FRIEDRICH-ALEXANDER-UNIVERSITÄT ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT • DEPARTMENT INFORMATIK

Lehrstuhl für Informatik 10 (Systemsimulation)

Simulation eines Solar-Wind-Luftschiffes

Sören Koenen

Bachelorarbeit

Simulation eines Solar-Wind-Luftschiffes

Sören Koenen

Bachelorarbeit

Aufgabensteller:	Prof. Dr. C. Pflaum
Betreuer:	Prof. Dr. C. Pflaum
Bearbeitungszeitraum:	13.05.2019 – 14.10.2019

Erklärung:

Ich versichere, dass ich die Arbeit ohne fremde Hilfe und ohne Benutzung anderer als der angegebenen Quellen angefertigt habe und dass die Arbeit in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen hat und von dieser als Teil einer Prüfungsleistung angenommen wurde. Alle Ausführungen, die wörtlich oder sinngemäß übernommen wurden, sind als solche gekennzeichnet.

Der Universität Erlangen-Nürnberg, vertreten durch den Lehrstuhl für Systemsimulation (Informatik 10), wird für Zwecke der Forschung und Lehre ein einfaches, kostenloses, zeitlich und örtlich unbeschränktes Nutzungsrecht an den Arbeitsergebnissen der Bachelorarbeit einschließlich etwaiger Schutzrechte und Urheberrechte eingeräumt.

Erlangen, den 13. Oktober 2019 .....

1 Zusammenfassung

Der Antrieb eines Zeppelins durch auf der Zeppelinoberfläche gewonnene Solarenergie ist ein vielversprechender Ansatz für die emissionsfreie Luftfahrt. Bevor neue und kostspielige Vorhaben wie dieses jedoch in die Realität umgesetzt werden können, sind umfassende Simulationen erforderlich, um möglichst belastbare Daten schon vor der Umsetzung zu erhalten. Nachdem sich die Arbeiten von Kai Teichmann und Christopher Newman [16][13] entweder mit dem Einfluss der Sonne oder dem Einfluss des Windes beschäftigt haben, wurden in dieser Arbeit beide Teile miteinander verbunden.
Weiterhin wurde durch die Einführung einer Benutzeroberfläche die Arbeit mit der vorhandenen Simulationssoftware für die weitere Forschung vereinfacht. Durch weitere Anpassungen wurde auch die Änderung von Parametern in der Simulation ohne eine erneute Übersetzung des Programmcodes ermöglicht. Abschließend wurden die Auswirkungen von Jahres- und Tageszeiten auf die Flugdauer, sowie unterschiedliche Routen miteinander verglichen.

Inhaltsverzeichnis

1 Zusammenfassung **4**

2 Inhaltsverzeichnis **5**

3 Einleitung **6**

4 Benutzeroberfläche **7**
4.1 Aufbau der Oberfläche . . . 7
4.1.1 Zeppelin . . . 7
4.1.2 Route . . . 8
4.1.3 Simulation . . . 8
4.1.4 Log . . . 9
4.1.5 Results . . . 9
4.1.6 Settings . . . 9
4.2 Einfache Berechnungen in der Oberfläche . . . 10
4.2.1 Auftrieb und Nutzlast . . . 10
4.2.2 Maximale Flughöhe . . . 10
4.2.3 Kenndaten des Zeppelins . . . 11
4.3 Speichern und Laden . . . 11
4.4 Kommunikation mit der Simulation . . . 12

5 Erweiterung des Partikelschwarm Optimierungsalgorithmus von Christopher Newman **13**
5.1 Der Partikelschwarm Optimierungsalgorithmus . . . 13
5.2 Werteänderung ohne Recompiling . . . 13
5.3 Berechnung der Solarenergie . . . 14
5.3.1 Berechnung der Sonnenposition . . . 14
5.3.2 Berechnung der Abschwächung durch die Atmosphäre . . . 14
5.3.3 Diskretisierung der Zeppelinhülle . . . 15
5.3.4 Energieberechnung . . . 16
5.4 Geschwindigkeitsberechnung . . . 17
5.5 Analyse des besten Pfades . . . 17
5.6 Debugausgaben . . . 18

6 Auswirkungen der Einbeziehung der Sonnenenergie **19**
6.1 Auswirkungen der Sonne und des Windes . . . 19
6.2 Vergleich von durchschnittlicher und berechneter Geschwindigkeit . . . 21
6.3 Einfluss der Jahreszeiten . . . 22
6.4 Einfluss der Startzeiten . . . 23
6.5 Einfluss der Flugstrecke . . . 24

7 Auswirkungen auf veschiedene Flugstrecken **26**
7.1 Nürnberg - Vancouver . . . 26
7.2 New York - Vancouver . . . 26

8 Fazit **29**

9 Literaturverzeichnis **30**

10 Anhang **31**
10.1 Verwendete Tools und Bibliotheken . . . 31
10.2 Abkürzungsverzeichnis . . . 31
10.3 Klassendiagramm UI . . . 32

3 Einleitung

Im Zuge der aktuellen Klimadiskussionen rund um Greta Thunberg und die von ihr initiierte „Fridays for Future“-Bewegung ist der globale Flugverkehr und dessen Treibhausgasemmisionen wieder in den Fokus der Bevölkerung gerückt worden. Kommerzielle Flugreisen waren im Jahr 2018 für 2,4% des globalen CO_2 Ausstoßes verantwortlich [4]. Dieser Beitrag mag niedrig erscheinen, jedoch werden dabei weder andere ausgestoßene treibhausaktive Gase noch die große Höhe der Freisetzung beachtet. Auch die Auswirkungen von sich bildenden Kondensstreifen und deren Einfluss auf die Bildung und Veränderung von Zirruswolken werden dabei nicht berücksichtigt. Nach einer Studie des Umweltbundesamtes von 2009 ist der Treibhauseffekt von Flugreisen ungefähr 2 mal so hoch wie der alleinige Effekt des ausgestoßenen CO_2, jedoch wird bei dieser Analyse aufgrund von noch nicht ausreichender Sicherheit in den Berechnungen der Effekt von Kondensstreifen und Zirruswolken nicht einbezogen[18].
Ein Ansatz, um Treibhausemissionen zu verhindern, ist der Einsatz synthetischer Kraftstoffe. Diese setzen bei der Verbrennung die gleiche Menge an CO_2 frei, wie bei der Erzeugung unter Zuhilfenahme erneuerbarer Energien gebunden wurde. Der Flugverkehr mit dieser Art von Treibstoffen wäre dann zwar CO_2-neutral, jedoch nicht klimaneutral, da die Effekte in der oberen Atmosphäre, wie beispielsweise Kondensstreifen, erhalten bleiben würden. Klimaneutrales Fliegen ist daher vor allem durch Elektroflugzeuge möglich. Bei diesen Flugzeugen sind jedoch die verhältnismäßig schweren Akkumulatoren ein limitierender Faktor, da diese relativ groß sein müssen, um die Energie für den Vor- und vor allem den Auftrieb bereitstellen zu können. Die Energiegewinnung während des Fluges mittels Solarzellen ist aufgrund des derzeit verhältnismäßig geringen Wirkungsgrades dieser ebenfalls nicht sinnvoll umsetzbar.
Ein aktuell kommerziell weniger beachtetes Luftfahrzeug bietet für die Nutzung von Solarzellen zur Energieerzeugung bessere Voraussetzungen. Luftschiffe erhalten ihren Auftrieb durch den mit Traggas (meist Helium) gefüllten Auftriebskörper. Um das Luftschiff in der Luft zu halten ist also kein weiterer Energiebedarf vorhanden. Weiterhin befindet sich auf der Oberfläche der Auftriebskammer viel Platz, welcher mit Dünnschichtsolarzellen bestückt werden kann, um während des Fluges (zumindest tagsüber) laufend CO_2- und klimaneutrale Energie bereitstellen zu können. In dieser Arbeit wird die bestehende Simulation aus der Bachelorarbeit „Optimal Control of a Solar Airship“ von Christopher Newman erweitert, um die Möglichkeiten des Einsatzes eines Solar-Luftschiffs noch besser untersuchen zu können.

4 Benutzeroberfläche

Um die Erstellung von passenden XML-Dateien, welche die Werte der unterschiedlichen Parameter an die Simulation übergeben, zu erleichtern, wurde eine Benutzeroberfläche programmiert. Diese stellt einerseits sicher, dass die erzeugten XML-Dateien richtig formatiert vorliegen, um von dem Simulationsprogramm problemlos verarbeitet werden zu können, andererseits werden hier schon erste Plausibilitätsprüfungen und kleinere Berechnungen ausgeführt. Des Weiteren erlaubt die Benutzeroberfläche das Laden und Speichern vollständiger Konfigurationen. Auch einzelne Teile der Simulation und des Zeppelins können seperat gespeichert werden. Somit können einzelne Bauteile oder Zeppeline einmal erstellt werden und im Folgenden auf verschiedenen Routen und zu verschiedenen Zeiten simuliert werden. Für die Erstellung der Benutzeroberfläche wurde das Anwendungsframework `Qt` in der Version `5.13.1` genutzt, da dieses unter anderem plattformübergreifende Kompatibilität bereitstellt. Ein Klassendiagramm findet sich in Abschnitt 10.3.

4.1 Aufbau der Oberfläche

Die zugrundeliegende Simulation ist so gestaltet, dass nahezu alle Parameter dynamisch geändert werden können. Dadurch ergibt sich naturgemäß eine sehr große Anzahl an einstellbaren Werten. Um die Übersichtlichkeit der Anwendung zu wahren wurde das Hauptfenster in unterschiedliche Reiter unterteilt. Diese bündeln jeweils einen Aspekt der Simulation (zum Beispiel Größe und Antrieb des Zeppelins oder Routeninformationen). Beim Start der Anwendung sind in den für eine sinnvolle Simulation notwendigen Feldern Standardwerte eingetragen. Diese orientieren sich am bisher größten gebauten Luftschiff, dem Zeppelin LZ-129, besser bekannt als "Hindenburg"[2]. Aus brandschutztechnischen Gründen wurde das Traggas Wasserstoff durch Helium ersetzt, wodurch sich, aufgrund der höheren Dichte von Helium, eine geringere Nutzlast ergibt. Außerdem wurde das Leergewicht etwas reduziert, um modernere Werkstoffe zu simulieren. Die Länge von 246 m sowie der Durchmesser von 41 m wurden übernommen.

4.1.1 Zeppelin

Auf diesem ersten Reiter (siehe Abbildung 1) können allgemeine Daten zu dem zu simulierenden Luftschiff eingegeben werden. Neben Daten wie Leergewicht und Volumen für die Traggaskammer kann hier auch eine gewünschte Nutzlast bestimmt werden. Das Traggas kann nicht direkt angegeben werden, um die Möglichkeit von Gasgemischen mit anderer Dichte offen zu halten. Allerdings werden in einem Tooltip die Dichten der beiden Gase Wasserstoff und Helium angezeigt, welche die geringste Dichte aller Gase aufweisen. Damit können die Werte dieser beiden Gase ohne weitere Recherche eingegeben werden.
Während die meisten eingegeben Daten auf dieser Seite schon in der Nutzeroberfläche vorverarbeitet werden (siehe Kapitel 4.2), werden die technischen Daten der Motoren direkt an die Simulation übergeben, in welcher sie bei der Geschwindigkeitsberechnung zum Tragen kommen.
Bei der Festlegung der Daten für die Solarzellen kann ausgewählt werden, ob diese nur die obere Hälfte des Zeppelins oder die gesamte Außenhaut bedecken sollen. Mit den Eigenschaften heutiger Dünnschichtsolarzellen, die eine ähnliche Effizienz wie traditionelle Solarzellen aufweisen, von $\sim$2 $\frac{kg}{m^2}$[14] bringt die Abdeckung der gesamten Oberfläche große Einbußen in der verfügbaren Nutzlast und nur einen kleinen Geschwindigkeitsvorteil mit sich. Dies kann sich jedoch durch weitere Effizienzsteigerung und Gewichtsreduktion in Zukunft ändern. Schon verfügbare Solarfolien besitzen bei einem Gewicht von nur noch 220-450 $\frac{g}{m^2}$ (Stand 2019) nur eine Effizienz von $\sim$3 %[8]. Experimentelle Solarzellen erreichen Extremwerte von 3,6 $\frac{g}{m^2}$, jedoch ist diese Technik noch nicht für große Projekte geeignet[11]. Um die Benutzeroberfläche zukunftssicher zu halten stehen daher beide Optionen zur Auswahl. Mit einem Klick auf „To Route Parameter“ werden die eingegebenen Daten übernommen.

