Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Masterthesis

Matthies Becker

Dezentrale, autonome Flugsicherung für unbemannte Luftfahrzeuge

Fakultät Technik und Informatik
Studiendepartment Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Matthies Becker

Dezentrale, autonome Flugsicherung für unbemannte Luftfahrzeuge

Masterthesis eingereicht im Rahmen der Masterprüfung

im Studiengang Master of Science Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. rer. nat. Thomas Lehmann
Zweitgutachter: Prof. Dr. Bettina Buth

Eingereicht am: 4. Mai 2018

Matthies Becker

Thema der Arbeit
Dezentrale, autonome Flugsicherung für unbemannte Luftfahrzeuge

Stichworte
Flugsicherung, unbemannte Luftfahrzeuge, UAV, UAS, ATC, Drohnen, Car2Car-Kommunikation, autonome Luftfahrzeuge, UAV2UAV-Kommunikation, kooperative Flugsicherung

Kurzzusammenfassung
Autonome, unbemannte Luftfahrzeuge, auch Drohnen genannt, erlangen eine immer größere Verbreitung sowohl bei privaten als auch kommerziellen Nutzern. Diese Arbeit beschäftigt sich damit, eine dezentrale, autonome Flugsicherungslösung für diese unbemannten Luftfahrzeuge zu finden. Diese Flugsicherung soll sicherstellen, dass unbemannte Luftfahrzeuge einen Luftraum kooperativ nutzen können, ohne dabei mit anderen Luftfahrzeugen oder Infrastruktur zu kollidieren. Als Grundlage hierfür sollen möglichst etablierte Technologien und Verfahren verwendet werden, wie etwa die klassische Flugsicherung oder die Car2Car-Kommunikation.

Matthies Becker

Title of the paper
Decentralised, autonomous air traffic control for unmanned aircraft

Keywords
Air traffic control, unmanned aircraft, UAV, UAS, ATC, drone, Vehicle2Vehicle-Communication, autonomous aircraft, UAV2UAV-Communikation, cooperative air traffic control

Abstract
Unmanned autonomous aircraft, or drones, are spreading to private and commercial users. This master thesis researches an air traffic control solution for those unmanned autonomous aircrafts, allowing them to cooperatively use an airspace without colliding with each other or the environment. To easy the development, it is intended to use known technics and procedures such as conventional air traffic control or the Vehicle2Vehicle-Communication.

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

Durch technologische Weiterentwicklung und die Identifizierung von neuen Anwendungsgebieten wird weltweit der private und kommerzielle Einsatz von unbemannten Luftfahrzeugen, umgangssprachlich auch Drohnen genannt, immer attraktiver. So wurde 2016 mit privaten und kommerziellen unbemannten Luftfahrzeugen ein Umsatz von 4,5 Milliarden US-Dollar erwirtschaftet, was einem Umsatzwachstum von 35,5 % zum Vorjahr entspricht. Für das Jahr 2017 wird mit einem Umsatz von ca. 6 Milliarden US-Dollar erneut ein Umsatzwachstum von ca. 34 % erwartet. Die EU rechnet bis 2035 mit einem Umsatz von jährlich 10 Milliarden Euro alleine in Europa. Der wachsende Markt führt zu einer deutlich steigenden Anzahl von unbemannten Luftfahrzeugen. Im Jahr 2016 wurden knapp 2,2 Millionen Drohnen gefertigt, was einen Anstieg um 60,3 % zum Vorjahr bedeutet. Für 2017 wird ein Anstieg um ca. 40 % auf knapp 3 Millionen produzierte unbemannte Luftfahrzeuge prognostiziert. In Europa geht die EU von einer Gesamtzahl von 7 Millionen unbemannten Luftfahrzeugen im Jahr 2050 aus [12, 23, 31, 30].

Aus diesem Zuwachs an Drohnen entstehen vorher so nicht vorhandene Probleme. Mit einer immer größeren Anzahl an unbemannten Luftfahrzeugen im Luftraum steigt die Wahrscheinlichkeit, dass diese miteinander, mit konventionellen Luftfahrzeugen sowie topographischen Objekten kollidieren. Diese Kollisionsgefahr wurde sowohl von Regulierungsbehörden als auch von der Forschung erkannt. So haben mehrere Länder die Verwendung von Drohnen mittlerweile per Gesetz reguliert. Beispielsweise gilt in Deutschland, dass unbemannte Luftfahrzeuge nicht in allen Lufträumen und nur auf Sicht geflogen werden dürfen. Für Drohnen gelten außerdem je nach Gewichtsklasse unterschiedliche Kennzeichnungs-, Sachkundenachweis- und Aufstiegserlaubnispflichten. In der Forschung wird, neben anderen Themen im Bereich unbemannte Luftfahrzeuge, an der Integration von konventioneller Luftfahrt und unbemannter Luftfahrt gearbeitet. Vorangig ist hierbei das Problem der Flugsicherung. Beispiele hierfür sind das UTM-Projekt der NASA sowie das U-Space-Projekt der EU [4, 23, 5, 31, 32].

Sowohl bei der Integration von konventioneller und unbemannter Luftfahrt als auch bei der Entwicklung von reinen Multi-Drohnen-Anwendungen stellt die Flugsicherung das größte Problem dar. Die konventionelle Flugsicherung wird von dafür ausgebildeten Menschen anhand von Sicht und/oder technischen Hilfsmitteln sowie Sprechfunk durchgeführt. Kleinere unbemannte Luftfahrzeuge wären nicht in der Lage alle dafür nötigen technischen Geräte mitzuführen. Außerdem müsste die Flugverkehrskontrolle deutlich mehr Personal für die Überwachung der zusätzlichen Luftfahrzeuge bereitstellen und die Führer von unbemannten Luftfahrzeugen hätten eine Ausbildung in Funk- und Flugsicherungsverfahren zu absolvieren. Weiterhin sind unbemannte Luftfahrzeuge teilweise sehr klein und in der Lage, einen deutlich dichteren Verkehr als konventionelle Luftfahrzeuge zu bilden. Hinzu kommt, dass das zur konventionellen Flugsicherung benötigte Radar lediglich eine Auflösung erreicht, durch welche es kleine Objekte gar nicht und viele kleine Objekte nicht getrennt erfassen kann. Aus diesen Gründen ist die direkte Anwendung der Konzepte und Systeme der konventionellen Flugsicherung für unbemannte Luftfahrzeuge nicht möglich [20].

Intensiviert wird die Problematik wenn auch autonome unbemannte Luftfahrzeuge am Flugsicherungssystem teilnehmen sollen. Diese haben weder einen Führer noch sind sie in der Lage am Sprechfunk teilzunehmen. Eine zuverlässige visuelle Flugsicherung ist mit autonomen unbemannten Luftfahrzeugen ebenso nicht realistisch.

1.1 Ziel

Ziel dieser Arbeit ist es, ein Flugsicherungskonzept für autonome unbemannte Luftfahrzeuge zu entwickeln. Darüber hinaus ist zu zeigen, dass dieses Konzept funktioniert und in einer Demonstrationsanwendung umsetzbar ist.

Das Konzept soll auf reine autonome Multi-Drohnen-Anwendungen beschränkt sein und, im Gegensatz zum UTM- oder U-Space-Projekt, keine Integration in die bestehende Flugsicherung anstreben. Dieser Ansatz wird gewählt um ausschließlich das Kernproblem, eine Flugsicherung für autonome unbemannte Luftfahrzeuge, betrachten zu können. Die Schwierigkeiten der Regulierung und Flugverkehrskontrolle der internationalen Luftfahrt sowie die problematischen Kommunikation zwischen bemannten und unbemannten Luftfahrzeugen werden so abgegrenzt. Anwendung kann ein solches Konzept unter anderem in Business-to-Business-Szenarien finden, beispielsweise bei internen Logistikdienstleistungen innerhalb einer Firma, entweder auf einem Gelände oder zwischen verschiedenen Standorten. Dieser Bereich wird

voraussichtlich der Erste sein, in dem sich eine Nachfrage nach autonomen Logistikdrohnen entwickelt [12].

In der konventionellen Flugsicherung stellt der menschliche Faktor durch die Sprachkommunikation sowie den Kapazitäts- und Ausbildungsbedarf ein wesentliches Hindernis für die Integration von unbemannten Luftfahrzeugen dar. Aus diesem Grund soll das zu entwickelnde Flugsicherungskonzept komplett autonom funktionieren. Dies ist auch für die Einbindung von autonomen Luftfahrzeugen von Vorteil.

Bei der Entwicklung des Konzepts der autonomen Flugsicherung sind mehrere Teilkonzepte zur Lösung verschiedener Probleme nötig. Um eine autonome Flugsicherung entwickeln zu können, muss der Luftraum so strukturiert und modelliert werden, dass er als Basis für Algorithmen dienen kann. Aufbauend auf der Luftraummodellierung muss ein Konzept entworfen werden, das definiert, wie eine sichere Separation der Luftfahrzeuge gewährleistet wird. Da es im Rahmen des Flugbetriebs zu Situationen kommt, wo die Gefahr besteht, dass diese Separation verletzt wird, und damit eine Kollision droht, muss ein Algorithmus zur Erkennung dieser Situationen entwickelt werden. Um die erkannten Konfliktsituationen bei der Separation zu lösen, bedarf es eines weiteren Algorithmus, der Ausweichmanöver für die unbemannten Luftfahrzeuge errechnet. Da es sich um autonome Luftfahrzeuge handelt, benötigen diese außerdem einen Wegfindungsalgorithmus um auf Basis des Luftraummodells selbstständig ihre Flugrouten planen zu können.

Eine rein passive Lösung für Flugsicherung, zum Beispiel auf Basis von visueller oder funkgestützter Ortung, ist aufwendig und nicht mit ausreichender Sicherheit realisierbar. Beispielsweise ist eine visuelle Erkennung bei Nacht oder Nebel unmöglich. Außerdem hat sowohl visuelle als auch funkgestützte Ortung hat ein „Sichtfeld“, welches durch Objekte so weit eingeschränkt werden kann, dass eine Kollision nicht rechtzeitig und sicher erkannt wird. Daher wird dieser Ansatz nicht verfolgt und stattdessen ein aktiver Ansatz auf Basis von Funkkommunikation angestrebt. Für diesen Ansatz bedarf es eines Kommunikationskonzepts, welches ein Protokoll definiert, das sicherstellt, dass immer alle Informationen zu jedem unbemannten Luftfahrzeug vorliegen. Die Anforderungen an einen Kommunikationsstapel, der dieses Protokoll sicher transportieren kann, werden in dieser Arbeit definiert. Eine Entwicklung dieses Kommunikationsstapels inklusive physikalischer Übertragung ist allerdings nicht Teil des Kommunikationskonzeptes oder seiner Umsetzung. Zur Verwendung des Konzeptes wird eine Kommunikation vorausgesetzt, die den Anforderungen entspricht und deren Implementierung dem Verwender der Umsetzung obliegt.

Da die Erprobung des Konzepts mit realen unbemannten Luftfahrzeugen umfangreiche Sicherheitsvorkehrungen benötigt und dabei eine hohe Wahrscheinlichkeit von kollisionsbedingten Totalausfällen zu erwarten ist, wäre sie mit hohen Kosten verbunden. Daher muss im Rahmen der Arbeit eine Simulationsumgebung zur Konzepterprobung sowie für die Demonstrationsanwendung gefunden werden. Die Demonstationsanwendung soll ähnlich dem wahrscheinlichen Einsatzgebiet, der Business-Logistik, gehalten sein. Die simulierten Luftfahrzeuge sollen daher mittelgroße Multikopter sein, die in der Lage sind kleinere Lasten zu tragen.

Bei der Umsetzung des Konzeptes soll eine Softwarearchitektur gewählt werden, die es ermöglicht, sowohl die Flugsicherung als auch andere grundlegende Mechanismen, ohne großen Aufwand in anderen Anwendungen einzusetzen. Dabei soll durch Abstraktion eine hohe Austauschbarkeit einzelner Komponenten, insbesondere des Kommunikationswegs, des Wegfindungsalgorithmus und der Hardwareschicht, erreicht werden. Weiterhin ist es erstrebenswert, die Architektur so zu wählen, dass sich bestehende Komponenten leicht erweitern oder neue Komponenten hinzufügen lassen.

1.2 Vorgehen

Um das Risiko auf systematische Fehler in der Entwicklung zu reduzieren, sollen möglichst viele erprobte Technologien als Grundlage für das Flugsicherungskonzept eingesetzt werden.

Eine der wichtigsten Quellen, die untersucht werden sollen, ist hierbei die konventionelle Flugsicherung. Diese ist zwar relativ grobmaschig und weitreichend, da der Verkehr mit großen Luftfahrzeugen zum einen nur begrenzt dicht werden darf und sich zum anderen über weite Entfernungen streckt. Zudem wird sie nicht autonom sondern von Menschen durchgeführt. Jedoch addressiert sie viele der Probleme, die bei der gemeinsamen Nutzung von Luftraum auftreten können, und definiert Vorgehensweisen zum Lösen dieser Probleme.

Da das Ziel eine autonome Flugsicherung ist, bietet sich der Automotive-Bereich als weitere Quelle für autonome Verkehrssteuerungstechniken an, weil hier schon länger an autonomem Verkehr geforscht wird. Verfahren für den autonomen Fahrzeugverkehr sind zwangsweise deutlich engmaschiger, da der Verkehr hier deutlich dichter werden kann. Auch die Navigation, die in der konventionellen Luftfahrt komplett von Menschen durchgeführt wird und bei einem autonomen Ansatz automatisiert erfolgen muss, ist im Fahrzeugbereich seit längerem im Einsatz. Zudem wird die Kommunikation zwischen verschiedenen Fahrzeugen bzw. Fahrzeugen

und Umgebung während der Fahrt in diesem Bereich bereits erforscht. Allerdings sind alle Verfahren aus dem Automotive-Bereich nur auf zweidimensionalen Verkehr ausgelegt und müssten für den Flugverkehr adaptiert werden.

Die Projekte U-Space und UTM beschäftigen sich mit Flugsicherung und müssen, im Gegensatz zur konventionellen Flugsicherung, auch mit den Eigenheiten des unbemannten Luftverkehrs zurechtkommen. Allerdings verfolgen diese Projekte einen anderen Ansatz und befinden sich noch in der Entwicklung. Trotzdem ist zu untersuchen, ob sie einen Beitrag zur Entwicklung einer Lösungsstrategie für das Problem dieser Arbeit leisten können.

Für die Simulation des Flugsicherungskonzepts ist es erforderlich, einen Simulator zu finden, der in der Lage ist, Luftfahrzeuge zu simulieren. Hierbei ist es wichtig, dass genug Luftfahrzeuge simuliert werden können, um ausreichend Luftverkehr zum Test des Lösungskonzepts erzeugen zu können. Daneben ist es wichtig, dass der Simulator eine Schnittstelle bietet, um die Luftfahrzeuge automatisisert von einer Anwendung aus zu steuern.

1.3 Aufbau

Diese Arbeit gliedert sich in sieben Kapitel. Im nachfolgenden Kapitel werden die Grundlagen der Arbeit erläutert. Dazu werden relevante Aspekte der konventionellen Flugsicherung und aus dem Automotive-Bereich zusammengefasst sowie die Projekte UTM und U-Space genauer vorgestellt. Das dritte Kapitel befasst sich mit der Entwicklung des Flugsicherungskonzepts. Die Teilaspekte Kommunikation, Kollisionserkennung und -management werden hierbei betrachtet und gelöst. Um die Softwarearchitektur geht es im vierten Kapitel. Dort wird eine Architektur vorgestellt, die die Anforderungen der Zielformulierung erfüllt, sowie grundlegende Komponenten des Systems erläutert. Im fünften Kapitel wird die Simulationsumgebung für unbemannte Luftfahrzeuge behandelt. Dabei werden Anforderungen an eine Simulationssoftware erarbeitet, eine Übersicht über geeignete Simulatoren erstellt und, nach einer Evaluation, ein Simulator ausgewählt. Danach folgt im sechsten Kapitel die Evaluation des Konzeptes in der Simulation, sowohl anhand von Testfällen als auch einer Demonstrationsanwendung. Zum Schluss wird die Arbeit in Kapitel Sieben zusammengefasst und es wird ein Ausblick auf weitere mögliche Forschungsthemen gegeben.

2 Grundlagen

In diesem Kapitel werden die wissenschaftlichen Grundlagen dieser Masterthesis erarbeitet und präsentiert. Hierbei wird auf die internationale Flugsicherung nach ICAO-Richtlinien, Themen aus dem Bereich Automotive Technologies und die Simulation von Anwendungen mit unbemannten Luftfahrzeugen eingegangen.

2.1 Grundbegriffe des Fliegens

Zum besseren Verständnis der Arbeit werden in diesem Abschnitt einige Grundbegriffe des Fliegens erläutert.

Die lateralen Entfernungen in der Luftfahrt werden in der Regel in nautischen Meilen (NM), die Geschwindigkeit in Knoten (kn) angegeben. Zur Beschreibung von lateralen Richtungen werden die Begriffe Kurs und Peilung verwendet (s. Abb. 2.1). Ein Kurs ist der Winkel zwischen einer Bezugsrichtung und der Bewegungsrichtung eines Fahrzeugs. Eine Peilung hingegen ist der Winkel zwischen einer Bezugsrichtung und der Richtung zu einem Objekt. Das Objekt kann hierbei sowohl stationär als auch beweglich sein. Angegeben werden die Winkel in Grad (°). Im Luftverkehr wird als Bezugsrichtung Norden verwendet, dementsprechend folgt ein Luftfahrzeug, das exakt nach Norden fliegt, dem Kurs 0 bzw. 360 Grad. Fliegt es genau nach Süden, ist sein Kurs 180 Grad [9, 20].

Vertikale Entfernungen werden hingegen meist in Fuß (ft) angegeben, wohingegen vertikale Bewegungen mit Steig- und Sinkraten beschrieben werden, welche in Fuß pro Minute (ft/min) angegeben sind [20].

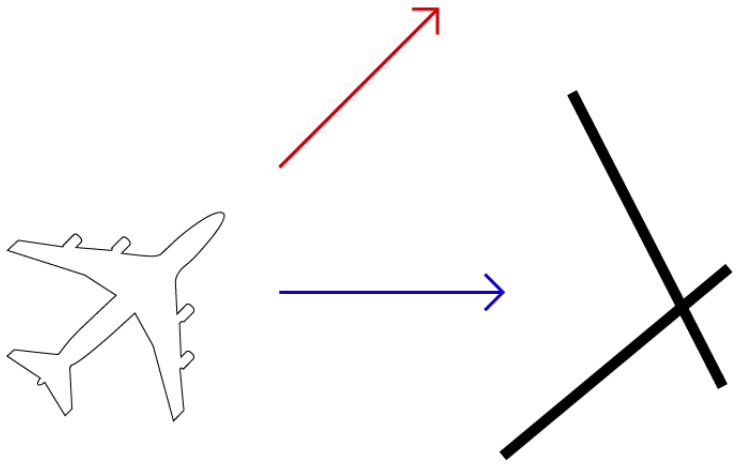

Abbildung 2.1: Kurs (rot) eines Flugzeugs und Peilung (blau) vom Flugzeug zu einem Flughafen. Ist Norden die Bezugsrichtung, ist der Kurs 45 Grad und die Peilung 90 Grad.

2.2 Flugsicherung

Das Konzept der autonomen Flugsicherung muss unter anderem eine Strukturierung des Luftraums, Protokolle für Kommunikation mit unbemannten Luftfahrzeugen sowie Verfahren zur Separation, Kollisionsvermeidung und dem gegenseitigen Ausweichen von unbemannten Luftfahrzeugen liefern. In den folgenden Abschnitten werden die Standards, Techniken und Verfahren der konventionellen Flugsicherung zusammengefasst, die diese Probleme im Bereich der bemannten Luftfahrt lösen. Soweit nicht anders angegeben, stammen die Informationen zur konventionellen Flugsicherung in diesen Abschnitten aus dem Buch „Moderne Flugsicherung“ von Heinrich Mensen [20].

Der Begriff der Flugsicherung beschreibt den Wegsicherungsprozess in der Luftfahrt und alle dazu nötigen Institutionen, Verfahren und Personen. Die Flugsicherung dient zur sicheren, geordneten und flüssigen Abwicklung des Luftverkehrs. Im nationalen und internationalen Luftverkehr basiert die Flugsicherung auf verbindlichen Standards der Internationalen Zivilluftfahrtorganisation (International Civil Aviation Organization, ICAO), deren Umsetzung den ICAO-Mitgliedsstaaten unterliegt. Diese Standards beschreiben unter anderem die allgemeinen Flugverkehrsregeln, die Prozesse der Flugverkehrskontrolle, die Kommunikationsverfahren sowie die Strukturierung und Klassifizierung des Luftraums.

2.2.1 Flugverkehrsregeln

Die Flugverkehrsregeln beinhalten diverse Richtlinien, unter anderem zu Mindestflughöhen, Ausweichregeln sowie Sicht- und Instrumentenflug. Die Richtlinie zur Mindestflughöhe schreibt beispielsweise vor, dass außer zu Start und Landung eine gewisse Höhe nicht unterschritten werden darf. Für diese Arbeit besonders interessant sind die Ausweichregeln und die Regeln zum Sicht- bzw. Instrumentenflug.

Ausweichregeln

Die Ausweichregeln definieren, wie ausgewichen werden muss, wenn sich zwei Luftfahrzeuge auf einer Weise nähern, die zu einer Kollision führen könnte. Die Regeln legen fest, welches Luftfahrzeug auszuweichen hat und wohin ausgewichen werden muss. Dadurch soll vermieden werden, dass Korrekturmanöver von den Luftfahrzeugen ausgeführt werden, die sich gegenseitig neutralisieren oder dass kein Ausweichen ausgeführt wird, da die Zuständigkeit unklar ist. Die wichtigsten Ausweichregeln sind:

1. Befinden sich Luftfahrzeuge im Gegenanflug zueinander und es besteht die Gefahr einer Kollision, haben beide Luftfahrzeuge nach rechts auszuweichen.

2. Kreuzen sich die Flugwege zweier Luftfahrzeuge, muss das Luftfahrzeug, das von links kommt, nach rechts ausweichen. Wenn es sich allerdings um verschieden klassifizierte Luftfahrzeuge handelt, können hier Sonderregeln greifen. So müssen motorisierte Luftfahrzeuge zum Beispiel immer ausweichen, wenn das zweite Luftfahrzeug ein Luftschiff, Segelflugzeug, Hängegleiter oder Ballon ist.

3. Beim Überholen von Luftfahrzeugen muss das überholende Luftfahrzeug nach rechts ausweichen. Als Überholen gilt, wenn sich ein Luftfahrzeug einem anderen von hinten in einem Winkel von weniger als 70 Grad der Flugrichtung nähert.

4. Luftfahrzeugen, die erkennbar in der Manövrierfähigkeit behindert sind, muss immer ausgewichen werden.

5. Ein Luftfahrzeug, das nach diesen Regeln nicht auszuweichen hat, muss Kurs und Geschwindigkeit halten, bis die Kollisionsgefahr abgewendet ist.

Zusätzlich gibt es noch einige weitere Regeln, bei denen es sich hauptsächlich um allgemeine oder Ausweichregeln im Zusammenhang mit Starts und Landungen von Luftfahrzeugen handelt.

Sicht- und Instrumentenflugregeln

Sicht- und Instrumentenflugregeln beschreiben, auf welche Weise ein Luftfahrzeug einen Flug durchführen darf. Grundsätzlich kann ein Luftfahrzeugführer die Lage seines Luftfahrzeugs im Luftraum auf zwei Arten bestimmen: visuell oder mit Hilfe von Instrumenten. Für beide Flugarten gelten bestimmte Regeln.

Sichtflugregeln Um einen Flug nach Sichtflugregeln durchführen zu können, muss ein Luftfahrzeugführer in der Lage sein, die Lage des Flugzeugs visuell zu bestimmen. Dazu ist eine ausreichende Sicht nötig, beispielsweise muss der Horizont immer sichtbar sein. Daher schreiben die Sichtflugregeln Mindestwerte für die Flugsicht und die Entfernung zu Wolken fest. Des Weiteren gibt es Mindestwetterbedingungen, die erfüllt sein müssen. Diese umfassen die Bodensicht auf dem Flugplatz und die Wolkenuntergrenze. Auch ist festgelegt, welche Ausrüstung ein Luftfahrzeug für einen Flug nach Sichtflugregeln in bestimmten Lufträumen benötigt, beispielsweise UKW-Sprechfunkanlage, Transponder und Flächennavigationsgerät. Ein Sichtflug bei Nacht ist möglich, obliegt jedoch weiteren Regeln und Einschränkungen. Da der Luftfahrzeugführer in der Lage ist, andere Luftfahrzeuge zu erkennen und diesen eigenständig auszuweichen, ist eine Flugverkehrskontrolle für den Sichtflug nicht erforderlich, wenn auch möglich.

Instrumentenflugregeln Beim Instrumentenflug wird die Lage des Luftfahrzeugs im Luftraum nur anhand von Instrumenten bestimmt. Das macht die Flugdurchführung deutlich unabhängiger von Wetter- und Sichtbedingungen sowie der Tageszeit. Daher werden keine Sicht- und Wetterbedingungen in den Regeln definiert, besondere Regeln für den Nachtflug gibt es ebenfalls nicht.

Allerdings ist die benötigte Ausrüstung im Flugzeug deutlich umfangreicher. So werden diverse Instrumente zur Bestimmung der Flugzeuglage benötigt: zwei barometrische Höhenmesser, Fahrtmesser mit Eis- und Kondensationsschutz, Steigmesser, Wendezeiger und ein künstlicher Horizont. Auch die Flugsicherungsausrüstung ist deutlich umfangreicher. Hier bedarf es zwei

UKW-Sprechfunkanlagen, einem Transponder, zwei Empfängern für gerichtete Drehfunkfeuer (VOR), einem automatischen Funkpeilgerät für ungerichtete Funkfeuer (ADF), einem Funkentfernungsmessgerät (DME) und einem Flächennavigationsgerät (B-RNAV/P-RNAV). Bei Flugzeugen über 5.700 kg höchstzulässiger Startmasse ist außerdem noch ein Kollisionsschutzsystem (TCAS) vorgeschrieben. Für besondere Lufträume oder Flugverfahren können noch weitere Instrumente hinzukommen.

Des Weiteren ist es im Instrumentenflug nicht immer möglich andere Luftfahrzeuge zu erkennen, was eine Flugverkehrskontrolle zwingend notwendig macht. Um eine Flugverkehrskontrolle zu ermöglichen, muss für Instrumentenflüge vorab ein Flugplan erstellt und bei der Flugverkehrskontrollstelle eingereicht werden, welche diesen bestätigen muss.

2.2.2 Luftraumstruktur

Der Luftraum ist auf verschiedenste Weise strukturiert, zum einen lateral und vertikal, zum anderen nach Art des Luftraums. Diese Strukturierungen bringen unterschiedliche Zuständigkeiten und Regeln für die Luftverkehrskontrolle und den Luftverkehr mit sich.

Flugflächensystem

Für die vertikale Strukturierung des Luftraums wird das Flugflächensystem verwendet. Als Grundlage dient die barometrische Höhenmessung, bei der anhand des Luftdrucks die Flughöhe errechnet wird. Allerdings ist der Luftdruck in der Atmosphäre nicht konstant, sondern ändert sich mit dem Wetter. Daher ist ein Höhenmesser, der beim Start auf Flughafenhöhe eingestellt wurde, in einer anderen Wetterzone völlig unbrauchbar. Um dieses Problem zu umgehen, wird der Höhenmesser während des Fluges in verschiedenen Modi verwendet. Beim Start und bei der Landung wird der gemessene Druck am Flughafen über Funk mitgeteilt und als Referenzwert am Höhenmesser eingestellt, so dass dieser die tatsächliche Höhe über dem Flughafen anzeigt. Überschreitet ein Luftfahrzeug die sogenannte Transition Altitude, wird der Höhenmesser auf den Druck der Normatmosphäre eingestellt. Ein so eingestellter Höhenmesser zeigt bei Normatmosphäre die Höhe über Normalnull an, bei allen anderen Wetterlagen ist er zur genauen Höhenbestimmung ungeeignet (s. Abb. 2.2). Da allerdings alle Luftfahrzeuge über der Transition Altitude in diesem Modus fliegen, haben sie alle denselben Fehler gegenüber der

tatsächlichen Höhe und können anhand der vom Höhenmesser bestimmten Flughöhe separiert werden.

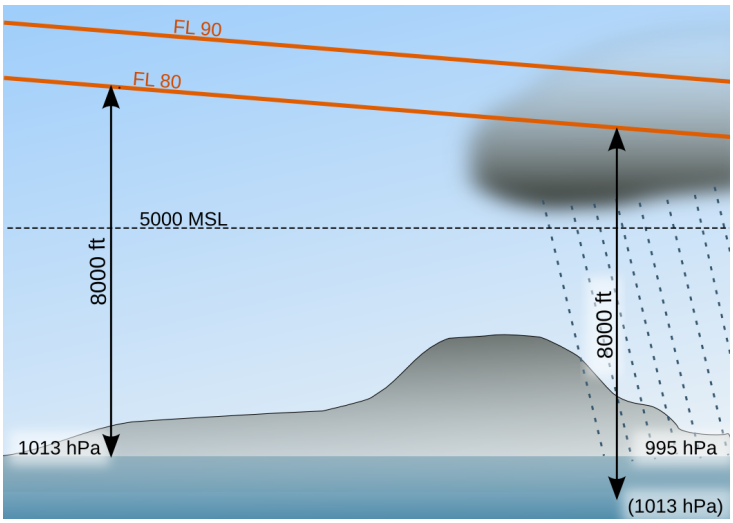

Abbildung 2.2: Flugflächen bei unterschiedlichen Wetterlagen.[1]

Basierend auf diesem Prinzip der Höhenmessung wurde das Flugflächensystem eingeführt. Hierbei wird der Luftraum vertikal in Flugflächen aufgeteilt, die jeweils 500 ft (ca. 152 m) auseinander liegen. Die Flugflächen werden mit der Höhe in Fuß durch 100 bezeichnet, so hat FL 50 beispielsweise eine Flughöhe von 5000 ft (ca. 1.520 m). Die Zuteilung der Flugflächen erfolgt nach den jeweiligen Kursen und Flugregeln der Luftfahrzeuge. So sind die Flugflächen mit ungraden Zehnerstellen für Kurse von 0° bis 179°, jene mit graden Zehnerstellen für Kurse von 180° bis 359° vorgesehen. Weiterhin sind Flugflächen, die auf 5 enden, für Flüge unter Sichtflugregeln reserviert, die Flugflächen mit Endung auf 10 für Instrumentenflüge. Auf diese Weise werden Sicht- und Instrumentenflüge separiert und Luftfahrzeuge auf entgegenkommenden Kursen haben immer mindestens 500 ft, bei zwei Luftfahrzeugen unter Instrumentenflugregeln sogar 1000 ft vertikalen Abstand. Zugewiesen werden die Flugflächen den Luftfahrzeugen von der Luftverkehrskontrolle, die die Einhaltung mittels Sekundärradar kontrollieren kann.

[1] Foto: Kreuzschnabel/Wikimedia Commons, Lizenz: Cc-by-sa-3.0

ATS-Routen

Die laterale Strukturierung des Luftraums erfolgt durch Air Traffic Service Routes (ATS-Routen). Diese ATS-Routen beschreiben einen Verkehrsweg zwischen zwei Kontrollpunkten im Luftraum, klassischerweise durch zwei Navigations-Funkfeuer oder moderner, zwei GPS-Koordinaten identifiziert (s. Abb. 2.3). Eine Flugstrecke setzt sich in der Regel aus mehreren, aneinandergereihten ATS-Routen zusammen.

