



MOHNE – MOBILE DROHNE

Abschlussbericht

Entwicklung eines autonomen Fahrzeuges für extraterrestrische
Erkundungsmissionen

Eingereicht bei:

Fachhochschule Kufstein Tirol Bildungs GmbH

Studiengang Smart Products & Solutions

Verfasser:

Christian Gruber

Abgabedatum:

11.03.2018

TABLE OF CONTENTS

List of Figures	V
List of Tables	VI
List of Abbreviation	VII
Abstract	VIII
1. System Beschreibung	1
1.1 Mechanisches Konzept	1
1.1.1 Flugmodus	1
1.1.2 Fahrmodus	3
1.1.3 Funktionsprinzip der Propeller	3
1.2 Sensing Concept	4
1.2.1 Picamera	5
1.2.2 Ultraschallabstandssensor	6
1.2.3 Gesamtsystem (big picture)	6
2. Materialien und Methoden	7
2.1 Materialien	8
2.2 Manufacturing Technologies	8
2.3 Software Tools	10
2.3.1 Autodesk Inventor Professional	11
2.3.2 Autodesk EAGLE	12
2.3.3 Arduino IDE1	13
2.3.4 Raspbian	13
2.3.5 Python	14
2.3.6 Samba	15
3. System Life Cycle	16
3.1 System requirements	16
3.1.1 Purpose	16
3.1.2 Goal	16
3.1.3 Limitations	17

Switching on and off takes place in manual mode, while flying and driving, the "Mohne" acts autonomously. The distance to the ground is not measured. Obstacles are overcome by switching from the driving mode to the flying mode and then returning to the driving mode.....	17
The system only must work in good weather conditions and on given courses, which is why only one straight-ahead flight is realized.	17
3.2 Design alternatives.....	17
3.3 Model the System	18
3.3.1 System and Subsystem.....	18
3.3.2 Make or Buy	19
3.3.3 System Architecture.....	20
3.3.4 Functional Decomposition	20
3.3.5 Failure Mode and Effects Analysis (FMEA)	20
3.4 Integrate System Components	22
3.5 Launch the System.....	23
3.5.1 Timeline	23
3.5.2 Budget.....	24
3.6 Assess Performance	25
3.7 Results, Conclusion and Recommendations	26
3.7.1 Results.....	26
3.7.2 Conclusion	26
3.7.3 Recommendations.....	27
4. Diagrams.....	28
4.1 Use Case Diagram (Anwendungsfalldiagramm)	28
4.2 Structural Model (Strukturmodelle).....	29
4.3 Interaction Model (Interaktionsmodell).....	30
4.4 Behavior Model (Verhaltensmodell)	31
5. Version Control.....	31
6. Design of PCB and Electrics	32
6.1 Electrics	32
6.2 Schematic.....	33

6.3	Board.....	33
7.	Lessons Learned	35
	Anhang.....	36

LIST OF FIGURES

Abbildung 1: Quadrocopter: Motordrehrichtung.....	2
Abbildung 2: Quadrocopter: Vorwärtsbewegung.....	2
Abbildung 3: Quadrocopter: Rückwärtsbewegung.....	2
Abbildung 4: Tragfläche mit Propeller: Verstellbar um +90° und -90°	3
Abbildung 5: Quadrocopter: Mechanisches Konzept.....	4
Abbildung 6: Bild zum Anlernen von Formen und Farben	5
Abbildung 7: Ultraschallsensor (USS)	6
Abbildung 8: Wirkprinzip USS	6
Abbildung 9: Quadrocopter: Sensorkonzept.....	7
Abbildung 10: Fertiges CAD Model der "Mohne"	11
Abbildung 11: High level overview systems and subsystems	18
Abbildung 12: System architecture and communication	20
Abbildung 13: System integration – not used and critical parts	22
Abbildung 14: Milestone plan	23
Abbildung 15: Use case diagramm "Mohne"	28
Abbildung 16: Komponentenmodell "Mohne"	29
Abbildung 17: Sequenzdiagramm "Mohne"	30
Abbildung 18: Aktivitätsdiagramm "Mohne"	31
Abbildung 19: Verkabelung der elektrischen Komponenten	32
Abbildung 20: Schema des ersten PDB	33
Abbildung 21: Eagle Konstruktion des ersen PDB	34
Abbildung 22: Defektes/durchgebranntes PDB.....	35

LIST OF TABLES

Tabelle 1: Verfügbare Materialien.....	8
Tabelle 2: Verfügbare Maschinen und Werkzeuge	8
Tabelle 3: Stückliste des CAD Modells.....	12
Tabelle 4: Make or Buy table	19
Tabelle 5: Function table	20
Tabelle 6: Failure mode and effects analysis matrix	21
Tabelle 7: Planed tasks and responsibilities	23
Tabelle 8: Budget planning of "Mohne"	24
Tabelle 9: Tracks to be overcome.....	25

LIST OF ABBREVIATION

BDI	Bundesverband der Deutschen Industrie e.V.
CAD	Computer aided design
CEO	Chief Executive Officer
DBT	Digital Business Transformation
EY	Ernst & Young
F&E	Forschung und Entwicklung
GM	Geschäftsmodell
GMI	Geschäftsmodell-Innovation
HR	Human Resources
IKT	Informations- und Kommunikationstechnik
IoT	Internet of Things
IT	Informationstechnik
KPI	Key Performance Indicator
MIT	Massachusetts Institute of Technologie
PDB	Power distribution board
PLA	Polylactide
ROI	Return on Investment
USS	Ultrasonic Sensor

ABSTRACT

In this project our team had an easy mission. Build a vehicle that can be transported to the mars and move there autonomous. Just kidding, it was obviously not an easy mission. You need to know exactly about your resources, know-how and a huge amount of planning.

To get the breath of a chance we setup concrete roles, made an overall plan where we already saw, our goal of an autonomous drone was hard to reach within time. We decided to mix traditional and agile project management methods. For the overall planning we setup a milestone plan. The milestones correlated with our agile sprint rota. Within the single project phases, we accomplished our tasks agile step by step. Therefore, we used the Kanban board on github. We tried only to setup tasks that can be done within one sprint.

We thought about several concepts, but we were focused on the task, that the vehicle must get over a 60cm high obstacle, although the vehicle may only be 60cm long. After a few “round table meeting”, we decided to build a hybrid drone, which can drive and fly, because flying could be the easiest way to get over an obstacle.

After the drone was constructed with CAD, we decided to use acrylic glass and 3D printed PLA as main material. We invented a new mechanism, so that the rear motor mountings could be turned with two servos. With this mechanism we changed the moving modus between flying and driving. For power distribution we made a tiny single board to connect all Controller with the accumulator. As controller we needed the 4 ESCs for the motors, a Raspberry Pi as main unit and an Arduino Uno as fast control unit e.g. for PID regulation.

From the five virtual tracks we had to accomplished, we could only overcome successful two. So, you can’t really say our mission to mars was successful at this point.

But we laid the foundation for the next project of autonomous driving vehicles and even more important our team learned a lot. We learned totally new technical aspects, like aviation and what’s necessary to actual fly. Also, we learned much about project management for huge projects within no time... that’s not possible within an acceptable quality. Although we used our methods quite well, we just couldn’t achieve the huge scope we’ve set ourselves in this limited time. If we have to bring out the most important lesson learned, then we learned “don’t decide emotionally that you want to do something absolutely, if all rational measurements say no!”

1. SYSTEM BESCHREIBUNG

1.1 Mechanisches Konzept

Die Mohne ist ein sogenannter „Hybrid-Quadrocopter“, der sowohl zum Fahren als auch zum Fliegen konzipiert ist. Durch diese Hybrid-Nutzung ergeben sich große Vorteile. Auf dem Boden spart der Roboter Energie (im Verhältnis zur Fortbewegung in der Luft), während er in der Luft auch meterhohe Hindernisse überwinden kann. Nachfolgend wird das mechanische Konzept beider Varianten näher erläutert sowie und in Abbildung 5 veranschaulicht.

1.1.1 Flugmodus

Die Mohne gehört zur Gruppe der Quadrocopter, bestehend aus vier Motoren die in H-Konfiguration angeordnet sind. Bei der H-Konfiguration sind die vier Motoren in 45° zur Flugrichtung versetzt, wobei der Abstand vom Kreuzmittelpunkt zu den einzelnen Motoren immer gleich groß ist.

Vier Propeller sind fest an den Motoren montiert und erzeugen den Auftrieb der Drohne, welcher sich über die Drehzahl der einzelnen Motoren verändern lässt. Übersteigt die durch die sich drehenden Propeller erzeugte Auftriebskraft die Anziehungskraft der Erde bzw. des Mars, so beginnt die Drohne zu schweben. Eine Erhöhung bzw. Verringerung der Drehzahl hat ein Steigen bzw. ein Sinken der Drohne zur Folge.

Dadurch ergibt sich ein relativ einfacher mechanischer Aufbau, in dem es außer den Rotoren der Motoren keine weiteren beweglichen Teile für den Flugmodus gibt.

Überkreuz liegende Rotoren drehen dabei in die gleiche Richtung (Abbildung 1) und heben somit das um die Mitte der Drohne übertragene Drehmomente auf.

Bei exakt gleicher Drehgeschwindigkeit aller Propeller und keinen äußeren Störeinflüssen, steht die Drohne also exakt lotrecht über dem Boden. Dies eliminiert die Notwendigkeit einer Gierstabilisierung. Um ein nach vorne fliegen zu ermöglichen, drehen sich die vorderen Rotoren langsamer als die hinteren, was ein nach vorne abkippen der Drohne zur Folge hat (Abbildung 2). Entsprechend wird ein nach hinten fliegen ermöglicht, indem die hinteren Rotoren langsamer als die vorderen drehen (Abbildung 3).