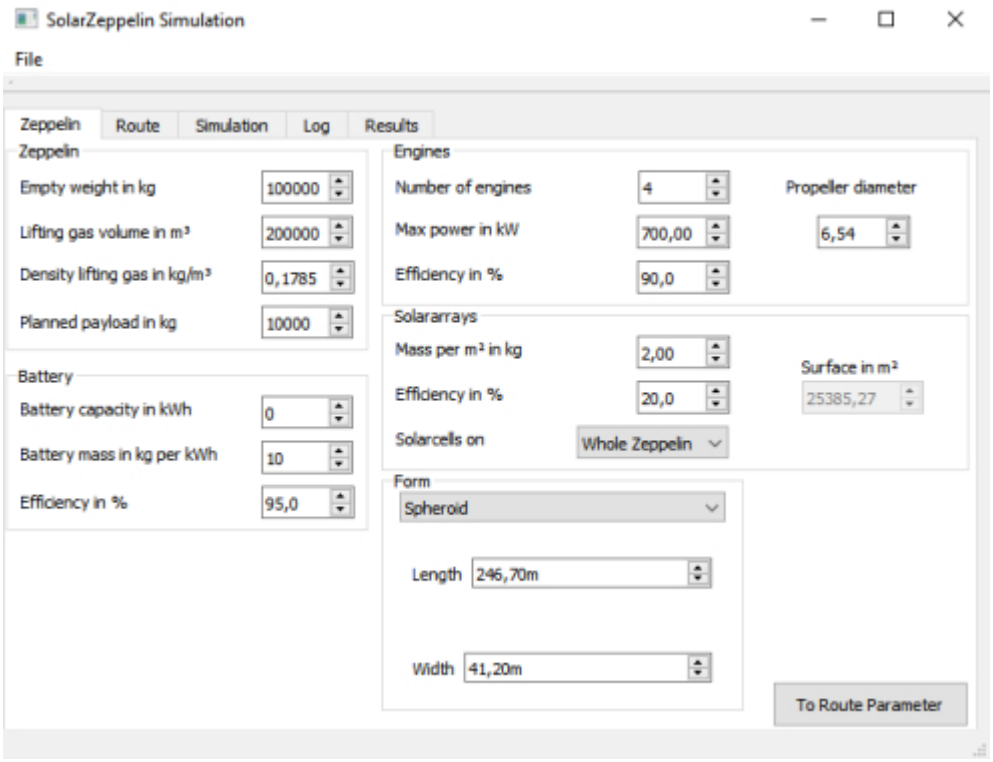

Abbildung 1: UI: Eingabe der Zeppelindaten

4.1.2 Route

Im zweiten Reiter werden die für die Simulation notwendigen Informationen über die Flugroute und Flugzeit abgefragt, sowie erste Ergebnisse aufgrund der eingegebenen Daten für den Zeppelin angezeigt. Auf die Berechnung dieser Werte wird in Kapitel 4.2 näher eingegangen. Als Standardwerte für die zu fliegende Route sind hier die Koordinaten für die Strecke von Nürnberg nach New York eingegeben. Um die Route aucg unkompliziert in umgekehrter Richtung simulieren zu können, können Start und Ziel per Knopfdruck getauscht werden, was Eingabefehler bei einer Neueingabe der geänderten Koordinaten verhindern soll.
In diesem Reiter findet auch die Eingabe des Starttages und der Startuhrzeit statt. Die Wahl von unterschiedlichen Uhrzeiten an einem Tag hat auf längeren Strecken keine maßgebliche Auswirkung auf die Flugzeit, wie in Kapitel 6.4 gezeigt wird. Die Wahl des Startdatums hat jedoch je nach gewählter Route einen großen Einfluss auf die erreichte Flugzeit, siehe hierzu auch Kapitel 6.3. Dabei ist allerdings zu beachten, dass die Änderung des Starttages keine Änderung der genutzten Winddaten bewirkt. Diese müssen vom Benutzer selbstständig geändert werden.
Die schematische Anzeige des gewählten Formfaktors auf dieser Seite dient dem Ausschluss von Fehlern, die möglicherweise bei der Dateneingabe auf der vorigen Seite entstanden sind.

4.1.3 Simulation

Im Reiter „Simulation“ erfolgt nun die Auswahl des Algorithmus, mit dem die Flugzeit der gewählten Route simuliert werden soll. Da die Simulation von Christopher Newman erweitert wurde, besteht hier die Möglichkeit alternative Algorithmen neben dem Partikelschwarm Optimierungsalgorithmus (PSO) zu verwenden. Zur Auswahl stehen daher

- Dijkstra
- Brute Force
- PSO

- PSO - solar
- PSO - recursive
- PSO - recursive solar

Bei der Auswahl jedes Algorithmus werden Werte vorgegeben, welche sich für die Simulation als günstig erwiesen haben oder aufgrund der Implementierung notwendig sind. Damit nur die für die gewählte Simulationsart benötigten Parameter eingegeben werden müssen, werden alle anderen Eingabefelder deaktiviert.
Die ersten beiden Algorithmen dienen als Vergleichsobjekte für den Partikelschwarm Optimierungsalgorithmus ohne Sonneneinfluss. Der implementierte Dijkstra nutzt ein 25 Punkte stencil, wie in der Arbeit von Christopher Newman beschrieben[13]. Aufgrund der Implementierung muss hier die Pfadlänge genau 2 betragen.
Auch der PSO in der Auswahl entspricht bis auf einzelne Optimierungen dem Algorithmus aus Newmans Arbeit [13]. Die hier vorgegebenen Werte ermöglichen eine möglichst schnelle Konvergenz bei moderater Rechenzeit. Auch wird hier eine konstante Geschwindigkeit von 104 $\frac{km}{h}$ und eine Windauflösung von 30 km genutzt.
Der „PSO - solar“ genannte Algorithmus basiert auf dem Partikelschwarm Optimierungsalgorithmus, nutzt jedoch eine, abhängig vom Sonnenstand, errechnete Geschwindigkeit im Verlauf des Fluges. Damit das Luftschiff auch während der Nachtstunden, ohne Energie von den Solarzellen, manövrierbar bleibt, muss hier nun eine Mindestgeschwindigkeit angegeben werden. Es wird vorausgesetzt, dass diese unabhängig von der gewonnenen Energie der Solarzellen erreicht wird. Standardmäßig sind hier 30 $\frac{km}{h}$ angegeben.
Anschließend kann die Simulation entweder direkt aus der Benutzeroberfläche gestartet oder die gewählte Konfiguration in Form einer speziellen XML-Datei gespeichert werden, welche der Simulation mittels Kommandozeilenparameter `-xml` übergeben werden kann. Wird die Simulation von hier aus gestartet, wird in Hintergrund eine XML-Datei mit den benötigten Daten erstellt und direkt der Simulation übergeben. Diese wird dann in einem unabhängigen Thread gestartet. Daraufhin wird automatisch der Reiter gewechselt, um den Fortschritt der laufenden Simulation anzeigen zu können.

4.1.4 Log

Während der Laufzeit der Simulation werden hier die Ausgaben der Simulation angezeigt, um den Benutzer über deren Fortschritt zu informieren. Dazu wird die Ausgabe der Simulation in ein Logfile geschrieben, welches ausgelesen und in der UI angezeigt wird. Durch den Umweg über ein gesondertes Logfile können die Ergebnisse einzelner Simulationsdurchläufe leicht gespeichert werden.
Die Laufzeit der Simulation kann, falls 'PSO' oder 'PSO - solar' gewählt wurde an einem Fortschrittsbalken abgeschätzt werden. Dieser zeigt prozentual an, in welcher Iteration der letzte schnellste Pfad gefunden wurde.

4.1.5 Results

Nach erfolgreichem Abschluss der Simulation werden die Werte im „Results“ Reiter aktualisiert, welcher die Flugzeit und kürzeste mögliche Distanz sowohl in Kilometern als auch in, in der Luftfahrt üblichen, Nautischen Meilen anzeigt. Zusätzlich wird noch einmal die Laufzeit der Simulation ausgegeben.

4.1.6 Settings

Während im Normalfall davon ausgegangen wird, dass sich die Simulation, das Konfigurations XML-File, das Logfile sowie das Verzeichnis mit den aufbereiteten Winddaten im gleichen Verzeichnis wie die Anwendung befinden, können diese durchaus an verschiedenen Speicherorten verteilt gespeichert sein. In der Anwendung kann dazu unter File>Settings für jede Datei der Speicherort angegeben werden, sodass die Dateien nicht kopiert oder verschoben werden müssen und dadurch das Risiko von unterschiedlichen Versionen einer Datei verringert wird.

Die zu Beginn standardmäßig angegebenen Pfade sind relativ zum Ausführungsverzeichnis angelegt. Beim ersten Start wird eine Sicherungsdatei angelegt, sodass geänderte Pfade auch über den Neustart der Benutzeroberfläche erhalten bleiben. Diese Sicherungsdatei wird bei jedem Neustart eingelesen und sobald Pfade geändert wurden aktualisiert.

4.2 Einfache Berechnungen in der Oberfläche

Nach der Eingabe der technischen Daten des Zeppelins werden diese auf einfache Bedingungen überprüft. Hierbei wird vereinfachend angenommen, dass der Auftriebskörper vollständig mit Traggas gefüllt ist, ein eventuell vorhandenes Ballonett dementsprechend leer ist. Sollte in dieser Konfiguration der Auftrieb (bei 15 ∘C und auf Meereshöhe) kleiner sein als die Gewichtskraft, wird eine Warnmeldung angezeigt. Weiterhin werden aus den eingegebenen Zeppelindaten erste weitere Daten berechnet. Die hier getätigten Berechnungen sind nicht sehr rechenintensiv, um Verzögerungen in der Benutzeroberfläche zu vermeiden.

4.2.1 Auftrieb und Nutzlast

Die Ergebnisliste beginnt mit dem Gesamtauftrieb des Zeppelins. Die zugrundeliegende Berechnung hier ist

$$M_{uplift} = M_{air} - M_{airship}$$

wobei M_{air} der durch den Auftriebskörper verdrängten Luftmasse und $M_{airship}$ der Masse des Luftschiffes entspricht. Als Dichte der Luft wird hierbei 1,225 $\frac{kg}{m^3}$ angenommen, was einer Temperatur von 15 °C auf Meereshöhe unter Normdruck von 1013,25 hPa entspricht[19]. Darauf folgt die Anzeige der maximal möglichen Nutzlast des Zeppelins, wobei das Gewicht der aufgebrachten Solarzellen bereits berücksichtigt ist. Es ist jedoch zu beachten, dass ein Zeppelin mit der maximalen Nutzlast nicht mehr ohne Motorhilfe den Boden verlassen könnte, da sich Auftriebs- und Gewichtskraft nur auf Meereshöhe und bei Normluftdruck sowie 15 °C Außentemperatur aufheben. Sobald das Luftschiff steigt, verringert sich die Dichte der umgebenden Luft, wodurch die Gewichtskraft die Auftriebskraft übersteigen und das Luftschiff zu sinken beginnen würde. Hierbei wird auf den eingegebenen Wert des Volumens des Luftschiffs zurückgegriffen, da dort Strukturelemente berücksichtigt werden sollten, welche das nutzbare Volumen für das Traggas verringern.

4.2.2 Maximale Flughöhe

Für eine genaue Berechnung der Höhe ist eine genaue Kenntnis der Luftdichte vorauszusetzen. Da die Luftdichte jedoch von diversen Parametern wie Temperatur, Luftfeuchtigkeit, Höhe und Hochbeziehungsweise Tiefdruckeinfluss anhängig ist, steht der Aufwand einer exakten Berechnung in einem schlechten Verhältnis zum Nutzen der genaueren Werte. Daher wird in dieser Arbeit auf die ICAO-Standardatmosphäre (ISA) zurückgegriffen, welche von der Internationalen Zivilenluftfahrtorganisation (ICAO) definiert wurde[5]. Diese bildet ungefähr die in der Atmosphäre herrschenden Mittelwerte ab, wobei jedoch die Luftfeuchtigkeit immer mit 0 % angenommen wird. Weiterhin herrschen auf Meereshöhe 15 °C und 1013,25 hPa, sowie daraus resultierend eine Luftdichte von 1,225 $\frac{kg}{m^3}$. Bis zur Höhe der Tropopause in 11 km verhält sich hier die Temperatur in der ISA linear mit einer Abnahme von $\frac{6{,}5K}{1000m}$.
Daher wird als nächstes Ergebnis die maximal erreichbare Höhe in der internationalen Standardatmosphäre unter Berücksichtigung der vorher eingegebenen Nutzlast berechnet. Hierzu wird das Gesamtgewicht des Zeppelins durch die Addition der einzelnen Komponenten ermittelt. Zusammen mit dem eingegebenen Volumen wir eine durchschnittliche Dichte des Zeppelins ρ_{zep} errechnet. Nun wird unter der Verwendung der Formel für die Dichtebestimmung der Luft[6]

$$\rho_{zep} = \frac{p(h)}{R_s \cdot T(h)}$$

die Höhe ermittelt, in welcher die umgebende Luft die gleiche Dichte wie das Luftschiff aufweist. Dabei bezeichnen

$$T(h) = 288,15K - 0,0065\frac{K}{m} \cdot h \quad und$$

den Temperaturverlauf mit zunehmender Höhe und

$$p(h) = 1013,25hPa \cdot \left(1 - \left(\frac{0,0065\frac{K}{m} \cdot h}{288,15K}\right)^{5,255}\right)$$

die barometrische Höhenformel[9]. R_s bezeichnet die Gaskonstante der Luft in der ISA. Da die Luftfeuchtigkeit dort 0% beträgt, gilt $R_s = 287,058\frac{J}{kg \cdot K}$. Für die Nutzung in der Benutzeroberfläche wurde der Druck nur näherungsweise berechnet, daher wird statt 5,255 im Exponenten 5 angenommen. Dann ergibt sich die Höhe abhängig von einer gegebenen Luftdichte d durch

$$h = 44330,8 - 42137,9 \cdot \sqrt[4]{d} \tag{1}$$

Dies entspricht der maximal (ohne zusätzlichen motorisierten Auftrieb) erreichbaren Höhe für diese Konfiguration. Der Wert ist in der Benutzeroberfläche fettgedruckt, da er direkt für die Simulationsparameter übernommen wird.

