

CRANFIELD UNIVERSITY

Jonathan Gerald Pelham

**Artificial Immune Systems for
Case Based Reasoning of
Unmanned Aircraft Flight Data**

School of Aerospace, Transport, & Manufacturing
PhD in Manufacturing

PhD
Academic Year: 2012 - 2016

Supervisor: Dr. Ip-Shing Fan, Jim McFeat, & Prof. Ian Jennions
September 2017

CRANFIELD UNIVERSITY

School of Aerospace, Transport, & Manufacturing

PhD in Manufacturing

PhD

Academic Year: 2012 - 2016

Jonathan Gerald Pelham

**Artificial Immune Systems for Case
Based Reasoning of Unmanned Aircraft
Flight Data**

Supervisor: Dr. Ip-Shing Fan, Jim McFeat, & Prof. Ian Jennions

September 2017

This thesis is submitted in full fulfilment of the requirements for
the degree of Doctor of Philosophy

© Cranfield University, 2017. All rights reserved. No part of this
publication may be reproduced without the written permission of
the copyright owner.

Law Number XIV: After the year 2015, there will be no airplane crashes. There will be no takeoffs either, because electronics will occupy 100 percent of every airplane's weight. (Norman Augustine - CEO Lockheed Martin 1986[[1](#)])

Abstract

UAS(Unmanned Aerial Systems) mishaps are high, and their pilots face many control challenges. The reliability of UAS has been seen as a dominant mishap cause but in several instances the aircraft could have been saved if the health state of the aircraft had been understood at an earlier point by the pilot. Manned and unmanned aircraft pilots both benefit from the use of their own experience in the detection and mitigation of faults during flight. However it has been suggested that pilots within a GCS(Ground Control Station) face difficulties in maintaining their situational awareness due to the nature of remote control. The use of a cognitive framework as a basis for case based reasoning is suggested as a way to integrate through life learning into the Safety Management System. The population of the case base for such a system would require a large investment of time to create. The use of machine learning is suggested and evaluated to address this issue by generating cases for CBR. This has seen some success and even the use of an AIS(Artificial Immune System) in this thesis. An AIS was used in order to try to address the problem of cost and time caused by high pre-processing required by common machine learning methods. A simulation of the Aerosonde UAS was created and multiple flights simulated to build up a set of representative set flight data. Several fault cases were included in the simulated flights of varying severities. Different machine learning schemes were evaluated using the data set and their effectiveness compared in order to evaluate the ability of the algorithm to learn from flight data without extensive pre-processing. The complex dataset made the problem difficult but in analysis the AIS performed slightly better than the neural network with which it was compared. In due time and with development it's computational cost could be reduced and its effectiveness increased. The benefit of an automated method to learn from aircraft incidents and mishaps can readily be seen in a fleet scenario where it would be uneconomical to analyse flight data of unmanned aircraft in the same way that it would be done for manned aircraft.

This semi-supervised approach reduces personnel requirements and enhances the ability of operators to learn from mishaps by relating mishap cases to the current situation and being transparent in their alerting criteria.

Acknowledgements

This research was kindly sponsored by the BAE Systems and EPSRC grant 1135839

Thanks go to my supervisors Ip-Shing Fan, Ian Jennions, & Jim McFeat

Thanks go to Dr James Whidborne for the use of the Aerosonde Matlab model.

Also to the support of my wonderful wife Rasa

Contents

Abstract	v
Acknowledgements	vii
List of Figures	xv
List of Tables	xvii
List of Algorithms	xix
Abbreviations	xx
1 Introduction - Problem	1
1.1 Introduction	4
1.2 Gap Analysis	4
1.2.1 Knowledge Gap	4
1.3 Direction	5
1.4 Research Approach	5
1.5 Thesis Logic	9
1.6 Summary	11
2 Industrial Background	13
2.1 Introduction	13
2.2 UAS	14
2.2.1 The Aircraft	15
2.2.2 The Ground Control Station	17
2.2.3 The communication backbone	17
2.2.4 The Human element	19
2.2.5 Supporting Systems	19
2.3 Current Usage	20
2.3.1 Mishap	21
2.3.2 Flight Analysis	22
2.3.2.1 Flight Data	22
2.3.2.2 FOQA/FDM	25
2.3.2.3 Incident Analysis	27

2.4	Integrated Vehicle Health Management	28
2.4.1	Introduction	28
2.4.2	History	29
2.4.3	Philosophy	31
2.4.4	Architecture	32
2.4.5	IVHM System Design	33
2.4.5.1	Introduction	33
2.4.5.2	System elements	33
2.4.6	System Requirements	37
2.4.6.1	Non-functional Requirements	37
2.4.6.2	Functional Requirements	38
2.4.7	IVHM for UAS	39
2.4.7.1	IVHM requirement changes with mode	39
2.5	Cognitive Control	42
2.6	The cognitive gap	45
2.7	CAHM (Cognitive Asset Health Management)	47
2.7.1	Features of a more cognitive Asset Health Management system	51
2.7.2	General Requirements	54
2.7.3	Framework of data types	55
2.8	Summary	56
3	Literature Review - Applicable Machine Learning	57
3.1	Introduction	57
3.2	Case Based Reasoning	57
3.3	Clustering and Classification	61
3.3.1	Clustering	61
3.3.2	Classification	62
3.3.2.1	Supervised	63
3.3.2.2	Un-supervised	64
3.4	UAS related Machine Learning State of the Art	65
3.4.1	Machine Learning for adaptive control of UAS	65
3.4.2	Fault diagnosis and prognosis	65
3.4.2.1	Diagnosis	66
3.4.2.2	Prognosis	67
3.4.3	Machine Learning for health management of UAS	67
3.5	Summary	68
4	Machine Learning Method Comparison - AIS	69
4.1	Introduction	69
4.2	AIS - Artificial Immune System	69
4.2.1	Human Immune System	72
4.2.1.1	Key concepts	72
4.2.2	AIS Development	76
4.2.3	AIS Discussion	82

4.2.3.1	Comparison to cognitive control	82
4.2.3.2	Advantages	82
4.2.3.3	Disadvantages	84
4.2.4	AIS vs classification	85
4.3	AIS Case Based Reasoning	86
4.4	The AIS (artificial immune system) as a candidate method for cognitive AHM	89
4.5	An AIS for through life health management of UAS	90
4.5.1	Representation	91
4.5.2	Affinity measures	91
4.5.2.1	Real valued representation affinity metrics	92
4.5.2.2	Boolean matching affinity metrics	92
4.5.3	Maturation process	93
4.5.4	Danger Theory	94
4.6	Comparison of CAHM, CBR, and AIS-CBR characteristics	95
4.7	Summary	96
5	Experiment Design	97
5.1	Introduction	97
5.2	Aircraft Flight Data	98
5.2.1	Flight Simulation	98
5.3	Aerosonde	102
5.4	Mishap Design	104
5.4.1	Permanent Fault Scenarios	104
5.4.1.1	Fuel Leak	104
5.4.1.2	Control Jam	105
5.5	Mission Design	107
5.6	Pre-Processing	108
5.7	Machine Learning Evaluation	110
5.8	CBR Analysis and Application	110
5.8.1	Mishap4840	112
5.8.2	Mishap4894	115
5.8.3	Mishap4992	120
5.8.4	Mishap5080	123
5.9	Dataset Creation	128
5.10	Summary	129
6	Algorithm Design	131
6.1	Introduction	131
6.2	AIS Design elements	132
6.2.1	Application Domain	133
6.2.2	Immune Entities	133
6.2.3	Representation	134
6.2.4	Affinity Measures	135

6.2.4.1	Real Vector Representation	135
6.2.4.2	Affinity Metrics	136
6.2.4.3	Recognition Threshold	137
6.2.5	Immune Algorithms	140
6.2.5.1	Algorithms Elements	140
6.3	Population Type AIS (v1 Algorithm)	143
6.4	Race Type AIS (v2 Algorithm)	144
6.4.1	AIS-FOQA v2 computation complexity	148
6.5	Design Validation using the Fisher Iris Dataset	149
6.5.1	New data, new antibodies	149
6.6	Neural Network	152
6.6.1	N-S-FBEC computational complexity	153
6.7	Fisher Iris AIS Performance	153
6.7.1	AIS Fisher Iris Results	153
6.7.2	NN-S-FBEC Fisher Iris Results	154
6.7.3	Fisher Iris Comparison	155
6.8	Summary	155
7	Results	157
7.1	Introduction	157
7.2	NN-S-FBEC Flight Data Results	158
7.3	AIS Flight Data Results	161
7.4	Usage Comparison	169
7.4.1	CBR in Application	169
7.4.2	NN-S-FBEC in Application	170
7.4.3	AIS-FOQA in Application	170
7.5	Summary	170
8	Discussion, Conclusion, and Future Work	173
8.1	Discussion	173
8.1.1	Research Question	174
8.1.2	Challenges and Opportunities	174
8.1.3	Realistic dataset	175
8.1.4	Learning from Mishaps	175
8.1.5	The New Method	176
8.1.5.1	Comparison of AIS-FOQA and NN on Fisher Iris Data	176
8.1.5.2	Comparison of AIS-FOQA and NN on Flight Data	177
8.1.6	AIS-FOQA Application discussion	179
8.2	Conclusions	181
8.2.1	Key Findings and Summary	181
8.3	Future Work	183

A	AIS Race Type Algorithm	209
A.1	Inputs	209
A.1.1	Mandatory Inputs	209
A.1.2	Optional Inputs	210
A.2	Outputs	210
A.2.1	Output before completion	210
A.2.2	Output at completion	211
A.3	Code	211
B	AIS Population Type Algorithm	235
B.1	Inputs	235
B.1.1	Mandatory Inputs	235
B.2	Outputs	236
B.2.1	Output at completion	236
B.3	Code	236
C	Persistent UAS Mishap Analysis	241
C.1	Persistent UAS	241
C.1.1	Common Features	242
C.1.2	Feature related issues	242
C.1.3	MQ-1/RQ-1 Predator	243
C.1.4	RQ-4 Global Hawk	244
C.1.5	MQ-9 Predator B/Reaper	244
C.2	UAS Mishap Papers	245
C.2.1	Hundley et al	245
C.3	Data Sources	246
C.3.1	Safety Investigation Board Reports	246
C.3.2	Accident Investigation Board	246
C.3.3	Mishap Classification	247
C.4	Aircraft Featured in Study	247
C.4.1	MQ-1/RQ-1 Predator	247
C.4.2	RQ-4 Global Hawk	247
C.4.3	MQ-9 Predator B/Reaper	248
C.5	Study Discussion	252
C.5.1	Mishap Categorisation	252
C.5.2	Mode vs. Cause	253
C.6	Discussion	255

List of Figures

1.1	Research Approaches	6
1.2	Demon UAS	8
1.3	Thesis Logic	10
2.1	UAS System Architecture	14
2.2	PACT Framework	16
2.3	USAF Class A mishap rate trends(Various aircraft)	21
2.4	V-N Diagram	25
2.5	FOQA Risk Analysis cycle	25
2.6	IVHM System Architecture	32
2.7	IVHM System Operational Cycle	34
2.8	Fitts List	42
2.9	UAS Decision Timescales	43
2.10	OODA Loop	46
2.11	Cognitive Task Hierachy	49
3.1	Case Based Reasoning	58
3.2	CBR-FOQA within Safety Management System	60
3.3	Voronoi Diagram	62
3.4	Single Neuron	63
4.1	Maintenance Transformation Map	72
4.2	B-Cells and Bacteria	74
4.3	AIS Development	78
4.4	Immune System Algorithm	81
4.5	AIS vs Supervised Machine Learning	83
5.1	Aircraft Orientation Conventions	101
5.2	Aerosonde UAS	103
5.3	Aerosonde Simulink Overview	103
5.4	Aerosonde Ruddervator Fault Block	107
5.5	Aerosonde Simulation Flowchart	108
5.6	Sim4840 Flight Data 1 of 2	113
5.7	Sim4840 Flight Data 2 of 2	114
5.8	Sim4894 Flight Data 1 of 2	116
5.9	Sim4894 Flight Data 2 of 2	117
5.10	Sim4894 Flight Data 3 of 4	118

5.11	Sim4894 Flight Data 4 of 4	119
5.12	Sim4992 Flight Data 1 of 2	121
5.13	Sim4992 Flight Data 2 of 2	122
5.14	Sim5080 Flight Data 1 of 4	124
5.15	Sim5080 Flight Data 2 of 4	125
5.16	Sim5080 Flight Data 3 of 4	126
5.17	Sim5080 Flight Data 4 of 4	127
6.1	AIS Design Method	132
6.2	Antibody responses	138
6.3	Real Valued Negative Selection Algorithm	141
6.4	AIS Mutation Methods	142
6.5	Race Type AIS Algorithm structure	146
6.6	Fisher Iris Early Stage	150
6.7	Fisher Iris Late Stage	151
6.8	Fisher Iris Antibody Visualisation	152
7.1	Sim4840 NN Response	160
7.2	Sim4840 NN Response Result	161
7.3	Pairwise Plot 1 of 7	163
7.4	Pairwise Plot 2 of 7	163
7.5	Pairwise Plot 3 of 7	164
7.6	Pairwise Plot 4 of 7	164
7.7	Pairwise Plot 5 of 7	165
7.8	Pairwise Plot 6 of 7	165
7.9	Pairwise Plot 7 of 7	166
7.10	AIS Algorithm learning progress figure	166
7.11	Sim4840 AIS Response	168
7.12	Sim4840 AIS Response Result	169
C.1	DOD Roadmap RPAS Mishap Statistics	251

List of Tables

4.1	Application areas of AIS	71
5.1	Flight data Minimums and Maximums	109
5.2	Data Summary	129
6.1	AIS-FOQAv2 run variable values	147
6.2	AIS-FOQAv2 Settings	148
6.3	AIS-FOQA Fisher Iris Results	154
6.4	AIS-FOQA Fisher Iris Performance	154
6.5	NN Fisher Iris Results	154
6.6	NN Fisher Iris Performance	155
7.1	Mishap Details	158
7.2	NN-S-FBEC Results	159
7.3	AIS Algorithm learning progress table	162
7.4	AIS-FOQA Results	167
8.1	Table 1 from Ji 2004[2]	177
8.2	Results Comparison	178
C.1	USAF UAS Class A Mishaps	249
C.2	USAF Mishap Statistics	252

List of Algorithms

1	Dendritic Cell Algorithm	79
2	CLONALG AIS Algorithm	80
3	Equation for Scaling into Unit Interval	109
4	Equation for Minkowski Distance	135
5	Equation for calculating volume of hypersphere	135
6	Equation for Antibody Recognition affinity metric	139
7	Equation for random distribution of points on the surface of an n dimensional sphere	141
8	AIS-FOQA Population Type (v1 Algorithm)	144
9	AIS-FOQA Race Type (v2 Algorithm)	145
10	Mishap Probability per Mission	251

Abbreviations

AAIB	A ir A ccident I nvigation B ranch
ACARS	A ircraft C ommunications A ddressing and R eporting S ystem
ACM	A utomated C ontingency M anagement
ACMS	A ircraft C ondition M onitoring S ystem
AFRL	A ir F orce R esearch L aboratory
AHM	A sset H ealth M anagement
AI	A rtificial I ntelligence
AIB	A ccident I nvigation B oard
AIS-FOQA	A rtificial I mmune S ystem for F light O perations Q uality A ssurance
ALIS	A utonomic L ogistics I nformation S ystem
AO	A rea of O perations
AOA	A ngle O f A ttack
ARINC	A eronautical R adio I ncorporated (Standards Organisation)
ATEX	Appareils destinés à être utilisés en A Tmosphères E Xplosibles
AUROC	A rea U nder R eceiver O perating characteristic C urve
BIT	B uilt in T est
BITE	B uilt in T est E quipment
BVLOS	B eyond V isual L ine of S ight
C3	C ommunications, C ommand, & C ontrol
CAS	C rew A lerting S ystem
CAHM	C ognitive A sset H ealth M anagement
CBM	C ondition b ased m aintenance
CBR	C ase B ased R easoning
CMC	C entral M aintenance C omputer
CMS	C ontingency M anagement S ystem
CND	C an N ot D uplicate
CPU	C entral P rocessing U nit
DERA	D efence E valuation and R esearch A gency
DTC	D amage T olerant C ontrol
DFDAU	D igital F light D ata A cquisition U nit

EICAS	Engine Indicating and Crew Alerting system
ESC	Electronic Speed Controller
EVLOS	Extended Visual Line of Sight
FBW	Fly by Wire
FDHMI	Flight Deck Health Monitoring Indication
FDR	Flight Data Recorder
FMECA	Failure Modes, Effects, & Criticality Analysis
FMS	Flight Management System
FOQA	Flight Operations Quality Assurance
GWG	Geospatial intelligence standards Working Group
HHMAG	Helicopter Health Monitoring Advisory Group
HMI	Human Machine Interaction
HUMS	Health and Usage Management/Monitoring System
ICAO	International Civil Aviation Organisation
IP	Ingress Protection (specification methodology)
ISHM	Integrated System Health Management
IVHM	Integrated Vehicle Health Management
LMA	Large Model Association
LOS	Line of sight
LRU	Line Replaceable Unit
MEMs	Micro Electronic Mechanical Systems
MIMOSA	Maintenance Information & Management Open Systems Alliance
ML	Machine Learning
MTBF	Mean time Between Failures
MTTR	Mean Time to Repair
NDE	Non Destructive Evaluation
NDT	Non Destructive Testing
NLOS	Non Line of Sight
NN-S-FBEC	Neural Network - Sigmoid - Feedback Error Correction
OEM	Original Equipment Manufacturer
OSA-CBM	Open Software Architecture - Condition Based Maintenance
OSA-EAI	Open Software Architecture - Enterprise Application Integration
PACT	Pilot Authority and Control of Tasks
PAMP	Pathogen-Associated Molecular Pattern
PDF	Probability Density Function
PHM	Prognostic Health Management
PIO	Pilot Induced Oscillation
PWM	Pulse Width Modulation
QAR	Quick Access Recorder

ROC	R eceiver O perating C haracteristic
RPA	R emotely P iloted A ircraft
RPAS	R emotely P iloted A erial(or Aircraft) S ystem
RTOK	R eturned T ested O K
SA	S ituational A wareness
SIB	S afety I nvestigation B oard
SIMICA	S oftware I nterface for M aintenance I nformation C ollection and A nalysis
S/N	S erial N umber
TAT	T urn A round T ime (Typical)
T/N	T ail N umber
UA	U nmanned A ircraft
UAS	U nmanned A ircraft S ystem
UAV	U nmanned A erial V ehicle
VLOS	V isual L ine of S ight
VHM	V ibration H ealth M onitoring
VMSC	V ehicle M anagement S ystem C omputer
VPP	V ariable P itch P ropellor

*Dedicated to my wife Rasa, and to my sons Lukas and
Tomas.*

Also to the memory of Elizabeth Goodeve Foster

Chapter 1

Introduction - Problem

Introduction

The use of unmanned aircraft for aerial work is a growing market within both the military and civilian sectors. The safety of these aircraft and the problems associated with remote operation are a barrier to wider adoption and the ability of operators to access of new markets and offer novel services.

Research Problem

The use of unmanned aircraft for various tasks has become a growth industry and one in which the dominant mode is direct remote control of the aircraft. In order to encourage further use and to lower the operational cost certain challenges must be overcome. See Section 2.2 for further information. One of these challenges is the health management of multiple aircraft where the supervisory attention of the operation may be split between multiple tasks, or even in future between multiple aircraft. Current health management methods(Section 2.4 and 3.4) are insufficient for this challenge as they either require a large investment of personnel(Section 2.3.2) or they struggle to integrate new knowledge gained through the life of the aircraft(Section 3.4.3).

Research Aims and Objectives

Research Aim

Due to the large and increasing demand for UAS and the very large and growing volumes of flight data expected to be available to any competent operator methods are needed to be able to make use of this asset to help improve operations and safety without requiring investment out of proportion to the possible value of the improvements as until improvements are realised their value difficult to quantify and so the investment a risk. This project aims to create a machine learning approach to learn from the operation of unmanned aircraft, in order to improve the ability of the operator to manage their systems during flight, with particular attention to mitigating the risk of incident and mishap. This method to be able to offer benefit without creating an unreasonable increase in workload.

Research Objectives

Evaluate approaches to enhancing the through life health management of unmanned aircraft, their supporting systems and to examine computing concepts identified as highly relevant machine learning paradigms to the health management field.

1. Survey the challenges and opportunities presented to the remote pilot by the health management of an unmanned aerial system
2. Using realistic aircraft data to create representative aircraft flight characteristics for evaluation of health management learning algorithms
3. Use real UAS mishap reports in the creation of fault scenarios to give realistic failure data
4. Develop a method to enable health management and through life learning for future large fleets of unmanned aircraft
5. Discuss the application of this new method to the management of unmanned aircraft fleets

Research Approach

In brief it was decided to construct a representative dataset of unmanned aircraft flight data including some reasonable mishaps. This could then be used to evaluate machine learning algorithms and their ability to assist in the health management of unmanned aircraft. The different research approaches that were considered are discussed further in Section 1.4. The experiment was structured to try to ensure that the data was a representative of real use as possible, that the tests were fair and repeatable, and that any complicating factors caused by the practicalities of UAS operation were considered. The design of the experiment is discussed further in Chapter 5.

Research Contributions

A novel AI algorithm was created to process aircraft flight data and create methods to recognise mishaps in a way to assist in future diagnostics and the capturing of aircraft failure modes and responses that could be used within a fleet management perspective.

1. Use of a priori knowledge to generate AI detectors(See Chapter 6)
2. Semi-Supervised learning of an AI through flight data analytics(See Chapter 6)
3. Putting IVHM within a cognitive framework and exploring a method which moves IVHM more towards cognitive automation. Knowledge based tasks are still performed by humans. The AI and its supporting elements support this task through new detector suggestion in response to their identification of non-normal situations.(See Section 4.4)
4. Use of an AI method to aid continual learning for operators and maintainers through use of flight data.(See Chapter 6)
5. Use of mission simulator to evaluate data requirements of a system and enable future work and comparison of IVHM diagnostic and prognostic methods for UAS(See Chapter 5)

6. A well discussed method for conducting integrating health management and FOQA into the UAS lifecycle. (See Chapter 8.2)

Summary

The research has been introduced in brief as the investigation through simulation of methods to detect and capture fault information in UAS. Particular attention being given to ways which enhance the ability of the human operators to understand why any future incident gets classified as a fault.

1.1 Introduction

The gap chosen in section 1.2 can be explored in a number of ways and in Section 1.4 the most appropriate way will be identified in order to yield original knowledge within this area.

1.2 Gap Analysis

As can be seen from the various aspects of UAS health management discussed in Section 2.4 a method is needed to bring together health management for UAS. Some machine learning approaches to elements of this are discussed in section 3.4 but no whole system approaches have been suggested other than those which require a significant long term cognitive burden on the human elements of the system. This is expanded and discussed in Section 2.7

1.2.1 Knowledge Gap

One of the underlying theories used in evaluating possible solutions during this investigation is that there is a restriction within UAS on the amount of health management systems that can be on the aircraft. This is examined more widely along with the requirements for any algorithm. However part of the gap is understanding exactly how much should be on the aircraft and how much should be

at work on the ground. During the system design and test phase fault trees can be predicted and case based reasoning constructed to help with automated fault detection and identification during use. However this is a time consuming exercise and is not done for some small UAS as they are time consuming to create for the full breadth of possible faults that could occur. It also is weak to unanticipated emergent faults that only become apparent through regular operation of the aircraft. UAS do however have more sensors on-board than an equivalent manned aircraft. This allows the use of aircraft sensors to detect faults. There is a wider question regarding how such a “dual use” system could be configured and at what point the control data ceases to be useful for determining the health of systems. A human pilot suspecting an issue with an UAS would institute diagnostic procedures which would use on-board systems and processing as well as flight telemetry data and off-board processing to come to their conclusions.

1.3 Direction

It was decided to pursue an evaluation of a novel machine learning concept for the through life learning of mishap cases. Due to the scarcity of flight data from UAS and the specifically the lack of opportunity to replicate flights of the existing research UAS Demon or to conduct ground tests it was decided to construct an evaluation based around representative flight data which could be generated in simulation. This is discussed in Section 5.

1.4 Research Approach

The research approach as illustrated in Figure 1.1 had a number of potential ways to explore how IVHM could be used to address the problem of through life health management in UAS. The two central elements of any UAS are the assets themselves which consist of aircraft, controls, maintenance equipment, data systems, etc and are described in Section 2.2; and the operation performing means which consist of the operators(described in Section 2.2 and 2.5) in both the individual and corporate sense and the other organisation actors that use and choose the assets.

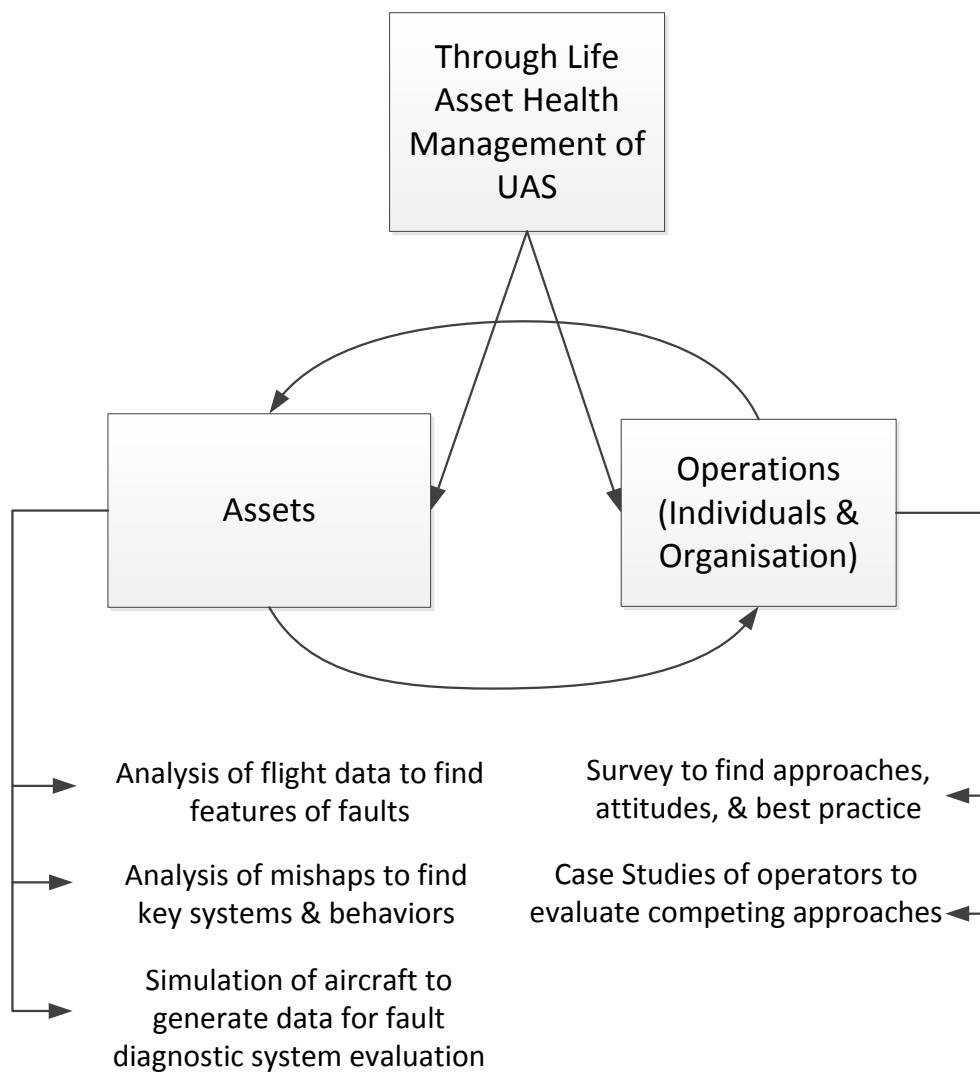


FIGURE 1.1: Research Approaches

Operations

Operator Surveys

The investigation of attitudes and assumptions in the field of UAS has been examined elsewhere.^[3] Some initial work was done to discuss with operators methods and mishaps in an interview setting to establish what sorts of mishaps were the most important to examine. This approach is susceptible to skewing if the population of individuals taking part is too small. It can also be difficult to structure

the questions of a survey in a way that avoids leading the respondent to a specific answer. It can however show the perceptions in the community and act as a pointer to areas suitable for future research. It can also be very useful for understanding attitudes and practices within a domain as long as the questions are well constructed and easy for the respondents to parse and answer appropriately.

Operator Case Studies

The use of case studies to compare and contrast different operator approaches and try to determine best practice is a worthwhile pursuit but requires cooperative operators willing to be open about their practices. Some such as the LMA (Large Model Association)[4] have been willing to discuss how they assure safety but do not form a good basis for comparison with commercial operators, and commercial operators are frequently reticent to disclose potentially competitive insight to any rivals. The small population size result skew associated with surveys also applies here and can have an even greater effect due to the associated lower number cases.

Assets

Analysis of Flight Data

The analysis of UA flight data to identify lessons and suitable methods starts with the search for that data. A ready source was available with the Demon UA which is shown in Figure 1.2. This was developed by Cranfield University[5] as part of the FLAVIR project[6]. However there was not enough data to generate significant contribution and the likelihood of doing experiments to gain more is low. The data that could be generated would also be biased towards data generated from flight test activity rather than practical use and faults could only be partially simulated without risking the aircraft. See Section 2.3.2 and 2.3.2.1 for further information.

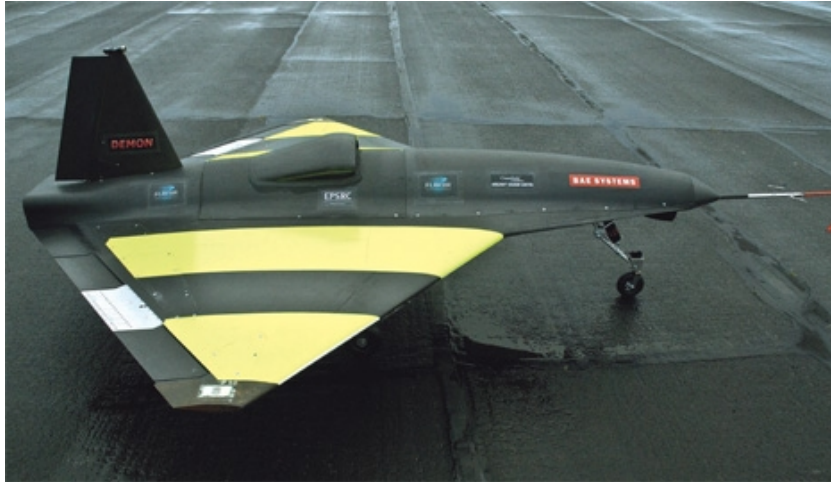


FIGURE 1.2: Demon unmanned aircraft[6]

Analysis of Mishaps

The analysis of incidents and mishaps has been a fruitful field for understanding how to better operate and design aircraft. There are some sources of data but none at present provides enough information to perform third party analysis of the type required for detailed operation and design conclusions. Appendix C looks at the data that is available and discusses some conclusions but beyond that significant contribution seemed to require a different approach. See Section 2.3.1 and Appendix C for further information. However the original data used to generate the conclusions for the various investigatory organisations is not available for analysis by third parties. Where reports have been published they are often also partially redacted to disguise operational issues and avoid assigning blame to named parties. These issues can sometimes make it difficult to translate conclusions within one mishap classification system such as that of the USAF into a system used for research. One example is their approach to human error. The USAF system does not assign conclusions of human error but instead strives to identify training issues or human machine interface challenges. This is in contrast to other approaches which are more comfortable listing mishap cause as aircrew.

Fault Simulation

Simulation has been used to explore aspects of machine behaviour too time consuming or too dangerous to explore in a practical manner. Unmanned aircraft have frequently been explored in this way for many research areas from formation flying[7], through to aircraft faults[8], aircraft flight envelope prediction[9], and

simulation is also used to evaluate flight crew[10]. Within fault simulation there are several ways the faults could have been simulated at varying levels of detail. The need to generate large amounts of flight data to create a solution in the CBR-FOQA space as shown in Figure 3.2 drove the choice to simpler fault scenarios that could be represented and simulated efficiently with an acceptable level of detail rather than taking a long time to generate only a few highly detailed fault scenarios. Fault simulation is susceptible to mistakes in the aircraft simulation, the fault simulation, and the mission simulation and so to ensure validity existing models already in use for research were investigated. Aircraft fault simulation is further discussed in Chapter 5 and specifically in Section 5.2.

1.5 Thesis Logic

The flowchart shown in Figure 1.3 shows the research plan of this thesis. In the initial stages separate literature reviews were performed to improve understanding of both IVHM and health management in general as well as of unmanned aircraft and their particular challenges. A review was then performed of available mishap data for UAS and existing studies done in this area to assess the vulnerability of UAS to mishaps compared to manned aircraft. Analysis of mishap led to the need for work in flight data analysis to review how UAS flight data could be reviewed and studied. In parallel the need for a strong business case for health management before equipment could be allotted to an aircraft design was identified and led to a strategy of evaluating what approaches could enhance the health management of the UAS without adding equipment to the aircraft if possible. This works very well with the flight data analysis work as the aircraft flight data recorder is a prime source of information for analysis and health management and with the telemetry stream to the GCS common in UAS offered the opportunity to have reasoning offboard the aircraft or in certain circumstances to deploy reasoners to the aircraft itself when a business benefit can be demonstrated. In order to study this concept different approaches were evaluated until the simulation of aircraft with faults and the construction of an artificial immune system was chosen as the approach to explore this area. The design of the simulation was informed by the work into understanding UAS and the review of real UAS mishaps from the USAF. The results of the AIS-FOQA algorithm were then evaluated compared to a neural network.

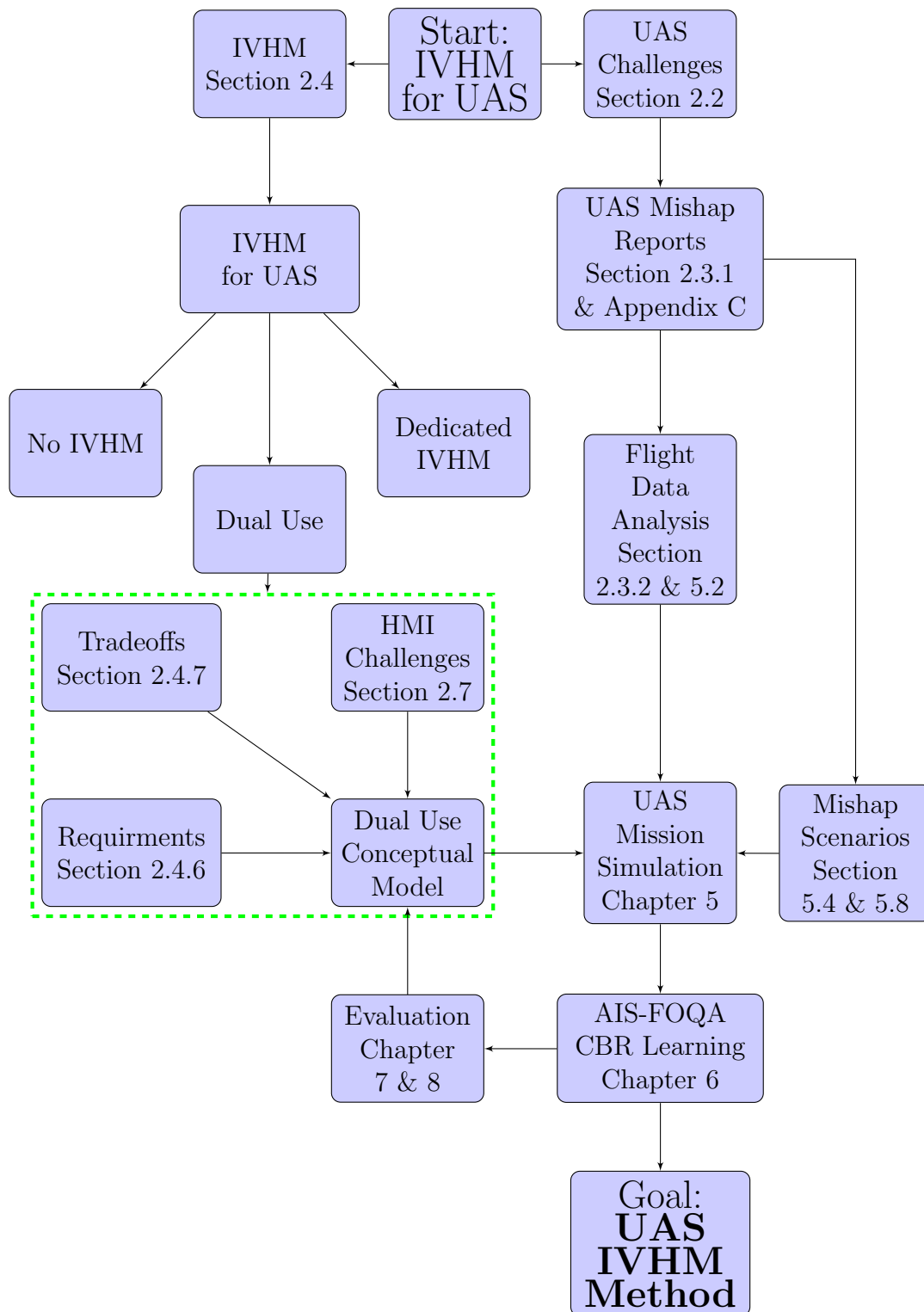


FIGURE 1.3: Thesis Logic

1.6 Summary

In this section the methods by which health management for UAS could be explored have been discussed. There is a gap for machine learning solutions to leaving from flight data in a way that preserves the ability of operators to learn from the information. Case based Reasoning helps by structuring the mishaps and if automated methods can be created to connect cases to new potential faults this will help operators to manage large fleets of UAS. It was decided to create a dataset of flight data including faults using fault simulation within a sequence of simulated UAS flights. This has the benefit of allowing the generation of a large dataset suitable for evaluating different machine learning approaches.

Chapter 2

Industrial Background

2.1 Introduction

UAS(Unmanned Aerial Systems) have grown from their genesis almost a hundred years ago with the Sperry aerial torpedo[\[11\]](#), to become sophisticated, highly developed, and complex systems. In this section their use will be examined, the needs of their operators, and how their operations may be improved in future though some original research.

- In Section 2.2 UAS will be introduced and discussed
- In Section 2.3 UAS usage will be discussed
- In Section 2.3.2 the methods by which UAS operations can be analysed will be discussed and how conclusions can be applied to improve operations.
- In Section 2.4 IVHM will be introduced and its applicability to the problem of UAS health management discussed.
- In Section 2.5 the difficulties in competent control of a remote vehicle are discussed as well as the implications for asset health management.
- Cognitive Asset Health Management in Section 2.7

2.2 UAS

Unmanned aircraft are aircraft that are intended to fly without a pilot on board. They consist of several main system elements. This section will introduce UAS and discuss their use. Some of the difficulties of operating UAS and problems unique to UAS will be discussed. A generic UAS architecture is shown in Figure 2.1. The individual elements of the UAS shown in the figure are discussed within this chapter and the contribution of each to the through life health management of the system discussed. Briefly the chief portions of the system are the aircraft(See Section 2.2.1), the communication means(See Section 2.2.3), the GCS(Ground Control Station)(See Section 2.2.2), the pilot or pilots(See Section 2.2.4),and the wider support system(See Section 2.2.5).

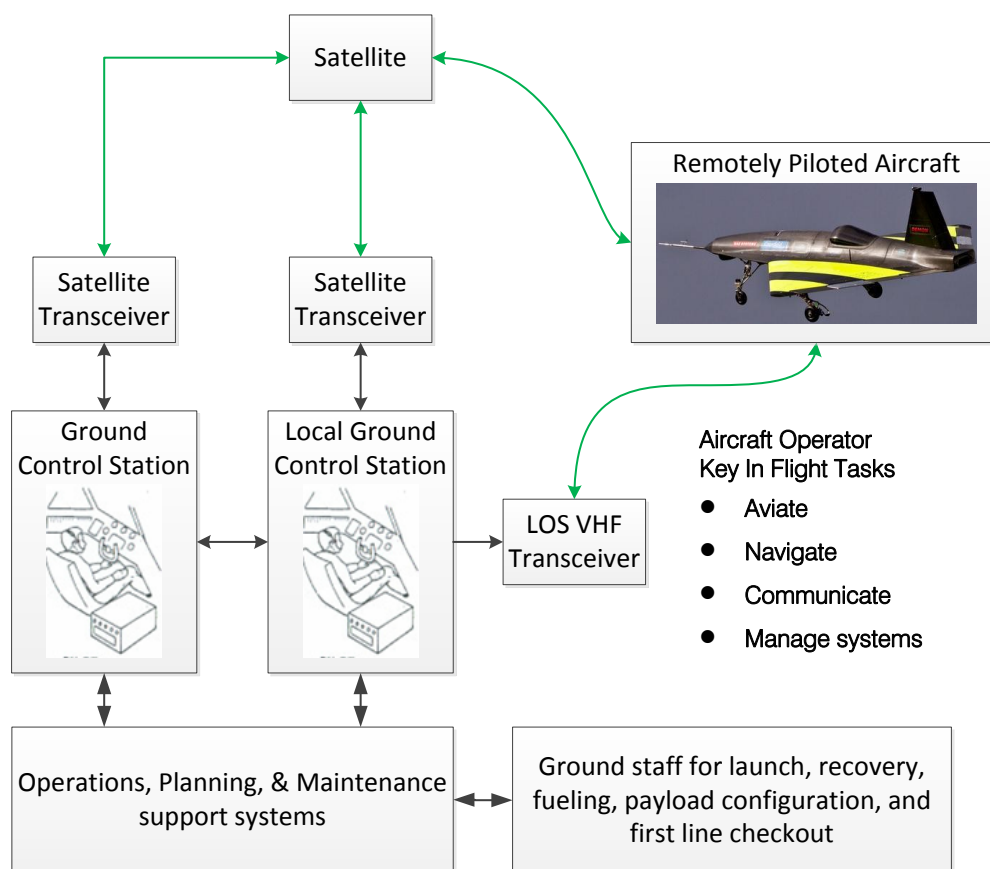


FIGURE 2.1: UAS System Architecture

2.2.1 The Aircraft

In this thesis the terms UAS(Unmanned Aerial System) and UA(Unmanned Aircraft) will be used as the methods discussed while primarily intended for remotely piloted aircraft systems are also applicable to autonomous systems. In situations where the whole system of UA, GCS, and supporting systems is meant then this thesis will refer to the system as a UAS. This is in keeping with general industry practice. Historically these aircraft types have also been referred to as drones. However this is confusing to those with an understanding of insects and the functions of a biological drone. Insect drones are typically male colony type insects whose functions do not extend beyond mating. However in communication with the man in the street acronyms can confuse but “drone” seems to now have a well understood meaning of a flying robot. Though it is easier to say drone to the general public to convey meaning the drone nomenclature will not be used in this thesis.

Autonomy

In the ICAO (International Civil Aviation Organisation) regulatory framework unmanned aircraft come in two general types[12]. Autonomous aircraft and Remotely piloted aircraft. Autonomous means that the aircraft can make independent decisions without reference to prepared rules or needing external guidance. ICAO has decided not to regulate autonomous aircraft as there is currently no method to assure control in all situations. The issue is one of safety and of liability in the event of an incident. However some manned aircraft or unmanned aircraft can have autonomous elements. In contrast, in remotely piloted aircraft, decisions can only be made by the remote pilot or a rule based deterministic system on-board. An issue unique to UAS is the discussion on the appropriate level of Autonomy in the aircraft. As discussed ICAO does not regulate autonomous UA as there is poor understanding of the risks and no framework for liability in the event of an incident. This area is discussed well in an AFRL(Air Force Research Laboratory) paper which sought to define levels of autonomy.

Automatic means that a system will do exactly as programmed, it has no choice. Autonomous means that a system has a choice to make free

of outside influence, i.e., an autonomous system has free will. For instance, let's(sic) compare functions of an automatic system (autopilot) and an autonomous guidance system:

- Autopilot: Stay on course chosen.
- Autonomous Guidance: Decide which course to take, then stay on it

(Quote is from B.T.Clough[13])

UAS therefore while not autonomous may have varying levels of autonomy/automation. This may vary during the flight depending on the situation and some regulation does require a reversionary direct control mode. Parasuraman and Sheridan[14] did a study on how the use of automation within complex systems could be specified for the purpose of contracts and man machine interface design. Their work produced a well-respected table which compares well to other metrics. This work was used in the creation of the PACT framework as part of the DERA (Defence Evaluation and Research Agency) Cognitive Cockpit project[15]and[16]. This is the PACT (Pilot Authority Control of Tasks) framework shown in Figure 2.2 which shows different combinations of control authority levels for the pilot and the computer within a system.

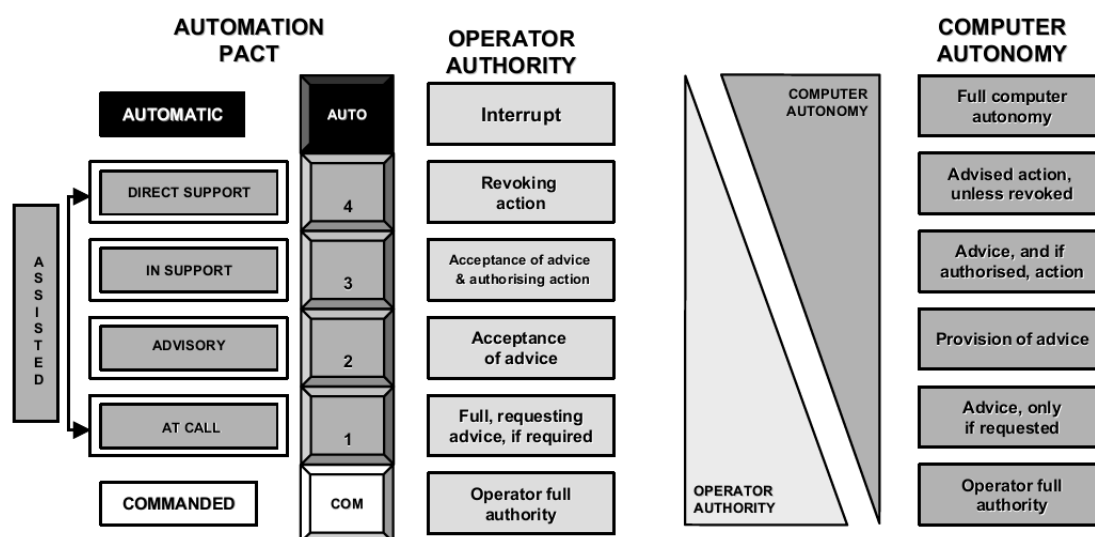


FIGURE 2.2: Pilot Authority Control of Tasks Framework (Figure 7 p11[15])

Aircraft Control Architectures

Unmanned aircraft come in a variety of sizes, architectures, and complexity. Listed here are some general architecture types in order of increasing complexity and capability.

UA Aircraft Architecture Types

- Analog → Sophisticated PWM(Pulse Width Modulation) signals created using shift registers and resistor ladders.
- Single Board → One circuit board processes commands from the controller and emits necessary signals to the servos and ESC(Electronic Speed Controller) of the aircraft.
- Federal → One central board which manages the aircraft. Additional boards controlled directly with no configuration changes during flight.
- Distributed → Multiple flight management computers

2.2.2 The Ground Control Station

UA pilots control their aircraft from a ground station. This forms an integral part of the aircraft system. These include a piloting control system and a method of communication with the aircraft. More sophisticated UAS can have ground control systems similar to the cockpit of a manned aircraft in size and complexity. Less sophisticated UAS have ground control stations that are very similar to those used with radio controlled models. There are many variations in between these two extremes. It is also typical for there to be some way to access data or imagery from the aircraft for analysis by an observer.

2.2.3 The communication backbone

There must be some way to communicate between the GCS and the UA. There are quite a few methods in use which vary widely. A small selection is listed here.

Communication methods

- Visible and Near Visible

- Infra-Red - Typically used for micro size UAS for use indoors. Widely expected to be a communication means for certain high altitude drone Internet hotspots.
- Visible and Ultra-Violet coherent laser used for slow moving or stationary aircraft such as an aerostat.
- Radio
 - 433MHZ ISM Band Various modes (Long range low bandwidth control Small and medium UAS)
 - 2.4GHZ ISM Band Spread Spectrum (Typically RC models and small commercial UAS)
 - 2.4GHZ ISM Band 802.11 (Commercial short range multirotors)
 - 2.4GHZ ISM Band ZigBee (Small UAS)
 - VHF Band Various modes (Long range LOS (line of sight) large UAS)
 - X Band Satellite Various modes (Long range BVLOS (beyond visual line of sight) very large UAS)

The main large UAS such as Predator[17] and Global Hawk[18] both use LOS VHF radio for take-off and landing from a local GCS before handing off to an X band satellite radio control from a remote GCS. The Cranfield University Demon[5] UA shown in Figure 2.1 uses a LOS ISM band spread spectrum radio for remote control and a different type of LOS ISM band spread spectrum radio for telemetry and digital communications to the aircraft flight computer. The architecture depicted in Figure 2.1 is more representative of a large UAS like Predator or Global Hawk. It should be noted that no communications method has infinite bandwidth or zero latency and so will always create some design compromise which will need to be taken into account when operating the vehicle.

The main conclusion should be that despite the wide variety of methods to get data to and from the vehicle no solution exists with zero latency and no solution has infinite bandwidth so any health management solutions will need to operate within those constraints. The ability to diagnose a fault, give a prognosis, and plan appropriate responses, while the human in the loop is on the end of some communications chain, is a challenge which has no perfect solution.

2.2.4 The Human element

Arbitrary delays, spatial separation of cause and effect, and discrete, discontinuous sub-systems tend to obscure cause-effect relationships. The pilot is insulated from the aircraft and develops a completely different image of the system he is operating than he would if the computers were not there. Consequently, any failure of the computers (either due to electro-mechanical failure or an unexpected situation) requires the pilot to intervene in a system with which he is not currently familiar. (Quote from Air Marshal MacFadyen[19])

The ability of the pilot to anticipate the actions and potential emergencies of the aircraft has been noted to vary widely with experience and takes some time to master[20]. This situational awareness built from experience and specific training is widely understood to be a force multiplier in dealing with emergency situations but the approaches to developing this competence remain varied due to the natural variation in the mindset and personality of pilots. The ability of the human to manage complex aircraft has been discussed widely and different strategies for a more human centred aviation have been proposed[21]. The key challenge that has drawn the most focus is how to present data in such a way that it is suitable for the pilot to easily absorb it and understand it in context. As expressed by Billings and paraphrased by the author here.

Data does not become information until it has been understood in the mind of the pilot. (paraphrase of Billings[21])

Even if the system trying to inform the pilot is 100% right every time and reliable if it cannot communicate its message effectively to the pilot it will remain a hinderance rather than a benefit. This issue is compounded in unmanned aircraft as the control link latency within UAS and the reduced available perceptions are thought to create additional difficulty to human control of the UA[22].

2.2.5 Supporting Systems

The supporting systems for UAS vary widely but often include the following elements

- ATC (Air Traffic Control)
- Maintainers
- Logistics for consumables and parts
- Management Structure - The management structure of a UAS is the structure which decides what operational procedures should be followed, the training needs of operators, and whether current operations are cost-effective and safe.

2.3 Current Usage

UAS are now used by a wide range of operators. Civil UAS are becoming available for a multitude uses. These include

UAS missions

- Photographic survey for mapping → Cheaper than an aircraft and more detailed than a satellite.
- Inspection at height → Safer and cheaper than rigging scaffold or ropes
- Crop assessment → Fertiliser usage can be reduced by monitoring foliage, pesticides and fungicides can be deployed only where needed given good aerial assessment.
- Geophysics surveys → Oil and mineral exploration - Able to replace manned aircraft in what is a dangerous flight regime due to the low altitudes.
- Border Patrol → Deployable resource with a better camera than could be deployed permanently while offering more flexible use and operation than fixed cameras or vehicle mounted pole cameras.

UAS compete with manned aircraft on a cost and safety basis. Small UAS can be deployed more quickly, safely and at lower operating cost than an aircraft. However aircraft can go places UAS currently cannot due to deficiencies in the ability of the UAS to cope with sharing airspace responsibly with other users. In the UK UAS are also limited in the distance they can go from their pilot during

operation. Larger UAS are slowly starting to emerge for use by civil operators. A wide variety of UA types were listed in the 2002 USA DOD Roadmap[23]. Several of the types listed did not enter production but it shows the breadth of types of UA envisaged for use. Different types of UAS currently experience different challenges associated with their use.

2.3.1 Mishap

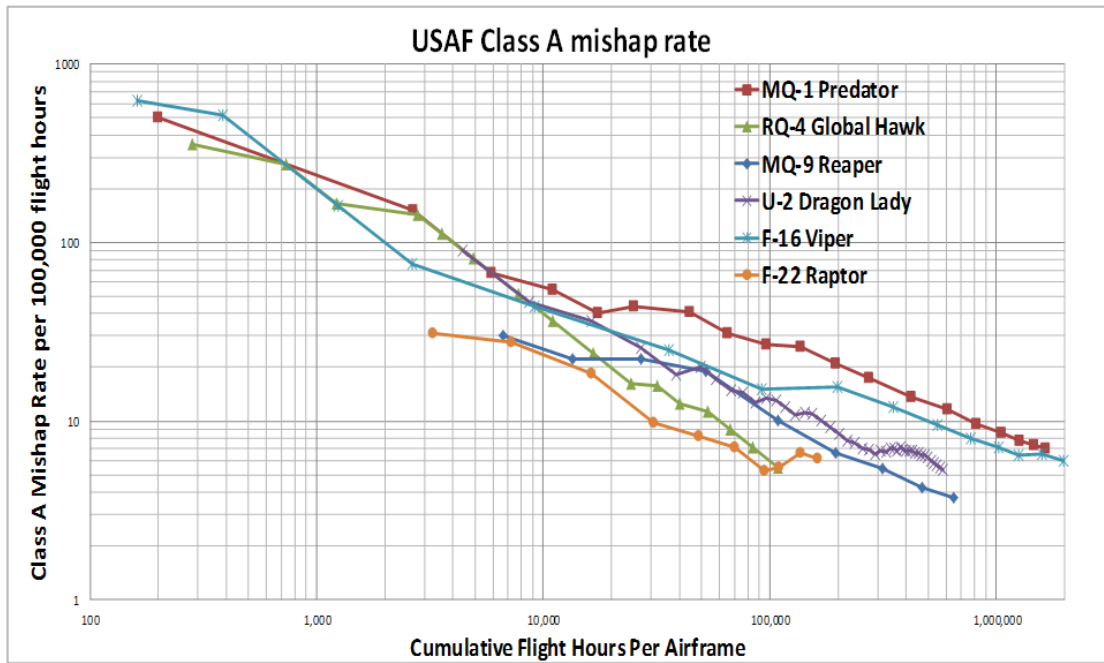


FIGURE 2.3: USAF Class A mishap rate trends(Various aircraft)[24]

All aircraft from time to time will have an incident or mishap. The difference between incidents and mishaps is that incidents *could* have caused damage while mishaps *did* result in damage, injury, or death.

Not all UAS have comprehensive statistics which can be examined but Predator and Global Hawk do have some well published and discussed mishap statistics. Incidents are not included but mishaps of varying severities over a ten year period were published for Predator as part of a study by Hartfield et al[25]. A summary of the class A mishaps for Predator and Global Hawk are shown in Figure 2.3 alongside class A mishaps for some comparable manned aircraft. It can be seen that after some initial development the UAS do not exhibit dramatically different mishap rates to manned military combat aircraft. It should be noted however

that the missions performed by UAS are cruise dominated. A better target for a mishap rate is that of commercial transport aircraft. Their rate is around 0.5 class A mishaps per hundred thousand flying hours. If UAS are to gain wide acceptance and see their operational costs go down this is the safety target for which the industry should aim.

2.3.2 Flight Analysis

There are two main methods to gain knowledge from aircraft operations. One is the generation of learning from lots of flights such as FOQA (Flight Operations Quality Assurance). The second is through detailed analysis of individual flights. Individual flight analysis does take place within FOQA but it is after attention has been drawn to a specific flight due to some anomaly.

2.3.2.1 Flight Data

Learning from aircraft flights in order to improve safety, reliability, and efficiency requires a set of flight data with examples to learn from. This presents us with a fundamental problem in creating a machine learning approach as within flight data there are several possible data types. The data provided by aircraft sensors throughout a flight is made up of sequence of tuples. A tuple is a way of representing data within a computer and can be explained as a structured set of values. In this research a tuple is a set of one samples worth of data from the flight data of an aircraft. Each of the values within a tuple represent the value for a single aircraft sensor from the time the tuple was sampled.

Types of flight data

... as we know, there are known knowns; there are things we know we know. We also know there are known unknowns; that is to say we know there are some things we do not know. But there are also unknown unknowns the ones we don't know we don't know. And if one looks throughout the history of our country and other free countries, it is the latter category that tend to be the difficult ones. The absence of

evidence is not evidence of absence, or vice versa. (Quote from Donald Rumsfeld 2002[26])

The flight data which we can analyse to develop health management systems for UAS can come from two general sorts of flights. The data can come from mishap flights or from non-mishap flights. However there is additionally two other characterisations that can be made of the data within these mishap and non-mishap flight data sets. The first is that we should expect to see flight data and aircraft behaviour which is similar or the same as data which we have seen before. We can create methods to recognise this data as normal behaviour. However it is should be expected that we will encounter flight data and behaviours which while novel are not a symptom of either a mishap, an impending mishap, or even an incident. Therefore within both mishap flight data and non-mishap flight data we should expect to see known and unknown(to our system) data. The second type of characterisation that can be made within the data from mishap flights is that within a mishap flight not all of the flight data and aircraft behaviour will be due to the mishap. Some portion of the flight data leading up to the mishap may well have been completely normal and consisting of the known and unknown data we would expect from a non-mishap flight. It is only after the mishap onset either by fault or some other event that we would wish to characterise the data as fault or mishap data. If the data cannot be segmented in this way it could be very difficult to use it to train any type of artificial intelligence to learn the features of faults and mishaps. A brief summary of the data types discussed follows.

Non-Mishap Types of data potentially present in Normal Flights

- **Known normal** data - tuples of flight data which have been seen before in normal flight
- **Unknown(or novel) normal** data - tuples of flight data which while representing normal flight have not been recorded previously

Mishap Types of data potentially present in mishap flights

- **Known normal** data - tuples of flight data which have been seen before in normal flight
- **Unknown(or novel) normal** data - tuples of flight data which while representing normal flight have not been recorded previously

- **Known mishap** data - tuples of flight data which only occur during a mishap
- **Unknown(or novel) mishap** data - tuples of flight data which while representing mishap flight have not been recorded previously

Thus data from mishap flights can include a subset of data within it which is normal flight data. Without exhaustive expert analysis it is difficult to determine where within a mishap flight the fault originated and where its effects might first be seen in the flight data. This creates a large set of potential false positives within classification due to the existence of normal flight data from the flight before the onset of the fault and the evidence of the fault is seen in the data. A conventional supervised learning machine learning would correctly be rewarded for classifying a non fault portion of the mishap flight as mishap. However as discussed this would greatly increase the false alarm rate of the algorithm. What is needed is a way to combine the classification strength of supervised classification algorithms(As discussed in section 3.3.2) and the space exploration and grouping methods of a clustering algorithm.(As discussed in section 3.3.1)

Flight Data Feature Space

The use of multi variable plots to illustrate boundaries and features within the flight data of an aircraft is reasonably well understood. Many manned aircraft have their flight characteristics predicted during design and these characteristics validated during the testing phase. These are then published in the manual to assist pilots in understanding the abilities and limitations of the aircraft. One well known type is shown in Figure 2.4 and is known colloquially as a dog house plot and is part of the family of plots which together define the flight envelope of the aircraft. This is a boundary plot of velocity vs flight load in order to help a pilot understand the key speeds of the aircraft such as the stall speed V_S , the critical manoeuvring speed V_C , and the never exceed speed V_D . Many UAS do not have this information due to a lack of understanding and investment during the design phase. This restricts the ability of operators to predict these properties before they have themselves encountered them in practice. Some aircraft make use of predicted flight envelope to restrict pilot choices and make it more difficult to damage the aircraft or risk an incident. Typically only used with large and expensive aircraft where the investment can be justified it is sometimes referred

to as care free manoeuvring and due to the benefits it is starting to be considered for small and cheaper aircraft[27].

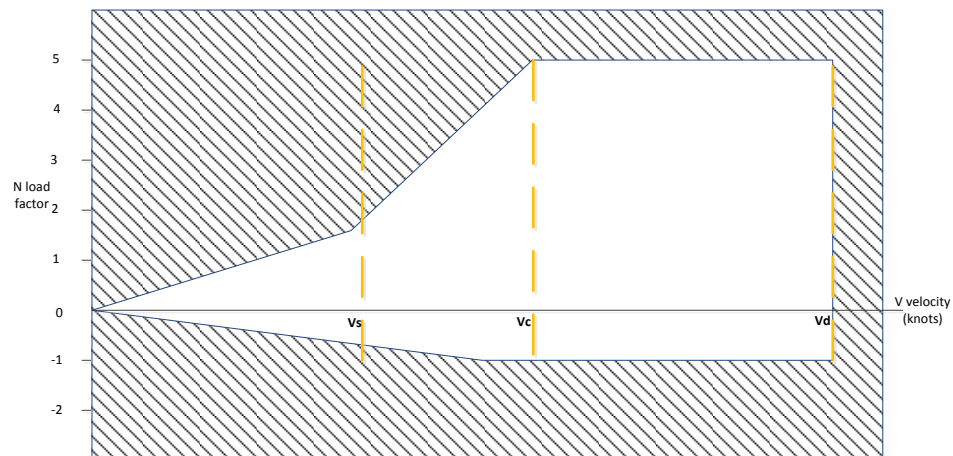


FIGURE 2.4: Aircraft V-N diagram. Used to show part of the flight envelope of the aircraft.

