Projektarbeit PREN2 – FS22

Dmitriy An

Stefano Nicora

David Pfenniger

Michael Tschan

Cyrill Küttel

Garteroboter.li

Hochschule Luzern – Technik & Architektur

Horw, 10.06.2022

Projektarbeit PREN2 – FS22

Dmitriy An,
Studiengang Digital Engineering

Stefano Nicora,
Studiengang Elektrotechnik und Informationstechnologie

David Pfenniger,
Studiengang Elektrotechnik und Informationstechnologie

Michael Tschan,
Studiengang Maschinentechnik

Cyrill Küttel,
Studiengang Informatik

Garteroboter.li

Betreuender Dozent:
Pierre Kirchhofer

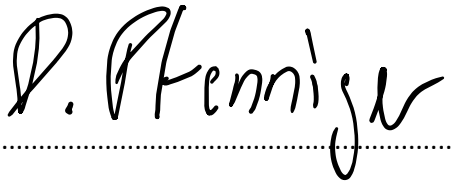
Hochschule Luzern – Technik & Architektur
Horw, 10.06.2022

Redlichkeitserklärung

Die Gruppe 38 erklärt somit, dass die folgende Arbeit von uns allen und selbstständig geschrieben wurde. Die verwendeten Quellen wurden alle im Literaturverzeichnis gemäss den Richtlinien einer wissenschaftlichen Arbeit festgehalten.

Stefano Nicora
Luzern, 27.05.2022
....

David Pfenniger
Büron, 27.05.2022
....



Michael Tschan
Vitznau, 27.05.2022
....

Cyrill Küttel
Luzern, 27.05.2022
....

Dmitriy An
Kriens, 27.05.2022
....

Ort, Datum Unterschriften

Abstract

Die Digitalisierung nimmt immer mehr zu und davon möchte auch der Garten- und Gartenbaubetrieb profitieren. In einem grossen Lager kann es zu langen Suchzeiten kommen und bei vielen Arten von Pflanzen besteht eine grössere Verwechslungsgefahr. Ziel dieses Projektes ist es einen «Garten-Roboter» zu konzipieren. Im Auftrag der Hochschule Luzern – Technik & Architektur, im Modul Produktentwicklung, soll der sogenannte «Garteroboter.li» entwickelt werden. Er soll in der Lage sein, autonom Topfpflanzen zu erkennen und daraufhin eine andere, gattungsgleiche Topfpflanze in einer Menge anderer Pflanzen wiedererkennen. Der Fortschritt des Roboters sowie die Art der Pflanze sollen auf einer Webseite in Echtzeit dokumentiert werden. Die Konzeption wurde bereits im Modul Produktentwicklung 1 erstellt und im Modul Produktentwicklung 2 wird die Konzeption realisiert und getestet. Zu Beginn wurde ein Projektplan und eine neue Risikoanalyse erstellt bevor mit der Realisierung begonnen wurde. Das Garteroboter.li wurde schrittweise aufgebaut und gleichzeitig auch optimiert. Gegen Ende der Realisierung wurde der Garteroboter.li verifiziert und validiert und weil es immer Raum für Verbesserungen gibt, wurden für den Rest der Arbeit Optimierungen durchgeführt. Es wurde festgestellt, dass vieles aus der ursprünglichen Konzeption angepasst werden musste und mehrere Aspekte nicht beachtet wurden, welche erst nach den ersten Tests ersichtlich wurden. Schlussendlich bewiesen mehrere Tests, dass der Garteroboter.li fähig ist, den Parcours zu überwinden und somit wurde der Auftrag der Hochschule Luzern erfolgreich erfüllt.

Inhaltsverzeichnis

1. Einleitung

Vom stetigen technischen Fortschritt möchte auch der Garten- und Gartenbaubetrieb profitieren. Je nach Pflanzenlager kann es mühsam sein, bestimmte Pflanzen zu finden, weil viele von blossem Auge schwer zu unterscheiden sind und bei einer grösseren Anzahl Pflanzen schnell die Übersicht verloren wird.

Diese Dokumentation ist eine Fortsetzung einer vorherigen Arbeit, welche im Modul PREN1 an der Hochschule Luzern – Technik und Architektur begonnen wurde. Die vorherige Dokumentation beschrieb das Vorgehen der Suche für Lösungsvarianten und erläuterte detailliert das Konzept eines Gartenroboters, welcher «Garteroboter.li» getauft wurde. Der Garteroboter.li soll in der Lage sein eine vorgegebene Strecke abzufahren, eine Pflanzenart zu erkennen und daraufhin die gleiche Art ebendieser in einer Menge verschiedenster Pflanzen wiederzuerkennen. Dies muss alles auf einer Webseite in Echtzeit dokumentiert und visualisiert werden.

Für den zweiten Teil dieses Projektes wurde ein neuer Projektplan auf Trello erstellt, damit Meilensteine und Termine bzw. Abgaben ersichtlich werden. Daraufhin wurde eine neue Risikoanalyse durchgeführt, um neue Risiken zu bewerten und nicht mehr gültige Risiken zu entfernen. Danach wurde beschrieben wie der Garteroboter.li realisiert und anschliessend auch optimiert wurde.

Diese Arbeit wurde in vier Kapitel aufgeteilt. Als erstes wurde die Organisation und das Projektmanagement erläutert und daraufhin die Realisierung und Optimierung dokumentiert. Im Kapitel vier wurden Tests und Validierungen erfasst und schlussendlich die ganze Dokumentation in einer Schlussdiskussion zusammengefasst.

2. Organisation und Projektmanagement

In diesem Kapitel wird die Organisation und das Vorgehen im Projektmanagement erklärt. Es wird auf die Gruppe 38, deren Projektplanung, die Kommunikation, die Kosten und die Risikoanalyse, welche im Verlauf des Projekts entstanden sind, eingegangen.

2.1 Gruppe 38

Die Gruppe 38 besteht neulich nur noch aus fünf Teammitgliedern, weil Oswald Zum Wald das Team verlassen hat. Die Untergruppen bestehen immer noch aus Stefano Nicora und David Pfenniger, dem Elektrotechnik Team, Cyrill Küttel welcher alleinig das Informatik Team bildet, Michael Tschan der jetzt auch alleinig das Mechanik Team bildet und Dmitriy An, dem Projektleiter und Ansprechperson der Gruppe. Ein neues Team-Organigramm wurde für die Visualisierung erstellt (9.1 Team-Organigramm Gruppe 38).

2.2 Projektplanung

Für die Projektplanung wurde in PREN2 Trello benutzt, da es mehr Übersicht darüber schafft, welches Team was zu erledigen hat. Im Projektplan wurden die Aufgaben in Informatik, Mechanik, Elektronik, Organisatorisches und Meilensteine eingeteilt. Der Gant-Chart ist ein Bonus, wurde jedoch mehrheitlich nicht benutzt, weil wir uns an den Endterminen jeder Aufgabe orientiert haben. Die vorgegebenen Testat-Abgaben wurden als Richtlinien für das Tempo des Projektes benutzt.

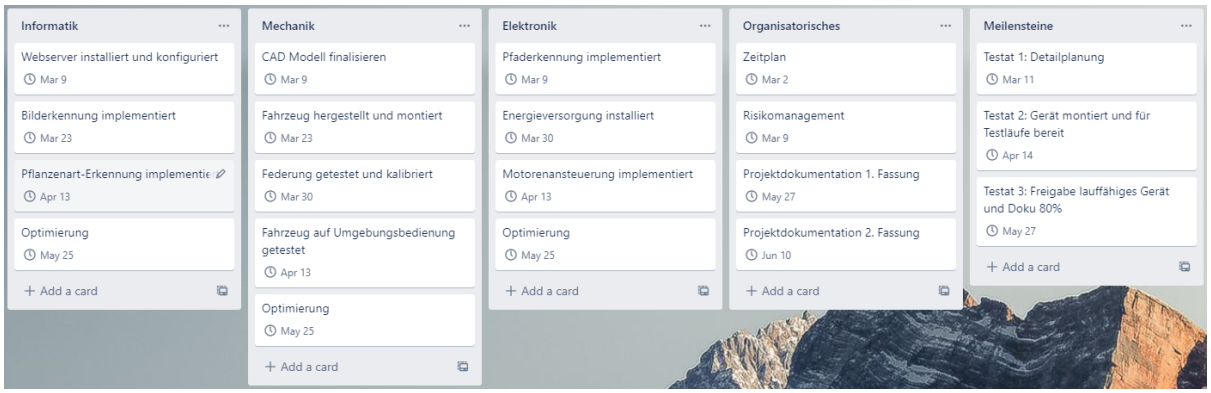


Abbildung 1: Trello Detailplan

2.3 Kommunikation

Die Kommunikation wurde gleich wie im PREN1 organisiert. Für die meisten Angelegenheiten wie z.B. Informationsaustausch und Notizen wurde Discord benutzt, weil Text- und Videokanäle sehr einfach zu organisieren sind. Für aktuelle Fortschritte und dringende Angelegenheiten wurde weiterhin WhatsApp verwendet. Jeweils am Donnerstag wurde ein Coaching mit dem betreuenden Dozenten durchgeführt, um aktuelle organisatorische Informationen zu vermitteln und sonstige Fragen zu stellen. Zusätzlich war der Dozent via E-Mail erreichbar.

2.4 Kosten

Im Verlauf vom ganzen Projekt wurde eine Kostentabelle (9.2 Kostentabelle) geführt, weil ein festes Budget vorgegeben wurde. Das Limit von 200 CHF aus PREN1 wurde aufgehoben und nur noch das Budget für 500 CHF bleibt bestehen. Gegen Schluss vom Projekt hatten wir ca. 400 CHF in den Garteroboter.li investiert.

2.5 Risikoanalyse

Zu Beginn von PREN2 wurde die vorhandene Risikoanalyse aus PREN1 aktualisiert (9.3 Risikoanalyse), damit entfallene Risiken nicht mehr beachtet und neue Risiken definiert werden. Die Risiken sind in die Disziplinen Elektronik, Mechanik und Informatik unterteilt und wurden in einer 3x3 Risikomatrix bewertet, welche in drei Gefahrenstufen unterteilt wurden.

Tabelle 1: Ausschnitt Risikoanalyse

Nummer	Risiko	Beschreibung	Massnahmen Eindämmen Schadenausmass	Massnahmen Eindämmen Eintrittswahrscheinlichkeit
1	**Übergreifende Themen**			
1.1	Fahrzeug fährt beim Start nicht los			Frühe Tests, Berechnungen, Programmierung kontrolliert
1.2	Finanzen - Geld geht aus			Budgetplanung und die Finanzen regelmässig kontrollieren

3. Realisierung und Optimierung

Dieser Abschnitt beschreibt die Vorgehensweise der Realisierung und der Optimierung des Garteroboter.li. Dazu gehören neue Erkenntnisse, neue Recherchen, Organisierung der Bauteile, Änderungen, welche vom ursprünglichen Konzept abweichen, Implementierung und Aufbau des Roboters. Die Unterkapitel wurden in die Disziplinen Mechanik, Elektronik und Informatik eingeteilt.

3.1 Mechanik

Im Bereich der Mechanik wurde die Essenz des Gesamtkonzepts aus dem PREN1 übernommen und nur wenig angepasst. Jedoch wurden noch kleinere Änderungen vorgenommen, um Bedienung, Platz und Verhalten des Fahrzeugs weiter zu optimieren.

3.1.1 Antriebseinheit

An der Antriebseinheit (Abbildung 2) hat sich vom Konzept her nichts verändert. Es werden noch immer die gleichen Getriebemotoren des Typs RB350018-2A723R verwendet. Das Kernproblem des Fahrzeugkonzeptes aus dem PREN1 war jedoch die etwas zu geringe Fahrzeughöhe aufgrund der kleinen Räder. Um diesem Problem entgegenzuwirken, wurde die Lenkeinheit und die Motorenhalterungen unter und nicht wie bisher auf die Grundplatte montiert. Mit dieser Massnahme konnte die Fahrthöhe um etwa 3 Zentimeter auf 7.5 Zentimeter erhöht werden. Da die Motoren aufgrund dieser Änderung stärker dem Wasser ausgesetzt werden, schützt eine neu konstruierte Motorhalterung aus PLA (Polyactic Acid) zusätzlich vor diesem Problem.

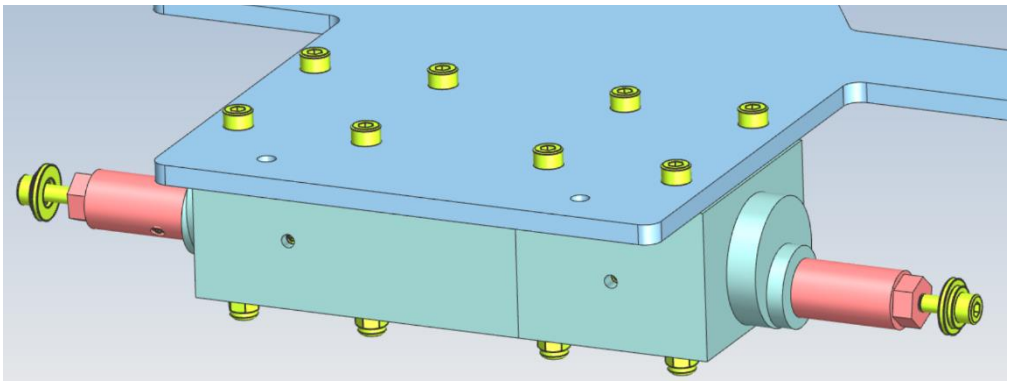


Abbildung 2: Antriebseinheit

3.1.2 Lenkeinheit

Das Konzept für die Lenkeinheit (Abbildung 3) blieb ebenfalls gleich. Eine Achsschenkellenkung soll unser Fahrzeug sicher auf dem Rundkurs bewegen. Dabei wurde die Lenkung auf einen Kurvenradius von 780 Millimeter konstruiert. Da die Lenkeinheit ebenfalls unter das Fahrzeug versetzt wurde, musste eine zusätzliche Aussparung in der Grundplatte vorgenommen werden, um Platz für den stehenden Lenkservo zu schaffen. Im Prototyp aus dem PREN1 waren die verschiedenen Komponenten aus MDF (Medium-density fiberboard) gefertigt. Da sich die Mehrheit der Bauteile nun unterhalb des Fahrzeugs befinden und nun häufiger mit Wasser in Kontakt kommen, wurden die betroffenen Bauteile durch Plexiglasteile ersetzt.