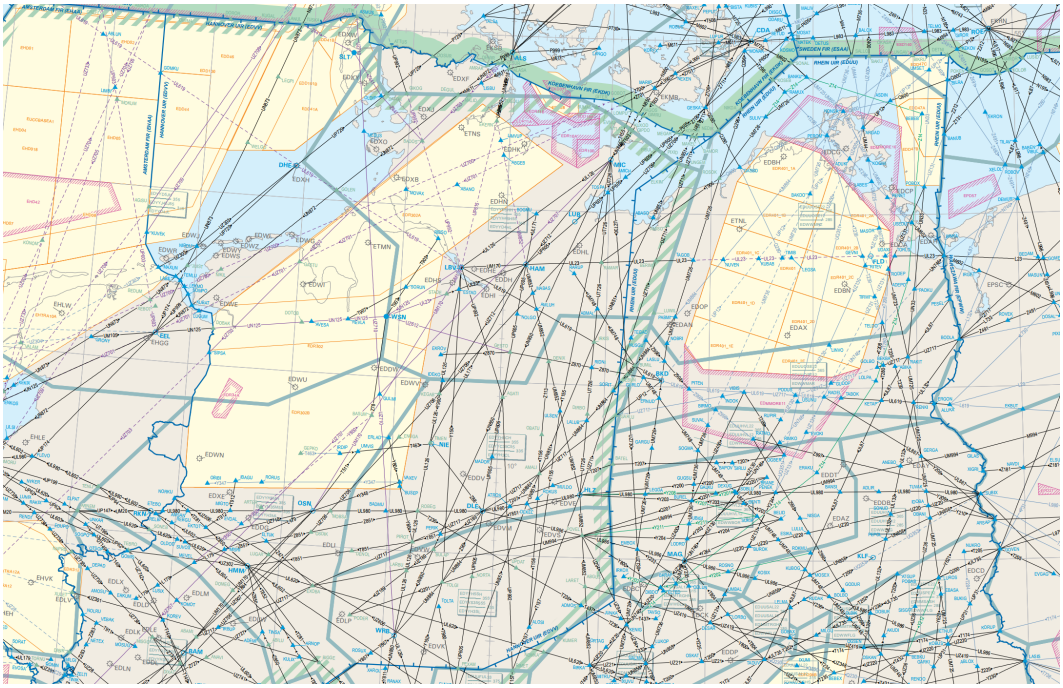

Abbildung 2.3: ATS-Routen über Norddeutschland.[2]

Luftraumklassen

Neben der vertikalen und lateralen Strukturierung wird der Luftraum auch noch in von der ICAO definierte Luftraumklassen eingeteilt (s. Abb. 2.4). Diese Klassifizierung beinhaltet für jede Luftraumklasse bestimmte Regeln, die für Sicht- und Instrumentenflug sowie Flugverkehrskontrolle gelten. Die wichtigste Unterteilung ist hier zwischen kontrollierten und unkontrollierten Luftraumklassen. Außer in Bereichen um Flughäfen ist beispielsweise der Luftraum vom Boden

[2]Foto: © Courtesy of EUROCONTROL

bis zu einer Flughöhe von bis zu 2500 ft (ca. 760 m) in der Regel unkontrollierter Luftraum der Klasse G, das heißt hier findet keine Flugverkehrskontrolle statt. In Bereichen, wo Verkehrsflugzeuge hauptsächlich unterwegs sind, ist der Luftraum normalerweise als Luftraumklasse C oder D klassifiziert. Das bedeutet, dass hier eine Flugverkehrskontrolle stattfindet, eine Freigabe zum Einflug nötig ist und ständige Funkverbindung möglich sein muss.

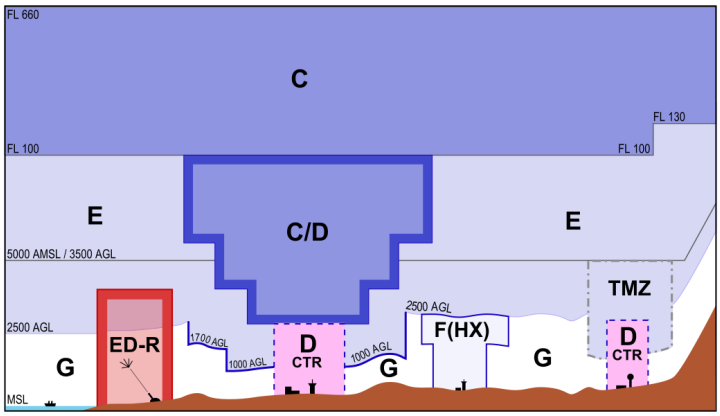

Abbildung 2.4: Luftraumklassen im deutschen Luftraum.[3]

2.2.3 Luftfahrtkommunikation

Zur Durchführung der Flugsicherung bedarf es umfangreicher Kommunikation, hauptsächlich zwischen Luftfahrzeugen und der Flugverkehrskontrolle, aber auch zwischen den Luftfahrzeugen selbst oder mit anderen Organisationen. Dabei kommen sowohl Datenkommunikation als auch gesprochene Kommunikation zum Einsatz. In diesem Abschnitt wird eine Übersicht über verschiedene Kommunikationssysteme in der Luftfahrt gegeben.

Sprechfunk

Kommuniziert wird in der Luftfahrt primär über Sprechfunk im UKW-Frequenzbereich. Beim Flug durch Gebiete ohne UKW-Abdeckung, beispielsweise über Ozeane, wird auch der KW-Frequenzbereich oder Satellitenfunk verwendet. Für Flugverkehrskontrolle und damit für alle

[3] Foto: Philipp Fischer, Lizenz: Cc-by-sa-3.0

Instrumentenflüge ist dauernde Hörbereitschaft vorgeschrieben. Lediglich im Sichtflug in Lufträumen ohne Flugverkehrskontrollpflicht kann auf Sprechfunk verzichtet werden.

Navigationsfunk

Zu Navigationszwecken im Luftfahrzeug werden bodengestützte Funkfeuer verwendet, deren Sendungen allerdings lediglich ihre Bezeichnung enthalten. Je nach Art des Funkfeuers kann anhand von Funkpeilung oder durch zeitbasierte Berechnungen die Peilung zu diesem Funkfeuer festgestellt werden. Wenn ein Funkentfernungsmesser vorhanden ist, kann ebenfalls die Entfernung berechnet werden. Auch Flächennavigation, meist GPS, kann im Luftfahrzeug empfangen werden.

Radarsysteme und Transponder

Neben dem Sprechfunk steht der Luftverkehrskontrolle durch Radarsysteme eine weitere Möglichkeit zur Verfügung, Informationen über den Luftraum und einzelne Luftfahrzeuge zu beziehen. Dabei ist zwischen dem Primärradar und dem Sekundärradar zu unterscheiden. Ein Primärradar sendet Funkpulse aus, analysiert die zurückkommenden Reflektionen und stellt damit die Positionen von Luftfahrzeugen fest. Ein Sekundärradar hingegen verwendet einen sogenannten Interrogator, welcher ein Funksignal an das Luftfahrzeug sendet. Dort wird dieses vom sogenannten Transponder empfangen, der daraufhin eine Antwort mit bestimmtem Informationen zurück sendet. Die Verfahren, die Anfrage- und Antwortsignal definieren, wurden über die Jahre weiter entwickelt. Für die zivile Luftfahrt sind drei Verfahren (Modes) relevant:

Mode A
: Mode A ist der erste zivile Transpondermodus. Angefragt wird in diesem Modus über zwei Funkpulse mit einem definierten Zeitabstand. Daraufhin wird einer von 4096 möglichen Identifizierungscodes zurück gesendet.

Mode C
: Mode C wurde entwickelt, um die Nutzung von zweidimensionalen Primärradarsystemen abzusichern. Der Transponder antwortet auf eine Mode C Anfrage, die analog der Mode A Anfrage aber mit anderem Zeitabstand funktioniert, indem er eine barometrische

Höhenangabe zurücksendet. Diese wird benutzt, um die fehlende Höhenangabe des Radars auszugleichen.

Mode S

Mode S dient dazu, die Unzulänglichkeiten des Mode A/C Systems zu beheben. Zum einen ist Mode A/C ein Broadcast/Broadcast-System, daher empfangen alle Transponder im Empfangsgebiet des Sekundärradars eine Anfrage und antworten darauf, was zu sehr viel Last auf der Sekundärradarfrequenz führt. Außerdem sind die Mode A Identifizierungscodes nicht eindeutig und müssen pro Radargebiet dynamisch zugeteilt und im Luftfahrzeug eingestellt werden, was das Risiko eines Fehlers birgt. Daher führt Mode S eindeutige 24-Bit-Adressen ein, die einem Transponder und damit dem Luftfahrzeug fest zugewiesen werden. Auch das Anfragesystem umfasst jetzt die Möglichkeit Daten zu versenden. Diese Daten enthalten die 24-Bit-Adresse und ermöglichen so das gezielte Abfragen eines bestimmten Transponders. Des Weiteren lassen sich nun im Datenblock der Anfrage Anfragecodes versenden, so dass gezielt verschiedene Parameter des Luftfahrzeugs abgefragt werden können.

Da Radar bodengestützte Infrastruktur benötigt, ist es nicht global verfügbar. In abgelegenen Gebieten und über Meeren gibt es keine Radarabdeckung, eine unterbrechungsfreie Überwachung eines Luftfahrzeugs ist daher nicht möglich.

TCAS

Da es mit Zunahme des Luftverkehrs Ende der fünfziger Jahre vermehrt zu Flugunfällen durch Zusammenstöße in der Luft kam, wurde die Entwicklung eines Systems zur Verhinderung von Flugzeugkollisionen gestartet. Erste Entwicklungen zeigten sich allerdings nicht funktionsfähig oder praxistauglich. Erst 1981 wurde die Entwicklung des Traffic Collision Avoidance System (TCAS) gestartet, welches nach einer weiteren Flugzeugkollision 1986 ab Anfang 1994 in den USA gesetzlich vorgeschrieben wurde.

Zur Kollisionserkennung benötigt ein Luftfahrzeug Daten von anderen Luftfahrzeugen in der Umgebung. Während frühe Systeme versuchten, eigene Übertragungswege zu finden oder mittels Bodensystemen Informationen zu sammeln, setzt TCAS auf die verpflichtend vorhandenen Sekundärradartransponder. Durch die Verwendung der Sekundärradartransponder kann ein TCAS-ausgestattetes Luftfahrzeug jedem transponderpflichtigen Luftfahrzeug ausweichen,

auch wenn dieses kein TCAS einsetzt. Für die Verwendung von TCAS wird ein Luftfahrzeug neben dem vorhandenen Transponder auch mit einem Interrogator versehen. Analog zum Bodensekundärradar werden nun vom Luftfahrzeug aus Anfragesignale versendet, allerdings mit geringerer Signalstärke, so dass ein Umkreis von etwa 75 km erreicht wird. Anhand der Signallaufzeit, der Funkpeilung des Antwortsignals und der Höhenangabe in der Transponderantwort kann so die Position und, durch mehrfaches Anfragen, der Kurs von anderen Luftfahrzeugen berechnet werden. Dieser Kurs wird mit dem Kurs des eigenen Luftfahrzeugs verglichen um zu prüfen ob eine Kollisionsgefährdung vorliegt. Luftfahrzeuge im Umkreis von 40 NM (ca. 75 km) werden wie folgt eingeteilt:

NO COLLISION THREAT
: Luftfahrzeug näher als 40 NM, keine Kollisionsgefahr.

NO COLLISION THREAT: PROXIMATE
: Luftfahrzeug näher als 6 NM (ca. 11 km) und +/- 1200 ft (ca. 365 m), keine Kollisionsgefahr.

TRAFFIC ALERT
: Luftfahrzeug näher als 6 NM (ca. 11 km) und +/- 1200 ft (ca. 365 m), Kollisionsgefahr in maximal 45 Sekunden. Es erfolgt eine akustische Warnung „Traffic, Traffic“, welche den Luftfahrzeugführer auffordert, das andere Luftfahrzeug visuell zu erfassen und dann auf Sicht auszuweichen.

RESOLUTION ALERT
: Luftfahrzeug näher als 6 NM (ca. 11 km) und +/- 1200 ft (ca. 365 m), Kollisionsgefahr in maximal 35 Sekunden. Es erfolgt eine akustische Ausweichempfehlung, etwa „Climb, climb“ oder „Descend, descend“, welche vom Luftfahrzeugführer umgehend ausgeführt werden muss, um eine Kollision zu verhindern.

Während das TCAS I System nur Traffic Alerts unterstützt, gibt TCAS II auch Resolution Alerts aus. Haben beide Luftfahrzeuge TCAS II, werden die Ausweichempfehlungen abgestimmt. Allerdings sind mit dem TCAS II System nur Ausweichempfehlungen über die Steig- bzw. Sinkrate möglich, da die Bestimmung des exakten Kurses zu ungenau ist. Derzeit werden Möglichkeiten erforscht, den Kurs genauer zu bestimmen und so ein angedachtes TCAS III zu entwickeln, das seitliche Ausweichempfehlungen ermöglichen soll.

Trotz des TCAS sind Flugzeugkollisionen durch Anwenderfehler weiter möglich. Zum einen kann das System fehlkonfiguriert oder abgestellt werden, zum anderen kann ein Pilot die Anwei-

sungen des TCAS missachten oder entgegengesetzt ausführen. Letzteres kann beispielsweise passieren, wenn die Anweisungen des TCAS und der Flugverkehrskontrolle sich unterscheiden.

ADS

Automatic Dependent Surveillance (ADS) ist ein Ortungs- und Überwachungssystem, das nach einem neuen Konzept funktioniert. Die bisherigen Ortungs- und Überwachungssysteme, Radar und TCAS, bestimmen die Position eines Objekts im Luftraum, indem sie passive oder aktive Funkantworten mittels Funkpeilung und Laufzeitberechnungen auswerten. Bei ADS hingegen wird die Positionsbestimmung mittels vorhandenen Flächennavigationssystemen von den Luftfahrzeugen selbst durchgeführt und die Position über Datenfunk verbreitet. Vorteil an diesem Konzept ist die höhere Genauigkeit der Positionsbestimmung. Bei ADS kommen zwei Verfahren zur Anwendung. ADS-C (C = contract) beinhaltet, dass eine Kontrollstelle angefordert, gewisse Informationen zu gewissen Zeitpunkten oder in gewissen Intervallen zu senden. ADS-B (B = broadcast) hingegen sendet periodisch eine Meldung mit aktuellen Positionsdaten des Flugzeugs an alle Empfänger im Empfangsbereich. Werden alle über ADS gewonnenen Daten auf einer graphischen Anzeige angezeigt, lassen sich so Luftlagen ähnlich einem Radarbild, sowohl in Luftfahrzeugen als auch bei der Luftverkehrskontrolle, darstellen. Umgesetzt wird der Datenfunk entweder als eigenes Verfahren oder unter Verwendung des Mode-S des Transponder/Interrogator-Systems, welches schon für Sekundärradar und TCAS vorhanden ist.

ACARS

Das Aircraft Communication Addressing and Reporting System (ACARS) ist ein UKW Datenfunksystem. Es dient der Übertragung von Routine- und Wartungsmitteilungen zwischen Luftfahrzeugen und Fluggesellschaften, was zur Entlastung der Sprechfunkfrequenzen nötig ist. Ursprünglich war das Protokoll zur Übertragung der sogenannten OOOI-Nachrichten gedacht, es lassen sich aber auch beliebige andere Nachrichten mit bis zu 220 Zeichen versenden. OOOI steht für Out, Off, On, In. Eine Out-Meldung wird versendet, wenn das Flugzeug seine Parkposition verlässt. Die Off-Meldung wird generiert, wenn das Fahrwerk kein Gewicht mehr registriert, also das Flugzeug abgehoben ist. Die On- bzw. In-Mitteilungen sind analog dazu, On wird bei der Ladung versendet, In beim Erreichen der Parkposition. Diese OOOI-Nachrichten sind wichtig für die Flottenplanungen der Fluggesellschaften. Neben der terristrischen Versen-

dung der ACARS-Nachrichten gibt es auch eine satellitengestützte Übertragung, die Funklöcher über abgelegenen Gegenden abdecken soll [36].

2.2.4 Flugverkehrskontrolle

Die Flugverkehrskontrolle dient dazu, den Flugverkehr so zu leiten, dass ein sicherer und reibungsloser Flugverkehrsablauf gewährleistet ist. Hauptaufgabe hierbei ist, für eine ständige räumliche Separation aller Luftfahrzeuge im kontrollierten Luftraum zu sorgen, um Kollisionen zu vermeiden. Durchgeführt wird die Flugverkehrskontrolle von Menschen am Boden, die mittels Sprechfunk Anweisungen an Luftfahrzeuge geben und über Radar deren Ausführung kontrollieren können. Während Instrumentenflüge immer von der Luftverkehrskontrolle voneinander separiert werden müssen, kann es bei Sichtflügen je nach Luftraumklassifizierung (s. Abschn. 2.2.2) sein, dass die Separation ganz oder teilweise dem Luftfahrzeugführer übertragen wird. Zur Sicherstellung der Separation wird die sogenannte Staffelung angewandt.

Staffelungsverfahren

Um eine vollständige Separation sicherzustellen, muss der Luftverkehr sowohl vertikal als auch lateral gestaffelt werden. Hierzu wird sich der festgelegten Struktur des Luftraums bedient (s. Abschn. 2.2.2). Die (vertikale) Höhenstaffelung wird hierbei durch die Verwendung des Flugflächensystems erreicht, zur Ermöglichung der lateralen Staffelung werden die ATS-Routen verwendet. Grundsätzlich gilt, dass sich keine zwei Luftfahrzeuge zur selben Zeit auf dem selben Flugroutenabschnitt bei gleicher Flugfläche befinden dürfen. Während die Flugflächen durch die Verwendung des Höhenmeters eindeutig sind und die Seitenstaffelung durch größeren Abstand zwischen den ATS-Routen sichergestellt ist, bedarf es zur Längsstaffelung auf den Luftrouten weiteres Aufwandes. Hierbei kommen verschiedene Verfahren zum Einsatz.

Zeitstaffelung Bei der Zeitstaffelung sind auf der Luftroute Meldepunkte definiert, deren Überflug unter Angabe der Überflugzeit per Sprechfunk gemeldet werden muss. Zwei Luftfahrzeuge auf derselben Flugfläche und -route dürfen eine definierte Mindestzeit dabei nicht unterschreiten, um die Staffelung sicherzustellen. Ebenso müssen Mindestzeiten an Routenkreuzungen eingehalten werden.

Entfernungsstaffelung Die Entfernungsstaffelung basiert auf der Verwendung von Funkentfernungsmessern. Für eine Route oder einen Routenabschnitt wird ein Funkfeuer mit Entfernungsmessung bestimmt. Die Luftverkehrskontrolle fragt von allen Luftfahrzeugen auf der Route regelmäßig per Sprechfunk die Entfernungen zu diesem Funkfeuer an. Luftfahrzeuge auf derselben Flugroute und -fläche sowie an Luftroutenkreuzungen dürfen Mindestentfernungen nicht unterschreiten.

Radarstaffelung Steht Radar zur Verfügung, wird die Staffelung ohne Mithilfe der Luftfahrzeuge kontrolliert. Auch hierbei gilt, dass Mindestentfernungen nicht unterschritten werden dürfen. Allerdings sind diese abhängig von der Radarauflösung, welche nahe an der Radarantenne besser ist, als in entfernten Abdeckungsgebieten. Generell sind die Mindestentfernungen der Radarstaffelung deutlich kleiner als die der Entfernungsstaffelung.

Radarstaffelung ist in heutiger Zeit das gängigste Verfahren. Lediglich in Gebieten ohne Radarabdeckung und über Ozeanen wird nach Entfernungs- oder Zeitstaffelung geflogen.

2.3 NASA UTM

Das UAS Traffic Management (UTM) ist ein von der NASA vorgeschlagenes Luftraummanagementsystem für unbemannte Luftfahrzeuge. Die Erforschung und Entwicklung des Systems ist bisher nicht abgeschlossen. Die Entwicklungphase wird in mehreren Schritten durchgeführt und soll voraussichtlich 2019 beendet werden. Mit UTM soll der unkontrollierte Luftraum der Klasse G (s. Abschn. 2.2.2) zwischen 200 bis 500 ft (ca. 60 bis 152 m) verwaltet werden, unter Berücksichtigung von sowohl bemannten wie auch unbemannten Luftfahrzeugen (s. Abb. 2.5). Es wird hierfür die Verbindung von mehreren Systemen vorgesehen.

Für die unbemannte Flugplanung und -durchführung soll ein cloud-basiertes Luftraummanagementsystem entwickelt werden, in welchem Flugrouten geplant und Luftraum reserviert werden kann. Dabei sollen geografische Eigenschaften, andere unbemannte Luftfahrzeuge sowie Wind- und Wetterbedingungen mit in die Routen- und Flugplanung einfließen. Dieses bodengestützte System soll mittels Mobilfunk mit den unbemannten Luftfahrzeugen kommunizieren.

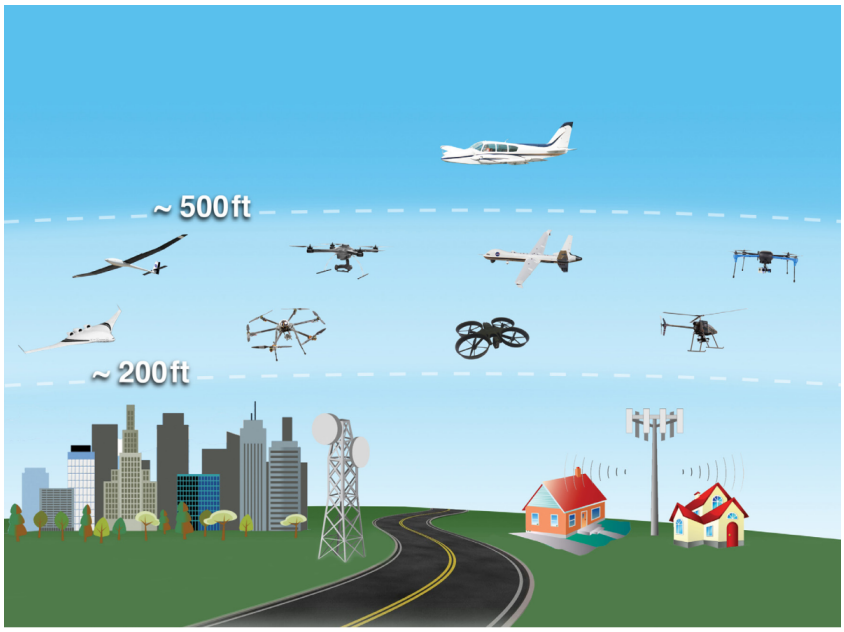

Abbildung 2.5: Schematische Darstellung des UTM-Systems.[4]

Die Separation von bemannten Luftfahrzeugen soll dynamisch und auf Basis bestehender Kommunikationsmöglichkeiten (s. Abschn. 2.2.3) erfolgen. Dazu sollen unbemannte Luftfahrzeuge mit ADS-B Empfängern ausgestattet werden, die es ihnen erlauben, die Positionen von bemannten Luftfahrzeugen zu erfassen. Ähnlich zu TCAS soll dann eine Ausweichlösung für das unbemannte Luftfahrzeug berechnet werden. Im Gegensatz zu TCAS sieht UTM kein Ausweichen von beiden Luftfahrzeugen vor. Analog dazu sollen auch unbemannte Luftfahrzeuge direkt miteinander kommunizieren und so bei Ausfall des Bodensystems sicher weiterfliegen können [23, 18].

2.4 SESAR U-Space

U-Space ist Teil des „Single European Sky ATM Research"-Projekts (SESAR), welches ein EU-Projekt zur Erneuerung des europäischen Luftverkehrsmanagements ist. Es setzt ähnlich wie das UTM Projekt auf eine Cloud-basierte Lösung, mit einem annähernd gleichen Umfang. Allerdings ist der Zeitplan weiter angelegt, so soll 2019 erst ein Grundgerüst aus Registrierung,

[4] Foto: National Aeronautics and Space Administration

Identifizierung und Geofencing bereit stehen. Die weiteren Bestandteile sollen später folgen, ohne dass konkrete Roll-Out-Termine genannt werden [31].

2.5 Automotive Technologies

Viele Probleme, die bei der Umsetzung einer autonomen Navigations- und Flugsicherungsplattform gelöst werden müssen, wurden in ähnlicher Form auch im Bereich der Automobilelektronik schon behandelt oder gelöst. Hierzu zählt die Navigation, die im Bereich der Automobilelektronik für den zweidimensionalen Raum eingesetzt und kontinuierlich weiterentwickelt wird. Auch eine Punkt-zu-Punkt-Kommunikation ist in der Automobilelektronik unter dem Begriff Car2Car-Kommunikation in der Entwicklung. Analog zur Flugverkehrskontrolle im Luftraum werden auch in der automobilen Verkehrsleitung Techniken entwickelt, die einen reibungslosen und sicheren Verkehrsfluss garantieren sollen, insbesondere im Bereich von Straßenkreuzungen.

2.5.1 Navigation

Die Navigation im automobilen Umfeld besteht aus drei Teilaspekten: der Positionsbestimmung, der Routenberechnung und der Zielführung.

Die Positionsbestimmung dient der Feststellung der aktuellen Fahrzeugposition und erfolgt über Satellitenpeilung, zum Beispiel mit dem GPS-System.

Für die Ermittlung der eigentliche Route zum Ziel, wird die sogenannte Routenberechnung eingesetzt. Da ein Straßennetz leicht als Graph repräsentiert werden kann, wird in automobilen Navigationsgeräten ein graphenbasierter Ansatz gewählt. Hierbei werden Kreuzungen als Knoten und Straßen als Kanten interpretiert. Erweitert wird der Graph durch eine Datenbank mit Zusatzinformationen: die tatsächliche geometrische Form einer Straße, Hausnummern, Geschwindigkeitsbegrenzungen, Warnhinweise, Abbiegehinweise und Points of Interests. Diese Informationen werden zur graphischen Darstellung, adressbasierten Zielsuche und zur Optimierung der Routenberechnung benötigt. Die eigentliche Routenberechnung verwendet Algorithmen zur Bestimmung des kürzesten Pfads in Graphen, beispielsweise den Algorithmus von Dijkstra oder dessen Erweiterung A*. Diese Wegfindungsalgorithmen analysieren den Graphen ausgehend vom Startpunkt und bewerten anhand der Kantenkosten die verschiedenen

Verbindungen, um so die optimale Route zu finden. Die Kantenkosten sind hierbei variabel und vom Fahrzeugführer beeinflussbar. Beispielsweise kann eine Route auf die kürzeste Strecke oder die kürzeste Fahrzeit optimiert werden. Der Unterschied zwischen Dijkstra und A* ist hierbei, dass Dijkstra alle unbekannten Knoten nacheinander exploriert, A* hingegen die Knoten anhand einer Heuristik bewertet und jeweils den Knoten als nächstes analysiert, der die beste Bewertung hat. Die Heuristik gibt hierbei die Kosten an, die ein Knoten mindestens noch bis zum Zielpunkt braucht. Damit wird vermieden, dass der Algorithmus Knoten analysiert, die in eine ungünstige Richtung führen.

Der letzte Teil der Fahrzeugnavigation ist die Zielführung. Diese soll es dem Fahrzeugführer anhand von akustischen und/oder visuellen Fahranweisungen ermöglichen, der berechneten Route zu folgen. Um die Fahranweisungen korrekt auszugeben, müssen die berechnete Route analysiert und aktuelle Fahrtparameter berücksichtigt werden. Beispielsweise müssen die Fahranweisungen der Geschwindigkeit angepasst werden, sodass der Fahrer ausreichend Zeit hat, um auf diese reagieren zu können. Auch die Art der Fahranweisung muss genauer definiert werden, da es beispielsweise mehrere Möglichkeiten geben kann, an einer Kreuzung abzubiegen. Grundsätzlich wird anhand der aktuellen Position des Fahrzeugs eine Fahranweisung entlang der aktuellen Kante (Straße) zum nächsten Knoten des Graphen (Straßenkreuzung) gegeben. Bei Erreichen des Knotens wird eine Fahranweisung zum Erreichen der nächsten Kante gegeben, beispielsweise ein Abbiegemanöver. Nachfolgend wird der Vorgang wiederholt, bis das Ziel erreicht wird [27].

2.5.2 Car2Car Kommunikation

Die Kommunikation zwischen verschiedenen Kraftfahrzeugen oder Kraftfahrzeugen und Infrastruktur zum Zweck der Verkehrsleitung, der Verkehrssicherheit sowie für Infotainmentangebote ist ein aktuelles Forschungsthema in der Automobilindustrie. Um eine einheitliche Lösung zu entwickeln und Interoperabilität zwischen allen Fahrzeugen zu gewährleisten, haben sich europäische Automobilhersteller zu einem Car2Car-Konsortium zusammengeschlossen. Dieses Car2Car-Konsortium hat ein Konzept für ein interautomobiles Kommunikationssystem (s. Abb. 2.6) entwickelt [38, 13].

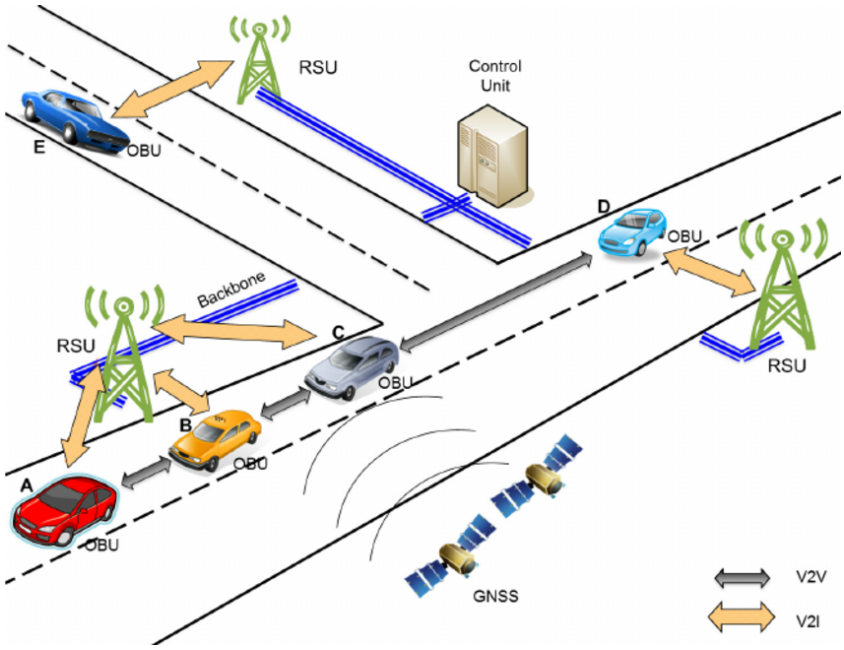

Abbildung 2.6: Schematische Darstellung des Car2Car-Kommunikationssystems.[5]

Das Kommunikationskonzept des Konsortiums sieht sowohl Kommunikation zwischen verschiedenen Kraftfahrzeugen (Car2Car/Vehicle2Vehicle) als auch zwischen Infrastruktur und Kraftfahrzeugen (Car2Infrastructure/Vehicle2Infrastructure) vor. Dazu sind verschiedene Kommunikationseinheiten vorgesehen, On Board Units (OBUs) für Fahrzeuge und Road Side Units (RSUs) für die Infrastruktur. Die Kommunikation zwischen diesen Einheiten soll auf mehrere verschiedene Protokolle aufgeteilt werden, da die Anwendungen der Kommunikation unterschiedliche Prioritäten haben und sich nicht behindern sollen (s. Abb. 2.7). Für Infotainmentinhalte zwischen OBUs und RSUs ist die Verwendung von Mobilfunkprotokollen vorgesehen, beispielsweise GSM, UMTS und LTE. Über diese wird mittels üblicher Internetprotokolle wie TCP/IP kommuniziert. Für Verkehrsleitdienste und Verkehrssicherheitsdienste soll zwischen den OBUs und auch den RSUs ein auf konventionellem WLAN (IEEE 802.11) basierendes Protokoll verwendet werden. Kommuniziert wird hier allerdings unterschiedlich. Die Verkehrsleitdienste werden ebenfalls über TCP/IP abgewickelt, da sie nicht sicherheitskritisch sind und die Fahrsicherheit bei ihrem Ausfall nicht gefährdet ist. Die Verkehrssicherheitsdienste hingegen sind sicherheitskritisch, daher muss hierfür ein eigener Protokollstapel entwickelt werden. Auch im Funksystem des WLANs müssen diese Dienste besonders behandelt werden

[5]Foto: Sensing Traffic Density Combining V2V and V2I Wireless Communications - Scientific Figure on ResearchGate. Available from: https://www.researchgate.net/

und erfordern neben dem normalen Datenkanal einen Sicherheitskanal. Dabei ist noch nicht festgelegt, ob dazu mehrere Sendeempfänger nötig sind oder ein Scheduling des WLANs für Sicherheits- und Datenkanal ausreicht [38].