4.2.3 Kenndaten des Zeppelins

Weiterhin wird nun die Gesamtoberfläche in Abhängigkeit der gewählten Form und Dimensionen errechnet. Diese ist unabhängig von der gewählten Anordnung der Solarzellen.
Im nächsten Feld wird das voraussichtliche Gesamtgewicht der Solarzellen ausgegeben. Dieses berechnet sich aus dem Gewicht der Solarzellen pro m^2, der gewählten Form und Größe des Zeppelins und der Anordnung der Solarzellen. Hierbei ist zu beachten, dass noch keine weitere benötigte Infrastruktur für die Solarzellen berücksichtigt wird. Es wird davon ausgegangen, dass die notwendigen Verkabelungen und Wechselrichter bereits im Leergewicht des Luftschiffs enthalten sind.
Im letzten Feld wird nun noch das aufgrund der Form und Maße des Zeppelins errechnete Volumen der Traggaskammer ausgegeben. Dies dient als Kontrollmechanismus für die Eingabe des Traggasvolumens auf der vorigen Seite. Daher wird ab einem Unterschied von 25 % eine Warnmeldung angezeigt, die jedoch keinen weiteren Einfluss auf die Simulation hat und nur als Hinweis dient.

4.3 Speichern und Laden

Um von überall im Programm auf die eingegebenen bzw. geladenen Werte zugreifen zu können wurde die Klasse `parameter` erstellt. In dieser werden während der Laufzeit alle Werte mit Hilfe eines Vektors verwaltet. Dabei wird jeder Wert zusammen mit einem Schlüssel (Bezeichnung des Wertes) als Schlüssel-Wert-Paar im Vektor abgelegt. Da die hier übergebenen Schlüssel gleichzeitig als Bezeichnung der Werte in den generierten XML-Dateien dienen, wurde darauf geachtet, möglichst selbsbeschreibende Schlüsselnamen zu verwenden. Die Benutzung eines Vektors als Struktur bietet den großen Vorteil einer Erweiterbarkeit in auf die Simulation aufbauenden Forschungsarbeiten durch weitere Studierende. Für einen einfachen Zugriff auf die gespeicherten Werte wurden zudem Methoden erstellt, über welche (auch ohne Kenntnis der zugrunde liegenden Struktur) Werte aktualisiert, gespeichert und geladen werden können.
In der Klasse `parameter` wurde das Programmiermuster des Singleton umgesetzt. Dieses Muster bietet den Vorteil, dass es keine zwei Objekte von `parameter` geben kann, wodurch asynchrone Werte in unterschiedlichen Bereichen der Software verhindert werden.
Der Methodenaufruf von `parameter::getInstance()` liefert dementsprechend die aktuelle und einzige Instanz von `parameter` zurück. Beim ersten Aufruf wird entsprechend des Entwurfsmusters eine neue Instanz erzeugt. Neue beziehungsweise geänderte Schlüssel-Wert-Paare werden mit Hilfe der Methode `addTuple(String, value)` an das Parameter-Singleton übergeben.
Neben der Möglichkeit, alle eingegeben Daten in einer Datei abzuspeichern, wird auch die Option angeboten, lediglich einzelne Teile zu speichern und zu laden. Das Speichern einzelner Komponenten

ist über File>Save parts configuration möglich. Dabei werden nur die zu dieser Komponente gehörenden Daten in die erzeugte Speicherdatei geschrieben. Wird eine so erzeugte Datei geladen, werden lediglich die dort gespeicherten Werte in der Oberfläche aktualisiert. Alle anderen Datenfelder behalten ihre Werte und werden nicht auf den Standard zurückgesetzt. Somit besteht beispielsweise die Möglichkeit einen definierten Zeppelin auf einer festgelegten Route mit unterschiedlichen Solarzellen zu simulieren, wobei nach jedem Simulationsdurchlauf nur neue Daten für die Solarzellen geladen werden müssen.

4.4 Kommunikation mit der Simulation

Ein Ziel während der Implementierung der Benutzeroberfläche war, die eigentliche Simulation stets unabhängig zu halten. Deshalb wurde als Schnittstelle zwischen den beiden Programmen ein XML-Dokument gewählt. Dieses besitzt den Vorteil, dass es programmtechnisch sowohl einfach geschrieben als auch gelesen werden kann und gleichzeitig auch noch menschenlesbar bleibt. Nachdem alle Einstellungen und Simulationsparameter in der Benutzeroberfläche (UI) getätigt wurden, wird ein spezielles XML-Dokument erstellt, welches alle notwendigen Parameter enthält. Der Speicherpfad wird der Simulation zusammen mit dem Parameter `-xml` übergeben. Durch dieses Verfahren können aufwendige Simulationen auch ohne die Benutzeroberfläche gestartet werden. Es ist zudem möglich, im Voraus verschiedene Konfigurationen anzulegen und diese nacheinander ohne zusätzliches Eingreifen abzuarbeiten, indem die Simulation mit den jeweils unterschiedlichen XML-Dateien als Parameter gestartet wird. Die Ergebnisse werden dann auf der Konsole ausgegeben.
Wird die Simulation direkt aus der UI gestartet, werden der Fortschritt und neueste schnellste Pfade direkt in der Oberfläche angezeigt.

5 Erweiterung des Partikelschwarm Optimierungsalgorithmus von Christopher Newman

Als Grundlage dieser Arbeit diente das Simulationsprogramm aus Christopher Newmans Bachelorarbeit „Optimal Control of a solar Airship“[13], das im Rahmen der Forschung weiterentwickelt wurde. Im Folgenden wird kurz auf den zugrundeliegenden Algorithmus sowie die Änderungen und Erweiterungen dieses eingegangen.

5.1 Der Partikelschwarm Optimierungsalgorithmus

Der Partikelschwarm Optimierungsalgorithmus ist ein Algorithmus, dessen Grundprinzip auf dem Schwarmverhalten von Tieren in der Natur basiert. Eine in diesem Anwendungsfall genutzte Eigenschaft dieses Algorithmus aus dem „Organic Computing“ ist die der Selbstoptimierung. Die einzelnen Partikel in diesem Algorithmus speichern jeweils das eigene lokale Maximum P_{loc}, gleichzeitig wird das globale Maximum P_{glob} gespeichert. In jeder Iteration bewegt sich jeder Partikel unabhängig von den anderen Partikeln mit einer vom lokalen und globalen Maximum beeinflussten Geschwindigkeit. Diese ergibt sich aus

$$V_i = V_{i-1} \cdot \omega + r_1 \cdot c_1 \cdot (P_{loc} - X_i) + r_2 \cdot c_2 \cdot (P_{glob} - X_i) \tag{2}$$

wobei V_{i-1} die vorige Geschwindigkeit dieses Partikels, c_i Skalierungsfaktoren für die beiden Maxima, r_i Zufallszahlen und X die alte Position sind. Die neue Position ergibt sich mit

$$X_i = X_{i-1} + V_i \tag{3}$$

Der Faktor ω, welcher den Einfluss der vorigen Geschwindigkeit bestimmt, beträgt zu Beginn der Simulation 1 und wird im weiteren Verlauf auf 0,95 gesenkt[13].
Als Partikel dienen in dieser Implementierung einzelne Pfade vom Start- zum Zielpunkt. Diese bestehen aus einer festgelegten Anzahl von Wegpunkten, welche im Laufe des Fluges überflogen werden. Der PSO optimiert die Partikel durch die Verschiebung dieser Wegpunkte unter der Zielsetzung, den schnellsten Weg vom Start zum Ziel zu finden.
Die Flugdauer eines Pfades wird in jeder Iteration anhand der Wind- und Sonneneinflüsse neu berechnet. Da eine kontinuierliche Berechnung zu aufwendig ist, wird jeder Pfad in Unterpfade unterteilt. Im Verlauf eines Unterpfades werden sowohl der Wind als auch die Sonneneinstrahlung als konstant angenommen. Die Länge eines Unterpfades ist standardmäßig festgelegt, kann jedoch durch die Übergabe einer XML-Datei angepasst werden.

5.2 Werteänderung ohne Recompiling

Ein Nachteil der ursprünglichen Version der Simulation ist, dass neue Werte für den Algorithmus nur in der Datei `Parameters.h` als Änderung von Konstanten umgesetzt werden können. Zu diesen Werten gehören unter anderem die Anzahl der Iterationen des PSO, die Anzahl der Partikel und die Anzahl der Wegpunkte auf einem Pfad. Nach jeder Änderung muss somit das Programm neu kompiliert werden. Dies ist insbesondere von Nachteil, wenn viele Simulationen mit verschiedenen Werten durchgeführt werden sollen. Weiterhin sind die Daten für den Zeppelin nicht variabel. Dieser hat in der vorherigen Version immer eine maximale Flughöhe von 2700 m und eine konstante Geschwindigkeit von 104 $\frac{km}{h}$.
Um Daten zur Laufzeit dynamisch aus einem bereitgestellten XML-Dokument lesen zu können ist diese Implementierung folglich nicht geeignet. Daher wurde die Klasse `Parameters` zu einer Singleton-Klasse umgeschrieben. Diese ermöglicht es, Werte zu speichern und unkompliziert an alle anderen Klassen auszuliefern. Das Singleton-Entwurfsmuster sorgt weiterhin dafür, dass Änderungen an Werten sofort in anderen Klassen sichtbar sind und somit immer auf einer konsistenten Datengrundlage gearbeitet werden kann.
Um die Werte aus dem übergebenen XML-Dokument zu lesen wurde die Bibliothek TinyXML-2[17] eingebunden. Mit Hilfe dieser Bibliothek kann die übergebene Datei in der Simulation wie ein Objekt behandelt werden. Um die Übersichtlichkeit des Codes zu gewährleisten werden alle Werte

zu Beginn der Hauptmethode eingelesen und mit den entsprechenden Methoden der `Parameters`-Klasse eingepflegt.
Sollte die Simulation ohne eine zugehörige XML-Datei gestartet werden, werden vordefinierte Standardwerte genutzt, so dass auch ohne XML-Datei die Route von London nach New York simuliert werden kann.

5.3 Berechnung der Solarenergie

Die Berechnung der gesamten Solarenergie des Luftschiffes setzt sich aus mehreren Abschnitten zusammen, welche der Übersichtlichkeit halber in unterschiedlichen Methoden realisiert wurden. Dazu gehört die Bestimmung der Sonnenposition und aufbauend darauf die Bestimmung der Abschwächung der Sonnenenergie durch die Atmosphäre, sowie die Berechnung der auf den Zeppelin treffenden Energie.

5.3.1 Berechnung der Sonnenposition

Die Geschwindigkeit eines solarbetriebenen Zeppelins hängt hauptsächlich von der Menge an Sonnenenergie ab, die auf den Zeppelin trifft. Um diese korrekt zu berechnen ist es unerlässlich, die genaue Position der Sonne am Himmel bestimmen zu können. Diese Bestimmung wird unter Zuhilfenahme der Bibliothek SolTrack, welche von Marc van der Sluys entwickelt wurde, vorgenommen[15]. Die Position der Sonne kann unter Verwendung von zwei Winkeln, Altitute und Azimuth, angegeben werden. Dabei ist der Altitude-Winkel definiert als der Winkel über dem Horizont. Somit beträgt der Wert bei Sonnenauf- und untergang 0° und wenn die Sonne im Zenit steht 90°. Mit Hilfe dieses Winkels kann berechnet werden, welche Entfernung das Sonnenlicht durch die Atmosphäre zurücklegen muss und wie stark die Energie aufgrund des schrägen Winkels verteilt wird. Der Azimuthwinkel gibt hingegen an, aus welcher Richtung die Sonnenstrahlen einfallen. Dieser Winkel könnte bei einem kugelförmigen Luftschiff vernachlässigt werden, bei einem Luftschiff in Form eines Rotationsellipsoiden ergeben sich jedoch bedeutende Unterschiede, je nachdem aus welcher Richtung das Sonnenlicht auf den Zeppelin trifft. Der Wert des Azimut-Winkels liegt zwischen 0° und 360°, wobei 0° eine aus dem Norden und 90° eine aus dem Osten scheinende Sonne repräsentiert.
Für den Aufruf der SolTrack Bibliothek sind sowohl die Koordinaten des Punktes, für den die Sonnenposition bestimmt werden soll, als auch das genaue Datum und die Uhrzeit nötig. Dieser Aufruf wird mit `Sun::getSunPosition(coordinates, UTC)` so gekapselt, dass die in der Simulation genutzten Zeit- und Koordinatenformate verwendet werden können.

