Westfälische Hochschule

University of Applied Sciences
Gelsenkirchen Bocholt Recklinghausen

Bachelorarbeit

Titel der Arbeit (Deutsch) // Title of Thesis (German)

Integration von kommerziellen Flugdrohnen in eine Leitstandsinfrastruktur für Rettungseinsätze

Titel der Arbeit (Englisch) // Title of Thesis (English)

Integration of commercial UAVs in a Search and Rescue control infrastructure

Akademischer Abschlussgrad: Grad, Fachrichtung (Abkürzung) // Degree

Bachelor of Science (B.Sc.)

Autorenname, Geburtsort // Name, Place of Birth

Alexander Schmitz, Wesel

Studiengang // Course of Study

Technische Informatik

Fachbereich // Department

Informatik und Kommunikation

Erstprüfer // First Examiner

Prof. Dr. Hartmut Surmann

Zweitprüfer // Second Examiner

Dipl.-Inf. Rainer Worst

Abgabedatum // Date of Submission

14.05.2018

Eidesstattliche Versicherung

Schmitz, Alexander

Name, Vorname // Name, First Name

Ich versichere hiermit an Eides statt, dass ich die vorliegende Abschlussarbeit mit dem Titel *„Integration von kommerziellen Flugdrohnen in eine Leitstandsinfrastruktur für Rettungseinsätze“* selbstständig und ohne unzulässige fremde Hilfe erbracht habe. Ich habe keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie wörtliche und sinngemäße Zitate kenntlich gemacht. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Gelsenkirchen, den 14.05.2018

Ort, Datum, Unterschrift // Place, Date, Signature

Danksagung

An dieser Stelle möchte ich einigen Menschen meinen Dank aussprechen. Mein Betreuer, Herr Prof. Dr. Hartmut Surmann, der gleichzeitig mein Robotik- und Bildverarbeitungs-Professor an der Hochschule war, hat mir über die letzten Jahre seine Themen auf eine sehr interessante Weise gelehrt. Dabei gab er meinen Kommilitonen und mir mit interessanten Projekten viel Raum und Möglichkeiten, um das Gelernte praktisch umzusetzen. Er ermutigte uns stets, Neues auszuprobieren. Dadurch konnte ich meine Fähigkeiten stark ausbauen und habe viel dazu gelernt, wofür ich ihm hier ausdrücklich danken möchte.

Ebenso möchte ich meinem Zweitprüfer und lokalem Projektleiter des TRADR Projektes Dipl.-Inf. Rainer Worst für die gute und angenehme Zusammenarbeit, Abstimmung und Unterstützung meiner Tätigkeiten beim Fraunhofer Institut IAIS danken.

Die Arbeit im Forschungsprojekt TRADR beim Fraunhofer Institut IAIS habe ich als sehr angenehm empfunden und durfte dort wichtige Erfahrungen sammeln. Bei der Arbeit für Fraunhofer traf ich auf viele interessante Menschen aus den unterschiedlichsten Bereichen, wodurch ich in Gesprächen mein Wissen deutlich erweitern konnte. Daher möchte ich mich auch bei allen Kollegen, mit denen ich dort zusammengearbeitet habe, für die netten, konstruktiven Gespräche, Hilfestellungen und die gute Betreuung bedanken.

Ein besonderer Dank geht dabei an die Kollegen, die mit ihren Arbeiten direkt und indirekt angrenzende Komponente zu meiner Arbeit lieferten und mir bei Problemen halfen. Ich danke Gerhard Senkowski, Artur Leinweber, Tim Buschmann und Daniel Reuter für die gute Zusammenarbeit bei der Umsetzung des gesamten UAV-Workflows.

Zu guter Letzt möchte ich auch meiner Familie danken, die seit Anbeginn immer für mich da ist und mir in einigen Aspekten den Rücken frei hält, um mich so stark bei meinem langen Studium der Computer Wissenschaften zu unterstützen.

Kurzfassung

Die Bachelorarbeit „Integration von kommerziellen Flugdrohnen in eine Leitstandsinfrastruktur für Rettungseinsätze“ beschreibt eine exemplarische Lösung für die Integration der UAV-Modelle Phantom 3/4 und Mavic Pro des Herstellers DJI in die Leitstandsinfrastruktur des Rettungsrobotik Projektes TRADR. Die Integration wird mittels einer auf Basis des DJI Mobile SDKs neu entwickelten Smartphone App realisiert, welche die UAVs überwacht und steuert. Die App empfängt das Livebild und die Telemetriedaten vom UAV und kann diesem Aufgaben übermitteln, die die Drohne autonom ausführt. Der Aufbau und die Funktionsweise der App werden in dieser Arbeit im Detail erklärt. Zudem beschreibt sie ein neu entwickeltes Anwendungsprotokoll, das eine UAV Schnittstelle für das TRADR System spezifiziert. Über diese Schnittstelle ist die Smartphone App des Piloten mit dem TRADR System einschließlich der Leitstandskomponenten verbunden. Im Leitstand existieren verschiedene Überwachungs-, Anzeige- und Planungskomponenten, die untereinander für die Kommunikation das Robotic Operating System (ROS) einsetzen. Auf dem TRADR System arbeitet eine neu entwickelte Softwarekomponente, die zum einen die Schnittstelle für die sich im Einsatzgebiet befindenden UAVs bereitstellt und zum anderen die am System angemeldeten UAVs über ROS-Topics und -Services für die Überwachungs- und Anzeigekomponenten im TRADR Leitstand nutzbar macht. Es wird dabei genau beschrieben, in welcher Form der Datenaustausch zwischen den Instanzen UAV und Smartrphone App beim Piloten im Einsatzgebiet und den sich im Leitstand befindenden Planungs-, Anzeige- und Überwachungskomponenten, die von einem UAV-Operator bedient werden, abläuft. Dabei wird die technische Umsetzung der Übertragung von Status- und Telemetriedaten des UAVs sowie des hoch aufgelösten Livebildes der UAV-Kamera mit einer Latenz von unter einer Sekunde im Dateil erläutert.

Diese Bachelorarbeit entstand im Rahmen des Europäischen Forschungsprojektes TRADR und wurde durch die Europäische Kommission finanziell gefördert. Das TRADR Projekt ist finanziert durch EU-FP7-ICT, zu finden unter der Nummer 609763.

Schlüsselwörter: TRADR, Rettungseinsatz, Rettungsrobotik, Luftroboter, Drohne, UAV, Integration, Leitstand, Kommunikationsschnittstelle, Smartphone App, DJI, DJI Phantom, DJI Mavic Pro, DJI Mobile SDK, ROS

Abstract

The bachelorthesis „Integration of commercial UAVs in a Search and Rescue control infrastructure“ describes an exemplary solution for the integration of the UAV models Phantom 3/4 and Mavic Pro of the manufacturer DJI into the control center infrastructure of the Search and Rescue (SaR) robotics project TRADR. The integration is realized by a newly developed smartphone app based on the DJI Mobile SDK, which monitors and controls the UAVs. The app receives the live image and telemetry data from the UAV and can transmit user specified tasks to the drone for autonomous execution. The architecture and procedures of the app are explained in detail in this document. In addition, this thesis describes a newly developed application protocol that specifies a UAV interface for the TRADR system. This interface connects the pilot's smartphone app to the TRADR system, including the control room components. The control center has various monitoring, display and planning components that use the Robotic Operating System (ROS) for communication. The TRADR system uses a newly developed software component, which provides the interface for the UAVs in the field, as well as ROS topics and services for the components in the TRADR control center, so that they can communicate with the registered UAVs. This thesis describes in detail the form of the data exchange between the pilot's instances in field (UAV and Smartphone App) and the display and monitoring components in the control room, which are operated by a UAV operator. The technical implementation of status and telemetry data tramission and the streaming of the high definition live image with a latency of less than one second will be explained in detail in this document.

This bachelor thesis was part of the European research project TRADR and was partially funded by the European Commission. The TRADR project is funded by EU-FP7-ICT, which can be found by its reference number 609763.

Keywords: TRADR, Rescue, Disaster Response, Search and Rescue (SaR), Drone, UAV, Integration, Control Center, Communication Interface, Smartphone App, DJI, DJI Phantom, DJI Mavic Pro, DJI Mobile SDK, ROS

Inhaltsverzeichnis

1 Einleitung

Inhaltsangabe

1.1 Ausgangssituation

1.1.1 Das Forschungsprojekt TRADR

Das Europäische Forschungsprojekt TRADR „Long-Term Human-Robot Teaming for Robot Assisted Disaster Response“ beschäftigt sich mit der Thematik Teambildung zwischen menschlichen Rettungskräften und Rettungsrobotern im Kontext von Rettungseinsätzen in Katastrophengebieten. [35][54][53]

Als Ergebnis der Forschungsarbeiten entstehen im TRADR Projekt viele zu einen Gesamtsystem integrierte Software-Lösungen, die es möglich machen, dass Rettungsroboter von einem Komandoleitstand aus gesteuert und überwacht Rettungskräften im Einsatzgebiet assistieren.

Ein wichtiger Teilaspekt der Forschungsaktivitäten besteht im Entwurf und der Evaluierung von Konzepten zur Mensch-Roboter-Teambildung. Diese entwickelten Konzepte sehen Roboter-Operatoren vor, die in einem Leitstand arbeiten, aus dem heraus sie die Roboter oder Teams aus Menschen und Robotern überwachen und befehligen. [35][42]

Im Katastrophengebiet ist es den Robotern eher möglich, in Gefahrengebieten zu operieren, in die menschliche Rettungskräfte aufgrund der Gefahrenlage nur sehr eingeschränkt oder gar nicht eingesetzt werden können. Die Rettungsroboter dringen in diese Areale vor und liefern mit ihren Sensoren Messdaten. Zu diesen zählen unter anderem verschiedene Typen von Kameras, die Livebilder aus den Gefahrengebieten liefern. Manche davon stellen omnidirektionale Aufnahmen, Wärmebilder oder Bilddaten mit Tiefeninformationen zur Verfügung. Hinzu kommen Laser-Scaner und chemische Sensoren. Die Roboter übermitteln während des Einsatzes kontinuierlich Daten an das TRADR System. Aus den im zentralen Leitstand eingehenden Daten werden mittels intelligenter Software Systeme Informationen gewonnen, die den Rettungskräften bei ihrem Einsatz helfen. Die Roboter werden dabei entweder von einem Operator aus dem Leitstand heraus teleoperiert gesteuert oder verrichten Aufgaben autonom. Zusätzlich zur Informationsgewinnung besteht bei einem Teil der Bodenroboter auch die Möglichkeit der Manipulation mittels eines auf der mobilen Roboter-Plattform angebrachten Roboterarmes.

Abbildung 1.1: Wissenschaftler des TRADR Projektes mit Robotern in Amatrice, Italien, nach einem Erdbeben 2016 (Quelle: [29])

1.1.2 Einsatz von Luftrobotern im TRADR Projekt

Neben der Verwendung von Bodenrobotern bestehen bei Rettungseinsätzen auch Aufgaben, die vorrangig von Luftrobotern (UAVs) umgesetzt werden können. Die Nutzung von UAVs ist daher heute ein wichtiger Bestandteil vieler Einsätze in Katastrophengebieten.

Livebilder, die mit Kameras bestückte UAVs aus der Luft liefern, bieten der Einsatzleitung eine Übersicht der Lage und des Geschehens am Einsatzort. Ebenfalls können solche Livebilder den Roboter-Operatoren im Leitstand bei Fahrten und Manipulationen, die mit den Bodenrobotern durchgeführt werden, helfen.

Zusätzlich zur Nutzung des Livestreams ist ein weiterer sehr wichtiger Teilaspekt des TRADR Projektes, den Rettungskräften frühzeitig visuelle Darstellungen des Katastrophengebietes und der Objekte in diesem Gebiet zu liefern. Erforscht wird ein weites Spektrum an Darstellungsformen, z.B. orthografische Karten (Luftbilder), 3D-Modelle und Panoramen. Um die Darstellungen zu generieren, bestehen im Projekt bereits Verarbeitungsmethoden und Darstellungslösungen oder werden aktuell entwickelt. Diese Verarbeitungssoftware benötigt Bild- oder Video-Daten angereichert mit Metainformationen wie z.B. Telemetrie-Daten des Aufnahmegerätes. Besonders für diese Anwendungsfälle ist die Nutzung von UAVs wichtig.

Zur Erfüllung der genannten Aufgaben stehen im TRADR Projekt verschiedene Typen von UAVs zur Verfügung.

Vom Projektbeginn an werden mehrere UAVs der Forschungsserie AscTec NEO[73] sowie UAVs des Modells AscTec Falcon[72] des Herstellers Ascending Technologies (AscTec)[71] eingesetzt. Der Hersteller AscTec ist gleichzeitig Projektpartner gewesen. Nach der Übernahme von Ascending Technologies durch Intel und dem Ausscheiden des UAV Herstellers aus dem TRADR Projekt kümmert sich die Firma Smart Robotic Systems (SRS)[69] stellvertretend um den Einsatz der AscTec UAVs im Projekt. SRS nutzt ihr selbst entwickeltes SKIDER System[70] als Kommunikationsgegenstelle für den Betrieb der AscTec UAVs.

Ab Mitte des Projektzeitraumes wird parallel zur Nutzung der AscTec UAVs verstärkt der Einsatz von Kompakt-UAVs des Herstellers DJI[10] erprobt. Im TRADR Projekt werden dazu drei UAVs der Modelle DJI Phantom 3 Professional[21], DJI Phnatom 4 Pro[22] und DJI Mavic Pro[15] verwendet.

Problematisch ist, dass die UAVs nicht in die vorhandenen Konzepte und technische Infrastruktur integriert sind, die jedoch ein Hauptforschungsaspekt des Projektes darstellen. Weder für die DJI Kompakt-UAVs sowie für die AscTec UAVs und das SRS SKIDER System besteht eine Anbindung an die vorhandene Leitstandsinfrastruktur des TRADR Systems. Zudem fehlt das autonome Agieren der UAVs. Die Fluggeräte werden alleine durch die Piloten ferngesteuert, die auch die Aufnahmen von Fotos und Videosequenzen manuell vornehmen.

Verschiedene Arten von Luftrobotern

Das Spektrum an verfügbaren UAVs ist breit. Bisher werden im Kontext von Rettungseinsätzen häufig große UAVs mit speziellen Sensoraufbauten eingesetzt. Zu nennen sind hier z.B. die Matrice Serien von DJI[11], UAVs des Herstellers MikroKopter[48] oder Intels Tochterunternehmen Ascending Technologies (AscTec)[71]. Diese UAVs können mit einem diagonalen Achsenabstand von über einem Meter als größere Drohnen bezeichnet werden. In der Regel sind sie mit sechs oder acht Propellern ausgestattet und können größere Nutzlasten tragen. Die dementsprechend größer dimensionierte Akkuausstattung führt zu höheren Gewichten. Eine DJI Matrice 600 besitzt ein Gewicht von über 500 Gramm [14] ohne Sensoraufbauten und zusätzliche Nutzlast. Dies ist im Hinblick auf neuere rechtliche Regulierungen des Drohnenfluges ein immer wichtigerer Faktor. Je höher die Gewichtsklasse, in das ein UAV einzustufen ist, desto restriktivere Vorschriften und Gesetze sind beim Betrieb solcher Fluggeräte zu beachten. Die Vorteile dieser UAVs bestehen in der Möglichkeit sehr flexibel eigene Hardware auf den Fluggeräten zu installieren. Ein breites Spektrum an verschiedenen Sensoren (RGBD/IR-Kameras, LiDAR, GeoRadar, Strahlungsmesssensoren, usw.) kann mit Messrechnern auf den Plattformen der UAVs installiert werden[13]. Die Hersteller bieten für ihre Forschungs-Serien auch Möglichkeiten, eigene Steuerungs- und Kommunikationssoftware in die Systeme zu integrieren[20]. Die Zielgruppe solcher UAVs sind der professionelle, kommerzielle Einsatz sowie die Forschung. Privatanwender gehören nicht dazu, dies ist auch an den hohen
Anschaffungspreisen von 5000 € bis 50000 € je nach UAV und Sensor-Ausstattung erkennbar.

In Rettungseinsätzen kommen jedoch neuerdings auch kleinere und preiswertere Varianten von UAVs zum Einsatz, die entscheidende Vorteile bieten. Bei diesen handelt es sich nicht um Bausätze oder Forschungsserien, sondern um marktreife Produkte, die in sehr hoher Stückzahl mit standardisierter Ausstattung angeboten werden. Zielgruppe sind dabei neben kommerziellen Einsätzen vor allem Privatanwender. Der Nachteil dieser UAVs ist, dass keine Möglichkeit vorgesehen ist, individuell benötigte Sensoren und Auswertungsrechner an den UAVs zu befestigen. Zudem ist es nicht möglich, eigene Softwareverfahren zur Auswertung oder Steuerung auf der UAV Hardware zu installieren. Es handelt sich um weitgehend proprietäre geschlossene Systeme, bei denen keine Modifikation vorgesehen ist. In der Regel sind solche UAVs für einen bestimmten Anwendungsfall konzipiert und nur für diesen gut zu verwenden. Im Falle der Kompakt-UAV Serien von DJI (Phantom, Mavic, Sparc) handelt es sich um die Aufnahme von hochauflösenden Bildern und Videos. Die Kameras sind bei allen Modellen über computergesteuerte Kardanische Aufhängungen (Gimbal)[87] mit dem Rumpf der UAVs verbunden. Die Steuerungen der Gimbals halten die Kameras stets im eingestellten Winkel und gleichen Lageänderungen der UAVs beim Flug aus. Die Vorteile solcher Kompakt-UAVs liegen darin, dass es sich um marktreif entwickelte Produkte handelt. Ihre Hard- und Software ist im Vergleich zu UAV-Bausätzen und Eigenkonstruktionen sehr ausgereift und gut getestet, da die UAVs in einer großen Stückzahl für den Massenmarkt produziert werden. Ein weiterer Vorteil ergibt sich aus dem geringen Gewicht und der kompakten Bauform. Aufgrund der deutlich geringeren Abmessungen ist es möglich, auch schwierigere Indoor-Einsätze zu fliegen. Zudem rechtfertigen

geringere Anschaffungskosten manchmal das Eingehen von risikoreicheren Missionen, da im Falle eines Absturzes der finanzielle Verlust nicht zu groß ist. Zusätzlich bieten solche UAVs eine im Vergleich sehr einfache Handhabung und Bedienung. Durch die kompakte Technik sind sie zudem sehr einfach zu transportieren und schnell einsatzbereit. Gerade letztere Punkte sind starke Argumente für den Einsatz solcher kompakten UAVs in Rettungseinsätzen.

In Abgrenzung zu den erst genannten großen UAVs (z.B. DJI Matrice) werden diese kleineren UAVs (z.B. DJI Phantom, DJI Mavic) in dieser Arbeit als Kompakt-UAVs bezeichnet.

Abbildung 1.2: DJI Mavic Pro mit Fernsteuerung

Abbildung 1.3: DJI Phantom 3

Verwendung eines Kompakt-UAVs im realen Einsatz

Im TRADR Projekt ist im Jahr 2016 bereits in diesem Kontext Erfahrung gesammelt worden. Nach einem schweren Erdbeben in Amatrice, Italien, durfte das TRADR Projekt ihre bis dahin erforschten Roboter-Lösungen in einem realen Katastrophengebiet evaluieren. Die Aufgabe bestand unter anderem darin, mittels Boden- und Luftroboter Innen- und Außenaufnahmen sowie LiDAR-Scans von schwer beschädigten Gebäuden zu liefern, aus denen z.B. 3D-Modelle zur Hilfe des Wiederaufbaus erzeugt wurden. Die Gebiete und Gebäude waren Gefahrenzonen, die nur stark eingeschränkt und zum großen Teil gar nicht von Menschen betreten werden durften. TRADR setzte als Luftroboter zwei UAVs vom Typ AscTec Falcon ein und ein Kompakt-UAV vom Typ DJI Phantom 3. Alle UAVs wurden teleoperiert von Piloten gesteuert. Während es bei den großen und deutlich teureren UAVs vom Typ AscTec Falcon zu risikoreich erschien, beispielsweise durch ein Fenster oder durch eine offene Stelle im Dachstuhl in die zerstörten Gebäude hinein zufliegen, wurde dies mit dem Kompakt-UAV vom Typ DJI Phantom 3 erfolgreich durchgeführt. Die Phantom 3 wurde ohne Sichtkontakt teleoperiert vom Piloten innerhalb des Gebäudes geflogen und machte wichtige Aufnahmen der Innenräume. Dabei wurde der Flug zum Teil einzig auf Basis des übertragenen Livebildes der UAV-Kamera durchgeführt. Beim hinein und hinaus fliegen assistierten die zwei Falcons, die mit ihren Kameras das Manöver von Außen aus der Luft beobachteten und somit dem Piloten weitere Blickwinkel boten. Bei dieser Mission waren die geringe Abmessungen der Phantom 3 als auch der geringere finanzielle Schaden bei einem möglichen Absturz mit eine Ursache dafür, dass das Risiko durch enge Stellen in einem zerstörten Gebäude zu fliegen, eingegangen wurde. [41] [68] [66] [64] [65]

Abbildung 1.4: Phantom 3 fliegt in eine zerstörte Kirche; AscTec Falcon assistiert mit Livebild aus der Luft. (Quelle: [29])

1.2 Zielsetzung

Die aus der Ausgangssituation hervorgehende Zielsetzung ist, das für die Bodenroboter vorhandene Konzept von Operatoren im Leitstand, welche die Roboter im Einsatzgebiet befehligen, in einer angepassten Form auch für Luftroboter umzusetzen. Bei den Flugrobotern soll das Team im Einsatzgebiet zum einen aus rechtlichen und zum anderen aus praktischen Gründen aus einem menschlichen Piloten und dem UAV bestehen. Im Leitstand befindet sich der auch zum Team gehörende UAV-Operator, der den Akteuren im Einsatzgebiet die Befehle gibt.

Zudem sollen die UAVs zukünftig nicht nur manuell teleoperiert geflogen werden können, sondern auch autonom vorher definierte Flugbahnen abfliegen und automatisch Fotos und Videos aufnehmen.

Dazu soll der UAV-Operator mit einer Planungssoftware Aufgaben, z.B. das Abfliegen eines bestimmten Gebietes um Luftaufnahmen für eine Kartierung zu machen, definieren können. Spezifizierte Aufgaben sollen in Folge durch den UAV-Operator an ein Piloten-UAV-Team weitergegeben werden, sodass die Aufgabe vom UAV unter Beaufsichtigung durch den Piloten autonom ausgeführt werden kann. Die gesammelten Daten, z.B. die aufgenommenen Luftbilder, sollen an das TRADR Postprocessing System übertragen werden, um von diesem weiterverarbeitet werden zu können. Zudem sollte dauerhaft das UAV vom Leitstand aus überwacht werden können, in einer Form, dass der Operator sehen kann wo sich das UAV befindet, welche Aufgabe es gerade ausführt und wie der Fortschritt dieser Ausführung ist.

Zusätzlich soll das Livebild der UAV Kamera bei Bedarf zu jedem Zeitpunkt im Leitstand darstellbar sein.

Um dies zu gewährleisten, ist eine Integration der eingesetzten UAVs und des Pilotenequipment in das TRADR System notwendig.

1.3 Problemstellung

Die Kompakt-UAVs des Herstellers DJI besitzen keine Hersteller-Lösung für eine Anbindung an Bodenstationskomponenten wie z.B. eine Leitstandsinfrastruktur. Die Produkte umfassen das UAV und die Fernsteuerung, mit der der Pilot das UAV steuern kann. Autonomie kann dem System dadurch gegeben werden, dass an die Fernsteuerung ein Smartphone oder Tablet mit Apple iOS oder Android Betriebssystem angeschlossen wird. Für beide Betriebssysteme gibt es vom Hersteller DJI Apps[12], um das Livebild des UAVs während des Fluges zu sehen, sowie Flugaufgaben zu planen, die dann vom UAV autonom ausgeführt werden. Neben der Hersteller App gibt es weitere Apps anderer Anbieter mit zum Teil sehr ähnlichem Funktionsumfang. Die meisten Apps bieten keine Möglichkeiten, während des Fluges die Telemetriedaten über das Netzwerk vom Smartphone aus live zu einem weiteren Überwachungs- und Planungssystem zu übertragen. Ausschließlich das Hochladen der aufgenommenen Bilder und der aufgezeichneten Telemetrie-Daten zu Cloud-Diensten der App-Anbieter wird in einigen Fällen ermöglicht. Darüber hinaus bieten wenige Apps, darunter auch die Hersteller App, die Übertragung des Livebildes als RTMP/RTSP-Stream an. Dieser weist jedoch eine Latenz von einigen Sekunden auf. Standardisierte Schnittstellen für die Kommunikationen zwischen Bodenstationen (GCS) und UAVs wie z.B. MAVLink werden von keiner App unterstützt. Zusammenfassend ist festzuhalten, dass keine verfügbare App zur Steuerung von DJI Kompakt-UAVs die Möglichkeit der Anbindung an eine Leitstandsinfrastruktur, wie sie im TRADR Projekt vorgesehen ist, bietet. Einzig der Hersteller UgCS[74] bietet eine kommerzielle, proprietäre Lösung an, bei der DJI Kompakt UAVs über eine App an das Leitstandsystem von UgCS angeschlossen werden[77][75][76]. Die Nutzung einer solchen Lösung ist im Rahmen des TRADR Projektes jedoch nicht sinnvoll, da im Projekt eine Leitstandsinfrastruktur vorhanden ist, deren Komponenten eines der zentralen Forschungsschwerpunkte des Projektes darstellen. Zudem sind die UgCS Lösungen nicht als freie OpenSource Software verfügbar.

1.4 Lösungsansatz

Der Hersteller DJI bietet ein Software SDK für seine Kompakt-UAVs an. Dieses SDK nennt sich „DJI Mobile SDK“ [19][43] und ist deutlich von den anderen SDKs des Herstellers zu unterscheiden. Das Mobile SDK ist das einzige, mit dem sich Steuerungssoftware für die Kompakt-UAVs (Phantom, Mavic, Sparc) erstellen lässt. Zudem können mit dem Mobile SDK nur Apps für die Mobilbetriebssysteme Apple iOS und Android entwickelt werden. Mittels dieses Mobile SDKs ermöglicht es DJI Drittanbietern eigene Smartphone Apps zur Steuerung der Kompakt-UAVs zu entwickeln.

Der Lösungsansatz, dessen Realisierung in dieser Arbeit beschrieben wird, ist die Entwicklung einer eigenen Smartphone App zur Steuerung von DJI Kompakt-UAVs (Phantom, Mavic) speziell für das TRADR Projekt, die sich über eine neu entwickelte Kommunikationsschnittstelle an die TRADR Leitstandsinfrastruktur anbinden lässt. Die Anforderungen an die Kommunikationsschnittstelle und die App ergeben sich aus den Anwendungsfällen des UAV Einsatzes im Projekt. Zu der Entwicklung der App und Kommunikationsschnittstelle wird auf der Gegenseite beim TRADR System eine Softwarekomponente benötigt, welche die UAVs im TRADR Projekt verwaltet und die Kommunikation zwischen den UAVs und Planungs-, Überwachungs- und Anzeigekomponenten des Leitstandes regelt.

1.5 Geplante Umsetzung

Die im TRADR Projekt verwendeten UAVs DJI Mavic Pro und DJI Phantom 3 werden in die vorhandene Leitstandsinfrastruktur integriert.

Dazu wird eine Smartphone App auf Basis des DJI Mobile SDKs entwickelt, die auf dem Smartphone des UAV Piloten, das mit der Fernsteuerung verbunden ist, ausgeführt wird. Die Smartphone App wird die Position des UAVs auf einer Karte darstellen. Darüber hinaus wird dem UAV Pilot das Livebild der UAV-Kamera auf dem Smartphone angezeigt. Mittels der App wird es dem Piloten möglich, sein Erkundungsaufgaben für spezielle Anwendungsfälle zu definieren. Zum einen können Explorationsaufgaben erstellt werden, die es ermöglichen, ein Objekt kreisförmig durch das UAV zu umfliegen, wobei aus verschiedenen Perspektiven Fotos von diesem Objekt aufgenommen werden. Zum anderen können Aufgaben definiert werden, die es möglich machen, ein Gebiet zu erkunden, indem das UAV dieses mäanderförmig abfliegt und Luftaufnahmen tätigt.

Die Hauptanforderung besteht jedoch darin, dass die Smartphone App die empfangenen Telemetriedaten des UAVs an die Komponenten des TRADR Leitstandes weiterleitet. Darüber hinaus wird es möglich sein, Erkundungsaufgaben mit Hilfe der Planungskomponenten des Leistandes zu definieren und diese zu Pilot und UAV zu übertragen, damit diese vom UAV autonom ausgeführt werden können. Während das UAV im Einsatz ist, wird das Livebild der UAV-Kamera, das

vom Smartphone empfangen wird, ebenfalls in den TRADR Leitstand weitergeleitet.

Zur Realisierung der Kommunikation zwischen der Smartphone App und dem TRADR System wird eine neu entwickelte Kommunikationsschnittstelle eingesetzt. Diese wird jedoch so allgemein gehalten und offen spezifiziert, dass es auch der Firma Smart Robotic Systems (SRS) möglich ist, ihre Systeme über dieselbe Schnittstelle anzubinden, um so auch die Integration der AscTec UAVs in das TRADR System umzusetzen. Die Kommunikationsschnittstelle wird ein Anwendungsprotokoll sein, das die Softwarekomponenten (TRADR System, Smartphone App, SRS SKIDER System), die sich auf entfernten Rechnern befinden, über Netzwerkverbindungen (kabelgebunden und drahtlos) kommunizieren lässt.

