Simulation Fidelity, Abstraction and Resolution in Real-Time Multi-objective Optimisation of Air Traffic Complexity

Author:
Amin, Rubai

Publication Date:
2016

DOI:
https://doi.org/10.26190/unsworks/18956

Simulation Fidelity, Abstraction and Resolution in Real-Time Multi-objective Optimisation of Air Traffic Complexity

Rubai Amin

B.Eng. Australian National University, Australia

A thesis submitted for fulfilment of the requirements
for the degree of Master of Science at the
School of Engineering & Information Technology
University of New South Wales, Canberra

Abstract

Understanding the role resolution, abstraction and fidelity play when solving problems is critical to the quality of decisions produced by automated systems. Resolution is the lens to see what is relevant and what is not in a system. Abstraction offers a mechanism to simplify systems by eliminating those factors that are not relevant to the phenomena of interest. Fidelity is a decision on the level of details in the data we need to have on those factors that are relevant.

In particular, in real-time time-constrained environments, it is important to understand the relationship between resolution, abstraction and fidelity on the one hand, and the speed and accuracy to obtain a decision on the other hand.

In this thesis, we will explore the effect of the level of resolution, abstraction and fidelity of simulators on decisions in the context of air traffic control. We design and use four simulators with different levels of abstraction and fidelity and compare their operation and output. We model reality with a very high resolution simulator that works at a higher level of fidelity than those used for comparison. This allows us to have a ground-truth to compare against.

We then evaluate the effectiveness of the four simulators on optimising air traffic controllers task load in real-time. Each simulator is used to perform look-ahead operations within a multi-objective optimization algorithm to identify an aircraft-specific action to either reduce or increase complexity. Given that an air-traffic scenario has a minimum energy required to perform the task, the optimization finds opportunities to load-balance the workload over the time horizon of the scenario. This load balancing causes upward and downward shifts of complexity. This phenomenon is analysed in details in the thesis.

Despite that a simulator may produce a large deviations from reality, if these deviations are systematic, we can predict it with a static model like an artificial neural network and use the prediction to correct for the simulator's deviation. We conduct a series of analysis using artificial neural networks and linear regression to study the nature of the deviations.

In summary, this thesis demonstrates that decisions on resolution, fidelity and abstraction have a great impact on performance. This impact can be studied and quantified. If used appropriately, it offers an evidence-based rational for the modeller

to justify decisions made on resolution, abstraction and fidelity.

Keywords

Air traffic management, Air traffic simulation, Simulation fidelity, Differential evolution, Airspace complexity, Machine learning

Acknowledgement

Undertaking this Masters has been a truly life-changing experience and it would not have been possible to complete without the support and guidance I received from many people.

I would like to say a very big thank you to my supervisors Dr Hussein Abbass and Dr Jiangjun Tang for their patience and their tireless efforts. Without their guidance and constant feedback this Masters thesis would not have been achievable.

I would also like to thank my family, friends and colleagues for their endless support throughout the entire process.

Certificate of Originality

I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person, nor material which to a substantial extent has been accepted for the award of any other degree or diploma at UNSW or any other educational institution, except where due acknowledgement is made in the thesis. Any contribution made to the research by colleagues, with whom I have worked at UNSW or elsewhere, during my candidature, is fully acknowledged.

I also declare that the intellectual content of this thesis is the product of my own work, except to the extent that assistance from others in the project's design and conception or in style, presentation and linguistic expression is acknowledged.

Rubai Amin

Contents

List of Figures

List of Tables

List of Acronyms

ATM	Air Traffic Management
ADS-B	Automatic Dependent Surveillance - Broadcast
ATCs	Air Traffic Controller/s
ATOMS	Air Traffic Operations & Management Simulator
BADA	Base of Aircraft Data
CD&R	Conflict Detection and Resolution
CPA	Closest Point of Approach
DE	Differential Evolution
EC	Evolutionary Computation
ICAO	International Civil Aviation Organization
NN	Neural Networks
ROAD	Rate of Accelerate and De-accelerate
ROCD	Rate of Climb Descent
ROHC	Rate of Heading Change
TAS	True Air Speed
TCAS	Traffic Collision Avoidance System
TOC	Top of Climb
TOD	Top of Decent

List of Publications

Peer-reviewed publications arising from research work conducted in this thesis are listed chronologically below (latest to earliest):

Conference Publications

1. Amin, R., Tang, J., Ellejmi, M., Kirby, S., & Abbass, H. (2014). Trading-off simulation fidelity and optimization accuracy in air-traffic experiments using differential evolution. In *Evolutionary Computation (CEC), 2014 IEEE Congress on* (pp. 475-482). IEEE.

2. Amin, R., Tang, J., Ellejmi, M., Kirby, S., & Abbass, H. (2014). An evolutionary goal-programming approach towards scenario design for air-traffic human-performance experiments. In *Computational Intelligence in Vehicles and Transportation Systems (CIVTS), 2013 IEEE Symposium on* (pp. 64-71). IEEE.

3. Amin, R., Tang, J., Ellejmi, M., Kirby, S., & Abbass, H. (2013). Computational red teaming for correction of traffic events in real time human performance studies. In *USA/Europe ATM R&D Seminar, Chicago.*

Chapter 1

Introduction

1.1 Overview

Over the last few decades, increases in commercial air travel and air freight movements have seen the level of global air traffic continue to grow year after year (ICAO, 2004; Airbus, 2014). It is estimated that commercial air travel volumes, in terms of passenger-kilometres travelled, will double within the next 15 years (Airbus, 2014), potentially leading to higher density of air traffic. In response to the continued growth in global air traffic, international, regional and national organisations; researchers and commercial entities have continued to evolve the air traffic management system by introducing new concepts, new air traffic management systems and adaptations of existing airspace designs and procedures (Loft *et al.*, 2007). We have seen many new automation tools and procedures that have been developed to assist the human air traffic controllers and pilots in maintaining an efficient and safe flow of air traffic (Ozeki, 2014).

Computer simulation has become an integral part of the development, and often implementation, of new tools and procedures in several areas of the air traffic control and management industry (Chen and Cheng, 2010). The continued development and implementation of computer simulation oriented tools has also seen the development of the many complex air traffic simulation environments. The more complex the environment the more financial, personnel, time and hardware resources are required to develop and operate the system. Given the safety critical nature of some of the applications in the air traffic domain, such as collision detection and resolution, it is often justified in favouring more complex systems as the safety of many people are potentially dependent on the accuracy of the system.

In some applications, though, the highly complex simulation systems may not be desired due to their computationally demanding nature. Examples of applications areas where high complex simulation systems may not be desirable include those requiring real-time prediction and those incorporating optimisation. In these cases, particularly those with real time needs, speed is an important factor which can be influenced by the fidelity of the simulation.

In this thesis, we aim to explore the effect of simulation complexity, more specifically the simulation fidelity, has on the simulation system in the air traffic management domain. First we will investigate the aspects of air traffic simulation system which can influence the fidelity. Then we will evaluate the use of simulation systems of different levels of fidelity for tactical air traffic operations. This will be done by developing a system to adjust the air traffic controller's expected workload using multi-objective optimisation. Finally we will evaluate some methods which may be used to minimise the effect of simulation fidelity on the predictions provided.

1.2 Motivation

With an increasing demand for simulation of highly complex systems, there has been a trend to build simulators as close as possible to a real world process for the prediction of future events or states (Hancock *et al.*, 2008). As technology advances, new iterations of more advanced simulators get developed which provide us with a new suite of more accurate replications of the real world. These new versions require additional resources (such as manpower, money and hardware) to develop (Hancock *et al.*, 2008; Hughes and Rolek, 2003). But with the appropriate identification of the degree of detail required for these simulators, it may be possible to maintain an acceptable level of accuracy for these simulators while minimising the required resources.

High fidelity representations that do not directly effect the objectives of the simulation may be diverting resources from where it is needed most (Hughes and Rolek, 2003). The level of fidelity is one of the most important factors when assessing the quality of a simulation and is a big factor in determining the cost-effectiveness of a simulator and simulators with excessive levels of fidelity may not be as cost-effective as those with lower levels of fidelity (Hancock *et al.*, 2008). This is particularly important for simulators dependant on computer models as excessive levels of fidelity could result in unnecessary computationally intensive environments, leading to higher hardware requirements and more time for completion (Nikoletseas *et al.*, 2008). This may become problematic for real time applications as predictions are required in a timely manner. In some applications areas it may be possible to develop simulators which trade-off levels of fidelity with some loss of result accuracy (Rodriguez, 2008). This presents us with a dilemma that can be visualised by the plot in Figure 1.1. On the one hand we have a simulation model, M_1, which has a high level of fidelity, low error but a long completion time. While on the other hand we have two models, M_2 and M_3, with lower levels of fidelity, higher levels error and shorter time for completion. We can also see that the time required for completion for M_1 is longer than the cut off time for the application, which means that the simulation finishes and provides a prediction later than when it is required. There may be methods by which the completion time for M_1 may be shortened, such as distributed computing, so that it is lower than the cut off time without changing the level of fidelity. In many cases, however, this may be very difficult to achieve or even impossible to implement. If that is the case, then other methods may have to be considered. These other methods include selecting alternate simulation models by trading off the level of fidelity with the level of error and time for completion and selecting a model which meets the time requirements and is within an acceptable level of error.

One area which can benefit from such a trade-off is the aviation industry, espe-

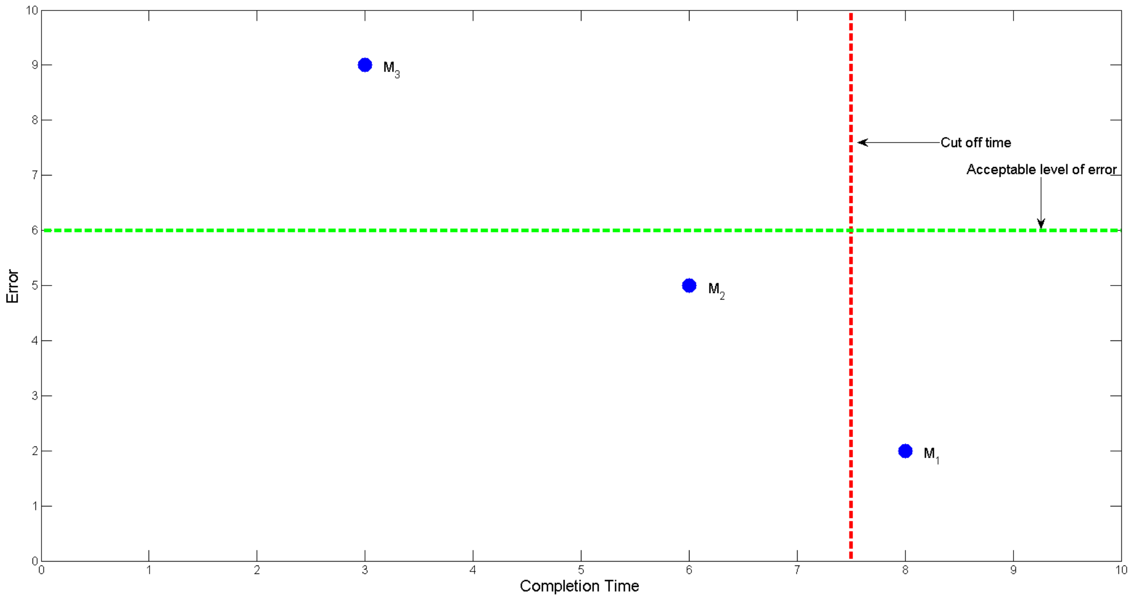


Figure 1.1: Comparison of simulation run time and error for three models with different levels of fidelity.

cially in air traffic management. The increases in global air traffic has shown that there is a need for the development of new automation tools and procedures to help the human air traffic controllers deal with the ever expanding traffic loads (Kuchar and Yang, 2000). Developing lower fidelity air traffic simulators may help reducing the resources required to develop and operate new evaluation tools and procedures and also potentially provide a platform for tools to operate faster while still providing an approximate, but still acceptable, solution.

1.3 Research Questions and Hypothesis

Simulation is used in a wide range of areas and applications, but how well the simulation operates (in terms of the project aims) is dependent on how well its level of fidelity, abstraction and resolution has been identified and implemented. So, in this thesis we wish to specifically answer the following research question:

What is the role of fidelity in simulation for tactical air traffic operations?

Our hypothesis is that by only selecting a sub-set of components of varying levels of fidelity to incorporate into a simulation system, it should be possible to produce acceptable results for the domain. Our main objectives is to develop and use a set of simulators of varying levels of fidelity to simulate air traffic and investigate the effect of fidelity when compared to higher fidelity simulators or the real world.

In order to answer the main research question, a number of other related sub-questions will need to be investigated:

1. **How to influence fidelity decisions in an air traffic simulator? What are the important fidelity validation indicators in an air traffic simulator?**
   The air traffic management system is composed of several important subsystems. Simulators in the air traffic domain are developed to explore different aspects of the domain, which leads to a wide range of simulators with varying levels of fidelity for each subsystem. For example, some traffic flow simulators use simple back box approaches, while the more safety critical applications, such as collision detection, may use high fidelity simulators. The level of fidelity for the different subsystems of the air traffic management system is dependent on the aims of the application. The success of any air traffic simulator will be dependent on the identification of the appropriate subsystems which contribute to the aims for the simulator and also their method of implementation. A simulator with a too high level of fidelity may hinder its effectiveness, while a too low level of fidelity may not achieve quality results. Ultimately the success of the simulator will be determined by how it performs in comparison to existing simulators and the real world. Reducing the fidelity or abstraction may improve the performance of a simulator (speed and computation), but if it can not provide accurate results then it may not be useful in many applications. For this reason a through investigation of the validity of the results from the simulation is required.

2. **What is the role of fidelity on air traffic complexity estimation?**

With the use of simulators of different levels of fidelity, it is expected that there will be some variations in the results obtained when they are used for simulation of air traffic. If the simulators are used as part of an application in a real world operational or tactical environment, the assumptions, limitations and quirks of the simulators and their effects on the results must be first fully understood before implementation. These assumptions, limitations and quirks may be a result of the decisions taken when deciding on the level of fidelity for the simulator. Without understanding the effect of fidelity on the results obtained from the simulator, we may be led to making incorrect conclusions.

3. **What steps can be taken to minimise the effect of fidelity on air traffic complexity estimation?**
   We know that using simulators of different fidelity may produce different results. However if we can develop a method by which to quantify these deviations or identify conditions under which these deviations occur we can produce results which are more accurate. If a methodology can be developed which can be used to accurately and consistently quantify the deviations, we are able use to simulators of different fidelity with more confidence.

1.4 Organization of the Thesis

This thesis consists of six chapters and is organised as follows: In Chapter 1, an introduction to the thesis is presented. First an overview of the research field is provided, followed by the motivation and the research questions addressed in the thesis. The chapter concludes with a list of scientific contributions resulting from the work presented in this thesis.

In Chapter 2, a background of the research conducted is provided. First we discuss the processes and considerations involved in developing simulation models. Next we discuss the present day air traffic environment and discusses some issues effecting the modern air traffic management domain. Finally we discus the use of simulation in modern air traffic management systems.

In Chapter 3, the design and development of several air traffic simulators of different levels of fidelity is explained along with their validation. Firstly, a description of the air traffic simulation architecture is presented along with the airspace model, geographic reference, atmospheric modelling and the computations of the aircraft performance and trajectories are also presented for each simulator.

In Chapter 4, we present a methodology for adjusting the expected workload of an air traffic controller by using a system to adjust airspace complexity in real-time using multi objective optimisation and shadow simulation. This methodology is experimented with several different configurations and the results of this experimentation are summarised.

In Chapter 5, we explore methods by which we can predict future airspace complexity. We use multiple linear regression, neural networks and simulation to predict future airspace complexity using current air traffic characteristics. We also investigate a method by which we can adjust the prediction obtained from low fidelity simulation in order to produce a more accurate prediction of the airspace complexity.

In Chapter 6, the main findings from this thesis is summarised. The chapter concludes the thesis with a discussion of possible future research directions.

1.5 Original Contributions

A list of the scientific contributions arising from this thesis is given below:

- Several shadow simulators are designed which are capable of simulating air traffic in any airspace in the world and are capable of simulating air traffic starting from intermediate points in their flights based on positioning data (Chapter 3). These simulators are designed with different levels of fidelity and we explore the effect of the different design decisions taken to influence the level of fidelity of these simulators on the results obtained from these simulators.

- A methodology is developed for adjusting air traffic controller's expected workload in real-time using multi-objective optimisation and shadow simulation with simulators of different levels of fidelity (Chapter 4). The ATC's expected workload is adjusted by using the optimisation system to generate manoeuvres for the aircraft to deviate from their flight plans in order to achieve a target level of complexity in the airspace. Simulation is used to determine the airspace complexity resulting from the implementation of the manoeuvres while goal programming is used to evaluate the effectiveness of the manoeuvres for reaching the predefined target levels of airspace complexity.

- We explore several methods for real-time prediction of airspace complexity for sectors comprising the entire Australian airspace (Chapter 5). These methods are capable of simultaneously predicting the airspace complexity in multiple sectors in the Australian airspace.

- A methodology is developed for adjusting the prediction of airspace complexity obtained from low fidelity simulators using the current air traffic conditions (Chapter 5). Using this method we can adjust the prediction obtained from the low fidelity simulators to obtain a more accurate estimate of the airspace complexity and overcome the deviations which are introduced as a result of the use of the low fidelity simulators.

Chapter 2

Background

2.1 Simulation

Simulation is a tool that is used to reproduce the behaviour of a real world process in order to evaluate the performance of the system under different configurations over a period of time when it is undesirable to experiment with the system itself (Banks, 1998; Maria, 1997). Simulators can be run in the physical form, such those used to create interactive training environments (eg. environments for training military personnel or fire fighters), as mathematical and computer models (such those used for environmental and climate predictions), or a combination of the two, such as flight simulators for pilots.

Simulation can be used before a new system is built or before an existing system is altered to predict a future state in order to reduce the chances of failure to meet requirements, to eliminate unforeseen problems, to prevent improper resources utilisation and to optimise system performance. Simulation, particularly computer simulation, also allows for time to be sped up so that more events can be simulated in a given time period compared to the real world. It allows us to diagnose, understand and solve problems before they occur and it allows us to ask "what if ...?" questions and investigate their effect on the system. These advantages are particularly useful when dealing with systems where it would be dangerous, impossible or very expensive to observe certain process in the real world (Sokolowski and Banks, 2011). For example, it is more cost-effective and more practical to simulate and evaluate new flight schedules using a computer simulation than it is to implement the schedule and then evaluate it using the real network. For this reason, we can use computer simulation to imitate real world systems to overcome their limitations for experimentation. This makes simulation an important tool in research and development in engineering, science and beyond.

The characteristics of a simulation environment can be described by the following formula (Jacobs and Dempsey, 1993):

$$S = R - T \tag{2.1}$$

where S is the simulation, R is reality and T are the task irrelevant elements. Equation 2.1 shows that for a simulation we need to consider what aspects of a real system needs to be simulated and to what extent they need to be replicated. We then exclude all other irrelevant aspects of the system from the simulation.

2.1.1 Simulation and modelling

A simulation is the operation of a model which is an approximate representation of a real world system (Maria, 1997). A great variety of models exist in a broad range of areas in science and industry. These models can represent systems from financial markets (Spišák and Šperka, 2011), factory assembly lines (Rogalski, 2012) and transport systems (Volf *et al.*, 2011) to more complex systems such as the Earth's climate and environment (Dunne *et al.*, 2012), other planets and even entire galaxies (Teyssier, 2015). Although some of the mentioned systems can be more complex than the other systems, the models used to represent the respective systems, particularly computer models, may not have the same relative complexities. This will be discussed further in later sections below.

Despite there existing a wide range of model types and application domains, Brooks and Tobias (1996) and Maria (1997) provide several key steps that are integral for developing, operating and experimenting with a simulation model. The flowchart in Figure 2.1 outlines these key processes. This figure shows that the process is iterative in order to ensure that the model continues to represent the system in its intended manner.

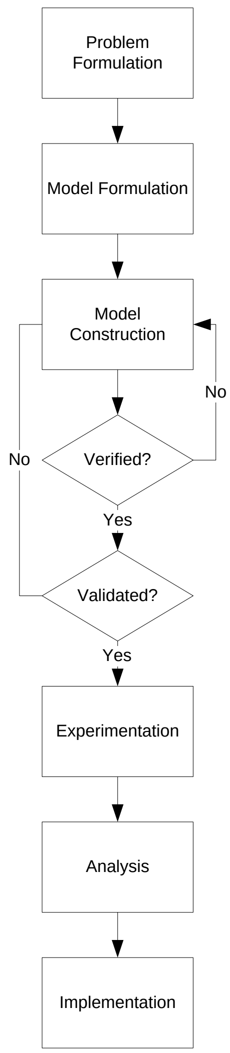


Figure 2.1: A flowchart of the simulation model development process

The process begins with the problem formulation. During this step, the objectives of the study are defined, the features of interest are identified and the basis on which different configurations will be ranked is decided.

Model formulation is a process in which a conceptual model is produced which consists of the specifications and assumptions of the model to be built (Maria, 1997).

This process is an iterative process in which different models are produced with varying levels of fidelity. Once a set of conceptual models have been produced, they are ordered by their level of fidelity and the model most closely adhering to the requirements is chosen. The formulation of the conceptual model involves one of the more difficult aspects of the modelling process: the task of selecting the appropriate level of fidelity. The results of this task has a major influence on the success of the simulation. A model that is too simple could become unrealistic and therefore misleading while building a complex model could require a considerable amount of resources. It is generally harder to understand the relationships contained in a complex model and this makes the interpretation of the results more difficult, possibly leading to incorrect conclusions being drawn (Brooks and Tobias, 1996). The guidelines and principles for selecting the appropriate level of detail are generally vague and what is appropriate is also influenced by the aims of the project.

Next the conceptual model is translated into computer program form using the specifications and assumptions outlined during the model formulation stage (Carson, 2004). This also involves designing the data structures to represent the different components of the model and their relationships.

Before we can conclude the model development stage, we must verify and validate the model with our aims, requirements and assumptions. Verification involves testing whether the model is consistent with its specifications (Sokolowski and Banks, 2011). This means making sure that the computer program for the model correctly implements the conceptual model. Validation is the process of assessing the level to which the model is accurate at representing the system being simulated (Sokolowski and Banks, 2011). This includes operating the model under known conditions and comparing its output with the real system. This step can also uncover the range of operation for which this model is accurate and credible. If the model cannot be verified or validated, we must return to the model development stage and make appropriate changes before again attempting to verify and validate it. Once the model has been successfully verified and validated, we can implement the model.

Once a model has been developed, its performance is evaluated against other models or the real system before it is implemented. Although there are no absolute measures of a model's performance as the evaluation is relative to the project's aims, a meaningful assessment of several models can be made using the following qualitative assessments (Brooks and Tobias, 1996):

- *The extent to which the model output describes the behaviour of interest and the accuracy of the model's results*
- *The ease with which the model and its results can be understood*
- *The portability of the model and the ease with which it can be combined with other models*
- *The probability of the model containing errors*
- *The accuracy with which the model fits the known historical data*
- *The strength of the theoretical basis of the model including the quality of input data*
- *The time and cost to build, run and analyse the model*
- *The hardware requirements of running the model*

One of the key factors which influences the effectiveness of a simulation model is its level of fidelity (Goncalves, 2006). This topic is further discussed in the following section.

2.1.2 Simulation fidelity, abstraction and resolution

Fidelity is a measure of the extent to which a simulation model reproduces the attributes and behaviours of a real system (Hughes and Rolek, 2003). It is very difficult, if not impossible, to completely replicate the real world in a simulation environment (Moon and Hong, 2013) and so we must use simulation models with lower levels of fidelity. Determining an appropriate level of fidelity for a model is one of the most difficult stages of the model development process. The level of fidelity is largely influenced by the aims of the project and the intended application of the simulation. In an ideal world would incorporate as much detail into a model as possible to create a realistic simulator, but this is not always possible or practical for many applications. So we must limit the coverage of the model somewhere. In order to determine an appropriate level of fidelity we need to conduct an analysis of the functions and operations of the real system in the context of the application domain. This allows us to determine the areas in which fidelity can be influenced. For example, in a road traffic flow simulator, modelling the interaction between a car's tyres and the road may not be important. Although this interaction can have some impact on the traffic flow, modelling the precise interaction may not be deemed important in the scope of the project. Through analysis of the system in focus we can determine which components are important for our simulation's purpose, which are of little influence to our results or which can be simplified for our purpose.

A model with the highest coverage of behaviours is referred to as a high fidelity model, while models with less coverage are considered to be of lower fidelity (Abbass, 2014). High fidelity representations of behaviours that do not directly effect the objectives of the simulation may potentially be diverting resources from where it is need most (Hughes and Rolek, 2003). This is particularly important for computer models as excessive levels of fidelity for certain features and process which have a small impact on the simulation results may cause unnecessary computation, leading to higher hardware requirements, more time for completion and a more difficult to understand model (Pachepsky, 2006). When developing a simulation model these consideration need to be kept in mind for deciding an appropriate level of fidelity. The level of fidelity required for one project may be different to that for another project. Determining an appropriate level of fidelity for a simulation is a difficult task as it requires simulation modellers to have sufficient domain specific knowledge of the application area to make decisions which will ensure that the results from the model remain within the bounds of its requirements. The modeller can make decisions to influence level of fidelity of a model in two aspects: the model's level of resolution and it's level of abstraction. Resolution relates to the level of detail of the model, while abstraction determines the model's level of complexity. First the modeller must determine the level of resolution that is appropriate for the simulation (based on the project's aims) and then based on this decision he/she must decide on a level of abstraction that is appropriate for the chosen level of resolution (Abbass, 2014).

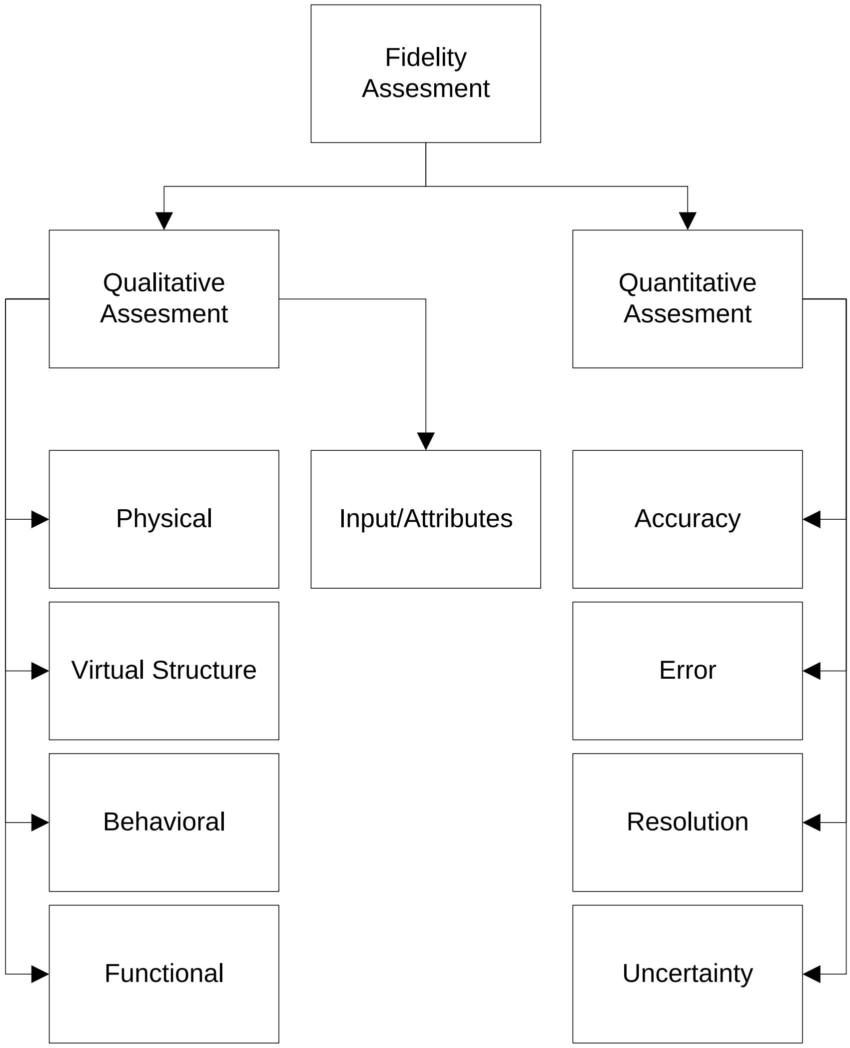


Figure 2.2: Assessment of simulation model fidelity (Goncalves, 2006)

The criteria for assessing the level fidelity of a model can be seen in Figure 2.2. We require both a qualitative and quantitative approach when assessing fidelity. The qualitative assessment is based largely on the level of abstraction of the model while the quantitative assessment is a measure of the model's accuracy, error, resolution and uncertainty. When deciding on the appropriate level of fidelity for a model, we must also consider its time-to-answer and resource usage. The time-to-answer refers to the time required to obtain an answer from when a question is asked and is a measure of the ability of the simulation to provide an answer in a timely manner (Goncalves, 2006). This measure is influenced by the the time required to setup and configure the simulation, the computational time and post-process analysis time. Resource usage refers to both the hardware requirements of the model (for example, the processing power and speed, data storage) and the manpower and skill required to operate the simulation and interpret the results. When selecting an appropriate simulation model for an application, we must trade-off the fidelity, time-to-answer and the resource usage in order to meet project constraints. In the absence of any time or resource constraints we could build simulation models with high levels fidelity that very closely replicate the real world. There are however, very few circumstances or applications which allow for the use of a simulation model with no time or resource constraints, and so we must trade off fidelity in favour of lower time-to-answer and lower resource usage. One of the most common methods of developing simulation models of different levels of fidelity is to alter the level of abstraction of the processes and interactions within the model. Before we can alter the level of abstraction, we must first determine an appropriate level of resolution for the model.

Determining the appropriate level of resolution for a simulation is dependant on the required level of accuracy for the simulation, as determined by the intended scope of the project (Sisti and Farr, 1998). Resolution defines the granularity, or the depth, of the characteristic properties of the real world that are to be simulated (Yilmaz and Ören, 2009). The more detail that is included in a simulation model, the higher that resolution of the model. The simulation of a single car would be of higher resolution than the simulation of traffic on a highway. The level of resolution, however, may not necessarily determine the level of fidelity (Sokolowski and Banks, 2010). For example, consumer level (ie. for casual gamers) and professional level flight simulators may have the same high level of resolution, but the professional level simulator will usually have a higher level of fidelity. An air traffic simulator on the other hand may be of low resolution when compared to a flight simulator, but at the same time the air traffic simulator may also have a high level of fidelity. The difference in fidelity between the consumer and professional flight simulators is typically defined by their level of abstraction. Once we have determined our desired level of resolution, we can then select the level/s of abstraction that are appropriate for the simulation aims.

Abstraction is the process of reducing the behavioural complexity of simulation model components and interactions while still maintaining the validity of the simulation in the context of the project aims (Fishwick, 1988; Frantz, 1995). As real systems cannot be completely replicated by simulation models, even models with the highest level of fidelity are already a somewhat abstracted versions of the real system. These models though can be further abstracted depending on the aims of the use of the simulation.

There are several reasons as to why we may consider the use of more abstract versions of existing models. These abstract models are generally less computationally complex and easier to understand as they include simplifications of components and behaviours (Fishwick, 1988). The process of creating abstract models also often results in a library of models with different levels of abstraction and/or resolution for the same process which can be utilised for different purposes. These models can be ordered into a hierarchy, such as the one shown in Figure 2.3, based on their level of abstraction and fidelity. Each level on the pyramid has a different level of abstraction and fidelity. The model on the lowest level of the pyramid has the highest level of fidelity and the lowest level of abstraction. This model will generally produce the lowest amount of error among the models in the pyramid. The higher the model is on the pyramid, the higher the level of abstraction is for the model. The model at the top of the pyramid has the lowest level of fidelity among the models in the pyramid and the highest level of abstraction.

The different levels of abstraction may introduce different levels of error into

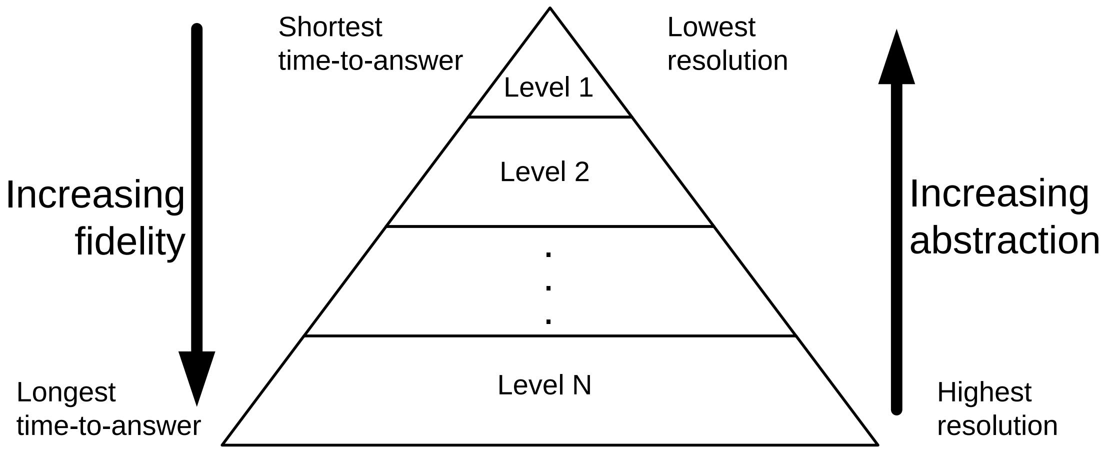


Figure 2.3: Levels of model hierarchy

the results of the simulation. The modeller must consider the positive benefits of reduced runtime and reduced hardware requirements and trade this off with the drop in result accuracy due to the level of abstraction (Rodriguez, 2008). The final choice of the level of abstraction and resolution of a model is dependent on the aims, objectives, and requirements of the simulation. This choice is heavily influenced by the acceptable level of error set by these requirements during the model formulation stage of the development process. As can be seen in Figure 2.4, selecting an appropriate level of abstraction is a matter of trading off the complexity of the model and the level of error in the results. With lower levels of abstraction, we can expect to see small, but acceptable, levels of error. As the level of abstraction increases further, the level of error becomes larger and into the unacceptable range.

The techniques involved in the process of model abstraction can be placed into three categories: model boundary modification, modification of behaviours and modification of model form. Model boundary modification involves the modification of the input variable space, modification of behaviours involves the modification of behaviours within the model and also involves combining some aspects of the model while modification of model form involves the simplification of the input-output transformation within the model (Frantz, 1995). Each of these classes can be divided

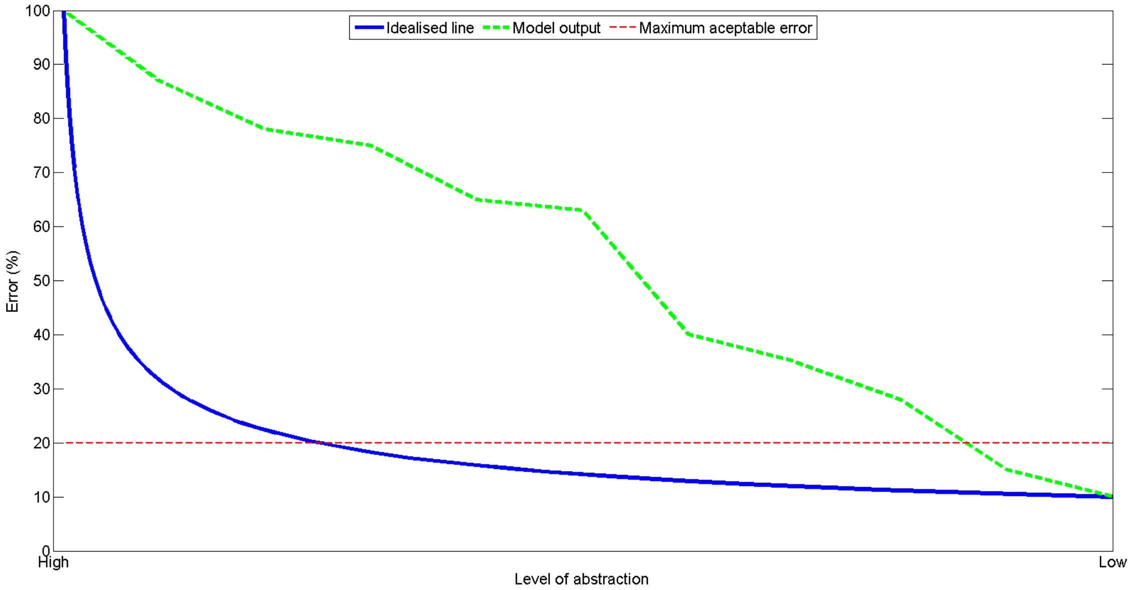


Figure 2.4: Appropriate level of abstraction determination

into further subclasses.

Model boundary modification involves the simplification of the model by modifying input variables, often by eliminating some of the variables as input to the model (Frantz and Ellor, 1996). Explicit abstraction techniques (hierarchies of models and delimit input space) involves the identification of exogenous variables for elimination, while derived abstraction techniques (approximation and selection by influences) involves the determination of the minimum number of input variables required to meet the simulation requirements (Frantz, 1995).

- *Hierarchies of models:* A set of models for components which includes simplified and more complex models for the same process.

- *Delimit input space:* Limiting the domain of inputs provided to the model and thus limiting the area of operation of the model. For example, limiting the model to water temperatures between 0 and 100 degrees, eliminates the need to model the transition from solid or gas states (Frantz, 1995).

- *Approximation:* Eliminating parameters based on the aims of the simulation and providing approximate results.

- *Selection by influences:* Simplifying the relationship between variables.

Modification of behaviours involves aggregating some aspects of the models to modify the behaviour of the internal elements of the model (Frantz and Ellor, 1996).

- *Behaviour aggregation:* Combining states which are irrelevant to the simulation aims

- *Causal decomposition:* Dividing the model into separate components and executing each component individually, but while maintaining the interactions between the components.

- *Aggregation of cycles:* Combining similar states which are irrelevant to the final result.

- *Numeric representation:* Replacing continuous variables by categorical or nominal variables.

- *Temporal aggregation:* Changing the time advance.

- *Entity aggregation:* Replacing lower level entities with higher level entities.

- *Function aggregation:* Combining several functions into one.

Modification of model form is a simplification of the input to output transformation (Frantz, 1995). This process does not necessarily modify the model boundary or behaviour, but modifies the manner in which the parameters are determined.

- *Look-up table:* The output value is determined by retrieving an indexed value of the input.

- *Probability distribution:* Replacing computation with randomly generated values.

- *Linear function interpolation:* Linearly interpolating known data.

- *Metamodeling:* Analysis of input-output pairs by treating the model as a black box.

The decision to use a particular abstraction technique and the extent to which it is used requires a trade-off for the simulation result accuracy, computation time and resources, and so once again is dependent on the aims of the project.

While our discussion has been mainly focused on computer simulation, it must be noted that the issue of determining an appropriate level of fidelity, resolution and abstraction are also important for non-computer based simulations. Training, particularly in the medical field, is one application where non-computer based simulations are often used. Norman *et al.* (2012) discusses several studies into the use of high and low fidelity simulators for training medical students. One such study (Issenberg *et al.*, 1999) investigated the use of a computer controlled manikin for the simulation of sounds relating to several human heart conditions. It was found that the use of this expensive and sophisticated manikin improved the transfer of learning for identifying the heart conditions when compared to learning through traditional ward visits. But another study (Hatala *et al.*, 2008) also found that the use of a low fidelity simulation system which played the sounds on a laptop computer achieved similar results as when using the manikin. From this study we can see that by appropriately analysing the requirements and aims of the simulation we can reduce the complexity of the simulation environment and also the resources required while still achieve comparable results.

The analysis of selecting an appropriate level of fidelity are also important for computer based simulation too. Reshetin and Regens (2003) investigated the dispersion of anthrax spores during a bio-terrorism incident using computer simulation. This study involved the simulation of the dispersion of anthrax spores through a 50 storey building to provide a picture of the expected level of human exposure and surface contamination throughout the building before medical intervention. The simulation model is capable of modelling the interaction between individual anthrax spores and the interaction between the spores and the building's walls and the flow of air through the building. But in order to minimise computation time and provide a quicker prediction of the level of contamination, a number of simplifications where made. This includes modelling every floor of the building to be identical with the same air flow throughout the day and assuming that the distribution of the spores are uniform throughout the floor. In reality, the layout, size and shape of each floor of a building may be different and each floor may have different air flow patterns, but these factors are not the primary focus of the model. The aim was to model the dispersion of the spores through the building and how they interact with the surroundings and so more resources (time for model construction and computation resources at run time) were allocated to more accurately model the dispersion instead of the building.

Selecting an appropriate level of fidelity, abstraction and resolution is also important in the simulation of transportation systems. Many traffic simulation approaches implement macroscopic models which aggregate vehicles at a specific level, such as a single road segment, and model their flow using methods similar to those found in the fluid dynamics domain (Stanica *et al.*, 2011). In macroscopic models, the vehicles are not represented individually and the properties of the vehicles are represented as the mean values of the traffic stream within a segment. There are, however, some limitations inherent with this aggregate approach as it does not allow individual routing of vehicles nor the modelling of vehicle, road and weather conditions; vehicle types and driver behaviour (Sewall *et al.*, 2010). Alternatively, microscopic models allow us to model the behaviour of individual vehicles and their interaction with other vehicles and the surrounding environment, and also allows us to model the behaviour of the operator/driver of the vehicle (Stanica *et al.*, 2011). Sewall *et al.* (2011) proposed a hybrid traffic simulation model which takes advantages of both the macroscopic and microscopic approaches. In this approach the vehicles are simulated and visualised using the aggregated macroscopic approach, but when the visualisation is zoomed to a certain level, the vehicles were simulated using a microscopic approach which allowed for vehicles to be individually simulated and visualised. While this method takes advantage of the more computationally efficient macroscopic approach and utilises the more intensive agent based model only when needed, this hybrid model also has its own limitations. This approach does not allow for a smooth transition of the vehicles from the macroscopic region to the microscopic region as the actual positions of the vehicles are not known at the transition boundary and there is a potential for a mismatch with acceleration and distances of nearby vehicles during the transition. To overcome the limitations of the macroscopic and hybrid approaches, we can use purely microscopic approaches. The microscopic approach allows each individual vehicle to have it's set of properties individually defined. This may include parameters relating to the vehicle's acceleration rate, braking capabilities, intended route, fuel usage, driver behaviour, etc; which allows for finer control of the behaviour of the vehicle. This approach has been widely used for simulating transportation systems (Chen and Cheng, 2010) including in areas as diverse as road traffic management (Wang, 2005; Hernández *et al.*, 2002), railways (Böcker *et al.*, 2001), and shipping (Henesey, 2004). This approach is also widely used in the air traffic management domain (Alam *et al.*, 2008; Volf *et al.*, 2011; Agogino and Tumer, 2012). In the following section we will briefly introduce the modern air traffic management system, then discus the use of simulation within the domain and introduce the problems associated with the use of simulation within this domain.

2.2 Air Traffic Management

The modern air traffic management and control systems have evolved into rather complex systems from their humble beginnings. In the 1930s pilots operated on a "see and be seen" approach where the pilots themselves were responsible for maintaining safe separation from other aircraft by visual inspection. Pilots relied on distance and time estimates for navigation and used distinctive landmarks to verify position and progress (Wise *et al.*, 2012). When the pilots wanted to land at an airport, they would often fly over the airport to assess the wind and traffic conditions and decide how they wanted to land, often in open fields instead of designated run-ways (Nolan, 2010). Throughout this procedure the pilots also had to maintain their own separation and form their own priority queues with other aircraft approaching and departing the airport with minimal communication. As the air traffic increased, larger airports started employing human air traffic controllers. These early air traffic controllers directed the aircraft using flags from a prominent position on the ground. As the air traffic increased even further, new, more advanced communication, navigation and surveillance technologies and procedures were developed and incorporated into the system.

The implementation of radio communication systems in the 1950s introduced a new generation of air traffic control systems and sparked a series of evolutions in the air traffic control system resulting new generations of systems approximately every two decades (Gilbert *et al.*, 1973). Today, electronic instruments, radio communications, GPS navigation, ADS-B data exchange and automation tools for ATCs, among other technologies play a vital role in the air traffic control system. The pilots depend on high accuracy GPS systems along with an array of ground based navigation systems to navigate along designated airways. While air traffic controllers have many automated tools in their arsenal to maintain a safe and efficient flow of air traffic.

Today, much of the world's airspace forms part of a highly structured system composed of rigid, predefined structures or volumes called sectors (Stein *et al.*, 2006). The sectors are defined by historical traffic patterns to ensure that the workload is appropriately distributed to the one or more ATCs responsible for the safe and efficient flow of air traffic through each of the sectors. Throughout the course of a flight, a pilot will typically interact with a number of different ATCs. Before, during and shortly after take-off the pilot interacts with the tower controllers at the departure airport. There may be more than one tower controller (with each controller being assigned a separate responsibility) depending on the size and function of the airport. The tower controllers are responsible for confirming the flight's plan and advising the pilot about movement on the ground and departure procedures; and the climb phase of the aircraft to 3,000-6,000ft, depending on the configuration of the airspace (Nolan, 2010). The aircraft then becomes the responsibility of the terminal area (TMA) controller. During the final portion of the climb phase, the flight is handed-off to an en route controller who is responsible for the aircraft as it travels through his/her designated sector. The flight will remain the responsibility of en route controllers as the flight reaches it's cruise altitude and continues on it's cruise phase. The en route controllers hand-off the flight to the controller of neighbouring sectors as the flight approaches the boundary between these sectors. Each sector comprises of several airways, often called jet routes, which intersect with other airways throughout the sector. The separation of the aircraft at these intersections forms a major part of the responsibility of en route controller's job. As the aircraft descents to it's destination airport, it is handed-off to the TMA controller and eventually the tower controller during the final stages of the approach. Maintaining appropriate separation between the aircraft in the short to mid-term time horizon in different stages of flight is the fundamental task of all controllers involved in the system (Prandini *et al.*, 2011).