2.3.2.2 FOQA/FDM

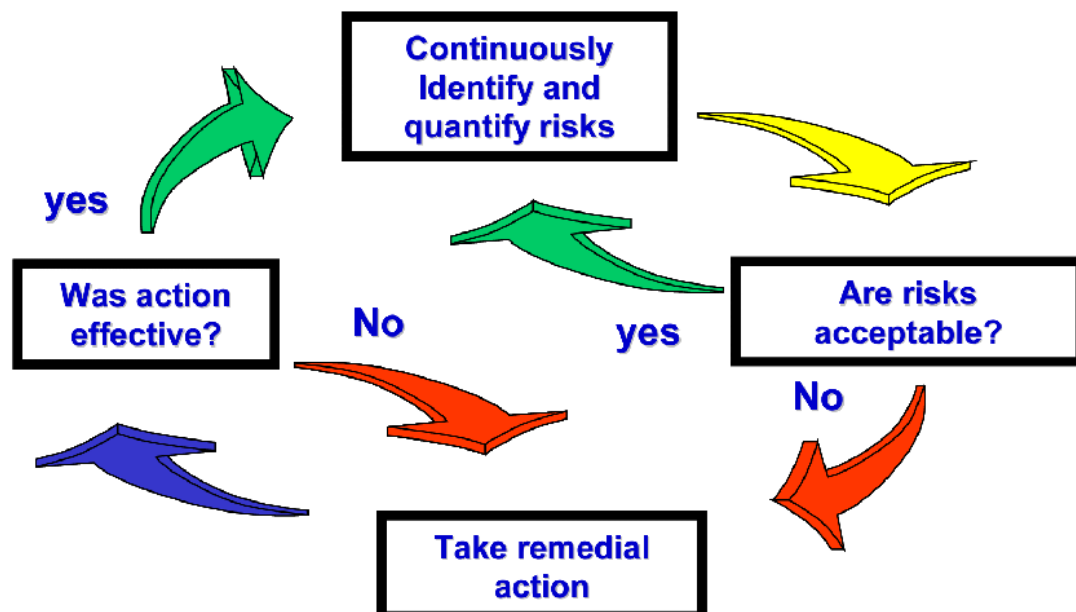


FIGURE 2.5: FOQA Risk Analysis cycle taken from CAP739[28]

FOQA/FDM (Flight Operations Quality Assurance/Flight Data Management) is a process pioneered by the UK CAA[28] which is intended to help improve aircraft operations by performing analysis of aircraft flights. This is structured as part of the operators SMS(Safety Management System) which attempts to identify and reduce risks while also reducing operational cost where possible. Figure 2.5 shows the risk identification and management cycle recommended within CAP739[28]. This is so that pilot training can be improved and that operations can be assessed to see if they can also be improved. This technique has become common among large airlines as a way of improving operations to ensure that everything possible is being done to reduce fuel burn and maintenance due to operations. Typically this is done by gathering the data held on the aircraft QAR(Quick Access Recorder). This holds flight data and cockpit recordings. Data mining techniques can then be used to compare pilot and aircraft performance across the fleet. Typically the performance of the same aircraft type is compared. However large FOQA datasets containing multiple aircraft types have been used to investigate more complex transport system integration issues[29]. One key issue with FOQA is understanding the pilots intent. The pilot may have a reason for diverting from flight norms which is not reflected in the flight data monitoring but which was too trivial for him to consider mentioning on a flight report. For continuing training statistical analysis is used to identify outlier events from analysis of flights which are otherwise unremarkable. Feedback on pilot quality can help improve operations as if they improve their flying this can which lower operating costs by reducing fuel consumption. It can also lead to less stress on landing and smoother flight transitions. IVHM data can actually be gathered in the same way as the FOQA system[30, 31] and used to help manage the health of the aircraft. FOQA can be seen as a process of novelty detection[32, 33], identifying unusual things and using expert judgement to decide whether they represent something that can be used to improve operations. Rolling average style analysis is sometimes used to try to allow understanding of the time sensitive context of some data compared to previous years and allowing accounting for seasonal variation. It also makes use of ranking for fuel consumption[33], climb rates, take-off rotation[34], descent rates[35, 36] etc. A potential use of FOQA for UAS is to use previous flight data to construct a model of normal range of values for throttle response at that altitude, speed, humidity etc which can then be compared with current flight values in the GCS to highlight anomalies. Various machine learning methods are used to try reduce the workload on analysts. Some examples include neural networks, nearest

neighbour classification systems, expert systems, and cluster analysis[29, 37–39].

FOQA Data Characterisation

The data recorded into the QAR (Quick Access Recorder) consists of several data types. The majority of the data is a record of continuous variables such as airspeed, pitch, roll, etc. The frequency with which the data is recorded is typically between 1 and 8Hz. There are various schemas used to format the data and these are described in standards such as ARINC 767 and 717[40]. The most common format is a fixed length frame based layout with a number of sub frames within it. Some logical values, alarms, and warnings are also logged to the QAR. New aircraft use more modern formats and methods and as the number of variables and the frequency of logging increase the file sizes can become very large. One airline quoted a particular new aircraft type could produce half a Terabyte in a single flight[41].

2.3.2.3 Incident Analysis

Incident analysis is the study of flight data from aircraft incidents and mishaps. The detailed analysis of individual flights often yields very valuable data. It is sometimes the only way to reconstruct what the aircraft was doing during an incident or mishap. Through the application of expert knowledge faults and errors can be identified and new diagnostic methods and/or training procedures developed to guard against them. The use of comparison between individual flights and fleet norms can also be useful for pilot training[42, 43]. Ground track and commanded heading can be extracted and examined to assist in the analysis. As an example the impact of external factors and pilot response into runway excursion can be examined to help improve future training[39]. In Appendix C USAF mishaps are discussed as reported through the USAF Safety Centre. Although Hartfield et al[25] ruled out the ground control link as the primary cause in many incidents it can be shown that the link is a contributory factor. Particular interest was given to a high profile incident where the main engine fuel valve was closed during flight. This led to a loss of aircraft power which prevented aircraft in-flight recovery procedures[44]. In the UK incident analysis is the responsibility of the AAIB(Air Accidents Investigation Branch) for civil incidents[45]. At present the view has been taken that unmanned aircraft are not included in the AAIB remit. However this may be changing as the wording of their empowering act lists

their responsibility as the investigation of aircraft mishaps and does not exempt unmanned aircraft. Various methodologies have been used to review incidents to try to identify lessons for the future. One lesson that is frequently discussed is the cognitive one and the burden aircrew may be under. The cognitive burden for UAS pilots is suggested to be higher than for manned aircraft and represent a risk due to the nature of operating over a remote link[46–48].

2.4 Integrated Vehicle Health Management

2.4.1 Introduction

Integrated Vehicle Health Management (IVHM) is the unified capability of systems to assess the current or future state of the member system health and integrate that picture of system health within a framework of available resources and operational demand. (Quote from p3 Integrated Vehicle Health Management: Perspectives on an Emerging Field[49])

The aims of IVHM or Integrated System Health Management (ISHM) are to enable better management of vehicle and vehicle fleet health. It could also be described as Integrated Asset Health Management (IAHM) as the application of integrated health management is also now frequently applied to fixed assets as well as vehicles.

Aims of IVHM

- Improve safety through use of diagnostics and prognostics to fix faults before they are an issue.
- Improve availability through better maintenance scheduling
- Improve reliability through a more thorough understanding of the current health of the system and prognosis based maintenance
- Reduce total cost of maintenance through reduction of unnecessary maintenance and avoidance of unscheduled maintenance

This is achieved through correct use of reliable sensing and prognosis systems to monitor part health and also using usage data to assist in understanding the load experienced and likely future vehicle load. However this is a very difficult proposition as the most optimal trade-off between cost and notional safety is very difficult to establish.

The principal barrier to the adoption of IVHM is the need to accurately assess the trade-offs between associated costs, risks, and rewards. (Finding number 8 from Benedettini, O. et al[50].)

This refers to sensor choice and the choice to use IVHM with the system. The lack of historical comparison makes it difficult to assess the exact amount of IVHM required to achieve a specific result and how to budget for it during conceptual design.

Unavoidable design defects, quality variations in the production process as well as different usage patterns make it infeasible to foresee all possible faults that may occur to the vehicle. (Quote from Djurdjanovic et al[51])

As the IVHM concept has matured, it has become clear that IVHM requires the same sensors as the real-time avionics monitoring functions to support diagnosis of subsystem problems. New sensors have been proposed to augment the avionics sensors to support diagnostics and prognostics. As the designs have been considered, a symbiosis has been realized where the real-time avionics can make use of sensors proposed for diagnostics and prognostics to make better real-time decisions in response to detected failures. (Quote from Paris et al[52])

2.4.2 History

It has been suggested that IVHM as a named concept has been around since the 1970s[53]. IVHM as a concept grew out of popular aviation maintenance methods such as CBM(Condition based maintenance) and was first named as a concept by NASA as part of their vision for intelligent maintenance and support for difficult to

repair assets. As sensors improved and the understanding of the systems concerned grew it became possible to not just detect failure but also to predict it. Faults causing unscheduled maintenance come with a high cost and can make the vehicle unsafe. The high unit cost and high maintenance cost of aircraft and spacecraft made any advance in maintenance methods very attractive. NASA used the name IVHM to describe how they wanted to approach maintenance of spacecraft in the future. They created NASA-CR-192656 in 1992 with the assistance of the General Research Corporation and the Orbital Technologies Corporation. This was a goals and objectives document in which they discussed the technology and maintenance concepts that they believed would be necessary to enhance safety while reducing maintenance costs in their next generation vehicles. Many companies since then have become interested in IVHM and body of literature has increased substantially. There are now IVHM solutions for many different types of vehicle from the JSF (Joint Strike Fighter) to commercial haulage vehicles. One of the key milestones in the creation of IVHM for aircraft was the series of ARINC standards that enabled different manufacturers to create equipment that would work together and be able to send diagnostic data from the aircraft to the maintenance organisation on the ground[31]. One example is ACARS (Aircraft Conditions Addressing and Reporting System) which is frequently used to communicate maintenance and operational data between the flight crew and the ground crew. This has led to concepts which have been adopted in IVHM. Another milestone was the creation of HUMS (health and usage monitoring systems) for helicopters operating in support of the Oil rigs in the North Sea. The progression of incidents and development of HUMS to try to make North Sea Helicopters safer is covered well by Land et al[54]. HUMS focused on vibration monitoring but also recorded helicopter usage as a way of tracking exactly how much airframe life was accumulating. This is concept that usage data can be used to assist maintenance planning is also key to the IVHM approach. FOQA or Flight Data systems are similar to HUMS as they monitor the vehicle usage. They are useful for IVHM in the same way as they allow the usage of the vehicle to be thoroughly understood which aids in the design of future vehicles. It also allows excessive loads and usage to be identified and corrected. For example if an aircraft was experiencing frequent heavy landings the maintenance schedule for the undercarriage could be changed to ensure that they are not wearing too fast under the increased load. The load carried by the aircraft could be lessened in future or operators could be given additional training to improve the quality of the landings.

2.4.3 Philosophy

IVHM is concerned not just with the current condition of the vehicle but also with health across its whole life cycle. IVHM examines the vehicle health against the vehicle usage data and within the context of similar information for other vehicles within the fleet. In use vehicles display unique usage characteristics and also some characteristics common across the fleet. Where usage data and system health data is available these can be analysed to identify these characteristics. This is useful in the identification of problems unique to one vehicle as well as identifying trends in vehicle degradation across the entire fleet. IVHM is a concept for the complete maintenance life cycle of a vehicle (or machine plant installation). It makes extensive use of embedded sensors and self-monitoring equipment combined with prognostics and diagnostic reasoning. In the case of vehicles it is typical for there to be a data acquisition module on-board and a diagnostic unit. Some vehicles can transfer selected data back to base while in use through various RF systems. Whenever the vehicle is at base the data is also transferred to a set of maintenance computers that also process that data for a deeper understanding of the true health of the vehicle. The usage of the vehicle can also be matched to the degradation of parts and improve the prognostics prediction accuracy. The remaining useful life is used to plan replacement or repair of the part at some convenient time prior to failure. The inconvenience of taking the vehicle out of service is balanced against the cost of unscheduled maintenance to ensure that the part is replaced at the optimum point prior to failure. This process has been compared to the process of choosing when to buy financial options as the cost of scheduled maintenance must be balanced against the risk of failure and the cost of unscheduled maintenance[55]. This differs from Condition-based maintenance (CBM) where the part is replaced once it has failed or once a threshold is passed[56]. This often involves taking the vehicle out of service at an inconvenient time when it could be generating revenue. It is preferable to use an IVHM approach to replace it at the most convenient time. This allows the reduction in waste component life caused by replacing the part too early and also reducing cost incurred by unscheduled maintenance. This is possible due to the increased prognostic distance provided by an IVHM solution. There are many technologies that are used in IVHM. The field itself is still growing and many techniques are still being added to the body of knowledge.

2.4.4 Architecture

Health monitoring sensors are designed into the vehicle and report to a data processing unit. Some of the data may be manipulated on-board for immediate system diagnosis and prognosis. Less time critical data is processed off-board. All the historical data for the vehicle can be compared with current performance to identify degradation trends at a more detailed level than could be done on-board the vehicle. This is all used to improve reliability and availability and the data is also fed back to the manufacturer for them to improve their product[49]. The conceptual architecture of the IVHM system as expressed by Jennions et al[49] can be seen in Figure 2.6. The main constituent elements are Sense, Acquire, Transfer, Analyse, and Act. The system is not intended to replace safety critical warnings such as an

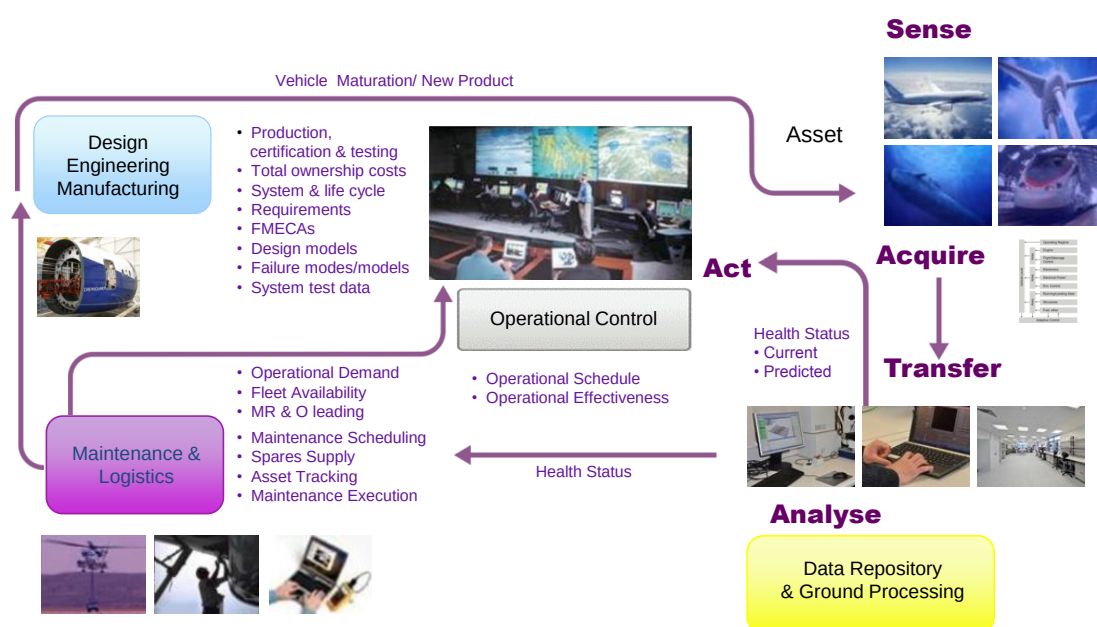


FIGURE 2.6: IVHM System Architecture from Jennions et al[49]

aircraft flight management system but instead to complement them and perhaps also leverage existing sensors for assistance with system health monitoring. Ideal systems to monitor are those systems, subsystems and structural elements which are likely to show graceful degradation so that they can be repaired or replaced at a convenient time prior to failure. This gives a saving over condition based maintenance as once a part has failed often a vehicle cannot be used until repaired. This often results in scheduling difficulties if the vehicle fails when it was needed for revenue generation and cannot be used. In contrast when IVHM is used the need for the part to be replaced can be detected ahead of time. This means that the

part can be replaced during normal scheduled vehicle downtime prior to failure. This ensures that it can continue to generate revenue as scheduled. Communications between the vehicle and the maintenance organization are crucial to fixing faults in a timely manner. The balance of how much data should be sent to the maintainer during use and how much should be downloaded while in maintenance is a one that must be judged carefully. One example of this is what is known as fault forwarding. When an aircraft experiences a fault the flight management system reports it to the flight crew but also sends a message through ACARS to the maintenance team so that they can start their maintenance planning before the aircraft has landed. This yields a time advantage as they know some of the parts and personnel required to fix the fault before the aircraft has landed. However the communication link does cost money and has a limited bandwidth so the worth of this health and usage data must be judged carefully with consideration given as to whether it should be transmitted or merely downloaded during the next maintenance or as part of the operator shutdown process.

2.4.5 IVHM System Design

2.4.5.1 Introduction

In this section the design of an IVHM system is discussed along with elements specific to UAS, their constituent parts, their functions, and how they affect system use. The IVHM system characteristics must be understood if designers, pilots, and managers are to make informed choices during UAS design and operation.

2.4.5.2 System elements

There are several initial questions which must be addressed in the design of an IVHM system for an aircraft, for example identifying which systems require IVHM is the first step. This question will be addressed in the unmanned aircraft section of this review where system failure rates and criticality will be assessed. The aircraft and its on board systems cannot be considered in isolation. As an IVHM system is a system of systems, it can only be successful for an asset if consideration is given to how it would be used and whether the addition of the system would be operationally effective. Some asset systems such as lighting may be easy and

quick for maintenance staff to assess and repair without affecting operations and so it would not necessarily benefit the through life support of the asset for an IVHM system to be developed at a cost that is unlikely to be recovered through operational savings of decreased or convenient maintenance.

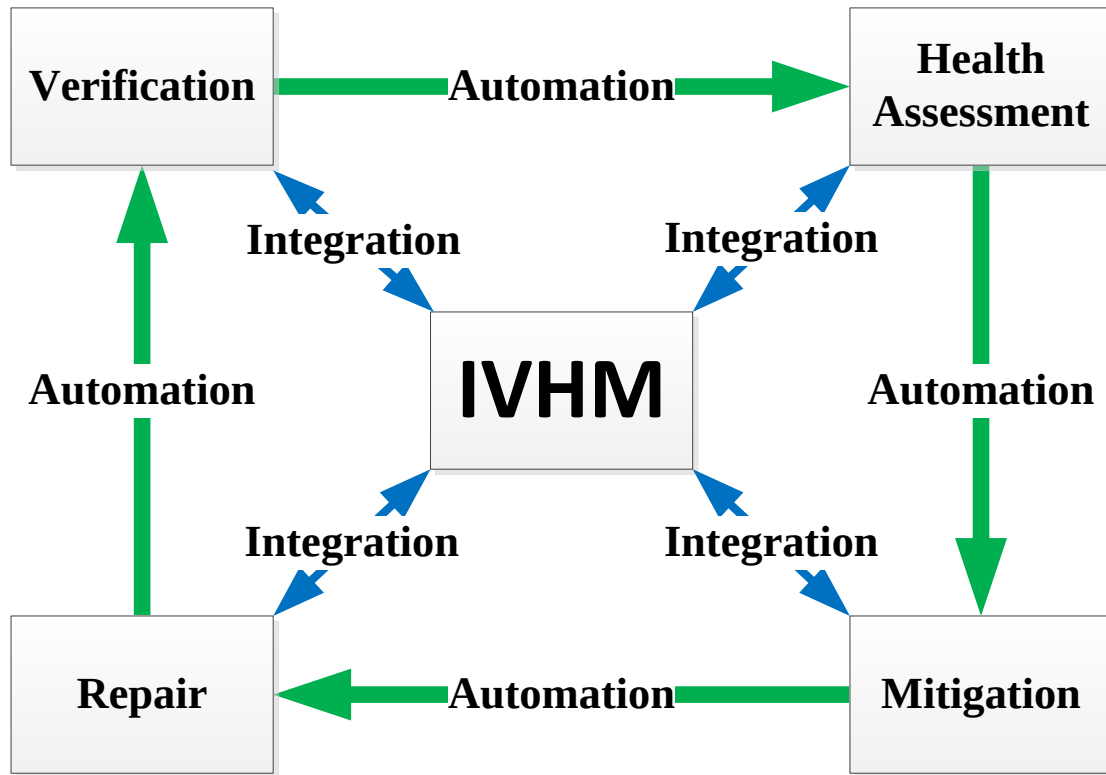


FIGURE 2.7: IVHM System Operational cycle from redrawn for clarity from Scandura et al[57]

The operation of an IVHM system comprises of four major elements as discussed by Scandura et al[57] and shown in Figure 2.7. Health Assessment, mitigation of health issues, repair of the asset to restore its health, and verification to ensure that the health has truly been restored to the asset. Figure 2.7 also emphasises the importance of integration and automation to try and keep these elements linked together and cohesive so that the operators can use it effectively. There are several alternative phrases used within literature when discussing an integrated system of sensors, analysis and maintenance. IVHM and its associated technologies are intended to address the entire vehicle lifecycle. This thesis focusses on aircraft usage of IVHM but much of the topic is common with other vehicle and asset management methods. There are really two things generally meant by IVHM and the associated acronyms.

- Health management of an asset → Asset Management
- Health management of the asset in the context of a fleet of similar assets → Fleet Management

In general the field is one of AM(Asset Health management) but some of the different approaches are identified here and some of their subtle differences drawn out.

- PHM (Prognostic Health Monitoring)
- CBM (Condition Based Monitoring/Maintenance) Typically condition based maintenance involves triggering a maintenance action when some monitored condition rises above a pre-determined threshold.
- BIT (Built in Test) or BITE (Built in Test Equipment) A health management strategy originated for avionics. Avionics by their nature can be harder to diagnose than mechanical components. It is not immediately apparent that they have failed except through their output which can be incorrect instead of merely absent. BIT was added so maintainers would be able to tell from a code or indicator whether a system was working correctly or not. Sometimes this took the form of push to test buttons.
- ISHM (Integrated System Health Monitoring)
- IVHM (Integrated Vehicle Health Management)

IVHM also covers the use of health trend information for assets to assist in planning the replacement and disposal of assets. The health data can inform the operational paradigm the new assets are designed to meet and also lead to changes in current operations to optimise use. The information informs parts ordering and maintenance scheduling. However when IVHM is discussed sometimes the only thing meant is the monitoring the health of the asset. This ignores the other system elements which add value but can be complex to implement.

Health State Determination

Health state determination is the primary focus of IVHM. It encompasses the methods by which IVHM monitors a system or part and diagnoses its present

state of health. It also covers those prognostic methods that predict the RUL (Remaining Useful Life) of the system. The mechanisms by which this is done will be discussed further later in this thesis.

Mitigation

Mitigation is generally considered to be an in-flight mechanism for improving the survival chances of an aircraft following a system failure. There has been a large amount of research in this area. It is an area of weakness for unmanned aircraft as it can be difficult to design a robot that meets deterministic avionic software requirements and is also able to deal with in-flight emergencies that have not been foreseen or which are presenting in an unusual way. Mitigation methods include

- **Operational restriction** Operational restriction is aimed at reducing the impact of a failure and/or preventing an impending failure from occurring by reducing the flight envelope. It can also take the form of reducing the mission time to land before a fault becomes critical. One well known method is the use of an MEL (Minimum Equipment List). If a piece of equipment fails which is not on this list an aircraft operator can decide to keep operating the aircraft and repair the system at a later date. The restriction is that the pilots are instructed that they cannot use that system and must adjust their flight plan to take any impacts into account. Typically the MEL consists of safety or mission critical hardware. The RAF have a similar method where they either agreed the fix of a fault can be deferred or that no action is required in the short term as long as the operation is restricted in some way.
- **In flight reconfiguration** During flight when a fault is found or is expected to appear sometimes systems can be reconfigured to reduce the risk of failure or its impact. Common examples include flight control systems which use multiple modules with forms of voting between them to ensure that if a module fails the healthy modules outputs are used rather than the failed module.
- **In flight repair** Methods to improve flight control software and its ability to cope with damage have been demonstrated by Jourdan et al[58]. It was demonstrated in a particularly dramatic way as their UA ejected 80% of one wing during flight and without prior preparation developed a method

of coping with the damage and was able to land safely automatically. However their method may be difficult to flight certify due to questions over its predictability.

Repair

This section of aircraft domain knowledge will be only briefly addressed. If the benefits of IVHM are to be realised then repair and maintenance efforts need to be well integrated to the system. There are unfortunate examples of health management techniques that have not been effective because they have not been successfully integrated with those actually maintaining the aircraft[59].

Verification

Verification concerns the series of ground and flight tests that must be completed following repair to return the aircraft to flight. Unmanned Aircraft cannot make a pilot in command judgement as there is no pilot on board. The quality of data available to the remote pilot is not generally good enough to make a useful assessment. However it may be possible through use of FOQA (Flight Operations Quality Assurance) that the control of the aircraft can be compared to its control prior to failure and a judgement made on the success of the repair action. A judgement can then be made on whether it is safe to return the aircraft to service.

These four areas of integration will be addressed more thoroughly later in this paper as how an unmanned aircraft will fit within this system and where the differences to a manned aircraft will be.

2.4.6 System Requirements

2.4.6.1 Non-functional Requirements

Non Functional Requirements specify constraints and qualities of the equipment such as weight, volume, physical mounting and environment codes (ATEX, IP rating etc)[60]. The aircraft the system will be in and its mission will determine exactly how much power, mass and volume will be available for the IVHM system. Some requirements such as usability for maintenance engineers have a large impact on how successful an IVHM system will be. The system must support maintenance

action and also be suited to data mining to support long term analysis of usage and long term maintenance forecasting. There are several types of user for IVHM data and they all have different access requirements that must be considered when designing the system[59]. Non-functional IVHM requirements could be said to limit the search space of all possible IVHM solutions. In IVHM the non-functional requirements of the system are some of the major challenges for IVHM on an unmanned aircraft. Unmanned aircraft are typically smaller than their manned equivalent and are more highly optimized. This restricts the amount of mass, power and volume can be made available for integration of an IVHM system.

2.4.6.2 Functional Requirements

Functional requirements specify how equipment behaves, what it does, and how it does it[60]. The functional requirements of the IVHM system listed here are all concerned with the function of the diagnostic and prognostic algorithms. Whether they are physics or data based algorithms their success cannot be accurately assessed until the aircraft has been in service for some time and a quantity of data from their use can be assessed. This creates a risk during design as it can be difficult to predict how much effort should be spent to ensure a system meets the requirements. Traditional de-risking methods have involved system and component suppliers providing their own prognostic and diagnostic methods. However this has proven complex to integrate into a whole aircraft system as different manufacturers often each have their own preferred methods, protocols, and system philosophies and each will frequently claim that theirs is the world leader. Only through common methods and standards agreed between all the system suppliers can IVHM for the whole asset realistically be achieved. Finding the optimum IVHM solution for a manned or unmanned aircraft is a very complex multiple domain problem. This is made more difficult by the lack of available data from previous IVHM system designs that could be used to assess what the constituent elements of a successful IVHM system are. Aircraft conceptual design is aided by historical data on typical mass, volume, and power requirements of systems[61]. This has been built up for many aircraft types over several decades. Clear trends have been established which can inform how much of an aircraft will be fuel, navigation or engine for given aircraft types. This information does not yet exist for IVHM systems so the conceptual sizing based on historical trends approach cannot be used to describe how large a part an IVHM system should take of the finished

aircraft. This type of method must wait until IVHM has become a more mature technology. This would be when operational data for several aircraft equipped with representative IVHM systems is available. This would allow a reliable comparison of design choices and identification of characteristics of a successful IVHM system.

2.4.7 IVHM for UAS

The choices to be considered when designing an IVHM system for a UAS are varied but the first one to be considered is whether any on-board systems are required at all. The examination of these choices is part of the scope of this PhD and will be narrowed down in the gap analysis(section 1.2) to focus on a particular area of those choices.

- How does the in-flight reporting to ground change for an UAS?
- How does the choice of which systems require IVHM change for a UAS?
- Does the GCS require some form of IVHM?

2.4.7.1 IVHM requirement changes with mode

During the flight of an aircraft the pilot will be evaluating the aircraft response to their control input. This enables them to take action when they judge an issue to have arisen. This may be anything from gusts causing a high pilot workload to keep the aircraft stable or an engine failure requiring an immediate preparation for landing. Multiple types of issue may occur during the same flight and even at the same time. They can be caused by human error, machine failure, or environmental conditions. All of these have different operational planning techniques to help guard against some forms of each type. Weather forecasts are examined to guide operations and avoid weather likely to imperil the flight. Pilots are required to plan their flight beforehand and practise regularly to keep their skill level up. Maintainers use various checks to try to ensure maintenance tasks and aircraft prep is carried out correctly. IVHM is intended to assist with addressing machine failure issues in flight. FOQA is intended to assist with the other two issues. While trying to understand the environment within which IVHM for UAS will

be employed it seems that there are different scales of risk that cause a need for different types of management. Particular attention should be paid to the level of abstraction of information the pilot is given. As the pilot becomes more remote from the aircraft it can be more difficult for them to perceive issues that have arisen.

VLOS - Visual Line of Sight

Hazard assessment for VLOS operations tends to lead to a safety plan where risks to life and property are mitigated by creating a virtual sterile environment within which the UAS can be flown and if it starts to leave this area on-board algorithms terminate the flight. If radio contact is lost the flight is also terminated automatically to keep it away from property and people. This impacts operations as a standoff distance must be observed to objects and life to ensure that if radio contact is lost the vehicle can be terminated before impacting something valuable. The failsafe system used by Demon and many other VLOS UAS is based on the concept of *cutting down* the aircraft following loss of communications before it can leave the sterile area. The LMA (Large Model Association) base their safety case on the ability of the pilot to remotely direct the aircraft to crash in a safe place in the event of an imagined systems failure where there is some amount of control authority remaining. This is appropriate for their typical operating environment of flying within the boundaries of an airfield[4]. An IVHM system in a VLOS UAS then may have as a requirement a prognostic distance to critical system failure such that the aircraft could be turned away from a course that could lead it outside the safe zone in the event of the failure.

EVLOS - Extended Visual Line of Sight

EVLOS operations are operations, either within or beyond 500 m / 400 ft, where the Remote Pilot is still able to comply with his collision avoidance responsibilities, but the requirement for the Remote Pilot to maintain direct visual contact with the UA is addressed via other methods or procedures. It is important to note, however, that collision avoidance is still achieved through visual observation (by the Remote Pilot and/or UA Observers). (Quote from CAP722[62])

If UAS are to operate outside of semi sterile confines and beyond LOS (Line of Sight) of the pilot then the ability to crash in as safe a manner as possible without pilot intervention is something that some thought will need to be put into. It would be pointless to use a flight termination style failsafe if the pilot cannot guarantee the terrain the aircraft is likely to terminate its flight into. EVLOS is a confusing definition as the pilot is expected to ensure safe separation through visual means and part of this is estimation of probable trajectory and impact point in the event of failure. As the distance to the aircraft increases it becomes very difficult to estimate this accurately. It is expected that computer aids will be used to assist the pilot in this estimate while they maintain visual observation. Operations plans could also be part of this by restricting operations to over areas where a crash would be costly but not a risk to life. One operator with EVLOS permission is Project Ursula[63] and their operations are over forests and fields. This presents a low safety risk to crash into in the event of a mishap. The specifics of their operations have not been released but it is understood their UAS flies an automatic path via waypoints and the waypoints are selected to allow pictures of the points of interest to be gathered.

BVLOS - Beyond Visual Line of Sight

BVLOS flight can only be assured if the entirety of the route planned for the UAS is over semi sterile environments and if the UAS can terminate its flight without pilot intervention in the event of loss of radio link or some flight emergency which causes the UAS to deviate from its course. This would be equivalent to the behaviour of manned aircraft. Pilots of aircraft in uncontrolled descent tend to try to direct their aircraft to descend in such a manner to increase their survival and reduce the risk to others.

It is unlikely in the short term that an UAS below a certain weight would be able to sustain the amount of computational power to dynamically plan the response required to handle an in-flight emergency after communications have been cut. If as seems likely from CAP722 airworthiness requirements are needed to gain permission for flight in such a mode the amount of redundancy required would almost certainly put the requirements beyond any small UAS.

Current UAS abort their mission in the event communications are lost. This is often implemented as a return to base protocol or the UAS will orbit while trying to get back into communications with the GCS. However instances have come to

light where the wrong coordinates are loaded into the UAS to return to when communications are lost due to launch and recovery site changes. The mishaps seen from the USAF RQ-4, RQ-1, and MQ-9 fleets show that losing the link often leads to a class A mishap[64]. However this is a negative information bias situation[65]. It is not known how often the link is lost and the aircraft is recovered without mishap. The ratio between these two would be more informative than just examining the lost link mishaps.

It seems apparent that IVHM for BVLOS aircraft would need to pay close attention to the health of the communications link.

2.5 Cognitive Control

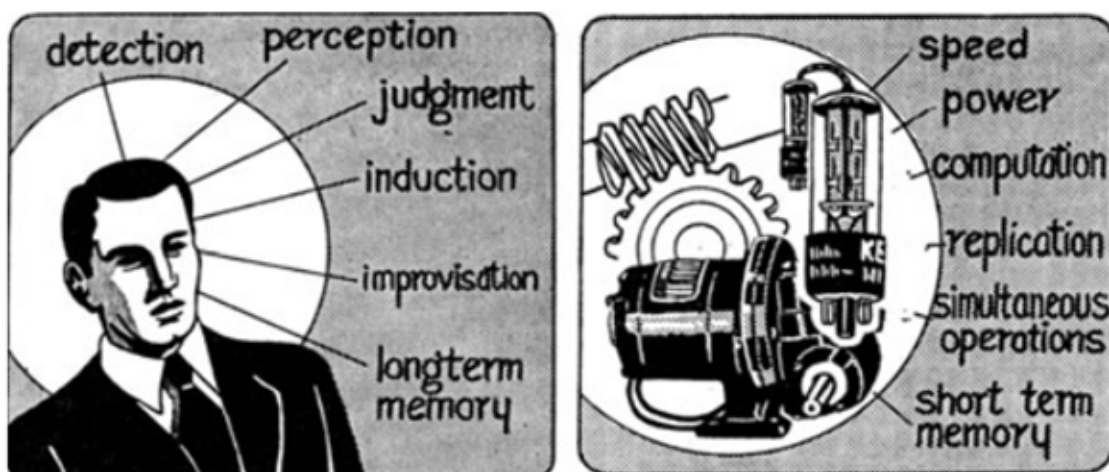


FIGURE 2.8: Fitts List. A decomposition approach to man machine interaction[66].

A remote pilot has more barriers to maintaining their situational awareness of the aircraft and its health state than would the pilot of a manned aircraft. As indicated by Fitts[66] and later research a machine does not suffer from cognitive overload as long as the system was designed to handle and process the input. Figure 2.8 shows the Fitt's list and illustrates the idea that from a list of possible types of task humans will be better in some areas and machines better in others due to their unique qualities. Coming from the era of vacuum tubes and mechanical computer this shows foresight. It is a decomposition approach and does not discuss the interaction between humans and machines which has become a much larger problem in the years since. People cannot be designed in this way

and much of aircraft in-flight reporting systems have been specifically designed to balance information content against pilot workload. Detailed health information does no good to the pilot if they cannot process it. Maintenance for UAS has also been approached in a different way to maintenance for manned aircraft[67]. Maintainers have adopted different attitudes due to the perceived low severity of an unmanned aircraft mishap compared to a manned aircraft mishap. No personnel in danger leads to relaxation on the part of maintainers rather than adhering to the same standards found in manned aviation. There were similar concerns during the introduction of glass cockpit systems which continue to the present day. Pilots exhibit a wariness of fly by wire aircraft as it is not always transparent why an automated system takes a specific action and many automated actions are not directly reported to the pilot. There are three common questions pilots have when interacting with glass cockpits and FMS(Flight Management system)[68]. “What is it doing?”, “Why did it do that?”, and “What will it do next?”. These questions occur when manned aircraft pilots lose situational awareness of the aircraft automation. Aircraft systems are very complex and generally designed to report rather than discuss information. Thus it is easy for the pilot to lose context and situational awareness. This is also a high risk for UA control from a GCS. UA have inherent in their nature a higher degree of automation than manned aircraft, and in addition to this they are operated via a remote link.

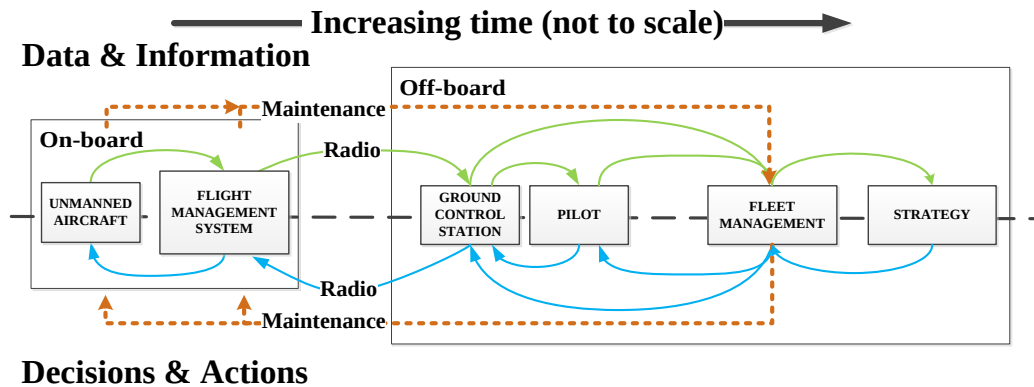


FIGURE 2.9: Depiction of process flows in UAS operation to illustrate the different decision timescales

The operation of UAS has many different decisions that need to be made across varying time scales for decisions within multi scale OODA loops[69]. The maintenance of a fleet of UAS also encounters multiple timescales and layers. This is

illustrated within Figure 2.9 to show how these multi scale decisions can complicate the process of learning from operations and planning effective strategies for use. This sort of learning is often referred to as through life learning as the need to learn from operations in order to improve is present from design and test, through aircraft acceptance, up until the aircraft is retired from service. After the aircraft type is retired the lessons learned through life still need to be understood to help inform the design of the next generation of aircraft. This multi-scale nature serves to demonstrate how the operational experience of the aircraft and best practice needs to be captured through life. Modern aircraft are often in service for decades and faults discovered in early years and learned by operators will need to be retained in a way that new operators can use after the early adopters have since retired.

UAS through life learning UAS generate large amounts of data during flight and some of this may be repackaged and sent over the telemetry link to the GCS for use by the operator to assist them in understanding the state of the UAS and making mission decisions. The remainder is used within the UAS by its systems. Some portion of the dataset is retained on the flight data recorder for later analysis and for investigation in the event of a mishap[70]. This gives two sources of data which can be integrated to enable analysis of operations. This much would be present within many SMS (Safety management systems) for civil aircraft[71]. The review of flights by an expert and the generation of knowledge which can be used to inform future operations. The personal experience of the pilot has often been cited as an important asset in the ability of the pilot to react appropriately in emergency situations. Understanding the aircraft state is agreed as a key step to mitigating faults and failures during flight. This presents a problem within the UAS community however as several cognitive factors have been identified which place additional burden upon the pilot in their effort to understand the current state of the aircraft[72]. The use of IVHM methods to give advice to the pilot regarding mitigation options may be helpful in addressing this problem. Asset health management of any sort relies on a deep understanding of the asset under management, its operations, degradation modes, maintenance methods, and part requirements. During the design phase of an asset these things can be difficult to predict with sufficient accuracy. This problem is especially acute when novel systems are present which have not had IVHM systems used with them in the past. UAS of one form or another have been in existence for almost a

hundred years[73] but their modern form has not been the subject of IVHM work. The predator RQ-1/MQ-1 has been one of the most widely used modern UAS. It has undergone significant development since its initial use but it has attracted criticism for its reliability and design[74]. The evolution of design solutions can be compared to biological evolution of agents within a population. This is often cited as the survival of the fittest but can perhaps be better understood in a way that is more relevant to engineering design if instead the phrase is parsed as “survival of the most informed”[69]. The most informed agents within a population are able to adapt to their changing environment in the most beneficial way. In order to adapt UAS to their environment all parties need to ensure they are well informed regarding their use, successes, and failures. FOQA within manned aerospace in order to learn from operational experience. This is especially difficult for unmanned aircraft as they attract reduced investment compared to manned aircraft for health management systems. In order for the resources available to do the most good a system is needed which can evaluate the current corpus to highlight those sub-system areas within UAS which would benefit most from health management.

It is worth pointing out that health information and management decisions are taken across a number of differently scales of time and complexity which makes the search within data available to us for meaning to inform the operations a data mining operation of epic proportions. This matches well as an extension to the FOQA philosophy which seeks to identify improvement and contextualise anomalies to better understand the use of the aircraft. The addition of AI to draw additional conclusions which could then be supported or not would need to be based on the information provided before and after each mission.

2.6 The cognitive gap

The process of handling UAS flight data for the purposes of IVHM is best understood through the concept of a human perception action loop. This was well expressed by Boyd as an OODA(Observe, Orient, Decide, Act) loop[69, 75]. See Figure 2.10 for an illustration of the loop. It was Boyd’s contention that every time new information entered the system the human would need to restart their loop at observe in order to process it and suggested that there would always be an advantage to being able to complete the loop faster than the competition as the

competition would need to restart their loop to react to the action taken by others. It has also been widely discussed as a concept for organisational control with the conclusion that an organisation with a fast OODA loop is able to adapt to their own mistakes more quickly and so gain advantage while those with a long loop time would struggle to adapt to changing situations. A manned aircraft can complete

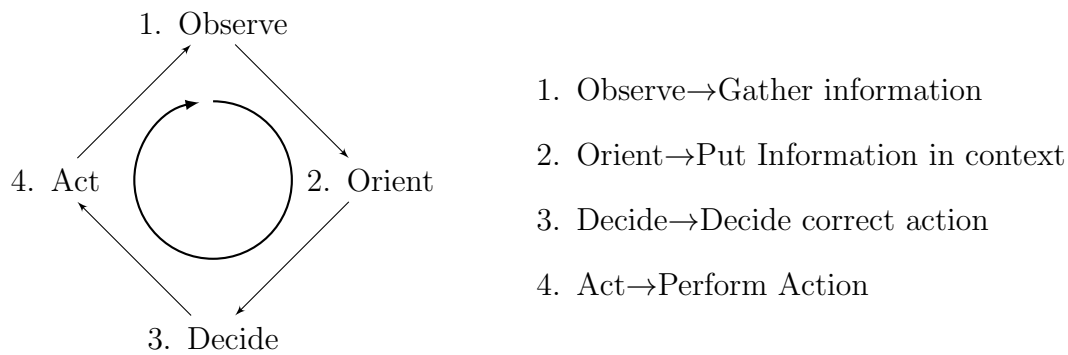


FIGURE 2.10: OODA Loop[75]

the whole loop through its pilot. However an unmanned aircraft can struggle with the Orient and Decide steps. Robots situational awareness and understanding of their environment can be very limited compared to a human as their sensors are much more restricted. In addition they require exhaustive programming to cover the uncommon situations they will encounter. The addition of sensors to improve situational awareness if not carefully done can result in a large increase in the system integration cost. The area of cognitive control is one that has been developed from an understanding of the ways in which machines need to improve to make them more like humans in their ability to handle complex situations and recognise when situations are outside the predefined task rules and new rules need to be created based on knowledge and understanding of the situation. The avionics software regulatory environment is such that knowledge based behaviour can be difficult to programme for an aircraft in such a way that it will be approved for use. It thus falls to the operator to perform the knowledge based tasks for the aircraft. It has been found however that humans are at their most error prone when they are performing knowledge based tasks[76]. Therefore the design of the system must ensure that all possible support is given to the decision maker such that the right decision can be made in a timely fashion. The exact nature of how this could be done is difficult to determine as no study has been performed to evaluate the relative merits of different HMI(Human Machine Interaction) approaches. It has been shown that some health management arrangements can negatively impact

the lifecycle costs[59]. Cognitive factors involving operators and HMI arrangements are certainly a factor that requires careful attention but it is not a simple matter to identify the precise causes[59]. There has been some work done to try to identify the cognitive challenges involved with the control of a UA from a GCS[72]. However this has not yet been extended to explore the cognitive factors within the health management of an aircraft or fleet of aircraft from a GCS. Cognitive factors in UAS health management are expected to be a serious issue which will need to be overcome if IVHM is to deliver on its promises.

2.7 CAHM (Cognitive Asset Health Management)

The operation of an aircraft is complex and consists of several levels. To understand the challenges correctly faced by the operators there is a need to understand how the operators process tasks. The cognitive framework proposed by Rasmussen[77] has been widely used in this context and is shown in Figure 2.11. This hierarchy consists of three levels of behaviour. The first is skill based behaviour which consists of learned skills in the linking of features from sensory input and some schema for action in response to signals. The second level is rules within which situations are recognised from the features formed within this mind and then associated to a specific learned rule. This then guides the schema used in response to the situation within the skill based behaviour layer. The highest layer is knowledge based behaviour within which situations are identified and with a much higher level abstraction goals are used in the decision making process and planning of new rules to improve response to known situations and to cope with novel situations. The cognitive framework as outlined by Rasmussen[77] (see Figure 2.11) encompasses the same stretch across time scales as Boyds OODA (Observe Orient Observe Act) loops[69, 75]. Aircraft operations and fleet management decisions(including health decisions) are taken across a number of different scales of time and complexity. This has particular application to the problem of health management to UAS as there are at least four classes of health impacting decision which may need to be made during any flight but which have different response time scales. See Figure 2.9 for clarification. The classes are listed here from the shortest time scale to the longest.

1. Machine Decisions → Issues requiring a decision before a human could react

2. Pilot Decisions → Issues which a human can decide on during the flight
3. Fleet management decisions → Issues which a human can decide on before the next flight
4. Strategic decisions → Issues which a corporation can decide

This is worth discussing as the system of operations and maintenance consists of multiple persons and assets with different training levels, experience, and skills. As an organisation however they must be able to actively manage their assets. Understanding their process and trying to ensure the system is formulated to work with them instead of against them is critical. As the time scales increase the ability to link aircraft behaviour to asset health and maintenance results becomes increasingly difficult. This is especially the case when operations management systems restrict the ability of personnel to clearly communicate knowledge and experience. Health management systems have failed on occasion due to the complexity of human-machine interaction[59, 78]. Some examples of some of the HMI problems are given below.

Human understanding challenges

Pilot and maintainer reports often use phrases and grammar which can be ambiguous to interpret and often does not answer questions an asset health management AI would need to know[79].

Pilot actions often make use of knowledge not integrated into the machine which can cause the machine to interpret input as an error[68].

Machine understanding challenges

Users often are forced to change their workflow to adapt to the machine rather than the machine being designed to suit the users[80].

Machine actions often make use of data not known to the pilot which can cause the pilot to interpret input as an error. Often referred to as automation surprise or cognitive mismatch[68].

In the past the need and the difficulty of assisting humans during critical tasks has been well discussed. The automation catch-22 is that the use of automation can assist the user but also degrade their competency by shielding them from normal system behaviour[81]. This has led to the call for cognitive systems which

might be better able to assist operators without distracting them or leading to a loss of situational awareness[82]. In this theme there have also been calls for human centered automation[21, 83]. These cognitive systems feature more of the abilities shown in Figure 2.11 from Rasmussens framework to assist in making a decision in an intelligent way with a method that is transparent to the user. Despite the increasing use of more automated systems and the productivity benefits expected to be gained it seems certain that in order to gain acceptance and trust any automation must always be human centred[21]. The cognitive concept explained by Rasmussen and used elsewhere to understand the human-machine interaction includes a rule-based behaviour layer. This is often explained through the metaphor of single dimension control strategies but this layer is also very applicable to the codification of incident response once the features exhibited by a particular incident have been recognised.

The population of a set of recognition agents connected to behaviour rules is a CBR(Case based reasoning) structure[84, 85]. This shows CBR approach conforms to some elements of a CAHM system but does not supply the necessary mechanisms to make it work effectively. CBR starts from the characterization of

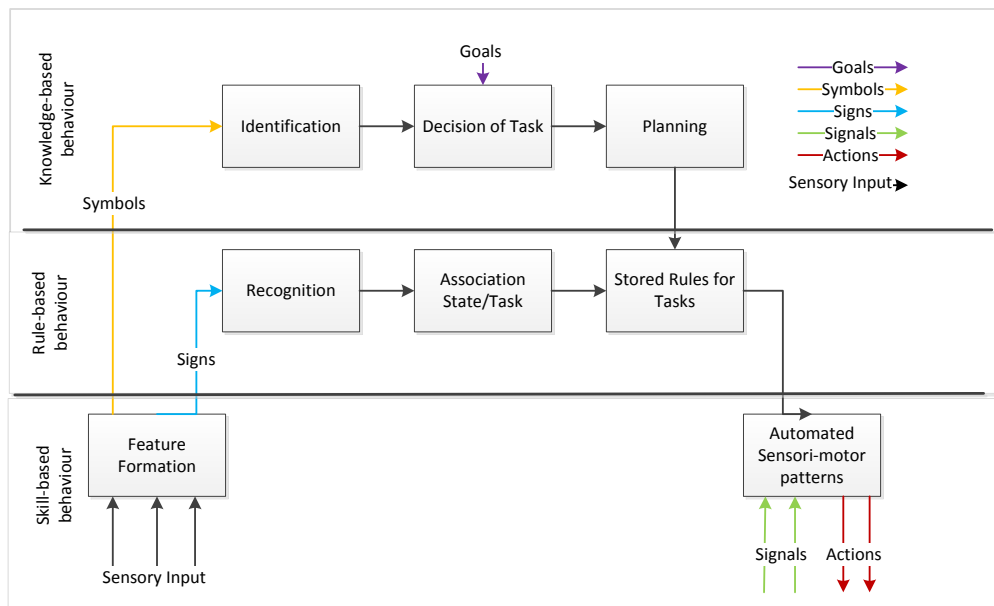


FIGURE 2.11: Rasmussen's Cognitive Task Hierachy[77]

a problem and searching the case base for a similar problem. (see Figure 3.1)

The closest match is retrieved. If it is suitable it is used with its solution. If it is not suitable it is revised to make it suitable so it can be used. This new case can then be added to the case base for future use. The population of the case base for CBR can be a large problem. This makes it less suitable for the UAS health management problem as there might be several thousand individual faults each of which may require their own case. These faults could also occur in combination to create a more complex situation which is more difficult for the operator to understand without sophisticated support. CBR is very straightforward for operators and maintainers to interrogate or add to the case base and be able to understand how they should be applied. CBR is then a good example of how a cognitive health management system might look. The use and management of aircraft, their operations, and health involve multiple assets, resources, and personnel. This complex environment can present some challenges to creating a system which can capture issues arising and formulate responses to them as rules or schema which can then be used. The identification of rules and schema to trigger maintenance or mission actions represents part of the normal management process. There is a regulatory question for health management as with many aircraft systems there is a requirement to meet regulatory scrutiny. This often restricts the type of system that can be used or designed into an aircraft. One strategy to improve operations while meeting regulatory scrutiny is to use off-board systems only while creating operation and maintenance rules. Once these rules are codified and proven through use the simpler and more effective ones could be deployed to the FMS(Flight Management System) of the aircraft. Computational cost is a critical consideration to this process but the use of on-board rules greatly reduces the time scale within which the FMS can respond to in-flight situations. Another example of a possible improvement which would meet regulatory scrutiny is the change from unscheduled maintenance to planned maintenance at a suitable time to reduce operational disruption. This represents early maintenance rather than a situation where a maintenance credit might be needed to maintain a system later than scheduled.

2.7.1 Features of a more cognitive Asset Health Management system

The system discussed here is described as a more cognitive system due to the attempt to identify the features which are needed to go a little further down the road towards a (CAHM) Cognitive Asset Health Management system. A fully cognitive AHM system would be very interesting but is not expected to be realized in the near future. The next steps are to go a little further using the current framework and see if this has a useful effect. This could then be built upon to further improve the ability of the system to operate in a way that is conducive to human operations managers.

The use of health management itself is not new as has been noted elsewhere but the formalization of health management systems for integration into assets and systems represents the difference between IVHM and legacy health management methods. IVHM has also specifically been discussed as an enabler for PSS(Product service systems) type operations. These operations are characterized by the supply of a product to a user as a service rather than the sale of a product. This creates a need for better health management as the failure of the service during use often involves hefty cost penalties to the service provider. The IVHM system has been drawn in Figure 2.7 to show the role of automation and integration.

IVHM system within CAHM

The difference between CAHM and IVHM is that IVHM may form a part of a CAHM system but only if the IVHM system operates in such a way as to enable the whole system to be more cognitive. It is important to note that there is more to IVHM than just health assessment(see Figure 2.7). A more cognitive AHM will need to ensure it has additional functionality integrated into itself. The ability to identify actions that can be taken to mitigate the fault during use is often discussed by those studying fault tolerant and self-healing methods for systems. If the fault can be successfully mitigated it can reduce the mishap rate and allow the asset to be safely recovered or at least allow the mishap to occur in an unoccupied area. It is also necessary for the IVHM functionality within the cognitive AHM to be able to assist with prediction of likely repair actions for a fault identified so that maintenance planning can be done ready for when the asset has returned. The coupling of confirmed faults with their identification criteria

also improves the ability of a CAHM to identify similar but different faults. The IVHM system within the CAHM must also be able to provide some ability to verify that a repair or mitigation action has had some effect and that the effect was positive. Confirming an asset as healthy and ready to return to service is a key part of the IVHM/CAHM cycle.

Features of a CAHM

In the features listed an UA will be given as the example asset. See Section 4.6 for a mapping of features to requirements, CBR, and the Artificial Immune System suggested as a CAHM.

- The system will be able to be populated with a priori rules and data known to be central to the control and health management of the asset. E.g. Vne (Velocity never exceed) triggered inspection, Engine max rpm, sensor limits to trigger sensor error flag, etc
- The system will monitor the sensors and systems of the asset. Learning from use the normal behaviour of the asset.
- The system will be able to identify when an operating condition has not been encountered before (novelty detection)
- The system will use machine learning to suggest rule sets appropriate to the health management of the asset
- The system will through application of rule sets to its usage history create estimates of the ROC(Receiver Operating Characteristic) curve of the new rules.
- The system will be able to share rules and operating information with other assets within the fleet.
- When a novel condition is identified or rule triggered the system will identify the nearest historical case in the fleet usage history to form a basis for comparison and proposed action.
- Consideration is to be given to the method by which rules and cases can be shared between assets of the same model, type, species, and genus. That is the degree to which learning can be generalized. E.g. RQ-1/MQ-1 Predator

Block 1 compared to RQ-1/MQ-1 Predator Block 1(same “model”) , RQ-1/MQ-1 Predator Block 2(same “type”), MQ-9 Reaper(same “species”), and U-2S Dragon Lady(same “genus”)

- A method to allocate which rules (if any) to be deployed within the asset and which should be used within the system support structure instead. Aircraft have a particular need to restrict the allocation of on-board weight and power. They are however by no means unique in this.
- The system will be considerably designed to work as a native part of the operators workflow rather than an annoyance.

System data sources

A short summary of some of the types of data which would form the sources from which a CAHM could draw to inform its decision making.

1. UA in flight telemetry → This data is typically less than 10 Hz and has been compressed for transmission to the GCS. Only a subset of the aircrafts sensors would typically be selected for transmission during flight. Some signals may be accessible by request of the pilot however.
2. UA QAR data (Quick Access Recorder) → This data would need to be retrieved from the aircraft between flights and can range in sample rate from 0.1 Hz to 10 Hz. The number of channels varies from aircraft to aircraft. The amount of data generated during a long flight can be substantial however[43]. A simple calculation for 20 channels of data sampled at 10Hz to a 16bit ADC (Analogue to Digital Converter) for a 7hr flight yields 164Mb. This is on the low end of the amount of data that should be expected. UAS have more digital control systems on board than an equivalent mission manned aircraft and are likely to log a larger amount of data. As a guide the more sophisticated civil aircraft are reported to generate on the order of 512 Gigabytes per flight[41].
3. Maintenance reports → During the maintenance of an aircraft reports are generated to specify what was investigated, what was replaced, and what was certified as correct.
4. Flight plans → A flight plan will typically contain the necessary information to conduct the flight. Any deviations could be the result of anomalous

behavior or simply due to a change in mission tasking during the flight. It is difficult to know for certain unless mission debriefs are also available to the system in a codified way.

System data processing

It is clear from the different types of data that any system to enable the through life support of UAS and the learning through life of their faults and operations will need to be able to handle potentially very high amounts of data and also to do analysis on this data. Some of this will be highly structured such as the flight telemetry and QAR data. However some such as the flight plans and maintenance reports will not be as highly ordered. In particular the use of linguistic frameworks to codify written reports which would otherwise contain information phrased in a way the system does not recognize or which uses idiom which again cannot be parsed. In order to reduce the cost of storing and processing data it is likely that a distributed file system will need to be used[86]. If the CAHM is structured to interrogate the system data within something like the mapReduce[87] framework this would enable parallelization and further reduction in cost for profiling the data and identifying novelty events[88]. An interesting parallel can be drawn to the world of Formula 1 where they make use of a distributed computing model to process the large volumes of telemetry they gather during a race in order to predict faults and identify competitive advantage[89].

2.7.2 General Requirements

The key to learning from flight data is structuring the data in a useful way. Most current methods work by applying some label to data relevant to the context the data was generated within. Hence if a mishap occurred within a flight it might be labelled as a mishap flight. This is straightforward and minimally intrusive. Further pre-processing might be required by some algorithms to identify where within the data the fault began. Other labels that could be applied to a flight include maintenance information such as the recent replacement of a part/LRU (Line Replaceable Unit). These and other relevant events could be included in the metadata of the flight to assist algorithms and humans in learning from the flight. It can be used to determine what about the flight was unusual, what differed from a normal flight. This will create a ruleset which can then be used to

create an advisory and also to guide future research for a more highly optimized diagnostic/prognostic method for the specific failure mode involved. Thus each time a fault occurs a case is generated for it and added to the case base. Items within the flight data are scrutinized to generate a rule set which would diagnose it. If faults share a common mode or subsystem this should be exhibited as a clustering behaviour.

- What elements would such a system need to have and why
- The identification of rules for action
- To require minimal additional work to fleet managers
- To tie in well to the way engineers analyze, collate, and make decisions from data.
- The logic of suggested actions to be clear from details and tied to the evidence used to come to the conclusion.

Despite higher mishap rates than some other aircraft the occurrence frequency of individual faults is low. This hinders the development of fault diagnostic methods through the large number of potential faults and the low volume of available case data.

2.7.3 Framework of data types

- Flight plan → Intended actions
 - How can this data be used?
 - Some of it is detailed and specific but the free text areas can be difficult to interpret consistently as they are subject to personal style and emphasis.
- Flight data → What happened
- Flight Report → what deviations were made to the plan and why?
- Aircraft log → Aircraft husbandry notes and actions
 - List of changes

- List of reported faults
- Could the presentation of operator written fault descriptions improve the system? If so then these could be attached to the case as a response. i.e. Given conditions in a case similar to this the following comments were made about the aircraft.
- Maintenance log → Fleet faults and repair actions
 - Maintenance Work Orders
 - Consumables and parts usage

2.8 Summary

Unmanned aircraft face some of the same challenges as manned aircraft in terms of management difficulty and complexity but also some additional difficulties in the area of aircrew situational awareness shown through analysis of mishaps.

All aircraft have fundamental characteristics which aircrew learn through experience and through remapping of their experience with other aircraft onto their current aircraft. Learning these characteristics for a remotely piloted aircraft places a high cognitive burden on the pilot. Cockpit(or Ground Control station) assistant systems[90] are highly desirable to extend the abilities of the aircrew but are difficult and expensive to develop. This problem is compounded in UAS due to the difficulty of obtaining good data for analysis. Learning the unique features of mishaps during the operational life of an UAS is a necessary feature of IVHM systems as part of an SMS. Due to the problems with obtaining data for study it was therefore decided to explore this gap using a simulation based approach to evaluate a family of novel machine learning techniques.

Chapter 3

Literature Review - Applicable Machine Learning

3.1 Introduction

In this section UAS will be examined and the methods by which information useful to the through life health management of a fleet of UAS can be obtained and processed to yield useful information will be discussed. In order to choose an appropriate method to learn from flight data in this Chapter the environment within which the methods would be applied and some of the different methods that could be used to learn within the environmental constraint will be examined and discussed.

- Case Based Reasoning Section 3.2
- Clustering Section 3.3.1
- Classification Section 3.3.2
- Domain Machine Learning Start of the Art Section 3.4

3.2 Case Based Reasoning

Case based reasoning as shown in Figure 3.1 is a generic method to store knowledge as a number of cases[84, 85]. Each unique situation type which may be encountered

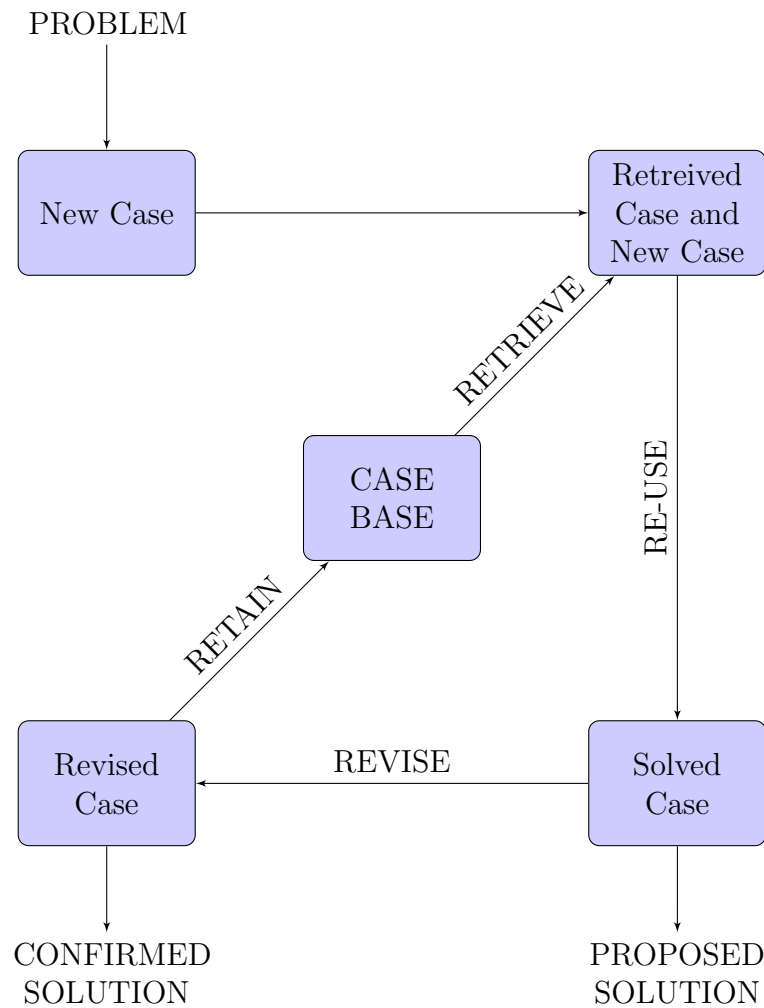


FIGURE 3.1: Case Based Reasoning[85]

is compared to known cases which have already been recorded with an action to be taken. If the case is not an exact match then the solution is either altered to suit or a new solution tried. In either case a new case is created for the case database. Thus over time a case base is built up for the domain which can reduce the time needed in future to respond to challenges. Where case based reasoning is used to capture learned solutions to problems one of the major challenges is identifying which of the cases stored in the system is most appropriate to the situation at hand. Often then methods used to recognize which case is most relevant are qualitative and difficult to use in a machine learning environment; however there are several quantitative comparisons metrics.

One method previously used represented the recognition criteria of the cases as a bit string and used the hamming method to evaluate which case was the most

appropriate for a problem[91]. The application of a CBR within an UAS operators SMS is visualised within Figure 3.2 which show how traditional FOQA methods as discussed in Section 2.3.2.2 can be combined with CBR to retain experience and build up a corpus of operationally relevant problems and solutions as cases.