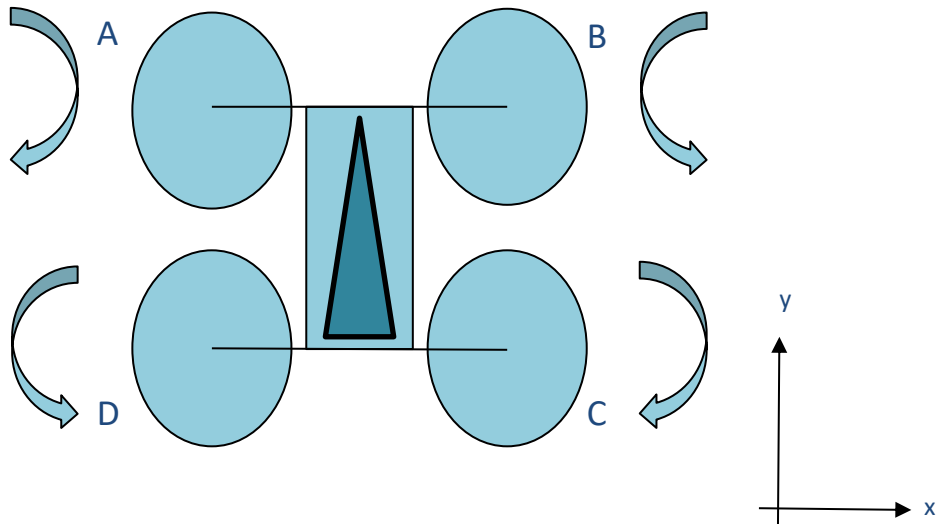


Abbildung 1: Quadrocopter: Motordrehrichtung

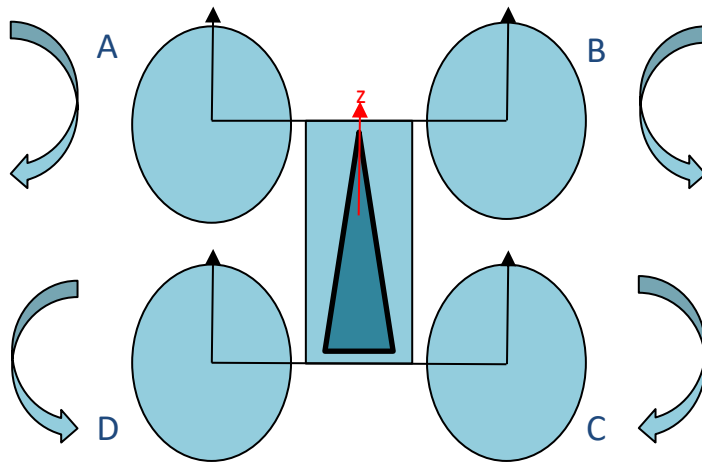


Abbildung 2: Quadrocopter: Vorwärtsbewegung

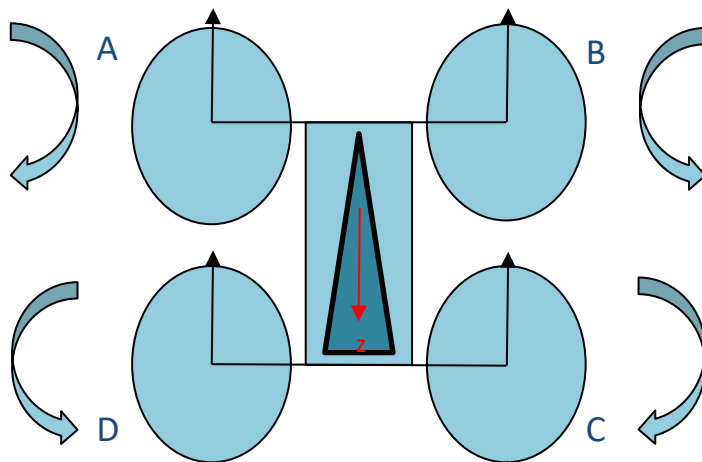


Abbildung 3: Quadrocopter: Rückwärtsbewegung

Wichtig für das Verständnis des Funktionsprinzips der Drohne ist, dass man sich grob phänomenologisch die Wirkungsweise von Propellern veranschaulicht, weil sich schon aus recht simplen Überlegungen Hinweise darauf ergeben, wie die Propeller sinnvollerweise ausgelegt werden müssen. Die Wirkungsweise eines Propellers, dessen Aufgabe sowohl im Flug- als auch im Fahrmodus darin besteht eine Drehleistung von $2\pi Qn$ optimal in eine Schubleistung T_{v_A} umzusetzen, kann aus zwei Anschauungen gewonnen werden, die dann in jeweils unterschiedlichen Berechnungsmodellen münden:

- **Impulsbetrachtung:** Um einen Schub T nach vorne bzw. nach oben zu erzeugen, muss der Propeller die Luft nach hinten bzw. nach unten beschleunigen. Die Änderung des Impulses entspricht dabei dem Propellerschub. Die Theorie, die auf diesem Modell aufbaut, ist die Strahltheorie.
- **Tragflügelbetrachtung:** Die Wirkungsweise eines Propellers entspricht der eines Tragflügels. Der Auftrieb entspricht dabei dem Schub, der Widerstand dem Moment. Das Tragflügelmodell erklärt die Propellerwirkung an den einzelnen Flügelschnitten und ist als Traglinientheorie (nur radiale Diskretisierung) oder Tragflächentheorie bekannt.

Aus diesen beiden Wirkungsweisen lassen sich schon einige grundlegende Dinge qualitativ ableiten, ohne dass genaue mathematische Begründungen nötig sind. Die Auslegung der Propeller erfolgt in Kapitel ...???

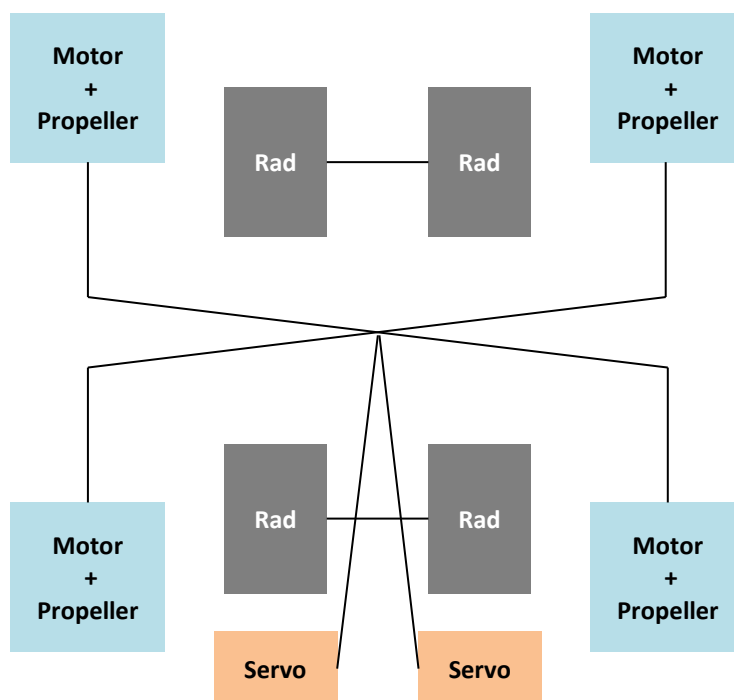


Abbildung 5: Quadcopter: Mechanisches Konzept

1.2 Sensing Concept

Um die Mohnen autonom fliegen bzw. fahren zu lassen und Hindernisse und Ziele zu erkennen, wurde eine Kamera und ein Ultraschallsensor verbaut. Nachfolgend werden beide Sensoren näher beschrieben und das System im Gesamtkontext betrachtet.

1.2.1 Picamera

Mit der Kamera sollen vorher definierte Formen und Farben erkannt werden. Die Rückgabewerte an den Raspberry Pi sollen in der Logik die vorgesehene Funktion auslösen. Beispiele hierfür sind Markierte Hindernisse und das Ziel. Anbei ein Auszug an angelernten Formen und Farben, hier ist auch das vorgegebene rote Quadrat enthalten. Die Exakte Größe hat für uns keine Rolle gespielt, da wir in der Logik Ecken und Kanten zählen haben lassen und Farben erkennen.

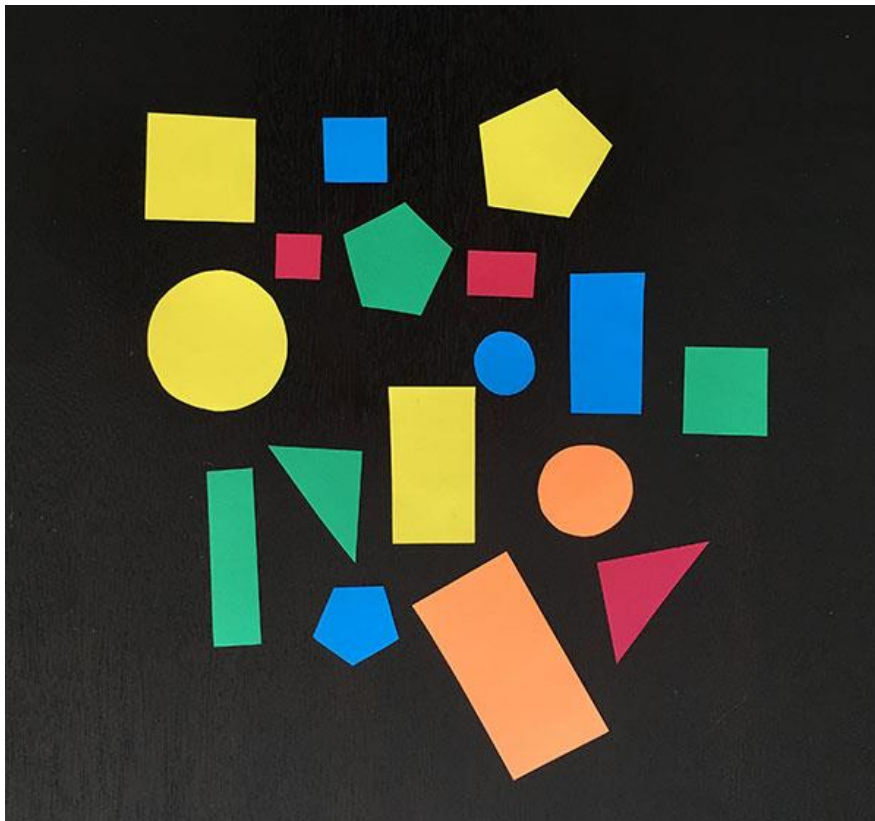


Abbildung 6: Bild zum Anlernen von Formen und Farben

1.2.2 Ultraschallabstandssensor



Abbildung 7: Ultraschallsensor (USS)

Mit dem Ultraschallabstandssensor Abbildung 7: Ultraschallsensor (USS) kann man mittels eines Ultraschalllautsprechers und eines Mikrofons den Abstand berührungslos zu einem Objekt messen. Das Prinzip Abbildung 8: Wirkprinzip USS basiert darauf, dass die Schallgeschwindigkeit in der Luft bei gleichbleibender Temperatur nahezu konstant bleibt - bei 20°C beträgt sie 343,2m/s. Aus diesem Fakt kann man die Abstandsmessung in eine Zeitmessung überführen, welche dann von Mikrocontrollern einfach übernommen werden kann.

