

Universität der Bundeswehr München
Fakultät für Luft- und Raumfahrttechnik
Institut für Flugsysteme

**WISSENSBASIERTE KONFIGURATION EINES
UNBEMANNTEN FLUGGERÄTES ALS ARCHITEKTURANSATZ
ZUR KOGNITIVEN FLUGFÜHRUNG**

Dipl.-Ing. Michael Kriegel

Vollständiger Abdruck der bei der
Fakultät für Luft- und Raumfahrttechnik
der Universität der Bundeswehr München
zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs (Dr.-Ing.)

eingereichten Dissertation

Vorsitzender:	Prof. Dr.-Ing. Kristin Paetzold
1. Berichterstatter:	Prof. Dr.-Ing. Axel Schulte
2. Berichterstatter:	Prof. Dr.-Ing. Florian Holzapfel

Diese Dissertation wurde am 24. Oktober 2011 bei der Universität der Bundeswehr München, 85577 Neubiberg, eingereicht und durch die Fakultät für Luft- und Raumfahrttechnik am 26. Oktober 2011 angenommen.

Tag der Prüfung: 07. März 2012

Vorwort

Die vorliegende Arbeit entstand im Rahmen meiner Tätigkeit als wissenschaftlicher Mitarbeiter am Institut für Flugsysteme der Fakultät für Luft- und Raumfahrttechnik an der Universität der Bundeswehr München.

Ganz herzlich möchte ich Herrn Prof. Dr.-Ing. Axel Schulte danken, der während meiner Zeit am Institut fortwährend an mich geglaubt hat und es mir ermöglichte, mich frei von äußerem Projektdruck, bei dem Aufbau von einigen UAV-Demonstratoren und meiner wissenschaftlichen Arbeit zu entfalten. Seine beständige Unterstützung und Motivation, sowohl fachlich als auch menschlich, haben mich immer wieder auf den richtigen Weg gebracht und letztlich auch diese Arbeit ermöglicht. Auch will ich ihm danken für die Möglichkeiten, meine Arbeiten auf internationalen Konferenzen vorzustellen, von denen ich viele Erfahrungen und Eindrücke mit nach Hause bringen konnte.

Ganz herzlich bedanken möchte ich mich auch bei Prof. Dr.-Ing. Florian Holzapfel für die Übernahme des Koreferats und seine fachliche Kompetenz und Unterstützung während der Entwicklungsphase des Flugreglers und der anschließenden Versuchsflugphase zum Tunen der Reglerparameter.

Frau Anja Linke und meiner lieben Schwägerin Martina 'Miezi' Klepl danke ich für die sicherlich nicht einfache „Durchsicht mit Durchblick“ des Manuskripts und die Identifizierung meiner statistisch verteilten Schreib- und vor allem Kommafehler.

Allen meinen Kolleginnen und Kollegen, die nach und nach den Weg nach Neubiberg gefunden haben, möchte ich für die sehr gute Zusammenarbeit und die ab und an geleistete Schützenhilfe in Sachen Kernel-Kompilierung, CPL-Probleme, rauchende elektronische Bauteile und noch einigen weiteren Sachgebieten ganz herzlich danken. Es war und ist mir eine Ehre, Teil dieses tollen Teams zu sein, das für das beste Arbeitsklima gesorgt hat, was ich jemals erlebt habe.

Vielen Dank auch an die guten Seelen im Sekretariat, Madeleine Gabler und Gitta Hanson, für die Unterstützung bei organisatorischen Dingen und die ein oder andere „Möglichmachung“ von „Unmöglichem“.

Besonderer Dank geht auch an den Einsatz Werkstatt, also Eduard Österreicher, Richard Stömmer sowie den beiden nachfolgenden Mitarbeitern Peter Pratter und Alexander Gebhardt, für den unermüdlichen Einsatz meine Ideen umzusetzen und die wertvollen Beiträge und Anmerkungen, um die Theorien in die Praxis zu bringen.

Auch der Flugversuchcrew, bestehen aus dem unersetzbaren Sicherheitspiloten Peter Pratter, Alexander Gebhardt, dessen helfende Hände immer und überall gebraucht wurden, dem Feuerlöschfrosch Florian Böhm und dem Kameramann und „Mädchen“ für alles, Tobias Kloss möchte ich meinen herzlichen Dank für Ihren Einsatz und den an den Tag gelegten Enthusiasmus aussprechen.

Bei der Durchführung des Tunings des Autopiloten möchte ich mich ganz herzlich bei Leonhardt Höcht und Sebastian Colditz bedanken, die fachlich und auch persönlich viel Einsatz gezeigt und letztlich ein funktionierendes Vorgehen entwickelt haben, um die Reglerparameter im Fluge einzustellen. Für die Ausarbeitung von perfekten „Flight test cards“, die umfassende Dokumentierung jeder noch so kleinen Aktion und die Fürsorge, wenn sich die Arbeit mal bis in die späten Abendstunden zog, möchte ich auch Dorothea Prang ganz herzlich danken.

Allen voran möchte ich noch meiner Familie danken, meinem Vater, der mich mit seinen Fragen zu meiner Arbeit auf seine eigene Art motivierte und vor allem meiner lieben Frau Giulia, die mich während der Promotionszeit stets unterstützt und alle Höhen und Tiefen mit mir zusammen durchgestanden hat, obwohl sie selbst genug Stress im eigenen Job hatte.

Inhaltsverzeichnis

1	Einleitung	1
1.1	Überblick der Arbeit	3
2	Automation unbemannter Fluggeräte	5
2.1	State of the Art	5
2.1.1	UAV	7
2.1.2	Flugregelungssystem / Flugmanagementsystem	8
2.1.3	Sensoren	10
2.1.4	Funk-Kommunikation	12
2.1.5	Missionsnutzlast	12
2.1.6	Bedienung und Informationsdarstellung	14
2.1.7	Missionsmanagement	16
2.1.7.1	Common Operating System – COS	18
2.1.7.2	Auftragsbasierte Führung	18
2.1.7.3	Maschinelle Kooperation	20
2.2	Grenzen konventioneller Automation	20
2.3	Perspektiven durch künstliche Intelligenz (KI)	25
2.4	Kognitive Automation	27
2.4.1	Das Arbeitssystem	28
2.4.2	Kognitive Automation im Arbeitssystem	29
2.4.3	Der Kognitive Prozess	30
2.4.4	Kognitive Systemarchitektur – COSA	32
2.4.4.1	Weiterentwicklung - COSA ²	34
2.5	Problemstellung	37
2.5.1	Entwicklungsumfeld für einen intelligenten Agenten	38
2.5.2	Problemidentifikation	42
3	Knowledge configured vehicle – KCV	44
3.1	Analogien für die Entwicklung des Konzepts	44
3.1.1	Control Configured Vehicle – CCV	45
3.1.2	Integrierte Modulare Avionik / Betriebssysteme	46
3.1.3	Wissen eines Piloten	47
3.2	Konzept	48
3.3	Abstraktion und Interaktion der Wissenspakete nach dem KCV - Konzept ..	50
3.4	KCV Konzeptumsetzung mit dem Paradigma des Kognitiven Prozesses	53
4	Realisierung der Wissensmodelle	57
4.1	Serverkomponenten	58
4.2	Wissensmodellierung	58
4.2.1	Darstellung von situativem und a-priori Wissen	61
4.3	Wissenspakete	63
4.4	A-priori Wissen der KCV-Schicht	66
4.5	Maschinenspezifisches Wissen	67
4.5.1	System- / Softwarekomponenten	67
4.5.2	Aufbau eines Systemmodells im situativen Wissen	71
4.5.3	Abstraktion von Fähigkeiten	73
4.6	Missionsspezifisches Wissen	74

4.7	Anfordern von gewünschtem Verhalten	76
4.8	Zusammenfassung – Methodisch algorithmische Umsetzung	78
5	Systemtechnische Umsetzung und Integration.....	81
5.1	Anforderungen	81
5.2	Demonstratorumgebungskonzept.....	83
5.3	Simulationsumgebung	84
5.4	UAV – Demonstrator BIG I.....	86
5.4.1	Hard- und Softwareumsetzung der Avionik.....	87
5.4.1.1	Hardwareaufbau	88
5.4.1.2	Softwareumsetzung	89
5.4.2	Avionikkomponenten	91
5.4.2.1	Stromversorgung.....	91
5.4.2.2	FCC / MMC	93
5.4.2.3	IMU.....	94
5.4.2.4	Datenfunk.....	95
5.4.3	Mechanische Integration	96
5.4.4	Sicherheitskonzept und Umsetzung	98
5.4.5	Flugregelungssystem – FCS.....	102
5.4.5.1	FCS-Betriebsarten.....	104
5.4.5.2	Tuning und Einstellen der FCS-Parameter	108
5.5	Kontrollstation.....	111
5.5.1	Plattform.....	111
5.5.2	Hardwarekonzept	112
5.5.3	Softwarekonzept.....	114
5.5.3.1	BKS-Displays	115
5.6	Kommunikationsmanagement.....	117
5.6.1	Datenfunk-Protokoll.....	117
5.6.2	Implementierung	120
6	Evaluierung.....	125
6.1	Evaluierungsumgebung.....	125
6.1.1	Evaluierungsmission	125
6.1.2	HIL-Simulation	127
6.2	Evaluierung „Real World“	127
6.3	Evaluierung in der Simulation.....	137
6.3.1	Durchführung der Evaluationsmission in der Simulation	137
6.3.2	Evaluation des Anwendungsfalls “Einbruch der Avionikstromversorgung”	142
6.4	Fazit.....	144
7	Zusammenfassung & Ausblick	147
A	Literaturverzeichnis.....	150
B	Abbildungsverzeichnis	173
C	Abkürzungsverzeichnis.....	176

1 Einleitung

Der Bau und Einsatz von unbemannten Luftfahrzeugen (engl.: Uninhabited Aerial Vehicle – UAV) geht zurück bis auf den Anfang des 20. Jahrhunderts, als während des ersten Weltkrieges einerseits Zieldarstellungsdrohnen für die Ausbildung von Flugabwehrkanonieren und andererseits „fliegende Bomben“ entwickelt wurden. Nachdem Marconi 1896 erstmals eine Funkübertragung und 1903 der erste andauernde, gesteuerte Motorflug durch die Gebrüder Wright demonstriert wurden, waren die Voraussetzungen für den Bau von UAVs erfüllt [Pearson, 1969]. Weniger als 15 Jahre später flog das Army Signal Corps ihr erstes unbemanntes Fluggerät – die *Kettering Bug Flying Bomb*. Die Stabilisierung und Lenkung des Luftfahrzeugs übernahmen Gyroskope. Die Reichweite konnte mittels eines Zählwerkes festgelegt werden. Sobald die voreingestellte Umdrehungszahl des Propellers erreicht war, schaltete sich der Motor ab, die Befestigung der Tragflächen lösten sich und der Rumpf stürzte ballistisch, mit dem am Boden zündenden Sprengstoff ins Ziel [Sullivan, 2006].

Bereits 1911 erkannte Elmer Sperry, welcher neben Herman Anschütz-Kaempfe als einer der Erfinder von mechanischen Kreiselkompassen gilt, dass Automationsfunktionen nötig sind, um eine effektive, funkferngesteuerte (engl.: Radio-Controlled – RC) Führung von unbemannt fliegenden Systemen (engl.: Uninhabited Aerial System – UAS) zu realisieren.

{Sperry} realized at once that for radio control to be effective, automatic stabilization would be essential, so he again turned to the gyroscope as a promising device. [Pearson, 1969]

Durch den Einsatz von mechanischen Gyroskopen und Kreiselkompassen konnten erste einfache Flugregler und Autopilotenfunktionen realisiert werden, die jedoch im Vergleich zu heutigen Systemen nur einen Bruchteil der Funktionalität und Genauigkeit aufwiesen.

Heutzutage finden UAVs in vielen Bereichen Anwendung. Deren Spektrum erstreckt sich von der zivilen Nutzung – wie beispielsweise bei Such- und Rettungsmissionen, Waldbrandüberwachung, Luftbildfotographie über industrielle Überwachungsaufgaben im Bereich von nuklearen Anlagen oder Pipelines – bis hin zu militärischen Einsätzen, wie Aufklärung, Überwachung aber auch Kampfeinsätze. Gemessen am finanziellen Umsatz lag nach [Sarris, 2001] vor einer Dekade der zivile Anteil jedoch nur bei etwa drei Prozent.

Die Arten und Ausprägungen von unbemannten Fluggeräten sind dabei sehr divers. Sie zeigen sich – abhängig vom entsprechenden Einsatzspektrum – in verschiedensten Formen und Größen mit sich deutlich unterscheidenden Leistungsmerkmalen wie zum Beispiel die maximale Reichweite oder Flughöhe, welche in Tabelle 1 als Kriterien für

die Kategorisierung und Klassifizierung von UAVs nach [Bento, 2008] verwendet wurden.

Kategorie	Reichweite	Höhe
Handstartfähig	2 km	600 m
Nahbereich	10 km	1500 m
NATO Klasse	50 km	3000 m
Taktische UAVs	160 km	5500 m
MALE (medium altitude, long endurance)	> 200 km	< 9000 m
HALE (high altitude long endurance)	> 1500 km	> 9000 m

Tabelle 1: Klassifikation nach Reichweite und Höhe von UAVs nach [Bento, 2008]

Wie schon von Elmer Sperry erkannt, benötigen UAV-Systeme Automationsfunktionen wie Flugregler und Flugmanagementsysteme, welche die automatische Stabilisierung, Lageregelung, Bahnführung und Navigation des Fluggerätes übernehmen, um den Bediener von der Aufgabe der manuellen Steuerung zu entlasten. Eine solche *konventionelle* Automatisierung, die in modernen Systemen beispielsweise das selbständige Abfliegen von Wegpunktlisten erlaubt, bietet eine Fülle von Betriebsarten und Einstellmöglichkeiten. Dies bedeutet für den Bediener, dass er eine hochkomplexe Maschine handhaben, beherrschen und überwachen muss, die jedoch auf der anderen Seite über eine Vielzahl von Fähigkeiten verfügt. In diesem Zusammenhang wird häufig der Terminus „Autonomie“ verwendet, jedoch:

Most UAVs are not truly autonomous or even unmanned but are more correctly termed 'uninhabited' or 'remotely piloted'. [Karim & Heinze, 2005]

Durch den hohen Automationsgrad unbemannter Fluggeräte verbleibt während der Durchführung der Mission als Hauptaufgabe für den Operateur die Überwachung des UAVs und seiner Subsysteme. [Sheridan, 1992] beschreibt diese Art des Zusammenwirkens zwischen Mensch und Maschine als *supervisory control* (Kontrolle durch Überwachung).

In der Regel dient der Einsatz von UAVs – abgesehen von deren Anwendung in der Forschung mit dem Zweck der Entwicklung, Demonstration und Erprobung neuer Technologien – als Werkzeug, um eine gestellte Aufgabe oder Mission zu erfüllen. In diesem Zusammenhang fungiert *alleinig* der Mensch als Manager der Fähigkeiten eines Fluggerätes. Hierzu zählen einerseits der Gebrauch und die Bedienung der entsprechenden Missionssensoren oder Effektoren sowie das Führen des Fluggerätes durch Absetzen von Automationskommandos oder auch manuelle Steuereingaben. Angesichts der Tatsache, dass der Mensch sich bei UAVs jedoch nicht mehr an Bord des Luftfahrzeugs befindet, sondern in einer entfernten Kontrollstation, muss die Kommunikation durch eine Funkverbindung sichergestellt werden. Fällt dieses

Kommunikationsmittel jedoch aus (engl.: Datalink Loss) oder wird durch externe Beeinflussung gestört, so verfügen manche Systeme über automatisierte Verfahren, um die Funkverbindung wiederherzustellen, wie z.B. das Zurückkehren zur Basis. In einem solchen Fall stellt sich die Frage, ob es ethisch vertretbar ist, dass ein vermeintlich „intelligentes“ Fluggerät eigenverantwortlich mit der Erfüllung der Mission fortfährt oder lieber die Mission abbricht, da der Bediener nicht mehr in der Lage ist mit dem UAV zu interagieren oder dessen Aktionen zu überwachen.

Pannen und Flugunfälle zeigen, dass heutige, kommerzielle UAV-Systeme noch immer ein Stück weit davon entfernt sind als „intelligent“ oder „autonom“ bezeichnet zu werden. Dies belegt beispielsweise der Fall eines irischen UAVs, das aufgrund des Verlustes der Datenfunkverbindung (engl.: datalink-loss) bei einem Einsatz in Zentralafrika versuchte wieder nach Irland zurückzukehren [Evan Ackerman, 2009]. Diese Fehlleistung verdeutlicht, dass aktuelle, unbemannte Fluggeräte trotz ihres hohen Automatisierungsgrades *ohne den Menschen* nicht in der Lage sind, sich zielgerichtet im Sinne der Mission zu verhalten oder auf unbekannte Situationen rational zu reagieren.

1.1 Überblick der Arbeit

Kapitel 2 gibt zunächst einen Überblick über ein UAV Führungs- und Missionssystem, in dem die relevanten Bestandteile identifiziert, benannt und – in Ausschnitten anhand von derzeit käuflich verfügbaren Komponenten (engl: Commercial Of The Shelf – COTS) und auch Funktionalitäten, die Gegenstand aktueller Forschungsarbeiten sind – beschrieben werden. Nachfolgend werden einige Probleme erläutert, die bei dem Zusammenwirken von Mensch und konventionell automatisierten Maschinen entstehen, und es wird eine Möglichkeit aufgezeigt diesen Problemen durch den Einsatz von Künstlicher Intelligenz (KI) zu begegnen. Im Hinblick auf die Erzeugung rationalen, wissensbasierten Verhaltens von Maschinen werden nachfolgend einige Theorieansätze und verfügbare Architekturen vorgestellt. Das Kapitel schließt mit der Beschreibung einer sich daraus ergebenden Problemstellung, im Hinblick auf die Modularität und Wiederverwendbarkeit von Wissenspaketen solcher Systeme.

Kapitel 3 beschreibt anfangs einige Analogien, auf deren Basis das Konzept eines durch Wissen konfigurierten Fluggerätes (engl.: Knowledge Configured Vehicle – KCV) entwickelt wurde, das eine möglich Lösung des geschilderten Problems darstellt. Neben einer genauen Beschreibung des Konzeptes werden nachfolgend die Interaktion und die einzelnen Ebenen der Kommunikation zwischen den für die kognitive Flugführung eines UAVs benötigten Wissenspaketen spezifiziert. Abschließend wird der Ansatz und die Herangehensweise der Umsetzung dieses Konzepts auf der Grundlage des Paradigmas des Kognitiven Prozesses erläutert.

Kapitel 4 schildert die methodisch, algorithmische Umsetzung des Konzepts unter Verwendung der Kognitiven Systemarchitektur COSA (engl.: Cognitive System Architecture). Diese bietet ein Framework für die Entwicklung von kognitiven, wissensbasierten Systemen nach dem Paradigma des Kognitiven Prozesses. Eine

applikationsunabhängige Inferenzmaschine ermöglicht die Verarbeitung von modelliertem, objektorientiertem Anwendungswissen – formuliert in der Programmiersprache CPL – für die Umsetzung von Wissen in Verhalten. Schwerpunkt des Kapitels ist die Darstellung und Beschreibung der wichtigsten Wissensmodelle, sowie deren Interaktion.

Kapitel 5 stellt die UAV-Demonstratorumgebung vor, welche für die Entwicklung, Evaluierung und Funktionsdemonstration der implementierten *KCV-ACU* (engl.: ACU – Artificial Cognitive Unit, *Künstlichen Kognitiven Einheit*) realisiert wurde. Anfangs werden einige Anforderungen an die Experimentalumgebung, deren Komponenten und benötigte Funktionalitäten definiert und anschließend die Hauptbestandteile – nämlich die Simulationsumgebung, das unbemannte Fluggerät auf Basis eines turbinengetriebenen Modellhubschraubers und die mobile Bodenkontrollstation (BKS) – näher beschrieben. Neben den Hard- und Softwarekonzepten des UAVs und der BKS werden im weiteren Verlauf die entsprechenden Umsetzungen sowie Softwareimplementierungen erläutert. Das Kapitel schließt mit der Beschreibung eines proprietären, seriellen Protokolls, welches zur Sicherstellung der Datensicherheit und Integrität, im Rahmen der Funkübertragung von Informationen zwischen dem UAV und der BKS, entwickelt und realisiert wurde.

Kapitel 6 stellt die Ergebnisse der durchgeführten Flugversuche des in Kapitel 4 aufgezeigten Funktionsprototyps einer *KCV-ACU* dar. Das gewählte Anwendungsbeispiel für die Erprobung entspricht einer einfachen Aufklärungsmission, bei der eine Flugroute automatisch abgeflogen und in einem bestimmten Flugabschnitt ein erdfester Punkt überwacht werden soll. Nach der Vorstellung der Evaluierungsumgebung werden die Resultate eines Realflugs präsentiert, welcher unter Anwendung der in Kapitel 5 beschriebenen UAV-Demonstratorumgebung durchgeführt wurde. Nachfolgend werden noch Simulationsergebnisse präsentiert, die unter anderem einen Anwendungsfall und die von der *KCV-ACU* getroffenen Entscheidungen bei einem fingierten technischen Defekt der Avionikstromversorgung näher betrachten.

Abschließend fasst Kapitel 7 die Arbeit zusammen und gibt einen Ausblick über mögliche, zukünftige Arbeiten im Umfeld der wissensbasierten Flugführung auf Basis von künstlicher Kognition von unbemannten Fluggeräten.

2 Automation unbemannter Fluggeräte

Die Führung unbemannter Fluggeräte nach dem klassischen Paradigma der *supervisory control* setzt seinen Betrachtungsschwerpunkt auf die klar voneinander getrennte Aufgabenteilung zwischen Mensch und Maschine. Dies führt zu einer hierarchischen Beziehung, in welcher der Mensch als Hauptaufgabe die Überwachung der Automation übernimmt.

Im Folgenden wird anfangs das Mensch-UAV-Führungssystem kurz beschrieben und die dafür benötigten Komponenten identifiziert. An einigen Beispielen werden die technisch möglichen Umsetzungen präsentiert und zudem ein kleiner, ausschnittsweiser Überblick über die aktuellen Themen der Forschung gegeben (Abschnitt 2.1). Anschließend wird auf die Grenzen konventioneller Automation eingegangen (Abschnitt 2.2) und mögliche Perspektiven und Methoden aufgezeigt, um dieser Problematik zu begegnen (Abschnitt 2.3). Das Kapitel schließt mit einem Theorieansatz und der Beschreibung einer kognitiven Systemarchitektur (Abschnitt 2.4), welche für die Umsetzung eines Konzepts (vgl. Kapitel 3) verwendet wurde und eine mögliche Lösung für die Problemstellung darstellt, die in Abschnitt 2.5 präzisiert wird.

2.1 State of the Art

Ein unbemanntes Fluggerät wird meist durch einen bzw. mehrere Operateure, ähnlich einem bemannten Fluggerät, geführt – jedoch mit dem Unterschied, dass sich der UAV-Operator nicht mehr an Bord des Fluggerätes befindet, sondern in einem räumlich getrennten Kontrollsegment. Folglich ist eine Funkverbindung erforderlich, welche die Übertragung der benötigten Informationen zwischen der Kontrollstation (KS) und dem Fluggerät gewährleistet. Ferner entlasten Automationsfunktionen den Bediener von der Aufgabe der manuellen Steuerung und übernehmen die maschinelle Stabilisierung, Lageregelung, Bahnführung und Navigation des Fluggerätes.

Abbildung 2-1 zeigt den schematischen Aufbau des Kontroll- und des fliegenden Segments, in dem die gängigsten Komponenten, welche für die Führung eines unbemannten Fluggerätes benötigt werden, dargestellt sind – nämlich:

- UAV: Die Plattform des Fluggerätes, das sich in verschiedensten Konfigurationen und Größen manifestiert. Hierzu zählen der Airframe und die Grundsysteme.
- Flugregelungssystem (engl.: Flight Control System - FCS) / Flugmanagementsystem (engl.: Flight Management System - FMS): Automatische Stabilisierung, Bahnführung, sowie Navigation und Lenkung des unbemannten Fluggerätes.

- Sensoren: Umfasst sämtliche Sensorik, die für die Erfassung des Flugzustands verwendet wird. Hier sind Inertialsensorik, positionsbestimmende, abstandsmessende Sensoren sowie Sensoren zur Erhebung von Umgebungsbedingungen wie Temperatur und Druck aufzuführen.
- Funk-Kommunikation: Übertragung von Führungs-, Telemetrie- und Missionsnutzlastdaten zwischen dem Kontroll- und fliegenden Segment.
- Missionsnutzlast: Umfasst ausgewählte Sensoren wie Kameras – elektro-optisch oder infrarot – Radar (engl.: Radio Detection and Ranging), Lidar (engl.: Light Detection and Ranging), aber auch Effektoren, die für die Durchführung einer Mission oder für höherwertige Automationsfunktionen benötigt werden.
- Bedienung und Informationsdarstellung: Über eine Mensch-Maschine-Schnittstelle (engl.: Human machine interface – HMI) werden dem Bediener einerseits Daten des zu führenden Fluggerätes angezeigt, andererseits ermöglichen entsprechende Eingabegeräte Eingriffe des Operators zur Flugführung.
- Missionsmanagement (MM) / Autonomy enabling functions (AEF): Überwachung und/oder hochautomatisierte Durchführung einer Mission. Bietet dem Operateur hierarchisch organisierte Funktionalitäten auf dem Abstraktionsniveau der Mission, aber auch UAV-spezifische Funktionen zur Gewährleistung eines sicheren Flugbetriebes in Übereinstimmung mit Flugbetriebsregularien (z.B. Sense & Avoid).

Die herausgestellten Komponenten und deren technische Ausprägung werden in den folgenden Abschnitten näher beschrieben, sowie einzelne, exemplarische Forschungsarbeiten auf dem jeweiligen Gebiet vorgestellt. Insgesamt ist das heutige Feld der UAV-bezogenen Technologien und Forschungsaktivitäten allerdings so umfangreich und unübersichtlich, dass hier nur ein sehr selektiver Überblick gegeben werden kann. Dazu wird im Folgenden einzeln auf die oben genannten Funktionsblöcke eingegangen.

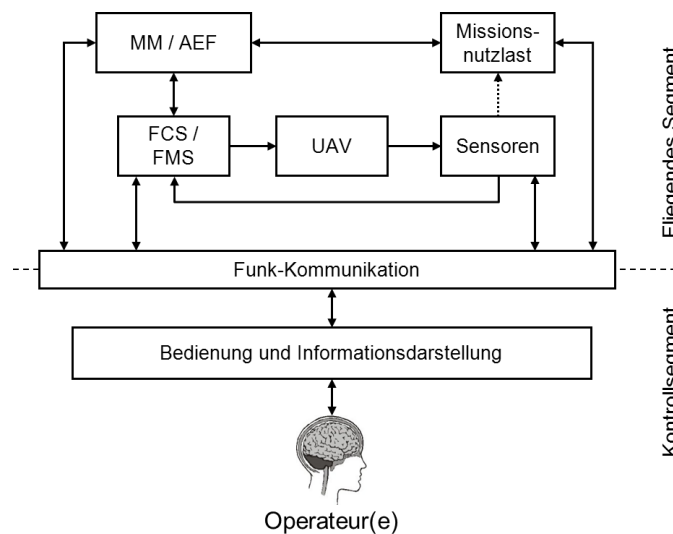


Abbildung 2-1. Schematischer Aufbau eines Führungs- und Missionssystems eines UAVs

2.1.1 UAV

Die Art der Fluggeräte hinsichtlich ihrer Fähigkeiten und Charakteristik ist sehr breit gefächert. Neben kommerziellen Produkten für militärische Anwendungen – wie Global Hawk [Northrop Grumman Corporation, 2006], Predator/Reaper [General Atomics Aeronautical Systems, Inc., 2011] oder Luna und Aladin [EMT Ingenieurgesellschaft, 2009] als Vertreter von Flächenflugzeugen, sowie der FireScout [Northrop Grumman, 2010] und Hummingbird [Boeing Defense, Space and Security, 2011] als Vertreter von unbemannten Hubschraubern – werden auch industrielle Systeme – wie der Yamaha RMAX (Hubschrauber, vgl. Abbildung 2-2) [Yamaha Motor Co., Ltd., 2004], die entsprechenden Multikopter-Systeme der Firmen *Microdrones* [Microdrones GmbH Homepage] oder Ascending Technologies [Ascending Technologies Homepage] – angeboten. In der Forschungsdomäne kommen auch diese Systeme neben Eigenentwicklungen und frei verfügbaren (COTS) Komponenten und Fluggeräten aus dem Bereich des Modellbaus, sowie Ballone und Luftschiffe [Sasa et al., 2008], zum Einsatz.



Abbildung 2-2. Industrie-Hubschrauber RMAX der Firma Yamaha, [Yamaha Motor Co., Ltd., 2004].

So werden beispielsweise schwebefähige Quadkopter von [Achtelik et al., 2009] für die Demonstration von optischen Regelungsverfahren und von [Hermans & Decuypere, 2005] [Cowling et al., 2007] für die Entwicklung von miniaturisierter Inertialsensorik verwendet. [Wyeth et al., 2000a] [Wills et al., 2002] [Roberts et al., 2003] [Buskey et al., 2005] [Taamallah et al., 2005] [Höse et al., 2007] [Schwarzer et al., 2007] [Kriegel et al., 2009] setzen Modellhubschrauber als Basis ihrer Flugdemonstratoren für Forschung im Bereich der nicht-linearen Flugregelung, Systemidentifikation, Bildverarbeitung und die Entwicklung von neuartigen UAV-Führungstechnologien ein. Das kommerzielle Hubschraubersystem RMAX wird von [Calise & Rysdyk, 2002] [Johnson & Kannan, 2002] [Johnson & Mishra, 2002] [Dittrich & Johnson, 2002] [Johnson & Schrage, 2003] [Ludington et al., 2006] ebenfalls für die Entwicklung adaptiver Regelungstechnologien und Navigation auf Basis von Bildverarbeitung verwendet. Demonstratoren auf der Grundlage von Modellflugzeugen sind Gegenstand der Arbeiten von [Wong, 2001] [How et al., 2004] [Quigley et al., 2005] [Höse et al., 2007] (vgl. Abbildung 2-3) [Kaminer et al., 2007] [Brisset et al., 2008] [Dobrokhodov et al., 2008] [Jensen et al., 2009] [Reuder et al., 2009] [Schwarzbach et al., 2009] und vieler Anderer.



Abbildung 2-3. UAV Forschungsdemonstrator der Universität der Bundeswehr München (UniBwM) auf Basis eines Modellflugzeugs.

Die Liste an verfügbaren UAV-Plattformen wird täglich länger und die Konzepte und Technologien, welche aufgrund der hohen Kosten früher nur militärischen Projekten vorbehalten waren, finden auch langsam den Einzug in das Consumer-Segment, wie beispielsweise das System AR-Drone der Firma Parrot [Parrot S.A. Homepage] bestätigt.

2.1.2 Flugregelungssystem / Flugmanagementsystem

Um den Bediener von der Aufgabe der manuellen Steuerung, Stabilisierung, Lagehaltung, Bahnführung und gegebenenfalls auch Navigation eines unbemannten Fluggerätes zu entlasten, kommen Automationstechniken zum Einsatz, die diese Aufgabe übernehmen. Eine Regelung – vorgenommen von einer abgesetzten Kontrollstation – würde eine hochfrequente Übertragung von Steuer- und Telemetriedaten erfordern. Sie schränken als Folge den maximalen Abstand des Fluggerätes von dem Kontrollsegment deutlich ein, da eine Funkübertragung über Relaisstationen bzw. Satelliten zu hohe Lauf- und Latenzzeiten mit sich bringt [van Erp, 2000] [McCarley & Wickens, 2004] [Tvaryanas, 2006]. Damit sorgen sogenannte Autopiloten für die automatische Stabilisierung, Bahnführung sowie Navigation und Lenkung des UAVs. Technisch unterscheiden sich diese Systeme nicht von den bereits seit langem in bemannten zum Einsatz kommenden Systemen. Hierbei kann zwischen der eigentlichen Stabilisierung des Fluggerätes (Flugregelung, engl.: Flight Control System – FCS), dem sogenannten inneren Regelkreis (engl.: Inner Loop), und dem äußeren Regelkreis (Flugmanagement, engl.: Outer Loop), verantwortlich für die Navigation, unterschieden werden. Das Gebiet der Regelungstechnik ist sehr umfassend und in zahlreichen (Lehr-) Büchern (vgl. [Brockhaus, 2001] [Lunze, 2008]) beschrieben und ausführlich dokumentiert. Die Flugregelung als Teildisziplin dieser Wissenschaft beschäftigt sich mit der (von uns im Folgenden so bezeichneten, konventionellen) Automatisierung von Fluggeräten.

Die im weiteren Verlauf exemplarisch genannten Arbeiten sollen nur einen kleinen Überblick über die angewendeten Herangehensweisen und Methoden für die Regelung von UAVs geben und erfüllen in keinerlei Hinsicht den Anspruch auf Vollständigkeit. Der Unterschied zu den bereits erwähnten kommerziellen Systemen in der bemannten Luftfahrt ist hierbei in der Verfügbarkeit kostengünstiger Komponenten (Sensoren, Rechner und Aktuatoren) und im ausgeprägten Trend zur Miniaturisierung zu sehen.

Diese Randbedingungen haben geradezu einen regelrechten Boom ausgelöst, der es Forschergruppen selbst bei eingeschränktesten Budgets erlaubt, eigene fliegende Demonstratorplattformen zu entwickeln und zu betreiben. Die in der Regel geringen bis gar nicht vorhandenen Sicherheitsanforderungen für Mini-UAVs ermöglichen zudem auch die Entwicklung und Flugprobung unkonventioneller und innovativer Regelungs- und Führungskonzepte.

Viele Forschergruppen untersuchen die Gestaltung und Auslegung von verschiedenen Regelgesetzen, um den automatisierten Betrieb eines Fluggerätes zu ermöglichen. So wendet [Brisset et al., 2006] [Brisset et al., 2008] konventionelle Methoden der Regelungstechnik (z.B. PID-Regler) für die Entwicklung eines frei verfügbaren Open-Source Autopiloten an, der von [Chao et al., 2008] [Jensen et al., 2009] [Schwarzbach et al., 2009] im Forschungsfeld der Präzisionslandwirtschaft und von [Reuder et al., 2009] für die Erfassung von meteorologischen Daten eingesetzt wird. Andere kommerzielle Autopilotensysteme, die ebenfalls PID-basierte Reglerarchitekturen verwenden (vgl. [Chao et al., 2007]), sind der MNAV+Stargate Autopilot der Firma Crossbow [Jang & Liccardo, 2006], der Kestrel Autopilot der Firma Procerus [Procerus Technologies, 2010], die MP Autopilotenserie der Firma MicroPilot [MicroPilot, 2005] und die Piccolo Reihe der Firma Cloud Cap Technologies [Cloud Cap Technologies, 2006].



Abbildung 2-4. Kommerzielle Autopilotensysteme: Links der Kestrel Autopilot der Firma Procerus [Procerus Technologies, 2010]; Rechts das System Piccolo II der Firma Cloudcap Technologies [Cloud Cap Technologies, 2006]

Neben diesen konventionellen Regelgesetzen, die konservativ ausgelegt werden und nur in einen bestimmten Bereich der möglichen Flugenveloppe des Fluggerätes funktionieren (vgl. [Johnson & Kannan, 2002]), untersuchen Forschergruppen den Einsatz von adaptiven Elementen innerhalb der Regelstrukturen. Das sind beispielsweise künstliche neuronale Netze (engl.: Artificial Neural Network – ANN), um eine vollständige Ausnutzung der möglichen Bandbreite des fliegenden Systems zu realisieren bzw. Modellungenauigkeiten im Falle von dynamischen Inversionsreglern zu kompensieren [Calise & Rysdyk, 2002] [Johnson & Kannan, 2002] [Holzapfel, 2004] [Johnson & Turbe, 2006] [Unnikrishnan & Balakrishnan, 2006] [Christophersen et al., 2006]. Andere Arbeiten nutzen direkt ANNs für die Lagestabilisierung eines Fluggerätes [Kaneshige et al., 2000] [Wyeth et al., 2000a] [Dierks & Jagannathan, 2009].

Die Arbeiten von [Wyeth et al., 2000a] und [Buskey et al., 2005] zeigen die Ergebnisse der Stabilisierung eines Mini-UAVs, auf Basis eines JR Ergo 60 Modellhubschraubers, durch ein *künstliches neuronales Netz* mit drei Ebenen (engl.: Layer) – nämlich einem Eingangslayer, einem versteckten Layer (engl.: Hidden layer) und einem Ausgangslayer

(vgl. Abbildung 2-5). Bei der Durchführung eines manuellen Fluges durch einen erfahrenen Modellhubschrauberpiloten werden die Drehraten, Lage- und Beschleunigungsdaten von einem Trägheitsnavigationssystem (engl.: Inertial Navigation System – INS) erfasst und zusammen mit den Steuerdaten des Piloten aufgezeichnet. Diese Daten werden weiterverarbeitet und für jeden Eingangs- und Ausgangswert in einem Zehnpunktevektor in Form einer nicht-normalisierten Gauss-Verteilung diskretisiert, um einerseits die Datenmenge zu reduzieren und andererseits eine Generalisierung zwischen den Eingangsdaten (INS-Daten) und den Ausgangsdaten (Steuerdaten des Piloten) zu erreichen. Für das Training des Netzes wurde der Backpropagation Algorithmus verwendet. Eine Anzahl von zehn Neuronen im Hidden Layer hat sich laut Buskey als bester Kompromiss zwischen geringem, mittleren quadratischen Fehler (engl.: Mean Square Error – MSE) und der Generalisierungsfähigkeit des ANNs ergeben. Eine ähnliche Arbeit wurde auch an der UniBwM durchgeführt, bei der ebenfalls ein Flugregler auf Basis eines neuronalen Netzes (Backpropagation Algorithmus) implementiert und erprobt wurde. Die Ergebnisse der erfolgreichen Simulatorversuche sind in [Jaeger, 2007] dokumentiert.

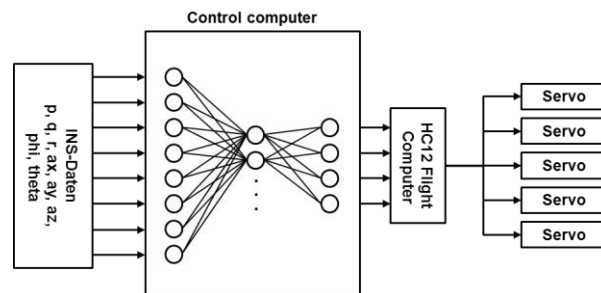


Abbildung 2-5. Lagestabilisierung eines Mini-UAV Hubschraubers mit einem ANN nach [Buskey et al., 2005]

Andere Forschergruppen nutzen *Fuzzy Logic* (vgl. [Doitsidis et al., 2004]) oder auch *evolutionäre Algorithmen* (vgl. [Oh & Barlow, 2004] [Khantsis & Bourmistrova, 2005]) für die Regelung von unbemannten Fluggeräten. Um die Navigationsmöglichkeiten – im Hinblick auf eine koordinierte Multi-UAV Bahnführung – zu erweitern, setzen [Kaminer et al., 2007] und [Dobrokhodov et al., 2008] mathematische Verfahren wie die $\mathcal{L}1$ -Optimierung ein, um nichtlineare, nach räumlichen und zeitlichen Randbedingungen optimierte Trajektorien abzufliegen [Aurich, 2011] [Claus, 2011].

Es gibt eine Vielzahl verschiedenster Ansätze, Technologien und fertiger Produkte – alle mit dem Ziel ein Mini-UAV zu befähigen sich automatisch zu stabilisieren, vorgegebene Wegpunkte abzufliegen oder Flugpfaden zu folgen. Abhängig von der Mission bzw. des Einsatzzwecks des UAVs können hier verschiedene Fähigkeiten voraussetzend sein, die wiederum entsprechende Auswirkungen auf die eingesetzte Technologie haben und gegebenenfalls spezielle Regelalgorithmen erfordern.

2.1.3 Sensoren

Für eine automatisierte Stabilisierung und Navigation von unbemannten Fluggeräten werden entsprechende Flugzustandsinformationen benötigt, die von Sensoren erfasst,

aufgenommen und dem FCS bzw. FMS zur Verfügung gestellt werden müssen. Die wichtigsten zu messenden Größen umfassen hierbei die Position und Flughöhe des Fluggerätes, die Fluggeschwindigkeit, Flugrichtung, Lagewinkel (z.B. Hänge- und Nickwinkel) und die Drehraten. Für die Bestimmung der Position und Flughöhe werden häufig GPS-Empfänger eingesetzt, deren Messgenauigkeit durch differentielles GPS (ein statischer GPS-Empfänger mit bekannter Position sendet Korrektursignale an den bewegten GPS-Empfänger) noch verbessert werden kann. Zudem liefern GPS-Empfänger die Bewegungsrichtung (jedoch sehr ungenau bei geringer Geschwindigkeit) und die Geschwindigkeit über Grund. Bei schwebefähigen UAVs werden für die Ermittlung der aktuellen Ausrichtung der Längsachse Magnetfeldsensoren eingesetzt. Die Ermittlung der Fluggeschwindigkeit aber auch der Flughöhe erfolgt mittels Luftdrucksensoren (z.B. Staudruck-, statische Druck- oder Totaldrucksensoren). Die aktuellen Drehraten können mit Gyroskopen (Kreiseln) erfasst werden. Deren Kombination mit Beschleunigungssensoren und entsprechenden Filtertechniken, wie zum Beispiel Kalman Filter [Kalman, 1960] [Welch & Bishop, 1995] oder Komplementärfiltern [Higgins, 1975] [Euston et al., 2008] ermöglicht die Errechnung der Lagewinkel des Fluggerätes.

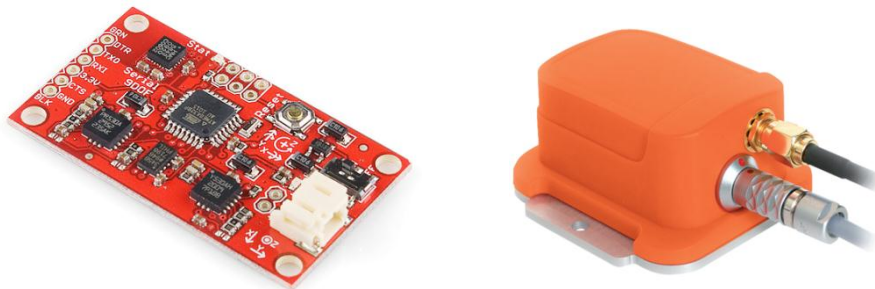


Abbildung 2-6: Links: Hobby-IMU-Bord mit Mikrocontroller der Firma Sparkfun [Sparkfun Electronics Homepage]
Rechts: COTS IMU mit integriertem GPS-Empfänger der Firma XSens [Xsens Homepage]

Abbildung 2-6 zeigt zwei unterschiedliche IMUs der Firmen Sparkfun und XSens. Links dargestellt ist eine IMU-Platine für den Hobbybereich, bestückt mit MEMS- (engl.: Micro Electro Mechanical System) und Magnetfeldsensoren und einem 8-bit Mikrocontroller, auf dem entsprechende Filteralgorithmen zur Sensordatenfusion programmiert und gespeichert werden können. Das rechte Bild zeigt die kombinierte MTI-G IMU-Einheit mit GPS Empfänger und integriertem, digitalen Signalprozessor (engl.: Digital Signal Processor – DSP) und bereits implementierten Sensordatenfusionsalgorithmen.

[Hermans & Decuyper, 2005] [Christophersen et al., 2006] [Gurdan et al., 2007] [Jang & Liccardo, 2006] [Achtelik et al., 2009] setzen ebenfalls MEMS bzw. Piezo-Sensoren für die Entwicklung und den Aufbau eines Trägheitsnavigationssystems (engl.: Inertial Measurement Unit – IMU) ein. Im Gegenzug nutzen [Wyeth et al., 2000b] [Johnson & Schrage, 2003] [Buskey et al., 2005] [Taamallah et al., 2005] [Kriegel et al., 2009] COTS IMUs für die Lagebestimmung ihrer Fluggeräte. Zudem gibt es noch andere Methoden, um den Flugzustand zu erfassen. So verwenden [Brisset et al., 2006]

[Kawalec et al., 2006] [Adiprawita et al., 2007] [Brisset et al., 2008] Thermopilesensoren für die Horizonterkennung (basierend auf dem Temperaturunterschied zwischen der Erde und dem Himmel) und [Proctor & Johnson, 2004] die Daten einer einzelnen Kamera mit extensiver Bilderverarbeitung für die Regelung und Navigation eines Starrflügler-UAVs.

2.1.4 Funk-Kommunikation

Der Transfer von Daten zwischen der Kontrollstation und den Subsystemen des UAVs erfolgt mittels Funkübertragung. Abhängig von der geplanten Reichweite und Flugdauer des UAVs, sowie die Menge der zu übertragenden Daten werden hier verschiedenste Anforderungen an die Kommunikationseinrichtung gestellt. Im Allgemeinen kann zwischen einem sogenannten *Führungslink*, der durch eine geringe Bandbreite mit hoher Reichweite und Störfestigkeit charakterisiert ist, und einer Verbindung mit hoher Bandbreite für die *Übertragung von gewonnenen Nutzlastdaten* unterschieden werden (vgl. [Barnard, 2008]). Forschungsdemonstratoren, die im universitären Umfeld betrieben werden, nutzen hier meist serielle Datenfunkmodems für die Übertragung von Führungskommandos und Telemetriedaten (vgl. [Dittrich & Johnson, 2002] [Roberts et al., 2003] [Johnson & Schrage, 2003] [Quigley et al., 2005] [Taamallah et al., 2005] [Johnson & Turbe, 2006] [Brisset et al., 2006] [Achtelik et al., 2009] [Kriegel et al., 2009]), während entweder ein dediziertes Funkmodem für die Versendung eines Kamerastreams (vgl. [Hermans & Decuypere, 2005] [Taamallah et al., 2005] [Johnson & Turbe, 2006]) oder auch WLAN (engl.: Wireless Local Area Network – drahtloses lokales Netzwerk) (vgl. [Dittrich & Johnson, 2002] [Taamallah et al., 2005] [Ludington et al., 2006] [Lange et al., 2008]) für die Übertragung größerer Datenmengen von Sensoren eingesetzt wird. Die Frequenzen liegen in der Regel innerhalb freigegebener Bereiche wie beispielsweise dem gebührenfreien ISM-Band (Industrial, Scientific & Medical Band), für das eine Allgemeinzulassung des Gerätes ausreicht, ohne dass es einer Einzel-Frequenzzuweisung durch Regulierungsbehörden bedarf. Kommerzielle UAV-Systeme – die im militärischen Umfeld Anwendung finden – verfügen zudem je nach Aufgabenspektrum über Funkanlagen, die Reichweiten von bis zu 100km bei Sichtverbindung (engl.: LOS – Line of Sight) (vgl. System Luna – [EMT Ingenieurgesellschaft, 2009]) oder so gut wie unbegrenzte Reichweiten mittels Satellitenkommunikation (vgl. [Sarris, 2001] [Northrop Grumman Corporation, 2006] [Tvaryanas, 2006] [Barnard, 2008] [Bento, 2008]) ermöglichen.

2.1.5 Missionsnutzlast

Die bisher genannten Flugzustandssensoren sind für den Betrieb der Fluggeräte und deren Automation notwendig, jedoch dient der Einsatz von UAVs immer einer zu erfüllenden Aufgabe oder Mission. Das Ziel kann entweder darin bestehen, erforschte Technologien zu demonstrieren, benötigte Daten zu erheben oder im militärischen Umfeld auch Aufklärungs- bzw. Kampfeinsätze durchzuführen. Hierfür werden zusätzliche Missionssensoren benötigt.

In vielen Szenarien geht es um die Aufnahme von Luftbildern und Quasi-Echtzeit Videostreams und bezeichnet damit den Dual-Use Einsatz solcher Systeme sowohl für

den zivilen als auch für den militärischen Zweck. Typische zivile Anwendungen sind das Erstellen von Luftaufnahmen für Fotografen, Immobilienmakler, Architekten oder Bauunternehmen [Microdrones GmbH Homepage] sowie für den Bereich der Archäologie [Lambers et al., 2007] oder der Landwirtschaft (vgl. [Reidelstürz & Schulte, 2010]). Auch für das Erzeugen von aus der Luft gefilmten Videos für die Filmindustrie, Nachrichten oder zur Durchführung von Inspektionsarbeiten an Pipelines, Windrädern oder Bohrinseln kommen UAVs zum Einsatz. Im militärischen Bereich finden diese Fähigkeiten zu Erkundungszwecken, Überwachung von bestimmten Gebieten bis hin zur Schadensbewertung und Wirkaufklärung (engl.: Battle Damage Assessment – BDA) von Angriffszielen Anwendung. Abhängig vom bildgebenden Sensor können verschiedene Lichtspektren erfasst werden, nämlich das sichtbare Licht (elektro-optisch – EO, Wellenlänge: 380nm – 640nm), der Nahinfrarotbereich (NIR, Wellenlänge: ca.780nm – 2µm) und der mittlere Infrarotbereich (MIR, Wellenlänge: ca. 2-50µm). Das Ergebnis von Aufnahmen im Spektrum des sichtbaren Lichts ist offensichtlich, denn das Resultat kennt man von heutigen Digital- und Videokameras. Die Auswertung von NIR Bildern ermöglicht beispielsweise die Berechnung eines normalisierten differenzierten Vegetationsindex (engl.: Normalized Difference Vegetation Index – NDVI), um Aussagen über die photosynthetische Aktivität und Vitalität von Pflanzen treffen zu können [Chao et al., 2008] [Schwarzbach et al., 2009]. Bildgebende Sensoren im MIR Bereich erfassen die Wärmestrahlung von Objekten und ermöglichen auch den Einsatz bei schlechten Lichtverhältnissen und sogar bei Nacht. [Biass, 2008] bietet eine kurze Übersicht über kommerzielle Systeme, welche verschiedene EO und IR-Sensoren in einer stabilisierten, kardanischen Aufhängung vereinen. [Breiter et al., 2006] beschreiben die IR-Sensoren der Firma AIM, welche unter anderen auf den deutschen Drohnensystemen KZO, Luna und Aladin zum Einsatz kommen.

Weitere Sensoren sind aufgrund ihrer hohen Kosten bzw. des hohen Energiebedarfes fast ausschließlich in militärischen Systemen unter Verwendung, wie beispielsweise Radarsensoren zur Fernerkundung. Radargeräte mit synthetischer Apertur (engl. Synthetic aperture radar – SAR) sind eine Unterart von bildgebenden Radaren, die eine zweidimensionale Darstellung (engl.: Image Intelligence – IMINT) von Geländeabschnitten durch aktive Abtastung erstellen können und auch das aktive Verfolgen von bewegten Zielen erlauben (engl.: Moving Target Indicator – MTI). Der Vorteil zu EO und IR bildgebenden Sensoren ist einerseits die hohe Reichweite und andererseits ihre nahezu uneingeschränkte Einsatzfähigkeit, da sie unabhängig der vorherrschenden Witterungsbedingungen funktionieren (z.B. Nebel, Wolken, Regen). Ein Nachteil ist jedoch, dass die Datengewinnung eine Emission von elektromagnetischen Wellen impliziert, die mit entsprechenden Emitter-Locator Systemen vergleichsweise einfach zu orten ist. Aktuelle, mit SAR-Geräten ausgerüstete UAV Systeme sind beispielsweise die deutsche Aufklärungsdrohne Luna-X 2000 (weiterentwickelte Version der Ur-Luna) und die amerikanischen Drohnen Predator sowie Global Hawk, um nur einige zu nennen.

Andere Sensoren umfassen den Bereich der elektronischen Aufklärung in den Bereichen Signalaufklärung (engl.: Signal intelligence – SIGINT) und elektronische Aufklärung (engl.: Electronic intelligence – ELINT). Aufgrund der hohen Sensibilität von solchen Systemen sind darüber jedoch keine genaueren Informationen oder technische Daten öffentlich zugänglich.

Die Vielfalt an verfügbaren Sensorsystemen ist sehr breit gefächert und stark abhängig von der Applikationsdomäne. Manche Daten werden auch mehrfach für die Bestimmung des Flugzustands durch Bildverarbeitung oder für die eigentlich bildgebende Aufklärung verwendet, wodurch sich der Bereich der Flugzustandssensorik und der Missionssensorik nicht mehr eindeutig distinguieren lässt.

2.1.6 Bedienung und Informationsdarstellung

Die Interaktion des Menschen mit dem Fluggerät kann in zwei Richtungen unterteilt werden, nämlich die Darstellung der Informationen von dem UAV für den Bediener und die Eingabe von Führungskommandos des Menschen an das Fluggerät. Nach [Chen et al., 2007] verfügen Kontrollstationen von UAVs typischerweise über mehrere Anzeigen, welche die Daten der Nutzlast (z.B. Kamerabilder), Telemetrie (z.B. aktuelle Position und Fluglage) und den aktuellen Systemzustand darstellen.

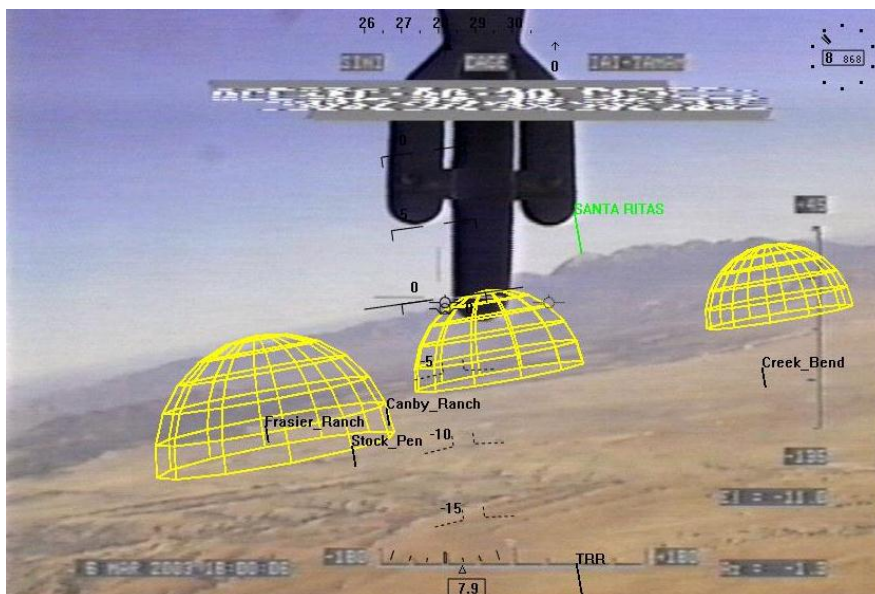


Abbildung 2-7. Echtzeitvideo eines UAVs mit überlagerten, computergenerierten Symbolen [Draper et al., 2006].

Für das Aufrechterhalten eines Situationsbewusstseins (engl.: situation awareness, vgl. [Endsley, 1996]) des Operateurs und die Erleichterung von Navigationsaufgaben dienen Kartendisplays. Gegenstand der Forschung ist das Kombinieren von tatsächlich gewonnenen Sensordaten mit synthetischen Informationen – auch hier mit dem Ziel, das Situationsbewusstsein des Bedieners zu verbessern [Tadema & Theunissen, 2003] [Calhoun et al., 2005] [Tadema et al., 2006] [Draper et al., 2006] oder die optimale Darstellung nebenläufiger Aufgaben mehrerer UAVs zu gewährleisten [Cummins & Mitchell, 2005].

Des Weiteren benötigt der Bediener Eingabegeräte, um entsprechende Kommandos oder auch Pläne für das Fluggerät erstellen und diese abschicken zu können. Hierfür eignen sich Touchscreen-Displays, die sowohl für die Anzeige als auch für die Nutzereingaben geeignet sind, ebenso wie Steuerknüppel (engl.: Joystick), Schalter, Computermäuse und Tastaturen. Neben diesen konventionellen Eingabegeräten wurde von [Draper et al., 2003] untersucht, verschiedene Steuerungsaufgaben mittels Sprachkommandos durchzuführen. Hierfür sollten verschiedene Teilnehmer ein simuliertes UAV manuell steuern und navigieren und gleichzeitig eine Dateneingabe entweder manuell (durch Drücken entsprechender Knöpfe) oder durch Spracheingabe vornehmen. Die gewonnenen Ergebnisse haben gezeigt, dass durch Spracheingabe eine deutlich bessere Performanz – sowohl bei der Steueraufgabe des UAVs (akkuratere Navigation) als auch bei der Dateneingabe (schneller und genauer) – erreicht werden konnte, da zahlreiche, sequentielle Tastendrucke durch ein Sprachkommando ersetzt werden.



*Abbildung 2-8. Links: Arbeitsplätze innerhalb der Bodenkontrollstation für das System Predator (Bild von General Atomics Aeronautical Systems [General Atomics Aeronautical Systems, Inc., 2011])
Rechts: Tragbare Bodenkontrollstation für das System Aladin (Bild von EMT Penzberg [EMT Ingenieurgesellschaft, 2009])*

Weitere Forschungsaktivitäten von [Aracil et al., 2002] im Bereich der Wartung von aktiven Hochspannungsleitungen und [Goza et al., 2004] für die Steuerung eines humanoiden Roboters haben ebenfalls ergeben, dass die Erweiterung der Steuerungsmöglichkeiten durch Spracheingaben geeignet ist. Das ist dann sinnvoll, wenn für die Steuerung und Kontrolle von komplexeren Robotern mit vielen Freiheitsgraden schon beide Hände und Arme benötigt werden oder der Operator sich in einer bewegten Umgebung befindet, die eine feinmotorische Kontrolle seiner Extremitäten verhindert [Hill & Tauson, 2001] [Scribner & Dahn, 2008]. Im Gegenzug kann ebenso das Feedback der Maschine – im Sinne der Informationsdarstellung für einen Operator – im Vergleich zu einem rein uni- bzw. monomodalen und visuellen Ansatz, durch eine multimodale Art der Anzeige erweitert werden. Forschergruppen untersuchen in diesem Umfeld den Nutzen von auditiven Displays als Ergänzung zu visuellen Anzeigen. Die Ergebnisse von [Tachi et al., 2003] [Simpson et al., 2004] [Trouvain & Schlick, 2007] zeigen, dass hierdurch eine Verbesserung des Situationsbewusstseins des Operators, eine gezielte Lenkung der Aufmerksamkeit des Menschen auf visuell dargestellte

Objekte und die Übermittlung von komplexen Informationen erreicht und somit auch generell einer Überlastung des visuellen Informationskanals entgegengewirkt werden kann.

Eine weitere, ergänzende Displayart ist die hautbasierte und propriozeptive Informationsvermittlung durch Druck- bzw. Vibrationserzeugungsaktuatoren (engl.: haptic or tactile displays, vgl. [van Erp, 2002] [Gemperle et al., 2002] [Chen et al., 2007] [Haas & Stachowiak, 2007]). Das Forschungsgebiet umfasst dabei die Weitergabe von Warnungen und die Kommunikation von Orientierungs- und Positionsinformationen [Cholewiak & Collins, 2000] [van Erp & Van Veen, 2003].

Die genannten Arbeiten zählen zu dem Bereich des „Human Factors Engineering“, allerdings benötigen diese Technologien auch entsprechende Hardware für die technische Umsetzung im Rahmen einer UAV-Kontrollstation.

Die Erscheinungsformen von Kontrollstationen reichen hier von sogenannten Smartphones [Fong & Thorpe, 2001] [Quigley et al., 2005], Notebook-Computern [Taamallah et al., 2005] [Brisset et al., 2006] [EMT Ingenieurgesellschaft, 2009] (vgl. Abbildung 2-8, rechts) bis hin zu voll ausgerüsteten Fahrzeugen bzw. Containern [Höse et al., 2007] [Kriegel et al., 2009] [General Atomics Aeronautical Systems, Inc., 2011] (vgl. Abbildung 2-8, links), in denen dem/den Operateur/en die Telemetriedaten des Fluggerätes angezeigt und entsprechende Eingabegeräte zur Verfügung gestellt werden. Auch sind Kontrollstationen innerhalb von bemannten, fliegenden Plattformen, wie beispielsweise bei dem aktuellen Forschungsprojekt *Manned-Unmanned Teaming – MUM-T* der Universität der Bundeswehr München (UniBwM) denkbar [Rauschert et al., 2008] [Uhrmann et al., 2009] [Uhrmann et al., 2010b] [Strenzke et al., 2011].

2.1.7 Missionsmanagement

Die Missionsführung von aktuellen militärisch genutzten UAVs erfolgt derzeit mehrheitlich durch die Vorgabe von Sollwerten für den Autopiloten bzw. die Wegpunktnavigation mittels FMS auf Basis von vor dem Einsatz festgelegten Flugrouten. Für die Bedienung sind typischerweise mindestens zwei Operateure vorhanden, einer verantwortlich für die Flugführung und der/die Andere/n für die Payload und Missionssensorik [Tvaryanas, 2006].

Einige Drohnen werden in der Start- und Landephase manuell gesteuert, wie etwa die Systeme Predator (MQ-1/9, USAF), Hunter (RQ-5, US Army), Pioneer (RS-2, USN) [Williams, 2004]. Andere Systeme wie Shadow (RQ-7, US Army), Aladin, Luna (Bundeswehr) und Sperwer (Französische Armee) werden mit Hilfe von Katapulten gestartet, um diese auch im unwegsamen Gelände und von Schiffen aus einsetzen zu können. Die Landungen solcher Systeme erfolgt entweder durch Fallschirmsysteme (z.B. Luna), das Fliegen in aufgespannte Fangnetze (z.B. Pioneer, Luna) oder als Durchführung von automatischen Landungen (z.B. Shadow, Global Hawk). Seitens der Betreiber besteht hier ein großes Interesse nach höherwertigen Automationsfunktionen im Hinblick auf eine gesteigerte „Autonomie“ der Fluggeräte (engl.: Autonomy Enabling Functions – AEFs). Durch zusätzliche Fähigkeiten der UAVs soll nicht nur die

Missionsdurchführung erleichtert sondern auch gleichzeitig deren Betriebssicherheit und Effizienz gesteigert werden.

Ein Beispiel für eine solche Funktion ist das automatische Landen eines Fluggerätes, das zwar technisch möglich aber mit zusätzlichem Aufwand verbunden ist, da entsprechend qualifizierte Geräte an Bord und/oder am Boden benötigt werden, um die geforderte Positionsgenauigkeit zu erreichen. So wird etwa beim System Global Hawk eine eigene Bodenkontrollstation für das automatische Starten und Landen unter Einsatz von differentielltem GPS und bei dem System Shadow ein zusätzliches Landeradar (Tactical Automatic Landing System – TALS [SNC - Sierra Nevada Corporation, 2006]) verwendet. In diesem Umfeld forschen [Sharp et al., 2001] und [Lange et al., 2008], um mittels Bildverarbeitung Präzisionslandungen beispielsweise auf Schiffen durch Erkennung und Identifikation von eindeutigen Markierungen durchzuführen. Auch [Garcia-Pardo et al., 2002] und [Bosch et al., 2006] setzen Bildverarbeitung ein, um geeignete und hindernisfreie Landeplätze für schwebefähige UAVs in unbekanntem Gelände zu bestimmen.

Andere Arbeiten von [Wyeth et al., 2000b] [Bhanu et al., 2002] [He et al., 2006] [Watanabe et al., 2007] beschäftigen sich mit der Thematik des Erkennens und Ausweichens (engl.: Sense/See and Avoid) von anderen Luftfahrzeugen bzw. Hindernissen im bodennahen Flug. Dabei werden wiederum Methoden der Bildverarbeitung aber auch die Auswertung von Daten aktiver Sensoren (z.B. Radar, Lidar) eingesetzt.

Ein weiteres in der Forschung adressiertes Gebiet ist die automatisierte Berechnung von Flugpfaden nach vorgegebenen Randbedingungen, um zum einen die statische Vorauslegung von wegpunktbasierten Flugrouten flexibler zu gestalten und zum anderen diese Fähigkeit auch an Bord des fliegenden System nutzen zu können. So verwendet [Bortoff, 2000] Voronoi-Graphen im ersten und künstliche Kräftefelder im zweiten Schritt für die Berechnung und Optimierung von Flugpfaden durch ein bedrohtes Gebiet. Hingegen legt [Oomkens et al., 2008] virtuelle Kostenkarten (engl.: Cost maps) über einem Gebiet an, die von einem A*-Algorithmus (informierter Suchalgorithmus) für die Bestimmung des optimalen Flugpfades herangezogen werden. Neben den entsprechenden Regelalgorithmen, die für das automatisierte Abfliegen dieser Trajektorien notwendig sind, untersuchen [Sinha et al., 2005] [Wise & Rysdyk, 2006] [Tisdale et al., 2009] verschiedene Algorithmen mit dem Ziel, mehrere unbemannte Fluggeräte und deren Sensoren optimiert für eine Aufklärungsmission (engl.: Cooperative Search) einzusetzen.

Forschungsgegenstand von [Sinopoli et al., 2001] und [Ludington et al., 2006] ist die sichtbasierte Navigation mit dem Ziel, die Positionsgenauigkeit auch in Gebieten zu verbessern, in denen eine Positionsbestimmung mittels GPS sehr ungenau (z.B. in urbanen Gebieten), gestört (z.B. Störsender) bzw. gar nicht verfügbar ist (engl.: GPS-denied environment, z.B. innerhalb von Gebäuden). Darüber hinaus untersuchen [Dobrokhodov et al., 2006] und [Ludington et al., 2006] – unter Einsatz eines bildgebenden EO-Sensors – das automatische Verfolgen von bewegten Bodenzielen.

Ferner ist bei Ludington das Finden und Erkennen eines mit einem speziellen Symbol versehenen Gebäudes mit der nachfolgenden Identifikation von offenen Fenstern und Türen Gegenstand der Forschung.

Vereint man die bereits bestehenden Technologien mit Ergebnissen aus dem Bereich der Wissenschaft, so würde ein solch unbemanntes Fluggerät über eine Vielzahl von Funktionen und Fähigkeiten verfügen, die zur Durchführung und Erfüllung eines Auftrags oder einer Mission sinnvoll eingesetzt werden müssen. Diese Aufgabe wird derzeit noch vornehmlich von menschlichen Operateuren, folgend dem klassischen Paradigma der *supervisory control*, wahrgenommen. Allerdings kommen hierfür auch technische Systeme in Frage, wie aktuelle Forschungsthemen belegen. Ein solches, sogenanntes Missionsmanagementsystem (MMS) muss für die automatisierte Erfüllung einer Aufgabe bzw. Mission ein Verständnis darüber haben, wie diese durchzuführen ist und auch welche Fähigkeiten und Möglichkeiten die Automation des Fluggerätes bietet, um letztlich einen Vorgehensplan zu erstellen und diesen abzuarbeiten. So untersuchen [Grecu & Gonsalves, 2000] in simulierten Missionen zur Unterdrückung feindlicher Luftabwehr (engl.: Suppression of Enemy Air Defence – SEAD) und [Karim et al., 2004] [Karim & Heinze, 2005] bei vergleichsweise einfachen Demonstrationsmissionen (Umplanung von Flugwegen aufgrund von simulierten Wetterbedingungen) den Einsatz von künstlichen Agenten für die Umsetzung eines MMS. [Sullivan et al., 2004] setzen ebenfalls künstliche Agenten auf Basis der APEX Architektur, für die simulierte Durchführung von Waldbranderkennungsmissionen im Gebiet der Westlichen Staaten von Amerika (engl.: Western States Fire Mission) ein.

2.1.7.1 Common Operating System – COS

Im Rahmen des von der DARPA (engl.: Defence Advanced Research Projects Agency) initiierten Projekts J-UCAS (engl.: Joint Unmanned Combat Air System) – einem Gemeinschaftsprogramm der US Air Force und der US Navy – soll durch ein einheitliches Betriebssystem (engl.: Common Operating System – COS) die Vernetzung (z.B. mit Link 16) und das Management verschiedener Dienste und System Ressourcen – wie beispielsweise Sensoren, Waffen und Kommunikationseinrichtungen – ermöglicht werden [DARPA, 2005]. Ziel ist es, durch eine geschichtete Softwarearchitektur die Unabhängigkeit der Software von der Hardware zu erreichen (Hardware-Abstraktion), um die Portabilität der einzelnen Softwareprogramme zu erhöhen und die Modernisierung der Software zu vereinfachen [Adams, 2005]. Das System soll auf neu entwickelten Fluggeräten, wie beispielsweise der X-45 der Firma Boeing oder der X-47 der Firma Northrop Grumman, integriert und erprobt werden. Derzeit wurden jedoch keine genaueren Informationen über die verfolgten Konzepte und Details der Umsetzung veröffentlicht und erlauben demnach auch keine weitergehende Betrachtung bzw. Bewertung.

2.1.7.2 Auftragsbasierte Führung

Gegenstand der Arbeiten von [Valenti et al., 2004] und [Schouwenaars et al., 2005] ist das auftragsbasierte Führen eines UAVs von einem bemannten Flugzeug mittels Sprachkommandos. Mögliche Befehle umfassen dabei beispielsweise das Anfliegen

eines vorgegebenen Wegpunkts, das Aufklären eines bestimmten Gebiets oder das Zurückkehren zur Basis. Die Mission sieht ein bemanntes Flugzeug und ein UAV vor, die einen Aufklärungseinsatz in einem teilweise bekannten, feindlichen Gebiet durchführen sollen. Das UAV verfügt dabei über die Fähigkeit Bedrohungen zu erkennen und Aufklärungsinformationen (Bilder) zu sammeln, wohingegen das bemannte Flugzeug in der Lage ist, entsprechende Effektoren freizusetzen, um mögliche Bedrohungen zu bewirken (Waffeneinsatz). Über das entwickelte Missionssystem und dessen Umsetzung sind keine weiteren Details der Implementierung veröffentlicht, jedoch wurde bei diesem Projekt der Schwerpunkt auf die konfliktfreie und echtzeitfähige Planung des Flugwegs auf Basis von Methoden der ganzzahligen linearen Optimierung (engl.: Mixed-Integer Linear Programming – MILP) gelegt. Während der Missionsdurchführung werden dem UAV aktuelle Informationen über eine bis dato unbekannte, auftauchende Bedrohung (engl.: Pop-Up Thread) eingespielt, woraufhin das UAV selbständig eine Neuplanung für einen bedrohungsfreien Flugpfad vornimmt und das bemannte Flugzeug über die Gefahr informiert. Nach Erreichen des Zielgebietes beginnt das UAV mit der Suche nach dem Ziel. Im Anschluss an dessen Identifikation schickt das UAV die Position an das bemannte Flugzeug, welches das Ziel abschließend bewirkt. Das Projekt wurde in realen Flugversuchen evaluiert und die Systeme in eine Boeing F-15 (bemanntes Flugzeug) und eine Lockheed T-33 als UAV Surrogat integriert. Die Ergebnisse zeigen die Machbarkeit der auftragsbasierten Führung durch Sprachkommandos von einer bemannt fliegenden Plattform. Zudem erwähnenswert ist, dass jeweils immer nur *eine* Aufgabe an das UAV gestellt und von dem MMS des UAVs bearbeitet werden kann.

Gegenstand der Forschung bei [Uhrmann et al., 2009] ist ebenfalls die auftragsbasierte Führung, allerdings von mehreren unbemannten Fluggeräten im Umfeld des bereits erwähnten Forschungsprojekts *Manned-Unmanned Teaming – MUM-T*. Die definierte Mission entspricht dabei einem luftgestützten Einsatz, bei dem von einem Helikopter aus Truppen an einer ausgewiesenen Landezone abgesetzt werden und drei UAVs vor dem Eintreffen des bemannten Hubschraubers eine zeitnahe Aufklärung der Flugroute und der Landezone vornehmen. In einer sehr komplexen Simulationsumgebung mit dynamischem Szenario werden die Vorteile der Führung auf der abstrakten, symbolischen Missionsebene im Vergleich zu der parameter-getriebenen Wegpunkt-navigation untersucht. Hierbei können dem UAV *mehrere* Aufgaben übertragen werden, die von einem intelligenten Agenten in einer sinnvollen Reihenfolge in eine Aufgabenagenda einsortiert werden. Diese Aufgabe erfüllt eine künstliche kognitive Einheit, welche auch die spätere Abarbeitung der Aufgaben steuert und überwacht. Während der Durchführung können zudem weitere Aufgaben hinzugefügt bzw. bereits zugewiesene Aufgaben verändert oder wieder entfernt werden. Die bis dato gewonnenen Ergebnisse, dargestellt in [Uhrmann et al., 2010a], zeigen die Vorteile und auch das Potential dieser Technologie.

2.1.7.3 Maschinelle Kooperation

Abschließend seien noch Aktivitäten im Bereich der maschinellen Kooperation erwähnt. So untersuchen [Campbell & Whitacre, 2007] und [Shima et al., 2007] die Planung und das Einnehmen einer räumlich und zeitlich optimierten Sensorkonfiguration für mehrere UAVs, um beispielsweise in einem urbanen Gebiet verschiedene Bodenziele zu identifizieren und zu verfolgen. Hierbei kommen mathematische Optimierungsverfahren zum Einsatz, deren Ergebnisse entsprechende Anwendung in der Regelungstechnik finden, jedoch nicht in den Gesamtkontext einer Mission gestellt werden. Im Gegensatz dazu untersuchen [Meitinger & Schulte, 2007] im Rahmen des Projekts COSY^{team} die wissensbasierte, kooperative Missionserfüllung von mehreren UAVs mit unterschiedlicher Bewaffnung in einer simulierten SEAD/Angriffsmission. Für die Umsetzung einer künstlichen kognitiven Einheit wurde als Grundlage die kognitive Systemarchitektur COSA [Putzer, 2004] (engl.: Cognitive System Architecture) und die Programmiersprache CPL (engl.: Cognitive Programming Language) verwendet (vgl. Abschnitt 2.4.4). Die Evaluationsergebnisse von Simulationsversuchen hinsichtlich der Umsetzbarkeit und der Eignung für Mensch-Maschine Kooperation sind in [Meitinger, 2008] dargestellt.

Die beschriebenen, notwendigen Komponenten für ein UAV-Führungssystem bieten eine breit gefächerte, multidisziplinäre Forschungslandschaft, in der eine hohe Diversität von Arbeiten existiert. Die angestrebten, höherwertigen Funktionalitäten und Fähigkeiten der Fluggeräte bedingen neben dem Menschen eine weitere technische Komponente, nämlich das MMS, um diese Systeme im Hinblick auf eine hochautomatisierte Erfüllung von gegebenen Aufträgen und Missionen einzusetzen. Durch die Differenziertheit der technischen Systeme – und der sich daraus ergebenden hohen Systemkomplexität – resultieren hohe Anforderungen an den Bediener und führen zu Problemen, über die der folgende Abschnitt einen kurzen Überblick gibt.

