden.

Im Vergleich zum Projekt der IMM entstehen bei dieser Konfiguration weniger Messpunkte in Richtung des Roboters und in Richtung der Z-Achse (himmelwärts). Dies ist für Rauminterne Messungen unvorteilhaft. Für die Messung im freien Feld ist diese Konfiguration besser geeignet, sofern sich keine hohen Objekte in der Nähe des Sensors befinden.

### 2.2.3 Schlussfolgerung

An der EnRich 2017 nutzen mehrere Teams den Velodyne VLP-16 mit unterschiedlichen Konfigurationen. Die zwei betrachteten Konfigurationen nutzen die Möglichkeit einer endlos drehenden mechanischen Konstruktion. Im Unterkapitel 2.5 werden mögliche Antriebsmöglichkeiten für eine solche Konstruktion evaluiert. Eine endlos drehende Konstruktion benötigt ein spezielles Handling der Kabelführung, daher wird im Unterkapitel 2.7.2 die Möglichkeit eines Schleiftrings beschrieben. Beide Konfigurationen besitzen Vor- und Nachteile bei der Umgebungserkennung. Um eine möglichst detaillierte Umgebungswolke zu erstellen müssen vorwiegend Translation und Rotation um die Drehachse kompensiert werden. Diese lassen sich durch Koordinatentransformation und entsprechender Sensorik ermöglichen. In Unterkapitel 2.6 werden dazu mögliche Varianten zur Positionsbestimmung erläutert.

## 2.3 Software

In diesem Kapitel wird die notwendige Software beschrieben. Es erläutert einerseits das Robot Operating System (ROS) und dessen Funktion im Projekt. Daneben werden weitere Softwareapplikationen und -packages erwähnt, welche für die Aufgabenstellung nützlich sind.

### 2.3.1 ROS Robot Operating System

Die gesamte Kommunikation mit Sensoren und Aktoren findet auf dem Packbot mit ROS statt. Da das 3D-Laser-Modul auf dem Packpot, sowie selbstständig funktionieren soll, ist ROS nicht zwingend einsetzbar. Dennoch wurde für die Aufgabenstellung ROS gewählt. Im Zusammenhang mit Velodyne und 3D Mapping bietet ROS neben Visualisierungstools und bereits bestehender Treiber-Packages auch die Integrationen der Point Cloud Library (PCL). Die PCL ist eine umfangreiche Softwarebibliothek, welche viele Funktionen und Codebeispiele bereit stellt, um Punktwolken zu erstellen.

Grundsätzlich wird ROS, wegen seiner Nähe zu Linux Distributionen, auf einem Ubuntu Betriebssystem aufgesetzt und ist ein Software-Framework, dass die Programmiersprachen C++ und Python nutzt. Diese in 2007 entwickelte Open Source Software erhielt in den letzten Jahren ständig neue und überarbeitete Versionen.

Die Empfehlung von ROS und diversen Literaturen liegt bei der neusten Distribution ROS "Kinetic Kame". [Jos17] Es handelt sich um eine Langzeitversion, welche bis 2021 Unterstützung bietet. Im Zug der ersten Versuchen mit ROS wurde auf einem Laptop mit X64 (AMD64)-Architektur gearbeitet. Auf diesem wurde ROS Kinetic Kame vollumfänglich ermöglicht. Damit das 3D-Lasermodul selbstständig agieren kann, wird jedoch ein einbaubarer Einplatinencomputer benötigt. In Unterkapitel 2.4 werden aktuelle Einplatinencomputer beschrieben .

Während der Einarbeitung mit Kinetik Kame konnten einige Nachteile der Distribution festgestellt werden. Es wurde festgestellt, dass diverse Packages nur für AMD64-Architekturen zur Verfügung stehen und nicht für Acorn Risc Machine (ARM)-Architekturen. [tfo16] Dies könnte zu Eingrenzungen während der Realisierung führen, daher werden nötige Packages geprüft.

### 2.3.2 Velodyne Package

Wie bereits im vorherigen Kapitel erwähnt, besitzt ROS ein Treiber-Package für den Velodyne VLP-16. Dieses Paackage wird für ARM- und AMD64-Architekturen ermöglicht. In der Abbildung 2.6 ist der Aufbau dieses Treibers mit dem ROS-Tool rqt-graph visualisiert.

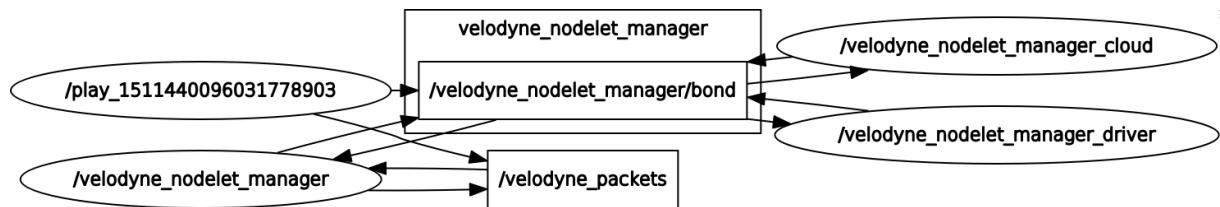


Abbildung 2.6: Velodyne Treiber aus rqt-graph

Dieses Package vereinfacht die Sensordatenverarbeitung bedeutend. Mittels dem Package werden die hexadezimalen Rohdaten von den Ethernet-Datenpaketen aus Abbildung 2.3 bereits ausgewertet und kalibriert zur Verfügung gestellt. Dabei laufen mehrere Programmsegmente (*Nodes*) parallel.

Für die Aufgabenstellung ist nur der Node `velodyne_packets` relevant, da dieser den verarbeiteten Sensor-Datenstream zur Verfügung stellt. Es werden dabei zwei unterschiedliche Typen von *Messages* kontinuierlich ausgegeben (*published*). Die Message `velodyne_packets` (*vom Typ: velodyne\_msgs/VelodyneScan*) beinhaltet die Rohdaten eines einzelnen Laserstrahls. Die Message `velodyne_points` (*vom Typ: sensor\_msgs/PointCloud2*) beinhaltet die angesammelten Datenpunkte, welche bereits zu einem festen Koordinatensystem fixiert sind. [ros16a]

Das Package bietet die Abstraktion der Datenerfassung und Integration in ein festes Koordinatensystem. Werden die PointCloud2 Messages zusammengefügt und mittels Koordinatentransformation räumlich angepasst, lässt sich somit eine Umgebungsabbildung erzeugen.

bild

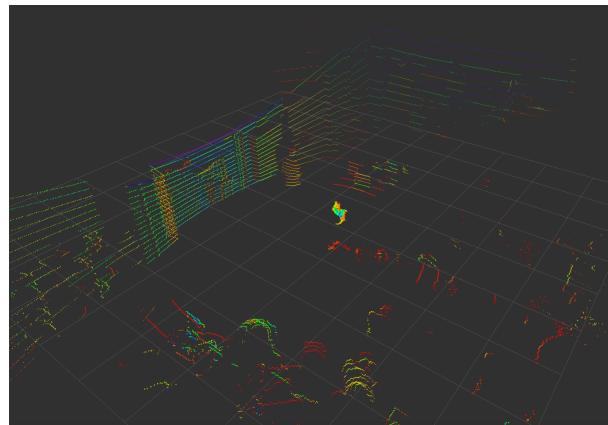


Abbildung 2.7: PointCloud2 Datenstream mit Rviz

Mittels dem ROS-Tool Rviz lässt sich der Datenstream über `velodyne_packets` direkt visualisieren. In Abbildung 2.7 ist ein statischer Datenstream visualisiert. Dies geschieht indem das Tool kontinuierlich die Messages empfängt. (*subscribed*).

ROS bietet zudem die Möglichkeit über den `rosbag` Befehl, Messungen aufzunehmen und abzuspeichern. Dies ermöglicht somit nicht nur den aktuellen Sensordatenstream zu visualisieren, sondern auch getätigte Messungen zu archivieren [ros17].

### 2.3.3 Schlussfolgerung

Für die Aufgabenstellung eignet sich ROS gut, wegen den bereitstehenden Tools und den Packages, mit denen ein grosser Teil der Software abstrahiert werden kann. Der aktuelle Code muss um einige Nodes erweitert werden. Die einzelnen Sensordaten werden einerseits auf die aktuelle Position verschoben und anderseits zu einer einzigen Punktwolke zusammengefüg. Die Einarbeitung mit ROS ist jedoch relativ umfangreich und bietet eine Vielzahl an Tools, die zuerst kennen gelernt werden müssen.

## 2.4 Datenverarbeitung

Um die Datenmenge zu verarbeiten und die Ansteuerung der Komponenten zu realisieren, eignen sich Einplatinencomputer. Einplatinencomputer besitzen den Vorteil, dass sie gegenüber üblichen Mikroprozessoren grössere Speichermöglichkeiten, höhere Prozessorleistungen und bootbare Betriebssysteme ermöglichen. Gegenüber Barebone PC (XPC) besitzen sie den Vorteil, dass sie Hardware näher sind, indem sie direkt ansteuerbare Pins besitzen (GPIOs). Aus diesem Grund werden nachfolgend diverse Einplatinencomputer betrachtet. Kriterien bei der Auswahl eines geeigneten Boards sind Prozessorleistung, Random Access Memory (RAM), Preis, Ethernet-Schnittstelle, Speichermöglichkeit, GPIO-Verfügbarkeit und die verbaubare Dimension. Diese Kriterien wurden durch die Vorgaben des Pflichtenhefts im Anhang A festgelegt. Geeignete Prozessorleistung, Speichermöglichkeit und RAM sind für die erfolgreiche Verarbeitung der Sensordaten nötig, da die Punktwolke mit zunehmender Zeit an Datengröße zunimmt. Die Datenübermittlung des Velodyne und

des Packpots erfolgt über Ethernet, daher benötigt der Einplatinencomputer mindestens einen RJ45-Anschluss.

### 2.4.1 Raspberry Pi 2 & 3

Das Raspberry Pi ist eines der bekanntesten Einplatinencomputer und bietet daher eine grosse Community. Da bereits ein Raspberry Pi 2 zur Verfügung gestanden ist, konnten die ersten Erfahrungen mit einem Raspberry Pi gemacht werden. Das Raspberry bietet zusammen mit ROS Kinetic Kame und Ubuntu Mate LTS 16.04 eine Lösung für die Datenverarbeitung. Das Raspberry Pi 2 bzw. 3 basiert auf einem Broadcom System-on-a-Chip (SOC) und ist mit einem ARM Cortex A7 bzw. A53 Prozessor mit vier Kernen ausgestattet. Die Taktfrequenz liegt bei diesem lediglich bei 1.2 GHz. Beide Modelle besitzen 1 GB RAM. Das Betriebssystem wird auf einer Secure Disk Memory Card (SD) gebootet. Speichermöglichkeiten sind über USB oder die SD-Karte vorhanden. Auch GPIOs und eine Ethernetschnittstelle sind beim Raspberry Pi vorhanden. Der Preis eines Raspberry Pi liegt momentan bei ca 50 Fr. [Tea16]

### 2.4.2 Banana Pi M3

Der Banana Pi M3 bietet zur Zeit (Stand Oktober 2017) die höchste Performance bei Einplatinencomputern mit ARM-Architektur durch den Allwinner A83T Achtkern-Prozessor, der mit 1.8 GHz taktet und den 2 GB RAM. Neben USB-Anschlüssen bietet es eine SATA-USB-Schnittstelle, die den internen 8 GB embedded Multimedia Card (eMMC)-Speicher um bis zu 2 TB erweitern lässt. Es bietet auch direkt integrierte WLAN-Schnittstellen, sowie eine RJ45 Gigabit Netzwerkschnittstelle. Der Preis eines BananaPi M3 liegt momentan bei ca 90 Fr.. Im Zusammenhang mit Banana Pi M3 und Ubuntu werden mehrfach Komplikationen und Probleme veröffentlicht [Tea16] [TEA16]. Dies ist ein bedeutender Nachteil für dieses Einplatinencomputer.

### 2.4.3 Odroid C2 & XU4

In diversen Literaturen (siehe [Jos17] Kapitel 4) werden neben dem Raspberry Pi, Odroid Boards als empfohlene Einplatinencomputer aufgelistet. Dabei stehen die aktuelle Modelle Odroid-C2 oder Odroid-XU4 zur Verfügung. Sie bieten eine höhere Prozessorleistung, 1.5 GHz bzw. 2 GHz mit je 2 Gigabyte RAM. Betriebssysteme können via SD oder eMMC gebootet werden. Beide Boards sind jedoch noch nicht lange auf dem Markt und bieten in vielen Anwendungen nur Beta-Versionen. Vor allem die Unterstützung von Ubuntu LTS 16.04 ist nicht restlos geklärt. Der Preis dieser Boards ist ca. 80 - 90 Franken [Tea16].

### 2.4.4 Up Board Squared

Dieser Einplatinencomputer unterscheidet sich wesentlich von den bisherig betrachteten Boards. Der Prozessor arbeitet nicht mit ARM-Architektur, sondern mit AMD64-Architektur. Somit können Ubuntu und ROS Distributionen voll umfänglich genutzt werden. Er besitzt mit Intel Pentium ein vier kerniger Prozessor und taktet mit 2.5 GHz. Zusätzlich bietet er einen separaten 500 MHz Grafikprozessor (GPU) und bis zu 8 GB RAM. Neben USB werden auch zwei RJ45-Schnittstelle und ein GPIO Pinlayout angeboten, welches dem des Raspberry Pis entspricht. Das Up Board Squared bietet sich an, falls die Performance und die Kompatibilität der ARM-Architektur mit ROS nicht ausreicht. Der Preis eines Up Board Squared kostet je nach Ausführung zwischen 180-380 Fr.

### 2.4.5 Schlussfolgerung

Für die Aufgabenstellung eignet sich lediglich das Raspberry Pi 2 oder 3, um ROS mit Ubuntu LTS 16.04 zu betreiben. Das Banana Pi und die Odroid Boards sind im Kriterium Speichermöglichkeit und Prozessorleistung besser geeignet, können jedoch wegen fehlender Betriebssystem-Kompatibilität nicht genutzt werden. Die Dimensionen aller betrachteten Boards sind, mit Ausnahme des Up Board Squared sehr kompakt. Im Punkt Ethernetschnittstelle bietet lediglich das Up Board Square zwei Ethernetanschlüsse, mit welchen ein zusätzlicher Ethernet Switch hinfällig wird. Das Up Board Squared bieten in

fast allen Kriterien eine bessere Lösung. Aus Kostengründen wurde für den zu erarbeitenden Prototypen das Raspberry Pi als genügend bewertet. Für eine allfällige Optimierung bietet sich das Up Board Squared an.

## 2.5 Antriebsmöglichkeiten

Um den Velodyne VLP-16 um eine Achse drehen zu lassen, müssen Motoren eingesetzt werden. Nachfolgend werden zwei verschiedene Motorenarten beschrieben, die sich für die Aufgabenstellung eignen. Wichtige Kriterien für die Aufgabenstellung sind einerseits die Ansteuerung und die Dimension. Andererseits muss die Möglichkeit bestehen die Winkeländerung zu eruieren.