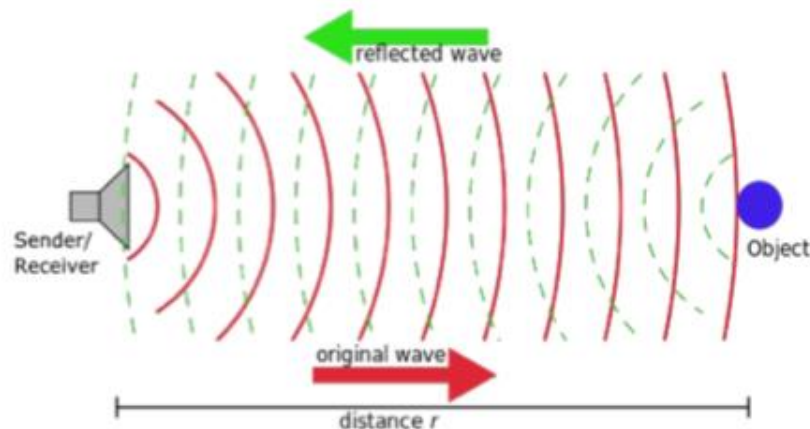


Abbildung 8: Wirkprinzip USS

1.2.3 Gesamtsystem (big picture)

Die unter Punkt 1.2.1 und 1.2.2 beschriebenen Sensoren unterstützen den Hybrid-Quadrocopter sich in seiner Umgebung zurechtzufinden und sich autonom fortzubewegen, Ziele zu erkennen und Hindernisse gekonnt zu überfliegen. Die Picamera dient dazu ein Ziel zu erkennen und dieses anzusteuern, mit Hilfe des Ultraschallsensors kann ein Hindernis erkannt werden, woraufhin in den Flugmodus gewechselt wird, um das Hindernis zu überfliegen und um nach einer bestimmten Zeit x zu landen und wieder in den Fahrmodus zu wechseln. Wie

in Abbildung 9: Quadrocopter: Sensorkonzept veranschaulicht sind dabei beide Sensoren mit einem Microcontroller verbunden.

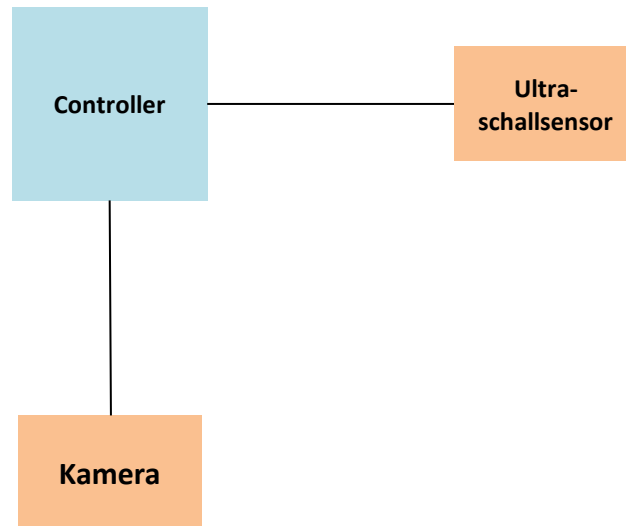


Abbildung 9: Quadrocopter: Sensorkonzept

2. MATERIALIEN UND METHODEN

Um das Projekt zu realisieren und die Mohnke zu entwickeln, wurde Seitens des Auftraggebers verschiedene Materialien und Werkzeugmaschinen sowie ein Budget von 40€ zur Verfügung gestellt.

Es gab viele Faktoren, die dazu beitrugen, wie dieser Hybrid-Quadrocopter konstruiert werden sollte. Das Team musste zuerst berücksichtigen, dass die Drohne Testflüge absolvieren musste, von denen sie gegebenenfalls nicht in einem Stück zurückkehrt. Hinzu kam, dass das Gewicht der für die Herstellung der Drohne verwendeten Materialien sorgfältig geprüft werden musste, damit das Gewichtslimit und auch die Größe für den Wettbewerb nicht überschritten wurde. Schließlich musste das Design auf eine Art und Weise erstellt werden, welches leicht mit den zur Verfügung stehenden Materialien und Werkzeugmaschinen umgesetzt werden konnte. Zudem musste das Design die Elektronik vor Wasser schützen. Aufgrund des entscheidenden Faktors, die für die Produktion zur Verfügung stehenden Zeit, wurde entschieden, Lasercutten, 3D-Druck, Schneiden, Schleifen, Fräsen und verbinden mit Schrauben zu verwenden. Dieser Prozess wurde als am schnellsten und am leichtesten reproduzierbar bestimmt, falls erforderlich. Der erste Schritt in diesem Prozess war das Design

der Drohne zu entwerfen. Dafür wurde die in Abschnitt 2.3.1 beschriebene CAD-Software Autodesk Inventor verwendet. Im nächsten Schritt wurde bestimmt, wie die einzelnen Komponenten hergestellt und welche Materialien dafür verwendet werden sollten. Komponenten wie Arduino, Raspberry Pi und Sensoren wurden zur Verfügung gestellt. Da jedoch kein Breadboard verwendet werden durfte, wurde mit Hilfe der in 2.3.2 beschriebenen Software ein Leiterplatten-Layout erstellt, welches mit Hilfe einer Fräsmaschine gefertigt wurde.

In diesem Kapitel werden die bereitgestellten Materialien aufgelistet, die verfügbaren Maschinen vorgestellt sowie eine Vorstellung der verwendeten Software.



2.1 Materialien




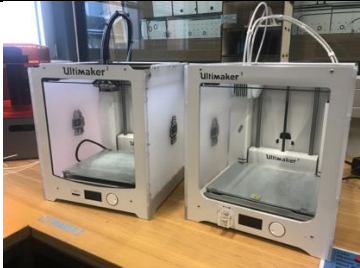

Tabelle 1: Verfügbare Materialien






Material	Maße in mm (l x b x h)
Holz	800 x 600 x 3,8
Plexiglas	600 x 400 x 3

2.2 Manufacturing Technologies

Tabelle 2: Verfügbare Maschinen und Werkzeuge

Werkzeug / Maschine	
Bohr-Fräsmaschine Optimum BF 16 Vario	
Schraubstock	

Schleifmaschine Holzstar BTS 150	
Bandsäge Metabo BAS 317 Precision	
Digitaler Messschieber Scala 150mm	
Ultimaker 2 3D-Drucker	
Ultimaker 3 Dual Extruder 3D-Drucker	
Fräsmaschine Roland SRM-20	

Lötstation Weller WS 81	
Lasergravierer Trotec Speedy 360	
Metallsäge	
Schraubenzieher	
Schraubenschlüssel	

2.3 Software Tools

Um das Projekt zu realisieren, wurden zweierlei verschiedene Softwareprogramme verwendet. Zum einen wurde Autodesk Inventor Professional verwendet, um ein 3D CAD-Modell

der Mohne zu erstellen, zum anderen wurde für die Konstruktion der Leiterplatte das Programm EAGLE der Firma Autodesk verwendet. Beide Programme werden nachfolgend kurz beschrieben.

2.3.1 Autodesk Inventor Professional

Autodesk Inventor ist eine von Autodesk entwickelte 3D-CAD-Software für die Volumenmodellierung, mit der digitale 3D-Prototypen erstellt werden können. Es wird für 3D-Konstruktion, Design-Kommunikation, Werkzeugherstellung und Produktsimulation verwendet. Diese Software ermöglicht es Benutzern, genaue 3D-Modelle zu erstellen, um Produkte zu entwerfen, zu visualisieren und zu simulieren, bevor sie gebaut werden.

Die Software ermöglicht zudem 2D- und 3D-Daten in eine einzige Designumgebung zu integrieren und so eine virtuelle Darstellung des Endprodukts zu erstellen, mit der die Form, Passform und Funktion des Produkts vor dem Bau validiert werden kann. Autodesk Inventor enthält leistungsstarke parametrische, direkte Bearbeitungs- und Freiform-Modellierungswerkzeuge sowie Multi-CAD-Übersetzungsfunktionen und DWG Zeichnungen nach Industriestandard, wodurch Entwicklungskosten gesenkt, schnellere Time-to-Market realisiert und großartige Produkte entwickelt werden können. Somit haben wir die "Mohne" zunächst vollständig im CAD Programm entworfen, um schnell Anpassungen vornehmen zu können und gleichzeitig keine Bauteile zu vergessen.

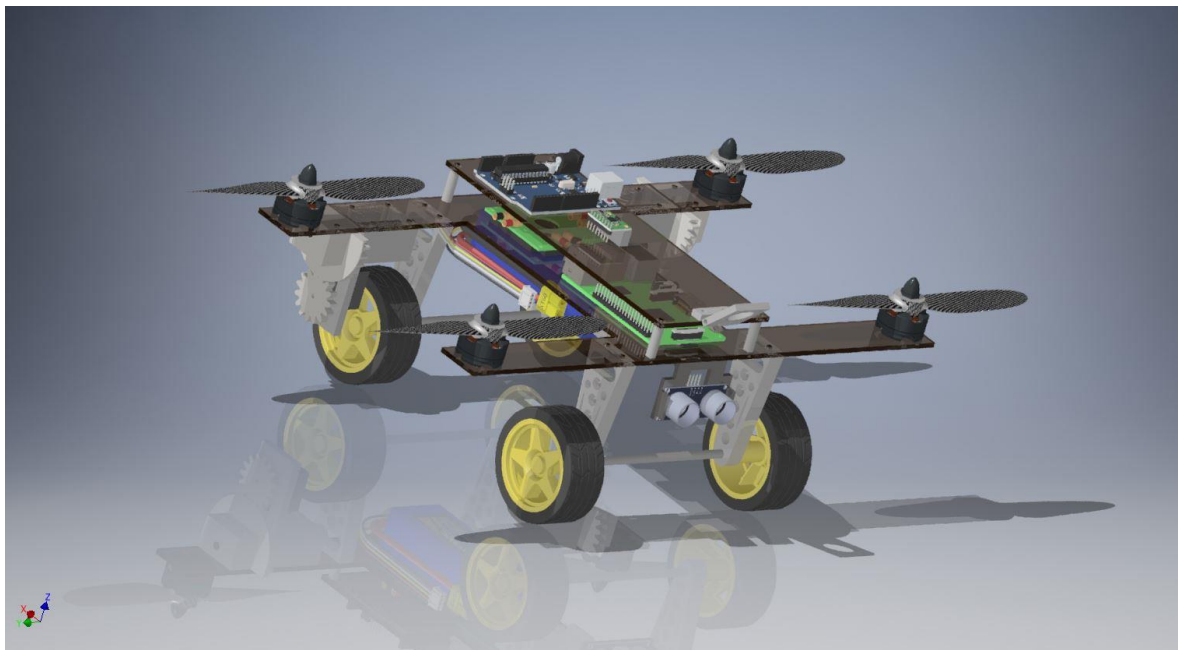


Abbildung 10: Fertiges CAD Model der "Mohne"

Tabelle 3: Stückliste des CAD Modells

Name	Anzahl
Chassis mit motorlöcher	1
Platzhalter Zylinder Klein	4
Metall Stab M5	1
Chassis-2	1
Kegelrad	1
EMax Motor	4
RaspberryPi2 STEP214	1
Metall Stab M5_MIR	1
StirnradUNDServo_MIR	1
Propeller	4
arduino uno	1
MPU 6050 IMU	1
Abstandhalter_MPU	1
LiPo Akku	1
PDB	4
Ultrasonic Sensor Halterung	1
USS	1
Wheel ALL	2
Radaufhängung	4
Wheel ALL1_MIR	2
Achse	2
Kamerahalterung	1
ESC	4
M3 Schrauben	27
M3 Beilagscheiben	27
M3 Muttern	27
Raspberry Pi Camera	1
Stirnzahnrad1	2
Stirnzahnrad2	2
servo_SM-S2309S	2

2.3.2 Autodesk EAGLE

EAGLE ist eine skriptfähige EDA-Anwendung (Electronic Design Automation) mit Schaltplan-Erfassung, Leiterplatten-Layout, Auto-Router und CAM-Funktionen (Computer Aided Manufacturing). EAGLE steht dabei für Easy Applicable Grafischer Layout-Editor.