2.2 Grenzen konventioneller Automation

Die aktuellen Führungsparadigmen von UAVs, welche meist Wegpunktnavigation und Kontrolle durch Überwachung basieren, verbunden mit dem Einsatz konventioneller Automation, führen zu Problemen in der Wechselwirkung zwischen dem technischen System und dem menschlichen Bediener. Hierdurch reduziert sich einerseits die Effizienz des Gesamtsystems Mensch & Maschine und andererseits ist, im Vergleich zur bemannten Luftfahrt, eine deutlich erhöhte Flugunfallquote (30 – 300 fach) zu verzeichnen [DoD, 2001] [Schaefer, 2003] [Weigmann, 2003]. Neben technischen Problemen ist der Faktor Mensch die Hauptursache für Flugunfälle, was im ersten Moment ein wenig paradox erscheinen mag, da es sich hier ja um *unbemannte* Fluggeräte handelt. Die Unfallstatistik führt das System *Predator* an, bei dem mehr als zwei Drittel der Abstürze auf menschliche Fehlbedienung zurückzuführen sind, wie die Untersuchung von [Williams, 2004] belegt. Im weiteren Verlauf des Abschnitts sollen einige der Gründe hierfür aufgezeigt und erläutert werden.

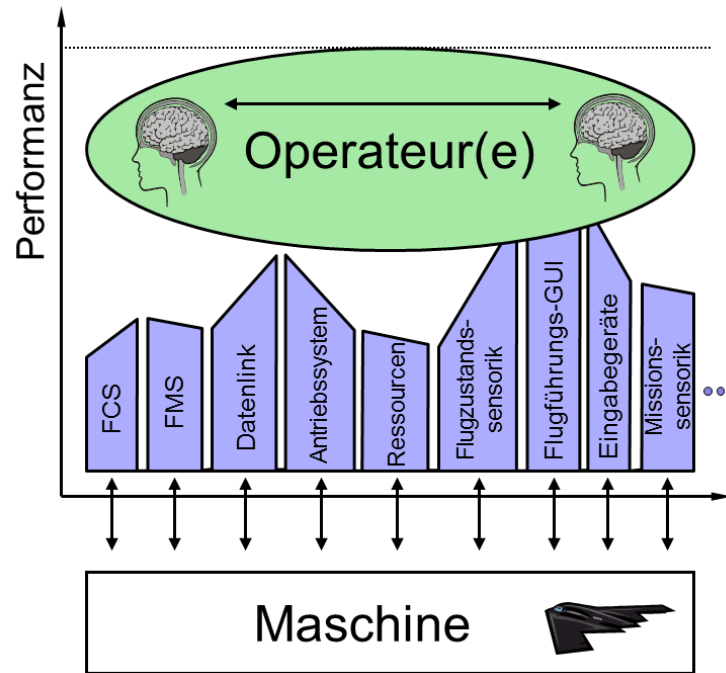


Abbildung 2-9. Der Mensch als Überwacher und Manager vieler unabhängiger Subsysteme und Automationsfunktionen

Funktionen auf Basis sogenannter konventioneller Automation ermöglichen einem Bediener ein disloziertes Fluggerät zu führen. Der Mensch fungiert dabei als Überwacher, Konfliktlöser und vor allem als Manager einer Vielzahl voneinander unabhängiger Subsysteme und Automationsfunktionen, die jeweils spezielle Teilaufgaben erledigen, für die sie entworfen und ausgelegt wurden (vgl. Abbildung 2-9). Diese unterstützen ihn jedoch meist nur in Arbeitsphasen, die an sich eine relativ geringe Belastung aufweisen und in zeitkritischen bzw. hoch dynamischen Situationen entweder keine Unterstützung bieten oder sogar einen zusätzlichen Anstieg der Belastung („clumsy automation“ – clumsiness, vgl. [Wiener, 1989]) verursachen [Sarter et al., 1997] [Onken & Schulte, 2010].

Schon während der Entwurfsphase von Automationsfunktionen muss deren Entwickler versuchen sich die Aufgaben und Situationen eines späteren Nutzers vorzustellen, um diese im Hinblick auf ein mögliches, anwendungsspezifisches Arbeitsziel auszulegen. Dies bedeutet, dass in der Designphase alle Eventualitäten und situationsbezogenen Möglichkeiten antizipiert und geplant werden müssen. Die Automation soll sich mit den späteren Arbeitszielen eines Operateurs übereinstimmend verhalten. Dabei hat konventionelle Automation jedoch keine explizite Kenntnis über die Mission und kann demnach auch keine eigenen Handlungsziele daraus ableiten. Somit kann auf eine unvorhergesehene Änderung der aktuellen Situation nicht im Sinne des Arbeitsziels reagiert werden, sondern diese Aufgabe muss von dem menschlichen Operateur bzw. Piloten wahrgenommen werden.

The crucial fact is that the automated function cannot verify on its own here and now in the course of the work process, whether it really acts in compliance with the motivational contexts of the human operator [Onken & Schulte, 2010].

An folgendem Beispiel soll dies veranschaulicht werden. Konfiguriert ein Pilot seinen Autopiloten dahingehend, dass eine vorgegebene Zielhöhe erreicht bzw. gehalten wird, führt die Automation die Aufgabe nach besten Möglichkeiten durch, auch wenn ein Hindernis, wie etwa ein Berg, im Flugweg liegt. Die Flugsicherheit, sicherlich eines der höchsten Arbeitsziele eines Piloten, ist dabei dem technischen System jedoch nicht bekannt und kann demnach auch nicht verfolgt werden. Aufgaben, die mittels konventioneller Automation von Maschinen übernommen werden, berücksichtigen bei deren Ausführung in der Regel keine etwaigen, übergeordneten Ziele und sind demzufolge anfällig für menschliche Bedienfehler. Durch den unbedachten Einsatz konventioneller Automation, ohne vorher deren Auswirkungen auf die Gesamtleistung des Arbeitssystems abzuschätzen, werden die Möglichkeiten für Bedienfehler sogar noch begünstigt.

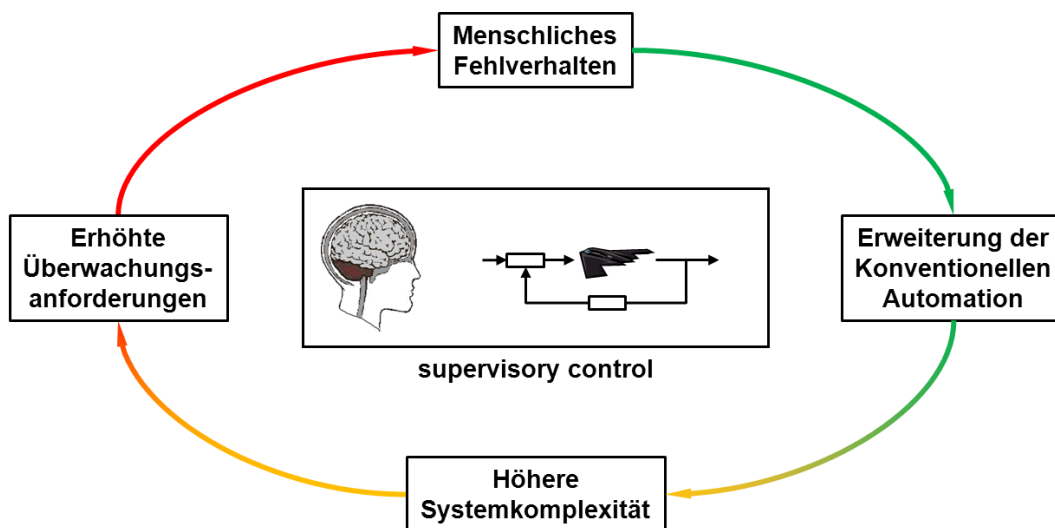


Abbildung 2-10. Teufelskreis der konventionellen Automation

Gängige Praxis ist es, bei menschlichem Fehlverhalten weitere Automationsfunktionen einzuführen und einen Teil der Autorität des Bedieners an die Maschine zu übertragen. Die Belastung des Operateurs wird infolgedessen im ersten Schritt gesenkt, jedoch muss dieser auch entsprechendes Wissen über die Art, Ausprägung und Betriebsarten der neuen Automation aufbauen, um die nachfolgende, zusätzliche Aufgabe im Sinne der Überwachung und dem Management dieser Technik wahrnehmen zu können. Einhergehend mit der dadurch steigenden Systemkomplexität wird wiederholt der Punkt eintreten, an dem der Bediener einen Fehler in der Anwendung und Überwachung des automatisierten Systems begeht. Die gängige Lösung dafür ist wiederum die Einführung weiterer, konventioneller Automation und daraus resultiert eine weiter gesteigerte Systemkomplexität. Dieser Kreislauf, (vicious circle of automation – Teufelskreis der Automation vgl. [Kriegel & Schulte, 2006] [Schulte et al., 2008] [Onken & Schulte,

2010]) in Abbildung 2-10 dargestellt, bewirkt, dass die angestrebte Zunahme der Performanz, Produktivität und Sicherheit sich reduziert und letztlich sogar zu einer Abnahme führen kann.

Der Punkt der Komplexität wurde auch von [Billings, 1997] als eine der Ursachen von Problemen, im Rahmen der Nutzung konventioneller Automationstechnologie in der bemannten Luftfahrt, identifiziert. Diese Probleme führt Billings hauptsächlich auf die folgenden fünf Automationseigenschaften zurück:

- *complexity* Ein Bediener kann den hohen Funktionsumfang und die Details der Automation und die damit einhergehende Systemkomplexität nur schwer verstehen.
- *brittleness* undefinierte bzw. falsche Handlungsweise der Automation, jenseits der spezifizierten Betriebsgrenzen.
- *opacity* Das mentale Modell des Bedieners stimmt nicht mit den Funktionen der Automation überein.
- *literalism* Die Automation führt die während des Entwicklungsprozesses programmierten und von dem Bediener aktivierten Funktionen exakt aus, ohne deren Sinnhaftigkeit im Kontext der aktuellen Situation zu überprüfen.
- *inadequate feedback* Die Automation vermittelt dem Bediener entweder gar nicht oder nur dürftig, was und warum sie gerade etwas bearbeitet bzw. was sich verändert hat.

Für eine effektive und sichere Führung eines Fluggerätes benötigt der Operateur umfassende Kenntnis über das Luftfahrzeug, die verwendete Automation, deren Leistungsgrenzen und Betriebsmodi. Hier kommt es zu Fehlern bei der Interaktion mit der Maschine, wenn durch Unkenntnis etwa die spezifizierten Betriebsgrenzen der Automation des Fluggerätes verlassen werden und sich eine undefinierte bzw. fehlerhafte Handlungsweise der Automation einstellt (*brittleness*). Ebenfalls führt es zu Fehlern wenn der Bediener durch ein falsches oder unzureichendes mentales Modell für einen bestimmten Anwendungsfall einen Betriebsmodus aktiviert, der dafür eigentlich nicht geeignet bzw. ausgelegt worden ist (*opacity*). Aufgrund der bereits oben genannten Unkenntnis des Arbeitsziels, kann konventionelle Automation dabei die aktuelle Betriebsart nicht im Kontext der vorherrschenden Situation überprüfen, sondern führt die, während des Entwicklungsprozesses programmierten Funktionen exakt aus (*literalism*). Dabei wird dem Bediener entweder gar keine bzw. nur eine sehr dürftige Rückmeldung gegeben, in welchem Modus sich die Automation gerade befindet bzw. was sich gerade verändert hat, worüber der Operateur in Kenntnis gesetzt werden sollte (*inadequate feedback*). Diese Probleme werden durch die räumliche Trennung von Bediener und Fluggerät sogar noch verstärkt, da die Operating Force nur noch vermindert mit Umgebungsinformationen des Fluggerätes versorgt wird. Dem Piloten eines bemannten Luftfahrzeugs stehen vor allem visuelle (Sehsinn), vestibuläre

(Gleichgewichtssinn) sowie auditive (Gehörsinn) und andere Informationen für die System- und Flugzustandswahrnehmung zur Verfügung. Ein UAV-Operator erhält von dem Fluggerät, das er führt, jedoch nur Daten und Informationen, welche von den Onboard-Sensoren erfasst und der Automation über die Funkverbindung bereitgestellt werden [van Erp, 2000].

As compared to the pilot of a manned aircraft, thus, a UAV operator can be said perform in relative „sensory isolation“ from the vehicle under his/her control [McCarley & Wickens, 2004].

Der Mensch kann dadurch die aktuelle Situation nicht mehr direkt mit seinen eigenen Sinnen wahrnehmen. Er bekommt durch entsprechende Displays und Anzeigen ein virtuelles Abbild des selektiv sensierten Flugzustands neben dem aktuellen Systemzustand präsentiert, das er interpretieren muss. Die Zuverlässigkeit der Sensoren spielt dabei eine große Rolle, da sich der Mensch nicht mehr direkt an Bord des Fluggerätes befindet, um den aktuellen Flugzustand mit seinen eigenen Sensoren (Sinnen) zu plausibilisieren. Dies soll folgendes Beispiel verdeutlichen:

Am 4. November 2000 stürzte ein unbemannter Hubschrauber vom Typ RQ-8A Fire Scout UAV der U.S. Navy aufgrund einer beschädigten Antenne des Radarhöhenmessers ab. Durch den Defekt wurden fehlerhafte Signale ausgesendet, die dazu führten, dass der Sensor eine Höhe von 2ft ermittelte, die jedoch nicht mit der wirklichen Flughöhe von rund 500ft über Grund übereinstimmte. Nachdem der Operator das Kommando zum automatischen Landen an das UAV gesendet hatte, sank das Fluggerät um 2ft und schaltete, entsprechend seiner Programmierung daraufhin die Triebwerke ab, da es seinen Systemzustand als gelandet interpretierte [Williams, 2004].

Die Ursache für diesen Flugunfall lag zwar in der mechanischen Beschädigung von wichtigen Komponenten des UAVs durch das Bodenpersonal, hätte jedoch bei einem korrekten Situationsbewusstsein (situation awareness) des Operators über den wirklichen Zustand des Fluggerätes, verhindert werden können.

Situation awareness is the perception of the elements in the environment within a volume of time and space, the comprehension of their meaning and the projection of their status in the near future [Endsley, 1996].

Allerdings ist nicht nur Kenntnis des aktuellen Zustands des fliegenden Systems notwendig, sondern auch die aktuelle Konfiguration der Kontrollstation respektive die Stellung und der Status der Eingabegeräte, wie im Folgenden exemplarisch aufgezeigt:

Am 25. April 2006 stürzte eine MQ-9 Predator B Drohne der Heimatschutzbehörde (engl.: Department of Homeland Security – DHS) bei einem Grenzschutzeinsatz durch die Zoll und Grenzschutzbehörde (engl.: Customs and Border Patrol – CBP) ab. Die Bodenkontrollstation sieht nach Arbeitsvorschrift zwei Bediener vor, nämlich einen Piloten und einen Sensoroperator. Hierfür sind zwei annähernd identische Bedienkonsolen vorhanden, deren Eingabegeräten jedoch unterschiedliche Funktionen – abhängig vom aktuellen Betriebsmodus (Pilot oder Sensoroperator) – zugewiesen sind. Kurz nach dem Start fiel einer der beiden Monitore der Bedienkonsole des Piloten aus,

woraufhin der Pilot die Kontrolle an die andere Konsole umschaltete, ohne jedoch die hierfür notwendige Checkliste abzuarbeiten. Diese sieht unter anderem vor, dass sämtliche Eingabegeräte inklusive einem speziellen Schalthebel (engl.: Condition lever), abgeglichen werden müssen. Dieser Schalthebel steuert, im Pilotenmodus der Konsole, die Freigabe der Kraftstoffzufuhr für das Triebwerk (Vordere Hebelstellung) und im Sensormodus die Blende der Onboardkamera (Vordere Hebelstellung öffnet die Blende, mittlere Stellung arretiert die Blende und die hintere Stellung schließt die Blende). Bei dem Umschaltvorgang war dieser Schalthebel an der zweiten Sensoroperateurkonsole in der mittleren Stellung, was dazu führte, dass die Treibstoffzufuhr zum Triebwerk abgeschaltet wurde. Der Pilot erkannte zwar relativ schnell, dass das Fluggerät nicht in der Lage war die Flughöhe zu halten, jedoch war er nicht in der Lage, das Problem zu identifizieren und zu beheben, was den Absturz des UAVs zur Folge hatte [Carrigan et al., 2008].

Das Situationsbewusstsein eines Bedieners, also sein mentales Modell des aktuellen Zustands (sowohl der KS als auch des UAVs) und der vorherrschenden Umgebungsbedingungen, ist unerlässlich für die rationale, sinnvolle Entscheidungsfindung, Führung und Überwachung von dislozierten, automatisierten Systemen.

Bei der Fernführung eines UAVs kann der Bediener durch den Einsatz von entsprechenden Technologien, wie beispielsweise Missionsmanagementsystemen, unterstützt werden. Dies dient der Verhinderung möglicher Bedienfehler und der Verbesserung der Zusammenarbeit zwischen Mensch und Maschine und deren Effizienz im Hinblick auf die Missionserfüllung. Für die Umsetzung solcher Missionsmanagementsysteme soll der folgende Abschnitt eine Übersicht über die verfügbaren Theorien, zugrunde liegenden Paradigmen und Methoden geben.

2.3 Perspektiven durch künstliche Intelligenz (KI)

Dem Bestreben nach Fluggeräten, die mit dem Menschen zusammen in unbekanntem Situationen noch zielorientiert arbeiten, die auf veränderte Umgebungsbedingungen sinnvoll reagieren, die eigene situationsangepasste Entscheidungen treffen und gegebenenfalls auch maschinelle Teams zur gemeinsamen Bearbeitung einer Aufgabe bilden, ist durch die genannten Defizite und dem Einsatz von konventionellen Automationstechnologien Grenzen gesetzt, die es zu erweitern gilt. Hier besteht der Wunsch nach mehr *Intelligenz* auf der Seite der automatisierten Fluggeräte, dem durch Anwendung von Methoden aus dem Bereich der künstlichen Intelligenz (KI) begegnet werden kann. Um den vagen Begriff der *Intelligenz* zu konkretisieren, soll im Folgenden unter

intelligentem Verhalten

das

rationale Treffen von Entscheidungen und den daraus resultierenden Handlungen, in Einklang mit dem/den Arbeitsziel/en, unter den vorherrschenden Umgebungsbedingungen und der aktuellen Ressourcenversorgung

verstanden werden. Der Einsatz dieser *Intelligenz* bezieht sich dabei auf den bestimmten, abgegrenzten Bereich einer bestimmten Domäne (in dem Fall der vorliegenden Arbeit die UAV Flugführung) und kann nach [Putzer, 2004] auch als *operative Intelligenz* bezeichnet werden. Das angestrebte Ziel ist es, durch den Einsatz von KI-Methoden die Effektivität und Leistungsfähigkeit des Mensch-Maschine-Systems zu verbessern und den bekannten Problemen entgegenzuwirken. Unbemannte Fluggeräte haben zwar keine menschliche Komponente mehr im Cockpit, dennoch ist an autonom handelnde Maschinen, insbesondere im militärischen Umfeld, ohne Führung und Interaktion durch und mit dem Menschen nicht zu denken.

An important recognition is that UAVs retain human control, albeit from outside, rather than from on-board the vehicle. Thus the operator is removed from the cockpit but not from the mission {...} [Lax & Sutherland, 1996].

Künstliche Intelligenz bezeichnet die enge Verbindung von kognitiven Leistungen und Computermodellen, die aus der Forschung im Bereich der *Kognitionswissenschaft* hervorgehen. Wesentliche Beiträge zu *Kognitionswissenschaften* liefern Psychologie, Philosophie, Informatik, Sprachwissenschaften, Neurowissenschaften und die Anthropologie (vgl. [Russel & Norvig, 2003]).

Menschliche Kognition umfasst sämtliche Wahrnehmungs-, Gedächtnis- und Denkprozesse, die menschlichem Verhalten zu Grunde liegen. Ziel ist es, Theorien und Testumgebungen für die Arbeitsweise des menschlichen Verstandes bzw. der menschlichen Informationsverarbeitung zu erhalten und diese mit Computermodellen nachzubilden. Die kognitive Psychologie, als wissenschaftliche Disziplin untersucht dabei menschliches Denken und unterscheidet zwei Ansätze zur Modellierung von Kognition. Die beiden daraus resultierenden, kontroversen Strömungen sind zum einen der *Symbolismus*, der das traditionelle Paradigma der Kognitionswissenschaft darstellt und von der expliziten Existenz von Symbolen ausgeht, die von Prozeduren (kognitive Funktionen) manipuliert bzw. verarbeitet werden und auch den Menschen als symbolverarbeitendes System interpretiert. Zum anderen vertritt der *Konnektionismus* – konträr zum *Symbolismus* – den Ansatz, dass komplexes Verhalten durch die Verbindung vieler, aber einfacher Einheiten entsteht. In solchen konnektionistischen Systemen (z.B. (künstlichen) neuronalen Netzen) gibt es keine explizite Darstellung von Symbolen, sondern die Verarbeitung erfolgt verteilt und parallel in allen Einheiten (vgl. [Rumelhart, 1986]). Fortführend gibt es noch den Bereich des *künstlichen Lebens*, der wahre Intelligenz als eine sich ergebende Eigenschaft von künstlichen Lebensformen ansieht [Adami, 1998].

Jede Strömung entwickelte *Architekturen*, welche die jeweilige Theorie in nutzbare Computermodelle umsetzen. Eine der prominentesten symbolistischen, kognitiven Architekturen ist Soar (State, Operator and Result) [Laird et al., 1987] [Newell, 1990] [Lehman et al., 1998]. Soar wurde mit dem Ziel der Nachbildung menschlicher Kognition entworfen und wird hauptsächlich für die Modellierung von rationalem Verhalten genutzt [Jones, 1996]. Der Kern basiert auf einem Produktionensystem, das

aus einem Langzeitspeicher, in dem Applikationswissen (a-priori Wissen) in Form von Regeln hinterlegt ist, und einem Kurzzeitspeicher (Arbeitspeicher – Working memory – WM) besteht. Situatives Wissen wird symbolisch als semantisches Netz repräsentiert, von dem auch Teile für die Interaktion mit der Umgebung reserviert sind.

Ein weiterer Vertreter einer kognitiven Architektur ist ACT-R (Adaptive Control of Thought – Rational) [Anderson et al., 2004] [ACT-R Homepage], deren Schwerpunkt auf der Modellierung menschlicher Kognition mit ihren Stärken und Schwächen liegt, um Vorhersagen über menschliches Verhalten treffen zu können. ACT-R basiert ebenso wie Soar auf einem Produktionensystem, dessen prozedurales Wissen als Regeln hinterlegt ist. Der symbolischen Repräsentation von Fakten und Prozeduren werden subsymbolische Informationen zur Seite gestellt, welche die Verwendung des symbolischen Wissens bestimmen, um Effekte wie Vergessen und menschliches Fehlverhalten modellieren zu können.

Im Bereich der Agententechnologie gibt es eine Vielzahl zur Verfügung stehender Architekturen, wie PRS (*Procedural Reasoning System*), dMARS (*distributed Multi-Agent Reasoning System*) und auch kommerzielle Entwicklungsumgebungen wie JACKTM der Firma Agent Oriented Software Group. Diese nutzen die BDI-Theorie (*Belief-Desire-Intention*) [Rao & Georgeff, 1995], auf deren Basis rationales Verhalten von Agenten erzeugt werden kann und in der Lage ist, Ziele explizit zu repräsentieren.

Im folgenden Abschnitt soll eine weitere Theorie für die Abbildung von wissensbasiertem Verhalten mit einer auf dieser Theorie begründeten Architektur vorgestellt werden.

2.4 Kognitive Automation

Kognitive Automation ist ein weiterer Ansatz, der (künstliche) Fähigkeiten von Automation in einem Mensch-Maschine-System beschreibt und hierfür auch entsprechende Richtlinien bzw. Anforderungen definiert. Nach [Onken & Schulte, 2010] muss *kognitive Automation* in der Lage sein, im Gesamtkontext einer Aufgabe bzw. Mission zielgerichtet zu handeln. Voraussetzung hierfür ist unter anderem ein Verständnis der aktuellen Situation sowie des Arbeitszieles, um bewerten zu können, welche Teilaufgaben noch bearbeitet werden müssen bzw. in wie weit laufende Aufgaben schon bearbeitet wurden. Im Falle von unvorhergesehenen Ereignissen sollen entsprechende Maßnahmen bzw. Handlungen im Kontext der aktuellen Arbeitsziele initiiert werden. Durch Berücksichtigung von explizit modellierten Zielen wird ein Verständnis über die nötige Sequenz von Handlungen aufgebaut, um die gestellte Aufgabe als Teil des Gesamtauftrags zu erfüllen. Die Entscheidungen über das Auslösen von Handlungen erfolgt dabei stets rational, also unter Verwendung des gesamten, vorhandenen Wissens. Neben der Differenzierung zwischen wichtigen und unwichtigen Informationen sowie dringenden bzw. weniger dringenden Handlungen wird die automatische Durchführung dieser Handlungen vorgenommen, sofern dies vom Menschen autorisiert wurde. Zudem soll eine effektive Kommunikation mit anderen Einheiten, die für die Durchführung einer Mission relevant sind, veranlasst werden.

Um das Verhältnis und die Wechselbeziehung in Mensch-Maschine-Systemen anschaulich beschreiben und analysieren zu können, wird nachfolgend das Arbeitssystem nach [REFA, 1984] verwendet und anschließend die Einführung *kognitiver Automation* erläutert.

2.4.1 Das Arbeitssystem

Abbildung 2-11 zeigt ein Arbeitssystem mit seinen wesentlichen Elementen. Diese sind der/die Bediener (nach [Onken & Schulte, 2010] als Operating Force – OF bezeichnet) und die Arbeitsmittel (nach [Onken & Schulte, 2010] als Operation Supporting Means – OSM bezeichnet). Die Aufgabe der Operating Force ist es, (durch Einsatz und Bedienung der Operation Supporting Means und das Treffen von Entscheidungen), die Mission unter den aktuellen Umgebungsbedingungen zu erfüllen bzw. das Arbeitsziel zu erreichen. Dabei gilt es auftretende Probleme zu lösen und den Zustand der OSM zu überwachen. Auf der Seite der Operating Force kann die Aufgabe der Missionserfüllung auch von mehreren Menschen und technischen Systemen, welche zusammen im Team arbeiten, wahrgenommen werden. Per Definition kann jedoch ein Arbeitssystem nur bestehen, wenn mindestens eine menschliche Komponente in der Operating Force vorhanden ist (vgl. [Kriegel & Schulte, 2006] [Meitinger, 2008] [Onken & Schulte, 2010]). Zudem kann nur ein menschlicher Bediener seine Mission bzw. das Arbeitsziel verändern und beeinflussen, was nach [Onken & Schulte, 2010] das Kriterium für Autonomie ist.

Die Arbeitsmittel umfassen die technische Ausstattung, die der Operating Force zur Verfügung steht. Hierzu zählen die mechanischen Komponenten und Automationsfunktionen, die zwar bestimmte Teilaufgaben übernehmen, dabei jedoch keine Kenntnis der Mission haben und somit das Arbeitsziel nicht direkt verfolgen.

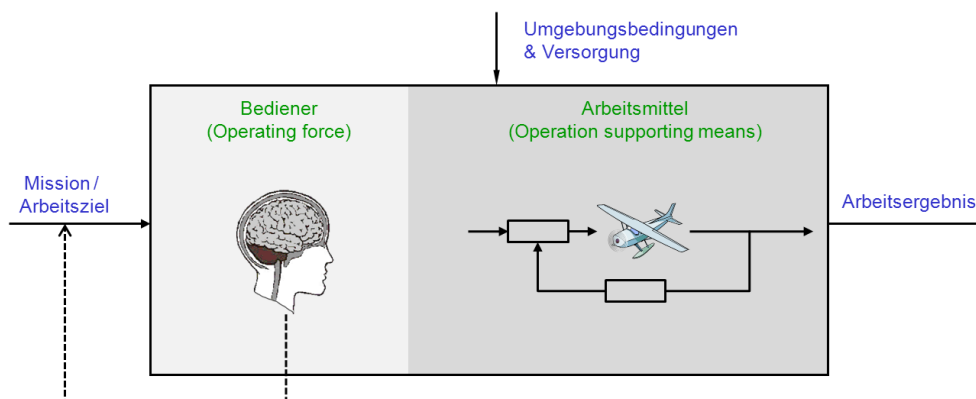


Abbildung 2-11. Das Arbeitssystem

Die Mission bzw. das Arbeitsziel (Eingangsgröße des Arbeitssystems) wird gängiger Weise als Auftrag formuliert und definiert das Arbeitsergebnis (Ausgangsgröße des Arbeitssystems), das es unter den vorherrschenden Umgebungsbedingungen und dem aktuellen Versorgungszustand zu erreichen gilt.

2.4.2 Kognitive Automation im Arbeitssystem

Betrachtet man die Möglichkeiten, kognitive Automation in das Arbeitssystem einzuführen, so ergeben sich nach [Schulte et al., 2008] und [Onken & Schulte, 2010] zwei Fälle (engl. *Dual mode*), nämlich die Einführung auf Seiten der Operation Supporting Means (Mode 1) und der Operating Force (Mode 2, vgl. Abbildung 2-12). Zusätzliche, kognitive Automation in Form von sogenannten künstlichen kognitiven Einheiten (*Artificial Cognitive Unit – ACU*) auf Seiten der Arbeitsmittel muss dabei von der Operating Force überwacht werden, während zusätzliche Automation auf Seiten der Operating Force mit anderen Elementen der Operating Force (z.B. einem menschlichen Operateur) kooperieren muss (vgl. [Meitinger, 2008] [Onken & Schulte, 2010]).

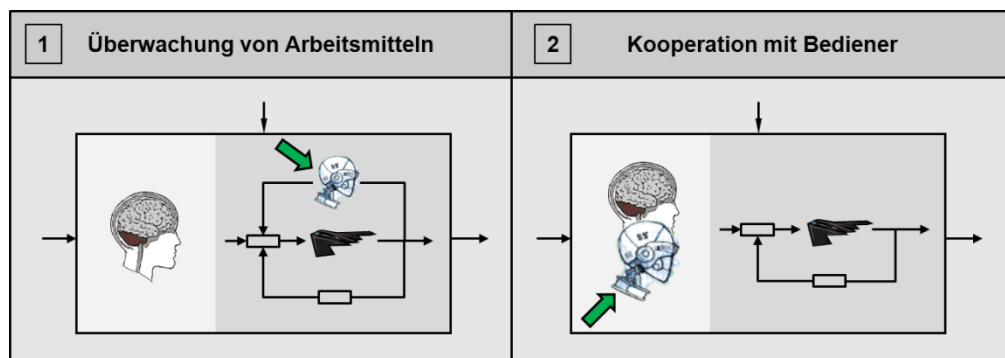


Abbildung 2-12. Einbringen von kognitiver Automation in das Arbeitssystem

ACUs als Teil der Operation Supporting Means werden als Supporting Cognitive Units (SCUs) bezeichnet. Die Einführung in das Arbeitssystem erfolgt üblicherweise im Rahmen eines künstlichen semi-autonomen Systems (vgl. Abbildung 2-13). Hier soll nochmals darauf hingewiesen werden, dass ein semi-autonomes System kein Arbeitssystem ist, da es keine menschliche Komponente beinhaltet, und wird deshalb immer in ein übergeordnetes Arbeitssystem eingebettet. Zudem ersetzt die SCU dabei nicht die schon vorhandene (konventionelle) Automation, sondern nutzt die anderen technischen Systeme, um die von der Operating Force gegebene Aufgabe zu bearbeiten. Im Gegensatz zur konventionellen Automation besitzt die SCU explizit modellierte Arbeitsziele, um die Anweisungen des Auftraggebers bestmöglich umzusetzen und in komplexen Umgebungen und bei unvorhergesehenen, im Designprozess nicht antizipierten Ereignissen und Zuständen sinnvoll und zielgerichtet agieren zu können (vgl. [Kriegel & Schulte, 2006] [Meitinger & Schulte, 2006b] [Kriegel et al., 2007] [Onken & Schulte, 2010]).

Demzufolge kann den in Abschnitt 2.2 benannten Defiziten konventioneller Automation wie *brittleness*, *literalism* und *clumsiness* teilweise und *opacity* gänzlich begegnet werden [Meitinger, 2008]. Die Komplexität der von der Operating Force zu überwachenden Arbeitsmittel wird hingegen zunächst durch die Einführung zusätzlicher Automation weiter erhöht und kann im ersten Schritt auch nicht verhindert werden. Jedoch kann die Handlungsweise einer komplexen ACU einem Operateur durch die

Berücksichtigung expliziter Handlungsziele und der Selbsterklärungsfähigkeit als eine der Eigenschaften von kognitiver Automation verständlich gemacht werden.

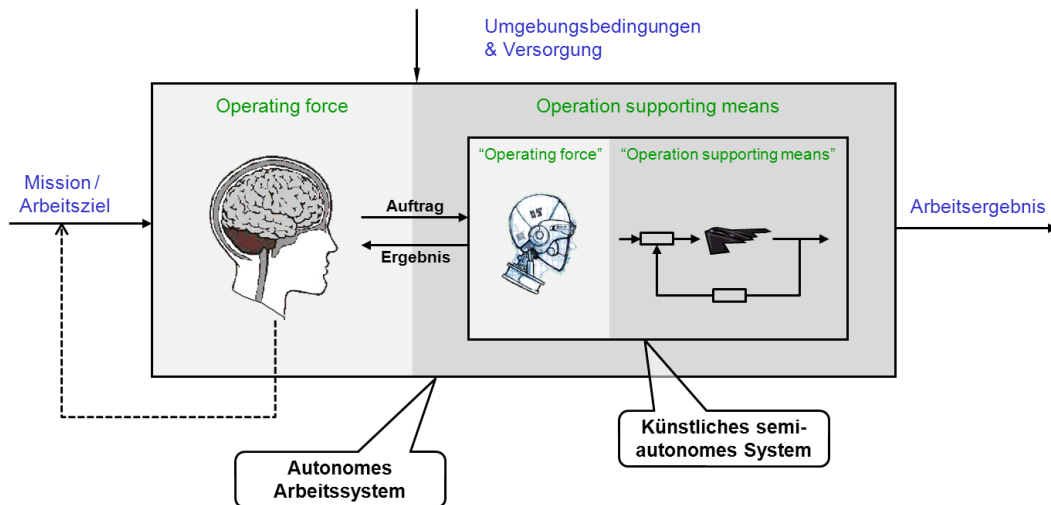


Abbildung 2-13. Semi-autonomes System als Arbeitsmittel eines übergeordneten Arbeitssystems

ACUs als Teil der Operating Force werden als Operating Cognitive Units (OCUs) bezeichnet und übernehmen im Gegensatz zu SCUs eine aktive Rolle bei der Durchführung einer Mission. Voraussetzung dafür ist die Kenntnis des übergeordneten Arbeitsziels und die Kooperation mit anderen Elementen der Operating Force. Dazu kann die OCU im Prinzip über alle vorhandenen Arbeitsmittel verfügen, auch wenn SCUs bzw. semi-autonome Systeme Teil davon sind. In diesem Zusammenhang ergeben sich für eine OCU zwei verschiedene Aufgabenschwerpunkte. Nimmt sie die Position eines Mitglieds eines menschlichen Teams ein (Substitution), so besteht ihre Hauptaufgabe in der Durchführung von Teilaufgaben im Rahmen des Arbeitsziels. Wird sie jedoch zusätzlich zu einem oder mehreren Menschen in die Operating Force eingebracht (Ergänzung), ist sie als Assistenzsystem ausgeprägt und hat als Hauptaufgabe, die Operating Force bei der Durchführung von Teilaufgaben im Rahmen der Erreichung des Arbeitsziels zu unterstützen [Onken, 1994] [Kriegel & Schulte, 2006] [Meitinger, 2008] [Onken & Schulte, 2010].

2.4.3 Der Kognitive Prozess

Um eine solche *Artificial Cognitive Unit*, das heißt einen Agenten mit kognitiven Fähigkeiten realisieren zu können, wird sowohl ein zugrundeliegendes Paradigma als auch ein Architekturansatz, ähnlich den zuvor beschriebenen kognitiven Architekturen Soar und ACT-R, benötigt. Folgend der Theorie der *Kognitiven Automation* wurde das Paradigma des *Kognitiven Prozesses* (KP) [Putzer & Onken, 2003] [Putzer, 2004] [Frey, 2005] [Onken & Schulte, 2010] entwickelt, welches in Abbildung 2-14 dargestellt ist. Basierend auf dem von [Rasmussen, 1983] vorgeschlagenen Modell menschlichen Verhaltens werden explizit modellierte Ziele berücksichtigt, die die Erzeugung von rational, zielgerichtetem Verhalten im Umfeld einer sich dynamisch verändernden Situation erlauben.

Der KP besteht aus einem Rumpf (Wissen; ovaler Bereich) und den *KP-Transformatoren* (Pfeile; wissensverarbeitende Funktionalitäten), wobei das Wissen sich in *a-priori Wissen* und *Situationswissen* unterteilen lässt. Das *a-priori Wissen* wird von einem Entwickler des kognitiven Systems modelliert und besteht aus Umweltmodellen (engl.: *environment models*), Wünschen (engl.: *desires*), Handlungsalternativen (engl.: *action alternatives*) und Anweisungsmodellen (engl.: *instruction models*), die jeweils als eigene Klassen im Sinne der objektorientierten Programmierung formuliert werden. Das Situationswissen entsteht durch Verarbeitung des *a-priori Wissens*, zur Laufzeit des Systems, mittels der Transformatoren und wird als Instanzen der definierten Klassen repräsentiert. Alle Transformatoren haben prinzipiell lesenden Zugriff auf das gesamte Situationswissen (zentrale Wissensrepräsentation), verarbeiten jedoch nur das Wissen in einem bestimmten (gestrichelte Bereiche in Abbildung 2-14) Bereich, wodurch neues Situationswissen erzeugt wird. Die Darstellung vermittelt zwar den Eindruck, dass die Transformatoren das Wissen sequentiell verarbeiten, tatsächlich erfolgt die Verarbeitung jedoch parallel und ereignisgesteuert, wodurch in jeder Situation das gesamte, zentral repräsentierte, situationsrelevante Wissen zum Einsatz kommt [Putzer, 2004] [Meitinger, 2008].

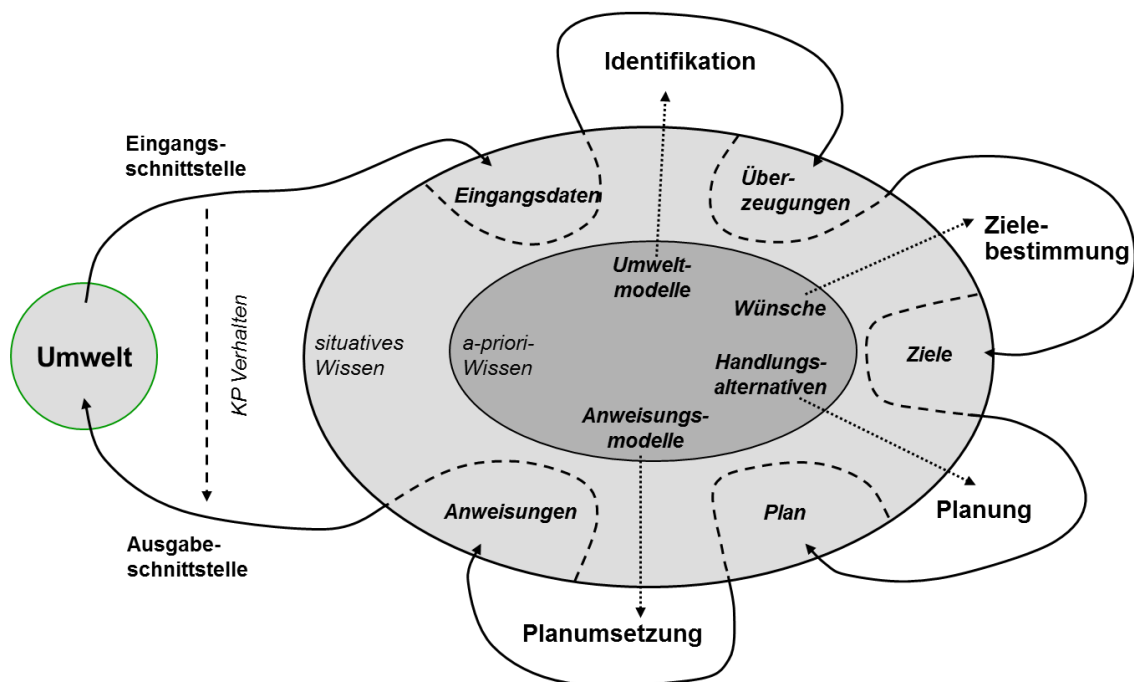


Abbildung 2-14. Der Kognitive Prozess

Die Eingangsschnittstelle sammelt alle in der Umwelt verfügbaren Informationen und stellt diese dem künstlichen kognitiven System als Eingangsdaten zur Verfügung. Die Interaktion mit der Umwelt erfolgt über die Ausgabeschnittstelle. Durch Betrachtung der Eingangs- und Ausgangsschnittstelle kann das sogenannte Makroverhalten des KPs wahrgenommen werden, während das Zusammenspiel der einzelnen a-priori Wissensmodelle das Mikroverhalten darstellt.

Das Paradigma des KPs ermöglicht die Ausrichtung des erzeugten Verhaltens an Zielen und berücksichtigt den wissensbasierten Ansatz im Sinne der Informatik (Trennung von

applikationsspezifischem Wissen und applikationsunabhängiger Verarbeitung des Wissens) [Putzer, 2004], sowie die Repräsentation und Verarbeitung des Wissens auf symbolischer Ebene. Somit wird das Makroverhalten des Kognitiven Prozesses einzig durch das a-priori Wissen und die Architektur definiert, wodurch das Schema des KPs für die Abbildung von maschinellen Denkprozessen allgemeine Gültigkeit hat. Die Modellierung des Wissens erfolgt analog zu der Semantik der menschlichen Informationsverarbeitung, in Anlehnung an das Modell von Rasmussen, und ermöglicht dadurch einerseits eine intuitive Ontologie-Erstellung eines Wissensingenieurs, und andererseits ergibt sich eine leicht zu verstehende und selbsterklärungsfähige Schnittstelle zu einem menschlichen Bediener.

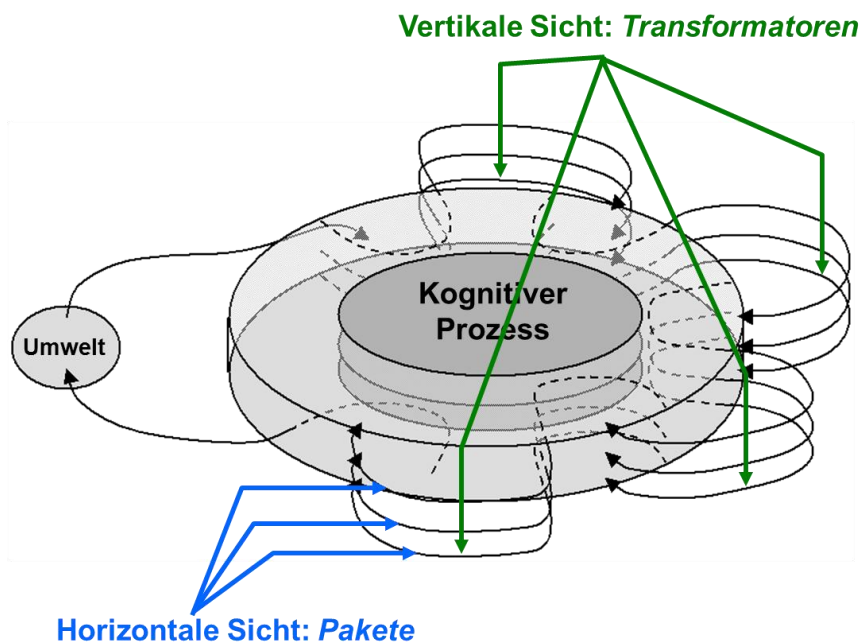


Abbildung 2-15. Sichtweisen auf den Kognitiven Prozess

Bei der Entwicklung komplexer Systeme ist eine Modularisierung von einzelnen Systemfunktionen bzw. Systemfähigkeiten wünschenswert, die mit dem Paradigma des Kognitiven Prozesses architekturell umgesetzt werden kann. Neben der vertikalen Sicht auf den Kognitiven Prozess wie in Abbildung 2-14 dargestellt, können mehrere Schichten (synonym auch als Pakete bezeichnet) mit schematisch gleichem Aufbau (Anordnung der Transformatoren) kombiniert werden (vgl. Abbildung 2-15). Die Kommunikation der einzelnen Pakete erfolgt durch Verweise im a-priori Wissen und das zentrale, für alle Pakete verfügbare Situationswissen. Als Folge können damit entsprechende Funktionalitäten gekapselt, nach Bedarf und Anforderungen wiederverwendet und mit anderen Paketen zusammengeführt werden [Putzer, 2004].

2.4.4 Kognitive Systemarchitektur – COSA

Die von [Putzer, 2003, 2004] entwickelte kognitive Systemarchitektur COSA (engl.: Cognitive System Architecture) stellt, basierend auf dem Paradigma des Kognitiven Prozesses, ein Framework zur Verfügung, welches schematisch in Abbildung 2-16

aufgezeigt wird. Es erlaubt die applikationsunabhängige Entwicklung von Wissenspaketen (a-priori Wissen) mit der *Cognitive Programming Language (CPL)* [Putzer, 2004], die eine objektorientierte Erweiterung der Soar-Syntax darstellt und die Modellierung von Wissen auf der Abstraktionsebene mentaler Begriffe in Anlehnung an die Nomenklatur des Kognitiven Prozesses, ermöglicht.

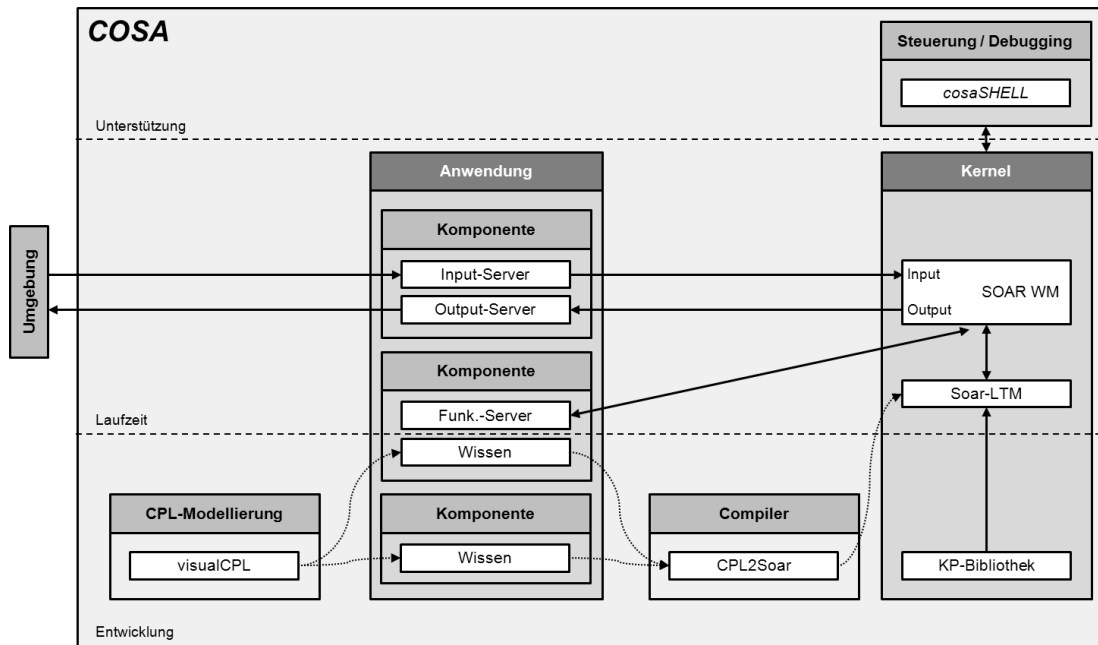


Abbildung 2-16: Überblick über die Kognitive Systemarchitektur COSA nach [Meitinger, 2008]

Zentrales Element von COSA ist der Kernel, welcher Soar (vgl. 2.3) als Prozessor nutzt und um eine KP-Bibliothek erweitert. Die Soar-Inferenzmaschine erlaubt jedoch nur die Verarbeitung von nativen Soar-Produktionen (Regeln). Demnach muss in CPL formuliertes Wissen noch mit einem speziellen Compiler (*CPL2Soar*) in das Soar Format überführt werden. Hierbei wird jedoch kein ausführbarer Code (weder eine ausführbare Datei noch eine Bibliothek) erzeugt. Vielmehr wird eine Anpassung und Erweiterung der CPL Syntax vorgenommen, um das a-priori Wissen in Soar-Produktionen zu übersetzen, welche dann zur Laufzeit in den COSA-Kernel geladen und im Langzeitspeicher (engl.: Long Term Memory – LTM) des Soar-Kernels hinterlegt werden können. Für die Entwicklung stehen Dienstprogramme zur Verfügung, wie etwa die *cosaSHELL* für die Steuerung und Fehlersuche (Debugging) des Kernels. Zudem bietet die *cosaSHELL* die Möglichkeit zur Laufzeit das situative Wissen des kognitiven Systems zu betrachten. Des Weiteren stellt COSA *visualCPL* bereit – eine Benutzeroberfläche (engl.: Graphical User Interface – GUI) zur graphischen Modellierung von CPL-Wissen. Die tatsächliche Applikation enthält mehrere Komponenten, welche sich aus Wissen und/oder Servern für die Kommunikation und Interaktion mit der Umgebung (Input-/Output-Server) sowie zur Berechnung numerischer Größen (Function-Server) zusammensetzt (vgl. Abbildung 2-16). Eine bestehende Anwendungsschnittstelle (engl.: Application Programming Interface – API) bietet entsprechende Funktionen für das Ankoppeln der Server (zur Laufzeit) und ermöglicht sowohl die Übertragung von Eingangsdaten aus der

Umgebung in den Soar Arbeitsspeicher (engl.: Working Memory – WM) als auch den Transfer von Ausgangsdaten für die Interaktion mit der Umgebung.

Im Hinblick auf eine Verbesserung der Laufzeitgeschwindigkeit und Leistungsfähigkeit von COSA wurde von [Matzner, 2010] die bestehende Implementation untersucht und der Kernel (bisher Soar) durch einen graphenbasierten Algorithmus zur Mustersuche in Produktionensystemen ersetzt. Die daraus resultierende Steigerung der Performanz basiert vor allem auf einer Optimierung des Suchplans mittels genetischer Algorithmen zur effizienten Erkennung von Graphmustern. Um die beiden Systeme in puncto Leistungsfähigkeit zu vergleichen, wurde eine Benchmark Mission durchgeführt. Die in [Matzner et al., 2008] präsentierten Ergebnisse zeigen eine Verringerung der durchschnittlichen Zeit für das Treffen einer Entscheidung von 97.5 Prozent der neu implementierten Algorithmen im Vergleich zu dem Original COSA von Putzer.

2.4.4.1 Weiterentwicklung - COSA²

Basierend auf der Implementierung von Matzner läuft derzeit an der UniBwM eine Weiterentwicklung im Rahmen der Erweiterung und Verbesserung der Planungsfähigkeiten. COSA² (Cognitive System Architecture with Centralized Ontology and Specific Algorithms, vgl. [Brüggenwirth et al., 2011]) orientiert sich dabei an der Interpretation des Modells der menschlichen Informationsverarbeitung auf drei Ebenen von J. Rasmussen nach [Onken & Schulte, 2010]. Abbildung 2-17 zeigt die einzelnen Schichten, welche die verschiedenen Verhaltensebenen unterscheiden, nämlich das fertigkeitbasierte Verhalten (hier: *Skill-based behaviour*), das regelbasierte Verhalten (hier: *Procedure-based behaviour*) und das wissensbasierte Verhalten (hier: *Concept-based behaviour*).

Die grau hinterlegten Blöcke in Abbildung 2-17 stellen einzelne kognitive Subfunktionen (engl.: Cognitive Sub-Function – CSF) dar, welche Eingangsdaten (eingehender Pfeil mit roter Beschriftung) – unter Anwendung des zugehörigen a-priori Wissens (Pfeil mit blauer Beschriftung) – in Ausgangsdaten (ausgehender Pfeil mit roter Beschriftung) überführen. Dem Prinzip der zentralen Wissensrepräsentation folgend, können die Funktionen generell im gesamten situativen Wissen lesen (= Eingangsdaten), jedoch werden die Ausgangsdaten ausschließlich in den dafür vorgesehenen Bereich geschrieben. Ausgangspfeile mit roter Beschriftung bezeichnen dabei symbolisch repräsentiertes Wissen, während die grünen Pfeile subsymbolische Informationen (Un- bzw. vorverarbeitete Wahrnehmungsdaten oder Aktionskommandos für die Interaktion mit der Umwelt) transportieren. Die folgenden Absätze beschreiben nach [Brüggenwirth et al., 2011] die wesentlichen Funktionen und Abläufe innerhalb von COSA².

Auf der untersten, fertigkeitbasierten Verhaltensebene werden die subsymbolisch repräsentierten Wahrnehmungsdaten (z.B. aktueller Betriebszustand der Automation, Sensordaten) durch die CSF „*Feature Formation*“ in aussagekräftige Symbole abstrahiert. Eine typische Regel des „*Cue Models*“-a-priori Wissens könnte zum Beispiel besagen, dass eine Entfernung des Fluggerätes kleiner zehn Meter ($d < 10m$) zu dem aktiven Wegpunkt in ein Symbol „Wegpunkt erreicht“ übersetzt wird, welches

nachfolgend dann im situativen Wissen abgelegt (*identification relevant formations*) und vornehmlich von den CSFs „*Identification*“ und „*Task Determination*“ verarbeitet wird.

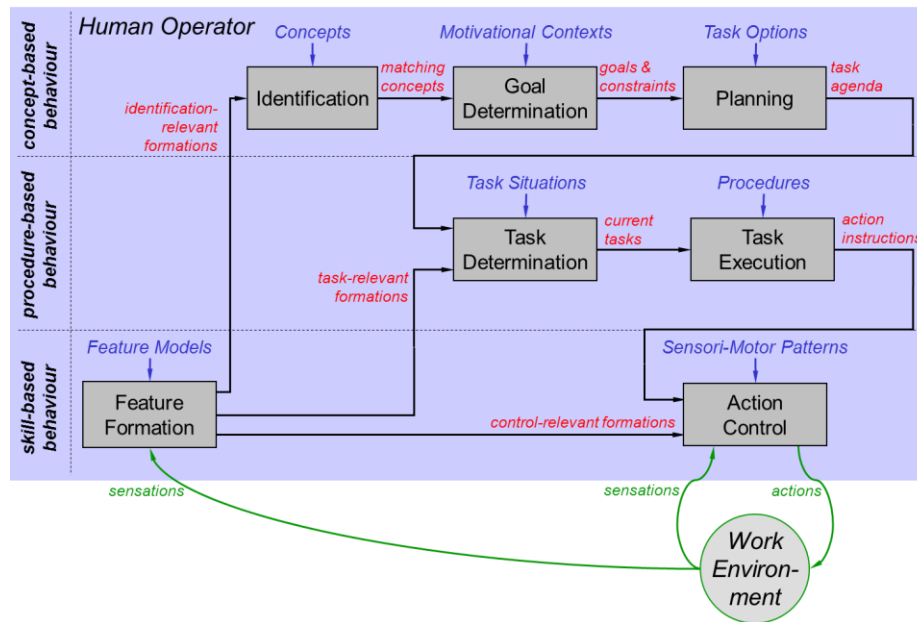


Abbildung 2-17. Interpretation des Modells von Rasmussen zur menschlichen Informationsverarbeitung auf drei Ebenen [Onken & Schulte, 2010]

Die CSF „*Identification*“ ist für den Aufbau des maschinellen Situationsbewusstseins verantwortlich. Hierfür werden die Symbole der „*Feature Formation*“ zusammengefügt und durch Anwendung des a-priori Wissens „*Concepts*“ entsprechend zu einem breiteren und umfassenderen Lagebild (*matching concepts*) verknüpft.

Die wissensbasierte Verhaltensschicht von COSA² arbeitet zielebasiert. Analog zu den „*Concepts*“ werden Ziele (*Motivational Contexts*) als Klassen beschrieben, die sowohl mit Attributen und spezifischen Informationen über das Ziel als auch mit Regeln versehen sind. Diese beschreiben, unter welchen Voraussetzungen das Ziel aktiviert werden soll. Die Semantik ist hier, dass dann eine Instanz von einem Ziel im Arbeitsspeicher (*goals & constraints*) erzeugt wird, wenn der aktuelle Zustand nicht dem gewünschten Zustand entspricht.

Planung beschreibt das Problem der Erzeugung einer Abfolge von Aufgaben, welche der Agent ausführen muss, um von dem aktuellen Zustand in den gewünschten Soll-Zustand zu gelangen. Der derzeitige Zustand ist in erster Linie durch die „*matching concepts*“ beschrieben, während sich der Soll-Zustand aus den „*goals & constraints*“ ergibt. Die Menge der möglichen Aktionen des Agenten stellen die „*Task Options*“ dar und das Resultat einer erfolgreichen Planung ist eine „*task agenda*“. Während der Abarbeitung der Aufgabenagenda prüft die CSF „*Planning*“, ob die Aufgabenagenda noch Gültigkeit besitzt und auch noch konform mit den aktuellen Zielen ist.

Die Erstellung der Aufgabenagenda ist durch einen Vorausplanungsalgorithmus realisiert, der den Effekt der verfügbaren Handlungsalternativen in die Zukunft projiziert. Hierfür werden die erwarteten Umweltreaktionen durch eine simulierte

Umgebung (engl.: Simulated Environment) ersetzt und der resultierende Zustand, wie er von dem Agenten wahrgenommen werden würde (Verwendung des a-priori Wissens „*Concepts*“ und „*Motivational Contexts*“, vgl. Abbildung 2-18), vorgegeben. Für diesen Vorgang wird das identische Wissen verwendet, welches ebenfalls für die Bewertung der realen, wahrgenommenen Welt genutzt wird. Dadurch wird die Konsistenz zwischen den zukünftigen, vorhergesagten Zuständen und den real eintretenden Zuständen, bei Ausführung der Aufgabenagenda, gewahrt.

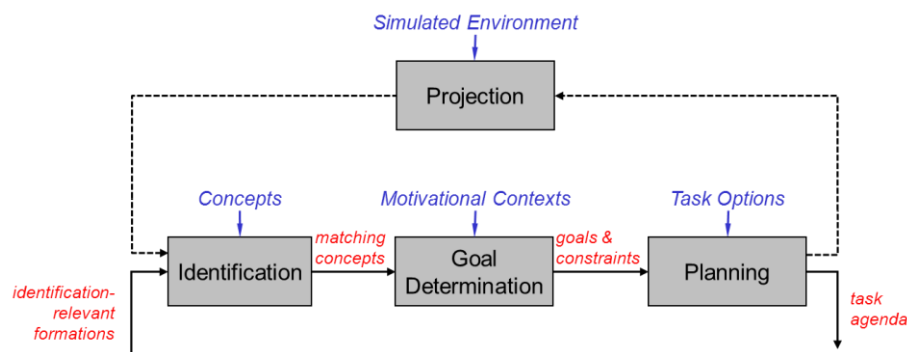


Abbildung 2-18. Ansatz des Voraussplanungsalgorithmus in COSA²

Nach [Rasmussen, 1983] ist das menschliche Verhalten von alltäglichen und routinierten Handlungsweisen dominiert. Dieses regelbasierte Verhalten erhöht die Performanz in Umgebungen mit wiederkehrenden Situationen („*Task Situations*“), die wiederum entsprechende Handlungsabläufe als Reaktion auslösen. Verantwortlich hierfür ist die CSF „*Task Determination*“, welche nicht nur die Umstände sondern auch bestimmte Situationsmuster selbst identifiziert und entsprechende, hinterlegte Handlungsweisen initiiert. Diese regelbasierte Schicht ist für die Abarbeitung der Aufgabenagenda verantwortlich und durch Auswertung der „*task-relevant cues*“ kann identifiziert werden, ob eine Aufgabe durchgeführt und mit der Bearbeitung der nächsten Aufgabe der Liste begonnen werden kann. Neben der sequentiellen Bearbeitung von Aufgaben wird auch die simultane Ausführung von nebenläufigen Maßnahmen unterstützt.

Eine Aufgabe kann sich jedoch aus mehreren Einzelaktionen zusammensetzen, welche durch die Handlungsalternativen des Agenten repräsentiert sind und von der CSF „*Task Execution*“, in Form von korrespondierenden Anweisungen und Instruktionen („*action instructions*“), für diese Aktionen veranlasst werden. Durch Verwendung des a-priori Wissens „*Procedures*“ können Aufgaben mit entsprechenden Sequenzen von solchen atomaren Aktionen in Beziehung gesetzt werden. Mittels der CSF „*Action Control*“ werden die symbolischen Aktionen in subsymbolische, numerische Anweisungen übersetzt und über die Schnittstelle zur Umwelt ausgegeben.

Das COSA von Putzer sieht zwar die Planung auch als eigenen, kognitiven Schritt an, stellt hierfür jedoch keine direkte algorithmische Unterstützung bereit. Vielmehr muss der Entwickler hierfür eigene Planungsfunktionen schaffen, wie beispielsweise die Mittel-Ziel-Analyse (engl.: Means-Ends-Analysis) bei [Meitinger, 2008]. Im Vergleich bietet COSA² eine erheblich gesteigerte Planungsfunktionalität und damit mehr

Flexibilität für die Entwicklung eines intelligenten Agenten. Zudem hat der Wissensingenieur keinen zusätzlichen Mehraufwand, da für den Vorgang der Planung Wissen zu Anwendung kommt, das es ohnehin zu modellieren gilt.

Die Modellierung des a-priori Wissens für COSA² basiert auf der Programmiersprache CML (Cognitive Modeling Language), eine Weiterentwicklung von CPL [Meitinger et al., 2009], in Anlehnung an UML (Unified Modeling Language) und XML (Extensible Markup Language). Zudem wurden entsprechende Erweiterungen eingebracht, um nicht nur die Anforderungen des Kognitiven Prozesses zu erfüllen sondern auch eine Trennung zwischen dem Datenmodell und dessen Repräsentation, mit dem Ziel einer automatisierten Erzeugung von Code und den zugehörigen Dokumenten.

Aufgrund der sich noch in der Entwicklung befindlichen, neueren Version der kognitiven Systemarchitektur COSA² und deren eingeschränkter Verfügbarkeit wurde jedoch für die Umsetzung einer ACU im Rahmen dieser Arbeit das COSA von Putzer verwendet.

2.5 Problemstellung

Bei bemannten Fluggeräten sitzt immer ein Mensch im Cockpit, der seinen Auftrag von einer übergeordneten Instanz (z.B. Command and Control – C²) erhält. Abbildung 2-19 zeigt ein solches Arbeitssystem, in dem der Pilot die *Operating Force* und das Fluggerät mit der verfügbaren Automation die *Operation Supporting Means* darstellen. Der Auftrag bzw. die Mission wird von dem Menschen im Cockpit verstanden, interpretiert und in dementsprechende Handlungen umgesetzt, welche im besten Fall zu einer Erfüllung der gegebenen Aufgabe führen, jeweils unter den vorherrschenden Umgebungs- und Versorgungszuständen.

Im Gegensatz ist jedoch bei UAVs keine menschliche Komponente mehr an Bord, die diese Funktion übernimmt. Um den in Abschnitt 2.2 genannten, möglichen Problemen bei der Führung durch Überwachung und der parametergetriebenen Wegpunktnavigation zu begegnen und den UAV-Operateur bei der Durchführung seiner Mission zu unterstützen, soll ein intelligenter Agent – ausgeprägt als eine *Artificial Cognitive Unit (ACU)* – eingesetzt werden.

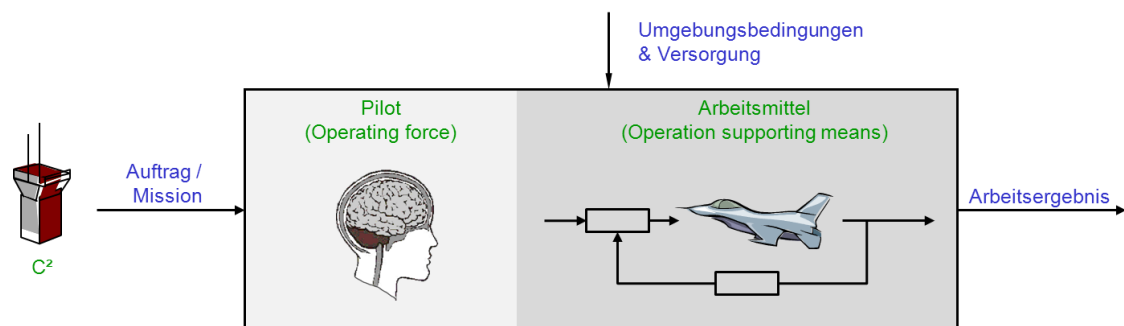


Abbildung 2-19. Arbeitssystem eines bemannten Flugzeugs

Abbildung 2-20 zeigt ein solches Arbeitssystem mit einem UAV-Operateur als Bediener (*Operating Force*) und einem semi-autonomen System als Teil der

Arbeitsmittel (*Operation Supporting Means*). Durch den Einsatz der ACU soll eine Interaktion auf einem höheren Abstraktionsniveau ermöglicht werden, wie beispielsweise die auftragsbasierte Führung. Daraus können zwei grundsätzliche Anforderungen an den intelligenten Agenten abgeleitet werden:

- **Verständnis des Auftrags:** Die ACU muss in der Lage sein, den *symbolisch* formulierten Auftrag zu erfassen, daraus einen Vorgehensplan zu entwickeln, situationsangepasste Entscheidungen zu treffen und die geeigneten Handlungen für die Erreichung der Zielsetzung durchzuführen.
- **Verständnis der Maschine:** Die ACU muss das UAV bedienen können und die subsymbolische Welt des UAVs mit einer Vielzahl an Parametern, Zahlenwerten und die verfügbare Automation mit ihren Betriebsarten verstehen. Zudem gilt es, die Maschine mit ihren funktionalen und hardware-spezifischen Eigenschaften und deren Zustand zu überwachen.

Demnach benötigt die ACU zum einen Wissen, um die von der *Operating Force* formulierte Aufgabe zu verstehen, und zum anderen Wissen über die Eigenschaften der *Operation Supporting Means*, in diesem Fall das UAV mit seinen Automationsfunktionen, um dessen Fähigkeiten zielgerichtet einsetzen zu können.

Die folgenden Abschnitte beschreiben anfangs ein übliches Entwicklungsumfeld für die Entwicklung von ACUs bzw. intelligenten Agenten im Allgemeinen (vgl. Abschnitt 2.5.1) und eine sich daraus ergebende Problemstellung (vgl. Abschnitt 2.5.2).

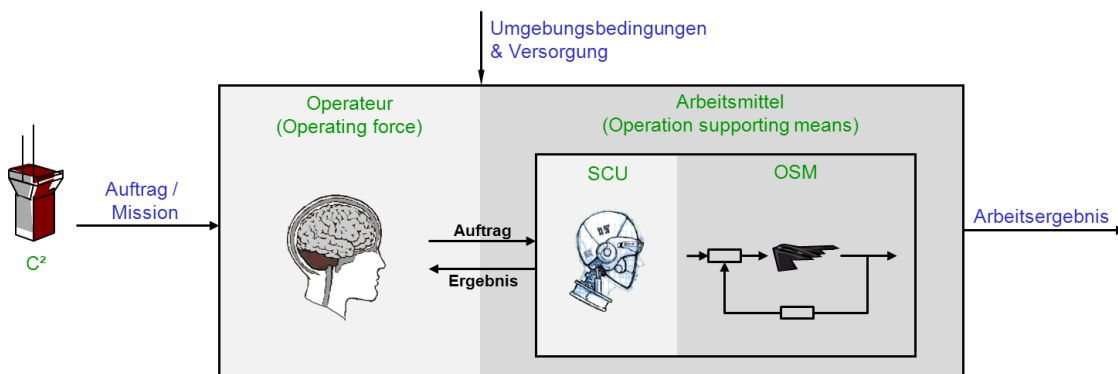


Abbildung 2-20. Arbeitssystem zur UAV-Führung mit einem semi-autonomen UAV System als Teil der Arbeitsmittel

2.5.1 Entwicklungsumfeld für einen intelligenten Agenten

Die Entwicklung und Implementation einer ACU (für die Domäne der UAV-Flugführung) erfolgt meist im Umfeld einer geeigneten Simulationsumgebung, in der das Vehikel mit seiner Automation und Sensorik sowie die missionsrelevanten Objekte abgebildet werden. Dabei ist wichtig, einen angemessenen Echtheitsgrad der Simulation zu wählen, die den Aufbau eines Szenarios ermöglicht, in der der intelligente Agent im

Hinblick auf dessen Interaktionen und Entscheidungsfindungen entwickelt werden kann. Der Schwerpunkt einer solchen Missionssimulation liegt in der Regel auf den Missionselementen, um die hochautomatisierte Auftrags- bzw. Missionserfüllung, oder auch die Interaktion mit einem menschlichen Operateur (engl.: Human in the loop) zu untersuchen. Dabei werden in der Regel Systemeffekte – wie beispielsweise die elektromagnetische Verträglichkeit von Komponenten, welche von realer Hardware verursacht werden – vernachlässigt, da diese rein technische Probleme beschreiben. Diese sind zwar für den Einsatz einer kognitiven Einheit in einem realen Umfeld zu berücksichtigen, jedoch anfangs für die eigentliche Entwicklung der kognitiven Funktionen irrelevant.

Abbildung 2-21 zeigt ein Diagramm, in welchem einige der aktuellen und abgeschlossenen Forschungsprojekte der Professur für Flugmechanik und Flugführung, der Universität der Bundeswehr München, nach den Gesichtspunkten des Echtheitsgrades der physikalischen Modelle und der Missionselemente kategorisiert sind (Unmanned Cognitive System for the Flight Domain – COSY^{flight}: [Putzer, 2004] [Frey, 2005], Unmanned Cognitive System – Teaming – COSY^{team}: [Meitinger, 2008], Manned Unmanned Teaming – MUM-T: [Rauschert et al., 2008] [Uhrmann et al., 2009], Using Cognitive Automation for Aircraft General Systems Management – CAAGSM [Pecher et al., 2010], Knowledge Configured Vehicle – KCV: [Kriegel et al., 2011] und die vorliegende Arbeit). Diese Darstellung erlaubt die Bewertung der Simulations- und oder Demonstrationsumgebungen, in Bezug auf das Maß der simulierten bzw. tatsächlichen Realität. Jedoch kann daraus keine Aussage über die eigentliche Komplexität und die Fähigkeiten der in den Projekten entwickelten und demonstrierten Kognitiven Einheiten abgeleitet werden.

Die Abszissenachse des Diagramms bezeichnet den Echtheitsgrad als Maß für die Wiedergabetreue und den Realismusgrad des Verhaltens der Missionselemente mit folgenden Einteilungen:

- **Statisch:** Die Elemente des Szenarios sind während der Laufzeit statisch (keine Aktionen oder Bewegungen) und nehmen nur durch deren Anwesenheit Einfluss auf die Entscheidungsfindung des intelligenten Agenten, wie etwa Bauwerke oder Geländebegebenheiten.
- **Zeitveränderlich:** Die Elemente des Szenarios können sich zur Laufzeit bewegen und Aktionen ausführen. Diese Handlungen sind jedoch zur Entwicklungszeit schon bekannt und werden entweder nach einer vorher festgelegten Zeitdauer oder einem definierten Ereignis initiiert. Beispiele hierfür sind andere Fluggeräte oder Fahrzeuge, die selbst nicht auf Umwelteinflüsse reagieren.

- **Stochastisch:** Die Elemente des Szenarios reagieren stochastisch auf das Verhalten des intelligenten Agenten. Dadurch kann mit einfachen Mitteln eine nicht deterministische, unvorhersehbare Handlungsweise von Szenarienelementen realisiert werden.
- **Intelligent:** Die Elemente des Szenarios besitzen selbst eine Art von „Intelligenz“ und versuchen eigene Ziele zu verfolgen. Deren Verhalten wird durch das Verhalten des zu entwickelnden intelligenten Agenten beeinflusst. Hierunter sind Bestandteile eines Szenarios zu verstehen, die in Wechselwirkung mit dem intelligenten Agenten treten und gegebenenfalls versuchen dessen Missionserfüllung zu verhindern (z.B. Feindliche Kräfte – Computer Generated Forces – CGF). Werden Versuche mit einem menschlichen Operateur durchgeführt, so kann dieser ebenfalls als ein „intelligentes“ Element des Szenarios angesehen werden.
- **Real:** Die Elemente des Szenarios entsprechen tatsächlichen Entitäten der Umwelt. Voraussetzung hierfür ist, dass die Entwicklung bzw. der Test des intelligenten Agenten ebenfalls in einer realen Umwelt stattfindet.

