HTL | MÖSSINGERSTRASSE

HÖHERE TECHNISCHE BUNDESLEHRANSTALT

KLAGENFURT, MÖSSINGERSTRASSE

ABTEILUNG ELEKTROTECHNIK

DIPLOMARBEIT

Titel der Diplomarbeit Deutsch

Automated-Factory-Storage-System

JAHRGANG 5AHET

eingereicht von	Benedikt Simbürger
	Vincent Sonvilla
	Nikolaj Voglauer
	Elena Widmann
Projektbetreuer	Dipl.-Ing. Christian Sallinger

Diese Diplomarbeit entspricht den Standards gemäß dem Leitfaden zur Umsetzung der Reife- und Diplomprüfung des BMBWF in der letztgültigen Fassung.

Klagenfurt, am 04.04.2025

HTL | MÖSSINGERSTRASSE

EIDESSTATTLICHE ERKLÄRUNG

Ich versichere an Eides statt, dass ich diese Diplomarbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe. Alle Gedanken, die im Wortlaut oder in grundlegenden Inhalten aus unveröffentlichten Texten oder aus veröffentlichter Literatur übernommen, oder mit künstlicher Intelligenz generiert wurden, sind ordnungsgemäß gekennzeichnet, zitiert und mit genauer Quellenangabe versehen.

Verfasser/Verfasserin

Benedikt Simbürger	Vincent Sonvilla
Nikolaj Voglauer	Elena Widmann

Klagenfurt, am 04.04.2025

Kurzbeschreibung

Beschreiben Sie das Projekt an dieser Stelle mit maximal zwei aussagekräftigen Sätzen, die dem Leser die Projektidee kompakt vermitteln und eine thematische Zuordnung ermöglichen. Die Kurzbeschreibung, im Umfang von einer A4-Seite, umfasst die wesentlichen Aspekte des Projektes in sozialer und technischer Hinsicht. Die Zielgruppe der Kurzbeschreibung sind auch Nicht-Techniker! Diese Beschreibung wird für Wettbewerbe und PR-Aktivitäten verwendet. Diese Inhalte sind unter dem Menüpunkt „About“ bzw. „Beschreibung“ auf der Projekthomepage dar-zustellen. Auf die Formulierung der Kurzbeschreibung sollte sehr viel Wert gelegt werden, weil viele Leser oft nur diese Seite lesen und den Rest lediglich durchblättern. Erstellen Sie die finale Kurzbeschreibung erst nach der Fertigstellung der Diplomarbeit.

Aufgabenstellung

- Warum ist die Themenstellung von Interesse?
- Was ist die vorgegebene Zielsetzung?
- Welche Ergebnisse sind zu erreichen?

Realisierung

- Von welchem Stand der Technik im Umfeld der Aufgabenstellung wurde ausgegangen?
- Welche Lösungsansätze sind grundsätzlich möglich?
- Warum wurde ein bestimmter Lösungsansatz gewählt?
- Welche experimentelle, konstruktive oder softwaretechnische Methodik wurde angewendet?
- Auf welche fachtheoretischen Grundlagen wurde aufgebaut?
- Welche wirtschaftlichen Überlegungen wurden angestellt?

Ergebnisse

- Worin besteht der konkrete Beitrag zur Lösung der Aufgabenstellung (Prototyp, Entwurfsplanung, Softwareprodukt, Businessplan etc.)?
- Kann das Ergebnis durch eine typische Grafik, ein Diagramm bzw. ein Foto illus-triert werden?
- Kann in die Vollversion der Diplomarbeit Einsicht genommen werden?

Kurztitel: Kurztitel der Diplomarbeit (etwa. 30 Zeichen)

Schlüsselwörter: Maximal fünf aussagekräftige und durch Kommas getrennte Schlüsselwörter, welche die Inhalte möglichst gut beschreiben.

Abstract

Die Kurzbeschreibung in englischer Sprache sollte sinngemäß exakt der Kurzbeschreibung in deutscher Sprache entsprechen, siehe Abschnitt „Kurzbeschreibung“.

Short title:

Keywords:

Inhaltsverzeichnis

1 Allgemeiner Teil **7**
- 1.1 Ausgangssituation 1-2 . . . 7
  - 1.1.1 Anforderungen . . . 7
- 1.2 Potentielle Lösungen 2 . . . 7
  - 1.2.1 Lagermethoden . . . 7
  - 1.2.2 Steuerung 4/9 . . . 8
  - 1.2.3 Schaltschrank 4/9 . . . 8
- 1.3 Verfolgter Lösungsansatz 3/4 . . . 8
- 1.4 Methodik 5 . . . 8
  - 1.4.1 Software 2 . . . 8
  - 1.4.2 Hardware 2 . . . 9
  - 1.4.3 Fertigung 1 . . . 9
- 1.5 Sicherheitstechnik 1 . . . 9
- 1.6 Kooperationspartner . . . 9

2 Hardwareentwicklung, Softwarebackend und Benutzeroberfläche (Benedikt Simbürger) **10**
- 2.1 Hardware . . . 10
  - 2.1.1 Planung - Grundanforderungen . . . 10
  - 2.1.2 Vorgehensweise . . . 11
  - 2.1.3 Rahmen . . . 12
  - 2.1.4 X-Achse . . . 13
  - 2.1.5 Y-Achse . . . 14
  - 2.1.6 Z-Achse . . . 18
  - 2.1.7 Lager . . . 19
  - 2.1.8 Querförderer . . . 19
  - 2.1.9 Fertigung der Einzelteile . . . 20
  - 2.1.10 Aufbau . . . 23
- 2.2 Software und Benutzeroberfläche . . . 24
  - 2.2.1 Grundlegendes . . . 24
  - 2.2.2 Aufbau . . . 24
  - 2.2.3 Benutzeroberfläche . . . 25
  - 2.2.4 Backend . . . 27
  - 2.2.5 APIs . . . 28
  - 2.2.6 Lageralgorithmus für SPS . . . 29
  - 2.2.7 Datenbanken . . . 30
  - 2.2.8 Docker . . . 31
  - 2.2.9 Artikelsuche . . . 32

3 Integration und Programmierung der Steuerungstechnik (Vincent Sonvilla) **36**

3.1 Aufgabenstellung . . . 36
3.2 Tia-Portal . . . 36
3.2.1 Grundlagen . . . 36
3.2.2 Programmiersprachen . . . 36
3.2.3 Libraryeinbindung . . . 36
3.3 Motorenansteuerung . . . 36
3.4 SPS-Server Kommunikation . . . 36
3.4.1 Zur Auswahl stehende Kommunikationsprotokolle . . . 36
3.4.2 Verbindungsherstellung . . . 37
3.4.3 Datenfilterung . . . 37
3.5 Herausforderungen . . . 38

4 Elektroplanung und Realisierung (Nikolaj Voglauer) 39

5 Sensorik und Sicherheitstechnik (Elena Widmann) 40
5.1 Aufgabenstellung . . . 40
5.2 Sensorik . . . 40
5.2.1 Endschalter . . . 40
5.2.2 Referenztaster . . . 41
5.2.3 Lichttaster . . . 42
5.2.4 Barcode-Scanner . . . 42
5.3 AS-Interface . . . 42
5.3.1 Allgemeines . . . 42
5.3.2 Programmierung im TIA-Portal . . . 42
5.3.3 Unterverteilerplatine . . . 42
5.4 Sicherheitstechnik . . . 42
5.4.1 Grundanforderungen und Planung . . . 42
5.4.2 Realisierung . . . 42

6 Anhang 43
6.1 Projektmanagement . . . 43
6.1.1 Aufgabenstellung des Gesamtprojekts . . . 43
6.1.2 Scrum-Projektplan . . . 43
6.1.3 Terminplanung . . . 45
6.2 Inbetriebnahme . . . 46
6.3 Kostenaufstellung . . . 47
6.4 Besprechungsprotokolle . . . 48
6.5 Arbeitsnachweis . . . 49
6.6 Wettbewerbe . . . 50
6.7 Businessplan (optional) . . . 51

Literaturverzeichnis 52

Abbildungsverzeichnis **53**

Tabellenverzeichnis **54**

1 Allgemeiner Teil

1.1 Ausgangssituation 1-2

1.1.1 Anforderungen

Um die Automatisierung dieses Prozesses zu ermöglichen, soll ein System entwickelt sowie gebaut werden, welches eine lagernde Box automatisch zu einer Kommissionierstation bringt. An dieser soll die Möglichkeit bestehen, lagerndes Inventar anzufordern, Lagerbestand auszufassen oder aufzufüllen sowie Boxen wieder einzulagern.
Weiters muss dieses System erweiterbar sein, um zukünftig neuen Lagerplatz hinzuzufügen, als auch die Möglichkeit zu bieten, andere Systeme wie Bauteilvereinzelung einzubinden.
Auch soll das AFSS mobil sein – dies gilt sowohl für Mechanik als auch Elektrik, da es nicht am endgültigen Standort errichtet wird und um bei möglichen Umbauarbeiten einen einfachen Transport zu ermöglichen.
Die Rahmenbedingungen zur Umsetzung dieser Ziele sind stark davon geprägt, dass alle mechanischen Bauteile, die eigens gefertigt werden müssen, so geplant werden, dass dies mit HTL-Mitteln möglich ist.
Es soll nicht nur produzierbar, sondern auch reproduzierbar sein. Daher soll die Dokumentation der Funktion sowie des Umsetzungsprozess so erfolgen, dass die Weiterführung dieses Projekts durch andere Schülerinnen und Schüler möglich ist.

1.2 Potentielle Lösungen 2

1.2.1 Lagermethoden

Die Industrie gibt viele Vorbilder in, wie ein Boxenbasierendes Lagersystem aufgebaut sein kann.
Eine sehr Platzeffiziente Variante ist beispielsweise die PickEngine von KNAPP [1]. bei dieser Variante werden Boxen in mehreren Ebenen übereinander Gelagert. Auf jeder Ebene gibt es bewegliche Roboter, welche die Boxen abholen und zu einem Lift bringen, von dem aus die Box dann zur Kommisionierstation gelangt. Dieses System ist ist sehr platzeffizient und ausfallsicher. Jedoch ist es schwer möglich ein solches System in Miniatur zu Bauen, da die Hardwarefertigung sehr komplex ist.
Eine weitere Möglichkeit wäre ein Rotierendes Magazin, in welchem die Boxen auf einem horizontalen Karusell gelagert sind. Wenn ein bestimmtes Produkt benötigt wird, werden die Platformen soweit weiterrotiert, bis auf die gewünschte Box zugegriffen werden kann. Das Prinzip dieses Systems ist zwar recht simpel, jedoch ist es es schwierig die Mechanik der Rotation mit HTL-mitteln so zu bauen, dass ein Dauerbetrieb möglich ist.
Weiters gibt es die Möglichkeit, Ware vertikal, von oben zu Lagern. Über der Lagerstätte fährt dann ein Roboter, der in der Lage ist, die Boxen auszuheben und dann umzuschichten

oder ein- und auszulagern. So kann eine recht hohe Lagerdichte erreicht werden, jedoch ist der Durchsatz etwas begrenzt, da, wenn Boxen benötigt werde, welche nicht am obersten Platz sind, erst alle anderen umgeschichtet werden müssen. Ausserdem ist so eine Lagervariante erst dann platzeffizienz, wenn sie sehr hoch gebaut wird. Wenige Lagen lohnen sich noch nicht da trotz geringer Höhe, sehr viel Fläche verbraucht wird.

1.2.2 Steuerung 4/9

1.2.3 Schaltschrank 4/9

1.3 Verfolgter Lösungsansatz 3/4

Nach Abwägung der Alternativen wurde sich bei der Auswahl der Lagermethode, an einem klassischen Palettenlager inspiriert. Der grundgedanke eines Portalsystems, welches mit einer Gabel Boxen in einem Regal ein- und aushebt. Diese Lagervariante behält die Ballance aus technischer Kopmlexität, sowie dem Umsetzungsvermögen an der HTL. Zudem ist gibt es bei dieser Variante ebenfalls effizienzsteigerungspotential, da die möglichkeit besteht, links und rechts des Roboters Bestand zu lagern. jedoch wird diese Variante nicht forciert, da die Komplexität eines solchen Mechanismus im kleinen Maßstab und mit eingeschränkter Fertigungstechnik recht schwierig ist.
Es wird also ein Lagersystem gewählt, bei dem zwei Achsen hin- und her Verfahren und eine art Gabel die Boxen ein und Aushebt. Um die Boxen zur Kommisionierstation zu Bringen wird ein Förderband verwendet, welches Längs zum Lager verläuft. Jedoch kann der LAgerroboter die Boxen nicht selbstständig auf das Förderband laden, zu diesem Zweck wid ein sog. Querförderer verwendet, der die Box von einem Temporären Lagerplatz auf das Förderband und wieder zurück schiebt. An diese Förderband können ausserdem Weiterer solcher Lagerschränke aufgestellt werden, um mehr Lagerplatz zu schaffen, als auch weitere andere Systeme angschlossen werden. Ausserdem wird ein Lagerschrank als Komplettsystem konzipiert. Durch unterbringugn aller Systeme in einem einzigen Objekt, kann durch montage von Rollen einfach Transportfähigkeit sichergestellt werden.
Zur Steuerung dieser Anlage wird auf eine SPS-Steuerung zurückgegriffen. Dies ist Industriestandard, und so eine möglichst nahe Abbildung einer großen Logistikanlage darstellen. Die Steuerung sowie die Versorgung der Anlage finden in einem Schaltschrank platz, der Ebenfalls fahrbar ist. Weiters sind alle Anschlüsse usw. darauf ausgelegt, den Schaltschrank vom Lagerschrank zu trennen.