Auf der Seite des TRADR Leitstandes wird dazu eine Software Komponente entwickelt, die als Gegenstelle agiert, zu der sich die verschiedenen UAV-seitigen Systeme wie beispielsweise die Smartphone App verbinden. Diese Software wird zwischen den UAV-seitigen Systemen und den Planungs-, Überwachungs- und Anzeige-Komponenten des Leitstandes, welche auf Basis der Middelware ROS (Robotic Operating System)[28] kommunizieren, vermitteln. Zur einen Seite wird die UAV Kommunikationsschnittstelle als UAV Server über das Netzwerk angeboten, an dem sich die UAVs registrieren können. Zur anderen Seite wird mittels ROS Topics und ROS Services mit den Komponenten im TRADR Leitstand kommuniziert. Eingehende Telemetriedaten und Ergebnisse der Explorationsaufgaben, z.B. aufgenommene Bilder werden von dieser zentralen Komponente in der TRADR Missionsdatenbank gespeichert.

Alle Bereiche, die Benutzerschnittstellen zum Endanwender aufweisen, werden so gestaltet, dass sie von geschulten Feuerwehrleuten im Rettungseinsatz leicht zu verstehen und zu bedienen sind.

2 Stand der Technik

Inhaltsangabe

2.1 MAVLink

MAVLink „Micro Air Vehicle Link“ [46][56][90] ist ein Übertragungsformat zur Kommunikation zwischen UAVs und Bodenstationen über serielle Verbindungen. Es besteht eine feste Spezifikation des Formates sowie Bibliotheken zu dessen Nutzung. Zielgruppe sind dabei vor allem UAV Eigenbau-Projekte, in denen UAVs auf Basis von Mikrocontrollern und Mikrorechnern designed werden. MAVLink bietet diesen Projekten Lösungen an, um über serielle Punkt-zu-Punkt Funkverbindungen Daten zu übertragen. Dabei liefert es alle notwendigen Mechanismen von der Serialisierung der Nutzdaten bis zur Synchronisation des Übertragungsdatenstroms. Für die Serialisierung verwendet MAVLink ein eigenes binäres Datenformat. Bei den Inhalten (Nutzdaten), welche über die Verbindung transportiert werden, lässt MAVLink den Entwicklern viel Spielraum durch die Möglichkeit, eigene Nachrichtentypen zu definieren und Interaktionsabläufe umzusetzen. Entwickler können den Aufbau einer Nachricht in einer MAVLink-eigenen Interface Definition Language (IDL) definieren, die auf XML basiert[2]. Ein MAVLink-Nachrichten-Compiler erzeugt aus den in XML beschriebenen Nachrichtentypen Quelltext für eigene Projekte.

Neben selbst zu definierenden Nachrichten gibt es jedoch eine MAVLink Common Message Spezifikation [55], die eine große Menge an Referenz-Nachrichtentypen und Referenz-Interaktionsabläufe beschreibt[47]. Der Großteil der UAV-Anwendungen kann alleine auf Basis dieser Referenz-Nachrichten realisiert werden. Die MAVLink Common Message Spezifikation definiert Unterprotokolle, welche die Interaktionsabläufe für das Übertragen von Flugmissionen, aufgenommenen Standbildern, die Konfiguration von Parametern und die Übertragung von Livestreams regeln[1].

In Eigenbau-Projekten werden auf den UAVs meist fertige Flugcontroller eingesetzt. Viele dieser für Eigenbau-Projekte vorhandenen Komponenten verwenden MAVLink als Kommunikationsschnittstelle. Die Technologie ist in diesem Bereich der Standard für die Realisierung der Kommunikation zwischen UAV Komponenten und Bodenstationen. Der Großteil frei verfügbarer Bodenstations-Software unterstützt daher ebenfalls MAVLink. Neben Eigenbau-Projekten bieten mittlerweile auch einige kommerzielle UAV Hersteller eine MAVLink Schnittstelle zur Kommunikation zwischen ihren Produkten und eigenen Bodenstationskomponenten oder denen von Drittanbietern an. Durch die offen gehaltene Spezifikation sind die mit MAVLink umgesetzten Protokolle nicht immer vollständig kompatibel[90].

MAVLink kann auch als Anwendungsprotokoll in TCP/IP-Netzwerken eingesetzt werden. In diesem Fall wird jedoch nur die Common Message Spezifikation und die Serialisierung genutzt. Der Großteil der MAVLink Technologie, die auf den technisch darunter liegenden Ebenen anzusiedeln sind, werden in diesem Falle nicht genutzt, da dies durch die Instanzen des Netzwerkstapels (TCP/IP) gelöst wird. Zudem existieren Bridge-Lösungen, die das MAVLink mit der in der Robotik häufig genutzten Middleware ROS (Robotic Operating System) koppeln.

2.1.1 Übertragungsformat

MAVLink spezifiziert ein Übertragungsformat auf tiefer technischer Ebene, um eine Kommunikation über eine beliebige serielle Verbindung zu gewährleisten. Die Implementierungen, die in Form von Bibliotheken bereitgestellt werden, lassen sich auf vielen verschiedenen Mikrocontroller Plattformen nutzen. Dieses Übertragungsformat ermöglicht es dem Sender MAVLink-Nachrichten in einer solchen Weise zu versenden, dass diese vom Empfänger in einem dauerhaft empfangenen Bitstrom erkannt werden können. Die MAVLink Empfangsbibliothek liest jedes Byte des dauerhaft empfangenen Datenstroms, bis das Startbyte eingeht. Die Bitfolge des Startbytes ist fest definiert. Soll die gleiche Bitfolge im Inhalt der Nachricht übertragen werden, muss dies über eine Escape-Sequenz passieren. Durch das Erkennen des Startbytes ist die Kommunikation synchronisiert und der Beginn einer Nachricht erkannt. Es folgen auf das Startsymbol Bytes, die die Länge, den Sender und den Typ der Nachricht definieren, was notwendig ist, um den Inhalt korrekt zu deserialisieren[2].

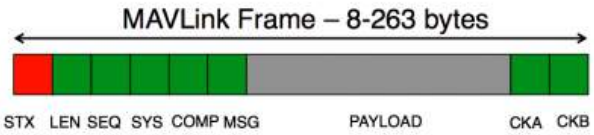

Abbildung 2.1: Aufbau des MAVLink-Übertragungsformats (Quelle: [56])

Byte Index	Name	Value	Erklärung
0	Start Byte	v1.0: 0xFE (v0.9: 0x55)	Zeigt dem Empfänger an, dass ein neues Datenpaket beginnt (Synchronisierung).
1	Payload length	0 - 255	Gibt die Länge des Daten-Feldes (Payload) in Bytes an.
2	Packet sequence	0 - 255	Jede Komponente nummeriert die von ihr gesendeten Pakete, sodass Paketverluste erkannt werden können
3	System ID	1 - 255	ID des sendenden Systems. Über dieses Feld kann der Empfänger erkennen, von welchem UAV das Paket versandt wurde
4	Component ID	0 - 255	ID der sendenden Komponente. Über dieses Feld kann der Empfänger erkennen, welche Komponente das Paket gesendet hat (z.B. IMU, Flight Controller, Kamera, Gimbal, usw.).
5	Message ID	0 - 255	ID des Nachrichtentyps. Beschreibt welchen Typ der Inhalt des Datenfeldes hat, dies ist notwendig, damit der Inhalt vom Empfänger korrekt interpretiert und dekodiert werden kann.
6 to (n+6)	Data	(0 - 255) bytes	Inhalt der Nachricht
(n+7) to (n+8)	Checksum	(low byte, high byte)	ITU X.25/SAE AS-4 Checksumme. Mittels der Checksumme kann der Empfänger überprüfen ob das Datenpaket fehlerfrei übertragen wurde.

Tabelle 2.1: Format eines MAVLink-Datenpaketes[56]

2.2 UAVs als IoT-Gerät in der Cloud

Forscher von der UAE-University in den Vereinigten Arabischen Emiraten haben mit ihren Kollegen von der Robert Morris University in Panselvanier, USA, gezeigt, wie sich UAVs als Internet of Things (IoT) Geräte in Cloud Computing Infrastrukturen integrieren lassen. [44]

Der ungewöhnliche neuartige Ansatz bestand darin, bei der Integration an so vielen Stellen wie möglich Technologien und Konzepte, die in den Bereichen Web und IoT eingesetzt werden, zu benutzen. Dies ist gut durch eines der beschriebenen Szenarien zu verdeutlichen. Die UAVs sind über Drahtlosnetzwerkverbindungen oder Mobilfunk-Technologien wie LTE direkt mit dem Internet verbunden. Über diese Verbindung kann die Software, die auf den Onboard-Rechnern der UAVs ausgeführt wird, im Internet befindliche Cloud Services wie z.B. Google Maps nutzen. Die UAVs erhalten Anfragen von einem Cloud Service, der die Missionen verwaltet, die von ihnen ausgeführt werden sollen. Die Missionen sind innerhalb des Cloud Service in aus dem Web üblichen MySQL-Datenbanken relational abgespeichert. Um einem UAV eine Mission zuzuweisen, kann ein Operator über eine Web-Anwendung, die im Browser ausgeführt wird, mit dem Cloud Service kommunizieren und eine Mission in der Webanwendung zusammenstellen. Die Mission, die aus Wegpunkten besteht, wird in der Datenbank des Cloud Service abgelegt und kann von diesem an das UAV gesendet werden. Die Kommunikation zwischen den UAVs und dem Cloud Service wird mittels einer RESTful-API umgesetzt, genauso wie Kommunikation zwischen der beim Client laufenden Web-Anwendung und dem Cloud Service. Bei der RESTful-API handelt es sich um eine zustandslose Kommunikation, die Remote Procedure Calls (RPC) auf Basis von HTTP und HTTPS nutzt. Der Nachrichteninhalt wird in Form von definierten XML- oder JSON-Nachrichten übertragen. Dabei findet die Interaktion nach den Regeln der SOAP-Architektur (Service Orientated Architecture Protocol) statt. Jedes UAV stellt somit eine IoT Ressource dar, die über den Cloud Service angefragt und benutzt werden kann. Die Sensoren, Aktoren und weiteren Komponenten jedes UAVs können über im Web gebräuchlichen Unified Ressource Identifyers (URI) adressiert werden. Die RESTful-API sieht die vier Kommandos "Get", "Post", "Putünd "Delete"vor. In Kombination mit URIs als Parameter können so zum Beispiel mit dem Get-Kommando Status-Informationen vom UAV abgefragt werden. Hingegen kann mit dem Post-Kommando einer Ressource Daten geschickt werden. Beispielsweise könnte man der Kamera den Auftrag schicken, ein Foto von einer bestimmten Position zu machen. Dann ist es möglich, diesen der Kamera erteilten Auftrag mit dem Delete-Kommando wieder zu löschen. Die Kommandos werden als HTTP-Anfrage formuliert. Hier wird für den Inhalt der Anfrage entweder JSON oder XML als Format benutzt. Auf jede Anfrage antwortet die IoT Ressource, also das UAV, mit einer HTTP-Antwort. In dieser wird unter anderem mitgeteilt, ob das RESTful-Kommando erfolgreich ausgeführt wurde. [44]

2.3 Kommerzielle Komplettlösung von UgCS

UgCS ist ein Anbieter von Software-Lösungen zur Steuerung und Überwachung autonom fliegender UAVs. Zielgruppe sind Firmen, die UAVs geschäftlich einsetzen, z.B. im Vermessungswesen, bei Filmproduktionen oder der Inspektion von Bauwerken. UgCS bietet dabei mit mehreren Produkten einen sehr großen Funktionsumfang. Darunter fallen Software-Lösungen für die Missions-Planung und Missionsüberwachung sowie dem Postprocessing für Vermessungsmissionen auf Basis von Photogrammetrie mittels 2D Bilddaten, LiDAR-Daten, Geo-Radar und Magnetfeld-Messungen. Hinzu kommen Spezialanwendungen für die Inspektion von z.B. Photovoltaik Anlagen und Hochspannungsfreileitungen. Auch Software-Lösungen für die Koordination von Drohnen-Schwärmen nennt UgCS auf seiner Webseite.[74]

Das jedoch im Kontext dieser Arbeit interessanteste Paket ist die ÜgCS for Command CenterSSoftware. Ihre Zielgruppe sind Polizei, Sicherheitsbehörden und Grenzschutz, Feuerwehr und Rettungskräfte. Die Rettungsrobotik mit Search and Rescue Missionen in Katastrophengebieten nennt der Hersteller konkret als Anwendungsgebiet. UgCS beschreibt dabei eine Struktur, die der im TRADR Projekt sehr stark gleicht. Ein zentraler Kommandoleitstand mit UAV Operatoren, die die Mission überwachen und UAVs steuern können, bildet die Basis des Produktes. Alle gesammelten Daten laufen an dieser Stelle zusammen und werden zentral ausgewertet und allen Instanzen zur Verfügung gestellt. Zudem gibt es jedoch Einsatzkräfte sowie UAV Piloten im Einsatzgebiet. Die Piloten besitzen Sichtkontakt zu den jeweiligen UAVs. Die Vernetzung zwischen allen Instanzen wird durch die Software UgCS in Kombination mit von UgCS referenzierter Hardware realisiert. UgCS hebt besonders hervor, dass die Datenübertragung an allen Stellen verschlüsselt und sicher realisiert werden kann.[77]

UgCS bietet selber keine UAVs an, sondern Adapter-Software, um die kommerziellen UAVs vieler Hersteller an die UgCS Systeme anzubinden. Eine große Auswahl an DJI UAVs werden dabei ebenfalls unterstützt. Die Anbindung der Kompakt UAVs von DJI wie z.B. Phantom 3/4 realisiert UgCS ebenfalls über eine eigene auf dem DJI Mobile SDK basierenden Piloten-App, die über eine proprietäre Schnittstelle mit den Leitstandsystemen von UgCS kommuniziert[75][76].

3 Systemdesign

Inhaltsangabe

3.1 Topologie

Dieser Abschnitt dient zum einen dazu, alle für den Sachverhalt notwendigen Hardware- und Software-Komponenten sowie Akteure und deren Rollen vorzustellen. Zum anderen sollen die Beziehungen und Verbindungen der genannten Dinge aufgezeigt und erklärt werden.

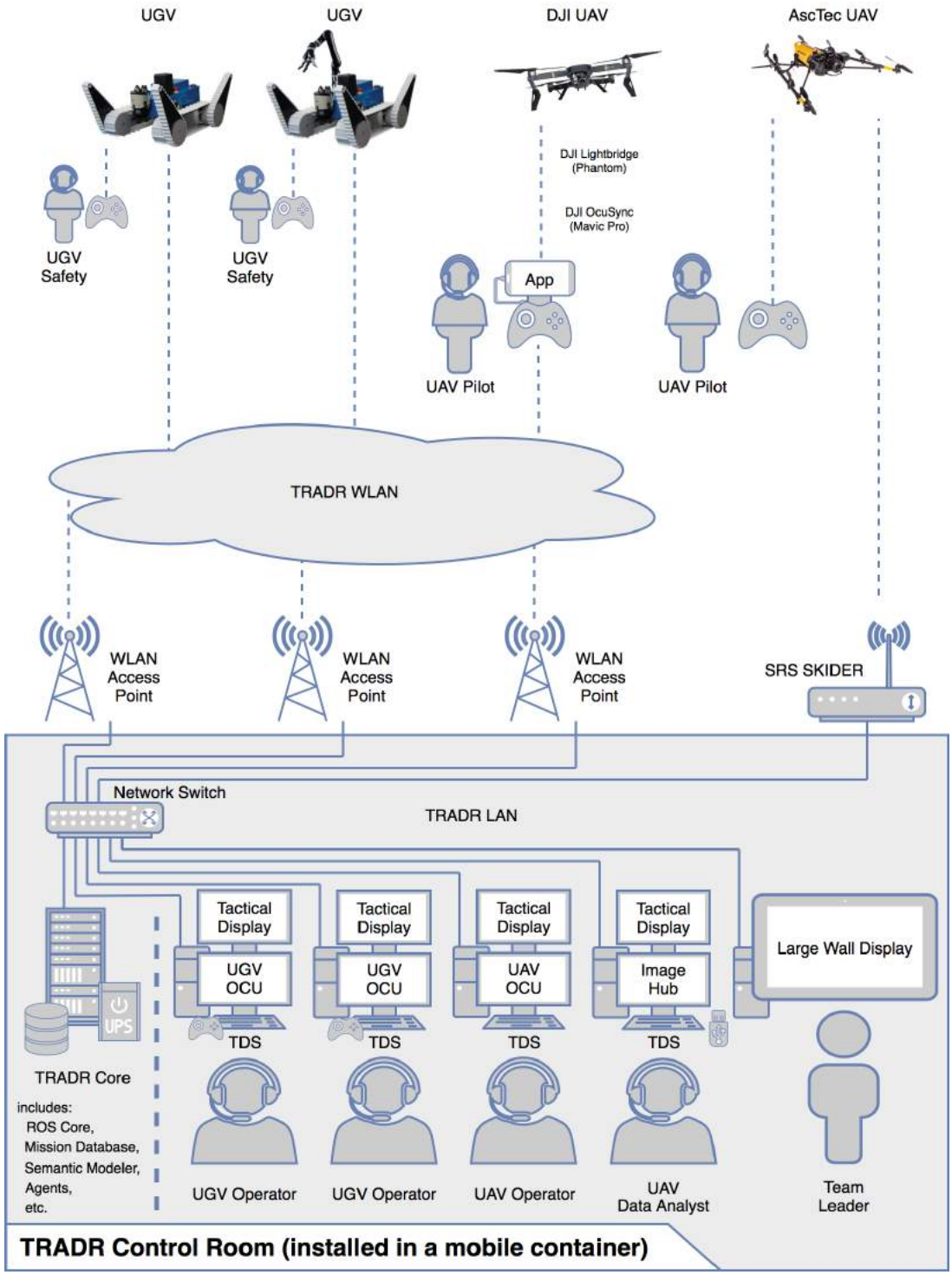

Abbildung 3.1: Topologie der Hardwarekomponenten des TRADR Systems

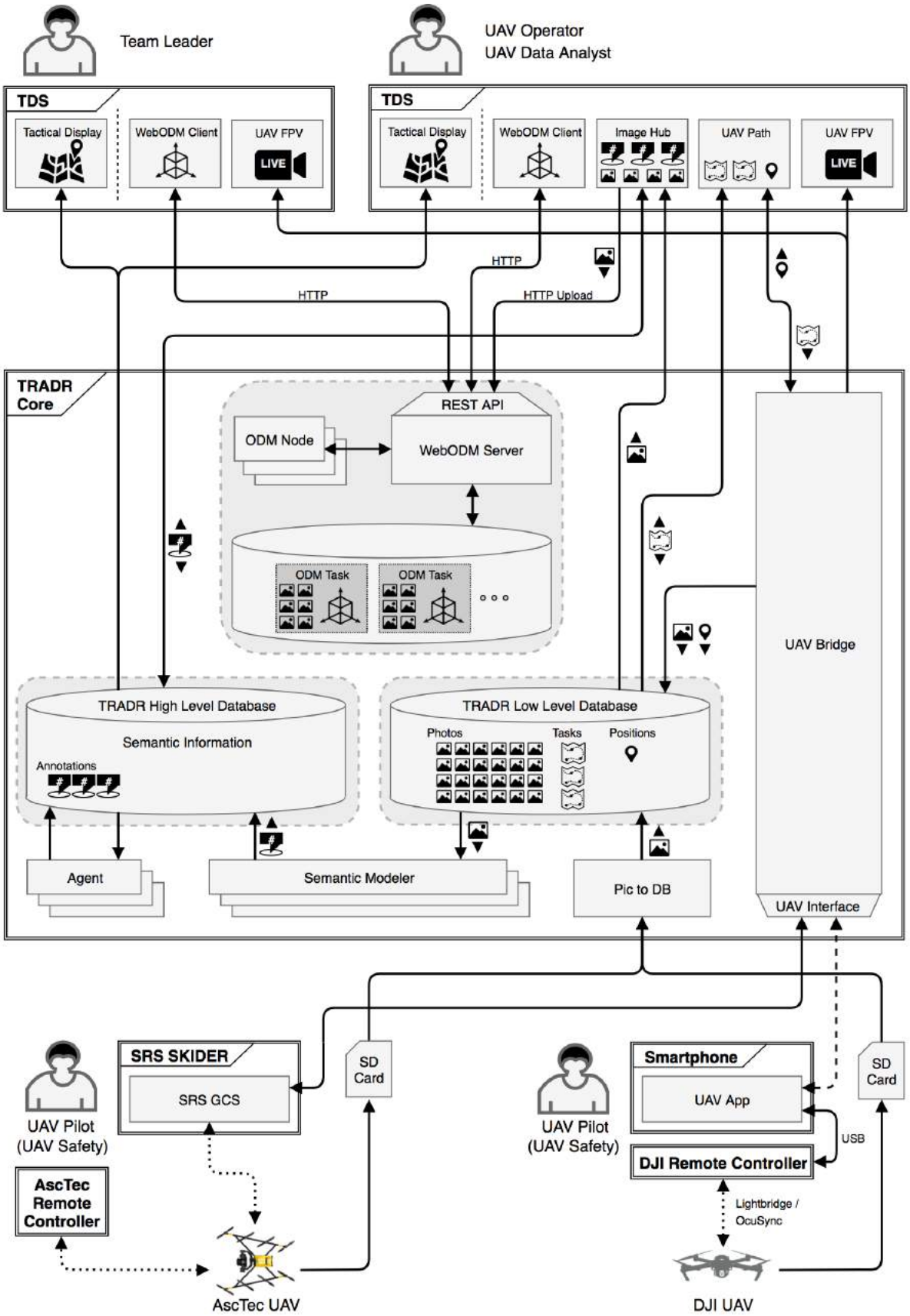

Abbildung 3.2: Topologie der für die UAV-Integration relevanten Softwarekomponenten des TRADR Systems

3.1.1 TRADR Missions Datenbank

Eine zentrale Instanz im TRADR System ist die Missionsdatenbank. Für jede Mission wird eine neue Datenbank angelegt, in der alle im Missionsverlauf anfallenden Daten persistent gespeichert werden. Eine Mission ist z.B. ein Rettungseinsatz, der mehrere Tage andauern kann. Der gesamte Einsatz ist aufgeteilt in Untereinsätze, den Sorties. Diese bestehen wiederum aus Aufgaben, den Tasks. Das Explorieren eines Gebietes durch ein UAV ist in der Definition von TRADR eine Aufgabe. Der Datenbank können während einer Mission nur Daten hinzugefügt werden. Es ist nicht möglich einmal hinzugefügte Daten zu löschen oder zu verändern. Gibt es Datensätze, die einer Änderung über die Missionsdauer hinweg unterliegen, müssen die Änderungen als neuer Datensatz hinzugefügt werden. Dies schützt davor, dass einmal generierte Daten verloren gehen oder verfälscht werden können.

Die Missionsdatenbank teilt sich in eine High Level Database und eine Low Level Database auf. Beide Teile sind für verschiedene Aufgabenbereiche im Projekt zuständig und technisch mit unterschiedlichen Datenbank Management Systemen realisiert. Von den Robotern und Einsatzkräften gesammelte Rohdaten wie Photos, Laserscanns, Messwerte, usw. werden in der Low Level Database abgelegt. Zudem werden dort auch von den Operatoren geplante Aufgaben für die Roboter gespeichert. Intelligente Softwarekomponenten analysieren die Rohdaten der Low Level Daterbase, erzeugen daraus semantische Informationen und speichern diese in der High Level Datenbank. Diese Softwarekomponenten heißen Semantic Modeller. Ein Beispiel dafür ist eine Software, die bestimmte Objekte in den Luftbildern eines von einem UAV abgeflogenen Gebietes erkennt und die Position der erkannten Objekte in der High Level Database ablegt. Die semantischen Informationen der High Level Database werden von intelligenten Software Agenten durchsucht, die in diesen Zusammenhänge erkennen und daraufhin dem Einsatzteam weitere Erkenntnisse und Handlungsvorschläge anbieten.

Bezüglich der UAV Integration in das TRADR System werden zum einen die Telemetriedaten des UAVs sowie die während der Flüge aufgenommenen Bilder in der Low Level Database abgelegt. Zum anderen werden die mit der Planungssoftware erstellten Explorationsaufgaben ebenfalls dort gespeichert.

3.1.2 TRADR Leitstand

Der Leitstand im TRADR Projekt besteht aus mehreren Bedienplätzen, die in einem mobilen Einsatz-Container eingerichtet sind. Jeder Bedienplatz besteht aus zwei großen Monitoren und den für die jeweilige Rolle notwendigen Eingabegeräten. Die Hard- und Software jedes Bedienplatzes wird als TRADR Display System (TDS) bezeichnet. Es gibt TDS-Arbeitsplätze für Roboter Operatoren und für den Team-Leiter.

Auf einem der zwei Monitore jedes TDS wird das „Taktische Display“ angezeigt. Auf diesem wird gemeinsam die Mission geplant und begutachtet. Die Erkenntnisse, die in den einzelnen Teams gewonnen werden, sind dort zusammengefasst darstellt und stehen somit jeder Einsatzkraft im Leitstand zur Verfügung. Auf dem anderen Monitor befindet sich die Benutzeroberfläche zur Umsetzung der rollenspezifischen Aufgaben. Ein Operator sieht alle notwendigen Anzeigeinstrumente und Kamerabilder, um den von ihm kontrollierten Roboter teleoperiert zu steuern oder diesem autonom auszuführende Aktionen zu befehlen.

Die Softwarekomposition, die der Operator auf seinem TDS zur Steuerung des Roboters präsentiert bekommt, wird im TRADR Projekt als Operator Control Unit (OCU) bezeichnet. Es gibt zwei Typen von OCUs. Die eine ist zur Steuerung der in TRADR eingesetzten Bodenroboter (UGV) entwickelt, die andere ist für die Arbeit mit Flugdrohnen (UGV) gedacht.

Die OCU für Flugdrohen besteht aus zwei Softwarekomponenten, die dem UAV Operator zur Verfügung stehen. Erstens ist das die UAV Planungs- und Überwachungssoftware (UAV Path), mit der Explorationsaufgaben erstellt und UAVs zur Ausführung zugewiesenen werden können. Diese Software dient ebenfalls der Überwachung aller UAVs. Zweitens gibt es die UAV Livebild Anzeige (UAV FPV), mit der der Videostream einer UAV Kamera angefordert und dargestellt wird.

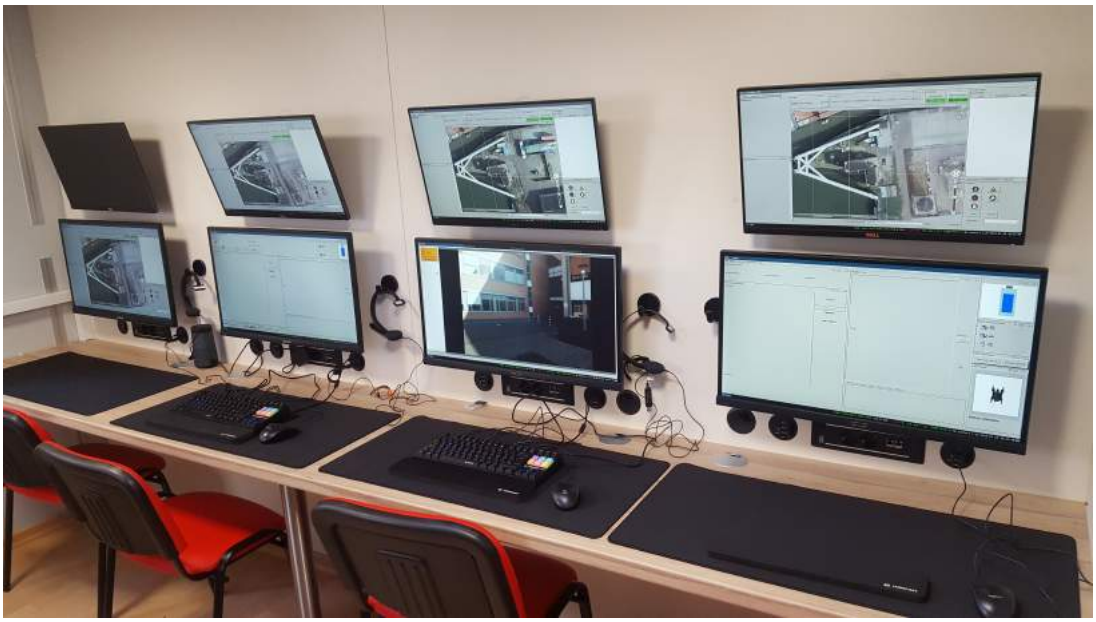

Abbildung 3.3: TRADR Leitstand mit 4 TDS Arbeitsplätzen für Roboter Operatore

UAV Planungs- und Überwachungssoftware

Die UAV Planungs- und Überwachungssoftware zeigt dem Operator eine Karte, auf der dieser die Flugbahnen möglicher Explorationsaufgaben entwirft, die zu einem gewünschten Zeitpunkt von UAVs abgeflogen werden sollen.