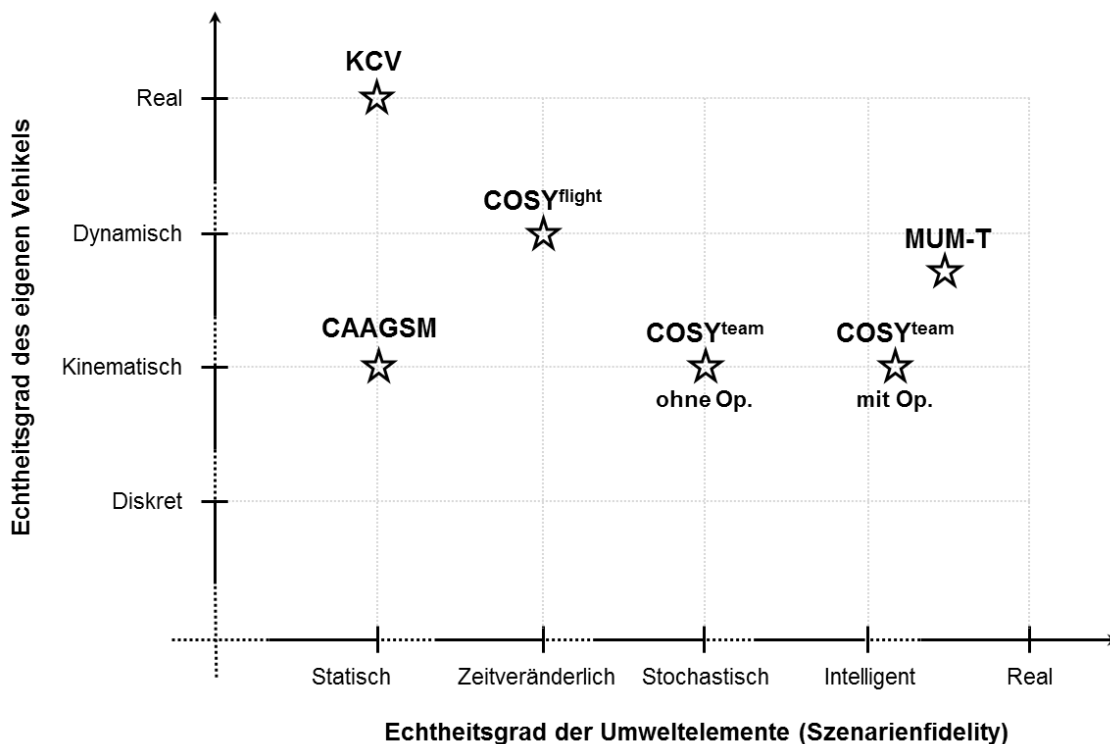


Abbildung 2-21. Kategorisierung aktueller und abgeschlossener Forschungsprojekte der UniBwM im Hinblick auf die Echtheitsgrade der Hardwaresysteme und der Missionselemente

Die Ordinatenachse beschreibt im Gegenzug den Echtheitsgrad der physikalischen Modelle des eigenen Vehikels mit folgenden Einteilungen:

- **Diskret:** Der Zustand von Systemkomponenten oder die Daten von simulierten Sensoren sind diskret und vordefiniert. So könnten beispielsweise die Geschwindigkeits- und Positionsdaten eines simulierten Fluggerätes zwar von extern gesetzt werden, es findet aber keine explizite Berechnung der Werte statt.
- **Kinematisch:** Verwendung einfacher Berechnungsmodelle für physikalische Komponenten, wie beispielsweise ein Fluggerät (z.B. Massenpunktmodell). Dadurch lassen sich einfache Bewegungen von Objekten simulieren, ohne dafür aufwändige Regelalgorithmen entwickeln zu müssen.
- **Dynamisch:** Anwendung von physikalischen Modellen, die das dynamische Verhalten oder die erzeugten Daten von Komponenten realitätsnah wiedergeben. Beispiele hierfür sind die Simulation von Sensordaten, die zusätzlich mit Störfaktoren (z.B. Rauschen, Bias, Einbaufehler) belegt werden oder die Flugdynamiksimulation eines Fluggerätes (z.B. Simulation aller Freiheitsgrade – 6-DOF).
- **Real:** Einsatz von realer Hardware (z.B. Sensorik, Fluggerät), als Entwicklungs- bzw. Demonstrationsumgebung eines intelligenten Agenten.

Sowohl bei steigendem Echtheitsgrad der physikalischen Modelle des eigenen Luftfahrzeugs als auch bei steigendem Echtheitsgrad der Szenarienelemente ist ein Anstieg der Gesamtsystemkomplexität zu verzeichnen. So kann man beispielsweise bei einem simulierten Fluggerät von idealen Sensoren ausgehen, die frei von Einbau- (z.B. Sensormisalignment) und Messfehlern (z.B. Rauschen, Bias, GPS-Loss) sind und keine wechselseitige Beeinflussung durch ungewollte elektrische bzw. elektromagnetische Effekte (elektromagnetische Verträglichkeit – EMV) mit anderen Avionikkomponenten aufweisen. Eine künstliche Verschlechterung der Sensordaten durch Aufgeben von synthetischen Fehlern ist zwar möglich, dennoch ist ein simuliertes Flugmodell in der Regel frei von Vibrationen oder anderen technischen Problemen jeglicher Art. Zudem haben Automationsfunktionen die Möglichkeit, direkt über Softwareschnittstellen auf Sensordaten zuzugreifen. Auf diese Weise können Probleme und Seiteneffekte, die durch reale Hardware-Schnittstellen, die nötige Dekodierung und Konvertierung der Daten eines realen Sensors mit den entsprechenden Tot- und Berechnungszeiten und die nachfolgende Verteilung der Daten (Interprozesskommunikation) vermieden werden. Auch wird die Darstellung von Ressourcen vornehmlich in der Simulation dahingehend simplifiziert, dass entweder keine Betriebsmittel verbraucht werden oder nur für manche einfache Modelle (z.B. keine Modellierung des schwappenden Kraftstoffs innerhalb eines Tanks) vorhanden sind.

Im Vergleich dazu benötigt ein reales Fluggerät eine Vielzahl von Ressourcen, wie Kraftstoffe und andere Betriebsmittel – z.B. die elektrische Versorgung von Avionikkomponenten (z.B. Bordcomputer, Sensoren, Triebwerk) und Aktuatoren innerhalb der

gerätspezifischen Grenzen. Zudem müssen für einen realen Flugversuch entsprechende Einrichtungen geschaffen und Vorkehrungen getroffen werden, um einen sicheren Flugbetrieb zu gewährleisten, die bei einer Simulation in der Regel irrelevant sind. Manche dieser Vereinfachungen können durch Integration der später auf dem realen Fluggerät genutzten Hardware in die Simulationsumgebung (engl.: Hardware in the loop – HIL) kompensiert werden, jedoch wird die Gesamtkomplexität eines realen fliegenden Systems immer deutlich höher sein als das simulierte Abbild.

2.5.2 Problemidentifikation

Für den Übergang bei der Entwicklung von kognitiven UAV-Führungsfunktionen von einer Simulationsumgebung – in der alle Sensoren, die Automation mit ihren Schnittstellen und das Fluggerät normalerweise störungsfrei funktionieren – auf ein real fliegendes System, sind die im vorigen Abschnitt genannten Probleme und die erhöhte Komplexität zu berücksichtigen und bei der Wissensentwicklung zu adressieren. Insbesondere die Einhaltung der Betriebsgrenzen, die Überwachung der Flugzustandssensorik und Ressourcen sowie der aktuelle Zustand der Automation fallen in den Aufgabenbereich des intelligenten Agenten, um einen sicheren, automatisierten Betrieb zu gewährleisten. Zudem benötigt dieser ein Verständnis über die Eigenschaften der Avionikkomponenten bis hin zu deren Verschaltung, um den Zustand und die Einsatzbereitschaft des Gesamtsystems bewerten zu können. Diese Art von Wissen ist spezifisch für *ein* bestimmtes Fluggerät und kann in der Regel nicht auf eine andere Maschine übertragen werden.

Für die Durchführung einer gegebenen Aufgabe oder Mission benötigt der intelligente Agent zudem Wissen und Verständnis, wie diese umgesetzt und erfüllt werden kann. Kombiniert man dieses missionsspezifische Wissen (MisW) mit dem maschinenspezifischen Wissen (MasW), so versetzt man den intelligenten Agenten in die Lage, einen gegebenen Auftrag hochautomatisiert zu erfüllen. Während der Durchführung des Auftrags oder der Mission steuert die ACU die Subsysteme des Fluggerätes und nutzt damit die Fähigkeiten der vorhandenen Automation des Fluggerätes und ausserdem die Möglichkeiten der Nutzlastkomponenten unter Verwendung des Wissens über deren Funktionen und Merkmale.

Besteht zum Beispiel eine Mission darin, an einem bestimmten Ort ein Luftbild aufzunehmen, so könnte ein, von einem Missionsmanagementsystem (technisches System, dessen Aufgabe es ist, die gegebene Aufgabenstellung zu bearbeiten und zu erfüllen) erzeugter Handlungsablauf wie folgt aussehen:

1. Einen Wegpunkt an die gegebenen Ortskoordinaten legen und das FMS entsprechend konfigurieren, so dass dieser Wegpunkt angefliegen wird.
2. Nach Erreichen der Zielkoordinaten eine Kamera auslösen, damit die gewünschte Aufnahme erstellt wird.

Stehen für diese einfache Mission zwei verschiedene Fluggeräte zu Verfügung, welche die benötigten Autopiloten- und Flugmanagementfunktionen besitzen, so wird es dennoch nicht möglich sein, dasselbe MMS in beiden Fluggeräten einzusetzen. Die

Verwendung der Funktion „*Fliege zu Wegpunkt*“, welche von der Automation bereitgestellt wird, bedarf aufgrund der unterschiedlichen Systeme der UAVs auch eine unterschiedliche Bedienung und Konfiguration. Auf Missionsebene ist wiederum das resultierende Verhalten des Fluggerätes, nämlich das Anfliegen eines bestimmten Wegpunkts, unter Vernachlässigung der Flugleistungen so gut wie identisch.

Diese *ausgeprägte Korrelation und Abhängigkeit* von *missions- und fluggerät-spezifischem Wissen* mit einem sehr hohen Verflechtungsgrad führt zu einer schlechten Wiederverwendbarkeit der Wissenspakete für nachfolgende Projekte, obgleich die intelligenten Missionsmanagementfunktionen in der Regel in erheblichem Maße unabhängig von der der Maschine zugrunde liegenden Umsetzung sind.

Ein ähnlicher Umstand wurde bereits 1998 von Muscettola im Rahmen der Entwicklung des *Remote Agents* für das Projekt *Deep Space One*, beschrieben:

The most effective way to reduce software development cost is to make the software “plug and play” and to amortize the cost of the software across successive applications. [Muscettola et al., 1998]

Gängige Praxis in der Softwareentwicklung ist es, die einzelnen Aufgabengebiete der Software zu modularisieren, damit diese gegebenenfalls für spätere Projekte wiederverwendet werden können. Nach Muscettola gestaltet sich dies schwierig:

This is difficult to achieve for the breadth of tasks that constitute an autonomous system architecture since each task requires the programmer to reason through system-wide interactions to implement the appropriate function. [Muscettola et al., 1998]

Der hohe Verflechtungsgrad der Wissenspakete resultiert aus der Notwendigkeit, dass der Programmierer für jede Aufgabe die systemweiten Wechselwirkungen einer entsprechenden Funktion zu berücksichtigen hat und damit der gewünschten Modularisierung widerspricht:

Hence this software lacks modularity, and has a use that is very restricted to the particulars of the hardware. The one of a kind nature of NASA’s explorers means that the cost of reasoning through system-wide interactions cannot be amortized, and must be paid over again for each new explorer. [Muscettola et al., 1998].

Als Folge ist es nachträglich kaum oder nur mit erheblichem Aufwand möglich, dieses Wissen voneinander zu trennen und wiederzuverwenden, was jedoch in vielerlei Hinsicht wünschenswert ist. Neben den von Muscettola schon erwähnten, ökonomischen Gründen könnten generische Missionsmanagementfunktionen (MMF) entwickelt, bedarfsabhängig auf verschiedenen UAV Plattformen eingesetzt und somit die Einsatzflexibilität deutlich gesteigert werden. Zudem würde die sich daraus ableitbare Kommunalität im Hinblick auf eine einheitliche Interaktion des Bedieners mit dem MMS des UAVs zu einer Senkung des Ausbildungsaufwands führen und ermöglichen, dass ein ausgebildeter UAV-Operator in der Lage wäre, verschiedene UAVs auf Basis unterschiedlicher Konfigurationen und Plattformen zu handhaben.

3 Knowledge configured vehicle – KCV

Bei der Führung von unbemannten Fluggeräten lag bisher die Verantwortlichkeit und der für die Verfolgung des Arbeitsziels benötigte Sachverstand bei dem/den menschlichen Bediener/n. Zwar wird die Operating Force eines solchen Arbeitssystems durch entsprechende Automationsfunktionen (Operation Supporting Means) seitens des UAVs bei der Erfüllung der Aufgabe oder Mission unterstützt, jedoch nur Teilaufgaben, wie beispielsweise die Stabilisierung, Bahnführung und Navigation des Fluggerätes durch ein FCS und nicht gesamtheitlich im Sinne der Missionserfüllung. Durch den Zugang zu fortschreitenden Technologien aus dem Umfeld der Künstlichen Intelligenz stehen heutzutage Mittel und Möglichkeiten zur Verfügung, die eine Entwicklung von Missionsmanagementfunktionen mit höheren Kompetenzen ermöglichen, welche zuvor nur dem Menschen vorbehalten waren.

Um der in Abschnitt 2.5.2 beschriebenen Problemstellung zu begegnen, wird in diesem Kapitel ein Konzept vorgestellt, das eine Wiederverwendung und flexible Austauschbarkeit von Wissens-elementen erlaubt und somit eine Möglichkeit darstellt, nicht nur die Entwicklungskosten zu senken sondern auch eine flexible Kombination von missionsspezifischem und fluggerätspezifischem Wissen zu ermöglichen. Das Konzept soll im Rahmen einer ACU, auf Grundlage der kognitiven Systemarchitektur COSA, umgesetzt, in realen Flugversuchen demonstriert und nachfolgend evaluiert werden.

Die Kernfrage dieser Arbeit lautet daher:

Wie kann eine Trennung von applikationsspezifischem (Mission) und maschinenspezifischem (UAV) Wissen mit einem hohen Wiederverwendungsgrad in der Domäne der UAV-Flugführung (für eine ACU) realisiert werden?

Abschnitt 3.1 gibt anfangs einen Überblick über drei verschiedene Analogien, auf deren Basis das Konzept (Abschnitt 3.2) eines wissensbasiert-konfigurierten Fluggerätes (engl.: *Knowledge Configured Vehicle – KCV*) entwickelt wurde. Der nachfolgende Abschnitt 3.3 erläutert die Interaktion und den organisierten Austausch von situativem Wissen zwischen den beiden Wissensarten und Abschnitt 3.4 beschreibt die theoretische Umsetzung des KCV-Konzepts mit dem Paradigma des Kognitiven Prozesses.

3.1 Analogien für die Entwicklung des Konzepts

Die semi-autonome Missionserfüllung eines unbemannten Fluggerätes soll durch den Einsatz eines intelligenten Agenten ermöglicht und die Führung eines solchen Fluggerätes für den menschlichen UAV-Operateur vereinfacht werden. Zwar kann ein technisches System einen Menschen weder ersetzen noch die Gesamtheit seiner Fähigkeiten abbilden, dennoch ermöglicht der Einsatz einer ACU die Leistungsfähigkeit des gesamten Arbeitssystems (vgl. Abbildung 2-20) zu steigern.

Ein solch intelligenter Agent benötigt für die Unterstützung des Menschen als auch für die hochautomatisierte Durchführung einer UAV-Mission zwei verschiedene Arten von Wissen – zum einen über die Mission und zum anderen über das Fluggerät. Um eine flexible Kombinierbarkeit dieser beiden Arten von Wissen zu erlauben, soll in den folgenden Abschnitten eine Beschreibung verschiedener Ansätze und Paradigmen für eine Konzeptentwicklung herangezogen werden, die eine mögliche Realisierung der Zielsetzung aufzeigen.

3.1.1 Control Configured Vehicle – CCV

Anfang der 70er Jahre entwickelte die Firma MBB (Messerschmitt-Bölkow-Blohm) ein Experimentalflugzeug auf Basis einer Lockheed F104-G mit dem Ziel der Verbesserung der Flugleistungen und Flugeigenschaften durch Reduktion der statischen Stabilität [Brockhaus, 2001]. Die Flugeigenschaften sollten dabei von einem Flugregler bestimmt werden, was den Terminus des *Control Configured Vehicle (CCV)* prägte:

Control Configured Vehicle: Bezeichnung für ein Flugzeug, dessen Flugverhalten komplett von einem Flugregler abhängig ist. [Klußmann & Malik, 2007]

Ziele des Experimentalprogramms nach [Brockhaus, 2001] waren:

- Nachweis der künstlichen Stabilisierbarkeit eines hochgradig instabilen Flugzeugs.
- Entwicklung eines fortschrittlichen digitalen Flugregelungssystems für Kampfflugzeuge auf der Grundlage des aktuellen Technologiestands.
- Nachweis der zuverlässigen Funktion eines CCV-Systems im Unter- und Überschallbereich.

In der letzten Phase des Projekts wurde der Erprobungsträger mit einem Entenflügel hinter dem Cockpit auf dem Flugzeugrumpf versehen (vgl. Abbildung 3-1), um eine Vorverlagerung des Neutralpunkts zu bewirken und ein instabiles Verhalten zu erzielen [Brockhaus, 2001]. Der Erstflug dieses Trägers wurde von dem Piloten Nils Meister am 20.11.1980 in Manching erfolgreich durchgeführt und in weiteren Testflügen bis 1985 der Nachweis für die Stabilisierbarkeit instabiler Flugzeuge, sowohl für den Unter- als auch den Überschallbereich, erbracht.

Die Erkenntnisse aus dem Projekt legten die Basis für die moderne, instabile Auslegung von Kampfflugzeugen als auch den Einsatz von sogenannten *Fly-by-wire* – Systemen (mittlerweile Stand der Technik in Verkehrs- und Militärflugzeugen). Der Einsatz der CCV-Technologie und die Verfügbarkeit von zuverlässiger Digitaltechnik bezeichnete zudem den Ausgangspunkt für das Design und die Entwicklung des Flugregelungssystem des Eurofighters [Bommer, 2000].

Erst durch den extensiven Einsatz von Regelalgorithmen ist der Mensch in der Lage instabile Fluggeräte, wie beispielsweise fortschrittliche, militärische Düsenflugzeuge zu steuern und zu bedienen, deren Flugeigenschaften und Fähigkeiten ausschließlich durch den Einsatz von Software ermöglicht und definiert werden. Bei allen modernen Kampf- und Verkehrsflugzeugen kommt diese Technologie zum Einsatz und entlastet den

menschlichen Piloten durch Stabilisierungs-, Autopiloten- und Flugmanagement-funktionalitäten von der Aufgabe der manuellen Steuerung, damit dieser sich vornehmlich um die eigentliche Missionserfüllung kümmern kann.



Abbildung 3-1. CCV-F 104-G Erprobungsträger, Bild: Peter Mühlböck collection

Analog zu einem CCV, dessen Flugverhalten maßgeblich von der Automation bestimmt wird, sollen bei KCV die missionsrelevanten Fähigkeiten sowie das Management der Maschine und ihrer Automation durch hinterlegtes Wissen definiert werden.

3.1.2 Integrierte Modulare Avionik / Betriebssysteme

Der Aufbau von konventionellen Avioniksystemen ist weitestgehend durch separate Teilsysteme gekennzeichnet, die bestimmte Funktionen bzw. Funktionsgruppen übernehmen, welche für eine anderweitige Nutzung nicht geeignet sind. Über die Zeit wurden diese Komponenten zu immer komplexeren Gesamtsystemen verknüpft, was einerseits zu einem sehr hohen Aufwand bei Tests und der Integration von solchen Systemen führte. Andererseits sind dadurch hohe Entwicklungskosten, eine schwierige Beschaffung und Nutzung sowie insbesondere ein hoher Zertifizierungsaufwand solcher Avioniksysteme bei Systemupgrades die Folge [Balsler et al., 2003].

Der Begriff der *Integrierten Modularen Avionik* (engl.: Integrated modular avionics – IMA) beschreibt ein neuartiges Konzept (erste Veröffentlichungen zu diesem Thema seit Ende der 80er / Anfang der 90er Jahre, vgl. [Balsler et al., 2003]) zum Aufbau von Avioniksystemen für zeitgemäße Kampf- und Verkehrsflugzeuge. Moderne IMA-Architekturen bieten eine hochintegrierte, partitionierte Umgebung mit gemeinsam nutzbaren, leistungsfähigen Datenverarbeitungseinheiten (engl.: Computing platform), die es erlaubt verschiedene Avionikfunktionen mit unterschiedlichen Prioritäten zu verarbeiten [Littlefield-Lawwill & Viswanathan, 2007]. Die Recheneinheiten werden in standardisierten Einbauvorrichtungen, sogenannten Racks, gruppiert und ermöglichen durch den Einsatz von entsprechenden Betriebssystemen, mit konformen Software-Schnittstellen (engl.: Application Programming Interface – API), eine hardware-unabhängige Verarbeitung von Applikationssoftware. Globale, gemeinsam genutzte Datenbusse, wie beispielsweise ARINC 659, ermöglichen den Zugriff über standardisierte Schnittstellen auf Sensoren, Aktuatoren oder andere Subsysteme.

[Littlefield-Lawwill & Viswanathan, 2007] beschreiben eine mögliche, theoretische Grundstruktur für einen offenen IMA Standard, welcher sich an einem bereits bestehendem Standard, dem Generic-Open-Architecture (GOA) Standard der Gesellschaft der Automobil Ingenieure (engl.: Society of Automotive Engineers – SAE) orientiert (Dokument AS4893, [Roark, 1996]). Auf Basis des GOA Standards schlagen Littlefield-Lawwill und Viswanathan eine mögliche Anwendung dieses Standards für den Bereich der Integrierten Modularen Avionik vor – die Generic-Open-Integrated-Modular-Avionics Architektur, deren Grundstruktur sich in vier Schichten gliedert, nämlich:

- Applikationsschicht
- Systemdienste-Schicht
- Ressourcenzugriffs-Schicht
- Physikalische Schicht

Diese Schichten nehmen eine Abstraktion der darunterliegenden Schichten vor, wie es beispielsweise auch bei dem OSI-Referenzmodell, als Grundlage für Kommunikationsprotokolle von Rechnernetzen oder auch bei Betriebssystemen umgesetzt wird. Dies führt zu einer klaren Abgrenzung der Schichten und verringert zudem die Systemkomplexität. Tanenbaum beschreibt diesen Sachverhalt für die Domäne der Betriebssysteme mit den folgenden Worten:

Abstraktion ist der Schlüssel, um Komplexität zu verwalten. Gute Abstraktionen verwandeln eine fast unmögliche Aufgabe in zwei handhabbare. [Tanenbaum, 2009]

So werden durch den Einsatz eines Betriebssystems die speziellen Eigenschaften der Hardware, wie etwa dem Prozessor, Speicher oder anderen Geräten (z.B. Festplatten, Disketten, USB-Laufwerke) abstrahiert, um der (Anwendungs-) Software und deren Entwicklern

{...} hübsche, saubere, elegante, konsistente Abstraktionen bereitzustellen. [Tanenbaum, 2009]

Für die Domäne der UAV-Flugführung mit dem Ziel einer flexiblen Austauschbarkeit von Wissenselementen erfordert die Umsetzung dieses Prinzips, wie von [Tanenbaum, 2009] für Betriebssysteme beschrieben, im ersten Schritt geeignete Abstraktionen zu definieren und diese nachfolgend in geeigneter Weise zu implementieren.

3.1.3 Wissen eines Piloten

Der Pilot eines bemannten Fluggerätes (z.B. eines Passagierflugzeuges) muss über zwei unterschiedliche Arten von Wissen verfügen. Er benötigt Grundkenntnisse über die Grundlagen des Fliegens, wie ein Flugzeug zu steuern ist, gängige Manöver (Start, Landung, Steigflug, Kurvenflug) durchgeführt und welche Steuerflächen dafür verwendet werden. Weiterhin benötigt der Pilot Sachverstand über die Arten von Lufträumen, Navigation (bodenreferenzierter Sichtflug, Funknavigation, Satellitenavigation), das Führen des Fluggerätes nach unterschiedlichen Flugregeln

(Sichtflugregeln – visual flight rules - VFR, Instrumentenflugregeln - instrument flight rules – IFR) sowie Wetterkunde (Meteorologie). Diese eine Art von Wissen ist unabhängig eines Fluggerätyps und kann demnach der Domäne „*Luftfahrt*“ zugeordnet werden (*allgemeines Domänenwissen*).

So verfügt ein ausgebildeter Pilot für einen bestimmten Flugzeugtyp (z.B. Boeing 737) zwar über solch allgemeines Domänenwissen, wäre aber nicht in der Lage das Muster eines anderen Herstellers (z.B. Airbus) zu fliegen. Demnach benötigt der Pilot auch einen fundierten Kenntnisstand, wie ein bestimmter Flugzeugtyp zu bedienen ist. In einer speziellen Ausbildung, der sogenannten Musterberechtigung (engl.: Type rating), wird ihm dabei das Wissen, über das Gesamtsystem vermittelt. Diese beinhaltet eine Flugzeugbeschreibung mit seinen Systemen, deren Bedienung (engl.: Aircraft and system description and operation), die Normal- und Notfallverfahren (engl.: Normal & emergency procedures) sowie die Aufteilung der entsprechenden Aufgaben der Crewmitglieder (engl.: Crew duties). Ein weiterer wichtiger Punkt in der Ausbildung ist die Vermittlung der Betriebsgrenzen (engl.: Operating limitations) und Flugeigenschaften (engl.: Flight characteristics). Ein Entwicklungsingenieur verfügt möglicherweise über diese Kompetenz, ist aber dennoch nicht in der Lage das Flugzeug zu führen, da ihm das *allgemeine Domänenwissen* der Flugführung fehlt. Dieses maschinenspezifische Wissen ist der *spezielle* Teil des *Domänenwissens*, welches für die Bedienung der Maschine benötigt wird.

Erst die *Kombination* von *allgemeinem Domänen-* und *maschinenspezifischem Wissen* ermöglicht, in Verbindung mit Anwendungswissen (Wissen über die Möglichkeiten der Durchführung einer Aufgabe), die Durchführung einer Mission bzw. die Erfüllung einer gestellten Aufgabe.

Eine der Stärken des Menschen ist seine Generalisierungsfähigkeit. Verfügt ein Pilot bereits über *allgemeines Domänenwissen* aus dem Bereich der Flugführung, so ist er in der Lage dieses Wissen auch bei der Ausbildung auf einen anderen Flugzeugtyp anzuwenden, ohne es sich neu aneignen zu müssen. Eine ähnliche Flexibilität soll durch das im folgenden Abschnitt beschriebenen Konzept realisiert und ermöglicht werden.

3.2 Konzept

Um eine Trennung und eine Wiederverwendbarkeit von missionsspezifischem und fluggerätspezifischem Wissen für die Domäne der semi-autonomen UAV Flugführung zu ermöglichen, wird das Konzept des wissensbasiert-konfigurierten Fluggerätes (engl.: *Knowledge configured vehicle – KCV*) vorgestellt. Korrespondierend mit dem Paradigma des *Control Configured Vehicle* (vgl. Abschnitt 3.1.1), dessen Flugverhalten und Flugeigenschaften ausschließlich durch das Flugregelungssystem bestimmt werden, sollen die *Fähigkeiten* und *Missionseigenschaften* eines *Knowledge Configured Vehicle* durch hinterlegtes *Wissen* definiert und konfiguriert werden.

Analog zu den Wissensarten eines menschlichen Piloten (vgl. Abschnitt 3.1.3) entspricht bei einem *KCV* das Anwendungswissen dem missionsspezifischen Wissen und das Type-Rating- und Flugführungs-Wissen dem Domänenwissen. Um eine

Verbindung zwischen diesen beiden unterschiedlichen Ebenen zu schaffen, nämlich der Missionsebene (z.B. Auftrag, Missionselemente) und der flugzeugtechnischen Ebene (z.B. Autopilot, Flugmanagementsystem), wird zum einen, seitens des maschinenspezifischen Wissens, das Instrument der Abstraktion eingesetzt (vgl. Abschnitt 3.1.2). Zum anderen wird durch die Einführung einer Zwischenschicht (im folgenden KCV-Schicht genannt) eine Schnittstelle geschaffen, die eine Austauschbarkeit und flexible Kombinierbarkeit der beiden Wissensarten in Form von Paketen erlaubt (vgl. Abbildung 3-2). Somit könnten für einen bestimmten, spezifischen Auftrag sowohl das bestgeeignete Missionswissen als auch Fluggerät und dessen entsprechendes Wissenspaket ausgewählt und zusammengefügt werden.

Die KCV-Schicht stellt dabei eine Art symbolische Wissensschnittstelle dar, deren einzige Einschränkung die Forderung einer gemeinsamen Ontologie ist und sich auf diese Weise erheblich von statischen, datentypenbezogenen Schnittstellen von prozeduralen bzw. objektorientierten Programmiersprachen unterscheidet. Diese Wissensschnittstelle gibt eine *Organisationsstruktur* vor, in der situatives Wissen, erzeugt von den missions- und maschinenspezifischen Wissenspaketen, eingeordnet und verwaltet wird. Auf diese Weise lässt sich eine geordnete Kommunikation und Interaktion erreichen.

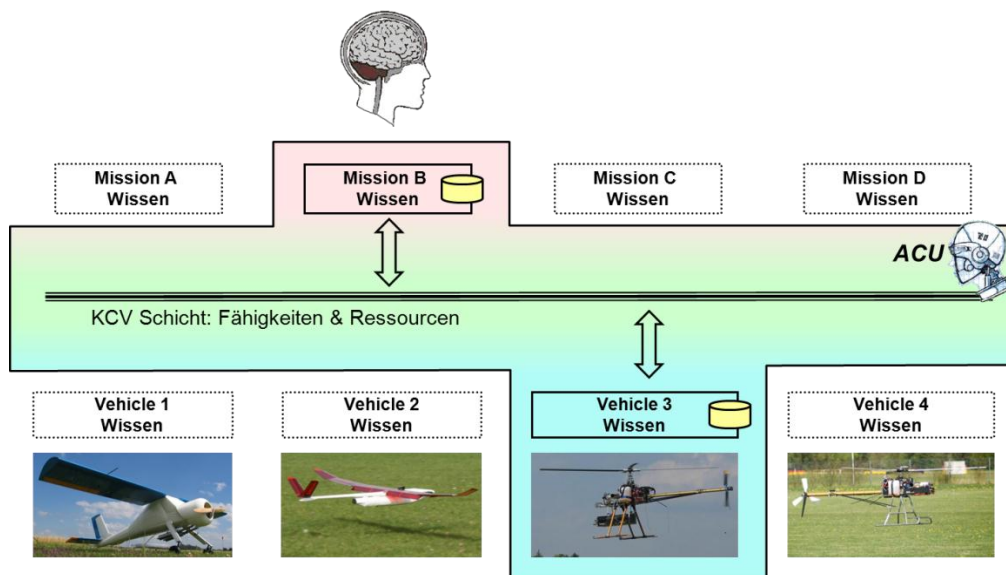


Abbildung 3-2. KCV Grobkonzept

Die Ausprägung von situativem Wissen, innerhalb einer ACU, ist von hohem Abstraktionsgrad und in der Regel ausschließlich symbolisch kodiert. Aus diesem Grund muss das maschinenspezifische Wissen die Aufgabe übernehmen, die Transformation der subsymbolisch repräsentierten Zustände und Parameter der Maschine sowie der vorhandenen Nutzlastkomponenten (z.B. Sensoren) durchzuführen. Eine Möglichkeit wäre, sämtliche Systemparameter der Maschine zu symbolisieren, wie zum Beispiel den Wert der aktuellen Treibstoffmenge von einer numerischen Repräsentation von 0-100 Prozent in symbolischen Werte wie *sehr wenig*, *wenig*, *mittel*, *viel*, *voll* abzubilden. Allein durch die hohe Anzahl an Parametern würden aktuell

verfügbare, kognitive Systeme (in Verbindung mit aktuell verfügbaren Rechnern) an ihre Leistungsgrenzen stoßen. Gesetzt den Fall, das kognitive System wäre in der Lage die Datenmengen zu verarbeiten, so könnte der nächste Schritt darin bestehen, Missionsmanagementfunktionen – auf Basis von missionsspezifischem Wissen – zu entwickeln, welche die Bedeutung und Verwendung der Parameter versteht und deren Überwachung und die Ansteuerung der Automation übernimmt. Diese Parameter sind jedoch wiederum fluggerätspezifisch und möglicherweise nur in einigen Bereichen mit den Parametern eines anderen Fluggerätes kongruent, was dazu führt, dass das Missionswissen für die andere Maschine angepasst werden müsste. Dies wiederum würde der Zielsetzung der flexiblen Austauschbarkeit und Wiederverwendbarkeit von generischem Missionswissen widersprechen.

Durch eine geeignete Abstraktion der parametrischen und subsymbolischen Repräsentation der hardwarespezifischen Parameter eines Fluggerätes soll die Entwicklung eines wiederverwendbaren, fluggerätspezifischen Wissenspakets realisiert werden. Ziel ist es die hohe Anzahl an Systemparametern zu verringern und eine symbolische Kommunikation auf *missionsrelevanter* Ebene zu ermöglichen.

In dem folgenden Abschnitt wird die Transition von den subsymbolisch repräsentierten Parametern des Fluggerätes auf die symbolische Ebene sowie deren Ausprägung in Form von situativem Wissen beschrieben. Darüber hinaus wird die Funktion der bereits benannten KCV-Schicht sowie deren Interaktion mit den maschinenspezifischen und missionsspezifischen Wissenspaketen erläutert.

3.3 Abstraktion und Interaktion der Wissenspakete nach dem KCV - Konzept

Für die Durchführung eines speziellen Auftrags bzw. einer Mission durch generische Missionsmanagementfunktionen (hier als Teil der kognitiven Einheit), benötigen diese Kenntnis über die Eigenschaften und die Möglichkeiten des Fluggerätes zur Entwicklung eines geeigneten Plans, um das Arbeitsziel zu erreichen. Weitere, missionsbeeinflussende Faktoren sind zum Beispiel der aktuelle Zustand der Avionik als auch die Menge an verfügbaren Ressourcen, wie Kraftstoffe, elektrische Energie, verbleibende Effektoren oder der noch zu Verfügung stehende Speicherplatz für das Aufnehmen der Daten von Aufklärungssensoren.

Abbildung 3-3 zeigt den schematischen Aufbau des Konzepts und die Wechselbeziehung zwischen den drei unterschiedlichen Ebenen, nämlich die Missionsebene (rot, oben), die Organisationsebene (grün, mittig) und die Maschinenebene (blau, unten) *innerhalb* der *Artificial Cognitive Unit*.

Im unteren Bereich von Abbildung 3-3 sind die technischen Systeme (Avionik, Automation, Missionsnutzlast) und der aktuelle Systemzustand und Ressourcen des UAVs dargestellt. Durch Verarbeitung des *MasW* werden, abhängig von Art und Ausprägung der verfügbaren, technischen Ausstattung der Maschine, entsprechende *Fähigkeiten* abstrahiert. Dieser Abstraktionsschritt wird in Abbildung 3-3 durch den Farbübergang von blau nach grün symbolisiert und entspricht dem in Abschnitt 3.1

beschriebenen Übergang von den Parameter-dominierten und subsymbolischen Informationen der Sensoren und der Automation auf ein symbolisches Level. Hierbei bezeichnen *Fähigkeiten* einerseits die Kombination von Automationsbetriebsarten, die ein bestimmtes *Verhalten* des Fluggerätes erzeugen, wie etwa das automatische Anfliegen eines Wegpunkts, das Abfliegen einer Wegpunktliste und das Fliegen mit einer vorgegebenen Fluggeschwindigkeit auf einer bestimmten Flughöhe. Zum anderen können *Fähigkeiten* abstrahiert werden, die im Zusammenhang mit Nutzlastkomponenten einhergehen. Ist beispielsweise eine Kamera Teil der Nutzlast, so können das Aufnehmen eines Luftbildes oder das Senden eines Echtzeit-Videostreams ebenfalls als *Fähigkeiten* der Maschine aufgeführt werden, insofern die dafür notwendigen, technischen Voraussetzungen erfüllt sind. Die Liste aus sich ergebenden *Fähigkeiten* wird in die KCV-Schicht (mittlere, grüne Schicht in Abbildung 3-3) eingetragen.

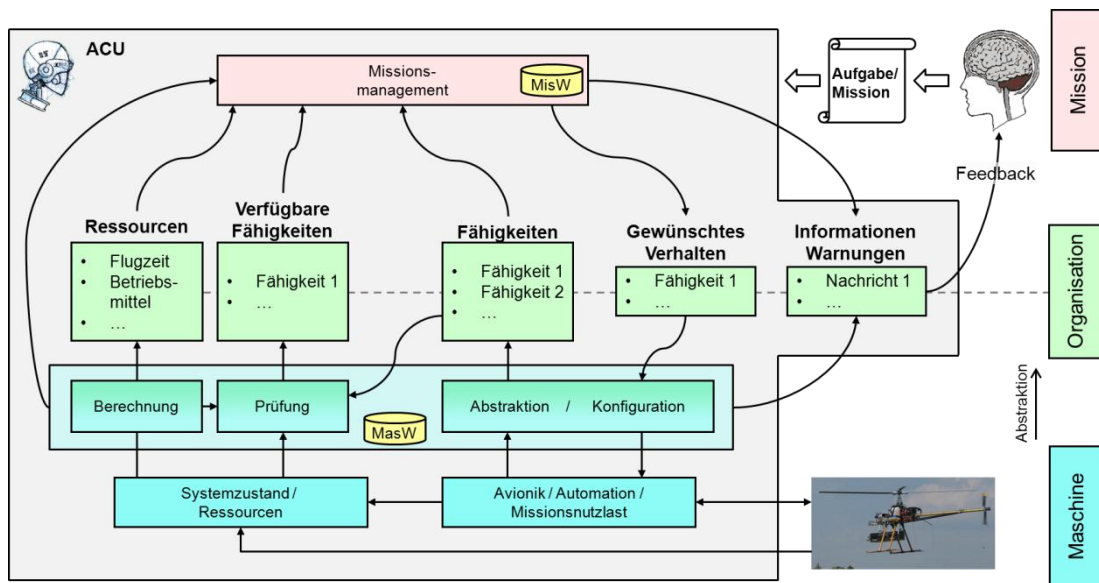


Abbildung 3-3. Interaktion zwischen missionsspezifischem und maschinenspezifischem Wissen über die KCV – Organisationsstruktur

Demnach haben die Konfiguration des UAVs (z.B. Flugzeug, Hubschrauber), die Fertigkeiten der Automation, die aktuellen Ressourcen sowie die vorhandene Nutzlast Einfluss auf die *Fähigkeiten* der Maschine. Der Einsatz dieser *Fähigkeiten* bedingt, ausgehend von dem Betrieb eines realen Fluggerätes, die Einhaltung der gerätspezifischen Betriebsparameter von Avionik- und Missionsnutzlastkomponenten und die Überwachung der Betriebsgrenzen der Automation durch den intelligenten Agenten. Hierfür wird im *MasW* zusätzliches a-priori Wissen hinterlegt, welches die Anforderungen und Vorbedingungen der jeweiligen Soft- bzw. Hardwarekomponenten definiert. Beispielsweise ist für den Gebrauch einer *Fähigkeit*, wie das Abfliegen von Wegpunkten, eine zuverlässige Lage- und Positionsbestimmung durch die Flugzustandssensorik (z.B. Abgeschlossene Initialisierung der IMU und ungestörter Empfang der Signale von genügend GPS Satelliten) und das Vorhandensein einer Wegpunktliste erforderlich. Durch Anwendung und Verarbeitung des *MasW* kann eine Prüfung des aktuellen Systemzustands durchgeführt werden, in welcher die System-

parameter analysiert und die Einsatzbereitschaft der entsprechenden *Fähigkeit* determiniert wird. Sind alle im maschinenspezifischen Wissen definierten Erfordernisse erfüllt, so wird diese als *verfügbare Fähigkeit* in der KCV-Organisationsstruktur gelistet.

In der oberen Schicht von Abbildung 3-3 ist der menschliche Operateur dargestellt, welcher einen Auftrag oder eine Mission definiert, die von den Missionsmanagementfunktionen – repräsentiert durch das missionspezifische Wissen der ACU – bearbeitet und durchgeführt werden soll. Für die Aufgabe der Planung können die MMF alle *Fähigkeiten* des zu führenden UAVs aus der KCV-Schicht abfragen und darauf aufbauend ein Vorgehen ausarbeiten, den angestrebten Zielzustand zu erreichen. Grundvoraussetzung hierfür ist neben einer kongruenten Syntax und Semantik eine gemeinsame Ontologie, die ein beiderseitiges Verständnis der Wissenspakete gewährleistet. Die durch das *MasW* abstrahierten *Fähigkeiten* des UAVs sollten durch signifikante Bezeichnungen beschrieben werden, die das *MisW* verstehen und interpretieren kann. Ferner erleichtern selbsterklärende Begriffe zum einen die Entwicklung des a-priori Wissens für einen Wissensingenieur und zum anderen könnte, im Hinblick auf eine Führung eines solchen Fluggerätes durch den Menschen (in Analogie zur *auftragsbasierten Führung* als *fähigkeitsbasierte Führung* benannt – engl.: *Ability Based Guidance – ABG*), die Interaktion verständlicher und intuitiver gestaltet werden. Es müssten dann keine komplexen Kombinationen von Automationsbetriebsarten mehr geschaltet werden, sondern das *gewünschte Verhalten* der Maschine könnte direkt angefordert werden.

Für die Durchführung einer missionsrelevanten Aktion können die MMF aus der Liste der *verfügbaren Fähigkeiten* wählen und die jeweils passende *Fähigkeit* als *gewünschtes Verhalten* anfordern. Der Begriff *Verhalten* bezeichnet dabei die von der Automation erzeugte Bewegung des Fluggerätes oder die durchgeführte Aktion der Missionsnutzlast als Ergebnis der genutzten *Fähigkeit*. Auch das simultane Anfordern mehrere *Fähigkeiten* ist möglich, insofern dies nicht konkurrierende Betriebsarten der Automation oder der Missionsnutzlast voraussetzt.

Das Abfragen der (*verfügbaren*) *Fähigkeiten* durch die MMF vor dem Anfordern eines *gewünschten Verhaltens* ist jedoch nicht obligatorisch und kann auch nicht notwendig sein, wenn etwa das Missionswissen über keine ausgeprägten Planungsfunktionalitäten verfügt. Damit ist eine Verwendung von vorgeplanten Aufgabenagendas möglich, die skriptartig abgearbeitet werden können, wenn die gegebene Mission eine geringe Komplexität aufweist. Wird eine *Fähigkeit* angefordert, die jedoch nicht *verfügbar* ist, so sollen sowohl dem Missionswissen und auch dem menschlichen Operateur hierfür die Gründe (z.B. aktueller Systemzustand entspricht nicht dem geforderten, nötige Umgebungsbedingungen nicht vorhanden) kommuniziert werden. Das Anfordern einer *verfügbaren Fähigkeit* führt jedoch nicht direkt zu deren Ausführung, sondern nach der ersten Prüfung des voraussetzenden Systemzustands (Verfügbarkeitsprüfung) erfolgt eine Abschätzung der benötigten Ressourcen für dieses *gewünschte Verhalten* (Ressourcenprüfung). So ist es beispielsweise denkbar, dass der aktuelle Systemzustand die *Fähigkeit* eine Wegpunktliste abzufliegen erlaubt (z.B. IMU und GPS Sensoren

liefern zuverlässige Werte), jedoch aufgrund der Länge des Flugwegs, die Menge der benötigten Ressourcen (z.B. Treibstoff) die Vorhandenen übersteigt. In diesem Fall werden sowohl das Missionswissen als auch der Operateur darüber in Kenntnis gesetzt. Sind nach Abschluss beider Prüfungen alle Vorbedingungen und Anforderungen erfüllt, so übernimmt das maschinenspezifische Wissen die Rücktransformation des symbolisch repräsentierten, *gewünschten Verhaltens* für die parametrische und numerische Konfiguration der Automation bzw. Nutzlastkomponenten.

Folglich ermöglicht das Konzept des *Knowledge Configured Vehicle* die Entwicklung von Missionsmanagementfunktionalitäten, welche sich durch den koordinierten Einsatz der vom Fluggerät angebotenen Fähigkeiten (engl.: Capability management) charakterisieren. Im anschließenden Abschnitt soll die Umsetzung des beschriebenen Konzepts des *Knowledge Configured Vehicle* mit dem Paradigma des Kognitiven Prozesses im Hinblick auf die Interaktion der Wissenspakete und die Verwendung der Nomenklatur des KP, beschrieben werden.

3.4 KCV Konzeptumsetzung mit dem Paradigma des Kognitiven Prozesses

Abbildung 3-4 zeigt den Ansatz der Umsetzung des Konzepts eines *Knowledge Configured Vehicle* mit dem Paradigma des Kognitiven Prozesses für eine künstliche kognitive Einheit. Das missionsspezifische und das maschinenspezifische Wissen werden durch Pakete repräsentiert, die in Anlehnung an Abbildung 3-3 übereinander angeordnet werden. Die dazwischen liegende KCV-Organisationsschicht stellt ebenfalls ein Wissenspaket dar, das zur besseren Übersicht nur als Balken dargestellt ist, da hier eigentlich keine direkte Verarbeitung, sondern eine reine Organisation von situativem Wissen stattfindet. An dieser Stelle sei darauf hingewiesen, dass die Darstellung zwar eine asynchrone Verarbeitung der beiden Arten von Wissen durch den Kognitiven Prozess vermuten lässt, jedoch wird zur Laufzeit des kognitiven Systems das gesamte Wissen *quasi-synchron* verarbeitet. Zudem werden die Eingangsdaten, sowohl für das MisW und das MasW, gleichsam über eine gemeinsame Eingangsschnittstelle in die ACU geladen und sind nur zur besseren Übersicht getrennt dargestellt.

Durch Verarbeitung des maschinenspezifischen a-priori Wissens (dargestellt im unteren Bereich von Abbildung 3-4) werden die Eingangsdaten, insbesondere Informationen über die vorhandene Automation, Nutzlast sowie deren Status und der Typ des Fluggerätes interpretiert (KP-Transformator: Identifikation) und daraus die entsprechenden *Fähigkeiten* der Maschine abstrahiert.

Diese *Fähigkeiten* werden folgend im situativen Wissen als Überzeugungen (engl.: beliefs) zentral repräsentiert und könnten theoretisch direkt von dem missionsspezifischen Wissen gelesen werden, was jedoch die Gefahr der unerwünschten Erzeugung einer Abhängigkeit zwischen dem MisW und dem MasW mit sich bringt. Aus diesem Grund werden die abstrahierten *Fähigkeiten* in die von der KCV-Schicht vorgegebene Organisationsstruktur eingetragen. Die Kennzeichnung, ob eine *Fähigkeit*

auch *verfügbar* ist, wird durch das MasW festgelegt, welches den aktuellen System- und Ressourcenzustand bewertet.

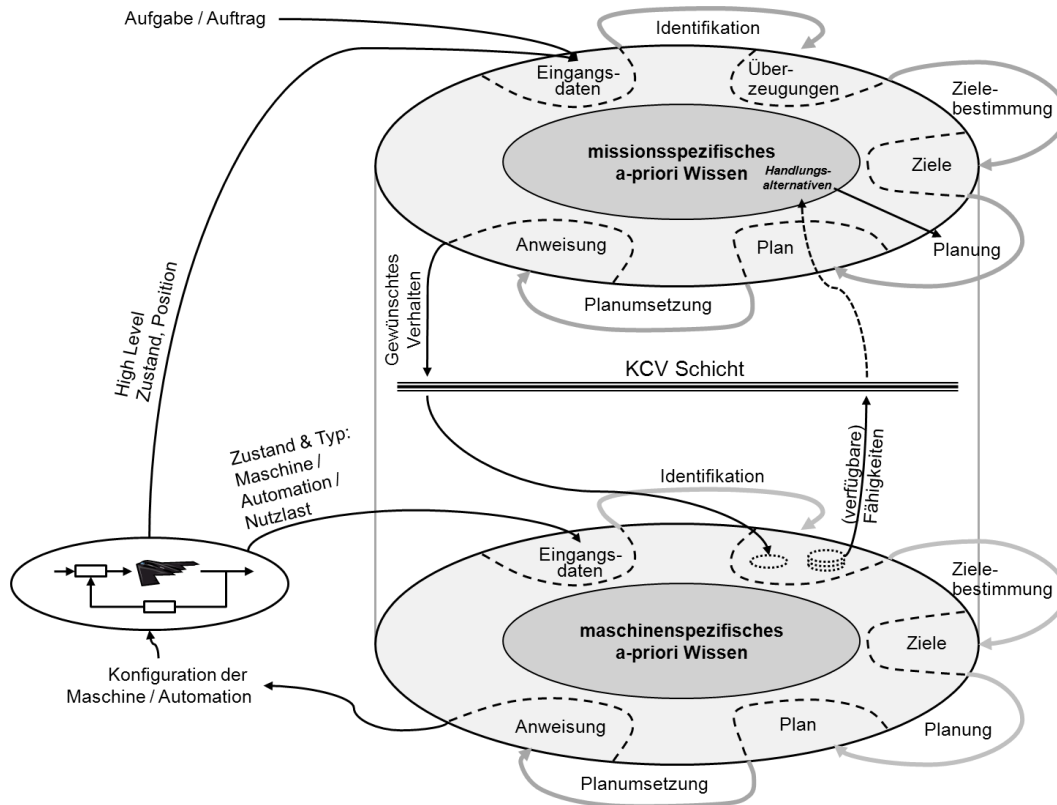


Abbildung 3-4. Aufbau des KCV-Konzeptes mit dem Paradigma des Kognitiven Prozesses

Die Missionsmanagementfunktionen verfügen über mehrere Ziele (Instanzen von Wünschen), wovon eines der Erfüllung des gegebenen Auftrags entspricht. Andere Ziele könnten etwa sicherheitsrelevante Interessen wahren, wie etwa die Vorbeugung einer Kollision mit anderen Fluggeräten oder bodengebundenen Hindernissen oder das Informieren des menschlichen Operators über bestimmte Ereignisse oder den aktuellen Erfüllungsgrad der Aufgabe. Für die Durchführung des Auftrags ist es die Aufgabe der Missionsmanagementfunktionen einen Vorgehensplan mit den möglichen Handlungsalternativen (engl.: *action alternatives*) zu entwickeln, um das Arbeitsziel zu erreichen. Hierbei entsprechen die Handlungsalternativen den *verfügbaren Fähigkeiten* des UAVs, welche durch das maschinenspezifische Wissen *dynamisch* abgeleitet werden – abhängig des aktuellen System- und Versorgungszustand der Maschine.

Besteht die aktuelle Aufgabe beispielsweise darin, ein Luftbild an einem vorgegebenen Ort aufzunehmen, so könnten die folgenden Handlungsalternativen, wie „*Mache Luftbild*“, „*Richte Sensor aus*“, „*Fliege Wegpunkt an*“, „*Fliege auf Höhe X*“, „*Sende Nachricht an Operateur*“ für die Erfüllung verwendet werden – vorausgesetzt, dass das UAV über die hierfür notwendige technische Ausrüstung (z.B. Automation, Kommunikationsequipment, Nutzlast) und die notwendigen Ressourcen verfügt.

Damit das Ziel der Aufgabenerfüllung erreicht werden kann, gilt es für das MisW, die aktuelle Situation zu bewerten und zu erfassen. Neben dem nötigen Verständnis für die

Art der gestellten Aufgabe, müssen die MMF auch in der Lage sein, die vorherrschende Situation zu verstehen (Aufbau eines Situationsbewusstseins), um die Unterschiede zwischen dem aktuellen Zustand und dem Zielzustand zu erkennen und zielführende Handlungen zu initiieren. Dafür gilt es, die dafür benötigten Ressourcen abzuschätzen und die Nutzung der *Fähigkeiten* des UAVs in eine sinnvolle Reihenfolge zu bringen. Dieser Vorgang der Erarbeitung eines geeigneten Vorgehens wird von dem kognitiven Schritt der Planung durchgeführt. Ist die Planung in Anlehnung an das aktuelle Arbeitsziel, der aktuellen Situation, sowie des aktuellen System- und Ressourcenzustands abgeschlossen, so kann mit der Umsetzung dieses Plans begonnen werden. Die entsprechende(n) Anweisung(en) wird/werden, passend zur vorherrschenden Situation, von dem missionsspezifischen Paket im situativen Wissen erzeugt und in der KCV-Schicht als *gewünschte(s) Verhalten* eingetragen.

Das maschinenspezifische Wissen erkennt ein neu eingetragenes *gewünschtes Verhalten* in der KCV-Schicht und interpretiert dieses als eine neue Überzeugung, deren Umsetzung es zu erfüllen gilt, wenn hierfür ein entsprechender Wunsch im MasW modelliert wurde. Zusätzlich muss geprüft werden, ob der aktuelle Systemzustand die Erzeugung dieses *Verhaltens* erlaubt und genügend Betriebsmittel zur Verfügung stehen. Bescheinigt die Prüfung der Ressourcen die Machbarkeit, so wird ein Plan erstellt, wie und gegebenenfalls auch in welcher Reihenfolge die Automation konfiguriert wird und welche Parameter es zu setzen gilt. Mittels Anweisungen werden nachfolgend die Konfiguration der Automation bzw. Komponenten des Fluggerätes vorgenommen und die hardwarespezifischen Kommandos und Parameter über die Ausgabeschnittstelle an die korrespondierenden Subsysteme verteilt.

Gesetzt den Fall, dass der Transformer *Planung* des Kognitiven Prozesses über sehr elaborierte Funktionalitäten verfügt, so könnten auch Lösungen für nicht vorhandene, aber dennoch für die Erfüllung einer Aufgabe benötigte *Fähigkeiten* gefunden werden. Verfügt die Automation eines UAVs etwa über kein Flugmanagementsystem, welches die Wegpunktnavigation des Fluggerätes übernimmt, sondern nur über einfachere Autopilotenfunktionen, wie das Fliegen in einer vorgegebenen Richtung, Flughöhe und Geschwindigkeit, so könnte durch einen kombinierten Einsatz der vorhandenen Handlungsalternativen (*verfügbare Fähigkeiten*) ein solches *Verhalten* erzeugt werden.

Der Vorgang des beschriebenen KP-Transformers *Planung* bezieht sich dabei jedoch auf vergleichsweise *symbolische* Planer, die Handlungsagendas erzeugen, welche durch den Einsatz und die Anwendung der verfügbaren oder auch aus einzelnen, kombinierten *Fähigkeiten* abgearbeitet werden können (vgl. [Strenzke & Schulte, 2011]). Bei militärischen Szenarien, ist es jedoch denkbar, dass das kognitive System für die Durchführung einer Mission, die Erzeugung eines optimalen Flugwegs, oder einer bedrohungsminimalen Route durch ein feindliches Gebiet benötigt. Solche *numerischen* Planungsfunktionalitäten erfordern sehr viel Rechenleistung und werden sinnvollerweise als separate Planer außerhalb einer kognitiven Einheit realisiert. Die Aufgabe der ACU besteht folgend darin, entsprechende Vorgaben und Randbedingungen für den Planer zu definieren, um als Ergebnis die gewünschte Flugroute zu erhalten. Neben diesen wichtigen Informationen benötigt der Planer zudem noch

detaillierte Angaben und Randbedingungen (engl.: Constraints) über die Flugleistungen des Fluggerätes. Nur dann kann eine optimale, für die Maschine fliegbare Trajektorie errechnet werden. Folgend dem Konzept des *Knowledge Configured Vehicle* sollen auch diese Daten und Informationen über die KCV-Schicht von dem maschinen-spezifischen Wissen erfragt werden, um die Möglichkeit zu schaffen, generische und fluggerätenabhängige Planer zu entwickeln und zu verwenden.

Im Rahmen der vorliegenden Arbeit wurde der Fokus bei der Entwicklung des a-priori Wissens der KCV-ACU auf das Management, die Überwachung der Maschine und dessen Automation gelegt und nur vergleichsweise einfaches Missions- und Planungswissen modelliert, um die Funktionalitäten und das Konzept eines *Knowledge Configured Vehicle* demonstrieren zu können. Angesichts der Tatsache, dass die möglichen Betriebszustände der Automation (vgl. Abschnitt 5.4.5.1) statisch und diskret sind und es bei der Bedienung und Aktivierung der entsprechenden Modi nur in manchen Fällen der Einhaltung einer bestimmten Reihenfolge (Bedingt durch die sogenannte *Moding-Logic* des FCS/FMS) bedarf, werden auch für das *MasW* nur geringe Planungsfähigkeiten benötigt.

Im folgenden Kapitel wird die Realisierung der einzelnen, benötigten Wissensmodelle beschrieben, die jeweils dem maschinen- und missions-spezifischen Wissen zugeordnet werden können. Die Modellierung erfolgt auf Grundlage der Kognitiven Systemarchitektur COSA mit der Programmiersprache CPL (vgl. Abschnitt 2.4.4).

4 Realisierung der Wissensmodelle

Auf Basis des im vorigen Kapitel beschriebenen Konzepts eines *Knowledge Configured Vehicle* wurde mit der kognitiven Systemarchitektur COSA ein Prototyp für eine Funktionsdemonstration entwickelt, welcher in diesem Kapitel beschrieben wird. Abbildung 4-1 zeigt das Systemschaubild, wie die KCV-ACU mit den wesentlichen Komponenten des UAV-Systems interagiert.

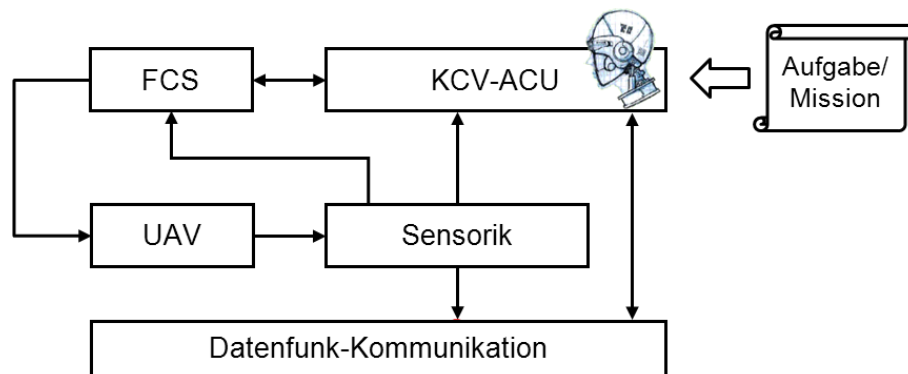


Abbildung 4-1. Systemschaubild der Integration der KCV-ACU in die wesentlichen Komponenten eines UAVs

Bei der Entwicklung der ACU wurde der Fokus auf die Umsetzung des KCV-Konzepts – dabei vor allem der Schwerpunkt auf die fluggerätspezifischen Wissensanteile – und weniger auf die Missionsfunktionen gelegt. Auch beschränkt sich die Interaktion der KCV-ACU mit dem Menschen auf das Empfangen eines Missionsdokuments (vgl. Abschnitt 6.1.1) und das Senden von Nachrichten über den aktuellen Verlauf der Mission und gegebenenfalls von Warnungen, bei technischen Problemen. Aus diesem Grund wurde für die Ausprägung der ACU die Form einer *Supporting Cognitive Unit* – als Teil der *Operation Supporting Means* – gewählt. Die von einem menschlichen Operateur definierte Mission beschreibt eine vergleichsweise einfache Aufklärungsaufgabe, welche hochautomatisiert durchgeführt werden soll (vgl. Kapitel 6).

In Abschnitt 4.1 werden zunächst die verwendeten COSA-Komponenten, notwendig für die Interaktion der ACU mit der Umwelt, erläutert und nachfolgend die Programmiersprache CPL, welche für die Modellierung des a-priori Wissens verwendet wurde, beschrieben (Abschnitt 4.2). Die Gliederung von entwickeltem a-priori Wissen in entsprechende Wissenspakete wird anschließend, im Hinblick auf die von [Putzer, 2004] behauptete generelle Wiederverwendbarkeit von a-priori Wissen, diskutiert (Abschnitt 4.3). Im weiteren Verlauf werden die für die Umsetzung des Konzepts eines KCVs nötigen Wissenspakete, beginnend bei dem Wissen über die KCV-Schicht (Abschnitt 4.4), das maschinenspezifische Wissen (Abschnitt 4.5) und das missionspezifische Wissen (4.6) beschrieben. Abschließend erläutert Abschnitt 4.7 die Interaktion aller Wissensmodelle an dem Beispiel des Anforderns von gewünschtem

Verhalten durch das Missionswissen. Abschnitt 4.8 gibt eine kurze Zusammenfassung dieses Kapitels mit einer Übersicht über die Anzahl der implementierten Klassen und CPL-Produktionen.

4.1 Serverkomponenten

Der Aufbau der ACU, mit der kognitiven Systemarchitektur COSA, entspricht der in Abbildung 2-16 gezeigten Konfiguration. Für die Ein- / Ausgabe und Berechnung von Informationen kommen die drei möglichen Servertypen zum Einsatz, nämlich jeweils ein Input-Server, ein Output-Server und ein Function-Server. Jeder dieser Server entspricht einer Komponente, welche zur Laufzeit des COSA, ähnlich einem Plug-In, eingebunden wird und der ACU die Interaktion mit der Umwelt (Soft- und Hardwaremodule des unbemannten Fluggerätes) und die ausgelagerte Berechnung von Informationen ermöglicht.

Der Input-Server sammelt Umweltinformationen und bringt diese in das kognitive System ein. Wesentliche Informationen umfassen hierbei, die Attribute der Mission (z.B. Flugroute dargestellt durch mehrere Wegpunkte, an welchem Wegpunkt gestartet und gelandet werden soll und was die eigentliche Aufgabe der Mission ist), den aktuellen Zustand des UAVs, dessen Automation (Autopilot, Flugmanagementsystem) sowie relevante Informationen über den Zustand von Subsystemen. Diese umfassen die Trägheitsnavigationseinheit sowie den aktuellen Status und die Daten des Fluggerätes, des Antriebssystems, des Datenfunks, der Energieversorgung und der Nutzlast.

COSA bietet zwar einfache Optionen für die Berechnung und den Vergleich von numerischen Werten, jedoch ist eine Verarbeitung von komplexeren, mathematischen Funktionen nicht ohne Weiteres möglich bzw. nicht vorgesehen. Diesem Defizit kann durch den Einsatz eines Function-Servers begegnet werden. Im Fall der vorliegenden Arbeit berechnet der implementierte Function-Server unter anderem die Berechnung der Länge und abhängig der gewählten Fluggeschwindigkeit, die Flugdauer für das Abfliegen einer Wegpunktliste (vgl. Abschnitt 4.7).

Der Output-Server ermöglicht der ACU die Ausgabe von Anweisungen und Nachrichten an die Umwelt und wird für die Konfiguration der UAV-Automation oder das Absenden von Statusinformationen an den menschlichen Operateur verwendet.

Im folgenden Abschnitt wird die Modellierung des a-priori Wissens beschrieben, welches für den Aufbau und die Umsetzung des Konzepts eines *Knowledge Configured Vehicle* implementiert wurde.

4.2 Wissensmodellierung

Die Modellierung des a-priori Wissens erfolgt mit der *Cognitive Programming Language* – CPL (vgl. Abschnitt 2.4.4), in Form von Paketen (engl.: *package*) und Klassen (engl.: *class*). Über die Klassendefinition wird der Namensraum des Modells festgelegt und in Verbindung mit der KP-Bibliothek somit einem Transformator des Kognitiven Prozesses zugewiesen. Somit entsprechen die vier a-priori Wissensmodelle aus Abbildung 2-14 den folgenden vier Namensräumen:

belief	→	Umweltmodell (engl.: <i>Environment model</i>)
goal	→	Wunsch (engl.: <i>desire</i>)
plan	→	Handlungsalternative (engl.: <i>action alternative</i>)
schedule	→	Anweisungsmodell (engl.: <i>instruction model</i>)

Eine Klasse setzt sich einerseits aus Attributen (*attributes*) und andererseits aus Verhalten (*behavior*) zusammen. Dabei kennzeichnen Attribute die Klasse und beschreiben deren Charakteristik. Das Verhalten beschreibt, wann eine Instanz dieser Klasse erzeugt und auch wieder entfernt wird und mit welchen Werten (z.B. String oder numerischer Wert) die Attribute der Klasse belegt werden sollen. Zudem können Attribute auf andere Instanzen von Klassen im situativen Wissen verweisen.

Das folgende CPL-Beispiel beschreibt einen kleinen Ausschnitt des Wissensmodells für ein Inertialnavigationssystem der Firma Crossbow, welches später auch für die Flugzustandsmessung des UAV Demonstrators verwendet wurde (vgl. Abschnitt 5.4.2.3). Das in Abbildung 4-2 dargestellte Verhalten der Klasse **IMU**¹ umfasst drei Produktionen, nämlich:

1. Erzeuge Instanz
2. Definiere nominalen Status der Komponente
3. Definiere Anforderungen der Komponente

¹ Die in der Arbeit verwendeten Klassen- und Instanzennamen von Wissensmodellen wurden aus Darstellungsgründen gekürzt.

```

package IMU_XBOW_NAV420
{
    ...

    class <belief> IMU
    {
        attributes:
            string name                := |IMU|;
            string system-class        := |Avionic|;
            string description          := |Inertial measurement unit|;

        behavior:

            sp {IMU_XBOW_NAV420*instance*creation
                ( instance[belief::EM_IMU::*]<instofimu> )
                ( <instofimu> ^Manufacturer |CrossBow| )
                ( <instofimu> ^Model |NAV420| )
            -->
                ( create <this> + = )
                ( <this> ^Requires <reqs> )
                ( <this> ^NominalState <nomstate> )
            }

            sp {IMU_XBOW_NAV420*nominal*state*init
                ( instance <this> )
                ( <this> ^NominalState <nomstate> )
            -->
                ( <nomstate> ^DataRate 50 )
                ( <nomstate> ^AlgorithmAccuracy |FULL_NAV| )
                ( <nomstate> ^GPSStatus |LOCKED| )
            }

            sp {IMU_XBOW_NAV420*requirements*init
                ( instance <this> )
                ( <this> ^Requires <reqs> )
            -->
                ( <reqs> ^Computer <comp> )
                ( <comp> ^providing |Computational power Flightcontrol| )
                ( <reqs> ^PowerSupply <ps> )
                ( <ps> ^providing |Electric power Avionic| )
            }

            ...

        }; // End of class
    }; // End of package

```

Abbildung 4-2. CPL-Codebeispiel

Wird eine Produktion aufgrund von erfüllten Vorbedingungen ausgeführt, so spricht man vom „Feuern“ der Regel. Die erste Produktion definiert, unter welchen Begebenheiten eine Instanz der Klasse *IMU* angelegt wird und ist wie folgt zu lesen:

„Wenn es eine Instanz der Klasse EM_IMU (EM_IMU ist eine Klasse, welche die IMU-Eingangsdaten des Input-Servers beinhaltet) im situativen Wissen gibt
und diese über ein Attribut „*Manufacturer*“ verfügt, das mit dem Wert „*CrossBow*“ belegt ist
und diese über ein weiteres Attribut „*Model*“ verfügt, das mit dem Wert „*NAV420*“ belegt ist,
dann erstelle eine Instanz der Klasse IMU mit den Attributen *name*, *system-class*, *description*, *Requires* und *NominalState* (wovon nur die ersten drei Attribute mit den Werten „*IMU*“, „*Avionic*“ und „*Inertial measurement unit*“ belegt werden)“.

Für das Anlegen der Instanz einer Klasse gibt es zwei verschiedene CPL-Anweisungen, nämlich „*create*“ (welches im Beispiel verwendet wurde) und „*elaborate*“. Sie unterscheiden sich darin, dass mit „*elaborate*“ erzeugte Instanzen sich automatisch

wieder zerstören, wenn eine der Vorbedingungen der entsprechenden Produktion nicht mehr erfüllt ist. Im Gegenzug sind Instanzen, die mit „create“ angelegt werden, dauerhaft im Situationswissen, bis diese durch eine eigene Regel zerstört werden (CPL-Anweisung „destroy“).

Dieses Beispiel der ersten Regel (IMU_XBOW_NAV420*instance*creation) zeigt, dass Attribute sowohl anfangs in der Klasse definiert und mit Standard-Werten belegt, als auch zur Laufzeit angehängt werden können. Dabei müssen die Attribute jedoch nicht zwingend mit Werten beschrieben, sondern können auch als Aufhängepunkte – wie beispielsweise *Requires* und *NominalState* – für weitere Attribute verwendet werden.

Die beiden anderen Produktionen übernehmen die Initialisierung der Klassenattribute *Requires* und *NominalState*, welche bei der Instanziierung angelegt werden. Die zweite Produktion definiert dabei die drei wichtigsten Zustände der CrossBow NAV420 IMU, welche auf die volle Einsatzbereitschaft des Systems schließen lassen und somit dem erwünschten bzw. nominalen Status entsprechen. Diese Produktion liest sich wie folgt:

„Wenn es eine Instanz der eigenen Klasse im situativen Wissen gibt
und diese Instanz über ein Attribut mit dem Namen „*NominalState*“ verfügt,
dann hänge ein Attribut mit dem Namen „*DataRate*“ und dem Wert „50“ an das Attribut „*NominalState*“
und hänge ein Attribut mit dem Namen „*AlgorithmAccuracy*“ mit dem Wert „*FULL_NAV*“ an das Attribut „*NominalState*“
und hänge ein Attribut mit dem Namen „*GPSStatus*“ mit dem Wert „*LOCKED*“ an das Attribut „*NominalState*“.

Die dritte und letzte Produktion dieses Beispiels nimmt die Initialisierung der Vorbedingungen und benötigten anderen Avioniksysteme vor, welche die IMU für den Betrieb benötigt. Sie liest sich wie folgt:

„Wenn es eine Instanz der eigenen Klasse im situativen Wissen gibt
und diese Instanz über ein Attribut mit dem Namen „*Requires*“ verfügt,
dann hänge ein weiteres Attribut mit dem Namen „*Computer*“ an das Attribut „*Requires*“
und hänge ein Attribut mit dem Namen „*providing*“ und dem Wert „*Computational power (flight control)*“ an das Attribut „*Computer*“
und hänge ein Attribut mit dem Namen „*PowerSupply*“ an das Attribut „*Requires*“
und hänge ein Attribut mit dem Namen „*providing*“ und dem Wert „*Electric power Avionic*“ an das Attribut „*PowerSupply*“.

4.2.1 Darstellung von situativem und a-priori Wissen

Nachfolgend soll das, im Zuge der Umsetzung des Konzepts eines *Knowledge Configured Vehicle*, implementierte Wissen anhand von Abbildungen dargestellt werden, welche Ausschnitten des situativen Wissens mit den zugehörigen Modellen des a-priori Wissens entsprechen. Die gewählte Darstellung orientiert sich an der von [Meitinger, 2008] verwendeten, welche fünf graphische Elemente vorsieht:


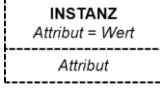
- Kästen (umrandet mit durchgezogenen Linien, ) symbolisieren a-priori Wissensmodelle. Oberhalb des Namens der Klasse wird der Namensraum (engl.: *namespace*) angezeigt, welcher die Zugehörigkeit zu einer Modellart (*belief*, *goal*, *plan*, *schedule*) bezeichnet und entsprechend farbig kodiert (gelb, blau, grün, rot) ist.
- Kästen (umrandet mit gestrichelten Linien, ) symbolisieren Instanzen im Situationswissen, welche ebenfalls entsprechend der Zugehörigkeit farbig kodiert sind (gelb, blau, grün, rot). Unterhalb des fett gesetzten Namens der Instanz werden die Attribute der Klasse in kursiv dargestellt, welche beim Anlegen der Instanz erzeugt und ggf. auch mit Standardwerten belegt werden. Weitere Attribute werden durch eigene, abgesetzte, gestrichelte Kästen symbolisiert (entweder direkt unterhalb des Kastens der Instanz, oder mit gepunktetem Pfeil abgesetzt), wenn diese erst zur Laufzeit (nach der Instanziierung) durch Produktionen angefügt werden.
- Gestrichelte Pfeile (---->) zeigen an, zu welcher Modellklasse des a-priori Wissens eine Instanz im situativen Wissen gehört („ist Instanz von“).
- Gepunktete Pfeile (.....>) zeigen an, dass ein Attribut an einem anderen Attribut angegliedert ist („hängt an“).
- Durchgezogene Pfeile (—>) stellen Attribute in Form von Verweisen dar, welche auf andere Instanzen im Situationswissen zeigen. Der Name dieses Verweis-Attributs wird in kursiv an dem Pfeil dargestellt.

Abbildung 4-3 zeigt einen Ausschnitt des Situationswissens nach Abarbeitung der Produktionen der Klasse **IMU** nach dem CPL-Beispiel (vgl. Abbildung 4-2) des vorigen Abschnitts. Sobald es eine Instanz der Klasse **EM_IMU** mit den Attributen *Manufacturer* und *Model* und den Werten *CrossBow* und *NAV420* gibt (diese Daten werden von dem Input-Server bereitgestellt und von der Klasse **EM_IMU** vorverarbeitet), feuert die Regel „IMU_XBOW_NAV420*instance*creation“ der Klasse **IMU**. Dadurch wird eine Instanz dieser Klasse, mit den vordefinierten Klassen-Attributen (*name*, *system-class*, *description*) und den zusätzlichen Attributen *Requires* und *NominalState* angelegt.

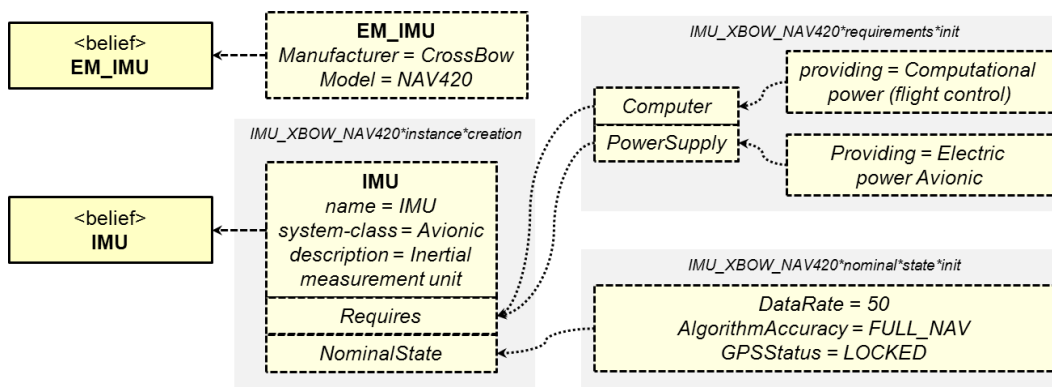


Abbildung 4-3. Beispiel für einen Ausschnitt aus dem situativen Wissen

Nachfolgend feuern die Regeln „IMU_XBOW_NAV420*nominal*state*init“ und „IMU_XBOW_NAV420*requirements*init“, welche zum einen den nominalen Betriebszustand für diese hardwarespezifische Komponente definieren und zum anderen eine Liste an für den Betrieb notwendigen Elementen auflisten, welche später für den dynamischen Aufbau eines Systemverständnisses der ACU verwendet werden (vgl. Abschnitt 4.5.2).