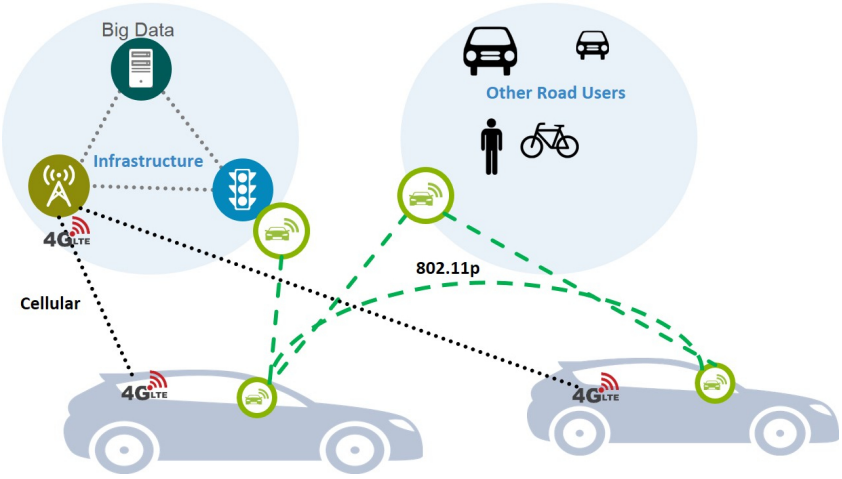

Abbildung 2.7: Kommunikationsprotokolle des Car2Car-Kommunikationssystems.[6]

An das WLAN-basierte Protokoll stellen sich besondere Anforderungen durch die Verwendung im Automobil, die von der vorgesehenen Verwendung als quasi-stationäres Netzwerk abweicht. Die Fahrzeuge bewegen sich sehr schnell, so dass das System für Geschwindigkeiten bis zu 250 km/h ausgelegt werden soll. Daraus ergeben sich relative Geschwindigkeiten von maximal 500 km/h. Das macht die Verwendung von langwierigen Anmelde-, Authentifizierungs- und Abmeldeprozessen unmöglich. Trotzdem müssen Sender sicher identifiziert werden, um Manipulation zu verhindern. Weiterhin gibt es keine zentrale Verwaltungsinstanz im Netz, so dass die Teilnehmer das Netz selbst verwalten müssen. Auch die Fahrzeugdichte kann sehr gering sein, so dass die Teilnehmer des Netzes auch Routingaufgaben übernehmen müssen. Gleichzeitig muss eine Überlastung des Netzes verhindert werden, wozu die Nachrichten beim Routing nur in einem begrenzten Rahmen weitergeleitet werden dürfen. Um diesen Anforderungen der Car2Car Kommunikation gerecht zu werden, wurde der WLAN Standard 802.11 um den Modus 802.11p erweitert und zusätzlich der Standard IEEE 1609 geschaffen. Dadurch wurde die konventionelle WLAN-Schnittstelle zur Nutzbarkeit in hochmobilen Netzen modifiziert und erweitert [38].

[6] Foto: NXP

Das Car2Car-Kommunikationprotokoll für Verkehrssicherheitsdienste sieht zwei verschiedene Nachrichtentypen zum Austausch von Informationen vor. Für die Verkehrssicherheitsdienste werden nur jene Informationen übertragen, die für die Sicherheit der Fahrzeuge nötig sind. RSUs sind für die Übertragung dieser Informationen nicht zwingend nötig, das System funktioniert auch komplett dezentral. Trotzdem können RSUs am Protokoll teilnehmen, entweder um Sicherheitsinformationen weiterzuleiten oder selbst zu verbreiten [38, 13].

Der erste Nachrichtentyp sind die sogenannten Cooperative Awareness Messages (CAMs). Diese Nachrichten werden von allen Teilnehmern des Netzes, also sowohl OBUs als auch RSUs, periodisch versendet. Das Senden erfolgt hierbei in einem Intervall von 0,1-1,0 Sekunden, was einer Frequenz von 1 - 10 Hz entspricht. Der Zweck dieser Nachrichten ist das Mitteilen der eigenen Existenz. Eine CAM umfasst folgende Informationen [38, 13]:

- Header mit Version, Message-Identifier und Zeitstempel.
- Stationskennung mit Position und Stationstyp. Der Stationstyp umfasst unter anderem normale Fahrzeuge, Sonderfahrzeuge, öffentlichen Nahverkehr und feste Stationen.
- Zusätzliche Daten je nach Typ. Fahrzeuge teilen Fahrtgeschwindigkeit, Beschleunigung und Dimensionen mit, Sonderfahrzeuge zusätzlich noch den Einsatz von Sonderrechten. Sind Messdaten mit einer Ungenauigkeit behaftet, werden Genauigkeitsschätzungen mit übertragen.

Bei dem anderen Nachrichtentyp handelt es sich um die Decentralized Environmental Notification Messages (DENMs). Mit diesen Nachrichten soll auf besondere Umgebungssituationen, beispielsweise Stau, Notbremsungen, liegen gebliebene Fahrzeuge oder Geisterfahrer, aufmerksam gemacht werden. Sie enthalten folgende Daten [38, 13]:

- Header mit Version, Message-Identifier und Zeitstempel.
- Einen Update-Counter, mit Zusatzinformationen zu der Anzahl der Wiederholungen, der voraussichtlichen Wiederholfrequenz und eine Time-To-Live Angabe bei ausbleibender Wiederholung.
- Status der gemeldeten Ursache, mit Verlässlichkeitsangabe bei nicht eindeutig erkannten Situationen.

- Situation Container mit Ursache, Bewertung der Schwere der Ursache und eventuell Referenzen zu anderen DENMs, die ergänzt oder bestätigt werden.
- Location Container mit Position der Situation sowie eventuell der eigenen Stationsposition. Es kann nicht nur ein Punkt sondern auch eine Fläche als Position angegeben werden.

Auch die DENMs werden periodisch gesendet, solange die Situation vorliegt. Die Frequenz der Wiederholungen ist hierbei variabel, sie wird vom Versender festgelegt und in den DENMs mitgeteilt [38, 13].

3 Konzept der autonomen Flugsicherung

Dieses Kapitel befasst sich mit dem Konzept des Flugsicherungsverfahrens und den dazugehörigen Teilkomponenten. Dazu zählen die Modellierung des Luftraums, die Strategie zur Separation der unbemannten Luftfahrzeuge, die Kommunikation sowie die Algorithmen zur Kollisionserkennung und -vermeidung.

Da ein rein passives Konzept für die Flugsicherung von unbemannten Luftfahrzeugen nicht praktikabel scheint, soll ein aktiver Ansatz auf Basis eines Kommunikationsprotokolls verfolgt werden (s. Kap. 1). Hierbei stehen zwei konzeptionelle Möglichkeiten offen: eine infrastrukturgestützte zentrale Flugsicherungsstelle, analog zur Flugverkehrskontrolle der konventionellen Flugsicherung (s. Abschn. 2.2.4) und den Projekten UTM (s. Abschn. 2.3) und U-Space (s. Abschn. 2.4), oder infrastrukturunabhängige dezentrale Entscheidungsfindung wie bei der Car2Car-Kommunikation (s. Abschn. 2.5.2) oder, stark limitiert, beim TCAS-System (s. Abschn. 2.2.3).

Der Nachteil des dezentralen Konzepts gegenüber dem Zentralen ist, dass es keine einheitliche Verkehrssteuerung gibt. Eine zentrale Flugsicherung könnte schon vor dem Start eine geplante Route mit anderen geplanten Routen abgleichen und bei Kollisionen den Flug nicht freigeben bzw. eine alternative Route vorgeben. Außerdem wäre durch die zentrale Flugsicherung nahezu immer sichergestellt, dass eine genehmigte Route auch verfügbar ist. Allerdings ist es auch beim zentralen Konzept nötig, dynamisch auf äußere Einflüsse oder Defekte reagieren zu können.

Ein dezentrales System hingegen hätte den Vorteil, dass Single Points of Failure (SPOF) in der benötigten Infrastruktur vermieden werden. Eine zentrale Flugsicherungsstelle könnte ausfallen, genauso wie lokale Basisstationen für die Kommunikation mit den unbemannten Luftfahrzeugen oder weitere Infrastrukturkomponenten. Dynamische Einflüsse spielen bei einem dezentralen System keine besondere Rolle, da die komplette dezentrale Flugsteuerung auf dynamischen Reaktionen basiert. Dies liegt daran, dass bei diesem Konzept andere Luft-

fahrzeuge erst in die Verkehrsplanung mit aufgenommen werden, sobald sie in Funkreichweite sind. Die Funkreichweite ist dabei begrenzt, da nicht unendlich große und schwere Antennen mitgeführt werden können und auch die verfügbare elektrische Leistung begrenzt ist. Daraus resultiert auch, dass ein dezentrales System besser skalierbar ist, da die Last durch mehr Luftfahrzeuge nicht an einem zentralen Punkt anfallen würde sondern sich auf die beteiligten Luftfahrzeuge in Funkreichweite beschränkt. Ein weiterer Vorteil des dezentralen Konzepts ist das einfachere Deployment. Es muss keine Infrastruktur zum Betrieb dieses Systems geschaffen werden. Es reicht aus, die unbemannten Luftfahrzeuge damit auszustatten.

Zwar ist anhand der genannten Argumente nicht eindeutig festzustellen, welches der beiden Konzepte besser geeignet ist. Allerdings gibt es einen leichten Überhang zu der dezentralen Lösung. Besonders das Vermeiden von SPOFs und das einfache Deployment sprechen für diese Variante. Weiterhin ist eine dezentrale Verkehrssteuerung im Bereich der Flugsicherung bisher nur im Sichtflug (s. Abschn. 2.2.1) und beim Notfallsystem TCAS vorzufinden, zentrale Systeme werden aber bereits von der konventionellen Flugverkehrskontrolle als auch den UTM/U-Space Projekten verwendet. Aus diesen Gründen wird für diese Arbeit das dezentrale Konzept gewählt.

3.1 Luftraummodell

Um ein Flugsicherungskonzept zu entwickeln, wird eine Modellierung des Luftraums benötigt, auf der alle weitergehenden Algorithmen arbeiten können. Die Modellierung ist hierbei eng mit der Strukturierung des Luftraums verbunden. In der konventionellen Luftfahrt ist der Luftraum durch die Flugflächen in der Vertikalen gleichmäßig strukturiert, in der Lateralen durch ATS-Routen und Kontrollpunkte eher ungleichmäßig (s. Abschn. 2.2.2). Letzteres liegt daran, dass die ATS-Routen historisch gewachsen sind. Sie wurden nur nach Bedarf und bei gegebenen technischen Voraussetzungen, beispielsweise Funkfeuer (s. Abschn. 2.2.3), angelegt. Eine solch ungleichmäßige Struktur ist für dichteren Luftverkehr ungünstig, da viel Luftraum ungenutzt bleibt.

Da das Anwendungsgebiet des Flugsicherungskonzepts eher in lokal begrenzten Szenarien statt im globalen Luftverkehr liegt (s. Kap. 1), liegt der Fokus bei der Auslegung des Luftraummodells auf kleineren Szenarien. Die Luftraumgröße ist zwar theoretisch uneingeschränkt, allerdings muss das Luftraummodell durch die Verwendung eines dezentralen Ansatzes auf

allen Luftfahrzeugen vorhanden sein und wird daher in der Praxis durch den vorhandenen Speicher limitiert.

Eine autonome Flugsicherung benötigt einen Wegfindungsalgorithmus, um eine Flugroute zu einem Flugziel bestimmen zu können (s. Kap. 1). Daher muss die Modellierung des Luftraums so gewählt werden, dass ein Wegfindungsalgorithmus darauf arbeiten kann. Hierzu gibt es verschiedene Ansätze, von denen zwei in dieser Arbeit näher erläutert und verglichen werden.

3.1.1 Modellauswahl

Ein weit verbreiteter Ansatz für Wegfindungsalgorithmen ist die Verwendung eines Graphen als Basis. In der Graphentheorie sind Algorithmen zur Bestimmung des kürzesten Pfads ein bekanntes Forschungsgebiet. Diese Algorithmen kommen beispielsweise in der Fahrzeugnavigation zum Einsatz, da sich das Straßennetz in einem zweidimensionalen Modell darstellen lässt, welches gut als Graph interpretiert werden kann (s. Abschn. 2.5.1). Eine Interpretation eines dreidimensionalen Luftraums als Graph ist mit einem passenden Modell ebenfalls möglich. So kann der Luftraum in Quader aufgeteilt werden, deren Mittelpunkte als Knoten im Graphen dienen (s. Abb. 3.1). Als Kanten des Graphen dienen im Quader-Modell die Nachbarschaftsbeziehungen [27, 2].

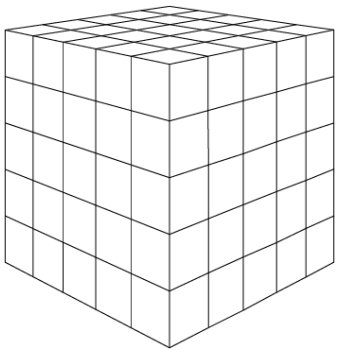

Abbildung 3.1: Luftraum eingeteilt in Quader. Der Mittelpunkt eines jeden Quaders dient als Knoten im Graphen.

Eine weitere Variante zur Luftraumstrukturierung ist ein Modell auf Basis von Octrees (s. Abb. 3.2). Bei dem Octree-Modell wird der Luftraum ebenfalls in Quader aufgeteilt, diese sind allerdings hierarchisch aufgebaut. Es gibt einen großen Quader über den gesamten Luftraum,

der in acht weitere Quader aufgeteilt ist, die wiederrum acht Subquader haben. Die Tiefe der Hierarchie lässt sich hierbei beliebig festlegen. Navigiert werden kann auf einem solchen Modell beispielsweise mittels eines Wegfindungsalgorithmus basierend auf der Potentialfeldtheorie [26, 2].

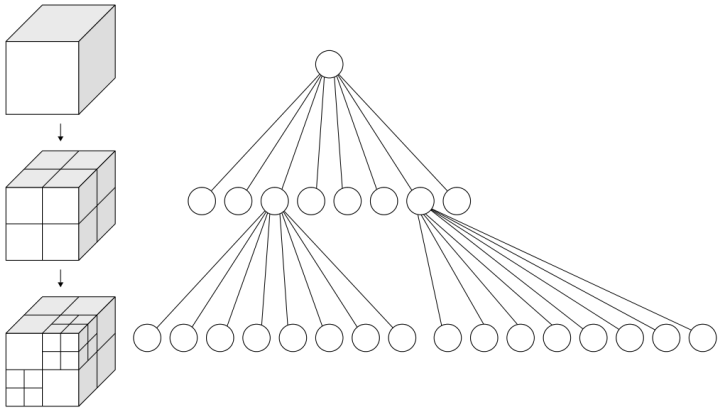

Abbildung 3.2: Luftraummodellierung mit Octrees.[1]

Ein Modell mit einem Graphen als Basis bietet den Vorteil, dass es sehr viele und auf unterschiedliche Szenarien ausgelegte graphenbasierte Wegfindungsalgorithmen gibt. Diese Algorithmen lassen sich bei Verwendung eines solchen Modells ohne größeren Aufwand für die Implementierung des Konzeptes übernehmen. Im Gegensatz dazu ist die Auswahl bei dem Octree-Modell begrenzt und der Potentialfeldalgorithmus komplex und rechenintensiv. Für die Modellierung heterogener Lufträume ist das Octree hingegen im Vorteil, da durch die hierarchische Strukturierung eine variable Granularität gegeben ist. So können Luftraumbereiche mit höherem Verkehrsaufkommen mit kleineren Segmenten modelliert werden und erlauben so einen dichteren Verkehr. Bereiche, die nicht so viel Verkehrslast haben, lassen sich mit größeren Segmenten abbilden, so dass hier Ressourcen gespart werden können. Einfache Luftraummodelle, die einen Graphen als Grundlage verwenden, haben hingegen eine einheitliche Granularität im ganzen Modell. Nur mit erhöhtem Entwicklungsaufwand lassen sich hier Modelle entwickeln, die eine variable Granularität bieten [27, 26, 2].

Da das Anwendungsszenario dieser Arbeit einen räumlich begrenzten, relativ homogenen Luftraum vorgibt, ist der größte Vorteil des Octree-Modells, die variable Granularität, für diese

[1] Foto: Nü/Wikimedia Commons, Lizenz: Cc-by-sa-3.0

Arbeit nicht relevant. Die Möglichkeit, aus spezialisierten bestehenden Wegfindungsalgorithmen auswählen zu können, ist hingegen für die weitere Entwicklung von Bedeutung. Daher wird für diese Arbeit ein Modell auf Basis eines Graphen vorgezogen.

3.1.2 Implementierungsansatz

Der Ansatz eines graphenbasierten Luftraummodells ist die Aufteilung des Luftraums in Quader, welche Gleichzeitig als Knoten des Graphen fungieren (s. Abb. 3.1). Um die Quader optimal im Speicher anzulegen und effizienten Zugriff auf sie zu gewähren, bedarf es einer Verwaltungsstruktur.

Als Verwaltungsstruktur bieten sich zwei naheliegende Möglichkeiten an. Eine Möglichkeit ist die Verwendung eines Voxelspaces zum Verwalten der Quader, die andere ein Modell auf Basis von verketteten Quadern. Ein Voxelspace unterteilt einen dreidimensionalen Raum in gleich große, würfelförmige Voxel, analog zu Pixeln in zweidimensionalen Bildern. Abgelegt werden diese Voxel in einem dreidimensionalen Array. Bei einem verlinkten Modell werden die Quader ungeordnet abgelegt und beinhalten jeweils Verweise auf ihre Nachbarn [14, 2].

Da das Luftraummodell bei einer dezentralen Flugsicherungslösung auf den unbemannten Luftfahrzeugen implementiert werden muss, sollte ein möglichst optimales Modell verwendet werden. Es sollte sowohl einen geringen Speicherbedarf haben als auch effizient in der Verwendung sein [2].

Vergleicht man den Speicherbedarf beider Modelle ohne weitere Optimierung, schneidet das verlinkte Modell besser ab. Das liegt daran, dass beim verlinkten Modell Quader mit statischen Hindernissen und No-Fly-Zonen nicht mit gespeichert werden müssen sondern die Verweise der Nachbarn auf diese Quader einfach leer bleiben können. Leere Verlinkungen werden dann als „uneindringbarer Raum“ interpretiert. Allerdings kann man das Voxelspace-Modell optimieren, so dass im dreidimensionalen Array lediglich Verweise auf Quader angelegt werden. Dann können Quader mit statischen Hindernissen und No-Fly-Zonen ebenfalls weggelassen werden. Nach dieser Optimierung hängt der bessere Speicherbedarf vom Szenario ab. Bei einem Luftraum mit wenig No-Fly-Zonen schneidet das Voxelspace-Modell besser ab, bei einem Luftraum mit vielen No-Fly-Zonen das verlinkte Modell. Ursächlich dafür ist, dass im Voxelmodell immer genau so viele Verweise angelegt werden wie es maximal Quader gibt. Im

verlinkten Modell werden hingegen n Verweise pro wirklich vorhandenem Quader angelegt, wobei n die Anzahl der möglichen Nachbarn ist [2].

Bei der Bewertung der Effizienz stehen zwei für die Flugsicherung und die Wegfindung relevante Anwendungsfälle im Fokus: das Finden eines Quaders anhand von Koordinaten und das Finden der Nachbarn eines Knotens [2].

Beim Finden von Nachbarn eines Quaders sind beide Modelle ähnlich effizient. Im verlinkten Modell ist ein Verweis auf den Nachbarn vorhanden, beim Voxelspace muss der entsprechende Array-Index um eins verringert oder erhöht werden. Daraus ergibt sich bei beiden Modellen eine Laufzeit von $\mathcal{O}(1)$. Im Vergleich hierzu hätte ein Modell auf Basis von Octrees eine Laufzeit von $\mathcal{O}(\log n)$ gehabt, wobei n die Anzahl der Quader ist [26, 2].

Die Effizienz beim Finden von Quadern anhand von Koordinaten unterscheidet sich hingegen stark. Durch die einheitliche Größe der Voxel lässt sich im Voxelmodell, anhand der Kantenlänge der Voxel und einem möglichen Offset zwischen Array- und Koordinatennullpunkt, aus den Koordinaten der Arrayindex ausrechnen. So liegt auch hier die Laufzeit bei $\mathcal{O}(1)$. Beim verlinkten Modell müssen hingegen mit Hilfe eines Suchalgorithmus alle Knoten durchsucht werden. Daher beträgt die Laufzeit hier $\mathcal{O}(n)$, wobei n als Anzahl der Quader definiert ist [2].

Basierend auf dieser Untersuchung wird ein Implementierungsansatz auf Basis eines Voxelmodells gewählt. Welches Modell im Speicherbedarf optimaler ist, lässt sich dabei ohne Kenntnis des zu modellierenden Luftraums nicht bewerten. Ausschlaggebend für die Auswahl ist daher die deutlich bessere Effizienz des Voxelmodells beim Finden von Quadern anhand von Koordinaten.

3.1.3 Bedeutung des Luftraummodells für das System

Das Luftraummodell hat als grundlegende Datenstruktur großen Einfluss auf das Gesamtkonzept. Für große Teile des weiteren Systems ist hierbei die Größe der Quader besonders wichtig.

In der Grapheninterpretation dient es als Grundlage für den Wegfindungsalgorithmus, der aus den Kanten des Graphen einen Flugpfad bestimmt. Der Wegfindungsalgorithmus muss allerdings die physikalischen Eigenschaften des Luftfahrzeugs berücksichtigen. Ist nun der Luftraum sehr fein strukturiert, also die Quadergröße gering, und das Luftfahrzeug relativ

träge, gibt es im Graphen möglicherweise Kombinationen von Kanten, die das Luftfahrzeug physikalisch nicht entlang fliegen kann. Um diese Kombinationen zu verhindern, müsste der Wegfindungsalgorithmus die Auswahl der Kanten einschränken, die an einem Knoten als Nachfolgekanten gewählt werden können. Ist das Modell hingegen relativ grob strukturiert und das Luftfahrzeug klein und wendig, wird es möglichweise zu sehr viel weiteren Strecken gezwungen als eigentlich nötig wären.

Einen weiteren Zusammenhang gibt es zwischen der Quadergröße und der Luftfahrzeuggröße. Ist die Quadergröße zu klein dimensioniert, können größere Luftfahrzeuge größer sein als der Quader. Damit würden Luftfahrzeuge, die auf benachbarten, parallelen Kanten unterwegs sind, möglichweise zusammenstoßen. Zusätzlich müssen auch hier die physikalischen Eigenschaften der Luftfahrzeuge berücksichtigt werden. Die Kanten des Graphen stellen eine ideale Verbindung zwischen zwei Quadermittelpunkten dar, allerdings sind einige Luftfahrzeuge möglichweise physikalisch nicht in der Lage, eine solche Ideallinie zu fliegen, oder weichen bei Richtungsänderungen an den Knoten von der idealen Flugbahn ab. Daher können sogar Luftfahrzeuge, deren Größe eigentlich geringer ist als die Größe der Quader, unter ungünstigen Bedingungen mit Luftfahrzeugen auf benachbarten Kanten zusammenstoßen. Besteht die Möglichkeit, dass Luftfahrzeuge die Quadergrenzen überschreiten, müssen der Kollisionserkennungs- und der Kollisionsvermeidungsalgorithmus entsprechend auch die benachbarten Kanten mit einbeziehen.

Um den Umfang der Arbeit einzugrenzen, wird bezüglich der Quadergröße eine Festlegung getroffen: die Quader müssen immer mindestens so groß gewählt werden, dass alle teilnehmenden Luftfahrzeuge Manöver zum Folgen aller möglichen Kantenkombinationen fliegen können, ohne die Quader dabei zu verlassen. Auf diese Weise wird sowohl eine Anpassung des Wegfindungsalgorithmus an die physikalischen Eigenschaften des Luftfahrzeugs als auch eine Erweiterung der Kollisionsvermeidung auf benachbarte Kanten vermieden. Nach oben wird die Größe nicht begrenzt, so kann das Modell weiterhin an verschiedene Einsatzzwecke angepasst werden.

3.2 Separation

Unter Separation versteht man die ständige räumliche Trennung von Fahrzeugen. Im Automobilbereich wird das beispielsweise an Kreuzungen durch Ampeln realisiert, im Zugverkehr mittels Signalen und im konventionellen Luftverkehr durch Staffelungsverfahren (s. Abschn.

2.2.4). Alle diese Verfahren basieren darauf, dass eine bestimmte Ressource (Streckenabschnitt, Kreuzung, Flugfläche und ATS-Route) immer nur von einem Fahrzeug zur Zeit belegt werden darf. Davon abgeleitete Konzepte in der Informatik sind Semaphoren und Mutexe [10, 19, 35].

Für die Separation im Flugsicherungskonzept für unbemannte Luftfahrzeuge dient das Luftraummodell als Basis. Im Flugsicherungskonzept ist ein Quader des Luftraummodells eine Ressource, die nur von einem unbemannten Luftfahrzeug zur Zeit belegt werden darf. Um die Separation sicherzustellen darf ein Quader niemals von mehr als einem Luftfahrzeug zur Zeit durchflogen werden. Da das System dezentral ist, kann eine Synchronisation nicht anhand eines globalen Mutexes erfolgen, sondern muss mittels Kommunikation und anhand von Regeln zwischen den Luftfahrzeugen ausgehandelt werden.

3.3 Kommunikation

Die Kommunikation der Luftfahrzeuge untereinander ist ein zentraler Bestandteil des Flugsicherungskonzepts. Durch die Auswahl eines dezentralen Ansatzes muss jedes unbemannte Luftfahrzeug für sich in der Lage sein, eigenständig die Verkehrssituation im Umkreis nach den Regeln des Flugsicherungskonzepts zu bewerten. Dazu muss mittels eines Kommunikationsprotokolls sichergestellt werden, dass alle dafür notwendigen Informationen jederzeit zur Verfügung stehen.

3.3.1 Verfahren

Für die Verteilung der Informationen gibt es grundsätzlich zwei Möglichkeiten: das dauerhafte Anfragen von Informationen bei allen Luftfahrzeugen im Umkreis und das dauerhafte Mitteilen von eigenen Informationen. Ersteres wird beim TCAS (s. Abschn. 2.2.3) verwendet, zweiteres kommt bei der Car2Car-Kommunikation (s. Abschn. 2.5.2) zum Einsatz. Allerdings basiert bei TCAS die Entscheidung, Informationen aktiv abzufragen, darauf, dass das vorhandene Transponder-System genutzt werden sollte. Für eine Neuentwicklung hingegen ergibt die Verwendung dieses Ansatzes weniger Sinn. Das liegt einerseits daran, dass ein Overhead durch die Anfrage-Nachrichten zusätzlich zu den Informationsnachrichten entsteht. Andererseits ist das System weniger ausfallsicher, da es nur funktionieren kann wenn an beiden Teilnehmern sowohl Empfänger als auch Sender funktionieren. Bei einem System mit periodischem Senden von Informationen kann ein Luftfahrzeug auch bei ausgefallenem Sender noch Informationen

empfangen und in einem Notbetrieb allen anderen Luftfahrzeugen ausweichen. Aus diesen Gründen wird ein Kommunikationsprotokoll auf Basis des Car2Car-Protokolls entwickelt.

3.3.2 Anforderungen an den Kommunikationsstapel

Die Entwicklung eines Kommunikationsstapels inklusive physikalischer Übertragung ist nicht Ziel dieser Arbeit (s. Kap. 1). Daher wird eine UAV2UAV-Kommunikation mit gewissen Eigenschaften vorausgesetzt. Hierbei sind viele Eigenschaften analog zum Car2Car-Protokollstapel (s. Abschn. 2.5.2). Es muss eine sichere Kommunikation inklusive Authentifizierung auch bei hohen relativen Geschwindigkeiten möglich sein. Eine zentrale Netzverwaltung soll es nicht geben. Die Kommunikation darf nicht durch anderen Netzwerkverkehr behindert werden. Stationäre Sender dürfen an der Kommunikation teilnehmen. Nachrichten der UAV2UAV-Kommunikation müssen eine beschränkte Reichweite haben, um eine Überlastung des Systems bei vielen Luftfahrzeugen zu verhindern. Umsetzbar ist das beispielsweise durch die physikalische Reichweite eines Funksystems. Ein Multi-Hop-Routing der Nachrichten ist ebenfalls möglich, allerdings müssen die Nachrichten dann mit einer Time-To-Live oder einem maximalen Hop-Count ausgestattet sein, um die Reichweite einzugrenzen. Gleichzeitig muss die Reichweite immer mehr als doppelt so groß sein wie der maximale Anhalteweg aller teilnehmenden Luftfahrzeuge. Damit wird sichergestellt, dass alle Luftfahrzeuge auch im Gegenanflug in der Lage sind, rechtzeitig auf Kollisionssituationen zu reagieren.

3.3.3 Protokoll

Das Kommunikationsprotokoll des Flugsicherungskonzepts sieht als Teilnehmer sowohl unbemannte Luftfahrzeuge als auch bodengebundene Stationen vor. Diese können entweder zum Zweck der reinen Informationsverteilung dienen und beispielsweise vor Gefahren im Luftraum warnen oder selbst Verkehrsteilnehmer sein. Letzteres soll es ermöglichen, beispielsweise Fahrzeuge auf Landestellen, Kräne, die im Luftraum operieren, oder andere Fahrzeuge in die Flugsicherung einzubeziehen. Zum Informationsaustausch sieht das Kommunikationsprotokoll des Konzeptes zwei Nachrichtentypen vor, die an das Konzept der Car2Car-Kommunikation (s. Abschn. 2.5.2) angelehnt sind.

Der erste Typ sind sogenannte Flight Awareness Messages (FAMs). Diese Nachrichten basieren auf den CAMs der Car2Car-Kommunikation und dienen dazu, Informationen von mobilen

Teilnehmern des Flugsicherungssystems zu verbreiten und so die Separation zu ermöglichen. Versendet werden FAMs periodisch. Die Frequenz kann dabei, unter Berücksichtigung der maximal möglichen relativen Geschwindigkeit und der Funkreichweite im konkreten Anwendungsfall, variiert werden. Die Nachrichten umfassen folgende Informationen:

- Header mit Version, Nachrichtentyp, eindeutigem Message-Identifier und Zeitstempel.
- Eindeutige Stationskennung mit Stationstyp.
- Geplanter Pfad, bestehend aus einer Abfolge von Quadern mit jeweils einer geschätzten Eintritts- und Austrittszeit. Die Zeiten müssen dabei mit einem Sicherheitsaufschlag versehen werden, so dass der Quader garantiert nur in diesem Zeitraum belegt ist. Der erste Quader ist dabei die aktuelle Position des Senders. Eindeutig identifiziert werden die Quader über ihre Koordinaten.
- Priorität der Bewegung.

Anhand des Stationstyps und der Priorität kann ein Ausweichalgorithmus bestimmen, welches Luftfahrzeug ausweichpflichtig ist. Damit soll sichergestellt werden, dass beispielsweise unbemannte Luftfahrzeuge mit Energiemangel Vorrang haben oder stationären Stationen immer ausgewichen wird.

Als zweiter Nachrichtentyp sind die Airspace Awareness Messages (AAMs) angedacht. Basierend auf den DENMs des Car2Car-Konzepts dienen diese Nachrichten dazu, dynamische Informationen über Teilbereiche des Luftraums zu verbreiten. Dabei kann es sich beispielsweise um Verkehrsaufkommen oder Gefahren in bestimmten Quadern handeln. Sie beinhalten folgende Informationen:

- Header mit Version, Nachrichtentyp, eindeutigem Message-Identifier und Zeitstempel.
- Eindeutige Stationskennung mit Stationstyp.
- Einen Update-Counter, mit Zusatzinformationen zu der Anzahl der Wiederholungen, der voraussichtlichen Wiederholfrequenz und eine Time-To-Live Angabe bei ausbleibender Wiederholung.
- Status der gemeldeten Ursache, mit Verlässlichkeitsangabe bei nicht eindeutig erkannten Situationen.

- Ursache mit Bewertung der Schwere.
- Liste der betroffenen Quader.

Die AAMs haben im Flugsicherungskonzept keinen Einfluss auf die Separation, sondern dienen lediglich zur Situationsmeldung. Allerdings kann je nach Situation eine Änderung der Flugroute nötig sein, beispielsweise wenn die aktuelle Route viel Verkehr hat oder blockiert ist.

3.3.4 Datenverwaltung

Alle Informationen, die über die Kommunikationsschicht empfangen werden, werden im lokalen Luftraummodell der unbemannten Luftfahrzeuge abgelegt. Dazu müssen in den Quadern Informationen über temporäre oder dauerhafte Blockierungen gespeichert werden. Um alle Informationen über die Hintergründe der Blockierung des Quaders bereitstellen zu können, muss auf die jeweils zugehörige Nachricht referenziert werden. Die Nachrichten werden deshalb einmal in der lokalen Kommunikationsverwaltung zwischengespeichert. Hierbei wird bei periodischen Nachrichten jeweils immer nur die neueste Nachricht der Reihe gespeichert.

3.4 Kollisionserkennung

Das ausgewählte dezentrale Flugsicherungskonzept sieht es nicht vor, dass zu Beginn eines Fluges alle Informationen zu anderen Luftfahrzeugen im Luftraum vorliegen. Weiterhin können auch unvorhersehbare externe Einflüsse auftreten, die eine geplante Route nachträglich blockieren. Daher bedarf es eines Algorithmus, der mögliche Kollisionen auf der Flugroute erkennen kann.