Das Programm bietet eine grafische Benutzeroberfläche und ein Menüsystem mit mehreren Fenstern für die Bearbeitung, das Projektmanagement und die Anpassung der Schnittstellen- und Designparameter. Das System kann über Maus, Tastaturkürzel oder durch Eingabe bestimmter Befehle in einer eingebetteten Befehlszeile gesteuert werden. Mehrere Wiederholungsbefehle können zu Skriptdateien kombiniert werden. Es ist auch möglich, Entwurfsdateien unter Verwendung einer EAGLE-spezifischen objektorientierten Programmiersprache zu untersuchen.

2.3.3 Arduino IDE

Arduino bringt eine eigene integrierte Entwicklungsumgebung (IDE) mit, die auf Wiring IDE basiert. Dabei handelt es sich um eine Java-Anwendung, die für die gängigen Plattformen Windows, Linux und MacOS kostenlos verfügbar ist. Sie basiert auf der IDE von Processing, einer auf die Einsatzbereiche Grafik, Simulation und Animation spezialisierten Entwicklungsumgebung. Die Arduino-IDE bringt einen Code-Editor mit und bindet gcc als Compiler ein. Zusätzlich werden die avr-gcc-Library und weitere Arduino-Librarys eingebunden, die die Programmierung in C und C++ stark vereinfachen. Arduino kann verwendet werden, um eigenständige interaktive Objekte zu steuern oder um Softwareanwendungen auf Computern zu interagieren (z.B. Adobe Flash, Processing, Max/MSP, Pure Data, diverse Skriptsprachen, etc.)

Für ein funktionstüchtiges Programm genügt es, zwei Funktionen zu definieren:

- `setup()` – wird beim Start des Programms (entweder nach dem Übertragen auf das Board oder nach Drücken des Reset-Tasters) einmalig aufgerufen, um z. B. Pins als Eingang oder Ausgang zu definieren.
- `loop()` – wird durchgehend immer wieder durchlaufen, solange das Arduino-Board eingeschaltet ist.

2.3.4 Raspbian

Raspbian ist ein kostenloses Betriebssystem, das auf Debian basiert und für die Raspberry Pi-Hardware optimiert ist. Ein Betriebssystem ist die Menge der grundlegenden Programme und Dienstprogramme, die den Raspberry Pi laufen lassen. Raspbian bietet jedoch mehr als

ein reines Betriebssystem: Es enthält mehr als 35.000 Pakete, vorkompilierte Software in einem schönen Format für die einfache Installation auf dem Raspberry Pi. Dieses Betriebssystem basiert auf einem Debian-9-System (Debian Stretch) der ARM-hard-float-Architektur (armhf) mit Anpassungen an den Befehlssatz für den ARMv6-Prozessor. Als grafische Oberfläche wird LXDE vorkonfiguriert. Das etwa 3 GB große Image kann auf SD-Karten mit 4 GB oder mehr übertragen werden. Nach dem Bootvorgang kann die Größe der Raspbian-Partition auf die gesamte SD-Karte erweitert werden. Die Raspberry Pi Foundation erstellt auf Basis der Raspbian-Distribution ein eigenes Raspbian-Image mit passender Firmware für die Raspberry-Pi-Modelle, es wird daher empfohlen, die Distribution immer von der Raspberry Pi Foundation zu beziehen.

2.3.5 Python

Das Pi im Namen des Raspberry Pi leitet sich von der Programmiersprache Python ab: Die Schöpfer des Minicomputers wollten damit zum Ausdruck bringen, dass diese moderne Skriptsprache besonders zu empfehlen ist.

Python ist eine universelle, üblicherweise interpretierte höhere Programmiersprache. Sie hat den Anspruch, einen gut lesbaren, knappen Programmierstil zu fördern. So werden beispielsweise Blöcke nicht durch geschweifte Klammern, sondern durch Einrückungen strukturiert.

Python unterstützt mehrere Programmierparadigmen, z. B. die objektorientierte, die aspektorientierte und die funktionale Programmierung. Ferner bietet es eine dynamische Typisierung. Wie viele dynamische Sprachen wird Python oft als Skriptsprache genutzt.

Die Sprache hat ein offenes, gemeinschaftsbasiertes Entwicklungsmodell, das durch die gemeinnützige Python Software Foundation, die de facto die Definition der Sprache in der Referenzumsetzung CPython pflegt, gestützt wird.

Wegen ihrer klaren und übersichtlichen Syntax gilt Python als einfach zu erlernen. Sie besitzt eine umfangreiche Standardbibliothek und zahlreiche Pakete im Python Package Index.

Zur Standardinstallation von Python gehört eine integrierte Entwicklungsumgebung namens IDLE.

Raspbian bringt die Python-Entwicklungsumgebung IDLE bereits vorinstalliert auf dem Desktop mit, sodass bei Bedarf direkt losgelegt werden kann.

IDLE besteht im Wesentlichen aus drei Komponenten:

- **Die Python-Shell:** Beim Starten von IDEL öffnet sich zuerst das Python-Shell-Fenster. Die Shell ist eine Anwendung, mit der Sie direkt mit dem Python-Interpreter kommunizieren können: Sie können auf der Kommandozeile einzelne Python-Anweisungen eingeben und ausführen lassen. Ein Python-Programm, das eine Bildschirmausgabe liefert, gibt diese in einem Shell-Fenster aus.
- **Der Programmmeditor:** Das ist eine Art Textverarbeitungsprogramm zum Schreiben von Programmen. Sie starten den Programmmeditor vom Shell-Fenster aus (FILE|NEW WINDOW).
- **Der Debugger:** Er dient dazu, den Lauf eines Programms zu kontrollieren, um logische Fehler zu finden.

2.3.6 Samba

Samba ist ein freies Programmpaket, das es ermöglicht Windows-Funktionen wie die Datei- und Druckdienste unter anderen Betriebssystemen zu nutzen und die Rolle eines Domain Controllers anzunehmen. Es implementiert hierfür unter anderem das Server Message Block/Common Internet File System-Protokoll (SMB/CIFS-Protokoll). Im Projekt wurde Samba als DC und DNS verwendet, damit man via RDP über WLAN auf den Pi zugreifen kann, ohne eine statische IP im Pi hinterlegen zu müssen. Der hostname der Drohne lautet „Mohne“.

BEGINN Systems Engineering – Christian Gruber

3. SYSTEM LIFE CYCLE

As part of the Mars Explorer project, the insights gained from the courses Systems Engineering, Embedded Systems and Project Management will find application as well as a Fab-lab environment, mechanical and electronic component production and board production.

3.1 System requirements

3.1.1 Purpose

The purpose of the project, with the provided components, materials, machines and budget, is the development of an autonomous vehicle suitable for extra-terrestrial exploration missions. The suitability is checked on five test tracks. The requirements for the vehicle are as follows: Unbeschadeter Sturz aus einer maximalen Fallhöhe von 0,5m

- Water and dust protected
- Dimensions for the transportation by rocket:
 - length < 65 cm
 - width < 48 cm
 - height < 30 cm
- Weight of the vehicle:
 - $m_{\text{Rover}} < 10 \text{ kg}$ (Earth)
 - $m_{\text{Rover}} < 4,5 \text{ kg}$ (Mars)
- Type of movement freely selectable, but autonomous
- Budget: 40 €

3.1.2 Goal

The project is to lead to an autonomous hybrid quadrocopter, which can overcome the tracks described under 3.6, by switching between driving and flight mode and detects its surroundings using an ultrasonic sensor and camera and reacts accordingly to different situations. At the heart of the vehicle is a Raspberry Pi and an Arduino, which receive data, process data and forward data.

Sub goals for the project are:

- Design and manufacture of the chassis
- Construct and manufacture the rotating mechanism of the rear wings to switch between driving and flying modes
- Design and manufacture circuit board
- Programming the motor control incl. ESC calibration to create the conditions for flying
- Realize autonomous mode with the help of sensors (programming for ultrasonic sensor and camera)
- Control code and PID control to keep the quadcopter stable in the air

3.1.3 Limitations

Switching on and off takes place in manual mode, while flying and driving, the "Mohne" acts autonomously. The distance to the ground is not measured. Obstacles are overcome by switching from the driving mode to the flying mode and then returning to the driving mode.

The system only must work in good weather conditions and on given courses, which is why only one straight-ahead flight is realized.

3.2 Design alternatives

In order to decide on a design for this project, certain limiting factors had to be taken into account, namely the maximum size and maximum permissible weight that the vehicle may have in order to be able to be transported from the launcher to Mars. Even under the condition that the electronic components, which must be enclosed and fastened somewhere inside the fuselage, must be protected from dust and water, the possibilities were further narrowed. The given budget and time limit as well as the available materials and machines limited further possibilities. After some deliberations and research on how to overcome the individual hurdles of the test tracks, it was decided that a hybrid Quatrocopter would be developed. Also due to the fact that drones are becoming increasingly popular and are being used in more and more areas of application, we wanted to meet this challenge and renounce the construction of a conventional Mars rover.

3.3 Model the System

As a rule, models are developed for almost all alternative designs. The model for the preferred alternative is then extended and used to manage the system throughout its life cycle. Due to the short development time, no models for alternative designs were developed. In the first place, the following models are also intended to describe the system.

3.3.1 System and Subsystem

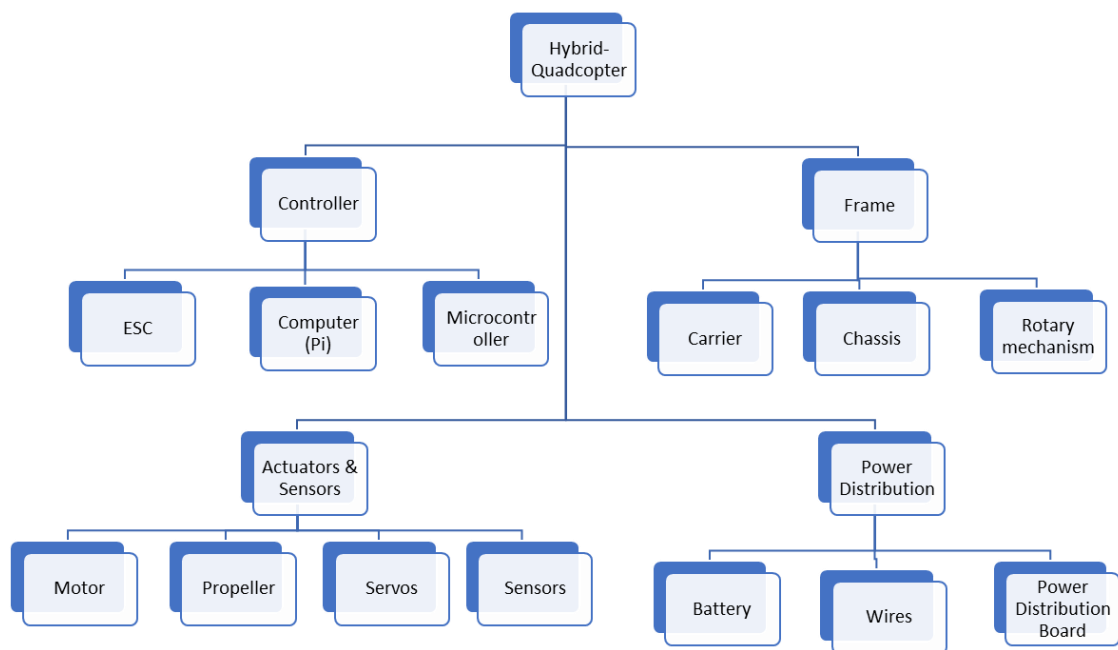


Abbildung 11: High level overview systems and subsystems

3.3.2 Make or Buy

Make or buy was a quite easy story. Most of the electrics couldn't be made by us. So, we had to buy them. But especially the mechanical parts could be designed and manufactured by us.