4.3 Wissenspakete

Ziel bei der Modellierung und Entwicklung der Wissenspakete durch einen Wissensingenieur ist es, eine hohe Wiederverwendbarkeit der einzelnen Pakete zu gewährleisten. Hierdurch können Synergieeffekte genutzt werden, welche einerseits zu einer Reduktion der Entwicklungszeit führen und andererseits mögliche Projektkosten senken, da bereits implementierte und gegebenenfalls validierte Wissensmodelle zur Anwendung kommen können. Putzer beschreibt diesen Sachverhalt mit folgenden Worten:

Pakete sichern einen hohen Grad an Wiederverwendung, indem die Funktionalität eines Paketes in mehreren Kontexten genutzt werden kann. Dabei wird die Granularität von Paketen allein durch den Entwickler bestimmt. So sind Pakete für verschiedene Teilfunktionen denkbar. [Putzer, 2004]

Putzer postuliert pauschal den hohen Grad der Wiederverwendbarkeit der einzelnen Wissenspakete einer ACU und deren Verwendung für verschiedene Kontexte. Er beschreibt, dass *generell* verschiedene Wissensarten aus dem bestimmten Bereich einer Domäne (z.B. Landfahrzeug, Luftfahrzeug, Wasserfahrzeug) mit Anwendungswissen (z.B. Tutor, Assistent, autonom) kombiniert werden können, um ein Zielsystem zu erhalten (vgl. Abbildung 4-4).

Dies ist per se durch das COSA und die Programmiersprache CPL zwar möglich, jedoch bietet das COSA Framework keinerlei formale Unterstützung, im Hinblick auf die Wissensorganisation und eine einheitliche Ontologie, welche aber Voraussetzung für die Entwicklung von wiederverwendbaren a-priori Wissenskomponenten ist. So bleibt es dem Wissensingenieur überlassen, aussagekräftige und intuitive Beschreibungen zu finden und diese auch entsprechend zu dokumentieren. So könnte beispielsweise die Klasse **IMU** (vgl. Abschnitt 4.2) für ein anderes Projekt verwendet werden, wenn ebenfalls das Inertialnavigationssystem NAV420 der Firma Crossbow eingesetzt würde. Voraussetzend hierfür ist jedoch, dass die anderen verwendeten Wissensmodelle Instanzen der Klasse **IMU** erwarten und nicht etwa Instanzen einer Klasse **TraegheitsNavigationsSystem**.

Putzer unterscheidet nach Abbildung 4-4 zwischen drei verschiedenen Arten von Domänenwissen, nämlich Wasser-, Luft- und Landfahrzeuge, und differiert zudem zwischen drei Arten von Anwendungswissen, welche folgend kurz erläutert werden:

- **Autonom:** Das Wissen sorgt für eine selbständige Erfüllung der Mission oder Aufgabe.
- **Assistent:** Das Wissen unterstützt den Operateur bei der Durchführung einer Mission oder Aufgabe.
- **Tutor:** Das Wissen trainiert den Operateur und zeigt ihm wie eine Maschine bedient bzw. wie eine Mission oder Aufgabe erfüllt werden kann.

Die Kombination von Anwendungs- und Domänenwissen führt nach Putzer zu einem Zielsystem. In dieser Darstellung ist allerdings nicht klar ersichtlich, dass es neben dem menschlichen Operateur für die Ausbildung eines Zielsystems – vor dem Hintergrund der Einbettung auf einer realen Hardware – auch an Wissen über die Maschine, ihre spezifischen Komponenten und der verfügbaren Automation bedarf. Zwar könnte dieses Wissen implizit in dem Domänenwissen verankert sein, was jedoch wiederum zu einer starken Abhängigkeit und einer schlechten Wiederverwendbarkeit der Wissenspakete führen kann.

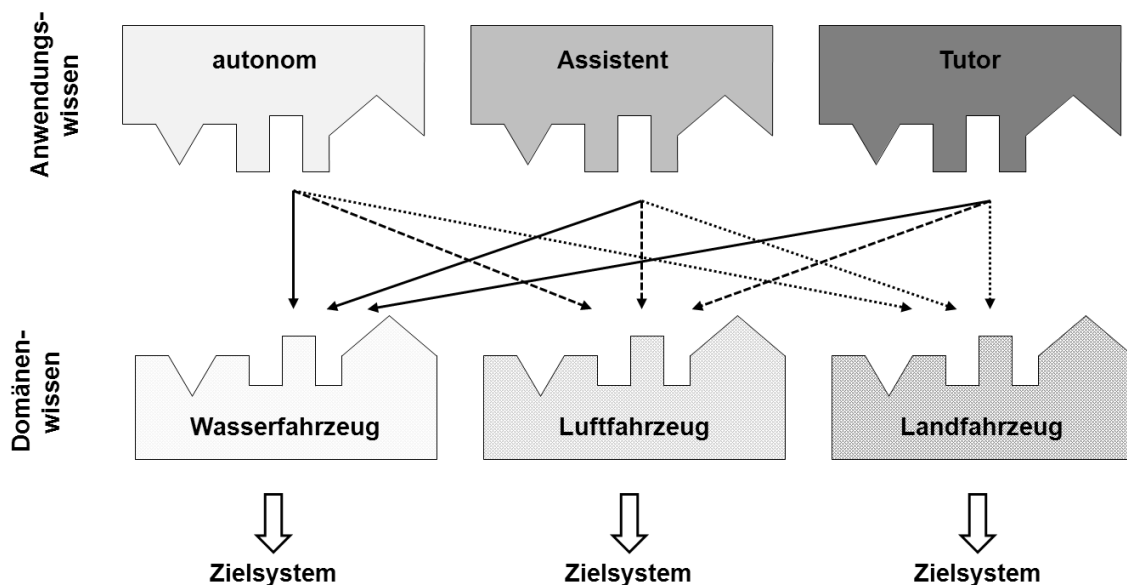


Abbildung 4-4. Kombination verschiedener Wissenspakete nach [Putzer, 2004]

Abbildung 4-5 zeigt eine erweiterte Darstellung der Wissenskombination nach Putzer, für den Fall eines „autonomen“, im Sinne eines hochautomatisierten Luftfahrzeugs. Das Anwendungswissen kann dabei in zwei Arten von einsatzspezifischem Wissen untergliedert werden:

- **Operateur-Interaktionswissen:** Ausgehend vom Einsatzzweck der ACU – wie beispielsweise „Autonom“, „Assistent“ und „Tutor“ – muss verschieden charakterisiertes Wissen hinterlegt werden, wie die Interaktion mit dem menschlichen Operateur erfolgt. Im Falle von „Autonom“ würden ggf. nur missionsrelevante Meldungen von der ACU an den Bediener gegeben werden, wogegen im Falle von „Tutor“ dem Menschen eine Anleitung bzw. ausführliche

Beschreibung des Vorgehens zur Erfüllung der Aufgabe oder Mission kommuniziert werden muss.

- **Spezielles Anwendungswissen:** Die Aufgabe der Kognitiven Einheit bestimmt die Eigenschaften und Charakteristik des einsatzspezifischen Anwendungswissens. So muss das Wissen im Falle von „Autonom“ in der Lage sein, selbständig eine Aufgabe bzw. Mission durchzuführen. Ist im Gegenzug die Aufgabe der ACU, den Menschen bei der Durchführung zu unterstützen („Assistent“). So wird Wissen benötigt, damit das kognitive System erkennt wann der Mensch Unterstützung benötigt und diese, entsprechend der aktuellen Situation, anbietet.

Im Gegenzug unterteilt sich das Domänenwissen ebenfalls in zwei verschiedene Wissensarten:

- **Allgemeines Domänenwissen:** Beschreibung von allgemeinem Wissen über die Domäne – in diesem Beispiel für ein Luftfahrzeug. Dieses beinhaltet hierbei generisches Wissen über den Vorgang des Fliegens und der Flugführung (vgl. Abschnitt 3.1.3).
- **Maschinenspezifisches Wissen:** Spezielles Wissen über die Eigenschaften und Fähigkeiten des Fluggerätes. Entspricht dem Type-Rating Wissen eines menschlichen Piloten (vgl. Abschnitt 3.1.3).

Die Interaktion zwischen dem *allgemeinen Domänenwissen* und dem *maschinenspezifischem Wissen* erfolgt über die KCV-Schicht, um eine Trennung zwischen diesen beiden Wissensarten zu erreichen und eine flexible Austauschbarkeit der Wissenspakete zu ermöglichen. Die Kombination der in Abbildung 4-5 dargestellten, oberen drei Wissenspakete (*Operateur-Interaktionswissen*, *spezielles Anwendungswissen*, *allgemeines Domänenwissen*) bezeichnet im Rahmen dieser Arbeit das missionspezifische Wissen, welches im Rahmen einer Beispielanwendung zur Erprobung des entwickelten KCV-Funktionsprototypen umgesetzt wurde.

Die Gesamtheit aller Wissenspakete entspricht dabei dem a-priori Wissen der ACU, welches in Verbindung mit der Hardware und dem menschlichen Operateur das Zielsystem im Sinne eines Arbeitssystems (vgl. Abschnitt 2.4.1) repräsentiert.

Die KCV-Schicht selbst ist ebenfalls durch ein a-priori Wissenspaket repräsentiert, welches eine feste Nomenklatur im Situationswissen erzeugt. Deren Notation ist zwar während der Entwicklungszeit frei wählbar, aber nachfolgend soll sie einen organisierten Austausch von situativem Wissen zwischen den missions- und maschinenspezifischen Wissenspaketen ermöglichen, ohne jedoch schnittstellen-spezifische Einschränkungen zu oktruieren.

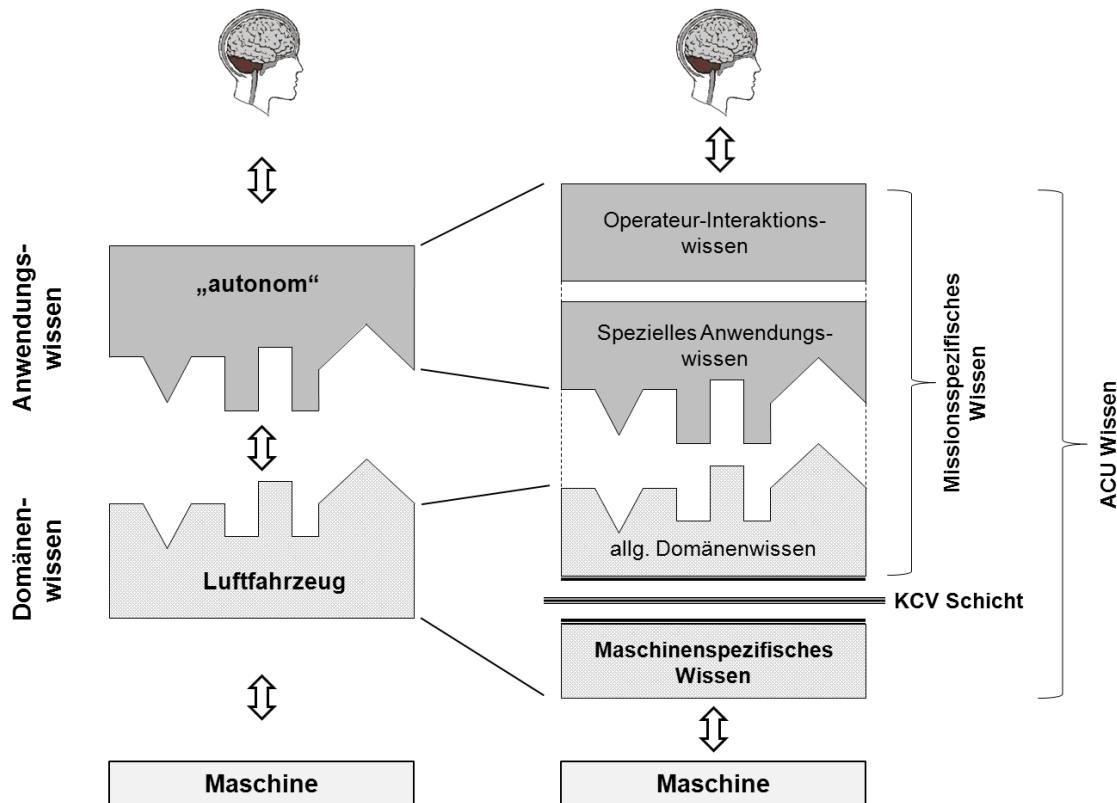


Abbildung 4-5. Erweiterung der Kombination von Wissenspaketen nach [Putzer, 2004] mit Berücksichtigung von maschinenspezifischem Wissen und der Hardware nach dem Konzept eines KCV

Die folgenden Abschnitte beschreiben einige der Implementierungsdetails der KCV-Schicht, sowie Teile der maschinen- und missionsspezifischen Wissensmodelle, welche mittels der kognitiven Programmiersprache CPL umgesetzt wurden. Ferner wird die Prozedur der Erstellung eines Systemverständnisses – der kognitiven Einheit – erläutert und die Kommunikation zwischen den Wissenspaketen sowie der Mechanismus des Anforderens von *gewünschtem Verhalten* detailliert.

4.4 A-priori Wissen der KCV-Schicht

Im Situationswissen der ACU übernimmt die Klasse `kcv` mittels verschiedener Attribute die in Abschnitt 3.3 beschriebene Organisationsschicht nach dem Konzept eines *Knowledge Configured Vehicle*. Diese (Organisations-)Attribute bezeichnen die Aufhängepunkte für die (*verfügbaren*) *Fähigkeiten* (engl.: *(Available) Abilities*) oder *gewünschtes Verhalten* (engl.: *WantedBehavior*), in Anlehnung an Abbildung 3-3.

Durch die Repräsentation von situativem Wissen in Form eines semantischen Netzes ermöglicht es COSA, eine beliebige Anzahl von weiteren Attributen an die Organisationsattribute anzuhängen (vgl. Abschnitt 4.5.3). So können beispielsweise die abstrahierten *Fähigkeiten* eines UAVs, abhängig von der verwendeten Automation und der Nutzlastkomponenten, an dem Attribut *Abilities* gelistet und somit der übergeordneten Missionsschicht verfügbar gemacht werden.

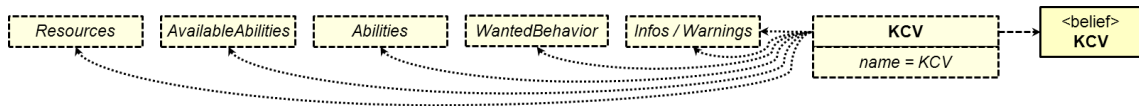


Abbildung 4-6. Aufbau der Organisationsstruktur der KCV-Schicht im Situationswissen

Die durch diese Klasse repräsentierte Organisationsschicht verfügt selbst über kein Verarbeitungswissen hinsichtlich eigener Wünsche, Handlungsalternativen und Anweisungsmodellen und hat demnach eine ausschließlich strukturierende Funktion.

4.5 Maschinenspezifisches Wissen

Das maschinenspezifische Wissen übernimmt einerseits die Aufgabe, das Fluggerät zu überwachen und zu konfigurieren, wofür die ACU Wissen über die einzelnen Subsysteme der Maschine sowie deren Betriebsgrenzen und Verschaltung benötigt. Andererseits ist Wissen über die Nutzlastkomponenten, die verfügbare Automation und ein Verständnis derer Betriebsarten erforderlich, um die Abstraktion der *Fähigkeiten* des Luftfahrzeugs vornehmen zu können.

Abschnitt 4.5.1 gibt anfangs einen Überblick über die wichtigsten Umweltmodelle von System- und Softwarekomponenten. Deren Beziehung und Verschaltung und das daraus erzeugte Systemverständnis der ACU wird im folgenden Abschnitt 4.5.2 detailliert. Im weiteren Verlauf beschreibt Abschnitt 4.5.3 den Vorgang der Abstraktion von *Fähigkeiten* und das Zusammenspiel mit der KCV-Schicht, welche im vorigen Kapitel beschrieben wurde. Alle Wissensmodelle wurden in Anlehnung an die realen Hard- und Softwarekomponenten implementiert und entwickelt, welche für die spätere Evaluation des *Knowledge Configured Vehicle*-Prototypen zum Einsatz kommen (vgl. Kapitel 5).

4.5.1 System- / Softwarekomponenten

Die ACU benötigt für die Überwachung und die Bewertung des Systemzustands, Wissen über die einzelnen System- und Softwarekomponenten, welche für einen automatisierten Betrieb des UAVs notwendig sind. Dieser Abschnitt beschreibt einen Ausschnitt an hierfür notwendigen Wissensmodellen, welche für den späteren Aufbau eines Systemverständnisses der ACU verwendet werden und den realen Komponenten (vgl. Abschnitte 5.4.1 und 5.4.2) zugeordnet sind. Diese wurden für den Aufbau des UAV-Demonstrators verwendet, im Folgenden:

- Flight Control Computer (FCC)
- Sensordatenerfassungssystem (SDSYS)
- Stromversorgung (Power supply)
- Funkverbindung (Datalink)
- Trägheitsnavigationssystem (IMU)
- Autopilot (inklusive Flugmanagementsystem)

Ähnlich dem Beispiel des Wissensmodells für das Trägheitsnavigationssystem (IMU) in Abschnitt 4.2 werden die folgenden Instanzen der Wissensmodelle auf Basis der Eingangsdaten durch den Input-Server aufgenommen. Durch entsprechende Klassen

werden diese Daten aus technischen Gründen vorverarbeitet (gepuffert), ohne jedoch verändert zu werden, um die Verarbeitungsgeschwindigkeit des COSA zu erhöhen.

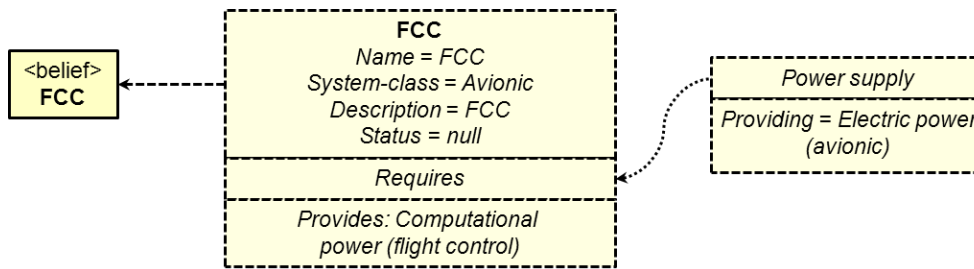


Abbildung 4-7. Situationswissen des Modells FCC

Abbildung 4-7 stellt eine Instanz der Klasse **FCC** dar, welche dem Wissensmodell für einen Rechner entspricht, auf dem später ein Großteil der FCS-Softwarekomponenten (vgl. Abschnitt 5.4.1.2) ausgeführt werden. Für den Betrieb des Rechners (Attribut *Requires*) definiert die Klasse **FCC** nur ein einziges, benötigtes Element, nämlich die Stromversorgung (Attribut *Power supply*). Das Attribut *Provides* im Gegenzug gibt an, was diese Komponente bereitstellt. In diesem Fall ist es die Rechenleistung für Softwareprozesse, welche für die Flugregelung und die Dekodierung von Sensordatenpaketen (z.B. IMU, Sensordatenerfassungssystem) des UAV erforderlich sind (*Computational power (flight control)*).

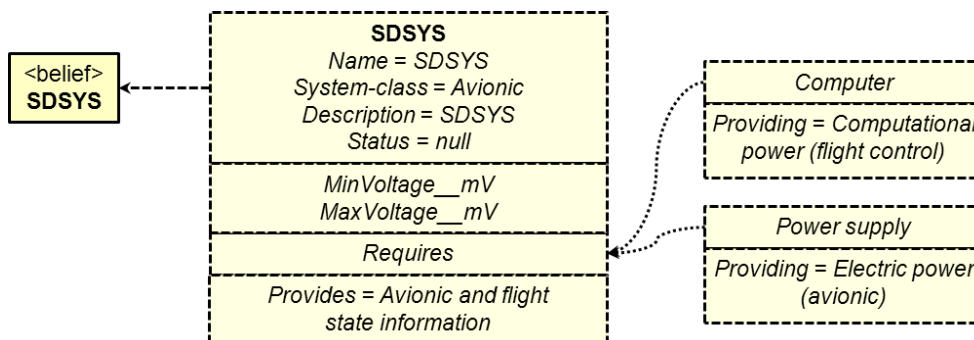


Abbildung 4-8. Situationswissen des Modells Sensordatenerfassungssystem (SDSYS)

Abbildung 4-8 zeigt einen Ausschnitt des Situationswissens für das Wissensmodell **SDSYS**, welches einem zusätzlich verwendeten Sensordatenerfassungssystem entspricht und Werte erfassen kann, die nicht von der IMU oder anderen Sensoren der Avionik sensiert werden (vgl. Abschnitt 5.4.1). Für einen Betrieb dieser Komponente werden die Elemente Stromversorgung (Attribut *Power supply*) und ein Rechner (Attribut *Computer*) definiert, auf welchem später der Softwaretreiber für diese Komponente ausgeführt wird (vgl. Abschnitt 5.4.2.2) und für die Dekodierung der erfassten Sensordaten sorgt. Die Parameter *MinVoltage_mV* und *MaxVoltage_mV* definieren die Grenzen des Eingangsspannungsbereichs dieses Gerätes, wodurch die elektrische Versorgung des Gerätes überwacht werden kann.

Abbildung 4-9 zeigt eine Instanz der Klasse **PowerSupply** im situativen Wissen. Durch die Attribute *Manufacturer* (Hersteller), *Model* (Modell), *Type* (Stromversorgungstyp) werden die Eigenschaften der Stromversorgung festgelegt. Aufgrund dieser Angaben können Berechnungsvorschriften (Entladekurven) gewählt werden, welche eine

Vorhersage der verbleibenden Zeit, für eine zuverlässige Stromversorgung der Avionikkomponenten, in Verbindung mit dem implementierten Function-Server erlaubt. Der aktuelle Status der Stromversorgung wird durch die beiden Attribute *Voltage_mV* (aktuelle Spannungslage) und *Current_mA* (momentaner Stromverbrauch) beschrieben, dessen Messgrößen von dem Sensordatenerfassungssystem (SDSYS) erhoben werden.

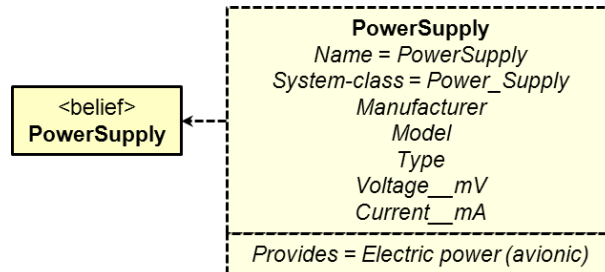


Abbildung 4-9. Situationswissen des Modells Stromversorgung (PowerSupply)

Ein weiteres Element im situativen Wissen ist die Instanz *Datalink*, wovon ein Ausschnitt in Abbildung 4-10 dargestellt ist. Die Funkverbindung stellt während der Flugversuche (vgl. Abschnitt 5.4.2.4) die Kommunikation zwischen der Bodenkontrollstation und dem UAV her, was auch durch das Attribut *Provides* mit dem Wert *Long range datalink connection* formuliert wird. Für den Betrieb dieser Komponente werden ein Computer und eine Stromversorgung als benötigt definiert. Die beiden Attribute *Integrity* und *Datarate* sind ein Maß für die aktuelle Zuverlässigkeit und Auslastung der Funkverbindung und werden von der Softwarekomponente *Kommunikationscenter* (COMC, vgl. Abschnitt 5.6) berechnet. Mittels dieser Werte kann die Kommunikationseinrichtung durch weitere Produktionen bewertet und der Zustand (Attribut *Status*) entsprechend gesetzt werden.

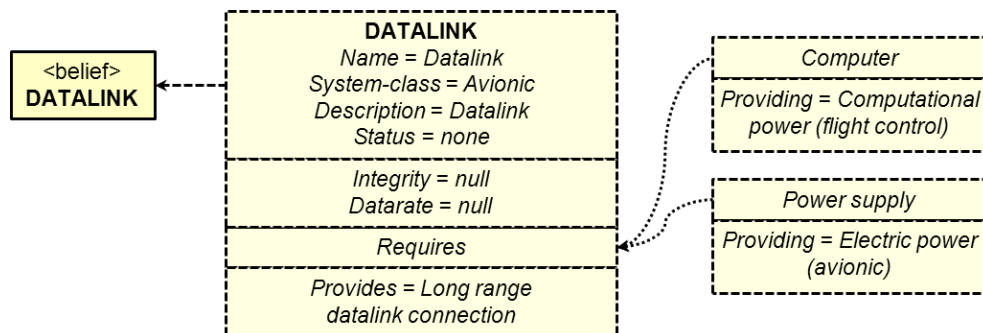


Abbildung 4-10. Situationswissen des Modells Datenfunkverbindung (Datalink)

Abbildung 4-11 stellt eine Instanz der Klasse **IMU** dar, deren Attribute *Provides* die elementaren Daten benennen, welche von diesem Sensor erfasst werden. Zugleich ist hier zu sehen, dass es CPL erlaubt, mehrere Attribute mit dem gleichen Namen zu verwenden und diese mit unterschiedlichen Werten zu belegen. Das Attribut *Status* gibt Auskunft über den aktuellen Status des Trägheitsnavigationssystems, im Hinblick auf den definierten nominalen Zustand, welcher bereits in Abschnitt 4.2 beschrieben wurde und hier aus Gründen der Übersichtlichkeit nicht nochmals abgebildet wird.

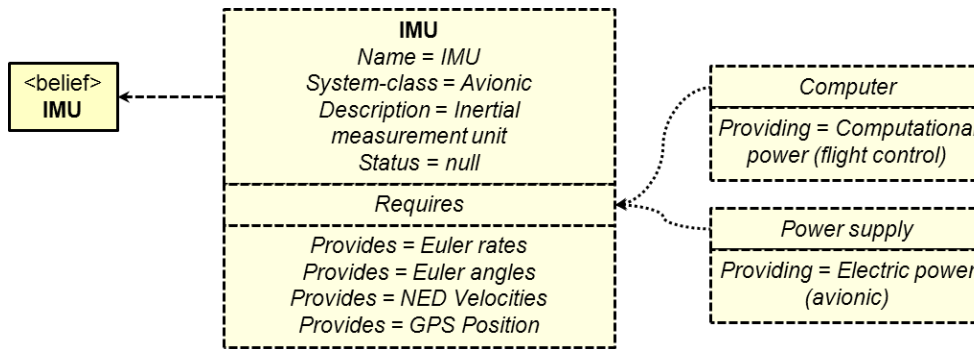


Abbildung 4-11. Situationswissen des Modells Inertialnavigationssystem (IMU)

Im Gegensatz zu den bereits beschriebenen Hardwarekomponenten stellt Abbildung 4-12 einen Auszug aus dem Situationswissen für das Softwareelement **Autopilot** (Autopilot und Flugmanagementsystem) dar. Hier definiert das Attribut *Requires* eine Vielzahl an Anforderungen, welche für den Betrieb dieses Moduls benötigt werden – nämlich ein Inertialnavigationssystem (*IMU*) für die Messung des aktuellen Flugzustands, ein Rechner (*Computer*) – auf welchem diese Softwarekomponente ausgeführt werden kann – eine entsprechende Stromversorgung (*Power supply*) und zusätzlichen Systeminformationen. Neben den deskriptiven Attributen *Name*, *System-class* und *Description* gibt das Attribut *Model* mit dem Wert *BIG I* an, für welches spezielle Fluggerät der Flugregler ausgelegt wurde. Das Attribut *Providing* definiert hier sehr pauschal, dass dieses Modul in der Lage ist – in Verbindung mit der speziellen Maschine – Verhalten (engl.: *Behavior*) bereitzustellen, was in Abschnitt 5.4.5 noch detaillierter erläutert wird.

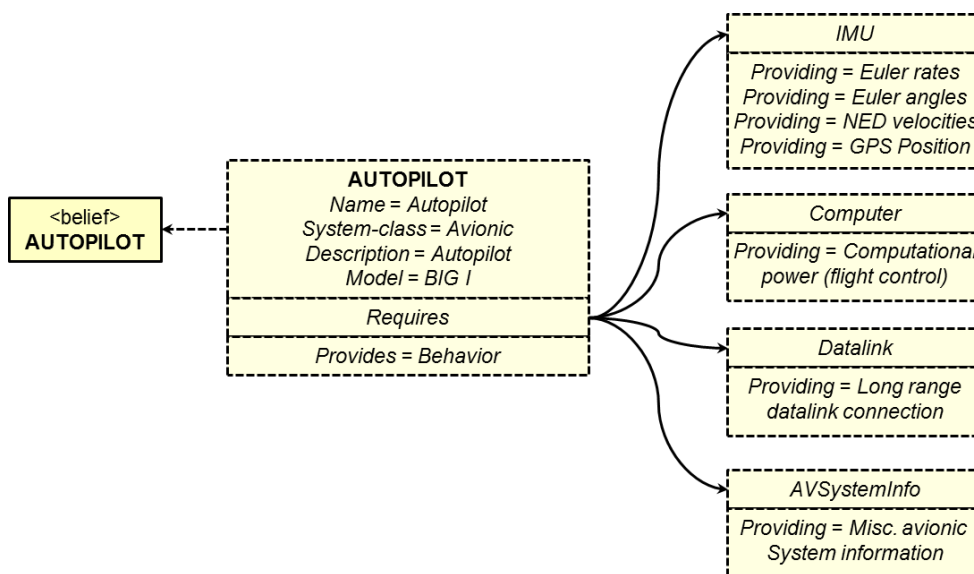


Abbildung 4-12. Situationswissen des Modells Autopilot

Die vorgestellten Klassen und die zugehörigen Instanzen geben einen kurzen Überblick über die verwendeten Wissensmodelle. Durch weitere Produktionen der einzelnen Modelle werden vor allem, abhängig von den verwendeten Hardwarekomponenten, die zugehörigen Betriebsgrenzen definiert und durch entsprechende Regeln überwacht. Somit kann einerseits die physikalische Überwachung (z.B. Überwachung des

Eingangsspannungsbereichs bei dem Sensordatenerfassungssystem) für einzelne Hardwareelemente realisiert und andererseits können auch funktionale Betriebsparameter (Überwachung der Integrität und der Auslastung der Funkverbindung) bewertet werden. Basis und Ursprung dieser Informationen sind meist die Anleitungen und Betriebsvorschriften der entsprechenden Geräte, in (Labor-)Versuchen ermittelten oder von einem Systemingenieur festgelegten Limitationen.

Die funktionale und physikalische Überwachung einzelner Komponenten ermöglicht es jedoch nicht, eine Aussage über die Einsatzfähigkeit des Gesamtsystems zu treffen. Hierfür muss zusätzliches a-priori Wissen hinterlegt werden, das ein Verständnis der ACU, im Hinblick auf die Verschaltung der einzelnen Komponenten, ermöglicht. Hierfür sollen die bereits beschriebenen Wissensmodelle durch weitere Wissensmodelle in Beziehung gesetzt werden, worauf der folgende Abschnitt näher eingehen wird.

4.5.2 Aufbau eines Systemmodells im situativen Wissen

Im vorausgegangenen Abschnitt wurden einige Umweltmodelle beschrieben, welche aufgrund von Eingangsdaten entsprechende Instanzen der Klassen für die zugehörigen Hardwarekomponenten und Softwaremodule im Situationswissen erzeugen. Für die Bewertung des aktuellen Systemzustands benötigt das System jedoch zudem Kenntnis über die Verschaltung der einzelnen Subsysteme, um bei etwaigen Problemen sowohl die Auswirkungen beurteilen als auch die Abwesenheit von benötigten Komponenten identifizieren zu können. Wird beispielsweise vergessen, ein Subsystem einzuschalten, oder es liefert aufgrund eines technischen Defekts keine Daten, so würden die zugehörigen Informationen nicht an den Input-Server gereicht und somit auch keine entsprechende Instanz dieses Gerätes im Situationswissen erstellt werden.

Der hier beschriebene Regelfall geht allerdings davon aus, dass sämtliche Systeme einwandfrei arbeiten und keine technischen Probleme vorhanden sind. Demnach sind alle in Abschnitt 4.5.1 beschriebenen Instanzen der Wissensklassen im Situationswissen vorhanden (vgl. Abbildung 4-13). Neben den bereits beschriebenen Umweltmodellen sind noch drei weitere Klassen dargestellt – eine vom Typ Wunsch (**HAVE_AVC_REQ_FULFILLED** – Erfülle Anforderung der Avionik-komponente), eine vom Typ Handlungsalternative (**SEARCH_REQ_COMPONENT** – Suche erforderliche Komponente) und eine vom Typ Anweisungsmodell (**LINK_AVC_REQ** – Verbinde Anforderung mit gefundener Komponente) – die für den Aufbau des (technisch mentalen) Systemmodells im situativen Wissen der ACU verantwortlich sind.

Die Instanzen der Umweltmodelle definieren mittels eines *Requires* Attributs benötigte Geräte bzw. Komponenten, welche für deren Betrieb erforderlich sind. Im Gegenzug wird anhand der *Provides* Attribute festgelegt, welchen Beitrag diese Komponente erbringt bzw. welche Daten – im Falle eines Sensors – erhoben werden. So wird für jede beliebige Umweltmodellinstanz, die über ein Attribut *Requires* verfügt und deren Anforderung noch nicht erfüllt wurde, eine Instanz der Klasse **HAVE_AVC_REQ_FULFILLED** („Have avionic component requirement fulfilled“ – „Erfülle Anforderung eine Avionikkomponente“) erstellt und zudem mit einem Attribut *AVC_Req* („Avionic component requirement“ – „Benötigte Avionikkomponente“) versehen, welches auf das zugehörige Attribut verweist. Die diesem Wunsch angegliederte

Handlungsalternative sucht im Situationswissen nach Instanzen, die die Anforderung erfüllen kann (Abgleich der Attribute *Provides* und der *Providing* Attribute, welche an dem Attribut *Requires* angehängt sind). Wird eine solche Komponente gefunden, so wird eine Instanz dieser Klasse **SEARCH_REQ_COMPONENT** („Search requested component“ – „Suche benötigte Komponente“) mit einem Attribut *AVC_SAT* („Satisfying avionic component“ – „Anforderung erfüllende Avionikkomponente“) – welches einerseits auf die gefundene Instanz und andererseits mit dem Attribut *GOAL_AVC_REQ* auf den zugehörigen Wunsch verweist – erzeugt.

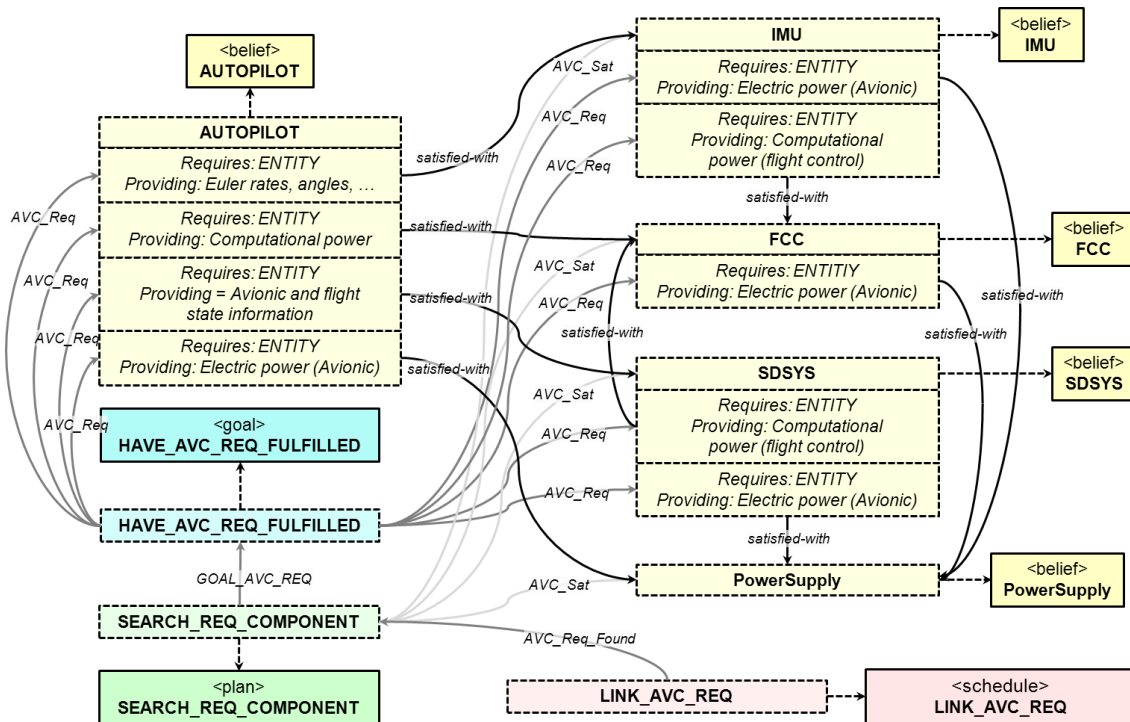


Abbildung 4-13. Aufbau eines Systemschaltbilds der Avionikkomponenten

Nachfolgend wird für jede angelegte Instanz der Handlungsalternative **SEARCH_REQ_COMPONENT** eine Instanz des Anweisungsmodells **LINK_AVC_REQ** erzeugt, die wiederum auf die entsprechende Handlungsalternative mit dem Attribut *AVC_Req_Found* verweist. Das Anweisungsmodell übernimmt nun die Verknüpfung der Anforderung mit der zugehörigen, gesuchten Komponente, indem ein Verweis *satisfied-with* („erfüllt mit“) zwischen dem *Requirement* Attribut und der Umweltmodellinstanz hergestellt wird. Sobald diese Verbindung erstellt wurde, zerstören sich die entsprechenden Instanzen der Klassen **HAVE_AVC_REQ_FULFILLED** und **SEARCH_REQ_COMPONENT** (aufgrund deren Erzeugung mittels der CPL-Anweisung „elaborate“). Sind alle Anforderungen erfüllt (also mit einem *satisfied-with* Verweis versehen), so ist das Resultat ein dynamisch erzeugtes Abbild der Avionikkomponenten und deren Verschaltung im Situationswissen der ACU. Im Falle der Abwesenheit einer Komponenteninstanz (aufgrund von ausbleibenden Daten über den Input-Server), die jedoch von anderen Elementen benötigt wird, ist die ACU in der Lage diese Komponente zu benennen und – entsprechendes a-priori Wissen vorausgesetzt – sowohl die MMF, als auch den menschlichen Operateur darüber in Kenntnis zu setzen.

4.5.3 Abstraktion von Fähigkeiten

Die Abstraktion von *Fähigkeiten* eines UAVs orientiert sich ausschließlich an den Möglichkeiten und Betriebsarten der verfügbaren Automation und der vorhandenen Nutzlastkomponenten. Abbildung 4-14 zeigt einen Ausschnitt aus dem Situationswissen der ACU, in dem exemplarisch zwei *Fähigkeiten* und deren zugehörige Anforderungen bezüglich des Systemzustands und der benötigten Ressourcen dargestellt sind.

Ist der Aufbau eines Systemverständnisses, welches im vorigen Abschnitt beschrieben wurde, abgeschlossen, so beginnt die ACU mit der Ermittlung der *Fähigkeiten* und ordnet diese in die entsprechenden, im oberen Bereich der Abbildung dargestellten Attribute (*Abilities* und *Available Abilities*) der Klasse **KCV** ein (Die anderen Attribute der Klasse sind aus Übersichtsgründen nicht dargestellt). Diesen Vorgang übernimmt die Klasse **ABILITIES**, welche aufgrund der vorhandenen Instanzen der Umweltmodelle, wie beispielsweise **AUTOPILOT** und **VEHICLE** die möglichen Betriebsmodi der Automation abstrahiert. So stellt zum Beispiel das Attribut *AttitudeControl*, welches an das Attribut *Abilities* der Klasse **KCV** angehängt wurde, die *Fähigkeit* dar, eine bestimmte vorgegebene Fluglage einzunehmen. Zusätzlich definiert die Klasse **ABILITIES** die beiden, weiteren Attribute *StatusRequirements* und *ResourceRequirements*. An diesen Attributen werden die Systemzustände und Ressourcen festgelegt, die für die Ausführung dieser *Fähigkeit* benötigt werden.

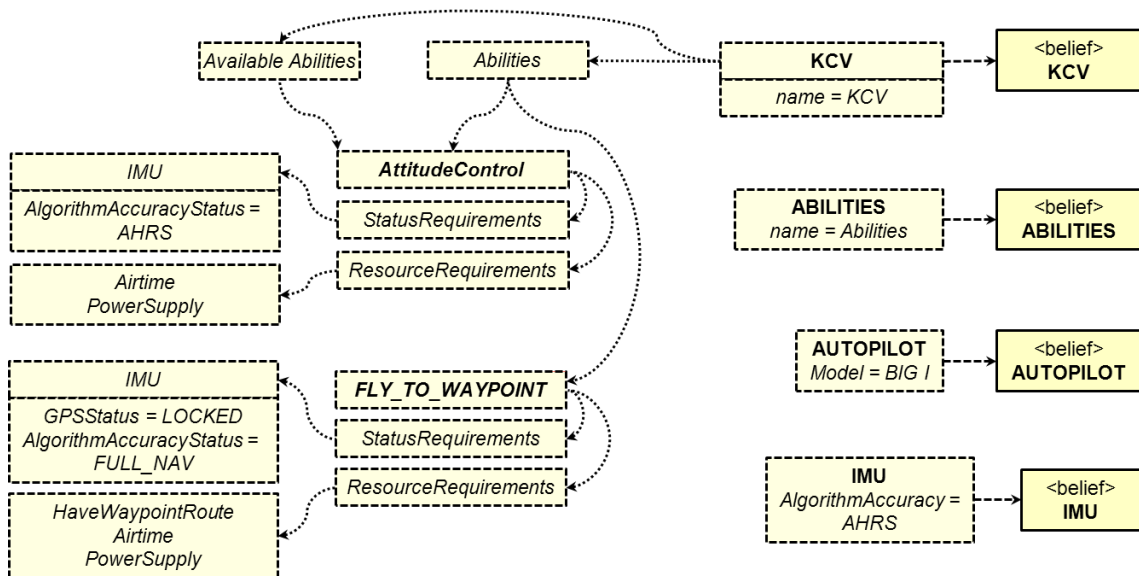


Abbildung 4-14. Abstraktion der Fähigkeiten eines UAV und deren Einordnung in die KCV-Schicht

In diesem Fall definiert die *Fähigkeit AttitudeControl*, dass die IMU mindestens den Systemzustand *AHRS* (engl.: Attitude and heading reference system – Das IMU-System liefert bereit zuverlässige Werte im Hinblick auf die Fluglage, jedoch ist noch keine Navigationslösung errechenbar) aufweisen muss. Die *Fähigkeit FLY_TO_WAYPOINT* („Fliege einen Wegpunkt an“) hingegen erfordert den Systemzustand *FULL_NAV* (engl.: Full navigation – Zuverlässige Navigationslösung inklusive Fluglageinformationen). Für die Ausführung beider Fähigkeiten werden sowohl Flugzeit (engl.: *Airtime*) – als Maß für die verbleibende Ressource Treibstoff – als auch Stromversorgung (engl.: *PowerSupply*) genannt.

Die Erstellung der entsprechend abstrahierten Fähigkeiten des UAVs erfolgt unabhängig vom aktuellen Systemzustand. Dieser wird durch die entsprechenden Umweltmodelle interpretiert und mit den benötigten Anforderungen der einzelnen *Fähigkeiten* abgeglichen. Sind alle Vorbedingungen erfüllt, so wird die zugehörige *Fähigkeit* als *verfügbar* gelistet und kann von dem missionsspezifischen Wissen für die Durchführung der Mission angefordert werden (vgl. Abschnitt 4.7).

4.6 Missionsspezifisches Wissen

Für die automatisierte Durchführung einer vom menschlichen Operateur definierten Mission benötigt die ACU Wissen, um diese einerseits interpretieren und verstehen zu können und andererseits wie und mit welchen Mitteln die gestellte Aufgabe erfüllt werden kann. Die Implementierung des missionsspezifischen Wissens orientiert sich stark an einer einfachen Aufklärungsmission (vgl. Abschnitt 6.1.1), welche für die Demonstration des implementierten Konzepts eines KCV definiert wurde. Die eigentlichen Missionsmanagementfunktionen verfügen dabei nur über geringe Planungsfunktionalitäten. Der eigentliche Missionsverlauf wird skriptartig abgearbeitet, da der Schwerpunkt bei der Wissensentwicklung und -modellierung auf das maschinenspezifische Wissen gelegt wurde.

Abbildung 4-15 zeigt einen Ausschnitt aus dem Situationswissen für die Klasse **Mission**, die sich dann instanziiert und mittels weiterer Produktionen initialisiert, wenn die Missionsdetails von dem Input-Server bereitgestellt werden. Neben dem Attribut *MissionStatus* – das während der Initialisierungsphase mit dem Wert *INIT* belegt wird – werden die Flugroute durch das Attribut *MissionRoute*, mit den zugehörigen Wegpunkten, sowie die Indizes des Start- und Landepunkts definiert. Die eigentliche Aufgabe während der Mission besteht darin, einen erdfesten Punkt zu überwachen, indem ein bildgebender Sensor auf diesen Punkt ausgerichtet wird. Dies wird durch die Attribute *PointOfInterest* mit der entsprechenden Ortskoordinate und dem Attribut *GroundSurveillance* festgelegt, dessen angehängte Attribute den Flugabschnitt bestimmen, in dem die Überwachung durchgeführt werden soll (vgl. Abschnitt 6.1.1).

Neben dem beschriebenen Umweltmodell verfügt die Klasse **Mission** noch über Wünsche und Handlungsalternativen, welche in Abbildung 4-16 dargestellt sind. Ausgehend von einem Zustand, dass alle Systeme nominal arbeiten und die Mission, wie oben beschrieben, initialisiert wurde (Attribut *MissionStatus* der Instanz **MISSION** hat den Wert *READY*), beginnt das kognitive System mit der Durchführung der Mission (Attribut *MissionStatus* der Instanz **Mission** hat den Wert *PROGRESS*) unter Verwendung der vorhandenen Handlungsalternativen (= *verfügbare Fähigkeiten*).

Abbildung 4-16 zeigt hier einen Ausschnitt verschiedener Handlungsalternativen (*plans*), entsprechend der durch das MasW abstrahierten Fähigkeiten. Im Fall der vorliegenden Arbeit wurden diese Handlungsalternativen aufgrund der geringen Planungsfunktionalität der MMF a-priori implementiert und um bei der späteren Evaluierung des Systems die Reaktion zeigen zu können, wenn Fähigkeiten in Form

von *gewünschtem Verhalten* angefordert werden, die nicht von der Automation bereitgestellt werden.

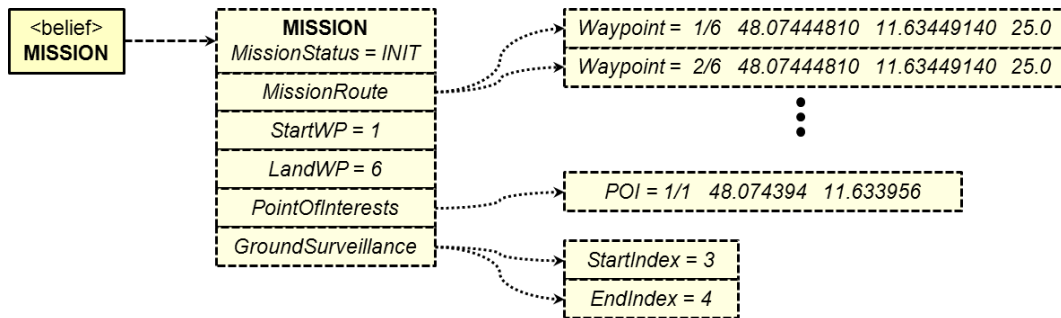


Abbildung 4-15. Beispiel eines Ausschnitts aus dem Situationswissen

Das Beispiel in Abbildung 4-16 zeigt einen Ausschnitt aus dem Situationswissen der ACU zu Beginn der Durchführung der Mission. Nach dem Abschluss der Missionsinitialisierung wird der Wunsch `ACCOMPLISH_MISSION` („Führe Mission durch“) instanziiert, was beispielsweise in diesem Stadium der Mission zur Erzeugung einer Instanz der Handlungsalternative `FOLLOW_WP_ROUTE` („Fliege Wegpunktliste ab“) führt. Die Ausführung dieser Handlungsalternative übernimmt die Klasse `REQUEST_BEHAVIOUR`, welche durch eine Instanz das Anfordern des *gewünschten Verhaltens* über die KCV-Schicht erwirkt (vgl. Abschnitt 4.7). Die Verweisattribute *related-goal* und *related-plan* geben zudem Aufschluss auf die zugehörigen Wünsche und Handlungsalternativen.

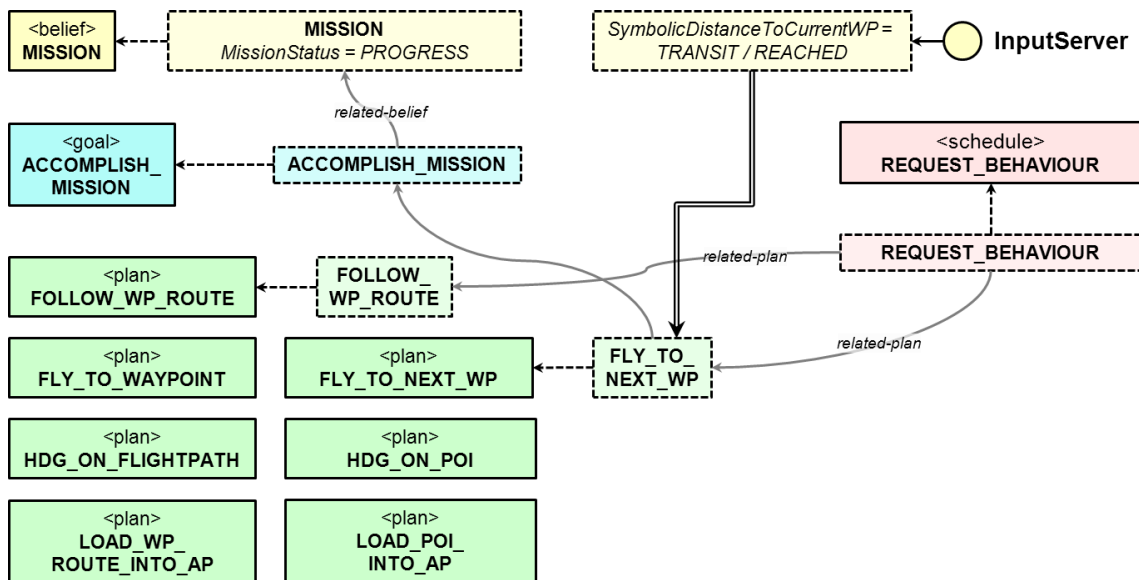


Abbildung 4-16. Beispiel für das Abfliegen einer Wegpunktroute und das Weiterschalten des Wegpunkts

Die Identifikation, ob ein Wegpunkt erreicht ist, wird durch den Input-Server vorgenommen und durch das Attribut *SymbolicDistanceToCurrentWaypoint* repräsentiert, welches aufgrund der aktuellen Positionsdaten des UAVs und des Wegpunkts eine Berechnung des Abstands der beiden Orte vornimmt. Fällt diese Längenangabe unterhalb eines vorher definierten Schwellwertes, so wird der Wert auf

REACHED gesetzt. Ist im Gegenzug der Abstand größer als ein zweiter Schwellwert, so nimmt das Attribut den Wert *TRANSIT* („Aktueller Wegpunkt noch nicht erreicht“) an (vgl. Abschnitt 6.2).

Wird zum Beispiel der derzeit aktive Wegpunkt erreicht und gibt es laut Mission keine an diesem Wegpunkt durchzuführende Aufgaben, so wird die Handlungsalternative *FLY_TO_NEXT_WP* erzeugt und mittels des Anweisungsmodells *REQUEST_BEHAVIOR* das Weiterfliegen zum nächsten Wegpunkt der Liste initiiert.

4.7 Anfordern von gewünschtem Verhalten

Nachdem in Abschnitt 4.4 der Aufbau der KCV-Schicht, in Abschnitt 4.5 Teile des maschinenspezifischen Wissens und in Abschnitt 4.6 Ausschnitte des missionspezifischen Wissens beschrieben wurden, widmet sich dieser Abschnitt allen Wissenspaketen und erläutert die Prozedur des Anforderns von *gewünschtem Verhalten* durch das MisW über die KCV-Schicht und die nachfolgende Konfiguration der Automation durch das MasW.

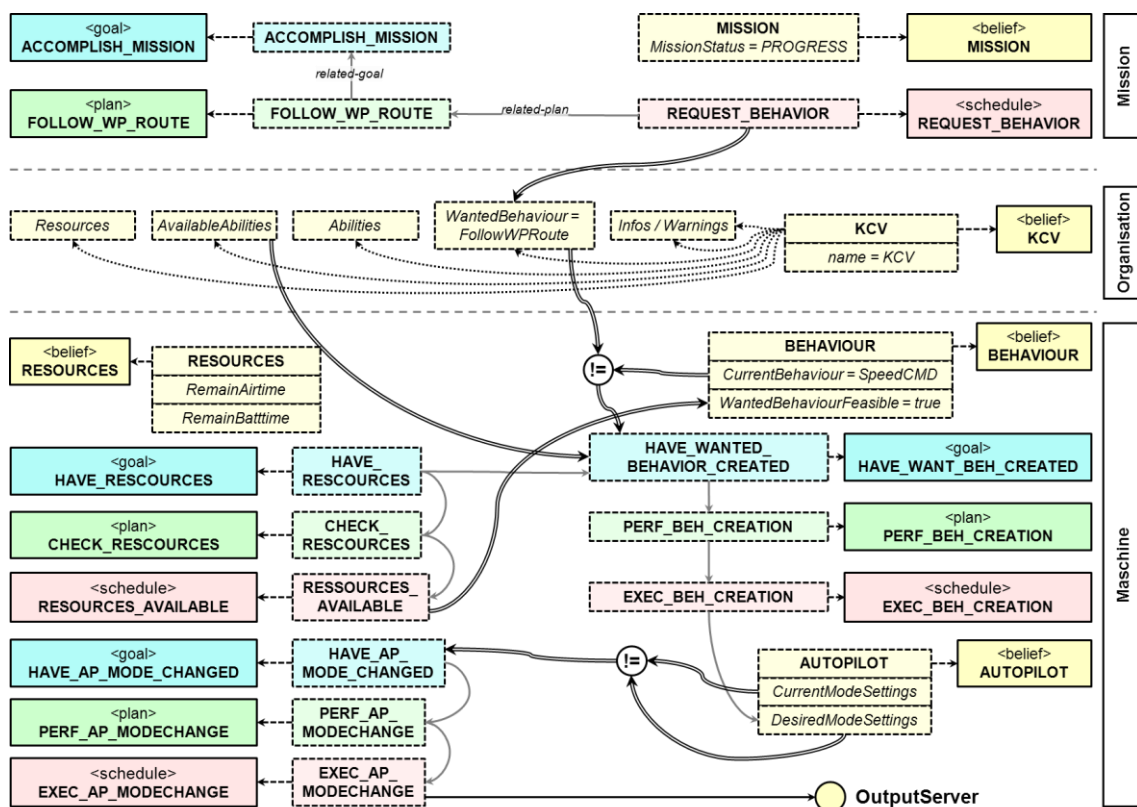


Abbildung 4-17. Anfordern von gewünschtem Verhalten und die nachfolgende Prüfung der Ressourcen und die Konfiguration der Automation

Abbildung 4-17 zeigt im oberen Bereich ausgewählte missionspezifische Wissensmodelle, im unteren Bereich einige maschinenspezifische Wissensmodelle und die dazwischen liegende Organisationsschicht, repräsentiert durch das Umweltmodell der Klasse *kcv*. Die Darstellung belegt zudem die Analogie, des in Abschnitt 3.3 in Abbildung 3-3 aufgezeigten Konzepts eines *Knowledge Configured Vehicle* (vgl.

Abbildung 3-3) und die Interaktion und den Austausch von Informationen über die Attribute der KCV-Schicht.

Nach dem Treffen der Entscheidung – durch das missionsspezifische Wissen – für das Anfordern eines *gewünschten Verhaltens*, beschreibt die Instanz der Klasse **REQUEST_BEHAVIOR** (Anweisungsmodell) das Attribut *WantedBehaviour* (Instanz der Klasse **KCV**) mit dem der angeforderten Fähigkeit entsprechenden Wert (in diesem Beispiel *FollowWPRoute* – „Fliege die Wegpunktliste ab“).

Im maschinenspezifischen Wissen ist der aktuelle Modus und die Betriebsart des Autopiloten (und FMS) durch das Attribut *CurrentModeSettings* repräsentiert, welches von der Klasse **BEHAVIOUR** gelesen und interpretiert wird. Gemäß dem Betriebszustand erfolgt die Abstraktion des aktuell erzeugten Verhaltens der Automation, dargestellt durch das eigene Attribut *CurrentBehaviour* und dessen gesetztem Wert (in diesem Beispiel *SpeedCMD* – „Fliege eine vom Operateur kommandierte Geschwindigkeit“). Unterscheiden sich die Werte der Attribute *WantedBehaviour* der Instanz der Klasse **KCV** und das Attribut *CurrentBehaviour* der Instanz der Klasse **BEHAVIOUR**, so wird zunächst geprüft, ob das *gewünschte Verhalten* auch einer *verfügbaren Fähigkeit* entspricht. Ist das der Fall, so wird eine Instanz der Klasse **HAVE_WANTED_BEH_CREATED** („Erzeuge gewünschtes Verhalten“) erzeugt.

Wird ein Verhalten angefordert, so gehört es zu den Aufgaben der ACU zu prüfen, ob für die Erzeugung dieses Verhaltens auch genügend Betriebsmittel vorhanden sind – repräsentiert durch den den instanziierten Wunsch **HAVE_RESSOURCES**. Die Prüfung, ob die vorhandenen Ressourcen für die Erzeugung des *gewünschten Verhaltens* ausreichen, übernimmt die Handlungsalternative **CHECK_RESSOURCES** – einerseits durch Berechnung der benötigten Betriebsmittel und andererseits durch deren Vergleich mit dem momentanen Versorgungszustand. Dieser wird durch die Attribute *RemainAirtime* und *RemainBatitime*, der Klasse **RESSOURCES** dargestellt, deren Werte entweder direkt im Input-Server oder mittels des Function-Servers ermittelt werden. Da beide Attribute die vorhandenen Betriebsmittel in einer Zeitskala bemessen, berechnet sich der Ressourcenbedarf für das Abfliegen der Wegpunktliste zu Beginn der Mission wie folgt:

$$\text{Flugdauer [s]} = \frac{\text{Länge des Flugwegs [m]} + \text{Abstand Fluggerät, WP1 [m]}}{\text{Fluggeschwindigkeit } \left[\frac{\text{m}}{\text{s}}\right]}$$

Formel 1. Berechnung der Zeitdauer für das Abfliegen einer Wegpunktliste zu Beginn der Mission

Ist die so ermittelte Flugdauer für das Abfliegen der Flugroute geringer als die Zeiten, für welche es noch eine sichere Ressourcenversorgung gibt, so erzeugt das Anweisungsmodell **RESSOURCES_AVAILABLE** das Attribut *WANTED_BEHAVIOUR_FEASIBLE* mit dem Wert *true* an der Instanz der Klasse **BEHAVIOUR**. Dieses stellt eine der Vorbedingungen der Handlungsalternative **PERF_BEH_GENERATION** (*Perform behaviour generation* – „Plane die Erzeugung des gewünschte Verhaltens“) dar, welche das Vorgehen für die Erzeugung des *gewünschten Verhaltens* ausarbeitet. Bietet die

Automation direkt die Betriebsarten, die zur gewollten Handlungsweise des Fluggerätes führt, so werden nachfolgend – mittels des Anweisungsmodells `EXEC_BEHAVIOUR_GENERATION` – die benötigten Automationsbetriebsarten durch Anpassung und Aktualisierung der Werte des Attributs `DesiredModeSettings` der Klasse `AUTOPILOT` konkretisiert. Verfügt im Gegenzug die Automation nicht über entsprechende Betriebsarten, kann durch zusätzliches a-priori Wissen (vor allem in der Klasse `PER_BEH_GENERATION`, für die Planung des Vorgehens), das *gewünschte Verhalten* durch Kombination von *verfügbaren Fähigkeiten* erzeugt werden (vgl. Abschnitt 3.4). Voraussetzung hierfür ist die vorherige Abstraktion dieser, aus mehreren *Fähigkeiten* zusammengesetzten Fähigkeit und deren Eintragung in der KCV-Schicht an dem entsprechenden Attribut (*Abilities*).

Für die Konfiguration der Automation sorgt die Klasse `AUTOPILOT`, mittels der beiden Attribute `CurrentModeSettings` und `DesiredModeSettings`. Unterscheiden sich die Werte der beiden Attribute, so instanziiert sich der Wunsch `HAVE_AP_MODE_CHANGED` (*Have autopilot mode changed* – „Ändere aktuelle Autopiloten/FMS Betriebsart“). Anschließend übernimmt die Instanz der Handlungsalternativenklasse `PERF_AP_MODE_CHANGE` (*Perform autopilot mode change* – „Plane die Betriebsartenumschaltung des Autopiloten“) die Ausarbeitung einer Vorgehensweise der Konfiguration des Autopiloten/FMS, da es beispielsweise aufgrund von bestimmten Moding-Eigenschaften eines Autopiloten nötig sein kann, eine bestimmte Reihenfolge beim Schalten der Betriebsarten einzuhalten.

Nach erfolgreichem Abschluss der Planung sendet abschließend die Instanz `EXEC_AP_MODE_CHANGE` (*Execute autopilot mode change* – „Führe Betriebsartenänderung des Autopiloten durch“) die entsprechenden Konfigurationskommandos, über den Output-Server, an die Softwarekomponente FCS (Flight control system – vgl. Abschnitt 5.4.5).

4.8 Zusammenfassung – Methodisch algorithmische Umsetzung

Die ACU wurde mit den in den vorhergehenden Abschnitten beschriebenen Wissensmodellen in Form einer COSA-Applikation implementiert. Die Interaktion und Anbindung der ACU an die Umwelt erfolgt über mehrere Komponenten, die für die Eingabe von System- und Zustandsdaten (Input-Server) und auch für die Ausgabe (Output-Server) von Konfigurationskommandos für das FCS zur Verfügung stehen. Numerische Berechnungen werden von einem Function-Server durchgeführt, wie etwa die Berechnung der Flugdauer für eine Wegpunktliste (vgl. Abschnitt 4.7).

Das a-priori Wissen ist die Basis für das von der ACU erzeugte Verhalten und setzt sich aus drei hauptsächlichen Wissenspaketen zusammen, nämlich:

- **Missionsspezifisches Wissen:** Enthält Wissen zum Verständnis und zur Durchführung der Mission. Übernimmt das Management der vom Fluggerät angebotenen

Fähigkeiten, um den gegebenen Auftrag zu erfüllen.

- **KCV-Schicht:** Strukturiert und organisiert den Austausch von Situationswissen zwischen dem missions-spezifischen und dem maschinenspezifischen Wissen.
- **Maschinenspezifisches Wissen:** Enthält Wissen über die Maschine und deren Automation und Nutzlast. Daraus werden die Fähigkeiten des UAVs abstrahiert und dem MisW über die KCV-Schicht vermittelt. Neben der Überwachung der Maschine übernimmt das MasW auch die Konfiguration und Bedienung der Automation und Missionssensorik.

	Paket(e)	Anzahl der Klassen					Anzahl CPL Regeln	
		Umweltmodelle	Wünsche	Handlungsalternativen	Anweisungsmodelle	gesamt		
MisW	Missionsmanagement	5	4	9	3	21	49	49
KCV	KCV-Schicht	1	-	-	-	1	6	6
MasW	Überwachung	20	3	2	-	25	116	185
	Ressourcenmonitoring	1	1	4	1	7	17	
	Abstraktion	1	2	2	2	7	23	
	Konfiguration der Automation	2	2	9	3	16	29	
	gesamt	30	12	26	9	77	240	

Tabelle 2. Überblick über den Umfang der Pakete und die Anzahl der Wissensmodelle des KCV-Funktionsprototypen

Tabelle 2 gibt einen Überblick über die Wissenspakete und die Anzahl der verwendeten Klassen, unterteilt in die vier Namensräume (Umweltmodell, Wünsche, Handlungsalternativen und Anweisungsmodell). Diese Übersicht verdeutlicht, dass das Hauptaugenmerk bei der Implementierung auf dem maschinenspezifischen Wissen und vor allem auf den Umweltmodellen liegt, welche für die Überwachung und die Bedienung des Fluggerätes benötigt werden. Dieses Wissenspaket macht mehr als drei Viertel der implementierten Regeln aus, wohingegen auf das missionspezifische Wissen und das Organisationswissen der KCV-Schicht nur ein geringer Anteil der Produktionen entfällt.

Tabelle 3 zeigt eine Übersicht über die Anzahl der CPL-Regeln, sortiert nach den vier möglichen Namensräumen. Daran wird deutlich, dass die Umweltmodelle den relativ größten Anteil an Produktionen tragen und für den Aufbau eines Situations- und Systemverständnisses im Sinne der kognitiven Verarbeitung sorgen. In diesem Zusammenhang weist [Meitinger, 2008] in Ihrer Arbeit darauf hin, dass diese Verteilung der Produktionen charakteristisch für wissensbasiertes Verhalten ist.

	Umweltmodelle	Wünsche	Handlungsalternativen	Anweisungsmodelle	gesamt
Anzahl der CPL Regeln	173	25	29	13	240

Tabelle 3. Überblick über die Anzahl der CPL Regeln, gruppiert nach den CPL-Namensräumen

Im folgenden Kapitel wird die systemtechnische Umsetzung der Demonstratorumgebung beschrieben, in welche die entwickelte ACU, als ein zentrales Element für die Evaluation des Konzepts eines Knowledge Configured Vehicle, integriert wurde.

5 Systemtechnische Umsetzung und Integration

Um eine kognitive Einheit in Form einer *ACU*, wie in Abschnitt 2.4.2 beschrieben, zu befähigen mittels klar getrenntem Missions- und Maschinenwissen eine Mission bzw. einen Auftrag durchzuführen, wurde in Kapitel 3 das Konzept eines *Knowledge Configured Vehicle* vorgestellt. Dessen Umsetzung, die in CPL entwickelten a-priori Wissensmodelle sowie deren Interaktion und Zusammenwirken wurden im vorigen Kapitel detailliert. Thema dieses Kapitels ist nun die Beschreibung der Experimentalumgebung sowie die Integration der *ACU* in das UAV-System, welches für die spätere Demonstration der Funktionalitäten der *KCV-ACU* aufgebaut wurde.

Abschnitt 5.1 definiert anfangs einige Anforderungen an die Demonstratorumgebung – bestehend aus einer Bodenkontrollstation und einem UAV – gefolgt von deren Konzept (Abschnitt 5.2) und der entwickelten Simulationsumgebung (Abschnitt 5.3). Der UAV-Demonstrator und dessen Systemtechnik als auch die entsprechenden Softwarekomponenten für die automatische Flugstabilisierung sind Gegenstand von Abschnitt 5.4. Für die Führung des UAVs und die Überwachung der Subsysteme durch einen menschlichen Operateur wurde eine Kontrollstation aufgebaut, deren Hardwareelemente und Softwaremodule in Abschnitt 5.5 näher erläutert sind. Das Kapitel schließt mit der Beschreibung eines proprietären Protokolls, welches im Hinblick auf ein sicheres Kommunikationsmanagement, auf Basis einer seriellen Datenfunkverbindung zwischen der Kontrollstation und dem UAV (Abschnitt 5.6), umgesetzt wurde.

5.1 Anforderungen

Ziel des Aufbaus der Experimentalumgebung ist die Demonstration der entwickelten *KCV-ACU* mit einem real fliegenden System, im Rahmen der Durchführung einer einfachen Aufklärungs-Mission. Hierfür müssen einige Anforderungen erfüllt werden, die sich einerseits von dem Betrieb eines Mini-UAVs in einem universitären Umfeld und andererseits von der Bereitstellung einer geeigneten System- und Software-Infrastruktur, für die Aufnahme und den Betrieb der kognitiven Einheit, ableiten. Des Weiteren können die Anforderungen an die entsprechenden Hardware-Systeme nach Abbildung 2-1 auf die Bereiche *Kontrollsegment* (KS – Kontrollstation) und *Fliegendes Segment* (UAV) aufgeteilt werden. Die wichtigsten Anforderungen sind:

- **Allgemeine Anforderungen:**

- **Sicherer Betrieb:** Bei Testflügen muss ein möglichst gefahrloser Betrieb des UAVs gewährleistet werden. Im Fall von technischen Problemen oder ungewünschtem Verhalten aufgrund von Softwarefehlern soll es zu jedem Zeitpunkt möglich sein, die manuelle Kontrolle über das Fluggerät zu übernehmen.

- Rechtliche Rahmenbedingungen: Für den Betrieb eines UAVs im Rahmen der Möglichkeiten einer Universität soll das Fluggerät keine Zulassung benötigen und somit unterhalb der gesetzlichen Grenze von 25kg (max. Abfluggewicht für Modellflugzeuge) bleiben.
- **Kontrollsegment:**
 - Kontrollstation: Für die Führung und vor allem die Überwachung des Fluggerätes wird eine Kontrollstation benötigt, die eine entsprechende Rechner- und Netzwerkinfrastruktur sowie Monitore und Eingabegeräte (z.B. Joystick) für einen Operateur-Arbeitsplatzes bietet. Zudem muss die Stromversorgung dieser Geräte durch Bordmittel sichergestellt werden, da Flugversuche und Testflüge meist auf Flugplätzen durchgeführt werden, an denen keine externe Stromversorgung verfügbar ist.
 - Kommunikation: Für die Übermittlung von Telemetrie- und Kommandoinformationen zwischen der Kontrollstation und dem UAV sind Kommunikationseinrichtungen erforderlich.
- **Fliegendes Segment:**
 - Fluggerät: Das Fluggerät muss einerseits über entsprechende Flugleistungen, als auch Vorrat an Ressourcen verfügen, die es ermöglichen die Avionikkomponenten für eine der Mission angemessenen Flugdauer in der Luft zu tragen.
 - Automation: Um einen von der ACU geführten Flug zu ermöglichen, sind konventionelle Automationsfunktionen voraussetzend (FCS – Autopiloten- und FMS-Funktionalitäten).
 - Sensoren: Für eine automatisierte Fluglagestabilisierung und Bahnführung sind verschiedene Arten von Flugzustandserfassungssensoren erforderlich.
 - Rechenkapazität: Für die Aufnahme von Softwarekomponenten, wie zum Beispiel die ACU und das FCS, werden Computer benötigt. Diese müssen genügend Rechenleistung für die Ausführung und Berechnung der Flugregelungsalgorithmen und die Ausführung der COSA-Applikation zur Verfügung stellen und dabei jedoch einen geringen Leistungsbedarf aufweisen.

Im folgenden Abschnitt wird das Konzept der Demonstratorumgebung präsentiert, das die genannten Anforderungen berücksichtigt und zudem einen ersten Einblick in die später umgesetzte Verschaltung der Komponenten erlaubt.

5.2 Demonstratorumgebungskonzept

Abbildung 5-1 zeigt den konzeptionellen Aufbau der Demonstratorumgebung mit drei Hauptsegmenten, nämlich das *Fliegende Segment*, das *Kontrollsegment* und das *Sicherheitssegment*. Letzteres ist für den eigentlichen Betrieb und die Funktionsdemonstration nicht notwendig (vgl. gestrichelte Verbindung zwischen dem FCS und dem UAV in Abbildung 5-1) und dient ausschließlich der Sicherstellung eines risikoarmen Flugbetriebs. Hierfür ist ein Sicherheitspilot vorgesehen, der – durch Halten einer permanenten Sichtverbindung zu dem Fluggerät – bei außerplanmäßigem Verhalten des UAVs oder bei technischen Problemen die manuelle Kontrolle über das Fluggerät zu jedem Zeitpunkt übernehmen kann. Der Sicherheitspilot muss demnach bei Flügen mit größerer Entfernung zu der BKS dem Fluggerät folgen, um eine gute visuelle Lageerkennung sicherzustellen. Vor diesem Hintergrund soll, durch den Einsatz einer funkbasierten Interkommunikationsanlage, die verbale Kommunikation zwischen dem Sicherheitspiloten und dem Operateur in der BKS ermöglicht werden.

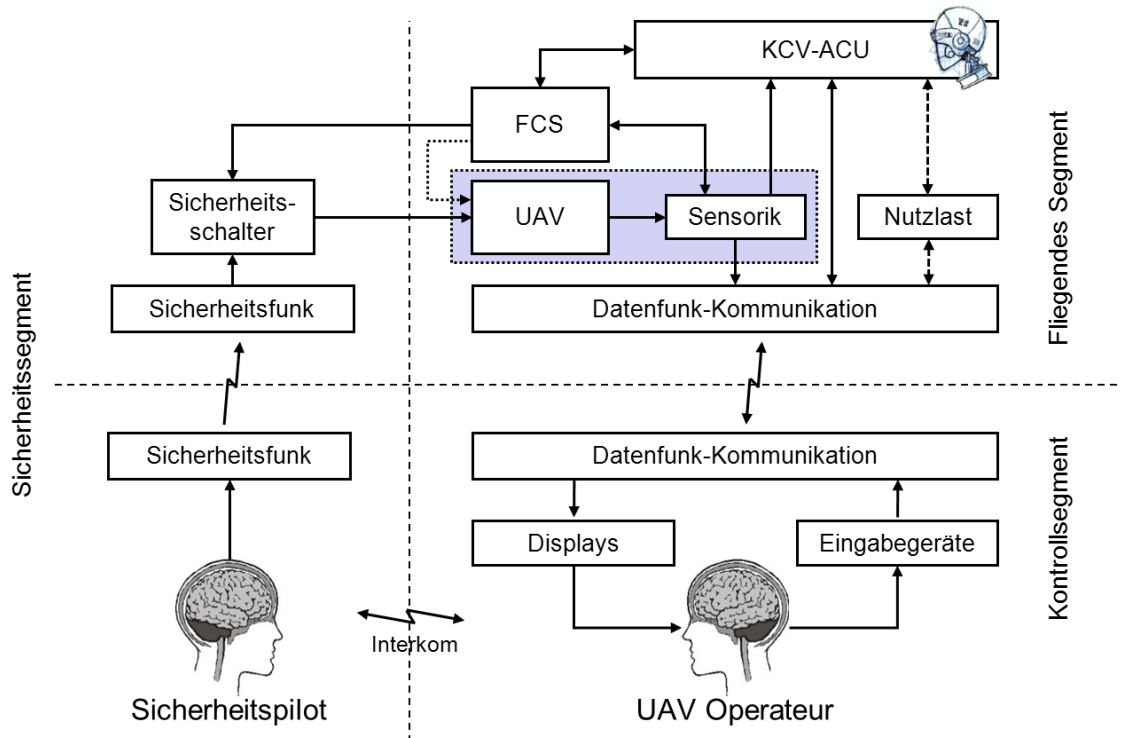


Abbildung 5-1. Demonstratorumgebungskonzept

Für die Interaktion mit dem Fluggerät steht dem UAV Operateur eine Funkverbindung zur Verfügung, über die sowohl Steuer- und Automationsanweisungen von Eingabegeräten abgesetzt als auch Telemetriedaten (z.B. Position, Fluglage, Zustand des FCS, Statusmeldung und Warnungen der KCV-ACU) des Luftfahrzeugs empfangen werden können. Die empfangenen Nachrichten werden nachfolgend aufbereitet und auf Anzeigegeräten (Displays) in der Bodenkontrollstation zur Darstellung gebracht.

