5-2014

Cyber-Physical Systems Enabled By Unmanned Aerial System-Based Personal Remote Sensing: Data Mission Quality-Centric Design Architectures

Calvin Coopmans
Utah State University

CYBER-PHYSICAL SYSTEMS ENABLED BY UNMANNED AERIAL SYSTEM-BASED PERSONAL REMOTE SENSING: DATA MISSION QUALITY-CENTRIC DESIGN ARCHITECTURES

by

Calvin Coopmans

A dissertation submitted in partial fulfillment
of the requirements for the degree

of

DOCTOR OF PHILOSOPHY

in

Electrical Engineering

Approved:

Dr. YangQuan Chen Major Professor	Dr. Mac McKee Committee Member
Dr. David Geller Committee Member	Dr. Edmund Spencer Committee Member
Dr. Ryan Gerdes Committee Member	Dr. Mark R. McLellan Vice President for Research and Dean of the School of Graduate Studies

UTAH STATE UNIVERSITY
Logan, Utah

2014

Abstract

Cyber-Physical Systems Enabled by Unmanned Aerial System-Based Personal Remote Sensing: Data Mission Quality-Centric Design Architectures

by

Calvin Coopmans, Doctor of Philosophy

Utah State University, 2014

Major Professor: Dr. YangQuan Chen
Department: Electrical and Computer Engineering

In the coming 20 years, unmanned aerial data collection will be of great importance to many sectors of civilian life. Of these systems, Personal Remote Sensing (PRS) Small Unmanned Aerial Systems (sUASs), which are designed for scientific data collection, will need special attention due to their low cost and high value for farming, scientific, and search-and-rescue uses, among countless others. Cyber-Physical Systems (CPSs: large-scale, pervasive automated systems that tightly couple sensing and actuation through technology and the environment) can use sUASs as sensors and actuators, leading to even greater possibilities for benefit from sUASs. However, this nascent robotic technology presents as many problems as possibilities due to the challenges surrounding the abilities of these systems to perform safely and effectively for personal, academic, and business use. For these systems, whose missions are defined by the data they are sent to collect, safe and reliable mission quality is of highest importance. Much like the dawning of civil manned aviation, civilian sUAS flights demand privacy, accountability, and other ethical factors for societal integration, while safety of the civilian National Airspace (NAS) is always of utmost importance. While the growing popularity of this technology will drive a great effort to integrate sUASs into the NAS, the only long-term solution to this integration problem is one of proper

architecture. In this research, a set of architectural requirements for this integration is presented: the Architecture for Ethical Aerial Information Sensing or AERIS. AERIS provides a cohesive set of requirements for any architecture or set of architectures designed for safe, ethical, accurate aerial data collection.

In addition to an overview and showcase of possibilities for sUAS-enabled CPSs, specific examples of AERIS-compatible sUAS architectures using various aerospace design methods are shown. Technical contributions include specific improvements to sUAS payload architecture and control software, inertial navigation and complementary filters, and online energy and health state estimation for lithium-polymer batteries in sUAS missions. Several existing sUASs are profiled for their ability to comply with AERIS, and the possibilities of AERIS data-driven missions overall is addressed.

(202 pages)

Public Abstract

Cyber-Physical Systems Enabled by Unmanned Aerial System-Based Personal Remote Sensing: Data Mission Quality-Centric Design Architectures

by

Calvin Coopmans, Doctor of Philosophy

Utah State University, 2014

Major Professor: Dr. YangQuan Chen
Department: Electrical and Computer Engineering

In the coming 20 years, unmanned aerial data collection will be of great importance to many sectors of civilian life. Of these systems, Personal Remote Sensing (PRS) Small Unmanned Aerial Systems (sUASs), or "data drones," will need special attention due to their low cost and high value for farming, scientific, and search-and-rescue uses, among countless others. Cyber-physical systems (large-scale, pervasive automated systems that tightly couple sensing and actuation through technology and the environment) can use sUASs as sensors and actuators, leading to even greater possibilities for benefit from sUASs. However, this nascent robotic technology of small unmanned aerial systems (sUASs) presents as many problems as new possibilities due to the challenges surrounding the abilities of these systems to perform safely and effectively for personal, academic, and business use. For these systems, whose missions are defined by the data they are sent to collect, safe and reliable mission quality is of highest importance. Much like the dawning of civil manned aviation, civilian sUAS flights demand privacy, accountability, and other ethical factors for societal integration, while safety of the civilian National Airspace (NAS) is always of utmost importance. While the growing popularity of this technology will drive a great effort to integrate sUASs into the NAS, the only long-term solution to this integration problem

is one of proper architecture. In this research, a set of architectural requirements for this integration is presented: the Architecture for Ethical Aerial Information Sensing or AERIS. AERIS provides a cohesive set of requirements for any architecture or set of architectures designed for safe, ethical, accurate aerial data collection.

To everyone. To the future!

Acknowledgments

This dissertation has been a very long time in the making. I applied to Utah State University seeking a Ph.D degree seven years ago, and without the help of several people I would not have succeeded in even starting graduate work. Although once thanked in my master's thesis, thanks again to Dr. David Klumpar of the Montana State University Space Science and Engineering Lab (SSEL) for providing me with the tools, environment, and employment to learn real aerospace engineering skills.

Dr. William A. Hiscock, previously of the Montana State University Physics Department, was kind enough to send me to Jet Propulsion Laboratories for one of the best summers of my life. You are deeply missed.

I still owe an incalculable debt of gratitude to Dr. Gary Bohannan, who showed me—and continues to show me—crucial mentorship and guidance. Thank you, Gary.

Thanks to Dr. Joseph A. Shaw and Dr. Bob Gunderson (formally of the Center for Self-Organizing Systems at Utah State University) and Dr. Mark D. Ivey from the Montana State University Electrical and Computer Engineering Department for their support by way of writing me letters of recommendation and for their assistance and guidance in applying to the graduate program. Thanks again to Dr. Todd Moon of the USU ECE Department; for his crucial support in the graduate program I am most grateful. Thank you, Mary Lee Anderson, ECE graduate advisor and department secretary extraordinaire, for always keeping me on track, from drowning in a sea of paperwork, and for your attention to the text and formatting during the editing of this and all other ECE theses.

During the development of AggieAir I have had the pleasure of working with many talented engineers and researches. Austin Jensen has been wonderfully helpful and a much appreciated source of sanity over these past years. Here's to many more! Brandon Stark, now of U.C. Merced, has been a trusted research partner and is a pro at processing papers. Michal Podhradský has been with AggieAir, and out, and back again; in the process many great things have come to pass like RT-Paparazzi and the battery estimation paper

included partly in this dissertation. Nathan Hoffer has one of the strongest stomachs for data processing I have ever encountered, as well as the uncanny ability to keep his cool when managing twenty stressed-out engineers fighting uncooperative robots.

I am very thankful for my family, who have kept their suspension of belief in my ideas and supported me this whole way. My wife, Stacey Frisk–ever a positive force in my life and research–balances me, pushing me away from stacks of papers and piles of circuit boards, towards dusty trails, snowy hills, and rapid rivers.

Very heartfelt thanks are due to my dissertation committee members: Drs. Mac McKee, David Geller, Edmund Spencer, and Ryan Gerdes. Giving this research your attention is one of the highest honors I can receive. Feedback given from the defense of this dissertation–and the motivation to "get it all in one place" resulted in Section 4.3 (Dr. Geller and all members), Section 4.5.4 (Drs. Gerdes and Spencer), and Section 4.7.7 (all committee members).

Lastly, but not in the least bit least: My advisor, Dr. YangQuan Chen, who has believed in me from the beginning of this adventure, and continues to push me toward greater things. I am forever grateful for your ability to see untapped potential, as well as for supporting my own. Thank you, Dr. Chen.

Calvin Coopmans

Contents

Page

Abstract . . . . . **iii**

Public Abstract . . . . . **v**

Acknowledgments . . . . . **viii**

List of Tables . . . . . **xiii**

List of Figures . . . . . **xiv**

Acronyms . . . . . **xx**

1 Introduction **1**
1.1 Personal Remote Sensing 1
1.2 Unmanned Aerial Remote Sensing 4
1.3 Dissertation Theme and Contribution 5
1.4 Dissertation Organization 7
1.5 Chapter Summary 7

2 AggieAir Unmanned Aerial Vehicle-Based Sensing Platforms **8**
2.1 AggieAir Fixed-Wing Unmanned Platform 11
2.2 AggieAir Rotary-Wing Platform 12
2.3 Chapter Summary 12

3 Cyber Physical Systems Enabled by sUAS Remote Sensing **14**
3.1 CPS Enabled by PRS: Wildfire Tracking and Fighting 16
3.2 CPS Enabled by PRS: Real-time Irrigation Control 17
3.3 CPS Enabled by PRS: Algal Bloom Tracking for Alternative Energy 19
3.4 Chapter Summary 22

4 Architectures for Ethical Remote Information Sensing **23**
4.1 Cyber-Physical Philosophy 27
4.2 Biologically-Layered Robotic Scheduling and Functionality 28
4.2.1 The Triune Brain 29
4.2.2 Robotic Processing Requirements 30
4.2.3 Ethical Behavior as Implementation of Morals: Robots vs. Weapons 36
4.2.4 Successful Data Missions 37
4.3 Existing Civil UAS Architectural and Ethical Research 38
4.4 The Three Elements of Ethical Aerial Scientific Data Collection 40
4.4.1 Airspace Management for Safety 40
4.4.2 Data Mission Success ("Dataworthiness") 42

4.4.3 Ethics: Privacy by Design . . . 43
4.5 AERIS PRS Architectures and Implementations . . . 44
4.5.1 Multiscale UAS-CPS Architectures . . . 44
4.5.2 General Small UAS Control Architecture . . . 44
4.5.3 Optimal Multicore CPU Configuration Scenarios . . . 45
4.5.4 Implementation of Architectures: Formality, Security, and Safety . . . 47
4.5.5 Robotic Computing Hardware Outlook . . . 49
4.6 Control, Estimation, and Behaviors in AERIS . . . 49
4.7 An Analysis of Selected Open Source Robotic and UAS Architectures and Comparison to AERIS . . . 52
4.7.1 RT-Paparazzi . . . 54
4.7.2 PixHawk PX4 . . . 56
4.7.3 ROS . . . 57
4.7.4 DJI Phantom 2 Vision . . . 58
4.7.5 Procerus Kestrel . . . 60
4.7.6 AggieAir 2.0 . . . 61
4.7.7 AERIS in Current UASs . . . 63
4.8 Chapter Summary . . . 66

5 A Payload Verification and Management Framework for Small UAV-based Personal Remote Sensing Systems . . . 69
5.1 Data Mission Assurance . . . 71
5.2 AggieAir Architecture . . . 72
5.2.1 Personal Remote Sensing Payload Requirements and Functionality . . . 73
5.2.2 ISaAC: The Intelligent Safety and Airworthiness Co-Pilot . . . 74
5.3 AggieCap Modular Software Architecture . . . 76
5.3.1 AggieCap Design . . . 77
5.3.2 AggieCap Implementation . . . 77
5.4 Standardized Testing and Verification for DMA . . . 78
5.5 AggieCap Results . . . 80
5.6 AggieCap Payload Implementation Example . . . 80
5.7 AERIS Significance of Payload Management and Chapter Summary . . . 81

6 Small Unmanned Aerial System Navigation and a Fractional-Order Complementary Filter . . . 83
6.1 IMU Basics and Notation . . . 85
6.2 Sensor Packages . . . 87
6.3 Attitude Estimation Algorithms . . . 89
6.3.1 General Extended Kalman Filter . . . 90
6.3.2 Quaternion-Based Extended Kalman Filter . . . 90
6.3.3 Euler Angle-Based Extended Kalman Filter . . . 92
6.3.4 AggieEKF: GPS Aided Extended Kalman Filter . . . 93
6.3.5 Complementary Filters . . . 94
6.4 Example Low-Cost IMUs . . . 94
6.4.1 Attitude Estimation IMUs . . . 94
6.4.2 GPS-Coupled IMUs . . . 94
6.4.3 Hobbyist Grade IMUs . . . 95

6.4.4 IMU Comparisons . . . 96
6.4.5 Sensors and Navigation . . . 97
6.4.6 Kalman Filtering for Small UASs . . . 99
6.4.7 Complementary Filters . . . 99
6.5 Frequency Domain Complementary Filters (FDCFs) . . . 101
6.5.1 FDCF Example – Modeless Second Order Filter . . . 103
6.5.2 Other FDCF Design Options . . . 106
6.6 State Space Complementary Filters . . . 106
6.6.1 SSCF Example–Using a Gain Controller . . . 109
6.6.2 SSCF Example – Using a PI Controller . . . 110
6.6.3 Other SSCF Design Options . . . 115
6.7 Fractional-Order Complementary Filters . . . 116
6.7.1 Fractional-Order Calculus . . . 116
6.7.2 Fractional-Order Filtering . . . 119
6.7.3 Alpha-stable Noise . . . 122
6.7.4 Fractional-Order Complementary Filter Simulations . . . 123
6.7.5 Fractional-Order Complementary Filter Simulation Results . . . 125
6.8 Guidelines for Complementary Filter Use in sUAS . . . 127
6.9 AERIS Significance of Navigation and Chapter Summary . . . 130

7 Battery State-of-Charge Aware Altitude Controller for Small, Low-Cost Multirotor Unmanned Aerial Vehicles . . . 132
7.1 sUAS Altitude Control Obstacles . . . 133
7.2 Battery and Actuator Models . . . 134
7.2.1 Battery Model . . . 134
7.2.2 Actuator Model . . . 137
7.3 Comparison of Existing Solutions . . . 140
7.3.1 Classical Control . . . 141
7.3.2 Adaptive Control . . . 142
7.4 Battery State-Of-Charge-Based Controller . . . 142
7.5 Battery Dynamics Measurements . . . 143
7.5.1 Instrumentation . . . 143
7.5.2 Experimental Setup . . . 145
7.5.3 Experimental Verification . . . 149
7.6 Flight Verification . . . 152
7.6.1 Battery Monitoring Subsystem . . . 153
7.6.2 SOC Estimation . . . 153
7.6.3 Flight Data . . . 155
7.7 AERIS Significance of Battery Estimation and Chapter Summary . . . 157

8 Conclusion . . . 158

References . . . 161

Vita . . . 178

List of Tables

Table Page

4.1 Table of FAA safety priorities. 40

4.2 UAS behaviors which are simple and useful. 53

4.3 Table of UAS and autopilots viable for personal remote sensing. 54

4.4 Table of AERIS elements and AggieAir features. 65

5.1 AggieCap sensor and actuator list. 80

6.1 IMU categories. 89

6.2 IMU comparisons. 97

7.1 Least-squares spline approximation for thrust measurements. 149

7.2 Piecewise-linear approximation of the thrust bonus. 151

7.3 Laboratory experiment error. 152

List of Figures

Figure		Page
1.1	Passive remote sensing.	2
1.2	Active remote sensing requires some kind of stimulation to reveal the desired data.	2
1.3	sUAS-sized sensors for water management: thermal and short-wave infrared imagers from Infrared Cameras Inc. and Goodrich.	5
1.4	A 10cm-resolution mosaic of thermal imagery collected by AggieAir from test flight under COA in Cache Junction, UT.	6
2.1	AggieAir: Closing the UAV-based science-to-information loop (from Dr. Jensen).	9
2.2	AggieAir sUAS system diagram.	10
2.3	AggieAir flying wing bungee launch and skid landing.	10
2.4	AggieAir platform system diagram.	11
2.5	Data collected from AggieAir.	12
2.6	AggieAir Fixedwing Platform (Minion), during landing maneuver.	12
2.7	AggieAir Multirotor Platform (Hexarotor), ready for flight.	13
3.1	The big picture of closed-loop cyber-physical system control.	15
3.2	Systems of systems: How many CPSs interact to perform complex tasks.	16
3.3	Several UASs work together to control a wildfire with sensors and retardant.	17
3.4	Water distribution and use as a closed-loop cyber-physical control problem.	18
3.5	Multispectral data collected from an agricultural scene by AggieAir.	19
3.6	A view of the Logan Lagoons from Google Earth.	20
3.7	Algaeland: A UAS Enabled CPS for SEP. A water treatment lagoon as a UAS-enabled CPS to create biofuel from waste water.	21

4.1 An Architecture for Ethical Remote Information Sensing (AERIS) will allow RS data missions in the civil airspace. 24

4.2 The most general representation of an architecture. 25

4.3 From Doyle: An example of a successful layered architecture approach. . . . 26

4.4 An intersection of cyber-physical domains: uncommon and important behavior. 27

4.5 Homogeneous CPU architectures and the human brain adapted from Cory. 30

4.6 Robotic functionality demands many different processes at many different levels of priority and computation. 30

4.7 Evolutional history of civil avionics architectures from "Civil Avionics Systems, 2nd Edition" by Moir, Seabridge, and Jukes. 39

4.8 Tree representation of COncept of OPerationS. 41

4.9 Nested, multiscale UAS-CPS architectures. 45

4.10 General UAS flight control loops. 45

4.11 The Rasmussen and Svedung socio-technical model of system operations, from Leveson, adapted from Rasmussen. 47

4.12 Homogeneous CPU with interface assignments. 50

4.13 The ISaAC flowchart for system estimation, behavior determination, and control adaptation. 51

4.14 An overview of the Paparazzi control scheme from the Paparazzi project. . 55

4.15 An overview of the PixHawk avionics system from the Pixhawk project. . . 56

4.16 The node-based architecture of ROS, from the ROS project documentation. 58

4.17 The DJI Phantom 2.0 rotary UAS from DJI. 59

4.18 Procerus Kestrel 3 autopilot from Procerus' documentation. 60

4.19 Paparazzi UAS control loops (image from Paparazzi), interfaced to the NAS (image from NASA) via DMQ. 64

4.20 Sample code of conduct for sUAS operations from Dr. Paul Voss of Smith College (provided privately). Used with permission. 67

5.1 Flight states for AggieCap sensors. 74

5.2 AggieCap ISaAC data flow block digram. 76

5.3 AggieCap inheritance. 78

5.4 AggieCap example payload block diagram. 81

5.5 Multispectral payload module ready for flight. 82

6.1 Aircraft coordinate systems from Chao et al. 86

6.2 Microstrain GX2 IMU. 95

6.3 Xsens Mti-g IMU from Xsens' documentation. 95

6.4 AggieNav IMU. 96

6.5 Ardu-IMU. 96

6.6 Sparkfun Razor IMU. 97

6.7 Simple complementary filter diagram. 100

6.8 Frequency domain complementary filter. 102

6.9 Bode plots of $G_1(s)$ and $G_2(s)$. 103

6.10 Simulink model of modeless complementary filter. 104

6.11 Response of the inclinometer to the input chirp signal. 105

6.12 Response of the gyro to the input chirp signal. 106

6.13 Variance of the error vs. cutoff frequency (ω_0). 107

6.14 Estimated orientation vs. actual orientation at $\omega_0 = 0.22$ rad/s (optimal frequency). 108

6.15 Squared error for $\omega_0 = 0.22$ rad/s (optimal frequency). 109

6.16 Estimated orientation vs. actual orientation at $\omega_0 = 3$ rad/s (inclinometer dominant). 110

6.17 Squared error for $\omega_0 = 3$ rad/s (inclinometer dominant). 111

6.18 Estimated orientation vs. actual orientation at $\omega_0 = 0.01$ rad/s (gyro dominant). 112

6.19 Squared error for $\omega_0 = 0.01$ rad/s (gyro dominant). 113

6.20 Alternative frequency domain complementary filter. 113

6.21 Closed-loop complementary filter system. 114

6.22 Variance of the error vs. controller gain (P) for SSCF. 114

6.23 Estimated orientation vs. actual orientation with $P = 0.84$ (optimal gain). . 115

6.24 Squared error for $P = 0.84$ (optimal gain). 116

6.25 Simulink model of state space complementary filter. 116

6.26 Variance of the error vs. controller gains (P and I) for SSCF. 117

6.27 Estimated orientation vs. actual orientation with $P = 0.44$ and $I = 0.048$ (optimal PI gains). 118

6.28 Squared error for $P = 0.44$ and $I = 0.048$ (optimal PI gains). 119

6.29 Estimated bias plots for $P = 0.44$ and $I = 0.048$. 120

6.30 Bode plots of $H_{\mathrm{Oust}}(s)$, corresponding to the approximation of a fractional-order integrator of order 0.45 with the Oustaloup method, with solid lines for $G_O1(s)$, dash lines for $G_O2(s)$ and dotted lines for the theoretical Bode plot. 122

6.31 Inclinometer signal with non-Gaussian noise. 124

6.32 Gaussian and non-Gaussian PDFs used in numerical simulations. 125

6.33 Gyro signal with non-Gaussian noise and offset. 126

6.34 Simulink model used for all simulations. 126

6.35 Variance vs. gain: integer controller with Gaussian noise. 128

6.36 Variance vs. gain: integer controller with non-Gaussian noise. 129

6.37 Variance vs. gain: fractional-order controller with non-Gaussian noise and $\alpha = -0.18$ found numerically. 130

7.1 Change in throttle command during indoor hexarotor flight–the red box indicates the beginning of collapse. 135

7.2 Chen and Rincon-Mora's battery model. 135

7.3 Dependency of OCV on SOC. 138

7.4 An example of a multirotor actuator–Av-Roto BLDC motor (without a propeller) and Mystery 40A ESC. 139

7.5 A typical ESC and motor connection scheme. Microcontroller (left), MOSFET driver (middle), MOSFETs and BLDC motor (right), from Allegro. . . 139

7.6 Above: altitude tracking during autonomous outdoor hexarotor flight–the green box indicates increasing oscillations. Below: closed-loop battery voltage. 141

7.7 Battery-based altitude control diagram with battery compensation block. . 144

7.8 3D Model of the motor testbench apparatus. 144

7.9 Experimental setup: testbench, Arduino board and computer. 144

7.10 Zippy 5000mAh 40C 4-cell battery (above); MaxAmps 11000mAh 40C 4-cell LiPo battery (below). 145

7.11 Measured current during battery discharge. 147

7.12 Thrust variations during battery discharge (blue: raw data, red: filtered data). Filtered with Exponential Moving Average (EMA) filter, $\alpha = 0.01$. . 147

7.13 Dependency of thrust on PWM command (Mystery 40A ESC, T-motor MT2814 KV770 actuator and 12×3.8 propeller), blue: measured data, red: linear approximation. 148

7.14 Thrust dependency on SOC (blue: raw data, red: least-square spline approximation, green: thrust at 10% and 90% SOC) 148

7.15 Normalized spline approximation of dependency of thrust on PWM command (green: 10% mark, red: 90% mark). 150

7.16 Inverted nominal thrust (blue) and its piecewise-linear approximation (red). 150

7.17 Laboratory test of the controller with Zippy 5000mAh battery. 151

7.18 Laboratory test of the controller with MaxAmps 11000mAh battery. 152

7.19 Diagram of the developed battery monitoring subsystem. 153

7.20 An assembled battery monitoring system, mounted on a hexarotor platform: ready to fly. 154

7.21 Square wave discharge experiment, MaxAmps 11Ah battery. Top: measured closed loop voltage, middle: measured current, bottom: calculated SOC. . . 155

7.22 Piecewise-linear approximation of OCV-to-SOC dependency for MaxAmps 11Ah battery. 155

7.23 Altitude hold with the AggieAir hexarotor within ±1m using the controller. Blue: estimated altitude, red: altitude setpoint, green: ±1m threshold. . . . 156

7.24 Flight test with the AggieAir hexarotor. Top: total current consumption, middle: battery voltage, bottom: estimated SOC. 156

Acronyms

ADS-B	Automatic Dependent Surveillance-Broadcast
AERIS	Architecture for Ethical Remote Information Sensing
ARD	Alpha-Stable Random Distribution
BLDC	Brushless Direct Current
CCOTS	Consumer Commercial Off-the Shelf
CLV	Closed-Loop Voltage
COA	Certificate of Authority
CONOPS	COncept of OPerationS
COTS	Commercial Off-the-Shelf
CPS	Cyber-Physical System
CSOIS	Center for Self-Organizing and Intelligent Systems
DCM	Direction Cosine Matrix
DIO	Domain of Interest
DJI	Da-Jiang Innovations
DMA	Data Mission Assurance
DMQ	Data Mission Quality
DoF	Degrees of Freedom
EMF	ElectroMotive Force
ESC	Electronic Speed Controller
FAA	Federal Aviation Administration
FMU	Flight Management Unit
GPS	Global Positioning System
I2C	Inter-Integrated Chip Protocol
IMU	Inertial Measurement Unit
INS	Inertial Navigation System
ISaAC	Intelligent Safety and Airworthiness Co-Pilot

LIDAR	LIght Detection And Ranging
LiPo	Lithium-Ion Polymer
LTI	Linear Time Invariant
MAGICC	Multiple Agent Coordination and Control
MEMS	MicroElectroMechanical Systems
MPC	Model Predictive Control
NAS	National Air Space
NDVI	Normalized Difference Vegetation Index
NIR	Near-Visible InfraRed
OCV	Open-Circuit Voltage
PDFs	Probability Distribution Functions
PIC	Pilot in Charge
PID	Position, Integral, Derivative
PRS	Personal Remote Sensing
PWM	Pulse Width Modulation
RADAR	RAdio Detection And Ranging
RFID	Radio Frequency IDentifier
RPM	Revolutions Per Minute
RS	Remote Sensing
SEP	Sustainable Energy Pathways
SIMD	Single Instruction, Multiple Data
SOC	State of Charge
SSCF	State Space Complementary Filter
sUAS	Small Unmanned Aerial System
SWIR	ShortWave InfraRed
TIR	Thermal InfraRed
UAS	Unmanned Aerial System
UAV	Unmanned Aerial Vehicle
USB	Universal Serial Serial Bus
UWRL	Utah Water Research Laboratory

Chapter 1

Introduction

As Unmanned Aerial Systems (UASs) grow in functionality and mature in safety, applications for these versatile platforms will become abundant in the coming years. Civilian applications for UASs are an emerging field–one that has great potential with the possibility of explosive growth as their places in science and industry become secured. Within the U.S., civilian small unmanned aerial system (sUAS) deployments have been exceedingly rare due to the severe restrictions currently in place by the US Federal Aviation Administration (FAA) [1]. This is likely to change in the near future, however. Recently the US Congress issued the FAA Modernization and Reform Act of 2012, calling in Sec. 322 a renewed focus towards advancing the integration of civilian UAS into the National Airspace System (NAS) [2]. However, this integration will not be determined by policy alone; the challenges and opportunities of addressing ethics and the public perception of UASs [3] will forever be a part of aviation and robotics as levels of integration and technology progress. The FAA's UAS access rules for the NAS may eventually be in the familiar form of "file-and-fly," and domestic UAS operations will be commonplace. With the proper certifications and standards in place, government and commercial operations will enable regular UAS use, including integration into large-scale operations.

1.1 Personal Remote Sensing

Remote sensing is simply detection at a distance. Humans have a given set of perceptions granted by a suite of sensors (eyes, ears, nerves, etc.), but with science, through engineering, it is possible to significantly extend these perceptions. While information can come from many sources, the best way to collect specific information is by the use of a sensor–a device specifically constructed for the purpose of data collection. Multiple sensors

may work in concert providing many sources of useful information at once; depending on the information demanded, these sensors can change modes via communication to gather better data as time passes and requirements or environments change. Remote sensing allows us to gather data at a physical distance, including data for which there is no other way, or no safe way, to collect in person. In difficult-to-access or inhospitable environments, remote sensing is the only feasible method for accurate and reliable data collection.

There are two methods of remote sensing: passive (Fig. 1.1) and active (Fig. 1.2). Passive remote sensing is as described above: "waiting" for data where they are expected to be found. Active sensing involves stimulation of the sensed environment to excite or otherwise illuminate the situation in question, not unlike using a flashlight to inspect a noise heard in the dark. Many examples of active remote sensing can be found today: RADAR systems [4], the use of lasers for detection of aerosols in the atmosphere [5], detection of evasive species [6], or detection of land mines via honeybees [7].

Humanity, of course, has been attempting to extend the senses to gain advantages or additional knowledge for as long as tools have existed. Devices such as binoculars allow eyes to see further, and the hearing aid extends the range of an ear. These are examples of

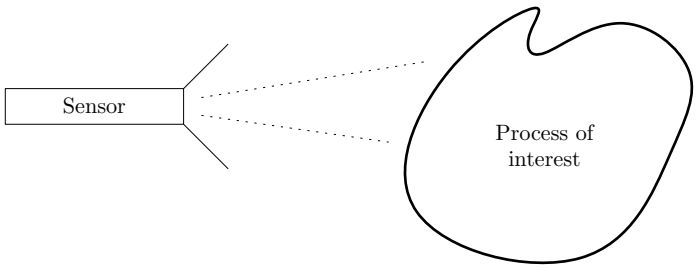


Fig. 1.1: Passive remote sensing.

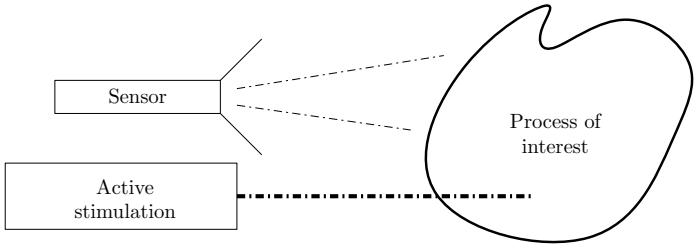


Fig. 1.2: Active remote sensing requires some kind of stimulation to reveal the desired data.

Personal Remote Sensing (PRS): the detection of data useful to a small number of people—perhaps only a single person—on short time-scales and within small budgets.

However, to be convenient, sensing requires aspects of autonomy: traditional systems require a large amount of user interaction for useful function. Flying, pointing, or otherwise controlling a remote sensing system can be time consuming and in some cases demands high levels of skill or training for consistent quality data. An autonomous system can be pre-programmed to patrol for changes in a scene or "follow" a data trail, such as an airborne pollutant, to collect data automatically. In the case of the use of Normalized Difference Vegetation Index (NDVI) in agriculture, a farmer could use a UAS for remotely sensing the plant growth and vigor, vegetation cover, and biomass in her fields, and for determining where water, fertilizer, etc., are needed most. Such a PRS could reduce the costs of the inputs to agricultural production through greater efficiency in application, and increase sustainability, yields, and profits.

The key driver for success with PRS systems is cost. Since traditionally remote sensing required a high cost to provide useful information, this high standard must be met with lower cost to make the technology widely available. Personal remote sensing can only be personal if individuals and small groups can afford to own and operate systems capable of delivering the data they need. Progression of technology (Microelectromechanical (MEMS) devices, embedded processing power, battery power density, etc.) coupled with acceptance of PRS systems by industry, regulators, and legislators, will allow PRS to become commonplace. Low cost sUAS platforms, constructed for a little as 500 USD, have been demonstrated to be usable for remote sensing [8]. Much like the personal computing revolution of the past 40 years, advancements in sUASs will bring remote sensing capabilities down to the personal level, allowing unprecedented levels of knowledge and empowering change for the greater good at smaller scales of operation than ever before.

Personal remote sensing systems are poised to provide the next generation of useful data about our world. While large-scale remote sensing (like large-scale computing) will always have applications, such as weather prediction, PRS allows individuals or small groups

to collect valuable data about situations or processes that concern them specifically. Personal remote sensing is an upcoming and perhaps disruptive technology which will show efficacy when combined with UASs and other autonomous systems. With PRS, new ways of interaction and feedback can be used–with scientific accuracy–to make better, more efficient decisions about what concerns us most.

Although sensing is fundamental, once a decision about managing the world around has been made, action must be taken. Even a non-action can be considered an active response if properly considered, and therefore sensing is naturally coupled with actuation. Sensing and actuation cycles can be coupled in a multitude of configurations, and the use of robotic mobile platforms in addition to traditional static sensors can lead to new scenarios which have never before been conceived or implemented.

1.2 Unmanned Aerial Remote Sensing

While the general public might still view the use of UAS in terms of military applications such as espionage and warfare, domestic UAS use will be significantly more wide-ranging. Civilian UAS applications will range from science exploration to high-valued precision agriculture. Law enforcement, search and rescue operations, intelligence gathering and fire-fighting applications have all been championed as the future of UASs, but these are only the beginning: remote sensing and other technical activities will benefit greatly from the utilization of UASs. In fact, real remote sensing sUASs are not only feasible but have significant advantages.

The rise of sUASs has auspiciously coincided with significant advances in sensors. Currently there exist sensors capable of scientific data collection perfectly suited to fly in small, low-cost unmanned aerial vehicles. As in Fig. 1.3 [9, 10], ShortWave InfraRed (SWIR), Thermal InfraRed (TIR), as well as Near-Visible Infrared (NIR) and visible light cameras can be used to improve agricultural, civil, and scientific data collection in new and exciting ways. Thermal imagery (Fig. 1.4) [11] can be utilized for vegetation monitoring [12], multispectral imagery can be utilized effectively in crop monitoring [13], as well as many wetlands measurement applications [14].

Fig. 1.3: sUAS-sized sensors for water management: thermal and short-wave infrared imagers from Infrared Cameras Inc. and Goodrich.

The collection of data is only one part of the solution to the larger problem of management, which requires some upper-level decision making processes, coupled with a kind of manipulation or actuation. This cycle of real-world management and control is known as a Cyber-Physical System (CPS). These cyber-physical systems are the future of large-scale sensing and actuation. A key factor in CPS success is data that is detailed with enough temporal and spatial resolution for effective operation, and the use of sUASs as mobile sensing platforms will deliver data more frequently and with higher quality than ever before.

In the kingdom of data, quality is the coin of the realm—that is, in any way data can be defined as better, it is. For PRS sUAS-enabled CPSs, this means a quality metric based on frequency of data collection, reliability of data collection flights, as well as the accuracy of the data itself. The ethical concerns of the data collected (e.g. privacy concerns), must also be respected. Safety is a prerequisite for any flight, and the use of any airspace constitutes a degradation of the safety therein. Therefore, safety, morality, and data quality must be balanced together for successful PRS UAS data collection, and for reliable CPS operation enabled by these technologies.

1.3 Dissertation Theme and Contribution

Architecture design is the key to integration of PRS UASs in the national airspace and beyond. Since data is the mission for PRS UASs, a DMQ-based architecture is needed, which will integrate all of the critical factors, metrics, and concerns into a unified architecture for

Fig. 1.4: A 10cm-resolution mosaic of thermal imagery collected by AggieAir from test flight under COA in Cache Junction, UT.

PRS flights. The design of such an architecture will facilitate UAS-societal integration and is critical to the success of unmanned aerial systems for civilian applications.

In the United States, the FAA imposes the system architecture for flight and controls airspace tightly. As soon as 2015, these highly restrictive limits on UAS flight may change, but further rules and architecture need to be implemented for PRS and other civilian UAS use. If DMQ-based metrics are used, airspace can be shared and surrendered safely as more and more "data drones" are added every year.

In this study, an attempt is made to enumerate the major features such an architecture will need to be successful. The Architecture for Ethical Remote Information Sensing (AERIS) is proposed to allow scientific data collection flights while adhering to standards reasonably expected by civilians (privacy, safety, etc.).

Any architecture that fulfills the requirements described in this text will adhere to the proposed AERIS guidelines. Privacy, data quality, and airspace compliance are equally important for sustainable operation of cyber-physical systems enabled by sUAS RS systems.

The architecture of the rules and standards which control the airspace will determine the behaviors of the aircraft as well as the operators therein. Therefore, the architecture is of critical importance to the functionality of unmanned systems for data collection in the civil airspace.

1.4 Dissertation Organization

This thesis presents background about personal remote sensing (PRS) and the AggieAir unmanned aerials system (UAS) platforms in Chapter 2, and the coming technologies of cyber-physical systems (CPSs), with motivating examples of how the two technologies can work together in Chapter 3. Chapter 4 shows how architecture drives the values of a system, which drives the functionality, risks, and rewards of the system dynamics and interactions with the environment around it, as well as introducing AERIS, the architecture for ethical remote information sensing. Enabled by AERIS, Chapter 5 shows detailed system implantation details about a high-dataworthiness payload control structure, emphasizing the idea of data as a mission, and equating data mission assurance partly with payload resilience. Chapter 6 shows how low-cost inertial sensors are a good option for increasing mission quality in PRS UASs, and how fractional calculus can help glean even more information from them. Then, in Chapter 7, advanced battery management techniques are shown to increase the dataworthiness of a vertical takeoff and landing craft by compensating for power loss depending on the lithium-polymer battery power system, adding a state-of-charge-dependent gain, and providing satisfactory control in situations when feedback about actuator thrust or velocity is not available.

1.5 Chapter Summary

This chapter presents an overview of what remote sensing is and how it can be personal. The goal of remote sensing systems is to collect data, and aerial PRSs are no different. However, they are regulated by the FAA or other governmental organizations and must be integrated in the common civil airspace in specific and planned ways with an overall architecture guiding this integration. Also included are dissertation contributions, and the overall organization with overview of the chapters in this dissertation.

Chapter 2

AggieAir Unmanned Aerial Vehicle-Based Sensing Platforms

Current sources of remote sensed aerial imagery (e.g. manned aircraft and satellites) are either expensive, have low spatial and/or temporal resolution, or have long turnover times. These shortcomings make it difficult to use remote sensing effectively for many applications, yet the market for these services is growing tremendously as costs fall. There are many potential applications for low-cost, effective remote sensing platforms including emergency response and disaster management, land management, resource monitoring, and civil engineering. For example, in emergency response, lives and money can be saved with the ability to send out small UASs to autonomously fly grid patterns over areas of interest for search and rescue operations or disaster zones. Additionally, in many watershed science applications it is important to have up-to-date, aerial images of a river multiple times throughout the year at different flow levels. However, due to the price of aerial imagery from conventional sources, researchers performing work on rivers are currently limited in their ability to obtain multiple maps over their area of interest within the timeframes required. In addition, since a river system can be constantly changing, if the processing turnaround time is too long, the imagery used by field crews will not be up-to-date and will be almost impossible to use. Satellite imagery can be especially difficult to work with because the time when a satellite passes overhead and local weather conditions cannot be controlled.

The AggieAir remote sensing sUAS developed by the Center for Self-Organizing and Intelligent Systems (CSOIS) and the Utah Water Research Laboratory (UWRL) at Utah State University, is designed to be the ideal sUAS remote sensing platform in response to a market demand requiring high-quality aerial imagery for agricultural, riparian, wetland, and civil engineering applications. To accomplish these missions, AggieAir is active at all

levels of the science-to-information loop seen in Fig. 2.1 created by Dr. Austin Jensen [15]. AggieAir utilizes both vertical takeoff and landing (VTOL) and fixed-wing platforms.

The AggieAir UAS platform in Fig. 2.2 is an autonomous, multispectral remote sensing platform [16]. AggieAir makes aerial imagery more accessible and convenient by allowing users to choose when and where data is acquired by giving the user full control of the platform. A most important distinguishing feature is the platform's independence of a runway. The fixed-wing aircraft launches using a bungee or pneumatic launcher and glides to the ground for a skid landing seen in Fig. 2.3.

The avionics in AggieAir's low-cost UASs consist of an Inertial Navigation Unit (IMU), which measures acceleration, angular rate and magnetic field in three axes, Attitude Heading and Reference System (AHRS) which combines IMU measurements and provides attitude estimation, and a GPS sensor providing absolute position altimeter (altitude above mean sea level), as well as pressure sensors for precise altitude estimation relative to a certain setpoint. Optionally, a GPS Inertial Navigation System (GPS/INS) combines measurements from all aforementioned sensors—fused together to estimate attitude and position can be used instead of an AHRS [17].

In addition, a data radio transmitter/receiver is necessary for telemetry and remote

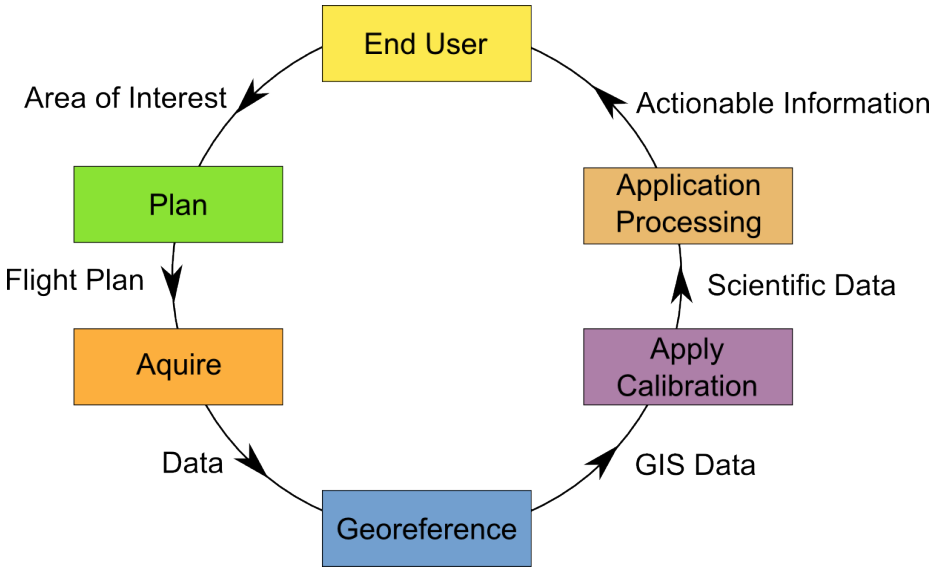


Fig. 2.1: AggieAir: Closing the UAV-based science-to-information loop (from Dr. Jensen).

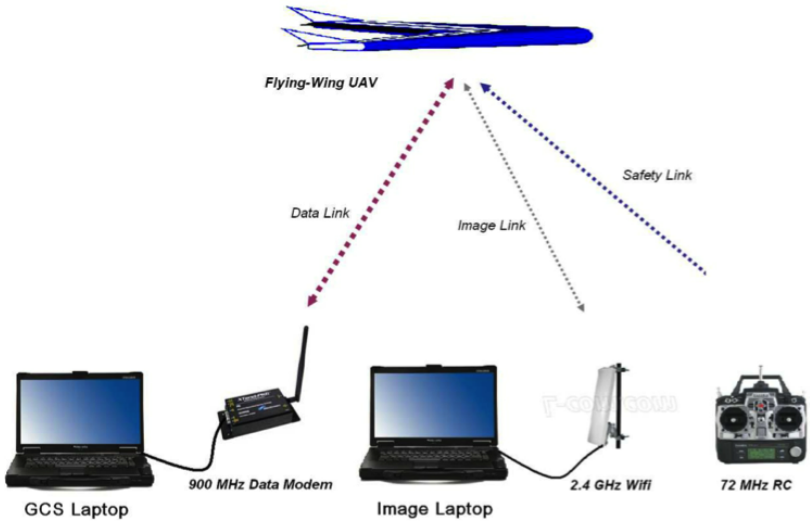


Fig. 2.2: AggieAir sUAS system diagram.

Fig. 2.3: AggieAir flying wing bungee launch and skid landing.

controlled (manual) flight. The autopilot unit runs control loops on-board the UAS and controls the actuators to keep the desired attitude and altitude as well as flight plan execution. Optionally a safety copilot, black box memory, and payload can be added [11]. An overview of AggieAir system is shown in Fig. 2.4.

During flight, data such as images are captured at a predetermined cadence along with data describing the GPS position and orientation of the UAS at time of exposure of each image. After the UAS has completed a data collection mission and successfully landed, camera imagery and flight information are downloaded from the payload system. The images and their corresponding position and orientation information can be quickly approximately georeferenced to generate a map within 30 minutes after flight [18]. However, due to the errors in position and orientation estimation by the craft's sensors, images are not perfectly

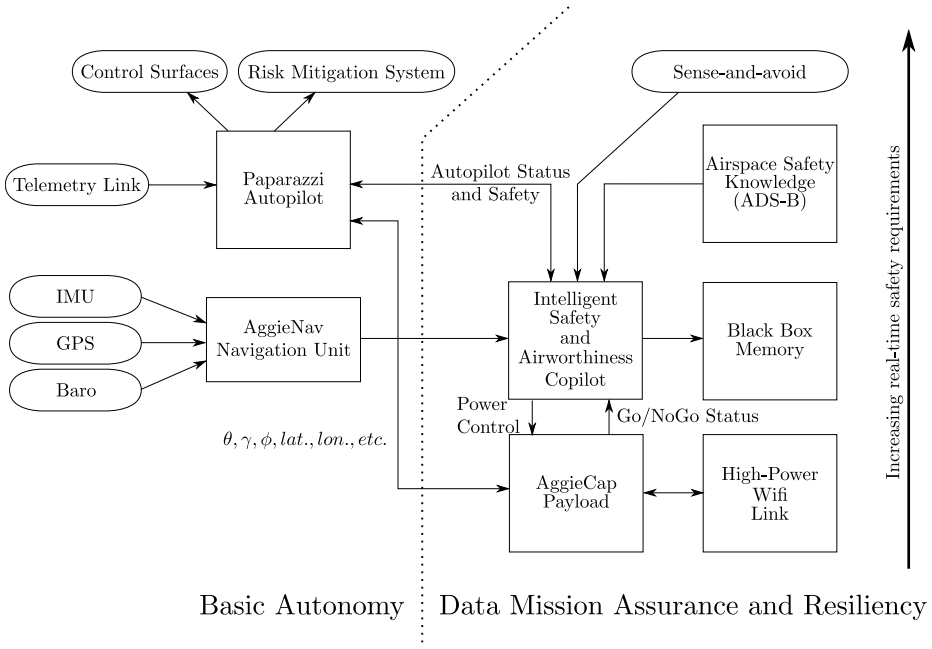
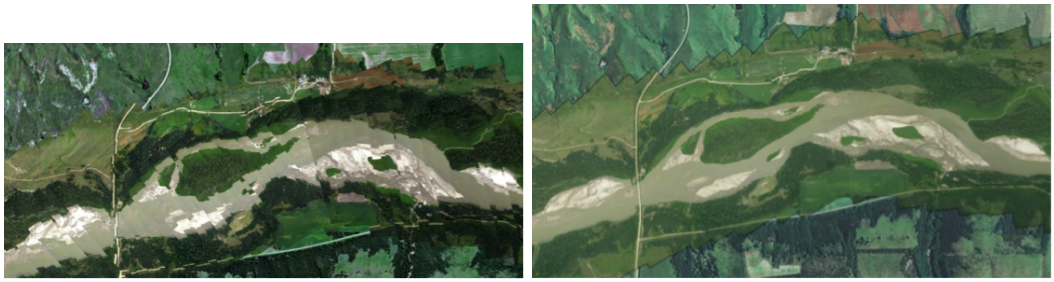


Fig. 2.4: AggieAir platform system diagram.

georeferenced during flight and represent a lower-quality data set. Figure 2.5(a) shows a set of images that were taken over a river and directly georeferenced. Notice that although the images appear approximately in the correct locations, they do not align well with each other. While directly georeferenced images might be sufficient for some applications, more demanding applications require more accurate results. For this EnsoMOSAIC UAV, developed by Mosaic Mill of Finland, is used [19]. EnsoMOSAIC UAV is an image processing software package that reads aerial images captured with sensors such as compact digital cameras and stitches them into seamless orthorectified image mosaics. Figure 2.5(b) shows an orthorectified mosaic made from the imagery displayed in Fig. 2.5(a), (generated by post-processing) was prepared and available only 72 hours after the data collection fight.

2.1 AggieAir Fixed-Wing Unmanned Platform

A current AggieAir fixed-wing platform, Minion is shown in Fig. 2.6. This platform is based on the open-source Paparazzi autopilot [20], and has been in service since 2007. The Minion series of airframes have been designed from the ground up at CSOIS and have been



((a)) Before postprocessing ((b)) After postprocessing

Fig. 2.5: Data collected from AggieAir.

in service flying data missions since 2009.

2.2 AggieAir Rotary-Wing Platform

A current AggieAir hexarotor platform is shown in Fig. 2.7. This VTOL rotary-wing craft is currently under development (Chapter 7), and will be in service by the second half of 2014. Also based on the Paparazzi platform and designed to carry the same payload systems as the fixed-wing aircraft in Sec. 2.1, the AggieAir VTOL is inter-compatible with the existing AggieAir architecture, allowing the rotary-wing system to benefit from all of the testing, verification, and most of the training of the fixed-wing system.

2.3 Chapter Summary

This chapter introduces the AggieAir overall systems and small-UAS airframes. AggieAir includes both fixed- and rotary-wing aircraft, making the overall system applicable

Fig. 2.6: AggieAir Fixedwing Platform (Minion), during landing maneuver.

Fig. 2.7: AggieAir Multirotor Platform (Hexarotor), ready for flight.

to many small and medium-scale unmanned aerial sensing tasks.

AggieAir's distinguishing features include:

1. Low cost;
2. High accuracy (scientific quality);
3. Runway-free mission requirements;
4. Safe, dependable, and easy-to-use.

In the following chapters, more AggieAir system design specifics are described and the overall philosophy is espoused.

Chapter 3

Cyber Physical Systems Enabled by sUAS Remote Sensing

Control and management of real-world processes are challenging, complex tasks. Processes can interact at many spatial and temporal scales, but when dealt with properly some can be controlled just as a simple traditional process. With appropriate choices of sensors and actuators, closed-loop control can be implemented around any controllable process, limited only by the desired outcomes of the controlled system and the complexity of the controller.

This chapter is based on a book chapter found in the upcoming UAS Handbook by Springer [21].

A cyber-physical controller (Fig. 3.1) has a general structure that includes a real-world process (the "plant" in control terminology), estimations of the dynamics of a process under control (an "observer"), and a controller. This allows closed-loop feedback to be instantiated for control of processes that might be outside of traditional definitions of control, such as the moisture in a farm field, or the population of fish at an important aquatic site.

Models of physical processes can be used to help predict the behavior of a system in a particular scenario, or the reaction of a process to a given input. Cyber-physical control is enabled by modeling (usually digitally) a target process and utilizing that model to make decisions and exert control over the process by way of an actuator or a set of actuators. Since computer modeling and control of complex processes naturally require higher computational resources for higher performance, a computational cloud may be employed, distributing modeling tasks spatially and increasing reliability. Using advanced techniques such as fractional calculus [22], sophisticated models of physical processes can be constructed with proper boundary conditions, and, when refined by quality data, can reliably and robustly controlled in real time.

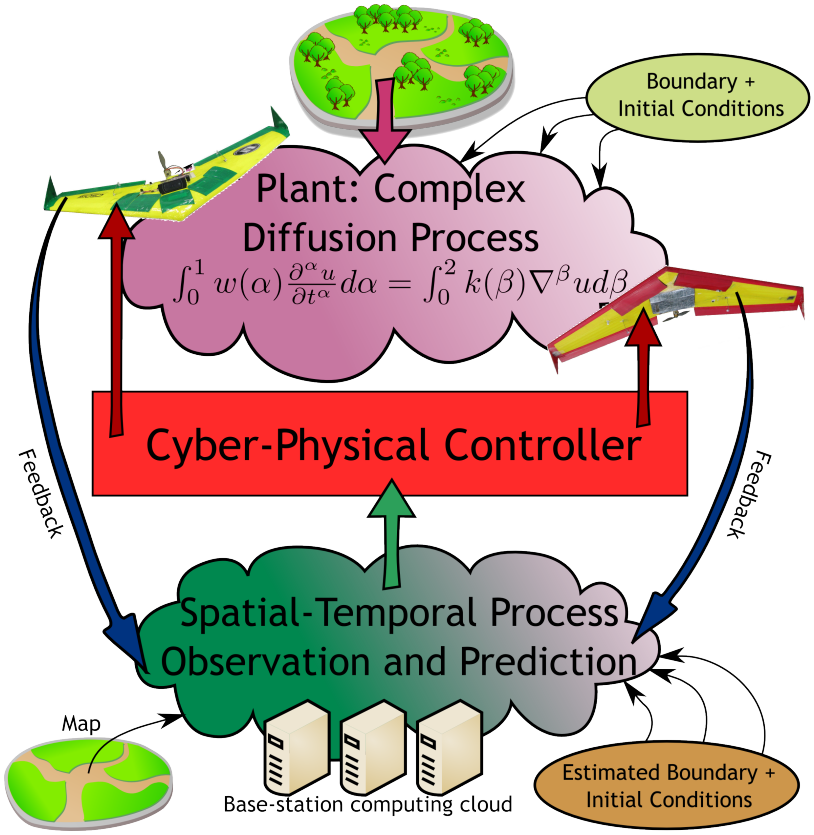


Fig. 3.1: The big picture of closed-loop cyber-physical system control.

Control loops cannot be closed without feedback data. Due to the nature of cyber-physical control, the process under control may be multivariate, and therefore Cyber-Physical Controllers (CPCs) potentially have very broad requirements for sensing. Since in many closed-loop systems better sensing equates to better control, providing CPCs with the most useful, or "best," data is of primary importance. This is one requirement UASs can effectively fulfill in CPSs: they can generate data with high spatial and temporal resolution about complex processes, allowing for CPC loops to operate on new and heretofore difficult-to-control plants.