### 2.5.1 Schrittmotor

Der Schrittmotor ist eine Antriebsmöglichkeit, welche für das Projekt in Frage kommt. Es gibt sehr kostengünstige und kompakt dimensionierte Motoren dieser Art. Ein interessanter Aspekt ist das gezielte Steuern des Motors. Durch einen Stromimpuls bewegt sich ein Schrittmotor nur einen festgelegten Winkelschritt weiter. Er kann bereits ohne zusätzliche Sensorik definierte Schritte anfahren, aus denen die Winkeländerung eruiert werden kann. Schrittmotoren besitzen die Eigenschaft, dass in der Ruhelage ein Haltemoment entsteht. Diese Eigenschaft wird jedoch für die Aufgabenstellung nicht zwingend benötigt. Preislich muss bei einem geeigneten Schrittmotor mit 40 Fr. gerechnet werden.

Nachteilig für die Aufgabenstellung am Schrittmotor ist der höhere Stromverbrauch, vor allem zum Aufbringen des Haltemoments. Da nur ein Schritt ausgeführt wird, wenn das entsprechende Drehmoment nicht überschritten wird, müsste dieses sorgfältig berechnet werden. Ein bedeutender Nachteil im Zusammenhang mit der Aufgabenstellung ist, dass durch Schrittverluste die Winkeländerung nicht mehr quantitativ ermittelt werden kann. Schrittmotoren mit integrierten Encodern würden in diesem Fall Abhilfe schaffen. Aus Kostengründen (Preise ab 300 Fr. [Mou17]) wurden Schrittmotoren mit integrierten Absolutencodern nicht weiter vertieft. Ein weiterer Nachteil ist das verhältnismäßig hohe

Gewicht. Dies ist kein Kriterium für die Aufgabenstellung, sollte jedoch bei der Realisierung beachtet werden.

Um mit einem Einplatinencomputer einen Schrittmotor anzusteuern, empfiehlt sich ein Schrittmotorentreiber. Mit solchen Treibern lässt sich der Motor mittels 2 Steuerpins rotieren. Um die Drehgeschwindigkeit zu senken, bieten diese Treiber die Möglichkeit, die Schritte in 2, 4, 8 und 16 Teilschritte zu senken. Es ermöglicht zudem eine feinere Bewegung. [Pol17a].

## 2.5.2 Gleichstrommotor

Als Alternative zum Schrittmotor bietet sich ein üblicher Gleichstrommotor. Diese Motoren sind für viele Einsatzbereiche geeignet und es gibt sie in verschiedenen Größen und Umdrehungszahlen. Im Gegensatz zu Schrittmotoren laufen Gleichstrommotoren kontinuierlich, aufgrund eines Stroms, der durch die Wicklungen fließt. Nachteilig ist somit, dass weder die genaue Anzahl Umdrehungen noch die momentane Phasenlage bekannt ist. Es gibt jedoch eine Vielzahl an Möglichkeiten diese zu eruieren. Einerseits gibt es die Möglichkeit mittels Hall-Sensoren oder mittels Quadraturencodern, die aktuelle Drehrichtung und die Drehzahl zu ermitteln. Um dies zu ermöglichen, braucht es zusätzlich einen Vierquadrantensteller (H-Brücke), sowie eine Sensorlogik, damit der Sensor mittels Pulsweitenmodulation (PWM) angesteuert werden kann. Im Preissegment um 40 Fr. gibt es leistungsfähige Gleichstrommotoren für den Anwendungsbereich [Cor17].

## 2.6 Positionsbestimmung

Damit die Messdaten des Velodyne auf eine feste Koordinatenachse abgebildet werden können, benötigt eine drehende Konstruktion eine absolute Positionsbestimmung. Nachfolgend sind Varianten erläutert, welche sich dafür eignen.

### 2.6.1 LED und Photodiode

Eine simple Variante ist der Einsatz einer LED, welche im rotierenden Teil konstant leuchtet. Dabei wird die Photodiode an den stationären Teil angebracht und als Nullpunkt genutzt. Durch das Licht der LED wird ein Strom an die Photodiode übertragen und dies kann mit einer einfachen Schaltung über einen GPIO gemessen werden. Nachteilig bei dieser Variante ist der Lichtkegel der LED. Die LED müsste so verbaut werden, dass ein fokussierter Lichtstrahl auf die Photodiode fällt. Ansonsten kann der Nullpunkt stark abweichen. Zudem ist diese Variante sehr stark abhängig vom Umgebungslicht. Diese Variante bedingt, dass mittels einer weiteren Komponente die Winkeländerung gemessen werden kann.

### 2.6.2 Sensor QRE 1113

Eine weiter ähnliche Variante ist der Einsatz eines QRE 1113 Sensors. Dieser Sensor baut auf dem Prinzip der Infrarotreflektion auf. Dabei werden dunkle Oberflächen schlechter reflektiert, als helle Oberflächen. Diese Eigenschaft wird genutzt, um den Nullpunkt zu detektieren. Auch diese Variante bedingt, dass mittels eines weiteren Komponenten die Winkeländerung gemessen werden kann.

Eine weitere Variante ist die Aneinanderreihung dieser Sensoren. Dies bietet die Möglichkeit einen absoluten Encoders mit Binär- oder Gray-Code, wie in Abbildung 2.8 dargestellt, zu realisieren. Es können mit  $N$  Sensoren  $2^N$  Zustände unterschieden werden. Aus den Zuständen kann der Winkel berechnet werden. Für diese Aufgabe werden jedoch  $N$  GPIOs benötigt.

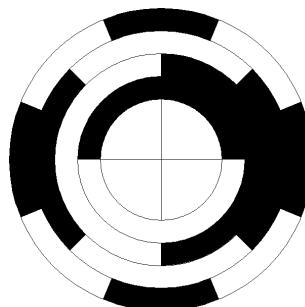


Abbildung 2.8: Quadraturencoder mit 4 Sensoren

Auch diese Variante ist stark abhängig vom Umgebungslicht und zusätzlich können die einzelnen Sensoren einander durch Streuung beeinflussen. Um dies zu minimieren, sollte der Absolutencoder mittels Gray-Code realisiert werden, damit nur immer eine gleichzeitige Zustandsänderung stattfindet. Die Abtastung der Zustände muss nach dem Nyquist-Shannon-Abtasttheorem mit mindestens doppelter Abtastfrequenz gemessen werden.

## 2.7 Speisung und Verkabelung

Dieses Kapitel erläutert weitere Komponenten, welche je nach Konzept nötig sind, um die Speisung und Verkabelung zu ermöglichen.

### 2.7.1 Abwärtswandler

Der Abwärtswandler D24V50F5 von Pololu eignet sich sehr gut für die Stromversorgung von Einplatinencomputern. Er bietet eine stabile 5 Volt Ausgangsspannung bis zu einer ausgangsseitigen Belastung von 8 Ampere. Der Wirkungsgrad beläuft sich dabei auf über 90 Prozent bei einer eingesetzten Betriebsspannung von 12 Volt. Diese Angaben gehen aus Abbildung 2.9 hervor, die aus dem Datenblatt des Abwärtswandlers stammen.[Pol17b]

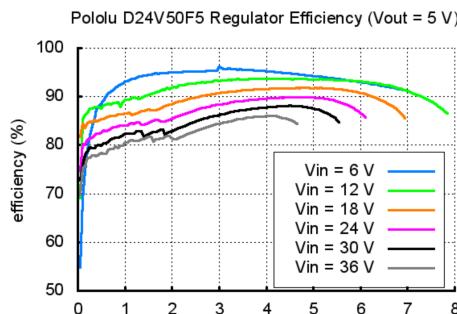


Abbildung 2.9: Wirkungsgrad D24V50F5 [Pol17b]

### 2.7.2 Ethernet Schleifring

Da übliche Kabel nur für einige wenige Verdrehungen ausgelegt sind und so bei drehenden Vorrichtungen kaputt gehen, wird ein Schleifring als Drehübertrager eingesetzt.

Ein Schleifring überträgt das Eingangssignal über eine Bürste und bildet somit einen Gleitkontakt. Die Problematik liegt beim Velodyne VLP-16, dass die Datenübertragung über Ethernet übermittelt werden muss. Gängige Schleifringe bieten kein Gewähr für die Übertragung von Ethernet, da die Datenrate verhältnismässig hoch ist. Es gibt jedoch einige wenige Hersteller aus dem amerikanischen und asiatischen Markt, welche Ethernet-Übertrager vertreiben. Ein Ethernet-Schleifring, der zusätzlich Leistungsübertragung bis 5 Ampere zulässt kostet nach Abklärungen bei diversen Herstellern zwischen 120 - 600 Fr.

### 2.7.3 Ethernet Switch

Die Einplatinencomputer aus Kapitel 2.4 benötigen mit Ausnahme des Up Board Squared ein weiteren Ethernetanschluss, damit die Kommunikation gewährleistet ist. Ethernet Switches gibt es in diversen Variationen und Grössen. Für die Aufgabenstellung wird ein möglichst kompakter Switch benötigt, der mindestens drei Anschlüsse bietet. Im Preissegment von 30Fr. können bei Mouser, Digikey und Farnell sehr kompakte 5-Port Switch bestellt werden. Eine gute Lösung ist dafür der Ethernet Switch GS105 der Marke Netgear. Dieser benötigt eine 12 Volt Speisung, mit dem sich eine Spannungsanpassung erübrigen würde. Zudem ist er im direkten Vergleich das Produkt mit den kleinsten Abmessungen.

## 2.8 Zwischenfazit

Für die Einarbeitung in das Projekt ist die Informationsbeschaffung bzw. Recherche ein wesentlicher Bestandteil. Dabei werden wichtige Grundlagen für die Konzeption geschaffen. ROS bietet für 3D Mapping gute Tools und vereinfacht die Datenverarbeitung des Velodyne VLP-16 bedeutend. Mittels Schrittmotoren oder Gleichstrommotoren kann der Velodyne gedreht werden. Mit entsprechender Sensorik können damit die räumliche Messungen erfolgen. Für die Datenverarbeitung stehen mehrere Einplatinencomputer zur Verfügung, dabei müssen diese jedoch die Kriterien erfüllen. Für endlos drehende Vorrichtungen wird zusätzlich ein Ethernet Schleifring benötigt, da ansonsten die Kabelführung nicht gewährleistet ist.

# Kapitel 3

## Konzeption

Dieses Kapitel beschreibt die Resultate und Ergebnisse aus der Konzeptionsphase. Nachfolgend werden auf die verschiedenen Konzeptionsvarianten eingegangen. Dabei werden Überlegungen und Bezüge zur Aufgabenstellung gemacht. Die Varianten werden abschliessend mit einem kurzen Fazit beurteilt. Danach wird mit einem morphologischen Kasten die Komponenten evaluiert für das gewählte Konzept.

### 3.1 Konzeptionsgrundlage

Als Konzeptionsgrundlage dient das Pflichtenheft, welches im Anhang A einsehbar ist. Wie darin erwähnt, soll das Modul möglichst frei drehend konzipiert werden. Dies ist das ausschlaggebendste Kriterium für die Bauform des Moduls. Da der Velodyne VLP-16, wie in Kapitel 2.1.1 erläutert, einen begrenzten vertikalen Öffnungswinkel besitzt, wurde bei den folgenden Konzepten die Möglichkeit eines beweglichen Moduls geprüft. Durch einen grösseren Öffnungswinkel kann der Raum besser ausgemessen werden. Ein weiteres relevantes Kriterium ist die Einsatzmöglichkeit des Moduls. Es soll einerseits auf dem Packbot, sowie auch als eigenständiges Produkt funktionieren. Daher sind als Schnittstellen einen Speiseanschluss, welcher 12 Volt Gleichstrom liefert und eine Ethernet RJ45-Anschluss nötig. Nachfolgend sind auf diesen Grundlagen drei Varianten dargelegt.

### 3.2 Variante 1: Plattform

Die Variante 1 Plattform ist in der Skizze in Abbildung 3.1 ersichtlich. Bei dieser Konstruktion werden alle elektronischen Komponenten, welche für die Signalverarbeitung und

die Energieversorgung nötig sind, in einem rechteckigen Gehäuse im unteren Teil verbaut. Die Interface Box des Velodyne VLP-16 wird auch in diesem Gehäuse untergebracht. Lediglich der Laserscanner VLP-16 befindet sich ausserhalb des Gehäuses, nämlich auf der Plattform oberhalb des Gehäuses.

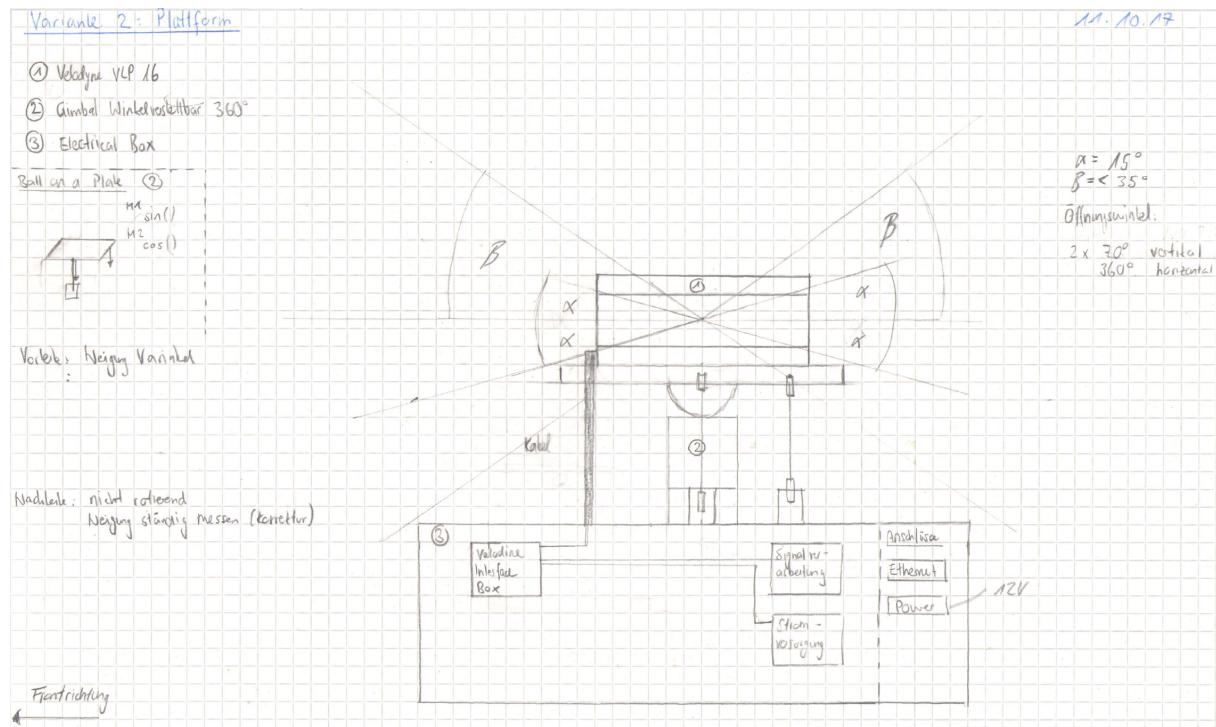


Abbildung 3.1: Skizze Variante 1

Die Eigenheit dieser Konzeption ist die Plattform, auf welcher sich der Velodyne VLP-16 befindet. Der Einsatz dieser Plattform erklärt sich durch ein bekanntes Regelungsexperiment namens "Ball on a Plate". In Abbildung 3.2 ist eine CAD-Zeichnung eines solchen Regelungsexperiment dargestellt. Bei "Ball on a Plate" werden zwei Servomotoren, welche je mit einem Gelenk mit einer Seite der Plattform verbunden sind, angesteuert. Indem die Servomotoren mittig auf der Seite mit der rechteckigen Plattform verbunden sind, kann durch Drehen der Servomotoren die Plattform zweiachsig geneigt werden. Im Regelungsexperiment kann durch Zuhilfenahme einer PID-Regelung ein Ball darauf balanciert werden. [Ben15]

