Blimp

Planung, Konstruktion und Programmierung
eines ferngesteuerten Prallluftschiffes

Eine Maturaarbeit von
Hans Kieninger,
Keanu Gleixner und
Paul Fouché

Inhaltsverzeichnis

1 Einleitung

Ziel unseres Projektes war ein ferngesteuertes Prallluftschiff zu bauen.

Wir experimentierten mit unterschiedlichen Materialien und Verfahren, um eine möglichst gasdichte Hülle zu fabrizieren. Die Hülle dient als Grundgerüst, an dem die Ruder und Gondel samt Motoren und Sensoren befestigt werden. Durch einen leichten Überdruck erhält die Hülle Stabilität und ihre Form, welche durch einen Kompromiss zwischen einem kleinem Oberflächen-zu-Volumen-Verhältnis und Aerodynamik entstand. Des Weiteren überlegten wir uns verschiedene Steuerungsmechanismen mit deren Vor- und Nachteile.

Eine grosse Herausforderung bestand darin, das Gewicht des Luftschiffes zu reduzieren. Denn ein geringeres Gewicht erlaubt uns die Hülle kleiner zu dimensionieren und somit an teurem Traggas zu sparen.

Als zentrale Steuerungseinheit am Luftschiff fungiert ein Raspberry Pi Zero W. Er empfängt über WLAN die Steuersignale, verarbeitet diese und betätigt die Motoren und Servos. Ausserdem liest er die in der Gondel befestigten Sensoren aus. Das Luftschiff ist mit zwei Kameras, einem GPS, einem Beschleunigungssensor, einem Gyroskop, einem Barometer, einem Kompass, einem Analogdigitalwandler für die Batteriespannung und zwei Ultraschall-Distanzsensoren ausgestattet. Um die vorher genannten Aufgaben zu erfüllen, schrieben wir ein Programm in C++. Anhand der Sensordaten liesse sich die Lage und Orientierung des Zeppelins berechnen und somit ein Autopilot programmieren. Als Fernsteuerung dient ein Java Programm, das zusätzlich die Sensordaten und die Videostreams anzeigt. Als Traggas entschieden wir uns Helium, statt Wasserstoff zu verwenden, damit wir auch innerhalb von Gebäuden fliegen dürfen. Da Helium relativ teuer ist, suchten wir nach einem Sponsor. «Lillo’s Fitness-Träff» willigte ein und stellte uns das Helium zur Verfügung.

Unser Luftschiff wiegt samt Hülle leicht über 450 g, ist rund 2.40 m lang und hat einen Durchmesser von 60 cm. Dank dem geringen Gewicht dürfen wir rechtlich über Menschenansammlungen fliegen. Zudem kann das Luftschiff im Vergleich zu einer Drohne länger in der Luft bleiben.

2 Recherche

Um eine Vorstellung zu erhalten, suchten wir auf dem Internet nach ähnlichen Projekten. Die wichtigste Inspirationsquelle ist der Silentrunner, ein ferngesteuerter Zeppelin, der von Studenten entworfen und gebaut wurde. Das Projekt ist unter auf http://silent-runner.net/index.php?title=Main_Page ausführlich dokumentiert. In Diskussionen von verschiedenen RC-Foren sind wir auf weitere nützliche Hinweise und Ideen gestossen.

3 Traggas

Der Auftrieb des Zeppelins basiert auf dem Dichteunterschied zwischen der Umgebungsluft und dem Traggas (Gesetz des Archimedes). Das Traggas muss eine geringere Dichte als Luft aufweisen. Dies erfüllen zum Beispiel Wasserstoff, Helium, Ammoniak, Methan oder Neon.

Ausgangswerte aus dem Periodensystem der Elemente.

Gas	Dichte (bei 20°C und 1 bar)
Luft 21% (O_2) + 79% (N_2)	1.20 kg/m³
Wasserstoff (H_2)	0.084 kg/m³
Helium (He)	0.17 kg/m³
Ammoniak (NH_3)	0.73 kg/m³
Methan (CH_4)	0.66 kg/m³
Neon (Ne)	0.90 kg/m³

Weil die Differenz der Dichte zwischen Luft und den meisten dieser Gase gering ist, kommen nur Wasserstoff oder Helium als Traggas in Frage. Wasserstoff ist weniger Dicht und preislich günstiger als Helium. Ausserdem könnte es durch eine Elektrolyse selber hergestellt werden. Es birgt jedoch in Kombination mit Sauerstoff die Gefahr zu brennen oder einer Explosion. Da wir unser Zeppelin auch in Gebäuden betreiben wollen, entschieden wir uns als Traggas Helium zu verwenden.

Warmluft

Der Auftrieb könnte statt mit einem Traggas auch mit Warmluft realisiert werden. Bei 100° Celsius beträgt die Dichte von trockener Luft etwa 0.95 kg/m³. Da der Unterschied der Dichte, bei Temperaturen die eine Hülle aushält, zur Umgebungsluft recht klein ist, würde man für den selben Auftrieb ein grösseres Volumen als mit Helium benötigen.

4 Hülle

4.1 Volumen

Damit der Zeppelin nicht davonfliegt oder am Boden bleibt, muss dessen Gewichtskraft der Auftriebskraft entsprechen.

$$F_{Auftrieb} = F_{Gewicht}$$

$$\rho_{Luft} \cdot V_{Traggas} \cdot g = \left(m_{Nutzlast} + \rho_{Traggas} \cdot V_{Traggas}\right) \cdot g$$

$$\left(\rho_{Luft} - \rho_{Traggas}\right) \cdot V_{Traggas} = m_{Nutzlast}$$

$$V_{Traggas} = \frac{m_{Nutzlast}}{\rho_{Luft} - \rho_{Traggas}}$$

Dichte der Luft

Das Volumen des Zeppelins hängt von der Dichte der Luft und des Traggases ab. Die Dichte der Luft hängt wiederum von der Temperatur, dem Druck und der Luftfeuchtigkeit ab. Feuchte Luft ist leichter als trockene Luft, da Wasserdampf eine geringere Dichte als Luft hat. (1 mol Gas benötigt bei 20°C 24 Liter Platz und Wasser hat eine kleinere Molmasse als Stickstoff oder Sauertstoff)

$$\rho_{Luft} = \frac{p}{R_{Luft} \cdot T}$$

Wobei die Gaskonstante R_{Luft} von der Luftfeuchtigkeit abhängt.

Zeichen	Erklärung	Einheit / Wert
p	Umgebungsdruck	Pascal
T	Temperatur	Kelvin
R_{Luft}	Gaskonstante der trockenen Luft	287.058 Joule / (Kilogramm * Kelvin)

Abhängigkeiten des Volumen

Nicht nur die Dichte der Luft sondern auch die Dichte unseres Traggases hängt von der Temperatur und dem Druck ab. Diese Abhängigkeit lässt sich ebenfalls mit dem Gasgesetz formulieren..

$$(\rho_{Luft} - \rho_{Traggas}) \cdot V_{Traggas} = m_{Nutzlast}$$

$$\left(\frac{p}{T \cdot R_{Luft}} - \frac{p}{T \cdot R_{Traggas}}\right) \cdot V_{Traggas} = m_{Nutzlast}$$

$$\frac{1}{R_{Luft} - R_{Traggas}} \cdot \frac{p \cdot V}{T} = m_{Nutzlast}$$

Nun ist die Abhängigkeit unseres Volumen vom Luftdruck und der Temperatur klar ersichtlich. Der Einfluss der Luftfeuchtigkeit befindet sich in der Gaskonstante der Luft. Sie spielt aber im Vergleich zu den anderen Faktoren eine vernachlässigbare Rolle. Halbiert sich der Luftdruck, muss sich das Volumen verdoppeln, weil die Nutzlast konstant ist. Wollten wir unseren Zeppelin in grösseren Höhen über Meer betreiben, müsste das Volumen dementsprechend grösser ausfallen. Der Zeppelin muss also für eine bestimmte Region entworfen werden. Die benötigte Anzahl Helium Teilchen ist im Gegensatz zum Volumen unabhängig von der Region.

Im Praktischen

Unsere Erfahrungen haben gezeigt, dass sich die Hülle bei weitem nicht auf den Liter genau bauen lässt. Des weiteren schwanken Luftdruck und Temperatur Wetter bedingt. Dies stellen aber schlussendlich keine Probleme dar. Bei zu grossem Auftrieb lässt sich der Zeppelin einfach beschweren, indem man mit Klebeband Münzen aufklebt. Ein leicht zu kleiner Auftrieb lässt sich ebenfalls durch die Kraft der Motoren kompensieren.

4.2 Form

Die Form des Zeppelins entsteht aus dem Kompromiss der folgenden drei Punkte:

- kleines Verhältniss von Oberfläche zu Volumen
  - führt zu geringerem Gewicht der Hülle
  - führt zu tieferem Gasverlust
  - ideale Form Kugel
- kleine Querschnittsfläche und kleiner c_w Wert
  - erlaubt schnellere Fluggeschwindigkeiten
  - erlaubt längere Flugzeiten
  - ideale Form Stromlinienkörper
- einfach zu bauen und zu berechnen

Illustration 1: Stromlinienkörper

Die Form der Hülle unseres Zeppelins ist von einer Gertler Form abgeleitet. Das Verhältniss Länge zu Durchmesser beträgt 4 zu 1. Die Gertler Form wurde um 1950 vom US-Militär für U-Boote entwickelt und wird auch für den Silentrunner verwendet. Sie zeichnet sich sowohl durch einen tiefen Luftwiderstandskoeffizienten und ein gutes Verhätniss von Volumen zu Oberfläche aus. Ausserdem lässt sich der Querschnitt der Form als mathematische Funktion beschreiben. Die Form ist der Rotationskörper des Graphen der Wurzel einer Polynomfunktion sechsten Grades im Bereich zwischen 0 und 1:

$$g(x) = \sqrt{a_6 \cdot x^6 + a_5 \cdot x^5 + a_4 \cdot x^4 + a_3 \cdot x^3 + a_2 \cdot x^2 + a_1 \cdot x}$$

Die Parameter sind:

a_6	a_5	a_4	a_3	a_2	a_1
19.621670	-56.480030	59.357210	-26.960996	3.462153	1.000000

Der Graphenteil wird noch in die gewünschte Länge gestreckt und mit dem Durchmesser multipliziert:

$$f(x)=D\cdot g(x/L)$$

Der Querschnitt des Zeppelins sieht dann folgendermassen aus:

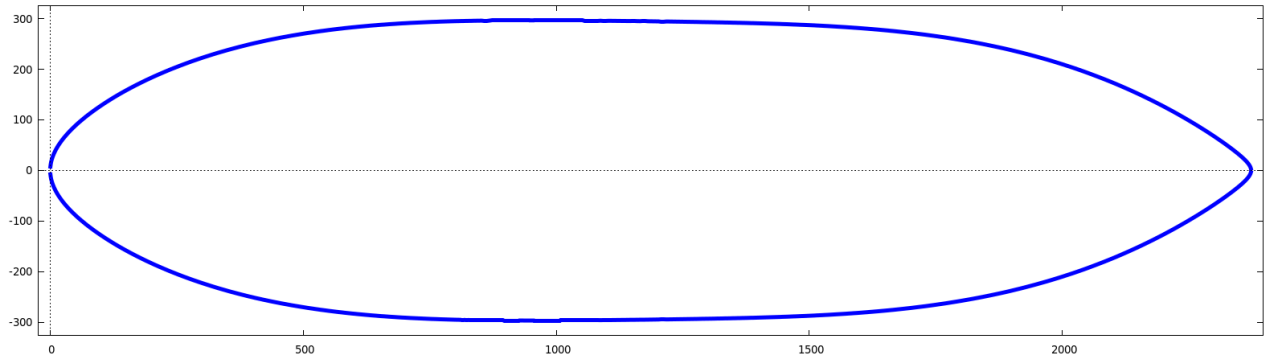


Illustration 2: Querschnitt des Zeppelins

4.3 Berechnung

Gores

Für den Bau der Hülle soll die Oberfläche des Rotationskörper in *n* gleiche Stücke zerlegt werden. Diese Stücke nennen wir Gores. Um die Form dieser Gores zu berechnen, gehen wir vom Querschnitts des Zeppelins aus.

Illustration 3: Gore

Stellen wir uns den Zeppelin als eine Summe unendlich vieler, unendlich dünner Kreisscheiben vor, dann liefert uns die Funktion für den Querschnitt den Radius einer dieser Kreisscheiben an einem bestimmten x. Multiplizieren wir diesen Radius mit 2π, erhalten wir den Kreisumfang an diesem x. Nun müssen wir jedem dieser Kreisumfänge eine Länge des Umfangs des Graphen vom Origo aus zuordnen. Würden wir eine Funktion l(x), die einem x die Länge des Umfangs des Graphen vom Origo aus bis zu diesem x liefert, finden, könnten wir die halbe Form der Gores mit dem Graphen folgender Funktion beschreiben. Durch Spiegeln an der x-Achse erhielten wir die ganze Form.

$$g(x)=\frac{f(l^{-1}(x))\cdot 2\cdot\pi}{2\cdot n}=\frac{f(l^{-1}(x))\cdot\pi}{n}$$

Die Umkehrfunktion von l(x) ordnet jeder Länge des Umfangs ein x zu. n ist die Anzahl Gores. Wir teilen durch n, weil der Umfang der Kreisscheibe an einem x sich aus n Gores zusammensetzt. Die Division durch 2 kommt daher, dass wir nur den Graphen eines halben Gores berechnen.

Numerische Methode

Nun haben wir aber ein Problem es lässt sich keine Funktion l(x) finden, welche für ein x die Länge des Graphen bis zum x liefert. Mit einer numerischen Methode lässt sich die Form der Gores trotzdem finden. Da die Gores zur x-Achse symetrisch sind, reicht es wenn wir nur eine Hälfte berechnen. Wir wählen ein möglichst kleines Δh und beginnen bei x gleich 0. Wir berechnen die

Differenz Δy zwischen f(x) und f(x + Δh). Mit dem Pythagoras von Δy und Δh erhalten wir eine Näherung der Graphenlänge zwischen x und x + Δh. Nun erhöhen wir x um Δh, berechnen erneut den Pythgoras von Δy und Δh und addieren ihn zur Gesamtlänge. Dies führen wir solange durch bis x gleich gross wie die Länge des Querschnitts ist. Gleichzeitig notieren wir uns den inneren Kreisumfang des Zeppelins für unsere x. Auf diese Weise erhalten wir Paare von Längen des Umfangs des Querschnitts und des inneren Kreisumfangs. Diese Paare können wir plotten und erhalten dadurch die Form unserer Gores.
Diese Rechnungen führen natürlich nicht wir durch, sondern ein Comptuerprogramm, das wir schreiben. Umso kleiner wir unsere Δh wählen, desto genauer wir unser Resultat. Dafür benötigt der Computer länger.

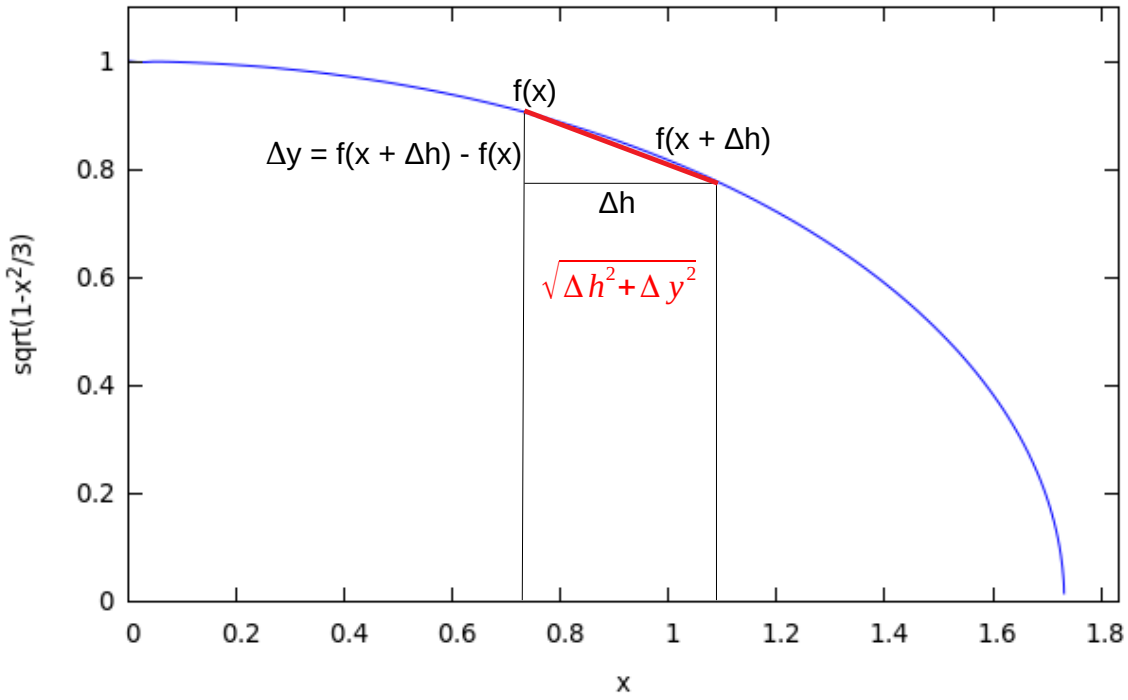


Illustration 4: Berechnung des Pythagorases für ein Graphenstück

Oberfläche und Volumen
Um das Gewicht der Gores zu berechnen, müssen wir deren Oberfläche kennen. Ähnlich wie für die Funktion l(x) für die Berechnung der Gores, lösen wir dies mit einer nummerischen Methode. Wir unterteilen unsere Gores in sehr kleine Rechtecke, dessen Flächen wir ausrechnen und aufsummieren. Oberfläche ohne Rand und Volumen lassen sich auch mit einem Integral berechnen.