Die Software ist als MapViz Plugin realisiert, welches es ermöglicht auf den Kartenebenen von MapViz Flugbahnen für Explorationsaufgaben zu definieren. Dabei ist es möglich, kreisförmige oder mäanderförmige Flugbahnen zu generieren. Nach dem Erstellen einer Aufgabe wird diese automatisch in der Missionsdatenbank abgespeichert. In einer Liste werden die am System angemeldeten UAVs angezeigt. Erstellte Aufgaben können einem Piloten mit UAV übermittelt werden, damit dieser sie ausführt. Die Positionen und bereits geflogenen Flugspuren aller UAVs werden ebenfalls über den MapViz Karten eingeblendet. Zudem sind der Batteriestatus und die Telemetriedaten der UAVs ablesbar.[61][62][67]

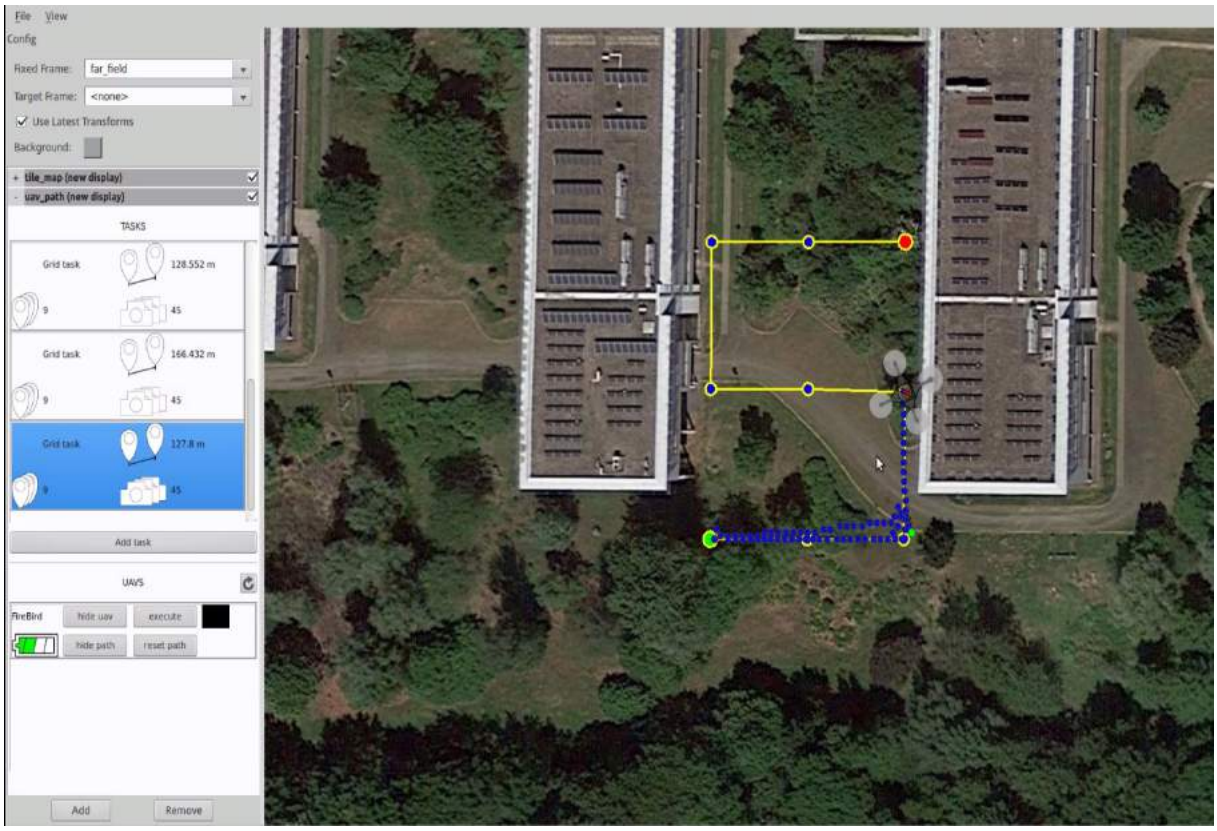

Abbildung 3.4: Planungs- und Überwachungssoftware der UAV OCU des TRADR Leitstandes

UAV Livebild Anzeigesoftware

Bei der Livebild-Software handelt es sich um eine Anzeigesoftware, die den Livestream skalierbar auf dem Bildschirm darstellt. Die Software listet alle angemeldeten UAVs mit ihren Kameras. Über Schaltflächen kann der UAV Operator das Livebild eines bestimmten UAVs anfordern und anzeigen lassen. Der Videostream erreicht die Anzeige-Software als H.264 kodierter Datenstrom. Innerhalb der Anzeige-Software läuft ein Dekoder, der den Videostream dekodiert, um diesen darstellen zu können.

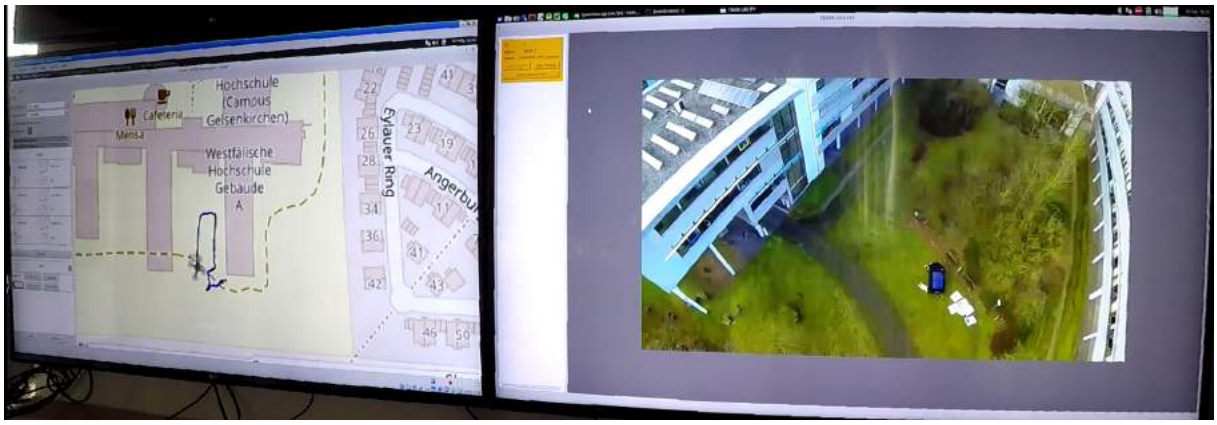

Abbildung 3.5: Planungs- und Überwachungssoftware (links) mit Livebild-Anzeige (rechts)

3.1.3 TRADR UAV App

Im Einsatzgebiet befindet sich der UAV Pilot mit der Flugdrohne, deren Flug er mittels einer Fernsteuerung kontrolliert. An der Fernsteuerung ist ein Smartphone oder Tablet befestigt. Beide Geräte sind über die USB-Schnittstelle miteinander verbunden. Auf dem Smartphone wird die TRADR UAV App ausgeführt, mit der der Pilot entweder selbst mittels der App erstellte oder vom UAV Operator im Leitstand definierte Explorationsaufgaben vom UAV ausführen lassen kann. Die Smartphone App empfängt das Livebild der UAV Kamera sowie die Telemetriedaten des UAVs und sendet diese über eine WLAN-Verbindung an den TRADR Leitstand.

Abbildung 3.6: UAV Pilot mit Fernsteuerung und Smartphone

3.1.4 TRADR UAV Bridge

Die UAV Bridge ist die zentrale Softwarekomponente der UAV Integration in das TRADR System. Diese Bodenstationssoftware läuft auf dem zentralen TRADR Core. Alle an einer Mission teilnehmenden UAVs melden sich bei dieser Instanz an. Jedes UAV wird intern mit einer ID-Nummer geführt und kann über diese von den OCU-Komponenten im Leitstand angesprochen werden.

Diese Bridge kommuniziert mit den UAV-seitigen Systemen (Smartphone App, SRS SKIDER System) mittels dem TRADR UAV Interface über Netzwerkverbindungen (kabelgebunden oder drahtlos). Leitstands-seitig ist sie mit den verschiedenen Softwarekomponenten des UAV-OCU und der TRADR Missionsdatenbank verbunden. Für diesen Datenaustausch innerhalb des TRADR Systems wird die Middelware ROS (Robotic Operating System)[28] verwendet.

3.2 Smartphone App

Im folgenden Abschnitt ist die für das TRADR Projekt entwickelte Piloten App beschrieben, die auf dem Android Smartphone des Piloten installiert ist. Sie ist in der Lage, über die USB-Schnittstelle des Smartphones mit der angeschlossenen Fernsteuerung des DJI UAVs zu kommunizieren. Die App ist auf Basis des von DJI herausgegebenen Mobile SDKs[19][43] entwickelt worden und eignet sich daher technisch ausschließlich für UAVs der Firma DJI. Mithilfe der App kann der Pilot auf seinem Smartphone die Position des UAVs in einer Karte sehen und das Livebild der UAV Kamera betrachten. Zudem ermöglicht die App das durch einen Assistenten unterstützte Erstellen von Explorationsaufgaben.

Die große Besonderheit dieser App ist die Anbindung an den TRADR Leitstand über ein allgemein gehaltenes Anwendungsprotokoll. Das Smartphone ist über seine Drahtlosnetzwerkkarte mit dem WLAN-Netz des TRADR Systems verbunden. Über diese Netzwerkverbindung kommuniziert die App mit dem TRADR Leitstand und stellt diesem Telemetrie-Daten und den Livestream des UAVs zur Verfügung. Im Weiteren können vom UAV Operator im Leitstand geplante Aufgaben auf die App übertragen und dem Piloten dargestellt werden. Dieser kann daraufhin die Ausführung durch das UAV veranlassen.

3.2.1 Startbildschirm

Nach dem Starten der Smartphone App wird der Startbildschirm angezeigt. Auf diesem ist zu erkennen, ob bereits eine Verbindung zur Fernsteuerung des UAVs und zwischen dieser und dem UAV aufgebaut ist. Besteht bereits eine Verbindung, ist dies durch den Schriftzug „DJIAircraft connected“ zu erkennen. Ansonsten ist „connection loose“ zu lesen. Im diesem Fall muss erst eine Verbindung aufgebaut werden. Sind UAV und Fernsteuerung eingeschaltet und befinden sich in Reichweite, sollte die erfolgreiche Kopplung beider Geräte entweder auf dem Display der Fernsteuerung oder durch das Blinken der UAV-LEDs zu erkennen sein. Ist die Fernsteuerung darüber hinaus noch über ein USB-Kabel mit dem sich im richtigen Modus befindenden Smartphone angeschlossen, kann die Kopplung stattfinden. Das Android Betriebssystem zeigt eine Meldung, ob das angeschlossene Gerät mit der TRADR UAV App gekoppelt werden soll. Einige Sekunden nach dem Bestätigen sollte der Schriftzug in der App von „connection loose“ zu „DJIAircraft connected“ wechseln und auf dem Startbildschirm der App wird zusätzlich die Modell-Bezeichnung des UAVs angezeigt.

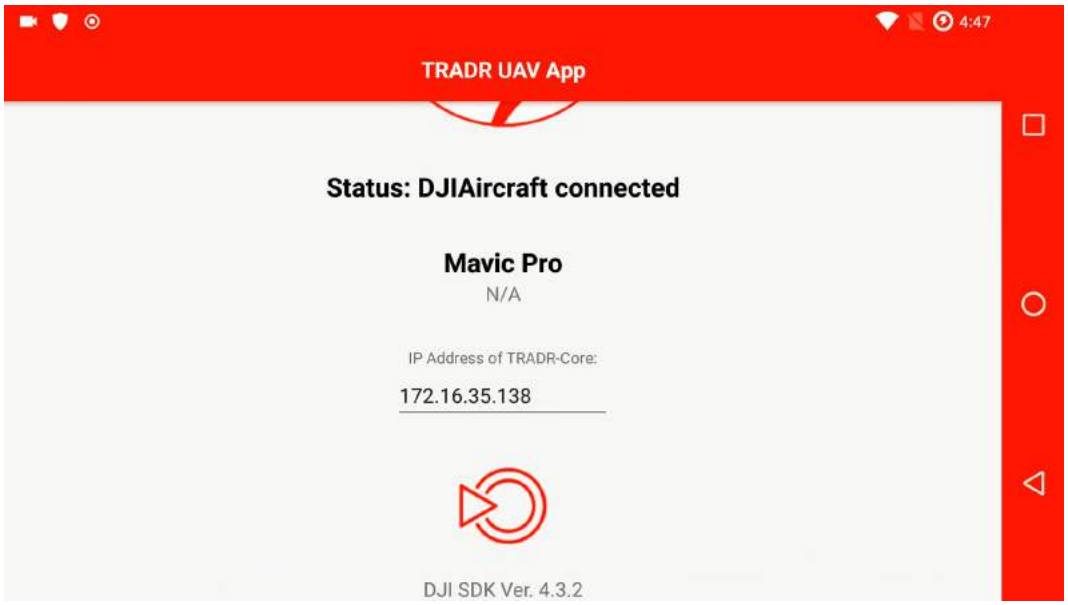

Abbildung 3.7: TRADR UAV App - Startbildschirm

Darüber hinaus ist auf dem Startbildschirm die voreingestellte IP-Adresse des TRADR Core zu sehen, auf dem die TRADR UAV Bridge läuft. Sollte die voreingestellte Adresse nicht korrekt sein, kann sie hier geändert werden. Die Kenntnis der IP-Adresse ist notwendig, da das Smartphone versucht, sich mit der Bridge auf dem TRADR Core zu verbinden, um mit dem Leitstand zu kommunizieren.

Nach korrekter Verbindung zwischen App und UAV kann über Berühren des roten Symbols am unteren Rand zur Kartenansicht gewechselt werden.

3.2.2 Kartenansicht

Auf der Benutzeroberfläche ist im Vollbild die Karte zu erkennen. Standardmäßig erscheint diese lokalisiert auf den Ort, an dem sich das UAV befindet. Dazu liest die App den Standort des UAVs aus und bewegt den Kartenausschnitt an diesen Ort. Besitzt das UAV keinen validen Standort, da es z.B. kein GPS Signal empfängt, wird die Karte an einem Standard-Standort wie z.B. dem Schloss Birlinghoven in Sankt Augustin angezeigt. Sobald das UAV jedoch einen validen Standort besitzt, wird dies von der App erkannt und dieser wird automatisch ausgelesen. Der Kartenausschnitt wird dann aktualisiert. Diese automatische Aktualisierung geschieht jedoch nur, wenn der Standort des UAV von invalide zu valide wechselt. Bei weiteren folgenden Standortaktualisierungen bleibt der Kartenausschnitt stabil.

Mit den von der Smartphonebedienung bekannten Berührungs-Gesten wie z.B. dem Wischen mit einem Finger und dem Spreizen zweier Fingern auf dem Bildschirm kann der Kartenausschnitt bewegt sowie vergrößert und verkleinert werden.

Die Position des UAVs wird durch ein blaues Quadrokopter-Symbol in der Karte dargestellt. Auch die Orientierung und Drehbewegungen um die Gier-Achse des UAVs lassen sich durch die Orientierung und Drehung des Symbols ablesen.

An der oberen rechten Seite wird in einem rechteckigen Bereich das Livebild der UAV Kamera angezeigt. Durch das Berühren dieses Bereiches gelangt man in die FPV-Ansicht.

An der unteren rechten Seite ist das Symbol zum Wechseln der Kartenansicht. Durch das Berühren dieses Symbols wechselt man in einer Schleife von einem Ansichtstyp zum nächsten. Die Ansichtstypen sind Luftbild, Luftbild fusioniert mit Straßenkarte, Straßenkarte und Geländekarte.

Durch das Berühren des allgemein bekannten Menü-Symbols an der linken oberen Ecke öffnet sich das Menü. Über die Symbole des Menüs ist es möglich, zu den verschiedenen Assistenten zu gelangen, mit denen Flugbahnen für autonome Explorationsaufgaben erstellt werden können.

Zentriert am oberen Rand steht in grün der Schriftzug „connected“. Dies symbolisiert, dass die Smartphone App korrekt mit der Fernbedienung des UAVs und dem UAV verbunden ist. Funktioniert eine der Verbindungen nicht korrekt, wechselt der Schriftzug in ein rot geschriebenes „disconnected“. Wird die unterbrochene oder gestörte Verbindung wiedererlangt, wechselt der Schriftzug automatisch und die Daten in der App werden aktualisiert. Die App muss also nicht in jedem Fall eines Verbindungsabbruchs neu gestartet werden. In den meisten Fällen kann die App ohne Neustart die Verbindung zum UAV und zur Fernbedienung wiedererlangen.

Abbildung 3.8: TRADR UAV App - Kartenansicht mit geöffnetem Menü

3.2.3 FPV Ansicht

Die FPV Ansicht (First Person View) zeigt das Livebild der UAV-Kamera im Vollbildmodus. Der Pilot kann diese Ansicht verwenden, um die Kamera und deren Ausrichtung durch den Gimbal genau einzustellen. Zudem kann er in der Vollbildansicht auch Details im empfangenen HD-Bild erkennen.

Darüber hinaus bietet diese Ansicht mehrere Kamera Funktionen an. Dazu zählen zum einen das manuelle Auslösen einer Fotoaufnahme. Das aufgenommene Bild wird dabei auf die Speicherkarte des UAVs geschrieben und kann zu einem späteren Zeitpunkt von dieser heruntergeladen werden. Es handelt sich also nicht um einen Screenshot des Livebildes, sondern um ein digitales Foto in voller Auflösung. Zum anderen kann die Aufnahme eines Videos manuell gestartet und gestoppt werden. Die Videodatei, in der das Video gespeichert wird, liegt ebenfalls auf der Speicherkarte des UAVs.

Abbildung 3.9: TRADR UAV App - FPV Ansicht

3.2.4 Software-Assistenten

Zum Erstellen von Explorationsaufgaben enthält die Smartphone App zwei Software-Assistenten, mit denen kreis- oder mäanderförmige Flugbahnen geplant werden können.

Die App ist hinsichtlich ihrer Architektur so designed worden, dass es einheitliche und klare Schnittstellen für die Interaktion zwischen einem Assistenten und den restlichen Komponenten der App gibt, wie beispielsweise der Kartendarstellung. Zudem ist das Output-Format für Explorationsaufgaben, dass von der Ausführungskomponente erwartet wird, einheitlich beschrieben. Dies erlaubt es, in einfacher Weise weitere Assistenten zu entwickeln und in die App zu integrieren, ohne dass Änderungen an anderen Stellen der App nötig sind.

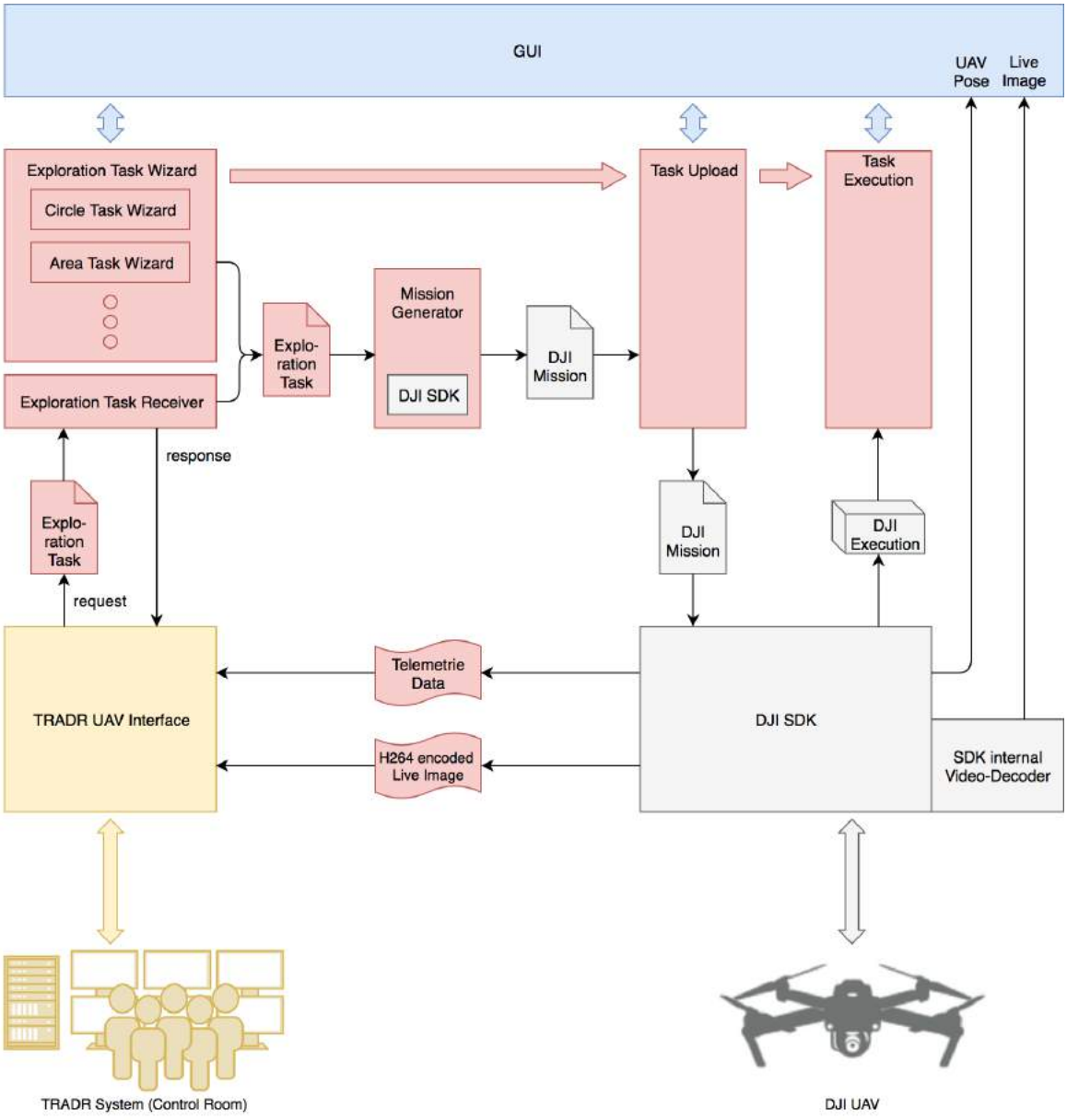

Abbildung 3.10: interner Aufbau der TRADR UAV App

Circle Task Wizard

Der „Circle Task Wizard“ ermöglicht dem Anwender mittels einer einfach verständlichen Benutzerführung die Erstellung von „Circle Tasks“. Ein „Circle Task“ ist eine Explorationsaufgabe, bei der ein Objekt kreisförmig umflogen wird. Dabei nimmt das UAV an verschiedenen Punkten auf der Kreisbahn Fotos von dem Objekt auf. Das UAV stoppt an jedem Punkt auf der Kreisbahn, an dem Aufnahmen getätigt werden, damit auch bei schlechteren Lichtverhältnissen keine Bewegungsunschärfe entsteht. Dieser Aufgabentyp eignet sich besonders gut, wenn aus den aufgenommenen Fotos mittels eines Structure-From-Motion-Verfahrens (SFM) ein 3D Modell des Objektes erzeugt werden soll.

Im Assistenten gibt der Benutzer durch Berühren von Punkten auf der Karte den Ort des Objektes und den Radius des Kreises an. In Folgeschritten können durch das Wischen von eingeblendeten Schiebereglern weitere Parameter wie Flughöhe, Abstände, Nick-Winkel des Gimbals, usw. festgelegt werden. Beim Verändern der Parameter werden zum Teil direkt Ergebnisse wie beispielsweise die resultierenden Wegpunkte angezeigt.

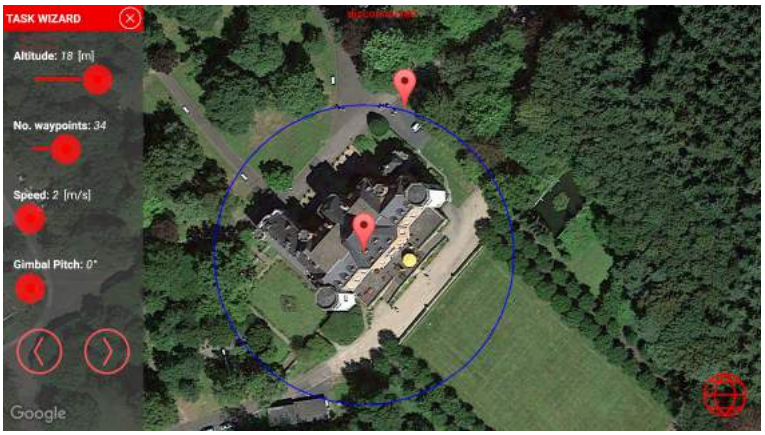

Abbildung 3.11: TRADR UAV App - Assistent zum Erstellen kreisförmiger Flugbahnen

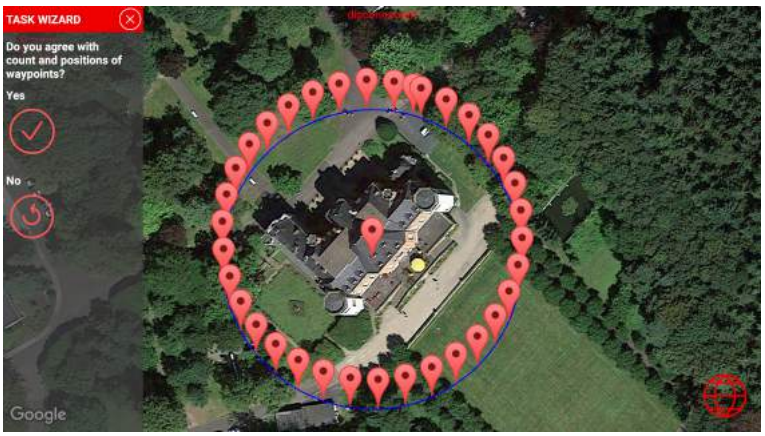

Abbildung 3.12: TRADR UAV App - Assistent zum Erstellen kreisförmiger Flugbahnen

Area Task Wizard

Der „Area Task Wizard“ führt den Anwender durch einen Benutzerdialog zum Erstellen von „Area Tasks“. Ein „Area Task“ ist eine Explorationsaufgabe, bei der ein durch ein Polygon beschriebenes Gebiet mäanderförmig von der Drohne überflogen wird. Beim Überfliegen werden an entsprechenden Wegpunkten mehrere Bilder aufgenommen. Die Wegpunkte sind in einem Raster über dem Gebiet verteilt. Zum Aufnehmen der Bilder stoppt das UAV an jedem Wegpunkt, damit auch bei schlechteren Lichtverhältnissen keine Bewegungsunschärfe entsteht. Für die Aufnahmen wird die Kamera nacheinander in verschiedene Himmelsrichtungen ausgerichtet und unterschiedlich geneigt. An jedem Wegpunkt ist es möglich, durch das Zusammensetzen der gemachten Bilder dieses Punktes ein nach unten gerichtetes Halbkugel-Panorama zu erhalten. Dieser Aufgabentyp eignet sich besonders gut, um aus den gemachten Fotos eine orthografische Karte (Ortho-Map) zu erstellen.

Im Assistenten legt der Benutzer durch Berühren von Punkten auf der Karte die Eckpunkte eines Polygons fest, um so das zu explorierende Gebiet zu spezifizieren. Der Innenraum des Polygons ist die Fläche, die vom UAV mäanderförmig angeflogen wird. Weitere Parameter wie Flughöhe, Abstände des Rasters, Nick-Winkel des Gimbals, usw. kann der Benutzer nacheinander durch das Wischen von eingeblendeten Schiebereglern festlegen. Auch in diesem Fall werden die resultierende Flugbahn und die Wegpunkte während des Einstellens in der Karte dargestellt.

Abbildung 3.13: TRADR UAV App - Assistent zum Erstellen mäanderförmiger Flugbahnen

Abbildung 3.14: TRADR UAV App - Assistent zum Erstellen mäanderförmiger Flugbahnen

3.2.5 Empfang von Aufgaben

Neben dem Erstellen von Explorationsaufgaben mittels eines der Assistenten, ist es auch möglich, solche Aufgaben von einer externen Quelle wie dem TRADR Leitstand zu empfangen. Die App ist, sobald eine Verbindung zum TRADR System herstellt ist, dauerhaft empfangsbereit für Aufgaben, die vom UAV Operator aus dem Leitstand eingehen.

Nach dem Empfang einer Aufgabe wird dieser dem Piloten auf dem Smartphone angezeigt. Er kann, sobald er bereit dazu ist, entscheiden, ob er die Aufgabe angezeigt bekommen möchte, oder diese ungesehen ablehnen will. Durch Berühren von Schaltflächen auf dem Bildschirm kann der Pilot seine Entscheidung dem System und dem UAV Operator mitteilen.

Lässt er sich die Aufgabe anzeigen, wird die Flugbahn in der Kartenansicht eingeblendet. Der Pilot kann nun evaluieren, ob der autonome Flug sicher durchgeführt werden kann. Bestätigt der Pilot die Flugbahn ebenfalls, gelangt er in die Benutzeroberfläche für den Upload und die Ausführung der Aufgabe.