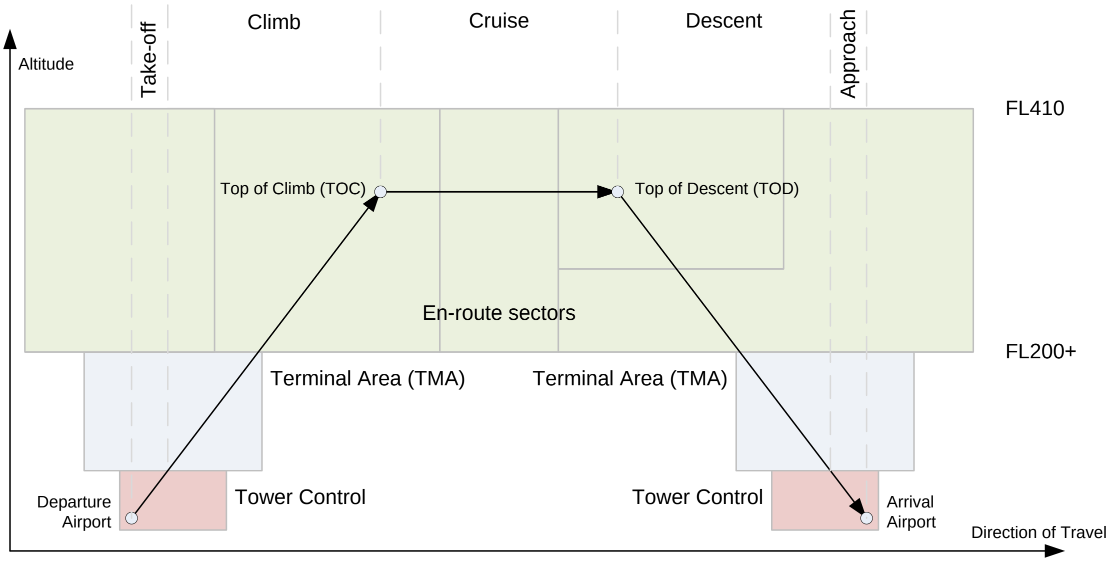


Figure 2.5: Overview of air traffic controller services for a typical flight

Although the main objectives from the early days of air traffic control for the safe, orderly and efficient flow of air traffic still apply today, the air traffic control system has become an integral part of a much broader system - the air traffic management system. The air traffic management system has brought a wider range of objectives and considerations into the air traffic system (Wise *et al.*, 2012). These additional objectives and considerations include emissions, noise, fuel usage and human factors. The inclusion of these considerations has largely been due to the continued increase in air traffic and it's implication on the broader society. In response to the continued increase in demand for air travel, currently development is under way for a new generation of the air traffic control system. This new generation aims to incorporate modern sensing and information technologies and procedures derived from their implementation to provide reliable communications, real-time situation awareness and prompt decision supports (Zhang *et al.*, 2012). These new technologies incorporate many new air traffic management procedures which aim to alleviate the pressure on the strained air traffic control system which cannot physically change as the airspace volume and airport locations are fixed. This includes a move from the current ground-based navigation and communications systems to more reliable satellite based data-link communications and global positioning system (GPS) services and greater use of automation tools as outlined in the plan for the Next Generation Air Transportation System (NextGen) in the United States (FAA, 2014) and the Single European Sky ATM Research (SESAR) project (SESARJU, 2015).

Advances in technology, more and better data and enhanced facilities for computation has seen automation play a vital role in various areas of the air traffic control system (Wise *et al.*, 2012). The aim of automation in the air traffic environment has been to make the air traffic control system more efficient and more capable of handling higher amounts of traffic without sacrificing the existing safety standards. This is achieved by providing tools which can assist the human ATCs reduce their workload. If the traffic density is low, then the ATC can effectively control the traffic manually, but as the traffic grows, manual approaches may lead to inefficient utilisation of the airspace and other tools may become essential (Menon *et al.*, 2004). The continued increase in air traffic has seen the workload for ATCs grow as they become responsible for greater number of movements within similar geographic boundaries. Historically, automation tools have been relatively simple, such as visualisation of flight tracks and data and the notification of conflicts; but with the implementation of concepts such as NextGen and SESAR, a range of more complex tools are being introduced (Bekier *et al.*, 2012). When designing automation tools, two key aspects must be considered: it's accuracy and it's effect on the human users. It is counter-productive to use a tool whose aim is to assist the human ATCs which results in higher workloads for the ATC. The nexus between automation tools and the human ATCs has been investigated for several decades (Wise *et al.*, 2012). Time and time again a common theme emerges from these investigations. While computers are capable of processing large amounts of data and making decisions quickly, the human ATCs can easily become overwhelmed with too much information provided by the tools and begin rejecting or even ignoring the advise provided (Bekier *et al.*, 2012; Crück and Lygeros, 2007b). This could possibly lead to the ATC missing a potentially dangerous scenario.

Like many complex and dynamic systems, the ATC can not temporarily halt the air traffic system to take a break when the workload becomes too high. For this purpose a strategy needs to be in place to shift this load between the human and machines in order to maintain an ideal level of complexity according to the ATC's capabilities (Abbass *et al.*, 2013). Presently, air traffic flow management strategies rely on centralised systems to produce routes for aircraft (Tumer and Agogino, 2007). This is conducted over a large time frame, ranging from one hour to one year in advance and often encompass large regions, such as the entirety of the Australian airspace. This causes the system to be slow in responding to developing localised uncertainties such as adverse weather conditions (eg. storms, volcanic ash clouds) and other exceptional events (eg. aircraft breakdown at airport, emergency landings, 9/11 terrorist attacks). Heidt and Gluchshenko (2012) outlines four groups of uncertainties which effect the air traffic system: human, data, meteorological and equipment. Human caused uncertainties include the actions and decisions taken by ATCs, pilots and ground staff and are influenced by the the psychological and mental capabilities and limitations of the human decision makers. The unavailability of appropriate data makes planning difficult. Meteorological uncertainties include conditions such as fog, wind and storms, while the equipment group includes events such as aircraft breakdown. Among the four groups of uncertainties, the human uncertainty is the most difficult to predict and model (Heidt and Gluchshenko, 2012).

The emergence of uncertain events can potentially lead to local delays as pilots, ATCs and airports scramble to avoid them or deal with their implications (Agogino and Tumer, 2012). The local disruptions can grow to form larger regional congestion that push the dynamics of the system beyond the point of safe operation and negatively affect the performance of the system (Cook *et al.*, 2015). The effects can include exceeding the initially planned level of traffic within the sectors leading to exceeding the capacity of the sector and at the same time exceeding the capabilities of the ATCs allocated to these sectors. In order to handle these uncertainties we require methods by which to predict them and make changes to the state of the air traffic before they occur and cause major disruptions to the system. To facilitate these changes, particularly in a real-time environment, it may be necessary for involved parties (ie. ATCs, pilots, airports, etc.) to participate cooperatively in any such method.

To overcome the possibility of overwhelming the ATC, a concept called subliminal control (Villiers, 2004) has been introduced. The idea is that by making minor speed variations to aircraft in the en route phase (as advised by an automated system to the pilot, working alongside the ATC) it is possible to prevent a potential conflict in advance using available flight data and trajectory prediction. Manual conflict detection and resolution (CD&R), that is the method of ensuring that two or more aircraft do not lose separation by flying too close to each other; results in high levels of workload for ATCs (Galster *et al.*, 2001). Through subliminal control, the speed adjustment may be sufficiently small enough that it may go unnoticed by the ATC, but at the same time reduce their workload due to the conflict being avoided and no other major changes being made to the system or individual flight plans. Drogoul *et al.* (2009) showed that speed changes as large as 12% can go mostly unnoticed by the ATC. Archambault (2005) demonstrated that by making speed changes between the range of -20% and 20% over 4, 6 and 10 minute look-ahead periods that it is possible to reduce conflicts. While Crück and Lygeros (2007a,b) presented a dynamic game approach where subliminal control was used to minimise a cost associates with the risk perceived by the ATC. A dynamical game approach based on Viability Theory was used to solve optimisation problem is selecting the appropriate speed. The risk was modelled as a function of the minimum separation distance between one or more aircraft within a given time frame. Chaloulos *et al.* (2010) examined the use of subliminal control in reducing the risk perceived by an ATC for a several two aircraft crossing situations. Chaloulos showed that by examining aircraft crossing at 45, 90 and 135, it is possible to significantly reduce the perceived risk for a range of minimum separation distances (0 NM, 5 NM and 10 NM) and a variety of time to minimum separation scenarios. This study however only considered two aircraft flying at the same altitude and modelled the ATC's perceived risk as introduced by Crück and Lygeros (2007a). Rey *et al.* (2015) developed a subliminal control method which incorporated economic considerations. This method used a goal programming model to minimise the cost to airline operators (fuel usage and time delay) as part of a mixed-integer linear programming system when determining speed changes in order of avoid conflicts.

In order to implement this subliminal control concept, it is necessary to have a means to predict aircraft trajectories with high accuracy for a period greater than the ATC's own look-ahead time (or prediction horizon) for evaluation of a given situation and identify potential conflict situations early. With higher accuracy longer time horizons for application can be used (Chaloulos *et al.*, 2010). Additionally this method must also be capable of accurately modelling the small changes required in the trajectory in order to appropriately measure the change in risk. As the proposed systems would be implemented in a dynamic real-time environment it is unlikely, due to the uncertainties of the environment, that the prediction would hold true over long periods of time. This means that the system will need to periodically re-evaluate the environment and calculate a new optimal solution.

Studies in subliminal control have primarily limited their focus to speed changes while some effort has been given to flight level changes. There has been no investigation into the use of other methods of conflict resolution such as heading change or combinations of speed, flight level or heading changes.

All of these subliminal control approaches also limited their focus to detecting and resolving conflicts. While their stated aim was to reduce the ATC's workload, none of these studies focused on investigating the effect's on the ATC's workload. Although potential conflicts do have a considerable bearing on the ATC's workload, they are not the sole contributor to increases in ATC workload. The workload experienced by an ATC is a combination of (Majumdar and Ochieng, 2002):

1. The state of the air traffic (configuration of sector, movement of air traffic)
2. The state of the equipment being used (design, ease of use, level of data)
3. The state of the air traffic controller (age, experience, skills)

The interaction between these three factors and the resulting workload experienced by the ATC is complex. The effects of the ATC's experience, skill and the quality of the equipment being used are however, to some degree, influenced by the state of the air traffic (Majumdar and Ochieng, 2002). Bypassing the ATC and issuing changes to the state of traffic directly to a pilot can also result in additional workload for the ATC. Typically an ATC will asses the state of the traffic, predict the future state of traffic, identify any potential conflicts, and then plan and control the traffic according to this prediction and conflict identification. Any state changes that bypass the ATC could potentially have a negative impact on the ATC's workload as they would have to first identify this change and reassess the scenario. Although the ATC is continually reassessing the state of the traffic, changes in the state are usually known before hand or approved by the ATC. For example there is some warning before an aircraft enters the sector and aircraft changing flight level or heading are first approved by the ATC. It may be wise to make the suggestions from the subliminal control system available to the ATC instead of the pilot.

The small changes suggested by these systems are more akin to the requests made by pilot during the course of the flight, particularly during the cruise portion of the flight.

The collective effect of the air traffic events and other sector characteristics at any given time on the ATC's workload is referred to as airspace complexity (Kopardekar *et al.*, 2009). Numerous airspace complexity measures have been introduced in the literature (Hilburn, 2004). It is generally agreed that airspace complexity is influenced by more factors than just the number of aircraft in the sector. Additional factors that have been identified include aircraft separation, closing rates, aircraft speeds, mix of aircraft types, altitude changes and heading changes. A measure of complexity provides a broader picture of the activities within the sector that may effect the ATC's workload when compared to collision risk. The broader picture provided by the airspace complexity is important as the complex environment of air traffic control relies significantly on and is limited the capabilities of the human ATC (Inoue *et al.*, 2012).

In order to generate the actions required to alter the ATC's workload and evaluate their appropriateness to the problem at hand, we require an optimisation system and a simulation system. Optimisation can be a time-consuming activity, especially when simulation is required to evaluate possible solutions. This can pose a problem for real-time applications as a slow optimisation system may not be able to provide a solution in a timely manner or may not be able sufficiently search the solution space. From the survey of literature regarding subliminal control, we have found that the simulation system, ie. the means to predict aircraft trajectories and evaluate the small changes in trajectories, needs to be sufficiently accurate that small changes in trajectories can be properly evaluated. In this case it would make sense to use a high fidelity simulator as they are generally the most accurate. However, high fidelity simulators are also usually more computationally complex, which means each evaluation may take significant time for completion.

2.2.1 Simulation in air traffic management

The modern air transportation system incorporates a large network of airports, airlines, ATM/ATC centres and jurisdictions. The network facilitates their interconnections and interactions with other airports, airlines, ATM/ATC centres and jurisdictions. Each of these system stakeholders have their own duties, goals and objectives to fulfil which ads another level of complexity to the operation of the network. As a result of the multiple levels of complex interactions within the system, a wide range of air traffic simulators have been developed over the past few decades using numerous different approaches with each focusing on different aspects of the system. The application areas of simulation in the air traffic domain are as varied as conflict detection and resolution, noise modelling (Zaporozhets and Tokarev, 1998), cost/benefit modelling for airlines (Bazargan *et al.*, 2013), airspace capacity modelling (Clarke *et al.*, 2012) and for automation tools. The level of abstraction, resolution and fidelity in the developed simulators is varied depending on its application and purpose.

Early air traffic simulators focused on discrete event simulation techniques. These simulator assigned states to various entities which were changed based on events in an event list. The events have an associated time and are triggered to change the state of the relevant entity when the system clock reach that time. For example, in the model developed by Lee *et al.* (2003) there are separate events to indicate that an aircraft is scheduled to depart, has departed from the gate, entered the runway queue, has entered the airspace, etc. The system developed by Lee *et al.* (2003) and other discrete event simulation models rely on a network of queues for the simulation of aircraft from one state to another. This method however is deemed to not be accurate in representing the real air traffic network and too many events are required to be defined and stored to develop and operate an accurate model (Kim *et al.*, 2015).

When modelling air traffic flow problems it has been common to use an Eulreian approach. This approach utilises methods similar to those found in fluid mechanics whereby the airspace is divided into several control volumes and the dynamic behaviour of multiple individual aircraft in a given region is aggregated into the single one dimensional control volumes (Menon *et al.*, 2004). In this method, the characteristics and objectives of individual aircraft are lost which allows for a much simplified model of the airspace as only the flow between the control volumes are modelled. An example of the representation of the air traffic environment for the system developed by Menon *et al.* (2004) can be seen in Figure 2.6. In this figure the airspace is divided in five control volumes (named ATC Center 1 through to 5) and there are several merge and diverge nodes where traffic from several streams may merge into a single stream or diverge into two or more streams. The flow of air traffic between the control volumes can be calculated using a series of linear algorithms. This method may be efficient as the complexity of computation required to simulate the airspace scales with the number of control volumes and not with the number of aircraft (Sun *et al.*, 2007). However, the high level of abstraction provided by the Eulreian approach is not suitable for measuring several metrics such as collision risk or airspace complexity as the positioning and other dynamic behaviours (such as heading and speed changes) of individual aircraft cannot be accurately modelled. The Future ATM Concepts Evaluation Tool (FACET) (Bilimoria *et al.*, 2001) is an air traffic flow developed by NASA and has been used extensively by the FAA and other commercial and academic organisations.

On the other hand, multi-agent based modelling (Weiß, 1999) allows us to represent individual components of the system. This method of modelling facilitates the representation of characteristics and behaviours of individual agents and it's interactions with other agents. The simulation is processed with each agent's local activties based on their local rules for interacting with other agents and the environment (Kim *et al.*, 2015). When modelling the air traffic system, the various components of the system, such as airlines, airports, aircraft, pilots and air traffic controllers; can all be modelled as individual agents. These agents act as autonomous decision makers which make decisions that effect the environment in order to achive their own goals or objects. For example, an airline agent may set it's

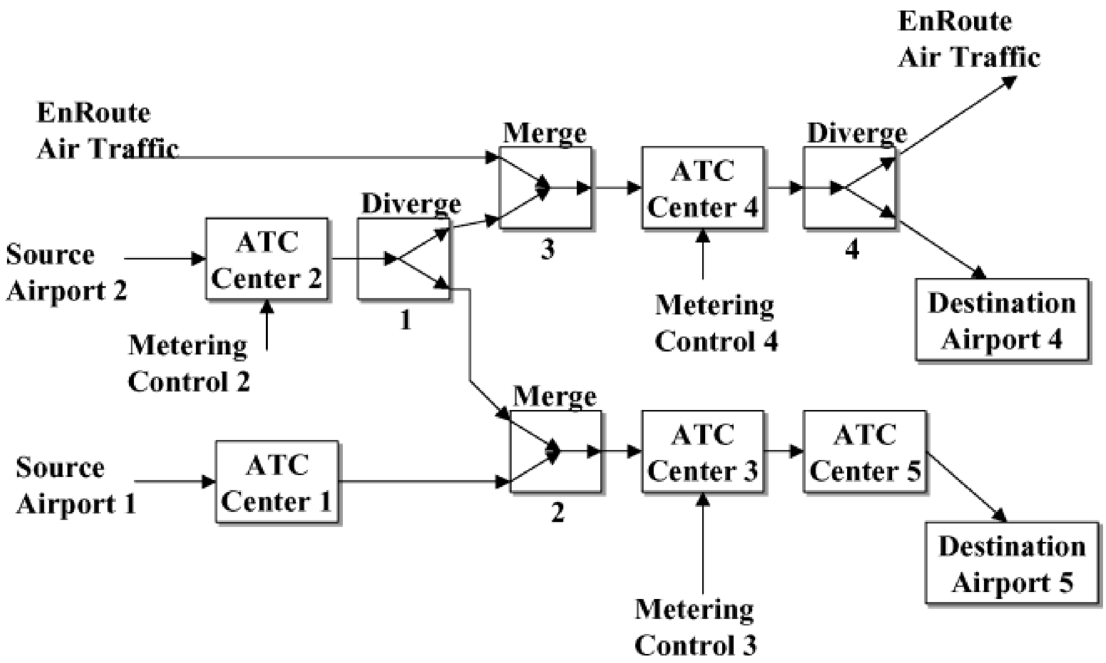


Figure 2.6: Example air traffic environment model used in the Eulerian traffic flow simulation model (Menon *et al.*, 2004)

flight network based on demand from customer agents and availability of aircraft agents. The schedules for these flights might be determined by availability of gates as a result of the goals and objectives of the airport agents and the other airline agents. The operation of the flight itself will be determined by the aircraft agent and it's interaction with other aircraft agents and the air traffic controller agent. As the hierarchy of agents closely maps to real world subsystems, multi-agent modelling presents itself with some advantages for simulating complex systems. Multi-agent modelling also provides some more advantages as different aspects of the system can be modelling with different levels of abstraction.

When using simulation for real time applications, it is important to use a simulator that is capable of providing a prediction or an answer to a problem in a timely manner. A possible solution for overcoming this problem is to trade-off the level of fidelity and using a model which is of a higher level of abstraction. The more highly abstracted model will usually produce predictions with more error than models with lower levels of abstraction, but the completion time for this model will usually be lower. In the air traffic context, a tool using simulation which provides an alert for a potential loss of separation between two aircraft several minutes after the aircraft have left the sector, or even resulted in a collision, is not useful. But at the same time, a tool which produces a prediction very quickly and is poor at detecting loss of separation is also not useful. For this reason need to carefully select models by trading off the level of fidelity and abstractions to meet the time and accuracy requirements of the application.

In the next chapter we will walk through the process of designing air traffic simulation models. In the following chapters we will then use these models to investigate the effect of fidelity in operational environments. First we will use the models to estimate airspace complexity and attempt to adjust the expected workload for the ATC in real time. Next we will discuss several methods which can be used to minimise the error in airspace complexity prediction as a result of using each of the models.

Chapter 3

Designing an Air Traffic Simulator

Work in this chapter has been partially published in the following papers:

1. Amin, R., Tang, J., Ellejmi, M., Kirby, S., & Abbass, H. (2014). An evolutionary goal-programming approach towards scenario design for air-traffic human-performance experiments. In *Computational Intelligence in Vehicles and Transportation Systems (CIVTS), 2013 IEEE Symposium on* (pp. 64-71). IEEE.

2. Amin, R., Tang, J., Ellejmi, M., Kirby, S., & Abbass, H. (2014). Trading-off simulation fidelity and optimization accuracy in air-traffic experiments using differential evolution. In *Evolutionary Computation (CEC), 2014 IEEE Congress on* (pp. 475-482). IEEE.

3. Amin, R., Tang, J., Ellejmi, M., Kirby, S., & Abbass, H. (2013). Computational red teaming for correction of traffic events in real time human performance studies. In *USA/Europe ATM R&D Seminar, Chicago.*

In this chapter we will discuss the process of developing a set of air traffic simulation models with different levels of fidelity, abstraction and resolution by following the processes introduced in the previous chapter for formulating, constructing and verifying simulation models and describe their architecture, design and validity. The overall goal is to design an air traffic simulation model that is suitable for use in a real time environment. This means the model must be able to simulate a given scenario and return an accurate prediction in a timely manner. Real time prediction in an operational environment is a difficult task due to the uncertainty of the environment and it requires a fast methodology. When using simulation for prediction, it is possible to achieve shorter times required for prediction by trading off the simulation model's level of fidelity, abstraction and resolution. Doing so, however, may introduce some error into the prediction. Selecting an appropriate model for an application is a matter of selecting a model with a level of fidelity, abstraction and resolution which meets the time constraints of the application and also produces results within a tolerable level of error. The level of error is determined by comparing the output of the model with the expected output or real world observations. The tolerable level of error is dependent on the application and must be decided as part of the modelling process. In the following sections we will compare and contrast the decisions taken to influence the level of fidelity, abstraction and resolution of four developed simulation models. We will also compare the levels of error that occur in the predictions as a result of these decisions.

Our aim is to use these models for predicting the future airspace complexity for a given sector or combination of sectors in a given period. This means that it is important that any such model must be able to accurately represent operations in the airspace in order to accurately measure these metrics. This includes modelling the physical airspace with the airways, waypoints and sectors which make up the airspace along with the geography of the airspace. But most importantly, it must also be able to accurately model individual aircraft trajectories and aircraft manoeuvres. Any such system should also be flexible enough that changes to the structure of the airspace and aircraft can be made easily. This will allow the model to be applicable for simulating airspaces other than a single area of focus and will also allow changes in the configuration of the airspace, such as changes in sector boundaries, to be modelled easily. Allowing for flexibility in modelling aircraft allows for different aircraft types to be simulated, from small turbo-prop aircraft through to large jet aircraft which have different flight characteristics. The scope of this model is limited to the reduced vertical seperation minima (RVSM) airspace, ie. from altitude 29,000ft to 40,000ft, and thus only the en-route portion of flight is our primary area of focus.

3.1 Model Formulation and Construction

As our goal is to develop simulation models for use in a real time environment, we would like to develop a model that requires less time to make a prediction than usually required for high fidelity models. To achieve this goal we need to reduce the computational intensity of the model (when compared to a high fidelity model). This can be done by trading off the level of fidelity, abstraction and resolution of the model with time and possible error.

To begin the process of developing an air traffic simulation model, a minimum set of components which allows for the most basic level of air traffic simulation were identified. This basic set of components were then used to design the first of four models, Basic Simulator 1 (BS1). More complex component features and interactions were identified and gradually added to this model in an iterative fashion. This resulted in three more models, Basic Simulator 2 (BS2), Basic Simulator 3 (BS3) and Basic Simulator 4 (BS4). Of the four models, the BS1 model has the lowest fidelity and is also the most abstract, while BS3 have the highest fidelity. Figure 3.1 depicts the relative positioning of three models, BS1, BS2 and BS3; within the model hierarchy. BS1, at the top of the hierarchy, has the lowest fidelity and the highest level of abstraction, which means it should be the fastest to provide an answer or prediction. Of the three models in the hierarchy BS3 has the highest fidelity and lowest level of abstraction which means it should be the slowest to provide and answer or prediction. BS4 does not fit into this particular linear hierarchy, the reasons for this will be discussed in the following sections.

3.1.1 Assumptions

The choice of components and interactions to include in each of the three models is dependent on a set of assumptions. The assumptions taken when designing each of the three models are outlined below.

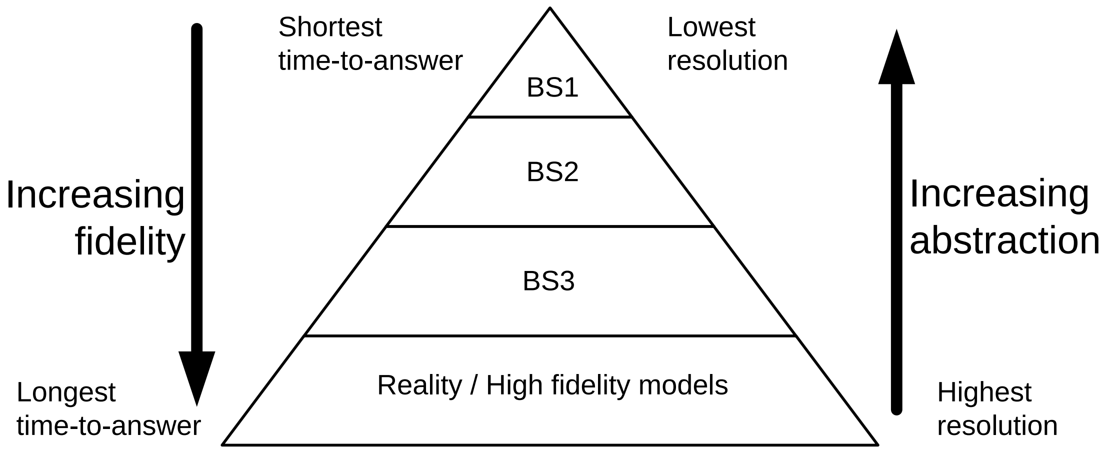


Figure 3.1: Levels of model hierarchy

Basic Simulator 1

The assumptions made when designing Basic Simulator 1 include:

- The movement of the aircraft can be modelled using the equations of motion
- The characteristics of different types of aircraft can be generalised and the aerodynamic properties of the aircraft has no effect on it's movement
- Fuel usage, emissions and aircraft mass has no effect on the movement
- The curvature and rotation of the Earth has no effect on the movement
- Wind and air pressure has no effect on the movement of the aircraft

Basic Simulator 2

The assumptions made when designing Basic Simulator 2 include:

- The movement of the aircraft can be modelled using the equations of motion
- Fuel usage, emissions and aircraft mass has no effect on the movement
- The curvature and rotation of the Earth has no effect on the movement
- Wind and air pressure has no effect on the movement of the aircraft

Basic Simulator 3

The assumptions made when designing Basic Simulator 3 include:

- The movement of the aircraft can be modelled using the equations of motion
- Fuel usage, emissions and aircraft mass has no effect on the movement
- The rotation of the Earth has no effect on the movement of the aircraft
- Wind and air pressure has no effect on the movement of the aircraft

Basic Simulator 4

The assumptions made when designing Basic Simulator 3 include:

- The movement of the aircraft can be modelled using the equations of motion
- The characteristics of different types of aircraft can be generalised and the aerodynamic properties of the aircraft has no effect on it's movement
- Fuel usage, emissions and aircraft mass has no effect on the movement
- The rotation of the Earth has no effect on the movement of the aircraft
- Wind and air pressure has no effect on the movement of the aircraft

It is assumed that the Earth is a static body with no fluctuation in wind, weather or air pressure in the atmosphere. It is also assumed in all models that the aircraft will operate according to their designated route and that they will not be required to make autonomous decisions to optimise factors such as fuel usage and flight time or avoid collisions during the simulation. These decisions, however, may be made by external systems, and so each model includes a method whereby deviations to the route can be communicated before the start of the simulation.

3.1.2 Architecture

A multi-agent simulation platform was developed to represent the aircraft in the simulation environment. This platform formed the basis for all four of the developed models. An overview of the operation of the platform can be seen in Figure 3.2. The multi-agent approach allows for the mapping of an environment to individual agents capable of autonomous actions in the environment to meet design objectives (Weiß, 1999). Agent-based simulation has become a common method for modelling and studying complex traffic and transport systems (Chen and Cheng, 2010). The complex interactions that take place between aircraft, airports, ATCs and other components of the air traffic system makes agent based simulation an ideal choice for air traffic simulation (Agogino and Tumer, 2012). This approach allows for agents to be individually assigned goals, characteristics and actions, which means each aircraft (the agents) can be assigned their own route and also allows for different types of aircraft to be simulated. The differences between our models arise when considering the composition of the agent and the interaction between the environment and the agent.

The developed platform takes the airspace configuration data, flight plans for each aircraft and the aircraft positioning data (if available and only for starting the simulation with aircraft with only a portion of their flight plans completed) as input. During the initialisation phase the airspace configuration data, flight plans and aircraft positioning data are used to initialise the simulation environment and the aircraft agents. A fixed increment time advance is used for a simulation time clock to trigger events in the environment. At every time advance the platform will trigger the movement phase and the evaluation phase. During the movement phase, the platform will iterate through each of the aircraft agents which have flight plans available and calculates the aircraft's trajectory for that time step. The manner in which the trajectories are calculated are dependent on the model being used. The aircraft is considered to have moved at the end of the time advance. The time advance increment is fixed throughout the simulation, but the simulation clock can be changed to operate at a faster rate than real-time to provide fast-time simulation.

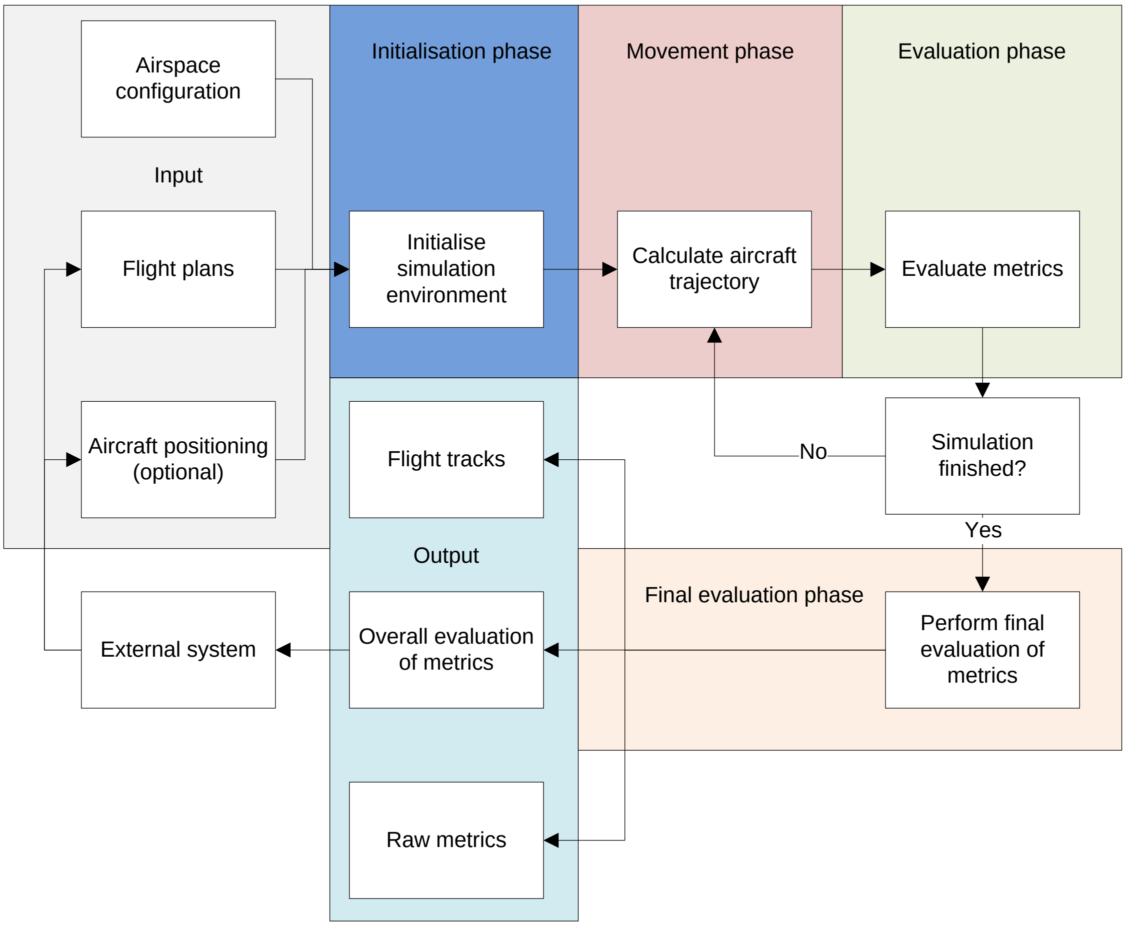


Figure 3.2: Multi-agent simulation platform design

After the movement phase, the evaluation phase is triggered. During this phase a range of different air traffic metrics are calculated. The metrics that are calculated and the frequency at which they are calculated are set depending on the aims of the study. Once the simulation is complete a final round of evaluation of metrics is conducted during which aggregated statistics are calculated, such as average airspace complexity for a given time period. The data for these metrics are then returned to an external system or recorded for later analysis.

This platform was written using the C# programming language. There is no GUI associated with this platform for visualisation of the air traffic or viewing data as the platform is designed to run independently of any other systems in order to reduce the time and computational resource requirements for running a single simulation. The platform can however be exported and run as an external library in other systems, such as an optimisation system.

In the following sections we will discuss the decisions which were taken to influence the level of fidelity, abstraction and resolution in each of the simulations models. We will also discuss some of the decisions which are common for each of the three models as a result of the common simulation platform.

3.1.3 Airspace, Weather and Geographic Modelling

The simulation platform and the four models are capable of simulating any airspace around the world. This is achieved by loading a set of airspace configuration files relating to the airspace before the start of the simulation. These airspace configuration files include details about the sectors in the airspace, the waypoints and airports located in the region; and any parameters necessary for the representation of the airspace, such as the parameters required for the projection of geographic coordinates between different systems.

Sectors, Waypoints & Airports

Sectors are represented by a series of points, minimum altitude, maximum altitude and a classification. The sector is a volume bounded by the points, the minimum altitude and the maximum altitude. The classification indicates whether the sector is a high altitude sector or a low altitude sector. The details of the sectors are drawn from resources provided by air traffic service providers from the appropriate region. The waypoints and airports are input with name and point pairs. The elevation of the airport is also included. In an effort to reduce computation time, the details of the waypoints and airports are not retained beyond the initialisation stage of the simulation. The relevant information is merged with the flight plan data for each aircraft agent. Instead of retaining a long list of waypoints and iterating through this list every time we are required to obtain the location of the waypoint, we can simply iterate through the small list of waypoints which are relevant to the flight plan of the aircraft. For example, the Australian airspace consists of more than

5000 waypoints, while only a handful (5 to 20) are relevant for a particular flight and iterating through a list containing 20 items is quicker than a list containing 5000 items, especially when we are required to repeat this process multiple times during every time advance. As multiple aircraft may have flight plans which include the same waypoints, this however means we require slightly more memory as we are storing the same information within multiple agents.

Flight Plan

As our aim is to simulate air traffic, we require a flight plan in order to simulate each aircraft agent. A flight plan provides us with the route the aircraft will take along with other vital information regarding the operation of the aircraft, such as the type of aircraft, number of passengers, safety equipment, communications equipment, aircraft design and crew details. Much of this information is not important for the scope of air traffic simulation, so the information that is required for the simulation of individual flights are extracted and a shorter flight plan is created containing the relevant information. The shorter flight plan contains the flight's origin, destination, route, cruise altitude and speed, aircraft model number and activation time. An example of two flight plans can be seen in Listing 3.1 and 3.2. The flight plan shown in Listing 3.1 is for a flight departing and arriving at an airport within the simulated airspace, while the flight plan shown in Listing 3.2 is for a flight which departed from an airport outside the simulated airspace and arriving at an airport within the airspace. A breakdown of the information contained in the example flight plans can be seen in Table 3.1.

Listing 3.1: Flight plan for aircraft starting simulation from an airport

```
JST443;YMML;YBCG;ROKDL SALLY NONIX KACEY MDG TW BERNI
ROONY GREAV KERRI;YMML;3;A320;350;390;0;030011;VHVWT
```

Listing 3.2: Flight plan for aircraft starting simulation in-flight

```
QFA21;WSSS;YPPH;LAMOB IDOKU CAR MRW PH;LAMOB;
1;A333;330;410;0;030059;VHQPB
```

Table 3.1: Breakdown of data contained within the example flight plans

Desription	**Listing 3.1**	**Listing 3.2**
Callsign	JST443	QFA21
Origin	YMML	WSSS
Destination	YBCG	YPPH
Route	Via waypoints ROKDL, SALLY, NONIX, KACEY, MDG, TW, BERNI, ROONY, GREAV and KERRI	Via waypoints LAMOB, IDOKU, CAR, MRW and PH
Activation point	YMML	LAMOB
Flight type	3	1
Aircraft model	A320	A333
Cruise speed	350 kts	330 kts
Cruise altitude	390 (FL390 or 39,000ft)	410 (FL410 or 41,000ft)
Activation time	030011 (Day 03 00:11)	030059 (Day 03 00:59)
Aircraft registration	VHVWT	VHQPB

The biggest difference we can see between the two example flights plans in Table 3.1 are the activation points for the simulation of the two flights. Flight JST443 begins simulation from the origin airport YMML, while flight QFA21 starts in-flight (en-route) from the waypoint LAMOB as it's origin is outside of the simulated airspace. This difference is also noted by the flight type identifier. A summary of the flight types can be seen in Table 3.2. Flight types 2 and 3 begin simulation from origin airport at it's designated elevation, while flight types 1 and 4 begin simulation from their activation point (waypoint) at their cruise altitude. Flight types 2 and 4 are simulated until their final waypoint within the airspace is reached.

The flights are activated when the simulation clock reaches the flight's activation time as found in the flight plan. The activation time includes a day, hour and minute component. Day 0 00:00 is the earliest possible activation time. All activation times are converted to the number of seconds after this time as the simulation clock operates by the number of seconds elapsed. The simulation clock can be started at any time on or after 0 seconds. If starting beyond 0 seconds, then the activation point for flights whose activation time has already passed may be updated to an in-flight point and activated after the initialisation phase. Once a flight has been activated it will begin simulation and it's movements are calculated in the following

Table 3.2: Portion of flight simulated based on the location of origin and destination

Flight Type Identifier	**Inside or outside simulated airspace**		**Portion of flight simulated**	
	Origin	**Destination**	**Activation point**	**Deactivation point**
1	Outside	Inside	First waypoint in airspace	Destination
2	Inside	Outside	Origin	Last waypoint in airspace
3	Inside	Inside	Origin	Destination
4	Outside	Outside	First waypoint in airspace	Last waypoint in airspace

time advances. Flights of type 1 and 4 are activated at their cruise altitude and speed as found in the flight plan. Fights of type 2 and 3 are activated with a speed of 0 kts and altitude equal to the elevation of the origin airport. These flights then start to climb to the cruise altitude and accelerate to the cruise speed as found in the flight plan.

During the simulation initialisation phase the list of waypoints from the flight plan are converted from a single string to a list of waypoint objects containing the waypoint's name and it's location. This list is then stored as a property of the aircraft agent. The same process is followed for the origin and destination points.

Wind and Air Pressure

It is assumed that the aircraft agent self corrects any effect that may be caused by wind. This assumption is advantageous in two ways. First, it reduces computation time as we do not need to consider additional components when calculating aircraft movement. Secondly, we are not required to dynamically generate any wind conditions, also reducing computation time and time for simulation initialisation. It is also assumed that the air pressure remains constant throughout the airspace. This assumption also leads to fewer components being required to calculate aircraft movement and performance.

Geographic Projection

Two different coordinate systems were used in the four air traffic simulation models for representing the airspace: the Lambert conformal conic (LCC) projection (Deetz *et al.*, 1918) and the geodetic datum (latitude and longitude). The LCC projection was selected for representing the Earth as it is more convenient to use a flat rectangular coordinate system for measuring linear distances than the geodetic coordinate system (Calvert, 2002), as we are able to use equation common to the Cartesian coordinate system. This also reduces the number of steps and the complexity required to measure distances when compared to the geodetic coordinate system. The LCC projection is made by placing a cone over the Earth such that it intersects the Earth's surface at two selected parallels, as can be seen in Figure 3.3. The cone is then unrolled and scaled with reference to the two parallels to form a flat rectangular map. The parallels used for the projection vary depending on the area of interest. Recommended parallels for each region can be obtained from national GIS organisations' data sheets, such as from Geoscience Australia and Land Information New Zealand. When using the LCC in our simulation platform we must include the correct parallel parameters for the initialisation phase. During the initialisation phase all geodetic coordinates are converted to LCC projected coordinates based on these parameters. This includes the location of waypoints, airports and sector boundaries. In this case, all lateral movements are conducted within the LCC coordinate system, but vertical movement is still calculated in feet.

The other coordinate system which is used by the air traffic simulation models is the geodetic coordinate system. This system provides a standard method for representing any location on the Earth by assigning the location a latitude and longitude. When using this coordinate system, any movement calculations take into consideration the curvature of the Earth. This system provides us with a higher fidelity representation of the Earth and the movement of the aircraft than the LCC system, but some extra steps are necessary when calculating aircraft movement. The LCC projected coordinate system is used in BS1 and BS2 while the geodetic coordinate system is used in BS3 and BS4.

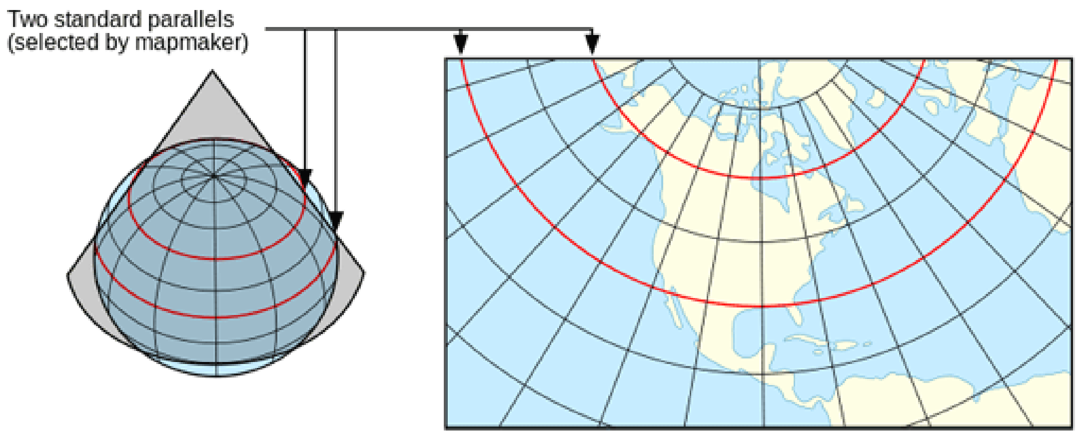


Figure 3.3: Lambert conformal conic (LCC) projection (Quist, 2011)

3.1.4 Aircraft Modelling

In all four of the air traffic simulation models, the aircraft are assumed to be point masses with no inputs or outputs. This allows us to forego calculations relating to drag, fuel flow, emissions and engine operation. In BS1 and BS4 the aerodynamic properties and aircraft performance are not considered, however in BS2 and BS3 we use use data obtained from look-up tables, such as climb rate and acceleration, which are influenced by these characteristics.

Aircraft Trajectory

The overall trajectory taken by an aircraft during simulation is determined by the origin (or activation point), route and destination (or deactivation point) as found in the flight plan. The aircraft will fly directly from point to point as per the order provided in the flight plan. In the real world and many high fidelity air traffic simulation models, aircraft take some considerable time to turn when a major change in heading is desired, such as when a waypoint has been reached. This turning begins some time before the point is reached and the aircraft's trajectory forms an arc joining the direct paths from the previous and following points with the upcoming point. In our models, the aircraft continue to fly until it has reached the point and will immediately change its heading to the following point on it's route. An example of this turning manoeuvre can be seen in Figure 3.4 for an aircraft with a flight plan with waypoints ABC. In our simulation models the aircraft will directly travel from A to B and then to C. In the real world aircraft will typically take the route AB'C'C due to the time required to turn. By not including this turning manoeuvre we are able to eliminate the need to calculate the bank and turning angles. A point is reached when the distance between the aircraft and point (d) is smaller than or equal to the distance the aircraft can move at it's current speed in the time step ($v/\Delta t$, where v is the current speed of the aircraft and Δt is the length of the time advance).

$$\text{Point reached if } d \leq \frac{v}{\Delta t} \tag{3.1}$$

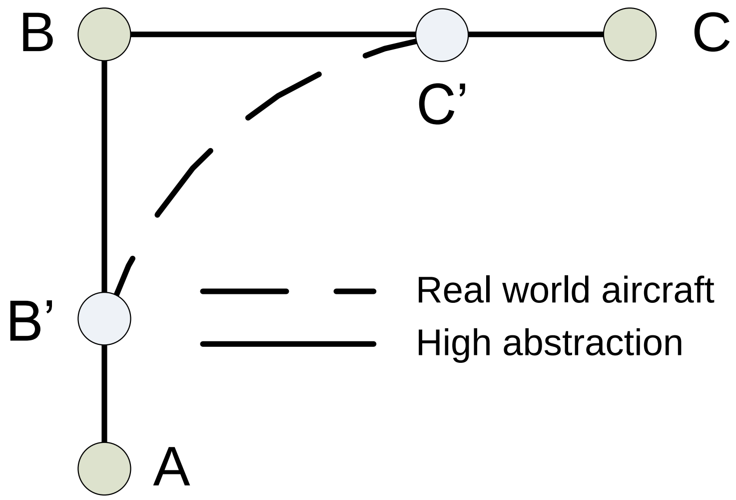


Figure 3.4: Aircraft turning manoeuvre overview

As our focus is primarily on the en-route phase of the aircraft's flight, all movement and procedures on the ground at the airport are ignored. There is also no departure or arrival queue at the airports. Aircraft which have conflicting departure times at airports (ie. similar activation times for type 2 and 3 flights at the same airport) are activated at their scheduled time irrespective of the traffic at the airport. This means that there is no sequencing or assurance that flights have sufficient spacing for take-off and landing. It is assumed that the flight plans are produced in such a way that these assurances have already been conducted. Flights of type 1 and 3 simply start and/or finish their flight at a point on the ground as determined by the location of the airport.

During the course of a flight, the flight can be classified as being in one of three phases: climb, cruise and descent. The take-off and climb out phases are treated as part of the climb phase while the approach and landing phases are treated as an extension of the descent phase. Taxi in, taxi out, runway hold, runway out and runway in phases are not considered as we do not simulate movement on the ground.