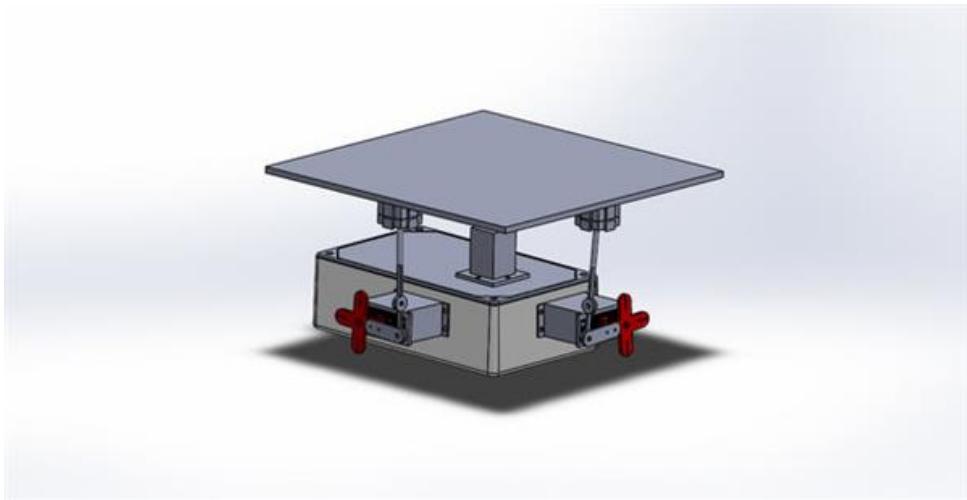


Abbildung 3.2: Ball on a Plate CAD [Ben15]

Diese Überlegungen bieten die Möglichkeit den Laserscanner VLP-16, welcher in der Konzeption auf der Plattform befestigt ist, durch das Ansteuern der Servomotoren in alle Richtungen zu neigen. Somit kann der Öffnungswinkel in jede Richtung vergrößert werden. Die Grenzen liegen dabei im Wesentlichen bei der mechanischen Begrenzung durch die Verbindungsstifte von Platte zu Servomotor.

Hauptproblematik bei dieser Konzeption ist die zusätzlich nötige Sensorik. Durch die Neigung der Plattform kann ohne zusätzliche Sensorik nicht auf die Messausrichtung zurück geschlossen werden. Dies erschwert die Visualisierung der 3D-Laserscanner Messdaten. Die Plattform müsste mittels einer eigenen IMU, welche aus Gyrosensor, Accelerometer und Magnetometer besteht, erweitert werden. Zusätzlich ist die Realisierung der Konzeption sehr stark abhängig von den mechanischen Komponenten, da die Verbindungsstifte von Servomotor und Grundplatte die Winkeländerung festlegen.

### 3.3 Variante 2: Turm stationär

Das zweite Konzept ist eine turmartige Konstruktion, die in Abbildung 3.3 dargestellt ist. Die Idee dazu lieferte das Unterkapitel 2.2.1. Dabei befindet sich der Velodyne VLP-16 abgesetzt von den elektrischen Komponenten in der Höhe.

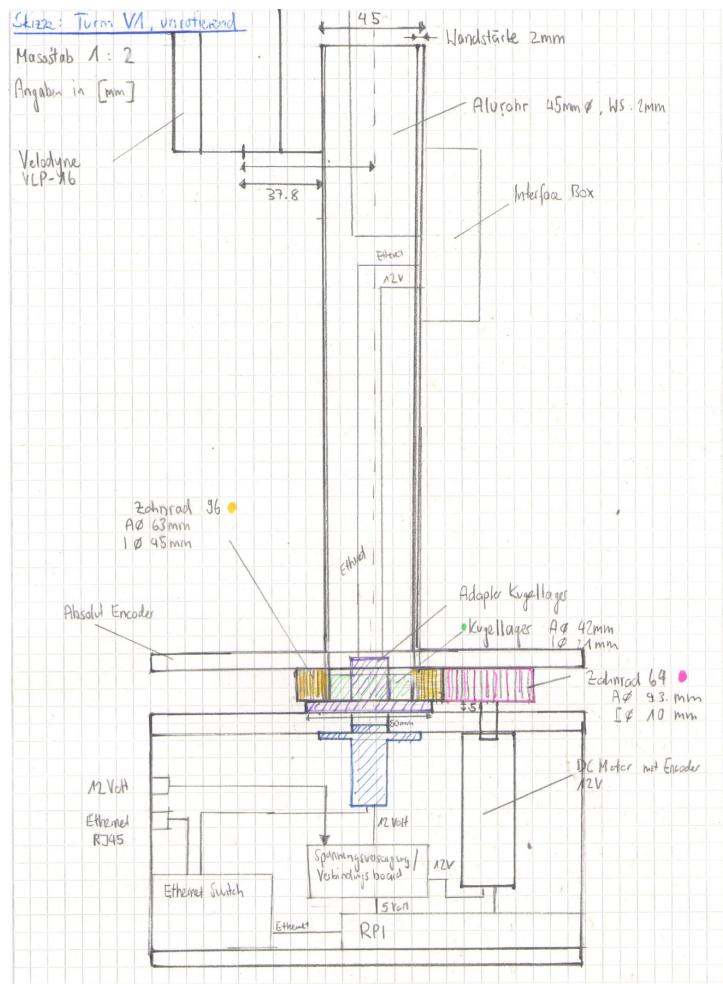


Abbildung 3.3: Skizze Variante 2

Eigenheit dieses Konzept ist, dass sich der Turm endlos drehen lässt. Diese Konfiguration verbessert die Problematik des Öffnungswinkels, welche in Unterkapitel 2.1.1 erläutert ist. Durch die interne Rotation des VLP-16 und den drehenden Turm können Messungen in alle Richtungen getätigt werden.

In dieser Konfiguration wird der VLP-16 mit einem Motor, der über zwei Zahnräder mit dem Alurohr verbunden ist, gedreht. Der untere Teil des Moduls, in dem sich die Datenverarbeitung befindet, ist stationär. Mittels einem Adapter, an dem ein Kugellager befestigt ist, kann eine möglichst geringe Reibung erzielt werden.

Die gesamte Datenverarbeitung und Ansteuerung wird im stationären Teil getätigter. Die

Kabelverbindung des VLP-16 wird über einen Schleifring nach unten geführt. Die Interface Box wird in der Skizze im stationären Teil verbaut. Es ist jedoch auch möglich diese direkt gegenüber des Velodyne zu befestigen. Die Ansteuerung der Motoren und Sensoren, sowie die Datenverarbeitung wird mittels Einplatinencomputer getätigt.

Mittels Nullpunktterkennung und Drehencoder können die aktuellen Positionen an den Einplatinenencoder übermittelt werden.

Nachteil dieser Konfiguration ist, dass es schlecht erweiterbar ist. Da der Schleifring begrenzte Kabeldurchführungen besitzt, können keine weiteren Komponenten beim drehenden Teil angeschlossen werden. Ein weiterer Nachteil ist die Problematik, wenn die Rotationsgeschwindigkeit des Turms auf mehrere Umdrehungen pro Minute ansteigt. Es können so fehlerhafte Messresultate beim VLP-16 entstehen. Die Drehgeschwindigkeit des Turms muss verhältnismäßig langsam gegenüber der Drehgeschwindigkeit VLP-16 sein.

Die mechanische Konstruktion ermöglicht auch die Umpositionierung des Velodynes. Wird der Sensor oberhalb des Alurohr befestigt, kann auch die zweite betrachtete Konfiguration des Teams Hector in Unterkapitel 2.2.2 ausgetestet werden.

### 3.4 Variante 3: Turm rotierend

Das dritte Konzept ist wiederum eine Turmartige Konstruktion. Eine Skizze ist in Abbildung 3.4 ersichtlich. Es handelt sich hier um eine abgeänderte Version von Variante 2. Dabei befindet sich der Velodyne VLP-16 erneut abgesetzt von den anderen elektrischen Komponenten. Die wesentlichsste Änderung liegt darin, dass nun auch die Datenverarbeitung im unteren Teil mitdreht. Dies wurde ermöglicht indem alle mechanischen Drehkomponenten und der Motor umgedreht wurden. Somit kann das Problem der Erweiterbarkeit des vorherigen Konzepts gelöst werden. Die gesamte Elektronik und Sensorik dreht nun mit. Lediglich die Schnittstellen Ethernet und 12 Volt Speisung führen über den Schleifring zum Packpot. Für diese Konfiguration wird lediglich zusätzlich ein Alusockel benötigt, welche das Alurohr mit dem Gehäuse verbindet.

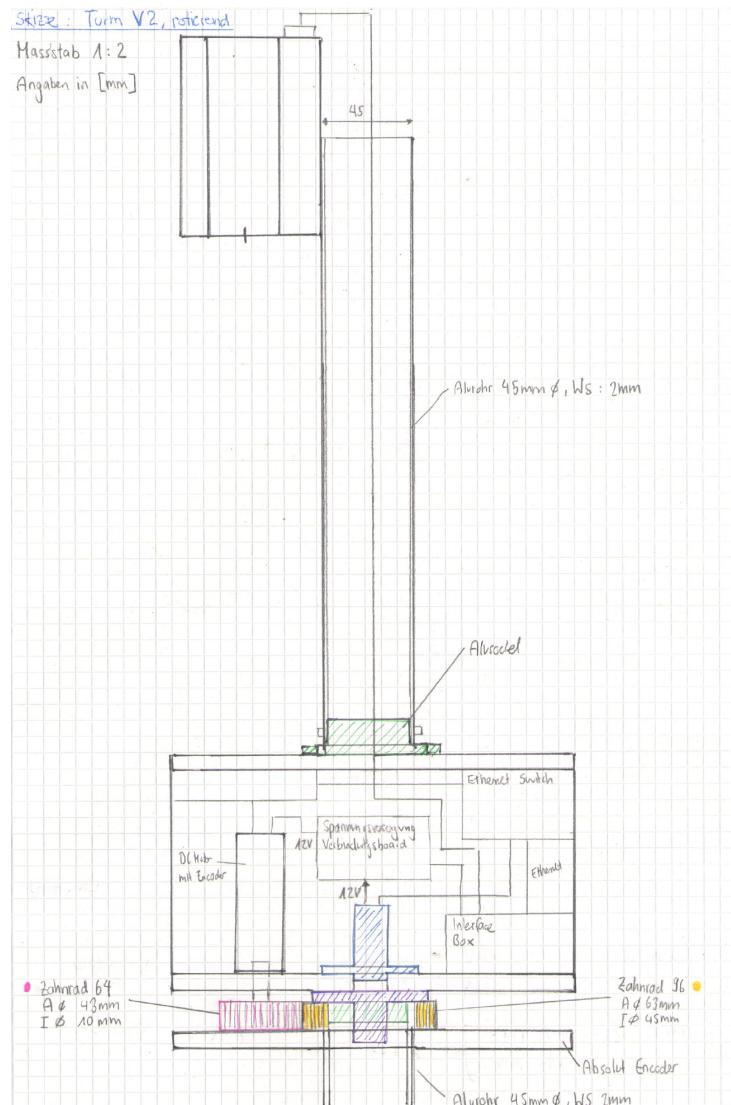


Abbildung 3.4: Skizze Variante 3

Diese Variante bietet jedoch den Nachteil das der Schleifring bedeutend mehr belastet wird. Da alle Komponenten über den Schleifring gespeist werden, müsste eine sorgfältige Berechnung der Strom- und Spannungswerte erfolgen, damit der Schleifring entsprechend ausgelegt wird.

## 3.5 Konzeptfazit

Da Konzeptvariante 1 bereits frühzeitig verworfen wurde, stehen nur noch die zwei Turm Konzepte gegenüber. Anfänglich wurde die Variante 3 bevorzugt. Während der Zwischenpräsentation vom 8. November 2017 und einer nachträglichen Überarbeitung konnten bedeutende Nachteile der Variante 3 aufgezeigt werden. Einerseits kann die Variante 3 nicht mit einer Bewegungskompensation mittels IMU erweitert werden, die für den Einsatz auf dem Packpot nötig ist. Andererseits ist ein bedeutend höheres Drehmoment des Motors nötig, da durch die zusätzliche Elektronik ein bedeutend höheres Gewicht entsteht. Um diese Problem zu verhindern, wurde die Konzipierung der Variante 3 abgebrochen und die Variante 2 ausgewählt. Da die mechanischen Komponenten von Variante 2 und 3 sich kaum unterscheiden, mussten keine groben Zeitverluste hingenommen werden.

## 3.6 Evaluation der Komponenten

Dieses Kapitel beschreibt die evaluierten Komponenten für die verschiedenen Teifunktionen. Die Teifunktionen wurden bereits im Kapitel 2 Informationsbeschaffung unterteilt. Für die Entfernungsmessung ist der Velodyne VLP-16 vorgegeben. Als Software wurde Ubuntu LTS 16.04 mit ROS Kinetic Kame gewählt. Die Gründe dazu wurden bereits im Unterkapitel 2.3 erläutert. Die weiteren Teifunktionen werden im nachfolgenden morphologischen Kasten gegenübergestellt und danach ausgewählt.

TEILFUNKTIONEN	Lösungsmöglichkeiten		
Entfernungsmessung	Velodyne VLP-16		
Datenverarbeitung	Raspberry Pi 3 + Ethernet Switch	Raspberry Pi 2 + Ethernet Switch	Up Board Squared -
Software	Ubuntu LTS 16.04 + ROS Kinetic Kame		
Antrieb	Gleichstrommotor mit Encoder Pololu 25Dx52L (inkrementell)	Schrittmotor (Mikrostepping) Trinamic QSH4218-51-10-049	Schrittmotor Absolutencoder -
Positionsbestimmung	Nullposition LED/Photodiode	Nullposition QRE113	Absolutencoder QRE113 Array
Drehübertrager	Senring SNE01-24GV	MOOG SRA-73799	LTN Precision ESE264
Speisung	Layout-Print	Laborprint	-

Tabelle 3.1: Morphologischer Kasten

### 3.6.1 Datenverarbeitung: Raspberry Pi 3

Im Unterkapitel 2.4 wurden bereits diverse Einplatinencomputer betrachtet und bewertet. Für die Konzeption wurde aus Performancegründen das Raspberry Pi 3 dem Raspberry 2 vorgezogen. Da das Raspberry Pi 3 nur einen Ethernetanschluss zur Verfügung stellt, wurde der Ethernet Switch DGS-105 von Netgear, welcher bereits im Unterkapitel 2.7.3 erwähnt wurde hinzugefügt. Dieser Ethernet Switch kann direkt über die verfügbare Speisung von 12 Volt angeschlossen werden. Somit werden alle Datenschnittstellen ermöglicht.