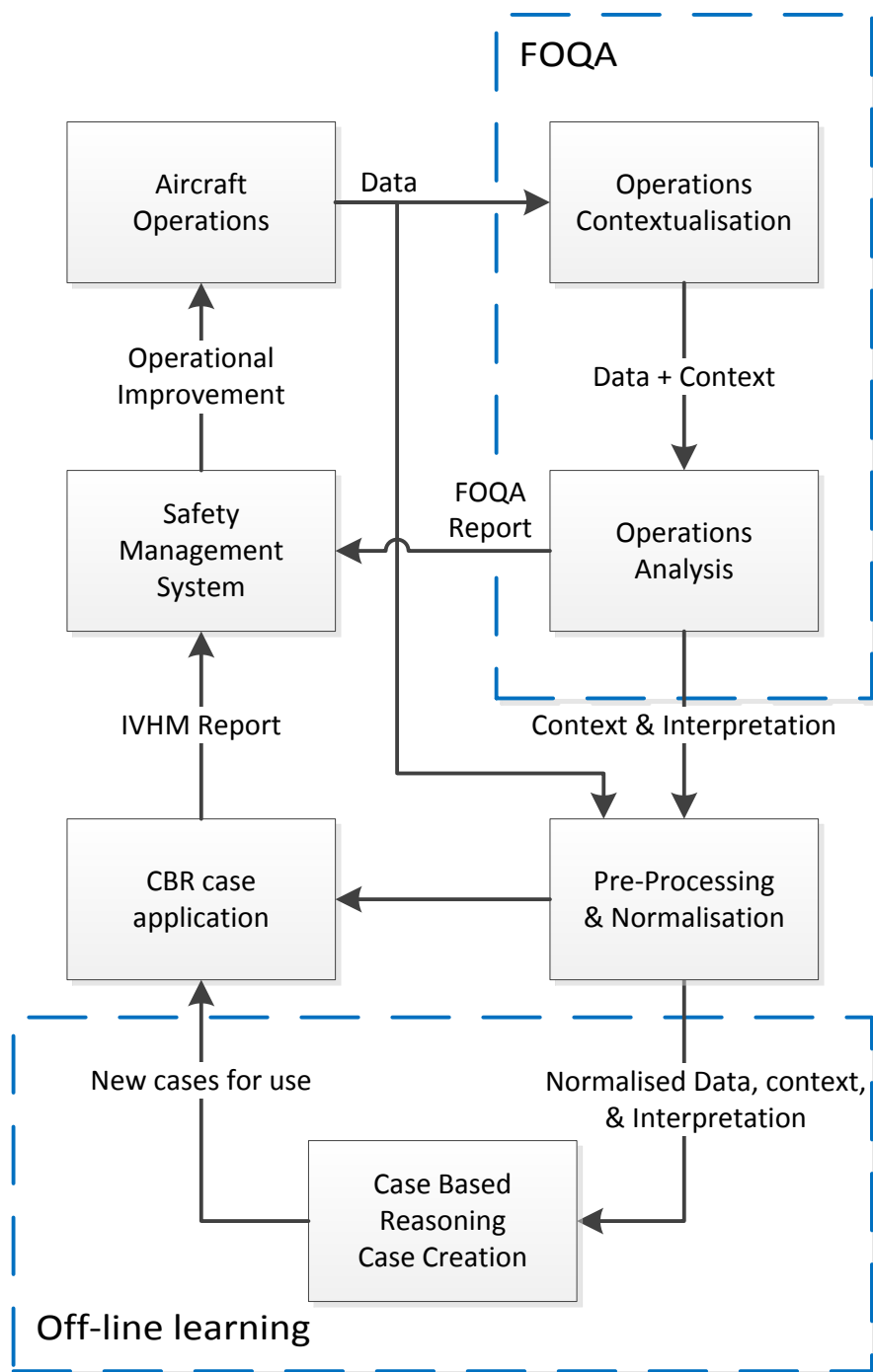


FIGURE 3.2: CBR-FOQA shown as part of the Safety Management System of an aircraft fleet.

3.3 Clustering and Classification

In the problem of exploring and using data to learn underlying features and lessons which can be applied to operations where it is believed or known that the data to be made of several sets there are two broad families of methods that can be applied.

Clustering

Clustering methods are used if that data which is being examined has sets the total number of which may be known but where each data point is not labelled and the intention is to reveal groupings within the data which can be mapped to specific concepts.

Classification

Classification methods are used if the data being examined has labels which are known and the intention is to create a method to classify future data into the most relevant category.

3.3.1 Clustering

Clustering is the group of methods which process some dataset with the aim of revealing any underlying structure within the dataset. The structure typically is in the form of groupings of similar entries from the dataset. No classification or conclusion is assigned to individual clusters. They are significant only for the specific metrics by which they are similar. The clusters found from data may be relevant to some operational purpose or underlying system paradigm but without expert analysis it can be difficult to understand their true significance. In order for more than one cluster to be found the data must be separable to some degree. Generally the methods are split into two groups.

Voronoi

Voronoi graphs are a very interesting method of dividing up sub regions of a multi dimensional space but can be very intensive to calculate^[92]. Their use is perhaps not strictly to cluster but to calculate appropriate boundaries between

regions within a feature space. They have been included them within clustering methods for consideration because their use can help to gain understanding of the regions within a feature space. Conceptually a Voronoi diagram is similar to several intersecting soap bubbles. Where two bubbles intersect a near straight intersection is created in the union between the two. The boundary between the two neighbouring bubbles(or regions) is placed at the lowest energy point of field equality between the two. In regions of equal weight(or strength) the boundary will be roughly central. Figure 3.3 shows a Voronoi method applied to define boundaries between several intersecting 2d disks in a plane.

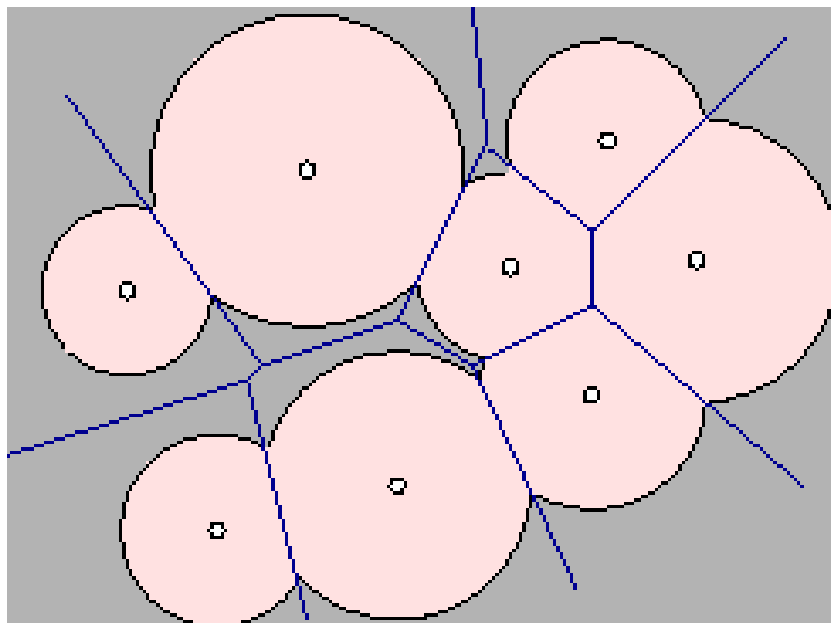


FIGURE 3.3: Figure shows a Voronoi method applied to define boundaries between several intersecting 2d disks in a plane. Reproduced from[93]

Self Organising Map

Self organizing maps is a type of neural network that attempts to learn the grouping within data using similarity.

3.3.2 Classification

Classification is the group of methods by which a classifier(or method) is developed to identify from the dataset the class to which each entry belongs. This is typically involves training on a subset of expert classified data in order to train the chosen

method and to evaluate its performance. This group of methods do not necessarily require an understanding of what makes the elements of one class different from another but to be successful they classes must have some consistent differences.

3.3.2.1 Supervised

Algorithms are supervised during a training period and after testing if successful can then be used for its intended task.

Neural Networks

Neural Networks are a family of classifiers inspired by the workings of the human neural systems[94]. They are based on the concept that this complex system is made up of many interacting neurons. Each neuron takes in a number of weighted inputs and has an activation function which determines, when, based on it's inputs, it will trigger it's own output. Figure 3.4 shows a single neuron with inputs x and output y . The most common depiction of this system is as each neuron being a linear classifier dividing the feature space of n dimensions(for n weighted inputs) into an active output side and an inactive output side. The combination of many neurons allows for more complex hyper plane classification. It should be noted however that this type of classifier requires the sets it is used with to be linearly separable for full accuracy to be possible. Different functions can be used for each

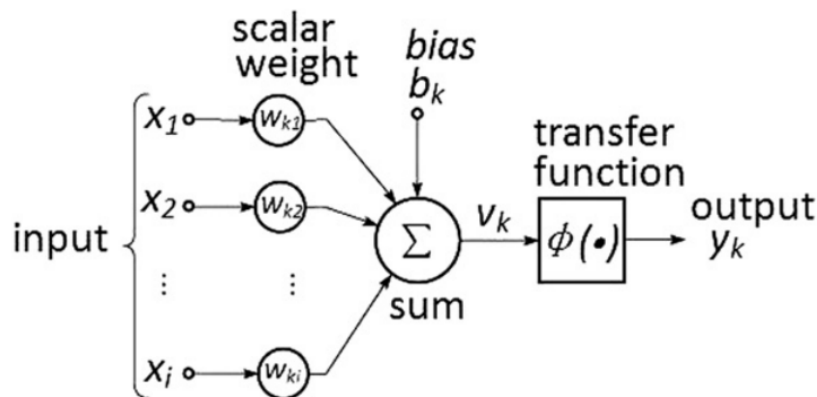


FIGURE 3.4: Figure shows a single neuron of a neural network from Liu et al[95].

neuron to change the behaviour of the network and it's ability to classify sets. One of these is the radial basis function. which has the advantage of being able to

cope with discontinuous sets but is less useful at accurately classifying new points in gaps between the neurons[96]. The training of these networks is achieved by evaluating their output and then calculating the contribution from each neuron so that their weights and offsets can be adjusted. This type of training is termed back propagation as the error signal is propagated back through the network to correct more strongly the neurons which more strongly contributed to an incorrect classification[97].

Bayesian Reasoner

Bayesian reasoning is a supervised learning technique which attempts to create a reasoner that can base its classification ability on the probability that a given input belongs to a specific class. This can be used as a general supervised learning technique[98] and has also been used successfully in the aerospace domain[99–101].

Genetic Algorithm

Genetic Algorithms (also known as Evolutionary Algorithms) use the paradigm of evolution to try to create a solution to a problem. Often the problem is the detection of faults. The algorithm represents potential solutions as individuals within a population. Each solution is expressed as a “genetic code” which contains the required elements for the individual to represent a solution. Each individual in the population is assessed using a fitness metric which is constructed based on the problem and given a score. High scoring individuals are referred to as high fitness and different strategies are used over multiple generations of the population to explore the space around high fitness individuals using mutation operators and gene splicing to “breed” the individuals in the hope of producing high fitness offspring and over time not only finding higher fitness individuals but also increasing the average fitness of the population[102].

3.3.2.2 Un-supervised

K-nearest neighbour

K-nearest neighbour Algorithms attempt to classify the correct class of a new data point based on the class of k nearest neighbour points within the feature space. This technique relies on having some data points classified before the Algorithm

starts and then as new points are added they are classified based on their k nearest neighbours. The number of neighbours used k can be any positive integer.

3.4 UAS related Machine Learning State of the Art

3.4.1 Machine Learning for adaptive control of UAS

A class of algorithms have been developed which have as their aim the enhancement of the vehicle flight so as to maintain mission functionality despite the presence of some damage unanticipated by the designers. A number of examples can be shown. One of the most impressive is by Jourdan et al[58]. This demonstrated the use of a neural network to adapt flight control gains to maintain effective flight control where possible and safely recover the vehicle. This was especially impressive given that one of the fault scenarios involved the ejection of 80% of one wing.

There are some difficulties with using a neural network(or other complex stochastic method) in a real aircraft as an active part of the control system. This is due to a difficulty in rigorously proving the algorithm for all conditions in order to meet regulatory requirements for manned aircraft[103]. Unmanned aircraft face many of the same hurdles as most CAA's (Civil Aviation Authority) require unmanned aircraft systems to be equivalent to manned systems.

3.4.2 Fault diagnosis and prognosis

There are many different ways to diagnose and predict faults and they can be split into three groups.

1. **Physics based** → This group of methods attempt to use a direct mathematical model of the behaviour of the system fault to relate the output of some sensor to the status of the fault. Typically preferred for use by the OEM(Original Equipment Manufacturer) as the physics of the device are known to the designers while they may have difficulty obtaining operational data.

2. **Process history based**(Data based) → This group of methods attempt to use data gathered either from historical use or experimental process. Once sufficient cases have been generated many different statistical and artificial intelligence(machine learning) methods can be tried to create a method to diagnose or predict faults. This method is used more frequently by operators as the OEM is unlikely to explain the physics of a part and so they will need to treat the part as a black box and use their operational data instead to help them create useful methods.
3. **Hybrid** methods → These methods attempt to combine 1 and 2. For further discussion see work by Eker[104]

3.4.2.1 Diagnosis

Health Assessment (HA): diagnoses any faults and rates the current health of the equipment or process, considering all state information.
(Walther, B page 8[105])

There is a selection of machine learning approaches to detecting and diagnosing faults. Typically this will involve a deep knowledge regarding the fault and its presentation within a specific platform. A supervised learning method is then used to create a machine learning solution which can detect and identify a specific fault or faults. This is sometimes also called FDIE (Fault Detection, Identification, and Evaluation). The necessary information needed to develop a fault diagnosis system is the details about the faults to be found. Some sensor data, either historical or experimental are expected to be relevant to the selected faults. This can then be used to train and evaluate different diagnostic methods. Neural networks have been used in a large number of areas to create diagnostic classifiers. They have been applied to identifying operational problems from misfires in a diesel engine[95], to detecting system faults in turbofan engines[106], and a wide variety of other applications. One drawback of neural networks is the difficulty in predicting their output across the whole of the feature space due to their inherent complexity. This has limited the ability of neural networks to be applied in safety critical applications.

3.4.2.2 Prognosis

Prognostics Assessment (PA): determines future health states and failure modes based on the current health assessment and projected usage loads on the equipment and/or process, as well as remaining useful life. (Walther, B page 8[105])

The prediction of faults before they occur is a difficult field but a number of approaches have been taken. As with fault diagnosis to develop a fault prognosis system details about the faults are required. Some sensor data either historical or experimental expected to be relevant to the selected faults. This can then be used to train and evaluate different prognostic methods. In addition to data during a failure prognostics systems must be trained using data leading up to failure to enable the assessment of the predictive power of the system. [107]

3.4.3 Machine Learning for health management of UAS

Health management machine learning methods are rather thinner on the ground. The area is not precisely defined but an understanding that health management system would include features for fault detection (diagnosis), fault prediction (prognosis), learning new faults, and integrating maintenance knowledge and system usage at some level. This has been discussed from the perspective of a cognitive health management system in section 2.7. The IVHM cycle shown in Figure 2.7 references well the requirement for a whole cycle system and machine learning can be applied to all sections of this cycle in an integrated way to realise a solution for health management rather than just for sub sections of the IVHM cycle. The RAF Eurofighter uses an integrated logistics and maintenance system which goes some of the way towards integrated health management and it has been suggested that the ALIS(autonomic logistics information system) system intended for the Joint Strike Fighter[108] will come even closer to this goal. However since limited information is available for these programs and their effectiveness it does not present a useful point of reference.

3.5 Summary

Process history based methods are more appropriate for through life learning than physics based methods due to the need for a system that integrates new information into the health management system. The two main groups are those which are intended to distinguish between separate situations which they have been trained on based on a recognised order (Classification) and those which are intended to help expose some underlying order (Clustering). Clustering approaches could be useful but do not typically result in a running system that can alert users to hazards so a classifier which perhaps shares some features of a clustering approach would be most useful as it would give the benefits of a classifier while also revealing some structure of the feature space.

Chapter 4

Machine Learning Method Comparison - AIS

4.1 Introduction

The artificial immune system will be introduced in detail and compared to other approaches for analysis of aircraft flight data for the purposes of through life health management. Due to the ability of AIS to learn with less structure than other machine learning methods they are very interesting for the health management of UAS.

- In Section 4.2.1 the human immune system is discussed with reference to its ability to learn from damage without being taught and its application to asset health management
- In Section 4.2.2 the history of the development of AIS will be examined and best practices drawn out.
- In section 4.3 the compatibility between CBR(case based reasoning) and AIS is discussed.
- CAHM, CBR, and AIS-CBR approaches are compared in Section 4.6

4.2 AIS - Artificial Immune System

...good maintenance requires, as we have outlined for the immune system, a program of three parts:

1. Recognition → you have to see what is right and what is wrong
2. Cognition → you have to interpret signs, evaluate results, and make decisions
3. Action → you have to actually do the job

(Quote from Irun R. Cohen[109])

Unmanned Aircraft Systems consist of several system elements and produce a large amount of data as discussed in section 2.3.2.1. The development and analysis of UA flight data is also discussed in section 2.3.2. Some of the difficulties of operating UAS and some of the problems unique to UAS were discussed in Chapter 2. Artificial immune systems have developed since their initial appearance in Forrest et al[110]. AIS have been applied in numerous areas where their unique characteristics can offer useful abilities vs other machine learning methods. Some examples of the applications are shown in Table 4.1. The original embodiment of artificial immune systems in Forrest et al was based upon the negative selection algorithm and the self/non-self paradigm. In this context self is all the normal things a system may do and non-self is all the unexpected things. The paradigm of self/non-self is a biological one explained further later in this chapter but is essentially about the immune systems ability to distinguish between the body's tissue and other biological items that might be encountered. Non-self items are not automatically classed as harmful but the understanding is that anything harmful would not look the same to the system as normal tissue and so the search for useful detectors(antibodies) is restricted to those that would alert against items that are not within the self description. A detector with some number of features n is compared using one of several available distance(affinity) measures to samples of self and if the detector alarmed it was removed from the population. This reduced the false alarm rate but is difficult to improve over time as the understanding of self and non-self would be expected to improve over time for a system which is under supervision. One of the issues with self/non-self discrimination in a learning situation is the existence of unknown self and unknown non-self. Data is constantly being generated during use and must be categorized into normal and potential anomalous behaviour[109]. It can also be seen from Figure 4.1 that AIS have been recognised by Lee et al[111] as offering a useful option to the

area of health management of UA as their problems cannot be fully defined and will only be discovered through life. Some of the applications shown in Table 4.1 are discussed later in this chapter but there are some fascinating use cases where systems are able to take advantage of the AIS ability to learn in an unsupervised or semi-supervised manner. One interesting example highlighted is in the field of virus detection[112] and computer security for intrusion detection[113] which is an application with understandable biological parallels and where adversaries in the same way as viruses adapt themselves to try to look as much like normal behaviour as possible. It is helpful that faults in an aircraft context do not adapt in a similar way to make it less likely that they will be detected. In time however we may encounter the same problem the human immune system has of diseases and viruses being selected in an evolutionary manner so that only the hardest to detect ones persist in the system.

TABLE 4.1: Application areas of AIS from Hart and Timmis[114]

Major	Minor
Clustering/classification	Bio-informatics
Anomaly Detection	Image Processing
Computer Security	Control
Numeric function optimisation	Robotics
Combinatoric optimisation	Virus detection
Learning	Web mining

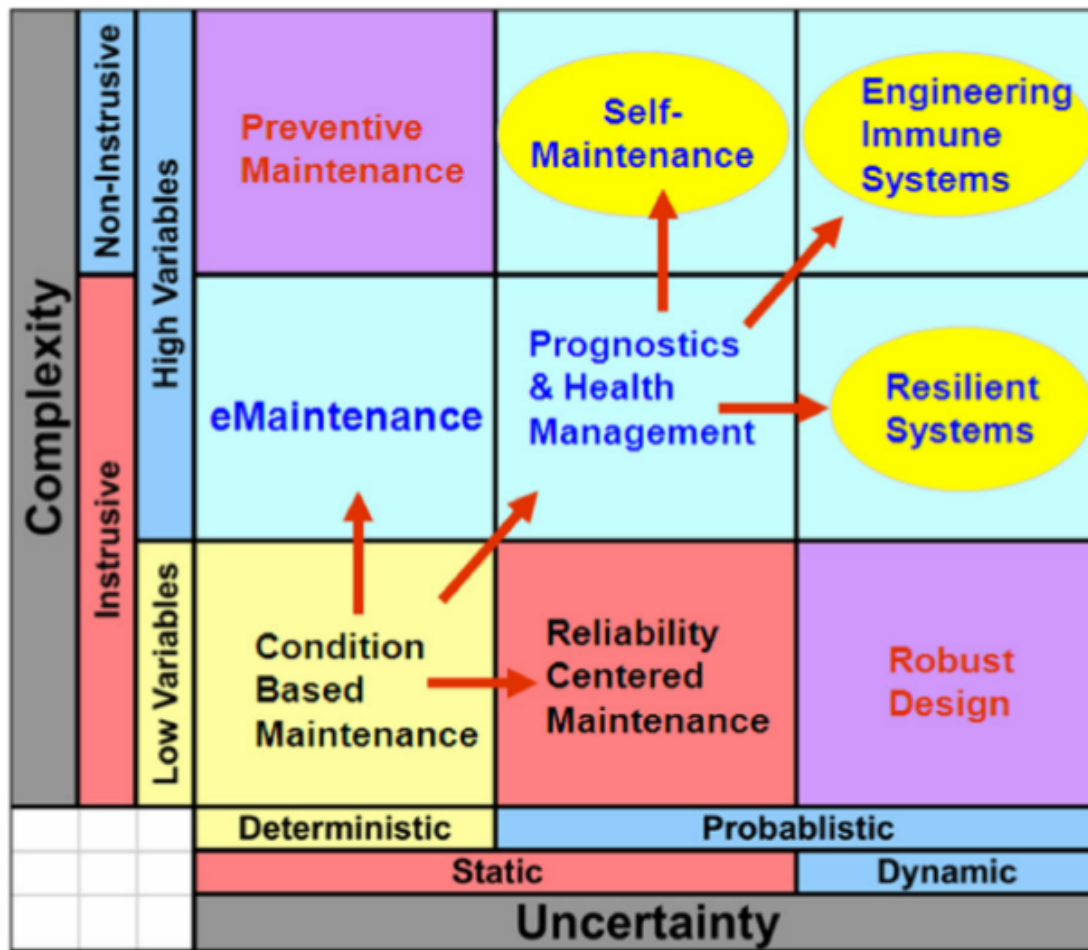


FIGURE 4.1: Maintenance Transformation Map from Lee[111]

4.2.1 Human Immune System

... the main properties of an engineering immune system, in addition to robustness and resilience, are the ability to detect anomalies, adaptability and autonomy. (Quote from Lee)[111]

4.2.1.1 Key concepts

The human immune system is a remarkable system of multiple interacting autonomous agents.

Abilities of the human immune system

- Identifies new threats

- Remembers old threats
- Dynamically adjusts priorities to threats
- Has a safety mechanism to avoid attacking itself
- Is in two parts
 - B-Cells Identifies Threats
 - T-Cells Removes Threats

Role of B-cells

an antigen is any molecule which can be recognized by an antigen receptor. The part of an antigen that actually fits into the receptor is called the epitope of the antigen (Quote from Cohen[109])

Immune systems consist of a set of B-Cells as shown in Figure 4.2 which are created through pseudo random genetic variation. B-cells match themselves against the proteins found on the outsides of living cells. If a match is found this causes the B-cell to bind to the cell and release a chemical which attracts a T-Cell to come and absorb the matched cell and all the B-cells which have attached to it. Two B-cell antibodies are shown binding and alerting (red lines) to a yellow bacteria in Figure 4.2. This is the body's way of reacting to foreign bodies such as microbes. Many different B-Cells may bind against different sites on the antigen. This gives the system the ability to determine different features are the distinctive elements of an antigen and use the best ones. It also gives the ability to not use those matching characteristics that might cause the system to alert on protein sites in the human body rather than antigens. The different elements of the binding site are called epitopes. These are the individual binds which when combined create a successful match between antibody and antigen.

Role of T-cells

T-cells are the type of lymphocytes which upon the alert of the B-Cells will come and consume the antigen which the B-cell has alerted against as well as the B-cell. This removes the antigen and the alarm signal from the environment. In an

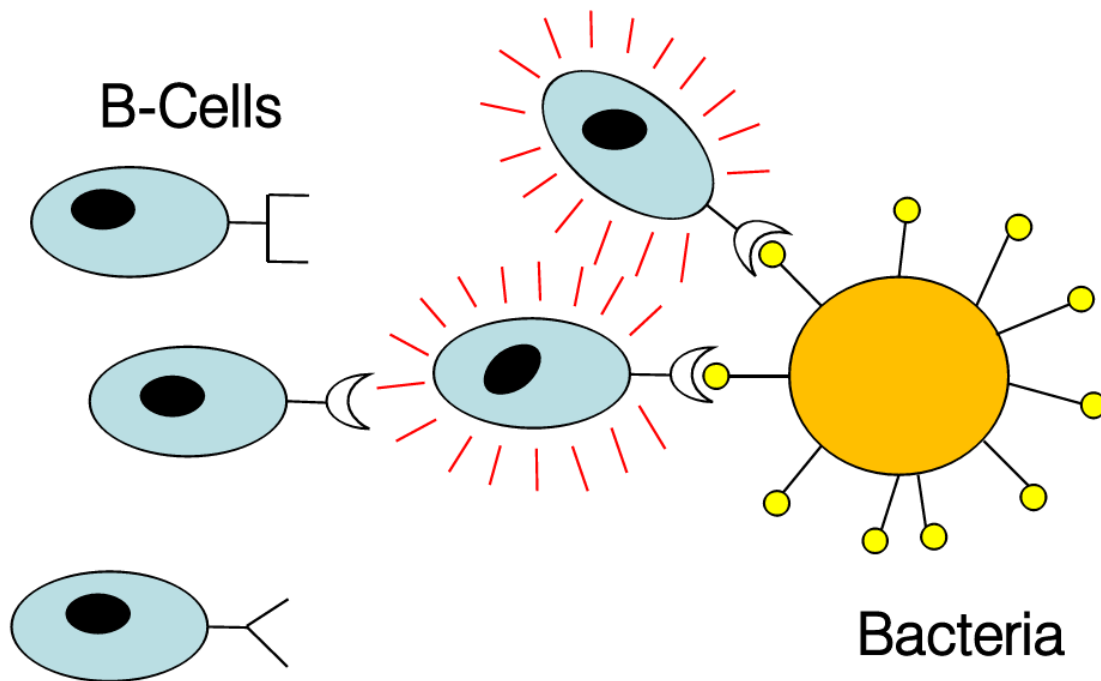


FIGURE 4.2: B-Cells and Bacteria

artificial Immune system this function is performed by whichever system(human or machine) which takes the action indicated by the information associated to the antibody which has alarmed.

Negative selection

As the system can learn to respond to new threats and all its potential future alert conditions cannot be mapped beforehand the immune system takes steps to ensure that any B-cells produced do not false alarm. The false alarm in this context would be by attaching themselves(alerting) to normal and otherwise healthy human cells. Reducing this false alarm problem is believed to be performed via a process called negative selection where the immune system compares the alarm patterns for each new B-cell against those patterns which have been identified as present in normal healthy cells. All those which would alarm against a healthy cell are destroyed before entering the bloodstream.

Aging

After the cell has been created it ages and once it has reached a certain age it dies. This limits the total population of cells but new cells are continually produced through clonal maturation. This keeps the population from becoming too

large, retains learned immunities and means the system can readily adapt to new threats. Artificial immune systems have developed since their initial appearance in computing[110] The immune system exhibits properties associated with a cognitive system such as threat identification, neutralization, memory of threats, and recognition of new threats[109]. AIS have been identified as a useful type of AI and some of the reasons for this are discussed. The original embodiment of AIS was based upon the negative selection Algorithm and the self/non-self paradigm. A detector with some number of features n was compared using one of several available distance(affinity) measures to samples of self and if the detector alarmed it was removed from the population. This reduced the false alarm rate but is difficult to improve over time as the understanding of self and non-self would be expected to improve over time for a system which is under supervision. Data is constantly being generated during use and must be categorized into normal and potential anomalous behaviour. However not all possible coordinates within the feature space(for number of features n the feature space has n dimensions) can be known or classified as self/non-self a priori. This creates the issue of unknown self and unknown non-self showing up as novel within the system and needing to be classified as self/non-self[65]. Immune systems consist of a set of B-Cells which are created through pseudo random genetic variation. B-cells match themselves against the proteins found on the outsides of living cells. If it matches this causes the B-cell to bind to the antigen and release a chemical which attracts a T-Cell to come and absorb the matched cell and all the B-cells which have attached to it. This is the body's way of reacting to foreign bodies such as microbes. Multiple epitopes may exist within the immune system for the same antigen. This maintains coverage and gives some protection to mutation in the antigen to a new form. See Figure 4.2

Clonal Maturation

When a B-cell is successful in its role its production is repeated through clonal maturation. This creates replicas of the B-Cell which alarmed correctly and also creates subtly different mutations of the B-cell which cover small differences in the epitope and provide protection from antigen mutation. This allows for the system to both respond to known threats(antigens) via the identified epitopes and also respond to threats not previously identified but which share some similarity to previously identified threats.

Danger Theory

When a cell is under stress it can emit a danger signal which lowers the level of affinity required for a B-Cell to bind to a suspect cell. This reduces the number of false alarms in other areas where there is no stress but allows the system to deal more quickly with an issue when the antigen is not a precise match for the current B-Cell. The B-cells within the AIS have for some time formed the basis for detection and action for AIs using this paradigm. The use of T-cells and bone marrow metaphors has not gained widespread use but may in time. The AIS is best described as a cartoon of the human immune system. It is however a cartoon with useful properties. The identification of antigens within data depends on their correlation with some measure which has been identified as useful for classifying the end results as normal or anomalous.

4.2.2 AIS Development

The development of the different types of AIS is plotted out and discussed by Greensmith, Whitbrook, and Aikelin[115]. Their arrangement has been redrawn and is shown in Figure 4.3.

The four major paradigms of AIS

- NSA(Negative Selection Algorithm)[116]
- CSA(Clonal selection Algorithm) [117] See Algorithm 2
- Idiotypic network models[114]
- Dendritic cell Algorithm[118] See Algorithm 1

The NSA (negative selection Algorithm) as discussed by Dasgupta et al[116] has the following steps

1. The system initialises certain amount of random detectors, named “naive detectors”
2. The system generates a self set from the training data that only contains normal instances

3. The naive detectors are compared to the self set, to produce a population of detectors that only react to intruders, termed “mature detectors”
4. The data instances in the testing data are compared with each mature detector, and if any detector reacts to the incoming data instance, an alarm is triggered.

This negative selection algorithm thus replicates one of the features of the human immune system in an attempt to focus the algorithm detection on novelty in the system and reduce false alarms. The development of this type of self/non-self discrimination using negative selection can be seen to have been developed as shown in Figure 4.3 into methods for fault tolerant control and into methods for simulation of the biological immune system. AIS have found a place as both a tool to test our understanding of the immune system and predict responses and also as a tool that can be applied to various problems in other domains. The AIS has been developed to include further more detailed concepts from the human immune system such as danger theory[119] which is embodied within the dendritic cell algorithm[120](See Algorithm 1) and the idea that successful antibodies are given accelerated cloning through the clonal maturation process[117](See Algorithm 2) to ensure any similar threats are dealt with immediately and also to explore the feature space around the successful solution. It can be seen in Algorithm 1 that it is attempting to add context sensitivity to the detectors so they can be suitable to both a normal situation and also to a recognised situation of elevated threat. In the body this used mechanisms such as the inflammation response to signal to antibodies that there is a problem in this specific local area without causing an immune reaction at a site distant to the actual problem. It is in fact this mechanisms that fails in some immune system disorders as the signal fails to operate correctly and a small stimulation of the system can result in the immune system reacting against other areas within the body which are actually healthy. The dendritic algorithm does this via danger and PAMP(Pathogen-Associated Molecular Pattern) signal. The clonal selection algorithms in Dasgupta are based on Burnetts theory of clonal selection[121]. See Immunology: A short course[155] for more information. The system initialises certain amount of random solutions to a given problem. These solutions are then being exposed to training in each generation, the best solutions are selected based on an affinity measure, to reproduce with multiple clones. Current solutions then undergo a mutation process with high frequency, which is proportional to the affinity measure. Optimal solutions are selected in the current

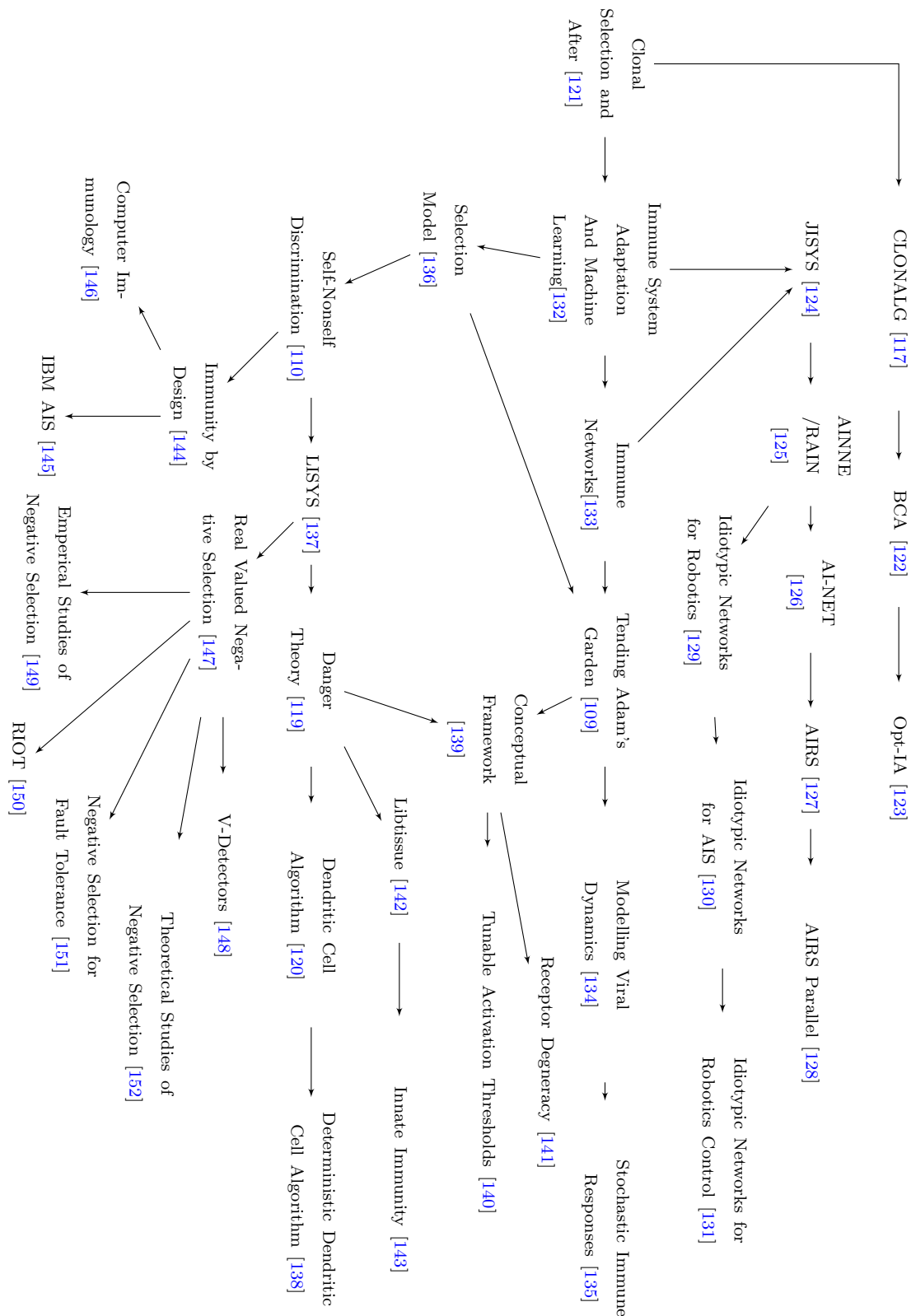


FIGURE 4.3: AIS Development redrawn from Greensmith, Whitbrook, and Aikelin[115]

input : S = set of data *items* to be labelled safe or dangerous
output: M = set of data *items* labelled safe or dangerous

```

begin
  Create an initial population of dendritic cells ( $DC's$ ),  $D$ ;
  Create a set to contain migrated  $DC's$ ,  $M$ ;
  forall data items in  $S$  do
    Create a set of  $DC's$  randomly selected from  $D$  ,  $P$ ;
    forall  $DC's$  in  $P$  do
      Add data item to  $DC's$  collected list;
      Update danger, PAMP and safe signal concentrations;
      Update concentrations of output cytokines;
      Migrate the  $DC$  from  $D$  to  $M$  and create a new  $DC$  in  $D$  if
        concentration of co-stimulatory molecules is above a threshold;
    end
  end
  forall  $DC's$  in  $M$  do
    Set  $DC$  to be semi-mature if output concentration of semi-mature
      cytokines is greater than mature cytokines, otherwise set as mature ;
  end
  forall data items in  $S$  do
    Calculate number of times data item is presented by a mature  $DC$ 
      and a semi-mature  $DC$ ;
    Label data item a safe if presented by more than semi-mature  $DC's$ 
      than mature  $DC's$ , otherwise label as dangerous;
    Add data item to labelled set  $M$  ;
  end
end

```

Algorithm 1: Dendritic Cell Algorithm from Brownlee[153]

population, to form memory cells. The steps above keep repeating until a termination point is reached The Dendritic Cell Algorithm for Intrusion Detection[113] is shown in Algorithm 1 The Dendritic cell Algorithm with its 3 stage danger concept is not suitable for flight data analysis as it requires a health measure to generate the danger signal. If there is already a context sensitive health measure why would the system need fault diagnostics?

DCA Signal Types

- Danger
- PAMP (Pathogen-Associated Molecular Pattern)
- Safe

input : S = set of patterns to be recognised, n the number of worst elements to select for removal

output: M = set of memory detectors capable of classifying unseen patterns

begin

- Create** an initial random set of antibodies, A ;
- forall** *patterns in S* **do**
 - Determine** the affinity with each antibody in A ;
 - Generate** clones of a subset of the antibodies in A with the highest affinity. The number of clones for an antibody is proportional to its affinity;
 - Mutate** attributes of these clones to the set A , and place a copy of the highest affinity antibodies in A into the memory set, M ;
 - Replace** the n lowest affinity antibodies in A with new randomly generated antibodies;
- end**

end

Algorithm 2: CLONALG AIS Algorithm as described by[117] and shown in Figure 4.4

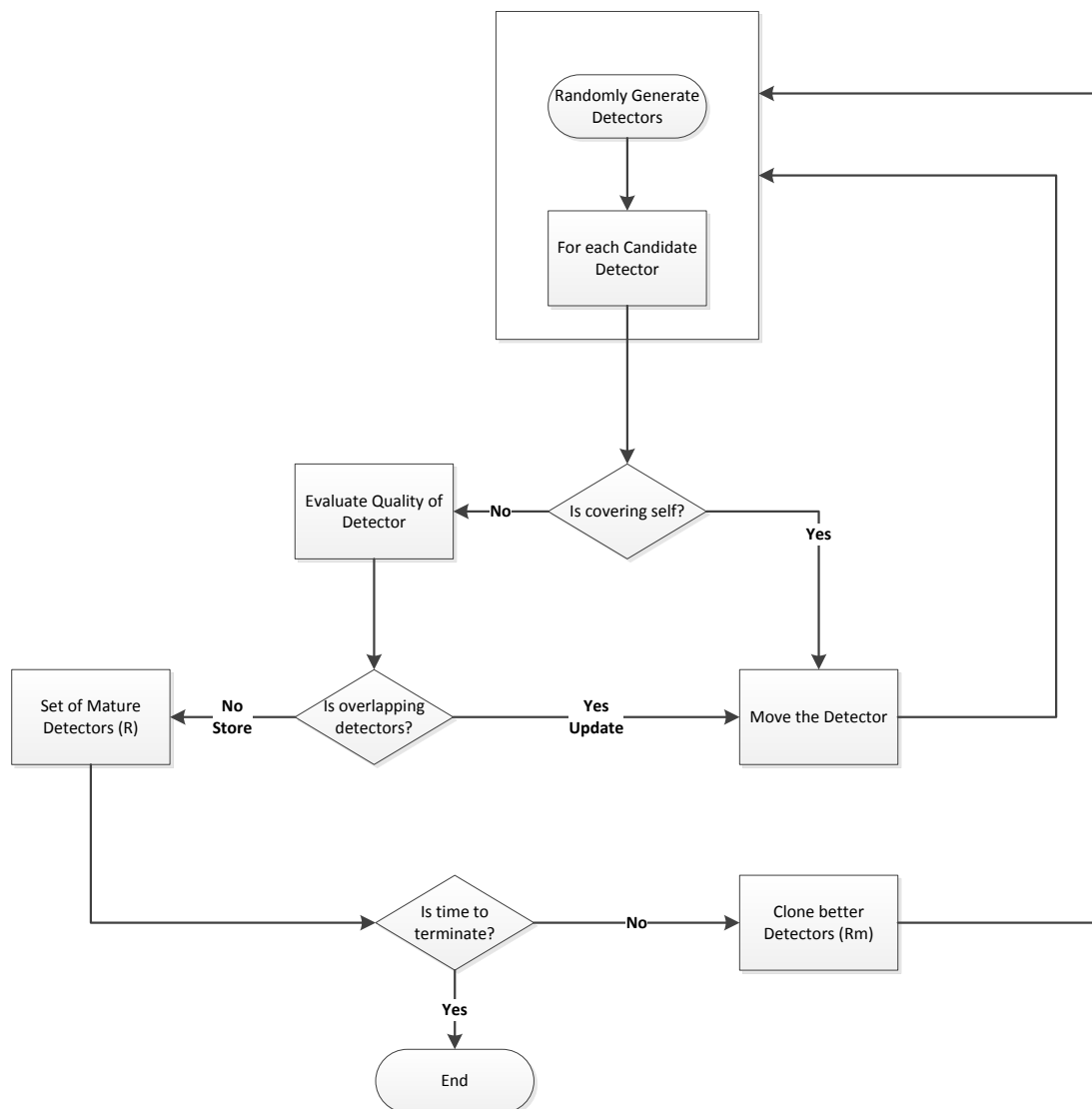


FIGURE 4.4: Immune System Algorithm for generating variable size detectors redrawn from Dasgupta et al [154] Quality is measure of how well antibody detects antigen. e.g. Fitness

4.2.3 AIS Discussion

4.2.3.1 Comparison to cognitive control

Cognitive control shares some similarities with AIS. Its skill based behaviour layer could be compared to the normal behaviour of the body. Its rule based behaviour can be compared to the rule based classification of a T-cell. The T-cell attaches to a cell when it conforms to certain rules after all. The knowledge based layer then is the area for creating new rule based behaviours for the T-cells (to create new types of T-cell). Elements of the AIS not covered by Rasmussens[77] model are the clonal maturation and negative selection. They could well occur within the planning section however. It must also be remembered that Rasmussens model was explained using a single variable control example so to expand it to multiple variable problems is possible but the designer must be careful not to over extend the analogy of the model. It is only intended as a possible way of ordering thought in the design of a system. The challenge of AIS for aircraft health management is balancing the long term health detection aspects against the human control and review of the system. A system that cannot pass regulatory scrutiny is not a useful system. Humans could use knowledge based behaviour to analyse potential alert scenarios proposed by the AIS. The system would need to perform negative selection on those scenarios which match previously identified health scenarios. The humans in the system can then also provide a second tier of negative selection to the new proposed scenarios formed through clonal maturation of previously identified alert scenarios. The slight variation caused by clonal maturation could help the system cope with aircraft configuration changes. In the initial stages of use the system would alert often but would gradually learn. Examining how the system could be structured around the UAS and GCS is a very interesting problem. The knowledge based behaviour would in all probability be performed by a flight engineer or fleet engineer and then the identified scenarios split into two tiers. Those which could be identified from aircraft telemetry in the GCS and those which need to be on-board to be used.

4.2.3.2 Advantages

The use of artificial immune systems for fault diagnosis has a number of advantages. The antibodies make the system inherently parallel as multiple antibodies

can be created that alert on different parts of the same fault. This mapping out of the fault non-self space is analogous to rule based recognition agents except antibodies can be created automatically and do not require manual intervention. Each antibody can be tied to a specific mishap case. Each antibody goes through negative selection to reduce the potential for it to alarm against normal flight occurrences. This also enables the antibodies to be created without the need to arduously segment the training mishaps flight data into fault and non-fault portions. Another advantage of using an artificial immune system is that as new mishaps are learned the antibodies for old ones can still retain the information and the whole system improves rather than having to reconfigure the whole system for a new solution as you would with a neural network.

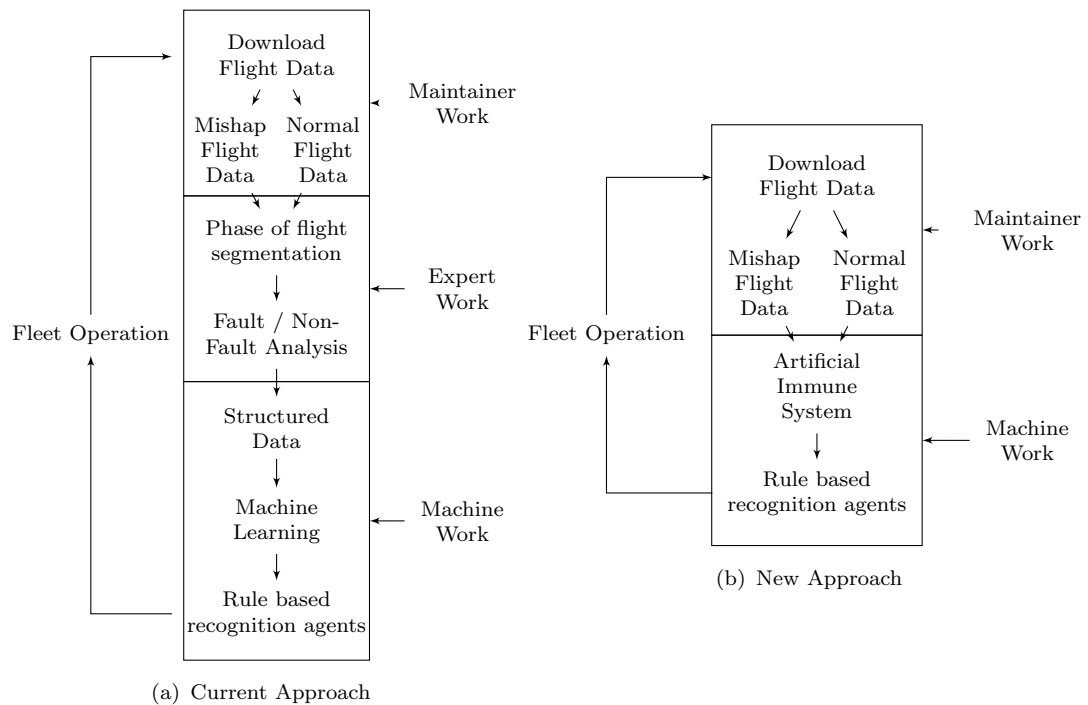


FIGURE 4.5: AIS vs Supervised Machine Learning showing both the current approach to learning from flight data in sub-figure 4.5(a) and the proposed new approach in sub-figure 4.5(b)

FOQA comparison

FOQA makes use of heuristics to compare aircraft flights actions to historical aircraft flight actions so that the actions taken can be evaluated in context with previous examples[156]. This relies on the use of segmentation of the flight into phases to aid comparison between takeoffs, landings, climb, descent, and cruise

portions of flight[157]. This enables the comparison of very different flights based on the portions of the flight which are comparable. The use of segmentation is very useful and should be strongly considered. The behaviour of the aircraft and its intended behaviour is very strongly affected by the phase of flight and some actions can only be understood through knowledge of which flight phase the aircraft is in at the time. This intensive analysis requirement for traditional machine learning approaches is depicted in sub Figure 4.5(a). The advantage of using an AIS as shown in sub Figure 4.5(b) is a distinct advantage in terms of pre-processing of flight data. This is based on an AIS which is able to do self/non-self discrimination of input data.

4.2.3.3 Disadvantages

The use of artificial immune systems does have some disadvantages however, AIS are relatively obscure and this reduces the amount of people that can competently design and use an AIS. They also at the current time have very high computational requirements and although there are ways to mitigate this to some extent this remains a weakness. A running AIS also has no specific end point. Sometimes the algorithm can be run for several multiples of the amount of data and it can be observed to achieve a steady state in terms of the antibodies created but it can be difficult to determine if there is further improvement that could be achieved as estimating coverage of the problem space is a non trivial task. Given a large dataset(500,000+ elements) the dataset will take a long time to fully explore. The algorithm evaluated in the experiment for this thesis was estimated to have required a total of 6 months to evaluate every item of data as a potential antibody.

In section 7.3 the results are discussed and the average time taken to go through each iteration was 92 seconds on a high performance workstation(Intel i7 CPU and 16GB RAM). Applying this to the dataset size and the need to go over at least $2n$ iterations to allow the AIS to continue to develop past initial stimulation the time required on a single PC becomes very large.

The immune network concept

B-Cells maintained dynamically using feedback mechanisms to ensure that useful B-cells are retained in the immune system until they are no longer required. The level of b-cell stimulation is governed by how well it matches an antigen and also

its affinity (how well it matches) other B-cells in the network. If the stimulation level of a B-cell falls below a certain level it will die. So use of the ROC(Receiver Operating characteristic)[158] curve to evaluate how well a B-cell matches an antigen and a new affinity measure to compare how similar different B-cells are. Which rules are used and what constants and variables are used? There are many different ways to measure the similarity of cases to a current situation to assist in their retrieval[159].

CBR Similarity measures

- distance based computational approaches
 - Euclidean
 - Hamming (bit string representation)
 - Manhattan
 - Mahalanobis
- indexing (representational) approaches. cases are connected by indexing structures

The most appropriate for any situation is a complex question to answer and it is a difficult thing to try to go back over cases and add a new similarity retrieval metric. The data needs to be populated at case creation and the methods be included during the design of the system.

4.2.4 AIS vs classification

As discussed in section 3.3.2 classification type machine learning or data mining has an issue regarding how the algorithm can be taught to discern which class data or cases should belong to. Often the solution adopted is to train the algorithm on some training set and after testing to use it on the application. This assumes that the training set is sufficiently representative of the data which will be encountered in use by the algorithm. In this research the AIS will be expected to learn new faults during the lifespan of the aircraft. A method will need to be found to classify the flight data. This method shown in Figure 4.2 classifies the flight into either self or non-self. This is a simpler level of classification that could be done

by a maintainer. This contrasts with the complex and difficult classification of sections of the flight itself into fault and non fault sections. This exercise requires a subject matter expert and a time consuming process of analysis. This contrast in methodologies is shown in Figure 4.5 which illustrates the difference in processing requirements.

False positive results are of concern for any diagnostic or prognostic tool as the cost of maintenance action and the lost revenue from the out of service aircraft is significant. False positive diagnosis incidents will be monitored with a false positive count attached to the offending B-cell. This count would increase the rate of cell aging to remove the B-cell from the system unless. This count can also be decremented if the same B-cell has a series of successes before it goes through apoptosis. The occurrence of false positive does not necessarily mean that there is no fault but it does mean that there was not sufficient information within the system to justify ignoring the alarm and also not sufficient information to identify the fault(if there was a fault which wasnt found). Careful attention to false positive rates and the B-cells which have historical false positive incidents associated with them will be necessary. The associated cases from previous false positive alerts from the same B-cell should at least be useful in diagnosing incidences or ruling out certain faults. The classification of aircraft case histories as members of classes of fault is very useful from a cognitive perspective as it is a way of clustering cases to assist analysis. This assists users of the system in their ability to see similarity between cases and also to be able to contrast the areas where cases are not similar despite sharing a common fault. This represents a crucial ability of AIS to assist operators in the understanding and codification of system use. It means that AIS suggested actions can be traced to the rule that suggested the action, the data which triggered the rule, and also the cases from which the rule was derived. This organization of case data, rules, and faults can be a great asset to users in managing their system efficiently.

4.3 AIS Case Based Reasoning

The inherently case based nature of the immune system was demonstrated by Watson et al by demonstrating an AIS doing CBR for a travel problem[91]. This was an early look into one of the main problems for case based reasoning which is how to store and retrieve appropriate cases. The similarity of cases to current

situations is matched in this concept to the affinity of antibodies to their antigen. The matching method is really very simple. It is a straight field comparison between the potential antigen and the past case. Where there is a match a 1 is assigned else 0. Any sequences of sequential TRUE results are used to give a larger weight to that result as it is judged to be a better match than others which might have the same number of identical fields but which does not have sections which match exactly. This mimics in some ways the action of real antibodies matching against an antigen. When adjacent segments of an antibody match the antigen there is a stronger bond than the effect of those individual segments. Adjacent matches reinforce each other. The smallest divisible matching element in real antibodies is referred to as an epitope. New cells (detectors) are generated which have been slightly mutated from previous successfully used detectors. This differs from some other AI methods such as Genetic Algorithms as mutation is used but not crossover between different detector bit strings. The original detectors can be randomly generated when training the detector but for this research cells are generated from a priori cases where a fault is known to exist. This allows us to start from known anomalous conditions and the system learns the range of variation in the data where the fault exists but its symptoms may be being masked. This overcomes one of the problems of case based reasoning as technical knowledge can be incorporated in the AIS using notes and case details but the burden of giving all possible input combinations for a specific fault is reduced. Case based reasoning (CBR) as introduced in Section 3.2 and shown in Figure 3.1 is an approach to collating learned knowledge into cases and comparing new situations to known cases and either using a previously identified solution from an existing case or creating a new solution and adding the new case to the case base. Once the detector is generated it can be compared against self to ensure it does not alarm in normal conditions. It can then be compared against other detectors already in operation to ensure it is not overlapping coverage of the n -space that is already covered by another detector. Overlapping can cause duplication of effort within the system and is undesirable from a computational burden perspective. If the detector is acceptable it is then stored for use online in much the same way to accepted CBR cases. Overtime the detector ages until it is terminated. While it has been aging new detectors have been added to the population to cover its duties. Detectors which are identified as very good by the system become a priority for clonal maturation to ensure good coverage of the fault they alert against. Especially critical faults and their detectors can also be highlighted in

this manner. With more detectors different characteristic signals of alarm can be found to give detection with earlier onset and better discrimination against no fault conditions.

The CBR System as discussed by Aamodt and Plaza[84] is straightforward to understand in the use of classic engineering analysis to respond to new problems. The elements of a CBR System are the 4 Rs as shown in Figure 3.1.

- Retrieve similar cases
- Reuse the case(s) to solve the problem
- Revise the proposed solution as necessary
- Retain the solution as part of a new case.

This approach to health management is well understood and is easy for operators to relate to. Many will be familiar with and understand the process whereby when a problem is recognized the operator recalls to mind operational instructions or check lists on dealing with the issue and then uses this as a guide for dealing with the problem. However it can be very expensive and time consuming to populate these action scenarios for operators. When the CBR is used as an automated method the development requirements are even more onerous as computer programs are not able to understand the context of problems and impart their own insight to the issue when identifying a solution. This makes CBR Algorithms very expensive to produce and limits the number of cases that can be present. However one of the advantages of CBR for health management is that they can be certified for flight in a much more straightforward manner than many AI systems because they are deterministic. Given a specific set of input conditions a single set of actions will be chosen for those input conditions each time. Other AI methods use elements of probability to improve their solution identification and this adds non-deterministic behaviour which is often difficult to predict.

We illustrate how the immune system is inherently case based and how it relies on its content addressable memory, and a general pattern matcher, to help it identify antigens (new situations) which are similar to existing antibodies (past situations). (Quote from Aamodt and Plaza[84])

Important characteristics of CBR shared with the immune system operation.

- content addressable memory
- decentralised self organisation (immune network)
- inherent pattern matching mechanisms
- retrieval mechanism

4.4 The AIS (artificial immune system) as a candidate method for cognitive AHM

Artificial immune systems have developed since their initial appearance in computing[110]. The immune system exhibits properties associated with a cognitive system such as threat identification, neutralization, memory of threats, and recognition of new threats[109]. AIS have been identified as a useful type of AI and some of the reasons for this are discussed. The original embodiment of AIS was based upon the negative selection Algorithm and the self/non-self paradigm. A detector with some number of features n was compared using one of several available distance(affinity) measures to samples of self and if the detector alarmed it was removed from the population. This reduced the false alarm rate but is difficult to improve over time as the understanding of self and non-self would be expected to improve over time for a system which is under supervision. Data is constantly being generated during use and must be categorized into normal and potential anomalous behaviour. However not all possible coordinates within the feature space(for number of features n the feature space has n dimensions) can be known or classified as self/non-self a priori. This creates the issue of unknown self and unknown non-self showing up as novel within the system and needing to be classified as self/non-self[65]. Immune systems consist of a set of B-Cells which are created through pseudo random genetic variation. B-cells match themselves against the proteins found on the outsides of living cells. If it matches this causes the B-cell to bind to the antigen and release a chemical which attracts a T-Cell to come and absorb the matched cell and all the B-cells which have attached to it. This is the bodies way of reacting to foreign body's such as microbes. Multiple epitopes may exist within

the immune system for the same antigen. This maintains coverage and gives some protection to mutation in the antigen to a new form. See Figure 4.2

4.5 An AIS for through life health management of UAS

To manage the through life health of UAS is a complex task which faces a number of challenges. The need for an IVHM system to assist in the management of through life health has been laid out in Section 4.2 and 4.4 and the need for cognitive features within that system is discussed in section 2.6. AIS share several features with CBR and indeed AIS have been used in the past to power CBR machine learning[160]. This is useful for two reasons. One it provides an opportunity to map out the feature space around a fault. AIS use clonal maturation to create additional detectors similar to a successful detector. This covers the nearby areas within feature space in case of similar faults. This linking of historic case data to the immune cells created from it helps give context to maintainers investigating new alarms from the system. The second reason this is useful is that because the system ties case data to immune cells it acts as a data miner to create a clustered case base populated with flights where faults occurred to aid further engineering work to develop fault detection systems for the faults identified within the case base. The epitopes used by the immune cells in the case base can act as a sign to help aid identify the underlying mechanism of the fault and its degradation mode. This is because they are present within the immune cell as their presence in flight data is highly correlated with the fault for that case. The use within AIS of detectors modelled on B-cells creates a situation where there may be multiple cells alerting against different epitopes of the same antigen. This improves coverage of a single fault with multiple symptoms within the flight data. It also means that the occurrences of a set of symptoms for a fault can be stored as a case for future reference. The use of an AIS for health management has been addressed by Lee et al[111] and a suggested implementation demonstrated by Zuccolotto[161]. The implementation has been for single systems and not been intended as an operational vehicle level system.

4.5.1 Representation

Due to the large amount of real valued data the AIS should use a real value representation. These frequently scale the individual features of the schema to unit length to remove weight issues from the data. All values encountered by the AIS are therefore between 0 and 1 creating a unitary hypercube[116]. This real valued representation takes the form of a string of data fields each with a value between 1 and 0. High dimensionality within AI learning can create a high degree of computational complexity and also reduce the effectiveness of some of the simpler affinity measures such as hyper spheres[162]. This creates a need to allow variable length representations within the schema so that the system can attempt to optimise for shorter schema where possible. This will have the effect of concentrating the features used to those which are diagnostically useful rather than those which over fitting with many different features of the same fault. Coverage of multiple epitopes is better achieved through multiple cells in exchange for this reduced complexity schema. An additional representation can be created for aircraft metadata to enable different aircraft to gain from faults identified in other aircraft. If the serial number, model number, type, and other distinguishing markers are included in a representation schema then Boolean matching can be conducted. Where the fields match a 1 is assigned else 0. This affinity measure can then be evaluated for distance through use of the hamming affinity metric[160]. The immune cell representation will also contain such things as the creation time to enable cell ageing, the current calculated ROC curve, and the case data from which the cell was created. The case data will include the flight reference, the identified fault, the maintenance action taken, and any additional notes the operators may consider relevant.

4.5.2 Affinity measures

The use of real valued representation and bit Boolean matching means that the data portion of the schema will match using a different affinity metric to the metadata portion of the schema.

4.5.2.1 Real valued representation affinity metrics

Tuple based Real valued representations use the “real values” from some sensor data. Whether that value is in Volts, Newtons, or Pascals it is typically a continuous variable which exists within some expected range. A single variable is regarded as one dimension and n variables is represented as a n dimensional feature space. The simplest affinity metric to use is some variant of Euclidean distance calculation within the n dimensional feature space. This has been criticised for $n > 41$ however due to the majority of the volume of the hyper sphere being located close to its surface. This reduces its effectiveness at giving a good distance measures to other locations within the space and makes it difficult to calculate its volume so an estimate can be made on the self/non-self coverage of the feature space[162]. Hyper spheres do have the benefit of being a simple distance measure however. Only a radius is needed to specify the coverage of a hyper sphere which greatly simplifies the changing of the volume of the detector coverage to adjust its ROC curve. As the dimensionality of the feature space in the simulation is less than 40 the hyper sphere method was chosen. In future other methods can be evaluated for higher dimension feature spaces.

Signal based Some alternative real valued representation affinity metrics can focus more on the signal itself rather than the value of the signal in comparison to other system signal values as used in the tuple based approaches. Affinity between two signals or groups or signals can be based on shape comparison or on signal characteristics such as frequency or spectra. These methods are more complex however, and are more suitable for situations where only a few variables are being examined.

4.5.2.2 Boolean matching affinity metrics

Measuring the distance between two bit strings has been worked over several times[116, 160]. The two strings are compared using the field values and assigning a true for a match and a false otherwise. This is then evaluated for the number of matches and consecutive matches are given a higher weight due to the implied higher affinity.

4.5.3 Maturation process

Initial immune cells are populated stochastically to generate random string detectors. Each string represents a position within the feature space of the algorithm. Each string has a metadata section which records the asset for which it was generated. These detector strings consist of epitopes for possible antigens. The generated detectors are then presented with flight data to evaluate their ability to correctly diagnose faults. The ability of each detector to identify a fault is evaluated via the chosen affinity measure. Those detectors which have sufficient affinity to the epitope of a known fault to create a match are retained and the remainder discarded. The detectors can then be evaluated based on the strength of their affinity to the fault against which they alarmed. Stronger affinity is represented by a smaller Euclidean distance within the feature space between the detector string and the fault case data. The detector can be evaluated with different settings for the radius of the matching rule. This allows an ROC curve to be built up for each detector in relation to a fault identified from previous flights. Gradually immune cells which do not exhibit enough specificity die through the ageing process. New cells are generated stochastically and through clonal maturation where cells have demonstrated a high enough specificity through their ROC curve. As the cells continue to mature the lower specificity cells die off while the more effective classifiers are cloned and go through clonal maturation. This process continues until a threshold is met for the specificity of a percentage of the population. The key is the use of some specific feature such as the early replacement of a part/LRU and binding it to a flight dataset as a classifier. The AIS can then determine what about the flights preceding this maintenance event was unusual(non-self). This is the identification of the epitopes of the flight data symptom which caused the maintenance event.. This will create a rule set which can then be used to create an advisory and also to guide future research for a more highly optimised diagnostic/prognostic method for the specific failure mode involved. Thus each time there is a fault a case is generated for it and added to the case base. Items within the flight data are scrutinised to generate a rule set which would diagnose it. Faults sharing a common mode or subsystem should will be clustered within the feature space so that they can be identified as sharing a common root.