An overview of the calculation of the trajectory during each time advance can be seen in the flowchart in Figure 3.10. This flowchart forms the basis of the movement phase in Figure 3.2. During each time step, the distance remaining between the aircraft and the next (target) point on it's route is recalculated. The distance to this point is calculated using Equation 3.2 in the BS1 and BS2 models. In this equation x_T and y_T are the current x and y coordinates of the aircraft, while x_g and y_g are the coordinates of the target point. Equation 3.3 is used in BS3 and BS4 instead as it uses the geodetic coordinate system and the great circle distance between the two points must be calculated to take into account the curvature of the Earth. The data required to calculate the distances using Equation 3.3 is the aircraft's current latitude and longitude, λ_T and τ_T respectively, and the latitude and longitude of the target point, λ_g and τ_g respectively. From these equations we can easily see that calculating distances in BS3 and BS4 is more computationally complex than in BS1 and BS2.

$$d = \sqrt{(x_T - x_g)^2 + (y_T - y_g)^2} \tag{3.2}$$

$$d = R \times c \tag{3.3}$$

$$\text{where } c = 2 \times atan2(\sqrt{b}, \sqrt{(1-b)})$$
$$b = sin^2(\frac{\lambda_T - \lambda_g}{2}) + cos(\phi_T)cos(\phi_g)sin^2(\frac{\phi_T - \phi_g}{2})$$

The heading to the target point is calculated using Equation 3.4 in BS1 and BS2, while Equation 3.5 is used in BS3 and BS4. The heading to the target point needs to be recalculated during each time step when using the BS3 and BS4 model as we advance our aircraft in each time step in a linear motion despite working in an environment where the world is round. This makes it is necessary to frequently re-evaluate the heading to account for the curvature of the Earth. Doing so in BS1 and BS2 is not necessary as the heading along a direct line between two points remains constant in the Cartesian coordinate system. From these equations we can see that calculating the heading in BS3 and BS4 is more computationally complex than in BS1 and BS2, as was the case for calculating the distance between two points. Once a target point has been reached, the following point in the aircraft's flight plan is set as the target point and the heading and distance remaining are recalculated. If the aircraft's deactivation point has been reached then the aircraft agent is deactivated and removed from the simulation.

$$h = arctan\left(\frac{x_T - x_t}{y_T - y_g}\right) \tag{3.4}$$
$$h = arctan\left(\frac{sin(\phi_g - \phi_T)cos(\lambda_g)}{sin(\lambda_g)cos(\lambda_T) - sin(\lambda_T)cos(\lambda g)cos(\phi_g - \phi_T)}\right) \tag{3.5}$$

Once we have determined the target point for the aircraft, we can start the process of determining the aircraft's trajectory for this time step. This starts by determining the speed of the aircraft at the end of the time step, then determining the distance the aircraft will travel in the time step based on the speed and then using this distance along with the heading to the target point to estimate the location of the aircraft at the end of the time step. The speed of the aircraft at the end of the time step is dependent on the phase of the flight and the cruise speed found in the flight plan. If the aircraft is in the climb or cruise phases the aircraft will aim to accelerate to or maintain the cruise speed from the flight plan. If the aircraft is in the descent phase then the aircraft will decelerate towards 0 kts. The acceleration and decelerations rates ($ROCD$) will be discussed in the Aircraft Performance section below. Maximum and minimum speed ranges for the aircraft are maintained so that aircraft do not decelerate to 0 kts mid flight or exceed speeds which are technologically possible. It is assumed that the aircraft will undergo constant linear acceleration throughout the time step. Once the speed of the aircraft at the end of the time step has been determined we can calculate the distance the aircraft will travel in this time step using Equation 3.7. In this equation v_T is the current speed of the aircraft and $v_{T+\Delta t}$ is the speed at the end of the time step. v_T, $v_{T+\Delta t}$ and $ROAD$ are scaled to match the time advance increment before using Equations 3.6 and 3.7. For example, if the time advance increment is one second and the speed is measured in knots, it is converted from nautical miles per hour to nautical miles per second before using these equations.

$$v_{T+\Delta t} = v_T + ROAD \tag{3.6}$$

$$d_{\Delta t} = \frac{v_{T+\Delta t} + v_T}{2} \tag{3.7}$$

Now that we have determined the distance the aircraft will travel in this time step and the direction it will travel, we can determine the coordinates of the aircraft's position at the end of the time step. The coordinates of the aircraft's position at the end of the time step when using the BS1 and BS2 models are calculated using Equations 3.8 and 3.9 where x_T and y_T are the current coordinates of the aircraft and $x_{T+\Delta t}$ and $y_{T+\Delta t}$ are the coordinates of the aircraft at the end of the time step. When using BS3 and BS4 we use Equations 3.10 and 3.11 to determine the coordinates of the aircraft's position. In these equations ϕ_T and λ_T are the current coordinates of the aircraft and $\phi_{T+\Delta t}$ and $\lambda_{T+\Delta t}$ are the coordinates of the aircraft at the end of the time step. Equations 3.10 and 3.11 also take into account the effect of the curvature of the Earth by considering the distance of the aircraft from the centre of the Earth using the average radius of the Earth (R) and the current altitude of the aircraft, z_T.

$$x_{T+\Delta t} = x_T + d_{\Delta t} cos(h) \tag{3.8}$$

$$y_{T+\Delta t} = y_T + d_{\Delta t} sin(h) \tag{3.9}$$

$$\phi_{T+\Delta t} = \phi_T + \frac{d_{\Delta t} cos(h)}{R + z_T} \tag{3.10}$$

$$\lambda_{T+\Delta t} = \lambda_T + \frac{d_{\Delta t} sin(h)}{(R + z_T) cos(\lambda_T)} \tag{3.11}$$

$$z_{T+\Delta t} = z_T + ROCD \tag{3.12}$$

The altitude of the aircraft at the end of the time step is determined using Equation 3.12 by simply adding the rate of climb or descent (ROCD) to the current altitude. This equation is common to all models. During the climb and cruise phase the aircraft will aim to reach the cruise altitude from the flight plan and during the descent phase the aircraft will aim to reach the altitude of it's deactivation point. The method for determining the ROCD will be discussed in the aircraft performance section below.

Aircraft Performance

One of the key difference between the developed models is how the aircraft's acceleration/deceleration rates (ROAD) and climb/descent rates (ROCD) are obtained. For BS1 and BS4, the ROCD and ROAD were set as pre-determined constant rates irrespective of the aircraft's model, phase of flight or altitude. These rates were set to be representative of the most commonly used commercial aircraft.

In BS2 and BS3 the aircraft performance data from the Base of Aircraft Data (BADA) (Eurocontrol, 2004) was used to determine an aircraft's ROCD and ROAD. BADA is an aircraft performance model developed and maintained by Eurocontrol for use in research and development applications for trajectory simulation and contains operational performance parameters and performance summary tables for over 300 aircraft types. The BADA documentation, Eurocontrol (2004), also provides us with algorithms for high fidelity trajectory simulation through the calculations of engine thrust, aerodynamic drag, fuel consumption and dynamic mass of the aircraft while using the operational parameters included in the dataset. This high level of fidelity modelling is outside the scope of this study, so instead we use the performance summary tables which are also included in the dataset. The BADA dataset provides us with a summary table for each aircraft type which specifies cruise, climb and descent performance at different flight levels. The summary tables are produced using the operational parameters and algorithms mentioned previously for an aircraft of nominal mass. The data available includes the nominal speed and fuel flow during each of the three phases of flight and also the ROCD during the climb and decent phases for a range of flight levels. Using these tables to obtain the ROCD and speed of the aircraft allows us to forego the use of the complex set of algorithms required for their calculation and simply looking up this data from a table, thus reducing computation time. Although these tables are limited to aircraft of fixed mass, this method provides us with a higher level of fidelity for modelling individual aircraft when compared to BS1 and BS4 as we are able to model a larger range of aircraft and the effects of aerodynamic properties and engine performance of the different aircraft types at different stages of their flight.

The flight plan includes the type of aircraft that will be operating on the flight. In the example flight plans, in Listing 3.1 and 3.2, we have an A320 (Airbus A320-200) and an A333 (Airbus A330-300). The aircraft type is necessary if we wish to model any aerodynamic properties or aircraft performance characteristics using the BADA tables. Every time we want to calculate an aircraft's speed or ROCD when using the BS2 and BS3 models, it is a matter of selecting the table for that aircraft type and obtaining the data for it's flight level. As the data is only available for a set of flight levels (usually in 2000 feet increments) we are required to interpolate the data when the flight level of the aircraft does not match the flight levels available in the table. We decided whether the aircraft should accelerate or decelerate based on the nominal speed for the flight level and flight phase pair. If the nominal speed is higher than the aircraft's current speed, but lower than the cruise speed in the flight plan, we accelerate. If the nominal speed is lower than the aircraft's current speed, but higher than the stall speed, we decelerate. The ROAD and stall speed are fixed for all aircraft types. In comparison, BS1 and BS4 will continue to accelerate or deccelerate irrespective of the nomial speed for each flight level.

Summary

A summary of the key air traffic simulation functions and their implementation in the three simulation models can be seen in Table 3.3. The combination of these implementations determines the level of fidelity, abstraction and resolution of the three models.

From Table 3.3 we can see that BS1 incorporates the most basic level of simulation components and interactions, and therefore has the lowest level of fidelity. All aircraft have same performance irrespective of their type. This means that the climb profile and speeds for all aircraft are the same irrespective of their type. The physical environment in BS1 is represented by a Lambert conformal conic projected coordinate system which provides the approximate position of objects in the 3D space. The effect of wind, weather or atmospheric conditions do not influence the movement of aircraft in BS1.

The major difference between BS1 and BS2, an extension of BS1, is the use of data from BADA tables to determine ROCD and target speeds. This allows for the modelling of different aircraft types and more accurately represents the different climb and speed profiles of different aircraft types. The BADA tables also incorporate an approximation of the effect of atmospheric conditions (eg. air pressure) on the aircraft performance. This is useful as the nominal speed and ROCD at different flight levels may not be constant.

There is only one key difference between BS2 and BS3, itself an extension of BS2. The difference is the use of the geodetic datum instead of the Lambert conformal conic projected coordinate system. This allows for more accurate estimation of aircraft positioning when compared to the real world. This however results in more complex equations for calculating motion, distances and headings.

BS4 is also an extension of BS1, but is branched in a different direction to BS2. The difference between BS1 and BS4 is that the geodetic datum instead of the Lambert conformal conic projected coordinate system in BS4. All other aspects of BS1 are carried over to BS4.

In all models the fuel usage, emissions and aircraft mass assumed to have little implication on the movement of the aircraft. The effect of the use of geodetic datum vs the LCC projected coordinate system, different target speeds and ROCD on the level of error in each of the three models will be discussed in the validation section below.

3.2 Air Traffic Simulation Validation

Validation of abstract process models and methods is an important step to ensuring that we can trust our models and that they are an adequately accurate representation of the real-world system (Fishwick, 1988; Frantz, 1995). It is often too costly and time consuming to confirm that these models are valid for the entirety of the domain they represent, so instead, evaluations are conducted until there is enough confidence that the model can be considered valid for its intended application (Sargent, 2005). The validation process can be broken into two stages - conceptual and operational (Rao *et al.*, 1998). Conceptual validation involves evaluating the model's theory and comparing and contrasting the model's logic with alternate methods, while the operational validation involves the measurement of the consistency between the model and the real world system. For the air traffic domain, the real world data for the evaluations could include flight tracks of real aircraft and notifications of losses of separation between aircraft. Unfortunately this data is often difficult to obtain and when it is obtained, the data usually also incorporates actions taken by the ATCs and pilots to avoided situations such as loss of separation. In order to adequately evaluate our model we require data free of the actions taken by ATCs and pilots. An alternative to using real world data is to use synthetic obtained data from a high fidelity models and treat this model as the real world (Glover and Lygeros, 2004). For this reason we used the Air Traffic Operations and Management Simulator (ATOMS) (Alam *et al.*, 2008) as the pseudo real world and evaluate our models, BS1, BS2, BS3 and BS4; against the data obtained from this model. The high fidelity model used in ATOMS allows us to capture every factor and uncertainty involved in real world flight operations, but within a synthetic real world. From here on we will refer to ATOMS and its high fidelity model as the 'perfect model' or the real world.

We found that conceptually both the perfect model and our low fidelity models were in agreement regarding the core aspects of the theory. The major difference arose when determining the ROCD and acceleration. In the perfect model these parameters are determined using the thrust, drag and mass algorithms found in the BADA documentation. In the operational validation section below we validate the flight tracks from our models with those expected from the real world.

3.2.1 Operational validation

The operational validation is undertaken to present a quantitative measure to determine whether the output of the simulation model closely resembles those expected from the real world or the real system for the same set of inputs (Sargent, 2005). The output from the simulation model and the real world data do not necessarily have to have a 100% correlation for a model to be valid. A model is considered valid if there is a favourable correlation with the simulation results and the real world observations (Rao *et al.*, 1998). However the desired level of accuracy of the model is dependent on the requirements of the application.

Phillips and Marsh (2000) provides several methods for validating an air traffic simulation model:

- Compare the vertical profile of individual aircraft
- Replay aircraft's route in plan view
- Graph results such as number of aircraft in sector or number of conflicts

In order to use these methods to evaluate the operational validity of our models we require a set of common flights for which we can compare our models and the real world operations. As we wish to evaluate the validity of the models for simulating air traffic, we can use a set of flight plans representing real world flight operations as the input to the models. The flight plans that were used for the validation of our models were artificial flight plans generated using historical data for real flight operations in the Australian airspace. A Gaussian distribution is used to determine the number of aircraft departing each airport for each hour of the day based on the average number of departures for that hour from the historical data for one year. A Poisson distribution, again based on the historical data, is then used to determine the exact departure time for the flights. The route, cruise altitude and cruise speed assigned to the aircraft for each airport pair are also determined by the historical data. Further details about the generation of the artificial flight plans can be seen in Tang (2012). This system was used to generate flight plans for operations within the Australian airspace for a period of 30 days. The flight plans were then used as input in the models and simulated. An overview of the 30 day scenario can be seen in Figure 3.11. We can see that the traffic level is varied on each day of the 30 day scenario and the distribution of aircraft types operating these flight is also varied throughout the scenario. We can also see that majority of flight remain active within the simulated airspace for less than three hours.

Validation of air traffic simulation models by visual inspection is straightforward. This involves plotting a flight's track and comparing it with the flight plan. It was found that in all cases that were tested, the tracks produced by the simulation model were consistent with the flight plan, ie. the aircraft visited each waypoint in the correct order and adhered to altitude requirements. In Figure 3.8 we can see a comparison of the altitude, speed and heading profile for a 15 minute snippet of a flight climbing towards it's cruise altitude, simulated using BS3 and BS4 and the profile from the real world operation (ie. obtained from ATOMS). An overview of the flight plan for the aircraft can be seen in Figure 3.5 for this snippet. The snippet begins with the aircraft at an altitude of 9,500ft climbing towards a waypoint at an altitude of 24,000ft and then continuing to climb towards it's cruise altitude of 30,000ft. One of the key differences that we can see from Figure 3.8 is that the simulators use different climb rates to reach the cruise altitude. BS4 uses a fixed rate, while BS3 uses an ROCD obtained from the BADA lookup tables at each time step for the current flight level. In the real world, the rate is similar to those obtained from the BADA lookup tables, but the ROCD for the aircraft is set in such a way that the target altitude is reached when the target point is reached in the lateral axis (if technologically feasible). This is unlike our models where the target altitude is reached as soon as possible. The speed profile for BS3 follows the trend of profile from the real world during the climb phase, but there is a big disparity when the aircraft reaches its cruise altitude. We can also see from this figure that there is a small difference between the cruise speed of the aircraft when simulated with BS4 and the real world (the aircraft did not reach it's cruise speed when simulated with BS3 in the 15 minute snippet). This is due to the aircraft only accelerating/decelerating until it is within 2 kts of the requested cruise speed in the real world while our models aim to accelerate/decelerate until the difference from the target speed is less than the acceleration rate (ie. $v_g - ROAD \leq v \geq v_g + ROAD$). This may create a disparity of upto 2.5 kts in speed between our models and the real world. We can also see that the distance of the aircraft at each corresponding time point between in the real world and BS3 and BS4 remains small as the relative speeds remain similar, but quickly increases as BS4 accelerates much faster than what is expected in the real world and the cruise speed is reached faster in the real world than BS3. A comparison of the vertical, speed and heading profile for an aircraft in the en route (cruise) phase can be seen in Figure 3.9, also for a 15 minute snippet, when using BS3, BS4 and comparing to the real world. The flight plan for this flight for this snippet can be seen in Figure 3.6. We can see that the starting altitude and the altitude targets for waypoints for this flight are the same. From Figure 3.9 we can see that the deviation distance at each time step between the real world and the our two simulators is less than 0.2 NM for this period. The deviations are primarily caused by the way the simulators handle heading changes. If we take a look at the heading profile we can see that in the real world there is a gradual change in heading when approaching a waypoint while the other simulators use an instantaneous heading change when the waypoint is reached. We can see that major changes in the distance between the simulators and the real world occur when there is a heading change. The plot of the vertical, speed and heading profile for BS1 and BS2 are the same as that for BS4 and BS3 respectively as the assumptions underlying those components are common among those pairs. However the deviation in the position of the aircraft for each time step when compared to the real world is quite different. In Figure 3.7 we can see a plot of the distance between the aircraft at each time step for the snippet from the real world and the corresponding time step in BS1 and BS2. We can see that the heading changes are no longer the main contributing factor for track deviations in these simulators. Despite the aircraft travelling at the same speed we see that the deviation from the real world continues to grow in a linear fashion. This is due to the LCC projection system being used by these simulators while in the real world the aircraft operate in an environment where the Earth is curved. Although the aircraft cover similar distances along the ground in each time step (ie. have similar ground speeds) in their respective environments, the distance covered in each time step when converted to the other environment was vastly different.

In order to get a better idea of the deviation in position between the real world and our simulators we will use a 30 day scenario of air traffic in the Australia airspace. First we simulated the 30 days of air traffic with the perfect model and recorded the tracks of each flight at an interval of one minute. As we are primarily interested in the simulation of air traffic in the cruise phase in the RVSM airspace, a list of aircraft whose cruise altitude from the flight plan was greater than or equal to 29,000ft were selected. This resulted in around 90,000 flights being selected. We then found the point at which each of these flights transitioned from the climb phase to the cruise phase. After finding the this transition point, the flight's position two minutes prior to reaching this point was found and the corresponding time became the activation time and point for these flights in our simulation models. Similarly

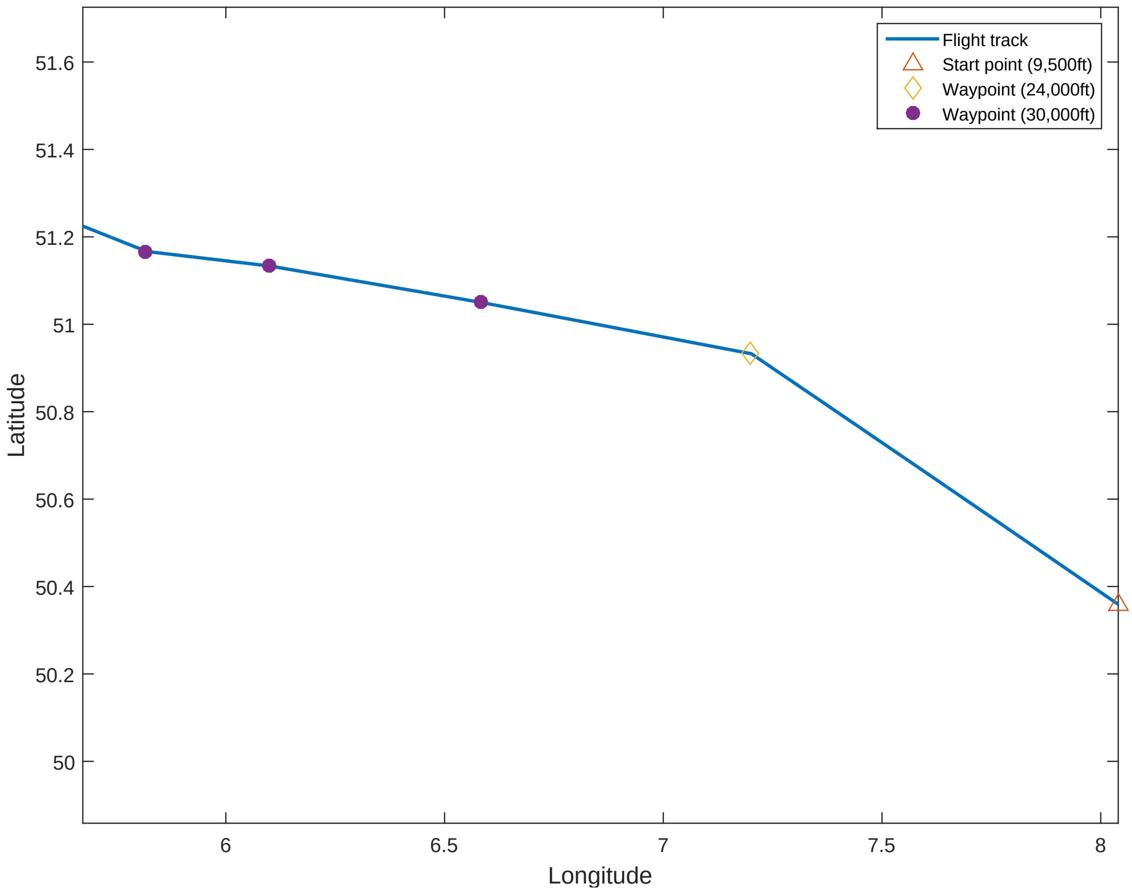


Figure 3.5: Flight plan for 15 minute snippet for aircraft in the climb phase

the position of the aircraft two minutes after the commencement of the descent phase was found and set as the deactivation point for the flight. We then used the simulator BS1 to BS4 to simulate these aircraft and also recorded their tracks at an interval of one minute.

Once all of the selected flights in the 30 day scenario were simulated with each of our four models, the tracks for each flight were matched to the corresponding point in the flight (ie. time since the activation time in our simulation) and the distance between the aircraft at each one minute interval from the track recorded from the real world tracks were calculated. The plots in Figures 3.12 to 3.16 show the track deviations among all flight for one minute interval for the first five hours of flight. From Figure 3.12 and 3.13 we see that the longer we simulate the flight, the larger the distance between the aircraft in the two simulators becomes and after five hours

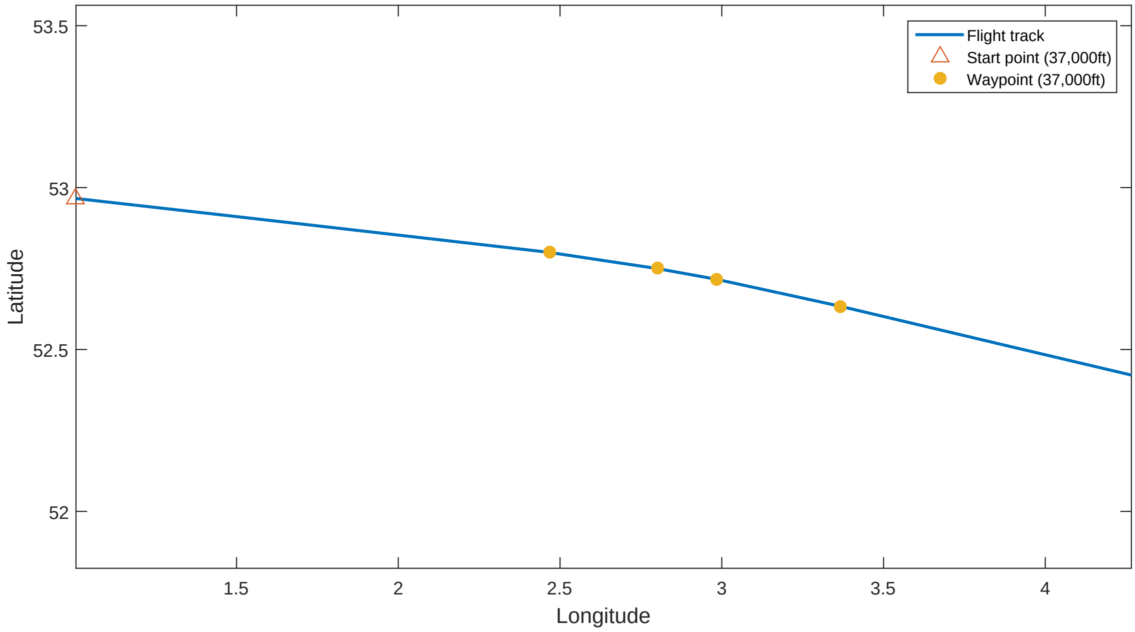


Figure 3.6: Flight plan for 15 minute snippet for aircraft in the en route phase

of simulation the average distance between the aircraft in the real world and BS2 becomes as large as 70 NM. The main contributing factor to the continued growth in the distance between the position of the aircraft among the two simulators is due to the conversion between the projection systems. Conformal projections systems such as LCC preserve the shapes of small areas on the ground, but the scale of the maps produced with this projection vary from point to point (Calvert, 2002). This means a movement of 1 NM in one part of the map may not be the equivalent of moving 1 NM in another. From Figure 3.14 and 3.15 we can see that after an initial increase in the average deviation in the first hour the average deviation stays relatively constant for the following few hours. The initial increase is due to final stages of the climb phase which is part of the flight plan for these simulators. A comparison of the deviation caused by the four simulators can be seen in Figure 3.16. From this figure we can see that the two simulators working in the geodetic environment, BS3 and BS4, have significantly lower deviations than those using the LCC projection. The differences in the deviations between BS3 and BS4 primarily arise from the ROCD used in the final stages of the climb. We saw in Figure 3.8 that the final stages of the climb in BS3 is somewhat longer than what is expected in the real world while

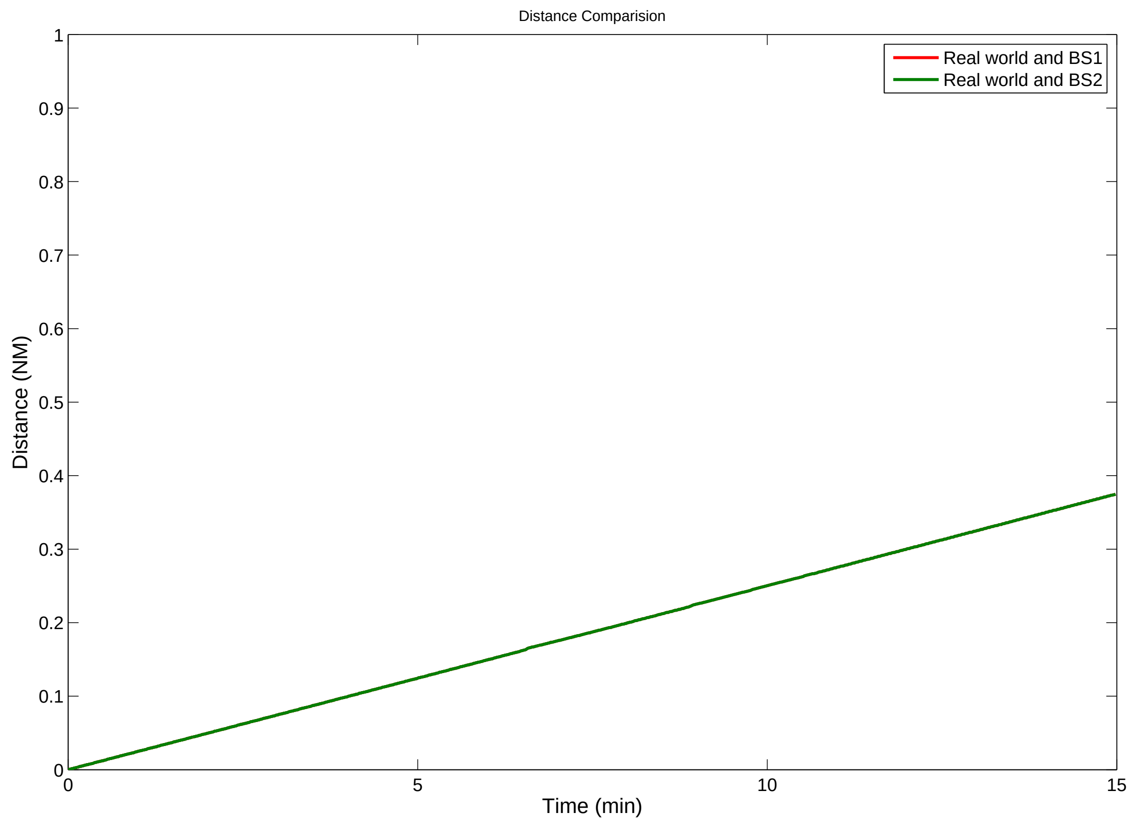


Figure 3.7: Comparison of flight track deviation for BS1 and BS2 from the real world

a short constant climb rate used in BS4 does not have as big of an effect.

As our aim is to use these simulators to predict the future airspace complexity, we must be able to predict position of an aircraft to within 5 NM as the group with the smallest proximity distance in one of the common complexity measurement methods falls in the the range 0-5 NM (see Chapter 4 for more details). From Figure 3.16 we see that each of our four models, based on the average track deviations, will only be acceptable for predicting airspace complexity using this method for different time frames. BS1 and BS2 would only be acceptable in the short term (less than 15 minutes), BS3 for the short to mid term (less than 60 minutes) while BS4 would be acceptable for the long term (long than 60 minutes).

Table 3.3: Summary of differences in simulation models

Simulation fuction	**Simulation model**			
	Basic Simulator 1	**Basic Simulator 2**	**Basic Simulator 3**	**Basic Simulator 4**
Level of fidelity	Lowest		Highest	
Simulation limit	Limited to airspace of interest (eg. entire Australian FIR or selected set of sectors)			
	No ground movements			
Coordinate system	Lambert conformal conic projection		Geodetic datum	
Wind & weather	Not considered			
Atmospheric conditions	Not considered	Approximated by BADA tables		Not considered
Aircraft	Point mass			
	All aircraft types are treated as the same	Different aircraft types are treated differently		All aircraft types are treated as the same
Flight phahses	Limited to climb, cruise and descent			
Rate of acceleration/ deceleration (ROAD)	Fixed and linear			
Target speed - climb and cruise phase	Cruise speed	Nominal speed for flight level from BADA tables until max speed or curise speed is reached		Cruise speed
Target speed - descent phase	0 kts	Nominal speed for flight level from BADA tables until stall speed is reached		0 kts
Rate of climb/ decent (ROCD)	Fixed	Obtained from BADA tables		Fixed
Fuel usage & emissions	Not calculated			
Aircraft mass	Fixed			
Aircraft turning	Instantaneous			

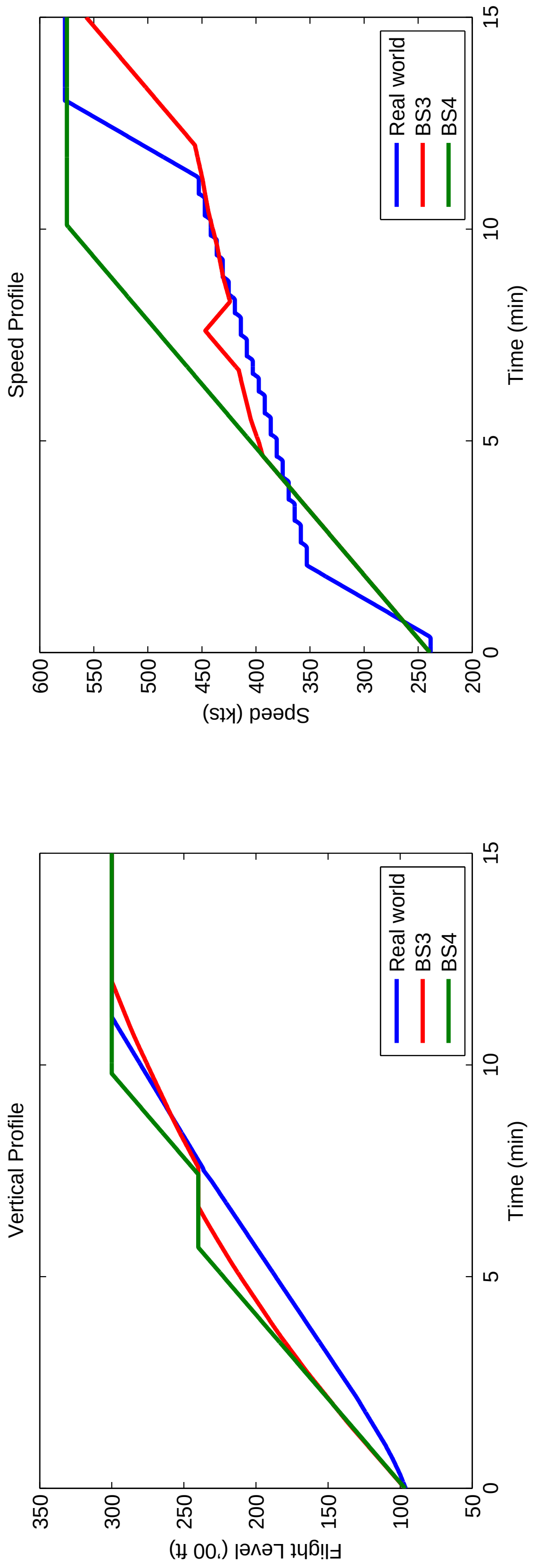


Figure 3.8: Comparison of vertical, speed and heading profile for an aircraft in the climb phase when using BS3 and BS4

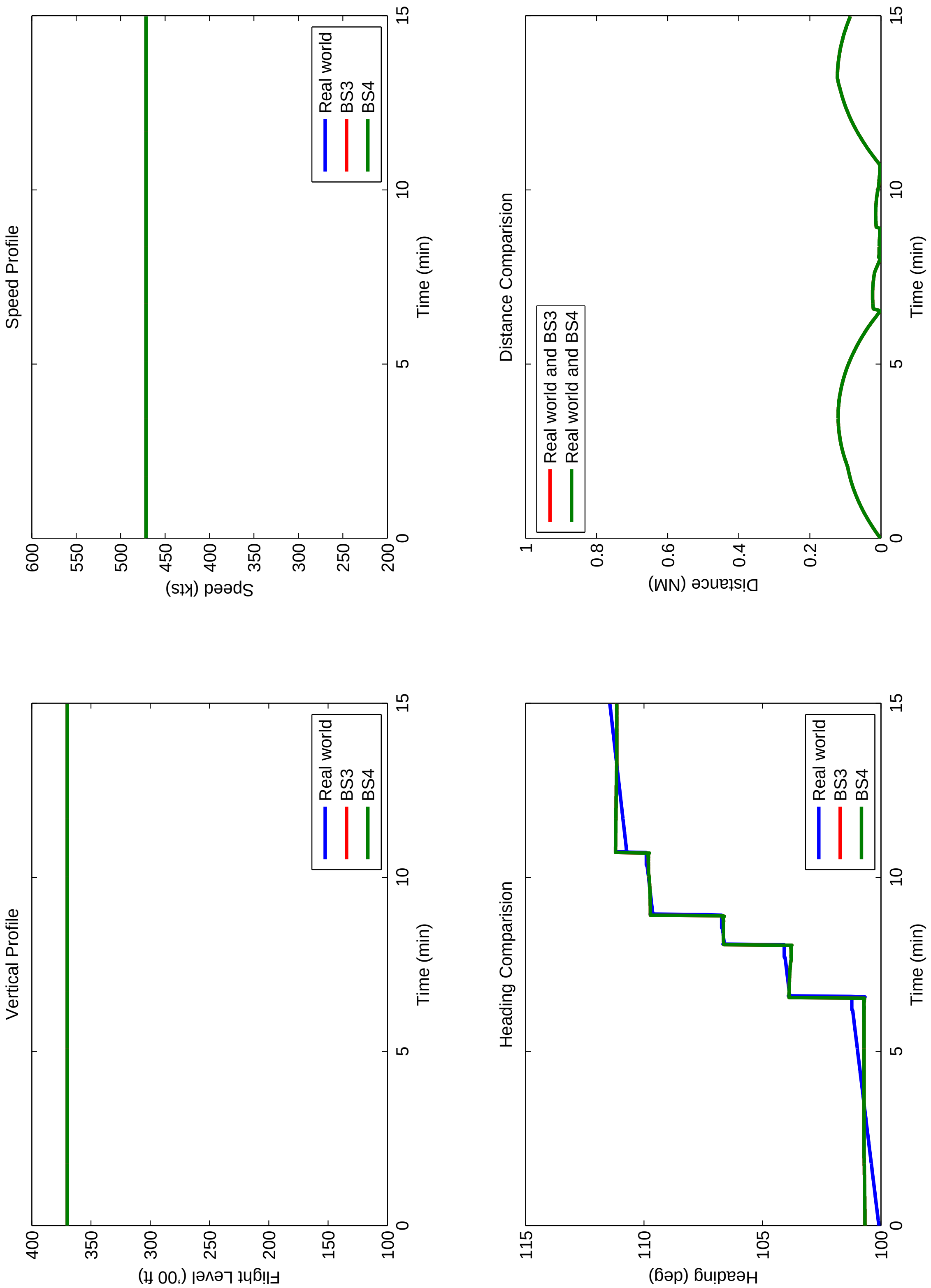


Figure 3.9: Comparison of vertical, speed and heading profile for an aircraft in the cruise phase when using BS3 and BS4

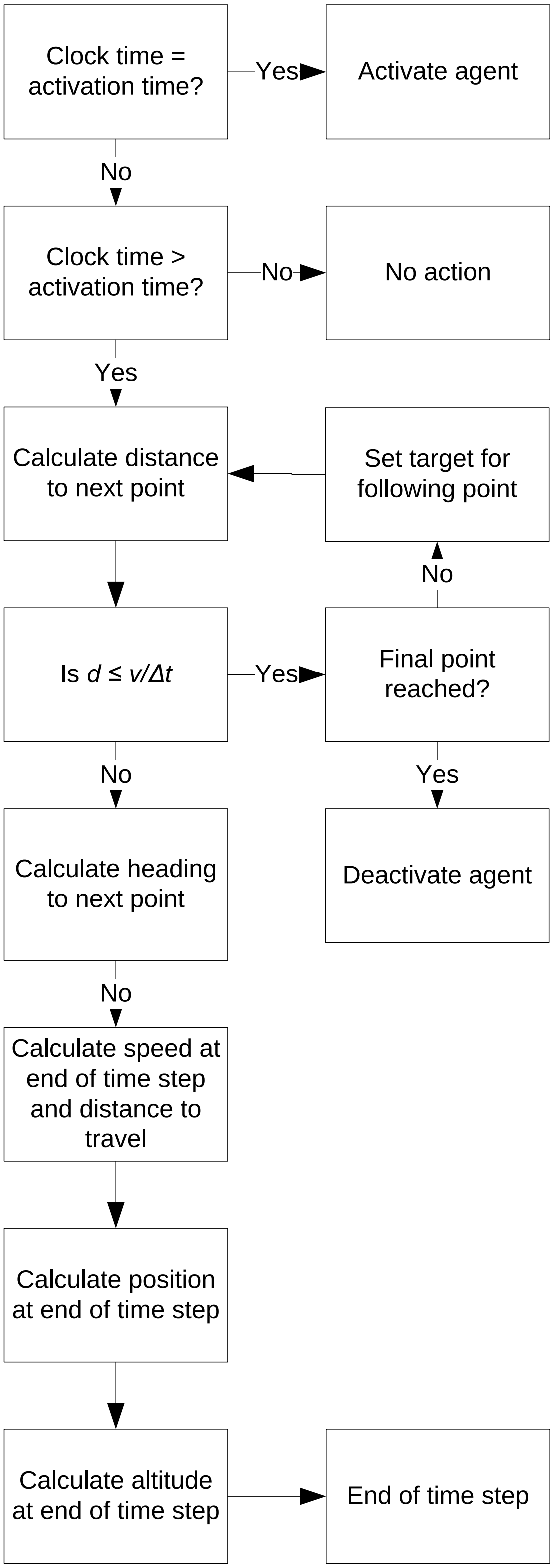
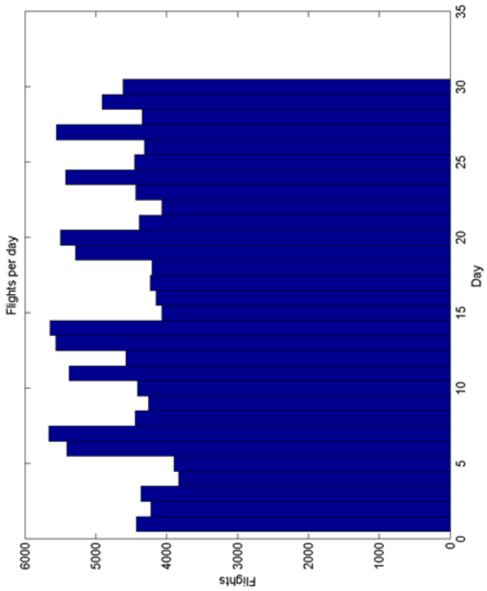
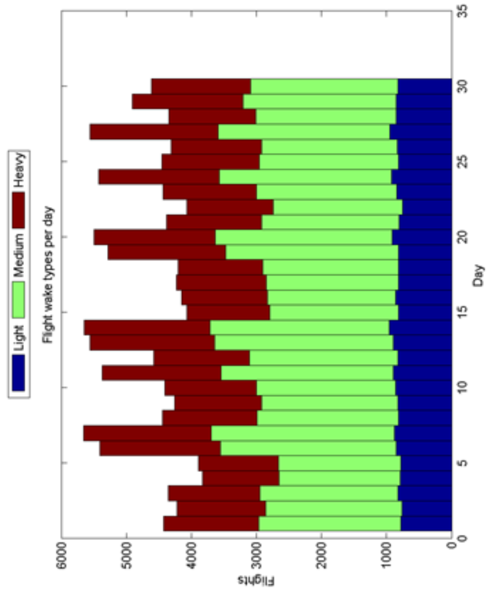


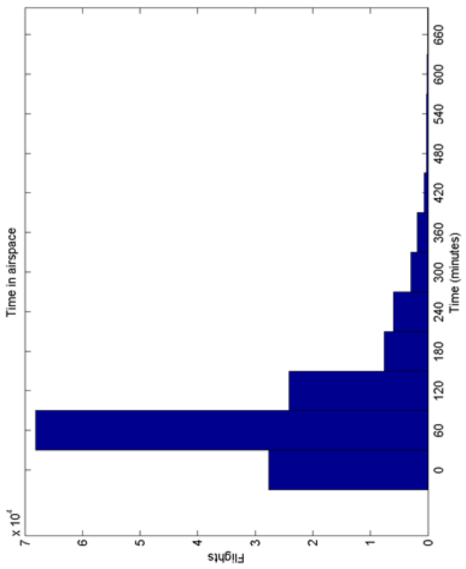
Figure 3.10: Overview of the aircraft trajectory calculation process