Tabelle 4: Make or Buy table

Motor	Buy
Propeller	Buy
ESC	Buy
Servo	Buy
Gyro and USS	Buy
Arduino	Buy
Raspberry Pi	Buy
Carrier	Make
Chassis	Make
Rotary mechanism	Make
Battery	Buy
Wires	Buy
Power Distribution Board	First Make, after "burn out" Buy
Wiring	Make

3.3.3 System Architecture

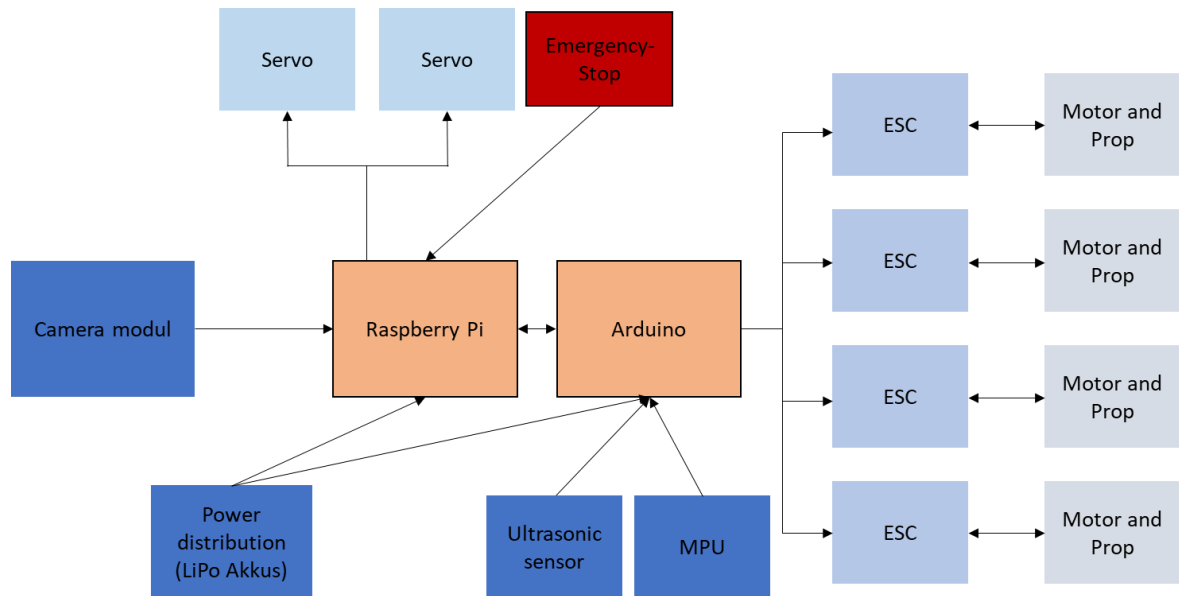


Abbildung 12: System architecture and communication

3.3.4 Functional Decomposition

Tabelle 5: Function table

Function	Physical Component
Take-off, drive and land	Wheels
Sense objects	Sensors
Navigate	Microcomputer
Produce horizontal thrust and vertical lift	Propeller and wings

3.3.5 Failure Mode and Effects Analysis (FMEA)

The procedure used to evaluate the various modes of failure in our drone design takes into consideration the severity, probability of occurrence and detectability for each failure mode as well as the root causes. Based on the values assigned to each category, the risk priority and criticality of each cause are calculated. Our failure analysis focused on four failure modes; mobility, navigation, structural and data acquisition. We determined the key failures

modes to be mobility and navigation. In each of these two cases, control system and wiring failure are the root causes with the highest risk priorities and criticalities.

Tabelle 6: Failure mode and effects analysis matrix

Possible Effect	Root Cause	Potential Indicators	Potential Cause(s) of Failure	Severity	Probability of Occurrence	Detectability	Risk Priority	Criticality
Mobility	dead battery	will not power motor	Battery not charged	5	3	1	15	15
	controls failure	Motors, servos and PID controller failure	Programming error	5	9	9	405	45
	wiring harness connections	will not start or run correctly	Grounding corrosion	8	8	9	576	64
Navigation	Wiring failure	Complete navigation failure	Wiring error	8	7	2	112	56
	Controls Failure	Complete navigation failure	Programming error	5	9	9	405	45
	Object sensing hardware failure	Part damage	Wiring control failure	5	3	3	45	15
	Rotary mechanism failure	Steering failure	Impact on gear	5	5	1	25	25
Structural	Frame cracking	No component support	Impact on frame	10	3	4	120	30
	Engin mount	Engin falls out	Engin mount wear	2	1	1	2	2
Data Acquisition	Camera not working	No video recording	Weather proofing controls/wire	1	5	2	10	5
	Ultrasonic sensor not working	No distance measurement	Weather proofing controls/wire	1	5	3	15	5
	Incorrect wiring	Will not move	Wiring error	2	2	4	16	4
	Controls failure	No/incorrect data	Programming error	1	2	5	10	2

3.4 Integrate System Components

For system integration one important point is that every part for its self is well developed and kind of works alone. That makes it easier to fix things, because it is easier to fix a little part, than the whole system. The second import point of system integration is combination of all subsystems to one completely working system. Picture X from <http://www.livewires-automation.co.uk> fits perfect for showing the integration of subsystems in the overall System called “Mohne”. I crossed out the parts of the puzzle, we didn’t really have in our project and still there are many parts we had to combine. I marked two parts of the puzzle, which haven’t “worked” perfectly in our project, because of we couldn’t finish the perfect solution in time, this will be explained later but clearly shows the importance of well integrated system components.

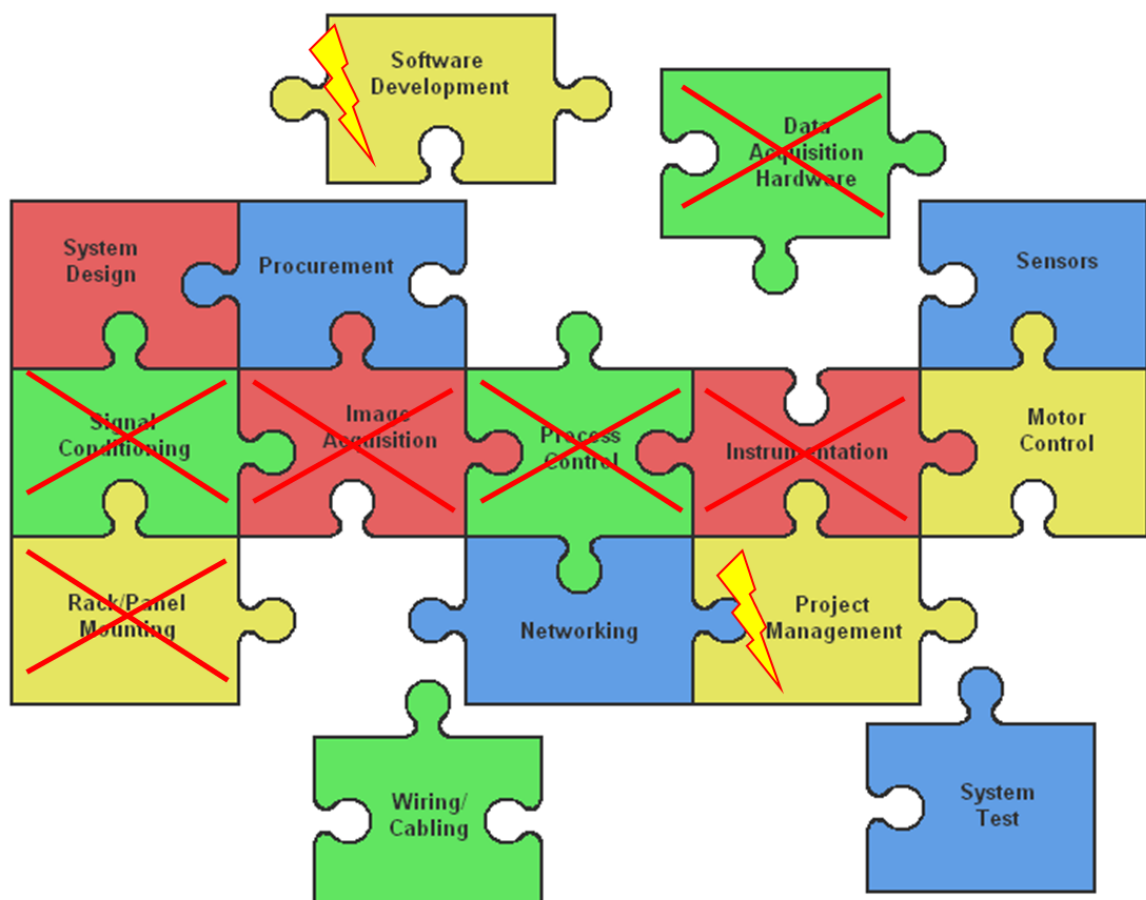


Abbildung 13: System integration – not used and critical parts

3.5 Launch the System

3.5.1 Timeline

For the overall timeline we decided to use a traditional planning Method “waterfall”. We setup discussed and described the project phases and setup milestones with goals we wanted to reach after each phase.

Conception Phase	Activities	Week1	Week2	Week3	Week4	Mars Landing
	Internet Research					
	Requirements (5 Tracks)					
	Possibilities for the execution and timeline					
	Strengths and Weaknesses of our teammates					
	Consider period of time (4 weeks)					
Define and Planning Phase						
	Design/Technology/Basic structure/Code/ Electronic of our Mohne					
	Budget and timeline					
	Distribution of tasks (construction, programming, electronic, PCB etc.)					
	Create CAD construction for the whole vehicle					
Execution Phase						
	Create Prototype at the 1st date in Wattens					
	Testing (Construction, Code, Electronic etc.)					
	Optimization of our Mohne					
Mars Landing						
	Finalisation of our Mohne					
	Mars Landing					

Abbildung 14: Milestone plan

Therefore, we setup tasks in our agile planning tool on github. We combined traditional planning methods (overall planning), with agile project methods (scrum mode doing the tasks).

