IBS Workshop Micro Air Vehicle Technologie – Konzepte und Anwendungen 2019

Prof. Dr. Wolfram Hardt

TUD*press*

IBS Scientific Workshop Proceedings

Herausgegeben von Stiftung IBS, Wolfram Hardt

Band 9

IBS Workshop Micro Air Vehicle Technologie – Konzepte und Anwendungen 2019

Prof. Dr. Wolfram Hardt (Hrsg.)

TUD*press*
2020

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.d-nb.de abrufbar.

Bibliographic information published by the Deutsche Nationalbibliothek
The Deutsche Nationalbibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data are available in the Internet at http://dnb.d-nb.de.

ISBN 978-3-95908-201-3

© 2020 TUDpress
THELEM Universitätsverlag
GmbH und Co. KG
http://www.tudpress.de

Alle Rechte vorbehalten. All rights reserved.
Gesetzt von den Herausgebern.
Printed in Germany.

IBS Scientific Workshop Proceedings

Die Stiftung Internationales Informatik- und Begegnungszentrum Sachsen (IBS) fördert die interdisziplinäre Vernetzung von Informatik-Expertisen auf nationaler und internationaler Ebene. Für wissenschaftliche Begegnungen und fachlichen Austausch bietet die Stiftung IBS ein ideales Umfeld. Die Stiftung IBS koordiniert und organisiert Konferenzen, Workshops und Seminare zur Förderung des wissenschaftlichen Dialogs.

Die wissenschaftliche Schriftenreihe *IBS Scientific Workshop Proceedings* veröffentlicht begutachtete Beiträge der IBS-Veranstaltungen. Die einzelnen Bände der wissenschaftlichen Schriftenreihe sind auch online auf der Webseite der Stiftung IBS kostenfrei als open access Publikation erhältlich.

Die *IBS Scientific Workshop Proceedings* stellen dauerhaft die präsentierten Ergebnisse der IBS- Veranstaltungen bereit.

Vorwort

Der vorliegende Band der *IBS Scientific Workshop Proceedings* veröffentlicht die Beiträge zum IBS Workshop Micro Air Vehicle Technologie – Konzepte und Anwendungen vom November 2019. Micro Air Vehicle oder Air Robots erobern den industriellen Markt für Luftbeobachtung, Inspektion aus der Luft bis hin zur Paketauslieferung aus der Luft. Die Anwendungen für diese unbemannten Flugobjekte in Modellbaugröße sind so vielfältig, dass der Markt für Hersteller, Anwender und Piloten rasant wächst. Erhöhung der Betriebssicherheit, Verbesserung der Servicequalität und Reduzierung von Wartungskosten sind einige wichtige Faktoren für diese Marktentwicklung. Die Stiftung IBS fördert Zusammenarbeit und Nachwuchsentwicklung und initiierte 2019 bereits zum 3. Mal den IBS Workshop Micro Air Vehicle Technologie und führte Experten aus Bereichen der Air Robots zusammen. Mit Keynotes, Fachvorträgen und Demonstrationen wurden neueste Entwicklungen vorgestellt.

Editorial Board

- Dr. Ariane Heller
  Technische Universität Chemnitz
- Dr. André Meisel
  Technische Universität Chemnitz
- Prof. Dr. Wolfram Hardt, Editor-in-Chief
  Technische Universität Chemnitz, Stiftung IBS

www.ibs-laubusch.de

IBS Scientific Workshop Proceedings

IBS Workshop Micro Air Vehicle Technologie – Konzepte und Anwendungen 2019

Inhalt

APOLI - Adaptive Mission for Insulator Detection…….…….....3
Battseren B. und Harradi R., TU Chemnitz

Urbane „Letzte Meile“ – Transport mittels Lieferdrohne…….……13
Gageik N. und Bihl M.., Emqopter GmbH

Mit High-Tech gegen Fachkräftemangel: Wie ein aufstrebendes Start Up aus Bayern mit Drohnentechnik frischen Wind in die Ausbildung von Fachkräften bringt……..17
Gageik N. und Benz P., Emqopter GmbH

Geschäftsmodellentwicklung für Start-ups und Finanzplanung……...21
Geißler M., TU Chemnitz und Gränitz D., adcius Vermögenstreuhand GmbH

Experience Report on Reuse in Avionics..…… …….....28
Reich M., und Schweiger A. und Lorenz J., Airbus Defence and Space GmbH und Margull U., Technische Hochschule Ingolstadt

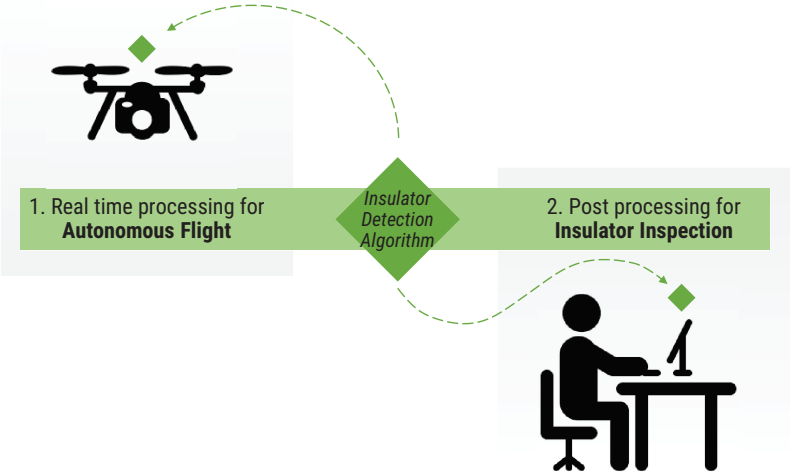
Thermografie mit Drohnen…….……..……...36
Schneider D., Hochschule Ansbach

Professur Technische Informatik
Prof. Dr. Wolfram Hardt

APOLI - Adaptive Mission for Insulator Detection

Batbayar Battseren, MSc.
Reda Harradi, MSc.

Insulator Detection
Why do we need it?



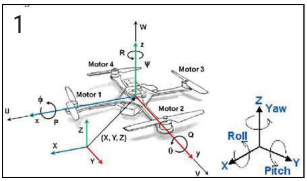
Insulator Detection
Use case 1: Autonomous Flight

Position sensor data based navigation

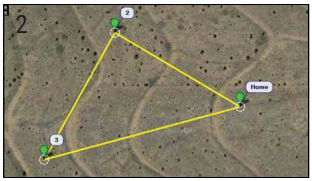
1. Best and built-in solution for flight controller
2. Pre-defined and fixed flight waypoints or paths

Visual data based navigation

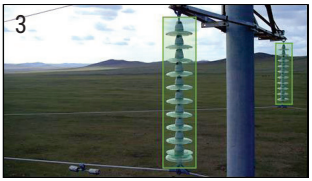
3. Real-time calculated dynamic flight paths



DOI:10.1016/j.ifacol.2016.11.118

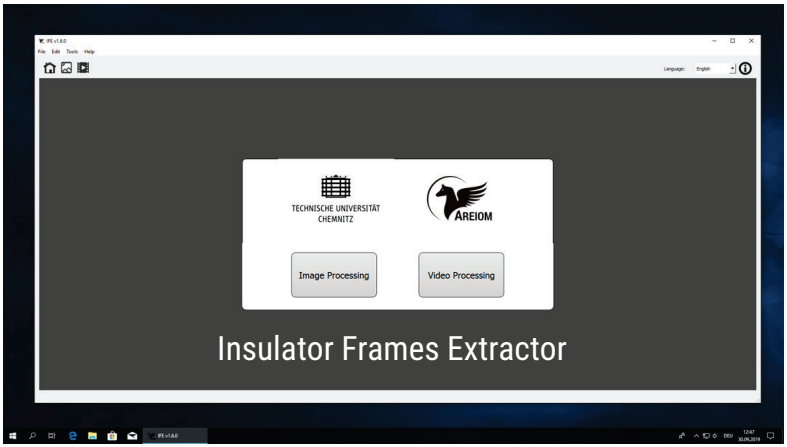


http://ardupilot.org



TECHNISCHE UNIVERSITÄT CHEMNITZ
Fakultät für Informatik
Professur Technische Informatik

Insulator Detection
Use case 2: Insulator Inspection

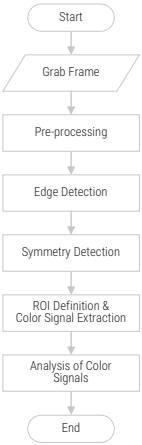


Insulator Detection
Methodology

- **Deep Learning**
  - Requires huge number of training dataset
  - Computationally expensive
  - Confident Results

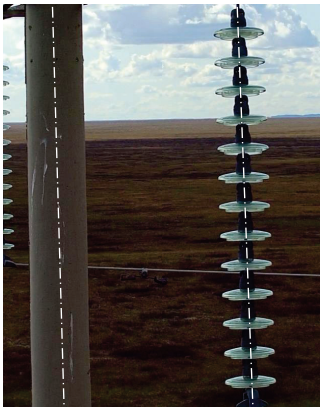
✓**Image Processing**
  - Needs small amount of sample data
  - Can be computationally inexpensive

Insulator Detection
Algorithms

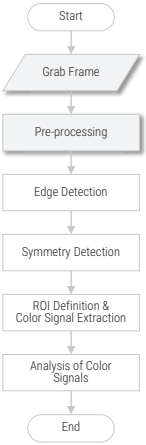


- Key feature:

1. Symmetrical object
2. Repeated color pattern



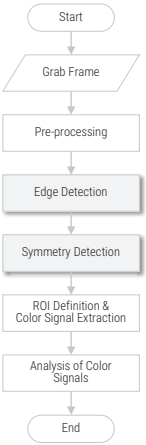
Insulator Detection
Algorithms



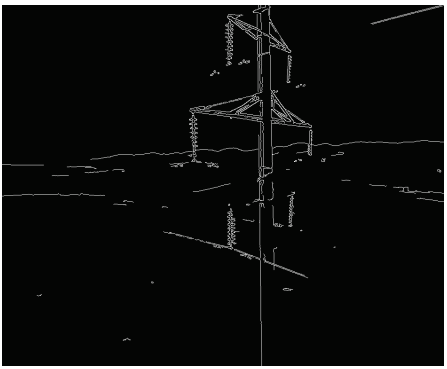
- Grab Frame
  - Load new frame
- Pre-processing
  - Resizing to fixed resolution
  - Noise filtering
  - Grayscale color conversion

TECHNISCHE UNIVERSITÄT CHEMNITZ
Fakultät für Informatik
Professur Technische Informatik

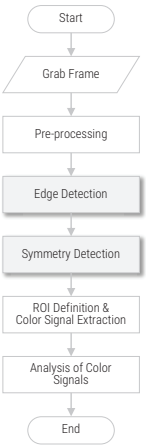
Insulator Detection
Algorithms



- Edge Detection
  - Detect edges
- Symmetry Detection
  - Detect symmetrical edges
  - Returns symmetry axis



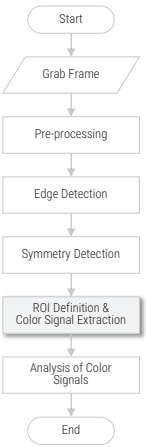
Insulator Detection
Algorithms



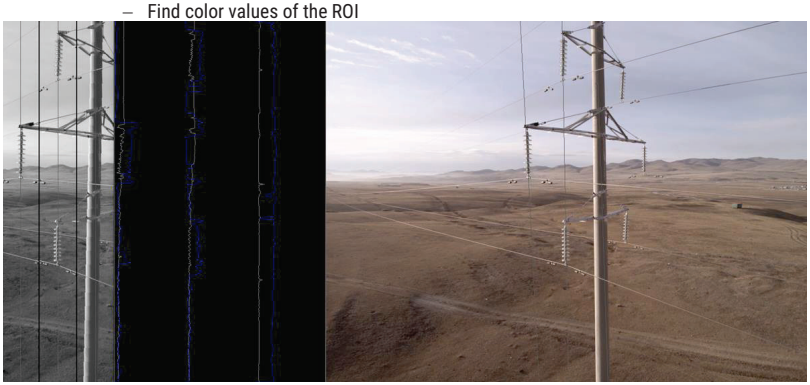
- Edge Detection
  - Detect edges
- Symmetry Detection
  - Detect symmetrical edges
  - Returns symmetry axis

TECHNISCHE UNIVERSITÄT CHEMNITZ
Fakultät für Informatik
Professur Technische Informatik

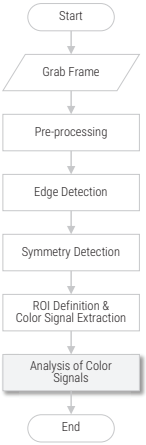
Insulator Detection
Algorithms



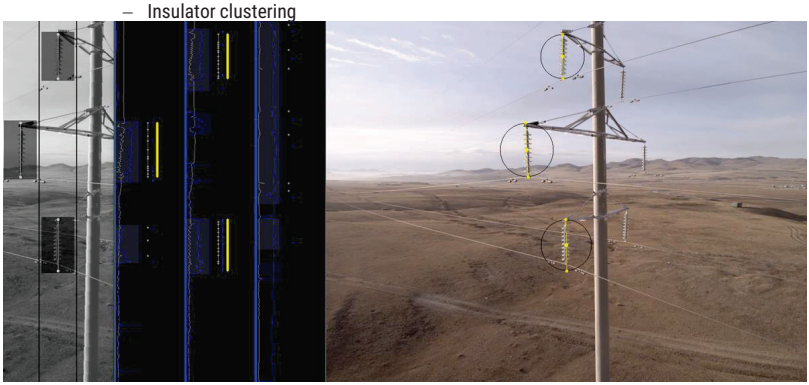
- ROI Definition & Color Signal Extraction
  - Defining the ROI based on symmetry axis
  - Find color values of the ROI