4.5.4 Danger Theory

In use the AIS can respond to stressor signals either created by individual health enabled sub-systems or in response to a danger signal initiated by the pilot. The algorithm could then check the affinity requirements for alert to generate a list of the most similar immune cells with attached cases to the current aircraft condition as assessed from the flight telemetry. The case data presented can include the checklist for the fault identified within it to reduce the workload on the pilot in identifying the correct procedure to use. Danger Theory is the ability of the cell to alarm when a lower quality with a lower quality match if a specific context sensitive *Danger Signal* is present[163]. This permits a lower false alarm rate than might otherwise be possible as in locations where the danger is not present the cell uses its normal alerting threshold but where there is danger it will use a lower threshold. Figure 6.2(a) depicts how this functions within the paradigm of a euclidean distance metric for affinity. The concept of context sensitive classifiers was introduced in AIS to explain the behaviour of antibodies which alert in certain areas of the body during infection but do not react in other areas where there is no infection. It was suggested that affinity threshold for an alert was lowered in the presence of a danger signal and thus the overall false alarm rate could be reduced while maintaining fast reaction times and high true positive rates when danger was known to exist.

Different types of Danger Theory Implementation

1. Relationship between a context sensitive “danger” signal and the functioning of an online AIS.
2. Use of danger signal with rules created by offline learning.
3. Use of danger signal to enhance offline learning

The use of danger signal is intriguing and merits further investigation but is not of immediate use as if the user of a through life CAHM system could classify data elements as being more likely to have faults(danger) there would be little need for the CAHM as the user would already be able to perform that function on their own. It has been used during online AIS used as panic button method. During flight it has been proposed that the use of emergency settings of the declaration of an in-flight emergency could be used as a danger signal within an AIS for aerospace use but that will remain something for future development.

4.6 Comparison of CAHM, CBR, and AIS-CBR characteristics

A suitable system to manage the health of assets in a cognitive way needs to be able to manage a body of knowledge concerning how assets should be used and managed. In a CBR this is done through a body of cases that have been developed over time and contain recognition criteria and the likely solution. This can then be adopted or adapted depending on how well the most relevant case fits the current situation. This maps nicely to the AIS concept as threats to the system are remembered by a population of antibodies that have within them a specific solution. In the biological context this is typically removal of the threat but in asset management there is nothing to prevent this being more detailed and having similar information to the CBR. If the solution is to be more cognitive then one way to do this is to implement an architecture compatible with a cognitive framework such as Rasumssens in Figure 2.11. Developing and using rules to handle specific situations is similar to the development of solutions within CBR and within AIS. The tricky part about use of CBR has always been the recognition criteria and the method to establish the most relevant case. AIS is able to assist in this process by mapping cases to antibodies so the recognition and relevance is performed by the reaction of the antibodies. Adding new cases to the system in a useful way for CBR requires specific work and no additional work is required for AIS to assist in this process as once a flight has been identified as having a specific problem antibodies can be trained to recognise the fault features within that flight. There is thus a reasonable way to add new cases to the system during the life of the asset. As cases are added and the population of antibodies becomes more robust they can also be compared between fleets of asset to examine commonalities and to see if antibodies developed for one aircraft type can be usefully applied to another. If this can be achieved in mimicry of the way antibodies are shared in humans this would be a major improvement over current methods of learning from the operations and management of separate and different assets. The use of an AIS for CAHM could be applied particularly to the UAS case as there is a need to create different solutions onboard to those that will be used offboard. Figure 3.2 in Section 3.2 shows a higher level view of how a CBR machine learning method could be integrated into the SMS and the same could be done with the CBR type of AIS.

4.7 Summary

The use and comparative advantage of artificial immune systems have been discussed and their potential benefits for the through life support of a fleet of aircraft as part of a maintenance information system examined. The cognitive properties of the system have been discussed and the case based nature of such a system also examined to demonstrate the suitability of such a system for capturing faults into a system for future exploitation.

Chapter 5

Experiment Design

5.1 Introduction

In this section the structure an experiment to examine the effectiveness of an AIS will be discussed. This will be specifically for the problem of learning faults from flight data and how to obtain a comparison to other methods. The use of simulation is discussed along with its limitations and the mishap flights used for training are examined.

- In section 5.2 the nature of the data that would be encountered in a real unmanned aircraft shall be examined and how this will guide the design of the experiment in order to generate representative flight data to evaluate using machine learning.
- In section 5.3 the Aerosonde unmanned aircraft will be discussed as a candidate for simulation, why it is a good example of unnamed aircraft more widely and how it's missions may be simulated.
- in Section 5.4 the design of the mishap scenarios for the experiment will be discussed, suitable mishaps identified and their representation within simulation explained.
- In Section 5.5 the design of the simulation missions will be discussed and how the overall simulation was structured to enable multiple repeatable simulated missions in order to build a large data set.

- In section 5.6 the methods by which the flight data created through simulation will be preprocessed shall be discussed and how this may be done in a representative way ready for machine learning
- In section 5.7 the methods by which the different machine learning methods may be evaluated shall be discussed.
- In section 5.8 the mishap cases generated through simulation will be examined and the methods by which the mishaps could be learned from using current practice shall be discussed.

5.2 Aircraft Flight Data

To understand how the simulated aircraft should be structured the type and nature of the data that could be obtained from a real UAS must be discussed. There is a need for solutions that can be applied across architecture scales or at least single board through to distributed. The ability of these boards to log data to on-board memory is a small addition and already present in a great many lower cost devices. General aviation is also starting to see lower cost FDR(Flight Data Recorder) devices[164]. In order to generate some representative aircraft data that could have been logged using these newly available FDR devices a simulation was constructed within Matlab. A selection of the variables that would be expected to be seen within the flight data of real manned and unmanned aircraft are shown in figure 5.1. Within the limits of flight simulation a repeatable and robust dataset of normal and mishap flights can be generated in order to train the AIS. These are shown because they represent some of the more easily misunderstood variables and sign conventions.

5.2.1 Flight Simulation

There are a large number of ways to generate aircraft flight data to include mishap data. They range in complexity and scale from simple ballistic physics calculations to whole aircraft CFD (Computational Fluid Dynamics) simulation including moving control surfaces. The question for this research is to what is appropriate for simulating faults for unmanned aircraft in order to generate an FOQA style fleet dataset. There are many different ways to simulate the flight of an aircraft

in a pc. There are black box methods where a proprietary code block is used to represent the aircraft state and control signals are supplied to it and it outputs aircraft position and state data. Faults are implemented in control channels rather than as physical effects in feedback to aircraft performance. More sophisticated flight simulation such as x-plane or flight simulator required but this increases runtime massively and the current offerings must be performed in real time thus creating a time problem. This increase in time does not offer a significant advantage so was not pursued. There are linear flight models which attempt to use a map of the change in aerodynamic performance with aircraft state to represent the aircraft and give a good approximation of performance across the expected flight envelope[165, 166]. However the accuracy degrades for large bank angles, large pitch, large yaw, and low speed due to turbulence creating uncertainty. This would be an issue if it is intended to deploy the system created to a real aircraft but since this is a simulation evaluation only it is not necessary for this study. There are also more sophisticated partial CFD solvers which take a geometry model of the aircraft and details about it's engines and control surfaces and attempts to accurately simulate real flight conditions. Two examples of this are JSBSim[167] and Xplane[168] which are both frequently used in flight simulation. These provide very good fidelity for many applications but are very complex to integrate fault simulations into and so can represent a large investment of time. On the upper end of the scale are commercial grade simulators which often include some degree of motion cueing within a cockpit to further enhance the training aspect of flight simulation. There has been data generated through these simulators but access is expensive and when the simulation target is the automated flight of an UAS it does not make sense to use a 6-DOF (Degree of Freedom) flight simulator. Some excellent work has been done using this type of system[169, 170] but in this research where the need is to generate several thousand hours of flight it cannot be justified. Due to the need to generate a large amount of flight data including custom designed fault simulations it was decided to use a linear flight model to take advantage of an existing flight model within Matlab for which an autopilot already existed and only faults, mission generation, and simulation automation needed to be done.

Advantages

The advantage of a linear flight model in this situation are that it can be quickly iterated and tested to evaluate new fault models, mission scenarios, and control

methods. It reduced code requirements to the Matlab environment only instead of needing to integrate a third party program into a experiment written in a language like Python which could automate the experiment and add fault information. The use of the linear flight model also meant that large amounts of flight data could be generated. The total amount of data generated is discussed in Section 5.5.

Disadvantages

The use of the linear flight model does however have some disadvantages and while they are not show stoppers they need to be understood in order to design the experiment correctly. The chief among them is that the model cannot be validated against real Aerosonde flight data. This is due to lack of access to that commercial sensitive data rather than any theoretical model limitations. If a safety system was being developed for the real Aerosonde system using this method that problem could be overcome. In this situation true verisimilitude is not required and it is more useful that the model will remain deterministic and consistent. Another disadvantage is that the accuracy of the model decreases in situations where the flow over the aircraft is likely to be turbulent rather than laminar. This particular disadvantage as with the last is more critical if a system is being developed for deployment with a real aircraft rather than for evaluation of machine learning methods as it is here.

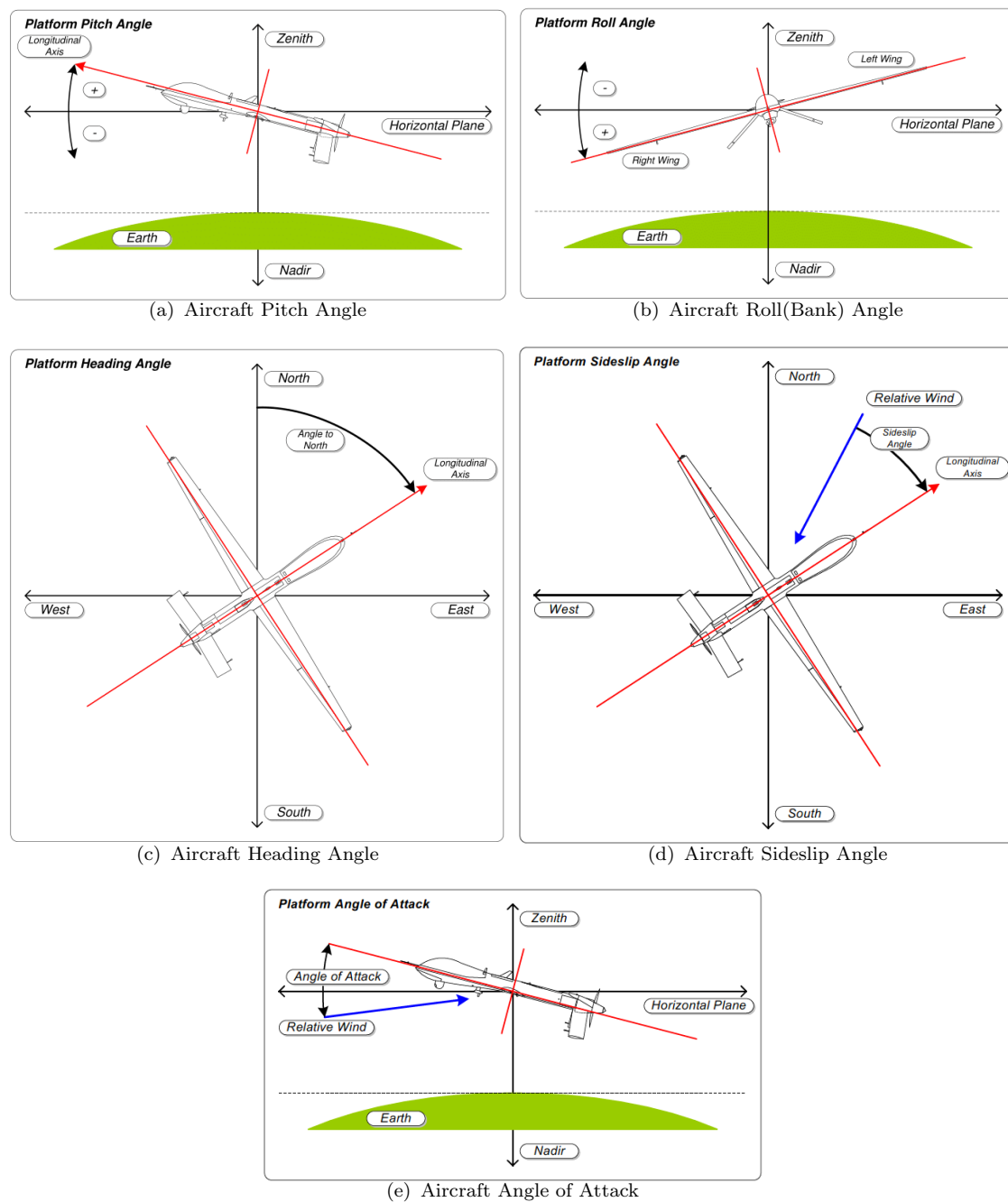


FIGURE 5.1: Aircraft Orientation Conventions illustrated using a line drawing of a Predator from GWG UAS Datalink Standard[171](Geospatial intelligence standards Working Group)

5.3 Aerosonde

The Aerosonde model provided by Dr James Whidborne is a model in Simulink of the Aerosonde unmanned aircraft(see Figure 5.2)[172]. The autopilot was created by Dr Deborah Saban as part of her research into wake modelling[7]. The Aerosonde model was chosen due to its status within current aircraft simulation. The AeroSim model[173] from which this model is derived has been used in several studies by other researchers and thus has some standing[174–178]. Initially a predator simulation was evaluated for the experiment but the low level of documentation and the lack of a method to evaluate the models aerodynamic performance restricted the options. The Aerosonde model was therefore chosen. It does however have one quirk. The Aerosonde itself has a tail of a type commonly referred to as an inverted v tail but the simulation assumes a more conventional cruciform tail. This was accounted for by the developers by the use of offsets to ensure the standard methods for a linear flight model could be used[165, 166]. The Aerosonde was originally used in a semi-autonomous and disposable way to perform atmospheric sampling. It is this early version of the Aerosonde which is represented in the Matlab simulation. This same version was used for the first transatlantic crossing by an unmanned aircraft[179]. Aerosonde is launched and the after ground control during initial stabilisation and climb it flies a preplanned route out of line of sight of the operator. As is typical of many smaller unmanned aircraft there is no position feedback of the servos that control the various mechanical components such as the throttle, ailerons, and ruddervator. This means that any mechanical problem or servo failure can only be detected by its effect on the flight of the aircraft.

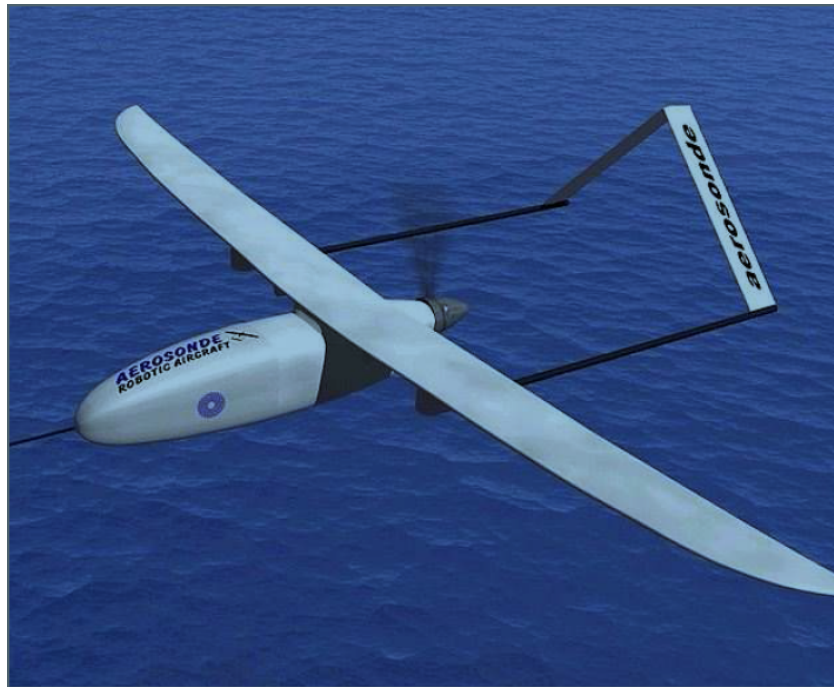


FIGURE 5.2: Aerosonde Aircraft[180]

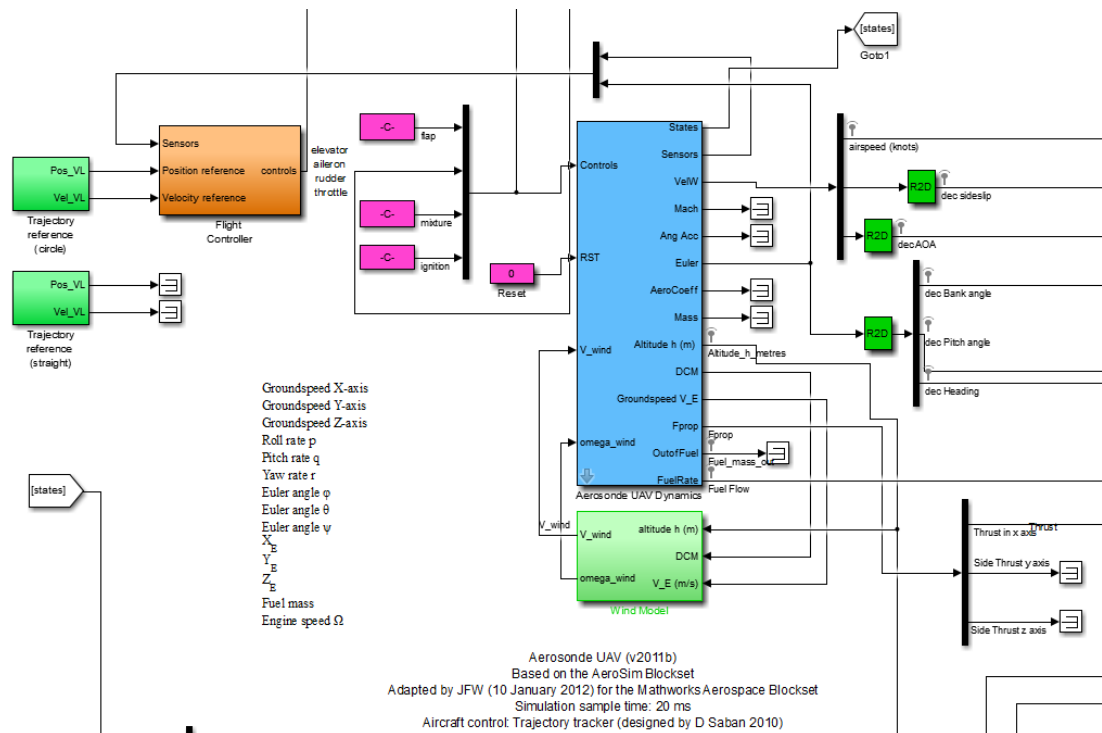


FIGURE 5.3: Aerosonde Simulink Overview Aircraft Model by Dr Whidborne[172], Autopilot by Dr Saban[7].

5.4 Mishap Design

Examination of some of the methods used to simulate faults in simulation and the methods used in the experiment. Several mishap types were selected from the mishap records of the USAF Predator fleet[24, 181] to represent realistic failure modes for the Aerosonde simulation. Fault onset time in all cases was a time chosen to be normally distributed about a suitable point during the mission. This was done to avoid all the faults happening at the same point but avoid the need to manually specify the failure time for each mission. This was set at $1,100 \text{ seconds} \pm (1,1000)$. This created a fault onset time of somewhere between 25 to 41.6 minutes into the flight. The other scenarios involved either a permanent jam of a control mechanism or a jam then latency. The mechanism for the jam and jam plus delay is mostly the same with that of the ruddervator being the most complex due to the nature of differential controls. A simple sample and hold was used with the hold either being for a short time to simulate a jam and then delay or an effectively infinite amount of time to simulate a jam. Details for the arrangement within Simulink can be seen in Figure 5.4. The ruddervator fault design is shown as it is the most complex fault because the use of an inverted v tail in a linear model as discussed in Section 5.3 requires a different configuration than would otherwise be needed. The affect is that for the development of the fault case affecting the ruddervator both the rudder and the elevator signals needed to be affected at the same time to simulate the effect of a differential surfaces being jammed.

5.4.1 Permanent Fault Scenarios

In this section we will discuss the fault scenarios which involve a permanent fault.

5.4.1.1 Fuel Leak

This fault scenario is a simplistic leak of fuel within the fuel tank. It manifests in a simple way as a constant rate fuel leak starting at a specific time. The failure scenarios of a fuel leak happened at various rates normally distributed around two kilograms per hour plus a random amount of between 10 and 200 grams per hour in increments of 10 grams per hour. Mishap4840 shown in Section 5.8.1 on page 112 is a mishap case which was generated using this scenario. The mishap of

USAF Predator S/N (Serial Number) 99-003056 on the 27th March 2005 was the inspiration for this mishap scenario as it involved a fuel leak which resulted in a fire and the loss of the aircraft[182]. In that mishap the fuel leak was believed to be caused by a maintenance failure. In this scenario the fault does not include the fire due to the complexities of modelling this but instead simulates fuel leaking until running out. The Aerosonde unlike the predator does not have the facility to send fuel status back to its controller and has a much smaller fuel tank and so a fuel leak is a much bigger problem.

5.4.1.2 Control Jam

The fault scenarios described in this section are all types which shall be called control jams are all faults which manifest as a failure of a control surface of the aircraft (ailerons, elevators, rudder, throttle, etc) to respond to commands from the autopilot and remain at the position to which they were last commanded.

Throttle Control Jam

In the throttle control jam scenario the throttle signal within the simulation is sampled and at when the fault is initiated the throttle signal is held constant at the level it was when the fault was initiated. This is representative of a mechanical jam within the throttle linkage or some other similar situation where the throttle setting cannot be changed. Some linkages may be spring loaded and would cause the throttle to go to the idle position with the loss of actuation force. The mechanical arrangement of Aerosonde is not known but the effect of a loss of thrust or an inability to change the thrust of an aircraft was a key factor in several Predator mishaps and so this fault mode was added to the fault scenarios. Mishap4894 shown in Section 5.8.2 on page 115 is a case which was generated using this scenario. The mishap of USAF Predator T/N (Tail Number) 06-3178 on the 20th August 2011 was the inspiration for this mishap scenario as it involved the loss of the ability to change the thrust level in the aircraft. In that mishap it was due to a failure of the quill shaft in the VPP(Variable Pitch Propeller)[183]. In this scenario the propeller is fixed pitch but the effect of loss of thrust control is similar.

Aileron Control Jam

The aileron control jam scenario explores the reaction of the aircraft to the mechanical jamming of an aileron. Due to the way the simulation is constructed it could only be modelled with both the port and starboard aileron jammed at the same time. The fault is initiated at a chosen time and the aileron control signal is held constant at the level it was when the fault was initiated. One example of this is Mishap5080 which is shown in Section 5.8.4 on page 123 is a case which was generated using this scenario. The use of electro-mechanical actuators in many small and medium sized unmanned aircraft leaves them vulnerable to a mechanical failure rendering one control surfaced jammed. The corresponding lack of feedback control to the autopilot makes it difficult to detect this type of failure except through the flight response of the aircraft. The mishap of USAF Predator T/N 06-3173 on the 7th May 2011 was the inspiration for this mishap scenario as it involved the failure of an aileron servo which resulted in uncommanded roll[184]. In that mishap it was due to a failure of the servo in one wing only. In this scenario the fault affects both ailerons but the effect of uncommanded roll is the same.

Ruddervator Control Jam

The ruddervator control jam scenario explores the reaction of the aircraft to the mechanical jamming of the ruddervator. Due to the way the simulation is constructed it could only be both surfaces being jammed at the same time. The fault is initiated at a chosen time and the ruddervator control signals are held constant at the level it was when the fault was initiated. The Simulink mechanism for this can be seen in Figure 5.4. The two blocks labelled sample and hold take the control inputs one and two from the autopilot and store the last observed value. When the fault initiation time labelled as “ruddervator_lag_start_time” is reached this triggers the sample and hold blocks to keep their last value and not sample any more and also sends their output to the flight model as the new control signal. One example of this is Mishap4992 shown in Section 5.8.3 on page 120 is a case which was generated using this scenario. The mishap of USAF Predator T/N 05-3147 on the 14th September 2009 was the inspiration for this mishap scenario as it involved the failure of an tail servo which resulted in a loss of attitude control[185]. In that mishap it was due to a failure of the servo in one tail section only. In this scenario the fault affects both ruddervators but the effect of loss of attitude control is the same.

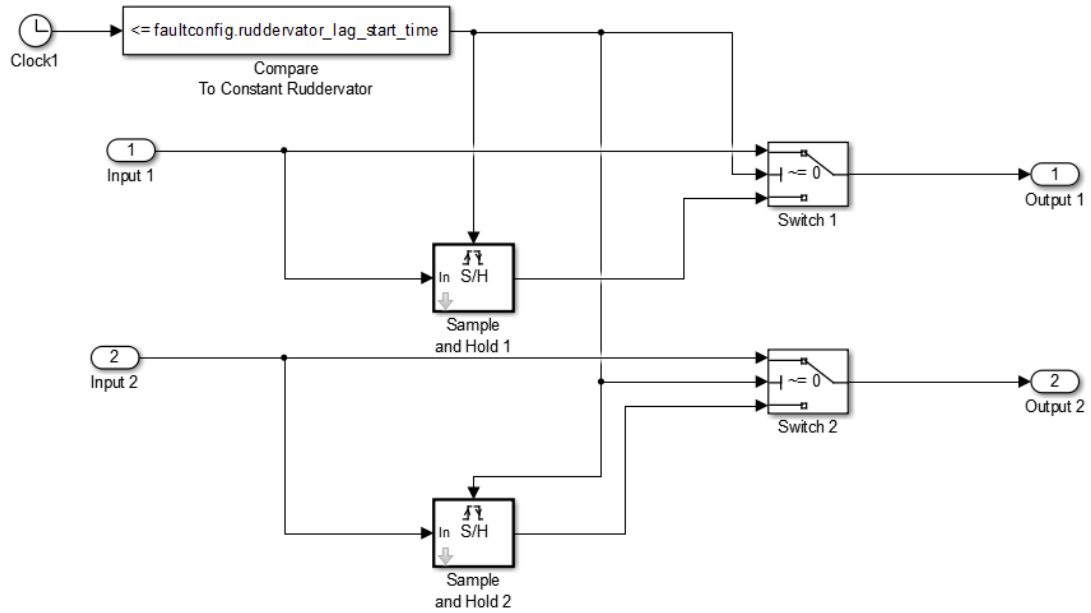


FIGURE 5.4: Aerosonde Simulink Ruddervator Fault Block. During a fault mission if this fault is selected it will simulate a jam of the ruddervator control servo starting at a randomly chosen time during the mission.

5.5 Mission Design

The mission profiles chosen for the Aerosonde simulations needed to be something that could be generated algorithmically and with a high degree of repeatability in order to generate thousands of flights worth of representative data. Each mission would have similar elements but would be different and the simulation would present random factors such as wind strength, direction, and gust level during the flight to ensure even with very similar mission profiles the resulting data would be representative of a similar aircraft flying a similar profile on a different day rather than the same aircraft flying the same profile in the same conditions. A number of different mission types were examined to evaluate how they would meet the requirement and how the different Mishap scenarios would play out within the different mission profiles. The process for this can be seen in Figure 5.5. The details for each flight were logged in an excel spreadsheet to allow review of certain summary details and the tracking of the total number of flights and mishaps. During early evaluation it was found that despite the crippling nature of the chosen faults a crash would not result in all situations. This led to a change of the mission profiles to ensure a long flight time and maximise the potential for a mishap. The Aerosonde has a long endurance as shown by the Laima transatlantic flight[179]

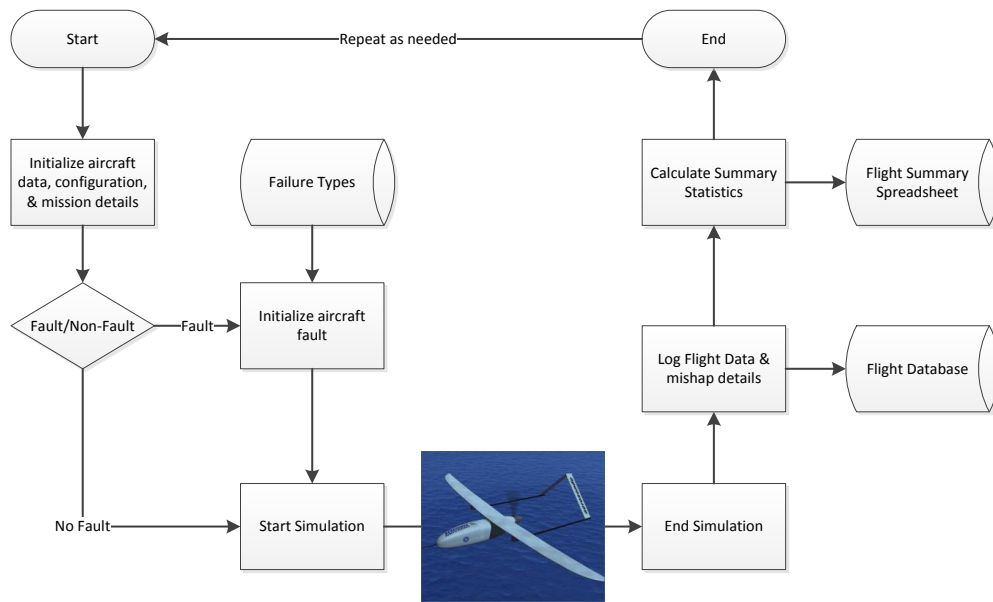


FIGURE 5.5: Aerosonde Simulation Flowchart

and this was used by creating mission profiles with a long duration with relatively few way points in order to keep profile generation simple. There is therefore a high proportion of straight and level flight within the mission profiles.

5.6 Pre-Processing

In order to use the flight data developed through simulation it was necessary to pre-process the data to remove any skewing effects that might be caused by some of the data having different ranges to others. Due to the euclidean distance metric being used to calculate affinity by the AIS if only one variable was being used this would not need to be done but as soon as multiple interacting variables are being used they need to be mapped into a unit interval so that there is no scale effects skewing the results. This is sometimes referred to in some papers as normalisation but is more properly known as scaling into a unit interval. The formula for this is given in Algorithm 3. It is a very simple equation and to ensure the antibodies are able to be used later with flight data the details of the minimum and maximum values were retained to ensure the scaled values could be related to the real values at a later date. These scaled data is then used as training

data in the creation of antibodies through the AIS-FOQA Algorithm. This allows multiple antibodies within the population to have been trained at different times with different minimum and maximum values for each variable but they can all be mapped into a single n dimensional unit space.

$$x_{\text{norm}} = \frac{x - x_{\text{set minimum}}}{x_{\text{set maximum}} - x_{\text{set minimum}}}$$

Algorithm 3: Equation for Scaling into Unit Interval

TABLE 5.1: Minimum and Maximum values recorded for each channel within the flight data.

Variable Names	Units	Max	Min
elevator_out	Radians	0.14945	-0.39087
aileron_out	Radians	0.39313	-0.65551
rudder_out	Radians	0.25481	-0.32459
throttle_out	%	1	0
airspeed__knots_	Knots	37.7288	16.435
dec_sideslip	Degrees	10.2151	-18.8297
dec_AOA	Degrees	7.9641	-0.69256
dec_Bank_angle	Degrees	51.767	-88.6878
dec_Pitch_angle	Degrees	22.9824	-3.12
Altitude_h_metres	Metres	633.9165	0
Fuel_mass_out	Kilogram	2.2579	0
Thrust_in_x_axis	Newtons	30.3266	-3.6381
Fuel_Flow	Kilogram/Second	0.00062197	0
X_E_out	Metres	127955.7569	-15722.8086
Y_E_out	Metres	8044.5659	-8151.4107
Z_E_out	Metres	420.9035	-633.9165
Distance_travelled	Metres	802480.1748	0

5.7 Machine Learning Evaluation

The Algorithm to be evaluated using the flight data discussed is an artificial immune system and will be compared to a neural network trained in Matlab using the same data and to a representation of a case based learning method of through life flight safety. The AIS to be used is discussed in Chapter 6. A smaller dataset will be used to evaluate the AIS during development and to assess its capability and integrating new knowledge and adapting to change. The dataset selected is the Fisher Iris dataset supplied with Matlab[186]. This dataset is derived from a paper by R. A. Fisher[187] and is widely used to evaluate machine learning approaches which use data from multiple continuous variables as an input. The use of the fisher iris dataset to validate the test the Algorithm during development is discussed in Section 6.5. Due to the multi class nature of this experiment the true positive class is limited strictly to the most relevant mishap class and only for the period in the simulation after which the fault occurred. This in some cases artificially reduces the True Positive Ratio where the Algorithms are matching against elements they were trained with as representing a fault data point. However this is required as the section prior to fault onset represents health flight and so should not be included in any assessment as True Positive despite it being data from a mishap flight. This will have the effect of increasing the False Positive rate where the flight has been correctly identified but this method is correct as otherwise the algorithm alerting against healthy flight data would be scored as a success. This contrasts with the Algorithm training portion where the entirety of a mishap flight is classified as True Positive. This is intended to take advantage of the self/non-self discrimination properties of an artificial immune system to avoid the need to manually segment mishap data into fault and non fault portions. This is because as discussed in Section 2.5 there is not always the resource available to do such a costly exercise for what may be one of several mishap events.

5.8 CBR Analysis and Application

In order to evaluate the flight data from simulation mishaps a flight trace strip was created to replicate the methodology used within Flight Data Management and taught at Cranfield as part of the Flight Data Monitoring short course by David Wright et al[188]. As discussed in section 2.3.2 there are a wide variety of

tools that can be used to analyze flight data but due to the use of Matlab and Simulink to simulate the flight a custom solution needed to be created in order to make use of these expert methods in the analysis of the flight data. An overlaid strip graph type was created and access methods written to generate Figures for flights showing the data for specific time intervals. In addition an animator method was created to enable the creation of a animation of the flight data so that the whole flight could be reviewed quickly and when discrepancies noticed the time intervals can be noted down for further analysis. The use of expert analysis is restricted to the channels being logged to the flight data recorder of the aircraft or those channels which are streamed over the telemetry link to the ground control station. This can restrict the knowledge that can be created from the data and in some instances specific sensors most useful to diagnostics may not be available in a specific aircraft. In particular small aircraft frequently do not have side-slip indicators or true angle of attack which can make it difficult to understand the environment conditions the aircraft may have been encountering. Each mishap will be examined to see what details can be brought out without reference to the Mishap Scenarios in Section 5.4. Although in each case the failure mode is known and listed on the top of the figure with the fault onset time it is intended to demonstrate the analysis method and how lessons for future operations can be found in a situation where the fault cannot be found through examination of the aircraft due to loss or damage. In each mishap analysis this will be put in the CBR section with both the identifying features and the proposed solution for future operations. The process for identifying the root cause of mishaps and creating practical in flight responses is identified on a high level in Figure 3.1 showing the case based reasoning cycle. The key items required to complete a case are how the case can be distinguished from other cases, the details of the case, and the appropriate response to the fault.

Mishap data and analysis

- **Section 5.8.1 shows the analysis and data for Mishap4840**
 - Figure 5.6 showing controls data point of interest one on page 113
 - Figure 5.7 showing state data point of interest one on page 114
- **Section 5.8.2 shows the analysis and data for Mishap4894**
 - Figure 5.8 showing controls data point of interest one on page 116

- Figure 5.9 showing state data point of interest one on page 117
- Figure 5.10 showing controls data point of interest two on page 118
- Figure 5.11 showing state data point of interest two on page 119
- **Section 5.8.3 shows the analysis and data for Mishap4992**
 - Figure 5.12 showing controls data point of interest one on page 121
 - Figure 5.13 showing state data point of interest one on page 122
- **Section 5.8.4 shows the analysis and data for Mishap5080**
 - Figure 5.14 showing controls data point of interest one on page 124
 - Figure 5.15 showing state data point of interest one on page 125
 - Figure 5.16 showing controls data point of interest two on page 126
 - Figure 5.17 showing state data point of interest two on page 127

5.8.1 Mishap4840

Mishap Overview

It can be observed in Figure 5.6 that between time 4800 and time 4850 a sudden drop in airspeed occurs while at the same time the throttle demand from the autopilot goes to 100%. This is suggestive of a loss of thrust possibly from engine cutout. Further analysis shows a low AOA for this phase of the flight but without a statistical comparison to the fleet averages it is difficult to confidently find the true cause.

Identifying Features

It can be observed in Figure 5.6 that between time 4800 and time 4850 a sudden drop in airspeed occurs while at the same time the throttle demand from the autopilot goes to 100%. This is suggestive of a loss of thrust possibly from engine cutout. Further analysis shows a low AOA for this phase of the flight which would otherwise suggest a low amount of lift required for that phase of flight but without a statistical comparison to the fleet averages it is difficult to confidently find the true cause of this loss of airspeed.

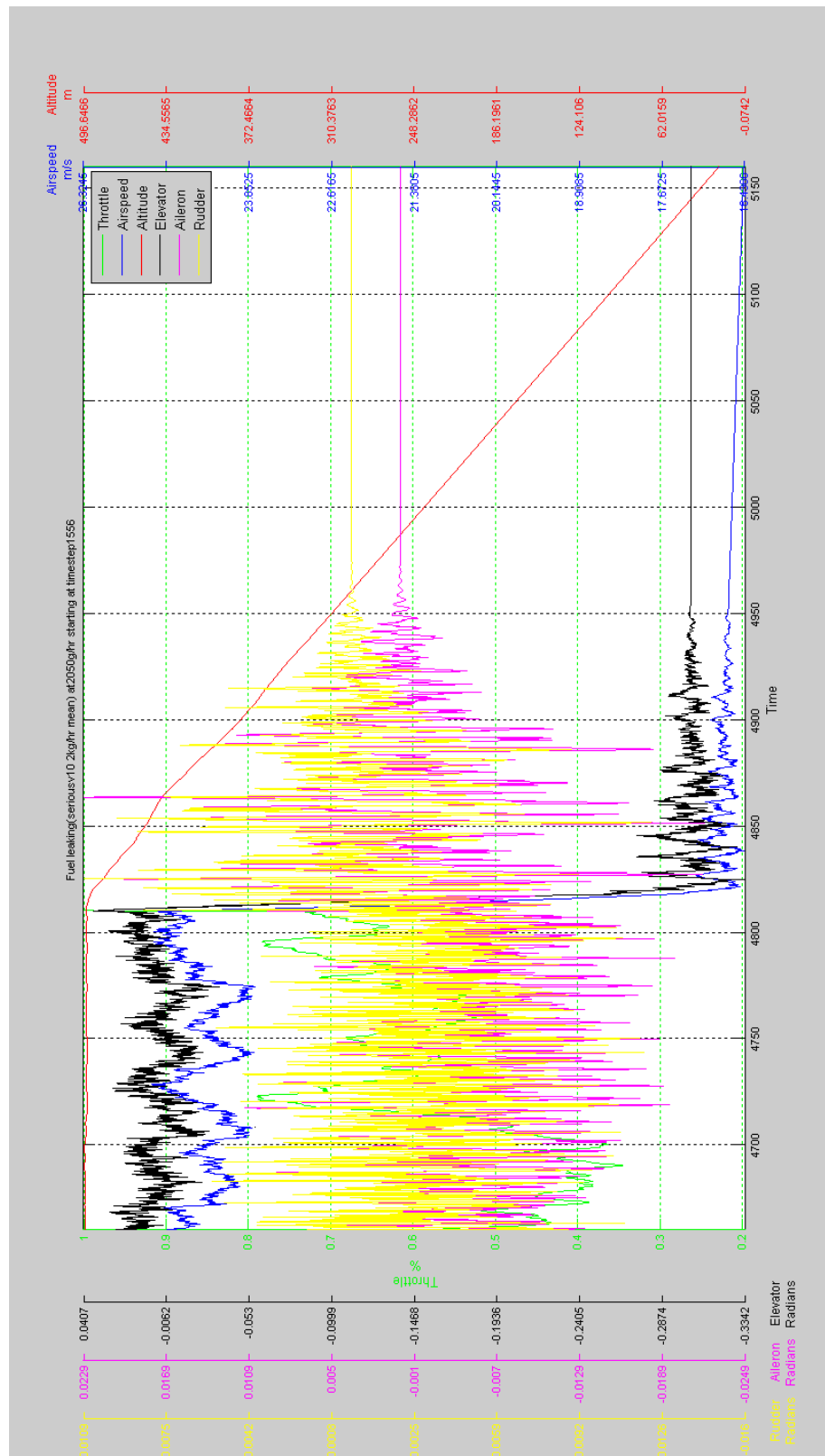


FIGURE 5.6: Sim4840 Flight Data 1 of 2 displayed in an overlaid format. From 4650s into the flight to the crash. The engine stopped due to lack of fuel. The Fuel leak was 10.2kg/hr and started at 1558 seconds into the flight.

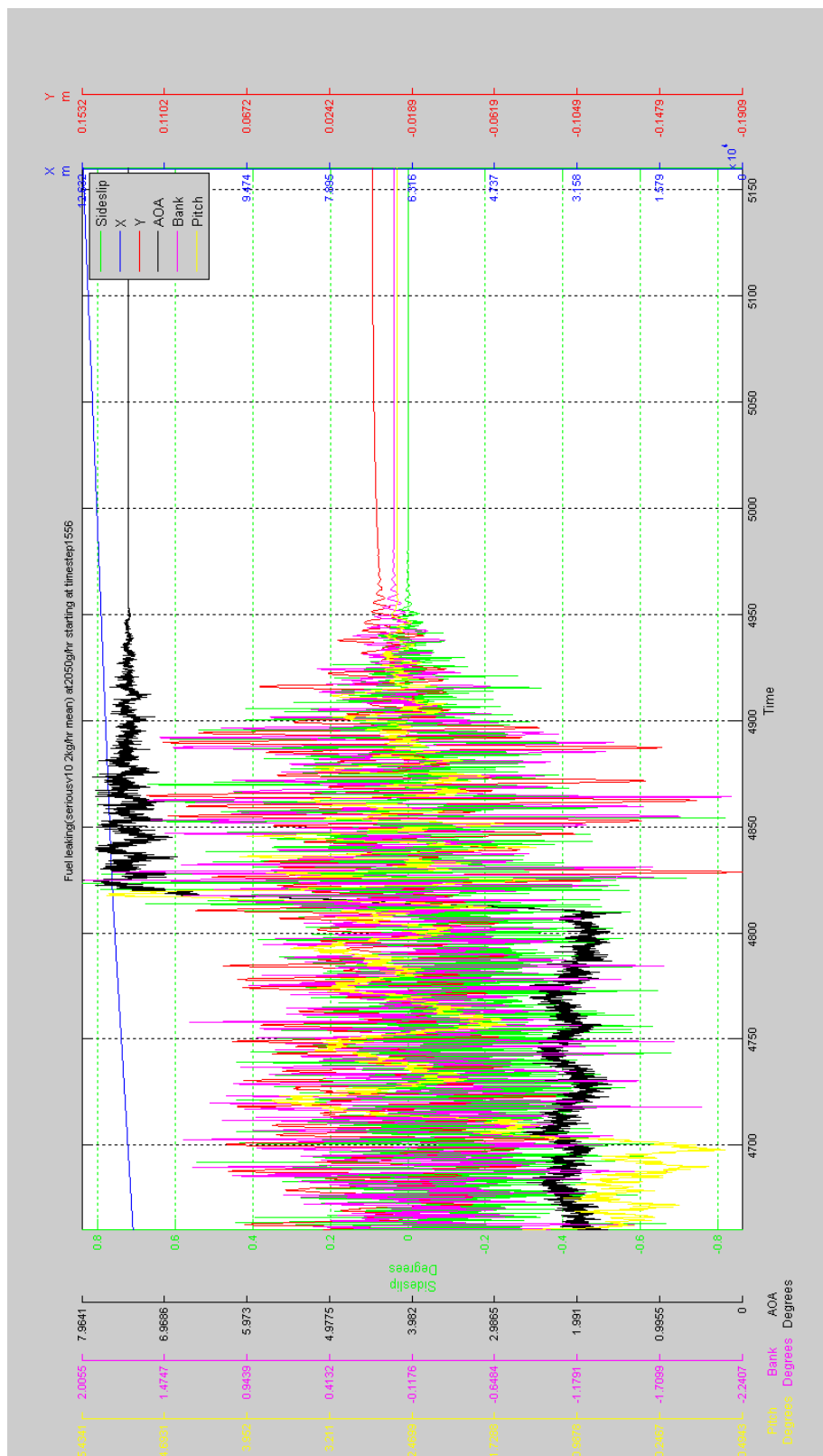


FIGURE 5.7: Sim4840 Flight Data 2 of 2 displayed in an overlaid format. From 4650s into the flight to the crash. The engine stopped due to lack of fuel. The Fuel leak was 10.2kg/hr and started at 1558 seconds into the flight.

Solution

In this situation the likely effective solution would be to take remote manual control and attempt to stabilise the aircraft using the more sophisticated reasoning and skill of a human pilot. As it seems there has been a loss of thrust stable flight is unlikely and an evaluation of possible safe landing areas within gliding range would need to be established. In the event that it is judged too risky to attempt to return the aircraft to a safe landing then a controlled ditching would be the preferred option.

5.8.2 Mishap4894

Mishap Overview

At time 1550s it can be observed in Figure 5.8 that the throttle is held at 0% while the airspeed continues to increase. The throttle is held at 0 for over 300 seconds without a substantial reduction in airspeed. The airspeed for the remainder of the flight is AOA dominated and this can also be seen at time 2250, 2600, and 2750 the aircraft goes out of control due to excessive nose up attitude. During each of these departures from normal flight the aircraft rapidly loses altitude until recovery and climbs again to the altitude which the autopilot was set to cruise at. Finally the aircraft stalls again at 4050 and is not able to recover.

Identifying Features

At time 1550s it can be observed in Figure 5.8 that the throttle(green trace) is held at 0% while the airspeed continues to increase and the altitude remains roughly constant. This is an unusual feature and as such is suggestive that whatever happened during this mishap has some relationship to this unusual event. The throttle is held at 0 for over 300 seconds without a substantial reduction in airspeed. The airspeed for the remainder of the flight is AOA dominated (Figure 5.9 black trace) and it can be seen that the airspeed of the aircraft is strongly affected by changes in commanded elevator angle(Figure 5.8 black trace). During periods of high airspeed the aircraft assumes a large AOA and in some instances stalls. This can also be seen at time 2250, 2600, and 2750 seconds(not shown in this Section) the aircraft goes out of control due to excessive nose up attitude. During each of these departures from normal flight the aircraft rapidly loses altitude until recovery and climbs again to the altitude which the autopilot was set to cruise at. Finally the

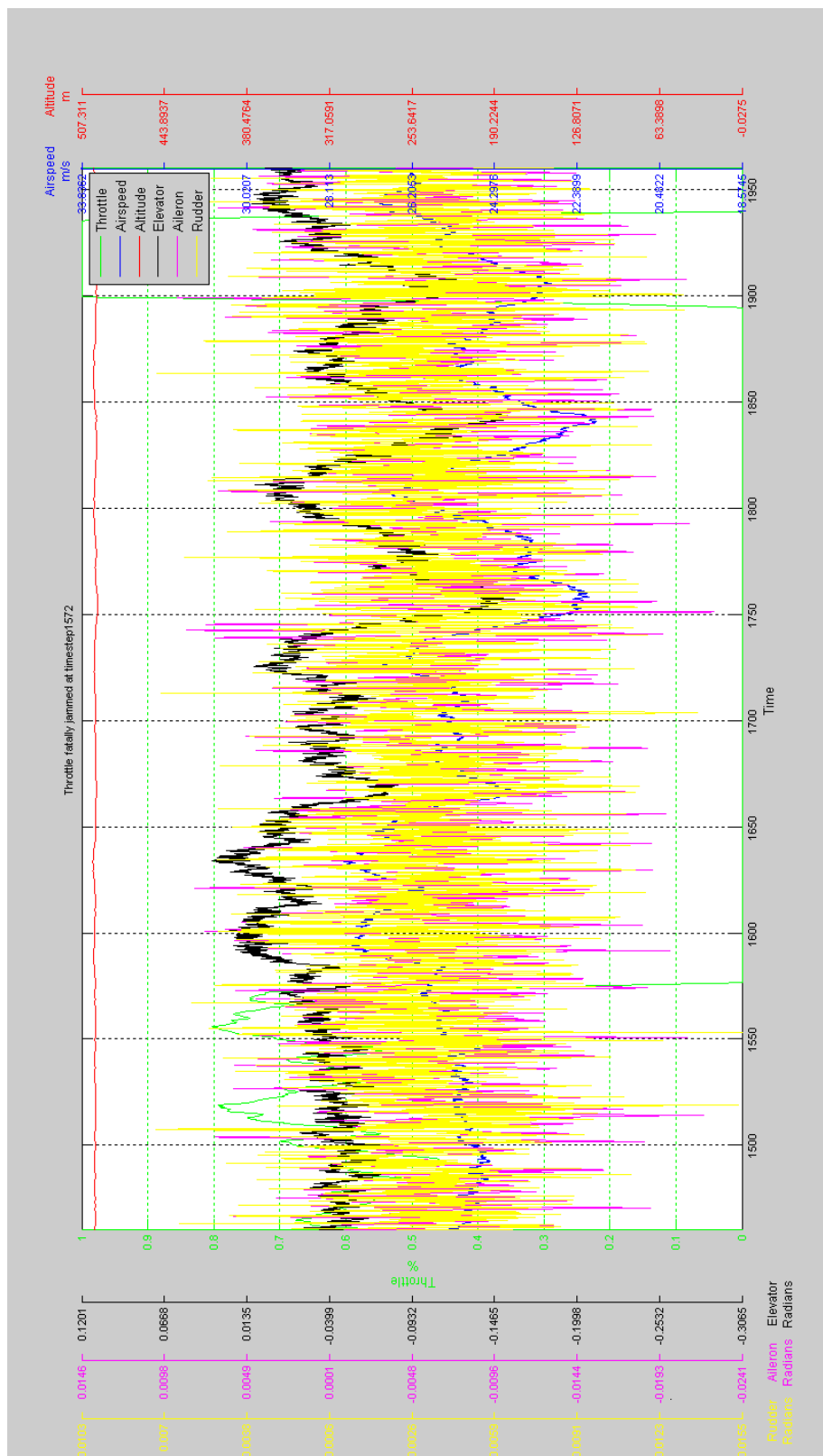


FIGURE 5.8: Sim4894 Flight Data 1 of 4 displayed in an overlaid format. This Figure shows the aircraft behaviour during fault onset.

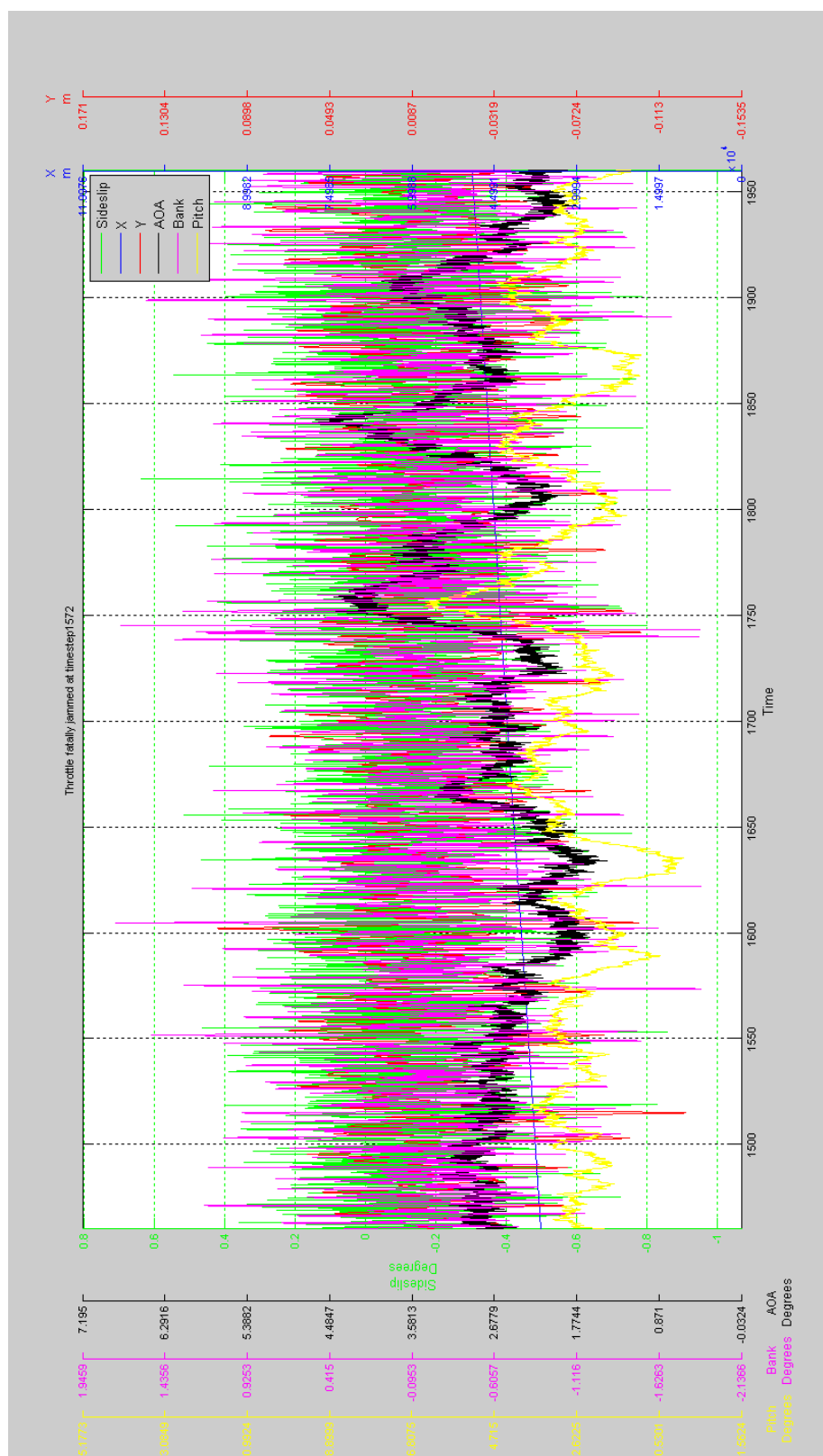


FIGURE 5.9: Sim4894 Flight Data 2 of 4 displayed in an overlaid format. This Figure shows the aircraft behaviour during fault onset.

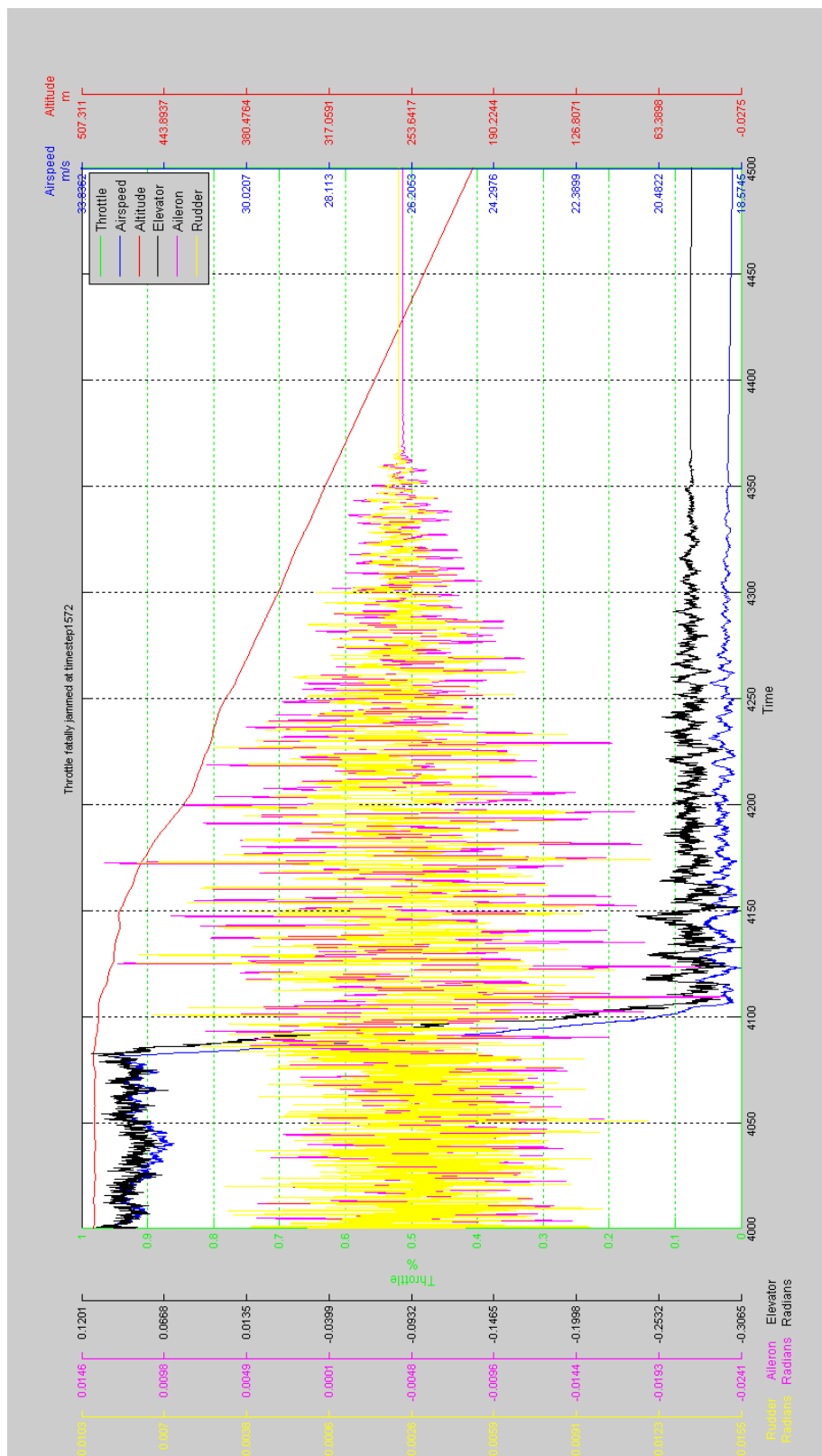


FIGURE 5.10: Sim4894 Flight Data 3 of 4 displayed in an overlaid format. This Figure shows the final control departure of the aircraft and the tumbling for a certain period before the aircraft stabilizes but continues descending and crashes.

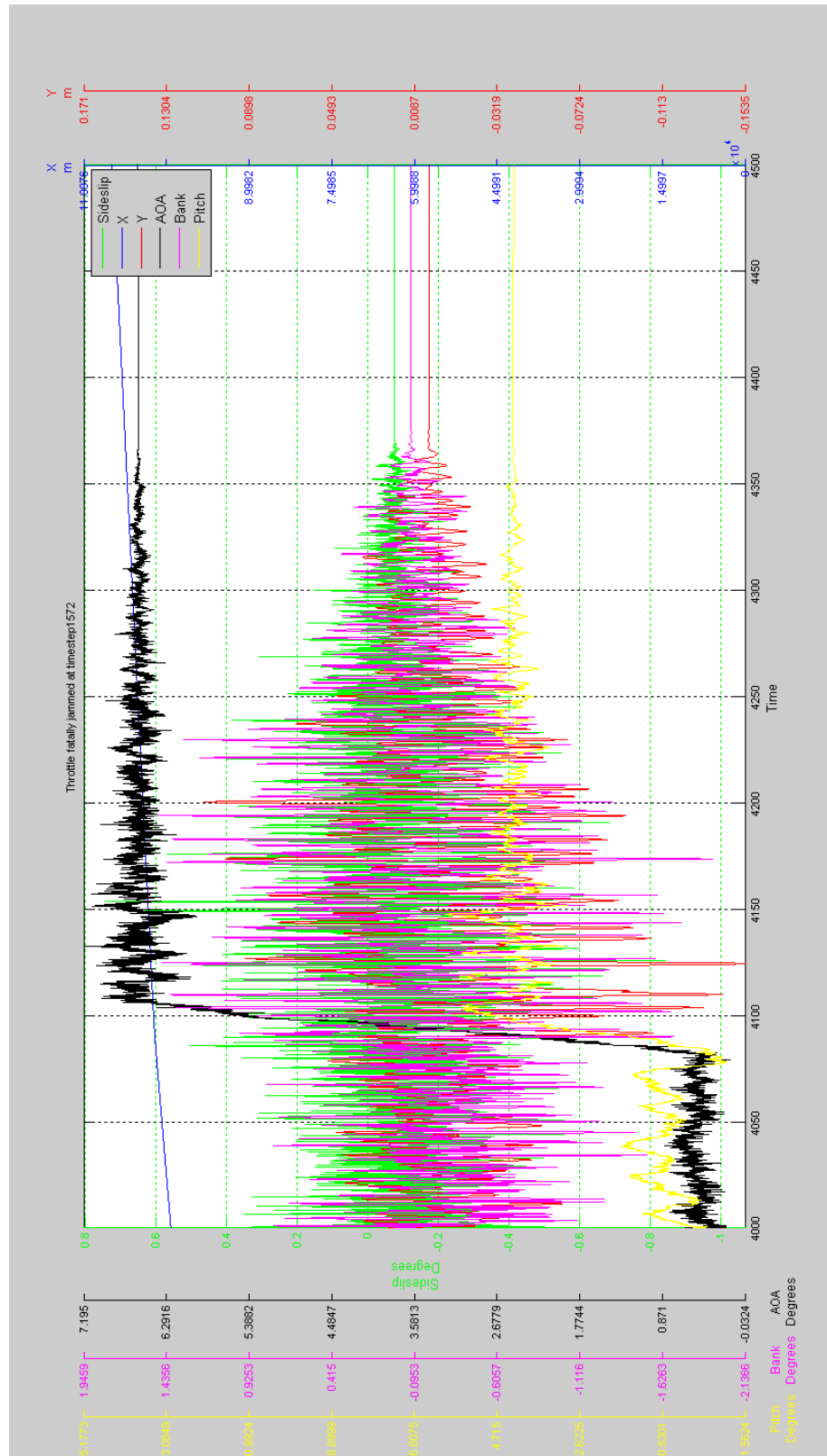


FIGURE 5.11: Sim4894 Flight Data 4 of 4 displayed in an overlaid format. This Figure shows the final control departure of the aircraft and the tumbling for a certain period before the aircraft stabilizes but continues descending and crashes.

aircraft stalls again at time 4050 seconds as seen in Figure 5.10 and is not able to recover. A potential identifying feature would be to incorporate an alert rule which triggered if the throttle was commanded to zero percent for an extended period of time. AOA levels outside of norms might also be used to attempt to warn the remote pilot of the need for additional supervision.

Solution

In this situation the likely effective solution would be to take remote manual control and attempt to stabilise the aircraft using the more sophisticated reasoning and skill of a human pilot. Once this was done the handling of the aircraft could be evaluated and options for mission abort discussed. In the event that it is judged too risky to attempt to return the aircraft to a safe landing then a controlled ditching would be the preferred option.

5.8.3 Mishap4992

Mishap Overview

It can be observed in Figure 5.12 that there is a section during the flight where the aircraft oscillated while the autopilot tried to compensate for the fault. The oscillation was caused because the combined elevator and rudder of the Aerosonde was deflected at the time it jammed. The oscillation around the direction of flight can be seen in Figure 5.13 by looking at the sideslip(green trace) and bank(pink trace) angles of the aircraft. Within 150 seconds after the onset of the fault the aircraft had plummeted 245m. At this point the autopilot control signals stabilised as it tried to get the aircraft back to its assigned altitude but all this served to do was to reduce the oscillation. The aircraft continues to descend at the same rate until it impacts the ground and the simulation terminates.