Um eine Kollision zu erkennen, werden die Quader der eigenen geplanten Route sequenziell im Luftraummodell nachgeschlagen und analysiert. Hat einer der Quader eine Blockierung in der Zeitspanne, wo er durch die eigene Bewegung belegt sein müsste, liegt eine Kollision vor.

Die maximale Laufzeit des Algorithmus liegt bei $\mathcal{O}(n * m)$, wobei n als Anzahl der Quader in der geplanten Route und m als Anzahl der Blockierungen pro Quader definiert ist. Allerdings ist die tatsächliche Laufzeit eher mit $\mathcal{O}(n)$ zu veranschlagen, da die Anzahl der Blockierungen m nicht bei jedem Quader identisch ist und, aufgrund der eingeschränkten Reichweite des

Systems, bei der Mehrzahl der Quader lediglich bei 0 oder 1 liegen sollte. Allerdings kann eine zu groß gewählte Reichweite in Verbindung mit einem hohen Verkehrsaufkommen an dieser Stelle zu Performance-Problemen führen.

Der Kollisionserkennungsalgorithmus wird sowohl vor dem Start eines Fluges als auch nach jeder Aktualisierung des Luftraummodells ausgeführt. Nach dem Flugsicherungskonzept erfolgt eine Aktualisierung immer dann, wenn eine Nachricht über die UAV2UAV-Kommunikation eingegangen ist. Allerdings ist es möglich, die Kollisionserkennung in Zukunft auch mit weiteren Systemen zu verbinden. So könnten Daten aus Radar- oder Kamerasystemen ebenfalls im Luftraummodell abgelegt werden und so in die Kollisionserkennung einfließen.

3.5 Kollisionsvermeidung

Wird eine Kollision erkannt, muss eine Lösung für diese Konfliktsituation gefunden werden. Hierzu wird ein fester Ablauf vorgeschrieben.

Da nicht garantiert ist, dass das Verfahren zur Kollisionsvermeidung eine Lösung findet, und es bei Luftfahrzeugen zu hohen relativen Geschwindigkeiten kommen kann, muss aus Gründen der Sicherheit als erster Schritt der Kollisionsvermeidung der Flug unterbrochen werden. Dazu wird im nächsten Quader der aktuellen Flugroute ein Wartemanöver durchgeführt. Je nach Art der beteiligten Luftfahrzeuge kann es sich bei dem Manöver beispielsweise um Warteschleifen oder einen Schwebeflug handeln, was bei der Dimensionierung der Quader berücksichtigt werden muss.

Noch während dem Einleiten des Wartemanövers wird als zweiter Schritt der Kollisionsvermeidungsalgorithmus aktiv, der eine Lösung für die Auflösung des Kollisionsszenarios sucht. Handelt es sich bei der Blockierung der Route um unbewegte Einflüsse, die mittels AAMs gemeldet wurden, reicht eine einfache Neuplanung der Route. Dabei werden alle weiteren bekannten Luftfahrzeuge im Luftraum automatisch mit berücksichtigt, da ihre Informationen im Luftraummodell abgelegt sind. Handelt es sich allerdings um andere bewegliche Teilnehmer am Flugsicherungskonzept darf nicht unkoordiniert gehandelt werden. Andernfalls könnten Ausweichmanöver geflogen werden, die sich gegenseitig aufheben und es kommt trotz des Ausweichversuchs zu einer Kollision. Daher wird ein Konzept für eine kooperative Konfliktlösung benötigt.

Der letzte Schritt, das Fortsetzen des Flugs, kann nur durchgeführt werden, wenn der Kollisionsvermeidungsalgorithmus erfolgreich eine Lösung gefunden hat. Andernfalls muss die Sequenz aus Kollisionserkennung und -vermeidung periodisch erneut ausgeführt werden, um entweder das Auflösen der Konfliktituation zu erkennen oder eine kollisionsfreie, alternative Flugroute zum Zielpunkt zu finden. Da auch vor dem Start eines Fluges die Kollisionserkennung durchgeführt wird, ist auf diese Weise garantiert, dass niemals eine Flugroute mit erkennbarer Kollisionsstuation beflogen wird.

3.5.1 Konzeptauswahl

Um eine kooperative Konfliktlösung zu erreichen, sind auf Basis der UAV2UAV-Kommunikation (s. Abschn. 3.3) zwei Verfahren denkbar. So könnten sich die Luftfahrzeuge abstimmen. Dafür würden sie bei erkannter Kollision das andere Luftfahrzeug mittels eines weiteren Nachrichtentyps informieren, dass sie eine Kollision erkannt haben und entweder zum Ausweichen auffordern oder selbst ein Ausweichmanöver ankündigen. Die Alternative wäre ein passives Verfahren auf Basis von eindeutigen Ausweichregeln, in denen festgelegt ist, welches Luftfahrzeug wohin auszuweichen hat.

Vorteil des abgestimmten Verfahrens ist, dass eine Situation, in der kein Luftfahrzeug ausweicht, deutlich unwahrscheinlicher ist, da schon im ersten erfolgreichen Nachrichtenaustausch die Zuständigkeit geklärt wird. Allerdings ist das mit deutlichem Mehraufwand und Reaktionszeitverlusten verbunden, da vom Protokoll dann Konfliktsituationen in der Kommunikation, beispielsweise gleichzeitig gesendete Initialnachrichten, gelöst werden müssen. Ein weiterer Verlust an Reaktionszeit entsteht dadurch, dass ein Manöver erst eingeleitet werden kann, wenn die Aushandlung über den Kommunikationsweg erfolgreich war, da ansonsten wieder Manöver drohen, die sich aufheben. Nachteilig ist auch, dass dieses System ebenfalls nur bei vollständiger Kommunikation in beide Richtungen funktioniert. Ein Notbetrieb nur mit Empfänger ist wegen fehlender Abstimmung nicht möglich.

Aus den genannten Gründen wird ein Verfahren mit eindeutigen Regeln ohne Abstimmungsverfahren angestrebt. Hierbei muss allerdings sichergestellt werden, dass diese Regeln immer eindeutig sind und es nicht zu einer Situation kommen kann, in der kein Luftfahrzeug ausweicht oder beide Luftfahrzeuge gleichartig ausweichen.

3.5.2 Kollisionsszenarien

Als Grundlage zum Aufstellen sinnvoller Ausweichregeln muss als erstes ein Überblick über alle Kollisionsszenarien, die im Luftraummodell möglich sind, gewonnen werden. Auf Basis dieser Kollisionsszenarien und den Ausweichregeln im Flugverkehr (s. Abschn. 2.2.1) werden dann Regeln aufgestellt. Diese Regeln sollen die Zuständigkeit für ein Ausweichmanöver und die Art des Ausweichens eindeutig festlegen.

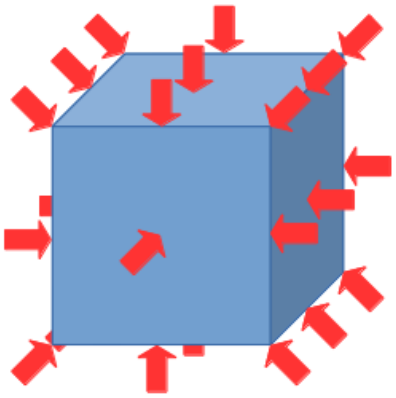

Abbildung 3.3: Alle 26 Einflugrichtungen aus den benachbarten Quadern.

Um alle Kollisionsmöglichkeiten zu bestimmen, müssen alle möglichen Flugbewegungen bekannt sein. Die möglichen Flugbewegungen im Luftraummodell entsprechen den Kanten des übergelegten Graphenmodells, welche wiederrum den Nachbarschaftsbeziehungen der Quader entspricht. Bezieht man die Nachbarn über die Kanten und Ecken des Quaders mit ein kommt man so auf 26 Einflugrichtungen für einen Quader (s. Abb. 3.3).

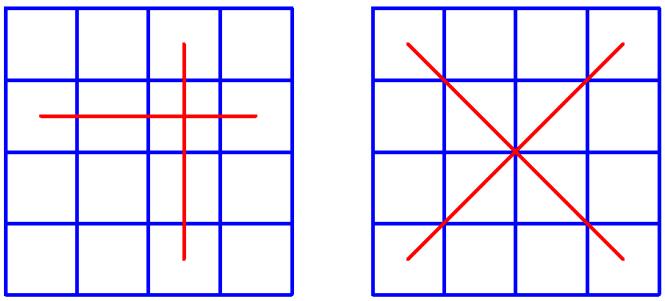

Abbildung 3.4: Mögliche Arten des Zusammentreffens, zweidimensional dargestellt: links in einem Quader, rechts ohne gleichzeitiges befliegen eines Quader.

Aus diesen Einflugmöglichkeiten ergeben sich zwei Basisszenarien für mögliche Kollisionen: das Zusammentreffen zweier unbemannter Luftfahrzeuge in einem Quader, und das Kreuzen zweier unbemannter Luftfahrzeuge, ohne das diese einen gemeinsamen Quader durchfliegen (s. Abb. 3.4).

Ein Sonderfall des Zusammentreffens in einem Quader stellt das mehrfache Zusammentreffen zweier unbemannter Luftfahrzeuge in direkt aufeinander folgenden Quadern dar. In diesem Szenario folgen beide Luftfahrzeuge der selben Route und ein einfaches Ausweichen in einem Quader ist hier nicht zielfürend. Daher wird dieser Fall wie ein weiteres, eigenständiges Szenario behandelt.

Zusammentreffen in einem Quader

Die Kollision in einem Quader würde immer dann auftreten, wenn ein Luftfahrzeug in einen Quader einfliegen würde, in dem sich bereits ein zweites Luftfahrzeug stationär befindet oder dieses ebenfalls einfliegt. Während das erste Szenario recht eindeutig zu erkennen und in einer Regel zu behandeln ist, gibt es bei dem zweiten Szenario die Schwierigkeit, dass es 26 Möglichkeiten gibt, in einen Quader einzufliegen. Bei zwei einfliegenden Luftfahrzeugen ergeben sich so 52 Unterszenarien. Um nicht für alle 52 Unterszenarien eigene Regeln aufstellen zu müssen, muss diese Anzahl reduziert werden.

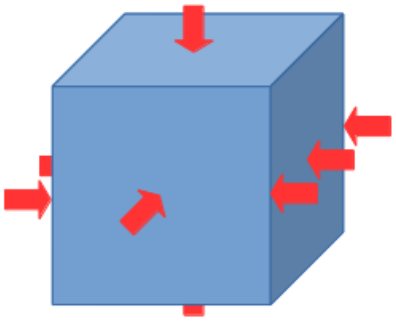

Abbildung 3.5: Einflugrichtungen der zehn jeweils zusammengefassten Äquivalenzklassen.

Dazu bietet es sich an, Einflugrichtungen, die sich auf die selbe Art beschreiben und behandeln lassen, in Äquivalenzklassen einzuteilen. Für diese Einteilung wird die Peilung verwendet, aus der Luftfahrzeuge in den Quader einfliegen (s. Abschn. 2.1). Als äquivalent werden jeweils

jene Einflugrichtungen angesehen, bei denen die Luftfahrzeuge aus gleicher Peilung in einen Quader einfliegen, unabhängig vom vertikalen Flugwinkel. Damit ergeben sich zehn Einflug-Äquivalenzklassen: direkt von Oben, direkt von Unten und acht Peilungen, in 45 Grad Schritten von 0 bis 360 Grad (s. Abb. 3.5).

Kreuzen in der Diagonale

Die komplexere Kollisionsmöglichkeit ist die Kreuzung zweier Luftfahrzeuge in der Diagonale. Hierbei fliegen beide Luftfahrzeuge jeweils entweder über eine Kante oder über eine Ecke von einem Quader in den anderen und kreuzen sich so, ohne jemals einen gemeinsamen Quader belegt zu haben. Problematisch an diesen Kreuzungen ist die Detektion. Dazu muss der Kollisionserkennungsalgorithmus (s. Abschn. 3.4) so erweitert werden, dass während der sequenziellen Analyse der Route bei diagonalen Übergängen zwischen Quadern auch bestimmte Nachbarquader mit auf Kollisionen überprüft werden. Bei Übergängen über die Kante sind das zwei Nachbarquader, die zusätzlich analysiert werden. Bei dem Quaderwechsel über eine Ecke müssen sogar sechs zusätzliche Quader überprüft werden.

Die tatsächliche Laufzeit des Kollisionserkennungsalgorithmus liegt damit weiterhin im Bereich von $\mathcal{O}(n)$. Trotz weiterhin linearer Laufzeit würde allerdings der Aufwand für das Analysieren einer Route mit diagonalen Übergängen nur über Quaderkanten drei mal höher ausfallen als der Aufwand für eine Route mit Flächenübergängen. Für Übergänge über die Ecke wäre es sogar ein siebenfach höherer Aufwand.

Da der Flugsicherungsalgorithmus auf eingebetteten Systemen mit begrenzter Rechenleistung eingesetzt werden soll, ist ein siebenfach höherer Aufwand nicht wünschenswert. Auch die zusätzliche Komplexität der Implementierung eines Kollisionserkennungsalgorithmus, der auch Übergänge über Ecken absichern kann, resultiert in einem Arbeitsaufwand, der über den geplanten Aufwand dieser Arbeit hinaus geht. Aus diesen Gründen wird der Graph des Luftraummodells so begrenzt, dass lediglich Quaderübergänge über Flächen und Kanten möglich sind, nicht aber über Ecken.

Folgen der selben Route

Das Folgen derselben Route ist ein Sonderfall, bei dem mehrfach in aufeinander folgenden Quadern ein Zusammentreffen festgestellt wird. Ein normales Ausweichen ist hierbei nicht

unbedingt sinnvoll, da viel Luftraum verschenkt wird, sollten beide Luftfahrzeuge parallel fliegen. Daher müssen hierfür gesonderte Regeln aufgestellt werden.

3.5.3 Ausweichregeln

In diesem Abschnitt werden die Ausweichregeln des autonomen Flugsicherungskonzeptes definiert. Diese Regeln basieren stark auf dem langerprobten Konzept der Ausweichregeln der konventionellen Flugsicherung (s. Abschn. 2.2.1). Durch diese starke Anlehnung an das bestehende Konzept soll vermieden werden, schwerwiegende konzeptionelle Fehler zu machen, und so für ein sicheres autonomes Flugsicherungskonzept gesorgt werden.

Die Regeln teilen sich auf in die grundsätzlichen Regeln zum Kreuzen oder Zusammentreffen im Luftraum und die besonderen Regeln im Falle einer gleichen Route.

Basisregeln

Die folgenden Regeln gelten für ein Zusammentreffen oder Kreuzen sowohl in einem Quader als auch auf einem diagonalen Pfad.

B.1 Befinden sich Luftfahrzeuge im Gegenanflug zueinander und es besteht die Gefahr einer Kollision, haben beide Luftfahrzeuge nach rechts auszuweichen. Im Gegenanflug befinden sich zwei unbemannte Luftfahrzeuge dann, wenn sie in einen Quader einfliegen und ihre Einflugklasse sich genau um 180 Grad unterscheidet.

B.2 Kreuzen sich die Flugwege zweier Luftfahrzeuge, muss das Luftfahrzeug, das von links kommt, nach rechts ausweichen. Ein von links kommendes Luftfahrzeug ist hierbei definiert als ein Luftfahrzeug, das eine Einflugklasse mit entweder 45, 90 oder 135 Grad mehr als die eigene Einflugklasse hat. Umgekehrt hat ein Luftfahrzeug, das von rechts kommt, eine Einflugklasse mit 45, 90 oder 135 Grad weniger als die eigene Einflugklasse.

B.3 Stationären Luftfahrzeugen oder welchen, die direkt von oben oder unten in den Quader einfliegen, muss immer nach rechts ausgewichen werden.

B.4 Luftfahrzeugen, die erkennbar in der Manövrierfähigkeit behindert sind (Priorität Notlage), muss immer ausgewichen werden.

B.5 Wollen zwei Luftfahrzeuge sowohl von oben als auch von unten in einen Quader einfliegen, muss das Luftfahrzeug ausweichen, welches von oben kommt. Ausgewichen wird in Peilung 0 Grad des Kollisionsquaders, da Richtungsangaben wie rechts und links bei senkrechten Bewegungen nicht eindeutig sind.

B.6 Ein Luftfahrzeug, das nach diesen Regeln nicht auszuweichen hat, muss Kurs und Geschwindigkeit halten, bis die Kollisionsgefahr abgewendet ist.

Folgeregeln

Diese Regeln gelten im Fall einer gemeinsamen Route. Sie sorgen neben der Flugsicherung auch für eine geregeltere Verkehrssteuerung.

F.1 Sich folgende Luftfahrzeuge sind definiert als Luftfahrzeuge, die über mehrere Quader hinweg dieselbe Ein- und Ausflugsklasse haben.

F.2 Haben die beiden Luftfahrzeuge dieselbe oder das nachfolgende einen schnelleren Fahrzeugtyp, darf das nachfolgende Luftfahrzeuge nur überholen, wenn die eigene Reisegeschwindigkeit höher ist als die angestrebte des vorausfliegenden Luftfahrzeugs.

F.3 Darf oder soll nicht überholt werden, muss das nachfolgende Luftfahrzeug die Geschwindigkeit drosseln, so dass sichergestellt ist, dass nur in leere Quader eingeflogen wird und ein ausreichender Bremsweg gewährleistet ist.

F.4 Darf oder soll nicht überholt werden und das Drosseln der Geschwindigkeit ist dem nachfolgenden Luftfahrzeug nicht möglich, darf dieses Manöver zum Verlangsamen oder ein Parallelkurs fliegen.

F.5 Darf und soll überholt werden, muss das überholende Luftfahrzeug nach rechts ausweichen.

3.5.4 Erkennung der Szenarien

Wenn von der Kollisionserkennung (s. Abschn. 3.4) eine Kollision erkannt wird, ist erstmal nur der betreffende Quader der Route bekannt. Für die Erkennung des genauen Szenarios müssen nun weitere Informationen akkumuliert werden. Dazu werden als erstes die Informationen zur

Blockierung abgefragt und so die Ein- und Ausflugszeiten und die zugehörige FAM aquiriert. Anhand der übertragenen Route in der FAM und der eigenen Route werden jeweils die Ein- und Ausflugrichtungen in den kritischen Quader bestimmt.

Auf Basis der gesammelten Informationen wird entschieden, welche Situation vorliegt und welche Regel anzuwenden ist. Hat das fremde Luftfahrzeug einen stationären Stationstypen oder die Ausflugszeit ist unendlich, ist oder wird dieses Luftfahrzeug hier stationär und nach Basisregel B.3 ist ein Ausweichen nötig. Hat die FAM eine Notfall-Priorität, greift die Regel für Notfälle. Entsprechend der Ein- und Ausflugrichtungen werden die weiteren Fälle identifiziert und auf Basis der Regeln behandelt.

3.5.5 Ausweichverfahren

Wird erkannt, dass ein Ausweichen nötig ist, muss ein entsprechendes Manöver eingeleitet werden. Dazu müssen die Kantengewichte des Graphen des Luftraummodells so manipuliert werden, dass eine nachfolgende Neuplanung einer Route die entsprechenden Regeln berücksichtigt.

Grundsätzlich wird immer nach rechts ausgewichen, also sollte idealerweise durch den Quader rechts vom Kollisionsquader geflogen werden. Dafür muss dieser Quader als Zwischenziel zur Flugroute hinzugefügt werden. Allerdings hängt das genaue Manöver von mehreren Faktoren ab. Zum einen muss berücksichtigt werden, ob der eigene Kurs ein gerader oder ein diagonaler Kurs ist, da eine Auswahl des rechten Quaders in diesen Fällen unterschiedlich ist. Zum anderen muss in manchen Fällen mehr als ein Quader als Zwischenziel ausgewählt werden, um einen hinreichend guten Ausweichkurs zu erreichen. Einfluss darauf hat hauptsächlich der Winkel, in dem die beiden Luftfahrzeuge sich zueinander bewegen. Handelt es sich um einen 90 Grad Winkel, reicht die einfache Auswahl des rechten Quaders als Zwischenziel. Bei 45 oder 135 Grad Winkeln müssen sowohl der Rechte als auch ein weiterer Quader gewählt werden, um die Luftfahrzeuge sicher zu separieren. Im Fall einer 90 Grad Kreuzung ist es außerdem möglich, dass sich das eigene Luftfahrzeug im Steig- oder Sinkflug befindet. In diesem Fall müssen weitere Quader als Zwischenziele hinzugefügt werden, da wegen der fehlenden Eck-Übergänge der rechte Quader nicht ordentlich angeflogen wird.

Etwas aufwendiger ist das Manöver, wenn sich zwei Luftfahrzeuge im Gegenanflug befinden. In dem Fall wird zwar auch nach rechts ausgewichen, allerdings reicht es hier nicht aus, nur in

dem Quader auszuweichen, in dem die Kollision zuerst erkannt wurde. Das liegt daran, dass es durch den Sicherheitsaufschlag bei den Zeitberechnungen sein kann, dass die wirkliche physikalische Kollision gar nicht in diesem Quader stattfindet sondern in einem danach. Ein einfaches Ausweichen würde es möglich machen, dass danach wieder auf den Ursprungskurs gewechselt wird und es trotzdem noch zu einer Kollision kommt. Daher muss für alle Quader, die eine zeitliche Kollision haben, jeweils der rechte Quader als Zwischenziel ausgewählt werden. Eine Rückkehr auf den Ursprungskurs ist danach nicht zwingend notwendig.

4 Softwaredesign und Implementierung

Die Implementierung des Konzeptes ist das Thema dieses Kapitels. Dabei wird der Entwurf einer Software-Architektur vorgestellt und die Details der Umsetzung von Kernkomponenten präsentiert. Zu den Kernkomponenten zählen das Luftraummodell (s. Abschn. 3.1), ein Wegfindungsalgorithmus (s. Kap. 1), die UAV2UAV-Kommunikation (s. Abschn. 3.3), die Hardware-Abstraktion der Luftfahrzeuge und ein Controller für das Verhalten der Luftfahrzeuge. Auch die Implementierung der Kollisionserkennung (s. Abschn. 3.4) und -vermeidung (s. Abschn. 3.5) wird als Controller realisiert. Der Inhalt dieses Kapitels baut auf einer früheren eigenen Arbeit auf [2].

4.1 Softwarearchitektur

Das Hauptaugenmerk bei dem Entwurf der Architektur liegt darauf, eine möglichst hohe Wiederverwendbarkeit zu erreichen anstatt eine einmalig benutzte Proof-of-Concept-Anwendung zu schaffen. Dazu wird eine Implementierung in Form eines Frameworks gewählt, das unabhängig von der eigentlichen Zielapplikation und der eingesetzten Hardware ist (s. Abb. 4.1). Außerdem wird eine höchstmögliche Austauschbarkeit von Komponenten angestrebt sowie der Code in verschiedene Abstraktionsebenen aufgeteilt.

Da eine Architektur vorgesehen ist, die neben wiederverwendbaren Implementierungen auch Relationen zwischen den Komponenten und den Logikfluss in der Anwendung vorgibt, handelt es sich um ein Framework und nicht um eine Klassenbibliothek [28].

4.1.1 Austauschbarkeit

Die Austauschbarkeit soll es ermöglichen, das Framework mit anwendungsspezifischen Komponenten verwenden zu können. Denkbar sind beispielsweise ein anderer Wegfindungsalgo-

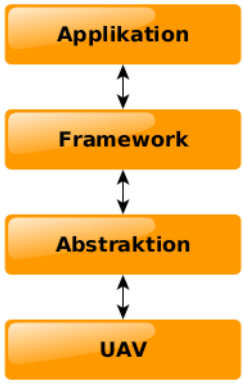

Abbildung 4.1: Aufbau eines Framework-basierten UAV Projekts.

rithmus oder bestimmte Hardware. Um eine solche Austauschbarkeit zu erreichen, werden zwei Techniken eingesetzt.

So werden für alle Kernkomponenten Interfaces definiert und diese damit abstrahiert. Durch diese Abstraktion können neue Komponenten geschrieben werden, die ein solches Interface erfüllen. Diese können von anderen Komponenten genutzt werden, ohne dass die genaue Implementierung bekannt sein muss. Durch diese Abstraktion ist das Framework auch nicht von der eingesetzten Hardware abhängig. Es arbeitet lediglich auf einem Interface, welches für die tatsächliche Hardware implementiert weren muss.

Als zweite Technik werden Containerklassen verwendet. Containerklassen erlauben es, in einer Datenstruktur Informationen abzulegen, von deren Art und Struktur die Containerklasse aber nichts weiß. Diese Technik wird verwendet, um es beispielsweise Algorithmen zu ermöglichen, Informationen in Datenstrukturen abzulegen, die aber für die Datenstruktur transparent sind. Damit kann ein Algorithmus ausgetauscht werden, ohne dass eine neue Datenstruktur entwickelt werden muss.

4.1.2 Abstraktionsebenen

Um das Framework möglichst vielfältig einsetzbar und erweiterbar zu entwickeln, wird ein Architekturmodell gewählt, was auf Abstraktionsschichten oder -ebenen basiert. Als Basis dafür wird die Subsumption-Architektur gewählt, die eine Aufteilung in Kompetenzschichten

einführt. Als Quelle für den folgenden Abschnitt dient das Paper, in dem diese Architektur erstmalig vorgestellt wurde [3].

Abbildung 4.2: Klassische Robotersteuerungssystem-Architektur ohne Subsumption [3].

Die Subsumption-Architektur wurde als ereignisgesteuerte Steuerungssystemarchitektur für mobile Roboter entwickelt. Ziel der Entwicklung war es, das Steuerungssystem eines Roboters nicht mehr aus funktionalen Blöcken zusammenzusetzen, sondern aus Komponenten mit aufgabenbasiertem Verhalten, die in Kompetenzschichten angeordnet werden und leicht erweiterbar sind.

Abbildung 4.3: Subsumption Robotersteuerungssystem-Architektur [3].

Die klassische Architektur einer Robotersteuerung (s. Abb. 4.2) besteht aus hintereinandergeschalteten funktionalen Blöcken, die auf Sensordaten und Ereignisse reagieren. Eine Subsumption-Architektur hingegen besteht aus verschiedenen Kompetenzschichten, die jeweils ein bestimmtes Verhalten steuern sollen (s. Abb. 4.3).

In der Subsumption-Architektur sind die Kompetenzschichten hierarchisch organisiert (s. Abb. 4.4). Untere Schichten implementieren Basisfunktionen des Roboters, höhere Schichten darauf aufbauende aufwendigere Funktionen. Dabei können höhere Schichten auf Funktionen niedrigerer Schichten zurückgreifen sowie deren Ein- und Ausgangssignale beeinflussen. Erweitert werden kann die Subsumption-Architektur sehr leicht dadurch, dass sich einfach weitere übergeordnete Kompetenzschichten hinzufügen lassen.

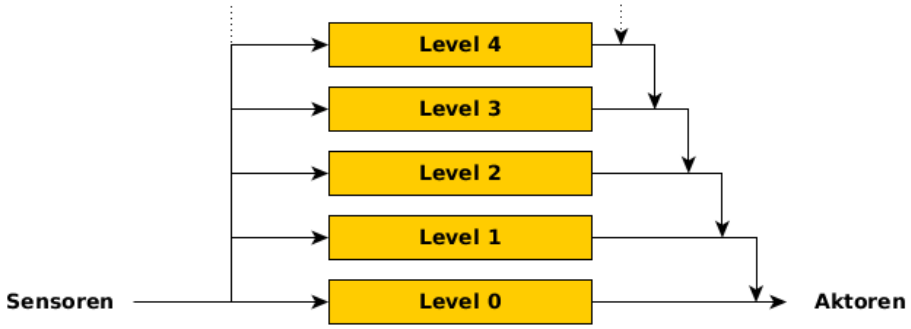

Abbildung 4.4: Kompetenzschichten mit Beeinflussung [3].

Die ursprüngliche Subsumption-Architektur basiert allerdings auf verteilten Prozessoren und Hardware-Signalen, deshalb kann sie nicht direkt für das Softwareframework verwendet werden. Stattdessen wurde ein objektorientiertes Software-Design auf Basis der Subsumption-Architektur entwickelt (s. Abb. 4.5).

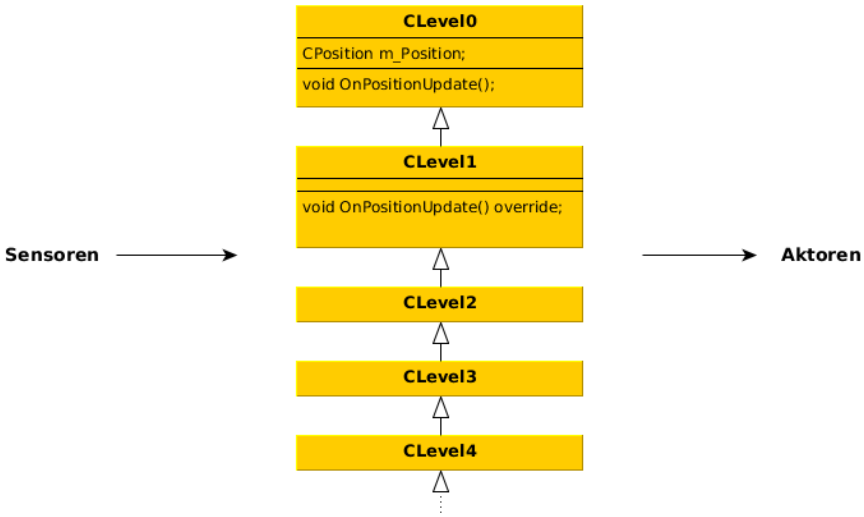

Abbildung 4.5: Objektorientierte Kompetenzschichten. Level 1 unterdrückt Level 0.

Die Kompetenzschichten sind hierbei als voneinander erbende Klassen implementiert, die höheren Schichten erben von der jeweils niedrigeren Schicht. Jede Schicht stellt hierbei einen bestimmten Controller dar. Eingangssignale werden durch Methodenaufrufe realisiert, auf

die ein Controller reagiert. Die übergeordneten Controller können, wie in der originalen Architektur, durch Methodenaufrufe Funktionen der unteren Schichten mitnutzen. Durch Polymorphie lassen sich auch Signale an untere Schichten beeinflussen, indem Methoden für Eingangssignale überschrieben werden. So kann in den überschreibenden Methoden das Signal verändert oder unterdrückt werden, ehe die ursprüngliche Methode aufgerufen wird. Auch das Ausgangsignal der ursprünglichen Methode kann so nachträglich verarbeitet werden.

Anwendern des Frameworks bietet dieses Design die Möglichkeit, einfach an beliebigen Stellen das Framework zu erweitern. Dazu muss lediglich eine Klasse von dem Controller, auf welchem aufgebaut werden soll, abgeleitet werden.

4.2 Implementierung

Bei der Zielhardware auf unbemannten Flugzeugen handelt es sich meist um Mikrocontroller mit begrenzter Leistung. Daher steht bei der Implementierung des Frameworks eine ressourcensparende Umsetzung im Vordergrund. Als Sprache wird C++ ausgewählt, da für die meisten Mikrocontroller-Architekturen C++-Compiler existieren und die Sprache hohe Performanz, Hardwarenähe und Objektorientierung verbindet.

4.2.1 Controller

Die Kernkomponente des Frameworks sind die auf der Subsumption-Architektur aufbauenden, hierarchischen Controller. Jeder Controller kann für sich alleine benutzt werden, damit ist das Framework für diverse Applikationen mit unterschiedlichen Anforderungen an die Schnittstelle benutzbar. Die Controller benötigen dabei lediglich Komponenten aus ihrer und den untergeordneten Kompetenzschichten, um unnötige Abhängigkeiten bei einfacheren Schnittstellen zu verhindern. Wie bei der Subsumption-Architektur definiert, haben die Controller von unten nach oben aufbauende Kompetenzen.

Der einfachste Controller implementiert die Steuerung eines simplen Fluges zu einem Quader und informiert, sobald das Luftfahrzeug in den Quader eingeflogen ist. Als Abhängigkeiten werden lediglich einfache Datentypen wie Quader und 3D-Punkte sowie eine vom Anwender wählbare UAV-Implementierung benötigt.

Davon erbend erlaubt der nächsthöhere Controller das Verfolgen einer Flugroute, die als sequenzielle Liste von Quadern repräsentiert wird. Er greift dabei auf die Funktion des ersten Controllers zurück, um von einem Wegpunkt der Route zum nächsten zu kommen. Allerdings beeinflusst er dabei das Ausgangssignal des untergeordneten Controllers, so dass längere Flüge ohne Richtungsänderungen optimiert werden. Dies ist nötig um zu verhindern, dass das unbemannte Luftfahrzeug in jedem Wegpunkt abbremst. Desweiteren berechnet der Controller geschätzte Flugzeiten für die Route und informiert andere Komponenten bei Erreichen des letzten Wegpunktes. Zusätzliche Abhängigkeiten entstehen durch diesen Controller nicht.