Programm
Das Python Programm zur Berechnung der Form und der Oberfläche der Gores berechnet noch den Mittelpunkt der Oberfläche, was dem Schwerpunkt der Hülle entspricht und den Mittelpunk des Volumen, was dem Auftriebsmittelpunkt entspricht. Länge, Druchmesser und Breite des Randes müssen im Programm eingetragen werden. Die Ausgabe kann in eine .csv Datei umgelenkt und dann mit LibreOffice Calc oder Excel eingelesen werden.

4.4 Material und Bau

Das Material der Hülle wurde uns vom Silentrunner Team empfohlen und verkauft. Sie hat den Vorteil, dass nur eine Seite verschweissbar ist. Dies vereinfacht den Bau enorm. Wir können 3 Folien übereinander legen, wobei die Mittlere gefaltet ist, und in einem Zug verschweissen. Dazu

müssen wir nur mit dem Bügeleisen die Form einer Gore abfahren. Um das zu vereinfachen, druckten wir den Graphen einer Gore aus.
Nachdem man dann die übrige Folie getrimmt hat, ist der Blimp fertig. Theoretisch. Bei uns mussten wir bei beiden Hüllen die Form mehrfach abfahren und dann trotzdem noch mit Seifenwasser nach Löcher suchen. Vielleicht war die Temperatur des Bügeleisen falsch oder die Unebenheiten in der Unterlage führten zu kleinen Löcher.

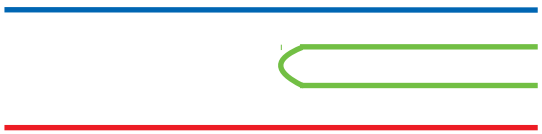


Illustration 5: Schichtung der Gores

Den Verschluss gestalteten wir als langer Schlauch, welchen man falten und mit einer Lebensmittelklammer abklemmen konnte. Schlussendlich entschieden wir uns den Verschluss immer wieder zu verschweissen, da es so um einiges dichter ist.

Illustration 6: Ungetrimmter Blimp

Nach dem ersten Test gestalteten wir die zweite Hülle kleiner, da wir zu viel Traggas in der ersten hatten. Doch da wir sonstige Verbesserungen anbrachten und ein bisschen überkorrigierten, verblieben wir mit einer zu kleinen Hülle für einen zu schweren Blimp. Dadurch mussten wir Distanzsensor und GPS ausbauen.

5 Steuerung und Antrieb

Anforderungen

Die Auswahl des Steuerungs- und Antriebsmechanismus hängt von Gewicht, Geschwindigkeit, Wendigkeit und Machbarkeit ab. Die Geschwindigkeit ist vor allem für das Fliegen im Freien wichtig. Hohe Geschwindigkeiten erlauben bei stärkerem Wind zu fliegen. Bei Baden beträgt die durchschnittliche Windgeschwindigkeit 4 m/s. Damit die Sensordaten von Nutzen sind, wird eine gewisse Wendigkeit benötigt. Da die Distanzsensoren Distanzen bis maximal 3 Meter messen können, soll in dieser Distanz gedreht bzw. angehalten werden können. Die wichtigsten Faktoren sind das Gewicht und die Machbarkeit. Die Komponenten lassen sich nicht ohne Weiteres an der Hülle befestigen.

Damit der Zeppelin während dem Flug nicht kippt, muss beim Geradeausfliegen die Summe der angreifenden Drehmomente null sein. Die Summe der Kräfte ist ebenfalls null, wenn der Zeppelin nicht beschleunigt oder bremst. Ausserdem muss der Zeppelin so konzipiert sein, dass er selbststabilisierend ist. Bei Windstösse soll der Zeppelin von alleine in die ursprüngliche Lage zurückkehren.

Steuerungen lassen sich in zwei Arten einteilen: Vektorsteuerung und Rudersteuerung. Die Steuerung kann auch aus einem Gemisch von beiden realisiert werden. Bei der Vektorsteuerung wird die Manövrierbarkeit des Zeppelin durch das Ausrichten der Motoren erreicht. Bei der Rudersteuerung wird der Motor unbeweglich befestigt. Durch Ruder kann der Zeppelin aber in die gewünschte Richtung geneigt werden.

Aufbau

Die Steuerung und der Antrieb unseres Luftschiffes wird durch drei Ruder, die ihn einem 120° Winkel von einander am Schwanz des Zeppelins befestigt sind, und zwei Motoren, die an einer Achse an der Gondel angebracht sind, realisiert. Die Ruder wirken erst ab einer gewissen

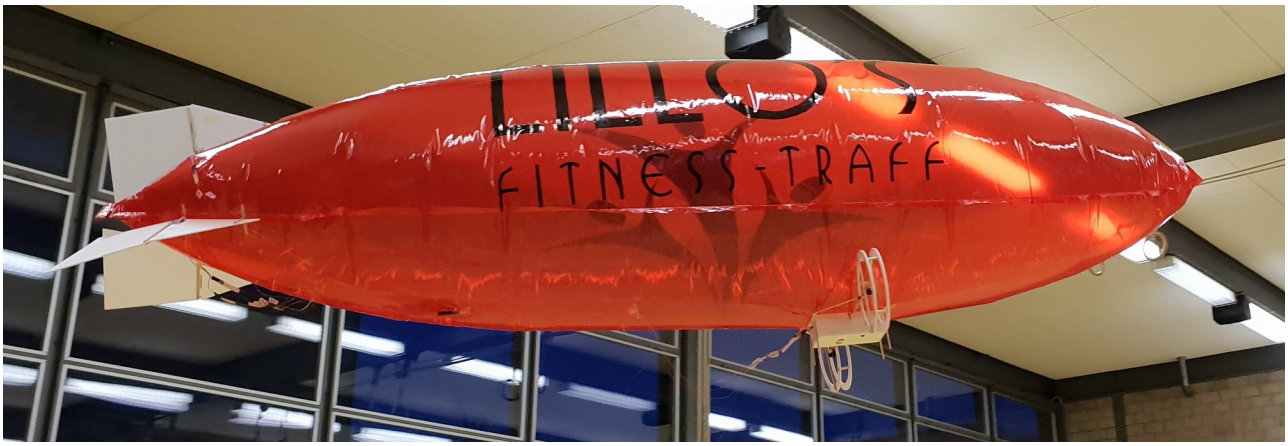


Illustration 7: Zeppelin im Flug

Geschwindigkeit. Bei tiefen Geschwindigkeiten kann der Zeppelin durch unterschiedliches ansteuern der Motoren trotzdem gieren.

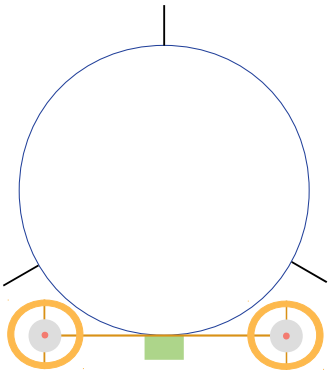


Illustration 8: Skizze des Zeppelins von Vorne

Illustration 9: Skizze Zeppelin von Seite

Steuerung

Um dem Benutzer das Steuern zu erleichten, reicht es dem Zeppelin den Winkel, den Radius und die Geschwindikeit zu geben in der bzw. mit der er fliegen soll. Daraus berechnet der Zeppelin dann wie er die Motoren und die Ruder ansteuern soll.

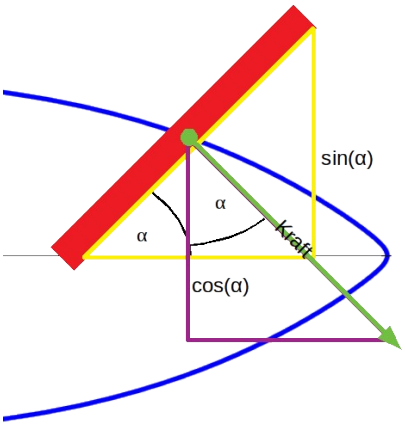


Illustration 10: Abhängigkeit Drehmoment von Auslenkung Alpha

Die Kraft am Ruder lässt sich nicht genau berechnen. Sie muss experimentell bestimmt werden. Jedoch wissen wir, dass sie proportional zu den folgenden Faktoren ist:

- Dichte der Luft
- Fläche der Ruder
- Quadrat der Geschwindigkeiten (einmal v für die Masse m die im Zeitraum t auf das Ruder prallt und ein zweites mal v für den Impuls der Masse m)
- sin(α)*cos(α) ~ sin(2 * α), wobei α der Winkel der Ruder ist

Die maximale Kraft erreichen wir wenn α 45° beträgt. Die Ruder erlauben dem Zeppelin zu gieren, nicken und teilweise zu rollen. Das Rollen ist unerwünscht. Dem Rollen wird einerseits durch die Gewichtskraft der Elektronik, die unten am Zeppelin befestigt ist, und anderseits durch das Ansteuern der Ruder entgegengewirkt.

Um beim Steuern das Rollen (Wanken) zu verhindern, muss die Summe der Drehmomente um die Längsachse null sein. Da die Ruder alle auf dem selben Radius montiert sind, bedeutet dies, dass die Summe der Kräfte an den Ruder ebenfalls null sein muss. Die Steuerrichtung wird dem Zeppelin als Winkel und Radius übergeben.
Das Verhältnis der benötigten Kräfte an den drei Rudern können wir mit drei um 120° verschobenen Sinuswellen beschreiben.

- Oberes Ruder: sin(x)
- Rechtes Ruder: sin(x + 2 * π / 3)
- Linkes Ruder: sin(x + 4 * π / 3)

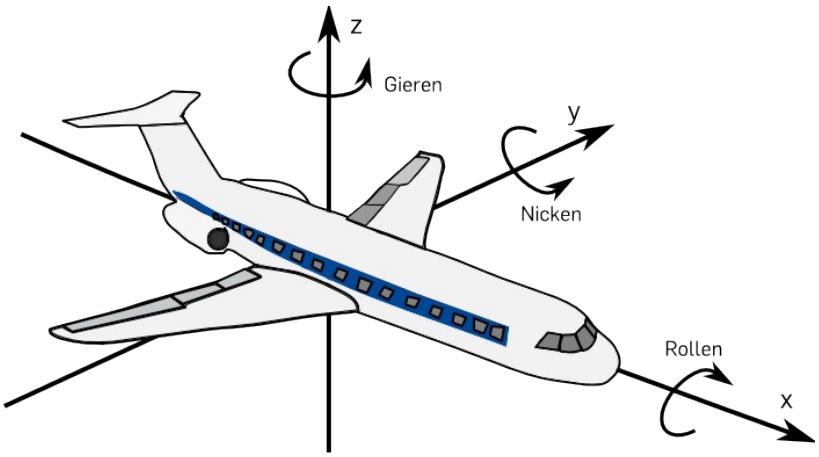


Illustration 11: Rotationen eines Fliegendenkörpers, Quelle: https://moodle.ruhr-uni-bochum.de

Die Summe der Sinuswellen ist 0, somit wirkt auch kein Drehmoment um die Längsachse, welches ein unerwünschtes Rollen (Wanken) hervorrufen würde. Das Drehmoment um die Quer- und Hochachse des Zeppelins ist ungleich null. Abhängig vom gewünschten Radius werden die Kräfte um einen Faktor skaliert.

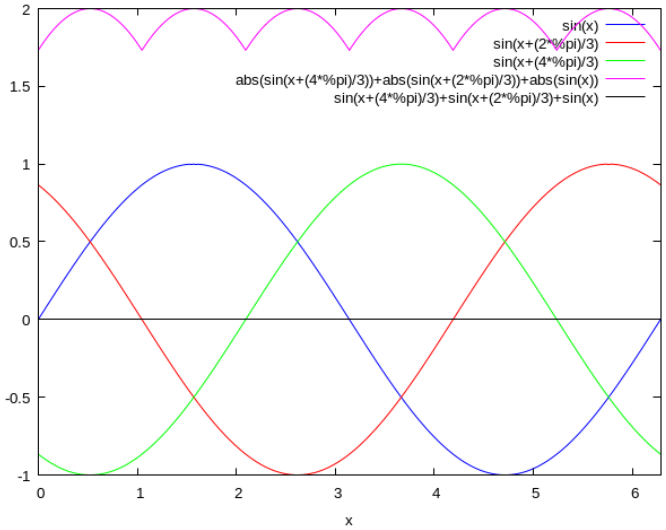


Illustration 12: 3 um 120° verschobene Sinuswellen und die Summe deren absolut Werte

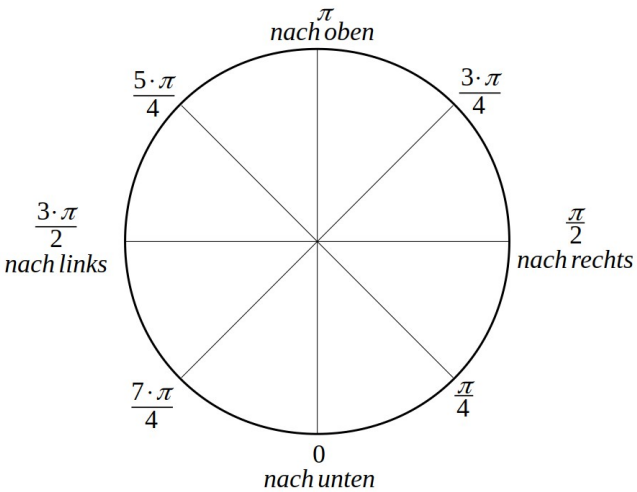


Illustration 13: Winkel zur Steuerrichtung

In der folgenden Tabelle werden die drei Steuerungsparameter zusammengefasst:

Steuerungsparameter	Wertebereich	Bemerkung
Winkel	0 – 2π / 0 – 360	0 = nach unten, 90 = nach rechts, 180 = nach oben,

		270 = nach links
Radius	0 – 100	0 = geradeaus
Geschwindigkeit	-100 – 100	< 0 = vorwärts, > 0 = rückwärts, 0 = stehen bleiben

Abschätzung Geschwindigkeit

Beim Fliegen wirkt der Luftwiderstand gegen die Flugrichtung. Die Maximale Geschwindigkeit ist erreicht, wenn der Luftwiderstand gleich dem Schub der Motoren ist.

$$F_{Motor} = F_{Widerstand} = \frac{A \cdot \rho \cdot c_w \cdot v^2}{2}$$

Laut Datenblatt erzeugt jeder unserer 2 Motoren einen Schub von 70g, was rund 0.7N entspricht. Der Durchmesser der Querschnittsfläche beträgt 60 cm und als c_w Wert schätzen wir 0.2. Dieser liegt zwischen dem einer Kugel 0.45 und dem eines Flugzeugs 0.08. Dies resultiert in eine maximale Geschwindigkeit von rund 6 m/s.

Austarierung

Die Komponente haben einen bestimmten Platzt auf der Hülle. Zum Beispiel wird die Gondel an der Stelle befestigt, an der die Steigung der Hülle 0 ist. Um das Drehmoment, das durch den Auftrieb und die Gewichtskraft der Komponente entsteht, auszugleichen, befindet sich die Batterie nicht im Kasten. Sie kann an einem beliebigen Platzt angebracht werden damit sich die Summe der Drehmomente zu 0 aufaddieren.

Bezeichner	Kraft [N]	Distanz von der Nase [cm]	Drehmoment [Nm]
Gewicht Batterie	0.71	95	67.45
Gewicht Kasten	1.55	95	147.25
Auftrieb	-4.34	113	-490.42
Gewicht Hülle	1.71	114	194.94
Gewicht Ruder	0.32	207	66.24
Gewicht Ventil	0.06	237	14.22
Summe	0		0

6 Komponente und Elektronik

Neben den zwei Motoren und den drei Servos für die Steuerung, ist der Zeppelin mit einer Palette an Sensoren ausgestattet. Der Kern der Elektronik ist ein Raspberry Pi Zero W, dieser hat im Vergleich zu den anderen Modellen ein geringes Gewicht sowie ein geringeren Stromverbrauch. Er ist aber auch Leistungsschwächer. Vorteil gegenüber eines Arduinos ist eine höhere Rechenleistung, integrierte Grafik- und Netzwerkchips und ein vollwertiges Linux als Betriebssystem.

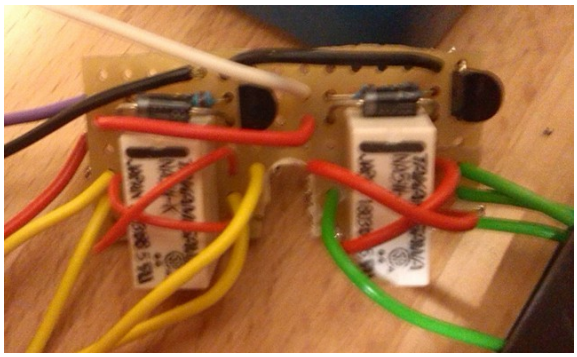


Illustration 14: Schaltung zum Vertauschen zweier Phasen des Drehstroms

ESC / Motor / Servo

Mit zwei Pins des Raspberry die ein PWM-Signal[1] an die ESCs leiten, lässt sich die Geschwindigkeit der Motoren variieren. Ein ESC (Electronic-speed-controller) reguliert abhängig der Dauer des PWM-Signals die Frequenz der Drei-Phasenanschlüssen. Mithilfe von Relais lässt sich der Drehstrom invertieren, dadurch läuft der Motor rückwärts. Die drei Servos der Ruder werden ebenfalls über PWM gesteuert.