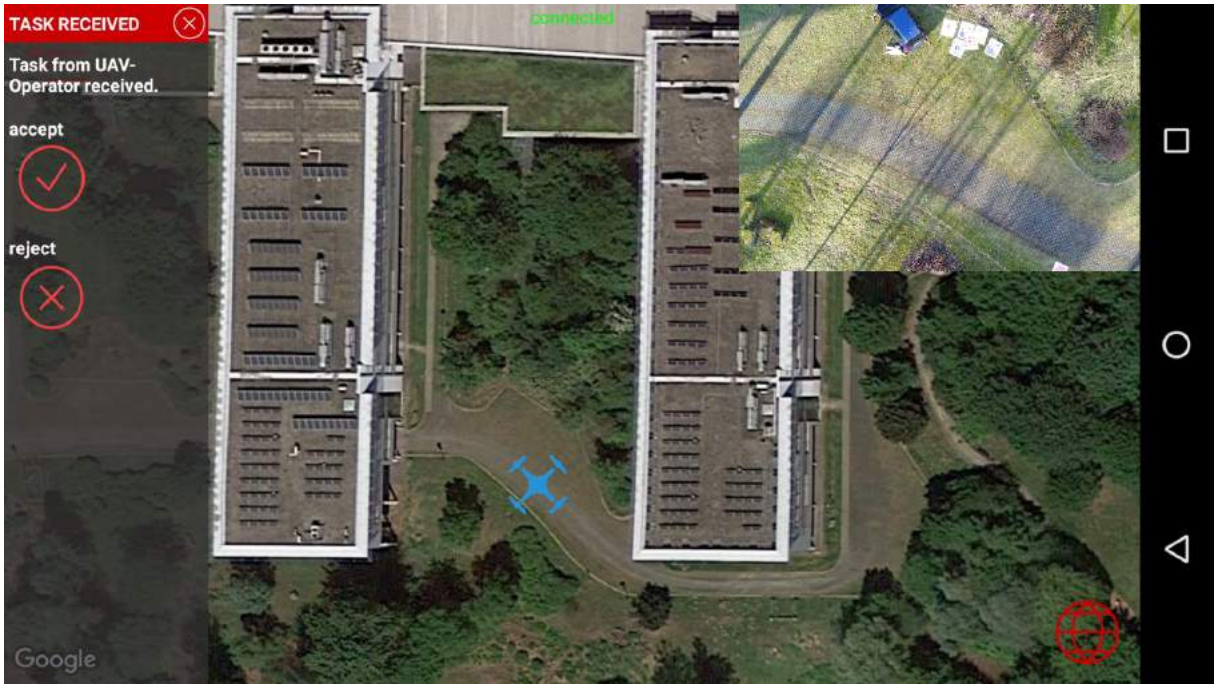

Abbildung 3.15: TRADR UAV App - Empfang einer vom UAV Operator erstellten Explorationsaufgabe

3.2.6 Ausführung

Nach dem Erstellen oder Empfangen einer Explorationsaufgabe kann diese ausgeführt werden. Dafür muss zuerst ein Upload der Anweisungen auf das UAV erfolgen. Ist das Hochladen erfolgreich abgeschlossen worden, kann die Ausführung gestaret werden. Die Flugbahn mit den Wegpunkten, die vorher weiß dargestellt waren, werden nun blau angezeigt. In der Karte ist zu beobachten, wie sich das UAV bewegt und seine Position ändert. Die Wegpunkte werden nacheinander abgeflogen und die Aktionen an jedem Wegpunkt ausgeführt. Der Fortschritt wird dabei grafisch dargestellt.

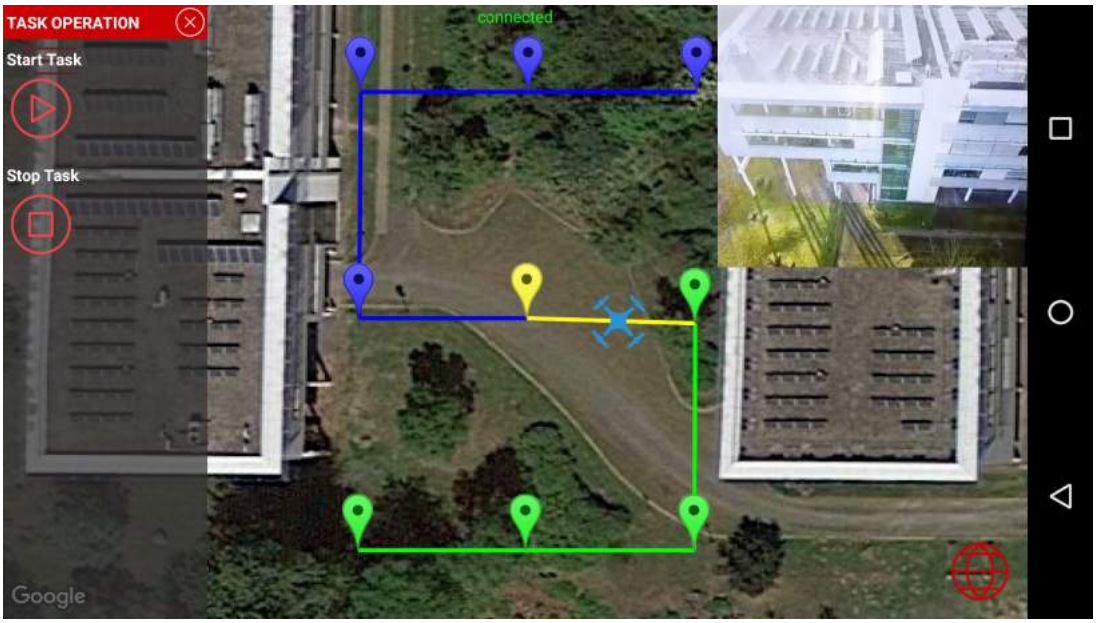

Abbildung 3.16: TRADR UAV App - Ausführung einer Explorationsaufgabe

Farbe	Bedeutung
blau	Wegabschnitt, der noch abgeflogen werden muss
gelb	Wegabschnitt, der aktuell abgeflogen wird
grün	Wegabschnitt, der erfolgreich abgeflogen ist
rot	Wegabschnitt, der nicht abgeflogen werden konnte

Tabelle 3.1: Bedeutung der Trajektorie-Farben

Farbe	Bedeutung
blau	Wegpunkt, der noch angeflogen werden muss
gelb	Wegpunkt, der als nächstes angeflogen wird
orange	Wegpunkt, an dem aktuell Aktionen durchgeführt werden
grün	Wegpunkt, an dem alle Aktionen erfolgreich durchgeführt wurden
rot	Wegpunkt, der entweder nicht erreicht werden konnte oder an dem nicht alle Aktionen durchgeführt werden konnten

Tabelle 3.2: Bedeutung der Wegpunkt-Farben

Abbildung 3.17: Pilot mit Fernsteuerung und Smartphone überwacht den autonomen Flug des UAVs

3.3 Kommunikationsschnittstelle zwischen Bridge und UAV App

Die Netzwerkkommunikation zwischen der auf dem TRADR Core ausgeführten TRADR UAV Bridge und verschiedenen UAV Systemen im Einsatzgebiet ist durch ein neu entwickeltes Netzwerkprotokoll realisiert, welches auf Anwendungsebene arbeitet. Eine Anforderung dieses TRADR UAV Interface ist eine allgemein und offene Spezifikation, die es ermöglicht, neben der in dieser Arbeit primär beschriebenen Integration von DJI Kompakt UAVs, auch beliebige UAVs anderer Hersteller zu integrieren.[59]

Im Folgenden wird die Kommunikationsschnittstelle beispielhaft an der Kommunikation zwischen der TRADR UAV Bridge und der TRADR UAV App auf dem Piloten-Smartphone beschrieben. Letztere steht exemplarisch für die UAV-seitigen Komponenten. Im Falle der Integration von DJI Kompakt UAVs ist dies die Smartphone App. Jedoch können über die hier beschriebene Schrittstelle auch andere Systeme angebunden werden wie beispielsweise das SRS SKIDER System[70] zur Integration der AscTec UAVs.

Dieser Abschnitt ist zusammen mit den im Anhang beigefügten Dokumenten als Spezifikation des Protokolls zu sehen.

3.3.1 Aufbau

Das Kommunikationsprotokoll besteht aus einem rahmenden Protokoll, dass den Anmeldevorgang beschreibt und drei eingeschlossenen Unterprotokollen, welche für die Umsetzung der einzelnen Funktionen nötig sind. Für jedes dieser Protokolle bestehen auf beiden Seiten der Kommunikation Zustandsautomaten.

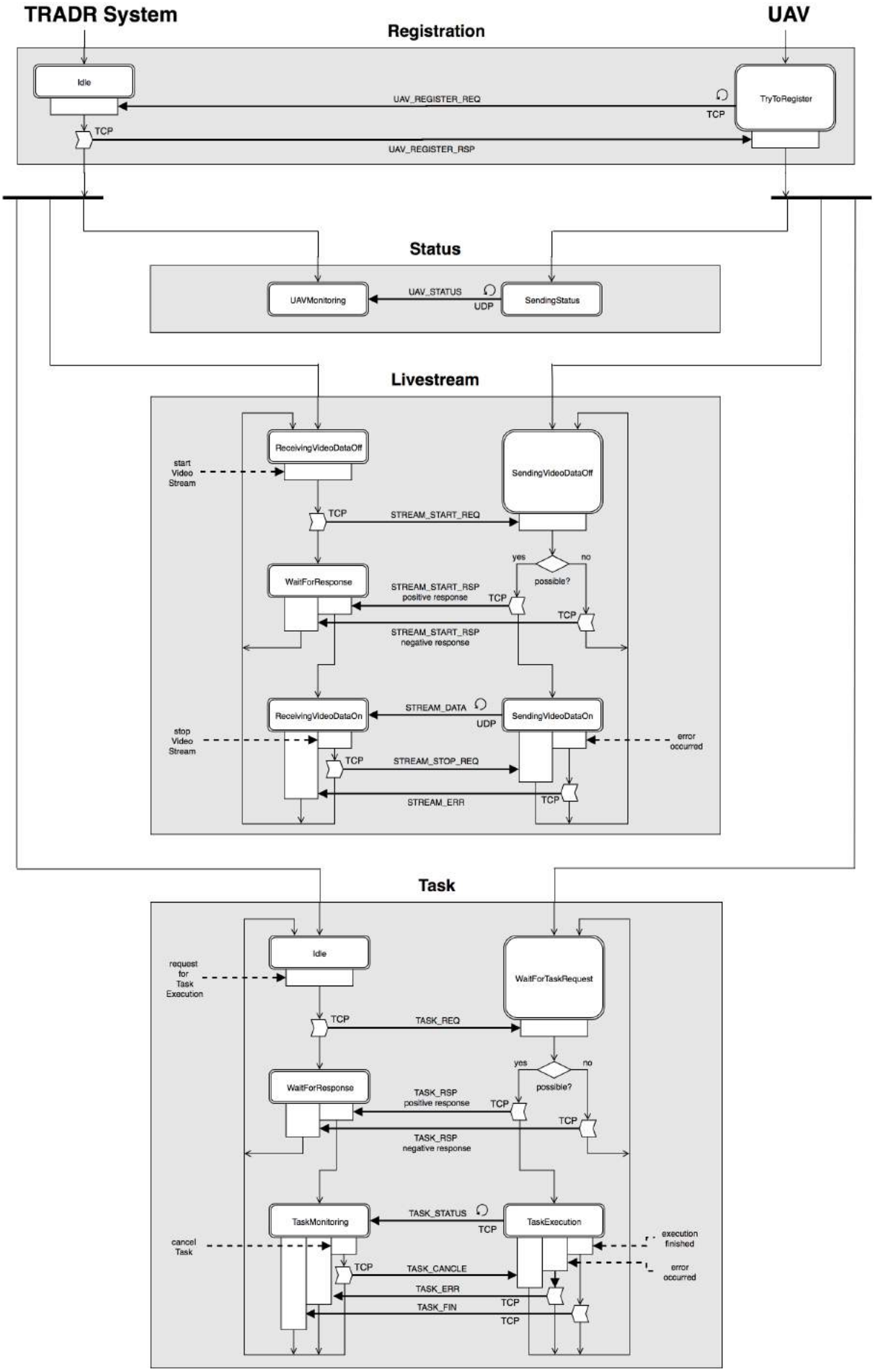

Abbildung 3.18: Interaktionsdiagramm des gesamten TRADR UAV Interface

3.3.2 Netzwerk

Die UAV Bridge fungiert als Server, die Smartphone App als Client. Die Bridge wartet bis ein Client an dem Well-Known-Port versucht, eine TCP-Verbindung[97] zu starten. Danach richtet sie eine UAV-Sitzung ein und verknüpft diese mit der neu geöffneten TCP-Verbindung. Über die Eigenschaften der TCP-Verbindung kennt die Bridge die IP-Adresse des Clients. Gehen von derselben IP-Adresse UDP-Datagramme[98] ein, können diese der UAV-Sitzung zugeordnet werden.

Die App versucht, sobald sie gestartet wird, so lange mit einem gewissen Intervall eine TCP-Verbindung zur Bridge aufzubauen, bis dies erfolgreich ist. In den Einstellungen der Smartphone App sind dazu IP-Adresse oder DNS-Name des Rechners, auf dem die Bridge ausgeführt wird, sowie der Well-Known-Port änderbar hinterlegt.

Wird auf einer der beiden Seiten der Prozess, der eine offene TCP-Verbindung besitzt, beendet oder stürzt ab, wird die TCP-Verbindung durch das Betriebssystem auf der jeweiligen Seite geschlossen, indem der FIN-Handshake[96] durchführt wird. Somit merkt die Gegenseite im Regelfall, wenn auf einer der beiden Seiten die Verbindung gewollt beendet wird oder durch einen Fehler abbricht.

Stellt die Bridge fest, dass die TCP-Verbindung zum Smartphone geschlossen wurde, beendet sie die UAV-Sitzung.

Stellt die App fest, dass die TCP-Verbindung zur Bridge von dieser geschlossen wurde, setzt sie den Zustandsautomaten für die Sitzung zurück und versucht erneut eine TCP-Verbindung aufzubauen.

Im Falle, dass die Netzwerk-Verbindung unterbrochen wird oder eine der beiden Seiten ausfällt, ohne dass das Betriebssystem in der Lage ist, den FIN-Handshake[96] durchzuführen, wird die Unterbrechung der Verbindung nicht bemerkt.

Um dieses Problem zu lösen, gibt es die Möglichkeit einer zyklischen Verbindungsüberprüfung mittels Heartbeat-Nachricht. In diesem Fall schickt die Smartphone App in einem bestimmten Intervall zyklisch eine Nachricht. Die Bridge kennt das Intervall und überprüft intern mit Hilfe eines Timers, ob eine Heartbeat-Nachricht des Clients eingegangen ist. Wenn der Heartbeat deutlich über der Intervall-Zeit ausbleibt, weiß die Bridge, dass die Verbindung zur App unterbrochen ist. [85]

Da bei Verwendung des TCP-Protokolls der Empfang jedes Datums bestätigt wird, indem der Empfänger automatisch eine Empfangsbestätigung zurücksendet, kann auch die App, die die Heartbeat-Nachricht versendet, einen möglichen Verbindungsabbruch erkennen, da der Empfang nicht bestätigt wird.

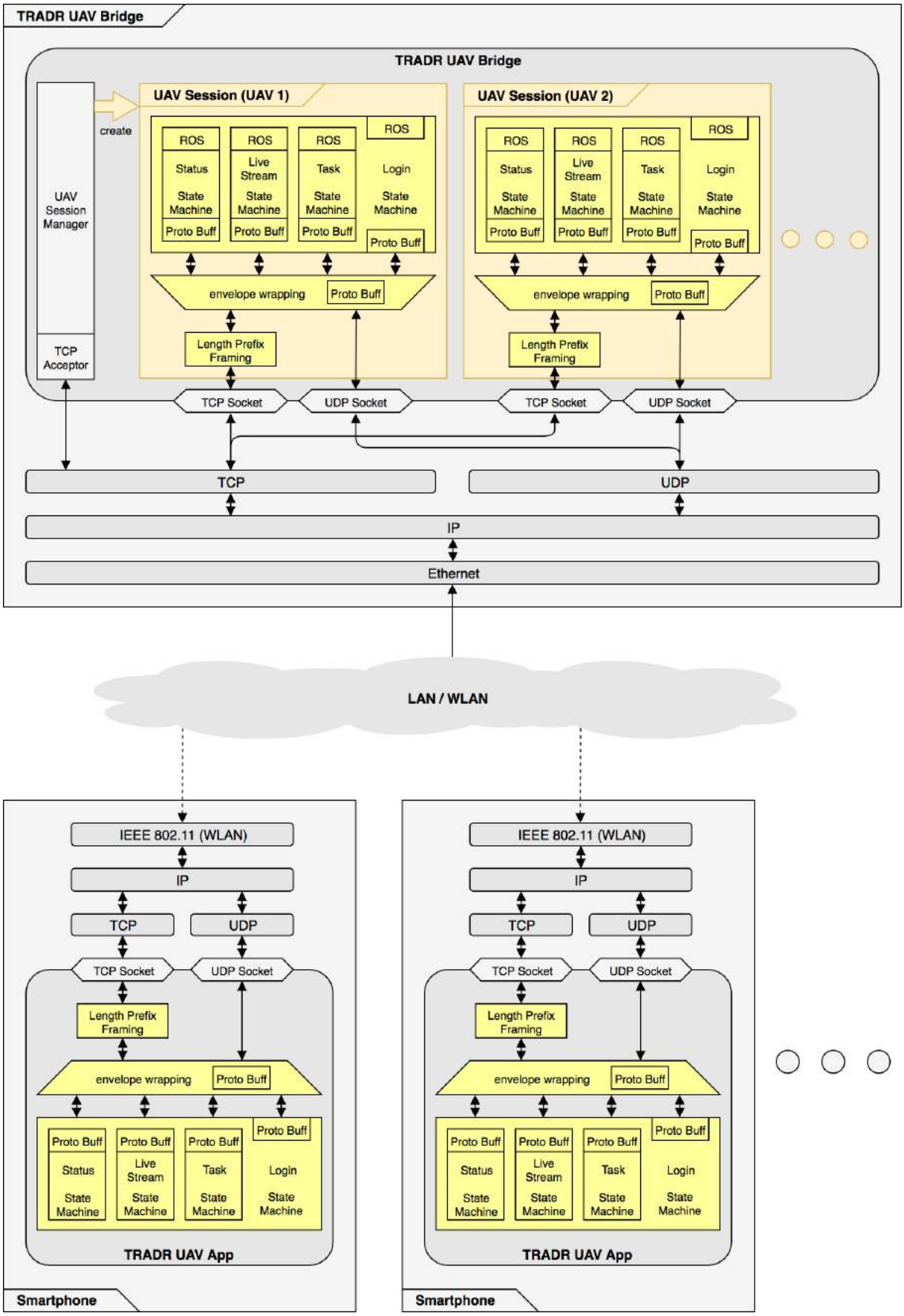

Abbildung 3.19: Softwareschichten des TRADR UAV Interface

Die schwierige Frage ist, welche Reaktion auf die zuletzt beschriebene Variante von Störungen im TRADR Anwendungskontext brauchbar ist. Eine Möglichkeit ist es, bei einem Verbindungsabbruch durch nicht empfangene Heartbeat-Nachrichten, genauso wie in der vorherigen Variante zu verfahren und die Sitzung zu beenden. Für den Fall, dass auf der Gegenseite tatsächlich ein sehr harter Ausfall des Systems stattgefunden hat, wäre dies sinnvoll, da hier eine Wiederaufnahme der alten Sitzung sehr unwahrscheinlich ist. Im anderen Falle, dass die Verbindung jedoch nur vorübergehend unterbrochen ist, beispielsweise bei Störungen des Netzwerkes, die im Falle von Funknetzwerken (WLAN) vorkommen, wäre jedoch das Beenden der Sitzung ungünstig, da mit der vorhandenen Sitzung weiter verfahren werden könnte, wenn die Störung wieder verschwindet.

TCP Length Prefix Framing

Unter dem Begriff „TCP Length Prefix Framing“ wird eine Zwischenschicht verstanden, die das Zeichen orientierte TCP Protokoll, welches nur eine Datenstrom-Schnittstelle anbietet, um eine Schnittstelle zum Versenden und Empfangen von Nachrichten erweitert. Eine Nachricht in diesem Kontext ist dabei ein zusammenhängender Datenstrom, der eine Informationseinheit darstellt, die nur komplett interpretiert sinnvolle Informationen liefert. TCP bietet von sich aus für den Empfänger keine Möglichkeit zu erkennen, an welcher Stelle eine Nachricht im empfangenen Datenstrom beginnt, wie lang diese ist und an welcher Stelle diese endet. Dies ist aber notwendig, um einen korrekten Transfer von Nachrichten-Objekten zu gewährleisten. [4][34][6][7][40]

Eine einfache Lösung für dieses Problem ist es, dass der Sender beim Senden einer Nachricht mittels des TCP-Protokolls die Länge dieser in einem für alle Kommunikationsteilnehmern bekannten Format der eigentlichen Nachricht voranstellt. Der Empfänger wechselt zwischen den Vorgängen des Lesens der Längen-Information und des Lesens der eigentlichen Nachricht. Die Längen-Information hat dabei eine feste bekannte Anzahl an Bytes, die immer zuerst gelesen bzw. empfangen werden. Die daraus ermittelte Länge gibt an, wie viele Bytes als nächstes gelesen bzw. empfangen werden müssen. Sind diese vollständig empfangen worden, ist die Nachricht komplett eingegangen und kann zur weiteren Auswertung zur Verfügung gestellt werden. Da festgelegt ist, dass vor der nächsten Nachricht wieder die Längenangabe versandt worden ist, interpretiert der Empfänger die nächsten empfangenen Bytes wieder als Länge der nächsten Nachricht. Durch dieses einfache Verfahren können Nachrichten-Objekte korrekt über die vom TCP-Protokoll angebotene Datenstrom-Schnittstelle übertragen werden. [4][34][6][7][40]

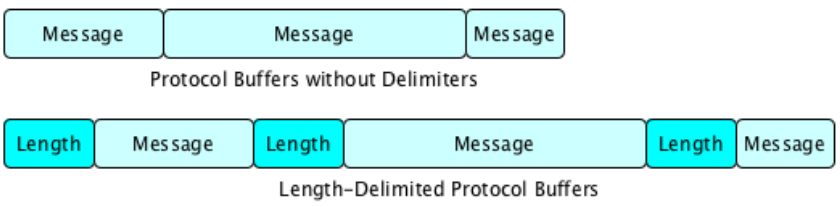

Abbildung 3.20: Length Prefix Framing (Quelle: [40])

Nachrichten Serialisierung

Die Daten einer zu versendenden Nachricht befinden sich vor dem Versandt in Form eines Nachrichten-Objektes oder einer Nachrichten-Struktur im Speichersegment des Prozesses. In dieser Form kann durch Lesen und Schreiben der Instanzvariablen eines solchen Nachrichten-Objektes die Nachricht aufgebaut oder interpretiert werden. Um eine solche Nachricht sicher zu verschicken, müssen die Daten des Nachrichten-Objektes serialisiert werden. Der Serialisierungsvorgang bringt die Daten in ein allen Kommunikationsteilnehmern bekanntes, einheitliches und plattformunabhängiges Format der Nachricht.

Die Serialisierung (Marshalling) stellt den Vorgang dar, der die Datenstruktur einer Nachricht, die meist in Form von Objekten (Objektorientierte Sprachen) oder Strukturen (C Structs) verwaltet wird, in eine eindeutige serielle Bytefolge wandelt. Die Deserialisierung (Demarshalling) dagegen wandelt diese serielle Bytefolge wieder zurück in das Nachrichten-Objekt. Damit dies möglich ist, müssen sich die im Objekt gespeicherten Informationen in der Bytefolge befinden und mit weiteren Informationen so strukturiert sein, dass sich das Objekt aus der Bytefolge wieder rekonstruieren lässt. [95][89]

Für die Serialisierung wird Google Protocol Buffer eingesetzt. Protocol Buffer (kurz: ProtoBuf) bezeichnet ein von Google entwickeltes binäres Serialisierungs-Format und eine Reihe von Software-Tools und Programm-Bibliotheken, um dieses zu nutzen. ProtoBuf ist quelloffen und wird von Google wie von einer großen freien Community weiter gepflegt. Dabei gilt es als sprach- und plattformunabhängig. Es werden eine sehr große Anzahl an Programmiersprachen durch passende Bibliotheken unterstützt (C++, Objective C, C#, Java, GO, Python, Ruby, PHP, Javascript). Zudem bietet ProtoBuf viele Features, welche die Abwärtskompatibilität zwischen Protokollversionen bei der Weiterentwicklung stark erleichtern. [33][60][32]

Umschlag-Nachrichten

Ein Kommunikationsprotokoll, wie das hier beschriebene, bei dem die Kommunikation mittels vieler verschiedener Nachrichtentypen abläuft, macht es für den Empfänger notwendig, den Typ der eingehenden Nachricht korrekt zu deserialisieren und das resultierende Nachrichten-Objekt an die dafür zuständige Instanz weiterzuleiten. Um dies zu gewährleisten, ist es bei Verwendung von Google ProtoBuf üblich jede übertragene Nachricht auf der Sender-Seite vor dem Übermitteln in eine Umschlag-Nachricht zu verpacken. Da ProtoBuf in dem serialisierten Datenstrom kenntlich macht, welchen Typ ein Feld besitzt, das mehrere Typen annehmen kann, können somit Nachrichten verschiedenen Typs korrekt unterschieden werden.

UAV Envelope Message		
Name	Datentyp	erforderlich
uavMessage	RegisterReq	ja
	Status	
	StreamStartRsp StreamData	
	TaskRsp TaskFeedback TaskFinalize	

Tabelle 3.3: UAV Envelope Message

Bridge Envelope Message		
Name	Datentyp	erforderlich
bridgeMessage	RegisterRsp	ja
	StreamStartReq StreamStopReq	
	TaskReq TaskCancel	

Tabelle 3.4: Bridge Envelope Message

3.3.3 Anmeldevorgang

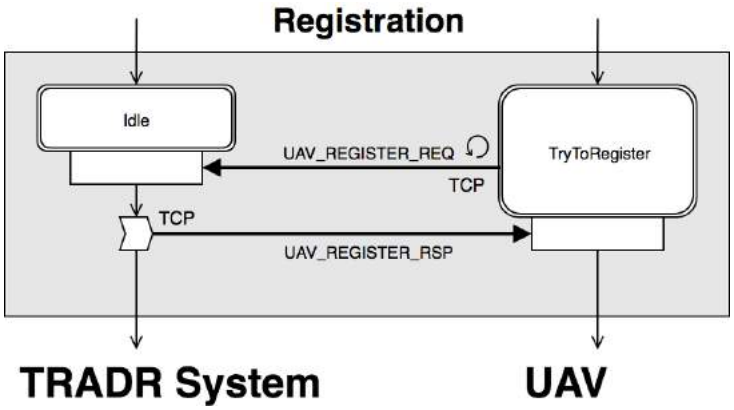

Abbildung 3.21: Registrierungsvorgang

Ist die TCP-Verbindung hergestellt und somit auch eine entsprechende UAV-Sitzung auf Seiten der Bridge eingerichtet, erfolgt als erster Schritt die Anmeldung der UAVs durch die Smartphone App am TRADR Systen. Dazu sendet die App mittels TCP eine Nachricht an die Bridge und teilt dieser den Wunsch mit, das UAV am TRADR System zu registrieren. Hierzu werden der Name des UAVs in der Mission und eine GUID (Globally Unique Identifier)[80], welche das UAV eindeutig identifiziert, übertragen. Im Weiteren werden die Model-Bezeichnung und die Versionsnummer der verwendeten Protokoll-Version übermittelt sowie die Eigenschaften und Fähigkeiten des UAVs. Letzteres soll den Softwarekomponenten im Leitstand ermöglichen, die Parameter, die der Benutzer einstellen kann, so einzuschränken, dass nur Dinge eingestellt werden können, die vom UAV auch leistbar sind.

Registration Request Message		
Name	Datentyp	erforderlich
name	string	ja
model	string	ja
guid	string	ja
apiVersion	string	ja

Tabelle 3.5: Registration Request Message

Als Reaktion auf die Registrierungsanfrage legt die Bridge korrespondierend zur Sitzung einen UAV-Eintrag im System an und teilt allen verbundenen Komponenten des Leitstandes mit, dass ein neues UAV verfügbar ist und stellt die Informationen zu diesem bereit. Durch den Eintrag erhält das UAV eine ID innerhalb der TRADR Mission. Diese ID wird mittels einer Bestätigungsnachricht der Smartphone App mitgeteilt.

Registration Response Message		
Name	Datentyp	erforderlich
registerACK	boolean	ja
uavID	uint32	nein

Tabelle 3.6: Registration Response Message

Bei den Nachrichten für den Anmeldevorgang muss sichergestellt sein, dass diese die Gegenseite erreichen, da z.B. die Antwort-Nachricht nur einmal gesandt wird. Käme diese nicht an, würde der Anmeldevorgang in einer Deadlock-Verklemmung stagnieren und könnte nicht korrekt abgeschlossen werden. Aus diesem Grund müssen alle Nachrichten des Anmeldevorgangs auf Transportebene mittels des TCP Protokolls übertragen werden.