UASs can also act as actuators in closed-loop scenarios. Chemicals, such as herbicides, can be applied with high precision to areas determined to be in need, or a fire-fighting agent can be precisely dropped in advance of a fire's predicted path. A heterogeneous configuration of fixed- and rotary-wing aircraft in teaming scenarios can be beneficial as well; fixed-wing

flying sensors can determine areas of interest, and rotary-wing craft can take more specific, detailed data or drop payloads such as ground sensor pods or radio relays for extending network communication. In addition, ground or even underwater vehicles can act as part of this team, with robotic systems of many different kinds working together to become a cohesive cyber-physical system. Networked systems, or "systems of systems" will pervade the physical world in the coming years. These networks are themselves CPSs, and must be engineered with the goal of long-term functionality, integration, and inter-operability. CPSs of many temporal and spatial scales will eventually be integrated, and as in Fig. 3.2, these systems will work together for added functionality and overall system resilience [23]. This chapter serves both as an introduction of CPS concepts as well as a high-level overview of example research topics for possibilities for CPS applications enabled by UAS-based PRS.

3.1 CPS Enabled by PRS: Wildfire Tracking and Fighting

Wildfires are common in much of the Western US. Many brave men and women give their lives to fight these fires and many millions of dollars are spent controlling the spread of fire to protect the lives and property of the public.

Figure 3.3 shows a CPS set up to control a wildfire. The UASs pictured on the left include wind measurement sensors, as well as thermal mapping sensors. While the UASs

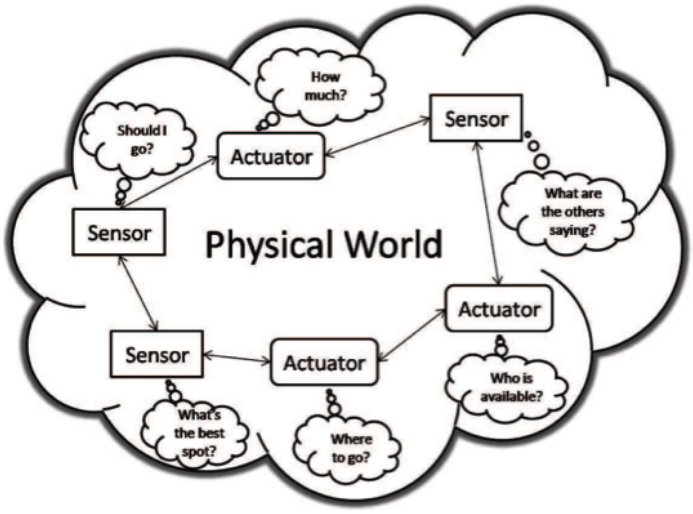


Fig. 3.2: Systems of systems: How many CPSs interact to perform complex tasks.

provide real-time data about the fire, a base station computes the most probable path of the fire, and automatically dispatches a larger UAS (to the right in Fig. 3.3) with fire retardant such as water, etc. to preemptively lessen the fire's destructive power by dropping payload in the most useful locations.

Many solutions to the problem of estimating a fire's location have been proposed [24–27], and this problem is now well studied. To close a CPS loop around this problem, both sensors and actuators must be employed, and therefore predictions must be made of the fire's probable path. Fire path estimation is itself a research topic [28, 29]; these frameworks can be integrated as needed for a particular application scenario as needed. As this prediction is produced and improved, the estimate of the fire's path can also be used to warn the surrounding populace about the impending arrival of the fire automatically, allowing the residents the most possible time to evacuate before the fire arrives.

3.2 CPS Enabled by PRS: Real-time Irrigation Control

Aerial remote sensing can be used to better regulate and control water use by optimally distributing irrigation supplies on the basis of predicted water demand.

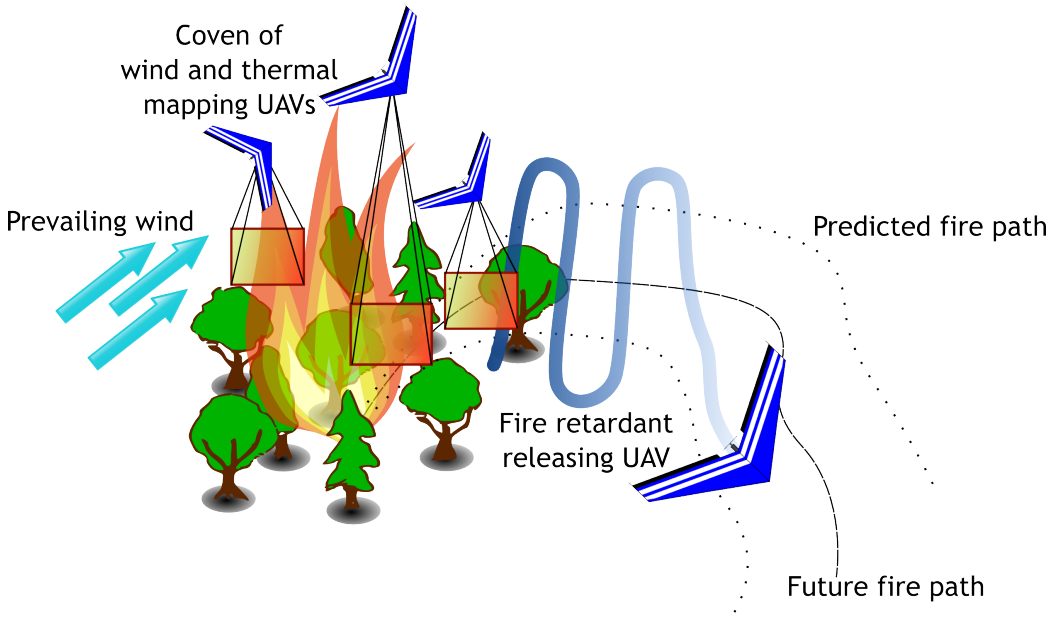


Fig. 3.3: Several UASs work together to control a wildfire with sensors and retardant.

As seen in Fig. 3.4, a water domain of interest (DOI) is depicted as a closed-loop cyber-physical control system. Starting with weather and climate, the water becomes available. Combined with water rights, an optimal irrigation policy can be established [30] that will allow the gates and flumes which control water flow to direct the water to its best use. While some water is lost to evaporation and leakage or seepage, most will reach the geo-domain of interest, allowing use of the water.

During an example use of the water (a farmer's field), traditional ground sensor pods will determine the moisture level in the application areas, as well as UAS-borne sensors flying above the field. Aerial imagery collected in the visible, near-infrared, and thermal bands can be combined to determine the water content of the vegetation and the surface soils in the scene [31], and data fusion can combine the ground and aerial sensor data into the controller, thereby closing the water loop on the particular field in question.

Use of multispectral sensors can provide deep insight into moisture content as seen in Fig. 3.5. Thermal infrared, shortwave infrared, and near-visible infrared, as well as visible

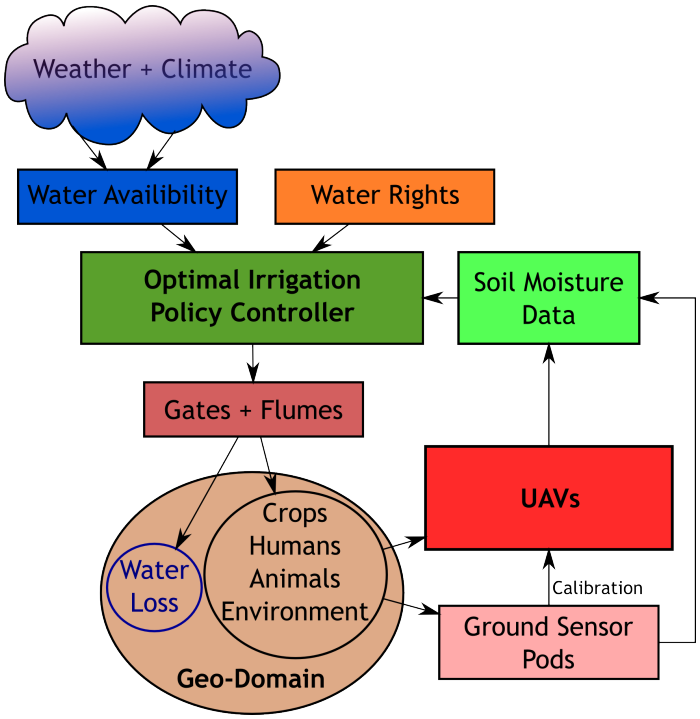


Fig. 3.4: Water distribution and use as a closed-loop cyber-physical control problem.

light sensors are needed to complete the picture and provide detailed moisture maps, allowing more data about the distribution of water and enabling greater control or management.

3.3 CPS Enabled by PRS: Algal Bloom Tracking for Alternative Energy

According to the NSF, "The United States faces a critical challenge to transform our current fossil fuel based energy economy to a stable and sustainable energy economy" [32]. Sustainable Energy Pathways, SEP, is a new research program by the NSF, to encourage and "support interdisciplinary efforts by teams of researchers to address the challenges of developing efficient pathways towards a sustainable energy future. The overarching theme of the solicitation is "sustainability" in all of its facets" [33].

((a)) Before flood (visible and NIR)

((b)) During flood (visible and NIR)

Fig. 3.5: Multispectral data collected from an agricultural scene by AggieAir.

In one example of sustainability, algae grows in wastewater lagoons (such as Logan, Utah City wastewater Lagoons, shown in Fig. 3.6 from Google Earth [34]) and feeds on nutrients such as phosphorus from detergents, etc. These algae can be processed and transformed into energy: both biofuels and methane can be produced from algae, allowing small communities to transform local waste into power. Once the algae has been harvested, it is consumed in a bioreactor such as a microbial fuel cell [35].

The algae harvesting cycle:

1. Algae grows in a controlled environment (lagoon) fed by waste water;
2. Algae "blooms," or rapidly increases its biomass;
3. When sufficient biomass is available, the algae is harvested, providing raw materials for energy production;
4. The algae lagoon begins growing another batch of algae.

To maximize biomass in Step 3 above, feedback is needed to determine when to harvest the algae to maximize production and minimize production times. This feedback can be provided by a UAS in a PRS configuration, flying at an appropriate frequency to estimate and predict the peak algae biomass.

One perspective configuration for using a PRS UAS as a CPS to establish a SEP is seen above in Fig. 3.7. Stations like this one allow production of algae-based power by drawing

Fig. 3.6: A view of the Logan Lagoons from Google Earth.

on waste water from a local community, keeping the energy pathway short, and allowing the highest efficiency. When coupled with other sustainable energy sources such as solar power, the efficiency and autonomy (i.e., stand-alone, off-grid dependability) is higher, allowing for better localization of energy resources, avoiding loss of power transmission line loss, leading to higher overall efficiencies.

While algae grow in the phosphorus-rich wastewater of the lagoon, the algae's natural predator, Daphnia present a threat to the water treatment and energy production processes. When Daphnia are present in the water, the algae are only able to decrease the phosphorus by 1/40th the amount if they were alone [36]. This presents a compelling problem: harvest the algae before the Daphnia do, when the algae are at peak saturation.

UASs are used to survey the algae and detect the optimum harvesting time for the algae crop. Once this is determined, the algae are harvested by some automated means, and the reactor uses the algae as fuel to produce power. The solar grid shown is mainly for operation of the algae farm; however, if there is a power surplus it can be directed out as

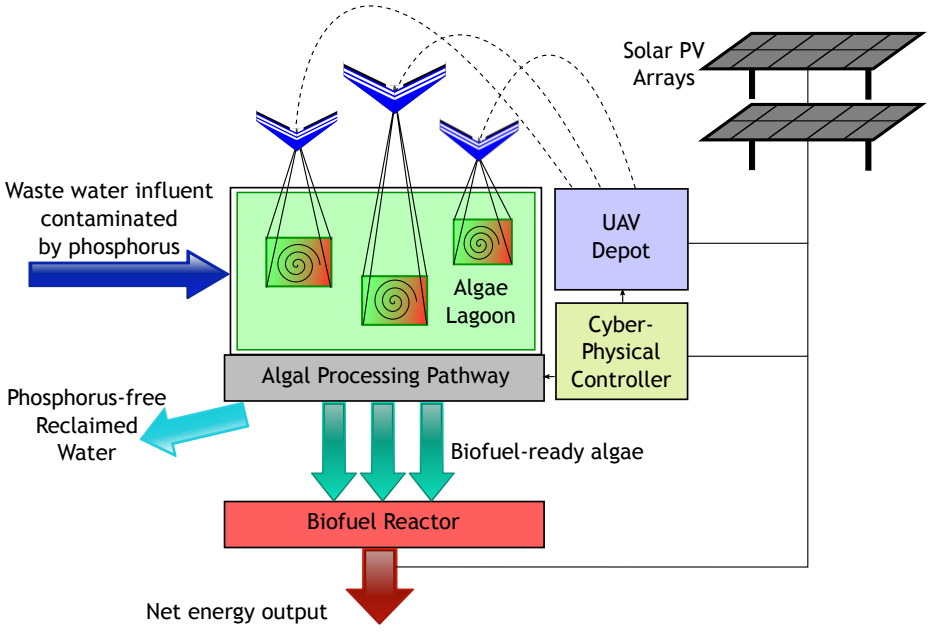


Fig. 3.7: Algaeland: A UAS Enabled CPS for SEP. A water treatment lagoon as a UAS-enabled CPS to create biofuel from waste water.

product energy.

As seen in Fig. 3.6, the Logan Utah Lagoons support eight local districts and, at 460 acres, are ideally sized for small-UAS data collection. Relative to traditional static (i.e., non-mobile) pod sensors embedded in the ponds themselves, UASs cost less to install and operate. This is a vision of cyber-physical systems' promise: a self-sustaining facility with a net power gain.

3.4 Chapter Summary

This chapter introduces cyber-physical systems and the roles that sUASs can play in coming years. UAS applications such as search and rescue, precision agriculture, etc., are now well known, and more applications appear nearly every day. Small UASs in particular are expected to play a major role in domestic UAS operations. Versatile, inexpensive, and easily maintainable, these smaller systems can be utilized effectively in many applications where fast responses are needed with precision information. Further applications of these systems will lead to their utilization as part of larger cyber-physical systems as sensors or even actuators, providing integration and control on larger scales and higher complexities.

Cyber-physical systems are the future of solving real-world problems such as water management and alternative energy production. By using sUASs as sensors, better data can be collected and used to make informed control decisions, transforming difficult, complex, or abstract problems into closed-loop systems that can be analyzed and controlled.

Within the next 15 years, the development of unmanned technology will enable UASs to be mobile actuators, actively exerting control in optimal locations for precision in both sensing and actuation, and cognitive control systems for large-scale complex systems that were previously uncontrollable or impractical to control. Management of crisis situations such as food and water shortages, energy production, floods, and nuclear disasters can be assisted by sUASs. Difficult problems can be solved with cyber-physical systems. This chapter serves as a motivator to position small, inexpensive UASs as flying sensors and actuators within the closed loops of cyber-physical control.

Chapter 4

Architectures for Ethical Remote Information Sensing

As Unmanned Aerial Systems grow in functionality and utility and mature in safety, applications for these versatile platforms will rapidly become abundant in the coming years. Civilian applications for UASs are an emerging field–one that has great potential and the possibility of explosive growth as their places in science and industry become secured. The US Congress is aware of these possibilities and has given the Federal Aviation Administration (FAA) a deadline of 2015 for the creation of rules allowing the inclusion and integration of UASs into the greater US National Airspace System (NAS). This integration will not be determined by policy alone; the challenges and opportunities of addressing ethics and the public perception and acceptance of UASs will forever be a part of aviation and robotics as levels of integration and technology progress [3]. It is hoped that the FAAs UAS access rules for the NAS will eventually be in the familiar form of "file-and-fly" (that is, a licensure and law-based system with insurance) and domestic UAS operations will be commonplace. With the proper certifications and standards, government and commercial operations will include regular UAS use, including integration into large-scale operations such as Cyber-Physical Systems discussed in Chapter 3.

Unmanned aerial systems for personal remote sensing are, like any unmanned system, defined by their missions. Personal remote sensing, however is focused on data collection, and therefore a PRS mission can be defined directly as data. While quality as a concept is notoriously difficult to define [37], Data Mission Quality (DMQ) is a measure of the relative quality of the data mission, and can be used as a yardstick to compare multiple missions, different UASs, and different payloads.

Mission assurance, in a PRS sense, is Data Mission Assurance (DMA). Data Mission Assurance is the upholding of a minimum DMQ. This means, for example, without a fully

functional payload, there is little reason to fly and make use of the airspace allocated to the sUAS.

To achieve data mission assurance in current AggieAir payloads, modular approaches to PRS UAS systems design are taken, from development to testing, allowing overall performance and system faults to be quickly and accurately diagnosed [11]. This concept must be extended to the context of the greater airspace and expressed in a cyber-physical systems context to enable PRS systems to interact with manned aircraft and other UASs while in operation in a safe and reliable way, while delivering DMQ overall.

Architectures must be implemented to allow UAS use while preserving order and respecting many different kinds boundaries (safety-related as well as ethical). AERIS is proposed as a set of guidelines as well as examples to allow scientific data collection flights while adhering to standards reasonably expected by civilians (privacy, safety, etc.), as illustrated in Fig. 4.1.

All architectures adhere to the same general structure. The purpose of an architecture is to induce some kind of desired order from chaos–that is, to set "rules" so the possible states of a system are set. In general, an architecture can be viewed as in Fig. 4.2. This arrangement shows the chaos (above) and the desired behavior for resources to manage below.

Once cyber-physical systems themselves have been accepted and implemented, the important work moves to the intersection of cyber-physical domains. In the real world,

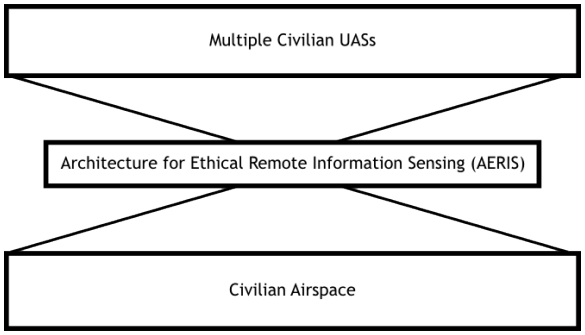


Fig. 4.1: An Architecture for Ethical Remote Information Sensing (AERIS) will allow RS data missions in the civil airspace.

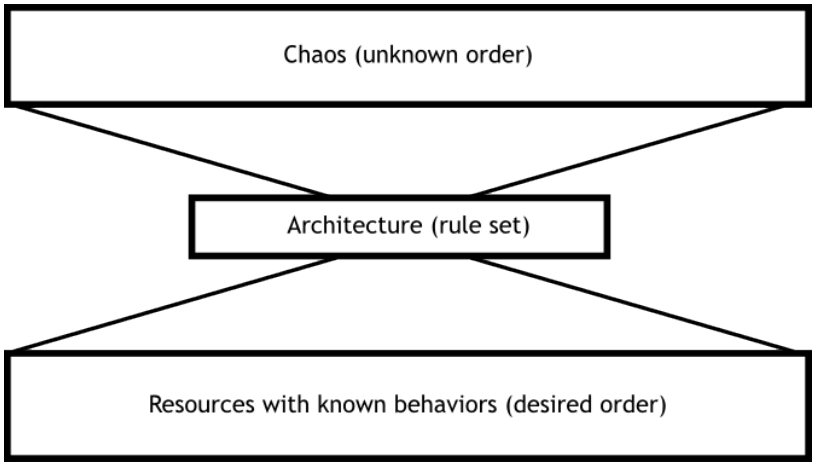


Fig. 4.2: The most general representation of an architecture.

decisions can be complex [38]; all possible options must be considered for the guaranteed best option to be chosen. Uncertainty naturally scales with complexity [39], which can be described by power-law statistics [40], and in turn may be modeled by fractional calculus [41]. The concept of "scale-richness" is also important: the mixing of different exponents in power-law distributions, for example in biological networks [42].

This scale-free aspect of nature can be categorized as a form of chaos; that is the inputs or requests from the environment or from other CPSs to a given CPS cannot be predicted completely. Architecture that acts as a gatekeeper limits actions on the plant (the critical minimums for preservation goals in a given case) will allow outside interactions–new or innovative behaviors–which are not originally designed or predicted, yet are still compliant with the "rules of the game" due to proper architecture design. These new behaviors can be considered groundbreaking or innovative, allowing progress, improvement and research to continue without violation of (and possibly permanent damage to) the system being protected.

Proper segmentation of the layers in system architecture is of critical importance (as espoused by Doyle [43]), and much can be learned about robustness to complexity from systems occurring in nature. Lessons can also be learned from poor layer separation in systems not designed for robustness such as email (originally, email was designed without security features [44]). Although email has become a critical infrastructure element, for

years malicious emails have been able to commandeer servers and workstations. Improper segmentation of data and processes created this situation almost by design–the legacy of poor architecture design. Other examples include insecure computer operating systems such as Microsoft Windows 95, or conflict-of-interest situations such as bribery in government.

Prof. John Doyle has written about these issues and has encapsulated this idea as a "bow-tie" as in Fig. 4.3 from Slide 46 of Doyle's CDS212 lecture slides [43]. Since failures and uncertainties lie at the edges of domains, an operating system or architecture allows a variety of different processes to share data and communicate, provided they adhere to a common set of standards. Proper layering and engineering of security of architectures is critical to their stability, robustness, and resilience [45].

As shown in this dissertation, AERIS-compliant architectures enable functionality at many levels for sUASs.

- Control and behaviors (Sec. 4.6),
- Payload control and data mission performance metrics (Chapter 5),
- Inertial measurement and navigation (Chapter 6),
- Battery state estimation (Chapter 7),

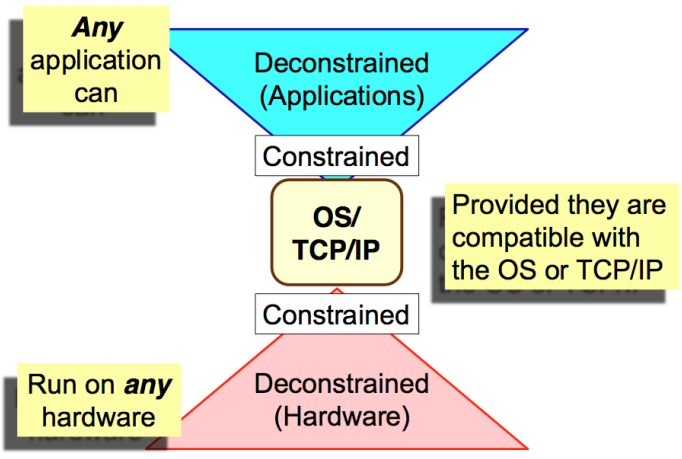


Fig. 4.3: From Doyle: An example of a successful layered architecture approach.

- Any other behavior or system feature as required or innovated.

4.1 Cyber-Physical Philosophy

As an example: in the near future, there may exist a fully automated air traffic system (i.e., mixed human passengers and air cargo flights, coordinated by cyber-physical algorithms). A fully automated ground transportation system (self-driving cars in a networked platooning configuration) is equally likely in this time frame. In the event of a life-threatening airborne emergency such as an engine failure, how would these two highly complex systems coordinate a landing of the stricken aircraft onto the busy interstate highway?

Figure 4.4 shows an intersection of CPS domains, such as the above scenario. The general solution to problems like the above is architecture: standardizations that allow communication to take place and interactions to be quantified. Once the contact is defined, out-of-band or "cross-layer" [46] communication is no longer a possibility and both systems are more robust.

The Internet and data networks are examples areas this kind of thinking is needed for security and robustness. Allen Turing is referred to as the father of layered computer system design [47], and in modern times Prof. John Doyle has written about these issues and has encapsulated this idea as a "bow-tie" as in Fig. 4.3. Since failures and uncertainties lie at the edges of domains, an operating system allows a variety of different processes to

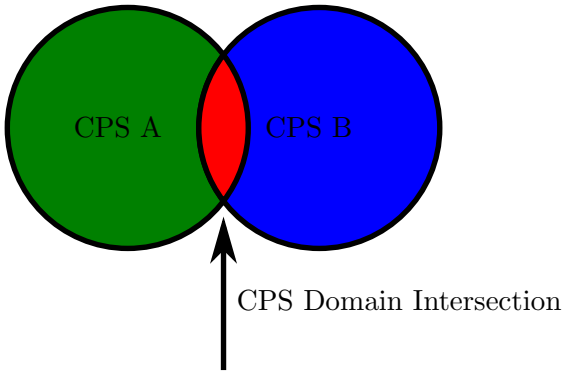


Fig. 4.4: An intersection of cyber-physical domains: uncommon and important behavior.

share data and communicate, provided they adhere to a common set of standards.

Doyle also points out Turing's choice of base-2 as an architecture has enabled digital calculus, and computing as we know it in general [45]. C# (or Ada, a more extreme case) are more heavily typed and more popular for robust applications vs. assembly language, which has practically no layering at all by design. Frameworks and architecture give systems robustness through standards such as in autonomic computing [48].

Other examples of systems designed for complexity management include legal systems and government [49], any number of biological systems [50], and psychological interaction models [51]. Proper architecture design will allow both robustness and flexibility while minimizing long-term fragility [52]. Once again, proper architecture is the only way forward for DMQ-based sUAS remote sensing.

4.2 Biologically-Layered Robotic Scheduling and Functionality

As in Doyle [43], biology can teach important lessons about segmentation and layering. Robotic goals are complex, and their general formulation is not a trivial task. Additionally, there is a constant drive to lower robotic cost, while increasing demand for performance. Consumer electronic hardware has similar attributes, however, due to the basic physical nature of robotics, reliability is required beyond most consumer applications; in some cases aerospace-grade reliability is required for dangerous or expensive robotic missions. To achieve this reliability, the system should be resilient to software failures and faults, both from within (software bugs) and from without (hardware failures, intentional hacking attempts, etc.).

Take, for example, multicore "application processor" level computing platforms. These platforms are designed for high-level consumer interactive hardware such as cellular phones and GPS navigation systems, DVD/Blu-ray players, etc., and have a considerable amount of processing power and features due to a high level of integration. These processors are ideal for low-cost robotic platforms due to their availability, and low power and size requirements, and will continue to become more integrated as time passes. Robotics can benefit from this trend: processors with multiple cores can be used to create robotic systems with higher

reliability beyond a traditional unicore system [53].

4.2.1 The Triune Brain

In evolutionary biology, neurologist Paul MacLean's triune brain theory [54] presents a segmented picture of the human brain. The theory claims that as the brain evolved, the different layers developed on top of one another. In Fig. 4.5 (Adapted from Cory [54]), the triune brain is compared to the heterogeneous computing platform implementation described in this chapter with some remarkable similarities. The three sections of the human brain according to the theory are:

1. **The Low-Level Lizard.** First, the most basic processing is done by the Lizard or Reptilian brain. This brain is physically contained in the brain stem, cerebellum, and spinal cord, and can heavily influence commands given to the body from the higher brains, effectively blinding the higher brains in extreme situations, creating phenomena such as the heartbeat and respiratory system. The Lizard brain has no long-term sense of time.

2. **The Emotional Limbic.** The Limbic system has also been called the old mammalian brain, and is responsible for higher-level basic behavior, which handles fighting, fleeing, feeding, and fornication [55]. Emotions are the language of the Limbic system; the act of interpreting a vast amount of input data from the senses and producing an overall "feeling" about a situation is the realm of the mammalian brain. The part of the brain is found in more evolved mammal animals, and occurred later in the evolution process. The Emotional brain also has no sense of long-term time.

3. **The High-Minded Rational.** At the top level, and most recently evolved, is the neo-cortex, or "rational" brain. This part of the brain makes high-level decisions and is responsible for cognition in the traditional sense. Physically, this is the part of the brain divided into left and right (creative and logical), and consumes two thirds of the brain mass in a normal Human brain. The rational brain makes key decisions about

"good" and "evil;" it allows individuals to have preferences and think abstractly in long-term timescales.

4.2.2 Robotic Processing Requirements

Robotic requirements are application dependent. However, they can be grouped into general categories relating to common robotic system components as seen in Fig. 4.6.

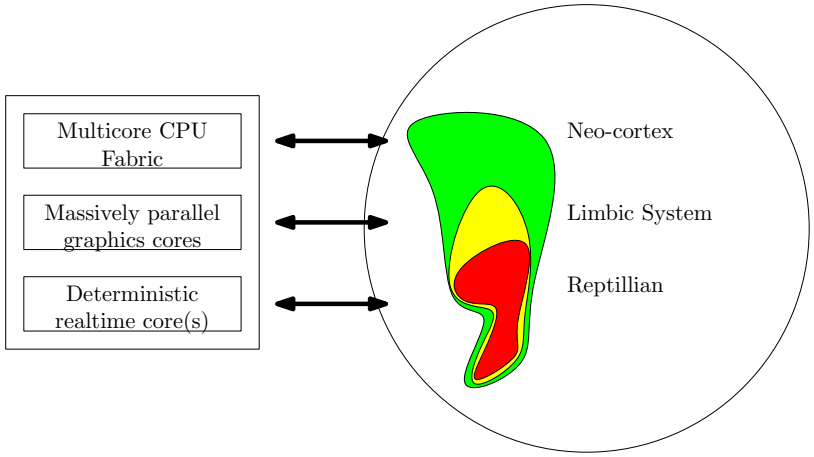


Fig. 4.5: Homogeneous CPU architectures and the human brain adapted from Cory.

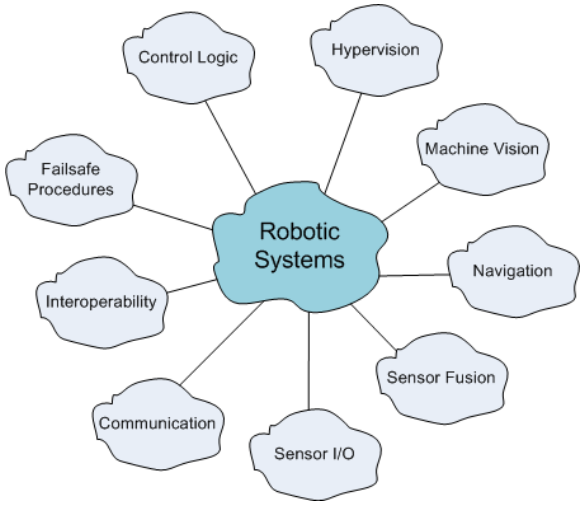


Fig. 4.6: Robotic functionality demands many different processes at many different levels of priority and computation.

Deterministic Control:

All robots require actuators for movement, whether it be propellers in water or air, wheels or tracks for ground movement, or manipulators for moving parts, etc. These require deterministic processing for feedback control, on the time-constants of the actuators. In the triune brain, these "fast" closed-loop behaviors are handled at low levels which might bypass the higher brains. If a nerve senses something resembling pain, a limb can be withdrawn from the source very quickly, leaving the owner with little choice if the action is not anticipated. The spinal cord and cerebellum provide near-deterministic functionality for the animal's body. Similarly, the real-time control and signaling a Digital Signal Processor provides can close the control loops required for locomotion or manipulation at high rates, allowing for other systems to provide setpoints for control without involvement in the specific control mathematics.

Massively Parallel Data Processing:

Processing sensor data is crucial to any robotic system. Although some systems are simple and need relatively little sensor processing, modern robotic systems can be exposed to gigabytes of sensor data from laser range systems, computer vision systems, raw GPS data, etc. Much like the millions of nerves and sensing systems in the animal body, sensors could easily overwhelm a processor with information. However, in the biological Limbic system, experiences and knowledge are used to process data from eyes, ears, and other complex sensing systems very rapidly to produce emotions and other signals. With the modern computing architectures' parallel graphics processing cores, properly formulated parallel algorithms can process gigabytes of sensor data very quickly for science, simultaneous localization and mapping (SLAM), or other important tasks.

Multicore Decision Making:

The most important task given to autonomous robots is decision making. Once data is summarized, actions must be taken. Other tasks are also running on an autonomous system: peripheral management, logging, safety monitoring, etc. The biological neo-cortex

makes logical decisions, acting in the owner's best interest and keeping a sense of time. While these logical processes can indeed also be parallel, they need not require the vast amounts of computational resources that data processing does.

Machine Vision:

Machine vision (MV) is the application of making computers understand what is perceived visually. Unlike human vision, machine vision systems perform more narrowly defined tasks, and are mostly relied on digital imaging systems and computing devices that exam individual pixels of the images.

Image processing techniques are commonly used in machine vision to develop the ability of deriving visual results with the assistance of knowledge bases and features. However, the machine vision algorithms usually require high computational effort. Therefore, high computational throughput hardware are commonly used in order to process the images in real-time.

Machine vision allows robotic systems to see its surroundings, and is widely used to aid the control logic of the robotic systems. A lot of previous works have integrated image classification and real-time object detection into robotics vision [56–58]. A 3D object tracing system was implemented for humanoid to allow complex locomotion such as stair climbing [59].

To meet the need of high computational efforts, a lot of machine vision implementations for robotics system use graphics processing unit (GPU) to accelerate the computation. By its nature, the computation required by image processing adapts high proportion of data-level parallelism. These algorithms are able to fully exploit the high throughput of SIMD-based computing architectures, such as GPUs. The increasing demanding of computational throughput for machine vision is likely to be solved with the thriving of general purpose GPU computing. Currently, many robotics systems have been integrated with GPUs to allow on-board processing for complex algorithms [60].

Navigation:

Navigation for robotics are widely deployed in mobile robot systems. The most commonly used navigation scheme is based on pre-defined waypoints and waypoint following. In this scenario, the robotic system is aware of its current global position to a certain extent. The navigation system conducts motion planning by comparing the robot's current position with the next waypoint position.

A more complex navigation technique is based on real-time information of the robot's surroundings. A very common case is target tracking, where the robotic systems are designed to follow certain objects. The paths of such objects might be arbitrary, therefore the robots need to be equipped with certain technology to detect the objects. The position information of the objects is fed back to the navigation system to derive the subsequent movement of the robot. Advanced robotics navigation requires combination of technologies such as machine vision, 3D modeling, path finding, and decision making [58, 61–65].

Due to the complexity and ambiguity of the problems, navigation algorithms commonly rely on high-level processing architectures such as general purpose processors. In most cases, navigation does not require absolute real-time processing and certain lag in processing time is usually acceptable. The complex logic of navigation systems is better implemented on more powerful general purpose processors that are separated from the low-level real-time controllers or DSPs, so that complicated decision making algorithm does not interfere with the real-time control logic of the robots. This segmentation is in line with Turing and Doyle's layered architectures.

Sensor Fusion:

Sensor fusion is a class of techniques of combining sensory data from disparate sources to derive more accurate or dependable information (such as shown in Chapter 6). Due to corruption from noises and disturbances, data acquired by different sensors have heterogeneous attributes and components. Sensor fusion is able to extract consensus information from distributed sensors, or derive new information based on independent measurements in a recursive sense.

Sensor fusion is important for modern robotics systems [66–68]. Using stereo image fusion technology, robotic vision sensor is able to operate similarly as human vision, and its ability to detect and recognize objects can be significantly improved. Image fusion also allows robotic vision to abstract and identify information from multispectral or hyperspectral imagery. For mobile robots, sensor fusion allows several complementary and redundant sensors to assist the robot accurately locating itself within complex environments.

Communication:

Communications between mobile robots and central control stations needs to be handled on higher level computing architecture, where user can deploy WiFi, Bluetooth, or other communication devices with the support of operating systems.

This level of communications creates a channel for information exchanging amongst robot agents. It also provides users with an interface for remote monitoring and control of each robot agent. Each node within the network can be configured with equivalent or master/slave relationship, depending on the data flow direction.

Formation Control:

Based on the communication link among multi-agent robotics system, we can control the movement of a group of robots, or a networked swarm system. Numerous works have done to emphasize the robustness of formation control due to changes in network topology and information consensus in networks [69].

The framework of the formation control system is largely based on matrix theory and algebraic graph theory. In practice, the algorithm combines both control logic and mathematically intensive computation. For real-time applications, formation control algorithms can be handled on-board by high-level computing architecture, combined with general purpose processor with general purpose GPU accelerating the parallel portion of the computation.

Interoperability:

The interoperability is the ability of diverse robotic systems to communicate, organize and operate together. Projects such as Joint Architecture for Unmanned Systems (JAUS) [70] aim at supporting interoperability of robotics systems by providing a standard for messaging among robots or unmanned systems. To achieve such interoperability requires proper support of JAUS software platform and operating system, and therefore it is supported by on-board hardware and software drivers.

Failsafe Procedures:

Due to the increasing complexity, it is inevitable for robotic systems to experience malfunctioning or failure. A well-designed robotic system should consider all possible causes of malfunction and means to minimize the cost of a possible failure. This is especially the case for unmanned aerial systems, for which any failure might lead to disastrous results. Therefore, in order to improve the airworthiness for such systems, failsafe procedures are indispensable in their designs. A robotics system can deploy both hardware and software level of failsafe procedures. On the hardware level, such as a power failure, catastrophic accidents can be prevented by designing certain mechanisms, such as projecting parachute for aerial systems. Software mitigations such as the one presented in Chapter 5 can also be used.

Control Logic:

Control logic of robotics systems are the basic mechanism that drives the movements. The control logic can be handled by the DSP or microcontrollers that are placed at the lowest level of the processing hierarchy, where they have direct real-time access to sensor and actuator I/O.

Hypervision:

A hypervisor provides oversight to the computing architecture by checking the virtual machine states of various processes [71]. When a state is detected to be out of sequence,

the corresponding state machine can be reset or other mitigation techniques employed.

Sensor I/O:

On the lower system level, sensor I/O is connected to the micro-controller or DSPs which directly have access to input and output signals from sensors and actuators. On the higher processing level, sensor I/O can be abstracted from software drivers on operating systems that controls certain hardware.

4.2.3 Ethical Behavior as Implementation of Morals: Robots vs. Weapons

The study of ethics is a broad concept. However, unmanned ethics is becoming a pertinent topic and is absolutely critical to the operational success of civil RS UASs due to the significant ethical considerations of the operating environment and in the management of a RS UAS. Civil UAS missions can be defined as robotic tasks. In this dissertation, "morals" are defined as the "rulebook for doing or being good," where "ethics" are defined as the behavioral implementation of morals.

David Wright, a managing partner in Trilateral Research and Consulting [72] proposes a Privacy Impact Assessment (PIA) and give these unassailable ethical principles which must be adhered to for any project that deals with information [73]:

1. Respect for human autonomy,
2. Avoiding harm,
3. Beneficence,
4. Justice.

These compare favorably with Isaac Asimov's classic Laws of Robotics. Asimov's robots serve for the good of humanity:

1. A robot may not injure a human being or, through inaction, allow a human being to come to harm;

2. A robot must obey the orders given to it by human beings, except where such orders would conflict with the First Law;

3. A robot must protect its own existence as long as such protection does not conflict with the First or Second Laws.

Any unmanned system which violates the above principals or laws cannot be considered a robot. In most cases, if these omissions are by design, an unmanned system is a weapon or strategic asset as in the case of military operations.

There have been some attempts to discuss the ethics of UASs, but they have been focused on military or law enforcement applications [74, 75]. These applications have more apparent moral dilemmas, but are more commonly covered by literature. Remote sensing applications more closely follow computer ethics and the ethics of information technology.

Some these concerns parallel the ethical concerns of a Google Street View vehicle collecting information as it navigates routes on each continent. During its mission, a PRS UAS may collect inadvertent information–but what information is too private? What information and operations may affect the human operators? What effects does this have on the operating environment and airspace management? Only scale of operations and applicable laws can ultimately determine the answers. It is also unethical to remain in the airspace if the quality of the data being collected is below a predefined threshold, since the data mission will have failed in this case.

4.2.4 Successful Data Missions

Three main components are needed for successful data missions flying in civil airspace:

1. **Data as the Mission.** Unmanned aerial systems for personal remote sensing (PRS) are, like any unmanned system, defined by their missions. Personal remote sensing, however is focused on data collection, and therefore a PRS mission can be defined directly as data. While quality as a concept is notoriously difficult to define [37], Data Mission Quality (DMQ) is a measure of the relative quality of the data mission,

and can be used as a yardstick to compare multiple missions, different UASs, and different payloads. Mission assurance, in a PRS sense, is Data Mission Assurance (DMA). Data Mission Assurance is the upholding of a minimum DMQ. This means, for example, without a fully functional payload, there is little reason to fly and make use of the airspace allocated to the sUAS.

2. **Ethics.** Remote sensing applications more closely follow computer ethics and the ethics of information technology and medical record handling, and deserve specific, detailed, and clear directions for accountability and transparency in remote data collection. In addition, RS data has been collected for years in manned aircraft, while privacy complaints have not been prominent.

3. **Management of Airspace Safety.** The US Federal Aviation Administration (FAA) manages the civil airspace and sets standards for airworthiness in the US An overall AERIS-compliant architecture is needed for the thousands of small UAS to interact, and fly safely. Therefore, thoughtful architectures and policies based on those architectures are needed.

4.3 Existing Civil UAS Architectural and Ethical Research

Although the problem of integration of sUAS into the civil airspace has many similarities to integration of manned civilian air flights, there are some key differences which make the challenges for sUAS–and specifically sUAS PRS-style missions–different from the style that the existing civil airspace management system is intended to handle. The metrics influencing the outcomes for information collection missions of PRS systems are based on factors such as sensing coverage (e.g., covered area of interest), time of day, weather, and many other factors. From a robotic perspective, this difference can be seen as one of tasking; i.e., data collection (PRS mission) vs. transportation (the main goal of the majority of civilian passenger flights). In addition, the ADS-B protocol used for linking current flights' data is inherently insecure and is not predicted to scale to meet the needs of a complex airspace [76].

Current sUAS PRS avionics architectures, for example, are at a state similar to one in the history of manned civil avionics. This is necessarily because sUASs occupy smaller scales than current civilian architectures overall. Comparing Fig. 2.4 (the AggieAir block diagram) with Fig. 4.7 from Moir [77], it can be seen that AggieAir is mostly "distributed digital" in its current form. As technological progressions are made, civil sUAS avionics architectures will evolve to be more integrated, standardized, and reliable. So too can all parts of the civilian aerial architecture evolve, to more integrated and layered systems which allow PRS flights and smaller scales.

AERIS compliments existing sUAS architecture literature. Most research in sUAS is robotic (path planning, teamwork, tasking, etc.). Quality work has been done targeting the large-scale integration problems UASs in the NAS will experience, but mainly from a communications and control perspective [78], and more encompassing work such as Heisey et al. [79]. Cyber-physical perspectives on crop management and actuation are also being investigated [80], but without inclusion of ethical or safety concerns. Other literature focuses on ethical ramifications of UAS data collection (in the case, video and spy-style flight) [75] or cloud-enabled civil applications [81], without the inclusion of an overall architectural

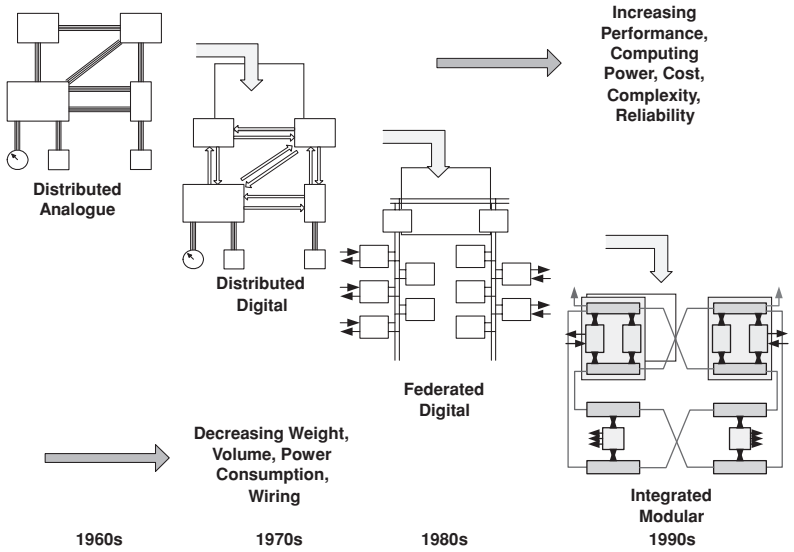


Fig. 4.7: Evolutional history of civil avionics architectures from "Civil Avionics Systems, 2nd Edition" by Moir, Seabridge, and Jukes.

viewpoint. A broader view of safety and ethics is taken in publications such as Lin et al. [82] and Clarke [83], but no treatment of scientific data collection or CPS is included.

4.4 The Three Elements of Ethical Aerial Scientific Data Collection

An AERIS-compatible architecture includes three main areas of definition. They are Airspace Management, Data Mission Success, and Privacy by Design.

4.4.1 Airspace Management for Safety

For more information on Airspace Management for sUASs as a topic, see Brandon Stark [84].

The FAA has defined the Safety Order of Precedence [85]. Seen in priority Table 4.1, this order of precedence refers to how a safety management system should be designed. The most reliable safety systems have the highest priority, while the least reliable systems must be used sparingly as they are the highest risk to system safety. The safety order of precedence in Table 4.1 is the approach for all safety considerations that is needed to enforce safety through architecture. Airworthiness addresses the first two priority levels of safety by incorporating fail-safes, system redundancies, and automatic termination sequences. Any active part of the system not requiring human intervention is the first priority in designing safety.

1. **Airworthiness Issues.** Automatic safety elimination, or hazard reductions;

2. **Human Factor Issues.** Incorporating human-automation interaction developments, such as heads-up displays to improve Situational Awareness;

Table 4.1: Table of FAA safety priorities.

Description	Priority
Design for minimum risk	1
Incorporate safety devices	2
Provide warning devices	3
Develop procedures and training	4

3. **Concept of Operations (CONOPS).** Standardized procedures and operations, detailed information on general and specific mission requirements (see Fig. 4.8);

4. **Training and Documentation.** Standardized systems (the FAA has given example mission logging requirements in their "Unmanned Aircraft Systems Test Site Selection" document, Appendix B, page 24 [86]).

When automatic safety systems are not capable, the burden of safety becomes placed on the human operators. Situational awareness and the optimization of operator cognitive load will help ease human-UAS interaction. Training systems, documentations and certifications fall into the last level of safety. These are traditionally the least reliable methods for ensuring safety. Loss of safety is always the end of mission. In UAS missions loss of safety can happen for a variety of reasons, in part:

1. Human factors failures;

2. Airworthiness issues;

3. Flight hours (wear) on components;

4. A given airframe's safety history;

5. Crew training hours and history;

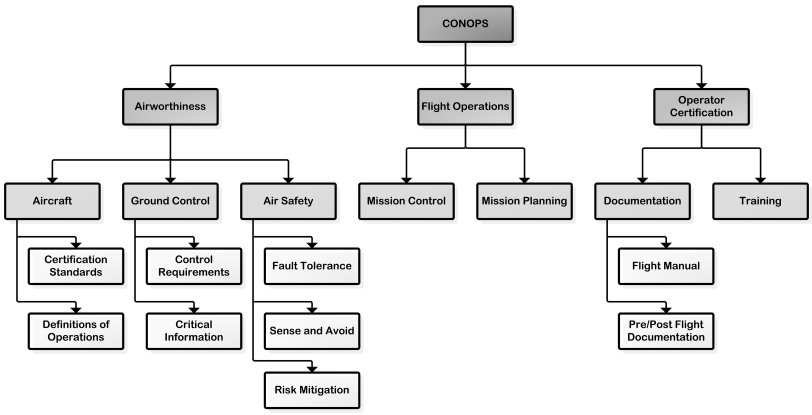


Fig. 4.8: Tree representation of COncept of OPerationS.

6. Failure of forward-looking cameras or other safety pilot hardware;

7. Failure of automatic dependent surveillance-broadcast (ADS-B [87]) or applicable future technologies, such as NextGen [88].

4.4.2 Data Mission Success ("Dataworthiness")

Since data is the mission during PRS, a UAS must be able to estimate the quality of the data during progress. Data mission quality estimation can then lead to data mission assurance, and eventually to mission success, and scientific data fidelity. Technical aspects of instrumentation and UAS flight-specific factors effect dataworthiness. Data mission quality is most directly affected by the quality of the data itself. Sensor quality, weather, and many other factors are at play when a PRS system is collecting data:

1. Airframe (data from autopilot and other autopilots in a swarm):
   (a) Control efforts (detect actuator damage, etc.);
   (b) Battery performance and quality;
   (c) Motor performance (engine efficiency and wear for combustion powered-aircraft);
   (d) Flight hours on components;
   (e) Wind and other environmental effects;
   (f) Navigation performance (Kalman filter convergence, sensor noise estimates).
2. Payload (Data quality metrics):
   (a) Voltage and current on power supply rails;
   (b) Vibration statistics;
   (c) Sensor-specific quality metrics, such as thermal camera core temperature drift, camera capture times, and many others;
   (d) Data mission completeness (from other UASs in the mission as well).

4.4.3 Ethics: Privacy by Design

Ethical considerations for data collection are the major political barriers for allowance of RS UASs into the civil airspace. The collection of data by RS systems could capture sensitive or private details and lead to a violation of the right to reasonable privacy (for those inadvertently "caught" by the data collection system). This lack of control and lack of accountability naturally leads to fear, and therefore to intolerance.

Personal remote sensing systems need to have good boundaries with data (prevention of misuse), before (planning), during (flight path rules), and after (data security), as well as prevention of hacking or other unauthorized access overall. Privacy is an expected right [73], and therefore must be preserved.

The ethics surrounding using UAS RS systems to collect data is most closely related to computer ethics [89]. One approach to solving these problems is called "privacy by design," or PBD [90]. Privacy by design is an architecture-based solution to the privacy issue by engineering a data collection system to account for privacy from the ground up.

In part, PBD defines seven rules for data sensing UAS missions (from Cavoukian [90], where the term "drone" is used for UAS):

1. The purpose for which the UAS will be used and the circumstances under which its use will be authorized and by whom;
2. The specific kinds of information the drone will collect about individuals;
3. The length of time for which the information will be retained;
4. The possible impact on individuals privacy;
5. The specific steps the applicant will take to mitigate the impact on individuals privacy, including protections against unauthorized disclosure;
6. The individual responsible for safe and appropriate use of the drone;
7. An individual point of contact for citizen complaints.

Since the PBD effort is comprehensive and has momentum, AERIS-compatible architectures should abide by the most current PBD ruleset. Explicit use of PDB is crucial for long-term public acceptance.

4.5 AERIS PRS Architectures and Implementations

4.5.1 Multiscale UAS-CPS Architectures

Since architectures are useful and any scale (from the very small to the very large), it is reasonable for some of the rules to apply to all of the architectures in a given system. Multiscale hardware is seen depicted in Fig. 4.9. Since smaller architectures are included in larger ones, failure at any of the scales can possibly contribute to overall failure unless properly engineered by segmentation and layering. This means that individual layers be given standards to be resistant to the common cross-layer information sharing. Although only loosely fitting the definition of "information," this can include external disturbances such as electromagnetic interference or physical energy such as shock or vibration.

Figure 4.9 shows the layered integration of many sUAS subsystems into first a single cyber-physical system, then finally a cyber-physical system of systems. To achieve this level of reuse and integration, architectural design standards must be followed. Some of the following technical aspects are relevant for sUAS-CPS use:

1. Per-layer timing requirements (such as in Sec. 4.2.2);
2. Software architecture and software auditing standards;
3. Security standards for intrusion prevention.

4.5.2 General Small UAS Control Architecture

The control loops of a small UAS consist of attitude control, altitude control, and position control, as shown in Fig. 4.10. The attitude controller controls the UAS's orientation, and uses IMU/AHRS to close the feedback loop. An altitude controller typically combines

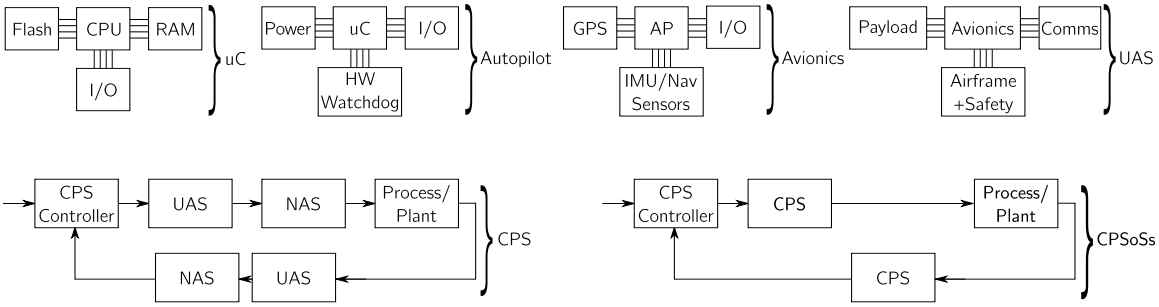


Fig. 4.9: Nested, multiscale UAS-CPS architectures.

measurements from a pressure sensor and GPS altitude. Ultrasonic and laser altimeters can be used as well. The position controller navigates the UAS in the airspace according to the mission requirements or safety copilot/operator input. Attitude and altitude controllers feed commands to the actuators, whereas position control calculates the desired altitude and orientation.