### 3.6.2 Antrieb: Pololu 25Dx56L

Schrittmotoren mit Absolutencoder wurden aus Kostengründen in der Evaluation ausgeschlossen. Reine Schrittmotoren sind für die Aufgabenstellung ebenfalls geeignet. Lediglich die Schrittverluste können einen integrierenden Fehler verursachen. Daher wurde für die Aufgabenstellung ein Gleichstrommotor gewählt. Der Pololu 25Dx52 ist ein kompakter 12 Volt Gleichstrommotor mit integriertem 48 Counts Per Revolution (CPR) Encoder. D.h. es werden pro Umdrehung 48 Zustände evaluiert, wenn die zwei vorhandenen Hallsensoren verwendet werden. Durch die interne Übersetzung von 46.85:1 ergeben sich 2248.86 CPR. Daraus ergibt sich somit die folgende Auflösung  $\frac{360^\circ}{2248.86} = 0.16^\circ$ .

In einem Messaufbau wurde der Motor angesteuert und ausgetestet. Die zwei Hallsensoren werden über 5 Volt gespeist und über zwei GPIOs auf die fallende und steigende Flanke getriggert. In einer Callback Funktion werden diese Flanken gezählt und über das Verhältnis der aktuelle Winkel eruiert. Die maximale Umdrehungszahl des Motors ist 110 RPM. Durch das lineare Verhalten eines Gleichstrommotors kann durch den Tastgrad eines PWM die Umdrehungsgeschwindigkeit mit einem Motorentreiber eingestellt werden.

### 3.6.3 Positionsbestimmung: Fairchild QR1113

Da die inkrementelle Winkeländerung mittels dem Pololu 24Dx56 eruiert werden kann, benötigt es für die Positionsbestimmung lediglich einen definierten Nullpunkt. Der Absolutencoder mit einem QRE 1113 Array fällt für diese Funktion aus. Um eine Auflösung wie im vorherigen Kapitel zu erreichen benötigte es 11 dieser Sensoren. ( $2^{11}$  Zustände

= 2048). Einerseits werden dazu viele GPIOs benötigt und anderseits ergibt sich durch den Mindestabstand von 4 mm zwischen den Sensoren eine grosse Dimension des Drehencoders. Die Dimension entspricht einem Hohlzylinder mit Außendurchmesser 10 cm. In einem Messversuch wurde mit einem Kathodenoszilloskop (KO) das Verhalten der LED/Photodiode-Variante und eines einzelnen QRE 1113 ausgetestet. Dabei konnte festgestellt werden, dass mit dem QRE1113 eine bessere Trennschärfe erreicht wird. Dieser Sensor kann bei einem Abstand von 2 mm von der Oberfläche eine praktisch binäre schwarz/weiss Detektion erzielen. Daher wurde die Variante der Nullposition mit dem Fairchild QR1113 evaluiert.

### 3.6.4 Drehübertager: Senring SNE01-24GV

Wie bereits in Unterkapitel 2.7.2 erläutert ist die Auswahl von Ethernet-Schleifringen bisher noch klein. Es wurde der Senring SNE01-24GV für die Aufgabenstellung evaluiert. Dieser Schleifring ist einerseits mit 120 Fr. der kostengünstigste Schleifring und bietet die erforderlichen Übertagungsmöglichkeiten von Ethernet. Der Schleifring MOOG SRA-73979 und LTN Precision ESE264 sind mit 435 Fr. bzw. 610 Fr. bedeutend teurer. Zusätzlich besitzt der evaluierte Schleifring acht weitere Kabeldurchführungen, von denen zwei Kabel mit 5 Ampere und sechs Kabel mit 2 Ampere belastet werden können.[Sen15]

## 3.7 Zwischenfazit

Die evaluierten Komponenten wurden in einigen Messversuchen ausgetestet und für die Aufgabenstellung als geeignet beurteilt. Durch die Sensorkombination kann mit entsprechender Logik die aktuelle Position ermittelt werden. Dies ist das wichtigste Kriterium für eine erfolgreiche Softwareimplementation. Die Dimension des Prototypen sind grösstenteils durch die evaluierten Komponenten gegeben. Einzige Verbesserung des Konzepts ist die Lage des Velodyne VLP-16. Wird dieser im Drehpunkt befestigt, kann die Differenz zur Drehachse vernachlässigt und somit ein Messfehler kompensiert werden. Bei der Realisierung muss eine möglichst optimale mechanische Konstruktion erstellt werden. Mechanische Ungenauigkeiten und Toleranzen führen dazu, dass die angestrebten Winkeländerungen mit dem Gleichstrommotor nicht eingehalten werden.

# Kapitel 4

## Realisierung

Dieses Kapitel beschreibt die Realisierung des Prototyps. Der Prototyp ist eine überarbeitete Version der Konzeptvariante 2, siehe Unterkapitel 3.3. Es mussten Änderungen bei verschiedenen Komponenten durchgeführt werden, diese werden im Unterkapitel 4.1.1 erläutert. In Abbildung 4.1 ist das realisierte Konzept ersichtlich.

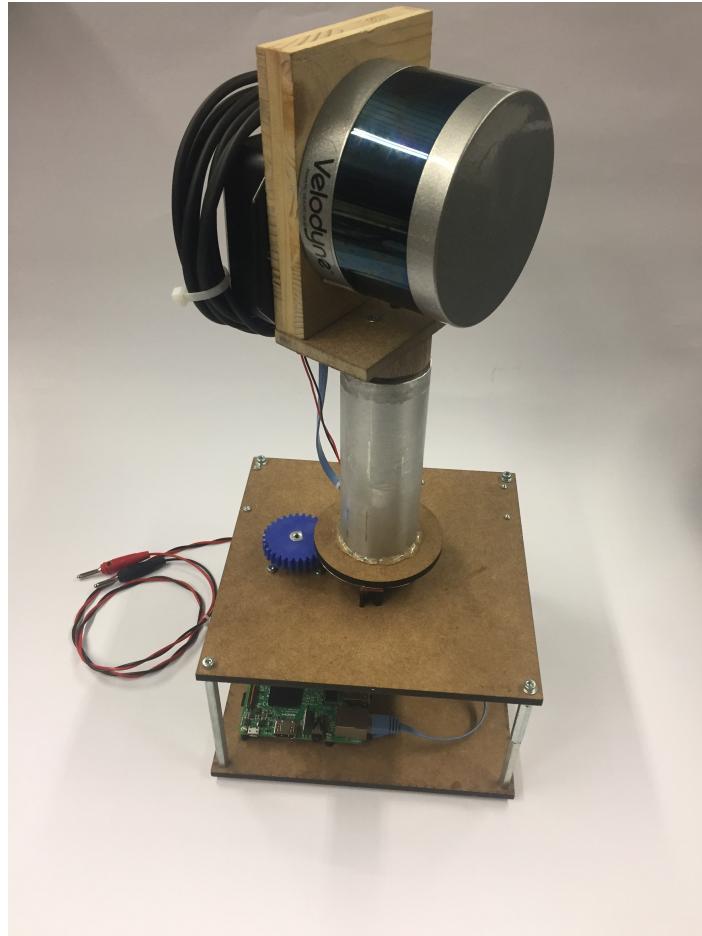


Abbildung 4.1: realisiertes Konzept

## 4.1 Hardware

Da beide Projekte, welche im Unterkapitel 2.2 eine endlos drehende Konstruktion nutzen, sowie die Aufgabenstellung im Anhang A eine drehende Konstruktion vorgibt, wurde eine endlos drehende mechanische Konstruktion angefertigt.

### 4.1.1 mechanische Komponenten & Gehäuse

Um eine möglichst einfache und schnelle Lösung zu realisieren, wurden die mechanischen Komponenten nach Verfügbarkeit ausgewählt. Die Masse des Alurohrs und der Zahnräder waren größtenteils durch die verfügbare Größe des Kugellagers gegeben. Das Kugellager wurde so gewählt, dass ein Ethernet RJ45 Stecker hindurchgeführt werden kann. Daraus ergibt sich einen Innendurchmesser von 22 mm und der entsprechende Aussendurchmesser von 44 mm. Das Kugellager wurde direkt in das Alurohr gepresst und zusätzlich seitlich mit Schrauben verkeilt. Das Alurohr wurde mit einem Aussendurchmesser von 50 mm und einer Wandstärke von 3mm gewählt. Einerseits ist so der Innendurchmesser des Alurohrs passend zum Aussendurchmesser des Kugellagers und anderseits können bei dieser Wandstärke Gewinde geschnitten werden, damit Schrauben als Keil verschraubt werden können.

Der innere Ring des Kugellagers wurde auf den Kugellageradapter gesteckt. Dieser ist wiederum wie in Abbildung 4.2 an der Deckplatte befestigt. Somit lässt sich das Alurohr mit wenig Reibung drehen.

Die zwei Zahnräder wurden mit dem CAD-Tool OnShape gelayoutet und im FabLab mit einem 3D-Drucker erzeugt. Das grosse Zahnrad mit 48 Zähnen besitzt den Innendurchmesser 50 mm, welcher dem Alurohr entspricht. Dieses wurde auf das Alurohr gepresst. Das zweite Zahnrad ist im Verhältnis 1.5:1 kleiner und besitzt 32 Zähne. Die Übersetzung wurde so gewählt, dass die Komponenten mit genügend Abstand nebeneinander montiert werden können. Dieses Verhältnis muss bei der Softwareimplementierung berücksichtigt werden.

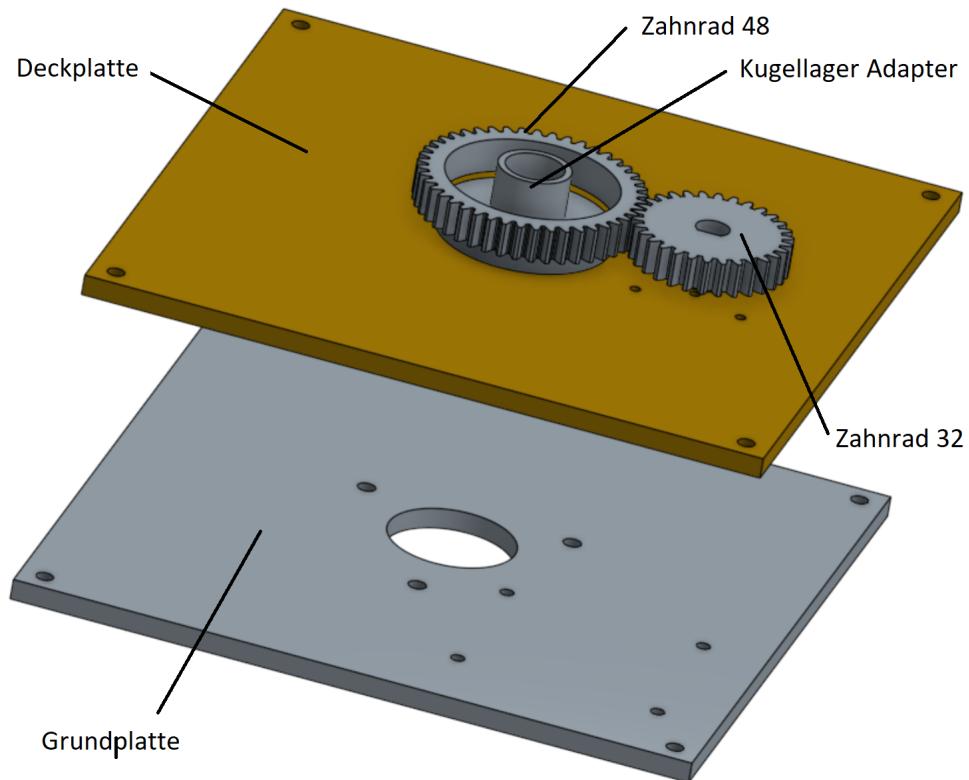


Abbildung 4.2: gelayoutete Komponenten in OnShape

Die Grösse der Platten wurde so gewählt, dass alle Komponenten dazwischen verbaut werden können. Daher wurden zwei MDF-Platten mit den Massen 200 mm x 200 mm x 6 mm (HxBxT) im FabLab erstellt. Die Grund- und Deckplatten wurden in OnShape gelayoutet und besitzen bereits vorgefertigte Löcher für die Montage der elektrischen Komponenten.

#### 4.1.2 elektrische Komponenten

Der Velodyne VLP-16 wurde auf einen rechtwinklige Konstruktion montiert, so dass der Lasermittelpunkt möglichst auf der Drehachse liegt. Die Interface Box wurde direkt gegenüber des Lasers montiert. Über den Schleifring führen eine 12 Volt Speisung und das Ethernetkabel direkt zu den erforderlichen Anschlüsse. Mit dieser Konfiguration lässt sich der Velodyne über die Zahnräder endlos drehen.

Die Nullposition wird mittels dem QRE 1113 und einem Hohlzylinder detektiert. Der Hohlzylinder ist dabei am rotierenden Alurohr befestigt. Der QRE 1113 ist mit dem Abstand von 2 mm unterhalb des Hohlzylinders fest montiert. Indem auf dem Hohlzylinder ein schwarzer Streifen auf eine weiße Oberfläche gedruckt ist, kann der Nullpunkt detektiert werden.

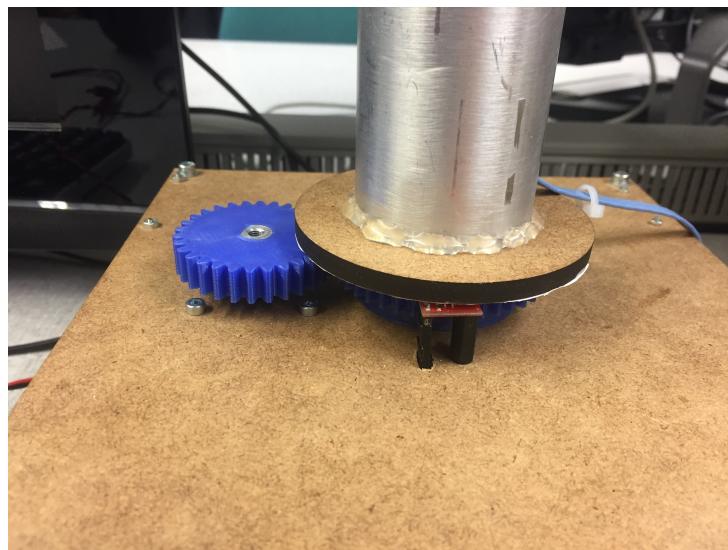


Abbildung 4.3: Nullpunktterkennung mittels QRE 1113

In der Konzeption wurde ein Gleichstrommotor der Marke Pololu für den Antrieb ausgewählt (siehe Unterkapitel 2.5.2). Während der Realisierung wurden sehr spät (Kalenderwoche 11) bedeutsame Fehlverhalten des Motors festgestellt. Während den Leerlaufmessungen (Kalenderwoche 7) konnten keine Abweichungen der Encodersignale festgestellt werden. Nach dem Verbauen des Gleichstrommotors entsprachen die Encoder-Signale nicht den zu erwartenden Ergebnissen. Die Messwerte weichen mit bis zu 200 CPR, welches umgerechnet ca.  $36^\circ$  ist vom Sollwert von 2248 ab. Dies verursacht unbrauchbare Resultate beim zusammenfügen der Punktwolken. Der Motor besitzt zusätzlich eine nicht nachvollziehbares Fehlverhalten. In einer Messung des Motorentreibers wurde ein fehlerfreies PWM ausgemessen und auch die nötigen Strom- (1 Ampere) und Spannungswerte (12 Volt) verifiziert. Dennoch blockiert der Motor kurzzeitig willkürlich bei verschiedenen Drehpositionen. Dieses Fehlverhalten könnte auf einen internen Getriebefehler zurück zu führen sein. Ursache für dieses Verhalten kann die Befestigungsschraube sein, welche bei der Montage hinein gedreht wurde. Aufgrund längerer Lieferfrist wurde darauf verzichtet

den identischen Gleichstrommotor zu bestellen.