Teil des fliegenden Segments ist das Fluggerät selbst, die Kommunikationseinrichtung(en), etwaige Nutzlastkomponenten und die Flugzustandssensorik. Deren

Daten werden zum einen an die konventionelle Automation (FCS) und zum anderen an die kognitive Automation (KCV-ACU in Form einer COSA-Applikation) weitergereicht. Die konventionelle Automation des UAVs, bestehend aus einem FCS, welches gängige Autopiloten- und FMS-Funktionalitäten bietet, übernimmt die automatische Stabilisierung und Bahnführung des Fluggerätes. Die errechneten Stellkommandos für die Aktuatorik des Fluggerätes werden folgend in einen Sicherheitsschalter geführt. Dessen Stellung – festgelegt vom Sicherheitspiloten – determiniert, welche Steuersignale (entweder die manuellen Steuereingaben, oder die Stellsignale des FCS) an die Stellmotoren des UAVs weitergeleitet werden.

Auf Basis der aktuellen Flugzustandsdaten kann beispielsweise die ACU erkennen, ob der aktuell anzufliegende Wegpunkt erreicht ist, dies dem Operator über die Datenfunkverbindung mitteilen und gegebenenfalls – entsprechend der aktuellen Aufgabe bzw. Mission – die Konfiguration bzw. Bedienung von Nutzlastkomponenten vornehmen oder eine Rekonfiguration bzw. Umschaltung der aktuellen Betriebsmodi des FCS bewirken.

Für die Umsetzung des vorgestellten Demonstratorkonzepts wird im folgenden Abschnitt die Simulationsumgebung (blauer Kasten mit gestrichelter Umrandung in Abbildung 5-1) beschrieben, die für die Entwicklung und Erprobung der Softwarekomponenten (FCS, ACU) aufgebaut wurde.

5.3 Simulationsumgebung

Ziel bei dem Aufbau der Simulationsumgebung war es, möglichst viele Softwaremodule und vor allem reale Hardwarekomponenten, die später auch auf dem realen Fluggerät eingesetzt werden, zu verwenden (engl.: Hardware in the loop – HIL). Hierdurch können schon während der Entwicklungsphase von der Hardware induzierte Probleme und Seiteneffekte identifiziert und beseitigt werden (vgl. Abschnitt 2.5.1).

Kern der Umgebung ist die kommerzielle Modellflugsimulation *Reflex* [simWerk Homepage], welche sich durch eine subjektiv sehr realitätsnahe Nachbildung des dynamischen Verhaltens von Flächenflugzeugen und Hubschraubern auszeichnet und zudem auch in der Lage ist, atmosphärische Bedingungen, wie beispielsweise Wind und thermische Ablösungen, zu simulieren. Der Zugriff auf die internen Zustandsdaten des simulierten Fluggerätes erfolgt über eine vom Hersteller definierte Software-Schnittstelle in Form einer dynamischen Softwarebibliothek (engl. Dynamic linked library – DLL). Diese wird beim Programmstart eingebunden und führt während des Simulationszyklus vordefinierte, proprietär implementierte Funktionen für die Übertragung der aktuellen Simulationsdaten aus. Der Aufruf dieser Funktion ist jedoch an die Anzeige der virtuellen Außensicht der Simulation gekoppelt und ist demnach abhängig von der Bilderwiederholrate. Daraus resultiert eine variierende Frequenz der Flugzustandsdaten, welche durch ein Sensormodell innerhalb der DLL an die gewünschte Frequenz von 50Hz angepasst werden (vgl. Abbildung 5-2). Darüber hinaus übernimmt das Sensormodell die Transformation der nicht-luftfahrtgenormten Koordinatensysteme (KS) der Simulation in luftfahrtstandardisierte KS nach LN 9300

(LN – Luftfahrtnorm) und bietet ferner die Möglichkeit, die simulierten Sensordaten künstlich zu verschlechtern (z.B. Sensorrauschen, Sensorbias, Einbaufehler, Achsverschiebungen, Orthogonalitätsfehler, Quantisierungsfehler, Latenzzeiten, ...) [Höcht, 2007].

Die von dem FCS erzeugten Steuergrößen müssen aufgrund des Primärzwecks der *Reflex* Simulation in ein modellbautypisches Signal gewandelt werden, das von gängigen Fernsteueranlagen bereitgestellt wird. Dieses sogenannte PPM- (Puls-Puls-Modulation) Signal wird von einem eigens entwickelten Elektronikboard mit einem 8bit Mikrocontroller erzeugt und kodiert die über eine serielle Punkt-zu-Punkt Verbindungsschnittstelle (RS-232) empfangenen Steuergrößen des FCS und gibt diese an die Simulation weiter (vgl. Abbildung 5-2, Logikpegel: 0-5V, TTL – Transistor-Transistor-Logik).

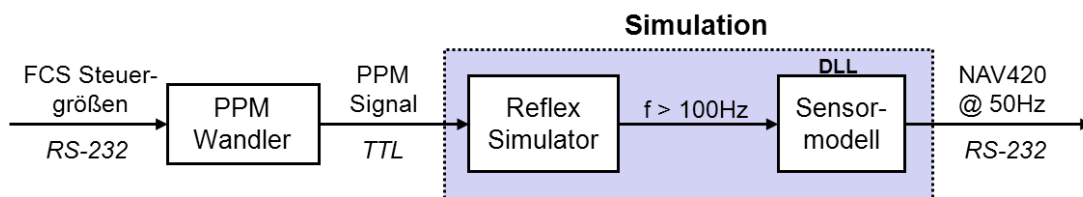


Abbildung 5-2. Simulation Flugdynamik und der Sensordaten des UAVs

Die Ausgabe der simulierten Sensordaten erfolgt ebenfalls über serielle Verbindungen (RS-232) – mit der gewünschten Frequenz von 50Hz – in einem Protokollformat, welches zum einen dem Primärsensor der Flugzustandserfassung (CrossBow NAV 420 IMU, vgl. Abschnitt 5.4.2.3) und zum anderen einem weiteren Sensorboard (Eigenentwicklung, vgl. Abschnitt 5.4.1.1) entspricht. Somit können die, normalerweise von der realen Sensorik erfassten, Flugzustandsdaten des Fluggerätes simuliert werden, ohne dass es einer Anpassung der entsprechenden Treiber-Prozesse (vgl. Abschnitt 4.5.1), welche den Datenstrom der Sensoren dekodieren, bedarf. Im vorliegenden Fall der Umsetzung können damit die Software-Treiber nicht mehr zwischen simulierten und realen Sensordaten unterscheiden.

Das in der Simulation verwendete Simulationsmodell wurde anhand zweierlei Gesichtspunkte ausgewählt, nämlich

- **Allgemeine Eigenschaften:** Das Simulationsmodell soll, im Hinblick auf Gewicht, Größe, Antriebssystem und den mechanischen Aufbau des Rotorkopfs, über ähnliche physikalische Eigenschaften wie der reale Flugdemonstrator verfügen.
- **Flugeigenschaften:** Das Simulationsmodell soll ähnliche Flugeigenschaften aufweisen wie das reale Flugmodell. Der verwendete Flugsimulator *Reflex* ermöglicht die Anpassung von Parametern zur Beeinflussung der Flugeigenschaften des Simulationsmodells, wie zum Beispiel die Roll- und Nickstabilität, Aufbäumneigung und die Reaktion des Flugmodells auf zyklische und kollektive Steuereingaben. Hierfür wurden von dem Sicherheitspiloten

Testflüge in der Simulation und mit dem realen UAV-Demonstrator durchgeführt, um eine subjektive Angleichung der Flugeigenschaften zu erreichen.

Die aufgebaute Simulationsumgebung ermöglicht damit die Bereitstellung einer zumindest subjektiv ähnlichen Flugdynamik, des im folgenden Abschnitt beschriebenen UAV-Demonstrators BIG I, ohne jedoch eine genaue und aufwändige Systemidentifikation durchzuführen. Hauptanwendungsgebiete der Simulationsumgebung waren die Entwicklung eines empirischen Vorgehens zum Einstellen (engl.: *Tuning*) der regel- und streckenspezifischen Parameter des FCS (vgl. Abschnitt 5.4.5) sowie die Entwicklung und Evaluation der KCV-ACU und der dafür erforderlichen Wissensmodelle.

5.4 UAV – Demonstrator BIG I

Der UAV-Demonstrator BIG I (vgl. Abbildung 5-3) basiert auf einem Modellhubschrauber, der nach eigenen Vorgaben von der Firma XXL Modelhelikopter [XXL Modelhelicopter Homepage] konstruiert und gefertigt wurde. Trotz der höheren Systemkomplexität gegenüber einem Flächenflugzeug fiel die Wahl auf einen Drehflügler, um von den Vorteilen des senkrechten Startens, Landens sowie die Fähigkeit in der Luft zu schweben, zu profitieren.



Abbildung 5-3. Der UAV Demonstrator BIG I im Flug

Das Modell verfügt über einen Rotordurchmesser von 2.55m und einen mechanisch stabilisierten Zweiblatt-Rotorkopf. Die Taumelscheibe wird von vier, um 90° versetzten Aktuatoren (Servos) durch Push-Pull-Anlenkungen bewegt, um einerseits die Steuergenauigkeit und Betriebssicherheit zu erhöhen und andererseits Problemen wie

Getriebeispiel der Aktuatorik entgegenzuwirken. Die Steuerkommandos (Kollektiv, Rollen, Nicken) werden elektronisch gemischt (engl.: Cyclic Collective Pitch Mixing – CCPM) und die Stellsignale nachfolgend an die einzelnen Stellmotoren verteilt.

Das Fluggerät wird von einer Zweiwellenturbine (SPT5-H) der Firma Jetcat angetrieben, die mit einer maximalen Leistung von 8kW einen sicheren Betrieb mit genügend Leistungsreserven erlaubt. Eine ECU (engl.: Engine control Unit – Turbinensteuergerät) übernimmt die Steuerung (Kerosinstart, Kraftstoffeinspritzung) und Überwachung der Turbine (Drehzahlen, Temperaturen) sowie die Drehzahlregelung des Hauptrotors. Zwei Tanks mit je drei Litern Fassungsvermögen erlauben Flugzeiten von rund 25 – 30 Minuten.

Tabelle 3 gibt einen Überblick über die Gewichtszusammensetzung des UAV-Demonstrators BIG I.

Komponente	Gewicht
Fluggerät mit RC Komponenten	~ 14 kg
Treibstoff (6l Kerosin)	~ 5 kg
Avionikkomponenten & Nutzlast	~ 6 kg
	~ 25 kg




Tabelle 4. Übersicht über die Anteile des Systemgewichts des UAV-Demonstrators BIG I



Das Abfluggewicht des betankten Demonstrators liegt mit den notwendigen RC-Komponenten (engl.: Radio controlled – Empfangsanlage, Stromversorgung und Aktuatoren für den manuell gesteuerten Flug) bei rund 19kg und bietet somit genug Freiheiten für die Integration der Avionik und Nutzlastkomponenten. Abhängig von deren Gewicht könnte zudem die Flugzeit durch eine Erhöhung der Treibstoffkapazität bis zur maximalen Gewichtsgrenze von 25kg gesteigert werden.

Im folgenden Abschnitt werden der Aufbau und die Umsetzung der Avionik beschrieben, sowie die verwendeten Hardwarekomponenten kurz erläutert.

5.4.1 Hard- und Softwareumsetzung der Avionik

Abbildung 5-4 zeigt die Umsetzung des Avionik- und Softwarekonzepts für den UAV-Demonstrator BIG I. Die einzelnen Elemente sind durch entsprechende Farben und Formen codiert, mit folgenden Bedeutungen:

- Kästen (umrandet mit gepunkteten Linien, ) bezeichnen Hardware-schnittstellen für den Datenaustausch mit angeschlossenen Geräten (farbige Codierung: blau).
- Kästen (umrandet mit durchgezogenen Linien,  bzw. ) stellen Hardwarekomponenten (blau) und Softwareprozesse (grün) dar. Grau hinterlegte Kästen stellen Rechner dar.

- Kreise (mit durchgezogener Umrandung, ● bzw. ●) repräsentieren Schnittstellen, die entweder mit einer Hardwareschnittstelle des entsprechenden Rechners verbunden sind (blau) oder mit anderen Softwareelementen interagieren (grün).
- Folgende Symbole wie  (Datakey – DK) oder  (Queue – Q) stellen Interprozesskommunikationselemente dar, die einen Austausch von Daten zwischen Softwareprozessen erlauben. Ein Datakey ermöglicht die Verteilung der Daten eines Prozesses an viele Empfangsprozesse (ein Prozess schreibt, mehrere Prozesse lesen). Im Gegenzug beschreiben mehrere Prozesse eine Queue, welche von einem Prozess nach dem FIFO (engl.: First in first out) Prinzip ausgelesen wird.

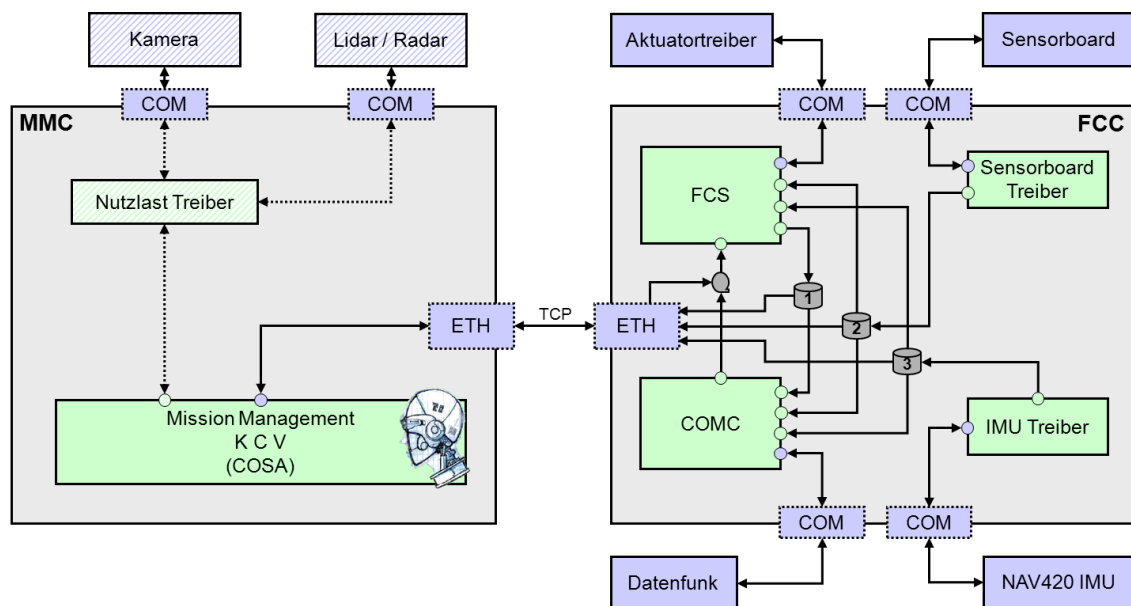


Abbildung 5-4. Avionikkonzept des UAV-Demonstrators BIG 1

Die folgenden Abschnitte beschreiben und erläutern den Aufbau und die Verschaltung der verwendeten Hardwarekomponenten auf funktionaler Ebene (Abschnitt 5.4.1.1) als auch die Softwareprozesse, welche für die automatisierte Stabilisierung des Fluggerätes durch das Flugkontrollsystem und die Flugführung durch die KCV-ACU benötigt werden. Zudem werden die Funktion und die Interaktion der einzelnen Softwarekomponenten beschrieben und abschließend die verwendeten Betriebssysteme für die Bordrechner näher spezifiziert (vgl. Abschnitt 5.4.1.2).

5.4.1.1 Hardwareaufbau

Die Umsetzung sieht zwei dedizierte Rechner vor, einen für die Aufgabe der Flugregelung (engl.: Flight control computer – FCC) und einen für die Aufgabe des Missions- und Vehikel-Managements (engl.: Mission management computer – MMC), im Rahmen des vorgestellten und implementierten Konzepts eines *Knowledge Configured Vehicle*, in Form einer künstlichen kognitiven Einheit.

An dem FCC wird, gemäß den Anforderungen aus Abschnitt 5.1, die erforderliche Sensorik für die Flugzustandserfassung angeschlossen, die aus dem Primärsensor, einer

Crossbow NAV420 IMU (vgl. Abschnitt 5.4.2.3) und einem zusätzlichen Sensorboard besteht. Das Sensorboard stellt zusätzliche Messwerte bereit, die zum einen für die Regelung des Fluggerätes und zum anderen für die Überwachung des Systemzustands der Avionik und des Fluggerätes benötigt werden. Diese Messwerte umfassen die aktuelle Spannung der Avionikstromversorgung (vgl. Abschnitt 5.4.2.1), den aktuellen Stromverbrauch der Avionikkomponenten, die Höhe über Grund (Erfassung mittels Ultraschallentfernungsmessern, verfügbar jedoch nur im bodennahen Bereich bis drei Meter), die barometrische Höhe und Umgebungstemperatur sowie den Zustand der beiden Bodenkontaktschalter, die an den Kufen des UAV-Demonstrators angebracht sind. Eine genaue funktionale Beschreibung des Sensorboards und dessen Aufbau ist in [Höse, 2006] [Höse et al., 2007] dargestellt. Des Weiteren ist eine Datenfunkeinrichtung (vgl. Abschnitt 5.4.2.4) mit dem FCC verbunden, welche die Übertragung von Telemetriedaten an die Kontrollstation (vgl. Abschnitt 5.5) und den Empfang von Kommandoinformationen von der Kontrollstation erlaubt. Für die Ansteuerung der Aktuatoren des UAV-Demonstrators ist letztlich noch ein Konverter notwendig, der die FCS-Steuergrößen in entsprechende Stellsignale umwandelt (vgl. Abschnitt 5.4.4). Alle genannten Geräte verfügen über serielle RS-232-kompatible Schnittstellen (Bezeichnung COM in Abbildung 5-4), die ebenfalls von dem FCC zur Verfügung gestellt werden müssen.

Der MMC verfügt derzeit noch über keine extern angeschlossenen Geräte bzw. Sensoren. Hierfür wurden jedoch bereits entsprechende Vorbereitungen für deren Konfiguration und Ansteuerung mittels seriellen Schnittstellen getroffen. Teil der Missionssensorik des UAV-Demonstrators BIG I ist eine bereits integrierte, schwenk- und neigbare E/O-Kamera (vgl. Abschnitt 5.4.3), die jedoch während der Testflüge (vgl. Kapitel 6) nicht durch die Missionsmanagementfunktionen bedient und angesteuert wurde. Die Kommunikation zwischen den beiden Avionikrechnern wurde durch eine Netzwerkverbindung realisiert.

Der folgende Abschnitt beschreibt und erläutert die verwendeten Betriebssysteme sowie die wichtigsten Softwareprozesse – die für die automatisierte Stabilisierung und Flugführung des UAV-Demonstrators benötigt werden – sowie deren Kommunikation und Interaktion.

5.4.1.2 Softwareumsetzung

Die Softwareumgebung beider Avionikrechner basiert auf Linux-Betriebssystemen, welche einerseits anhand der verfügbaren Hardware und den Anforderungen der eingesetzten Applikationssoftware und andererseits im Hinblick auf eine gute Portabilität zwischen den Entwicklungs- und dem entsprechenden Zielrechner ausgewählt wurden. Seitens des MMC entspricht die Hauptanwendung der COSA-Applikation – die *KCV-ACU*. Auf Seiten des FCC sind die Softwareprozesse zu nennen, welche die automatisierte Stabilisierung und Lenkung des Fluggerätes realisieren.

Das Betriebssystem des MMC basiert auf einer SUSE Distribution der Version 10.2 mit Standard-Kernel (Version 2.6.18) und wurde, der Einfachheit halber, von einem Entwicklungsrechner auf den SSD-Datenspeicher des MMC (vgl. Abschnitt 5.4.2.2) gespiegelt. Dadurch müssen für die Entwicklung der ACU und dem späteren Einsatz an Bord des UAVs keine Unterschiede zwischen der Entwicklungs- und Zielplattform

gemacht werden. Die auf dem MMC ausgeführte COSA-Applikation verfügt über keine feste Taktung, da die Verarbeitung der Eingangsdaten durch das a-priori Wissen, das Treffen von Entscheidungen im Hinblick auf die Missionserfüllung sowie die Konfiguration der Automation und die Überwachung der UAV-Subsysteme so schnell wie möglich erfolgt. Somit hängt die Arbeitsfrequenz der ACU von der Menge des a-priori Wissens, der Menge und der Änderungen der Eingangsdaten sowie der verfügbaren Rechenleistung des MMC ab. Dies führt zu einer sehr hohen Auslastung eines Prozessor-Kerns. In Verbindung mit dem verwendeten, leistungsfähigen *Intel® Core 2 Duo* Prozessor ist das Resultat ein sehr hoher Bedarf an elektrischer Leistung des MMC von circa 38W. Um diesen hohen Energieverbrauch zu senken, wurde die Aktualisierungsfrequenz der Eingangsdaten für die KCV-ACU auf maximal 3Hz festgelegt und das COSA durch zusätzliche Warteschleifen innerhalb der entsprechenden Input-Server Komponente künstlich verlangsamt. Hierdurch konnte der Strombedarf des MMC um knapp ein Ampere und damit der Leistungsbedarf um circa 14W auf rund 24W gemindert werden.

Die Kommunikation der KCV-ACU mit den Prozessen des FCC erfolgt über eine TCP-basierte (engl.: Transmission control protocol – Übertragungssteuerungsprotokoll) Netzwerkverbindung, die einerseits die Übertragung der aktuellen System- und Flugzustandsdaten sowie des aktuellen Betriebsmodus des FCS ermöglicht. Andererseits werden über diese Schnittstelle Umschaltungen der FCS-Betriebsarten, auf Basis der aktuellen Mission bzw. Aufgabe und der Interpretation des aktuellen Flug- und Systemzustands, durch die ACU initiiert.

Die hauptsächlich auf dem FCC ausgeführten Softwareprozesse umfassen den Flugregler (FCS), entsprechende Treiber für die Dekodierung der Daten der verwendeten IMU und eines zusätzlichen Sensorboards sowie den COMC-Prozess (engl.: Communication management center – COMC – Datenlinkmanagement, vgl. Abschnitt 5.6). Die Verteilung der Daten zwischen den einzelnen Softwareprozessen erfolgt mittels Interprozesskommunikation auf Basis von gemeinsam genutzten Speicherbereichen (engl.: Shared memories), die eine hochperformante Verteilung der Daten, bei geringem Ressourcenaufwand ermöglichen. So werden die Daten, die nach dem Empfang der seriellen Daten der Sensorsysteme und durch die entsprechenden Treiber dekodiert wurden, ebenso wie der aktuelle Zustand des FCS, jeweils in einen Datakey (DK1-3, vgl. Abbildung 5-4) geschrieben. Auf Basis dieser Daten können zum einen Telemetrienachrichten durch das COMC verfasst und an die Kontrollstation verschickt werden. Zum anderen kann das FCS, je nach gewähltem Betriebsmodus (vgl. Abschnitt 5.4.5), die Steuergrößen für die Aktuatoren berechnen, die nachfolgend von dem Aktuatortreiber in Stellsignale umgesetzt werden.

Im Gegensatz zu dem MMC sind die Taktung und die zeitgenaue Ausführung der FCS-Routinen jedoch sehr wichtig und stellt entsprechende Anforderungen hinsichtlich geringer Latenz- und Reaktionszeiten an das Betriebssystem des FCC. Normalerweise werden für den Prozess der Flugregelung Echtzeitbetriebssysteme (engl.: Realtime operating system – RT-OS) eingesetzt, welche jedoch die Verwendung von Cross-Compilern und speziellen Toolchains bedingen. Um dennoch einen konventionellen und keinen RT-Kernel einsetzen zu können, wurden die Einstellungen eines Standard-

Kernels (Version 2.6.18) verändert, dieser neu kompiliert und die interne Schedulerfrequenz von 250Hz auf 1kHz erhöht. Versuche in der Simulationsumgebung sowie auf dem realen UAV-Demonstrator haben gezeigt, dass die Zeitdauer von der Dekodierung der seriellen Daten der Trägheitsnavigationseinheit durch den IMU-Treiber Prozess, die nachfolgende Verteilung der Daten über die Interprozesskommunikation (DK3) und letztlich die Berechnung der Stellgrößen durch den FCS-Prozess im Schnitt rund 3.0ms beträgt und somit deutlich unterhalb der Dauer einer Regelperiode von 20ms liegt (Taktfrequenz des FCS entspricht 50Hz, vgl. Abschnitt 5.4.5). Als Taktgeber für die Ausführung der Regelalgorithmen des FCS wurde der Datentakt der IMU verwendet, da diese die erfassten Flugzustandsdaten mit einer sehr präzisen Frequenz von 50Hz ausgibt.

Insgesamt wurden für die Kommunikation auf dem FCC fünf Queues und zehn Datakeys, zur Übertragung und Verteilung der Daten zwischen den einzelnen Softwareprozessen, verwendet. Neben den genannten Softwareanwendungen für die Dekodierung der Sensordaten (Gerätetreiber), das Kommunikationsmanagement (COMC) und die Flugregelung (FCS) werden noch einige weitere Prozesse ausgeführt, welche beispielsweise die Flugdaten mit der Systemfrequenz von 50Hz aufzeichnen (engl.: Logging).

5.4.2 Avionikkomponenten

Im Nachfolgenden wird zunächst die realisierte Stromversorgung beschrieben, welche die elektrische Versorgung der Avionikkomponenten übernimmt. Anschließend werden die einzelnen, verwendeten Hardwareelemente, nämlich die Bordrechner (FCC, MMC), die Trägheitsnavigationseinheit (IMU) und die Datenfunkmodems näher beschrieben und erläutert.

5.4.2.1 Stromversorgung

Die Stromversorgung der Avionikkomponenten ist, aufgrund der Anforderung für einen sicheren Betrieb des Fluggerätes, unabhängig und galvanisch getrennt von der Stromversorgung der RC-Komponenten, wie beispielsweise die Aktuatoren und die RC-Empfänger (vgl. Abschnitt 5.4.4). Hierdurch kann sowohl eine elektrische Beeinflussung der RC-Komponenten durch die Avionik weitestgehend ausgeschlossen als auch die Auswirkungen von aktiv sendenden Komponenten – beispielsweise den Datenfunkmodems (vgl. Abschnitt 5.4.2.4) – auf die zum Teil sehr empfindlichen RC-Empfänger gemindert werden.

Die elektrische Energie für die Avionikkomponenten wird von LiFePO₄-Akkumulatoren (engl.: Lithium Ferro Phosphate – Lithium Eisenphosphat, Nennspannung 3,3V bei 2300mAh Kapazität pro Zelle) bereitgestellt, welche im Vergleich zur aktuellen LiPo-Technologie (engl.: Lithium Polymere) über eine geringere Energiedichte verfügen, jedoch bei Tiefenentladung und Beschädigung der Zelle nicht die Gefahr bergen, einen Brand zu verursachen. Zwar wurde es in Erwägung gezogen, einen Generator für die elektrische Versorgung der Avionik zu verwenden, jedoch hätte dies einen höheren Integrationsaufwand des Stromerzeugers mit nachgeschalteter Regelelektronik und einem möglicherweise benötigten Puffer-Akku

mit entsprechend kleinerer Kapazität bedeutet. Die Folge wäre demnach ein höheres Systemgewicht im Vergleich zu der reinen Akkustromversorgung gewesen. Des Weiteren ist bei dem derzeitigen Entwicklungs- und Ausbaustand des UAV-Demonstrators der limitierende Faktor, im Hinblick auf die erreichbare maximale Flugzeit, die verfügbare Treibstoffmenge des Fluggerätes und nicht die Kapazität der Energieversorgung.

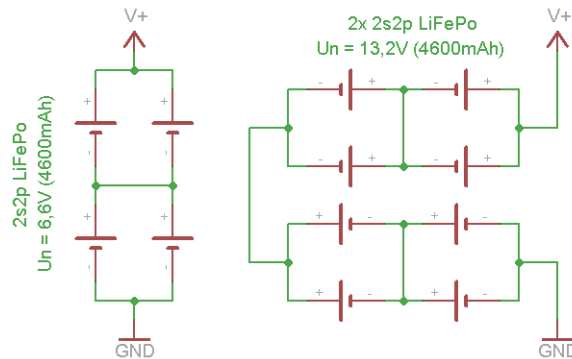


Abbildung 5-5. Aufbau eines Akkupacks (links) und die Verschaltung von zwei Packs für die Stromversorgung für die Avionikkomponenten (rechts)

Aufgrund der von den meisten, in den folgenden Abschnitten näher beschriebenen Avionikkomponenten benötigten Mindestspannung von 9V, wurde im ersten Schritt aus vier Zellen ein Akkupack konfektioniert, wovon jeweils zwei Zellen parallel geschaltet (2s2p Konfiguration, vgl. Abbildung 5-5 links) wurden. Die daraus resultierende Nennspannung von 6,6V erlaubt es, sowohl die Akkupacks für die Stromversorgung der RC-Komponenten (vgl. Abschnitt 5.4.4) als auch – wenn jeweils zwei in Serie geschaltet sind – für die Avionikkomponenten zu verwenden (vgl. Abbildung 5-5 rechts).

Für die Vorbereitung des Fluggerätes bei Flugversuchen wurde, für die Stromversorgung der Avionikkomponenten am Boden, noch zusätzlich ein Anschluss für ein Bodenstromaggregat (engl.: Ground power unit – GPU) vorgesehen. Somit können die Subsysteme hochgefahren, der Zustand der Rechner und Softwarekomponenten gestartet und überprüft werden, ohne die Energiespeicher and Bord des UAVs bereits zu belasten.

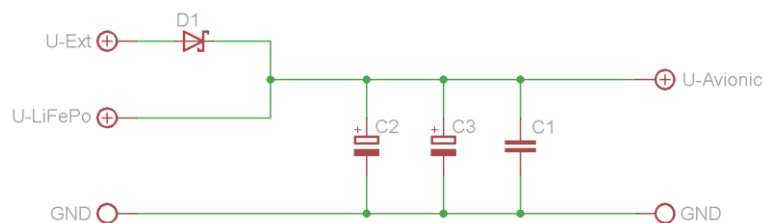


Abbildung 5-6. Entkopplung der externen Stromversorgung mittels Schottky-Diode

Die Umsetzung (vgl. Abbildung 5-6) sieht eine Entkopplung der beiden Stromquellen durch eine Schottky-Diode (D1) vor, welche seriell in die Zuleitung der externen Stromversorgung (U-Ext) geschaltet ist. Im Gegenzug werden die Avionikkakus (U-

LiFePo) direkt mit dem Stromkreis der Avionikkomponenten verbunden, um den Spannungsabfall und die daraus resultierende Verlustleistung der Diode zu vermeiden.

Folglich können durch diesen Aufbau auch die Onboard-Akkus, für die Stromversorgung der Avionik, im laufenden Betrieb gewechselt und ausgetauscht werden. Die Unterdrückung von Spannungstransienten beim An- und Abstecken der externen Stromversorgung übernehmen mehrere, parallel geschaltete Kondensatoren (C1-C3).

5.4.2.2 FCC / MMC

Die Computer – in Abbildung 5-7 (FCC links und MMC rechts) dargestellt – welche die geforderte Rechenkapazität (vgl. Abschnitt 5.1) erbringen, wurden nach den folgenden Gesichtspunkten ausgewählt:

- *Leistungs-/Energiebedarf*
- *Leistungsfähigkeit (Rechenleistung)*
- *verfügbare Peripherie*
- *Größe/Gewicht*

Aufgrund der limitierten Energieversorgung durch Akkumulatoren (vgl. Abschnitt 5.4.2.1) sollen die Rechner für die Ausführung der entsprechenden Applikationen (das FCS mit den benötigten Zusatzprozessen seitens des FCC und die KCV-ACU als COSA-Applikation seitens des MMC) genügend Rechenleistung zur Verfügung stellen, ohne jedoch dabei übermäßig viel elektrische Energie zu benötigen.

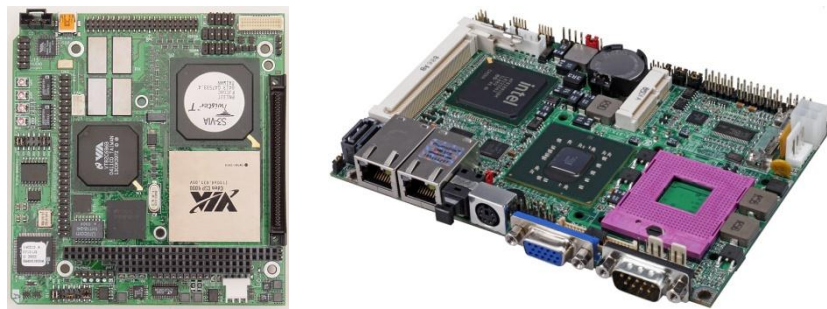


Abbildung 5-7. FCC auf Basis des Diamond Athena, Formfaktor PC-104 (links) und MMC auf Basis des Commell LS-373, Formfaktor 3,5“ (rechts)

Abbildung 5-7 zeigt auf der linken Seite den als FCC verwendeten Single-Board Computer *Athena* der Firma *Diamond*. Der Hauptprozessor, auf Basis eines passiv gekühlten Pentium III, verfügt über eine Taktfrequenz von 400MHz. Messungen haben eine max. Leistungsaufnahme von 11W (100% CPU Auslastung) und 6W (ca. 12% CPU Auslastung) im nominalen Betrieb mit den für die Flugregelung benötigten Prozesse ergeben. Zudem verfügt das *Diamond Athena* Board über vier serielle Onboard-Schnittstellen, die den direkten Anschluss der Sensor- und Datenfunk- und Aktuatortreiberkomponenten erlauben, ohne die Anzahl der benötigten Schnittstellen durch zusätzliche Hardware erweitern zu müssen. Durch den PC-104-Formfaktor sind die Abmessungen (10,6 x 11,4cm) des Rechners relativ klein, was im weiteren Verlauf

die mechanische Integration (vgl. Abschnitt 5.4.3) der Komponente erleichterte. Als Datenspeicher für den FCC (vgl. Abschnitt 5.4.2.2) wurde eine vibrationsunempfindliche Compact-Flash-Karte (Festspeicher) mit einer Kapazität von 2GB verwendet.

Auf der rechten Seite von Abbildung 5-7 ist der MMC auf Basis des Rechners *LS-373* der Firma *Commell* zu sehen. Im Vergleich zu dem FCC sind die Abmessungen (10,1 x 14,6cm) dieses Computers nur unwesentlich größer, jedoch können aufgrund der modernen Architektur des Boards beliebige *Intel® Core 2 Duo* Prozessoren eingesetzt werden. Hierdurch kann die benötigte Rechenleistung an die Anforderungen der COSA-Applikation angepasst werden (Letztlich wurde ein *Intel® Core 2 DUO P9700* mit einer Taktfrequenz von 2.8GHz eingesetzt). Die Speicherung des Betriebssystems übernimmt eine 32GB fassende, ebenfalls vibrationsunempfindliche SSD (engl.: Solid State Disc).

5.4.2.3 IMU

Für die Erfassung des aktuellen Flugzustands, wie etwa Drehraten, Lagewinkel, Geschwindigkeiten und Position, wird das Trägheitsnavigationssystem *NAV420* der Firma *Crossbow* eingesetzt (vgl. Abbildung 5-8). Nach [Crossbow Technology, 2007] verfügt die Sensoreinheit über eine Reihe von Inertialsensoren (jeweils drei Beschleunigungsmesser und Kreisel), ein Dreiachs-Magnetometer und einen GPS-Empfänger. Die Rohdaten der Sensoren werden von einem DSP (engl.: Digital signal processor – Digitaler Signalprozessor) erfasst, mittels eines FIR-Filters (engl.: Finite impulse response filter – Filter mit endlicher Impulsantwort) vorverarbeitet und nachfolgend durch einen Full-State EKF (engl.: Extended Kalman Filter – Erweiterter Kalman Filter) mit einer Frequenz von 100Hz fusioniert.



Abbildung 5-8. Das verwendete Trägheitsnavigationssystem NAV420 der Firma Crossbow

Die Sensoreinheit kann für die Ausgabe drei verschiedener Datenpaketformate mit einer einstellbaren Datenrate von 2 – 100Hz (ganzzahlige Teilung der Filterfrequenz von 100Hz) konfiguriert werden. Die verfügbaren Datenformate umfassen jeweils verschiedene Sensordaten, die der eigenen Applikation nach ausgewählt und verwendet werden können. Im Fall der vorliegenden Arbeit wurde das Datenformat *Nav Mode Packet* mit einer Ausgabefrequenz von 50Hz bestimmt, da dieses alle notwendigen Sensordaten für das FCS zur Verfügung stellt. Sie sind in Tabelle 5 mit den zugehörigen, maximalen Wertebereichen dargestellt.

Messgröße	Max. Wertebereich	Einheit
Drehraten (Körperfestes KS)	[-200, 200]	Grad/s
Euler Winkel (Kurswinkel umgerechnet auf TN)	[-180, 180]	Grad
Geschwindigkeit über Grund (NED KS)	[-256, 256]	m/s
Position (WGS84)	[-180, 180]	Grad
Höhe (WGS84)	[-100, 16284]	Meter

Tabelle 5. Fusionierte Sensordaten des gewählten Datenpakets der NAV420

5.4.2.4 Datenfunk

Für den Datenaustausch zwischen dem UAV und der BKS wurden serielle Datenfunkmodems vom Typ *Satellite 3ASd* der Firma *Satel* eingesetzt (vgl. Abbildung 5-9), welche nach den Gesichtspunkten Reichweite, Bandbreite, Datensicherheit, Leistungsbedarf und Gewicht ausgewählt wurden. Der Frequenzbereich der Modems liegt dabei im Bereich um 400MHz, was zwar einerseits eine Beschränkung der Bandbreite im Hinblick auf die maximal übertragbare Datenmenge (max. 19200 bps – bits per second) bedeutet, jedoch andererseits eine sehr robuste Funkverbindung ermöglicht (Zum Beispiel bei Verlust der direkten Sichtverbindung zwischen UAV und Kontrollstation durch Hindernisse oder Vegetation).



Abbildung 5-9. Seriellles Datenfunkmodem *Satellite 3AS(d)* der Firma *Satel*

Die maximale Sendeleistung der Funkmodule beträgt 1W (30dbm), allerdings hat sich bei Versuchen und den späteren Testflügen eine Sendeleistungen von 100mW (20dbm) als ausreichend erwiesen. Die Geräte bieten neben einer direkten Punkt-zu-Punkt Kommunikation auch die Möglichkeit, ein Funknetz mit mehreren Teilnehmern und automatischer Verteilung von Nachrichten (engl.: Message routing) aufzubauen, wie es bei der simultanen Führung und Übertragung von Telemetriedaten mehrerer UAVs nötig wäre. Zudem können einzelne Funkmodems als Repeater konfiguriert werden, wenn ein bestimmtes Szenario oder eine Mission den Einsatz einer Relaisstation erforderlich macht.

Um Daten bidirektional übertragen zu können wurden jeweils zwei Modems auf dem UAV-Demonstrator und in der Kontrollstation integriert. Der Anschluss der Modems erfolgt über ein Y-Kabel, das die Datenleitungen für ein dediziertes Empfangs- und ein Sende-Modem aufteilt. Die Verschaltung sowie die verwendeten Frequenzen der Modems können Abbildung 5-10 entnommen werden. Der Frequenzabstand beträgt

rund 10MHz. Durch eine räumliche Trennung der Antennen (Antennenabstand > 1m) kann eine gegenseitige Beeinflussung beim simultanen Senden und Empfangen weitgehend ausgeschlossen werden.

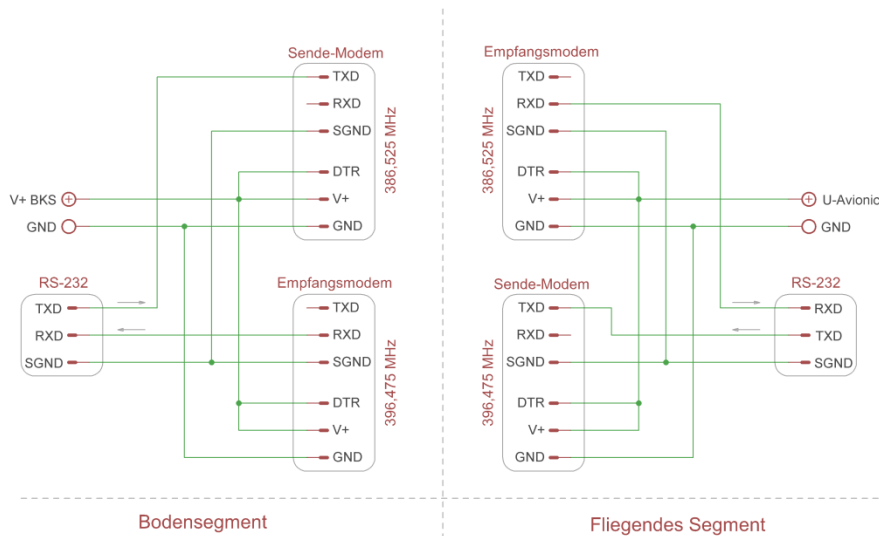


Abbildung 5-10. Verschaltung der Datenfunkmodems

Die Datenfunkmodems in Abbildung 5-10 entsprechen *DCE-Geräten* (engl.: Data communication equipment – Daten-Übertragungs-Einrichtung), während die RS-232 Schnittstellen im Boden- als auch im fliegenden Segment als *DTE-Gerät* (engl.: Data terminal Equipment – Daten-End-Einrichtung) zu interpretieren sind. Demnach stellen die TXD-Ports der Funkmodems Signaleingänge und die RXD-Ports Signalausgänge dar. Über den DTR-Port (engl.: Data Terminal Ready) kann die Betriebsbereitschaft des Gerätes festgelegt werden. Wird diese Leitung mit der Versorgungsspannung verbunden, so ist das Funkmodem eingeschaltet und ansonsten im Standby-Modus. Dies könnte für eine gesenkte Leistungsaufnahme und damit zum Einsparen von elektrischen Ressourcen genutzt werden.

5.4.3 Mechanische Integration

Abbildung 5-11 zeigt die in einem CAD-System (engl.: Computer aided design – Rechnergestützte Konstruktion) entwickelte Avionikbox für die mechanische Integration der Avionikkomponenten. Deren Bestandteile setzen sich hauptsächlich aus Carbon-faserverstärktem Kunststoff (CFK) und gefrästen Aluminiumteilen für die Aufhängung und Stabilisierung zusammen. CFK bietet neben einer sehr geringen Dichte und dem daraus resultierendem niedrigen Systemgewicht einerseits eine sehr hohe Stabilität und andererseits, aufgrund der elektrischen Leitfähigkeit der Carbonfasern, auch Schutz vor EMV Problemen. Der Aufbau sieht zwei übereinanderliegende Ebenen für die Unterbringung der Avionikkomponenten vor.

Die untere Ebene verfügt über drei Fächer für die Aufnahme der beiden Bordrechner FCC (rechts) und MMC (links) und die mittig angeordnete IMU. Sämtliche Anschlüsse der Computer sind stirnseitig angeordnet (Abbildung 5-11 zeigt nur eine schematische Darstellung der Rechner) und ermöglichen, durch große Ausschnitte an der Seitenwand, einen einfachen Zugang zu den Steckverbindern (vgl. Abbildung 5-12). Für die

Kühlung der Rechner sorgen PC Lüfter, die in der gegenüberliegenden Seitenwand verschraubt sind und durch die Ausschnitte und die Lüftungsausfräsungen (Schriftzug *UniBwM* in Abbildung 5-11) die Luft ansaugen und sie gegenüber wieder ausblasen, damit einen Hitzestau innerhalb der Rechnerkompartments verhindert wird.

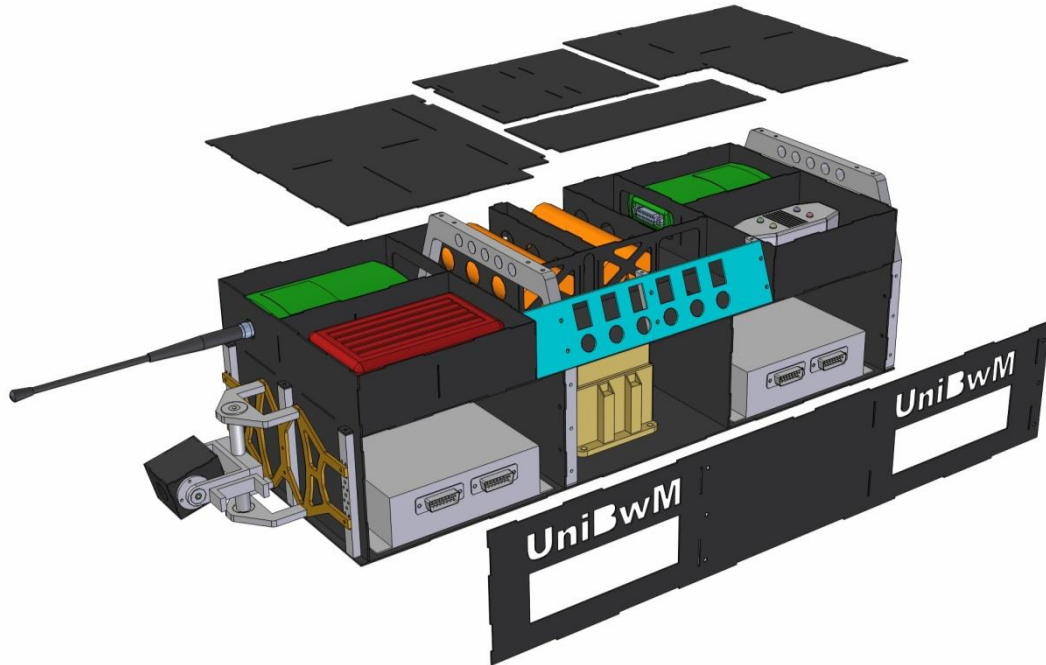


Abbildung 5-11. Mechanischer Aufbau und Platzierung der Avionikkomponenten in der Avionikbox

In der oberen Ebene sind die Stromversorgungsakkus (orange), die beiden Datenfunkmodems (grün) und der galvanischer Trenner (grau, vgl. Abschnitt 5.4.4) untergebracht. Über eine Schalttafel (blau) können die einzelnen Avionikkomponenten zu- bzw. abgeschaltet werden (Schalter sitzen in den rechteckigen Ausschnitten). Zudem verfügt jede Komponente über eine eigene Sicherung (Runde Ausschnitte in der Schalttafel), welche an den Strombedarf des jeweiligen Gerätes angepasst wurde und im Falle eine technischen Defekts bzw. Kurzschlusses einer Komponente den Betrieb der anderen Geräte, im Hinblick auf deren Stromversorgung, sicherstellt.

Für die Übertragung von Nutzlastendaten, die im Fall der vorliegenden Arbeit von einem bildgebendem E/O Sensor erzeugt werden, sorgt ein digitaler Videofunksender (rot, O.R.C.A. Videoübertragungssystem der Firma VTQ), der ebenfalls in der oberen Ebene der Avionikbox integriert wurde. Die Kamera ist an der Vorderseite der Avionikbox in einer schwenk- und neigbaren Aufhängung montiert – aktuiert durch Modellbauservos (vgl. Abbildung 5-12) – die innerhalb der mechanischen Grenzen eine Ausrichtung der Kamera erlaubt.

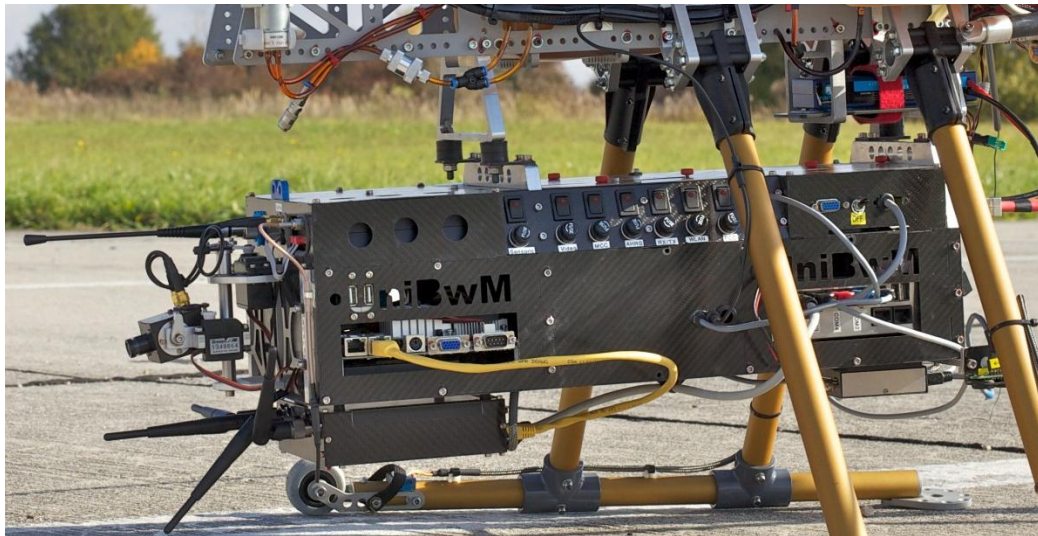


Abbildung 5-12. Die montierte Avionikbox zwischen den Landekufen des UAV-Demonstrators BIG I

Die Montage der Avionikbox an dem UAV-Demonstrator erfolgt durch Verschraubung mittels Schwingungsdämpfern an einer Halterung, die wiederum an der Grundplatte des Hubschrauberchassis befestigt ist (vgl. Abbildung 5-12). Hierdurch können, aufgrund des Systemgewichts der Avionikbox (ca. 6kg) und die daraus resultierende Trägheit, effektiv hochfrequente Schwingungen des Rotorkopfsystems gedämpft und reduziert werden. Das ist im Hinblick auf die von der IMU erfassten Drehraten- und Lageinformationen des Fluggerätes wünschenswert.

5.4.4 Sicherheitskonzept und Umsetzung

Folgend der Anforderung nach einem sicheren Betrieb des UAVs, muss der Sicherheitspilot in die Lage versetzt werden, über seine Fernsteuerung eine Umschaltung zwischen manueller Kontrolle und der Kontrolle durch das FCS vorzunehmen. Im Folgenden werden anfangs einige Grundlagen der verwendeten, bewährten Modellbautechnik erläutert und nachfolgend die Umsetzung des Sicherheitsschalters (vgl. Abschnitt 5.2) beschrieben.

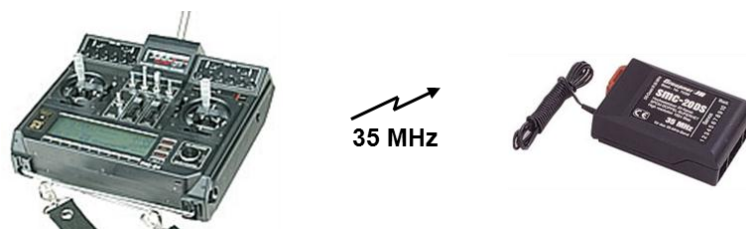


Abbildung 5-13. Übertragung der Steuerdaten eines Modellbausenders zu einem Modellbauempfänger auf dem 35MHz-Band

Die Übertragung der Steuerdaten eines Modellbausenders (Knüppelausschlag bzw. eine Schalterstellung) erfolgt auf einer Trägerfrequenz von 35 (vgl. Abbildung 5-13) bzw. 40MHz auf vordefinierten Kanälen oder bei moderneren Sendern im 2.4GHz Band. Letzteres weist zwar eine deutlich höhere Störsicherheit auf, konnte jedoch aufgrund der Verwendung eines digitalen Videofunks und WLAN (Konfiguration der

Bordrechner von der Kontrollstation aus), welche im gleichen Frequenzband agieren, nicht verwendet werden.

Ein Empfänger dekodiert die einzelnen Steuereingaben, verteilt auf entsprechende Kanäle und gibt diese, in Form von PWM-Signalen (Puls-Weiten-Modulation, vgl. Abbildung 5-14) aus, die direkt für die Ansteuerung der Aktuatoren verwendet werden können.

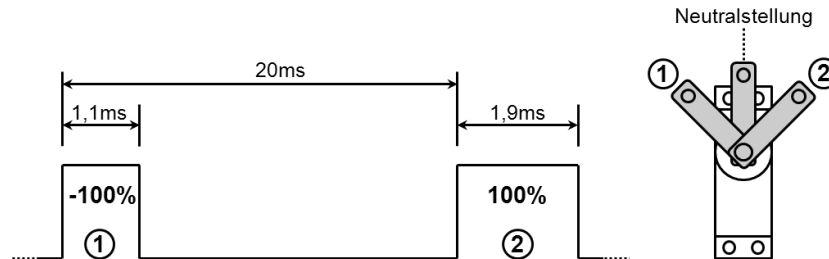


Abbildung 5-14. Pulsweiten moduliertes Signal für die Ansteuerung von Modellbauservos

Abbildung 5-14 zeigt eine nicht maßstabsgetreue Darstellung eines PWM-Signals für die Ansteuerung von modellbautypischen Servos. Über die Länge des Rechtecksignals, welches sich normalerweise im Bereich zwischen 1,1 und 1,9ms (1,5ms entspricht der Neutralstellung) bewegt (Periodendauer ca. 20ms), kann der mechanische Ausschlag des Aktuators vorgegeben werden (vgl. Abbildung 5-14 rechts).

Im Fall der vorliegenden Arbeit wurden als Sender eine *Graupner (JR) MC-24* und als Empfänger zwei *Graupner SMC 20DS* eingesetzt, die eine unidirektionale, digitale Übertragung der Steuerdaten des Sicherheitspiloten mittels PCM (Puls-Code-Modulation) bietet. Diese Übertragungstechnik erlaubt es, für jeden Kanal einen bestimmten Modus zu definieren, welches Signal ausgegeben wird, wenn die Funkverbindung abbrechen oder gestört werden sollte, nämlich:

- **Hold-Modus:** Der zuletzt, valide empfangene Steuerwert eines Kanals wird während einer Störperiode solange ausgegeben, bis neue, gültige Werte empfangen werden.
- **Failsafe-Modus:** Während der Periode des gestörten Empfangs wird ein vordefinierter Signalwert ausgegeben.

Aufgrund der Störanfälligkeit des verwendeten 35MHz Bands wurde das System *HF-Twin* der Firma *Emcotec* integriert, welches durch den Einbau eines weiteren Sendemoduls (HF-Modul) – mit einer Trägerfrequenz von 40MHz – die redundante Übertragung der Steuerdaten des Sicherheitspiloten auf zwei unterschiedlichen Frequenzen ermöglicht (vgl. Abbildung 5-15). An Bord des UAVs sind entsprechend zwei Empfänger vorhanden, deren PWM-Signale in ein *DPSI Twin* der Firma *Emcotec* geführt werden. Dieses verfügt über zwei mal neun Signaleingänge (für Haupt- und Backupempfänger). Abbildung 5-15 zeigt die schematische Umsetzung der Verschaltung der beiden Empfänger mit dem *DPSI Twin*.

Jeweils acht der Kanäle beider Empfänger werden für den Hold-Modus konfiguriert und an den Haupt- und Backup-Servoingängen des *DPSI Twin* angeschlossen. Zusätzlich wird ein neunter Kanal der Empfänger im Failsafe-Mode konfiguriert, der bei

störungsfreiem Empfang ein Signal von 100% und bei Störungen ein Signal von -100% ausgibt, und jeweils mit einem speziellen Failsafe-Eingangskanal des *DPSI Twin* verbunden ist. Durch Auswertung dieses Failsafe-Kanals ist das *DPSI Twin* in der Lage zu erkennen, ob ein Empfänger derzeit valide Werte empfängt und kann im Falle einer Störung auf den anderen Empfänger umschalten (vorausgesetzt dessen Failsafe-Signal belegt einen ungestörten Empfang) und nachfolgend dessen Signale an die Servoausgänge weitergeben (vgl. Abbildung 5-15).

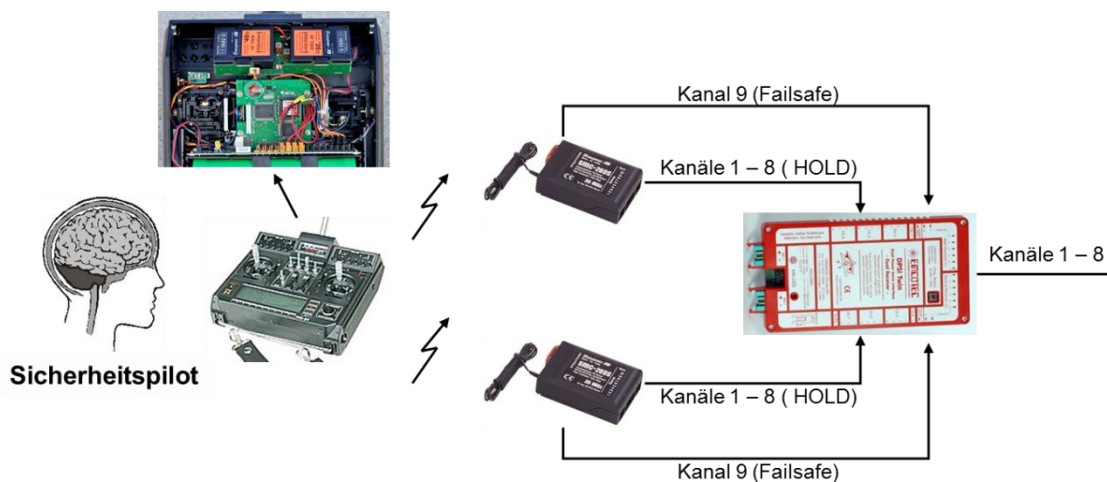


Abbildung 5-15. Einsatz und Verschaltung des HF-Twin und DPSI Twin der Firma Emcotec

Dieser Aufbau führt zu einer erheblichen Steigerung der Zuverlässigkeit und Störfestigkeit der Funkstrecke des Sicherheitspiloten. Zusätzlich verfügt das *DPSI Twin* über eine Doppelstromversorgung mit geregelter Ausgangsspannung von 5,5V und verringert damit die Gefahr eines Empfangsverlustes durch einen Akkudefekt. Außerdem ist die Verfahrensgeschwindigkeit der Aktuatoren, aufgrund der konstanten Betriebsspannung, im Hinblick auf die Einstellung der FCS-Parameter gleichbleibend (keine Änderung der Flugdynamik durch unterschiedliche Stellgeschwindigkeiten der Servos).

Im Anschluss wird durch ein nachgeschaltetes *DPSI Twin* die Möglichkeit geschaffen, über einen Servokanal – bedient durch den Sicherheitspiloten – die Umschaltung zwischen manueller und automatischer Stabilisierung und Bahnführung des Fluggerätes vorzunehmen. Abbildung 5-16 zeigt schematisch die Umsetzung der beiden kaskadierten *DPSI Twins*, wobei der Failsafe-Eingangskanal des zweiten von einem Schaltkanal des Sicherheitspiloten bedient wird. Durch Konfiguration dessen Senders werden Signale von 100% für den manuellen Flug und -100% für den durch das FCS gesteuerten Flug erzeugt, welche das Umschalten der Ausgangssignale der Aktuatoren des zweiten *DPSI Twins* bewirken.

Der in Abschnitt 5.4.1 erwähnte Aktuatortreiber, der die Stellgrößen des FCS in PWM-Signale wandelt, übernimmt das kommerzielle Elektronikboard *HBC-101* der Firma Pontech [Pontech, 2004]. Neben acht Servoausgängen verfügt das Board über acht Servoeingänge für das Einlesen bzw. Digitalisieren der Pulsweiten von PWM-Signalen. So wird beispielsweise der Wert des Schaltkanals für die Umschaltung zwischen

manuellem und automatischem Flug ausgewertet, da es für die ACU relevant ist zu wissen, ob die aktuelle Steuerautorität bei dem Sicherheitspiloten oder dem FCS liegt.

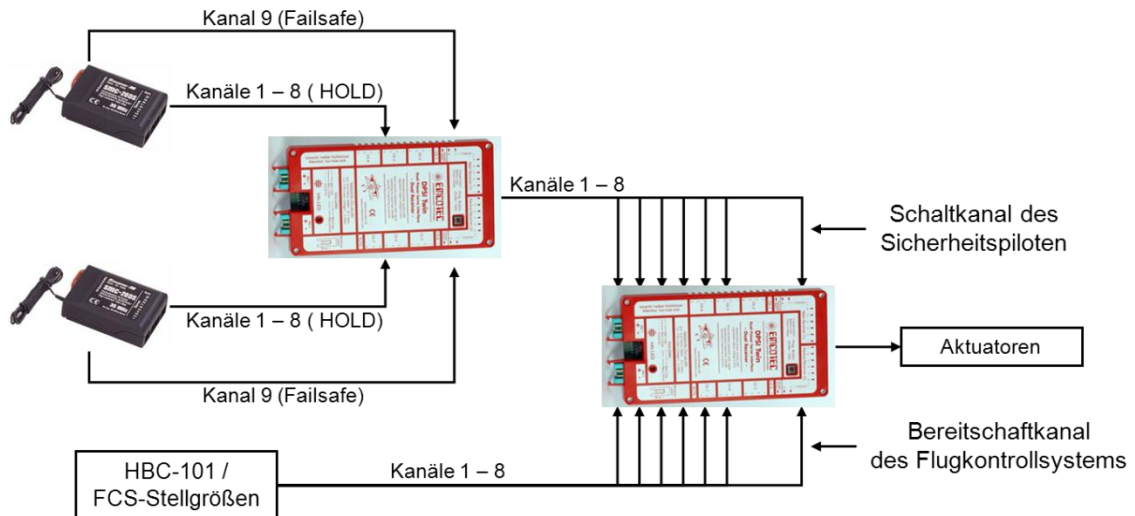


Abbildung 5-16. Schematische, kaskadierte Verschaltung der DPSI-Twins

Der Failsafe-Eingangskanal des zweiten *DPSI Twins* wird durch das FCS bzw. das nachgeschaltete HBC-101 bedient und determiniert, ob überhaupt die Kontrolle an das FCS übergeben werden kann. Liegt der Signalwert hier über 75%, so ist der Flugregler bereit. Alle Voraussetzungen für den automatischen Flugbetrieb sind erfüllt. Liegt im Gegenzug jedoch ein Wert von kleiner -75% an, so verhindert das *DPSI Twin* die Umschaltung und der Sicherheitspilot hat weiterhin die manuelle Kontrolle über das Fluggerät. Durch den Einsatz eines Rundum-Lichts (engl.: Rotating beacon) bekommt der Sicherheitspilot eine optische Rückmeldung, ob die Umschaltung erfolgreich war und das FCS die Kontrolle über die Stabilisierung und Lenkung des Fluggerätes übernommen hat. Der Fall, dass eine Umschaltung auf Automatik durch den Sicherheitspiloten nicht erfolgreich war, trat allerdings bei den durchgeführten Versuchsflügen nie auf. Gängige Praxis war hier, dass vor der Umschaltung der BKS-Operator den Zustand der Subsysteme und Automation eingehend prüfte und nachfolgend eine verbale Anweisung – mittels der Gegensprechanlage – an den Sicherheitspiloten zur Übergabe der Kontrolle an das FCS gab.

Abbildung 5-17 zeigt die komplette Verschaltung der beiden kaskadierten DPSI-Twins. Im oberen linken Bereich sind die beiden RC-Empfänger zu sehen, deren Kanäle größtenteils mit den Eingängen des ersten, linken DPSI-Twins verbunden sind (die blauen dicken Linien bezeichnen Busse, in die zur besseren Übersicht Signale eingespeist und wieder abgegriffen werden können). Einzige Ausnahme hier ist der Sensitivitätskanal für das Gyro zur Stabilisierung der Hochachse des Hubschraubers. Dieser Kanal wird einmalig bei einem manuellen Flug eingestellt und bedarf bei nachfolgenden Flügen keiner weiteren Anpassung. Die Ausgangskanäle des linken DPSI-Twin sind mit den Eingängen des zweiten, rechten DPSI-Twin verbunden (Anschluss an den „Hauptempfänger“-Eingängen) und entsprechen den manuellen Steuerdaten des Sicherheitspiloten. Die zweiten Servoeingänge des zweiten DPSI-Twin (Backupempfänger-Eingänge) sind an den Ausgängen des Pontech HBC101 angeschlossen und repräsentieren die von dem FCS errechneten Steuerkommandos. Die

- Raten-Regler (engl.: *Inner loop*)
- Lage-Regler (engl.: *Attitude loop*)
- Geschwindigkeits-Regler (engl.: *Path loop*)
- Navigations-Regler (engl.: *Navigation loop*)

Hierbei übernimmt der *Inner loop* die Regelung der körperfesten Drehraten (p = Rollrate, q = Nickrate, r = Gierrate). Die Ausgangswerte dieses Regelkreises entsprechen dabei direkt den Stellgrößen für die Aktuatoren des Fluggerätes. Neben einem klassischen, linearen Regler wurde zusätzlich ein dynamischer Inversionsregler implementiert, welcher jedoch aufgrund der guten Performanz der linearen Regelgesetzte nicht verwendet werden musste. Für alle weiter genannten Regler wurden dynamische Inversionsregler verwendet – Details der Umsetzung sind in [Höcht, 2007] beschrieben.

Die Lagestabilisierung des UAVs übernimmt der *Attitude loop*, wobei die Regelgrößen den Eulerwinkeln des Fluggerätes entsprechen (Φ = Roll- bzw. Hängewinkel, Θ = Nickwinkel, Ψ = Flugrichtung). Die Ausgabe der Berechnung dieses Regelkreises entspricht den kommandierten Drehraten (p_c , q_c , r_c) für den darunter liegenden *Inner loop*.

Der *Path loop* übernimmt die Regelung der Größe und Richtung des Geschwindigkeitsvektors des Hubschraubers ($v_{FW,c}$ = Vorwärtsgeschwindigkeit, $v_{LAT,c}$ = Seitwärts-geschwindigkeit, $v_{DOWN,c}$ = Abwärtsgeschwindigkeit). Ist dieser Modus aktiv, so werden dem darunterliegenden *Attitude loop* die entsprechenden Lagewinkel kommandiert (Φ_c , Θ_c , Ψ_c), um die gewünschte Fluggeschwindigkeit und -richtung zu erreichen.

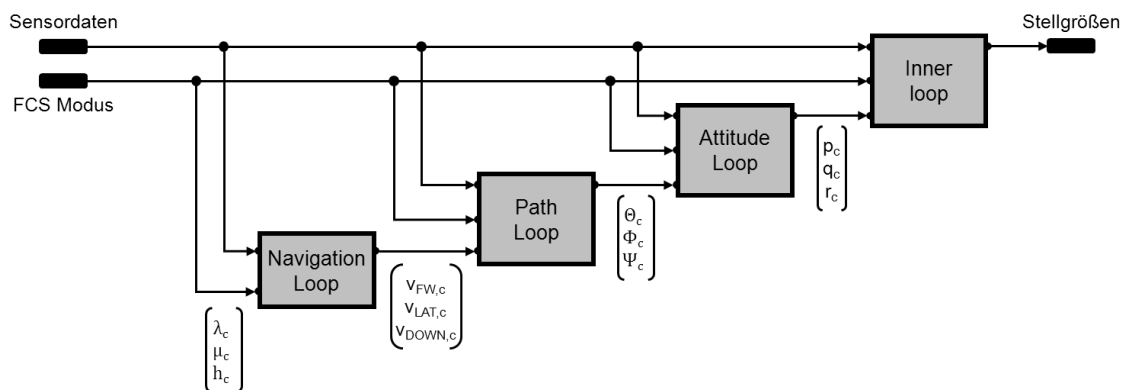


Abbildung 5-18. Kaskadierter Aufbau der vier Regelkreise des FCS nach [Höcht, 2007]

Der letzte Regelkreis ist der *Navigation loop*, der das Anfliegen von definierten Wegpunkten erlaubt. Intern werden von dem FCS insgesamt drei Wegpunkte (WGS84 Format) in einem Ringpuffer verwaltet, welche für verschiedene Arten des Anfliegens eines Wegpunkts benötigt werden (vgl. Abschnitt 5.4.5.1). Somit sorgt der Navigationsregelkreis aufgrund einer kommandierten Position (λ_c = kommandierte Longitude, μ_c = kommandierte Latitude, h_c = kommandierte Höhe) dafür, dass durch entsprechende Geschwindigkeitskommandos an den *Path loop* der aktuelle Wegpunkt angefliegen wird.

Für die Entwicklung und die Umsetzung der Regelgesetze wurde ein Matlab-Simulink-Modell erstellt. Es wurde für die spätere Integration des Reglers, mittels des Embedded-Realtime-Coders, in die Programmiersprache C++ übersetzt und nachfolgend als eigenständiger Softwareprozess kompiliert. Für die Einbettung des erzeugten Codes wurde eine Wrapperklasse (C++) entwickelt, die sowohl das Schreiben der Eingangswerte (Sensorwerte und gewünschter Betriebsmodus des FCS) als auch das Auslesen der berechneten Stellgrößen und die Ausgabe von Aktuatorbefehlen an das HBC-101 (vgl. Abschnitt 5.4.4) über eine serielle Schnittstelle übernimmt.

Im folgenden Abschnitt werden nach [Höcht, 2007] die verschiedenen Betriebsarten des FCS näher erläutert und deren Verwendung und Konfiguration durch die ACU, im Hinblick auf die Evaluierung des KCV-Konzepts, beschrieben.

5.4.5.1 FCS-Betriebsarten

Das FCS verfügt über eine Vielzahl verschiedener Betriebsarten, welche die automatische Stabilisierung und Navigation des Fluggerätes erlauben und abhängig vom aktiven FCS-Modus zu einem unterschiedlichen Verhalten des UAVs führt. Die verfügbaren Betriebsarten sind in einer Baumstruktur organisiert und unterscheiden sich auf der obersten Ebene durch folgende Modi:

- Start- und Landemodi (engl.: *Start and landing modes*): Spezielle Betriebsarten für das automatische Starten und Landen des Hubschraubers. Wurden jedoch aus Sicherheitsgründen nicht verwendet.
- Flugmodi (engl.: *Flight modes*): Umfasst alle Betriebsarten für das Fliegen des Fluggerätes. Diese reichen von der manuellen Stabilisierung des Fluggerätes durch den Operator bis zur vollautomatischen Bahnführung und das Anfliegen von Wegpunkten.

Innerhalb der *Flight modes* kann zudem zwischen den beiden Betriebsarten *manuelle Steuerung* (engl.: *Steering modes*) und den eigentlichen *Autopilotenmodi* (engl.: *Autopilot modes*) unterschieden werden.

Die *Steering modes* erlauben dem Operator die Vorgabe von Steuer- als auch Führungsgrößen mittels eines Joysticks (Eingabegerät innerhalb der BKS, vgl. Abschnitt 5.5.2) für die drei Regelkreise *Inner loop* bis *Path loop*. Hierbei verfügt der *Inner loop* über zwei verschiedene Submodi, nämlich die sogenannte *RPV*-Betriebsart (engl.: *Remotely piloted vehicle*), bei der die Stellwerte der Aktuatoren direkt von den aktuellen Joystick-Kommandos und in der zweiten Betriebsart (*Rate*) körperfeste Drehraten vorgegeben werden. Darüber hinaus können im *Attitude loop* Lagewinkel und im *Path loop* entsprechende Geschwindigkeiten kommandiert werden, wobei die jeweiligen Maximalwerte hierbei einstellbar und parametrisiert sind. Für die automatische Flugführung durch die KCV-ACU wurden diese Betriebsarten jedoch nicht verwendet, da hierfür der menschliche Operator zur Erzeugung kontinuierlicher Sollgrößen mittels eines Eingabegerätes, wie beispielsweise eines Joysticks, benötigt wird.

Im Gegenzug ermöglichen die *Autopilot modes* eine Verhaltenszeugung des Fluggerätes durch das FCS auf Basis von parametrisierten Sollgrößen. Diese Modi nutzen mindestens den Regelkreis *Path loop* oder den darüber liegenden *Navigation loop*. Zudem kann hier zwischen horizontalen und vertikalen Geschwindigkeitsmodi (engl.: *Speed modes*) sowie Positionsmodi (engl.: *Position modes*) unterschieden werden. Bei der Durchführung der Evaluationsmission nutzt die KCV-ACU hauptsächlich diese Betriebsarten, um das gewünschte Verhalten des Fluggerätes zu erzeugen. Tabelle 6 gibt eine Übersicht über die verschiedenen Betriebsarten der *Autopilot modes* sowie eine kurze Beschreibung des daraus resultierenden Verhaltens des Fluggerätes.

<i>Horizontale Geschwindigkeitsmodi (Speed_Hx modes)</i>		
<i>Modus</i>	<i>Geschwindigkeits-/Winkelvorgaben</i>	<i>Beschreibung</i>
Speed_H1	Vorwärts-, Seitwärts-geschwindigkeit	Der Operateur kann mittels Parametern eine Vorwärts- und Seitwärts-geschwindigkeit vorgeben (vergleichbar mit dem höchsten <i>Steering mode</i> , wobei die Vorgaben nicht von dem Eingabegerät des UAV-Operateurs erzeugt werden).
Speed_H2	Nord-, Ost-geschwindigkeit	Vorgabe einer Nord- und Ostgeschwindigkeit (NED-Koordinatensystem – North – East – Down). Betriebsart, welche hauptsächlich intern von dem FMS verwendet wird.
Speed_H3	Absolute Geschwindigkeit und Bahnazimutwinkel (χ)	Vorgabe einer absoluten Geschwindigkeit und eines Bahnazimutwinkels (χ).
<i>Vertikale Geschwindigkeitsmodi (Speed_Vx modes)</i>		
<i>Modus</i>	<i>Geschwindigkeits-/Winkelvorgaben</i>	<i>Beschreibung</i>
Speed_V1	Flugpfadwinkel (γ)	Vorgabe eines Flugpfadwinkels (γ). Die Berechnung der Steiggeschwindigkeit ist gekoppelt an die aktuelle Horizontalgeschwindigkeit des Fluggerätes.
Speed_V2	Steiggeschwindigkeit (\dot{h})	Vorgabe einer absoluten, vertikalen Geschwindigkeit (\dot{h}).
<i>Horizontale Positionsmodi (Pos_Hx modes)</i>		
<i>Modus</i>	<i>Wegpunktgrößen</i>	<i>Beschreibung</i>
Pos_H1	Track aus Wegpunktkoordinaten	Abfliegen des horizontalen Tracks zwischen dem aktuellen (P_{akt}) und dem vorigen Wegpunkt (P_{akt-1}) (engl.: Tracking, vgl. Abbildung 5-19)
Pos_H2	Aktueller Wegpunkt (horizontale Koordinaten)	Direktes Anfliegen des aktuellen Wegpunkts (P_{akt}) unabhängig der aktuellen Position (Pos_H) (engl.: Homing, vgl. Abbildung 5-19)
Pos_H3	Aktueller Wegpunkt (horizontale Koordinaten)	Der aktuelle Wegpunkt wurde erreicht und die Koordinate des Wegpunkts wird gehalten.