1.4 Methodik 5

1.4.1 Software 2

Zur mechanischen Planung des AFSS wird vorrangig Fusion360 genutzt. Dies ist ein 3D-CAD (Computer-Aided-Design) Programm, welches eine breite Palette von Funktionen bietet, jedoch noch sehr bedienerfreundlich ist. Weiters ist es sehr nützlich, dass Fusion360

eine integrierte Cloud-Speicherung anbietet. So können die Konstruktionen von anderen Personen einfach eingesehen werden.

1.4.2 Hardware 2

1.4.3 Fertigung 1

1.5 Sicherheitstechnik 1

1.6 Kooperationspartner

2 Hardwareentwicklung, Softwarebackend und Benutzeroberfläche (Benedikt Simbürger)

2.1 Hardware

2.1.1 Planung - Grundanforderungen

Grundlegende Anforderungen zur Planung der AFSS-Mechanik sind Transportfähigkeit, eine möglichst einfache Realisierung mit HTL-Mitteln und möglichst wenige Kompromisse in der Funktion oder Zuverlässigkeit eingehen zu müssen.
Die Anforderung der Transportfähigkeit limitiert die Größe des Lagers auf 2.3 m Länge, um im Lift transportiert werden zu können, und auf 1.9 m Höhe aufgrund der Türhöhe im Keller. Weiters müssen auch noch Rollen an den Rahmen angebracht werden, um das Lager ohne großen Mehraufwand bewegen zu können. Diese Extrahöhe der Räder (ca. 80 mm) limitiert den Rahmen weiter.
Nun soll dieser rund 2.25 m lange und 1.8 m hohe Raum optimal genutzt werden, um eine möglichst große Lagerdichte sicherstellen zu können.
Um eine möglichst gute Erweiterbarkeit sowie eine Fertigung an der Schule zu ermöglichen, sollen für die mechanische Trägerkonstruktion sogenannte Item-Profile verwendet werden.

Item

Abbildung 2.1: Item Profil mit Standartverbindungssatz, Quelle: [2]

Das Item-Profilsystem ist ein System, welches Aluminium-Extrusionen in verschiedenen Ausführungen sowie viele Verbindungsmöglichkeiten zu sich selbst sowie anderen mechanischen Elementen bietet. Hierbei gibt es eine breite Auswahl an Profilen, von 20x20 mm bis 40x40 mm Querschnitt. Für alle Komponenten mit hoher mechanischer Beanspruchung werden 40x40-Extrusionen verbaut, da diese eine besonders hohe Biegefestigkeit aufweisen. Für Anwendungen mit geringerer Beanspruchung sowie aus Platz- und Gewichtsparmaßnahmen werden 20x20-Extrusionen verwendet. Zur Verbindung zu anderen Bauelementen gibt es die Möglichkeit, sogenannte Nutensteine mit verschiedenen Gewinden in die Nut einzulegen und dort Platten o. Ä. anzuschrauben. Um Item-Profile untereinander zu verbinden, können Standardverbindungssätze wie in Abb. 2.1 verwendet werden.

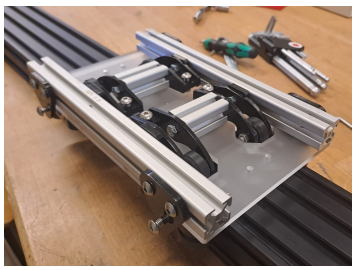
Die Anwendung im AFSS erfordert ausserdem recht lange Verfahrwege. Um dies kostengünstig umsetzen zu können, werden V-Slot-Profile verwendet.

V-Slot
Auch V-Slot-Profile sind Aluminium-Extrusionen. Diese können grundsätzlich auch ähnlich wie Item-Profile mit Nutensteinen etc. verwendet werden, sind aber zusätzlich darauf ausgelegt, dass ein V-Wheel in einer Narbe des Profils rollen kann (siehe Abb. 2.2). Diese Profile sind in einer C-Profil-Form erhältlich. Diese sind für die langen Verfahrwege optimal, da einerseits auch breitere und mehr V-Wheels verwendet werden können, sowie einen sehr großen Widerstandsmoment aufweisen.

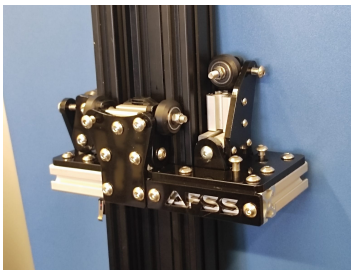
2.1.2 Vorgehensweise

Aufgrund der besonderen und sehr komplexen Anforderungen dieser Mechanik erfolgt die Entwicklung in mehreren Iterationen. Nach Auslegung der Grundparameter wird eine Grundkonstruktion erstellt, um mögliche Lösungsansätze für die jeweiligen Komponenten zu skizzieren. Durch diese grobe Planung können viele Konzepte mit geringerem Zeitaufwand iteriert werden und auch mögliche Missverständnisse o. Ä. frühzeitig aufgeklärt und überarbeitet werden. Weiters werden bei diesem Prozess wichtige Fähigkeiten in der Bedienung der CAD-Software gewonnen und so die Geschwindigkeit der zukünftigen Designiterationen beschleunigt.
Um bestimmte Elemente der Mechanik einzeln zu testen, werden auch mehrere Prototypen gebaut und die gewonnenen Erkenntnisse in die finale Konstruktion miteingebunden.

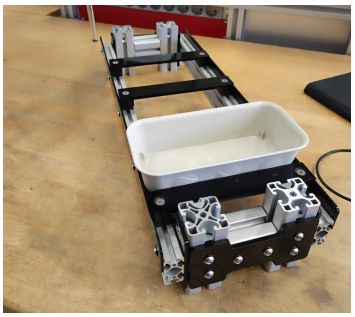
Abbildung 2.2: V-Slot-Profil mit V-Wheel, Quelle: [3]



(a) X-Achse



(b) Y-Achse



(c) Lagerregal

Abbildung 2.3: Prototypen

Rahmenbedingungen für die Fertigung
Damit die Fertigung der Mechanik an der HTL-Mössingerstraße in einem realistischen Zeitrahmen möglich ist, müssen gewisse Rahmenbedingungen bei der Planung beachtet werden.

- Für Verbindungen mit geringer bis mittlerer mechanischer Beanspruchung werden Bauteile so geplant, dass diese im Lasercutter gefertigt werden können, sowie in den Stärken, die der Werkstätte zur Verfügung stehen (3, 4 und 5 mm). Dabei muss beachtet werden, dass die Acrylplatten gegossen sind und dadurch recht hohe Toleranzen (bis zu +0.3 mm) aufweisen.
- Verbindungen zwischen Platten und Item- oder V-Slot-Profilen werden möglichst einheitlich gestaltet. Grundsätzlich gilt: Verbindungen mit Item-Baureihe (BR) 5 (20x20) werden grundsätzlich in M5 ausgeführt, außer bei Platzmangel in M3. Verbindungen mit BR 8 (40x40) werden in M8 ausgeführt.
- Bei Bauteilen mit hoher mechanischer Beanspruchung wird Aluminiumblech mit 5 mm Stärke verwendet. Dieses wird zwar CNC-gefräst, doch um bei der Fertigung Zeit zu sparen, wird davon abgesehen, Taschen o. Ä. einzuplanen. Stattdessen wird immer ein flaches Profil verwendet, welches sich mit 2.5D-Fräsverfahren mit Durchgangsfräsen, ähnlich einem Lasercutter, fertigen lässt. Bei Aluminiumteilen wird darauf geachtet, dass diese immer aus 5 mm dickem Aluminiumeloxal gefertigt werden.

Konstruktionsvorgang

Um ein so umfangreiches Projekt umsetzen zu können, muss auch bei der 3D-Planung möglichst viel Ordnung herrschen. Um möglichst effizient zu arbeiten, werden wiederverwendete Bauteile in einer eigens angelegten Bauteilbibliothek abgelegt. So können beispielsweise Aluminiumprofile in Normlängen, oder Sensoren, einfach die aktuelle Konstruktion eingefügt werden, ohne diese jedes mal neu zu Konstruieren.
Auch in einer Konstruktion selbst ist auf die übersicht zu achten. Zu diesem Zweck ist es sehr sinnvoll, zusammenhängende Objekte in Komponenten zusammenzufassen. Diese sind vergleichbar mit Ordnern. Wie auch Ordner können auch Komponenten ineinander Verschachtelt werden. So ist es möglich eine sauberes Struktur umzusetzen, welche es beispielsweise vereinfacht einzelne Konstruktionsteile zu betrachten, oder Änderungen an nur diesen durchzuführen.

2.1.3 Rahmen

Der Rahmen des AFSS bezeichnet jene Struktur, welche als äußerstes Gehäuse, sowie grundlegende mechanische Stabilität bietet. Es ist geplant, dass es die Anforderungen von Maximalhöhe und -länge optimal ausnutzt sowie breite minimal hält.
Umgesetzt wird dies mit einem Gerüst aus 40x40-Aluminium Extrusionen. Dieses hat weiters oben udn unten noch zwei Längsstreben, die dazu dienen, eine möglichst platzsparende Aufhängung, der X-Achsen V-Slot-Profile, zu realsisieren. Auf einer Seite wird weiters eine Aussparung eingeplant, um genug Freiraum übrig zu lassen, damit der Querföreder die Boxen reibungslos auf das Förderband bringen kann.
Weiters muss noch eingeplant wieder, dass noch Rollen zum Transport angebracht werden

müssen. Dadurch, dass diese Auswirkungen auf die Maximalhöhe haben, muss dass in die Auslegung der Y-Achse einfließen, um Bewegung durch Türen zu ermöglichen.

2.1.4 X-Achse

Die X-Achse des AFSS ist das Mechanische Herzstück. Sie ist jene Achse, welche sich horizontal bewegt. Sie hat einen langen Verfahrweg und hohen mechanische Anforderungen (sie muss ihr Eigengewicht und das Gewicht der YZ-Achse stützen). Grundelement der X-Achse sind V-Slot C-Profile, welche oben und unten aufgehängt werden. Diese sind in 1.5 m Länge erhältlich, deshalb wird die ganze Achse darauf ausgelegt, 2 Stück also 3 m Schiene zu verwenden. Diese sind weiters mit 6 Nivelierungsschrauben pro V-Profil ausgestattet, um diese möglichst Parallel zueinander sowie in Waage zu positionieren.

Grundanforderungen
Auf der X-Achse verläuft der X-Schlitten, dieser hat die Aufgabe die Y-Achse zu positionieren. Um dies zu bewerkstelligen, muss dieser alle nötigen Motoren und Sensoren beinhalten. Weiters muss er einen Angriffspunkt bieten, um diesen zu bewegen. Ausserdem soll der Schlitten horizontal und vertikal möglichst kompakt sein, um einen möglichst großen Verfahrweg in X- als auch Y- Richtung zu ermöglichen.

Antrieb
Angetrieben wird die X-Achse mit zwei Schrittmotoren, welche jeweils an einer Schiene (oben und unten) angebracht werden. Diese treiben mithilfe eines Zahnriemen die zwei X-Schlitten an. Als Zahnriemen wurde aufgrund der hohen Kraftübertragung, sowie großer Länge auf ein AT5-Zahnprofil mit 16 mm Riemenbreite der Firma Mähdler gesetzt. Dieser wird vom Motor über eine Entsprechende Zahnscheibe angetrieben, und am Shuttle befestigt. Als Schrittmotor wird aus Einfachheitsgründen das selbe Modell wie bei der Y-Achse verwendet. Diese erzeigen auch genügend Moment um die X-Achse anzutreiben.