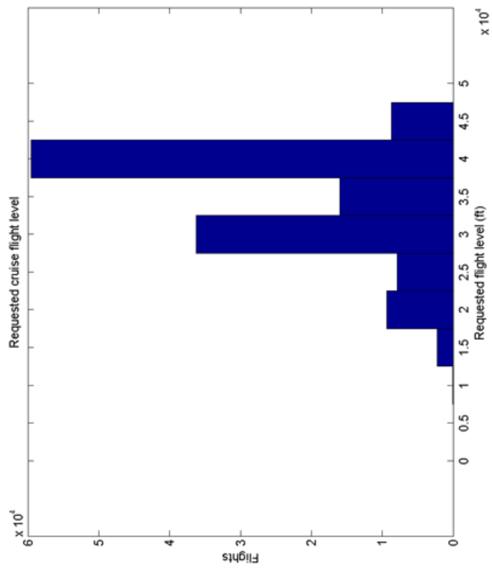
(a) Distribution of flights per day



(b) Distribution of aircraft types for flight per day



(c) Distribution of flight durations



(d) Distribution of cruise flight levels

Figure 3.11: Overview of flights generated for the 30 day scenario

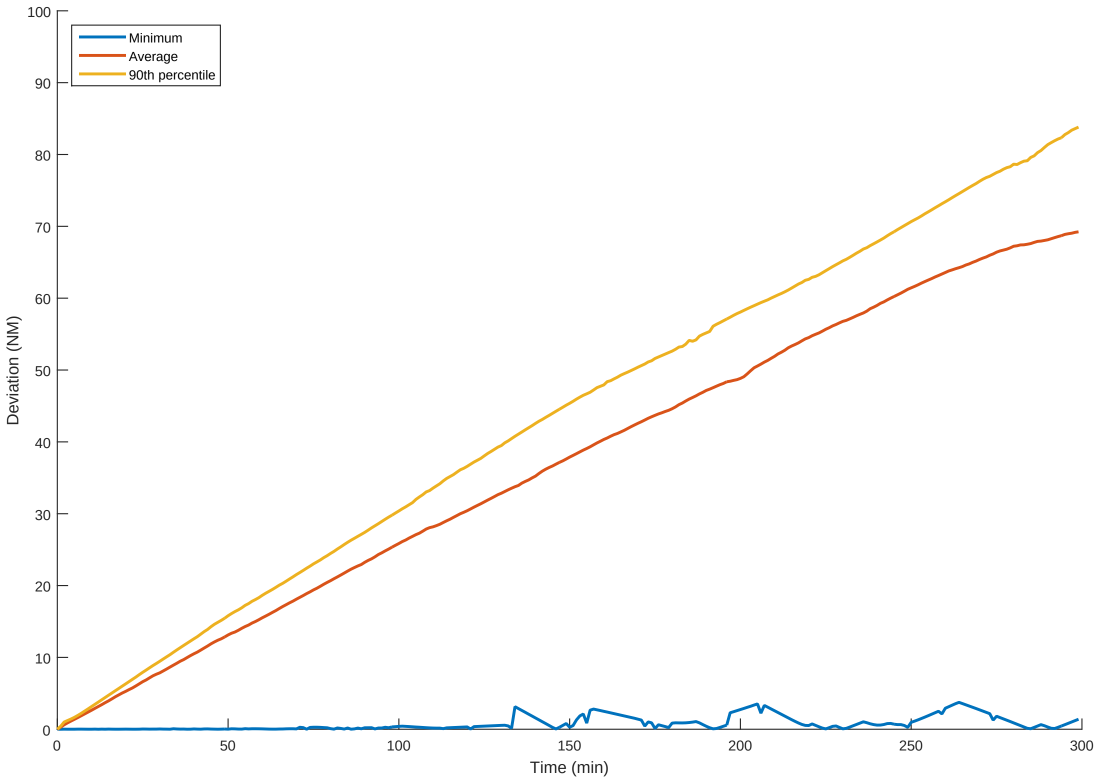


Figure 3.12: Flight track deviation comparison between the real world and BS1

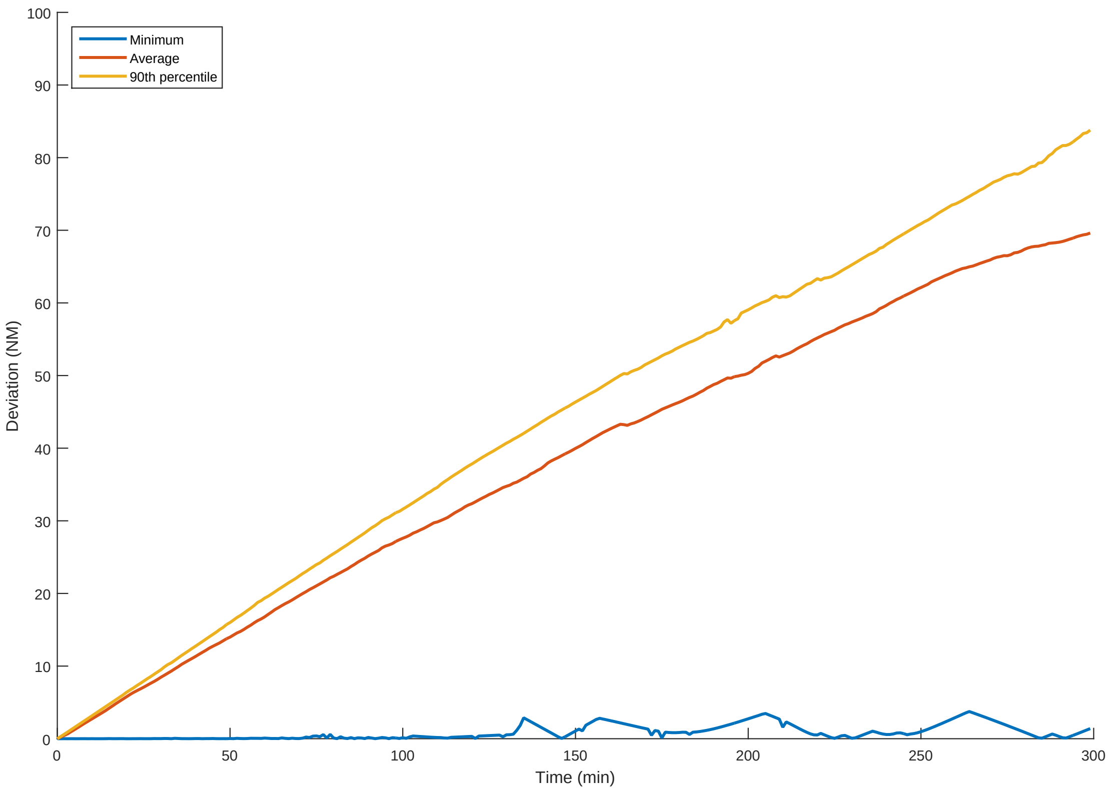


Figure 3.13: Flight track deviation comparison between the real world and BS2

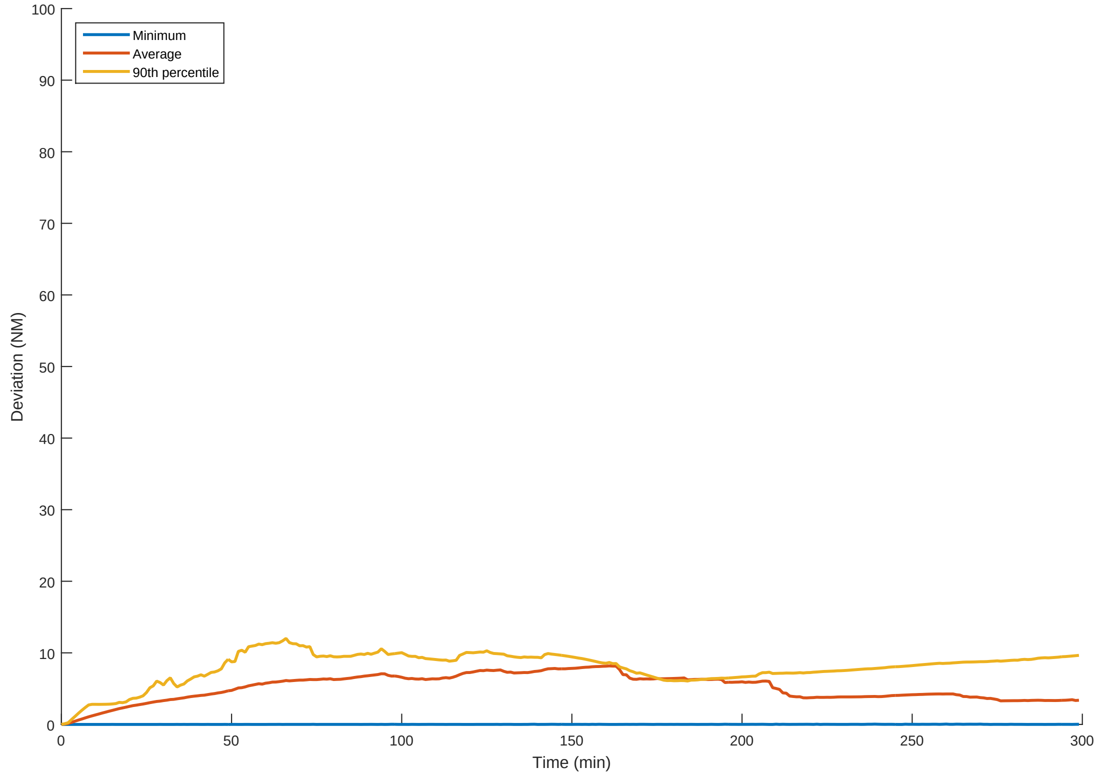


Figure 3.14: Flight track deviation comparison between the real world and BS3

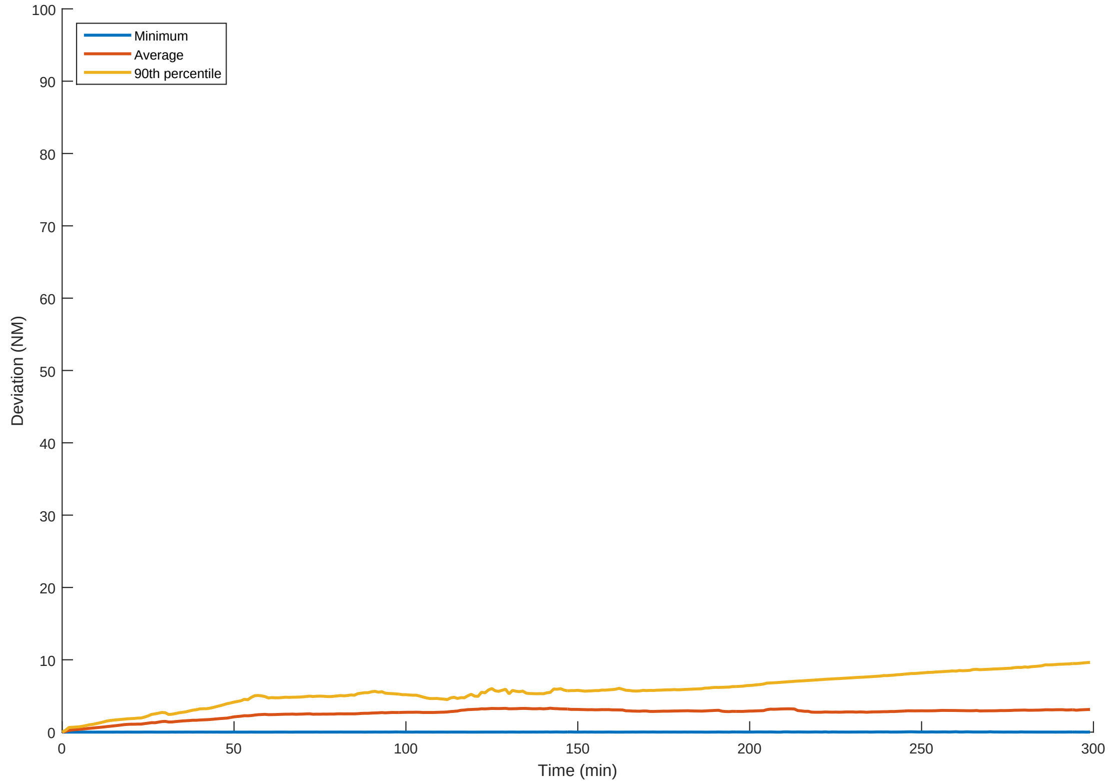


Figure 3.15: Flight track deviation comparison between the real world and BS4

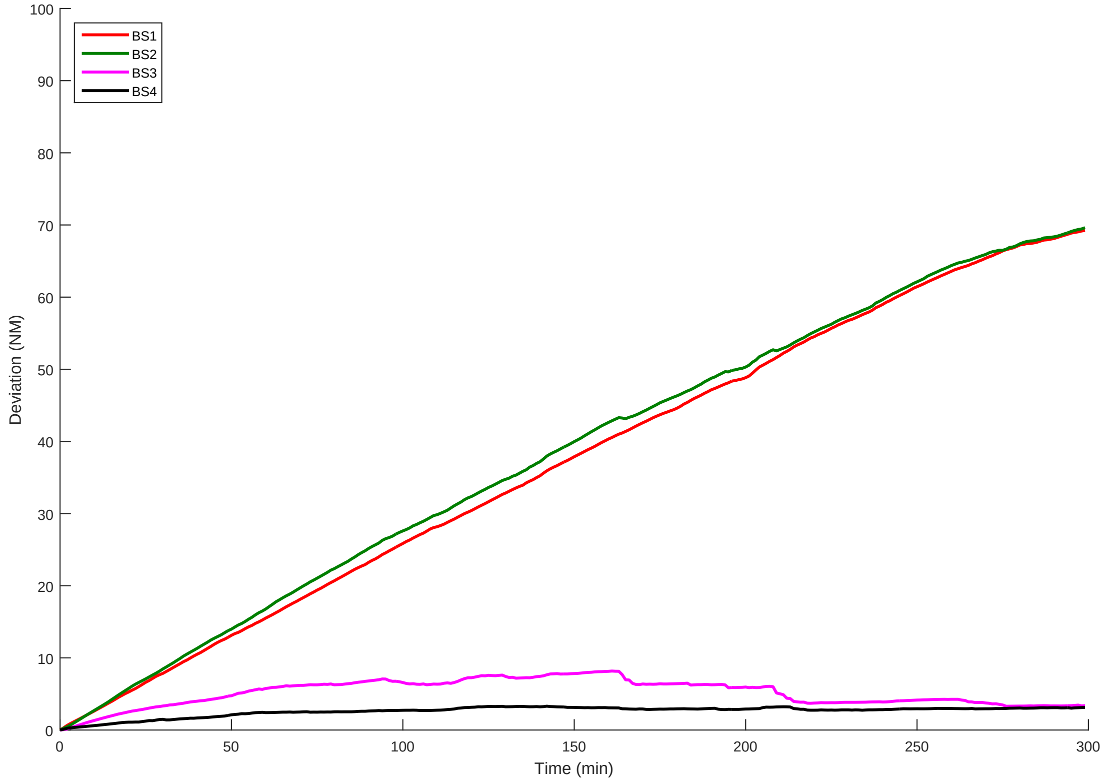


Figure 3.16: Flight track deviation comparison between the real world and the four simulators

Chapter 4

Real-Time Airspace Complexity Adjustment

Work in this chapter has been partially published in the following papers:

1. Amin, R., Tang, J., Ellejmi, M., Kirby, S., & Abbass, H. (2013). Computational red teaming for correction of traffic events in real time human performance studies. In *USA/Europe ATM R&D Seminar, Chicago.*
2. Amin, R., Tang, J., Ellejmi, M., Kirby, S., & Abbass, H. (2014). Trading-off simulation fidelity and optimization accuracy in air-traffic experiments using differential evolution. In *Evolutionary Computation (CEC), 2014 IEEE Congress on* (pp. 475-482). IEEE.

In order to facilitate the adjustment of the ATC's workload in real-time, it may be necessary for involved parties (ie. ATCs, pilots, etc.) to take actions to change from their current or planned states. To generate these actions and evaluate their effectiveness we require an optimisation system. Optimisation can be a time-consuming activity, especially when simulation is required to evaluate possible solutions.

In this chapter we explore the effectiveness of using low fidelity simulators for real time optimisation. In order to explore this problem we will first present an approach for adjusting ATC workload in real-time. This approach requires a shadow simulator and an optimisation system. Based on air traffic conditions obtained from a real-time operational environment, the optimisation system aims to push the workload up or down towards a target using goal programming by periodically generating suggestions for changes to the state, such as aircraft climbing to a given flight level and skipping waypoints. The four different shadow simulators introduced in Chapter 3, with different levels of fidelity, were used to evaluate the solutions as part of the optimisation system. We saw in Chapter 3 that the output from these simulators are varied and is likely to result in different outcomes when compared to each other and compared to what is expected in reality.

4.1 Airspace Complexity

The operation of the air traffic system is somewhat limited by the capabilities of the human air traffic controllers. The air traffic sectors are designed, and sometimes even re-designed, so as to maintain traffic levels which do not exceed the capabilities of the ATC assigned to those sectors. Due to these human limitations, over work may decrease the efficiency of the system and may lead to operational errors such as violation of separation. Meanwhile, under work can lead to wasting resources. Therefore, it is important to accurately measure the workload for ATC. The ATC's workload is a subjective measure that is driven by airspace complexity (Hilburn, 2004). The airspace complexity is a measure of the collective effect of various factors and variables which determine the difficulty and effort required to safely manage the air traffic for a particular situation (Prandini *et al.*, 2011). The measure of complexity is important for maintaining levels of air traffic which do not exceed the workload capabilities of the human ATC.

Numerous metrics have been proposed in the literature for measuring air traffic complexity over the last few decades. Many indicators, particularly the early developments, primarily focused on the number of aircraft within a sector as the key driver for ATC workload and therefore the effect of this characteristic has been studied the most. After further research in this field, it has been generally agreed that factors other than the number of aircraft contribute to the ATC workload. These factors include both structural and flow characteristics of the sector. Mogford *et al.* (1995) identified 40 such factors which effect the airspace complexity. These factors were grouped into 5 broad categories by Song *et al.* (2012) and a summary of these categories is presented below.

- *Static factors:* the physical or fixed characteristics of the sector including the flight routes, sector geometry and size and any terrain features (such as mountains).
- *Dynamic factors:* the dynamic flow of air traffic including the density of aircraft in the sector, conflicts, types of aircraft, altitude changes and trajectory changes.
- *Human factors:* communication requirements between aircraft and other sectors and the amount of communication that is required.
- *Equipment factors:* the type and availability of equipment such as radar and radio.
- *Unexpected factors:* adverse weather and other uncertain events.

The static factors for a particular sector remain constant throughout the operation of the sector and cannot be changed with ease or regular frequency. The human and equipment factors are difficult to influence in an operational environment and are therefore outside the scope of our study. The unexpected factors, particularity adverse weather conditions and equipment failure, cannot be be called upon on demand and are also outside our scope. This leaves our focus with the dynamic factors which can be easily influenced by various stakeholders of the system at short notice.

Hilburn (2004) provides a comprehensive review of airspace complexity metrics proposed prior to 2004, all of which, to some degree, incorporate the factors identified by Mogford *et al.* (1995). Since Hilburn's publication numerous extensions of these metrics have also been proposed to predict ATC workload and airspace complexity. Examples include Vogel *et al.* (2013); Simić and Babić (2015); Lee *et al.* (2009). However the dynamic density (DD) model proposed by Laudeman *et al.* (1998) remains a popular method for measuring airspace complexity and other related metrics among various applications. The dynamic density models are a measure of the airspace complexity where the dynamic factors of the traffic are combined linearly or through neural networks using weights which have been determined based on ratings of different traffic scenarios by professional ATCs and regressions with traffic activity data. This means the DD models incorporates both subjective and objective workload measurements. The model proposed by Laudeman *et al.* (1998) incorporates nine factors which are outlined below.

- N - the number of aircraft in the sector
- NH - the number of aircraft in the sector that made a heading change greater than 15° during an interval of 2 minutes
- NS - the number of aircraft in the sector which had a speed change of greater than 10 kts during an interval of 2 minutes
- NA - the number of aircraft in the sector which had an altitude change greater than 750 ft during an interval of 2 minutes
- $S5$ - the number of aircraft in the sector with 3D Euclidean distance between 0-5 NM
- $S10$ - the number of aircraft in the sector with 3D Euclidean distance between 5-10 NM
- $S25$ - the number of aircraft in the sector with lateral distance between 0-25 NM and vertical separation less than 2000 ft
- $S40$ - the number of aircraft in the sector with lateral distance between 25-40 NM and vertical separation less than 2000 ft
- $S70$ - the number of aircraft in the sector with lateral distance between 40-70 NM and vertical separation less than 2000 ft

Using this model the complexity, C, of a given situation at time T can be calculated using Equation 4.1 which is a weighted sum of the different characteristics.

In this equation $d_{i,T}$ is the value of the ith characteristic for the current time and W_i is the weight assigned for the characteristic.

$$C_T = \sum_{i=1}^{9} W_i d_{i,T} \tag{4.1}$$

In order to calculate the values for the characteristics we require track data from the aircraft. For calculating NH, NS and NA we are also required to keep track log of the traffic for the preceding two minute interval. The weights ensure that events which pose a greater risk, which may need immediate ATC attention, have a bigger influence on the complexity. For example, the number of aircraft pairs in the $S25$ group have a bigger effect on the complexity than those in the $S40$ group, as those in the $S25$ group are a bigger immediate concern for the ATC and pose a greater risk of collision.

4.2 Optimisation

Adjusting the airspace complexity is a multi-objective optimisation problem as we are required to alter various aspects of the air traffic environment in order to achieve different complexity levels. As the complexity is a weighted sum of the various aspects of the air traffic environment we are trying to optimise, it makes the problem ideal for a goal programming approach (Lee *et al.*, 1972). Goal programming is a common technique that is particularly useful when it is required to simultaneously consider multiple criteria for stratifying a solution. This technique allows for setting target values for each criterion and then optimizing for the sum of the deviations of each criterion from their respective aspiration level. There are several methods available for optimisation for problems when using goal programming such as in our real-time complexity adjustment system. One of these methods is the weighted sum goal programming method where the optimization is done by assigning weights to each goal and then minimizing the weighted sum of the deviations from targets (Hu *et al.*, 2007). Other alternative goal programming optimization methods include the MINMAX and Lexicographic methods (Bertolini and Bevilacqua, 2006). In the MINMAX method, the maximum deviation from the target is minimized instead of minimizing the weighted sum of the deviations. This method also makes use of weight factors. The lexicographic method assigns priorities for different goals and goals with the highest priority are are considered first(Bertolini and Bevilacqua, 2006). All three of these goal programming methods rely on the use of user defined factors which are subjective to the user (Coello, 2000; Deb, 1999; Bertolini and Bevilacqua, 2006). For this reason it has been suggested that evolutionary algorithms are more flexible for the optimization phase of the problem(Deb, 1999). Evolutionary algorithms are a desirable method of solving optimization problems as the algorithms simultaneously work with a set of possible optimal solutions in a single run instead of a series of separate runs as required by some other methods (Coello, 1998). Using evolutionary algorithms for optimization also has the potential to produce multiple solutions in one run. This means we also obtain a selection of several near-optimal solutions and are able to make a choice from other solutions if necessary.

Evolutionary computation is a search technique using computational models of processes of evolution and selection inspired by the concepts and mechanisms of biological evolution (such as reproduction, mutation and selection) (Qin *et al.*, 2009). These models are implemented through evolutionary algorithms (EA) which have been used to solve problems in a variety of fields in engineering and science. EAs are a group of algorithms which simulate the natural evolution, selection, reproduction and variation of individuals in a population (Kicinger *et al.*, 2005). EAs work with a pool of solutions, often referred to as a population of individuals where the individuals are individual solutions to a problem. Each solution is encoded (represented) by a chromosome which is comprised of various attributes which describe the individual and the solution. Some of the commonly used evolutionary algorithms include genetic algorithms, evolutionary programming, genetic programming and differential evolution. All of these algorithms operate according to the following common procedure (Kicinger *et al.*, 2005):

1. Initialise a starting population
2. Evaluate all members of the population
3. While the termination condition is not satisfied:
   (a) Select individuals in the population to be parents
   (b) Create new individuals by applying the variation operators to the copies of parents
   (c) Evaluate new individuals
   (d) Replace some or all of the individuals in the current population with new individuals
   (e) Repeat steps a to d until the termination condition has been met

The key differences between the different EAs are primarily found when comparing their operations within the loop in part 3 of the above procedure representation. One of the key differences between EAs is the variation process which is used to generate new individuals. There are two main variation processes: mutation and recombination. Mutation involves selecting an individual and applying some variation to the elements of its chromosome, while recombination involves the the combination of parts of chromosomes from various individuals from the population. The newly created individual is evaluated and given a fitness value. The fitness indicates the solution's quality in relation to the problem. Then either all or only some of the new individuals replace the individuals in the population to create a new generation. The method of selection of individuals also varies between the different algorithms but is usually dependant on the fitness values of the individuals. Therefore it is important to design the fitness evaluation functions in a way that appropriately reflects the problem as this will be used to guide the direction of evolution of the individuals.

Sometimes it is necessary to evaluate several conflicting objectives in a problem. For this reason it may be necessary to employ a multi-objective fitness evaluation function or multi-objective evolution techniques. Multi-objective optimisation, unlike single-objective optimisation, requires a trade-off between objectives and usually there is no single optimal solution (Justesen, 2009). A common way to avoid the intricacies of multi-objective optimisation is to convert the fitness function into a single-object fitness function, such as by using goal programming.

4.2.1 Differential Evolution

Differential evolution (DE) (GA and Okdem, 2004) is an evolutionary algorithm that leverages direction information to guide the search. DE compares the fitness of an offspring directly to the fitness of the corresponding parent which results in faster convergence speeds than other EAs (He *et al.*, 2008). In addition DE is also easy to use and requires fewer control parameters and can find near optimal solutions regardless of the initial parameter values (Abou El Ela *et al.*, 2010). DE has been applied to a range of topics in science, engineering and management, such as logistics (Erbao and Mingyong, 2009; Mingyong and Erbao, 2010) and crew rostering for airlines (Santosa *et al.*, 2010).

Population and Chromosome Encoding

In order to make use of the DE algorithm, a structure needs to be formulated which can be used to represent any possible solutions to a problem. This structure, called a chromosome, contains a chain of elements, or genes, which represent different attributes of the problem and the values can be permuted or altered to create new solutions. The values for the genes often take the form of binary values, integer values or decimal values. The type of values for each gene is dependant on the goal of the problem being solved and usually incorporates some domain knowledge in its design.

For example, for an aircraft routeing problem the values in the chromosome could be $[1, 2, 3, 4, 5, 6]$ where each number represents the order in which six different cities are to be visited. The algorithm handles a population of NP individuals, where each individual is allocated a chromosome. Equation 4.2 shows an example of the notation that will be used to represent the ith individual in the population in further

DE operations. In this notation $x_{i,G}$ is the ith individual from generation G, j is the element number in the chromosome and N is the total number of elements. Each element is initialized with a random number within the range $min_j \leq x_{j,i,1} \leq max_j$ where min_j and max_j are predefined ranges for each element.

$$x_{i,G} = [x_{1,i,G}, x_{2,i,G}, ..., x_{j,i,G}], j = 1, ..., N \tag{4.2}$$

Mutation, Recombination and Selection

At each generation, *NP* number of mutant vectors are produced. Each mutant vector corresponds to an individual in the population, the target vector, and are generated by randomly selecting three individuals from the population, $x_{r1,G}$, $x_{r2,G}$ and $x_{r3,G}$, and applying Equation 4.3. In this equation $v_{i,G+1}$ is the donor vector corresponding to the individual $x_{i,G}$ and F is the scaling factor, a predefined fixed number between 0 and 2. The weighted difference of the elements of two of the selected individuals, $x_{r2,G}$ and $x_{r3,G}$; are summed with the elements of the third individual, $x_{r1,G}$, to produce the elements of the donor vector $v_{i,G+1}$. Equation 4.3 is the most basic version of the DE mutation strategies and is also the most commonly used strategy.

$$v_{i,G+1} = x_{r1,G} + F(x_{r2,G} - x_{r3,G}) \tag{4.3}$$

Once the donor vector, $v_{i,G+1}$, has been generated a trial vector, $u_{i,G+1}$, is generated by combining elements from both the donor vector, $v_{i,G+1}$, and the corresponding target vector, $x_{i,G}$, based on Equation 4.4. Parameter j in the trial vector is equal to the parameter j of the donor vector if a random number between 0 and 1 is less than or equal to the crossover rate, *CR*, otherwise parameter j in the trial vector is equal to parameter j from the target vector.

$$u_{j,i,G+1} = \begin{cases} v_{j,i,G+1} & \text{rand([0,1])} \leq \text{CR} \\ x_{j,i,G} & \text{rand([0,1])} > \text{CR} \end{cases} \tag{4.4}$$

Next the fitness of the trial vector is determined by using the elements from the trial vector as parameters in a simulation. The fitness is a score given to the vector based on the evaluation of the events during the simulation. If the fitness of the trial vector is better than the fitness of the target vector then the trial vector is selected to appear in the next generation, otherwise the target vector is selected.

4.3 Adjusting Airspace Complexity

For adjusting airspace complexity in real-time we require three components: a real-time air traffic environment, an optimisation system and a shadow simulator. The real-time air traffic environment may be the real air traffic system, a simulated environment which mimics the operation of the real world system, or even a combination of the two. The most important aspect that is required of the real-time component is that we are able to obtain data regarding the state of the air traffic on an ongoing basis. The optimisation component is used to generate solutions (ie. the changes to the state) which may lead to the airspace complexity towards a target. The shadow simulator is required to evaluate the solutions. An overview of the system can be seen in Figure 4.1. From this figure we can see the entire system operates as a loop. Periodically a snapshot of the air traffic is sent to the optimisation system. The optimisation system uses differential evolution to generate multiple lists of probabilities during each generation. Each of the lists of probabilities are used separately to generate aircraft, request and execution time combinations. These combinations are used as input into the shadow simulation for evaluation. The shadow simulator simulates the scenario from the snapshot time for a predefined period, the lookahead period, with these requests being executed. At the end of the simulation the measured complexity for this period is used to evaluate the effect of implementing these requests. Once the optimisation system has reached it's limiting condition, the best solution is returned to the real-time environment for implementation. This loop is continually repeated. This is to reflect the fact that the air traffic environment is a dynamic environment as aircraft can enter or exit sectors at any given time, deviate from their trajectories and take other actions. Due to these uncertainties, it is unlikely that any prediction about the future state of the traffic will hold valid for a very long time. For this reason we can make predictions on a receding time horizon, $T + T_h$, and produce new optimal solutions after a given time interval, T_i, to incorporate new data that may have become available during that interval (Rawlings and Muske, 1993; Chaloulos *et al.*, 2010).

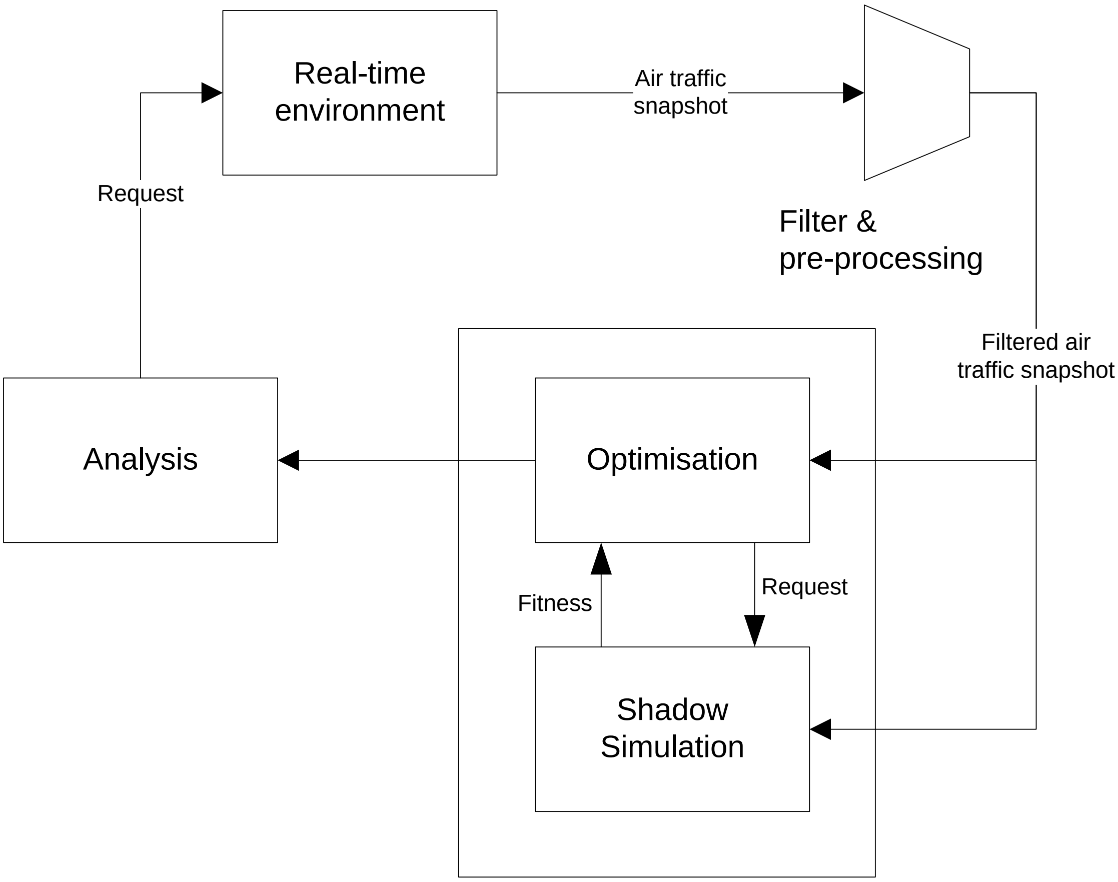


Figure 4.1: Overview of the real-time airspace complexity adjustment system

The data obtained from the real time environment consists of a set of aircraft $A = \{a_i\}_{i=1}^{N}$ where

$$a_i = (r, T_a, v_c, z_c, \lambda_T, \phi_T, z_T, v_T) \tag{4.5}$$

a_i includes the aircraft's flight plan and current positioning data. The flight plan data includes the aircraft's route (r), activation time (T_a), cruise speed (v_c) and cruise flight level (z_c). The aircraft's route, $r = W_1, W_2, , W_j$, contains j waypoints, W, each with a latitude-longitude coordinate and optionally an elevation. Waypoint W_1 is the activation point for the aircraft and W_j is the deactivation or final point. The snapshot data may also optionally contain the current positioning data for the aircraft. This includes the aircraft's current latitude, λ_T, longitude, ϕ_T, altitude, z_T, and speed, v_T.

Before the snapshot data is passed to the optimisation system a pre-processing phase occurs. During this phase the positioning and waypoint data is converted to a different projection system, if required. Additionally, aircraft which don't enter the measured sector, ie. the sector for which the complexity is being adjusted, during the time horizon are filtered out. This is done by simulating the aircraft according to their flight plans and current positioning data from the time of the snapshot for a period equal to the time horizon. During this simulation all the aircraft which entered the measured sector are recorded and the aircraft which did not enter during this period are disregarded for use in further simulations for the generation of requests for this interval. This is done in order to reduce the total number of aircraft being simulated which do not have a bearing on the final outcome of the optimisation as only aircraft which enter or are already inside the measured sector during this period determine the complexity. Additionally as there are fewer aircraft being simulated, the overall computation time of the shadow simulation will be faster than simulations which included these inconsequential flights and will therefore allow for more simulations in the same time frame. The aircraft which enter the measured sector are then recorded and their data from the snapshot is passed to the optimisation system. This initial simulation of the aircraft also provides us with a baseline measurement of airspace complexity. The optimisation system then produces potential solutions which are then evaluated using the shadow simulator.

4.3.1 Aircraft requests

The potential solutions generated by the optimisation system are in the form of actions which are to be taken by the aircraft. These actions can be thought of as requests made by pilots and ATCs to change the trajectories of the aircraft. The requests are generated as a list of requests $Q = \{q_k\}_{k=1}^{M}$, where $q_k = (a_i, T_r, Y, \delta)$. M is the total number of requests in the list, a_i is the aircraft which the request will effect, T_r is the execution time of the request, Y is the type of request to be made and δ is a value specific to the request, such as the number of feet to climb. The types of request that can be selected are:

- *Climb*: The aircraft climbs 2000ft
- *Descend*: The aircraft descends 2000ft
- *Change speed*: The aircraft exits the sector up to 5 minutes earlier
- *Change speed*: The aircraft exits the sector up to 5 minutes later
- *Turn right*: Change heading by 5° in a clockwise direction, then return to original route after 2 minutes
- *Turn left*: Change heading by 5° in a counter-clockwise direction, then return to original route after 2 minutes
- *Skip upcoming waypoints*: Skip a number of upcoming waypoints such that it results in a net heading change greater than 5°

In an effort to not overwhelm the ATC, only a limited number of requests can be made during a time interval. Limiting the number of requests in a given period also ensures that the requests can be issued and implemented within a sufficient time frame. In order to obtain a wider variety of request types, a predefined number of instances of each request type are allowed to be selected within a given period. In real air traffic environments the pilot and air traffic controller may need to communicate and approve the action before some of these requests can be executed. But for the scope of this study it is assumed that all requests are instantly approved by both the ATC and the pilot. Examples of the effect of some of the requests types on the trajectories of the aircraft can be seen in Figure 4.2. The blue lines in this figure indicates the flight level, speed or route of the aircraft as according to the flight plan. The blue circles indicate waypoints in the flight plan, while the green circles indicate the position of the aircraft when the request is executed and the red line indicates the alternate flight level, speed or path taken by the aircraft as a result of the execution of the request. The climb/descent and speed change requests are expected to have a direct effect on the number of aircraft changing altitude (NA) and speed (NS) components of the complexity calculation. However the effect of the turn and skip waypoint requests may not necessarily influence the number of aircraft changing heading (NH) as the minimum 5° change in heading resulting from the request is lower than the threshold for the component. If the request is executed within close proximity of a waypoint or the flight plan provides a favourable combinations of waypoints, the execution of these requests may also contribute to the NH component. All of these requests, however, are likely to effect the number of aircraft in the sector (NA) and the various proximity components.

4.3.2 Request Probabilities

The optimisation system aims to produce one request which is to be executed within the time horizon, T_h. The combination of which aircraft which will make the request and the type of the request is based on probabilities obtained from the chromosomes (individual solutions) which were generated using differential evolution. The given probabilities includes a set of probabilities for each request type, $R = \{P(r_j)\}_{j=1}^{N}$, where N is the number of request types and r_j is request type number j; and another set of probabilities for each aircraft, $A = \{P(a_i)\}_{i=1}^{M}$, where M is the number aircraft entered the measured sector within the time horizon, a_i is aircraft number i. The probability of a particular request being generated for a certain aircraft at a given time is found using Equation 4.6.

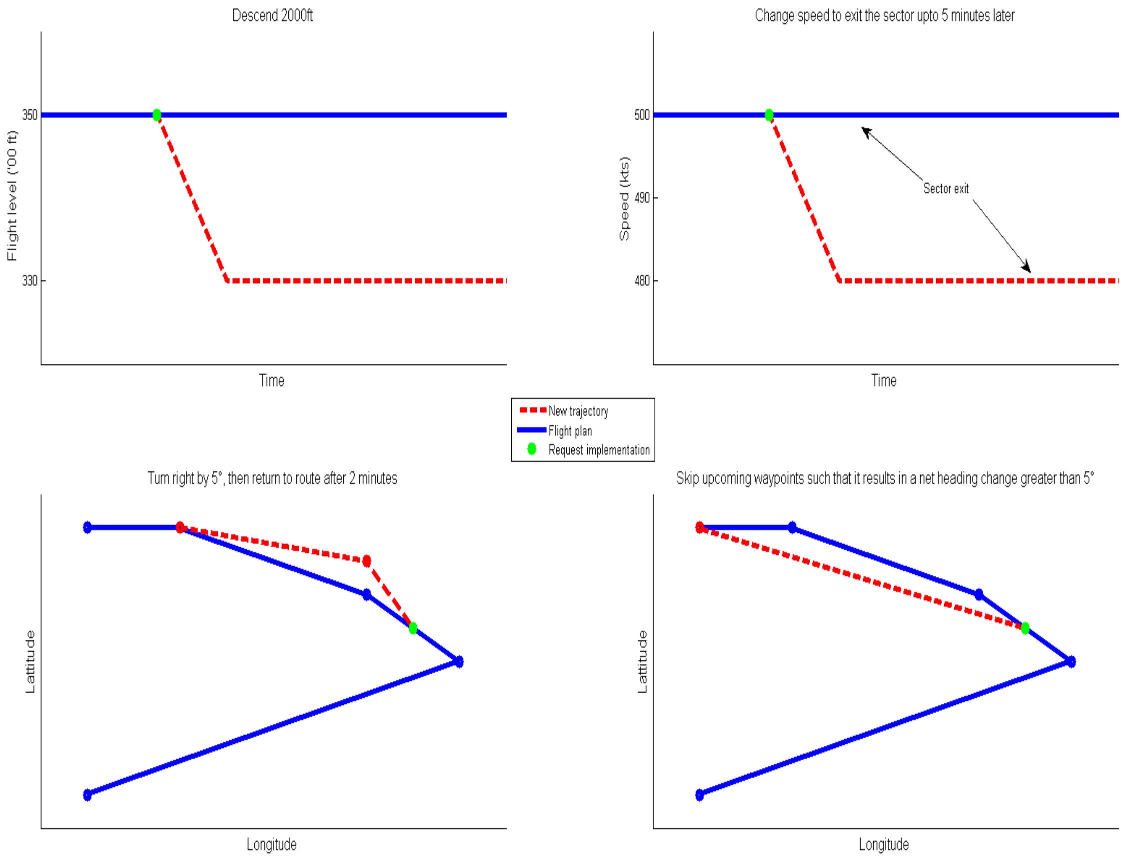


Figure 4.2: Examples of the effect of the implementation of the different requests types.

$$P(r_i|a_j) = P(r_i) \times P(a_j) \tag{4.6}$$

The execution time for each request were also obtained from the chromosomes generated using differential evolution.

4.3.3 Request Execution Time

The execution time for each request is determined from a time range relative to the time of the snapshots. The snapshot are produced at an interval of T_i. Details of the time ranges can be seen in Table 4.1 and a graphical representation of the ranges can be seen in Figure 4.3. The look ahead period begins at the time of the snapshot, T_{LS}, and lasts until T_{LE}, a period of T_h after T_{LS}. If T_{LE} is a time after the end of the scenario, T_{SE}, then the the end of the look ahead period is set to the end of the scenario. When the optimisation cut-off time, T_{OE}, is reached in the real-time environment the optimisation is stopped and the best evaluated set of solutions thus far are sent to the post processing component. This time is set at time T_{OT} after the snapshot time. The execution time range for the request starts at the optimisation cut-off time and lasts for a period equal to T_i. The earliest execution time for the request is T_{ES}, a time after the optimisation cut-off time, and the latest time is T_i after T_{ES}. As we can see from Figure 4.3, the latest possible execution time for the request occurs after the following snapshot is sent, but not after the earliest possible execution time for the following request.

Table 4.1: Time window for optimisation and request execution

Event	Time
Scenario start time	T_{SS}
Scenario end time	T_{SE}
Traffic snapshot interval	T_i
Look ahead horizon	T_h
Look ahead time start	T_{LS} = Traffic snapshot time $= T_{SS} + iT_i$ where i = interval number
Look ahead time end	$T_{LE} = MIN[(T_{LS} + T_h), T_{SE}]$
Optimisation cut-off time	$T_{OE} = T_{LS} + T_{OT}$
Request execution time range start	$T_{ES} = T_{OE}$
Request execution time range end	$T_{EE} = T_{ES} + T_i$

4.3.4 Shadow Simulation and Request Evaluation

Once the aircraft, request type and execution time combination for each solution have been generated they are evaluated using the shadow simulator. The shadow simulation is run without visualisation and at a much faster rate than real-time for the period stating from T_{LS} until T_{LE}. During the simulation the airspace complexity is measured at a frequency of 10 times a minute. At the conclusion of each simulation, the complexity is evaluated against a predefined target level. This evaluation is conducted using a goal programming approach.

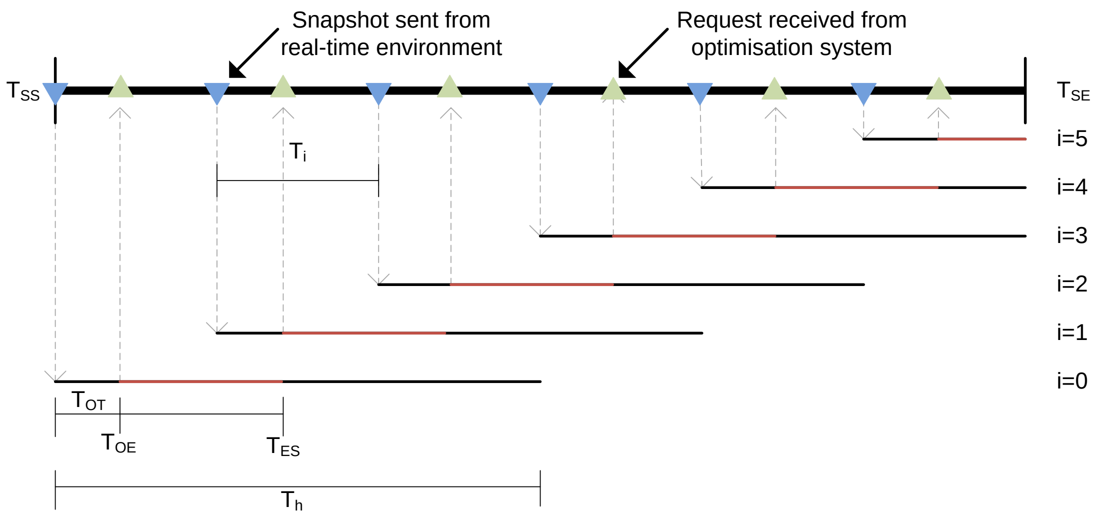


Figure 4.3: Time window for optimisations and request execution

For our system we calculate the deviation of each of the samples from the target level of complexity using Equation 4.7, where x_i is the ith complexity sample, C_g is the goal (or target) level of complexity, d_i^+ is positive deviation from the goal and d_i^- is negative deviation from the goal. The negative deviation quantifies the number of units by which the target has not been satisfied while the positive deviation quantifies the number of units by which the target has been surpassed. At least one of the negative deviation or positive deviation will be equal to zero and both will be equal to zero if the goal is achieved (ie. $x_i = C_g$). As we are only using one type of attribute no weights are required for the comparison of the deviations. Once we have calculated the deviation from the target for each of the samples, we can determine the objective function, f, for this solution using Equation 4.8. The objective function is determined by finding the average deviation from the target level during the lookahead period for the M number of samples calculated during this period. This objective function is then returned to the optimisation system. As all deviations are non-negative, a global optimal solution for our problem occurs when this value is zero and therefore the aim of the optimisation is to minimise f.

$$x_i - d_i^+ + d_i^- = C_g \tag{4.7}$$

$$f = \frac{\sum_{i=1}^{M} d_i^+ + d_i^-}{M} \tag{4.8}$$

4.3.5 Differential evolution

Differential evolution (DE) was used as the search technique to optimize the objective function. A list of un-normalized probabilities were generated randomly and then was used as an input to generate a list of requests for use in the shadow simulation. Based on the evaluation from the shadow simulation, each of the lists were given an objective function to determine the fitness of each list. DE is used to find the list of probabilities which can generate the minimum value for the objective function.

Each solution (called a chromosome or individual) is represented naturally as a vector of real numbers. As can be seen in Figure 4.4, the chromosome used in this system included one parameter for each aircraft and another for each of the seven requests types. There is also a parameter for the execution time of the request. The seven parameters for the request types represent the probability of that request type being made while the parameters for the aircraft represents the probability of the request coming from the corresponding aircraft. The parameters are initialised with random values. The minimum and maximum values for the execution time are obtained from Table 4.1 while the minimum and maximum values for the remaining parameters are 0 and 100 respectively. If a particular request type or aircraft are not allowed to be selected for the current interval then its maximum value is set to 0, thus having zero probability of being selected and making it impossible to be selected using Equation 4.6.

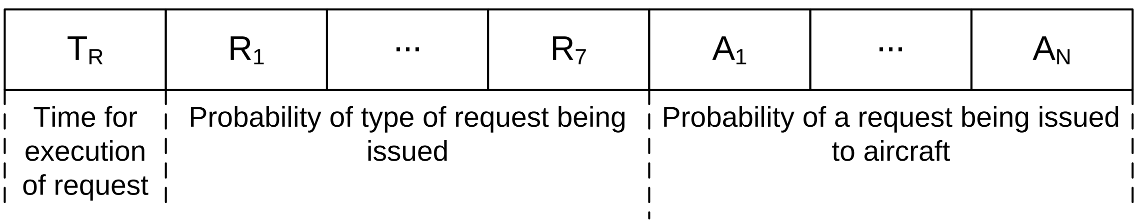


Figure 4.4: Chromosome representation for use in the differential evolution process

4.4 Experiment - effect of differential evolution parameters

Several different experiments were conducted in order to demonstrate different aspects of the system. During each experiment a number of different simulators were also used for the shadow simulation component with the aim of demonstrating the effect of simulator fidelity on the system. These simulators included the four low fidelity simulators and the perfect model introduced in Chapter 3. Using the perfect model as a shadow simulator allows us to mimic reality with the high fidelity simulator. It is not practical, nor necessary, to use the real air traffic system for the real-time environment in order to demonstrate the operation of the system or to demonstrate the effects of the different simulators. Therefore we used the perfect model in real-time mode to represent the real world real-time environment.

The first experiment involves operating the system with different parameters for the optimisation component. This provides us with an indication of the sensitivity of the system to the parameters and also provides us with a basis for selecting parameters for use in the following experiments. Two of the differential evolution parameters, the crossover rate (CR) and the scaling factor (F), were altered for this experiment. Three different values were tested for each parameter, giving us 9 different combinations. Each combination was run with each of the five simulators as the shadow simulator and repeated 20 times for each combination, giving us a total of 900 runs.

Scenario and airspace complexity target

Each of these 900 runs (and also runs in the subsequent experiments) were conducted with the same input scenario. This input scenario contained 45 aircraft with a range of different characteristics. Some aircraft were activated outside the measured sector, some had origin airports within the sector and some had destination airports within the sector, but all aircraft passed through the sector at some point of their flight. The measured sector was defined in 3D space between the elevations of 29,000ft and

41,000ft. Most aircraft entering the sector had an intersecting route with another aircraft that was also within the sector, but intersection may not necessarily result in a violation of separation required to increase the airspace complexity. A screenshot from the real-time simulation can be seen in Figure 4.5. This figure shows the measured sector and the flight plans of some of the aircraft in the input scenario. The real-time simulation was run for 60 minutes in each of the experiments.

