AAE

Release 3.0 Heavy

Malik Fazlic, Fabian Teppan, Felix Slama

Jun 29, 2022

CONTENTS

Teammitglieder

- Slama Felix
- Teppan Fabian
- Fazlic Malik

Projektbetreuer

- Prof. Mikl Peter
- Prof. Mairer Herwig

Supporters

- Extrudr | FD3D GmbH

CHAPTER

ONE

DOKUMENTATION

1.1 Projektplan

1.1.1 Beschreibung:

Idee:

Die Idee hinter unseren „Aether“-Projekten war es zu beweisen, dass wir per Luftantrieb, 3D-Druck und unseren derzeitigen Programmierkenntnissen in der Lage sind, Raketen zu Entwickeln, zu Bauen und im Endeffekt zum Abheben & Landen zu bringen.

Funktion:

Um den selbstständigen Start und die selbstständige Landung zu ermöglichen, müssen einige Dinge beachten werden. Damit ist gemeint: Die Rakete muss in der Lage sein sich selbst aufrecht zu halten, um unkontrollierte Neigungen im Flug zu unterbinden und somit einen Absturz zu verhindern.

Beschaffung der Teile:

Um die für unser Projekt benötigen Ressourcen und Teile zu beschaffen, mussten wir einige Entscheidungen treffen.

- Welches Material wird für den Druck verwendet?
- Welche Motoren werden gebraucht um genug Schub zu erzeugen?
- Welcher Mikrocontroller wird verwendet?
- Welche Servomotoren sind stark genug?
- Wo kann eingespart werden?
- Wo sollte nicht gespart werden?

All diese Fragen mussten wir uns stellen und beantworten, um ein bestmögliches Endergebnis zu erzielen. Alle elektronischen Bauteile wurden von europäischen Händlern bestellt, um Zölle und zusätzliche Versandkosten zu vermeiden. Das verwendete Filament wurde uns von Extrudr, einem österreichischem 3D-Drucker Filament Hersteller, zur Verfügung gestellt.

1.2 Design 1.0 - Perserverance

1.2.1 Design 1.0 Überblick:

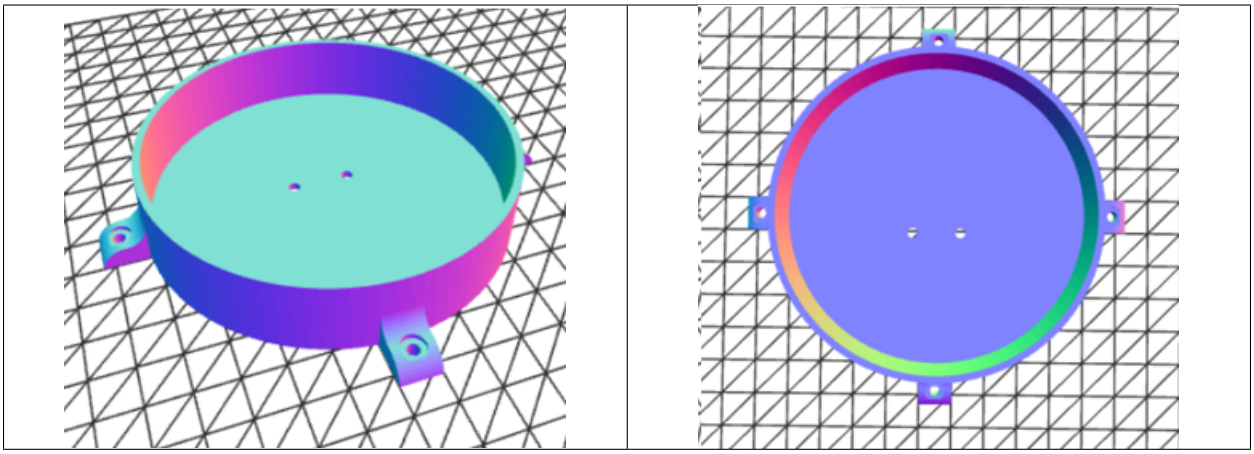
Was ist Perserverance:

Design 1.0 aka Perserverance war unser erster Prototyp der Rakete. Bei welchem es uns nicht ums Aussehen oder Gestalten der Rakete ging. Wir testeten uns bauten uns langsam immer weiter hinauf. Das Ziel von Perserverance war es uns ein gewisses Grundverständnis für unseren zukünftigen Raketen zu geben. Hierbei Probierten wir verschiedene Möglichkeiten um die Rakete in einen sicheren Start und Haltung zu bringen. Die Rakete Bestand mehreren Modulen siehe {Hyperlink to 3.2} für weitere Details.

1.2.2 Design 1.0 Struktur:

EPM - Electronic Processing Module.:

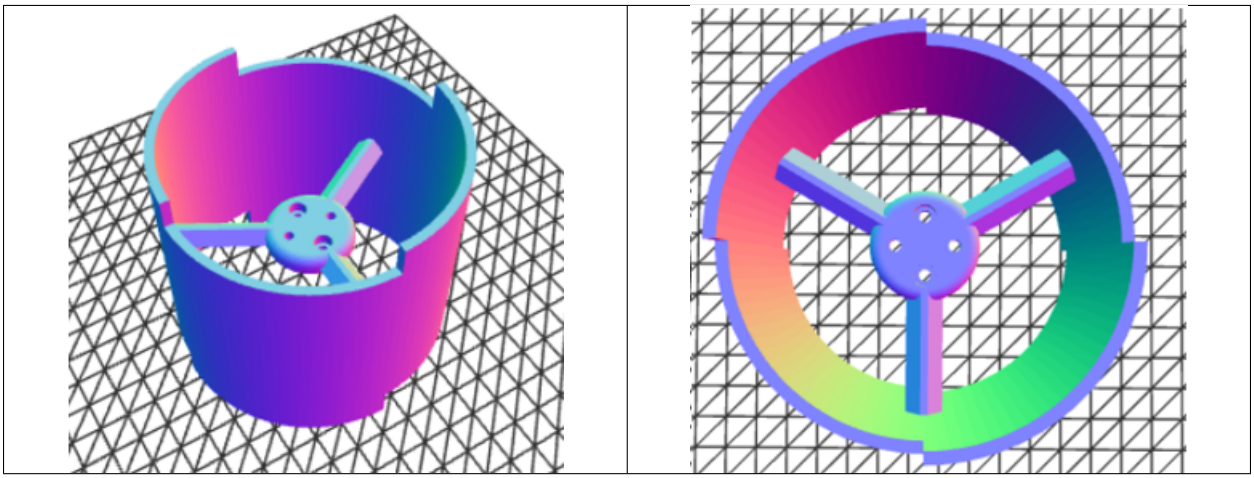
Die Aufgabe des EPM- Electronic Processing Modules war es unser Gyrosensor zu beinhalten und uns konstant Daten über die Lora Kommunikation zu übertragen als auch diese über ein Integriertes Data-Logging zu speichern und später auszuwerten. Der Gyrosensor wurde hierbei in den zwei oben abgebildeten Löchern befestigt. Zudem Zeitpunkt waren keine weiteren Gedanken für dieses Modul gedacht und Kabelführung musste händisch abgesichert werden.



TDM - Thrust Delivery Module:

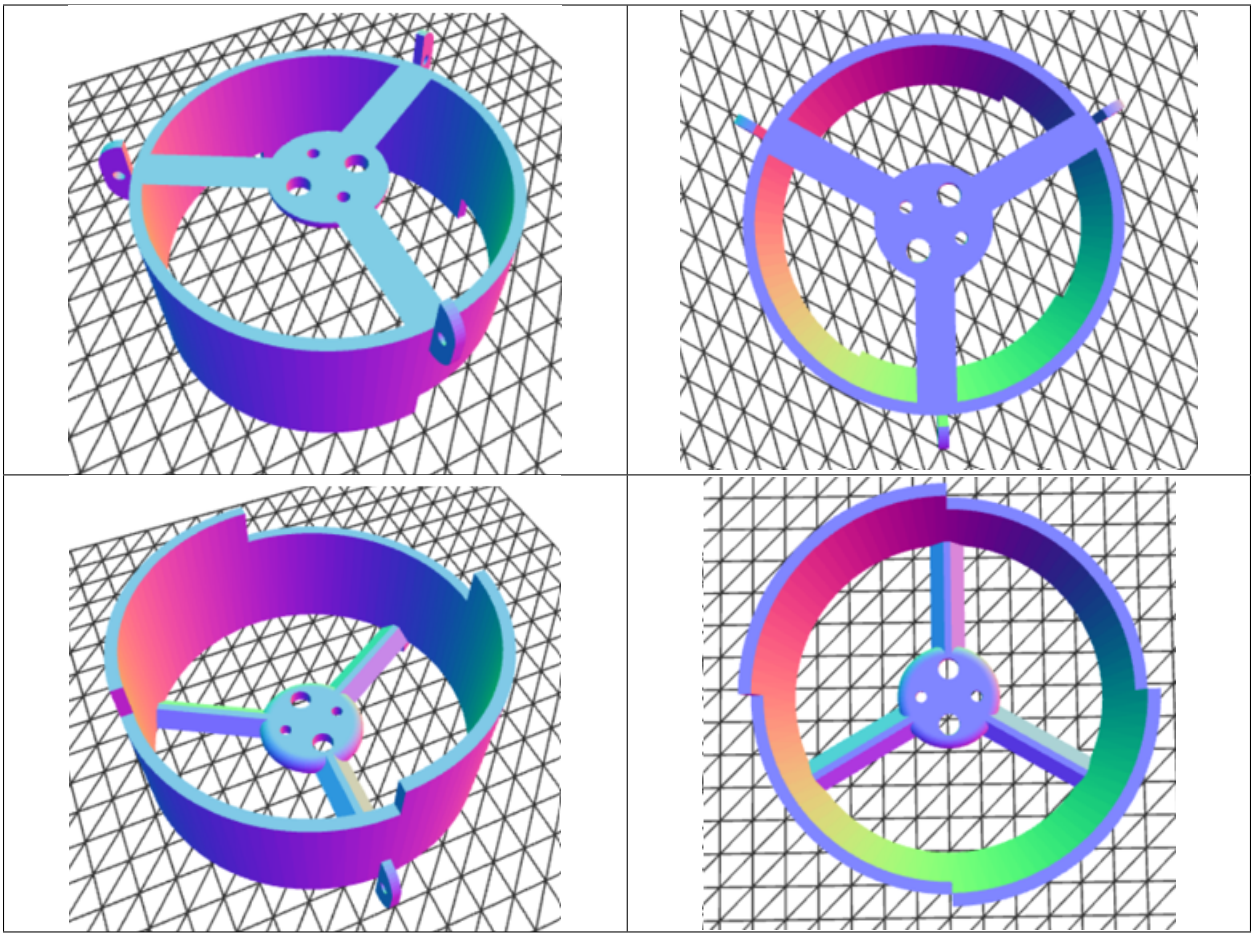
VERSION 1.0:

Die Aufgabe des TDM- Thrust Delivery Modules war es unsere Motoren zu stabilisieren, welche wir für den Start als auch die Haltung in der Luft benötigten. Verwendet wurden dabei 2 Motoren. Einer der Motoren wurde an der Oberseite des ihnen oben gezeigten Bildes angehängt, der andere auf der Unterseite. Damit unsere Rakete nicht außer Kontrolle geraten würde, entschieden wir uns dazu die beiden Motoren in die jeweils entgegengesetzte Richtung zu rotieren, um so einen Ausgleich in Schwungrichtung zu erhalten.



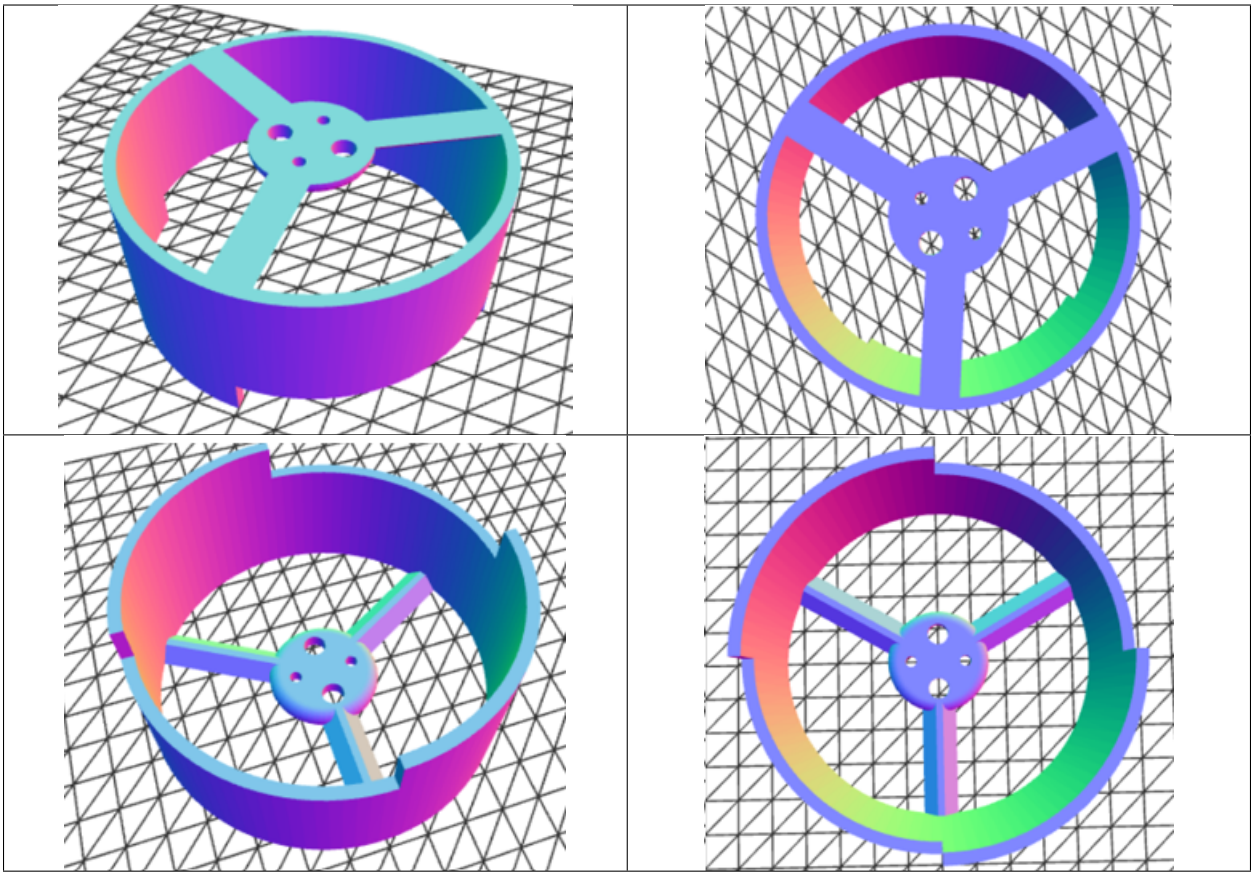
VERSION 2.0:

Bei Version 2 änderte sich nicht viel am TDM. Grund für das Aufspalten des alten Moduls war das Vereinfachte Drucken und Zusammenbauen der Rakete. Zudem fügten wir draußen 2 Schraubenlöcher an den jeweiligen 2 Teilen an, um diese Stabil zusammenzuhalten.



VERSION 3.0:

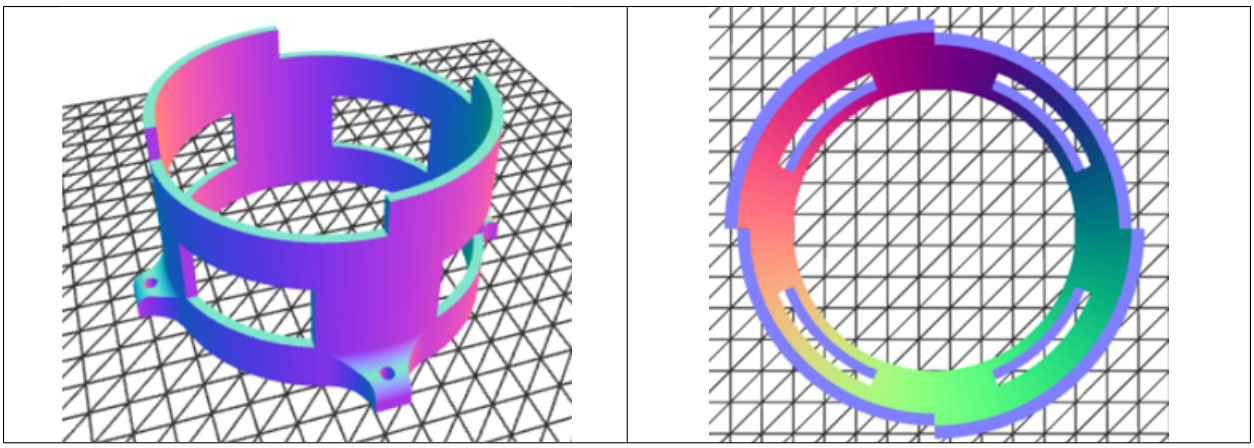
Version 3 folgte dem Beispiel von Version 2. Die Teile wurden erneut ums minimale geändert diesmal waren nur die 2 Schraubenlöcher außerhalb des TDM davon betroffen. Wie sich herausstellte erwiesen die Schrauben nicht den gewünschten Zusammenhalt. Um nun jedoch nicht wieder auf Version 1 zurückzuweichen wurde hier entschieden durch einfaches Löten der Beiden Teile den Zusammenhalt zu sichern.