Kameras: HBV-1517 S1.0 / Raspberry Pi camera 1.3

Zwei Kameras, eine über USB die andere an einem CSI-Anschluss, liefern uns sowohl Sicht nach Vorne und eine Vogelperspektive vom Zeppelin aus.

Distanzsensor: HC-SR04 (aus Gewichtsgründen entfernt)

Zwei Distanzsensoren messen mittels Ultraschall die Distanz nach vorne und nach unten zum nächsten Hindernis. Die maximale Reichweite der Sensoren ist 3 m. Die Sensoren sind für 5V konzipiert. Da der Raspberry Pi mit 3.3 V arbeitet, mussten wir die „Echo“ Pins mit einem Spannungsteiler versehen.

GPS: NEO6M

Das GPS dient der Ortung, so kann der Zeppelin bei Flügen im Freien seine aktuelle Position auslesen. Dies würde selbständige Flüge nach einer vorprogrammierten Route oder auch eine „Return Home“ Funktion ermöglichen, welche den Zeppelin immer zurück zum Start steuert bei Signalverlust. Kommunikation mit dem Raspberry erfolgt über das UART Protokoll.

Barometer: BMP280

Das Barometer misst den aktuellen Luftdruck. Da in höheren Altitude der Druck abnimmt kann man so die Höhe des Zeppelins errechnen.

Kompass: QMC5883L

Dieser Chip kann Magnetfelder messen und dient als elektronischer Kompass. Mithilfe von ihm lässt sich die Aktuelle Orientierung des Zeppelins berechnen.

Giro/Accelerometer: MPU6050

Dieser Sensor kann sowohl die aktuelle Drehbewegung sowie die Beschleunigung in jede Richtung messen. Er erlaubt die Daten des Kompasses und des GPS zu verbessern.

I2C Bus

I2C ist ein Protokoll, das erlaubt mit mehreren Chips anhand von nur 4 Kabeln (GND, 5V, SDA, SCL) zu kommunizieren. Der Barometer, der Kompass, der Beschleunigungssensor und der Analogdigitalwandler hängen alle am I2C-Bus. Das Problem war, dass dessen Platinen alle bereits einen Pullup-Widerstand verbaut hatten. Hätten wir diese einfach parallel an den Bus angeschlossen, was die Idee hinter I2C ist, wäre das Resultat ein viel zu kleiner Pullup-Widerstand gewesen, der zur Beschädigung des

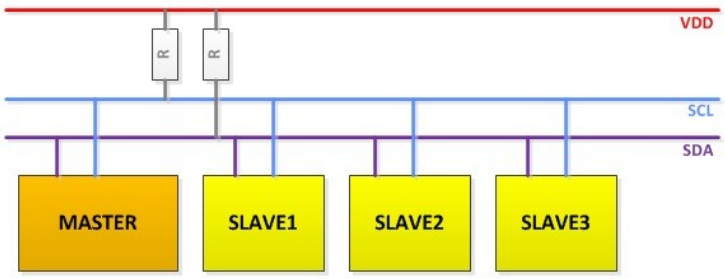


Illustration 15: Schema Funktionsweise I2C, R sind die Pullup-Widerstände, Quelle: http://www.lucadentella.it

1 Die Servos und Motoren erhalten kein richtiges PWM Signal, sondern alle 20 ms einen Puls mit einer Dauer zwischen 1 und 2 ms. Die Dauer des Pulses entspricht einem Winkel oder einer Drehzahl.

Raspberry Pis oder der Sensoren geführt hätte. Aus diesem Grund mussten wir auf jeder Platine die Pullup Widerstände der SCL und SDA Linie suchen und ablöten.

Batterie:

Als Stromversorgung benutzen wir Lithium-Polymer-Batterien, da diese sehr Energiedicht sind. Zwei Zellen decken die erwünschte Spannung von 7.6 V ab. Da LiPo-Batterien sehr empfindlich sind, müssen sie mit einer Schaltung, die unter der Mindestspannung von 3.3V oder über der Maximalspannung von 4.2V den Entladung respektive Ladungsvorgang abbrechen, ausgestattet sein. Die Batterie von 1300mAh lässt uns bei einem verbrauch von 3A 26 Minuten fliegen. Der Verbrauch variiert jedoch mit der Fluggeschwindigkeit. Um zu wissen wann die Batterieladung sich dem Ende neigt, kann man ihre Spannung messen. Da sie sich im Bereich von 6.6-8.4V aufhält, können wir mit einem ADC (analog to digital converter) die Spannung und somit die verbleibende Ladung messen. Um den Raspberry Pi und die Servos zu versorgen, bauten wir einen Spannungswandler ein der 5V und max. 3A liefern kann.

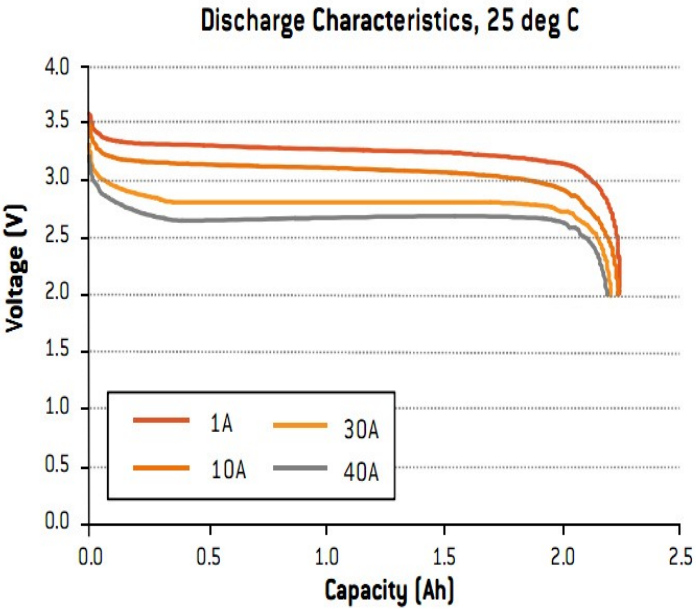


Illustration 16: Entladekurve einer Lipo Batterie, Quelle: https://www.pedelecforum.de

Propeller

Die Propeller erzeugen aus der Rotation der Motoren eine Kraft. Die Kraft ist die Änderung des Impulses der durch die Propeller beschleunigten Luft über der Zeit. Laut dem Impulserhaltungssatz muss der Zeppelin den Impuls erhalten, den die Motoren der Luft in die gegenseitige Richtung verleihen. Da die kinetische Energie mit dem Quadrat der Geschwindigkeit zunimmt, ist es interessanter eine grosse Masse an Luft auf eine kleine Geschwindigkeit als eine kleine Masse an Luft auf eine grosse Geschwindigkeit zu bringen, um denselben Impuls zu erhalten. Dementsprechend sollten die Propeller für einen hohen Wirkungsgrad eine grosse Oberfläche also viele Rotorblätter und einen grossen Radius besitzen. Die Propellersteigung und die Drehzahl des Motors sollte tief sein. Eine hohe Rotorblätterzahl bedeutet, dass das selbe Luftstück öfters in der gleichen Zeit durchschnitten wird. Die Rotorblätter erzeugen Wirbel, die den Wirkungsgrad des folgenden Blattes reduzieren. Als Kompromiss zwischen der hohen Oberfläche durch eine grosse Anzahl an Rotorblättern und dem Störfaktor der verwirbelten Luft werden meistens zwei blättrige Rotoren verwendet. Die Probleme des hohen Radius sind einerseits der benötigte Platz und anderseits die hohe Geschwindigkeit der Spitzen der Rotorblätter, welche in Lärm resultiert. Im Modellbau wird meistens von den Herstellern des Motors ein Propellertyp empfohlen. Generell lohnt es sich diesem zu folgen.

Ruder und Kästchen

Wir bauten ein Kästchen um die Elektronik des Zeppelins zu verstauen. Sowohl bei den Rudern als auch beim Kästchen bauten wir die grossen Flächen aus Depron, ein Polystyrol. Die Achsen zum

Befestigen der Ruder und des Kästchen an der Hülle, der Motoren am Kästchen und zum Übertragen der Kraft der Servos auf die Ruder fertigten wir aus Balsaholz an. Zum Verkleben der Komponenten verwendeten wir UHU Por und Epoxidharz. Zum Schutz der Propeller sind die Motoren mit einem Doppelring aus Depron umgeben.

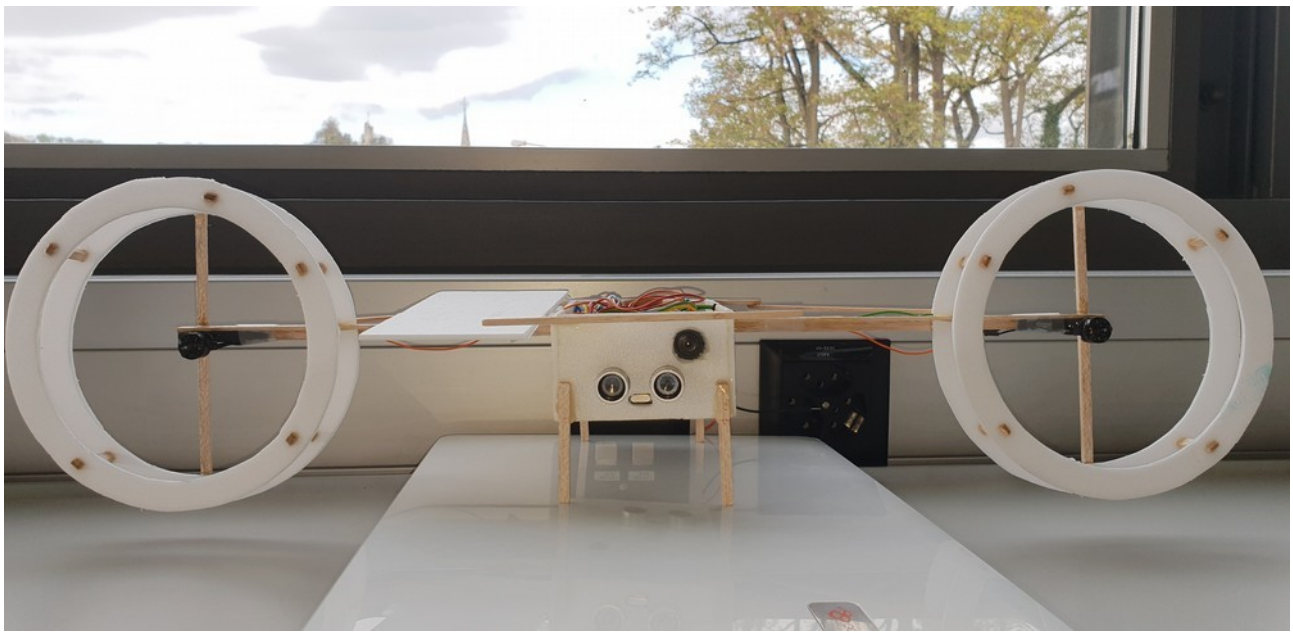


Illustration 17: Kästchen von vorne, an der Unterseite ist eine zweite Kamera und ein zweiter Distanzsensor befestigt.

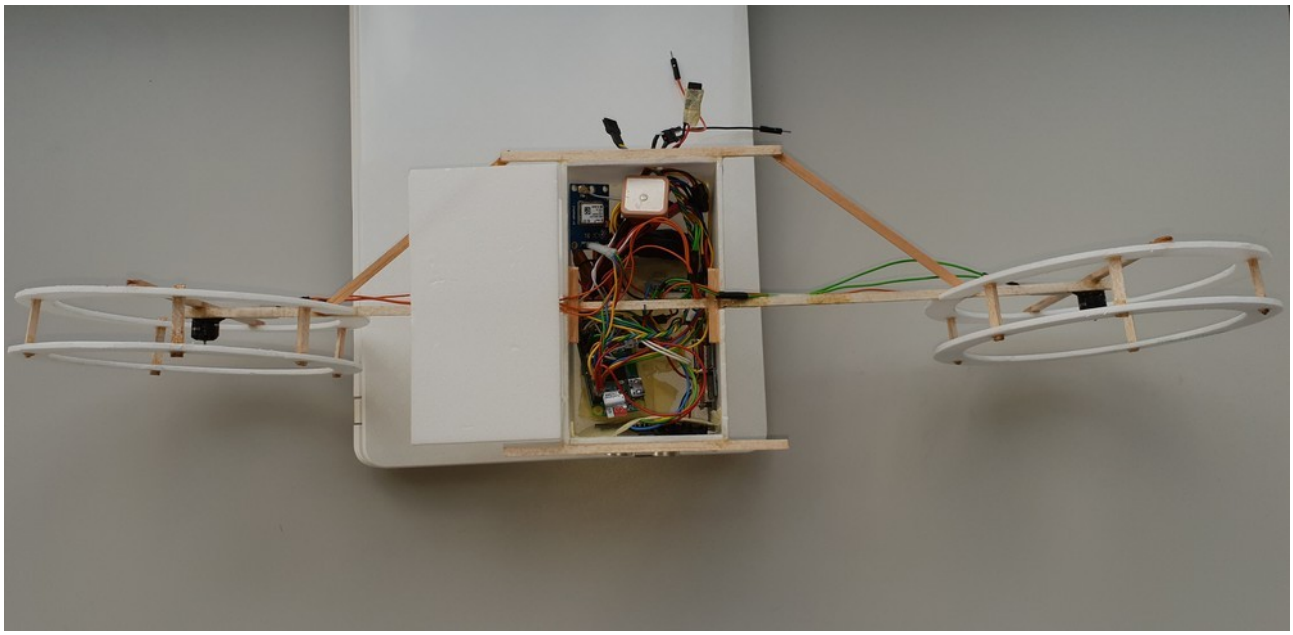


Illustration 18: Kästchen von vorne, hinten sind die Anschlüsse für den Strom und die Servos ersichtlich