Identifying Features

It can be observed in Figure 5.12 that there is a section during the flight where the aircraft oscillated while the autopilot tried to compensate for the fault. The oscillation was caused because the combined elevator(black trace) and rudder(yellow trace) of the Aerosonde was deflected at the time it jammed. The oscillation around the direction of flight can be seen in Figure 5.13 by looking at the sideslip(green trace) and bank(pink trace) angles of the aircraft. Within 150 seconds after the

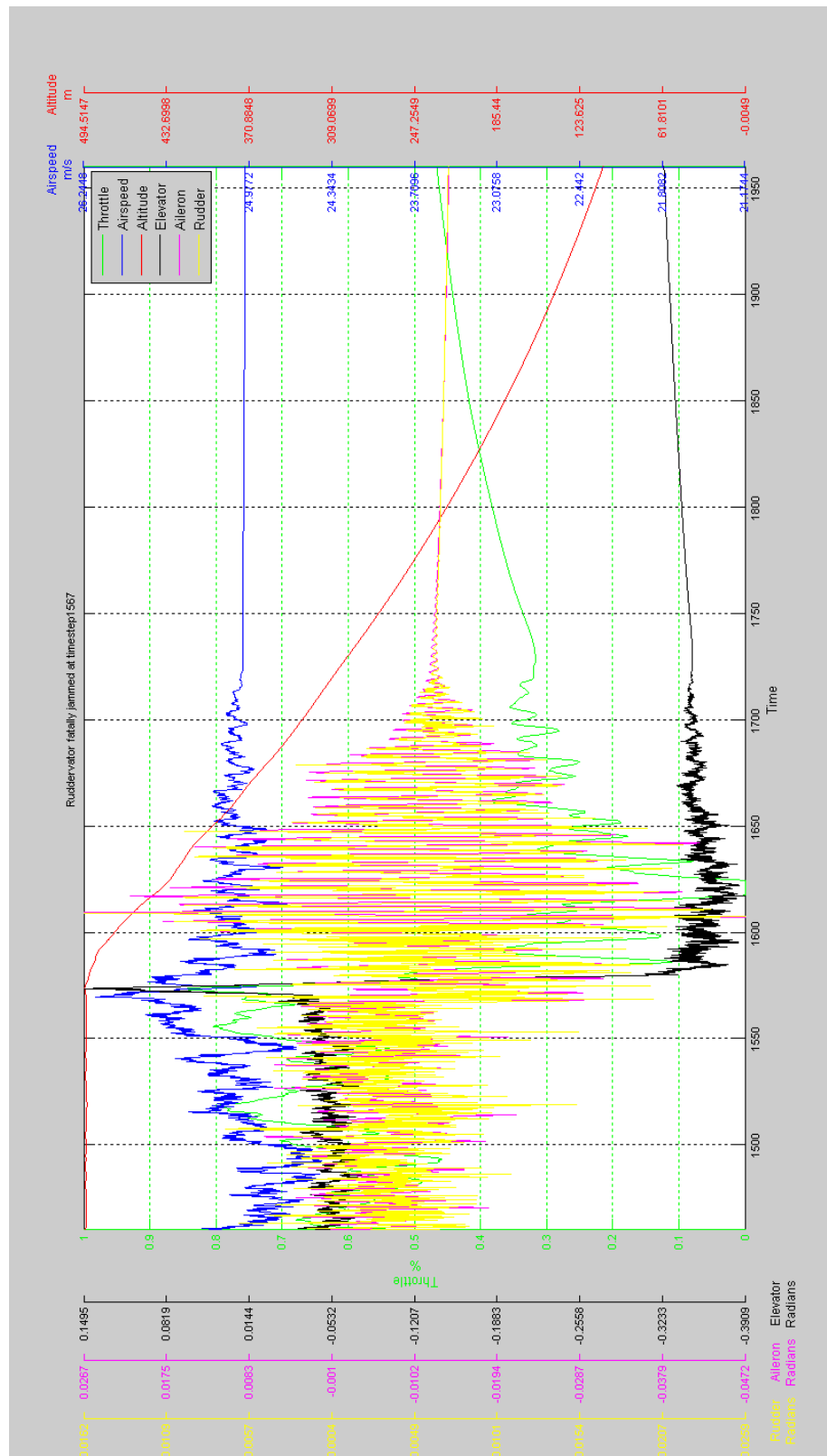


FIGURE 5.12: Sim4992 Flight Data 1 of 2 displayed in an overlaid format. The Ruddervator servo jammed at 1567s into the flight. The period shown is from 1450 seconds to 1960 seconds. The aircraft crashed shortly after this period.

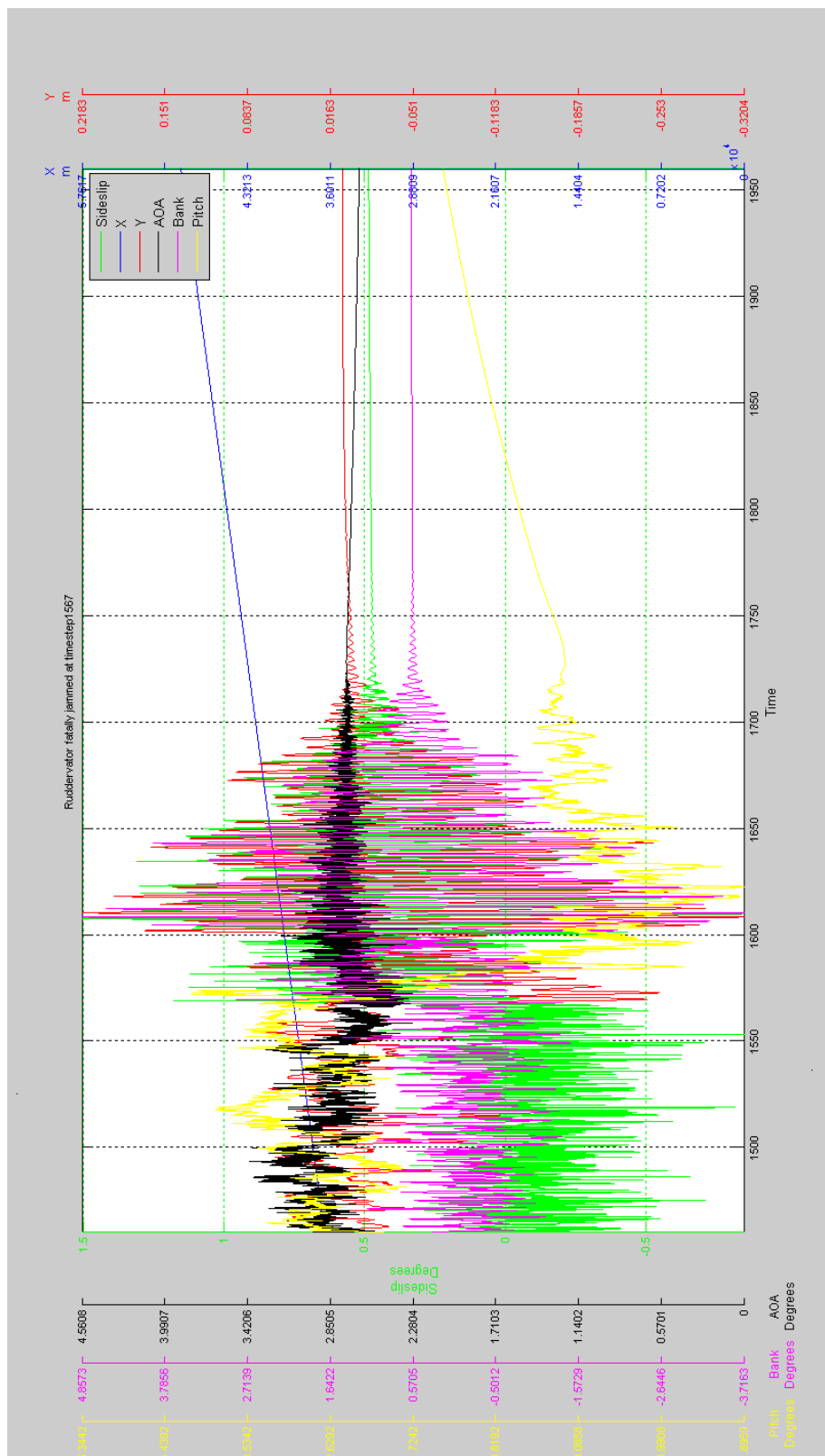


FIGURE 5.13: Sim4992 Flight Data 2 of 2 displayed in an overlaid format. The Ruddervator servo jammed at 1567s into the flight. The period shown is from 1450 seconds to 1960 seconds. The aircraft crashed shortly after this period.

onset of the fault the aircraft had plummeted 245m. At this point the autopilot control signals stabilised as it tried to get the aircraft back to its assigned altitude but all this served to do was to reduce the oscillation. The aircraft continues to descend at the same rate until it impacts the ground and the simulation terminates.

Solution

In this situation the likely effective solution would be to take remote manual control and attempt to stabilise the aircraft using the more sophisticated reasoning and skill of a human pilot. Once this was done the handling of the aircraft could be evaluated and options for mission abort discussed. In the event that it is judged too risky to attempt to return the aircraft to a safe landing then a controlled ditching would be the preferred option.

5.8.4 Mishap5080

Mishap Overview

It can be observed from the flight data of this flight (Figure 5.14 and 5.15) that at the point of fault onset the aircraft begins a sequence of rolls around its own axis. This continues until the aircraft runs out of fuel and continues its characteristic rolling as it descends and crashes as shown in Figures 5.16 and 5.17 at time 23,525 seconds.

Identifying Features

In Figure 5.14 we can see a group of regular periodic waveforms that start between time 2400 and 2450 seconds. These can be seen in all the traces. Of particular interest is the blue trace which represents the airspeed and we can see that at peak the aircraft is 12 knots faster than the cruise speed of 25 knots and that this varies about 7 knots peak to peak. The commanded throttle position is the green trace which is at 100% for the duration of this waveform so unless there is a problem with the throttle on the engine of the aircraft it seems that this large velocity change is not commanded and instead is connected to the attitude of the aircraft. It can be observed from the yellow and pink traces which represent the commanded aileron and rudder positions that at the beginning of this periodic phase they are within a more normal range and then transition to negative positions and

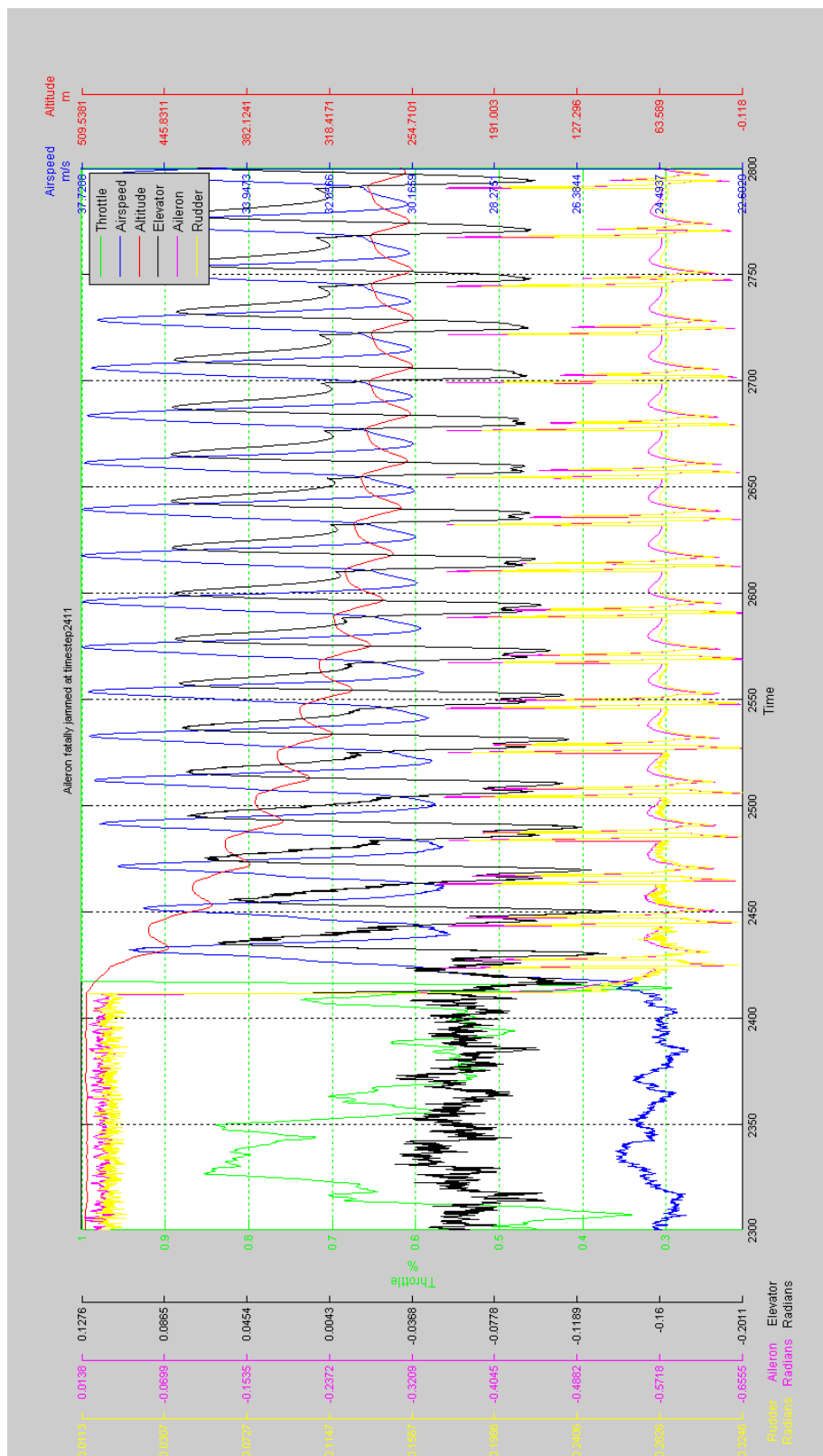


FIGURE 5.14: Sim5080 Flight Data 1 of 4 displayed in an overlaid format. The aileron servo jammed at 2411s into the flight.

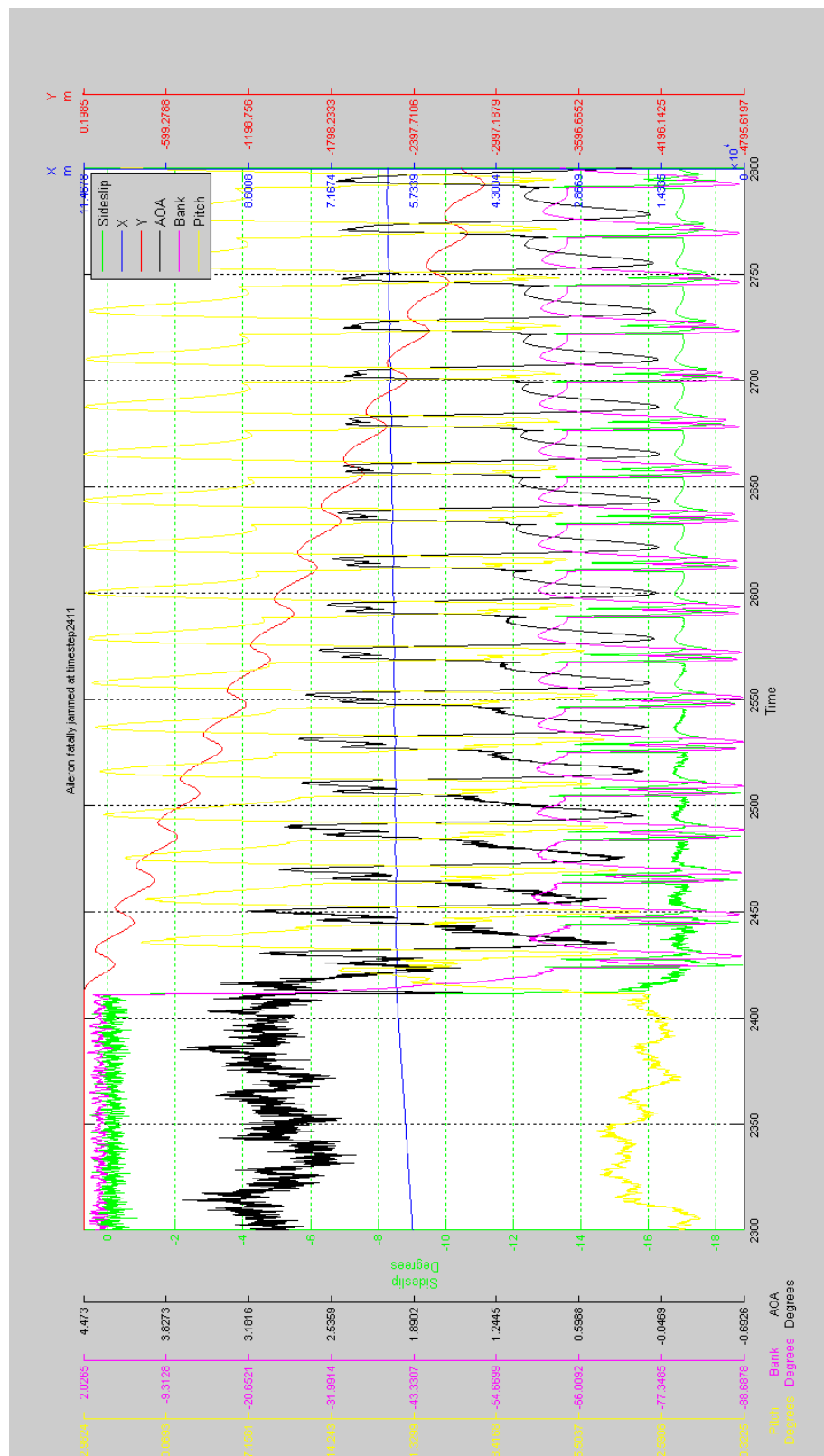


FIGURE 5.15: Sim5080 Flight Data 2 of 4 displayed in an overlaid format. The aileron servo jammed at 2411s into the flight.

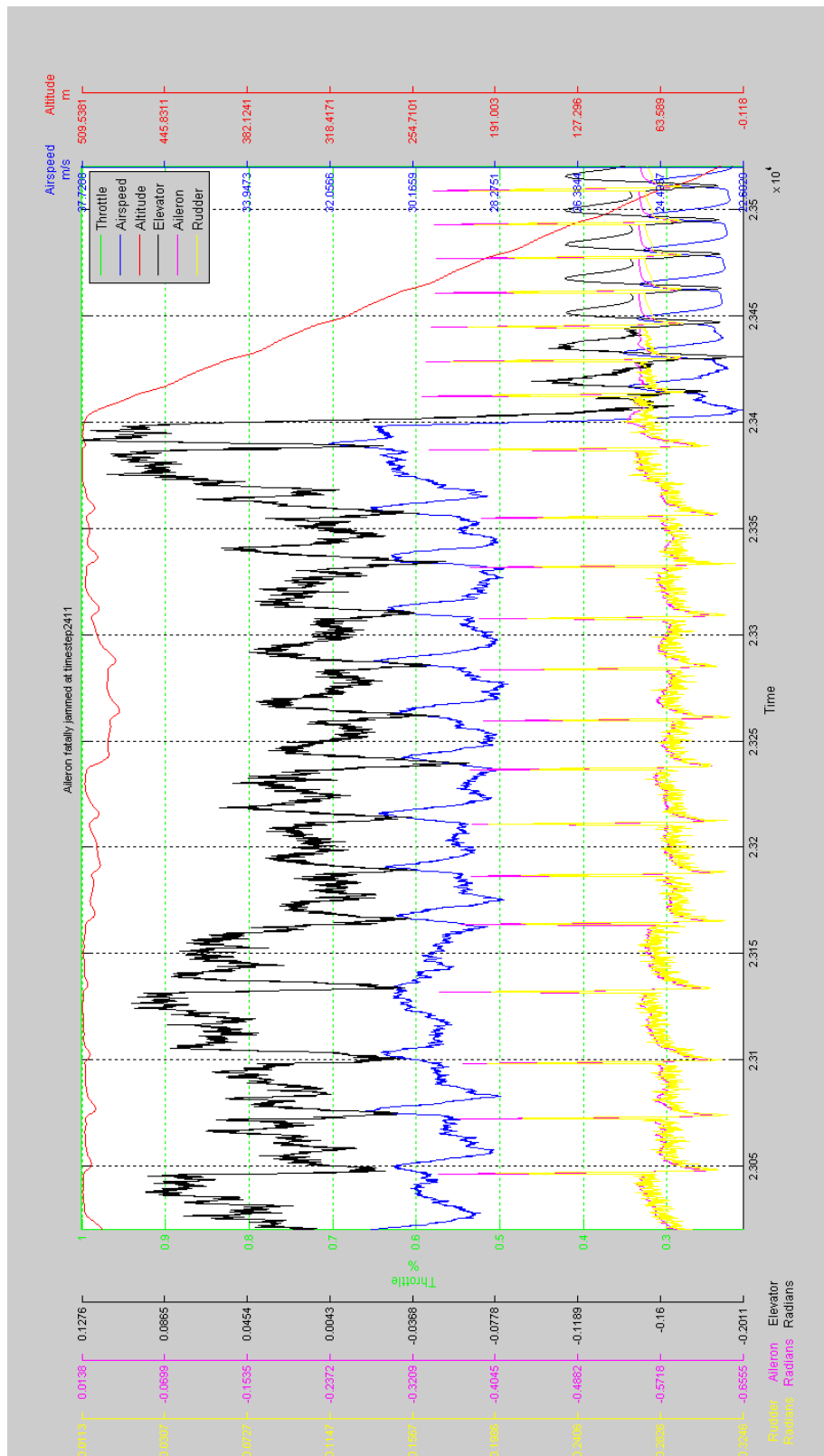


FIGURE 5.16: Sim5080 Flight Data 3 of 4 displayed in an overlaid format. The aileron servo jammed at 2411s into the flight. This Figure shows the aircraft state up to the time of the crash.

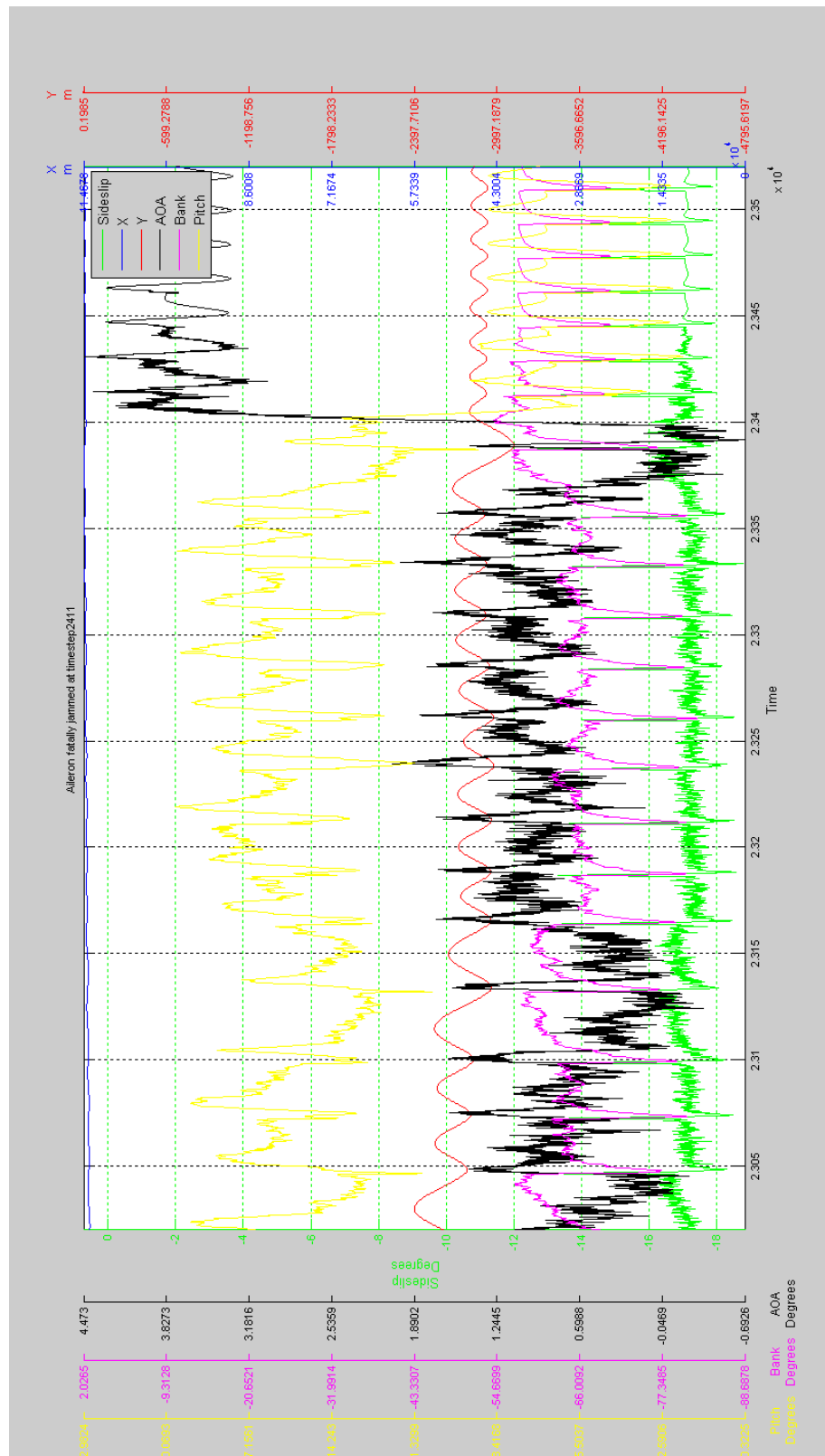


FIGURE 5.17: Sim5080 Flight Data 4 of 4 displayed in an overlaid format. The aileron servo jammed at 2411s into the flight. This Figure shows the aircraft state up to the time of the crash.

despite large commanded changes in position do not return to a more neutral position. If the corresponding state Figure(Figure 5.15) is inspected a similar set of periodic waveforms can be seen starting at the same time period. In this Figure the situation becomes clear as the pitch and bank of the aircraft(The yellow and pink traces) show that the aircraft is going through multiple rolls to starboard and deviating far from it's intended course. The cause of this is likely to be a trim problem in the roll axis of some sort for which the autopilot cannot compensate. In this situation the high angles of bank, pitch, angle of attach, large heading changes could serve to identify the case within the case base and the correct levels could be determined by a survey of the flight data of the fleet to find how frequently similar levels are observed. The next two Figures 5.16 and 5.17 show the aircraft finally lose altitude quickly following a loss of airspeed. Given the flight duration it can be surmised that the aircraft ran out of fuel. Following the loss of thrust it then descends rapidly until it finally crashes.

Solution

In this situation the likely effective solution would be to take remote manual control and attempt to stabilise the aircraft using the more sophisticated reasoning and skill of a human pilot. Once this was done the handling of the aircraft could be evaluated and options for mission abort discussed. In the event that it is judged too risky to attempt to return the aircraft to a safe landing then a controlled ditching would be the preferred option.

5.9 Dataset Creation

Using the database of normal and mishap flights created from the simulation process. a dataset was created using a sampling process to create a set representative of the data from which it was drawn while optimizing for size. A self(no mishap) dataset was created from a random sampling of the normal(no fault) flights within the database. These were then normalised into a unit interval using Algorithm 3. The normal flights dataset(also referred to as the self dataset) was then given a class label of self and combined with the mishap dataset into one Table. This matlab Table object was then used to train both the AIS-FOQA Algorithm(See Section 6.4) NN-S-FBEC Algorithm(See Section 6.6). The total proportion of self

to non-self tuples is given and Table 5.2 shows the total number of tuples in each category.

Mission Statistics

- The total number of flights simulated: 4485
- The total number of fault flights: 977
- Average duration of all flights: 11 hours and 22 minutes
- Number of flights which ended in a crash: 205

TABLE 5.2: Details of the data used during the training and evaluation of the AIS Algorithm

Class Name	Fault Details	Tuple Qty
Self(Non-Mishap) sub-total		546,000
Various	Self data populated by random sampling from Self(Non-Mishap) flights. The total number of flights from which samples were taken was over 1000.	546,000
Non-Self(Mishap) sub-total		357,765
Mishap4840	Serious Fuel Leak 2kg/hr mean	51,744
Mishap4894	Throttle Control Jam	47,727
Mishap4992	Ruddervator Control Jam	23,047
Mishap5080	Aileron Control Jam	235,247

The total ratio of self/non-self tuples

- Non-Self(Mishap) tuples 37.097%
- Self(Non-Mishap) tuples 62.903%

5.10 Summary

An experiment was designed to replicate learning from the logged flight data of a fleet of aircraft. An aircraft model was selected and modified to suit repeated mission simulations with the presence of occasional faults calculated to be of a

sufficient level to cause a departure from normal flight ending with an unplanned hard landing. The aircraft in each case was only able to use its autopilot to attempt recovery. This is consistent with some of the class A mishaps encountered by the USAF as satellite communications were typically disrupted by loss of power or unplanned manoeuvres. The data generated from the repeated normal and mishap flights was entered into a tracking database and used to create a dataset for learning. The mishap cases created were discussed and their characteristics highlighted.

Chapter 6

Algorithm Design

6.1 Introduction

In this section design of an Algorithm to meet the requirements laid out in Chapter 4.5 will be discussed as well as how it can be tested, and it's characteristics validated prior to its evaluation. Some different methods that were investigated but not used will also be discussed as well as an explanation of why they were not used for this study.

- AIS Design Elements in Section 6.2 as illustrated in Figure 6.1
 - AIS Application Domain in Section 6.2.1
 - AIS Immune Entities in Section 6.2.2
 - AIS Representation in Section 6.2.3
 - AIS Affinity Measures in Section 6.2.4
 - AIS Immune Algorithms in section 6.2.5
- v1 Algorithm in Section 6.3
- v2 Algorithm in Section 6.4
- The Fisher Iris data will be discussed in Section 6.5 as a tool for validation of the Algorithm and as a test set for the AIS Algorithm.
- The Neural Network used as a comparison will be examined in Section 6.6

6.2 AIS Design elements

The design method for AIS identified by Dasgupta et al[154] and shown in Figure 6.1 will be followed to create an AIS for case based reasoning from unmanned aircraft flight data.

Example

To help communicate the application of the different steps two points in a notional feature space shall be used to illustrate how the algorithm works.

- Aircraft situation A where airspeed is 17 knots and its altitude is 200 metres
- Aircraft situation B where airspeed is 35 knots and its altitude is 100 metres
- Aircraft situation B where airspeed is 30 knots and its altitude is 250 metres

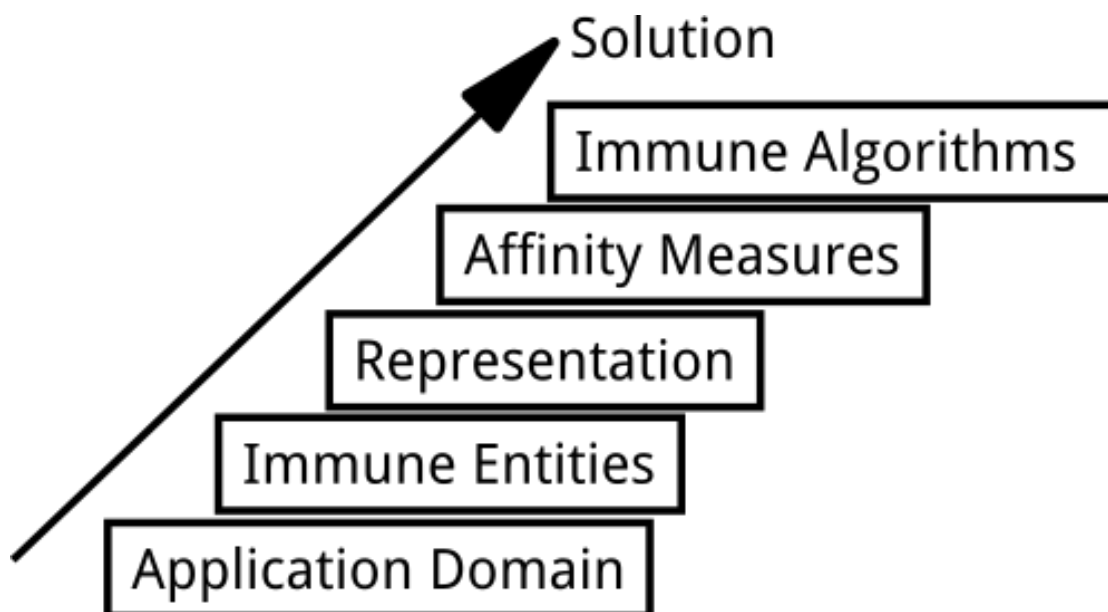


FIGURE 6.1: AIS Design Method[154]

6.2.1 Application Domain

The domain the Algorithm will work within is expressed in the descriptions of the system elements and data in section 2.3.2.1. The development and analysis of UA flight data is also discussed in section 2.3.2. Some of the difficulties of operating UAS and problems unique to UAS are identified in Chapter 2. The data to be used consists of a number of variables which vary overtime during the flight. The state of all these variables at any point during flight can be represented as a tuple of length n . Each variable at that time being represented by an element of the tuple.

Example

- Aircraft situation A as a tuple is (17, 200)
- Aircraft situation B as a tuple is (35, 100)
- Aircraft situation C as a tuple is (30, 250)

Using the scaling method from Algorithm 3 and the minimum and maximum flight data values from Table 5.1 the tuples are then scaled into a unit interval to reduce the distorting effects of different variables having different units and magnitudes.

- Aircraft situation A as a scaled tuple is (0.026534, 0.315499)
- Aircraft situation B as a scaled tuple is (0.87185, 0.157749)
- Aircraft situation C as a scaled tuple is (0.63704, 0.394374)

6.2.2 Immune Entities

The entities within the domain are the data classified as self and the various classes of non-self data. Some approaches to representation have used different ways to represent self and non self data. One in particular has been the use of hyperspheres and other methods to represent self data regions.

Example

- Training Data
 - Aircraft situation A is associated with a stall and so is classified non-self and fault
 - Aircraft situation B is associated with normal flight and so is classified as self and non-fault
- New Data
 - Aircraft situation C is a tuple yet to be classified

6.2.3 Representation

An antibody class was developed which is used by both the v1 and v2 AIS-FOQA Algorithms 8(page 144) and 9(page 145) The antibody was developed to have a number of useful features. These include the time the antibody was generated so its age could be calculated as part of the apoptosis process. The schema consisting of a string of real valued variables scaled into a unit interval using their individual minimum and maximum values. The names of the schema variables are also stored within the antibody class. The affinity measurement type is stored so that multiple affinity measures could be used if desired. The hypersphere method was chosen for all current work. The threshold of the recognition region is stored within the class. The performance of the antibody is stored within the class object in the form of the AUROC (Area Under the Receiver Operating Characteristic curve) metric. The affinity is captured in the form of the number of correct classifications performed by the antibody for the class of mishap to which it is associated. The mishap information is stored within the antibody as a reference to link to the mishap details during any alert. Functionality was also included to add information on the correct mitigation action in response to the mishap where known although this was not included during the experiment. In addition the minimum and maximum values for all the elements of the schema are stored so that whenever the antibody is evaluated it is clear how the antibody maps into real valued space from the unit scaled interval within which it is positioned.

a priori data

This antibody enables the use of masking techniques to create antibodies which only measure affinity using a subset of the total number of features. This can be used to introduce known fault cases from the design and test phase of the UAS. This could be used to add an antibody which would alert when the aircraft is within a certain range of the stall speed. This class of a priori antibodies can be given different settings for apoptosis to ensure they persist and could also be used with special alerting rules to ensure this different class do not interfere with the active antibodies being generated. This class represent an opportunity to define regions of warning that can either have antibodies within them or exclude antibodies from being generated within them. The decision would rest upon the long term effects of a priori antibodies which has yet to be determined.

6.2.4 Affinity Measures

6.2.4.1 Real Vector Representation

$$dist(x, a) = \left(\sum_{i=1}^n |x_i - a_i|^\lambda \right)^{1/\lambda}$$

Algorithm 4: Equation for Minkowski distance d between two n dimensional points x and a . When λ is 2 the distance is the Euclidean distance. When λ is 1 it is the Manhattan distance which is obtained.

$$V(n, r) = r^n \cdot \frac{\pi^{\frac{n}{2}}}{\Gamma(\frac{n}{2} + 1)}$$

Algorithm 5: Equation for calculating volume of n dimensional hypersphere and radius r [162]

Poor classification results have been reported due to the effect of n increasing. Stibor et al suggested that hyperspheres should not be used for n of greater than 41 [162, 189]. This is due to the volume of the hypersphere becoming smaller exponentially as the number of dimensions n increases. The formula to calculate the volume is given in Algorithm 5 The number of variables tracked shown in table 5.1 is seventeen which falls well short of this suggested limit. The representation in Matlab scales the data into the interval zero to one. Antibodies could be created outside that range through mutation but it is difficult to assess how far outside the range they could be and so to have a simple low bound on complexity we will

assume all antibodies will have positions within the n dimensional space bounded by zero and one. The smallest increment between two values in Matlab for a double is found by using the “eps” function and is 2.2204×10^{-16} . If the reciprocal of this is taken the result there can be 4.50369×10^{15} possible values within the range zero to one for each element of the antibody schema. The number of elements in the schema is seventeen as shown in Table 5.1. To evaluate how many unique antibodies are possible the binomial theorem should be used to evaluate the number of possible combinations. However finding the factorial of a sixteen digit number was evaluated as not feasible and is not likely to tell us much beyond $\binom{n}{k}$ being a very large number indeed. This means the potential solution space in this representation is very large as well and increasing the the number of possible solutions by allowing variable length schema increases that solution space even further. This is without considering the algorithm complexity increase that would be required to enable it to cope with variable length antibody representation schema. These considerations drove the decision to choose a fixed length schema and consider variable length schema for a future version of the algorithm following initial work. This and other future items is discussed in Chapter 8.3.

6.2.4.2 Affinity Metrics

There were several affinity metrics under consideration for use in the AIS-FOQA algorithm. There are four main options for real valued data and they are discussed here. They are in two general categories. Position based matching where the distance between two points is used as the metric and Rule based where the truth of various rules are evaluated

Hyper Sphere

Hyper spheres are sphere of known radius in an n dimensional euclidean space where n is greater than three. If the center of the hypersphere and its radius is known then any point can be evaluated to check whether it is within the hypersphere by calculating the euclidean distance between it and the center point of the hypersphere. The equation for euclidean distance is given in Algorithm 4.

Hyper Cube

Hyper cube is a cube in n dimensional euclidean space where n is larger than three.

The distance from the center to the boundary is used to determine membership of the classifier class set in a similar way to the hypersphere. An important difference is that the volume of the hypercube does not decrease as n increases. An n dimensional hypercube and hypersphere both of the same radius and the same origin would have very different volumes and boundaries.

Manhattan Distance

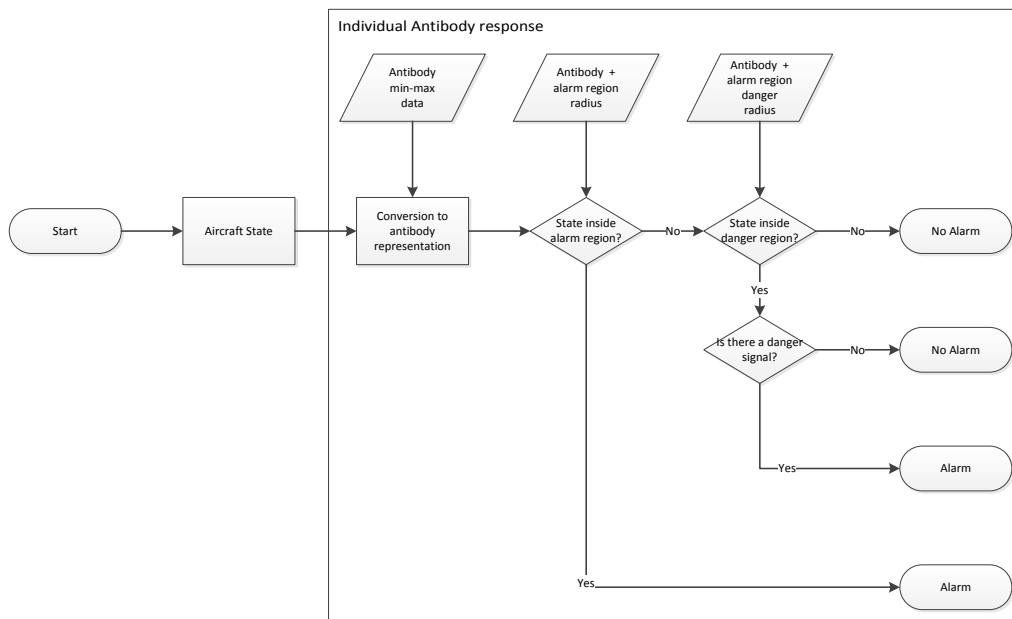
Manhattan distance is a simplified euclidean distance metric which can be obtained from the Minkowski distance equation by setting λ equal to one. Algorithm 4 shows the method by which the euclidean distance is calculated. It is called the Manhattan distance because conceptually if the dimensions are visualised as streets in a city map and intersections as points of interest the euclidean distance is obtained by taking distance of the straight crow flies path between two intersections. The Manhattan distance uses the distance travelled along the streets to find the distance from one intersection to another.

Rule String Matching

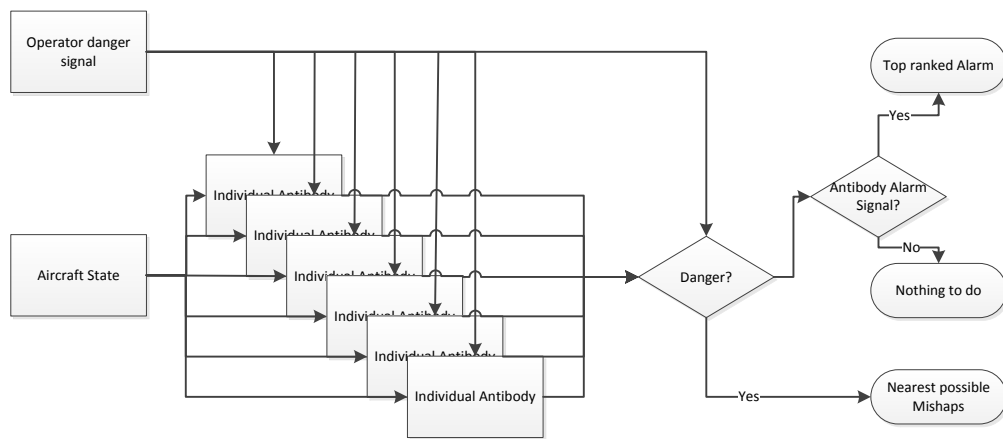
Rule string matching attempts to replicate the epitope matching of a real antibody by logical evaluation of a series of rules. The output of all the rules in the schema yields a string of true and false values which can then be rated as a measure of the affinity between the antibody and the data being evaluated. If a substring within the string s evaluated as true for all elements this is given a higher weight than a string with the same number of true values but which does not have a unbroken string of true values. This attempts to match the effect of multiple neighbouring epitopes in a real antibody having a reinforcing effect on the bond between the antibody and the antigen. The use of large numbers of weak matches gives a strong affinity signal[109].

6.2.4.3 Recognition Threshold

Recognition Threshold is the boundary in the metric between a datum being classified as part of that antibodies antigen class or not of the antigen class. In the hyper sphere affinity metric this is the surface of the hyper sphere in euclidean space. If a datum is found within this boundary it is classified as belong to the



(a) Antibody Response with Danger Signal



(b) Multiple Antibody Response

FIGURE 6.2: Figure 6.2(a) illustrates the potential responses of a single antibody to a situation. Figure 6.2(b) shows the response of multiple antibodies to a situation.

antibody class. The closer the data point(tuple) is to the centre of the antibody the higher the affinity(or match strength) is judged to be. The equation to determine excess threshold and thus affinity can be seen in Algorithm 6. Using a method which classifies and also returns a metric of by how much the classified object exceeded the classification threshold allows ranking of matches and the use of the ROC curve to evaluate the classifier[158]. This is a great benefit with affinity based methods such as the antibodies in an AIS.

$$et = r_{\text{antibody}} - d$$

Algorithm 6: Equation for Antibody Recognition affinity metric using excess threshold et , antibody radius r and distance to antigen d

Example

The tuples having been scaled into a unit interval can now be used to evaluate the affinity of situation C with situations A and B. This uses Algorithm 4.

- The distance between A and C is 0.389052
- The distance between B and C is 0.339149

On basic distance it can be seen that situation C has higher affinity with situation B than with situation A. Depending on the self radius assigned in this implementation of the algorithm Situation C could be classified as either self or non-self. Now that the distance between these tuples has been found the affinity can be calculated using Algorithm 6. Due to the small number of tuples a recognition threshold has been set as ten percent less than the distance to the other tuple set. In this case that is a distance of 0.673718 units.

- The affinity between A and C is 0.284667
- The affinity between B and C is 0.334569

As situation C is within the recognition regions of both A and B both would alert but the algorithm would judge the most relevant situation to be situation B as it has a higher affinity to C.

6.2.5 Immune Algorithms

6.2.5.1 Algorithms Elements

Clonal Maturation

Clonal maturation is the process to identify known good solutions and to explore space around them using clones of the original solution with some mutation. The exact amount of mutation can vary depending on the method used. The hyper spherical operator depicted in Figure 6.4

Negative Selection

The process by which the immune system destroyed those antibodies which would alert against self. This process is performed prior to the antibodies being released into the system. Negative selection should reduce the false positive rate by removing those antibodies which would alert on normal flight events. However this can become more difficult to implement where fault events occur during flight and are not reported so they may then create false negative results reducing the effectiveness of the Algorithm as a whole.

Mutation

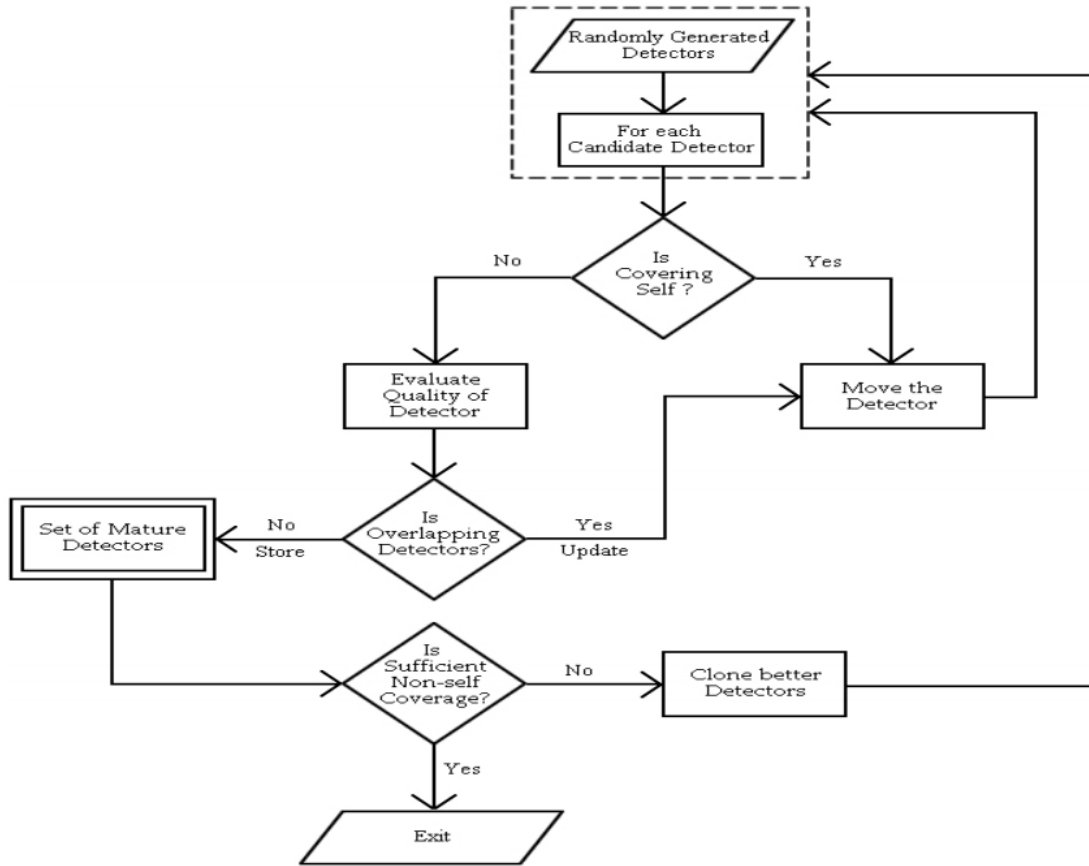


FIGURE 6.3: Real Valued Negative Selection Algorithm as shown in Dasgupta et al[154, 190]

1. $U(a, b)$ is a continuous random distribution between a and b
2. $N(c, d)$ is a normal distribution with mean c and variance d
3. Generate a random vector $X = (X_1, \dots, X_n)$ with *iid* $N(0, 1)$ components
4. Generate $R = U^{\frac{1}{n}}$, with $U \sim U(0, 1)$
5. Return $Z = R \cdot \frac{X}{\|X\|}$
6. Repeat m times to create m points.

Algorithm 7: Equation for random distribution of m points on the surface of an n dimensional sphere of radius 1.

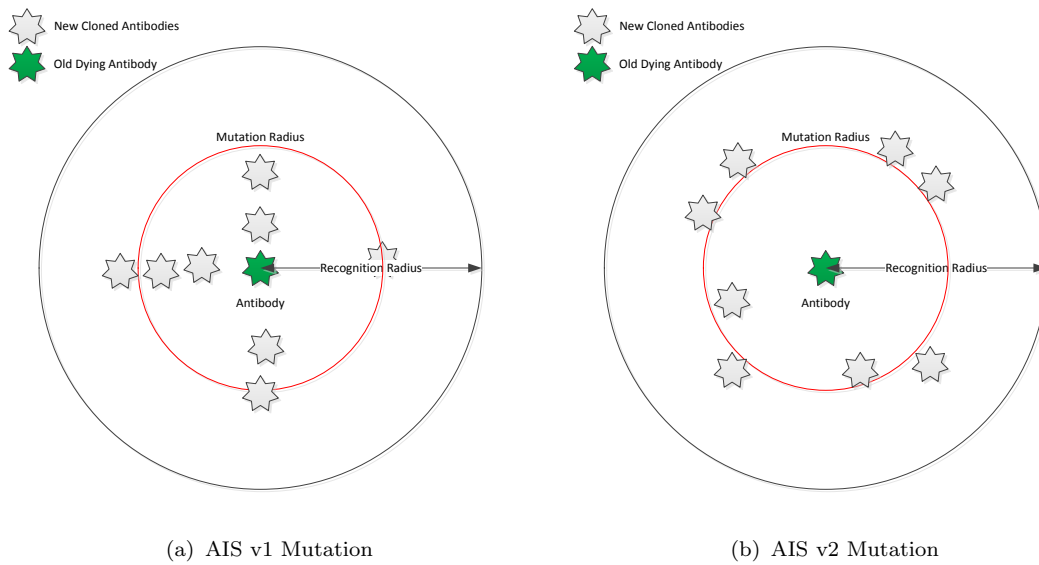


FIGURE 6.4: Figure 6.4(a) shows AIS v1 2-dimensional Mutation Illustration. Single variable mutation method. Figure 6.4(b) shows AIS v2 2-dimensional Mutation Illustration. Use of a hyper-spherical mutation operator[191, 192].

Incomplete gamma function used to smoothly mutate antibodies. Previous method as illustrated in Figure 6.4(a) modified single elements only and created “cross shaped” mutation results. This limited the number and potential variability of valid mutation solutions. Using a method to generate random uniformly dense points on an n -dimensional hypersphere allowed more variability in the result of mutations. The use of the incomplete gamma function to generate random points on the surface of a hypersphere is provided through a user defined function in Matlab[191] and discussed in Rubinstein et al and can be seen on page 69 Algorithm 2.5.4[193]. This method was therefore adopted and is depicted in Figure 6.4(b). Once the randomly distributed vectors of length 1 are created the vector origin is remapped from zero to the location of the antibody being mutated the vector magnitude scaled to match the amount of mutation required. This transformation changes the end of the vector into the location of the mutated antibody.

Apoptosis

Apoptosis is the death of a cell once it has reached the end of its useful life. This is then used as a trigger to clone the antibody which has died.

6.3 Population Type AIS (v1 Algorithm)

Initial progress was made to create an Algorithm for analysis of flight data based on the CLONALG Algorithm from DeCastro et al[117]. The Algorithm was written from scratch in Matlab to take advantage of experience there and to make use of common data structures with the simulation output. The Algorithm worked effectively at a small scale but did not scale well. As the size of the data was increased beyond multiple megabytes the time taken for the algorithm to converge extended to several days. As larger datasets were used the time taken increased and efforts were made to identify how the Algorithm could be made more efficient. The “parfor” operator in matlab is a variant of the for operator intended to allow different cycles of the for loop to be processed by different cpu cores and so increase speed through parallelisation. However this was only able to give a partial speed increase as not all the operations within the loop were compatible with the matlab “parfor” operator. Some operations needed to be performed one after the other and this limited the effectiveness of the parallelisation methods. The full matlab code for the race type AIS Algorithm 8 can be found in Appendix B of this Thesis on page 240.

	$R_{Threshold}$	Affinity threshold above which antibodies reproduce on death
	$HM_{Threshold}$	Affinity threshold above which antibodies reproduce using clonal selection
	$HMClones_{Qty}$	No of clones to be created via hyper mutation(clonal selection)
input :	$MClones_{Qty}$	No of clones to be created via normal mutation
	$NS_{Threshold}$	Affinity threshold below which antibodies will be negatively selected
	$Apoptosis_{Age}$	Age at which antibodies die
	$Problem_{set}$	Self and Non-self training data
	$AUROC_{Threshold}$	Classifier performance threshold of antibody

output: *Population* set of detectors capable of classifying patterns into self/non-self

Population CreateRandomCells(*Population*_{size}, *Problem*_{size});

while $\neg StopCondition()$ **do**

foreach $p_i \in Population$ **do**

 | Affinity(p_i)

end

*Population*_{select} $\leftarrow$ Select(*Population*, *Selection*_{size});

*Population*_{clones} $\leftarrow$ 0;

foreach $p_i \in Population_{select}$ **do**

 | *Population*_{clones} $\leftarrow$ Clone(p_i , *Clone*_{rate})

end

foreach $p_i \in Population_{clones}$ **do**

 | Hypermutate(p_i , *Mutation*_{rate});

 | Affinity(p_i)

end

Population Select(*Population*, *Problem*_{clones}, *Problem*_{size});

*Population*_{rand} $\leftarrow$ CreateRandomCells(*RandomCells*_{num});

 Replace(*Population*, *Population*_{rand});

end

return *Population*

Algorithm 8: AIS-FOQA Population Type (v1 Algorithm) CLONALG inspired[153] p272

6.4 Race Type AIS (v2 Algorithm)

The full Matlab code for the race type AIS-FOQA Algorithm 9 can be found in Appendix A of this Thesis on page 233. The details of the various thresholds discussed in Algorithm 9 can be found in table 6.2. Table 6.2 shows the settings that were used in the evaluation of Algorithm 9. The v1 and v2 Algorithms make use of vectorisation to a large extent to speed up the process of finding the affinity of antibodies. Vectorisation allows a large set of calculations to be done in batches using vector instructions on the CPU (Central Processing Unit) instead of one at a time. This is achieved through low level programming of the

	$R_{Threshold}$	Affinity threshold above which antibodies reproduce on death
	$HM_{Threshold}$	Affinity threshold above which antibodies reproduce using clonal selection
	$HMClones_{Qty}$	No of clones to be created via hyper mutation (clonal selection)
input :	$MClones_{Qty}$	No of clones to be created via normal mutation
	$NS_{Threshold}$	Affinity threshold below which antibodies will be negatively selected
	$Apoptosis_{Age}$	Age at which antibodies die
	$Problem_{set}$	Self and Non-self training data
	$AUROC_{Threshold}$	Classifier performance threshold of antibody

output: *Population* set of detectors capable of classifying patterns into self/non-self

```

while  $\neg StopCondition()$  do
  Increment age for all Antibodies;
  forall Antibodies  $\in$  Population do
    if Age >  $Apoptosis_{Age}$  then
      if Affinity >  $R_{Threshold}$  then
        if Affinity >  $HM_{Threshold}$  then
           $HMClones_{Qty}$  of Antibody to be created with mutation
        else
           $MClones_{Qty}$  of Antibody to be created with mutation
        end
      else
        Don't Reproduce;
      end
    end
  end
  Select a random Non – Self pattern in  $Problem_{set}$  which has not been used before;
  Create an antibody from the selected pattern;
  if New Antibody is within the recognition region of an existing Antibody then
    Negatively Select;
  else if Antibody alerts against Self then
    Negatively Select;
  else if Antibody  $AUROC$  below  $AUROC_{Threshold}$  then
    Negatively Select;
  else if Affinity >  $NS_{Threshold}$  then
    Add new Antibody to population;
    Negatively Select;
  end

```

Algorithm 9: AIS-FOQA Race Type (v2 Algorithm)

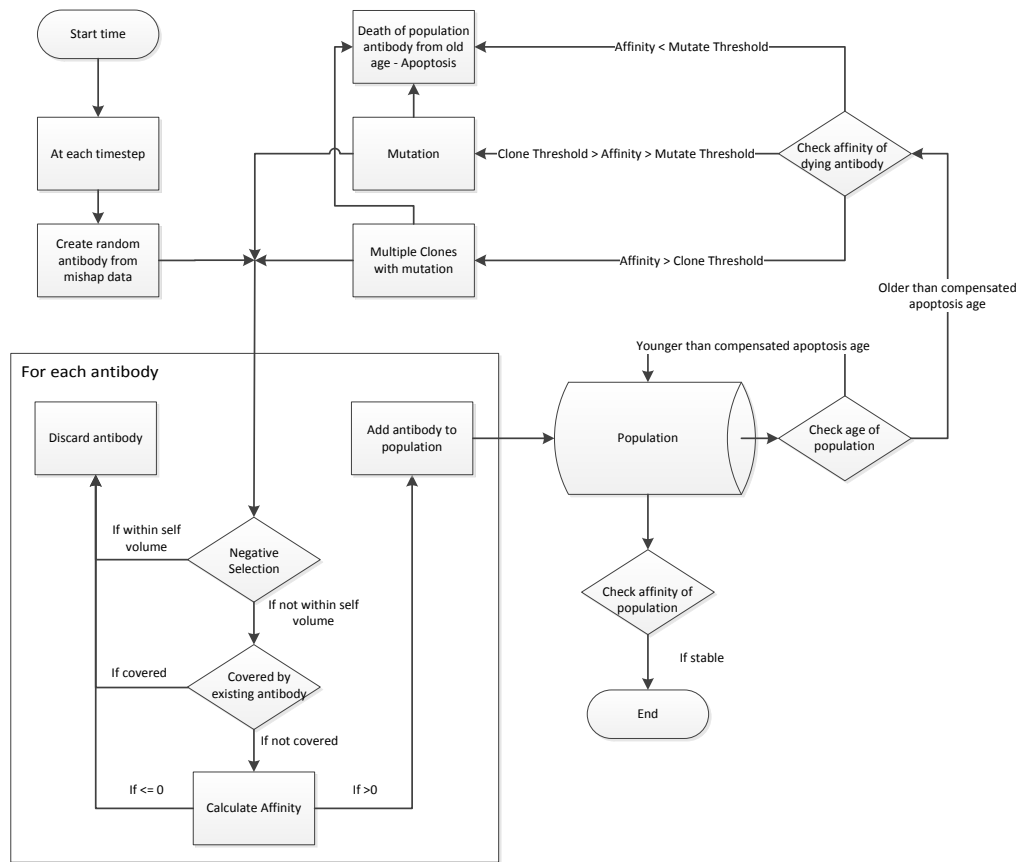


FIGURE 6.5: Race Type AIS Algorithm structure. The process shows the iterative process of creating each antibody and adding it to the population. Once in the population the antibody remains until its age is greater than a value at which time it goes through apoptosis as shown.

vectorised function being abstracted to offer a vector operation in the language being used. Typically the efficient code is in a language like C and is called from Matlab. Instead of multiple sequential calls to the distance Algorithm via a for loop or something similar all of the calls are formatted into a list(or *vector*) and passed to the Algorithm in one call. Despite this efficient process the computer being used for development was not able to handle all the distance operations for all the antibodies in the population being called at one time. It exceeded the amount of memory available to Matlab. To overcome this problem each antibody was examined sequentially and the distance of the potential antibody to each data tuple passed as a *vector* to the distance Algorithm. This process change to how antibodies are created was the main reason behind the v2 form of the Algorithm. This v2 Algorithm has a fundamentally different structure to the

v1 Algorithm even though the antibodies produced have the same structure. The sequential rather than parallel evaluation of potential antibodies allowed a desktop computer to cope with a much larger dataset than might otherwise be possible. The structuring the Algorithm to process a larger data set one antibody at a time vs a population of antibodies is a decision that needs to be balanced against computational resources available and other available processing paradigms which can add increase speed at the cost of complexity such as distributed processing[86], GPU processing[194], and FPGA processing[195].

TABLE 6.1: AIS-FOQAv2 run variable values as used during experiment to generate the results found in the Results Chapter.

Variable	Value	Description
$R_{Threshold}$	1	All antibodies were cloned upon death.
$HM_{Threshold}$	0.5	As shown in element 2 of table 6.2 This equates to a a threshold 50% of the maximum affinity of the population at that time.
$HMClonesQty$	40	As shown in element 3 of table 6.2
$MClonesQty$	2	As shown in element 4 of table 6.2
$NS_{Threshold}$	2	Affinity threshold below which antibodies will be negatively selected
$Apoptosis_{Age}$	150	Age at which antibodies die
$Problem_{set}$	-	Self and Non-self training data
$AUROC_{Threshold}$	0.65	As shown in element 1 of table 6.2. Area under the Receiver operating characteristic curve for an antibody. This is a metric to evaluate the antibody for misclassification alarms against mishap data belonging to other mishap cases other than the one the antibody was developed from.
$SelfRadius$	1×10^{-9}	This is the radius used to determine whether a potential antibody falls within self space. It is a very small number based on the distance between the two closest self data points.

TABLE 6.2: AIS-FOQAv2 Settings used to tune the AIS Algorithm.

Element	Value	Purpose
1	65	AUROC % Cutoff value below which antibodies will be negatively selected
2	0.5	Mutation Operator for clonal selection
3	40	Number of clones to be created during clonal selection
4	2	Mutation Operator for cloning

6.4.1 AIS-FOQA v2 computation complexity

The maximum computational complexity of an AIS during training is bounded in the worst case as $T(n) = O(m(3n)^2)$. As the number of input samples increase n increases the computational times required T increases by a factor of three to the power two. The letter m represents the number of iterations of the algorithm. This would be expected to be a multiple of the number of input samples n . This bounding was estimated following the procedure in Sipser[196]. This is because during each iteration of the Algorithm there will be a number of pairwise distance calculations to the data.

Euclidean Distance checks in algorithm

1. Is proposed antibody Non-Self $\rightarrow$ Pairwise distance to all self ($n - NonSelf$)
2. Is proposed antibody covered by existing antibodies $\rightarrow$ Pairwise distance to all existing antibodies (Number of existing antibodies)
3. What is the affinity of the proposed antibody? $\rightarrow$ Pairwise distance to all data in set.(n)

The number of existing antibodies as mentioned in item two above would not be known until runtime and would vary during the runtime of the algorithm. It could be a larger number than the total number of inputs but this is highly unlikely. Experience during development of the algorithm has shown that one antibody will cover all adjacent non self points until it meets another antibody or a self point and the algorithm uses number of points covered as a metric for a useful antibody.