Tabelle 7: Planed tasks and responsibilities

Task	Team Member(s)
Design decision	all
Chassis construction and manufacture	Duregger, Jungwirth, Gruber
Image recognition	Paula, Gruber
Camera mount	Duregger, Jungwirth
Suspension, axles and shaft	Duregger, Jungwirth, Gruber
PCB Layout	Möbes
Electronics	Möbes, Gruber

Servo mount and turning mechanism	Duregger, Jungwirth, Gruber
Motor mount	Duregger, Jungwirth, Gruber
Spacers between boards	Duregger, Jungwirth, Gruber
ESC Calibration and motor control	Möbes, Gruber
Plug-in connection HV battery and circuit board	Möbes
Control code	Möbes, Paula, Gruber
Project Management and documentation	Wiesmüller, Gruber
Assembly	Wiesmüller, Gruber
Cost Analysis	Gruber
Component Selection and Ordering	all
Test and System test	all

3.5.2 Budget

Because of the very small budget of 40€ we had to do a little budget planning. But we still couldn't exactly hold the 40€. For this small budget our project was much too ambitious.

Tabelle 8: Budget planning of "Mohne"

Budget Kalkulation	
Material	Kosten
Ladegerät und Akku	0€ (geliehen)
LHI 2CW 2CCW MT2204 II 2300KV Brushless Motor	16,99 €
Carbon propeller	5,60 €
4X MR.RC 12A Speed Controller ESC	12,92 €
MPU 6050	5,99 €
Kleinteile(Kabelbinder, Tape, Aluachsen)	5 €
Summe	46,50 €

3.6 Assess Performance

During the entire development phase, the performance of the system was measured and tested again and again. These measurements were necessary to verify that the system meets its requirements. First of all it was tested at the beginning whether the rotation mechanism works as intended, which was the basis for the switch between the driving and the airplane mode.

Tests were also carried out repeatedly to ensure the image recognition and distance measurement in order to ensure autonomous navigation. Another important point that was repeatedly tested was the motor control to ensure that the Quatrocopter can also move forward.

Finally, to determine whether the project is suitable for use on Mars, to move autonomously and to fulfil the wishes and desires of the client, 5 test routes had to be completed, which are described below.

Tabelle 9: Tracks to be overcome

Track	Dimensionsn (l x b)	Surface area	Environment condition	Target
1	3m x 1m	Concrete	Covered Room	Drive Straight
2	3m x 1m	Concrete	Dust simulation with spray bottles	Defect-Free Passage
3	3m x 1m	Wooden steps with different heights (10cm, 20cm, 30cm, 40cm, 50cm, 60cm)	Covered Room	Overcome obstacles
4	3m x 1m	Boulders of different heights (10-60 cm)	Covered room, boulders coloured and marked with symbol	Traversing the track, passing the obstacles
5	unbekannt	Unknown (snow, ice, sand, etc.), rocks of	In the open air, the rocks are coloured	Survive

		varying height (10-60cm)	and marked with a symbol	
--	--	--------------------------	--------------------------	--

The contractors also inspected whether the required minimum requirements for size, weight and stability were met.

3.7 Results, Conclusion and Recommendations

3.7.1 Results

This project lasted 4 weeks and included a lot of research, testing and analysis. At the beginning of the project, the team created initial project specifications as described in section 3.1, which served as guidelines for poppies. A review was carried out to see which specifications the project meets and which does not.

- The drone fulfilled the maximum permissible size and weight
- The drop test of 0.5 m height was survived without prejudice
- The drone met the requirements for dust and water resistance
- The drone did not meet the budget of €40. The required budget was €46.50
- The drone fulfilled the requirement of emergency stop
- The drone fulfilled the requirement to perceive its surroundings
- The switch between flight and travel mode could be realized
- The drone could go straight
- The requirement to fly was only partially achieved. Although several flight tests were carried out, it could not be implemented that all four propellers were evenly supplied with power and the drone could thus stand stable in the air. The problem was detected, but the PID control needed for this could not be implemented due to time constraints.
- Due to the previous problem, the obstacles could not be overcome

3.7.2 Conclusion

The final results showed that not all specifications and targets set at the beginning of the project were achieved. Although the fully autonomous flight has not been achieved, the team

has made a significant development towards a low-cost, lightweight unmanned hybrid vehicle that can be monitored with the implementation of a camera system and an ultrasonic sensor. Several design iterations have been completed to create a vehicle that should meet the project goals and the final prototype shows the potential for the success of the project goals. As described in section 3.6, several tests were performed to ensure that the prototype that was designed could fly and drive successfully. If more time were available, the team would have been able to design a more robust control system that would ensure a stable flight. In total, an unmanned aircraft was fully realized in this project. The hybrid quadcopter showed that he was able to drive and fly, but he lacked the stability. Further iterations of this project would secure its future success.

3.7.3 Recommendations

During the project several lessons have been learned, and the team made several recommendations. From the control point of view, more sensors would be helpful for more precision or capabilities. Another ultrasonic sensor could be used to determine the altitude. The implementation of additional camera modules could realize a 360-degree perception in order to avoid obstacles. Also, in the next step could be realized curves to drive and fly. Performing simulations and calculations would lead to a more effective design.

ENDE Systems Engineering, ABER Lessons Learned noch in Kap. 7

4. DIAGRAMS

4.1 Use Case Diagram (Anwendungsfalldiagramm)

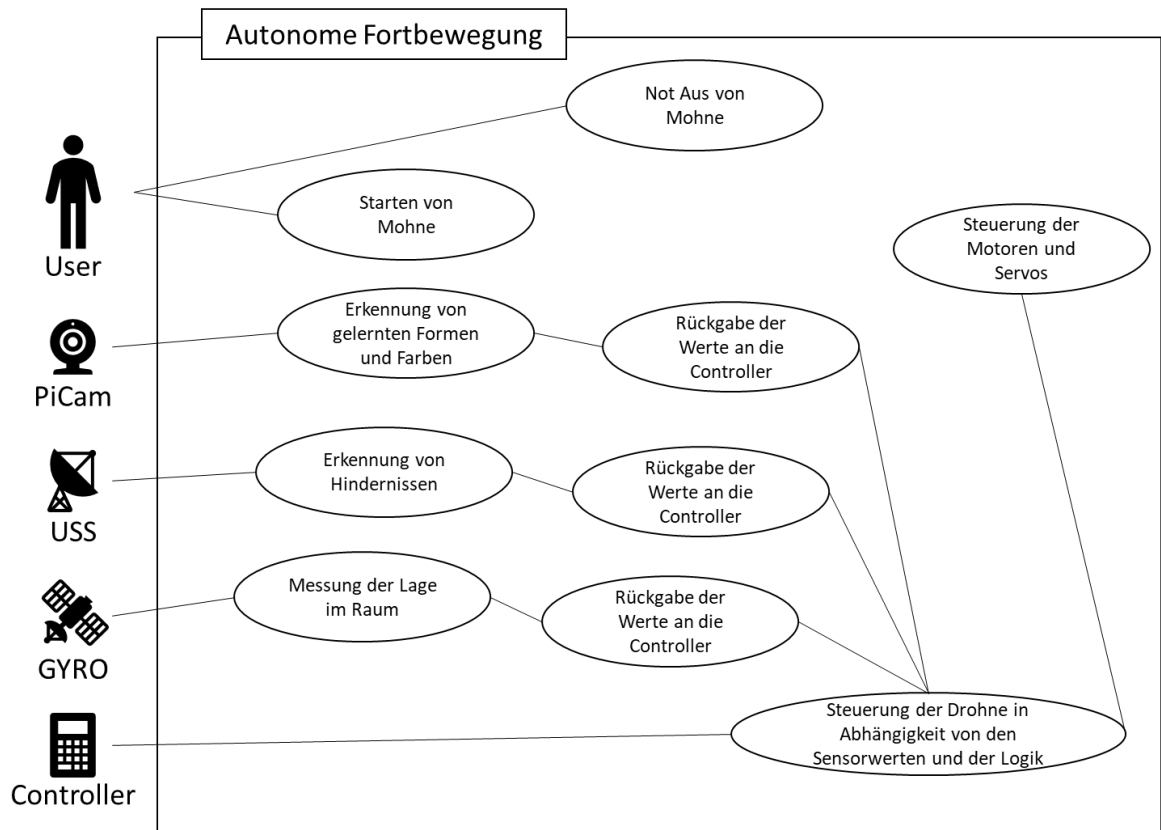


Abbildung 15: Use case diagramm "Mohne"

4.2 Structural Model (Strukturmodelle)

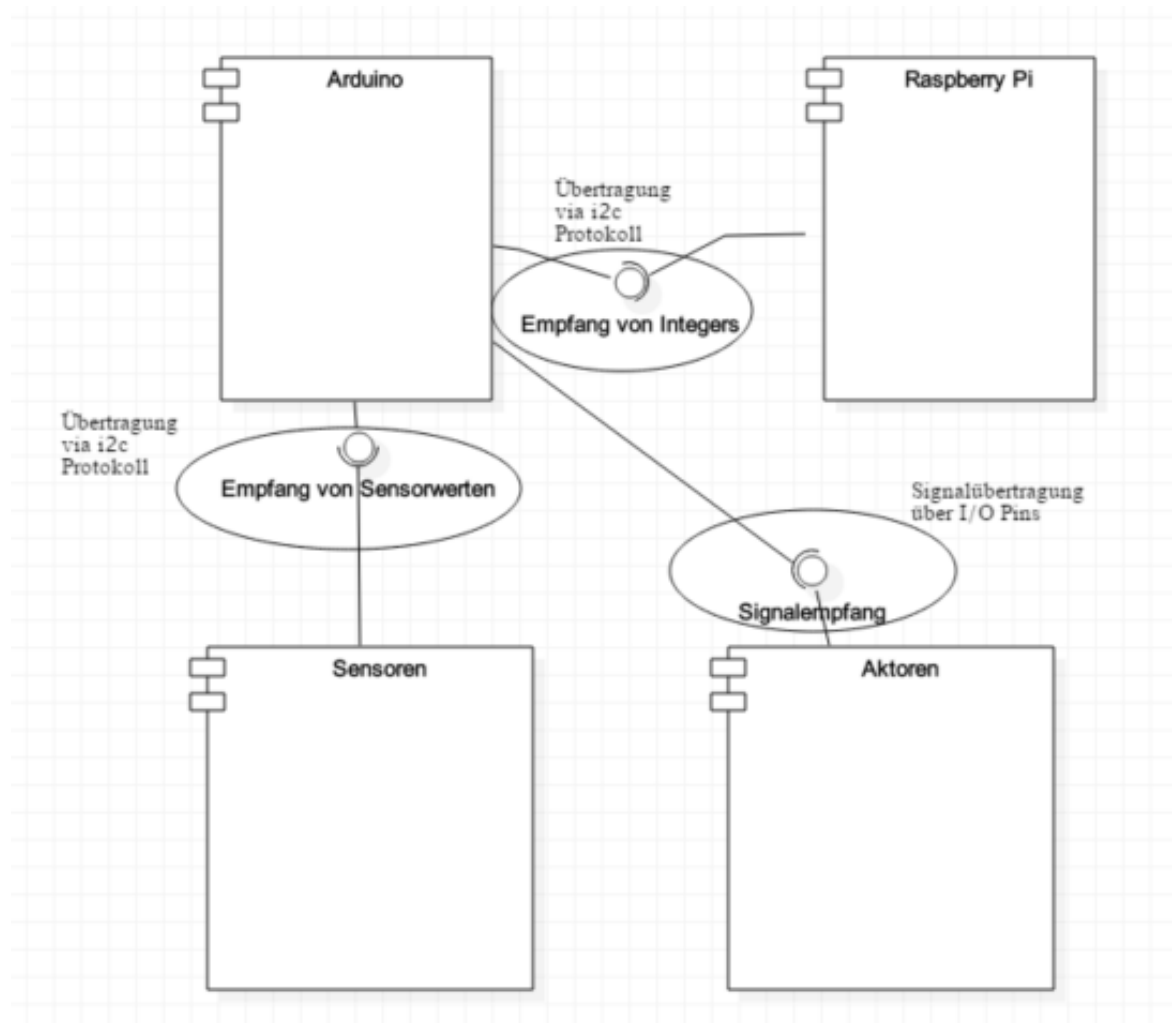


Abbildung 16: Komponentenmodell "Mohne"