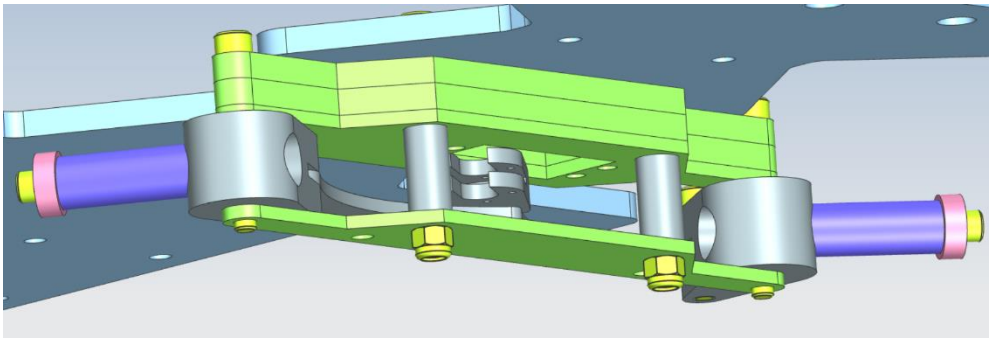


Abbildung 3: Lenkeinheit

3.1.3 Abdeckung und Aufbau

Der Aufbau und die Abdeckung hat sich grundsätzlich, im Gegensatz zum PREN1 Prototyp, verändert. Damals war geplant, dass der Aufbau aus Blech gefertigt werden sollte. Als wir das Bauteil «blexonierten», bemerkten wir, dass sich das Bauteil mit knapp hundert Franken zu Buche schlagen würde. Daraufhin unternahmen wir den Versuch, das Bauteil selbst aus Plexiglas herzustellen. Dieser Versuch schlug jedoch aufgrund fehlender Ausrüstung und Werkzeugen fehl. Das Bauteil konnte die geforderte Präzision, welche wir für die Führung der Federung benötigen, nicht erfüllen.
Der alte Ansatz war nicht der korrekte für unsere Fertigungsmöglichkeiten. Ausserdem konnte der Aufbau aufgrund der umplatzierten Lenkung und Antrieb grösser gestaltet werden.
Der neue Ansatz sieht eine grosse rechteckige Bodenplatte vor, an welche an den Rändern Stufen eingelassen wurden. Dieser Ansatz hat bemerkenswert gut funktioniert und es war an keinem Bauteil Nacharbeit notwendig, um diese zusammenzustecken.
An der Abdeckung hat sich nichts verändert. Sie musste jedoch dem neuen Unterbau angepasst werden und wurde dadurch etwas länger und hatte neue Lochpositionen.

3.1.4 Federung

Die Federung gehört nebst der Lenkung zu den komplexeren Baugruppen unseres Fahrzeugs. Dabei werden vier Führungsstifte auf die Grundplatte verschraubt, auf welchen Messingbuchsen laufen. An den Messingbuchsen wird der obere Aufbau verschraubt. Mit einer Kombination von Federrate und Schaumstoffdicke kann nun das Dämpfungsverhalten des Aufbaus und damit auch der Kamera bestimmt werden.
Das Anfangsproblem war hierbei, dass sich die Führungen oft auf den Führungsstiften verkanteten, obwohl die grundsätzliche Konstruktionsregel für das Verkanten von Führungen eingehalten wurde. Diese besagt, dass die Führungslänge mindestens 1.3 bis 1.5-mal länger ist als der Führungsdurchmesser. (Lobeck)

$$\frac{Geführte\ Länge}{Geführte\ Breite} = \frac{l}{b} > 1.3 \dots 1.5$$

Als zusätzliche Massnahme wurde ausserdem mit einer Passung H8 zwischen Buchse und Führungsstange gearbeitet, um das Verkanten weiter einzuschränken.

Die Ursache für das Verkanten war schliesslich der sehr grosse Hebelarm auf die Buchse über die gesamte Fahrzeuglänge. Dieser Effekt nennt man auch den Schubladeneffekt. Dieser tritt auf, wenn

die Bedingung $l < 2 \cdot a \cdot \mu$ nicht erfüllt ist. Folglich müssten unsere Führungsbuchsen aufgrund des Abstandes eine Höhe von mindestens 60 Millimeter haben, was uns konstruktiv nicht möglich war.

$$l < 2 * a * \mu \; = \; l < 2 * 300mm * 0.1 = l < 60mm$$

Durch das Lockern der Schrauben in den Führungsbuchsen, konnte das Problem entschärft werden. Nun kann sich die Bodenplatte des Aufbaus um einige Grad bewegen, bevor es ein Drehmoment auf die Buchsen ausübt.

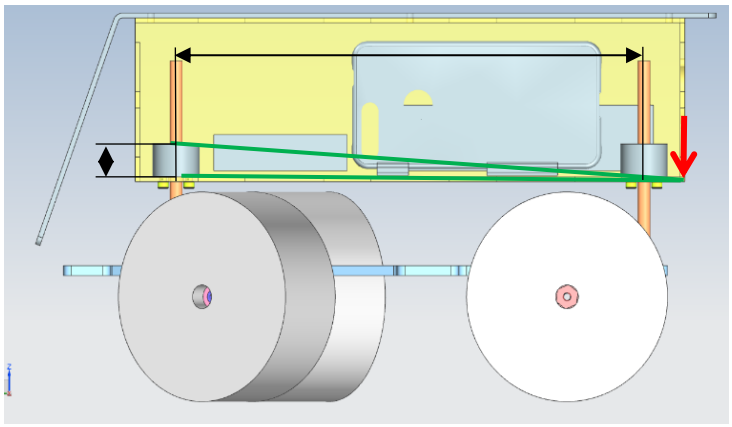


Abbildung 5: Seitenansicht

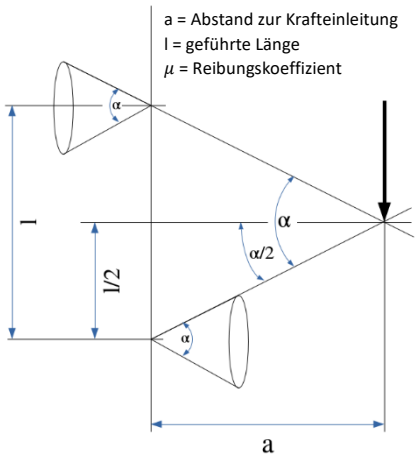


Abbildung 4: Schematische Darstellung einer Führung mit aussermittigem Kraftangriff und Reibkegeln (Schubladeneffekt, 2022)

3.1.5 Montage

Da das ganze Fahrzeug aus verschiedenen Funktionsbaugruppen wie Lenkung, Antrieb, Federung oder Elektronik und Bild besteht, ist die Montage dementsprechend einfach.

Grundsätzlich empfiehlt es sich so viele einzelne Teile der Funktionsgruppe bereits zu montieren, ehe sie am Fahrzeug befestigt werden. Dies macht die Handhabung wesentlich einfacher. So können zum Beispiel die Räder bereits auf ihre Achsen montiert werden. Auf der Vorderachse wird dabei eine Messingbuchse in das Rad eingepresst, welche wiederum auf einen Gleitsitz der Vorderachse passt. Diese Welle wird später mit dem Achsschenkel verstiftet. Auf der Hinterachse ist ein Sechskant im Rad eingelassen. Dieser passt auf die Antriebswelle der Hinterachse, welche so das Motordrehmoment optimal übertragen kann.

Alle genannten Baugruppen werden auf einer Grundplatte verschraubt. Dabei ist es einfacher mit den unten liegenden Baugruppen zu beginnen. So werden zuerst die Getriebemotoren in ihre Halterungen eingesetzt und unter die Bodenplatte montiert. Dabei ist darauf zu achten, dass die Antriebswelle gegen den Boden gerichtet ist. Da die Antriebswellen nicht zentrisch aus den Motorenhalterungen schauen, wird andernfalls die Bodenfreiheit um einige Zentimeter reduziert. Auf die herausschauenden Antriebswellen kann nun die vormontierte Radeinheit der Hinterachse montiert werden.

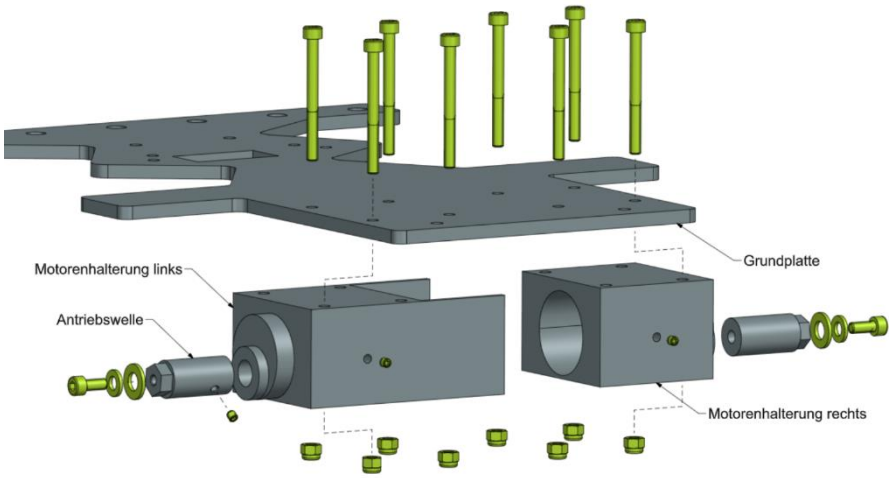


Abbildung 6: Explosionsdarstellung Antrieb

Nach der Antriebseinheit bietet es sich an, die Lenkeinheit als nächstes zu montieren. Dabei wird der Lenkservo in Kombination mit der unteren Achshalterung auf der Bodenplatte verschraubt. Auf diese werden die Distanzhalter und die Achsschenkel aufgesetzt. In die Achsschenkel wird nun die vormontierte Vorderachse eingeschoben und verstiftet. Die beiden Achsschenkel werden durch die Lenkstange in ihrer Position gehalten. In der Mitte dieser Lenkstange befindet sich der Kontaktpunkt für den Lenkservo. Dieser wird durch den Lenkhebel mit der Lenkstange befestigt. Um den ganzen Aufbau nun zu fixieren, kommt von oben die obere Achshalterung und klemmt das ganze Paket zusammen.

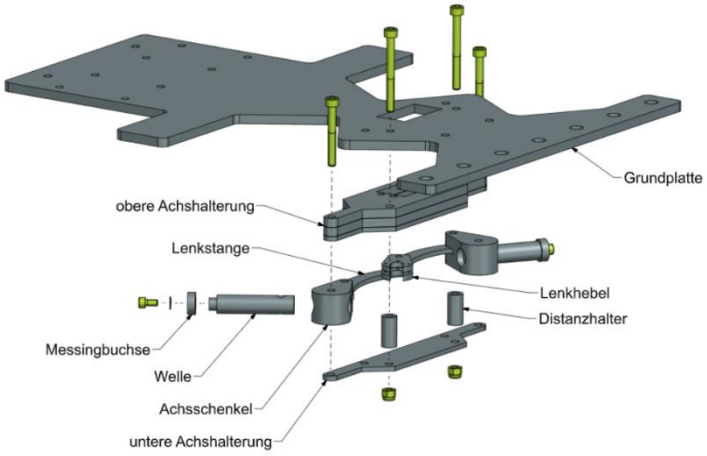


Abbildung 7: Explosionsdarstellung Lenkeinheit

Weiter geht der Aufbau auf der Oberseite der Grundplatte. Hier können nun die vier Führungsstangen der Federung montiert werden. Auf diese Führungsstangen werden die Federn positioniert. Die Messingbuchsen, welche von den Führungsstangen geführt werden, sind auf der Bodenplatte des oberen Aufbaus befestigt. Der obere Aufbau besteht aus Plexiglasscheiben, welche durch ein stufenförmiges Profil aufeinander abgestimmt sind. Diese individuellen Plexiglasscheiben werden durch Silikon in ihrer endgültigen Position verklebt. In diesem Zustand wird der obere Aufbau dann auf die Führungsstangen aufgeschoben. Um ein Herausspringen des Aufbaus aus den Führungen zu verhindern, werden am Ende der Führungsstangen mittels Schrauben Endanschläge gesetzt.

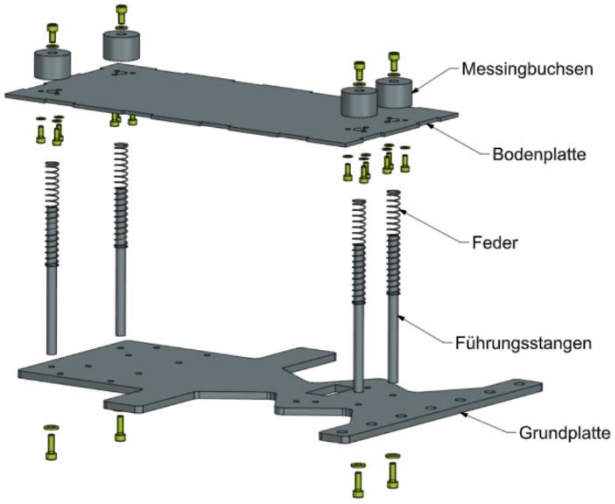


Abbildung 8: Explosionsdarstellung Federung

Nun können die ganzen Elektronikkomponenten und Schalter an ihren designierten Plätzen montiert werden. Für die Schalter sind Positionen an der rechten Seite vorgesehen. Die Kamera, respektive das Handy, befindet sich in Fahrtrichtung links. Sind alle Komponenten an ihrem Platz, so kann die Abdeckung des Fahrzeugs mit sechs Schrauben befestigt werden.

3.2 Elektronik

3.2.1 Antriebselektronik

In der Entwicklungsphase wurde der Antrieb der Motoren mittels eines OPs (Operationsverstärker), Leistungs-MOSFETs (metal-oxide-semiconductor field-effect transistor) sowie des DAC-Ausganges (digital-to-analog converter) konzipiert. Das Kernproblem dieser Lösung ist der schlechte Wirkungsgrad bei tiefer Drehzahl respektive der hohen Verlustleistung am MOSFET. Um diesem Problem Herr zu werden, wurde das Konzept von Grund auf überarbeitet. Die neu entwickelte Schaltung basiert im Kern auf einem Buck-Converter, also einem LC-Glied welches abhängig vom Duty Cycle eine variable Ausgangsspannung als Output generiert.