6.5 Design Validation using the Fisher Iris Dataset

During development of the Algorithm a small multi-class dataset was used to allow faster evaluation of the Algorithm properties and any changes that might need to be made to the structure and the settings of the Algorithm. The dataset chosen is the Fisher Iris set[187] which is available through Matlab and is widely used to evaluate euclidean based classifiers. The Fisher Iris dataset consists of 150 samples of 3 classes and a total of 4 variables consistent throughout all the samples. The four measured features are sepal length, sepal width, petal length, and petal width.

- *Iris setosa*: 50 samples
- *Iris virginica*: 50 samples
- *Iris versicolor*: 50 samples

The Fisher Iris dataset is often used as of the 3 classes one is linearly separable from the other two but the final two are closely interspersed and difficult for Algorithms to correctly classify. The results from the AIS-FOQA Algorithm 9 being run on the fisher iris set can be seen in Figures 6.6 and 6.7. It can be seen that the solution was achieved relatively quickly and remained stable despite the Algorithm re-evaluating every sample several thousand times.

The effect of apoptosis age in this validation was found to alter the period of peaks in the number of antibodies. These appeared in a regular frequency which closely matched the apoptosis age. A large apoptosis age meant longer periods between peak antibody population but could also mean new improved solutions took longer to be found. Several different apoptosis ages were evaluated but were found to have little overall effect on the final stable antibody solution and may only have a computational cost effect. However evaluating this rigorously was not part of this study. This apoptosis process can be seen in Figure 6.5 and a method of compensated apoptosis age was used so that antibodies with higher affinity were allowed to reach an older age before they started the apoptosis process.

6.5.1 New data, new antibodies

To evaluate the ability of the Algorithm to handle additional data a test was made where the Algorithm was run against only two of the Fisher Iris sets and after some

antibodies had been generated the third class was added later. It was observed that while suitable antibodies were created to identify the two classes apoptosis allowed the space they occupied to be opened up and when new antibodies created they reacted to the presence of the new set by shrinking to accommodate new antibodies devoted to classifying the new set data. This is a necessary feature to support operational learning. The antibodies themselves represent cases into the flight data feature space but can also move and adjust to integrated newly discovered regions as the data to support them becomes available.

The antibodies for the Fisher Iris set can be seen visualised in Figure 6.8. The red points are the class nominated as the fault classes and the blue points are those nominated as the non fault class. The recognition regions are shown for the different antibodies as green circles. The position of these is correct for the antibody location but the size is a representation of a 4 dimensional hyper sphere radius scaled to a 2 dimensional representation and so should be used to inform the reader regarding relative size of antibodies rather than as a measure of which data points are covered by which antibody. This method of representing multidimensional antibodies within a feature space colour coded helps the reader to

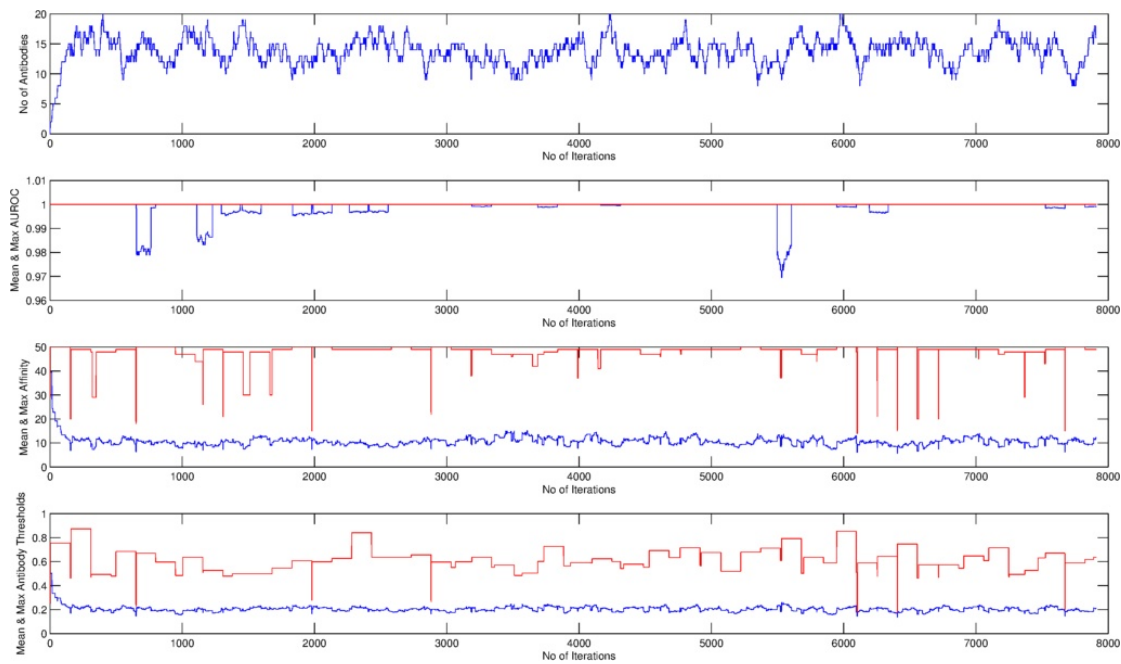


FIGURE 6.6: Results of the AIS-FOQA on a normalised version of the Fisher Iris dataset. Blue is the average for the variable of all the current antibodies in the population and Red is the Maximum value of the variable. Results through to iteration 7900

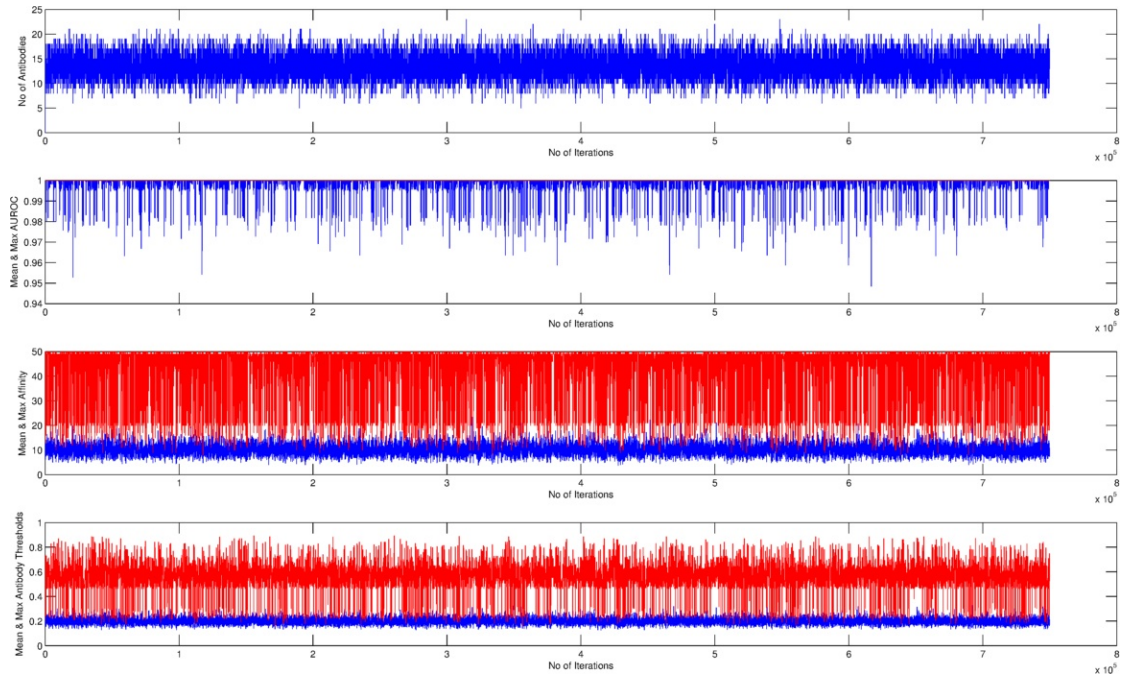


FIGURE 6.7: Results of the AIS-FOQA on a normalised version of the Fisher Iris dataset. Blue is the average for the variable of all the current antibodies in the population and Red is the Maximum value of the variable. Results through to iteration 7.5×10^5

understand the structure of the different classes in the set and the relative position of faults and the antibodies that detect them within the feature space. .

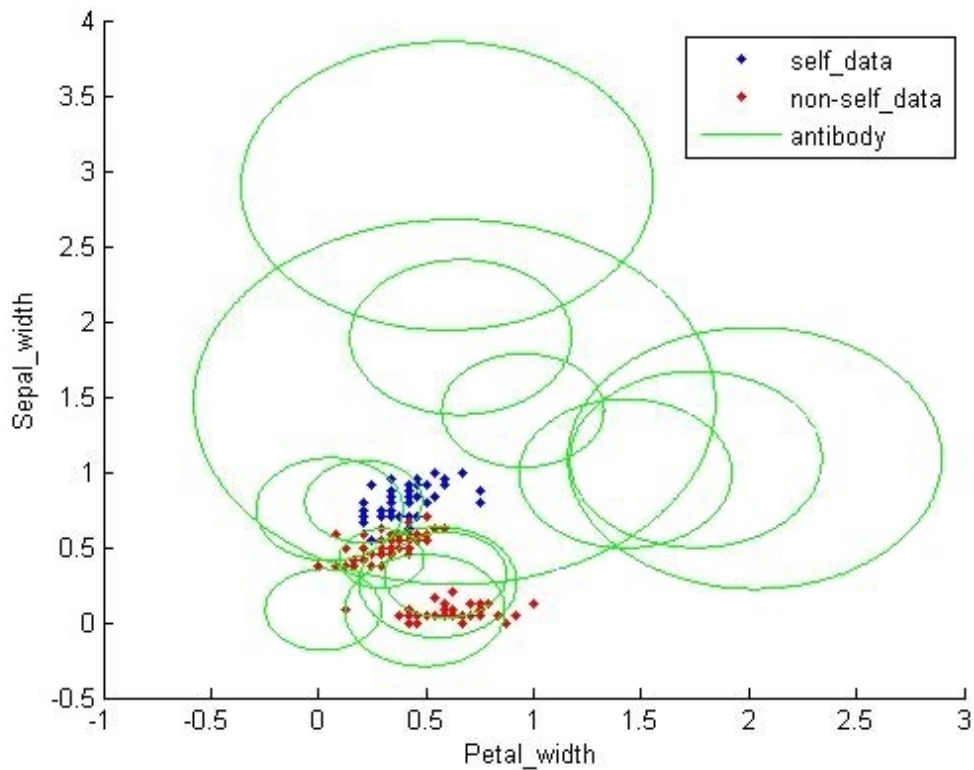


FIGURE 6.8: Results of the AIS-FOQA on a normalised version of the Fisher Iris dataset. Shown are two of the four elements used by the dataset.

6.6 Neural Network

The Neural Network toolbox within Matlab is mature and offers a large number of ways to train a Neural Network for a classification task. Neural Networks in general were briefly discussed in Section 3.3.2 but here we will discuss specifically how a Neural Network was trained for the Flight Data classification task. The Matlab Neural Network Toolbox User Guide[197] was used as a guide to correctly set up the Neural Network. The version of the Neural Network Toolbox was version 8.1 and the version of Matlab used was release 2013b. This Neural network is of the multi layer perceptron type. Each neuron uses a sigmoid function to control activation and the weights of the individual inputs to each neuron are trained via use of the feedback error correction method. Each case is identified through use of a binary output in the final layer of the neural network. The neural network was created and trained using the Matlab Neural Network Pattern Recognition tool. The Matlab “nntrainingtool” and the “nprtool” was used to train the Neural network in the same way for the fisher iris data(Section 6.5)

as for the flight data. There were ten hidden nodes used and they all used a sigmoid function as their activation function. This was the “tansig” function and the network was trained using the “trainscg” which makes use of scaled conjugate gradient back propagation. As shorthand this algorithm will be referred to as NN-S-FBEC (Neural Network - Sigmoid - Feedback Error Correction) The Neural network created was then evaluated in the same way as the AIS-FOQA algorithm both against the flight data the training set was created from and against mishap flights and non mishap flights which the algorithm had not been trained on. The flight data used with the NN algorithm is the same set used with the AIS-FOQA algorithm presented in Section 6.4. The results of the application of the Neural Network on the flight data can be found in Section 7.2 and the results of the Neural Network on the Fisher Iris data can be found in Section 6.7.2.

6.6.1 N-S-FBEC computational complexity

Computational complexity of a NN during training is on the order $T(n) = O(n^k)$ [198]. As the number of input samples increase n increases the computational times required T increases linearly. As the number of neurons k increases the computational time increases exponentially.

6.7 Fisher Iris AIS Performance

In order to compare the performance of the algorithm to more well known artificial immune system implementations and to other algorithms the algorithm was applied to the Fisher Iris data set described in Chapter 6.5.

6.7.1 AIS Fisher Iris Results

The classification performance shown in Table 6.3 amounts to only $3\frac{1}{3}\%$ of data points(5 points in total) being misclassified.

TABLE 6.3: Results from using the AIS-FOQA Algorithm against the Fisher Iris dataset

Class	Predicted		
Actual	<i>Iris setosa</i>	<i>Iris versicolor</i>	<i>Iris virginica</i>
<i>Iris setosa</i>	50	0	0
<i>Iris versicolor</i>	1	47	2
<i>Iris virginica</i>	0	2	48

TABLE 6.4: Detailed Results from using the AIS-FOQA Algorithm against the Fisher Iris dataset

Class	TPR	FPR	TNR	FNR
<i>Iris setosa</i>	98.04%	1.96%	100%	0%
<i>Iris versicolor</i>	95.92%	4.08%	97.03%	2.97%
<i>Iris virginica</i>	96%	4%	98%	2%

6.7.2 NN-S-FBEC Fisher Iris Results

The classification performance shown in Table 6.5 amounts to only 2% of data points(3 points in total) being misclassified.

TABLE 6.5: Results from using the NN Algorithm against the Fisher Iris dataset

Class	Predicted		
Actual	<i>Iris setosa</i>	<i>Iris versicolor</i>	<i>Iris virginica</i>
<i>Iris setosa</i>	50	0	0
<i>Iris versicolor</i>	0	48	2
<i>Iris virginica</i>	0	1	49

TABLE 6.6: Detailed Results from using the NN Algorithm against the Fisher Iris dataset

Class	TPR	FPR	TNR	FNR
<i>Iris setosa</i>	100%	0%	100%	0%
<i>Iris versicolor</i>	96%	2%	98%	4%
<i>Iris virginica</i>	98%	3.9%	96.1%	2%

6.7.3 Fisher Iris Comparison

The NN-S-FBEC and AIS-FOQA algorithms yield a similar performance on the Fisher Iris data set. The NN-S-FBEC algorithm is the better at 2% misclassification but the AIS-FOQA is not far behind at $3\frac{1}{3}\%$. The visualisation of the different antibodies created by AIS-FOQA shown in Figure 6.8 shows the large coverage created by AIS-FOQA. It can be seen that the recognition radius of some of the antibodies is very large relative to the scale of the values in the data itself. If more data and classes were present it would be likely that the antibodies would shrink in size to adapt. This behaviour of antibodies shrinking away from self space and so reducing misclassification as data is added is described in the discussion of V-detectors aka known as variable size detectors[148]. This demonstrates that the algorithm works as intended and is suitable to evaluate against the NN in the experiment.

6.8 Summary

The paradigms of AIS were evaluated and the v1 AIS algorithm created based on the CLONALG paradigm with some differences to incorporate clonal maturation and to incorporate vectorisation in the v2 algorithm in an attempt to reduce the computational cost of applying the Algorithm. The Algorithm was developed and evaluated using the Fisher-Iris dataset as a test set prior to use with the experimental data. It's ability to reach a stable solution was demonstrated and the effect of apoptosis shown to be one of preventing stagnancy in the antibody coverage while enabling new data and antibodies to be introduced over time.

Chapter 7

Results

7.1 Introduction

The results for the AIS-FOQA Algorithm are presented along with the results for the neural network and the two different sets compared. A series of illustrations are used to show how the classification performance of the neural network differs to the AIS and the relative advantages and disadvantages.

- NN Results on Aerosonde Data set in Section 7.2
- AIS Results on Aerosonde Data set in Section 7.3
- AIS Results on Fisher Iris Data set in Section 6.7
- Usage Comparison in Section 7.4

TABLE 7.1: Details of the Mishaps used during the training and evaluation of the AIS Algorithm

Sim	Fault Type	Fault Onset(s)	Closest AB Class	Fault tuple qtys	Non-Fault tuple qtys
Sims used during Training and Evaluation					
Sim 4840	Serious Fuel Leak 2kg/hr	1556	Mishap 4840	37,184	15,560
Sim 4894	Throttle Control Jam	1572	Mishap 4894	33,007	15,720
Sim 4992	Ruddervator Control Jam	1567	Mishap 4992	8,377	15,670
Sim 5080	Aileron Control Jam	2411	Mishap 5080	212,137	24,110
Sims used during Evaluation only					
Sim 4889	No Failure	N/A	N/A	0	320,994
Sim 4890	No Failure	N/A	N/A	0	320,994
Sim 5107	Ruddervator Control Jam	1529	Mishap 4992	305,531	15,290
Sim 4559	Serious Fuel Leak 2kg/hr	1501	Mishap 4840	31,459	15,010
Sim 4884	No Failure	N/A	N/A	0	320,994

7.2 NN-S-FBEC Flight Data Results

The construction of the Neural Network is described in Section 6.6 and the results of it's application to experiment data is shown in Table 7.2. The scenarios used in the evaluation were split between those used in the training of the algorithm and those not used in training in order to examine the ability of the algorithm to recognise unlearned features which are similar to features in scenarios it had been trained on. The results shown in Table 7.2 show the NN is effective in general at avoiding False positives. The ability of the NN to learn how to positively identify mishap data from the unsegmented flight data is disappointing and with the exception of Mishap5080 in Sim5080 it does not manage a True Positive ratio greater than 33%. This improved ability in positively identifying Mishap5080 may be due to the larger number of samples available from that flight. Table 5.2 shows there are more 183,503 more samples in the Sim5080 data than are present in any other Mishap scenario used during testing. This is due to the long time between fault onset and the final crash in that scenario. The details of the crash

TABLE 7.2: Results from using the NN-S-FBEC Algorithm against various stored flight files.

Sim File	Relevant Mishap Case	True Positive	False Negative	True Negative	False Positive
Sims used during Training and Evaluation					
Sim4840	Mishap4840	32.1%	67.8%	100%	0%
Sim4894	Mishap4894	0.006%	99.9%	100%	0%
Sim4992	Mishap4992	30.5%	69.4%	100%	0%
Sim5080	Mishap5080	87.9%	12%	100%	0%
Sims used during Evaluation only					
Sim4889	NA	0%	0%	85%	14.9%
Sim4890	NA	0%	0%	99.5%	0.4%
Sim5107	Mishap4992	6.3%	93.6%	100%	0%
Sim4559	Mishap4840	12.2%	87.7%	100%	0%
Sim4884	NA	0%	0%	96.7%	3.2%
Summary					
Totals	NA	35.7%	64.2%	94.3%	5.6%

are discussed further in Section 5.8.4 but in summary the aircraft aileron jammed and it entered into a slow roll which continued for a long time until the aircraft ran out of fuel and crashed. This long period of consistent rolling would have meant that the other control surfaces of the aircraft were in unusual positions as the autopilot tried to correct the roll which would perhaps be easier for the NN to recognise as a feature given the early onset of the fault relative to the duration of the simulation. This creates a situation where the fault segment of the flight is the a much larger proportion of the data unlike in the other scenarios where the fault segment is the closer in duration to the non-fault segment. This reality of fault/non-fault data within mishap flights are what makes it so difficult to learn from mishap data in an automated way.

To examine how the NN reacts to mishap in the time domain we can look at Figure 7.1 and Figure 7.2. Figure 7.1 shows the raw response of the NN to the data as the flight progressed and Figure 7.2 shows where the NN actually classified the flight as mishap. This is an interesting contrast to look at to understand the

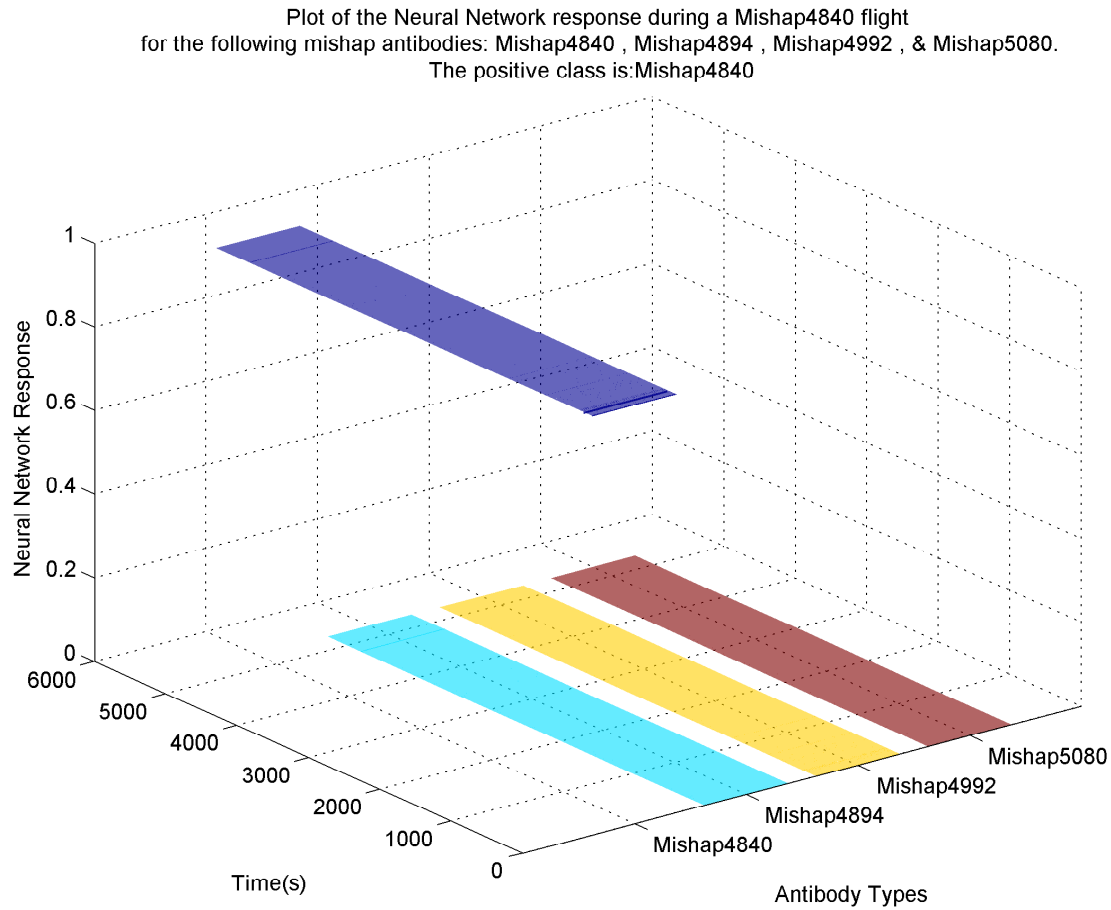


FIGURE 7.1: Sim4840 NN Response showing full neural network response from output nodes

algorithm response as it allows us to see how the algorithm responds. In Figure 7.1 we can see that the NN response is high for the entire duration of the flight but only actually alerts during a short section around the initial fault onset. This suggests that the NN is able to identify the unique attributes of this flight from other flights but is poor at actually separating the fault portions of the flight from the non fault portions of the flight. This behaviour is as expected for this type of algorithm given that it was trained as discussed in a semi-supervised way and for it to learn effectively NN requires the flight to have been segmented into fault and non fault parts not just for the flight to be identified as one within which a mishap occurred.

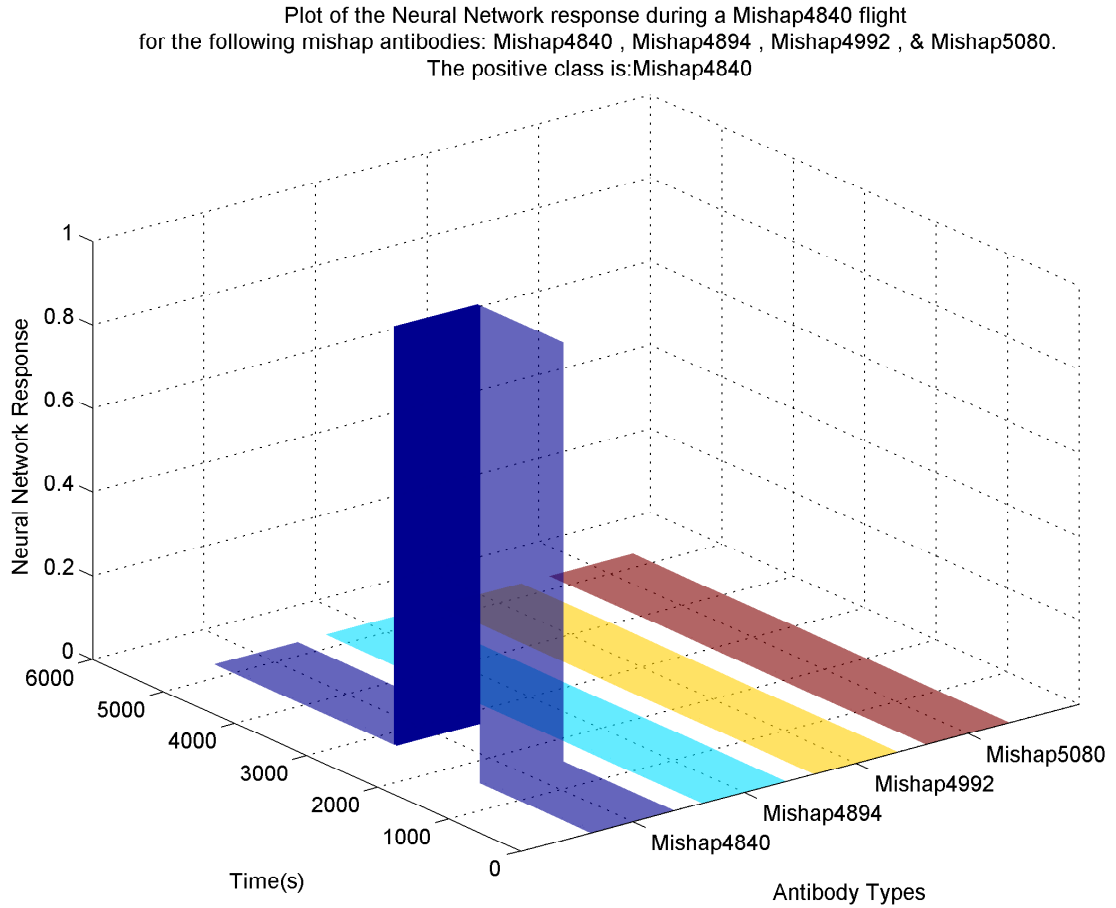


FIGURE 7.2: Sim4840 NN Response showing the final output.

7.3 AIS Flight Data Results

During the longest run of the version 2 Algorithm the average time taken to complete each iteration was around 92 seconds. The amount of time required during each iteration of the Algorithm is not stable due to the many possible actions that might be taken. If an antibody dies additional computation is required. As shown in Algorithm 9 each call to check the affinity of an antibody is a very computationally expensive process. The bounding of this process is estimated as $T(n) = O(m(3n)^2)$ and this is discussed in Section 6.4.1.

The pairwise plots are shown in Figures 7.3 to 7.9. These are an illustration of the feature space within which we see the self data coloured blue, the mishap data coloured red, and the notional location and relative size of antibodies shown in green. It should be noted that the radius of the antibodies shown is scaled in proportion to the total number of elements in the schema. The actual recognition

TABLE 7.3: AIS Algorithm learning results after 23,410 iterations. 25 days continuous run.

Class	No. of Antibodies	Max AUROC	Mean AUROC	Max sum Affinity	Mean sum Affinity	Samples in class
Mishap 4840	7	0.99971	0.59878	32,530	6,886	51,744
Mishap 4894	31	1	0.92071	9,605	1,402.5161	47,727
Mishap 4992	13	1	0.90025	1,112	195.0769	23,047
Mishap 5080	40	1	0.94873	204,566	5,431.075	235,247

radius is likely to be larger. The representation of n dimensional recognition regions into a two dimensional Figure will always be confusing and so the antibodies are shown for illustration only and to demonstrate the large range of sizes and positions. It can be seen that where the mishap created data that differs from the norm the antibodies alerting against this shown in green are large compared to the antibodies nearer to the blue data which represents data from normal flights. This shows that the variable sizing method is working as intended to maximise detection sensitivity without increasing false alarm rate so that antibodies can be located near to normal data and so create an early alert when things go wrong and also provide robust alerting when there is obvious features available. Figure 7.3 in particular is a good example of this showing as it does the relationship between elevator angle, airspeed, angle of attach and aircraft pitch for normal data(in blue) and Mishap4840 data(in red) with the antibodies for Mishap4840 shown in green.

The results of the AIS-FOQA algorithm as applied to the flight dataset can be seen in Table 7.4. In common with the results of the NN shown in Table 7.2 the scenarios used in the evaluation were split between those used in the training of the algorithm and those not used in training in order to examine the ability of the algorithm to recognise unlearned features which are similar to features in scenarios it had been trained on. In contrast to the NN results in Table 7.2 the AIS-FOQA algorithm

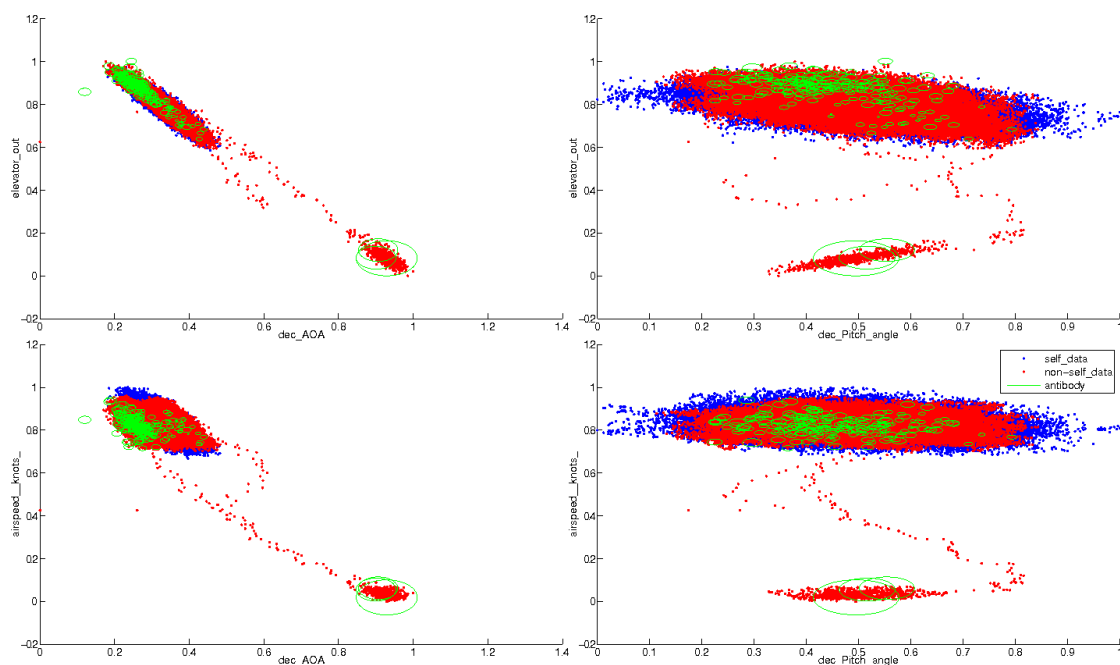


FIGURE 7.3: Pairwise Plot 1 of 7 Showing data from Mishap 4840 in red, all non-mishap flights in blue, and Mishap 4840 antibodies in Green

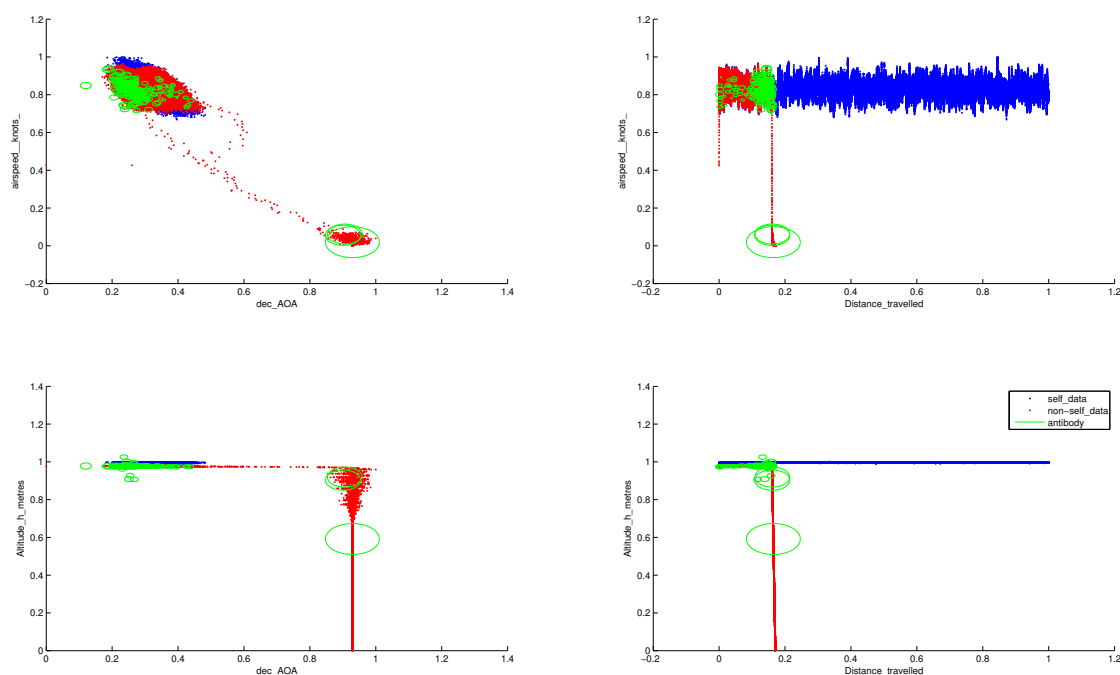


FIGURE 7.4: Pairwise Plot 2 of 7 Showing data from Mishap 4840 in red, all non-mishap flights in blue, and Mishap 4840 antibodies in Green

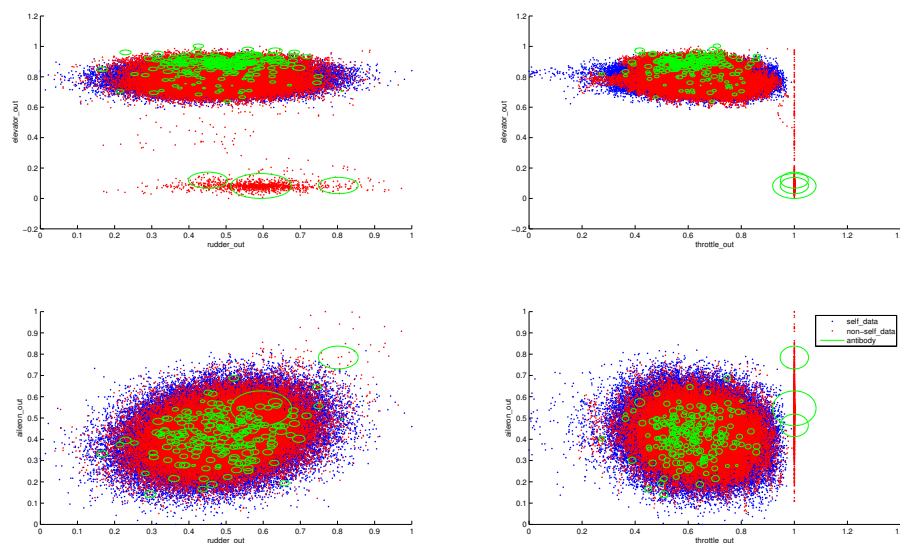


FIGURE 7.5: Pairwise Plot 3 of 7 Showing data from Mishap 4840 in red, all non-mishap flights in blue, and Mishap 4840 antibodies in Green

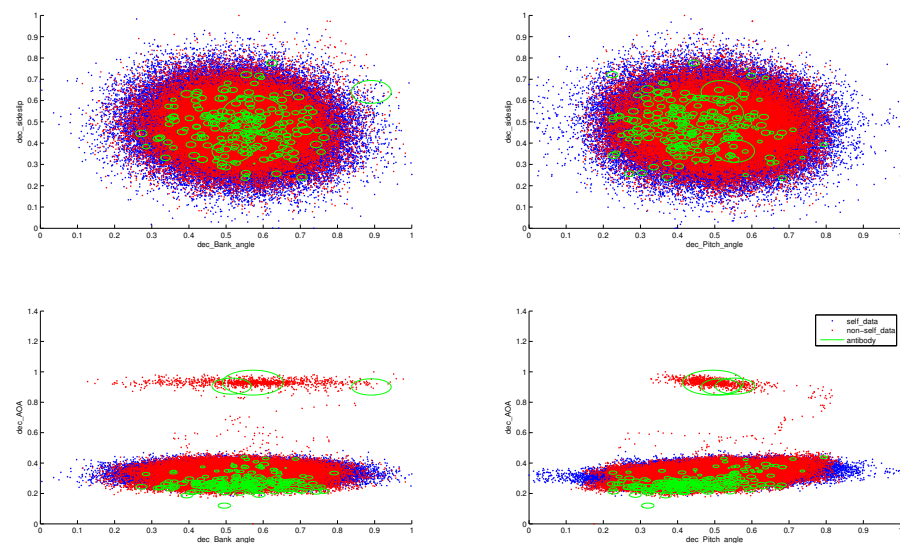


FIGURE 7.6: Pairwise Plot 4 of 7 Showing data from Mishap 4840 in red, all non-mishap flights in blue, and Mishap 4840 antibodies in Green

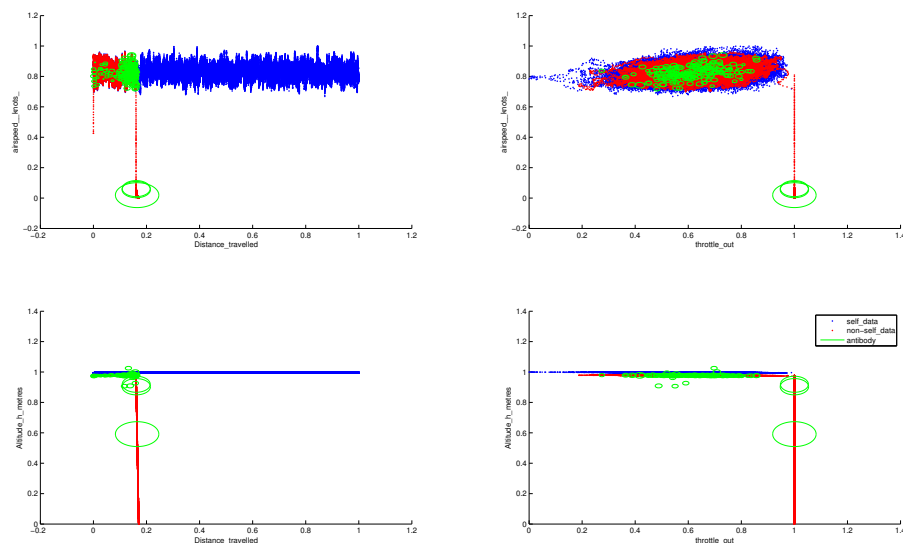


FIGURE 7.7: Pairwise Plot 5 of 7 Showing data from Mishap 4840 in red, all non-mishap flights in blue, and Mishap 4840 antibodies in Green (Airspeed, Altitude, AOA, and Distance travelled)

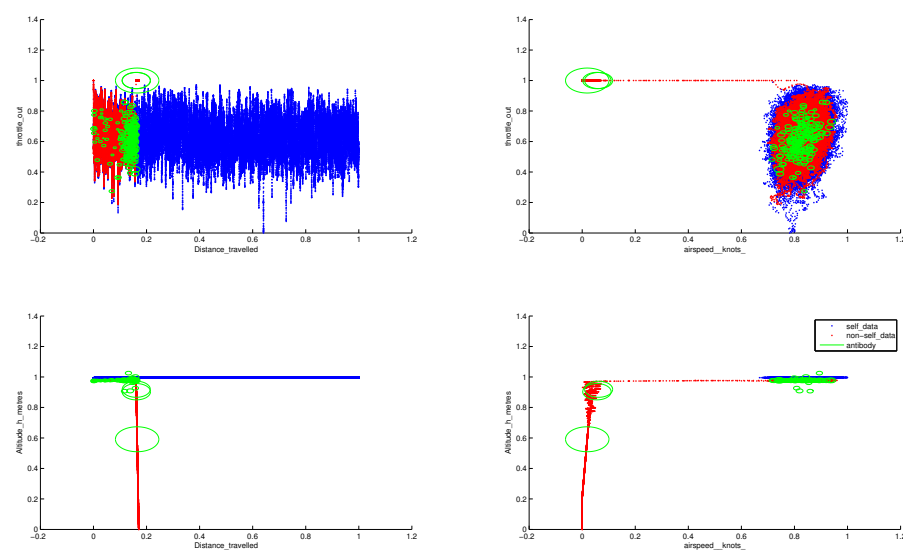


FIGURE 7.8: Pairwise Plot 6 of 7 Showing data from Mishap 4840 in red, all non-mishap flights in blue, and Mishap 4840 antibodies in Green

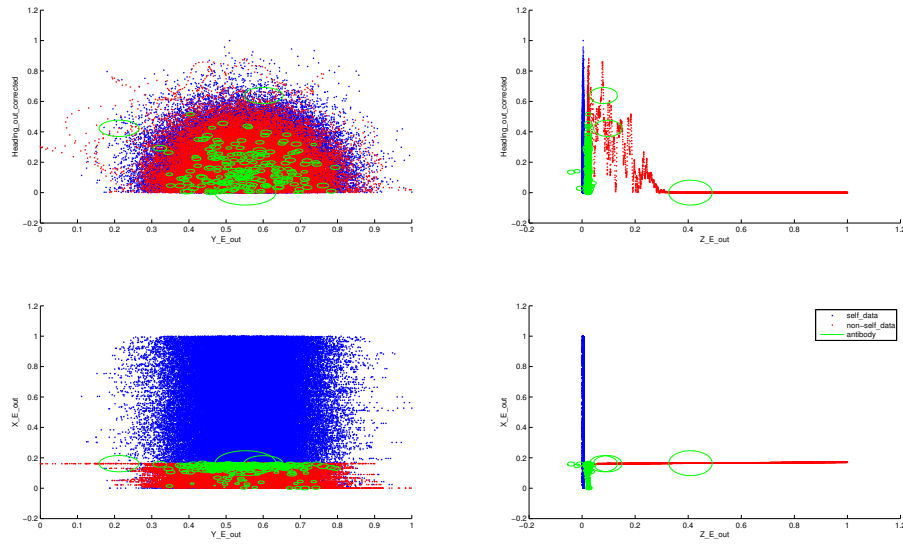


FIGURE 7.9: Pairwise Plot 7 of 7 Showing data from Mishap 4840 in red, all non-mishap flights in blue, and Mishap 4840 antibodies in Green

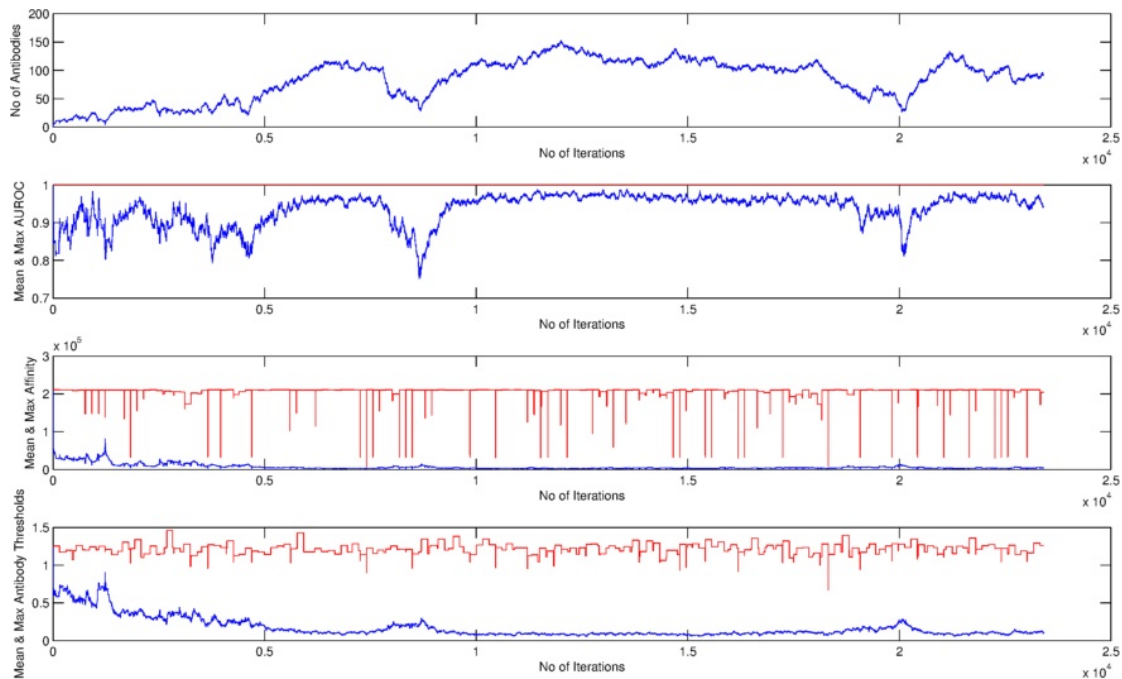


FIGURE 7.10: AIS Algorithm learning progress after 23,410 iterations. Blue is the average for the variable of all the current antibodies in the population and Red is the Maximum value of the variable.

TABLE 7.4: Results from using the AIS-FOQA Algorithm against various stored flight files.

Sim File	Relevant AB	True Positive	False Negative	True Negative	False Positive
Sims used during Training and Evaluation					
Sim4840	Mishap4840	99.9%	0.05%	36.9%	63%
Sim4894	Mishap4894	97.12%	2.8%	29%	70.9%
Sim4992	Mishap4992	98.8%	1.1%	15.9%	84%
Sim5080	Mishap5080	96.4%	3.5%	17.2%	82.7%
Sims used during Evaluation only					
Sim4889	NA	0%	0%	89.9%	10%
Sim4890	NA	0%	0%	89.9%	10%
Sim5107	Mishap4992	0.004%	99.9%	27.4%	72.5%
Sim4559	Mishap4840	50.9%	49%	99.7%	0.2%
Sim4884	NA	0%	0%	89.3%	10.6%
Summary					
Totals	NA	47.5%	52.4%	84.5%	15.4%

has a greatly improved True positive ratio on the data which it was trained on. The false positive ratios are however high but it should be kept in mind that if the algorithm alerts to a mishap but assigns the wrong case it is marked as false positive by the scoring algorithm despite it being a positive identification of a fault. The NN due to it's much lower power in this type of problem consequently has a lower false positive ratio but it is not because it is performing better at identifying fault from non-fault. Examining how the AIS-FOQA algorithm performs when faced with scenarios it has not been trained on we can see there is some false alerting of around 10%. On the scenarios which had mishaps the algorithm in one instance(Sim4559) manages a 50.9% True positive ratio which is very encouraging. The other mishap scenario Sim5107 the algorithm performs much worse and only manages a 0.004% True positive ratio. However if we look at the details we can see that the scoring system is assigning false positive to the algorithm correctly alerting as a mishap but matching the wrong mishap case. Unlike the NN the AIS-FOQA algorithm was able to learn from unsegmented flight data and was also able to alert against untrained data though in some instances the incorrect

case details were suggested as relevant to the mishap. Figure 7.12 shows the final output of the Algorithm when used during a flight. This shows which Mishap case the Algorithm thinks is most relevant at any particular time. The underlying immune response for each of these mishap cases can be seen in Figure 7.11. This shows some initial confusion of the algorithm to handle the non-fault portion at the beginning of the flight but we can then see after the mishap begins there is a strong response shown in Figure 7.11 and we can see the algorithm then settles on Mishap5840 as the relevant mishap. This is in contrast to the NN performance shown in Figure 7.1 which shows the NN was responding at a similar level throughout the flight. It can be seen that the AIS-FOQA algorithm is able to recognise within this flight the onset of the mishap. a benefit over the Neural Network as the output of each node is only True or False and so when multiple output nodes are True at the same time the most appropriate warning cannot be determined in a simple manner.

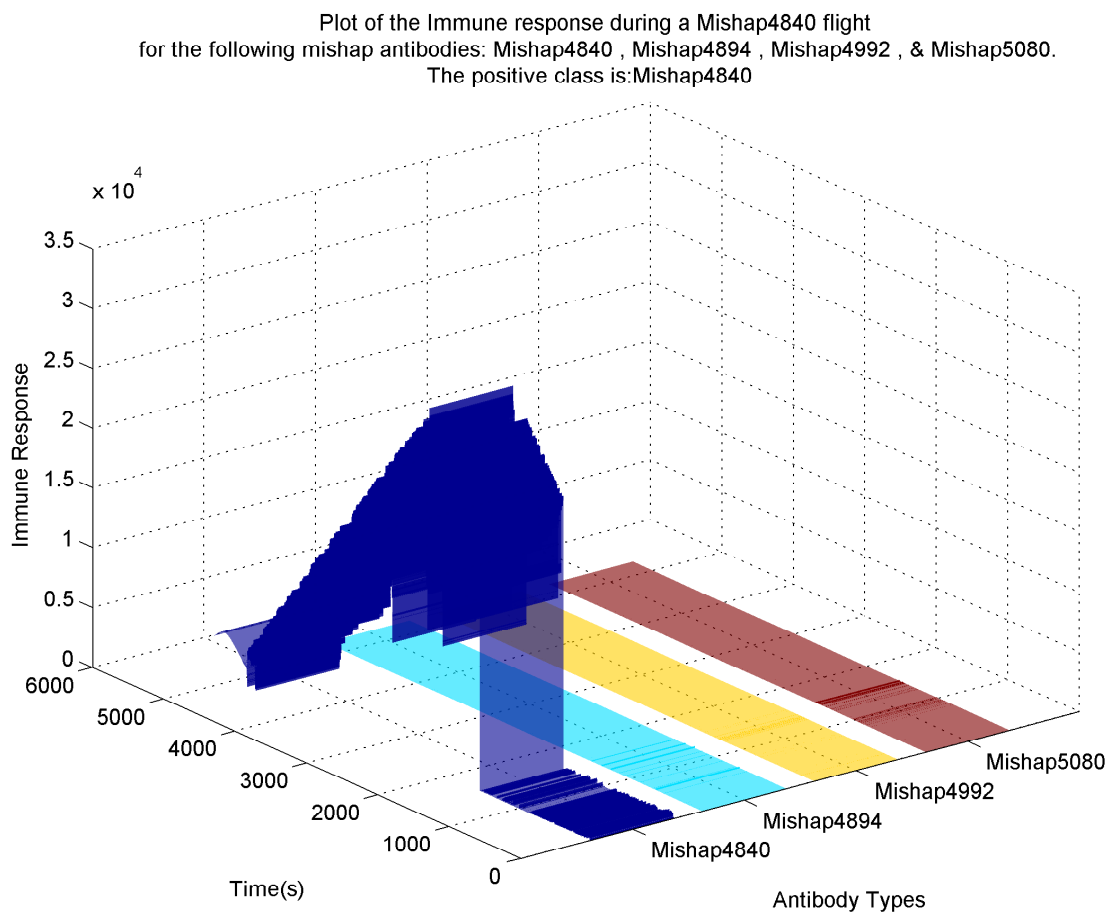


FIGURE 7.11: Sim4840 AIS Response showing full immune response from antibodies

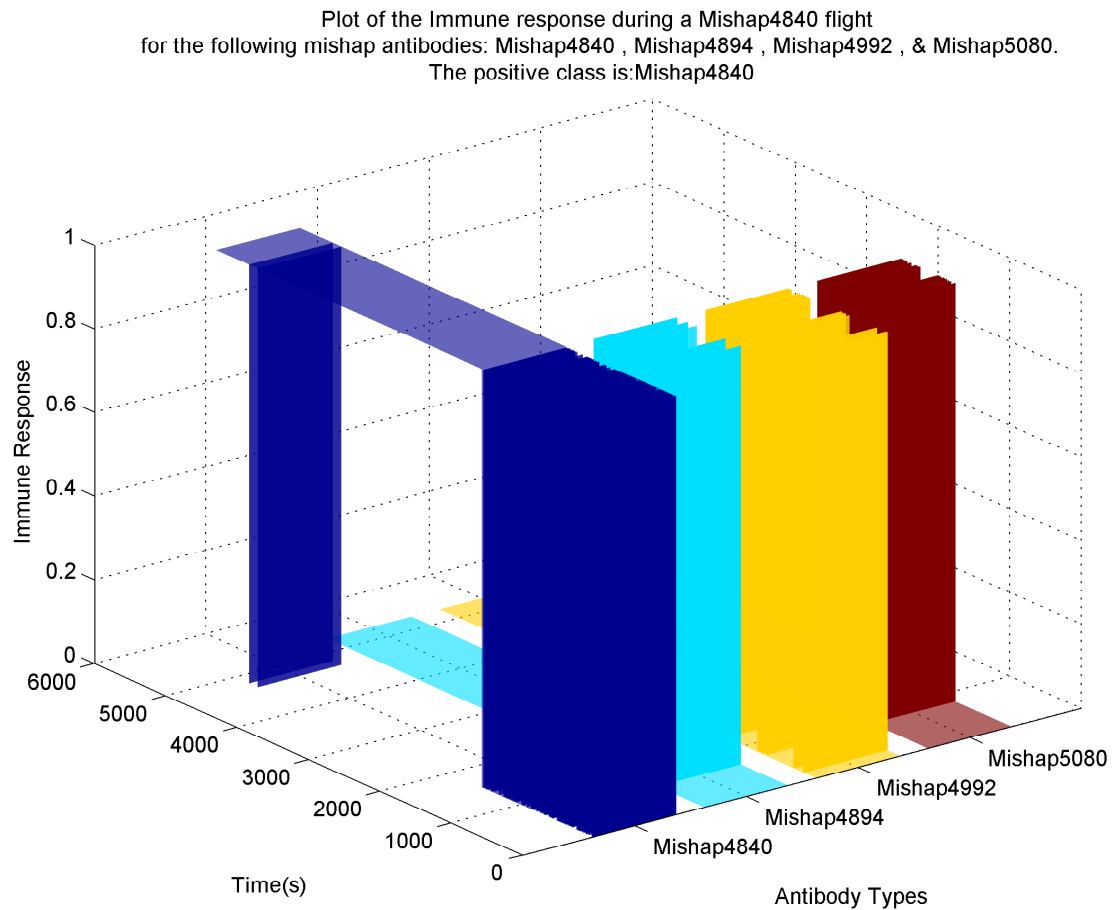


FIGURE 7.12: Sim4840 AIS Response showing the final output.

7.4 Usage Comparison

7.4.1 CBR in Application

Figure 3.2 shows how CBR can be used as part of the management of a fleet of assets. A CBR approach as shown in Figure 3.1 and Chapter 3.2 attempts to create a case base which can be used when training operators to recognise problems and understand appropriate responses. In CBR following any incidents or mishaps an expert would be required to analyse flight data and in a similar way to that examined in Chapter 5.8 they would need to ascertain the cause of the incident and determine the appropriate ways by which the incident could be recognised in future and the responsive actions that the operator should take in order to ensure safe operation of the aircraft. Growth of the case base increases the burden on the operators in terms of the number of possible scenarios to recognize and act upon when found.

7.4.2 NN-S-FBEC in Application

In use the NN-S-FBEC would alert and the operator would need to decide what action to take based on the case data associated to the alert. New incidents would be logged and the flight data processed to be included in the training data for the NN-S-FBEC algorithm. The NN-S-FBEC would need to be retrained with this new increased data set and an additional output node linked to the case data for the new incident. Each time a case is added to the system the diagnostic ability of the system for all cases will change. This would cause problems from a safety perspective and a performance assurance perspective. Strict version control could be exercised and different versions of the NN-S-FBEC could be run simultaneously to try cross validation and to evaluate changes but there would need to be additional work done to ensure sufficient performance.

7.4.3 AIS-FOQA in Application

In use the AIS-FOQA would alert and the operator would need to decide what action to take based on the case data associated to the alert. New incidents would be logged and the flight data processed to be included in the training data for the AIS-FOQA algorithm. In contrast to the NN-S-FBEC new information could be integrated and depending on the nature of the new non-self data being added either non of the existing antibodies or a subset of the antibodies most similar to the new case would be affected. This was checked using the FisherIris data and discussed in Section 6.5. This simplifies the addition of new cases to the database as the antibody thresholds would shrink where new antibodies are created close to them to maintain performance while permitting new antibodies to be added to the population.

7.5 Summary

The results from the assessment of the two classification methods have been presented and the AIS has been found to have some improvement in True Positive ratio over the neural network but due to the complex nature of the problem data neither of the two methods demonstrated a level of classification accuracy that would warrant deployment to an UAS system. The AIS-FOQA algorithm did not

performed at the level hoped but demonstrated the ability to learn from unsegmented flight data in contrast to the NN which tended to learn the whole flight rather than be able to reject the non-fault portions of the flight. In the discussion Chapter we will examine some of the reasons for this and what this result means in a wider context.

Chapter 8

Discussion, Conclusion, and Future Work

8.1 Discussion

Introduction

In this discussion the results will be explained, the results compared to other research, evaluated for importance and its limitations discussed.

- The research question is restated so its fulfilment can be discussed.
- Key findings of Challenges and opportunities for health management in UAS will be discussed
- The generation of a realistic dataset of flight data and mishap data for evaluation of machine learning algorithms.
- The use of UAS mishaps to guide future improvement is discussed and the application of these methods to health management
- The evaluation of the new AIS-FOQA approach
 - The results from the AIS-FOQA and the NN-S-FBEC processing the fisher iris data and their performance will be discussed. These results will be compared to other AIS performing the same task.
 - The results from the AIS-FOQA and the NN-S-FBEC processing the flight data and their performance will be discussed.

- Discussion and evaluation of the AIS-FOQA algorithm performance in general and its applicability to the health management of fleets of UAS are examined.

8.1.1 Research Question

As discussed in Chapter 1 the main research aim is to create a machine learning approach to learn from the operation of unmanned aircraft, in order to improve the ability of the operator to manage their systems during flight, with particular attention to mitigating the risk of incident and mishap. This dual use is an essential way to improve the health management of an aircraft without adding additional systems to the aircraft which would increase complexity, decrease mission performance, and create additional potential faults. The chosen approach was to examine how different machine learning approaches could make use of the flight data already captured by all aircraft with an FDR and in the case of UAS where no FDR exists the telemetry data can be used instead. It was decided after reviewing different types of machine learning methods (Chapter 3) to evaluate AIS for the health management of UAS due to the need for any workable solution to be able to learn from flight data with as little pre-processing as possible.

8.1.2 Challenges and Opportunities

Challenges

There are several challenges identified for the health management of UAS. There is a significant difficulty in adding new systems to aircraft due to the design problems associated with adding mass and power demands. This is especially true when the system to be added has not already been accepted within the industry and does not have historical metrics to guide the budgeting of mass and power within the conceptual design of the aircraft. The ability of the human operator to understand the aircraft state is also a difficult challenge and one within which there are difficult to implement solutions. Sophisticated health management systems have previously been implemented and failed due to the difficulty of correctly designing and managing this man machine interaction. Unmanned aircraft have been exposed to lower maintenance standards than manned aircraft even in situations where the aircraft is a significant capital investment. The analysis of flight data to

learn from operations is a developing field and FOQA has demonstrated improvement within civil airlines but is expensive to implement and when combined with incident analysis is a very expensive and time consuming way to add conclusions and warnings of future mishaps into the safety management system.

Opportunities

Unmanned aircraft however have a number of opportunities for health management. They typically benefit from more automation and digital sensors than many manned aircraft. They have very recent designs and in the small UAS market their designs have taken the same life cycles as mobile phones and so see very quick innovation and improvement. The majority of UAS from small to large systems have the capability to log flight data which can be downloaded between flights for retention. This feature is missing in the vast majority of manned aircraft and is only common in commercial aircraft and military aircraft. This flight data is a good opportunity to develop health management reasoners and as many UAS stream telemetry during flight to the remote pilot to assist them in their supervision of its mission this allows health management reasoners to be deployed off-board of the aircraft where they do not experience the same restrictions on mass and power by which any on-board system would be limited.

8.1.3 Realistic dataset

In order to construct a robust and respectable dataset it was essential to ensure it would be representative of real aircraft flight data. An appropriate flight model was chosen which has been used in previous research and has a pedigree attached. The missions were constructed based on known missions of the Aerosonde aircraft and fault scenarios were constructed in a critical way taking advantage of mishap reports from real UAS. The data was processed in a way that reduced the distorting effects of different units and scales between features.

8.1.4 Learning from Mishaps

The mishaps of real UAS were examined and used during this work in the construction of fault scenarios. The techniques used for incident analysis of real aircraft mishaps were used to review the fault data that was created and evaluate what

could be learned by the average UAS operator attempting to understand what went wrong during the flight.

8.1.5 The New Method

8.1.5.1 Comparison of AIS-FOQA and NN on Fisher Iris Data

The results for the application of the AIS-FOQA algorithm and the NN-S-FBEC algorithm on the Fisher Iris data are in Section 6.7.1.

- AIS-FOQA Confusion Table 6.3 is on page 154
- AIS-FOQA Classification Performance Table 6.4 is on page 154
- NN-S-FBEC Confusion Table 6.5 is on page 154
- NN-S-FBEC Classification Performance Table 6.6 is on page 155

It can be seen from the results in Table 6.6 and Table 6.4 that the AIS-FOQA algorithm is not quite as good as the NN-S-FBEC in this application but it is very close to it in performance. There is only $1\frac{1}{3}\%$ difference in the misclassification rate between the two algorithms. A previous AIS comparison effort demonstrated a very similar level of performance although the number of antibodies used was much greater. See Table 8.1 for the performance achieved in that research. Their algorithm was optimised for a much smaller volume of data and was developed over a longer period. The AIS-FOQA can be improved further but demonstrates that it has the ability to correctly classify multi class data at an appropriate performance level. It is also demonstrated that the AIS-FOQA algorithm is able to identify fault onset within the data and learn from that though it's ability to avoid alerting against self data(data from non mishap flights) still needs to be improved.

TABLE 8.1: Table 1 reproduced from Ji[2] describing the differing detection and false alarm rates achieved by variable sized detectors in their AIS algorithm.

		Detection Rate	False Alarm Rate
Setosa	MILA	95.16	0
100%	NSA	100	0
	V-detector	99.98	0
Setosa	MILA	94.02	8.42
50%	NSA	100	11.18
	V-detector	99.97	1.32
Versicolor	MILA	84.37	0
100%	NSA	95.67	0
	V-detector	85.95	0
Versicolor	MILA	84.46	19.6
50%	NSA	96	22.2
	V-detector	99.98	8.42
Virginica	MILA	75.75	0
100%	NSA	92.51	0
	V-detector	81.87	0
Virginica	MILA	88.96	24.98
50%	NSA	97.18	33.26
	V-detector	93.58	13.18

8.1.5.2 Comparison of AIS-FOQA and NN on Flight Data

The true positive rates of the two classifiers shown in Table 8.2 initially might be read to suggest that the the classifiers are better used as a negative indicator of faults. However this table should be read in combination with Tables 7.4 and 7.2 which show their effectiveness is greatly decreased when they are used with data they have not been trained on. Table 8.2 shows how the classification effectiveness of the AIS-FOQA Algorithm compares to the effectiveness of the NN-S-FBEC Algorithm. The low score achieved by the NN-S-FBEC and the data interspersion of the self and non-self data which can be seen in Figures 7.3 to 7.9 all show how closely interpenetration the data from the self and non-self flights is and the

TABLE 8.2: Comparison of results between the NN-FBEC and AIS-FOQA Algorithms.