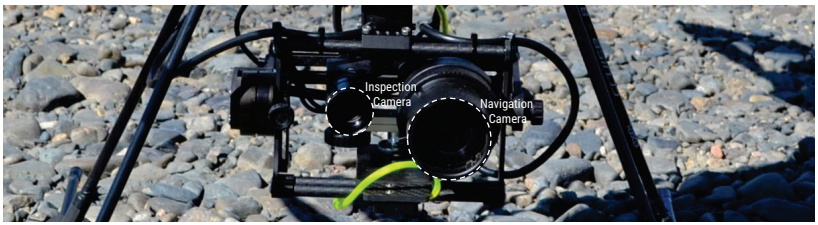
Insulator Detection
Algorithms



- Analysis of Color Signals
  - Repeated color pattern/fluctuation detection
  - Insulator clustering



Current Approach
Hardware Setup



AREIOM SETUP (for Autonomous flight)

1. Inspection camera
   - **12MP** resolution
   - **25mm** lens
     - *Horizontal Field of View* ***19.9°***
     - *Vertical Field of View* ***15°***
2. Navigation camera
   - **3.2MP** resolution
   - **14mm** lens
     - *Horizontal Field of View* ***33°***
     - *Vertical Field of View* ***24.8°***

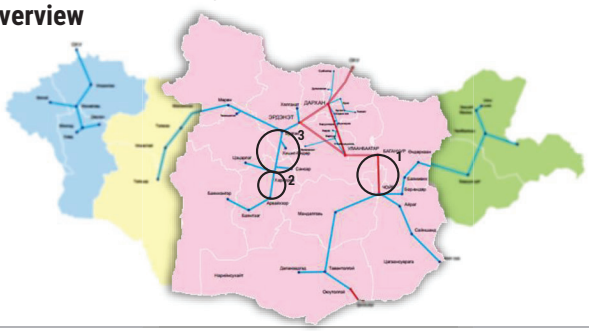
Current Approach

Positive detection rate ≈ 80%

- *Closest insulator set*
- *Distance 5 - 15m*

APOLI Test Flight 2019

Overview



№	Power line name	Voltage level /kV/	Power Line Length /km/	Total number of Power Poles	Circuit type	Spent days
1	207	220	178.00	561	Single	3
2	Arvaikheer	110	122.98	442	Single	1
3	Kharkhorin A/B	110	207.08	832	Double	3

APOLI Test Flight 2019
Statistics

- Inspected power poles: 122
- Covered area: 180.69 km
- Total flights: 134
- Total flight time: 11:27 h
- Flight days: 7

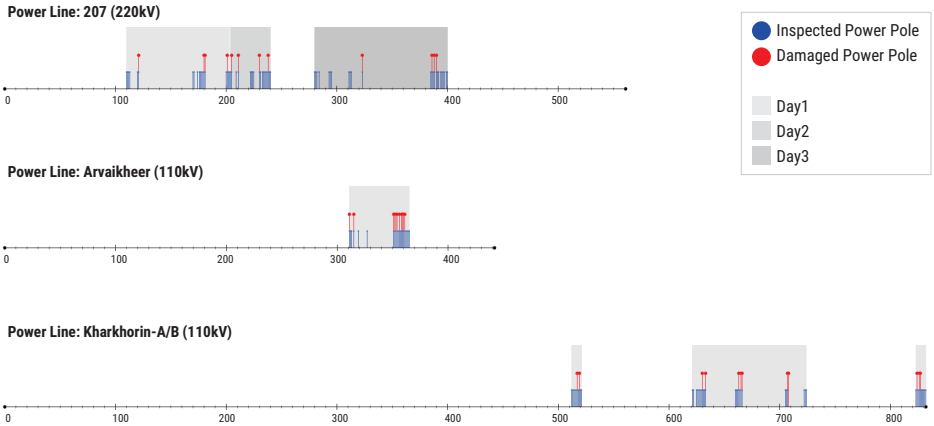
№	Power line name	Inspected Power Poles	Covered area /km/	Total flight number	Total flight time	Spent days
1	207	59	91.06	67	5:49	3
2	Arvaikheer	19	14.46	21	1:50	1
3	Kharkhorin A/B	44	75.16	46	3:48	3
	Total	**122**	**180.69**	**134**	**11:27**	**7**

APOLI Test Flight 2019
Results

- Inspected power poles: 122
- Power poles with damages: 32
- Detected burn marks: 51
- Total file size: 188 GB
- Average Flight Time: 6 min

№	Power line name	Inspected Power Poles	Power Poles with damage	Detected Burn Marks	Damaged Power Pole Numbers
1	207	59	11	15	121, 180, 181, 201, 205, 211, 230, 323, 386, 388, 400
2	Arvaikheer	19	11	26	311,315, 351, 352, 353, 354, 356, 358, 359, 360, 361
3	Kharkhorin A/B	44	10	10	517, 519,630,633,663,665, 666,707,708,824,826,827
	Total	**122**	**32**	**51**	

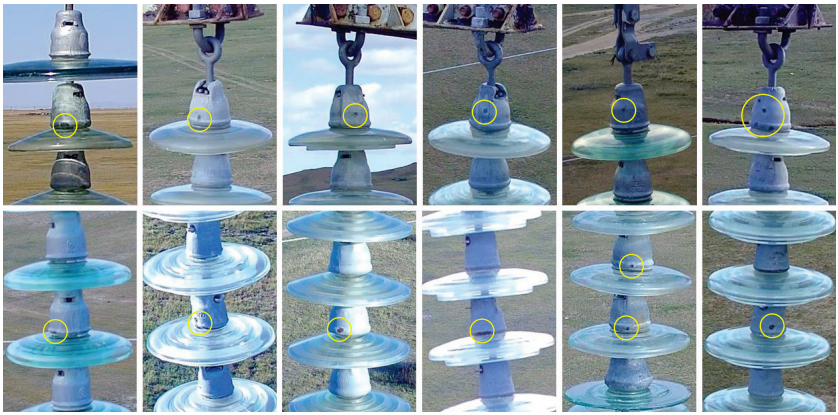
APOLI Test Flight 2019
Results



APOLI Test Flight 2019
Results

Burn marks at the bottom of the caps of the top insulators

APOLI Test Flight 2019
Results



Burn marks in random positions

APOLI Test Flights
2016, 2017, 2019

- Total flight number: 206
- Total inspected poles: 157
- Total flight duration: 22:04
- Total gathered data size: 275.3 GB

Year	Power Line Name	Flight Number	Inspected Poles	Spent Days	Flight Duration	File Size /GB/
2016	Kharkhorin A/B	23	15	2	05:59	-
2017	Kharkhorin A/B	51	20	4	04:38	87.3
2019	207, Arvaikheer, Kharkhorin A/B	132	122	7	11:27	188.0

Urbane “Letzte Meile” – Transport mittels Lieferdrohne

N. Gageik, *Geschäftsführer (tech.), Emqopter GmbH*, M. Bihl, *Geschäftsführer (kaufm.)*, *Emqopter GmbH*

***Abstract*— Der Transport auf der letzten Meile ist eines der Trendthemen im Bereich der Mobilität der Zukunft, insbesondere im urbanen Bereich. Gründe dafür sind zum einen, die steigende Nachfrage an Lieferungen im B2C-Bereich durch den wachsenden Onlinehandel, zum anderen die anteilig hohen Kosten für den Transport auf der letzten Meile in Relation zum gesamten Transportweg. Emqopter hat die erste vollautonom flugfähige Lieferdrohne entwickelt, die regelmäßig im urbanen offenen Luftraum in Deutschland betrieben wird. Der komplett redundant ausgelegte Oktokopter wird für den Transport von Industrie- sowie Medizingütern im Gewichtsbereich bis zu 2kg zwischen festen Punkten eingesetzt (Peer-to-Peer). Die Einsatzmöglichkeiten von unbemannten Flugsystemen (UAS, engl. „unmanned aerial system“) im offenen Luftraum wird in Deutschland derzeit durch die Vorgaben im Genehmigungsverfahren noch stark beschränkt. Dies wird sich aber im kommenden Jahr durch die bereits verabschiedete EU-Richtlinie zur europaweiten Standardisierung der Drohnen-Regulierung ändern, was zu einem signifikanten Anstieg der in der EU betriebenen unbemannten Flugsystemen führen kann.**

I. Einleitung

Im Dezember 2013 verkündete *Jeff Bezos*, Gründer und Geschäftsführer von *Amazon Inc*., im US-Fernsehen im Rahmen der Sendung *60 Minutes* auf *CBS*, dass Amazon plane Pakete in der Zukunft mit Hilfe von vollautonomen Drohnen ausliefern zu wollen – obwohl *Bezos* bewusst war, dass es in den USA zum damaligen Zeitpunkt keine regulatorische Grundlage gab ein solches Vorhaben zu implementieren. *Bezos* stellte in einem Bericht einen Oktokopter vor, der in einer Lagerhalle automatisiert mit einem Paket bestückt wurde, diese dann autonom verlässt, zu einem Kunden fliegt, dort vor der Haustür landet, das Paket selbstständig ablegt und anschließend wieder zurückfliegt. Einen operativen Lieferbetrieb mittels Drohne hielt er damals frühestens 2015 für realistisch. Mittlerweile ist *Amazon* das größte börsennotierte Unternehmen der Welt, hat schätzungsweise mehrere Milliarden US-$ in die Erforschung und Entwicklung von Lieferdrohnentechnologien investiert und bereits die vierte Generation UAS zum Transport vorgestellt. Dabei war *Amazon* eines der ersten Unternehmen, das sich aktiv mit der Thematik beschäftigte, ist aber bei weitem nicht mehr der Einzelfall. So unterhalten beispielsweise auch *Alphabet, Alibaba, DPDHL, JD.com* und viele weitere Großunternehmen Lieferdrohnenprojekte. Hierin spiegelt sich die Relevanz wider, die der Technologie insbesondere im Bereich „Letzte Meile“-Logistik zugesprochen wird. Die Entwicklung der „Drone Industry Map“, eine Übersicht der, auf Marktforschung im Bereich der Drohnenindustrie spezialisierten, Organisation *Drone Industry Insights,* die die Entwicklung und die aktiven Teilnehmer auf dem Drohnenmarkt grafisch darstellen soll, belegt dies. Im Vergleich zwischen den Jahren 2015 und 2019 zeigt sich eine starke Verdichtung des Marktes durch eine steigende Anzahl an Marktteilnehmern.

Abbildung 1: Entwicklung des Drohnenmarktes

Im Februar 2019 veröffentliche Marktdaten, erhoben im Auftrag des *Verband Unbemannte Luftfahrt,* zeigen diesen Trend auch in Deutschland. So stieg der Absatz von Drohnen in Deutschland von noch 170T Stück im Jahr 2015 auf 474T Stück im Jahr 2018 an. Demgegenüber stehen ein Marktvolumen in Höhe von Rund 300Mio. € im Jahr 2015 und 600Mio. € im Jahr 2018, wobei der private Drohnenmarkt bei Rund 100Mio. € Marktvolumen in dem Zeitraum nahezu konstant blieb. Das übrige Volumen ist jeweils kommerziell genutzten Drohnenlösungen zuzuschreiben.

07.11.2019

N. Gageik, Geschäftsführer (tech.), Emqopter GmbH, n.gageik@emqopter.de.

M. Bihl, Geschäftsführer (kaufm.), Emqopter GmbH, m.bihl@emqopter.de.

II. Einsatzmöglichkeiten für Lieferdrohnen

Dass die Bedeutung von Drohnentechnologien gegeben ist, legen die Marktdaten nahe. Noch größer ist aber das Potenzial der Technologie, insbesondere, wenn man die regulatorischen Hemmnisse in Relation setzt, denen UAS insbesondere im outdoor-Bereich heute noch ausgesetzt sind. So ist es bisher beispielsweise nur möglich, Drohnen, insbesondere auch Lieferdrohnen, auf bestimmten, von der jeweils zuständigen Landesluftfahrtbehörde erlaubten, vordefinierten Routen zu betreiben. Die Zeit, die verstreicht, bis eine solche Erlaubnis erteilt wird, variiert in Abhängigkeit der bisherigen Erfahrung der entsprechenden Landesluftfahrtbehörde mit der Thematik und ihrer Auslastung. Eine flexible und bedarfsgerechte Routenplanung in Echtzeit wird dadurch nahezu unmöglich. Des Weiteren treten Schwierigkeiten auf, wenn der Betrieb eines UAS grenzübergreifend in die Zuständigkeit mehrerer Landesluftfahrtbehörden fallen sollte. Der Einsatz einer Lieferdrohne ist in Deutschland heute daher nur auf Linienrouten auf der kurzen Distanz zwischen festen Punkten, zwischen denen regelmäßig eilige Transporte anfallen, ohne Weiteres möglich. Grundsätzlich käme hierbei auch der Betrieb im Rahmen eines Netzwerkes in Betracht, das mehrere Knotenpunkte als Start- und Landeplätze, entweder nach dem „hub and spoke“- oder „peer-to-peer“- Prinzip miteinander verbindet. Das kommerzielle Potenzial der Drohnen, das insbesondere auch Unternehmen wie *Amazon* in der bedarfsgerechten und flexiblen Lieferung auf der „Letzten Meile“, wie sie im B2C-Bereich typisch ist, sehen, bleibt dabei bislang ungenutzt. Um dieses Potenzial zu heben, hat die EU sich auf eine einheitliche Regulierung für UAS im kommerziellen Bereich verständigt. Mit der Einführung des sogenannten U-Space soll ein kollaborativer Luftraum geschaffen werden, den sich bemannte und unbemannte Flugsysteme gleichermaßen teilen. Mit Hilfe einer SIM-Karte lässt sich eine kollaborative Drohne im Mobilfunknetz europaweit lokalisieren. Führt man diese Daten mit den vorhandenen Radardaten der bemannten Luftfahrt zusammen, lässt sich der gemeinsame Luftraum koordinieren. Eine Remote-ID soll jedes UAS zusätzlich jederzeit eindeutig identifizierbar machen. Gepaart mit einheitlichen Regeln über notwendige Spezifikationen für die Genehmigung von spezifischen Missionen lassen sich dann auch Aufstiegserlaubnisse über ein Traffic Management System in Echtzeit erteilen. Die erste Phase der Implementierung des U-Space startet 2019 mit der Initialisierung der Grundlagen des Service. Bis 2035 sollen der Service im Rahmen von U-Space dann vollumfänglich den Dienst aufnehmen.