TIM - Thrust Intake Module:

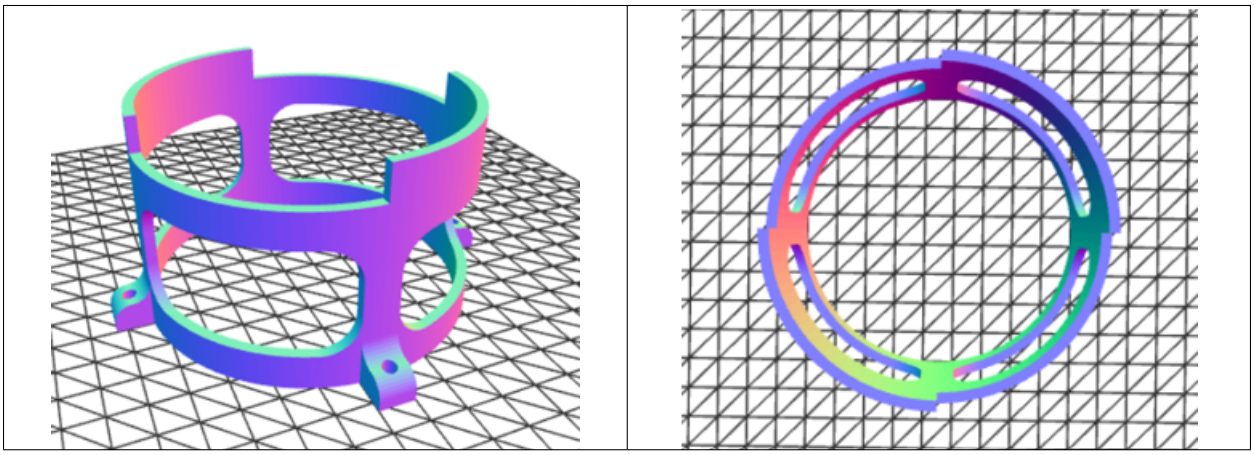
VERSION 1.0:

Die Aufgabe des TIM- Thrust Intake Modules war es wie im Namen bereits erkennbar den Motoren, welche im TDM befestigt waren, ausreichend Luft zufuhr zu geben. Jedoch war die hier zugefügte Luft nicht ausreichend um unseren Motoren die Möglichkeiten zu bieten genug Schub zu erzeugen um abzuheben.



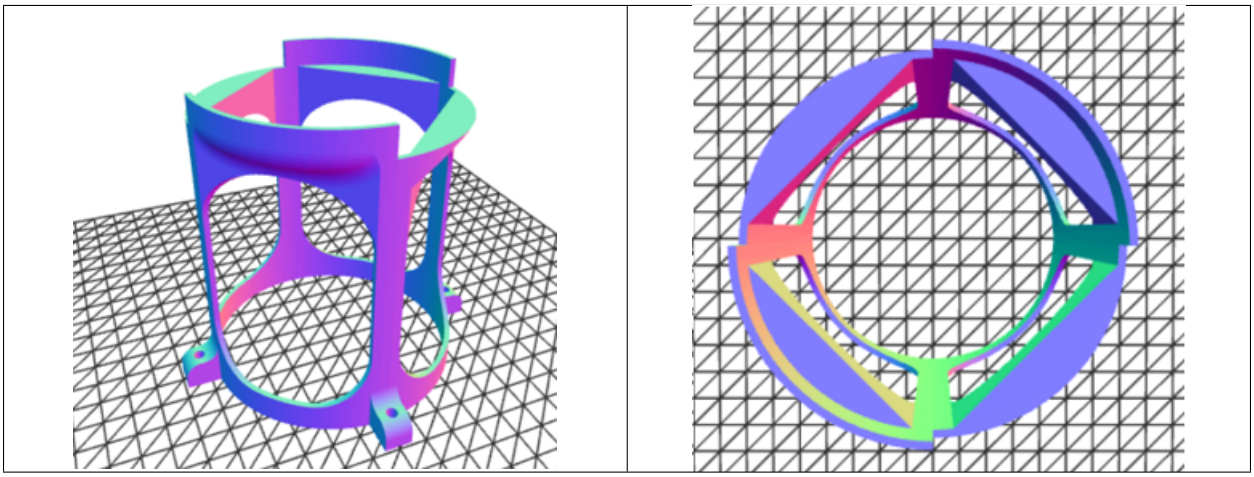
VERSION 2.0:

Version 2.0 des TMI war mit viel größeren Öffnungen ausgestattet. Unsere Ergebnisse waren schon um ein einiges besser, jedoch konnten wir trotz größeren Öffnungen nicht das gewünschte Ziel erreichen, welches wir uns vorgenommen hatten. Wir mussten improvisieren um Testungen besser durchzuführen und nahmen das EPM für weitere Tests hinunter, um ideale Lufteinnahmen zu erreichen.



VERSION 3.0:

Version 2.0 des TMI war mit viel größeren Öffnungen ausgestattet. Unsere Ergebnisse waren schon um ein einiges besser, jedoch konnten wir trotz größeren Öffnungen nicht das gewünschte Ziel erreichen, welches wir uns vorgenommen hatten. Wir mussten improvisieren, um Testungen besser durchzuführen und nahmen das EPM für weitere Tests hinunter, um ideale Lufteinnahmen zu erreichen. Jedoch gelang es uns trotz der Vergrößerung und Ausweitung des Modules trotzdem nicht die Rakete in eine Stabile und ausreichend starke Position zu bringen. Des Weiteren wurde bei der Erstellung der Version 3.0 nicht ganz auf die Ausreichende Menge an Platz für die Verkabelung gedacht.

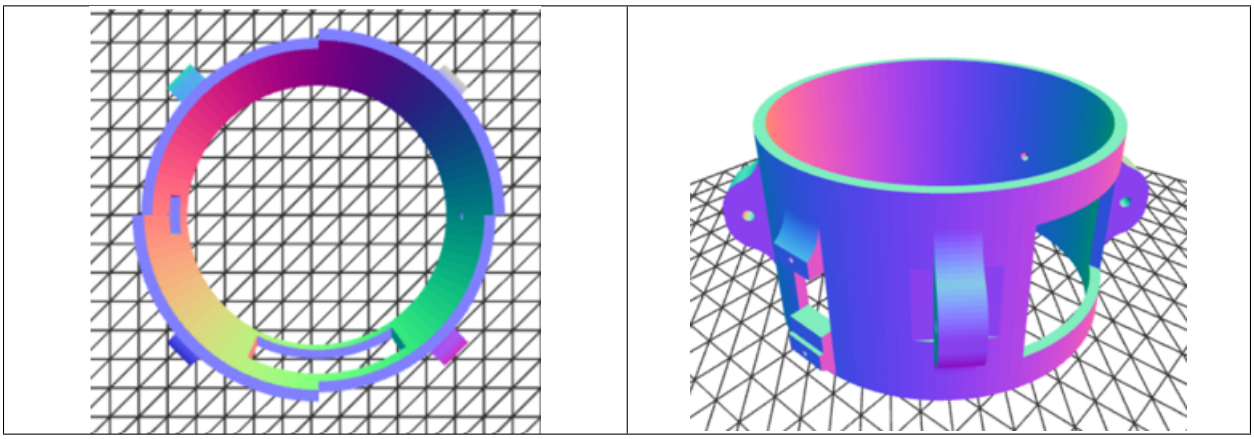


TVC - Thrust Vectoring Module:

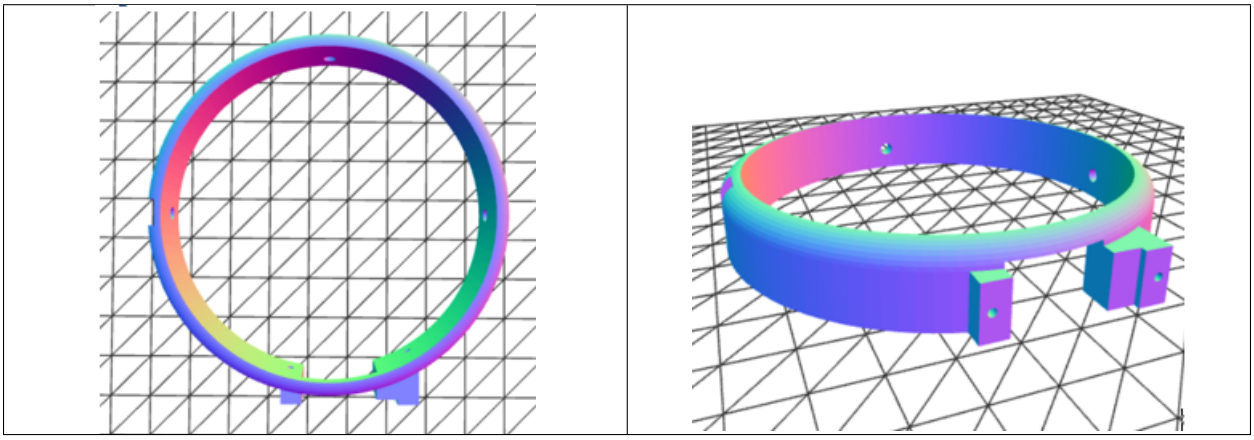
VERSION 1.0:

Die Aufgabe des alten TVC- Thrust Vectoring Modules war es die Rakete Zu Stabilisieren und die Luftausgabe in eine bestimmte Richtung zu lenken. Das Modul befand sich direkt unterm dem TDM und war sozusagen direkt für den Air Flow der Rakete verantwortlich. Der Main Part war mit mehreren Servos an den Inneren Ring befestigt mit welchem es uns ermöglicht wurde die Nozzle zu bewegen. Problem hierbei aber war, dass ein gewisser Anteil der Luft vorbeiströmte und es uns nicht möglich war alle Lücken abzudichten.

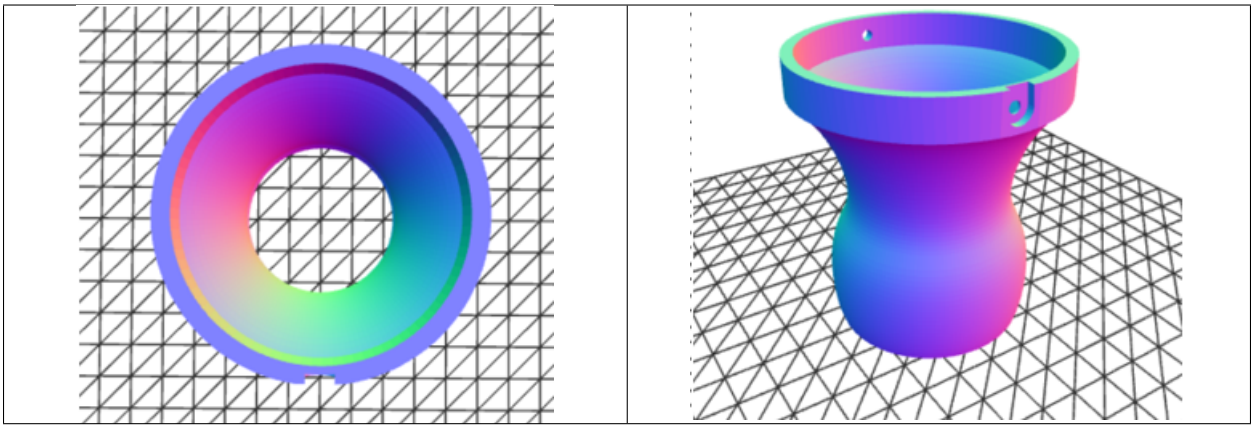
Main Part



Inner Ring



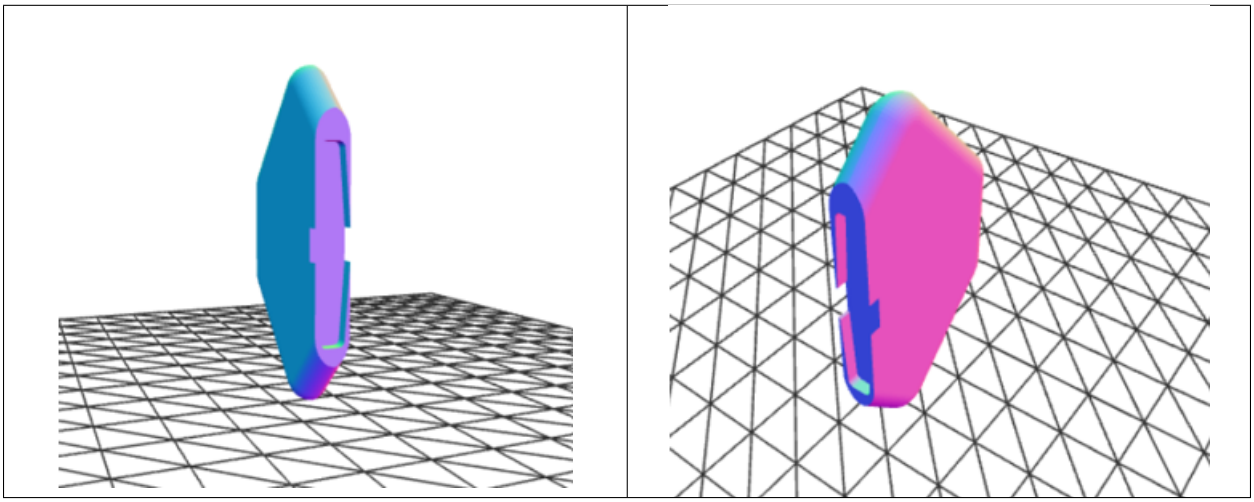
Nozzle



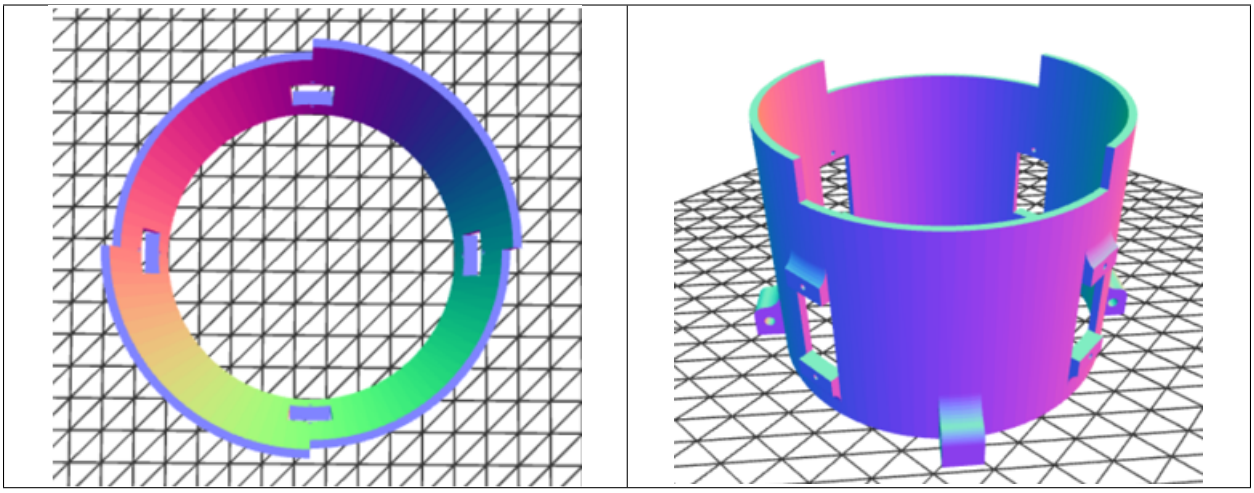
VERSION 2.0:

Um unserem größten Problem mit der Luft Ausweichung entgegenzutreten, entschieden wir uns das TVC-Thrust Vectoring Module von Grund auf zu ändern. Anstatt zu versuchen unseren gesamten Luftstrom durch eine Nozzle verlaufen zu lassen entschieden wir uns den Luftstrom auf übliche weise durch eine Weite Öffnung am TVC durchströmen zu lassen. Um jetzt trotzdem in der Lage zu sein die Rakete zu halten oder zu stabilisieren, wurden vier jeweils gegenübergestellte Fins verwendet, welche die Luftrichtern beim Verlassen der Rakete steuerten.