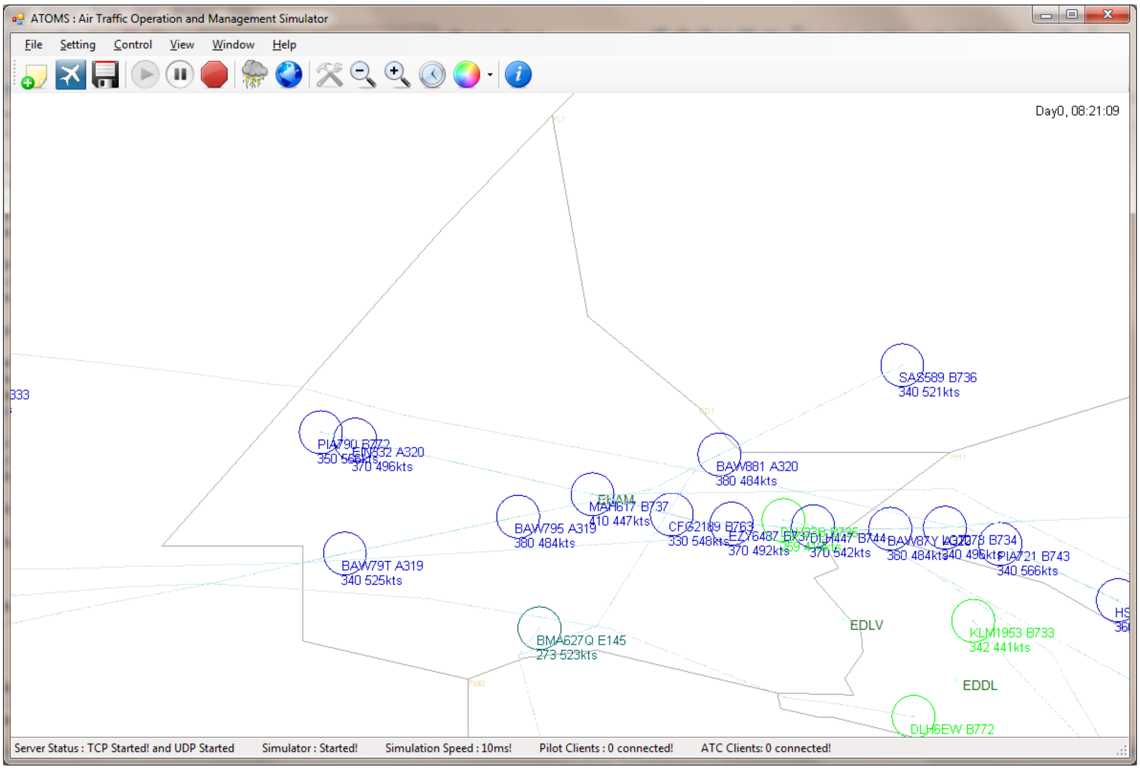


Figure 4.5: Screenshot of the realtime visual simulation

The target for the airspace complexity was set as the 50th percentile of the baseline complexity for the input scenario. To determine the value for this percentile, the entire scenario was simulated using the perfect model without the execution of any requests and the complexity was measured at a frequency of ten times a minute based on the previous two minutes of activity. This measurement is what we would expect to obtain from the real world. The 50th percentile of the measured complexity was found and set as the target. The complexity for this scenario can be seen in Figure 4.6. From this figure we can see that the complexity slowly rises to a peak around 20 minutes from the start of the scenario. This peak is

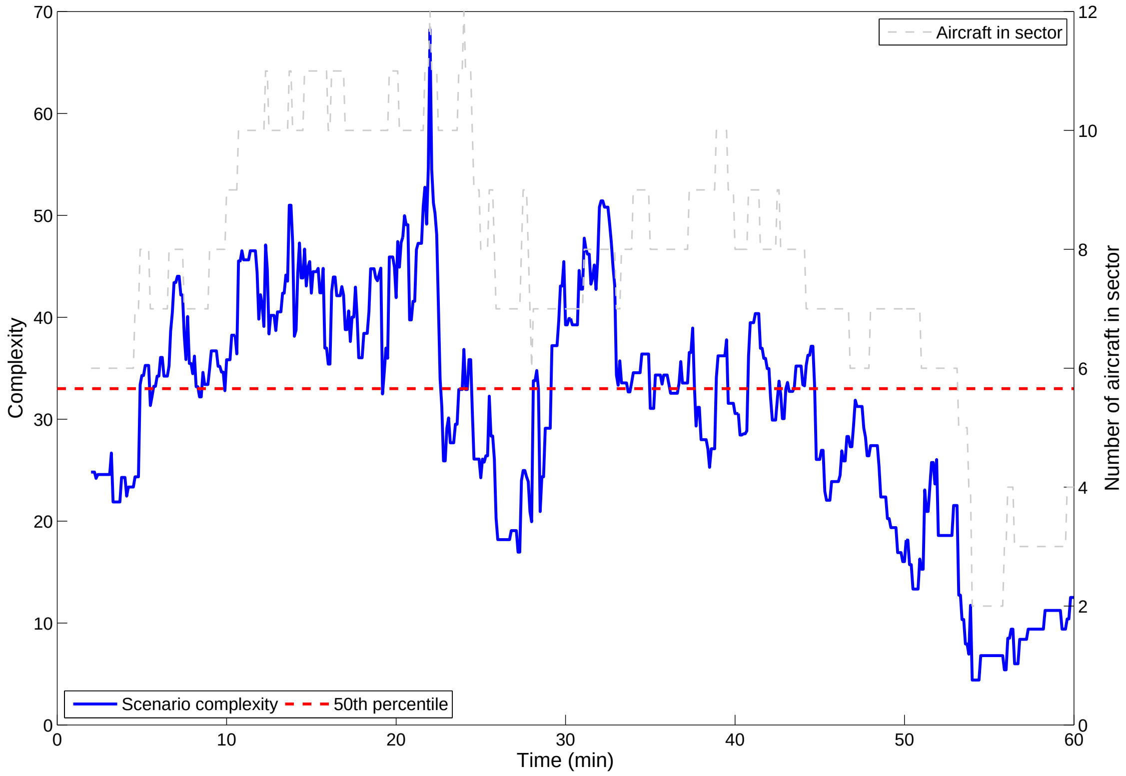


Figure 4.6: Airspace complexity for the input scenario measured using the perfect model

closely related to the number of aircraft in the sector and represent the time during this scenario at which the greatest number of aircraft are present in the sector. A comparison of the baseline complexity as measured from the perfect model and the four Basic Simulators can be seen in Figure 4.12. While the absolute values of the complexities may be different from what was measured using the perfect model, we can see that the general trend of the complexity measured by the four simulators follows the one measured using the perfect model. This is evident when assessing the correlation coefficient between the plot generated using the perfect model and the ones generated by the other simulators. Using the data presented in Figure 4.12, we have a correlation coefficient of 0.904 between the perfect model and BS1, 0.8948 with BS2, 0.901 with BS3 and 0.907 with BS4. A comparison of the complexity as measured by BS1 and BS2 and their higher fidelity versions, BS4 and BS3 respectively, can be seen in Figure 4.13 and a plot of the cumulative complexity as measured by each of the simulators can be seen in Figure 4.14. The differences in the complexities as measured by the different simulators can be attributed to their fidelity. According to the scenario plan, a set number of aircraft will be within the sector in any given period of time, and so we can see that the general trend of the plots are similar. However, there are still discrepancies at several locations. While the scenario plan may dictate the number of aircraft within the sector for a given period, it does not dictate the exact moment the aircraft enter or exit the sector. These times are dependent on basis of movement of the aircraft as determined by the simulator being used. The differences in entry and exit times also leads to small differences in times when waypoints are reached and therefore also leads to small differences in speeds, altitudes, headings and proximity with other aircraft which may all lead to differences in the complexity.

Snapshot interval and lookahead horizon

Snapshots containing the traffic state from the real-time simulator were received at an interval of 5 minutes. Once the snapshot is been received, the scenario is simulated for a 15 minutes lookahead horizon from the snapshot time for each evaluation and the fitness is calculated using Equation 4.8 for the 15 minute period following the snapshot time. During this period the aircraft were simulated according to their flight plan and based on new positioning information obtained from the snapshot. The previously generated requests are not taken into consideration for subsequent optimisations. For example, if during the optimisation which started at 8:15 aircraft DLH447 was given a request to climb from it's planned cruise altitude of 35,000ft to 37,000ft, then the snapshot at 8:20 would indicate that the aircraft is flying at or climbing towards 37,000ft. But during the optimisation starting at 8:20 the simulator will descend the aircraft back to 35,000ft as this is the altitude filed in the flight plan. These manoeuvres may result in high complexities. The effect of incorporating previously generated and executed requests will be investigated in a subsequent section.

During each snapshot-optimisation cycle we obtain four sets of complexity measurements:

- *Perfect model snapshot baseline complexity (C_{BA}):* The scenario is simulated using the perfect model to obtain a baseline complexity list for the lookahead period using the snapshot data. This simulation is conducted on the assumption that no further actions will interfere with the operation of the flights from their flight plans within the lookahead period. This complexity list represents the expected complexity in the real world. The perfect model snapshot baseline average deviation, f_{BA}, is obtained by applying Equation 4.8 to this list of complexity measurements.

- *Shadow simulator snapshot baseline complexity (C_{BS}):* The scenario is simulated using the shadow simulator to obtain another baseline complexity list for the lookahead period using the snapshot data. This simulation is conducted on the assumption that no further actions will interfere with the operation of the flights from their flight plans within the lookahead period. The shadow simulator snapshot baseline average deviation, f_{BS}, is obtained by applying Equation 4.8 to this list of complexity measurements.

- *Best fitness complexity (C_{RS}):* Once the optimisation is complete, the complexity from the individual with the best fitness, f_{RS}, is also obtained. This measurement shows the expected complexity as a result of the implementation of the generated request.

- *Perfect model best fitness complexity (C_{RA}):* The request generated by the best individual is used as part of another simulation using the perfect model. This list provides us with the actual expected effect of the generated request on the real-time environment for the lookahead period on the assumption that the request is the only action within this period that deviates the flights from their flight plan. This complexity list represents the expected effect of the request on the real world. The expected average deviation, f_{RA}, is obtained by applying Equation 4.8 to this list of complexity measurements.

All of these lists consists of complexity measurements taken at a frequency of ten times a minute on a rolling two minute window. The request from the individual with the best fitness is submitted for execution in the real-time environment if $f_{RS} < f_{BS}$, otherwise no requests will be executed from that iteration.

Differential evolution parameters

When using differential evolution (DE) there are no pre-established parameter settings which are suitable for solving all types of problems. Therefore investigating the effect of different parameters, such as the crossover ratios (CR), scaling factors (F) and population sizes, are required to assist in the selection of values suitable for the application.

Many different variants to the classic DE algorithm have been developed which incorporate different methods of selecting appropriate parameters for the problem at hand. Abbass *et al.* (2002) developed a self-adapted operator for multi objective optimisation problems where the CR was evolved simultaneously with other parameters and the F was generated from a Gaussian distribution. While Sarker *et al.* (2014) developed a method where the best performing combinations of parameters (CR, F and population size) were dynamically chosen during the course of a single run. Although these techniques may assist in obtaining suitable parameters during operations, their use is outside the scope of this study. These techniques may be investigated in the future as it would allow us to use dynamic parameters suited to the problem at hand. For the purpose of this study we will only focus on the classic DE algorithm where CR, F and population size are constant values throughout the run.

A large population size has a higher probability of finding a global optimum, but there is a slower convergence rate and requires more evaluations (Mallipeddi *et al.*, 2011). Smaller population sizes can speed up the convergence. The crossover ratio CR is the probability of mixing occurring between the trial and target vectors and is number between 0 and 1. A small CR may lead to no convergence as only a small number of parameters are changed in the trial vector. Meanwhile a large values for CR (approaching 1) may often speed up convergence (Gämperle *et al.*, 2002), but it may also lead to stagnation as most of the parameters in the trial vector are changed in each generation. Gämperle *et al.* (2002) also states that a good choice for CR is between 0.3 and 0.9. The scaling factor F is a number between 0 and 2. Typical values of F are between 0.4 and 0.95 (Rönkkönen *et al.*, 2005). Applications where a value greater than 1.2 is an absolute necessity are yet to be found and also values of F smaller than 0.4 are rarely useful (Mallipeddi *et al.*, 2011). A larger value of F may help increase the probability of not falling into a local optimum. Rönkkönen *et al.* (2005) recommends 0.9 as a starting value for F as this provides a good compromise between speed and probability of convergence. Storn and Price (1997) suggest using 0.5 as an initial choice for F while Gämperle *et al.* (2002) suggests using 0.6.

For our study we selected three values for CR and another three values for F. For CR we selected 0.3 and 0.9 as they are on the extreme ends of the range suggested by Gämperle *et al.* (2002). We also selected 0.6 as this is the mid point of the range. For F we selected 0.5, 0.6 and 0.9 as they fall into the range suggested by (Rönkkönen *et al.*, 2005) and are the values suggested for initial settings by Storn and Price (1997), Rönkkönen *et al.* (2005) and Gämperle *et al.* (2002). This gives us 9 combinations of CR and F values for experimentation. The number of individuals in the population were kept constant at 20.

During each optimisation cycle, the differential evolution was operated for 100 generations. If the 100 generations were completed before the time T_{OE} was reached in the real-time simulation or a stagnation point had been reached, then the differential evolution was started again with for same snapshot, but with a different seed for the random number generator. This process continued until the time T_{OE} was reached in the real-time visual simulation. At this time the request with the best fitness from all differential evolution runs for this snapshot was selected for implementation in the real-time simulation.

The input scenario was simulated 20 times for each of the 9 DE parameter combinations with each of the 5 simulators as the shadow simulator, giving us a total of 900 runs. The results from these runs are presented below.

Results

For each of the 900 runs we have 12 sets of results. Each set corresponds to each of the snapshots from the real-time simulation. This set includes the perfect model snapshot baseline complexity, the shadow simulator baseline complexity, statistics about the optimisation, the request which produced the individual with the best fitness during the optimisation and it's corresponding complexity and the complexity from using the perfect model to simulate using the snapshot data and this request. Using these four complexities from one snapshot, we can construct plots such as the one in Figure 4.7. This figure shows the baseline complexity based on the snapshot and the flight plan when simulated using the perfect model and the shadow simulator (blue and green, respectively). The figure also show the effect on the complexity as a result of including a request generated from the optimisation. The expected effect when simulated using the shadow simulator can be seen in black and this line is the basis for evaluating fitness during the optimisation. The actual expected effect in the real-time environment can be seen in purple and this line is the basis for our evaluation for this experiment. The figure also indicates the target level of complexity and the execution time for the request. The same information can also be seen in Figure 4.8 where the complexities are plotted as cumulative sums.

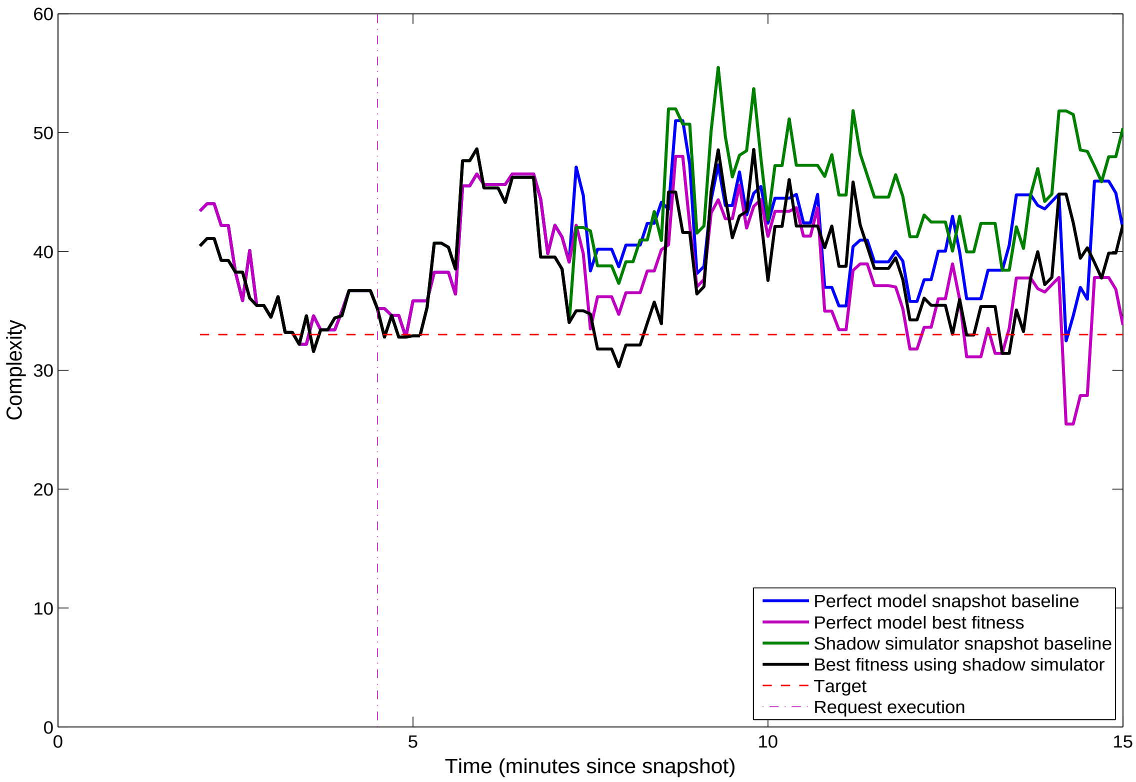


Figure 4.7: Comparison of the snapshot baseline airspace complexity and the expected airspace complexity as a result of the execution of a request

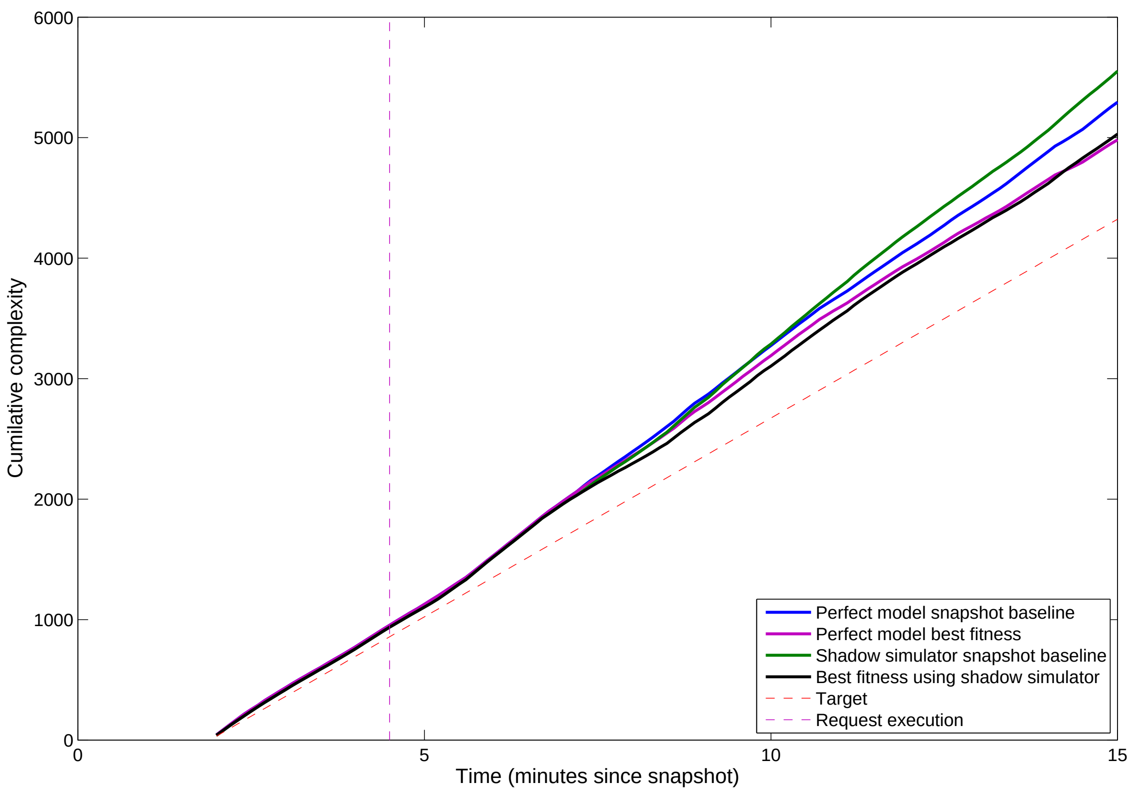


Figure 4.8: Comparison of the cumulative snapshot baseline airspace complexity and the expected airspace complexity as a result of the execution of a request

Using the information in these figures we can obtain the average deviation values f_{BA}, f_{BS}, f_{RS} and f_{RA}. These average deviation values can be used to construct a table such as the one in Table 4.2. This table shows the average deviation values for each snapshot from a single run using the parameter settings of $CR = 0.9$ and $F = 0.6$ for a lookahead period of 15 minutes and using BS2 as the shadow simulator. In this table we can see five values for average deviations corresponding to each snapshot time. The first value, baseline, is the average deviation of the complexity from the target for the 15 minute lookahead period (starting from the snapshot time) for the input scenario with the aircraft simulated according to the flight plan without any deviations. We can see that the average deviation for the snapshot baseline complexity using the perfect model, f_{BA}, does not always match with that of the baseline. This is due to the flow on effects of the request being implemented. Although the outcome of the optimisation results in most cases leading to expected deviations which are more favourable than the snapshot baselines for the lookahead period, we do not get a picture of the effect of the request beyond the lookahead period. As we can see from this table, in some cases the effect of the request has a flow on effect into the following snapshot intervals where f_{BA} ends up being greater than the original baseline for the corresponding time. But we can see that in most cases the execution of the request is expected to result in a positive effect. From this table we can also see the effect of the difference in fidelity between the perfect model and BS2 in the complexity measurements as there is some deviation between f_{BA} and f_{RA} from the perfect model and the corresponding values f_{BS} and f_{RS} from BS2. A plot of the complexity measurements leading to the values for the snapshot at time 0 minutes (sent immediately at the start of the scenario) can be seen in Figure 4.7 and 4.8. Once we have obtained the data for each of the 900 runs and generated similar summary tables for each run we are able to aggregate this data for the different settings.

The number of evaluations that were made within the 2 minute period for each optimisation when using the different shadow simulators is one metric that can be obtained from this aggregated data and can be seen in Figure 4.9. For this figure,

Table 4.2: Average deviations from target level of complexity using the perfect model (f_{BA}) and the shadow simulator (f_{BS}) to obtain a snapshot baseline and the best fitness using the shadow simulator (f_{RS}) and the perfect model (f_{RA}) for a single run

CR = 0.9, F = 0.6, Lookahead period = 15 minutes, Shadow simulator = BS2						
Snapshot time (min since start of scenario)	**Average deviation**					**Evaluations**
	Baseline	f_{BA}	f_{BS}	f_{RS}	f_{RA}	
0	7.02	7.02	7.81	4.71	5.13	1020
5	6.79	7.45	9.43	5.63	5.61	1040
10	8.93	9.24	10.9	8.28	8.27	1200
15	8.63	7.46	8.85	6.83	7.51	1440
20	8.72	8.47	9.59	5.53	8.45	1740
25	6.36	5.5	4.34	3.38	6.57	2240
30	4.42	4.05	5.03	4.41	4.45	2760
35	4.60	5.00	4.61	3.88	4.11	3460
40	9.83	11.2	9.43	6.86	11.2	4940
45	16.86	17.88	15.93	14.9	16.91	8360
50	21.16	23.72	24.47	23.25	22.81	13600
55	24.09	20.38	20.69	19.84	20.38	40860

the 900 runs were grouped based on the shadow simulator used for that run. The optimisation statistics for each corresponding snapshot were also grouped together, irrespective of their DE parameters. From these groups we can obtain the average number of evaluations made using the optimisation run when using each of the simulators for each snapshot time. From these aggregated statistics we also obtain the number of agents that were simulated during these evaluations. The number of agents simulated during each optimisation in the different run may not be the same as previous request may have lead to aircraft entering or exiting the sector at different times. The average number of agents simulated from each snapshot can also be seen in the figure. Details for the optimisations which took place using the snapshot data from the 55th minute of the scenario are not shown in this figure as the low number of agents (6 on average) in this snapshot lead to significantly higher number of evaluations than those shown in the figure. From this figure we can conclude that the simulators with the highest fidelity conducted the least number of evaluations at similar stages of the run. As expected, BS1 and BS2, the simulators with the lowest

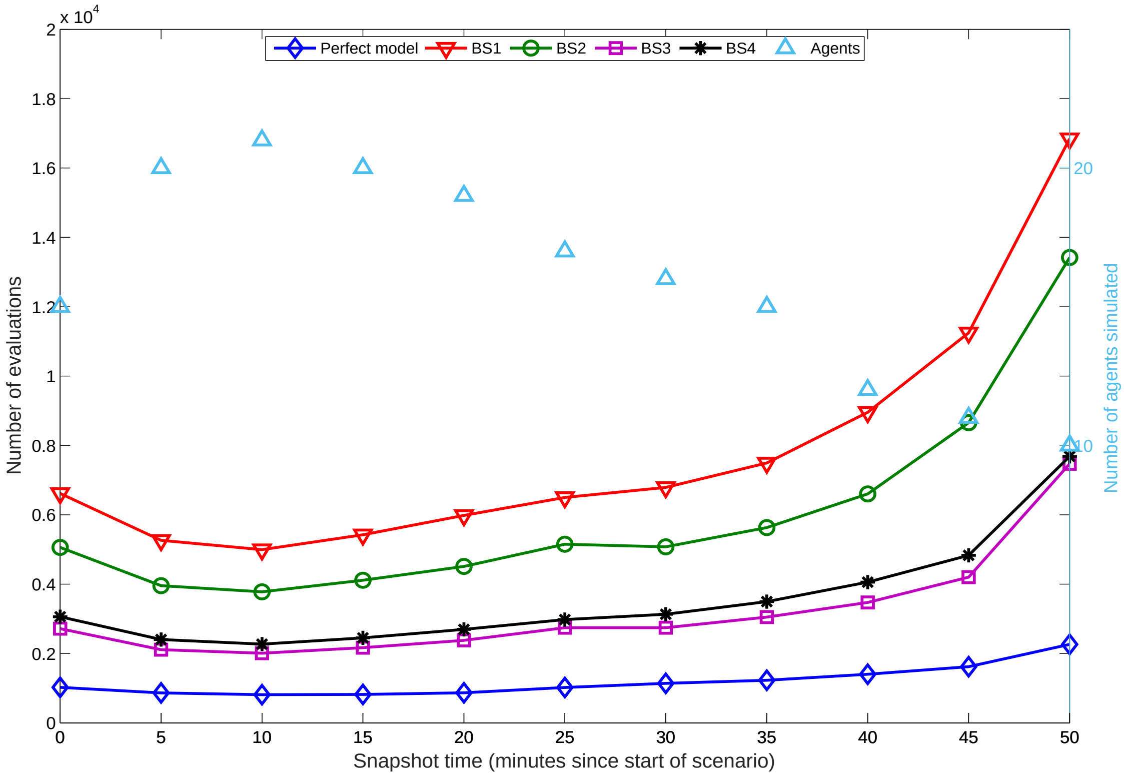


Figure 4.9: Average number of evaluations completed with each simulator for each snapshot

fidelity, conducted the most evaluations. We can also see from this figure that BS1 and BS4 completed more evaluations than BS2 and BS3 respectively. This confirms the fact that the calculations related to the geodetic coordinate system has a bigger effect on the simulation run time than the using the BADA lookup tables to obtain flight data. It is worth noting the difference in the ratios of evaluations conducted when using BS1 and BS2 and the ratio of evaluations conducted when using BS3 and BS4. On average, when using BS1 as the shadow simulator there were 30% more evaluations completed than when using BS2. However, when we use BS4 as the shadow simulator we are able to complete only 12% more evaluations on average when compared to BS3. It is also worth noting that the ratio between the number of number evaluation completed and the number of agents is not constant, but however the ratio scales with the number of agents.

We can also group the results from these runs based on their DE parameter setting combinations in selecting an appropriate combination of settings for our problem. To do this we grouped each of the runs based on their CR, F and shadow simulator combination. Then we combined their the four average deviation values for each optimisation for each snapshot time (see Table 4.2), and produced a statistical summary of these values for each combination. This provided a statistic based on the 20 runs for each combination with 12 data points for each run. The averages and standard deviations of the f_{RA} values for each combination can be seen in Table 4.3. This value represents the expected effect of the execution of the request in the real world. From this table we can see that with the exception of one or two combinations for each shadow simulator, there is little difference in the averages for each combination for each simulator. A t-test was conduced for each combination for each simulator against the combination with the lowest average for that simulator. It was found that there are no combinations where the difference of averages are convincing enough to conclude that there is a significant difference. Therefore we will decide on the best parameter combination simply by selecting the combination which provides the lowest average value. From Table 4.3 we find that the value $CR = 0.9$ was the value of CR which provided the lowest or equal lowest average f_{RA} value for all of the shadow simulators. Among these combinations we also find that three out of the five shadow simulators had their lowest (or equal lowest) average f_{RA} value when using the $CR = 0.9$ and $F = 0.6$ combination. Therefore we will select $CR = 0.9$ and $F = 0.6$ as the DE parameter combination for all further experiments.

4.5 Experiment - effect of lookahead time and airspace complexity target

Once we have selected the parameters for the optimisation component we can explore the effect of other system parameters. The setup for this experiment is identical to that of the previous experiment in all aspect expect instead of varying the differential evolution parameters, the second experiment involves varying the

Table 4.3: Average deviations from target level of complexity for different DE parameters using the four shadow simulators (BS1 to BS4) and the perfect model (Perfect)

CR	**F = 0.5**				
	BS1	**BS2**	**BS3**	**BS4**	**Perfect**
0.3	9.52 ± 5.09	9.92 ± 6.1	9.84 ± 5.9	9.22 ± 5.62	6.92 ± 5.03
0.6	9.49 ± 5.12	10.01 ± 6.06	10.26 ± 5.95	9.18 ± 5.7	6.9 ± 5.00
0.9	9.48 ± 5.12	9.92 ± 6.14	9.86 ± 5.83	9.16 ± 5.68	7.05 ± 5.12

CR	**F = 0.6**				
	BS1	**BS2**	**BS3**	**BS4**	**Perfect**
0.3	9.5 ± 5.1	9.95 ± 6.02	9.85 ± 5.93	9.22 ± 5.66	7.16 ± 5.54
0.6	9.49 ± 5.1	9.96 ± 6.13	9.83 ± 5.82	9.22 ± 5.62	7.29 ± 5.53
0.9	9.44 ± 5.13	9.92 ± 6.16	9.82 ± 5.8	9.23 ± 5.63	6.81 ± 4.75

CR	**F = 0.9**				
	BS1	**BS2**	**BS3**	**BS4**	**Perfect**
0.3	9.48 ± 5.1	9.9 ± 6.15	9.89 ± 6	9.27 ±,5.62	7.29 ± 5.49
0.6	9.45 ± 5.12	9.94 ± 6.08	9.83 ± 5.85	9.2 ± 5.64	6.9 ± 4.99
0.9	9.43 ± 5.13	9.93 ± 6.18	9.82 ± 5.89	9.22 ± 5.66	6.98 ± 5.18

parameters for the lookahead horizon and the airspace complexity target level.

Scenario and airspace complexity target

The scenario used for this experiment was the same as the one used for the previous experiment. The target for the airspace complexity was however set as the 25th, 50th and 75th percentiles of the baseline complexity when using the perfect model without the execution of any requests and the complexity was measured at a frequency of ten times a minute based on the previous two minutes of activity. A plot of the baseline airspace complexity can be seen in Figure 4.10 along with the 25th, 50th and 75th percentile complexity target levels.

Snapshot interval and lookahead time

The 5 minute snapshot interval from the real-time simulator was also maintained for this experiment. The lookahead horizon was, however, varied. Values of 15, 30 and 60 minutes were used for the lookahead period. In this experiment the aircraft also continued to follow their flight plans or proceed to correct their intentions in order to

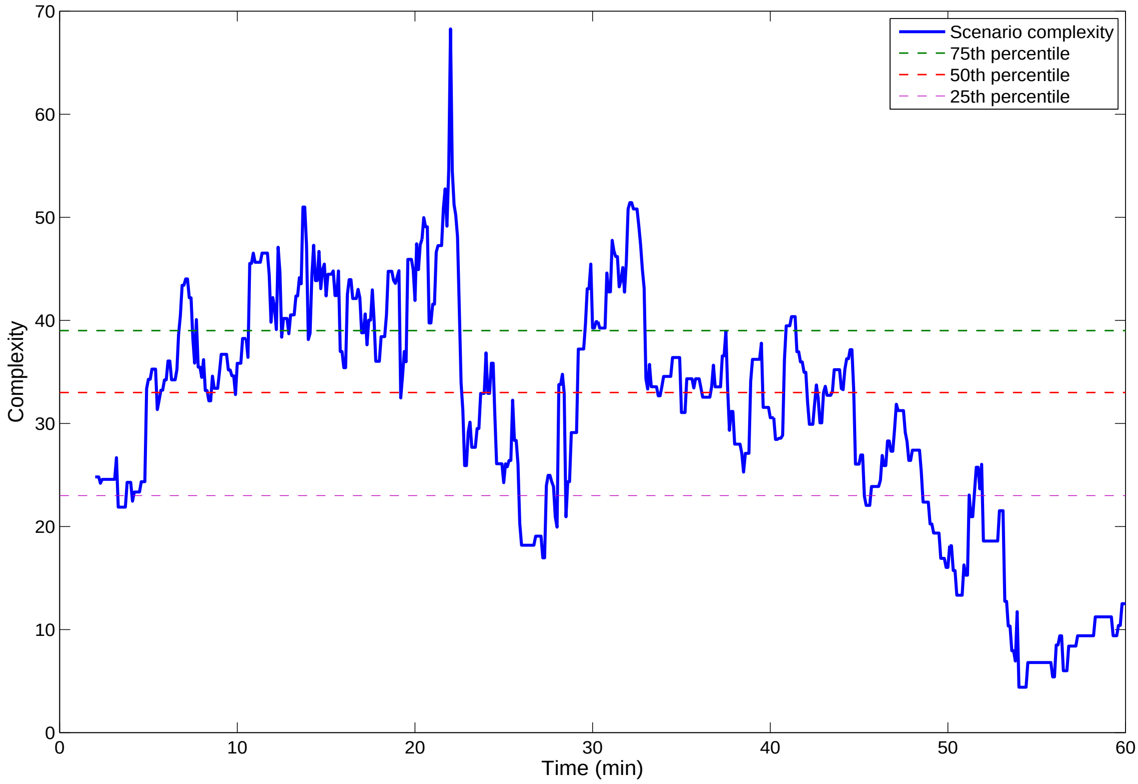


Figure 4.10: Airspace complexity for the input scenario measured using the perfect model

follow the flight plans irrespective of previously generated or executed requests and their position from the snapshot. The four complexity measurements, the perfect model snapshot baseline complexity, shadow simulator baseline complexity, best fitness complexity and perfect model best fitness complexity; were also recorded for each snapshot for each run.

Differential evolution parameters

As per the results of the previous experiment, the DE parameters CR and F were fixed at 0.9 and 0.6 respectively for all runs. The population size was also fixed at 20 individuals.

The input scenario was simulated 20 times for each lookahead horizon and complexity target levels with each of the 5 models (BS1, BS2, BS3, BS4 and the perfect model) as the shadow simulator. These combinations also resulted in a total of 900 runs.

Results

During this experiment we again collected the four average deviations from the airspace complexity target level measurements (perfect model snapshot baseline complexity, shadow simulator snapshot baseline complexity, best fitness complexity and perfect model best fitness complexity) for each snapshot during each run. The number of evaluations completed during for each snapshot was also recorded. As the latest possible end time for the lookahead period is the end of the scenario time, there will be a point in the scenario where the lookahead period will be less than the lookahead horizon. This means that there will also be a point in the scenario where the lookahead period for lookahead horizons of 30 and 60 minutes become the same. For our 60 minute scenario this occurs 30 minutes after the start of the scenario.

We find that the ratio of the average number of evaluations completed for each snapshot is similar throughout the scenario, irrespective of the snapshot time, in relation to the average number of evaluations completed using the perfect model for the corresponding snapshot time. There is less than 10% variation in this ratio throughout the scenario. The average of these ratios for each of the shadow simulators can be seen in Table 4.4. We also found that the ratio between the average number of evaluations conducted with a 15 minute lookahead horizon to the average number of evaluations conducted with a 30 minute lookahead horizon using the same shadow simulators is similar across all snapshot times, irrespective of the shadow simulator used. This is also the same when a 60 minute lookahead horizon is used. Table 4.5 shows the ratio between the average number of evaluations completed when using a 15 minute lookahead horizon to when using a 30 minute lookahead horizon and a 60 minute horizon for the snapshot at 0 minutes (ie. immediately after the start of the scenario) when using the same shadow simulator. This particular time was chosen as it is the only snapshot for which all three lookahead horizons can be fully fit into the 60 minute scenario. As the scenario time nears 60 minutes, both ratios begin to approach 1 as the lookahead period becomes similar in length, if not identical. It is interesting to note that the ratio does not have a one-to-one scaling with the lookahead horizon. This indicates that this ratio is also influenced by the number of agents which are required to be simulated during the lookahead horizon.

Table 4.4: Ratio of average evaluations completed when using each shadow simulator in relation to the perfect model (the real world) for each lookahead period (LKA)

Shadow simulator	**Ratio of evaluations relative to the perfect model (Perfect)**		
	LKA = 15	**LKA = 30**	**LKA = 60**
Perfect	1	1	1
BS1	6.47	6.49	6.27
BS2	4.95	5.01	4.81
BS3	2.66	2.58	2.48
BS4	2.99	2.99	2.90

Table 4.5: Ratio of evaluations in relation to lookahead horizion of 15 minutes (LKA = 15) for the same shadow simulator

Shadow simulator	**Ratio of evaluations realtive to LKA = 15**	
	LKA = 30	**LKA = 60**
Perfect	0.38	0.22
BS1	0.38	0.21
BS2	0.39	0.21
BS3	0.37	0.20
BS4	0.38	0.21

Next we group the results from these runs based on their lookahead horizon, airspace complexity target level and shadow simulator combination. We then find the average and standard deviation of the f_{RA} values (the expected average deviation from the target airspace complexity level when executing a request from the optimisation component in the real world) for each snapshot time for each run. This information is presented in Table 4.6 for each group. We can see from this table that for most shadow simulator and lookahead horizon combinations, the runs with a target level of airspace complexity set at the 25th percentile were able to achieve better average f_{RA} values than their corresponding runs with 50th or 75th percentile target levels. This indicates that it is easier, for this particular scenario, to produce requests which reduce the complexity than it is to increase the complexity of the scenario. A t-test was conducted for each lookahead period and target level combination for the simulators BS1 to BS4. The simulator with the lowest average for each combination was tested against each of the other three simulators for that combination. The instances where we can conclude that there are significant differences from the simulator with the lowest average are indicated in boldface. From Table 4.6 we see again that the results are influenced by the shadow simulator that is being used to generate the requests, but there is no clear winner in each case nor are there any patterns which allow us to pick one simulator over another in any given lookahead period or target level setting.

Table 4.6: Average deviations from target level of complexity for different lookahead period and target level parameters using the four shadow simulators (BS1 to BS4) and perfect model (Perfect)

Target (%ile)	**LKA = 15**				
	BS1	**BS2**	**BS3**	**BS4**	**Perfect**
25th	9.76 ± 2.40	9.79 ± 3.43	9.68 ± 3.56	9.79 ± 2.58	6.07 ± 2.70
50th	9.44 ± 5.13	9.92 ± 6.16	9.81 ± 5.79	9.23 ± 5.63	6.81 ± 4.75
75th	11.60 ± 8.48	11.85 ± 7.78	12.00 ± 9.05	11.85 ± 8.63	7.84 ± 5.71
Target (%ile)	**LKA = 30**				
	BS1	**BS2**	**BS3**	**BS4**	**Perfect**
25th	9.63 ± 2.97	9.37 ± 2.30	9.77 ± 3.20	**10.16 ± 3.17**	6.24 ± 2.51
50th	10.80 ± 4.72	11.17 ± 5.09	11.11 ± 5.35	10.74 ± 4.87	8.02 ± 4.16
75th	13.74 ± 7.76	14.13 ± 7.46	14.09 ± 7.87	13.70 ± 7.67	10.32 ± 5.80
Target (%ile)	**LKA = 60**				
	BS1	**BS2**	**BS3**	**BS4**	**Perfect**
25th	8.62 ± 1.91	**9.21 ± 1.77**	**9.58 ± 2.06**	**9.42 ± 2.15**	6.49 ± 2.22
50th	11.74 ± 3.46	11.17 ± 5.09	12.57 ± 4.40	12.04 ± 3.96	8.49 ± 2.24
75th	16.38 ± 5.47	14.13 ± 7.46	16.28 ± 5.92	16.26 ± 6.02	10.41 ± 3.24

If we take a look at the differences between the complexities measured by the perfect model and the other simulators we can start to see why this may be the case. To do this we will find the ratio between the complexity as measured by the perfect model and the corresponding complexity as measured by the other simulators for the same situation. We find the ratio between the snapshot baseline complexities for each snapshot from each run using Equation 4.9. In order to use this equation we sum all of the complexity measurements from the shadow simulator snapshot baseline complexity list ($\sum C_{BS}$) and divide this value by the sum of the complexity measurements from the corresponding perfect model snapshot baseline complexity ($\sum C_{BA}$). This ratio is found for each snapshot for every run. If this ratio is equal to 1, then the complexity estimate the simulator is in perfect agreement with the real world. If this ratio is greater than 1, then the simulator has overestimated the complexity in relation to the real world for the lookahead period while a ratio lower than 0 will mean that the simulator has underestimated the complexity for the lookahead period. Similarly we can also find the ratios for the expected effect on the complexity of the execution of the generated requests using Equation 4.10. For this equation we sum the complexity measurements obtained from the individual with the best fitness from the optimisation ($\sum C_{RS}$) and divide this value by the sum of the complexity measurements expected in the real world based on the execution of the request generated by this individual ($\sum C_{RA}$). Once we have obtained the ratios for each snapshot for each run we can group them by their lookahead period and the simulator that was used to make the estimates.

$$R_B = \frac{\sum C_{BS}}{\sum C_{BA}} \tag{4.9}$$

$$R_R = \frac{\sum C_{RS}}{\sum C_{RA}} \tag{4.10}$$

A histogram of these ratios for BS1 can be seen in Figure 4.15, 4.16 and 4.17 and for BS2 in Figure 4.18, for BS3 in Figure 4.19 and for BS4 in Figure 4.20. The average ratios for each of the groups can be seen in Table 4.7. From this table and the these figures we can see that for all simulators the ratio R_B is generally greater than 1. This suggests that the simulators are overestimating the complexity when compared to the expected real world complexity. We can also see that the ratio R_R is less than 1, which suggests that the estimated complexity as a result of the execution of the request is underestimated when compared to the expected complexity in the real world as a result of the execution of the same request. This leads to an overestimation of the expected effect that the request will have on the scenario's complexity. This overestimation leads to the selection and execution of requests which may not necessarily influence the complexity of the scenario as significantly as expected and in some cases it may not lead to a positive effect on the complexity in terms of the target level. From these figures we can also see that when the lookahead horizon is set at 30 minutes or 60 minutes, the ratios are closer to 1 than when the lookahead horizon is set at 15 minutes. This suggests that minor differences in the complexity estimates become less significant when simulating over longer periods.

Table 4.7: Average ratios of airspace complexity estimates between the perfect model (the real world) and the simulators BS1 to BS4. A value close to 1 indicates a better match.

Lookahead	**BS1**		**BS2**		**BS3**		**BS4**	
period	R_B	R_R	R_B	R_R	R_B	R_R	R_B	R_R
LKA = 15	1.22	0.95	1.07	0.89	1.06	0.89	1.20	0.95
LKA = 30	1.13	0.98	1.06	0.93	1.02	0.91	1.10	0.97
LKA = 60	1.09	1.03	1.02	0.97	1.00	0.95	1.07	1.00

4.6 Experiment - keeping a record of previously generated requests

In the third experiment we, once again, simulated the input scenario with the three different settings for the lookahead horizon and the three different target airspace complexity levels. But in this experiment we also retained a record of all requests which were generated in each run and these requests were used during the simulations to establish the snapshot baseline complexities and for the evaluations during the optimisation phases for that particular run. This means that if a request was generated from the optimisation starting as a result of the snapshot at 8:15 for the aircraft DLH447 to climb from it's cruise altitude of 35,000ft to 37,000ft, then during simulations relating to the following snapshot (at 8:20 in the case of this example) we will see the aircraft continue to climb to 37,000ft or maintain the altitude of 37,000ft if it has already reached the altitude. If the request is not scheduled to be executed before the next snapshot time then the request is also passed as part of the snapshot and will be executed as part of the simulations relating to this snapshot.

For this experiment we also used the three lookahead horizon settings from the previous experiment of 15, 30 and 60 minutes and also used the three airspace complexity target levels of the 25th, 50th and 75th percentile of the baseline scenario. We also conducted 20 runs with each setting combination. Once again we find the average and standard deviations of the f_{RA} values (the expected average deviation from the target airspace complexity level when executing a request from the optimisation component in the real world) for each snapshot time for each run and can be seen in Table 4.8. If we compare these results to those from Table 4.6 where the execution of previously generated requests were not known, we can see some of the differences caused by knowing about these requests during the optimisation. We can see that for BS2, BS3 and the perfect model there is an improvement in the average f_{RA} values when the target level is set at the 25th percentile across all three lookahead horizons. In most other combinations we see that average is actually higher than in the previous experiment with higher standard deviations.

In the previous experiment we established that the shadow simulator will generally overestimate the snapshot baseline complexity when compared to the real world and will usually underestimate the complexity when executing a request. Although we can see in Table 4.9 that there is a drop in the average R_B ratio (which is the ratio between the snapshot baseline complexity estimated by the simulators to that of the estimated real world complexity using Equation 4.9), this does not necessarily indicate that there is a lower range of underestimation in this experiment. As can be seen from the histogram in Figure 4.11 for BS1 and a lookahead horizon of 15 minutes, this is in fact caused by a bigger range of instances with underestimation. This is possibly caused by the execution of requests during the snapshot baseline simulation.

While we have seen that it is possible to adjust the airspace complexity, and therefore the ATC's workload, in real-time we cannot always reach a specified target level of airspace complexity. There are several reasons for this, but chief among these

Table 4.8: Average deviations from target level of complexity for different lookahead period and target level parameters using the four shadow simulators (BS1 to BS4) and the perfect model (Perfect) and keeping a request archive

Target (%ile)	**LKA = 15**				
	BS1	**BS2**	**BS3**	**BS4**	**Perfect**
25th	10.45 ± 3.43	9.28 ± 2.11	8.72 ± 2.59	10.45 ± 3.37	4.82 ± 1.83
50th	10.11 ± 6.43	9.51 ± 6.91	11.39 ± 5.40	9.79 ± 4.75	7.26 ± 5.51
75th	11.89 ± 8.62	12.44 ± 8.84	12.23 ± 8.77	11.80 ± 8.57	8.31 ± 6.62
Target (%ile)	**LKA = 30**				
	BS1	**BS2**	**BS3**	**BS4**	**Perfect**
25th	9.12 ± 2.84	8.51 ± 1.89	9.59 ± 3.08	9.90 ± 3.23	5.70 ± 2.32
50th	10.86 ± 4.65	11.42 ± 5.58	10.93 ± 5.40	10.55 ± 4.75	8.34 ± 4.47
75th	13.78 ± 7.88	14.66 ± 8.08	14.37 ± 7.95	14.47 ± 7.87	10.07 ± 5.72
Target (%ile)	**LKA = 60**				
	BS1	**BS2**	**BS3**	**BS4**	**Perfect**
25th	9.14 ± 1.93	8.66 ± 1.30	9.27 ± 2.24	9.11 ± 2.33	5.95 ± 2.34
50th	10.86 ± 4.65	12.38 ± 4.04	12.23 ± 4.24	12.00 ± 3.94	8.17 ± 2.76
75th	13.78 ± 7.88	16.64 ± 5.82	16.18 ± 6.04	16.39 ± 5.93	10.05 ± 3.09

Table 4.9: Average ratios of airspace complexity estimates between the perfect model (the real world) and the simulators BS1 to BS4

Lookahead period	**BS1**		**BS2**		**BS3**		**BS4**	
	R_B	R_R	R_B	R_R	R_B	R_R	R_B	R_R
LKA = 15	1.06	0.87	1.18	0.99	1.07	0.91	1.08	0.92
LKA = 30	1.05	0.95	1.03	0.90	1.03	0.92	1.06	0.97
LKA = 60	1.07	1.01	0.99	0.93	1.02	0.97	1.09	1.05

reasons is the input scenario. According to the scenario each aircraft must travel from A to B, which cannot be altered. As a result certain events must happen throughout the course of the scenario that cannot be changed. This means that no matter how much optimisation we conduct there will be a minimum level of complexity for the scenario at any given time which must be maintained. But, as demonstrated by using the perfect model as the shadow simulator, we see that there is still some room for improvement in the results obtained when using the other simulators as the shadow simulator.