4.3 Interaction Model (Interaktionsmodell)

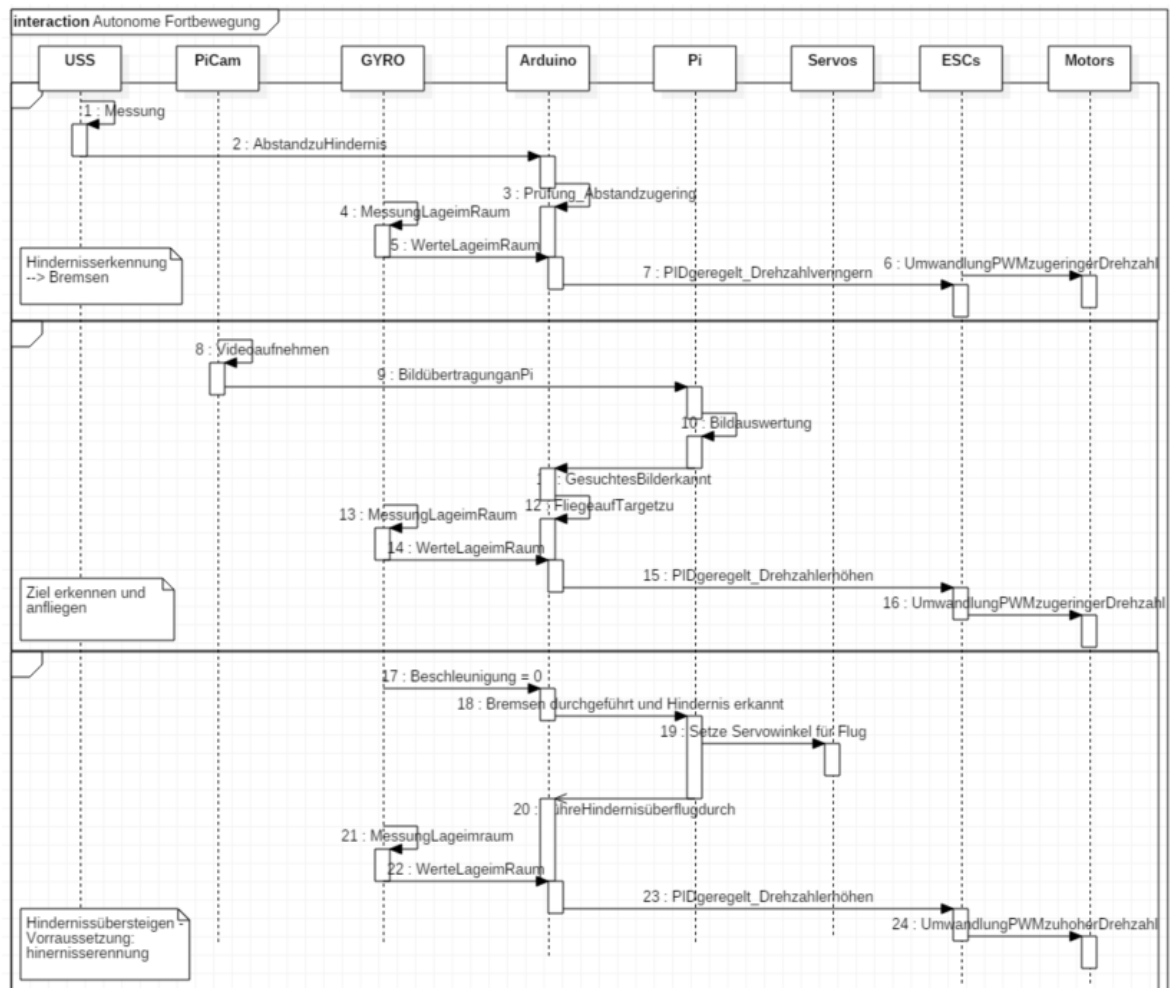


Abbildung 17: Sequenzdiagramm "Mohne"

4.4 Behavior Model (Verhaltensmodell)

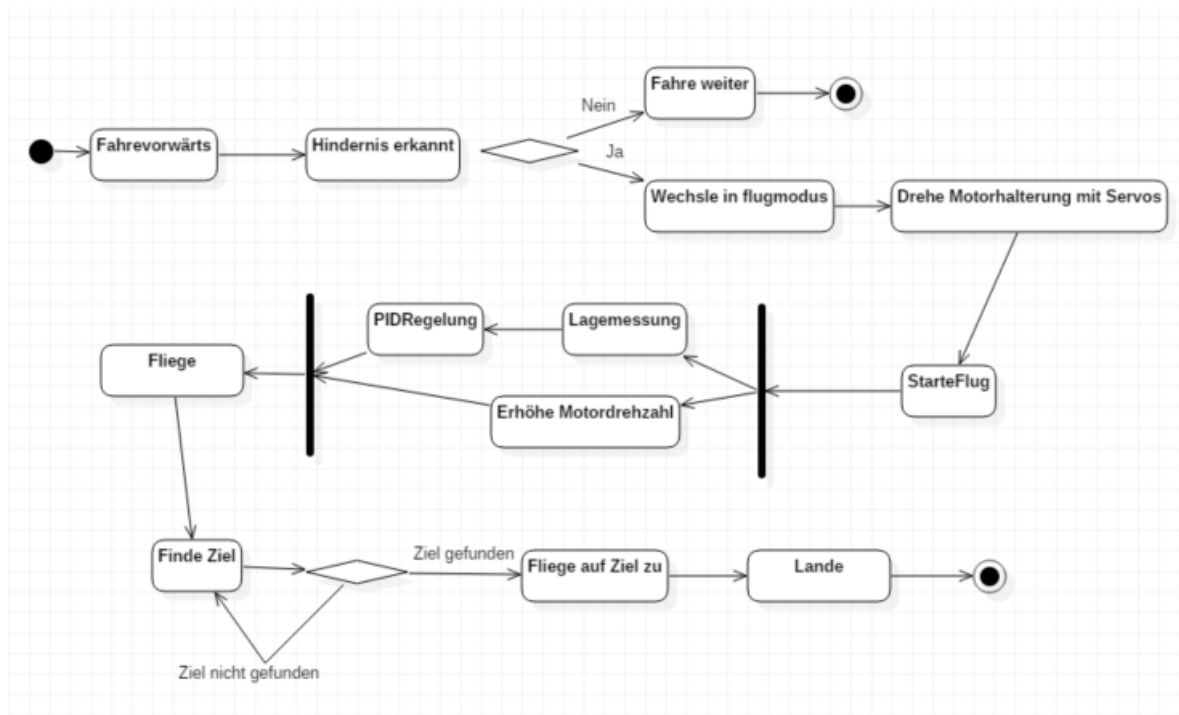


Abbildung 18: Aktivitätsdiagramm "Mohne"

5. VERSION CONTROL

Projektplanung und alle Artefakte sind in github zu finden.

<https://github.com/gchrizZz/RTW>

6. DESIGN OF PCB AND ELECTRICS

6.1 Electrics

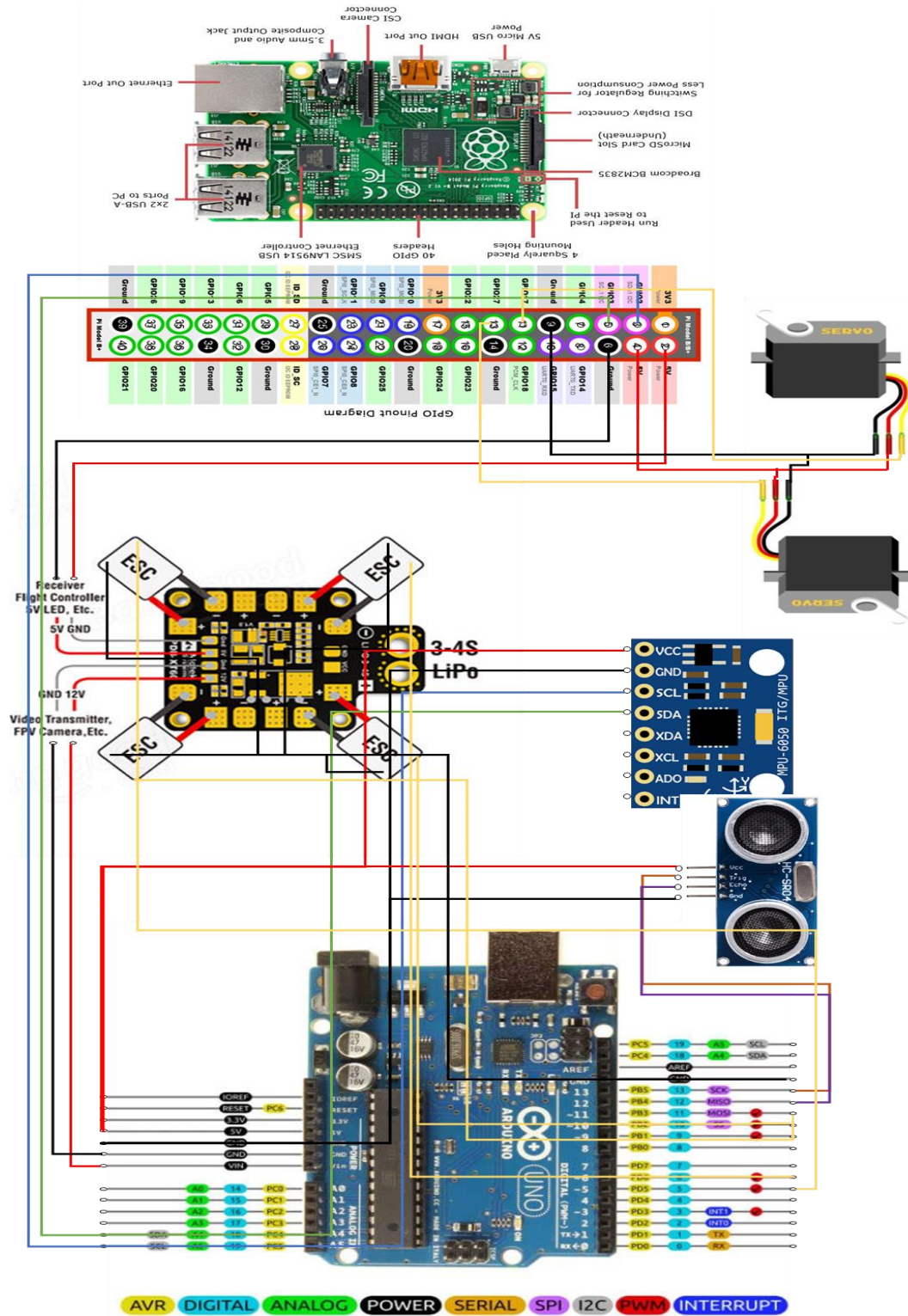


Abbildung 19: Verkabelung der elektrischen Komponenten

6.2 Schematic

Nach einigen erfolgten Diskussionen im Projektteam fiel die Entscheidung auf das Anfertigen einer Platine im Stile eines PCBs. Der ausschlaggebende Faktor für diese Entscheidung war die Prämisse, den Schaltkreis und die letztendliche Verkabelung so einfach wie möglich zu halten, um die Fehleranfälligkeit zu reduzieren.