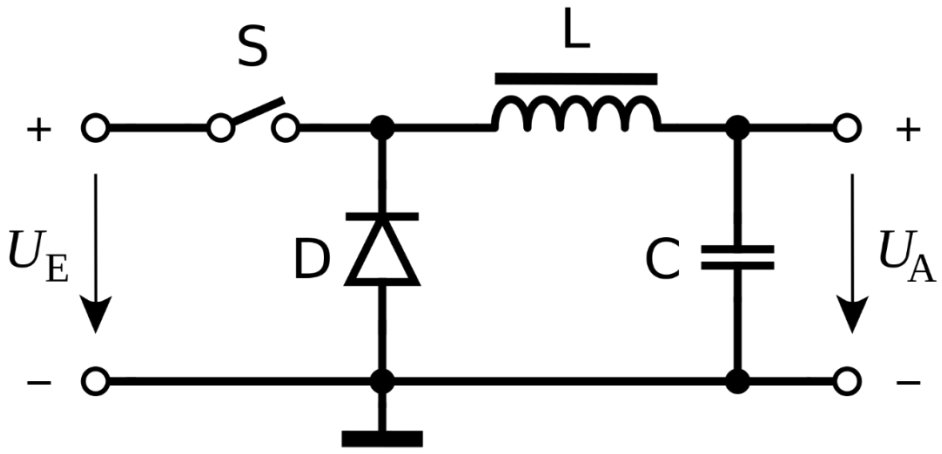


Abbildung 9: Buck-Converter (Abwärtswandler, 2021)

3.2.1.1 Highside-Switch

Um den Leistungs-P-Channel-MOSFET (Q4, siehe Abbildung 10) als Schalter und nicht in der "ohmic region" zu betreiben, muss der Transistor komplett durchgesteuert werden. Da die Threshold-Spannung des Transistors Q4 (Spannung zwischen Pin 1 und 3) herstellungsbedingt zwischen -2V und -4V variiert, kann dieser nicht mit den 3.3V des Mikrocontroller-Ausganges durchgesteuert werden. Um dieses Problem zu lösen, wird Q4 in der Highside-Switch-Konfiguration betrieben. Mittels des PWM-Signales des Mikrocontrollers (MOTOR_RIGHT) wird die Verbindung gegen Masse via Q2 Ein- bzw. Ausgeschaltet. Dadurch entsteht am Gate des Q4 (Pin 1) Masseverbindung respektive VCC (Speisespannung). R9 dient als Pullup-Widerstand um undefinierte Zustände des Gates von Q4 zu vermeiden, sollte Q2 ausgeschaltet sein. R10 ist optional, und wird eingesetzt sollte $V_{CC} > U_{GS\ max}$ überschreiten (gemäss Datenblatt $U_{GS\ max} = \pm 20V$). Somit wird dieser nicht benötigt.

3.2.1.2 Buck-Converter

Der Buck-Converter variiert mittels Wechselwirkung zwischen Spule (L) und Kondensator (C) die Ausgangsspannung, indem der Duty-Cycle (Verhältnis der Einschalt- und Ausschaltzeit vom Transistor Q4) angepasst wird. Je länger Q4 sich im leitenden Zustand befindet, desto höher wird das in der Spule gespeicherte Magnetfeld. Sperrt Q4 anschliessend wieder, wird durch die im Magnetfeld der Spule gespeicherte Energie ein Strom in die Leitung induziert. Da dieser weder durch das C noch die Freilaufdiode D9 abfliessen kann, muss er durch den Motor fliessen um anschliessend via Rückführungsdiode D5 wieder an den negativen Pol der Spule zu gelangen. Nebst der

Signalglättung wird der Kondensator dazu verwendet, Potentialsprünge zu unterbinden. Ohne Kondensator springt die Spannung an einer Spule in einer Schaltung schlagartig und ohne Spule springt der Strom.

3.2.1.2.1 Dimensionierung der Bauteile

Um den Buck-Converter dimensionieren zu können, müssen gewisse Parameter definiert werden oder aus den Datenblättern gelesen werden (Tabelle 2).

Tabelle 2: definierte Parameter für die Dimensionierung des Buck-Converters

Eingangsspannung (V_{IN})	15	V	Schaltfrequenz Q4 (f_{SW})	10	kHz
Max. Ausgangsspannung (V_{OUT})	12	V	$R_{DS\,ON}$ (gemäss Datenblatt)	60	mΩ
Max. Ausgangsstrom (I_{OUT})	4	A	Spannungsabfall über Diode D5 (V_F)	0.7	V

$$\text{DutyCycle}_{@12V\,V_{OUT}} = \frac{V_{OUT} + V_F}{V_{IN} - V_{RDSon}} = \frac{12V - 0.7V}{15V - (R_{DSon} \cdot I_{OUT})} = 0.86$$

$$Einschaltzeit_{Q4} = t_{ON} = \text{DutyCycle}_{@12V\,V_{OUT}} \cdot \frac{1}{\frac{f_{sw} \cdot 10^3}{10^{-6}}} = 0.86 \cdot 100\mu s = 86\mu s$$

$$Induktivität_{Spule} = \frac{(V_{IN} - V_{OUT} - V_{RDSon}) \cdot t_{on}}{2 \cdot 0.1 \cdot I_{OUT}} \approx 280\mu H$$

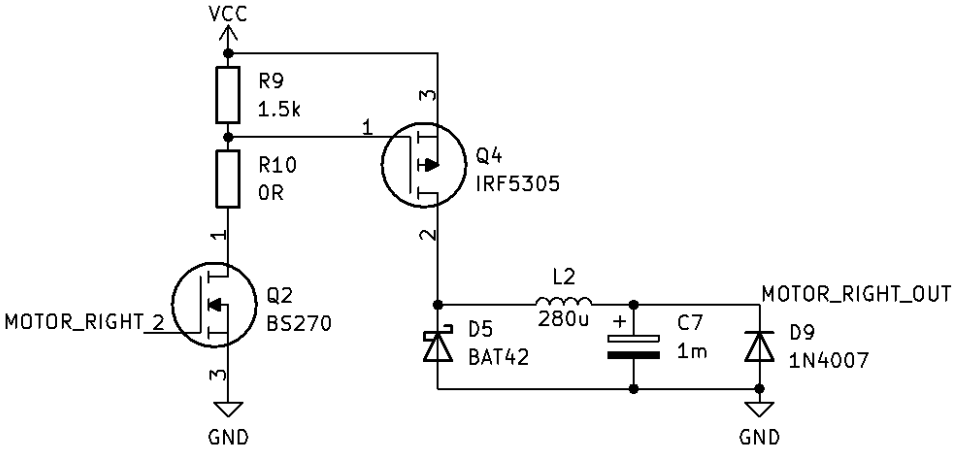


Abbildung 10: Highside-Switch

3.2.2 PCB

Im Gegensatz zu PREN1 befindet sich die Antriebselektronik sowie die Kabeldetektion nicht mehr auf Lochrasterplatinen, wie dies bei Prototypen üblich ist, sondern auf einem PCB (Printed Circuit Board). Die Flexibilität ist nach der Bestellung bei einem PCB zwar beschränkt, da alles im Vorhinein in einer CAD-Software geplant und gezeichnet wird, jedoch ist das Ergebnis sauberer und liefert konstant dieselbe Qualität.
Auf dem PCB wurden die beiden Funktionen (Antrieb und Kabeldetektion) möglichst geographisch getrennt, um Interferenzen, vor allem seitens der Antriebseinheit, zu minimieren. Die Speisung der Komponenten erfolgt extern, via Akku für V_{CC} sowie mittels des 5V DC-DC-Wandlers.

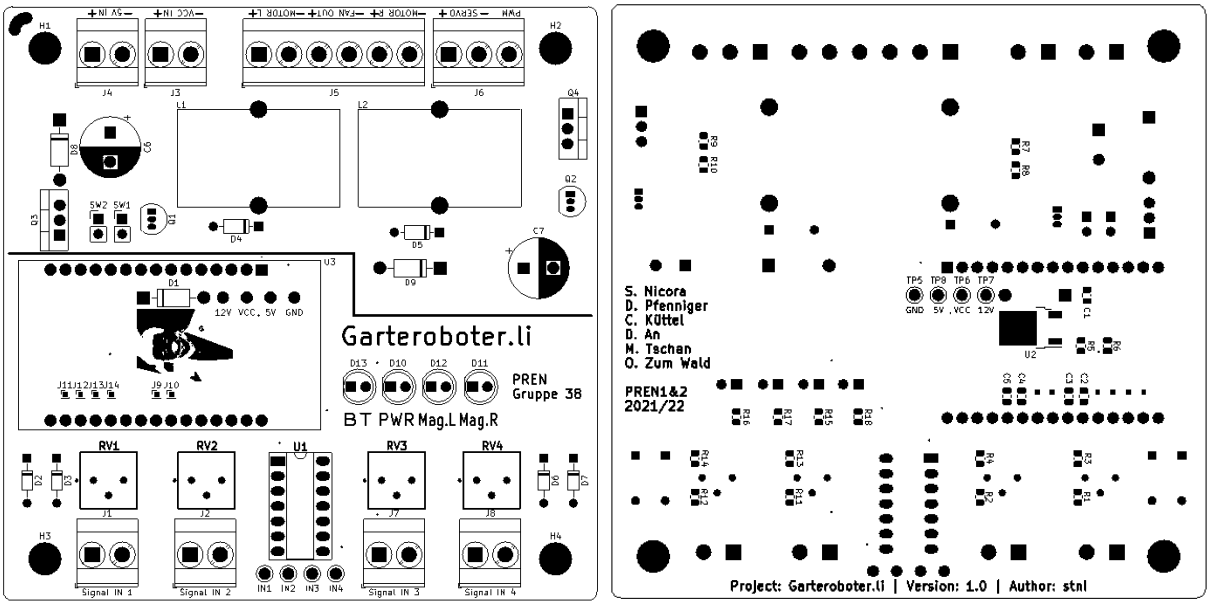


Abbildung 11: Layout PCB

3.2.3 Wegerkennung – Sender

Zusätzlich zur Leistungsoptimierung der Motorenansteuerung wurde auch der Signalgeber angepasst. Während die Induktionsschleife vorher mithilfe vieler Leistungswiderständen und einer 12V Speisung mit Strom durchflossen wurde, wird neu mit einem Step-Down-Converter 12 -> 5V gearbeitet. Dadurch verringert sich die Verlustleistung mit gleichem Strom und Magnetfeld um Faktor 2.4. Da ein Kabel mit mehreren Leitern zur Verfügung steht, wird der Strom zweimal durch die Induktionsschleife geführt. Dadurch erhöht sich zwar die Induktivität des Kabels, was dank der niedrigen Frequenz und der neuen Freilaufdiode aber keine Probleme verursachen wird.

3.2.4 Software

Zusätzlich zum im PREN1 abgelieferten Code kam dieses Semester verhältnismässig nur wenig dazu. Die Regelung, welche vorher nur aus einem P-Anteil (proportional zum Messfehler) bestand, wurde jetzt um einen zusätzlichen D-Anteil (derivative) erweitert. Das bedeutet, es wird auch proportional zur zeitlichen Ableitung des Fehlers geregelt. Dieser Anteil soll das Einschwingverhalten verbessern, sodass die Regelung aggressiver ausgelegt werden kann, ohne dass grössere Überschwinger (Slalomfahrten) entstehen. Der I-Anteil (integral) wird hier weggelassen, da ein kleiner stationärer Fehler nicht schlimm ist und so auch kein wind-up Problem entstehen kann.

Um die richtigen Parameter für die Regelung zu finden, wird während des Fahrens via Serial Bluetooth App mit dem Fahrzeug kommuniziert. So kann beispielsweise der D-Anteil der Regelung mittels Trial-and-Error eingestellt werden. Zudem werden laufend Statusmeldungen und Messwerte auf dem Smartphone ausgegeben.
Da mit der neuen Regelung schneller gefahren werden soll, ergibt sich auch ein Problem in der Beschleunigung. Vor allem beim Start soll das Fahrzeug nicht schlagartig Drehmoment auf die Räder geben, sondern langsam beschleunigen. Dafür wurde in der Software das Konzept der Motorenansteuerung überarbeitet. Auch während der Fahrt sind ruckartige Änderungen der Drehzahlen zu vermeiden, um zu verhindern, dass die Räder durchdrehen oder die Vorderachse in die Höhe springt. Auch hier, wie beim Einstellen der PD-Regler Parameter, ergibt sich ein trade-off zwischen stabilem Fahren ohne Oszillation und einer genug aggressiven Lenkung, um die engen Kurven zu fahren.
Eine weitere Herausforderung stellten die Totzeiten im Regelkreis, beziehungsweise die Ausführzeit eines Programmzyklus auf dem Mikrocontroller. Um diese Zeit zu verkürzen, wird die Ein- und Ausgabe in der Endversion auf das Mindeste reduziert, da jedes via Bluetooth versendete oder eingelesene Zeichen seine Rechenzeit braucht. Zudem wurden, wo immer möglich, alle Berechnungen mit ganzzahligen Werten gemacht, da Rechnungen mit Kommastellen (floatingpoint operations) auch mehr Rechenzeit in Anspruch nehmen, gerade weil auf dem ESP32 keine FPU (floating point unit, Kommazahlenrechner in Hardware) verbaut ist.
Falls der Regler trotz allem doch versagen sollte und das Fahrzeug sich so weit vom Kabel entfernt, dass man im Messignal nicht mehr zwischen Nutz- und Rauschsignal unterscheiden kann, dann springt die «Notfall-Lenkung» ein. Im Hintergrund wird bei genügend intensivem Signal gespeichert, in welche Richtung man zu lenken hat. In diese Richtung wird dann die Notfall-Lenkung voll einsteuern, bis das Fahrzeug wieder ein Signal vom Kabel empfängt.

3.3 Informatik

Im Bereich der Informatik wurde das Konzept aus PREN1 weitgehend beibehalten. Es wurde Software implementiert und getestet. Danach wurde wieder Software implementiert und getestet, das Ganze in einem sich immer wiederholenden Zyklus. Im Verlauf der Entwicklung wurde stets darauf geachtet, trotz zunehmender Komplexität den Code in einfache, verständliche Teile zu zerlegen.

3.3.1 Smartphone

Der Ansatz der Informatik ist unkonventionell und keinesfalls eine Standardlösung, denn anders als bei vielen anderen Teams wird kein Raspberry Pi verwendet. Gruppe 38 verwendet ein Handy, auf dem eine Android-App läuft.
Das Herzstück der Informatik ist dieses Smartphone. Es ist die zentrale Komponente, welche Informationen erhält, verarbeitet und Befehle an den ESP32 Mikrokontroller sendet. Das Google Pixel 3 wurde ausgewählt, da es über ausreichend Perfomance verfügt und kostengünstig erhältlich war. Ausserdem verfügt es über eine hochauflösende Kamera. Das Handy ist somit eine elegante Komplettlösung, welches folgende Hardware-Features vereint.