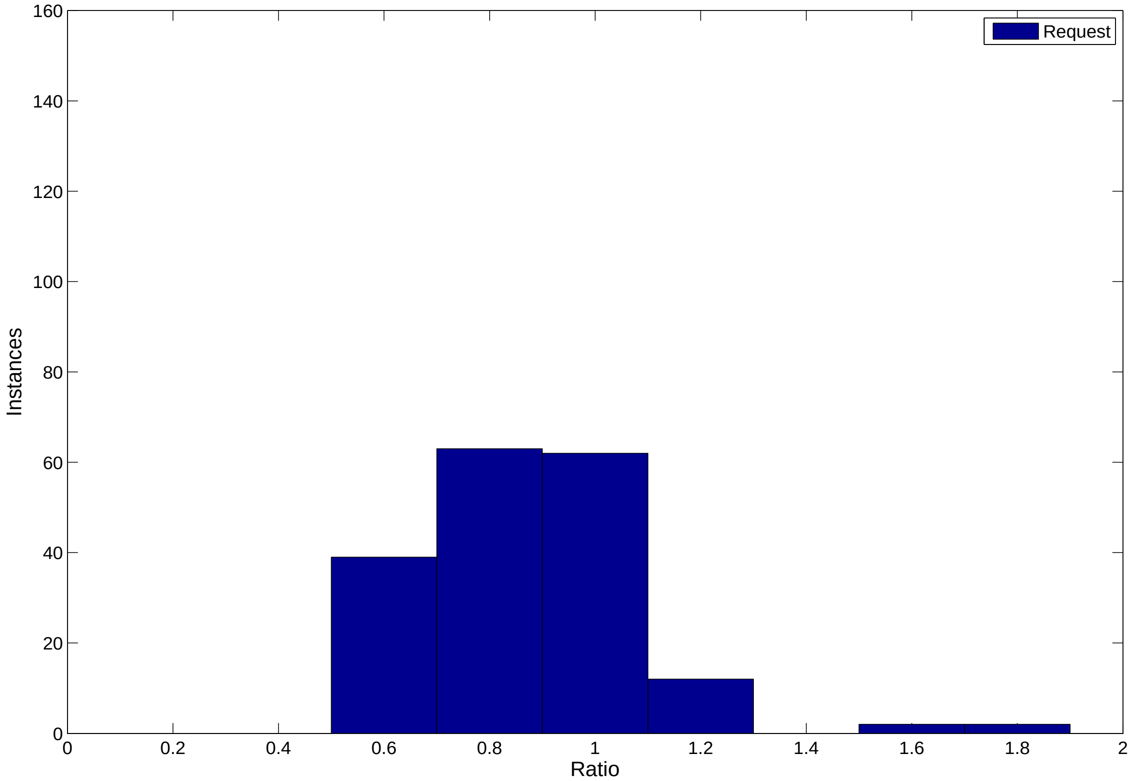
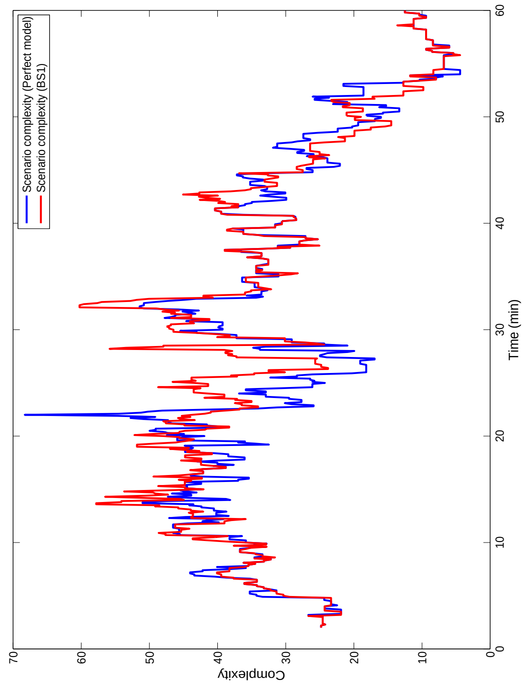


Figure 4.11: Comparison of the ratios of the snapshot baseline complexity measured by the perfect model and BS1

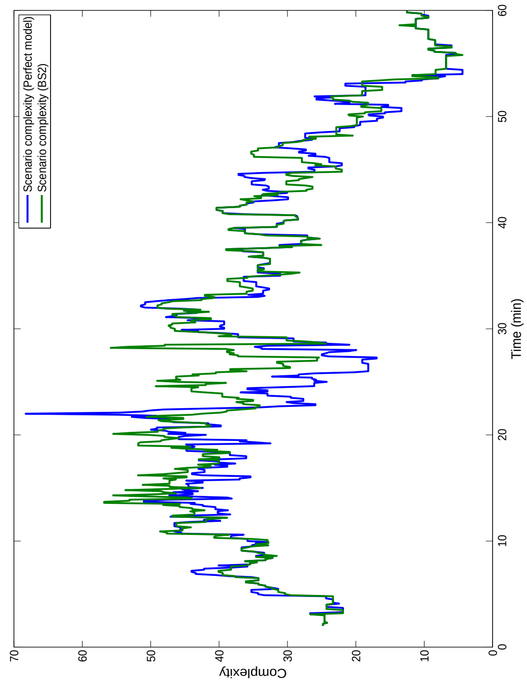
4.7 Summary

In this chapter we described the design of a real-time airspace complexity adjustment system using a multi-objective optimisation approach and also demonstrated its effectiveness. With the use of this system we aim to adjust the expected future workload for air traffic controllers in real-time. The airspace complexity was adjusted by generating requests, or actions, which led to aircraft deviating from their initial flight plans upon the execution of the requests. The actions included simple manoeuvres which the ATCs can request the pilots to undertake or the pilot can request to undertake. Examples of these manoeuvres include climbing or descending to a different flight level and skipping waypoints. The requests and the time for execution of these requests were generated using an optimisation system. A goal programming approach was used as the objective function and was used to steer the complexity towards a predefined target level using differential evolution for optimisation. This resulted in a multi-objective optimisation problem as we were required to alter various aspects of the air traffic environment to alter the airspace complexity. Five different simulators, with different levels of fidelity, were used to obtain the objective function for each request and execution time implementation. The optimisation was conducted in real-time periodically using current air traffic data with the aim of influencing the airspace complexity in the short to mid term future.

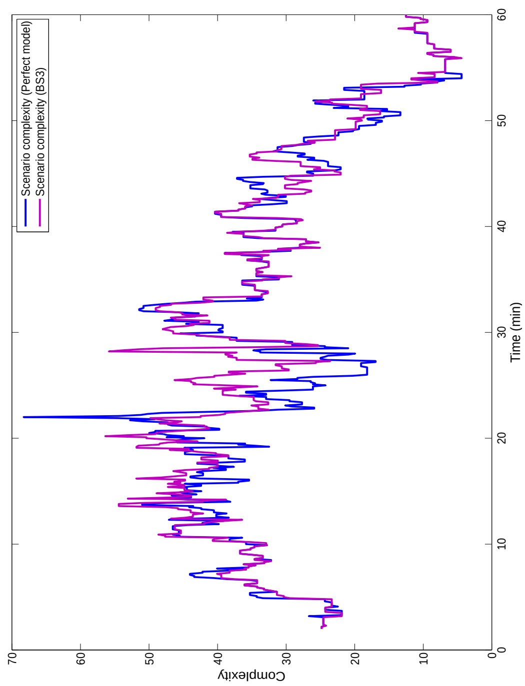
This system was used in several different experiments to demonstrate its effectiveness. It was found that when using BS1 as the shadow simulator we are able to achieve the highest number of evaluations in a fixed time frame for the optimisation component, while when using the perfect model we were able to achieve the fewest number of evaluations in the same time frame. The number of evaluations completed by each simulator was relative to it's fidelity and computational complexity. It was found that when using the perfect model as the simulator we were able to obtain the best results under all configurations we tested. There were no clear winners amongst the other four lower fidelity simulators, although these simulators were also able to successfully adjust the airspace complexity towards the target level. When comparing the complexities estimated by the four lower fidelity simulators, we found that BS2 and BS3 were generally able to provide a more accurate estimate of the complexity relative to the perfect model (the real world), which is assumed to an accurate representation of the real world, when compared to BS1 and BS4.



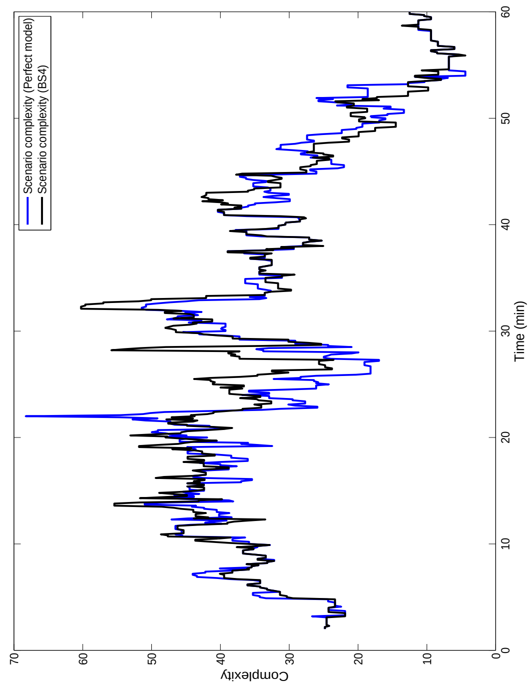
(a) Comparison of the baseline airspace complexity as measured by the perfect model and BS1



(b) Comparison of the baseline airspace complexity as measured by the perfect model and BS2

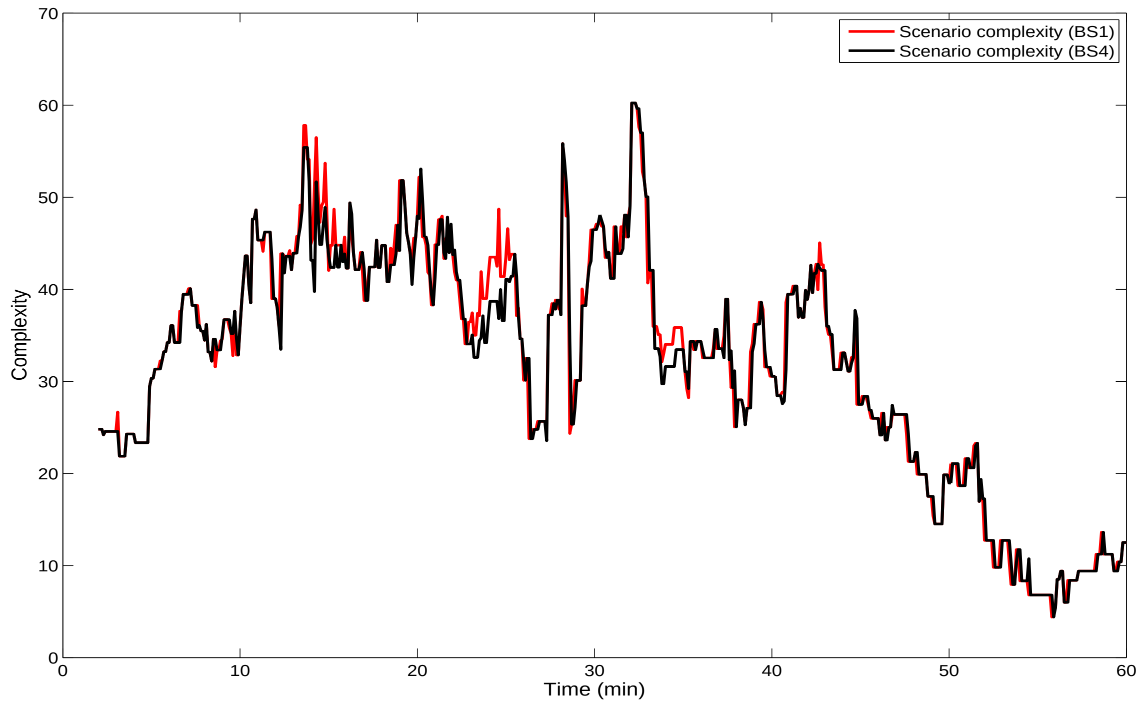


(c) Comparison of the baseline airspace complexity as measured by the perfect model and BS3

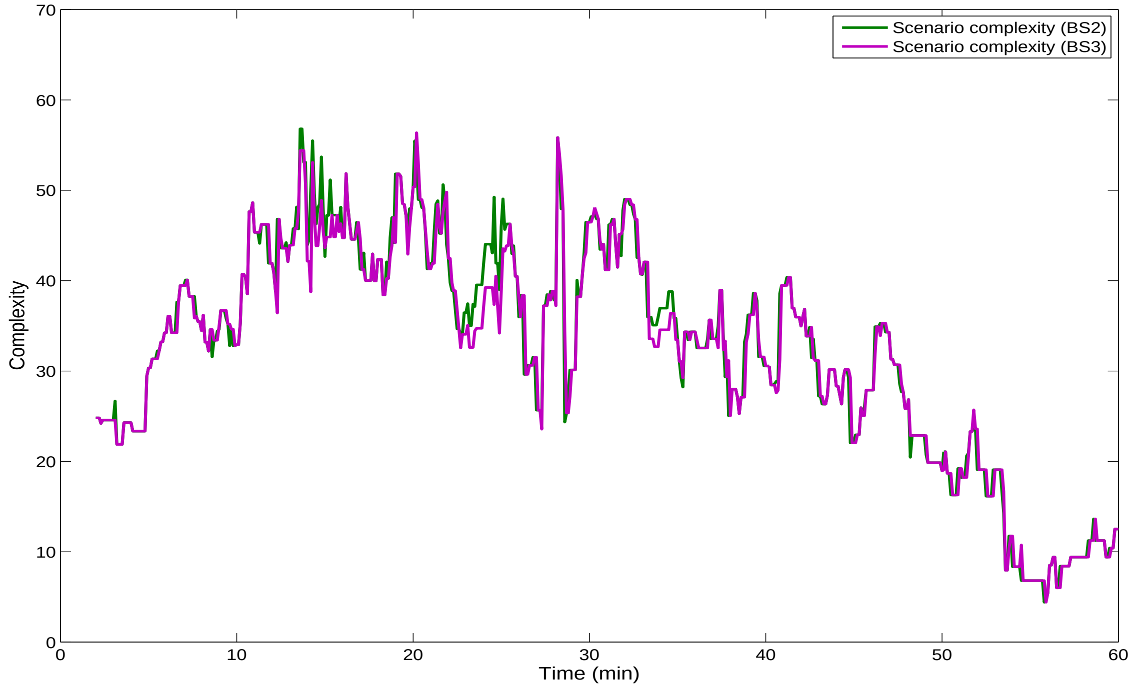


(d) Comparison of the baseline airspace complexity as measured by the perfect model and BS4

Figure 4.12: Comparison of the baseline airspace complexity as measured by the perfect model and the four Basic Simulators



(a) Comparison of the baseline airspace complexity as measured by BS1 and BS4



(b) Comparison of the baseline airspace complexity as measured by BS2 and BS3

Figure 4.13: Comparison of the baseline airspace complexity as measured by the four Basic Simulators

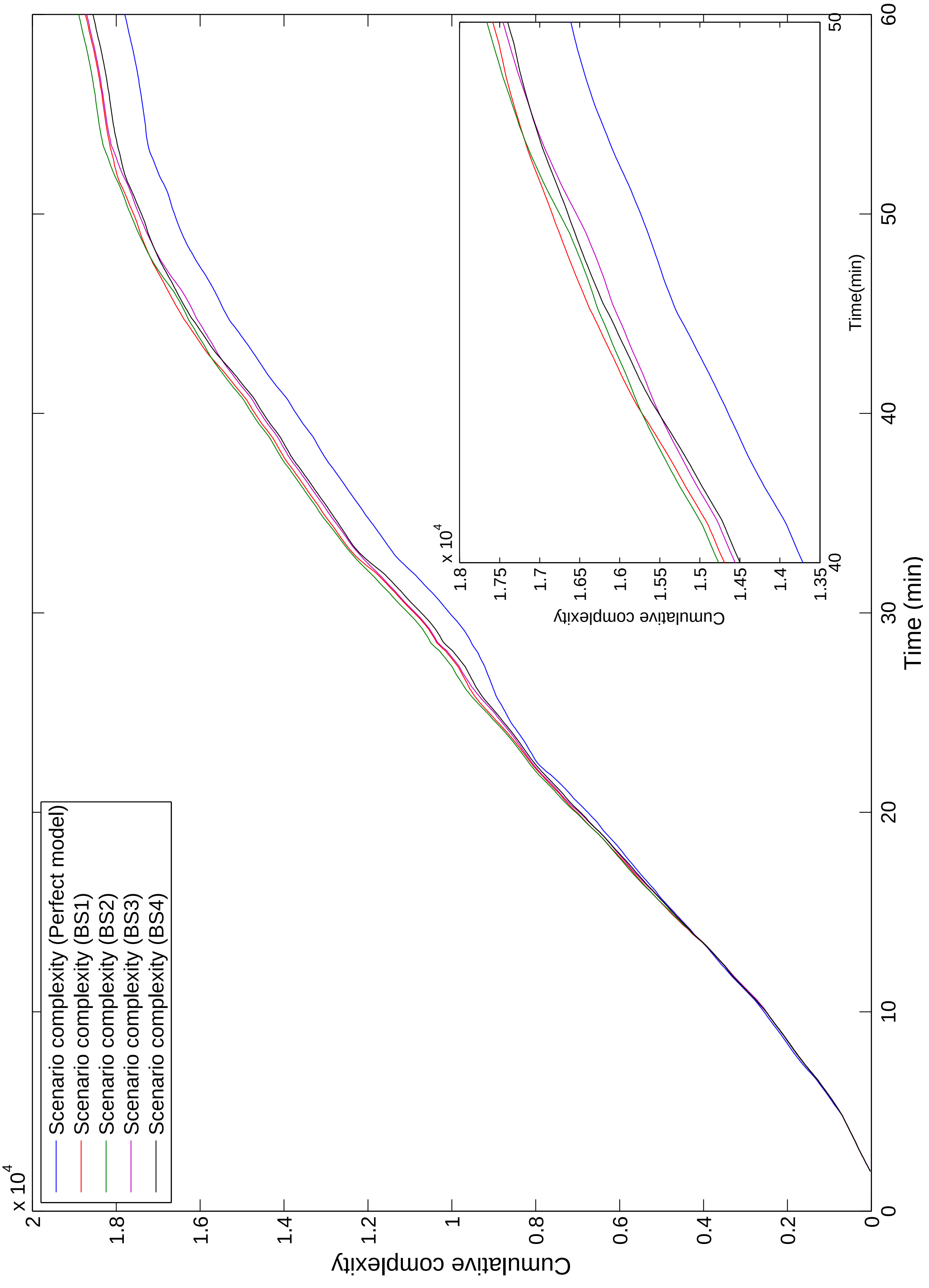
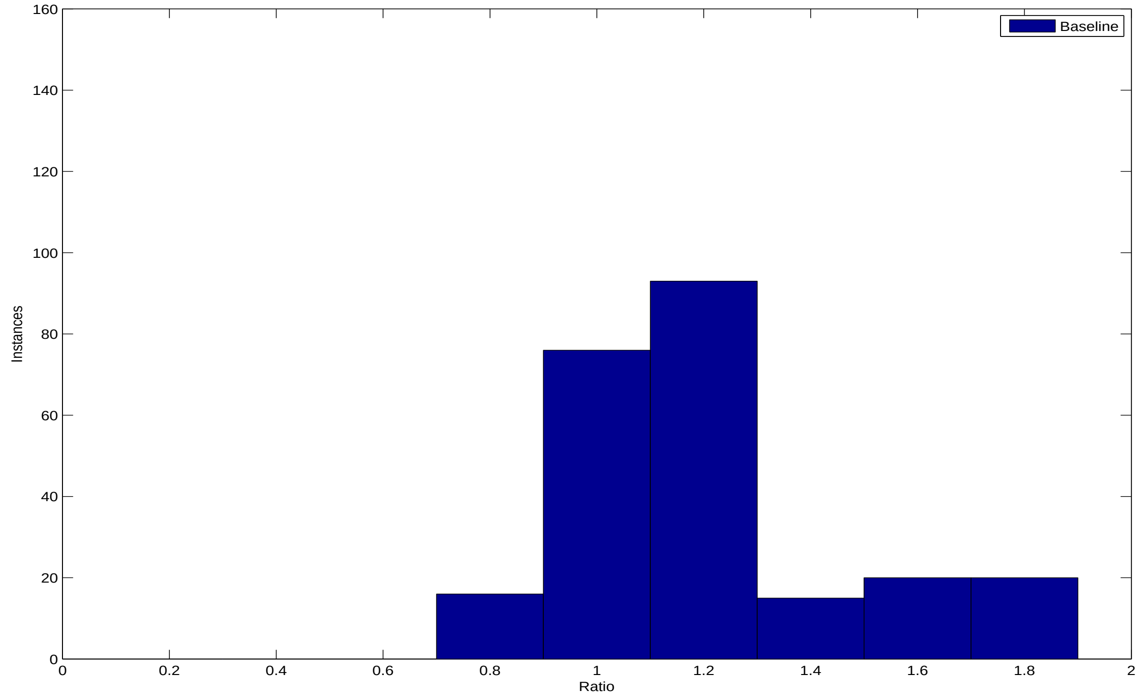
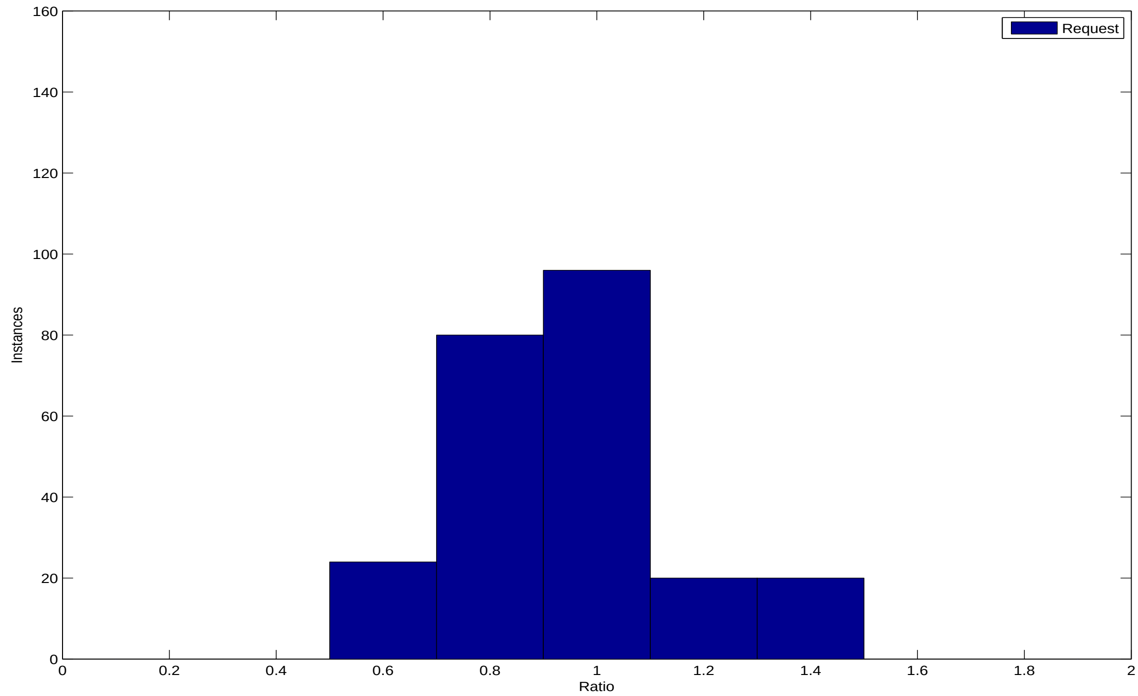


Figure 4.14: Cumilative airspace complexity for the input scenario measured using the perfect model and the four Basic Simulators

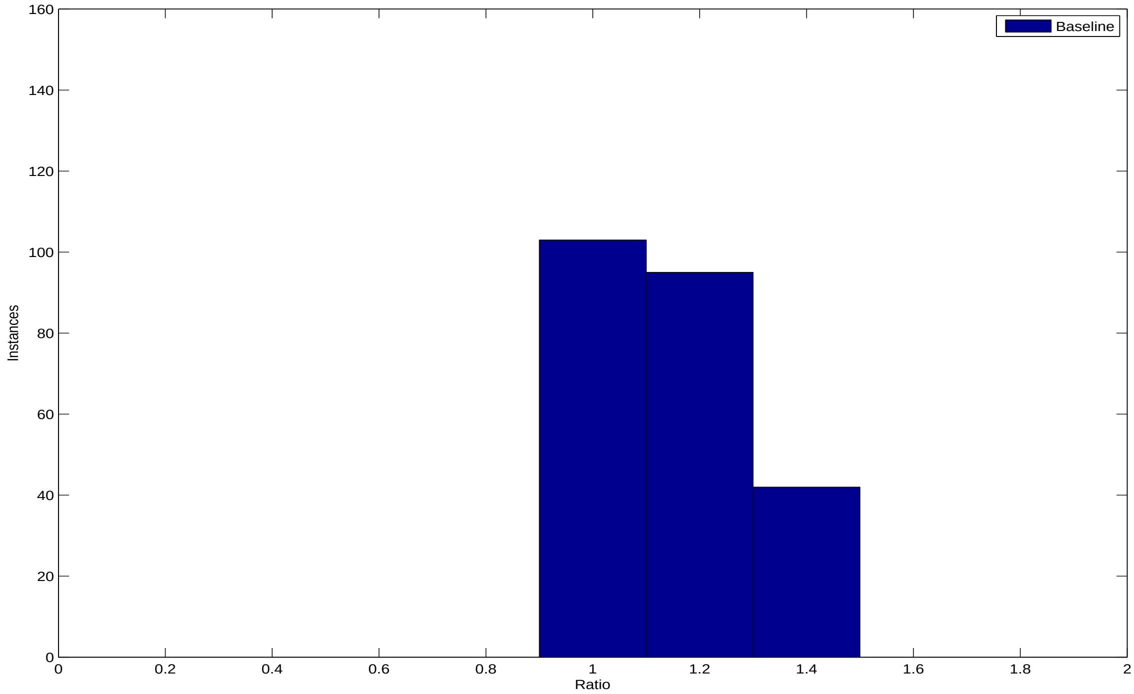


(a) R_B with LKA = 15 minutes for BS1

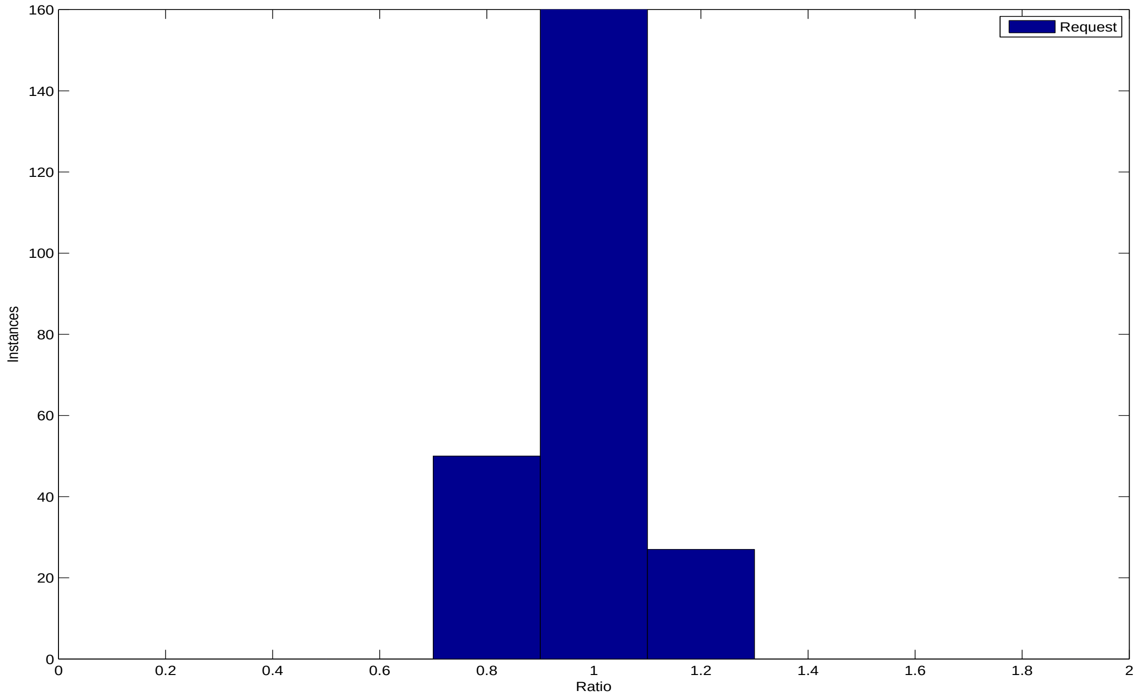


(b) R_R with LKA = 15 minutes for BS1

Figure 4.15: Comparison of the ratio of airspace complexity measured by the perfect model to the complexity measured by BS1 for a lookahead horizon (LKA) of 15 minutes

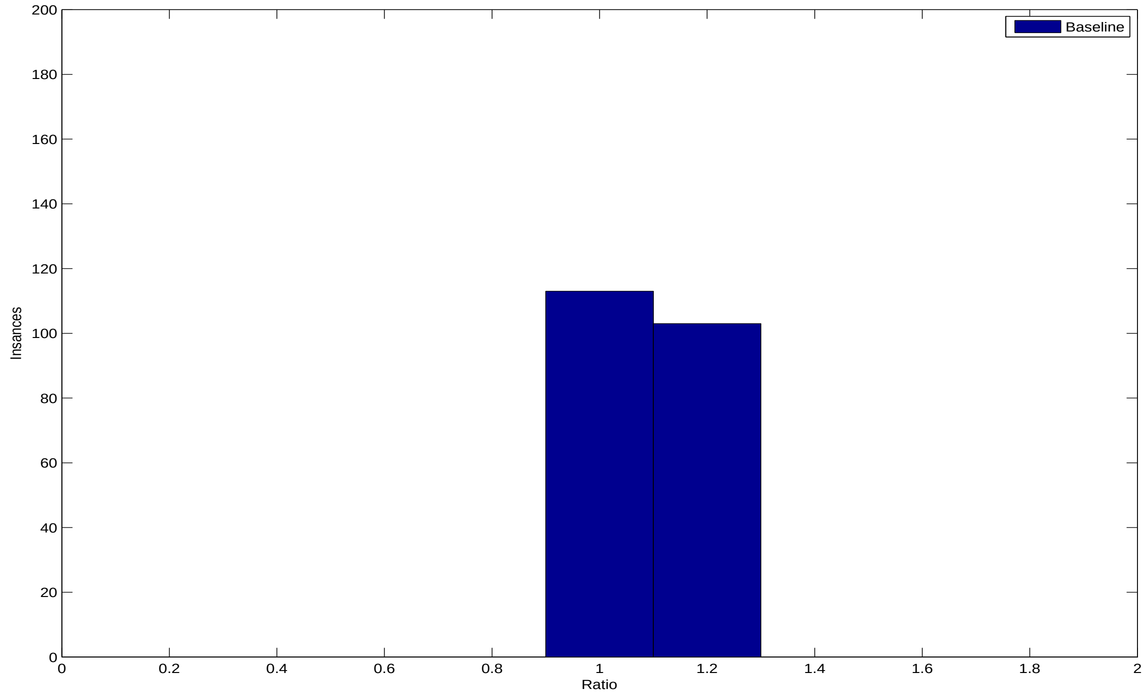


(a) R_B with LKA = 30 minutes for BS1

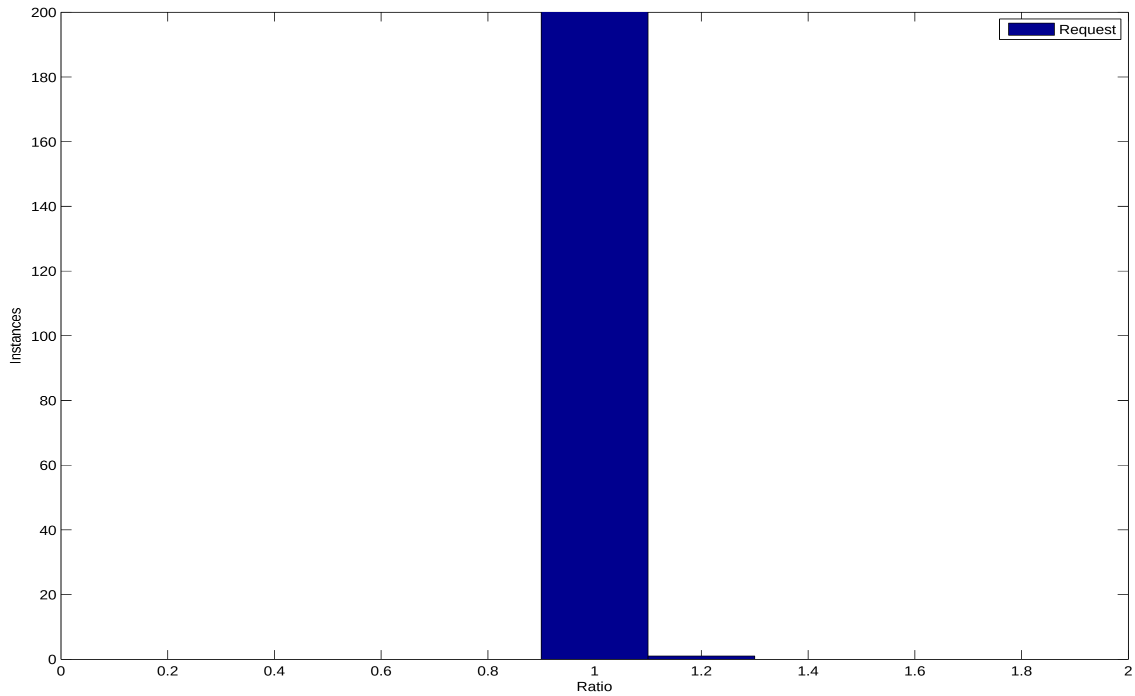


(b) R_B with LKA = 30 minutes for BS1

Figure 4.16: Comparison of the ratio of airspace complexity measured by the perfect model to the complexity measured by BS1 for a lookahead horizon of 30 minutes

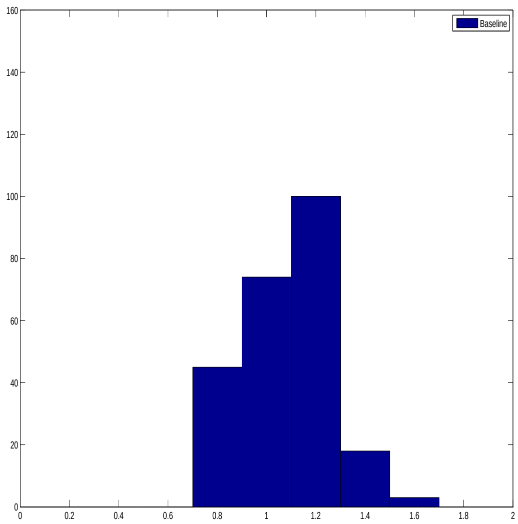


(a) R_B with LKA = 60 minutes for BS1

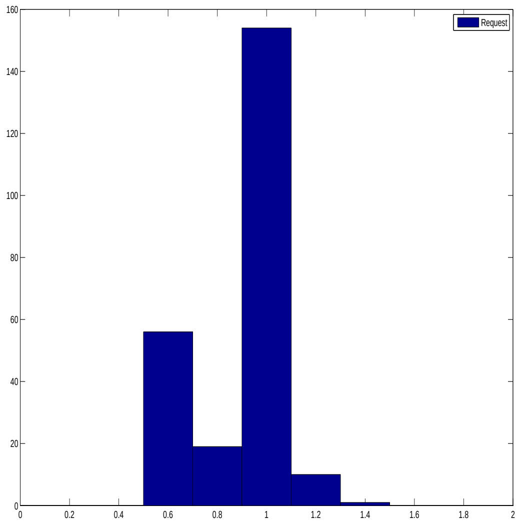


(b) R_B with LKA = 60 minutes for BS1

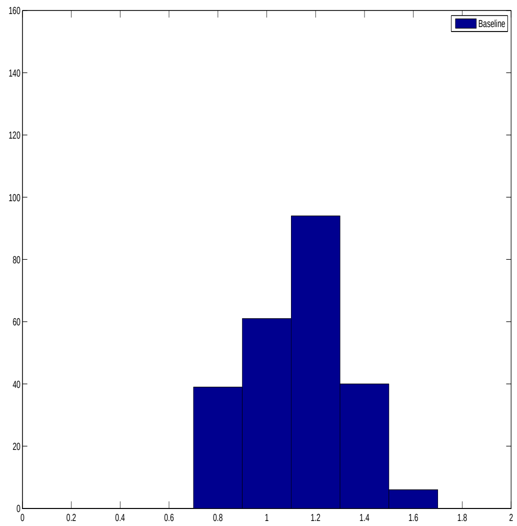
Figure 4.17: Comparison of the ratio of airspace complexity measured by the perfect model to the complexity measured by BS1 for a lookahead horizon of 60 minutes



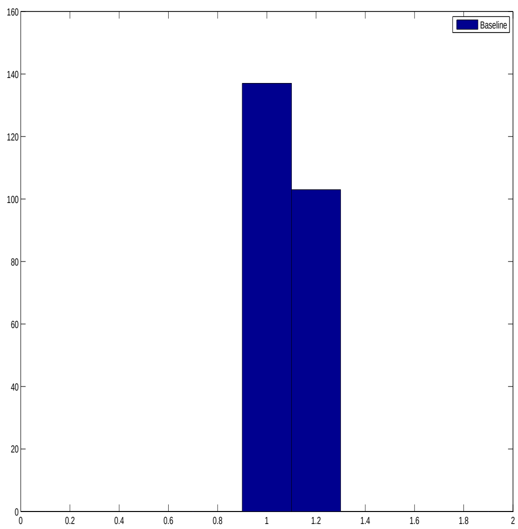
(a) R_B with LKA = 15 minutes for BS2



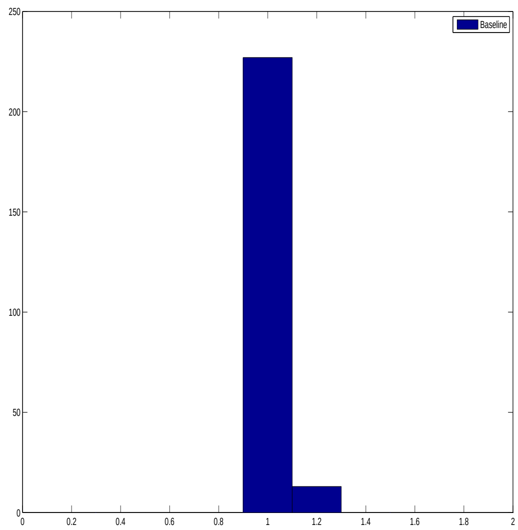
(b) R_R with LKA = 15 minutes for BS2



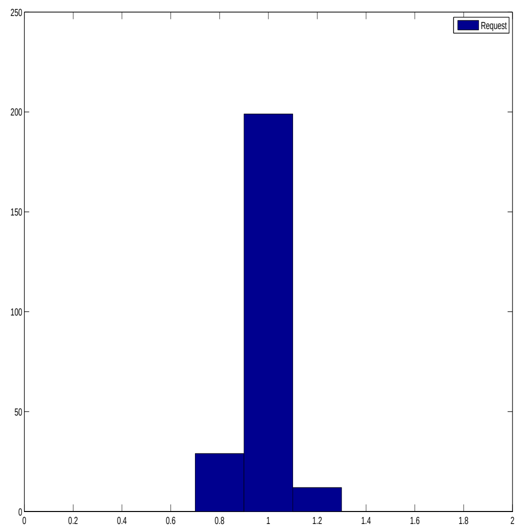
(c) R_B with LKA = 30 minutes for BS2



(d) R_R with LKA = 30 minutes for BS2

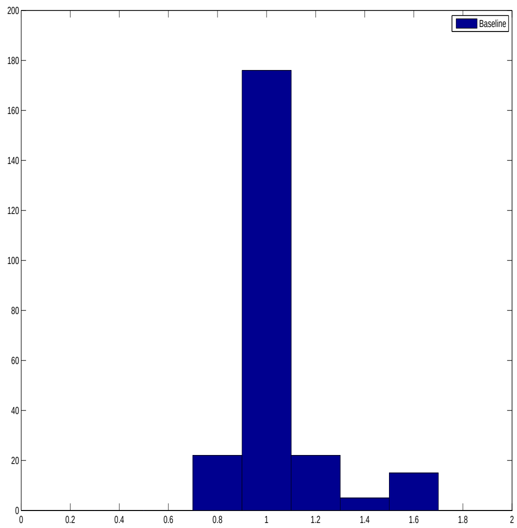


(e) R_B with LKA = 60 minutes for BS2

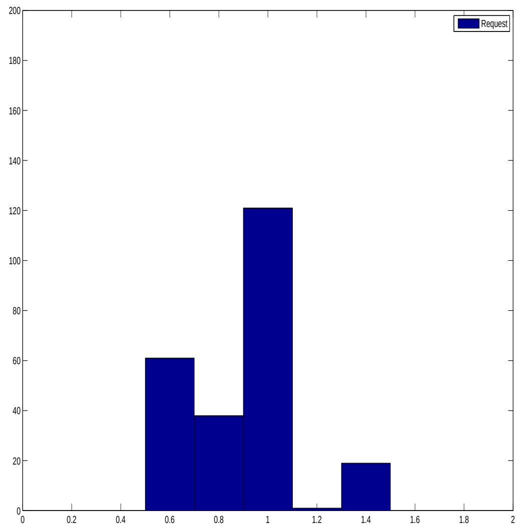


(f) R_R with LKA = 60 minutes for BS2

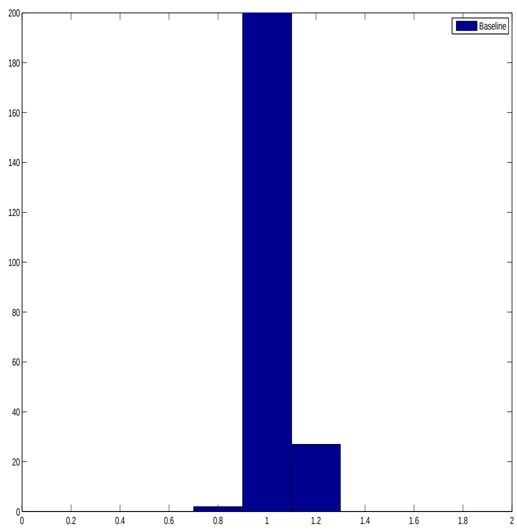
Figure 4.18: Comparison of the ratio of airspace complexity measured by the perfect model to the complexity measured by BS2



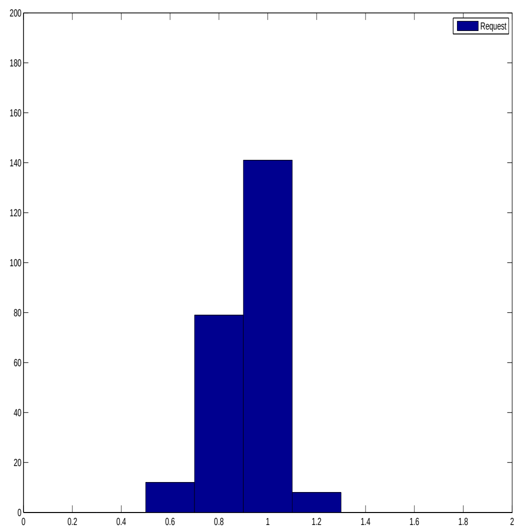
(a) R_B with LKA = 15 minutes for BS3



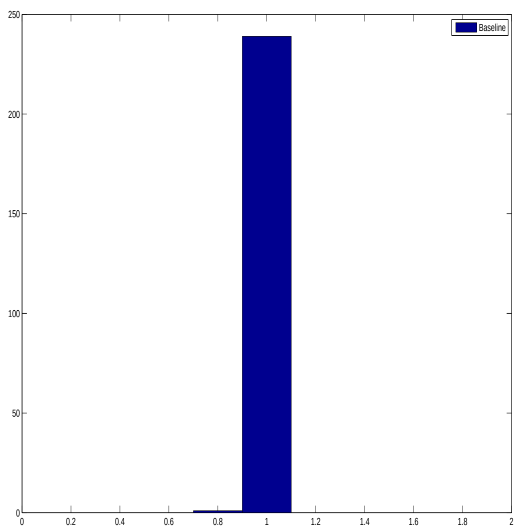
(b) R_R with LKA = 15 minutes for BS3



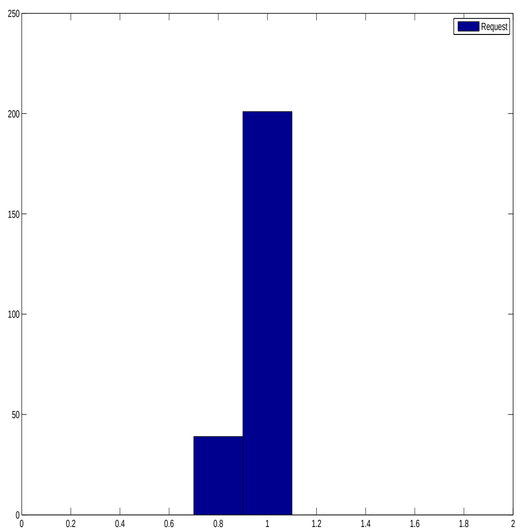
(c) R_B with LKA = 30 minutes for BS3



(d) R_R with LKA = 30 minutes for BS3

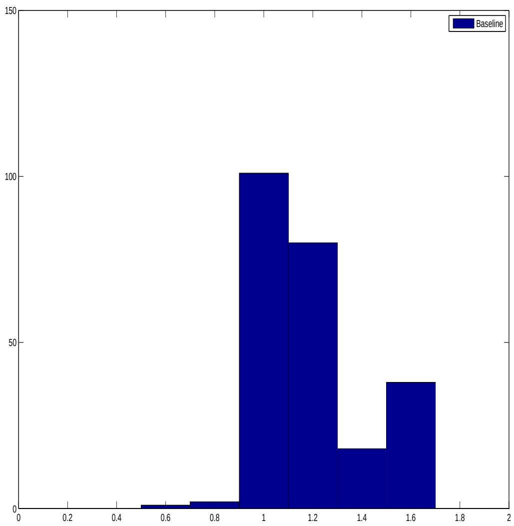


(e) R_B with LKA = 60 minutes for BS3

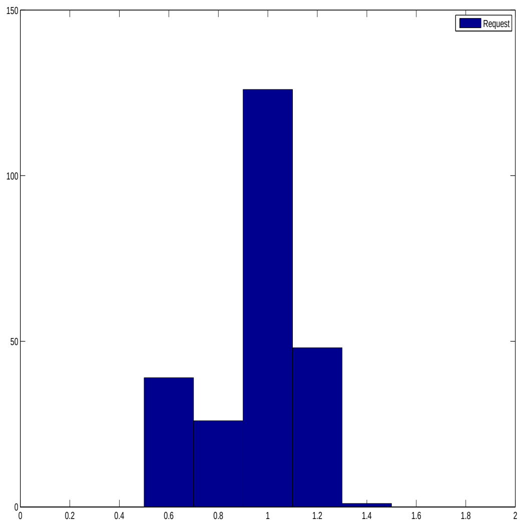


(f) R_R with LKA = 60 minutes for BS3

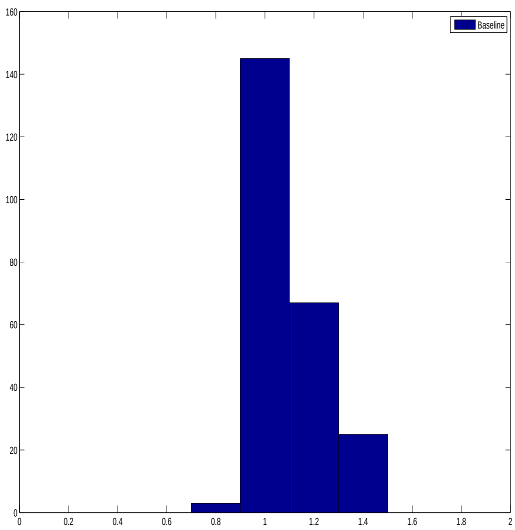
Figure 4.19: Comparison of the ratio of airspace complexity measured by the perfect model to the complexity measured by BS3



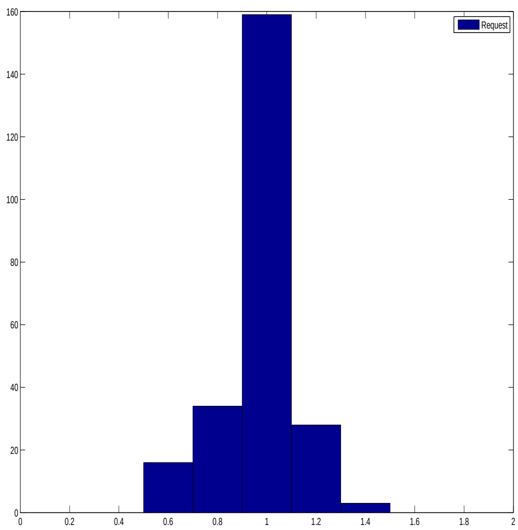
(a) R_B with LKA = 15 minutes for BS4



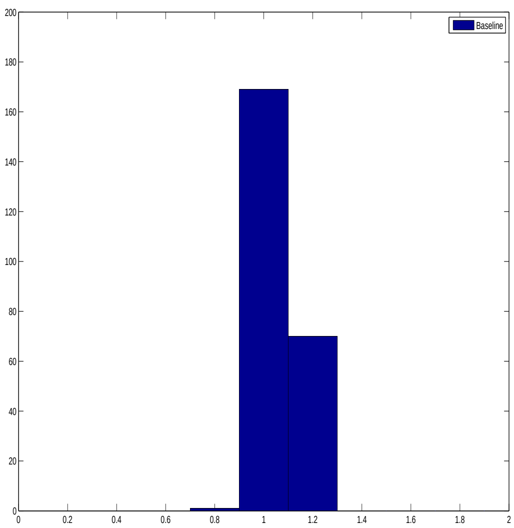
(b) R_R with LKA = 15 minutes for BS4



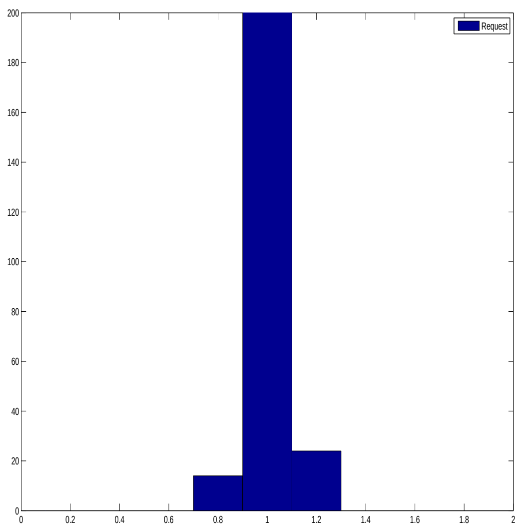
(c) R_B with LKA = 30 minutes for BS4



(d) R_R with LKA = 30 minutes for BS4



(e) R_B with LKA = 60 minutes for BS4



(f) R_R with LKA = 60 minutes for BS4

Figure 4.20: Comparison of the ratio of airspace complexity measured by the perfect model to the complexity measured by BS4

Chapter 5

Real-time Simulation-based Complexity Prediction

5.1 Introduction

Numerous automation tools have been developed to assist ATC's in maintaining safe and efficient air traffic flows (Nolan, 2010). These tools mainly focus on conflict detection and resolution (Paielli *et al.*, 2009) and do not give an assessment of the expected change in airspace complexity should these detected future scenarios occur. If these future airspace complexities assessments were available, particularly in real time, it would assist managers and supervisors at air traffic control facilities to appropriately allocate or re-allocate resources to handle peaks in the workload to reduce risk to the system and appropriately apply advanced procedures such as dynamic sectorisation (Ehrmanntraut and McMillan, 2007). However few systems exist which can be used to predict future workload (Chatterji and Sridhar, 1999), these systems focus mainly on single controllers or single sectors. For this reason we require a system which is able to predict the future workload of a large set of sectors in a large airspace, such as the entire Australian, US or European airspaces; in real-time. Artificial intelligence techniques have been used extensively in many domains to solve prediction problems, and the transportation domain is no exception (Avineri, 2012).