4.5.3 Optimal Multicore CPU Configuration Scenarios

Future system chips, besides including high-performance graphics cores and realtime processing/control units, will include many (on the order of 16-128) CPU cores. The proper use of these cores is an unanswered question in general practice, and remains an unasked question in robotics. Since each CPU core has its own resources, a natural solution is to

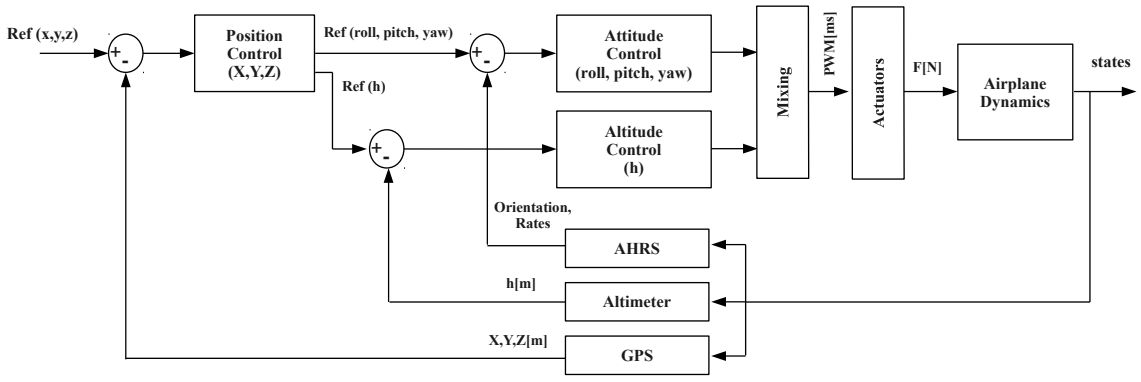


Fig. 4.10: General UAS flight control loops.

provide each software process in the decision-making or mission-management code with its own core, allowing maximum separation between memory and physical processing for each process running the logic for a given robotic system. However, there are more resources available, and they should be exploited for fault-detection and tolerance as well as mission assurance through active software process validation. This will allow use of the available CPU resources to the fullest extent safely. Hard- and soft-error recovery are both possible.

- **Soft-Error Recovery.** A soft-error is an error solely of software; commonly called a "bug." Each software process created by the mission software be put on its own CPU core, with several "helper" processes which will be spawned along with the main process. Through a message-passing interface, these threads will check the main thread for one of many fault conditions, and will be given access through the system kernel to restart the main process in question, thus killing and restarting themselves.

- **Hard-Error Recovery.** Hard-errors are, naturally, errors induced by hardware failure. More specifically, hard-errors are caused by a breakdown of the CPU structure itself, lending the system to produce errors ranging in severity from subtle numerical errors to quite unexpected jumps in the system state machine. Hard-errors are notoriously difficult to detect, and in this conception, all unused CPU cores in a given processor can be assigned similar "helper" processes, which are checking the processes on the other task-used cores. These processes, are also given the authority to reset and restore the operation of the main cores, even to move a process from one core to another if several resets have been triggered and a particular core is probably damaged. In addition to an off-chip hardware watchdog, such internal helper processes will make the best use of the computing resources available, and will extend mission safety and assurance. These helper processes will effectively produce a "distributed hypervisor," allowing many different internal check to be spread across many more simple threads, while the overall system state machines are all checked and assured.

4.5.4 Implementation of Architectures: Formality, Security, and Safety

Architecture defines rules for systems, and from this, properties such as safety and efficiency result. This makes architecture crucial to setting policy; indeed architecture is the overriding force behind good policies and laws. Figure 4.11 (made by Leveson [91], adapted from from Rasmussen [92]) shows this interaction from top to bottom; a so-called "social-technical" model.

Other examples of systems designed for complexity management include legal systems and government [49] and any number of biological systems [50]. Proper architecture design will allow both robustness and flexibility while minimizing fragility [52] for the long-term. Therefore, proper architecture is the only way forward for DMQ-based sUAS remote sensing.

Leveson also writes about safety models and formal methods of establishing architectures specifications [93]. AERIS-compliance someday needs to be concrete: formal methods such as those used in software engineering (for example, SpecTRM-RL [94]) could be used

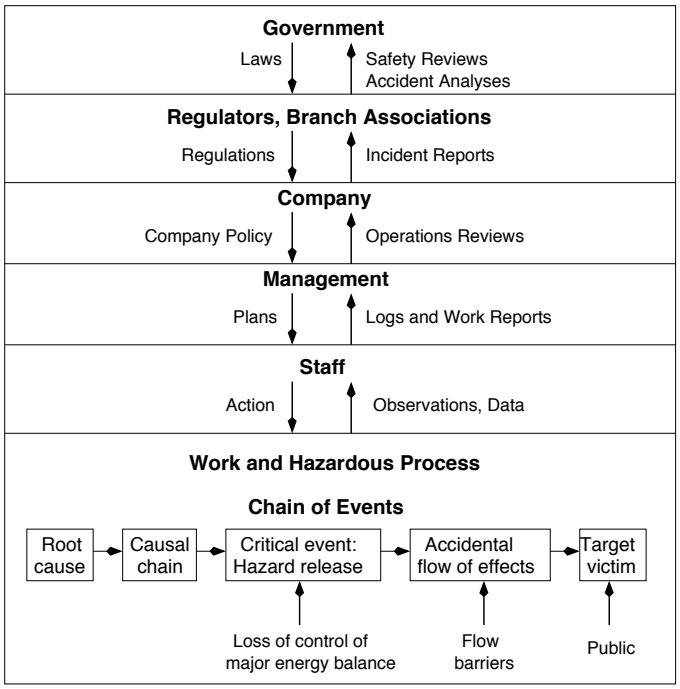


Fig. 4.11: The Rasmussen and Svedung socio-technical model of system operations, from Leveson, adapted from Rasmussen.

to establish a formal description of AERIS-compliant architectures and to assure implementations hew to AERIS specifications via Lustre or other descriptive languages [95].

- **Timing Requirements for DMA.** The data volume generated by PRS missions under AERIS will most likely be large, the timing requirements of actionable information delivery (including scheduling of data collection) differ from mission to mission and in a CPS are specific to the CPS functionality. These requirements are also linked to the physical constraints of the sensing platform, and environmental constraints such as weather, etc. This is an ongoing avenue of research; interesting interpretations can be found e.g., from Bogdan et al. [96, 97].

- **Software Architecture and Software Standards.** Software at all levels of AERIS is important: from the low-level autopilot code (such as the as-yet uncertified Paparazzi), to the payload software such as in Chapter 5, to the storage infrastructure which keeps the information after AERIS-enabled missions. Software testing standards can apply to AERIS and should be used. For example DO-178B/C describes the safety standards of airworthy software [98], where IEEE 29119 is a testing standard rubric which can be applied more generally [99].

- **Security Concerns for AERIS-Compliant Systems.** As globalization of data networks grows and efforts like the "Internet of things" become a reality, PRS sUASs which are necessarily made from similar basic blocks, will be vulnerable to attacks and compromises. Just as current Internet servers, cellular phones, and other embedded systems have architectures which can allow for outside access, all levels of a PRS sUAS (such as those in Fig. 4.9 and beyond) are vulnerable to such attacks. In addition to error recovery and other system robustness attributes, CPSs and indeed PRSs must be designed with security in mind.

  The possibility of intrusion and exterior control of physical systems such as self-driving passenger vehicles is a real physical threat due to their kinetic energy, and sUASs are no exception. Also, since no security is perfect, the goal of any security effort is to

delay the inevitable (penetration or compromise). Techniques such as hypervision [71] allow an active system (e.g., autopilot) to be running on a virtualized processor hosted virtually on hardware, and in the case the autopilot state becomes compromised or corrupted, problems can can be detected and mitigated [100].

An example of a successful hypervision implementation in the consumer product space is Microsoft's Xbox 360, which used a system hypervisior to resist the execution of unauthorized (unsigned) program code [101]–a technique which upheld security and withstood longer than 2 years' time against intrusion with a very wide market for exploitation [102]. For sUASs, an option for supported hypervision in ARM-core processors is Atmel's Trango platform [103].

4.5.5 Robotic Computing Hardware Outlook

Mobile and cellular applications are driving consumer computing platforms to be more integrated with lower cost. Graphics hardware co-processors are becoming the on-chip norm for many application processors, and integrated "peripherals" like digital signal processors are being included in applications processors as default. The trend will continue to future computing platforms such as NVidia's Tegra platforms, and other computing systems designed for integration consumer electronics. This will allow the creation of systems such as in Fig. 4.12.

4.6 Control, Estimation, and Behaviors in AERIS

As a PRS/CPS mission progresses, the top priorities of AERIS are to keep safety, then data quality as high as possible. To this end, flight control loops are part of any AERIS-compliant system. At a basic level, this equates to well-tuned PID (for example) control loops, which are applicable to most (if not all) application scenarios. However, many factors effect the performance of a UAS during a PRS mission. To build true, long-term CPSs enabled by PRS UASs, the UASs must be prepared to deal with factors such as weather, payload changes, physical wear (such as actuators, control surfaces, etc.), and

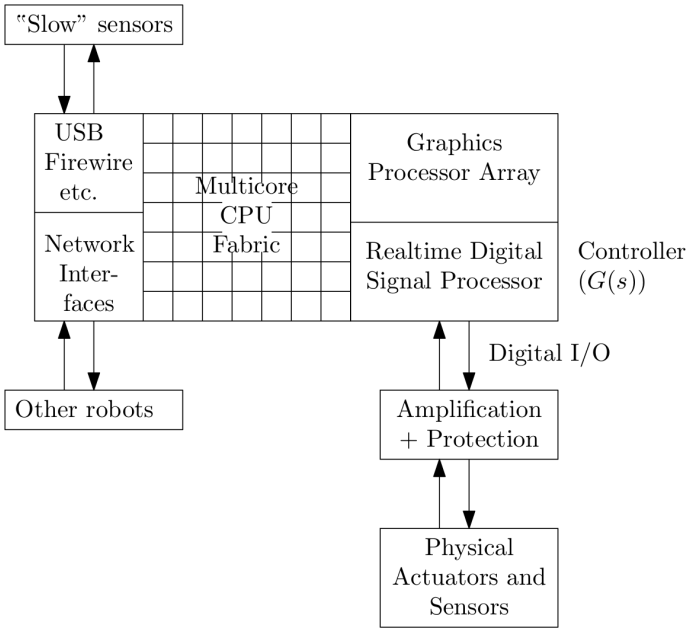


Fig. 4.12: Homogeneous CPU with interface assignments.

external factors such as other aircraft in the nearby airspace.

In the AggieAir system, Intelligent Safety and Airworthiness Co-Pilot, ISaAC (also discussed in Chapter 5) is a general-purpose, relatively high-performance computer. ISaAC acts as a safety co-pilot, processing detailed data from the autopilot such as mission status, navigation information, and control efforts. A general flowchart of ISaAC's system estimation, behavior, and control algorithm is in Fig. 4.13.

ISaAC is also in communication with the payload, receiving data about the readiness of the sensors, and other performance metrics determined per-payload. From these payload and avionics data, ISaAC is able to determine mission quality. To estimate mission safety, ISaAC is connected to any number of safety sensors such as forward-looking cameras and sense-and-avoid systems. In this way, ISaAC can help avoid unsafe conditions. These behaviors can take into account component history, global location, or any number of other factors, and dictate mission modification states such as mission end or abort conditions much faster than a ground observer.

In order to determine aircraft health, ISaAC can be running iterative system estimation routines (many techniques exist, such as full-state extended Kalman filter (EKF) [104],

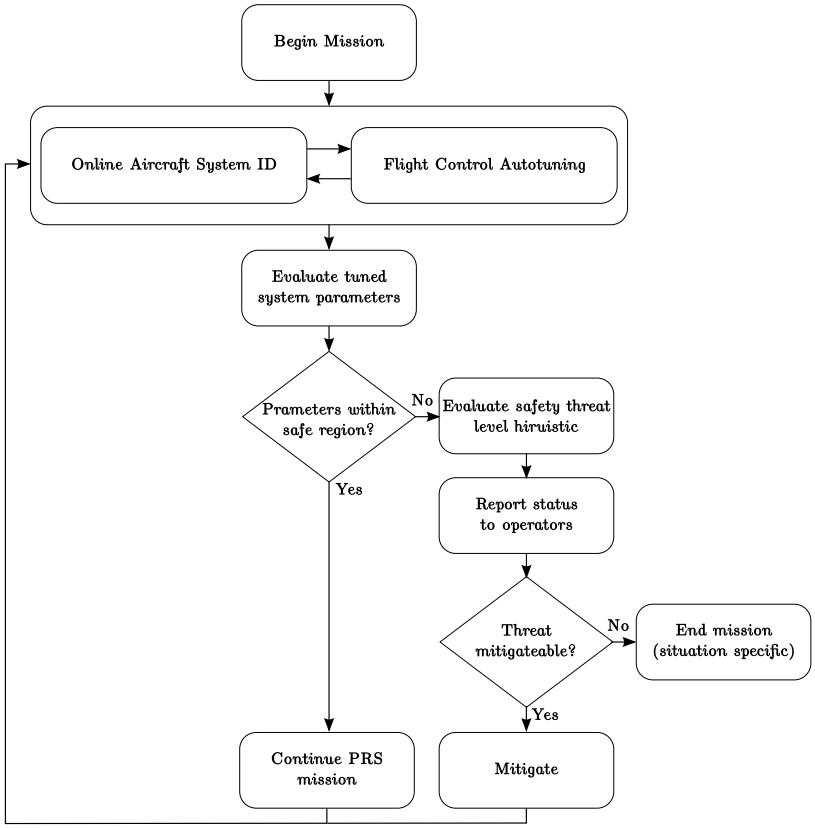


Fig. 4.13: The ISaAC flowchart for system estimation, behavior determination, and control adaptation.

Fourier Transform Regression [105], Neural Net Unscented Kalman filter [106], etc.), with a known aircraft model for comparison. This means that as the estimated aircraft model diverges from the known model, control gains or schemes might need to be adjusted for optimal control performance and safety. Therefore, ISaAC can also be running concurrent auto-tuning or adaptive control algorithms, allowing for optimal control parameters to be set according to the UAS and payload design's specific requirements. Note that this process allows for adaptation to new payloads–ISaAC can be made aware of payload requirements for system response and adjust the system accordingly if these parameters are known beforehand.

If at any time the parameters generated by the system ID and estimation are beyond some pre-defined boundaries set by the aircraft limits, ISaAC can attempt to determine the cause of the fault. Such prognostics are heavily dependent on the aircraft and the

mission, but incremental warning signs can help determine mission risk and allow ground crews to avoid unsafe flights by repairing or replacing suspect components before an in-flight emergency occurs. Once the fault cause is determined, the severity can be evaluated based on safety and payload requirements, and either mitigation can occur (again, control adaptation such as degraded-mode actuation), or mission termination in accordance with the specific safety situation.

4.7 An Analysis of Selected Open Source Robotic and UAS Architectures and Comparison to AERIS

AERIS compliance is challenging for current UASs, mainly because it involves more than simple engineering and technical solutions. Nevertheless, several existing architectures for robotic missions are analyzed in this Section, to determine their ability for AERIS compliance or lack thereof. Since AERIS is all-encompassing and is based in many different aspects of UAS design, implementation and operation, the following systems are evaluated by their contributions to AERIS compliance if not their fulfillment. Table 4.3 summarizes the information collected here about currently available UASs for PRS use.

- **Safety and Airworthiness.** Aerospace applications are high-risk, so engineering and testing standards are common. The ultimate goal of aerospace system design is to be testable and verifiable so that risks are categorized and quantifiable. To this end, testability is very important for ethical use of airspace, and higher engineering standards can be used to provide lower risk levels and better AERIS compliance.

- **Airspace Management.** The best way to measure how a UAS fulfills the airspace management segment of an AERIS-compliant architecture is the FAA's Safety Order of Precedence [85]. The highest priority is designing for minimum risk. A UAS can fulfill this by using a clearly written source code which is well tested. From a hardware point of view it is important to know the maximal drift of the navigation system–Inertial Measurement Unit (IMU) and Attitude and Heading Reference System (AHRS). Usually these values are provided in deg/hour; the magnitude of drift

limits the GPS-denied mission time (beyond which the estimated attitude can be too far beyond a safe level of variance to be considered safe).

Fail-safe devices are implemented as a state machine covering possible situations with emergency flight termination procedures (see Table 4.2 for basic useful failsafe behaviors). For example: warning devices (priority three) are provided to the Ground Control Station operator, who sees the airframe telemetry. However, during complicated missions when the cognitive load is too large, a simple visualization of data is not sufficient. To get the operator and pilot's attention, additional devices have to be implemented–for example audio and visual warnings. Since many UASs do not implement these features, they must be added to be AERIS compliant. Safety procedures for the crew are important part of the overall airworthiness, but most UASs do not yet provide specific instructions.

- **Dataworthiness.** Currently most UASs are developed as an autopilot and a ground control station only. In the autopilot code, they can provide data about the airframe (navigation, control efforts, battery monitoring, etc.), however the payload data (Data Quality Metrics) are not part of most UASs and are not included with their standard mission code.

- **Ethics: Privacy By Design.** Ethical constraints are the most difficult to implement. Most UASs are not designed for privacy from inception. Currently the responsibility for ethical use of the UAS with most systems depends solely on the operational crew. Although some ethical measures are commonly implemented (such as a predefined flight plan), much work is still needed in this field.

Table 4.2: UAS behaviors which are simple and useful.

Situation	Behavior
Loss of telemetry link	Return to the base
Loss of radio link	Emergency landing or return to the base
Loss of GPS	Emergency landing
Low battery	Return to the base

Table 4.3: Table of UAS and autopilots viable for personal remote sensing.

UAS Name	Inception	(Hardware) Platforms	Cost	CPU(s)	Nav. Sensor Suite	Sensor Options
(RT) Paparazzi	2003	(Multi) Fixed-wing, Rotary	$	168Mhz 32bit w/FPU	Many	Many
3D-Robotics Pixhawk (PX4)	2009	(Single) Fixed-wing, Rotary	$	168Mhz 32b w/FPU + 72Mhz 32b	Proprietary IMU	Many
Procerus Kestrel (3.2)	2004	(Proprietary) Fixed-wing, Rotary	$$$$$	Not published	Proprietary IMU	Many
Robot Operating System (ROS)	2007	(Linux-based) General	$	Dependent on CPU	None/Many	Many
DJI Phantom (2.0)	2011	(Proprietary) Rotary	$$	Not published	Proprietary IMU	Few
AggieAir (2.0)	2007	(Multiple) Fixed-wing, Rotary	$$$	168Mhz 32bit w/FPU + 1200 MIPS Linux	3DM GX3/Many	Many

UAS Name	Control BW (Hz)	Ext. Airborne Interfaces	Payload for Remote Sensing?	AERIS Compliance Possible?
(RT) Paparazzi	500	I2C, SPI, CAN-Bus, Serial, A/D	No	Not as implemented
3D-Robotics Pixhawk (PX4)	200	I2C, SPI, CAN-Bus, Serial, A/D	No, Possible	Not as implemented
Procerus Kestrel (3.2)	500	I2C, SPI, CAN-Bus, Serial, A/D	No, Possible	Not as implemented
Robot Operating System (ROS)	Dependent on CPU	Any Linux	No, Possible	Not as implemented
DJI Phantom 2 Vision	200	Micro USB, CAN-Bus	No, Inc. Gimbal Camera	Not as implemented
AggieAir (2.0)	500	I2C, SPI, CAN-Bus, Serial, A/D, Ethernet	Yes	Yes

4.7.1 RT-Paparazzi

Paparazzi is a free and open-source hardware and software project intended to create an exceptionally powerful and versatile autopilot system for fixed-wing aircrafts as well as multicopters by allowing and encouraging input from the community [20]. Paparazzi is released under the GNU [107] license. A real-time port of the Paparazzi autopilot code, RT-Paparazzi, was shown in 2013, and for this study it is considered from a features and functionality standpoint.

RT-Paparazzi has two main parts, the Airframe segment (containing code for avionics and additional sensors) and the Ground segment (Ground Control Station interface, compiler tool-chain and some additional tools). The Airframe segment is based on ChibiOS [108] and written in C (for embedded hardware), the Ground segment in OCAML [109] and Python with some specific tools in MATLAB.

The overall Paparazzi structure is pictured in Fig. 4.14, and is a generalized version

of the command and control loop of a flying UAV. This system is controlled by a ground station, which commands the UAV to fly from waypoint to waypoint, however the actual control law remains as in Fig. 4.14 from the Paparazzi project.

- **Safety and Airworthiness.** Since RT-Paparazzi is implemented in a real-time OS, it is possible to measure timing and verify each thread's performance both on the ground during testing and in flight. Software upsets can be detected in flight to detect errant code or hardware, but these mitigation decisions are left up to the operators during the UAS mission.

- **Airspace Management.** Paparazzi implements all of the behaviors in Table 4.2.

- **Dataworthiness.** Paparazzi alone does not implement remote sensing payload functionality directly. However, due to the open-source nature of the project, different payload interfaces are possible through defined I/O on the autopilot board.

- **Ethics: Privacy By Design.** Ethical constraints are the most difficult to implement for any UAS. Paparazzi is not an exception; it was not designed for privacy from

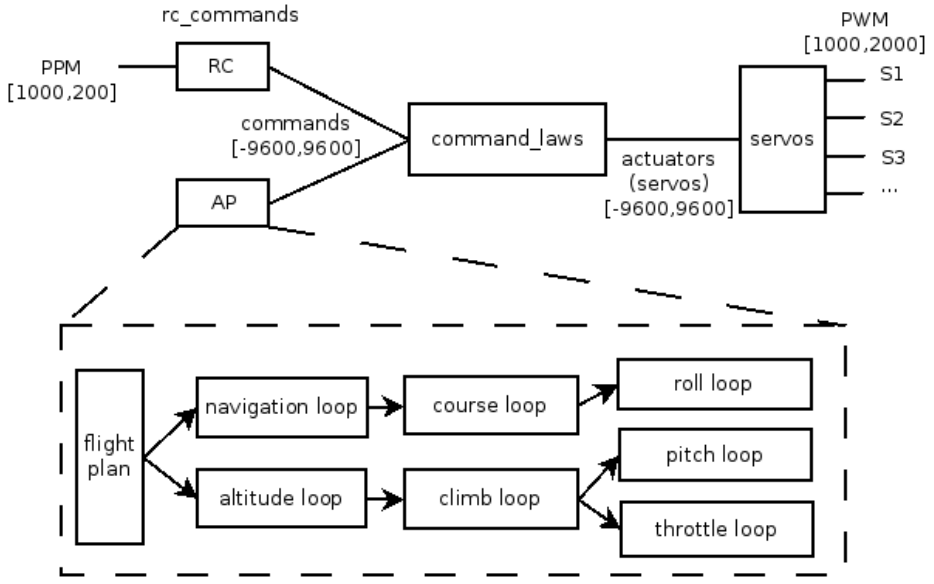


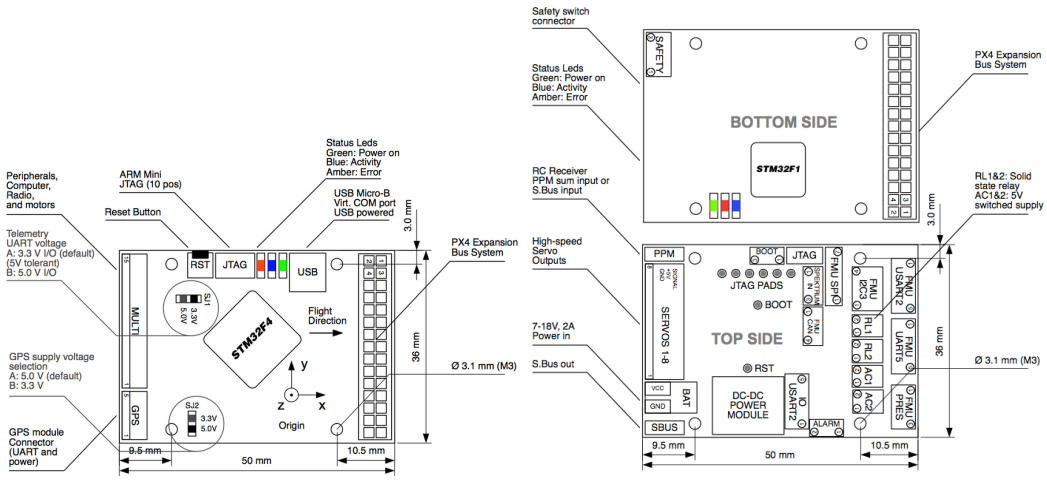
Fig. 4.14: An overview of the Paparazzi control scheme from the Paparazzi project.

inception. Currently the responsibility for ethical use of the UAS with Paparazzi system depends solely on the crew.

4.7.2 PixHawk PX4

The Pixhawk autopilot system is a project of many contributors, managed by ETH Zurich [110] and released under the BSD/Creative Commons license [111]. Like Paparazzi, Pixhawk's PX4 autopilot (diagramed in Fig. 4.15 from the Pixhawk project [110]) is not a true UAS, 3D-Robotics [112] sells fully integrated and tested rotary unmanned platforms based on the PX4.

- **Safety and Airworthiness.** The Pixhawk PX4 autopilot system is comprised of two major parts (both seen in Fig. 4.15): the Flight Management Unit (FMU) and an I/O module with power, data interfaces, and a backup/override processor to allow for redundant manual flight control, should the main autopilot encounter an error condition. Like RT-paparazzi, the PX4 autopilot system is implemented in a real-time operating system (NuttX [113]), and therefore is expandable and testable for software errors before and during flight.



((a)) The Pixhawk PX4 FMU from [110] ((b)) The Pixhawk PX4 IO board from [110]

Fig. 4.15: An overview of the PixHawk avionics system from the Pixhawk project.

- **Airspace Management.** The Pixhawk system implements all of the behaviors in Table 4.2, depending on its configuration.

- **Dataworthiness.** The Pixhawk system alone does not implement remote sensing payload functionality directly, although throughout the project's history image capture and processing as been targeted as a mission parameter. Due to the open-source nature of the project, different payload interfaces are possible through predefined I/O on the avionics system boards. In addition, at the time of this writing, the Pixhawk project reports upcoming integration with ROS (see Sec. 4.7.3) for a more integrated robotic system, which could give the Pixhawk system enough computational oversight to comply with AERIS.

- **Ethics: Privacy By Design.** As with most of the listed UASs, ethical constraints are not implemented in a technical sense and depend solely on the crew. Since the PX4 system can fly fixed- and rotary-wing craft with pre-programmed waypoint flight, it is possible to avoid untended data gathering during mission planning.

4.7.3 ROS

One example of a successful robotic architecture is the Robot Operating System, ROS [114]. As with any good operating system, ROS is flexible and extensible, and implements the main aspects of Doyle's architecture requirements (data and process separation). ROS is node-based, allowing designers to implement data and behavior flows as they deem necessary at a high, graph-based level (seen in Fig. 4.16 from the ROS project documentation [114]). The operating system is in the middle of the bow-tie, coordinating data and action in a robust way.

- **Safety and Airworthiness.** ROS is based on solid design principles, but does not target high-speed hard real-time computation. Since ROS is implemented as software running on a Linux kernel, ROS is missing a critical element of feedback and control to operate a UAS safely.

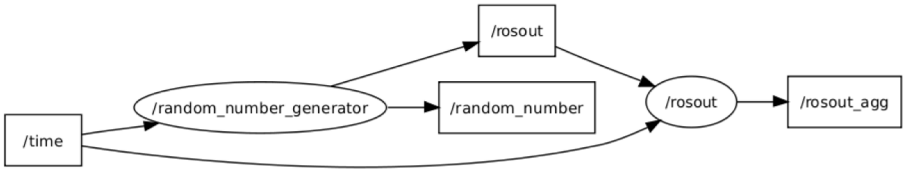


Fig. 4.16: The node-based architecture of ROS, from the ROS project documentation.

- **Airspace Management.** Since ROS is not specifically targeting UAS use, it is not specifically oriented to airspace management. However, it is possible to implement all of the behaviors in Table 4.2.

- **Dataworthiness.** ROS is superior to many UASs with respect to dataworthiness, since it is designed to provide reliable software/hardware interfaces to sensors such as 3D cameras, GPS, etc. However, without pairing with a true autopilot-based UAS (such as Paparazzi or Pixhawk), ROS is not a good candidate for a full AERIS-compliant UAS.

- **Ethics: Privacy By Design.** Ethical constraints are not implemented in a technical sense within ROS, but due to ROS's high-level functionality it is more possible to implement computational oversight such as no-fly boundaries, etc. As stated above, ROS must be paired with a true UAS autopilot system, as well as an ethical support network to fulfill AERIS requirements.

4.7.4 DJI Phantom 2 Vision

Developed by Da-Jiang Innovations Science and Technology Co., Ltd. of Shenzhen, China, the DJI Phantom 2 Vision UAS [115] is a full flight system (unmanned aircraft, safety pilot control system) with gimbaled HD camera pictured in Fig. 4.17 from the DJI corporate website [115]. Complete with a 14mp camera, when coupled with an iOS or Android mobile device as a ground station, the DJI Phantom allows for pre-planed flight maneuvers (i.e., beyond line-of-sight), and 1080p HD video or still pictures to captured and transmitted down to the operator. Easy social sharing features (Facebook, Twitter,

and more) allow for instant propagation of media captured when the ground station device (iPhone, etc.) has an Internet connection. The low cost of entry and ease of use of the Phantom 2 make UAS technology truly more available than before.

- **Safety and Airworthiness.** Although DJI has been making autonomous rotary hardware for hobbyist/enthusiast market for many years, the Phantom 2 Vision represents the first mass-market full image and video platform with GPS waypoint and other autonomous pre-programmed functionality. Although DJI does not make full system specifications open to users or developers, interfaces like CAN bus show an outward commitment to airworthiness.

- **Airspace Management.** While the Phantom 2 Vision does implement the behaviors in Table 4.2, it is intended solely for noncommercial use in the US However high-quality camera options (such as the now film industry-standard GoPro) show the obvious (and subtly marketed) possibilities for commercial video and still image collection e.g., movie/commercial camerawork or realtor house imagery. Since these applications are commercial and therefore forbidden by the FAA, there is no way to manage them safely in the airspace.

- **Dataworthiness.** Although camera options such as gimbals and GoPro sensors are attractive for its target market, they have little to no value in the AERIS-style of

Fig. 4.17: The DJI Phantom 2.0 rotary UAS from DJI.

remote sensing. No information about the quality of the data recorded by the camera is stored. For AERIS-compliant missions, the Phantom 2 is at a low level of dataworthiness for PRS CPS work.

- **Ethics: Privacy By Design.** Since ethics have not been considered in the DJI design, and since the Phantom 2 and other DJI and DJI-style autonomous aircraft are mass produced and are more capable and accessible than before, it is likely that the mass production of this kind of craft actually represents a net lowering of the ethical qualifications. Crew training, ethical training, no-fly considerations are not given with the DJI system and are therefore left to the consumer users of the hardware.

4.7.5 Procerus Kestrel

The Procerus UAS [116] was developed for many years at Brigham Young University's MAGICC Lab before being purchased by Northrop Grumman corporation in 2011. The Procerus Kestrel 3.2 autopilot (Fig. 4.18 from Procerus' documentation [116]) is able to control both fixed- and rotary-wing aircraft, both of which are vended by Northrop Grumman (such as the Unicorn flying wing in Fig. 2.3). Microsoft Windows-based ground station software, training, and many military uses for the Procerus system add to the relatively high cost factor for Kestrel-based UASs.

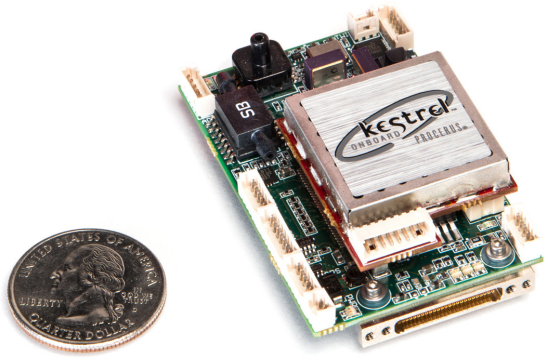


Fig. 4.18: Procerus Kestrel 3 autopilot from Procerus' documentation.

- **Safety and Airworthiness.** Years of proven flight history and the backing by a major global defense contractor, with many flight-hours of the 3.2 and previous versions of the Kestrel UAS indicate a high-level of airworthiness and safety.

- **Airspace Management.** Many behaviors, including all of Table 4.1, are or can be implemented in the Kestrel UAS. Many UAS projects use the Kestrel autopilot for UASs,

- **Dataworthiness.** The Procerus system allows for many payloads such as visible, thermal IR, etc., but is not targeted toward remote sensing and PRS use directly. Metrics about mission quality could certainly be added, but are not present in the current form of the Kestrel UAS, and therefore the system is at a low-level of dataworthiness.

- **Ethics: Privacy By Design.** As with many UASs, the Procerus UAS was not designed with ethical considerations in mind. However, the high cost of entry and the probability for use in tactical or emergency use mean the AERIS constraints to not apply to such a UAS. For civilian PRS and CPS use, as in AERIS, the Kestrel scores no higher than any other UAS listed here.

4.7.6 AggieAir 2.0

Introduced in Chapter 2 and originating at CSOIS [117] in 2006, the AggieAir 2.0 UAS [118] uses a modified Paparazzi autopilot and GCS environment for PRS missions in various environments and missions. Paparazzi is evaluated specifically in Sec. 4.7.1, controls all AggieAir UASs, and is shown in Fig. 4.14. This system is controlled by a ground station, which commands the UAV to fly from waypoint to waypoint, however the actual control law remains as in Fig. 4.14. The AggieAir UAS flight system diagram is seen in Fig. 2.4, which includes the ISaAC safety co-pilot detailed in Chapter 5.

- **Safety and Airworthiness.** AggieAir has many hours over several years of autonomous flight by way of the Paparazzi UAS. With the advent of the RT-Paparazzi

branch, much like the Pixhawk system, hard real-time processing targets can be made and verified, allowing for better testing before flight, and more knowledge of software inconsistencies during flight. Mitigation strategies such as emergency landing or mission termination via parachute can be deployed depending on the failure analysis.

- **Airspace Management.** In addition to the behaviors in Table 4.2, AggieAir flies under many FAA-provided COAs (certificate of authority) with a PIC (pilot in charge) to interface with the local airspace and avoid collision conditions. Coupled with comprehensive training, this allows AggieAir to fly safely and manage airspace.

- **Dataworthiness.** To achieve data mission assurance in current AggieAir payloads, modular approaches to system design were taken, from development to testing, allowing performance and system faults to be quickly and accurately diagnosed. AggieAir has augmented Paparazzi with a modular payload architecture which embodies AERIS-compliant payload design (Chapter 5), also seen integrated in the AggieAir UAS in Fig. 2.4, this design has captured remote sensed data successfully over many hours of safe, autonomous flight (for example, thermal infrared data collection [119]). This is a higher-level approach, which brings Paparazzi to a layered PRS based on DMQ.

  A higher-level approach, which would bring Paparazzi to a CPS based on DMQ (along the lines of Doyle and ROS) is seen in Fig. 4.19 (with Paparazzi image from the Paparazzi project [20], and airspace image from NASA [120]). In this way, Paparazzi has been augmented (via implementation in ISaAC) and controlled by a DMQ estimator for DMA and mission success.

  These concepts must be extended to the context of the greater civil airspace and expressed in a cyber-physical systems context to enable PRS systems to interact with manned aircraft and other UASs while in operation in a safe and reliable way, while delivering overall DMQ.

- **Ethics: Privacy By Design.** Because the Paparazzi UAS only provides a base for the AggieAir operations, ethical concerns are handled at the crew level. This means that AggieAir's standard mission practice is to keep a high-level of security around collected data, as well as only flying missions around targeted agricultural and natural resource areas, avoiding populated and otherwise sensitive subjects. Also, data containing personally identifiable content is deleted, assuring the privacy of any private subjects who might be unwittingly imaged. This makes AggieAir the most AERIS compliant of all the UASs profiled.

4.7.7 AERIS in Current UASs

Overall, many current UASs provide solid bases for AERIS-compliant architecture, but additional systems still have to be implemented to fully comply with the requirements. Current UASs like Paparazzi were not originally developed with AERIS-stye requirements in mind, and therefore the next generation of UASs must be AERIS-compliant at the design stage due to the system complexity required and the impracticality of retrofitting older designs. What follows is an example AERIS Implementation Based on AggieAir.

Since AggieAir allows for the best AERIS compliance of all surveyed UASs, Table 4.4 summarizes the different how AggieAir can comply with AERIS.

Making assumptions for networking and databases, an AERIS-enabled use case is remarkably simple.

1. Remote sensing user (group) acquires AERIS-compliant sUAS,
2. User completes training and is registered as the operator of the sUAS,
3. Pre-flight checks are performed and logged on ground control station,
4. Flight plan is generated and registered with airspace regulators,
5. Aircraft is launched and is monitoring mission performance during flight,
6. End of mission (either planned or exceptional): landing and post-flight information is automatically logged and relevant information is added to a flight database.

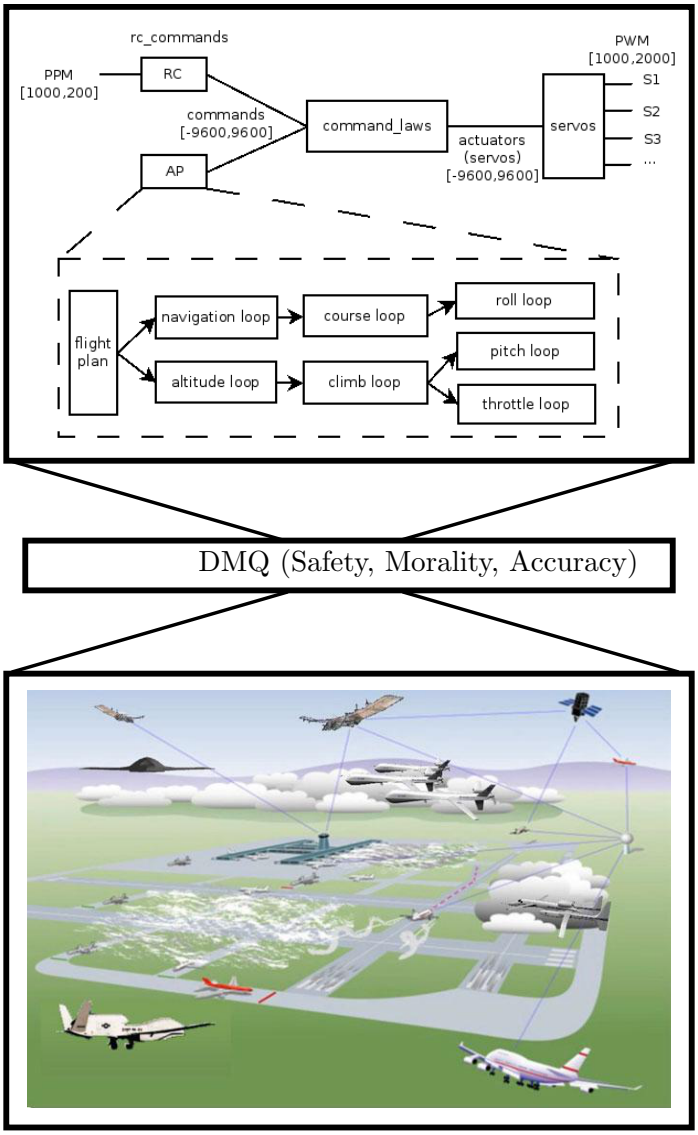


Fig. 4.19: Paparazzi UAS control loops (image from Paparazzi), interfaced to the NAS (image from NASA) via DMQ.

Table 4.4: Table of AERIS elements and AggieAir features.

AERIS Element	AggieAir Component	Comments
Safety and airworthiness	Modular testing	See Chapter 5 for software-related testing examples and details
Safety and airworthiness	In-flight system ID	For more details and overview, see [121]
Dataworthiness	Data mission modeling	Mission dependent
Safety and airworthiness	Sense-and-avoid integration	SnA technology is an active area of research [122]
Safety and airworthiness	ADSB and airspace awareness via Pilot in Charge (PIC)	Airspace management is an active area of research [123]
Privacy protection, Safety and airworthiness	Code of conduct, statement of operations	See Fig. 4.20 for an example code of conduct from Paul Voss of Smith College [124]
Privacy protection	Operational policies and privacy ethical training	As described in PbD literature [90]
Privacy protection	Secure storage of collected data	Strict policies about releasing datasets in addition to security
Dataworthiness	Actionable information production	AggieAir provides actionable information based on quality data
Safety and airworthiness, Privacy protection	Crew training and certification	AggieAir implements a safety pilot and crew training program with tracked hours per crew member, etc.
Safety and airworthiness, Dataworthiness	Data mission quality estimation	AggieAir can land in the case of poor data mission quality and return the airspace to a more safe state
Safety and airworthiness	Redundancy and failsafe design	AggieAir system design allows for redundancy and failsafes as technology and complexity allows

There are many parts of AERIS implementations which are infrastructural, logistical, or simply future research topics; however the core functionality is needed for sustaining sUAS PRS-style data collection flights in the NAS and civil society.

4.8 Chapter Summary

The future of UASs is undoubtedly bright. While the current public perception of UASs is one of espionage and warfare, they will become more accepted into domestic use as their potential value becomes apparent and as the airspace rules change to include them. While current regulations of UASs are restrictive and limited in the US, soon UASs will become available for regular use as standards for certification and airworthiness are developed.

With these standards, mission quality metrics are needed to determine if the UAS is truly in need of the airspace. Along with ethics (privacy by design), an AERIS-compliant airspace access requirement architecture will allow civil flights of many kinds with minimal concern for right violations, allowing humans and unmanned robotic systems to peacefully coexist and grow together.

This chapter discusses cyber-physical concepts, and shows that human biology is layered, and can be compared to modern small application processor hardware for robotic mission execution. The difference between robotic systems and weapons is shown to be ethical guidelines. Then, AERIS is presented as a way for unmanned aerial data collection flights to operate in an ethical way, designed into the systems architectures' many scales, separated by layers. AERIS is shown in a level of detail, and popular current sUAS options are evaluated based on the AERIS criteria are evaluated based on AERIS concepts for their viability in implementing AERIS-compliant architectures.

The concepts of layering and modularity must be applied to all levels of the greater civil airspace and expressed in a cyber-physical systems context to enable PRS systems to interact with manned aircraft and other UASs while in operation in a safe and reliable way, while delivering DMQ overall.

The concepts of layering and modularity must be applied to all levels of the greater civil airspace to enable PRS systems to interact with manned aircraft and other UASs

Code of Conduct for the Use of Small Airborne Objects on Smith College Property

This code of conduct governs the use of Small Airborne Objects (SAOs) on Smith College property. For the purposes of this code, SAOs are understood to include any balloon, kite, rocket, projectile, model aircraft, drone, small unmanned aircraft, or flying toy that is used exclusively below the federal navigable airspace for manned aircraft. SAOs have long been used for teaching, research, sport, and recreation[1]; in the future, they may find additional applications in facilities management, public safety, and campus planning. Authority for this code derives from the long-standing principle that the landowner, in the words of the Supreme Court, has "exclusive control of the immediate reaches" of the airspace[2]. The purpose of this code is therefore to ensure safe and orderly use of Smith College property.

Except for objects used for sanctioned sports and under the purview of athletic director, any Small Airborne Object (SAO) used on Smith College property:

- **shall not weigh more than 2 pounds or exceed 60 mph without institutional authorization[3];**
- **shall not exceed 400 feet altitude without authorization from the FAA;**
- **shall not enter any other property below 400 feet altitude without landowner permission;**
- **shall not create a hazard or nuisance to any person or property unaffiliated with the use[4];**
- **shall not be used for observing individuals or their property without their permission;**
- **shall not carry any weapon or significant amount of any hazardous substance[5];**
- **shall display contact information if the total travel distance could exceed 400 feet;**
- **shall obey local ordinances including those regarding nuisance, privacy, and land use;**
- **shall give right of way to, and not be used in proximity of, any full-scale aircraft;**
- **shall adhere to FAA Advisory Circular 91-57 as appropriate for model aircraft.**

Any SAO not conforming to this code of conduct, or any unidentified SAO of concern below 400 feet on Smith College property should be reported to Campus Police (x800 or 413-585-2490).

---

[1] Model aircraft similar to drones and unmanned aircraft have been freely used for teaching, research, and recreation since the 1930's. Kites, balloons and projectiles have been in use for centuries. Such tools are used in fields ranging from aeronautics, robotics, and art, to environmental science and agriculture to name a few.

[2] *United States v. Causby*, 328 U.S. 256 (1946). See also *Griggs v. Allegheny County*, 369 U.S. 84 (1962); *California v. Ciraolo*, 476 U.S. 207 (1986); *Florida v. Riley*, 488 U.S. 445 (1989); *Argent v. United States*, 124 F. 3d 1277 (1997). In July, 2013, the Northampton City Council unanimously passed a resolution affirming landowner and local control of the immediate reaches of the airspace within the city limits, an area that includes the Smith College campus.

[3] The Institutional Chemical Hygiene Committee (ICHC) currently oversees general safety matters in the sciences at Smith College and will be responsible for the safe use of SAOs on college property. The weight and speed limits specified are based on the Academy of Model Aeronautics (AMA) definition of a "Park Flier", a model aircraft that is considered to be small and safe enough to be used in a public park. In no case shall a SAO exceed 55-pounds or otherwise exceed the physical limits of a model aircraft as defined by the FAA.

[4] Additional care should be exercised when using SAOs that by virtue of their mass, speed, power, or construction could potentially cause a serious injury. Such SAOs shall not be used within 100 feet of any unaffiliated person or within 500 feet of any public road, event, or unaffiliated group.

[5] Low-toxicity batteries of reasonable size are not considered hazardous but should be protected by appropriate placement, padding, and containment. Batteries with capacities greater than 800 mAhr should have a safety fuse.

Fig. 4.20: Sample code of conduct for sUAS operations from Dr. Paul Voss of Smith College (provided privately). Used with permission.

while in operation in a safe and reliable way, while delivering DMQ overall. Eventually, cyber-physical systems will be enabled by AERIS-compliant systems.

Chapter 5

A Payload Verification and Management Framework for Small UAV-based Personal Remote Sensing Systems

Over the past decade, small unmanned aerial systems (sUAS) have become a major area of academic research and a growing sector of the aerospace industry. While these systems have traditionally been used in military applications, civilian applications for these platforms are rapidly becoming a reality. It is expected that these sUAS will experience the greatest growth in civilian and commercial applications [125]. The content in this chapter is based on content from "A Payload Verification and Management Framework for Small UAV-Based Personal Remote Sensing Systems" by Coopmans et al. [11].

For a specific class (typically defined as 50 airborne pounds and under) of sUAS, known as Personal Remote Sensing sUAS (PRS sUAS), the major application is the collection of data. This data can be of any variety: aerial imagery, multispectral analysis (published previously [126]), airborne particle collection, RFID tag locations, etc. These small yet versatile platforms are designed to be controlled by non-experts, yet still provide the highest level of performance for its given mission.

Because the purpose of these sUAS are to gather data, the data is the most important facet of the mission. An architecture devoted to gathering diverse sets of data and ensuring that the data is retrieved successfully is necessary to accomplish the mission. This is known as Data Mission Assurance or DMA.

Because PRS sUAS are intended for personal use, cost must be kept to a minimum. In order to achieve this goal commercially available sensors are used when possible. There has been much research literature in the area of integrating commercial-off-the-shelf (COTS) components into systems that require a high degree of accuracy. It is possible to use the IEEE-1324 and I2C buses more reliably by using various optimizations in tandem, such as

fail-silence and watchdog timers [127]. COTS is also used on satellites, because of the low use of power. sUAS also have power limitations so COTS can also help in this respect [128].

It is also possible to solve the unreliability problem by using multiple sensors to make sure an accurate reading is being obtained [128–130]. It is difficult to have very many redundant sensors on a PRS sUAS so exhaustive testing is used to ensure DMA Testing for fault-tolerance and resilience of COTS systems as also been previously discussed [130, 131]. By injecting faults into the system to ensure proper behavior; both hardware and software simulations were discussed. The testing process shown in this chapter uses the same hardware that will be used in flight, but also modularly tests the sensors. This allows sensors to be combined in any reasonable way and ensures fault tolerance regardless of sensor combination.

Other groups have designed architectures for connecting sensors together but none of them suit the needs of PRS sUAS. Middleware, or a software abstraction layer between services is a possibility [132]. This allows services to be connected to the program physically in multiple ways, but the way in which the main program interacts with these services remains constant. Middleware can not only customizable but dynamically changeable as well. This allows for a sensor network that adapts to changing situations in real time [133]. Long-term sensor networks and power consumption management are also in need of software verification. In order to do this, a central system can be designed that is responsible for triggering all the sensors on the network [134]. The use of many middleware layers can have a significant resource overhead, but algorithms can detect when layers of abstraction can be skipped [135]. All of these methods have merit but none fit the specific needs of PRS sUAS. The system outlined in this chapter utilizes many of the same concepts. What is needed is a system modular in design, giving a high level of customization, as well as very concise descriptions of faults or failures. Since this does not exist in the context of sUAS, a new system must be implemented.

Prior work has been done on centralized control software for UASs and their payloads. Modularity is a main focus and is the ability to add different sensors to a new system and be

assured of compatibility and functionality. It is possible to create a middleware architecture that acts as the information hub for all the components of the UAS, and is flexible in what sensors are used [132]. This is modular, however data acquisition is not a focus of other work. This chapter presents a system that, by design, accepts certain levels of fragility. Others have created a service based architecture that is very robust. If any component fails the rest of the system is still functional [136]; this is very robust but for data purposes if one sensor fails, the data mission is in danger.

Mission assurance is a leading topic in sUAS research and much has been done on the subject. However in the case of a PRS sUAS, data is the mission itself, and there has been little research done. Some work has focused on mission assurance with regards to cyber-security and resilience against malicious attacks. Various methods exist that can be used to attempt to ensure that the system will not be compromised [137].

The gauging risk is also a part of mission assurance. A focus is on managerial decisions that can be made with the right data, and what type of data can be gathered about a particular mission [138]. In order to assure mission success for a PRS sUAS, extensive standardized testing must take place. By simulating a flight and sending that data to a payload, a particular sensor can be tested to ensure correct operation. This increases the resilience of the overall platform. Overall, research on mission assurance exists, but on data mission assurance it is non-existent.

The main contributions of this chapter are the concepts of data as a mission, and of mission assurance as resilience. Coupled with software architecture, this chapter shows that to ensure data mission success, modular, standardized testing and flexible fault handling is needed, especially when consumer COTS (CCOTS) hardware and sensors are used.

5.1 Data Mission Assurance

Unmanned aerial systems for personal remote sensing are, like any unmanned system, defined by their missions. PRS, however is focused on data collection, and therefore a PRS mission is directly defined as data. Mission assurance, in a PRS sense is then Data Mission Assurance (DMA). This means, in effect, without a fully functional payload, there is little

reason to fly and make use of the airspace allocated to the sUAS.

To achieve data mission assurance, a modular approach to the PRS system design is taken, from development to testing, allowing overall performance and system faults to be quickly and accurately diagnosed.

AggieAir PRS systems are designed to be low-cost, and are therefore based on commonly available consumer-grade hardware. To achieve DMA with low-cost hardware, system testing and validation is required to assure performance at all levels (hardware and software). This allows a small UAS to achieve low-cost and high-reliability, a combination needed for PRS.

5.2 AggieAir Architecture

In order to assure mission success for a PRS sUAS, namely the acquisition of data, extensive testing must take place. Payload handling and fault detection is a very important part of the AggieAir system. Since consumer-level cameras and other sensors are not constructed to a relatively high quality level, allowances must be made in payload control software architecture. Figure 2.4 shows this software system design in detail. The performance of this system can be verified through extensive testing: especially useful is a hardware-in-the-loop simulation testing capability, providing simulated flight data into the payload system for confirmation of performance and reliability over any specific mission.

AggieAir at Utah State University has been actively developing the AggieAir architecture, a cohesive framework and system design to bring PRS sUAS to a higher level of safety and functionality. AggieAir encompass the entire sUAS: from the airframe, servos and control surfaces, autopilot and ground station (from the Paparazzi project [20]), to the high-level payload, safety systems and risk mitigation such as parachutes. See Sec. 4.7.6 for more detailed information on AggieAir.

In Fig. 2.4, AggieCap receives the current mission data (global position, roll, pitch, yaw, etc.) from the navigation unit AggieNav (published previously [139, 140]), allowing payloads to geo-spatially timestamp their data as it is collected. This is critical for data collection, since PRS data need to be geolocated to be useful to the end users. Figure 2.4

also shows the separation between simple autonomy (the ability to fly, for instance) to the left, and on the right side, the features of the AggieAir system that enable data mission assurance and resilience such as the Go/NoGo signal from AggieCap and input from ISaAC (see Sec. 5.2.2), with the ability to evaluate the safety of the current mission in the scope of the surrounding airspace.

The AggieAir architecture allows PRS missions to proceed with whatever level of complexity is needed. For example, if a given payload does not need an IP network connection to its respective ground station during flight (as is the case for nearly all current imaging payloads), this can be omitted for simplicity. AggieCap itself is also optional, however no data would be collected from such a mission.