5.3.2 Berechnung der Abschwächung durch die Atmosphäre

Während das Sonnenlicht und die damit transportierte Energie auf dem Weg durch das Weltall nur unbedeutend gestreut und damit abgeschwächt wird, hat die Menge der zu durchquerenden Luft einen erheblichen Einfluss auf die finale Energieausbeute. So beträgt die Solarkonstante, also die Leistung der Sonne pro Quadratmeter außerhalb der Atmosphäre, 1367 $\frac{W}{m^2}$ [7].
Von diesen 1367 $\frac{W}{m^2}$ werden 6 % direkt beim Auftreffen auf die Atmosphäre in den Weltraum refklektiert, somit verbleiben 1285 $\frac{W}{m^2}$, welche nun weiter durch die Atmosphäre abgeschwächt werden[12].
Die Entfernung, die das Sonnenlicht durch die Erdatmosphäre zurücklegen muss, wird in Air Masses angegeben. Dabei bezeichnet eine Air Mass die Enfernung, die das Sonnenlicht einer im Zenit stehenden Sonne durchqueren muss. Die Berechnung der durchquerten Air Masses mit

$$am = \frac{1}{sinh}$$

wobei h der Höhe der Sonne über dem Horizont entspricht, wäre nur bei einer flachen Erde möglich. Stattdessen wird das 1994 von Andrew T. Young entwickelte Modell verwendet. Dieses berücksichtigt sowohl die Erdkrümmung als auch die Brechung des Lichtes in der Atmosphäre[20]. Daher wird in der Simulation für die Berechnung der Air Masses

$$am = \frac{1.003198 \cdot cosz + 0.101632}{cos^2 z + 0.090560 \cdot cosz + 0.003198} \tag{4}$$

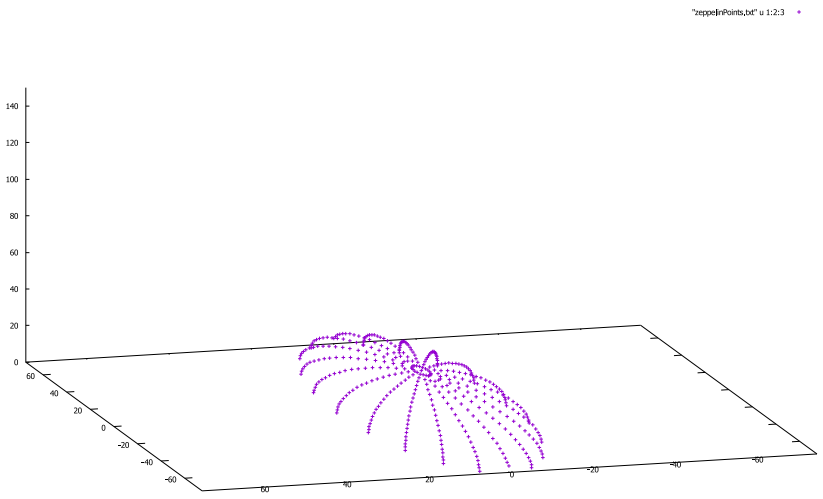

Abbildung 2: Punkte auf der Zeppelinhülle

genutzt, wobei der Zenitwinkel der Sonne mittels $z = 90 - h$ ermittelt wird.
Steht die Sonne im Zenit, werden bis zur Erdoberfläche ∼20 % der solaren Energie von der Atmosphäre absorbiert. Da in der Simulation eine homogen beschaffene Atmosphäre angenommen wird, kann die verbleibende Energie nach der Durchquerung der Atmosphäre mit

$$E_{solar} = 1285 \frac{W}{m^2} \cdot 0,8^{am} \tag{5}$$

berechnet werden. E_{solar} bezeichnet somit, ohne Berücksichtigung von Wolken, die Energie, die auf eine zur Sonne ausgerichtete Fläche trifft.

5.3.3 Diskretisierung der Zeppelinhülle

In der Simulation wird die Form des Luftschiffes als Rotationsellipsoid angenommen. Daher kann die der Sonne zugewandte Fläche nicht trivial über die Flächenberechnung des geworfenen Schattens erfolgen. Dies bietet sich unter anderem bei kugelförmigen Objekten an, da die bestrahlte Fläche dort die Größe eines Kreises mit dem gleichen Radius wie die betrachtete Kugel hat.
Eine Aufteilung der Oberfläche in kleinere Einzelflächen bietet für die weitere Forschung daher eine gute Ausgangslage. Dafür werden wie in der Arbeit von Kai Teichmann gleichmäßig Punkte auf der Zeppelinoberfläche verteilt. Hierbei wirkt sich die Anzahl der platzierten Punkte und damit die Anzahl der berechneten Flächen stark auf die Genauigkeit aus. In der vorliegenden Simulation wird eine Diskretisierung von 70 Schritten angewendet, wodurch sich für die Größe der Oberfläche ein Fehler von <0,1 % bei einer Gesamtpunktzahl von $70^2 + 1 = 4901$ ergibt.
Bei der Diskretisierung wird das Luftschiff mit seinem Mittelpunkt auf dem Ursprung eines dreidimensionalen Koordinatensystems platziert, die Flugrichtung wird entlang der x-Achse angenommen. Dann ergeben sich die Punkte auf der Oberseite des Luftschiffes wie folgt:

$$\begin{aligned} x &= \frac{l}{2} \cdot cos\theta \cdot cos\phi \\ y &= \frac{w}{2} \cdot cos\theta \cdot sin\phi \\ z &= \frac{h}{2} \cdot sin\theta \end{aligned}$$

mit $-\pi \leq \phi \leq \pi$ und $0 \leq \theta \leq \frac{\pi}{2}$ [16]. l, w und h bezeichnen hierbei die Länge, Breite und Höhe des Luftschiffes. Zusätzlich wird noch ein Punkt mit den Koordinaten $(0,0,\frac{h}{2})$ gespeichert.

Zwischen drei nebeneinander liegenden Punkten wird nun ein Dreieck aufgespannt. Dreiecke bieten hier im Vergleich zu Trapezen den Vorteil, dass drei (voneinander verschiedene) Punkte immer eine eindeutige Ebene aufspannen, was bei vier Punkten in einem dreidimensionalen Raum nicht gegeben ist. Hierbei ergibt sich bei N Diskretisierungsschritten eine Anzahl von N Kreissektoren, wobei jeder Kreissektor in N-1 Trapeze unterteilt wird. Jedes Trapez wird nun noch in zwei Dreiecke unterteilt. Zusätzlich mit den N an den Punkt $(0,0,\frac{h}{2})$ angeschlossenen Dreiecken ergibt sich somit eine Anzahl von

$$2(N(N-1)) + N = 2N^2 - N \tag{6}$$

Flächen. Bei 70 Punkten beläuft sich die Anzahl der Flächen auf der oberen Hälfte des Rotationsellipsoids folglich auf 9730.
Für jede dieser Flächen wird nun der Flächeninhalt über die Ortsvektoren der Eckpunkte A, B und C mit

$$\vec{x} = \begin{pmatrix} b_1 - a_1 \\ b_2 - a_2 \\ b_3 - a_3 \end{pmatrix} \qquad \vec{y} = \begin{pmatrix} c_1 - a_1 \\ c_2 - a_2 \\ c_3 - a_3 \end{pmatrix} \tag{7}$$

$$A = \frac{1}{2} \cdot |\vec{x} \times \vec{y}| \tag{8}$$

ermittelt. Aufbauend darauf wird mit

$$\vec{n} = (\vec{x} \times \vec{y}) \cdot \frac{1}{|\vec{x} \times \vec{y}|} \tag{9}$$

ein nach außen zeigender und normalisierter Normalenvektor ermittelt und gespeichert. Soll hingegen ein vollständig mit Solarmodulen bestückter Zeppelin simuliert werden, werden $4N^2 - 2N$ Flächen angelegt. Diese Flächen übernehmen die Werte der Zellen mit der äquivalenten Position an der Oberseite des Zeppelins. Um die Spiegelung an der xy-Ebene durchzuführen, wird der schon vorhandene Normalenvektor n_{up} mit

$$\vec{n}_{down} = \vec{n}_{up} \cdot \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix} \tag{10}$$

an der xy-Ebene gespiegelt.

5.3.4 Energieberechnung

Während der Simulation einer Flugstrecke wird jeweils nach der Zurücklegung einer definierten Teilstrecke der Einfluss der Sonne und des Windes auf das Luftschiff berechnet. Da der Energieertrag der Solarzellen primär vom Ort des Zeppelins und der Zeit an diesem Ort abhängt, kann er nicht vorab berechnet werden. Bei einem Zeppelin in Form eines Rotationsellipsoids wird die Energie für jede der in Kapitel 5.3.3 definierten Flächen einzeln berechnet. Durch die Berechnungen aus Kapitel 5.3.1 ist die Höhe der Sonne über dem Horizont und der Azimutwinkel bekannt. Die Flugrichtung des Zeppelins wird durch die Methode `nautic::rhumbBearingTo(coordinates, coordinates)`, welche schon in der Simulation von Christopher Newman implementiert wurde, bestimmt [13]. Statt alle Flächen des Zeppelins zu rotieren, wird die Flugrichtung mit dem Azimutwinkel der Sonne verrechnet, wodurch sich ein neuer Winkel relativ zum Zeppelin ergibt.
Aus dem neu berechneten Azimuth-(θ) und dem bekannten Altitudewinkel(ϕ) der Sonne wird durch

$$\vec{s} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \cdot \begin{pmatrix} cos\phi & 0 & sin\phi \\ 0 & 1 & 0 \\ -sin\phi & 0 & cos\phi \end{pmatrix} \cdot \begin{pmatrix} cos-\theta & -sin-\theta & 0 \\ sin-\theta & cos-\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{11}$$

ein Vektor $\vec{s}$ gebildet, welcher in die Richtung der Sonnenstrahlen gerichtet ist. Die zum Zeitpunkt der Berechnung auf den Zeppelin treffende Energie ergibt sich aus der Summe der auftreffenden Energie auf die N einzelnen Flächen mit der Oberfläche A. Dadurch, dass sowohl der Normalenvektor

jeder Fläche, als auch der Vektor, der dem Sonnenlicht entspricht, normalisiert sind, kann die Gesamtenergie, mit

$$\sum_{n=0}^{N} A_n \cdot (\vec{n}_n \times \vec{s}) \tag{12}$$

berechnet werden. Hierbei ist zu beachten, dass die ermittelte Energie nicht vollständig zur Verfügung steht, da Solarzellen einen Wirkungsgrad von <100 % besitzen. Dieser wird jedoch erst in Kapitel 5.4 berücksichtigt.

5.4 Geschwindigkeitsberechnung

Um die Geschwindigkeit des Zeppelins abhängig von der Sonneneinstrahlung zu berechnen, wird zunächst die zur Vortrieberzeugung verfügbare Energie ermittelt. Dazu wird die in Kapitel 5.3.4 berechnete Gesamtenergie mit dem Wirkungsgrad des Gesamtsystems des Zeppelins multipliziert. Als Standardwerte werden in der Simulation ein Wirkungsgrad der Solarzellen von 20 % und ein Wirkungsgrad der Motoren inklusive Leitungs- und Wandlungsverluste von 90 % angenommen. Daraus ergibt sich ein Gesamtwirkungsgrad von 18 %. Mittels diesem kann die für den Vortrieb verfügbare Leistung P_{ges} bestimmt werden.
In der Simulation wird keine Beschleunigung des Zeppelins berücksichtigt. Stattdessen wird an jedem zu berechnenden Punkt die maximal erreichbare Geschwindigkeit angenommen. Die Bestimmung der Geschwindigkeit erfolgt in der Methode `Parameters::calculateSpeed`, der als Parameter die gesamt auf den Zeppelin treffende Sonnenenergie sowie die aktuelle Flughöhe übergeben werden. Diese Geschwindigkeit ergibt sich aus dem Lösen von

$$\begin{aligned} F(v_z) = F_w(v_z) - F_p(v_z) = 0 \quad & mit \\ F_w(v_z) = \tfrac{1}{2} cw \cdot A_{zep} \cdot \rho \cdot v_z^2 \quad & und \\ F_p(v_z) = P_{ges} \cdot \tfrac{2}{v_l - v_z} & \end{aligned} \tag{13}$$

wobei F_w den Strömungswiderstand des Zeppelins [1] und F_p den Vortrieb durch die Motoren nach [16] bezeichnet. Allerdings wird nicht der Schub für jeden Motor einzeln berechnet, sondern der von allen Motoren zusammen erzeugte Schub. Dies setzt die Annahme voraus, dass alle Motoren baugleich sind und die Gesamtleistung gleichmäßig auf die vorhandenen Motoren verteilt wird. Weiterhin berechnet sich nach der Arbeit von Kai Teichman v_l durch

$$\begin{aligned} v_l &= \sqrt[3]{-\tfrac{q}{2} + \sqrt{d}} + \sqrt[3]{-\tfrac{q}{2} - \sqrt{d}} - \tfrac{v_z}{3} \quad mit \\ d &= v_z^6 \cdot \left(\tfrac{56}{729} + \tfrac{32}{27} \cdot \tfrac{4P_{ges}}{A_{ges} \cdot \rho \cdot v_z^3} + \left(\tfrac{P_{ges}}{2A_{ges} \cdot \rho \cdot v_z^3} \right)^2 \right) \\ q &= \begin{cases} \tfrac{P_{ges} \cdot 4}{A_{ges} \cdot \rho} & falls \quad v_z = 0 \\ -v_z^3 \cdot \left(\tfrac{16}{27} + \tfrac{P_p \cdot 4}{A_{ges} \cdot \rho \cdot v_z^3} \right) & sonst \end{cases} \end{aligned} \tag{14}$$

wobei A_p die gesamte Angriffsfläche aller Propeller ist. Diese Anpassung ist möglich, da in die Teilformeln jeweils das Verhältnis zwischen verfügbarer Energie und Angriffsfläche eingeht.
Zum Lösen von Gleichung (14) wird nun das Newtonverfahren angewendet. Daher ergibt sich die Geschwindigkeit durch

$$\begin{aligned} v_{n+1} := v_n - \tfrac{F(v_n)}{F'(v_n)} \quad & ; v_0 = 20 \tfrac{m}{s} \\ solange \quad & |v_{n+1} - v_n| > 0,1 \end{aligned} \tag{15}$$

Die so ermittelte Geschwindigkeit wird nun zur Berechnung der Flugzeit dieser Teilstrecke verwendet.