- Hochauflösende 12-Megapixel Hauptkamera
- Internet: WLAN und LTE
- Bluetooth für die Kommunikation mit dem Mikrokontroller.

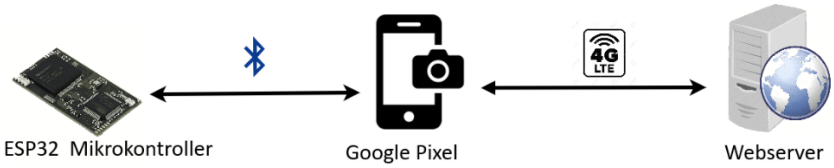


Abbildung 12: High-Level Übersicht

Um die Unabhängigkeit von der vorhandenen WLAN-Infrastruktur zu gewährleisten, ist dem Handy eine SIM-Karte hinzugefügt worden. Das minimiert das Risiko einer unzuverlässigen Internetverbindung. Sollte der Fall eintreten, dass der WLAN-Empfang unzureichend ist, ist das keineswegs ein Desaster, da wir durch das Mobilfunknetz eine Absicherung haben. Internetzugang ist unerlässlich, folglich wurde eine redundante Lösung geschaffen.

Das Smartphone wurde gerootet und mit einer massgeschneiderten Firmware (LineageOS 18.1) ausgestattet. Hauptmotivation für dieses Unterfangen war die Erhöhung der Leistungsfähigkeit. Diese Custom ROMs sind in der Regel sehr schlank, denn sie kommen mit dem absoluten Minimum an vorinstallierten System-Apps aus. Das ist von Vorteil, weil dann möglichst wenig Prozesse im Hintergrund laufen und wertvolle Ressourcen (CPU, RAM) verbrauchen.

3.3.2 Android-App

Abbildung 13: Die Android-App nach dem Startsignal

3.3.2.1 Entwicklung

Die Android-App ist die mit Abstand grösste Komponente der Informatik. Die App wurde nativ in Android geschrieben. Das ist notwendig, um Hardwarezugriff auf Schnittstellen zu Kamera und Bluetooth zu ermöglichen. Der ganze Software-Entwicklungsprozess konnte schon früh gestartet werden, da dieses System weitgehend unabhängig vom Roboter ist. Bereits Mitte Oktober 2021 sind erste Softwarekomponenten geschrieben worden. Seither ist die App stetig weiterentwickelt und verbessert worden.

Als Versionsverwaltungssystem wurde Git verwendet. Das Projekt ist öffentlich auf GitHub gehostet. Es sind kontinuierlich issues erstellt worden, weil die Softwareentwicklung ein iterativer Prozess ist. Ein Issue ist ein Tracking-Tool, welches im GitHub-Repository integriert ist und die Priorisierung von aktuellen Aufgaben unterstützt.

3.3.2.2 Testen und Feedback

Ein massiver Vorteil der Entwicklung eines Smartphone-Apps (besonders im Kontext von PREN) ist der Fakt, dass jederzeit, überall und augenblicklich getestet werden kann. Das liegt daran, dass das Smartphone sehr schwach mit dem Rest des Systems gekoppelt ist. Anders formuliert: Die Implementation der Software ist völlig unabhängig vom physischen Vorhandensein des Roboters. Fortlaufend sind einzelne Komponenten getestet worden. Das läuft folgendermassen ab: Man erstellt einfach einen Button und verbindet die zu testende Funktion mit diesem Button. Ab dann ist jeder Testdurchlauf wortwörtlich nur ein Knopfdruck entfernt.

Funktionen der Software mittels Buttons zu testen hat auch praktischen Wert: Die Teammitglieder konnten so die App mit minimalem Aufwand testen und fortlaufend Rückmeldungen geben.

3.3.2.3 Programmiersprachen

Die App ist in Java und Kotlin programmiert worden, wobei der Einsatz von Kotlin proportional mit der gemachten Erfahrung zugenommen hat. Kotlin hat einige Vorteile gegenüber Java in der Android Programmierung. Ein grosser Vorteil ist, dass der Boilerplate-Code abnimmt – weniger Code muss geschrieben und gelesen werden. Im Übrigen kann Kotlin-Code nahtlos mit bereits vorhandenem Java-Code integriert werden (Interoperabilität).

Der Codeabschnitt für die Echtzeit-Bildverarbeitung ist in C++ programmiert. Dieser Teil wird in einem weiteren Kapitel noch ausführlicher dokumentiert.

3.3.2.4 Zustandsübergänge der App

Im Verlauf der Objektdetektion bis zur Identifikation der Pflanze wechseln sich die Zustände, in welchen die App sich befindet. Die wichtigste Zustandsänderung ist die Objektdetektion, denn sie setzt die Hauptfunktionalität in Gang.

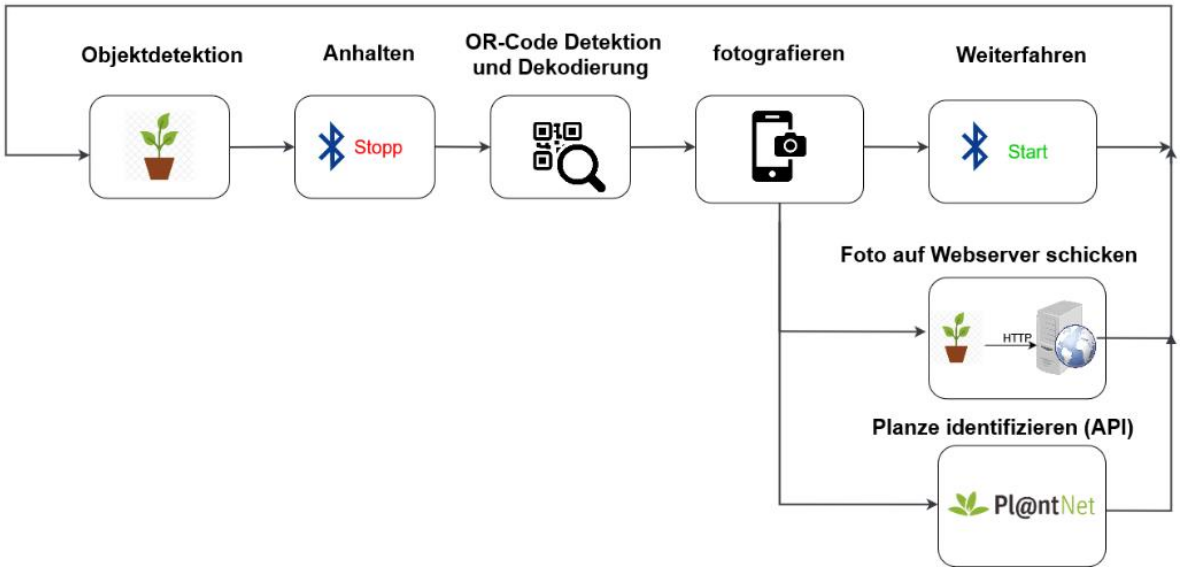


Abbildung 14: Ablauf Bilderkennung

Die obige Abbildung zeigt den Ablauf eines Durchgangs, welcher als Ganzes Teil einer grösseren Iteration ist. Die Abbildung ist eine simple Darstellung der Realität, denn sie zeigt nur den «happy-path» an. Das bedeutet das einfachste Szenario, in welcher der Codeabschnitt funktionieren soll, ohne dass Ausnahmen oder Fehlerzustände eintreten.
Fehlerzustände und deren Behandlungen sind nicht eingezeichnet. Betrachten wir zum Beispiel folgende Situation:

1. Die App kommt in den Zustand der *QR-Code Detektion*.
2. Es wird innerhalb 3 Sekunden kein QR-Code erkannt.

   Falls dieser Fall eintrifft, wird eine Annahme gemacht, sodass der Fehlerzustand überbrückt werden kann.

3. Es wird gefolgert, dass die *Objektdetektion* in Schritt 1 ein falsch-positives Resultat war.
4. Befehl zum Weiterfahren geben.
5. Zustandswechsel zurück auf *Objektdetektion*.

3.3.3 Objekterkennung

3.3.3.1 Objekterkennungsmodell

Um Objekterkennung in Echtzeit zu realisieren, wird ein besonderes leichtgewichtiges Modell verwendet. NanoDet ist ein Echtzeit Objekterkennungsmodell, welches speziell für Mobilgeräte ausgelegt ist (NanoDet, 2022). Die Leistung dieses Modells ist aussergewöhnlich und entspricht unseren Anforderungen. Um maximale Performanz zu erreichen, verwendet wir zusätzlich das Framework NCNN. NCNN ist ein hoch leistungsfähiges Softwareframework für neuronale Netze. Es ist ebenfalls explizit für den Einsatz auf mobilen Geräten konzipiert worden. Dieses Framework macht reichlich Low-Level Optimierungen, insbesondere für ARM-Prozessoren (Tencent/ncnn, 2022).

Abbildung 15: Detektion einer Topfpflanze

Basierend auf dem NCNN Framework ist ein vortrainiertes neurales Netzwerk verwendet worden, dass auf dem Coco-Datenset trainiert worden ist. Praktischerweise sind darin die Objekte *Topfpflanze* und *Vase* enthalten. Das sind exakt die Objekte, welche für den Gartenroboter relevant sind. Das Objekt 'Vase' mag vielleicht etwas fehl am Platz sein, jedoch kann damit der Topf als solches erkannt werden. Die beiden Objekte sind mit einer ODER-Verknüpfung miteinander verbunden. Sobald eines der beiden erkannt wird, wird der Befehl zum Anhalten des Fahrzeuges gegeben.

Natürlich hätte auch ein selbst trainiertes Modell verwendet werden können. Die Idee, selbst ein Modell zu trainieren, ist kurz aufgekommen. Schlussendlich ist dieses Unterfangen aber nicht unternommen worden. Ausschlaggebend für diesen Entscheid war primär der enorme Zeitaufwand, ein eigenes Objekterkennungsmodell zu trainieren. Ein vortrainiertes Modell ist für unseren Anwendungsfall adäquat.

3.3.3.2 Implementierung der Objekterkennung

Um die maximale Performance herauszuholen, wird der rechenintensivste Teil der App auf C++ ausgelagert. Die konkrete Implementierung der Objekterkennung ist jedoch der einzige Teil der Applikation, welcher in C++ geschrieben worden ist. Sie wurde von einem anderen Projekt übernommen (nihui, 2021). Der Code ist auf unsere Bedürfnisse angepasst worden. Zum Beispiel ist die Objekterkennung nur auf Topfpflanze und Vase reduziert worden. Zusätzlich sind sämtliche grafischen Operationen (wie etwa das Zeichnen der Bounding-Box) im Code deaktiviert worden. Ausserdem ist eine Schnittstelle erstellt worden, um den Code mit dem Rest der App interoperabel zu machen.

Echtzeit-Objekterkennung zählt zu den rechenintensivsten Operationen (Zhang, 2020). In unserem Fall werden die Operationen auf dem CPU des Smartphones ausgeführt, wobei das Framework auch

GPU Berechnungen ermöglicht, (nihui, 2021). Tatsächlich ist aber die CPU-Variante wesentlich schneller. Das liegt daran, dass Grafikprozessoren eines Smartphones generell nicht besonders leistungsfähig sind, verglichen mit jenen, die in Desktop-Computern üblicherweise verbaut sind. Ausserdem ist das NCNN Framework in der Lage, die Arbeit auf mehrere Prozessorkerne auszulagern.

Programmiert ist dieser Teil des Codes mit dem Android Native Development Kit (NDK).
Dieser Kunstgriff ermöglicht es, nativen (C++ oder C) Code in einer Android App auszuführen. Google gibt folgende Empfehlung für Einsatzmöglichkeiten bezüglich NDK:

> "Squeeze extra performance out of a device to achieve low latency or run computationally intensive applications" (Get started with the NDK, 2020)

Ein weiterer guter Grund um C++ und NDK einzusetzen, spricht Google nicht an: Die Standardgrösse des Heaps ist für Android-Apps drastisch limitiert. Das gilt aber *nicht* für NDK. (C / C++ Code ist natürlich nicht an de JVM-Heap gebunden). Somit kann mit NDK diese Memory-Einschränkung umgangen werden.

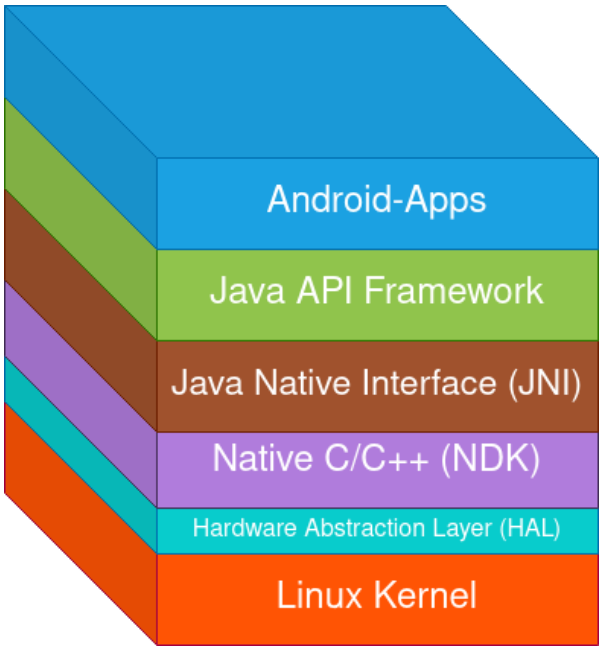


Abbildung 16: Schichten des Android Betriebssystems

Damit der C++ Code auf den Programmfluss der Android-App einwirken kann, muss er mit der Java-Schicht kommunizieren. Es braucht also eine Verbindung zwischen der C++ Schicht und Java API-Schicht. Das war ein durchaus nicht triviales Problem. Die Lösung dieser Knacknuss heisst JNI (Java Native Interface).
Mit JNI kann eine Verbindung zwischen diese beiden Schichten hergestellt werden. In Abbildung 16 ist diese verbindende Schicht JNI dargestellt.

Realisiert wurde eine sprachübergreifende, bidirektionale Kommunikation zwischen den erwähnten Schichten.