3.3.4 Übertragung der Telemetriedaten

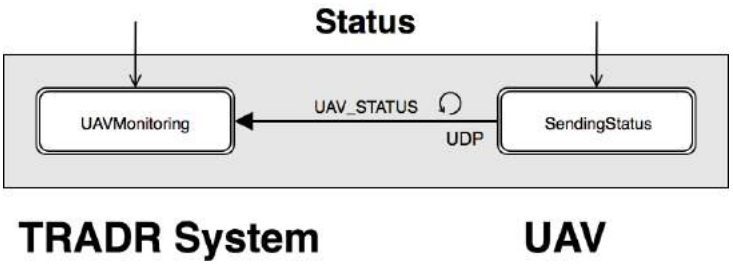

Abbildung 3.22: Übertragung der Status-Nachrichten

Zur Übermittlung von Telemetriedaten und weiteren Statusinformationen des UAVs kann die App, sobald der Anmeldevorgang abgeschlossen ist, Status-Nachrichten an den Leitstand schicken. Die Nachrichten können entweder synchron in einem von der App frei wählbaren Intervall übertragen werden oder asynchron, wenn die Werte sich geändert haben und aktualisiert werden sollen.

Der Inhalt einer Status-Nachricht setzt sich aus den Telemetrie-Daten UAV-Position, UAV-Orientierung, den UAV-Geschwindigkeiten, dem Batteriestatus und einem Zeitstempel zusammen.

Status Message		
Name	Datentyp	erforderlich
timestamp	uint32	ja
batteryStatus	float	nein
uavPose	Pose	nein
uavVelocity	Velocity	nein

Tabelle 3.7: Status Message

Da eine Status-Nachricht für sich eine abgeschlossene Informationseinheit darstellt, führt das Fehlen einzelner Nachrichten in der Sequenz an wiederkehrenden Status-Nachrichten zu keinen Problemen. Daher werden sie zur Reduktion der Netzwerkauslastung auf Transportebene mittels UDP übertragen. Die bei UDP nicht sicher gewährleistete Empfangsreihenfolge hat im TRADR Kontext keine Bedeutung, da es in der Netzwerktopologie nur eindeutige Routen gibt, zudem könnte eine fehlerhafte Ordnung der eingehenden Nachrichten durch den Zeitstempel erkannt werden.

Telemetriedaten

Pose Message					
Name	Datentyp	erforderlich	Beschreibung	Einheit	Wertebereich
latitude	float	nein	WGS84 Latitude	Degree	[0.0; 360.0)
			NaN: unknown		
longitude	float	nein	WGS84 Longitude	Degree	[-90.0; 90.0]
			NaN: unknown		
altitude	float	nein	AMSL Altitude	m	[-10.0; +100.0]
			NaN: unknown		
roll	float	nein	Absolute Roll Angle	Degree	[-180.0; + 180.0)
			0.0: horizontal +90.0: perpendicular with right wing aligned to ground -90.0: perpendicular with left wing aligned to ground -180.0: horizontal upside down NaN: unknown		
pitch	float	nein	Absolute Pitch Angle	Degree	[-180.0; + 180.0)
			0.0: aligned to horizon +90.0: aligned to space -90.0: aligned to ground -180.0: aligned to horizon backwards NaN: unknown		
yaw	float	nein	Absolute Yaw Angle	Degree	[-180.0; + 180.0)
			0.0: aligned to north +90.0: aligned to east -90.0: aligned to west -180.0: aligned to south NaN: unknown		

Tabelle 3.8: Pose Message

Velocity Message			
Name	Datentyp	erforderlich	Beschreibung
velX	float	nein	translational velocity of uav's horizontal forward/backward movement
			Einheit: m/s
			Wertebereich: [-100.0; 100.0]
			Richtung: negativ: backward movement positiv: forward movement
			Werte: NaN: unknown
velY	float	nein	translational velocity of uav's horizontal left/right movement
			Einheit: m/s
			Wertebereich: [-100.0; 100.0]
			Richtung: negativ: left movement positiv: right movement
			Werte: NaN: unknown
velZ	float	nein	translational velocity of uav's vertical up/down movement
			Einheit: m/s
			Wertebereich: [-100.0; 100.0]
			Richtung: negativ: down movement positiv: up movement
			Werte: NaN: unknown
velRoll	float	nein	rotational velocity of the uav about it's roll axis
			Einheit: Degree/s
			Wertebereich: [-2160.0; +2160.0]
			Richtung: negativ: left roll positiv: right roll
			Werte: NaN: unknown
velPitch	float	nein	rotational velocity of the uav about it's pitch axis
			Einheit: Degree/s
			Wertebereich: [-2160.0; +2160.0]
			Richtung: negativ: down pitch positiv: up pitch
			Werte: NaN: unknown
velYaw	float	nein	rotational velocity of the uav about it's yaw axis
			Einheit: Degree/s
			Wertebereich: [-2160.0; +2160.0]
			Richtung: negativ: left swivel positiv: right swivel
			Werte: NaN: unknown

Tabelle 3.9: Velocity Message

3.3.5 Übertragung des Livebildes

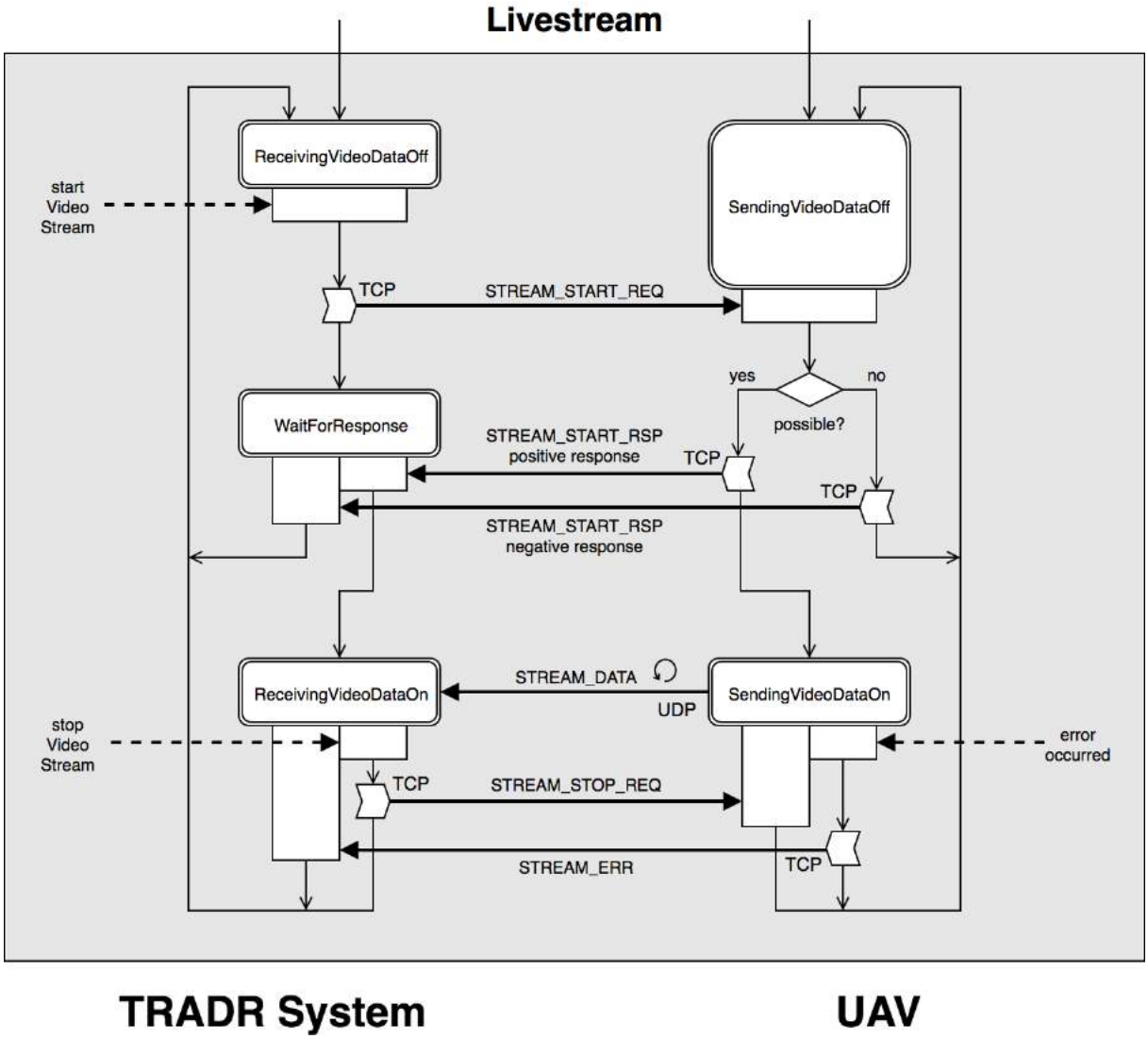

Abbildung 3.23: Übertragung des Livebildes

Das Livebild wird einerseits aufgrund der Reduzierung unnötiger Netzwerkauslastung und andererseits aufgrund der in einem späteren Abschnitt beschriebenen I-Frame-Injection nicht automatisch dauerhaft übertragen.

Stattdessen stellt der Operator im Leitstand eine Anfrage an die Smartphone App, das Livebild einer UAV Kamera, des von der App kontrollierten UAVs, in den Leitstand zu übertragen. Für diese Anfrage sendet die Bridge eine Anfrage-Nachricht (Stream Start Request) an die App. Diese Anfrage enthält die ID der Kamera, von der das Livebild übertragen werden soll. In der Regel befindet sich nur eine Kamera an Board des UAVs, diese besitzt die ID 0. Hier ist die Möglichkeit geschaffen, auch UAVs über diese Schnittstelle anzubinden, die mehrere Kameras besitzen. Im Weiteren kann das Livebild in verschiedenen Auflösungen und Formaten angefordert werden. Dies geht natürlich nur, wenn diese von der UAV-Kamera unterstützt werden. Die Felder für Auflösung und Format sind optional. Werden diese nicht belegt, sendet das UAV das Livebild in seinem Standard- oder in dem zuletzt eingestellten Format.

Stream Start Request Message		
Name	Datentyp	erforderlich
cameraID	uint32	ja
streamFormat	VideoFormat	nein

Tabelle 3.10: Stream Start Request Message

Die Anfrage wird von der App mit einer Antwort-Nachricht (Stream Start Response) beantwortet.

Ist die Übertragung des Livestreams in diesem Moment nicht möglich, schickt die Smartphone App eine negative Antwortnachricht (negative Stream Start Response), um der Bridge und der dahinter liegenden Livebild-Anzeige-Software (UAV FPV) dies zu signalisieren. Andernfalls bestätigt die Smartphone App die Anfrage mit einer positiven Antwortnachricht (positive Stream Start Response) und signalisiert dadurch der Bridge, dass sie jetzt beginnt Livestream-Datenpakete zu senden. Die Bridge ist dadurch informiert, dass nun in Folge Datenpakete mit Streaminhalt eingehen werden und teilt dies der Anzeige-Software mit, um den Videodecoder für die Übertragung zu initialisieren.

Die Antwort-Nachricht enthält das gesamte Start-I-Frame, wenn I-Frame-Injection verwendet wird und weitere wichtige Parameter wie die Auflösung, das verwendete Container-Format, usw., die notwendig sind, um die Daten des komprimierten Videostreams korrekt zu dekodieren.

Stream Start Response Message		
Name	Datentyp	erforderlich
cameraID	uint32	ja
startACK	boolean	ja
streamFormat	VideoFormat	nein
iFrameInjection	boolean	ja
iFrameData	byte array	nein

Tabelle 3.11: Stream Start Response Message

Während die Videoübertragung läuft, befindet sich der Zustandsautomat der Smartphone App im Sendezustand. Der Automat der UAV-Sitzung der Bridge ist dagegen im Empfangszustand. Solange dies der Fall ist, werden die Datenpakete mit den kodierten Videobild-Inhalten von der App zur Bridge übertragen. Die Daten werden in Form einer Daten-Nachricht (Stream Data) übermittelt. Diese Nachricht enthält nicht nur die Daten des Videostream-Pakets, sondern auch die Telemetriedaten von UAV und Aufnahmegerät des Zeitpunktes, der mit dem Aufnahmezeitpunkt des Inhaltes des Videostream-Pakets korrespondiert.

Stream Data Message		
Name	Datentyp	erforderlich
cameraID	uint32	ja
data	byte array	ja
size	uint32	ja
uavPose	Pose	nein
cameraPose	Pose	nein

Tabelle 3.12: Stream Data Message

In der Weise, in der der Operator den Livestream einer UAV Kamera anfordern kann, kann er die Übertragung desselben zum Leitstand wieder stoppen. Hierzu sendet die Bridge eine Stopp-Nachricht (Stream Stop Request) an die App und der Automat der UAV-Sitzung der Bridge wechselt in den Zustand „ReceivingVideoDataOff“. Der Zustandsautomat der Smartphone App wechselt bei Empfang der Stopp-Nachricht in den Zustand „SendingVideoDataOff“ und das Senden der Daten-Nachrichten wird eingestellt. In diesem Zustand ist die App erneut bereit, eine neue Start-Anfrage von der Bridge entgegenzunehmen.

Stream Stop Request Message		
Name	Datentyp	erforderlich
cameraID	uint32	ja
stopCode	uint32	nein

Tabelle 3.13: Stream Stop Request Message

Es ist möglich, dass aus technischen Gründen keine Videodaten mehr an die Bridge gesandt werden können. In diesem Fall schickt die App eine Fehler-Nachricht (Stream Error) an die Bridge. Die Bridge und die dahinter arbeitende Anzeigesoftware wissen durch den Erhalt der Fehler-Nachricht, dass keine Daten-Nachrichten mehr eingehen werden. Somit kann die Bridge ihren zur UAV-Sitzung gehörenden Zustandsautomaten auf den Ausgangszustand „ReceivingVideoDataOff“ zurücksetzen und der Decoder kann beendet werden.

Stream Error Message		
Name	Datentyp	erforderlich
cameraID	uint32	ja
errorCode	uint32	nein

Tabelle 3.14: Stream Error Message

Alle Nachrichten, die in diesem Abschnitt beschrieben werden, ausgenommen der Daten-Nachrichten (Stream Data), werden auf Transportebene mittels des TCP-Protokolls übertragen. Abgesehen von den Daten-Nachrichten handelt es sich bei allen Nachrichten um signalgebende Kommunikation. Bei dieser Form der Kommunikation wird jede Nachricht nur einmal versandt und es muss sichergestellt sein, dass diese auf der Gegenseite eingeht und zur erwarteten Reaktion führt. Der verlustfreie Transport der Nachrichten wird durch das TCP-Protokoll sichergestellt.

Die Daten-Nachrichten werden im Gegensatz dazu auf Transportebene mittels des UDP-Protokolls übertragen, da diese in hoher Frequenz versandt werden und der Verlust einzelner Nachrichten zwar zu Bildartefakten führt, jedoch kein Problem für die Stabilität der Kommunikation darstellt. Hingegen würde der Overhead bei einem Transport durch das TCP-Protokoll in Kombination mit oft auftretenden Paketverlusten bei Drahtlosverbindungen zu einem Aufschaukeln von Wiederholungsanforderungen führen, die in der Bilanz die Kommunikation verschlechtern oder sogar unmöglich machen können, sollte es zu Pufferüberläufen kommen.

3.3.6 Übertragung von Explorationsaufgaben

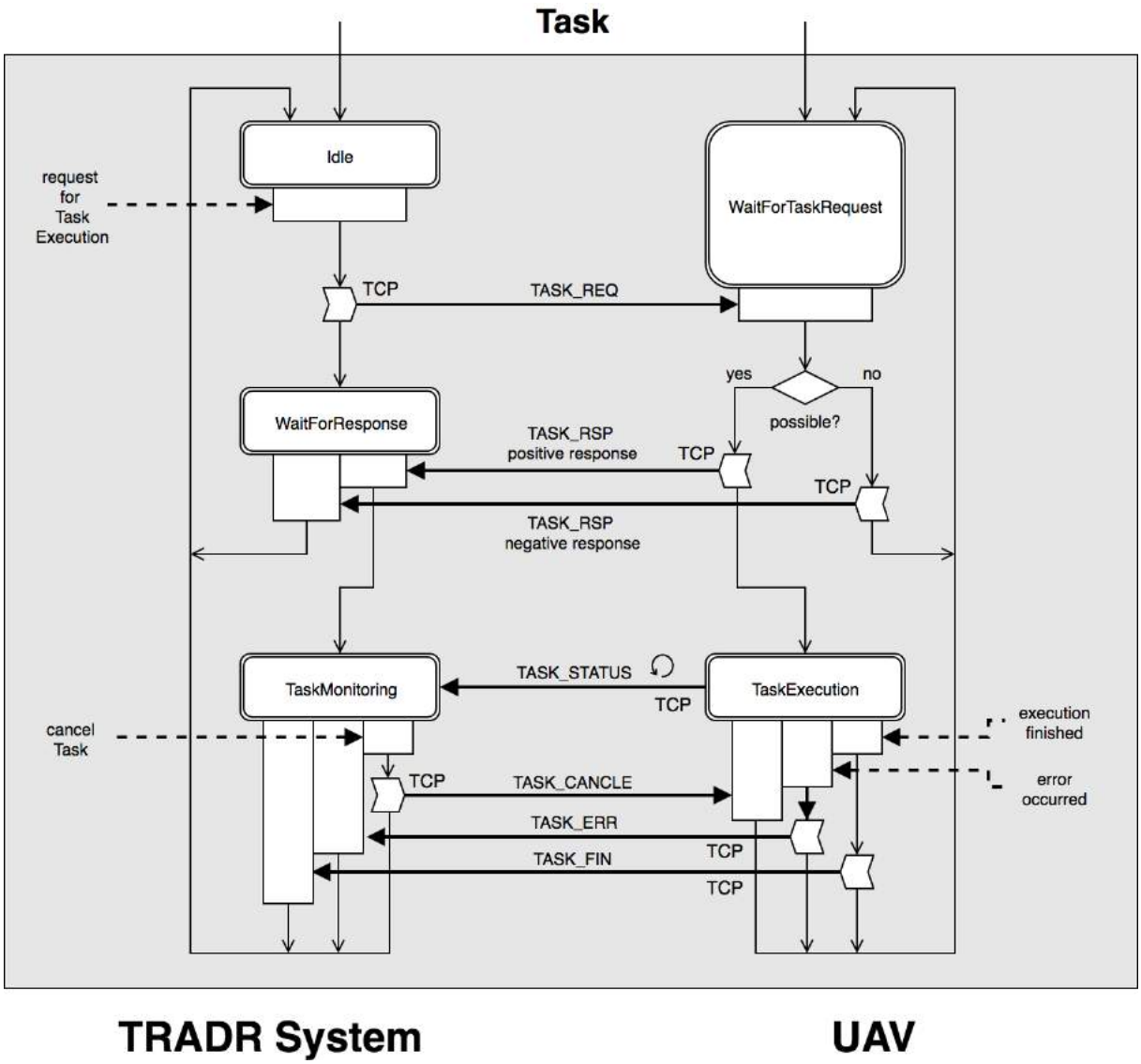

Abbildung 3.24: Ausführung einer externen Explorationsaufgabe

Eine durch den Operator im Leitstand festgelegte Explorationsafgabe kann zum Smartphone des Piloten übertragen werden, damit dieser die Aufgabe durch das UAV ausführen lassen kann. Dazu sendet die Bridge eine Anfrage-Nachricht (Task Request) an die App. In dieser Anfrage wird die Explorationsaufgabe direkt mitübertragen. Vorgänge wie diese werden durch die Sprechfunkverbindung zwischen Pilot und Operator zusätzlich abgestimmt.

Task Request Message		
Name	Datentyp	erforderlich
task	Task	ja

Tabelle 3.15: Task Request Message

Führt das UAV gerade eine Aufgabe aus oder ist der Pilot nicht bereit, eine solche entgegenzunehmen, wird die Anfrage direkt mit einer negativen Antwort (negative Task Response) abgelehnt.

Ist das UAV und der Pilot prinzipiell bereit, die Anfrage annehmen zu können, wird dem Piloten dies auf dem Display mitgeteilt. Er kann zu diesem Zeitpunkt die Anfrage direkt ablehnen oder sich die Flugbahn der Aufgabe anzeigen lassen. Nach der Evaluierung dieser kann er die Aufgabe zur Ausführung bringen oder sie ablehnen. In jedem Fall, indem es zu einer Ablehnung der Ausführung kommt, wird, wie schon beschrieben, eine negative Antwort (negative Task Response) von der App zur Bridge übertragen. Nur beim Starten der Ausführung durch den Piloten wird die Anfrage mit einer positiven Antwort (positive Task Response) beantwortet.

Aus der Beschreibung der Interaktionen des Piloten ist zu erkennen, dass zwischen dem Eingehen der Anfrage und der Beantwortung dieser mittels einer Antwortnachricht eine beliebig große Zeitdauer existieren kann. Solange die Anfrage weder angenommen noch abgelehnt wird, behalten die Zustandsautomaten auf beiden Seiten ihren Zustand. Es kann erst dann für die UAV-Sitzung eine neue Anfrage gestellt werden, wenn die bereits gestellte Anfrage in einer Form beantwortet wurde.

Task Response Message		
Name	Datentyp	erforderlich
taskACK	boolean	ja

Tabelle 3.16: Task Response Message

Während die Aufgabe ausgeführt wird und das UAV fliegt, befindet sich der Zustandsautomat der Smartphone App im Zustand der Ausführung ("Task Execution"). Auf Seiten der Bridge ist der Automat im Überwachungs-Zustand ("Task Monitoring"). In dieser Zeit kann die App beliebige Status-Nachrichten übertragen, die den Status der Ausführung der Aufgabe aktualisieren. Somit ist es möglich, dass die hinter der Bridge liegende Planungs- und Überwachungssoftware den Status der im Moment ausgeführten Aufgabe darstellt.

Ergebnisse der Exploration können als Anhang einer Status-Nachricht in den Leitstand übertragen werden, beispielsweise die Daten eines gemachten Fotos noch während sich das UAV in der Luft befindet.

Die Übertragung der aufgenommenen Bilder ist aufgrund von Zeitmangel und verlängerter Flugzeiten nicht implementiert und getestet worden.

Task Status Changed Message		
Name	Datentyp	erforderlich
status	TaskStatus WaypointStatus ActionStatus	ja

Tabelle 3.17: Task Status Changed Message

Task Status Message		
Name	Datentyp	erforderlich
taskID	uint32	ja
status	enum TaskStatus	ja

Tabelle 3.18: Task Status Message

Waypoint Status Message		
Name	Datentyp	erforderlich
taskID	uint32	ja
waypointID	uint32	ja
status	enum WaypointStatus	ja

Tabelle 3.19: Waypoint Status Message

Action Status Message		
Name	Datentyp	erforderlich
taskID	uint32	ja
waypointID	uint32	ja
actionID	uint32	ja
status	enum ActionStatus	ja

Tabelle 3.20: Action Status Message

Eine laufende Ausführung kann durch den Operator am Leitstand abgebrochen werden. Dies kann z.B. dann sinnvoll sein, wenn eine gestartete Exploration in der Form, in der sie begonnen wurde, doch nicht benötigt wird, oder das UAV dringend für eine andere Aufgabe gebraucht wird. Bricht der Operator im Leitstand die sich in Ausführung befindliche Aufgabe ab, sendet die Bridge eine Abbruch-Nachricht (Task Cancel). Dem Piloten wird in diesem Fall auf dem Display mitgeteilt, dass die Aufgabe vom Operator abgebrochen worden ist und die Ausführung wird gestoppt. Der Pilot kann das UAV dann teleoperiert steuern, zurückholen und landen oder für die nächste Aufgabe präparieren. Zur Abstimmung eines Abbruchvorgangs tauschen sich Pilot und Operator zusätzlich über Sprechfunk aus.

Task Cancel Message		
Name	Datentyp	erforderlich
cancelCode	uint32	nein

Tabelle 3.21: Task Cancel Message

Es kann Gründe geben, dass entweder der Pilot die in Ausführung befindliche Aufgabe abbrechen muss oder das UAV aufgrund eines Fehlers die Ausführung abbricht. In diesem Fall schickt die Smartphone App der Bridge eine Fehler-Nachricht, dass die Ausführung abgebrochen worden ist. Durch einen Fehler Code kann dem Operator mitgeteilt werden, warum der Abbruch zustande kam. Der Fehler Code dient dazu, dass in der Planungs- und Überwachungssoftware des Operators der Abbruch korrekt signalisiert wird und zwischen verschiedenen Varianten wie z.B. einem Abbruch durch den Piloten, einen Fehler des UAVs oder der Smartphone App unterschieden werden kann. Genaue Informationen über den Abbruch bzw. Fehler tauschen Pilot und Operator über Sprechfunk aus.

Task Error Message		
Name	Datentyp	erforderlich
errorCode	uint32	nein

Tabelle 3.22: Task Error Message

Ist die Aufgabe erfolgreich ausgeführt, sendet die App eine Fertigstellungs-Nachricht (Task Finish). Der Zustandsautomat der Smartphone App wechselt nach dem Versenden der Fertigstellungs-Nachricht (Task Finish) wieder in den Zustand „wait for Task request“ und ist somit erneut bereit, die Anfrage für die nächste Aufgabe in Empfang zu nehmen. Durch den Erhalt der Fertigstellungs-Nachricht (Task Finish) wechselt der Zustandsautomat der Bridge vom „Task Monitoring“ in den „Idel“ Zustand. Dem Operator wird in der Planungs- und Überwachungssoftware die Fertigstellung der Aufgabe signalisiert. Gleichzeitig ist das UAV für die Komponenten im Leitstand wieder als „bereit für neue Aufgaben“ erkennbar.

Task Finish Message		
Name	Datentyp	erforderlich
successful	boolean	ja

Tabelle 3.23: Task Finish Message

Sobald eine Anfrage zur Ausführung einer Explorationsaufgabe an ein UAV durch einen Operator im Leitstand geschickt worden ist, ist dieses UAV für weitere Anfragen gesperrt. Erst wenn die gestellte Anfrage entweder abgelehnt oder die daraus resultierende Ausführung der Aufgabe erfolgreich beendet oder abgebrochen worden ist, ist das entsprechende UAV wieder freigegeben für weitere Anfragen.

Alle in diesem Abschnitt genannten Nachrichten werden auf Transportebene mittels des TCP-Protokolls übertragen, das eine verlustfreie Übertragung sicherstellt, da bei der in diesem Abschnitt vorgestellten Kommunikation kein Verlust auftreten darf. Bei den hier beschriebenen Nachrichten werden der jeweiligen Gegenseite Ereignisse signalisiert, die zum Teil zu Zustandsänderungen in den Automaten führen. Da diese Signal-Nachrichten für jedes Ereignis nur einmal versandt werden, würde der Verlust einer solchen Nachricht zu Inkonsistenzen führen, die sich im ungünstigen Fall in Deadlock-Verklemmungen zeigen würden. Aus diesem Grund muss die verlustfreie Übertragung sichergestellt werden.

Format einer Explorationsaufgabe

Eine Explorationsaufgabe ist technisch repräsentiert durch eine Liste, die eine Folge von Wegpunkten enthält, die nacheinander vom UAV angeflogen werden. Die Wegpunkte sind durch ihre Position und Höhe angegeben. Zudem ist zu jedem Wegpunkt festgelegt, welche Orientierung das UAV an diesem einnehmen soll. Es ist möglich, von dem UAV an jedem Wegpunkt mehrere Fotos aufnehmen zu lassen. Dazu gehört zu jedem Wegpunkt eine weitere Liste, in der eine Folge von Aktionen festgelegt ist, die das UAV nacheinander ausführen soll. Als Aktionen ist verschiedenes denkbar, beispielsweise die Drehung um die Gier-Achse des UAVs, das Schwenken der Kamera mittels dem steuerbaren Gimbal, das Aufnehmen eines Fotos mit bestimmten Aufnahme-Parametern oder das Starten und Stoppen einer Video-Aufnahme.

Task		
Name	Datentyp	erforderlich
taskID	uint32	ja
waypointList	Waypoint array	ja

Tabelle 3.24: Task

Waypoint		
Name	Datentyp	erforderlich
waypointID	uint32	ja
uavPose	Pose	ja
actionList	Action array	ja
velocity	float	nein

Tabelle 3.25: Waypoint

Action		
Name	Datentyp	erforderlich
actionID	uint32	ja
action	FotoAction StartVideoRecordingAction StopVideoRecordingAction RotationAction MoveGimbalAction	ja

Tabelle 3.26: Action

Umgesetzt ist nur die Foto-Aktion, jedoch sind die Strukturen in den Softwarekomponenten so designed, dass weitere Aktionstypen auf einfache Weise implementiert und ergänzt werden können.

Die Foto-Aktion enthält die Richtung, in der das Foto aufgenommen werden soll, in absoluten Werten. Das UAV dreht sich und/oder die Kamera vor der Aufnahme so, dass die Ausrichtung den festgelegten Werten entspricht. Erst danach wird die Aufnahme ausgelöst. Je nach UAV Modell kann die Ausrichtung der Kamera alleine durch Konfiguration des Gimbals erreicht werden, ohne dass eine Drehung des gesamten UAVs um seine Gier-Achse nötig ist, oder durch eine Kombination aus Gimbal-Konfiguration und Drehung des gesamten UAVs. Im Falle der DJI Kompakt UAVs (Phantom, Mavic) wird nur der Nick-Winkel der Kameraausrichtung mittels des Gimbals eingestellt. Der Gier-Winkel wird durch die Drehung des gesamten UAVs in die entsprechende Richtung umgesetzt. Der Roll-Winkel wird durch die automatische Stabilisierung immer so eingestellt, dass der Horizont gerade ist. Er kann bei den DJI UAVs nicht frei gewählt werden. Diese Zusammenhänge werden durch die Software auf der UAV Seite (App) umgesetzt und müssen beim Erstellen des Foto-Aktions-Objektes auf Seiten des Leitstandes (Bridge) nicht berücksichtigt werden.