5.2.1 Personal Remote Sensing Payload Requirements and Functionality

In an sUAS payload, software should exist separately from the autopilot for resilience reasons. If the payload software suffers a fault, the UAS should remain functional and continue flight in safe manner before landing. The payload should only collect data when necessary, to reduce energy consumption. To accomplish this, the payload must have a state machine that defines discrete mission states. The AggieCap Sensor State Machine (Fig. 5.1) is comprised of three main sections: Pre-Takeoff, Active State, and Post Landing Data Processing stage. In the Pre-Takeoff stage the payload is monitoring all of the sensors attached to it. If faults are detected an external signaling device such as a klaxon is activated, and the operators can locate and fix the error. This assures an sUAS with a faulty payload will not be flown. Once the sUAS has taken off, it transitions to the Active state. This state has two substates: Warm and Cool. In the Cool state, the payload is active, but is not yet capturing data. Bay doors are closed and sensors are suspended. From the Cool state the payload can transition to the Warm state, usually at some desired altitude above ground level. In the Warm state the payload is actively gathering data at preprogrammed intervals. During the transition, bay doors open and the sensors are prepped for data acquisition. While in the active state the payload can transition freely between Warm and Cool states. Before landing, all sensors are Cooled. After landing the

payload transitions to the Data Processing state. In this state the payload prepares all of the data for post-processing, data buffers are flushed and file handles are closed so the data can be retrieved correctly by the sUAS operators.

5.2.2 ISaAC: The Intelligent Safety and Airworthiness Co-Pilot

A PRS sUAS is able to fly and record data solely with an autopilot, payload module, and software (such as AggieCap). However, in the quest to deliver the most reliable and safe data over many flight missions and in various conditions, an sUAS can integrate knowledge about internal system status (such as autopilot and payload faults) and external data to the extent that it is known. The Intelligent Safety and Airworthiness Co-Pilot or ISaAC is designed to evaluate the safety of flight plan actions given up-to-date airspace information. ISaAC provides high levels of detail about sensor failures and autopilot status (such as a navigation data log) during flight, and provides a path for airspace-based safety oversight for systems requesting flight plan changes such as smart payloads.

Multiple subsystems including the ground station, payload, and autopilot, must be ready before a launch procedure can be initiated. For DMA, launch is only possible once the payload is fully ready to fly. Determining the status of the various payload parts is part of the functionality of ISaAC. Since the sensor hardware can be complex, and the software interface stack (drivers) can be equally complex, many different states of the various sensors are possible. In the AggieCap block diagram (seen in Fig. 5.2), the Sensors and their Trigger all report Go/Nogo (GnG) signals to ISaAC, allowing ISaAC to ascertain the overall

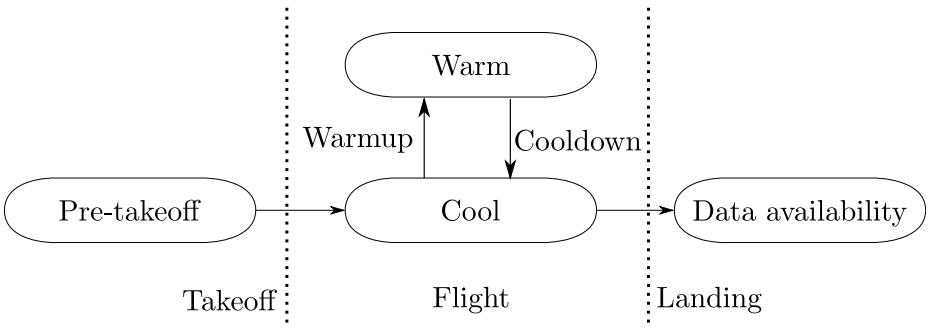


Fig. 5.1: Flight states for AggieCap sensors.

operational status of the payload. Only when all sensors report a "go" status will ISaAC show a "go" and indicate to the operators that the payload is ready for flight, allowing the rest of the flight operation to continue.

In the scope of this work, resilience is defined as keeping DMA at the highest possible level. Despite fragile system components such as CCOTS cameras, USB hubs, and other hardware, the AggieAir system is able to deliver DMA by handling component failures (for instance, a USB bus reset by a loose cable mitigated by detecting and restarting the Linux USB subsystem while in flight). Should more drastic actions be needed as defined by a pre-determined fault tree (such as a full payload kernel panic), ISaAC has control of the payload power via a MOSFET switching system, also seen in Fig. 5.2. ISaAC detects GnG signal timeouts or special requests from troubled payloads, and hard-resets payloads to a "known-good" system state. In this way, ISaAC increases DMA during flight/data collection in real time.

Seen in Fig. 2.4, ISaAC receives data from the navigation system about the current global position and attitude (pose) of the aircraft. This along with the data from AggieCap about the go status of the payload(s), allows for an overall picture of the success of the flight to be produced. Also, ISaAC allows a black box memory module (flash-based in reality) to record the payload and aircraft status during flight, and thereby allowing operators or investigators to reconstruct the faults that may have occurred after an airborne malfunction or crash, so improvements in safety and accountability are made possible.

ISaAC is implemented in practice on a Gumstix Overo [141] computer, connected into the airborne flight system as seen in Fig. 2.4, to AggieCap via Ethernet or local Unix sockets as needed. This allows for separation of the autopilot, the payload, and ISaAC itself. ISaAC is tasked with both recording the status of the AggieAir system (GnG and autopilot errors, etc.) and with interfacing AggieCap payloads with the autopilot. ISaAC is also responsible for sense-and-avoid behavior, allowing a camera system or other sensor to be integrated into the safety architecture and connected to the autopilot to achieve desired maneuvers.

Should ISaAC be given knowledge about the greater airspace by way of an Automatic Dependent Surveillance-Broadcast (ADS-B) receiver, or the most up-to-date equivalent [142], the global position and flight plan of the sUAS can be evaluated against the other airborne entities in the airspace, and cognitive path planning is made possible. This means payloads can autonomously request new sampling locations, and ISaAC will only allow the autopilot to move the airframe to the new location after evaluation of the safety factors of such a move. ISaAC therefore, increases overall safety of the aircraft and improves DMA by allowing better data to be requested and recorded by PRS payloads.

5.3 AggieCap Modular Software Architecture

PRS sUASs have a multitude of configurations for missions, and can be quickly reconfigured, even in the field. Enabling this functionality requires PRS payloads to be modular and readily tested and verified for a given mission, thus providing DMA as part of the payload development and testing process.

A modular software architecture allows for "plug and play" payload configuration after a given payload module is tested and available for integration. Modularity also allows for fast payload prototyping, giving system designers the ability to quickly demonstrate functionality. Abstraction of the sensor and actuator architectures is the key to achieving

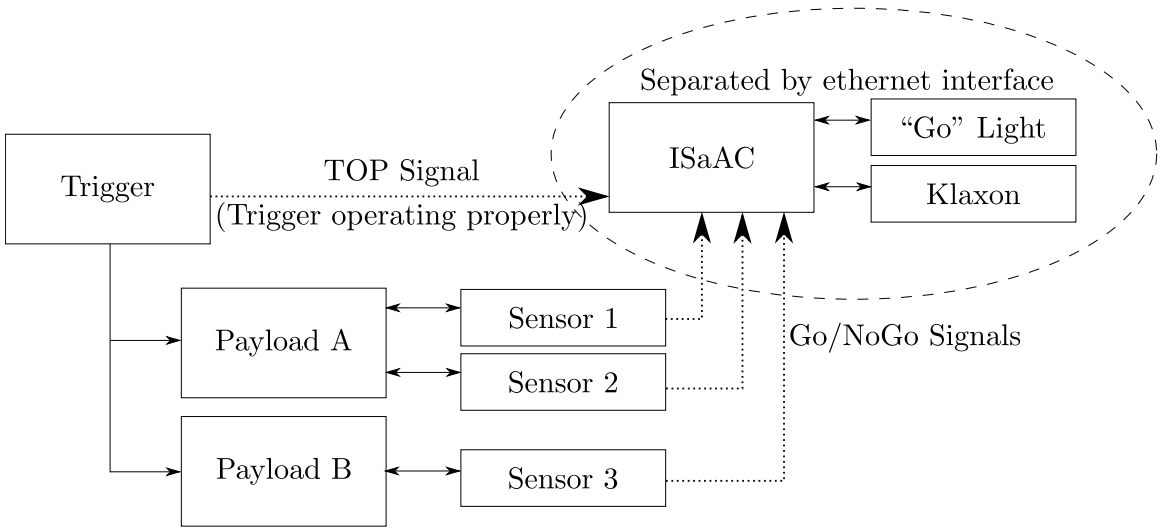


Fig. 5.2: AggieCap ISaAC data flow block digram.

modularity. Modularity and abstraction allow for code reuse and standardized testing, which increases the reliability of a payload and gives the best possible DMA, the overall goal of a PRS sUAS. Once a payload module is well tested, it can be added to future payloads with minimal testing overhead, and be verified more readily.

Small UASs for PRS make use of consumer-level COTS hardware to perform their remote sensing tasks, and fragility at many levels of the system is expected. Therefore, intelligent error logging is an important part of assuring functionality and performance. Modularity allows standardized error logging to be included automatically via inheritance in new payload designs, increasing re-use of tested payload components.

5.3.1 AggieCap Design

AggieCap is composed of three main parts: the sensors and actuators, the trigger, and the Go/NoGo logging interface. The sensors are responsible for gathering data and making logs of sensor status. Similarly the Actuators are responsible for controlling the physical actuation in the payload. The sensors are organized into payload objects which are then further inherited by the Trigger. The Trigger is responsible for timing all of the sensor captures. It also receives the platforms current attitude and position (pose) which is received from the autopilot. The Trigger associates this position data with each Sensor capture. The Trigger also sends the ISaAC regular messages ("TOP" as seen in Fig. 5.2) to verify operational status. The GnG Logger produces a log of all errors—including unhandled exceptions—that occur in the system. These logs are sent to the ISaAC system. This implementation of AggieCap is an excellent example of standardized testing protocols, facilitating resilience and data mission assurance.

5.3.2 AggieCap Implementation

Python was chosen as the AggieCap programing language for a variety of reasons. Maintainability, readability, platform independence, ease of prototyping, and the ability to link to shared object libraries.

Maintainability of code is highly linked to its readability. Python is a very high-level

language, and as such is easy to read and understand. This makes code maintainability more viable, and eases third-party code evaluation for more robust software. In addition, the lack of pointers in Python help avoid a common class of programming-related errors. Python is interpreted, so there is no need to recompile for different architectures. Since many sensors require precompiled driver interface code, Python allows linking and loading of shared libraries: precompiled C-code can still be used in AggieCap, provided the binaries exist for the target platform. Python has automatic garbage collection, which incurs a speed penalty compared to a compiled language. This is mostly irrelevant because AggieCap and payload control is IO bound, meaning the majority of runtime is spent interfacing with hardware devices. If there is a computationally expensive operation that must be performed in one of the sensors, the bottleneck can be alleviated in software implemented partially in C. This does not need to be done often, so the cross-platform benefits are retained for most applications. The inheritance hierarchy for AggieCap can be seen in Fig. 5.3.

5.4 Standardized Testing and Verification for DMA

Assurance of resilience is a challenging task, especially when remote sensing systems such as AggieAir are implemented with cost as a driving factor. This results in sensors being chosen that are more prone to faults. Consumer COTS frequently fail either due to

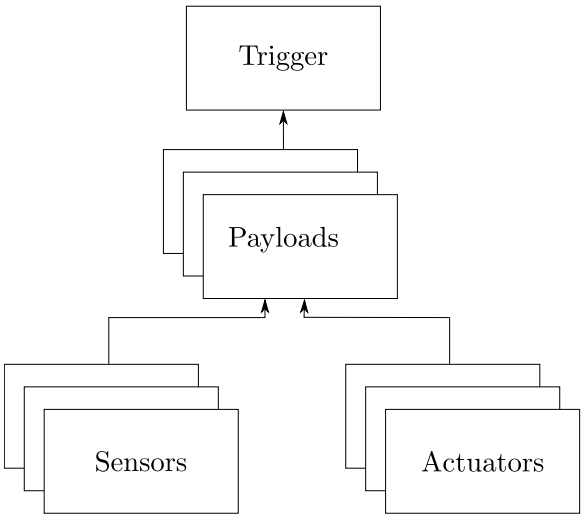


Fig. 5.3: AggieCap inheritance.

their inexpensive nature, or as is often the case, the requirement of an "unofficial" interface library such as gPhoto that is not supported by the manufacturer. To achieve data mission assurance with such a system, AggieCap is part of a standardized testing framework that allows payloads under development to undergo a payload-in-the-loop simulation of various mission parameters to prove experimentally their airworthiness and resilience during flight.

Standardized testing is a critical part of the process of testing and verifying a payload and ensuring functionality. Before a mission, the flight plan is constructed to allow the operators to train and prove the feasibility of the mission. During this testing, a stream of virtual navigation data is fed into the payload, and just as if a real flight was occurring, the payload will warm up and deploy, capture data, and land, all in the laboratory. Since the mission requirements are known in a general way during creation of payload modules, and more specifically after missions are chosen, it is possible to set up simulated missions for verification of payload modules in more and more specific scenarios during payload development.

This payload-in-the-loop style of testing helps assure data missions by revealing the great majority of software and hardware flaws before the payload modules are integrated into the aircraft. This means also that the aircraft need not be finished for the payload development, allowing efforts during development to progress in parallel. By using specific payload test data streams, the issue of fragility of consumer COTS sensors can be explored in a scientific closed environment. When coupled with the ISaAC framework described above, the interaction of various parts of the payload can be determined and verified as one system, with error logging allowing automated testing to proceed.

By implementing a suite of standardized tests, a payload verification rubric can be established. Starting at the physical levels: power usage, electro-magnetic interference and wiring problems can be diagnosed. Environmental factors such as humidity and temperature can be analyzed using test chambers designed for those specific purposes. Physical interfaces that might be prone to interruptions which can end the mission such as a USB hub disconnection during flight can be simulated, and the proper fault-tree-based response can

be tested. Given expected fault conditions, the logs produced from a payload module can be analyzed in real time to assure compliance with design specifications and performance in flight.

5.5 AggieCap Results

Table 5.1 contains a summary of all of the sensors implemented in AggieCap at the time of this writing. The variety of sensors shows how flexible the AggieCap architecture and system can be. Several implementations of the Actuator class show that AggieCap can be used and reused in many different configurations, and that once tested, the modularity of the payload components allow for new payloads to be built quickly out of well-tested software.

5.6 AggieCap Payload Implementation Example

PRS sUASs can be used for any number of civilian applications, such as natural resource management. For such a multispectral PRS mission (previously, thermal data was published

Table 5.1: AggieCap sensor and actuator list.

Hardware	Type	Interface
Nikon D90	DSLR camera (visible and NIR)	gPhoto
Canon T1i	DSLR camera (visible and NIR)	gPhoto
Canon s95	Small still camera (visible and NIR)	CHDK
Canon is110/115	Small still camera (visible and NIR)	gPhoto
Sensoray 2255	Video framegrabber	USB
FLIR Photon 320	Thermal IR camera	Sensoray
ICI7640	Digital thermal IR camera	USB
Lotek Nanotag Radio	Small animal radiotag locator	CSOIS internal
System	Computer hardware health	Linux System
Pololu Micro Maestro	6-channel servo controller and I/O	USB
Microswitch	Limit switches for motion control	Pololu Maestro
Standard servo motor	General servo actuator	Pololu Maestro
Firgelli linear actuator	Miniature actuator w/ position feedback	Pololu Maestro
Overo System LED	System I/O	Linux Kernel
Overo audio CODEC	Audible GnG Warning	Linux Kernel
Overo system analog inputs	System I/O	Linux Kernel

by Sheng [126]), AggieCap was used to create a payload (see block diagram in Fig. 5.4), consisting of an Infrared Cameras Incorporated model 7640 Thermal Infrared (TIR) camera [9], paired with a Canon s95 visible light camera. This payload also included an actuated sliding bay door, designed to protect the image sensors before takeoff and after landing. Images of this door in the Open and Closed position can be seen in Fig. 5.5. The payload enclosure was constructed from carbon-fiber and Kevlar weave, designed to be durable even during the stressful belly landings of the AggieAir system.

During this flight, 365 images were captured, on a cadence of 4 seconds with a total flight time of 37 minutes. These individual images were successfully "stitched" to form a combined image of the target area, seen in Fig. 1.4.

5.7 AERIS Significance of Payload Management and Chapter Summary

AERIS presents the idea of data as a mission, and equates mission assurance to payload resilience. Data mission assurance is introduced, along with a flexible, modular software architecture for DMA, as well as standardized tests and verification for such a system using consumer-level COTS hardware for PRS sUAS data collection missions such as the given example of TIR imaging for natural resource management.

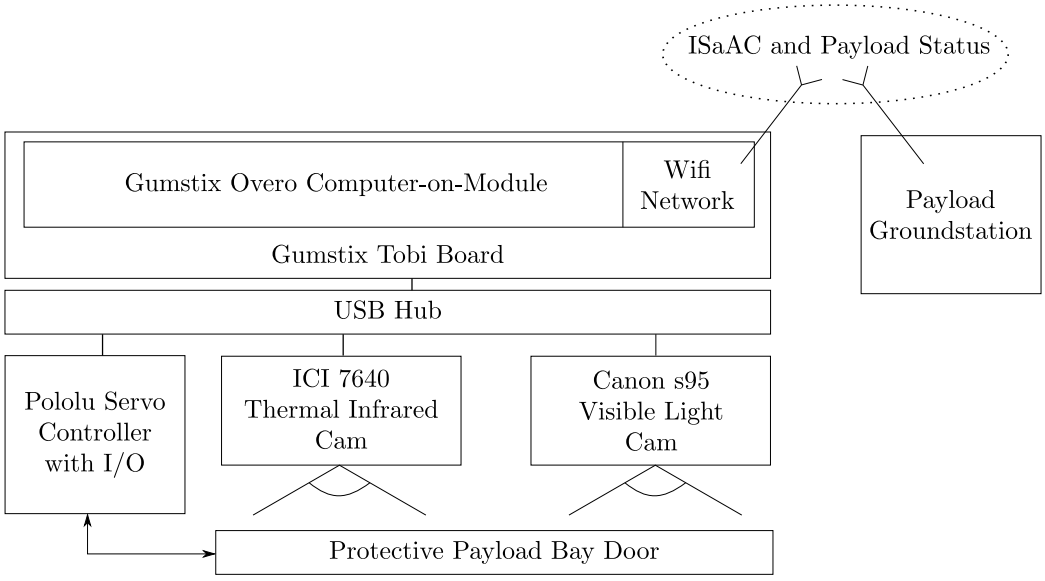
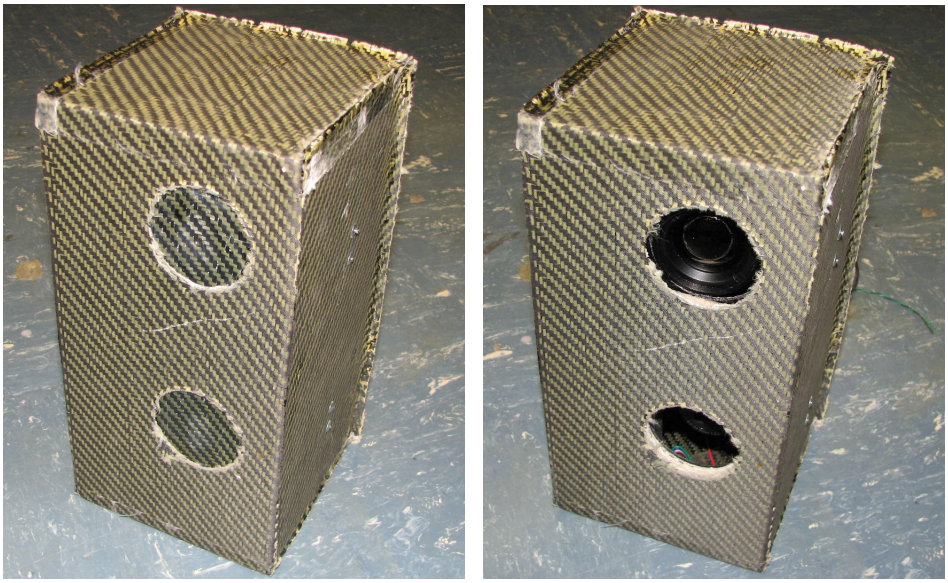


Fig. 5.4: AggieCap example payload block diagram.



((a)) Payload door closed. ((b)) Payload door open.

Fig. 5.5: Multispectral payload module ready for flight.

Future work will include a larger framework for holistically testing both airframe-and payload-in-the-loop during development, improvements to ISaAC allowing more cognitive processing of flight situational awareness to improve overall PRS missions, and increase data mission assurance.

Chapter 6

Small Unmanned Aerial System Navigation and a Fractional-Order Complementary Filter

To accomplish mission goals or to fly passengers to their destinations, all aircraft rely on sensors for the core of their navigational systems. During PRS flights, it is of critical importance to determine the attitude of the aircraft, for stable flight, and more importantly, for data quality. The knowledge of the states of the aircraft is used to post-process and merge raw collected data into contiguous, useful datasets such as maps. This process requires many different sensors (rate gyros, accelerometers, etc.) to be combined into a single set of states useful for flight and data. In AERIS, accurate information about the attitude of the aircraft is important for both safe flight in a variety of conditions, as well as payload date accuracy. Better navigational estimation means higher DMQ. This chapter is adapted from two publications, "A Comparative Evaluation of low-Cost IMUs for Unmanned Autonomous Systems" by Chao et al. [143] and "Fractional-Order Complementary Filters for Small Unmanned Aerial System Navigation" by Coopmans et al. [144].

An inertial measurement unit (IMU) is a device to measure the relative states of a static or mobile unit with respect to the inertial reference frame. Recently, many Micro Electro-Mechanical systems (MEMS) IMUs have emerged for under $300USD [145]. These low-cost IMUs can be used on unmanned vehicles for navigation [146], or can be combined with imaging sensors for georeferecing purposes. For example, the accurate orientation data is needed for the interpretation of the images from an airborne LIDAR radar. Actually, an accurate IMU accounts for a large portion of the total cost for an unmanned autonomous system [147]. The emergence of low-cost IMUs makes it possible to use more unmanned vehicles for agricultural or environmental applications like precision farming and real-time irrigation control [16, 148]. With the current trend of modularization and standardization

in the unmanned system design, the developers can either use an inexpensive commercial-off-the-shelf (COTS) IMU as a part of the navigation system, or develop their own system with low-cost inertial sensors.

In this study, low-cost IMUs are defined as those with the price around or less than $3000 USD. Low-cost MEMS IMUs are widely used on small or micro unmanned vehicles since they are small, light, yet still powerful. However, these lower-cost IMUs have bigger measurement errors or noise compared with expensive navigation grade or tactical grade IMUs [149]. It is challenging to design, test, and integrate these low-cost inertial sensors into a powerful IMU for navigation uses. More considerations for system design and sensor fusion algorithms need to be addressed to achieve autonomous navigation missions.

IMUs are usually used to measure the vehicle states like orientation, velocity, and position. The orientation measurement is especially important for missions requiring accurate navigation. However, the orientation is not directly measurable with the current COTS MEMS sensors. It has to be estimated from a set of correlated states like angular rates (gyros), linear accelerations (accelerometers), and magnetic fields (magentometers). Therefore, the estimation accuracy of IMUs heavily relies on the sensor fusion algorithm. Many researchers have looked into the state estimation problem using nonlinear filtering techniques [150]. Different kinds of Kalman filters are widely used in the aeronautics societies for spacecraft attitude estimations [151]. However, many of these algorithms are developed for highly accurate inertial sensors. Besides, those algorithms may have high demands for the computational power, which may not be possible for low-cost IMUs. The sensor fusion algorithms based on low-cost IMUs are focused in this chapter. A short survey of the current available state estimation filters for low-cost unmanned autonomous systems is provided with several representative examples like complementary filters [152], extended Kalman filters [153–155], and other nonlinear filters [156].

Kalman or Kalman-style combining filters are the established solution for combining sensor data into navigation-ready data. Extended Kalman filter approaches allow nonlinear models to be linearized and used with the Kalman techniques, however, they have proved

difficult to apply to sUASs with low-cost, high-noise sensors [157], and even EKF-based techniques are known to give unsatisfactory results [158]. Complementary filters are not mathematically rigorous like the Kalman filter, and therefore are not well-suited for high-risk applications like space missions. However, compared to Kalman filtering, complementary filter techniques are less computationally intensive and more readily performed on small, low-power processing hardware ideal for small, low-cost UASs.

Due to their inherent advantages, enriching complementary filters with new ideas may be beneficial. One paradigm which is gaining momentum in research is that of fractional calculus. The idea of fractional calculus has been known since the development of the regular calculus, with the first reference probably being associated with letter between Leibniz and L'Hospital in 1695. Fractional-order calculus is experiencing a resurgence as applications are found for the fractional calculus' more physical representation of dynamics of the real world.

The links between fractional–order calculus and stable power-law statistics are also well known [41]. Power-law statics are ubiquitous [159], and although Kalman-style power-law aware filters (such as the Kalman-Levy filter) have been considered in research, they are not well studied and have even higher mathematical complexity compared to standard Kalman techniques [160]. Complimentary filters work without assuming Gaussian noise statistics, however, and are applicable to a wider variety of sensor hardware like commonly-available MEMS devices with longer-tail noise tendencies [161]. Since complementary filters require less computational power, they couple well with low-cost MEMs sensors and microcontrollers. Fractional-order filters are being implemented in other mechatronic applications such as lithography [162] with good success.

6.1 IMU Basics and Notation

Most IMUs are employed for the measurement of the movements of a craft or a vehicle in 3D space. To describe the vehicle movements in 3D space, the coordinate frames are defined as follows, shown in Fig. 6.1 from Chao et al. [16]:

1. **Vehicle Body Frame.** F_{body}, the reference frame with the origin at the gravity center and the axes pointing forward, right and down.

2. **Inertial Navigation Frame.** F_{nav}, the reference frame with a specific ground origin and the axes pointing the North, East, and down to the Earth center.

3. **Earth-Centered Earth-Fixed (ECEF) Frame.** F_{ECEF}, the reference frame with the origin at the Earth center. The z axis passes through the north pole, the x axis passes through the equator at the prime meridian, and the y axis passes through the equator at 90° longitude.

Instead of making a direct measurement, IMUs rely on the sensor fusion algorithm to provide an accurate estimation of the system states. More precisely, the following states need extra estimation since no direct measurements are available or the update rate is not fast enough:

1. **Position.** The position information can greatly affect the georeferencing result (see Chapter 2). However, civilian GPS receivers can only provide measurements at 4-10 Hz or slower with the 3D accuracy of no less than three meters;

2. **Attitude.** The orientation information is very important for both flight control and image georeferencing;

3. **Velocity.** The ground velocity of the autonomous vehicle from GPS can not be updated fast enough for many applications.

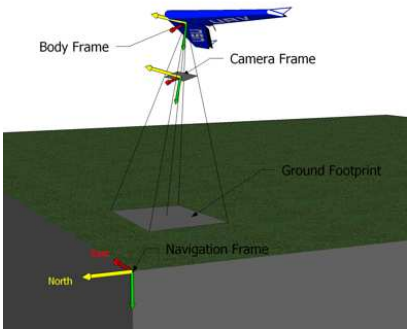


Fig. 6.1: Aircraft coordinate systems from Chao et al.

The available direct measurements for low-cost IMUs include:

1. **Position.** For example, longitude (p_e), latitude (p_n), altitude (h) (LLH) from GPS in 4-10 Hz or lower, the altitude or height can also be measured by pressure or ultrasonic sensors;

2. **Velocity.** Ground speed from GPS (v_n, v_e, v_d) and the air speed from pressure sensors;

3. **Rate gyro.** Angular velocity expressed in the body frame (p, q, r);

4. **Acceleration.** Linear acceleration expressed in the body frame (a_x, a_y, a_z).

The sensor fusion problem is defined as making an optimal estimation of the required vehicle states with the direct measurements from multiple sensors. This problem is also called a state estimation or a nonlinear filtering problem [150]. There are many possible solutions to this problem such as Kalman filters or complementary filters.

6.2 Sensor Packages

The developments of the low-cost MEMS inertial sensors can be traced back as early as 1970s [149]. In this section, the possible sensor packages for IMUs are introduced with an emphasis on the error models and the IMU categories.

1. **Gyro.** A gyro sensor is to measure the angular rate around the pre-specified axis observed from the earth coordinated in the body frame. Most manned or unmanned aircraft have three-axis gyros onboard. The gyro error model can be expressed as:

$$\hat{\omega} = (1 + s_g)\omega + b_g + \mu_g, \tag{6.1}$$

where $\hat{\omega}$ is the measurement value, s_g is the scale error, ω is the true value, b_g is the gyro bias, and μ_g is the random noise. Gyro sensors can be integrated to get the estimate of the angle. However, angle estimates based only on gyro data are heavily prone to drifting since any gyro bias is integrated over time.

2. **Accelerometer.** Accelerometers used on low-cost IMUs are to measure the linear acceleration. In fact, accelerometers measure the acceleration minus the gravity vector. For example, the default output of the accelerometer (static) is -1 when the axis is pointing down into the earth center. The accelerometer output can be expressed as:

$$\hat{a} = (1 + s_a)a + b_a + \mu_a, \tag{6.2}$$

where $\hat{a}$ is the measurement value, s_a is the scale error, a is the true value, b_a is the accelerometer bias, and μ_a is the random noise.

The accelerometer can also be used to measure the vehicle attitude since three-axis accelerometers can measure the gravity vector under the condition of zero acceleration. However, angle estimates from accelerometers suffer from high frequency noise when the unmanned vehicles are moving.

3. **Magnetometer.** Magnetometers are to measure the magnetic fields of the Earth, which can be approximated as an absolute value assuming the vehicle is not moving too fast. Three-axis magnetometer can be used for heading estimation and gyro bias compensation. One disadvantage of magnetometer sensors is that the hard-iron and soft-iron calibrations are needed for every vehicle.

4. **GPS.** GPS sensors can provide measurements of the absolute position, velocity, and course angle. The position packets can either be Latitude, Longitude, Height (LLH), or x, y, z expressed in the ECEF frame. The velocities include v_n, v_e, v_d, all with respect to the inertial frame. The course angle is defined as the angle relative to the north clockwise [163]. The GPS measurements have advantages of bounded errors, which can be used to reset the system error infrequently. The disadvantages of GPS include low update rate (<4 Hz mostly) and vulnerability to weather and terrain interference.

5. **Pressure Sensor.** Pressure sensors include absolute pressure sensors and relative pressure sensors. The former can be used to measure air pressure and to estimate the altitude. The latter can be used to measure air speed, which is especially useful to unmanned aerial vehicles.

   Based on the performance and the characteristics of the above sensors, the commercial inertial measurement units (IMUs) can be categorized into four types: navigation grade, tactical grade, industrial grade, and hobbyist grade. It is worth mentioning here that most of the industrial grade and hobbyist grade IMUs use MEMS inertial on-chip sensors, which greatly reduce the unit sizes and weights. The brief specifications are shown in Table 6.1. It can be seen that low-cost IMUs mostly fall into the industrial grade or the hobbyist grade due to their low cost and bigger errors compared with navigation or tactical grade IMUs.

6.3 Attitude Estimation Algorithms

Given the above sensor packages, an efficient sensor fusion algorithm is needed for the optimal estimation of the vehicle attitude. Extended Kalman filters are frequently used in nonlinear estimation problems, especially attitude estimation problems of rigid bodies like a spacecraft or an aircraft. The extended Kalman filter can recursively estimate the system states from system measurements corrupted with Gaussian noises. It has advantages here

Table 6.1: IMU categories.

IMU Type	Navigation Grade	Tactical Grade	Industrial Grade	Hobbyist Grade
Cost ($)	$>$ 50k	10-20k	0.5-3k	$<$ 500
Weight	$>$ 5 lb	about 1 lb	$<$ 5 oz	
Gyro Bias	$<$ 0.1 deg/h	0.1-10 deg/h	$\leq$ 1 deg/sec	$>$ 1 deg/sec
Gyro Random Walk Error	$<$ 0.005 deg/$\sqrt{h}$	0.2-0.5 deg/$\sqrt{h}$		
Accel Bias	5-10 μg	0.02-0.04 mg		
Example	Honeywell HG9848	Honeywell HG1900	Microstrain GX2	ArduIMU

since both gyro and accelerometer sensors have drifts and Gaussian-like noises. The general extended Kalman filter and three representative approaches for the attitude estimation problem are introduced in the following sections.

6.3.1 General Extended Kalman Filter

Assume that a general nonlinear discrete-time system can be modeled as follows:

$$x_k = f(x_{k-1}, u_k) + w_k, \quad s.t. \quad w \sim N(0, Q), \tag{6.3}$$

$$y_k = g(x_k) + v_k, \quad s.t. \quad v \sim N(0, R), \tag{6.4}$$

where x_k is a m×1 vector, y_k is a n×1 vector, $f(x_{k-1}, u_k)$ and $g(x_k)$ are nonlinear functions. The first equation is called the propagation equation and the second one is called the measurement equation.

Given the initial values P_0, the measurement covariance R and the state covariance Q, the optimal Kalman estimate of the states can be updated using the following steps [164]:

1. State estimation extrapolation: $\hat{x}_{k|k-1} = f(x_{k-\hat{1}|k-1}, u_k)$,
2. Error covariance extrapolation: $P_{k|k-1} = F_k P_{k-1|k-1} F_k^T + Q_k$,
3. Kalman gain: $K_k = P_{k|k-1} G_k^T (G_k P_{k|k-1} G_k^T + R_k)^{-1}$,
4. State estimate update: $\hat{x}_{k|k} = x_{k|\hat{k}-1} + K_k(y_k - g(x_{k|\hat{k}-1}))$,
5. Error covariance update: $P_{k|k} = (I - K_k G_k) P_{k|k-1}$.

The F_k is the Jacobian matrix of $f(x_{k-1}, u_k)$, and the G_k is the Jacobian matrix of $g(x_k)$.

6.3.2 Quaternion-Based Extended Kalman Filter

Unit quaternions have many applications in state estimation problems because their simplicity. A quaternion-based extended Kalman filter was proposed originally for the MNAV IMU from Crossbow Technology [155]. UAV developers have used this approach on their specific platform [165]. The system state variables include both the unit quaternion

(q) and the gyro biases (b_p ,b_q, b_r). The measurements or observation of the system are the accelerations: a_x ,a_y, a_z, and the yaw angle ψ derived from the magnetometer. The propagation equation and the measurement equation are listed below [155].

$$\dot{q} = \frac{1}{2}\left[\begin{array}{ccccccc} 0 & -\hat{p} & -\hat{q} & -\hat{r} & 0 & 0 & 0 \\ \hat{p} & 0 & \hat{r} & -\hat{q} & 0 & 0 & 0 \\ \hat{q} & -\hat{r} & 0 & \hat{p} & 0 & 0 & 0 \\ \hat{r} & \hat{q} & -\hat{p} & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] q + w_k, \tag{6.5}$$

$$q = \left[\begin{array}{c} q_0 \\ q_1 \\ q_2 \\ q_3 \\ \hline b_p \\ b_q \\ b_r \end{array}\right], \left[\begin{array}{c} \hat{p} \\ \hat{q} \\ \hat{r} \end{array}\right] = \left[\begin{array}{c} p \\ q \\ r \end{array}\right] - \left[\begin{array}{c} b_p \\ b_q \\ b_r \end{array}\right], \tag{6.6}$$

$$\left[\begin{array}{c} a_x \\ a_y \\ a_z \\ \psi \end{array}\right] = \left[\begin{array}{c} 2g(q_1q_3 - q_0q_2) \\ 2g(q_2q_3 + q_0q_1) \\ g(q_0^2 - q_1^2 - q_2^2 + q_3^2) \\ \tan^{-1}\frac{2(q_1q_2+q_3q_0)}{q_0^2+q_1^2-q_2^2-q_3^2} \end{array}\right] + v_k, \tag{6.7}$$

where $w_k \sim N(0, Q)$, $v_k \sim N(0, R)$.

It is worth pointing out that the measurement equation has an assumption that the acceleration measured is only the projection of the gravity vector. However, this assumption may not be true for small UAVs.

6.3.3 Euler Angle-Based Extended Kalman Filter

Euler angles can also be chosen as the system states for the attitude estimation problem. Assuming that the system state is a vector x (representing the roll and pitch angle) and the system output is a vector $\hat{y}$ (representing the accelerometer readings), the system can then be modeled [166]:

$$x = \begin{bmatrix} \phi \\ \theta \end{bmatrix}, \qquad \hat{y} = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix}, \tag{6.8}$$

$$\dot{x} = \begin{bmatrix} p + q\sin\phi\tan\theta + r\cos\phi\tan\theta \\ q\cos\phi - r\sin\phi \end{bmatrix} + v_w, \tag{6.9}$$

$$\hat{y} = \begin{bmatrix} \dot{u} - rv + qw - g\sin\theta \\ \dot{v} + ru - pw - g\cos\theta\sin\phi \\ \dot{w} - qu + pv - g\cos\theta\cos\phi \end{bmatrix} + v_k, \tag{6.10}$$

where $v_w \sim N(0, Q)$, $v_k \sim N(0, R)$.

In fact, the velocities (u, v, w) are not easily measurable at high frequencies. The air speed can be used instead to simplify the measurement equation for small fixed-wing UAVs. The following assumptions can be made [166]:

1. $\dot{u} = \dot{v} = \dot{w} = 0$. The small UAV will not accelerate all the time;
2. $v = 0$. The small fixed-wing UAV will not go sideways;
3. $u = v_a\cos\theta$, $w = v_a\sin\theta$;

where V_a is the air speed measured by a pitot tube [153]. The measurement equation can then be simplified [166].

$$\hat{y} = \begin{bmatrix} qV_a\sin\theta + g\sin\theta \\ rV_a\cos\theta - pV_a\sin\theta - g\cos\theta\sin\phi \\ -qV_a\cos\theta - g\cos\theta\cos\phi \end{bmatrix} + v_k, \tag{6.11}$$

where $v_k \sim N(0, R)$.

The attitude can then be estimated using the extended Kalman filter following the steps described in the above section.

6.3.4 AggieEKF: GPS Aided Extended Kalman Filter

AggieEKF, a GPS aided extended Kalman filter is proposed in this section with considerations from both filters designed in the above sections. An extended Kalman filter similar to Jang and Liccardo [155] is developed. However, the measurement equation is replaced by a more accurate estimation of the gravity vector with the help from the GPS speed measurements. The system equations are shown as below where V_g is the ground speed measured by the GPS.

$$\dot{q} = \frac{1}{2}\left[\begin{array}{ccccccc} 0 & -\hat{p} & -\hat{q} & -\hat{r} & 0 & 0 & 0 \\ \hat{p} & 0 & \hat{r} & -\hat{q} & 0 & 0 & 0 \\ \hat{q} & -\hat{r} & 0 & \hat{p} & 0 & 0 & 0 \\ \hat{r} & \hat{q} & -\hat{p} & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] q + w_k, \tag{6.12}$$

$$\begin{bmatrix} \hat{a}_x \\ \hat{a}_y \\ \hat{a}_z \end{bmatrix} = \begin{bmatrix} 2g(q_1q_3 - q_0q_2) \\ 2g(q_2q_3 + q_0q_1) \\ g(q_0^2 - q_1^2 - q_2^2 + q_3^2) \end{bmatrix} + v_k, \tag{6.13}$$

$$\text{where} \begin{bmatrix} \hat{a}_x \\ \hat{a}_y \\ \hat{a}_z \end{bmatrix} = \begin{bmatrix} a_x \\ a_y - rV_g + g\sin\phi_{t-1} \\ a_z + qV_g + g\cos\phi_{t-1} \end{bmatrix}, \tag{6.14}$$

and $v_k \sim N(0, R)$. The attitude state estimation can be calculated using the steps described in the above section.

6.3.5 Complementary Filters

Aside of the Kalman filter approaches, there are other nonlinear filters like complementary filters. One example complementary filter is shown below [167]:

$$\dot{\hat{R}} = [(R\Omega)_\times + k_{est}\pi_a(\tilde{R})\hat{R}^T]\hat{R}, \tag{6.15}$$

$$\pi_a(\tilde{R}) = \frac{1}{2}(\tilde{R} - \tilde{R}^T), \quad \tilde{R} = \hat{R}^T R, \tag{6.16}$$

where Ω is the body angular velocity, R is the rotation matrix estimated from accelerometers, and k_{est} is the gain to tune. The key idea here is to fuse the estimation from gyros and from accelerometers [152].

6.4 Example Low-Cost IMUs

Several example low-cost IMUs are compared in this section focusing on hardware sensors, estimation accuracy, and software modification ease. The detailed hardware specifications are compared in the end of this section.

6.4.1 Attitude Estimation IMUs

The Microstrain 3DM-GX2 IMU has typical inertial sensor sets including 3-axis gyros, 3-axis accelerometers, and 3-axis magnetometers [168]. The 3DM-GX2 IMU can output the angle estimations in either Euler angles or rotational matrix at up to 250 Hz and the sensor bandwidth is 100 Hz. The 3DM-GX2 IMU can be easily connected with other units through RS232/422 or USB interfaces. Another advantage of The 3DM-GX2 IMU (Fig. 6.2) is its resistance to shock interferences of up to 500g when powered. There are also similar IMUs like the 3DM-GX3 from Microstrain and VN100 from VectorNav Technologies [169].

6.4.2 GPS-Coupled IMUs

The GPS/inertial navigation system could be only one board or two separate units. Unfortunately, the GPS/INS integrated unit with the built-in sensor fusion algorithm is very expensive. The MTi-G IMU from Xsens has both attitude and position estimations at

Fig. 6.2: Microstrain GX2 IMU.

up to 120Hz, shown in Fig. 6.3 from Xsens documentation [170]. However, this IMU costs a great deal more than $3000, and is just shown here for comparison reasons.

A GPS-coupled IMU can also be combined with two isolated units (GPS and IMU) and a central processor. AggieNav was designed and implemented at the Center for Self-Organizing and Intelligent Systems (CSOIS) at Utah State University following this idea [139]. Based around Analog Devices' ADIS1635X 6-DOF IMU part, AggieNav includes a GPS unit, a magnetic compass, as well as pressure sensors for airspeed measurement. AggieNav also has a Gumstix Verdex Pro (Fig. 6.4) or Overo connected for heavy computations like further processing of the sensor data, and greater control over other aspects of the UAV mission.

6.4.3 Hobbyist Grade IMUs

Recently, several hobbyist grade IMUs have become available with "flat" three-axis gyros and accelerometers. Hobbyist grade IMUs can also be used by the hobbyist for the

Fig. 6.3: Xsens Mti-g IMU from Xsens' documentation.

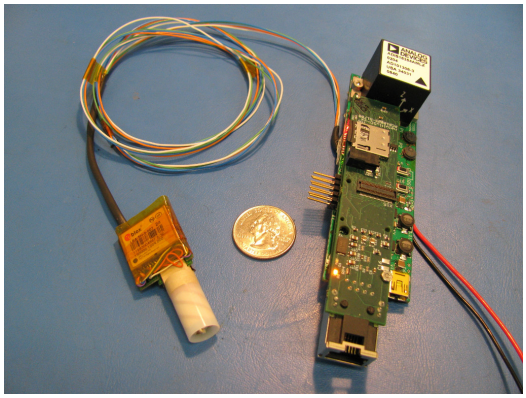


Fig. 6.4: AggieNav IMU.

navigation mission of easy-configured UAVs [145].

Ardu-IMU is one representative IMU made from an open source project, shown in Fig. 6.5. It uses complementary filters derived from Mahony's work [146]. It can output attitude estimates up to 50 Hz.

Sparkfun Razor IMU is another representative IMU made from "flat" three-axis sensors [171], shown in Fig. 6.6.

6.4.4 IMU Comparisons

A detailed comparison of the specifications for all the IMUs mentioned above is provided in Table 6.2. It is worth mentioning that several low-cost IMUs like Ardu IMU are using single-chip dual axis gyro sensors which is convenient for micro and small unmanned vehicles.

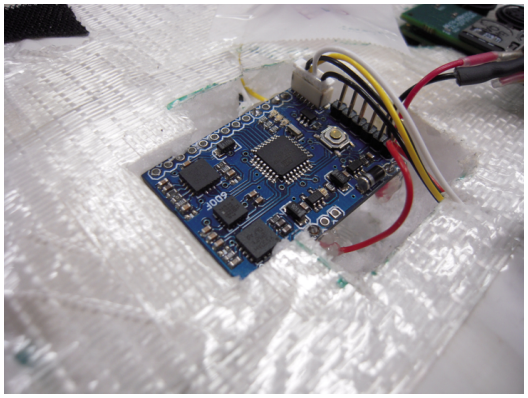


Fig. 6.5: Ardu-IMU.

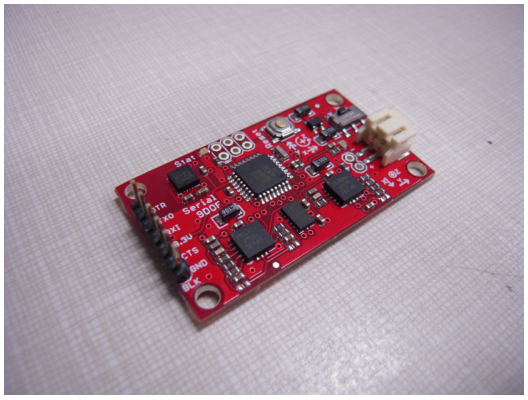


Fig. 6.6: Sparkfun Razor IMU.

Table 6.2: IMU comparisons.

Specifications	Microstrain 3DM-GX2	Microstrain 3DM-GX3	Xsens Mti-g (w. GPS)	ADIS16405	Ardu-IMU	VectorNAV VN-100
Size (mm)	$41 \times 63 \times 32$	$44 \times 25 \times 11$	$58 \times 58 \times 33$	$23 \times 23 \times 23$	$28 \times 39 \times 12$	$75 \times 75 \times 17$
Wight (g)	39	18	68	16	6	36
Orientation Accuracy (static)	±0.5° typical	±0.5° typical	< 0.5°(roll/pitch)	Only raw data	N/A	< 0.5°(roll/pitch)
(dynamic)	±2.0° typical	±2.0° typical	1° RMS			< 2.0° typical
Update Rate	$< 100Hz$	$< 1000Hz$	$< 120Hz$	$< 330Hz$	$< 50Hz$	$< 200Hz$
Gyro Bias (°/sec)	± 0.2	±0.2	1	±3(initial) 0.007 (in-run)	N/A	<0.028 ($25°C$)
Gyro Range(default)	±300°/sec	±300°/sec	±300°/sec	±300°/sec	±300°/sec	±500°/sec
Gyro Nonlinearity	0.2%	0.2%	0.1%	0.1%	1%	< 1%
Gyro Random Walk Error	N/A	N/A	0.05°/sec/$\sqrt{}$Hz		N/A	N/A
Accel Bias	±0.005g	±0.005g	±0.002g	±0.05g(initial) 0.0002g(in-run)	N/A	±0.0005g(X,Y) ±0.0016g(Z)
Accel Range(default)	±5g	±5g	±50m/s^2	±18g	±3.6g	±2g
Accel Nonlinearity	0.2%	0.2%	0.2%	0.1%	0.3%	< 0.5%
Mag Bias (Gauss)	±0.01	±0.01	0.0001	±0.004	N/A	±0.000125
Mag Range (Gauss)	±2.5	±1.2	±0.75	±2.5	N/A	±6
Mag Nonlinearity	0.4%	0.4%	0.2%	0.5%	N/A	< 1%

6.4.5 Sensors and Navigation

For small UAVs targeted toward remote sensing, the navigation sensor suite determines much about the usefulness and capabilities of such a system, and is mostly limited because of the relatively high cost of high-quality navigational sensors and equipment.

Although technology is always improving, the options for high-quality, low-cost miniature attitude navigation sensors are few; many inexpensive, small fixed-wing UAVs use infrared thermopiles to sense the relative heat differential from the black body radiation of the Earth's surface. While this technique does work, it is not accurate enough for some remote measurement tasks and has problems when flying near large land features such as

mountains and through valleys. Combined with a modern civilian GPS receiver, navigation is possible with these sensors, but applications in acquiring scientific data requires higher accuracy in attitude estimation.

Inertial measurement units with roll-rates and accelerometers are the standard sensor for attitude determination in flight. Several integrated solutions exist for navigation in small UASs, most of which use a Kalman or other state-estimation filter of some kind to provide a high-accuracy estimation of the current position and attitude of the system at a given instant. While these systems work well for their intended applications, even the academic prices are prohibitive for integration into inexpensive UAVs. Additionally, the source code is not made available, and users of the systems are forced to accept the filter/algorithm performance of the systems as they are given.

In any navigation system, performance, and reliability are primary design drivers. In the world of small UAVs, size, weight, and price are also of very high importance. Recently, due to the high-availability of low-cost sensing chips, many consumer-level devices such as Apple's iPhones, Nintendo Wii, and Sony PlayStation 3 have integrated accelerometers into their hardware for more immersive user-experiences. Fully integrated 6-degree-of-freedom chips are being produced by Invensense, and other companies who target the consumer motion interface segments of the market. These sensors work well for static human motion applications, and not as well for dynamic applications such as aerial robotics.

Many organizations are producing IMUs for low-cost hobbyist-level (i.e., under $200USD) UAV flight such as Sparkfun, DIYDrones, etc. [143] based on complementary filters. These products are good enough for flight (and basic data collection in some cases), but with current data filtering techniques it is not possible to derive high-quality data from inexpensive sensor systems due to nonlinearities, relatively poor noise performance, and poor temperature calibration [172].

6.4.6 Kalman Filtering for Small UASs

Estimating the orientation of the aircraft is a very important part of a UAS. Commonly a Kalman filter is used [154, 157, 173–175] and combines information from multiple sensors and models to estimate the orientation and variance. Kalman filters are model-based and require system models to propagate state estimates. When the Kalman Filter receives measurements, it uses those measurements and their models to update state estimates. One requirement of the Standard Kalman Filter is that these models must be linear. The Extended Kalman Filter does not share this same requirement and can be used when the models are nonlinear. However the Extended Kalman Filter linearizes the models in the operation of the filter and assumes that the state estimates will not deviate far from the nominal solution. Knowledge about the noise attached to these models is also necessary to design a Kalman Filter, and most Kalman Filters assume that this noise is Gaussian. These assumptions make Kalman filters difficult to use with with small, low-quality inertial sensors due to the nonlinearity and non-Gaussian characteristics of these sensors [167, 176]. In these cases, complementary filters are often used [158, 177, 178] because no assumptions are made about linearity and noise statistics [179].

6.4.7 Complementary Filters

The basic idea of a complementary filter is to take two or more sensors, filter out the unreliable frequencies for each sensor, and combine the filtered outputs to get a better estimate throughout the entire bandwidth of the system. To achieve this, the sensors included in the filter should complement each other by performing better over different parts of the system bandwidth than the other sensors. For example, one sensor could have reliable data at high frequencies and unreliable data at low frequencies. To complement this sensor, one or more sensors should be added to the filter that have reliable low frequency data. An example of two sensors that complement each other are an accelerometer and gyro. The gyro is reliable at high frequencies while the accelerometer is reliable at lower frequencies [167]. The complementary filter filters out the unreliable parts of the frequency for each sensor and combines their output. Figure 6.7 shows a diagram of a simple complementary filter

with two sensors.

Complementary filters were first introduced in a patent in 1951 [180]. The patent describes an automated steering system for keeping a manned aircraft on a proper flight path during landing using the Instrument Landing Approach System (ILAS) along with a gyro-stabilized accelerometer. The ILAS was used to measure the rate of change between the distance from the aircraft and the desired path, and the gyro-stabilized accelerometer was used to measure changes in heading. The ILAS had reliable low frequency data but unreliable high frequency data, while the accelerometer had reliable high frequency data. Therefore, a complementary filter was used to combine this data, although the term "complementary filter" was not used. This term was coined in a journal paper in 1953 that describes a similar automated steering system [181]. These early complementary filters were designed in the frequency domain and realized with simple RC circuits.

Many other applications for the Frequency Domain Complementary Filter (FDCF) were presented after their introduction. Shaw and Srinivasan [182] used a complementary filter to combine low frequency position data from strain sensors and high frequency data from an accelerometer to estimate the position of the end of a beam. Zimmermann and Sulzer [183] used low frequency data from an inclinometer and high frequency accelerometer data to measure orientation. Baerveldt and Klang [177] introduced a similar system to measure orientation for an autonomous helicopter with an inclinometer and a gyro. Roberts

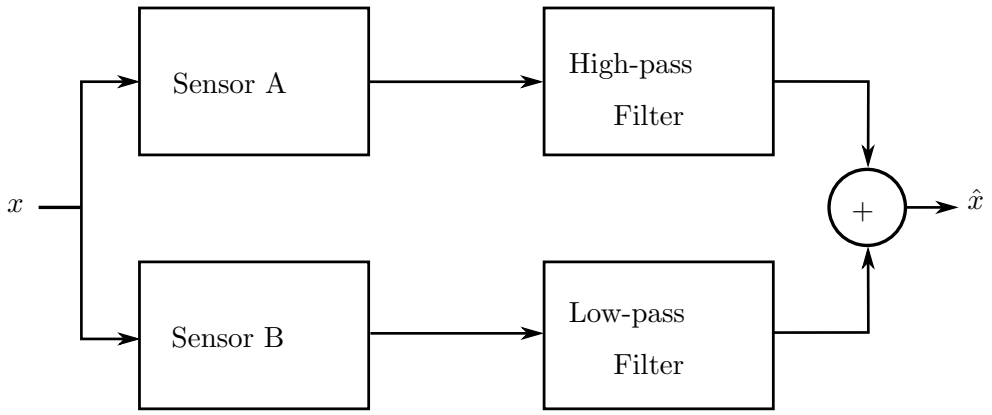


Fig. 6.7: Simple complementary filter diagram.

et al. [158] used an accelerometer and a gyro to measure 3D orientation. For the simplified purposes of this chapter, accelerometers and inclinometers are considered to be equivalent.