Vertikale Positionsmodi (Pos_Vx modes)		
Modus	Wegpunktgrößen	Beschreibung
Pos_V1	Track aus Wegpunktkoordinaten	Abfliegen des vertikalen Tracks zwischen dem aktuellen und dem vorigen Wegpunkt (vgl. Abbildung 5-20 links).
Pos_V2	Aktueller Wegpunkt (Höhe)	Direktes Anfliegen der Höhe des aktuellen Wegpunkts (vgl. Abbildung 5-20 rechts).
Pos_V3	Aktueller Wegpunkt (Höhe)	Die Höhe des aktuellen Wegpunkts wurde erreicht und wird gehalten.

Tabelle 6. Verfügbare Autopilot modes des FCS nach [Höcht, 2007]

Die genannten horizontalen Positionsmodi Pos_H1 und Pos_H2 unterscheiden sich dahingehend, wie bzw. auf welcher Route der aktuelle Wegpunkt angefliegen wird. So regelt die Betriebsart Pos_H1 die Position des Hubschraubers auf die horizontale Verbindungslinie (engl.: Track) zwischen dem aktuellen und dem vorigen Wegpunkt. Im Falle einer Ablage von dem geplanten Flugweg, zum Beispiel durch Seitenwind, versucht der Regelkreis diese Ablage zu minimieren und somit den Wegpunkt aus der Richtung des vorigen Wegpunkts anzufliegen (vgl. Abbildung 5-19 links). Im Gegensatz versucht jedoch die Betriebsart Pos_H2, unabhängig von der aktuellen Position des UAVs, den Wegpunkt zu erreichen, was durch Windeinflüsse zu einer gekrümmten Flugbahn führen kann (vgl. Abbildung 5-19 rechts).

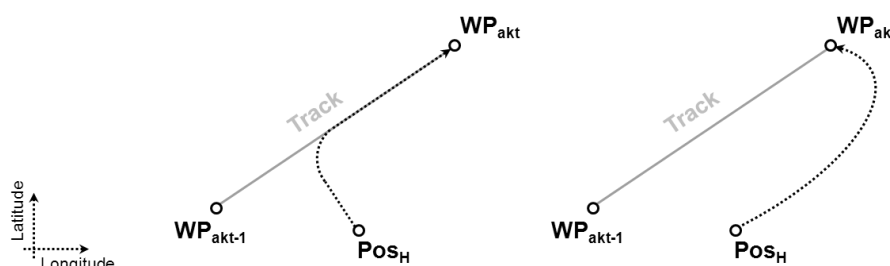


Abbildung 5-19. Horizontaler Flugpfad des UAVs mit den horizontalen Positionsmodi Pos_H1 (links) und Pos_H2 (rechts) nach [Höcht, 2007]

Ähnlich den horizontalen Positionsmodi verfügt das FCS über vertikale Modi, die ein unterschiedliches Anfliegen der Zielhöhe eines Wegpunkts bzw. der Verbindungslinie zwischen der Höhe des aktuellen und des vorigen Wegpunkts erlauben. Bei Aktivierung der Betriebsart Pos_V1 wird abhängig von der aktuellen Position des Hubschraubers die Zielhöhe durch Interpolation zwischen den beiden Höhenangaben der Wegpunkte berechnet und die Abweichung von dem Track durch den *Navigation loop* ausgeregelt (vgl. Abbildung 5-20 links). Ist jedoch die Betriebsart Pos_V2 aktiv, so wird direkt die Zielhöhe des aktuellen Wegpunkts angefliegen (vgl. Abbildung 5-20 rechts).

Für die Ausrichtung und Regelung der *Hochachse* des Hubschraubers (nach [Höcht, 2007] die sogenannten *Attitude modes*) gibt es drei verschiedene Betriebsarten (vgl. Tabelle 7). Diese sind weitestgehend entkoppelt (vgl. Abschnitt 3.2.9 *Moding logic* in [Höcht, 2007]) von den Geschwindigkeits- und Positionsmodi des FCS und erlauben die Regelung des Steuerkurses des Hubschraubers (Formelzeichen Ψ_H) nach unterschiedlichen Vorgaben.

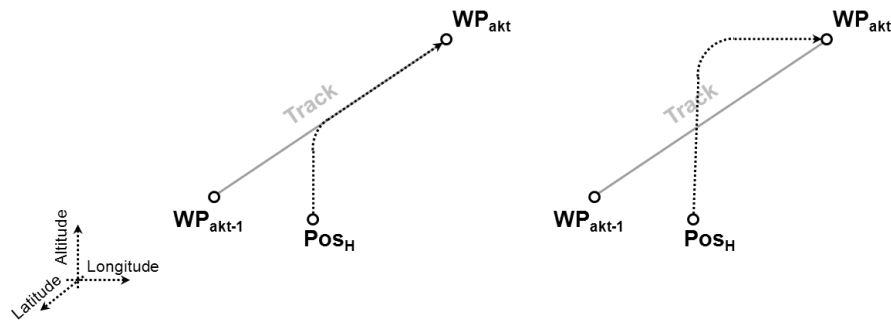


Abbildung 5-20. Vertikaler Flugpfad des UAVS mit den vertikalen Positionsmodi Pos_V1 (links) und Pos_V2 (rechts) nach [Höcht, 2007]

Bei der Vorgabe eines Steuerkurses (Formelzeichen Ψ , engl.: Heading) sorgt die erste Betriebsart (Modus *Att1*) für die Ausrichtung der Längsachse des Hubschrauberheadings in eine kommandierte Richtung ($\Psi_H = \Psi_{cmd}$, Modus *Att1*, vgl. Abbildung 5-21) – unabhängig von der aktuellen Bewegungsrichtung und -geschwindigkeit des Fluggerätes. Da ein Hubschrauber in der Lage ist auch seitlich oder rückwärts zu fliegen sowie durch den Einfluss von Wind können sich dabei verschiedene Schiebewinkel (engl.: Sideslip Angle, Driftwinkel zwischen der Längsachse eines Fluggerätes und der Richtung der Luft-Anströmung) einstellen.

Bewegt sich der Hubschrauber über Grund, so wird diese Positionsänderung von der IMU sensiert. Dabei wird sowohl die (horizontale) Grundgeschwindigkeit (Formelzeichen V) als auch die Richtung (der sogenannte Bahnazimutwinkel, Formelzeichen χ) ermittelt. Der zweite *Attitude Mode* (Modus *Att2*) sorgt, ab einer parametrisierten Mindestgeschwindigkeit, für die Angleichung des Headings in Richtung der horizontalen Fluggeschwindigkeit ($\Psi_H = \chi$, vgl. Abbildung 5-21). Wird diese Mindestgeschwindigkeit unterschritten, etwa wenn ein Wegpunkt erreicht wurde, so wird der zuletzt anliegende Headingwert beibehalten.

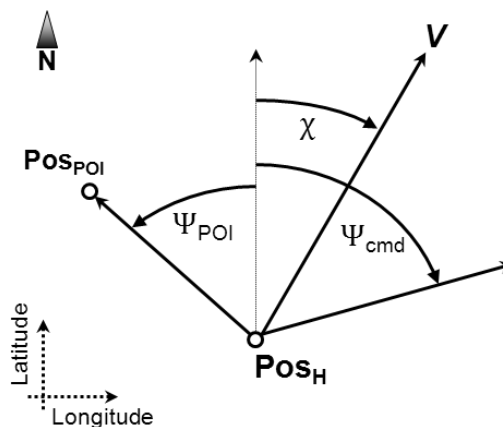


Abbildung 5-21. Ausrichtung der Längsachse des Fluggerätes durch verschiedene FCS-Betriebsarten

Die letzte Betriebsart (Modus *Att3*) sorgt für das Ausrichten des Hubschraubersteuerkurses auf einen erdfesten Zielpunkt (engl.: Point of Interest – POI, vgl. Abbildung 5-21). Die Peilung (engl.: Bearing, Ψ_{POI}) wird dabei aus der aktuellen Position des Hubschraubers und der des erdfesten Punkts berechnet. Dieser Modus ermöglicht es einem durch zwei Wegpunkte definierten Track zu folgen (vorausgesetzt,

die entsprechenden Positionsmodi des FCS sind aktiviert) während das Heading des UAVs auf den Zielpunkt ausgerichtet bleibt.

<i>Betriebsarten für die Regelung der Hochachse des UAVs (ATTx modes)</i>		
<i>Modus</i>	<i>Richtungsvorgaben</i>	<i>Beschreibung</i>
Att1	Kommandiertes Heading ($\Psi_H = \Psi_{cmd}$)	Ausrichten des Headings des Hubschraubers (Ψ_H) in eine kommandierte Richtung (Ψ_{cmd}), unabhängig des horizontalen Geschwindigkeitsvektors (V).
Att2	Heading = Bahnazimutwinkel ($\Psi_H = \chi$)	Ausrichten des Headings des Hubschraubers (Ψ_H) in Richtung des horizontalen Geschwindigkeitsvektors (Bahnazimutwinkel χ).
Att3	Erdfester Punkt ($\Psi_H = \Psi_{POI}$)	Ausrichten des Heading (Ψ) des Hubschraubers auf einen erdfesten Punkt.

Tabelle 7. Verfügbare Betriebsarten für die Regelung der Hochachse des UAVs nach [Höcht, 2007]

Die zuvor beschriebenen Betriebsarten des FCS werden nach dem in Kapitel 3 beschriebenen Konzept eines *Knowledge Configured Vehicle* durch das maschinen-spezifische Wissen abstrahiert und über die KCV-Schicht dem missions-spezifischen Wissen in Form von *Fähigkeiten* für die automatisierte Durchführung der Mission bereitgestellt (vgl. Kapitel 4). So entspricht die Kombination der FCS-Betriebsarten *Pos_H1* und *Pos_V1* der *Fähigkeit FOLLOW_WP_ROUTE*, wenn in Folge mehrere Wegpunkte abgeflogen (Das Weiterschalten der Wegpunkte erfolgt jedoch nicht automatisch, sondern wird durch das missions-spezifische Wissen ausgelöst) bzw. der *Fähigkeit TRACK_WP*, wenn nur der aktuelle Wegpunkt (auf dem Track) angefliegen werden soll. Im Gegenzug kommt die Kombination der Betriebsarten *Pos_H2* und *Pos_V1* der *Fähigkeit FLY_TO_WP* gleich, welche in der Evaluationsmission für das Zurückkehren zum Startpunkt verwendet wird, falls die ACU technische Probleme bzw. eine zu geringe Ressourcenversorgung erkennt. Des Weiteren entsprechen die Attitude-Betriebsart *Att2* der *Fähigkeit FLIGHT_PATH* und die Betriebsart *Att3* der *Fähigkeit TARGET_POINT*.

Bei der Auslegung und der Entwicklung des FCS wurde versucht einen generischen Regler für einen Drehflügler zu entwickeln, um den langwierigen Prozess der Identifikation von Modellparametern möglichst zu vermeiden. Eine Beschreibung des Konzepts und des Vorgehens für das Tuning der FCS-Parameter, für die oben genannten Regelkreise, ist Gegenstand des folgenden Abschnitts.

5.4.5.2 *Tuning und Einstellen der FCS-Parameter*

Für das Einstellen der FCS-Parameter wird ein BKS-Operator benötigt, der ebenfalls – wie der Sicherheitspilot – über die Fertigkeiten verfügt, einen Modellhubschrauber manuell zu stabilisieren. Das Konzept sieht vor, dass der Sicherheitspilot die Kontrolle über das UAV an den BKS-Operator übergibt und dieser das Fluggerät über das FCS, in einer *Steering-Mode* Betriebsart, steuert. Hierdurch können während der Testflüge die Verstärkungsfaktoren und Parameter der entsprechenden Regelkreise empirisch ermittelt, eingestellt sowie direkt die Reaktionen des Hubschraubers bewertet werden.

Um reproduzierbare Systemreaktionen des Hubschraubers auf Steuergrößen zu erhalten, wurde zudem im FCS die Erzeugung von sogenannten klinischen Signalen vorgesehen. Diese können auf jede Steuergröße (Roll, Nick, Gier, Kollektiv) aufgeschaltet und im Hinblick auf deren Amplitude, Dauer sowie Anfangs- und Endfrequenz (für das Wobbeln – engl.: Frequency sweep) parametrisiert werden. Die möglichen klinischen Signale umfassen nach [Höcht, 2007] die folgende Signalformen:

- Doublette
- Sinus Schwingung mit konstanter Frequenz
- Sinus Schwingung mit linear zunehmender Frequenz
- Sinus Schwingung mit logarithmisch zunehmender Frequenz
- Rechtecksignal mit konstanter Frequenz

Aufgrund der vier kaskadierten Regelkreise des FCS kann das Einstellen der Reglerparameter von dem innersten Regelkreis, dem *Inner loop*, bis zum dem äußersten, dem *Navigation loop*, erfolgen. Das von [Colditz, 2009] entwickelte Konzept und Vorgehen wurde anfangs in der in Abschnitt 5.3 beschriebenen Simulationsumgebung validiert und nachfolgend, wie in der Arbeit von [Prang, 2009] beschrieben, am realen UAV-Demonstrator BIG-I durchgeführt.

Das Vorgehen teilt sich in sechs Phasen mit insgesamt rund 30 unterschiedlichen Flugversuchen (engl.: Flight tests), in welchen sukzessiv die entsprechenden Werte der Verstärkungsfaktoren und Modellparameter der einzelnen Regelkreise erfolgen und nachfolgend validiert werden. Diese sind nach [Colditz, 2009]:

- Bodentests (engl.: *Ground tests*): Überprüfung der Sensorwerte auf deren Validität und Richtigkeit. Besonderes Augenmerk wurde hier bei dem horizontal und nach Norden ausgerichteten Fluggerät (Lagewinkel $\pm 2^\circ$) einerseits auf die Genauigkeit der Sensorwerte – im Hinblick auf das Rauschen und andere Fehler (z.B. Sensorbias und Misalignment) – und andererseits auf die Sinnrichtigkeit der Messgrößen gelegt.
- Verstärkungsfaktoren der Rückführung im *Inner loop* (engl.: *Inner loop feedback gains*): Diese Verstärkungsfaktoren sorgen für eine Dämpfung der Drehraten des Fluggerätes (Roll-, Nick-, Gierrate). Zur Ermittlung der höchsten Dämpfung wurden die Steuergrößen des Hubschraubers mit klinischen Signalen beaufschlagt (Doubletten) und Testflüge mit verschiedenen Dämpfungsfaktoren durchgeführt. Die resultierenden Flugdaten wurden nachfolgend ausgewertet und anhand der Systemreaktionen auf den jeweiligen Achsen die Parameter mit der bestmöglichen Dämpfung selektiert.
- Verstärkungsfaktoren des Vorwärtszweigs im *Inner loop* (engl.: *Inner loop feedforward gains*): Diese Parameter stellen Proportionalitätsfaktoren dar, um eine gewünschte bzw. kommandierte Drehrate auf den zugehörigen Steuerausschlag umzurechnen. Aufgrund der hohen Nichtlinearität des Systemverhaltens des Hubschraubers wurde eine

Referenzrate von zehn Grad pro Sekunde ausgewählt und die entsprechenden Faktoren erflogen und validiert.

- Verstärkungsfaktoren im *Attitude loop* (engl.: *Attitude loop error gains*): Dieser nichtlineare Regelkreis übernimmt die Lageregelung der Fluggerätes. Neben den Verstärkungsfaktoren müssen die Zeitkonstanten der entsprechenden Referenzmodelle (vgl. [Höcht, 2007]) bestimmt und ermittelt werden. Da das FCS nicht für extreme Manöver ausgelegt ist, wurden die maximalen Roll- und Nickwinkel auf $\pm 60^\circ$ limitiert und die Zeitkonstanten nach Vorgaben der ADS-33 Spezifikation [USAA & MC, 2000] (Aeronautical design standard – Performance specification handling qualities requirements for military rotorcraft) berechnet [Colditz, 2009].
- Verstärkungsfaktoren im *Path loop* (engl.: *Path loop gains*): Ähnlich dem *Attitude loop* müssen im *Path loop* entsprechende Reglerverstärkungsfaktoren und Zeitkonstanten ermittelt werden, welche Einfluss auf die Geschwindigkeiten (longitudinal, lateral und vertikal) des Fluggerätes nehmen. Einer der wenigen modellspezifischen Parameter dieses Regelkreises stellt der sogenannte „*ThrustAllocationGain*“ Verstärkungsfaktor dar, der den Zusammenhang zwischen dem aktuellen Schub und dem kollektiven Anstellwinkel der Rotorblätter beschreibt. Er wird durch ein iteratives Verfahren in Bodennähe ermittelt und später außerhalb des Bodeneffekts validiert und feinabgestimmt.
- Verstärkungsfaktoren im *Navigation loop* (engl.: *Navigation loop gains*): Der letzte und oberste Regelkreis übernimmt die Navigation und Positionsregelung des UAVs nach Wegpunktvorgaben (WGS-84-Koordinatensystem). Für diese Betriebsarten können die Verstärkungsfaktoren – aufgrund des Aufbaus der darunterliegenden Regelkreise und der Annahme, dass sich das Geschwindigkeitsverhalten des Hubschraubers wie ein PT1-System verhält – für den *Navigation loop* direkt berechnet werden (Auslegungskriterium: Kritische Dämpfung der Positionsregelung).

Für die genannten Phasen wurden von Colditz einerseits Herangehensweisen für das Einstellen und Erfliegen der FCS-Parameter der jeweiligen Regelkreise entwickelt und nachfolgend in der Simulation evaluiert. Andererseits wurden für die Auswertung der Flugdaten zum Teil heuristische Einstellregeln angewendet, wie beispielsweise das Ziegler-Nichols Verfahren [Lunze, 2008] und auch entsprechende Unterstützungssoftware in Form von Matlab-Skripten erstellt. Diese erlauben eine einfache und schnelle Auswertung, mit dem Ziel der Ermittlung, Berechnung und Validierung der jeweiligen FCS-Parameter.

Die Arbeit von [Prang, 2009] dokumentiert ausführlich die Ergebnisse der Einstellflüge, die zugehörigen Flugdaten sowie die ermittelten Verstärkungsfaktoren und FCS-Parameter. Zudem wurden im Rahmen dieser Arbeit Änderungen an den einzelnen Flugversuchen durchgeführt, um einerseits das Vorgehen zu operationalisieren und

möglichst pro Testflug mehrere Parameter zu erfliegen und zu validieren und andererseits risikoreiche Versuche (wie beispielsweise die Erfliegung des *ThrustAllocationGains*) zu entschärfen und zu vereinfachen.

5.5 Kontrollstation

Die entwickelte Kontrollstation dient dem Operator als Arbeitsplatz für die Überwachung und die Führung des im vorigen Abschnitt vorgestellten UAV-Demonstrators BIG I. Die folgenden Abschnitte beschreiben die Plattform, den Aufbau, die Umsetzung und Integration der verwendeten Hardwarekomponenten, um die in Abschnitt 5.1 definierten Anforderungen zu erfüllen.

5.5.1 Plattform

Basis der Kontrollstation ist ein Mercedes Sprinter, welcher einerseits für den Transport des Fluggerätes zum Fluggelände und andererseits als Führungsleitstand während des Flugbetriebs verwendet wird. Diese bodengebundene Kontrollstation (BKS) soll auch als Surrogat für fliegende Kontrollstationen, wie beispielsweise für das Projekt MUM-T (vgl. Abschnitt 2.1.7.2) benötigt, eingesetzt werden. Deshalb wurde eine interne Stromversorgung mit Batterien, einem AC-Wandler und einem Diesel-Generator integriert, um einen mobilen Einsatz zu ermöglichen. Durch die Verfügbarkeit eines leistungsfähigen 220V Bordnetzes wird die Versorgung der für die Führung des UAVs erforderlichen Hardwarekomponenten sichergestellt. Es erlaubt zudem den Einsatz von Standard-PC Komponenten und anderen notwendigen Gerätschaften, die mit konventioneller Netzspannung betrieben werden können.



Abbildung 5-22. Mobile Bodenkontrollstation auf Basis eines Mercedes Sprinters

Abbildung 5-22 zeigt die Bodenkontrollstation und den Ausbau, im hinteren Bereich des Fahrzeugs mit Auszugsschienen, für den Transport des/der Fluggerätes/Fluggeräte

und Stauraum in Form von Schubladen für die Aufnahme von Ersatzteilen und Werkzeug.

Abbildung 5-23 zeigt den Aufbau des Operateur-Arbeitsplatzes. Zwei übereinander angeordnete Touchscreen-Monitore bieten dem Bediener entsprechende Eingabemöglichkeiten für die Interaktion mit dem unbemannten Fluggerät. Zudem können die aktuellen Flug- und Systemzustände überwacht werden, welche in verschiedenen, vom Operateur wählbaren, Displayarten (vgl. Abschnitt 5.5.3.1) dargestellt werden.



Abbildung 5-23. Arbeitsplatz des Operateurs innerhalb der Bodenkontrollstation

Für den Betrieb der BKS und die Führung des UAVs BIG I werden mindestens zwei Personen – nämlich der BKS-Operateur und der Sicherheitspilot – benötigt. Die Kommunikation zwischen beiden wird durch eine bidirektionale, funkbasierte DECT-Gegensprechanlage (Hersteller: Fa. Peltor) mit einer Reichweite von ca. 500m LOS (engl.: Line of sight) sichergestellt. Zudem bietet dieses System eine Audio-Eingangsschnittstelle, über die computergenerierte Meldungen, wie etwa Status-Meldungen oder kritische System Warnungen, in das Kommunikationssystem eingespeist werden können (vgl. Abbildung 5-24).

5.5.2 Hardwarekonzept

Abbildung 5-24 zeigt die Umsetzung des Hardwarekonzepts für die Bodenkontrollstation. Die Rechenleistung für die Darstellung des Flug- und Systemzustands in Form von Displays, die Anbindung von weiteren externen Geräten (GPS, Datenfunk, Intercom, ...) und die Ausführung der BKS-Softwareprozesse wird von vier Standard-PCs (untergebracht in zwei 19-Zoll Racks) bereitgestellt. Die Anzeige erfolgt über eine VGA-Matrix (engl.: VGA – Video graphics array, Computergrafik-Standard), welche eine frei Zuordnung der Videosignale auf zwei Touchscreen Monitoren erlaubt.

Für die Kommunikation mit dem UAV wird ebenfalls ein Datenfunk-Transceiver auf Basis der Sateline 3AS(d) Modems, für den Versand von Kommando- und den Empfang von Telemetrie- und Statusnachrichten, verwendet (vgl. Abschnitt 5.4.2.4).

Ein GPS-Empfänger G2L der Firma Novatel erfasst die Position der BKS und kann auch als Basisstation für die Erzeugung von DGPS (engl: Differential global positioning system) Korrekturdaten fungieren, um die Positionsgenauigkeit des UAVs zu erhöhen. Im Fall der vorliegenden Arbeit wurde zwar der Einsatz eines weiteren Novatel G2L GPS Empfänger an Bord des unbemannten Fluggerätes vorbereitet, jedoch aufgrund der ausreichend hohen Positionsgenauigkeit der Crossbow NAV420 IMU nicht umgesetzt. Die Position der BKS wird dennoch für die Berechnung der Ausrichtung eines automatischen Kameraverfolgers (vgl. Abbildung 5-22) benötigt, der das automatische Filmen des unbemannten Fluggerätes und das Positionierung von Richtantennen übernimmt und sich derzeit in Entwicklung befindet.

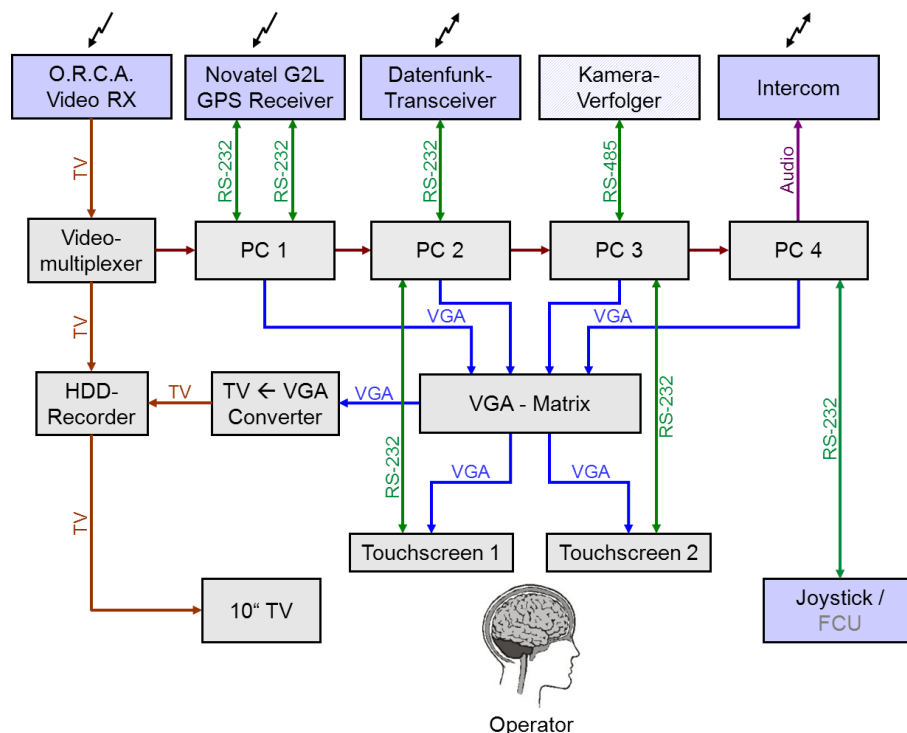


Abbildung 5-24. Umsetzung des Hardwarekonzepts der Bodenkontrollstation

Für den Empfang der Videofunkdaten des auf dem UAV installierten O.R.C.A. Systems (vgl. Abschnitt 5.4.3) wurde der zugehörige Empfänger in der BKS integriert, welcher die digitalen Signale wandelt und in Form eines analogen Fernsehsignals (PAL-Standard, Auflösung 720x576 Pixel) ausgibt. Ein Multiplexer übernimmt die Vervielfältigung dieses Signals. Nachfolgend wird es einerseits jedem PC über eine Video-Eingangsschnittstellenkarte – für die Darstellung des Live-Videostroms in einem Display – und andererseits einem Festplatten-Videorekorder (engl.: HDD – Hard disc drive) – für die Aufzeichnung der Flugvideos – zugeführt. Zudem können die VGA-Signale der einzelnen Rechner mittels eines Konverters in das analoge TV-Signal überführt und ebenfalls mit dem HDD-Rekorder aufgezeichnet werden (vgl. Abbildung 5-24). Zur Anzeige des Videosignals wird dem Bediener noch ein kleiner 10“ Bildschirm bereitgestellt.

An Eingabegeräten stehen dem Bediener, neben den Touchscreen-Monitoren, noch ein analoges Eingabegerät in Form eines Joysticks (umgebaute Flybox der Firma BG Systems, Inc.) und eine FCU (engl.: Flight control unit – Gerät zum Einstellen von Autopilotenmodi und -parametern) zur Verfügung. Sie ist jedoch in Abbildung 5-23 nicht dargestellt, da sie derzeit im MUM-T Projekt eingesetzt wird.

5.5.3 Softwarekonzept

Abbildung 5-25 zeigt die wichtigsten Prozesse, die innerhalb der BKS für die Flugführung des UAV-Demonstrators BIG I entwickelt wurden. Die Datenverteilung mittels Interprozesskommunikation erfolgt analog dem auf dem FCC eingesetzten Prinzip der Queues und Datakeys. Für den Transport der Daten können jedoch nicht gemeinsam genutzte Speicherbereiche verwendet werden, da die einzelnen Prozesse auf mehrere Rechner verteilt werden müssen. Zum einen reicht die Anzahl der verfügbaren Schnittstellen nicht aus, um alle Geräte an einen Rechner anzuschließen und zum anderen ist die Rechen- bzw. die Grafikleistung für die Erzeugung der Display-Anzeigen zu gering.

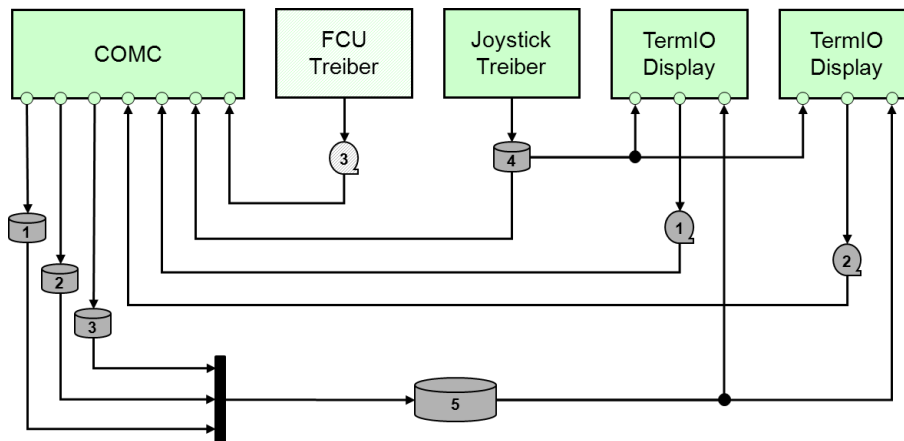


Abbildung 5-25. Prozessstruktur und Verteilung der Daten innerhalb der BKS

Für Verteilung der Daten wurde deshalb die Interprozesskommunikation, auf Basis von MICO entwickelt, die eine Implementierung des CORBA-Standards darstellt (engl.: Common Object Request Broker Architecture – Allgemeine Architektur für Vermittler von Objekt-Anforderungen) und die netzwerkbasierte Kommunikation von Prozessen über Rechengrenzen hinweg ermöglicht. Die in Abbildung 5-25 dargestellten Queues und Datakeys werden von einem zentralen MICO-Serverprozess (BKS-Server) repräsentiert, der aus Übersichtgründen nicht explizit abgebildet ist.

Der COMC-Prozess, zuständig für die Abwicklung der Datenfunkkommunikation, empfängt von dem UAV eine Reihe verschiedener Datenpakete und beschreibt, mit den entsprechenden Informationen, die zugehörigen Datakeys (DK1-3). Diese Daten umfassen kontinuierlich gesendete Daten wie beispielsweise Telemetrieinformationen (DK1) und die von dem FCS errechneten Steuergrößen (DK2), aber auch den Zustand und die aktuellen Parameter des FCS (DK3). Diese werden allerdings nur versendet, wenn sich die Betriebsart aufgrund von externen Kommandos durch den Operateur, die ACU, oder durch das interne FCS-Moding ändert. Für die Bereitstellung dieser Daten für andere Prozesse innerhalb der BKS werden diese durch den BKS-Server-Prozess zu

einem gesamtheitlichen Datenpaket zusammengefasst (enthält sämtliche Sensordaten sowie Informationen über den aktuellen Betriebszustand und -parameter des FCS, sowie Informationen über den aktuellen Zustand der Datenlinkverbindung vgl. Abschnitt 5.6.2) und können über DK5 ausgelesen werden. Teile dieser Daten werden von den TermIO-Prozessen in Form verschiedener Displays zur Anzeige gebracht und werden im folgenden Abschnitt noch weiter beschrieben und erläutert.

Die Touchscreen-Monitore werden ebenfalls, abhängig von der durch den Bediener gewählten Displayart, auch als Eingabegeräte verwendet. Eingegebene Kommandos werden nachfolgend über die Queues Q1 und Q2 an dem COMC-Prozess weitergeleitet und anschließend zu dem unbemannten Fluggerät gesendet. Ein weiteres dem BKS-Operateur zur Verfügung stehendes Eingabegerät ist ein Joystick. Dessen Steuerdaten werden ausgelesen und mittels Datakey DK4 an den COMC-Prozess weitergereicht und kontinuierlich, mit einer Frequenz von 20Hz, verschickt. Das ist vor allem für die manuelle Stabilisierung und die Vorgabe von Sollgrößen durch den BKS-Operateur für das FCS des UAVs unabdingbar.

5.5.3.1 BKS-Displays

Für die Anzeige von aktuellen Flug- und Systemzustandsdaten wurden verschiedene Anzeigen entwickelt und in den TermIO-Prozess integriert. Ziel war es, dem menschlichen Operateur die Möglichkeit zu geben, den gegenwärtigen Status des UAVs zu überwachen und auch mit diesem zu interagieren.

Abbildung 5-26 zeigt ein Bildschirmfoto des TermIO-Prozesses mit der gewählten Anzeigenart eines Kartendisplays. Es erlaubt dem Bediener eine sehr intuitive Erfassung der aktuellen Position des Fluggerätes (dargestellt durch das Hubschraubersymbol), des aktuellen Gierwinkels (Ψ , Ausrichtung des Hubschraubersymbols), des momentanen Geschwindigkeitsvektors (gelber Pfeil zeigt auf die Position, an der sich das UAV in fünf Sekunden befinden wird, wenn die aktuelle Bewegung nicht verändert wird) und des zurückgelegten Flugweges (gelb, verblassende Spur). Neben einigen Zahlenwerten, die Auskunft über die Höhe des Startplatzes (Galt – engl.: Ground altitude), die barometrische Flughöhe des UAVs (BAlt – engl.: Barometric altitude), die aktuellen Höhe über dem Startplatz (AGL – engl.: Above ground level) und der Vertikalgeschwindigkeit (HDot) geben, hat der Bediener einige Tasten zur Verfügung, die es ihm erlauben den Flugplan (Wegpunktlisten) zu modifizieren (laden, speichern, ändern), die Darstellung, im Hinblick auf die Ausrichtung, den Zoomfaktor und die Helligkeit der Karte vorzugeben.

Die einzelnen Anzeigarten können über entsprechende Reiter (Map, Autopilot, Telemetry, ...) ausgewählt werden und erlauben dem Operateur die Auswahl des gewünschten Displays.

Oberhalb der Reiter sind verschiedene Anzeigen in fünf Gruppen zusammengefasst. Ganz links zeigen mehrere Schaltflächen den aktuellen Modus, ob derzeit der Sicherheitspilot (OFF-MANUAL) oder das FCS (ON-AUTOMATIC) die Stabilisierung des UAVs übernimmt, die Einsatzbereitschaft des FCS, dessen aktuelle Betriebsart und den Status der Bodenkontaktschalter (WOW L. und R. – engl.: Weight on wheels).

Rechts daneben wird der aktuelle Status des Log-Prozesses angezeigt, der auf dem FCC verschiedene Arten von Daten (DR NAV = Durch den NAV420-Treiber dekodierte Sensordaten der IMU, AP NAV = Sensordaten bei der Berechnung der Stellgrößen durch das FCS, ...) aufzeichnen kann. Durch Klicken auf die entsprechenden Schaltflächen kann der Bediener die Aufzeichnung bestimmter oder auch aller Daten starten und beenden.



Abbildung 5-26. Der Prozess TermIO mit der gewählten Anzeigeart: Kartendisplay (engl.: Map)

Die mittlere Gruppe zeigt die Daten des Sensorboards (vgl. Abschnitt 5.4.1.1), wie beispielsweise die Werte der beiden Ultraschallentfernungsmesser, die aktuelle Spannungslage der Avionikstromversorgung sowie den Stromverbrauch der Avionikkomponenten, die aktuelle Lufttemperatur und den barometrischen Atmosphärendruck.

Ganz rechts werden aktuellen Steuergrößen (Wertebereich [-1; 1]) des FCS (rot) und der Steuereingaben des UAV-Operators (grün) dargestellt. Das rechte Quadrat zeigt dabei die Magnitude der zyklischen Steuergrößen Roll (horizontal) und Nick (vertikal). Das Linke zeigt die kollektive Steuergröße (vertikal) und den Ausschlag des Heckrotors (horizontal).

Die letzte Gruppe (zweite von rechts) zeigt die momentan Auslastung und die Integrität der Datenfunkverbindung. Dieser Werte werden jeweils für den BKS- und den UAV-Transceiver berechnet und in zwei Diagrammen dargestellt. Hierbei stellt in Abbildung 5-26 jeweils die obere Kurve die Integrität und die untere Kurve die Auslastung dar. Die Berechnung und die Umsetzung des implementierten Datenlink- bzw. Kommunikationsmanagements zwischen der Bodenkontrollstation und dem UAV ist Gegenstand des folgenden Abschnitts.

5.6 Kommunikationsmanagement

Der Austausch von Telemetriedaten und Kommandoinformationen zwischen dem UAV und der Bodenkontrollstation (vgl. Abbildung 5-27) erfolgt über Datenfunkmodems (vgl. Abschnitt 5.4.2.4), welche sowohl in der BKS als auch an Bord des UAVs integriert wurden.

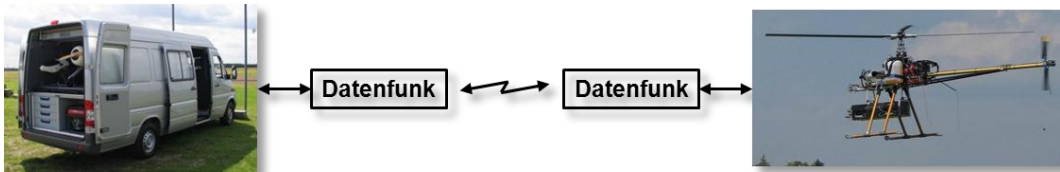


Abbildung 5-27. Informationsaustausch zwischen der MBKS und dem UAV

Im Vergleich zu einem kabelgebundenen Transfer von Informationen ist die Übermittlung mittels elektromagnetischer Wellen jedoch deutlich anfälliger hinsichtlich äußerer Einflüsse. So können externe Störungen die Signale überlagern oder geographische Begebenheiten und im Signalweg liegende Vegetation die Übertragung beeinflussen und beeinträchtigen. Daraus können fehlerhafte Empfangsdaten bzw. im schlimmsten Fall überhaupt keine verwertbaren Nutzdaten resultieren und somit die Verbindung zu dem unbemannten Fluggerät verloren gehen.

Bei der Führung und Überwachung eines UAVs durch einen menschlichen Operateur ist dieser jedoch auf die Funkverbindung angewiesen, um weiterhin mit dem Fluggerät interagieren zu können oder entsprechende Telemetrie- oder auch Missionsnutzlastinformationen bzw. Sensordaten zu empfangen. Neben der aktuellen Auslastung des Datenlinks im Hinblick auf dessen Bandbreite (Menge der maximal übertragbaren Daten) ist die aktuelle Integrität (erlaubt Rückschlüsse auf etwaige EMV Probleme bzw. die maximale Reichweite der Funkverbindung) der Verbindung eine wichtige Information, welche dem Bediener zur Verfügung gestellt werden sollte. Zudem ist durch entsprechende Mechanismen sicherzustellen, dass empfangene, fehlerbehaftete Daten identifiziert und nachfolgend verworfen bzw. wiederhergestellt werden. Auch essentielle Nachrichten von der Kontrollstation zum UAV und vice versa müssen zuverlässig, innerhalb der physikalischen und technischen Rahmenbedingungen übertragen werden.

In den folgenden Abschnitten soll das für die Kommunikation zwischen der Bodenkontrollstation und dem UAV implementierte Protokoll vorgestellt werden, welches für die Umsetzung der oben genannten Anforderungen verwendet wurde.

5.6.1 Datenfunk-Protokoll

Im Allgemeinen beschreibt die Bezeichnung *Protokoll* eine Vereinbarung zwischen einem Sender und einem Empfänger, das durch Regeln die Syntax, Semantik und Synchronisation des Datenaustauschs definiert [Sharp, 2008].

Protocol: A set of rules and formats (semantic and syntactic) which determines the communication behavior of entities in the performance of functions. [International Standards Organisation, 1984]

Das implementierte ARQ-Protokoll (engl.: Automatic Repeat reQuest – Automatische Wiederholungsanfrage) gewährleistet die sichere Übertragung der Daten zwischen der Bodenkontrollstation und dem Fluggerät durch ggf. nötige Sendewiederholungen. Der Empfänger kann durch Fehlererkennung die aufgetretenen Übertragungsfehler identifizieren und dies dem Sender über einen Rückkanal mitteilen, wodurch gestörte bzw. nicht empfangene Nachrichten solange erneut übertragen werden, bis sie den Empfänger ohne Fehler erreichen. Die Fehlererkennung basiert auf einer zyklischen Redundanzprüfung (engl.: Cyclic Redundancy Check – CRC), bei der Prüfwerte auf Basis der Nutzdaten errechnet und mit übertragen werden. Auf Empfangsseite wird ebenfalls eine Prüfsumme ermittelt und mit der Empfangenen verglichen. Abhängig des verwendeten CRC-Verfahrens kann bei Übereinstimmung mit einer hohen Wahrscheinlichkeit davon ausgegangen werden, dass die empfangenen Nutzdaten valide übertragen wurden. Die Berechnung des CRC-Werts basiert auf einer Polynomdivision, wobei die Folge der einzelnen Bits der übertragenen Daten als dyadisches Polynom (Polynom des Dualsystems) betrachtet wird. Das verwendete Polynom ist in Formel 2 dargestellt und wird auch bei dem HDLC-Netzwerk- (engl.: High Level Data Link Control), dem X.25- und dem XModem-Protokoll für die Erkennung von Übertragungsfehlern verwendet.

$$x^{16} + x^{12} + x^5 + 1$$

Formel 2. Verwendetes dyadisches CRC-16 (CRC-CCITT) Polynom für die zyklische Redundanzprüfung

Zur besseren Übersicht wird in den folgenden Abbildungen dieses Abschnitts der Initiator für die Übertragung von Daten als *Sender* und die Gegenstelle als *Empfänger* bezeichnet. Diese bestehen *jeweils* aus einem Paar Datenfunkmodems (vgl. Abschnitt 5.4.2.4) und stellen damit – in Kombination – einen Transceiver (engl.: *Transmitter* + *Receiver* – Kombiniertes Sender und Empfänger) dar, der sowohl Daten versenden als auch empfangen kann.

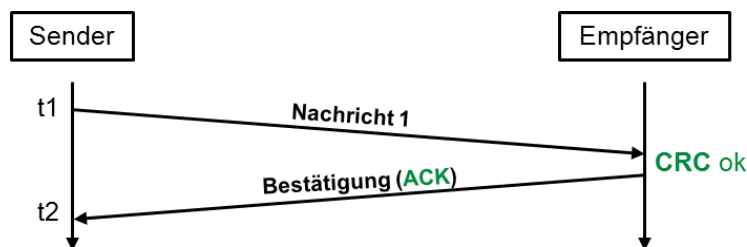


Abbildung 5-28. Nominelle Übertragung von Daten zwischen dem Sender und dem Empfänger

Der Normalfall einer Übertragung ist in Abbildung 5-28 dargestellt. Nach dem Versand einer Nachricht (enthält die Nutzdaten und die senderseitig errechnete Prüfsumme) zum Zeitpunkt *t1* wird diese vom Empfänger empfangen und auf Basis der Nutzdaten erneut, nach dem gleichen Verfahren wie auf Senderseite, eine Prüfsumme berechnet. Sind beide Werte gleich, so schickt der Empfänger eine Bestätigungsnachricht (engl.:

Acknowledgement – ACK) zurück an den Sender und quittiert damit den validen Empfang der Daten (Zeitpunkt t_2).

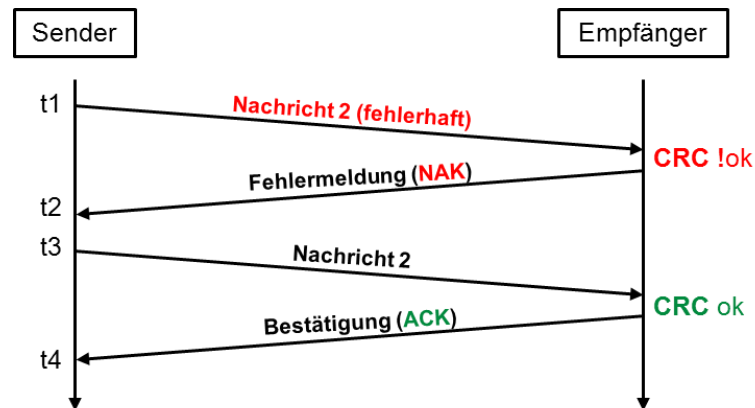


Abbildung 5-29. Erneuter Versand einer Nachricht, wenn Übertragungsfehler erkannt werden

Sind im Gegenzug die Nutzdaten jedoch fehlerhaft, so kann dies die Empfangsseite aufgrund einer nicht übereinstimmenden CRC-Prüfsumme identifizieren und durch Senden einer Fehlerrückmeldung (engl.: Negative Acknowledgement – NAK) dem Sender mitteilen, welcher daraufhin einen erneuten Versand dieser Nachricht initiiert. Ist der wiederholte Versand der Nachricht fehlerfrei, so wird der erfolgreiche Empfang der Daten bestätigt und der Transfer dieser Nachricht ist abgeschlossen (vgl. Abbildung 5-29).

Geht eine versendete Nachricht während der Übertragung gänzlich verloren bzw. wird vom Empfänger aufgrund von Empfangsproblemen durch beispielsweise Abschattung überhaupt nicht wahrgenommen, so wartet der Sender eine vordefinierte Zeitperiode auf die Bestätigungsnachricht des Empfängers. Bleibt diese jedoch aus, so wird bei Zeitüberschreitung (engl.: Timeout) ein erneuter Versand der Nachricht veranlasst (vgl. Abbildung 5-30).

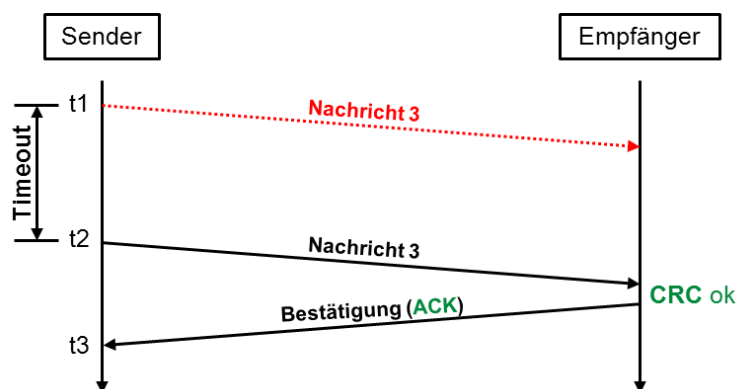


Abbildung 5-30. Erneuter Versand einer Nachricht nach einer vordefinierten Zeitperiode, wenn diese nicht durch den Empfänger bestätigt wird

Im folgenden Abschnitt soll die Implementierung des ARQ-Protokolls und das proprietäre Datenformat näher beschrieben und erläutert werden, über welches die

Kommunikation zwischen der Bodenkontrollstation und dem unbemannten Fluggerät abgewickelt wird.

5.6.2 Implementierung

Das im vorigen Abschnitt beschriebene ARQ-Protokoll wurde für den Softwareprozess COMC (vgl. Abbildung 5-4) umgesetzt, wobei die Datenkommunikation auf Byteebene von einem Softwaremodul abgewickelt wird, welches im Rahmen einer C++ Klasse implementiert wurde.

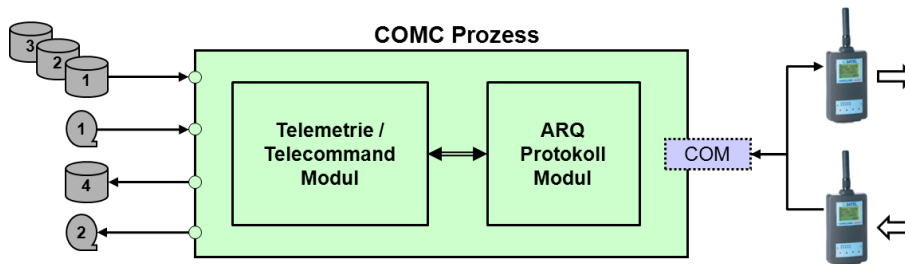


Abbildung 5-31. Modularer Aufbau des Softwareprozesses COMC

Abbildung 5-31 zeigt den schematischen Aufbau des Softwareprozesses COMC mit zwei hauptsächlichen Softwaremodulen. Das rechte Modul (ARQ-Protokoll-Modul) übernimmt dabei die Datenkommunikation mit der seriellen Schnittstelle (COM), die Kodierung als auch die Dekodierung der Datenkommunikation nach dem ARQ-Prinzip. Das zweite Modul ist für das Zusammentragen der Daten anderer Softwareprozesse (mittels der Interprozesskommunikation, DK1-3 vgl. Abbildung 5-31), wie zum Beispiel von dem FCS und den Treibern der IMU und dem Sensorboard, für die Erstellung und den nachfolgenden Versand von Telemetrienachrichten, verantwortlich. Des Weiteren werden empfangene Nachrichten dekodiert und per Queue (Q2, vgl. Abbildung 5-31) an andere Prozesse weitergeleitet (beispielsweise der FCS-Prozess für die Umschaltung von Betriebsarten durch den Operator, vgl. Abbildung 5-4). Sollen Nachrichten von anderen Prozessen (z.B. Statusmeldungen der ACU für den Operator) verschickt werden, so können diese dem COMC-Prozess über eine Queue (Q1, vgl. Abbildung 5-31) übermittelt werden. Informationen über den aktuellen Zustand der Datenfunkverbindung sowie die momentane Auslastung und die errechnete Integrität werden anderen Prozessen mittels eines Datakeys (DK4) zur Verfügung gestellt.

Für die Übertragung von Nachrichten wurde ein proprietäres, byteorientiertes Datenformat mit variabler Länge definiert, wodurch die Größe eines Datenpakets an die Menge der Nutzdaten angepasst werden kann. Zudem müssen keine Vorkehrungen bei der Übertragung zwischen verschiedenen Rechner- und Softwarearchitekturen getroffen werden, um dem Problem der unterschiedlichen Byte-Reihenfolge (engl.: Endianness) von Datentypen zu begegnen, welche mehr als ein Byte Speicherplatz benötigen (z.B. einige der ordinalen Datentypen der Programmiersprache C: short, word, int, float, double, ...). Neben den eigentlichen Nutzdaten – im Folgenden Nachricht genannt – werden zusätzlich noch sieben Bytes für ein Datenpaket benötigt – deren Position und Bedeutung zeigt Tabelle 8. Der verwendete Datentyp der einzelnen Bytes entspricht einer vorzeichenlosen Ganzzahl (C-Datentyp: unsigned char – 8 bit).

Byte-Position	Byte-Bedeutung
1, 2	Datenpaket-Header
3	Nachrichtenlänge
4	Fortlaufender Nachrichtenindex
5	Nachrichten-ID
6-n	Nutzdaten
n+1	CRC 16 – MSB (engl.: Most significant byte)
n+2	CRC 16 – LSB (engl.: Least significant byte)

Tabelle 8. Datenformat

Der Anfang eines Datenpakets wird über dessen Header (Bytes 1 und 2) identifiziert. Dieser definiert auch die Versandart. Soll eine Nachricht verschickt werden, ohne dass eine Bestätigung der Nachricht benötigt wird (im Folgenden unidirektional – UNI – genannt, da keine Quittierung der Nachricht erfolgt und demnach auch keine Rückübertragung von Daten stattfindet), so werden die ersten beiden Bytes mit folgenden Werten belegt:

```
Byte 1:  0x55  `U`  „Unidirectional
Byte 2:  `0x50  `P`  Package“
```

Die Header-Bytes von Nachrichten, für die eine Bestätigung (im Folgenden ARQ genannt) benötigt wird, werden im Gegenzug mit folgenden Werten belegt:

```
Byte 1:  0x41  `A`  "Arq
Byte 2:  `0x50  `P`  Package“
```

Die dritte und letzte Art von Nachrichten sind die Rückmeldungen der Gegenstelle, welchen entweder den erfolgreichen Empfang der Nachricht quittieren oder einen Fehler bei der Übertragung signalisieren. Der Header sieht dann für eine Bestätigungsnachricht wie folgt aus:

```
Byte 1:  0x41  `A`  "Acknow-
Byte 2:  `0x50  `K`  ledge“
```

Und für den Fall einer Fehlermeldung:

```
Byte 1:  0x41  `N`  "Negative
Byte 2:  `0x50  `K`  Acknowledge“
```

Das dritte Byte des Datenpakets bezeichnet die Länge der Nachricht (Nutzdaten), wodurch die Größe des Datenpakets bestimmt und die Position der CRC-Bytes errechnet werden kann.

Auf Byte-Position vier wird eine vom ARQ-Protokoll-Modul erzeugte und für jede Nachricht fortlaufend inkrementierte Nummer (Wertebereich 0-255) übertragen, die den aktuellen Nachrichtenindex darstellt. Dieser wird einerseits für die Bestätigung von

Nachrichten verwendet und andererseits für die Errechnung der Datenlinkintegrität benötigt. Wird beispielsweise eine Nachricht mit der Indexnummer 38 erfolgreich und fehlerfrei empfangen, so setzt sich die Bestätigungsnachricht wie folgt zusammen:

Byte 1:	0x41	`A`	Header-Byte 1
Byte 2:	0x50	`K`	Header-Byte 2
Byte 3:	0x31	`1`	Länge der Nachricht = 1 Byte
Byte 4:	0x34	52d	Fortl. Index der Gegenstelle
Byte 5:	0x00	NUL	Nachrichten-ID = ACK Nachricht
Byte 6:	0x26	38d	Bestätigung der Nachricht mit dem Index 38
Byte 7:	0x3D		CRC-16 MSB
Byte 8:	0x1E		CRC-16 LSB

Der interne Aufbau des ARQ-Protokoll-Moduls ist in Abbildung 5-32 dargestellt. Über die Eingangsqueue Q1 empfangene oder von dem Telemetrie/Telecommand Modul erzeugte Nachrichten werden, abhängig von deren Versandart (UNI – unsicher oder ARQ – gesichert), in verschiedenen Zwischenspeichern (engl.: Buffer – Puffer) abgelegt. Hierdurch können von mehreren Prozessen Nachrichten an den COMC-Prozess übergeben werden, welcher nachfolgend deren Versand vornimmt.

Für den ungesicherten Versand von Nachrichten wird nur ein einzelner Puffer benötigt (UNI-Sendepuffer, vgl. Abbildung 5-32). Sind hier Nachrichten für den Versand hinterlegt, so werden diese mit einem Nachrichtenindex versehen (vgl. Byte 4 in Tabelle 8), die Prüfsumme (CRC) berechnet, über die Datenfunkmodems verschickt und nachfolgend aus dem Puffer entfernt.

Sollen im Gegenzug jedoch Nachrichten gesendet werden, für die eine Empfangsbestätigung benötigt wird, so werden diese nach deren Aufnahme in einem ARQ-Eingangspuffer zwischengespeichert. Der Wertebereich des Nachrichtenindex beschränkt sich auf Werte zwischen 0 und 255 und wird für die Quittierung einer erfolgreich empfangenen Nachricht verwendet. Gesetzt den Fall, der ARQ-Sendepuffer würde mehr als 255 Nachrichten beinhalten, so hätten mindestens zwei Nachrichten den gleichen Nachrichtenindex, wodurch bei der Bestätigung der Nachricht durch ein ACK von dem Empfänger der Index nicht mehr eindeutig einer Nachricht zugeordnet werden kann. Demnach werden nur Nachrichten aus dem ARQ-Eingangspuffer in den ARQ-Sendepuffer verschoben, wenn dessen Nachrichtenanzahl geringer als 255 ist. Beim Verschieben einer Nachricht wird diese ebenfalls mit einem Nachrichtenindex versehen und nach dem Versand dieses Datenpakets, zusammen mit einem Zeitstempel, in den ARQ-Puffer verschoben (vgl. Abbildung 5-32). Wird nun auf dem Rückkanal eine Bestätigungsnachricht empfangen, so wird die Nachricht mit dem gleichen Index aus dem ARQ-Puffer gelöscht (vgl. Nachricht B/4 in Abbildung 5-32). Wird eine Fehlermeldung für eine Nachricht empfangen oder bleibt eine Rückmeldung aus (vgl. Nachricht A/3), so wird diese nach dem Ablauf einer vordefinierten Zeitperiode (im Fall der vorliegenden Arbeit beträgt die Zeitspanne 150ms) wiederum in den ARQ-

Eingangspuffer verschoben, um erneut versendet zu werden (vgl. Nachricht A/5 in Abbildung 5-32).

Von der Gegenstelle empfangene Nachrichten werden ebenfalls in einem Puffer (Empfangspuffer vgl. Abbildung 5-32) abgelegt und können über eine Member-Funktion der C++-Klasse nach dem FIFO-Prinzip (engl.: First in first out) ausgelesen werden. Im Falle von empfangenen ARQ Nachrichten wird zudem eine Bestätigungsnachricht (ACK rx, vgl. Abbildung 5-32) mit dem entsprechenden Index für den Versand erzeugt und in dem UNI-Sendepuffer abgelegt.

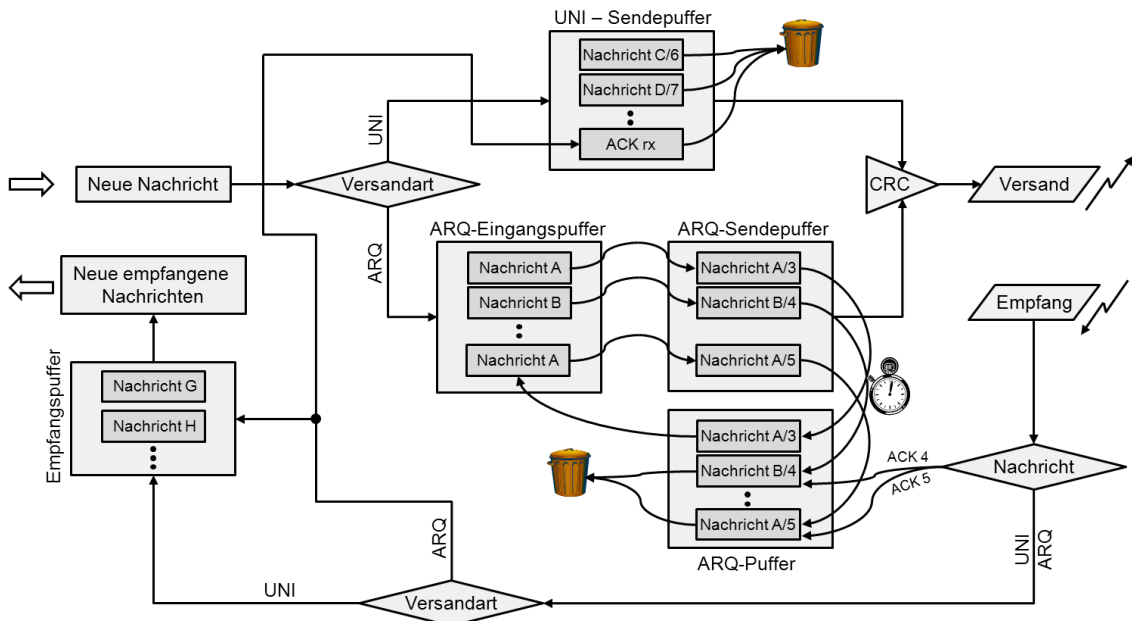


Abbildung 5-32. Mechanismus des ARQ-Protokolls

Die Berechnung der aktuellen Auslastung (engl.: Usage – Nutzung der verfügbaren max. Übertragungsrate) des Datenlinks zeigt Formel 3. Hierfür werden die Anzahl der versendeten Bytes (*NumberOfTXBytes*) für eine Zeitperiode von einer Sekunde aufaddiert und durch die maximal übertragbare Datenrate (*MaxDateLinkRate*) dividiert. Im Fall der verwendeten Sateline 3AS(d) Modems beträgt diese laut Datenblatt 19200 bps (engl.: Bits per second), also 2400 Bytes pro Sekunde. Da dieser Wert nur theoretisch erreichbar ist, wurde dieser um 15%, auf einen realistischeren Wert von 2040 Bytes pro Sekunde reduziert.

$$\eta_{DL} = \frac{NumberOfTXBytes}{MaxDataLinkRate} \quad [\%]$$

Formel 3. Berechnung der aktuellen Auslastung des Datenlinks

Für die Errechnung der Integrität des Datenlinks (vgl. Formel 4) ist die hauptsächliche Informationsquelle der Index der einzelnen empfangenen Nachrichten. Aufgrund der fortlaufenden Zahl können verloren gegangene Nachrichten ($n_{verloren}$, vgl. Formel 4), durch Sprünge zwischen erfolgreich empfangenen Nachrichten ($n_{erfolgreich}$, vgl. Formel 4) identifiziert und gezählt werden. Wird eine Nachricht mit dem Index 6 und

nachfolgend eine Nachricht mit dem Index 8 erfolgreich empfangen, so kann die Nachricht mit dem Index 7 als verloren angesehen werden.

Die Berechnung der Datenlinkintegrität wird – ebenso wie die aktuelle Auslastung – für eine Zeitperiode von einer Sekunde berechnet. Es kann davon ausgegangen werden, dass die Gegenstelle entweder nicht erreichbar oder nicht eingeschaltet ist, wenn innerhalb dieses Zeitintervalls keine Nachrichten erfolgreich empfangen wurden.

$$\eta_{Int} = 1 - \frac{n_{erfolgreich}}{n_{erfolgreich} + n_{verloren}} \quad [\%]$$

Formel 4. Berechnung der Integrität des Datenlinks

Da jeweils in der Bodenk Kontrollstation als auch auf dem UAV ein COMC-Prozess ausgeführt wird, gibt es insgesamt jeweils zwei Werte für die Datenlinkauslastung und – integrität für den entsprechenden Transceiver. Die Ermittlung der Datenlinkauslastung kann direkt von jedem Softwaremodul selbst bestimmt werden, wohingegen die Integritätsrechnung von der entsprechenden Gegenstelle durchgeführt wird.

Unter Normalbedingungen lagen bei den durchgeführten Testflügen die Werte der Datenlinkintegrität bei rund 95-100% und die Datenlinkauslastung im Bereich von 20-35%, abhängig von der vom Operateur wählbaren Übertragungsrate (1-10Hz) von Telemetrieinformationen.

Das folgende Kapitel spezifiziert die durchgeführte Evaluation des implementierten Konzepts eines *Knowledge Configured Vehicle*. Neben der Umgebung werden sowohl das Szenario und die Mission vorgestellt als auch die Ergebnisse aus Simulator- und realen Flugversuchen präsentiert.

6 Evaluierung

Basierend auf dem in Kapitel 3 beschriebenen Konzept eines *Knowledge Configured Vehicle* wurde auf Grundlage der kognitiven Systemarchitektur COSA eine *künstlich kognitive Einheit – ACU* (vgl. Abschnitt 2.4.2) entwickelt, deren a-priori Wissensmodelle in Kapitel 4 detailliert wurden. Die Integration und die Einbettung dieser Softwarekomponente in die Avionikarchitektur des entwickelten UAV-Demonstrator BIG I sowie die zur Fernführung des UAVs benötigte Bodenkontrollstation wurden im vorigen Kapitel erläutert. Dieser Funktionsprototyp soll nun, im Rahmen einer einfachen Aufklärungsmission, im Hinblick auf das Konzept und die Implementierung evaluiert werden.

Zunächst wird in Abschnitt 6.1 die Evaluierungsumgebung vorgestellt, die sich aus der eigentlichen Mission (vgl. Abschnitt 6.1.1) und der aufgebauten Hardware-in-the-Loop-Simulation (vgl. Abschnitt 6.1.2) zusammensetzt. Die Ergebnisse, Flugdaten und das gezeigte Verhalten der kognitiven Einheit bei einem real durchgeführten Versuchsflug werden darauffolgend in Abschnitt 6.2. beschrieben. Abschließend werden Simulationsergebnisse präsentiert (vgl. Abschnitt 6.3), die sowohl einen Vergleich mit den realen Flugergebnissen erlauben (vgl. Abschnitt 6.3.1) als auch einen Anwendungsfall (engl.: Use case) behandeln, in dem ein technischer Defekt der Avionikstromversorgung simuliert wird (vgl. Abschnitt 6.3.2).

6.1 Evaluierungsumgebung

6.1.1 Evaluierungsmission

Die Erprobung der entwickelten KCV-ACU findet anhand einer einfachen Aufklärungsmission statt, bei der ein Sensor für einen bestimmten Flugabschnitt auf einen erdfesten Punkt (engl.: Point of interest – POI) ausgerichtet werden soll. Der Ablauf der Mission umfasst insgesamt sechs Wegpunkte sowie den Zielpunkt und ist in Abbildung 6-1 dargestellt.

Nach dem Start soll das UAV den ersten Wegpunkt anfliegen, anschließend der Flugroute folgen und währenddessen das Heading (Formelzeichen Ψ) des Hubschraubers nach dem Bahnazimutwinkel (Formelzeichen χ) ausrichten (Nase des Hubschraubers zeigt in Flugrichtung), bis Wegpunkt 3 erreicht wird. Zwischen den Wegpunkten 3 und 4 ist eine Überwachung und Aufklärung des POIs vorgesehen, was durch eine entsprechende Drehung der Hochachse des UAVs in Richtung des erdfesten Punkts erreicht werden soll, um den bildgebenden Sensor an der Vorderseite des UAVs entsprechend zu orientieren. Nach dem Passieren des Wegpunkts 4 soll wiederum das Heading dem Bahnazimutwinkel entsprechen und weiter der Flugroute bis zum letzten Wegpunkt gefolgt werden. Wegpunkts 6 bezeichnet den Landepunkt und damit auch das Ende der Mission.

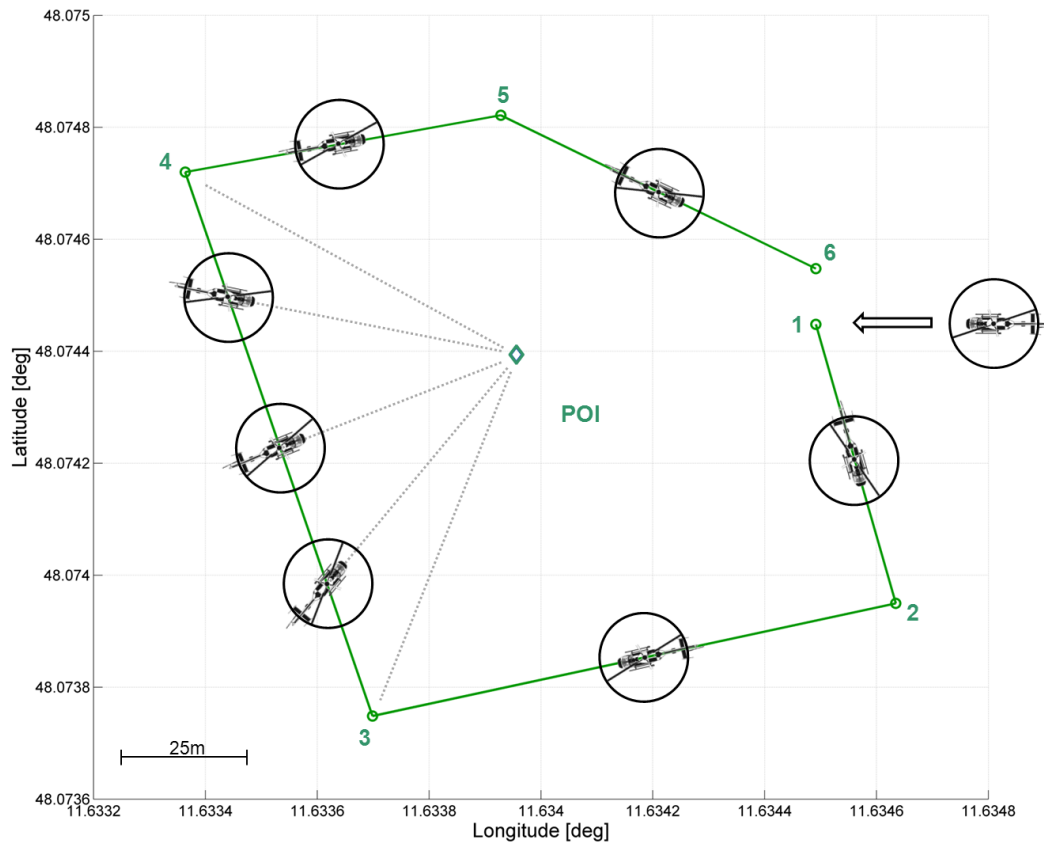


Abbildung 6-1. Vorgeplante Flugroute der Aufklärungsmission

Die Details und Einzelheiten der Mission – wie etwa die Koordinaten der einzelnen Wegpunkte, des POIs und in welchem Wegpunktintervall der POI überwacht werden soll – werden von dem UAV-Operator festgelegt und in ein textbasiertes Format und Dokument übersetzt, welches in Abbildung 6-2 dargestellt ist.

```

Waypoint route:
WP 1/6 48.07444810 11.63449140 25.000000
WP 2/6 48.07394970 11.63463430 25.000000
WP 3/6 48.07374860 11.63369870 25.000000
WP 4/6 48.07472010 11.63336390 25.000000
WP 5/6 48.07482160 11.63392810 25.000000
WP 6/6 48.07454770 11.63449140 25.000000
Start point index: 1
Land point index: 6
Point of interests:
POI 1/1 48.0743294 11.6340246
Ground surveillance:
GS 1/1 start index 3
GS 1/1 end index 4
    
```

Abbildung 6-2. Missionsdokument für die Evaluationsmission der KCV-ACU

Zu Anfang wird die Flugroute (engl.: Waypoint route) und die einzelnen Koordinaten mit den Indizes der Wegpunkte definiert. Die Spezifizierung eines Wegpunkts beginnt in einer Zeile mit der Abkürzung *WP*, gefolgt von dem Index des Wegpunkts und die gesamte Anzahl an Wegpunkten, getrennt durch einen Schrägstrich. Abschließend folgt die Festlegung der lateralen Position des Wegpunkts in WGS84-Koordinaten (Latitude,

Longitude im Gradmaß) sowie die Flughöhe über dem Startplatz des UAVs in Metern. Der erste Wegpunkt der Mission (engl.: Start point index) sowie der Punkt an dem gelandet werden soll (engl.: Land point index), werden durch Angabe der entsprechenden Wegpunktindizes bestimmt. Abschließend wird die Position des erdfesten Punkts (Abkürzung *POI* in Abbildung 6-2, ebenfalls mit einem Index und der Anzahl der POIs versehen) und der Flugabschnitt – durch Angabe von Wegpunktindizes – definiert, in welchem die Überwachung (Abkürzung *GS* in Abbildung 6-2) durchgeführt werden soll.

Die durch dieses Dokument formulierte Mission wird zur Laufzeit an die KCV-ACU geschickt, durch einen Parser innerhalb des Input-Servers aufbereitet und an die Missionsmanagementfunktionen, repräsentiert durch das missionsspezifische Wissen (vgl. Abschnitt 4.6), übermittelt. Ziel der MMF ist es, nach der Umschaltung auf die automatische Flugführung durch den Sicherheitspiloten, die gegebene Mission durch Nutzung der von dem Fluggerät und dessen Automation gestellten Fähigkeiten, die Aufgabe zu erfüllen. An dieser Stelle sei darauf hingewiesen, dass im Fall der vorliegenden Arbeit durch die Missionsmanagementfunktionen auch Fähigkeiten angefordert werden, welche zu manchen Zeitpunkt nicht verfügbar sind. Dies ist einerseits damit begründet, dass bei der Wissensentwicklung und -modellierung der Schwerpunkt auf das maschinenspezifische Wissen gelegt wurde und somit das missionsspezifische Wissen über nur geringe Planungsfähigkeiten verfügt (vgl. Abschnitt 4.6). Andererseits soll das Verhalten und die Reaktion der ACU aufgezeigt werden, wenn derzeit nicht verfügbare Fähigkeiten angefordert werden.

6.1.2 HIL-Simulation

Die Entwicklung der Wissensmodelle der KCV-ACU und Teile der Evaluierung des entwickelten KCV-Funktionsprototypen wurden im Umfeld der aufgebauten HIL Simulationsumgebung durchgeführt, die dem Demonstratorumgebungskonzept aus Abbildung 5-1 entspricht. Neben einer Nachbildung des Operateur-Arbeitsplatzes der Bodenkontrollstation wurden sämtliche Hardwareelemente, wie zum Beispiel der Missionsrechner (MMC, vgl. Abschnitt 5.4.2.2), der Flugregelungsrechner (FCC, vgl. Abschnitt 5.4.2.2), die Datenfunkverbindung (vgl. Abschnitt 5.4.2.4 und 5.6) sowie die entsprechenden Eingabegeräte (Touchscreen-Monitore und Joystick) in die Simulationsumgebung integriert. Somit werden ausschließlich die Flugdynamik und die daraus resultierenden Sensordaten von der Reflex-Simulation (vgl. Abschnitt 5.3) generiert. Der restliche gesamtheitliche Aufbau und die Verschaltung der Soft- und Hardwarekomponenten ist identisch mit der Architektur und der Anordnung der Bestandteile der realen Demonstratorumgebung (vgl. Kapitel 5).

6.2 Evaluierung „Real World“

Die realen Versuchsflüge wurden, mit einer entsprechenden Aufstiegsgenehmigung des Luftfahrtbundesamt Süd, auf einem abgesperrten Areal der UniBwM durchgeführt und die einzelnen Positionen und Abstände der Wegpunkte nach zwei Gesichtspunkten ausgewählt. Einerseits sollte das Gelände eben und ohne Hindernisse sein, damit der

Sicherheitspilot in der Lage ist, dem Fluggerät zu Fuß zu folgen, um die visuelle Lageerkennung des UAVs sicherzustellen. Dieser Umstand limitiert zudem die maximale Fluggeschwindigkeit des UAV-Demonstrators auf 2m/s, damit der Sicherheitspilot noch in der Lage ist diesem zu folgen. Andererseits musste die Länge der Flugroute, im Hinblick auf die verfügbaren Ressourcen (Stromversorgung und Kerosin) des UAVs, mit entsprechenden Sicherheitsreserven ausgelegt werden, um eine risikoarme Durchführung der Mission zu gewährleisten.