Zusätzlich zu der Kameraausrichtung enthält die Foto-Aktion Aufnahmeparameter, beispielsweise Blendenöffnung, ISO-Zahl und Fokus.

Foto Action		
Name	Datentyp	erforderlich
camSettings	CameraSettings	nein
camPose	Pose	ja

Tabelle 3.27: Foto Action

3.4 Kommunikationsschnittstelle zwischen Bridge und OCU-Komponenten

Zwischen der TRADR UAV Bridge, die auf dem TRADR Core und den Planungs- und Anzeigeprogrammen, die auf den TRADR Display Systemen (TDS) ausgeführt werden, ist die Kommunikation durch das Robotic Operating System (ROS)[28] realisiert.

3.4.1 ROS Grundlagen

Die Software ROS (Robotic Operating System)[28] bietet unter anderem eine Kommunikations-Middelware, die auf eine einfache standardisierte Art die Kommunikation zwischen Prozessen auf demselben Rechner oder über Rechnergrenzen hinweg ermöglicht. Die Softwarekomponenten, welche die ROS-Bibliotheken nutzen, erscheinen zur Laufzeit als ROS-Knoten, die mit einem zentralen Prozess kommunizieren, der als eine Art Naming-Server die Organisation der Kommunikation regelt. Diese Komponente, genannt ROS Core, wird auf dem TRADR Core ausgeführt. Innerhalb von ROS ist es den Knoten möglich, in baumartigen geschachtelten Namensräumen ROS-Topics und ROS-Services anzubieten. Sie dienen der Kommunikation zwischen Knoten. ROS-Knoten können Topics und Services, die von anderen ROS-Knoten angeboten werden, nutzen, indem sie sich in fremde Topics einschreiben und Services aufrufen. Knoten können unter einem angelegten Topic eine Nachricht veröffentlichen. Das ROS System leitet diese Nachricht automatisch an alle Knoten weiter, die sich für das Topic, unter dem die Nachricht veröffentlicht wurde, eingeschrieben haben. Der ROS Core sorgt in diesem Fall nur für den Prozess des Topic Anlegens und der Einschreibung, sodass alle Knoten wissen, welche Knoten für welche Topics eingetragen sind. Der Prozess, der eine Nachricht veröffentlicht, schickt technisch gesehen selber die Nachricht direkt an alle auf das Topic eingeschriebenen Prozesse. Die Kommunikation läuft somit direkt und dezentral zwischen den Knoten. Diese technischen Vorgänge sind innerhalb der ROS Bibliothek gekapselt und müssen nicht vom Entwickler realisiert werden.[27]

Die Kommunikation innerhalb eines ROS-Systems läuft über den Netzwerkstapel. Daher ist im Prinzip keine Unterscheidung nötig, ob Knoten auf demselben Rechner oder auf einem entfernten Rechner im Netzwerk ausgeführt werden. Auf Transportebene wird das TCP-Protokoll eingesetzt. Das ROS-Basissystem bietet an dieser Stelle keine weiteren Möglichkeiten. Da die Kommunikation als nicht leichtgewichtig angesehen werden kann, eignet sie sich auf leistungsstarken Rechnern, die in einem stabil betriebenen Kabel-Netzwerk miteinander verbunden sind. Gibt es Performance-Probleme bei der Interprozesskommunikation auf demselben System, kann mit ROS-Nodelets Abhilfe geschaffen werden[3]. Dabei wird zur Kommunikation nicht der Netzwerkstapel benutzt sondern von mehreren Knoten gemeinsam genutzte Speichersegmente. Mehrere Nodelets laufen dabei als einzelne Threads eines gemeinsamen Prozesses, dem Nodelet-Manager. Zusammenarbeitende Nodelets lassen sich selbstverständlich nur auf demselben Rechner betreiben. Um die ROS-Kommunikation zwischen entfernten Knoten auf verschiedenen

Rechnern auch bei schlechten Funkverbindungen zu nutzen, gibt es externe Erweiterungen, die so etwas ermöglichen. Zu nennen ist hier das bei den „ROS Nutzern“ gut bekannte und oft verwendete Nimbro-Network-Pakage der Universität Bonn[78]. Das Paket ermöglicht den Betrieb mehrerer ROS Cores auf verschiedenen Rechnern und einer losen Kopplung zwischen den Knoten mittels ROS-Nachrichten-Transport über sich selbst wiederaufbauenden TCP Verbinden oder verbindungsloser UDP Kommunikation.

Während sich über die Topics ein asynchroner Nachrichtenaustausch zwischen Prozessen realisieren lässt, stellen die Services eine synchrone Schnittstelle für Remote Procedure Calls (RPC) zur Verfügung. Ein Knoten kann den angebotenen Service eines anderen Knoten aufrufen. Beim Aufruf wird dem anbietenden Knoten eine Service-Anfrage-Nachricht mit Parametern zugesandt. Der Knoten führt die Anfrage aus und beantwortet diese mit einer Antwort-Nachricht, indem sich die Ergebnisdaten befinden.

Die Typen bzw. Formate der Nachrichten werden in ROS ähnlich wie bei Google Protocol Buffers mittels einer Interface Definition Language (IDL) festgelegt. Ein eigener ROS-Nachrichten-Compiler wandelt die Definitionen in C++ Klassen, die mit Hilfe der ROS-Bibilothek bei der Entwicklung von ROS-Knoten genutzt werden können.

3.4.2 Aufbau der Schnittstelle

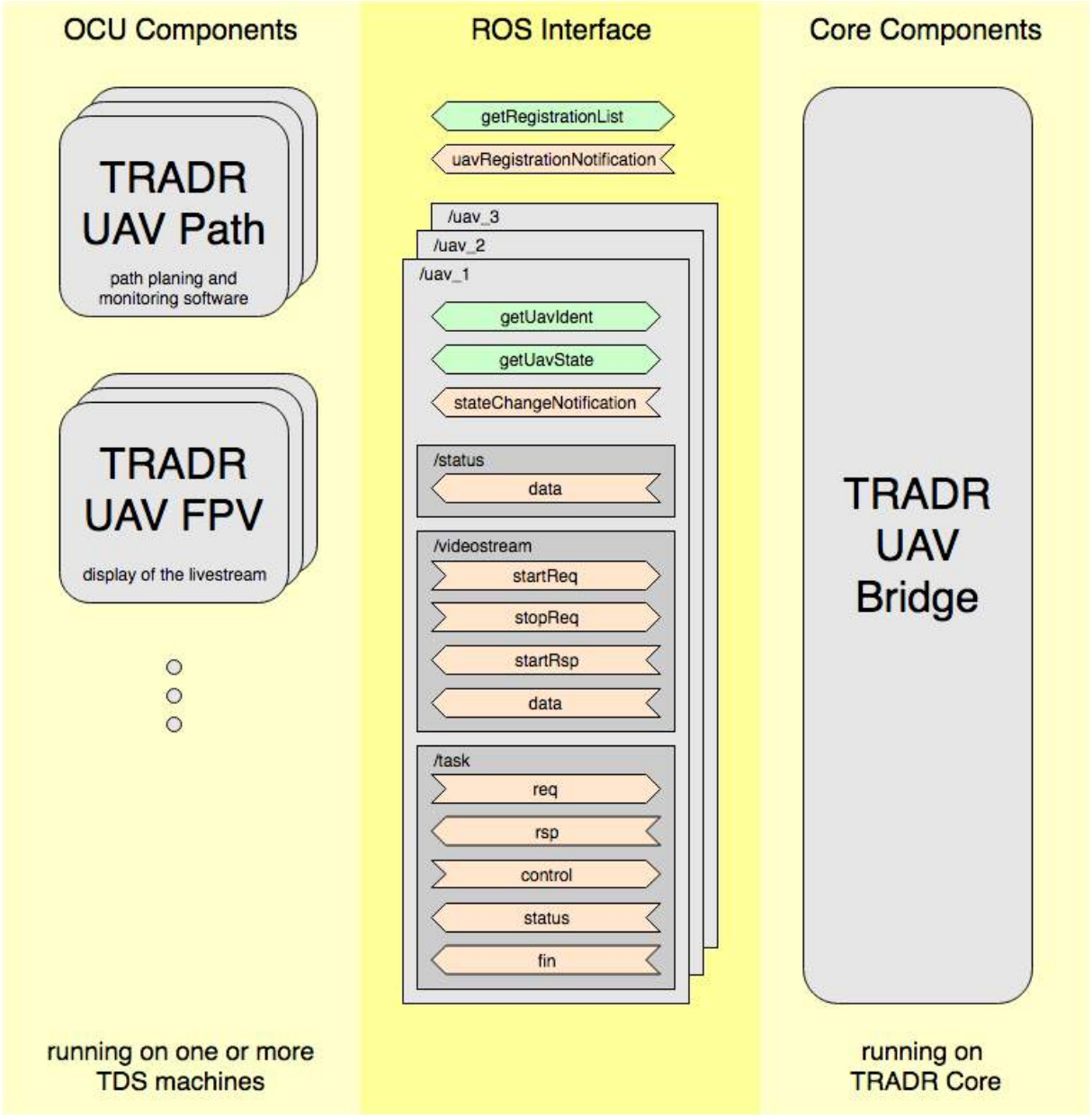

Abbildung 3.25: ROS-Schnittstelle zwischen Bridge und OCU-Komponenten (ROS-Services: hellgrün | ROS-Topics: hellrot)

Verwaltungsschnittstelle

Die TRADR UAV Bridge bietet direkt nach ihrem Start den Service „getRegistrationList“ an, der dem Aufrufer eine Liste mit allen am TRADR System registrierten UAVs liefert.

Zusätzlich wird den OCU Komponenten das Topic „uavRegistrationNotification“ angeboten, auf das diese sich einschreiben können. Unter dem Topic veröffentlicht die Bridge eine Nachricht, wenn sich ein UAV am TRADR System angemeldet oder abgemeldet hat bzw. nicht mehr erreichbar ist. Die Nachricht enthält die ID des UAV und Informationen über den Typ des Ereignisses, ob es sich z.B. um eine Anmeldung oder eine Abmeldung handelt.

Für jedes neue UAV, das sich anmeldet, legt die TRADR UAV Bridge einen Namensraum für dieses an. Die Bezeichnung des Namensraumes enthält die ID-Nummer des UAVs innerhalb des TRADR Systems. Wird das UAV wieder am TRADR System abgemeldet oder wird die Verbindung zum Fluggerät getrennt, löscht die Bridge den Namensraum wieder.

Unter dem angelegten Namensraum bietet die Bridge Topics und Services an, die sich genau auf dieses UAV beziehen. Sind mehrere UAVs gleichzeitig angemeldet, existieren auch mehrere Namensräume.

Namensräume der UAVs

Für alle mit den UAVs zusammenarbeitenden OCU Komponenten sind die Services „getUavIdent“ und „getUavState“ sowie das Topic „stateChangedNotification“ interessant.

Über den Service „getUavIdent“ ist es möglich, die Informationen abzufragen, die das UAV identifizieren. Dazu zählen Angaben wie Name, GUID[80], Model-Bezeichnung, usw. des jeweiligen UAVs.

Der Service „getUavState“ liefert Auskunft darüber, in welchem Zustand sich die Zustandsautomaten der einzelnen Unterprotokolle befinden.

Über das Topic „stateChangedNotification“ informiert die Bridge die OCU Komponenten, wenn es zu einer Änderung eines der Zustände gekommen ist.

Neben dieser allgemeinen Schnittstelle erstellt die Bridge im Namensraum des jeweiligen UAVs drei Unternamensräume für die Unterschnittstellen. Dies sind die Namensräume „status“ zum Empfang von Status- und Telemetrie-Informationen, „videostream“ zum Empfang des Livebildes und „task“ zur Übermittlung, Steuerung und Überwachung autonom zu fliegender Explorationsaufgaben.

Namensraum „status“

Im Namensraum „status“ existiert das Topic „data“ unter dem die Bridge Status- und Telemetriedaten des UAVs veröffentlicht, wenn diese solche empfängt.

Namensraum „task“

Zur Übermittlung, Steuerung und Überwachung von autonom auszuführenden Explorationsaufgaben existieren im Namensraum „task“ mehrere Topics.

Über das Request-Topic „req“ kann die Planungssoftware eine Anfrage senden mit einer Explorationsaufgabe samt Flugbahn.

Auf das Topic „rsp“ veröffentlicht die Bridge die vom UAV eingehende Antwort-Nachricht auf die gestellte Anfrage. Die Planungssoftware kann dadurch erkennen, ob eine angefragte Ausführung vom Piloten angenommen oder abgelehnt wurde.

Mittels des Topics „control“ ist es der Planungssoftware möglich, Nachrichten zur Steuerung der auszuführenden Aufgabe zu senden. Damit kann eine Aufgabe z.B. pausiert oder abgebrochen werden.

Während das UAV die Aufgabe ausführt, veröffentlicht die Bridge unter dem Topic „status“ vom UAV eingehende Status-Nachrichten, die den Fortschritt des Prozesses beschreiben. Über diese Nachrichten kann die Planungs- und Überwachungssoftware auf ihrer Karte darstellen, welche Ziele der Aufgabe bereits erfolgreich angeflogen und welche Aktionen schon ausgeführt worden sind. Somit kann der Operator stets den Fortschritt der Ausführung grafisch mitverfolgen. Er weiß zu jedem Zeitpunkt, welche Daten bereits gesammelt worden sind und bei welchen dies noch aussteht. Dadurch kann er abschätzen, wie lange bestimmte Vorgänge noch dauern werden und wann die Ausführung abgeschlossen sein wird. Über dieses Topic werden keine Ergebnisse in Form von gesammelten Daten übermittelt, da diese direkt von der TRADR UAV Bridge in die Missionsdatenbank geschrieben werden. Die OCU Komponenten können sich diese von der Datenbank beziehen.

Über das „fin“ Tpoic teilt die Bridge der Planungs- und Überwachungssoftware mit, dass die Ausführung der Explorationsaufgabe beendet wurde, wenn dies geschehen ist.

Nemensraum „videostream“

Unter dem Namensraum „videostream“ bietet die Bridge vier Topics für den Übertragungsvorgang des mit Telemetriedaten angereicherten Livestreams der UAV Kamera.

Die Topics „startReq“ und „stopReq“ ermöglichen es der Anzeigesoftware zu veranlassen, dass das UAV die Übertragung des Videostreams startet oder stoppt. Hierzu schickt die Anzeigesoftware eine Anfrage-Nachricht auf dieses Topic, die die Bridge an die UAV Seite weiterleitet.

Vor der Übertragung des Livebildes veröffentlicht die Bridge unter dem Topic „startRsp“ die Antwort-Nachricht. Im positiven Fall, bei dem die Start-Anfrage von der UAV Seite angenommen worden ist, enthält die Anwort-Nachricht auch das zum Dekodieren notwendige Start-I-Frame.

Unter dem Topic „data“ sendet die Bridge die mit Telemetrie-Daten angereicherten H.264-Datenpakete des Livestreams.

3.5 Besonderheiten der Livebild-Übertragung

3.5.1 Der „Infinite GOP“-Modus

Bei den DJI UAVs wird das Livebild mittels der fest eingebauten Kameras aufgenommen und hardwarebeschleunigt komprimiert. Das verwendete Video-Codec ist MPEG4-AVC (H.264)[63][8][84]. Die verwendeten Parameter der Kompression entsprechen jedoch keinem standardisierten H.264 Profil[83]. Das Missachten der vorgegebenen MPEG-Profile führt dazu, dass die Kompatibilität mit H.264 Decoder-Software/Hardware nicht garantiert werden kann[39][37][23]. Es ermöglicht jedoch höhere Kompressionsraten, geringere Latenzen und einfachere Implementierungen als es bei Verwendung eines der Standard-Profile der Fall wäre[49]. Unter anderem benutzt der Hersteller DJI als Modus für die GOP-Länge[5][79][57][36] den sehr unüblichen „Infinite GOP“-Modus. Normalerweise besteht ein Videostream aus einer Folge von vollen Referenzbildern (I-Frames / Keyframes), gefolgt von mehreren Differenzbildern (P-Frames). Die P-Frames enthalten dann nur die Änderungen der geänderten Bildbereiche in Referenz zum vorangegangenen I-Frame. Eine solche Gruppe aus einem I-Frame und folgenden P-Frames wird als GOP (Group of Pictures)[5][79][57][36] bezeichnet. Bei Verwendung von „Infinite GOP“ besitzt der Stream jedoch keine I-Frames. Stattdessen wird in den P-Frames zu den sich geänderten Bereichen auch immer ein ganzer Bereich fester Größe erneuert. Dieser Bereich wandert über eine größere Menge von P-Frames hinweg über das gesamte Bild. Aufgrund dieser Arbeitsweise ist das seltene „Infinite GOP“-Verfahren auch als „Rolling Macroblock Refresh“ bekannt. Ein Decoder, der diesen Datenstrom dekodieren soll, muss in der Lage sein mit einem „Infinite GOP“ also einer unendlichen Folge aus P-Frames ohne I-Frames umgehen können. Zudem muss ein leeres Start-Keyframe (Schwarzes Bild in korrekter Größe) als erstes Datenpaket des Videostreams dem Decoder übergeben werden. Alle weiteren P-Frames beziehen sich dann in Folge auf dieses Start-Keyframe[49][50][5].

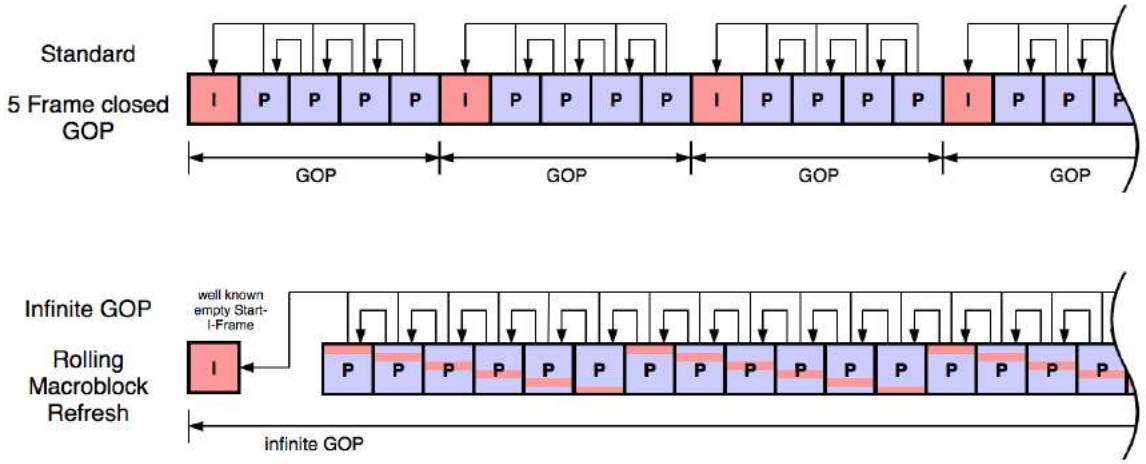

Abbildung 3.26: Unterschiede zwischen „Infinite GOP“ und Standard GOP-Formaten

„For low latency operator-in-the-loop applications, an infinite GOP Length sequence can help reduce the latency at the expense of interoperability."[49]

„Infinite GOP or P-frame only uses a technique known as rolling macroblock refresh to remove the need for I-frames. In effect the I-frame is smeared over a number of P-frames. Infinite GOP or P-frame only is not widely supported but can provide the lowest latency form of encoding (along with I-frame only) due to the ability to use relatively fixed frame sizes and a constrained buffer model."[50]

„In infinite GOP mode a single I-picture is transmitted at the start of the sequence, and then all other pictures are coded as P-pictures. In this mode, there are no random access points. The benefit of this mode is twofold. First, since all the coded pictures are roughly the same size (excluding the first I-picture), there is a significant reduction in the buffering that is required in the encoder and the decoder during constant bitrate operation. This reduction in buffering translates into a reduction in latency which is critical in many real time applications. Second, since no I-pictures are transmitted, only more efficient P-pictures are used. In addition to the loss of random access, this mode also is very susceptible to transmission errors which would persist forever if unchecked. To correct this, intra refresh is used. In intra refresh, a small number of blocks are intra coded in each picture. The intra coded blocks are like I-pictures for just that block, correcting any transmission errors that may have occurred. By selecting the number of intra coded blocks per picture, you can select the number of pictures over which the entire picture will be refreshed. The SPS and PPS can still be sent periodically to aid decoders which receive the stream in mid sequence."[49]

In den seltenen Echtzeitanwendungen, die "Infinite GOP„ verwenden, wird in der Regel das leere Start-Keyframe nicht mitübertragen sondern ist dem Empfänger bekannt, der dieses beim Initialisieren des Decoders einmal zu Beginn dem Datenstrom voranstellt. Auch bei DJI wird das Start-Keyframe nicht übertragen[23][9][24]. Dieser rohe H.264-Datenstrom kann nicht in ein Container-Format (z.B. MPEG Transport Stream) gekapselt werden, wie es in anderen Anwendungen üblich ist, da seine innere Struktur keinem standardisierten MPEG-Profil entspricht[39][37]. Er wird daher roh über die Funk-Schnittstellen DJI Lightbridge (bei Phantom 3/4) oder DJI OcuSync (bei DJI Mavic Pro) zum Fernsteuerungscontroller übertragen. Von dort aus gelangen die Datenpakete über das USB-Kabel zum Smartphone des Piloten, das an der Fernsteuerung befestigt ist. Über die SDK-Bibliothek des Herstellers ist es als Entwickler einer App möglich, die Pakete des rohen H.264-Datenstrom als Bytearray zu nutzen. Da der Datenstrom roh ist, fehlen jegliche Metainformationen und das Start-Keyframe, die ein Decoder benötigt, um den Datenstrom zu dekodieren. DJI hat Start-Keyframes mit Metainformationen für verschiedene UAVs und Auflösungen als Ressourcen im Mobile-SDK hinterlegt[24][9], um diese in einem internen Decoder zu nutzen. Leider erlaubt der interne Decoder nur die Anzeige des Livestreams auf der Benutzeroberfläche der App. Offiziell stehen zum Zeitpunkt der Erstellung dieser Arbeit keine Funktionen und Dokumentationen bereit, um auf die Start-Keyframes zuzugreifen[39][37]. Jedoch lassen diese sich dennoch bei einer Suche innerhalb der internen Ressourcen des SDKs

finden. Versuche mit dem manuellen Einbringen dieser Binärdaten am Beginn des Datenstroms haben gezeigt, dass die Videoanzeigesoftware VLC-Player in der Lage ist, einen Videostrom im „Infinite GOP"-Modus darzustellen[24]. Wurden die Binärdaten des zur UAV-Kamera passenden Start-Keyframes nach dem Initialisieren des VLC-Decoders als erstes Datenpaket dem Decoder übergeben, war dieser in der Lage, die vom UAV eingehenden P-Frames zu dekodieren.

3.5.2 Realisierung

Um die Latenz möglichst gering zu halten, wird in der gewählten Realisierung der Datenstrom nicht verändert, sondern in der am Smartphone empfangenen Form an das TRADR System weitergeleitet. Auf dem TDS im Leitstand wird eine Anzeigesoftware (UAV FPV) ausgeführt, die den Videostream dekodiert und darstellt. Das Programm nutzt die Software-Bibliothek libVLC[52][51][58], die die aus vielen Decodern bestehende Video-Playback-Engine des VLC-Players zur Verfügung stellt. Durch die Verwendung der Bibliothek wird zur Laufzeit dem Anzeigeprogramm eine VLC-Decoder-Instanz als Kindprozess zur Verfügung gestellt. Die Interprozesskommunikation läuft sowohl über eine Pipe als auch über ein gemeinsames Speichersegment. Das Anzeigeprogramm empfängt das zur UAV-Kamera passende Start-Keyframe über die Antwortnachricht auf eine Start-Anfrage. Die App sucht das zum verwendeten UAV passende Start-Keyframe und überträgt dieses in der Antwort-Nachricht zur Bridge, die dieses über ROS-Topics an die Anzeigesoftware weiterleitet. Als Reaktion auf eine eingehende positive Antwort-Nachricht initialisiert das Anzeigeprogramm eine neue VLC-Decoder-Instanz und schreibt das Start-Keyframe in die Input-Pipe, die mit dem Decoder verbunden ist. Danach wird jedes eingehende H.264-Datenpaket, das von der App über die Bridge beim Anzeigeprogramm eingeht, in die Pipe geschrieben. Dieses manuelle Einfügen eines leeren Keyframes in den Datenstrom auf Empfängerseite wird in Entwicklerkreisen auch als „I-Frame-Injection" bezeichnet. Asynchron zu den hineinlaufenden Daten entstehen als Ergebnis des Dekodiervorgangs die Einzelbilder des Videostreams. Über die Bibliothek libVLC wird das Anzeigeprogramm benachrichtigt, wenn ein Einzelbild fertig synthetisiert (gerendert) ist. Dieses kann dann von der Anzeigesoftware aus dem gemeinsamen Speichersegment gelesen und angezeigt werden.

Ein Grund, das Dekodieren auf jedem anzeigenden TDS-Rechner dezentral durchzuführen, ist, dass so der Videostream auch innerhalb des Leitstandes in komprimierter Form verteilt werden kann. Bei einer zentralen Dekodierung innerhalb der Bridge auf dem TRADR-Core hätten die resultierenden Bilddaten unkomprimiert zu den verschiedenen TDS-Rechnern geschickt werden müssen. Zwar sind TRADR-Core und die TDS-Rechner über ein Kabelnetzwerk miteinander verbunden, indem in der Regel keine Paketverluste auftreten, jedoch sind die Datenraten von 664 Mbit/s bis 1,5 Gbit/s auch für ein solches Netzwerk ein Problem. Zudem würde eine so extreme Netzwerkauslastung die Kommunikation zwischen anderen Komponenten im TRADR Leitstand stören.

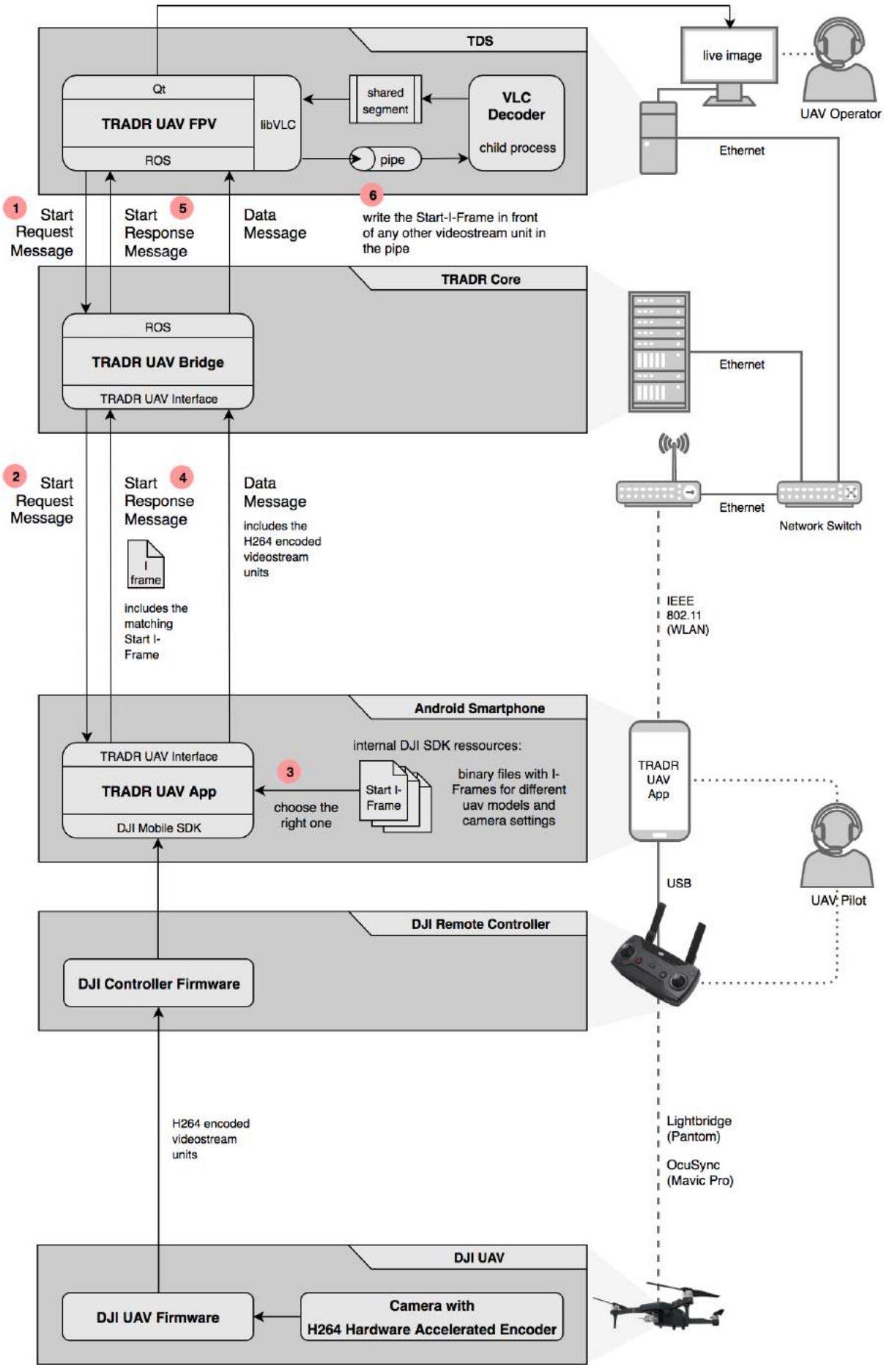

Abbildung 3.27: Initialisierung des Videostreams

3.5.3 Alternative Möglichkeiten

Eine alternative Möglichkeit zur I-Frame-Injection auf dem TDS im Leitstand wäre es, den H.264 kodierten Videostream innerhalb der Smartphone App mit dem Decoder des DJI Mobile SDKs zu dekodieren und danach zum Leitstand zu übertragen. Der Dekodiervorgang würde bei Verwendung des DJI Decoders auch hardwarebeschleunigt geschehen [9]. Eine Folge aus Einzelbildern wäre das Ergebnis.