Eventually researchers began to design complementary filters in state space due to their advantages with multiple-input multiple-output systems, realization in digital systems, and closed-loop control designs. A common application of the State Space Complementary Filter (SSCF) is to estimate the orientation of a rigid body. Bachmann et al. [184–186] used such a complementary filter based on quaternions to track the orientation of robotic flexible links. Quaternions are used to represent orientation because they represent the axis and the rotation angle directly without singularities [187]. Instead of using quaternions to represent orientation, Mahony et al. [167] introduced a direct and passive complementary filter using a rotation matrix in the special orthogonal group, SO(3) (also referred to as a direct cosine matrix (DCM) [188], or attitude matrix [187]). Although the design of the filter was carried out using the SO(3), Mahony et al. [152] recommended that realization be done with quaternions. Hamel and Mahony [178] introduced another type of complementary filter based on SO(3), called the explicit complementary filter.

6.5 Frequency Domain Complementary Filters (FDCFs)

To illustrate how a frequency-domain complementary filter works, assume that two sensors complement each other and can be used to measure a signal x. Both of these sensors can be modeled and have transfer functions $H_1(s)$ and $H_2(s)$. According to the complementary filter design, the unreliable frequencies of each sensor are filtered out by passing the outputs through filters $G_1(s)$ and $G_2(s)$. Figure 6.8 shows a block diagram of this system [177, 189].

The objective of the estimator is to find an estimate of x ($\hat{x}$) such that $\hat{x} = x$. This is possible, according to the block diagram in Fig. 6.8, if the transfer functions $G_1(s)$ and $G_2(s)$ are designed according to the following equation:

$$H_1(s)G_1(s) + H_2(s)G_2(s) = 1, \quad \forall s. \tag{6.17}$$

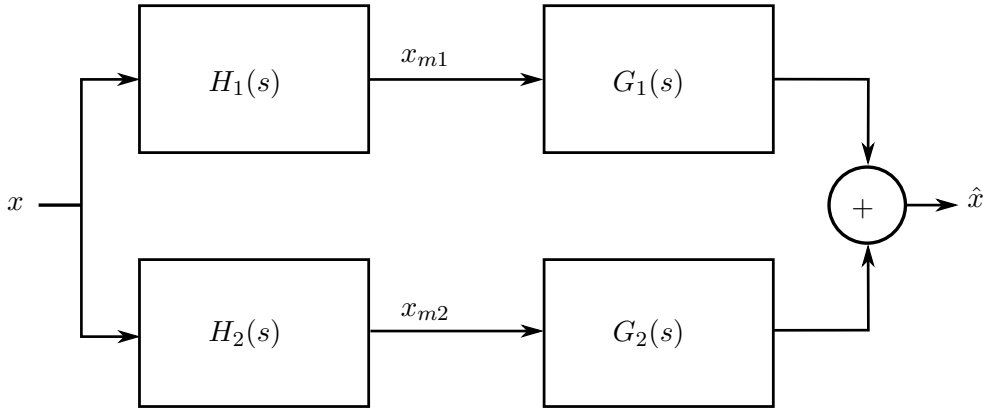


Fig. 6.8: Frequency domain complementary filter.

If $H_1(s)$ and $H_2(s)$ are known, they can be used with Eq. (6.17) to find $G_1(s)$ and $G_2(s)$. If $H_1(s)$ and $H_2(s)$ are not known, a modeless approach can be used which assumes that $H_1(s)$ and $H_2(s)$ are both ideal ($H_1(s) = H_2(s) = 1$). Instead of specifically designing $G_1(s)$ and $G_2(s)$ based on the sensor models, $G_1(s)$ and $G_2(s)$ are set as generic filters and tuned to minimize the error ($e = x - \hat{x}$).

With $H_1(s) = H_2(s) = 1$, Eq. (6.17) becomes the following:

$$G_1(s) + G_2(s) = 1, \quad \forall s, \tag{6.18}$$

and is used as a constraint when choosing filters for $G_1(s)$ and $G_2(s)$.

While using the modeless approach, one might choose the second order filters in Eq. (6.19) and Eq. (6.20) for $G_1(s)$ and $G_2(s)$. Since ω_0 determines where the filters are cut-off, it is used as a tuning parameter. Figure 6.9 shows a Bode plot of $G_1(s)$ and $G_2(s)$ with $\omega_0 = 1$. If ω_0 is reduced, more of $H_1(s)$ is filtered out and more of $H_2(s)$ is allowed to pass through ($H_2(s)$ dominant). If ω_0 is increased, more of $H_2(s)$ is filtered out and more of $H_1(s)$ is allowed to pass through ($H_1(s)$ dominant). A well-designed filter will have a value for ω_0 such that the best parts of both $H_1(s)$ and $H_2(s)$ are allowed to passed through while the unreliable parts are filtered out.

$$G_1(s) = \frac{2\omega_0 s + \omega_0^2}{(s + \omega_0)^2} \tag{6.19}$$

$$G_2(s) = \frac{s^2}{(s+\omega_0)^2} \tag{6.20}$$

6.5.1 FDCF Example – Modeless Second Order Filter

A modeless FDCF with second order filters can be used to combine the outputs of an inclinometer and a gyroscope to measure orientation. An inclinometer is a general term given to devices that directly measure orientation; one example is an accelerometer. The accelerometer measures orientation by the acceleration due to gravity. In general, inclinometers have a slow response time and will filter out high frequency movement. A gyro measures angular rate, which is converted to orientation by integration. While the gyro does well at high frequencies, the inherent noise and integration introduces drift. The complementary filter improves the orientation estimate by combining the reliable high frequency data from the gyro and the low frequency data from the inclinometer.

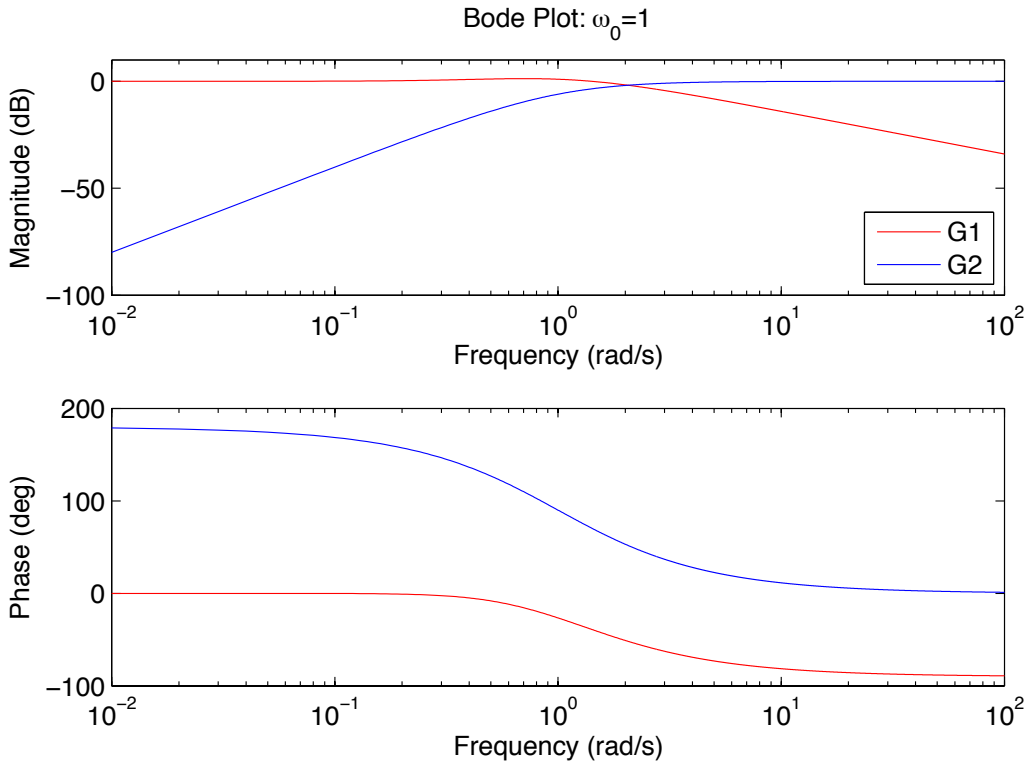


Fig. 6.9: Bode plots of $G_1(s)$ and $G_2(s)$.

Simulink is used to test the Second Order FDCF with the inclinometer and gyro. As shown in the simulink model in Fig. 6.10, the inclinometer is modeled as the transfer function in Eq. (6.21) [177]. The gyro is modeled as the derivative of the actual orientation plus a bias. Noise is also added to the gyro before the signal is integrated to get orientation. The output of the sensors are then passed through the filters from Eq. (6.19) and Eq. (6.20) and combined to get the orientation estimate.

$$H_1(s) = \frac{1.88}{s + 1.88} \tag{6.21}$$

The input orientation to the simulation is a chirp signal that produces a sine wave between ± 10 degrees at frequencies between 0.05Hz and 1Hz. At the beginning of the simulation, the sine wave will start at 1Hz and gradually decrease to 0.05Hz after 100 seconds. At that point, the frequency will gradually increase until the end of the simulation.

Figure 6.11 and Fig. 6.12 show the response of the inclinometer and gyro to the input chirp signal. As expected, the inclinometer cannot track the input at high frequencies. However, as the frequency is reduced, the inclinometer is more capable of tracking the sine wave. The output of the gyro is also expected; it tracks the input well but drifts away from

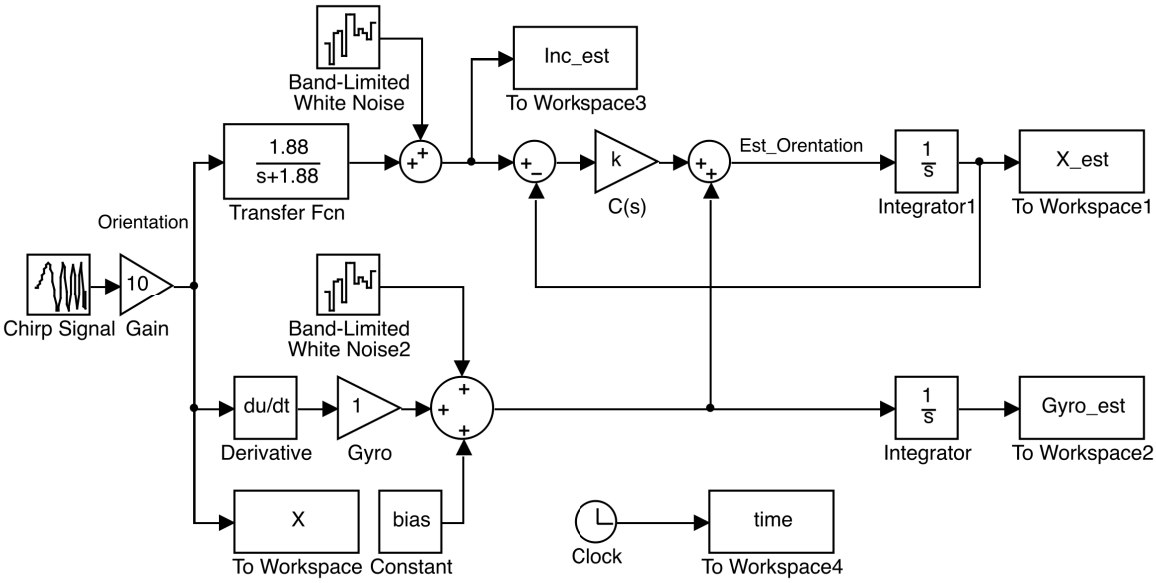


Fig. 6.10: Simulink model of modeless complementary filter.

the actual orientation at 1 degree per second.

The filter is tuned by finding the optimal value of ω_0 that minimizes the error e. This is done by running multiple simulations with varying values of ω_0 (0.01 rad/s to 0.5 rad/s). At the end of each simulation, the squared error and the variance are calculated for each value of ω_0. Figure 6.13 shows the result of these simulations. The lowest variance is $\sigma^2 = 2.23$ with an ω_0 of 0.22 rad/s. Figure 6.14 and Fig. 6.15 show plots of the filter output (estimated orientation) and the squared error with this optimal frequency.

While exploring the concept of a complementary filter, it is also worthwhile to look at the filter output at non-optimal frequencies. As Fig. 6.13 explains, if ω_0 is too high, the inclinometer will dominate the output of the filter. If ω_0 is too low, the gyro will dominate the output of the filter. Figure 6.16 and Fig. 6.17 show the filter output and error plots when ω_0 is too high and the output is dominated by the inclinometer. Notice how similar the output plot matches the output of the inclinometer in Fig. 6.11. Figure 6.18 and Fig. 6.19

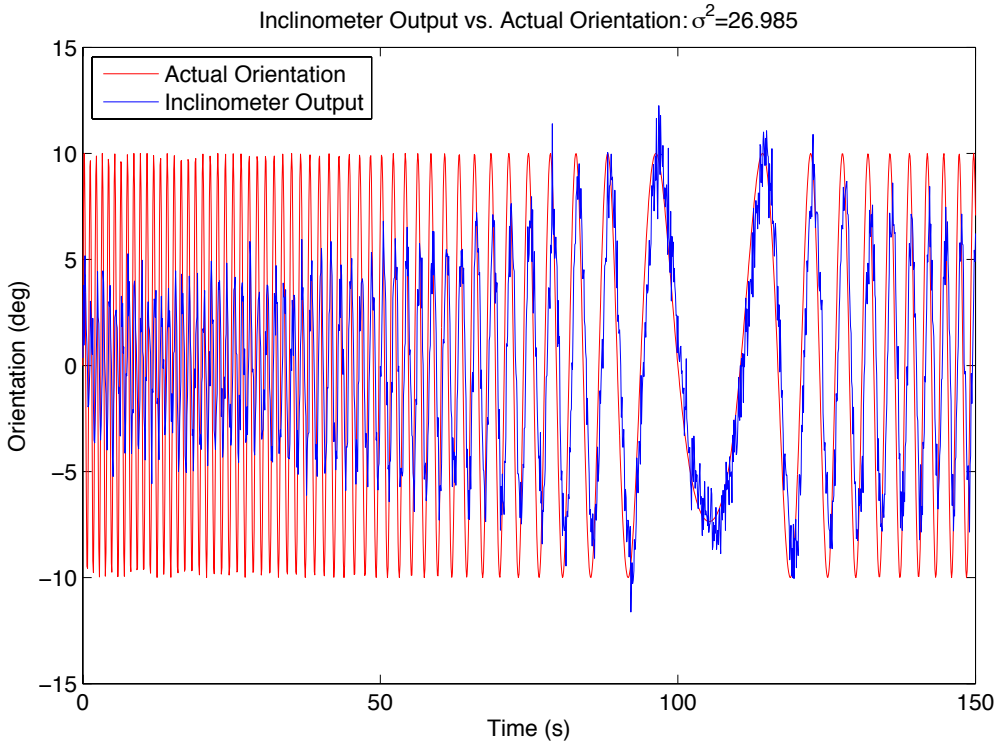


Fig. 6.11: Response of the inclinometer to the input chirp signal.

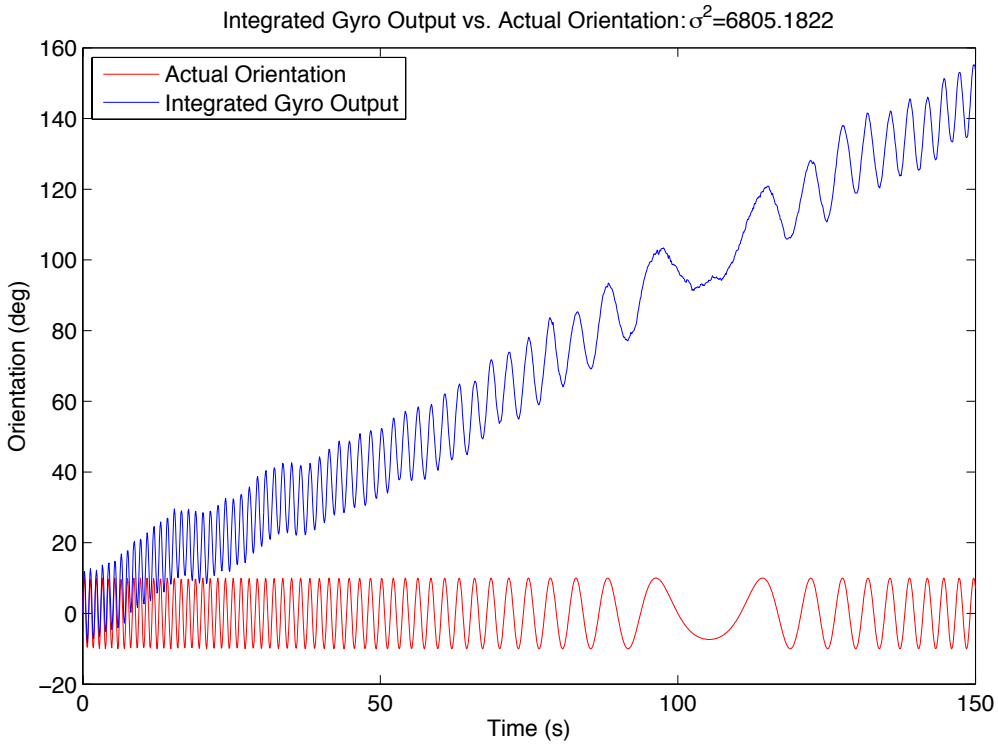


Fig. 6.12: Response of the gyro to the input chirp signal.

show the filter output and error plots when ω_0 is too low and the output is dominated by the gyro. These gyro dominated plots also show similarities to the gyro output in Fig. 6.12.

6.5.2 Other FDCF Design Options

The previous example simulated for this study shows how a modeless FDCF with second order filters can be used to combine the output of two sensors to get a better estimate of orientation. Other design options for FDCFs include more complex, higher order filters and filters based on the sensor models. Some of these options may achieve better performance than was shown in the previous example [177].

6.6 State Space Complementary Filters

The complementary filter can be interpreted in state space form in order to easily realize the complementary filter with digital systems, implement multiple-input multiple-

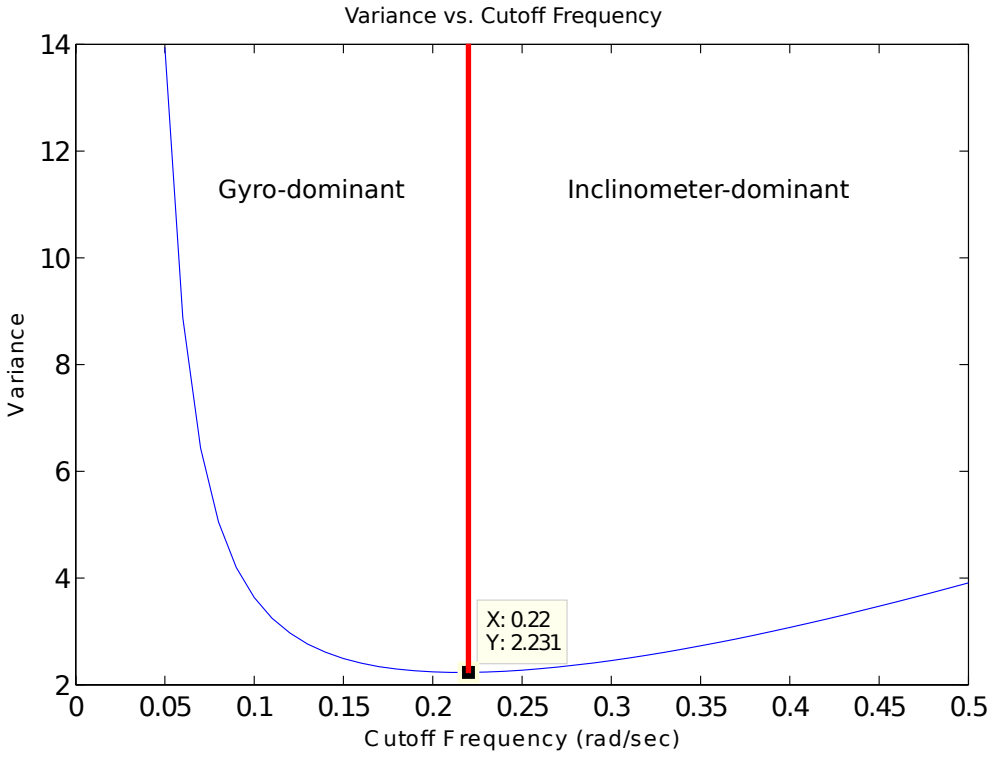


Fig. 6.13: Variance of the error vs. cutoff frequency (ω_0).

output systems, and use classical control theory for filter design [167, 190]. Figure 6.20 illustrates this conversion with an example of the frequency domain complimentary filter where ϕ_i and ϕ_g are the outputs from the inclinometer and the gyro, $G_i(s)$ and $G_g(s)$ are the filters for the inclinometer and gyro, and $\hat{\phi}$ is the estimate of ϕ.

To be a complementary filter, $G_i(s) + G_g(s)$ must equal 1 (as shown in Eq. (6.18)). Therefore $G_i(s)$ and $G_g(s)$ could be set as the following:

$$G_i(s) = \frac{C(s)}{C(s) + s}, \tag{6.22}$$

$$G_g(s) = \frac{s}{C(s) + s}. \tag{6.23}$$

According to the diagram in Fig. 6.20 and the definitions of $G_i(s)$ and $G_g(s)$, $\hat{\phi}$ is defined as

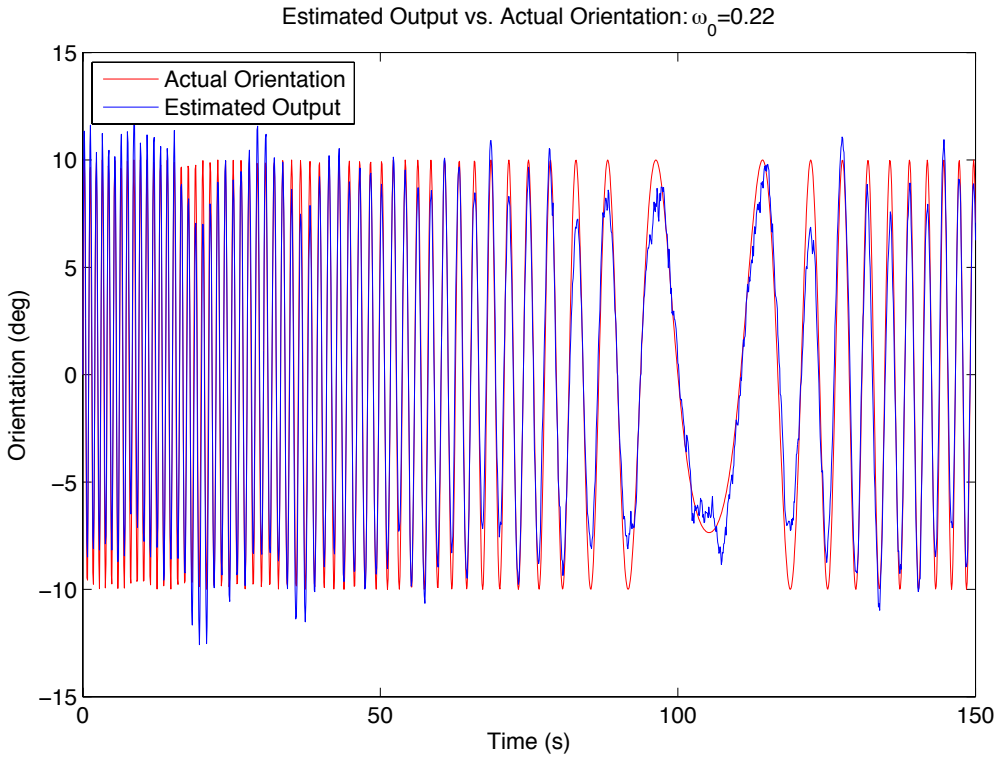


Fig. 6.14: Estimated orientation vs. actual orientation at $\omega_0 = 0.22$ rad/s (optimal frequency).

$$\begin{aligned}
\hat{\phi} &= \phi_i \frac{C(s)}{C(s)+s} + \frac{\phi_g}{s}\frac{s}{C(s)+s}, \\
\hat{\phi} &= \left(\phi_i C(s)s + \phi_g - C(s)\hat{\phi}\right)\frac{1}{s}, \\
s\hat{\phi} &= (\phi_i - \hat{\phi})C(s) + \phi_g. \qquad (6.24)
\end{aligned}$$

Equation (6.24) represents the frequency domain complementary filter in the closed-loop form as displayed in Fig. 6.21. With this design, classical closed-loop control theory can be used to design the complementary filter.

After choosing a controller, Eq. (6.24) can be converted to state space. Using a gain controller, P, Eq. (6.25) shows the state space representation of a complementary filter with $\hat{\phi}$ set as the state space variable x.

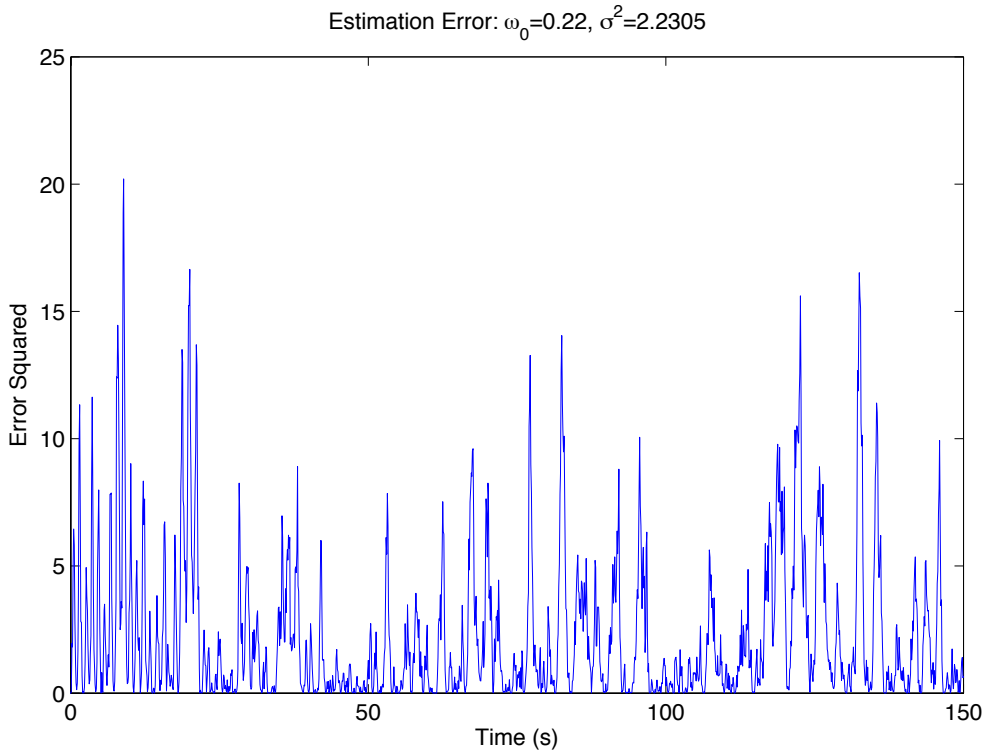


Fig. 6.15: Squared error for $\omega_0 = 0.22$ rad/s (optimal frequency).

$$\dot{x} = (\phi_i - x)P + \phi_g \tag{6.25}$$

Additionally, classical closed-loop control theory can now be used to design the complementary filter. Since there is a good deal of established research on application of fractional calculus in closed-loop control [191], it is a simple matter to change the integer-order controller to a fractional-order one as seen in Sec. 6.7.

6.6.1 SSCF Example–Using a Gain Controller

To test the SSCF from Eq. (6.25), the simulator from Sec. 6.5.1 was used. After running this simulation multiple times for different values of P, they are plotted with their respective error variance to find the gain P with the lowest error. Figure 6.22 shows this plot and a minimum variance of $\sigma^2 = 4.15$ with a gain of 0.84. Figure 6.23 and Fig. 6.24 show the filter output and squared error plots using the optimal gain.

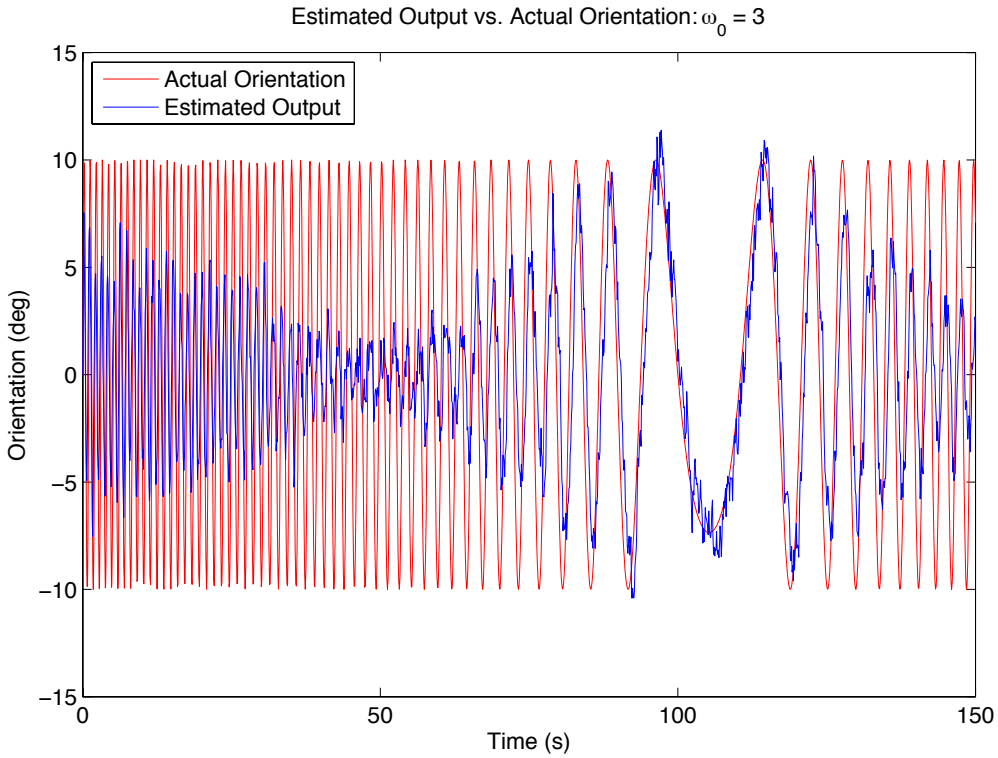


Fig. 6.16: Estimated orientation vs. actual orientation at $\omega_0 = 3$ rad/s (inclinometer dominant).

Even though it was subjected to the same input, the performance of this filter was not as good as the frequency domain filter designed in Sec. 6.5. This could be because of constant bias in the gyro. Introducing an integrator (of requisite order) in the controller could improve the performance by removing this bias.

6.6.2 SSCF Example – Using a PI Controller

After adding an integrator to the controller $C(s)$, Eq. (6.24) can be written as shown below where P is constant gain and I is integrator gain.

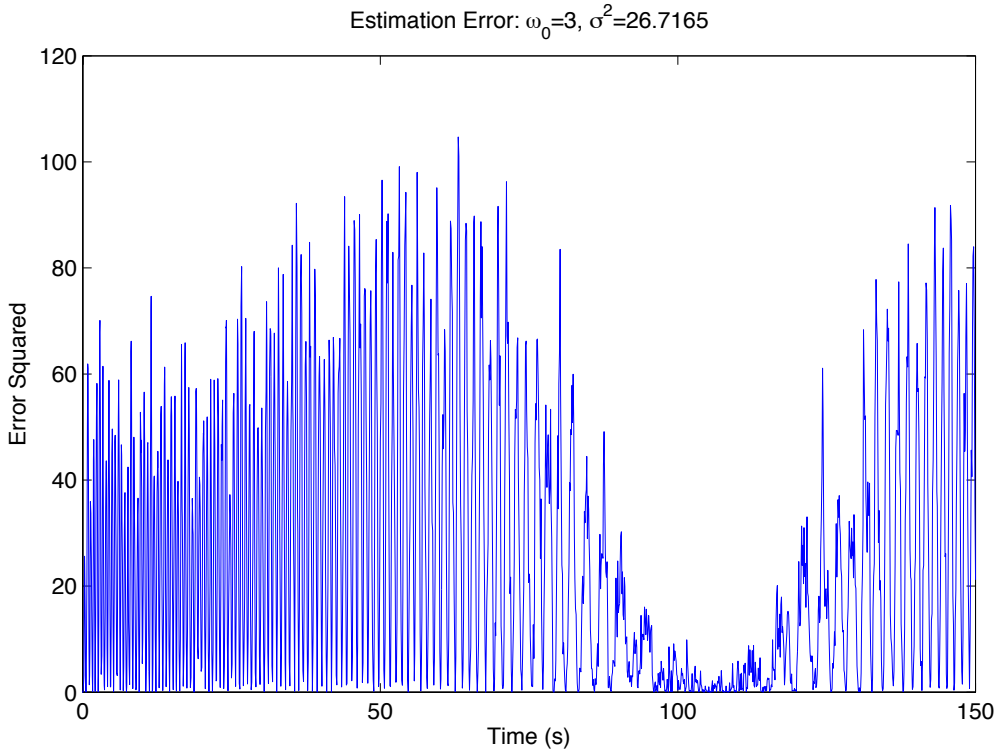


Fig. 6.17: Squared error for $\omega_0 = 3$ rad/s (inclinometer dominant).

$$s\hat{\phi} = (\phi_i - \hat{\phi})(P + I/s) + \phi_g,$$
$$s\hat{\phi} = (\phi_i - \hat{\phi})P + (\phi_i - \hat{\phi})I/s + \phi_g. \tag{6.26}$$

The term $(\phi_i - \hat{\phi})I/s$ is compensation for the bias and can be set as one of the state variables x_2. x_1 will remain equal to the estimated angle $\hat{\phi}$.

$$\dot{x_1} = (\phi_i - x_1)P + x_2 + \phi_g,$$
$$\dot{x_2} = (\phi_i - x_1)I. \tag{6.27}$$

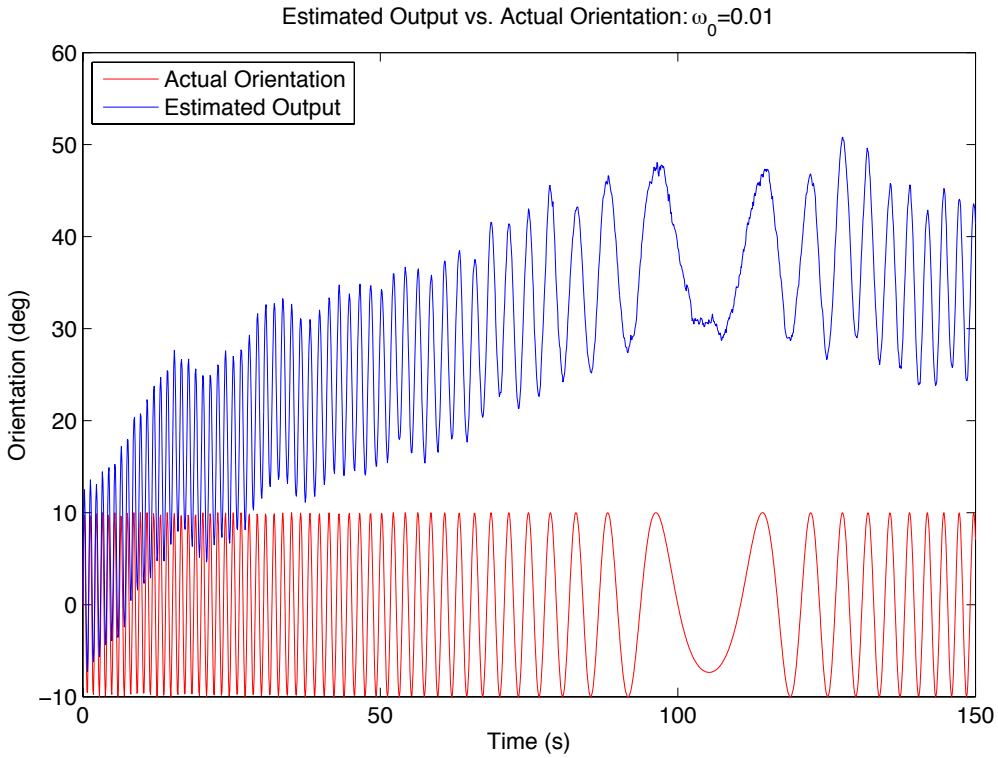


Fig. 6.18: Estimated orientation vs. actual orientation at $\omega_0 = 0.01$ rad/s (gyro dominant).

Notice the similarities between Eq. (6.27) and Eq. (6.25). Equation (6.27) can now be rearranged into its matrix representation (Eq. (6.28)) where b is the gyro bias. The -1 found in the C matrix is negative in order to cancel out the gyro bias.

$$\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \end{bmatrix} = \begin{bmatrix} -P & 1 \\ -I & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} P & 1 \\ I & 0 \end{bmatrix} \begin{bmatrix} \phi_i \\ \phi_g \end{bmatrix},$$
$$\begin{bmatrix} \hat{\phi} \\ \hat{b} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}. \tag{6.28}$$

Figure 6.25 shows the realization of Eq. (6.28)) in Simulink. This model was used with the same input signal and sensor models from the simulation in Sec. 6.5.1. Multiple values of P and I were simulated, the variance of the error was found, and the contour plot in

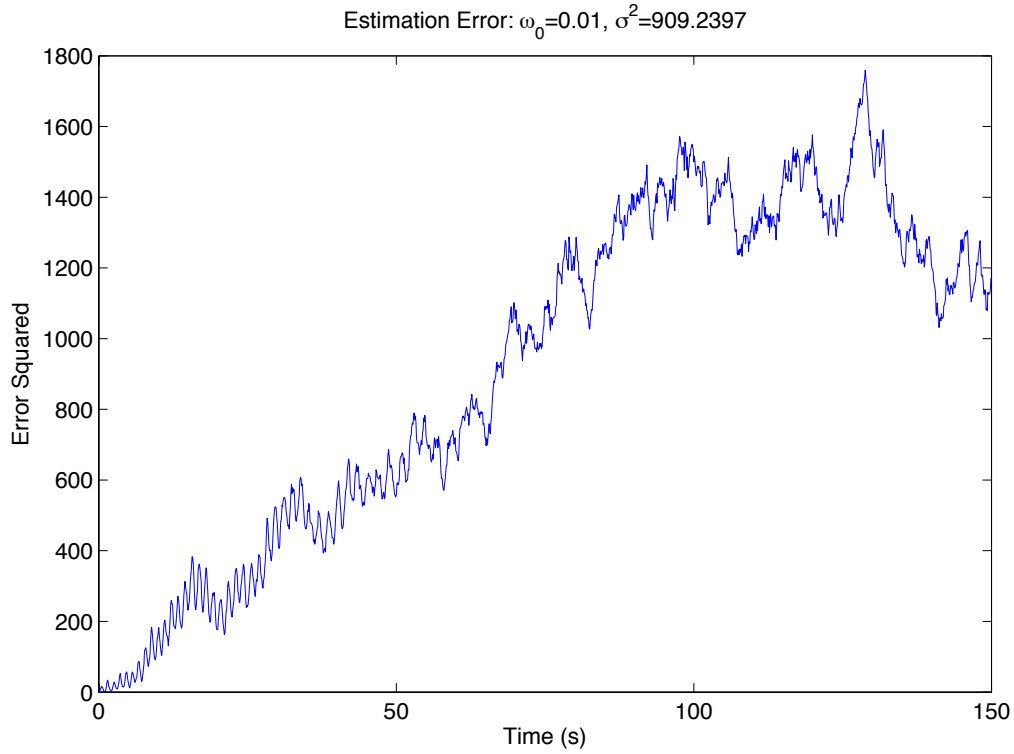


Fig. 6.19: Squared error for $\omega_0 = 0.01$ rad/s (gyro dominant).

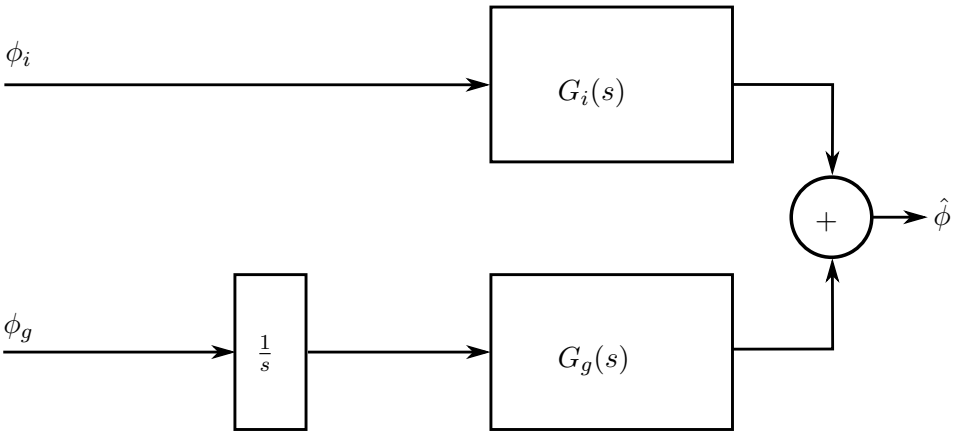


Fig. 6.20: Alternative frequency domain complementary filter.

Fig. 6.26 was generated to find the optimal values for P and I that minimize the variance of the error. As shown in Fig. 6.26, the optimal values for P and I in this example are 0.44 and 0.048, respectively, with a variance of $\sigma^2 = 2.31$.

Figure 6.27 and Fig. 6.28 show the output of the filter with optimal gains. With a

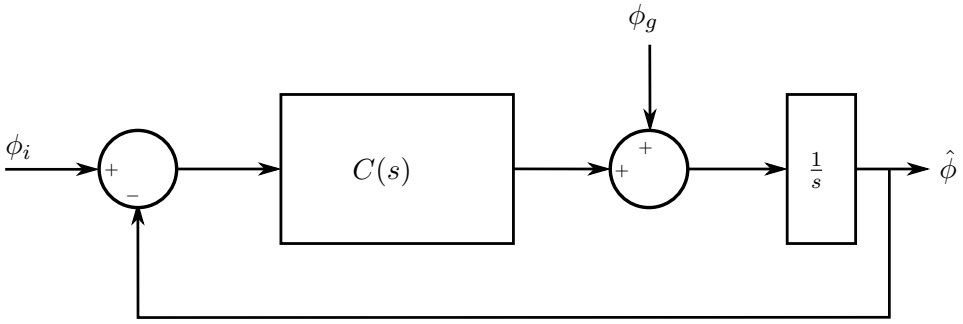


Fig. 6.21: Closed-loop complementary filter system.

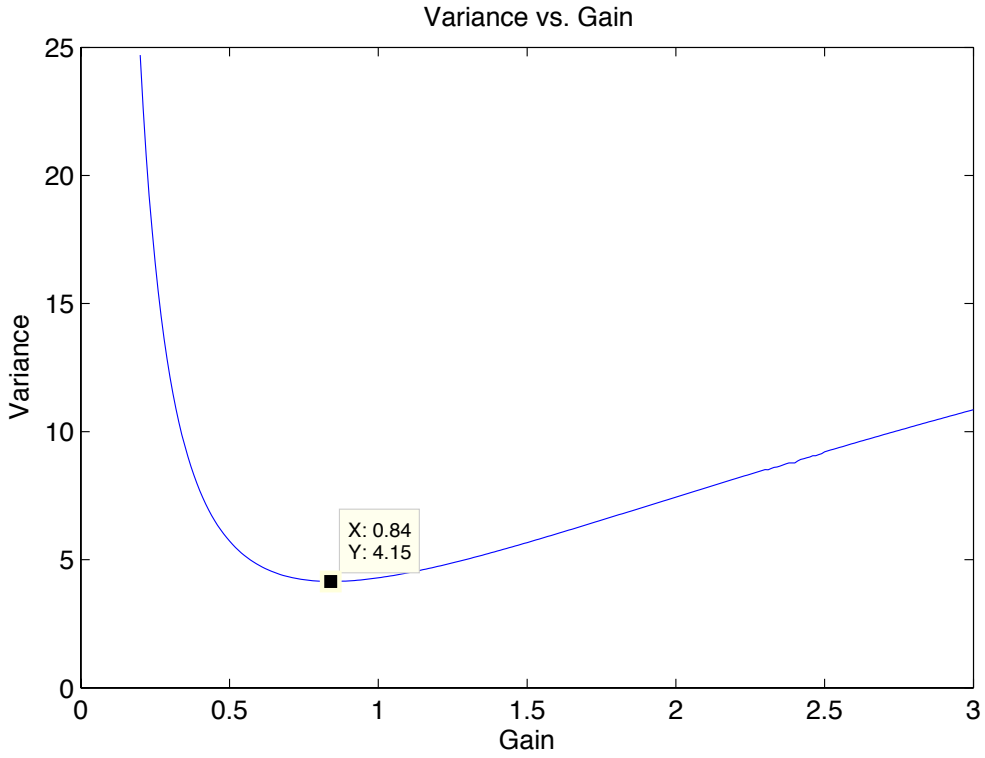


Fig. 6.22: Variance of the error vs. controller gain (P) for SSCF.

variance of $\sigma^2 = 2.31$, the performance of this filter is very similar to the performance of the frequency domain filter designed in Sec. 6.5. This similarity is expected. If the complementary filter from Eq. (6.28) were converted back into its frequency domain form shown in Fig. 6.20, the system would be identical to the second-order filters shown in Eq. (6.19) and Eq. (6.20), where $P = 2\omega_0$ and $I = \omega_0^2$. Therefore, these two systems are equivalent. However, the state space system would be easier to implement using a digital

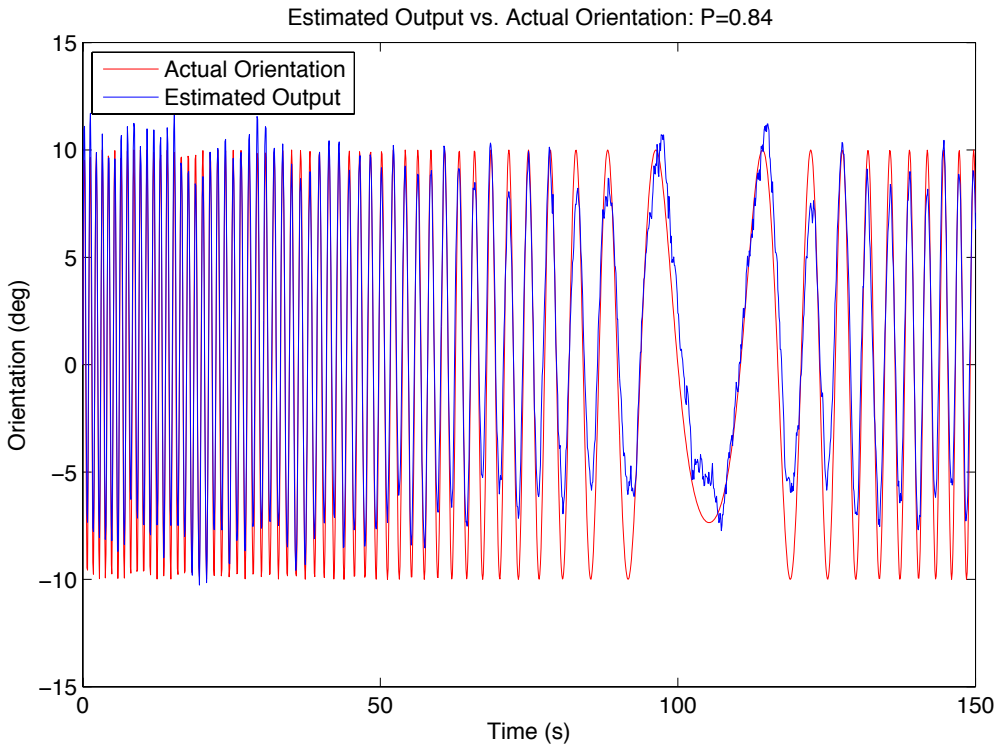


Fig. 6.23: Estimated orientation vs. actual orientation with $P = 0.84$ (optimal gain).

computer. In addition, it is possible to estimate the bias in the state space system since it is added as an additional state. Figure 6.29 shows the estimated bias vs. the actual bias.

6.6.3 Other SSCF Design Options

Since the SSCF is derived using the closed-loop control version of the complementary filter, any other type of controller could be used to help improve the performance of the filter. For example, a full PID controller could possibly perform better in some systems than the PI controller. Other possibilities include model based controllers and fractional order controllers.

SSCFs are also used for multiple-input multiple-output systems. These include SSCFs to estimate 3D orientation using Euler-angles, quaternions [184], and direct cosine matrices [188,192]. Work has also been done on estimating position and velocity using complementary filters [193].

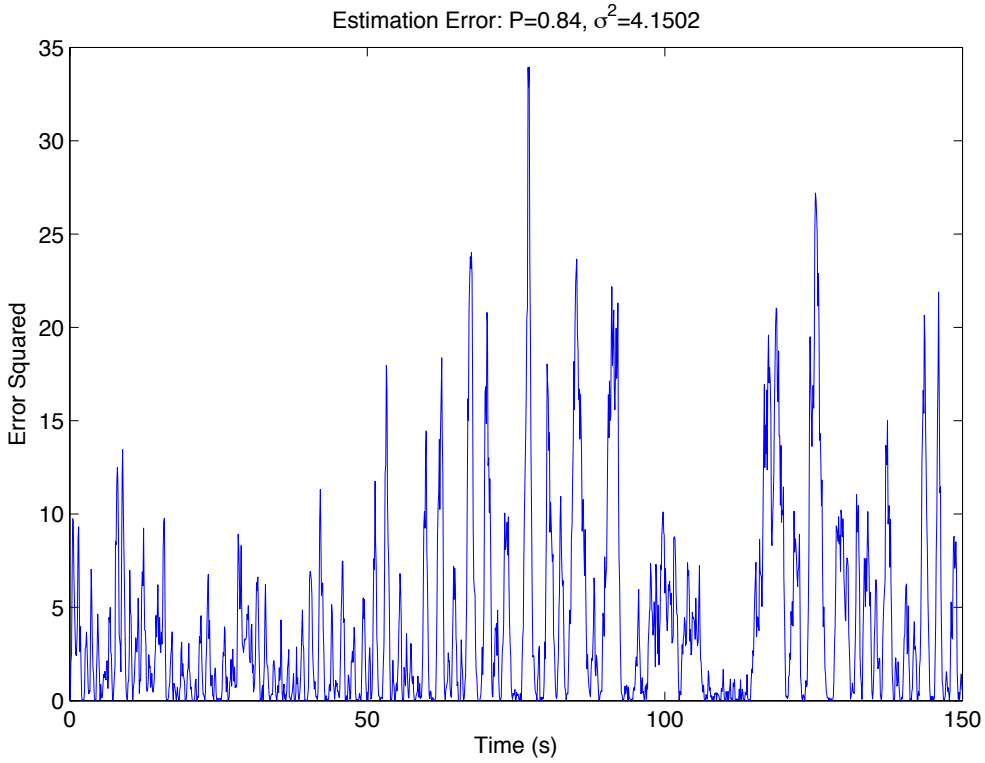


Fig. 6.24: Squared error for $P = 0.84$ (optimal gain).

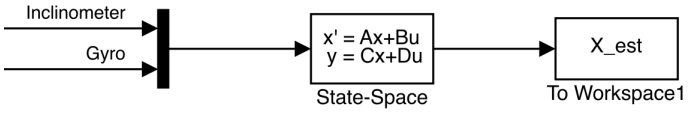


Fig. 6.25: Simulink model of state space complementary filter.

6.7 Fractional-Order Complementary Filters

What follows is a new interpretation of the complementary filter–one with fractional "integro-differential" operators in place of traditional calculus operators.