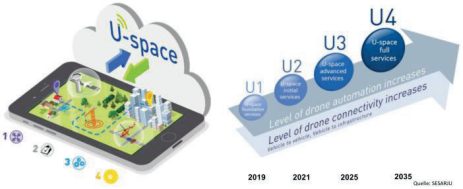


Abbildung 2: Kollaborativer Luftraum U-Space

III. Das Lieferdrohnenprojekt von Emqopter

Das Ausgangsszenario auf dem das Pilotprojekt „Lieferdrohne“ von Emqopter aufsetzt, ist prototypisch für große und mittlere Unternehmen weltweit. Der Kunde, ein Automobilzulieferer, betreibt an seinem Stammsitz zwei Werke zwischen denen regelmäßig Transporte von Kleinteilen mit Gewicht im niedrigen Kilogramm-Bereich anfallen. Die beiden Werke liegen innerhalb von 650m Luftlinie zueinander in einem Mischgebiet. Allerdings gibt es keine direkte Straßenverbindung zwischen den Werken, da kreuzende Bahngleise, eine Wohnanlage sowie Industrieanlagen anderer Unternehmen die beiden Unternehmenseinheiten trennen. Bislang werden die Transporte mit dem Auto über eine umwegige Straße durchgeführt. Von Werkhalle zu Werkhalle dauert der Prozess mit dem Auto je nach Verkehrslage zwischen 10 und 15 Minuten.

Abbildung 3: Die Lieferstrecke im Pilotprojekt

Im Sommer 2018 sind die Entwicklungsarbeiten für die Lieferdrohne erfolgreich abgeschlossen. In einem drei-monatigen Pilot-Betrieb wird das System auf Herz und Nieren unter Realbedingungen erprobt. Das Ergebnis: auf über 90 dokumentierten Flügen kam es zu keinem einzigen Zwischenfall, Eingreifen, Fall-Back oder Fail-Safe. Ein Flug dauerte dabei zwischen 3 und 5 Minuten. Im Schnitt konnte eine Zeitersparnis von 2/3 realisiert werden. Auf Basis der Betriebskosten bedeutet dies eine Kostenersparnis von 20% im Vergleich zum Transport mit dem Auto.

IV. Spezifikationen und Komponenten der Lieferdrohne

Bei dieser ersten Version handelt es sich um einen Oktokopter mit einer Spannweite von 140cm und einem Leergewicht von 9kg. Die mögliche Zuladung beträgt dabei bis zu 2kg. Die Flugzeit liegt bei ca. 15 Minuten. Bei einer Geschwindigkeit von ca. 50km/h hat das System so eine Reichweite von ca. 2-5km in Abhängigkeit der Umweltbedingungen.

Abbildung 4: Die Lieferdrohne von Emqopter

Eine Schlüsselkomponente des Systems ist das intelligente Landeplatzdetektionssystem. Dabei Handelt es sich um ein Sensormodul auf LiDAR-Basis. Insgesamt fünf Sensoren erfüllen die Funktionen der Hinderniserkennung, Kollisionsvermeidung sowie Landeplatzdetektion. Ein Sensor ist dabei fest mit Ausrichtung nach unten angebracht. Die übrigen vier Sensoren sind schwenkbar außen an den Auslegern des Rahmens angebracht und zeigen jeweils in eine der vier Himmelrichtungen.

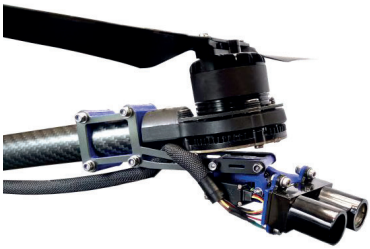


Abbildung 5: Schwenkbarer Sensor am Ausleger

Während des Fluges sind diese vier Sensoren horizontal ausgerichtet. Als Teil der umfangreichen Situationserfassung der Lieferdrohne sind sie elementar für die Funktionalität von „Detect & Avoid“. Im Landevorgang beginnen die Sensoren zu Schwenken und erfassen so die Umgebung unterhalt der Drohne. Dadurch ist es möglich unabhängig von externen Referenzsystemen, wie GPS, optischen Markern, Bodensensoren oder ähnlichem autonom zu Landen. Insbesondere im urbanen Bereich, wo es zwischen Gebäuden häufig zu GPS-Störungen kommt, ist das ein großer Vorteil. Darüber hinaus arbeitet das System „on the edge“, was zusätzliche Sicherheitsaspekte beinhaltet: zum einen findet keine Kommunikation zu einem externen System statt, die abbrechen oder angegriffen werden könnte, zum anderen werden so lange Latenzen vermieden. Da das intelligente Landeplatzdetektionssystem das erste und bislang auch einzige bekannte System ist, das das vermag sowie eine absolute technologische Neuheit, ist es von Emqopter auch zum Patent angemeldet worden.

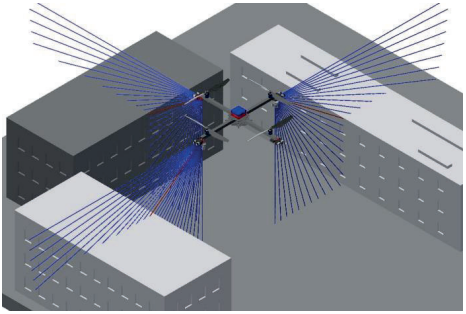


Abbildung 6: Skizze aus Patentantrag illustriert Funktionsweise

Neben dem intelligenten Landeplatzdetektionssystem fliegt an Bord der Lieferdrohne die EMQ Steuerung. Auf ihr laufen alle Autonomiefunktionen des Systems zusammen. Sie fungiert „quasi“ als „Companion Computer“ zu einer Pixhawk 2.1 Flugsteuerung. Diese voll-redundante Flugsteuerung ist das am weitesten verbreitetste und erprobteste System und bietet größtmögliche Sicherheit und Präzision. Die Kommandos der Flugsteuerung werden in bürstenlosen Elektromotoren umgesetzt. Diese arbeiten verschleißfrei und sind dadurch besonders zuverlässig und langlebig. Sie erfüllen außerdem die gängigen Industrienormen hinsichtlich Wasserdichte, Staubdichte sowie Korrosionsschutz. Seine Befehle erhält das System vom Betreiber über eine 2-fache voll-redundante Fernsteuerung oder die ebenfalls 2-fach voll-redundante Telemetrie-Telecommand-Einheit. In Kombination mit einem eigens für die Lieferdrohne entwickelten User-Interface können so alle Flugdaten des Systems während des Betriebs erfasst, ausgewertet und gespeichert werden. Zusätzlich ermöglichts das Interface eine intuitive und flexible Missionsplanung. Auf einer GEO-referenzierten Karte der Umgebung können individuelle Wegpunkte einfach per Mausklick eingegeben und bei Bedarf entsprechend der Anforderungen jederzeit angepasst werden.

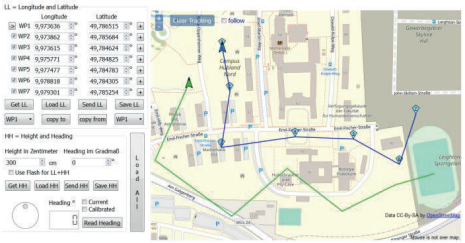


Abbildung 7: GEO-referenzierte Missionsplanung

Betrieben wird das System vollkommen elektrisch. Die dafür benötigte Energie stellen zwei 5800 mAh Akkumulatoren bereit, die mit Hilfe eines Schlittensystems Plug-n-Play mit nur einem Handgriff einfach und sicher angeschlossen werden. So kann nach dem Flug der Energiespeicher in wenigen Sekunden ausgetauscht werden. Mit sechs bis acht

Schlitten ist es dadurch, je nach Anwendungsszenario, möglich die Lieferdrohne 24/7 zu betreiben.
Um die Lieferdrohne auch auf noch anspruchsvolleren Missionen, als dem Pilotszenario betrieben zu können, gibt es sie auch in einer Premiumausführung mit Sonderkonfigurationen. Insbesondere für den Überflug von Menschenansammlungen verlangen Landesluftfahrtbehörden für die Betriebserlaubnis in der Regel ein Rettungssystem an Bord der Drohne. Hierfür ist auf der Lieferdrohne Premium ein Fallschirm-Notfallsystem implementiert. Im Worst-Case-Szenario, dem Absturz, öffnet sich in letzter Instanz der Fallschirm und bringt das System mit geringstmöglicher Aufprallenergie zu Boden. Ein akustisches Signal minimiert dabei weitere Gefahrenpotenziale und Verletzungsrisiken für Passanten am Boden. Eine weitere zusätzliche Sicherheitskomponente ist ein ADS-B Peilsender zur Helikopterortung. Großes Potenzial der Lieferdrohnentechnologie liegt im Transport eiliger Medizingüter, wie Blutkonserven seltener Blutgruppen, Proben oder Transplantate, im Krankenhausumfeld. Hier entscheiden bereits Zeitersparnisse im Minutenbereich über die Erfolgsaussichten einer Behandlung. Voraussetzung für den Einsatz von Drohnen in diesem Bereich ist allerdings, dass keine Gefährdung von bestehenden Einsatzsystemen, wie eben Rettungshubschraubern, entsteht. Im Sinne eine kollaborativen Luftraums sind dafür Sichtbarkeit und Koordination entscheidend. Durch den Einsatz des Peilsenders ist die Lieferdrohne für die anderen Teilnehmer im Luftraum keine unbekannte Variable, sondern eine berechenbare Konstante.

V. Ausblick und weitere Entwicklung

Auf dem Transport von Gütern und ganz allgemein in der Logistik entsteht in der Regel keine zusätzliche Wertschöpfung. Daher wird versucht die bestehenden Prozesse weitestgehend zu automatisieren. Die Lieferdrohne bildet dieses Bestreben ab, birgt aber selbst, in ihrem jungen Entwicklungsstadium, noch weiteres Automatisierungspotenzial. In der neusten Ausführung der Lieferdrohne von Emqopter wird die Transportbox durch einen Greifarm, der dem menschlichen Arm nachempfunden ist und Bewegungen in sechs Freiheitsgraden zulässt, ersetzt.

Abbildung 8: Lieferdrohne mit Greifarm

Mit Hilfe des Greifarms wird, zusätzlich zum Transport, auch das Aufnehmen und Ablegen eines Gutes automatisiert. Des Weiteren kann die Drohne so auch mit ihrer Umwelt interagieren, beziehungsweise diese manipulieren, zum Beispiel um Hebel, Schalter oder Werkzeug bedienen zu können. Das Beispiel zeigt, Drohnen sind Multifunktionswerkzeuge zur Luft, deren Einsatzmöglichkeiten nahezu keine Grenzen kennen. Gelingt die Integration von unbemannten Flugsystemen in bestehende Abläufe und Infrastruktur, sowohl gesellschaftlich, regulatorisch und betriebswirtschaftlich, kann die Technologien in den kommenden Jahren ihr disruptives Potenzial entfalten und so global zu einem nachhaltigen Mobilitätswandel führen.

Mit High-Tech gegen den Fachkräftemangel: Wie ein aufstrebendes Start Up aus Bayern mit Drohnentechnik frischen Wind in die Ausbildung von Fachkräften bringt

N. Gageik, *Geschäftsführer (tech.), Emqopter GmbH*, P. Benz, *Prozessmanager*, *Emqopter GmbH*

***Abstract*— Der Fachkräftemangel ist seit einigen Jahren ein heißes Thema im Bundestag. Immer mehr sind wir darauf angewiesen, Experten und ihr Wissen aus dem Ausland zu importieren. Immer weniger sind wir in der Lage, den Bedarf unserer Wirtschaft an gut ausgebildeten Frauen und Männern durch die eigene Ausbildung zu decken - und das als eine der führenden Industrienationen der Welt! Besonders im Bereich IT fehlt es an marktreifer Expertise. Der Kern des Problems liegt nach Ansicht von Dr. Nils Gageik, Geschäftsführer von Emqopter, im praxisfernen schulischen und akademischen Unterricht. An zahlreichen Beispielen kann man sehen, dass die Komplexität der Herausforderungen, welche in der Realität nach Schule und Universität auf uns warten, auf ein mundgerechtes Minimum herunter gebrochen wird, um die Schüler und Studenten nicht zu überfordern. Schließlich sind gute Noten ein Muss für einen guten Job und die sind eben nur zu holen, wenn der Lernstoff angenehm portioniert ist. Doch dass es eben auch anders geht sieht man an der Entwicklung des Start-Ups Emqopter. Bereits vor knapp 10 Jahren wurde mit der Gründung des Lehrstuhls für Luft- und Raumfahrtinformatik an der Universität in Würzburg der Grundstein für das heute 11 Mitarbeiter beschäftigende Unternehmen gelegt. Hauptgeschäftsfeld von Emqopter ist die Entwicklung von intelligenten Drohnen. Die quasi unbegrenzten Einsatzmöglichkeiten von Flugrobotern sind für die Beschäftigten die größte Motivation. Und genau hier setzt Emqopter den Hebel für eine bessere Ausbildung an: Mit dem Quadrotor Control System, kurz QCS, bietet das Start-Up ein umfassendes Lehrsystem für Schulen und Universitäten an, das durch Innovation begeistert und die Lernenden motiviert, über die Grenzen des Unterrichtstoffes zu blicken. Damit, so Gageik, schafft man es, die Ausbildung praxisnah zu gestalten und trotz der vollen Komplexität des Sachverhaltes ein tiefes Verständnis des Lerninhaltes zu erreichen, welches durch die kontinuierliche und konsequente Abstrahierung bei herkömmlichen Methoden verloren geht. Ein spannendes Konzept mit Zukunftsperspektive!**

07.11.2019

N. Gageik, Geschäftsführer (tech.), Emqopter GmbH, n.gageik@emqopter.de.

P. Benz, Prozessmanager, Emqopter GmbH, m.bihl@emqopter.de.

Abbildung 1: Dr. Nils Gageik am QCS im Austausch mit den Fachkräften der Zukunft.