3.3.4 QR-Code

Um die QR-Codes zu dekodieren, wird die Library zxing verwendet.
Die Zusammenarbeit des `ImageAnalysis.Analyzer` von Android und der QR-Code Library führen zu sehr guten Leistungen. Damit kann die Android-App in Echtzeit Bildverarbeitungen realisieren. Sobald ein QR-Code gefunden wird, wird die entsprechende Callback Funktion aufgerufen.

```
        imageAnalyzer = ImageAnalysis.Builder()
            .setTargetRotation(rotation)
            .setTargetResolution(Size(1280, 720))
            .setBackpressureStrategy(
             ImageAnalsis.STRAEGY_KEEP_ONLY_LATEST)
            .build()
            .also {
                it.setAnalyzer(
                    cameraExecutor,
                    QRCodeImageAnalyzer(object : QRCodeFoundListener1 {
                        override fun onQRCodeFound(qrCode: String?) {
                            actionOnQrCode(qrCode)
                        }

                        override fun qrCodeNotFound() {
                            // not used
                        }
                    })
                )
            }

        try {
            camera = cameraProvider.bindToLifecycle(
                this, cameraSelector, preview, imageCapture,
imageAnalyzer)
            // Attach the viewfinder's surface provider
            preview?.setSurfaceProvider(
fragmentCamerBinding.previewView.surfaceProvider)
        } catch (exc: Exception) {
            Log.e(TAG, "Use case binding failed", exc)
        }
    }
```

Abbildung 17: QR-Code Analyzer

Dem Analyzer wird ein `cameraExecutor` als Parameter mitgegeben, welcher vom Typ `ExecutorService` ist. Das ist insofern relevant, weil damit die prozessorintensive Arbeit auf separate Threads ausgelagert wird.
Die erforderliche Bildauflösung wurde auf (1280 x 720) festgelegt. Obwohl die Kamera grundsätzlich eine höhere Auflösung von bis zu (1920 x 1080) unterstützt, ist das gar nicht notwendig.
Es ist ein Balanceakt zwischen Performance und Auflösung.

3.3.5 Webserver

Der Webserver wird eingesetzt, um den Fortschritt des Fahrzeugs im Verlauf des Parcours anzuzeigen. Dazu wurde eine schlichtes, leichtes Frontend geschrieben welches Informationen dynamisch aktualiseren kann.

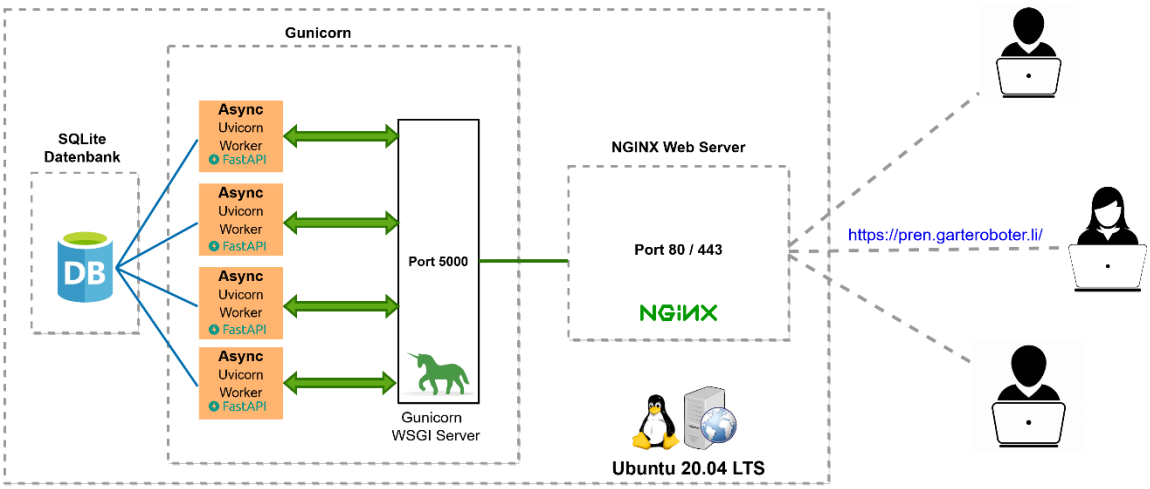


Abbildung 18 Systemarchitektur Webserver

Der Kern des Backends bildet eine FastAPI Webapplikation, welche hinter einem NGINX Server sitzt. Eine *blitzschnelle* Geschwindigkeit beim Laden der Seite (https://pren.garteroboter.li/) ist wünschenswert, denn es unterstützt ein optimales Nutzererlebnis und steigert die Besucherbindung. Niemand sollte warten, während die Website lädt. Geschwindigkeit ist folglich nicht fakultativ, sondern ein Anspruch.
Der Webserver besteht aus verschiedenen Schichten, wobei NGINX die äusserste Schicht bildet. Von dort werden eingehende Requests auf das Backend umgeleitet. NGINX ist nicht zwingend notwendig in dieser Konfiguration, es wird aber «dringend empfohlen» (Deploying Gunicorn, kein Datum).

Es laufen fortwährend mehrere Instanzen der Webapplikation parallel. Die parallele Verarbeitung wird auf Uvicorn-Worker Threads verteilt. Jeder einzelne Worker kann während seiner Lebensdauer viele Anfragen bearbeiten, wodurch der Overhead für die Erstellung und Beendigung von Prozessen pro Anfrage vermieden wird. Die Architektur der Webapplikation ist mit dem Hintergedanken von sehr hoher Geschwindigkeit konzipiert worden.

Die Datenbank steht vor der Herausforderung, dass zur gleichen Zeit mehrere Instanzen des Serverprogramms darauf zugreifen können. Grund dafür ist, dass es mehrere Threads gibt, die auf die Datenbank zugreifen. Das kann mit nicht blockierenden und asynchronen Ein- und Ausgabeoperationen (asynchronous and non-blocking IO) gelöst werden. In dieser Applikation wird SQLite zusammen mit SQLAlchemy (Async SQL (Relational) Databases, kein Datum) verwendet.

3.3.5.1 WebSockets

WebSockets ermöglichen eine bidirektionale Kommunikation zwischen Smartphone und Webserver. Das ermöglicht es, Statusupdates dynamisch zu aktualisieren.

Es kann also, ohne die Seite neu zu laden, eine Änderung an der Website vorgenommen werden. Die Website wird, während das Fahrzeug fährt, laufend aktualisiert.

Für das Aktualisieren vom Inhalt auf der Website wäre es rein technisch gesehen auch möglich, die Website periodisch neu zu laden. Diese Lösung ist aber deutlich weniger elegant als Websockets, weil bei einem full-page reload unnötig viele Daten über das Netzwerk transferiert werden.
Hier ist die Schlüsselerkenntnis, dass sich in jedem Statusupdate nur sehr wenig Inhalt ändert, bezogen auf die Menge des Inhaltes auf der Webseite.

3.3.6 Identifikation der Pflanzenspezies

Für die Identifikation der Spezies wird die API PlantNET verwendet. Der Dokumentation dieser API (My Pl@ntNet API, kein Datum) ist entsprechend gefolgt worden. Die Identifikation erfolgt asynchron in einem Background-Thread, sobald das Handy ein Foto gemacht hat. Die Requests sind mit Retrofit auf dem Handy realisiert worden. Retrofit ist eine typsichere Library für Android und Java, die sich grosser Beliebtheit erfreut. (Retrofit, kein Datum).

3.3.6.1 Bounding Box als Unterstützung

Die Wahrscheinlichkeit, dass die API eine Pflanze korrekt identifiziert, ist abhängig von der Qualität des Bildes. Um ein möglichst passendes Pflanzenbild für die API zu erhalten, wird das Pflanzenfoto in eine Vorverarbeitung eingespeist. Die Bounding-Box (Abbildung 15) wird aus dem Bild herausgeschnitten und daraus ein neues Bild erstellt, welches schlussendlich an den Webserver gesendet wird. Dies wird gemacht um sicherzugehen, dass möglichst nur die Pflanze auf dem Bild ist. Natürlich müssen wir damit rechnen, dass der QR-Code eventuell im Bild sein wird. Trotzdem ist es ratsam, möglichst nur die Pflanze selbst im Bild zu haben.

3.3.7 Quellcode

Der gesamte Quellcode war seit dem ersten Tag ein frei verfügbares Open Source Projekt. Gehostet ist es auf GitHub. Da es nur einen Entwickler gab, sind Zugriffsrechte und merges kein Thema gewesen.
Im Repository für die App sind mehrere branches erstellt, um zu experimentieren. Generell ist aber auf dem master branch entwickelt worden.
Den Quellcode der App und den der Website findet man unter den gegebenen Hyperlinks.

3.4 Betriebsanleitung

Um das PREN-Fahrzeug ordnungsgemäss bedienen zu können, befindet sich im Anhang eine Schritt-für-Schritt-Anleitung zur Handhabung sowie die notwendigen Vorkehrungen für den Transport (9.5 Betriebsanleitung).

4. Testen und Validieren

Um funktionale und nicht-funktionale Anforderungen zu testen, wurden in diesem Kapitel Testprotokolle erstellt und Tests ausgeführt. Die Tests wurden in die Disziplinen Mechanik, Elektronik und Informatik unterteilt.

4.1 Mechanik

4.1.1 Federung

Um die Federung zu testen, standen uns vier unterschiedliche Federn zur Verfügung. Alle jeweils mit einer unterschiedlichen Federkonstante. Aus dem PREN1 wurden Federn mit einer berechneten Federkonstante von 0.08 N/mm verwendet. Diese Federkonstante erwies sich jedoch unter maximalem Gewicht als zu klein und das Fahrzeug federte sehr stark ein. Deshalb wurden neue Federn mit den Federkonstanten 0.11, 0.19 und 0.25 N/mm bestellt.
Bei dem Test wurden alle vier Federvarianten auf das Fahrzeug montiert und es wurde die gleiche Strecke mit laufender Kamera abgefahren. Anschliessend wurde das Kameramaterial ausgewertet. Dies erwies sich als schwierig, da alle Federn zu sehr ähnlichen Resultaten mit kaum merkbaren Unterschieden führten. Uns gefiel jedoch das Einschwingverhalten der Feder mit der Federkonstante von 0.11 N/mm am besten, weshalb wir uns für diese entschieden (9.4.1.1).

4.1.2 Motorenleistung

Da wir in PREN1 unsere Motoren für optimistische 2 m/s ausgelegt haben, kam die Sorge auf, dass wir in den steilen Passagen des Parcours zu wenig Drehmoment zur Verfügung haben. Deshalb wollten wir testen, dass wir den Parcours mehrere male an der steilsten Stelle bewältigen können. Dies war jedoch kein Problem und wir hatten ausreichend Drehmoment zur Verfügung (9.4.1.3).

4.1.3 Wasserabweisung Kameraglas

Damit die Kamera ungestört aus dem Fahrzeug filmen und fotografieren kann, muss sichergestellt werden, dass kein Wasser an der Aussenabdeckung haften kann. Das Wasser verzerrt das Bild in einer Weise, so dass dieses unbrauchbar wird. Um das Wasser von der Oberfläche fernzuhalten, wird mit verschiedenen Nanobeschichtungen experimentiert. Unteranderem wird mit Imprägnierspray oder Autoglasversiegelung getestet.

Zu Beginn hatten wird vorgesehen, dass die Versiegelungen auf ein Stück Plexiglas aufgetragen werden. Jedoch stellte sich heraus, dass die Beschichtungen nur einen sehr geringen Effekt aufweisen. Der Verdacht, dass dieses Problem mit dem Material zusammenhängt, bestätigte sich, als wir dieselben Tests auf Glas ausführten. Auf dem Glas hatten wir viel bessere Ergebnisse erzielt und das Wasser kann viel besser abperlen. Schlussendlich erwies sich eine Glasscheibe mit einer Autoglasversiegelung als die beste Lösung (9.4.1.2).

4.1.4 Wasserdichtheit

Aus der Anforderung 4.14 und 4.15 geht hervor, dass das Fahrzeug schadenlos durch 5cm tiefes Wasser und im schlimmsten Fall eine 100m lange Strecke im Regen abfahren können soll. Um diesen Fall sicherzustellen, sind wir noch einen Schritt weiter. Wir haben in unseren Aufbau die Löcher und

Aussparungen zugeklebt und den ganzen Behälter mit Wasser gefüllt. Ganz nach dem Motto «kommt nichts raus, kommt auch nichts rein». Der Aufbau erwies sich als dicht (9.4.1.4).

4.2 Elektronik

4.2.1 Signalstärke

Bei der Spurhaltung innerhalb der Streckenführung ist es essenziell, dass möglichst die gesamte Bandbreite des Analog-zu-Digital-Wandlers (ADC) ausgenutzt werden kann. Dies ermöglicht es, auch bei grösserer Abweichung der idealen Fahrlinie noch ein nutzbares Signal zu empfangen und somit den Kurs zurück zur Ideallinie zu finden. Um einen gewissen anpassbaren Bereich für Korrekturen zu haben, wurde in die Rückkopplung der Verstärkung ein Potentiometer eingebaut. Getestet wurde mittels eines Signalgenerators, welcher an das signalführende Kabel angeschlossen wurde. Der maximale Ausgangspegel wurde am Signalgenerator anschliessend begrenzt und am Potentiometer anschliessend so justiert, dass bei kürzestem Abstand zwischen Kabel und Sensor ein Pegel von ca. 3.2V gemessen wird.

4.2.2 Buck-Converter

Damit die Spannung an den Motoren und somit die Drehzahl der Motoren variiert werden kann, wird ein Buck-Converter eingesetzt. Dieser ermöglicht es, mit einem bestimmten Duty-Cycle die Ausgangsspannung anzupassen. Der Buck-Converter wurde mit dem Oszilloskop erfolgreich auf sein lineares Verhalten getestet, was eine zuverlässige Steuerung der Motoren erlaubt.

4.3 Informatik

4.3.1 Anhalten

Die App erkennt Topfpflanzen in Echtzeit. Das Objekterkennungsmodell liefert einen Wert zurück, der als Wahrscheinlichkeit interpretiert werden kann. Für diesen Wert ist ein Schwellenwert von 50% programmiert worden. Das bedeutet, sobald ein Objekt mit 0.5 oder höherem Wert gefunden wurde, wird angehalten. Um anzuhalten wird eine Bluetooth-Nachricht an den ESP32 Mikrokontroller geschickt wird.
Es wurde festgestellt, dass bei der Maximalgeschwindigkeit das Fahrzeug an der Pflanze vorbeifährt. Des Weiteren hat sich herausgestellt, dass es aus Testgründen sehr praktisch ist, diesen Schwellwert von 0.5 auf dem Handy dynamisch zu kalibrieren.

4.3.2 Webclient

Um aktuelle Statusinformationen auf der Webseite anzuzeigen, werden Text und Bilddaten von der App auf den Webserver geschickt. Sobald diese angekommen sind, werden sie an alle Clients übertragen. Es wurden mehrere Ereignisse simuliert und um ein Testszenario zu erstellen. Der Webclient hat alle Szenarien erfolgreich umsetzen können, wie es erwartet wurde. Um die Übertragungsgeschwindigkeit der Bilder noch weiter zu verbessern, könnten diese zusätzlich noch mit dem WebP-Format komprimiert werden.