Aus den erwähnten Gründen wurde schnellst möglichst eine Alternative eruiert. Bei der Alternative handelt es sich um einen Schrittmotor der Marke Trinamic. Dieser Motor wurde aus Gründen der Verfügbarkeit und aus bereits bestehenden Kenntnissen über dessen Funktionsweise ausgewählt. Mit diesem Schrittmotor ist es möglich eine gleichmäßige Umdrehungsgeschwindigkeit zu erreichen. Der Schrittmotor benötigt 200 Schritte für eine Umdrehung, sofern kein Schrittteilung (Mikrostepping) Aus den getätigten Schritten kann der aktuelle Winkelposition ermittelt werden. Detaillierte Erläuterungen folgen im nächsten Kapitel.

## 4.2 Software

Dieser Abschnitt erläutert die wichtigsten Elemente der Softwareimplementierung. Die Software wurde in einem eigenen Package namens Laser\_3D erarbeitet und besitzt mehrere C++ und Python Files. Im digitalen Anhang ist das gesamte Package einsehbar.

### 4.2.1 Ansteuerung mittels GPIO

Wie bereits erwähnt, wurde für das Drehen des Turm ein Schrittmotor der Marke Trinamic verwendet. Dieser wird über einen Motorentreiber A4988 angesteuert. Es werden mittels 1/16 Schritte insgesamt 3200 Schritte für eine Umdrehung benötigt. Daraus ergibt sich mit der Übersetzung eine maximale Auflösung von  $\frac{360^\circ}{1.5 \cdot 3200} = 0.075^\circ$ . Die

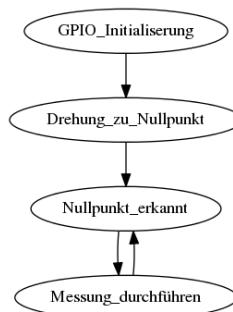


Abbildung 4.4: Ablaufdiagramm der GPIO-Ansteuerung

Die Ansteuerung wird mit einem Python Script ausgeführt. Dies besitzt den Vorteil, dass die GPIOs mit dem Python Package RPi.GPIO 0.6.3 einfach angesteuert werden können. Die Ansteuerung verläuft mit einer endlosen sequenzielle Ablauffolge (State-Machine). Dabei werden die benötigten GPIOs initialisiert. Danach wird der Schrittmotor schrittweise angesteuert, bis dieser sich beim Nullpunkt befindet. Über eine Konsoleneingabe wird der Messvorgang gestartet und der Schrittmotor wird mit eingestellten Umdrehungsgeschwindigkeit gedreht.

#### 4.2.2 Softwareaufbau

Wie bereits im Kapitel 2.3 erläutert, wird durch das Velodyne Package ein grosser Teil der Aufgabenstellung bereits abstrahiert. Die einzelnen Aufgabenblöcke wurden nochmals unterteilt und sind in Abbildung 4.5 dargestellt. Einerseits ist es nötig den Datenstream in einem weiteren Node zu empfangen und anderseits muss die Position ermittelt werden. Diese beiden Daten werden im Node */velodyne\_combined* zusammengefügt. Sobald die Daten kombiniert werden können, muss in einem weiteren Schritt die kombinierten Daten in einem Container gespeichert werden und bei Bedarf versendet.

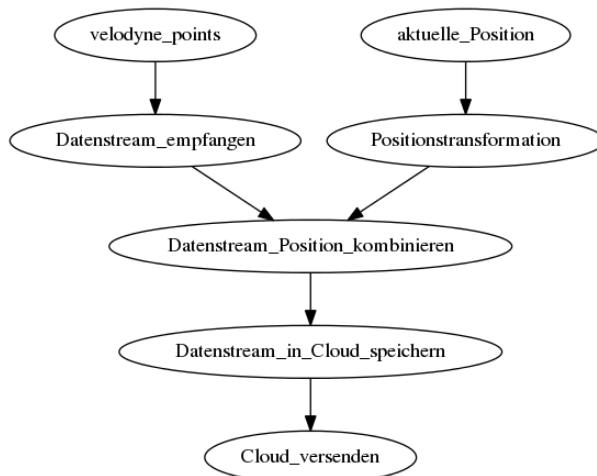


Abbildung 4.5: Software Ablaufstruktur

Die Software wurde aufbauend konzipiert. Dabei wurde in einer ersten Phase das Empfangen und Weiterleiten mittels ROS *Publisher* und *Subscriber* programmiert. In einem

zweiten Schritt wurde die aktuelle Position mit den Daten kombiniert. In der letzten Phase wurde die Software mit dem Zusammenfügen der Datenpunkte zu einer Punktwolke erweitert. Wesentliche Codeausschnitte werden in den entsprechenden Unterkapiteln erläutert.

In Abbildung 4.6 ist mit **roter** Umrandung die Softwareerweiterung ersichtlich.

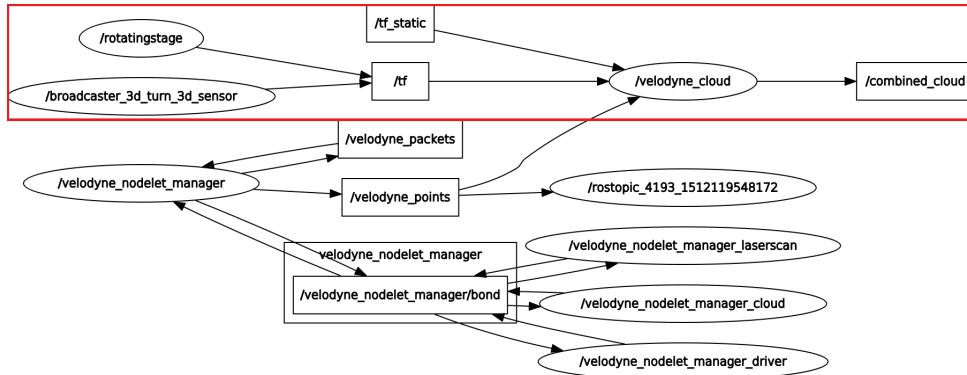


Abbildung 4.6: Softwareerweiterung mit Laser\_3D

### 4.2.3 Phase 1: Empfangen und Weitersenden

In dieser Phase wurden zu Beginn eine Programm implementiert, welche die Sensordaten (*Messages*) empfängt und direkt wieder weiterleitet. Diese Funktion bietet das Grundgerüst für die weiteren Phasen. ROS bietet für diese Funktion zwei generische Klassen an. Nachfolgend sind dazu die wesentlichsten Codeausschnitte mit Erläuterung wiedergegeben. [ros16b]

```

//Subscriber
ros::Subscriber points_sub = node.subscribe("/velodyne_points",100,points_callback);
//Publisher
ros::Publisher points_pub = node.advertise<sensor_msgs::PointCloud2>("/direct_points",100);

void points_callback(const sensor_msgs::PointCloud2ConstPtr& msg)
{
    cloud_message = *msg;
}

```

Der Publisher versendet Messages über sogenannte Topics, wie beispielsweise die *velodyne\_points* (vom Typ: *velodyne\_msgs/PointCloud2*). Subscriber können von Topics die

Messages mittels einer Callback-Funktion an Variablen übergeben.

#### 4.2.4 Phase 2: Positionstransformation

Der empfangene Sensorstream kann mit Koordinatentransformation in die aktuelle Position des Drehturms transformiert werden. Es wird an dieser Stelle für die nachfolgenden Betrachtungen kurz ein Exkurs zu Koordinatentransformation gemacht [Bit17]. Es gelten für dreidimensionale Drehbewegungen folgende Matrizen:

$$R_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \phi & \sin \phi \\ 0 & -\sin \phi & \cos \phi \end{bmatrix}, R_y(\theta) = \begin{bmatrix} \cos \theta & 0 & -\sin \theta \\ 0 & 1 & 0 \\ \sin \theta & 0 & \cos \theta \end{bmatrix}, R_z(\psi) = \begin{bmatrix} \cos \psi & \sin \psi & 0 \\ -\sin \psi & \cos \psi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Mit dem Rollwinkel  $R_x(\phi)$  (*engl. roll*), Nickwinkel  $R_y(\theta)$  (*engl. pitch*) und Gierwinkel  $R_z(\theta)$  (*engl. yaw*) können in die 3 Freiheitsgrade beliebige statische oder dynamische Transformationen vorgenommen werden. Alternative können die Bewegungen mittels Erweiterung der komplexen Zahlen (Quaternionen) angegeben werden. In Abbildung 4.7 sind die Drehbewegungen mit den Achsen visualisiert.

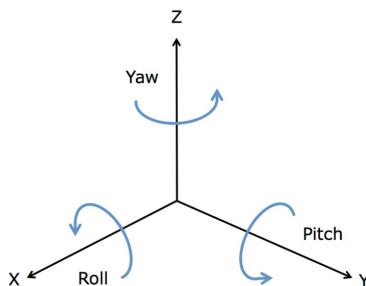


Abbildung 4.7: Roll-Pitch-Yaw mit X-Y-Z-Achsen

ROS bietet die Möglichkeit mehrere Koordinatensysteme miteinander zu verlinken und über die Klassen *TF Broadcaster* und *TF Listener* die aktuellen Positionen zu versenden und empfangen. In einem separaten Programmsegment *rotatingstage* (siehe Abbildung 4.6) wurde ein TF Broadcaster implementiert, welcher die aktuelle Ausrichtung des Drehturm versendet. Nachfolgender Codeausschnitte erläutern wesentliche Elemente. Es werden

die Objekte *TransformBroadcaster*, *Transform* und *Quaternion* verwendet, damit eine Bewegung erstellt wird.

```
tf::TransformBroadcaster br;

// set default params for transform
tf::Transform transform;
transform.setOrigin(tf::Vector3(0.0, 0.0, 0.5));
tf::Quaternion q;
q.setRPY(0, 0, 0);
transform.setRotation(q);
```

Wird die aktuelle Position des Motors als Winkel phi übergeben und fortlaufend als Drehung in der Z-Achse (yaw) versendet, lassen sich die Koordinatenachsen verschieben. Da der Velodyne VLP-16 um -90° (270°) geneigt ist, wurde die Transformation entsprechend gedreht (roll). Die versendete Transformation besitzt neben aktueller Position, einen Zeitstempel und die betreffenden Koordinatensysteme.

```
// broadcast transform
q.setRPY(3*1.5707, 0, phi);
transform.setRotation(q);
br.sendTransform(
tf::StampedTransform(transform, ros::Time::now(), "3d_base", "3d_turn"));
```

Der Node */velodyne\_combined* wurde mit einem TF Listener erweitert. Dieser ruft in einer lookup-Funktion die aktuelle Transformation ab und speichert diese in eine Variable.

```
// tf variables
geometry_msgs::TransformStamped velodyne_to_3dsensor;
tf::TransformListener listener;
try{
    tf::StampedTransform transform;
    listener.lookupTransform("3d_base", "3d_turn", ros::Time(0), transform);
    tf::transformStampedTFToMsg(transform, velodyne_to_3dsensor);
    ROS_INFO("received TF");
    ros::spinOnce();
}
catch (tf::TransformException ex){
    ROS_ERROR("%s",ex.what());
    ros::Duration(1.0).sleep();
```

In Abbildung 4.8 sind die zwei Koordinatensysteme mit Rviz visualisiert. Die drehende Achse 3d\_turn, die mit der Achse 3d\_base verlinkt ist.

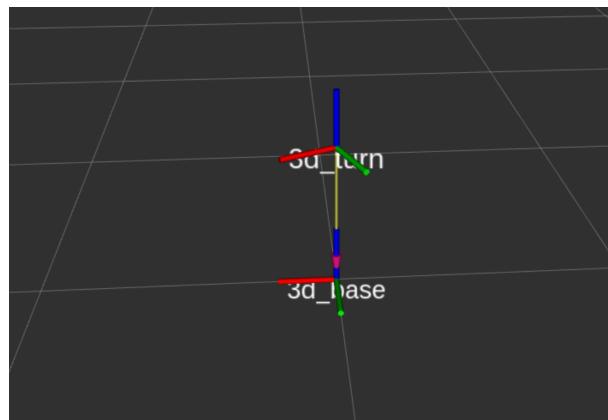


Abbildung 4.8: dynamische Koordinatensysteme

#### 4.2.5 Phase 3: Punktwolke zusammenfügen

Während dieser Phase wurde eine Strukturverbesserung (Refactoring) gemacht und eine Klasse SubListener eingeführt. Ohne die Klasse verfügen mehrere Variablen nicht über den nötigen Sichtbarkeitsbereich (Scope). In folgender Abbildung ist das Klassendiagramm mit entsprechenden Funktionen ersichtlich.

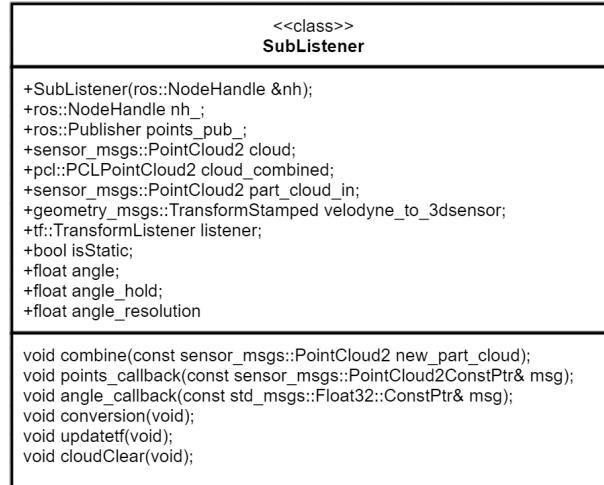


Abbildung 4.9: UML Klassendiagramm SubListener

Auf eine detaillierte Erläuterung der Funktionen wird in diesem Abschnitt verzichtet. Der nachfolgende Codeausschnitt erläutert konzeptionell der Ablauf, wie das Zusammenfügen

der Punktwolke funktioniert. Der Codeausschnitt wird in der *points\_callback* Funktion ausgeführt.

```
updatetf();
// need to be a range because of the tolerance of float datatype
if (angle >= 0.00f && angle < 0.01f)
{
    cloudClear();
}
// change angle_resolution, to change pointcloud resolution
if (angle >= angle_hold + angle_resolution)
{
    ROS_INFO("received pcl_msg");
    sensor_msgs::PointCloud2 part_cloud_out;
    tf2::doTransform (*msg, part_cloud_out, velodyne_to_3dsensor);

    //same frame to see rotation between static and dynamic
    part_cloud_out.header.frame_id = "velodyne";
    combine(part_cloud_out);
    conversion();
    points_pub_.publish(cloud);
    angle_hold = angle;
}
```

Mit dem *updatetf()* werden die aktuellen Positionskoordinaten, wie in Phase zwei erläutert, abgerufen und zwischengespeichert. Mit einem zusätzlichen Subscriber wird der aktuellen Drehwinkel *angle* als Fließzahl abgerufen. Diese ermöglichen einerseits den Nullpunkt zu erkennen, damit die Punktwolke nach einer Umdrehung gelöscht werden kann. Es bietet auch die Möglichkeit die Auflösung mittels einem einzigen Parameter *angle\_resolution* zu ändern. Es wurde wegen der Ungenauigkeit der Nachkommastellen des Datentyps ein tolerierbarer Bereich definiert.