6.7.1 Fractional-Order Calculus

Fractional calculus is a generalization of integration and differentiation to non-integer order fundamental operator ${}_aD_t^\alpha$, where a and t are the limits of the operation. The

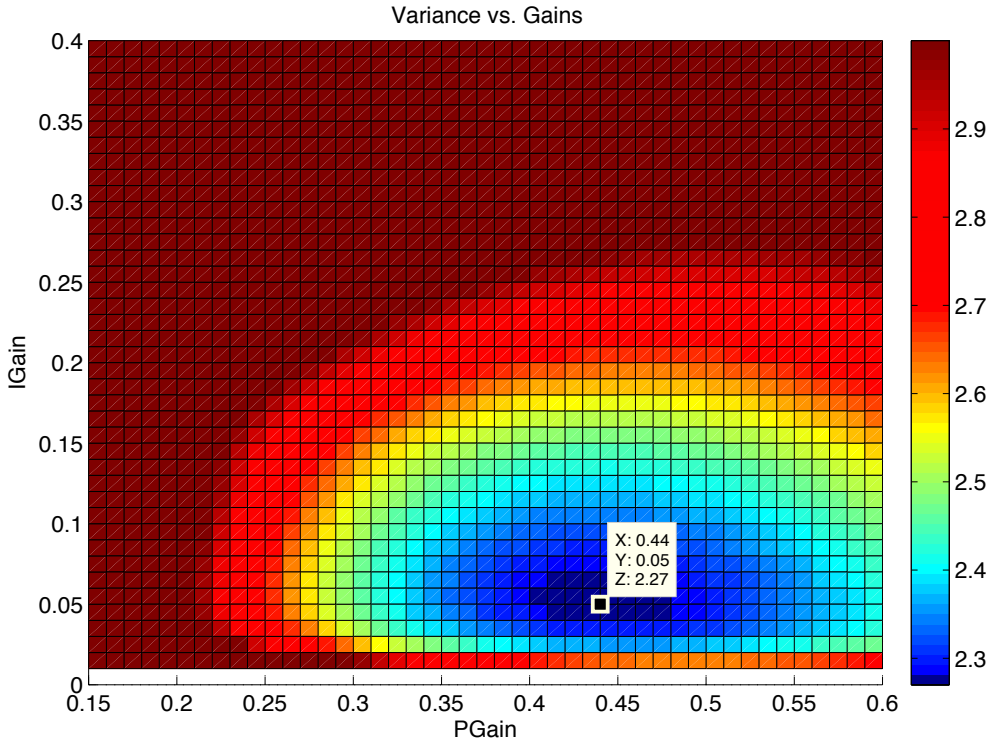


Fig. 6.26: Variance of the error vs. controller gains (P and I) for SSCF.

continuous integro-differential operator is defined as

$$ {}_aD_t^\alpha = \begin{cases} \frac{d^\alpha}{dt^\alpha} & : \ \alpha > 0, \\ 1 & : \ \alpha = 0, \\ \int_a^t (d\tau)^{-\alpha} & : \ \alpha < 0. \end{cases} \tag{6.29} $$

The two definitions used for the general fractional differintegral are the Grunwald-Letnikov (GL) definition and the Riemann-Liouville (RL) definition [194, 195]. The GL is given here

$$ {}_aD_t^\alpha f(t) = \lim_{h \to 0} h^{-\alpha} \sum_{j=0}^{[\frac{t-a}{h}]} (-1)^j \binom{\alpha}{j} f(t - jh), \tag{6.30} $$

where [.] means the integer part. The RL definition is given as

$$ {}_aD_t^\alpha f(t) = \frac{1}{\Gamma(n-\alpha)} \frac{d^n}{dt^n} \int_a^t \frac{f(\tau)}{(t-\tau)^{\alpha-n+1}} d\tau, \tag{6.31} $$

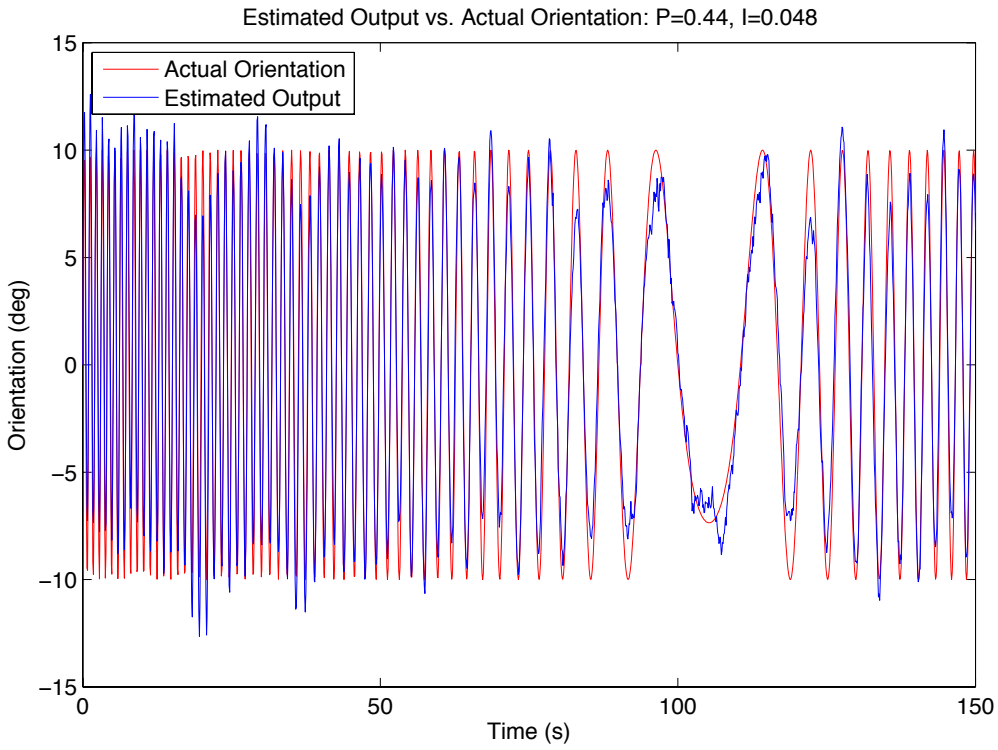


Fig. 6.27: Estimated orientation vs. actual orientation with $P = 0.44$ and $I = 0.048$ (optimal PI gains).

for $(n-1 < \alpha < n)$ and where $\Gamma(.)$ is the *Gamma* function.

The Laplace transform method is used for solving engineering problems. The formula for the Laplace transform of the RL fractional derivative (Eq. (6.31)) has the form [194]:

$$\int_0^\infty e^{-st}\, {}_0D_t^\alpha f(t)\, dt = s^\alpha F(s) - \sum_{k=0}^{n-1} s^k \; {}_0D_t^{\alpha-k-1} f(t)\Big|_{t=0}, \tag{6.32}$$

for $(n-1 < \alpha \leq n)$, where $s \equiv j\omega$ denotes the Laplace operator.

Some other important properties of the fractional derivatives and integrals can be found in many existing works (e.g., Oldham and Spanier [195], Podlubny [194], and others).

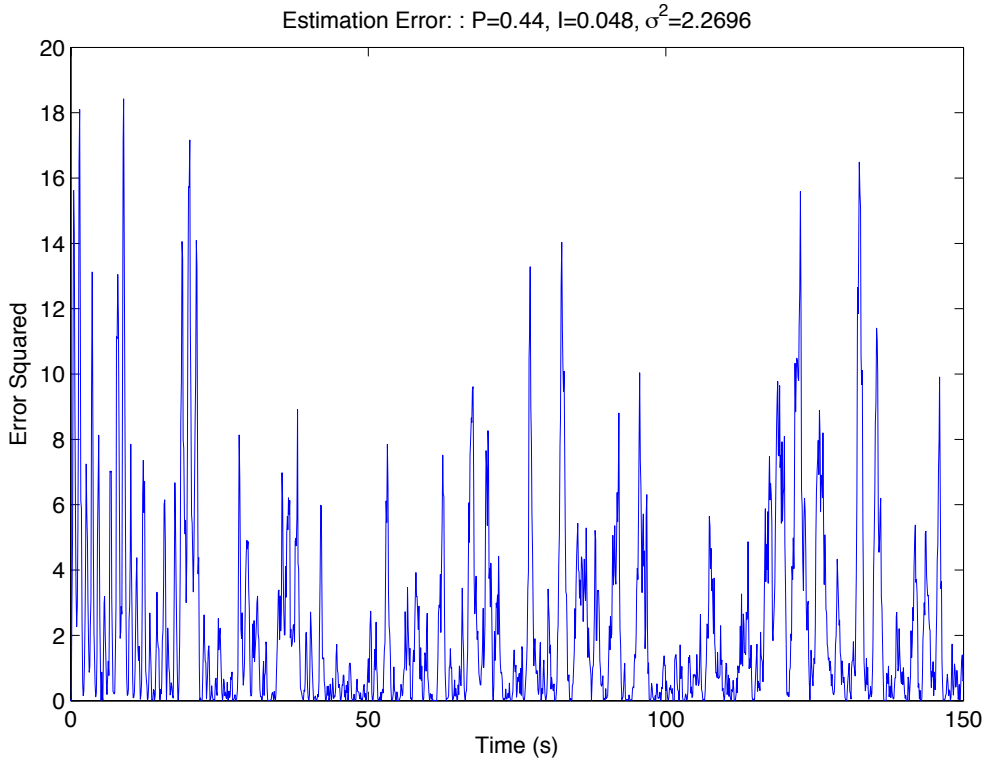


Fig. 6.28: Squared error for $P = 0.44$ and $I = 0.048$ (optimal PI gains).

6.7.2 Fractional-Order Filtering

In order to implement fractional-order calculus in a real system, a practical approximation must be made. While it is possible to realize fractional-order elements in analog hardware–"passive" hardware devices for fractional-order integrator, such as fractances (e.g., RC transmission line circuit and Domino ladder network) [196] and Fractors [197], there exist some restrictions, since these devices are currently difficult to tune. An alternative feasible way to implement fractional-order operators and controllers is to use finite-dimensional integer-order transfer functions, or fractional-order signal processing.

The fractional-order differentiator and fractional-order integrator are fundamental building blocks for fractional-order signal processing. The transfer function of fractional-order integrator (FOI) is simply

$$G_{FOI}(s) = \frac{1}{s^{\alpha}} \tag{6.33}$$

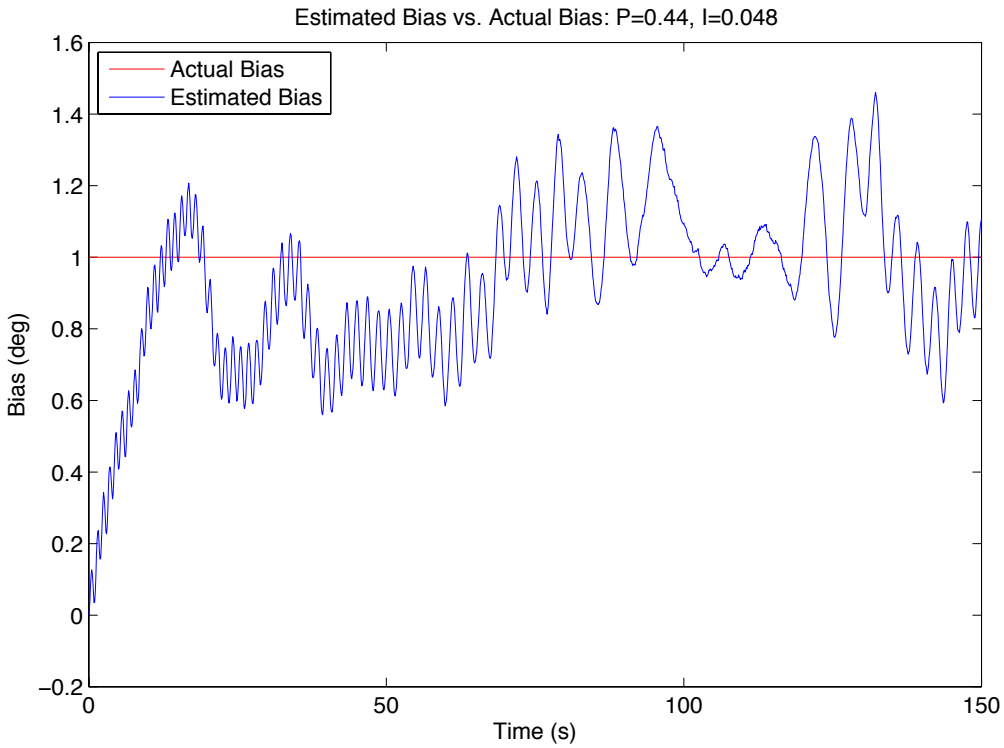


Fig. 6.29: Estimated bias plots for $P = 0.44$ and $I = 0.048$.

where α is a positive real number. Without loss of generality, we only consider $0 < \alpha < 1$. The transfer function of fractional-order differentiator (FOD) is simply

$$G_{FOD}(s) = \frac{1}{G_{FOI}(s)} = s^{\alpha}. \tag{6.34}$$

In time domain, the impulse response of $G_{FOI}(s)$ is

$$h(t) = L^{-1}G_{FOI}(s) = \frac{t^{\alpha-1}}{\Gamma(\alpha)}, \quad t \geq 0. \tag{6.35}$$

Replacing α with $-\alpha$ will give the impulse response of fractional differentiator s^{α}.

A FOI or FOD is an infinite-dimensional system. So, when implemented digitally, it must be approximated with a finite-dimensional discrete transfer function. This is the so-called "discretization" problem of FOI or FOD [198]. The reader is referred to Chen et al. [199], Monje et al. [191], and Krishna [200] for excellent reviews and tutorials on

discretization issues.

As said, an integer-order transfer function representation to a fractional-order operator s^{α} is infinite-dimensional. However, it should be pointed out that a band-limited implementation of a fractional-order controller (FOC) is important in practice, i.e., the finite-dimensional approximation of the FOC should be done in a proper range of frequencies of practical interest [201]. Interestingly, the fractional-order α could even be a complex number as discussed by Oustaloup et al. [201].

In this chapter, the recursive frequency-space discretization techniques of Oustaloup et al. [201] are used. A generalized Oustaloup filter can be designed as

$$G_f(s) = K \prod_{k=1}^{N} \frac{s + \omega'_k}{s + \omega_k}, \tag{6.36}$$

where the poles, zeroes, and gain are evaluated from

$$\omega'_k = \omega_b \omega_{\mathrm{u}}^{(2k-1-\gamma)/N}, \quad \omega_k = \omega_b \omega_{\mathrm{u}}^{(2k-1+\gamma)/N}, \quad K = w_h^{\gamma}, \tag{6.37}$$

where $\omega_{\mathrm{u}} = \sqrt{\omega_{\mathrm{h}}/\omega_{\mathrm{b}}}$. The term "generalized" is used because N can be an odd or even integer.

To illustrate the method, the approximation of the fractional-order integrator of order 0.45 can be obtained. In this particular case, the orders of the approximation are selected as 4 and 5, respectively, with $\omega_{\mathrm{h}} = 1000\,\mathrm{rad/sec}$ and $\omega_{\mathrm{b}} = 0.01\,\mathrm{rad/sec}$. Using MATLAB, from Sheng et al. [202], $H_{\mathrm{Oust}}(s)$ approximations:

$$G_O1(s) = \frac{0.04467s^4 + 21.45s^3 + 548.2s^2 + 783.2s + 59.57}{s^4 + 131.5s^3 + 920.3s^2 + 360.1s + 7.499}, \tag{6.38}$$

and

$$G_O2(s) = \frac{0.04467s^5 + 26.35s^4 + 1413s^3 + 7500s^2 + 3942s + 188.4}{s^5 + 209.3s^4 + 3982s^3 + 7500s^2 + 1399s + 23.71}. \tag{6.39}$$

The Bode plots are shown in Fig. 6.30. It can be seen that the Bode plots of the two filters are relatively close to that of the theoretical one over the frequency range of interest.

It can be seen that the fitting quality is much superior to those obtained with continued fraction-based approaches.

Detailed comparisons of different methods of FOI and FOD discritization can be found in Chapter 5 of the 2012 book by Sheng et al. [202].

6.7.3 Alpha-stable Noise

The Probability Distribution Functions (PDFs) of the sensors used for small UAS navigation are very important to filter choice and tuning. Using the Alpha-Stable Random Distribution (ARD), a standard Gaussian PDF as well as a non-Gaussian PDF was used for simulation.

The ARD is usually denoted $S(\alpha_n, \beta, \gamma, \delta)$. Note that this α_n differs from the fractional-order α. Due to the ARD's nature, only in select cases can the PDF be written analytically, however, from Nolan [203], the characteristic function φ and the PDF can be determined

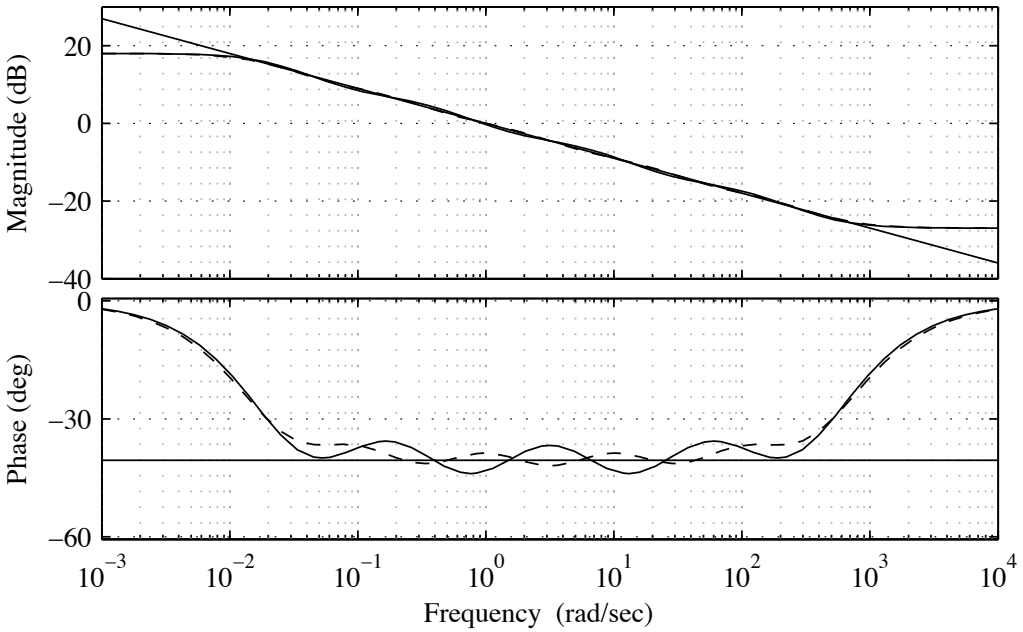


Fig. 6.30: Bode plots of $H_{\mathrm{Oust}}(s)$, corresponding to the approximation of a fractional-order integrator of order 0.45 with the Oustaloup method, with solid lines for $G_O1(s)$, dash lines for $G_O2(s)$ and dotted lines for the theoretical Bode plot.

by:

$$f(x) = \frac{1}{2\pi} \int_{-\infty}^{\infty} \varphi(t) e^{-ixt}\, dt, \tag{6.40}$$

where

$$\varphi(t; \mu, c, \alpha_n, \beta) = \exp\left[\, it\mu - |ct|^{\alpha_n}(1 - i\beta \mathrm{sgn}(t)\Phi)\right], \tag{6.41}$$

and

$$\Phi = \tan(\pi\alpha_n/2), \quad \alpha_n \neq 1. \tag{6.42}$$

The ARDs generalize many PDFs, for example, the well-known Gaussian PDF

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{6.43}$$

is conveniently produced by the ARD with $\alpha_n = 2$, $\gamma = \frac{\sigma}{\sqrt{2}}$. This means that the Gaussian distribution is encapsulated within the alpha-stable distribution, and a single function can be used to simulate both Gaussian and SRD noise cases.

For the purposes of this publication, the ARD was always used with $\alpha_n = 1.8$, and scale-parameter $\gamma = 1.3$. As seen in the plots such as Fig. 6.31, this generates noise sufficiently "spiky" (non-Gaussian) compared to the Gaussian case to be considered low-quality (i.e., low-cost). The effective PDFs of both of the distributions used are found in Fig. 6.32.

The differences in these PDFs can be clearly seen in Fig. 6.11 and Fig. 6.12, by comparison to Fig. 6.31 and Fig. 6.33.

6.7.4 Fractional-Order Complementary Filter Simulations

By their nature, the most simple integer-order and fractional-order controllers have different numbers of parameters and can be difficult to compare fairly. The experiments performed in this chapter were as follows: the integer-order controller was simply a P (or constant-gain) controller. The fractional-order controller was I^{α} (single-term fractional

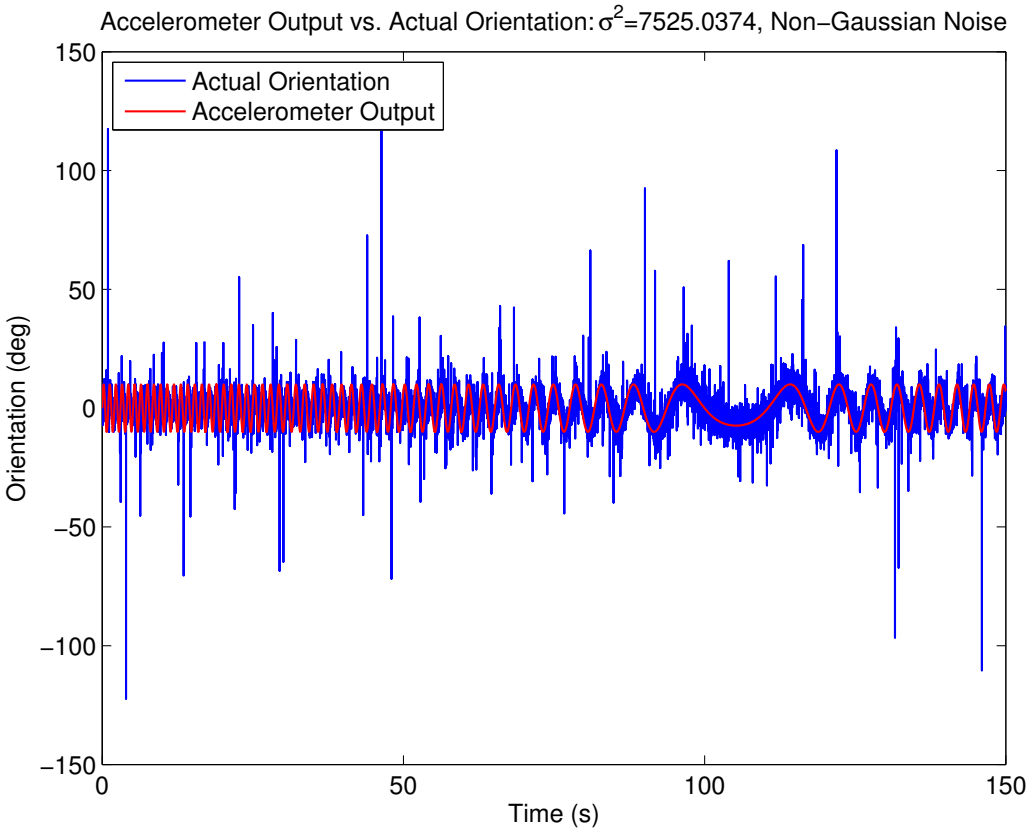


Fig. 6.31: Inclinometer signal with non-Gaussian noise.

with constant gain). This allows for nearly the same degrees of freedom in the simulation process and the most fair comparison. The block diagram for the example system is shown in Fig. 6.34. Since the fractional calculus encapsulates the integer-order calculus with proper choices of α, the same system diagram can be used for all the simulations. The P case is set when $\alpha = 0$.

To accomplish numerical simulations of FOCFs, the following codes from MATLAB-Central were used.

- "STBL: Alpha stable distributions for MATLAB by" Veillette [204] for ARD generation;
- "ninteger" by Valrio [205] for fractional operator simulation.

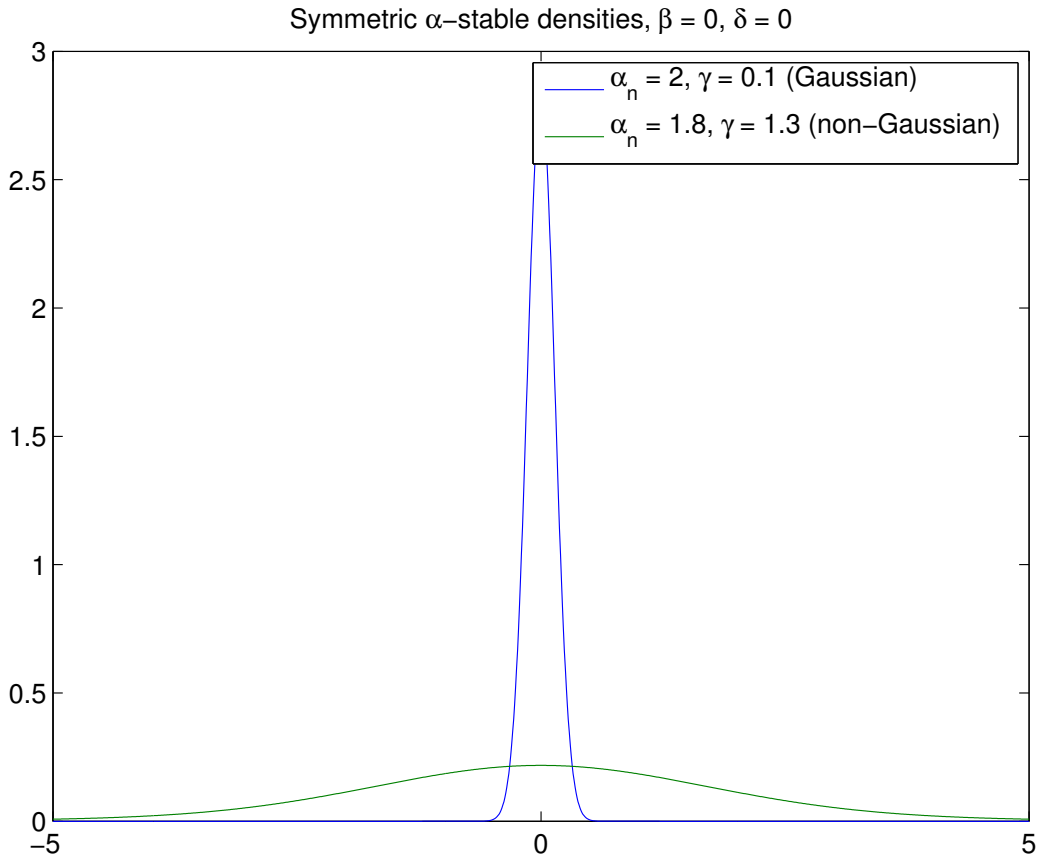


Fig. 6.32: Gaussian and non-Gaussian PDFs used in numerical simulations.

The "ninteger" package includes a full toolbox for simulation of fractional-order implementations in MATLAB Simulink. Due to its numerical reliability, the Crone distretization (as in Sec. 6.7.2) was chosen for the I^α feedback term, with $\omega_\mathrm{b} = 0.01\,\mathrm{rad/sec}$ and $\omega_\mathrm{h} = 1000\,\mathrm{rad/sec}$. Recall that negative values of α create an integrator-like operator.

6.7.5 Fractional-Order Complementary Filter Simulation Results

Five simulations were performed to investigate the performance of the system in question.

1. First, an integer system with a simple P controller was used. A range of gains was used and the variance tracked, to find the minimum. As seen in Fig. 6.35, this was $k = 0.75$, resulting in $\sigma^2 = 3.975$.

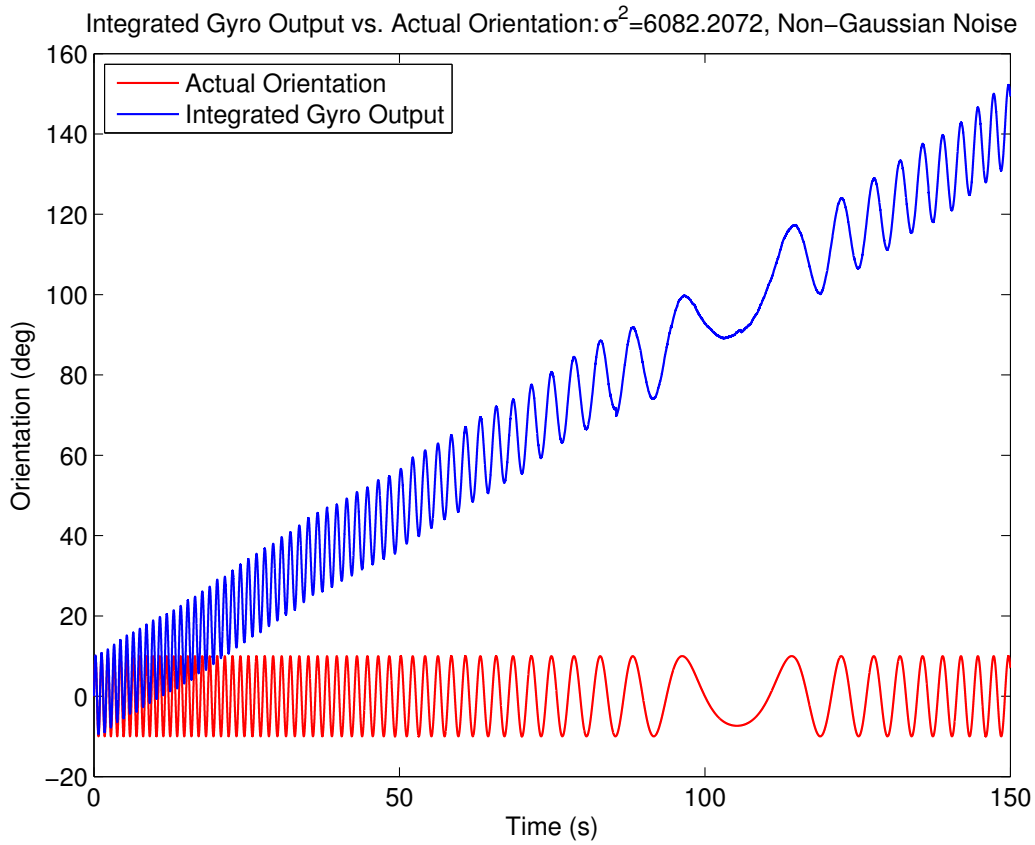


Fig. 6.33: Gyro signal with non-Gaussian noise and offset.

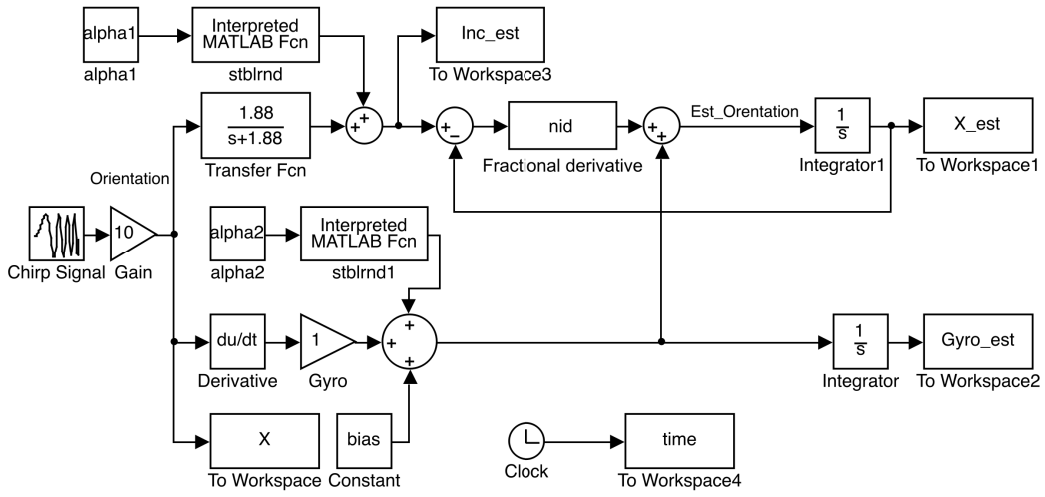


Fig. 6.34: Simulink model used for all simulations.

2. Then, this same simulation was performed with non-Gaussian noise, under which the system produced nearly the same plot (Fig. 6.36), however with the long waiting times involved with the ARD, the variance was not as consistent over the range of gains as the Gaussian case.

3. Then, this same simulation was performed with non-Gaussian noise, under which the system produced nearly the same results, however with the long waiting times involved with the ARD, the variance was not as consistent over the range of gains as the Gaussian case.

4. Next, the fractional-order system was used, with Gaussian noise. The value of the integrator order, α was varied to find a minimum variance. The gain of the I^{α} term was set to the optimal value found in simulation Item 1, $k = 0.75$. This produced an optimal value of $\alpha = -0.18$.

5. The fractional-order system was then tested with non-Gaussian noise, and shown to give the same $\alpha = -0.18$ minimum variance as with the Gaussian case.

6. Finally, using the optimum $\alpha = -0.18$ found above, the controller gain k was optimized for the fractional order controller and non-Gaussian noise case (Fig. 6.37). This gain, $k = 0.35$ is much lower than the integer-order case, while providing less than $\frac{1}{2}$ the variance, at $\sigma^2 = 1.52$.

From these simulations, it is clear that the fractional-order filter outperforms the integer-order filter. With a low-order integrator (less than 0.2), and comparatively low gain, the fractional-order controller nearly halves the variance of the integer-order controller.

6.8 Guidelines for Complementary Filter Use in sUAS

Kalman or Kalman-style combining filters are the established solution for combining sensor data into navigation-ready data. Extended Kalman filter approaches allow nonlinear models to be used and estimated with the Kalman techniques; however, these models are linearized and assumptions are made that the estimates do not deviate too far from the

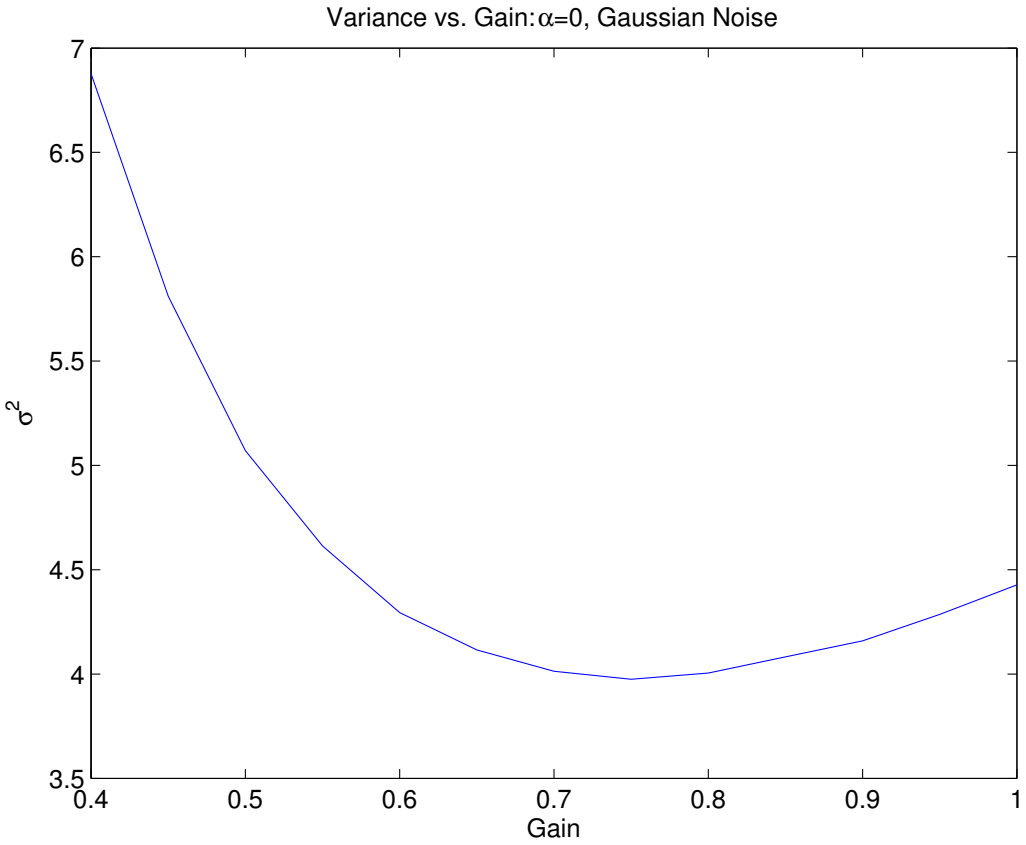


Fig. 6.35: Variance vs. gain: integer controller with Gaussian noise.

nominal solution. They have also proved difficult to apply to small UAS with low-cost, high-noise sensors [157,158]. Complementary filters are not mathematically rigorous like the Kalman filter, and are therefore not well-suited for high-risk applications like space borne missions. However, compared to Kalman filtering, complementary filter techniques are less computationally intensive and more readily performed on small, low-power computing hardware, which is ideal for small, low-cost UASs.

Although Gaussian statistical models are traditional and have a wide range of applications, non-Gaussian, power-law statics are ubiquitous in practice [159]. Although augmented Kalman-style filters (such as the Kalman-Levy filter) that allow non-Gaussian PDFs have been considered in research, they are not well studied and have even higher mathematical complexity than standard Kalman techniques [160].

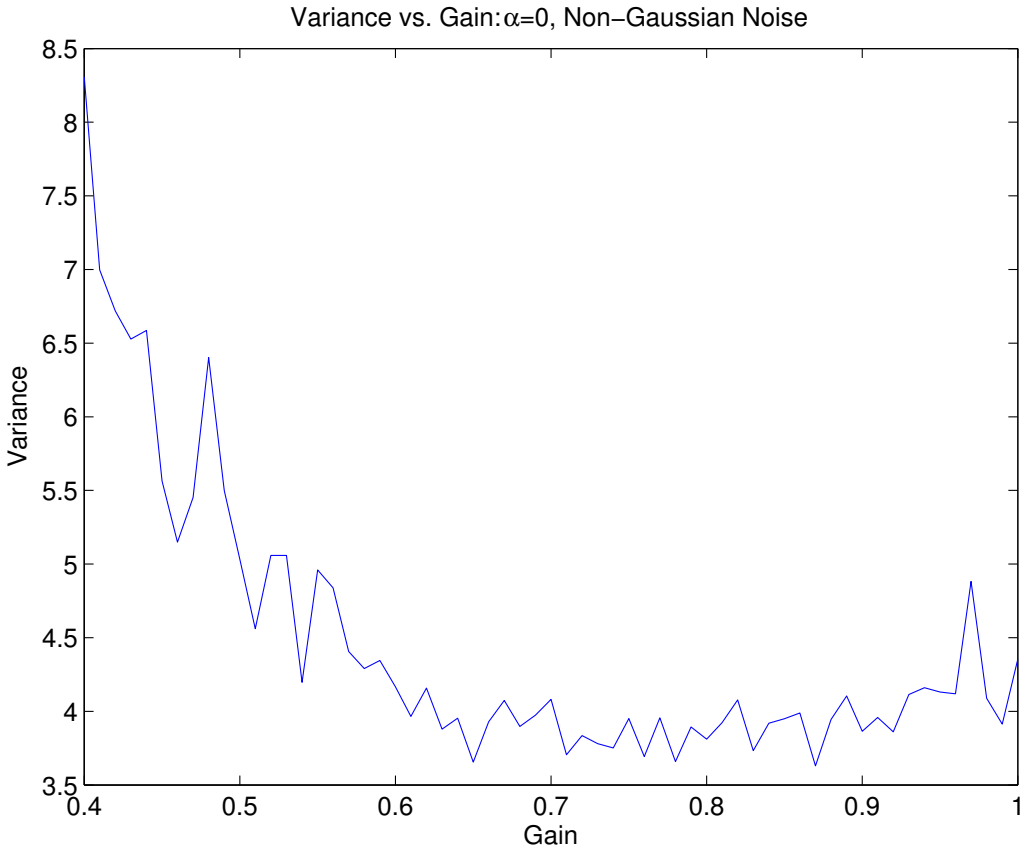


Fig. 6.36: Variance vs. gain: integer controller with non-Gaussian noise.

Complimentary filters work without assuming Gaussian noise statistics and are applicable to a wider variety of sensor hardware such as commonly-available MEMS devices with longer-tail noise tendencies [161]. Since complementary filters require less computational power, they couple well with low-cost MEMs sensors and microcontrollers and can be better suited to the navigation needs of small, low-cost UASs.

The suggested guidelines include

1. Understand each sensor's pros and cons;
2. Understand the tradeoff objectives;
3. Whenever possible, as demonstrated in this chapter, try to iteratively decide the best cut-off frequency;

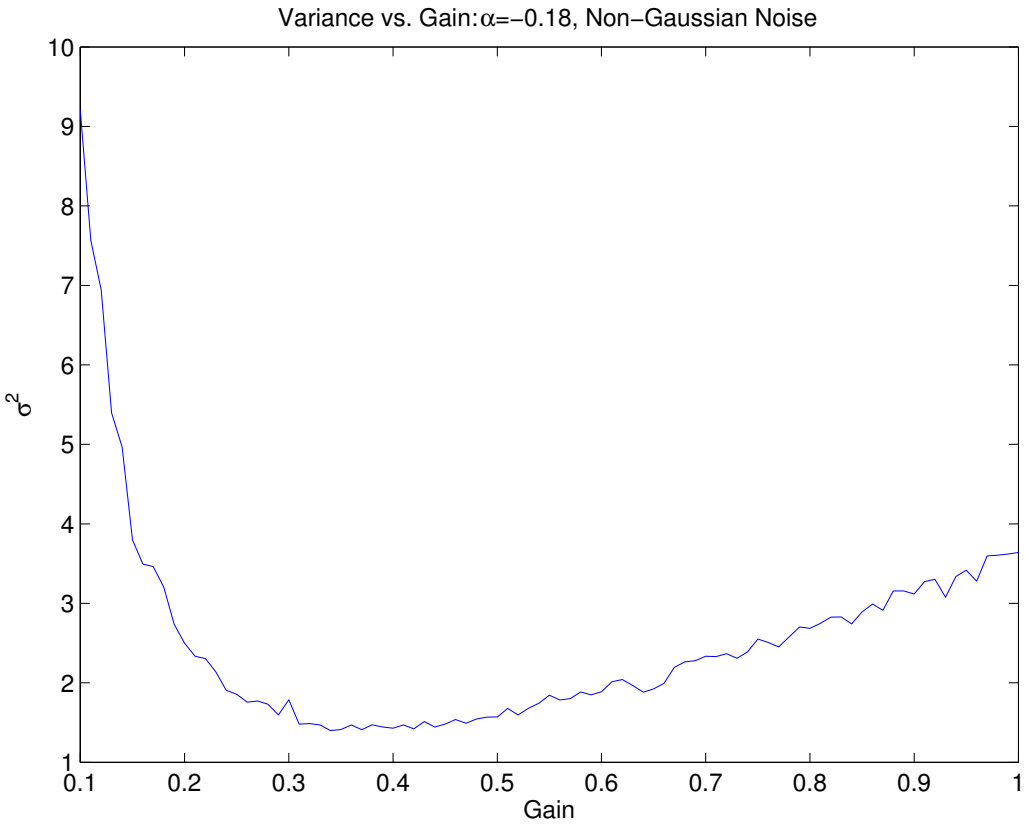


Fig. 6.37: Variance vs. gain: fractional-order controller with non-Gaussian noise and $\alpha = -0.18$ found numerically.

4. Use as much domain knowledge as possible when deciding the design parameters linked to the complementary filter design.

6.9 AERIS Significance of Navigation and Chapter Summary

Navigation is critical for UASs to fly safely and for collection of scientific mission data. In order to better comply with AERIS and perform this task with lower cost and higher accuracy and reliability better navigation data is critical, but must be weighed against the cost of the navigation systems if true low-cost personal remote sensing-level UASs are to be a reality.

Other than the now standard Kalman filter, one way to fuse data from different sensors is to use a complementary filter. These complementary filters can be formulated in a tra-

ditional control system format, and can then be tuned like any control system. Fractional calculus is as old as the traditionally used integer-order calculus, and its use is still being investigated. When the concepts of fractional calculus are introduced into the complementary filter, it is shown that it is possible to derive much better performance from noisy sensors, in this case, more than double the performance can be achieved as evaluated by variance.

Future work includes noise modeling of physical navigation sensors, testing of fractional-order complementary filters on real flight data, and eventually implementation and flight with a real UAS.

Chapter 7

Battery State-of-Charge Aware Altitude Controller for Small, Low-Cost Multirotor Unmanned Aerial Vehicles

Robust altitude control of a multirotor small Unmanned Aerial Vehicle (sUAV) is one of the most difficult control problems of Vertical Take-Off around Landing (VTOL) UAVs [206, 207], of which quadrotors and hexarotors are common.

Small, low-cost UAVs are typically driven by Lithium-ion Polymer (LiPo) batteries, because of their high energy density, high charge and discharge rates, long lifetime, relatively low memory effects compared to other battery technologies [208], and affordable cost. The internal dynamics of LiPo batteries changes during power discharge and degrade the flight and control performance due to the change in available power.

This change in dynamics affects the UAV's ability to maintain desired altitude. The goal of this chapter is to:

1. Analyze LiPo battery and actuator dynamics;
2. Describe their effects on the flight performance;
3. Briefly summarize existing control strategies for small, rotary-wing sUAS altitude control;
4. Propose a novel altitude controller and a battery monitoring system for small rotary-wing sUASs;
5. Experimentally verify the controller performance with real flight data.

AERIS-compliance requires detailed knowledge about mission performance, and because the AggieAir sUASs mainly are based on LiPo for their energy storage, it is critical for AERIS-compliant sUASs to have knowledge about the health of the battery system,

as well as the ability for the power system to transfer energy during any point in a given mission. This means that state-of-charge estimation is critical to enable mission risk estimation systems like ISaAC or the GCS operators to make command decisions such as safely landing or other early mission termination options.

This chapter, drawn from Podhradsky et al. [209], first gives a necessary background about UAV altitude control in Sec. 7.1. Then, battery and actuator models are described in Sec. 7.2. A summary of existing solutions to this problem is presented in Sec. 7.3. The proposed controller, then, is described in Sec. 7.4. The experimental test setup is described in Sec. 7.5.1, and the laboratory experimental results which prove the efficiency of the approach are shown in Sec. 7.5.3. Implementation of the controller on an existing AggieAir sUAV and its flight verification is presented in Sec. 7.6.3.

7.1 sUAS Altitude Control Obstacles

Precision agricultural applications, 3D mapping and other civilian applications of UAVs require relatively precise altitude control (i.e., within 1m [18]).

Relative altitude estimation in multirotor UAVs is typically based on measurements from an accelerometer and a pressure sensor. GPS provides absolute position, but its vertical accuracy is rarely better than few meters in perfect conditions so its not commonly used. Pressure sensors have resolution of $\pm$10cm, but they drift in time, thus introducing error into the relative estimate. Acceleration data are burdened with noise from motor induced vibrations, thus they have to be corrected with pressure readings. Additional sensors (such as ultrasonic or laser altimeters) can be used, however their use is limited to close proximity of the ground. Note that in altitudes above 20 meters, the UAV is usually subject to strong wind gusts ($\geq$ 10m/s) and pressure changes [210], which further affect the controller performance.

A traditional approach combining accelerometer and pressure sensor data and using a classical PID control system can provide satisfactory altitude hold despite the changing battery dynamics. However, its performance strongly depends on properly tuned altitude estimator, size of the platform (heavier UAVs with higher inertia are more difficult to

control [211]), and the operational altitude.

The proposed controller offers an alternative approach, which can be used when altitude estimation cannot be tuned, Electronic Speed Controllers (ESC) do not provide a feedback loop around Revolutions Per Minute (RPM) of the motors, or when classical PID control fails.

The significance of the battery dynamics for larger ($>$ 1kg) platforms can be seen in Fig. 7.1. It shows a radio controlled indoor flight of a hexarotor at constant altitude. A 4-cell LiPo battery is fully charged at the beginning, and discharges during this 19 minute flight. The pilot increases throttle command to keep constant power output from the battery and thus constant altitude. During the flight, the pilot had to increase the throttle by around 10% (comparing the beginning and end of flight). After the 18th minute of flight, the battery voltage suddenly drops as the battery is almost completely depleted and the voltage begins to collapse. Before the hexarotor lands the voltage dropped even below the minimal recommended limit for LiPo batteries, 12V (3V for each cell) which could damage the battery.

Clearly it is important to know the battery state of charge for safety reasons, as well as to adjust the control according to the battery dynamics.

7.2 Battery and Actuator Models

An actuator of a multirotor UAV typically consists of a LiPo battery, a Brushless Direct Current (BLDC) motor, and an Electronic Speed Controller (ESC) which controls the rotation of the motor (measured in Revolutions Per Minute–RPM).

7.2.1 Battery Model

An equivalent circuit representation of the Chen and Rincon-Mora's battery model [212] is shown in Fig. 7.2. Although the model was originally validated on NiMH and Li-ion batteries, it can still be successfully applied to LiPo batteries, since LiPo and Li-ion batteries have very similar characteristics [213].

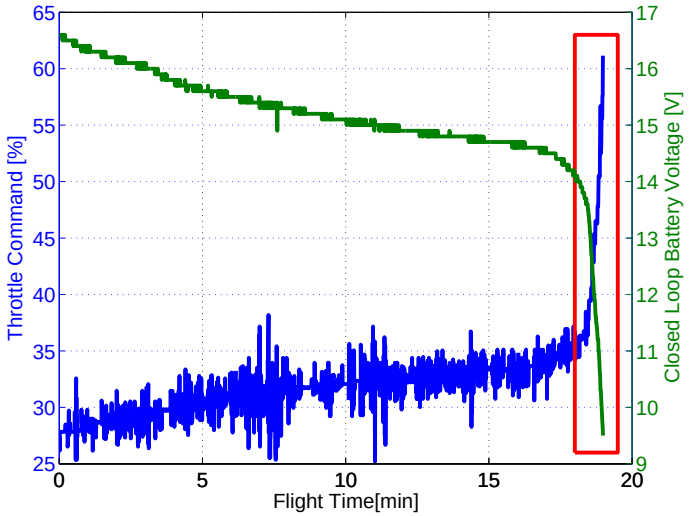


Fig. 7.1: Change in throttle command during indoor hexarotor flight–the red box indicates the beginning of collapse.

The left half of the model describes the variation of the State Of Charge (SOC) of the battery ($x_1 = \text{SOC}, x_1 \in [0, 1]$), the right half models the variation of battery output voltage y as a function of the charge/discharge current [213]. Note that SOC is the percentage of the maximum possible charge that is present inside a rechargeable battery [214]. All the

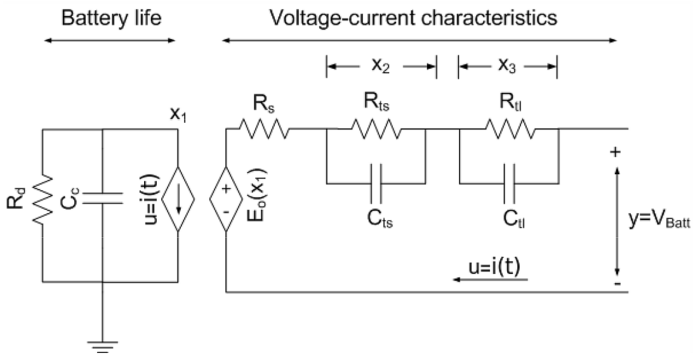


Fig. 7.2: Chen and Rincon-Mora's battery model.

circuit components $C_{ts}, C_{tl}, R_s, R_{ts}, R_{tl}$, and E_0 are functions of x_1:

$$\begin{aligned}
C_{ts}[\mathrm{F}] &= -k_4 e^{-k_1 x_1} + k_3, && (7.1)\\
C_{tl}[\mathrm{F}] &= -k_6 e^{-k_2 x_1} + k_5, && (7.2)\\
R_s[\Omega] &= k_7 e^{-k_8 x_1} + k_9, && (7.3)\\
R_{ts}[\Omega] &= k_{10} e^{-k_{11} x_1} + k_{12}, && (7.4)\\
R_{tl}[\Omega] &= k_{13} e^{-k_{14} x_1} + k_{15}, && (7.5)\\
E_0[\mathrm{V}] &= -k_{16} e^{-k_{17} x_1} + k_{18} + k_{19} x_1 && (7.6)\\
&- k_{20} x_1^2 + k_{21} x_1^3,\\
C_c[\mathrm{C}] &= 3600 C f_1 f_2, && (7.7)
\end{aligned}$$

where $k_i > 0$ for $i = 1 \ldots 21$ and $k_1 < k_2 < k_3 < k_4 < k_5 < k_6$. E_0 is the Open-Circuit Voltage (OCV) of the battery. In Eq. (7.7) $f_1, f_2 \in [0, 1]$ are factors taking into accounts the effects of temperature and charge-discharge cycles [213]. C[A] stands for maximal battery capacity. For a new battery at a room temperature $f_1 = f_2 = 1$, but f_2 will decrease after each charge-discharge cycle (in fact representing State Of Health - SOH of the battery). This allows the model to account for ageing of batteries, which is characterized by loss of maximal charge the battery can store [215]. The various resistances, capacitances, and constants ($k_1 \ldots k_2 1$) are independent of $i(t)$. Note that with decreasing SOC resistance of the battery (R_s, R_{ts}, R_{tl}) increases and capacitance (C_{tl}, C_{ts}) decreases [212].

State space realization of the battery model is provided by Knauff et al. [216] as:

$$\begin{aligned}
\dot{x_1} &= -\frac{1}{C_c} i, && (7.8)\\
\dot{x_2} &= -\frac{x_2}{R_{ts} C_{ts}} + \frac{i}{C_{ts}}, && (7.9)\\
\dot{x_3} &= -\frac{x_3}{R_{tl} C_{tl}} + \frac{i}{C_u}, && (7.10)\\
y &= E_0 - x_2 - x_3 - i R_s, && (7.11)
\end{aligned}$$

where y represents the voltage output from the battery (if $i = 0, y =$OCV), x_2 represents

the voltage drop across $R_{ts}\|C_{ts}$ and x_3 stands for the voltage drop across $R_{tl}\|C_{tl}; x_2, x_3 \in \mathbb{R}$ and $\mathbf{x}_0 = [1, 0, 0]^\top$.