Fins



Main Part



1.2.3 Design 1.0 Zusammenspiel:

Zusammengefasst waren die Ergebnisse von Perserverance als erste Rakete erstaunlich gut. Das alleinige Losfliegen beim allerersten Design war überraschend. Jedoch fehlte es der Rakete an Kabelmanagement Möglichkeiten und Air Flow was der Rakete im weiteren Verlauf des Projekts nicht fehlen durfte. Wodurch diese durch das Design 2.0 ersetzt wurde.

1.3 Design 2.0 - The Injector

1.3.1 Design 2.0 Überblick:

Was ist The Injector:

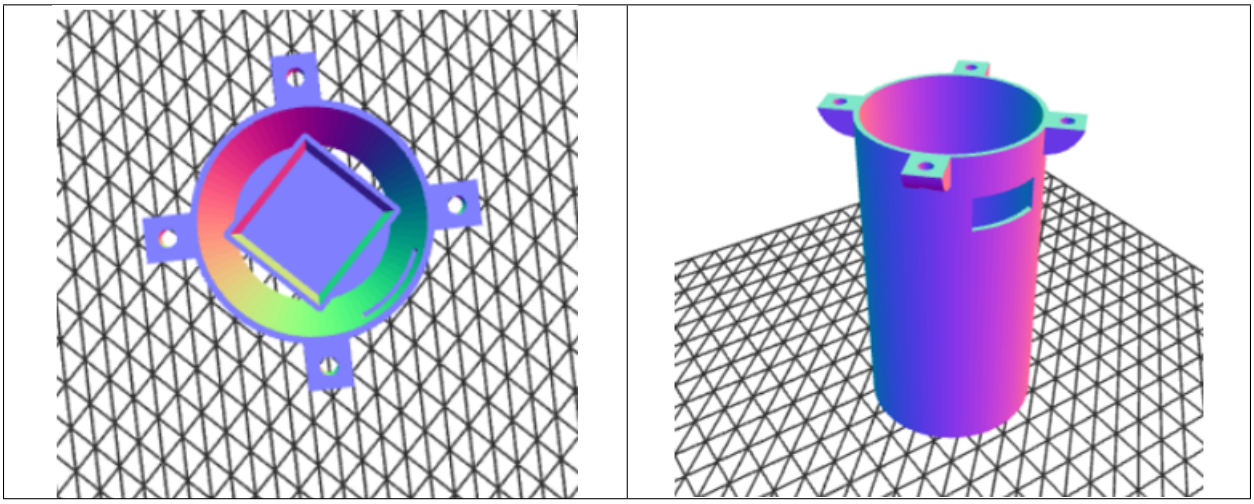
Design 2.0 aka “The Injector” war unser zweiter Prototyp der Rakete. Da wir nun jedoch genug Daten aus unseren Vorherigen Tests gesammelt hatten wussten wir nun, worauf wir beim Design der Rakete unbedingt achten mussten. Als erstes entschieden wir und das Design der Rakete grundlegend komplett zu verändern. Die Rakete musste unter allen Umständen aerodynamischer sein, weniger wiegen und bessere Möglichkeiten für Kabelmanagement und später eingefügte Raketenteile haben. Um hier nicht auf weitere Motoren ausweichen zu müssen entschieden wir uns die Breite gewisser Teile auf eine Breite von 70mm zu drucken. Gewisse Teile wie TVM, TDM und TIM blieben jedoch auf einer Breite von 90mm, da diese auf keinen Fall brechen durften. Das Ziel von “The Injector” war es nun in der Lage zu sein den Air Flow der Rakete so kompakt und smart zu designen wie möglich.

1.3.2 Design 2.0 Struktur:

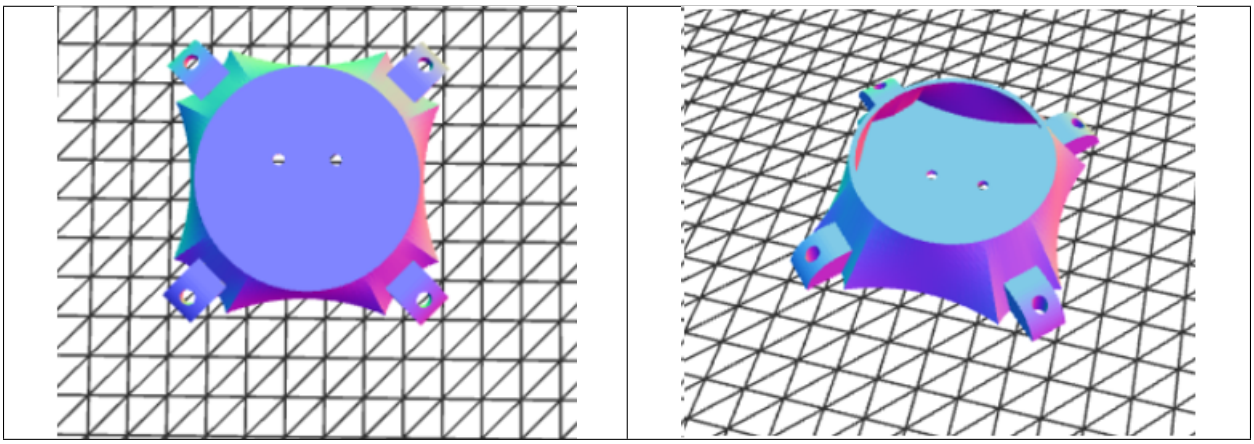
EDM - Energy Delivery Module:

Das EDM – Energy Delivery Module welches zuvor bei Design 1.0 Persevirance nicht benötigt worden war, wurde nun bei “The Injector” eingefügt. Bei Design 1.0 wurde die Stromzufuhr extern abgeregelt. Dies wollten wir bei “The Injector” natürlich ändern. Um der Rakete zu ermöglichen, ohne Einfluss von außen Stromzufuhr zu haben implementierten wir das EDM so ziemlich an der Spitze der Rakete. Dies hatte mehrere Gründe. Einerseits war es bei einer Rakete sehr wichtig die Nase schwerer zu gestalten als den Rumpf und die Lenkung und die Kontrolle zu erleichtern. Andererseits würde die Positionierung des EDM an einer niedrigeren Position Auswirkungen auf unseren Luft Fluss haben. Um das EDM nun noch besser zu stabilisieren und Kabelführung einfacher zu gestalten entwarfen wir den Coupler. Dessen Aufgabe war es die Batterie in Position zu halten und Kabelführung in der Rakete zu vereinfachen und beinhaltet außerdem den Gyro Sensor.

Main Part

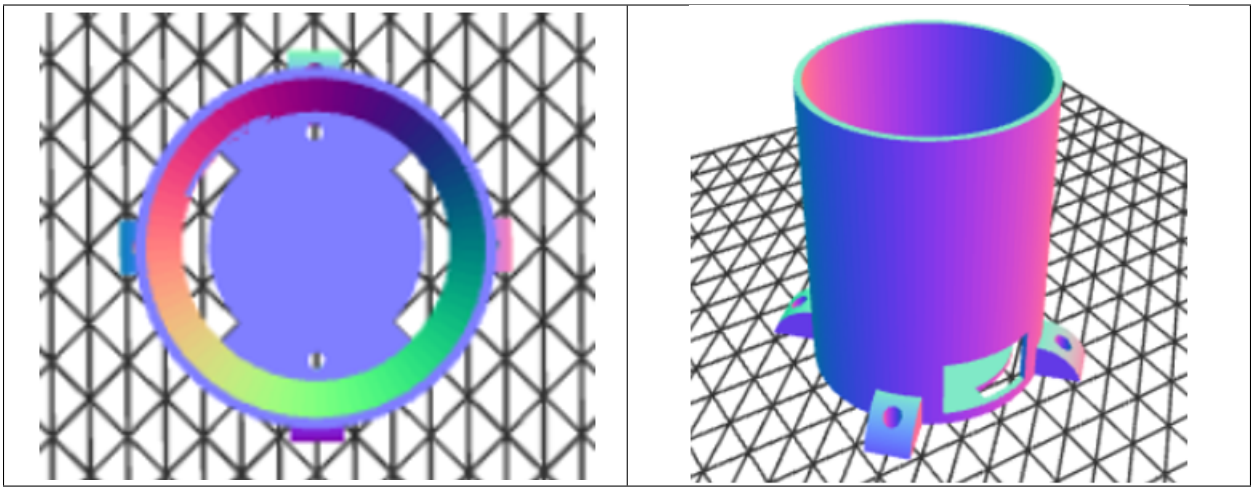


Coupler



EPM - Electronic Processing Module:

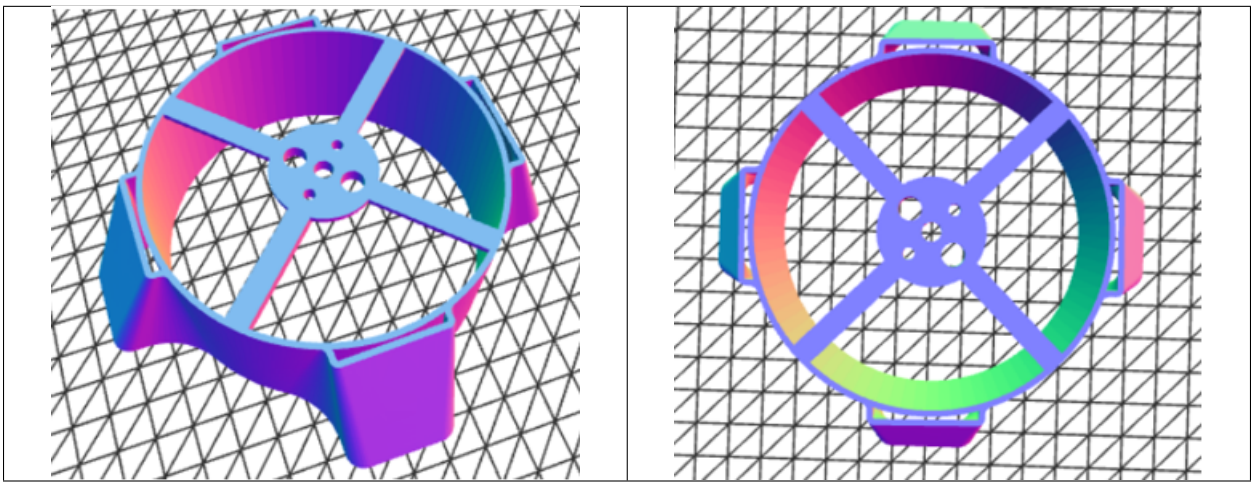
Die Aufgabe des EPM bei “The Injector” war es die ganze Elektronik zu verstauen und diese Damit für die Rakete leicht zugänglich zu machen. Das EPM befindet sich direkt über dem EDM und beinhaltet die gesamte Elektronik außer den Gyro Sensor, welches sich im unteren Teil des EDM befindet.



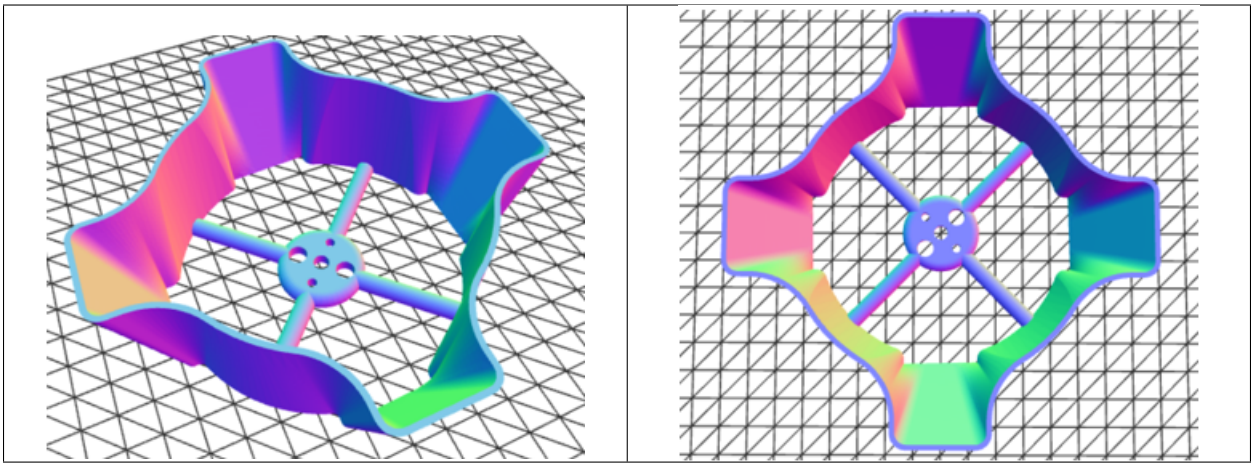
TDM - Thrust Delivery Module:

Wie zuvor bei Persevirance dient das TDM - Thrust Delivery Module der Aufgabe die von uns verwendeten Motoren zu befestigen. Bei diesem Design versuchten wir aber im oberen Bereich des TDM mehr Platz zu schaffen, um der eingezogenen Luft die Möglichkeit zu geben sofort von den Motoren eingesogen zu werden.

Bottom



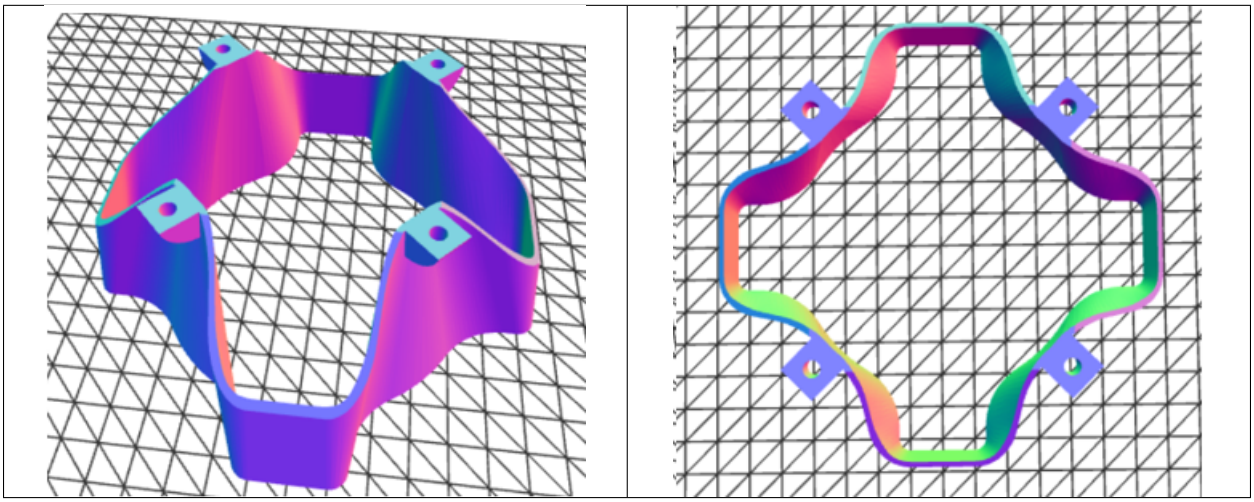
Top



TIM - Thrust Intake Module:

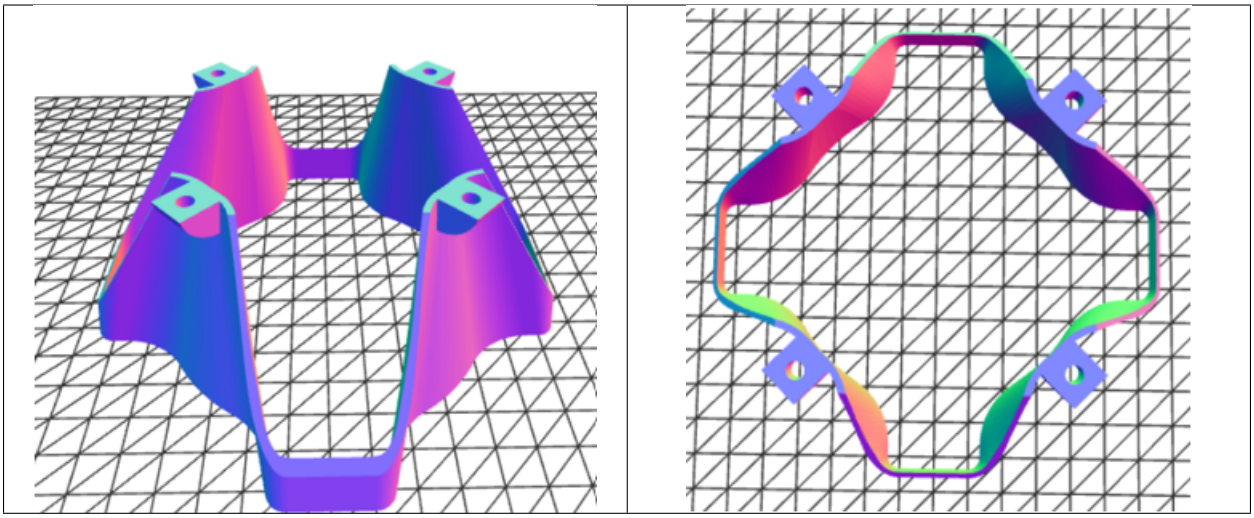
VERSION 1.0:

Aus den von Persevirance erhaltenen Ergebnissen wussten wir das wir etwas am TIM ändern mussten. Unser erster Ansatz war es einen Kleinen Überhang zu erschaffen über den die Luft leichter in die Rakete geriet. Erste Anläufe zeigten gute Ergebnisse, jedoch waren wir den von uns gewünschten Ziel noch weit entfernt.