Die gesamte Konstruktionsarbeit für die Platine erfolgte mit dem Programm Eagle von Autodesk. Der Vorteil bei der Benutzung dieses Programms besteht darin, dass im ersten Schritt die Logik komplett unabhängig vom späteren Layout erstellt werden kann. Die entsprechenden Komponenten (Schraubklemmen) des Herstellers Würth werden aus dessen Onlinebibliothek in Eagle importiert und per Drag & Drop auf das Bearbeitungsfeld gezogen. Im nächsten Schritt werden mithilfe von Labels die einzelnen Leiterbahnen so definiert, dass GND und VCC (Ground und Power) mit der richtigen Logik verbunden sind.

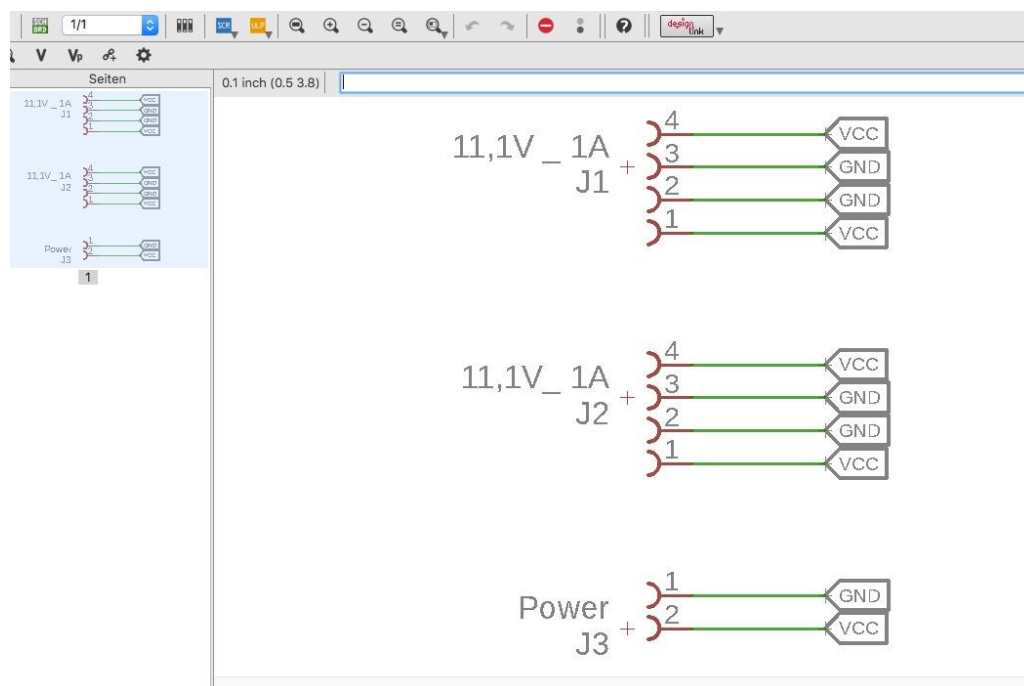


Abbildung 20: Schema des ersten PDB

6.3 Board

Nach der Erstellung der schematischen Zeichnung gelangt man durch die Aktivierung einer Schaltfläche in die Board-Layout-Ansicht. Hier kann man nun die vorher definierten Komponenten so platzieren, wie es das Projekt erfordert. Die Abmessungen des Boards werden nach Bedarf festgelegt und die Komponenten so angeordnet, dass bei den späteren Schritten

des Lötens und der Verkabelung noch genug Arbeitsraum vorhanden ist. Das ist insbesondere bei Projekten wichtig, wo die Platine vielen Tests unterzogen wird und die Verbindungen oft gelöst und wieder befestigt werden müssen.

Im letzten Schritt wird das Layout und die Dicke der Leiterbahnen festgelegt. Da die Platine mithilfe eines Fräasers erstellt wird, ist es wichtig in der Darstellung des Boards nur einen Layer (Ebene) auszuwählen. Bei einfachen Projekten, bei denen man die Funktion „Autorouting“ verwenden will, also die automatische Verbindung der Komponenten, ist das umso wichtiger. Wird das nicht korrekt definiert, kann es vorkommen, dass die Leiterbahnen sich überschneiden, da das Programm von mehreren Layern ausgeht. Sind alle Einstellungen korrekt vorgenommen, inklusive der Dicke der Leiterbahnen, kann die Autoroutingfunktion verwendet werden und das für den Fräser benötigte monochrome Bild exportiert werden.

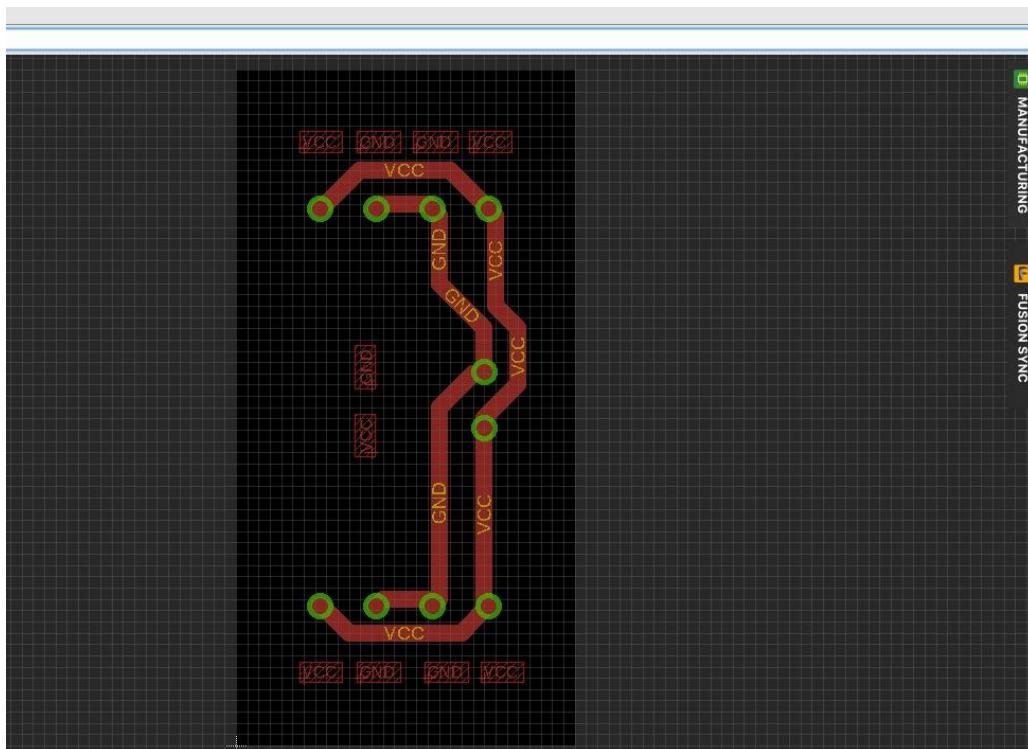


Abbildung 21: Eagle Konstruktion des ersten PDB

Leider konnte das PDB der Last nicht standhalten und brannte nach wenigen Tests durch. Weshalb Ersatz gekauft werden musste.

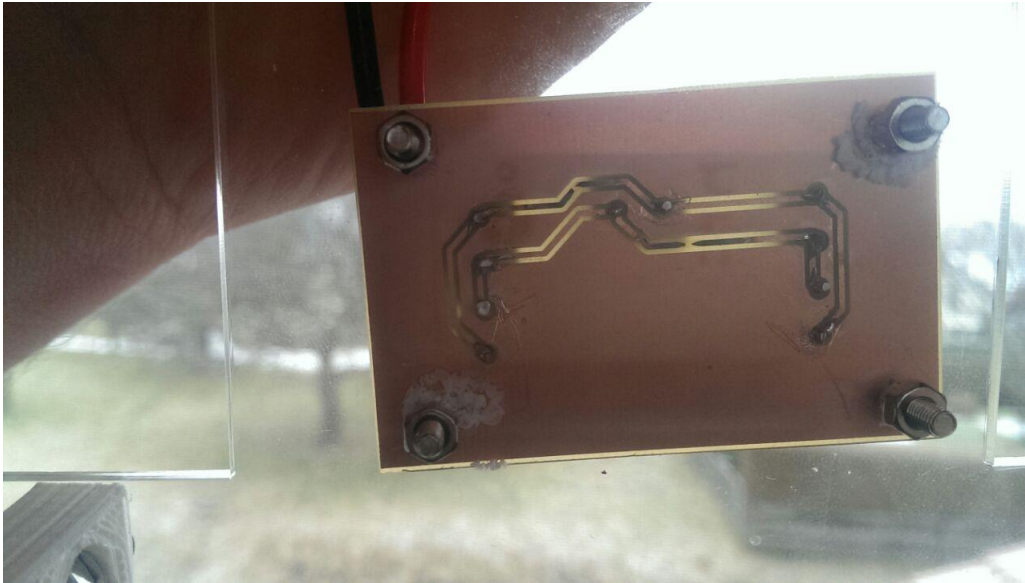


Abbildung 22: Defektes/durchgebranntes PDB

7. LESSONS LEARNED

- ❖ Be realistic! Only do projects where scope, resources and time fits together.
- ❖ So, don't be emotional! Don't try thinks which are utopic only because you like them.
- ❖ As PM give clearly defined and small tasks. So that a team member can work step by step and not got overwhelmed.
- ❖ Everyone who completes a task has to document his doings and achieves instantly.

ANHANG