Im Folgenden soll nun der durchgeführte Flugversuch zur Evaluierung der entwickelten und implementierten KCV-ACU (vgl. Kapitel 4) in einzelnen Flugabschnitten näher betrachtet und erläutert werden. Der Zeitpunkt der Umschaltung durch den Sicherheitspiloten von der manuellen Stabilisierung auf die automatische Stabilisierung und Flugführung durch das FCS und die KCV-ACU bezeichnet den Zeitpunkt Null ($t = 0$). Des Weiteren werden im folgenden Verlauf dieses Kapitels die Missionsmanagementfunktionen (repräsentiert durch das missionsspezifische Wissen) als Missionsmanagementsystem (MMS – Teilsystem des kognitiven Systems) bezeichnet.

Für die Durchführung der Evaluationsmission wurde der Demonstrator BIG I anfangs mittels externer Stromquelle am Boden versorgt und für den Versuchsflug vorbereitet. Dabei wurden zunächst alle Avionik-Systeme und die Sensorik eingeschaltet sowie die Softwarekomponenten des FCC hochgefahren, bis der Primärsensor, die NAV420 IMU, eine stabile Navigationslösung lieferte (GPS-Lock). Nachfolgend wurde die KCV-ACU auf dem MMC gestartet und die externe Avionikstromversorgung auf die interne Bordstromversorgung umgestellt. Der Start des UAVs erfolgte manuell durch den Sicherheitspilot beim Zeitpunkt $t = -82,4s$. Tabelle 9 zeigt die wichtigsten Ereignisse und internen Zustände der ACU während dieser Phase.

<i>Zeit [s]</i>	<i>Ereignis</i>	<i>Beschreibung</i>
-228,0	ACU Start	Start der KCV-ACU auf dem MCC.
-227,7	Initialisierungsphase	Verarbeitung der Eingangsdaten des Input-Servers durch die Umweltmodelle der ACU
-227,2	Systemmodell im situativen Wissen	Sämtliche Avionikkomponenten und deren Anforderungen des UAV-Demonstrators wurden auf Basis der Umweltmodelle instanziiert und das Systemmodell des Fluggerätes im situativen Wissen der ACU erstellt. Ergebnis: <i>Erfolgreich</i> .
	Prüfung des Systemzustands	Die ACU prüft den Zustand der Avionik- und Softwarekomponenten, im Hinblick auf deren im maschinenspezifischen Wissen definierten Nominalzustände. Ergebnis: <i>Alle Systeme arbeiten nominal</i> .
	Abstraktion der Fähigkeiten des UAVs	Auf Basis des maschinenspezifischen Wissens werden die durch die Automation und Nutzlast möglichen Fähigkeiten des UAVs abstrahiert.
-227,0	Interpretation des Missionsdokuments	Das Missionsdokument (vgl. Abbildung 6-2) wird von der ACU interpretiert.

Evaluierung

-226,7	Programmierung der Flugroute durch die ACU	Die KCV-ACU initiiert die Programmierung der einzelnen Wegpunkte der Flugroute der Mission in das FMS. Status: <i>Erfolgreich durchgeführt.</i>
-225,3	Missionsbereit	Die ACU meldet die Betriebsbereitschaft für sich selbst und das Fluggerät für die Durchführung der Mission.
-225,1	Anfordern von Verhalten	Das MMS fordert das Verhalten <i>AUTOMATIC_START</i> durch deren Eintragung in die KCV-Schicht an. Ergebnis: <i>Fähigkeit nicht verfügbar.</i>
-82,4	Manueller Start	Das Fluggerät wird manuell durch den Sicherheitspiloten gestartet und auf eine sichere Flughöhe gebracht.
0,0	Umschaltung auf Automatik	Der Sicherheitspilot übergibt die Steuerautorität an das FCS.

Tabelle 9. KCV-ACU Evaluationsflug – Initialisierungsphase, manueller Start des UAVs bis zur Umschaltung auf Automatik

Nach dem Starten der ACU beginnt diese, die Eingangsdaten des Input-Servers, durch Verarbeitung der Wissensmodelle, zu interpretieren ($t = -227,7s$). Auf Basis der im situativen Wissen angelegten Instanzen der einzelnen Systeme und Komponenten baut sie auf der einen Seite eine Repräsentation des Systemmodells auf (vgl. Abschnitt 4.5.2) und überprüft auf der anderen Seite deren aktuellen Zustand, mit dem im maschinenspezifischen Wissen hinterlegten Nominalzustand.

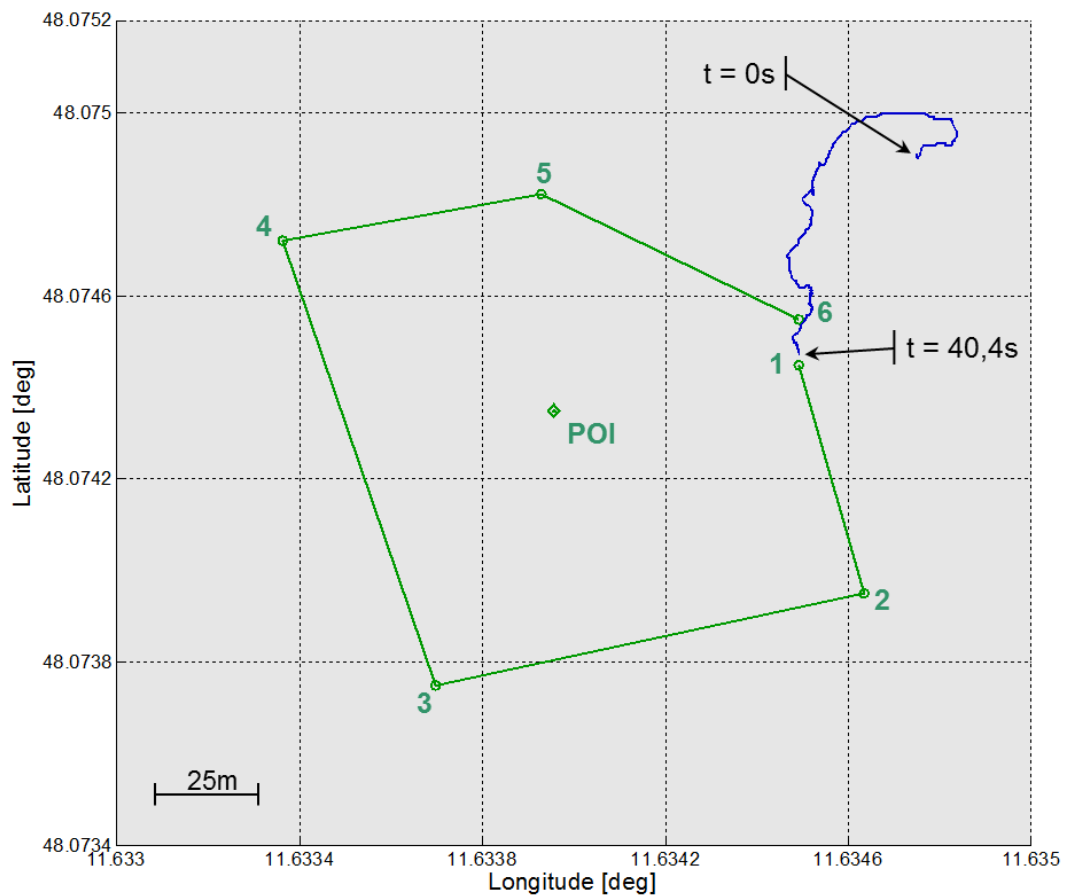


Abbildung 6-3. KCV-ACU Evaluationsflug – Flugabschnitt ab Umschaltung auf Automatik bis zum Erreichen des ersten Wegpunkts

Dieser ist für alle Subsysteme nominal, da zu diesem Zeitpunkt die Subsysteme des UAVs bereits komplett hochgefahren wurden und die Initialisierung der Sensoren abgeschlossen ist. Zudem werden die Fähigkeiten des UAVs, abhängig von der verfügbaren Automation und Nutzlastkomponenten, abstrahiert (Zeitpunkt $t = -227,2s$) und in der KCV-Schicht eingetragen. Nach dem Empfang des Missionsdokuments (Zeitpunkt $t = -227s$) startet das missionsspezifische Wissen mit der Interpretation des zu erfüllenden Auftrags sowie die Vorbereitung des Flugmanagementsystems durch Programmieren der einzelnen Koordinaten der Wegpunktliste (Zeitpunkt $t = -226,7s$).

Nachfolgend signalisiert die ACU ihre Betriebs- und Missionsbereitschaft dem menschlichen Operateur. Sie beginnt – auf zuvor gegebene Anweisung des Bedieners und basierend auf dem instanziierten Ziel *ACCOMPLISH_MISSION* (vgl. Abschnitt 4.6) – mit der Durchführung der Mission durch das Anfordern des gewünschten Verhaltens *AUTOMATIC_START* (Zeitpunkt $t = -225,1s$). Diese Fähigkeit wurde jedoch aus Sicherheitsgründen deaktiviert und nicht freigegeben (vgl. Abschnitt 5.4.5.1). Als Reaktion darauf informiert die ACU den menschlichen Operateur, dass das angeforderte Verhalten nicht erzeugt werden kann. Des Weiteren könnte durch zusätzliches Wissen eine mögliche Empfehlung an den Bediener initiiert werden – wie beispielsweise, dass der Startvorgang manuell vorgenommen werden muss. Dadurch kann die ACU mit der Durchführung der Mission beginnen, was jedoch im Rahmen der vorliegenden Arbeit nicht umgesetzt wurde. Nach Abschluss der letzten Systemüberprüfungen am Boden wurde der manuelle Start durch den Sicherheitspiloten zum Zeitpunkt $t = -82,4s$ durchgeführt, das UAV auf eine sichere Flughöhe von rund 30m über dem Startplatz gebracht und zum Zeitpunkt $t = 0$ die Steuerautorität an das FCS bzw. die KCV-ACU übergeben.

<i>Zeit [s]</i>	<i>Ereignis</i>	<i>Beschreibung</i>
0,0	Anfordern von Verhalten durch das MMS	Das MMS fordert das Verhalten <i>FOLLOW_WP_ROUTE</i> und <i>FLIGHT_PATH</i> durch deren Eintragung in die KCV-Schicht an.
0,3	Aktivierung der Betriebsart <i>Att2</i> des FCS	Aktivierung der Betriebsart <i>Att2</i> des FCS durch das maschinenspezifische Wissen. (Ausrichtung des Bahnazimutwinkels in Flugrichtung)
0,7	Ressourcenprüfung	Prüfung der vorhandenen Ressourcenmengen (Kerosin und Stromversorgung) für das angeforderte Verhalten <i>FOLLOW_WP_ROUTE</i> . Ergebnis: <i>Genügend Ressourcen vorhanden</i> .
	Aktivierung der Betriebsarten <i>Pos_HI</i> und <i>Pos_VI</i> des FCS	Aktivierung der Betriebsart <i>Pos_HI</i> und <i>Pos_VI</i> des FCS durch das maschinenspezifische Wissen. (Anflug des Wegpunkts eins auf dem Track, definiert durch Wegpunkt 6 und 1).
40,4	Wegpunkt 1 erreicht	Der Wegpunkt 1 wurde erreicht.

Tabelle 10. KCV-ACU Evaluationsflug – Umschaltung auf Automatik bis Wegpunkt 1

Abbildung 6-3 zeigt den Flugabschnitt ab der Umschaltung auf Automatik bis zum Erreichen des ersten Wegpunkts, während Tabelle 10 die wesentlichen Ereignisse dieser Flugphase aufzeigt. Dieser Autoritätswechsel wird von der kognitiven Einheit registriert

und das MMS beginnt im gleichen Zeitschritt mit der Arbeit zur Erfüllung der gegebenen Mission. Hierfür werden von dem MMS anfangs zwei Handlungsalternativen zum Anfordern von gewünschtem Verhalten vorgeschlagen (engl.: Propose). Diese sind einerseits die Fähigkeit *FOLLOW_WP_ROUTE* (Folgen der Wegpunktliste) und andererseits die Fähigkeit *FLIGHT_PATH* (Ausrichtung des Bahnazimutwinkels in Flugrichtung).

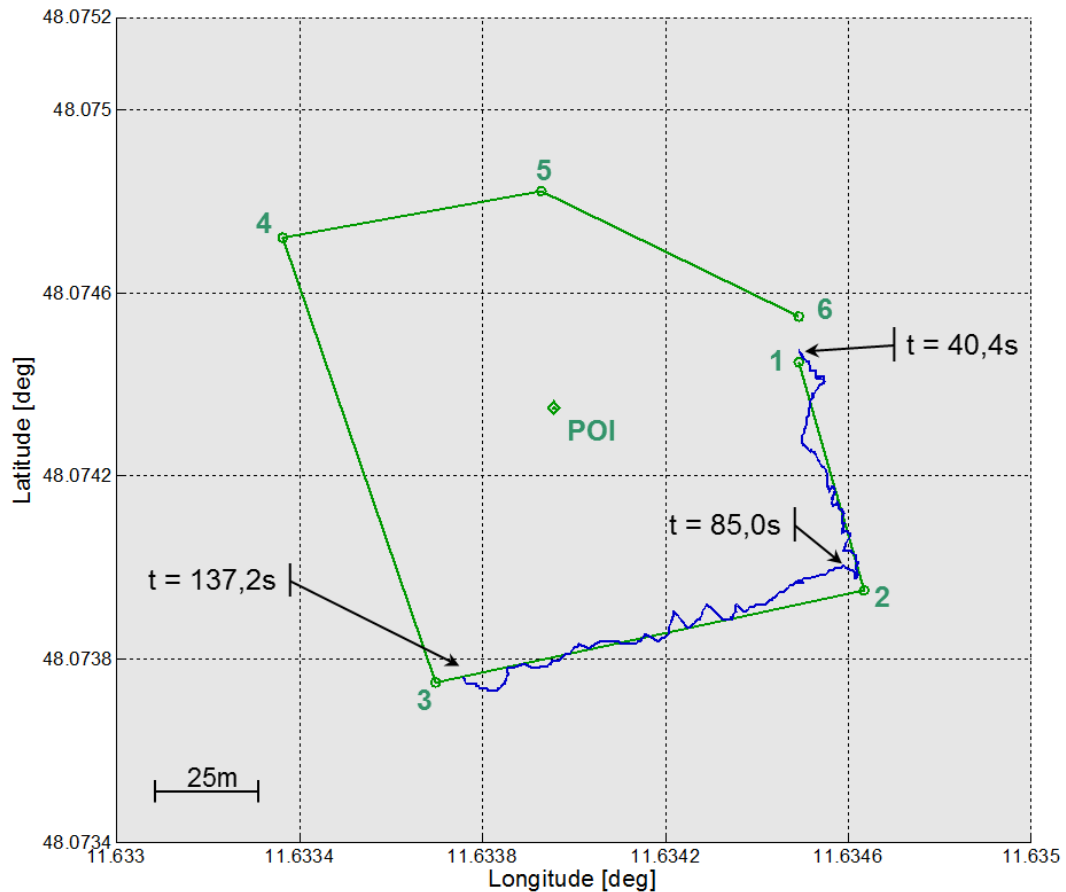


Abbildung 6-4. KCV-ACU Evaluationsflug – Flugabschnitt Wegpunkt 1 bis Wegpunkt 3

Da es für diese Fähigkeiten keine weiteren Handlungsalternativen gibt, werden diese von dem MMS, durch entsprechende Anweisungsmodelle, als gewünschtes Verhalten in der KCV-Schicht eingetragen. Aufgrund der Verfügbarkeit beider Fähigkeiten übernimmt nachfolgend das maschinenspezifische Wissen direkt die Aktivierung der Att2 Betriebsart des FCS. Vor der Konfiguration der Betriebsarten Pos_H1 und Pos_V1 (vgl. Abschnitt 5.4.5.1) werden noch die verfügbaren Ressourcen geprüft (vgl. Abschnitt 4.7). Unter Verwendung von hinterlegtem Wissen und Berechnungsverfahren – auf Basis der Entladekurven des Avionikkakus in dem Function-Server – können sowohl die noch verfügbare Zeitdauer für eine sicher Strom- und Kerosinversorgung errechnet als auch durch die vorgegebene Fluggeschwindigkeit die Dauer für das Abfliegen der einzelnen Wegpunkte abgeschätzt und anschließend verglichen werden. Aufgrund genügend gefundener Ressourcen wird das FCS/FMS, durch das maschinenspezifische Wissen, entsprechend konfiguriert. Der UAV-Demonstrator

beginnt mit dem automatischen Anflug auf Wegpunkt 1, welcher zum Zeitpunkt $t = 40,2s$ erreicht wird.

Die ACU symbolisiert, mittels eines Umweltmodells (mit den möglichen Zuständen *REACHED* und *TRANSIT*), ob ein Wegpunkt erreicht wurde bzw. noch angefliegen wird. Dies erfolgt durch eine Berechnung des Abstands zwischen der aktuellen Position des UAVs und dem aktiven Wegpunkt. Hier wurden Grenzen von sechs (Erreichen des Wegpunkts) und zehn (Verlassen des Wegpunkts) Metern festgelegt, welche als Kreise um den jeweiligen Wegpunkt angesehen werden können. Liegt die Position des Fluggerätes innerhalb des kleineren Kreises, so interpretiert die ACU den Wegpunkt als erreicht (vgl. kleiner Abstand zwischen der letzten Position des UAVs und Wegpunkt 1 in Abbildung 6-3).

Nach Erreichen des ersten Wegpunkts trifft die ACU die Entscheidung den nächsten Wegpunkt zu aktivieren und mit dem Folgen der Flugroute fortzufahren. Hierfür werden wiederum ein entsprechendes Ziel, eine Handlungsalternative und das zugehörige Anweisungsmodell instanziiert. Diese bewirken über die KCV-Schicht das Weiterschalten des Wegpunkts im FMS, durch das maschinenspezifische Wissen. Abbildung 6-4 zeigt den lateralen Verlauf des von dem UAV zurückgelegten Flugwegs (engl.: Ground Track) zwischen den Wegpunkten 1 und 3.

Zum Zeitpunkt $t = 85,0s$ erkennt das MMS, dass sich das Fluggerät an Wegpunkt 2 befindet und sorgt für die Aktivierung von Wegpunkt 3 im Flugmanagementsystem. Aufgrund des zuvor bereits angeforderten Verhaltens *FOLLOW_WP_ROUTE* ist das FCS immer noch in der für diesen Flugabschnitt passenden Konfiguration. Demnach bedarf es keiner weiteren Interaktionen des MMS mit der KCV-Schicht, bis auf die Weiterschaltung des aktuellen Wegpunkts.

<i>Zeit [s]</i>	<i>Ereignis</i>	<i>Beschreibung</i>
85,2	Wegpunkt 2 erreicht	Der Wegpunkt 2 wurde erreicht. Weiterschaltung auf Wegpunkt 3 durch das MMS.
137,2	Wegpunkt 3 erreicht	Der Wegpunkt 3 wurde erreicht. Weiterschaltung auf Wegpunkt 4 durch das MMS.
	Anfordern von Verhalten durch das MMS	Das MMS fordert das Verhalten <i>TARGET_POINT</i> durch dessen Eintragung in die KCV-Schicht an.
	Aktivierung der Betriebsart <i>Att3</i> des FCS	Aktivierung der Betriebsart <i>Att3</i> des FCS durch das maschinenspezifische Wissen. (Ausrichtung des Bahnazimutwinkels auf den POI).
208,7	Wegpunkt 4 erreicht	Der Wegpunkt 4 wurde erreicht. Weiterschaltung auf Wegpunkt 5 durch das MMS.
	Anfordern von Verhalten durch das MMS	Das MMS fordert das Verhalten <i>FLIGHT_PATH</i> durch dessen Eintragung in die KCV-Schicht an.
	Aktivierung der Betriebsart <i>Att2</i> des FCS	Aktivierung der Betriebsart <i>Att2</i> des FCS durch das maschinenspezifische Wissen. (Ausrichtung des Bahnazimutwinkels in Flugrichtung)

Tabelle 11. KCV-ACU Evaluationsflug –Wegpunkt 3 bis Wegpunkt 4

Die eigentliche Aufgabe dieser Mission ist die Überwachung eines erdfesten Punkts, auf den ein bildgebender Sensor an der Vorderseite des UAVs ausgerichtet werden soll, um Bilder des POI aus verschiedenen Richtungen aufzunehmen. Dies soll nach der Definition der Mission zwischen den Wegpunkten 3 und 4 erfolgen. Der geflogene Ground track ist zusammen mit dem Heading des UAVs für diesen Flugabschnitt in Abbildung 6-5 dargestellt. Tabelle 11 listet die einzelnen, relevanten Ereignisse auf.

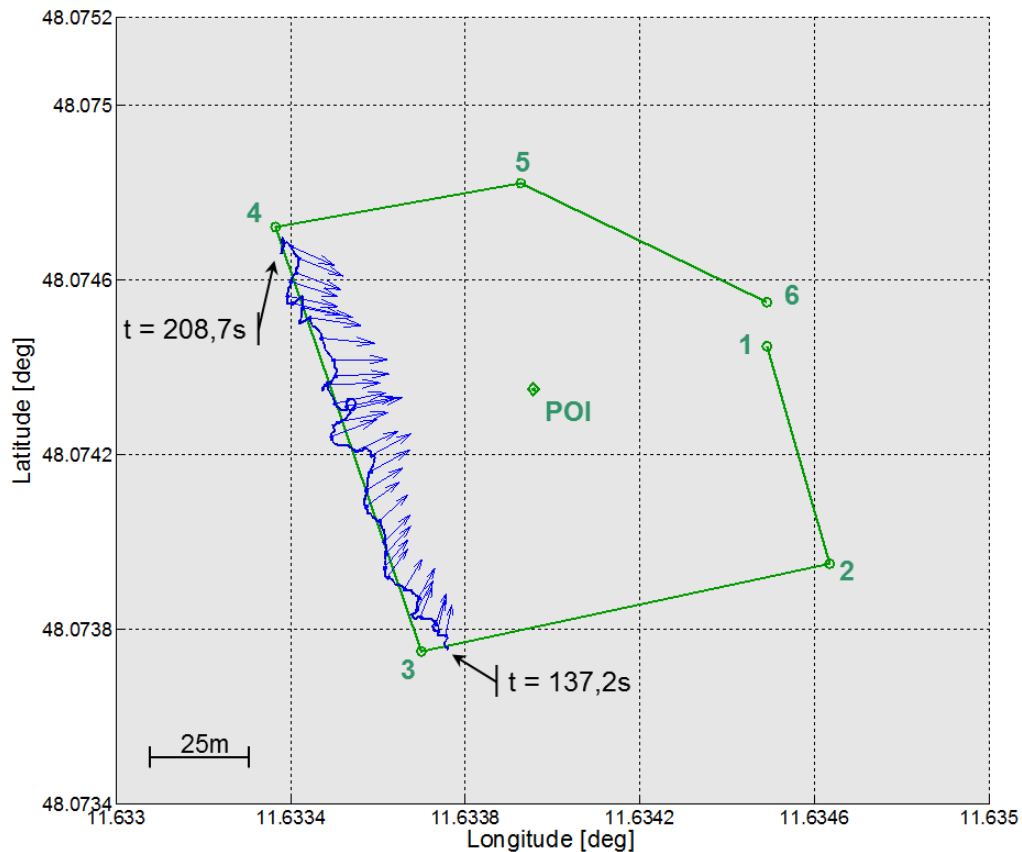


Abbildung 6-5. Flugabschnitt Wegpunkt 3 bis Wegpunkt 4 mit Ausrichtung des Heading des UAVs auf den erdfesten Punkt

Ab Erreichen des Wegpunkts 3 (Zeitpunkt $t = 137,2s$) fordert das MMS für die Erfüllung der Überwachungsaufgabe die Fähigkeit *TARGET_POINT* an, um die Front des Hubschraubers, mit dem bildgebenden Sensor, auf den POI auszurichten. Da für die Aktivierung der entsprechenden Betriebsart des FCS keine Ressourcenprüfung durchgeführt werden muss, übernimmt das maschinenspezifische Wissen direkt dessen Konfiguration. Zusätzlich müsste an Wegpunkt 3 die ACU eigentlich ein Verhalten, wie beispielsweise *ENABLE_CAMERA* anfordern. Es gibt jedoch für die kognitive Einheit, aufgrund der derzeitigen Verschaltung und Integration der Kamera und des zugehörigen Videofunks, keine Möglichkeit dieses System zu beeinflussen. Die Aktivierung der Kamera und des Videosenders kann nur manuell über die Schalttafel (vgl. Abschnitt 5.4.3) durch einen Menschen vorgenommen werden.

Nach Erreichen des Wegpunkts 4 ist die Überwachungsaufgabe des POI abgeschlossen. Das Heading des Hubschraubers kann wieder in Flugrichtung orientiert werden, was das

MMS mit der Anforderung des Verhaltens *FLIGHT_PATH* von der KCV-Schicht zum Zeitpunkt $t = 208,7s$ umsetzt. Zudem veranlasst das MMS die Weiterschaltung des FMS auf Wegpunkt 5.

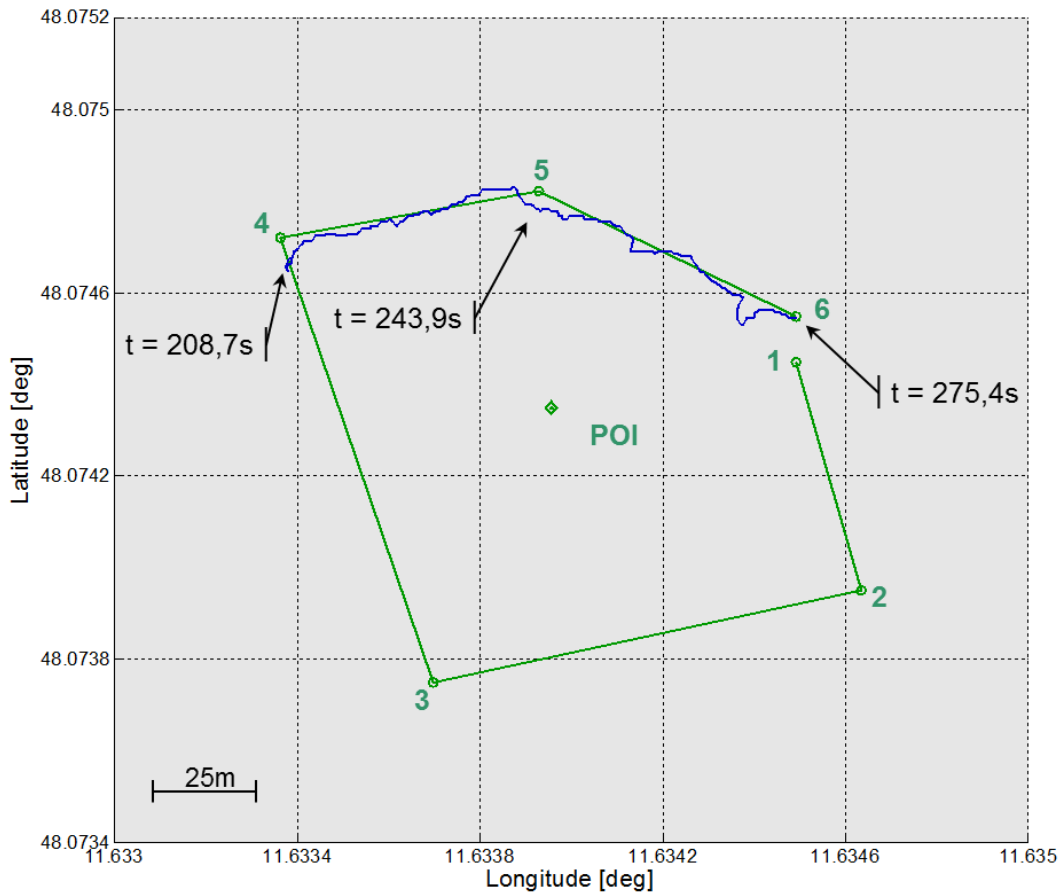


Abbildung 6-6. Flugabschnitt Wegpunkt 4 bis Wegpunkt 6

Der letzte Flugabschnitt der Evaluationsmission mit dem zugehörigen, zurückgelegten Ground Track des Fluggerätes ist in Abbildung 6-6 und die wichtigsten Ereignisse in Tabelle 12 dargestellt. Wegpunkt 5 wird zum Zeitpunkt $t = 243,9s$ erreicht. Aufgrund der Tatsache, dass der finale Wegpunkt der Flugroute angefliegen wird, entscheidet das MMS das Verhalten *TRACK_WP* anzufordern. Dieses sorgt für das Anfliegen auf dem durch Wegpunkt 5 und 6 definierten Track und das anschließende Verweilen am aktiven Wegpunkt des FMS. An dieser Stelle sei darauf hingewiesen, dass hierfür keine Änderung der FCS-Betriebsarten notwendig ist, da sich die Fähigkeiten *TRACK_WP* und *FOLLOW_WP_ROUTE* ausschließlich durch das Weiterschalten der Wegpunkte unterscheiden, welche durch das MMS initiiert werden (vgl. Abschnitt 5.4.5.1).

Bei der Ankunft des Fluggerätes am letzten Wegpunkt (Zeitpunkt $t = 275,4s$) interpretiert das MMS die Mission als erfüllt. Wie im Missionsdokument definiert, bezeichnet Wegpunkt 6 auch den Landepunkt (vgl. Abbildung 6-2) mit der Folge, dass das MMS das Verhalten *AUTOMATIC_LAND* anfordert. Diese Fähigkeit ist jedoch, ebenso wie die Fähigkeit *AUTOMATIC_START* (vgl. Anfang des Abschnitts), sicherheitshalber deaktiviert worden und kann demnach nicht angefordert bzw. durch das maschinenspezifische Wissen aktiviert werden.

<i>Zeit [s]</i>	<i>Ereignis</i>	<i>Beschreibung</i>
243,9	Wegpunkt 5 erreicht	Der Wegpunkt 5 wurde erreicht und das MMS informiert den Bediener über den Anflug auf den letzten Wegpunkt der Flugroute. Weiterschaltung auf Wegpunkt 6 durch das MMS.
	Anfordern von Verhalten durch das MMS	Das MMS fordert das Verhalten <i>TRACK_WP</i> durch dessen Eintragung in die KCV-Schicht an. (Der aktuelle Wegpunkt wird angefliegen, jedoch erfolgt nach Erreichen des Wegpunkts keine Weiterschaltung durch das MMS).
	Betriebsarten POS_H1 und POS_V1 des FCS sind bereits aktiv	Das Verhalten <i>TRACK_WP</i> und <i>FOLLOW_WP_TRACK</i> sind im Hinblick auf die Betriebsarten des FCS identisch: → <i>Keine Änderung der FCS Betriebsart</i>
275,4	Wegpunkt 6 erreicht	Der letzte Wegpunkt der Flugroute wurde erreicht.
	Anfordern von Verhalten durch das MMS	Das MMS fordert das Verhalten <i>AUTOMATIC_LAND</i> durch dessen Eintragung in die KCV-Schicht an. Ergebnis: <i>Fähigkeit nicht verfügbar.</i>
295,2	Umschaltung auf Manuell	Der Sicherheitspilot übernimmt die Steuerautorität des UAV-Demonstrators.
396,1	Manuelle Landung	Das Fluggerät wird manuell durch den Sicherheitspiloten gelandet.

Tabelle 12. KCV-ACU Evaluationsflug –Wegpunkt 4 bis Wegpunkt 6 mit anschließender manueller Landung durch den Sicherheitspiloten

Zum Zeitpunkt $t = 295,2s$ übernimmt der Sicherheitspilot die Kontrolle über das Fluggerät und führt anschließend eine manuelle Landung durch, welche zum Zeitpunkt $t = 396,1s$ abgeschlossen ist.

Während der Durchführung informierte die KCV-ACU den menschlichen Operateur kontinuierlich über den Verlauf und die wichtigsten Ereignisse der Mission – wie der Abschluss des Aufbaus des Systemmodells, die Prüfung der Nominalzustände der Avionikkomponenten, das Anfordern von Verhalten und die anschließende Konfiguration der Automation durch das maschinenspezifische Wissen sowie das Erreichen von Wegpunkten. Im Map-Display des TermIO-Prozesses der BKS werden dem Bediener die KCV-ACU Nachrichten und Informationen der KCV-ACU in Textform für einige Sekunden dargestellt. Abbildung 6-7 zeigt einen solchen Screenshot des TermIO-Prozesses bei einer simulierten Evaluationsmission zum Zeitpunkt, als das Fluggerät gerade den Wegpunkt 3 erreicht. Eine Nachricht setzt sich aus zwei Teilen zusammen. Am Anfang steht das Modul, welches der Initiator der Nachricht ist, und es folgt der eigentliche Informationsgehalt. Eine solche Nachricht kann wie folgt lauten:

„KCV-Mission: Requesting heading behavior: TargetPoint“

Diese Nachricht wurde vom MMS (*„-Mission“*) veranlasst, um dem Operateur über die Anforderung (*„Requesting“*) des gewünschten Verhaltens (*„TargetPoint“*) in Kenntnis zu setzen. Das Kürzel *„KCV-“* vor der Bezeichnung des Wissensmoduls beschreibt, dass die Nachricht über die KCV-Schicht versendet wurde (vgl. Block *„Informationen und Warnungen“* in Abbildung 3-3). Die nachfolgende Konfiguration der FCS-Betriebsart wird von dem *„AP“* Modul vorgenommen. Es stellt das

Wissensmodell des Autopiloten (FCS), als Teil des maschinenspezifischen Wissens, dar (vgl. Abschnitt 4.5.1) und informiert den Bediener über die Aktivierung der entsprechenden Fähigkeit („Heading mode TargetPoint enabled“).



Abbildung 6-7. Anzeige von Nachrichten der KCV-ACU im Map-Display des BKS TermIO-Prozesses

Abschließend für diesen Abschnitt zeigt Abbildung 6-8 den gesamten Ground track des durch den UAV-Demonstrator BIG I zurückgelegten Flugwegs im Automatik-Modus, stabilisiert durch das FCS und geführt durch die KCV-ACU. Die Gesamtflugdauer für diese automatisch durchgeführte Mission betrug 4 Minuten und 55,2 Sekunden. Die Positionsabweichungen des UAVs von den einzelnen durch die Wegpunkte definierten Tracks sind auf noch nicht optimal eingestellte FCS-Parameter des Navigation Loops, atmosphärische Störungen wie Wind sowie die GPS Messungenauigkeit und -rauschen zurückzuführen. Auch führt das Abstandskriterium für das Erkennen ob ein Wegpunkt erreicht wurde zu einem Anfangsfehler der Trackingaufgabe, der durch das FCS ausgeregelt werden muss.

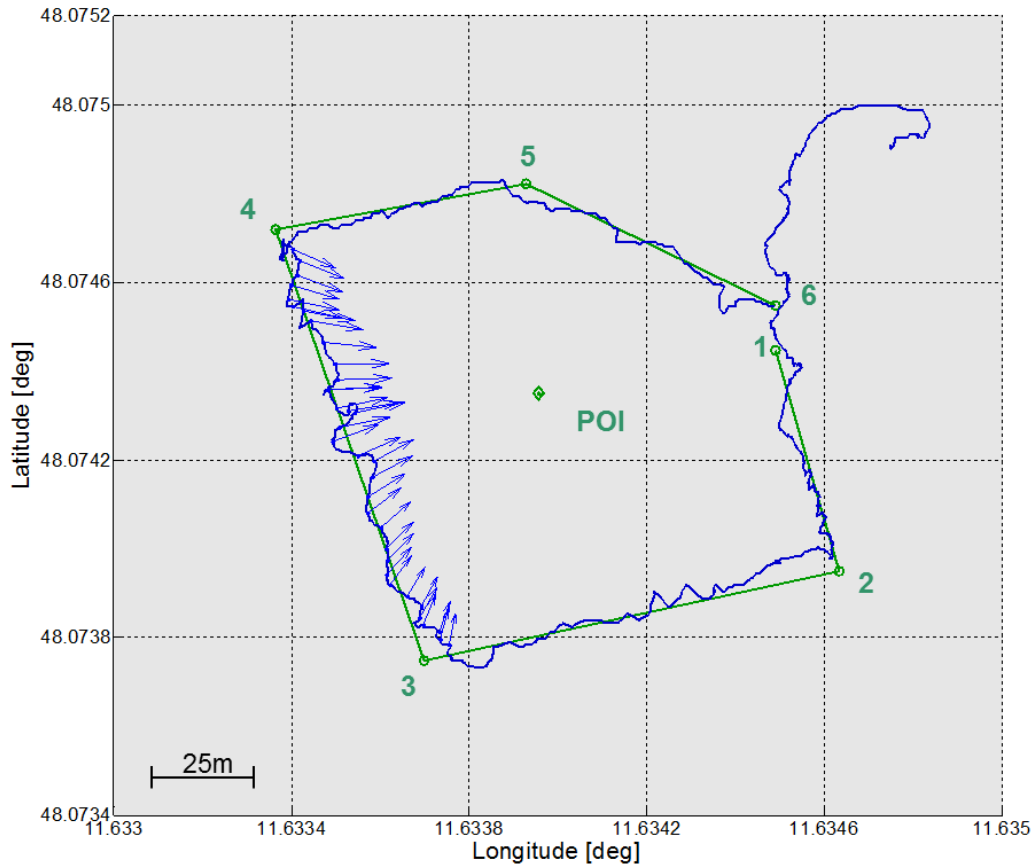


Abbildung 6-8. Gesamter Ground-Track des KCV-Versuchsflugs mit Headingvektoren zwischen den Wegpunkten 3 und 4

6.3 Evaluierung in der Simulation

6.3.1 Durchführung der Evaluationsmission in der Simulation

Die in Abschnitt 6.2 beschriebene KCV-ACU Evaluationsmission wurde ebenfalls in der HIL-Simulationsumgebung durchgeführt, um die gewonnen Flugdaten und das Verhalten der ACU vergleichen zu können. Abbildung 6-9 zeigt den zurückgelegten Flugweg des simulierten UAVs und die Ausrichtung des Bahnazimutwinkels zwischen den Wegpunkten 3 und 4 (der treppenartige Verlauf des Flugwegs ist auf Quantisierungsfehler zurückzuführen, welche bei der Positionsberechnung des Sensormodells innerhalb der DLL (vgl. Abschnitt 5.3) auftreten, jedoch für die Erprobung und Auslegung der FCS-Algorithmen unerheblich waren). Im Gegensatz zu den real durchgeführten Flugversuchen zeigt sich, dass die Bahnfolge des Fluggerätes auf den einzelnen Tracks zwischen den Wegpunkten erheblich genauer ist. Dies kann einerseits auf bessere Sensordaten der Simulation – weniger Störungen durch atmosphärische Bedingungen – und andererseits auf besser eingestellte FCS-Parameter und -Verstärkungsfaktoren zurückgeführt werden (Für die Simulation und das reale Fluggerät wurden zwar ähnliche, aber dennoch unterschiedliche Werte eingestellt).

Die Abweichung des Flugwegs vom Track, wie beispielsweise ab der Weiterschaltung an Wegpunkt 2 und das langsame Annähern an den Track (zwischen Wegpunkt 2 und 3, vgl. Abbildung 6-9), wurde in Kauf genommen, da es aufgrund der Positions-

bestimmung mittels GPS bei den Realflügen, durch die Bewegung des Fluggerätes oder durch EMV-Beeinflussung durch andere Avionikgeräte, zu Rauschen und Meßfehlern in den Positionsinformationen kommen kann. Würde hier versucht werden, die Ablage zu aggressiv auszuregeln, so wäre eine erhöhte Aktuatoraktivität und damit ein Anstieg der mechanischen Belastung des Hubschraubers die Folgen.

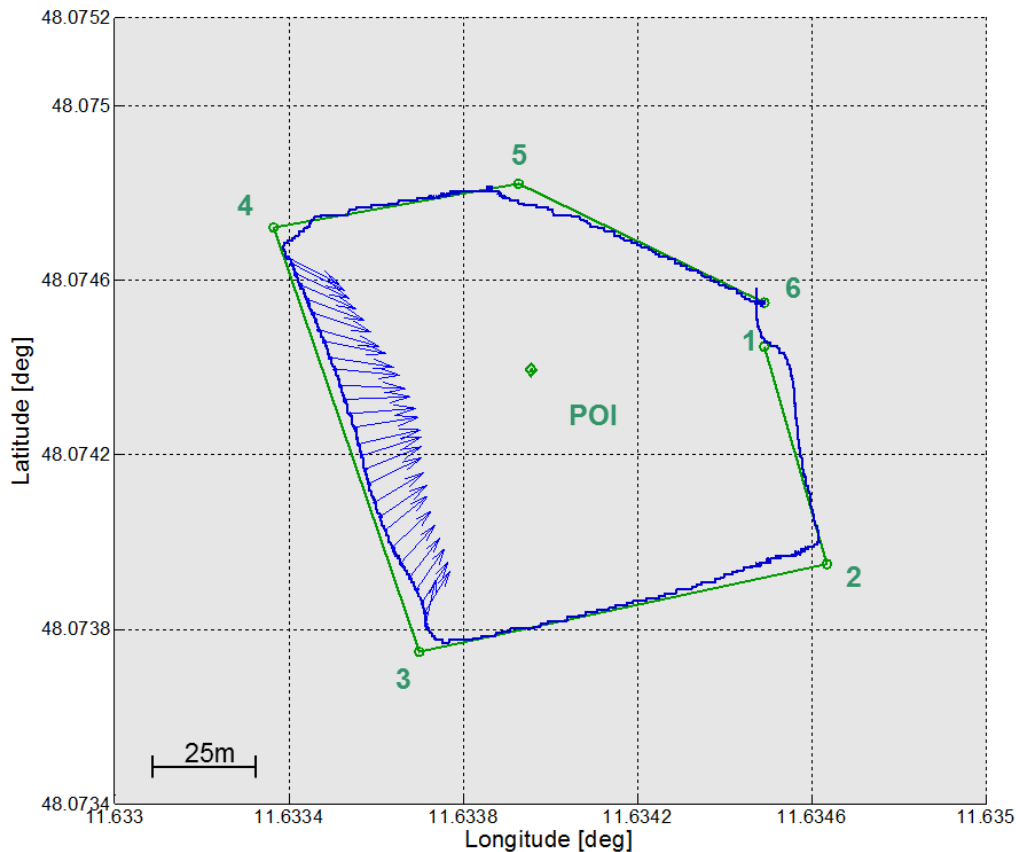


Abbildung 6-9. Durchführung der Evaluationsmission in der Simulationsumgebung

Bei der Durchführung der Evaluationsmission mit dem Versuchsträger BIG I wurde die ACU erst gestartet, als bereits sämtliche Avioniksysteme sich im Nominalzustand befanden und demnach waren ab dem produktiven Betrieb der ACU alle Fähigkeiten des Fluggerätes sofort verfügbar. Im Rahmen des in Abbildung 6-9 dargestellten Simulatorversuchs soll nun die Initialisierungsphase, das Prüfen der Avioniksysteme und der Sensoren und die sich daraus ableitende Verfügbarkeit der einzelnen Fähigkeiten des UAVs näher betrachtet werden. Die Startphase des Primärsensors (IMU) wird ebenfalls in der Simulation nachgestellt, da die Trägheitsnavigationseinheit eine bestimmte Zeitdauer benötigt, um die einzelnen Sensoren zu vermessen und den internen Filteralgorithmus zu initialisieren. Der aktuelle Status des simulierten Sensors wird dabei, wie bei der realen Komponente, durch einen Zahlenwert beschrieben. Dieser ermöglicht das Treffen einer Aussage über die derzeitige Genauigkeit und Zuverlässigkeit der Sensordaten (Algorithm accuracy, vgl. [Crossbow Technology, 2007]). Die möglichen Zustände und deren Bedeutung zeigt Tabelle 13.

Status-nummer	Zustand	Beschreibung
3	INIT	Initialisierung des Sensors
2	LOW_AHRS	AHRS-Lösung mit geringer Genauigkeit verfügbar
1	AHRS	Navigationslösung mit geringer und AHRS-Lösung mit hoher Genauigkeit verfügbar
0	NAV	GPS verfügbar, volle Navigationslösung mit hoher Genauigkeit verfügbar

Tabelle 13. Mögliche Status Algorithmusgenauigkeit des Primärsensors IMU-NAV420

In der Simulation werden die einzelnen Status der IMU und des integrierten GPS-Empfängers chronologisch, gekoppelt an die Simulationszeit, an vorher definierten Zeitpunkten weitergeschaltet. Tabelle 14 listet die einzelnen Zeitpunkte und die zugehörigen Status.

Simulationszeit [s]	Zustand	Beschreibung
< 5	INIT	Kein GPS-Empfang (engl.: Unlocked) NAV420 Status: <i>INIT</i>
< 15	LOW_AHRS	Kein GPS-Empfang (engl.: Unlocked) NAV420 Status: <i>LOW_AHRS</i>
< 20	AHRS	Kein GPS Empfang (engl.: Unlocked) NAV420 Status: <i>AHRS</i>
< 25		GPS Empfang (engl.: Locked) NAV420 Status: <i>AHRS</i>
>= 25	NAV	GPS Empfang (engl.: Locked) NAV420 Status: <i>NAV</i>

Tabelle 14. Abfolge der simulierten Status der Algorithmusgenauigkeit des Primärsensors IMU-NAV420

Die Reflex-Simulation und die ACU werden fast zeitgleich gestartet. Die wichtigsten Ereignisse und Zustände der IMU, im Hinblick auf die Genauigkeit des Algorithmus, für die ersten Sekunden der Simulationszeit sind in Tabelle 15 zusammengefasst. Nach der Initialisierungsphase der KCV-ACU und dem Anlegen der entsprechenden Instanzen der Avionik- und Softwarekomponenten im situativen Wissen beginnt die KCV-ACU mit dem Aufbau des Systemmodells, was zum Zeitpunkt $t = 0,5s$ abgeschlossen ist. In dieser Phase befindet sich der simulierte Zustand der IMU noch in der Initialisierungsphase (Zustand *INIT*, vgl. Tabelle 13) und keine der durch die entsprechenden Wissensmodelle abstrahierten Fähigkeiten (vgl. Abschnitt 4.5.3) ist verfügbar. Bei Zeitpunkt $t = 5,6s$ ändert sich der Status der IMU auf den Zustand *LOW_AHRS*. Das hat jedoch keine Verfügbarkeitsänderung der Fähigkeiten zur Folge, da die Sensorwerte der IMU in dieser Phase noch immer ein geringes Maß an Genauigkeit/Zuverlässigkeit aufweisen. Der globale Zustand des Sensors wird von dem Umweltmodell *IMU* mit dem Attribut *Status* (vgl. Abschnitt 4.5.1) beschrieben, welches zu diesem Zeitpunkt dem Wert *DEGRADED* entspricht.

<i>Zeit [s]</i>	<i>Ereignis / Zustand der IMU</i>	<i>Beschreibung / Verfügbarkeit der Fähigkeiten</i>
0,5	Systemmodell im situativen Wissen erstellt.	Sämtliche Avionikkomponenten und deren Anforderungen des UAV-Demonstrators wurden auf Basis der Umweltmodelle instanziiert und das Systemmodell des Fluggerätes im situativen Wissen der ACU erstellt. Ergebnis: <i>Erfolgreich.</i>
	<u>IMU Zustand:</u> GPSStatus: Unlocked Algorithm Accuracy: INIT Status: DEGRADED	<u>Fähigkeit</u> <u>Verfügbarkeit</u> ATTITUDE_CMD: nicht verfügbar SPEED_CMD: nicht verfügbar FLY_TO_WP: nicht verfügbar TRACK_WP: nicht verfügbar FOLLOW_WP_ROUTE: nicht verfügbar HDG_COMMAND: nicht verfügbar FLIGHT_PATH: nicht verfügbar TARGET_POINT: nicht verfügbar
5,6	<u>IMU Zustand:</u> GPSStatus: Unlocked Algorithm Accuracy: LOW_AHRS Status: DEGRADED	<u>Fähigkeit</u> <u>Verfügbarkeit</u> ATTITUDE_CMD: nicht verfügbar SPEED_CMD: nicht verfügbar FLY_TO_WP: nicht verfügbar TRACK_WP: nicht verfügbar FOLLOW_WP_ROUTE: nicht verfügbar HDG_COMMAND: nicht verfügbar FLIGHT_PATH: nicht verfügbar TARGET_POINT: nicht verfügbar

Tabelle 15. KCV-ACU Simulationsevaluierung –Genauigkeit des Algorithmus der IMU und die Verfügbarkeitsbewertung der Fähigkeiten des Fluggerätes für die Simulationszeit bis sechs Sekunden

Ab dem Zeitpunkt $t = 15,6s$ (vgl. Tabelle 16) erkennt die ACU die Weiterschaltung des Status der IMU auf den Zustand *AHRS*. Sie kennzeichnet daraufhin die Fähigkeiten *ATTITUDE_CMD* (Lageregelung des Fluggerätes, vgl. Abschnitt 5.4.5.1) und *HEADING_CMD* (Ausrichten des Headings des Hubschraubers in eine vom Operateur definierte Richtung) als verfügbar, da für die korrespondierenden Betriebsarten des FCS die notwendigen Sensordaten (Raten- und Lageinformationen) des Fluggerätes vorhanden sind und keine Geschwindigkeits- bzw. Positionsinformationen benötigt werden. Das Vorhandensein von validen Positionsinformationen durch den GPS-Empfänger beginnt (simuliert) ab Zeitpunkt $t = 20,9s$, woraufhin die Fähigkeit *TARGET_POINT* (Ausrichtung des Steuerkurses des Hubschraubers auf einen erdfesten Punkt, vgl. Abschnitt 5.4.5.1) verfügbar wird. Durch Kenntnis der Koordinaten des POI und der eigenen Position kann die Richtung errechnet werden, auf welche in dieser Betriebsart das Heading des Fluggerätes orientiert wird. Zu diesem Zeitpunkt befindet sich der Status der IMU noch immer im Zustand *AHRS*, da der interne Filter der Trägheitsnavigationseinheit eine kurze Zeitspanne benötigt, um sich einzustellen.

<i>Zeit [s]</i>	<i>Zustand / Ereignis</i>	<i>Beschreibung / Verfügbarkeit der Fähigkeiten</i>	
15,6	<u>IMU Zustand:</u> GPSStatus: Unlocked Algorithm Accuracy: AHRS Status: DEGRADED	<u>Fähigkeit</u>	<u>Verfügbarkeit</u>
		ATTITUDE_CMD:	verfügbar
		SPEED_CMD:	nicht verfügbar
		FLY_TO_WP:	nicht verfügbar
		TRACK_WP:	nicht verfügbar
		FOLLOW_WP_ROUTE:	nicht verfügbar
		HEADING_CMD:	verfügbar
		FLIGHT_PATH:	nicht verfügbar
		TARGET_POINT:	nicht verfügbar
20,9	<u>IMU Zustand:</u> GPSStatus: Locked Algorithm Accuracy: AHRS Status: DEGRADED	<u>Fähigkeit</u>	<u>Verfügbarkeit</u>
		ATTITUDE_CMD:	verfügbar
		SPEED_CMD:	nicht verfügbar
		FLY_TO_WP:	nicht verfügbar
		TRACK_WP:	nicht verfügbar
		FOLLOW_WP_ROUTE:	nicht verfügbar
		HEADING_CMD:	verfügbar
		FLIGHT_PATH:	nicht verfügbar
		TARGET_POINT:	verfügbar

Tabelle 16. KCV-ACU Simulationsevaluierung –Genauigkeit des Algorithmus der IMU und die Verfügbarkeitsbewertung der Fähigkeiten des Fluggerätes für die Simulationszeit bis 21 Sekunden

Tabelle 17 zeigt den Zustand und die Ereignisse ab einer Simulationszeit >25 Sekunden, in der die simulierte IMU die volle Einsatzbereitschaft durch die Änderung ihres Status auf den Zustand *FULL_NAV* signalisiert (Zeitpunkt $t = 25,7s$). Ab dieser Phase wechselt der Wert des **IMU**-Klassenattributs *Status* zu *NOMINAL*. Ab diesem Zeitpunkt werden alle durch die KCV-ACU abstrahierten Fähigkeiten des Fluggerätes als verfügbar gekennzeichnet. An diesem Punkt sei angemerkt, dass die Fähigkeiten *ATTITUDE_CMD* und *SPEED_CMD* Sollgrößen benötigen, welche von einem Operateur mittels eines Joystick-Eingabegerätes vorgegeben werden können. Aufgrund der Auslegung der Evaluationsmission – welche zwar von dem Bediener definiert wird, aber nachfolgend keine weitere Interaktion mit diesem vorgesehen ist (abgesehen von den Statusmeldung der KCV-ACU) – werden diese Fähigkeiten bei der Durchführung der Mission von der KCV-ACU nicht verwendet.

Zum Zeitpunkt $t = 26,0s$ meldet die ACU den Nominalzustand sämtlicher Avioniksysteme und beginnt mit der Interpretation des Missionsdokuments und der Programmierung der Flugroute in das Flugmanagementsystem (Zeitpunkt $t = 26,6s$). Abschließend meldet die KCV-ACU die Einsatzbereitschaft der UAV und beginnt mit der Durchführung der Mission.

<i>Zeit [s]</i>	<i>Zustand / Ereignis</i>	<i>Beschreibung / Verfügbarkeit der Fähigkeiten</i>
25,7	<i>IMU Zustand:</i> GPSStatus: <i>Locked</i> Algorithm Accuracy: FULL_NAV Status: NOMINAL	<i>Fähigkeit</i> <i>Verfügbarkeit</i> ATTITUDE_CMD: <i>verfügbar</i> SPEED_CMD: <i>verfügbar</i> FLY_TO_WP: <i>verfügbar</i> TRACK_WP: <i>verfügbar</i> FOLLOW_WP_ROUTE: <i>verfügbar</i> HEADING_CMD: <i>verfügbar</i> FLIGHT_PATH: <i>verfügbar</i> TARGET_POINT: <i>verfügbar</i>
	Prüfung des Systemzustands	Die ACU prüft den Zustand der Avionik- und Softwarekomponenten, im Hinblick auf deren, im maschinenspezifischen Wissen definierten, Nominalzustände. Ergebnis: <i>Alle Systeme arbeiten nominal.</i>
26,0	Interpretation des Missionsdokuments	Das Missionsdokument (vgl. Abbildung 6-2) wird von der ACU interpretiert.
26,6	Programmierung der Flugroute durch die ACU	Die KCV-ACU initiiert die Programmierung der einzelnen Wegpunkte der Flugroute der Mission in das FMS des FCS. Status: <i>Erfolgreich durchgeführt.</i>
26,7	Missionsbereit	Die ACU meldet die Betriebsbereitschaft für die Durchführung der Mission.

Tabelle 17. KCV-ACU Simulationsevaluierung – Genauigkeit des Algorithmus der IMU und die Verfügbarkeitsbewertung der Fähigkeiten des Fluggerätes für die Simulationszeit bis 27 Sekunden

Die nachfolgend getroffenen Entscheidungen der KCV-ACU im Rahmen der Missionserfüllung zeigten, abgesehen von zeitlichen Bedingungen, keine Unterschiede zwischen der Simulation und dem real durchgeführten Flugversuch.

6.3.2 Evaluation des Anwendungsfalls “Einbruch der Avionikstromversorgung”

In diesem Abschnitt wird das Verhalten der KCV-ACU, bei einem simulierten Problem mit der Avionikstromversorgung, aufgezeigt. Hierfür wurde das Sensormodell, innerhalb der DLL (vgl. Abschnitt 5.3) der Reflex-Simulation, dahingehend modifiziert, dass ab einer Simulationszeit von 140 Sekunden die vermeintlich gemessene Spannung der Avionikstromversorgung abrupt abfällt (wie zum Beispiel bei dem Defekt einer LiFePo-Zelle).

Zur Wahrung der Flugsicherheit des UAVs wurde eine CPL-Klasse **safety** implementiert, welche einerseits auf Basis der aktuellen Spannung der Stromversorgung eine Vorhersage über die verbleibende Zeit einer sicheren Energieversorgung errechnet. Andererseits wird von der aktuellen Position die Entfernung zum Startpunkt (beide Größen werden in dem Function-Server berechnet) und über eine definierte Fluggeschwindigkeit die Flugdauer bestimmt, welche zum Zurückkehren zum Startpunkt notwendig wäre. Ist der Zeitraum für eine zuverlässige Stromversorgung

geringer als die Rückkehrdauer, so trifft die KCV-ACU die Entscheidung, direkt zum ersten Wegpunkt der Mission zurückzukehren. Der zeitliche Verlauf und wesentliche Ereignisse sind in Tabelle 18 zusammengefasst. Abbildung 6-10 zeigt den zurückgelegten Ground track des Fluggerätes für diesen Simulationsflug.

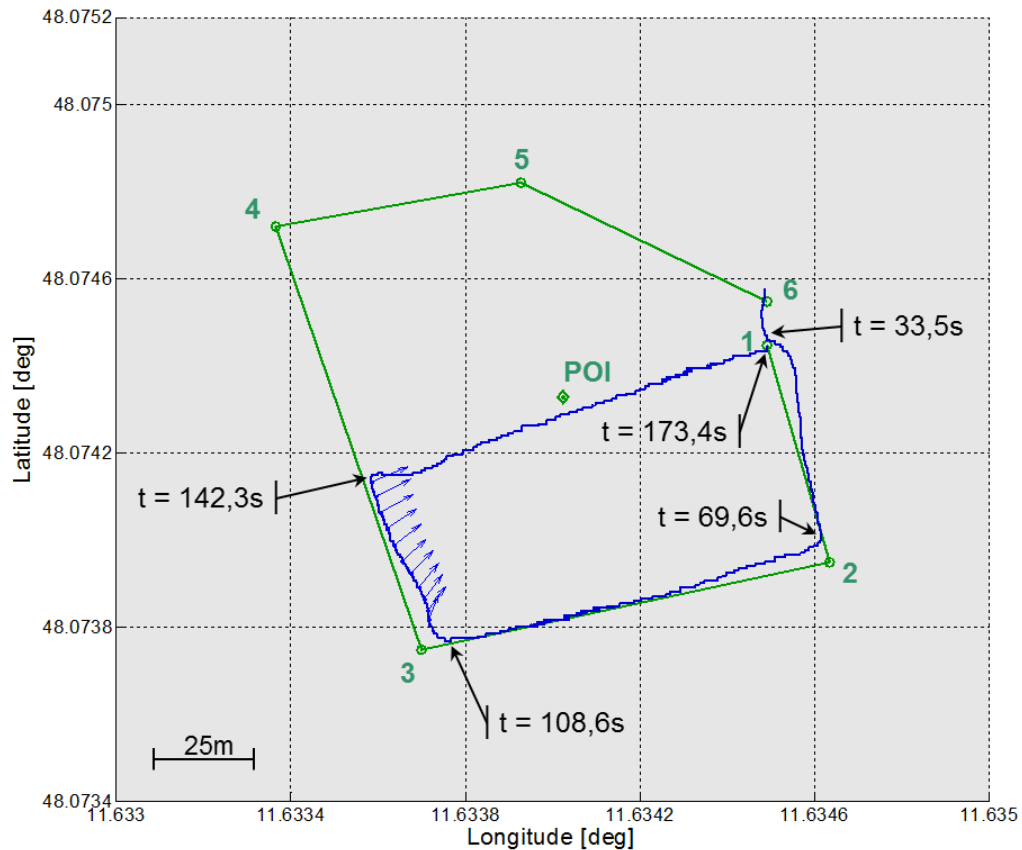


Abbildung 6-10. Simulierter Versuchsflug mit technischen Problemen bei der Avionikstromversorgung

Nach der Übergabe der Steuerautorität an das FCS startet die KCV-ACU mit der Durchführung der Mission, deren Verlauf anfangs identisch ist mit dem im vorigen Abschnitt beschriebenen Simulationslauf. Nach dem Passieren der Wegpunkte 1 (Zeitpunkt $t = 33,5\text{s}$) und 2 (Zeitpunkt $t = 69,6\text{s}$) wird Wegpunkt 3 zum Zeitpunkt $t = 108,6\text{s}$ erreicht. An diesem fordert die KCV-ACU wiederum das Verhalten *TARGET_POINT* an, um den erdfesten POI zu überwachen (vgl. Headingvektoren in Abbildung 6-10). Während dem Folgen des Tracks zwischen den Wegpunkten 3 und 4 tritt der simulierte Defekt der Stromversorgung ein und wird zum Zeitpunkt $t = 142,3\text{s}$ von der KCV-ACU erkannt. Die Reaktion der kognitiven Einheit ist das Informieren des Bedieners über das Stromversorgungsproblem. Weiterhin, um einen Absturz des Fluggerätes zu verhindern, aktiviert sie den ersten Wegpunkt der Flugroute und fordert das Verhaltens *FLY_TO_WP* (Zeitpunkt $t = 142,3\text{s}$) an. Diese FCS-Betriebsart ermöglicht das direkte Anfliegen eines Wegpunkts, ohne einem bestimmten Track zu folgen (vgl. Abschnitt 5.4.5.1 und Abbildung 5-19).

<i>Zeit [s]</i>	<i>Zustand / Ereignis</i>	<i>Beschreibung / Verfügbarkeit der Fähigkeiten</i>
33,5	Wegpunkt 1 erreicht	Der Wegpunkt 1 wurde erreicht. Weiterschaltung auf Wegpunkt 2 durch das MMS.
69,6s	Wegpunkt 2 erreicht	Der Wegpunkt 2 wurde erreicht. Weiterschaltung auf Wegpunkt 3 durch das MMS.
108,6	Wegpunkt 3 erreicht	Der Wegpunkt 3 wurde erreicht. Weiterschaltung auf Wegpunkt 4 durch das MMS.
140,0	Technischer Defekt der Avionikstromversorgung	Simulation eines technischen Defekts der Avionikstromversorgung durch einen abrupten Abfall der Spannung.
142,3	Detektion eines Problems mit der Stromversorgung	Die KCV-ACU erkennt ein Problem mit der Spannung der Stromversorgung und informiert den Bediener.
	Aktivieren des ersten Wegpunkts	Das MMS aktiviert den ersten Wegpunkt der Flugroute, um aufgrund des technischen Problems zum Startpunkt zurückzukehren.
	Anfordern von Verhalten durch das MMS	Das MMS fordert das Verhalten <i>FLY_TO_WP</i> durch dessen Eintragung in die KCV-Schicht an.
	Aktivierung der Betriebsart <i>Pos_H2</i> des FCS	Aktivierung der Betriebsart <i>Pos_H2</i> des FCS durch das maschinenspezifische Wissen. (Direkter Anflug auf den aktiven Wegpunkt)
173,4	Wegpunkt 1 erreicht	Der Wegpunkt 1 wurde erreicht.

Tabelle 18. KCV-ACU Simulationsevaluierung – Use case: Technisches Problem mit der Avionikstromversorgung – Simulationszeit bis 140 Sekunden

Zum Zeitpunkt $t = 173,4s$ erreicht das UAV den Wegpunkt 1 und das MMS fordert, wie auch bei den anderen durchgeführten Versuchsflügen, das Verhalten *AUTOMATIC_LAND* an, welches jedoch auch im Rahmen der Simulationsumgebung deaktiviert wurde.

In diesem simulierten Notfall wird für das angeforderte Verhalten *FLY_TO_WP* keine Ressourcenprüfung durchgeführt. Diese könnte ergeben, dass nicht mehr genügend Betriebsmittel vorhanden sind, um Wegpunkt 1 zu erreichen. Somit stünden zwar dem MMS einige Handlungsalternativen zur Verfügung (die Fähigkeiten des Fluggerätes sind an sich verfügbar), aber aufgrund geringer Ressourcen können nur wenige bis keine der Fähigkeiten mehr genutzt werden. Die Behandlung eines solchen speziellen Notfalls könnte es notwendig machen, dass durch zusätzliche Sensoren das umliegende Gelände erfasst wird, um einen geeigneten Landeplatz zu lokalisieren und dort eine automatische Landung durchzuführen, wie es beispielsweise Forschungsgegenstand von [Garcia-Pardo et al., 2002] ist.

6.4 Fazit

Die in diesem Kapitel vorgestellten Ergebnisse aus Simulations- und real durchgeführten Flugversuchen belegen die Machbarkeit, der im Rahmen dieser Arbeit

entwickelten ACU nach dem vorgestellten Konzept eines *Knowledge Configured Vehicle*.

Vergleicht man das von der ACU gezeitigte Verhalten sowohl bei der Durchführung der Mission in der Simulation als auch in der Realität, so sind – abgesehen von den Zeitpunkten der entsprechenden Ereignisse – keine Unterschiede ersichtlich. Damit konnte die Tragfähigkeit des Konzepts und die Möglichkeit der *fähigkeitsbasierten Flugführung* (vgl. Abschnitt 3.3) durch einen Softwareagenten demonstriert werden. Das Anfordern von gewünschtem Verhalten durch die Missionsmanagementfunktionen und die explizite Darstellung der von dem Fluggerät zur Verfügung gestellten Fähigkeiten, erlaubt die Anhebung des Abstraktionsniveaus von der *parametrischen Führung* – beispielsweise durch Setzen von Autopilotenmodi und -parametern oder Wegpunktnavigation – auf ein *symbolisches* Niveau. Dies schafft die Grundlage für den Einsatz von symbolisch arbeitenden Missionsmanagementfunktionen, wie sie im Fall der vorliegenden Arbeit als Teil einer künstlich kognitiven Einheit umgesetzt wurden.

Der Einsatz der symbolischen Informationsverarbeitung ermöglicht der ACU zudem ein umfassendes Systemverständnis – hinsichtlich der Verschaltung und Interaktion der Avionikkomponenten sowie der Subsysteme des Fluggerätes – aufzubauen. Damit ist der künstliche Agent im Stande zu identifizieren und zu benennen, *warum* und aufgrund *welcher* Umstände ein *gewünschtes Verhalten* nicht erzeugt werden kann, da sowohl die entsprechenden Beziehungen der Hardware- und Softwareelemente, als auch die erforderlichen Systemzustände und benötigten Ressourcen im maschinenspezifischen Wissen hinterlegt werden.

Weiterführend kann eine ACU durch den Einsatz von explizit modelliertem a-priori Wissen prinzipiell in die Lage versetzt werden adequat und rational auf Situationen zu reagieren, welche während der Designphase des kognitiven Agenten nicht berücksichtigt wurden. Dieses wissenbasierte Verhalten charakterisiert sich durch die Ausrichtung der Handlungen an explizit repräsentierten Zielen und der Verwendung des gesamten, verfügbaren a-priori Wissens für das Treffen von Entscheidungen. Besteht zum Beispiel aus Missionssicht die Notwendigkeit der Gewichtsreduktion des Luftfahrzeugs, so könnte das maschinenspezifische Wissen auch hierfür mögliche Fähigkeiten anbieten. Es ist denkbar, für abwerfbare Gegenstände bzw. ablassbare Betriebsmittel des Fluggerätes – wie Effektoren, Nutzlastkomponenten oder Kraftstoffe – entsprechende Fähigkeiten vorzusehen und deren physikalische Auswirkungen auf das Fluggerät zu berücksichtigen. Die Entscheidung, welche Maßnahme am geeignetsten ist um das Systemgewicht zu reduzieren obliegt letztlich den Missionsmanagementfunktionen. Die Vehikelmanagementfunktionen werden ihrerseits die möglichen Konsequenzen darstellen, zum Beispiel, dass nach dem Ablassen von Kraftstoff bestimmte Fähigkeiten nicht mehr angeboten werden können oder sich die Reichweite des Luftfahrzeugs verkleinert.

Trotz der erfolgreichen, automatisierten Durchführung der Evaluationsmission ergeben sich dennoch weitere Anforderungen im Hinblick an das Fähigkeitsspektrum des UAVs und der KCV-ACU. Für die kognitive Einheit wären einerseits elaboriertere Missionsmanagementfunktionen wünschenswert, die über weitreichendere – sowohl symbolische (z.B. für die Erzeugung von Handlungsagendas) als auch numerische (z.B.

für die Generierung der Flugroute, vgl. [Aurich, 2011]) – Planungsfähigkeiten verfügen. Andererseits beschränken sich im derzeitigen Entwicklungsstand der KCV-ACU die Handlungsoptionen, im Hinblick auf die Zusammenarbeit und Kommunikation mit dem Operateur, nur auf das Absetzen von Nachrichten und Warnungen. Um das Ziel einer wirklichen Zusammenarbeit zwischen dem Mensch und der Maschine zu erreichen (Kooperation), sind nach [Meitinger, 2008] seitens der kognitiven Einheit noch weitere Kompetenzen in dem Bereich Kommunikation (z.B. Führen eines Dialogs) und deren Anpassung an die menschlichen Eigenheiten notwendig.

Für eine hochautomatisierte Durchführung der Mission durch eine ACU, muss diese jedoch auch über ein entsprechendes Situationsverständnis verfügen, welches über die Kenntnis des aktuellen Zustands der Maschine und dessen Automation hinausgeht. Das Management und gegebenenfalls eine Rekonfiguration von Grund- und Subsystemen (engl.: Contingency Management) bei technischen Problemen ist sinnvoll, wie es auch Forschungsgegenstand von [Pecher et al., 2010] ist. Um aber situationsangepasste Entscheidungen treffen zu können, bedarf es der Wahrnehmung und Erkennung von Umweltelementen, wie etwa geographische Begebenheiten, Hindernissen, etwaigen anderen Luftfahrtteilnehmern aber auch missionsrelevanten Objekten, wie Gebäuden oder Fahrzeugen. Durch zusätzliche Missionssensoren wie Lidar, Radar oder bildgebende Sensoren (E/O bzw. I/R Kameras) könnten Daten erhoben, mit entsprechenden Algorithmen (z.B. Bildverarbeitung) verarbeitet, identifiziert (Symbolifizierung) und dem kognitiven System über einen Input-Server zur Verfügung gestellt werden.

Insbesondere auf Missionsseite können viele dieser Anforderungen, durch a-priori Wissen, in Form von weiteren Wissensmodellen, in eine ACU eingebracht werden. Dies dient der Verbesserung der kognitiven Fähigkeiten, im Hinblick auf angestrebte Funktionalitäten, wie Planung- und Missionsdurchführung und der Aufbau eines umfassenden Verständnisses der vorherrschenden Situation.

Im Gegenzug bedarf es für die Erfassung von Umweltelementen der Integration von weiterer Missionssensorik an das Fluggerät. Ferner muss diese an entweder zusätzliche oder bereits vorhandene Bordrechner angebunden, entsprechende Algorithmen zur Auswertung der Sensordaten in Prozesse implementiert und in die bestehende Softwarearchitektur integriert werden.

7 Zusammenfassung & Ausblick

Diese Dissertation markiert den ersten Flug eines speziellen KI-basierten Agenten zum kognitiven Missionsmanagement an Bord eines realen Mini-UAVs. Die präsentierten Ergebnisse zeigen die erfolgreiche Durchführung einer simplifizierten Aufklärungsmission durch eine sogenannte Artificial Cognitive Unit (ACU).

Ein UAV ist ein unbemanntes Luftfahrzeug, welches, obwohl dem Wortsinne nach „unbemannt“, doch letztlich immer von einem menschlichen Operateur gesteuert wird. Ungeachtet dessen sind heutige UAVs nicht mit ferngesteuerten Flugzeugen zu vergleichen, da sie in der Regel hoch automatisiert sind. In diesem Zusammenhang sind zum Beispiel Flugregelungssysteme zur Stabilisierung, Lageregelung, Bahnführung und Navigation des Luftfahrzeugs zu nennen. Eine solche Automatisierung wird in UAVs im Wesentlichen aus zwei Gründen eingeführt:

1. um den abgesetzten, menschlichen Operateur von schwierigen oder nicht mehr durchführbaren Aufgaben zu entlasten und auf diese Weise Überforderungen zu verhindern,
2. um im Fall des Verlusts der Datenlinkverbindung trotzdem noch eine sichere Fortführung und gegebenenfalls Beendigung des Flugs gewährleisten zu können.

Im ersten Fall sind die zu automatisierenden Aufgaben zunächst die Funktion der automatischen Steuerung des Luftfahrzeugs. Eine manuelle Steuerung bedingt entsprechende Fähigkeiten seitens eines Operateurs und darüber hinaus würden hohe Anforderungen an die dafür benötigte Funkverbindung resultieren, wie deren Zuverlässigkeit, maximale Zeitverzögerung und verfügbare Bandbreite. Weiterführend bedarf es seitens der Maschine auch an Funktionalitäten, die den Operateur bei einer Mission unterstützen und *gemeinsam* mit dem Menschen an der Erfüllung des Arbeitsziels arbeiten. Hierfür muss die Maschine befähigt werden eigene, missionsrelevante Entscheidungen zu treffen, um beispielsweise einen gegebenen Auftrag selbständig durchzuführen (z.B. Auftragsbasierte Führung, vgl. [Uhrmann et al., 2010b]).

Im zweiten Fall geht es weniger um den Mensch-Maschine Aspekt sondern vor allem um Sicherheitsaspekte. Bei einem Ausfall der Funkverbindung ist es gewünscht und auch unbedingt notwendig, dass das Fluggerät nicht nur stabil fliegt sondern auch weiterhin zuverlässig vorhersehbares Verhalten aufweist. Bei derartigen Verhaltensentscheidungen geht es zum Beispiel um die Auswahl eines geeigneten Notlandeplatzes oder im besten Fall die Entscheidung zur Fortsetzung der Mission.

In beiden Fällen muss eine Aufgabenteilung zwischen Mensch und Automation vorgenommen werden, bei der kognitive Aufgaben des Menschen auf die Maschine übergehen. Solche *kognitiven Fähigkeiten* umfassen allgemein gesprochen das Aufnehmen von Informationen, deren Bewertung und Verarbeitung, das Treffen von Entscheidungen, die Entwicklung von Vorgehensplänen und letztlich deren Ausführung.

Hierzu wurde im Rahmen dieser Dissertation ein spezieller KI-Agent entwickelt, an den bestimmte Anforderungen zu stellen waren. So bedarf die Erzeugung der gewünschten kognitiven Fähigkeiten in einem technischen System einer dedizierten Architektur und Art der Informationsverarbeitung, welche durch den Einsatz der verwendeten kognitiven Systemarchitektur COSA erfüllt werden konnte. Des Weiteren sollte der spezielle KI-Agent aus modularen, wiederverwendbaren Komponenten zusammengesetzt werden können, um gleichzeitig den Gegebenheiten des verwendeten Fluggerätes und der Aufgabenstellung der geforderten Mission gerecht werden zu können. Einen Lösungsansatz hierfür bietet das im Rahmen dieser Dissertation entwickelte Konzept eines sogenannten *Knowledge Configured Vehicles*, das die dedizierten Fähigkeiten eines menschlichen Piloten in kombinierbaren Wissenspaketen abbildet. Diese werden in den KI-Agenten eingebracht und lassen sich in Wissen über die Anwendung und die Mission sowie in Wissen über das Fluggerät, dessen Überwachung sowie die Bedienung und Konfiguration der Automation untergliedern. Diese Vorgehensweise zeigt eine Möglichkeit auf, wie künftige Entscheidungssysteme für komplex automatisierte, unbemannte Fluggeräte entwickelt werden können, mit dem Ziel der Erhöhung der Autonomie in den Feldern Missionsentscheidungen und Systemmanagement.