5.5 Analyse des besten Pfades

Aus Gründen der Speichereffizienz werden die Zwischenergebnisse der Berechnungen für die Geschwindigkeit auf einem Teilpfad verworfen, sobald eine Dauer für diesen errechnet wurde. Daher

wird jedes mal, wenn ein Partikel einen neuen globalen Bestwert gefunden hat, dieser Pfad erneut berechnet, um diesen analysieren zu können.
Im Zuge dieser Berechnung wird in der Datenstruktur, welche die einzelnen Solarflächen enthält, für jede Fläche die gesamt aufgenommene Energie gespeichert. Da aber nicht alle Flächen gleich groß sind, wird diese gleichzeitig noch auf 1 m^2 normiert.

5.6 Debugausgaben

Um die Übersichtlichkeit der Ausgabe trotz der zunehmenden Komplexität der Simulation und der damit verbundenen zahlreichen Ausgaben zu sichern, werden die einzelnen Ausgaben in Kategorien unterteilt. Dazu wurde in `debugutils.hpp` ein Makro definiert, welches jede Ausgabe auf die Konsole zusammen mit der Angabe eines Ausgabelevels kapselt. Das auszugebende Level kann im Code definiert werden.

6 Auswirkungen der Einbeziehung der Sonnenenergie

Damit die Ergebnisse der Simulationsdurchläufe mit vorigen Arbeiten und untereinander verglichen werden können, werden soweit möglich identische Zeppelindaten verwendet. Diese orientieren sich am LZ-129[2]. Als geplante Nutzlast wird in diesem Szenario ein 40'-ISO-Container[10] angenommen. Daher ergeben sich folgende Werte:

Länge	246,70 m
Durchmesser	41,20 m
c_w-Wert	0,04
Traggasvolumen	200 000 m^3
Traggas	Helium $\left(0,1785\frac{km}{m^3}\right)$
geplante Nutzlast	30,48 t
Anzahl der Motoren	4
max. Leistung der Motoren	700 kW
Propellerdurchmesser	6,54 m
Gewicht Solarzellen	2,0 $\frac{km}{m^2}$
Anbringung der Solarzellen	Nur obere Hälfte
Gesamteffizienz	18 %

Tabelle 1: Zeppelindaten in Anlehnung an LZ-129

6.1 Auswirkungen der Sonne und des Windes

In dieser ersten Simulationsreihe werden die Auswirkungen der Sonne und des Windes auf die Flugzeit und die Flugroute untersucht. Dazu wird die Simulation für die Route London - New York mit Wind- und Sonnendaten vom 20.03.2019 durchgeführt. An diesem Tag fand im Jahr 2019 die Tag-Nacht-Gleiche statt. Dieses Datum wurde gewählt, da die an diesem Tag zur Verfügung stehende Energie in etwa der im Jahresdurchschnitt verfügbaren Energie entspricht. Der Startzeitpunkt wurde auf 8 Uhr UTC gelegt, da so die Sonne des gesamten ersten Flugtages genutzt werden kann. Die Simulation wird unter Verwendung des PSO mit einer Pfadlänge von 5, also 4 Teilpfaden, durchgeführt. Pro Durchlauf werden 80 Partikel erzeugt, welche über 400 Iterationen optimiert werden. Ohne Berücksichtigung der Sonnenenergie wird eine konstante Geschwindigkeit von 104 $\frac{km}{h}$ angenommen. Unter Berücksichtigung der Sonnenenergie wird weiterhin eine Minimalgeschwindigkeit von 30 $\frac{km}{h}$ angenommen, welche zur Aufrechterhaltung der Manövrierfähigkeit benötigt wird.
Jede Simulation wird fünf Mal durchgeführt, um mögliche Abweichungen von der idealen Route durch die im PSO vorhandenen Zufallsfaktoren erkennen und berücksichtigen zu können.

Szenario	kürzeste Flugdauer	∅ Flugdauer	∅ Rechenzeit [1]
Kein Wind, konst. Geschwindigkeit	53,66 h	53,66 h	25 s
Kein Wind, var. Geschwindigkeit	86,35 h	86,38 h	328 s
Wind, konst. Geschwindigkeit	58,33 h	58,55 h	30 s
Wind, var. Geschwindigkeit	116,15 h	127,14 h	285 s

[1] auf AMD Ryzen 5 3600[3] Open-MP aktiv

Tabelle 2: Ergebnisse London - New York am 20.03.2019

Die kürzeste Flugzeit liefert in dieser Reihe die Konfiguration, welche ohne den Einfluss von Wind und mit einer konstanten Geschwindigkeit fliegt. Der Zeitvorteil gegenüber der Konfiguration mit Windeinfluss und konstanter Geschwindigkeit folgt daraus, dass über dem Atlantik auf der Nordhalbkugel hauptsächlich Westwinde vorherrschend sind und der Zeppelin daher oftmals gegen den Wind fliegen muss.
Unter Einbeziehung der verfügbaren, über die Solarzellen aufgenommenen Energie zeigt sich eine weitere Verschlechterung der Flugzeit. Diese ist damit zu erklären, dass die in der Arbeit von

Christopher Newman angenommene Durchschnittsgeschwindigkeit von 104 $\frac{km}{h}$ aus der genutzten Sonnenenergie vom 06.06.2018 berechnet wurden. In diesem ersten Test soll jedoch mit der im Jahresdurchschnitt verfügbaren Solarenergie gearbeitet werden, um einen Vergleichswert für weitere Tests zu erhalten.
Weiterhin zeigt sich, dass sich die Ergebnisse der Simulation ohne Wind- oder Sonneneinfluss pro Durchlauf nur vernachlässigbar, bei den vorliegenden Durchläufen um <1 s unterscheiden. Sobald der erste Freiheitsgrad, entweder die verfügbare Solarenergie oder der Wind, auf die Route Einfluss nimmt, steigt die Differenz zwischen den Durchläufen auf 3 bzw. 18 Minuten. Durch die Einbeziehung von beiden zum jetzigen Zeitpunkt simulierbaren Umwelteinflüssen ergibt sich ein Unterschied zwischen den schnellsten Routen der einzelnen Durchläufe von 33 Stunden. Aus diesem Grund sollten bei der Simulation einer Route unter dem Einfluss von Wind und Sonne immer mehrere Durchläufe ausgeführt werden.
Die Unterschiede in der Rechenzeit mit oder ohne Einbeziehung der Sonnenenergie sind durch die zusätzlich benötigten Berechnungen zu erklären, da für jeden Teilpfad mehrfach die auf den Zeppelin treffende Solarengergie und daraus folgend die Fluggeschwindigkeit berechnet werden müssen, wohingegen bei einer konstanten Geschwindigkeit hier keine Rechnung benötigt wird. Die Verkürzung der Rechenzeit bei der zusätzlichen Berücksichtigung der Winddaten liegt darin begründet, dass ein Pfad sofort verworfen wird, sobald die erreichte Geschwindigkeit des Zeppelins kleiner ist als die entgegengesetzte Windgeschwindigkeit. Dies tritt unter Berücksichtigung der Solarenergie häufig auf. Daher werden für diesen Pfad ab dann keine weiteren Berechnungen in dieser Iteration erforderlich.

Szenario	kürzeste Flugdauer	∅ Flugdauer	∅ Rechenzeit [1]
Kein Wind, konst. Geschwindigkeit	53,66 h	53,66 h	24 s
Kein Wind, var. Geschwindigkeit incomplete	100,11 h	100,13 h	311 s
Wind, konst. Geschwindigkeit	38,60 h	38,72 h	23 s
Wind, var. Geschwindigkeit	93,12 h	96,79 h	256 s

[1] auf AMD Ryzen 5 3600[3] Open-MP aktiv

Tabelle 3: Ergebnisse New York - London am 20.03.2019

Die Flugzeit auf dem Rückflug ohne Umwelteinflüsse ist erwartungsgemäß gleich lang wie auf dem Hinflug. Unter Einbeziehung der Winddaten verkürzt sich die Flugzeit gegenüber der Flugzeit ohne Wind, da der vorherrschende Westwind als Rückenwind genutzt werden kann.
Für den Rückflug wurde die Startzeit auf 12 Uhr UTC festgelegt, um die Zeitverschiebung nach New York zu berücksichtigen. Dennoch fällt auf, dass der Rückflug mit variabler Geschwindigkeit ohne Windeinfluss 14 Stunden länger dauert. Dieses Ergebnis ist auf die Richtung der Erdrotation in Verbindung mit der Flugrichtung zurückzuführen. Durch die Erdrotation verschiebt sich die Sonneneinstrahlung von Ost nach West über den Globus. Fliegt der Zeppelin nun auch in Richtung Westen, folgt er der Sonne und wird somit länger von der Sonne beschienen. Dadurch dauert ein Tag für einen in dieser Richtung fliegenden Zeppelin länger als für einen stillstehenden Beobachter. In der Nacht verringert sich die Fluggeschwindigkeit durch die fehlende Sonneneinstrahlung und somit existiert auch der Effekt einer verlängerten Nacht nur vernachlässigbar. Somit verschiebt sich das Verhältnis von Tag und Nacht zu Gunsten des Tages, wovon ein durch Solarenergie angetriebenes Luftschiff profitiert. Auf dem Rückflug kehrt sich nun die Wirkung um. Durch das Fliegen in Richtung Osten wird der Tag, im Vergleich zu einem stillstehenden Beobachter, verkürzt. Die Länge der Nacht ändert sich jedoch aufgrund der niedrigen Geschwindigkeit nicht in ausschlaggebendem Maße. Die so entstehende Veränderung des Tag-Nacht-Verhältnisses ist suboptimal für ein solargetriebenes Luftfahrzeug. Somit lässt sich die unterschiedliche Flugdauer auf Hin- und Rückflug damit erklären, dass auf dem Rückflug eine Nacht mehr in Kauf genommen werden muss, um das Ziel zu erreichen.
Unter der Berücksichtigung von Wind- und Sonneneinfluss zeigt sich, dass der soeben beschriebene Einfluss der Flugrichtung nur schwer durch die besseren Windverhältnisse ausgeglichen werden kann. Dennoch ist der Zeppelin auf dem Rückflug knapp 19 Stunden schneller als auf dem Hinflug.

6.2 Vergleich von durchschnittlicher und berechneter Geschwindigkeit

Im vorigen Kapitel wurde die Durchschnittsgeschwindigkeit von 104 $\frac{km}{h}$ angenommen, welche aus dem Sonneneinfluss vom 06.06.2018 berechnet wurde. In der nächsten Simulationssreihe wird untersucht, wie sich die Verwendung von einer, durch den Sonnenstand beeinflussten, variablen Geschwindigkeit auswirkt. Hierfür wird die Route London - New York als Grundlage angenommen. Die Winddaten werden auf die Daten vom 06.06.2019 aktualisiert. Als Startdatum und -zeit wird in London der 06.06.2019 um 05:00 Uhr UTC festgelegt. Alle weiteren Parameter bleiben im Vergleich zu Kapitel 6.1 unverändert.