Mit der Funktion *tf2::dotTransform()* lassen sich einzelne Messages mit der aktuellen Transformation zusammensetzen. Dabei muss die zusammengesetzte Messages in eine neue Variable gesichert werden. (hier: *sensor\_msgs::PointCloud2 part\_cloud\_out*).

Die zusammengesetzte Variable wird der Funktion *combine(...)* als Parameter übergeben, damit diese der Punktwolke hinzugefügt wird. Um die vollständige Punktwolke zu versenden ist noch eine Datenkonvertierung nötig, welche die Funktion *conversion()* tätigt.

#### 4.2.6 Anwendung der Software

In Abbildung 4.10 ist ein Ausschnitt einer zusammengefügten Punktwolke mit Rviz visualisiert. Dabei wurden eine Umdrehungsgeschwindigkeit des Motors von  $5^{\circ}/\text{s}$  gewählt.

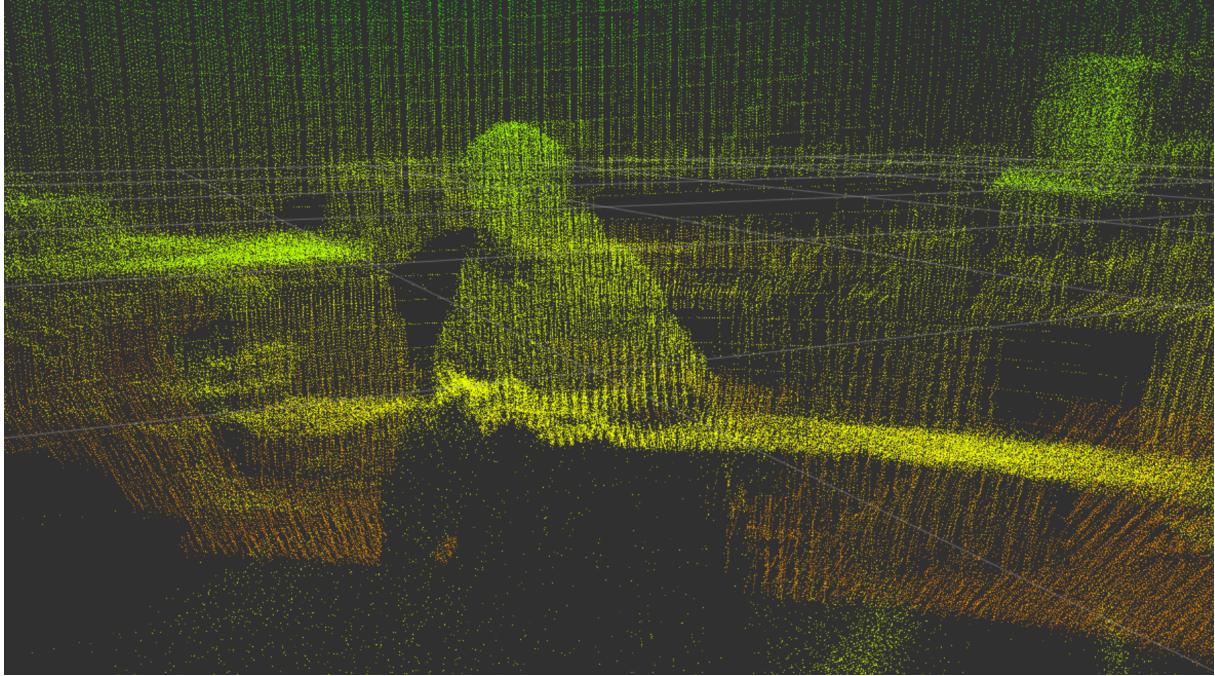


Abbildung 4.10: Punktwolke einer Person bei Distanz 1m

Durch das langsame Verschieben der 16 Laserstrahlen entsteht ein detaillierte Punktwolke, welche die Konturen von Körper sehr deutlich modelliert.

In Abbildung 4.11 ist das entsprechende Gesamtbild der vorherigen Punktwolke dargestellt. Es handelt sich um einen Raum der Grösse  $3 \text{ m} \times 15 \text{ m} \times 10\text{m}$  (HxBxT).

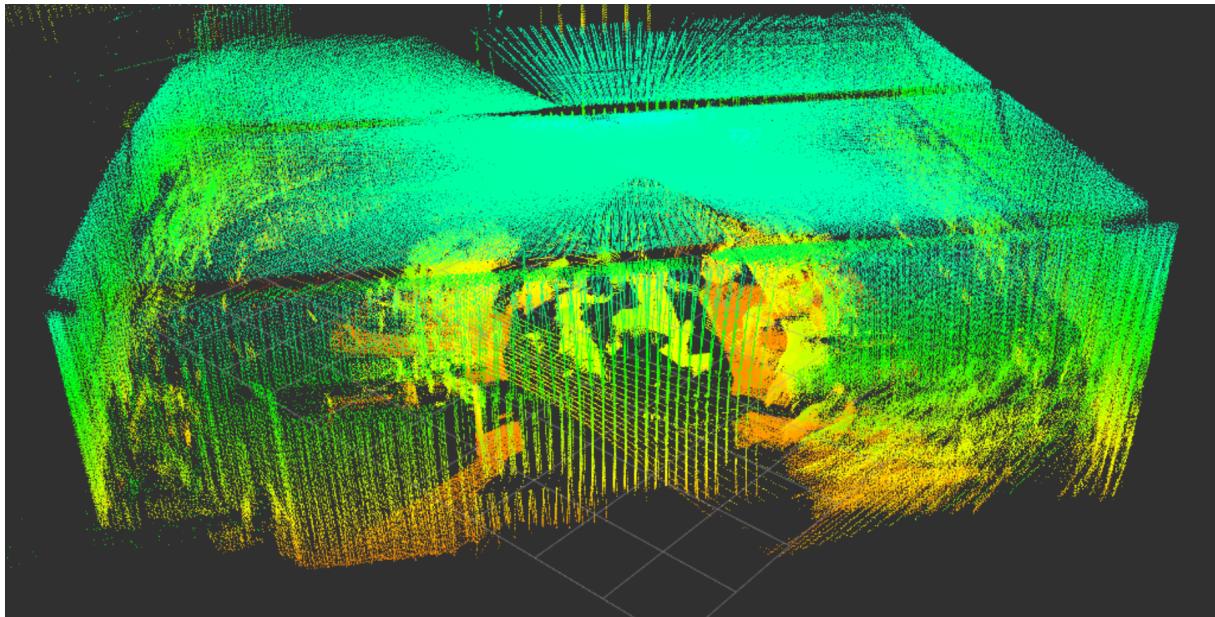


Abbildung 4.11: zusammengefügte Punktwolken

Die Visualisierung zeigt die Grenzen der Messung sehr deutlich auf. Da in dieser Messung nur aus einen festen Standort gemessen wird, entstehen durch Körper Schattierungen, welche keine Messpunkte zulassen. Dennoch entstehen von geeigneten Standorten sehr detaillierte räumliche Aufnahmen. Dabei kann durch die Konfiguration des Sensors während einer halben Umdrehung bereits eine 360° Erkennung stattfinden.

### 4.3 Zwischenfazit

ROS bietet viele Packages und Klassen, welche nützlich für die Softwareimplementation von Koordinatentransformation und Punktwolken sind. Die Freiheit von Open-Source bietet jedoch auch das Problem, dass eine Vielzahl an verschiedenen Klassen mit unterschiedlichen Datentypen vorhanden sind. Die Klassen bieten meist keine generischen Funktionen, daher müssen entsprechende rechenintensive Konvertierungen durchgeführt werden. Anfänglich waren kaum Kenntnisse über die Koordinatentransformation vorhanden, daher verzögerte sich diese Softwarephase 2 erheblich. Es mussten einerseits die theoretischen Grundlagen verstanden werden und entsprechende Packages dazu evaluiert werden.

# Kapitel 5

## Tests

Aus zeitlichen Gründen musste die Testphase gekürzt werden. Es konnten somit keine Testversuche auf dem Packpot durchgeführt werden. Nachfolgend wurden Funktionstests durchgeführt, damit eruiert werden kann, welche Funktionen erreicht wurden.

### 5.1 Testspezifikationen

Nachfolgende Tabelle, 5.1 gibt einen Überblick, welche Teifunktion getestet werden. Dabei wird kurz ein Beschrieb dazu geliefert und welche Parameter bzw. Spezifikationen ausgewertet werden.

Funktionstests	Nr.	Beschreibung	Testablauf	Status
Speisung	1	Aufnahme elektrischen Eckdaten und Kontrolle von Speisespannungen bei vollständigem Betrieb.	Stückweise Zuschalten der Komponenten und Kontrolle der Kabelverbindungen auf Wackel sowie Aufnahme Leistungsverbrauch.	bestanden
Mechanik	2	Kontrolle der mechanischen Komponenten, Toleranzen ermitteln und Drehfunktion bewerten.	Analyse der Verzahnung, gepresste Übergänge kontrollieren, Verhalten des Kugellagers im Gesamtsystem, Ungenauigkeit des Drehturm.	bestanden
Datenkomunikation	3	Funktionsfähigkeit der Datenübermittlung vom Veldoyne VLP-16 zu RPI bzw. Laptop über Schleifring prüfen.	Kontrolle der Drehübertragung bei festen und drehenden Schleifring.	bestanden
Datenverarbeitung	5	Visualisierung und Aufnahme der Daten auf dem RPI bzw. Laptop.	Packages wird vollständig auf dem Raspberry Pi ausgeführt und die Punktwolke wird visualisiert.	nicht bestanden
Positionbestimmung	6	Positionsbestimmung wird durch inkrementelle Winkeländerung erreicht.	Mittels vordefinierten Schritten wird eruiert, ob Schrittverluste entstehen.	bestanden
Nullpunktterkennung	7	Nullpunktterkennung während einer Umdrehung.	Es wird die Nullpunktterkennung augetestet. Einerseits als Teifunktion und im System integriert.	nicht bestanden
Software	8	Qualität der zusammengefügten Punktwolke und Funktion nach Aufgabenstellung.	Es wird die Messauflösung während einer Umdrehung bewerten, dazu die Datengröße ermittelt.	nicht bestanden

Tabelle 5.1: Funktionstests im Überblick

## 5.2 Testergebnisse

Die definierten Testergebnisse werden

### 5.2.1 Speisung

Mittels einem ROHDE & SCHWARZ NGSM32 wurde eine exakte 12 Volt Speisung zugeführt. Alle Komponenten konnten problemlos zugeschaltet werden. Wackelkontakte wurden keine festgestellt. Die Leistungsaufnahme der gesamten Modul variiert zwischen 9-15 Watt. Dies ist abhängig von der auszuführenden Funktion. Da keine Mängel festgestellt wurden, gilt dieser Funktionstest als erfüllt.

### 5.2.2 Mechanik

Die Mechanik wurde auf Mängel geprüft. Die zwei Zahnräder verlaufen sehr exakt ineinander und verkanten sich nicht. die gepressten Teile sitzen fest und verursachen keine Toleranzen. Einzig der Kugellageradapter ist nicht optimal gepresst. Dies verursacht, dass der drehende Hohlzylinder den Abstand zum QRE 1113 um einen Millimeter variiert. Da die Mechanik ansonsten einwandfrei funktioniert gilt dieser Funktionstest als bestanden.

### 5.2.3 Datenkommunikation

Die Datenkommunikation ist in allen Fällen gewährleistet. Es kann zwischen dem Velodyne VLP-16, dem Raspberry Pi 3 und dem Laptop problemlos kommuniziert werden. Es wurden mehrere Datensätze auf vollständige Messages geprüft. Dabei konnten keine fehlenden Messages festgestellt werden. Die Datensätze wurden mit rotierendem Turm, sowie mit den Drehgeschwindigkeiten von  $5^\circ/\text{s}$ ,  $20^\circ/\text{s}$  und  $120^\circ/\text{s}$  fehlerlos übermittelt.

### 5.2.4 Datenverarbeitung

Die Datenverarbeitung des Datenstreams wurde auf dem Raspberry Pi, wie auch auf dem Laptop geprüft. Beim Raspberry Pi 3 besteht das Problem, dass die Visualisierung mit Rviz mangelhaft funktioniert. Aufgrund der Datenrate von 8 Mbit/s steigt die Grösse der Punktwolke schnell an. Das Raspberry Pi stösst für die Visualisierung dieser Punktwolke an die Grenzen. Auf dem Laptop konnten die Punktwolke in Echtzeit visualisiert werden, da dieser bedeutend höhere Rechenleistungen bietet.

### 5.2.5 Positionsbestimmung

### 5.2.6 Nullpunktterkennung

Die Nullpunktterkennung besitzt Mängel. Der Nullpunkt wurde bei den Mündungsgeschwindigkeiten von  $5^\circ/\text{s}$ ,  $20^\circ$  und  $120^\circ$  nicht vollständig erkannt. Dabei wurden aus den Datensätzen

### 5.2.7 Software

## 5.3 Zwischenfazit

Die Test geben Überblick, welche Funktionsbereiche überarbeitet werden müssen bzw. bei welchen noch zufriedenstellende Ergebnisse geliefert werden. Dieser Testaufbau zeigt auf, dass für die Ansteuerung der Aktoren und Sensoren ein Raspberry Pi gute Dienste leistet. Für die Visualisierung benötigt es jedoch leistungsfähigere Rechenmaschinen.

# Kapitel 6

## Reflektion

Die Aufgabenstellung konnte während der zur Verfügung stehenden Zeit nicht erfüllt werden. In den nachfolgenden Unterkapiteln wird Stellung genommen, aus welchen Gründen die definierten Ziele nicht erreicht wurden. Im Fazit werden Ergebnisse und Ziele gegenübergestellt und offene Punkte erläutert. Danach werden zum Projektmanagement einzelne Punkte aufgelistet mit Bezug zur detaillierten Projektplanung in Anhang B. Der Ausblick bietet Aufschluss, welche weiteren Tätigkeiten zur Erfüllung der Aufgabenstellung geführt hätten. Zuletzt wird im Schlusswort eine persönliches Resümee gemacht.