I. Einleitung

Als im Oktober 2010 der Studiengang Luft- und Raumfahrtinformatik ins Leben gerufen wurde, war dem damals ersten wissenschaftlichen Mitarbeiter des neuen Lehrstuhls in Würzburg Nils Gageik von vorneherein klar, dass die komplexe Ausbildung von jungen Studenten in einem High-Tech Bereich wie es die Luft- und Raumfahrt ist, mehr braucht als die herkömmlichen Methoden. Allein durch Vorlesungen und die Bearbeitung von theoretischen Übungsaufgaben würde es für die Lernenden extrem schwer werden, die Materie entsprechend aufzuarbeiten, um das Studium erfolgreich abzuschließen. Es brauchte Praxis. Es brauchte ein Lehrwerkzeug, an dem die Studierenden all die Theorie der Vorlesungen und Übungsblätter hautnah einsetzen konnten. Denn Theorie gibt es in diesem Studium genug! Angefangen bei einfacher Lehre über Algorithmen und Datenstrukturen, über Luft- und Raumfahrtdynamik mit Orbitkalkulationen und Raketengleichungen, bis hin zur Robotik mit all ihren Integralen und Differenzialen. So entstand das Lehrkonzept des Quadrotor Control Systems.

Hauptbestandteil des QCS ist ein Quadrokopter, also ein Multikopter mit vier Rotoren. Dieser ist auf einer stabilen Aufhängung gegen das Abheben gesichert, sodass man direkt am Arbeitsplatz eine Drohne erhält, die im Rahmen des Unterrichts programmiert wird.

Abbildung 2: Das QCS auf dem 3DOF-Gelenk

Programmiert wird nicht durch simples Blockbild zusammenstecken analog zum LEGO-Roboter sondern in einer professionellen Entwicklungsumgebung in C. Dadurch erhalten die Studierenden – größtenteils zum ersten Mal in ihrer gesamten schulischen und studentischen Laufbahn – einen Einblick in die reale Welt von Softwareingenieuren. Natürlich geht das mit einem gewissen Maß an Komplexität einher, doch damit ist den Lernenden das komplette Spektrum an Möglichkeiten geboten, welches es mit sich bringt, an realen Objekten zu entwickeln. Und die Erfahrung zeigt auch, dass die Lernenden keineswegs überfordert sind, sondern anscheinend ihr ganzes Leben darauf gewartet haben, endlich auf diese eine Herausforderung zu stoßen, um ihr Können unter Beweis zu stellen. Die Veranstaltung, in der das QCS an der Universität in Würzburg seitdem eingesetzt wird nennt sich Luft- und Raumfahrtlabor. Andere nennen das immer noch Unterricht. Doch mit dem was wir noch aus der Schule und unserem Studium kennen hat das nur noch wenig zu tun. In Kleinstgruppen wird den Studierenden zu Beginn der Veranstaltung die Aufgabe gestellt, die Drohne an ihrem Arbeitsplatz zum Fliegen zu bringen. Bis die Drohne tatsächlich fliegt vergeht dann ein ganzes Semester. Denn schließlich gibt es eben keinen einfachen Funktionsblock mit dem schönen Namen „Flieg“. Stattdessen dreht sich das QCS-Lehrkonzept um alles was dazu gehört, um die Drohne zum Fliegen zu bringen.

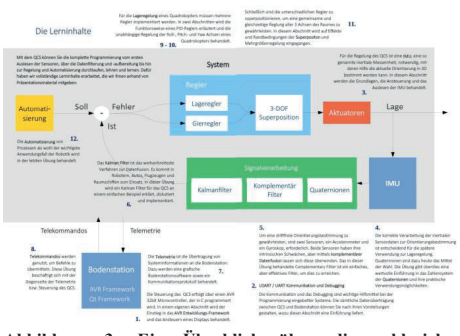


Abbildung 3: Ein Überblick über die zahlreichen adressierten Themen im QCS-Lehrkonzept.

II. Das QCS und das Lehrkonzept

Der Hauptprozessor des QCS ist ein Mikrokontroller. Auf diesem läuft das von den Lernenden zu entwickelnde Programm. Dieses verarbeitet die Lageinformationen der Drohne, die von einer *Inertial Measurement Unit* (IMU) gemessen werden und errechnet daraus Stellwerte für die einzelnen Motoren, um die Drohne in gewünschte Lage und später auch Position zu bringen. Zur Signalverarbeitung gehört einiges an digitaler Filtertechnik, um geeignete Messwerte zu extrahieren, die dann in einen dreidimensionalen *PID-Regler* gespeist werden. All diese Funktionen werden im Laufe des Semesters von den Studierenden händisch implementiert. Dies erfordert einiges an Fleiß und algorithmischem Denken, doch der Lohn einer selbstprogrammierten Drohne, der am Ende der Veranstaltung winkt, motiviert die Lernenden bereits hier zu unbezahlten Überstunden und eben auch dazu, sich viel spannendes Wissen anzueignen, was im späteren Arbeitsleben tatsächlich auch angewendet werden kann. Der Erfolg des Lehrkonzepts lässt sich leicht im direkten Vergleich zu anderen Lehrveranstaltungen ablesen (vgl. Abbildung 4).

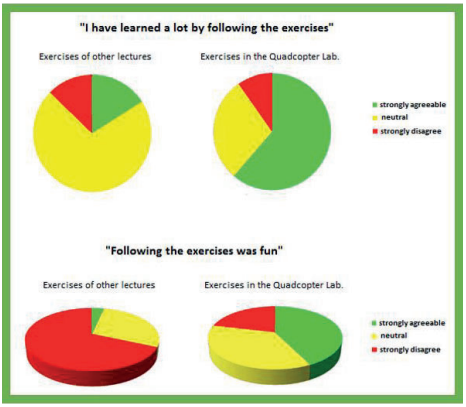


Abbildung 4: Eine im Rahmen der SPIE - Conference in San Francisco 2015 veröffentliche Statistik zur Wirkung des Einsatzes des QCS auf Lernende.

Das QCS begeistert vor allem durch seine Praxistauglichkeit. Anstatt von Beginn an den Anwender mit der vollen Komplexität und Vielschichtigkeit des Fluglagereglers einer Drohne zu konfrontieren, wird die Regelung in möglichst unabhängige Teilprobleme zerlegt. Dies geschieht durch den Einsatz der DOF-Gelenke. Mit ihnen ist es möglich, die Bewegungsfreiheit des auf dem Stativ befestigten Quadrokopters so einzuschränken, dass jeweils nur ausgewählte Freiheitsgrade (engl.: Degrees of Freedom) in der Bewegung aktiviert sind. So ist es möglich, die Parametrierung der einzelnen Achsen Gieren, Rollen und Nicken getrennt voneinander zu betrachten. Schließlich werden nach und nach die einzelnen Achsen aktiviert, bis eine Regelung um alle drei Rotationsachsen möglich ist. Ist die Regelung auf dem 3-DOF Gelenk stabil, ist der selbst programmierte Fluglaregler bereit für den ersten freien Flug. Das Gelenk wird mit einfachen Handgriffen durch die Akkuhalterung ersetzt, die Fernsteuerung angeschlossen und der Erfolg des Einsatzes des vergangenen Semesters gefeiert.

Abbildung 5: Schüler und Lehrer der Berufsbildenden Schule Burgdorf, die nach erfolgreichem Unterricht mit dem QCS die programmierte Flugsteuerung an der Drohne testen.

Wem der Weg durch die Entwicklung der einzelnen Teile der Flugsteuerung zu weit ist, dem bietet das QCS Lehrkonzept eine umfassende Bibliothek an Funktionen, die anstelle der Eigenimplementierung eingebunden werden können. Dabei ist es dem Dozenten komplett freigestellt, an welcher Stelle er die bereitgestellten Funktionen nutzen, und was er seinen Schülern und Studenten als Aufgabe stellen will. Damit kann zum Beispiel die gesamte Signalverarbeitung aus der Bibliothek übernommen werden, während die Parametrierung des Fluglagereglers die Hauptaufgabe wird. Eben ganz nach dem Anspruch der Veranstaltung. Und sogar das Programmieren kann dem Anwender abgenommen werden. Hierfür wurde das EMQ-Play Interface entwickelt, mit welchem der Benutzer in der Lage ist, sämtliche relevanten Parameter eines dreidimensionalen Fluglagereglers durch Schieberegler einzustellen und in Echtzeit das Verhalten des Quadrokopters am Stativ zu analysieren (vgl. Abbildung 6). Damit gelingt der Einstieg in die Drohnenentwicklung sogar ohne Programmiererfahrung – ohne dass der Weg und der Einblick in die Software der verwehrt wird!

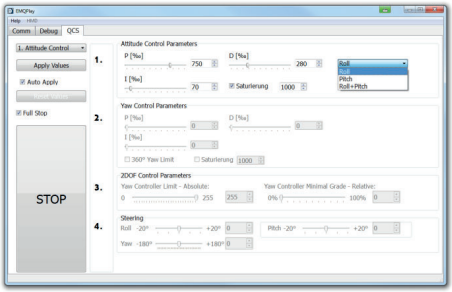


Abbildung 6: Das von Emqopter entwickelte EMQPlay, mit dem per Schieberegler die Regelparameter einer Drohnenflugsteuerung optimiert werden können.

Abbildung 7: Professor Montenegro an der Universität Würzburg forscht mit dem QCS stetig weiter an neuen Möglichkeiten Drohnen einzusetzen.

III. Tiefergehende Arbeiten mit dem QCS

Wo für die Lernenden das unterrichtsbasierte QCS-Lehrkonzept endet, öffnet sich der Freiraum für Projektarbeiten mit dem QCS. Gerade weil die Einsatzmöglichkeiten von Drohnen schier unendlich sind, ergeben sich durch sie auch immer wieder neue Forschungsfelder. Mit dem QCS in der Flugversion (QCS-F), für die zahlreiche Erweiterungen zur Verfügung stehen, ist es dem Anwender möglich, komplette Autonomie-Funktionen entweder selbst zu implementieren oder wiederum auf Basis von bereit gestellten Bibliotheksfunktionen die Autonomie zu nutzen, um komplexere Missionen mit dem QCS-F umzusetzen. So lassen sich beispielsweise durch den Einsatz von GPS Modulen eigene Wegpunkt- oder Point-Of-Interest-Manöver implementieren. Hinderniserkennungs-Module erlauben die Vermeidung von Kollisionen mit Objekten in der Umgebung. Alleine für die Kollisionsvermeidung bietet Emqopter ein Duzend Module unterschiedlichster Technologien an, um entsprechend zur jeweiligen Anwendung das optimale Equipment zu erhalten.

Abbildung 8: Eine Auswahl aus der Familie der Kollisionsvermeidungsmodule, die Emqopter passend zum QCS-F anbietet.

Durch Video-Übertragungssysteme kann die Drohne in Echtzeit im FPV-Modus navigiert werden, ohne in direkter Sichtlinie zu fliegen. Auch die Einbindung von externen Modulen ist durch zahlreiche Schnittstellen wie SPI, TWI und USART leicht möglich, sodass der Entwicklung von eigenen Drohnenlösungen nichts mehr im Weg steht. Sogar ein Entwicklungsframework zur Forschung an Drohnen mit Greifarm – von Emqopter „Quanipulator“ genannt – steht zur Verfügung!

Abbildung 9: Der Quanipulator mit integriertem LIDAR-Kollisionsvermeidungssystem.

Durch den Einsatz eines starken i7 Prozessors können auch anspruchsvolle Verfahren wie eine Objekt- oder auch Mustererkennung an Bord der Drohne implementiert werden. Gerade hier bieten sich Anwendungsfelder, welche nicht nur den Arbeiter entlasten, sondern überdies die Welt sicherer gestalten. Immer wieder klagen beispielsweise Bauern über Rehkitze die sich bei der Annäherung einer Erntemaschine im Feld verstecken, anstatt die Flucht zu ergreifen, was natürlich eine tödliche Gefahr für die jungen Tiere darstellt und darüber hinaus die Qualität der Ernte mindert. Durch den Einsatz von Drohnen können Rehkitze, die sich im Gras verstecken aus der Vogelperspektive leicht erkannt werden. Die Automatisierung der Rehkitzerkennung stellt dann einen noch größeren Schritt in der Unterstützung des Feldarbeiters dar.