Eine Weiterentwicklung der künstlich kognitiven Einheit könnte dahingehend erfolgen, dass deren Aufgabenstellungen und die Einsatzszenarien weiter detailliert und gesamtheitlich komplexer gestaltet werden. Hierbei wäre sowohl der Ausbau der kognitiven Fähigkeiten im Hinblick auf den Einsatz von zusätzlicher Missionssensorik für eine erweiterte Umwelterfassung (Sensormanagement) denkbar, als auch die mögliche Steigerung der Anzahl an Missionselementen, welche durch deren Anwesenheit und gegebenenfalls eigenes Verhalten, die Entscheidungsfindung des kognitiven Agenten beeinflussen. Auch der Bereich des Managements von Grund- und Subsystemen eines unbemannten Fluggerätes könnte im Bezug auf deren Rekonfiguration und Redundanzmanagement, vor allem beim Auftreten von technischen Problemen (engl.: Contingency management) durch kognitive Fähigkeiten weiter verbessert werden. Um in der Zukunft den Einsatz von kognitiven Agenten auf Basis wissensbasierter Systeme und deren angestrebte Verwendung in kommerziellen Produkten und UAV-Systemen zu realisieren, bedarf es darüber hinaus einem entsprechenden Systems-Engineering-Ansatz und der Entwicklung einer entsprechenden Methodik, die die Zulassung solcher Systeme erlaubt.

Im Hinblick auf die Integration eines UAV-Operators, der aktiv bei der Durchführung einer Mission eingebunden wird, könnten Versuche verschiedener Führungsparadigmen, wie etwa die Wegpunktnavigation, die fähigkeitsbasierte und die auftragsbasierte Führung (vgl. [Uhrmann et al., 2010a]) vorgenommen und anschließend miteinander verglichen werden. In diesem Zusammenhang wäre es auch wünschenswert, die ACU mit kooperativen Fähigkeiten (vgl. [Meitinger, 2008]) auszustatten, um hinsichtlich einer möglichen, simultanen Führung mehrerer UAVs durch einen Operateur, die Gesamteffizienz eines solchen Arbeitssystems zu steigern.

Abschließend könnten weitere Arbeiten darin bestehen, das entwickelte Konzept und die bereits vorhandene Implementierung der KCV-ACU auf andere Fluggeräte zu übertragen. Damit könnte die mit dieser Arbeit gelegte Basis auf andere Versuchsträger

ausgeweitet werden, um auch hier von den Vorzügen der symbolischen Interaktion zwischen einem KI-Agenten und einer komplex automatisierten Maschine zu profitieren. Mögliche Zielsysteme sind hierbei der institutseigene Forschungsdemonstrator *Graphite* [Reidelstürz & Schulte, 2010] – ein elektrisch angetriebener Motorsegler – oder der im Rahmen des aktuellen Forschungsprojekts – zusammen mit der Firma Cassidian und einer Vielzahl weiterer Universitäten – projektierte UCAV-Demonstrator *SAGITTA*.

Dieser Überblick über einige, mögliche Anwendungsgebiete und Forschungsthemen im Umfeld kognitiver Flugführung auf Basis von wissensbasierten Systemen vermittelt einen Eindruck davon, für welche Themengebiete, die im Rahmen dieser Arbeit erzielten Ergebnisse relevant sind.

A Literaturverzeichnis

[Achtelik et al., 2009]

M. Achtelik, T. Zhang, K. Kühnlenz & M. Buss (2009). Visual tracking and control of a quadcopter using a stereo camera system and inertial sensors. In: *Proceedings of IEEE International Conference on Mechatronics and Automation*. Changchun, Jilin, China, 9.-12. August 2009.

[Evan Ackerman, 2009]

Evan Ackerman (2009). Irish UAV Gets Lost, Tries To Make It Home From Africa, Fails. 16.01.2009. Online-Ressource: <http://www.botjunkie.com/2009/01/16/irish-uav-gets-lost-tries-to-make-it-home-from-africa-fails/> (Letzter Zugriff: 28 September 2011).

[ACT-R Homepage]

ACT-R Homepage. Online-Ressource: <http://act-r.psy.cmu.edu/>. Letzter Zugriff: 22. September 2011.

[Adami, 1998]

C. Adami (1998). *Introduction to artificial life*. Berlin. Springer Verlag.

[Adams, 2005]

C. Adams (2005). UAV Avionics: Raising the Bar. In: *Avionics Magazine* 29(9). S. 26.

[Adiprawita et al., 2007]

W. Adiprawita, A.S. Ahmad & J. Sembiring (2007). Hardware in the Loop Simulation for Simple Low Cost Autonomous UAV (Unmanned Aerial Vehicle) Autopilot System Research and Development. In: *International Conference on Electrical Engineering and Informatics, ICEEI*. Bandung, Indonesien, 17.-19. Juli 2007.

[UNMANNED AERIAL-SYSTEM Homepage]

UNMANNED AERIAL-SYSTEM Homepage. Online-Ressource: <http://www.unmanned-aerial-system.com>. Letzter Zugriff: 09. November 2010.

[Anderson et al., 2004]

John R. Anderson, Daniel Bothell, Michael D. Byrne, Scott Douglass, Christian Lebiere & Yulin Qin (2004). An Integrated Theory of Mind. In: *Psychological Review*, 111(4). S. 1036-1060.

[Aracil et al., 2002]

R. Aracil, M. Ferre, M. Hernando, E. Pinto & JM Sebastian (2002). Telerobotic system for live-power line maintenance: ROBTET. In: *Control Engineering Practice*, 10(11). S. 1271-1281.

- [Ascending Technologies Homepage]
Ascending Technologies Homepage. Online-Ressource: <http://www.asctec.de/>.
Letzter Zugriff: 11. März 2011.
- [Aurich, 2011]
Pierre Aurich (2011). *Entwicklung eines kognitiven Agenten zur auftragsbasierten UAV-Führung unter Nutzung von „Optimal Control“*.
Masterarbeit, Universität der Bundeswehr München, Fakultät für Luft- und Raumfahrttechnik.
- [Balsler et al., 2003]
B. Balsler, M. Foerster & G. Grabowski (2003). Modulare Avionik als Grundlage fuer Systemdefinition, Systemkonfiguration und Systemkontrolle. In: *GI Jahrestagung - Schwerpunkt "Sicherheit - Schutz und Zuverlaessigkeit"*.
Frankfurt am Main, 29. September - 02. Oktober 2003. S. 179-190.
- [Barnard, 2008]
J. Barnard (2008). Small UAV Command, Control and Communication Issues. In: *IET Seminar on Communicating with UAV's*. S. 75-85.
- [Bento, 2008]
Maria de Fátima Bento (2008). Unmanned Aerial Vehicles: An overview. In: *InsideGNSS (Januar/Februar)*. S. 54 - 61.
- [Bhanu et al., 2002]
B. Bhanu, S. Das, B. Roberts & D. Duncan (2002). A system for obstacle detection during rotorcraft low altitude flight. In: *Transactions on Aerospace and Electronic Systems*, 3. S. 875-897.
- [Biass, 2008]
E.H. Biass (2008). Drone Eye Capability and Cost. In: *Armada International* 32(2). S. 20.
- [Billings, 1991]
C. E. Billings (1991). Toward a human-centered aircraft automation philosophy. In: *The International Journal of Aviation Psychology*, 1(4). S. 261-270.
- [Billings, 1997]
C. E. Billings (1997). *Aviation Automation - the Search for a Human-Centered Approach*. Mahwah, NJ. Erlbaum.
- [Boeing Defense, Space and Security, 2011]
Boeing Defense, Space and Security (2011). Boeing: A160 Hummingbird.
Online-Ressource:
http://www.boeing.com/bds/phantom_works/hummingbird/docs/hummingbird_overview.pdf (Letzter Zugriff: 11 März 2011).
- [Bommer, 2000]
Bommer (2000). Online-Ressource: www.rolfferch.de/F104G/History_CCV-F-104G.pdf (Letzter Zugriff: 10 Mai 2011).

[Bortoff, 2000]

S.A. Bortoff (2000). Path planning for UAVs. In: *Proceedings of the American Control Conference*. S. 364-368.

[Bosch et al., 2006]

S. Bosch, S. Lacroix & F. Caballero (2006). Autonomous detection of safe landing areas for an UAV from monocular images. In: *International Conference on Intelligent Robots and Systems*. Beijing, China, 9. - 15. Oktober 2006. S. 5522-5527.

[Breiter et al., 2006]

R. Breiter, W. Cabanski, T. Ihle, K.H. Mauk & W. Rode (2006). AIM thermal imagers for reconnaissance and targeting applications. In: *Proceedings of SPIE Defence & Security*. Orlando, Florida, USA, 17.-21. April 2006.

[Briere & Traverse, 2002]

D. Briere & P. Traverse (2002). AIRBUS A320/A330/A340 electrical flight controls-a family of fault-tolerant systems. In: *The Twenty-Third International Symposium on Fault-Tolerant Computing, 1993. FTCS-23. Digest of Papers*. S. 616-623.

[Brisset et al., 2008]

P. Brisset, M. Bronz, M. Gorraz & J. Tyler (2008). YAP (Yet Another Paparazzi). In: *École Nationale de l'Aviation Civile*.

[Brisset et al., 2006]

P. Brisset, A. Drouin, M. Gorraz, P.S. Huard & J. Tyler (2006). The paparazzi solution. In: *Micro Air Vehicle Conference*. Sandestin, Florida, USA.

[Brockhaus, 2001]

R. Brockhaus (2001). *Flugregelung*. Berlin Heidelberg. Springer Verlag.

[Brüggenwirth et al., 2011]

Stefan Brüggenwirth, Wolfgang Pecher & Axel Schulte (2011). Design Considerations for COSA². In: *IEEE Symposium Series on Computational Intelligence, SSCI*. Paris, Frankreich, 11. - 15. April 2011.

[Brüggenwirth et al., 2010]

Stefan Brüggenwirth, Ruben Strenzke, Alexander Matzner & Axel Schulte (2010). A Generic Cognitive System Architecture applied to the Multi-UAV Flight Guidance Domain. In: *ICAART*. Valencia, Spanien, 22. - 24. Januar 2010.

[Buskey et al., 2005]

G. Buskey, G. Wyeth & J. Roberts (2005). Autonomous helicopter hover using an artificial neural network. In: *Proceedings of the International Conference on Robotics and Automation*. Barcelona, Spanien. S. 1635-1640.

[Calhoun et al., 2005]

G. L. Calhoun, M. H. Draper, M. F. Abernathy, F. Delgado & M. Patzek (2005). Synthetic vision system for improving unmanned aerial vehicle operator situation awareness. In: *Proceedings of SPIE Vol 5802, in Enhanced and Synthetic Vision*. Orlando, Florida, USA.

- [Calise & Rysdyk, 2002]
A.J. Calise & R.T. Rysdyk (2002). Nonlinear adaptive flight control using neural networks. In: *IEEE Control Systems Magazine* 18(6). S. 14-25.
- [Campbell & Whitacre, 2007]
M.E. Campbell & W.W. Whitacre (2007). Cooperative tracking using vision measurements on seascan UAVs. In: *IEEE Transactions on Control Systems Technology*. S. 613-626.
- [Carrigan et al., 2008]
G. Carrigan, D. Long, M.L. Cummings & J. Duffner (2008). Human factors analysis of Predator B crash. *Proceedings of AUVSI: Unmanned Systems North America*.
- [Cartwright et al., 2007]
James E. Cartwright, James R. Jr. Clapper, John J. Jr. Young & John G. Grimes (2007). *Unmanned Systems Roadmap 2007 - 2032. Secretaries of the military departments. U.S. Department of Defense*.
- [Caveney & Sengupta, 2006]
D. Caveney & R. Sengupta (2006). Architecture and application abstractions for multi-agent collaboration projects. In: *44th Conference on Decision and Control, CDC-ECC'05*. S. 3572-3577.
- [Chao et al., 2008]
H. Chao, M. Baumann, A. Jensen, Y.Q. Chen, Y. Cao, W. Ren & M. McKee (2008). Band-reconfigurable multi-UAV-based cooperative remote sensing for real-time water management and distributed irrigation control. In: *IFAC World Congress*. Seoul, Korea, 6. - 11. Juli 2008.
- [Chao et al., 2007]
H. Chao, Y. Cao & Y.Q. Chen (2007). Autopilots for small fixed-wing unmanned air vehicles: A survey. In: *International Conference on Mechatronics and Automation*. Harbin, Heilongjiang, China, 5. - 8. August 2007. S. 3144-3149.
- [Chen et al., 2007]
J.Y.C. Chen, E.C. Haas & M.J. Barnes (2007). Human performance issues and user interface design for teleoperated robots. In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews* 6(37). S. 1231--1245.
- [Cholewiak & Collins, 2000]
R. W. Cholewiak & A. A. Collins (2000). The generation of vibrotactile patterns on a linear array: Influences of body site, time, and presentation mode. In: *Attention, Perception & Psychophysics*, 62(6). S. 1220-1235.
- [Christophersen et al., 2006]
H.B. Christophersen, R.W. Pickell, J.C. Neidhoefer, A.A. Koller, K. Kannan & E.N. Johnson (2006). A Compact Guidance, Navigation, and Control System for Unmanned Aerial Vehicles. In: *Journal of Aerospace Computing, Information, and Communication*, 3(5). S. 187-213.

[Clarke, 2002]

R. Clarke (2002). Asimov's laws of robotics: Implications for Information Technology - Part I. In: *IEEE Computer*, 26(12). S. 53-61.

[Clauss, 2011]

Sebastian Clauss (2011). *Entwurf und Umsetzung eines UAS unter komplementärer Nutzung kognitiver Automation und „Optimal Control“ Augmentierung*. Masterarbeit, Universität der Bundeswehr München, Fakultät für Luft- und Raumfahrttechnik.

[Cloud Cap Technologies, 2006]

Cloud Cap Technologies (2006). Cloud Cap Technology -- Piccolo Family of Small Integrated Autopilots. Online-Ressource:
http://www.cloudcaptech.com/piccolo_system.shtm (Letzter Zugriff: 10 März 2011).

[Colditz, 2009]

Sebastian Colditz (2009). *Development of a Test Campaign to Tune a Helicopter Controller*. Semesterarbeit, Technische Universität München.

[Cowling et al., 2007]

I.D. Cowling, O.A. Yakimenko, J.F. Whidborne & A.K. Cooke (2007). A Prototype of an Autonomous Controller for a Quadrotor UAV. In: *European Control Conference*. Kos, Griechenland, 2. - 5. Juli 2007. S. 2-5.

[Cox & Wilfong, 1990]

Ingemar J. Cox & Gordon T. Wilfong (1990). *Autonomous robot vehicles*. Berlin, Heidelberg. Springer.

[Crossbow Technology, 2007]

Crossbow Technology (2007). NAV420 - Crossbow Technology. Online-Ressource:
http://bullseye.xbow.com:81/Support/Support_pdf_files/NAV420_Users_Manual.pdf (Letzter Zugriff: 07 Juli 2011).

[Cummings & Guerlain, 2004]

M. Cummings & S. Guerlain (2004). Human performance issues in supervisory control of autonomous airborne vehicles. In: *AUVSI Unmanned Systems North America Conference*.

[Cummings et al., 2006]

M. Cummings, A.T. Kirschbaum, A. Sulmistras & J.T. Platts (2006). STANAG 4586 Human Supervisory Control Implications. *Air and Weapon Systems Dept, Dstl Farnborough & the Office of Naval Research*.

[Cummings & Mitchell, 2005]

M. Cummings & P.J. Mitchell (2005). Managing Multiple UAVs through a Timeline Display. In: *Proceedings of InfoTech@Airspace*. Arlington, VA, USA, 26.-29. September 2005.

[DARPA, 2005]

DARPA (2005). Defense Advanced Research Projects Agency. Online-Ressource: <http://www.darpa.mil/WorkArea/DownloadAsset.aspx?id=1773> (Letzter Zugriff: 14 April 2011).

[Dierks & Jagannathan, 2009]

T. Dierks & S. Jagannathan (2009). Neural network output feedback control of a quadrotor UAV. In: *47th Conference on Decision and Control, CDC 2008*. S. 3633--3639.

[Dittrich & Johnson, 2002]

J. S. Dittrich & E. N. Johnson (2002). Multi-sensor navigation system for an autonomous helicopter. In: *The 21st Digital Avionics Systems Conference*.

[Dobrokhodov et al., 2006]

V. Dobrokhodov, I. Kaminer, K.D. Jones & R. Ghabcheloo (2006). Vision-based tracking and motion estimation for moving targets using small UAVs. In: *American Control Conference*. Minneapolis, Minnesota, USA, 14. - 16. Juni 2006.

[Dobrokhodov et al., 2008]

V. Dobrokhodov, I. Kaminer, K. Jones, E. Xargay, N. Hovakimyan, C. Cao, M. Lizarraga & I. Gregory (2008). Flight Validation of a Metrics Driven L1 Adaptive Control. In: *AIAA Guidance, Navigation, and Control Conference and Exhibit*. Honolulu, Hawaii, USA, 18. - 21. August 2008.

[DoD, 2001]

DoD (2001). *Unmanned Aerial Vehicle Roadmap, 2000 - 2025*. Office of the Secretary of Defense, Department of Defense, Washington, DC.

[Doitsidis et al., 2004]

L. Doitsidis, KP Valavanis, NC Tsourveloudis & M. Kontitsis (2004). A framework for fuzzy logic based UAV navigation and control. In: *Proceedings of the IEEE International Conference on Robotics and Automation*. New Orleans, LA, USA, 26. April - 1. Mai 2004. S. 4041-4046.

[Draper et al., 2006]

M. Draper, G. Calhoun, J. Nelson, A. Lefebvre & H. Ruff (2006). Synthetic Vision Overlay Concepts for Uninhabited Aerial Vehicle Operations: Evaluation of Update Rate on Four Operator Tasks. In: *NATO RTO-Meeting Proceedings HFM-135: Human Factors of Uninhabited Military Vehicles as Force Multipliers*. Biarritz, Frankreich, 9.-11. Oktober 2006.

[Draper et al., 2003]

M. Draper, G. Calhoun, H. Ruff, D. Williamson & T. Barry (2003). Manual versus speech input for unmanned aerial vehicle control station operations. In: *Human Factors and Ergonomics Society Annual Meeting Proceedings*. Santa Monica, CA, USA, 13. - 17. Oktober 2003. S. 109-113.

[Duc, 1978]

J.-M. Duc (1978). Control Configured Vehicle Design Philosophy. In: *AGARD Active Controls in Aircraft Design*.

[Ecker et al., 2009]

Wolfgang Ecker, Rainer Dömer & Wolfgang Müller (2009). *Hardware dependent software*. Berlin, Heidelberg. Springer.

[Edrich, 2006]

M. Edrich (2006). Ultra-lightweight synthetic aperture radar based on a 35 GHz FMCW sensor concept and online raw data transmission. In: *IEE Proceedings Radar, Sonar and Navigation*. S. 129-134.

[Sparkfun Electronics Homepage]

Sparkfun Electronics Homepage. Online-Ressource: <http://www.sparkfun.com/products/10736>. Letzter Zugriff: 22. September 2011.

[EMT Ingenieurgesellschaft, 2009]

EMT Ingenieurgesellschaft (2009). Online-Ressource: http://www.emt-penzberg.de/fileadmin/download/LUNA_de.pdf (Letzter Zugriff: 09 Februar 2011).

[EMT Homepage]

EMT Homepage. Online-Ressource: <http://www.emt-penzberg.de/index.php?id=18&L=1>. Letzter Zugriff: 19. Oktober 2010.

[Endsley, 1996]

Mica R. Endsley (1996). Automation and Situation Awareness. In: *Automation and human performance: Theory and applications*. S. 163-181.

[Euston et al., 2008]

M. Euston, P. Coote, R. Mahony, J. Kim & T. Hamel (2008). A complementary filter for attitude estimation of a fixed-wing uav. In: *International Conference on Intelligent Robots and Systems*. Nizza, Frankreich, 22. - 26. September 2008. S. 340-345.

[Fitts & Others, 1951]

P.M. Fitts & Others (1951). *Human engineering for an effective air-navigation and traffic-control system*. National Research Council, Division of Anthropology and Psychology, Committee on Aviation Psychology.

[Fong & Thorpe, 2001]

T. Fong & C. Thorpe (2001). Vehicle teleoperation interfaces. In: *Autonomous robots, 11(1)*. S. 9-18.

[Franke et al., 2005]

Jerry L. Franke, Vera Zaychik, Thomas M. Spura & Erin E. Alves (2005). Inverting the operator/vehicle ratio: Approaches to next generation UAV command and control. In: *Association for Unmanned Vehicle Systems International and Flight International, Unmanned Systems North America Baltimore, MD*.

[Frey, 2005]

Andreas Frey (2005). *Überwachung und Kontrolle in einem künstlichen kognitiven System zur autonomen Fahrzeugführung*. Dissertation, Universität der Bundeswehr München. Berlin: Köster.

[Garcia-Pardo et al., 2002]

P.J. Garcia-Pardo, G.S. Sukhatme & J.F. Montgomery (2002). Towards vision-based safe landing for an autonomous helicopter. In: *Robotics and Autonomous Systems*, 38(1). S. 19-29.

[Gemperle et al., 2002]

F. Gemperle, N. Ota & D. Siewiorek (2002). Design of a wearable tactile display. In: *Proceedings of the Fifth International Symposium on Wearable Computers*. Seattle, Washington, USA, 7. - 10. Oktober 2002. S. 5-12.

[General Atomics Aeronautical Systems, Inc., 2011]

General Atomics Aeronautical Systems, Inc. (2011). General Atomics Aeronautical Systems, Inc. - Predator B brochure. Online-Ressource: http://www.ga-asi.com/products/aircraft/pdf/Predator_B.pdf (Letzter Zugriff: 11 März 2011).

[Goza et al., 2004]

S. Goza, R. Ambrose, M. Diftler & I. Spain (2004). Telepresence control of the NASA/DARPA robonaut on a mobility platform. In: *Proceedings of the SIGCHI conference on Human factors in computing systems*. Wien, Österreich, 24. - 29. April 2004. S. 623-629.

[Grecu & Gonsalves, 2000]

D.L. Grecu & P.G. Gonsalves (2000). Agent-based simulation environment for UCAV mission planning and execution. In: *Proceedings of the AIAA Guidance, Navigation and Control Conference*. Denver, Colorado, USA.

[Northrop Grumman Homepage]

Northrop Grumman Homepage. Online-Ressource: <http://www.as.northropgrumman.com>. Letzter Zugriff: 26. Oktober 2010.

[Gurdan et al., 2007]

D. Gurdan, J., Achtelik, M. Stumpf, K.M. Doth, G. Hirzinger & D. Rus (2007). Energy-efficient autonomous four-rotor flying robot controlled at 1 khz. In: *International Conference on Robotics and Automation*. Rom, Italien, 10. - 14. April 2007. S. 361-366.

[Gutierrez et al., 2002]

L.B. Gutierrez, G. Vachtsevanos & B. Heck (2002). A hierarchical intelligent control architecture for unmanned aerial vehicles. In: *Proceedings of the 21st Digital Avionics Systems Conference*.

[Haas & Stachowiak, 2007]

E. Haas & C. Stachowiak (2007). Multimodal displays to enhance human robot interaction on-the-move. In: *Proceedings of the 2007 Workshop on Performance Metrics for Intelligent Systems*. Washington, DC, USA, 28. - 30. August 2007. S. 135-140.

[He et al., 2006]

Z. He, R.V. Iyer & P.R. Chandler (2006). Vision-based UAV flight control and obstacle avoidance. In: *American Control Conference*. Minneapolis, Minnesota, USA, 14. - 16. Juni 2006.

[Hermans & Decuypere, 2005]

D. Hermans & R. Decuypere (2005). A Challenge for Micro and Mini UAV: The Sensor Problem. In: *Advanced Sensor Payloads for UAV Symposium*. Lissabon, Portugal.

[Higgins, 1975]

W.T. Higgins (1975). A comparison of complementary and Kalman filtering. In: *IEEE Transactions on Aerospace and Electronic Systems*, 3. S. 321-325.

[Hill & Tauson, 2001]

S.G. Hill & R.A. Tauson (2001). *Soldier Performance Issues in C2 "On the Move"*. Human Research and Engineering Directorate, U.S. Army Research Laboratory. Aberdeen Proving Ground, MD, USA.

[Höcht, 2007]

Leonhard Höcht (2007). *Design and Implementation of a Flight Control System for an Experimental UAV Helicopter*. Diplomarbeit, Technische Universität München.

[Holzapfel, 2004]

Florian Holzapfel (2004). *Nichtlineare adaptive Regelung eines unbemannten Fluggerätes*. Dissertation, Technische Universität München.

[Höse, 2006]

Dennis Höse (2006). *Entwicklung eines Mikrokontrollerbasierten Sensordatenerfassungssystems für unbemannte Fluggeräte*. Diplomarbeit, Universität der Bundeswehr München, Fakultät für Luft- und Raumfahrttechnik.

[Höse et al., 2007]

Dennis Höse, Michael Kriegel & Axel Schulte (2007). Development of a Microcontroller based Sensor Acquisition System for Uninhabited Aerial Vehicles. In: *First CEAS European Air and Space Conference*. Berlin, 10. - 13. September 2007.

[How et al., 2004]

J. How, E. King & Y. Kuwata (2004). Flight demonstrations of cooperative control for UAV teams. In: *AIAA 3rd Unmanned Unlimited Technical Conference, Workshop and Exhibit*. Chicago, Illinois, USA, 20. - 23. September 2004.

- [International Standards Organisation, 1984]
International Standards Organisation (1984). *ISO7498: Information technology – Open Systems Interconnection – Basic Reference Model*.
- [Jaeger, 2007]
Christian Jaeger (2007). *Entwicklung eines Flugreglers für einen Modellhubschrauber mittels neuronalen Netzen*. Diplomarbeit, Universität der Bundeswehr München, Fakultät für Luft- und Raumfahrttechnik.
- [Jang & Liccardo, 2006]
J.S. Jang & D. Liccardo (2006). Automation of small UAVs using a low cost MEMS sensor and embedded computing platform. In: *25th Digital Avionics Systems Conference*. Portland, OR, USA, 15. - 19. Oktober 2006. S. 1-9.
- [Jensen et al., 2009]
A.M. Jensen, M. Baumann & Y.Q. Chen (2009). Low-cost multispectral aerial imaging using autonomous runway-free small flying wing vehicles. In: *International Symposium on Geoscience and Remote Sensing, IGARSS*. Kapstadt, Afrika, 12. - 17. Juli 2009.
- [Johnson & Kannan, 2002]
E.N. Johnson & S.K. Kannan (2002). Adaptive flight control for an autonomous unmanned helicopter. In: *AIAA Guidance, Navigation and Control Conference*. No. AIAA-2002-4439, Monterey, Kalifornien, USA.
- [Johnson & Mishra, 2002]
E.N. Johnson & S. Mishra (2002). Flight Simulation for the Development of an Experimental UAV. In: *Proceedings of the AIAA modeling and simulation technologies conference*. Monterey, CA, USA, 5. - 8. August 2002.
- [Johnson & Schrage, 2003]
E.N. Johnson & D.P. Schrage (2003). The Georgia Tech unmanned aerial research vehicle: Gtmax. In: *Proceedings of the AIAA Guidance, Navigation, and Control Conference*.
- [Johnson & Turbe, 2006]
E.N. Johnson & M.A. Turbe (2006). Modeling, control, and flight testing of a small ducted-fan aircraft. In: *AIAA Guidance, Navigation and Control Conference and Exhibit*. San Francisco, Kalifornien, USA. S. 769–779.
- [Jones, 1996]
Gary Jones (1996). The architectures of Soar and ACT-R, and how they model human behaviour. In: *Artificial Intelligence and Simulation of Behaviour Quarterly*, 96. S. 41-44.
- [Kalman, 1960]
R.E. Kalman (1960). A new approach to linear filtering and prediction problems. In: *Journal of basic Engineering*, 82(1). S. 35-45.
- [Kaminer et al., 2007]
I. Kaminer, O. Yakimenko, V. Dobrokhodov, A. Pascoal, N. Hovakimyan, C. Cao, A. Young & V. Patel (2007). Coordinated path following for time-critical

missions of multiple UAVs via L1 adaptive output feedback controllers. In: *AIAA Guidance, Navigation and Control Conference and Exhibit*. Hilton Head, South Carolina, USA, 20. - 23. August 2007.

[Kaneshige et al., 2000]

J. Kaneshige, J. Bull & J.J. Totah (2000). Generic neural flight control and autopilot system. In: *AIAA Guidance, Navigation, and Control Conference and Exhibit*. Denver, CO, USA, 14. - 17. August 2000. S. 14-17.

[Kannan et al., 2002]

S. Kannan, C. Restrepo, I. Yavrucuk, L. Wills, D. Schrage & J.V.R. Prasad (2002). Control algorithm and flight simulation integration using the open control platform for unmanned aerial vehicles. In: *Proceedings of the 18th Digital Avionics Systems Conference*. Irvine, CA, USA, 27. - 31. Oktober 2002.

[Karim & Heinze, 2005]

S. Karim & C. Heinze (2005). Experiences with the design and implementation of an agent-based autonomous UAV controller. In: *Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*. Utrecht, Niederlande, 25. - 29. Juli 2005. S. 26.

[Karim et al., 2004]

S. Karim, C. Heinze & S. Dunn (2004). Agent-based mission management for a UAV. In: *Proceedings of the 2004 Intelligent Sensors, Sensor Networks and Information Processing Conference*. Melbourne, Australien, 14. - 17. Dezember 2004. S. 481-486.

[Kawalec et al., 2006]

A. Kawalec, W. Komorniczak, J. Pietrasinski & W. Czarnecki (2006). Evaluation of a low-cost microUAV platform for sensor suite. In: *International Radar Symposium*. Krakau, Polen, 24. - 26. Mai 2006. S. 1-4.

[Khantsis & Bourmistrova, 2005]

S. Khantsis & A. Bourmistrova (2005). UAV controller design using evolutionary algorithms. *AI 2005: Advances in Artificial Intelligence* S. 1025-1030.

[Klußmann & Malik, 2007]

Niels Klußmann & Arnim Malik (2007). *Lexikon der Luftfahrt*. Berlin, Heidelberg. Springer.

[Kriegel et al., 2011]

Michael Kriegel, Stefan Brüggewirth & Axel Schulte (2011). Knowledge Configured Vehicle - A layered artificial cognition based approach to decoupling high-level UAV mission tasking from vehicle implementations. In: *AIAA GNC/AFM/MST Conference*. Portland, OR, USA, 8.-11. August 2011.

[Kriegel et al., 2007]

Michael Kriegel, Claudia Meitinger & Axel Schulte (2007). Operator assistance and semi-autonomous functions as key elements of future systems for multiple UAV guidance. In: *7th Conference on Engineering Psychology & Cognitive*

Ergonomics, in conjunction with HCI International. Beijing, China, 22. - 27. Juli 2007.

[Kriegel & Schulte, 2006]

M. Kriegel & A. Schulte (2006). Work system analysis of the integration of autonomous functions and intelligent operator assistance in UAV guidance. In: *NATO RTO HFM Symposium on Human Factors of Uninhabited Military Vehicles as Force Multipliers*. Biarritz, France.

[Kriegel et al., 2009]

M. Kriegel, A. Schulte & P. Stütz (2009). Testbed Environment for Future UAV Automation Technologies. In: *Proceedings of the 2nd International Workshop on Aircraft System Technologies*. Hamburg, Germany, 26. - 27. März 2009.

[Laird et al., 1987]

J. Laird, R. Jones & P. Rosenbloom (1987). Soar: An Architecture for General Intelligence. In: *Artificial Intelligence*, 33. S. 1-64.

[Lambers et al., 2007]

K. Lambers, H. Eisenbeiss, M. Sauerbier, D. Kupferschmidt, T. Gaisecker, S. Sotoodeh & T. Hanusch (2007). Combining photogrammetry and laser scanning for the recording and modelling of the Late Intermediate Period site of Pinchango Alto, Palpa, Peru. *Journal of Archaeological Science*, 34(10), S. 1702-1712.

[Lange et al., 2008]

S. Lange, N. Sünderhauf & P. Protzel (2008). Autonomous Landing for a Multirotor UAV Using Vision. In: *International Conference on Simulation, Modeling and Programming for Autonomous Robots, SIMPAR 2008*. Venedig, Italien, 3. - 7. November 2008. S. 482-491.

[Lax & Sutherland, 1996]

Mark Lax & Barry Sutherland (1996). *An extended Role for unmanned aerial vehicles in the royal Australian Air Force, Report 46*. Air Power Studies Centre. Fairbairn, Australia.

[Lehman et al., 1998]

Jill Fain Lehman, John Laird & Paul Rosenbloom (1998). A Gentle Introduction to Soar, an Architecture for Human Cognition. In D. Scarborough & S. Sternberg (Hrsg.), *Invitation to Cognitive Science (Vol. 4)*. Cambridge, MA: MIT Press.

[Littlefield-Lawwill & Viswanathan, 2007]

J. Littlefield-Lawwill & R. Viswanathan (2007). Advancing open standards in Integrated Modular Avionics: An industry analysis. In: *IEEE/AIAA 26th Digital Avionics Systems Conference, DASC'07*. Columbia, MD, USA, 25. - 26. September 2007.

[Ludington et al., 2006]

B. Ludington, E. Johnson & G. Vachtsevanos (2006). Augmenting UAV autonomy. In: *IEEE Robotics & Automation Magazine*, 13(3). S. 63-71.

[Lunze, 2008]

J. Lunze (2008). *Regelungstechnik I: Systemtheoretische Grundlagen, Analyse und Entwurf einschleifiger Regelungen*. Springer.

[Manning et al., 2004]

Sharon D. Manning, Clarence E. Rash, Patricia A. LeDuc & Robert K. McKeon, Joseph Noback (2004). *The role of human causal factors in U.S. army unmanned aerial vehicle accidents*. U.S. Army Aeromedical Research Laboratory. Fort Rucker, AL, USA.

[Matzner, 2010]

Alexander Matzner (2010). *Weiterentwicklung einer Kognitiven Systemarchitektur auf Basis von Graphtransformationen*. Dissertation, Universität der Bundeswehr München, Fakultät für Luft- und Raumfahrttechnik.

[Matzner et al., 2008]

A. Matzner, M. Minas & A. Schulte (2008). Efficient graph matching with application to cognitive automation. In: *Proceedings of the 3rd International Workshop on Applications of Graph Transformation with Industrial Relevance, (AGTIVE '07)*. Leipzig, 12. Mai 2008.

[McCarley & Wickens, 2004]

J.S. McCarley & C.D. Wickens (2004). Human factors concerns in UAV flight. *Urbana-Champaign, IL: Institute of Aviation, University of Illinois*.

[Meitinger, 2008]

Claudia Meitinger (2008). *Kognitive Automation zur kooperativen UAV-Flugführung*. Dissertation, Universität der Bundeswehr München, Fakultät für Luft- und Raumfahrttechnik.

[Meitinger et al., 2009]

Claudia Meitinger, Gregor Jarasch & Axel Schulte (2009). Development of Artificial Cognitive Units in UAV Guidance. In: *NATO SCI-202 Symposium on Intelligent Uninhabited Vehicle Guidance Systems*. Neubiberg, 29. Juni - 1. Juli 2009.

[Meitinger & Schulte, 2006a]

Claudia Meitinger & Axel Schulte (2006a). Cognitive machine co-operation as basis for guidance of multiple UAVs. In: *NATO RTO HFM Symposium on Human Factors of Uninhabited Military Vehicles as Force Multipliers*. Biarritz, France. 9th-11th October 2006.

[Meitinger & Schulte, 2006b]

Claudia Meitinger & Axel Schulte (2006b). Human-Centred Automation for UAV Guidance: Oxymoron or Tautology? - The Potential of Cognitive and Co-operative Systems. In: *1st "Moving Autonomy Forward" Conference*. Grantham, UK, 21 - 22 June 2006.

[Meitinger & Schulte, 2007]

Claudia Meitinger & Axel Schulte (2007). Onboard Artificial Cognition as Basis for Autonomous UAV Co-operation. In: *GARTEUR AG FM-14* –

Autonomy in UAVs. Technical Proceedings GARTEUR/TP-163. Group for Aeronautical Research and Technology in Europe.

[Microdrones GmbH Homepage]

Microdrones GmbH Homepage. Online-Ressource:
<http://www.microdrones.com>. Letzter Zugriff: 11. März 2011.

[MicroPilot, 2005]

MicroPilot (2005). MicroPilot - World Leader in Small UAV Autopilots. Online-Ressource: <http://www.micropilot.com/pdf/mp2028xp.pdf> (Letzter Zugriff: 10 März 2011).

[XXL Modelhelicopter Homepage]

XXL Modelhelicopter Homepage. Online-Ressource: <http://www.xxl-modellhelicopter.com>. Letzter Zugriff: 18. Januar 2011.

[Mouloua et al., 2001]

M. Mouloua, R. Gilson, J. Kring & P. Hancock (2001). Workload, Situation Awareness, and Teaming Issues for UAV/UCAV Operations. In: *Proceedings of the Human Factors and Ergonomics society 45th Annual Meeting*. S. 162 - 165.

[Mussettola et al., 1998]

N. Mussettola, P.P. Nayak, B.M. Pell & B.C. Williams (1998). Remote agent: To boldly go where no AI system has gone before. In: *Artificial Intelligence, 103(1-2)*. S. 5-47.

[Newell, 1990]

Allen Newell (1990). *Unified Theories of Cognition*. Cambridge, MA. Harvard University Press.

[Newell & A., 1972]

Allen Newell & Simon Herbert A. (1972). *Human problem solving*. Englewood Cliffs, NJ. Prentice Hall.

[Northrop Grumman Corporation, 2006]

Northrop Grumman Corporation (2006). RQ-4 Block 20 Global Hawk. Online-Ressource:
http://www.as.northropgrumman.com/products/ghrq4b/assets/GH_Brochure.pdf (Letzter Zugriff: 09 Februar 2011).

[Northrop Grumman, 2010]

Northrop Grumman (2010). Northrop Grumman - MQ-8B Fire Scout. Online-Ressource:
http://www.as.northropgrumman.com/products/mq8bfirescout_navy/assets/firescout-new-brochure.pdf (Letzter Zugriff: 11 März 2011).

[Oh & Barlow, 2004]

C.K. Oh & G.J. Barlow (2004). Autonomous controller design for unmanned aerial vehicles using multi-objective genetic programming. In: *Congress on Evolutionary Computation, CEC 2004*. Portland, OR, USA, 20. - 23. Juni 2004. S. 1538-1545.

[Onken, 1994]

Reiner Onken (1994). Basic Requirements Concerning Man-Machine Interactions in Combat Aircraft. In: *Proceedings of the Workshop on Human Factors / Future Combat Aircraft*.

[Onken, 2002]

Reiner Onken (2002). Cognitive Co-operation for the Sake of the Human-Machine Team Effectiveness. In: *NATO RTO-Meeting Procedures MP-088, HFM-084: The Role of Humans in Intelligent and Automated Systems*. Warschau, Polen.

[Onken & Schulte, 2010]

Reiner Onken & Axel Schulte (2010). *System-Ergonomic Design of Cognitive Automation: Dual-Mode Cognitive Design of Vehicle Guidance and Control Work Systems*. Heidelberg. Springer.

[Oomkens et al., 2008]

W. Oomkens, M. Mulder, M.M. Van Paassen & M.H.J. Amelink (2008). UAVs as aviators: Environment skills capability for UAVs. In: *International Conference on Systems, Man and Cybernetics, SMC 2008*. Singapur, 12. - 15. Oktober 2009. S. 2426-2431.

[Parasuraman et al., 2000]

R. Parasuraman, T.B. Sheridan & C.D. Wickens (2000). A model for types and levels of human interaction with automation. In: *IEEE Transactions on Systems, Man and Cybernetics, Part A, 30(6)*. S. 286-297.

[Parrot S.A. Homepage]

Parrot S.A. Homepage. Online-Ressource: <http://ardrone.parrot.com/parrot-ardrone/de/>. Letzter Zugriff: 17. März 2011.

[Pastor et al., 2007]

E. Pastor, J. Lopez & P. Royo (2007). UAV payload and mission control hardware/software architecture. In: *IEEE Aerospace and Electronic Systems Magazine, 22(6)*. S. 3-8.

[Pearson, 1969]

Lee Pearson (1969). Developing the flying bomb. *Naval Aviation in World War I*.

[Pecher et al., 2010]

Wolfgang Pecher, Stefan Brüggewirth & Axel Schulte (2010). Using Cognitive Automation for Aircraft General Systems Management. In: *System of Systems Engineering Conference (SoSE'10)*. Loughborough, UK, 22.-24. Juni 2010.

[Perzanowski et al., 2003]

D. Perzanowski, D. Brock, W. Adams, M. Bugajska, A.C. Schultz, J.G. Trafton, S. Blisard & M. Skubic (2003). Finding the FOO: A pilot study for a multimodal interface. In: *IEEE International Conference on Systems, Man and Cybernetics*. 8. Oktober 2003. S. 3218-3223.

- [Pontech, 2004]
Pontech (2004). Pontech Homepage. Online-Ressource:
<http://www.pontech.com/files/HBC101%20Manual%20X1%20Complete.pdf>
(Letzter Zugriff: 11 Juli 2011).
- [Posner, 1989]
M.I. Posner (1989). *Foundations of Cognitive Science*. Cambridge,
Massachusetts. MIT Press.
- [Prang, 2009]
Dorothea Prang (2009). *Anwendung und Validierung eines Konzepts zum
Einstellen eines Autopiloten für ein Miniaturhubschrauber UAV*. Diplomarbeit,
Universität der Bundeswehr München, Fakultät für Luft- und Raumfahrttechnik.
- [Procerus Technologies, 2010]
Procerus Technologies (2010). Procerus Technologies: Kestrel Autopilot.
Online-Ressource:
http://www.procerusuav.com/Downloads/DataSheets/Kestrel_2.4.pdf (Letzter
Zugriff: 10 März 2011).
- [Proctor & Johnson, 2004]
A.A. Proctor & E.N. Johnson (2004). Vision-only aircraft flight control methods
and test results. In: *Proceedings of the AIAA Guidance, Navigation, and Control
Conference*. Providence, RI, USA, 16. - 19. August 2004.
- [Putzer, 2004]
Henrik Putzer (2004). *Ein uniformer Architekturansatz für kognitive Systeme
und seine Umsetzung in ein operatives Framework*. Dissertation, Universität der
Bundeswehr München, Fakultät für Luft- und Raumfahrttechnik. Berlin: Köster.
- [Putzer & Onken, 2003]
Henrik Putzer & Reiner Onken (2003). COSA - a generic cognitive system
architecture based on a cognitive model of human behaviour. In: *Cognition
Technology and Work*, 5. S. 140-151.
- [Quigley et al., 2005]
M. Quigley, M.A. Goodrich & R.W. Beard (2005). Semi-autonomous human-
UAV interfaces for fixed-wing mini-UAVs. In: *International Conference on
Intelligent Robots and Systems, IROS*. Edmonton, Alberta, Canada, 2. - 6.
August 2005. S. 2457-2462.
- [Rao & Georgeff, 1995]
A. Rao & M. Georgeff (1995). BDI Agents: From Theory to Practice. In:
*Proceedings of the First International Conference on Multi-Agent Systems
(ICMAS-95)*. San Francisco, CA, USA, 12.-14. Juni 1995.
- [Rasmussen, 1983]
Jens Rasmussen (1983). Skills, Rules, and Knowledge; Signals, Signs, and
Symbols, and other Distinctions in Human Performance Models. *IEEE
Transactions on Systems, Man and Cybernetics*, 13(3), S. 257-266.

[Rauschert et al., 2008]

Andreas Rauschert, Claudia Meitinger & Axel Schulte (2008). Experimentally Discovered Operator Assistance Needs in the Guidance of Cognitive and Cooperative UAVs. In: *Conference on Humans Operating Unmanned Systems, HUMOUS*. Brest, Frankreich, 3. - 4. September 2008.

[REFA, 1984]

REFA (1984). *Methodenlehre des Arbeitsstudiums*. München. Carl Hanser.

[Reidelstürz & Schulte, 2010]

Patrick Reidelstürz & Axel Schulte (2010). *Unbemanntes Kleinflugzeugsystem für den Einsatz in der Präzisionslandwirtschaft*. Universität der Bundeswehr München, Neubiberg.

[Reuder et al., 2009]

J. Reuder, P. Brisset, M. Jonassen, M. Muller & S. Mayer (2009). The Small Unmanned Meteorological Observer SUMO: A new tool for atmospheric boundary layer research. In: *Meteorologische Zeitschrift*, 18(2). S. 141-147.

[Roark, 1996]

C. Roark (1996). SAE AS4893 Generic Open Architecture (GOA) framework. In: *15th AIAA/IEEE Digital Avionics Systems Conference*. S. 217-222.

[Roberts et al., 2003]

J.M. Roberts, P.I. Corke & G. Buskey (2003). Low-cost flight control system for a small autonomous helicopter. In: *International Conference on Robotics and Automation, ICRA'03*. Taipei, Taiwan, 14. - 19. September 2003. S. 546-551.

[Rosenstiel et al., 2005]

Lutz von Rosenstiel, Walter Molt & Bruno Rüttinger (2005). *Organisationspsychologie*. Stuttgart. Kohlhammer.

[Rumelhart, 1986]

D.E. Rumelhart (1986). The Architecture of Mind: A Connectionist Approach. In: *[Posner, 1989]*. S. 133-159.

[Russel & Norvig, 2003]

Stuart Russel & Peter Norvig (2003). *Artificial Intelligence - A Modern Approach*. Upper Saddle River, New Jersey. Prentice Hall.

[Sagahyroon et al., 2005]

A. Sagahyroon, M.A. Jarah & A.A.M. Hadi (2005). Design and implementation of a low cost UAV controller. In: *IEEE International Conference on Industrial Technology*. Hong Kong, 14. - 17. Dezember 2005. S. 1394-1397.

[Sarris, 2001]

Zak Sarris (2001). Survey of UAV applications in civil markets. In: *IEEE Mediterranean Conference on Control and Automation*. Dubrovnik, Kroatien, 27. - 29. Juni 2001. S. 11.

[Sarter et al., 1997]

N.B. Sarter, D.D. Woods & C.E. Billings (1997). Automation surprises. In: G. Salvendy (Hrsg.) *Handbook of Human Factors & Ergonomics*. 2. Hoboken, NJ: John Wiley. S. 1926 - 1943.

[Sasa et al., 2008]

S. Sasa, Y. Matsuda, M. Nakadate & K. Ishikawa (2008). Ongoing research on disaster monitoring UAV at JAXA's Aviation Program Group. S. 978-981.

[Schaefer, 2003]

R. Schaefer (2003). *Unmanned aerial vehicle reliability study*. Office of the Secretary of Defense, Washington, DC.

[Schouwenaars et al., 2005]

T. Schouwenaars, M. Valenti, E. Feron & J. How (2005). Implementation and flight test results of MILP-based UAV guidance. In: *IEEE Aerospace Conference*. Big Sky, MT, USA, 5. - 12. März 2005. S. 1-13.

[Schulte et al., 2008]

Axel Schulte, Claudia Meitinger & Reiner Onken (2008). Human Factors in the Guidance of Unmanned Vehicles: Oxymoron or Tautology? The Potential of Cognitive and Co-operative Automation. In: *International Journal of Cognition Technology and Work*. Heidelberg: Springer ISSN 1435-5558.

[Schwarzbach et al., 2009]

M. Schwarzbach, U. Putze, U. Kirchgaessner & M. Schoenermark (2009). Acquisition of High Quality Remote Sensing Data Using a UAV Controlled by an Open Source Autopilot. In: *Proceedings of the ASME 2009 International Design Engineering Technical Conferences & Computers and Information Engineering Conference*. San Diego, Kalifornien, USA.

[Schwarzer et al., 2007]

R. Schwarzer, F. Adolf & H. Gräbe (2007). Testverfahren für eventgetriebene Software am Beispiel der ARTIS Plattform. *Modellbasierte Software-Entwicklung für eingebettete Systeme* S. 127-141.

[Scribner & Dahn, 2008]

D.R. Scribner & D. Dahn (2008). *A Comparison of Soldier Performance in a Moving Command Vehicle Under Manned, Teleoperated, and Semi-Autonomous Robotic Mine Detector System Control Modes*. Human Research and Engineering Directorate, U.S. Army Research Laboratory. Aberdeen Proving Ground, MD.

[Sharp, 2008]

Robin Sharp (2008). *Principles of Protocol Design*. Berlin, Heidelberg. Springer.

[Sharp et al., 2001]

C.S. Sharp, O. Shakernia & S.S. Sastry (2001). A vision system for landing an unmanned aerial vehicle. In: *International Conference on Robotics and Automation, ICRA*. Seoul, Korea, 21. - 26. Mai 2001. S. 1720--1727.

[Sheridan, 1992]

Thomas B. Sheridan (1992). *Telerobotics, automation, and human supervisory control*. Cambridge, Massachusetts. The MIT Press.

[Shima et al., 2007]

T. Shima, S. Rasmussen & D. Gross (2007). Assigning micro UAVs to task tours in an urban terrain. In: *IEEE Transactions on Control Systems Technology*. S. 601-612.

[Simpson et al., 2004]

B.D. Simpson, R.S. Bolia & M.H. Draper (2004). Spatial audio display concepts supporting situation awareness for operators of unmanned aerial vehicles. *Human performance, situation awareness and automation: current research and trends: HPSAA II* S.61.

[simWerk Homepage]

simWerk Homepage. Online-Ressource: <http://www.simwerk.de/reflex-flugsimulator/>. Letzter Zugriff: 30. Juni 2011.

[Sinha et al., 2005]

A. Sinha, T. Kirubarajan & Y. Bar-Shalom (2005). Autonomous ground target tracking by multiple cooperative UAVs. In: *IEEE Aerospace Conference*. Big Sky, MT, USA, 5. - 12. März 2005. S. 1-9.

[Sinopoli et al., 2001]

B. Sinopoli, M. Micheli, G. Donato & T.J. Koo (2001). Vision based navigation for an unmanned aerial vehicle. In: *IEEE International Conference on Robotics and Automation, ICRA*. S. 1757-1764.

[Skinner, 1957]

B.F. Skinner (1957). *Verbal Behavior*. New Jersey. Prentice-Hall.

[SNC - Sierra Nevada Corporation, 2006]

SNC - Sierra Nevada Corporation (2006). SNC - Sierra Nevada Corporation | Tactical Automatic Landing System (TALS). Online-Ressource: <http://www.sncorp.com/prod/cnsatm/includes/pdf/TALS%20Product%20Sheet.pdf> (Letzter Zugriff: 21 März 2011).

[Spura et al., 2004]

Thomas M. Spura, Jerry L. Franke, Robert J. Szczerba & Mike Scarangella (2004). Command and Control of Unmanned Vehicles - where are we going? In: *American Helicopter Society 60th Annual Forum*. Baltimore, MD.

[Strenzke & Schulte, 2011]

Ruben Strenzke & Axel Schulte (2011). Mixed-Initiative Multi-UAV Mission Planning by Merging Human and Machine Cognitive Skills. In: *HCI International*. Orlando, Florida, USA, 9.-14. Juli 2011.

[Strenzke & Schulte, 2011]

Ruben Strenzke & Axel Schulte (2011). The MMP: A Mixed-Initiative Mission Planning System for the Multi-Aircraft Domain. In: *SPARK@ICAPS*. Freiburg, Deutschland, 13. Juli 2011.

[Strenzke et al., 2011]

Ruben Strenzke, Johann Uhrmann, Andreas Benzler, Felix Maiwald, Andreas Rauschert & Axel Schulte (2011). Managing Cockpit Crew Excess Task Load in Military Manned-Unmanned Teaming Missions by Dual-Mode Cognitive Automation Approaches. In: *AIAA Guidance, Navigation, and Control (GNC) Conference*. Portland, Oregon. 8-11 August 2011.

[Sullivan, 2006]

Jeffrey M. Sullivan (2006). Evolution or Revolution? Rise of UAVs. *IEEE Technology and Society Magazine*, 25(3), S. 43-46.

[Sullivan et al., 2004]

D. Sullivan, J. Totah, S. Wegener, F. Enomoto, C. Frost, J. Kaneshige & J. Frank (2004). Intelligent mission management for uninhabited aerial vehicles. In: *Proceedings of SPIE. Remote Sensing Applications of the Global Positioning System*. Eds. Michael Bevis, Yoshinori Shoji, Steven Businger. Vol. 5661.

[Taamallah et al., 2005]

S. Taamallah, A.J.C. de Reus & J.F. Boer (2005). Development of a rotorcraft mini-uav system demonstrator. In: *The 24th Digital Avionics Systems Conference, DASC 2005*. Washington, DC, USA, 30. Oktober - 3. November 2005.

[Tachi et al., 2003]

S. Tachi, K. Komoriya, K. Sawada, T. Nishiyama, T. Itoko, M. Kobayashi & K. Inoue (2003). Telexistence cockpit for humanoid robot control. In: *Advanced Robotics*, 3(17). S. 199-217.

[Tadema et al., 2006]

J. Tadema, G.J.M. Koeners & E. Theunissen (2006). Synthetic vision to augment sensor-based vision for remotely piloted vehicles. In: *Proceedings of SPIE*. Orlando, FL, USA, 19. April 2006.

[Tadema & Theunissen, 2003]

J. Tadema & E. Theunissen (2003). Feasibility of using synthetic vision technology for UAV operator support. In: *The 22nd Digital Avionics Systems Conference, DASC'03*. Indianapolis, IN, USA, 12. - 16. Oktober 2003. S. 8-81.

[Tanenbaum, 2009]

Andrew S. Tanenbaum (2009). *Moderne Betriebssysteme*. München. Pearson Studium.

[Theunissen et al., 2005]

E. Theunissen, A. Goossens, O.F. Bleeker & G.J.M. Koeners (2005). UAV Mission Management Functions to Support Integration in a Strategic and Tactical ATC and C 2 Environment. In: *Proceedings of the AIAA Modeling and Simulation Technologies Conference*. San Francisco, CA, USA, 15. - 18. August 2005.

[Tisdale et al., 2009]

J. Tisdale, Z. Kim & J. Hedrick (2009). Autonomous UAV path planning and estimation. In: *IEEE Robotics & Automation Magazine*, 16(2). S. 35-42.

[Trouvain & Schlick, 2007]

B. Trouvain & C. Schlick (2007). A comparative study of multimodal displays for multirobot supervisory control. *Engineering Psychology and Cognitive Ergonomics* S. 184-193.

[Tvaryanas, 2006]

Anthony P. Tvaryanas (2006). *Human Factors Considerations in Migration of Unmanned Aircraft System (UAS) Operator Control*. U.S. Air Force. Brooks City-Base.

[Tvaryanas et al., 2005]

Anthony P. Tvaryanas, William T. Thompson & Stefan H. Constable (2005). The U.S. Military Unmanned Aerial Vehicle (UAV) Experience: Evidence-Based Human System Integration Lessons Learned. In: *Strategies to Maintain Combat Readiness during Extended Deployments - A Human Systems Approach*. Neuilly-sur-Seine, France. S. 5-1 - 5-24.

[Uhrmann et al., 2009]

Johann Uhrmann, Ruben Strenzke, Andreas Rauschert & Axel Schulte (2009). Manned-unmanned teaming: Artificial cognition applied to multiple UAV guidance. In: *NATO SCI-202 Symposium on Intelligent Uninhabited Vehicle Guidance Systems*. Munich, Germany.

[Uhrmann et al., 2010a]

Johann Uhrmann, Ruben Strenzke & Axel Schulte (2010a). Human Supervisory Control of Multiple UAVs by use of Task Based Guidance. In: *Conference on Humans Operating Unmanned Systems (HUMOUS'10)*. Toulouse, France.

[Uhrmann et al., 2010b]

Johann Uhrmann, Ruben Strenzke & Axel Schulte (2010b). Task-based Guidance of Multiple Detached Unmanned Sensor Platforms in Military Helicopter Operations. In: *COGIS (COGNitive systems with Interactive Sensors)*. Crawley, UK, 22. November 2010.

[Unnikrishnan & Balakrishnan, 2006]

N. Unnikrishnan & SN Balakrishnan (2006). Neuroadaptive model following controller design for a nonaffine UAV model. In: *American Control Conference*. Minneapolis, MN, USA, 14. - 16. Juni 2006.

[USAA & MC, 2000]

USAA & MC (2000). ADS-33E-PRF. Aeronautical Design Standard. Performance specification – Handling qualities requirements for military rotorcraft. Online-Ressource: <http://www.everyspec.com/ARMY/ADS+-+Aero+Design+Std/download.php?spec=ADS-33E-PRF.003614.pdf> (Letzter Zugriff: 20 Juli 2011).

[Vachtsevanos et al., 2004]

G. Vachtsevanos, L. Tang, G. Drozeski & L. Gutierrez (2004). Intelligent control of unmanned aerial vehicles for improved autonomy and reliability. In: *Proceedings of the 5th IFAC Symposium on Intelligent Autonomous Vehicles*. Lissabon, Portugal, 5. - 7. Juli 2004.

[Valenti et al., 2004]

M. Valenti, T. Schouwenaars, Y. Kuwata, E. Feron, J. How & J. Paunicka (2004). Implementation of a manned vehicle-UAV mission system. In: *AIAA Guidance, Navigation, and Control Conference and Exhibit*. Providence, RI, USA, 16. - 19. August 2004.

[van Erp, 2000]

J.B.F. van Erp (2000). Controlling Unmanned Vehicles: the Human Factors Solution. In: *Warfare Automation: Procedures and Techniques for Unmanned Vehicles*. Ankara, Turkey.

[van Erp, 2002]

J.B.F. van Erp (2002). Guidelines for the use of vibro-tactile displays in human computer interaction. In: *Proceedings of Eurohaptics*. Edinburgh, UK. S. 18-22.

[van Erp & Van Veen, 2003]

J.B.F. van Erp & H. Van Veen (2003). A multipurpose tactile vest for astronauts in the international space station. In: *Proceedings of Eurohaptics*. Dublin, Irland, 6. - 9. Juli 2003. S. 405-408.

[Watanabe et al., 2007]

Y. Watanabe, A.J. Calise & E.N. Johnson (2007). Vision-based obstacle avoidance for UAVs. In: *AIAA Guidance, Navigation and Control Conference and Exhibit*. Hilton Head, SC, USA, 20. - 23. August 2007.

[Weigmann, 2003]

D.A. and Shappell, S.A. Weigmann (2003). *A human error approach to aviation accident analysis: The human factors analysis and classification system*. Burlington, VT. Ashgate Pub Ltd.

[Welch & Bishop, 1995]

G. Welch & G. Bishop (1995). An introduction to the Kalman filter. *University of North Carolina at Chapel Hill*, 7(1).

[Wiener, 1989]

E.L. Wiener (1989). Human factors of advanced technology ("glass cockpit") transport aircraft. (NASA Contractor Report No 177528). *Moffett Field, CA: NASA-Ames Research Center*.

[Williams, 2004]

Kevin.W. Williams (2004). *A Summary of Unmanned Aircraft Accident/Incident Data: Human Factors Implications*. Oklahoma City.

[Wills et al., 2002]

L. Wills, S. Kannan, S. Sander, M. Guler, B. Heck, JVR Prasad, D. Schrage & G. Vachtsevanos (2002). An open platform for reconfigurable control. In: *Control Systems Magazine*, 21(3). S. 49-64.

[Wise & Rysdyk, 2006]

R.A. Wise & R.T. Rysdyk (2006). UAV coordination for autonomous target tracking. In: *Proceedings of the AIAA Guidance, Navigation, and Control Conference*. Keystone, CO, USA, 21. - 24. August 2006.

[Wong, 2001]

KC Wong (2001). UAV Design Activities in a University Environment. In: *9th Australian International Aerospace Congress*. Canberra, Australien, 6. - 8. März 2001.

[Wyeth et al., 2000a]

G. Wyeth, G. Buskey & J. Roberts (2000a). Flight control using an artificial neural network. In: *Proceedings of the Australian Conference on Robotics and Automation, ACRA*. Melbourne, Australien, 30. August - 1. September 2000. S. 65-70.

[Wyeth et al., 2000b]

G. Wyeth, G. Buskey & J. Roberts (2000b). Vision-based obstacle avoidance for UAVs. In: *Proceedings of the Australian Conference on Robotics and Automation, ACRA*. Melbourne, Australien, 30. August - 1. September 2000. S. 65-70.

[Xsens Homepage]

Xsens Homepage. Online-Ressource: <http://www.xsens.com/en/general/mti-g>.
Letzter Zugriff: 22. September 2011.

[Yamaha Motor Co., Ltd., 2004]

Yamaha Motor Co., Ltd. (2004). Yamaha Motor. Online-Ressource: <http://www.yamaha-motor.co.jp/global/ir/material/pdf/2003/2003factbook-e.pdf>
(Letzter Zugriff: 11 März 2011).

B Abbildungsverzeichnis

Abbildung 2-1. Schematischer Aufbau eines Führungs- und Missionssystems eines UAVs	6
Abbildung 2-2. Industrie-Hubschrauber RMAX der Firma Yamaha, [Yamaha Motor Co., Ltd., 2004].	7
Abbildung 2-3. UAV Forschungsdemonstrator der Universität der Bundeswehr München (UniBwM) auf Basis eines Modellflugzeugs.	8
Abbildung 2-4. Kommerzielle Autopilotensysteme: Links der Kestrel Autopilot der Firma Procerus [Procerus Technologies, 2010]; Rechts das System Piccolo II der Firma Cloudcap Technologies [Cloud Cap Technologies, 2006]	9
Abbildung 2-5. Lagestabilisierung eines Mini-UAV Hubschraubers mit einem ANN nach [Buskey et al., 2005]	10
Abbildung 2-6: Links: Hobby-IMU-Bord mit Mikrocontroller der Firma Sparkfun [Sparkfun Electronics Homepage] Rechts: COTS IMU mit integriertem GPS-Empfänger der Firma XSens [Xsens Homepage]	11
Abbildung 2-7. Echtzeitvideo eines UAVs mit überlagerten, computergenerierten Symbolen [Draper et al., 2006].	14
Abbildung 2-8. Links: Arbeitsplätze innerhalb der Bodenkontrollstation für das System Predator (Bild von General Atomics Aeronautical Systems [General Atomics Aeronautical Systems, Inc., 2011]) Rechts: Tragbare Bodenkontrollstation für das System Aladin (Bild von EMT Penzberg [EMT Ingenieurgesellschaft, 2009])	15
Abbildung 2-9. Der Mensch als Überwacher und Manager vieler unabhängiger Subsysteme und Automationsfunktionen	21
Abbildung 2-10. Teufelskreis der konventionellen Automation	22
Abbildung 2-11. Das Arbeitssystem	28
Abbildung 2-12. Einbringen von kognitiver Automation in das Arbeitssystem	29
Abbildung 2-13. Semi-autonomes System als Arbeitsmittel eines übergeordneten Arbeitssystems	30
Abbildung 2-14. Der Kognitive Prozess	31
Abbildung 2-15. Sichtweisen auf den Kognitiven Prozess	32
Abbildung 2-16: Überblick über die Kognitive Systemarchitektur COSA nach [Meitinger, 2008]	33
Abbildung 2-17. Interpretation des Modells von Rasmussen zur menschlichen Informationsverarbeitung auf drei Ebenen [Onken & Schulte, 2010]	35
Abbildung 2-18. Ansatz des Vorausplanungsalgorithmus in COSA ²	36
Abbildung 2-19. Arbeitssystem eines bemannten Flugzeugs	37
Abbildung 2-20. Arbeitssystem zur UAV-Führung mit einem semi-autonomen UAV System als Teil der Arbeitsmittel	38
Abbildung 2-21. Kategorisierung aktueller und abgeschlossener Forschungsprojekte der UniBwM im Hinblick auf die Echtheitsgrade der Hardwaresysteme und der Missionselemente	40
Abbildung 3-1. CCV-F 104-G Erprobungsträger, Bild: Peter Mühlböck collection	46
Abbildung 3-2. KCV Grobkonzept	49
Abbildung 3-3. Interaktion zwischen missionsspezifischem und maschinenspezifischem Wissen über die KCV – Organisationsstruktur	51

Abbildung 3-4. Aufbau des KCV-Konzeptes mit dem Paradigma des Kognitiven Prozesses	54
Abbildung 4-1. Systemschaubild der Integration der KCV-ACU in die wesentlichen Komponenten eines UAVs	57
Abbildung 4-2. CPL-Codebeispiel	60
Abbildung 4-3. Beispiel für einen Ausschnitt aus dem situativen Wissen	62
Abbildung 4-4. Kombination verschiedener Wissenspakete nach [Putzer, 2004]	64
Abbildung 4-5. Erweiterung der Kombination von Wissenspaketen nach [Putzer, 2004] mit Berücksichtigung von maschinenspezifischem Wissen und der Hardware nach dem Konzept eines KCV	66
Abbildung 4-6. Aufbau der Organisationsstruktur der KCV-Schicht im Situationswissen	67
Abbildung 4-7. Situationswissen des Modells FCC	68
Abbildung 4-8. Situationswissen des Modells Sensordatenerfassungssystem (SDSYS)	68
Abbildung 4-9. Situationswissen des Modells Stromversorgung (PowerSupply)	69
Abbildung 4-10. Situationswissen des Modells Datenfunkverbindung (Datalink)	69
Abbildung 4-11. Situationswissen des Modells Inertialnavigationssystem (IMU)	70
Abbildung 4-12. Situationswissen des Modells Autopilot	70
Abbildung 4-13. Aufbau eines Systemschaltbilds der Avionikkomponenten	72
Abbildung 4-14. Abstraktion der Fähigkeiten eines UAV und deren Einordnung in die KCV-Schicht	73
Abbildung 4-15. Beispiel eines Ausschnitts aus dem Situationswissen	75
Abbildung 4-16. Beispiel für das Abfliegen einer Wegpunktroute und das Weiterschalten des Wegpunkts	75
Abbildung 4-17. Anfordern von gewünschtem Verhalten und die nachfolgende Prüfung der Ressourcen und die Konfiguration der Automation	76
Abbildung 5-1. Demonstratorumgebungskonzept	83
Abbildung 5-2. Simulation Flugdynamik und der Sensordaten des UAVs	85
Abbildung 5-3. Der UAV Demonstrator BIG I im Flug	86
Abbildung 5-4. Avionikkonzept des UAV-Demonstrators BIG I	88
Abbildung 5-5. Aufbau eines Akkupacks (links) und die Verschaltung von zwei Packs für die Stromversorgung für die Avionikkomponenten (rechts)	92
Abbildung 5-6. Entkopplung der externen Stromversorgung mittels Schottky-Diode ...	92
Abbildung 5-7. FCC auf Basis des Diamond Athena, Formfaktor PC-104 (links) und MMC auf Basis des Commell LS-373, Formfaktor 3,5“ (rechts)	93
Abbildung 5-8. Das verwendete Trägheitsnavigationssystem NAV420 der Firma Crossbow	94
Abbildung 5-9. Seriellles Datenfunkmodem Sateline 3AS(d) der Firma Satel	95
Abbildung 5-10. Verschaltung der Datenfunkmodems	96
Abbildung 5-11. Mechanischer Aufbau und Platzierung der Avionikkomponenten in der Avionikbox	97
Abbildung 5-12. Die montierte Avionikbox zwischen den Landekufen des UAV-Demonstrators BIG I	98
Abbildung 5-13. Übertragung der Steuerdaten eines Modellbausenders zu einem Modellbauempfänger auf dem 35MHz-Band	98
Abbildung 5-14. Pulsweiten moduliertes Signal für die Ansteuerung von Modellbauservos	99
Abbildung 5-15. Einsatz und Verschaltung des HF-Twin und DPSI Twin der Firma Emcotec	100
Abbildung 5-16. Schematische, kaskadierte Verschaltung der DPSI-Twins	101

Abbildung 5-17. Schaltplan des umgesetzten Sicherheitskonzepts mit zwei kaskadierten DPSI-Twins	102
Abbildung 5-18. Kaskadierter Aufbau der vier Regelkreise des FCS nach [Höcht, 2007]	103
Abbildung 5-19. Horizontaler Flugpfad des UAVs mit den horizontalen Positionsmodi Pos_H1 (links) und Pos_H2 (rechts) nach [Höcht, 2007]	106
Abbildung 5-20. Vertikaler Flugpfad des UAVS mit den vertikalen Positionsmodi Pos_V1 (links) und Pos_V2 (rechts) nach [Höcht, 2007]	107
Abbildung 5-21. Ausrichtung der Längsachse des Fluggerätes durch verschiedene FCS-Betriebsarten	107
Abbildung 5-22. Mobile Bodenkontrollstation auf Basis eines Mercedes Sprinters....	111
Abbildung 5-23. Arbeitsplatz des Operateurs innerhalb der Bodenkontrollstation.....	112
Abbildung 5-24. Umsetzung des Hardwarekonzepts der Bodenkontrollstation.....	113
Abbildung 5-25. Prozessstruktur und Verteilung der Daten innerhalb der BKS.....	114
Abbildung 5-26. Der Prozess TermIO mit der gewählten Anzeigeart: Kartendisplay (engl.: Map)	116
Abbildung 5-27. Informationsaustausch zwischen der MBKS und dem UAV	117
Abbildung 5-28. Nominelle Übertragung von Daten zwischen dem Sender und dem Empfänger.....	118
Abbildung 5-29. Erneuter Versand einer Nachricht, wenn Übertragungsfehler erkannt werden.....	119
Abbildung 5-30. Erneuter Versand einer Nachricht nach einer vordefinierten Zeitperiode, wenn diese nicht durch den Empfänger bestätigt wird.....	119
Abbildung 5-31. Modularer Aufbau des Softwareprozesses COMC	120
Abbildung 5-32. Mechanismus des ARQ-Protokolls	123
Abbildung 6-1. Vorgeplante Flugroute der Aufklärungsmission	126
Abbildung 6-2. Missionsdokument für die Evaluationsmission der KCV-ACU	126
Abbildung 6-3. KCV-ACU Evaluationsflug – Flugabschnitt ab Umschaltung auf Automatik bis zum Erreichen des ersten Wegpunkts	129
Abbildung 6-4. KCV-ACU Evaluationsflug – Flugabschnitt Wegpunkt 1 bis Wegpunkt 3	131
Abbildung 6-5. Flugabschnitt Wegpunkt 3 bis Wegpunkt 4 mit Ausrichtung des Heading des UAVs auf den erdfesten Punkt	133
Abbildung 6-6. Flugabschnitt Wegpunkt 4 bis Wegpunkt 6	134
Abbildung 6-7. Anzeige von Nachrichten der KCV-ACU im Map-Display des BKS TermIO-Prozesses.....	136
Abbildung 6-8. Gesamter Ground-Track des KCV-Versuchsflugs mit Headingvektoren zwischen den Wegpunkten 3 und 4	137
Abbildung 6-9. Durchführung der Evaluationsmission in der Simulationsumgebung. 138	
Abbildung 6-10. Simulierter Versuchsflug mit technischen Problemen bei der Avionikstromversorgung	143

C Abkürzungsverzeichnis

ACU	Artificial Cognitive Unit (künstliche kognitive Einheit)
AEF	Autonomy Enabling Functions (Autonomie unterstützende Funktionen)
AHRS	Attitude & Heading Reference System (Fluglagen- und Flugrichtungsreferenzsystem)
ANN	Artificial Neural Network (künstliches neuronales Netz)
API	Application Programming Interface (Programmierschnittstelle)
BDA	Battle Damage Assessment (Schadensbewertung und Wirkaufklärung)
BDI	Belief-Desire-Intention (Theorie über Software-Agenten)
BKS	Bodenkontrollstation
COS	Common Operating System (Generisches Betriebssystem)
COSA	Cognitive System Architecture (kognitive Systemarchitektur)
COTS	Commercial Off The Shelf (kommerziell verfügbare Produkte)
CPL	Cognitive Programming Language (kognitive Programmiersprache)
CSF	Cognitive Sub Function (kognitive Subfunktion)
DPSI	Dual Power Servo Interface (Servoschnittstelle mit Akkuweiche)
DSP	Digital Signal Processor (Digitaler Signalprozessor)
EO	Electro-Optical (Elektrooptisch)
FCC	Flight Control Computer (Flugregelungsrechner)
FCS	Flight Control System (Flugregelungssystem)
FMS	Flight Management System (Flugmanagementsystem)
GPS	Global Positioning System (globales Positionsbestimmungssystem)
GUI	Graphical User Interface (Graphische Benutzerschnittstelle)
HMI	Human Machine Interface (Mensch-Maschine-Schnittstelle)
IMU	Inertial Measurement Unit (Trägheitsnavigationssystem)
IR	Infrared (Infrarot)
ITOW	Interval Time Of Week (Internationale Zeit der Woche)
KCV	Knowledge Configured Vehicle (Wissenbasierte Konfiguration eines Fluggerätes)
KP	Kognitiver Prozess

LIDAR	Light Detection And Ranging (Abstands- und Geschwindigkeitsmessung mittels Licht)
LOS	Line Of Sight (Sichtverbindung)
LTM	Long Term Memory (Langzeitspeicher)
MEMS	Micro-Electro-Mechanical System (Mikroelektromechanisches System)
MM	Mission Management (Missionsmanagement)
MMC	Mission Management Computer (Missionsmanagementcomputer)
MMF	Missionsmanagementfunktionen
MMS	Mission Management System (Missionsmanagementsystem)
MSE	Mean Square Error (Mittlerer quadratischer Fehler)
NED	North-East-Down (Koordinatensystem, Nord-Ost-Abwärts)
OCU	Operating Cognitive Unit (kognitive Einheit)
OF	Operating Force (Bediener)
OSM	Operation Supporting Means (Arbeitsmittel)
RADAR	Radio Detection And Ranging (Funkortung)
RC	Radio Controlled (Ferngesteuert)
SAR	Synthetic Aperture Radar (Radar mit synthetischer Apertur)
SCU	Supporting Cognitive Unit (kognitive Einheit)
SOAR	State Operator And Result (kognitive Architektur, Produktionensystem)
UAV	Uninhabited Aerial Vehicle (unbemanntes Fluggerät)
UCAS	Uninhabited Combat Aerial Vehicle (unbemanntes Kampfflugzeug)
WM	Working Memory (Arbeitsspeicher)