The OCV(E_0) is a function of SOC(x_1), which itself depends on temperature and SOH. Assuming a constant ambient temperature, the dependence of OCV on SOC has the following properties:

1. The OCV increases monotonically with SOC,
2. The OCV drops in an exponential fashion as the SOC approaches 0%,
3. The OCV rises in an exponential fashion as the SOC approaches 100%,
4. In the large region between 20% and 85% of SOC the relation is nearly linear,

which are captured in the battery model. Figure 7.3 shows an experimental verification of the OCV vs. SOC dependency, which has the predicted properties. The SOC was measured using Coulomb counting technique (i.e., current integration), more details can be found in Podhradsky et al. [215].

The properties of LiPo battery dynamics can be summarized as:

1. Open Circuit Voltage decreases with SOC (discharge),
2. Closed Loop Voltage decreases with increased load and with SOC,
3. Internal battery resistance increases with lower SOC,
4. Battery capacity decreases as the battery ages (SOH is degrading).

For the purpose of this work, only new batteries are assumed, and no SOH is estimated.

7.2.2 Actuator Model

For actuation, Brushless Direct Current (BLDC) motors are typically used, since they provide higher efficiency and more torque per Watt than brushed DC motors [217]. BLDC motors for UAVs are almost always sensor less trapezoidal motors with stator coils in the core of the motor (outrunners). It is because of a combination of lower cost (no position

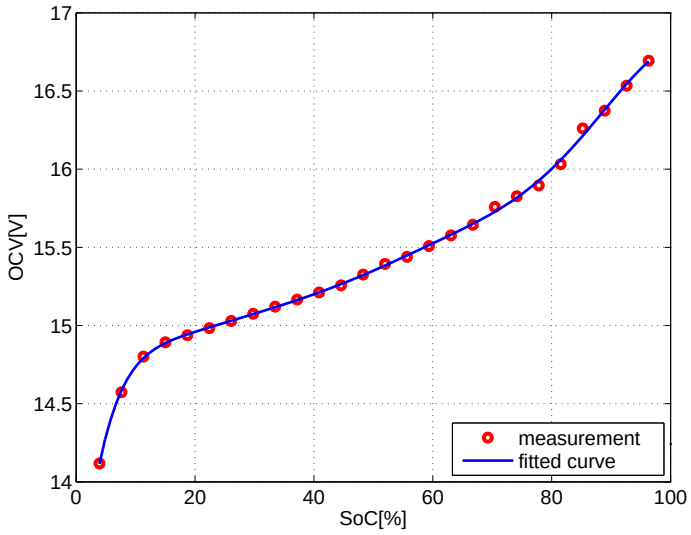


Fig. 7.3: Dependency of OCV on SOC.

sensor needed), and higher efficiency over sinusoidal/inrunner motors [217]. Motors are characterized by their number of magnetic poles and a motor constant [218]. An example of the actuator is shown in Fig. 7.4.

To drive a BLDC motor, an Electronic Speed Controller (ESC) is used. An ESC consists of a controller, MOSFET driver, and at least six MOSFET transistors. To sense the position of the motor, back ElectroMotive Force (back-EMF) is fed to the micro controller. An illustrative diagram of a typical ESC is shown in Fig. 7.5 from Allegro documentation [219]. The driver uses Pulse Width Modulation (PWM), typically at 16kHz, to open/close individual MOSFETs, according to commands from the microcontroller. MOSFETs then activate the phases with current.

Such actuator connected to a power supply, can be described as a first order system with delay [218]:

$$G(s) = \frac{K_p}{1 + T_p} e^{-T_d s}, \tag{7.12}$$

where T_d is a communication delay between the autopilot and the ECS. If the control rate is 512Hz, $T_d = 1.95$ ms. T_p is typically between 16 and 100ms. K_p is strongly dependent on the combination of a motor and a propeller [218]. The produced thrust directly depends

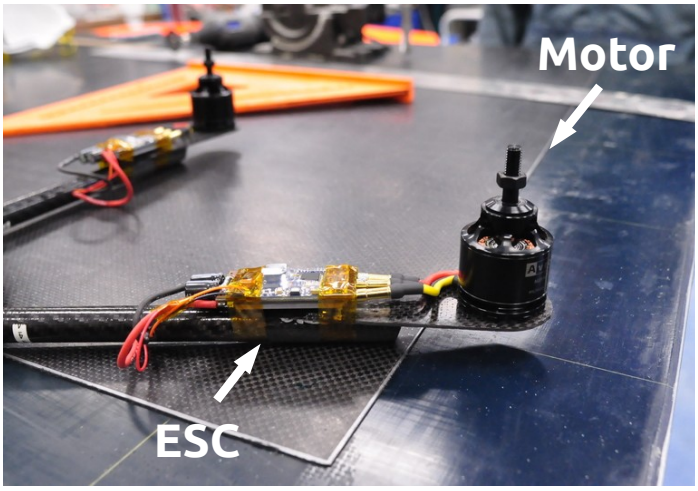


Fig. 7.4: An example of a multirotor actuator–Av-Roto BLDC motor (without a propeller) and Mystery 40A ESC.

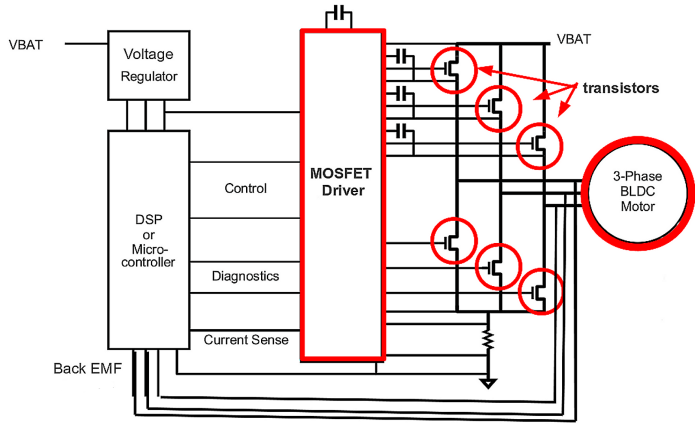


Fig. 7.5: A typical ESC and motor connection scheme. Microcontroller (left), MOSFET driver (middle), MOSFETs and BLDC motor (right), from Allegro.

on RPM.

However, battery dynamics affects the actuator and the real system is no longer linear time invariant (LTI). To understand this relationship, the ESC functionality has to be described in more detail. BLDC motor is rotated by induced voltage on its armature. For trapezoidal motors, a square current wave is used and the relation between current and induced voltage is as follows [217]:

$$V_k(t) = R_k i_k(t) + L\frac{di_k(t)}{dt}, \tag{7.13}$$

where $k = 1 \dots 3$ is a number of phase, R is resistance of the wiring and L is inductance of the coil. The motor speed depends on the duty cycle of the MOSFET driver, and on energy delivered to it during one cycle. Energy accumulated in the inductor during one cycle:

$$W[\mathrm{J}] = L\int_0^T idt. \tag{7.14}$$

T is constant as the duty cycle is the same. From Eq. (7.3), Eq. (7.4), and Eq. (7.5) it can be seen that the resistance of the battery increases during discharge. Since the battery is not an ideal current source, the rate of change of the supplied current is limited. With increased resistance, the maximal rate of change of the current become lower. Energy delivered to the motor during one cycle will decrease, so does RPM of the motor.

7.3 Comparison of Existing Solutions

In this Section the most common multirotor altitude control methods are compared. They can be divided into classical control (PID regulator with feedforward) which assumes a Linear Time Invariant (LTI) system, and Adaptive Control which tackles the problem of time-varying system.

There exist a number of vision-based [220, 221] and visual servoing altitude control techniques [222, 223] which can be implemented. However, they are not investigated in this work, because the typical mission is assumed to be outdoors, in higher altitudes (up to

hundreds of meters) and in rural area or wilderness. In such environments it is hard to guarantee sufficient amount of distinct features in the camera image during whole mission, which would affect the control performance.

7.3.1 Classical Control

The most common altitude control system used in multirotor UAVs is a PID regulator with feedforward terms. Feedforward is set manually and gives a baseline of thrust to be applied to keep a UAV in constant altitude because the PID feedback control input is small in comparison with nominal thrust. Although very simple to implement, this technique often does not provide acceptable performance because the real system is time-variant. In other words, if the controller is tuned for a full battery pack, performance degrade as the battery is depleted, as can be seen in Fig. 7.6.

The main advantage of classical control is simplicity (no system model required, can be tuned experimentally). However, it can be used only for applications with weak requirements on altitude control.

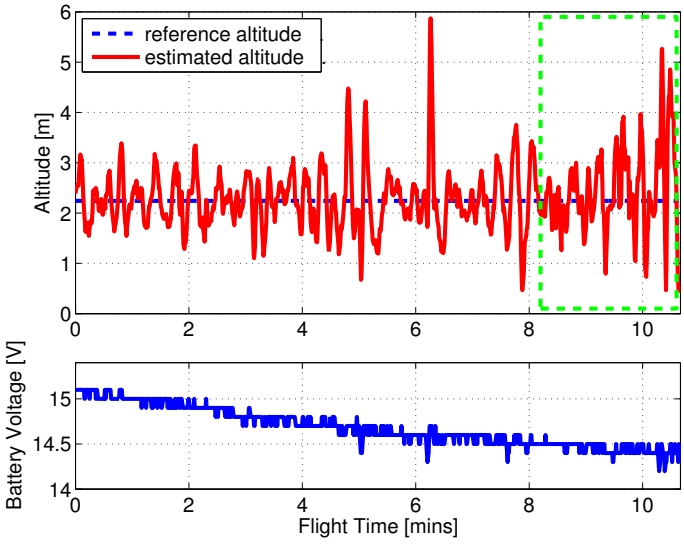


Fig. 7.6: Above: altitude tracking during autonomous outdoor hexarotor flight–the green box indicates increasing oscillations. Below: closed-loop battery voltage.

7.3.2 Adaptive Control

Unlike simple PID control, adaptive controllers require a kind of system model. Although models of multirotor dynamics are known [224, 225], identifying the model is a tedious process. The actuator, consisting of an ESC and a BLDC motor (described in Sec. 7.2.2), can be identified separately from the rest of the system [218] if necessary.

Adaptive control approaches can be divided as follows:

1. **PID + Adaptive Feedforward.** The aforementioned PID controller can be augmented with adaptive feedforward. Adaptively estimated is nominal thrust, required for hover. Full model and more details are given in the Paparazzi documentation [20]. A Kalman filter with a kinematic model is used, so no knowledge about the multirotor model is necessary. However, the controller still has to be tuned for a specific airframe.

2. **Model Predictive Control.** Model predictive control (MPC) is another option for altitude control. Although promising better performance, it requires a full model of the UAV, which can be difficult to obtain [226].

3. **Sliding Mode Control.** Another popular control solution is sliding mode control [227, 228]. Again, an identified model of the UAV is required.

7.4 Battery State-Of-Charge-Based Controller

The State-Of-Charge-based controller extends the classical PID controller with constant feedforward (see Sec. 7.3.1) with gain scheduling compensating for battery dynamics. The scheduling is based on the actual SOC of the battery. The advantage is that no state-space model or a transfer function of the system is needed, only the battery dynamics has to be measured.

Mathematically it can be expressed as (assuming a PID controller with feedforward):

$$u(t) = \left(k_p e + k_I \int_0^t e(\tau) d\tau + k_D \frac{de(t)}{dt} + k_{ff} \right) k_b(SOC).$$

Shown in Fig. 7.7 is the block diagram of the proposed controller. The feedback section and tilt compensation is unchanged, and the battery compensation block (function $k_b(SOC)$) is added to the feedfoward line. The battery dynamics measurements are described in next section. The main advantage is that the controller compensates for changes in the time-varying system and its performance is consistent. The actual function $k_b(SOC)$ is to be characterized in the next section.

7.5 Battery Dynamics Measurements

In order to design the SOC-based controller, the battery dynamics must be measured. For this reason an experimental test bench was built and batteries were characterized, as described in this section.

7.5.1 Instrumentation

In order to measure thrust of the actuators and SOC of the battery, a testbench based on Chéron et al. [218] was developed. The data acquisition and interface to sensors is done by an Arduino MEGA-2560 with a custom expansion board. The testbed solid model is shown in Fig. 7.8.

Force (Measurement Specialities FC2231) and current (Allegro MicroSystems ACS756SCA-050B) analog sensors are filtered with resistor-capacitor filters to prevent excessive noise. The force sensor error is $\pm 3.25\%$, the current sensor error is $\pm 5\%$ according to the datasheets. The whole system captures data at 12Hz and sends them to the computer via USB, with post-processing done in MATLAB. The actuator consists of a Mystery 40A ESC, T-motor MT2814 KV770 motor, and 12×3.8" propeller, which is a suitable combination for a quad or hexarotor. The ESC is controlled from an Arduino PWM port at 50Hz rate. The complete testbed prepared for the measurement is shown in Fig. 7.9.

Two different 4-cell LiPo batteries were used: Zippy 5000mAh 40C and MaxAmps 11000mAh 40C. Both batteries are shown in Fig. 7.10.

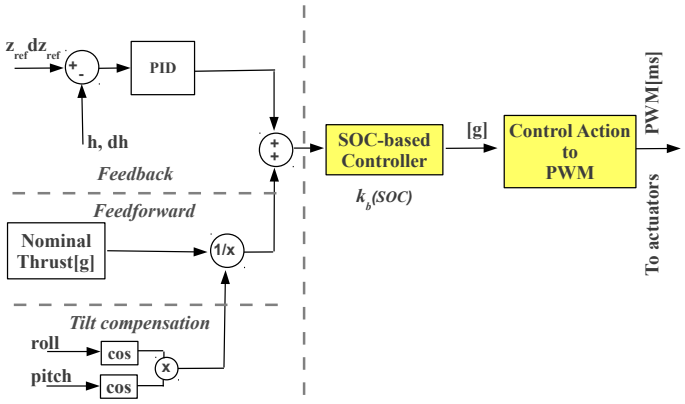


Fig. 7.7: Battery-based altitude control diagram with battery compensation block.

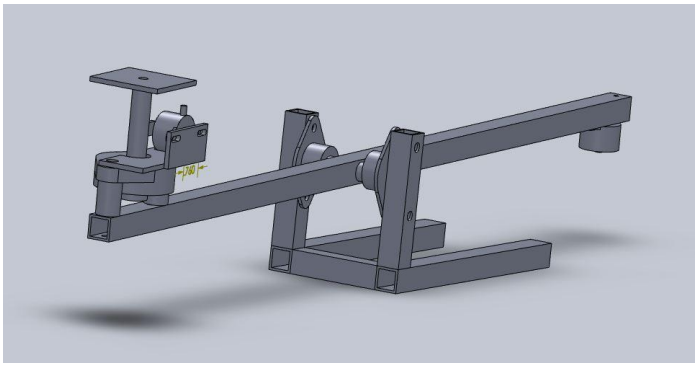


Fig. 7.8: 3D Model of the motor testbench apparatus.

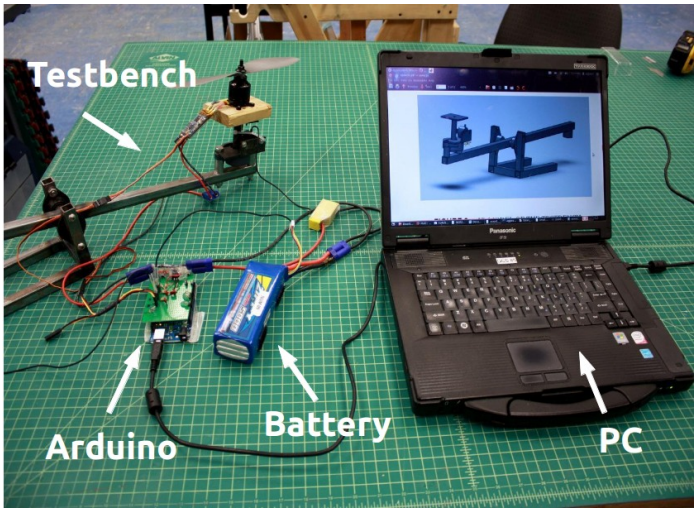


Fig. 7.9: Experimental setup: testbench, Arduino board and computer.

Fig. 7.10: Zippy 5000mAh 40C 4-cell battery (above); MaxAmps 11000mAh 40C 4-cell LiPo battery (below).

7.5.2 Experimental Setup

In order to measure the battery dynamics, the following experiment was conducted. The actuator was set to a constant throttle of 55% ($PWM = 1.54$ms), which produces around 1000g of thrust for a fully charged battery. The change in throttle was measured as well as current and battery voltage. The experiment ran until the battery was depleted, which was clearly marked by a sudden decrease in battery voltage, right before the collapse (i.e. until the closed-loop voltage dropped below 12V).

Measured battery discharge current is shown in Fig. 7.11. The current was integrated as:

$$Q_i(t) = \int_0^t i_b(\tau) d\tau. \tag{7.15}$$

The percentage of remaining SOC is defined [229] as:

$$SOC(t) = 100 \left(\frac{Q_c - Q_i(t)}{Q_c} \right), \tag{7.16}$$

where Q_c is the maximal current capacity present when $SOC = 100\%$. Note that for 11Ah battery(MaxAmps): $Q_c = 11[\text{A}] * 3600[\text{s}] = 39600[\text{C=A}\times\text{s}]$. For 5Ah battery (Zippy):

$Q_c = 18000$[C].

Note that as the batteries were new, their nominal maximal capacity was used. The discharge experiment is shown in Fig. 7.12. The thrust is proportional to the battery power output $F(grams) \propto P(Watts) = I(Ampers) \times U(Volts)$.

To obtain conversion from grams of thrust to PWM command, the actuator must be characterized [218]. Such conversion is necessary for the experimental verification of the controller, when thrust setpoint (instead of an altitude setpoint) is used (see Sec. 7.5.3). The actuator was connected to a power supply, simulating a fully charged battery. PWM commands were changed to cover whole admissible range of the ESC (1.1–1.9ms) and produced thrust was measured. The data for each command step was averaged to obtain the resulting plot in Fig. 7.13. The measured data was approximated with a linear function ($y = ax + k$, $a = 4.0323 \times 10^{-4}, k = 1.1722, x \in (300, 1700)$) to avoid nonlinearity.

The noisy force and current measurements were interpolated using least-squares approximation to obtain dependency of thrust on SOC. The end of the battery pack is identified as when the Closed-Loop Voltage (CLV) drops below 12V. Knowing the 5% measurement error of the current sensor, the estimated SOC aligns well with the battery capacity. Due to the inherent error in measurements, the flight should be terminated at 10% SOC, so the observed voltage drop does not occur.

The dependency of produced thrust on battery SOC is shown in Fig. 7.14. The overall change in thrust (100%-10% SOC) is about 20% for MaxAmps battery and about 25% for Zippy battery. The thrust curve is almost linear on this range of SOC, except for an exponential drop from fully charged battery to 90%, and then another drop before the battery collapses (below 10% SOC). The measured thrust was interpolated using least-squares spline approximation with coefficients from Table 7.1, the two knots were chosen to separate the almost linear piece and two highly nonlinear parts.

Assuming that the change in thrust over the SOC is identical for whole range of throttle, it can be normalized. The normalized spline approximation is shown in Fig. 7.15. To obtain function $k_b(SOC)$ ("Thrust-Bonus") of the battery compensator, the normalized throttle

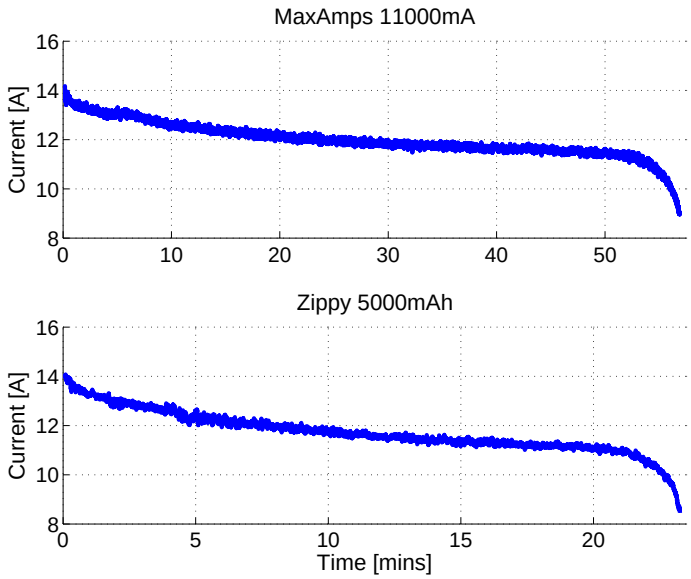


Fig. 7.11: Measured current during battery discharge.

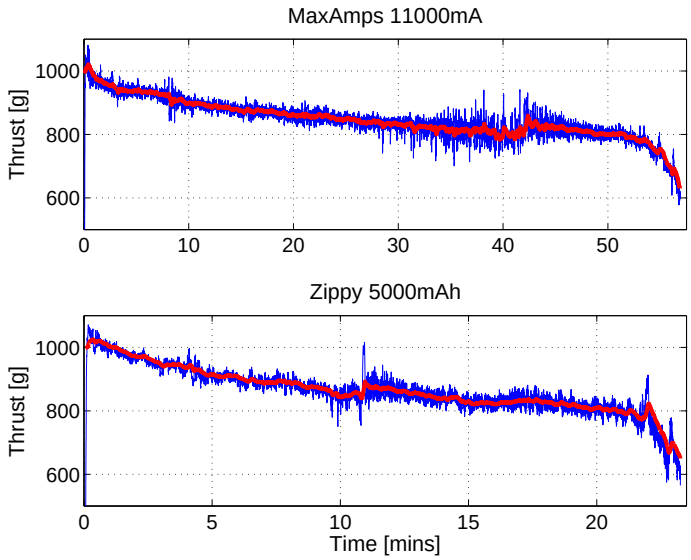


Fig. 7.12: Thrust variations during battery discharge (blue: raw data, red: filtered data). Filtered with Exponential Moving Average (EMA) filter, $\alpha = 0.01$.

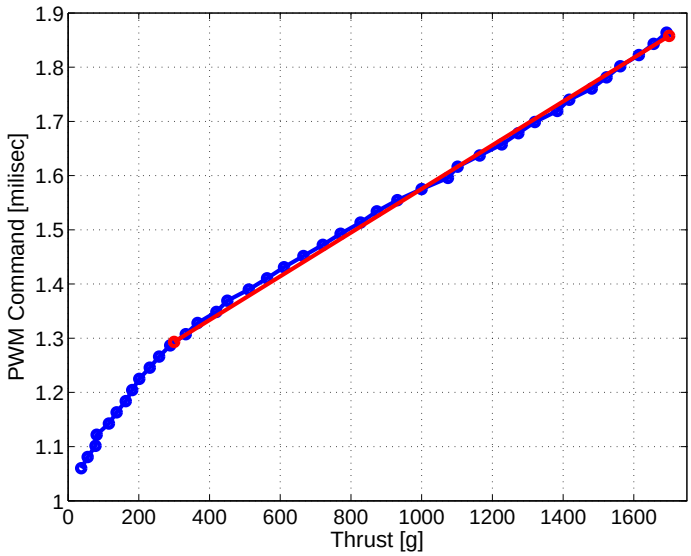


Fig. 7.13: Dependency of thrust on PWM command (Mystery 40A ESC, T-motor MT2814 KV770 actuator and 12×3.8 propeller), blue: measured data, red: linear approximation.

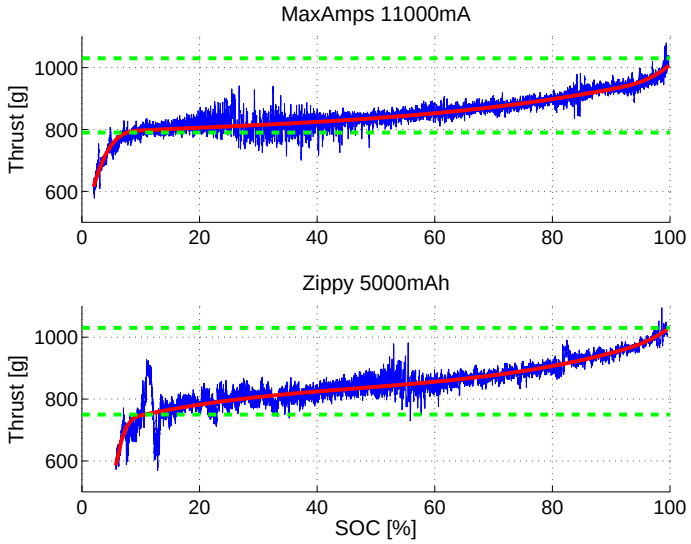


Fig. 7.14: Thrust dependency on SOC (blue: raw data, red: least-square spline approximation, green: thrust at 10% and 90% SOC)

Table 7.1: Least-squares spline approximation for thrust measurements.

Order	# Knots	Knot 1	Knot 2
4	2	10% SOC	90% SOC

Battery	c_1	c_2	c_3	c_4	c_5	c_6
MaxAmps	615.45	791.79	819.87	836.18	955.19	1006.53
Zippy	585.81	735.03	859.25	811.60	983.17	1025.00

curve must be inverted.

To avoid computing a nonlinear curve, the inverted thrust bonus is approximated with a piecewise linear function ($y = ax + k$), divided into four segments. The original and linearised curve is shown in Fig. 7.16, the parameters of piecewise linear function, including root-mean-square (RMS) error of the approximation, are in Table 7.2.

Having these models it is possible to implement the proposed controller.

7.5.3 Experimental Verification

The proposed controller was implemented on Arduino board in order to verify the controller performance on the testbench. The controller performance was measured from full battery to 10% SOC to avoid the voltage drop. Both produced thrust and battery output power were measured. The reference thrust was arbitrarily set at the beginning of the experiment ($T_{ref} = 900[g]$ for Zippy battery and $T_{ref} = 1100[g]$ for the MaxAmps battery). The measured thrust was smoothed with EMA filter ($\alpha = 0.001$).

The results for Zippy 5000mAh battery is shown in Fig. 7.17, for MaxAmps 11000mAh battery in Fig. 7.18. The average error of the controller is shown in Table 7.3. The error is calculated as the RMS error of EMA smoothed thrust from the reference thrust divided by the reference thrust ($e = RMS_T^{EMA}/T_{ref} * 100[\%]$).

The controller performed well in both cases (error under 6%), however MaxAmps battery controller provided dramatically better results (error under 3%). This is because the thrust gain curve for this battery (see that Fig. 7.16 is more linear and follows more closely

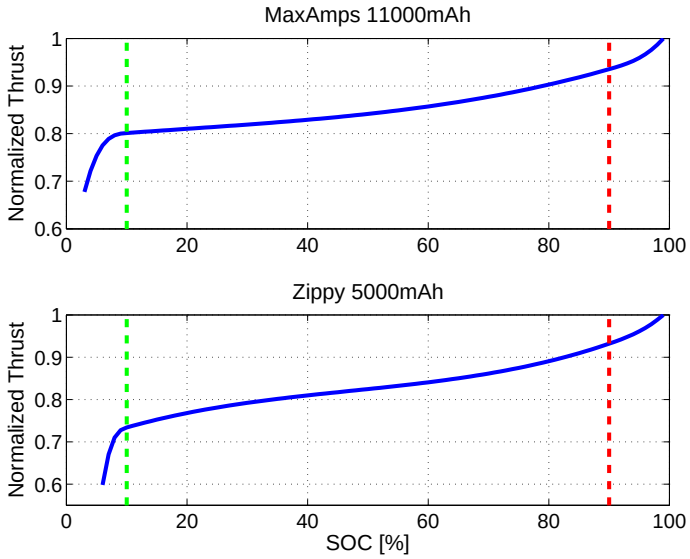


Fig. 7.15: Normalized spline approximation of dependency of thrust on PWM command (green: 10% mark, red: 90% mark).

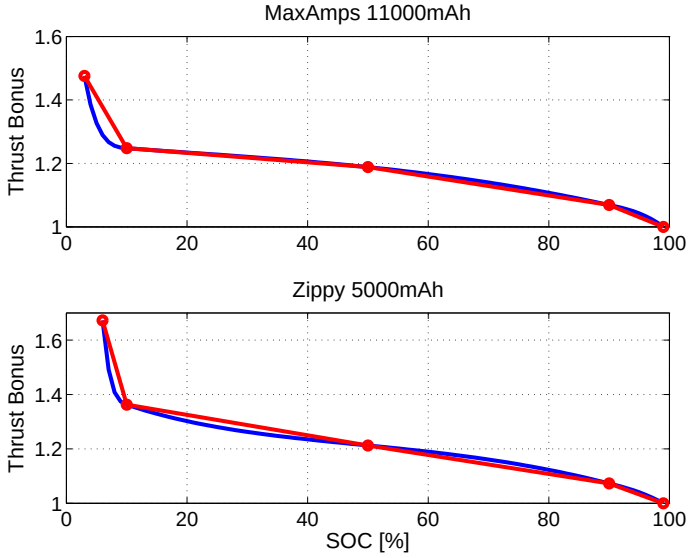


Fig. 7.16: Inverted nominal thrust (blue) and its piecewise-linear approximation (red).

Table 7.2: Piecewise-linear approximation of the thrust bonus.

MaxAmps 11000mAh

Segment	Min.SOC	Max.SOC	α	K
1	0	10	-0.0326	1.5737
2	11	50	-0.0015	1.2630
3	51	90	-0.0030	1.3367
4	91	100	-0.0069	1.6900

Zippy 5000mAh

Segment	Min.SOC	Max.SOC	α	K
1	0	10	-0.0775	2.1380
2	11	50	-0.0037	1.4005
3	51	90	-0.0035	1.3880
4	91	100	-0.0073	1.7300

Battery	RMS Error
MaxAmps 11000mAh	33.249
Zippy 5000mAh	32.222

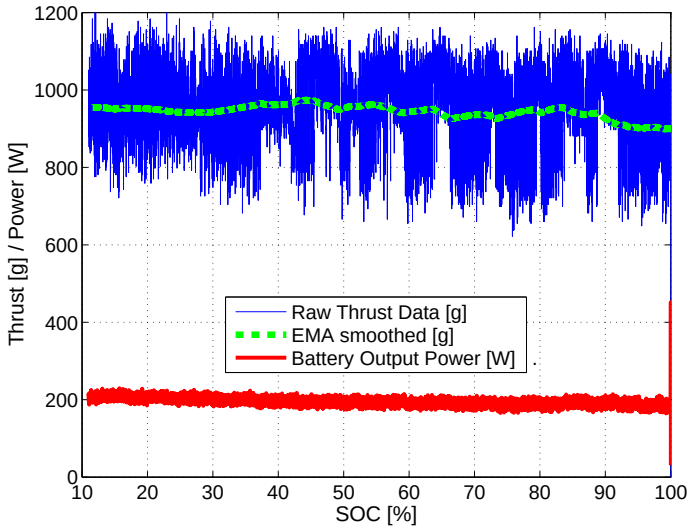


Fig. 7.17: Laboratory test of the controller with Zippy 5000mAh battery.

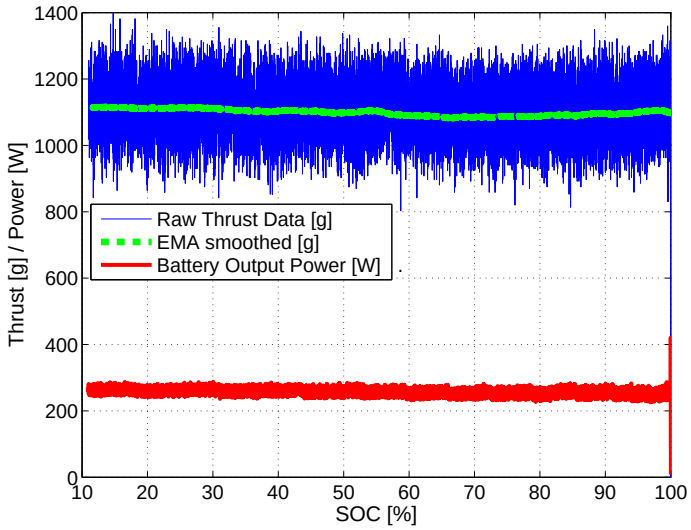


Fig. 7.18: Laboratory test of the controller with MaxAmps 11000mAh battery.

the piecewise linear approximation i.e. the battery dynamics are more linear between 100% and 10% SOC).

7.6 Flight Verification

The controller was tested on an AggieAir hexarotor platform (shown in Fig. 2.7). The platform uses same actuators as were used in the laboratory experiment in Sec. 7.5, and flies with a pair of MaxAmps 11Ah batteries.

To implement and verify the controller in flight, two obstacles must be overcome:

1. Initial SOC of the battery has to be known (it cannot be assumed that a fully charged battery pack is used);

2. Current has to be measured during flight.

Table 7.3: Laboratory experiment error.

Battery	Nominal Thrust[g]	Error[%]
Zippy	900	10
MaxAmps	1100	3

To solve both of the above issues, a battery monitoring system was designed.

7.6.1 Battery Monitoring Subsystem

The battery monitoring system serves as an interface between the autopilot and the battery. It measures battery bus current, voltage and calculates SOC, which is required for proper gain scheduling of the altitude controller. A diagram of such subsystem is shown in Fig. 7.19.

For the actual hardware realization a similar current sensor as for experimental verification was used (Allegro MicroSystems ACS756SCA-150B), just with higher rating (max 150A) as the current consumption was expected to be up to 130A peak. Both voltage and output of the current sensor were measured using a 10-bit ADC converted with I2C bus (Analog Devices AD7997). The software part of the subsystem (SOC calculation) was implemented on the current autopilot board. An assembled board is shown in Fig. 7.20.

7.6.2 SOC Estimation

It is possible to infer the initial SOC of a battery from its OCV. If there is no load placed on the battery, the battery closed-loop voltage (CLV) eventually converges to OCV (details

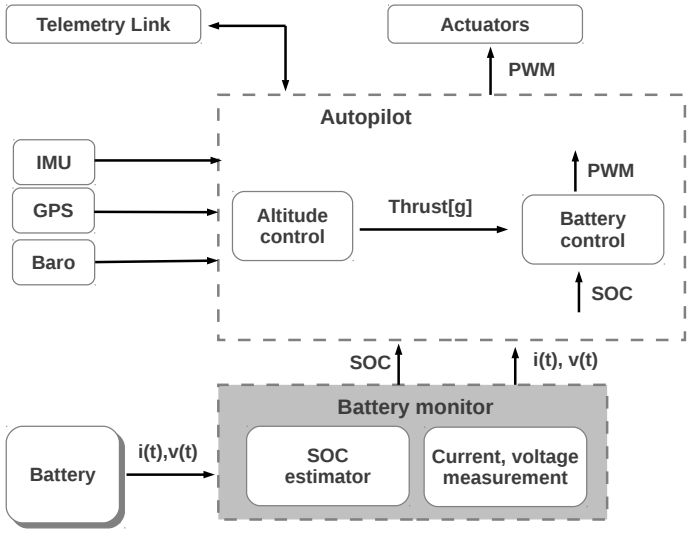


Fig. 7.19: Diagram of the developed battery monitoring subsystem.

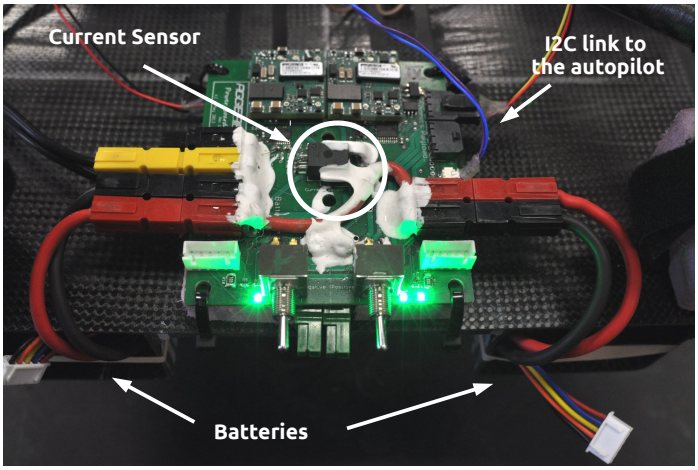


Fig. 7.20: An assembled battery monitoring system, mounted on a hexarotor platform: ready to fly.

can be found in Sec. 7.2.1). A square wave discharge experiment was conducted. MaxAmps 11Ah battery was placed on the testbench, the motor was run at 50% throttle command for a minute, then it was switched off for one minute to allow the voltage to recover. This cycle continued until the closed-loop voltage dropped below a safety threshold of 13V. The current was integrated and the SOC was mapped to the OCV, as shown in Fig. 7.21. Note that for two batteries in parallel, the capacity doubles but the mapping is still valid.

To avoid computing a nonlinear function representing OCV on SOC dependency (Eq. (7.7)) in flight, a piecewise linear approximation was used instead. This approximation is shown in Fig. 7.22. A lookup table was calculated from the obtained values and stored in the autopilot memory.

Note that if older batteries are used, their capacity is lower and the discharge test would have to be repeated to find the actual capacity. The measured OCV on SOC dependency would however stay the same. For battery pairs, assume using a pair of equally old/new batteries, because using old and new batteries together is not recommended by the battery manufacturers.

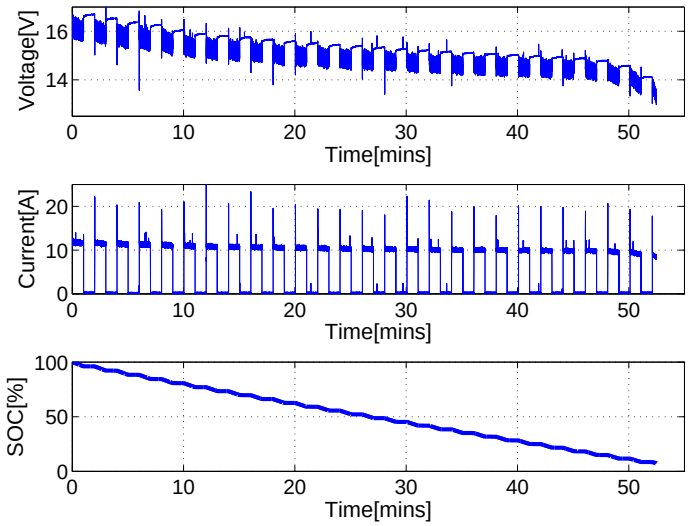


Fig. 7.21: Square wave discharge experiment, MaxAmps 11Ah battery. Top: measured closed loop voltage, middle: measured current, bottom: calculated SOC.

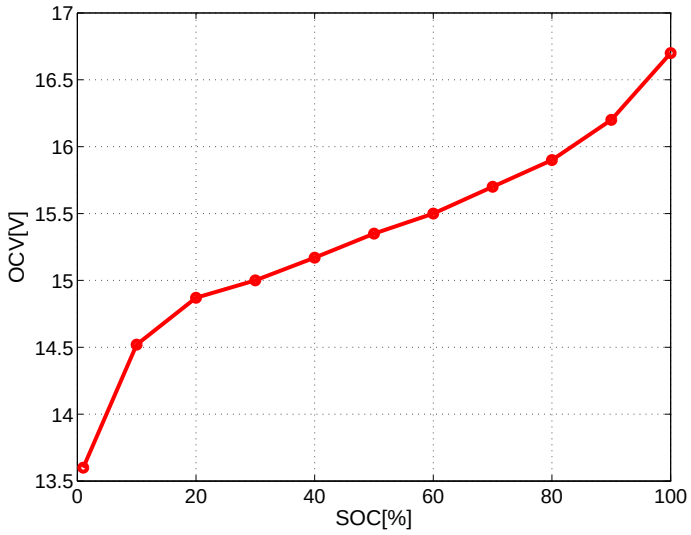


Fig. 7.22: Piecewise-linear approximation of OCV-to-SOC dependency for MaxAmps 11Ah battery.

7.6.3 Flight Data

After implementing the State-of-Charge controller and the battery monitoring system into the autopilot, and properly calibrating the SOC estimation, an actual outdoor flight test was conducted.

The flight consisted of a manual take-off and consequential switch to altitude hold mode. The altitude was held using PID control with constant feedforward, and the SOC-based controller. The flight time was 30 minutes and results are shown in Fig. 7.23 and Fig. 7.24. Overall the controller is able to keep the altitude within ±1m from the reference.

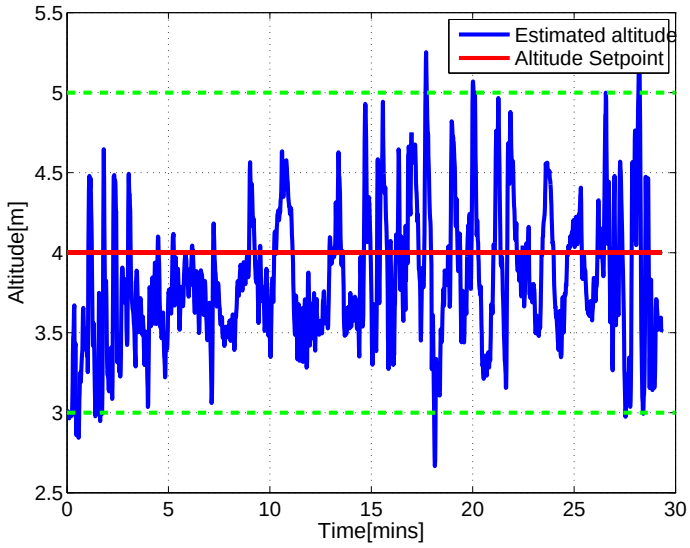


Fig. 7.23: Altitude hold with the AggieAir hexarotor within ±1m using the controller. Blue: estimated altitude, red: altitude setpoint, green: ±1m threshold.

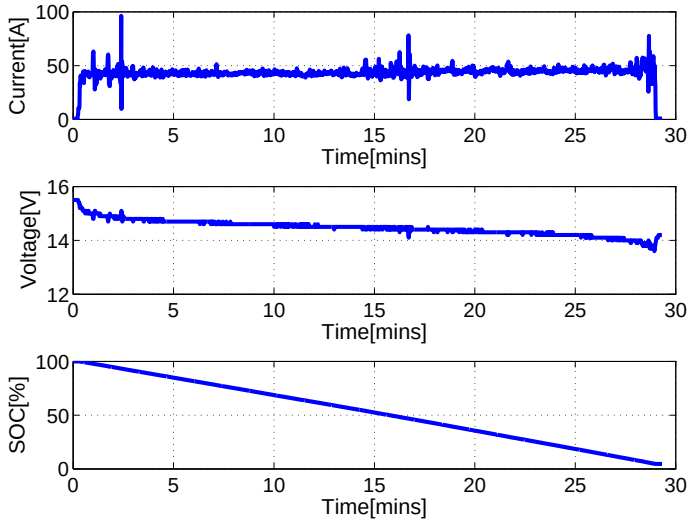


Fig. 7.24: Flight test with the AggieAir hexarotor. Top: total current consumption, middle: battery voltage, bottom: estimated SOC.

7.7 AERIS Significance of Battery Estimation and Chapter Summary

In AERIS, the more knowledge of the state of the airborne system, the better the mission quality. In this chapter a battery State-of-Charge (SOC)-based altitude controller was developed and experimentally verified. The dynamics and model of both LiPo batteries and actuators consisting of brushless DC motors and electronic speed controller were described. The proposed controller adds a SOC-dependent gain, and provides satisfactory control in situations when feedback about actuator thrust or RPM is not available.

After characterizing battery dynamics (dependency of nominal thrust on the SOC of the battery, relation between open circuit voltage and SOC of the battery), a complete description and implementation of the controller was shown. The controller was first tested in laboratory experiment, where it showed control errors of 3% and 10% for MaxAmps 11Ah and Zippy 5Ah batteries. The linearity of the battery dynamics affects the control performance, showing the higher-quality batteries (MaxAmps) are more linear.

Consequently, the controller was implemented to the autopilot of AggieAir hexarotor and tested in real flight. To do this, an additional battery monitoring subsystem was developed to correctly estimated the SOC of the batteries. The conducted flights prove that the controller is able to keep the airframe within 1m from the desired setpoint.

Chapter 8

Conclusion

This thesis presents background about personal remote sensing (PRS) and the AggieAir unmanned aerials system (UAS) platforms in Chapter 2, and the coming technologies of cyber-physical systems (CPSs), with motivating examples of how the two technologies can work together in Chapter 3. Chapter 4 shows how architecture drives the values of a system, which drives the functionality, risks, and rewards of the system dynamics and interactions with the environment around it, as well as introducing AERIS, the architecture for ethical remote information sensing. Enabled by AERIS, Chapter 5 shows detailed system implantation details about a high-dataworthiness payload control structure, emphasizing the idea of data as a mission, and equating data mission assurance partly with payload resilience. Chapter 6 shows how low-cost inertial sensors are a good option for PRS UASs, and how fractional calculus can help glean even more information from then. Then, in Chapter 7, advanced battery management techniques are shown to increase the dataworthiness of a vertical takeoff and landing craft by compensating for power loss depending on the lithium-polymer power system, adding a SOC-dependent gain, and providing satisfactory control in situations when feedback about actuator thrust or RPM is not available.

Cyber-physical systems are the future of solving real-world problems such as water management and alternative energy production. By using sUASs as sensors, better data can be collected and used to make informed control decisions, transforming difficult, complex, or abstract problems into closed-loop systems that can be analyzed and controlled.

Overall, the outstanding research question is: how will a DMQ-based UAS architecture be best tested, integrated, and implemented in an active airspace? Future work lies in many different directions.

- Future work with architectures and AERIS exists in all presented avenues of policy and engineering to create workable architectures (both inside and outside UASs) that protect rights, uphold safety, and better the world for both man and machine. Specifically this can include setting up flights to test systems, payloads, and safety performance estimations, and to show the benefits of interaction in real mission scenarios. This includes axiomatic interpretations, as well as implementation details such as formal methods, software and security standards, and standards for training and crew certification.

- Future work with PRS payload management software will include a larger framework for holistically testing both airframe and payloads in-the-loop during development, improvements to ISaAC allowing more cognitive processing of flight situational awareness to improve overall PRS missions, and increasing data mission assurance.

- Future work with navigation estimation includes noise modeling of physical navigation sensors, testing of fractional-order complementary filters on real flight data, and eventually implementation and flight with a real UAS.

- Future work with sUAS battery estimation includes collecting data with many different flights, with varying ages of batteries and actuators, showing that the proposed estimation techniques will or will not work over repeated missions and be useful in the long-term.

The future of UASs is undoubtedly bright. While the current public perception of UASs is one of espionage and warfare, they will become more accepted into domestic use as their potential value becomes apparent and as the airspace rules change to include them. While current regulations of UASs are restrictive and limited in the US, soon UASs will become available for regular use as standards for certification and airworthiness are developed.

Architectures like AERIS are of critical importance as the capabilities and demands for automation increase. As the human population grows and resources are ever more taxed,

techniques like sUASs for remote sensing will become of vital importance to conservation and long-term sustainability.

Along with these standards, mission quality metrics are needed to determine if the UAS is truly in need of the airspace. Along with ethics (privacy by design), an AERIS-compliant airspace access requirement architecture will allow civil flights of many kinds with minimal concern for right violations, allowing humans and unmanned robotic systems to peacefully coexist and grow together.

At the time of this writing, there is only one commercial group authorized for UAS operations within the 50 United States [1]. According to the FAA roadmap [230], integration with UAS into the NAS will be in a gradual manner in the next 5 to 10 years, with many improvements made before 2028.

In these next 15 years, the development of unmanned technology will enable UASs to be mobile sensors as well as actuators, actively exerting control in optimal locations for precision actuation, and cognitive control systems for large-scale complex systems that were previously impractical to control. Management of crisis situations such as food and water shortages, energy production, floods, and nuclear disasters can be assisted by sUASs. Difficult problems can be solved with cyber-physical systems: inexpensive sUASs as flying sensors and actuators within the closed loops of cyber-physical control.

References

[1] US Federal Aviation Administration, "One giant leap for unmanned-kind." [Online]. Available: http://www.faa.gov/news/updates/?newsId=73118

[2] United States Congress, "FAA Modernization and Reform Act of 2012," 2012.

[3] P. Oh, "Foreward," in *Remote Sensing and Actuation Using Unmanned Vehicles*. Hoboken, NJ: John Wiley & Sons, 2012, pp. xxi–xxii.

[4] J. Ward, "Space-time adaptive processing for airborne RADAR," 1995.

[5] R. A. Ferrare, C. A. Hostetler, J. W. Hair, A. Cook, D. Harper, S. P. Burton, M. D. Obland, R. Rogers, A. J. Swanson, A. D. Clarke, C. S. McNaughton, Y. Shinozuka, J. Redemann, J. M. Livingston, P. B. Russell, C. A. Brock, D. A. Lack, K. D. Froyd, J. A. Ogren, B. Andrews, A. Laskin, R. Moffet, M. K. Gilles, A. Nenes, T. L. Lathem, and P. Liu, "Airborne high spectral resolution lidar aerosol measurements during ARCTAS," *American Geophysical Union Fall Meeting Abstracts*, p. A164, 2009.

[6] J. A. Shaw, J. A. Churnside, J. J. Wilson, N. E. Lerner, R. R. Tiensvold, P. E. Bigelow, and T. M. Koel, "Airborne lidar mapping of invasive lake trout in Yellowstone lake," *in Proceedings of the 24th International Laser Radar Conference*, 2008.

[7] J. A. Shaw, N. L. Seldomridge, D. L. Dunkle, P. W. Nugent, L. H. Spangler, J. J. Bromenshenk, C. B. Henderson, J. H. Churnside, and J. J. Wilson, "Polarization lidar measurements of honey bees in flight for locating land mines," *Optics Express*, vol. 13, no. 15, pp. 5853–5863, 2005.

[8] L. Di and Y. Chen, "Autonomous flying under 500 USD based on RC aircraft," in *ASME/IEEE International Conference on Mechatronic and Embedded Systems and Applications*, vol. 2011, no. 54808. ASME, 2011, pp. 929–936.

[9] "Infrared Cameras Inc." [Online]. Available: http://www.infraredcamerasinc.com/

[10] "Goodrich ISR Systems." [Online]. Available: http://www.sensorsinc.com/

[11] C. Coopmans, B. Stark, and C. M. Coffin, "A payload verification and management framework for small UAV-based personal remote sensing systems," in *Proceedings of the 2012 Int. Symposium on Resilient Control Systems (ISRCS2012)*. IEEE, Aug. 2012, pp. 184–189.

[12] J. Berni, P. J. Zarco-Tejada, L. Suarez, and E. Fereres, "Thermal and narrowband multispectral remote sensing for vegetation monitoring from an unmanned aerial vehicle," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 47, no. 3, pp. 722–738, Mar. 2009.

[13] J. Hunt, W. D. Hively, S. J. Fujikawa, D. S. Linden, C. S. T. Daughtry, G. W. McCarty, and E. R. Hunt Jr., "Acquisition of NIR-green-blue digital photographs from unmanned aircraft for crop monitoring," *Remote Sensing*, vol. 2, no. 1, pp. 290–305, Jan. 2010.

[14] A. M. Jensen, T. Hardy, M. Mckee, and Y. Q. Chen, "Using a multispectral autonomous unmanned aerial remote sensing platform (AggieAir) for riparian and wetland applications," in *Proceedings of the IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, Aug. 2011, pp. 3413–3416.

[15] A. M. Jensen, "Innovative payloads for small UAS-based personal remote sensing and applications," Ph.D Dissertation, Utah State University, 2014.

[16] H. Chao, M. Baumann, A. M. Jensen, Y. Chen, Y. Cao, W. Ren, and M. McKee, "Band-reconfigurable multi-UAV-based cooperative remote sensing for real-time water management and distributed irrigation control," in *Proceedings of the IFAC World Congress*, Seoul, Korea, Jul. 2008.

[17] J. D. Barton, "Fundamentals of small unmanned aircraft flight," *Johns Hopkins APL Technical Digest*, vol. 31, no. 2, 2012.

[18] A. M. Jensen, Y. Han, and Y. Chen, "Using aerial images to calibrate the inertial sensors of a low-cost multispectral autonomous remote sensing platform (AggieAir)," pp. II–555–II–558, 2009.

[19] "MosaicMill EnsoMOSAIC." [Online]. Available: http://www.ensomosaic.com/

[20] Paparazzi Forum, "Open-source Paparazzi UAV project." [Online]. Available: http://paparazzi.enac.fr/

[21] C. Coopmans, B. Stark, A. M. Jensen, Y. Chen, and M. McKee, "Cyber-physical systems enabled by small unmanned aerial vehicles," in *Handbook of Unmanned Aerial Vehicles*, K. P. Valavanis and G. J. Vachtsevanos, Eds. Springer, 2014.

[22] Z. Jiao, Y. Chen, and I. Podlubny, "Introduction," in *Distributed-Order Dynamic Systems*. Springer London, 2012, pp. 1–10.

[23] C. G. Rieger, S. Member, D. I. Gertman, and M. A. Mcqueen, "Resilient control systems: next generation design research," in *Proceedings of the 2nd Conference on Human System Interactions*, 2009, pp. 632–636.