Riemenspannung X-Achse
Da der Riemen für eine zuverlässige Kraftübertragung bei so langen Verfahrwegen eine relativ große Vorspannkraft benötigt wird, ist das Zahnriemenklemmelement auf der X-Achse auch dementsprechend auszuführen. Um den Zahnriemen zu greifen, muss ein Negativ in Aluminium gefräßt werden, in dieses der Zahnriemen dann auf beiden Seiten des Schlittens geklemmt wird. Die Aufhängung der einen Seite ist einfach mit den Item-Profilen verschraubt, lässt aber noch etwas platz, um den Riemen mit der Hand vorzuspannen. Auf der anderen Seite ist das Klembrett über zwei Gewindestangen mit dem restlichen Shuttle verbunden. Diese können festgezogen werden, um die nötigen Vorspannkräfte zu erzeugen. Ausserdem ist diese Spannvorrichtung in einem Formfaktor ausgeführt, welcher es erlaubt, dierekt darüber die Motoren der Y-Achse anzubringen.

Achsenführung
Die Führung der Achse im V-Slot-Profil wird mit V-Wheels durchgeführt. Sechs V-Wheels werden von oben auf das C-Profil gedrückt. Jedes Rad wird einzeln aufgehangen. Weiters ist überall eine Schraube mit welcher das Rad weiter in die Führung hienein gedrückt werden kann, sowie ein bis zwei Klemmpunkte, um bei Betrieb die Last von der Spannschraube nehmen zu können. Durch den geringen Platz im Schlitten ist es durchaus eine Herausforderung, dass all diese Mechanik, neben der Spannelemente und Motoren in einem so kleinen Shuttel platz finden. Auf der Seite des Schlittens werden noch weitere V-Wheels angebracht, welche die Spurführung übernehmen. Im oberen Schlitten werden nur diese Führungsräder verbaut, da es nicht möglich und nötig ist, vertikale Kräfte zu unterstützen.

Kabelführung
Um Sensoren und Aktoren der Y- und Z-Achse zu unterstützen, müssen dementsprechende Leitungen auf das X-Shuttel verlegt werden. Die wird mithilfe einer Kabelschleppkette der Firma Igus umgesetzt. Diese wird parallel zum unteren C-Profil verlegt und unter Berücksichtigung der Biegeradien am X-Shuttel befestigt. Bei dieser ist darauf zu achten, dass sie alle benötigten Leitungen, sowie genügend Freiraum für die Biegung einhält [4]. Weiters müssen Signal- und Aktorstromkreise durch trennstege voneinander getrennt werden.
Um die weiteren Geräte am AFSS zu versorgen, müssen zusätzlich noch Verdrahtungskanäle am Rahmen angebracht werden.

Sensoren
Die Sensorik ist bei jeder Achse ähnlich aufgebaut. Immer zwei Endschalter und ein Referenzierschalter. Bei der X-Achse werden als Endschalter mechanische Rolltaster verwendet. Diese werden neben den V-Slot-Profilen befestigt und von einem, vom Schlitten abstrehenden Arm ausgelöst.

2.1.5 Y-Achse

Als Y-Achse wird jene Achse bezeichnet, die ihre Bewegung vertikal durchführt. Sie hat die Aufgabe, die Z-Achse bzw. das YZ-Shuttel auf Position zu bringen. Wichtig hierbei ist jedoch, dass die Y-Achse die Aufgabe des aufheben der Box übernimmt.

Antriebsauslegung
Dadurch, dass die Y-Achse sowohl YZ-Shuttle als auch die Boxen aufheben muss, muss der Antrieb dementsprechend dimensioniert werden. Als Formfaktor sol ein Nema23-Schrittmotor verwendet werden. Diese sind weit verbreitet und zu Servomotoren verhältnismäßig kostengünstig. Als grundformfaktor wird ein 2 Nm Motor gewählt, nun soll überprüft werden, ob dieser die Last der YZ-Achse auch antreiben kann.

$$F = \frac{M}{\frac{d}{2} \cdot 1000} \cdot n$$

$$F = \frac{2\,\mathrm{Nm}}{\frac{30\,\mathrm{mm}}{2} \cdot 1000} \cdot 2 = 266\,\mathrm{N}$$

F : Antriebskraft der Y-Achse $[N]$
M : Drehmoment eines Motors $[Nm]$
d : Durchmesser der Antriebszahnscheibe $[mm]$
n : Anzahl der Antriebe

Bei Konstruktion einer früheren Version des YZ-Shuttels wurde erfasst, dass das Shuttel bis zu 13 kg Wiegen kann. Dies wird zwar in einer späteren Iteration des Designvorgangs noch verbessert, dient jedoch als richtwert der Antriebsauslegung. 13 kg erzeugen ohne Berücksichtigung von Reibung knapp 130 N. Die Antriebskraft der Schrittmotoren reicht auf jeden fall aus. Doch diese Überdimensionierung ist unter dem Aspekt, dass die Antriebe über keine Bremse verfügen, also das gesamte Gewicht der Z-Achse und der Kabelschleppkette immer unterstützen müssen, durchaus sinnvoll.

Zahnriemen und Umlenkung

Als Zahnriemen wird auch bei der Y-Achse auf eine AT5x16-Profil gesetzt. Doch hier gestaltet sich die Positionierung ebendieses nicht so simpel wie bei der X-Achse. Da der Zahnriemen beim YZ-Shuttle an einem bestimmten Punkt befestigt werden muss, muss er auch dort wieder rückgeführt werden. Die Umlenkung des Zahnriemens gestaltet sich jedoch wesentlich anspruchsvoller als bei der X-Achse, da einerseits ain möglichst kompakter Formfaktor angestrebt werden muss, sowie andererseits eine aufhängungstechnisch sehr unvorteilhafte Positionierung vor dem V-Slot C-profil erforderlich ist. Aus diesem Grund wir eine konstrukiton aus Aluminium, welche sich selbst verhakt konstruiert. Sie muss Seitlich ann den Profilen verschraubt werden. Im vorderen Überhang werden Aluminiumelemente eingehängt, welche die Aufhängung der Achse für die Umlenkung erlauben.

Zahnriemenaufgängung

Um die optimale Postiton der Zahnriemenaufhängung für die Y-Achse bestimmen zu können, wird überschlagsmäßig ein Massenschwerpunkt in Z-Richtung berechnet. Um die Konstrukiton beginnen zu können, werden hierfür ungefähre Werte angenommen.

$$X_s = \frac{1}{M} \cdot \sum_{i=0}^{n} z_i \cdot m_i$$

Z_s : Z-Koordinate des Massenschwerpunkts $[m]$
M : Gesamtmasse $[kg]$
z_1 : Z-Koordinate der Teilmasse $[m]$
m_1 : Masse der Teilmasse $[kg]$

Gegenstand	Masse in kg	Position in mm
Motor	0.3	21
Z-Schiene	1	150
Z-Gable	0.3	180

Tabelle 2.1: X-Achse unbeladen und eingefahren

Gegenstand	Masse in kg	Position in mm
Motor	0.3	21
Z-Schiene	1	150
Z-Gabel	1.3	380

Tabelle 2.2: X-Achse beim Ladevorgang

So werden zwei Schwerpunkte errechnet: ca. 130 mm im unbeladen und 250 mm während dem Ladevorgang. Da die Stabilität des Y-Schuttels während dem Ladevorgang wichtiger ist als während einer Leerfahrt wird das Y-Shuttle so positioniert, dass die Aufhängung des Zahnriemen bei rund 200 mm liegt.
Weiters muss auch dieser Zahnriemen wieder gespannt werden, durch die geringere Zahnriemenlänge, wird auch der Spannmechanismus verkleinert. Leider muss dieser aus Platzgründen ausserflucht zum Zahnriemen angebracht werden. Die Folgen dieser Auslenkung auf die Riemenspannung kann durch folgenden Ausdruck, die Länge die der Zahnriemen bei einer bestimmten Position braucht, um Spannungslos zu sein, approximiert werden.(Hierbei wird von einem linearen Zusammenhang zwischen Kraft auf die Aufhängung und Längenkontaktion ausgegangen, in der Realität würde jedoch die Elastizität des Zahnriemens als auch jene der Aufhängung Auswirkungen haben, aber diese verringern die Auswirkungen nur zu unseren Gunsten).

$$K(h) = \sqrt{h^2 + d^2} + (l - h)$$
$$K(h) = \sqrt{h^2 + 28^2} + (1500 - h)$$

$K(h)$: theoretische Länge des Riemen $[mm]$
h : Position des Schlittens $[mm]$
l : Umlenkungsrollenabstand $[mm]$
d : Abstand der Klemme zur Flucht $[mm]$

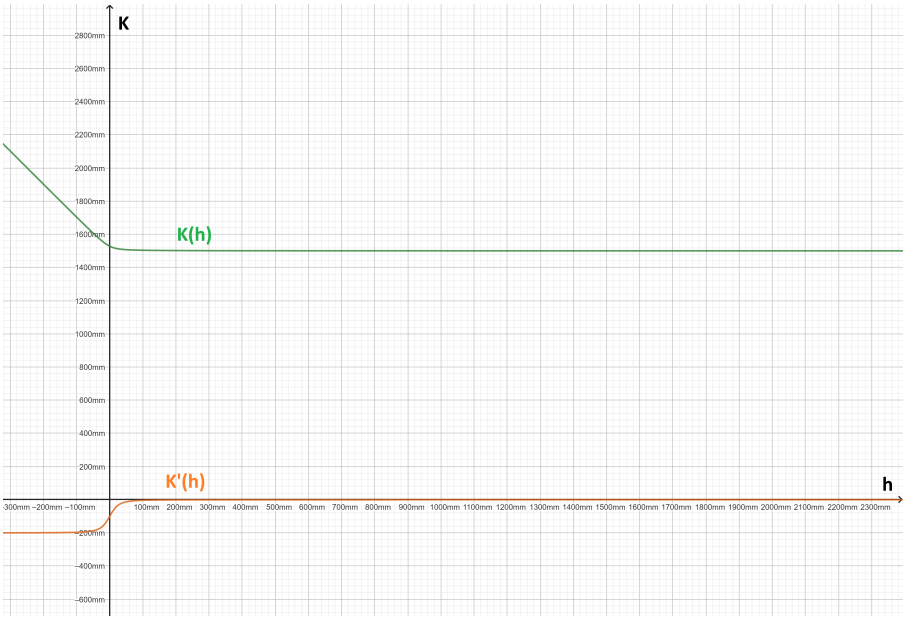


Abbildung 2.4: Zahnriemenkontraktion grafisch dargestellt

Aus der Abbildung 2.4 geht, wie schon intuitiv vermutet, hervor, dass ist diese Assymetrie nur dann ein Problem, wenn der Schlitten weit unten ist. Ein konkretes maß kann folgendermaßen erhalten werden: $K(80) - (750) = 4.24\,\mathrm{mm}$. Der unterschied in der benötigten Riemenlänge zwischen der Mittelstellung (750 mm) und Ruheposition (80 mm über der Zahnscheibe) ist also rund 4 mm. Im anbetracht dessen, dass dies Länge auf zwei Seiten (Hin- und Rücklaufseitig) des Zahnriemen aufgeteilt wird (welcher etwas elastisch ist), sowie etwas biegung in der Umlenkungsaufhängung auftritt, wird diese Extraspannung toleriert.

Shuttleführung

Das YZ-Shuttel wird aus Stabilitätsgründen ebenfalls mit einem V-Slot C-Profil geführt. Dieses sind auf beiden Seiten jeweils oben und unten befestigt. Die Länge wird so gewählt, dass die Verwendung eines 1.5 m langen Profils perfekt ausreicht. Da dass YZ-Shuttel wesentlich weniger Kraft auf die Führung auswirkt können weniger Führungsräder verwendet werden. Die Hauptführungsräder werden so Positioniert, dass sich ein Dreieck ergibt. Dieses dient dazu, dass sowohl beim Aufhebevorgang als auch beim Leerlauf immer eine Klemmung um die Führungsschiene entsteht. Hierbei sind jene V-Wheels, welche beim Aufhebevorgang belastet werden, doppelt ausgeführt. Wie schon bei der X-Achse sind auch hier wieder alle V-Wheel-abstände mit Schrauben einstellbar. Zusätzlich zur Hauptführung sind auch aussen jeweils noch zwei Führungsräder angebracht, um das Shuttel zusätzlich zu stabilisieren.

Sensoren

Da die Endschalter der Y-Achse kapazitiv ausgeführt sind, muss auf dem Shuttel ein Metallisches Gegenstück angebracht sein. Diese stehen uben und unten über und lösen so vor Kolission aus. Um die Sensoren anzubinden muss auch ein ASi-Client auf dem X-Shuttel angebracht werden.