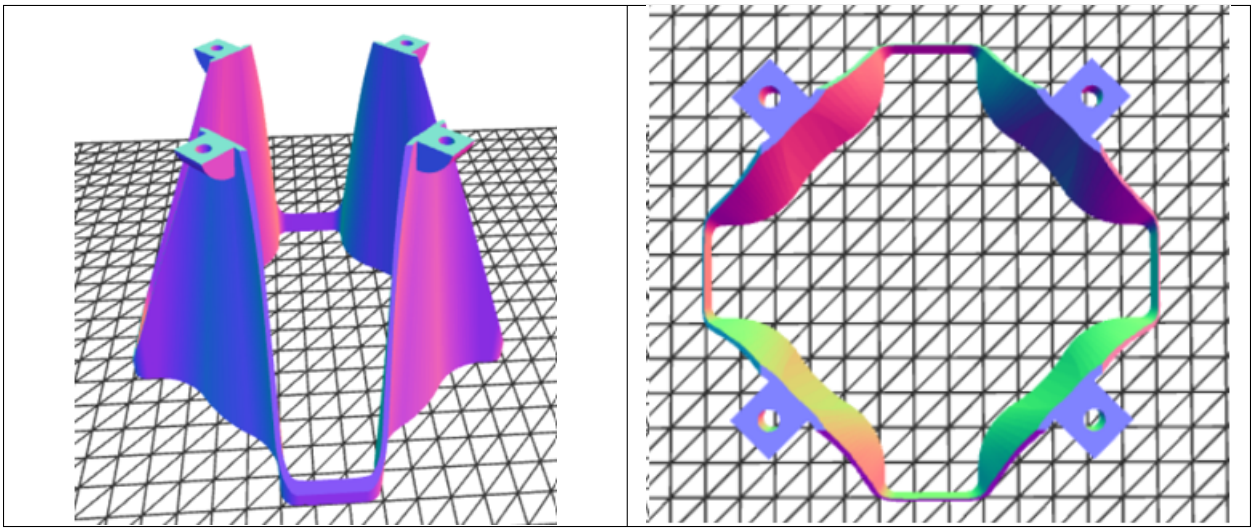
VERSION 2.0:

Bei Version 2, nahmen wir das zuvor verwendete TIM und erhöhten den Bereich in den Luft in die Rakete eindringen könnte. Die dadurch erhaltenen Ergebnisse variierten nicht stark von den aus Version 1.0.



VERSION 3.0:

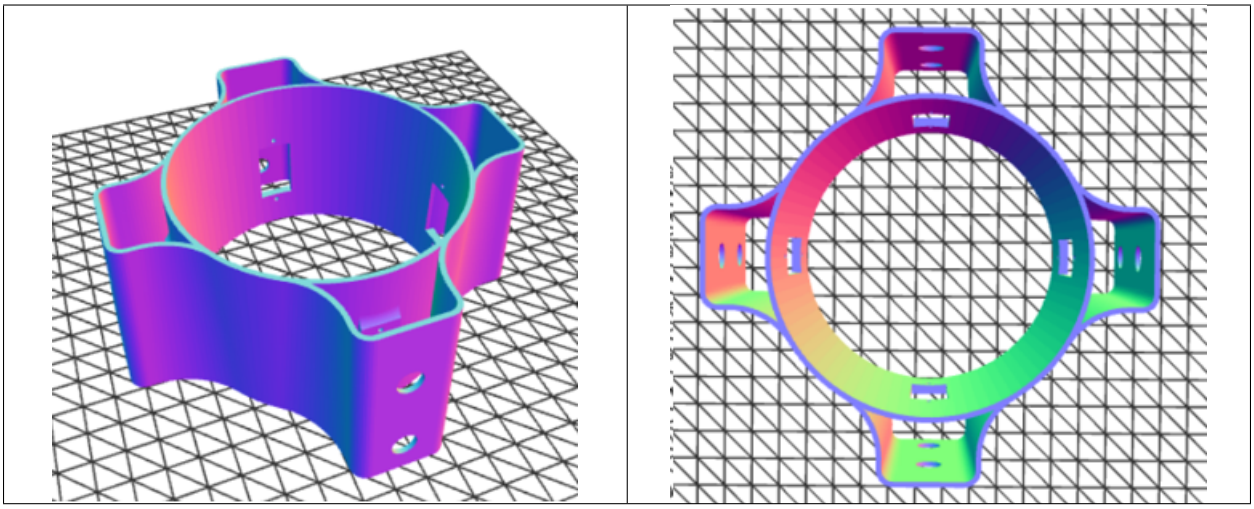
Um nun nicht wieder unsere gesamte Rakete umzu designen kam es zu einer erweiterten Version des TIM: ”Version 3.0”. Diese sprach mit dieser noch größeren und weiteren Öffnung noch mehr Luft an. Das Teil wurde zudem fast auf das doppelte des Vorherigen Designes erweitert.



TVC - Thrust Vectoring Module:

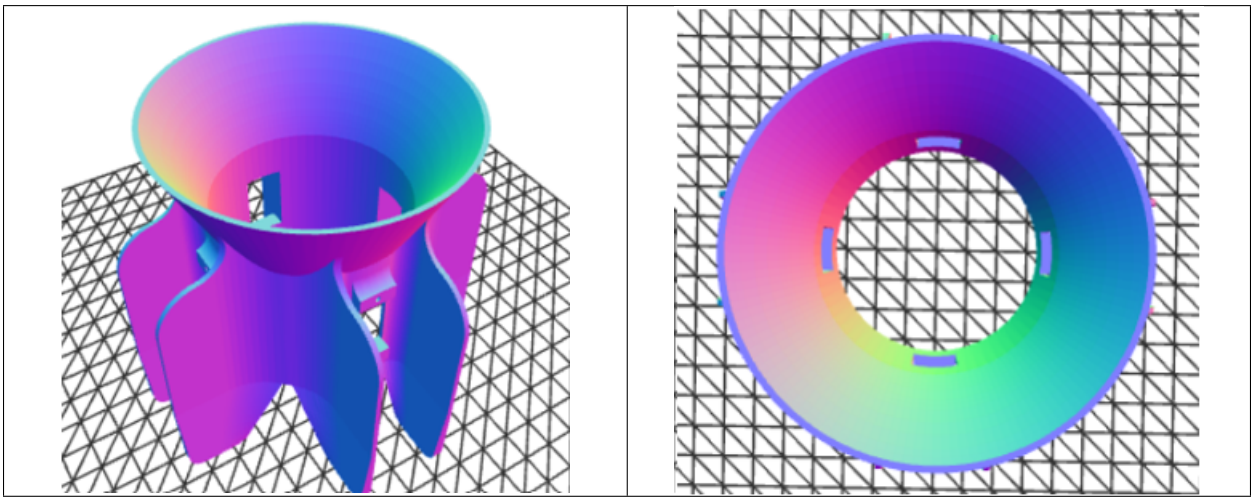
VERSION 1.0:

Das TVC - Thrust Vectoring Module der neuen Rakete unterschied sich in der Struktur und Kombination der Verwendeten Servo’s. Um nicht mehr die Aerodynamik durch die Servos zu verschlechtern, wurden diese in eigene Bereiche eingebaut, um so nicht mehr zu stören. Die benötigten Füße waren zu diesem Zeitpunkt noch nicht designend da wir uns noch in einer Testphase für das neue TVC befanden.



VERSION 2.0:

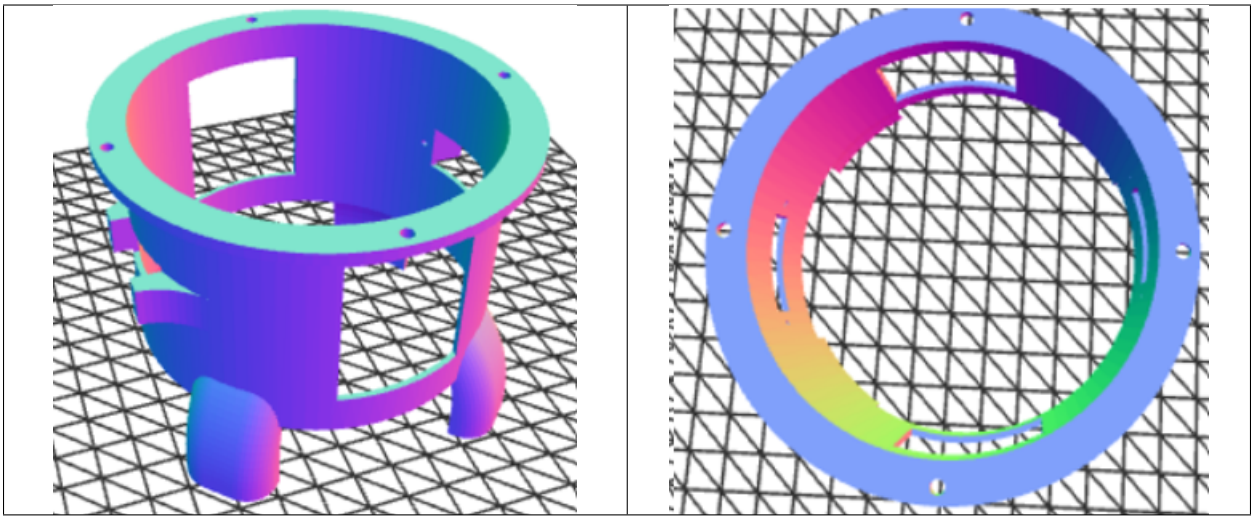
Version 2.0 des TVC hatte gewisse Vorteile gegenüber Version 1.0. Einerseits mussten wir nicht eigene Bereiche für die Servoren erstellen da diese in Design 2.0 Unter der Kreisförmigen Struktur verborgen waren. Andererseits waren die von uns benötigten Füße direkt am TVC angebunden und mussten nicht einzeln gedruckt werden. Dadurch nahm das Gewicht der Rakete ab, jedoch sankt dadurch unser Luftantrieb gewaltig ab da wir durch das offene untere Ende zu viel Luftausweich Möglichkeiten hatten.



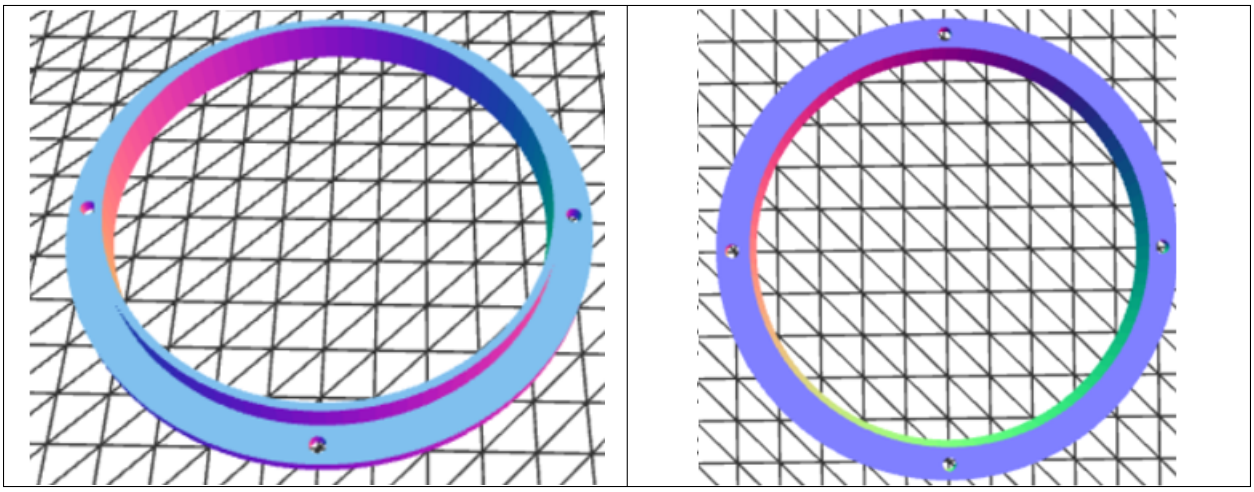
VERSION 3.0:

Version 3.0 war wegen der Verschwendeter Luft Nötig, da verglichen zu Fins eine Nozzle viel mehr Luftausgangmöglichkeiten abblockt. Der Adapter für das Main TVC wär nötig um eine Folie einzuklemmen welche und ermöglichen würde den gesamten Luftstrom auf die Nozzle umzuleiten und dadurch maximale Effizienz zu erreichen. Jedoch erlangt die Rakete durch die große Nozzle zusätzliches Gewicht. Ein großes Risiko welches wir jedoch trotzdem eingehen mussten.