Die dritte Kompetenzschicht führt ein autonomes Verhalten ein. Damit ist es möglich, lediglich einen Zielpunkt anzugeben und das unbemannte Luftfahrzeug sucht sich automatisch eine Route zu diesem Punkt. Für die Verfolgung der Route wird die Funktion des vorherigen Controllers verwendet. Zum Bestimmen der Route wird nun zusätzlich ein Wegfindungsalgorithmus benötigt (s. Kap. 1), der wiederrum eines Luftraummodells (s. Abschn. 3.1) bedarf. Welche konkreten Implementierungen von dem Wegfindungsalgorithmus und dem Luftraummodell verwendet werden, obliegt hierbei dem Anwender.

In der vierten Kompetenzschicht ist das Konzept der Flugsicherung implementiert. Damit ist es möglich, ein unbemanntes Luftfahrzeug autonom und abgesichert zu einem Zielpunkt zu schicken. Die Funktionalität des vorhergehenden Controllers wird hierbei um die Funktionen zur Kollisionserkennung (s. Abschn. 3.4) sowie Kollisionsvermeidung (s. Abschn. 3.5) erweitert. Dafür braucht der Controller direkten Zugriff auf das Luftraummodell (s. Abschn. 3.1) sowie eine Kommunikationsinstanz (s. Abschn. 3.3).

4.2.2 UAV Abstraction Layer

Um das Framework hardwareunabhängig zu gestalten, ist das unbemannte Luftfahrzeug (Unmanned aerial vehicle, UAV) durch ein Interface abstrahiert. Eine Implementierung des Interfaces muss einen physikalischen Flugregler beinhalten, so dass das Luftfahrzeug eigenständig in der Lage ist, einen dreidimensionalen Punkt auf direktem Wege anzufliegen. Des Weiteren muss die Implementierung regelmäßig Positionsupdates an den Controller schicken, der diese als Ereignisse verarbeitet, und einen Algorithmus zur Berechnung einer Flugzeit mitbringen. Dieser Algorithmus muss in der Lage sein, die Flugzeit zu jedem Punkt zwischen Start- und Zielpunkt eines Fluges abzuschätzen.

Da für diese Arbeit eine Simulationsumgebung eingesetzt werden soll (s. Kap. 1), wird das Framework eine Implementierung des UAV-Interfaces zur Ansteuerung der simulierten Luftfahrzeuge beinhalten.

4.2.3 Wegfindungsalgorithmus

Ein autonomes unbemanntes Luftfahrzeug muss eigenständig einen Weg zu seinem Ziel finden können. Dafür braucht der entsprechende Controller des Frameworks sowohl eine Modellierung des Luftraums als auch einen Wegfindungsalgorithmus der auf diesem Modell arbeitet. Im Rahmen der Konzeptionierung des Flugsicherungsalgorithmus wurde entschieden, dass ein Luftraummodell verwendet wird, was sich als Graph repräsentieren lässt (s. Abschn. 3.1.1). Daher muss auch der Wegfindungsalgorithmus graphenbasiert arbeiten.

Graphenbasierte Wegfindungsalgorithmen sind ein bekanntes Forschungsgebiet in der Informatik, sie kommen beispielsweise auch in der Fahrzeugnavigation zum Einsatz (s. Abschn. 2.5.1). Es gibt daher viele verschiedene derartige Algorithmen, die zum Beispiel auf besondere Probleme zugeschnitten oder in der Leistung optimiert sind. Da eine spätere Applikation eine starke Abhängigkeit zu einem bestimmten Wegfindungsalgorithmus haben kann, ist dieser ebenfalls über ein Interface abstrahiert.

Für den Einsatz im Zusammenhang mit dem Flugsicherungskonzept bedarf es eines Wegfindungsalgorithmus, der möglichst performant ist und schnell eine Lösung findet. Hier bieten sich heuristische Suchalgorithmen an. Diese Algorithmen erweitern konventionelle Suchalgorithmen wie den Algorithmus von Dijkstra um eine Heuristik. Diese Heuristik schätzt die Kosten des Weges von einem Knoten zum Zielknoten. Der Algorithmus exploriert immer in die Richtung des Knotens mit den geringsten geschätzten Kosten (s. Abb. 4.6). Auf diese Weise soll eine unnötige Exploration unbedeutender Knoten vermieden werden und der Algorithmus gewinnt bestenfalls an Performanz. Die Performanz des Algorithmus im schlechtesten Fall bleibt hingegen identisch. Bekanntester dieser Algorithmen ist A* [16].

Des Weiteren kann es bei dem Einsatz im Flugsicherungskonzept aufgrund von Ausweichmanövern (s. Abschn. 3.5.5) häufiger nötig sein die Route neu zu planen. Darauf sind besonders inkrementelle Suchalgorithmen spezialisiert, sie beschleunigen mehrfache ähnliche Suchen im selben Suchraum. Dazu werden Informationen aus dem vorherigen Suchlauf wiederverwendet anstatt den Algorithmus komplett neu laufen zu lassen. So erhöht sich die Performanz bei

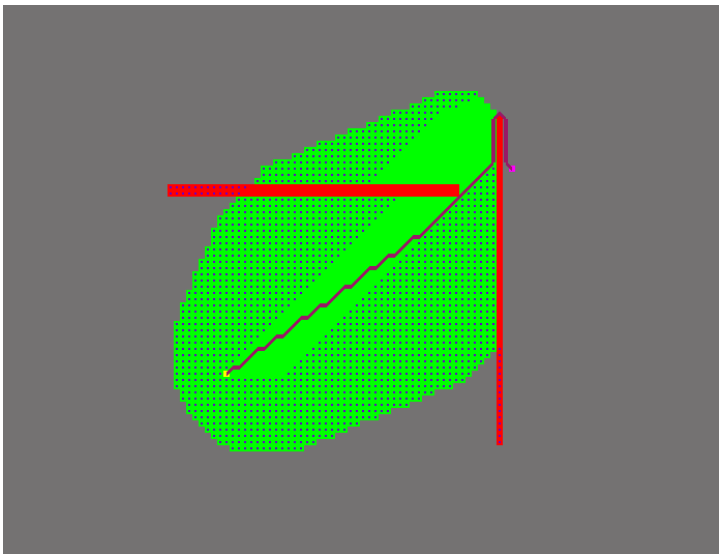

Abbildung 4.6: Heuristischer Algorithmus. Exploriert wird von Lila zu Gelb. (Grün: exploriert, Grün-blau: Open-List, Rot: Hindernis, Grau: nicht exploriert.)

wiederholten Suchen deutlich im Vergleich mit dem erneuten Ausführen von Algorithmen wie A*. Ein bekannter Vertreter dieser Algorithmen ist DynamicSWSF-FP [17].

Ideal für diese Arbeit wäre also ein Algorithmus, der beide Eigenschaften vereint. Um die Vorteile von heuristischen und inkrementellen Suchalgorithmen in einem hybriden Algorithmus zu verbinden, wurden verschiedene Weiterentwicklungen bestehender Algorithmen erforscht und veröffentlicht.

Beispielsweise kombiniert LPA* DynamicSWSF-FP und A*. Ziel hierbei ist, eine Suche vom Start- zum Zielpunkt mehrfach bei veränderten Kantenkosten möglichst effizient durchzuführen. Bei der Performanz ist LPA* mit A* vergleichbar und gleichzeitig einfach und kompakt zu implementieren. Allerdings ist LPA* nicht für Suchen mit verändertem Startpunkt, wie bei bewegenden Luftfahrzeugen nötig, geeignet [16, 17].

D* hingegen ist genau für die Problematik des beweglichen Startknotens entwickelt worden. Dieser Algorithmus ist ein inkrementeller Wegfindungsalgorithmus, der sich wie A* benutzen lässt aber dynamische Neuberechnungen der Route mit verändertem Startpunkt und Kantenkosten erlaubt. Da der Algorithmus allerdings keine Heuristik verwendet, ist er deutlich weniger performant als A*. Um diese Schwäche auszubessern wurde die Erweiterung Focused D* entwickelt. Focused D* erweitert D* um eine Heuristik und ist damit wie LPA* ein hybrider

Algorithmus und ähnlich performant wie A*. D* und Focused D* sind weit verbreitet bei Problemen mit unbekanntem Suchraum, wie sie oft in der Robotik auftreten. So kommt er beispielsweise bei der Entwicklung von Mars Rovern der NASA zum Einsatz. Der Algorithmus ist aber komplex, daher sind Implementierungen umfangreich und schwer wartbar [17, 33, 34].

Da Focused D* komplex ist und nicht mehr weiterentwickelt wurde, startete die Entwicklung von D* Lite. D* Lite verbindet die wünschenswerten Eigenschaften von LPA* und D*. Er ist kompakt und effizient wie LPA* und besitzt das Verhalten und die Schnittstelle von D*. Dabei ist er mindestens so effizient Focused D* bei signifikant weniger Code-Länge. Auch besonders verschachtelte Bedingungsabfragen wie bei D* konnten vermieden werden [16].

Für die Verwendung im Rahmen des Flugsicherungskonzeptes ist D* Lite daher bestens geeignet und wird im Framework implementiert. Da D* Lite als inkrementeller Algorithmus Informationen zur Suche in den Knoten zwischenspeichern muss, wurde das Luftraummodell bzw. die Implementierung der Knoten zu einer Containerklasse erweitert.

4.2.4 Kommunikation

Obwohl die Entwicklung eines Kommunikationsstapels nicht Teil dieser Arbeit ist (s. Kap. 1), bedarf es zur Implementierung und Evaluation des Flugsicherungskonzepts doch einer entsprechenden Komponente für die UAV2UAV-Kommunikation (s. Abschn. 3.3). Das Konzept der UAV2UAV-Kommunikation basiert auf dem Versenden von Broadcast-Nachrichten, die von allen Teilnehmern im Umkreis empfangen werden (s. Abschn. 3.3.1). Um diese Kommunikation nachbilden zu können, muss auch die Evaluationsimplementierung Broadcast-Nachrichten unterstützen.

Da das Konzept im Rahmen einer Simulation evaluiert wird (s. Kap. 1), die auf einem x86-System mit Betriebssystem läuft, kann auf einen kompletten Netzwerk-Stack zurückgegriffen werden. Damit steht das Internetprotokoll (IP) zur Verfügung, welches sowohl einen Broadcast- als auch einen Multicast-Mechanismus bietet. Während beim IP-Broadcast Pakete an alle Netzwerkteilnehmer gesendet werden, bietet IP-Multicast die Möglichkeit, sich an Multicastgruppen anzumelden und zu beteiligen. So ist sichergestellt, dass die Nachrichten nur bei interessierten Teilnehmern ankommen. Um die Netzwerklast für unbeteiligte Netzwerkteilnehmer gering zu halten, wurde das UAV2UAV-Netzwerk daher über eine IP-Multicast-Gruppe realisiert, auf der

sich alle UAVs registrieren. Wird eine Nachricht an diese Multicast-Gruppe gesendet, wird sie von allen UAVs empfangen. Der geforderte Broadcast zu allen UAVs ist damit möglich [8].

Die im UAV2UAV-Protokoll definierten Nachrichtentypen (s. Abschn. 3.3.3) wurden als Klassen implementiert. Neben den Daten der Nachrichten stellen diese auch Serialisierungs- und Deserialisierungsfunktionen bereit. Als Datenaustauschformat über das UAV2UAV-Netzwerk wurde hierbei JSON verwendet. Um die Anforderung nach einer begrenzten Reichweite (s. Abschn. 3.3.2) zu erfüllen, wurden die Nachrichten nach der Entfernung zwischen eigener Position und der Position des Absenders, welche Teil der Nachrichten ist, gefiltert.

5 Simulationsumgebung

Zur gefahrlosen Evaluation des Flugsicherungskonzepts soll eine Simulation verwendet werden (s. Kap. 1). Dieses Kapitel befasst sich mit der Anforderungserhebung, Softwarerecherche, Evaluation und Auswahl einer für UAV-Anwendungen geeigneten Simulationsumgebung. Für dieses Kapitel wurden Ergebnisse einer früheren, eigenen Arbeit zugrunde gelegt [1].

5.1 Anforderungen

Damit bei der Simulation verwertbare Daten für die Evaluation des Flugsicherungskonzeptes entstehen, muss der Simulator gewissen Anforderungen gerecht werden. Des Weiteren ist es wünschenswert, dass eine ausgewählte Simulationsumgebung möglichst universell und auch für andere Projekte verwendbar ist. Unter diesem Gesichtspunkt entstehen einige weitere Anforderungen.

Zu den grundlegenden Anforderungen an einen geeigneten Simulator zählt unter anderem, dass er in der Lage ist, unbemannte Luftfahrzeuge in Echtzeit zu simulieren, eine Physik-Engine mit Kollisionserkennung hat, 3D-Visualisierung besitzt und eine Kommunikationsschnittstelle zur Verfügung stellt. Die Kommunikationsschnittstelle muss es erlauben, Daten in Echtzeit abzufragen und verschiedene Luftfahrzeuge einzeln zu steuern. Benötigte Daten sind unter anderem die Positionen der Luftfahrzeuge und Kollisionsereignisse. Eine wichtige Anforderung ist außerdem, dass der Simulator skalierbar ist und eine variable Anzahl von Luftfahrzeugen simulieren kann. Besonders für die Simulation von Verkehrssituationen ist es notwendig, dass viele Luftfahrzeuge zeitgleich simuliert werden können. Außerdem soll das von dem Simulator simulierte Szenario leicht austauschbar sein.

5.2 Kandidaten

Die erste Idee, einen Trainings- oder Unterhaltungsflugsimulator zu verwenden, erwies sich aus diversen Gründen als unrealistisch. Diese Simulatoren haben in der Regel keine Kollisionserkennung, lassen keinen Zugriff auf Simulationsdaten zu oder erlauben die Simulation von mehr als einem Luftfahrzeug nicht. Daher wird der Fokus auf Roboter-Simulatoren gelegt, die auch unbemannte Luftfahrzeuge simulieren können.

In anderen wissenschaftlichen Arbeiten, die ebenfalls mit unbemannten Luftfahrzeugen arbeiten, werden verschiedene Simulatoren eingesetzt: Gazebo, V-REP und MORSE. Aufgrund einer Internetrecherche wird außerdem Webots mitbetrachtet. Alle diese Simulatoren sind in der Lage, unbemannte Luftfahrzeuge zu simulieren, haben Physik-Engines, 3D-Visualisierung und bieten eine Schnittstelle an [21, 24, 37].

Bei genauerer Analyse der Spezifikationen und Schnittstellenbeschreibungen zeigt sich, dass MORSE und V-REP alle so überprüfbaren funktionalen Anforderungen erfüllen. Webots und Gazebo hingegen offenbaren Schwächen. Gazebo erlaubt die Simulation von Luftfahrzeugen nur über ein Plugin, was auf dem Robot Operating System basiert. Damit wird der Einsatz für diese Arbeit zu aufwendig. Bei Webots hingegen ist das Simulieren von unbemannten Luftfahrzeugen eine nicht offiziell unterstützte Beta-Funktion, die selbst dazukompiliert werden muss. Daher werden Gazebo und Webots nicht weiter berücksichtigt [22, 6, 7, 11, 25].

5.3 Test

Um zu bestimmen, welcher Simulator die nicht-funktionalen Anforderungen besser erfüllt, wird ein Performance-Test durchgeführt. Damit ein Performance-Test aussagekräftige und quantifizierbare Ergebnisse liefert müssen exakte Testziele, Testmetriken und Testfälle definiert werden. Außerdem hängen die Ergebnisse vom Testsystem ab, daher müssen alle Tests auf demselben System durchgeführt und dieses System in den Ergebnissen dokumentiert werden [29, 15].

5.3.1 Ziele

Als Primärziel des Performance-Tests soll bestimmt werden, wie viele unbemannte Luftfahrzeuge die Simulatoren gleichzeitig simulieren können. Daraus soll ein Rückschluss auf die Anforderung nach Skalierbarkeit gezogen werden. Das Sekundärziel ist, den exakten Bedarf an Hardware-Ressourcen zu bestimmen. Dies soll Aufschluss darüber geben, welche Hardwarekomponenten, beispielsweise CPU oder RAM, den Simulator begrenzen. Mit dieser Information lässt sich die Hardware für die Ausführung des Simulators optimieren.

Betriebssystem	Linux Mint 18.2 Cinnamon 64-bit
Linux Kernel	4.13.0
CPU	Intel Xeon E3-1231 v3 @ 3.40GHz x 4
RAM	16GB
GPU	NVIDIA GeForce GTX 970
HDD	Samsung SSD 850 Pro 256GB

Tabelle 5.1: Spezifikationen des Testsystems

5.3.2 Metriken

Die Testmetrik für das Primärziel ist die Anzahl der simulierten Luftfahrzeuge kombiniert mit der Zeitabweichung zwischen Echtzeit und simulierter Zeit. Diese Metrik ergibt sich aus der Eigenschaft von Simulatoren, dass für jeden Simulationsschritt nur eine bestimmte Spanne an Zeit zur Verfügung steht. Ist die Anzahl der simulierten Objekte zu hoch, kann diese Zeitspanne nicht eingehalten werden und die simulierte Zeit vergeht langsamer als die Echtzeit oder der Simulator überspringt Simulations- bzw. Visualisierungsschritte. Für den Performance-Test sind die Simulatoren so konfiguriert, dass keine Schritte ausgelassen werden. Sobald die Last zu hoch wird, weichen die Simulatoren von der Zeit ab. Gemessen wird die Simulationszeit von den Simulatoren selbst, die Echtzeit wird von der Systemuhr genommen. Für das Sekundärziel ist die Testmetrik der anteilige Hardwareverbrauch des Simulators. Dieser wird mit gängigen Tools zur Messung von Hardwareauslastung wie vmstat und mpstat gemessen. Für die Erfassung der GPU-Auslastung wurde ein properitäres Tool des Herstellers verwendet.

5.3.3 Testfälle

Eine gute Auswahl an Testfällen ist nötig, um möglichst aussagekräftige Ergebnisse zu bekommen. Bei diesem Performance-Test werden vier Testfälle angewandt, bei der jeweils die maximale Anzahl an in Echtzeit simulierbaren Luftfahrzeugen bestimmt wird. Diese Testfälle sind definiert wie folgt:

- Maximale Anzahl an stationären Luftfahrzeugen
- Maximale Anzahl an bewegten Luftfahrzeugen
- Maximale Anzahl an stationären Luftfahrzeugen mit zusätzlichen Sensoren
- Maximale Anzahl an bewegten Luftfahrzeugen mit zusätzlichen Sensoren

Anhand der ersten beiden Testfälle soll neben der maximal möglichen Anzahl auch der Einfluss von Bewegungen auf den Simulator bestimmt werden. Bei den letzten beiden Testfällen hingegen wird zusätzlich der Einfluss von weiteren Sensoren überprüft.

5.4 Evaluation und Auswahl

Um eine möglichst optimale Simulationsumgebung für das Flugsicherungskonzept sowie das Framework auswählen zu können, müssen sowohl die funktionalen Anforderungen als auch die nicht-funktionalen Anforderungen bestmöglich betrachtet werden. Nachdem die funktionalen Anforderungen schon anfangs (s. Abschn. 5.2) abgeglichen wurden, werden in diesem Abschnitt nun die Ergbnisse des Performance-Tests ausgewertet, um Aussagen zu den nicht-funktionalen Anforderungen treffen zu können. Danach wird eine Auswahl für einen Simulator getroffen.

5.4.1 Ergebnisse

Der Performance-Test wurde entsprechend den Vorgaben auf dem Testsystem (s. Tab. 5.1) durchgeführt. Die Ergebnisse werden in den folgenden Abschnitten ausgewertet.

V-REP wird in der Version 3.2.3 getestet, der komplette Testablauf lässt sich über die grafische Oberfläche steuern. Auch alle Informationen zur Echt- und Simulationszeit lassen sich dort direkt ablesen (s. Abb. 5.1).

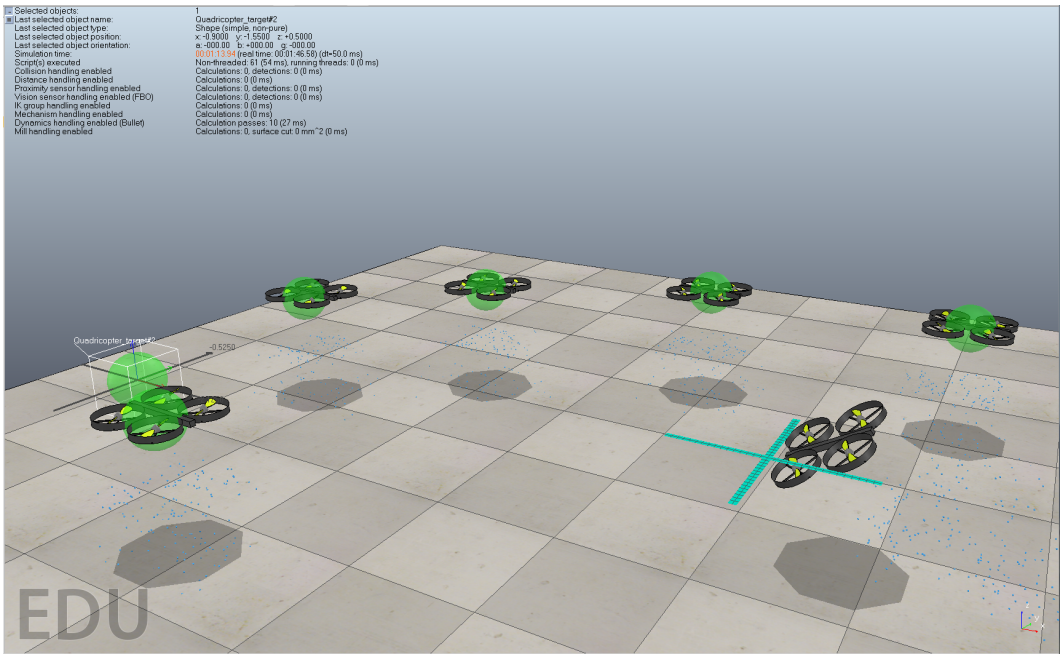

Abbildung 5.1: V-REP Evaluation mit sechs Quadrocoptern. Erkennbar sind die Echtzeitabweichung (rot) und die Particle-Flow-Simulation.

In den Testfällen ohne zusätzliche Sensoren beginnt V-REP ab fünf unbemannten Luftfahrzeugen geringe Abweichungen von der Echtzeit zu zeigen. Bei sechs Luftfahrzeugen liegt die Abweichung bereits bei mehreren Sekunden pro Minute simulierter Zeit (s. Abb. 5.2). Werden zusätzliche Sensoren hinzugefügt, fangen die Abweichungen ab vier unbemannten Luftfahrzeugen an, bei fünf liegen sie dann im zweistelligen Sekundenbereich pro Minute simulierter Zeit. Die Auswertung der Hardwareauslastung zeigt, dass weder Arbeitsspeicher, GPU-Leistung noch Festplattenleistung die Performance des Simulators beeinflussen. Die CPU hingegen läuft nicht auf allen Kernen unter Last, ein Kern ist aber zu 100% ausgelastet. Die Performance wird also eindeutig von der Leistung der CPU begrenzt, wobei der Simulator allerdings nicht auf die volle CPU-Leistung zurückgreift sondern nur einen Kern belastet.

Der Performance-Test von MORSE wird mit der Version 1.4 durchgeführt, welcher Blender in der Version 2.76 zugrunde liegt. Im Gegensatz zu V-REP besitzt MORSE keine umfangreiche Steuerung über die grafische Oberfläche. Daher wird ein kleines, selbstgeschriebenes Programm

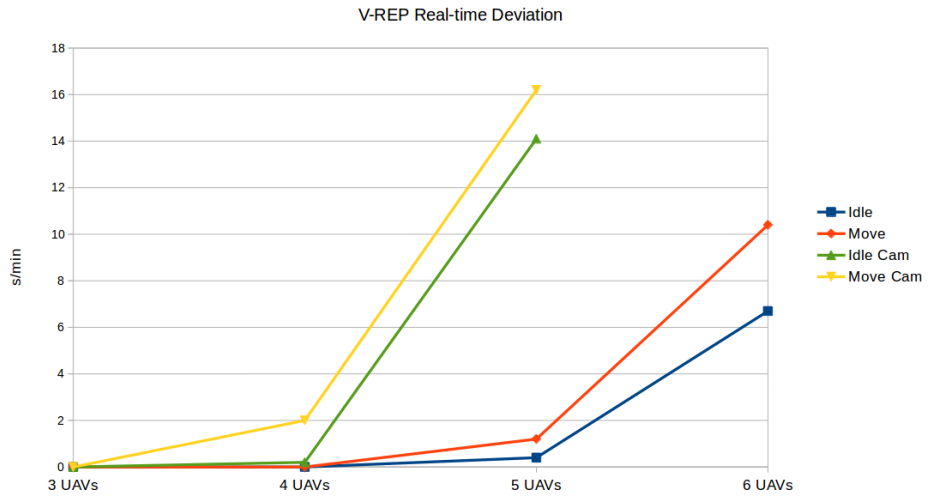

Abbildung 5.2: V-REP Echtzeitabweichung

verwendet, das über die Kommunikationsschnittstelle die unbemannten Luftfahrzeuge steuert und die Simulationszeit ausliest.

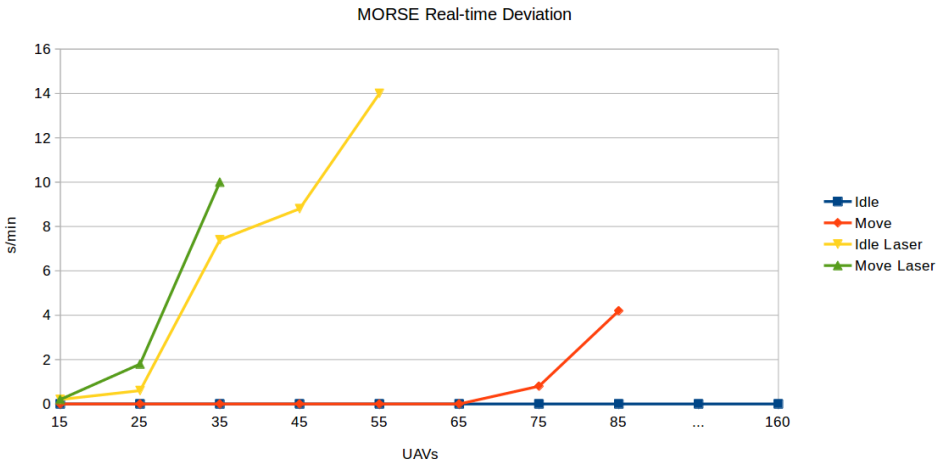

Abbildung 5.3: MORSE Echtzeitabweichung

Bei der Durchführung der Tests zeigt sich eine Besonderheit von MORSE. Im ersten Testfall, wo die maximale Anzahl von stationären unbemannten Luftfahrzeugen, die in Echtzeit simulierbar sind, bestimmt werden soll, lässt sich kein Ergebnis erzielen (s. Abb. 5.3). Auch die Messungen der Hardwareauslastung zeigten keinerlei Last zusätzlich zur Grundlast. Offensichtlich besitzt MORSE eine Optimierung, die verhindert, dass für Luftfahrzeuge, auf die keine Kraft einwirkt

oder die nicht in Bewegung sind, Berechnungen durchgeführt werden. Daher wird dieser Testfall abgebrochen.

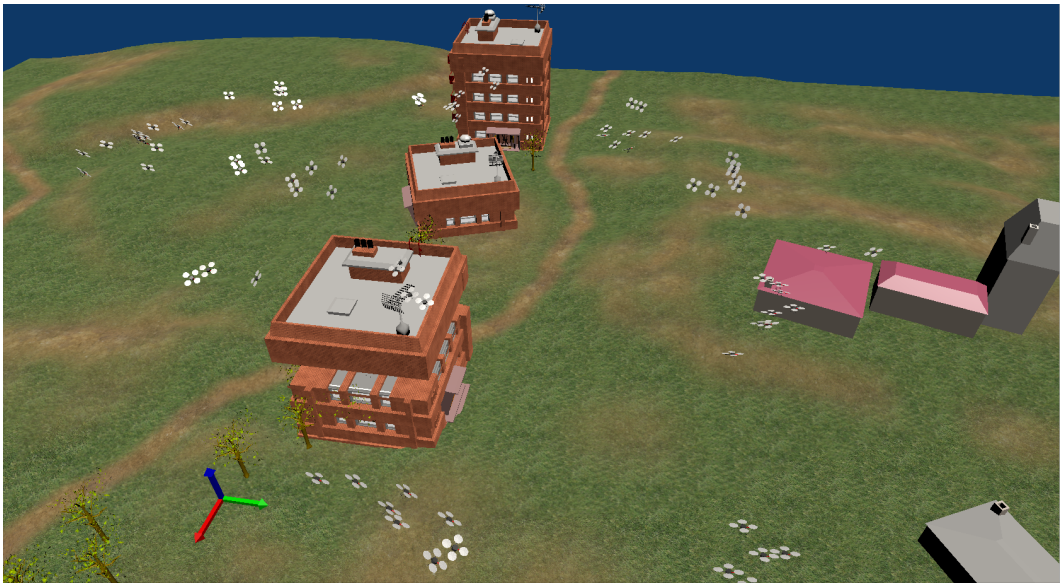

Abbildung 5.4: MORSE Evaluation mit 65 Luftfahrzeugen.

In den anderen Testfällen lassen sich aber Erkenntnisse gewinnen. So kann MORSE 65 bewegte unbemannte Luftfahrzeuge simulieren, ohne von der Echtzeit abzuweichen (s. Abb. 5.4). Ab 75 Luftfahrzeugen gibt es leichte, ab 85 deutliche Abweichungen der simulierten Zeit von der Echtzeit. Fügt man zusätzliche Sensoren hinzu, in diesem Fall Lasersensoren, steigt die Last auch bei stationären Luftfahrzeugen an. Da sowohl stehende als auch bewegte Luftfahrzeuge nur bis zu einer Anzahl von 15 Stück ohne Abweichung der simulierten Zeit von der Echtzeit simulierbar sind, liegt der Hauptteil der Last hierbei aber offensichtlich bei den Berechnungen für die Sensoren. Ab 25 Luftfahrzeugen sind leichte Abweichungen der simulierten von der realen Zeit erkennbar, bei 35 starke. Die Messung der Hardwareauslastung zeigt, wie bei V-REP, dass die CPU der limitierende Faktor ist. Allerdings lastet MORSE diese deutlich besser aus, da alle Kerne voll genutzt werden. Gegenüber V-REP werden außerdem 500 MB weniger Arbeitsspeicher benötigt.

5.4.2 Auswahl

Ziel dieses Kapitels ist es, eine Simulationsumgebung für unbemannte Luftfahrzeuge zu finden, die zur Evaluation des Flugsicherungskonzeptes (s. Kap. 3) verwendet werden kann. Diese Simulationsumgebung soll nach Möglichkeit auch für weitere Anwendungen auf Basis des Frameworks (s. Kap. 4) verwendbar sein. Anhand der daraus aufgestellten Anforderungen kommen vier mögliche Simulatoren in Frage. Bei genauerer Analyse zeigt sich allerdings, dass lediglich V-REP und MORSE die funktionalen Anforderungen ausreichend erfüllen. Um die Performance der beiden Simulatoren in den nicht-funktionalen Anforderungen vergleichen zu können, wird ein Performance-Test durchgeführt.

Anhand der Ergebnisse wird deutlich, dass MORSE die Anforderung nach Skalierbarkeit deutlich besser erfüllt, da hier deutlich mehr Luftfahrzeuge simulierbar sind, ohne dass die Simulationszeit von der Echtzeit abweicht, als mit V-REP. V-REP hingegen hat eine genauere Physik-Engine, die Flugeigenschaften mittels Particle Flow deutlich detallierter simuliert. Allerdings auf Kosten der Performance, denn V-REP kann nur wenige Luftfahrzeuge simulieren, ehe die Echtzeitanforderung nicht mehr erfüllt wird.

Die Anforderung nach Skalierbarkeit leitet sich direkt aus dem Szenario der Flugsicherung ab, da hier Verkehr mit mehreren Luftfahrzeugen stattfindet. Daher ist diese Anforderung höher zu bewerten als die Anforderung nach einer exakten Physik-Engine. Aus diesem Grund wird MORSE als Simulator für das Framework und das Flugsicherungskonzept ausgewählt. Als positiver Nebeneffekt lässt sich das Testprogramm als Basis für die Kommunikation zwischen Framework und Simulator weiterverwenden.