Kabelschleppkette
Um Versorgugng der Z-Achse herstellen zu können, muss eine Kabelschleppkette vom X- zum Y-Shuttel angebracht werden. Ach diese muss leider sonderanforderungen erfüllen. Da die Versorgung von unten ausgeht, muss die Kabelschleppkette stehend eingebaut werden. Dies ist grundsätzlich eine äußerst ungünstige situation, da diese Schlepkette der beschleinugung des X-Shuttels ausgesetzt ist. Um ein Schwingen möglichst zu verhindern werden seitlich noch Führungselemente angebracht. Ausserdem sind die Anschlusellemente fest um die ersten Kettenglieder extra zu unterstützen. [5]

2.1.6 Z-Achse

Als Z-Achse oder Gabel, wird jener Teil des AFSS bezeichnet, der die Boxen in das Lager ein- und ausfährt. Dieser ist in das YZ-Shuttel integriert. Es wird davon ausgegangen, dass um Boxen ein- und auszuheben ca. 210 mm überstand der Gabel benötigt wird. Dies ist also der Mindestverfahrweg der Z-Achse

Linearführung
Geführt wird die Gabel mit zwei Führungsschienen von Igus. Diese bieten Optimale Stabilität sowie, in Verbindung mit einem Führungswagen, eine reibungsarme Bewegung. Wichtig ist jedoch, dass der Schwerpunkt der, am Führungswagenbefestigten Last, nicht mehr als die zweifache Wagenlänge über den Wagen hienausgeht. Danach kommt es zu sehr starker Verklemnmung, und ein Betrieb ist nur mehr schwer möglich. Da, bei einer Auslgeroperation der Schwerpunkt sehr weit übersteht, wird eine TS-01 Führungsschinen und -wagen kombination verwendet. Diese wirkt zwar recht überdimnensioniert doch da der Führungswagen so lang ist, kann ein reibungsarmer Betrieb gewährleiste werden.

Motor
Als Antrieb für die Z-Achse sollen zwei Nema-17 Schrittmotoren verwendet werden. Hier ist es nicht nötig eine closed-loop Steuerung zu verwenden, da bei jedem Hub auch referenziert werden kann. Weiters kann dadurch Kabelschlepkettenplatz gespart werden.

Spindelauslegung
Die Z-Achse wird mit einer Spindel angetrieben. Dies bietet Vorteile in Positionsgenauigkeit und Kraftübertragung. Jedoch ist es wichtig, die richtige Spindelsteigung auszuwählen, um die Ballance zwischen Geschwindigkeit und Kraft zu halten. Ziel ist es, für eine richtung des Hubs maximal 5 Sekunden zu benötigen. Sie wird über eine Zahnscheibe und Zahnrimen mit dem Motor verbinden. Berechnet werden kann diese Zeit mit folgenden Ausdruck:

$$v = \frac{k}{n_{welle}} = \frac{k}{\frac{n_{motor}}{i}}$$

$$F = \frac{M_{mot} \cdot i \cdot k}{n_{welle}} \cdot 2\pi f \cdot \eta$$

M_{mot} : Drehmoment $\left[\frac{N}{m}\right]$

n_{mot} : Drehzahl $\left[\frac{1}{s}\right]$

k : Wellensteigung $\left[\frac{mm}{U}\right]$

i : Übersetzungsverhältniss []

η : Wirkungsgrad der Gewindeshraube []

So wird berechent, dass eine DS10x12 Spindel mit ihren 12 mm Steigung, bei einer Motordrehzahl von 600 U/min (0.42 Nm) und einem Übersetzungsverhältniss von 2:1, ca. 4 Sekunden pro Richtung benötigt und mit einer Kraft von ca. 90 N bewegt wird.
Dies entspricht unseren Anforderungen und somit wird diese Spindel gewählt. Um sie zu Lagern wird vorne und hinten der Durchmesser der Spindel verringert, sodass diese in Kugellager geführt werden können.

Sensorik
Um auch die Endschlater der Z-Achse, sowie weitere Sensoren einzulesen, wird auch hier ein ASi-Slave montiert.

2.1.7 Lager

Das Lager soll die Boxen beeinhalten, und die Möglichkeit zulassen, dass dies von der Gabel ein- und ausgehobenwerden. Weiters muss, die Box in X- und Z-Richtung geführt werden, um Positionsgenauigkeit sicherzustellen, da sonst die Gabel möflicherweise in die Box fährt. Das Lager soll darauf ausgelegt werden, dass Boxen mit den Maßen 50x100x200mm verwendet werden können. Diese Boxen sind nach unten hinn verjüngt und haben oben eine Lippe, an der die Gabel greift.
Umgesetzt wird dies mit eine Gerüst aus 40x40-Item-Profilen, welches im nachhienein in den restlichen Rahmen eingesetzt wird. Auf diese werden 20x40-Profile horizontal Aufgeschraubt, auf diesen stehen dann die Boxen. Vorne und hinten wird eine Kunststoffplatte aufgeschraubt welche leicht überstehen und somit die Box in Z-Richtung positioniert. Zwischen den Boxen wird ein Trennsteg eingebaut, welcher die Boxen in X-Richtung Positioniert sowie den richtigen Abstand zwischen den Boxen einzuhalten.

2.1.8 Querförderer

Da es dem Portalroboter nicht möglich ist, die Boxen dierekt auf das Förderband zu legen, muss hier noch ein System eingebunen werden, welches dies erledigt. Da die Boxen beim ein- und auslagern den gleichen Weg zurücklegen, muss dieser Querförderer, die Box sowohl auf das Förderband, sowie vom Förderband herunterbewegen kann. Zu diesem Zweck

wir eine weitere, der Gabel ähnlichen Konstruktion montiert, welche auf einem 20x40-V-Slot-Profil verläuft. Die Box wird dann hin- und her geschoben, um vom Lager auf das Fördeband umzuladen. Dadurch ist es noch möglich, dass die Gabel der Z-Achse die Box in der Nulstellung ein- und aushebt.

2.1.9 Fertigung der Einzelteile

Lasern
An der HTL-Mössingerstraße ist es den SchülerInnen möglich einen Lasercutter zur Kunstoffverarbeitung zu verwenden. Um ein gewünstes Teil Ferigen zu können, muss die Kontour dieses als .dwg zur verfügung stehen. Dieses kann dann unter berücksichtigung der Materialstärke aus verschiedenen Farben geschnitten werden.

Fräsen
Um Aluminiumteile zu fertigen steht eine 3-Achsen CNC-Fräse zur Verfügung. In dieser ist es möglich die Teile zu fräsen, die aufgrund ihrer hohen mechanischen Beanspruchung nicht aus Kunststoff gefertigt werden können, aus anderen Materialien herzustellen. Um dies zu bewerkstelligen, muss zuerst der G-Code in Filou-NC16 programmiert werden und kann dann in NC-Easy auf der CNC-Fräse ausgeführt weren. Da der Fräser jedoch einen größeren Durchnesser als der Laser hat, muss, wenn Ecken benötigt werden, eine Aussparung größer dem Durchmesser des Fräsers, eingeplant werden.
Als Aluminiumlegierung wird hier die Legierung EN-AW 5754 (AlMg3) verwendet. Diese Legierung aus Aluminium und Magnesium eignet sich sehr gut zum Fräßen und ist in der Lage, die mechanischen Beanspruchungen auszuhalten.

Aluminium-Extrusionen schneiden
Um Aluminiumextrusionen abzulängen wird eine eigens dafür ausgelegte Kreissäge verwendet. In dieser ist es möglich einen Anschlag für eine bestimmte länge einzustellen und dann abzuschneiden. Dadurch ist es möglich in kurzer Zeit viele verschiedene längen zuzuschneiden.

Umlenkrollen
Die Umlenkrollen sind jeweils am Ende der X- und Y-Achsen angebracht. Dadurch dass diese der gesamten Spannkraft ausgesetzt. Dies erfordert spezielle Anforderungen an die Aufhängen als auch an die Umlenkrolle selbst. Diese soll primär eine 180° Wende des Zahnriemens ermöglichen, sowie sekundär eine Führung für den Riemen bieten. Umgesetzt wird diese Anforderungen, durch Fertigung von vier Aluminium-Drehteilen in welche Kugellager eingepresst werden.

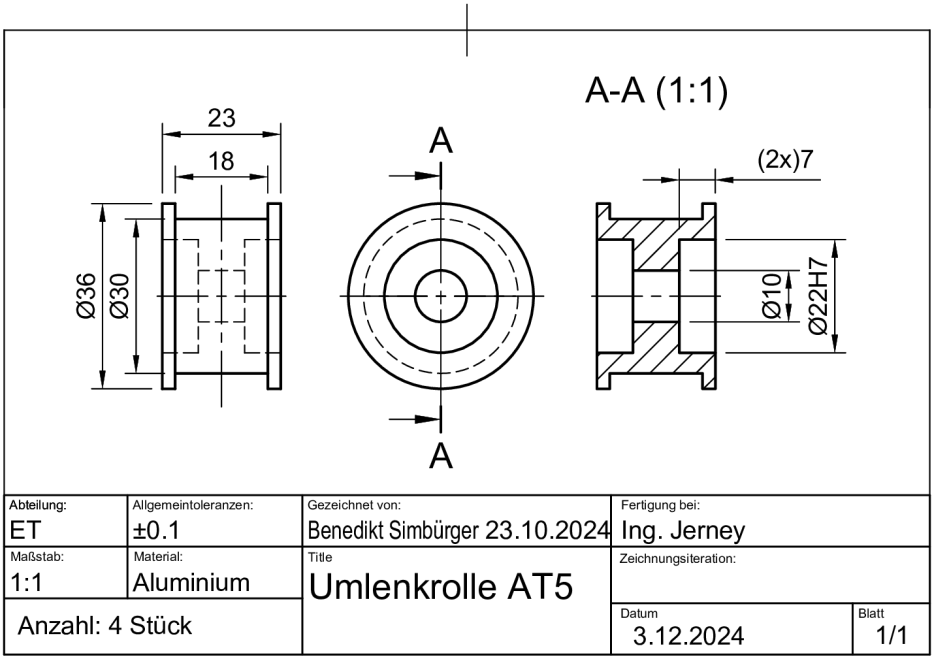


Abbildung 2.5: Bauteilzeichnung Umlenkrolle

Die Fertigung dieses Teils nach Abb. 2.5 lässt sich in folgende Teilschritte unterteilen:

- Zuerst die Frontfläche Plandrehen (Drehzahl: 900 U/min)
- Ungefär 30mm Länge auf das Aussenmaß von 36 mm Plandrehen
- Mit 9.8mm Bohrer das mittlere Loch vorbohren (540 U/min)
- Mit 10mm Reibeisen und viel Öl das Loch auf eine genaue Passung bringen (260 U/min)
- Die Position relativ zum Backenfutter markieren, um beim Neu-Einspannen Rundlaufgenauigkeit zu gewährleisten
- Zylinder bei ca. 26 mm Abstechen (540 U/min)
- Umspannen und auf Maß Plandrehen (900 U/min)
- Die Aussparung für die Lager mit Eckdrehmeißel beginnen, jedoch nach innen hin nur 6.8 mm
- Bei ca. 17 mm Lochdurchmesser den tatsächlichen Durchmesser mit der Digitalanzeige vergleichen und gegebenenfalls korrigieren
- Bei 21.5 mm den Oberschlitten die restlichen 0.2 mm zustellen und die gesamte Tiefe Plandrehen
- Den Lochdurchmesser auf 21.95 erweitern und dann in kleinen Inkrementen zustellen, bis das Lager gerade so nicht passt, um einen Presssitz zu gewährleisten. Dies tritt bei Lagern mit 22 mm Außendurchmesser bei rund 22.04 mm ein.

- Da für die Einsparung der Riemenführungsfläche kein Angriffspunkt verfügbar ist, wurde als Halterung ein Dorn nach Abb. 5.1a gedreht, auf welchen das Drehteil aufgeschraubt wird.
- Mit dem Abstechmeißel wird in 2 mm Inkrementen die Zahnriemenauflagefläche herausgedreht, bis auf 30.2 mm, sowie links und rechts den Rand 1 mm extra dick lassen (siehe Abb. 5.1b). (540 U/min)
- Am Schluss wird die Rand Tiefe auf Maß gedreht.
- Als letzten Schritt werden links und rechts die zwei Lager eingepresst.

Durch die verhältnismäßig großen Toleranzen bei den Lageraussendurchmessern wird bei 2 der 8 Lagerpassungen zusätztlich Lagerkleber verwendet um eine zuverlässigen Halt zu gewährleisten, da bei diesen die Toleranzen nicht eingehalten wurden.
Nach diesem Vorgang sind die vier umlenkrollen fertig (siehe 5.1c) und können auf einem 8 mm Schaft montiert werden.

(a) Dorn (b) Fertigung der Lauffläche (c) Fertiggestellte Umlenkrollen

Abbildung 2.6: Umsetzung der Umlenkrollen

Distanzhülsen

Bei den mesiten V-Wheels werden Distanzhülsen zur Klemmung benötigt. Dies müssen ein sehr spezielles Maß (13.4 mm Länge) haben. Deshalb müssen Hülsen mit einer Länge von 20 mm auf diese Länge heruntergedreht werden. Dies wird mit dem rechten Eckdrehmeißel bei einer Drehzahl von 740 U/min ausgeführt. Da relativ viele solcher Teile benötigt werden, wird beim Einspannen der Drehmeißel selbst als Endstopp verwendet, um ein relativ wiederhohlgenaues Maß zu erhalten sowie eine simple Durchführung zu erlauben. Es wird also der Meißel auf Position gefahren und das Werkstück eingespannt, so, dass es am Drehmeißel ansteht. Nun wird der Drehmeißel zurückgefahren und die Hülse kann einfach durch mehrere Plandrehoperationen, bis der Schlitten ansteht, gekürzt werden. Es muss also weder gemessen, noch die Digitalanzeige verändert werden, um schnell mehrere Teile hintereinander zu fertigen.