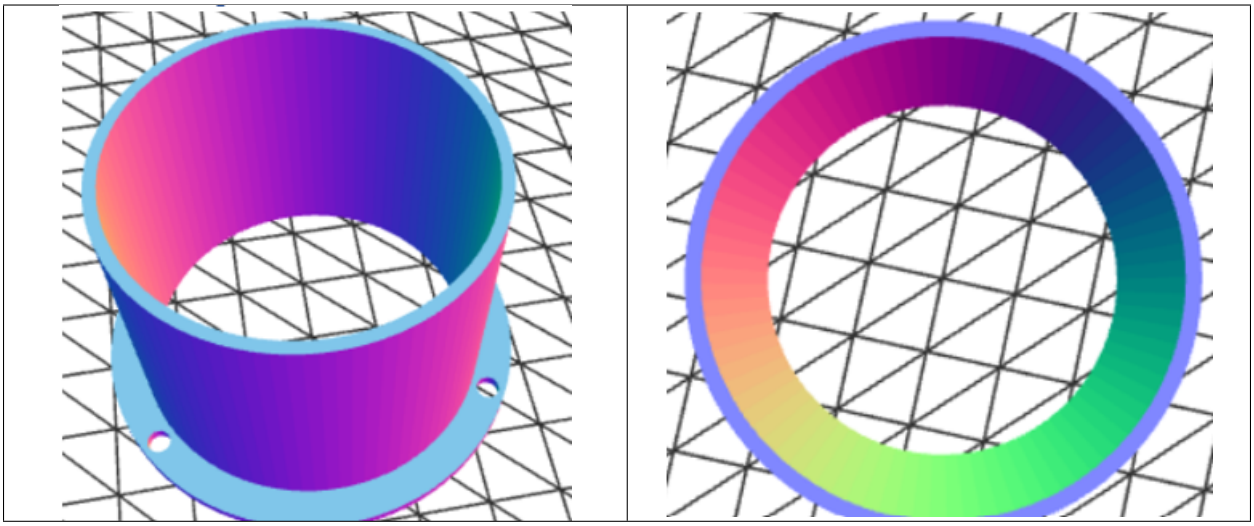
Main TVC



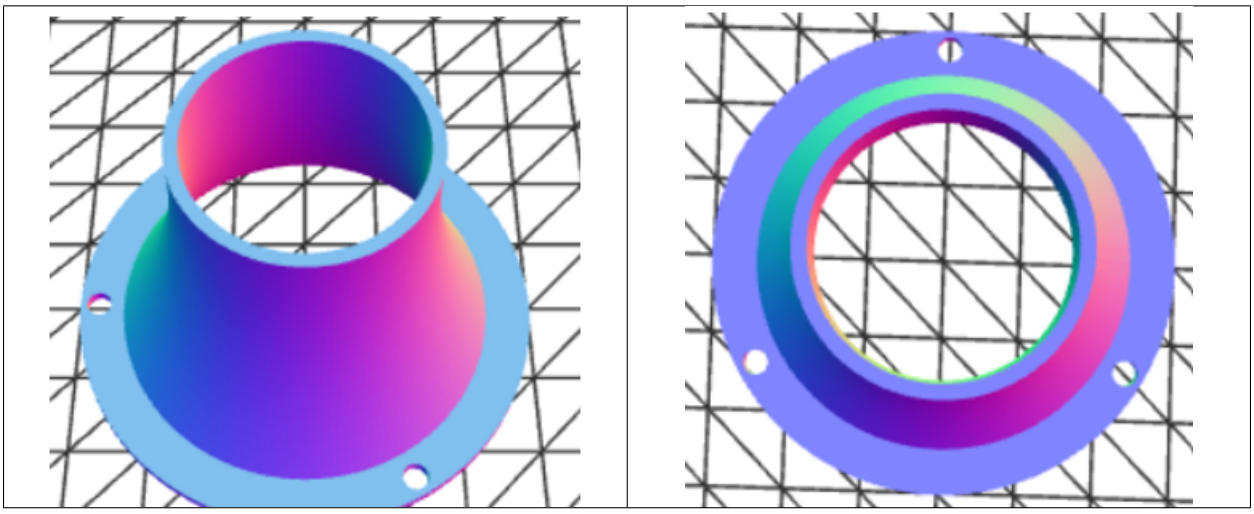
Adapter for TVC



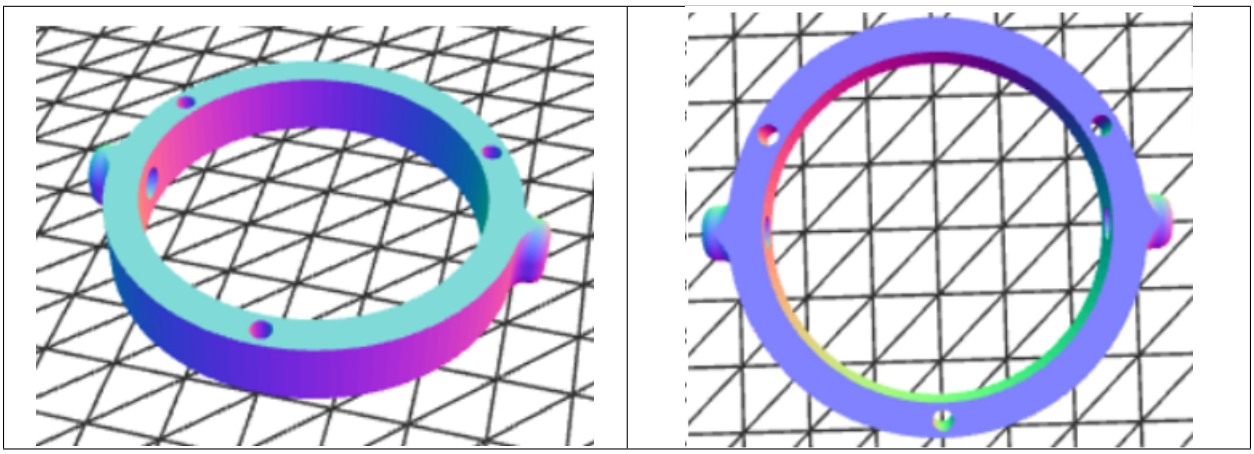
Nozzle Redesign Part 1



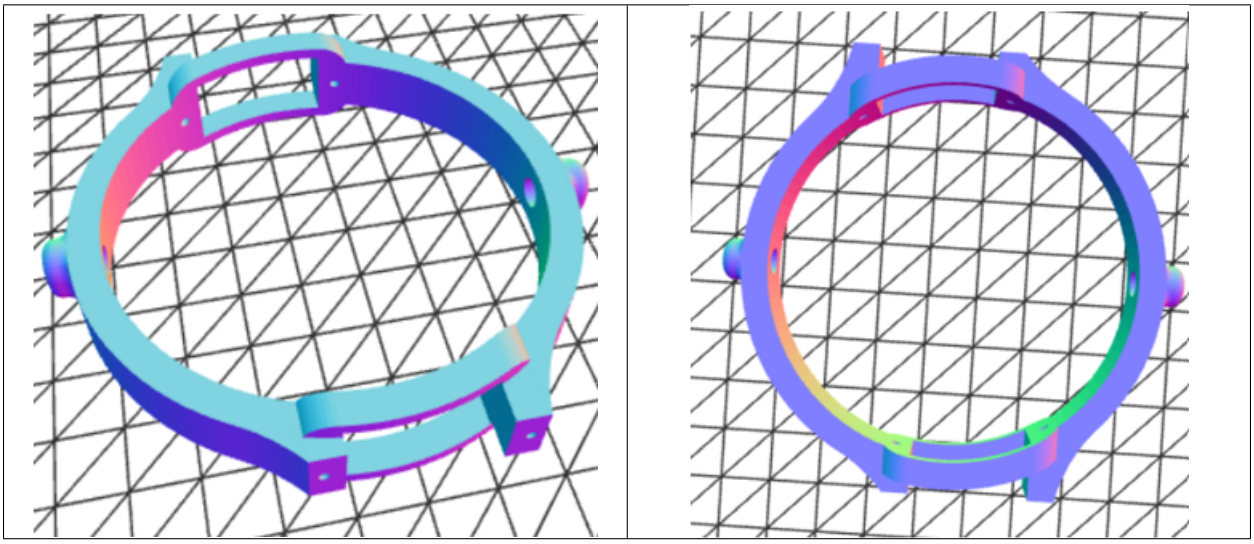
Nozzle Redesign Part 2



Nozzle Inner Ring Part 1



Nozzle Inner Ring Part 2



1.4 Design 3.0 - Endurance

1.4.1 Design 3.0 Überblick:

Was ist Endurance:

In der dritten Ausführung unserer Rakete wurde alles bisher erlernte aus den vorherigen Versionen mit einbezogen und vorallem auf eines geachtet: ***Gewicht*** Sowohl Perseverance und THe Injector waren eindeutig zu schwer, um einen stabilen Flug zu erzielen, oder überhaupt abzuheben. Also war der erste Schritt die Gewichsreduzierung aller Designs. So wurde zum Beispiel eine maximale Wanddicke von 1.2mm festgelegt, welche über die gesamte Außenhülle konstant ist.

Einige der einzelenen Sections wurden von The Injector übernommen und überarbeitet, um weiter Gewicht zu sparen.

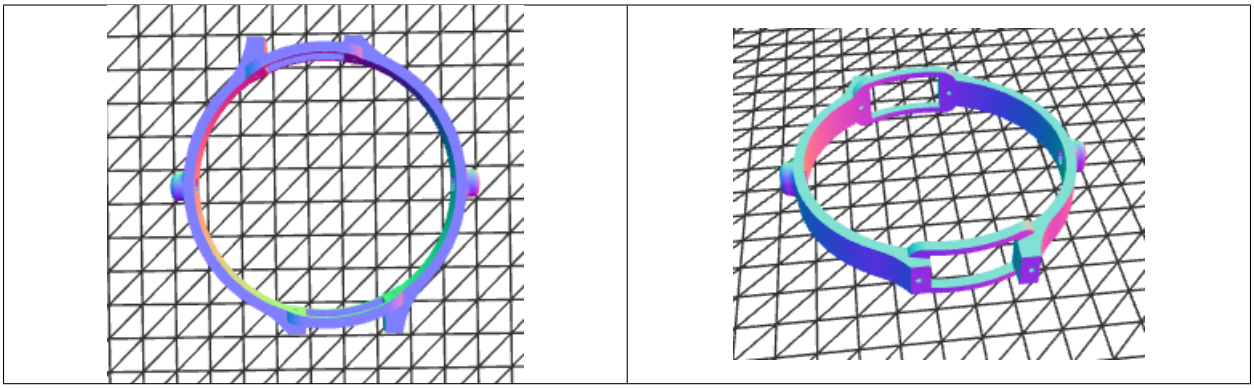
1.4.2 Design 3.0 Struktur:

TVC - Thrust Vector Control:

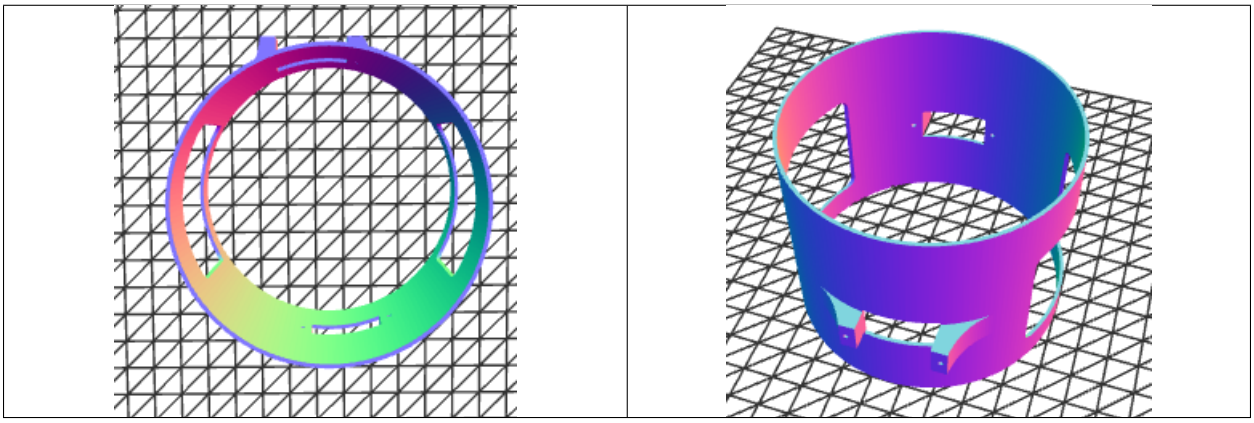
Die unterste Section ist natürlich der eigentlich wichtigste Teil der Rakete. Hier findet das Thrust Vectoring statt, welches den Luftstrom unserer zwei Motoren lenkt, um die Rakete stabil und aufrecht zu halten.

Hier kommt das bewährte System von The Injector zum Einsatz, welches noch ein wenig überarbeitet wurde, um weiter Gewicht einzusparen. Des weiteren wurde ein Seal Ring entworfen, welcher in flexiblem Material gefertigt wird, und die Nozzle fließend mit der Außenhülle verbindet. So wird sichergestellt das die Luft durch die Nozzle geleitet wird und nicht an den Seiten herrausströmt oder Verwirbelungen entstehen.