5. Schlussdiskussion

Der Projektimpuls entstand aufgrund des Auftrags, einen Garten-Roboter für den Garten- und Gartenbaubetrieb zu erstellen. Dieser dient in erster Linie dazu, Topfpflanzen zu erkennen. Mithilfe der Konzeption aus dem vorherigen Modul, Produktentwicklung 1, wurde das Garteroboter.li realisiert, getestet und optimiert. In dieser Arbeit wurde dokumentiert, wie das Garteroboter.li aufgebaut, angepasst und optimiert wurde, um ein möglichst gutes Resultat am Wettbewerb zu erhalten. Zuerst wurde der organisatorische Aspekt und der Projektplan definiert, damit die Situation klar ersichtlich wurde. Die Dokumentation der Realisierung wurde in die drei Disziplinen Mechanik, Elektronik und Informatik unterteilt. Daraufhin wurden Tests durchgeführt und anschliessend Optimierungen durchgeführt. Es wurde öfter festgestellt, dass die in der Produktentwicklung 1 erstellte Konzeption nicht immer optimal war und einige Anpassungen gemacht werden mussten. Einige Probefahrten zeigten auch Probleme auf, welche in der ursprünglichen Konzeption gar nicht bedacht wurden und im Nachhinein behoben werden mussten.

Es wurden schlussendlich mehrere Tests durchgeführt, um zu prüfen, ob das Garteroboter.li den Parcours bewältigen kann. Der Roboter konnte erfolgreich nach einem Start-Befehl dem Kabel folgen, die Pflanzen erkennen bzw. wiedererkennen, alle Events auf der Webseite aktualisieren und am Ziel wieder anhalten. Somit wurde jeder Aspekt der Aufgabenstellung dieses Projektes erfüllt und das Projekt war erfolgreich.

5.1 Lessons Learned

5.1.1 Mechanik

Der einfachste Ansatz ist der beste Ansatz

Dieser Leitsatz hat sich in unserem Maschinentechnik-Team schon von Beginn an schnell durchgesetzt. So haben wir auf ein komplexes Allradsystem verzichtet und haben uns für jeweils einen Motor pro Rad auf der Hinterachse entschieden. Auch auf dieser Hinterachse haben wir diesen Leitsatz weitergeführt. So haben wir nicht selbstständig ein komplexes Getriebe für die Umlenkung und Übersetzung des Antriebsmoments konstruiert, sondern haben uns für ein vormontiertes Getriebe auf einem Getriebemotor entschieden.
Das Motto wurde auf das ganze Fahrzeug angewandt. So ist der Ansatz auch in der Federung, der Lenkung und im Aufbau generell wiederzuerkennen.

Man muss das Rad nicht neu erfinden

So haben wir sprichwörtlich auch keine Räder selbst konstruiert, wie es sehr viele Teams getan haben, sondern wir haben die am einfachsten erhältlichen Räder gekauft, die unseren geplanten Abmessungen am nächsten kamen. Hier verbirgt sich auch einer unserer grössten Fehler. Da wir die Räder eingekauft hatten, konnten wir sie auch nicht anpassen. So schrieben die Raddimensionen mehr oder weniger unsere ganze Konstruktion vor. Eine Änderung der Raddurchmesser hätte auch die Änderung der gesamten Lenkgeometrie und der Fahrzeugabmessungen nach sich gezogen. Dies führte zu einer geringen Bodenfreiheit gegenüber anderen Teams. Jedoch ist es für unsere Lenkkonstruktion und die respektive Fahrzeuggrösse die optimale Dimension.
Bei der Lenkung haben wir ebenfalls auf einen automobil bewährten Ansatz gesetzt. So gehört die Achsschenkellenkung wohl zu den am weitest verbreiteten Lenksystemen. Dieses Konzept haben wir

übernommen. Es zeigt sich, dass auch dieser Entscheid richtig war, da sich diese Lenkart sehr einfach und schnell regeln lässt. Ausserdem erreichen wir dadurch ein sehr stabiles Fahrverhalten.

Im CAD passt alles besser

Zu dieser Erkenntnis kommt wohl jeder, der seine eigene Konstruktion auch selbst hergestellt hat. Die Toleranzen müssen grundsätzlich eher grösser gewählt werden, da die Fertigungsqualität der Realität nicht dem CAD entsprechen. Dies haben wir auch bei unserem Versuch erfahren, als wir versuchten unser Gehäuse selbst zu biegen, um der saftigen Rechnung von Blexon zu entgehen.
3D gedruckte Teile im STL Verfahren müssen ausserdem immer nachbearbeitet werden. Dies trifft umso mehr zu, wenn man beim STL-Export die „Angular Tolerance“ nicht auf 1 stellt und alle runden Geometrien plötzlich eckig gedruckt werden.
Mit diesen drei Leitsätzen haben wir die beiden PREN-Module gemeistert und werden sie wohl auch bei späteren Projekten weiter befolgen.

5.1.2 Elektronik

Magnetische Felder hat es überall

Unsere Wegerkennung basiert, wie bereits in der Arbeit mehrmals erläutert, auf der Detektion eines Magnetfeldes. Dieses wird mittels Funktionsgenerators erzeugt und anschliessend durch das signalführende Kabel abgestrahlt. Problematisch sind hierbei jedoch theoretisch alle anderen stromführenden Leitungen. Denn ein stromdurchflossener Leiter erzeugt immer ein Magnetfeld und dieses Magnetfeld kann, je nach Stärke, auch wieder von den Sensoren der Spurhaltung detektiert werden. Dies ist beispielsweise bei den Buck-Convertern unserer Motorensteuerung der Fall. Ist das Fahrzeug mit voller Drehzahl in Bewegung, fliesst durch die Spulen ein bis zu 9A hoher Strom. Dieser erzeugt dementsprechend ein starkes Magnetfeld, welches wiederum von den Sensoren erfasst wird. Im Gegensatz zum erwünschten und erzeugten 5kHz-Magnetfeld ist dieses zwar um einiges kleiner und somit vernachlässigbar. Wenn das Fahrzeug jedoch weit genug von der signalführenden Leitung entfernt ist, hat dieses plötzlich doch einen Einfluss und führt zu einem undefinierten Verhalten der Lenkung. Abhilfe hier schafft Abschirmung bzw. genug räumliche Trennung zwischen PCB und Sensoren oder mittels der Notfall-Lenkung der Software, die bei zu schwachen Magnetfeldern die Regelung übernimmt.

Hardware hui, Software pfui

Was in Anbetracht der Arbeit unseres Informatikers möglicherweise etwas weit hergeholt zu sein scheint, hat im Kern einen wahren Inhalt. Die Hardware-Komponenten des Fahrzeuges sowie des Senders waren schnell entwickelt und umgesetzt. Durch den relativ simplen Aufbau dieser Komponenten konnten sie früh auf Robustheit überprüft und optimiert werden. Da die Software sehr viel mehr Aufgaben meistern muss, kann diese oft nur schlecht in effektive Teilaufgaben unterteilt werden. So sind kleine Änderungen zwar schnell implementiert, jedoch benötigen diese oft viele Testzyklen bis das Gesamtsystem aus Hard- und Software harmoniert.
Im Grossen und Ganzen kann die Elektrotechnik-Gruppe jedoch auf eine erfolgreiche Zusammenarbeit innerhalb der Gruppe, sowie auch mit den anderen Gruppenmitgliedern zurückschauen. Probleme und Reibungspunkte gibt es und wird es immer geben, wie aber mit diesen umgegangen wird ist der springende Punkt. Und dies haben wir, zumindest aus Sicht der Elektrotechnik, erfolgreich gemeistert.

5.1.3 Informatik

Dieses Modul war sicherlich lehrreicher als fünf andere Module zusammen. Das gilt insbesondere für den Bereich der Android Entwicklung, aber auch für den Bereich der Web-Entwicklung. Bei der Entwicklung der App kam immer wieder die Herausforderung auf, gewisse Eigenschaften der Plattform zu umgehen. Apps sind explizit User-Interaktion basierende Anwendungen. Weil die App völlig autonom arbeiten muss (User-Interaktion ist undenkbar), kam des Öfteren die Herausforderung auf, den Programmfluss zu steuern, ohne dass eine Interaktion (wie etwa das Drücken eines Knopfes) stattfindet.

Entkopplung

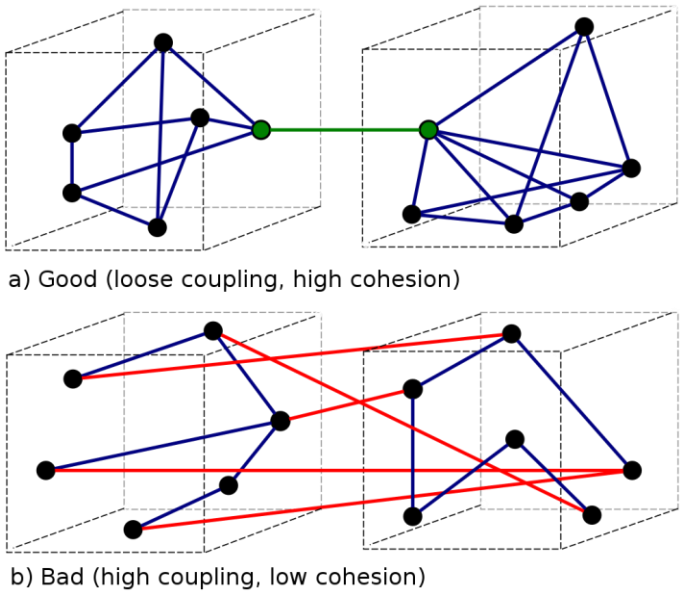


Abbildung 19: Kopplung (Coupling (computer programming), 2022)

Ein Grossteil der Softwarefunktionen konnten schon früh, völlig unabhängig vom Roboter getestet werden. Das war ein immenser Vorteil gegenüber anderen Teams, die häufig ihre Komponenten nur in Verbindung mit dem Roboter testen konnten. Das bedeutet, die einzelnen Komponenten sind fest miteinander gekoppelt. Bei unserer Architektur ist die Kopplung geringer, weil das Handy unabhängig vom Rest des Systems getestet werden kann.

Hier hat sich gezeigt, wie wichtig die Entkopplung für die Wartbarkeit und Testbarkeit eines komplexen Systems ist. Es war interessant, diese oftmals in Theorie behandelten Konzepte in der Praxis zu sehen.

Selbst die Bluetooth Kommunikation konnte ohne den Roboter getestet werden. Dazu wurde ein identischer Mikrokontroller am PC angeschlossen und schliesslich per Bluetooth mit dem Handy gekoppelt.

Informationsfluss zwischen C++ und Java Komponenten

Code aus verschiedenen Programmiersprachen zu verknüpfen, ist nicht immer ganz einfach. Hier standen wir vor dem Problem, dass die Objekterkennung (für maximale Performance) in C++ implementiert ist, die Kommunikation über Bluetooth aber auf der Android App läuft. Es ist also unausweichlich, eine schichtenübergreifende Kommunikation zu implementieren, die eine Schnittstelle über die Grenzen der Programmiersprachen bildet.

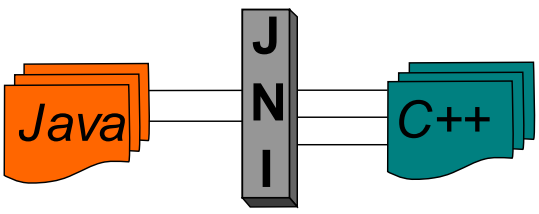


Abbildung 20: Schichtenübergreifende Kommunikation (Apertus, 2021)

Dank dem Java Native Interface (JNI) ist das möglich. Damit kann ein C++ Programm tatsächlich Java Funktionen aufrufen (und umgekehrt). Indirekt werden dadurch weitere Herausforderungen ausgelöst: Wie wird der Informationsfluss gesteuert? Wie interagieren die Objekte miteinander? Um den «Flow of Control» zu steuern, wurde oft das Observer-Pattern eingesetzt (Observer, S.293, 2003)

Android-Settings

Schon früh sind wir auf die Idee gekommen, der App eine Seite mit Einstellungen hinzuzufügen. Beim Testen hat sich als enorm praktisch erwiesen. Viele wichtige Parameter sind einfach über ein Menu einstellbar, anstatt hardcoded im Programm. So muss nicht jedes Mal das Projekt neu kompiliert werden, wenn man lediglich ein Wert anpassen möchte.

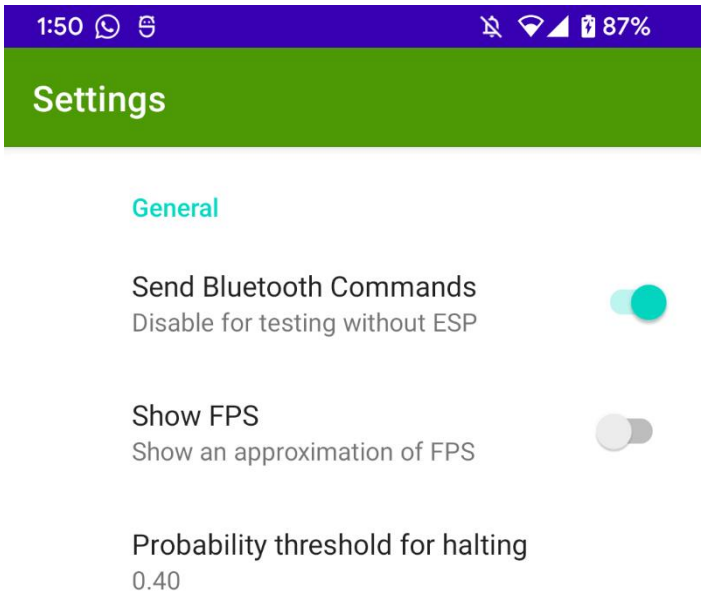


Abbildung 21: App-Einstellungen

Hyperparameter können in den App-Einstellungen angepasst werden. Ein gutes Beispiel dafür ist in Abbildung 21 beim «probability threshold for halting» zu sehen. Das ist ein Schwellwert für die Objekterkennung, welcher als Wahrscheinlichkeit interpretiert werden kann. In der App kann dieser Wert wahlweise im Bereich 0.4 bis 0.8 angepasst werden.