Zahnscheiben
Da die Welle des Motors recht kurz ist, als auch die Motoraufhängung Platz wegnimmt, ist es erfordelich die Zahnscheibe unkonventionell mit dem Motor zu verbinden. Es werden also in der Zahnrimen-Kontaktfläche zwei Löcher gebohrt, angesenkt und mit M4 Gewinde versehen, um dort 2 M4 Wurmschrauben einzuschrauben. Bei der Länge der Wurmschrauben muss darauf geachtet werden, dass sie, im montierten Zustand, sich komplett unter der Oberfläche befindet, um den Zahnriemen nicht zu beschädigen.

2.1.10 Aufbau

2.2 Software und Benutzeroberfläche

2.2.1 Grundlegendes

Um dem Endnutzer die möglichkeit zu geben, das AFSS möglichst einfach zu bedienen, sowie die komplexen Logik zu Lagersteuerung auszuführen, bedarf es eines Servers (Backend) und einer graphischen Benutzeroberfläche (GUI oder Frontend). Diese müssen eine Vielzahl an Verschiedenen Funktionen beinhalten.

2.2.2 Aufbau

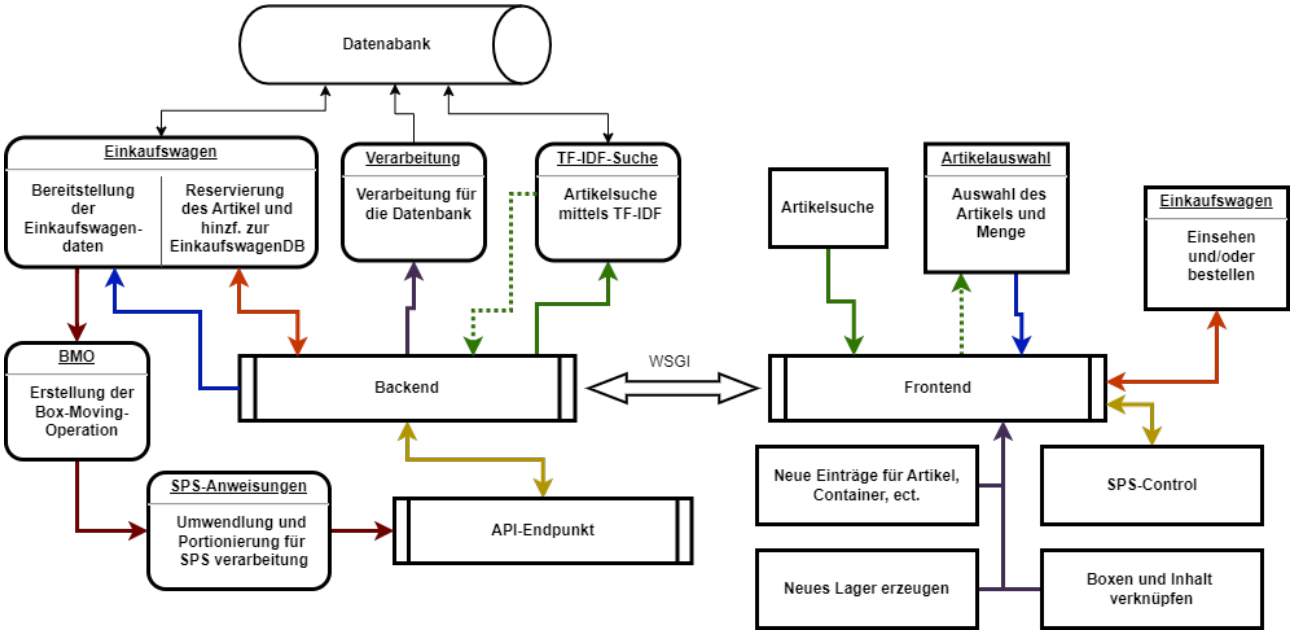


Abbildung 2.7: Gesamtüberblick des Servers

Um diesen Anforderungen gerecht zu werden, muss eine Lösung mit sehr hohem grad an Freiheit in der Logik sowie UI gestaltung gewählt werden. Weiters muss es möglich sein, dass zukünftige Schülerinnen und Schüler diesen instandhalten und erweitern. Aus diesen Gründen, sowie den bereits vorhandenen Kentnissen, wird die Programmiersprache 'Python' als Grundlage des Servers verwendet.

Python

Python ist eine vielseitige und hochentwickelte Programmiersprache, die für ihre Einfachheit und Lesbarkeit bekannt ist und sich sowohl für Anfänger als auch für Fortgeschrittene eignet. Sie unterstützt mehrere Programmierphilosophien, darunter objektorientierte und funktionale Programmierung, und wird in Bereichen wie Webentwicklung, Datenanalyse, künstliche Intelligenz und wissenschaftlichem Rechnen häufig eingesetzt. [6]

Flask
Flask ist ein leichtgewichtiges Web-Framework für Python, das durch seine Einfachheit und Flexibilität hervorsticht und sich ideal für kleinere Anwendungen oder Prototypen eignet. Es folgt einem minimalistischen Ansatz, bietet aber Erweiterungsmöglichkeiten, um komplexere Projekte zu realisieren. [6]

Python - Virtuelle Umgebung
Dieses Projekt enthält sehr viele externe Bibliotheken. In Python sind diese Bibliotheken mit dem Interpreter verknüpft, da bei Installation diese externen Bibliotheken dierekt beim Interpreter gespeichert werden. So kommt es allerdings dazu, dass wenn das Programm auf einer anderen Maschine ausgeführt wird, diese Bibliotheken nicht vorhanden sind. Um Abhängigkeitskonflikte und Portabilitätsprobleme zu verringern, werden Virtuelle Umgebungen verwendet. Diese enthalten den Interpreter sowie die Bibliotheken und könnten einfach auf eine andere Maschine kopiert werden.
Dies ist jedoch bei diesem Projekt nur bei der Entwicklung von nöten, da es bei Fertigsteilung im Docker Container ausgeführt wird.

2.2.3 Benutzeroberfläche

Um die Benutzeroberfläche zu realisieren muss eine Weboberflche erstellt werden. Auf dieser werden alle Inhalte angezeigt die für die Benutzung nötig sind. Sie wird vom Server zu verfügung gestellt, sobalt dieser eine http-Anfrage erhält. Um diese mit eigenen Inhalten und Funktionen zu befüllen, muss dies mit HTML geschehen.

HTML
HTML (HyperText Markup Language) ist die Standard-Auszeichnungssprache zur Strukturierung und Darstellung von Inhalten im Web. Sie definiert die grundlegende Struktur einer Webseite mit Elementen wie Überschriften, Absätzen, Links, Bildern und Formularen. [6]

Dadurch, dass sich viele Elemente des UI wiederholen, wie z.B. Navigationsleiste, bietet Falsk die Möglichkeit sog. 'Templates' zu verwenden. Diese können einmal definiert werden und dann an mehreren teilen der Webseite verwendet werden. Um elemente wie Formularflöchen oder Datenanzeige, einfach mit den benötigten Daten anzuzeigen, gibt die möglichkeit Macros zu erstellen, welche von Flask mit den bestimmten Daten vorgerendert und in das restliche HTML eingefügt werden. Um HTML, welches grundsätzlich ohne Formatierung auskommt, zu stylen, muss CSS verwendet werden.

CSS
CSS (Cascading Style Sheets) ist eine Stylesheet-Sprache, die verwendet wird, um das Design und Layout von Webseiten zu gestalten. Sie ermöglicht die Trennung von Inhalt und

Darstellung, indem sie Farben, Schriftarten, Abstände und andere visuelle Aspekte definiert.[6]

Da auch Logik in der Webseite verbaut werden muss, muss zusätzlich auch Javascript verwendet werden, da HTML und CSS alleine, noch nicht gut genug mit dem Server Kommunizieren können.

JavaScript
JavaScript (JS) ist eine vielseitige Programmiersprache, die hauptsächlich verwendet wird, um interaktive und dynamische Elemente auf Webseiten zu erstellen. Sie läuft direkt im Browser und ermöglicht Funktionen wie Animationen, Formularvalidierungen und die Kommunikation mit Servern in Echtzeit.[6]

Praktisch geschieht diese Server-Kommunkation immer mithilfe diese Programmblocks:

```
    function sendData(data, callback) {
        var xhr = new XMLHttpRequest();
        var url = "{{url_for('main.add_stock')}}"; //Flask markup, um die
    richtige url zu erreichen, dies wird vor ausgabe auf der Webseite noch
    eingesetzt

        xhr.open("POST", url, true);
        xhr.setRequestHeader("Content-Type", "application/json");

        xhr.onreadystatechange = function () {
            if (xhr.readyState === 4 && xhr.status === 200) {
                callback(xhr.responseText)  //die funktion wir aufgerufen
            }
        };
        var jsonData = JSON.stringify(data);
        xhr.send(jsonData);
    }
```

Dieser ermöglicht die Übergabe von Daten im JSON format, und einer Funktion, die die zurückgeschickten Daten verarbeitet. In der Praxis wird dieser so aufgerufen:

```
function add_to_db(){
    sendData({"add_stock": {"barcode": barcode, "quantity": quant, "article"
    : article}},
    set_gen_stock)
}

function set_gen_stock(req){
    document.getElementById("generated").innerHTML = req
}
```

Wie im Quellcode ersichtlich, werden Daten aus der Webseite ausgelesen, in JSON konvertiert. Danach werden diese Daten zusammen mit einer Funktion, an 'send_Data'. Wie bereits

erwähnt, gibt der Server dann Daten zurück. In diesm Fall, werden dann Datten aus der DB vorformattiert. Diese werden dann in der zuvor übergebenen Funktion, in der Webseite eingefügt.

2.2.4 Backend

Das Backend des Servers ist für die Datenverarbeitung verantwortlich. Es ist wie bereits erwähnt in Python geschrieben und stellt mit dem Flask-Framework die Benutzeroberfläche zur Verfügung.
Aufgebaut ist es in mehrere Bereiche. Einerseits die Webanwendung, sowie API (Application-Prgramming-Interface, Schnittstelle zwischen Anwendungen) als auch die Anbindung an die Datenbank sowie Verarbeitung der SPS-Befehle und API zugriff auf dies.

Serverseite der Benutzeroberfläche
In Flask könne 'bleuprints' definiert werden. Dies sind Webseitelemente die einen Bestimmten URL-Vorsatz haben. So werden anfanges 'blueprints' für Hauptfunktionen ('/', also ohne Vorsatz), Datenbankinteraktionen ('/db_interactions') usw. definiert. Die Funktionen dafür werden dann in jeweils eigenen Dateien geschrieben. Dies ermöglicht eine weit aus bessere Übersicht bei großen Projekten.
Eine Funktion, die für die Verarbeitung der Anfragen einer bestimmten URL verantwortlich ist, sieht immer ähnlich aus.

```
1     @main.route("/add_stock", methods=["GET", "POST"])
2     def add_stock():
3         if request.method == "POST":
4             if request.data:
5                 req = request.get_json()
6
7                 if "add_stock" in req.keys():
8                     dt = req["add_stock"]
9                     new = Stock(
10                        container=db.session.query(Container)
11                        .filter_by(barcode=dt["barcode"])
12                        .first()
13                        .id,
14                        article=dt["article"],
15                        quantity=dt["quantity"],
16                    )
17                    db.session.add(new)
18                    db.session.commit()
19                    return "Sucsess"
20
21        return render_template("add_stock.html")
```

Anfangs wird mit einem Decorator (@main.route(...)) die gewünschte URL, sowie unterstützte http-Requests definiert. Decoratoren verändern oder erweitern die Eigenschaften von Funk-

tionen. In diesem Fall wird in der Funktion (def ...()) definiert was geschieht, wenn ein erlaubter Request an der URL 'add_stock' eintrifft. Dieser Funktionsname kann auch in den Flask-Vorlagen verwendet werden um URLs dynamisch zu vergeben.
Weiters wird sortiert um welche art von Art von Anfrage es sich handelt. Bei GET-Anfragen wird typischerweise einfach nur das HTML der Webseite zurückgegeben. Bei POST-Anfragen werden zuerst die Daten dieser Anfrage eytrahiert und dann entschieden was damit gemacht werden soll. In diesem Fall wird, wenn des richtige Schlüsselwort in der Anfrage enthalten ist, ein Datenbankeintrag, mit den Daten aus dem Request, hinzugefügt. Schlussendlich wird ein Wert zurückgegeben, entweder ein HTML-Statuscode, vorgerendertes HTML oder, wie in diesem Fall, ein Text.