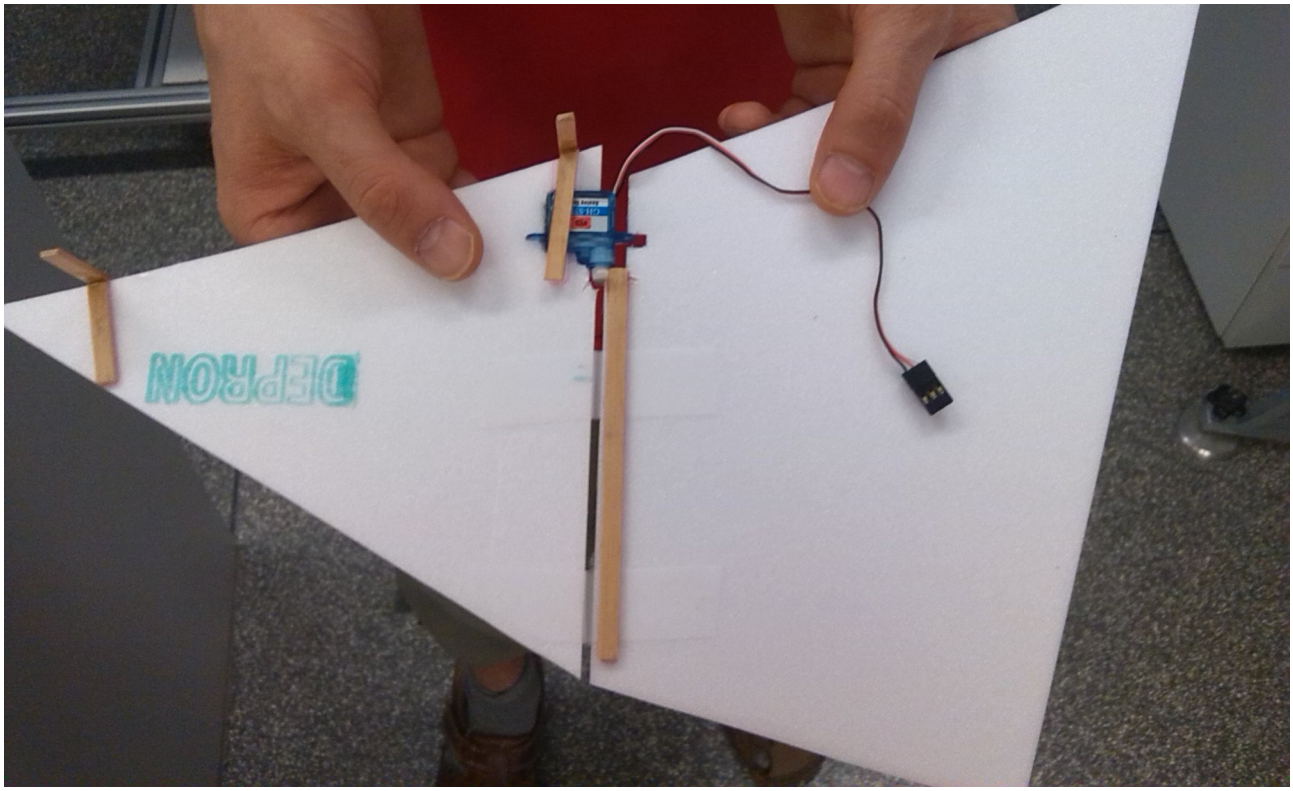


Illustration 19: Ruder mit Servo, Scharnier ist aus Klebeband

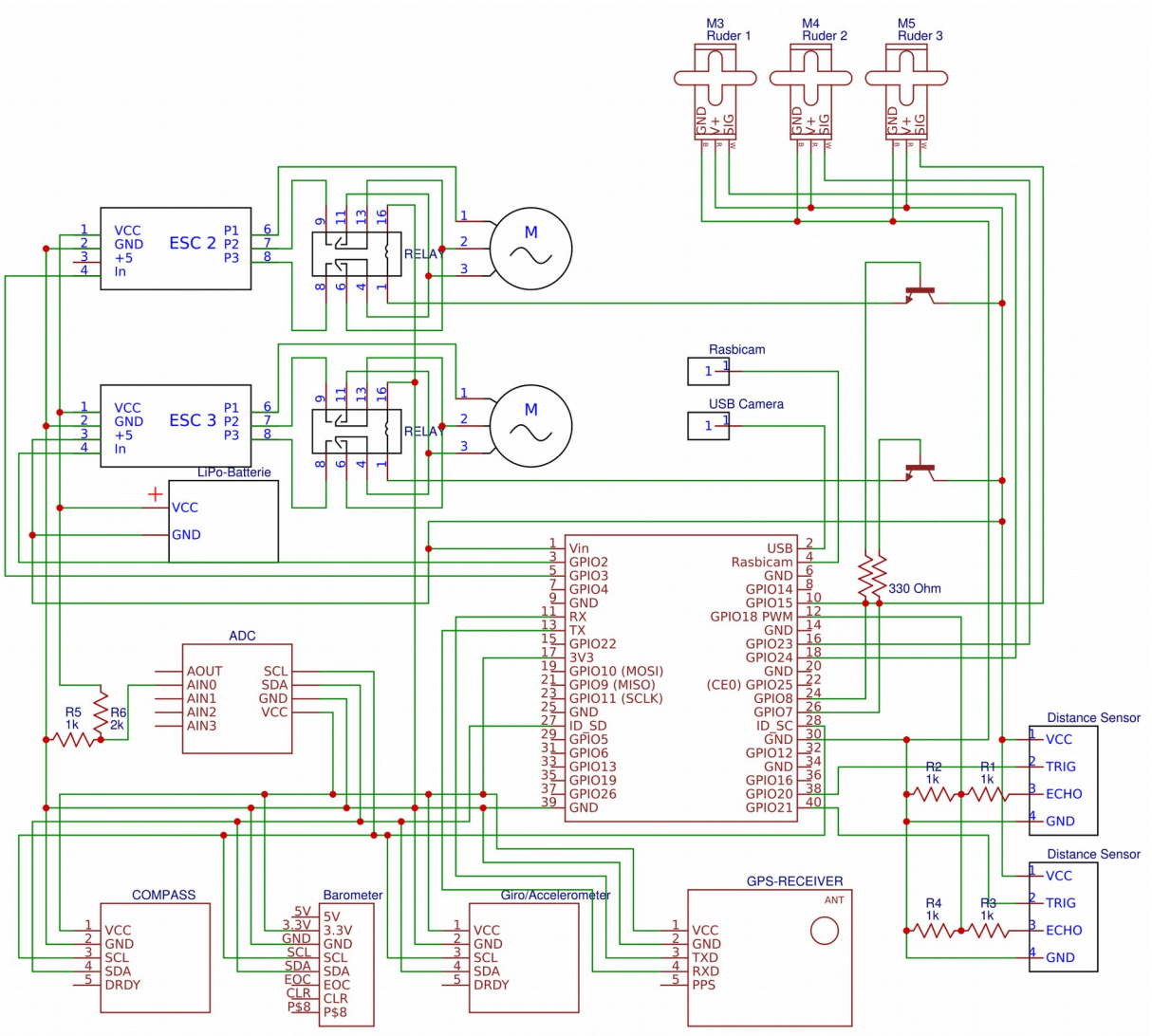
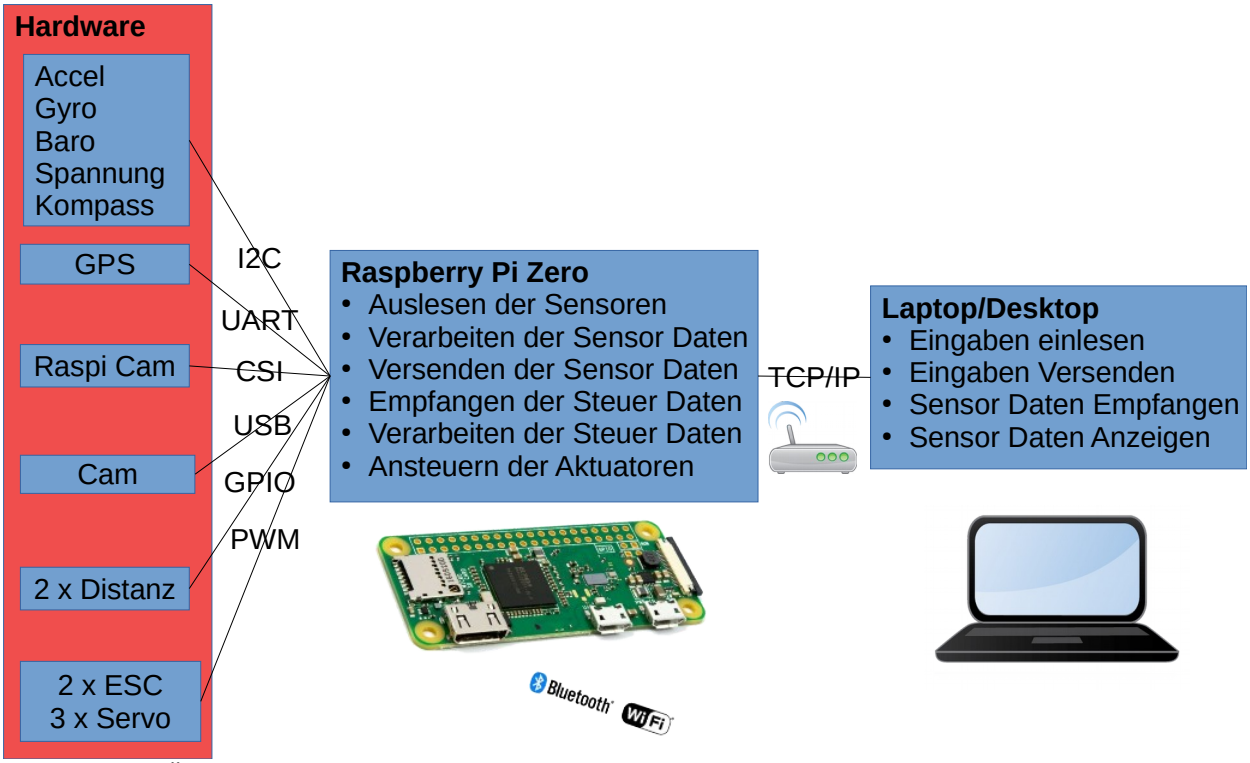


Illustration 20: Schaltbild, Widerstandswerte stimmen nicht

7 Software

Die Software für den Zeppelin besteht aus zwei Programmen, einerseits aus der grafischen Benutzeroberfläche, die auf dem Computer des Benutzers läuft, und anderseits aus dem Programm auf dem Raspberry Pi. Die zwei Programme kommunizieren über WLAN miteinander.



Drawing 1: Überblick Softwareteile

7.1 Kommunikation

Die Kommunikation zwischen dem Raspberry Pi und dem Computer des Benutzers baut auf TCP/IP Sockets auf. Sockets stellen die zwei Endpunkte einer bidirektionalen Verbindung dar. Sie erlauben dem Programm einen Strom von Bytes zu senden und zu empfangen. Im Gegensatz zu UDP, ein paketorientiertes Protokoll, wird garantiert, dass die Daten überhaupt und in der richtigen Reihenfolge ankommen. Vorteil von UDP im Vergleich zu TCP ist eine höhere Geschwindigkeit.

Wir mussten ein Protokoll, auf TCP/IP aufbauend, definieren wie unsere Daten versendet werden. Unsere Daten werden paketweise und wie bei Netzwerkprotokollen üblich Big-Endian[2] verschickt. Die Pakete sehen folgendermassen aus:

2 bytes	1 byte	1 byte	2 bytes	n bytes
Sync ($ABCD_{16}$)	Device	Parameter	Length	Playload
	ID			

2 Die Endianess beschreibt die Reihenfolge der Bytes. Bei Big-Endian kommt das höchstwertige Byte zuerst. Die meisten Computer arbeiten Little-Endian, wo das tiefstwertige Byte zuerst kommt.

Synchronisation und Daten

Die ersten zwei Bytes dienen der Synchronisation. Wurde vorher ein fehlerbehaftetes Paket versandt, kann der Datenstrom wieder in einen gültigen Zustand gebracht werden. Die Anzahl an Bytes für die Synchronisation wurde nicht zufällig gewählt. Umso mehr Bytes dafür verwendet werden, desto langsamer wird die Kommunikation. Verwendet man dafür aber nur ein Byte, ist die Wahrscheinlichkeit, dass sich im Strom ein anderes identisches Byte befindet sehr hoch: $1 / 2^8 = 1 / 255$. Verwenden wir zwei Bytes liegt die Wahrscheinlich deutlich tiefer: $1 / 2^{16} = 1 / 65‘536$. Die nächsten zwei Bytes, Device und Parameter, ergeben zusammen eine ID. Diese ID bestimmt, die Bedeutung des Inhalts des Pakets. Das „Length“ Feld enthält die Länge der folgenden „Playload“ in Anzahl Bytes. Die „Playload“ sind schlussendlich die Daten.

Videodaten

Um das Socket, durch welches wir die Konfiguration und die Sensormesswerte senden, nicht mit den Videodaten zu verstopfen, wird für beide Kameras jeweils ein eigenes Socket angelegt. Der Raspberry Pi hat dedizierte Hardware, um Videos mit dem h.264 Standard[3] zu komprimieren und zu dekomprimieren. Der Treiber für die Raspberry Pi Kamera kann den Datenstrom von der CSI Kamera direkt durch die Kompressionshardware leiten. Dieser Datenstrom wird dann in ein Socket weitergeleitet. Der Treiber der USB Kamera liefert Bilder im YUV[4] Format. Vor dem Versenden werden diese noch zu einem JPG komprimiert. Dies erlaubt die Datenmenge für die Bilder von der USB Kamera, um einen Faktor 15 zu reduzieren. Für die CSI Kamera mit der h.264 Kompression ist es sogar ein Faktor 40. Der Videostream der CSI Kamera wird mit 640 auf 480 Pixel aufgenommen. Höhere Auflösungen bis zu 2592x1944 Pixeln wären möglich. Die USB Kamera hingegen schafft nur bei 320 auf 240 Pixel ein flüssiges Bild zu liefern.
Unser Programm erreicht ein Stream von rund 30 Bildern pro Sekunde mit einer Verzögerung von wenigen Zehntelssekunden. Mit anderen Programmen wie „motion“ oder „uv4l“ erreichte ich nur einen Bruchteil der Bildrate. Mit „raspivid“ in Verbindung mit „vlc“ erreichte ich zwar ebenfalls hohe Bildraten, Problem war aber eine Verzögerung von ein paar Sekunden.

Abbruch der Verbindung

TCP/IP erlaubt es festzustellen, wenn der andere Rechner sein Socket ordnungsgemäss schliesst. Wird die Verbindung aber nicht ordnungsgemäss unterbrochen, erfahren die beteiligten Rechner nichts davon[5]. Aus diesem Grund und um einen Eindruck der Qualität der Verbindung zu erhalten, sendet der Computer des Benutzers jede Sekunde ein Paket mit einem Zeitstempel (Echorequest). Erhält der Raspberry Pi dieses Paket, schickt er es einfach zurück (Echoreply). Der Computer des Benutzers kann daraus die Zeit für diesen Rundgang berechnen. Erhält der Computer des Benutzer nach 5 Sekunden keine Antwort, wird die Verbindung als tot deklariert. Dasselbe gilt auf der Seite des Raspberry Pis. Ist der letzte Echorequest vor mehr als 5 Sekunden eingegangen, gilt die Verbindung als tot. Ist die Verbindung tot oder tritt sonst ein Fehler in der Verbindung auf, werden die Sockets beidseitig geschlossen. Danach wird versucht, die Verbindung erneut aufzubauen.

Sicherheit

Die Kommunikation zwischen Raspberry Pi und Computer verläuft völlig unverschlüsselt. Deshalb wäre es ein Leichtes mitzuhören oder das Programm zum Absturz zu bringen, indem Pakete manipuliert werden. Schlimmstenfalls könnte der Raspberry Pi gekapert werden. Dass das Programm wegen der „pigpio“ Bibliothek mit root Rechten läuft, stellt eine weitere Gefahr dar. Aus diesen Gründen wäre es sinnvoll für eine zukünftige Version die Daten zu verschlüsseln.

3 h.264 ist der heutige Industriestandard zur Videokompression
4 Auch YCbCr genannt
5 Keepalive ist ein optioneller Mechanismus von TCP/IP, der dies erlauben würde.

Wifi Hardware

Der Raspberry Pi Zero W hat ein integriertes Wifi Modul, das wir für die Kommunikation verwenden. Mit passender Software lässt sich der Raspberry Pi als „Accespoint“ konfigurieren. Der Zeppelin lässt sich dementsprechend entweder über ein bestehendes Netzwerk, wie zum Beispiel das WLAN der Kantonsschule oder über ein selber aufgespanntes Netzwerk steuern. Das vom Raspberry Pi aufgespannte Netzwerk hat eine Reichweite von rund 100 Metern. Für höhere Distanzen liesse sich eine gerichtete Antenne für die Bodenstation basteln. Laut Wikipedia (https://en.wikipedia.org/wiki/WokFi) liessen sich Distanzen über 3 km erreichen.

7.2 Grafische Benutzeroberfläche

Die grafische Benutzeroberfläche programmierten wir in Java. Der grosse Vorteil von Java ist die Unabhängigkeit vom Betriebssystem.

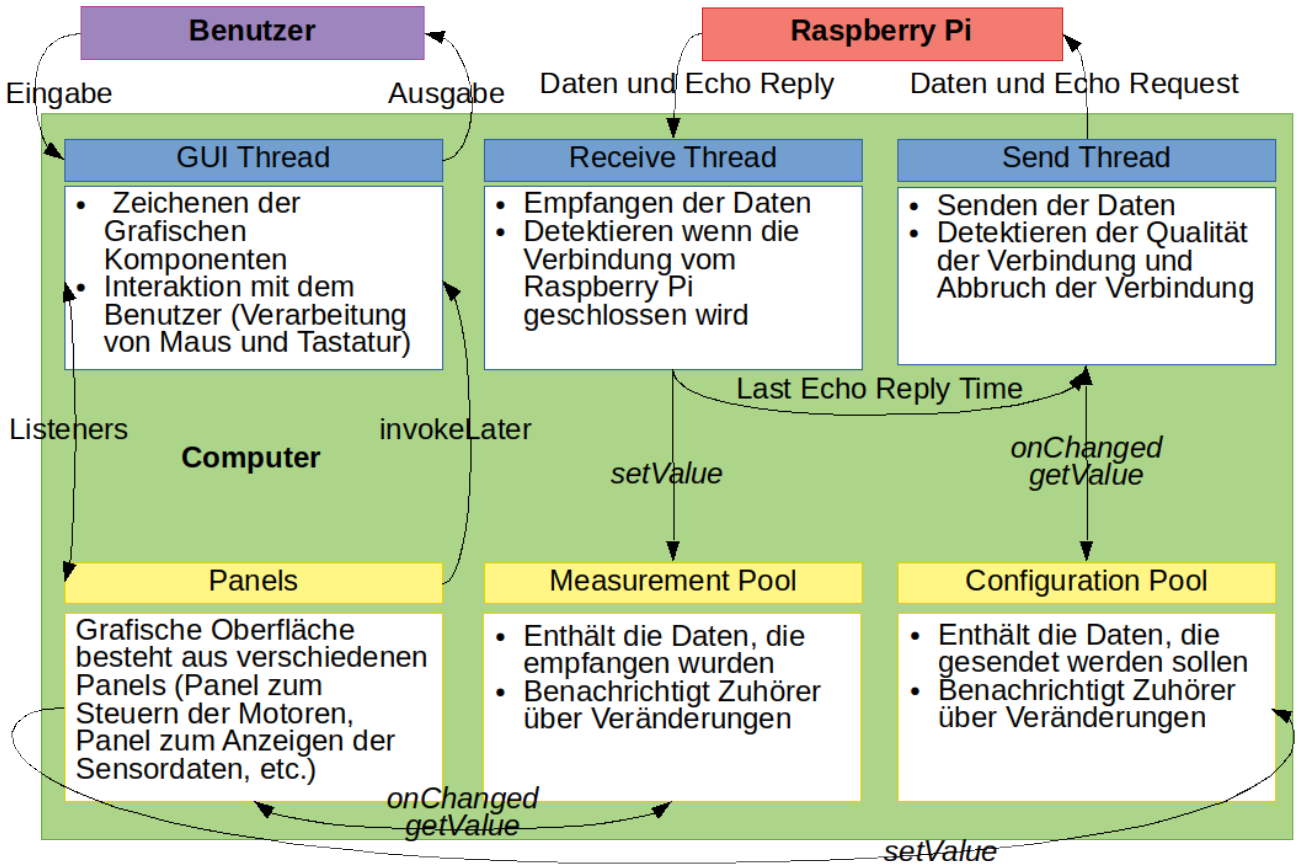
Threads

Das Programm besteht aus drei Hauptthreads[6]. Ein Thread ist ein Programmteil, das parallel also gleichzeitig zu den anderen Programmteilen ausgeführt wird. Das Programm liesse sich nicht einfach mit einem einzigen Thread realisieren, da das Empfangen der Daten ein blockierender Systemaufruf ist. Würden wir im GUI Thread die Daten empfangen wollen, würde dieser blockieren bis die gewünschten Daten angekommen sind. In dieser Zeit könnten wir keine Interaktion des Benutzers entgegennehmen, was zu einem ruckeligen Benutzerinterface führen würde. Der Nachteil von Programmen mit mehreren Threads ist, dass diese Schwieriger zum Debuggen sind, weil ihre Ausführung nicht immer genau gleich abläuft. Des weiteren müssen geteilte Ressourcen synchronisiert werden, um Race Conditions zu vermeiden.