5.2 Ausblick

Mit dem Erfüllen der Aufgabenstellung ist das Projekt somit abgeschlossen und es ist kein Nachfolgeprojekt in Planung. Das Garteroboter.li wird noch in einem Wettbewerb gegen alle anderen Gruppen des Moduls Produktentwicklung 2 antreten, um die Performance mit den anderen Gartenrobotern zu vergleichen. Betriebe in der Gartenbranche können sich an dieser Dokumentation orientieren, um eine eigene Version eines Gartenroboters zu entwickeln.

6. Abbildungsverzeichnis

7. Tabellenverzeichnis

8. Literaturverzeichnis

Abwärtswandler. (20. Oktober 2021). Abgerufen am 01. April 2022 von Wikipedia: https://de.wikipedia.org/wiki/Abw%C3%A4rtswandler

Alternatives, Inspiration and Comparisons - FastAPI. (kein Datum). Von FastAPI: https://fastapi.tiangolo.com/alternatives/ abgerufen

Apertus. (2021). Von https://apertus.gitbook.io/vr/plugins-android/jni-plugin-android abgerufen

Async SQL (Relational) Databases. (kein Datum). Abgerufen am 06. Mai 2022 von FastAPI: https://fastapi.tiangolo.com/advanced/async-sql-databases/

Coupling (computer programming). (2022). Von https://upload.wikimedia.org/wikipedia/commons/0/09/CouplingVsCohesion.svg?download abgerufen

Deploying Gunicorn. (kein Datum). Abgerufen am 06. Mai 2022 von Gunicorn: https://docs.gunicorn.org/en/stable/deploy.html

Get started with the NDK. (30. September 2020). Abgerufen am 06. Mai 2022 von developers: https://developer.android.com/ndk/guides

Lobeck, P. D.-I.-I. (kein Datum). *Fertigungstechnik II.* Abgerufen am 01. April 2022 von Uni Due: https://www.uni-due.de/imperia/md/content/vip/wkzm_ss2014.pdf

My Pl@ntNet API. (kein Datum). Abgerufen am 06. Mai 2022 von Swagger: https://my-api.plantnet.org/#/

NanoDet. (27. Januar 2022). *RangiLyu/nanodet*. Abgerufen am 06. Mai 2022 von GitHub: https://github.com/RangiLyu/nanodet

nihui. (22. März 2021). *nihui/ncnn-android-nanodet*. Abgerufen am 06. Mai 2022 von GitHub: https://github.com/nihui/ncnn-android-nanodet

Observer, S.293. (2003). In E. Gamma, *Design Patterns: Elements of Reusable Object-Oriented Software.* Addison-Wesley, 2003. Print.

Retrofit. (kein Datum). Abgerufen am 06. Mai 2022 von Square GitHub: https://square.github.io/retrofit/

Schubladeneffekt. (24. Januar 2022). Abgerufen am 01. April 2022 von Wikipedia: https://de.wikipedia.org/wiki/Schubladeneffekt

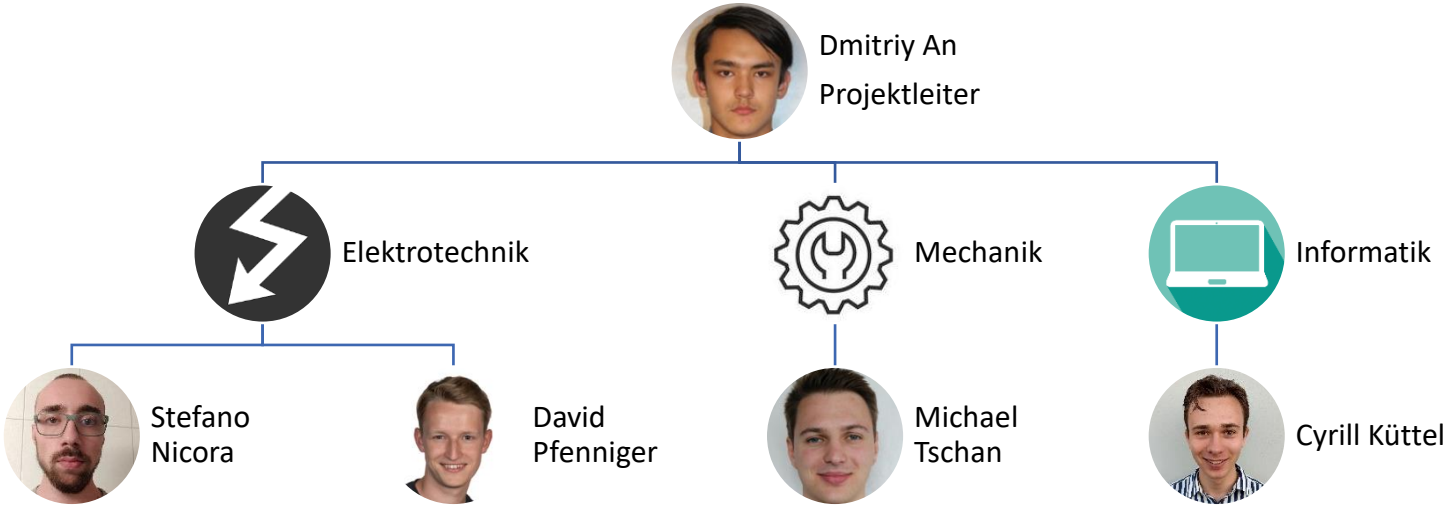
Tencent/ncnn. (06. Mai 2022). Abgerufen am 06. Mai 2022 von GithHub: https://github.com/Tencent/ncnn/

Welcome to NGINX Wiki! | NGINX. (kein Datum). Von NGINX: Alternatives, Inspiration and Comparisons - FastAPI abgerufen

Zhang, X. (29. Februar 2020). *SkyNet*. Abgerufen am 06. Mai 2022 von arxiv: https://arxiv.org/abs/1909.09709

9. Anhangsverzeichnis

9.1 Team-Organigramm Gruppe 38



Rolle – Name	Verantwortungsgebiet
Projektleiter – Dmitriy An Studiengang Digital Engineering	- Erstellt Projektplan - Anlaufstelle der Gruppe 38 - Schaut, dass Projektziele und Termingerechte Abgaben eingehalten werden - Ist die Sammelstelle aller Arbeiten und verantwortlich für die Dokumentation
IT-Experte – Cyrill Küttel Studiengang Informatik	- Ist verantwortlich für den Webserver - Ist verantwortlich für die Echtzeit-Objekterkennung - Ist verantwortlich für die Bluetooth-Kommunikation des Handys und ESP32.
Steuerungs-Experte – David Pfenniger Studiengang Elektrotechnik und Informationstechnologie	- Ist verantwortlich für die Verkabelung und Steuerung
Mechanik-Experte – Michael Tschan Studiengang Maschinentechnik	- Ist verantwortlich für den mechanischen Anteil der Arbeit - Ist verantwortlich für die Konstruktion
Elektrotechnik-Experte – Stefano Nicora Studiengang Elektrotechnik und Informationstechnologie	- Ist verantwortlich für den elektrotechnischen Aspekt der Arbeit

9.2 Kostentabelle

	Was?	Wer?	CHF
1	3D-Drucken	Michael Tschan	8
2	3D-Drucken	Stefano Nicora	6
3	Kleinmaterial	HSLU (1/2 Preis)	30
4	ESP32	David Pfenniger	18.1
5	Getriebemotoren	Oswald Zum Wald	41.9
6	Räder	Oswald Zum Wald	26.95
7	Lenkservo	HSLU (1/2 Preis)	10
8	4500mA LiPo Akku	HSLU (1/2 Preis)	17
9	3D-Drucken	Stefano Nicora	22
10	Domain Name	Cyrill Küttel	5
11	Google Pixel 3	Cyrill Küttel	60
12	Lüfter	HSLU	13.8
13	Spulen	HSLU	12.96
14	LM317	HSLU	4.25
15	PCB	HSLU	5
16	Dioden	HSLU	6.5
17	Div. Material Bau&Hobby	Stefano Nicora	19.35
18	Schalter, Taster Conrad	HSLU	14.15
19	3D-Drucken	Michael Tschan	20
21	Federn	Michael Tschan	47.93
22	Aufbewahrungsbox	HSLU	6.3
23	3D-Drucken	Stefano Nicora	29
			424.19

9.3 Risikoanalyse

Nummer	Risiko	Beschreibung	Massnahmen Eindämmen Schadenausmass	Massnahmen Eindämmen Eintrittswahrscheinlichkeit
1	**Übergreifende Themen**			
1.1	Fahrzeug fährt beim Start nicht los			Frühe Tests, Berechnungen, Programmierung kontrolliert
1.2	Finanzen - Geld geht aus			Budgetplanung und die Finanzen regelmässig kontrollieren
1.3	Löschen (aller) Daten		regelmässig Sicherungskopie erstellen und Verwendung einer Versionsverwaltung (Software)	
1.4	Krankheit / Quarantäne		Alle Daten auf Onedrive eintragen und Onlinekanäle wie Zoom oder Discord einrichten	Die BAG Regeln befolgen
1.5	Sonneneinstrahlung auf Kamera		mechanische Abdeckung	frühe Tests mit der Kamera
1.6	Zu lange Zeit für die Inbetriebnahme			Bei Tests die Zeit stoppen
1.7	Starker natürlicher Regenfall			Tests bei Bedingungen wie im Anforderungskatalog
1.8	Umgebungstemperatur zu hoch			Im Anforderungskatalog definiert

2	**Informatik**			
2.1	Software ist nicht rechtzeitig einsatzbereit			Sorgfältige Planung des Entwicklungsprozesses, regelmässige SOLL-IST-Vergleiche, Kommunikation zwischen IT und Projektplaner
2.2	Datenverarbeitung zu langsam		Früh testen	Übertragene Datengrösse anpassen
2.3	Pflanze und QR-Code wird nicht erkannt		Bearbeitung der Bildhelligkeit in Software oder Hardware	Bilderkennung für möglichst viele verschiedene Blickwinkel/Situationen testen
2.4	Zielbereich wird nicht erkannt		Software- oder Hardware-Nachbearbeitung des Kontrastes	Zielerkennung frühzeitig testen
2.5	Server hält den 200 gleichzeitigen Clients stand		Stresstests in der Testphase	Erhöhung der serverseitigen Rechenleistung

3	**Mechanik**			
3.1	Gerät nicht wasserdicht	Wasser kommt ins Gerät und zerstört die Elektronik	möglichst viele wasserdichte Elektronikbauteile benutzen, vor allem ausserhalb des Geräts und fortlaufend testen	Während der Planung die Normen nach Anforderungskatalog einhalten und alle Öffnungen wasserdicht verschliessen
3.2	Gefährt zu wenig robust	Teile fallen oder brechen während Parcours ab	Schon früh testen	Mit Sicherheiten berechnen und beim Umsetzen stabile Materialien verwenden.
3.3	Wendekreis kann nicht eingehalten werden		Simulation des Fahrverhaltens	Anpassung des Lenkwinkels / Verkürzung des Radstandes
3.4	Bodenkontakt	Das Antriebskonzept hat zu wenig Grip/Reibung und ist nicht wirklich fahrtauglich	Es wird eine andere Lösung genommen	Testen mit einem Prototyp vor der Lösungsauswahl
3.5	Gleichgewichtslage	Das Fahrzeug wird bei der Rampenüberfahrt instabil	Schwerpunkt möglichst tief halten	Testen mit einem Prototyp
3.6	Antrieb Funktion	Der Antrieb hat zu wenig Drehmoment und erreicht das Zeil nicht.	Es wird ein anderer Antrieb geplant und getestet	Testen mit einem Prototyp vor der Lösungsauswahl und richtige Planung des Antriebskonzepts / Berechnung der Motorenleistung
3.7	Antrieb Geschwindigkeit	Der Antrieb ist zu langsam, um den Parcours in 4 Minuten zu überwinden	Es wird ein anderer Antrieb geplant und getestet / Einen Motor kaufen, welcher mehr Leistung hat als ausgelegt.	Testen mit einem Prototyp vor der Lösungsauswahl und richtige Planung des Antriebs / Berechnung der Geschwindigkeit
3.8	Abrutschgefahr der Räder		Tests auf verschiedenen Untergründen	
3.9	Zu grosses Gefährt			Gute Planung, mit Einhaltung der Masse des Anforderungskatalogs
3.10	Gefährt zu schwer			Einbindungen des Gewichts im CAD und definiert im Anforderungskatalog

4	**Elektronik**			
4.1	Akku beginnt zu brennen		Physische Trennung zwischen Akku und den restlichen Komponenten, einfach austauschbar, Löschmassnahmen vorbereiten	Temperaturüberwachung, Sicherung
4.2	Kabel / Streckenführung wird nicht erkannt			Tests der Software / Der Erkennungselektronik
4.3	Steuercontroller ist defekt		Ersatzcontroller bestellen	
4.4	Akkus haben zu wenig Leistung			Gute Planung durch Berechnungen (Sicherheiten einrechnen) und fortlaufende Tests
4.5	EMV			Abdeckungen der kritischen Komponenten Motoren(treiber) vom Controller trennen / Pfaderkennungselektronik via EMF-Trennung von der restlichen Elektronik abschirmen
4.6	Akku geht kaputt		Mindestens 2 Akkus bestellen	Vor jedem Lauf Spannung messen
4.7	Motor geht kaputt		Ersatzantrieb / Notfalllösung vorhanden	
4.8	Sensor geht kaputt		Neuen Sensor bestellen	
4.9	Kommunikation funktioniert nicht (Schnittstellen)			Früh testen

9.4 Testprotokolle

9.4.1 Mechanik

9.4.1.1 Federung

Testbeschreibung

Bezeichnung	*Test Federung*
Beschreibung	*Um sicher zu stellen, dass mittels der Kamera auf der linken Seite die Umgebung analysiert werden kann (Anforderung 1.2), ist eine Federung / Dämpfung notwendig. Dazu werden verschiedene Kombinationen von Federlängen und Federraten getestet.*
Testschritte	*Die jeweiligen Federpakete werden auf alle vier Führungen montiert und das Fahrzeug fährt mit der laufenden Kamera über eine ruckelige Teststrecke. Danach wird das Videomaterial ausgewertet, um das beste Ergebnis zu bestimmen.*
Erwartetes Ergebnis	*Was erwarten wir vom Test (welche Ergebnisse/ Erkenntnisse)? Es wird erwartet, dass eine weiche Feder die beste Lösung sein wird.*