### 6.1 Fazit

Die Aufgabenstellung bietet eine grosse Fülle an Disziplinen, welche in diesem Projekt abgedeckt werden. Dazu zählen: Recherche, Komponentenorientiert, Hardwarerealisierung, Softwareimplementation und Testversuche. Zu Beginn der Arbeit waren keine Vorkenntnisse über ROS, Koordinatentransformation und C++ Softwareprojektierung vorhanden. Diese 3 Punkte erwiesen sich bei dieser Arbeit als die grössten Stolpersteine. Es musste viel Zeit aufgewendet werden, um ein grobes Verständnis Die Aufgabenstellung zielte auf eine Entwicklung eines Prototypen hin. Daher wurden die Zeiträume für die verschiedenen Projektphasen erstellt.

### 6.2 Erläuterungen Projektmanagement

Die Projektplanung wurde in der KW aufgegliedert. Dabei gab es einige bedeutende Arbeitspakete, welche das Projektmanagement stark beeinflusst haben. Die Einschätzung des

zeitlichen Aufwands für die Erstellung der Software wurde falsch bewertet. Es musste bedeutend mehr Zeit für die Softwareentwicklung aufgewendet werden als eingeplant. Dies hat einerseits mit den ungenügenden Kenntnissen über die Programmiersprache C++/Python und dem Framework ROS zu tun. Andererseits nahmen mehrere ROS-spezifische Fehlermeldungen, welche nur durch die Unterstήzung von Herr Jensen behoben werden konnten, nahmen viel Zeit in Anspruch.

## 6.3 Ausblick

Die Aufgabenstellung konnte nicht voll umfänglich gelöst werden. Es bietet sich an eine Verbesserung des bestehenden Prototyp in Ausblick zu stellen. Einerseits wurden mit der Analyse der State-of-the-art Projekte Grundlagen geschaffen, um ein Konkurrenz fähiges Produkt zu erstellen und andererseits konnte mit der mechanischen Konstruktion eine solche realisiert werden. Um das Lasermodul auch bei mobilen Anwendungen nutzbar zu machen, muss die aktuelle Konstruktion mit einer eigenen IMU erweitert werden. Da das Raspberry Pi bei den Tests bereits an die Grenzen der Leistungsfähigkeit gelangt, ist es ratsam ein leistungsfähigeren Einplatinencomputer, wie das Up Board Squared zu verwenden. Mit einem solchen Einplatinencomputer können in verschiedenen Bereichen Verbesserungen erreicht werden. Die Software bietet ein gutes Grundgerüst für die Möglichkeit von der Erstellung von PointClouds. Durch die mangelnde sensorischen Fähigkeiten müssen kann diese jedoch nicht richtig eingesetzt werden. Eine Überarbeitung der aktuellen Sensorik für die endlos drehende Konstruktion ist naheliegend.

## 6.4 Schlusswort

Aus persönlicher Sicht bin ich mit der geleisteten Arbeit zufrieden. Ich kann aus dieser Industriearbeit sehr viele positive Erkenntnisse herausziehen und werde diese in Hinsicht auf die nächste Arbeit einfließen lassen. Das zeitliche Fortschreiten in der Realisierungsphase und einige Fehlüberlegungen führten zu einem nicht vollständigen Prototyp. Es konnte ein drehende Konstruktion realisiert werden, mit welcher eine Punktwolke erstellt werden kann.

## 6.5 Danksagung

An dieser Stelle möchte ich bei allen bedanken, die mich bei der Anfertigung dieser Arbeit unterstützt haben.

Zuallererst gebührt der Dank an Dr. Björn Jensen, der mich bei dieser Industriearbeit tatkräftige unterstützt hat, sowie mit wertvollen Hinweisen und schnellen Rückmeldungen zur Seite gestanden ist. Mein Dank geht auch an den Assistent Jonas Räber, der mir eine grosse Hilfe für die Einarbeitung mit ROS war. Ebenfalls bedanken möchte ich mich bei den Gegenleser Andreas Zimmermann, Marie-Theres Zimmermann und Angela Burch für die textuelle und inhaltliche Korrektur der wissenschaftlichen Dokumentation.

# **Anhang A**

## **Pflichtenheft**

Das Pflichtenheft wurde während der Einarbeitung des Projekts erstellt und Herr Jensen vorgelegt. Es bestimmt einerseits den Umfang der Arbeit mit entsprechenden Erläuterungen.

# Pflichtenheft

## **3D Laserscanner für mobilen Roboter Industriearbeit PAIND+E1**

im Auftrag des Industriepartners

**RUAG AG**

an der

Hochschule Luzern Technik & Architektur

im Studiengang Elektrotechnik

### **Schwerpunkt**

Signalverarbeitung & Kommunikation,  
Automation & Embedded Systems

**Dozent:** Dr. Björn Jensen

**Experte:** Prof. Dr. Markus Thalmann

**Industriepartner:** Dr. Thomas Nussbaumer

**Klassifikation:** Rücksprache

Version	Datum	Änderung	Verantwortlich
1.0	23.09.2017	-Erstellt	Daniel Zimmermann
1.1	04.10.2017	-Ergänzungen nach Zwischenbesprechung	Daniel Zimmermann
1.2	16.10.2017	-Ergänzungen nach Besprechung mit Industriepartner (Dr. Nussbaumer)	Daniel Zimmermann

## 1 Projektziel

Das Ziel des Projektes ist die Realisierung eines 3D-Laser Moduls als Prototyp. In erster Priorität soll damit 3D Mapping in Echtzeit betrieben werden können. Das Modul wird mit dem bestehenden 3D Laserscanner von Velodyne des Typs VLP-16 realisiert. Dabei soll eine möglichst dichte Punktwolke erstellt werden, welche visualisiert werden kann. Zweite Priorität ist die Hinderniserkennung in Frontrichtung. Dazu muss in Frontrichtung eine detaillierte Punktwolke ermittelt werden können. Das Modul soll einerseits auf dem iRobot Packbot nutzbar, sowie auch eigenständig einsetzbar sein.

## 2 Anforderungen

In nächsten Abschnitt werden die verschiedenen Anforderungen rund um die Projektarbeit PAINd beschrieben. Diese sind einerseits in sechs Kriterien unterteilt, je nach Anforderungsgeber und Anforderungsform. Diese Kriterien sind nachfolgend mit Kürzel erläutert:

- F: Festanforderung
- M: Mindestanforderung
- W: Wunschanforderung
- E: Eigene Anforderung
- D: Dozentenanforderung
- P: Modulanforderung

Neben den Kriterien wurden die Anforderungen zusätzlich in Themengebiete unterteilt, welche in den folgenden Unterkapiteln ersichtlich sind.

## 2.1 Allgemeine Anforderungen

Nr.	Krit.	Bezeichnung	Vorgaben / Erläuterung	Verantw.
1.1	F/P	Eigenes Produkt	Das Produkt wird vom Student selbst entwickelt.	DZ
1.2	F/P	Abgabe Dokumentation	Die Dokumentation wird am 22. Dezember 2017 im D311 bei Herr Andrist abgegeben.	DZ
1.3	F/D	Abschlusspräsentation	Zwischen dem 18.12.2017 – 26.1.2018 muss die Abschlusspräsentation stattfinden. Voraussichtlich KW 3/ 18	DZ
1.4	F/D	Zwischenpräsentation	Zwischenpräsentation findet am 8. November 15:30 bis 16:30 statt.	DZ
1.5	F/P	Abgabe Poster	Abgabe Posterfile am 30. Januar 2018 per Mail an Betreuer und Herr R. Andrist.	DZ
1.6	M/E	Konzeption	Bis zur Zwischenpräsentation soll mehrere Konzept erstellt werden mit Bewertung.	DZ

## 2.2 Ressourcen

Nr.	Krit.	Bezeichnung	Vorgaben / Erläuterung	Verantw.
2.1	F/D	Weiterverwendung bestehender Komponenten	Es soll bestehende Komponenten wie den Velodyne VLP-16, Raspberry Pi soweit möglich weiterverwenden werden.	DZ
2.2	F/D	Open Source Libraries	Es dürfen im Rahmen der Arbeit bereits bestehende Open-Source Libraries und Open Source Software genutzt werden.	DZ
2.3	F/D	Nutzbare Räume	Für Testversuche, Messungen etc. dürfen ET-Labor, FabLab sowie ET-Werkstatt verwendet werden.	DZ
2.4	F/P	Poster-Vorlage und Drucken	Das Poster muss nicht selbst gedruckt werden. Eine Vorlage wird zu entsprechenden Zeitpunkt vom Dozenten übergeben.	DZ
2.5	F/D	Material Bestellungen	Bestellungen werden über das ET-Labor getätigt mit dem Vermerk «PAIND».	DZ
2.6	F/D	Budget	Das Budget wurde nicht fix definiert. Lediglich eine Begrenzung von einigen hundert Franken wurde als Obergrenze festgelegt.	DZ

## 2.3 Projektanforderungen

Nr.	Krit.	Bezeichnung	Vorgaben / Erläuterung	Verantw.
3.1	F/P	Eigenes Projekt	Projekt wird eigenständig von Daniel Zimmermann ohne Hilfe, sofern nicht bewilligt, erarbeitet.	DZ
3.2	M/D	Treffen mit Dozenten	Es sollte mindestens alle zwei Wochen mit Björn Jensen ein Treffen abgehalten werden.	DZ
3.3	W/E	Pflichtenheft erstellen	Eine Pflichtenheft soll die Aufgabenstellung und den Rahmen eingrenzen.	DZ

## 2.4 Produktanforderungen

Nr.	Krit.	Bezeichnung	Vorgaben / Erläuterung	Verantw.
4.1	F/D	3D Mapping	Es soll eine Umgebungskarte mittels Point Cloud erstellt werden.	DZ
4.2	F/D	Priorität in Frontrichtung	Für die Frontrichtung soll eine detailliertere Erkennung stattfinden, um Hindernisse zu erkennen.	DZ
4.3	F/D	Messdatenauswertung mittels PC	Die gemessenen Distanzen sollen von einem PC aufgenommen und dem mobilen Roboter zur Verfügung gestellt werden.	DZ
4.4	F/D	Test auf Packbot	Das entwickelte Laser-Modul soll im Rahmen der Arbeit auf dem Packbot-Roboter getestet werden.	DZ
4.5	W/ D	Bewegungen kompensieren	Die Bewegungen des Roboter soll gemessen und die Messdaten entsprechend kompensiert werden.	DZ
4.6	F/D	Drehbare Plattform	Das Produkt sollte, wenn möglich drehbar sein, um möglichst einen grossen räumlichen Wahrnehmung zu gewähren.	DZ
4.7	W/ D	Position nahe der 3D Kamera	Die Position des Lasermoduls sollte so nahe wie möglich bei der 3D Kamera platziert werden.	DZ
4.8	F/D	Energieversorgung	Die Energieversorgung muss über die Packbot-Schnittstellen ermöglicht werden.	DZ
4.9	W/ D	Betriebssystem und verwendete Programmiersprache	Es wird das Betriebssystem Ubuntu LTS 16.04 verwenden mit Programmiersprachen C++/Python.	DZ
4.10	F/D	Schnittstellen	Das Modul muss über alle nötigen Schnittstellen verfügen, damit das Modul eigenständig oder auf dem Packbot funktionieren kann.	DZ

## 2.5 Dokumentation und Poster

Nr.	Krit.	Bezeichnung	Vorgaben / Erläuterung	Verantw.
5.1	F/P	Termingerechte Abgabe Schlussbericht	Alle Exemplare des Schlussberichtes müssen termingerecht am 22. Dezember 2017 um 16:00 im D311, an Herr. R. Andrist abgegeben werden.	DZ
5.2	F/P	Termingerechte Abgabe Poster	Abgabe des Poster-Files am Montag 30. Januar 2018 per Mail an Betreuer und Herr R. Andrist.	DZ
5.3	F/P	Schlussberichte mit CD	Es ist ein gebundener Schlussbericht (nicht Ordner) mit CD in 3-facher Ausführung zu erstellen.	DZ
5.4	F/P	Inhalt	Doku muss mindestens Selbstständigkeitserklärung, Titelblatt, Abstract, CD-Hülle (innen), auf der Rückseite des Berichts beinhalten.	DZ
5.5	M/P	Abstract	Das Abstract umfasst einen Englischen Text mit maximal 2000 Zeichen.	DZ
5.6	F/P	Titelblatt	Name des Studierenden, Titel der Arbeit, Abgabedatum, Dozent, Experte, Abteilung, Klassifikation.	DZ
5.7	F/P	Poster	Ein Poster ist gemäss den offiziellen Layout-Vorgaben zu erstellen.	DZ
5.8	W/E	Dokumentation mittels Latex	Der Bericht soll mittels Latex umgesetzt werden.	DZ

## 3 Projektphasen

Nachfolgend werden die einzelnen Projektphasen erläutert. Sie geben Auskunft, wie lange man mittels den angegebenen Arbeitsmitteln an der Phase tätig sein soll und welche Ergebnisse daraus entspringen. Dazu wird eine entsprechende Aufwandseinschätzung mitgeliefert, welchen den zeitlichen Umfang der Phase in Stunden wiedergibt. Die effektive Aufwandsabrechnung wird separat im detaillierten Projektplan dargelegt.

### 3.1 Initialisierung & Projektplanung

Dieses Abschnitt umfasst die administrativen Aufgaben, welche für die Projektplanung und Projektdurchführung nötig sind. Sie sollen möglichst als Vorbereitung dienen, damit

mit den Ergebnissen die Projektdurchführung erleichtert wird.

Aufwand	12 h
Personen	Daniel Zimmermann
Arbeitsmittel	Aufgabenstellung, Besprechungen, Vorlagen, Literatur
Ergebnisse	Pflichtenheft, Backup-Möglichkeit, Detaillierter Projektplan, Grobplanung, Meilensteine Anforderungsliste, Risikoanalyse

### 3.2 Informationsbeschaffung

Diese Projektphase beinhaltet die Recherche nach geeigneten Komponenten, Implementierungsmöglichkeiten auf Mikrocontrollern oder Computer, sowie der Analyse von den bestehenden Hardware. Es wird ein sehr breites Themenfeld analysiert, um den Umfang des Produkts kennen zu lernen. Sie dient als Wissenserarbeitung für die Konzeptionsphase.

Aufwand	25 h
Personen	Daniel Zimmermann
Arbeitsmittel	Literatur, Vorlesungsfolien, Internet
Ergebnisse	Stichpunktliste mit Realisierungsmöglichkeiten, Ordner mit relevanten Unterlagen, Grobkonzept für Hard- und Software, übersichtliches Mindmap

### 3.3 Konzeptionsphase

Die Konzeptionsphase umfasst den Entwurf der Hardware und der Software für das 3D-Laser Modul. Es werden die benötigten Bauteile ausgewählt, dimensioniert und der Schaltplan des Gerätes erstellt. Die Software wird in einzelne Module aufgeteilt und die Schnittstellen zwischen den Modulen werden definiert. Der Programmablauf wird entworfen und geeignete Algorithmen für die Signalanalyse ausgewählt.