Struktur

Ziel war ein möglichst modulares Programm zu erschaffen. Dies erlaubt eine einfache Erweiterung und eine einfache Portierung auf Android, dessen Apps in Java geschrieben werden. Das Programm lässt sich in einen Benutzerinterface Teil und einen Verbindungsteil aufspalten. Der Benutzerinterface Teil besteht aus verschiedenen Panels, die sich um die Interaktion mit dem Benutzer kümmern. Die Frame Klasse stellt das Fenster dar und verwaltet die Panels. Der Verbindungsteil setzt sich aus verschiedenen Klassen zur Verwaltung der Verbindung, der generischen Klasse Pool und des Interfaces ConnectionData und dessen Subklassen zusammen. Die Subklassen vom Interface ConnectionData repräsentieren die möglichen Playload der Pakete und implementieren wie diese Empfangen und Versendet werden sollen. Die Klasse Pool fungiert als Schnittstelle zwischen den zwei Teilen. Eine Instanz dient dem Sammeln der Empfangenen Daten und dem Benachrichtigen der Panels über Änderungen. Eine zweite Instanz sammelt die Daten von den Panels, welche Versendet werden sollen und Benachrichtig den SendThread über Änderungen.

6 Abgesehen vom Garbage Collector, den Threads, welche die Karte laden und den Threads zum Empfangen und Dekomprimieren der Videodaten



Grafische Oberfläche

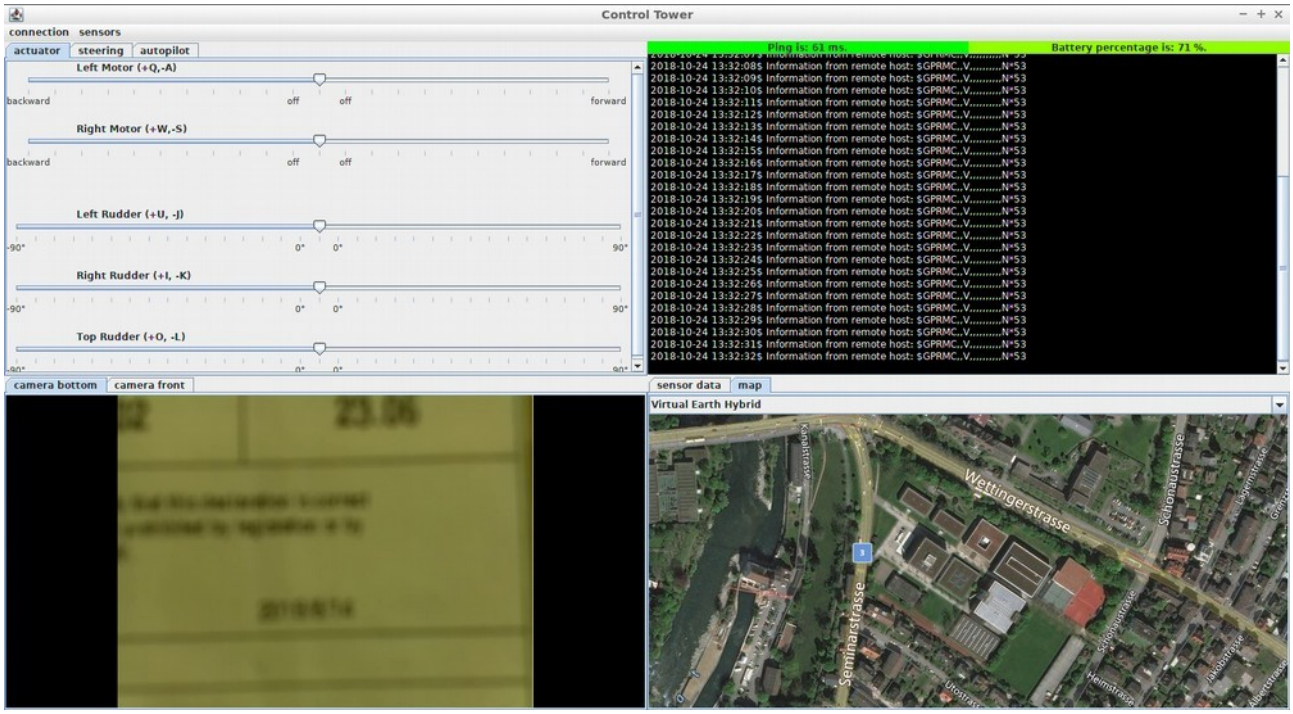


Illustration 21: Grafische Oberfläche

Das Fenster der grafischen Oberfläche wurde gevierteilt. Der linke, obere Teil enthält Tabs, die ein Zugriff auf verschiedene Bedienelemente für die Steuerung erlauben. Der rechte, obere, Teil zeigt die Zeit des letzten Echo Rundgangs, was ein Hinweis auf die Qualität der Verbindung ist, die verbleibende Batterieladung und Nachrichten in einem Textfeld an. Linksunten läuft jeweils in

einem Tab der Videostream der zwei Kameras. Schlussendlich werden rechts unten die gemessenen Sensorwerte tabellarisch dargestellt. Die vom GPS ermittelte Position lässt sich wahlweise auf einer Karte oder einem Luftbild darstellen. Die Menüleiste dient um sich mit dem Zeppelin zu verbinden und die Kameras ein und auszuschalten.

Steuerung

Die einzelnen Aktuatoren des Zeppelins lassen sich in der grafischen Oberfläche anhand Schieberegler steuern. Da dies umständlich und nicht intuitiv ist haben wir uns eine zweite Lenkungsmethode ausgedacht. Anhand eines Schieberegler lässt sich die Geschwindigkeit einstellen. Der Schieberegler definiert ebenfalls, ob das Luftschiff rückwärts oder vorwärts fliegt. Um die Flugrichtung anzugeben, befindet sich ein roter Punkt in einem kartesischen Koordinatensystem. Mit den Pfeiltasten kann der Benutzer den roten Punkt in x- und y-Richtung verschieben. Befindet sich der rote Punkt auf dem Origo fliegt das Luftschiff gerade aus. Desto weiter in positive y-Richtung der Punkt ist, umso steiler dreht das Luftschiff nach oben. Desto weiter in negitve y-Richtung der Punkt ist, umso steiler dreht das Luftschiff nach unten. Analog verhält es sich mit der x-Achse und dem Links- und Rechtsdrehen. Ist die vordere Kamera an, wird das Koordinatensystem mit dessen Bild hinterlegt.

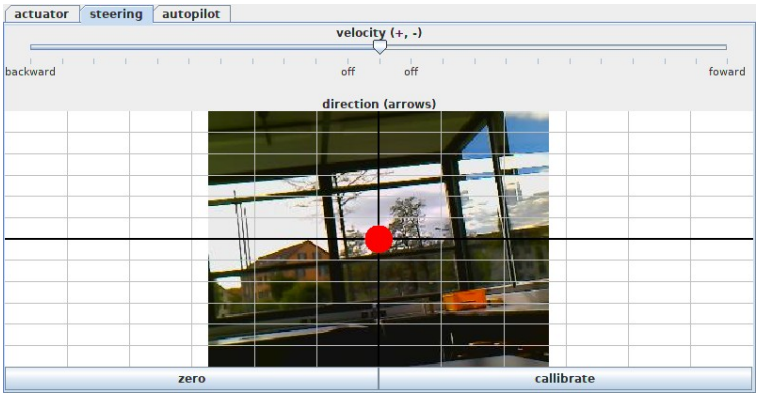


Illustration 22: intuitive Steuerungsmethode

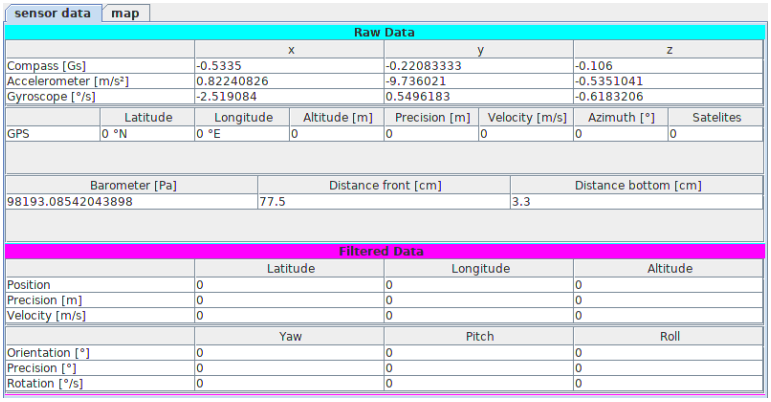


Illustration 23: Sensordaten Anzeige: Tabelle

Sensordaten

Die Sensordaten werden tabellarisch dargestellt. Die erste Tabelle zeigt die rohen Messdaten. Die zweite Tabelle sollte die verarbeiteten, gefilterten Messdaten anzeigen. Dies wurde aber noch nicht umgesetzt. Ausserdem wäre es sinnvoll die Daten grafisch anzuzeigen. Zum Beispiel mit einem künstlichen Horizont. Die vom GPS gelieferte Position kann bereits in einer Karte von Open Street Map oder einem Luftbild von Virtual Earth (Microsoft) angezeigt werden.

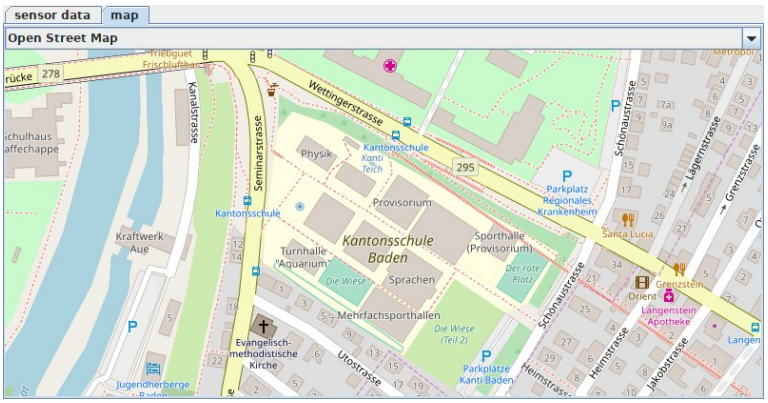


Illustration 24: Sensordaten Anzeige: Karte (Position würde mit einer Blase angezeigt werden)

Umfang

Der Quellcode wurde in einen grafischen Teil und einen Netzwerk, Datenverwaltung Teil aufgespalten. Möchte man das Programm als App für Android portieren, müsste nur der grafische Teil neu programmiert werden. Total umschliesst das Java-Programm über 3100 Quellcode Zeilen in 41 verschiedenen Dateien.

7.3 Raspberry Pi

Auf dem Raspberry Pi ist wegen der schwachen Hardware ein effizientes, systemnahes Programm unabdingbar. Deshalb schrieben wir dieses in C und C++. Ansonsten wird auf dem Raspberry Pi oft Python verwendet. Dadurch dass Python eine Interpretersprache ist, wäre es zu langsam für unseren Zweck gewesen.

Struktur

Das Programm auf dem Raspberry Pi besteht ebenfalls aus mehreren Threads. Auf dem Main Thread läuft eine Schleife, die eingehende Verbindung annimmt und diese dann Verwaltet. Die Schlaufe setzt die empfangenen Pakete in eine Warteschlange. Parallel dazu arbeitet der Control Thread die Warteschlange ab, liesst die Sensoren aus und versendet die Messdaten. Dadurch kann fast der gesamte Zugriff auf die Hardware von einem Thread aus erreicht werden, was den Aufwand für das Synchronisieren der Threads für einen Zugriff auf die Hardware deutlich reduziert. Nur der Zugriff auf den Neo6M (GPS) und auf die Kameras wurden in separate Threads ausgelagert. Der Grund ist, dass das Auslesen mit einem blockierendem Systemaufruf realisiert wird[7] und zu lange dauert. Für den Neo6M dauert das Auslesen ein Messzyklus, was bei den Standarteinstellungen einer Sekunde entspricht. Dadurch würden im Control Thread zu grosse Verzögerungen auftreten.

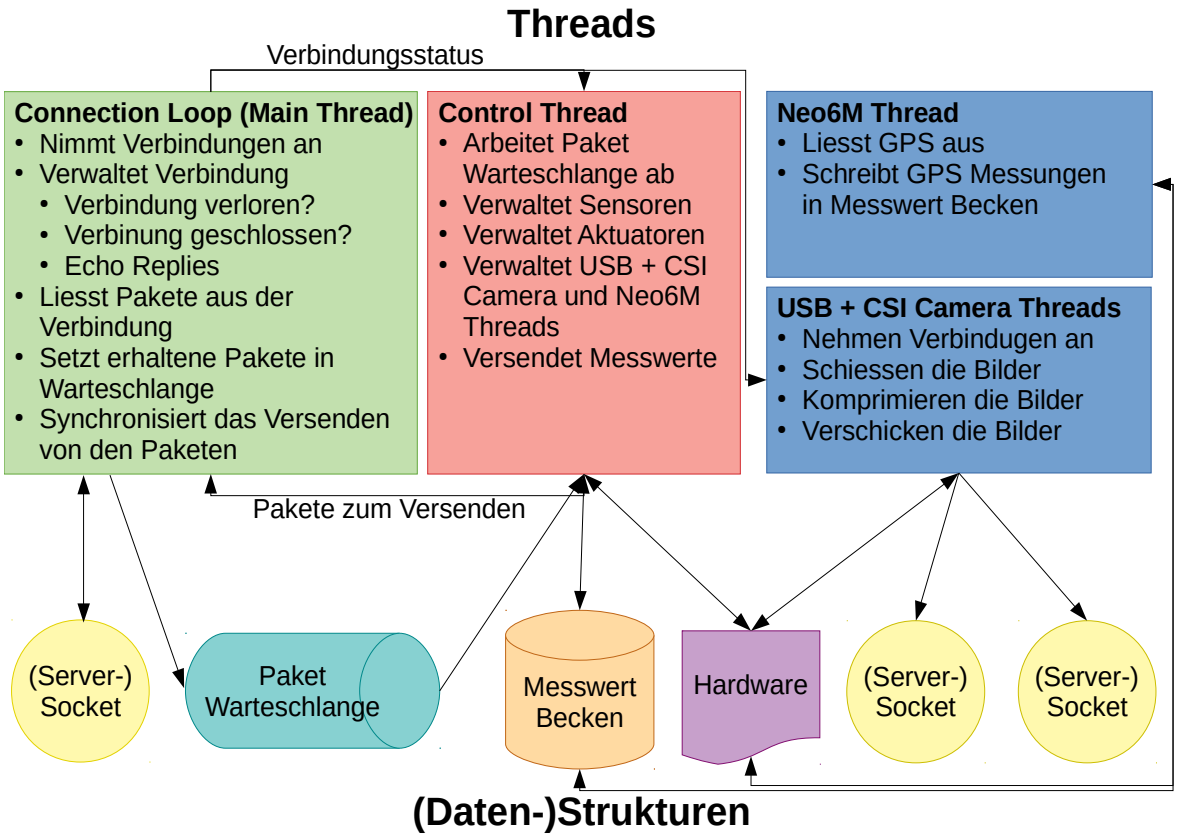


Illustration 25: Struktur Programm auf Raspberry Pi

7 Mit dem setzen von gewissen Flags kann der Aufruf nicht blockierend gemacht werden, das Programm würde dadurch aber nicht einfacher werden.

Die Kamera Threads verwalten jeweils eine eigene Verbindung und den zugehörigen Serversocket und Socket.

Bibliotheken

Das Programm auf dem Raspberry Pi greift auf drei Bibliotheken zurück. Libjpeg zum komprimieren der Daten von der USB Kamera, wiringPi und pigpio. Um auf die GPIO Pins des Raspberry Pis in C zuzugreifen, gibt es drei grosse Bibliotheken.