6 Evaluation

Das Kapitel „Evaluation“ befasst sich mit dem Test, der Auswertung von Testergebnissen und der Bewertung des Flugsicherheitskonzepts (s. Kap. 3). Betrachtet werden soll dabei, ob das Flugsicherungskonzept in der Lage ist, eine dauerhafte Separation (s. Abschn. 3.2) der Luftfahrzeuge sicherzustellen. Im Rahmen des Kapitels wird dazu eine Teststrategie entwickelt, anhand derer die Tests durchgeführt werden. Die Ergebnisse dieser Tests werden dann ausgewertet und beurteilt.

Ziel der Evaluation ist es, den Nachweis zu bringen, dass das Flugsicherungskonzept eine durchgehende Separation sicherstellen kann. Um dieses Ziel zu erreichen werden zwei verschiedene Teststrategien verfolgt. Zum einen sollen systematische Tests des Systems im Simulator durchgeführt werden, um zu zeigen, dass sich das System entsprechend den Festlegungen verhält. Zum anderen soll ein Anwendungsszenario simuliert werden, welches überwacht und ausgewertet wird. Dieser Test soll mögliche Lücken sowohl im Flugsicherungskonzept als auch im systematischen Test aufdecken.

6.1 Vorbereitungen

Da sowohl für die Testfälle als auch für die Testanwendung Kommandos von einer zentralen Instanz an die unbemannten Luftfahrzeuge gesendet werden müssen, wird die Multicast-Kommunikationskomponente (s. Abschn. 4.2.4) um einen Nachrichtentyp für Befehle erweitert. Dieser Nachrichtentyp ist nicht Teil der in dem Flugsicherungskonzept definierten UAV2UAV-Kommunikation (s. Abschn. 3.3) und folgt eigenen Regeln. So unterliegen diese Nachrichten beispielsweise keiner künstlichen Reichweitenbeschränkung, um eine durchgehende Übermittlung von Befehlen sicherstellen zu können. Des Weiteren ist der Nachrichtenverkehr dieser Nachrichten stark limitiert, es werden nur selten einzelne Nachrichten versendet, so dass keine Beeinflussung der eigentlichen Flugsicherung stattfindet.

Um die Flugspuren aufzuzeichnen wurde die Klasse, die die Luftfahrzeughardware auf Basis der Simulation implementiert, um eine Logging-Funktion erweitert. Wird das Logging aktiviert, wird jede empfangene Positionsmeldung, bestehend aus Koordinaten und Simulationszeit, als JSON-Objekt in eine Datei geschrieben. Da mit jedem Simulationsschritt eine Positionsmeldung empfangen wird, ist die Position eines jeden Luftfahrzeugs jederzeit nachträglich nachvollziehbar. Zur Analyse der Flugspuren wurden unterschiedliche Python-Scripte verwendet.

6.2 Systematische Tests

Das Ziel der systematischen Tests ist es zu zeigen, dass Kollisionserkennung und Kollisionsvermeidung entsprechend der Spezifikationen des Flugsicherungskonzepts (s. Kap. 3) funktionieren. Basierend darauf, dass das Flugsicherungskonzept alle Kollisionen durch sequenzielles Ausführen des Algorithmus auflöst und dabei keine Flugbewegungen ohne vollständige Auflösung aller Kollisionen zulässt (s. Abschn. 3.5), folgt aus dem erfolgreichen Test des Algorithmus für alle möglichen einzelnen Kollisionssituationen, dass das Flugsicherungskonzept für jegliches Anwendungsszenario funktioniert.

6.2.1 Strategie

Um zu zeigen, dass Kollisionserkennung und Kollisionsvermeidung entsprechend der Spezifikationen funktionieren, müssen mehrere Nachweise erbracht werden. Erstens muss nachgewiesen werden, dass Kollisionen aller drei möglichen Szenarien (s. Abschn. 3.5.2) erkannt werden. Als zweites muss bewiesen werden, dass der Kollisionsvermeidungsalgorithmus die erkannten Kollisionen den richtigen Einflug-Äquivalenzklassen zuordnet und nach der zugehörigen Regel behandelt (s. Abschn. 3.5.3). Weiterhin muss nachgewiesen werden, dass die Ausweichmanöver (s. Abschn. 3.5.5) für jede Äquivalenzklasse eine ausreichende Separation sicherstellen.

Zu dem Erbringen dieser Nachweise werden für alle drei möglichen Kollisionsszenarien, mit allen jeweils möglichen Äquivalenzklassen, Testfälle mit jeweils zwei Luftfahrzeugen geschrieben und in der Simulation durchgeführt. Anhand der aufgezeichneten Flugspuren dieser Testfälle lässt sich nachweisen, dass die richtigen Äquivalenzklassen erkannt wurden und ob dementsprechend korrekt nach den Regeln ausgewichen wurde. Aus den Daten lässt

sich auch die Entfernung zwischen beiden Luftfahrzeugen zu jedem Zeitpunkt errechnen, anhand welcher sich erkennen lässt, ob die Separation zu jeder Zeit gewährleistet war.

6.2.2 Durchführung

Statt für jeden Testfall eine neue zentrale Steuerungsanwendung, die Kommandos an die unbemannten Luftfahrzeuge sendet, kompilieren zu müssen, wurde eine kleine Anwendung entwickelt, die Skripte verarbeiten kann. So können die Testfälle als JSON-basiertes Skript geschrieben und in Textdateien gespeichert werden. Diese Textdateien lassen sich von der Steuerungsanwendung laden und ausführen. Dafür wird die Liste von Kommandos deserialisiert und mittels Multicast-Kommunikation an die unbemannten Luftfahrzeuge verschickt.

Als Testumgebung wird ein Luftraum ohne Hindernisse simuliert. Das Luftraummodell ist als Würfel mit einer Kantenlänge von 100 m angelegt. Die Quader sind ebenfalls würfelförmig und haben eine Kantenlänge von 10 m. Bei dem simulierten unbemannten Luftfahrzeug handelt es sich um einen Quadrocopter mit einer maximalen Breite von 70 cm und einer Höhe von 20 cm. Im Vergleich zu den Luftfahrzeugen ist der Luftraum daher relativ grob strukturiert, für die Simulation der Testfälle ist eine feinere Modellierung aber nicht notwendig. Der Festlegung zur Auswahl der Quadergröße des Luftraummodells (s. Abschn. 3.1.3) wird ebenfalls entsprochen, alle möglichen Manöver können ohne ein Verlassen des Quaders geflogen werden.

Um alle möglichen Fälle des Szenarios einer Kollision in einem Quader zu testen, müssen theoretisch alle Permutationen der Einflugklassen beider unbemannter Luftfahrzeuge getestet werden. Bei zehn Einflugklassen ergibt sich so die Anzahl von einhundert Testfällen. Da die Implementierung allerdings relative Winkel der Luftfahrzeuge zueinander aus den Einflugklassen errechnet statt mit den absoluten Peilungen zu arbeiten, lassen sich die Ergebnisse von lediglich zehn Testfällen als allgemeingültig betrachten. Bei diesen zehn Testfällen fliegt ein Luftfahrzeug einen geraden Kurs und ein Zweites kreuzt diesen Kurs, aus jeweils der Peilung entsprechend einer der zehn Einflugklassen. Mit diesen zehn Testfällen ist sowohl die Erkennung von allen Kollisionen in einem Quader als auch die korrekte Einteilung aller Einflug-Äquivalenzklassen abgetestet. Letzteres ist möglich, da die Implementierung der Äquivalenzklasseneinteilung komplett ohne Betrachtung von vertikalen Bewegungen auskommt, mit Ausnahme der Einflugklassen für senkrechte Steig- und Sinkflüge, und daher keine zusätzlichen Testfälle für diagonale Steig- und Sinkflüge nötig sind.

Da sich allerdings die Ausweichmanöver für diagonale Steig- und Sinkflüge sowie diagonale Flüge in der horizontalen von den Ausweichmanövern auf geraden Flügen unterscheiden, kommen trotzdem noch weitere Testfälle hinzu. So werden alle zehn Testfälle nochmal um 45 Grad rotiert durchgeführt, um die Ausweichmanöver sowohl diagonal als auch gerade zu testen. Weitere vier Testfälle kommen hinzu, um die Manöver für diagonale Steig- und Sinkflüge zu testen, die durch das Weglassen der Eckverbindungen (s. Abschn. 3.5.2) lediglich bei 90, 180, 270 und 360 Grad möglich sind. Da die Implementierung hier so gewählt ist, dass die Manöver für diagonale Steig- und Sinkflüge in der Berechnung identisch sind, reicht jeweiles ein Steig- oder Sinkflug pro möglicher Äquivalenzklasse. Durch diese Erweiterung auf 24 Testfälle sind auch alle möglichen Ausweichmanöver um einen Quader abgedeckt.

Für die komplette Abdeckung der implementierten Basisregeln (s. Abschn. 3.5.3) sind noch elf weitere Testfälle nötig. Bei diesen Testfällen werden bestimmte Kombinationen von Flugrichtungen getestet, so dass beide Luftfahrzeuge in jedem Test unterschiedliche Flugrichtungen haben. So werden Situationen wie die Kreuzung in der diagonalen ohne gemeinsamen Quader, das Ausweichen bei stationären Luftfahrzeugen, Flugroutenendpunkten und Notfallpriorisierung sowie das Folgen und Ausweichen zweier senkrecht fliegenden Luftfahrzeuge getestet.

Da die Simulationsumgebung lediglich ein physikalisches Modell für unbemannte Luftfahrzeuge zur Verfügung stellt, welches sich auch nur begrenzt parametrisieren lässt, ergeben sich einige Einschränkungen bei den Testfällen für die Folgeregeln (s. Abschn. 3.5.3). Aufgrund der einheitlichen Physik der simulierten Luftfahrzeuge gibt es keine Geschwindigkeitsunterschiede, die ein Überholen nach Regel F.2 erlauben. Auch handelt es sich um Quadrocopter, sodass ein bewegungsloses Abwarten in der Luft möglich ist, was das Fliegen von Warte-Manövern oder Alternativkursen nach Regel F.4 unnötig macht. Abgetestet wird daher nur die Erkennung des Folge-Szenarios (Regel F.1) sowie das Verzögern des Fluges (Regel F.3).

Alle Testfälle sind so ausgelegt, dass das jeweils modellierte Kollisionsszenario tatsächlich eintritt. Dazu wurden die Testfälle einzeln auf Positions- und Zeitabstimmung überprüft. Wird die Kollision nicht richtig erkannt oder behandelt, wird die Separation definitiv verletzt. In der Regel findet sogar eine physische Kollision statt. So ist sichergestellt, dass die Testfälle auch wirklich ein Kollisionsszenario testen und ein Test nicht für erfolgreich erklärt werden kann, weil die räumlichen und zeitlichen Bedingungen für die Kollision versehentlich gar nicht erreicht werden.

6.2.3 Auswertung

Neben der vollständigen Testabdeckung bedarf es sinnvoller Bewertungskriterien, um aus den Ergebnissen der Testfälle fundierte Schlüsse ziehen zu können. Das wichtigste Kriterium hierbei ist, ob die Separation dauerhaft sichergestellt ist. Dies kann anhand des minimalen Abstands zwischen den Luftfahrzeugen im Test bewertet werden. Da würfelförmige Quader mit einer Kantenlänge von 10 m verwendet werden, wäre ein wünschenswerter Mindestabstand ebenfalls 10 m. Allerdings kommen sich die Luftfahrzeuge bereits auf einem erlaubten diagonalen Flug näher als die Kantenlänge der Quader. Daher wird der minimale Abstand für den Einhalt der Separation auf 5 m festgelegt. Da eine tatsächliche Kollision der Luftfahrzeuge erst bei 70 cm eintreten würde, ist eine gegenseitige Beeinträchtigung der Luftfahrzeuge so sicher ausgeschlossen.

Das zweite Bewertungskriterium ist, ob beide Luftfahrzeuge die korrekte Regel anwenden und das Manöver entsprechend der Regel ausgeführt wird. Daraus lässt sich ableiten, ob die Luftfahrzeuge die Äquivalenzklassen richtig bestimmt haben und die Zuordnung der Äquivalenzklassen zu Regeln und Manövern korrekt ist.

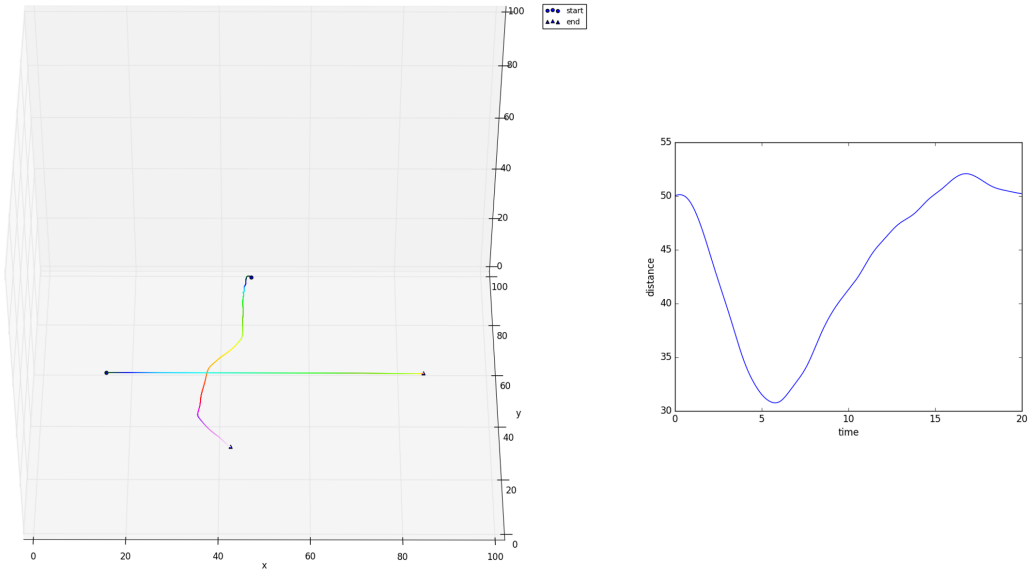

Abbildung 6.1: Beispiel eines Flugspur- und eines Distanzgraphen. Zeiten sind in Sekunden, Entfernungen in Metern angegeben.

Um Bewertungen anhand der Kriterien durchführen zu können, müssen die während der Testfälle aufgezeichneten Daten aufbereitet werden. Dazu werden sie mittels eines Python-Skripts eingelesen und mit Hilfe der matplotlib visualisiert. Dabei werden aus den Flugspuren 3D-Graphen gerendert, wobei ein Farb-Gradient der Kurven die Zeit repräsentiert. Die Entfernung in Metern zwischen den beiden Luftfahrzeugen wird in einem weiteren Graphen gegen die Zeit in Sekunden aufgetragen (s. Abb. 6.1).

Rel. Einflugwinkel UAV2	**Min. Abstand**	**Regel(n)**
0 Grad	14 m	F.1, F.3, B.6
0 Grad (rotiert)	14 m	F.1, F.3, B.6
0 Grad, sinkend	27 m	F.1, F.3, B.6
45 Grad	21 m	B.2, B.6
45 Grad (rotiert)	16 m	B.2, B.6
90 Grad	31 m	B.2, B.6
90 Grad (rotiert)	20 m	B.2, B.6
90 Grad, steigend	16 m	B.2, B.6
135 Grad	11 m	B.2, B.6
135 Grad (rotiert)	7 m	B.2, B.6
180 Grad	8 m	B.1, B.6
180 Grad (rotiert)	16 m	B.1, B.6
180 Grad, steigend	9 m	B.1, B.6
225 Grad	16 m	B.2, B.6
225 Grad (rotiert)	11 m	B.2, B.6
270 Grad	31 m	B.2, B.6
270 Grad (rotiert)	16 m	B.2, B.6
270 Grad, steigend	11 m	B.2, B.6
315 Grad	15 m	B.2, B.6
315 Grad (rotiert)	21 m	B.2, B.6
Senkrecht, steigend	12 m	B.3, B.6
Senkrecht, steigend (rotiert)	8 m	B.3, B.6
Senkrecht, sinkend	15 m	B.3, B.6
Senkrecht, sinkend (rotiert)	10 m	B.3, B.6

Tabelle 6.1: 24 Testfälle, Einflug nach Äquivalenzklassen. UAV1 flog waagerecht, entweder gerade oder diagonal (bei rotierten Testfällen).

Auf Basis der Graphen wurden die Testfälle ausgewertet. Die Ergebnisse für die 24 Äquivalenzklassen- und Manövertests (s. Tab. 6.1) zeigen, dass der geringste Minimalabstand während der Tests bei 7 m lag. Dabei treten die geringsten Minimalabstände bei Szenarien im Gegenanflug bzw. bei stumpfen Winkeln auf, da hier die relative Geschwindigkeit der unbemannten Luftfahrzeuge zueinander besonders hoch ist. Außerdem kommen die Luftfahrzeuge bei diesen

Szenarien später in Empfangsreichweite (s. Abschn. 3.3) als bei spitzen oder rechten Winkeln. Die Szenarien wurden von den unbemannten Luftfahrzeugen alle richtig erkannt und korrekt den Regeln zugeordnet. Die daraus resultierenden Manöver entsprachen ebenfalls den Regeln. Es gab bei diesen Testfällen trotz der guten Ergebnisse einige kleinere Probleme, darunter auch einen Mangel in der Regelspezifikation (s. Abschn. 6.2.4).

Flugbew. UAV1	**Rel. Einflugwinkel UAV2**	**Min. Abstand**	**Regel(n)**
Gerade, sinkend	0 Grad, steigend	19 m	F.1, F.3, B.6
Diagonal	90 Grad	26 m	B.2, B.6
Gerade, sinkend	180 Grad, sinkend	10 m	B.1, B.6
Diagonal	270 Grad	24 m	B.2, B.6

Tabelle 6.2: Vier Testfälle für Kreuzungen ohne gemeinsamen Quader in den Flugrouten.

Die elf weiteren Testfälle lassen sich aufteilen in vier Testfälle für Kreuzungen von Luftfahrzeugen, die ohne einen gemeinsamen Quader in den Flugrouten stattfinden (s. Tab. 6.2), und sieben Testfälle für weitere Regeln (s. Tab. 6.3).

Flugbew. UAV1	**Flugbew. UAV2**	**Min. Abstand**	**Regel(n)**
Gerade	Endet im Quader	9 m	B.3, B.6
Gerade	Stationär	7 m	B.3, B.6
Endet im Quader	Endet im Quader	30 m	F.3, B.6
Senkrecht, steigend	Senkrecht, sinkend	10 m	B.5, B.6
Senkrecht, steigend	Senkrecht, steigend	16 m	F.1, F.3, B.6
Senkrecht, steigend	Stationär	13 m	B.3, B.6
Gerade	90 Grad, Notlage	21 m	B.4, B.6

Tabelle 6.3: Sieben Testfälle für weitere Regeln.

Aus den Ergebnissen dieser Testfälle ergeben sich ähnliche Erkenntnisse wie bei den Äquivalenzklassentests. So liegt hier der kürzeste Minimalabstand bei 9 m. Viele Testfälle zeigen einen Minimalabstand von 10 m oder mehr, was sich aus den Szenarien ergibt. Es handelt sich um Szenarien mit stehenden, bremsenden oder langsamen, senkrecht fliegenden unbemannten Luftfahrzeugen. Hierbei ist die relative Geschwindigkeit gering und ein nötiges Ausweichen wird rechtzeitig erkannt. Auch bei diesen Tests wurden alle Szenarien richtig erkannt und den Regeln zugeordnet. Bei der Durchführung der Manöver zeigte sich allerdings in einem Testfall ein unspezifiziertes Verhalten (s. Abschn. 6.2.4).

6.2.4 Aufgetretene Probleme

Während der Testdurchführung sind einige Probleme aufgetreten, die in diesem Abschnitt erläutert werden.

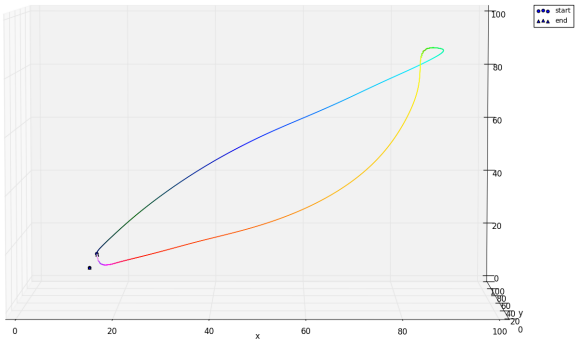

Abbildung 6.2: Flugspur der abweichenden Steig- und Sinkflüge auf identischem Flugpfad.

Bei den ersten Tests mit vertikalen Flügen kam es zu Problemen mit dem Wegpunktcontroller (s. Abschn. 4.2.1), so dass die Routenverfolgung nicht mehr funktionierte. Eine Auswertung der Flugspur (s. Abb. 6.2) zeigt, dass der physikalische Flugregler, der dem Wegpunktcontroller zugrunde liegt, diagonale Steig- und Sinkflüge nicht auf gerader Linie fliegt. Auf längeren Strecken führt dieses Verhalten dazu, dass das Luftfahrzeug so weit von der gewollten Fluglinie abweicht, dass es nicht mehr durch die Wegpunkte fliegt. Das führt dazu, dass der Wegpunktcontroller einen Wegpunkt nicht als erreicht erkennt und so die Flugroutenverfolgung nicht weiter funktioniert. Darüber hinaus folgt auf dem Verhalten eine Verletzung des Flugsicherheitskonzeptes, da in Quader eingeflogen wird, ohne dies mitzuteilen oder auf Belegung durch andere Luftfahrzeuge zu prüfen.

Der Wegpunktcontroller wurde auf Basis des physikalischen Flugreglers von MORSE implementiert. Daher liegt die übergeordnete Logik, die Festlegung der Zielkoordinaten und das Erkennen des Erreichens des Ziels, im Framework, die physikalische Flugregelung hingegen nicht. Physikalische Flugregler waren weder Teil der Evaluation und Bewertung der Simulatoren (s. Kap. 5) noch waren überhaupt alle Kandidaten damit ausgestattet. Der Flugregler von MORSE wurde im Rahmen der Evaluation allerdings eingesetzt und erwies sich als zuverlässig, sodass er als Basis für den Wegpunktcontroller verwendet wurde. Während der Evaluation der Simulatoren wurden allerdings hauptsächlich einfache Manöver geflogen, weshalb die Schwächen des Flugreglers nicht auffielen.

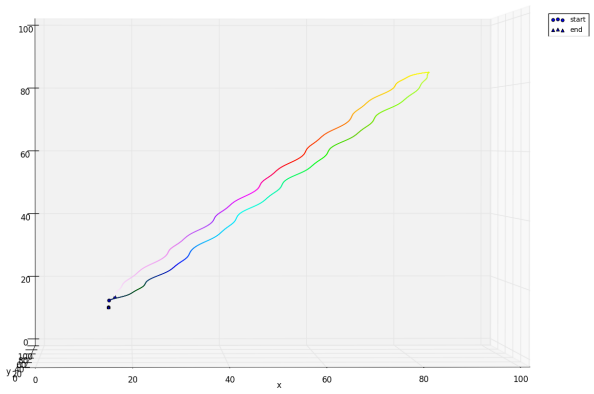

Abbildung 6.3: Flugspur von Steig- und Sinkflug mit geführter Rampe.

Da der Flugregler nicht Teil des Frameworks sondern in MORSE implementiert ist, sind die Möglichkeiten zur Einflussnahme begrenzt. Die Entwicklung eines eigenen physikalischen Flugreglers war zu diesem Zeitpunkt im Rahmen der Arbeit, sowohl zeitlich als auch aus Gründen des Umfangs, nicht mehr möglich. Versuche, das Reglerverhalten durch andere Parametrisierung zu verbessern, waren nicht erfolgreich. Es kam immer entweder im Steig- oder im Sinkflug zu Abweichungen von der geraden Linie. Ursächlich für das Problem scheint zu sein, dass der Flugregler nicht für die Verwendung mit einer konstant wirkenden, einseitigen Kraft, in diesem Fall die Schwerkraft, geeignet ist. Daher wurde die Optimierung für längere, gerade Strecken (s. Abschn. 4.2.1) bei vertikalen Strecken deaktiviert, so dass bei diesen Flugrouten jeder Wegpunkt als Zwischenziel für den Flugregler genutzt wird. Dadurch wird der Regler in einer Rampe geführt und hält so die Flugroute ein (s. Abb. 6.3), was aber eine deutlich niedrigere Fluggeschwindigkeit bei diagonalen Steig- und Sinkflügen zur Folge hat.

Ein weiteres Problem gibt es bei der Parametrisierung des Flugreglers für laterale Bewegungen. In manchen Testfällen mit längeren, geraden Flugstrecken zeigt sich ein leichtes Überschwingen des Reglers bei dem Erreichen des Zielpunktes. Ein weiterer Test mit einer sehr langen, geraden Flugbahn (s. Abb. 6.4) zeigt, dass die Weite des Überschwingens von der Länge der Flugstrecke abhängt. Bei entsprechend langen Flugstrecken kann so auch ein unzulässiges Verlassen des Zielquaders bis hin zu einer Kollision mit einem benachbarten Luftfahrzeug auftreten. Versuche, dieses Überschwingen über die Parametrisierung des Flugreglers zu verbessern, brachten nur begrenzten Erfolg. Wenn der Regler für eine getestete Entfernung optimiert ist, zeigen sich bei anderen Entfernungen Überschwinger oder schwingendes Verhalten vor Erreichen des Zielpunktes. Daher ist auch hier der Regler in MORSE nicht optimal ausgelegt. Um die Sicherheit

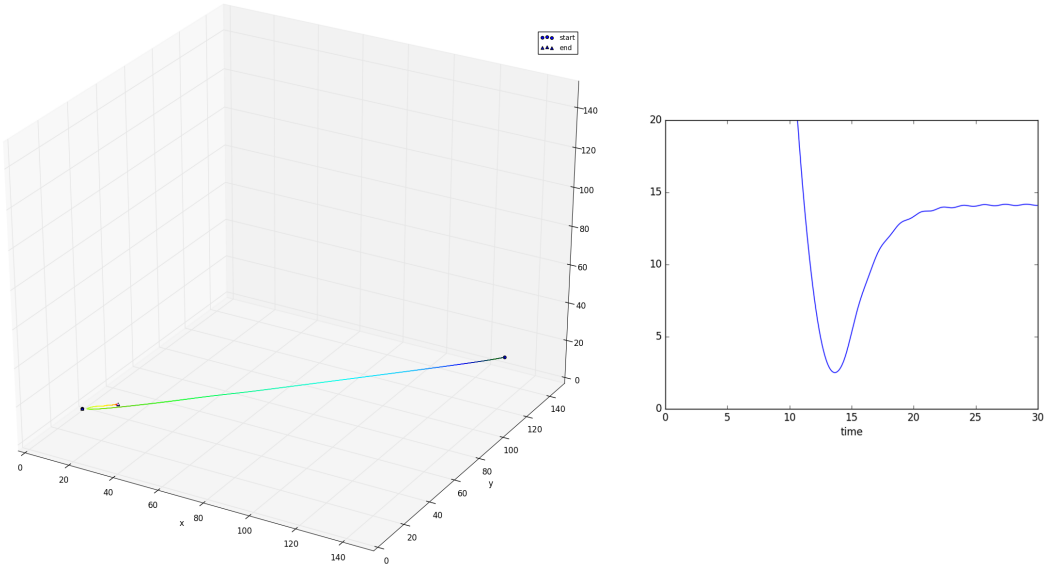

Abbildung 6.4: Flugspur einer langen Flugstrecke. Der Zielpunkt (Dreieck) wird überflogen, das benachbarte Luftfahrzeug (Punkt) fast getroffen. Min. Abstand 2,5 Meter.

des Flugsicherungskonzepts während der Simulationen trotz des unzureichenden Reglers sicherzustellen, muss dieser auf die längstmögliche Flugstrecke des jeweilig verwendeten Luftraummodells optimiert werden. Gleichzeitig muss die Sicherheitszeitspanne, die bei der Zeitschätzung der Ein- und Ausflugszeit der Quader eingerechnet wird (s. Abschn. 3.3), erhöht werden, um langsamere Anflüge durch schwingendes Verhalten zu kompensieren. Alternativ kann auch die Optimierung für gerade Flugstrecken (s. Abschn. 4.2.1) so adaptiert werden, dass bei Flugstrecken über einer gewissen Länge Zwischenziele eingebaut werden. Dadurch wird verhindert, dass der Regler bzw. das System sich so weit aufbauen können, dass ein Überschwingen entsteht.

Testfälle, welche die verschiedenen Varianten des Gegenanflugs prüfen, zeigen eine Problematik mit der praktischen Umsetzung der entsprechenden Regel B.1 auf. So wird zwar immer ausgewichen, allerdings weicht in manchen Durchführungen eines Testfalls nur ein Luftfahrzeug aus (s. Abb. 6.5) und in anderen, entsprechend der Regel, beide. Das Problem ist hierbei nicht abhängig vom Testfall sondern vom Timing. Ursache dafür ist, dass es vorkommen kann, dass ein Luftfahrzeug die Kollision detektiert und behandelt, ehe das zweite Luftfahrzeug überhaupt darüber informiert wird. Da durch das Handeln des ersten Luftfahrzeugs die Kollision aufgehoben wird, erkennt das zweite Luftfahrzeug sie nachfolgend nicht mehr und führt

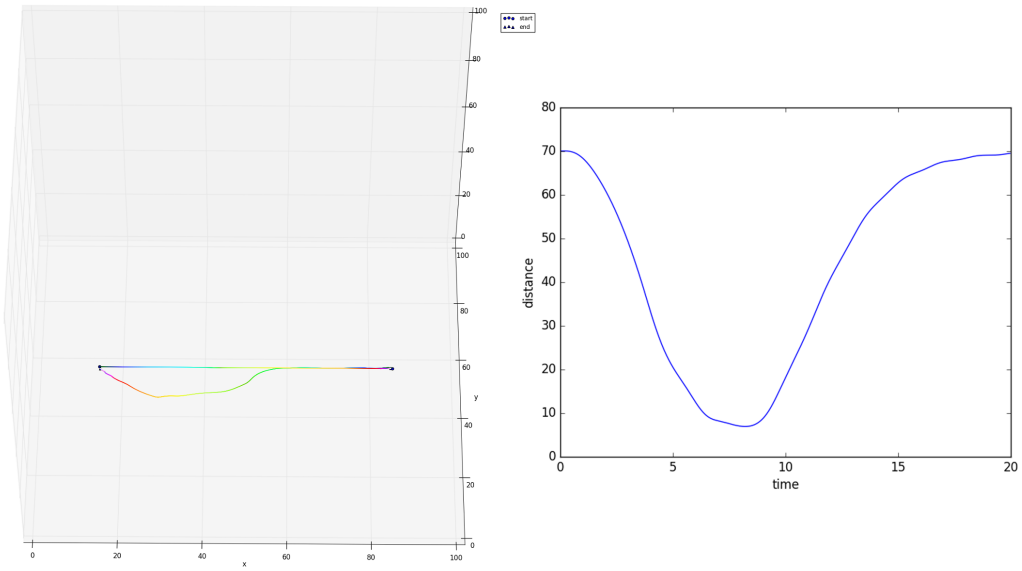

Abbildung 6.5: Flugspur und Distanz bei Gegenanflug.

kein Ausweichmanöver aus. Ob ein- oder beidseitig ausgewichen wird, ist also abhängig vom zeitlichen Ablauf der Kommunikation. Hier gibt es eine Race-Condition. Da ein Luftfahrzeug garantiert ausweicht, ist die Sicherheit der Separation dadurch nicht gefährdet. Auch wäre eine Behebung des Problems ohne Kommunikation in beide Richtungen nicht möglich. Daher sollte in diesem Fall nicht die Implementierung adaptiert werden, sondern die Spezifikation so erweitert, dass ein Luftfahrzeug auch auf ein Ausweichen verzichten kann, wenn ein Ausweichen des anderen Luftfahrzeugs bereits erkannt wurde.

Der Testfall „0 Grad, sinkend“ zeigt einen Fehler in der Spezifikation auf. Wie anhand der Flugspuren (s. Abb. 6.6) zu erkennen ist, stoppt das horizontal fliegende unbemannte Luftfahrzeug und wartet, bis das andere weit genug entfernt ist. Da beide Luftfahrzeuge die selbe Äquivalenzklasse beim Ein- und Ausflug haben, gehen die Luftfahrzeuge von einem „Folgen“-Szenario aus, da dies so in der Regel F.1 spezifiziert ist. Der Testfall „Grade, sinkend/0 Grad, steigend“ zeigt ein identisches Verhalten. Die Gefahr der Verletzung der Separation besteht in diesen Fällen nicht. Allerdings würde ein einfaches Ausweichmanöver zum Lösen der Kolliosion völlig reichen, was eine deutliche Zeitersparnis bringen würde. Daher sollte zur Behebung dieser Unschönheit die Spezifikation entsprechend angepasst und umgesetzt werden.