Algorithm	True Positive	False Positive	False Negative	True Negative
AIS-FOQA	47.5164%	15.4025%	52.4836%	84.5975%
NN-FBEC	35.7393%	5.6314%	64.2607%	94.3686%

difficulty of this classification problem. When assessed on a known dataset such as Fisher Iris (see Section 6.7.1 and 6.7.2) the AIS-FOQA algorithm performs almost as well as the NN-S-FBEC algorithm despite the dataset not being necessarily as conducive to hypersphere based classifiers versus the hyperplane classifiers used in the NN-S-FBEC algorithm. The results obtained for the AIS while comparable to those obtained for NN-S-FBEC fall short of the required levels for a potential solution in the area of in-flight health management. Some of this is probably due to the high computational cost of the AIS-FOQA Algorithm and that the Algorithm had only evaluated 6.5% of the non-self data as potential antibodies. This 6.5% figure can be found by taking the number of iterations of the main Algorithm while loop given as 23,410 (see table 7.3) and working out what percentage it represents of the total amount of data tuples which is given in table 5.2. This is straightforward to calculate because on each iteration of the Algorithm a new data tuple is evaluated as a potential antibody so the number of iterations directly gives the number of data tuples which have been evaluated as potential antibodies. The number of antibodies evaluated is higher than this however as when antibodies die through apoptosis new antibodies are new ones are created through cloning and clonal maturation as appropriate. This means that although only 6.5% of the data has been evaluated as potential antibodies the information contained that 6.5% has been acquired and the algorithm has been improving the antibodies create to cover that space. The remaining 93.5% of data has been used in the calculation of affinity and so is used in the algorithm so the antibodies created do cover the set. It is just a case of potentially better antibodies not having been tried until the data has been worked through. The use of random seeding for antibodies was tried early on during the development but was found to be slower than using randomly selected tuples from the mishap data. This approach limits the potential variety of antibodies but does tie in closely to the immune system paradigm as it relies on stimulation of the antibody creating mechanism by a antigen to create

an antibody.

8.1.6 AIS-FOQA Application discussion

UAS

As discussed in Section 2.2 UAS are complex system of which the aircraft are but one part. However it is the health of the aircraft with which operators are chiefly concerned. Due to the remote and difficult control of UA by ground staff it can be difficult for operators to notice faults and correctly diagnose them. This is made more difficult by the cognitive burdens of remote control making it more difficult to learn the unique behaviours of the aircraft and to recognise when something unusual is going on. Learning from flight data and helping operators make sense and collate fleet level data of UAS is needed to enable large fleets and less direct supervision which is the goal for many operators to enhance their business offerings.

IVHM

As discussed in Section 2.4 IVHM is the health management of assets and consists of four main related objectives. Health Assessment, Mitigation of health issues, repair of the asset to restore its health, and verification to ensure that the health has truly been restored to the asset. Integrating known possible failure modes and how to deal with them with the new failure modes that are discovered through life is a difficult challenge and requires a method such as case based reasoning to bring the data together and allow it to be access and understood in a meaningful way.

Cognitive Gap

As discussed in Section 2.6 there is a difficulty operators observing the necessary precursors to faults in order to orient themselves, recognise the fault and act appropriately in response to it. The addition of sensors can give additional data to a machine but it increases the burden on the operator without necessarily ensuring enough attention and knowledge for them to interpret it correctly. The codification of failure knowledge into cases, scenarios, and teaching materials to help the operator understand the likely outcomes are all expensive to develop and without additional budget may not be available to UAS operators.

CAHM

As discussed in Section 2.7 there are key attributes shared between the immune systems and a Cognitive Asset Health management system. These are a content addressable memory, decentralised self organisation (immune network), inherent pattern matching mechanisms, and a retrieval mechanism. These work within the a case based reasoning system but by using an AIS new cases can be generated in a more efficient way than requiring operator time and analysis. As discussed in Section 4.6. As discussed in Section 7.4 in a usage comparison the AIS algorithm can learn new mishap cases without losing existing cases in memory.

Machine Learning for IVHM

As discussed in Section 3.4 the data that can be accumulated through the life of an aircraft is process history which can be used to understand normal and non-normal behaviour as well as fault and non-fault behavior. Clustering and Classification are the two main groups of methods for learning from data. The need in the IVHM for UAS area combines some of the needs from both these areas. The ability to classify data as fault or non-fault but also the ability to map the feature space so that failure modes can be related to each other and investigated in a meaningful way. AIS as discussed presents a novel solution to this maintenance problem and is inspired by a working system that performs a similar role. AIS is able to identify new threats(mishaps), remember old threats, dynamically adjusts itself in response to frequency of threat, and has a mechanism to reduce the risk of false alarms. The AIS-FOQA algorithm is able to work semi-supervised using data classified at a high level rather than requiring lengthy preprocessing of training data to identify the fault and it's onset. As discussed in Section 6.2.3 a priori data can be added into the population of antibodies to allow warnings to be generated for known issues such as stall speed or low exceedance of bank angles during a turn.

AIS Results

The results of the AIS-FOQA algorithm that has been produced are shown in Chapter 7 and in Section 8.1.5.2 above. The challenge faced was large and due to the current large computational requirements a full analysis has not yet been completed but the current results look promising due to their ability to exceed the performance of the competing NN-S-FBEC algorithm despite not having finished

a complete run through the data. The AIS-FOQA algorithm was also evaluated on the Fisher iris dataset and was found to have comparable performance to the NN-S-FBEC algorithm as well as to other AIS algorithms. The ability of the algorithm to leave one mishap and then retrain that information while adding new data and more mishaps to the case base was investigated and verified which meets another of the requirements from the CAHM discussion.

Fleet Applicability

The use of an AIS-FOQA algorithm though currently not computationally efficient could yield benefits in the management of large fleets of UAS. As fleet size grows it becomes more and more expensive to analyse data using the traditional FOQA and incident analysis methods. Use of personnel is one of the most difficult practices to scale effectively. If on the other hand AIS-FOQA could be used it would reduce the costs associated with fleet learning and free up resources to concentrate on the most serious mishaps or those incidents brought to the attention of the managers by the use of the AIS algorithm. Larger fleet size would also make it sensible to convert the algorithm to run in a distributed environment so that its use could be spread across multiple computers and the processing time substantially reduced.

8.2 Conclusions

8.2.1 Key Findings and Summary

UAS have been shown to need improvements in health management if they are to experience more widespread adoption and new use cases to be enabled. The operator supervision of remote aircraft is a key criteria for improving both the supervision and understanding of the vulnerabilities of the asset in general. The use of structured methods to acquire useful knowledge during the life of the asset has been demonstrated and CBR inspired methods discussed to perform this function. The AIS-FOQA algorithm has been shown to have promise in the analysis of the complex flight data and is at least on par with a NN-S-FBEC for classification results if not on processing time. It has been demonstrated that the AIS-FOQA algorithm can learn from flight data and classify faults. It performs at a comparable level to NN-S-FBEC when allowed to run to a stable solution. The AIS-FOQA algorithm incorporates features of a CBR as well as features to enable its use as

part of a cognitive asset health management system. The use of a new race type algorithmic method (Algorithm 9) allows larger datasets to be processed on a single pc than would otherwise be possible using the more normal population type algorithm.

AIS represent an opportunity to deploy the learning abilities of AI in a way that will benefit the industry in general and improve mishap rates for UAS. At present only the humans in the system can learn and attempt to incorporate their experience into the system, without methods designed to assist them in this function it is often only the operators who are most familiar with the aircraft who can identify the first onsets on failure. However these are seldom the correct people to create new systems to incorporate this learning into their own operations and into the design of new aircraft concepts. Ensuring the close cooperation or alliancing of operators, clients, designers, and manufacturers is necessary to creating a useful system which will be able to learn to identify anomalous conditions and associate them with underlying fault mechanisms or dismiss them as transient environmental phenomena which do not need to result in expensive maintenance actions. UAS are machines which cannot take actions which they have not been programmed to take. This limits them in their ability to respond to unexpected problems. They also cannot learn except with direct reprogramming by others. Using lessons from experience to avoid re-occurrences of dangerous incidents is central to the philosophy of aircraft safety but without AI of some description the UAS cannot learn itself and without an AI that can be audited and released for service in an appropriate way human users cannot build trust in the system to use it effectively. The use of an AIS to create a population of detectors that can be checked, understood and deployed enables structured use of the system whereas NN-SFBEC algorithms are difficult to check and to release. As systems increase further in complexity the challenge remains to cope with their failure mechanisms which cannot all be known during the design phase. Thus using concepts such as AIS to incorporate managed learning ability into aircraft safety systems may well be critical in meeting these challenges. Complex devices have become very difficult to use and caused issues with operators. As discussed one critical failure of highly automated systems is the mode awareness problem where it has become difficult for the operator to predict why a system has taken an unexpected action. Using these systems effectively depends on our ability to make them work in ways more like our own that operators can more readily comprehend. That is why these systems addressing the cognitive gap is so critical. Even with an AI operators cannot

delegate responsibility to a machine. The responsibility remains with the human as only the control is given away. That is why AIS represents a good opportunity to meet these challenges. Robust learning, self-regulating, and able to incorporate a priori case based reasoning into the initial seeding of the AIS-FOQA algorithm. Realistic Aerosonde flight data was created alongside realistic fault data into a mission context suitable for evaluation of machine learning to learn from mishap cases and recognise future mishaps. The use of fully untrained similar mishaps as a test of the algorithms is a tough standard but represents the requirements of a real system. The AIS-FOQA algorithm demonstrates some promising features and equivalent ability to some other methods at the cost of some computational time. If the computational complexity issue can be addressed then it can be effectively used. Further work to improve the Algorithm to reduce the false positive rate is needed as this is one of the prime features of AIS algorithms in general which is not currently being achieved to the desired standard in the AIS-FOQA implementation. This AIS-FOQA method incorporating automated analysis of flight data without costly manual expert pre-processing represents a type of method that could be applied in a large UAS fleet for learning at scale from the incidents and mishaps of a fleet of unmanned aircraft. The ability of the algorithm to rely on maintainer and operator led flight outcome classification rather than expert led fault identification in the classification of training data frees up resources to concentrate on applying the lessons learned and interpreting antibodies where features have been identified on a fleet level. These innovations in learning from operations present a novel method which has great potential for application in the expected large scale UAS uses of the future. As understanding of how the immune system itself works further algorithmic concepts may emerge that may enable further improvement in the automatic handling, classification, and response to threats to the safe operation of unmanned aircraft in as robust and as reliable a way as the biological immune system.

8.3 Future Work

Algorithm Improvement

As discussed more work should be done to evaluate the AIS-FOQA algorithm with smaller datasets which consist of a larger number of classes than currently

available.

Measures to Improve the AIS results

Reducing False Alarm Rate

The threshold used to negatively select could be implemented in a different way to ensure it's effectiveness and restrict the population to those candidates with an AUROC of 0.65 and above as intended from table 6.2. The partial failure of this measure was not found during validation on the Fisher Iris data but only during code review during thesis writing. The measure was applied to antibodies generated from non-self data but not to those generated as a result of mutation. See code in appendix A for details or further discussion. A different method to reduce the false alarm rate would be to increase the self radius used when determining whether a proposed antibody exists within self space.

Reducing False Negative Rate

The false negative rate would hopefully have been addressed by the Algorithm having run to completion or at least to have gone through all the non-self data. However it is difficult to determine what the end false negative rate would be due to the non linearly separable nature of the data. It is likely that some amount of false negative results would remain.

Increasing True Negative Rate

The increase of the self recognition region and the change in use from hyper spheres to hyper ellipsoids would both help fit the data better increase the true negative rate. If a different size recognition region could be used for each data point that would also certainly improve the True negative rate but it was not implemented due to the computational complexity of such a policy. Calculating the boundary of self space for each of the data points within it would increase the complexity of the Algorithm by at least n function calls.

Increasing True Positive Rate

If the hyper spheres currently used to represent the antibody recognition region were replaced with hyper ellipsoids it would help the antibodies to be fitted better around the surface of the self data without intruding into it. This has been

demonstrated as a technique but was not evaluated for this application due to a lack of time for evaluation. It should perhaps become part of future work instead.

Representation changes

Antibody Representation One improvement that could be made is to convert the representation of antibodies into a table based representation. The current representation is object orientated with a custom class object for the antibodies with its own generator and access methods. This creates additional complication when performing computation compared to a method where each antibody is represented as a row in a table. If a different language was used such as Python it would be more straight forward to represent antibodies as a table and directly apply vectorised operations to them vs the use of similar methods in Matlab. Matlab tables could also be used but the use of Python allows the use of 3rd party optimisation tools and GPU processing which in Matlab is an optional module that would otherwise have to be purchased.

Antibody Recognition Region The use of hyperspherical recognition regions while effective has the downside of very slow processing, especially when the problem has a high number of dimensions. There are numerous ways speed up this process or to change the recognition region to allow it to conform to the underlying feature space more closely.

Volume representation of self region The use of methods such as principal component analysis to replace the representation of self space with a series of hyper spheres or hyper ellipsoids instead of a cloud of data points. This enables a much faster comparison when the antibody is checking to see whether it is within the *self* region compared to the current implementation which checks each point in the self data to see if it is within the self radius of any of them.

Bibliography

- [1] Norman R.; Augustine. *Augustine's Laws*. Viking Penguin Inc, 1986.
- [2] Zhou Ji and Dipankar Dasgupta. Augmented negative selection algorithm with variable-coverage detectors. In *Evolutionary Computation, 2004. CEC2004. Congress on*, volume 1, pages 1081–1088. IEEE, 2004.
- [3] Andrew Heaton. *A Method to Support the Requirements Trade-Off of Integrated Vehicle Health Management for Unmanned Aerial Systems*. Phd, School of Applied Sciences, Cranfield University, July 2014. URL https://dspace.lib.cranfield.ac.uk/bitstream/1826/9297/1/Heaton_Andrew_Edward_Thesis_2014.pdf.
- [4] Tony Hooper. Revised over 20kg scheme inspectors - 2010/11, 2011.
- [5] JP Fielding, Craig P. Lawson, R. Pires, and Guido A. I. Monterzino. Development of the demon technology demonstrator uav. In *27TH INTERNATIONAL CONGRESS OF THE AERONAUTICAL SCIENCES*, pages –, 2010. URL http://www.icas.org/ICAS_ARCHIVE/ICAS2010/PAPERS/271.PDF.
- [6] Cranfield University. Flaviir project, 2015. URL http://www.flaviir.com/flaviir_brochure.pdf.
- [7] Deborah Saban. *Wake Vortex Modelling and Simulation for Air Vehicles in Close Formation Flight*. PhD thesis, Cranfield University, 2010.
- [8] Mario.G. Al Azzawi, Dia.;Perhinschi and Hever. Moncayo. Dendritic cell mechanism for aircraft immunity-based failure detection and identification. In *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, pages –, 2013.

- [9] Mario George Perhinschi, Dia Al Azzawi, Hever Moncayo, Adil Togayev, and Andres Perez. Immunity-based flight envelope prediction at post-failure conditions. *Aerospace Science and Technology*, 46:264 – 272, 2015. ISSN 1270-9638. doi: <http://dx.doi.org/10.1016/j.ast.2015.07.014>. URL <http://www.sciencedirect.com/science/article/pii/S1270963815002217>.
- [10] Florian Jentsch and Clint A Bowers. Evidence for the validity of pc-based simulations in studying aircrew coordination. *The International Journal of Aviation Psychology*, 8(3):243–260, 1998.
- [11] L. Pearson. Developing the flying bomb. *Naval Aviation in World War I*, pages 70–73, 1969.
- [12] International Civil Aviation Organisation. Icao cir 328, unmanned aircraft systems (uas). Technical report, International Civil Aviation Organisation, 999 University Street, Montral, Quebec, Canada H3C 5H7, 2011. URL http://www.icao.int/Meetings/UAS/Documents/Circular%20328_en.pdf.
- [13] B. T. Clough. Metrics, schmetrics! how the heck do you determine a uav’s autonomy anyway. Technical report, Air Force Research Laboratory,, 2002. URL <http://www.dtic.mil/dtic/tr/fulltext/u2/a515926.pdf>.
- [14] R. Parasuraman, T. B. Sheridan, and C. D. Wickens. A model for types and levels of human interaction with automation. *IEEE Trans Syst Man Cybern Pt A Syst Humans*, 30(3):286–297, 2000. ISSN 10834427. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=844354.
- [15] R.M. Taylor. Technologies for supporting human cognitive control. Technical report, Defence Science and Technology Laboratory, Farnborough, 2001.
- [16] Robert M. Taylor. Capability, cognition and autonomy. Technical report, Dstl Human Sciences Air Systems, 2003.
- [17] United States Air Force. Mq-1b predator usaf fact sheet, 2010.
- [18] United States Air Force. Rq-4 global hawk usaf fact sheet, 2008.
- [19] I. MacFadyen. Keynote address: Combat automation for airborne weapon systems: Man/machine interface trends and technologies. In *Combat Automation for Airborne Weapon Systems: Man/Machine Interface Trends and Technologies (SEE N 93-28850 11-54)*, pages K–, 7 RUE ANCELLE, 92200,

- Neuilly Sur Seine, France, 1993. NATO Advisory Group for Aerospace Research and Development,.
- [20] John P. Machuca, Michael E. Miller, and John M. Colombi. A cognitive task analysis-based evaluation of remotely piloted aircraft situation awareness transfer mechanisms. In *Cognitive Methods in Situation Awareness and Decision Support (CogSIMA), 2012 IEEE International Multi-Disciplinary Conference on*, pages 179–182. IEEE, 2012.
- [21] Charles Billings. Human-centered aviation automation: Principles and guidelines. Technical report, National Air & Space Administration, Ames Research Center, Moffett Field, CA 94035 United States, 1996. URL <http://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/19960016374.pdf>.
- [22] Shane A. Dougherty. An examination of latency and degradation issues in unmanned combat aerial vehicle environments. Master’s thesis, Air Force Institute of Technology, 2002. URL <http://www.dtic.mil/cgi-bin/GetTRDoc?AD=ADA401808>.
- [23] Office of the Secretary of Defense. Unmanned aerial vehicles roadmap 2002-2027, December 2002. URL https://www.nasa.gov/centers/dryden/pdf/111759main_DoD_UAV_Roadmap_2003.pdf.
- [24] United States Air Force. United states air force accident investigation board reports. Technical Report March/25, United States Air Force, 2012. URL <http://usaf.aib.law.af.mil/>.
- [25] Roy Hartfield, Mark Carpenter, Wesley Randall, and Joshua Hundley. Unmanned air vehicles (uav): Safety event prediction, and classification. In - *Aviation Technology, Integration, and Operations (ATIO) Conferences*, pages -. American Institute of Aeronautics and Astronautics, 2012. URL [-http://dx.doi.org/10.2514/6.2012-5427](http://dx.doi.org/10.2514/6.2012-5427).
- [26] Donald Rumsfeld. Dod news briefing - secretary rumsfeld and gen. myers, 2002. URL <http://archive.defense.gov/Transcripts/Transcript.aspx?TranscriptID=2636>.
- [27] W. Falkena, C. Borst, Q. P. Chu, and J. A. Mulder. Investigation of practical flight envelope protection systems for small aircraft. *J Guid Control Dyn*, 34(4):976–988, 2011. ISSN 07315090. URL <http://arc.aiaa.org/doi/pdf/10.2514/1.53000>.

- [28] Civil Aviation Authority. Cap 739: Flight data monitoring. Technical report, Civil Aviation Authority, Published by TSO (The Stationery Office) on behalf of the UK Civil Aviation Authority, 2003. URL <http://www.caa.co.uk/CAP739>.
- [29] S. Das, B. L. Matthews, and R. Lawrence. Fleet level anomaly detection of aviation safety data. In *IEEE Int. Conf. Prognostics Health Manage., PHM - Conf. Proc.*, pages –, 2011. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6024356.
- [30] M. D. Sudolsky. Ivhm solutions using commercially-available aircraft condition monitoring systems. In *IEEE Aerosp. Conf. Proc.*, pages –, 2007. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4161679.
- [31] M. D. Sudolsky. Arinc 573/717, 767 and 647a: The logical choice for maintenance recording and ivhm interface control or frame updates. In *Annual Conference of the Prognostics and Health Management Society*,, pages –, San Diego, CA, 2009. PHM Society. URL http://www.phmsociety.org/sites/phmsociety.org/files/phm_submission/2009/phmc_09_63.pdf.
- [32] T. J. Logan. Error prevention as developed in airlines. *Int.J.Radiat.Oncol.Biol.Phys.*, 71(1 SUPPL.):S178–S181, 2008. ISSN 03603016. URL <http://www.sciencedirect.com/science/article/pii/S0360301607043027>.
- [33] E. Chu, D. Gorinevsky, and S. P. Boyd. Detecting aircraft performance anomalies from cruise flight data. In *AIAA Infotech Aerosp.*, pages –, 2010. URL <http://arc.aiaa.org/doi/pdf/10.2514/6.2010-3307>.
- [34] D. Gorinevsky, A. Smotrich, R. Mah, A. Srivastava, K. Keller, and T. Felke. Open architecture for integrated vehicle health management. In *AIAA Infotech Aerosp.*, pages –, 2010. URL <http://arc.aiaa.org/doi/pdf/10.2514/6.2010-3434>.
- [35] G. Berz and M. Amherd. The power of recorded flight data analysis to support its sustainment studies. In *AIAA IEEE Dig Avionics Syst Conf Proc*, pages 4D31–4D311, 2007. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4391915.

- [36] S. Charbonneau. Stabilized landing criteria: A practical application of the corporate flight operations quality assurance program. In *Proc Corp Aviat Saf Semin*, pages 603–687, 2010. URL <http://flightsafety.org/aviation-safety-seminars/corporate-aviation-safety-seminar/cass-2010-proceedings-contents>.
- [37] L. Li, M. Gariel, R. J. Hansman, and R. Palacios. Anomaly detection in onboard-recorded flight data using cluster analysis. In *AIAA IEEE Dig Avionics Syst Conf Proc*, pages 4A41–4A411, 2011. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6096068.
- [38] Thomas Marino, Brian Fuller, and Joel Walker. Analysis techniques for automatic flight maneuver detection and evaluation. In *Infotech@Aerospace Conferences*, pages –. American Institute of Aeronautics and Astronautics, 2010. URL [-http://dx.doi.org/10.2514/6.2010-3516](http://dx.doi.org/10.2514/6.2010-3516).
- [39] R. C. Chang, C. E. Lan, and S. J. Su. An expert system for the foqa program. *Journal of Aeronautics, Astronautics and Aviation*, 39(1):29–35, 2007. URL <http://job.cust.edu.tw/job/rdc/a/73.pdf>.
- [40] Armen Nahapetian. A comparison of arinc 717 and arinc 767 flight data recording. In *Aviation Maintenance Europe*, pages –, London, UK, 2014. AVM. URL <http://www.avm-summit.com/europe/wp-content/uploads/2014/01/AVE-Nahapetian-Presentation-08.pdf>.
- [41] M. Finnegan. ”boeing 787s to create half a terabyte of data per flight”. *Computerworld*, 2013. URL <http://www.computerworlduk.com/news/data/boeing-787s-create-half-terabyte-of-data-per-flight-says-virgin-atlantic-3433595/>.
- [42] M. Poole. Closing the gap between accident investigation and training. In *Int Air Saf Semin Proc*, volume 1, pages 249–289, Affiliation: CAE Flight scape, United States; Correspondence Address: Poole, M.; CAE Flight scapeUnited States, 2010. URL <http://flightsafety.org/aviation-safety-seminars/international-air-safety-seminar/iass-2010-proceedings-contents>.
- [43] J. Flemming. Utilizing foqa (flight operations quality assurance) in training. In *Proc Corp Aviat Saf Semin*, pages 52–60, 2012. URL

- <http://flightsafety.org/aviation-safety-seminars/corporate-aviation-safety-seminar/cass-2012-proceedings-contents>.
- [44] Geoff Carrigan, Dave Long, ML Cummings, and John Duffner. Human factors analysis of predator b crash. *Proceedings of AUVSI: Unmanned Systems North America*, pages –, 2008.
- [45] Government Digital Service. Air accidents investigation branch - about us. Online, 2015. URL <https://www.gov.uk/government/organisations/air-accidents-investigation-branch/about>.
- [46] A. P. Tvaryanas, B. T. Thompson, and S. H. Constable. U.s. military unmanned aerial vehicle mishaps: Assessment of the role of human factors using hfacs. In *Unmanned Syst. North America Proc.*, pages 1423–1438, 2005. URL http://www.colorado.edu/ASEN/asen5519_arg/papers/TVARYANA.pdf.
- [47] A. P. Tvaryanas, W. T. Thompson, and S. H. Constable. Human factors in remotely piloted aircraft operations: Hfacs analysis of 221 mishaps over 10 years. *Aviat.Space Environ.Med.*, 77(7):724–732, 2006. ISSN 00956562. URL <http://www.ingentaconnect.com/content/asma/asm/2006/00000077/00000007/art00008>.
- [48] Anthony P. Tvaryanas and William T. Thompson. Recurrent error pathways in hfacs data: analysis of 95 mishaps with remotely piloted aircraft. *Aviat.Space Environ.Med.*, 79(5):525–532, 2008.
- [49] SAE International and Ian K. Jennions. *Integrated Vehicle Health Management: Perspectives on an Emerging Field*. SAE International, 2011.
- [50] O. Benedettini, T. S. Baines, H. W. Lightfoot, and R. M. Greenough. State-of-the-art in integrated vehicle health management. *Proc.Inst.Mech.Eng.Part G J.Aerosp.Eng.*, 223(2):157–170, 2009. ISSN 09544100. URL <http://pig.sagepub.com/content/223/2/157.short>.
- [51] D. Djurdjanovic, J. Liu, K. A. Marko, and J. Ni. Immune systems inspired approach to anomaly detection, fault localization and diagnosis in automotive engines, 2010. ISSN 1860949X. URL http://link.springer.com/chapter/10.1007/978-3-642-10690-3_7.

- [52] D. E. Paris, L. C. Trevino, and M. D. Watson. A framework for integration of ivhm technologies for intelligent integration for vehicle management. In *Aerospace Conference, 2005 IEEE*, pages 3843–3852, 2005.
- [53] G. B. Aaseng. Blueprint for an integrated vehicle health management system. In *Digital Avionics Systems, 2001. DASC. 20th Conference*, volume 1, pages 3C1/1–3C1/11 vol.1, 2001.
- [54] J. E. Land. Hums-the benefits-past, present and future. In *Aerospace Conference, 2001, IEEE Proceedings.*, volume 6, pages 3083–3094 vol.6, 2001.
- [55] Gilbert. Haddad. *AN OPTIONS APPROACH TO QUANTIFY THE VALUE OF DECISIONS AFTER PROGNOSTIC INDICATION*. PhD thesis, University of Maryland, University of Maryland, College Park, 2011.
- [56] Mike Sondalini. How to use condition based maintenance strategy for equipment failure prevention, 2011.
- [57] P. A. Scandura Jr. Integrated vehicle health management as a system engineering discipline. In *AIAA IEEE Dig Avionics Syst Conf Proc*, volume 2, pages 7.D.1–1–7.D.1–10, 2005. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1563450.
- [58] D. B. Jourdan, M. D. Piedmonte, V. Gavrillets, D. W. Vos, and J. McCormick. Enhancing uav survivability through damage tolerant control. In *AUVSI Unmanned Systems North America Conference 2010*, volume 3, pages 1763–1788, 2010. URL <http://arc.aiaa.org/doi/pdf/10.2514/6.2010-7548>.
- [59] II Beale, J. K. and A. Hess. Experiences with a-7e and the av-8b engine monitoring systems: the good and the ugly. In *Aerospace Conference Proceedings, 2000 IEEE*, volume 6, pages 221–227 vol.6, 2000.
- [60] M. Paulitsch, H. Ruess, and M. Sorea. Non-functional avionics requirements, 2008. ISSN 18650929; 3540884785 (ISBN); 9783540884781 (ISBN). URL http://link.springer.com/chapter/10.1007/978-3-540-88479-8_26.
- [61] D. P. Raymer. *ENHANCING AIRCRAFT CONCEPTUAL DESIGN USING MULTIDISCIPLINARY OPTIMIZATION*. PhD thesis, Kungliga Tekniska Hgskolan Royal Institute of Technology, SE-100

- 44 Stockholm Sweden, 2002. URL <http://www.aircraftdesign.com/RaymerThesisFinalRevLowRes.pdf>.
- [62] Civil Aviation Authority. Cap 722: Unmanned aircraft system operations in uk airspace - guidance, 2012. URL <http://www.caa.co.uk/application.aspx?catid=33&pagetype=65&appid=11&mode=detail&id=415>.
- [63] Enviroment Systems and Associates, Callen-Lenz Ltd. Project ursula, 2013.
- [64] Joshua Hundley. *Civil UAV Type Certification: DoD Mishap Analysis 2000-2009 and FAA Certification Roadmap*. PhD thesis, Auburn University, <http://hdl.handle.net/10415/3360>, 2012.
- [65] D. L. Hall, R. J. Hansen, and D. C. Lang. The negative information problem in mechanical diagnostics. *J.Eng.Gas Turbines Power*, 119(2):370–377, 1997. ISSN 07424795. URL <http://proceedings.asmedigitalcollection.asme.org/proceeding.aspx?articleid=2110662>.
- [66] Paul Fitts. *Human engineering for an effective air navigation and traffic control system*, volume 1st. National Research Council, Div. of, Oxford, England, 1951.
- [67] A. Hobbs and S. R. Herwitz. Human factors in the maintenance of unmanned aircraft. *Small*, 15(100):100s–, 2005.
- [68] David.D.; Sarter, Nadine.b.; Woods. Cognitive engineering in aerospace application : pilot interaction with cockpit automation. Technical report, NASA, Ames Research Center, Moffett Field, California, USA, 1993. URL <http://ntrs.nasa.gov/search.jsp?R=19940008774>.
- [69] Frans PB Osinga. *Science, strategy and war: The strategic theory of John Boyd*. Routledge, 2007.
- [70] J. G. Pelham, I. Fan, I. Jennions, and J. McFeat. Dual use ivhm for uas health management (sae technical paper 2013-01-2202), 2013.
- [71] R. F. Healing. Flight operations quality assurance (foqa-homp). In *Annu. Forum Proc. AHS Int.*, volume 3, pages 2167–2179, Grapevine, TX, 2009. URL <https://vtol.org/store/product/flight-operations-quality-assurance-foqahomp-2461.cfm>.

- [72] Kelly Neville, Beth Blickensderfer, Julian Archer, Katherine Kaste, and Stephen P. Luxion. A cognitive work analysis to identify human-machine interface design challenges unique to uninhabited aircraft systems. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 56(1):418–422, 2012. URL <http://pro.sagepub.com/content/56/1/418.abstract>.
- [73] Laurence.R.; Newcome. *Unmanned aviation: a brief history of unmanned aerial vehicles*. AIAA, 2004.
- [74] Richard Whittle. *Predator's big safari*, volume 14. Mitchell institute for airpower study, 2011. URL http://www.afa.org/Mitchell/Reports/MP7_Predator_0811.pdf.
- [75] J. R. Boyd. A discourse on winning and losing. Collated presentations, 1987.
- [76] Alan Hobbs and Ann Williamson. Skills, rules and knowledge in aircraft maintenance: errors in context. *Ergonomics*, 45(4):290–308, 2002.
- [77] J. Rasmussen. Skills, rules, and knowledge; signals, signs, and symbols, and other distinctions in human performance models. *Systems, Man and Cybernetics, IEEE Transactions on*, SMC-13(3):257–266, 1983. ISSN 0018-9472.
- [78] M. Endsley and C. Bolstad. Human capabilities and limitations in situation awareness. In *AGARD, Combat Automation for Airborne Weapon Systems: Man/Machine Interface Trends and Technologies*, pages –, 1993.
- [79] Sylvain LeTourneau, Fazel Famili, and Stan Matwin. Data mining for prediction of aircraft component replacement. *Special Issue on Data Mining*, pages –, 1999.
- [80] Gordon Baxter and Denis Besnard. Cognitive mismatches in the cockpit: Will they ever be a thing of the past? In *The Flightdeck of the Future: Human Factors in Datalinks and Freeflight*, 2004. URL <http://www.dirc.org.uk/publications/inproceedings/papers/77.pdf>.
- [81] E. L. Wiener and R. E. Curry. Flight-deck automation: Promises and problems. *Ergonomics*, 23(10):995–1011, 1980. ISSN 00140139. URL <http://www.tandfonline.com/doi/abs/10.1080/00140138008924809>.

- [82] R. Onken and A. Walsdorf. Assistant systems for aircraft guidance: Cognitive man-machine cooperation. *Aerosp Sci Technol*, 5(8):511–520, 2001. ISSN 12709638. URL <http://www.sciencedirect.com/science/article/pii/S1270963801011373>.
- [83] Thomas B. Sheridan. Human centered automation: oxymoron or common sense? In Anon, editor, *Proc IEEE Int Conf Syst Man Cybern*, volume 1, pages 823–828. IEEE, 1995. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=537867.
- [84] Agnar Aamodt and Enric Plaza. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI communications*, 7(1):39–59, 1994.
- [85] Ian Watson and Farhi Marir. Case-based reasoning: A review. *Knowl.Eng.Rev.*, 9(4):327–354, 1994.
- [86] Malcolm P. Atkinson, Jano I. van Hemert, Liangxiu Han, Ally Hume, and Chee Sun Liew. A distributed architecture for data mining and integration. In *Proceedings of the second international workshop on Data-aware distributed computing*, pages 11–20. ACM, 2009.
- [87] Jeffrey Dean and Sanjay Ghemawat. Mapreduce: Simplified data processing on large clusters. *Commun. ACM*, 51(1):107–113, January 2008. ISSN 0001-0782. doi: 10.1145/1327452.1327492. URL <http://doi.acm.org/10.1145/1327452.1327492>.
- [88] Lucas. Campos. Puttini. Ivhm development and the big data paradigm. In *Proceedings of SAE 2013 AeroTech Congress & Exhibition*, number –, pages —. SAE International, 2013. URL <http://papers.sae.org/2013-01-2332/>.
- [89] Stacey Higginbotham. How formula 1 turns 3d printing, big data and crappy bandwidth into sport, 2013.
- [90] ROBERTG EGGLESTON. Cognitive interface considerations for intelligent cockpits. *AGARD, Combat Automation for Airborne Weapon Systems: Man/Machine Interface Trends and Technologies 16 p(SEE N 93-28850 11-54)*, pages –, 1993. URL <https://www.cso.nato.int/Pubs/rdp.asp?RDP=AGARD-CP-520>.

- [91] John E Hunt, Denise E Cooke, and Horst Holstein. Case memory and retrieval based on the immune system. In *Case-Based Reasoning Research and Development*, pages 205–216. Springer, 1995.
- [92] Franz Aurenhammer. Voronoi diagrams;a survey of a fundamental geometric data structure. *ACM Comput. Surv.*, 23(3):345–405, September 1991. ISSN 0360-0300. doi: 10.1145/116873.116880. URL <http://doi.acm.org/10.1145/116873.116880>.
- [93] Bala Krishnamoorthy. Alpha shapes: Geometry and biomolecules. online, March 2015. URL <http://www.math.wsu.edu/math/Mathnotes/Proteins/node2.html>.
- [94] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review, American Psychological Association*, 65(6):386, 1958.
- [95] Bolan Liu, Changlu Zhao, Fujun Zhang, Tao Cui, and Jianyun Su. Misfire detection of a turbocharged diesel engine by using artificial neural networks. *Applied Thermal Engineering*, 55(1):26–32, 2013.
- [96] M.T. Musavi, W. Ahmed, K.H. Chan, K.B. Faris, and D.M. Hummels. On the training of radial basis function classifiers. *Neural Networks*, 5(4):595 – 603, 1992. ISSN 0893-6080. doi: [http://dx.doi.org/10.1016/S0893-6080\(05\)80038-3](http://dx.doi.org/10.1016/S0893-6080(05)80038-3). URL <http://www.sciencedirect.com/science/article/pii/S0893608005800383>.
- [97] Robert Hecht-Nielsen. Theory of the backpropagation neural network. In *Neural Networks, 1989. IJCNN., International Joint Conference on*, 1989.
- [98] David Heckerman. *A tutorial on learning with Bayesian networks*. Springer, 1998.
- [99] Martha A. Zaidan, Andrew R. Mills, and Robert F. Harrison. Bayesian framework for aerospace gas turbine engine prognostics. In *Aerospace Conference, 2013 IEEE*, pages 1–8. IEEE, 2013.
- [100] Susana Ferreiro, Aitor Arnaiz, Basilio Sierra, and Itziar Irigoien. Application of bayesian networks in prognostics for a new integrated vehicle health management concept. *Expert Syst.Appl.*, 39(7):6402–6418, 2012. ISSN

- 0957-4174. URL <http://www.sciencedirect.com/science/article/pii/S0957417411016988>.
- [101] S. Ferreiro and A. Arnaiz. Improving aircraft maintenance with innovative prognostics and health management techniques case of study: Brake wear degradation. In *ICAART - Int. Conf. Agents Artif. Intell., Proc.*, volume 1, pages 568–575, 2010. URL <http://www.springer.com/gb/book/9783642198892>.
- [102] Liang Zhang, Lindsay B Jack, and Asoke K. Nandi. Fault detection using genetic programming. *Mechanical Systems and Signal Processing*, 19(2): 271–289, 2005.
- [103] Matt Webster, Michael Fisher, Neil Cameron, and Mike Jump. Formal methods for the certification of autonomous unmanned aircraft systems. In *Computer Safety, Reliability, and Security*, pages 228–242. Springer, 2011.
- [104] Omer F. Eker. *A Hybrid Prognostic Methodology and its Application to Well-Controlled Engineering Systems*. PhD thesis, Cranfield University, January 2015. URL <https://dspace.lib.cranfield.ac.uk/handle/1826/9269>.
- [105] Bob Walther. Open systems architecture for condition-based maintenance (osa-cbm) primer, 2006.
- [106] L. Tang, X. Zhang, and J. DeCastro. Diagnosis of engine sensor, actuator and component faults using a bank of adaptive nonlinear estimators. In *IEEE Aerosp. Conf. Proc.*, pages –, 2011. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5747565&tag=1.
- [107] C. S. Byington, M. J. Watson, and S. P. Bharadwaj. Gas turbine accessory health management evaluation using a hybrid prognostic approach. In *Proc. ASME Turbo Expo*, volume 2, pages 55–66, Affiliation: Director of Systems Engineering, Impact Technologies, LLC, 2571 Park Center Blvd., Suite 1, State College, PA 16801, United States; Affiliation: Manager of Dynamic Systems, Impact Technologies, LLC, 2571 Park Center Blvd., Suite 1, State College, PA 16801, United States; Affiliation: Impact Technologies, LLC, 2571 Park Center Blvd., Suite 1, State College, PA 16801, United States; Correspondence Address: Byington, C. S.; Director of Systems Engineering, Impact Technologies, LLC, 2571 Park Center Blvd., Suite 1, State College, PA 16801, United States,

2008. URL <http://proceedings.asmedigitalcollection.asme.org/proceeding.aspx?articleid=1623482>.
- [108] A. Hess and L. Fila. The joint strike fighter (jsf) phm concept: Potential impact on aging aircraft problems. In *Aerospace Conference Proceedings, 2002. IEEE*, volume 6, pages 6–3021–6–3026 vol.6, 2002.
- [109] Irun R Cohen. *Tending Adam’s Garden: evolving the cognitive immune self*. Academic Press, 2000.
- [110] Stephanie Forrest, Alan S. Perelson, Lawrence Allen, and Rajesh Cherukuri. Self-nonself discrimination in a computer. In *Research in Security and Privacy, 1994. Proceedings., 1994 IEEE Computer Society Symposium on*, pages 202–212. Ieee, 1994.
- [111] J. Lee, M. Ghaffari, and S. Elmeligy. Self-maintenance and engineering immune systems: towards smarter machines and manufacturing systems. *Annual Reviews in Control*, 35(1):111–122, 2011.
- [112] P. K. Harmer, P. D. Williams, G. H. Gunsch, and G. B. Lamont. An artificial immune system architecture for computer security applications. *Evolutionary Computation, IEEE Transactions on*, 6(3):252–280, 2002. ISSN 1089-778X.
- [113] Feng Gu, Julie Greensmith, and U Aickelin. The dendritic cell algorithm for intrusion detection. *Bio-Inspired Communications and Networking, IGI Global*, pages 84–102, 2011.
- [114] Emma Hart and Jon Timmis. Application areas of ais: The past, the present and the future. *Applied soft computing*, 8(1):191–201, 2008.
- [115] Julie Greensmith, Amanda Whitbrook, and Uwe Aickelin. Artificial immune systems. In *Handbook of Metaheuristics*, pages 421–448. Springer US, 2010.
- [116] Dipankar Dasgupta and Stephanie Forrest. Artificial immune systems in industrial applications. In *Intelligent Processing and Manufacturing of Materials, 1999. IPMM’99. Proceedings of the Second International Conference on*, volume 1, pages 257–267. IEEE, 1999.
- [117] Fernando.J.; De Castro, L.Nunes.; Von Zuben. The clonal selection algorithm with engineering applications. In *Proceedings of GECCO*, pages 36–39, 2000.

- [118] Julie.; Greensmith and Uwe.; Aickelin. Dendritic cells for syn scan detection. In *Proceedings of the 9th annual conference on Genetic and evolutionary computation*, pages 49–56. ACM, 2007.
- [119] Uwe Aickelin, Peter Bentley, Steve Cayzer, Jungwon Kim, and Julie McLeod. Danger theory: The link between ais and ids? In *Artificial Immune Systems*, pages 147–155. Springer, 2003.
- [120] Julie Greensmith. *The dendritic cell algorithm*. PhD thesis, University of Nottingham, 2007.
- [121] FM Burnet. Clonal selection and after. *Theoretical Immunology*, 63:85, 1978.
- [122] Johnny Kelsey, Jon Timmis, and Andrew Hone. Chasing chaos. In *Evolutionary Computation, 2003. CEC’03. The 2003 Congress on*, volume 1, pages 413–419. IEEE, 2003.
- [123] Vincenzo Cutello, Giuseppe Nicosia, Mario Pavone, and Jonathan Timmis. An immune algorithm for protein structure prediction on lattice models. *Evolutionary Computation, IEEE Transactions on*, 11(1):101–117, 2007.
- [124] J Hunt, J Timmis, D Cooke, M Neal, and C King. Artificial immune systems and their applications, chapter jisys: development of an artificial immune system for real world applications, 1998.
- [125] Jon Timmis. *Artificial immune systems: a novel data analysis technique inspired by the immune network theory*. PhD thesis, Department of Computer Science, 2000.
- [126] L Nunes de Castro and Fernando J Von Zuben. ainet: an artificial immune network for data analysis. *Data mining: a heuristic approach*, 1:231–259, 2001.
- [127] Andrew Watkins, Jon Timmis, and Lois Boggess. Artificial immune recognition system (airs): An immune-inspired supervised learning algorithm. *Genetic Programming and Evolvable Machines*, 5(3):291–317, 2004.
- [128] Andrew Watkins and Jon Timmis. Exploiting parallelism inherent in airs, an artificial immune classifier. In *Artificial immune systems*, pages 427–438. Springer, 2004.

- [129] Yuji Watanabe, Akio Ishiguro, Yasuhiro Shirai, and Yoshiki Uchikawa. Emergent construction of a behavior arbitration mechanism based on the immune system. *Advanced Robotics*, 12(3):227–242, 1997.
- [130] Emma Hart. *Immunology as a metaphor for computational information processing: Fact or fiction?* PhD thesis, University of Edinburgh, 2002.
- [131] Amanda Whitbrook, Uwe Aickelin, and Jonathan Garibaldi. An idiotypic immune network as a short-term learning architecture for mobile robots. In *Artificial Immune Systems*, pages 266–278. Springer, 2008.
- [132] J Doyne Farmer, Norman H Packard, and Alan S Perelson. The immune system, adaptation, and machine learning. *Physica D: Nonlinear Phenomena*, 22(1):187–204, 1986.
- [133] Hugues Bersini and Francisco J Varela. Hints for adaptive problem solving gleaned from immune networks. In *Parallel problem solving from nature*, pages 343–354. Springer, 1990.
- [134] Catherine Beauchemin, Stephanie Forrest, and Frederick T Koster. Modeling influenza viral dynamics in tissue. In *Artificial Immune Systems*, pages 23–36. Springer, 2006.
- [135] Anastasio Salazar-Bañuelos. Immune responses: a stochastic model. In *Artificial Immune Systems*, pages 24–35. Springer, 2008.
- [136] Rob J De Boer, Alan S Perelson, and Ioannis G Kevrekidis. Immune network behavior. from stationary states to limit cycle oscillations. *Bulletin of mathematical biology*, 55(4):745–780, 1993.
- [137] Steven A Hofmeyr and Stephanie Forrest. Architecture for an artificial immune system. *Evolutionary computation*, 8(4):443–473, 2000.
- [138] Julie Greensmith and Uwe Aickelin. The deterministic dendritic cell algorithm. In *Artificial Immune Systems*, pages 291–302. Springer, 2008.
- [139] Susan Stepney, Robert E Smith, Jonathan Timmis, and Andy M Tyrrell. Towards a conceptual framework for artificial immune systems. In *Artificial Immune Systems*, pages 53–64. Springer, 2004.

- [140] Nick DL Owens, Jon Timmis, Andrew Greensted, and Andy Tyrrell. Modelling the tunability of early t cell signalling events. In *Artificial Immune Systems*, pages 12–23. Springer, 2008.
- [141] Paul S Andrews and Jon Timmis. A computational model of degeneracy in a lymph node. In *Artificial Immune Systems*, pages 164–177. Springer, 2006.
- [142] Jamie Twycross and Uwe Aickelin. Libtissue-implementing innate immunity. In *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on*, pages 499–506. IEEE, 2006.
- [143] Jamie Paul Twycross. *Integrated innate and adaptive artificial immune systems applied to process anomaly detection*. PhD thesis, University of Nottingham, 2007.
- [144] Steven A Hofmeyr and Stephanie Forrest. Immunity by design: An artificial immune system. In *Proceedings of the Genetic and Evolutionary Computation Conference*, volume 2, pages 1289–1296. Citeseer, 1999.
- [145] Jeffrey Kephart, Gregory Sorkin, Morton Swimmer, and Steve White. *Blueprint for a Computer Immune System**. Springer, 1999.
- [146] Mark Burgess et al. Computer immunology. In *LISA*, volume 98, pages 283–298, 1998.
- [147] Fabio A González and Dipankar Dasgupta. Anomaly detection using real-valued negative selection. *Genetic Programming and Evolvable Machines*, 4 (4):383–403, 2003.
- [148] Zhou Ji and Dipankar Dasgupta. Real-valued negative selection algorithm with variable-sized detectors. In *Genetic and Evolutionary Computation GECCO 2004*, pages 287–298. Springer, 2004.
- [149] Jungwon Kim and Peter J Bentley. An evaluation of negative selection in an artificial immune system for network intrusion detection. In *Proceedings of GECCO*, pages 1330–1337, 2001.
- [150] Justin Lewis Balthrop. *RIOT: A responsive system for mitigating computer network epidemics and attacks*. PhD thesis, Citeseer, 2005.
- [151] Jon Timmis, Roghrio De Lemos, Modupe Ayara, and Ross Duncan. Towards immune inspired fault tolerance in embedded systems. In *Neural Information*

- Processing, 2002. ICONIP'02. Proceedings of the 9th International Conference on*, volume 3, pages 1459–1463. IEEE, 2002.
- [152] Thomas Stibor, Jonathan Timmis, and Claudia Eckert. On permutation masks in hamming negative selection. In *Artificial Immune Systems*, pages 122–135. Springer, 2006.
- [153] Jason Brownlee. *Clever Algorithms: Nature Inspired Programming Recipes*. Jason Brownlee, 2012.
- [154] Dipankar Dasgupta and Fernando Nino. *Immunological computation: theory and applications*. CRC Press, 2008.
- [155] Geoffrey.; Coico, Richard.; Sunshine. *Immunology: a short course*. John Wiley & Sons, 2015.
- [156] C. S. Hogeman. Enhancing pilot training with safety data. In *Int Air Saf Semin Proc*, pages 224–237, 2011. URL <http://flightsafety.org/aviation-safety-seminars/international-air-safety-seminar/iass-2011-proceedings-contents>.
- [157] CAST/ICAO Common Taxonomy Team. Phase of flight: Definitions & usage notes, 2013. URL <http://www.intlaviationstandards.org/Documents/PhaseofFlightDefinitions.pdf>.
- [158] ANDREW E BRADLEY. The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30(7):1145–1159, 1997. URL http://www.cse.ust.hk/nevinZhangGroup/readings/yi/Bradley_PR97.pdf.
- [159] T Warren Liao, Zhiming Zhang, and Claude R Mount. Similarity measures for retrieval in case-based reasoning systems. *Applied Artificial Intelligence*, 12(4):267–288, 1998.
- [160] John E Hunt, Denise E Cooke, and Horst Holstein. Case memory and retrieval based on the immune system. In *Case-Based Reasoning Research and Development*, pages 205–216. Springer, 1995.
- [161] Marcos Zuccolotto, Carlos Eduardo Pereira, Luca Fasanotti, Sergio Cavalieri, and Jay Lee. Designing an artificial immune systems for intelligent maintenance systems. *IFAC-PapersOnLine*, 48(3):1451

- 1456, 2015. ISSN 2405-8963. doi: <http://dx.doi.org/10.1016/j.ifacol.2015.06.291>. URL <http://www.sciencedirect.com/science/article/pii/S2405896315005303>. 15th {IFAC} Symposium on Information Control Problems in Manufacturing INCOM 2015.
- [162] Thomas Stibor, Jonathan Timmis, and Claudia Eckert. On the use of hyperspheres in artificial immune systems as antibody recognition regions. In *Artificial Immune Systems*, pages 215–228. Springer, 2006.
- [163] Polly Matzinger. Tolerance, danger, and the extended family. *Annual review of immunology*, 12(1):991–1045, 1994.
- [164] K. Mitchell, B. Sholy, and A. J. Stolzer. General aviation aircraft flight operations quality assurance: overcoming the obstacles. *IEEE Aerosp. Electron. Syst. Mag.*, 22(6):9–15, 2007. ISSN 08858985. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4271320.
- [165] WF Phillips, CE Hailey, and GA Gebert. Review of attitude representations used for aircraft kinematics. *Journal of aircraft*, 38(4):718–737, 2001.
- [166] Brian L Stevens, Frank L Lewis, and Eric N Johnson. *Aircraft Control and Simulation: Dynamics, Controls Design, and Autonomous Systems*. John Wiley & Sons, 2015.
- [167] Jon Berndt. Jsbsim- an open source flight dynamics model in c++. In *AIAA Modeling and Simulation Technologies Conference and Exhibit*, 2004.
- [168] Richard Garcia and Laura Barnes. Multi-uav simulator utilizing x-plane. In *Selected papers from the 2nd International Symposium on UAVs, Reno, Nevada, USA June 8–10, 2009*, pages 393–406. Springer, 2009.
- [169] Malcolm Christopher Cotting. *Evolution of Flying Qualities Analysis: Problems for a New Generation of Aircraft*. PhD thesis, Virginia Polytechnic Institute and State University, 2010. URL <http://scholar.lib.vt.edu/theses/available/etd-04112010-222856/>.
- [170] Frederick Beamer, Mario.G Perhinschi, Matthew Cunningham, and Jennifer. Davis. Development of a simulation environment to support aircraft health management education. In *AIAA Atmospheric Flight Mechanics Conference*, pages –, 2011. URL <http://arc.aiaa.org/doi/abs/10.2514/6.2011-6614>.

- [171] Uas datalink local set, October 2014. URL <http://www.gwg.nga.mil/misb/docs/standards/ST0601.8.pdf>.
- [172] James F. Whidborne, Ian D. Cowling, and Oleg A. Yakimenko. A direct method for uav guidance and control. In *Proceedings of 23rd International Unmanned Air Vehicle Systems (UAVS) Conference, Bristol, UK, University of Bristol, Bristol*, pages –, 2008.
- [173] Unmanned Dynamics. *AEROSIM BLOCKSET User's Guide*, volume 1.2. U-Dynamics, Unmanned Dynamics, LLC No.8 Fourth St. Hood River, OR 97031. URL http://www.u-dynamics.com/aerosim/aerosim_ug.pdf.
- [174] Sefer Kurnaz, Omer Cetin, and Okyay Kaynak. Adaptive neuro-fuzzy inference system based autonomous flight control of unmanned air vehicles. *Expert Systems with Applications*, 37(2):1229–1234, 2010.
- [175] M Jamal Afridi, Ahsan Javed Awan, and Javaid Iqbal. Awg-detector: A machine learning tool for the accurate detection of anomalies due to wind gusts (awg) in the adaptive altitude control unit of an aerosonde unmanned aerial vehicle. In *Intelligent Systems Design and Applications (ISDA), 2010 10th International Conference on*, pages 1125–1130. IEEE, 2010.
- [176] Zhi-jun Yang, Xiao-Hui Qi, and Gan-Lin Shan. Simulation of flight control laws design using model predictive controllers. In *Mechatronics and Automation, 2009. ICMA 2009. International Conference on*, pages 4213–4218. IEEE, 2009.
- [177] Abhijit Kallapur, Mahendra Samal, Vishwas Puttige, Sreenatha Anavatti, and Matthew Garratt. A ukf-nn framework for system identification of small unmanned aerial vehicles. pages 1021–1026, 2008.
- [178] Sikandar Samar, Dimitry Gorinevsky, and Stephen P Boyd. Embedded estimation of fault parameters in an unmanned aerial vehicle. In *Computer Aided Control System Design, 2006 IEEE International Conference on Control Applications, 2006 IEEE International Symposium on Intelligent Control, 2006 IEEE*, pages 3265–3270. IEEE, 2006.
- [179] Tad McGeer. Laima: the first atlantic crossing by unmanned aircraft. online, 1998. URL http://aerovelco.com/production/wp-content/uploads/2015/03/Laima_the-first-Atlantic-crossing-by-unmanned-aircraft1.pdf.

- [180] Textron Inc. Aerosonde, 2015. URL <http://www.aerosonde.com/>.
- [181] United States Air Force. Air force safety and accident board investigations, 2013.
- [182] unknown. Air accident investigation board report executive summary on mq-1l s/n 99-003056 mishap. Technical report, USAF Air Accident Investigation Board, 2005. URL http://usaf.aib.law.af.mil/ExecSum2005/MQ-1L_27March05.pdf.
- [183] Gardner.J. Joyner. Air accident investigation board report on mq-1b t/n 06-3178 mishap. Technical report, USAF Air Accident Investigation Board, 2011. URL http://usaf.aib.law.af.mil/ExecSum2011/MQ-1B_Afghanistan_20%20Aug%2011.pdf.
- [184] Henry.T. Rogers. Air accident investigation board report on mq-1b t/n 06-3173 mishap. Technical report, USAF Air Accident Investigation Board, 2011. URL http://usaf.aib.law.af.mil/ExecSum2011/MQ-1B_Djibouti_7%20May%2011.pdf.
- [185] unknown. Air accident investigation board report executive summary on mq-1b t/n 05-3147 mishap. Technical report, USAF Air Accident Investigation Board, 2009. URL http://usaf.aib.law.af.mil/ExecSum2009/MQ-1B_AOR_14%20Sep%2009.pdf.
- [186] Matlab. Matlab: Sample data sets. Online, 2015. URL http://uk.mathworks.com/help/stats/_bq9uxn4.html.
- [187] Ronald A Fisher. The use of multiple measurements in taxonomic problems. *Annals of eugenics*, 7(2):179–188, 1936.
- [188] Cranfield University. Flight data monitoring (fdm) and flight operational quality assurance (foqa) in commercial aviation. Website, 2015. URL <https://www.cranfield.ac.uk/Courses/training/flight-data-monitoring-fdm-foqa>.
- [189] Thomas Stibor, Jonathan Timmis, and Claudia Eckert. A comparative study of real-valued negative selection to statistical anomaly detection techniques. In *Artificial Immune Systems*, pages 262–275. Springer, 2005.

- [190] Dipankar Dasgupta, K. KrishnaKumar, D. Wong, and M. Berry. Negative selection algorithm for aircraft fault detection. In *Artificial immune systems*, pages 1–13. Springer, 2004.
- [191] Roger Stafford. Random points in an n-dimensional hypersphere. Matlab File Exchange, December 2005. URL <http://www.mathworks.com/matlabcentral/fileexchange/9443-random-points-in-an-n-dimensional-hypersphere>.
- [192] W.J. Cody. An overview of software development for special functions. *Numerical Analysis: Series of Lecture Notes in Mathematics*, 506, 2006. Lecture Notes in Mathematics.
- [193] Reuven Y. Rubinstein and Dirk P. Kroese. *Simulation and the monte carlo method*. John Wiley & Sons, 2008.
- [194] Duhu Man, Kenji Uda, Hironobu Ueyama, Yasuaki Ito, and Koji Nakano. Implementations of a parallel algorithm for computing euclidean distance map in multicore processors and gpus. *International journal of networking and computing*, 1(2):260–276, 2011.
- [195] Shuichi Asano, Tsutomu Maruyama, and Yoshiki Yamaguchi. Performance comparison of fpga, gpu and cpu in image processing. In *Field Programmable Logic and Applications, 2009. FPL 2009. International Conference on*, pages 126–131. IEEE, 2009.
- [196] Michael Sipser. *Introduction to the theory of computation*. Thomson Course Technology, 2006.
- [197] Mark Hudson Beale, Martin T. Hagan, and Howard. Demuth. *Neural Network Toolbox User Guide*. Matlab, The MathWorks, Inc. 3 Apple Hill Drive Natick, MA 01760-2098, 9 edition, March 2016. URL http://uk.mathworks.com/help/pdf_doc/nnet/nnet Ug.pdf.
- [198] Roi Livni, Shalev-Shwartz Shai, and Shamir Ohad. On the computational efficiency of training neural networks. In *Advances in Neural Information Processing Systems*, 2014.
- [199] Executive summary aircraft accident investigation mq-1b t/n 06-3167, November 2008. URL http://usaf.aib.law.af.mil/ExecSum2009/MQ-1B_AOR_2Nov08.pdf.

- [200] United States Air Force. Mq-9 reaper usaf fact sheet, 2010.
- [201] William Redmond. Air force instruction 91-204: Safety investigations and reports, 2008. URL <http://www.e-publishing.af.mil/shared/media/epubs/afi91-204.pdf>.
- [202] Office of the Secretary of Defense. Uav reliability study, 2003. URL <http://www.uadrones.net/military/research/acrobat/0302.pdf>.
- [203] Peter W. Merlin. *Crash Course: Lessons Learned from Accidents Involving Remotely Piloted and Autonomous Aircraft*. Government Printing Office, NASA, 2013. URL http://www.nasa.gov/pdf/732725main_crash_course-ebook.pdf.
- [204] USAF Safety Centre. U-2 mishap history, Nov 2015. URL <http://www.afsec.af.mil/shared/media/document/AFD-151120-038.pdf>.
- [205] V. Weissberg, A. Green, and H. Mey-Paz. The birth of specialized airframe technology for uavs. In *Israel Annu. Conf. Aerosp. Sci.*, volume 1, pages 324–336, 2012. URL <http://www.proceedings.com/14740.html>.
- [206] Office of the Secretary of Defense. Unmanned aircraft systems roadmap, 2005. URL http://www.fas.org/irp/program/collect/uav_roadmap2005.pdf.
- [207] USAF Safety Centre. Mq-1 rpa mishap history, Nov 2014. URL <http://www.afsec.af.mil/shared/media/document/AFD-141119-011.pdf>.
- [208] USAF Safety Centre. Rq-4 rpa mishap history, Nov 2014. URL <http://www.afsec.af.mil/shared/media/document/AFD-141119-012.pdf>.
- [209] USAF Safety Centre. Mq-9 rpa mishap history, Nov 2014. URL <http://www.afsec.af.mil/shared/media/document/AFD-141119-013.pdf>.

Appendix A

AIS Race Type Algorithm

This algorithm is a multi-class algorithm. It can handle multiple classes of Non-Self data within the Non-Self table. It is version 5 of the multi-class race type AIS and represents the furthest that this research has gone to date.

A.1 Inputs

A.1.1 Mandatory Inputs

self_table : A table each row of which represents a tuple of normalised flight data with a class name attached (Non Mishap)

non_self_table : A table each row of which represents a tuple of normalised flight data with a class name attached (Various different mishaps are each given their own class)

duration : Seconds for the algorithm to run

channel_names_for_ais : Fields from the table which are to be used in the creation of antibodies along with the order they will be within the schema

partition_planv2 : The settings used in this variable set rejection thresholds for new antibodies, mutation operators, & clonal selection settings. They can be found in table 6.2

self_radius : The self radius to be used with all self data points

`self_class` : (Non Mishap) The same class is listed here as can be found in the self table.

`apoptosis_age` : Age at which antibodies will die

`save_interval` : How frequently the algorithm is to save its current state.

`min_max_input` : The minimum and maximum value for each data channel.

A.1.2 Optional Inputs

The optional inputs are used when the algorithm has been previously run and it needs to be started up where the last run left off.