- bcm2835 https://www.airspayce.com/mikem/bcm2835/
- wiringPi http://wiringpi.com/
- pigpio http://abyz.me.uk/rpi/pigpio/

Zu Beginn des Projekts stellte sich die Frage auf welche der drei Bibliotheken wir unser Programm aufbauen wollten. Wir entschieden uns für wiringPi. Sowohl wiringPi als auch pigpio sind auf Raspbian[8] vorinstalliert. Vorteil von wiringPi im Gegensatz zu pigpio ist, dass wiringPi keine root Rechte benötigt und die Dateideskiptoren der unterliegenden Funktionalitäten offenlegt. Letzteres erlaubt den Funktionsumfang nach eigenen Bedürfnissen zu ergänzen und zu modifizieren. Da der Raspberry Pi nur zwei Hardware PWM Pins besitzt, wir 5 PWM Pins benötigten und wiringPi kein ausreichendes Software PWM unterstützt, mussten wir schlussendlich zusätzlich auf pigpio zurückgreifen.

Um einen einfachen Wechsel der GPIO Bibliothek zu ermöglichen ist der Programmcode vom Bibliothekszugriff getrennt. Unser gpio „Modul“[9] enthält Basisklassen, welche als Wrapper für die Bibliotheken dienen, auf denen die Programmteile, die auf die Hardware zurückgreifen müssen, aufbauen können.

Bugs

Das Programm auf dem Raspberry Pi ist nicht völlig fehlerfrei. Einerseits funktioniert das Auslesen der Distanzsensoren für grosse Distanzen nicht, anderseits misslingt manchmal das Initialisieren des Neo6M. Der erste Fehler liegt daran, dass das Programm eine Flanke messen muss die proportional zur Distanz ist. Dadurch dass auf dem Raspberry Pi mehere Threads und Prozesse laufen, kann während dem Warten auf das Fallen der Flanke ein Kontextwechsel auftreten. Kehrt der Prozessor zum, für die Messung verantwortlichen, Thread zurück, ist die Flanke längst gefallen. Folglich wird eine falsche Dauer gemessen. Dieses Problem könnte mit Interrupts gelöst werden. Das zweite Problem hat mit einem Fehler in der Synchronisation der Resourcen zwischen den Threads zu tun, dem wir leider noch nicht auf die Schliche gekommen sind.

Umfang

Das C/C++ Programm für den Raspberry Pi umfasst über 5000 Zeilen Code in über 30 Dateien. Um das Programm effizient auf dem schwachen Prozessor des Raspberry Pi zu kompilieren wurde ein Makefile System erstellt. Somit werden nur die veränderten Dateien neu kompiliert. Der Grund für den riesigen Umfang des Programms liegt darin, dass es für den Raspberry Pi weitaus weniger Bibliotheken für das Ansteuern von elektrischen Komponenten in C gibt als für den Arduino. Deshalb mussten wir für jeden Sensor das Datenblatt studieren und danach eine eigene Bibliothek schreiben.

8 Betriebssystem für den Raspberry Pi
9 Code im Ordner RaspberryPi/hardware/gpio/

7.4 Weiterentwicklung

Das gesamte Programm samt Dokumentation befindet sich auf Github, der populärsten Plattform für Quelloffene Softwareprojekte. Somit können andere Personen am Projekt weiterarbeiten. Besonders interessant für andere Entwickler sind die Bibliotheken für den Zugriff auf die Sensoren, da diese für beliebige Projekte mit dem Raspberry Pi wiederverwendet werden können. Wir haben uns besonders Mühe gegeben diese in den Headern sauber zu dokumentieren.

8 Rechtliches

Um die rechtliche Zulassung unseres geplanten Modellzeppelins abzuklären, mussten wir uns mit der Verordnung des UVEK über Luftfahrzeuge besonderer Kategorien genauer auseinandersetzen. Diese Verordnung gilt für Hängegleiter mit oder ohne elektronische Antrieb, Drachen, Drachenfallschirme, Fesselballone, Fallschirme und unbemannte Luftfahrzeuge. Der Zeppelin zählt als unbemanntes Luftfahrzeug, bzw. Modellluftfahrzeug. In diese Verordnung sind nur folgende Punkte für unser Projekt wichtig:

- Alle Luftfahrzeuge, die vorher erwähnt worden sind, müssen nicht in das Luftfahrzeugregister eingetragen werden.
- Für alle Luftfahrzeuge, die vorher erwähnt worden sind, ausser für Hängegleiter mit elektrischen Antrieb, besteht kein Zwang auf einem Flugplatz abzufliegen oder zu landen.
- Modellluftfahrzeuge dürfen nicht in fahrlässig oder vorsätzlich riskanter Weise betrieben werden, dass Menschenleben oder Sachen Dritter gefährdet werden.
- Ein Modellluftfahrzeug mit einem Gewicht bis 30 Kilogramm darf nur betrieben werden, wenn der Pilot direkten Augenkontakt mit dem Luftfahrzeug haltet und jederzeit die Steuerung gewährleisten kann.
- Modellflugzeuge mit einer Masse zwischen 0,5 und 30 Kilogramm dürfen nicht im Umkreis von weniger als 1000 Metern um Menschenansammlungen im Freien geflogen werden. (Die Masse des Zeppelins wird nur ca. 400 Gramm betragen. Das heisst wir dürfen den Zeppelin über Menschenansammlungen im Freien fliegen.)
- Die Sicherheit der Haftpflichtansprüche ist nicht erforderlich für Modellflugzeuge mit einem Gewicht von weniger als 0,5 Kilogramm.

9 Sponsor

Da das Heliumgas die grössten Kosten unseres Projektes ist, suchten wir nach jemandem, der uns das Helium sponsern würde. Als Gegenleistung würden wir das Logo des Sponsors auf die Hülle des Prallluftschiffes auftragen.

Die Firmen, die wir angefragt haben, einen Sponsor zu sein, sind im folgenden aufgelistet: Neue Aargauer Bank Baden, Raiffeisen Bank, A.L.K (Flugmodellbau Shop in Würenlingen), Aargauische Kantonalbank, Go easy, Sterk Kino, Schwimmbad Baden, Pangas, Messer, Lillo‘ Fitness-Träff.

Wir bekamen nur zwei Antworten auf das E-Mail. Die erste Antwort war vom Sterk Kino, welches die Möglichkeit, einen Sponsor zu sein absagte. Die zweite Antwort war vom Lillo’s Fitness-Träff (ein Fitnesszentrum in Döttingen). Wir kannten sie, da die Schwester von Paul dort eine Lehre als Fachfrau Bewegung- und Gesundheitsförderung hat. Sie stimmten zu, unseren Sponsor zu sein und die Kosten des Heliums zu übernehmen. Aus diesem Grund wird die Logo von Lillo’s Fitness-Träff auf die Hülle aufgetragen. Zudem würden wir, falls noch Helium nach der Präsentation übrig blieb, den Prallluftschiff an einem Zeit und Ort nach Wahl fliegen gehen, um Werbung zu machen.

Zum Auftragen des Logos erstellen wir eine Schablone anhand der Datei, die uns Lillo's Fitness-Träff zugeschickt hatten. Diese Schablone befestigten wir mit doppelseitigem Klebeband auf die Hülle und malten die Lücken mit schwarzen wasserfesten Eddingstifte aus. Das Mal-Prozess muss mehrmals durchgeführt werden, da die Striche des Stiftes ansonsten zu sehen sind.

10 Fazit

10.1 Vergleich Rechnung und Realität

Bei der Planung versuchten wir das Gewicht des Zeppelins möglichst auf das Gramm genau zu bestimmen, um das Volumen der Hülle und somit auch die Kosten für das teure Helium klein zu halten. Die folgende Tabelle vergleicht das geplante, berechnete Gewicht mit dem reellen, gemessen.

Komponente	Masse, geplant [g]	Masse, gemessen [g]	Differenz [g]
Hülle	174	203	29
Ventilklammer	6	4	-2
Ruder	38	38	0
Gondel + Elektronik	152	169	17
Batterie	72	69	-3
Total	442	483	41

Unschwer zu erkennen, liegen wir bei der Planung um rund 40g daneben. Ein Grund dafür ist, dass wir die Schutzringe und die Verstärkung der Motorenachse anfänglich nicht eingeplant hatten. Dies macht 10g aus. Die weiteren überschüssigen 30g sind der Hülle zu verschulden. Einerseits rechneten wir wahrscheinlich mit einer leicht zu kleinen Massenbelegung für die Folie, anderseits fiel die Hülle leicht grösser aus. Um das Überschüssige Gewicht zu kompensieren, entfernten wir die Distanzsensoren, die mit dem Stecker 23g ausmachten und das GPS, das 16g wiegt. Damit wir trotzdem mit GPS fliegen können, werden wir die Batterie mit einer kleineren ersetzen müssen.

Bereits ein Gewichtsunterschied von 5g hat beachtliche Konsequenzen für das Flugverhalten des Zeppelins. Am besten wird die Hülle zu gross gebaut und anschliessend mit kleinen Münzen beschwert.

Des weiteren darf der Einfluss der Temperatur nicht unterschätzt werden. Füllt man den Zeppelin in einem Gebäude mit 25 °C und geht danach nach draussen, wo es 5°C warm ist, reduziert sich das Volumen des Heliums um rund 7%, was in unserem Fall 35 Liter entspricht. Die Auftriebskraft bleibt aber identisch, weil die Luft draussen Dichter ist. Die Hülle wird dafür schlaffer. Hätte man den Zeppelin draussen gefüllt, hätte man etwa 35g zusätzlichen Auftrieb.

10.2 Schwierigkeiten

Am meisten Schwierigkeiten verursachte uns der Bau der Hülle. Bis in die Hälfte des Projektes befanden wir uns auf dem falschen Dampfer, indem wir versuchten eine Hülle aus Mylar zu bauen. Auch die rote Folie, die wir vom Silentrunner Team abgekauft hatten, löste die Probleme nicht vollständig. Die erste Hülle die wir aus der roten Folie anfertigten war völlig dicht. Wir füllten sie mit Luft und in den zwei Wochen, in der sie herumstand verlor sie kein Tröpfchen Luft. Indem wir sie mit Wasser füllten, um das Volumen zu bestimmen, was misslang, und sie sorglos zusammenfalteten, verlor sie ihre Dichtheit. Wir bauten eine zweite Hülle aus dem roten Material. Diese enthält aber noch kleine Lecks, die wir leider noch nicht finden konnten.

Eine weitere Herausforderung war das Testen. Wegen den hohen Kosten für das Helium lassen sich die Überlegungen nicht schnell überprüfen. Alle Versuche mussten vor dem ersten Flug sauber vorbereitet werden.

Dadurch dass beim Zeppelin jedes Gramm zählt, mussten wir im voraus an alle Komponente denken, die verbaut werden sollen. Möchte man im Nachhinein etwas hinzufügen, bedeutet das, dass man an einer anderen Stelle sparen muss, wenn man keine neue Hülle bauen will.

10.3 Resultat

Das wurde erreicht

Unser Ziel ein fernsteuerbaren Zeppelin zu planen, konstruieren und programmieren wurde erreicht. Der Zeppelin kann mit einem Radius von rund 3 m drehen und sogar rückwärts fliegen. Die grafische Oberfläche für die Steuerung zeigt ein flüssiges Bild der beiden Kameras. Die weiteren Sensoren können ausgelesen werden. Ihre Messwerte werden tabellarisch dargestellt. Die vom GPS gelieferte Position kann in einer Karte angeschaut werden. Des weiteren gelang es uns einen Sponsor für das Helium zu finden.

Ideen für die Erweiterung und Anwendung

Das Projekt bietet Unmengen an Erweiterungsmöglichkeiten. Eine Möglichkeit wäre einen Kalman-Filter zu erstellen um die Position und Orientierung des Zeppelins genauer zu bestimmen. Mit den Daten des Kalman-Filter liesse sich anschliessend ein Autopilot programmieren.

Eine weitere Idee wäre den Zeppelin für die Kartografie zu verwenden. Mit dem GPS, den Distanzsensoren, dem Barometer und den Kameras liesse sich das Profil eines Geländes scannen.

Eine völlig andere Anwendung wäre den Zeppelin kommerziell als Werbefläche zu fliegen.

10.4 Danksagung

Wir möchten uns bei unserem Sponsor „Lillo‘s Fitnessträf“ bedanken, der uns das Helium im Wert von über 200 CHF bezahlt hat. Desweiteren bedanken wir uns bei Andreas Burkart und André Sobotta vom Silentrunner Team für die rote Folie, unsere Eltern für finanzielle und technische Unterstützung und bei Stefan Widmer in dessen Werkstatt wir arbeiten und Material verwenden durften.

11 Anhang

11.1 Begriff Luftschiff, Zeppelin, Blimp

In diesem Dokument wurden die Begriffe Luftschiff und Zeppelin synonym verwendet. Eigentlich wäre dies nicht korrekt. Luftschiffe werden in 3 Klassen aufgeteilt:

- Bei Starrluftschiffe erhält die Hülle ihre Form durch ein Gerüst. Zu dieser Klasse gehören die Zeppeline.

- Bei Halbstarrluftschiffe erhält die Hülle ihre Form durch einen Überdruck. Die Hülle enthält aber trotzdem ein Teilgerüst für die Befestigung der Gondel, Motoren und Ruder.
- Prallluftschiffe erhalten ihre Form durch einen Überdruck. Der Englische Begriff ist Blimp. Zu dieser Klasse gehört der Silentrunner und unser Luftschiff.

11.2 Weitere Steuerungsmechanismen

Wir haben uns 6 zusätzliche Steuerungsmechanismen überlegt und verworfen, die wir im folgenden vorstellen. Anfänglich erschien uns eine Vektorsteuerung am günstigsten. Nach einem Austausch mit Andreas Burkart vom Silentrunner Team wurde uns klar, dass die Kräfte des Luftwiderstandes bereits bei relativ tiefen Geschwindigkeiten die Schubkraft der Motoren überwiegen. Aus diesem Grund entschieden wir uns schlussendlich für eine Steuerung mit den 3 Ruder.

Steuerung 1

Zwei Motoren, die unten am Zeppelin befestigt sind, sorgen für den Antrieb und Drehbewegungen in der x,y-Ebene. Ein Höhenruder, das mit einem Servo gesteuert wird, kompensiert die Drehmomente der Motoren und erlaubt ein Nicken gegen oben oder unten. Da die Motoren am Kasten befestigt sind, fällt eine Befestigungskonstruktion für die Motoren weg.

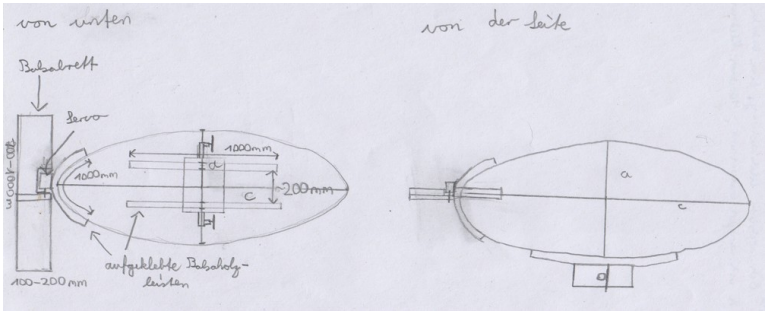


Illustration 26: Skizze Steuerung1

Material: 2 Motoren, 1 Servo, Höhenruderkonstruktion

Steuerung 2

Ein Motor kann anhand von zwei Servos in eine beliebige Richtung gedreht werden. Der Motor ist sowohl für das Steuern als auch für den Antrieb zuständig.

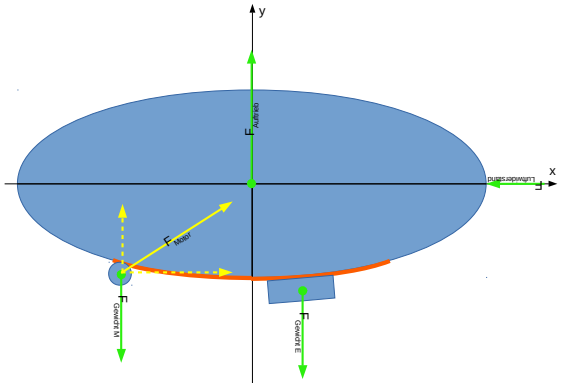


Illustration 27: Skizze Steuerung2

Material: 1 Motor, 2 Servos, Befestigungskonstruktion

Steuerung 3

Zwei Motoren werden parallel zur langen Halbachse an der Hülle fixiert. Mit zwei Servos können sie nach unten und oben Nicken.

Material: 2 Motoren, 2 Servos, Befestigungskonstruktion

Steuerung 4

3 Motoren werden fest in 120 Grad Winkel Abstand an der Hülle fixiert.

Material: 3 Motoren, Befestigungskonstruktion

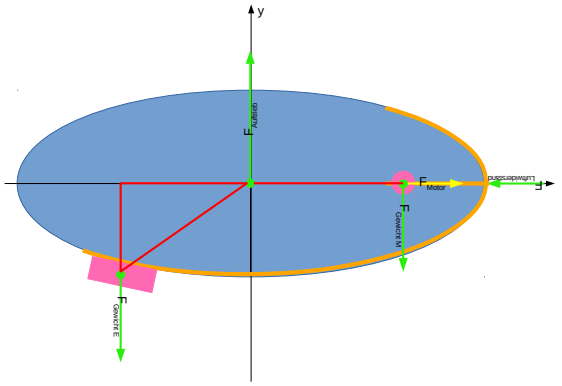


Illustration 28: Skizze Steuerung 3

Steuerung 5

Analog zu Steuerung 1, statt zwei Motoren wird ein Motor verwendet, der mit einem Servo in der x,y-Ebene Drehbar ist.