Szenario	kürzeste Flugdauer	∅ Flugdauer	∅ Rechenzeit [1]
Kein Wind, konst. Geschwindigkeit	53,66 h	53,66 h	24 s
Kein Wind, var. Geschwindigkeit	74,54 h	74,71 h	320 s
Wind, konst. Geschwindigkeit	66,49 h	66,50 h	28 s
Wind, var. Geschwindigkeit	112,87 h	162,11 h	395 s

[1] auf AMD Ryzen 5 3600[3] Open-MP aktiv

Tabelle 4: Ergebnisse London - New York am 06.06.2019

Da für diese Simulationsreihe keine Änderungen an der Route oder dem Fluggerät vorgenommen wurden, bleibt die Flugzeit ohne Umwelteinflüsse im Vergleich zu den Simulationsreihen aus Kapitel 6.1 unverändert.
Die längere Flugzeit im Szenario mit Windberücksichtigung und konstanter Geschwindigkeit zeigt die Auswirkungen der unterschiedlichen Windverhältnisse. Beim Vergleich der Szenarien mit konstanter und variabler Geschwindigkeit fällt jedoch auf, dass der Flug mit einer variablen Geschwindigkeit länger dauert als mit einer konstanten. Unter der Berücksichtigung, dass die konstante Geschwindigkeit aufgrund der Sonneneinstrahlung des 06.06.2018 am Startpunkt gewählt wurde, gibt es für diese Beobachtung zwei Gründe. Einerseits folgt der Zeppelin aufgrund der Kugelform der Erde bei einer konstanten Geschwindigkeit einer Great-Circle-Line. Diese stellt auf der kugelförmigen Erde die kürzeste Verbindung zwischen zwei Punkten dar. Für die Strecke zwischen London und New York liegt sie größtenteils nördlicher als der Startpunkt, wodurch die Sonneneinstrahlung auf den Zeppelin schwächer ausfällt. Andererseits wirkt sich die Geschwindigkeit des Luftschiffes quadratisch auf die benötigte Energie aus (siehe [14]). Dadurch ergeben sich im Vergleich zu einem System, welches die Energie speichert und damit gleichmäßig abgeben kann, Einbußen in der Höhe der erreichten Geschwindigkeit.

Szenario	kürzeste Flugdauer	∅ Flugdauer	∅ Rechenzeit [1]
Kein Wind, konst. Geschwindigkeit	53,66 h	53,66 h	24 s
Kein Wind, var. Geschwindigkeit	81,96 h	82,00 h	318 s
Wind, konst. Geschwindigkeit	40,88 h	40,90 h	23 s
Wind, var. Geschwindigkeit	105,00 h	119,26 h	310 s

[1] auf AMD Ryzen 5 3600[3] Open-MP aktiv

Tabelle 5: Ergebnisse New York - London am 06.06.2019

Um die Zeitverschiebung zu berücksichtigen wurde beim Rückflug die Abflugzeit auf 9 Uhr UTC festgelegt.
Die Zunahme der Flugdauer des Fluges unter dem Einfluss von Wind- und Sonnendaten kommt dadurch zustande, dass die beiden Einflüsse in diesem Szenario gegeneinander wirken. Die Windgeschwindigkeit und -richtung sind weiter im Norden vorteilhafter, dort ist jedoch die Sonneneinstrahlung niedriger. Auf einer südlicheren Route würde der Zeppelin aus eigener Kraft eine höhere Geschwindigkeit erreichen, wird dabei aber nicht von für den Flugverlauf hilfreichen Winden unterstützt.

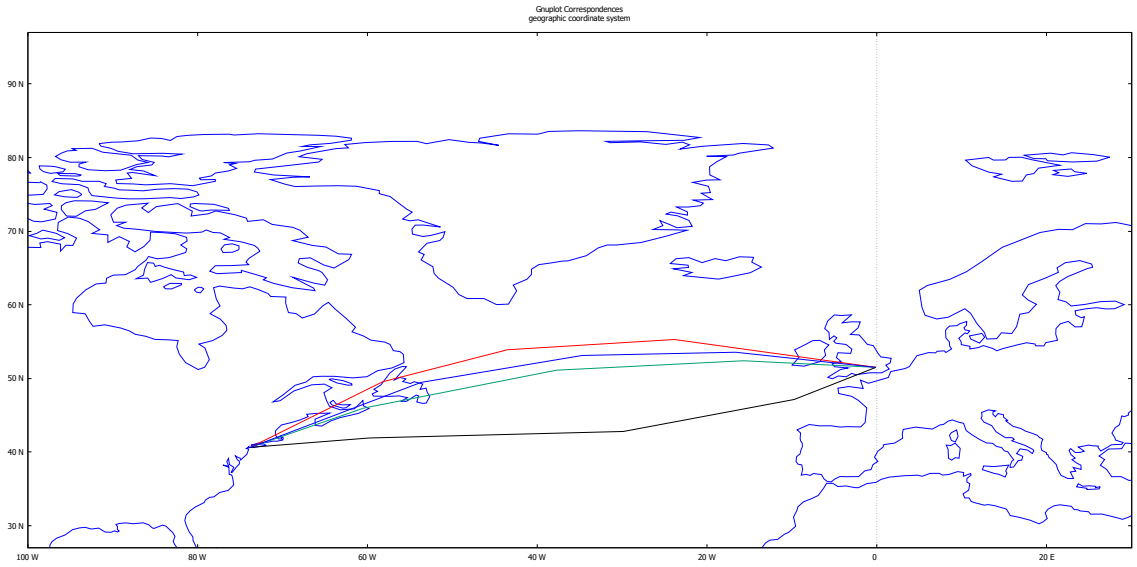

Abbildung 3: Flugrouten London - New York zu verschiedenen Jahreszeiten
rot: Sommer blau: Great Circle Line
grün: Sonnenwende schwarz: Winter

6.3 Einfluss der Jahreszeiten

In den beiden vorangegangenen Simulationsreihen zeigte sich schon die Auswirkung der Jahreszeit auf die Flugzeit. Diese Auswirkung soll nun für sich alleine betrachtet werden. Dafür wird der schon bekannte Zeppelin wieder auf der Route zwischen London und New-York eingesetzt. Die Simulationen werden ohne zusätzliche Winddaten durchgeführt, da verschiedene Winddaten für die einzelnen Jahreszeiten die Ergebnisse verfälschen würden und gleiche Winddaten für alle Tage keinen Mehrwert bedeuten. Die Startzeitpunkte orientieren sich an den Tag-Nacht-Gleichen sowie an der Sommer- und Wintersonnenwende. Die Startzeit wurde auf 08:00 UTC festgelegt. Somit ergeben sich für die Strecke von London nach New York folgende Flugzeiten:

Flugdatum	kürzeste Flugdauer	∅ Flugdauer
21.03.2019	86,18 h	86,23 h
21.06.2019	74,15 h	74,38 h
21.09.2019	86,30 h	86,32 h
21.12.2019	160,10 h	160,96 h

Tabelle 6: Ergebnisse London - New York ohne Windeinfluss

Diese Simulation verdeutlicht die Unterschiede der Flugzeiten je nach der Jahreszeit. In Abbildung 3 sind die Flugrouten für drei in der Tabelle 6 markierten Tage grafisch dargestellt. In Blau ist die Strecke der kürzesten Entfernung, der Great-Circle-Line, zwischen London und New York eingezeichnet. Anhand der Grafik kann man nun erkennen, dass der Zeppelin im Sommer, also bei hoher Sonneneinstrahlung, eine weiter nördliche Route fliegt. Diese liegt sogar nördlicher als die Great-Circle-Route. Das lässt sich damit erklären, dass im Sommer die Tage auf der Nordhalbkugel länger werden, je weiter nördlich man sich befindet. Daher nutzt der Zeppelin die längeren Tage zu seinem Vorteil, um trotz des etwas längeren Weges und der schwächeren Sonneneinstrahlung schneller zu sein als auf dem direkten Weg. Im Winter wird hingegen ein noch längerer und weiter südlich liegender Weg in Kauf genommen, um auf der Route mehr Sonnenenergie einzufangen. Da diese beiden Daten die Maximal- und Minimalwerte der Sonneneinstrahlung darstellen, liegt die Route zur Tag-Nacht-Gleiche zwischen den Routen bei Sommer- bzw. Wintersonnenwende.
Der Vollständigkeit halber wurde auch der Rückflug an den vier Tagen simuliert. Für diesen wurde die Startzeit auf 12 Uhr UTC angepasst um die Zeitverschiebung zu berücksichtigen. Für die Flug-

dauern ergeben sich Folgende Werte:

Flugdatum	kürzeste Flugdauer	∅ Flugdauer
21.03.2019	99,98 h	100,27 h
21.06.2019	79,50 h	79,55 h
21.09.2019	100,54 h	100,76 h
21.12.2019	171,56 h	171,94 h

Tabelle 7: Ergebnisse New York - London ohne Windeinfluss

Auch hier zeigt sich wieder die in Kapitel 6.1 beschriebene Auswirkung der Flugrichtung auf die Flugzeiten. Die Routenführung für die jeweiligen Tage unterscheidet sich nur minimal von der auf dem Hinflug, daher wurde der Rückflug nicht mit in die Abbildung 3 aufgenommen.
Da bei diesen Durchläufen keine Winddaten verwendet wurden, liegen die Ergebnisse der einzelnen Durchläufe wie in Kapitel 6.1 beschrieben nah beieinander. In der Laufzeit der Simulationen gab es keine auffälligen Differenzen zwischen den einzelnen Szenarien, deshalb ist diese nicht in den Tabellen gelistet.

6.4 Einfluss der Startzeiten

Bisher wurde bei den Simulationsreihen davon ausgegangen, dass der optimale Startzeitpunkt am Morgen nach dem Sonnenaufgang ist. Um diese Annahme zu überprüfen wird in dieser Testreihe die Flugzeit für die Route zwischen Nürnberg und London bei verschiedenen Startzeiten simuliert. Der Zeppelin entspricht auch hier wieder dem unter 6 angelegten Modell. Aufgrund der kürzeren Route wird nur noch eine Pfadlänge von 3 verwendet. Die Anzahl der Partikel und die Anzahl der Iterationen bleiben unverändert. Das Datum für die Simulation wird wieder auf den 20.03.2019 festgesetzt. Nun wird die Flugdauer mit Startzeitpunkten im Abstand von drei Stunden simuliert. Da die Startzeit der Simulation immer in UTC übergeben wird, ist in der Tabelle zusätzlich die Ortszeit am Startpunkt (MEZ) angegeben.

UTC	Ortszeit	∅ Flugdauer	Ankunftszeit
00:00	01:00	16,12 h	16:07
03:00	04:00	13,57 h	16:34
06:00	07:00	11,01 h	17:01
09:00	10:00	8,46 h	17:28
12:00	13:00	5,98 h	17:59
15:00	16:00	9,60 h	00:36 +1
18:00	19:00	20,98 h	14:59 +1
21:00	22:00	18,46 h	15:28 +1

Tabelle 8: Ergebnisse Nürnberg - London ohne Windeinfluss

Für diese Simulationsreihe wurde die kürzere Route zwischen Nürnberg und London genutzt, da diese an einem Tag vollständig geflogen werden kann. Wie bei einem durch Solarenergie angetriebenen Luftschiff zu erwarten, liefern Startzeiten in der Nacht verhältnismäßig lange Flugdauern, da erst ab Sonnenaufgang eine nennenswerte Geschwindigkeit erreicht wird. Die kürzeste Flugdauer ergibt sich bei einer Startzeit am Mittag. Zu dieser Zeit profitiert der Zeppelin von der stärksten Sonneneinstrahlung im Laufe des Tages. Eine Startzeit am Mittag bietet sich somit für Kurzstrecken an, wenn das Ziel noch am gleichen Tag erreicht werden kann. Auf längeren Strecken sollte der Startzeitpunkt früher gewählt werden, da so die Sonneneinstrahlung des gesamten Tages inklusive der Mittagssonne genutzt werden kann.
Ein interessantes Ergebnis liefert die Simulation mit Startzeit um 15:00 UTC. Der Zeppelin erreicht London um 00:36 Uhr Ortszeit, also mitten in der Nacht ohne Sonneneinstrahlung. Mit dieser Startzeit wird der Zielort nicht mehr während des Tages erreicht. Der Zeppelin ist jedoch nah genug, dass er in der Nacht durch die Mindestgeschwindigkeit von 30 $\frac{km}{h}$ bis zum Ziel weiterfliegt und

dieses in der Nacht erreicht.

UTC	∅ Flugdauer	Ankunftszeit[1]	Differenz zum Hinflug
01:00	15,37 h	17:22	-44 min
04:00	12,97 h	17:58	-36min
07:00	10,58 h	18:35	-26 min
10:00	8,93 h	19:56	+28 min
13:00	8,76 h	23:46	+2 h 46 min
16:00	15,57 h	08:34 +1	+5 h 58 min
19:00	20,15 h	16:09 +1	-50 min
22:00	17,75 h	16:45 +1	-43 min

[1] Ortszeit, MEZ(UTC+1)

Tabelle 9: Ergebnisse London - Nürnberg ohne Windeinfluss

Beim Vergleich der Flugdauern zwischen Hin- und Rückflug zeigt sich im Gegensatz zu den vorigen Szenarien nicht, dass der Rückflug immer länger dauert als der Hinflug. Dies widerspricht jedoch nicht der Begründung in Kapitel 6.1, vielmehr zeigt sich hier eine weitere Auswirkung dieses Effektes. Aus der Tabelle 9 ist erkennbar, dass sich die Flugzeiten verkürzen, bei welchen der Zeppelin während des Sonnenaufgangs und Vormittags in der Luft ist. Da der Zeppelin in dieser Flugrichtung der Sonne entgegen fliegt, steigt die Intensität der Sonnenstrahlung am Vormittag schneller an, während sie am Nachmittag schneller abfällt. Fliegt ein Solarzeppelin nun während des Vormittages von West nach Ost, so steigt die auftreffende Sonnenenergie und somit auch die Geschwindigkeit schneller an als bei einem von Ost nach West fliegenden Zeppelin. Daher wird das Ziel schneller erreicht. Fliegt der Zeppelin während der Abendstunden, sinkt die Intensität und damit auch die Geschwindigkeit schneller als bei seinem von Ost nach West fliegenden Pendant.