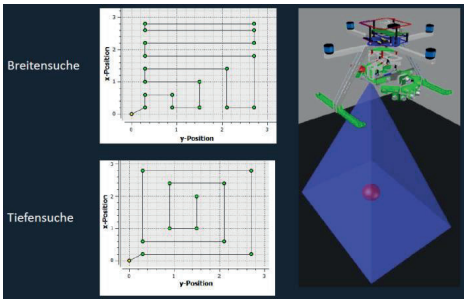


Abbildung 10: Algorithmen zur Objekterkennung und Suche mit einer Drohne

Ein weiteres zukunftsträchtiges Einsatzgebiet von Drohnen stellt der autonome Transport von Kleinteilen in urbanem Gebiet dar. Seit Jahren kämpfen Anbieter wie Amazon und DHL um Reputation in diesem Bereich. Dass Emqopter bereits seit mehr als 2 Jahren ebenfalls als respektabler Kontrahent in diesem Rennen mitmischt, zeigt wiederum die Expertise, die das junge Unternehmen im Bereich Drohnentechnik mitbringt. Und nun stellen Dr. Gageik und seine Mitarbeiter ihr Knowhow in diesem Innovationsfeld in einem spannenden Lehrkonzept unseren Schülern und Studenten zur Verfügung. Was braucht das Forscherherz mehr, um sich in der jungen Welt der Flugrobotertechnik einzufinden, wohl zu fühlen und durchzustarten?

IV. In die Zukunft mit dem passenden Equipment

Da wäre noch ein flexibles Development Board, welches nicht nur zur Programmierung von Drohnen geeignet ist, sondern alle gebräuchlichen Interfaces unterstützt, um eine Vielzahl beliebiger Entwicklungsthemen abzudecken. Zwei starke Prozessoren, vier serielle Kommunikationsschnittstellen, zahlreiche integrierte Spannungsversorgungen, SPI, TWI, 10 programmierbare LEDs, Potenziometer, Buttons und Joysticks und natürlich ein 80-Zeichen-Display zur Darstellung von Debug-Daten. Das EMQ3000 ist Teil des QCS-Entwicklungsframeworks, kann aber auch unabhängig erworben werden. Damit ist es jedem Entwickler vollkommen offen, seine eigene Drohne zu programmieren, sollte ihm das QCS mit seinen umfassenden Materialien und Lernhilfen zu viel erscheinen. So oder so kann man gespannt sein auf das, was die Generationen nach uns während ihrer Aus- und Weiterbildung mitnehmen, um die Herausforderungen in der Berufswelt der Zukunft zu bewältigen. Fest steht: Dass es an adäquatem Unterrichtswerkzeug gefehlt hat, kann Angesichts der Möglichkeiten des QCS zum Einsatz als Lehr- und Entwicklungstool keine passende Ausrede mehr sein!

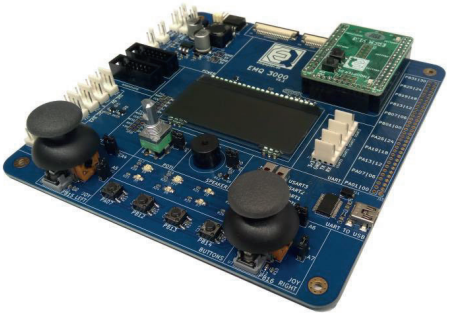


Abbildung 11: Das EMQ3000 Entwicklungsboard von Emqopter ist Teil des QCS-Lehrkonzepts, aber auch unabhängig für die Embedded-Entwicklung konzipiert und erhältlich.

Geschäftsmodellentwicklung für Start-ups und Finanzplanung

IBS Workshop Micro Air Vehicle Technologie 2019

Jun.-Prof. Dr. Mario Geißler (mario.geissler@wirtschaft.tu-chemnitz.de)
Technische Universität Chemnitz | Dittesstraße 15 | 09126 Chemnitz

Sparkasse Chemnitz

Was ist eine Innovation

Kreativität, Invention, Innovation

Schöpferische, gestalterische **Fähigkeit**, neue Ideen hervorzubringen

Neue Entwicklungen, Umsetzung einer neuen Idee

Erfolgreiche Markteinführung einer neuen Entwicklung

Innovationen und Start-ups

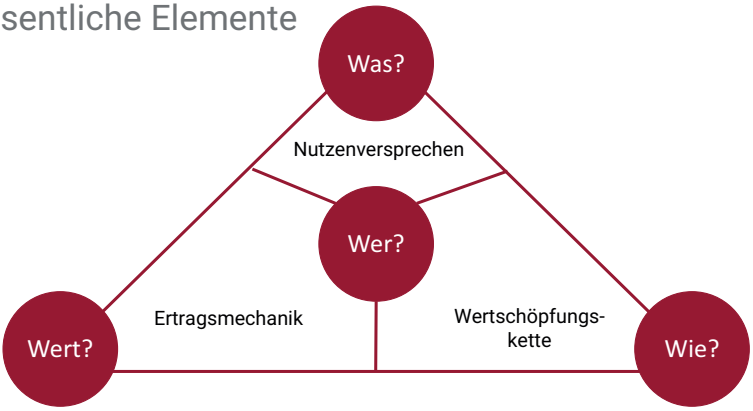
Technologie – Geschäftsmodell - Kundenbedürfnis

- Innovation ist keine Erfindung und keine reine Technologie
- Erfindungen müssen auf dem Markt funktionieren
- Erfolgreiche Innovationen brauchen ein funktionierendes Geschäftsmodell

Fakultät für Wirtschaftswissenschaften
Juniorprofessur Entrepreneurship in Gründung und Nachfolge, Stiftungsprofessur der Sparkasse Chemnitz

Geschäftsmodell

Drei wesentliche Elemente



Gassmann, O.; Frankenberger, K.; Csik, M. Geschäftsmodelle entweickeln – 55 innovative Konzepte mit dem St. Galler Business Model Navigator; Carl-Hanser Verlag, München, 2006, S. 16.

Einen guten Austausch und viel Erfolg!

Jun.-Prof. Dr. Mario Geißler

Telefon: +49 (0) 371 - 531 36685
Fax: +49 (0) 371 - 531 8 36685
Email: mario.geissler@wirtschaft.tu-chemnitz.de
Web: www.tu-chemnitz.de/wirtschaft/egn
www.startupassistent.de
www.startupoekosystemeindeutschland.de
Twitter: @mario_geissler

Sparkasse Chemnitz

IBS Workshop Micro Air Vehicle Technologie – 07.11.2019

Geschäftsmodellentwicklung für Start-Ups und Finanzplanung

Referent: Jun.-Prof. Dr. Mario Geißler & Daniel Gränitz

Agenda

1 Warum Finanzplanung?

2 Nutzen einer sorgfältigen Finanzplanung

3 Anbieter einer Finanzplanung

1 Warum Finanzplanung?

Praxisbeispiele:

Die geschäftsführenden Gesellschafter G1 und G2 gründen die Air Vehicle Start-up GmbH. Gesellschaftsanteil G1 50% und G2 50%. Der G1 hat im Jahr der Unternehmensgründung geheiratet (ohne Ehevertrag) und wurde im selben Jahr Vater eines Sohnes. G2 ist ledig. Weder G1 noch G2 verfügen über ein Testament. Im Gesellschaftsvertrag ist ebenfalls nichts zur Nachfolge geregelt.

Nachdem sich das Unternehmen in den ersten drei Jahren sehr gut entwickelt und der wirtschaftliche Erfolg eingestellt hat, stürzt der neuste Drohnen-Prototyp bei einem Testflug ab und erschlägt den G1.

Frage: Wie wirkt sich der Fall auf die Air Vehicle Start-up GmbH aus?

1 Warum Finanzplanung?

Praxisbeispiel – Prototyp stürzt bei Testflug ab und erschlägt G1:

Frage: Wie wirken sich der Fall auf die Air Vehicle Start-up GmbH aus?

1 Warum Finanzplanung?

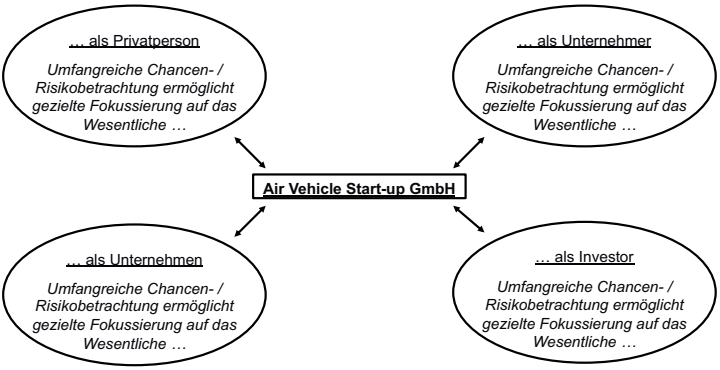
Finanzplanung ist …

- **ganzheitliche Beratungsdienstleistung**
- ***als Planungsprozess organisiert*** *(Datenaufnahme, Analyse & Planung, Realisierung & periodische Kontrolle)*

Finanzplanung soll Privatpersonen / Familien / Unternehmern …

- **helfen finanzielle Ziele zu erreichen**
- **sich auf das Wesentliche konzentrieren zu können**
- ***unter Berücksichtigung rechtliche, persönliche, familiäre, externe Rahmenbedingungen***

2 Nutzen einer sorgfältigen Finanzplanung



Agenda

1 Warum Finanzplanung?

2 Nutzen einer sorgfältigen Finanzplanung

3 Anbieter einer Finanzplanung

3 Mögliche Anbieter einer Finanzplanung

private Finanzplanung:

adcius VERMÖGENSTREUHAND + Steuerberater
Rechtsanwälte
Notare

Finanzplanung im Unternehmen:

adcius VERMÖGENSTREUHAND + Steuerberater
Wirtschaftsprüfer
Rechtsanwälte
Notare

Daniel Gränitz

32 Jahre, verheiratet, 1 Sohn

Master in Wealth Management
Dipl. Betriebswirt (BA) – Bank
Certified Financial Planner (CFP®)

Kontaktdaten:

adcius Vermögenstreuhand GmbH
Parkstraße 28 | 09120 Chemnitz

t 0371 – 66 612 932
f 0371 – 66 612 929
graenitz@adcius.de
www.adcius.de

Experience Report on Reuse in Avionics

Marina Reich* ‡, Andreas Schweiger*, Julian Lorenz*, Ulrich Margull †
*Airbus Defence and Space GmbH, †Technische Hochschule Ingolstadt, ‡Technische Universität Chemnitz

***Abstract*—Reuse in avionics software has always been a challenge and unfortunately there are many popular examples where it went wrong. In this paper we examine reuse from two different perspectives. First, we focus on the identification of potentially reusable components in existing projects. Second, we describe the reuse of a component in a new project.**

In addition to fulfillment of requirements for airworthiness certification, industrial processes and artifacts are so complex that a reusable part of previously developed software is hard to identify in an existing project. We propose an automated approach for the identification of potentially reusable components that ensures the traceability to their data items. Such components shall have the least dependency on their environment.

In our experiment targeting the reuse in a new project, we reuse the Flight Control Software ArduPilot for piloted drones for the development of autonomous drones in GPS denied areas or urban scenarios. In the course of this development the reuse considerations are observed. We present in detail all reuse considerations faced during this project and conclude with a list of open challenges that can be dealt with in future research projects.

An additional contribution is the implementation of the Distributed Localization System (DLS) that enables the drone navigation by Simultaneous Localization And Mapping (SLAM) in combination with inertial measurement sensors.

***Index Terms*—Avionics software, software reuse, flight control software, simultaneous localization and mapping**

I. Introduction

Reuse in avionics is seldom. The reasons for this are located first and foremost in software being usually customized for specific purposes, the complexity of the systems under development, and the development processes themselves.

The objective is to understand the challenges that are usually encountered during reuse. For this we examine the development process from two different perspectives. First, we point our attention to the industrial development processes and artifacts for the identification of procedural needs. In particular, we propose an approach of automated component identification. Second, we reuse an open source software component, identify the additional aspects to be considered, and derive lessons learned. With the described analysis and the demonstrator we contribute to the progress of application of reuse in industry.

The paper is organized as follows: Section II introduces the context of both approaches and raises the research questions. Section III presents the approach for component identification. Section IV reports on the reuse of a selected Flight Control Software (FCS). Section V concludes with an overview of reuse considerations and gives an outline of future work.

II. The Scope of Software Reuse

In [1] software reuse is defined as *"the systematic use of existing software assets to construct new or modified ones or products. Software assets in this view may be source code or executables, design templates, free standing Commercial-Off-The-Shelf (COTS) or Open Source Software (OSS) components, or entire software architectures and their components forming a product line or product family."*. This delivers the components that could be reused and pays attention to the systematic use. However, the definition lacks the consideration of requirements, tests, and integration during reuse. This brings us to the following research questions:

- **RQ1:** Which properties can be assigned to a systematic and automated approach for reuse?
- **RQ2:** Which properties can be assigned to a reusable asset?

In order to answer these questions, we first describe the constraints that are set by the requirements on the industrial reuse process in avionics by reviewing (1) different development approaches, (2) certification guidelines that are mandatory in the domain of avionics, and (3) the degree of reusability.

A. Development Approaches

[2] provides an overview of potential reuse artifacts for different development approaches. The authors differentiate between six categories: (1) Ad hoc/heritage/legacy, (2) Component, (3) Model based, (4) Product line, (5) Ontology, and (6) COTS. [2] enumerates, e.g. for the first approach, these reusable artifacts: code fragments, routines, requirements, use cases, architectures, models, drawings, test products, and code. This shows, that in addition to the definition in [1], the aspects testing and requirements are considered as potential reuse artifacts, as well. By bringing together the definitions from [2] and [1], we get a comprehensive definition of reuse and which artifacts are necessarily involved.

[3] analysed the reuse process in the automotive industry. This work emphasises modularization of software components. For reusable components, the authors require the reusable components to be hardware independent, to have interfaces that exchange data locally on an electronic control unit (ECU) or via data bus, and to have minimized code size and execution time. The proposed process is designed to develop reusable components (called core assets) that are then combined to a product considering product requirements. Different to this solution that requires structured development upfront, we pay attention to ad hoc development approaches, where legacy artifacts are supposed to be reused. This brings up the challenge of identification of a reusable component as it was not developed for reuse upfront. This is often the case, since in avionics the product life cycles are very long and it might turn out, that a product developed for a single customer might need to be reused unexpectedly in a later product.