2.2.5 APIs

SPS - Verbindung

Die Daten für die SPS werden in einem Ähnlichen vefahren zu verfügung gestellt. Nun schickt nicht die Benutzeroberfläche oder der Browser eine Anfrage an das Backend, sondern die SPS. Der Programmblock zur Verarbeitung dieser Anfrage sieht folgenderaßen aus:

```
@api.route("/afss", methods=["GET", "POST"])
def afss():
    request_data = {...} #aus platzgruenden verkuerzt
    if request.method == "POST":
        ... # Abfrage ob der Sender eine Siemens SPS ist
        if "next_bmos" in req.keys():
            if req["next_bmos"] == "":
                return "400"

            if is_SPS:
                return convert_instruction_for_PLC(afss_stack.
    get_current_bmos(int(req["next_bmos"])))

            return jsonify(afss_stack.get_current_bmos(
                int(req["next_bmos"])))

        if "inst_acknowledge" in req.keys():
            if is_SPS:
                ack = afss_stack.inst_acknowledge(
                    int(req["inst_acknowledge"]))
                if ack == "204":
                    return ack
                return convert_instruction_for_PLC(ack)
```

Die Funktion dieses Codeblocks besteht darin, herrauszufiltern, ob eine Anfrage einer Siemens-SPS eintrifft und dann dementsprechend Daten zurückzugeben. Wenn eine Anfrage mit dem Schlüssel 'next_bmos' (next-box-moveing-operation) empfangen wird, wird diese dem sog. 'stack' weitergegeben, welcher diese dann verarbeitet (dies wird im Anschluss näher behandelt). Sollte diese von einer SPS kommen, werden die Zurückgeschicten Daten vorher noch

so bearbeitet, dass diese von der SPS möglichst gut übersetzt werden können. Sollte die SPS ein ’inst_acknowledge’ sowie die dementsprechenden Daten, wird im ’Stack’ vermerkt dass die dementsprechende Operation bei der SPS angekommen ist.
Weiters bietet Simens auch die Möglichkeit über den Webserver der CPU auf diese zuzugreifen. Dies geschieht über das JSON-RPC Protokoll [7]. Aus Testzwecken ist des durchaus nützlcih auh dierekten Zugriff auf die CPU zu haben, wenngleich es für den Normalbetrieb nicht zwingend benötigt wird. Um eine möglichst einfache bedienung zu Ermöglichen wird ein Objekt angelegt, um JSON-RPC Befehle auszuführen.

- Login: Um die SPS zugreifen zu können muss erst ein Login geschehen. Hierbei werden Benutzername und Passwort benötigt, die zuerst in der SPS festgelegt wurden. Dieser führt bei Erfolg zu einem ’Session-Token’, welches bei jeder folgenden Kommunikation zur Authentifizierung mitgeschickt werden muss.
- Daten Lesen: Die Methode ’PlcProgram.Read’ gibt den Wert einer gegebenen Variable zurück.
- Daten Schreiben: Mit ’PlcProgram.Write’ kan der Wert einer oder mehrere Variablen geändert werden.

Diese Objekt kann dann einfach in einen Anderen Programmteil importiert werden, und dort dann die Verbindung zu einer SPS herstellen. Theoretsich könnt man auch mehrere dieser Objekte anlegen, um Zugriff mehreren Steuerungen umzusetzen.

2.2.6 Lageralgorithmus für SPS

Die Auftrage sind im Einkaufswagen o.ä. im Format ’Location - Location’ hinterlegt. Um dies in eine Opeation zu zerlegen die die SPS versteht, wird eine Box-Moveing-Operation erstellt. Diese enthält alle Teilschritte, wie Schlittenposition oder Fördebandstracke, um eine Box von Position A zu Position B zu bringen. Die BMO wird dann in die einzelnen verwendeten Module, Förderband oder Lager, aufgeteilt und dann dem ’stack’ zugeführt. Diese Instruktion enthält Daten wie:

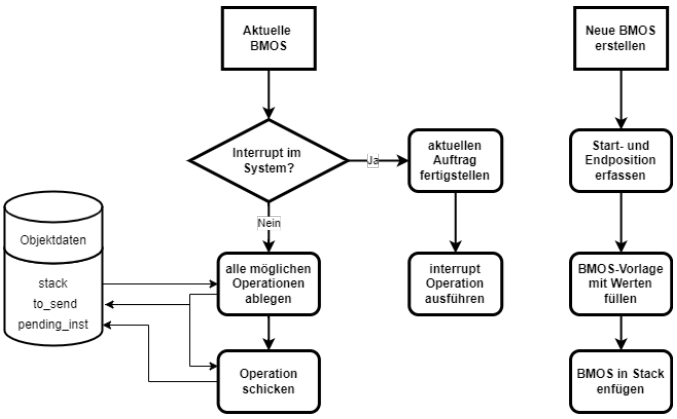


Abbildung 2.8: Diagramm der BMOS bereitstellung

- ’instruction_id’: Eine fortlaufende Nummer um jede Anweisung zu identifizieren

- 'order_id': Zusammenhängende Anweisungen, bzw. Anweisungen für die gleiche Box, haben eine gleiche Nummer
- 'relation': eine Liste mit 'instruction_id's', welch vor dieser Anweisung erfüllt sein müssen

Bei Auswahl der zu schickenden Anweisung, wird zuerst die aktuell von der SPS gemachte Anweisung vermerkt, danach weird in allen Modulen des 'stacks' nachgeschaut, welche Anweisungen jetzt ausgeführt werden können. Zu diesem Zweck wird die 'relation' einer Anweisung mit den bereits gemachten Anweisungen verglichen, wenn bereits alle relevanten Anweisungen gemacht wurden, wird der Auftrag in 'to_send' vermerkt. Schlussendlich wird einer der in 'to_send' vermerkten Aufträge für die SPS übersetzt und an diese geschickt. Wenn die SPS den Auftrag erhalten hat, schickt sie ein 'acknowledge' mit der erhaltenen ID zurück. Der Auftrag wird dann aus den 'instructions_to_send' gelöscht und entweder wird einfach der nächste Auftrag in 'to_send' geschickt oder der 'BMOS' Algorithmus wird erneut durchlaufen.

2.2.7 Datenbanken

Als Datenbanksystem wird aufgrund des guten Supports MySQL gewählt. Dies ist ein relationales Datenbankmanagementsystem welches in einem Docker-Container aufgesetzt wird. In diesem werden alle Daten gespeichert, die zur Auswahl sowie zur Ausliferung von Teilen nötig sind.

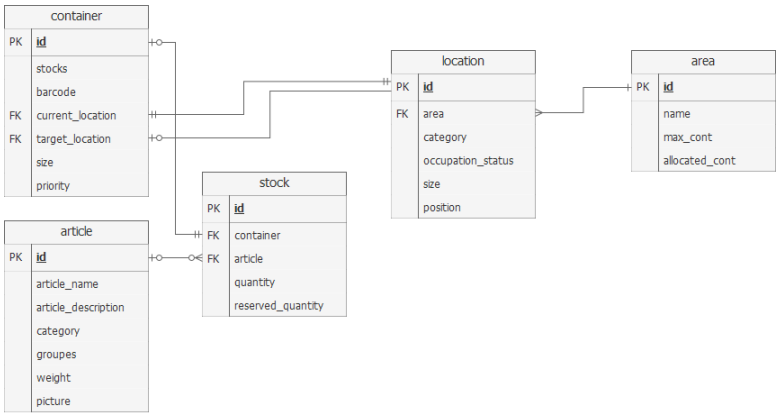


Abbildung 2.9: Datenbankschema des AFSS

Wie in 2.9 ersichtlich, beinhalt diese Datenbank fünf Tabellen. Diese hohe Komplexität resultiert daraus, dass diese Struktur eine 100%ige Flexibilität in der Ablage von Bauteilen in einem überliegendem System bietet.
Die Erste Tabelle beschreibt ein einziges theoretisches Bauteil. Dieses hat einen Namen, Gewicht, Beschreibung und Kategorien zur Filterung. Unter der Spalte 'picture' wird ein Dateiname gespeichert, der zu einem Bild zeigt, dass das Produkt abbildet. Die zweite beschreibt einen Container. Im Lagersystem entspricht dieser einer Box. Diese kann mehrere 'stocks' beinhalten, sowie durch einen Barcode identifiziert werden. Weiters muss jeder Container immer eine aktuelle Position ('current_location') besitzen, an der die Box gerade ist.

Im ausgelagerten (und noch nicht eingelagerten) Zustand ist diese ‘location’ Position 0. Das Ziel der Box wird in ‘target_location’ gespeichert. Stimmt die aktuelle mit der Zielposition überein, so ist die Box an ihrem Ziel angelangt. Die Kategorie ‘size’ beschreibt die Größe eines Containers und lässt somit theoretisch zu, dass in Zukunft auch unterschiedlich große Boxen zuverlässig in die richtigen Lagerplätze eingelagert werden. ‘priority’ wird nicht verwendet.
Container und Artikel werden im sog. ’stock’ verheiratet. Dieser kann als bauteilhaufen in einer Box verstanden werden. Es können also auch mehrere ’stocks’ mit dem selben Container geben, dies würde mehreren verschiedenen Bauteilen in einem einzigen Copntainer entsprechen. Auch ist es möglich mehrere Container mit den selben ’stocks’ abzubilden, welches ener aufteilung von Abuteilen auf mehrere Container entspräche.
Die Positionen der Container werden in Standorte (’locations’) abgebildet. Diese entsprechen den Lagerplätzen. Sie sind einer darüberliegenden ’area’ zugeordnet, welche einerseits einen Lagerschrank, aber weiters auch Module wie Vereinzelungsanalgen, abbilden kann. Standorte verfüghen weiters über eine Position welche in X, Y und Z-Richtung beschreibt, wo sich ein Standort im Referenzsystem des Lagers befindet. Auch die größe des Lagerplatzes wird abgebildet, um sicherzustellen, dass auf jeden fall nur die richtige Größe an Box eingelagert wird.

2.2.8 Docker

Abbildung 2.10: Docker-Logo: [8]

Docker ist eine Umgebung, in der Softwareprojekte isoliert werden können. Da besonders auch bei Projekten mit großem Python anteil, viele Pakete mit verschiedenen Versoinen benötigt werden, ist es sehr hilfreich diese zu bündeln.
Umgesetzt wird dies mithilfe von Containern welche einen gesamten Programmteil als alleinstehende Einheit enthält. Diese werden über ein ’Dockerfile’ configuriert, welches sich im selben Ordner wie die Python-Anwendung befindet. In diesem werden Parameter wie die Python-Version und die benötigten pip-Pakete sowie den Programmeinstiegspunkt angegeben.
Ein zweier Docker Conteianer wird mit einem MySQL-Image erstellt, dort wird die Datenbank aufgesetzt.
Um diese Zwei Container miteinander Kommunizieren zu lasen, ist es ntig ein sog. docker-compose.yml File zu erstellen. Dies enthält alle Informationen über verwendete container, deren Ports, sowie Speicher für Dateien (Volumes). Bei Testbetrieb wird der Datenbankcontainer aleinstehend bestrieben und mit einem anderen Port, keine Zugriffsprobleme zu generieren. In Produktion wird dann derselbe Container in den Containerverband übertragen und dort mit einem anderen Port weiterverwendet.
Erstellt wird dieser Containerverband mit den Consolenbefehlt der auf das docker-compose File zugreift.

```
docker build docker-compose.yml
```

Dann werden auch alle Logs in der Kommandozeile ausgegeben sowie

2.2.9 Artikelsuche

TF-IDF und Rust Implementierung

Der TF-IDF (Term Frequency-Inverse Document Frequency) Algorithmus, ist ein Weg um wichtige wörter aus Dokumenten zu extrahieren. Er wird verwendet um beispielsweise in Suchmaschienen, eine Suchanfrage mit Webpagecontent abzugleichen, und die am besten mit der Suchanfrage übereinsteimmenden Dokumente zu sortieren.
Im Fall dieser Anwendung werden die Daten aus der Artikeldatenbank als 'Dokumente' angesehen und die Suchanfrage aus dem Suchfeld wird dafür verwendet um die am besten passenden Artikel zu finden.
Durch den Relativ hohen Rechenaufwand bei dieser Suchoperation wird dieser in der Programmiersprache Rust implementert. Die Implementierung in Rust ist im vergleich zu Python schon bei relativ kleinen Datenmenge bis zu 5-mal schneller.

Rust

Rust ist eine sehr effiziente und schnelle Programmiersprache die in den späten 2000er und frühen 2010ern bei Mozilla und der Open-Source-Community entwickelt. Sie unterstützt unter anderem mehr Typensicherheit und verhindert viele Programmierfehler.[6]
Die Funktion dieses Algorithmus ist in drei unterteile Unterteilt.