Unkomprimierte Übertragung

Die resultierenden Einzelbilder unkomprimiert über das Funknetzwerk (WLAN) zum TRADR-System zu übertragen, ist keine gute Lösung, da die Datenrate extrem groß und somit nicht geeignet ist um über Drahtlosnetzwerke übertragen zu werden.

Modellrechnung				
Auflösung	Farbtiefe	FPS	Größe Einzelbild	Datenrate
1280x720	24 bit	30	2,8 MB	664 Mbit/s (663.552.000 bit/s)
1920x1080	24 bit	30	6,2 MB	1.500 Mbit/s (1.492.992.000 bit/s)

Tabelle 3.28: Modellrechnung für die Übertragung von unkomprimierten Videos

Zudem müsste eine Softwareschicht eingebaut werden, die es ermöglicht, größere Nachrichten, konkret die Daten eines kompletten einzelnen Frames (2,8 MB) durch mehrere UDP-Nachrichten zu übertragen, da die Nutzdaten einer UDP-Nachricht nicht größer als 65,5 KB sein dürfen. Kommen diese in der falschen Reihenfolge an, müssen sie korrekt umsortiert und wieder zum Gesamtbild zusammengesetzt werden. Fehlt eine Nachricht, muss der korrespondierende Bereich im Bild mit etwas anderem ersetzt werden.

Wird zum Transport statt UDP das TCP-Protokoll verwendet, ist eine solche Softwareschicht überflüssig, da die beschriebenen Vorgänge bereits Bestandteil des TCP-Protokolls sind. Die Verwendung von TCP führt aber bei hohen Datenraten und fortlaufend vorkommenden Paketverlusten aufgrund der Drahtlosübertragung dazu, dass der Overhead der Sendebestätigung und Übertragungswiederholung bei Paketverlusten immer weiter steigt und sich stark kontraproduktiv auswirkt.

Erneutes H.264-Kodieren

Alternativ zur unkomprimierten Übertragung besteht die Möglichkeit, den Datenstrom nach dem Dekodieren mittels des SDK internen Decoders erneut in das H.264-Format[84] zu kodieren und komprimiert zu übertragen. In diesem Fall ist die Nutzung eines standardisierten Parameter-Profiles[83] mit einer Standard-GOP-Länge und das Verpacken in einen Transport Stream (MPEG TS)[92] oder RTSP/RTMP Stream[93][94] möglich. Dies macht das Dekodieren auf der Empfängerseite deutlich einfacher und eine I-Frame-Injection ist nicht notwendig. Zu einem großen Problem wird hierbei jedoch der enorme Rechenaufwand des Kodierprozesses, der bei einer Verarbeitung auf dem Smartphone zu einer problematischen Latenz des Livebildes führt. Zudem wird der Smartphone Prozessor ausgelastet, was zu einer schlechteren Akkulaufzeit führt. Auch durch das Puffern der GOP-Folgen beim Kodieren und Dekodieren kommt es zu einer weiteren problematischen Erhöhung der Latenz.

Da eine geringe Latenz zwischen der real aufgenommenen Welt und dem angezeigten Bild im Leitstand gefordert ist, ist auch das erneute H.264-Kodieren keine gute Lösung.

Motion-JPEG Kompression

Im Gegensatz zu einer extrem aufwendigen H.264-Kompression könnte eine weniger kompakte dafür aber auch nicht so aufwendige Methode gewählt werden. Im Umfeld von ROS wird hier oft als Codec Motion-JPEG (MJPEG)[91][26] eingesetzt. Dabei bleibt der aus Einzelbildern bestehende Videostream weiterhin in Einzelbildern erhalten. Jedes Einzelbild des Videostreams wird jedoch mit dem JPEG Verfahren[86] komprimiert. Die Kompressionsrate bei Motion-JPEG ist zwar deutlich niedriger als bei H.264, jedoch ist der Rechenaufwand für das Kodieren so viel niedriger, dass eine solche Kompression auch auf dem Smartphone möglich wäre, ohne dass die Latenz zu stark ansteigt[45]. Auch ein Latenzanstieg aufgrund der Pufferung einer GOP-Folge entfällt bei Motion-JPEG, da jedes Bild alle Informationen enthält und Differenzbilder nicht vorkommen.

4 Zusammenfassung

Inhaltsangabe

4.1 Fazit

Die Integration von Kompakt UAVs des Herstellers DJI in die Leitstandsinfrastruktur des TRADR Systems kann als erfolgreich umgesetzt angesehen werden. Es konnte gezeigt werden, dass die UAVs vom Typ Mavic Pro und Phantom 3/4 per Smartphone App gesteuert werden konnten. Während des Fluges sendeten die UAVs Telemetriedaten und das Livebild der Kameras. Beides wurde dem Piloten in der App korrekt dargestellt. Der Pilot konnte mit der App Explorationsaufgaben erstellen, die von den UAVs wie geplant ausgeführt wurden. Darüber hinaus wurde das vom Piloten benutzte Smartphone mit der App über das TRADR Funknetzwerk (WLAN) mit dem TRADR Leitstand verbunden. Es bestätigte sich, dass während das UAV im Außenbereich flog, die Telemetriedaten im Inneren des Leitstandraumes eingingen und in der Überwachungssoftware (UAV Path) korrekt dargestellt wurden. Zeitgleich konnte im Leitstand das Livebild der UAV-Kamera, der im Außenbereich fliegenden Drohne, betrachtet werden. Das Bild erreichte überwiegend störungsfrei in einer Auflösung von 1280x720 Pixeln bei 30 FPS den Leitstand und wurde in der Anzeigesoftware (UAV FPV) dargestellt. Die Latenz lang dabei bei unter einer Sekunde. Somit war es den Personen im Leitstand möglich aus der Sicht des UAVs die Umgebung in Echtzeit zu betrachten, während die Drohne im Außenbereich flog. Zudem konnte demonstriert werden, dass Explorationsaufgaben, die von einer Person im Leitstandsraum mittels einer Planungssoftware erstellt wurden, erfolgreich zum Smartphone des Piloten übertragen wurden. Der Pilot konnte die empfangene Aufgabe durch das UAV ausführen lassen. Das UAV flog daraufhin die Flugbahn ab, wie diese im Leitstand geplant war und machte an den gegebenen Punkten Bildaufnahmen. Nach Ausführen der Aufgabe konnten die aufgenommenen Bilder auf der Speicherkarte der UAV-Kamera gefunden werden. Die Aufnahmen wurden in dem Winkel getätigt, der für den jeweiligen Wegpunkt in der Aufgabe spezifiziert war. Damit wurde prinzipiell gezeigt, dass es möglich ist, die UAVs Mavic Pro und Phantom 3/4 über den in dieser Arbeit beschriebenen Weg an den TRADR Leitstand anzubinden. Der TRADR Leitstand kann dabei exemplarisch für Leitstandsinfrastrukturen in Rettungseinsätzen gesehen werden.

Neben dem prinzipiellen Erfolg werden in den folgenden Abschnitten die aufgetretenen Probleme dieser Lösung ausführlich diskutiert, um in zukünftigen Projekten ähnlicher Thematik bessere Ergebnisse zu erzielen.

Abbildung 4.1: Luftaufnahme der mit dem Leitstand verbundenen Mavic Pro bei einem Integrationstest

Abbildung 4.2: Mit dem Leitstand verbundene AscTec Neo bei einem Flugtest

4.1.1 Probleme mit der Stabilität und Softwarequalität

Eines der Hauptkritikpunkte ist, dass bis zuletzt Stabilitätsprobleme beim Betrieb des in dieser Arbeit beschriebenen Gesamtsystems aufgetreten sind.

Konkret waren vereinzelt schwer vorhersagbare Programmabstürze aufgrund von Speicherzugriffsverletzungen (SegFault) und inkonsistente Zustände mit Dead-Lock-Verhalten in allen beteiligten Komponenten zu beobachten. Besonders stark war dieses im Kontext der Livebild-Übertragung zu erkennen.

Die Probleme lassen sich hauptsächlich auf zwei Ursachen zurückführen. Zum einen ist die Softwarequalität leider nicht ausreichend hoch, sodass Fehler in der Software vermutet werden und deren Vorhandensein als sehr wahrscheinlich eingeschätzt wird. Zum anderen wurden Software-Bibliotheken eingesetzt, die in der benutzten Version keine guten Schnittstellen anboten (libVLC). Dies führte dazu, dass auch immer wieder Workarounds benutzt wurden, um Probleme zu lösen. Vor allem im Kontext der Livebild-Übertragung wurde dies besonders deutlich, da es hier bis zuletzt beim Betrieb der Software zu Instabilitäten kam.

Als Ursache für die niedrige Softwarequalität ist zu nennen, dass die in dieser Arbeit entstandenen Softwarekomponenten im Rahmen von Forschungsaktivitäten entstanden und nicht dem Anspruch von kommerzieller Produktivsoftware entsprechen mussten. Vielmehr sollten Prototypen entstehen, an denen grundsätzlich das in dieser Arbeit beschriebene Konzept einer UAV Integration gezeigt wird. Mangelnde Kapazitäten von Arbeitskraft und Zeit führten zu Prototypen, die erfolgreich Lösungen für die Fragestellungen boten, jedoch nicht abschließend als stabile, produktiv einzusetzende Software angesehen werden kann.

Um aufgetretene Probleme in Folgeprojekten zu vermeiden, wird daher empfohlen, behutsamer zu entwickeln und einzelne Module der Softwarekomponenten während des Entwicklungsprozesses fortlaufend kleinschrittig zu testen. Solche Tests und ein achtsames Vorgehen beim Umsetzen einzelner Anforderungen können die gesamte Entwicklungszeit deutlich verlängern, sollten jedoch zu einer besseren Softwarequalität führen.

Besonders kritische verbesserungswürdige Punkte sind die Absicherung von konkurrierenden Zugriffen auf Ressourcen beim Einsatz von Multithreading, sowie ein gutes Management für den Umgang mit ungültig gewordenen Ressourcen. Für beide Probleme gibt es gute konzeptionelle Lösungen in den jeweiligen Programmiersprachen, die jedoch zum Teil nicht verwendet werden konnten, da benutzte Bibliotheken die notwendigen Sprach-Funktionalitäten nicht unterstützten (libVLC). Beim Auftreten solcher ungünstiger Konstellationen sollte entweder nach einer Alternative für die problematische Bibliothek gesucht werden oder es sollte viel Aufwand in die Entwicklung und Tests von Workarounds investiert werden, damit diese das Problem analytisch beweisbar sicher lösen, sodass es zu keinen unvorhersehbaren Fehlern kommt. Gerade im Kontext von Multithreading-Anwendungen in Kombination mit C/C++-Bibliotheken ist eine besondere Sorgfalt notwendig, um konzeptionell sauber zu arbeiten und nach Möglichkeit

fehlerfreie Software zu entwickeln.

4.1.2 Erfahrungen mit dem DJI Mobile SDK

Die Arbeit mit dem DJI Mobile SDK[19] nahm bei der App Entwicklung den größten Anteil ein. Daher sollen einige Erfahrungen hier beschrieben werden.

Positiv ist zu nennen, dass das SDK einen sehr großen Funktionsumfang bietet. Fast alle Einstellungsmöglichkeiten, die sich in der Hersteller App (DJI GO)[12] tätigen lassen, finden sich im SDK wieder. Es existieren umfangreiche Möglichkeiten Flugmissionen zu definieren. Gut ausgearbeitete Online-Tutorien[17] mit Beispiel-Quelltexten helfen zusätzlich zur Dokumentation[16] bei der Umsetzung der eigenen App.

Andererseits muss negativ angemerkt werden, dass Funktionen bestanden, die nicht korrekt arbeiteten, wie in der Dokumentation beschrieben. Ungünstiger weise ist es in solchen Fällen schwierig herauszufinden, dass der Fehler nicht beim Verwender der Bibliothek sondern in dieser selber liegt. Durch Nachforschungen in den Entwickler-Foren von DJI ist es gegebenenfalls möglich, die Ursache als einen Software-Fehler im SDK auszumachen. Hinzu kommt, dass die Dokumentation in der Tiefe deutlich schlechtere Qualität besitzt, bis hin zu Methoden, die selber als auch ihre Aufrufparameter keine sinnvollen Bezeichnungen mehr besitzen.

Als Ursache wird vermutet, dass das SDK relativ neu ist (September 2015) und sich noch stark im Wandel befindet. Neue Versionen erscheinen in kurzen Zyklen von einem bis wenigen Monaten. In jeder dieser Updates behebt der Hersteller zahlreiche Fehler. Die Dokumentation wird in der Tiefe nach und nach verbessert, während der Funktionsumfang mit Update stark wächst.

Bis zuletzt gab es in der aktuellen TRADR UAV App den Fehler, dass nicht alle Bilder, deren Aufnahme in einer Explorationsaufgabe spezifiziert wurde, aufgenommen wurden. Beim Testen kam es immer wieder vor, dass vereinzelt Bilder fehlten. Da die wenigen fehlenden Fotos zufällig in Anzahl und Position in der Reihenfolge wechselten, ist kein systematischer Fehler in der Verwendung des SDKs zu erkennen. Stattdessen gibt es Foreneinträge, in denen Entwickler von ähnlichen Problemen berichten. DJI kündigte an das Problem in einer späteren Version des SDKs zu beheben. Bis zuletzt konnte nicht eindeutig geklärt werden, ob das Problem vom eingesetzten UAV Modell und dessen Firmware-Version abhängig ist. Auch ein Fehler in der UAV-Firmware könnte hier eine mögliche Ursache sein.[25][38]

Zusammengefasst ist das SDK als eine sehr leistungsstarke und umfangreiche Möglichkeit zu sehen, die DJI Drittanbietern zur App Entwicklung zur Verfügung stellt, die jedoch beim praktischen Einsatz einige Schwächen zeigt. Aufgrund der beschriebenen Fehler, der in wenigen Teilen nicht vorhandenen Dokumentation und der kurzen Updatezyklen mit großen Änderungen, ist das SDK als eine noch nicht vollständig ausgereifte Software zu sehen, die aktuell intensiv weiterentwickelt wird.

4.1.3 Funktionsumfang beim Einsatz verschiedener UAV Modelle

Während der Smartphone App Entwicklung mittels des DJI Mobile SDKs ließen sich bestimmte Aspekte nicht wie geplant umsetzen, da Funktionen des SDKs nicht wie beschrieben funktionierten. Als Fehlervarianten traten zum einen das Ausbleiben von Reaktionen auf angeforderte Aufrufe, zum anderen wurden Funktionen mit einer Fehlermeldung abgebrochen. Bei erster Variante lässt sich die Ursache nicht sicher herleiten. Hier konnte nur aus dem Kontext spekuliert werden. Bei der zweiten Variante bestand die Beschreibung der Fehlermeldung, doch nicht in jedem Fall konnte daraus die Ursache ausgemacht werden.

Ein Grund, der häufiger entweder vermutet oder sicher aus der Fehlermeldung entnommen werden konnte, war die nicht vorhandene Verfügbarkeit bestimmter vom SDK angebotener Funktionen für das eingesetzte UAV Modell. Zum Begriff UAV Modell ist noch zu erwähnen, dass zu einer Modellserie wie z.B. der DJI Phantom 2 verschiedene Modellversionen beispielsweise Phantom 2 Standard, Phantom 2 Professional, Phantom 2 Advanced, usw. existieren. Der Funktionsumfang unterscheidet sich nicht nur zwischen Modellserien sondern auch zwischen den jeweiligen Modellversionen deutlich. Insgesamt ergibt sich dadurch eine große unübersichtliche Menge unterschiedlicher Modelle mit unterschiedlichem Funktionsumfang.

Da das DJI Mobile SDK für alle UAV Serien und Modelle sowie aller externer Erweiterungshardware des Herstellers DJI entworfen wurde, bietet es die gesamten Funktionen, die jegliche Hardwarekombinationen bieten können. Somit verwundert es nicht, dass bei Verwendung eines bestimmten UAV Modells nur ein kleiner Bruchteil der angebotenen Funktionen wirklich zur Verfügung steht. Vieles lässt sich als Entwickler erschließen. Sensoren, die auf dem verwendeten UAV nicht vorhanden sind, lassen sich logischerweise über das SDK nicht nutzen, auch wenn sämtliche Funktionen angeboten werden. Manchmal lässt sich dies jedoch nicht herleiten, beispielsweise bei der Übertragung von auf der Speicherkarte des UAVs befindlichen Dateien zum Smartphone.

Wünschenswert wäre, dass das SDK diese Problematik löst und sich die Verfügbarkeit jeder einzelnen Funktion für jedes einzelne Modell zur Laufzeit prüfen ließe.

Nicht so komfortabel aber ausreichend wären in den Dokumentationen verfügbare Listen, in denen abzulesen wäre, welche Funktionen des SDKs bei welchem Modell funktionieren. DJI bietet in den Dokumentationen entweder keine oder höchst unzureichende Listen, die bei weitem nicht alle Funktionen aufzählen oder in denen die Funktionen sich nicht zu den Methoden im SDK zuordnen lassen.

Zusammenfassend ist daher zu sagen, dass bei der Verwendung verschiedener UAV Modelle des Herstellers DJI in einem Projekt die Komplexität deutlich steigt, da bestimmte Funktionen, die auf dem einen Modell verfügbar sind und ordnungsgemäß arbeiten, diese auf dem anderen Modell eventuell versagen. Es ist im Vorfeld nicht sicher zu stellen, welche Funktionen ein UAV Modell hinsichtlich des SDKs bietet. Hier helfen Erfahrungsberichte von anderen Entwicklern.

Vor der Anschaffung eines UAVs, für das in einem Projekt eine App mittels des Mobile SDKs entwickelt werden soll, wird empfohlen den Funktionsumfang genau zu prüfen.

4.1.4 Bildübertragung

Die Übertragung des Livebildes stellte bis zuletzt einen der problematischsten und schwierigsten Punkte dar. Positiv ist, dass es prinzipiell funktioniert und ein Livebild mit einer Latenz von unter einer Sekunde bis in den Leitstand übertragen und angezeigt werden konnte. Die Übertragung erfolgte dabei die gesamte Strecke in digitaler Form mit einer Auflösung von 1280x720 Pixeln. Negativ ist aber, dass die Form, in der das Livebild übertragen wurde, kein standardisiertes Format besitzt, welches von allgemein gebräuchlichen Decodern dekodiert werden kann. Um den Videostream zu dekodieren, musste eine eigene Software implementiert werden, die eine allgemein gebräuchliche Decoder-Software auf eine untypische Weise mit speziellen Parametern startet und dieser Software die Daten in besonderer Form zuführt (I-Frame-Injection). Genau diese Form der besonderen Zuführung der Daten ist jedoch äußerst fehleranfällig und wirkt sich negativ auf die Stabilität der Softwarekomponenten aus. Es wäre an dieser Stelle sehr wünschenswert, dass der Videostream in Form eines MPEG Transport Streams (MPEG TS), wie es bei der Übertragung von digitalem Fernsehen benutzt wird, von der App zur Verfügung gestellt werden kann.

Zum Zeitpunkt der Entwicklung der App bestand kein akzeptabler Weg, dies mit dem DJI SDK zu gewährleisten[39][37][23]. Die aktuell in der App realisierte Variante mit I-Frame-Injection ist dagegen ein von DJI selbst beschriebener Workaround[24]. Auf Nachfrage und Kritik von App Entwicklern bot DJI diesen Software Hack an, um das externe Dekodieren ohne Umkodieren zu ermöglichen. Neben dieser Variante wurde auch eine Lösung angeboten, bei der der Videostream mit dem DJI eigenen Decoder dekodiert wird und danach erneut durch eine externe Software (z.B. FFMPEG) kodiert werden kann, um einen Transport Stream oder RTMP/RTSP-Stream zu erzeugen[9]. Bei dieser Variante entstand jedoch eine problematische Latenz sowie eine hohe Auslastung des Smartphones, das folglich zu geringeren Akkulaufzeiten führte. DJI gab an, die Wünsche der Entwickler zu berücksichtigen und in einer neueren Version des SDKs den H.264 Datenstrom in standardisierter Form auszugeben, sodass eine I-Frame-Injection nicht nötig wäre und jeder H.264 Decoder den Datenstrom dekodieren könne. Wenn dies der Fall wäre, würde auch die Möglichkeit bestehen, einen solchen Datenstrom in Container-Formate wie einen MPEG Transport Stream oder RTMP/RTSP-Stream zu verpacken. Bis zum Zeitpunkt der Erstellung dieser Arbeit war keine Version des SDK verfügbar, die diese Möglichkeit bot. Es ist zu vermuten, dass alle Drittanbieter-Apps für die Bereitstellung eines RTMP/RTSP-Streams den Stream auf dem Smartphone umkodieren.

Die Variante, die realisiert wurde, ermöglicht eine Video-Übertragung mit geringer Latenz, hat jedoch einige Stabilitätsprobleme und Einschränkungen.

4.1.5 Kritik an den verwendeten Karten

In der aktuellen App werden die Karten von Google Maps[81] eingesetzt, jedoch stellt dies keine gute Lösung dar. Es gab mehrere Gründe für die Entscheidung bei der Entwicklung der App für die Kartendarstellung als ersten Ansatz die Kartendienste von Google Maps zu verwenden. Google stellt sein eigenes Angebot an Karten, Satelliten- und Luftbildern, die unter dem Begriff Google Maps bekannt sind, für Android besonders komfortabel zur Verfügung[30]. Neben der Möglichkeit, bestimmte Kartenausschnitte der Daten von Google herunterzuladen, bietet Google eigene Steuerelemente für die Android-Benutzeroberfläche an, die als Software-Bibliothek in eigenen App Projekten benutzt werden kann[31]. Das Karten-Steuerelement von Google zeigt die verschiedenen Karten und Satellitenbilder in der eigenen App an. Die Steuerung mittels Berührungsgesten, um das Blickfeld zu bewegen oder zu zoomen, realisiert das Steuerelement ohne weitere Arbeit durch den App-Entwickler. Hinzu kommen Features, beispielsweise das Einzeichnen von Marken und geometrischen Figuren (Linien, Kreise, Polygone, usw.) in der Karte, Messen von Distanzen sowie die Umrechnung zwischen geometrischen Koordinaten auf dem Bildschirmausschnitt und geografischen Koordinaten auf der Karte. Diese Möglichkeiten erlauben es dem App-Entwickler, sich auf die eigentliche Problemstellung zu konzentrieren. Darüber hinaus bieten die Google Kartendienste in der Regel von den im Internet frei verfügbaren Karten, Satelliten- und Luftbildern zumeist die mit der höchsten Qualität. So ist es durch den Einsatz des Google Maps Android-Steuerelementes möglich, schnell und einfach eine Kartendarstellung in der eigenen App anzubieten mit sehr qualitativem Inhalt. Daher rechtfertigt sich die Entscheidung für diese Bibliothek als ersten Ansatz bei einer App Neuentwicklung. Im weiteren Verlauf des Projektes sollten jedoch die Nachteile bedacht werden und die Kartendarstellung sowie die dahinter liegenden Dienste gewechselt werden, wenn dies notwendig erscheint. Durch ein gut überlegtes Software Design mit Abstraktionsschichten sollte das Austauschen einer Komponente wie der Kartendarstellung unproblematisch sein. Bei der Entwicklung der in dieser Arbeit vorgestellten App ist im Design auf diesen Aspekt geachtet worden, sodass es möglich wäre, die Kartendarstellung durch eine von Google unabhängige Variante auszutauschen, ohne das Änderungen an anderen Teilen der App nötig wären.

Der Anwendungskontext der im TRADR Projekt erforschten Lösungen ist ein Rettungseinsatz im Katastrophengebiet. In diesem speziellen Anwendungsfall ist nicht davon auszugehen, dass die Computersysteme des Einsatzteams einschließlich der Smartphones der Piloten einen Zugang zum Internet haben. Ohne diesen kann jedoch nicht sicher gewährleistet werden, dass Karten, Satelliten- sowie Luftbilder von Google Maps angezeigt werden können, da diese im Normalfall während der Benutzung auf Abruf aus dem Internet heruntergeladen werden. Ausweichen kann man diesem Problem dadurch, dass man die Daten im Vorhinein bei bestehender Internetverbindung in den Cache des Smartphones lädt und zu späterem Zeitpunkt, wenn keine Internetverbindung mehr bestehen sollte, aus diesem benutzt. Da dieses Verhalten nicht der von Google vorgesehenen Nutzung entspricht, wird dies nicht als eine sichere und gute Lösung angesehen. Benötigt werden stattdessen offline verfügbare Karten, Satelliten- und Luftbilder, die entweder auf den Endgeräten (z.B. dem Smartphone) oder im Einsatzsystem der Rettungs-

kräfte verfügbar sind und von dort geladen werden können. Da die Google Kartendienste diese Möglichkeit nicht bieten, ist ihr Einsatz für das Anwendungsgebiet „Rettungseinsätze im Katastrophengebiet“ keine akzeptable Lösung.

Ein weiteres Problem des in TRADR realisierten Systems besteht darin, dass verschiedene Einsatzkräfte mit unterschiedlichen Karten arbeiten, da jede Anwendung seine eignen Karten oder Webdienste verwendet. Eine gute Lösung könnte daher so aussehen, dass es einen zentralen Server gibt, der für alle Komponenten verschiedene Karten, Satelliten- und Luftbilder zentral über standardisierte Schnittstellen bereitstellt. Alle Komponenten im Rettungseinsatz beziehen dann über das lokale Netzwerk ihre Karten bei diesem Server[88]. Für die über Drahtlosverbindungen (WLAN) angebundenen Komponenten ist es zudem wichtig, dass sie die Karten offline verfügbar machen und diesen Stand auffrischen, wenn auf dem zentralen Server neue Versionen verfügbar sind. Die Offline-Verfügbarkeit ermöglicht es bei unzureichenden Verbindungen zwischen mobilen Komponenten im Einsatzgebiet und dem zentralen Karten-Server trotzdem noch auf Kartendaten zu arbeiten. Die Ausgangsdaten werden in einem solchen Szenario vor dem Rettungseinsatz bei vorhandener Internetverbindung auf dem Server installiert, sodass den Rettungskräften im Einsatzgebiet ohne Internetverbindung schon vor Beginn des Rettungseinsatzes Karten und Luftbilder von einem früheren Zeitpunkt vor der Katastrophe zur Verfügung stehen. Während des Einsatzes kann es dann möglich sein, neu gewonnene Kartendaten, wie z.B. geo-referenzierte Luftbilder, die aus den Aufnahmen der UAVs gewonnen wurden, den Datensätzen des zentralen Karten-Servers hinzuzufügen, sodass diese neu gewonnenen Daten allen Einsatzkräften und Komponenten im System zur Verfügung stehen.

4.2 Ausblick

In diesem Abschnitt werden Erweiterungen vorgestellt, die bei der Entwicklung und Benutzung der App wünschenswert erschienen, jedoch leider nicht mehr im Rahmen dieser Arbeit umgesetzt werden konnten. Sie geben einen Ausblick auf weitere Möglichkeiten zur Nutzung dieser App.

4.2.1 Neue Assistenten

Nach aktuellem Stand bietet die App zwei Software-Assistenten für das Erstellen von Explorationsaufgaben an. Der eine ermöglicht es, kreisförmige Flugbahnen um Objekte herum zu planen. Der andere bietet die Erstellung von mäanderförmigen Flugbahnen über frei zu definierende Gebiete, die durch ein Polygon beschrieben werden können. Das Interessante an der App ist die Architektur, die eine einfache Entwicklung weiterer Assistenten ermöglicht. Die zwei vorhandenen Assistenten sind als Beispiele anzusehen für den Aufbau des Konzeptes. Um aus der jetzigen App eine leistungsfähige zu designen, die in der Funktionalität zu kommerziellen Alternativen konkurrenzfähig wird, ist es notwendig, die Fähigkeiten deutlich mit weiteren Assistenten auszubauen.