[24] D. W. Casbeer, D. B. Kingston, R. W. Beard, and T. W. McLain, "Cooperative forest fire surveillance using a team of small unmanned air vehicles," *International Journal of Systems Science*, vol. 37, no. 6, pp. 351–360, May 2006.

[25] L. Merino, "Cooperative fire detection using unmanned aerial vehicles," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, no. April, 2005, pp. 1884–1889.

[26] A. Ollero, J. R. Martínez-de Dios, and L. Merino, "Unmanned aerial vehicles as tools for forest-fire fighting," *Forest Ecology and Management*, vol. 234, p. S263, Nov. 2006.

[27] G. Zhou, C. Li, and P. Cheng, "Unmanned aerial vehicle (UAV) real-time video registration for forest fire monitoring," in *Proceedings of the 2005 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, vol. 31, no. 1, 2005, pp. 1803–1806.

[28] E. Albano, "Critical behaviour of a forest fire model with immune trees," *Journal of Physics A: Mathematical and General*, vol. 881, 1999.

[29] G. Yang and Y. Li, "Design and realization of the dynamic data driven system of forest fire simulation-the case study of Beijing forest fire prevention system," *Recent Advances in Computer Science and Information Engineering*, vol. 129, pp. 559–568, 2012.

[30] C. Tricaud and Y. Chen, *Optimal Mobile Sensing and Actuation Policies in Cyber-physical Systems*. Springer, 2011.

[31] S. G. Bajwa and E. D. Vories, "Spectral response of cotton canopy to water stress," in *Proceedings of the ASAE Annual Meeting*, Jul. 2006.

[32] NSF, "Cyber-physical systems program solicitation." [Online]. Available: http://www.nsf.gov/pubs/2011/nsf11516/nsf11516.htm

[33] "NSF sustainable energy pathways (SEP) program solicitation." [Online]. Available: http://www.nsf.gov/pubs/2011/nsf11590/nsf11590.htm,

[34] "Google Earth." [Online]. Available: http://earth.google.com/

[35] D. Dye, R. Sims, I. Hamud, R. Thompson, and E. Griffith, "Algae bioremediation and biofuels at the Logan, Utah wastewater treatment facility," in *First International Conference on Algal Biomass, Biofuels & Bioproducts*. Utah State University Sustainable Waste-to-Bioproducts Engineering Center, Poster, 2010.

[36] N. D. Chea, K. S. McCulloch, J. A. Powell, and R. C. Sims, "Daphnia-algae modeling of the Logan wastewater lagoons," in *Biological Engineering Regional Conference*. Utah State University Biological Engineering Department, IBE Poster, 2010.

[37] S. Consigny, "Rhetoric and madness: Robert Pirsig's inquiry into values," *Southern Speech Communication Journal*, vol. 43, no. 1, pp. 16–32, Dec. 1977.

[38] G. F. Cooper, "The computational complexity of probabilistic inference using Bayesian belief networks," *Artificial intelligence*, vol. 42, no. 2, pp. 393–405, 1990.

[39] D. Griffin, P. Shaw, and R. Stacey, "Knowing and acting in conditions of uncertainty: a complexity perspective," *Systemic Practice and Action Research*, vol. 12, no. 3, 1999.

[40] J. H. Brown, V. K. Gupta, B.-L. Li, B. T. Milne, C. Restrepo, and G. B. West, "The fractal nature of nature: power laws, ecological complexity and biodiversity." *Philosophical transactions of the Royal Society of London. Series B, Biological sciences*, vol. 357, no. 1421, pp. 619–26, May 2002.

[41] R. Gorenflo and F. Mainardi, "Fractional calculus and stable probability distributions," *Archives of Mechanics*, vol. 50, no. 3, pp. 1–10, 1998.

[42] R. Tanaka, "Scale-rich metabolic networks," *Physical Review Letters*, vol. 94, no. 16, p. 168101, Apr. 2005.

[43] J. Doyle, "Universal laws and architectures," *CDS 212 Lecture Notes*. [Online]. Available: http://www.cds.caltech.edu/~doyle/wiki/images/1/16/2_DoyleSageLec2_May14_2012.pdf

[44] M. Ferris and U. Treasury, "New email security infrastructure," in *New security Paradigms Workshop*, 1994, pp. 20–27.

[45] J. Doyle, "Rant on Turing." [Online]. Available: http://www.cds.caltech.edu/~doyle/wiki/images/8/84/DoyleRantOnTuring.pdf

[46] L. Chen, S. H. Low, M. Chiang, and J. C. Doyle, "Cross-layer congestion control, routing and scheduling design in ad hoc wireless networks," *Proceedings of the 25TH IEEE International Conference on Computer Communications*, pp. 1–13, 2006.

[47] D. Trossen, "Turing, the Internet and a theory for architecture: a (fictional?) tale in three parts," *ACM SIGCOMM Computer Communication Review*, vol. 42, no. 3, pp. 47–53, 2012.

[48] R. Sterritt, "Autonomic computing," *Innovations in systems and software engineering*, vol. 1, no. 1, pp. 79–88, 2005.

[49] J. B. Ruhl, "The fitness of law: using complexity theory to describe the evolution of law and society and its practical meaning for democracy," *Vanderbilt Law Review*, vol. 49, pp. 1406–1490, 1996.

[50] M. E. Csete and J. C. Doyle, "Reverse engineering of biological complexity." *Science (New York, N.Y.)*, vol. 295, no. 5560, pp. 1664–9, Mar. 2002.

[51] P. Dobransky, "Mind OS: how the operating system of the human mind is the ultimate solution to every personal or business problem," *Unpublished manuscript, Denver, Colo*, 1999.

[52] Q. Zhu and T. Basar, "Robust and resilient control design for cyber-physical systems with an application to power systems," in *Decision and Control and European Control Conference*, 2011.

[53] S. Borkar, "Thousand core chips: a technology perspective," in *Proceedings of the 44th annual Design Automation Conference*. ACM, 2007, pp. 746–749.

[54] G. A. Cory and R. Gardner, *The Evolutionary Neuroethology of Paul Maclean: Convergences and Frontiers*. Greenwood Publishing Group, 2002.

[55] R. L. Isaacson, *The Limbic System*, 2nd ed. Springer, 1982.

[56] F. Caballero, L. Merino, J. Ferruz, and A. Ollero, "Vision-based odometry and SLAM for medium and high altitude flying UAVs," *Journal of Intelligent and Robotic Systems*, vol. 54, no. 1-3, pp. 137–161, Mar. 2009.

[57] F. Bonin-Font, A. Ortiz, and G. Oliver, "Visual navigation for mobile robots: a survey," *Journal of Intelligent and Robotic Systems*, vol. 53, no. 3, pp. 263–296, Nov. 2008.

[58] N. X. Dao, B.-J. You, and S.-R. Oh, "Visual navigation for indoor mobile robots using a single camera," in *Proceedings of the 2005 IEEE International Conference on Intelligent Robots and Systems (IROS)*, Dec. 2005, pp. 1992–1997.

[59] P. Michel, J. Chestnutt, S. Kagami, K. Nishiwaki, J. Kuffner, and T. Kanade, "GPU-accelerated real-time 3D tracking for humanoid locomotion and stair climbing," in *Proceedings of the 2007 IEEE International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2007, pp. 463–469.

[60] J. Kim, E. Park, X. Cui, H. Kim, and W. A. Gruver, "A fast feature extraction in object recognition using parallel processing on CPU and GPU," in *Proceedings of the 2009 IEEE International Conference on Systems, Man and Cybernetics*. IEEE, 2009, pp. 3842–3847.

[61] H. Choset, K. M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun, *Principles of Robot Motion*. The MIT Press, 2005.

[62] Y.-H. Choi and S.-Y. Oh, "Visual sonar based localization using particle attraction and scattering," in *Proceedings of the 2005 IEEE International Conference on Mechatronics and Automation*, Jul. 2005, pp. 449–454.

[63] J. Chestnutt, K. Nishiwaki, J. Kuffner, and S. Kagami, "An adaptive action model for legged navigation planning," in *IEEE-RAS International Conference on Humanoid Robots, Pittsburgh, PA*, Nov. 2007.

[64] J. Antich and A. Ortiz, "Development of the control architecture of an underwater cable tracker: research articles," *International Journal of Intelligent Systems*, vol. 20, pp. 477–498, May 2005.

[65] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: a survey," *ACM Computing Surveys (CSUR)*, vol. 38, no. 4, 2006.

[66] R. R. Brooks and S. S. Iyengar, *Multi-Sensor Fusion: Fundamentals and Applications With Software*. Prentice-Hall, Inc., 1998.

[67] H. P. Moravec, "Sensor fusion in certainty grids for mobile robots," *AI Magazine*, vol. 9, no. 2, pp. 61–74, 1988.

[68] L. Xiao, S. Boyd, and S. Lall, "A scheme for robust distributed sensor fusion based on average consensus," in *Proceedings of the 4th international symposium on Information processing in sensor networks*, no. 9, 2005.

[69] W. Sun, Y. Li, C. Li, and Y. Chen, "Convergence speed of a fractional order consensus algorithm over undirected scale-free networks," *Asian Journal of Control*, vol. 13, no. 6, pp. 936–946, 2011.

[70] "OpenJAUS architecture." [Online]. Available: http://openjaus.com/

[71] T. C. Bressoud and F. B. Schneider, "Hypervisor-based fault tolerance," *ACM Transactions on Computer Systems (TOCS)*, vol. 14, no. 1, pp. 80–107, 1996.

[72] D. Wright and P. de Hert, "An integrated privacy and ethical impact assessment," *Presentation to PRESCIENT conference, Berlin*, 2012.

[73] D. Wright, "A framework for the ethical impact assessment of information technology," *Ethics and Information Technology*, vol. 13, no. 3, pp. 199–226, 2011.

[74] R. C. Arkin, "Governing lethal behavior: embedding ethics in a hybrid deliberative/reactive robot architecture part I: motivation and philosophy," in *Proceedings of the 2008 3rd ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 2008, pp. 121–128.

[75] R. L. Finn and D. Wright, "Unmanned aircraft systems: surveillance, ethics and privacy in civil applications," *Computer Law & Security Review*, vol. 28, no. 2, pp. 184–194, Apr. 2012.

[76] M. Strohmeier, V. Lenders, and I. Martinovic, "On the security of the automatic dependent surveillance-broadcast protocol," *Pre-print*, Jul. 2013. [Online]. Available: http://arxiv.org/pdf/1307.3664v2

[77] I. Moir, A. Seabridge, and M. Jukes, *Civil Avionics Systems*, 2nd ed. John Wiley & Sons, Inc., 2013.

[78] P. P. Narayan, P. P. Wu, D. A. Campbell, and R. A. Walker, "An intelligent control architecture for unmanned aerial systems (UAS) in the national airspace system (NAS)," May 2007.

[79] C. W. Heisey, A. G. Hendrickson, B. J. Chludzinski, R. E. Cole, M. Ford, L. Herbek, M. Ljungberg, Z. Magdum, D. Marquis, A. Mezhirov, J. L. Pennell, T. A. Roe, and A. J. Weinert, "A reference software architecture to support unmanned aircraft integration in the national airspace system," *Journal of Intelligent & Robotic Systems*, vol. 69, no. 1-4, pp. 41–55, Aug. 2012.

[80] B. S. Faiçal, F. G. Costa, G. Pessin, J. Ueyama, H. Freitas, A. Colombo, P. H. Fini, L. Villas, F. S. Osório, P. A. Vargas, and T. Braun, "The use of unmanned aerial vehicles and wireless sensor networks for spraying pesticides," *Journal of Systems Architecture*, vol. 60, no. 4, pp. 393–404, Apr. 2014.

[81] U. Pagallo, "Robots in the cloud with privacy: a new threat to data protection?" *Computer Law & Security Review*, vol. 29, no. 5, pp. 501–508, Oct. 2013.

[82] P. Lin, K. Abney, and G. Bekey, "Robot ethics: mapping the issues for a mechanized world," *Artificial Intelligence*, vol. 175, no. 5-6, pp. 942–949, Apr. 2011.

[83] R. Clarke, "What drones inherit from their ancestors," *Computer Law & Security Review*, vol. 30, no. 3, pp. 247–262, Jun. 2014.

[84] B. Stark, C. Coopmans, and Y. Chen, "A framework for analyzing human factors in unmanned aerial systems," in *Proceedings of the 5th International Symposium on Resilient Control Systems*, Salt Lake City, Utah, USA, 2012, pp. 13–18.

[85] US Federal Aviation Administration, "System safety handbook." [Online]. Available: http://www.faa.gov/regulations_policies/handbooks_manuals/aviation/risk_management/ss_handbook/

[86] US Federal Aviation Administration, "Unmanned aircraft systems test site selection (UASTSS)." [Online]. Available: https://www.fbo.gov/index?s=opportunity&mode=form&id=4eedc2c55ec854f220d031c2f3c4a783&tab=core&_cview=1

[87] C. P. Lai, Y. J. Ren, and C. Lin, "ADS-B based collision avoidance radar for unmanned aerial vehicles," in *Proc of 2009 IEEE MTT-S International Microwave Symposium Digest*, 2009, pp. 85–88.

[88] US Federal Aviation Administration, "FAA's NextGen performance assessment," Tech. Rep., 2011.

[89] D. G. Johnson, *Computer ethics*. DIANE Publishing Company, Dec. 1998, vol. 30, no. 4.

[90] A. Cavoukian, "Privacy by design the 7 foundational principles implementation and mapping of fair information practices," *Information and Privacy Commissioner of Ontario, Canada*, 2009.

[91] N. Leveson, "A new accident model for engineering safer systems," pp. 237–270, 2004.

[92] J. Rasmussen, "Risk management in a dynamic society: a modeling problem," *Safety Science*, vol. 27, no. 2-3, pp. 183–213, Nov. 1997.

[93] N. Leveson, "Completeness in formal specification language design for process-control systems," in *Proceedings of the 3rd workshop on Formal methods in software practice*. New York, New York, USA: ACM Press, 2000, pp. 75–87.

[94] "Safeware engineering corporation: SpecTRM features." [Online]. Available: http://www.safeware-eng.com/softwaresafetyproducts/features.htm

[95] D. Park, "Translation of safety-critical software requirements specification to Lustre," in *Innovations and Advanced Techniques in Computer and Information Sciences and Engineering*. Springer, 2007, pp. 157–162.

[96] P. Bogdan, S. Jain, and R. Marculescu, "Pacemaker control of heart rate variability: a cyber physical system perspective," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 1, no. November, pp. 1–22, 2013.

[97] P. Bogdan and R. Marculescu, "Towards a science of cyber-physical systems design," in *Proceedings of the 2nd International Conference on Cyber-Physical Systems*, vol. 28, no. 4. Ieee, Apr. 2011, pp. 99–108.

[98] L. A. Johnson, "Do-178B, software considerations in airborne systems and equipment certification," *Crosstalk, October*, 1998.

[99] S. Kawaguchi, "Trial of organizing software test strategy via software test perspectives," in *Proceedings of the 7th International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*. IEEE, 2014, p. 360.

[100] P. Hoffman, K. Scarfone, and M. Souppaya, "Guide to security for full virtualization technologies," *National Institute of Standards and Technology (NIST)*, pp. 125–800, 2011.

[101] "Xbox360 Security / Hypervisor / eFuses / Encryption MORE! — Se7enSins Gaming Community." [Online]. Available: http://www.se7ensins.com/forums/threads/xbox360-security-hypervisor-efuses-encryption-more.453621/

[102] "SecurityFocus." [Online]. Available: http://www.securityfocus.com/archive/1/461489

[103] "Trango hypervisor virtualizes Atmel's CAP customizable microcontroller." [Online]. Available: http://ir.atmel.com/releasedetail.cfm?ReleaseID=295313

[104] G. Chowdhary and S. Lorenz, "Control of a VTOL UAV via online parameter estimation," *AIAA Guidance, Navigation, and Control Conference and Exhibit*, pp. 1–15, Aug. 2005.

[105] W. M. Debusk, G. Chowdhary, and E. N. Johnson, "Real-time system identification of a small multi-engine aircraft," in *Proceedings of the AIAA Atmospheric Flight Mechanics Conference*, 2009, pp. 1–15.

[106] A. Kallapur, M. Samal, P. Vishwas, A. Sreenatha, Garratt, and Mathew, "A UKF-NN framework for system identification of small unmanned aerial vehicles," in *Proceedings of the 11th International IEEE Conference on Intelligent Transportation Systems*, 2008, pp. 1021–1026.

[107] "GNU, General Public License, full description." [Online]. Available: https://www.gnu.org/licenses/gpl.html

[108] "ChibiOS/RT." [Online]. Available: http://www.chibios.org/

[109] "OCaml, the Objective Caml." [Online]. Available: http://ocaml.org/

[110] "Pixhawk Project Homepage." [Online]. Available: http://pixhawk.org/

[111] "Creative Commons Licence Properties." [Online]. Available: http://wiki.creativecommons.org/License_Properties

[112] "3DRobotics Homepage." [Online]. Available: http://3drobotics.com/

[113] "NuttX Real-Time Operating System." [Online]. Available: http://nuttx.org/

[114] "ROS (Robot Operating System)." [Online]. Available: http://www.willowgarage.com/

[115] "DJI Phantom 2 Product Homepage." [Online]. Available: http://www.dji.com/product/phantom-2/

[116] Procerus Technologies, "Procerus Technologies Homepage." [Online]. Available: http://www.procerusuav.com/

[117] "Center for Self-Organizing and Intelligent Systems." [Online]. Available: http://www.csois.usu.edu/

[118] "AggieAir Flying Circus." [Online]. Available: http://aggieair.usu.edu/

[119] A. M. Jensen, B. T. Neilson, M. McKee, and Y. Q. Chen, "Thermal remote sensing with an autonomous unmanned aerial remote sensing platform for surface stream temperatures," in *Proceedings of the International Geoscience and Remote Sensing Symp. (IGARSS)*, 2012, pp. 5049–5052.

[120] UAS Task Force and US Department of Defense, "Unmanned aircraft system airspace integration plan," Department of Defense, Tech. Rep. March, 2011.

[121] N. V. Hoffer, C. Coopmans, A. M. Jensen, and Y. Chen, "A survey and categorization of small low-cost unmanned aerial vehicle system identification," *Journal of Intelligent and Robotic Systems*, vol. 74, no. 1-2, pp. 129–145, Oct. 2014.

[122] W. C. Barott, E. Coyle, T. Dabrowski, C. Hockley, and R. S. Stansbury, "Passive multispectral sensor architecture for RADAR-EOIR sensor fusion for low SWAP UAS sense and avoid," in *Position, Location and Navigation Symposium (PLANS)*. IEEE, 2014, pp. 1188–1196.

[123] M. DeGarmo and D. Maroney, "NextGen and SESAR: opportunities for UAS integration," in *26th International Congress of the Aeronautical Sciences*, 2008.

[124] P. Voss, "Code of conduct for the use of small airborne objects on Smith college property," May 2013.

[125] US Federal Aviation Administration, "Fact Sheet - Unmanned Aircraft Systems (UAS)." [Online]. Available: http://www.faa.gov/news/fact_sheets/news_story.cfm?newsId=14153

[126] H. Sheng, H. Sun, and C. Coopmans, "A physical experimental study of variable-order fractional integrator and differentiator," in *Proceedings of the 4th IFAC Workshop Fractional Differentiation and its Applications (FDA2010)*, vol. 2010, no. 1995, Badajoz, Spain, 2010.

[127] S. N. Chau, L. Alkalai, A. T. Tai, and J. B. Burt, "Design of a fault-tolerant COTS-based bus architecture," *IEEE Transactions on Reliability*, vol. 48, no. 4, pp. 351–359, 1999.

[128] S. M. Peter Behr, Wolfgang Bärwald, Klaus Brieß, "Fault tolerance and COTS: next generation of High Performance Satellite Computers," *Proceedings of DASIA 2003*, 2003.

[129] C. M. F. Rubira and A. Romanovsky, "A fault-tolerant software architecture for COTS-based software systems," in *SIGSOFT Software*, 2003, pp. 375–378.

[130] R. Barbosa, N. Silva, and J. Duraes, "Verification and validation of (real time) COTS products using fault injection techniques," in *Proceedings of the 6th International IEEE Conference on Commercial-off-the-Shelf (COTS)-Based Software Systems*, 2007.

[131] M. Pignol, "Methodology and tools developed for validation of COTS-based fault-tolerant spacecraft supercomputers," in *Proceedings of the 13th IEEE International On-Line Testing Symposium*, 2007.

[132] J. López, P. Royo, E. Pastor, C. Barrado, and E. Santamaria, "A middleware architecture for unmanned aircraft avionics," in *Proceedings of the 2007 ACM/IFIP/USENIX international conference on Middleware companion*, 2007.

[133] P. Grace, G. Coulson, G. Blair, B. Porter, and D. Hughes, "Dynamic reconfiguration in sensor middleware," in *Proceedings of the international workshop on Middleware for sensor networks*, 2006, pp. 1–6.

[134] P. Zhang, C. M. Sadler, and M. Martonosi, "Middleware for long-term deployment of delay-tolerant sensor networks categories and subject descriptors," in *Proceedings of the international workshop on Middleware for sensor networks*, 2006, pp. 13–18.

[135] E. Wohlstadter and S. Tai, "An aspect-oriented approach to bypassing middleware layers," in *Proceedings of the international workshop on Middleware for sensor networks*, 2007, pp. 25–35.

[136] M. Broy, I. H. Krüger, and M. Meisinger, "A formal model of services," *ACM Transactions on Software Engineering and Methodology*, vol. 16, no. 1, pp. 5–es, Feb. 2007.

[137] H. G. Goldman, "Building secure, resilient architectures for cyber mission assurance," in *Secure and Resilient Cyber Architectures Conference MITRE*, McLean, VA, 2010, pp. 1–18.

[138] C. J. Alberts and A. J. Dorofee, "Mission assurance analysis protocol (MAAP): assessing risk in complex environments," 2005. [Online]. Available: http://repository.cmu.edu/sei/431/

[139] C. Coopmans, "AggieNav: A small, well integrated navigation sensor system for small unmanned aerial vehicles," in *Proceedings of the ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference 2009, DETC2009*, vol. 2009, no. 49002. San Diego, CA, USA: ASME, Sep. 2009, pp. 635–640.

[140] C. Coopmans, H. Chao, and Y. Q. Chen, "Design and implementation of sensing and estimation software in AggieNav, a small UAV navigation platform," in *Proceedings of the ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference 2009, DETC2009*, vol. 2009, no. 49002. San Diego, CA, USA: ASME, 2009, pp. 649–654.

[141] "Gumstix, Inc." [Online]. Available: http://gumstix.com/

[142] D. Whalen, S. Rathinam, and C. Bagge, "Advanced developments in airport surface and terminal area traffic surveillance applications," in *Proceedings of the 22nd Digital Avionics Systems Conference*, vol. 2, 2003, pp. 9.B.3 –9.1–9 vol.2.

[143] H. Chao, C. Coopmans, L. Di, and Y. Chen, "A comparative evaluation of low-cost IMUs for unmanned autonomous systems," in *Proceedings of the 2010 IEEE Conference on Multisensor Fusion and Integration*, Sep. 2010, pp. 211–216.

[144] C. Coopmans, A. M. Jensen, and Y. Chen, "Fractional-order complementary filters for small unmanned aerial system navigation," in *Proceedings of the 2013 International Conference on Unmanned Aircraft Systems*, Atlanta, GA, 2013.

[145] "AuduIMU Open Source Project." [Online]. Available: https://code.google.com/p/ardu-imu/

[146] W. Premerlani and P. Bizard, "DCM Estimation," 2009. [Online]. Available: http://gentlenav.googlecode.com/files/DCMDraft2.pdf

[147] H. Chao, Y. Cao, and Y. Q. Chen, "Autopilots for small unmanned aerial vehicles: a survey," *International Journal of Control, Automation and Systems*, vol. 8, no. 1, pp. 36–44, 2010.

[148] H. Chao, A. M. Jensen, Y. Han, Y. Q. Chen, and M. McKee, "AggieAir: towards low-cost cooperative multispectral remote sensing using small unmanned aircraft systems," in *Advances in Geoscience and Remote Sensing*, G. Jedlovec, Ed. Vukovar, Croatia: IN-TECH, Oct. 2009, ch. AggieAir:, pp. 463–490.

[149] R. L. Greenspan, "Inertial navigation technology from 1970-1995," *Journal of The Institute of Navigation*, vol. 42, no. 1, pp. 165–185, 1995.

[150] J. L. Crassidis, J. L. Markley, and Y. Cheng, "Nonlinear attitude filtering methods," *AIAA Journal of Guidance, Control, and Dynamics*, vol. 30, no. 1, pp. 12–28, 2007.

[151] S.-G. Kim, J. L. Crassidis, Y. Cheng, and A. M. Fosbury, "Kalman filtering for relative spacecraft attitude and position estimation," *AIAA Journal of Guidance, Control, and Dynamics*, vol. 30, no. 1, pp. 133–143, 2007.

[152] R. Mahony, T. Hamel, and J.-M. Pflimlin, "Non-linear complementary filters on the special orthogonal group," *IEEE Transactions on Automatic Control*, vol. 53, no. 5, pp. 1203–1218, 2008.

[153] R. W. Beard, *State Estimation for Micro Air Vehicles*, ser. Studies in Computational Intelligence. Springer Berlin / Heidelberg, 2007, vol. 70, chap. 7, pp. 173–199.

[154] D. B. Kingston and A. W. Beard, "Real-time attitude and position estimation for small UAVs using low-cost sensors," in *Proceedings of the AIAA Unmanned Unlimited Technical Conference, Workshop and Exhibit*, 2004, pp. 2004–6488.

[155] J. S. Jang and D. Liccardo, "Small UAV automation using MEMs," *IEEE Aerospace and Electronic Systems Magazine*, vol. 22, no. 5, pp. 30–34, 2007.

[156] S. Bonnabel, P. Martin, and P. Rouchon, "Symmetry-preserving observers," *IEEE Transactions on Automatic Control*, vol. 53, no. 11, pp. 2514–2526, 2008.

[157] M. Jun, S. I. Roumeliotis, and G. S. Sukhatme, "State estimation of an autonomous helicopter using Kalman filtering," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 3, 1999, pp. 1346–1353.

[158] J. M. Roberts, P. I. Corke, and G. Buskey, "Low-cost flight control system for a small autonomous helicopter," in *Proceedings of the IEEE International Conference on Robotics and Automation*, vol. 1. Australian Robotics Automation Association, 2003, pp. 546–551.

[159] M. E. J. Newman, "Power laws, Pareto distributions and Zipfs law," *Contemporary physics*, vol. 46, no. 5, pp. 323–351, 2005.

[160] N. Gordon, J. Percival, and M. Robinson, "The Kalman-Levy filter and heavy-tailed models for tracking maneuvering targets," in *Proceedings of the International Conference on Information Fusion*, 2003, pp. 1024–1031.

[161] N. I. Krobka, "Differential methods of identifying gyro noise structure," *Gyroscopy and Navigation*, vol. 2, no. 3, pp. 126–137, 2011.

[162] H. Butler and C. de Hoon, "Fractional-order filters for active damping in a lithographic tool," *Control Engineering Practice*, vol. 21, no. 4, pp. 413–419, Apr. 2013.

[163] U. blox company, "u-Blox GPS protocol." [Online]. Available: http://www.astlab.hu/pdfs/protocol.pdf

[164] G. Welch and G. Bishop, "An introduction to the Kalman filter." [Online]. Available: http://www.cs.unc.edu/~welch/kalman/kalmanIntro.html

[165] Curtis L. Olson, "Microgear Project." [Online]. Available: http://sourceforge.net/projects/microgear/

[166] R. Beard, D. Kingston, M. Quigley, D. Snyder, R. Christiansen, W. Johnson, T. Mclain, and M. Goodrich, "Autonomous vehicle technologies for small fixed wing UAVs," *Journal of Aerospace Computing, Information, and Communication*, vol. 5, no. 1, pp. 92–108, 2005.

[167] R. Mahony, T. Hamel, and J.-M. Pflimlin, "Complementary filter design on the special orthoganal group SO(3)," in *Proceedings of the IEEE Conference on Decision and Control and European Control Conference*, 2005, pp. 1477–1484.

[168] Microstrain Inc., "Gx2 IMU specifications." [Online]. Available: http://microstrain.com/

[169] VectorNav Technologies, "VN-100 IMU specifications." [Online]. Available: http://www.vectornav.com

[170] Xsens Company, "Xsens-MTI-g IMU specifications." [Online]. Available: http://www.xsens.com

[171] Sparkfun Electronics, "Razor IMU specifications." [Online]. Available: http://www.sparkfun.com

[172] J.-K. Shiau, C.-X. Huang, and M.-Y. Chang, "Noise characteristics of MEMS gyro's null drift and temperature compensation," *Applied Science and Engineering*, vol. 15, no. 3, pp. 239–246, 2012.

[173] M. Bryson and S. Sukkarieh, "Vehicle model aided inertial navigation for a UAV using low-cost sensors," in *Proceedings of the Australasian Conference on Robotics and Automation*, 2004.

[174] J. L. Crassidis, L. F. Markley, and Y. Cheng, "Survey of nonlinear attitude estimation methods," *Journal of Guidance, Control, and Dynamics*, vol. 30, no. 1, pp. 12–28, 2007.

[175] H. Rehbinder and X. Hu, "Drift-free attitude estimation for accelerated rigid bodies," *Automatica*, vol. 40, no. 4, pp. 653–659, Apr. 2004.

[176] W. Higgins, "A comparison of complementary and Kalman filtering," *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-11, no. 3, pp. 321–325, May 1975.

[177] A.-J. Baerveldt and R. Klang, "A low-cost and low-weight attitude estimation system for an autonomous helicopter," in *Proceedings of the IEEE International Conference on Intelligent Engineering Systems*, 1997, pp. 391–395.

[178] T. Hamel and R. Mahony, "Attitude estimation on SO[3] based on direct inertial measurements," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2006, pp. 2170–2175.

[179] R. G. Brown and P. Y. C. Hwang, *Introduction to Random Signals and Applied Kalman Filtering*, 2nd ed., ser. Wiley, TK5102.5.B696. John Wiley & Sons, 1997, no. 4.

[180] W. H. Wirkler, "Aircraft course stabilizing means," pp. 23, Patent 2,548,278, 1951.

[181] W. Anderson and E. Fritze, "Instrument approach system steering computer," *Proceedings of the Institute of Radio Engineers*, vol. 41, no. 2, pp. 219–228, Feb. 1953.

[182] F. R. Shaw and K. Srinivasan, "Bandwidth enhancement of position measurements using measured acceleration," *Mechanical Systems and Signal Processing*, vol. 4, no. 1, pp. 23–38, 1990.

[183] M. Zimmermann and W. Sulzer, "High bandwidth orientation measurement and control based on complementary filtering," in *Proceedings of the SYROCO IFAC Symposium on Robot Control*, Vienna, Austria, Sep. 1991.

[184] E. R. Bachmann, I. Duman, U. Y. Usta, R. B. McGhee, X. P. Yun, and M. J. Zyda, "Orientation tracking for humans and robots using inertial sensors," in *Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automation*, 1999, pp. 187–194.

[185] E. R. Bachmann, R. B. McGhee, X. Yun, and M. J. Zyda, "Inertial and magnetic posture tracking for inserting humans into networked virtual environments," in *Proceedings of the ACM symposium on Virtual reality software and technology VRST*. ACM Press, 2001, p. 9.

[186] E. R. Bachmann, D. McKinney, R. B. McGhee, and M. J. Zyda, "Design and implementation of MARG sensors for 3-DOF orientation measurement of rigid bodies," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2003, pp. 1171–1178.

[187] S. Salcudean, "A globally convergent angular velocity observer for rigid body motion," *IEEE Transactions on Automatic Control*, vol. 36, no. 12, pp. 1493–1497, 1991.

[188] J.-M. Pflimlin, T. Hamel, and P. Souères, "Nonlinear attitude and gyroscope's bias estimation for a VTOL UAV," *International Journal of Systems Science*, vol. 38, no. 3, pp. 197–210, Jan. 2007.

[189] A. R. Plummer, "Optimal complementary filters and their application in motion measurement," *Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, vol. 220, no. 6, pp. 489–507, 2010.

[190] P. Oliveira, I. Kaminer, and A. Pascoal, "Navigation system design using time-varying complementary filters," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 36, no. 4, pp. 1099–1114, 2000.

[191] C. A. Monje, Y. Chen, B. Vinagre, D. Xue, and V. Feliu, *Fractional Order Systems and Control - Fundamentals and Applications*. Springer-Verlag, 2010.

[192] R. Mahony, T. Hamel, J. Trumpf, and C. Lageman, "Nonlinear attitude observers on SO(3) for complementary and compatible measurements: a theoretical study," in *Proceedings of the IEEE Conference on Decision and Control held jointly with the Chinese Control Conference*, 2009, pp. 6407–6412.

[193] G. Baldwin, R. Mahony, J. Trumpf, T. Hamel, and T. Cheviron, "Complementary filter design on the special Euclidean group SE(3)," in *Proceedings of the European Control Conference*, vol. 1, no. 3, 2007, pp. 3763–3770.

[194] I. Podlubny, *Fractional differential equations*. Academic press San Diego, 1999.

[195] K. B. Oldham and J. Spanier, *The Fractional Calculus*. Dover, 1974, vol. 17.

[196] I. Podlubny, I. Petráš, P. O'Leary, L. Dorčák, and B. M. Vinagre, "Analogue realizations of fractional order controllers," *Nonlinear dynamics*, vol. 29, no. 1, pp. 281–296, 2002.

[197] G. Bohannan, "Analog realization of a fractional control element-revisited," in *Proceedings of the 41st IEEE International Conference on Decision and Control, Tutorial Workshop*, vol. 1, no. 1, 2002, pp. 27–30.

[198] Y. Chen and K. L. Moore, "Discretization schemes for fractional order differentiators and integrators," *IEEE Trans. on Circuits and Systems I: Fundamental Theory and Applications*, vol. 49, no. 3, pp. 363–367, 2002.

[199] Y. Chen, B. M. Vinagre, and I. Podlubny, "Continued fraction expansion approaches to discretizing fractional order derivatives - an expository review," *Nonlinear Dynamics*, vol. 38, no. 1-4, pp. 155–170, Dec. 2004.

[200] B. T. Krishna, "Studies on fractional order differentiators and integrators: a survey," *Signal Processing*, vol. 91, pp. 386–426, 2011.

[201] A. Oustaloup, F. Levron, B. Mathieu, and F. M. Nanot, "Frequency-band complex noninteger differenciator: characterization and systhesis," *IEEE Transactions on Circuits and Systems, I: Fundamental Theory and Applications*, vol. 47, no. 1, pp. 25–39, 2000.

[202] H. Sheng, Y. Chen, and T. Qiu, *Fractional Processes and Fractional-Order Signal Processing: Techniques and Applications*. Springer, 2012.

[203] J. P. Nolan, *Stable Distributions - Models for Heavy Tailed Data*. Boston: Birkhauser, 2013.

[204] M. Veillette, "STBL: Alpha stable distributions for MATLAB." [Online]. Available: http://www.mathworks.com/matlabcentral/fileexchange/37514

[205] D. Valério, "ninteger." [Online]. Available: http://www.mathworks.com/matlabcentral/fileexchange/8312-ninteger

[206] Y. Luo, L. Di, J. Han, H. Chao, and Y. Chen, "VTOL UAV altitude flight control using fractional order controllers," in *4th IFAC Workshop on Fractional Differentiation and Its Application, Badajoz, Spain*, 2010.

[207] J. Kim, M.-S. Kang, and S. Park, "Accurate modeling and robust hovering control for a quad-rotor VTOL aircraft," in *Selected papers from the 2nd International Symposium on UAVs, Reno, NV, USA June 8-10, 2009*. Springer, 2010, pp. 9–26.

[208] I.-S. Kim, "The novel state of charge estimation method for lithium battery using sliding mode observer," *Journal of Power Sources*, vol. 163, no. 1, pp. 584–590, Dec. 2006.

[209] M. Podhradský, C. Coopmans, and A. M. Jensen, "Battery model-based thrust controller for a small, low cost multirotor unmanned aerial vehicles," in *Proceedings of the 2013 International Conference on Unmanned Aircraft Systems*, Atlanta, GA, 2013.

[210] A. M. Jensen, Y. Chen, M. McKee, T. Hardy, and S. L. Barfuss, "Aggieair - a low-cost autonomous multispectral remote sensing platform: new developments and applications," in *Proceedings of the 2009 IEEE International Geoscience and Remote Sensing Symposium*, vol. 4, Jul. 2009, pp. IV–995–IV–998.

[211] P. Pounds and R. Mahony, "Design principles of large quadrotors for practical applications," in *Proceedings of the 2009 IEEE International Conference on Robotics and Automation*, 2009, pp. 3265–3270.

[212] M. Chen and G. A. Rincon-Mora, "Accurate electrical battery model capable of predicting runtime and I-V performance," *IEEE Transactions on Energy Conversion*, vol. 21, no. 2, pp. 504–511, 2006.

[213] S. Mukhopadhyay and F. Zhang, "Adaptive detection of terminal voltage collapses for Li-Ion batteries," in *Proceedings of the 51st IEEE Conference on Decision and Control (CDC), 2012*, Maui, Hawaii, USA, Dec. 2012, pp. 4799–4804.

[214] V. Pop, H. Bergveld, D. Danilov, P. Regtien, and P. Notten, *Battery Management Systems: Universal State-of-Charge indication for battery-powered applications*, ser. Philips Research Book. Springer Netherlands, 2008.

[215] M. Podhradský, J. Bone, A. M. Jensen, and C. Coopmans, "Small low-cost unmanned aerial vehicle lithium-polymer battery monitoring system," in *Proceedings of the ASME International Design Engineering Technical Conferences & Computers and Information in Engineering Conference*, 2013.

[216] M. C. Knauff, C. J. Dafis, D. Niebur, H. G. Kwatny, and C. O. Nwankpa, "Simulink model for hybrid power system test-bed," in *IEEE Electric Ship Technologies Symposium*, 2007, pp. 421–427.

[217] G. Prasad, N. Sree Ramya, P. V. N. Prasad, and G. Tulasi Ram Das, "Modelling and simulation analysis of the brushless DC motor by using MATLAB," *International Journal of Innovative Technology and Exploring Engineering*, vol. 1, no. 5, pp. 27–31, Oct. 2012.

[218] C. Chéron, A. Dennis, V. Semerjyan, and Y. Chen, "A multifunctional HIL testbed for multirotor VTOL UAV actuator," in *Proceedings of the IEEE/ASME International Conference on Mechatronics and Embedded Systems and Applications (MESA), 2012*, Jul. 2010, pp. 44–48.

[219] Allegro Microsystems, "Allegro A4935 datasheet: automotive 3-phase MOSFET driver." [Online]. Available: http://www.allegromicro.com/

[220] S. Weiss, D. Scaramuzza, and R. Siegwart, "Monocular-SLAM-based navigation for autonomous micro helicopters in GPS-denied environments," *Journal of Field Robotics*, vol. 28, no. 6, pp. 854–874, 2011.

[221] J. Stowers, M. Hayes, and A. Bainbridge-Smith, "Altitude control of a quadrotor helicopter using depth map from Microsoft Kinect sensor," in *Proceedings of the 2011 IEEE International Conference on Mechatronics (ICM)*, Apr. 2011, pp. 358–362.

[222] D. Eynard, P. Vasseur, C. Demonceaux, and V. Fremont, "UAV altitude estimation by mixed stereoscopic vision," in *Proceedings of the 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2010, pp. 646–651.

[223] A. Cherian, J. Andersh, V. Morellas, N. Papanikolopoulos, and B. Mettler, "Autonomous altitude estimation of a UAV using a single onboard camera," in *Proceedings of the 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct. 2009, pp. 3900–3905.

[224] S. Bouabdallah and R. Siegwart, "Full control of a quadrotor," in *Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, Nov. 2007, pp. 153–158.

[225] P. Pounds, R. Mahony, and P. Corke, "System identification and control of an aerobot drive system," in *Proceedings of Information, Decision and Control*, Feb. 2007, pp. 154–159.

[226] K. Alexis, G. Nikolakopoulos, and A. Tzes, "Model predictive quadrotor control: attitude, altitude and position experimental studies," *IET Control Theory Applications*, vol. 6, no. 12, pp. 1812–1827, 2012.

[227] H. Bouadi, S. Simoes Cunha, A. Drouin, and F. Mora-Camino, "Adaptive sliding mode control for quadrotor attitude stabilization and altitude tracking," in *Proceedings of the 12th International Symposium on Computational Intelligence and Informatics (CINTI)*, Nov. 2011, pp. 449–455.

[228] B.-C. Min, J.-H. Hong, and E. T. Matson, "Adaptive robust control (ARC) for an altitude control of a quadrotor type UAV carrying an unknown payloads," in *Proceedings of the 11th International Conference on Control, Automation and Systems (ICCAS)*, Oct. 2011, pp. 1147–1151.

[229] Y. Hu and S. Yurkovich, "Battery state of charge estimation in automotive applications using LPV techniques," in *Proceedings of the American Control Conference (ACC)*, Jul. 2010, pp. 5043–5049.

[230] US Federal Aviation Administration, "Integration of civil unmanned aircraft systems (UAS) in the national airspace system (NAS) roadmap, first edition," US Federal Aviation Administration, Tech. Rep., 2013.

Vita

Calvin Coopmans

Utah State University AggieAir
4120 Old Main Hill
Logan, UT
84322-4120

Phone: +1 (435) 764-4579
Email: c.r.coopmans@ieee.org
Website: http://aggieair.usu.edu/coopmans/

Biography

Electrical, unmanned, and aerospace systems engineer. 5 years' experience in student satellite lab; 7 years' experience in unmanned aerial remote sensing lab. Currently serves as systems engineer and manages 25 student engineers on a comprehensive unmanned aerial remote sensing project.

Ph.D. Dissertation

"Cyber-Physical Systems Enabled by Unmanned Aerial System-Based Personal Remote Sensing: Data Mission Quality-Centric Design Architectures"
Focusing on architecture design of many scales as the key to integration of small, unmanned aerial remote sensing craft into civilian airspace and the challenges therein: safety, data quality, and ethics. Other research on unmanned aerial system design architectures, navigation estimation, and battery health-based multi-rotor VTOL craft control are also included.

Current Appointment

Head Researcher, Utah State University AggieAir

Appointments Held

2014-Current	Post-doctoral Researcher – AggieAir, Utah State University, Logan UT
2007-2014	Graduate Research Assistant – CSOIS/AggieAir, Utah State University, Logan UT
2007-2008	Robotics Engineer – Autonomous Solutions, Tremonton, UT
2002-2007	Electronics Engineer – SSEL, Montana State University Physics Dept., Bozeman MT
2006-2007	Electrical Engineer – Wavelength Electronics inc., Bozeman MT
2005-2006	Graduate Intern – Wavelength Electronics inc., Bozeman MT
2005-2006	DBA Engage Designs – Self-employed web/IT contractor, Bozeman MT
2004-2005	Embedded Systems Engineer – Quantum Design inc., Bozeman, MT
Summer 2004	Research Intern – Jet Propulsion Laboratory, Pasadena, CA

2001-2004	Student networking tech. – Montana State U. Information Technology Center, Bozeman, MT

Education

2014	Ph.D., Electrical Engineering at Utah State University
2010	M.Sc., Electrical Engineering at Utah State University
2004	B.Sc., Computer Engineering with minors: Computer Science, Mathematics at Montana State University

Writings and Publications

Ph.D. Dissertation

C. Coopmans, "Cyber-Physical Systems Enabled by Unmanned Aerial System-Based Personal Remote Sensing: Data Mission Quality-Centric Design Architectures," Ph.D Dissertation, Utah State University, 2014.

Master's Thesis

C. Coopmans, "Architecture, inertial navigation, and payload designs for low-cost Unmanned Aerial Vehicle-based Personal Remote Sensing," Master of Science (MS), Utah State University, 2010.

Book Chapters

C. Coopmans, B. Stark, A. M. Jensen, Y. Chen, and M. McKee, "Cyber-Physical Systems Enabled By Small Unmanned Aerial Vehicles," in *Handb. Unmanned Aer. Veh.*, K. P. Valavanis and G. J. Vachtsevanos, Eds. Salt Lake City, UT: Springer, 2014.

B. Stark, C. Coopmans, Y. Chen, A. M. Jensen, and M. Mckee, "Concept of Operations for Personal Remote Sensing Unmanned Aerial Systems," in *Handb. Unmanned Aer. Veh.*, 1st ed., K. P. Valavanis and G. J. Vachtsevanos, Eds. Salt Lake City, Utah, USA: Springer, Aug. 2014, vol. 69, no. 1-4, ch. TBD.

Journal Publications

C. Coopmans, A. M. Jensen, and Y. Chen, "Fractional-Order Complementary Filters for Small Unmanned Aerial System Navigation," *J. Intell. Robot. Syst.*, vol. 73, no. 1-4, pp. 429–, 2014.

H. Sheng, H. Sun, C. Coopmans, Y. Chen, and G. Bohannan, "A Physical Experimental Study of Variable-Order Fractional Integrator and Differentiator," *Eur. Phys. J. Spec. Top.*, vol. 193, no. 1, pp. 93–104, Apr. 2011.

M. Podhradský, C. Coopmans, and A. M. Jensen, "Battery State-Of-Charge Based Altitude Controller for Small, Low Cost Multirotor Unmanned Aerial Vehicles," *J. Intell. Robot. Syst.*, vol. 74, no. 1-2, pp. 193–207, 2014.

J. Han, L. Di, C. Coopmans, and Y. Chen, "Pitch Loop Control of a VTOL UAV Using Fractional Order Controller," *J. Intell. Robot. Syst.*, vol. 73, no. 1-4, pp. 187–195, 2014.

Conference Publications

C. Coopmans, "AggieNav: A small, well integrated navigation sensor system for small unmanned aerial vehicles," in *Proc. ASME Int. Des. Eng. Tech. Conf. Comput. Inf. Eng. Conf. 2009, DETC2009*, vol. 2009, no. 49002. San Diego, CA, USA: ASME, Sep. 2009, pp. 635–640.

C. Coopmans, "Architectures for Cyber-Physical Enabled Ethical, Accurate, and Resilient Unmanned Aerial Personal Remote Sensing," in *Proc. 2014 Int. Conf. Unmanned Aircr. Syst.*, Orlando, FL, 2014.

C. Coopmans, A. M. Jensen, and Y. Chen, "Fractional-Order Complementary Filters for Small Unmanned Aerial System Navigation," in *Proc. 2013 Int. Conf. Unmanned Aircr. Syst.*, Atlanta, GA, 2013.

M. Podhradsky, C. Coopmans, and A. M. Jensen, "Battery Model-Based Thrust Controller for a Small, Low Cost Multirotor Unmanned Aerial Vehicles," in *Proc. 2013 Int. Conf. Unmanned Aircr. Syst.*, Atlanta, GA, 2013.

N. Hoffer, C. Coopmans, and Y. Chen, "Small Low Cost Unmanned Aerial Vehicle System Identification: A Survey and Categorization," in *Proc. 2013 Int. Conf. Unmanned Aircr. Syst.*, Atlanta, GA, 2013.

J. Han, L. Di, C. Coopmans, and Y. Chen, "Fractional Order Controller for Pitch Loop Control of a VTOL UAV," in *Proc. 2013 Int. Conf. Unmanned Aircr. Syst.*, Atlanta, GA, 2013.

C. J. Hall, Daniel Morgan, A. M. Jensen, H. Chao, C. Coopmans, and M. Humpherys, "Team OSAM-UAV's Design for the 2008 AUVSI Student UAS Competition," in *Proc. ASME IDETC/CIE 2009, 1st Small Unmanned Aer. Veh. Technol. Appl. (SUAVTA), 2009 ASME/IEEE Int. Conf. Mechatron. Embed. Syst. Appl.*, San Diego, CA, USA, 2009.

H. Sheng, H. Sun, and C. Coopmans, "A Physical Experimental Study of Variable-Order Fractional Integrator and Differentiator," in *Proc. 4th IFAC Work. Fract. Differ. its Appl.*, vol. 2010, no. 1995, Badajoz, Spain, 2010.

C. Coopmans, L. Di, A. M. Jensen, A. A. Dennis, and Y. Chen, "Improved Architecture Designs for a Low Cost Personal Remote Sensing Platform: Flight Control and Safety," in *ASME/IEEE Int. Conf. Mechatron. Embed. Syst. Appl.*, Sep. 2011, pp. 937–943.

S. Mukhopadhyay, C. Coopmans, and Y. Chen, "Purely analog fractional order PID control using discrete fractional capacitors (Fractors): Synthesis and experiments," in *Proc. ASME IDETC/CIE 2009, 4th Symp. Fract. Deriv. Their Appl.*, San Diego, California, USA, 2009.

C. Coopmans and Y. Han, "Aggieair: an Integrated and Effective Small Multi-uav Command, Control and Data Collection Architecture," in *Proc. ASME Int. Des. Eng. Tech. Conf. Comput. Inf. Eng. Conf. 2009, DETC2009*, San Diego, CA, USA, 2009, pp. 1–7.

H. Chao, C. Coopmans, L. Di, and Y. Chen, "A Comparative Evaluation of Low-Cost IMUs for Unmanned Autonomous Systems," in *Proc. 2010 IEEE Conf. Multisens. Fusion Integr.*, Sep. 2010, pp. 211–216.

I. Petráš, Y. Chen, and C. Coopmans, "Fractional-order memristive systems," in *Proc. 14th IEEE Int. Conf. Emerg. Technol. Fact. Autom.*, Sep. 2009, pp. 1251–1258.

C. Coopmans, I. Petráš, Y. Chen, and I. Petras, "Analogue fractional-order generalized memristive devices," in *Proc. ASME IDETC/CIE 2009, 4th Symp. Fract. Deriv. Their Appl.*, San Diego, CA, USA, 2009.

C. Coopmans and Y. Q. Chen, "A General-Purpose Low-Cost Compact Spatial-Temporal Data Logger and Its Applications," in *Proc. 2008 IEEE AutoTestCon*, 2008.

C. Coopmans, B. Stark, and C. M. Coffin, "A Payload Verification and Management Framework for Small UAV-Based Personal Remote Sensing Systems," in *Proc. 2012 Int. Symp. Resilient Control Syst.* IEEE, Aug. 2012, pp. 184–189.

B. Stark, C. Coopmans, and Y. Chen, "A Framework for Analyzing Human Factors in Unmanned Aerial Systems," in *Proc. 2012 Int. Symp. Resilient Control Syst.*, Salt Lake City, Utah, USA, 2012, pp. 13–18.

C. Coopmans, H. Chao, and Y. Q. Chen, "Design and Implementation of Sensing and Estimation Software in AggieNav, a Small UAV Navigation Platform," in *Proc. ASME Int. Des. Eng. Tech. Conf. Comput. Inf. Eng. Conf. 2009, DETC2009*, vol. 2009, no. 49002. San Diego, CA, USA: ASME, 2009, pp. 649–654.

H. Sheng, H. Chao, C. Coopmans, J. Han, M. McKee, and Y. Chen, "Low-cost UAV-based thermal infrared remote sensing: Platform, calibration and applications," in *Proc. IEEE/ASME Int Mechatronics Embed. Syst. Appl. Conf*, Qingdao, ShanDong, China, 2010, pp. 38–43.

Poster Presentations

C. Coopmans, B. Stark, A. M. Jensen, Y. Chen, and M. Mckee, "Cyber-Physical Systems Enabled by Small Unmanned Aerial Vehicle-Based Personal Remote Sensing," in *2011 Symp. Emerg. Top. Control Model. Cyber-Physical Syst.*, Urbana-Champaign, IL, 2011.

C. Coopmans, A. M. Jensen, and M. McKee, "When is a drone not a drone? Performing Meaningful Scientific Work with the AggieAir Unmanned Aerial System," in *Kansas State Univ. Unmanned Syst. Conf.*, Poster Session, Manhattan, Kansas, 2013.

Tutorials Offered

C. Coopmans, Y. Chen, N. V. Hoffer, and A. M. Jensen, "Remote Sensing of Actionable Scientific Information using Unmanned Aerial Systems," in *2014 Int. Conf. Unmanned Aer. Syst.*, Invited Tutorial (Half Day), Orlando, FL., 2014.

C. Coopmans, "AggieAir Remote Sensing and Airworthy Multi-scale UAS Architectures," in *Drone Aer. Robot. Conf.*, Invited talk, Embedding Values in Design Session, New York, USA, 2013.

C. Coopmans, Y. Chen, B. Stark, and A. M. Jensen, "Tutorials on Airworthiness Enhancing Architectures and Human Factors for Small Unmanned Aerial Systems," in *2013 Int. Conf. Unmanned Aer. Syst.*, Invited Tutorial (Half Day), Atlanta, GA., 2013.

Y. Chen, C. Coopmans, B. Stark, and A. M. Jensen, "Low-cost UAV-based precision thermal infrared (TIR) mapping - A new Personal Remote Sensing capability: UAV platform, TIR payload, in-flight calibration and applications," in *2012 Int. Conf. Unmanned Aer. Syst.*, Invited Tutorial (Full Day), Philadelphia, PA., 2012.