1. Term Frequenz
   Die Termfrequenz gibt an, wie oft ein angegebenes Wort in einem Dokument vorhanen ist. Dies wird durch die Folgende Funktion kalkuliert.

```rust
fn term_frequency(document: &str, term: &str) -> f64 {
    // Store the lowercase document as a String to ensure it lives long enough
    let lower_document = document.to_lowercase();

    // Split the document into words
    let normalize_document: Vec<&str> = lower_document.split_whitespace().collect();
    // Make sure the searchterm is lowercase
    let normalize_term = term.to_lowercase();

    // Count occurrences of the term in the document
    let count = normalize_document
        .iter()
        .filter(|&&word| word == normalize_term) // Compare each word with the term
        .count();

```

```
16      // Calculate the term frequency as occurrences / total number of
    words
17      let total_words = normalize_document.len();
18      if total_words == 0 {
19          0.0 // Avoid division by zero if the document is empty
20      } else {
21          count as f64 / total_words as f64
22      }
23  }
```

Mithilfe dieser wird eine Liste aller Wörter und der Vorkommenshäufigkeit dieser erstellt.

2. Die zweite Komponente ist dann die Inverse Dokument Frequenz. Diese gewichtet, die Anzahl der Dokumente in dem das gesuchte Wort enthalten ist relativ zur Gesamtdokumentanzahl vorkommt. Häufig vorkommende Worte wie z.B. 'und' werden hierbei weniger gewichtet als einzigartige Wörter.

```
1  fn inverse_document_frequency(term: &str, all_documents: &Vec<String>)
       -> f64 {
2      let mut num_documents_with_this_term = 0;
3
4      // Iterate over all documents to check if they contain the term
5      for doc in all_documents {
6          // Normalize both term and document by converting them to
       lowercase
7          let lower_doc = doc.to_lowercase();
8          let normalized_doc: Vec<&str> = lower_doc.split_whitespace().
       collect();
9
10         // Check if the term exists in the document
11         if normalized_doc.contains(&term.to_lowercase().as_str()) {
12             num_documents_with_this_term += 1;
13         }
14     }
15
16     // Calculate IDF
17     if num_documents_with_this_term > 0 {
18         // Apply the IDF formula: 1 + log(total_documents /
       documents_with_term)
19         1.0 + ((all_documents.len() as f64) / (
       num_documents_with_this_term as f64)).ln()
20     } else {
21         // If the term is not found in any document, return 1.0
22         1.0
23     }
24 }
```

3. Nun liegt Liste davon vor, wie oft ein Wort in den Suchdaten vorkommt, als auch, wie oft ein Suchbegriff in einem bestimmten Dokument ist.

Als nächsten Schritt werden diese beiden Werte für jeden Suchbegriff miteinander multipliziert und ergeben somit einen Vektor der die Suchwörter in Relation zu jedem einzelnen Dokument stellt.

4. Als letzten Schritt wird der zuvor errechnete Dokumentenvektor (der IDF jedes Suchterms in jedem Dokument) mit dem Suchvektor verglichen. Die geschieht mit der sog. Kosinus-Ähnlichkeit.

```rust
fn cos_similarity(query_p: Vec<f64>, document_p: Vec<f64>) -> f64 {
    // Ensure that both vectors have the same length
    if query_p.len() != document_p.len() {
        return -1.0;
    }

    let mut dot_product = 0.0;
    let mut abs_doc_squared = 0.0;
    let mut abs_query_squared = 0.0;

    // Calculate the dot product and the magnitudes (squared)
    for i in 0..query_p.len() {
        dot_product += query_p[i] * document_p[i];
        abs_doc_squared += document_p[i].powi(2); // document_p[x] ** 2
        abs_query_squared += query_p[i].powi(2); // query_p[x] ** 2
    }

    // Calculate the magnitudes
    let abs_doc = abs_doc_squared.sqrt();
    let abs_query = abs_query_squared.sqrt();

    // Handle division by zero in case of zero vectors
    if abs_doc == 0.0 || abs_query == 0.0 {
        return 0.0;
    }

    // Return the cosine similarity
    return dot_product / (abs_doc * abs_query);
}
```

Nach der Berechnung dieser für jedes Dokument werden alle Dokumente sortiert und ja nach anforderung die benötigte Anzahl ausgegeben.

Artikelsuche nach Kategorien

Um auche einen simpleren weg der Artikelfindung zur verfügung zu stellen, wird ausserdem die Möglichkeit implementiert, Artikel anhand von Attributen zu suchen. Zu diesem Zweck werden schon bei der Artikelerstellung Attribute für die Einträge 'Gruppen' und 'Kategorien' vergeben. In 'Gruppen' wird eine art Pfad angelegt, der die Suche eingrenzt. Beispielsweise würde so ein Eintrag folgende Daten Einthalten: [Ïtem", "Verbindungssatz"] enthalten. So kann bei der Artikelsuche zuerst die Überkategorie Ïtemünd dann die Unterkategorie

”Verbindungssatzäus mehreren verschiedenen ausgewählt werden. Um Bauteile weiter zu unterschieden, da es z.B. viele verschiedene Widerstände gibt, weren unter ’Kategorien’ einzelheiten zum Produkt, wie Wert, Farbe o. ä. gespeichert.
Diese Informationen werden auch vom TF-IDF verwendet, dienen aber spezieller dazu, möglichst sicher das gewünschte Bauteil in einem Durchcklickmenue zu finden.

3 Integration und Programmierung der Steuerungstechnik (Vincent Sonvilla)

3.1 Aufgabenstellung

3.2 Tia-Portal

Allgemeines blabla wieso wirds verwendet oder warum –¿ vllt weil mix anderes verfügbar ist

3.2.1 Grundlagen

was muss man wissen um des Programm bzw. die Software bedienen zu können

3.2.2 Programmiersprachen

3.2.3 Libraryeinbindung

3.3 Motorenansteuerung

3.4 SPS-Server Kommunikation

3.4.1 Zur Auswahl stehende Kommunikationsprotokolle

Kommunikationsprotokolle ermöglichen den Datenaustausch zwischen unterschiedlichen Systemen, indem sie Standards und Regeln für die Kommunikation definieren. Diese Protokolle werden üblicherweise der 7. Schicht (Anwendungsschicht) des OSI-Modells zugeordnet. In diesem Projekt wurden zwei Protokolle getestet und miteinander verglichen: OPC-UA und das HTTP-Protokoll.

Allgemeines

HTTP (Hypertext Transfer Protocol):
HTTP ist eines der bekanntesten Protokolle, welches für die Datenübertragung zwischen Clients und Servern verwendet wird. Es basiert auf einem Anforderungs-Antwort-Prinzip, bei dem ein Client (Bsp.: Webbrowser) Anfragen an einen Server sendet, welcher anschließend die entsprechenden Daten zurückschickt. Die Anfrage wird als HTTP Request und die Antwort als HTTP Response bezeichnet.

OPC-UA (Open Platform Communications - Unified Architecture):
OPC-UA (Open Platform Communications Unified Architecture) ist ein plattformunabhängiges Kommunikationsprotokoll, das speziell für industrielle Anwendungen entwickelt wurde. Es ermöglicht eine herstellerunabhängige Kommunikation zwischen verschiedenen Geräten bzw. Systemen.

Funktionsweise

HTTP (Hypertext Transfer Protocol):

3.4.2 Verbindungsherstellung

Aus den in Punkt 3.4.1 genannten Gründen wurde das HTTP Protokoll ausgewählt. Um in TIA-Portal die Verbindung via HTTP aufzubauen, benötigt man bestimmte Libraries die von Siemens zu Verfügung gestellt werden. Diese müssen dann wie im Punkt 3.2.3 gezeigt eingebunden werden, um die Funktionsbausteine der Library nutzen zu können.

Funktionsbausteine
In der Library stehen dann folgende Bausteine zur Verfügung:

- GET
- POST-PUT
  Mit dem POST-PUT Befehl werden Daten zum Server geschickt aber auch Daten erwartet.

3.4.3 Datenfilterung

Die vom Server geschickten Daten werden in einem Befehl geschickt. Aus diesem Befehl muss herausgelesen werden um welchen Befehl es sich handelt, und die Daten die erforderlich sind um diesen Befehl auszuführen.

Datenformatierung Die Daten werden in einem String geschickt welcher in zwei Teile aufgeteitlt wird. Der zeite Teil ist jedoch abhängig vom ersten.

- 1.Teil:
  IDXXXXAXX
  Aus diesem Teil werden die ID-Nummer sowie der Auftrag herausgefiltert. Die ID-Nummer ist eine 4 stellige Nummer welche nach ID steht. Der Auftrag welcher ausgeführt werden muss steht in den zwei Stellen nach A.
  Diese werden nach folgender Codierung ausgelesem:

00: Kommissionierstation

01: Förderband

10: Lager 1 (Aus-/Einlagerung)

11: Lager 1 (Querförderer)

- 2.Teil:

3.5 Herausforderungen

4 Elektroplanung und Realisierung (Nikolaj Voglauer)

5 Sensorik und Sicherheitstechnik (Elena Widmann)

5.1 Aufgabenstellung

5.2 Sensorik

5.2.1 Endschalter

Beim Verplanen der Endschalter ist zwischen Software- und Hardware-Endschalter zu unterscheiden. Die Software-Endschalter begrenzen den Arbeitsbereich der Achse und sollten innerhalb des Bereichs der Hardware-Endschalter parametriert werden. Ihre Positionen werden direkt im Siemens TIA-Portal eingestellt und können falls notwendig einfach auf die aktuelle Geschwindigkeit angepasst werden. Werden die Software-Endschalter angefahren wird der Technologiealarm 533 ausgelöst und die Dynamikwerte werden gestoppt, das Technologieobjekt bleibt hierbei freigegeben. Werden sie jedoch überfahren wird das Technologieobjekt gesperrt.
Die Hardware-Endschalter begrenzen den maximal zulässigen Verfahrensbereich der Achse. Bei ihnen wird nicht unterschieden, ob die Endschalter angefahren oder überfahren werden. Beim Anfahren der Schalter wird der Technologiealarm 531 ausgelöst. Er sperrt das Technologieobjekt und muss, bevor der Auslösebereich der Hardware-Endschalter wieder verlassen werden kann, quittiert werden. [9]
Auf jeder der drei Achsen vom AFSS und auf dem Querförderer müssen Hardware-Endschalter montiert werden. Die Auswahl begrenzte sich hierbei auf die uns zur Verfügung gestellten Sensoren, welche ihren Funktion entsprechend auf den verschiedenen Positionen eingebaut werden.

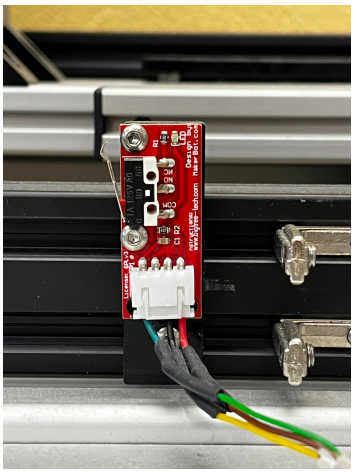
Positionsschalter mit Rollhebel
An der x-Achse werden als Hardware-Endschalter Positionsschalter mit Rollhebel verwendet. Davon besitzen drei jeweils einem Öffner- und einen Schließerkontakt [10], wohingegen einer der Endschalter aus zwei Öffnerkontakten besteht. [11] Um Einheitlich zu bleiben und da es sicherheitstechnisch auch von Vorteil ist (Drahtbruchsicherheit) verwenden wir jeweils einen der Öffnerkontakte der Endschalter.

Induktive Endschalter
Als Hardware-Endschalter an der y-Achse werden induktive Sensoren verwendet. Sie funktionieren so, dass durch eine Spule ein Magenetfeld erzeugt wird, welches dann in einem sich dem Sensor frontseitig nähernden elektrisch leitendem Material Wirbelströme erzeugt. Dadurch verändert sich das Magnetfeld und die Kontakte des induktive Sensors werden über einen Schmitt-Trigger geschaltet. Die Sensoren besitzen jeweils einen Öffner- und einen Schließerkontakt, wir verwenden jedoch ersteres um Drahtbruchsicherheit zu gewährleisten.

Endtaster
An der z-Achse und am Querförderer werden mechanische Endtaster verwendet. Auf einem Endtaster befindet sich ein Schließerkontakt in Form eines Tasters, welcher durch anfahren geschaltet wird.

(a) Induktiver Sensor [12]

(b) Rollendschalter [13]



(c) Endtaster

Abbildung 5.1: Endschalter

5.2.2 Referenztaster

Um die Motoren auf Position fahren zu können, müssen diese an allen drei Achsen und am Querförderer referenziert werden. Hierfür werden Opto Interrupter verwendet. In diesen befindet sich eine LED, welche auf einen Photo Transistor trifft. Dieser schaltet daraufhin durch und es liegt eine Spannung am Emitter an. Wird jetzt jedoch der Lichtstrahl der LED unterbrochen, sperrt der Transistor. Bei der SPS Programmierung ist daher zu beachten, dass der Ausgang des Sensor beim Auslösen auf LOW geschalten wird.