B. Certification Guidelines

Necessary requirements for reuse are raised by the certification guidelines for avionics. In Aerospace Recommended Practice (ARP) 4754A [4] for the reuse of existing certified systems and items it is required to:

- *"Meet the requirements of the new installation."* [4, Section 5.3.1.5]
- *"Even if no design changes are to be made to the system or item, the requirements to which the system or item was certificated should be validated according to the new application, and modified as necessary."* [4, Section 5.3.1.5]
- *"Any derived requirements, assumptions, compatibility of the interfaces and the operational environment should be validated as well."* [4, Section 5.3.1.5]
- *"Care should be taken in interface definitions that may be broad and may have been met in an earlier application but may not be met in the reuse instance due to things like different cable loading or bus termination conventions."* [4, Section 5.3.1.5]
- *"Verification activities should then be carried out against these requirements, including integration with the aircraft as required."* [4, Section 5.3.1.5]

For the process of reuse of unchanged remnants the authors of ARP 4754A [4] recommend the following practice:

- *"Evaluate the data available from the previous certification effort to determine which objectives are satisfied for the new application and which objectives require additional consideration."* [4, Section 6.5]
- *"Use reverse engineering to develop certification data necessary to satisfy the objectives of the certification requirements."* [4, Section 6.5]
- *"Use service history where applicable to satisfy the objectives of this document."* [4, Section 6.5]
- *"Specify the strategy for accomplishing compliance with the Certification Plan."* [4, Section 6.5]

Certification guidelines for the software development are provided in DO-178C[5]. Previously Developed Software (PDS) is defined as *"Software already developed for use. This encompasses a wide range of software, including COTS software through software developed to previous or current software guidance"* [5]. This includes the following situations: Modifications to PDS, Change of Aircraft Installation, Change of Application or Development Environment, Upgrading a Development Baseline, Software Consideration Management Considerations, and Software Quality Assurance Considerations. Different certification requirements are raised for all of these situations. For our approach we mainly consider the situation of "Change of Application or Development Environment". However, this may result in "Modifications to PDS" that is further elaborated in the next section that is concerned with the degree of reusability.

The cited aspects demonstrate, that reuse in avionics requires a considerable amount of additional effort, before the reused artifacts can be certified. This explains to a certain extent, why reuse in avionics is seen rather seldom.

C. The Degree of Reusability

In IEEE Standard Glossary of Software Engineering Terminology [6] reusability is defined as *"The degree to which a software module or other work product can be used in more than one computer program or software system"*. Where this definition focuses on the measurement of the common product characteristics, there are different other metrics that allow to judge about reuse of assets. For instance, in [7] the following categories of reuse metrics are mentioned: Size, reuse level, quality, effort, performance, and programmatic. These metrics incorporate many different aspects such as absolute and relative model size, amount of reused elements from models or code, reusable requirements, and defect density.

One additional aspect is the economical view on reusability of an asset, i.e., it is a matter of economical profitability. This can be measured and forecast by the measurements of e.g., development/simulation effort, but also the effort to understand the reused asset and the productivity measuring the effectiveness of producing new code [7].

The mentioned indicators do not consider particular characteristics of the projects and assets that influence these indicators from a technical perspective and which may lead to unfavorable decision in the context of reuse. We therefore execute the experiment of FCS reuse that shall demonstrate which technical additions are necessary when reusing a component.

III. Systematic approach

Our approach targets reuse for the ad hoc/heritage/legacy development approach. The reusable asset is the software component with its requirements at different abstraction levels and both, testing and integration artifacts, that shall be reused in a future target project. For this purpose, the component shall be identified in the already existing source project so that it can be reused in the target project. In this Section, we identify the requirements raised by certification guidelines in DO-178C. However, we do not consider additional technologies that are discussed in supplements of DO-178C.

A. Certification Considerations

The software conformity review as part of the software quality assurance process [5, Section 8.3 i.] requires traceability to the previous baseline and the approved changes, if certification credit is sought for the current software product baseline. The software verification plan describing the actions for the software verification process requires the reusable component to comply with DO-178C and to provide a description of methods to satisfy the objectives, if the initial software does not fulfill the objectives. In particular, [5, Section 12.1.5] requires traceability from the software product and the software life cycle data (artifacts) of the previous application to the new application. Furthermore, prior product certification/approval may be accepted as already certified software development process and the data as compliant to DO-178C objectives [5, Section 4.5.4.2]. This can be achieved by a mapping of DO-178C objectives to the prior product certification/approval objectives.

This brings us exactly to the requirement that we require as part of a systematic approach for reuse. We aim to benefit from PDS and its successful certification, where the source project is not dedicated to reuse. The idea is to extract reusable components and its attached artifacts from existing projects that fulfill the traceability within its data items. An additional requirement is to retrieve only those components and its data items that lead to the least dependency to other software components that it is interacting with in the source project.

B. Retrieval of Reusable Component in Source Project

For the source project we consider the following software life cycle data:

- System Requirements allocated to Software (SRATS)
- Software Requirements Data ([5, Sec. 11.9], High Level Requirements (HLR))
- Design Description ([5, Sec. 11.10], Low Level Requirements (LLR))
- Source Code (SC) [5, Sec. 11.11] that is decomposed into Computer Software Units (CSU), e.g., package(s), class(es), procedure(s), or function(s)
- Software Verification Cases ([5, Sec. 11.13b], Test Cases: SRATS test, HLR test, LLR test, CSU test)
- Trace Data [5, Sec. 11.21]

The trace data is the association between the contents of life cycle data items. In particular, we consider the following associations:

1) SRATS to HLR
2) HLR to LLR
3) LLR to SC (more precisely, CSU)
4) SRATS to SRATS test
5) HLR to HLR test
6) LLR to LLR test
7) CSU to CSU test

For the analysis of the considered artifacts the software life cycle data are encoded in a directed graph. A graph G is defined as a set of tuples with $G = (V, E)$, where V is a set of vertices and E is a set of edges. The mapping from software life cycle data to the structure of the graph is:

For the formal problem statement of the identification of components in a source project that have the least dependencies we define the following representation of life cycle data of the source project:

1) SRATS, HLR, LLR, and CSUs represent the vertices. Thus, $V = \{SRATS_i, HLR_j, LLR_k, CSU_l\}, i \leq I, j \leq J, k \leq K, l \leq L$ where I, J, K L is the number of SRATS, HLR, LLR, and CSU, respectively.
2) SRATS test, HLR test, LLR test, and CSU test also represent vertices. Thus, $V = \{SRATS.test_{i_s}, HLR.test_{j_t}, LLR.test_{k_p}, CSU.test_{l_q}\}$, $i \leq I$, $j \leq J$, $k \leq K$, $l \leq L$, where I, J, K L is the number of SRATS, HLR, LLR, and CSU, respectively, and $s \leq S, t \leq T, p \leq P, q \leq Q$ is the number of tests per requirement.
3) The associations SRATS to HLR, HLR to LLR, LLR to CSU, SRATS to SRATS test, HLR to HLR test, LLR to LLR test, and CSU to CSU test represent the set of edges. Thus, $E = \{SRATS_i - HLR_j, HLR_j - LLR_k, LLR_k - CSU_l, SRATS_i - SRATS.test_{i_s}, HLR_j - HLR.test_{j_t}, LLR_k - LLR.test_{k_p}, CSU_l - CSU.test_{l_q}\}$, where I, J, K, L is the number of SRATS, HLR, LLR, and CSU, respectively, and $s \leq S, t \leq T, p \leq P, q \leq Q$ is the number of tests per requirement. As these associations (or trace data) are of particular importance for the certification, we consider them to be part of the analysis, but assume that these can be skipped from the graph. This assumption is based on the observation that all tests are explicitly assigned to one requirement and by this do not change the structure of the graph with respect to the required analysis.
4) A caller/callee association between CSUs is added as an edge. This association is considered to take into account the fact that one CSU of one independent component can call a CSU of another independent component that renders the components depending on each other. As a result, these two components shall be considered as one independent component.

The above definition captures the life cycle data as a directed graph and allows to analyse the structure for different aspects. Our question is concerned with finding an independent component. For this, it is necessary to define the properties of the independent component (see also RQ1). The next step is then to transform the properties into the subgraph that can be searched for. Such an independent component shall have the following properties:

1) All CSUs of the component are traced to LLRs that are not traced to CSUs which do not belong to the component. If a CSU is added to the component, all LLRs, HLRs, and CSUs that are traced to these, are also part of the component.
2) All LLRs of the component are traced to HLRs that are not traced to HLRs which do not belong to the component.
3) CSUs of the component can call other CSUs whose LLRs and HLRs are also part of the component and fulfil bullet points 1 and 2.

Figure 1 shows two potential subgraphs. One is surrounded by a red oval and the other one by a blue one. However, these components are merged to one independent component as there exists a caller/callee association (shown by the red arrow) between one of the CSUs in each component.

We transform these properties into the following, more formal definition: The independent component is the subgraph that contains a set of CSUs, LLRs, and HLRs, so that all edges from HLR and LLR end in those LLR and CSUs that are already in the set. It is defined as a set $SG = \{CSU_1, ..., CSU_a, LLR_1, ..., LLR_b, HLR_1, ..., HLR_c\}$ with

$$\begin{aligned} &\forall \text{ CSU. } E(\text{CSU, LLR}) \rightarrow \text{LLR} \in SG \wedge \\ &\forall \text{ LLR. } E(\text{LLR, HLR}) \rightarrow \text{HLR} \in SG \wedge \\ &\forall i. E(\text{CSU}_i, \text{CSU}_{i+z}) \rightarrow \text{CSU}_{i+z} \in SG \end{aligned} \tag{1}$$

The proposed problem can be solved automatically with known algorithms, e.g. Depth First Search (DFS) [8]. The

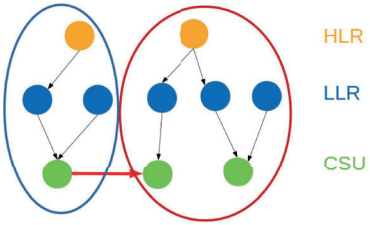


Fig. 1: Independent Component

found components can be identified visually as depicted in figure 2.

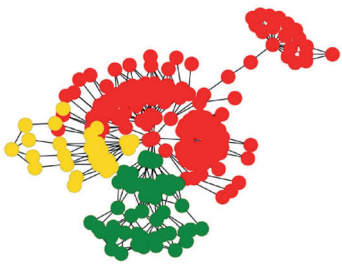


Fig. 2: Visualization of three components (red, green, and yellow): Each node represents a CSU, LLR, or HLR. The components have no edges which connect them with other components.

IV. Experience of FCS reuse

In this contribution we target the identification of challenges occurred during the reuse of an already existing software. In this Section we introduce the project in which open source software was reused and point out the most important observations on reuse.

A. Certification Considerations

We strive for the reuse of components and their attached traceability items (if available) in a target project previously developed in the source project. This falls into the category of "Change of Application or Development Environment" (cf. Section II) as we target an integration with other software than that used for the source project. We assume no change in the development environment. DO-178C [4, Section 12.1.3] describes required activities in addition to software life cycle process activities that address modifications. In order to identify, analyse, and re-verify changes to an application environment various activities are recommended by DO-178C, e.g.:

- Consider impact of change of software tools.
- Consider impact of complexity and sophistication of the programming language.
- Consider impact of autocode generator.
- Consider impact of compiler change.
- Consider impact of processor change.
- Verification of software interfaces should be conducted where PDS is used with different interfacing documents.

Only the last activity is relevant for this contribution. Different to certification requirements mentioned in II that are raised towards the process and artifacts, these activities affect the design decisions of the system under development where a PDS shall be reused.

B. Project and Requirements

When designing drones that shall be capable of autonomous flight in urban and indoor scenarios, the idea is to reuse existing FCS software and to extend its capabilities as required. This is what our project is about and a summary can be given by the following drone requirements:

- **(R1)** The architecture must use the existing hardware configuration as depicted in Figure 3.
- **(R2)** The ArduPilot[1] FCS shall stay as unmodified as possible.
- **(R3)** The architecture shall use as few additional software components as possible.

The first requirement describes our intended hardware setup. The second requirement was raised as modifications to existing FCS software may void the argument of proven performance for the FCS and also lead to instabilities in flight control. Intensive testing would be required after the modifications. The third requirement was raised as each additional driver leads to increased effort and costs.

Figure 3 presents the hardware architecture that shall be utilized in the development. The drone carries two computers, the open hardware Pixhawk[2] and the Intel® Aero Compute board. The FCS shall be operated on the Pixhawk and its functionality shall be called by the mission software that is hosted on the Intel Aero. The mission software includes e.g., the SLAM algorithm, communication, and health monitoring functionalities. This design is chosen due to separating the safety-critical FCS functionality from the other calculations.

The operational requirement that we raise for the development of the drone is:

- **(R4)** The drone shall fly without the usage of GPS.

With this requirement, it is not possible for the FCS to utilize GPS signals for the balance of the location deviation in the Kalman Filter [9], [10] of the FCS. Therefore, autonomous flights are enabled by the implementation of a solution for *Simultaneous Localization and Mapping* (SLAM) problem. This problem is concerned with the processing of mapping data in order to create a map of the environment and the positioning of the system under development (drone or robot) within this map. The challenge of the implementation of this problem is manifold, such as: (1) The algorithm has to identify

[1]See https://ardupilot.org/, last access 24/01/2020.
[2]See https://pixhawk.org/, last access 24/01/2020.