A common AI prediction method is simulation (Glasa, 2009). Simulation can also be used to determine the future air traffic conditions, and therefore predict the airspace complexities in the future. In the previous chapters we have seen that for real-time applications it is necessary to trade-off some aspects of the simulation environment, such as the level of fidelity, in order to produce results in an acceptable time frame. This is particularly true when simulating large amounts of air traffic for obtaining results for real-time applications. Every prediction system has errors, and the trade-offs in the simulation fidelity causes deviations in the prediction of the future air traffic conditions when compared to reality. For this reason we require a method to extract and model the error to facilitate more accurate predictions. If we can find the causality of the errors it will make for a more powerful prediction. This now involves a machine learning problem (Kotsiantis *et al.*, 2007).

AI also encompasses a range of methods and approaches inspired by intelligent biological systems (Sadek, 2007). The methods include knowledge-based systems (Akerkar and Sajja, 2010), neural networks (Baldi and Hornik, 1989), fuzzy systems (Cox, 1992) and evolutionary computing (Eiben and Smith, 2003). Problems in the transportation domain often deal with complex systems which are difficult to model using traditional statistical approaches. This is because it is not always fully understood what the interactions between different systems are and the system often include a level of uncertainty. Building empirical models using observed data may be the only option for modelling these complex systems. Neural networks (NN) is a machine learning method which presents itself as an ideal method for applications in these situations due to its universal function approximation capabilities (Sadek, 2007).

Neural networks are composed of a series of neurons which are connected in such a way that they are able to learn in a manner similar to how human brains learn (Sadek, 2007). The interactivity between the connected neurons are controlled by adjustable parameters called weights. The neurons are arranged into an input layer, a hidden layer or an output layer (Dougherty, 1995). There are many ways in which the neurons can be connected. One of the common methods is the multi-layer perceptron which is a feed-forward network. A feed-forward network is one where information flow is one directional, that is, the data from the input layer goes through the hidden layer and then to the output layer (Bebis and Georgiopoulos, 1994). The input layer contains one neuron per input variable, the hidden layer processes the information and encodes the knowledge in the network and the output layer contains the target output. NN can be trained using supervised learning and backpropagation (Dougherty, 1995). Supervised learning is an approach whereby a model learns the transformation of the inputs into the output and the model is adjusted so that the error between the model output and the known or desired output is minimised (Jordan and Rumelhart, 1992). The NN algorithm uses a training data set to learn the relationship between the input data and the target output data. The algorithm iteratively adjusts the weights connecting the neurons in order to minimise the error between the output of the network and the desired output (Dougherty, 1995).

In this chapter we present several methods for predicting the future airspace complexities for a set of sectors in a large airspace in real time. First we use multiple linear regression and neural networks to predict the future airspace complexity based on current air traffic conditions. Next we use low fidelity simulators to predict the future airspace complexity and use linear regression to predict the deviation of these prediction from the actual measured complexities.

5.2 Predicting Airspace complexity

The airspace complexity of any sector can be calculated using Equation 4.1. This equation takes into consideration the number of aircraft within the sector, the number of aircraft changing speed, heading or altitude and the number of aircraft pairs whose separation falls into several categories. The airspace complexity for a set of sectors, $\{s_k\}_{k=1}^{M}$ where s_k is the kth sector and M is the number of sectors in the set; at a future point in time can be predicted using a variety of different approaches. After measuring the complexities from the real-time environment and predicting the future complexities we have a set of data as follows:

$C = \{c_s\}_{s=1}^{N}$ List of measured complexities from the real-time environment, where c_s is a list of complexities for sector s and N is the number of sectors.

$c_s = \{c_{s,t}\}_{t=1}^{M}$ Measured complexities for each sector, where $c_{s,t}$ is the measured complexity for sector s at time t and M is the number of measurements made.

$d_{s,t} = d_{s,t,1}, d_{s,t,2}, ..., d_{s,t,n}$ List of air traffic characteristics used to measure the complexity ($c_{s,t}$) in sector s at time t. n is the number of characteristics.

$P = \{p_s\}_{s=1}^{N}$ List of predicted complexities, where p_s is a list of predicted complexities for sector s and N is the number of sectors.

$p_s = \{p_{s,t}\}_{t=1}^{M}$ List of predicted complexities for each sector, where $p_{s,t}$ is a list of predicted complexities for sector s made at time t. M is the number of times predictions were made.

$p_{s,t} = \{p_{s,t,i}, p_{s,t,i+1}, ..., p_{s,t,n}\}$ Predicted complexity, where $p_{s,t,i}$ is the predicted complexity for sector s. The prediction made at time t for time $t + i$ where i is the look ahead period, that is the amount of time into the future that the prediction was made for.

Using a snapshot of the traffic data at any given time we can use machine learning techniques to predict the future airspace complexity. This snapshot of air traffic data can also be used to simulate the air traffic for the given time horizon using different simulation models, such as the ones introduced in the previous chapters. As we have been using low fidelity simulators for predicting the air traffic conditions in the future, it is possible that there will be some deviation between the predicted level of complexity and the actual level of complexity as measured in the real time environment. For this reason we also require a method for correcting for this level of error in the prediction.

In the following section we will use the current traffic conditions, A_s, in each sector to generate multiple linear regression models and to train neural networks in order to predict the future airspace complexity. Next we will use a combination of the current traffic conditions, A_s, and the current airspace complexity characteristics, $d_{s,t}$, to predict the airspace complexity and finally we will use simulation to predict the airspace complexity and use linear regression to estimate the deviation of this prediction from the actual measured complexity to determine a more accurate prediction, $r_{s,t}$.

5.3 Predicting Airspace Complexity Using Air Traffic Characteristics

First, we used multiple linear regression and neural networks to predict the future airspace complexity using the current traffic characteristics. An overview of the process used to learn the current traffic conditions which can lead to the future airspace complexity levels can be seen in Figure 5.1. First the air traffic data from the real time environment was monitored for a period of time. During this period, a snapshot of air traffic data from the real time environment was periodically used to measure the airspace complexity for each sector in the airspace. A map of the sectors in the Australian airspace can be seen in Figure 5.2. From this figure we can see the relative position, size and geometry of the different sectors in the airspace. The snapshot used to calculate the airspace complexity contains the position of every active aircraft in the airspace, along with it's altitude, speed and heading and also it's expected route. The measured complexities and the air traffic characteristics at the corresponding time were recorded in a database.

At the end of the monitoring period measured complexities and the air traffic data were grouped by the sector (s) and time for offline analysis. The first step in the offline analysis was to aggregate the air traffic data for each snapshot to a form which can be used easily with the multiple linear regression and the neural networks.

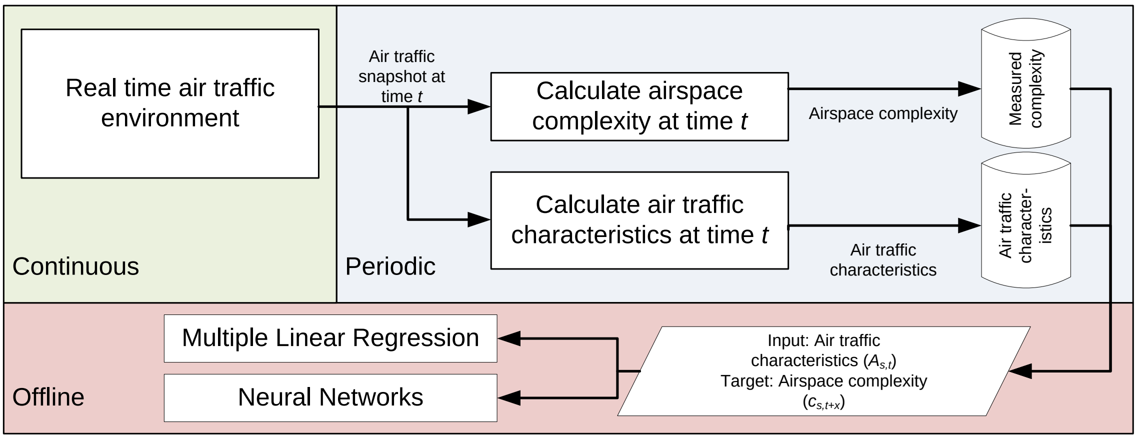


Figure 5.1: Learning traffic conditions in the airspace to predict future airspace complexity

Using the traffic data we calculate the following characteristics:

- Number of aircraft in the sector (a_1)
- The ratio of the sector area occupied by the area of the convex hull formed by the aircraft within the sector (a_2)
- Smallest separation distance among every pair of aircraft (a_3)
- Largest separation distance among every pair of aircraft (a_4)
- Average separation distance among every pair of aircraft (a_5)
- Average speed of aircraft within the sector (a_6)
- Average heading of aircraft within the sector (a_7)
- Average altitude of aircraft within the sector (a_8)
- Number of aircraft changing altitude (a_9)
- Average expected heading change of the aircraft within the sector over the following x minutes ($a_{10,x}$)

The convex hull was found by using the *convhull* function in Matlab (MathWorks, 2015a). This function takes a list of points (aircraft latitudes and longitudes

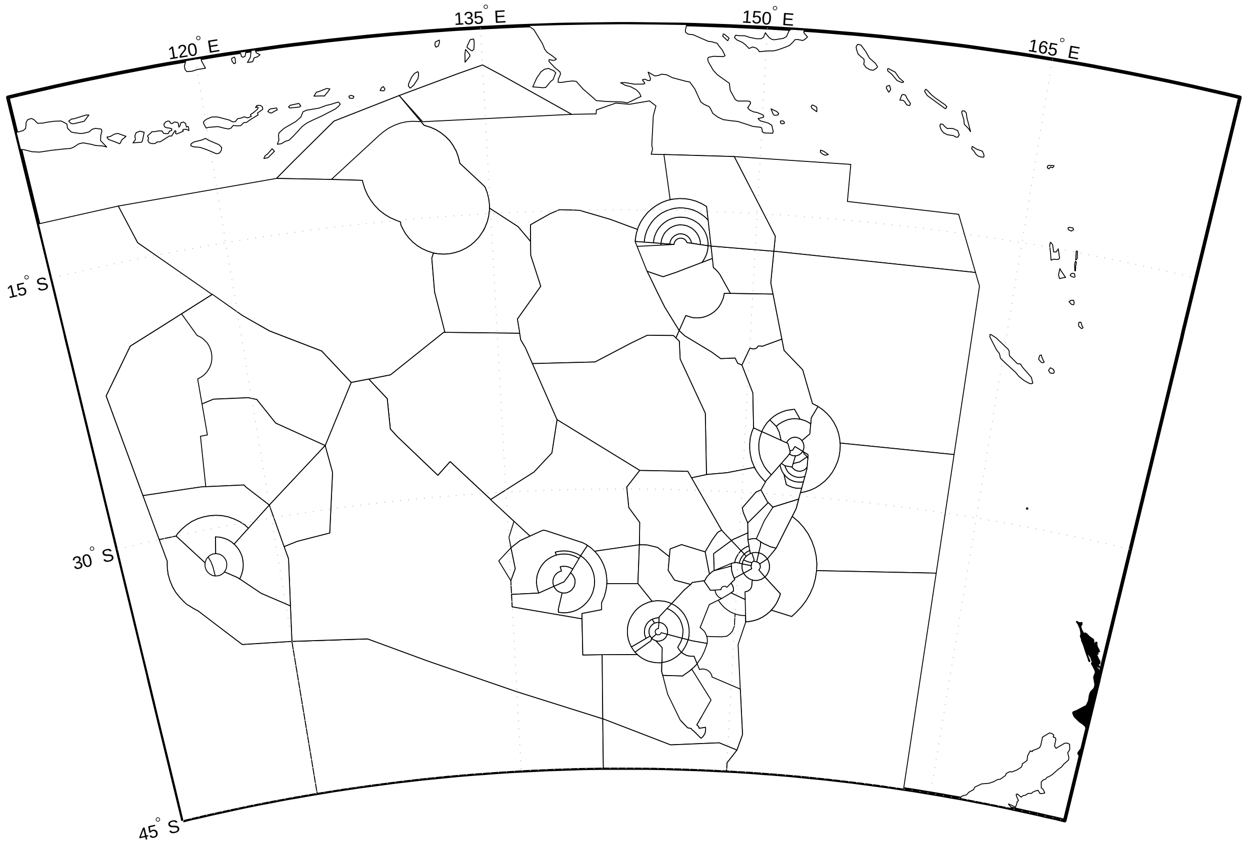


Figure 5.2: High sectors in the Australian airspace

in this case) and returns the smallest polygon that contains all of the points. The area formed by these points was then calculated. Once we have found the area of the convex hull we find the ratio, a_2, of the area of the sector occupied by the convex hull. An example of a convex hull for a snapshot for one sector can be seen in Figure 5.3. In this figure we can see the sector boundary (red) and the aircraft (blue) located within the sector at this particular snapshot. The aircraft is designated with a different symbol based on their altitude and a 5 minute intent line is also shown (pink) representing the aircraft's expected position within the next 5 minutes based on their flight plan and current speed. The convex hull formed by these aircraft is shown in green. The ratio between the sector area and the convex hull area represents the amount of freedom the ATC may have in altering the trajectories of the aircraft to handle problems which may arise.

For a_3, a_4 and a_5, the separation distance between every aircraft pair within

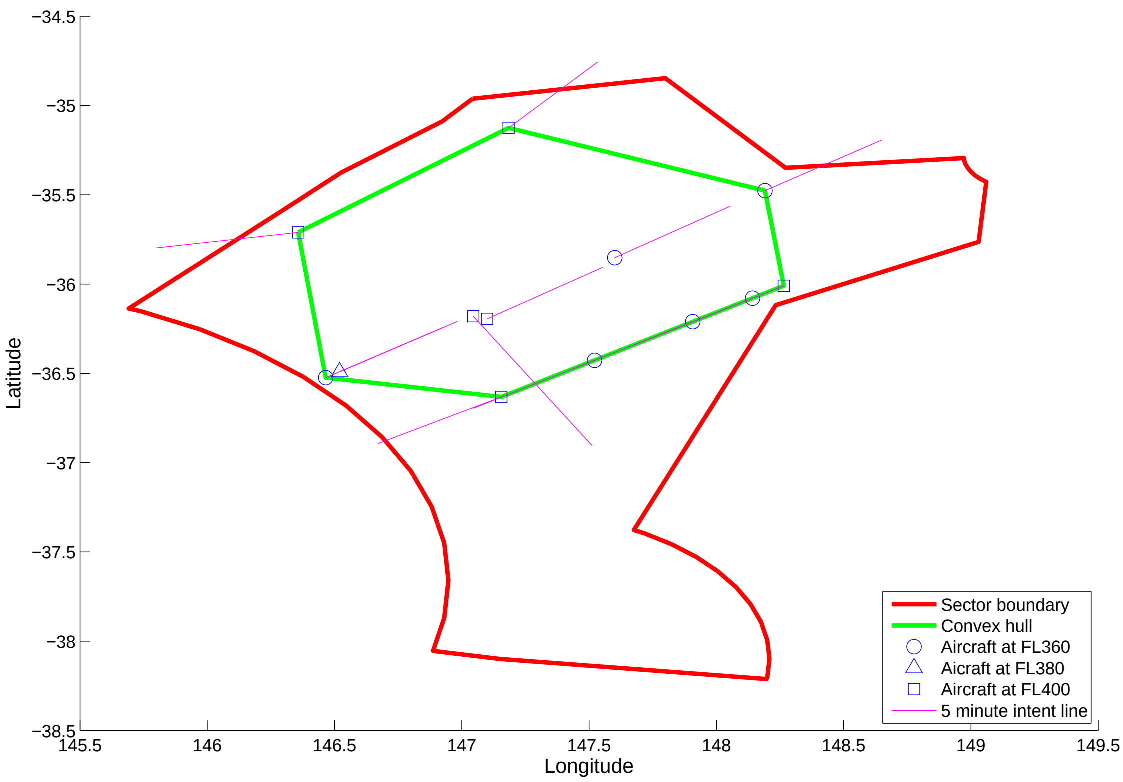


Figure 5.3: Convex hull of aircraft within the sector

the sector was measured. The shortest, longest and average separation distances were then found. This distance is measured in the lateral and longitudinal axis for aircraft pairs separated vertically by 2,000ft or less.

For $a_{10,x}$, it was determined how far the aircraft would travel in x minutes (the lookahead period) with its current speed. Based on this estimated distance, it was determined how many waypoints would have been passed by the aircraft in this time and therefore its expected heading after x minutes. This expected heading and the aircraft's current heading were used to calculate the expected heading change as follows:

$$h_{change} = min(|h_{current} - h_{expected}|, 360 - |h_{current} - h_{expected}|) \tag{5.1}$$

where $h_{current}$ is the current heading of the aircraft, that is the heading of the aircraft when the snapshot was taken, and $h_{expected}$ is the expect heading after x

minutes.

These characteristics were used to create a vector, $A_{s,t,x} = [a_1, a_2, ..., a_9, a_{10,x}]$, for each snapshot for each sector, each snapshot time and each lookahead period combination we are interested in investigating. Next each A vector is matched with the complexity measurement from the lookahead period ($c_{s,t+x}$). For example, if the lookahead time is 5 minutes, then we match the traffic characteristic vector for sector s at time $t =$ 08:10 with the measured complexity for sector s from $t + x =$ 08:15.

We used the $A_{s,t,x}$ vectors as the input for training the neural networks and for generating the linear regression models. The target for learning are the corresponding $c_{s,t+x}$ complexity values. As mentioned in the previous chapter, the real-time environment used to obtain this data could potentially be the real world or a simulated air traffic control environment, but in the case of this study, we once again use a high fidelity air traffic simulator to represent the real-time environment. The high fidelity air traffic simulator used in this study was also ATOMS (the perfect model) and it is assumed to be an accurate representation of the real world for our purposes. An air traffic scenario was generated consisting of 30 days of air traffic activity. This scenario consisted of flight operations typical for this period within the Australian airspace. This scenario is the same as the scenario used for the simulation validation section in Chapter 3. During the simulation of these 30 days of air traffic using the perfect model we recorded the airspace complexity in each sector and a snapshot of the air traffic conditions at a frequency of once every 5 minutes. The airspace complexity is measured using Equation 4.1, which is a weighted sum of several characteristics of the air traffic. The characteristics used to calculate the complexity are different from these contained in the A vector. The air traffic data snapshots were used to generate the A vectors for each sector and matched with the complexities for that sector for lookahead periods of 5, 10 and 15 minutes. This information is then used to generate the linear regression models and neural networks for each sector.

A separate multiple linear regression model was generated for each sector for each of the three lookahead periods. The MATLAB function *regress* (MathWorks, 2015c) is used to generate these models. This function takes a set of predictors, or explanatory variables, (the $A_{s,t,x}$ vectors in our case) and a set of observed values ($c_{s,t+x}$ in our case) to generate a set of coefficient estimates ($\beta_0, ..., \beta_{10}$). The coefficients attempt to model the relationship between the current traffic characteristics and the resulting airspace complexity in the lookahead period. These coefficients are used to produce the model shown in Equation 5.2 where we can multiply the traffic characteristics by their corresponding coefficient to obtain a complexity prediction. In this equation the values $a_1, a_2, ..., a_9, a_{10,x}$ are obtained from the vector $A_{s,t,x}$. We can use the β weights during the real-time flight operations to predict the future airspace complexity (in real-time) using the traffic characteristics observed in the real-time environment.

$$p_{s,t,x} = \beta_0 + \beta_1 a_1 + \beta_2 a_2 + ... + \beta_9 a_9 + \beta_{10} a_{10,x} \tag{5.2}$$

Once we have generated the models for each sector and lookahead combination we can begin to analyse the model for how well it fits the data. This analysis is conducted by measuring the coefficient of determination for each model. The coefficient of determination, R^2, indicates how well the data fits a statistical model on a range from 0 to 1 (Menard, 2000). A R^2 value of 1 means that the model perfectly represents all of the data while a value of 0 means that the model does not represent any of the data. Therefore, the closer the R^2 values is to 1, the better the models are at represent the data. R^2 is calculated using Equation 5.3 where n is the number of samples, y_i is the observed value of the ith sample, $\hat{y}_i$ is the predicted values from using the generated model and $\bar{y}$ is the mean of the observed values.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{5.3}$$

Once we have generated the models for each sector and lookahead period we can determine their R^2 values. As we are using data obtained for the entire Australian airspace it is not practical to present the results for each of the 120 sectors and their three lookahead period. So we will select a representative set of sectors. The sectors were ordered by their average complexity for the duration of this 30 day scenario and the ten sectors with the highest average complexity and the ten sectors with the lowest average complexity were selected. Additionally ten sectors around the median complexity were also chosen, five higher than the median and five lower than the median. The R^2 values for these selected sectors can be seen in Table 5.1. The average R^2 value among all of the sectors can also be seen in the table, as well as the R^2 values for the sector with the highest average R^2 for the 5, 10 and 15 minute lookahead period that is not already included in the table. From this table we can see that there are several sectors which have a R^2 values greater than 0.9. If we investigate the traffic characteristics of these sectors we find that these sectors have a low number of aircraft passing through them and usually have trajectories that are relatively simple (ie. no turning, climbing, descending or crossing tracks with other aircraft). Excluding these sectors, we see that the sectors with the highest average complexity also have the highest R^2 values. We can also see that for majority of the sectors the 5 minute lookahead results in the highest R^2, followed by the 10 minute lookahead period. This suggests that this method becomes less accurate as we increase the lookahead period. There are only two instances in this group where the R value is greater than 0.4 and this suggests that the multiple linear regression method is a poor predictor of future airspace complexity when using current air traffic conditions for the prediction.

The same set of traffic characteristics were also used to train a set of neural networks, also with the aim of predicting future airspace complexity using the current air traffic conditions. This resulted in a separate neutral network for each sector and lookahead period combination. We used the traffic characteristics as the input for training the neural networks and the measured complexity from the lookahead period as the target. This gives us ten inputs and one target. The neural networks were trained and generated using Matlab (MathWorks, 2015d). The configuration for the neural network included a two-layer feedforward network with ten sigmoid hidden neurons and a linear output neuron. Figure 5.4 shows an overview of the neural network architecture. During the training phase, a set of weights in the hidden layer and output layer are determined. These weights are used provide us with a prediction when using the network. The Levenberg-Marquardt backpropagation algorithm Moré (1978) was used to iterate and update the weights in order to minimise learning error. This iteration continues until a minimum error level is reached. We can use these networks during real-time flight operations to predict the future airspace complexity (in real-time) by using the traffic characteristics observed in the real-time environment as inputs into the network.

Once we have generated the various neural networks we can again analyse there fit of the data by calculating the coefficient of determination, R^2, for each network. Table 5.2 shows the R^2 values for lookahead periods of 5, 10 and 15 minutes for ten sectors with the highest average complexity and the ten sectors with the lowest average complexity were selected. Additionally ten sectors around the median complexity are also shown, five higher with an average complexity higher than the median and five lower. This table also shows the average R^2 value among all of the sectors can also be seen in the table, as well as the R^2 values for the sector with the highest average R^2 for the 5, 10 and 15 minute lookahead period that is not already included in the table. The sectors selected for this table are the same as those shown in Table 5.1 for the multiple linear regression. We can see, again, that the sectors with the highest average complexity have the highest R^2 values. We can also see that, for this group, when compared to the corresponding R^2 from Table 5.1 for the multiple linear regression, there is a general improvement in the R^2 values. However, in the two other groups we see that the results are largely the same. But the overall average R^2 for all sectors when using neural networks is greater than when using multiple linear regression. This shows that the neural networks produce models which on average produce a better fit for the data than when using multiple linear regression. However the best R^2 values still quite poor.

Table 5.1: Coefficient of determination for multiple linear regression for a selection of sectors

Sector	**Average complexity**	**R^2 for lookahead period**		
		5 min	**10 min**	**15 min**
Sectors with the highest average complexity				
YBBB YMMM NULLARBOR A	32.70	0.35	0.34	0.33
YMMM BENALLA	28.35	0.32	0.27	0.23
YBBB INVERELL A	25.25	0.31	0.22	0.14
YBBB YMMM MUDGEE D	22.44	0.31	0.27	0.24
YBBB TERRITORY	22.23	0.30	0.28	0.25
YBBB YMMM ALICE SPRINGS	21.41	0.37	0.36	0.35
YBBB YMMM ISA	19.41	0.34	0.32	0.29
YBBB YMMM WARREGO	18.91	0.37	0.35	0.34
YMMM BILLABONG	18.40	0.44	0.42	0.40
YBBB YMMM KIMBERLEY	17.37	0.36	0.33	0.31
Sectors with average complexity around the median				
YMMM BINDOOK SYDNEY CAP	6.88	0.04	0.01	0.01
YBBB DARWIN ARRIVALS	6.70	0.25	0.21	0.11
YMMM WOLLONGONG A	6.58	0.08	0.01	0.01
YMMM WONTHAGGI A	6.34	0.09	0.02	0.01
YBBB YMMM NICKEL	6.06	1.00	0.96	1.00
YBBB KENNEDY C	5.86	0.10	0.01	0.01
YMMM AUGUSTA HIGH C	5.65	0.21	0.14	0.08
YBBB DOWNS A	5.55	0.08	0.03	0.01
YMMM TAILEM BEND B	5.53	0.10	0.02	0.02
YBBB GOLD COAST C	5.49	0.03	0.03	0.01
Sectors with the lowest average complexity				
YMMM EILDON WEIR HIGH A	3.27	0.11	0.02	0.07
YBBB BURNETT C	3.15	0.03	0.03	0.04
YBBB YMMM KATOOMBA A	2.28	0.08	0.07	0.13
YMMM WONTHAGGI E	2.26	0.04	0.33	0.22
YBBB BARRA C	2.26	0.03	0.03	0.04
YBBB GOLD COAST B	2.04	0.12	0.06	0.07
YBBB BARRA E	2.04	0.07	0.09	0.09
YMMM KATOOMBA SYDNEY CAP	1.73	0.16	1.00	1.00
YBBB YMMM MUDGEE A	1.66	0.25	0.65	0.06
YMMM AUGUSTA HIGH B	1.33	0.50	0.61	1.00
Sector with highest average R^2 not already included	16.69	0.47	0.45	0.43
Overall average		0.20 $\pm$ 0.17	0.19 $\pm$ 0.22	0.18 $\pm$ 0.25

Table 5.2: Coefficient of determination for neural networks for a selection of sectors

Sector	**Average complexity**	R^2 **for lookahead period**		
		5 min	**10 min**	**15 min**
Sectors with the highest average complexity				
YBBB YMMM NULLARBOR A	32.70	0.56	0.55	0.53
YMMM BENALLA	28.35	0.45	0.37	0.29
YBBB INVERELL A	25.25	0.38	0.26	0.16
YBBB YMMM MUDGEE D	22.44	0.50	0.40	0.34
YBBB TERRITORY	22.23	0.48	0.44	0.40
YBBB YMMM ALICE SPRINGS	21.41	0.60	0.57	0.53
YBBB YMMM ISA	19.41	0.53	0.48	0.45
YBBB YMMM WARREGO	18.91	0.61	0.56	0.55
YMMM BILLABONG	18.40	0.63	0.60	0.56
YBBB YMMM KIMBERLEY	17.37	0.54	0.50	0.45
Sectors with average complexity around the median				
YMMM BINDOOK SYDNEY CAP	6.88	0.05	0.02	0.02
YBBB DARWIN ARRIVALS	6.70	0.08	0.02	0.07
YMMM WOLLONGONG A	6.58	0.12	0.03	0.02
YMMM WONTHAGGI A	6.34	0.18	0.03	0.02
YBBB YMMM NICKEL	6.06	0.76	0.62	0.53
YBBB KENNEDY C	5.86	0.22	0.04	0.01
YMMM AUGUSTA HIGH C	5.65	0.20	0.11	0.04
YBBB DOWNS A	5.55	0.19	0.04	0.05
YMMM TAILEM BEND B	5.53	0.14	0.02	0.02
YBBB GOLD COAST C	5.49	0.03	0.06	0.02
Sectors with the lowest average complexity				
YMMM EILDON WEIR HIGH A	3.27	0.14	0.07	0.02
YBBB BURNETT C	3.15	0.01	0.01	0.05
YBBB YMMM KATOOMBA A	2.28	0.09	0.13	0.03
YMMM WONTHAGGI E	2.26	0.09	0.33	0.35
YBBB BARRA C	2.26	0.05	0.14	0.04
YBBB GOLD COAST B	2.04	0.11	0.39	0.05
YBBB BARRA E	2.04	0.08	0.16	0.07
YMMM KATOOMBA SYDNEY CAP	1.73	0.23	0.63	0.46
YBBB YMMM MUDGEE A	1.66	0.17	0.25	0.12
YMMM AUGUSTA HIGH B	1.33	0.21	0.23	0.30
Sector with highest average R^2 not already included	16.69	0.63	0.60	0.56
Overall average		0.30 ± 0.21	0.25 ± 0.19	0.21 ± 0.23

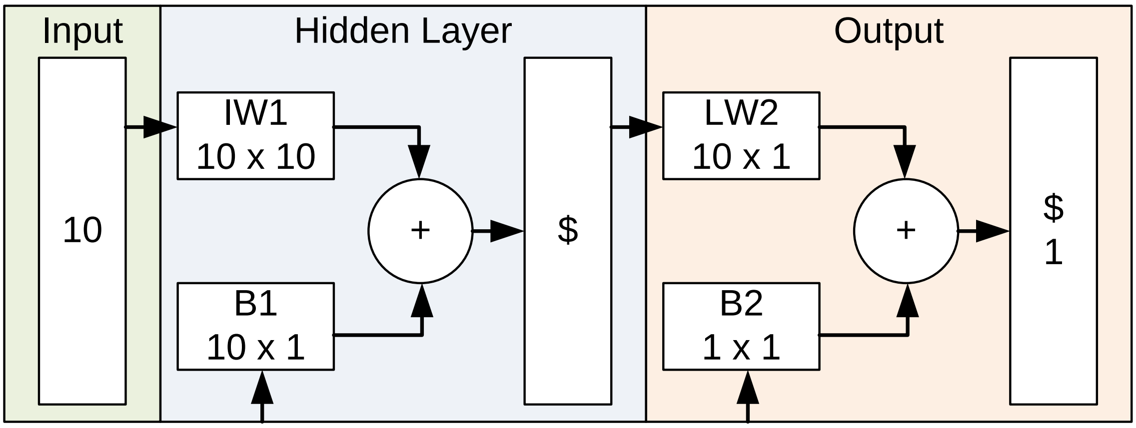


Figure 5.4: Neural network architecture

5.4 Predicting Airspace Complexity Using Complexity Metrics

In the previous section we saw that using the current traffic characteristics to predict the airspace complexity of a sector at a future time using multiple linear regression and neural networks was not very successful. We can attempt to make the prediction obtained from these systems more accurate by directly incorporating some of the characteristics which contribute to the airspace complexity. The airspace complexity at any one time is calculated by a weighted sum (see Equation 4.1) of the following characteristics at the time:

- N - the number of aircraft in the sector
- NH - the number of aircraft in the sector that made a heading change greater than 15° during an interval of 2 minutes
- NS - the number of aircraft in the sector which had a speed change of greater than 10 kts during an interval of 2 minutes
- NA - the number of aircraft in the sector which had an altitude change greater than 750 ft during an interval of 2 minutes
- $S5$ - the number of aircraft in the sector with 3D Euclidean distance between 0-5 NM

- *S10* - the number of aircraft in the sector with 3D Euclidean distance between 5-10 NM
- *S25* - the number of aircraft in the sector with lateral distance between 0-25 NM and vertical separation less than 2000 ft
- *S40* - the number of aircraft in the sector with lateral distance between 25-40 NM and vertical separation less than 2000 ft
- *S70* - the number of aircraft in the sector with lateral distance between 40-70 NM and vertical separation less than 2000 ft

These characteristics can be calculated for any particular time for any sector from the traffic snapshot data. We repeat the multiple linear regression model generation process introduced in the previous section. But this time we use a number of different components to generate these models. The components can be seen in Table 5.3. We have selected six sets of components. The first set consists of the number of aircraft in the sector, the number of aircraft changing heading, changing speed and changing altitude. The second set consists of the 5 aircraft separation characteristics while the third set consists of all nine of the characteristics that contribute to the complexity calculation. The fourth, fifth and sixth sets are a combination of the traffic characteristics a_1 through to a_{10} with the first, second and third sets respectively. It is expected that directly using the components which contribute to current airspace complexity in the sector may guide us towards making more accurate predictions of the future airspace complexity.

Table 5.3: Predictors for multiple linear regression

Set	**Predictors**
1	*N, NH, NS, NA*
2	*S5, S10, S25, S40, S70*
3	*N, NH, NS, NA, S5, S10, S25, S40, S70*
4	$a_1, ..., a_{10}$, *N, NH, NS, NA*
5	$a_1, ..., a_{10}$, *S5, S10, S25, S40, S70*
6	$a_1, ..., a_{10}$, *N, NH, NS, NA,* *S5, S10, S25, S40, S70*

We once again generated a separate regression model for each sector for each of the three lookahead periods of 5, 10 and 15 minutes. But this time we also have 6 separate models for each sector and lookahead period combinations. In order to generate these models we used the components for each of the sets as the predictors while we set the observed values as the measured complexity from the lookahead period. That is, if the components relate to snapshot from t = 8:10 and the lookahead period is 5 minutes then we set the observed value as the measured complexity from t = 8:15. We can use the weights generated for each of the models to predict the future airspace complexity (in real-time) using the traffic characteristics observed in the real-time environment. Once we have generated the models for each sector, lookahead period and set combination we calculate their coefficient of determination, R^2, using Equation 5.3. The average R^2 values for each of the lookahead period and set combinations for all of the sectors can be seen in Table 5.4. If we compare the values in Table 5.4 to the overall averages values for the multiple linear regression with the traffic characteristics in Table 5.1 and those for the neural networks in Table 5.2 we see that using the complexity characteristics as the predictors generally results in a better R^2, meaning we have a generated models with a better fit. With the exception of set 2, all other sets resulted in higher R^2 values than when using only the traffic characteristics as the predictors. We can see that for these sets the R^2 values for the 5 minute lookahead period are again the highest among the three periods. The R^2 values for the two sectors with the highest average complexities can be seen in Table 5.5. We can see from this table that for the sector 'Nullarbor A' all of the sets of predictors led to an improvement in the coefficient of determination when compared to only using the traffic characteristics as predictors. When we compare the R^2 values for the 'Benalla' sector we can see that there is significant improvement when comparing the 5 and 10 minute lookahead periods and a slight improvement for the 15 minute lookahead period. While we can see some improvment in the coefficient of determination for these sets, they are still quite poor and we require a method which is more accurate.

Table 5.4: Average R^2 values for multiple linear regression with each set of components

Set	R^2 for lookahead period		
	5 min	**10 min**	**15 min**
1	0.30	0.25	0.21
2	0.30	0.20	0.16
3	0.38	0.30	0.25
4	0.35	0.31	0.27
5	0.38	0.30	0.27
6	0.42	0.35	0.30

Table 5.5: Average R^2 values for multiple linear regression with each set of components for selected sectors

Set	R^2 for lookahead period for Nullarbor A			R^2 for lookahead period for Benalla		
	5 min	**10 min**	**15 min**	**5 min**	**10 min**	**15 min**
1	0.76	0.74	0.71	0.60	0.46	0.33
2	0.83	0.76	0.71	0.71	0.45	0.26
3	0.93	0.87	0.83	0.74	0.51	0.34
4	0.77	0.75	0.72	0.61	0.46	0.34
5	0.88	0.82	0.77	0.73	0.49	0.32
6	0.93	0.87	0.83	0.74	0.51	0.35

5.5 Predicting Airspace Complexity Using Simulation

We saw in the previous sections that using multiple linear regression and neural networks to predict future airspace complexity was not successful as these methods were not able to properly capture the dynamic environment and the uncertainties it brings. To overcome these challenges in prediction we can substitute the regression and neural networks components with a simulation system to obtain a prediction of the future airspace complexity. The simulation system can into account some of the dynamic characteristics of the air traffic system when making a prediction. The dynamic characteristics include events such as aircraft turning at waypoints, aircraft entering and exiting the sector and ATCs and pilots taking actions to deviate the aircraft from it's flight plan. Using simulation for real-time prediction can however be a time consuming and computationally intensive task. To overcome this issue a simulators with low or lower level of fidelity can be used. We have seen in the previous chapters that using simulators with lower levels of fidelity can also introduce some deviations in our prediction when compared to the observed data. For this reason we require a method by which we can minimise the level of deviation produced by the simulator. An overview of our developed method can be seen in Figure 5.5. In this method we use the periodic air traffic data obtained from the real-time environment to simulate the air traffic using a low fidelity simulator. The predicted airspace complexities from the simulator and the actual measured airspace complexities are stored for further analysis.

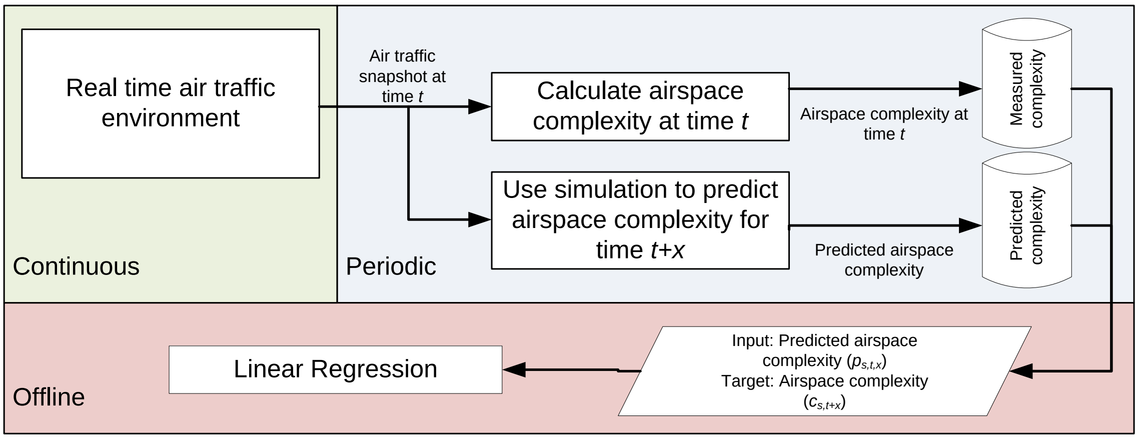


Figure 5.5: Learning the deviations in simulator prediction

Once we have obtained the measured and predicted complexities we can attempt to learn the relationship between the predicted and actual complexities. If we are able to establish a relationship between the deviations of the measured and predicted complexities we can obtain a more accurate prediction of the airspace complexity. In order to establish this relationship we will use linear regression. We will generate a separate linear regression model for each sector and lookahead period combination. In order to generate these models we use the predicted complexities, $p_{s,t,x}$ as the predictors in the regression model and the measured complexity, $c_{s,t+x}$, for the lookahead period, x, as the observed value. From this process we then obtain the weights β_0 and β_1 which we can use in Equation 5.4 to obtain a new, more accurate, prediction of the airspace complexity, $r_{s,t,x}$. This equation, in conjunction with the simulation, can be used to obtain a more accurate prediction of the airspace complexity in real-time.

$$r_{s,t,x} = \beta_0 + \beta_1 p_{s,t,x} \tag{5.4}$$

The traffic snapshot data recorded from 30 day scenario used for analysis in the previous sections was used to simulate the air traffic for lookahead periods of 5, 10 and 15 minutes. The simulations were conducted using snapshot data at an interval of 5 minutes. The simulation itself was conducted using the low fidelity simulators, BS1, BS2, BS3 and BS4; introduced in Chapter 3. The air traffic data from each snapshot time are used simulate all aircraft in the entire Australia airspace and simultaneously obtain a prediction of the airspace complexity for every sector for the lookahead periods of 5, 10 and 15 minutes. We then use these predictions to generate the regression models.

As we saw in Chapter 3, the four simulators (BS1 to BS4) each have a different set of assumptions behind their design. Each of these different assumptions can lead to a potentially different source of error in the simulation and therefore a potentially new source of deviation in the prediction provided by these simulators. The key sources of deviations from the observed events from the real world operations are expected to arise from assumptions regarding climb rates, speed, turning and their representation of the Earth.

An overview of the system developed to predict the airspace complexity with simulation and linear regression can be seen in Figure 5.6. Using the periodic air traffic snapshot data we use the low fidelity simulator to obtain an initial prediction for the airspace complexity and then apply Equation 5.4 to this prediction to obtain a potentially more accurate prediction. This process can provide us with a prediction of the airspace complexity for every sector in the Australian airspace (or any other large airspace) in a matter of seconds.

Handling a large number of regression models (such as one per sector) may provide us with models with a good fit to the data, but they will become time

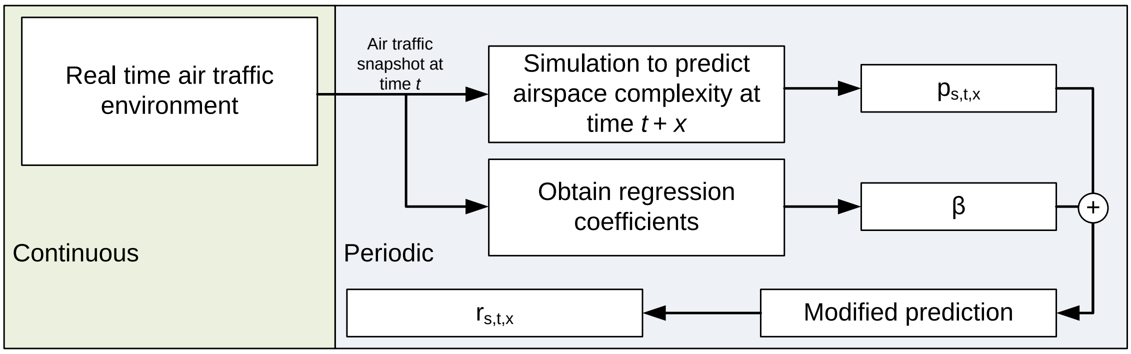


Figure 5.6: Using regression coefficients along with simulation to obtain a more accurate prediction of future airspace complexity

consuming to generate and maintain while it may be possible to use a smaller number of models to obtain with similar results. To determine an appropriate number of models to use we will also investigate the following types of models:

1. A single model encompassing all sectors
2. A separate model for each group where the sectors are grouped based on their traffic and physical characteristics
3. A separate model for each sector

First we will generate a single model to be used for all sectors. Next we will analyse the relationship between the physical and traffic characteristics of the sectors and group the sectors based on similar characteristics. Finally we will generate a separate model for each of the sectors.

5.5.1 Using Data From Every Sector

The first type of model uses data from every sector to create a combined linear regression model. This was done by combining the predicted complexities from each of the 120 sectors for the 30 days into one list. These values were used as the predictors when generated the regression models while the corresponding measured complexities from the lookahead periods of 5, 10 and 15 minutes were set as the observed values. This produced three models, one for each of the lookahead periods, for each of simulators giving us a total of 12 models to analyse. The overall fit of the data to the model was determined using the coefficient of determination, R^2, which is calculated using Equation 5.3 where n is the number of samples, y_i is the observed complexity value corresponding to the ith sample, $\hat{y}_i$ is the predicted value from using the generated model in conjunction with simulation and $\bar{y}$ is the mean of the observed values.

The coefficient of determination for each of the lookahead period and simulator combinations can be seen in Table 5.6. From this table we can see that the resulting regression models are a very good fit to the data as all R^2 values are around or greater than 0.9. This shows that by using any of the four simulators to obtain the complexity for any sector in the airspace and then applying Equation 5.4 we can produce a complexity prediction which is very accurate. This particular setup resulted in better R^2 values than when using neural networks and multiple linear regression to predict the future airspace complexity using the traffic characteristics. From Table 5.6 we can see that we were able to obtain the best R^2 values for BS3 and BS4.

Table 5.6: R^2 values for linear regression of airspace complexities prediction from four simulators

Simulator	**R^2 for lookahead periods**		
	5 min	**10 min**	**15 min**
BS1	0.94	0.92	0.90
BS2	0.94	0.91	0.88
BS3	0.95	0.92	0.91
BS4	0.95	0.94	0.93

If we are able to achieve such a good coefficient of determination for combined models for all sectors we may be able to obtain better results if we were to generate models focusing on a subset of sectors. In the following section we will groups the sectors based on common characteristics and generate regression models for these groups.