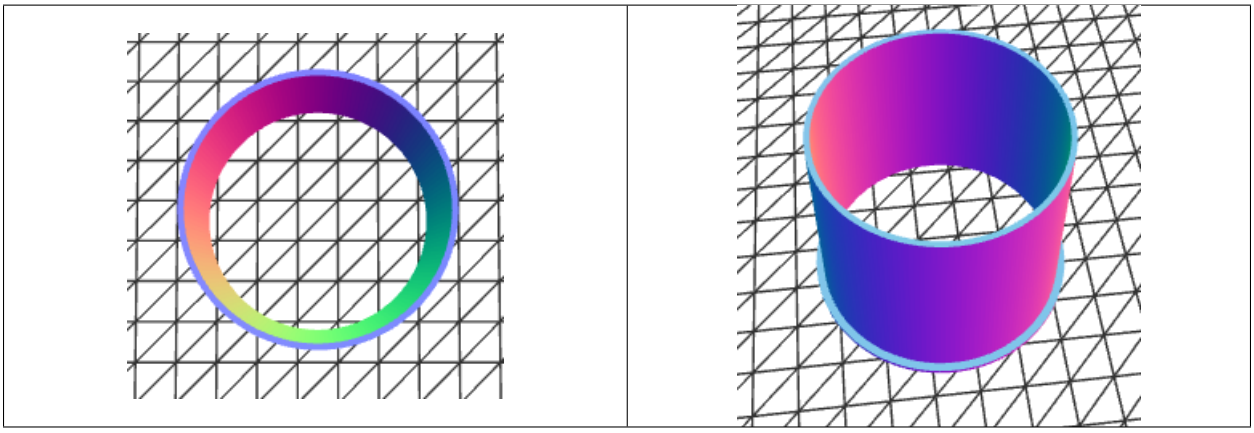
Inner Ring



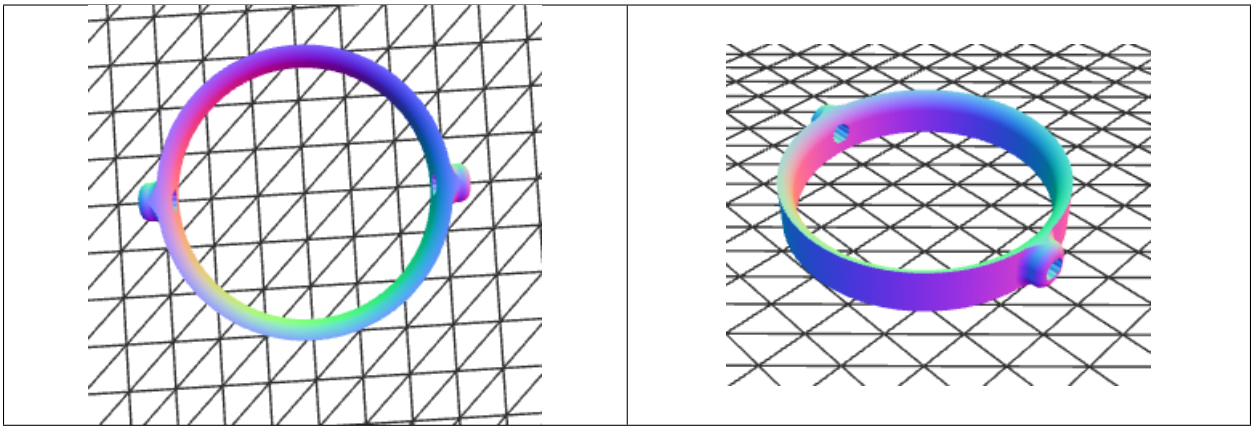
Main Section



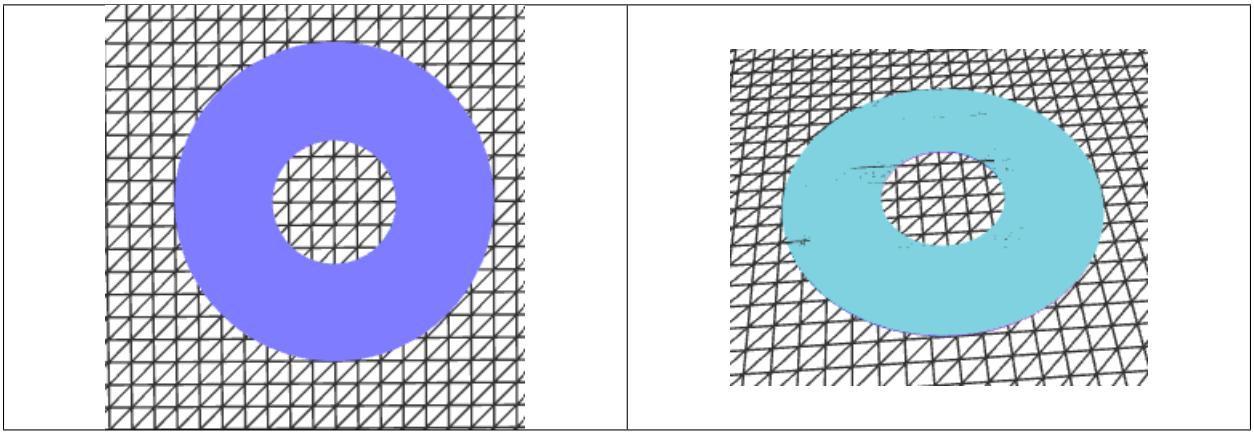
Nozzle



Nozzle Ring

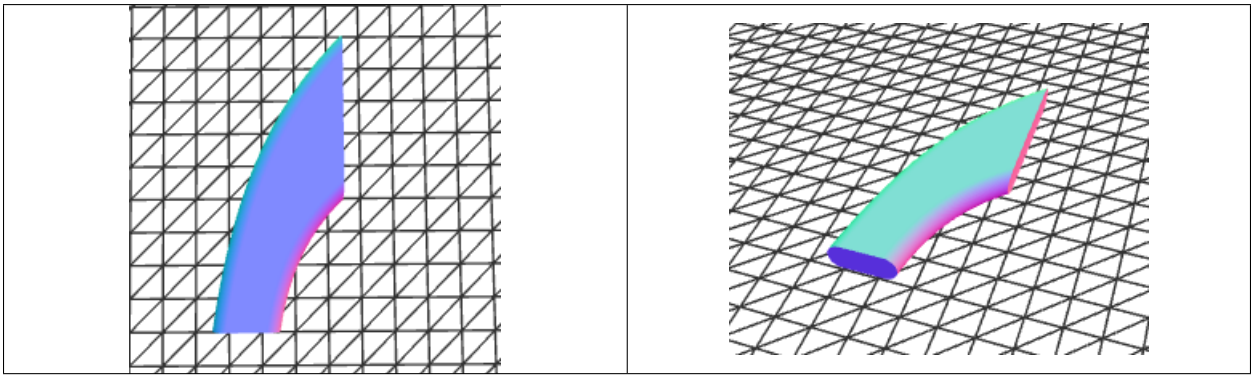


Seal Ring



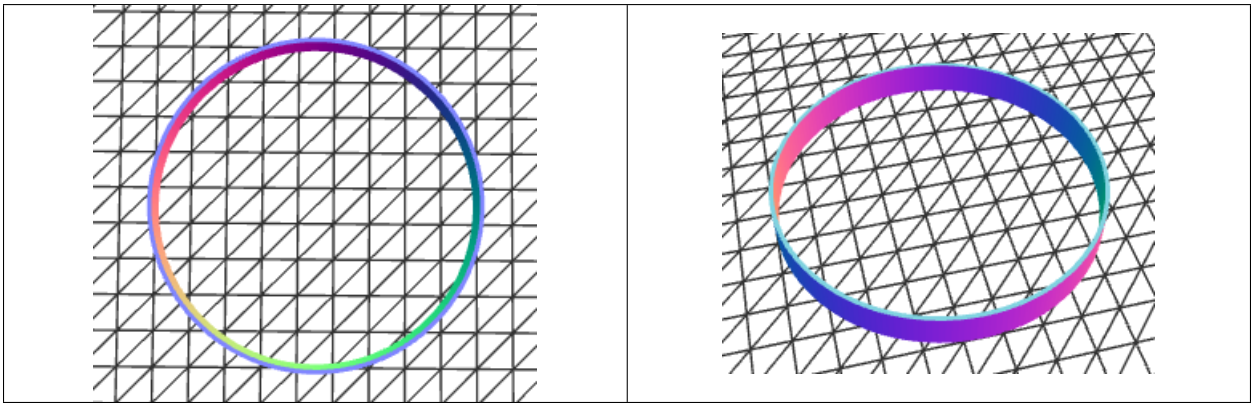
LL - Landing Legs:

Sie dienen für einen sicheren Stand der Rakete und sind wie alle anderen Teile gewichtsarm und gleichzeitig in ihrer Stabilität ausreichend.



ES - Extender Section:

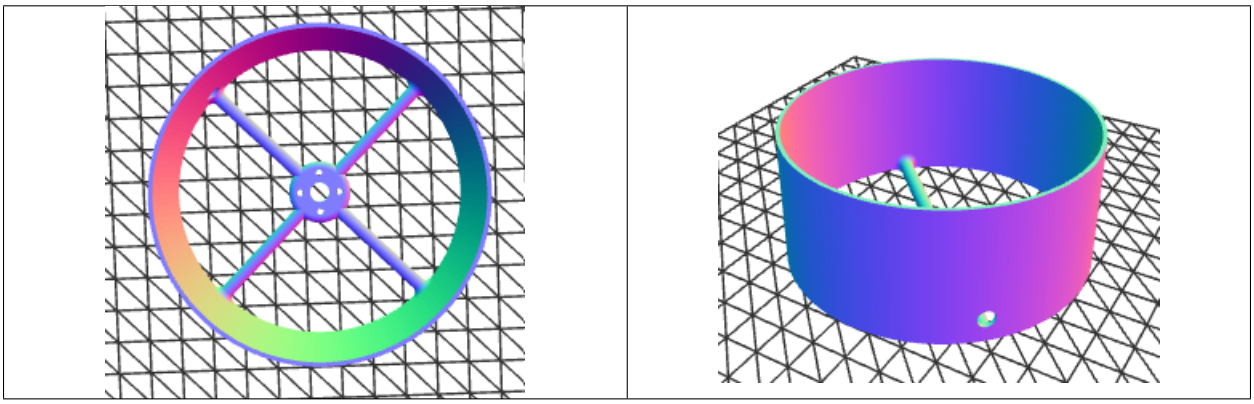
Diese Section wurde rein zur Verlängerung der Außenhülle designed. Einerseits haben wir dadurch der Thrust Vectoring Section mehr Platz beschafft und getestet, wie sich der größere Abstand der Motore zur Nozzle auf den Luftstrom auswirkt.



TS - Thrust Section:

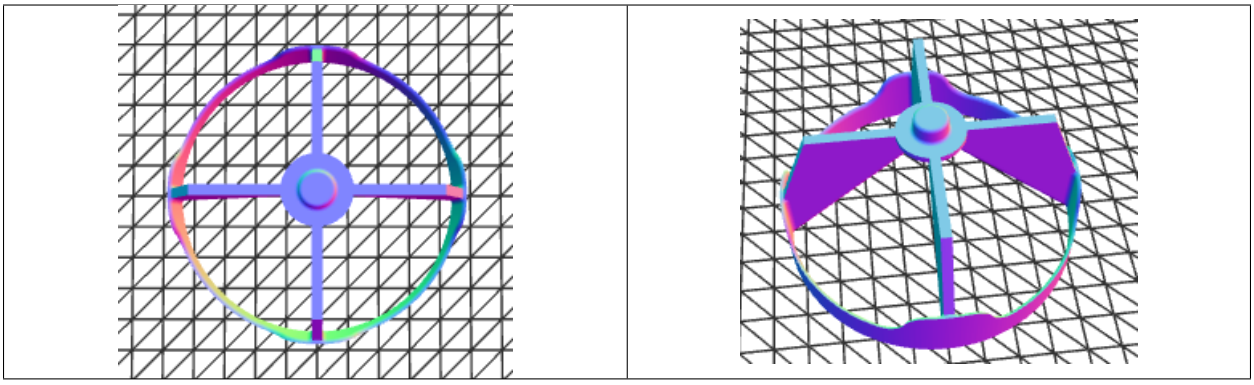
Die Thrust Section haust unseren neuen Powertrain, welcher aus zwei neuen, kleineren und leichteren Motoren besteht. Der Powertrain wurde vorallem aus Gewichtsgründen auf diese neuen Motoren umgebaut.

Das Section Design ist wiederum sehr vergleichbar mit The Injector und beinhaltet nur die neuen Anpassungen für Gewichtsreduktion und die Lochabstände für die neuen Motoren.



IS - Intake Section:

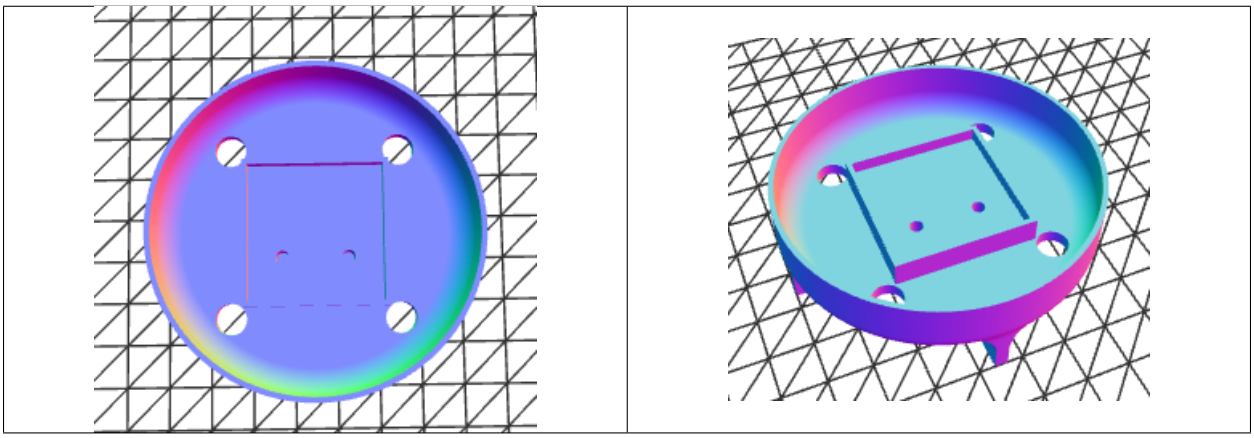
Nach mehreren Problemen mit der Luftzufuhr bei Perseverance und The Injector, musste ein neues Intake Design her. So entstand dieses Design, welches den Durchmesser der Rakete um etwa 10mm verringert. Es werden vier dünne Verstrebungen zur Mitte geführt und in einem Punkt vereint, auf welchem die nächste Section aufsetzt. So wird so viel wie möglich Fläche für die einströmende Luft geschaffen und gleichzeitig die Integrität und Stabilität wenig beeinträchtigt.



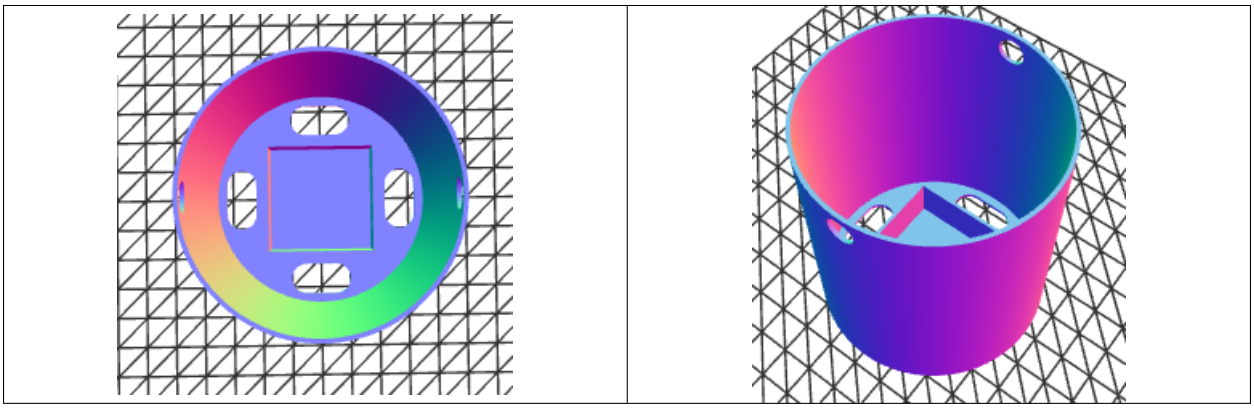
EBS - Electronic Battery Section:

Die Battery/Electronic Section behaust die gesamte Elektronik inklusive Gyro und Batterie. Sie wurde absichtlich so designed, das so wenig wie möglich Leerraum übrig bleibt. Das Design selbst ist sehr schlicht und ist sozusagen eine fließende Verlängerung der Intake Section.

Bottom

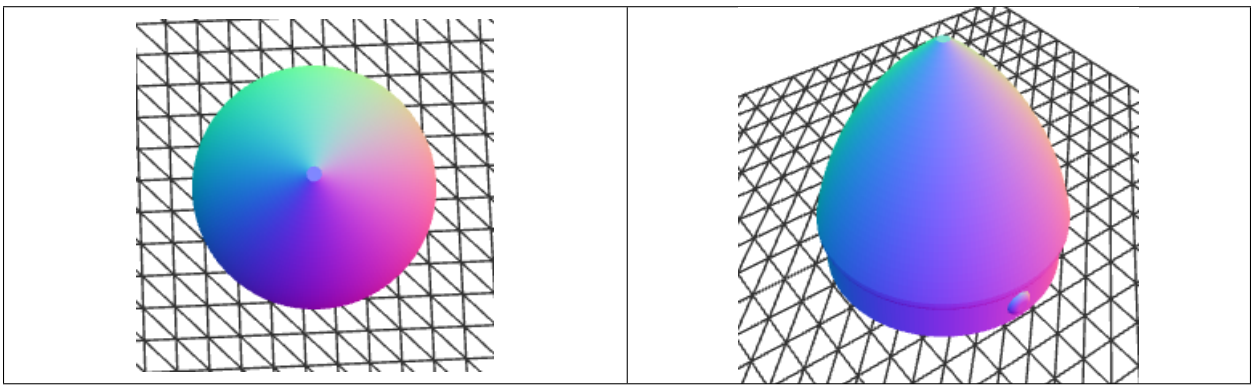


Top



NC - NoseCone:

Der Nosecone ist die letzte und oberste Section der Rakete. Sie dient zur Verschließung der EBS und ist aerodynamisch so designed, das der Luftstrom verwirbelungsfrei beleibt.



1.5 Software

1.5.1 Sprachen:

Verwendete Sprachen:

- C++ bzw. Arduino-C
- JavaScript
- Bash
- CSS
- HTML

Der Großteil des Codes ist in C++ geschrieben. Die anderen Sprachen dienen zur Ergänzung in manchen Teilbereichen des Hauptcodes.

1.5.2 Software Unterteilung:

Natürlich müssen unsere Raketen auch irgendwie in der Lage sein, sich autonom Steuern zu können. Hier kommt unsere Software inside Spiel, welche wir in zwei Bereiche aufteilen.

Flightcode:

Der Flightcode ist das Herzstück unserer Raketen. Zu den wichtigsten Komponenten zählen vorallem der PID-Controller Code welcher für die korrekte Ansteuerung der Servos zuständig ist, der LoRa-Communication Code welcher sicherstellt, dass stets Kommunikation zur Groundstation herrscht, der Power Control Code für die Ansteuerung der Motoren und natürlich der MPU Code, welcher für die korrekte Initialisierung und Ansteuerung des MPU6050 zuständig ist. Natürlich sind diese nur einige Kernkomponenten der kompletten Software, welche inzwischen über 1000 Zeilen Code umfasst.

Groundstation:

Die Groundstation ist in ihrer Funktion sehr einfach. Sie verschafft uns die Möglichkeit, Daten der Rakete während des Fluges zu empfangen. Die Station wurde ab The Injector ein fixer Bestandteil jeder unserer Raketen. Ein Sicherheitsaspekt der dadurch imlementiert wurde, ist die sofortige Terminierung des Fluges, sofern die Kommunikation zwischen Luft und Boden nicht mehr bestehen. Vor jedem Flug wird deshalb vom Code immer eine Check-Sequenz durchgangen, bei der sichergestellt wird, das die Kommunikation zwischen Luft und Boden stabil ist.