Material: 1 Motor, 2 Servos, Höhenruderkonstruktion

Steuerung 6

Analog zur Steuerung 1, der Motor wird aber unbeweglich befestigt. Dafür wird ein Seitenruder hinzugefügt.

Material: 1 Motor, 2 Servos, Höhenruderkonstruktion, Seitenruderkonstruktion

Vergleich

Das Gewicht der Steuerungen wurde abgeschätzt, indem den einzelnen Komponenten ein Gewicht zu geordnet wurde. Die in der Steuerung vorkommenden Komponenten wurden dann aufaddiert. Das Gewicht enthält nur die Komponenten, die für die Steuerung relevant sind und die Batterie. Raspberry Pi, Sensoren, usw. wurden nicht miteinberechnet. Um die Geschwindigkeit abzuschätzen wurde für Steuerung 1 ein Volumen von 450 Liter genommen. Das Volumen der anderen Steuerung wurden relativ zur Steuerung 1 berechnet. Für die Länge des Zeppelins wurde 1.75 m genommen.

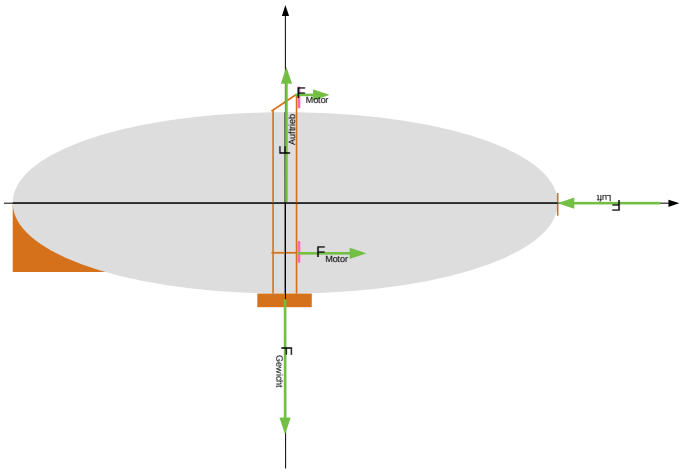
Steuerung	Gewicht mit Motor gross [g]	Gewicht mit Motor klein [g]	Geschwindigkeit mit Motor gross [m/s]	Geschwindigkeit mit Motor klein [m/s]
1	252	164	6,57	4,29
2	196	132	4,97	3,18
3	262	154	6,50	4,36
4	298	156	7,67	5,32
5	221	157	4,82	3,07
6	251	187	4,65	2,94

Steuerung	Vorteile	Nachteile
1	Gängige Lösung, Höhenruder bei hohen Geschwindigkeiten Effektiv	Ineffektiv bei kleinen Geschwindigkeiten, Höhenruderkonstruktion
2	Geringes Gewicht	Aufwendige Konstruktion mit Servos, kein einfaches gerade aus fliegen möglich, beim Neigen Schubkomponente in entgegengesetzte Richtung zur gewünschten
3	Wendigkeit	Komplizierte Befestigung der Motoren an der Hülle, Konstruktion mit Servos
4	Wendigkeit, keine Servokonstruktionen	Gewicht, Befestigung der Motoren
5	Kleineres Gewicht als Steuerung 1	Langsamer und Träger als Steuerung 1

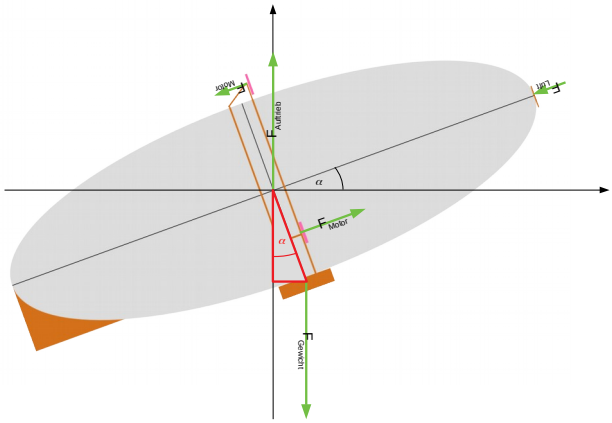
6	Höhen- und Seitenruder bei hohen Geschwindigkeiten Effektiv, Gängige Lösung	Ineffektiv bei kleinen Geschwindigkeiten, Seiten- und Höhenruderkonstruktion

Die Steuerung Nummer 4 mit den kleinen Motoren erschien uns interessant. Aus diesem Grund arbeiteten wir diese Steuerung genauer aus. Sie bietet eine hohe Geschwindigkeit und Wendigkeit. Nur Steuerung Nummer 2 mit den kleinen Motoren ist leichter. Ausserdem entfallen Konstruktionen mit Gelenke und Servos. Um die Stabilität um die z-Achse zu erhöhen, fügten wir hinten eine unbewegliche Finne hinzu, die es bei allen Steuerungen bis auf Nr. 6 benötigt.

Genauere Überlegungen



Drawing 2: Kräfte beim geradeaus Fliegen



Drawing 3: Kräfte beim Steigen

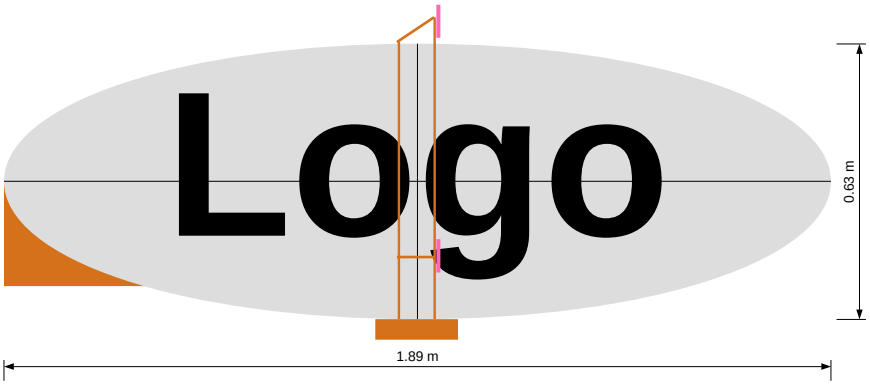
Kippen wir den Zeppelin entsteht durch die Gewichtskraft des Kästchens ein Drehmoment.

$$M_{Kasten} = r \cdot \sin(\alpha) \cdot F_{Gewicht} \approx 0.32\,m \cdot \sin(\alpha) \cdot 2.75\,N = 0.88\,Nm \cdot \sin(\alpha)$$

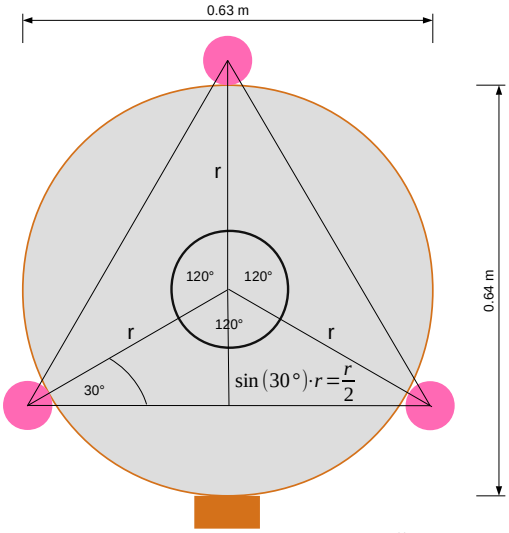
Mit den Motoren können wir maximal folgendes Drehmoment aufbringen.

$$M_{Motor} = 2 \cdot \frac{r}{2} \cdot F_{Motor} + r \cdot F_{Motor} \approx 0.32\,m \cdot 0.7\,N \cdot 2 = 0.45\,Nm$$

Folglich ist der maximale Neigungswinkel 31°. Wollen wir den oberen Motor nicht rückwärts laufen lassen, beträgt der Winkel 15°. Beim Beschleunigen und Bremsen entsteht ebenfalls ein Drehmoment. Da in diesem Fall die Summe der Kräfte ungleich null ist, müssen wir das Drehmoment um den Schwerpunkt und nicht einen beliebigen Punkt (vorher Zentrum des Ellipsoiden) rechnen. Dies bedeutet, dass die Nase unseres Zeppelins beim Beschleunigen leicht nach unten und beim Bremsen leicht nach oben schaut.



Drawing 5: Steuerung 4 von der Seite



Drawing 4: Querschnitt beim Äquator

11.3 Alternative Hülle aus Mylar

Bevor wir auf das Material für den Hüllenbau des Silentrunner gestossen sind, versuchten wir die Hülle aus Mylar, das als Rettungsdecke gekauft werden kann, zu bauen. Damit verbrachten wir das erste Drittel der von der Schule zur Verfügung gestellten Zeit. Das Material des Silentrunner erlaubt ein weitaus einfacheres und erfolgversprechendes Verfahren zum Hüllenbau als Mylarfolie, weil es sich sauber verschweissen lässt. Aus diesem Grund gaben wir den Ansatz mit Mylar auf. Eine Mylarhülle hätte hingegen den Vorteil eines deutlich geringern Gewichtes, da es um ein Faktor drei leichter ist. Die Ergebnisse unserer Versuche mit Mylar sind im folgenden dokumentiert.

Verbinden der Mylarfolie
Um eine optimales Verfahren zu finden, bei dem man zwei Mylarfolien zusammenfügt, mussten wir mehrere Materialienarten und Methoden des «Zusammenschweissens» ausprobieren. Als Verbindungsmaterialien benutzten wir unterschiedliche Klebstoffe und Klebebänder. Mit einem Heissluftföhn, einem Bügeleisen und einer speziellen Lötkolbenkonstruktion prüften wir die verschiedenen Arten des Zusammenschweissens. Die nächste Tabelle zeigt, ab welcher Kraft die zwei Folien auseinanderreissen. Als Referenz prüften wir auch die Kraft, die es braucht, bis ein Stück der Folie reisst.

Art des Zusammenfügens	Reisskraft	Kommentar
Reissfestigkeit der Folie (~35mm breite Folie)	50 Newton	Dieser Test dient als Referenzwert.
UHU Kontakt	Folie reisst bei 40 [N]	Die Folie riss bei 40 [N]. Die zusammengeklebte Seite war noch vorhanden.
Pattex Plastik	Folie reisst bei 40 [N]	Die Folie riss bei 40 Newton. Die zusammengeklebte Seite war noch vorhanden.
UHU Plus	Das Zusammenhalten der Folien ist ab 20[N]	Der Klebstoff kann die zwei Folien nur knapp bis 20 [N] zusammenhalten. Danach lösen sich die zwei Folien voneinander.

	nicht mehr stabil.	
Loctite, Sekundenkleber	Überlappungsbereich reisst bei etwa 25 [N]	Der Klebstoff löste beide Folien auf und schwächte sie dadurch. Dies führt dazu, dass das Überlappungsbereich geschwächt ist und bei niedriger Zugkraft reisst.
Scotch doppelseitig Kleber	Folie reisst bei 40 [N]	Die Folie riss bei 40 Newton. Die zusammengeklebte Seite war noch vorhanden.
Mit einem Heissföhn	-	Misserfolg: Die Folie verdunstet. Der Heissföhn ist zu warm.
Mit einem Bügeleisen	Folie reisst bei leicht unter 50 [N]	Die Folie riss bei leicht unter 50 Newton. Die zusammengebügelte Seite blieb stabil. Jedoch gibt es viele kleine Wellen bei der Schweissnaht.
Mit einer Lötkolbenkonstruktion	Folie reisst bei leicht unter 50 Newton	Die Folie riss bei leicht unter 50 Newton. Die zusammengeschweisste Stelle blieb stabil.

Die Resultate zeigen, dass die Folie gut zusammengeklebt werden kann, wenn wir UHU Kontakt, Pattex Plastik oder Scotch doppelseitig Kleber benutzen würden. Wir könnten die Folie auch mit unserer Lötkolbenkonstruktion zusammenschweissen. Da das Zusammenschweissen schneller, weniger material- und kostenaufwendig ist und eine höhere Reissfestigkeit aufzeigt, eignet es sich am besten, die Mylarfolie zusammenzufügen. Jedoch entstehen kleine Wellen beim Zusammenscheissen, welche zur undichten Stellen führen könnten.

Illustration 29: Klebeversuche

Erfahrungen aus dem Bau des Prototypen

Um Erfahrungen zu sammeln, bauten wir einen Protoypen der Hülle mit 60 cm Länge und einen Durchmesser von 26 cm.

Ausschneiden von Stücken aus der Folie

Zum Ausschneiden von Stücken aus der Folie baut man am besten eine Schablone aus Holz. Danach spannt man die Folie über ein glattes Brett und fährt mit einem Japanmesser der Schablone nach. Bei zu schnellem Schneiden oder ungenügendem Spannen bildet die Folie Falten. Dadurch wird die Folie zufällig verrissen und das auszuschneidende Stück ist unbrauchbar.

Eigenschaften der Rettungsdecke

Die Rettungsdecke ist eine Polyesterfolie, die einseitig mit Aluminium beschichtet wurde. Die beschichtete Seite glänzt silbrig, während die andere Seite, durch das Schimmern des Aluminium durch die Folie, goldig erscheint. Aus diesem Grund lassen sich zwei goldige Seiten am besten „verschweissen“. Laut Wikipedia lässt sich die Aluminium Schicht mit einer Natriumcarbonat-Lösung entfernen. Uns ist es nicht gelungen auf diese Weise die Aluminium Schicht abzulösen. Bei Hitze (Temperaturen über 150°C) zieht sich die Folie zusammen. Dies kann nicht genutzt werden, um die Folie in eine gewünschte Form zu schmelzen bzw. zu dehnen. Beim Dehnen reisst die Folie und beim Schmelzen zieht sich die Folie zu einer Art Blasen zusammen, die zu Löcher führen können.

Eigenschaften der Schweissnaht

Die Folie lässt sich nicht sauber verschweissen. Wenn man mit dem Bügeleisen oder einem Lötkolben über zwei Folienstücke fährt, heften sie aneinander. Bei Zug reisst die Folie vor der „Schweissnaht“. Die Folienstücke lassen sich aber mit Schälkräften einfach voneinander lösen. Beim Bewegen über dünne Radien löst sich die Schweissnaht, wegen den auftretenden Schälkräften.

Probleme

Indem ein Schlauch aus Mylar geschweisst, mit Luft gefüllt und in ein Wasserbecken getaucht wurde, konnten wir überprüfen, ob die Schweissnaht Gasdicht ist. Obwohl dieser Test positiv verlief, ist der Prototyp nicht dicht. Die Probleme sind in folgender Tabelle aufgelistet.

Nr.	Problem	Grund	Mögliche Lösungsansätze
1.	Die Mylarfolien werden durch unsere Konstruktion mit dem Lötkolben nicht richtig verschweisst. Bei Belastungen mit hohen Schälkräften, zum Beisbiel beim Knittern in der Hand, öffnen sich die “Schweissnähte”.	Die Gründe könnten sein: zu tiefe Schweiss Temperatur, zu kleiner Anpressdruck, zu kurze Zeit zum ineinander verschmelzen der Folien, eine Behandlung der Oberfläche die ein richtiges Schweissen verunmöglicht oder die Eigenschaften der PET-Folie.	1. Bau eines Lötfusses für einen Regelbaren Lötkolben. Lötfuss muss länger im Kontakt mit der Folie sein. 2. Verwenden von Kleber oder speziellem Doppelseitigem Klebeband. 3. Über Backpapier, direkt mit dem Lötkolben auf die Folie. Danach mit einer Walze der Naht nach fahren. 4. Schweissnähte mit Klebeband verstärken und vor Belastungen schützen.
2.	Enden des Ellipsoiden sind nicht dicht.	Beim Bau werden die Endstücke stark belastet, weil wir oft mit der Lötkolben-	1. Die Endstücke werden mit einer Kreisscheibe oder einem Kegel überklebt bzw. ersetzt.

		konstruktion darüber fahren. Des weiteren befinden sich an dieser Stelle viele Falten.	Kreisscheibe evtl. einschneiden um Falten zu minimieren. 2. Statt eines Ellipsoiden wählen wir eine andere Form. Zum Beispiel ein Zylinder mit zwei Kegel als Endstücke. Nachteil wären vor allem die Knicke an den Übergängen, die zu Brüchen im Mylar führen könnten.
3.	Letzte Schweissnaht	Um die Folien zu kleben oder zu verschweissen brauchen wir eine harte Unterlage. Dies wurde mit einer Holzkonstruktion realisiert. Das Problem ist, dass das letzte Stück der Schweissnaht ohne diese Konstruktion auskommen muss, da diese sich sonst in der Hülle befindet.	Kleben von zwei Holzscheiben am Ende zwischen denen das Mylar eingeklemmt ist. In die Holzscheiben kann dann das Ventil eingelassen werden. Da Holz nicht Gasdicht ist, müssen die Scheiben unbedingt lackiert werden.
4.	Ventil		1.Ventil nach Vorbild eines Alten Fahrradventils bauen 2.Ventil eines Folienballons verwenden 3.Röhrchen, das aus Mylar besteht und einfach zugerollt wird.

Vorgehen für den Bau der definitiven Hülle

Aus den Erfahrungen vom Bau des Prototypen, schlagen wir folgendes Vorgehen für den Bau einer Hülle vor. Punkte 4 und 5 haben wir nicht am Prototypen ausprobiert.