Aufwand	30 h
Personen	Daniel Zimmermann
Arbeitsmittel	Ergebnisse von 3.2
Ergebnisse	Schaltplan des Produkts, Bauteilliste, Platinenlayouts, CAD-Zeichnungen, Konzept-Modell

### 3.4 Realisierungsphase

Die Realisierungsphase umfasst den Einkauf der effektiven Bauteile, die Erstellung der Hardware, das Implementieren der Software sowie der Integration von Software und Hardware.

Aufwand	50 h
Personen	Daniel Zimmermann
Arbeitsmittel	Ergebnisse von 3.2 und 3.3
Ergebnisse	komplettes Produkt zum Test bereit

### 3.5 Testphase

Dieses Arbeitspaket umfasst den Test der entwickelten Hard- und Software. Es werden zunächst die mechanischen Tests durchgeführt, danach werden softwaresetige Tests getätig. Anschließend findet der Test auf dem Packbot statt. Diese Phase umfasst auch die Behebung der festgestellten Mängel.

Aufwand	20 h
Personen	Daniel Zimmermann
Arbeitsmittel	Ergebnisse von 3.4, Laborgeräte, Packbot
Ergebnisse	Dokumentation der Testergebnisse, Fehlerbereinigte Hard-/Software, Liste der verbleibenden Mängel

### 3.6 Dokumentation

Dieses Arbeitspaket umfasst die gesamte Erstellung der Projektdokumentation. Das beinhaltet die gesamten Vorgaben, welche in den Anforderungen protokolliert sind. Es werden Ergebnisse aus den verschiedenen Phasen detailliert präsentiert und entsprechende Erläuterungen zu Problemstellungen und Vorgehensweisen gemacht. Jedes Unterkapitel wird reflektiert.

Aufwand	60 h
Personen	Daniel Zimmermann
Arbeitsmittel	Latex, Word, Excel
Ergebnisse	Projektdokumentation, Projektmanagement, Schlussbericht,

### 3.7 Präsentation & Poster

Dieses Arbeitspaket beinhaltet alle Phasen der Präsentation. Es wurden somit das Erstellen und Abhalten der Zwischenpräsentation, der Abschlusspräsentation und des Posters zusammengetragen.

Aufwand	21 h
Personen	Daniel Zimmermann
Arbeitsmittel	Powerpoint, Bericht
Ergebnisse	Zwischenpräsentation, Abschlusspräsentation, Poster

Erstellt durch	Kunde (Dozent) einverstanden	Version	Anzahl Seiten
Daniel Zimmermann		1.2	9

## Anhang B

# Projektmanagement

Auf den nachfolgenden Seite ist ein detaillierter Projektplan einsehbar. Die Projektphasen sind dabei übergeordnet und dienen als Meilensteine. Weitere Meilensteine sind entsprechend markiert. Die Projektphasen wurden im Pflichtenheft Anhang A erläutert und geben Auskunft über den Umfang der Phase. Im detaillierten Projektplan sind für die jeweiligen Arbeitspakete entsprechende zeitliche Soll-/Istwerte in Stunden [h] angegeben. Diese sollen Auskunft geben, welche Arbeitspakete ungenügend eingeschätzt wurden. Bedeutende Aufwandsabweichungen werden im Kapitel 6 erklärt.

## Risikoanalyse: 3D Laserscanner für mobiler Roboter

Risikotyp	Nr.	Wahr-sch.	Aus-wirk.	Ampel	Beschreibung	Behandlung und Kontrolle	Hinweise Status	Massnahmen/ Nächster Schritt
<b>Standardrisiken</b>								
Ressourcen	1	1	4	4	Materialverlust / Diebstahl	Material geschützt verräumen und lagern	nicht eingetroffen	gleichwertige Alternativen suchen und evaluieren, Projektplanung ändern
Ressourcen	2	1	2	2	Lieferngpässe /Komponenten nicht verfügbar	Liefertermine und Stückzahl frühzeitig kontrollieren	nicht eingetroffen	gleichwertige Alternativen suchen und evaluieren, Projektplanung ändern
Ressourcen	3	1	2	2	Kosten zu hoch	Vor Bestellungen Kostentabelle erstellen, Ständig Kostenüberblick wahren	nicht eingetroffen	Materialien retournieren, billiger Ersatz finden
Dokumentation	4	1	3	3	Nachvollziehbarkeit nicht gewährleistet	Dokumentation au jour halten, Notizen machen	nicht eingetroffen	Dokumentation überarbeiten, an Randzeiten Überarbeitungen machen
Planung	5	1	3	3	Ausfall durch Krankheit/Unfall	Kommunikation mit Dozenten, Reserven einplanen	nicht eingetroffen	Kommunikation mit Dozenten wenn gravieren, Plangänderung
Ressourcen	6	2	1	2	Kenntnisse und Fähigkeiten ungenügend	gründliche Recherche, Tutorial schauen, Kenntnisse erwerben	nicht eingetroffen	Unterstützung suchen
<b>Projektbezogene Risiken</b>								
Kommunikation	101	3	1	3	Termine werden nicht eingehalten	Kalendereinträge, Erinnerungen und Bestätigungsmail nutzen	nicht eingetroffen	schnellstmöglichst neuer Termin festlegen
Kunde	102	3	1	3	Anforderungen ändern sich	Regelmässiger Abgleich mit Dozent und Pflichtenheft erstellen	nicht eingetroffen	Überarbeitung Pflichtenheft oder auf Pflichtenheft verweisen
Planung	103	2	1	2	Planungsfehler entstehen	ständiger Soll/Ist vergleich, Kontrolle der Arbeitsschritte und seriöse Aufwandseinschätzung	nicht eingetroffen	Zeitliche Anpassungen, Absprachen mit Dozenten, Verlängerungen
Planung	104	1	3	4	Abgabetermine und Meilensteine nicht einhalten	ständige Projektplanung, nicht übermässige Blöcke projektieren	nicht eingetroffen	schnellstmögliche Korrektur
Kunde	105	2	3	6	Kunde ist mit Zwischenprodukt nicht zufrieden	Pflichtenheft beziehen, Während Besprechungen Notizen machen	eingetroffen	Änderungen nach Wunsch/Möglichkeit tätigen
Dokumentation	106	2	3	6	Dokumentation unvollständig	Dokumentation au jour halten, Vorgaben früh erstellen	nicht eingetroffen	schnellstmögliche Korrektur
Planung	107	2	3	6	Zeitknappheit	ständige Projektplanung, Prioritäten setzen	eingetroffen	Prioritäten setzen, übermässige Arbeitsblöcke vereinfachen
<b>Produktbezogene Risiken</b>								
Hardware	201	2	3	6	Konzeption nicht umsetzbar	Mehrere Konzepte evaluieren und realistisch dimensionieren	nicht eingetroffen	Alternative suchen und realisieren
Software	202	1	3	3	Softwarecodes nicht implementierbar oder Packages nicht nutzbar	Verfügbarkeit und Kompatibilität prüfen vorgängig	nicht eingetroffen	Alternative suchen, selbstständig implementieren
Hardware	203	1	2	2	mech. Komponenten zu ungenau	sorgfältige Kontrolle vor Produktion	nicht eingetroffen	schnellstmöglichst ersetzen
Software/Hardw are	204	2	2	4	Datenverarbeitung zu rechenintensiv	Dimension ausloten, Unnütze Speichervorgänge freigeben, Unnötige Dienste ausschalten	nicht eingetroffen	Optimieren, bessere Hardware evaluieren, Alternative mit Laptop
Software/Hardw	205	2	2	4	Testbarkeit auf Packpot nicht	Schnittstellen beachten	eingetroffen	Anpassungen nach Situation, Besprechung mit
Software	206	2	3	6	Hardware mit Software nicht kombinierbar	simple Ideen vorziehen, Recherche betreiben, Alternative Implementierung machen	nicht eingetroffen	Alternative Implementierung
Software	207	2	2	4	C++ / Python Erfahrungen nicht ausreichend	Software einfach halten, keine Packages verwenden die nicht verstanden sind	nicht eingetroffen	Repetition, Unterstützung holen bei Assistent und Dozent

Legende	
Wertebereich	Risikotypen
1	Ressourcen
2	Planung
3	Kunde
	Kommunikation
	Dokumentation
	Software
	Hardware

# Anhang C

## Risikoanalyse

## Risikoanalyse: 3D Laserscanner für mobiler Roboter

Risikotyp	Nr.	Wahr-sch.	Aus-wirk.	Ampel	Beschreibung	Behandlung und Kontrolle	Hinweise Status	Massnahmen/ Nächster Schritt
<b>Standardrisiken</b>								
Ressourcen	1	1	4	4	Materialverlust / Diebstahl	Material geschützt verräumen und lagern	nicht eingetroffen	gleichwertige Alternativen suchen und evaluieren, Projektplanung ändern
Ressourcen	2	1	2	2	Lieferngpässe /Komponenten nicht verfügbar	Liefertermine und Stückzahl frühzeitig kontrollieren	nicht eingetroffen	gleichwertige Alternativen suchen und evaluieren, Projektplanung ändern
Ressourcen	3	1	2	2	Kosten zu hoch	Vor Bestellungen Kostentabelle erstellen, Ständig Kostenüberblick wahren	nicht eingetroffen	Materialien retournieren, billiger Ersatz finden
Dokumentation	4	1	3	3	Nachvollziehbarkeit nicht gewährleistet	Dokumentation au jour halten, Notizen machen	nicht eingetroffen	Dokumentation überarbeiten, an Randzeiten Überarbeitungen machen
Planung	5	1	3	3	Ausfall durch Krankheit/Unfall	Kommunikation mit Dozenten, Reserven einplanen	nicht eingetroffen	Kommunikation mit Dozenten wenn gravieren, Plangänderung
Ressourcen	6	2	1	2	Kenntnisse und Fähigkeiten ungenügend	gründliche Recherche, Tutorial schauen, Kenntnisse erwerben	nicht eingetroffen	Unterstützung suchen
<b>Projektbezogene Risiken</b>								
Kommunikation	101	3	1	3	Termine werden nicht eingehalten	Kalendereinträge, Erinnerungen und Bestätigungsmail nutzen	nicht eingetroffen	schnellstmöglichst neuer Termin festlegen
Kunde	102	3	1	3	Anforderungen ändern sich	Regelmässiger Abgleich mit Dozent und Pflichtenheft erstellen	nicht eingetroffen	Überarbeitung Pflichtenheft oder auf Pflichtenheft verweisen
Planung	103	2	1	2	Planungsfehler entstehen	ständiger Soll/Ist vergleich, Kontrolle der Arbeitsschritte und seriöse Aufwandseinschätzung	nicht eingetroffen	Zeitliche Anpassungen, Absprachen mit Dozenten, Verlängerungen
Planung	104	1	3	4	Abgabetermine und Meilensteine nicht einhalten	ständige Projektplanung, nicht übermässige Blöcke projektieren	nicht eingetroffen	schnellstmögliche Korrektur
Kunde	105	2	3	6	Kunde ist mit Zwischenprodukt nicht zufrieden	Pflichtenheft beziehen, Während Besprechungen Notizen machen	eingetroffen	Änderungen nach Wunsch/Möglichkeit tätigen
Dokumentation	106	2	3	6	Dokumentation unvollständig	Dokumentation au jour halten, Vorgaben früh erstellen	nicht eingetroffen	schnellstmögliche Korrektur
Planung	107	2	3	6	Zeitknappheit	ständige Projektplanung, Prioritäten setzen	eingetroffen	Prioritäten setzen, übermässige Arbeitsblöcke vereinfachen
<b>Produktbezogene Risiken</b>								
Hardware	201	2	3	6	Konzeption nicht umsetzbar	Mehrere Konzepte evaluieren und realistisch dimensionieren	nicht eingetroffen	Alternative suchen und realisieren
Software	202	1	3	3	Softwarecodes nicht implementierbar oder Packages nicht nutzbar	Verfügbarkeit und Kompatibilität prüfen vorgängig	nicht eingetroffen	Alternative suchen, selbstständig implementieren
Hardware	203	1	2	2	mech. Komponenten zu ungenau	sorgfältige Kontrolle vor Produktion	nicht eingetroffen	schnellstmöglichst ersetzen
Software/Hardw are	204	2	2	4	Datenverarbeitung zu rechenintensiv	Dimension ausloten, Unnütze Speichervorgänge freigeben, Unnötige Dienste ausschalten	nicht eingetroffen	Optimieren, bessere Hardware evaluieren, Alternative mit Laptop
Software/Hardw	205	2	2	4	Testbarkeit auf Packpot nicht	Schnittstellen beachten	eingetroffen	Anpassungen nach Situation, Besprechung mit
Software	206	2	3	6	Hardware mit Software nicht kombinierbar	simple Ideen vorziehen, Recherche betreiben, Alternative Implementierung machen	nicht eingetroffen	Alternative Implementierung
Software	207	2	2	4	C++ / Python Erfahrungen nicht ausreichend	Software einfach halten, keine Packages verwenden die nicht verstanden sind	nicht eingetroffen	Repetition, Unterstützung holen bei Assistent und Dozent

Legende	
Wertebereich	Risikotypen
1	Ressourcen
2	Planung
3	Kunde
	Kommunikation
	Dokumentation
	Software
	Hardware

# Anhang D

## Aufgabenstellung

Horw, 18. September 2017  
Seite 1/2

## **Industrieprojekt im Fachbereich Elektrotechnik & Informationstechnologie**

Aufgabe für Herrn Daniel ZIMMERMANN  
**3D Laserscanner für mobilen Roboter**

### **Fachliche Schwerpunkte**

Signalverarbeitung & Kommunikation, Automation & Embedded Systems

### **Einleitung**

Roboter, wie der iRobot Packbot sind für Einsätze in schwierigem Gelände konzipiert. Für solche Einsätze ist es notwendig eine Karte, am besten als dreidimensionales Modell der Umgebung zu erstellen. Im Rahmen dieser Arbeit, soll ein 3D Laser-Modul für einen mobilen Roboter entwickelt werden, welches es dem Roboter erlaubt die Umgebung während der Fahrt zu vermessen.

### **Aufgabenstellung**

Es soll ein 3D-Laser-Modul entwickelt werden, welches einen bestehenden 2D-Laser um eine Achse rotiert und so die Vermessung der Umgebung in 3D erlaubt. Die gemessenen Distanzen sollen von einem PC aufgenommen und dem mobilen Roboter einmal pro Umdrehung zur Verfügung gestellt werden.

Üblicherweise bewegt sich der Roboter während diesen Messungen. Im Idealfall wird die Bewegung des Roboters gemessen und die Messdaten entsprechend kompensiert.

Das entwickelte Laser-Modul soll im Rahmen der Arbeit auf dem Packbot-Roboter getestet werden.

### **Termine**

Start der Arbeit:

Montag 18.9.2017

Zwischenpräsentation:

Zu vereinbaren im Zeitraum 30.10.-24.11.2017

Abgabe Schlussbericht:

Freitag 22. Dezember 2017, 16:00 im D311, an R. Andrist

Abgabe Poster-File:

Montag 30. Januar 2018 per Mail an Betreuer und H. R. Andrist

Abschlusspräsentation:

Zu vereinbaren im Zeitraum 18.12.2017 - 26.1.2018