Bei der Durchführung des letzten Testfalls der erweiterten Tests, „Senkrecht, steigend/Stationär“, zeigte sich ein Fehlverhalten bei der Durchführung des Manövers. Es wurde ein

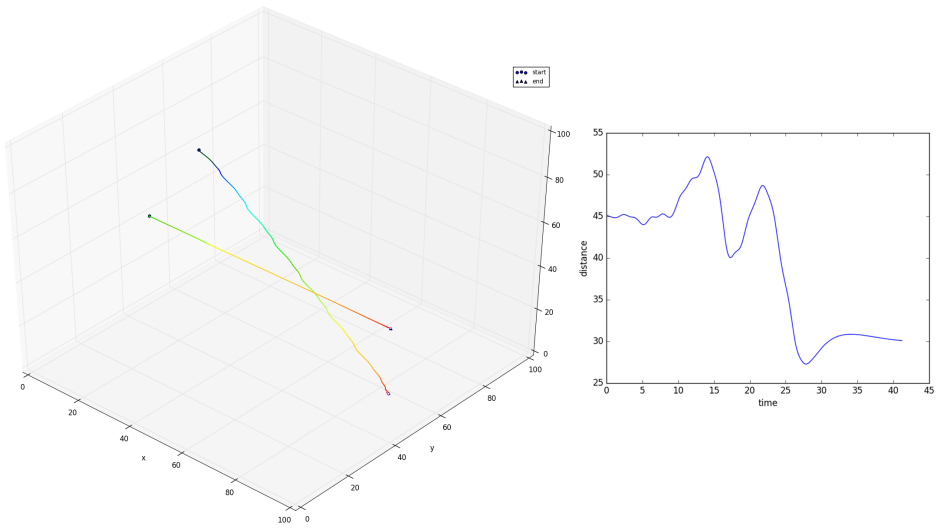

Abbildung 6.6: Flugspur und Distanz bei 360-Grad-Kreuzung. Das horizontale UAV wartet.

Ausweichen eingeleitet, allerdings entsprach die Ausweichrichtung nicht der Spezifikation. Stattdessen wurde ein diagonaler Nachbarknoten ausgewählt, der im Sink- bzw. Steigflug nicht direkt angeflogen werden kann, da ein Quaderübergang über Eck nicht gestattet ist (s. Abschn. 3.5.2). Daher fliegt das Luftfahrzeug eine Art Schraube, um den Ausweichquader zu erreichen und zum Ziel zurückzukehren (s. Abb. 6.7).

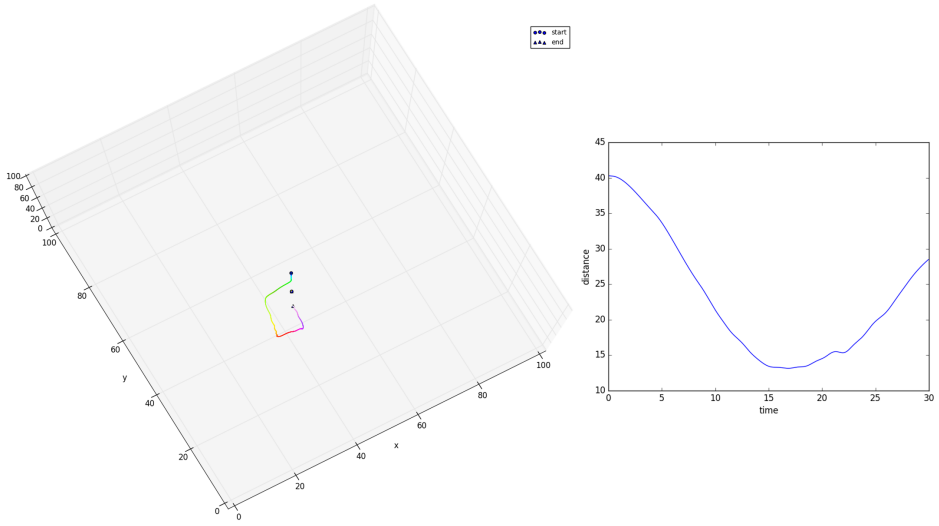

Abbildung 6.7: Flugspur (Sicht von oben) und Distanz bei vertikalem Ausweichen. Fälschlicherweise wird eine Schraube geflogen.

Aus diesem Verhalten entsteht keine Gefahr für die Separation, da ein Ausweichmanöver durchgeführt wird. Allerdings wird durch das undefinierte Ausweichen das Konzept zur Vermeidung von gegenläufigen Ausweichmanövern verletzt, so dass der Verkehr stärker, bis hin zu einem Deadlock, beeinträchtigt werden kann. Als Grund für das Fehlverhalten wurde ein Fehler in der Implementierung ausgemacht. Bei der Erkennung von stationären Luftfahrzeugen wird nicht berücksichtigt, ob der eigene Kurs ein senkrechter Steig- oder Sinkflug ist, bevor er zur Berechnung eines Ausweichkurses heran gezogen wird. Allerdings ist der Algorithmus zur Berechnung eines Ausweichkurses bei eigenen senkrechten Steig- und Sinkflügen nicht verwendbar und es entsteht ein unspezifiziertes Ergebnis. Zur Behebung dieses Problems muss die Implementierung korrigiert werden.

6.3 Simulation einer Anwendung

Der zweite Testansatz soll zeigen, dass das Flugsicherungskonzept auch in realen Anwendungen funktioniert. Für diesen Anwendungstest soll ein Szenario ähnlich einer Anwendung im späteren angedachten Einsatzgebiet, den business-to-business-Anwendungen, simuliert werden (s. Kap. 1). Als Szenario wurde daher ein Logistiksystem mit unbemannten Luftfahrzeugen in einer 4872 m^2 großen Produktions- und Lagerhalle gewählt.

6.3.1 Strategie

Ziel des Tests ist es, nachzuweisen, dass das Flugsicherungskonzept auch außerhalb von definierten Testfällen zuverlässig funktioniert. Da die Testfälle definierte, statische Situationen prüfen, liegt der Fokus des Anwendungstests auf dynamischen Situationen. Dazu sollen die unbemannten Luftfahrzeuge fiktive Transportaufträge zwischen mehreren definierten Start- und Landeplätzen durchführen. Die Aufträge werden dabei zufällig generiert, wobei darauf geachtet wird, dass ausreichend Verkehr im Luftraum entsteht. Be- und Endladezeiten sowie Ladezeiten für Akkus werden ebenfalls zufällig festgelegt.

6.3.2 Durchführung

Für die Anwendungssimulation wurde eine Anwendung geschrieben, die die Flottenverwaltung übernimmt und automatisch Flugaufträge generiert und an freie unbemannte Luftfahrzeuge

übermittelt. Es werden acht unbemannte Luftfahrzeuge eingesetzt. In der Produktionshalle sind 17 Zielorte an Produktionsmaschinen oder Lagerplätzen und acht Basisstationen für die unbemannten Luftfahrzeuge in freien Bereichen definiert. Ein Flugauftrag beginnt immer auf einer Basisstation und führt zu einem bis vier Zielorten. Nach Erreichen des letzten Zielorts kehrt das unbemannte Luftfahrzeug zu seiner Basisstation zurück und kann weitere Aufträge ausführen.

Für die Modellierung des Luftraums wurden würfelförmige Quader mit einer Kantenlänge von 2 m verwendet, sodass sich die 8 m hohe Produktionshalle in 4 Flugflächen (s. Abschn. 2.2.2) aufteilt. Der Luftraum ist aufgrund des Anwendungsszenarios deutlich feiner strukturiert als während der Testfälle, erfüllt aber weiterhin die festgelegte Anforderung, dass die Quader für alle Manöver ausreichend groß sein müssen (s. Abschn. 3.1.3). Von den vier Flugflächen dienen die oberen beiden den Streckenflügen und sind über sämtlichen hindernisfreien Luftraumbereichen der Produktionshalle durchgehend vorhanden. Die unteren zwei Flugflächen sind nur vereinzelt vorhanden und dienen Landeanflügen bzw. als Landeplätze.

Die Produktionshalle verfügt über verschiedene Produktionsmaschinen, die sich über die gesamte Fläche verteilen. An drei Standorten sind kleinere Regallager vorhanden, die lediglich die Verwendung der obersten Flugfläche erlauben. Weiterhin ist im Zentrum der Halle ein automatisches Hochregallager installiert, welches nicht überflogen werden kann. Daher wird der Luftverkehr beim Wechsel der Hallenseite auf den Korridor in der Mitte des Hochregallagers sowie den schmaleren Hallenteilen jeweils rechts und links daneben eingeschränkt (s. Abb. 6.8).

Da das Flugsicherungskonzept lediglich dauerhafte Separation sicherstellen soll, aber keine Verkehrs- oder Flusssteuerung hat und daher nicht gegen Deadlocks abgesichert ist, bedarf es weiterer Maßnahmen. Einerseits wurde bei der Auslegung des Luftraummodells der Produktionshalle darauf geachtet, dass keine Sackgassen existieren, in denen unbemannte Luftfahrzeuge durch andere „blockiert“ werden können. Alle Ziellandeplätze haben daher mindestens zwei Abflugmöglichkeiten und die Landeplätze, die als Basis für die Luftfahrzeuge dienen, sind eindeutig ihrem Luftfahrzeug zugeordnet. Darüber hinaus verfügt die Flottenverwaltung über einen Mechanismus um Deadlocks aufzulösen. Wird ein Ziel nicht innerhalb einer Zeitspanne von drei Minuten erreicht, wird der Flugauftrag abgebrochen und das Luftfahrzeug kehrt zu seiner Basisstation zurück. Eventuell blockierte Luftfahrzeuge kommen dadurch dann wieder frei und können den blockierten Bereich verlassen.

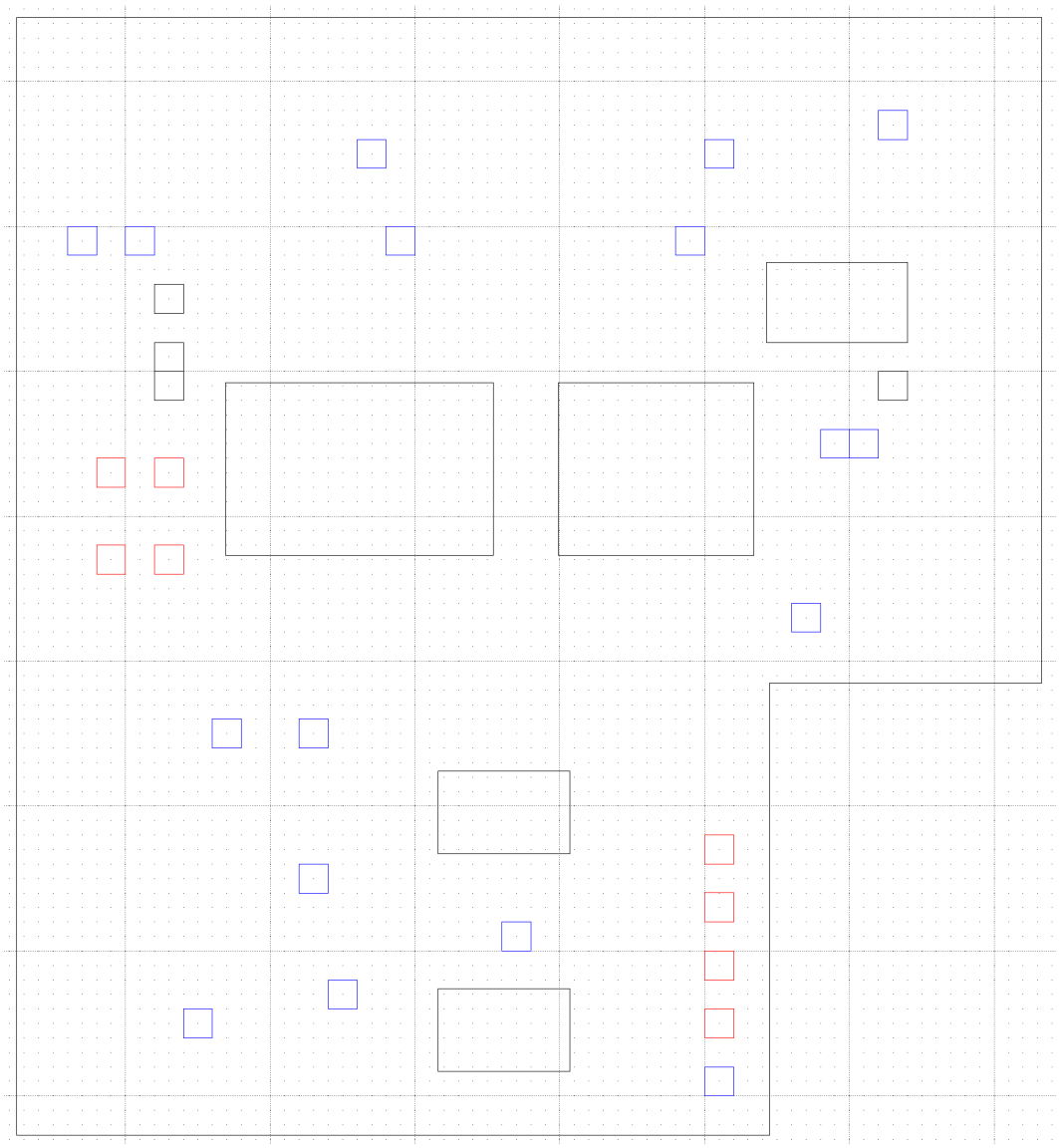

Abbildung 6.8: Layout der Halle. Blau: Zielorte, Rot: Basisstationen, Schwarz: Hindernisse.

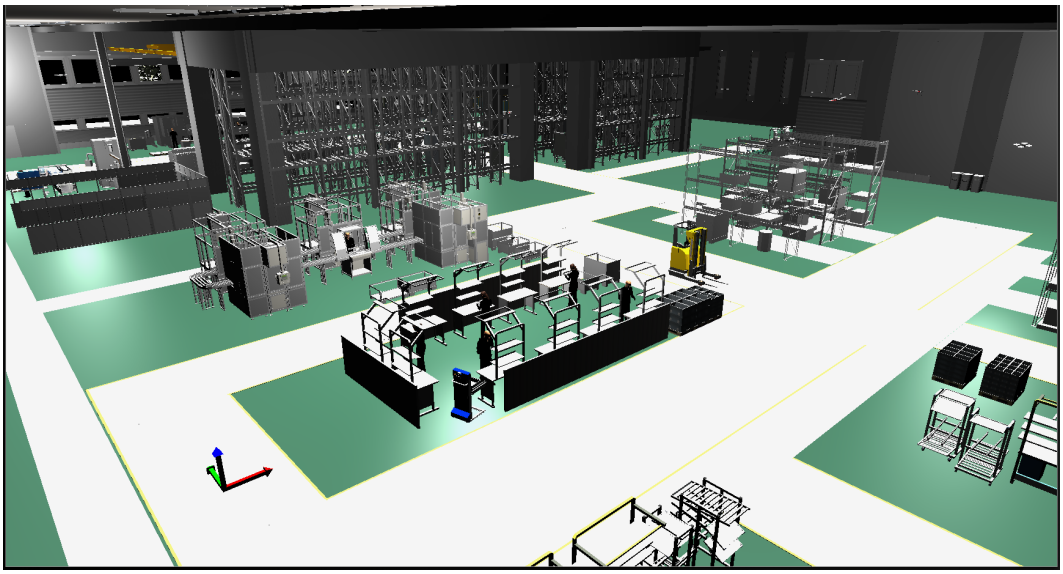

Abbildung 6.9: Simulation einer Drohnenanwendung in einer Produktionshalle.

Sowohl die Verteilung und Modellierung der Landeplätze als auch das Layout der Produktionshalle stellen sicher, dass es bei der Abhandlung der Flugaufträge zu Kollisionsszenarien kommt. So wird beispielsweise durch die Einschränkung der Nord-Süd-Verbindungen durch das Hochregallager, wobei in einem Durchgang sogar einige Basisstationen liegen, sowie das gleichzeitige Anfliegen derselben Landeplätze ein kollisionsfreier Flug deutlich weniger wahrscheinlich als einer mit Kollisionsszenarien. Allerdings kann die Anwendung nicht sicherstellen, dass alle Kollisionsszenarien eintreten bzw. alle Regeln angewendet werden. So sind auch hier die Luftfahrzeuge gleich schnell, so dass keine Überholmanöver geflogen werden. Auch wurde auf eine Simulation von Notlagen verzichtet, da dieses Szenario durch die Testfälle bereits abgedeckt ist und einen erhöhten Aufwand in der Flottenverwaltung bedeutet hätte.

Während der Tests werden alle Flugspuren als Basis für eine Auswertung aufgezeichnet. Anhand der Flugspuren lässt sich die Entfernung zwischen den unbemannten Luftfahrzeugen zu jedem Zeitpunkt berechnen, woraus sich auch hier erkennen lässt, ob die Separation verletzt wurde. Weiterhin führt die Flottenverwaltung eine Statistik über gestartete Aufträge, während der Flüge erreichte Zielpunkte sowie den Ausgang der Aufträge, daher ob ein Auftrag erfolgreich beendet oder wegen Deadlock abgebrochen wurde.

6.3.3 Auswertung

Da die Quadergröße des Luftraummodells der Produktionshalle anders gewählt ist als im Luftraummodell der Testfälle, müssen auch die Bewertungskriterien des Anwendungstests anders ausgelegt sein. So ist bei 2 m Kantenlänge der Quader der theoretische minimale Abstand bei diagonalen Flügen nur 1.4 m. Durch Schwingungen und andere Eigenschaften des Flugreglers wird dieser Abstand weiter reduziert, sodass die minimale Entfernung zur Bewertung der Separation auf 1 m festgelegt wird.

Zur Auswertung kommt wieder ein Python-Skript zum Einsatz. Dieses Skript berechnet anhand der Log-Files (s. Abschn. 6.1) den minimalen Abstand zwischen den unbemannten Luftfahrzeugen in jedem Simulationsschritt und plottet mithilfe der matplotlib einen Graphen der minimalen Entfernung über die Zeit. Sobald die Entfernung zwischen zwei Luftfahrzeugen geringer als 1 m ist, werden außerdem automatisch die Flugpfade der beteiligten Luftfahrzeuge als 3D-Graphen geplottet. Dies geschiet für einen Zeitabschnitt von ± 10 Sekunden um den Zeitpunkt der Distanzunterschreitung. Die 3D-Plots sind analog zu den Plots der systematischen Tests (s. Abschn. 6.2.3).

Durch den Wechsel des Szenarios, von dem offenen Szenario der Testfälle zu der kleineren Produktionshalle für die Anwendungstests, müssen diverse Änderungen am System vorgenommen werden. Die größte Änderung ist hierbei das neue Luftraummodell mit veränderter Quadergröße sowie die daraus folgenden Anpassungen, beispielsweise die Erhöhung der Anzahl der per UAV2UAV-Kommunikation veröffentlichten Quader des eigenen Flugpfades. Daher zeigten sich in den kürzeren, initialen Tests diverse Schwächen, die iterativ behoben wurden mussten. Die meisten Probleme resultierten aus Ungenauigkeiten im Luftraummodell, so dass unbemannte Luftfahrzeuge mit Objekten in der Produktionshalle kollidierten und daraufhin abstürzten oder ihren Flugpfad verließen. Weitere Probleme sind im Abschnitt 6.3.4 beschrieben. Lediglich eins der aufgetretenen Probleme machte eine Änderungen am eigentlichen Algorithmus der Flugsicherung nötig, die anderen beschränkten sich auf einige Implementierungsdetails. Bei der Änderung am Algorithmus handelt es sich um einen Implementierungsfehler bei der Erkennung einer Regel, der in den systematischen Tests nicht auftrat. Daher entspricht der getestete Algorithmus grundsätzlich dem, der auch in den systematischen Tests zum Einsatz kam und die Ergebnisse können zusammen betrachtet werden.

Der eigentliche Anwendungstest ging über 26 Stunden (Echtzeit). Während dieser Zeit wurden 2071 Flugaufträge erteilt, von denen 1858 erfolgreich beendet und 212 abgebrochen wurden.

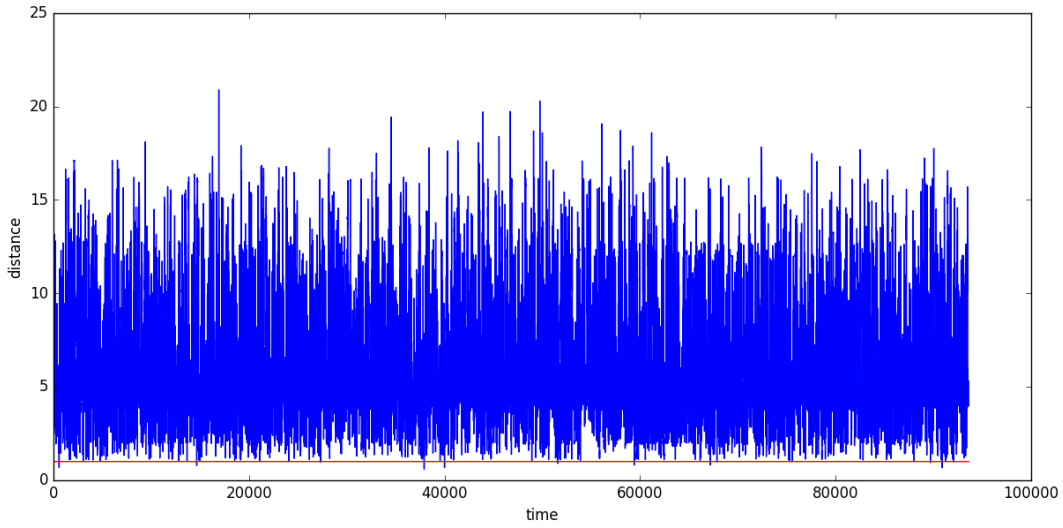

Abbildung 6.10: Minimaler Abstand zwischen allen Luftfahrzeugen über die Zeit. Rot: minimal erlaubte Entfernung. Zeitliche Auflösung (Simulationsschritt): 16,67 ms.

Ein Flugauftrag wurde nicht beendet, da zwischen dem Einstellen der Auftragsvergabe und dem Abschalten des Systems nicht ausreichend Wartezeit eingeplant wurde. Bei diesen 2071 Flugaufträgen wurden 4834 Zielpunkte erfolgreich angeflogen und mit den Rückflügen zu den Basen insgesamt 6905 Strecken geflogen.

Im Gegensatz zu den initialen Tests kam es zu keinerlei Abstürzen von den unbemannten Luftfahrzeugen. Im Graphen mit der minimalen Entfernung (s. Abb. 6.10) ist zu erkennen, dass die Separation die meiste Zeit eingehalten wurde, allerdings zeigen sich auch vereinzelt Verletzungen der minimalen Separationsentfernung. Die Auswertung der Daten zeigte sechzehn Situationen, bei denen die minimale Separationsentfernung von 1 m unterschritten wurde (s. Tab. 6.4). Bei acht Situationen war die Verletzung der Separation allerdings minimal, die minimale Separationsentfernung wurde um weniger als 6 cm unterschritten. Bei weiteren vier Situation war die minimale Entfernung zwischen den beiden Luftfahrzeugen immernoch größer als die Kollisionsentfernung von 70 cm. Lediglich in vier Situationen waren sich die Luftfahrzeuge so nahe, dass bei ungünstiger relativer Position zueinander eine Kollision hätte passieren können.

Eine genaue Analyse der Flugpfade der beteiligten Luftfahrzeuge zu diesen Zeitpunkten ergab allerdings, dass in keinem dieser sechzehn Fälle ein Fehlverhalten des Flugsicherungsalgorithmus zugrunde lag. Stattdessen beruhen diese Verletzungen der Separation alle auf Diskrepanzen

UAV1	UAV2	Min. Entfernung	Zeit
1	7	0,68 m	0:09:44
3	7	0,99 m	0:50:50
4	7	0,78 m	4:04:17
3	4	0,99 m	6:41:08
0	6	0,95 m	7:35:09
2	7	0,58 m	10:31:56
4	7	0,67 m	11:06:36
2	4	0,89 m	14:19:54
6	7	0,80 m	16:29:50
3	6	0,81 m	18:39:52
0	7	0,97 m	18:50:57
5	6	0,99 m	20:59:33
0	2	0,99 m	24:27:02
6	7	0,94 m	24:54:46
2	7	0,66 m	25:14:47
2	7	0,97 m	25:34:43

Tabelle 6.4: 16 Unterschreitungen der minimal erlaubten Entfernung. Rot: Kollisionsgefahr

zwischen dem vorgegebenen Flugpfad und dem vom Flugregler tatsächlich geflogenen Flugpfad. Diese Diskrepanzen wurden auch bei den initialen Tests schon beobachtet und werden im Abschnitt 6.3.4 erläutert. Eine besonders ungünstige Annäherung der Luftfahrzeuge ergibt sich aus zwei verschiedenen Kombinationen der Diskrepanzen und bestimmtem Flugsituationen, die allerdings erst im finalen Test entdeckt wurden.

6.3.4 Aufgetretene Probleme

Obwohl viele Probleme schon durch die Durchführung der systematischen Tests erkannt und teilweise behoben wurden, sind während des Anwendungstests noch weitere aufgetreten. Diese Probleme werden in diesem Abschnitt erläutert.

Zu Beginn der initialen Tests gab es einige Abstürze. Neben den im vorherigen Abschnitt bereits erwähnten Ungenauigkeiten bei der Modellierung des Luftraums lagen diese Abstürze auch an einem Fehler bei der Erkennung des Erreichens eines Quaders. Die Flugpfadverfolgung erkannte das Erreichen des Quaders bereits, sobald sich der Mittelpunkt des unbemannten Luftfahrzeugs innerhalb des Quaders befand. Bei Quadern auf gerader Strecke stellt das kein Problem dar. Bei Quadern, an denen allerdings die Flugrichtung geändert werden muss, ist das

fatal. Durch den Fehler wurden Manöver bereits ausgeführt, während ein Teil des Luftfahrzeugs noch gar nicht innerhalb des Quaders war. Daraus konnte resultieren, dass die Luftfahrzeuge in engen Kurven gegen Objekte in der Produktionshalle flogen oder anderen Luftfahrzeugen nicht weit genug ausgewichen sind. Da bei den Testfällen keine Objekte in den Luftraum ragten und die Toleranzen durch die 10-Meter-Quader deutlich höher waren, wurde das Problem dort nicht erkannt. Zum Beheben dieses Problems wurde die Erkennung des Erreichens eines Quaders, in dem ein Manöver stattfinden soll, so angepasst, dass ein Luftfahrzeug diese Quader erst als erreicht erkennt, wenn es sich der Mitte des Quaders bis auf 50 cm genähert hat.

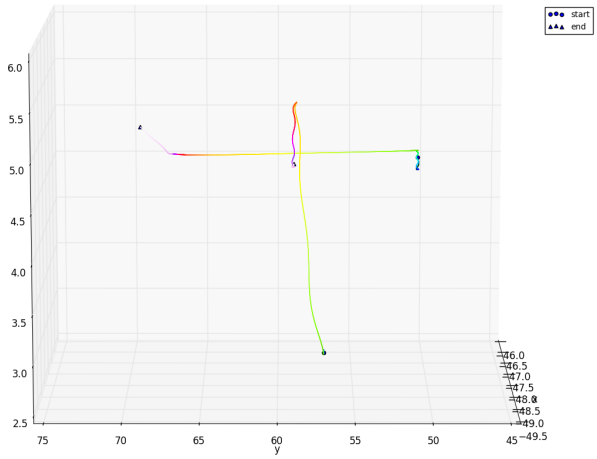

Abbildung 6.11: Luftfahrzeug 1 steigt diagonal in die selbe Richtung wie Luftfahrzeug 2, Kollision wird nicht erkannt, beide fliegen zeitgleich in den selben Quader.

Der gravierendste Fehler, der bei den initialen Tests auffiel, lag im Flugsicherungsalgorithmus. Bei Kollisionsszenarien, in denen sich Luftfahrzeuge folgen, werden in mehreren aufeinander folgenden Quadern Kollisionen detektiert. Um den Aufwand zu reduzieren wird das komplette Ausweichmanöver bei der ersten Erkennung des „Folgen“-Szenarios berechnet. In allen weiteren Quadern wird überprüft, ob die Luftfahrzeuge aus demselben Quader in den aktuell zu prüfenden Quader einfliegen. Ist das der Fall, kann der Algorithmus davon ausgehen, dass das Ausweichmanöver bereits im Quader vorher berechnet wurde und bricht ab. Allerdings war die Überprüfung, ob beide Luftfahrzeuge aus demselben Quader einfliegen, nicht korrekt implementiert. Alle Luftfahrzeuge, die von oben diagonal, unten diagonal oder gerade aus der selben Einflugrichtung in den Quader einflogen, wurden als „aus dem selben Quader kommend“ betrachtet, da sie dieselbe Einflugklasse haben. Allerdings kommen die Luftfahrzeuge, die aus zwei unterschiedlichen der drei Steigungswinkel einfliegen, nicht aus demselben Quader. Durch diesen Fehler wurde die Flugsicherung in diesen Fällen komplett ausgehebelt, da der

Algorithmus immer „kein Manöver“ entschied, sodass die Luftfahrzeuge zeitgleich in dieselben Quader einfliegen konnten (s. Abb. 6.11). Dieser Fehler war aufgrund seines Ausmaßes für das Scheitern von einigen initialen Tests verantwortlich und wurde nach Entdeckung sofort behoben. Da ein Folgen mit verschiedenen, initialen Einflugwinkeln in den Testfällen nie simuliert wurde, konnte dieser Fehler dort nicht entdeckt werden. Allerdings sollte hierfür ein Testfall geschaffen werden. Für andere Kollisionsszenarien hingegen ist dieser Fehler nicht zu erwarten, da nur bei „Folgen“ eine Überprüfung gemacht wird, die erlaubt, den Algorithmus ohne Ergebnis abzubrechen.

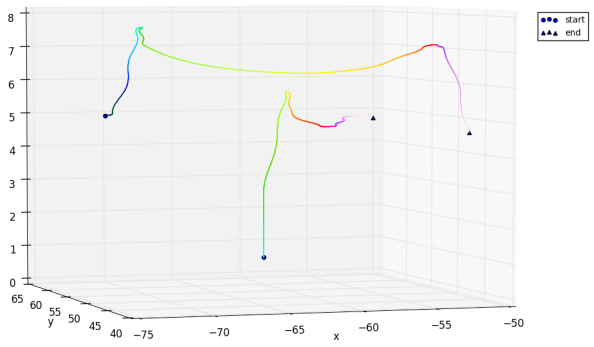

Abbildung 6.12: Reglerverhalten: Absinken in der Geraden, Überschwingen beim Steigen.

Neben den bereits bei den systematischen Tests aufgefallenen Schwächen des Flugreglers (s. Abschn. 6.3.3) zeigten sich sowohl bei den initialen Tests als auch beim Anwendungstest weitere Probleme. Initial gab es häufig Flugabbrüche, da das Luftfahrzeug seinen Flugpfad verlassen hatte und die Flugpfadverfolgung daraufhin nicht mehr funktionierte. Dieses Verhalten stellt eine Verletzung der Flugsicherung dar, da in Luftraum eingeflogen wird ohne diesen offiziell über die Flugsicherung abzusichern. Eine Analyse der Flugspuren zeigte, dass der Flugregler den Verlust an Höhe durch das Anstellen des Luftfahrzeugs zum Beschleunigen nicht komplett ausgleicht (s. Abb. 6.12). Dadurch verliert das unbemannte Luftfahrzeug auf längeren Flügen so viel Höhe, dass es den Flugpfad nach unten verlässt. Während der initialen Tests wurde versucht, dem Phänomen mit dem Anpassen der Flugreglerparameter zu begegnen. Dazu wurden die Parameter empirisch auf eine aggressivere Höhenregelung optimiert. Allerdings lassen sich die Parameter nur begrenzt dafür nutzen, da sonst beim senkrechten Steig- und Sinkflug zu große Überschwinger entstehen. Daher wurde zusätzlich die Flugroutenoptimierung (s. Abschn. 4.2.1) angepasst, sodass sie längere, gerade Flüge mit Zwischenzielen unterbindet. Damit konnte verhindert werden, dass die Luftfahrzeuge ihre Flugpfade nach unten verlassen.

Bei den systematischen Testfällen ist dieses Verhalten wegen der großen Toleranzen durch die größeren Quader nicht von Belang gewesen und wurde nicht entdeckt.