1. Ausschneiden der Gores und einer Kreisscheibe/Kegel aus einer Rettungsdecke. Rettungsdecke über ein glattes Brett spannen, Schablone darauf legen und mit dem Japanmesser der Schablone nachfahren.
2. Aussägen der zwei Balsaholz Kreisscheiben für das Ventil. Bohren von jeweils zwei Löcher. Ein Loch für das Einlassventil. Ein Loch für das Auslassventil. Bau des Basalholzröhrchen/ Ventil. Zurechtschneiden des Mylarstück und des Luftballonstück. Lackieren der Balsaholzteile.
3. Die Gores mit Hilfe einer Konstruktion verschweissen. Am einten Ende die letzten 4 cm noch unverschweisst lassen. Die Schweissnähte mit Verpackungsklebeband überkleben. Das letzte Gore nur bis zur Hälfte verschweissen.
4. Abschneiden der verschweissten Spitze. Einführen eines harten Gegenstandes (z.B. hart aufgepumpter Ball). Gegenstand als Unterlage für das Aufkleben der Kreisscheibe/Kegel an der Spitze verwenden. Stellen vor dem Kleben reinigen. (Den Rand nach dem Kleben mit Verpackungsklebeband abkleben.) Gegenstand aus der Hülle entfernen und fertig verschweissen. Vor dem Abschluss eine der Balsaholzkreisscheiben in die Hülle legen.

5. Vorher hineingelegte Balsaholzscheibe zur Öffnung bringen. Einlassventil in die Scheibe einkleben. Endstücke der Mylarfolie auf die Scheibe kleben. (Den Rand mit Verpackungsklebeband abkleben.) Zweite Scheibe über das Ventil stülpen und auf die Erste kleben. Ventil abdichten.

Materialliste für den Bau

Punkt	Material	Anzahl
1, 2	Rettungsdecke 210cm x 160cm	2
1	Schablone für Gores (Schablone für ¼ des Gores reicht)	1
1	Schablone für Kegel	1
1	Glattes Brett	1
2	Kleiner Luft/Wasserballon	1
2	Balsaholzbrett 3x80x80 mm	1
2	Balsaholzlatte 20x20x100 mm	1
2	Lack	-
3	Konstruktion zum verschweissen	1
3	Lötfuss	1
3	Verpackungsklebeband	-
4,5	Kleber (UHU Kontakt Kraftkleber)	-

Punkt	Werkzeug
1	Kutter
2	Bandsäge oder Stichsäge
2	Bohrmaschine mit Bohrer
2	Drehbank
2	Pinsel
3	Lötkolben
4	Harter runder Gegenstand (ungefähr Form der Spitze des Ellipsoiden) z.B. Fussball

Skizzen für den Bau

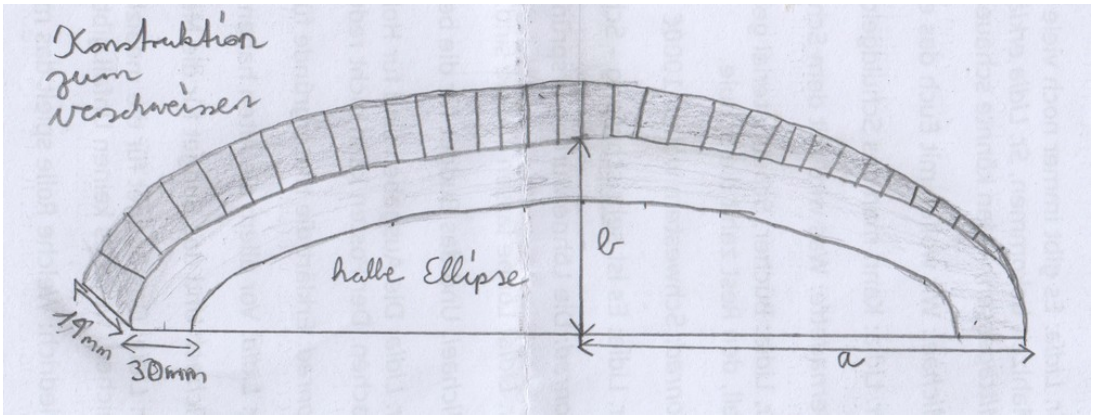


Illustration 30: Konstruktion für das Verschweissen

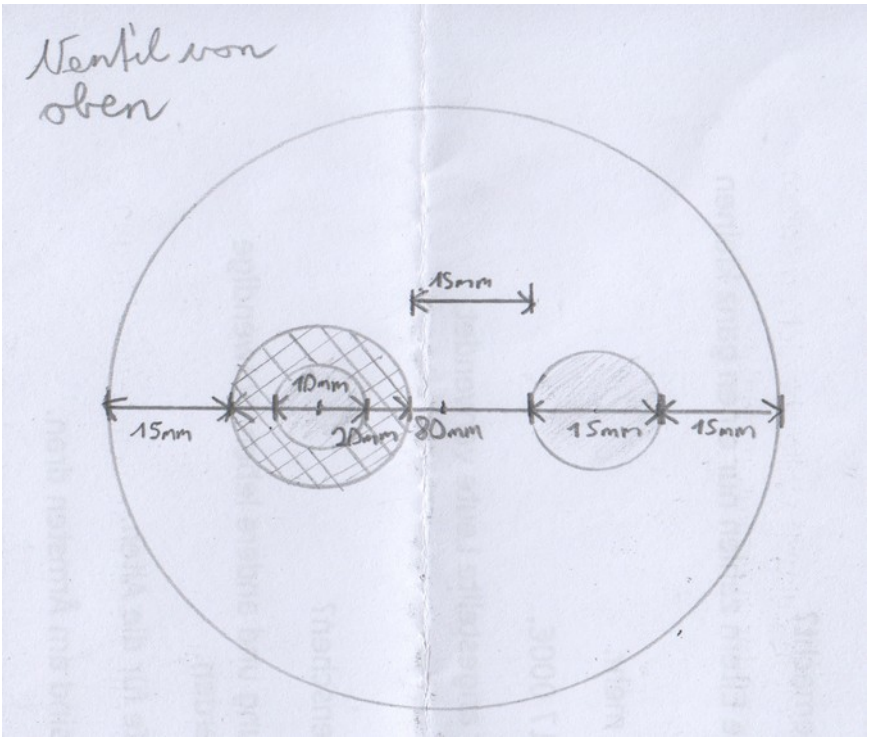


Illustration 31: Ventil von Oben

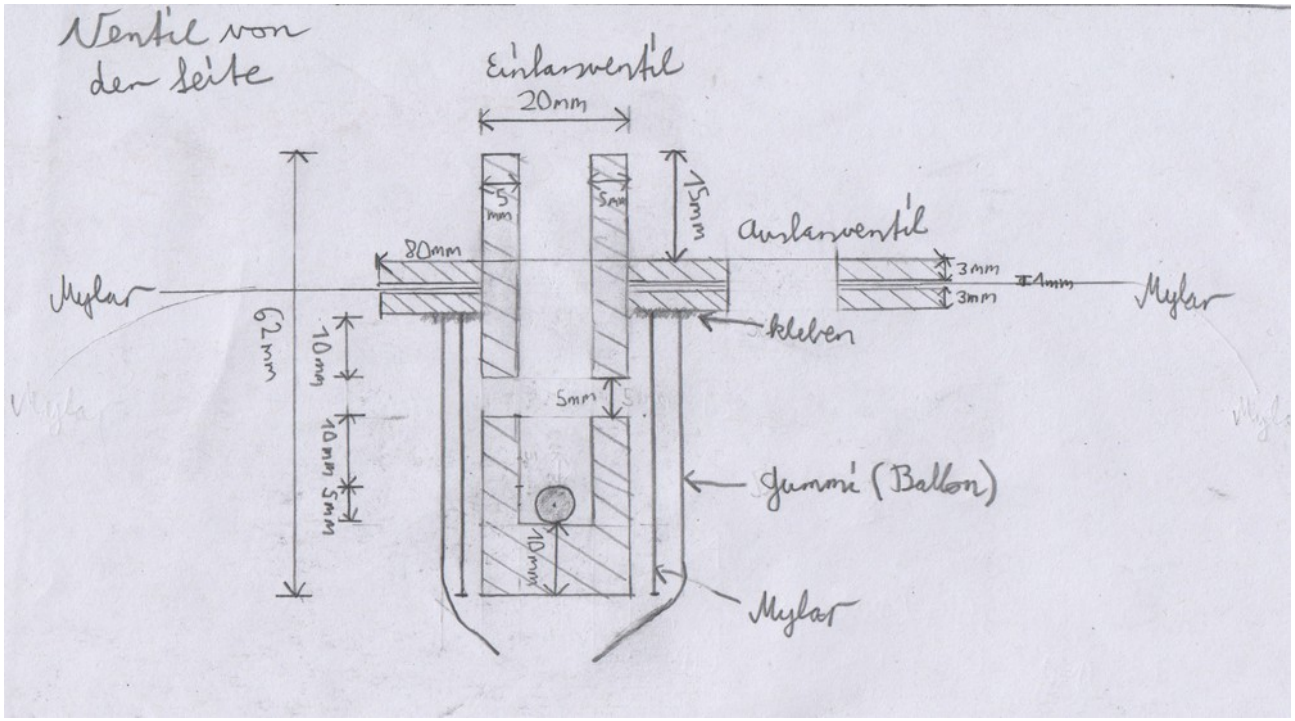


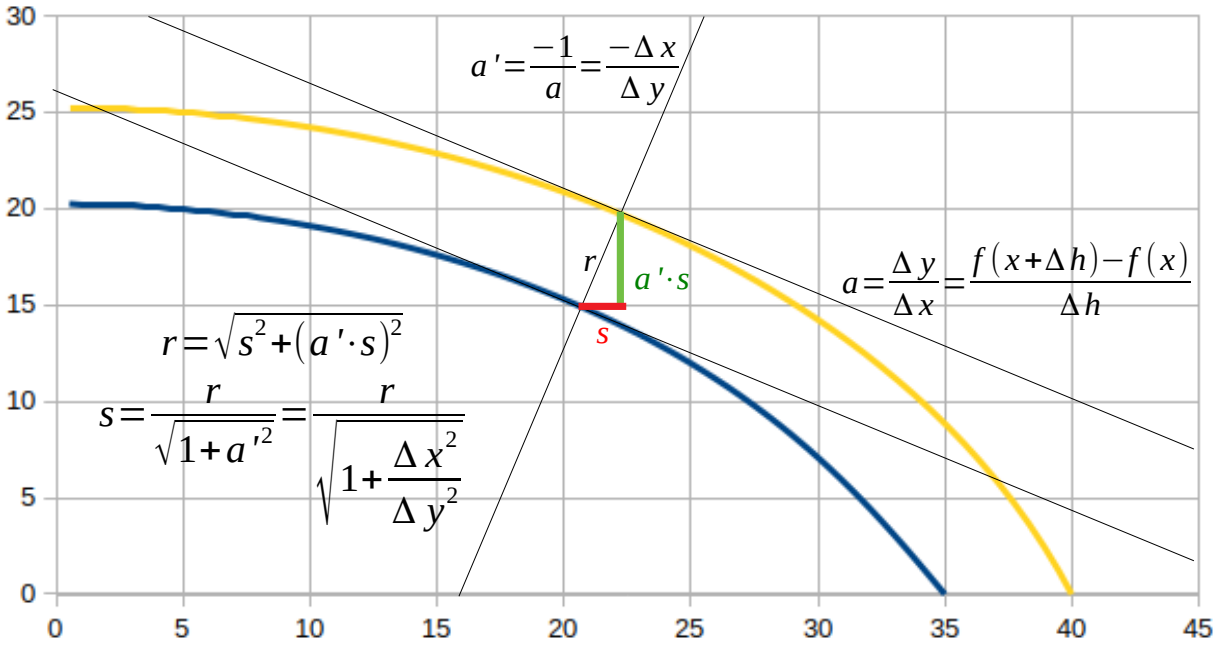
Illustration 32: Ventil von der Seite

Das Mylar für das Ventil kann weggelassen werden, da das Einlassventil ebenfalls zugeklebt wird. Statt den Ballon festzukleben, kann eine Nut in die Röhre gefrässt werden, um ein abrutschen zu verhindern.

Rand

Die Gores ergeben zusammen die Oberfläche des Rotationskörper. Bei diesem Bauverfahren wird noch ein Überlappungsbereich zum Verschweissen benötigt. Um diesen zu erzeugen, muss eine

Linie parallel zum Graphen in einem konstanten Abstand gezogen werden. Der Abstand soll dem halben Überlappungsbereich entsprechen, da beide Gores, die miteinander verschweisst werden, einen Rand erhalten.



Drawing 6: Herleitung der Berechnung des Randes

Die berechneten Punkte der Gores müssen um s in x-Richtung und um a‘ mal s in y-Richtung verschoben werden.

11.4 Position- und Orientationsbestimmung

Um einen Autopilot programmieren zu können, muss dem RaspberryPi die Lage des Zeppelins und dessen Orientierung bekannt sein. Das GPS liefert eine grobe Positionsbestimmung und einen Wert für den Azimut[10]. Der Magnetfeldsensor gibt Informationen zu den Winkeln des Zeppelins relativ zum Norden (Gieren), zum Boden (Nicken) und dem Rollen. Weitere Informationen zur Lage und Orientierung des Zeppelins können wir vom Barometer, vom Beschleunigungssensor und dem Gyroskop erhalten. Nun stellt sich die Frage wie man die Informationen dieser Sensoren zusammen führen kann, um genauere und höherfrequente Positions- und Orientationsbestimmungen zu bekommen. Ein Ansatz ist der Kalman-Filter. Die Zeit reichte uns leider nicht, um uns vollständig in dieses Thema einzulesen und einen Kalman-Filter umzusetzen. Wir konnten aber die Grundprinzipien verstehen.

Grundprinzip eines Kalman-Filters

Der Kalman-Filter enthält ein Model des Systems. Anhand der letzten bestmöglichen Information über den Zustand des System und der externen Einflüssen berechnet es eine Voraussage für den Zustand des Systems im nächsten Schritt. Beim Voraussagen wird an Genauigkeit verloren. Anchliessend erhält der Filter eine Messung mit einem gewissen Fehler. Die Messung und der Fehler der Messung werden mit der Voraussage und dessen Fehler verrechnet. Heraus kommt unsere bestmögliche Information über den neuen Zustand des Systems und einen zugehörigen Fehler. Die Information hat im Vergleich zur Voraussage wieder an Genauigkeit gewonnen.

10 Winkel zum Norden

Der Filter verwendet Matrizen um die verschiedenen Dimensionen (Position, Geschwindigkeit, etc.) darzustellen und arbeitet mit Gaussglocken für den Fehler.

Für detailreichere Lektüre sind folgende Webseiten empfohlen:
Intuitiv erklärt mit Bildern: https://www.bzarg.com/p/how-a-kalman-filter-works-in-pictures/
Interaktives Buch, geht in die Tiefe: https://github.com/rlabbe/Kalman-and-Bayesian-Filters-in-Python

12 Referenzen

Recherche
Webseite des Silentrunners, aufgerufen am 12.06.18
http://silent-runner.net/index.php?title=Main_Page

Webseite des Teams hinter dem Bau des Silentrunners, aufgerufen am 12.06.18
http://windreiter.de/wordpress/

Anleitung zum Bau eines einfachen Zeppelins aus Mylar, aufgerufen am 12.06.18
http://www.rc-network.de/forum/showthread.php/250045-Baubeschreibung-einfaches-RC-Luftschiff

Traggas
Erklärungen zur Auftriebskraft, aufgerufen am 18.02.18
https://www.leifiphysik.de/mechanik/auftrieb-und-luftdruck

Erklärungen zur Luftdichte, aufgerufen am 18.02.18
https://de.wikipedia.org/wiki/Luftdichte

Hülle
Bild des Stromlinienkörpers, aufgerufen am 17.04.18
http://www.gunt.de/images/datasheet/777/HM-170.08-Widerstandskoerper-Stromlinienkoerper-gunt-777-foto_totale.jpg

Elektronik und Komponente

Entladekurve Lipo Batterie, aufgerufen am 26.10.18
https://www.pedelecforum.de/forum/proxy.php?image=http%3A%2F%2Fwww.pedelec-forum.de%2Fforum%2Fattachment.php%3Fattachmentid%3D3150%26stc%3D1%26d%3D1253960585&hash=b448f46fc84cb78671e3acd473f6d592

Schema I2C Bus, aufgerufen am 26.10.18
http://www.lucadentella.it/blog/wp-content/uploads/2017/09/i2c-01.jpg

Steuerung
Windgeschwindigkeit bei Baden, aufgerufen am 30.05.18
https://wind-data.ch/windkarte/

Drehrichtungen Flugzeug, aufgerufen am 01.11.18

https://moodle.ruhr-uni-bochum.de/m/file.php/600/images/flugzeugrotationen.png

Rechtliches
Gesetze zur Zulassung von Modellflieger, aufgerufen am 06.06.18
https://www.admin.ch/opc/de/classified-compilation/19940351/index.html

Anhang
Rettungsdecke/Mylar, aufgerufen am 06.06.18
https://de.wikipedia.org/wiki/Rettungsdecke

Bau einer Hülle aus Mylar durch Verleimen, aufgerufen am 12.06.18
http://www.bobbear.org/rcblimp/mylar/index.htm