1.5.3 GitHub:

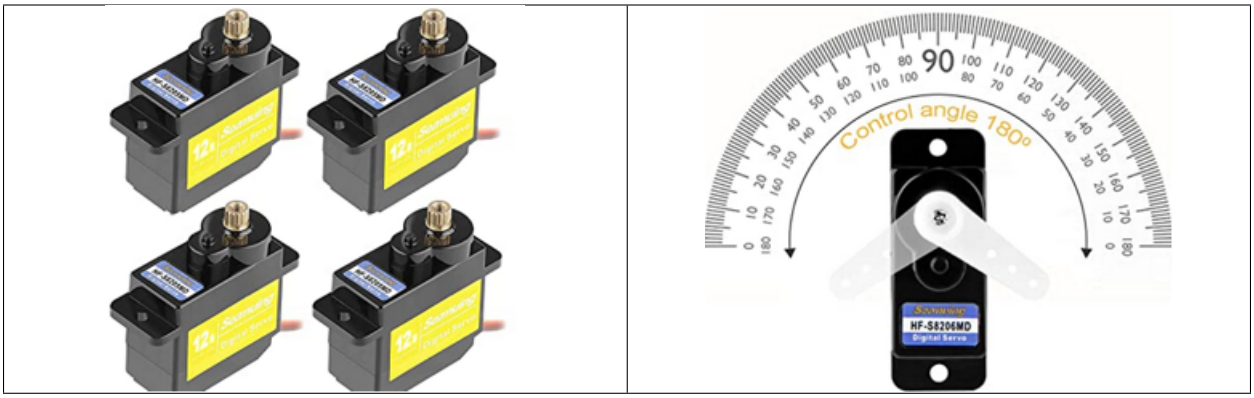
Da alle unserer Teammitglieder sehr viel Wert auf Open Source legen, wird unser gesamter Code auf GitHub zur Verfügung gestellt.

https://github.com/AetherAerospace

1.6 Hardware

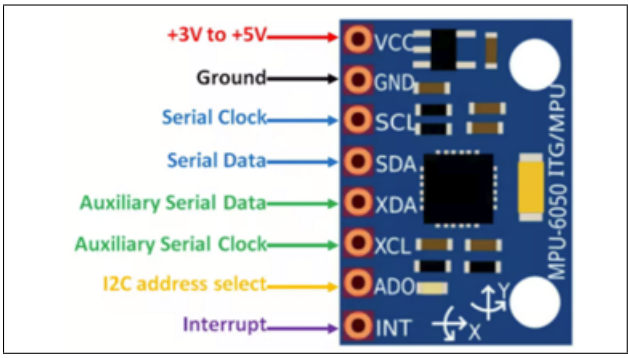
1.6.1 Micro Servos:

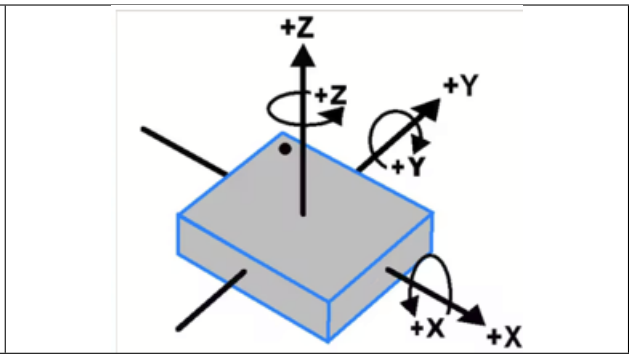
Servos werden durch das Senden eines elektrischen Impulses mit variabler Breite (Impulsbreitenmodulation) durch eine Steuerleitung gesteuert. Hierbei existiert ein minimaler Impuls, ein maximaler Impuls und eine Wiederholungsrate. Servomotoren ermöglichen uns das präzise Steuern unserer Raketen.



1.6.2 MPU6050 - Gyro:

Der MPU6050 ist ein Bewegungserfassungsgerät, welches für geringen Stromverbrauch und hohe Leistung entwickelt wurde. Zudem ist es sehr klein gehalten und ermöglicht uns viel Spielraum beim Einbau. Es besteht aus einer 16-Bit Analog-Digital-Wandler Hardware, welche es ermöglicht, Beschleunigung und Neigung auf allen drei Achsen gleichzeitig zu erfassen. Diese Rohdaten werden vom Flightcode direkt in den PID-Controller gespeist und dort weiter verarbeitet.



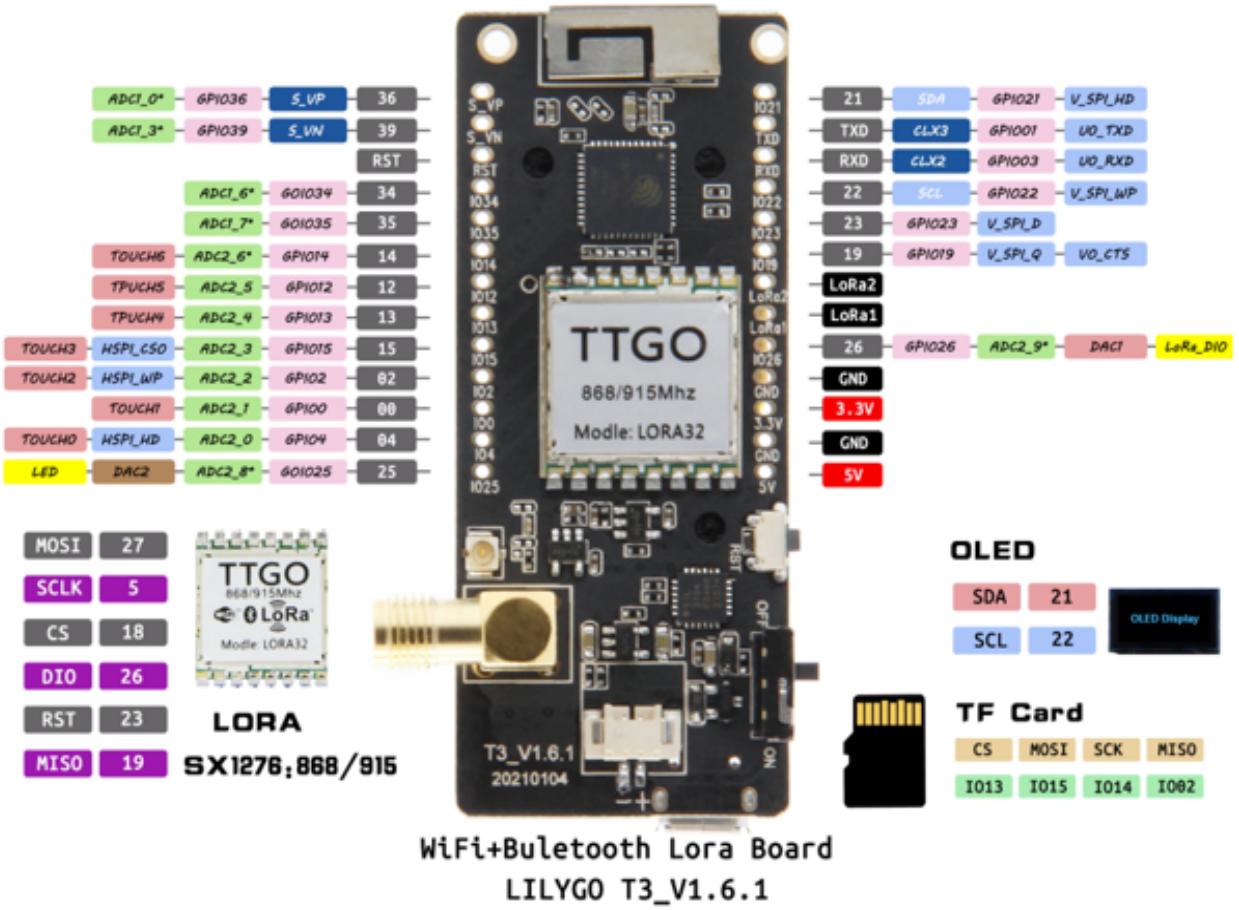


1.6.3 Motoren:

Für den Antrieb unserer Rakete verwenden wir Brushless-DC Motoren, welche heutzutage vorallem in RC-Drohnen, Baumaschinen und anderen elektronischen Geräten zum Einsatz kommen. Bei Perseverance und The Injector fiel unsrere Wahl auf 2206 2600KV Motoren, welche aber bei Endurance gegen leichtere und stärkere 1507 3600KV Motoren getauscht wurden.

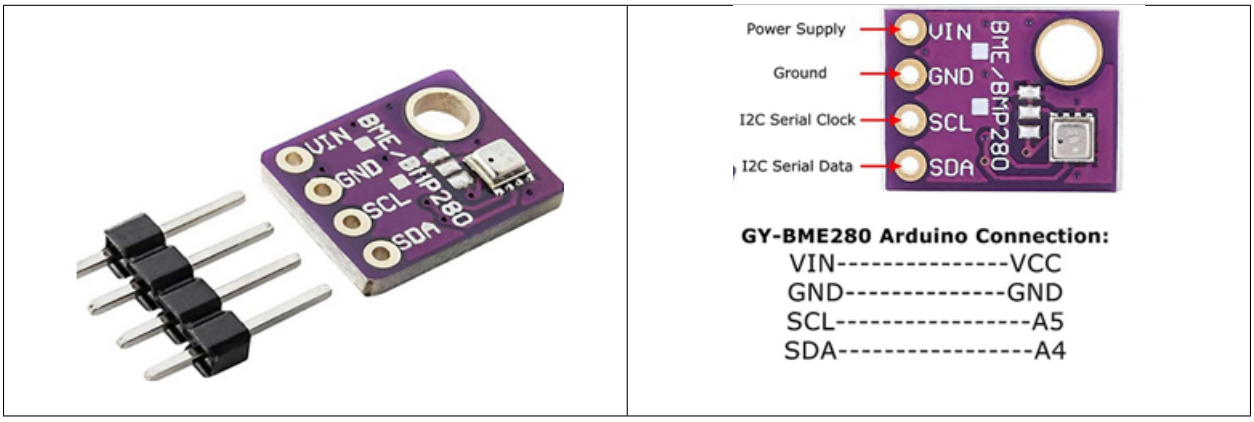
1.6.4 ESP32-LoRa:

Der TTGO Lora32 ist ein ESP32 Board, welches über ein integriertes LoRa Modul und SSD1306 OLED-Display verfügt. Der ESP32 ist ein funktionsreicher, für seine Größe leistungstarker und effizienter Mikrocontroller, weswegen er auch das Gehirn unserer Raketen ist. Auch für unsere Groundstation kommt solch ein Board zum Einsatz, um die reibungslose Kommunikation zwischen Luft und Boden sicherzustellen.



1.6.5 BMP280:

Das BMP280 Barometer ermöglicht es uns Temperatur und vorallem Luftdruck genau zu bestimmen. Durch den erfassten Luftdruck wird schließlich die aktuelle Höhe errechnet und im Flightcode mit einbezogen, um immer genaue Kontrolle über die Flughöhe zu haben.



1.7 Kommunikation

1.7.1 Kommunikation Überblick:

Zusammenfassung:

Die Kommunikation in unserem Projekt besteht aus drei Komponenten. Die Idee war es jeweils ein LoRa Module in der Rakete zu befestigen und das zweite LoRa Modul in der Groundstation anzubringen. Um dann noch mit dem Lora zu kommunizieren wurde ein Web-Interface erstellt, mit dem es uns möglich war, befehle per Knopfdruck an die Rakete zu senden und im Notfall eingreifen zu können. Das dritte Modul wird in der zweiten Stufe eingesetzt. Das verwendete Modul heißt TTGO ESP32 Lora32 V2.

TTGO Lora32:

Der TTGO Lora32 ist eine ESP32 Board, welches über ein integriertes LoRa module und einen SSD1306 LCD Display verfügt. Da es sich hierbei um 2 Geräte handelt kann man das eine Board einbauen und über das andere den aktuellen Status der Rakete abfragen.

Datenaustausch:

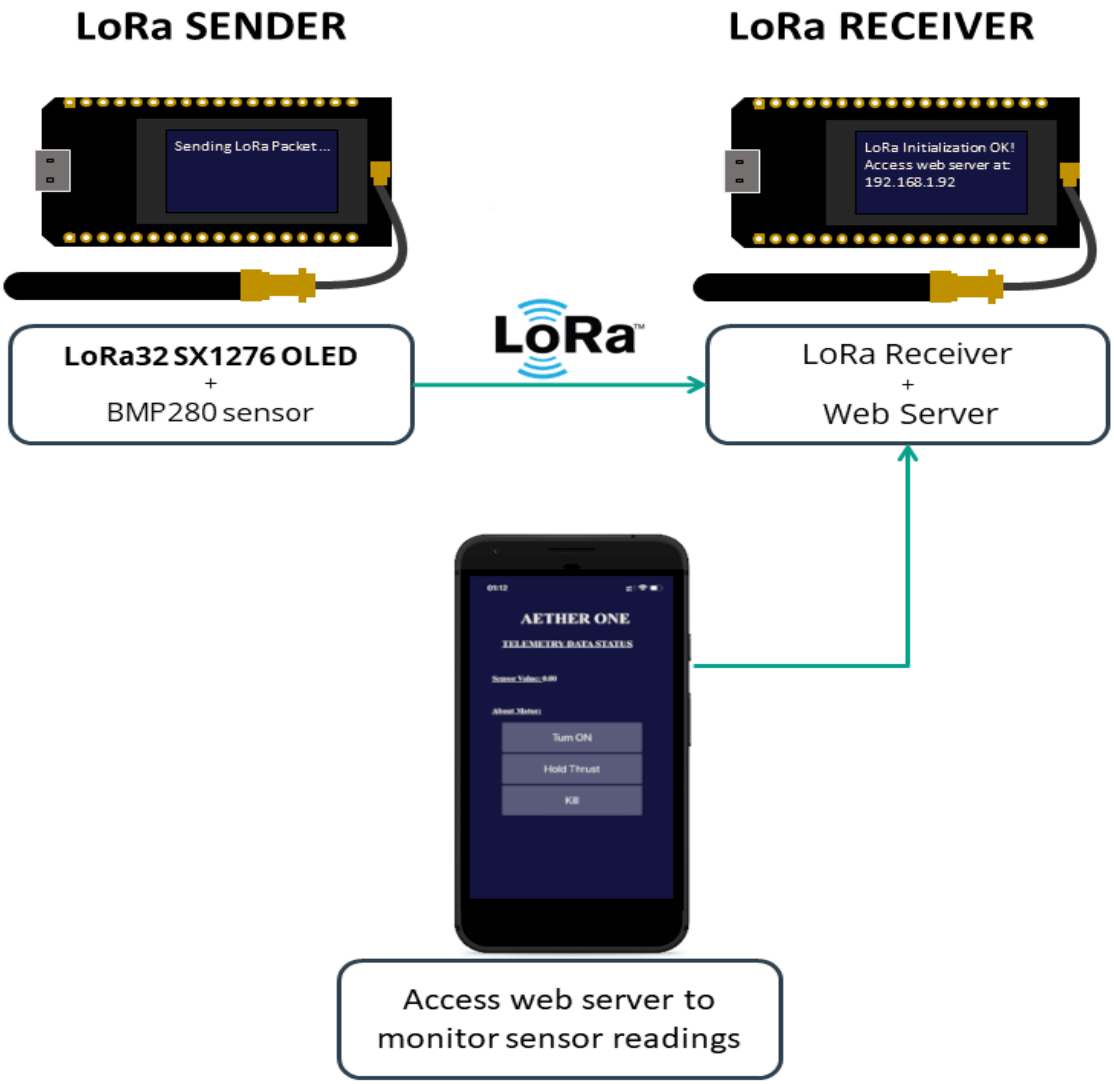
Der Datenaustausch erfolgt über eine Punkt-zu-Punkt Kommunikation, bei welcher einer der TTGO Lora32 als Sender und der andere als Empfänger fungiert. Mithilfe des Integrierten Displays erhalten wir eine Art Visualisierung der Übertragenen Daten.

Was ist LoRa:

LoRa steht für Long Range und ist ein Low-Power Wide-Area Network (LPWAN) und verwendet eine breite Palette von Funkfrequenzen. Europaweit wird 868MHz verwendet. Dadurch ist es möglich mit LoRa-Technologie Daten über eine Strecke von bis zu 10km zu übertragen.