6.5 Einfluss der Flugstrecke

In dieser letzten Testreihe wird der Einfluss der Flugstrecke auf die Flugdauer untersucht. Unter Vernachlässigung der Beschleunigung findet sich hier bei Fortbewegungsmitteln im Normalfall ein lineares Verhältnis. Da bei dem simulierten Zeppelin die Geschwindigkeit aber hauptsächlich von den Umwelteinflüssen abhängt, ist das Verhältnis zwischen Flugstrecke und Flugdauer nicht mehr linear.
Für diesen Test startet der schon bekannte Zeppelin bei den Koordinaten (0.0000;0.0000) und fliegt dann entlang des Äquators. Nach jeweils 556 km (5 Längengrade) wird ermittelt, wie lange der Zeppelin für den bisherigen Flug gebraucht hat. Als Startzeitpunkt dient der 20.03.2019 um 08:00 UTC, Winddaten werden nicht genutzt.

	Richtung Osten		Richtung Westen	
Strecke	kürzeste Flugdauer	∅ Geschwindigkeit	kürzeste Flugdauer	∅ Geschwindigkeit
556 km	7,87 h	70,6 $\frac{km}{h}$	8,11 h	68,6 $\frac{km}{h}$
1 112 km	13,14 h	84,6 $\frac{km}{h}$	11,03 h	100,8 $\frac{km}{h}$
1 668 km	29,60 h	56,4 $\frac{km}{h}$	26,69 h	62,5 $\frac{km}{h}$
2 224 km	32,35 h	68,7 $\frac{km}{h}$	33,35 h	66,7 $\frac{km}{h}$
2 780 km	44,85 h	62,0 $\frac{km}{h}$	36,42 h	76,3 $\frac{km}{h}$

Tabelle 10: Ergebnisse Entlang des Äquators ohne Windeinfluss

Auch hier zeigt sich bei der Strecke über 556 km wieder der in Kapitel 6.4 beschriebene Effekt, dass ein Flug Richtung Osten schneller ist als in Richtung Westen, solange nicht der gesamte Tag sondern nur die Zeit bis zum Nachmittag benötigt wird. Bei längeren Flugdistanzen, welche den gesamten Tag benötigen ist jedoch das Ziel in Richtung Westen schneller erreicht, was sich bei den Strecken über 1 112 km, 1 668 km und 2 780 km zeigt. Bei der Distanz über 2224 km überwiegt

noch einmal der Vormittagseffekt, jedoch wird dieser schon fast vollständig vom Zeitverlust am Vorabend ausgeglichen.

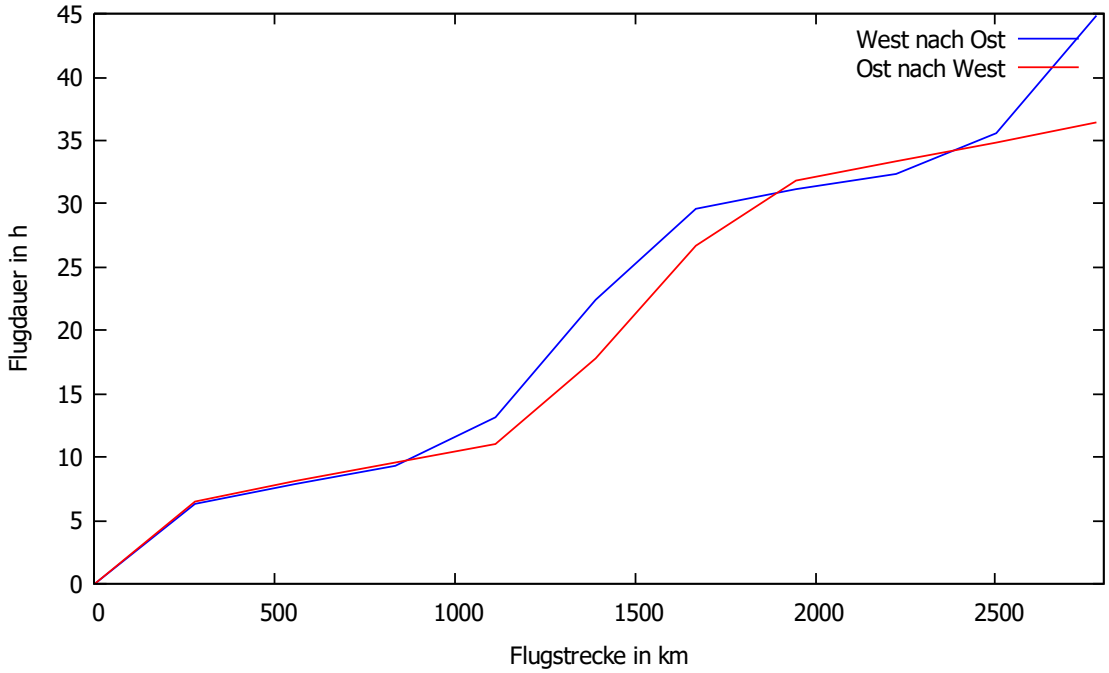

Abbildung 4: Flugdauer in Abhängigkeit der Flugstrecke am Äquator (06.06.2019)

In Abbildung 4 wurden die ermittelten Zeiten aus Tabelle 10 mit weiteren Werten alle 2,5 Längengrade ergänzt. Für die Simulation dieser wurden jedoch weniger Iterationen verwendet, da schon bei den vorher ermittelten Werten keine große Verbesserung mit einer höheren Anzahl von Iterationen feststellbar war. In blau ist die Flugzeit von West nach Ost eingezeichnet, in rot die Gegenrichtung. Hier sind noch einmal deutlich die durchgeflogenen Nächte anhand des starken Anstiegs in der Flugzeit erkennbar. Auch die scheinbare Verlängerung bzw. Verkürzung der Tage kann zwischen 834 km und 1 112 km deutlich erkannt werden

7 Auswirkungen auf veschiedene Flugstrecken

In diesem letzten Kapitel werden noch einmal zwei Flugstrecken unter Berücksichtigung aller Umwelteinflüsse simuliert. Die Route zwischen Nürnberg und Vancouver wurde gewählt, da die beiden Städte auf dem selben Breitengrad liegen, der direkte Weg auf der Great-Circle-Line dennoch sehr weit im Norden verläuft. Hierbei soll untersucht werden, ob der Solarzeppelin hier im Sommer durch den Polartag im Norden einen Vorteil hat.
Die zweite Strecke wurde gewählt, um die Routenfindung auf der Kurzstrecke zu testen. Die Konfiguration des Zeppelins bleibt unverändert, als Simulationsdatum wurde jeweils der 06.06.2019 gewählt. Die Winddaten stammen ebenfalls vom 06.06.2019.

7.1 Nürnberg - Vancouver

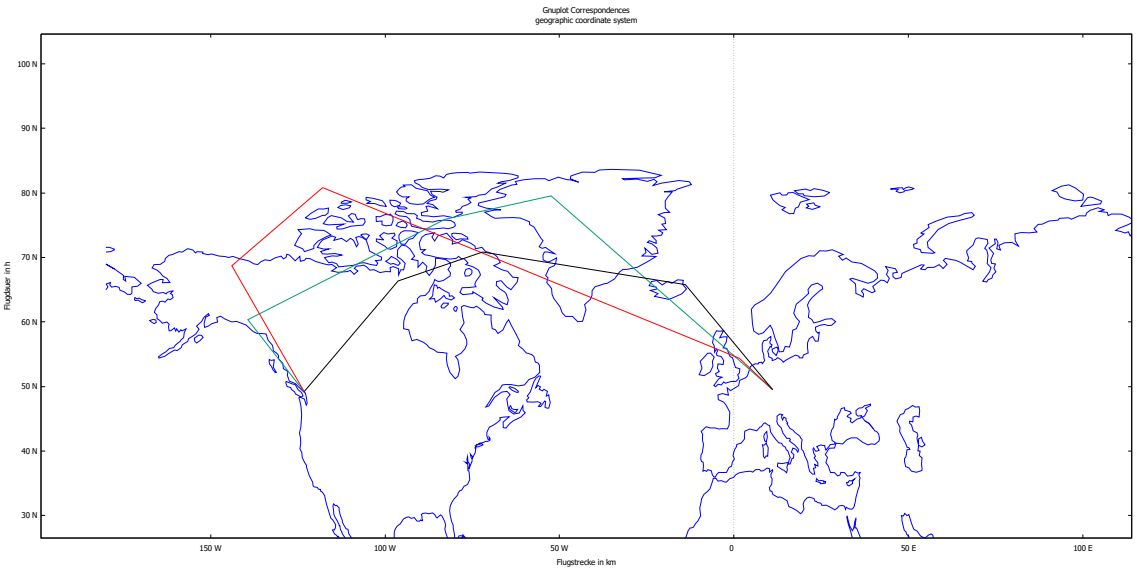

Abbildung 5: Routen Nürnberg - Vancouver am 06.06.2019 unter Berücksichtigung des Windes
rot: Vancouver - Nürnberg schwarz: Great Circle Line
grün: Nürnberg - Vancouver

In Abbildung 5 sind die Verläufe der schnellsten Flugrouten eingezeichnet. Die schwarze Route stellt den direkten Weg zwischen den beiden Städten dar. Durch die geringe Anzahl der Wegpunkte sind alle Routen sehr kantig.
In grün ist die Route von Nürnberg nach Vancouver, in rot der Rückflug eingetragen. In dieser Konfiguration beträgt die kürzeste errechnete Flugdauer nach Vancouver 115,5 Stunden, also 4 Tage, 19 Stunden und 30 Minuten. Der Rückflug nach Nürnberg dauert 121,5 Stunden und damit 6 Stunden länger. Bei einer Entfernung von 8200 km ergeben sich damit Durchschnittsgeschwindigkeiten von 71 bzw 67,5 $\frac{km}{h}$
Trotz der wenigen Wegpunkte findet der Algorithmus eine schnellste Route, welche auf den ersten Blick wie ein Umweg erscheint. Auch führen sowohl Hin- als auch Rückflug nördlich der Great-Circle-Line entlang, da der Zeppelin so den vorhandenen Polartag ausnutzen kann. Der große Umweg über Kanada lässt vermuten, dass hier starke Winde das Vorankommen den Solarzeppelins verhindern.

7.2 New York - Vancouver

Der abschließende Test dieser Arbeit simuliert den Zeppelin auf der Strecke zwischen New York und Vancouver. Hiermit soll die Annahme über die ungünstigen Winde aus der vorigen Simulation

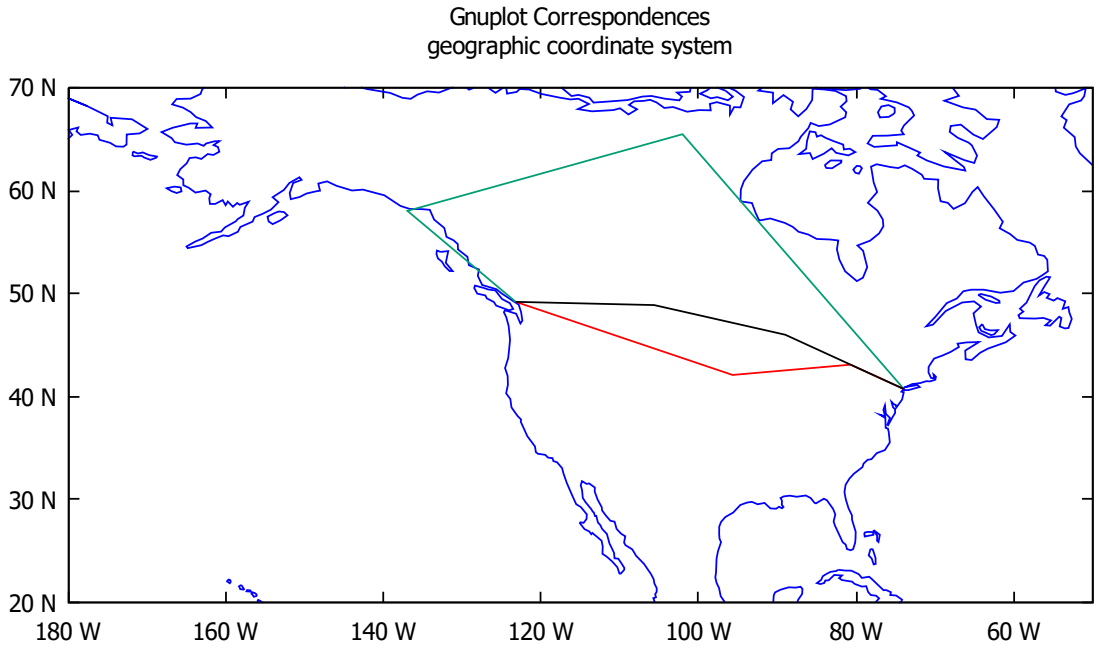

Abbildung 6: Routen New York - Vancouver am 06.06.2019 unter Berücksichtigung des Windes
rot: Vancouver - New York schwarz: Great Circle Line
grün: New York - Vancouver

überprüft werden. Weiterhin werden hier zwei gleichzeitig fliegende Zeppeline simuliert, die Startzeit wurde also nicht an die Ortszeit angepasst. In Abbildung 6 sind der Hin- und Rückflug in grün bzw. rot eingezeichnet. Die schwarze Route stellt wieder den direkten Weg dar.