Da die Opto Interrupter über ASi-Bus mit der SPS verbunden werden sollen, welche eine Spannung von 24V liefert, aber nur mit einer viel niedrigeren Spannung betrieben werden dürfen, muss eine eigene Platine entworfen und gelötet werden. Hierfür wird die Software Fusion 360 verwendet, welche das Designen von Leiterplatten ermöglicht. Hergestellt werden diese dann durch unsere eigene schulinterne Leiterplattenfertigung.

Die LED im Opto Interrupter hat einen Spannungsabfall von 1,2V und maximal dürfen 1,35V darüber abfallen. Deshalb benötigt die LED einen Vorwiderstand. Dieser kann über das Ohm'sche Gesetz berechnet werden.

5.2.3 Lichttaster

5.2.4 Barcode-Scanner

5.3 AS-Interface

5.3.1 Allgemeines

5.3.2 Programmierung im TIA-Portal

5.3.3 Unterverteilerplatine

5.4 Sicherheitstechnik

5.4.1 Grundanforderungen und Planung

5.4.2 Realisierung

6 Anhang

Im Anhang befinden sich weitere Detailinformationen des Projekts wie
• Datenblattauszüge, Fertigungsunterlagen (PCB-Layouts, Gehäusezeichnungen, 3D-Druckunterlagen, Montageanleitungen,. . .) etc.
• sämtliche geforderten Projektmanagementdokumente
• ein Businessplan (optional)

6.1 Projektmanagement

In diesem Kapitel soll auf das Projektmanagement des Projektes eingegangen werden. Zu Beginn empfiehlt es sich, die einzelnen Bereiche des Projektmanagements zu erklären und anschließend in einzelnen Kapiteln zu behandeln.

6.1.1 Aufgabenstellung des Gesamtprojekts

Fügen Sie an dieser Stelle den Text der genehmigten Aufgabenstellung ein, der in die Diplomarbeitsdatenbank eingegeben wurde.

6.1.2 Scrum-Projektplan

Fügen Sie hier den vollständigen Scrum-Projektplan, wobei die Nummern der Tasks mit der Arbeitszeitaufzeichnung übereinstimmen müssen. Der Scrum-Projektplan kann auf mehrere Seiten aufgeteilt werden.

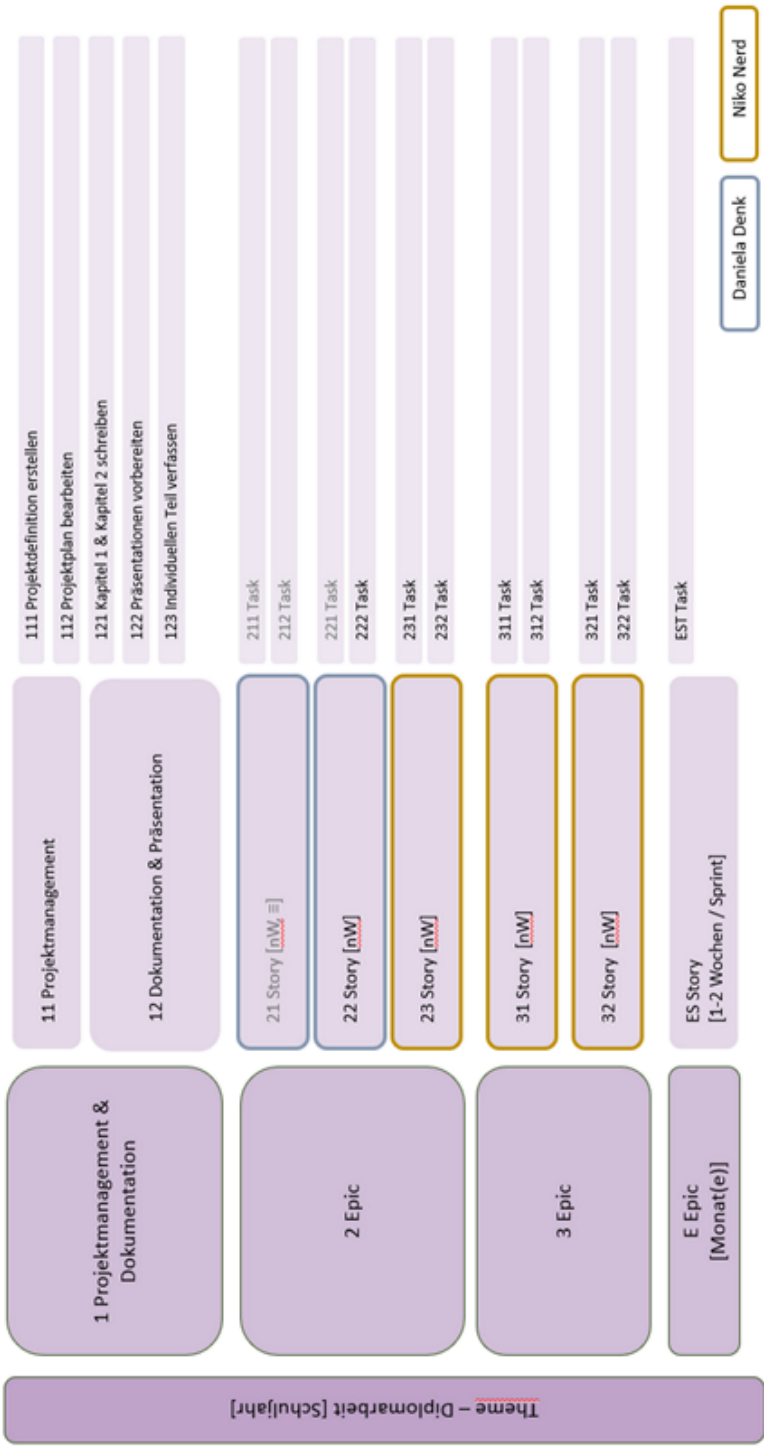


Abbildung 6.1: Scrum Projektplan mit Tasks

6.1.3 Terminplanung

6.2 Inbetriebnahme

Nachdem typische Projekte aus mehreren Komponenten bestehen, ist es oft nicht trivial die einzelnen Komponenten korrekt zu konfigurieren und das Gesamtsystem in Betrieb zu nehmen. In diesem Kapitel soll eine vollständige, präzise und trotzdem möglichst kompak-te Anleitung zur Inbetriebnahme des Systems dargelegt werden. Die Schritte sollen in dem Detailgrad beschrieben werden, dass ein durchschnittlicher Schüler des vierten Jahrganges das Projekt in Betrieb nehmen kann. Exemplarisch sollten Punkte wie die folgenden be-handelt werden – die Aufzählung ist nicht vollständig):

- Treiberinstallationen und Systemkonfigurationen
- Zu empfehlen wäre bei Server-Installationen ein Setup-Script, welches auf einem vor-definierten Docker-container aufbaut.
- Welche Schritte sind notwendig, um das Projekt mit dem vorhandenen Code / Schalt-plänen (auf GIT, CD, Netzlaufwerk, etc.) in Betrieb zu nehmen.
- Bei Schaltungen mit mehreren Platinen muss beschrieben werden, wie diese mit-einander verbunden werden müssen.

6.3 Kostenaufstellung

Für die Kalkulation im Gesamtprojekt sind folgende Kosten zu erfassen:
• Kosten für Material (Hard- und Software)
• externe Kosten (z.B.: Zukauf von Sensoren, Funkmodule, spezielle Entwicklungsum-gebungen, etc.)

Diplomarbeit: Titel der Diplomarbeit (DE), 03.05.2024		
Kostenart	**Stückpreis / Einheiten**	**Kosten in €**
Elektronische Bauteile		
Sonstige Materialien		
Software Lizenzen		
PCB Fertigung, Gehäuse		
Baugruppen, Module		
usw.		
	Gesamtkosten	

Abbildung 6.2: Kostenaufstellung

6.4 Besprechungsprotokolle

Eine entsprechende Vorlage wird auf den Schulrechner in Form einer Word-Vorlage bereitgestellt. Scannen Sie die vier Besprechungsprotokolle (laut Vorlage) ein und fügen Sie diese nachfolgend an (eine A4-Seite pro Protokoll)
Protokolle zu den einzelnen Iterationen:
• Vorprojektphase
• Iteration 1
• Iteration 3
• Iteration 5
Die Termine und Inhalte/Projektstatus entsprechen denen der Iterationspräsentationen.

6.5 Arbeitsnachweis

Jedes Teammitglied (Schüler/in) hat einen vollständigen Arbeitszeitnachweis, der außerhalb des Unterrichts verrichteten Tätigkeiten, in tabellarischer Form zu erbringen.
Eine entsprechende Vorlage wird auf den Schulrechner in Form einer Excel-Vorlage bereitgestellt.

6.6 Wettbewerbe

Teilnahme und Erfolge bei Wettbewerben.

6.7 Businessplan (optional)

Halten Sie sich, was die Struktur des Businessplanes angeht, an die Angaben, die Sie im WIRE-Unterricht bekommen. Das Anfügen eines Businessplans ist durch den/die WIRE Lehrer/in vorab zu genehmigen.

Literaturverzeichnis

[1] Pickengine. *Pickengine*. Online. URL: https://www.pickengine.com/.

[2] elektro4000de. URL: https://www.elektro4000.de/Kabel-Leitungen/Kabeltragsysteme/Stiel-Profilschienen/Verbinder/Item-Standard-Verbindungssatz-0-0-026-07::1743227.html (besucht am 19. 11. 2024).

[3] globalsources.com. URL: https://www.globalsources.com/Plastic-coated/plastic-coated-bearing-1197379614p.html (besucht am 19. 11. 2024).

[4] igus. *10 Tipps für eine langlebige „freitragende" Energieführung*. Online. 2024. URL: https://www.igus.at/service/kontakt/energieketten-tipps-fuer-freitragende-bewegung (besucht am 10. 12. 2024).

[5] igus. *9 Tipps für eine langlebige „stehende" Energieführung*. Online. 2024. URL: https://www.igus.at/service/kontakt/energieketten-tipps-stehende-anwendungen (besucht am 10. 12. 2024).

[6] OpenAI. *ChatGPT: OpenAI's Language Model*. https://openai.com/chatgpt. 2024. (Besucht am 21. 11. 2024).

[7] Siemens. *10 Tipps für eine langlebige „freitragende" Energieführung*. Online. 2024. URL: https://www.igus.at/service/kontakt/energieketten-tipps-fuer-freitragende-bewegung (besucht am 10. 12. 2024).

[8] cloudflight. 2024. URL: https://www.cloudflight.io/de/blog/docker-container-die-zukunft-moderner-applikationen-und-multi-cloud-deployments/ (besucht am 15. 11. 2024).

[9] Siemens. *S7-1500/S7-1500T Achsfunktionen V5.0 im TIA Portal V16*. Online.

[10] Schmersal. *Datenblatt Z4VH 335-11Z-M20-RMS*. Online. URL: https://products.schmersal.com/de_AT/export/product/846/pdf.

[11] Schmersal. *Datenblatt Z4VH 335-02Z-M20*. Online. URL: https://products.schmersal.com/de_AT/export/product/6992/pdf.

[12] Pepperl+Fuchs. *Datenblatt NBN8-18GM60-A2-V1*. Online. URL: https://files.pepperl-fuchs.com/webcat/navi/productInfo/pds/326162-0005_ger.pdf?v=20240130135820.

[13] Schmersal. *Produktbild*. Online. URL: https://products.schmersal.com/de_AT/z4vh-335-02z-m20-101167776.html#mz-expanded-view-708407000705.

Abbildungsverzeichnis

Abb. 2.1 Item Profil mit Standartverbindungssatz, Quelle: [2] 10
Abb. 2.2 V-Slot-Profil mit V-Wheel, Quelle: [3] 11
Abb. 2.3 Prototypen 11
Abb. 2.4 Zahnriemenkontraktion grafisch dargestellt 17
Abb. 2.5 Bauteilzeichnung Umlenkrolle 21
Abb. 2.6 Umsetzung der Umlenkrollen 22
Abb. 2.7 Gesamtüberblick des Servers 24
Abb. 2.8 Diagramm der BMOS bereitstellung 29
Abb. 2.9 Datenbankschema des AFSS 30
Abb. 2.10 Docker-Logo: [8] 31
Abb. 5.1 Endschalter 41
Abb. 6.1 Scrum Projektplan mit Tasks 44
Abb. 6.2 Kostenaufstellung 47

Tabellenverzeichnis

Tab. 2.1 X-Achse unbeladen und eingefahren 16
Tab. 2.2 X-Achse beim Ladevorgang 16