1.7.2 Kommunikation Funktion:

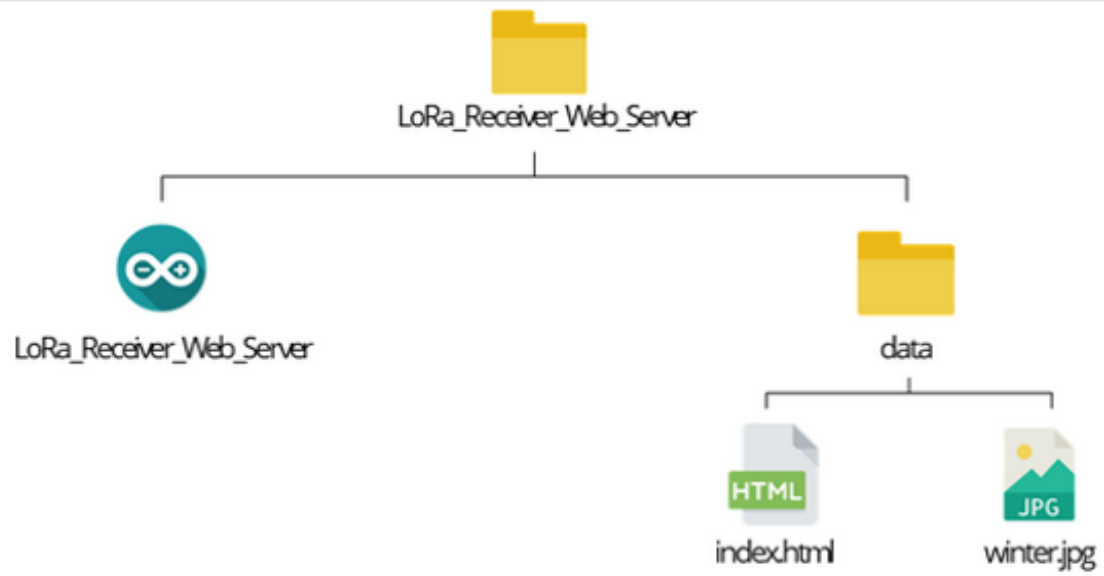
LoRa Kommunikation:



LoRa Sender:

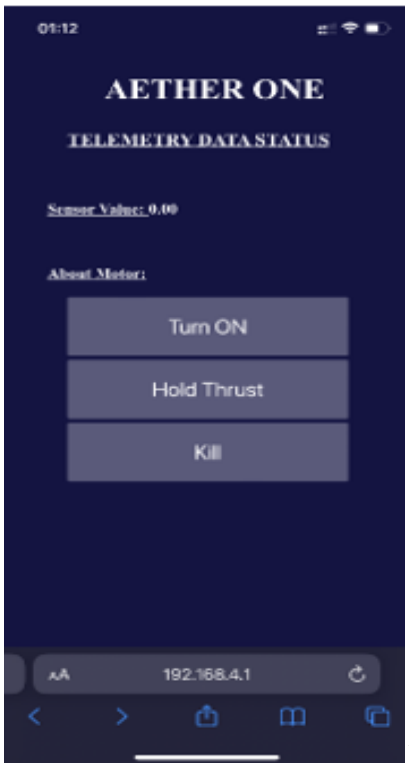
Der LoRa Sender der sich auf der Aether One Rakete befindet ist an einem ESP32 verbunden welche wiederum mit einem Gyroscope verkabbelt ist. Die vom Gyroscope erhaltenen Daten werden dann an einerseits an eine Eingebaute SD gesendet um Daten später auszuwerten und andereseits an den LoRa Sender. Die an den LoRa Sender gelangenden Daten werden dann wie in der zuvor veranschaulichten Grafik an den LoRa Receiver gesendet.

LoRa Receiver:



Der LoRa Receiver empfängt eingehende LoRa-Pakete und zeigt die empfangenen Messwerte auf einem laufendem Webserver an. Dieser laufende Webserver wird auf dem LoRa Receiver gehostet und ist durch ein erstelltes SSID verfügbar. Nun kann man sich in dieses SSID-Netzwerk mit einem von uns aus Sicherheitsgründen erstellten Password verbinden. Nun kann man über einen Webbrowser die jeweilige IP addresse des LoRa Receivers eingeben und gelangt somit auf das Web-Interface.

Web-Interface:



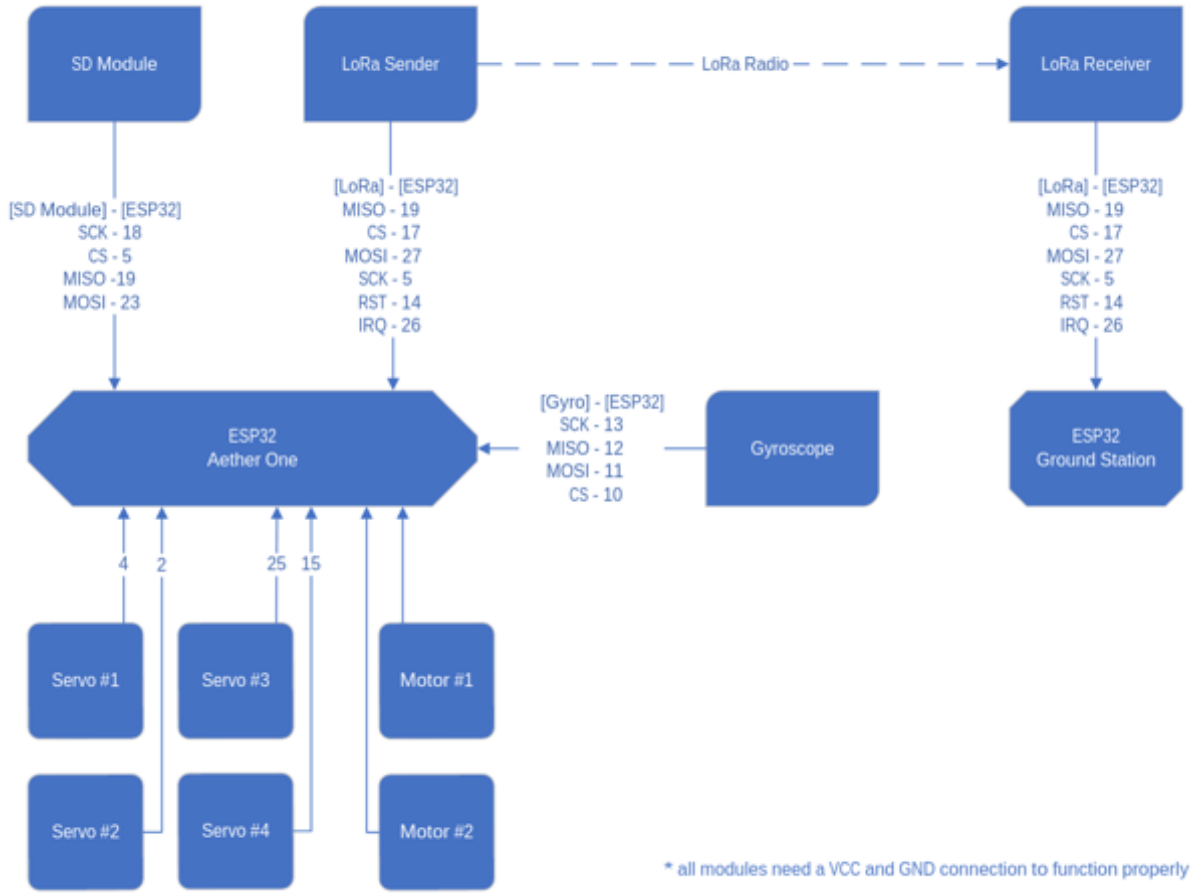
Um aktiv am LoRa Datenaustausch teilnehmen zu können, erstellten wir ein Webinterface, welches direkt auf dem LoRa Sender gehostet wird. Das Web-Interface verfügt dabei über einen Hold-Thrust und Kill-Knopf. Der Hold-Thrust Knopf verhindert, dass die Geschwindigkeit des Motors verändert wird. Der Kill-Knopf ist dafür da, jederzeit die Motoren stoppen zu können und unkontrolliertes Steigen zu unterbinden. Zudem wird wie im untenstehenden Bild zu erkennen der derzeitige Wert der Sensoren abgebildet, welche im späteren Flug in regulierten Schritten aufgenommen werden und dann auf einer SD gespeichert werden. Womit wir dann im Weiterem eine für sie Lesbare Grafik erstellen werden. Um jedoch das worst Case Scenario zu vermeiden das unser Web Interface nicht mehr auf User Input reagieren sollte bauten wir einen weiteren Kill-Knopf auf dem Lora-Receiver Selbst ein.

1.8 Flowchart

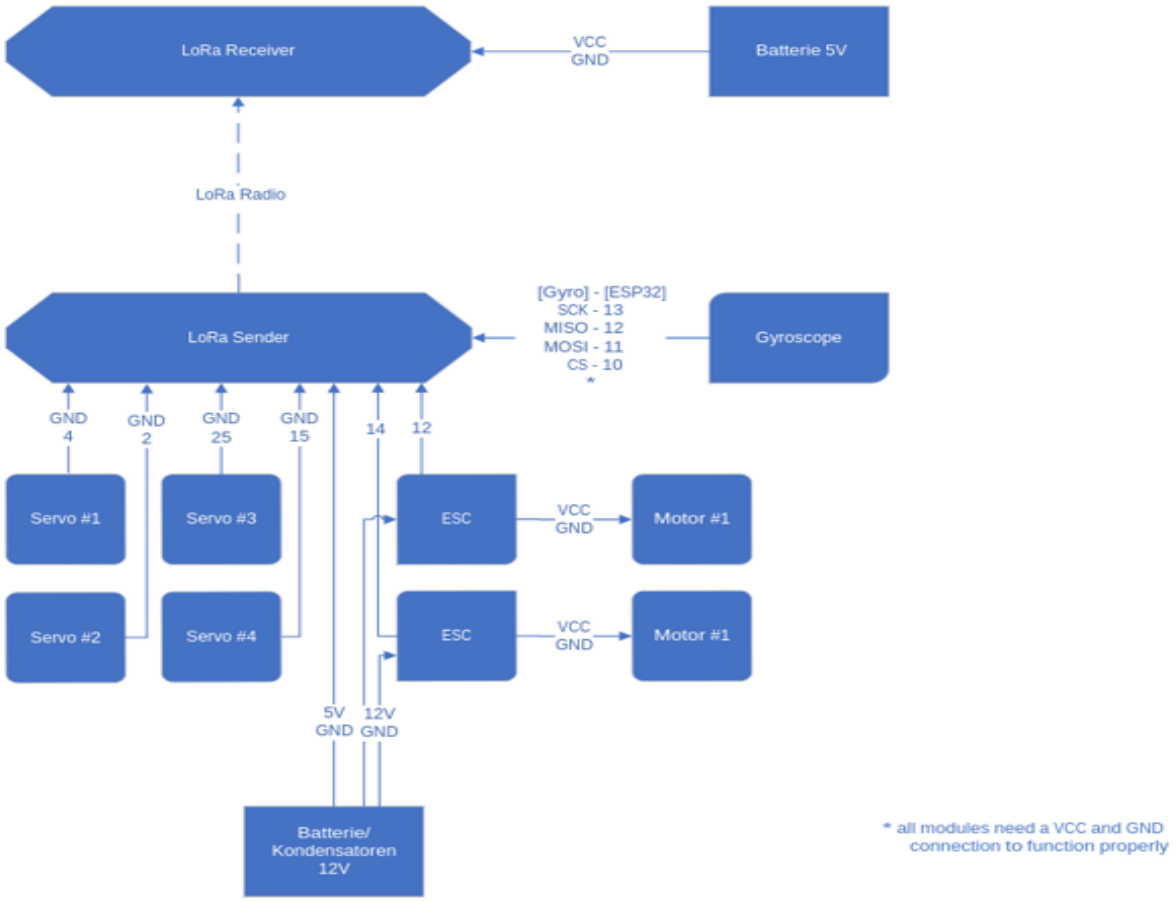
1.8.1 Was ist ein Flowchart:

Das Flowchart beschreibt die verwendete Hardware (z.B. Motoren, LoRa-module) und wie sie miteinander verbunden ist. Für weitere Information kann im Bereich 7.0 Hardware weiteres über die Hardware der einzelnen Geräte herausgefunden werden.

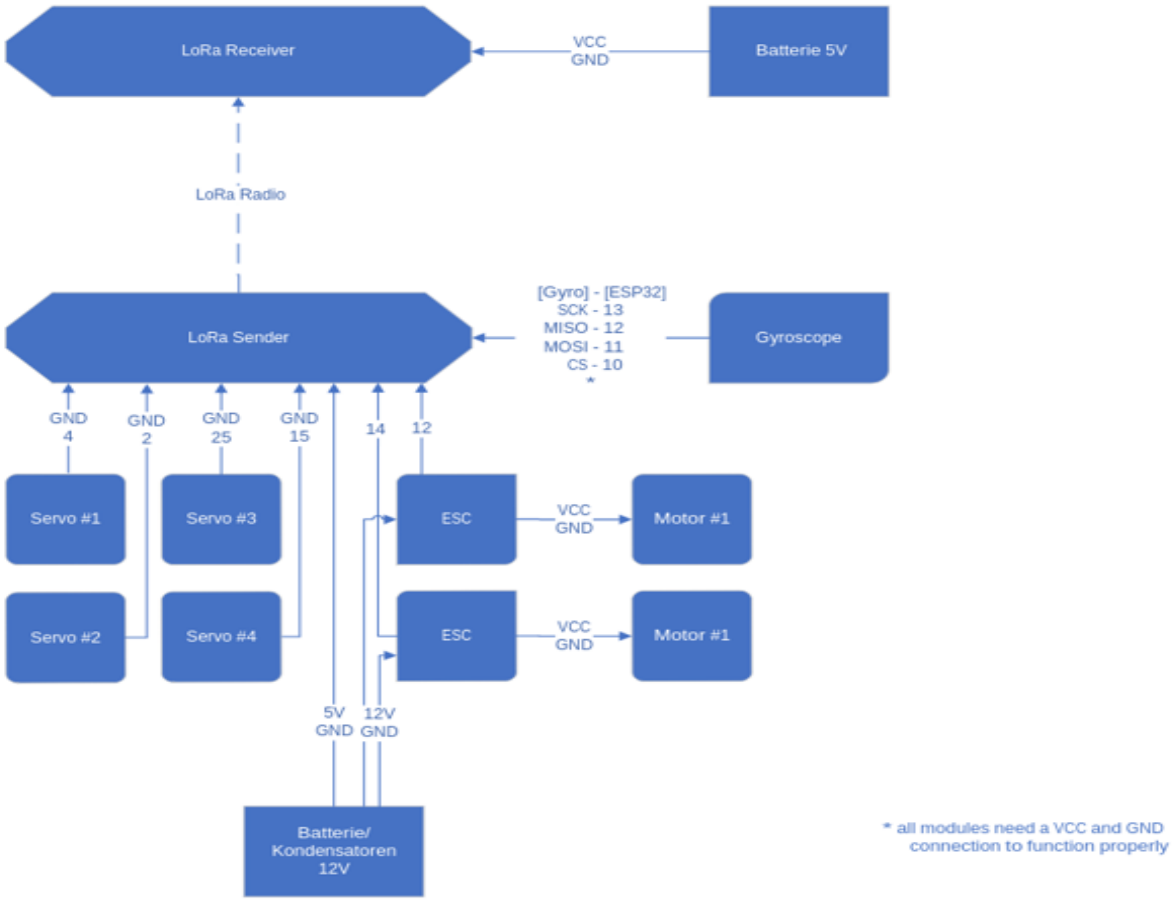
1.8.2 Perseverance:



1.8.3 The Injector:



1.8.4 Endurance:



1.9 Resümee

1.9.1 Was ist gut gelaufen:

- Kommunikation
- Zusammenarbeit
- Zeitmanagement
- Motivation

1.9.2 Wo gab es Probleme:

- Arbeitsaufwand einschätzen
- Teilweise Zeitmanagement

1.10 Ausblick

Wir können mit Sicherheit sagen, dass unser Projekt mit Abstand das schwierigste Projekt in Bezug auf die Umsetzung war. Zuerst waren wir uns nicht sicher ob wir dieses Projekt auf uns nehmen sollen, da wir im Prinzip wenig bis gar keine Erfahrung mit Modellraketen hatten. Während dem Entwicklungsprozess haben wir sehr viel Zeit darin investiert, die Physik und Funktion hinter Raketen kennenzulernen und scheuten nicht davor, Neues zu lernen. Eine Rakete zu entwickeln, benötigt sehr viel Fine-Tuning, Durchhaltevermögen, Wissen und vor allem Zeit.

1.11 License

(continues on next page)

(continued from previous page)

```
  For example, if you distribute copies of such a program, whether
gratis or for a fee, you must pass on to the recipients the same
freedoms that you received.  You must make sure that they, too, receive
or can get the source code.  And you must show them these terms so they
know their rights.

  Developers that use the GNU GPL protect your rights with two steps:
(1) assert copyright on the software, and (2) offer you this License
giving you legal permission to copy, distribute and/or modify it.

  For the developers' and authors' protection, the GPL clearly explains
that there is no warranty for this free software.  For both users' and
authors' sake, the GPL requires that modified versions be marked as
changed, so that their problems will not be attributed erroneously to
authors of previous versions.

  Some devices are designed to deny users access to install or run
modified versions of the software inside them, although the manufacturer
can do so.  This is fundamentally incompatible with the aim of
protecting users' freedom to change the software.  The systematic
pattern of such abuse occurs in the area of products for individuals to
use, which is precisely where it is most unacceptable.  Therefore, we
have designed this version of the GPL to prohibit the practice for those
products.  If such problems arise substantially in other domains, we
stand ready to extend this provision to those domains in future versions
of the GPL, as needed to protect the freedom of users.

  Finally, every program is threatened constantly by software patents.
States should not allow patents to restrict development and use of
software on general-purpose computers, but in those that do, we wish to
avoid the special danger that patents applied to a free program could
make it effectively proprietary.  To prevent this, the GPL assures that
patents cannot be used to render the program non-free.
```