`previous_antibody_pool` : Antibodies created so far

`tracker_input` : A matrix containing algorithm performance from the previous runs of the algorithm

`non_self_index_in` : A list tracking which data points have been tested for antibody potential

A.2 Outputs

A.2.1 Output before completion

`save_pool` : Antibodies created so far

`tracker` : A matrix containing algorithm performance over previous iterations

`non_self_index` : A list tracking which data points have been tested for antibody potential

A.2.2 Output at completion

`save_pool` : Antibodies created so far

`tracker` : A matrix containing algorithm performance over previous iterations

`non_self_index` : A list tracking which data points have been tested for antibody potential

A.3 Code

```

1 function [save_pool,tracker,non_self_index]=...
    series_race_antibody_mcv5(self_table,non_self_table,duration,...
    channel_names_for_ais,partition_planv2,self_radius,self_class,...
    apoptosis_age,save_interval,min_max_input,...
    previous_antibody_pool,tracker_input,non_self_index_in)
2 %series_race_antibody Summary of this function goes here
3 %   Detailed explanation goes here
4 %
5 % Outputs
6 %   save_pool --> Antibodies created so far
7 %   tracker --> A matrix containing algorithm performance over ...
    previous iterations
8 %   non_self_index --> A list tracking which data points have been...
    tested for antibody potential
9 %
10 % Inputs
11 %   self_table --> A table each row of which represents a tuple of...
    normalised flight data with a class name attached (Non Mishap)
12 %   non_self_table --> A table each row of which represents a ...
    tuple of normalised flight data with a class name attached (...
    Various different mishaps are each given their own class)
13 %   duration --> Seconds for the algorithm to run
14 %   channel_names_for_ais --> Fields from the table which are to ...
    be used in the creation of antibodies along with the order they...
    will be within the schema
15 %   partition_planv2 --> The settings used in this variable set ...
    rejection thresholds for new antibodies, mutation operators, & ...
    clonal selection settings.
16 %   self_radius --> The self radius to be used with all self data ...
    points

```

```

17 % self_class --> (Non Mishap) The same class is listed here as ...
    can be found in the self table.
18 % apoptosis_age --> Age at which antibodies will die
19 % save_interval --> How frequently the algorithm is to save its ...
    current state.
20 % min_max_input --> The minimum and maximum value for each data ...
    channel.
21 % Optional Inputs
22 % previous_antibody_pool --> Antibodies created so far
23 % tracker_input --> A matrix containing algorithm performance ...
    from the previous runs of the algorithm
24 % non_self_index_in --> A list tracking which data points have ...
    been tested for antibody potential
25 %
26 %
27
28
29 true_start_time=tic;
30 no_of_antibodies=0;
31 start_counter=0;
32 this_run_counter_offset=0;
33 combined_table = [self_table ; non_self_table];
34
35
36 no_of_channel_names_for_ais=size(channel_names_for_ais,2);
37 tracker=nan(1,8);
38 no_of_non_self_in=height(non_self_table);
39 non_self_index=[1:no_of_non_self_in];
40 if nargin <10
41     error('not enough arguments');
42 elseif nargin <13 && nargin >10
43     error('Too many arguments for initial start & too few for ...
        continuation of previous run');
44 elseif nargin ==13
45     antibody_pool=previous_antibody_pool;
46     no_of_antibodies=size(antibody_pool,2);
47     antibody_locations=nan(no_of_antibodies,...
        no_of_channel_names_for_ais);
48     antibody_thresholds=nan(no_of_antibodies,1);
49     for f=1:no_of_antibodies
50         for c=1:no_of_channel_names_for_ais
51             candidate_array(1,c)=antibody_pool(f).schema{c}.value;
52         end
53         antibody_locations(f,:)=candidate_array;

```

```

54         antibody_thresholds(f)=antibody_pool(f).alarm_threshold;
55
56     end
57     start_counter=tracker_input(end,1);
58     this_run_counter_offset=tracker_input(end,1);
59     tracker=tracker_input;
60     non_self_index=non_self_index_in;
61 elseif nargin >13
62     error('Too many arguments');
63 end
64 time_code_tmp=datestr(now);
65 time_code_tmp1=strrep(time_code_tmp,' ','_');
66 time_code_tmp2=strrep(time_code_tmp1,':','_');
67 time_code=strrep(time_code_tmp2,'-','_');
68
69 folder_name=strcat('AIS_progress_results_',time_code);
70 mkdir(folder_name);
71 cd(folder_name);
72
73 combined_data=table2array(combined_table(:,channel_names_for_ais))...
74     ;
75 self_data=table2array(self_table(:,channel_names_for_ais));
76
77 %determine no of unique classes
78 sample_classes=combined_table.sample_class;
79 no_of_entries=size(sample_classes,1);
80 unique_names=unique(non_self_table.sample_class);
81
82 no_of_classes=size(unique_names,1);
83 class_count=zeros(no_of_classes,1);
84 for a=1:no_of_entries
85     for b=1:no_of_classes
86         if strcmp(unique_names(b),sample_classes(a))
87             class_count(b)=class_count(b)+1;
88         end
89     end
90 end
91 self_sample_no=height(self_table);
92 self_message=strcat('Self class in set is:',self_class,' and this...
93     has:',num2str(self_sample_no),' no of samples');
94 disp(self_message);
95 disp('Mishap classes in set:');
96 no_of_classes=size(unique_names,1);
97 for a=1:no_of_classes

```



```

96     tmp_name=strcat(unique_names(a),' and this has :',num2str(...
    class_count(a),' samples');
97     disp(tmp_name);
98 end
99 antibody_no_tracker=no_of_antibodies;
100 %convert sample class names to num to speed checking later 0 is ...
    self class
101 non_self_classes=non_self_table.sample_class;
102 for a=1:no_of_entries
103     for b=1:no_of_classes
104         if strcmp(unique_names(b),sample_classes(a))
105             num_class(a,1)=b;
106         elseif strcmp(self_class,sample_classes(a))
107             num_class(a,1)=0;
108         end
109         if a <=no_of_non_self_in
110             if strcmp(unique_names(b),non_self_classes(a))
111                 non_self_num_class(a,1)=b;
112             elseif strcmp(self_class,non_self_classes(a))
113                 non_self_num_class(a,1)=0;
114             end
115         end
116     end
117
118 end
119 num_class_table=array2table(num_class);
120 combined_table=[combined_table num_class_table];
121 non_self_num_class_table=array2table(non_self_num_class);
122 non_self_table=[non_self_table non_self_num_class_table];
123
124 start_time=tic;
125 time_spent=0;
126 start_time_string=strcat('Main while loop start at','_',datestr(...
    now));
127 disp(start_time_string);
128 while time_spent<duration
129     time_spent=toc(start_time);
130     tracker_tmp(1,8)=no_of_antibodies;
131     start_counter=start_counter+1;
132     decade_check=start_counter/(save_interval*1000);
133     integerTest=~mod(decade_check,1);
134     if no_of_antibodies~=0
135         no_of_antibodies=size(antibody_pool,2);
136     end

```

```

137     if no_of_antibodies==0
138         %do nothing
139     else
140         ab_affinity_array=nan(no_of_antibodies,3);
141         for a_a=1:no_of_antibodies
142             ab_affinity_array(a_a,1)=antibody_pool(a_a)...
143             .sum_affinity;
144             ab_affinity_array(a_a,2)=antibody_pool(a_a)...
145             .alarm_threshold;
146             ab_affinity_array(a_a,3)=antibody_pool(a_a)...
147             .AU_ROC_affinity;
148             ab_affinity_array(a_a,4)=antibody_pool(a_a)...
149             .mean_distance_affinity;
150             ab_affinity_array(a_a,5)=a_a;
151         end
152     end
153     no_of_apoptosis_antibodies=0;
154
155     if no_of_antibodies==0
156         %do nothing
157     else
158         class_length_array=[];
159         for ab_no=1:no_of_antibodies
160
161             for class_no=1:no_of_classes
162
163                 if antibody_pool(ab_no).Mishap_reference_number==...
164                 class_no
165
166                     tmp_entry_array=nan(1,no_of_classes);
167                     tmp_entry_array(1,class_no)=ab_no;
168                     if ab_no==1
169                         class_length_array=tmp_entry_array;
170                     else
171                         class_length_array=vertcat(...
172                         class_length_array,tmp_entry_array);
173                     end
174                 end
175             end
176         end
177
178         for individual_class=1:no_of_classes
179             clear single_class_pool;
180             array_entry_index=1;

```

```

175         for array_entry=1:no_of_antibodies
176             tmp_antibody_reference=class_length_array(...
array_entry, individual_class);
177             if isnan(tmp_antibody_reference)
178                 else
179                     single_class_pool(array_entry_index)=...
antibody_pool(tmp_antibody_reference);
180                     array_entry_index=array_entry_index+1;
181                 end
182             end
183             no_of_class(individual_class)=array_entry_index-1;
184             if array_entry_index~=1
185                 no_of_single_class_antibodies =size(...
single_class_pool,2);
186                 tmp_ab_affinity_array=nan(...
no_of_single_class_antibodies,3);
187                 for a_a=1:no_of_single_class_antibodies
188                     tmp_ab_affinity_array(a_a,1)=single_class_pool...
(a_a).sum_affinity;
189                     tmp_ab_affinity_array(a_a,2)=single_class_pool...
(a_a).alarm_threshold;
190                     tmp_ab_affinity_array(a_a,3)=single_class_pool...
(a_a).AU_ROC_affinity^2;
191                     tmp_ab_affinity_array(a_a,4)=single_class_pool...
(a_a).mean_distance_affinity;
192                     tmp_ab_affinity_array(a_a,5)=a_a;
193                 end
194
195
196                 cm_roc_cutoff(individual_class)=prctile(...
tmp_ab_affinity_array(:,3),partition_planv2(1));
197                 max_roc_affinity(individual_class)=max(...
tmp_ab_affinity_array(:,3));
198                 ab_25_100_roc_affinity(individual_class)=prctile(...
tmp_ab_affinity_array(:,3),25);
199                 mean_roc_affinity(individual_class) =mean(...
tmp_ab_affinity_array(:,3));
200                 max_sum_affinity(individual_class)=max(...
tmp_ab_affinity_array(:,1));
201                 mean_sum_affinity(individual_class)=mean(...
tmp_ab_affinity_array(:,1));
202                 ab_95_100_range(individual_class)=max_roc_affinity...
(individual_class)-cm_roc_cutoff(individual_class);

```



```

231         removal_list=vertcat (removal_list,...
removal_entry);
232         antibodies_for_removal=horzcat (...
antibodies_for_removal,antibody_pool(a_b));
233     end
234
235
236     end
237
238     end
239     else
240         %no antibodies of this class
241     end
242
243 end
244 end
245
246 if no_of_apoptosis_antibodies>0
247     removal_list=sortrows (removal_list,1);
248     antibody_locations (removal_list (:,1),:)=[];
249     antibody_thresholds (removal_list (:,1))=[];
250     antibody_pool (removal_list (:,1))=[];
251     for removal_entry=1:no_of_apoptosis_antibodies
252         class_of_antibody=removal_list (removal_entry,2);
253
254         if antibodies_for_removal (removal_entry)...
.sum_affinity >=cm.sum_cutoff (class_of_antibody)
255
256             no_of_clones=partition_planv2 (3);
257
258             mutation=partition_planv2 (2)*...
antibodies_for_removal (removal_entry).alarm_threshold;
259             new_antibodies_from_cm=...
clonal_maturation (antibodies_for_removal (removal_entry),...
no_of_clones,(100*mutation));% the 100 is to compensate the ...
reduction which occurs insie normal clonal maturation
260
261             for b_a=1:no_of_clones
262                 new_antibodies_from_cm (b_a)...
.time=start_counter;
263                 candidate_array=zeros (1,...
no_of_channel_names_for_ais);
264                 for c=1:...
no_of_channel_names_for_ais

```

```

265             candidate_array(1,c)=...
new_antibodies_from_cm(b_a).schema{c}.value;
266             end
267             if b_a==1
268                 tmp_antibody_locations=...
candidate_array;
269             else
270                 tmp_antibody_locations=...
vertcat(tmp_antibody_locations, candidate_array);
271
272             end
273
274             end
275             if isempty(antibody_locations)
276             else
277                 tmp_antibody_locations=vertcat...
(tmp_antibody_locations,antibody_locations);
278             end
279
280
281             b_index=1;
282             for b_b=1:no_of_clones
283                 new_antibodies_from_cm(b_index...
).Mishap_classes_in_set=unique_names;
284                 candidate_array=zeros(1,...
no_of_channel_names_for_ais);
285                 for c=1:...
no_of_channel_names_for_ais
286                     candidate_array(1,c)=...
new_antibodies_from_cm(b_index).schema{c}.value;
287                 end
288                 [neg_select_result,...
nearest_self] = negative_selection(new_antibodies_from_cm(...
b_index),self_data,self_radius);%make assumption neg selection ...
& antibody coverage will not occur for cm produced antibodies
289                 [nearest_ab,ab_row_no]=pdist2(...
tmp_antibody_locations,candidate_array,'euclidean','Smallest'...
,1);
290                 if nearest_self>=nearest_ab
291                     new_antibodies_from_cm(...
b_index).alarm_threshold=nearest_ab;
292                 else
293                     new_antibodies_from_cm(...
b_index).alarm_threshold=nearest_self;

```

```

294                                     end
295
296                                     %now check affinity
297                                     [...]
distance.between.antibodies.and.samples,new_antibodies_from_cm(...
b_index)] = antibody_distance(combined_table,...
new_antibodies_from_cm(b_index));
298                                     new_antibodies_from_cm(b_index...
) = antibody_affinity(new_antibodies_from_cm(b_index),...
distance.between.antibodies.and.samples,unique_names);
299                                     [return_AUC,result_cm_affinity...
,new_antibodies_from_cm(b_index),mean_distance] = ...
return_Area_under_ROC_multiclassv3(new_antibodies_from_cm(...
b_index));
300                                     new_antibodies_from_cm(b_index...
).AU_ROC_affinity=return_AUC;
301                                     new_antibodies_from_cm(b_index...
).mean_distance_affinity=mean_distance;
302                                     new_antibodies_from_cm(b_index...
).sum_affinity=result_cm_affinity;
303                                     if result_cm_affinity<=0 || ...
return_AUC <0.5
304
305                                     new_antibodies_from_cm(...
b_index)=[];
306                                     tmp_antibody_locations(...
b_index,:)=[];
307                                     fail_flag=true;
308                                     else
309                                     b_index=b_index+1;
310                                     new_antibodies_from_cm(...
b_index).creation_method='Antibody from Clonal Maturation';
311                                     new_antibodies_from_cm(...
b_index).min_max_data=min_max_input;
312
313                                     end
314
315
316                                     end
317                                     no_of_surviving_clones=size(...
new_antibodies_from_cm,2);
318                                     if no_of_surviving_clones~=0
319                                     for e=1:no_of_surviving_clones
320                                     if e==1

```

```

321                                     cm_thresholds=...
new_antibodies_from_cm(e).alarm_threshold;
322                                     else
323                                     cm_thresholds=vertcat(...
cm_thresholds,new_antibodies_from_cm(e).alarm_threshold);
324                                     end
325                                     end
326
327                                     if isempty(antibody_pool);
328                                     antibody_pool=...
new_antibodies_from_cm;
329                                     else
330                                     antibody_pool=horzcat(...
antibody_pool,new_antibodies_from_cm);
331                                     end
332
333
334                                     if isempty(antibody_thresholds...
)
335                                     antibody_thresholds=...
cm_thresholds;
336                                     else
337                                     antibody_thresholds=...
vertcat(antibody_thresholds,cm_thresholds);
338                                     end
339
340                                     antibody_locations=...
tmp_antibody_locations;
341                                     end
342
343                                     elseif antibodies_for_removal(...
removal_entry).sum_affinity >= ab_25_100_sum_affinity(...
individual_class)
344                                     mutation=partition_planv2(4)*...
antibodies_for_removal(removal_entry).alarm_threshold*rand;
345                                     mutant_cell = FOQA_antibody_mutation(...
antibodies_for_removal(removal_entry),mutation) ;
346                                     mutant_cell.time=start_counter;
347                                     mutant_cell.Mishap_classes_in_set=...
unique_names;
348                                     for c=1:no_of_channel_names_for_ais
349                                     candidate_array(1,c)=...
mutant_cell.schema{c}.value;
350                                     end

```



```

351
352         [neg_select_result,nearest_self] = ...
negative_selection(mutant_cell,self.data,self.radius);%make ...
assumption neg selection & antibody coverage will not occur for ...
cm produced antibodies
353         [nearest_ab,ab_row_no]=pdist2(...
antibody_locations,candidate_array,'euclidean','Smallest',1);
354         if nearest_self>nearest_ab
355             mutant_cell.alarm_threshold=...
nearest_ab;
356         else
357             mutant_cell.alarm_threshold=...
nearest_self;
358         end
359         %now check affinity
360         [...
distance_between_antibodies_and_samples,mutant_cell] = ...
antibody_distance(combined_table,mutant_cell);
361         mutant_cell = antibody_affinity(...
mutant_cell,distance_between_antibodies_and_samples,...
unique_names);
362         [return_AUC,result_mutant_affinity,...
mutant_cell,mean_distance] = return_Area_under_ROC_multiclassv3...
(mutant_cell);
363         mutant_cell.AU_ROC_affinity=return_AUC...
;
364         mutant_cell.mean_distance_affinity=...
mean_distance;
365         mutant_cell.sum_affinity=...
result_mutant_affinity;
366         if result_mutant_affinity<=0 || ...
neg_select_result==true || return_AUC <0.5
367
368
369         else
370             mutant_cell.min_max_data=...
min_max_input;
371             mutant_cell.creation_method='...
Antibody from Mutation';
372             antibody_locations=vertcat(...
antibody_locations, candidate_array);
373             antibody_thresholds=vertcat(...
antibody_thresholds,mutant_cell.alarm_threshold);

```

```

374             antibody_pool=horzcat(...
antibody_pool,mutant_cell);
375             end
376
377             end
378         end
379         clear removal_list;
380     end
381
382
383
384     if no_of_antibodies==0
385         tracker_tmp(1,1)=start_counter;
386         tracker_tmp(1,2)=nan;%max_tracker
387         tracker_tmp(1,3)=nan;%mean_tracker
388         tracker_tmp(1,4)=nan;%mean_thres_tracker
389         tracker_tmp(1,5)=nan;%max_thres_tracker
390         tracker_tmp(1,6)=nan; %mean_auroc_tracker
391         tracker_tmp(1,7)=nan; %max_auroc_tracker
392         tracker=vertcat(tracker,tracker_tmp);%max_auroc_tracker
393     else
394         tracker_tmp(1,1)=start_counter;
395         tracker_tmp(1,2)=max(ab_affinity_array(:,1));%max_tracker
396         tracker_tmp(1,3)=mean(ab_affinity_array(:,1));%...
mean_tracker
397         tracker_tmp(1,4)=mean(ab_affinity_array(:,2));%...
mean_thres_tracker
398         tracker_tmp(1,5)=max(ab_affinity_array(:,2));%...
max_thres_tracker
399         tracker_tmp(1,6)=mean(ab_affinity_array(:,3)); %...
mean_auroc_tracker
400         tracker_tmp(1,7)=max(ab_affinity_array(:,3)); %...
max_auroc_tracker
401         tracker=vertcat(tracker,tracker_tmp);
402         no_of_antibodies=size(antibody_pool,2);
403     end
404
405     if integerTest==1 && no_of_antibodies~=0
406         save_name=strcat('antibody_pool-',num2str(decade_check));
407         display_string=strcat('There are :',num2str(...
no_of_antibodies),' antibodies at the :',num2str(start_counter)...
,' iteration of this AIS');
408         disp(display_string);
409         time_left=duration-time_spent;

```

```

410     time_left_string=strcat('There are :',num2str(time_left),'...
seconds left to go');
411     disp(time_left_string);
412     save_pool=antibody_pool;
413     for save_num=1:no_of_antibodies
414         save_pool(save_num).excess_threshold_gen=[];
415         save_pool(save_num).predicted_state_gen=[];
416         save_pool(save_num).affinity_gen=[];
417         save_pool(save_num).true_state_gen=[];
418     end
419     try
420         save(save_name,'save_pool','tracker','non_self_index','-...
v7.3');
421     catch AIS_err
422         disp([AIS_err.identifier]);
423         disp([AIS_err.message]);
424         for e=1:length(AIS_err.stack)
425             disp(['Error in ' AIS_err.stack(e).file ' at line ...
' num2str(AIS_err.stack(e).line)]);
426
427         end
428         disp(getReport(AIS_err,'extended'));
429         extended_report = getReport(AIS_err,'extended');
430         message_txt = strcat(extended_report, AIS_err.message,...
' Error in ', AIS_err.stack(e).file, ' at line ', num2str(...
AIS_err.stack(e).line));
431         sendmail('example@example.com','AIS Error',...
message_txt);
432     end
433     progress_notification(tracker,decade_check,...
channel_names_for_ais,antibody_pool,start_counter,...
max_roc_affinity,mean_roc_affinity,max_sum_affinity,...
mean_sum_affinity,unique_names,no_of_class,class_count,...
time_left);
434     end
435
436     %create new antibody
437     candidate = FOQA_antibody(no_of_channel_names_for_ais,0);
438     candidate.schema_names=channel_names_for_ais;
439     candidate.sample_selected='no_samples_selected_yet';
440     candidate.time=start_counter;
441     no_of_index_rows=size(non_self_index,2);
442     if no_of_index_rows<=0
443         non_self_index=[1:no_of_non_self_in];

```

```

444         no_of_index_rows=size(non_self_index,2);
445     end
446     non_self_sample_index_row=randi(no_of_index_rows);
447     non_self_sample_row=non_self_index(non_self_sample_index_row);
448     non_self_index(non_self_sample_index_row)=[];
449     for b_c=1:no_of_channel_names_for_ais
450         candidate.schema{b_c}.name=channel_names_for_ais{b_c};
451         current_channel_tmp=channel_names_for_ais{b_c};
452         candidate.schema{b_c}.value=non_self_table(...
current_channel_tmp)(non_self_sample_row);
453     end
454     candidate.Mishap_classes_in_set=unique_names;
455     candidate.Mishap_reference_number=...
non_self_table.non_self_num_class(non_self_sample_row);
456     candidate.self_class=self_class;
457     candidate.min_max_data=min_max_input;
458     candidate.creation_method='Antibody from Data';
459
460     antibody=candidate;%splash
461
462
463     candidate_array=zeros(1,no_of_channel_names_for_ais);
464     for a=1:no_of_channel_names_for_ais
465         candidate_array(1,a)=antibody.schema{a}.value;
466     end
467
468     %self coverage
469
470     if no_of_antibodies==0
471         antibody_locations(1,1:no_of_channel_names_for_ais)=...
candidate_array;
472         antibody_thresholds(1,1)=404;
473     else
474         [nearest_ab,ab_row]=pdist2(antibody_locations,...
candidate_array,'euclidean','Smallest',1);
475         distance_to_ab_boundary = nearest_ab-antibody_thresholds(...
ab_row,1);
476         if distance_to_ab_boundary <=0
477             %antibody is covered by another so jump back to start of ...
loop
478             continue
479         else
480             antibody_locations=vertcat(antibody_locations, ...
candidate_array);

```

```

481         antibody_thresholds=vertcat(antibody_thresholds,...
antibody.alarm_threshold);
482
483     end
484
485 end
486 %negative selection
487 [neg_select_result,nearest_self] = negative_selection(antibody...
, self_data, self_radius);
488 if neg_select_result==true
489     entry_no=size(antibody_thresholds,1);
490     antibody_thresholds(entry_no)=[];
491     antibody_locations(entry_no,:)=[];
492     continue
493 end
494 if no_of_antibodies==0
495     antibody.alarm_threshold=nearest_self;
496     antibody_thresholds(1,1)=antibody.alarm_threshold;
497 else
498     if nearest_ab<=nearest_self
499         antibody.alarm_threshold=nearest_ab;
500
501     else
502         antibody.alarm_threshold=nearest_self;
503     end
504     entry_no=size(antibody_thresholds,1);
505     antibody_thresholds(entry_no)=antibody.alarm_threshold;
506 end
507
508
509
510 %now check affinity
511 [distance_between_antibodies_and_samples,antibody] = ...
antibody_distance(combined_table,antibody);
512 [antibody] = antibody_affinity(antibody,...
distance_between_antibodies_and_samples);
513 [return_AUC,result_affinity,antibody,mean_distance] = ...
return_Area_under_ROC_multiclassv3(antibody);
514 if result_affinity<=0 || return_AUC <partition_planv2(1)
515     entry_no=size(antibody_thresholds,1);
516
517     antibody_thresholds(entry_no)=[];
518     antibody_locations(entry_no,:)=[];
519

```

```

520         continue
521     else
522         antibody.AUROC_affinity=return_AUC;
523         antibody.mean_distance_affinity=mean_distance;
524         antibody.sum_affinity=result_affinity;
525         if no_of_antibodies==0
526             antibody_pool(1,1)=antibody;
527         else
528             antibody_pool=horzcat(antibody_pool,antibody);
529         end
530
531     end
532     entry_no=size(antibody_thresholds,1);
533     if antibody_thresholds(entry_no)<=0
534         disp('threshold error');
535     end
536     no_of_antibodies=size(antibody_pool,2);
537
538
539
540 end
541
542
543 progress_notification(tracker,decade_check,channel_names_for_ais,...
544     antibody_pool,start_counter,max_roc_affinity,mean_roc_affinity,...
545     max_sum_affinity,mean_sum_affinity,unique_names,no_of_class,...
546     class_count,0);
547 end_message=strcat('Total no of iterations in set :',num2str(...
548     start_counter-this_run_counter_offset));
549 full_time_taken=toc(true_start_time);
550 alt_time_taken=toc(start_time);
551 time_message=strcat('Total time taken in seconds is :',num2str(...
552     full_time_taken));
553 alt_time_message=strcat('While time taken in seconds is :',num2str...
554     (alt_time_taken));
555 time_cycle_message=strcat('Average time taken per iteration is :',...
556     num2str(full_time_taken/(start_counter-this_run_counter_offset)...
557     ));
558 class_count_string = antibody_antigen_summary(antibody_pool);
559 disp(end_message);
560 disp(time_message);
561 disp(alt_time_message);
562 disp(num2str(alt_time_taken/full_time_taken));
563 disp(time_cycle_message);

```

```

556 disp(class_count_string)
557 end_save_name=strcat('end_of_run.mat');
558 save_pool=antibody_pool;
559 no_of_antibodies=size(antibody_pool,2);
560 for save_num=1:no_of_antibodies
561     save_pool(save_num).excess_threshold_gen=[];
562     save_pool(save_num).predicted_state_gen=[];
563     save_pool(save_num).affinity_gen=[];
564     save_pool(save_num).true_state_gen=[];
565 end
566 save(end_save_name,'save_pool','tracker','non_self_index','-v7.3')...
    ;
567 end
568
569 function progress_notification(tracker_in,decade_check_in,...
    channel_names_for_ais_in,antibody_pool_in,start_counter_in,...
    max_roc_affinity_in,mean_roc_affinity_in,max_sum_affinity_in,...
    mean_sum_affinity_in,unique_classes_in,no_of_class_in,...
    class_count_in,time_left_in)
570 no_of_signals=size(channel_names_for_ais_in,2);
571 no_of_antibodies_in=size(antibody_pool_in,2);
572 no_of_classes_in=size(unique_classes_in,1);
573 for a =1:no_of_classes_in
574     if a==1
575         class_message=strcat('Class :',unique_classes_in(a),' ...
consists of :',num2str(no_of_class_in(a)),' antibodies',' \nhas...
max ROC Affinity :',num2str(max_roc_affinity_in(a)),' \nhas ...
mean ROC Affinity :',num2str(mean_roc_affinity_in(a)),' \nhas ...
max Sum Affinity :',num2str(max_sum_affinity_in(a)) ...
576         , ' \nhas mean Sum Affinity :',num2str(...
mean_sum_affinity_in(a)),' \nThe total number of samples for ...
this class is :',num2str(class_count_in(a)));
577
578     else
579         class_message_update=strcat('\n\nClass :',...
unique_classes_in(a),' consists of :',num2str(no_of_class_in(a) ...
),' antibodies',' \nhas max ROC Affinity :',num2str(...
max_roc_affinity_in(a)),' \nhas mean ROC Affinity :',num2str(...
mean_roc_affinity_in(a)),' \nhas max Sum Affinity :',num2str(...
max_sum_affinity_in(a)) ...
580         , ' \nhas mean Sum Affinity :',num2str(...
mean_sum_affinity_in(a)),' \nThe total number of samples for ...
this class is :',num2str(class_count_in(a)));
581         class_message=strcat(class_message,class_message_update);

```

```

582     end
583 end
584 display_string=strcat('There are :',num2str(no_of_antibodies_in),'...
    antibodies at the :',num2str(start_counter_in),' iteration of ...
    this AIS');
585 time_left_string=strcat('There are :',num2str(time_left_in),' ...
    seconds left to go');
586 for a=1:no_of_signals
587     if a==1
588         name_string=char(channel_names_for_ais_in{1});
589
590     elseif a==(no_of_signals)
591
592         name_string=strcat(name_string,'_&_',char(...
            channel_names_for_ais_in{a}));
593     else
594         name_string=strcat(name_string,'_',char(...
            channel_names_for_ais_in{a}));
595     end
596 end
597 close all
598 figure
599 subplot(4,1,1)
600 plot(tracker_in(:,1),tracker_in(:,8))
601 xlabel('No of Iterations','Interpreter','none'); % x-axis label
602 ylabel('No of Antibodies','Interpreter','none'); % y-axis label
603
604 subplot(4,1,2)
605 plot(tracker_in(:,1),tracker_in(:,6))
606 hold on
607 plot(tracker_in(:,1),tracker_in(:,7),'r')
608 xlabel('No of Iterations','Interpreter','none'); % x-axis label
609 ylabel('Mean & Max AUROC','Interpreter','none'); % y-axis label
610
611 subplot(4,1,3)
612 plot(tracker_in(:,1),tracker_in(:,3))
613 hold on
614 plot(tracker_in(:,1),tracker_in(:,2),'r')
615 xlabel('No of Iterations','Interpreter','none'); % x-axis label
616 ylabel('Mean & Max Affinity','Interpreter','none'); % y-axis label
617
618 subplot(4,1,4)
619 plot(tracker_in(:,1),tracker_in(:,4))
620 hold on

```



```

621 plot(tracker_in(:,1),tracker_in(:,5),'r')
622 xlabel('No of Iterations','Interpreter','none'); % x-axis label
623 ylabel('Mean & Max Antibody Thresholds','Interpreter','none'); % y...
    -axis label
624
625
626 drawnow
627
628
629 filename=strcat('AIS_figure_generation',num2str(decade_check_in));
630 current_filepath=pwd;
631 attach_name{1}=strcat(current_filepath,'\ ',filename,'.jpg');
632
633 name_string=char(name_string);
634 message_txt_tmp=strcat('Figures for AIS generation :',num2str(...
    decade_check_in),' the variables in this run are :',name_string...
    ,'\n ',display_string,'\n ',class_message,'\n ', ...
    time_left_string);
635 message_txt_tmp=char(message_txt_tmp);
636 message_txt=sprintf(message_txt_tmp);
637 disp(message_txt);
638 JFrame = get(handle(gcf),'JavaFrame');
639 JFrame.setMaximized(true);
640 set(gcf, 'Color', 'white'); % white bckgr
641 export_fig( gcf, ...      % figure handle
642     (filename),... % name of output file without extension
643     '-painters', ...      % renderer
644     '-jpg', ...           % file format
645     '-r72' );             % resolution in dpi
646 if time_left_in <=0
647     sendmail('example@example.com','AIS session COMPLETED',...
        message_txt,attach_name);
648 else
649     sendmail('example@example.com','AIS progress figures',...
        message_txt,attach_name);
650 end
651
652 end
653
654 function [distance_between_antibodies_and_samples_out,antibody_in]...
    = antibody_distance(combined_table_in,antibody_in)
655 %antibody_sample_distance Summary of this function goes here
656 % Detailed explanation goes here
657 %Copyright 2014 Cranfield University & Jonathan Pelham

```

```

658
659
660 channel_names_for_ais=antibody_in.schema_names;
661 no_of_channel_names_for_ais_ad=size(channel_names_for_ais,2);
662
663 %create table with variables candidate no, distance, & ...
        sample_class
664
665
666 no_of_rows=height(combined_table_in);
667 candidate_segments=antibody_in.schema;%this will be a cell array ...
        of objects of class candidate_segment
668
669 paired_array=zeros(no_of_rows,no_of_channel_names_for_ais_ad);
670 candidate_array=zeros(1,no_of_channel_names_for_ais_ad);
671 for a=1:no_of_channel_names_for_ais_ad
672     tmp_name=candidate_segments{a}.name;
673     paired_array(1:no_of_rows,a)=combined_table_in.(tmp_name);
674     candidate_array(1,a)=candidate_segments{a}.value;
675 end
676
677 pairwise_distance_tmp=pdist2(candidate_array,paired_array,'...
        euclidean');
678 pairwise_distance=rot90(pairwise_distance_tmp,-1);
679 num_class=combined_table_in.num_class;
680 distance_between_antibodies_and_samples_out =table(...
        pairwise_distance,num_class);
681
682 end
683
684
685 function [updated_antibody] = antibody_affinity(antibody_in,...
        distance_between_antibodies_and_samples_in,unique_classes_in)
686 %antibody_affinity Summary of this function goes here
687 % Detailed explanation goes here
688 % Copyright 2014 Cranfield University & Jonathan Pelham
689
690
691 %compare_distance
692
693 candidate_results=distance_between_antibodies_and_samples_in;
694 alarm_threshold=antibody_in.alarm_threshold;
695 %excess threshold = alarm_threshold-distance
696 no_of_samples=height(candidate_results);

```

```

697 antibody_mishap_class=antibody_in.Mishap_reference_number;
698
699 if isempty(antibody_mishap_class)
700
701     %'Antibody has no mishap class assigned so we assign one based...
       on sum
702     no_of_classes=size(unique_classes_in,1);
703     class_sums=zeros(no_of_classes,1);
704
705     for b=1:no_of_classes
706         logical_tmp=candidate_results.num_class==b;
707         class_sums(b,1)=sum(logical_tmp);
708
709     end
710
711     class_num_index=1:no_of_classes;
712     class_sums(:,2)=class_num_index;
713
714     sorted_count=sortrows(class_sums,-1);
715     antibody_mishap_class=sorted_count(1,2);
716
717
718     antibody_in.Mishap_reference_number=antibody_mishap_class;
719     antibody_in.Mishap_reference_name=unique_classes_in(...
       antibody_mishap_class);
720 end
721 antibody_in.true_state_gen=candidate_results.num_class;
722 antibody_in.affinity_gen=1./(candidate_results.pairwise_distance);
723 alarm_samples=candidate_results.pairwise_distance<alarm_threshold;
724 antibody_in.excess_threshold_gen=alarm_threshold-...
       candidate_results.pairwise_distance;
725 for a=1:no_of_samples
726     if alarm_samples(a)==1
727         antibody_in.predicted_state_gen(a,1)=antibody_mishap_class...
       ;
728     else
729         antibody_in.predicted_state_gen(a,1)=0;
730     end
731 end
732
733
734 updated_antibody=antibody_in;
735 end
736

```

```

737 function [class_count_string] = antibody_antigen_summary(...
    antibodies)
738     No_of_mishap_classes=size(antibodies(1,1)...
    .Mishap_classes_in_set,1);
739     Mishap_Class_count=zeros(No_of_mishap_classes,1);
740     no_of_antibodies=size(antibodies,2);
741     for a=1:no_of_antibodies
742         Mishap_Class_count(antibodies(1,a).Mishap_reference_number...
    ,1)=Mishap_Class_count(antibodies(1,a).Mishap_reference_number...
    ,1)+1;
743     end
744     for a=1:No_of_mishap_classes
745         if a==1
746             class_count_string=strcat(char(antibodies(1,1)...
    .Mishap_classes_in_set{1}), ' Qty :', num2str(Mishap_Class_count(...
    a)));
747
748             elseif a==(No_of_mishap_classes)
749
750                 class_count_string=strcat(class_count_string, ', & ', ...
    char(antibodies(1,1).Mishap_classes_in_set{a}), ' Qty :', num2str...
    (Mishap_Class_count(a)));
751             else
752                 class_count_string=strcat(class_count_string, ', ', char...
    (antibodies(1,1).Mishap_classes_in_set{a}), ' Qty :', num2str(...
    Mishap_Class_count(a)));
753             end
754
755     end
756
757
758
759
760
761 end

```

Appendix B

AIS Population Type Algorithm

This variation of the AIS algorithm is only intended to create and mature antibodies against a single class of Non-self. Multiclass variants do exist but have not been shown here for the sake of clarity regarding how the algorithm works.

B.1 Inputs

B.1.1 Mandatory Inputs

self_table : A table each row of which represents a tuple of normalised flight data with a class name attached (Non Mishap)

self_antibodies : Any antibodies created in addition to self data points to enhance the negative selection process.

non_self_table : A table each row of which represents a tuple of normalised flight data with a class name attached (Mishap)

candidates : Antibodies created during previous runs of the algorithm.

generations : Generations is the number of generations which the maturation process

partition_plan : The settings used in this variable set rejection thresholds for new antibodies, mutation operators, & clonal selection settings.

self_radius : The self radius to be used with all self data points

`apoptosis_age` : Age at which antibodies will die

`unique_name` : Name to be used in algorithm run reports

B.2 Outputs

B.2.1 Output at completion

`puddle` : Antibodies population

`best_Affinity` : The max affinity of an antibody to the non-self data and the generation at which this antibody was found or last existed.

B.3 Code

```

1 function [puddle,best_Affinity] = antibody_maturationv7(self_table...
    ,self_antibodies,non_self_table,candidates,generations,...
    partition_plan,self_radius,apoptosis_age,unique_name)
2 %Maturation allows the antibodies to improve over generations
3 % Copyright 2014 Cranfield University & Jonathan Pelham
4 % Arguments
5 %     self_list is a table giving the locations of .mat files ...
    containing normalised FOQA data identified as self behaviour
6 %     non_self_list is a 1xn matrix giving the locations of .mat...
    files containing normalised FOQA data identified as non-self ...
    behaviour
7 %     candidates is the array of FOQA-antibody objects
8 %     generations is the number of generations which the ...
    maturation process
9 %
10 % Outputs
11 %     puddle --> Antibodies population
12 %     best_Affinity --> The max affinity of an antibody to the non-...
    self data and the generation at which this antibody was found ...
    or last existed.
13 %
14 % Inputs
15 %     self_table --> A table each row of which represents a tuple of...
    normalised flight data with a class name attached (Non Mishap)

```

```

16 %   self_antibodies --> Any antibodies created in addition to self...
    data points to enhance the negative selection process.
17 %   non_self_table --> A table each row of which represents a ...
    tuple of normalised flight data with a class name attached (...
    Mishap)
18 %   candidates --> Antibodies created during previous runs of the ...
    algorithm.
19 %   generations --> Generations is the number of generations which...
    the maturation process
20 %   partition_plan --> The settings used in this variable set ...
    rejection thresholds for new antibodies, mutation operators, & ...
    clonal selection settings.
21 %   self_radius --> The self radius to be used with all self data ...
    points
22 %   apoptosis_age --> Age at which antibodies will die
23 %   unique_name --> Name to be used in algorithm run reports
24
25
26 signals_in_antibody=candidates(1).schema_names;
27 no_of_signals=size(signals_in_antibody,2);
28 for a=1:no_of_signals
29     if a==1
30         name_string=char(signals_in_antibody{1});
31
32     elseif a==(no_of_signals)
33
34         name_string=strcat(name_string,','_&_',char(...
    signals_in_antibody{a}));
35     else
36         name_string=strcat(name_string,','_','_','char(...
    signals_in_antibody{a}));
37     end
38 end
39 non_self_table.sample_class(:,1)=1;%flag mishaps
40 ROC_X=nan;
41 ROC_Y=nan;
42 generation=nan;
43 best_Affinity=nan(generations,2);
44 average_Affinity=nan(generations,2);
45 std_Affinity=nan(generations,2);
46 min_Affinity=nan(generations,2);
47
48
49 samples_for_comparison=non_self_table;

```



```

50
51
52 for gen=1:generations
53     if gen>1
54         [candidates] = return_new_populationv4(self.table,...
55         self.radius,sorted_candidates,partition_plan,apoptosis_age);%...
56         mutation , clonal maturation, & negative selection
57     else
58     end
59
60     [distance_between_antibodies_and_samples,candidates] = ...
61     antibody_sample_distance_v3(self.antibodies,candidates,...
62     samples_for_comparison);
63
64     [candidates] = antibody_affinity(candidates,...
65     distance_between_antibodies_and_samples); %return Area under ...
66     ROC curve for each candidate
67
68     [sorted_candidates,current_max_Affinity,...
69     average_Affinity_current,std_affinity_current,...
70     min_affinity_current] = sort_antibodies_by_sum_Affinity(...
71     candidates);
72
73
74
75
76
77
78
79
80
81

```

```

69     best_Affinity(gen,1)=current_max_Affinity;
70     best_Affinity(gen,2)=gen;
71     average_Affinity(gen,1)=average_Affinity_current;
72     average_Affinity(gen,2)=gen;
73     std_Affinity(gen,1)=std_affinity_current;
74     std_Affinity(gen,2)=gen;
75     min_Affinity(gen,1)=min_affinity_current;
76     min_Affinity(gen,2)=gen;
77     no_of_candidates=size(candidates,2);
78     disp_string=strcat('Gen :',num2str(gen),' of :',num2str(...
79     generations),' current Affinity :',num2str(current_max_Affinity...
80     ),' the current no of antibodies is:',num2str(no_of_candidates)...
81     );
82     disp(disp_string);

```

```

82     decade_check=gen/10;
83     integerTest=~mod(decade_check,1);
84     try
85     if integerTest==1
86
87         close all;
88         figure
89         title(unique_name);
90         plot(best_Affinity(:,2),best_Affinity(:,1),'r');
91         hold on
92         plot(min_Affinity(:,2),min_Affinity(:,1),'r');
93         xlabel('Generation','Interpreter','none'); % x-axis label
94         ylabel('Antibody Affinity','Interpreter','none'); % y-axis...
95         label
96         plot(average_Affinity(:,2),average_Affinity(:,1));
97         title('Affinity distribution for the antibodies of each ...
98         generation');
99         plot(average_Affinity(:,2),(average_Affinity(:,1)+...
100         std_Affinity(:,1)),'k');
101         plot(average_Affinity(:,2),(average_Affinity(:,1)-...
102         std_Affinity(:,1)),'k');
103         drawnow
104         filename=strcat('AIS_figure_generation',num2str(gen),'...
105         of_batch-',unique_name);
106         current_filepath=pwd;
107         attach_name{1}=strcat(current_filepath,'\',filename,'.jpg'...
108         );
109
110         name_string=char(name_string);
111         message_txt=strcat('Figures for AIS generation :',num2str(...
112         gen),' the variables in this run are :',name_string);
113         message_txt=char(message_txt);
114         jFrame = get(handle(gcf),'JavaFrame');
115         jFrame.setMaximized(true);
116         set(gcf, 'Color', 'white'); % white bckgr
117         export_fig( gcf, ... % figure handle
118             (filename),... % name of output file without extension
119             '-painters', ... % renderer
120             '-jpg', ... % file format
121             '-r72' ); % resolution in dpi
122
123         sendolmail('example@example.com','AIS progress figures',...
124         message_txt,attach_name);
125     end

```

```
118     catch sim_err
119         disp([sim_err.identifier]);
120         disp([sim_err.message]);
121         for e=1:length(sim_err.stack)
122             disp(['Error in ' sim_err.stack(e).file ' at line ' ...
num2str(sim_err.stack(e).line)]);
123
124         end
125         disp(getReport(sim_err,'extended'));
126         extended_report = getReport(sim_err,'extended');
127         message_txt = strcat(extended_report, sim_err.message, ' ...
Error in ', sim_err.stack(e).file, ' at line ', num2str(...
sim_err.stack(e).line), ' sim no. ', simcountertxt);
128         sendolmail('example@example.com','Sim Error',message_txt);
129     end
130
131 end
132 best_Affinity_overall=max(best_Affinity);
133 message_txt=strcat('Gen :',num2str(gen),' of :',num2str(...
generations),' final Affinity :',num2str(current_max_Affinity),...
' and best Affinity :',num2str(best_Affinity_overall));
134 sendolmail('example@example.com','AIS Finished',message_txt)
135 puddle=candidates;
136 end
```

Appendix C

Persistent UAS Mishap Analysis

This report is intended to look into the use of persistent unmanned aircraft. To examine such mishap data and analysis as is available to try to better understand the challenges that face any persistent unmanned aircraft. First we must identify what a persistent unmanned aircraft is. Through looking at existing aircraft types and discussing common features we can come to a definition. This must be narrow enough that challenges are common among all the selected aircraft but broad enough that we can see some difference between aircraft and so see how the solution space is spread out. We must also be careful that the definition is not so broad that sub types appear within the definition.

C.1 Persistent UAS

Persistent UAS are UAS that operate for long periods of time. Their ability to persist in their mission is what distinguishes them from other UAS which are intended for short duration operations only. There is no definition for how long an UAS must be able to continue it's mission before it can correctly be named a persistent UAS or asset but all the aircraft discussed here have a flight endurance of at least 6 hours. In order to understand these aircraft and the types of mishaps to which they are susceptible we need some mishap reports and an available source can be found with the USAF.

- MQ-1/RQ-1 Predator → See Section C.1.3

- RQ-4 Global Hawk → See Section C.1.4
- MQ-9 Predator B/Reaper → See Section C.1.5

C.1.1 Common Features

The aircraft in this study all share a number of common features Unmanned, Payload use dominated mission requirements, High Aspect Ratio Wings, Single Engine, Long Endurance Implied high fuel fraction and thus large difference between Take-off & landing weights, Ku band satellite communications for BLOS (Beyond line of sight) control, VHF link for LOS (Line of sight) control for take-off and landing often via a second GCS (Ground Control Station) specifically configured and manned for this purpose, Transit and cruise flight phases controlled from a Ground Control station the location of which is limited only to the need for satellite uplink for C3.

C.1.2 Feature related issues

The design of an aircraft is sometimes referred to as an exercise in matching compromises to mission requirements. This has the effect of creating vulnerabilities which are common across families of aircraft. One famous example of this is the inertial coupling exhibited by some early supersonic fighter aircraft. This was a result of design decisions driven by the aircraft mission requirements. A large heavy radar set in the nose to detect adversaries and a large heavy engine situated towards the aft to propel the aircraft. The mass distribution of these aircraft has made them more susceptible to loss of control when rolled too suddenly. The small wings of such aircraft were inadequate to counter the centrifugal forces created by the high roll rate and the sudden pitch down effect. Another example is the high take-off and landing speeds of supersonic aircraft. In order to create an aircraft that performs acceptably during the cruise and supersonic cruise portions of flight a high wing loading is often chosen. This has the effect of requiring high take-off and landing speeds and high AoA (Angle of Attack) during landing. It often also causes large turn radius.

It is this synthesis of design choice and operational experience that is useful when examining persistent UAS and attempting to understand the types of issues this

family of aircraft may be prone to. These flight characteristics as we need to keep them in mind so that in the event of failure we fully understand how an impending failure will affect the margin to the edge of the cleared aerodynamic envelope. Several MQ-1 Reapers were lost due to poor envelope expansion including one known example of loss due to asymmetric loading conditions.[199]

C.1.3 MQ-1/RQ-1 Predator

General Characteristics [200]

Primary Function: Armed reconnaissance, airborne surveillance, and target acquisition

Contractor: General Atomics Aeronautical Systems, Inc.

Power Plant: Rotax 914F four-cylinder engine

Thrust: 115 horsepower

Wingspan: 55 feet (16.8 meters)

Length: 27 feet (8.22 meters)

Height: 6.9 feet (2.1 meters)

Weight: 1,130 pounds (512 kilograms) empty

Maximum takeoff weight: 2,250 pounds (1,020 kilograms)

Fuel Capacity: 665 pounds (100 gallons)

Payload: 450 pounds (204 kilograms)

Speed: Cruise speed around 84 mph (70 knots), up to 135 mph

Range: Up to 770 miles (675 nautical miles)

Ceiling: Up to 25,000 feet (7,620 meters)

Armament: Two laser-guided AGM-114 Hellfire missiles

Crew (remote): Two (pilot and sensor operator)

Unit Cost: \$20 million (includes four aircraft with sensors, ground control station and Predator Primary satellite link) (fiscal 2009 dollars)

Initial operating capability: March 2005

Inventory: Total force, 164

C.1.4 RQ-4 Global Hawk

General Characteristics [18]

Primary Function: High-altitude, long-endurance ISR

Contractor: Northrop Grumman (Prime), Raytheon, L3 Comm

Power Plant: Rolls Royce-North American F137-RR-100 turbofan engine

Thrust: 7,600 pounds

Wingspan: 130.9 feet (39.8 meters)

Length: 47.6 feet (14.5 meters)

Height: 15.3 feet (4.7 meters)

Weight: 14,950 pounds (6,781 kilograms)

Maximum takeoff weight: 32,250 pounds (14,628 kilograms)

Fuel Capacity: 17,300 pounds (7847 kilograms)

Payload: 3,000 pounds (1,360 kilograms)

Speed: 310 knots (357 mph)

Range: 8,700 nautical miles

Ceiling: 60,000 feet (18,288 meters)

Armament: None

Crew (remote): Three (LRE pilot, MCE pilot, and sensor operator)

Initial operating capability: fiscal year 2011 (Block 30); fiscal year 2014 (Block 40)

Inventory: Active force, 20

C.1.5 MQ-9 Predator B/Reaper

General Characteristics [17]

Primary Function: Intelligence collection in support of strike, coordination, and reconnaissance missions

Contractor: General Atomics Aeronautical Systems, Inc.

Power Plant: Honeywell TPE331-10GD turboprop engine

Thrust: 900 shaft horsepower maximum

Wingspan: 66 feet (20.1 meters)

Length: 36 feet (11 meters)

Height: 12.5 feet (3.8 meters)

Weight: 4,900 pounds (2,223 kilograms) empty

Maximum takeoff weight: 10,500 pounds (4,760 kilograms)

Fuel Capacity: 4,000 pounds (602 gallons)

Payload: 3,750 pounds (1,701 kilograms)

Speed: Cruise speed around 230 miles per hour (200 knots)

Range: 1,150 miles (1,000 nautical miles)

Ceiling: Up to 50,000 feet (15,240 meters)

Armament: Combination of AGM-114 Hellfire missiles, GBU-12 Paveway II and GBU-38 Joint Direct Attack Munitions

Crew (remote): Two (pilot and sensor operator)

Unit Cost: \$56.5 million (includes four aircraft with sensors, ground control station and Predator Primary satellite link) (fiscal 2011 dollars)

Initial operating capability: October 2007

Inventory: Total force, 104

C.2 UAS Mishap Papers

C.2.1 Hundley et al

Joshua Hundley Msc Report [64] <http://hdl.handle.net/10415/3360>

Unmanned Air Vehicles (UAV): Safety Event Prediction, and Classification[25]

Discussion of handling qualities from MIL-STD-1797B and their applicability to UAS. Gust response for 4 nominal UAS assessed and results compared to requirements in MIL-STD-1797B.

C.3 Data Sources

There are two data sources used in the analysis by Hundley et al [64]

United States Air Force (USAF) SIB (Safety Investigation Board) [181]

United States Air Force (USAF) AIB (Accident Investigation Board) [24]

C.3.1 Safety Investigation Board Reports

Safety Investigation Board reports are made for each mishap by the USAF but not released. The USAF released the 2000-2009 SIB reports for the aircraft in this study with the mishap class, year, type, cause, & some minor additional details. Short one line descriptions of the mishap was all that was released. They were released to Auburn University & Hundley and during discussion Hundley has stated that they are cleared for further distribution. See Table 1 of [25] for an example of the SIB data made available. The total dataset is with this report.

C.3.2 Accident Investigation Board

The summaries from the Accident Investigation Board report are published here.

<http://usaf.aib.law.af.mil/>

The AIB investigates class A mishaps only. Class A mishaps are those where the damage is valued at \$1million or greater, there is a fatality, or the aircraft is a not repairable.

Hundley summarised and classified these reports and the resultant dataset can also be found attached to this report. There have been some issues with classification however so it may be advisable to ensure that the readers understanding of a

failure cause or mode is the same as that of Hundley before basing any decisions on the statistics derived from them.

C.3.3 Mishap Classification

The USAF uses a system of mishap classification.[201]

Class A : \$1,000,000 or More in Damages, loss of Aircraft or fatality

Class B : \$200,000 to \$999,999 in Damages

Class C : \$20,000 to \$199,999 in Damages

Class D : \$1,000 to \$19,999 in Damages

Class E : Less than \$1000 in Damage

C.4 Aircraft Featured in Study

The study extended from 2000 to 2009 and examined 253 mishaps. The aircraft in this study are the Predator, the Global Hawk, & the Reaper. All numbers are from Hundley et al(2012)

C.4.1 MQ-1/RQ-1 Predator

<http://www.ga-asi.com/products/aircraft/predator.php>

Total Mishaps (A-E): 199

Total Class A mishaps: 65

Total fleet flying hours: 590,779 flying hours

Mishap rate (A-E): 33.7 per 100,000 flying hours

Class A Mishap Rate: 11 per 100,000 flying hours

C.4.2 RQ-4 Global Hawk

<http://www.as.northropgrumman.com/products/ghrq4a/index.html>

Total Mishaps (A-E): 17

Total Class A mishaps: 4

Total fleet flying hours: 31,596 flying hours

Mishap rate (A-E): 53.8 per 100,000 flying hours

Class A Mishap Rate: 12.7 per 100,000 flying hours

C.4.3 MQ-9 Predator B/Reaper

http://www.ga-asi.com/products/aircraft/predator_b.php

Total Mishaps (A-E): 34

Total Class A mishaps: 10

Total fleet flying hours: 53,075 flying hours

Mishap rate (A-E): 64 per 100,000 flying hours

Class A Mishap Rate: 18.8 per 100,000 flying hours

Data Sources

United States Air Force (USAF) SIB (Safety Investigation Board) [181]

United States Air Force (USAF) AIB (Accident Investigation Board) [24]

UAS (Unmanned Aircraft(or Aerial) System) have been in operation in one form or another for many years. The desire for an aircraft which can perform tasks without the need for a pilot on board is very attractive for tasks which are one of the 3Ds (Dull, Dirty or Dangerous). Classically these have been regarded as ideal tasks for robots to perform but utilization is limited due to the current cognitive limits of unmanned vehicles. It is only with the advances in microelectronics that they have started to make these vehicles become more useful for operations. However excessive supervision requirements are part of what currently hinders their use. Despite the sophistication of some UAS they lack the intuition and common sense generally displayed by humans. In addition to the three classic tasks robots are intended for a 4th D, Duration. Mission duration of some UAS is longer than

manned aircraft can commonly endure. This raises additional problems as aircraft designed for long endurance operations can be difficult to fly. These factors have all contributed to UAS acquiring a poor reputation for safety. If large UAS are to enter civil operations then the risk of damage to themselves and others needs to be reduced. The USA has invested heavily in this sector and the most successful large UAS are all designed and made in the USA, Predator, Global Hawk, and Reaper. To understand and address the specific issues which face UAS we need to learn from the experience the USAF has gained operating these aircraft. Mishaps and reliability of these aircraft systems was previously discussed well by the US Secretary of Defense.[202] NASA has also made a study of the mishaps of th unmanned aircraft which it has used in its research which has been summarized in a book entitled Crash Course.[203] The number of UA lost to mishaps can clearly be seen in the Class A mishap reports published by the USAF Safety Centre.[24]

As of 2009 the USAF had accrued a total of 675,450 flying hours with the Predator, Global Hawk & Reaper. A study by Hartfield et al[25] published last year went through the mishap data for these aircraft. The mishaps were classified by the

TABLE C.1: USAF UAS Class A Accrued Mishaps

Name	Predator	Reaper	Global Hawk
Designation	RQ-1/MQ-1	MQ-9	RQ-4
2000	1	0	1
2001	5	0	1
2002	12	0	3
2003	14	0	3
2004	20	0	3
2005	30	0	3
2006	35	2	3
2007	42	3	3
2008	52	6	3
2009	65	10	4
2010	72	11	5
2011	84	11	5
2012	88	11	5

USAF(Redmond, 2008) and the study further examined their data to identify mishap cause and system. The total accrued mishap rate was 57.3 mishaps (A-E) per 100,000 hours. The rates for Global Hawk & Predator B/Reaper are projected as during the period of the study the respective fleets did not exceed 100,000 flying hours.

Individual mishap rates (classes A-E, all categories)[25]

- Predator 33.7 Mishaps per 100,000 hours
- Predator B/Reaper 64 Mishaps per 100,000 hours
- Global Hawk 53.8 Mishaps per 100,000 hours

Accrued Class A mishap rates (all categories)[25]

- Predator 11 per 100,000 flying hours
- Predator B/Reaper 18.8 per 100,000 flying hours
- Global Hawk 12.7 per 100,000 flying hours

The largest single mishap cause identified was reliability at approx. 57% of class A mishaps.[25] This was dominated by power and propulsion reliability which should perhaps be expected as these are all single engine aircraft. These three aircraft all share a number of common features as well as being unmanned. They all feature high aspect ratio wings, a single engine, satellite modem for beyond line of sight control, & long endurance. These aircraft cannot currently refuel in flight and thus this long endurance suggests a large take-off weight fuel fraction. Over the 10 years of this study (2001-2009) there were 79 Class A mishaps but only 11 Class B mishaps (\$200,000 to \$999,999 in Damages). There are 61 class C mishaps over the period in question. This could suggest UAS are more prone to complete write off or low damage than a manned aircraft. It was suggested by Hartfield that these UAS may be susceptible to single item catastrophic failure due to their design.[25] The U-2 manned reconnaissance aircraft shares some of the features of these unmanned aircraft and also has a low rate of class B mishaps compared to its class A mishap rate.[204] The U-2 similar features include a high aspect ratio wing and a large fraction of its take-off weight is fuel. The very low wing loading during landing makes these aircraft increasingly susceptible to gusts

as fuel is burned and contributes to difficulty during the final touchdown phase due to ground effect. UA suffer more from this as their wing loading can be as low as a third of that expected for a manned aircraft.[205] The study revealed that these aircraft suffered 43 mishaps categorized as Heavy Landing during the period under study of which 15 were Class A mishaps.

Heavy Landing Mishap Rates[25]

- 6.4 Mishaps (A-E) per 100,000 flying hours
- 2.2 Mishaps (Class A) per 100,000 flying hours

$$p = d \cdot \frac{mr}{t}$$

Algorithm 10: Equation for calculating probability p that the aircraft will have a mishap given some average mission duration d and a mishap rate mr measured per t flying hours

		MTBF (hrs)	Availability	Reliability	Mishap Rate per 100,000 hrs (Series)	Mishap Rate per 100,000 hrs (Model)
RQ-1A/ Predator	Requirement	n/a	n/a	n/a	n/a	20
	Actual	32.0	40%	74%	43	
MQ-1B/ Predator	Requirement	40	80%	70%	n/a	
	Actual	55.1	93%	89%	17	
RQ-2A/ Pioneer	Requirement	25	93%	84%	n/a	281
	Actual	9.1	74%	80%	363	
RQ-2B/ Pioneer	Requirement	25	93%	84%	n/a	
	Actual	28.6	78%	95%	179	
RQ-5/ Hunter (pre-1996)	Requirement	10	85%	74%	n/a	47
	Actual	n/a	n/a	n/a	255	
RQ-5/ Hunter (post-1996)	Requirement	10	85%	74%	n/a	
	Actual	21.2	99%	97%	24	
RQ-7/ Shadow	Actual	n/a	85%	98.8%	191	191

FIGURE C.1: Table H-1 from OSD RPAS Roadmap 2005[206]

TABLE C.2: USAF Mishap Statistics Class A mishaps per 100,000 flying hours converted to mishap probability per mission

Aircraft Designation	Aircraft Name	Class A Mishap Rate	Estimated Mission Duration	Class A Mishap Probability per mission
RQ-4	Global Hawk	12.7	36	0.004572
MQ-1/RQ-1	Predator	11	36	0.00264
U-2	Dragon Lady	5.51	10	0.000551
F-22	Raptor	7.34	4	0.0002936
F-16	Viper	3.55	4	0.000142
A-10	Thunderbolt II	2.07	4	0.0000828

Mishap histories and fleet flying hours for these three aircraft can be found at

Predator [MQ-1 RPA Mishap History](#) [207]

Global Hawk [RQ-4 RPA Mishap History](#) [208]

Predator B/Reaper [MQ-9 RPA Mishap History](#) [209]

C.5 Study Discussion

The Hundley study was funded to examine the mishap data and see if control by internet was a causal factor in UAS mishaps. Hundley contends that it was found that this not a significant factor in UAS mishaps, however this may be an over interpretation of the available data. Hundley identified eight Class A mishaps and 5 B-E mishaps which his team agreed were caused by lost link. This does not actually rule out the effects of a satellite link contributing to any of the other mishaps. The AIB & SIB reports do not indicate one way or the other.

C.5.1 Mishap Categorisation

Hundley et al split the mishap data into separate categories to try to come to an understanding of the causes behind UAS mishaps. A panel was formed to categorize the mishaps consisting of unbiased graduate mathematician senior aerospace

analyst Industry airworthiness analyst The panel then assessed each mishap and put it into one of these categories.

The Mishap Cause Categories[\[64\]](#)

- Lost Link
- Reduced Perception
- Hard Landing
- Reliability
- Maintenance Error
- Pilot Error
- Environment
- Reliability (consisting of)
 - Power Plant
 - Auto Pilot
 - Electrical
 - Hydraulic
 - Structural

A criticism of Hundley et al is that there was no written explanation of how his panel chose which category to assign to each mishap. In one instance they assigned half of the mishap to lost link and half to reliability. This was a class C mishap and only a one line descriptor from the SIB was available to describe the mishap. There is also some confusion over the nature of these causes.

C.5.2 Mode vs. Cause

When describing a mishap it can be helpful to keep a clear distinction between the mode and the cause of the mishap. These are two separate and useful concepts. e.g. A heavy landing causes major damage to the aircraft while attempting to land. The mishap was caused by an oscillating elevator servo. Mode: Hard landing Cause: Malfunctioning controls. In this example the system which needs attention would be the control system. The control system may need a damper to keep it from running away during high frequency pilot input. Improving the

undercarriage would not have prevented the mishap. This distinction is not addressed by Hundley. Hard Landings are listed as a mishap category but we do not know which landings were performed by an autopilot and which were flown remotely by a human pilot. Therefore we cannot distinguish human error from system error. Hundley himself lists heavy landing as a composite of other issues. Hard landings are themselves a product of gust sensitivity, insufficient authority, and reduced perception. [64] This suggests that Hundley et al should have classified the hard landings as Environmental, Pilot Error, Reduced Perception, or perhaps Hydraulic. It is understood that Hundley et al could not identify the root cause in many cases as only a mishap mode was listed however this perhaps could have been better communicated by Hundley listing these categories as mishap modes rather than mishap causes. The UAS examined are all likely to suffer from high gust sensitivity as they have high aspect ratio wings and likely a high surface area to mass ratio. This causes a low wing loading which causes the high gust sensitivity.

As of 2009 this was 6.4 Heavy Landing mishaps per 100,000 flight hours The total accrued mishap rate was 57.3 mishaps (A-E) per 100,000 hrs The combined flight time of all 3 at that point of the study was 675,450 hours

Accrued individual rates all classes

- Predator RQ-1/MQ-1 33.7 Mishaps per 100,000 hrs
- Predator B/Reaper MQ-9 64 Mishaps per 100,000 hrs
- Global Hawk RQ-4 53.8 Mishaps per 100,000 hrs

Over the 10 years of this study (2000-2009) there were only 11 Class B mishaps (\$200,000 to \$999,999 in Damages). There are 72 B & C mishaps over the period in question. This could suggest UAS are more prone to complete write off as opposed to low damage than a manned aircraft where the USAF data seems to indicate Class B rate is close to the Class A mishap rate. U-2 shares this feature of a very low B rate compared to the A rate.[204] Perhaps this is a feature of long endurance high aspect ratio aircraft? If a mistake is made recovery is more difficult?

C.6 Discussion

The \$450 million estimate for the cost of mishaps is a low estimate I made based on the cost incurred values listed in the associated Class A Mishap Reports. Further detail on the class A mishaps can be gathered from the summaries in USAF Safety Centre website.[181] Hundley et al summarized the findings into their spread sheet attached for their classification effort. During a teleconference with Hundley one major source of discussion was the method used to classify mishaps into their categories. The methodology was one of independent analysis by the three members of their committee and then where the independent classifications disagreed the analysts discussed and agreed the correct classification. No record was kept of reasons or it was not fully explained in his MSc and we think has skewed the results. It can plainly be seen however that UAS do experience mishap the same as any other aircraft type and may have higher mishaps rates than might be expected given their mission profile. Due to their long mission duration it might be more appropriate to examine mishaps per mission as a comparison to manned aircraft but this is a metric which is not available for these aircraft as although the total fleet flying hours are published with the number of mishaps the total number of missions is not included which precludes that type of comparison.