Testdurchführung

Testdatum	*19.05.2022*
Testort	*Horw*
Tester	*Michael Tschan*
Resultate	*Die zweitweichste Feder erzielte die besten Resultate.*
Abweichungen zur Erwartung	*Es wurde festgestellt, dass die zweitweichste (zweitkleinste Federkonstante) die besten Resultate erbracht hat, anstatt die weichste.*
Neue Erkenntnisse/ Risiken	*Auf einer ruckeligen Teststrecke, erwies sich, dass die harten Federn zu wenig Federung bieten und die weiche Feder zu viel federt und somit das Bild nicht so gut stabilisiert.*
Bemerkungen	-
Bilder/ Video	-

9.4.1.2 Kameraglas

Testfallbeschreibung

Bezeichnung	*Test Kameraglas*
Beschreibung	*Damit die Kamera ungestört aus dem Fahrzeug filmen und fotografieren kann, muss sichergestellt werden, dass kein Wasser an der Aussenabdeckung haften kann. Das Wasser verzerrt das Bild in einer Weise, so dass dieses unbrauchbar wird. Um Das Wasser von der Oberfläche fernzuhalten, wird mit verschiedenen Nanobeschichtungen experimentiert.* *Unteranderem wird getestet mit Imprägnierspray oder Autoglasversiegelung.*
Testschritte	*Es werden Testplättchen mit den verschiedenen Nanobeschichtungen beschichtet und mit Wasser bestäubt. Die Beschichtung mit dem besten Abperleffekt wird verwendet.*
Erwartetes Ergebnis	*Wird finden eine Beschichtung, welche unseren Anforderungen genügt. Favorit ist die Autoglasbeschichtung.*

Testdurchführung

Testdatum	*19.05.2022*
Testort	*Horw*
Tester	*Michael Tschan*
Resultate	*Auf der Autoglasversiegelung bleiben weniger und kleinere Tropfen am Glas.*
Abweichungen zur Erwartung	*-*
Neue Erkenntnisse/ Risiken	*Die Unterschiede sind nur schwer erkennbar auf kleinen Glasflächen.*
Bemerkungen	*-*
Bilder/ Video	-

9.4.1.3 Motorenleistung

Testfallbeschreibung

Bezeichnung	*Leistung Motoren*
Beschreibung	*Aus der Anforderung 4.17 geht hervor, dass das Fahrzeug eine Steigung von 25% absolvieren können muss. Dies wird auf der vorgegebenen Strecke auf den verschiedenen Untergründen getestet.*
Testschritte	*Das Fahrzeug muss den steilsten Teil der vorgegebenen Strecke 3-mal nacheinander bewältigen müssen.*
Erwartetes Ergebnis	*Aus den Berechnungen weisen die Motoren ein doppelt so grosses Drehmoment auf, wie benötigt wird.* *Wir erwarten folglich keine Probleme*

Testdurchführung

Testdatum	*19.05.2022*
Testort	*Horw*
Tester	*Michael Tschan*
Resultate	*3x Erfolgreiche Überwindung vom steilten Teil der Strecke*
Abweichungen zur Erwartung	*-*
Neue Erkenntnisse/ Risiken	*-*
Bemerkungen	*-*
Bilder/ Video	-

9.4.1.4 Wasserdichte

Testfallbeschreibung

Bezeichnung	*Wasserdichte*
Beschreibung	*Aus der Anforderung 4.14 und 4.15 geht hervor, dass das Fahrzeug schadenlos durch 5cm tiefes Wasser fahren und eine 100m lange Strecke im Regen abfahren können soll. Getestet wird dies auf der zur Verfügung stehender Strecke und an einem Tag mit Regen. Falls kein Regen vorhanden sein sollte, werden wir selbst nachhelfen.*
Testschritte	*Einerseits wird das Fahrzeug durch das 5cm tiefe Wasser auf der Strecke gefahren. An einem regnerischen Tag wird der Schutz gegen Regen zuerst ohne die Elektronik und dann mit der Elektronik getestet.*
Erwartetes Ergebnis	*Es soll kein Wasser in das Gehäuse der Elektronik und der Motoren gelangen.*

Testdurchführung

Testdatum	*19.05.2022*
Testort	*Horw*
Tester	*Michael Tschan*
Resultate	*Das Gehäuse ist gemäss Anforderungen wasserdicht*
Abweichungen zur Erwartung	*-*
Neue Erkenntnisse/ Risiken	*-*
Bemerkungen	*-*
Bilder/ Video	-

9.4.2 Elektronik

9.4.2.1 Signalstärke

Testfallbeschreibung

Bezeichnung	*Signalstärke am Empfänger*
Beschreibung	*Um der Streckenführung effektiv folgen zu können, muss die Signalstärke jederzeit einen messbaren Pegel aufweisen.* *Der Empfänger wird in Verbindung mit dem eignen Sender getestet und abgestimmt.* *Voraussetzungen:* • *Verlegtes Kabel* • *5kHz Sender mit 320V-Anschluss* • *PREN-Fahrzeug mit montierten Sensoren*
Testschritte	*Das Fahrzeug wird mittig über das signalführende Kabel gestellt. Die Potentiometer der Verstärkerschaltung werden zu Beginn in eine Mittelposition gestellt und der verstärkte Signalpegel gemessen. Anschliessend werden die Pegel mittels der beiden Potentiometer soweit aufeinander abgestimmt, dass diese einen Wert von ca. 3.2V aufweisen.*
Erwartetes Ergebnis	*Die Verstärkerschaltung des Empfängers ist auf das Eingangssignal so abgestimmt, dass bei neutraler Fahrposition (das signalführende Kabel befindet sich mittig zwischen den beiden Sensoren) beide empfangene Signale auf ca. 3.2V verstärkt werden.*

Testdurchführung

Testdatum	*20.05.2022*
Testort	*Horw*
Tester	*Stefano Nicora & David Pfenniger*
Resultate	*In der neutralen Fahrposition beträgt der verstärkte Pegel rund 3.2V.*
Abweichungen zur Erwartung	*Da die Verstärkerschaltung über eine genug hohe Verstärkung verfügt, wäre auch ein Pegel von >3.2V möglich. Da der Analog-zu-Digital-Wandler jedoch mit bis maximal 3.3V betrieben werden kann, macht es keinen Sinn diesen mit mehr Pegel zu versorgen. Dadurch ergaben sich auch keine Abweichungen zu den Erwartungen.*
Neue Erkenntnisse/ Risiken	*Je höher die am Operationsverstärker anliegende Signalfrequenz, desto kleiner wird die maximal verfügbare Verstärkung. Diese Thematik wird auch "Gain-Bandwith-Product" genannt.*
Bemerkungen	*-*
Bilder/ Video	-

9.4.2.2 Buck-Converter

Testfallbeschreibung

Bezeichnung	*Buck-Converter*
Beschreibung	*Der Buck-Converter soll bei einer bestimmten Eingangsspannung eine kleinere Ausgangsspannung herausgeben in Abhängigkeit vom Duty-Cycle.*
Testschritte	*Es werden verschiedene Spannungen an den Buck-Converter angelegt und mittels eines Oszilloskops wird der Spannungsverlauf aufgezeichnet.*
Erwartetes Ergebnis	*Der Buck-Converter soll eine lineare Zu-/Abnahme abhängig vom Duty-Cycle ausweisen.*

Testdurchführung

Testdatum	*20.05.2022*
Testort	*Horw*
Tester	*Stefano Nicora & David Pfenniger*
Resultate	*Der Buck-Converter zeigt ein stark lineares Verhalten in Abhängigkeit vom Duty-Cycle auf.*
Abweichungen zur Erwartung	-
Neue Erkenntnisse/ Risiken	-
Bemerkungen	-
Bilder/ Video	-

9.4.3 Informatik

9.4.3.1 Anhalten

Testfallbeschreibung

Bezeichnung	*Anhalten*
Beschreibung	*Das Fahrzeug soll anhalten, wenn es eine Topfpflanze erkennt. Das Anhalten wird über eine Bluetooth Nachricht realisiert. Diese Nachricht wird vom Handy auf den Mikrokontroller geschickt, welcher den Befehl umsetzt.* *Das bedingt, dass das Handy mit dem ESP32 gekoppelt ist. Ausserdem muss die Bluetooth-Schnittstelle mit dem ESP32 implementiert sein.*
Testschritte	*Eine Probefahrt auf der vorgegebenen Strecke wird durchgeführt, um zu prüfen, ob das Fahrzeug vor einer Pflanze anhalten kann.* *1) Das Fahrzeug wird in die Startposition gebracht.* *2) Der Startbefehl wird gegeben.* *3) Es gibt eine Topfpflanze auf dem Weg.*
Erwartetes Ergebnis	*Das Fahrzeug hält an, weil der Stopp Befehl über Bluetooth gesendet wird.*

Testdurchführung

Testdatum	*19.05.2022*
Testort	*Horw*
Tester	*Cyrill Küttel*
Resultate	*Angehalten*
Abweichungen zur Erwartung	*-*
Neue Erkenntnisse/ Risiken	*Bei der Maximalgeschwindigkeit ist der Gartenroboter teilweise an der Pflanze vorbeigefahren.* *Der Bremsweg muss miteinberechnet werden. Dieser ist natürlich abhängig von der Geschwindigkeit des Fahrzeugs.* *Ausserdem muss softwareseitig verhindert werden, dass dieselbe Pflanze gleich mehrmals erkannt wird. Eine simple Lösung dafür wäre es, ein Timer zu programmieren, welcher die Objekterkennung bei einem positiven Resultat für einige Sekunden deaktiviert.*
Bemerkungen	*-*
Bilder/ Video	-

9.4.3.2 Webclient

Testfallbeschreibung

Bezeichnung	*Webclient*
Beschreibung	*Um aktuelle Statusinformationen auf der Webseite anzuzeigen, werden Text und Bilddaten vom App auf den Webserver geschickt. Sobald diese angekommen sind, werden sie an alle Clients übertragen.*
Testschritte	*Es werden mehrere Ereignisse simuliert und beobachtet, ob die Daten den Webclient erreichen.*
Erwartetes Ergebnis	*Der Webserver bekommt die Daten und speichert diese in einer Datenbank. Ausserdem werden die eingegangenen Daten an alle Clients verschickt. (Clients sind grundsätzlich Besucher der Website.) Die Informationen werden auf der Webseite angezeigt, ohne dass ein full page refresh nötig ist.*

Testdurchführung

Testdatum	*02.02.2022*
Testort	*Zuhause*
Tester	*Cyrill Küttel*
Resultate	*Der Webclient erfüllte alle Anforderungen.*
Abweichungen zur Erwartung	*-*
Neue Erkenntnisse/ Risiken	*Um die Übertragungsgeschwindigkeit der Bilder noch zu verbessern, können die Bilder auf dem Webserver noch komprimiert werden. Das WebP Format bietet sich an.*
Bemerkungen	*Für die Bild- und Textdaten wird jeweils eine separate WebSocket Verbindung aufgebaut. Das hat den Effekt, dass der Serverseitige Code simpler ist. Die Unterscheidung erfolgt über eine vordefinierte ID.*
Bilder/ Video	https://pren.garteroboter.li/websocketTest

9.5 Betriebsanleitung

9.5.1 Inbetriebnahme

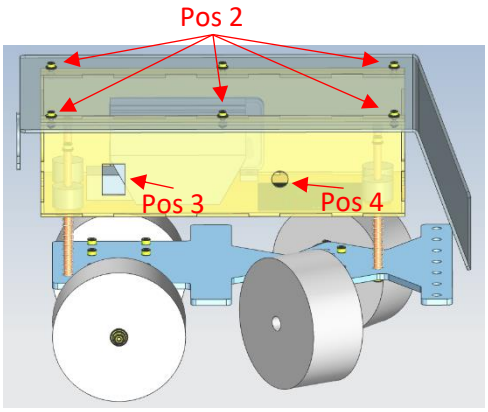
Material

- 6x M4x20 Schrauben (für Deckel)
- 6x M4 Muttern (für Deckel)
- Inbus Steckschlüsselsatz
- Abdichtband zur zusätzlichen Abdichtung zwischen Deckel und Gehäuse
- 14.9V Akku
- Signalgenerator inkl. Gerätenetzkabel

Vorgehen Signalgenerator

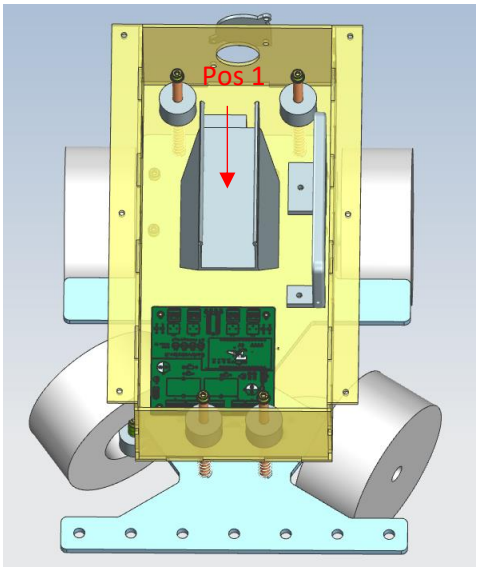
1. Signalgenerator an 230V-Netz anschliessen
2. Einen Leiter des Kabels (Farbe egal) an beiden Enden an den Signalgenerator anschliessen

Vorgehen Fahrzeug



1. Deckel (falls montiert) mittels Inbus-Schlüssel entfernen
2. Akku in die Akku-Halterung (Pos 1) einlegen (mit den Kabelenden in Richtung des Lüfters)
3. 2-poliger Akkustecker an das vorhandene Gegenstück anschliessen
4. Smartphone einschalten
5. Internet und Bluetooth auf dem Smartphone aktivieren
6. Einstellung “Auto-Rotate” deaktivieren
7. App starten, auf den Start Button drücken im Hauptmenu.
8. Deckel mittels Schrauben (Pos 2) und Muttern verschliessen
9. Fahrzeug mittels Schalter (Pos 3) einschalten und möglichst zentriert auf das signalführende Kabel platzieren
10. Nach dem Startsignal Taster (Pos 4) betätigen und den Fahrbereich verlassen

9.5.2 Ausserbetriebnahme



Material

- Inbus Steckschlüsselsatz

Vorgehen Signalgenerator

1. Signalgenerator vom 230V-Netz trennen
2. Litzen des Kabels ausziehen

Vorgehen Fahrzeug

1. Fahrzeug mittels Schalter (Pos 3) ausschalten
2. Deckel mittels Inbus-Schlüssel entfernen
3. Akku vom 2-poligen Akkustecker trennen und aus der Halterung entfernen

9.5.3 Transport

Material

- Transportbox Akku
- Transportbox Fahrzeug

Der Akku muss zwingen immer ausserhalb des Fahrzeuges mitgeführt werden. Lithium-Polymer-Akkus weisen eine hohe Energiedichte auf und können bei Brand nicht gelöscht werden.