Für den Hinflug braucht der Zeppelin auf dieser Route 87,05 h. Bei einer Entfernung von 3 920 km ergibt sich somit eine Durchschnittsgeschwindigkeit von nur 45,0 $\frac{km}{h}$. Ein Blick auf Abbildung 6 zeigt, dass der Zeppelin für ihn ungünstige Winde umfliegen muss und daher einen nicht unerheblichen Umweg fliegt.

Für den Flug von Vancouver nach New York braucht der Zeppelin an diesem Tag 75,18 h, was einer Durchschnittsgeschwindigkeit von 52,1 $\frac{km}{h}$ entspricht. Dieser Zeppelin kann den Vorteil der Winde nutzen, startet jedoch aufgrund der Zeitverschiebung vor Sonnenaufgang, was zu einem Zeitverlust führt.

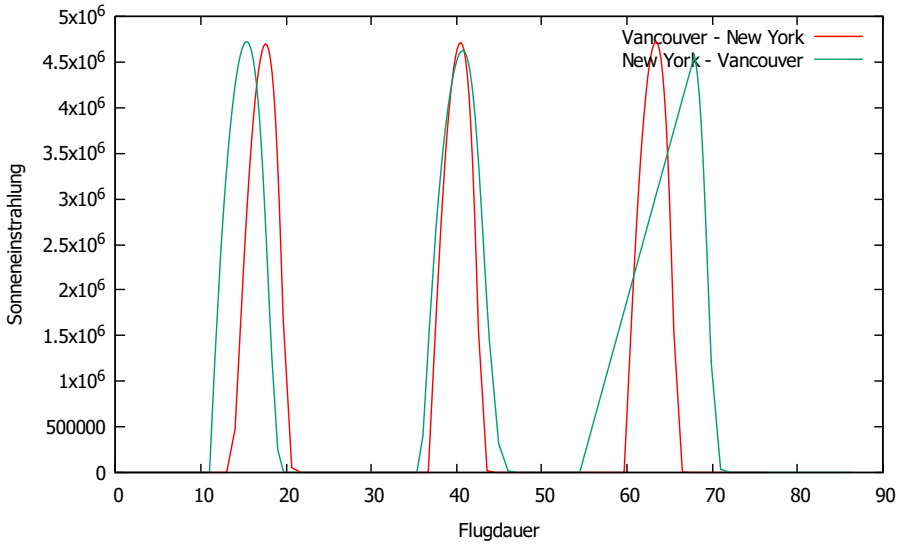

Abbildung 7: Energieertrag New York - Vancouver am 06.06.2019 unter Berücksichtigung des Windes

In Abbildung 7 sind die Energieerträge der beiden Zeppeline grafisch dargestellt. Am Ausschlag des ersten Tages erkennt man den späteren Sonnenaufgang und -untergang in Vancouver. Am zweiten Tag sind beide Zeppeline etwa auf einer Höhe, jedoch mit einem großen Abstand in Nord-Süd Richtung. Hier erkennt man, dass der Tag für den Zeppelin im Norden (grün) zwar länger ist als für den südlichen Zeppelin, jedoch ist die maximal aufgefangene Sonnenenergie durch den flacheren Einstrahlungswinkel der Sonne geringer.

8 Fazit

Aufbauend auf der Arbeit von Christopher Newman wurde in dieser Arbeit der Partikelschwarm Optimierungsalgorithmus modifiziert, um neben realen Winddaten auch reale Sonnendaten verarbeiten zu können. Zudem wurde eine Benutzeroberfläche eingeführt, welche zusammen mit weiteren Änderungen an der vorliegenden Simulation dazu führt, dass verschiedene Szenarien einfacher und schneller simuliert werden können.
Im Anschluss wurden die Auswirkungen eines Antriebs durch Solarenergie an verschiedenen Testszenarien untersucht. Dabei wurde festgestellt, dass die Jahreszeit einen erheblichen Einfluss auf die Flugzeit hat. So kann im Sommer eine nördliche und kürzere Route geflogen werden, als im Winter. Die Routen im Winter verlaufen weiter südlich, da dort der Vorteil der höheren Sonneneinstrahlung mehr wiegt als der Nachteil der längeren Flugdistanz. Der Einfluss der Wahl der Startzeit ist bei Kurzstrecken am größten und nimmt mit zunehmender Streckenlänge ab. Als weiterer Parameter der Flugzeit wurde die Flugrichtung von Ost nach West oder West nach Ost identifiziert.
Aufgrund der errechneten Flugzeiten ist bei dem jetzigen technologischen Stand von Dünnschichtsolarzellen nicht davon auszugehen, das Solarzeppeline im Passagierverkehr eine ernstzunehmende Konkurrenz zu klassischen Flugzeugen darstellen. Jedoch sind andere Nutzungsszenarien denkbar. Somit könnten mit groß genug dimensionierten Solarzeppelinen Hilfsgüter in abgelegene oder durch Umweltkatastrophen nicht erreichbare Gebiete gebracht werden. Ein Luftschiff braucht dabei keine intakten Strukturen am Boden und ist dank Solarbetrieb unabhängig von der Energieversorgung am Boden. Weiterhin kann es auch ohne Energieaufwand an einem Punkt stehen und somit zum Beispiel die Kommunikation in Katastrophengebieten sichern. Auch als Forschungsplattform in der Atmosphäre wären solarbetriebene Luftschiffe geeignet.
Die hier entwickelte Simulation bietet eine gute Grundlage für weitere Forschungen. Nächste mögliche Schritte zur Erweiterung wären die Einbeziehung der Abschwächung der Sonne durch Wolken sowie die Reflexion von Sonnenlicht an Wolken, wodurch für einen über den Wolken fliegenden Solarzeppelin mehr Energie zur Verfügung stehen würde.
Der Ansatz eines vollständig klimaneutralen Verkehrsmittels ist vielversprechend und sollte weiter verfolgt werden.

9 Literaturverzeichnis

[1] URL: https://de.wikipedia.org/wiki/Str%C3%B6mungswiderstand (besucht am 04. 10. 2019).

[2] airships.net. *Hindenburg Statistics*. URL: https://www.airships.net/hindenburg/size-speed/ (besucht am 27. 09. 2019).

[3] *AMD Ryzen 5 3600*. URL: https://www.amd.com/de/products/cpu/amd-ryzen-5-3600 (besucht am 08. 10. 2019).

[4] Dan Rutherford Ph.D. Brandon Graver Ph.D. Kevin Zhang. „CO2 emissions from commercial aviation, 2018“. In: (2019). URL: https://theicct.org/sites/default/files/publications/ICCT_CO2-commercl-aviation-2018_20190918.pdf (besucht am 24. 09. 2019).

[5] Mustafa Cavcar. *The International Standars Atmosphere (ISA)*. URL: http://fisicaatmo.at.fcen.uba.ar/practicas/ISAweb.pdf (besucht am 09. 10. 2019).

[6] *Exakte Dichtebestimmung der Luft*. URL: https://de.wikipedia.org/wiki/Luftdichte#Exakte_Dichtebestimmung_der_Luft (besucht am 06. 10. 2019).

[7] Heinz Eschrich Hans-Günther Wagemann. *Grundlagen der photovoltaischen Energiewandlung (= Teubner Studienbücher Physik)*. 1994.

[8] infinityPV. URL: https://infinitypv.com/images/infinityPV_OPV_organic_solar_cells.pdf (besucht am 01. 10. 2019).

[9] *Internationale Höhenformel*. URL: https://de.wikipedia.org/wiki/Barometrische_H%C3%B6henformel#Internationale_H%C3%B6henformel (besucht am 06. 10. 2019).

[10] *ISO-Container*. URL: https://de.wikipedia.org/wiki/ISO-Container (besucht am 08. 10. 2019).

[11] Vladimir Bulovic Joel Jean Annie Wang. „In situ vapor-deposited parylene substrates for ultra-thin, lightweight organic solar cells“. In: (2016). URL: https://doi.org/10.1016/j.orgel.2016.01.022 (besucht am 01. 10. 2019).

[12] Kimberly Lightle. „Solar Energy, Albedo, and the Polar Regions“. In: (2008). URL: https://beyondpenguins.ehe.osu.edu/issue/energy-and-the-polar-environment/solar-energy-albedo-and-the-polar-regions (besucht am 02. 10. 2019).

[13] Christopher Newman. „Optimal Control of a Solar Airship“. 2019.

[14] Global Solar. URL: http://www.globalsolar.com/sites/default/files/uploads/images/Copy%20of%20PowerFLEX%20BAPV%20Datasheet%20(PROD%20LIT%20-%201000781%20-%201%20-%20D).PDF (besucht am 01. 10. 2019).

[15] *SolTrack*. URL: http://soltrack.sourceforge.net (besucht am 22. 09. 2019).

[16] Kai Teichmann. „Simulation eines Solar-Zeppelin“. 2018.

[17] *TinyXML-2*. URL: http://www.grinninglizard.com/tinyxml2/ (besucht am 30. 09. 2019).

[18] Umweltbundesamt. „Klimawirksamkeit des Flugverkehrs“. In: (2012). URL: https://www.umweltbundesamt.de/sites/default/files/medien/377/dokumente/klimawirksamkeit_des_flugverkehrs.pdf (besucht am 24. 09. 2019).

[19] Deutscher Wetterdienst. *Luftdichte*. URL: https://www.dwd.de/DE/service/lexikon/Functions/glossar.html?lv2=101518&lv3=607748 (besucht am 25. 09. 2019).

[20] A. T. Young. „Air mass and refraction“. In: *Applied Optics 33(6)* (1994).

10 Anhang

10.1 Verwendete Tools und Bibliotheken

ecCodes
: Programm um Winddaten (grib2) zu entpacken und zu konvertieren

gnuplot
: Programm zur Visualisierung

SolTrack
: Bibliothek zur Berechnung des Sonnenstandes und der Sonnenposition

TinyXML-2
: Bibliothek zum Einlesen von XML-Dateien

10.2 Abkürzungsverzeichnis

ISA
: ICAO - International standard atmosphere

MEZ
: Mitteleuropäische Zeit, UTC +1

PSO
: Partikelschwarm Optimierungsalgorithmus

UI
: User interface / Benutzeroberfläche

UTC
: Coordinated Universial Time - koordinierte Weltzeit

10.3 Klassendiagramm UI

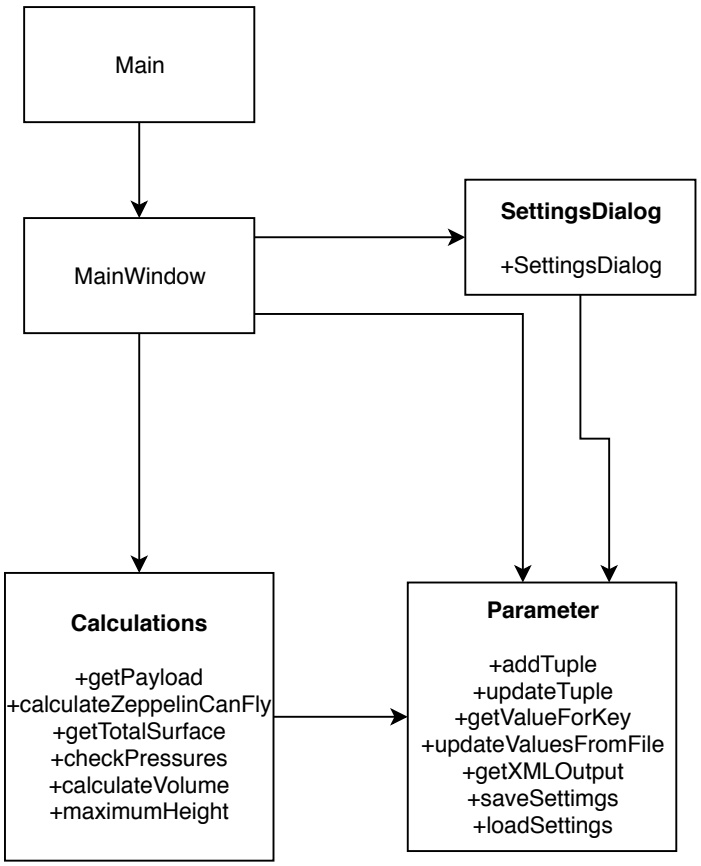

Abbildung 8: Klassendiagramm der Benutzeroberfläche