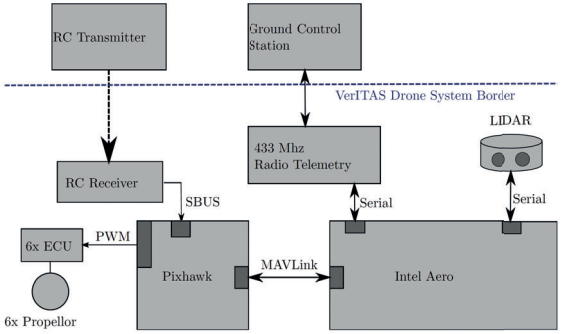


Fig. 3: VerITAS drone hardware architecture

landmarks. (2) The algorithm has to recognize landmarks after the drone or robot has moved.

C. Known Solutions

Since ArduPilot is a well-known and broadly used FCS, there are plenty of other projects where the SLAM algorithm is used together with ArduPilot for flight control. [11] demonstrates a system design of a quad-rotor drone with Visual SLAM (VSLAM) for attitude estimation. The attitude is computed on a more powerful companion computer (second onboard computer) and sent to ArduPilot for flight control. A modified version of ArduPilot is used here to communicate with the companion computer and uses the external attitude for flight control. The modification is necessary as the described communication path is not supported by the original ArduPilot.

A similar method for a LIDAR based SLAM algorithm has been proposed in [12]. They use an Android phone as a mission computer and for executing the SLAM algorithm. The results are sent to the ArduPilot FCS for the flight-control of the drone and are meant to spoof GPS data. This work does not propose the weighted combination of both, the SLAM aircraft state and the FCS aircraft state, for the flight control, as it is implemented in our approach.

D. Reuse Considerations

As the FCS software was not developed by the authors, we first identified all necessary functional properties, in order to find out which functionality can be reused and which can be handled with system configuration.

Interfaces: First, all sensors supported by ArduPilot were reviewed. These are: GPS, optical flow, motion capture systems, ultrasonic sensors, LASER range sensors, and rotating LIDAR.

Flight modes: From the functional perspective, we consider the flight modes that drive the functionality of the FCS. Flight modes[3] that could be utilized for autonomous flight are: Acro, Stabilize, Guided, Land, and Brake.

Inputs and configuration: Finally, we identified the inputs and configurations of the Extended Kalman Filter (EKF) that is implemented in the FCS. ArduPilot's EKF2 has a state vector[4] of length 24. It holds the attitude, the velocity (North, East, Down), the position (North, East, Down), the gyro bias offsets (X,Y,Z), the gyro scale factors (X,Y,Z), the Z accel bias, the earth magnetic field (North, East, Down), the body magnetic field (X,Y,Z), and the wind Velocity (North, East).

Parameter configuration: The FCS provides configuration parameters to setup the software according to the aircraft needs. The following parameters have been identified for our purpose:

- $EK2_GPS_TYPE$: This parameter sets the number of used GPS sensors. In our case it is set to 3 to disable GPS.
- $EK2_IMU_MASK$: This parameter describes the number of used inertial measurement units (IMUs). It can be set to 1, 2, or 3 indicating the use of the first IMU, the second IMU, or both IMUs, respectively.
- $EK2_POSNE_M_NSE$: This parameter sets the horizontal position measurement noise to the chosen value. The minimum value is 0.1 meters.
- $EK2_POS_I_GATE$: This parameter sets the percentage number of standard deviations, which will be accepted for the GPS position measurement innovation. The unit is permille [‰]. The values can be in the range between 100 ‰ and 1,000 ‰. A lower value within the range will result in more measurement samples being rejected.

MAVLink protocol: ArduPilot communicates with other

[3]All flight modes of ArduPilot and their description can be found at https://ardupilot.org/copter/docs/flight-modes.html, last access 29/01/2020.

[4]https://ardupilot.org/dev/docs/ekf2-estimation-system.html, last access 29/01/2020.

devices using the MAVLink protocol[5]. The following messages are identified as useful for our needs:

- HEARTBEAT – Message, which is sent periodically by a system component to all other system components. It contains important status information including the current flight mode and the current armed status.
- SET_GPS_GLOBAL_ORIGIN – This message sets the origin of the local coordinate frame of the FCS.
- MESSAGE_INTERVAL – This message requests to receive a specified message at a specified frequency. This can be used to get certain aircraft data at a reasonable frequency.
- PARAM_REQUEST_READ – This message requests the current value of a specified parameter.
- PARAM_SET – This message sets a parameter to a specified value. A parameter value has to be read once before being able to set it.
- PARAM_VALUE – This message contains the new value of a parameter. After a value is requested to change, this is the response that is sent back to the requester.
- COMMAND_LONG – This message sends a command to a component. There is a long list of supported commands available in the MAVLink specification. With this message it is possible to arm the throttle, set the current flight mode, or request a reboot of the system.
- TIMESYNC – This message is used for time synchronization.
- ATT_POS_MOCAP – This message can be used for external position input of the ArduPilot EKF2. It takes a North-East-Down (NED) position and an attitude quaternion as a payload.
- SET_POSITION_TARGET_LOCAL_NED – This message is used in flight mode Guided to set the current target coordinates of the aircraft.
- LOCAL_POSITION_ NED – This message contains the current estimate of the position and velocity by ArduPilot EKF2.
- ATTITUDE – This message contains the current attitude in Euler angles.

E. Solution

Figures 4, 5, and 6 show possible solutions that were developed in the course of the design.

Option A: SLAM only The SLAM-only software architecture is shown in Figure 4. It uses a modified version of the ArduPilot FCS. The ArduPilot EKF2 on the FCC shall be turned off here. ArduPilot will only be in charge of the flight controls and use the SLAM aircraft state estimation for its control loop. IMU or magnetometers need to be available for the SLAM algorithm. Both, the Pixhawk and the Intel Aero do have the required sensors. However, this option is not further considered as the stock software for the Intel Aero cannot be used due to the incompatibility with the operating system of the board and therefore additional drivers for the on-board sensors would have to be developed.

Option B: SLAM as sensor In this solution, depicted in Figure 5, the SLAM algorithm and the ArduPilot EKF2 run in parallel. Once the SLAM algorithm has finished one iteration, the current aircraft state estimation is sent to the ArduPilot EKF2 via one of the external position sensor interfaces.

Option C: Distributed Localization System (DLS) The option shown in Figure 6 is based on option B, but eliminates redundancies. First, only the IMU of the Pixhawk is utilized in order to exclude variances between these two. These are expected to occur as the boards are mounted on different locations in relation to the center of gravity of the drone. Second, ArduPilot EKF2 and the SLAM algorithm both predict the aircraft state based on the IMU data (odometry) in option B. We implement a loop that exploits the state estimations of both, the EKF2 and SLAM. For this, we establish the estimations of EKF2 that is based on the SLAM algorithm (as already seen in option B) and introduce the estimation within the SLAM algorithm based on the state of EKF2. Having this design, the EKF2 state is sent to the SLAM algorithm that is used for landmark re-observation. The SLAM module then uses the re-observed landmarks to improve the currently estimated state (hence to improve the odometry). The ArduPilot EKF2 treats the SLAM-improved aircraft state as a sensor input and performs the next iteration.

Table I presents a comparison of the design options and their satisfaction of raised requirements. The chosen option for the implementation is option C as it fulfills the requirements best. The chosen design was implemented by introducing a new software component, the Flight Companion, for the communication between the FCS and other software components hosted on the mission computer, e.g., the SLAM component. This component (1) defines the interfaces, (2) implements the finite state machine for autonomous flight with appropriate messages to FCS, (3) sets parameters of the FCS, (4) synchronizes the time, and (5) implements the influence of the EKF2 aircraft state by the SLAM aircraft state.

As the EKF2 aircraft state considers both estimations, the impact of the estimations has to be determined. The following formula describes this question:

$$x_{SLAM} = x_{EKF2} + \lambda(x_{real} - x_{EKF2}) + e, e \sim \mathcal{N}(0, \sigma^2) \quad (2)$$

$$0 \leq \lambda \leq 1 \quad (3)$$

The innovation factor λ represents value according to which the currently estimated EKF2 aircraft position s_{EKF2} updates the real aircraft position x_{real}. This is described with the innovation factor λ. It was determined with simulations to the values of greater than 0.4. A similar innovation factor was introduced for the attitude.

V. Conclusion

In this paper we presented two solutions for reuse:

- The (automatic) identification of components could already be applied successfully on example projects when ad hoc development was performed. We also assume that

[5]MAVLink is a lightweight protocol for message exchange between a drone and its ground control station. For further information see https://mavlink.io/en/, last access 26/04/2019.

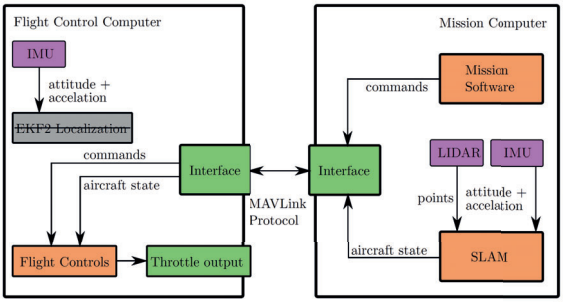


Fig. 4: Solution A: SLAM only

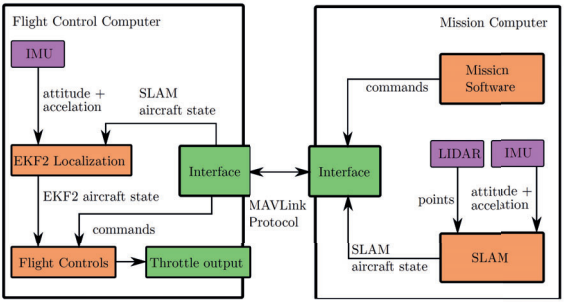


Fig. 5: Solution B: SLAM as sensor

TABLE I: Comparison of architectures

Requirement	SLAM-only	Sensor SLAM	DLS
Re-use of existing software	Yes, but modifications necessary	Yes, no modifications necessary	Yes, no modifications necessary
Modification of ArduPilot	Yes	No	No
Used kinematic sensors	4 Pixhawk sensors plus Intel Aero sensors	4 Pixhawk sensors plus Intel Aero sensors	4 Pixhawk sensors
Required additional software	MAVLink interface, SLAM implementation, ArduPilot modification, Intel Aero Drivers (IMU + LIDAR)	MAVLink interface, SLAM implementation, Intel Aero Drivers (IMU + LIDAR)	MAVLink interface, simplified SLAM implementation, Intel Aero Drivers (LIDAR)
Dependencies of SLAM	LIDAR driver, Intel Aero IMU driver	LIDAR driver, Intel Aero IMU driver	LIDAR driver, EKF2

this methodology can be applied in model based development, since many models can be viewed as vertices and connecting edges. However, this remains subject to future work.

- The reuse of FCS demonstrated further challenges on reuse activity. We identified additional aspects to be considered during reuse that are not obvious at first glance. On the way to understanding the reuse guiding considerations, we developed the solution of the DLS.

A. Conclusion on Component Identification

The traceability between life cycle data in avionics software has been mapped to a directed graph, where the life cycle data are represented by vertices and the traceability by edges between them. Such graphs have been created from real project data and stored in a graph database. From a process point of view the following properties have been investigated by the execution of appropriate database queries:

1) The completeness of the directed (traceability) graph is checked according to the definition in Section III. The

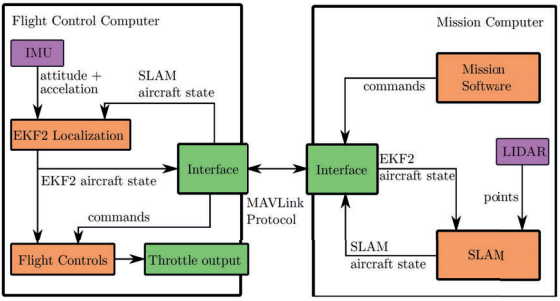


Fig. 6: Solution C: Distributed Localization System

following steps are recommended only, if the calculated completeness is sufficient.

2) According to the definition for the properties of a potentially reusable component as explained in Section III the populated graph database is queried. As a result, the identified components are output.
3) Additional metrics define the reuse quality of the identified component. These describe the extent, to which they have to be adapted for their integration into the target project.

The analysis of the project data shows, that the selected database structure is covering the requirements for performing the steps mentioned above. The data covers real project examples, where the amount of HLRs is in the range between 1,000 and 2,000. Though the computation time is in a reasonable amount of time (nightly build), the queries are not specified with scalability in mind. In order to bring the expertise into a productive environment, further optimisations are necessary.

B. Conclusion on FCS Reuse

The requirements of the product and the process mentioned in Section IV-B were met as shown in Table I. We particularly confirm the FCS reuse without any changes except the configurations listed in Section IV-D.

However, the effort of the reuse was higher than we expected. On project start we faced the fact, that the functionality of the FCS needed to be fully understood by the developer. This includes the design of the software (e.g. the EKF), the implemented modes and the parameters. This understanding goes beyond the required activity mentioned in Section IV-A, where the main attention is paid to the interfaces.

Already during early design stages we realized that we will need to introduce an additional component for our use case. The reasons for the introduction of the Flight Companion is given in Section IV-E. Moreover, despite reuse of the full component, not all functionalities were used, but instead deactivated. For instance, we limited the modes and commands that are sent to FCS to a certain set as presented in Section IV-D. Additional to this, we found some functionality in the FCS that could not be reused, as slightly different behavior was required by the our use case, and which had to be implemented again in the Flight Companion.

Acknowledgment

Part of the work described in this article has been conducted as part of the research project ProMethEus (FKZ 20Y1508C). It has been government-funded by the Federal Republic of Germany. The funding organization is the German Federal Ministry for Economic Affairs and Energy according to a resolution of the Lower House of German Parliament.