Bei der Anwendungssimulation zeigte sich, dass das Verlassen der Flugroute zwar nicht mehr vorkam, das Abweichen von der idealen Flugroute aber trotzdem problematisch sein kann. Bei einer der Unterschreitungssituationen zeigte sich, dass die Luftfahrzeuge sich so nahe kamen, weil das obere beim geraden Flug innerhalb des erlaubten Rahmens nach unten abgewichen war und das untere einen Überschwinger beim senkrechten Steigflug hatte. Auf diese Weise näherten sich die Luftfahrzeuge bis auf 58 cm an (s. Abb. 6.12), die kürzeste Entfernung zwischen zwei Luftfahrzeugen während des gesamten Tests (s. Tab. 6.4). Da beide Verhaltensweisen in dieselbe Richtung wirken, lässt sich das Problem nicht durch Anpassen der Flugreglerparameter reduzieren. Es handelt sich bei dem Verhalten allerdings nicht um eine Fehlfunktion des Flugsicherungsalgorithmus, daher ist diese Unterschreitungssituation zumindest für die Bewertung des Algorithmus nicht relevant. Bei der Festlegung der Quadergröße (s. Abschn. 3.1.3) sollten allerdings neben den physikalischen Eigenschaften des Luftfahrzeugs auch die Eigenschaften des Flugreglers berücksichtigt werden. Für spätere Anwendungen, die gewisse Mindestabstände bei der Separation verlangen, sollte daher darauf geachtet werden, dass entweder der Flugregler verbessert oder die Größe der Quader entsprechend großzügig gewählt wird, um eine größere Separationsentfernung sicherzustellen.

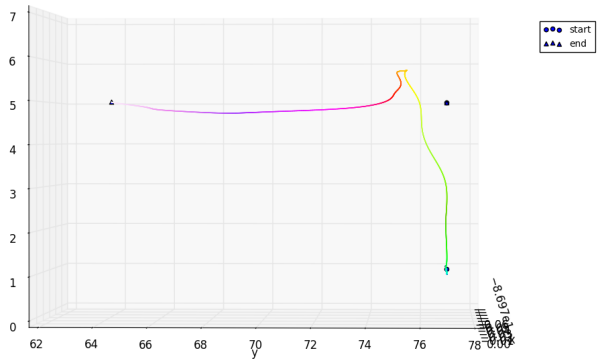

Abbildung 6.13: Luftfahrzeug stationär im Mittelpunkt eines Quaders (y=77), zweites Luftfahrzeug steigt zu schnell und verfehlt den Mittelpunkt des Zielquaders (y=75).

Ein weiteres Verhalten, welches ebenfalls in der Höhenregelung des Flugreglers seinen Ursprung hat, ist für die restlichen Unterschreitungssituationen verantwortlich. Durch die aggressivere Auslegung der Flugreglerparameter zum Ausgleich des Absinkens bei geraden Flügen ist der Regler zu aggressiv bei Steig- und Sinkflügen. Das resultiert zum einen in einem Über-

schwingen bei allen Steig- und Sinkflügen, zum anderen in einer nicht optimalen Flugbahn bei diagonalen Steig- und Sinkflügen. Bei diagonalen Steig- und Sinkflügen ergibt sich die Flugbahn des Luftfahrzeugs aus dem kombinierten Einfluss von sowohl dem Höhenregler als auch dem lateralen Regler des Flugreglers. Ist nun der Höhenregler aggressiver, ist die Flugbahn steiler als die optimale diagonale Strecke zwischen Start- und Zielquader. Befindet sich im Quader über bzw. unter dem Startpunkt ein Luftfahrzeug, kommen sich die Luftfahrzeuge trotz korrekter Flugsicherung zu nahe (s. Abb. 6.13).

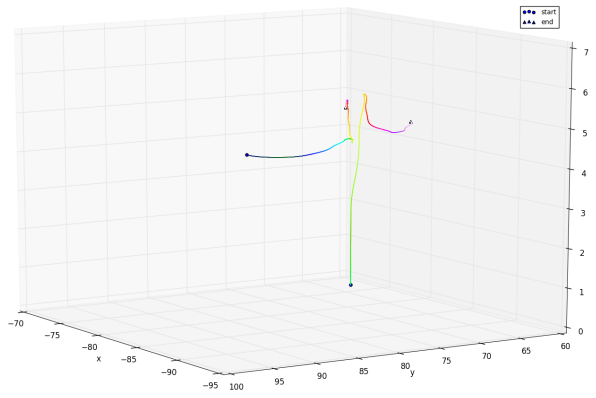

Abbildung 6.14: Luftfahrzeug steigt diagonal, zweites Luftfahrzeug überfliegt es diagonal.

Noch näher können sich die Luftfahrzeuge kommen, wenn das Luftfahrzeug über bzw. unter dem diagonal steigenden Luftfahrzeug nicht stationär ist oder gerade vorbei fliegt, sondern sich in einem Diagonalflug zwischen dem oberen bzw. unteren Quader und dem Zielquader befindet (s. Abb. 6.14). Diese Konstellation ist für die weiteren drei Unterschreitungssituationen mit Kollisionsgefahr verantwortlich (s. Tab. 6.4). Auch bei dieser Situation arbeitet der Flugsicherungsalgorithmus korrekt, der Fehler liegt in der Diskrepanz zwischen gewünschter Flugbahn und realer Flugbahn. Diese wird ausgelöst durch den problematisch einzustellenden Flugregler.

7 Schluss

In diesem abschließenden Kapitel werden die vorherigen Kapitel der Arbeit zusammengefasst, das Ergebnis der Arbeit im Hinblick auf die Problemstellung bewertet und ein Ausblick auf zukünftige Entwicklungen gegeben.

7.1 Zusammenfassung

Ziel der Arbeit war es, ein Flugsicherungssystem für autonome unbemannte Luftfahrzeuge zu entwickeln. Dieses Flugsicherungssystem soll autonom funktionieren und auf reine Multi-Drohnen-Anwendungen reduziert sein. Zur Überprüfung des Flugsicherungssystems sollte dieses in einer simulierten Beispielanwendung getestet werden.

Um ein fundiertes Konzept entwickeln zu können, wurden als erstes verschiedene, grundlegende Themenbereiche erforscht. Besonders die konventionelle Luftfahrt und die dort verwendeten Flugsicherungsverfahren sowie automobile Technologien, insbesondere die Car2Car-Kommunikation, standen hierbei im Fokus. Aber auch vergleichbare Arbeiten im Bereich der Flugsicherung für unbemannte Luftfahrzeuge, namentlich NASA UTM und SESAR U-Space, wurden betrachtet.

Nach Betrachtung der Grundlagen wurde ein Konzept für eine autonome Flugsicherung entworfen. Hierbei wurde, im Gegensatz zur konventionellen Flugsicherung sowie den Projekten U-Space und UTM, ein dezentraler Ansatz gewählt, der auf eine übergeordnete Luftverkehrskontrolle verzichtet. Begründet ist die Auswahl damit, dass ein dezentrales System ein einfacheres Deployment bietet, leichter skalierbar ist und Single Points of Failure vermeidet. Basierend auf diesem Ansatz wurde ein Luftraummodell entwickelt, das die Grundlage für die Separation der Luftfahrzeuge bildet und als Basis für die weiteren Algorithmen des Flugsicherungskonzepts dient. Das Luftraummodell teilt den Luftraum in Quader auf, die wie ein Voxelspace verwaltet werden. Dieses Modell zeigte sich in der theoretischen Betrachtung als effizienter für den An-

wendungsfall dieser Arbeit als andere, ähnliche Modelle wie ein Octree-Modell oder verlinkte Quader. Aus dem Luftraummodell ergibt sich die Grundlage der Separation der Luftfahrzeuge, denn ein Quader des Modells darf niemals von zwei Luftfahrzeugen zur selben Zeit beflogen werden. Die UAV2UAV-Kommunikation stellt ein Kernstück des Konzepts dar, das basierend auf der Car2Car-Kommunikation entwickelt wurde, um den unbemannten Luftfahrzeugen einen Informationsaustausch als Basis für die Flugsicherung zu ermöglichen. Bei dem Protokollentwurf wurde ein aktiver Ansatz gewählt, bei dem die Kommunikationsteilnehmer periodisch ihre Position und weitere Informationen über die Kommunikation verbreiten. Anhand der über die UAV2UAV-Kommunikation erhaltenen Informationen kann der nachfolgend konzeptionierte Kollisionserkennungsalgorithmus den eigenen Flugpfad auf Kollisionen überprüfen und gegebenenfalls den Kollisionsvermeidungsalgorithmus aktivieren. Zur Kollisionsvermeidung boten sich zwei Konzepte an, ein kooperatives, bei dem sich die Luftfahrzeuge aktiv über eine Lösung abstimmen, sowie ein regulatives, bei dem eindeutige Verhaltensregeln für alle möglichen Kollisionssituationen festgelegt werden, nach denen die Luftfahrzeuge handeln müssen, um eine Kollision zu vermeiden. Das kooperative Konzept bringt allerdings Overhead im Kommunikationsprotokoll und dem Kollisionsvermeidungsalgorithmus mit sich, daher wurde das regulative Konzept ausgewählt. Um eine eindeutige Spezifikation für die Ausweichregeln zu erreichen, wurden als erstes die möglichen Kollisionsszenarien ermittelt. Um dabei Overhead in der Spezifikation zu vermeiden, wurden die Szenarien teilweise zu Äquivalenzklassen kombiniert. Basierend auf den ermittelten Szenarien wurde dann eine Regel-Spezifikation geschrieben, die sich an den Richtlinien der konventionellen Flugsicherung orientiert und eine komplette Auflösung aller Kollisionssituationen ermöglicht. Als letzter Schritt des Konzepts wurde ein Ausweichverfahren geschaffen. Dieses Verfahren beschreibt, wie die Luftfahrzeuge die in den Regeln beschriebenen Ausweichmanöver in den aktuellen Flugpfad so einbinden können, dass eine ausreichende Separation sichergestellt ist.

Zur Umsetzung des Flugsicherungskonzepts wurde im folgenden Kapitel ein Softwaredesign entworfen. Als Grundkonzept wurde dafür ein Framework anstelle einer Klassenbibliothek verwendet. Hierbei wurde darauf geachtet, dass eine möglichst hohe Wiederverwendbarkeit erreicht wird, so dass Teile oder das komplette Framework in zukünftigen Anwendungen wiederverwende werden können. Dazu wurden alle Kernkomponenten, wie Hardwareanbindung, Wegfindungsalgorithmus und Luftraummodell abstrahiert und damit austauschbar gemacht. Für die Datenhaltung wurden außerdem Containerklassen verwendet, so dass die eigentlich gespeicherten Daten nicht bekannt sein müssen. Da ein Framework im Gegensatz zu einer Klassenbibliothek auch einen Logikfluss vorgibt, wurden die klassische, hardware-

basierte Subsumption-Architektur zur Robotersteuerung in eine moderne, objektorientierte Controller-Architektur weiterentwickelt und mehrere, hierarchisch aufeinander aufbauende Flugcontroller implementiert. Die Logik der Flugsicherung wurde hierbei in den in der Hierarchie am höchsten angeordneten Controller integriert. Des Weiteren benötigen autonome unbemannte Luftfahrzeuge einen Wegfindungsalgorithmus, um Flugpfade bestimmen zu können. Da das Luftraummodell als Graph interpretiert werden kann, wurden hierzu verschiedene Graphen-basierte Algorithmen verglichen und mit D*-Lite ein Hybridalgorithmus, der die Vorteile von inkrementellen und heuristischen Wegfindungsalgorithmen verbindet, ausgewählt und implementiert.

Da aus Sicherheits- und Kostengründen eine Erprobung in einer Simulation vorgesehen war, widmete sich das nächste Kapitel der Suche, Evaluation und Auswahl eines geeigneten Simulators zur Evaluation des Flugsicherungskonzepts. Dazu wurden inital Anforderungen an den Simulator aufgestellt, dann wurden Kandidaten gesucht und gegen die Anforderungen geprüft. Die beiden Simulatoren, die für die Aufgabe geeignet erschienen, wurden daraufhin einem Performance-Test unterzogen, um zu bestimmen, welcher den Anforderungen am genauesten entspricht. Letztlich ausgewählt wurde der MORSE-Simulator, der im Performance-Test überzeugte und außerdem einige weitere Vorteile mit sich bringt, wie beispielsweise einen Flugregler für Quadrocopter.

Um das Flugsicherungskonzept bewerten zu können, wurde im sechsten Kapitel eine umfangreiche Evaluation durchgeführt. Dazu wurde die Möglichkeit geschaffen, den unbemannten Luftfahrzeugen über eine Kommunikationsschnittstelle Flugaufträge zu übermitteln, sowie umfangreiche Datenerfassungsmöglichkeiten implementiert. Für die Evaluation wurden zwei Strategien verfolgt. Zum einen wurden systematische Tests der vorher ermittelten Szenarien durchgeführt, um die vollständige Abdeckung der Szenarien durch die Regeln sowie die ausreichende Separation durch die gewählten Ausweichmanöver zu testen. Zum anderen wurde eine reale Anwendung des Konzepts in einem Logistik-Szenario in einer Produktionshalle simuliert. Fokus des Tests war hierbei, dass die Separation der Luftfahrzeuge auch über einen langen Zeitraum nie verletzt wird. Für die systematischen Tests wurden Flugaufträge, die die Szenarien herbei führen, als Script geschrieben und über die Kommunikationsschnittstelle an die Luftfahrzeuge übertragen. Für den Anwendungstest wurde eine Software geschrieben, die Flugaufträge automatisch generiert und an freie Luftfahrzeuge vergibt. Die erhobenen Daten wurden mithilfe von Python-Scripten visualisiert, zum einen wurden die Flugpfade in einem Diagramm dargestellt, zum anderen wurde die minimale Entfernung zwischen den unbemannten Luftfahrzeugen zu jedem Zeitpunkt errechnet und in einem weiteren Diagramm

aufgezeichnet. Die dabei festgestellten Probleme wurden detailliert analysiert und die Ursachen bestimmt.

7.2 Ergebnis

Es wurde ein umfangreiches, theoretisches Konzept zur Flugsicherung von autonomen, unbemannten Luftfahrtzeugen entwickelt. Dieses Konzept wurde im Rahmen eines Frameworks umgesetzt und in einer Simulation erprobt.

Als äußerst vorteilhaft erwies sich hierbei die Teststrategie mit dem zweistufigen Test des Flugsicherungssystems. Da die Infrastruktur für die Testfälle schon während der Endphase der Entwicklung zur Verfügung stand, konnte diese mit ersten Testfällen unterstützt und überprüft werden. Durch dieses Vorgehen fand sich in der final getesteten Version lediglich ein schwerer Fehler. Der Erfolg der Teststrategie zeigte sich auch darin, dass die meisten Probleme bei dem nachfolgenden Anwendungstest auf den Flugregler und Anpassungsprobleme an das neue Luftraummodell zurückgingen anstatt auf Fehler im Flugsicherungsalgorithmus. Auch wurde durch die systematischen Tests sichergestellt, dass alle denkbaren Kollisionsszenarien vom Algorithmus abgedeckt werden. Ein reiner Anwendungstest hätte hierfür nicht ausgereicht, da bei diesem Test durch die Beschaffenheit der Flugaufträge und des Luftraummodell nicht alle Kollisionsszenarien sicher auftreten. Genauso zeigte sich, dass ein rein systematischer Test nicht ausreichend ist, da sich während des Anwendungstests noch ein gravierender Implementierungsfehler im Flugsicherungsalgorithmus zeigte, der durch keinen systematischen Testfall abgedeckt wurde. Aufgrund dieser Resultate kann die kombinierte Teststrategie als sehr erfolgreich gewertet werden.

Die Ergebnisse der Tests waren fast durchweg positiv. Während der systematischen Tests wurde die minimale Separationsentfernung nicht einmal verletzt, die aufgetretenen Probleme waren alle unkritisch. Der Anwendungstest zeigte zwar einige Verletzungen der minimalen Separationsentfernung, die allerdings alle auf einen nicht idealen Flugregler zurück gehen und nicht dem eigentlichen Flugsicherungssystem geschuldet sind. Das zwei mittelschwere Implementierungsfehler bei der Flugroutenverfolgung erst bei den Vorbereitungen des Anwendungstests auffielen, ist hauptsächlich auf die engeren Toleranzen in einem System mit kleineren Quadern zurückzuführen und nicht auf Fehler in den systematischen Tests. Dort fielen derartige geringe Abweichungen von der Flugbahn aufgrund der Ungenauigkeit der Betrachtung nicht auf. Dass ein Implementierungsfehler im Flugsicherungsalgorithmus erst

bei den Vorbereitungen des Anwendungstests aufgetreten ist, muss als einzig negatives Ergebnis gewertet werden. Solche Fehler sollten bereits im Vorhinein durch systematische Tests abgefangen werden.

Als einzig negatives Ergebnis muss das Auftreten eines Implementierungsfehlers im Flugsicherungsalgorithmus erst in den Vorbereitungen des Anwednungstests gewertet werden, da solche Fehler durch systematische Tests abgefangen werden sollen.

Das Flugsicherungskonzept erwies sich von vornherein als gut durchdacht. Parallel zur Implementierung mussten kaum noch Änderungen am eigentlichen Konzept gemacht werden, es kamen lediglich weitere Regeln hinzu. Als aufwendiger und komplizierter erwies sich die Implementierung des beschriebenen Algorithmus, da hierbei sichergestellt werden musste, dass alle möglichen Situationen anhand des Konzeptes sicher erkannt und behandelt werden sowie die nötigen Ausweichmanöver korrekt funktionieren. Die Testergebnisse spiegeln diese Beobachtung wider. Am Flugsicherungskonzept gab es keinerlei Fehler, lediglich Optimierungspotential bei zwei Regeln. Eine Verletzung der Separation durch Schwächen im Flugsicherungskonzept trat nie auf. Bei der Implementierung des Algorithmus hingegen gab es einen leichten und einen schweren Fehler. Während der leichte Fehler lediglich für undefiniertes aber abgesichertes Verhalten sorgt, führt der schwere Fehler zu einer Verletzung der Separation. Dieser schwere Implementierungsfehler basiert auf einem Sonderfall in der Implementierung, der nicht aus den Flugsicherungsregeln abgeleitet ist. Daher existierte für diesen Fall kein systematischer Testfall, sodass der Fehler erst in den Vorarbeiten zu dem Anwendungstest auffiel. Nach dem Beheben dieses Fehlers absolvierte das Flugsicherungssystem den finalen Anwendungstest erfolgreich. Über 26 Stunden kam es bei viel Luftverkehr zu keinerlei Kollisionen im Luftraum. Zwar wurde die minimale Separationsentfernung 16 mal unterschritten, allerdings lag dies an Fehlern durch die Verwendung des nicht idealen Flugreglers der Simulation, der außerhalb des Umfangs dieser Arbeit liegt. Keine der Unterschreitungen ist auf ein Fehlverhalten des Flugsicherungskonzeptes oder der Implementierung des Algorithmus zurückzuführen. Das Flugsicherungskonzept ist also sicher und zuverlässig, ebenso die Implementierung desselben.

Durch die Arbeit wird gezeigt, dass ein autonomer, kooperativer Flugsicherungsmechanismus unter Verwendung von dezentraler UAV2UAV-Kommunikation für unbemannte, autonome Luftfahrzeuge möglich ist. Unter Verwendung dieses Flugsicherungskonzepts ist es möglich, eine dauerhafte Separation zwischen autonomen, unbemannten Luftfahrzeugen sicherzustellen. Allerdings ist dieser Flugsicherungsmechanismus ohne weitere Entwicklung nur begrenzt für reale Anwendungsszenarien verwendbar. Dies ist darin begründet, dass die meisten Anwen-

dungsszenarien eine Separation auch von bemannten Luftfahrzeugen erfordern. Für Anwendungen mit rein unbemanntem, autonomen Verkehr, wie beispielsweise business-to-business Szenarien, ist das Flugsicherungskonzept allerdings schon einsetzbar. Auch ist die Verwendung der Arbeit als Grundlage für die weitere Erforschung einer Integration des unbemannten Luftverkehrs mit bemannten Luftfahrzeugen möglich. Durch die Implementierung im Kontext eines Frameworks steht der Flugsicherungsmechanismus, inklusive Simulationsumgebung, auch für weitere, zukünftige Arbeiten zur Verfügung.

7.3 Ausblick

Um die weitere Verwendung des Flugsicherungsalgorithmus, des Frameworks sowie der Simulation zu erleichtern und ein breiteres Anwendungsfeld zu schaffen, sind einige Verbesserungen und Weiterentwicklungen denkbar.

So ist eine Erweiterung des Algorithmus für Ausweichmanöver denkbar. Dieser könnte so erweitert werden, dass mehr als ein Ausweichmanöver vorgesehen wird. So könnte beispielsweise auch ein vom Kollisionsquader weiter entfernter Quader als Zwischenziel verwendet werden, wenn das Manöver über den direkten Nachbarquader keinen Erfolg brachte. Dadurch würde erreicht, dass die unbemannten Luftfahrzeuge ihre Flüge nicht so oft unterbrechen müssten, wenn das Standardmanöver keinen Erfolg hat. Allerdings sollte dabei betrachtet werden, ob ein längeres Ausweichmanöver tatsächlich effizienter ist als ein kurzes Abwarten.

Eine sinnvolle, zukünftige Verbesserung stellt das Implementieren eines eigenen physikalischen Flugreglers in der Simulationsumgebung des Framework dar. Auf diese Weise wird mehr Flexibilität bei der Parametrisierung und Anpassung an andere Luftfahrzeuge gewährleistet. Damit kann einerseits das Flugverhalten der simulierten Luftfahrzeuge verbessert werden, welches in den Tests für die Verletzungen der Separation verantwortlich war. Andererseits bieten sich erweiterte Möglichkeiten zur Nutzung im Rahmen von Anwendungen. Beispielsweise lassen sich mit einem eigenen Flugregler verschiedene Luftfahrzeuge einfacher simulieren. Auch eine dynamische Anpassung des Flugreglers, beispielsweise an den Beladungszustand des Luftfahrzeugs, ist so denkbar.

Da die Einsatzmöglichkeiten des Flugsicherungskonzepts durch den eingeschränkten Fokus dieser Arbeit bisher auf wenige Anwendungsszenarien begrenzt sind, sollte die weitere Forschung auf eine Integration des Flugsicherungskonzepts mit der bemannten Luftfahrt abzielen.

Ein Ansatz hierbei könnte die Verwendung eines ADS-B-Empfängers auf den unbemannten Luftfahrzeugen sein. Die Luftfahrzeuge könnten über ADS-B empfangene Daten anderer Luftfahrzeuge interpretieren und im eigenen Luftraummodell abbilden. Auf diese Weise wären sie in der Lage, auch bemannten Luftfahrzeugen mit ADS-B-Equipment auszuweichen. Deutlich aufwendiger wäre eine Implementierung eines ADS-B-Senders auf den Luftfahrzeugen. Allerdings böte dieser den Vorteil, dass auch bemannte Luftfahrzeuge von der Anwesenheit der unbemannten Luftfahrzeuge im Luftraum wüssten.

Um die Entwicklung weiterer Anwendungen mit heterogeneren Luftfahrzeugen zu ermöglichen, sollte die enge Kopplung von der Quadergröße im Luftraummodell und der Luftfahrzeuggröße gelöst werden. Dafür müssen die Kommunikation, der Kollisionserkennungs- und der Kollisionsvermeidungsalgorithmus angepasst werden. Ein Flugpfad könnte dann nicht mehr als Sequenz von Quadern beschrieben werden, sondern als Sequenz von Tupeln von Quadern. So lassen sich Flugpfade von kleinen Luftfahrzeugen analog wie bisher als Sequenz von 1-Tupel abbilden, größere Luftfahrzeuge hingegen könnten Sequenzen von n-Tupeln verwenden. Da die Algorithmen zur Kollisionserkennung und -vermeidung auf Basis des eigenen Flugpfades arbeiten, muss lediglich die Analyse des eigenen Pfades angepasst werden. Von den Flugpfaden anderer Luftfahrzeuge müssen, wie bisher, lediglich die daraus resultierenden Blockierungen im Luftraummodell und deren zeitlicher Verlauf gespeichert werden.

Ein weiterer Anwendungsfall von unbemannten Systemen ist das Erkunden von unbekannten Gebieten. Da das Flugsicherungskonzept aktuell ein statisches Luftraummodell verwendet, ist es dafür so nicht einsetzbar. Allerdings ist die Implementierung des Luftraummodells so gewählt, dass dynamische Anpassungen möglich sind. Daher wäre eine weitere zukünftige Arbeit denkbar, die eine Anbindung eines Sensorsystems an das Luftraummodell untersucht und implementiert. Denkbar wären hier beispielsweise Kameras, Laser- oder Radarsysteme. Das würde es den unbemannten Luftfahrzeugen ermöglichen, ihren Luftraum eigenständig zu explorieren. Weiterhin wäre auch der Einsatz des Sensorsystems zur weiteren Absicherung des Fluges möglich, so dass auch beweglichen Hindernissen ausgewichen wird, die nicht an der UAV2UAV-Kommunikation teilnehmen oder deren Kommunikationseinrichtungen beschädigt sind.

Literaturverzeichnis

[1] Becker, M. Simulation Environment for UAV Applications. Tech. rep., Hochschule für Angewandte Wissenschaften, Hamburg, Deutschland, 2016.

[2] Becker, M. Autonomous UAV Controller. Tech. rep., Hochschule für Angewandte Wissenschaften, Hamburg, Deutschland, 2017.

[3] Brooks, R. A. A Robust Layered Control System For a Mobile Robot. Tech. rep., Massachusetts Institute of Technology, Cambridge, MA, USA, 1985.

[4] Bundesministerium für Verkehr und digitale Infrastruktur. Die neue Drohnen-Verordnung: Ein Überblick über die wichtigsten Regeln. Tech. rep., Bundesministerium für Verkehr und digitale Infrastruktur, 2017.

[5] Bundesministerium für Verkehr und digitale Infrastruktur. *Verordnung zur Regelung des Betriebs von unbemannten Fluggeräten.* Bundesanzeiger Verlag, Bonn, 2017, pp. 683–688.

[6] Coppelia Robotics GmbH. V-REP. Website, 2016. http://www.coppeliarobotics.com/, accessed April 3, 2016.

[7] Cyberbotics S.a.r.l. Webots. Website, 2016. https://www.cyberbotics.com/, accessed April 3, 2016.

[8] Deering, S. Host Extensions for IP Multicasting. RFC 1112, IETF, August 1989.

[9] DIN 13312:2005-02. Navigation - Begriffe, Abkürzungen, Formelzeichen, graphische Symbole. Norm, Beuth Verlag, 2005.

[10] Dresner, K., and Stone, P. Multiagent Traffic Management: A Reservation-Based Intersection Control Mechanism. In *The Third International Joint Conference on Autonomous*

Agents and Multiagent Systems (2004), pp. 530–537.

[11] Echeverria, G., Lemaignan, S., Degroote, A., Lacroix, S., Karg, M., Koch, P., Lesire, C., and Stinckwich, S. Simulating Complex Robotic Scenarios with MORSE. In *Proceedings of the Third International Conference on Simulation, Modeling, and Programming for Autonomous Robots* (Berlin, Heidelberg, 2012), SIMPAR'12, Springer-Verlag, pp. 197–208.

[12] Forni, A. A., and van der Meulen, R. Gartner Says Almost 3 Million Personal and Commercial Drones Will Be Shipped in 2017. Press release, February 2017. http://www.gartner.com/newsroom/id/3602317, accessed October 10, 2017.

[13] Glas, B., Sander, O., Müller-Glaser, K. D., and Becker, J. Echtzeitfähige Car-to-X–Kommunikationsabsicherung und E/E-Architekturintegration. In *Vernetztes Automobil*, W. Siebenpfeiffer, Ed., 1 ed. Springer Vieweg, Wiesbaden, 2014, ch. 2, pp. 70–81. ISBN: 978-3-658-04018-5.

[14] Hughes, J. F. *Computer graphics : principles and practice*, 3 ed. Addison-Wesley, NJ, 2014. ISBN: 978-0-321-39952-6.

[15] Kleuker, S. *Qualitätssicherung durch Softwaretests*, 1 ed. Springer Vieweg, Wiesbaden, 2013. ISBN: 978-3-8348-2068-6.

[16] Koenig, S., and Likhachev, M. D* Lite. In *Eighteenth National Conference on Artificial Intelligence* (Menlo Park, CA, USA, 2002), American Association for Artificial Intelligence, pp. 476–483.

[17] Koenig, S., Likhachev, M., and Furcy, D. Lifelong Planning A*. *Artificial Intelligence 155*, 1-2 (May 2004), 93–146.

[18] Kopardekar, P. Unmanned Aerial Systems Traffic Management (UTM) - Safely Enabling UAS Operations in low-altitude Airspace. In *Proceedings of the 4^{th} UTM Convention July 28–30, 2015, Moffett Field, California, USA* (2015), National Aeronautics and Space Administration.

[19] Maschek, U. *Sicherung des Schienenverkehrs*, 1 ed. Vieweg+Teubner Verlag, 2012. ISBN: 978-3-8348-2070-9.

[20] Mensen, H. *Moderne Flugsicherung*, 4 ed. Springer Vieweg, Wiesbaden, 2014. ISBN: 978-3-642-54294-7.

[21] Meyer, J., Sendobry, A., Kohlbrecher, S., Klingauf, U., and von Stryk, O. *Comprehensive Simulation of Quadrotor UAVs Using ROS and Gazebo.* Springer Berlin Heidelberg, Berlin, Heidelberg, 2012, pp. 400–411.

[22] MORSE Community. MORSE. Website, 2016. https://www.openrobots.org/morse/, accessed April 3, 2016.

[23] National Aeronautics and Space Administration. UTM: Air Traffic Management for Low-Altitude Drones. *NASA Facts* (2015).

[24] Olivares-Mendez, M. A., Kannan, S., and Voos, H. Setting up a testbed for UAV vision based control using V-REP and ROS: A case study on aerial visual inspection. In *Unmanned Aircraft Systems (ICUAS), 2014 International Conference on* (May 2014), pp. 447–458.

[25] Open Source Robotics Foundation. Gazebo. Website, 2016. http://gazebosim.org/, accessed April 3, 2016.

[26] Rasche, C. *A cooperative and verifiable UAV behavior for 3D environments.* PhD thesis, Universität Paderborn, 2013.

[27] Reif, K. *Automobilelektronik*, 5 ed. Springer Vieweg, Wiesbaden, 2014. ISBN: 978-3-658-05048-1.

[28] Riehle, D. *Framework Design - A Role Modeling Approach.* PhD thesis, ETH Zürich, 2000.

[29] Schmalenbach, C. *Performancemanagement für serviceorientierte JAVA-Anwendungen*, 1 ed. Springer Verlag, Berlin Heidelberg, 2007. ISBN: 978-3-540-36632-4.

[30] SESAR Joint Undertaking. European Drones Outlook Study. Tech. rep., SESAR Joint Undertaking, 2016.

[31] SESAR Joint Undertaking. U-Space Blueprint. Tech. rep., SESAR Joint Undertaking, 2017.

[32] skyguide. skyguide und Projektpartner geben erste Drohnen Live-Demonstration in Europa zu U-Space-Kapazitäten bekannt. Press release, September 2017.

[33] Stentz, A. Optimal and efficient path planning for unknown and dynamic environments. *International Journal of Robotics and Automation 10* (1993), 89–100.

[34] Stentz, A. The Focussed D* Algorithm for Real-Time Replanning. In *In Proceedings of the International Joint Conference on Artificial Intelligence* (1995), pp. 1652–1659.

[35] Tanenbaum, A. S. *Modern Operating Systems*, 3 ed. Prentice Hall Press, 2007. ISBN: 978-0-1360-0663-3.

[36] Tooley, M., and Wyatt, D. *Aircraft Communications and Navigation Systems: Principles, Operation and Maintenance*, 1 ed. Elsevier/Butterworth-Heinemann, 2007. ISBN: 978-0-7506-8137-7.

[37] Vivaldini, K. C. T., Guizilini, V., Oliveira, M. D. C., Martinelli, T. H., Wolf, D. F., and Ramos, F. Route planning for active classification with UAVs. In *2016 IEEE International Conference on Robotics and Automation (ICRA)* (May 2016), pp. 2563–2568.

[38] Zimmermann, W., and Schmidgall, R. *Bussysteme in der Fahrzeugtechnik*, 5 ed. Springer Vieweg, Wiesbaden, 2014. ISBN: 978-3-658-02418-5.

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbständig verfasst und nur die angegebenen Hilfsmittel benutzt habe.

Hamburg, 4. Mai 2018 Matthies Becker