4.2.2 Assistent zum Erstellen von Übersichtsbildern

Sehr nützlich im Rettungseinsatz wäre es, wenn das UAV autonom Übersichtsbilder von einem festgelegten Gebiet aufnimmt. Ein Assistent, um solche Aufgaben zu erstellen, fehlte leider bis zuletzt in der App. Dieser würde ähnlich wie der „Area Task Assistent“ zuerst den Benutzer auffordern ein Gebiet festzulegen. Dies könnte ebenfalls durch das Spezifizieren eines Polygones gelöst werden, deren Inhalt das Gebiet repräsentiert. Des Weiteren ist es möglich, mehrere Höhen für die Aufnahmen festzulegen. Die Bestimmung der Positionen, von denen aus die Übersichtsbilder mit nach unten ausgerichteter Kamera auf der jeweiligen Ebene getätigt werden, geschieht automatisch. Diese werden so angeordnet, dass möglichst wenige Bilder ausreichen. Im praktischen Einsatz könnte dies ein Übersichtsbild sein, das mittig des Gebietes von großer Höhe aus aufgenommen wird.

4.2.3 Erweiterung der Einstellungsmöglichkeiten

Ein großes Defizit der TRADR UAV App in ihrem aktuellen Entwicklungsstand sind die mangelnden Einstellungsmöglichkeiten zum UAV, zum Flugverhalten und zur Kamera. Das DJI Mobile SDK[19] bietet sehr viele Konfigurationsmöglichkeiten für die meisten Komponenten des UAVs, sodass mit Hilfe des SDKs vom Prinzip fast alle Aspekte, die sich in der Hersteller App (DJI Go)[12] einstellen lassen, auch mit dem SDK konfiguriert werden können. Für diese Einstel-

lungsmöglichkeiten müssten Benutzeroberflächen-Dialoge geschaffen werden. Dies ist sehr zeit- und arbeitsintensiv, weshalb sie in der aktuellen Version der App weggelassen wurden. Jedoch dürfte eine solche Implementierung keine größeren technischen Probleme mit sich bringen. Daher wäre es sehr empfehlenswert, weitere Einstellungsmöglichkeiten der App hinzuzufügen. Dies würde die Nutzbarkeit vor allem im Vergleich zu anderen Apps stark verbessern.

4.2.4 Nutzung der DJI UI Library

DJI bietet neben dem Mobile SDK zur Entwicklung einer eigenen App zusätzlich eine UI Library[18] an. Diese Software Bibliothek enthält eine Menge an Steuerelementen, die aus den offiziellen Hersteller Apps bekannt sind. Die Steuerelemente ermöglichen zum einen die Darstellung vieler Werte auf einer ansprechenden Art, zum anderen bieten sie die Möglichkeit, eine große Menge an Parametern zu konfigurieren. Die App um diese Steuerelemente zu erweitern oder auf Basis dieser Elemente neu zu implementieren, ermöglicht es den Funktionsumfang extrem auszubauen.

4.2.5 Livebild im Assistenten zum Einstellen des Nick-Winkels

Ein entscheidender Parameter der Foto-Aktionen innerhalb der Explorationsaufgaben ist der Nick-Winkel der Kamera-Ausrichtung. Bisher kann dieser über Schieberegler in den Assistenten festgelegt werden. Dabei wird auf Erfahrungswerte und das Vorstellungsvermögen des Piloten spekuliert. Dieses Vorgehen erscheint sehr unzureichend, da der Pilot vom Boden aus einschätzen muss, was im Bildausschnitt der UAV-Kamera zu sehen sein wird, wenn diese sich in einer bestimmten Flughöhe und Ausrichtung befindet. Diese Einschätzung ist als sehr schwer anzusehen, da sowohl die Objektgröße, Flughöhe, Abstand zum Objekt und der Nick-Winkel in einem guten Verhältnis zueinander gewählt werden müssen.

Deutlich besser wäre es, wenn der Pilot diese Entscheidungen nicht blind treffen müsste, sondern, während er sich auf dem Smartphone im Assistenten-Dialog befindet, manuell das UAV auf eine gewisse Position vor dem zu inspizierenden Objekt fliegt und den Nick-Winkel korrekt einstellt. Der Abstand zum Objekt, der Nick-Winkel und die Flughöhe könnten dann direkt aus den Telemetriedaten des fliegen UAVs ermittelt werden. Mittels diesen Parametern könnte der Assistent wie gewohnt die gesamte Flugbahn entwerfen.

4.2.6 Speichern und Laden von erstellten Flugbahnen

Wie bereits beschrieben, bietet die App die Möglichkeit mit Hilfe verschiedener Assistenten Explorationsaufgaben zu planen. Aktuell ist es nach dem Planungsvorgang einer solchen Aufgabe nur möglich, diese durch das UAV ausführen zu lassen. Eine nützliche Erweiterung der App wäre es, die geplanten Aufgaben auch als Datei auf dem Smartphone zu speichern, um sie zu einem späteren Zeitpunkt öffnen und vom UAV ausführen lassen zu können.

4.2.7 Synchronisation von Explorationsaufgaben zwischen App und Planungssoftware

Aufgrund des gleichen internen Formats der Explorationsaufgaben, die mittels der App erstellt werden und derer, die mit der Planungssoftware im Leitstand erzeugt werden, wäre es denkbar, dass sowohl die App als auch die Planungssoftware im Leitstand erstellte Explorationsaufgaben in Dateien speichern könnten. Die Ordner mit den Dateien auf den verschiedenen Geräten würden automatisch per Datei-Synchronisation abgeglichen. Somit hätte der UAV Operator im Leitstand und der UAV Pilot im Einsatzgebiet gleichermaßen die geplanten Aufgaben zur Verfügung, unabhängig davon, wer diese erstellt hat.

4.2.8 Synchronisation von Bilddateien zwischen APP und TRADR System

Während das UAV fliegt und eine Explorationsaufgabe ausführt, werden mit der Kamera an den in der Aufgabe definierten Stellen Aufnahmen gemacht. Die Bilddaten können direkt nach der Aufnahme, während sich das UAV noch in Luft befindet, zur App gesandt und von dieser empfangen werden. Aktuell ist diese Funktion der Übertragung der aufgenommenen Bilddateien während des Fluges in der App implementiert. Die Funktion wurde aber deaktiviert, da ihre Verwendung zu einer deutlichen Verschlechterung der Livebild-Übertragung führte. Wird diese Funktion benutzt, empfängt die App, noch während das UAV die Aufgabe ausführt, die Bilddateien. Eine interessante Erweiterung des Funktionsumfanges wäre es, diese empfangenen Bilddateien in einen Ordner auf dem Smartphone zu speichern und diesen mit einem Verzeichnis auf dem TRADR Core zu synchronisieren. Somit würden die Bilder, noch während die Mission läuft, beim Datenanalysten eingehen.

4.2.9 Individuelles Erstellen und Bearbeiten von Flugbahnen

Flugbahnen von Explorationsaufgaben, die im Leitstand oder auf der App erstellt wurden, können aktuell nicht individuell angepasst werden. Mit individueller Anpassung ist gemeint, dass man einzelne Wegpunkte verschieben und deren Flughöhe ändern kann sowie vorhandene Wegpunkte der Route entnehmen und neue an beliebigen Stellen der Route hinzufügen kann. Solche Features zur individuellen Erstellung und Bearbeitung von Flugbahnen wären eine nützliche Erweiterung des Funktionsumfangs der App.

4.2.10 Speichern von Log-Daten auf dem Smartphone

Die Smartphone App empfängt im Einsatz dauerhaft alle Telemetrie-Daten des UAVs. Ein Teil dieser Daten wird direkt über das TRADR UAV Interface an das TRADR System weitergegeben. Eine Erweiterung der Fähigkeiten der App wäre es, diese Daten zusätzlich in einer für den Menschen lesbaren Form (z.B. GPX[82]) in eine Log-Datei zu schreiben, die sich auf dem Smartphone befindet. Nach jedem Start der App könnte eine neue Datei angelegt werden, die Datum und Uhrzeit als Angaben im Dateinamen enthält. Beginn und Ende autonomer Flüge könnten durch spezielle Schlüsselwörter innerhalb der Logdaten kenntlich gemacht werden.

A Anhang

A.1 Ressourcen auf DVD

Dieser Bachelorarbeit ist eine DVD mit allen zugehörigen Ressourcen beigefügt.

Enthalten sind:

- **ROS-Package der TRADR UAV Bridge**
  Das catkin-Paket beinhaltet alle Quelltexte und Buildfiles.

- **Android-Studio Projekt der TRADR UAV App**
  Das Projektverzeichnis beinhaltet alle Quelltexte, Grafiken, Buildfiles, Projektdateien und notwendige Bibliotheken (DJI Mobile SDK, ProtoBuff, usw.). Nach dem Öffnen dieses Projektverzeichnisses mit der Software Android-Studio sollte die App direkt ohne weitere Schritte zu bearbeiten und kompilieren sein.

- **Protocol Buffer Spezifikation des TRADR UAV Interface**
  Alle IDL-Quelltexte (*.proto) und Dokumentationen zum Generieren der ProtoBuff-Nachrichtentypen und Bibliotheken für Bridge und App.

- **ROS-Packages des ROS Interface zwischen Bridge und OCU-Komponenten**
  Die zwei catkin-Pakete beinhalten die IDL-Quelltexte (*.msg, *.srv) und Dokumentationen für alle ROS-Messages und ROS-Services, die zum Datenaustausch zwischen der TRADR UAV Bridge und den OCU-Programmen TRADR UAV Path und TRADR UAV FPV benötigt werden.

- **ROS-Packages der UAV-OCU-Komponenten des Leitstandes**
  Die zwei catkin-Pakete beinhalten alle Quelltexte und Buildfiles für die Softwarekomponenten TRADR UAV Path (von Gerhard Senkowski) und TRADR UAV FPV (von Alexander Schmitz).

- **Technische Dokumentation**
  Es handelt sich um eine Sammlung von im Entwicklungszeitraum erstellten Dokumenten, die die technischen Abläufe der oben genannten Software-Komponenten durch Texte und Zeichnungen erklären.

- **kompletter TRADR Workspace**
  Es handelt sich um den kompletten TRADR Workspace mit allen TRADR Datenbanken und TRADR Software-Komponenten (Quelltexte) inklusive der integrierten UAV-Komponenten (TRADR UAV Bridge, TRADR UAV Path, TRADR UAV FPV) zur Inbetriebnahme des in der Arbeit beschriebenen TRADR Systems (für Experten).

Die Inhalte der DVD sowie die Bachelorarbeit stehen auch im MediaWiki des Fachgebietes Autonome Robotik der Westfälischen Hochschule zum Download bereit:

`https://roblab.informatik.w-hs.de/mediawiki/index.php/Integration_von_kommerziellen_Flugdrohnen_in_eine_Leitstandsinfrastruktur_f%C3%BCr_Rettungseins%C3%A4tze`

Literatur

[1] Pedro Albuquerque. *MAVLink Commands Documentation*. 2016. URL: http://ardupilot.org/dev/docs/mavlink-commands.html (besucht am 28. 03. 2018).

[2] Pedro Albuquerque. *MAVLink Step by Step*. Juni 2016. URL: https://discuss.ardupilot.org/t/mavlink-step-by-step/9629 (besucht am 28. 03. 2018).

[3] Augusto Luis Ballardini. *What is a nodelet?* URL: https://answers.ros.org/question/230972/what-is-a-nodelet/ (besucht am 14. 04. 2018).

[4] Eli Bendersky. *Length-prefix framing for protocol buffers*. Aug. 2011. URL: https://eli.thegreenplace.net/2011/08/02/length-prefix-framing-for-protocol-buffers (besucht am 17. 04. 2018).

[5] Yen-Jen Chen, Yang-Jen Lin und Sheau-Ling Hsieh. "Analysis of Video Quality Variation with Different Bit Rates of H.264 Compression". In: 04 (Jan. 2016), S. 32–40.

[6] Stephen Cleary. *TCP/IP Protocol Design: Message Framing*. Juni 2009. URL: https://www.codeproject.com/Articles/37496/TCP-IP-Protocol-Design-Message-Framing (besucht am 17. 04. 2018).

[7] Jon Cole. *Simple Message Framing Sample for TCP Socket*. März 2006. URL: https://blogs.msdn.microsoft.com/joncole/2006/03/20/simple-message-framing-sample-for-tcp-socket/ (besucht am 17. 04. 2018).

[8] Panasonic Business Division. *H.264 Compression Technology*. Juli 2013. URL: https://www.youtube.com/watch?v=PmoEsPWEdOA (besucht am 28. 04. 2018).

[9] DJI. *Android Video Stream Decoding Sample*. URL: https://developer.dji.com/mobile-sdk/documentation/sample-code/index.html (besucht am 28. 04. 2018).

[10] DJI. *Company Website*. URL: https://www.dji.com (besucht am 26. 03. 2018).

[11] DJI. *Enterprise Products Website*. URL: https://enterprise.dji.com/ (besucht am 26. 03. 2018).

[12] DJI. *GoApp Website*. URL: https://www.dji.com/goapp (besucht am 26. 03. 2018).

[13] DJI. *Mainfold Onboard Computer Website*. URL: https://www.dji.com/manifold (besucht am 26. 03. 2018).

[14] DJI. *Matrice 600 Pro Product Website*. URL: https://www.dji.com/de/matrice600-pro (besucht am 26. 03. 2018).

[15] DJI. *Mavic Pro Product Website*. URL: https://www.dji.com/de/mavic (besucht am 26. 03. 2018).

[16] DJI. *Mobile SDK - API Documentation*. URL: https://developer.dji.com/api-reference/android-api/Components/SDKManager/DJISDKManager.html (besucht am 26. 03. 2018).

[17] DJI. *Mobile SDK - Documentation and Tutorials*. URL: https://developer.dji.com/mobile-sdk/documentation/introduction/index.html (besucht am 26. 03. 2018).

[18] DJI. *Mobile SDK - UX Library*. URL: https://developer.dji.com/mobile-sdk/documentation/introduction/ux_sdk_introduction.html (besucht am 26. 03. 2018).

[19] DJI. *Mobile SDK Website*. URL: https://developer.dji.com/mobile-sdk/ (besucht am 26. 03. 2018).

[20] DJI. *Onboard SDK Website*. URL: https://developer.dji.com/onboard-sdk/ (besucht am 26. 03. 2018).

[21] DJI. *Phantom 3 Professional Product Website*. URL: https://www.dji.com/de/phantom-3-pro (besucht am 26. 03. 2018).

[22] DJI. *Phantom 4 Pro Product Website*. URL: https://www.dji.com/de/phantom-4-pro (besucht am 26. 03. 2018).

[23] DJI Developer Forum. *Android Live Streaming H264 video from Mobile App*. URL: http://forum.dev.dji.com/thread-33280-1-1.html (besucht am 28. 04. 2018).

[24] DJI Developer Forum. *Differences in raw iframes in SDK ressources*. URL: http://forum.dev.dji.com/thread-32331-1-1.html (besucht am 28. 04. 2018).

[25] DJI Developer Forum. *Missing Photos*. URL: https://forum.dji.com/thread-119025-1-1.html (besucht am 30. 04. 2018).

[26] Open Sorce Robotic Foundation. *ROS Compressed Image Transport*. URL: http://wiki.ros.org/compressed_image_transport (besucht am 28. 04. 2018).

[27] Open Source Robotics Foundation. *ROS Documentation*. URL: http://wiki.ros.org/ (besucht am 14. 04. 2018).

[28] Open Source Robotics Foundation. *ROS: Robotic Operating System*. URL: http://www.ros.org/ (besucht am 08. 04. 2018).

[29] Luigi Freda. *TRADR Team and Vigili del Fuoco in Amatrice*. URL: http://www.luigifreda.com/2016/09/06/tradr-team-vigili-del-fuoco-amatrice/ (besucht am 03. 05. 2018).

[30] Google. *Google Maps Android API*. URL: https://developers.google.com/maps/documentation/android-api/?hl=de (besucht am 28. 04. 2018).

[31] Google. *Google Maps Android Widget*. URL: https://developers.google.com/android/reference/com/google/android/gms/maps/package-summary?hl=de (besucht am 28. 04. 2018).

[32] Google. *Protocol Buffers Language Guide*. URL: https://developers.google.com/protocol-buffers/docs/proto3 (besucht am 21. 04. 2018).

[33] Google. *Protocol Buffers Project Website*. URL: https://developers.google.com/protocol-buffers/ (besucht am 21. 04. 2018).

[34] Google. *Protocol Buffers Streaming Multiple Messages Techniques*. URL: https://developers.google.com/protocol-buffers/docs/techniques (besucht am 17. 04. 2018).

[35] Joachim de Greeff u. a. "Human-Robot Teamwork in USAR Environments: The TRADR Project". In: *Proceedings of the Tenth Annual ACM/IEEE International Conference on Human-Robot Interaction Extended Abstracts*. HRI'15 Extended Abstracts. Portland, Oregon, USA: ACM, 2015, S. 151–152. ISBN: 978-1-4503-3318-4. DOI: 10.1145/2701973.2702031. URL: http://doi.acm.org/10.1145/2701973.2702031.

[36] Greg Innes. *Using H.264 video compression in IP video surveillance systems*. Apr. 2009. URL: https://www.networkwebcams.com/ip-camera-learning-center/2009/04/03/using-h264-video-compression-in-ip-video-surveillance-systems/ (besucht am 28. 04. 2018).

[37] GitHub Issue. *Can i get ts-frame from DJI SDK camera?* URL: https://github.com/dji-sdk/Mobile-SDK-iOS/issues/80 (besucht am 28. 04. 2018).

[38] GitHub Issue. *Missing photos on waypoint mission*. URL: https://github.com/dji-sdk/Mobile-SDK-Android/issues/184 (besucht am 30. 04. 2018).

[39] GitHub Issue. *Please provide RTMP video streaming*. URL: https://github.com/dji-sdk/Mobile-SDK-iOS/issues/77 (besucht am 28. 04. 2018).

[40] Suresh Joshi. *You Stream, I Stream, We All Stream For Protocol Buffers*. Apr. 2016. URL: http://www.sureshjoshi.com/development/streaming-protocol-buffers/ (besucht am 17. 04. 2018).

[41] Ivana Kruijff-Korbayová u. a. "Deployment of Ground and Aerial Robots in Earthquake-Struck Amatrice in Italy (brief report)". In: *Proceedings of the 2016 IEEE International Symposium on Safety, Security and Rescue Robotics*. Hrsg. von Kamilo Melo. IEEE. IEEE, Okt. 2016, S. 278–279.

[42] G.J.M. Kruijff u. a. "Designing, developing, and deploying systems to support human–robot teams in disaster response". In: *Advanced Robotics* 28.23 (2014), S. 1547–1570. DOI: 10.1080/01691864.2014.985335. eprint: https://doi.org/10.1080/01691864.2014.985335. URL: https://doi.org/10.1080/01691864.2014.985335.

[43] Artur Leinweber. *DJI Mobile SDK - Functionality, Usability and User Experience*. URL: https://roblab.informatik.w-hs.de/mediawiki/index.php/DJI_Mobile_SDK_-_Functionality,_usability,_and_user_experience (besucht am 26. 03. 2018).

[44] Sara Mahmoud, Nader Mohamed und Jameela Al-Jaroodi. "Integrating UAVs into the Cloud Using the Concept of the Web of Things". In: *J. Robot.* 2015 (Jan. 2015), 10:10–10:10. ISSN: 1687-9600. DOI: 10.1155/2015/631420. URL: https://doi.org/10.1155/2015/631420.

[45] Helmut Manthey. *Kompressionsverfahren MJPEG vs. H.264*. URL: http://www.die-netzkamera.de/kompressionsverfahren.html (besucht am 28. 04. 2018).

[46] Lorenz Meier. *MAVLink Developer Guide*. Drone Project Inc. 2018. URL: https://mavlink.io/en/ (besucht am 28. 03. 2018).

[47] Lorenz Meier. *MAVLink Reference Protocols*. Drone Project Inc. 2018. URL: https://mavlink.io/en/protocol/overview.html (besucht am 28. 03. 2018).

[48] MikroKopter. *Products Website*. URL: http://www.mikrokopter.de/en/home (besucht am 29. 03. 2018).

[49] Motion Imagery Standards Board. *EG 0802 - H.264 / AVC Coding and Multiplexing*. Techn. Ber. Mai 2009, S. 19. URL: http://www.gwg.nga.mil/misb/docs/eg/EG0802.pdf.

[50] Motion Imagery Standards Board. *EG 0802 - H.264 / AVC Coding and Multiplexing*. Techn. Ber. Mai 2009, S. 11. URL: http://www.gwg.nga.mil/misb/docs/eg/EG0802.pdf.

[51] Video Lan Organisation. *libVLC Wiki Article*. URL: https://wiki.videolan.org/LibVLC/ (besucht am 25. 04. 2018).

[52] Video Lan Organisation. *libVLC, the VLC SDK*. URL: https://www.videolan.org/vlc/libvlc.html (besucht am 25. 04. 2018).

[53] TRADR Project. *TRADR YouTube Channel*. URL: https://www.youtube.com/channel/UCvrPijjrCFGWdJIHkKbFK-A (besucht am 28. 03. 2018).

[54] TRADR Project. *TRADR: Long-Term Human-Robot Teaming for Disaster Response*. URL: https://www.tradr-project.eu (besucht am 28. 03. 2018).

[55] MAVLink Micro Air Vehicle Communication Protocol. *MAVLINK Common Message Set*. QGroundControl. URL: http://mavlink.org/messages/common (besucht am 28. 03. 2018).

[56] MAVLink Micro Air Vehicle Communication Protocol. *Project Website*. QGroundControl. URL: http://qgroundcontrol.org/mavlink/start (besucht am 28. 03. 2018).

[57] Tiliam Research. *Effective use of Long GOP Video Codecs*. URL: http://www.tiliam.com/Blog/2015/07/06/effective-use-long-gop-video-codecs (besucht am 28. 04. 2018).

[58] Daniel Reuter. *ROS libvlc wrapper*. URL: https://github.com/dreuter/ros_libvlc (besucht am 25. 04. 2018).

[59] Alexander Schmitz. *Kommunikationsschnittstelle zwischen UAVs und GCS*. URL: https://roblab.informatik.w-hs.de/mediawiki/index.php/Kommunikationsschnittstelle_zwischen_UAVs_und_GCS (besucht am 21. 04. 2018).

[60] Alexander Schmitz. *Protocol Buffer (ProtoBuf)*. URL: https://roblab.informatik.w-hs.de/mediawiki/index.php/Kommunikationsschnittstelle_zwischen_UAVs_und_GCS_-_Seminararbeit#Protocol_Buffer_.28ProtoBuf.29 (besucht am 21. 04. 2018).

[61] Gerhard Senkowski. *Erzeugung von GPS Pfaden*. URL: https://roblab.informatik.w-hs.de/mediawiki/index.php/Erzeugung_von_GPS_Pfaden (besucht am 08. 05. 2018).

[62] Gerhard Senkowski. *UAV MapViz Plugin*. URL: https://roblab.informatik.w-hs.de/mediawiki/index.php/Erzeugung_von_GPS_Pfaden_T%C3%A4tigkeiten#UAV_Path_Mapviz_Plugin (besucht am 08. 05. 2018).

[63] G. J. Sullivan und T. Wiegand. "Video Compression - From Concepts to the H.264/AVC Standard". In: *Proceedings of the IEEE* 93.1 (Jan. 2005), S. 18–31. ISSN: 0018-9219. DOI: 10.1109/JPROC.2004.839617.

[64] Hartmut Surmann. *Amatrice: A video diary of the 3D reconstruction of two churches*. Hrsg. von Hartmut Surmann. Youtube. Sep. 2016. URL: https://www.youtube.com/watch?v=CzTj_jK_an0 (besucht am 08. 04. 2018).

[65] Hartmut Surmann. *Fantastic drone flight: Flying out of Sant'Agostino, Amatrice*. Hrsg. von Hartmut Surmann. Youtube. Dez. 2016. URL: https://www.youtube.com/watch?v=LluhlkKWsfs (besucht am 08. 04. 2018).

[66] Hartmut Surmann. *Robots used for disaster response in Amatrice: The TRADR mission. A video diary*. Hrsg. von Hartmut Surmann. Youtube. Sep. 2016. URL: https://www.youtube.com/watch?v=DHCQeBPAzQQ (besucht am 08. 04. 2018).

[67] Hartmut Surmann, Gerhard Senkowski und Alexander Schmitz. *Integrated Flight System*. Hrsg. von Hartmut Surmann. Youtube. URL: https://www.youtube.com/watch?v=5cnGbowtsrY (besucht am 08. 04. 2018).

[68] Hartmut Surmann, Rainer Worst und Erik Zimmermann. *Amatrice: The TRADR Mission*. Hrsg. von Fraunhofer IAIS. Youtube. Jan. 2017. URL: https://www.youtube.com/watch?v=fFeO-dM1RFA (besucht am 28. 03. 2018).

[69] Smart Robotic Systems. *Company Website*. URL: http://www.smartrobotics.de/ (besucht am 26. 03. 2018).

[70] Smart Robotic Systems. *SKIDER Product Website*. URL: http://www.smartrobotics.de/skider/index.html (besucht am 26. 03. 2018).

[71] Ascending Technologies. *Company Website*. URL: http://www.asctec.de (besucht am 26. 03. 2018).

[72] Ascending Technologies. *Falcon 8 Product Website*. URL: http://www.asctec.de/uav-uas-drohnen-flugsysteme/asctec-falcon-8/ (besucht am 26. 03. 2018).

[73] Ascending Technologies. *Neo Product Website*. URL: http://www.asctec.de/uav-uas-drohnen-flugsysteme/asctec-neo/ (besucht am 26. 03. 2018).

[74] UGCS. *Company Website*. URL: https://www.ugcs.com/ (besucht am 26. 03. 2018).

[75] UGCS. *Phantom 3 Ground Station Support Website*. URL: https://www.ugcs.com/en/supported_drones_autopilots#phantom-3-ground-station-software (besucht am 26. 03. 2018).

[76] UGCS. *Phantom 4 Ground Station Support Website*. URL: https://www.ugcs.com/en/supported_drones_autopilots#phantom-4-software (besucht am 26. 03. 2018).

[77] UGCS. *UGCS For Control Centre Website*. URL: https://www.ugcs.com/en/page/ugcs-for-command-centre (besucht am 26. 03. 2018).

[78] AIS Universität Bonn. *Nimbro Network*. URL: https://github.com/AIS-Bonn/nimbro_network (besucht am 14. 04. 2018).

[79] Wikipedia. *Bildergruppe*. URL: https://de.wikipedia.org/wiki/Bildergruppe (besucht am 28. 04. 2018).

[80] Wikipedia. *Globally Unique Identifier*. URL: https://de.wikipedia.org/wiki/Globally_Unique_Identifier (besucht am 21. 04. 2018).

[81] Wikipedia. *Google Maps*. URL: https://de.wikipedia.org/wiki/Google_Maps (besucht am 28. 04. 2018).

[82] Wikipedia. *GPS Exchange Format*. URL: https://de.wikipedia.org/wiki/GPS_Exchange_Format (besucht am 28. 04. 2018).

[83] Wikipedia. *H.264 - MPEG Profile*. URL: https://de.wikipedia.org/wiki/H.264#Profile (besucht am 28. 04. 2018).

[84] Wikipedia. *H.264 / MPEG-4 AVC*. URL: https://de.wikipedia.org/wiki/H.264 (besucht am 28. 04. 2018).

[85] Wikipedia. *Heartbeat*. URL: https://de.wikipedia.org/wiki/Heartbeat_(Informatik) (besucht am 18. 04. 2018).

[86] Wikipedia. *JPEG*. URL: https://de.wikipedia.org/wiki/JPEG (besucht am 28. 04. 2018).

[87] Wikipedia. *Kardanische Aufhängung*. URL: https://de.wikipedia.org/wiki/Kardanische_Aufh%C3%A4ngung (besucht am 26. 03. 2018).

[88] Wikipedia. *Map-Server*. URL: https://de.wikipedia.org/wiki/Mapserver (besucht am 28. 04. 2018).

[89] Wikipedia. *Marshalling*. URL: https://de.wikipedia.org/wiki/Marshalling (besucht am 21. 04. 2018).

[90] Wikipedia. *MAVLink*. URL: https://en.wikipedia.org/wiki/MAVLink (besucht am 28. 03. 2018).

[91] Wikipedia. *Motion JPEG*. URL: https://de.wikipedia.org/wiki/Motion_JPEG (besucht am 28. 04. 2018).

[92] Wikipedia. *MPEG Transportstrom*. URL: https://de.wikipedia.org/wiki/MPEG-Transportstrom (besucht am 28. 04. 2018).

[93] Wikipedia. *Real Time Messaging Protocol*. URL: https://de.wikipedia.org/wiki/Real_Time_Messaging_Protocol (besucht am 28. 04. 2018).

[94] Wikipedia. *Real Time Streaming Protocol*. URL: https://de.wikipedia.org/wiki/Real-Time_Streaming_Protocol (besucht am 28. 04. 2018).

[95] Wikipedia. *Serialisierung*. URL: https://de.wikipedia.org/wiki/Serialisierung (besucht am 21. 04. 2018).

[96] Wikipedia. *TCP Verbindungsabbau*. URL: https://de.wikipedia.org/wiki/Transmission_Control_Protocol#Verbindungsabbau (besucht am 17. 04. 2018).

[97] Wikipedia. *Transmission Control Protocol*. URL: https://de.wikipedia.org/wiki/Transmission_Control_Protocol (besucht am 17. 04. 2018).

[98] Wikipedia. *User Datagram Protocol*. URL: https://de.wikipedia.org/wiki/User_Datagram_Protocol (besucht am 17. 04. 2018).

Abbildungsverzeichnis

Tabellenverzeichnis