References

[1] P. Mohagheghi and R. Conradi, "Quality, productivity and economic benefits of software reuse: A review of industrial studies," Empirical Software Engineering, vol. 12, pp. 471–516, 10 2007.
[2] J. Varnell-Sarjeant, A. Amschler Andrews, J. Lucente, and A. Stefik, "Comparing development approaches and reuse strategies," Inf. Softw. Technol., vol. 61, no. C, pp. 71–92, May 2015. [Online]. Available: http://dx.doi.org/10.1016/j.infsof.2015.01.002
[3] B. Hardung, T. Kölzow, and A. Krüger, "Reuse of software in distributed embedded automotive systems," in Proceedings of the 4th ACM International Conference on Embedded Software, ser. EMSOFT '04. New York, NY, USA: ACM, 2004, pp. 203–210. [Online]. Available: http://doi.acm.org/10.1145/1017753.1017787
[4] S. Aerospace, "Aerospace Recommended Practice ARP4754, Guidelines for Development of Civil Aircraft and Systems," Tech. Rep., 2010.
[5] RTCA SC-205 and EUROCAE WG-71, "Software Considerations in Airborne Systems and Equipment Certification," Tech. Rep., 2011.
[6] "Ieee standard glossary of software engineering terminology," IEEE Std 610.12-1990, pp. 1–84, Dec 1990.
[7] J. Varnell-Sarjeant and A. A.A., "Comparing reuse strategies in different development environments," Advances in Computers, vol. 97, pp. 1–47, 12 2015.
[8] S. Even, Graph Algorithms, 2nd ed. Cambridge University Press, 2011.
[9] R. E. Kalman and Others, "A new approach to linear filtering and prediction problems," Journal of basic Engineering, vol. 82, no. 1, pp. 35–45, 1960.
[10] R. Stratonovich, "Optimum nonlinear systems which bring about a separation of a signal with constant parameters from noise," Izvestiya vuzov SSSR, Seriya Radiofizika, vol. 2, p. 862–901, 1959.
[11] C. B. Yigit and E. Altug, "Visual attitude stabilization of a unmanned helicopter in unknown environments with an embedded single-board computer," in 2012 IEEE International Symposium on Robotic and Sensors Environments Proceedings, Nov 2012, pp. 49–54.
[12] E. Chirtel, R. Knoll, C. Le, B. Mason, N. Peck, J. Robarge, and G. C. Lewin, "Designing a spatially aware, autonomous quadcopter using the android control sensor system," in 2015 Systems and Information Engineering Design Symposium, April 2015, pp. 35–40.

Dipl.-Ing.-Päd. (TU) Dietrich Schneider

„THERMOGRAFIE MIT DROHNEN“

Physikalisch-technische Grundlagen und praktische Anwendungen in der Industrie

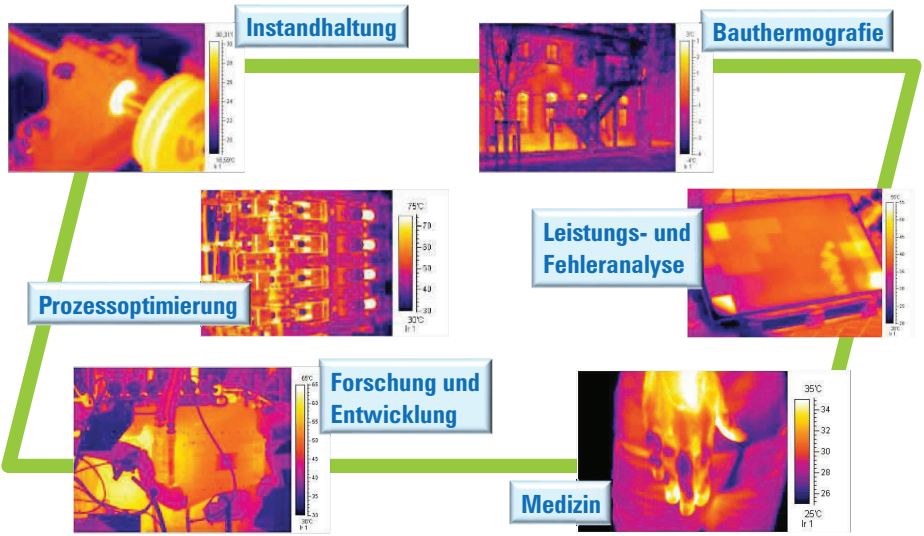
IBS Workshop MAV 2019

Laubusch, 06.11.2019

Infrarot-Thermografie

HOCHSCHULE ANSBACH

Anwendungsgebiete – Übersicht



Infrarot-Thermografie – was ist das?

Infrarot-Thermografie/Pyrometrie –

Infrarot-Temperaturmessverfahren, bei dem berührungslos die Intensität der elektromagnetischen Strahlung eines Körpers gemessen wird.

- Jeder Körper mit einer Temperatur oberhalb des absoluten Nullpunktes von 0 K (-273,15 °C) sendet elektromagnetische Strahlung aus.
- Für die Infrarottechnik ist der Wellenlängenbereich oberhalb des sichtbaren Spektrums (0,38 – 0,76 μm) interessant.
- Die technische Temperaturmessung nutzt den als **thermisches Infrarot** bezeichneten Wellenlängenbereich von **0,8 – 14 μm**. Hier wird der größte Teil der gesamten Wärmestrahlung emittiert.

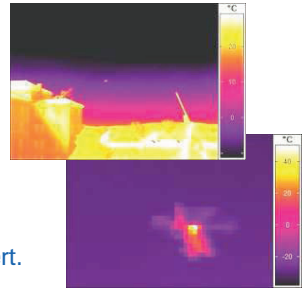
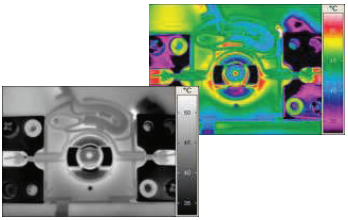


Abb.: Thermogramm „Hubschrauber"

Vorteile der Infrarot-Thermografie

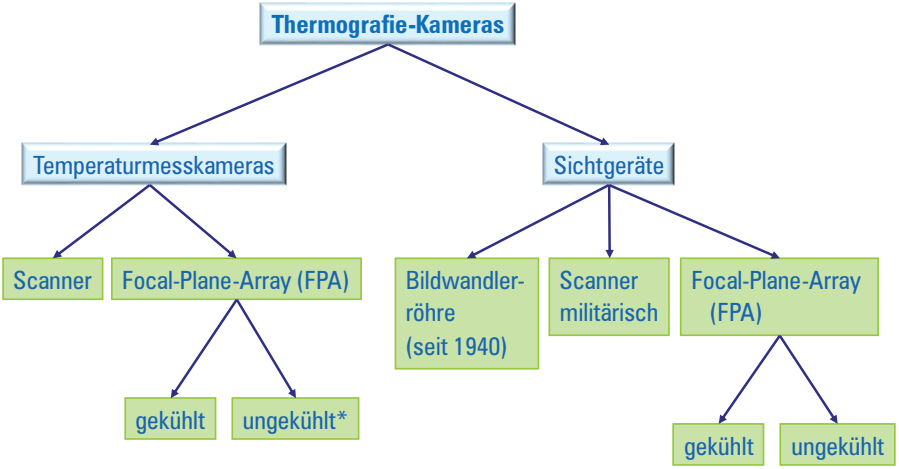
- berührungsloses Temperaturmessverfahren (praktisch rückwirkungsfrei)
- Echtzeitbetrieb/ -anzeige (50 Bilder/s)
- bildgebend (flächenhafte Temperaturinformation)
- genau (portable IR-Kameras: typ. ± 2 K)
- empfindlich (Temperaturauflösung: typ. < 0,05 K)



- entfernte, heiße, bewegte oder spannungsführende Objekte gefahrlos detektierbar
- Untersuchungen im laufenden Prozess
- hygienisch und ohne mechanische Einwirkung auf (empfindliche) Oberflächen

- **aber:** Objekt muss optisch sichtbar sein; i.d.R. nur Oberflächentemperaturen messbar !

Klassifizierung – Detektortechnologien



* (Mikro-)Bolometer = Strahlungsmesser (griechisch: bolo = Strahl)

Infrarot-Thermografie

HOCHSCHULE ANSBACH

FPA-Kamera – Aufbau

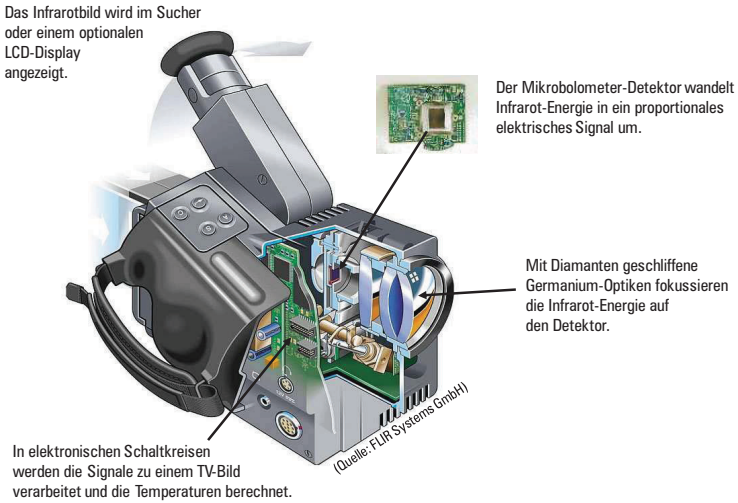


Abb. : Aufbau einer FPA-Kamera

Geräteparameter – Übersicht

- Die Qualität einer Infrarot-Kamera wird hauptsächlich durch drei Geräteparameter bestimmt. Je nach Einsatzzweck und Messaufgabe muss eine geeignete IR-Kamera ausgewählt werden:

thermische Auflösung

NETD (z.B. 0,03 K)

geometrische Auflösung

FOV (z.B. 25 °) und IFOV (z.B. 1,4 mrad)

zeitliche Auflösung

Bildrate (z.B. 50 Hz)

Infrarot-Thermografie

HOCHSCHULE ANSBACH

FPA-Kamera – Bildauflösung

- Ein wichtiges Qualitätsmerkmal von FPA-Kameras ist die **geometrische Bildauflösung**. Eine hohe Pixelanzahl pro Sensorfläche erlaubt eine hochaufgelöste Darstellung des Objekts im Thermogramm (Erhöhung der Messgenauigkeit!).

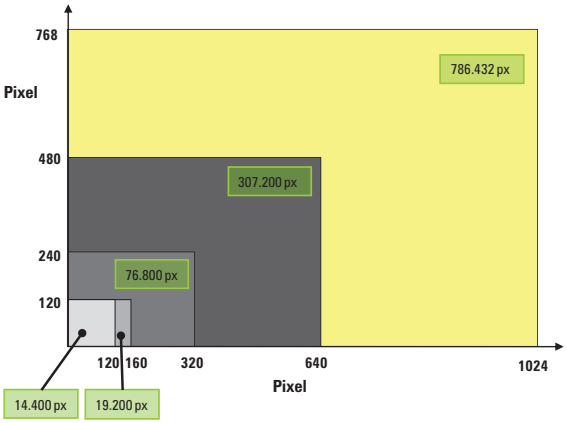


Abb.: Typische Bildauflösungen von IR-Kameras

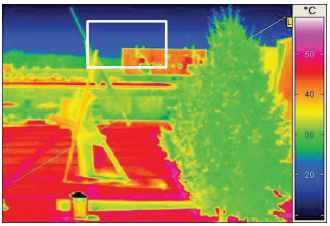


Abb.: Thermogramm „Campus HS Ansbach"

Geometrische Auflösung – reale Messfleckgröße

- Praxisbeispiel:

  - IFOV = 1,2 mrad
  - Objektabstand d = 0,5 m
  - Kabeldurchmesser ∅ 2 mm
  - ges.: x_{real} [mm] (realer Messfleck)

- $x_{real} = IFOV \cdot d \cdot f_{Optik}$
  $= 1{,}2 \text{ mrad} \cdot 0{,}5 \text{ m} \cdot 3$
  $= 1{,}8 \text{ mm}$

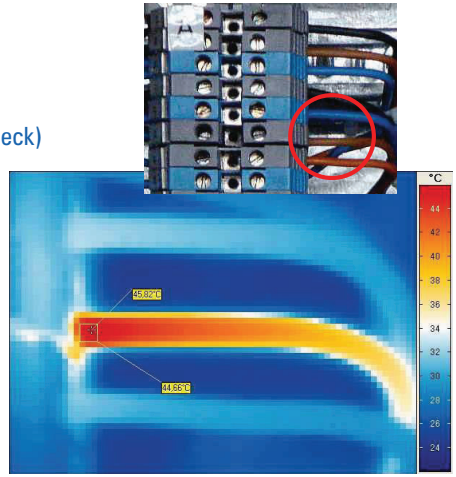
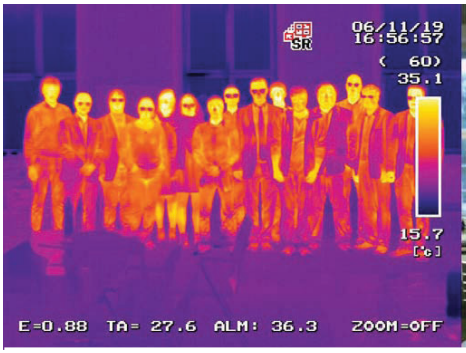


Abb. : Kabelklemme

Infrarot-Thermografie

HOCHSCHULE ANSBACH

Praktische Thermografie mit eigener oder Leihkamera

- *„Grau, teurer Freund, ist alle Theorie.*
  Und grün des Lebens goldner Baum." (Goethe, Faust I)