5.5.2 Clustering the Sectors

The second type of model investigated requires the clustering of sectors based on similarity of physical and traffic characteristics. The clustering was performed using the k-medoids method with the partition around medoids (PAM) algorithm (Kaufman and Rousseeuw, 2009). The k-medoids method divides a set of observations in k clusters by minimising the sum of the distances between observations and the center of it's cluster. In this method the center is of the cluster is one of the observations. There are several algorithms which can assist in reducing the sum of distances. One such method is the partition around medoids algorithm. This algorithm operate in two steps (MathWorks, 2015b):

- Build step: Each cluster is associated with a potential medoid.
- Swap step: Each point within each cluster is tested as a potential medoid by checking if the sum of distances for the cluster is reduced with the new medoid. If the sum is reduced, a new medoid is set. Every point is the reassigned to the cluster with the closest medoid.

The algorithm continues to iterate through these two steps until there is no change in medoid in successive iterations.

Physical and traffic characteristics

We formed a total of ten clusters based on each sector's traffic and physical characteristics. The previously calculated traffic characteristics for each sector using the air traffic snapshots, $A_{s,t}$, were used to generate a traffic characteristics summary vector, B_s, for each sector. This summary vector contains the mean, standard deviation and maximum values for each of the components in the A vector.

The following physical characteristics were also calculated for each sector:

- The area of the sector (g_1)
- The altitude range of the sector (g_2)

- Number of upper air routes in the sector (g_3)
- Number of intersecting points between one or more routes in the sector (g_4)
- Total distance of routes in the sector (g_5)
- Sector angle variation (g_6)
- Sector centroid distance ratio (g_7)

The altitude range of the sector, g_2, was calculated by finding the difference between the sector's minimum and maximum altitude limits.

The sector angle variation, g_6, was calculated by first making a list of the angle made by each of the points defining the sector. g_6 is determined by finding the variance of the angles within the list. A sector with a uniform shape, such as a circle or a square, will have a lower variances while more irregularly shaped sectors have a higher variance.

For g_7, the centroid of the sector was found and then the furthest and closest points on the sector boundary from the centroid were found. The ratio is calculated by dividing the distance to the closest point by the distance to the furthest point.

The physical characteristics for each sector were combined to form a vector for each sector, $G_s = [g_1, g_2, ..., g_7]$.

The traffic characteristic summary vector, B_s, and the physical characteristics vector, G_s, for each sector was combined to form a new vector, $H_s = [B_s G_s]$. Each of the components of this vector were then normalised to the range 0 to 1 with the largest value for each characteristic among all sectors being assigned 1 while the smallest being assigned 0.

Clustering

Once we have assembled the traffic and physical characteristics summary vector, H, for each sector we can use these vectors to cluster the sectors. By using these characteristics we produce groups of sectors which have similar characteristics such as traffic levels and patterns. We input the list of H vector to cluster the sectors into ten groups. After we obtain the composition of the ten groups we conduct a principle component analysis (Abdi and Williams, 2010) of the H vectors and plot the first two principle components, which can be seen in Figure 5.7. The principle component analysis (PCA) allows us to simplify the components of the H into a set of principle components. The principle components are a linear combination of the H vector. The principle components allows us to visualise the similarity of each of the sectors' traffic and physical characteristics. In Figure 5.7 each of the sectors are plotted with a maker based on the group it was clustered into. A plot of the convex hull for each group can be seen in Figure 5.8. Groups with less than 3 sectors are shown with their markers as a convex hull cannot be formed.

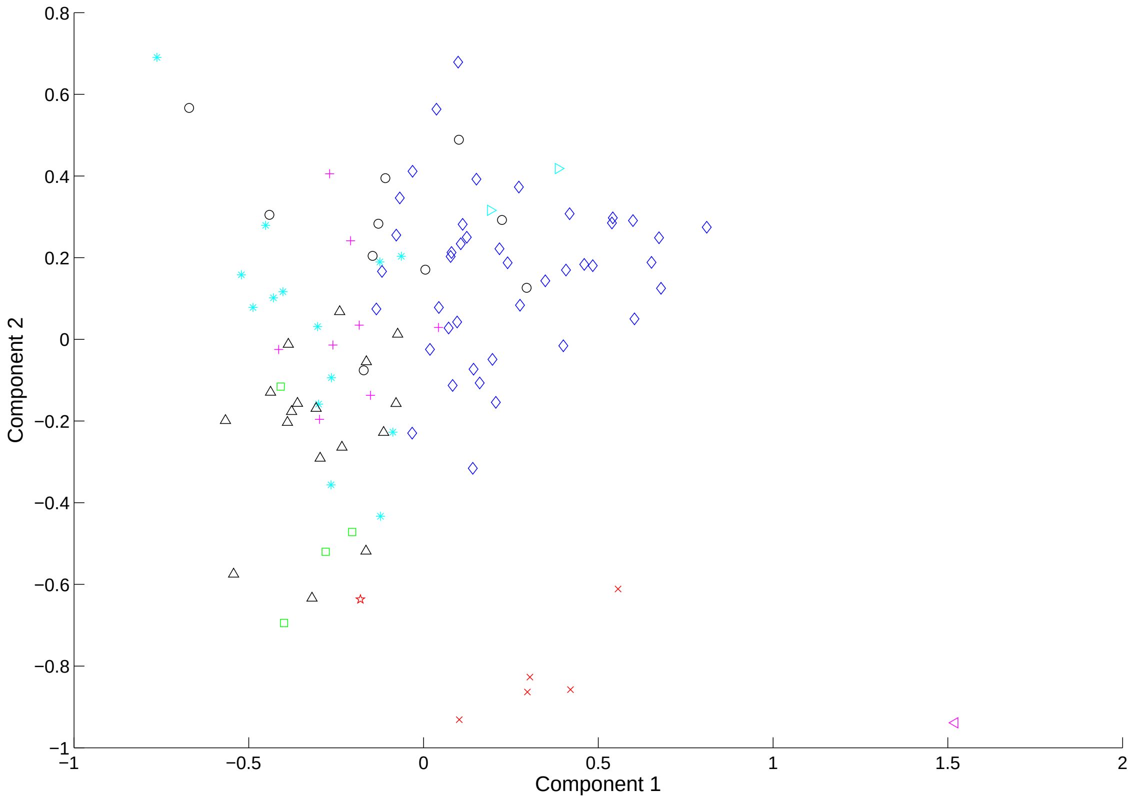


Figure 5.7: Plot of principle component analysis

The predicted complexities for each sector were combined based on the sector's cluster and used to generate a separate regression model for each cluster and lookahead period combination. As we have 10 cluster and we are using 4 different

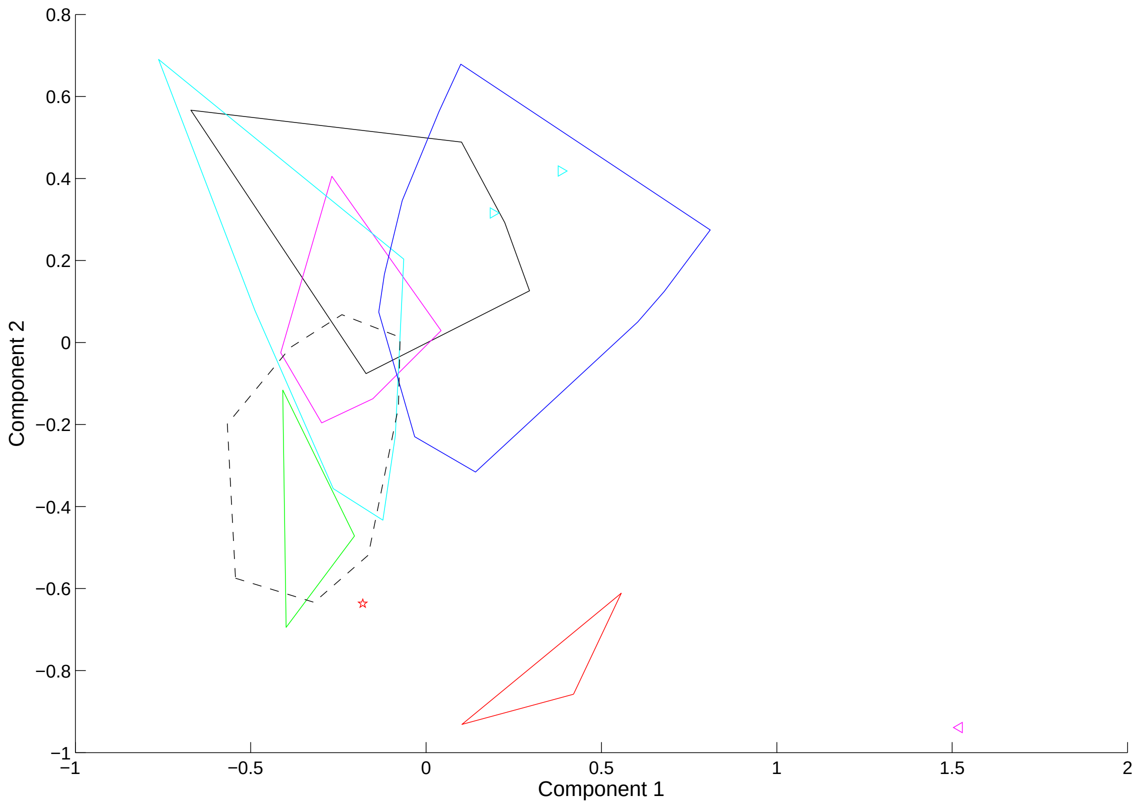


Figure 5.8: Plot of principle component analysis with the convex hull area of each clusters outlined

simulators to predict the complexity we generated a total of 120 models.

Results

The coefficient of determination for each of the generated models for each cluster, lookahead period and simulator combination can be seen in Table 5.7. From this table we can see that the R^2 values are closely related to the cluster rather than the simulator being used to make the initial complexity prediction. As we saw in the previous sections, the R^2 values for the 5 minute lookahead periods were the highest while the values for the 15 minute period were the lowest. Interestingly the cluster with the most sectors also achieved the best R^2 value across all three lookahead periods. A box plot of the R^2 values for each simulator and lookahead period combination can be seen in Figure 5.9. This figure displays a box for each simulator and lookahead period combination outlining the range from the 1st quartile to the 3rd quartile. The box is divided by a horizontal line indicating the 2nd quartile or median. The mean of the group is signified by the circle. From this figure we can see that the simulators BS3 and BS4 have the highest average R^2 values for each lookahead period when compared to BS1 and BS2. However the median value for BS1 and BS4 were higher than the medial value for BS2 and BS3 for all lookahead periods. The results for each individual cluster suggests that the sources of deviations in the complexity predictions from the simulators are not necessarily mapped to particular sector geometries or traffic levels, but more towards other factors of the traffic behaviour as the R^2 for some of the clusters are quite poor, resulting in the mean for some simulators to be lower than the 1st quartile. This is particularly evident from Cluster 4. We can see from Figure 5.7 that the sectors for this clusters, indicated by red crosses, are placed closely together from the principle component analysis of their traffic characteristics summary. However the traffic patterns in these sectors are different and the level of deviations in complexity prediction for these sectors are different. If that is the case then using separate models for each sector may provide us with more accurate results.

5.5.3 Individual Sectors

The third and final type of model we investigated were regression models generated for each sector. We generated three separate models for each sector, one for each of the three lookahead periods. The airspace complexity prediction obtained from the simulator is used as the predictor while the corresponding measured airspace complexity from the lookahead period form the real-time environment is set as the observed value to generate the linear regression model. As each sector has it's own unique traffic patterns which lead to different sources of deviations in the complexity prediction from the simulator, it is expected that using a separate simulator for each sector may result in models with a better fit.

The coefficient of determination for ten sectors with the highest average complexity and the ten sectors with the lowest average complexity were selected and can be seen in Table 5.8, 5.9 and 5.10 for the 5, 10 and 15 minute lookahead periods

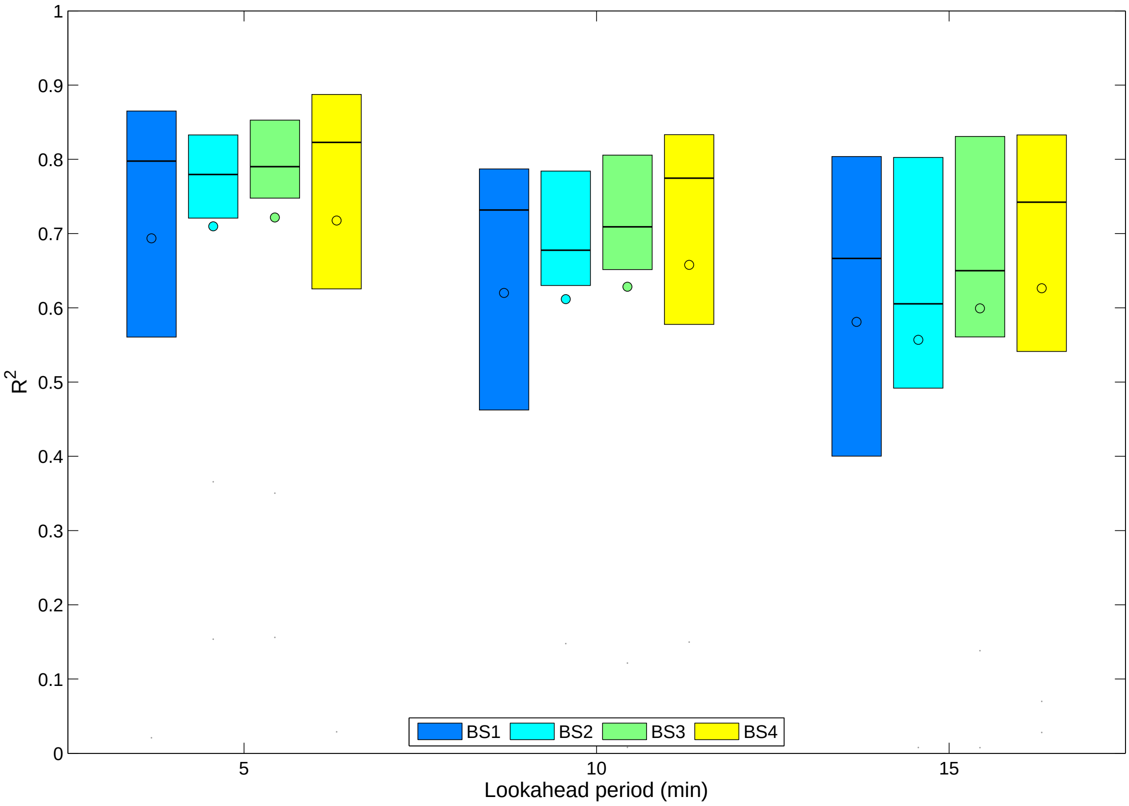


Figure 5.9: Box plot of R^2 values for clusters for each simulator for 5, 10 and 15 minute lookahead periods

respectively. Additionally ten sectors around the median complexity are also shown, five higher with an average complexity higher than the median and five lower, are also included in these tables. From these tables we can see that the overall average R^2 for all four simulators are higher than any of the previously analysed sector-wise models. However we can see that the R^2, across all three lookahead periods, are significantly higher than the other two groups. In fact the R^2 in this group are are close to 1, a perfect fit. From these tables we begin to see some variation in the R^2 values based on the simulator being used and also for the lookahead period, but we see that the sectors with the highest complexity still maintain very high R^2 irrespective of the simulator and lookahead period. A box plot of the R^2 values for each simulator and lookahead period combination can be seen in Figure 5.10. This figure displays a box for each simulator and lookahead period combination outlining the range from the 1st quartile to the 3rd quartile. The box is divided by a horizontal line indicating the 2nd quartile or median. The mean of the group is signified by the circle. We see that BS4 generally has the highest R^2 among the four simulators. Although the average R^2 for BS3 is higher than that of BS1, we can see that both of these simulators have a similar median value. Even though the R^2 values for the sectors with lower average complexities are quite poor, these values are still higher than the previously analysed sector-wise models. As the sectors with the highest complexity would most likely be the candidates for the application of procedures such as the real-time complexity adjustment system introduced in Chapter 4, these results are still encouraging. This means that we can confidently use simulation to predict the airspace complexity of the sectors with high levels of traffic and use this prediction in conjunction with linear regression to produce a more accurate prediction of the complexity for these sectors.

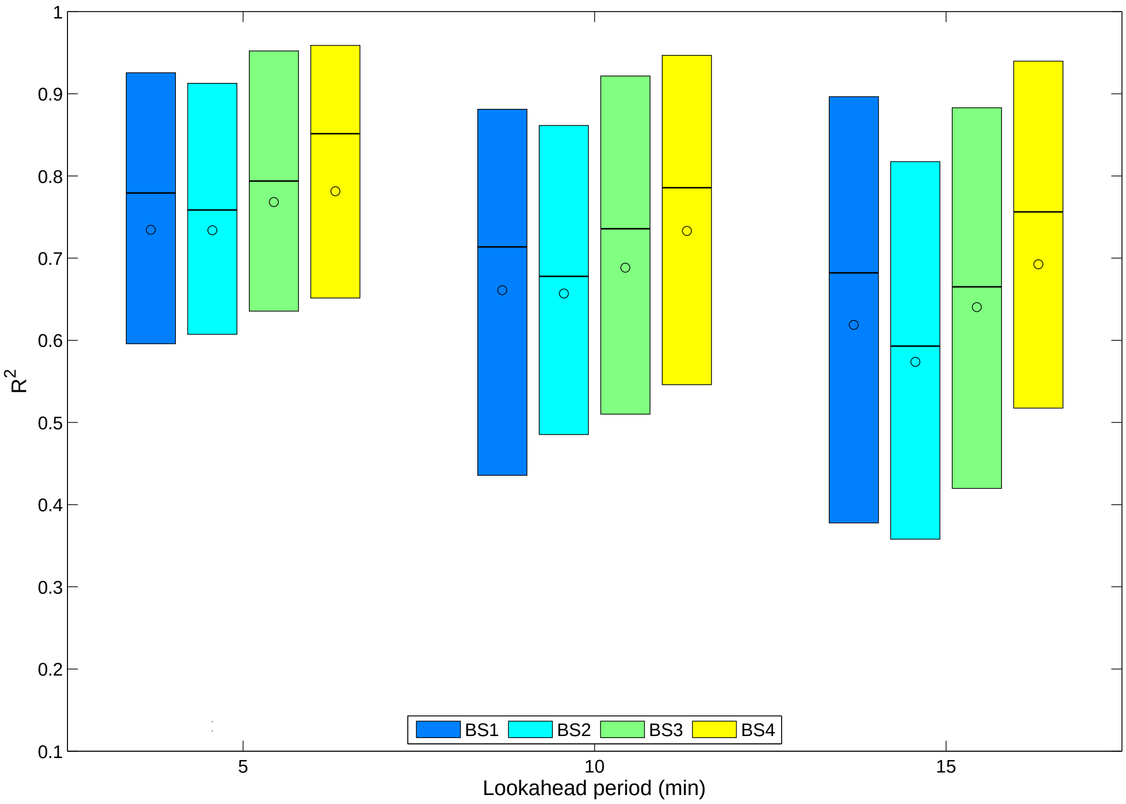


Figure 5.10: Box plot of R^2 values for clusters for each simulator for 5, 10 and 15 minute lookahead periods

5.6 Summary

In this chapter, we designed several methods for predicting the future airspace complexity in real-time simultaneously for multiple sectors in the Australian airspace using an air traffic snapshot from each sector. Initially we generated multiple linear regression models for each sector in order to predict the future airspace complexity using a summary of the air traffic conditions in a sector. The summary of traffic conditions was calculated from a periodic snapshot of the air traffic in the sector. We repeated the same process to generate a set of neural networks to be used to make the same prediction. Some additional traffic characteristics which are used to calculate the airspace complexity were also included to generate a new set of multiple linear regression models. However we found that these three methods did not provide us with a good fit for accurate prediction of future airspace complexity.

We also used simulation to predict airspace complexity. As we are predicting the airspace complexity for sectors encompassing the entire Australian airspace in real-time, we required a fast methodology and so we used several low fidelity simulators to make the predictions. Using low fidelity simulators may introduce some deviation into our results when compared to high fidelity simulators or real-world observations. We generated linear regression models using the measured and predicted complexities. The predicted complexities from the simulator can be combined with the linear regression models to obtain a more accurate estimation of the complexity. We found that the linear regression models for sectors with high traffic levels resulted in good fits to the data, while those with lower levels of traffic did not provide as convincing results. Using the linear regression models in conjunction with our simulators we can somewhat reduce the effect of fidelity and the resulting deviations in the prediction of airspace complexity for these high traffic sectors.

Table 5.7: R^2 values for linear regression of airspace complexities prediction from four simulators by clustering sectors based on common characteristics

Cluster	**Number of sectors**	**R^2 for simulator**			
		BS1	**BS2**	**BS3**	**BS4**
Lookahead period = 5 min					
1	10	0.87	0.83	0.85	0.89
2	12	0.96	0.94	0.96	0.98
3	15	0.81	0.77	0.78	0.83
4	5	0.02	0.15	0.16	0.03
5	13	0.56	0.72	0.75	0.63
6	40	0.96	0.95	0.96	0.97
7	17	0.85	0.80	0.83	0.89
8	1	0.78	0.79	0.79	0.78
9	2	0.79	0.77	0.79	0.82
10	1	0.35	0.37	0.35	0.37
Average		0.69 $\pm$ 0.28	0.71 $\pm$ 0.24	0.72 $\pm$ 0.25	0.71 $\pm$ 0.28
Lookahead period = 10 min					
1	10	0.75	0.73	0.75	0.79
2	12	0.94	0.89	0.92	0.97
3	15	0.71	0.65	0.68	0.76
4	5	0.02	0.01	0.01	0.02
5	13	0.46	0.63	0.66	0.58
6	40	0.94	0.93	0.95	0.96
7	17	0.76	0.71	0.73	0.83
8	1	0.79	0.78	0.81	0.81
9	2	0.69	0.64	0.65	0.72
10	1	0.14	0.15	0.12	0.15
Average		0.62 $\pm$ 0.30	0.61 $\pm$ 0.28	0.63 $\pm$ 0.30	0.66 $\pm$ 0.31
Lookahead period = 15 min					
1	10	0.72	0.70	0.74	0.77
2	12	0.91	0.84	0.87	0.95
3	15	0.66	0.58	0.62	0.72
4	5	0.03	0.01	0.01	0.03
5	13	0.40	0.49	0.63	0.54
6	40	0.92	0.91	0.93	0.95
7	17	0.68	0.63	0.67	0.76
8	1	0.80	0.80	0.83	0.83
9	2	0.58	0.52	0.56	0.63
10	1	0.11	0.09	0.14	0.07
Average		0.58 $\pm$ 0.29	0.56 $\pm$ 0.29	0.60 $\pm$ 0.29	0.63 $\pm$ 0.31

Table 5.8: R^2 values for linear regression of airspace complexities prediction from four simulators by for each sectors for lookahead period of 5 minutes

Sector	**Average complexity**	**R^2 of simulator**			
		BS1	**BS2**	**BS3**	**BS4**
Sectors with the highest average complexity					
YBBB YMMM NULLARBOR A	32.70	0.97	0.97	0.99	0.99
YMMM BENALLA	28.35	0.97	0.96	0.98	0.99
YBBB INVERELL A	25.25	0.97	0.96	0.98	0.99
YBBB YMMM MUDGEE D	22.44	0.98	0.98	1.00	1.00
YBBB TERRITORY	22.23	0.92	0.91	0.92	0.93
YBBB YMMM ALICE SPRINGS	21.41	0.99	0.98	0.99	0.99
YBBB YMMM ISA	19.41	0.97	0.97	0.98	0.98
YBBB YMMM WARREGO	18.91	0.98	0.99	1.00	0.99
YMMM BILLABONG	18.40	0.97	0.96	0.97	0.98
YBBB YMMM KIMBERLEY	17.37	0.87	0.86	0.87	0.88
Sectors with average complexity around the median					
YMMM BINDOOK SYDNEY CAP	6.88	0.58	0.75	0.77	0.65
YBBB DARWIN ARRIVALS	6.70	0.00	0.08	0.08	0.01
YMMM WOLLONGONG A	6.58	0.77	0.64	0.68	0.80
YMMM WONTHAGGI A	6.34	0.75	0.59	0.59	0.78
YBBB YMMM NICKEL	6.06	0.55	0.03	0.00	0.49
YBBB KENNEDY C	5.86	0.73	0.70	0.72	0.75
YMMM AUGUSTA HIGH C	5.65	0.89	0.88	0.91	0.91
YBBB DOWNS A	5.55	0.67	0.72	0.76	0.70
YMMM TAILEM BEND B	5.53	0.90	0.90	0.95	0.96
YBBB GOLD COAST C	5.49	0.52	0.64	0.64	0.61
Sectors with the lowest average complexity					
YMMM EILDON WEIR HIGH A	3.27	0.21	0.50	0.57	0.33
YBBB BURNETT C	3.15	0.78	0.65	0.69	0.86
YBBB YMMM KATOOMBA A	2.28	0.32	0.36	0.28	0.60
YMMM WONTHAGGI E	2.26	0.37	0.14	0.35	0.37
YBBB BARRA C	2.26	0.68	0.56	0.60	0.73
YBBB GOLD COAST B	2.04	0.27	0.68	0.72	0.30
YBBB BARRA E	2.04	0.65	0.65	0.67	0.64
YMMM KATOOMBA SYDNEY CAP	1.73	0.23	0.43	0.30	0.39
YBBB YMMM MUDGEE A	1.66	0.32	0.04	0.07	0.50
YMMM AUGUSTA HIGH B	1.33	0.07	0.07	0.03	0.01
Overall average		0.70	0.70	0.73	0.75

Table 5.9: R^2 values for linear regression of airspace complexities prediction from four simulators by for each sectors for lookahead period of 10 minutes

Sector	**Average complexity**	**R^2 of simulator**			
		BS1	**BS2**	**BS3**	**BS4**
Sectors with the highest average complexity					
YBBB YMMM NULLARBOR A	32.70	0.97	0.97	0.99	0.99
YMMM BENALLA	28.35	0.95	0.93	0.95	0.98
YBBB INVERELL A	25.25	0.95	0.90	0.93	0.99
YBBB YMMM MUDGEE D	22.44	0.97	0.97	0.98	0.99
YBBB TERRITORY	22.23	0.88	0.86	0.89	0.90
YBBB YMMM ALICE SPRINGS	21.41	0.98	0.98	0.99	0.99
YBBB YMMM ISA	19.41	0.96	0.96	0.97	0.97
YBBB YMMM WARREGO	18.91	0.98	0.98	0.99	0.99
YMMM BILLABONG	18.40	0.97	0.95	0.96	0.98
YBBB YMMM KIMBERLEY	17.37	0.73	0.71	0.75	0.75
Sectors with average complexity around the median					
YMMM BINDOOK SYDNEY CAP	6.88	0.47	0.66	0.68	0.59
YBBB DARWIN ARRIVALS	6.70	0.00	0.01	0.02	0.00
YMMM WOLLONGONG A	6.58	0.66	0.50	0.53	0.72
YMMM WONTHAGGI A	6.34	0.59	0.41	0.44	0.63
YBBB YMMM NICKEL	6.06	0.72	0.07	0.17	0.72
YBBB KENNEDY C	5.86	0.61	0.58	0.62	0.64
YMMM AUGUSTA HIGH C	5.65	0.82	0.84	0.88	0.86
YBBB DOWNS A	5.55	0.57	0.57	0.65	0.61
YMMM TAILEM BEND B	5.53	0.85	0.83	0.92	0.96
YBBB GOLD COAST C	5.49	0.39	0.47	0.54	0.52
Sectors with the lowest average complexity					
YMMM EILDON WEIR HIGH A	3.27	0.12	0.29	0.66	0.28
YBBB BURNETT C	3.15	0.70	0.55	0.58	0.80
YBBB YMMM KATOOMBA A	2.28	0.04	0.10	0.23	0.30
YMMM WONTHAGGI E	2.26	0.88	0.71	0.70	0.90
YBBB BARRA C	2.26	0.68	0.51	0.47	0.69
YBBB GOLD COAST B	2.04	0.21	0.16	0.00	0.31
YBBB BARRA E	2.04	0.79	0.78	0.84	0.83
YMMM KATOOMBA SYDNEY CAP	1.73	0.01	0.36	0.00	0.50
YBBB YMMM MUDGEE A	1.66	0.18	0.49	0.02	0.34
YMMM AUGUSTA HIGH B	1.33	1.00	1.00	1.00	1.00
Overall average		0.62	0.63	0.66	0.75

Table 5.10: R^2 values for linear regression of airspace complexities prediction from four simulators by for each sectors for lookahead period of 15 minutes

Sector	**Average complexity**	**R^2 of simulator**			
		BS1	**BS2**	**BS3**	**BS4**
Sectors with the highest average complexity					
YBBB YMMM NULLARBOR A	32.70	0.97	0.97	0.99	0.99
YMMM BENALLA	28.35	0.94	0.88	0.91	0.97
YBBB INVERELL A	25.25	0.92	0.84	0.87	0.97
YBBB YMMM MUDGEE D	22.44	0.96	0.95	0.97	0.99
YBBB TERRITORY	22.23	0.85	0.82	0.88	0.89
YBBB YMMM ALICE SPRINGS	21.41	0.98	0.97	0.98	0.99
YBBB YMMM ISA	19.41	0.96	0.95	0.97	0.97
YBBB YMMM WARREGO	18.91	0.98	0.98	0.99	0.99
YMMM BILLABONG	18.40	0.96	0.95	0.96	0.98
YBBB YMMM KIMBERLEY	17.37	0.73	0.70	0.75	0.76
Sectors with average complexity around the median					
YMMM BINDOOK SYDNEY CAP	6.88	0.38	0.50	0.64	0.53
YBBB DARWIN ARRIVALS	6.70	0.00	0.01	0.02	0.00
YMMM WOLLONGONG A	6.58	0.32	0.29	0.30	0.39
YMMM WONTHAGGI A	6.34	0.42	0.32	0.34	0.44
YBBB YMMM NICKEL	6.06	1.00	0.15	0.15	1.00
YBBB KENNEDY C	5.86	0.57	0.52	0.57	0.60
YMMM AUGUSTA HIGH C	5.65	0.78	0.80	0.83	0.83
YBBB DOWNS A	5.55	0.52	0.45	0.56	0.55
YMMM TAILEM BEND B	5.53	0.77	0.69	0.81	0.94
YBBB GOLD COAST C	5.49	0.35	0.36	0.51	0.48
Sectors with the lowest average complexity					
YMMM EILDON WEIR HIGH A	3.27	0.13	0.12	0.39	0.28
YBBB BURNETT C	3.15	0.52	0.40	0.43	0.66
YBBB YMMM KATOOMBA A	2.28	0.17	0.22	0.27	0.06
YMMM WONTHAGGI E	2.26	0.94	0.77	0.75	0.95
YBBB BARRA C	2.26	0.61	0.40	0.40	0.62
YBBB GOLD COAST B	2.04	0.36	0.35	0.35	0.29
YBBB BARRA E	2.04	0.73	0.60	0.80	0.84
YMMM KATOOMBA SYDNEY CAP	1.73	0.91	0.00	0.00	0.70
YBBB YMMM MUDGEE A	1.66	0.00	0.11	0.21	0.00
YMMM AUGUSTA HIGH B	1.33	0.89	1.00	1.00	1.00
Overall average		0.59	0.57	0.64	0.67

Chapter 6

Conclusion

6.1 Summary of Results

In this thesis we investigated the effect of simulation fidelity, abstraction and resolution in the air traffic domain; more specifically airspace complexity.

We explored the role of fidelity in air traffic simulation by designing four air traffic simulators with differing levels of fidelity, abstraction and resolution. The lowest fidelity simulator (BS1) used fixed climb/descent rates and acceleration rates irrespective of the aircraft model of flight level. The second simulator (BS2), with a higher fidelity than BS1, used different climb rates and acceleration rates depending on the aircraft model and flight level. Both of these simulators used a simplified coordinate system to represent the Earth. The third (BS3) and fourth (BS4) simulators were extensions of the second and first simulators respectively. These two simulators used a more advanced, geodetic, representation of the Earth than the previous two simulators and are therefore of higher fidelity than BS1 and BS2. These four simulators were compared with a perfect model, ATOMS, which is assumed as the real world in our study. It was found that the assumptions used to design our four simulators resulted in some considerable differences. The biggest difference occurred due to the representation of the Earth. The simplified representation used by BS1 and BS2 resulted in large deviations from the flight tracks expected from the real world for the same flights plans, as can be seen in Figure 3.16. From Figure 3.16 we also see that BS3 and BS4, using the geodetic system, produced flight tracks which resulted in significantly smaller deviations from those expected from the real world. It was found that the direct turning methodology used by the low fidelity simulators also had some contribution to these deviations, as can be seen in Figure 3.9, since real world aircraft require some time to turn instead of the instantaneous turning used by these simulators. We found that using the BADA lookup tables to determine whether there is a need for acceleration (or deceleration) based on the flight level in BS2 and BS3 we were able to more closely match the speed profile of real world aircraft, as can be seen from Figure 3.8. This method allows for tailoring the speed profile based on the model of aircraft being simulated, as opposed to the continuous fixed acceleration used in BS1 and BS4 where the aircraft continue to accelerate towards its cruise speed irrespective of flight level. Despite these minor sources of deviations, it was concluded that the track deviations resulting from both BS3 and BS4 may be acceptable for predicting airspace complexity for at least 60 minutes ahead, while BS1 and BS2 may only be acceptable for predictions upto 15 minutes ahead.

We also developed a real-time airspace complexity adjustment system aimed at altering the expected workload of air traffic controllers (ATCs) in the short to mid term. A number of different shadow simulators were used as part of an optimisation system to generate a set of request, or actions, to be undertaken by aircraft with the aim of changing the future airspace complexity towards a predefined target level. The target level was altered based on three different lookahead periods: 15, 30 and 60 minutes. It was found in a fixed time frame for optimisation, using the perfect model (ATOMS) as the shadow simulator we were able to complete the lowest number of evaluations. This was followed by BS3 with the next fewest evaluations, then BS4, BS2 and BS1 with the most evaluations, as can be seen in Figure 4.9. This order was expected based on the computational complexity of these five models. Although it was possible to alter the future airspace complexity towards the predefined target levels using each of these five simulators as the shadow simulator, we found that the perfect model was able to outperform the other four remaining simulators. Of the four remaining simulators, there is on outright winner as each simulator performed better than the rest for different lookahead and target level combinations (see Table 4.6 and 4.8. It was found that the these four simulators generally overestimate the airspace complexity when compared to the real world (see Table 4.7). When compared to the real world complexity measurements, we found that BS2 and BS3 were able to more accurately predict the airspace complexity with BS3 being more accurate than BS2. This indicates that the higher the fidelity of the simulator the more accurate the complexity estimation. However, this does not necessarily result in a direct correlation with results obtained from the real-time optimisation system.

Finally we investigated some methods for predicting future airspace complexity. The first method was to generate a set of multiple linear regression models and to train a set of neural networks to predict the future airspace complexity using a summary of the current air traffic conditions. Second we generated a set of multiple linear regression models to predict the future airspace complexity using the airspace complexity characteristics at the current time along with a summary of the current air traffic conditions. These two methods were not able to produce models which adequately fit our data (see Table 5.1, 5.2 and 5.4) when compared to our third and final method. For our third method we used simulation to obtain an initial prediction of the airspace complexity at a future point in time. We then generated a linear regression model with this predicted airspace complexity and the measured real world complexity. It was found that we could generate models with a very good fit using data from every sector (see Table 5.6). We found that clustering the sectors based on the physical or traffic characteristics and generating a model for each clusters was not was successfully in all cases, but provided some good fits for a selection of the clusters (see Table 5.7). When generating a separate model for each sector we found that we can generate models with a good fit for the data for sectors with high traffic loads (see Table 5.8 to 5.10). Using this method we can make predictions about the future airspace complexity and adjust the prediction in order to minimise the deviations from what we expect to experience in reality.

6.2 Future Work

The work in this thesis has uncovered various questions and work which can be explored and carried out in the future. Some of these are outlined below.

As the airspace complexity adjustment system is implemented in a real-time environment, speed is the key. Other than altering the fidelity of the shadow simulators, in order to improve the speed of the optimisation component, we can investigate different configurations for the optimisation system. In our implementation we used the standard differential evolution algorithm. In the future we can investigate the implementation of the various alternate differential algorithms which have been developed with variations in the mutation, recombination and selection strategies. It is possible that the alternate algorithms may result in faster convergence speeds. In addition to alternate differential evolution algorithms we can also investigate the use of different evolutionary algorithms and other optimisation algorithms.

We can also combine the linear regression models used to adjust the airspace complexity prediction obtained from the low fidelity simulators with the real-time complexity adjustment system to obtain more accurate evaluations. In terms of the method used to adjust the prediction, we can breakdown the traffic conditions even further than just a general model for each sector. We can try to learn the typical traffic patterns in each sector and classify different patters in the sector. It is expected that different pattern will lead to different sources of deviations in the airspace complexity, so if can further breakdown the traffic patterns in each sector and produce a separate model for that classification it is likely that we will be able to predict the deviation with better accuracy.

Bibliography

Abbass, H. *et al.* (2002). The self-adaptive pareto differential evolution algorithm. In *Evolutionary Computation, 2002. CEC'02. Proceedings of the 2002 Congress on*, volume 1, pages 831–836. IEEE.

Abbass, H., Tucek, D., Kirby, S., and Ellejmi, M. (2013). Brain-traffic integration. *Air Traffic International*, pages 34–37.

Abbass, H. A. (2014). *Computational Red Teaming: Risk Analytics of Big-Data-to-Decisions Intelligent Systems*. Springer.

Abdi, H. and Williams, L. J. (2010). Principal component analysis. *Wiley Interdisciplinary Reviews: Computational Statistics*, **2**(4), 433–459.

Abou El Ela, A., Abido, M., and Spea, S. (2010). Differential evolution algorithm for emission constrained economic power dispatch problem. *Electric Power Systems Research*, **80**(10), 1286–1292.

Agogino, A. K. and Tumer, K. (2012). A multiagent approach to managing air traffic flow. *Autonomous Agents and Multi-Agent Systems*, **24**(1), 1–25.

Airbus (2014). Global marker forecast 2014-2033. Technical report.

Akerkar, R. and Sajja, P. (2010). *Knowledge-based systems*. Jones & Bartlett Publishers.

Alam, S., Abbass, H. A., and Barlow, M. (2008). Atoms: Air traffic operations and management simulator. *Intelligent Transportation Systems, IEEE Transactions on*, **9**(2), 209–225.

Archambault, N. (2005). Potentiality of computer-assisted speed regulations. *Laboratoire dOptimisation Globale, Toulouse, France*.

Avineri, E. (2012). Travel behavior research. *Artificial Intelligence Applications to Critical Transportation Issues*, page 77.

Baldi, P. and Hornik, K. (1989). Neural networks and principal component analysis: Learning from examples without local minima. *Neural networks*, **2**(1), 53–58.

Banks, J. (1998). *Handbook of Simulation: Principles, Methodology, Advances, Applications, and Practice*. A Wiley-Interscience publication. Wiley.

Bazargan, M., Lange, D., Tran, L., and Zhou, Z. (2013). A simulation approach to airline cost benefit analysis. *Journal of Management Policy and Practice*, **14**(2), 54–61.

Bebis, G. and Georgiopoulos, M. (1994). Feed-forward neural networks. *Potentials, IEEE*, **13**(4), 27–31.

Bekier, M., Molesworth, B. R., and Williamson, A. (2012). Tipping point: The narrow path between automation acceptance and rejection in air traffic management. *Safety Science*, **50**(2), 259 – 265.

Bertolini, M. and Bevilacqua, M. (2006). A combined goal programmingahp approach to maintenance selection problem. *Reliability Engineering & System Safety*, **91**(7), 839 – 848.

Bilimoria, K. D., Sridhar, B., Chatterji, G. B., Sheth, K. S., and Grabbe, S. (2001). Facet: Future atm concepts evaluation tool. *Air Traffic Control Quarterly*, **9**(1).

Böcker, J., Lind, J., and Zirkler, B. (2001). Using a multi-agent approach to optimise the train coupling and sharing system. *European Journal of Operational Research*, **131**(2), 242–252.

Brooks, R. J. and Tobias, A. M. (1996). Choosing the best model: Level of detail, complexity, and model performance. *Mathematical and computer modelling*, **24**(4), 1–14.

Calvert, J. B. (2002). Lambert's map. `https://mysite.du.edu/~jcalvert/math/lambert.htm`.

Carson, II, J. S. (2004). Introduction to modeling and simulation. In *Proceedings of the 36th Conference on Winter Simulation*, WSC '04, pages 9–16. Winter Simulation Conference.

Chaloulos, G., Crück, E., and Lygeros, J. (2010). A simulation based study of subliminal control for air traffic management. *Transportation research part C: Emerging technologies*, **18**(6), 963–974.

Chatterji, G. B. and Sridhar, B. (1999). Neural network based air traffic controller workload prediction. In *American Control Conference, 1999. Proceedings of the 1999*, volume 4, pages 2620–2624. IEEE.

Chen, B. and Cheng, H. (2010). A review of the applications of agent technology in traffic and transportation systems. *Intelligent Transportation Systems, IEEE Transactions on*, **11**(2), 485–497.

Clarke, J.-P. B., Solak, S., Ren, L., and Vela, A. E. (2012). Determining stochastic airspace capacity for air traffic flow management. *Transportation Science*, **47**(4), 542–559.

Coello, C. A. (2000). An updated survey of ga-based multiobjective optimization techniques. *ACM Comput. Surv.*, **32**(2), 109–143.

Coello, C. A. C. (1998). A comprehensive survey of evolutionary-based multiobjective optimization techniques. *Knowledge and Information Systems*, **1**, 269–308.

Cook, A., Blom, H. A., Lillo, F., Mantegna, R. N., Miccichè, S., Rivas, D., Vázquez, R., and Zanin, M. (2015). Applying complexity science to air traffic management. *Journal of Air Transport Management*, **42**, 149–158.

Cox, E. (1992). Fuzzy fundamentals. *Spectrum, IEEE*, **29**(10), 58–61.

Crück, E. and Lygeros, J. (2007a). Hybrid modeling for the evaluation of risk perception by air-traffic controllers. In *Control Conference (ECC), 2007 European*, pages 1278–1283. IEEE.

Crück, E. and Lygeros, J. (2007b). Subliminal air traffic control: Human friendly control of a multi-agent system. In *American Control Conference, 2007. ACC'07*, pages 462–467. IEEE.

Deb, K. (1999). Solving goal programming problems using multi-objective genetic algorithms. In *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on*, volume 1, pages 3 vol. (xxxvii+2348).

Deetz, C. H., Lambert, J. H., and Coast, U. (1918). *Lambert Conformal Conic Projection with 2 Standard Parallels, Including Comparison of Lambert Projection with Bonne and Polyconic Projections*. US Government Printing Office.

Dougherty, M. (1995). A review of neural networks applied to transport. *Transportation Research Part C: Emerging Technologies*, **3**(4), 247–260.

Drogoul, F., Averty, P., and Weber, R. (2009). Erasmus strategic deconfliction to benefit sesar. In *Proceedings of the 8th USA/Europe Air Traffic Management R&D Seminar*.

Dunne, J. P., John, J. G., Adcroft, A. J., Griffies, S. M., Hallberg, R. W., Shevliakova, E., Stouffer, R. J., Cooke, W., Dunne, K. A., Harrison, M. J., *et al.* (2012). Gfdl's esm2 global coupled climate-carbon earth system models. part i: Physical formulation and baseline simulation characteristics. *Journal of Climate*, **25**(19), 6646–6665.

Ehrmanntraut, R. and McMillan, S. (2007). Airspace design process for dynamic sectorisation. In *Digital Avionics Systems Conference, 2007. DASC'07. IEEE/AIAA 26th*, pages 3–D. IEEE.

Eiben, A. E. and Smith, J. E. (2003). *Introduction to evolutionary computing*. Springer.

Erbao, C. and Mingyong, L. (2009). A hybrid differential evolution algorithm to vehicle routing problem with fuzzy demands. *Journal of computational and applied mathematics*, **231**(1), 302–310.

Eurocontrol (2004). User manual for the base of aircraft data (bada), revision 3.6. *EEC note*, (10/04).

FAA (2014). Nextgen programs. `https://www.faa.gov/nextgen/programs/`.

Fishwick, P. A. (1988). The role of process abstraction in simulation. *Systems, Man and Cybernetics, IEEE Transactions on*, **18**(1), 18–39.

Frantz, F. K. (1995). A taxonomy of model abstraction techniques. In *Proceedings of the 27th conference on Winter simulation*, pages 1413–1420. IEEE Computer Society.

Frantz, F. K. and Ellor, A. J. (1996). Model abstraction techniques. Technical report, DTIC Document.

GA, D. and Okdem, S. (2004). A simple and global optimization algorithm for engineering problems: differential evolution algorithm. *Turk J Elec Engin*, **12**(1).

Galster, S. M., Duley, J. A., Masalonis, A. J., and Parasuraman, R. (2001). Air traffic controller performance and workload under mature free flight: Conflict detection and resolution of aircraft self-separation. *The International Journal of Aviation Psychology*, **11**(1), 71–93.

Gämperle, R., Müller, S. D., and Koumoutsakos, P. (2002). A parameter study for differential evolution. *Advances in intelligent systems, fuzzy systems, evolutionary computation*, **10**, 293–298.

Gilbert, G. *et al.* (1973). Historical development of the air traffic control system. *Communications, IEEE Transactions on*, **21**(5), 364–375.

Glasa, J. (2009). Computer simulation and predicting dangerous forest fire behaviour. *International Journal of Mathematics and Computer in Simulation*, (2).

Glover, W. and Lygeros, J. (2004). A stochastic hybrid model for air traffic control simulation. In *Hybrid Systems: Computation and Control*, pages 372–386. Springer.

Goncalves, D. (2006). Approach to simulation effectiveness.

Hancock, P., Vincenzi, D., Wise, J., and Mouloua, M. (2008). *Human Factors in Simulation and Training*. CRC Press.

Hatala, R., Issenberg, S. B., Kassen, B., Cole, G., Bacchus, C. M., and Scalese, R. J. (2008). Assessing cardiac physical examination skills using simulation technology and real patients: a comparison study. *Medical education*, **42**(6), 628–636.

He, D., Wang, F., and Mao, Z. (2008). A hybrid genetic algorithm approach based on differential evolution for economic dispatch with valve-point effect. *International Journal of Electrical Power & Energy Systems*, **30**(1), 31–38.

Heidt, A. and Gluchshenko, O. (2012). From uncertainty to robustness and systems resilience in atm: a case study. In *Proceedings of the Third International Air Transport and Operations Symposium*, pages 421–426.

Henesey, L. (2004). A multi agent based simulator for managing a container terminal. In *Proceedings of the 2nd European Workshop on Multi-Agent Systems (EUMAS 2004), Barcelona, Spain. December*, volume 16, pages 291–302.

Hernández, J. Z., Ossowski, S., and Garcıa-Serrano, A. (2002). Multiagent architectures for intelligent traffic management systems. *Transportation Research Part C: Emerging Technologies*, **10**(5), 473–506.

Hilburn, B. (2004). Cognitive complexity in air traffic control: A literature review. *EEC note*, **4**(04).

Hu, C.-F., Teng, C.-J., and Li, S.-Y. (2007). A fuzzy goal programming approach to multi-objective optimization problem with priorities. *European Journal of Operational Research*, **176**(3), 1319 – 1333.

Hughes, T. and Rolek, E. (2003). Human systems modeling: fidelity and validity: issues of human behavioral representation requirements development. In *Proceedings of the 35th conference on Winter simulation: driving innovation*, pages 976–982. Winter Simulation Conference.

ICAO (2004). Outlook for Air Transport to the Year 2015. Technical Report Cir 304 AT/127, International Civil Aviation Organisation, Montreal, Canada.

Inoue, S., Furuta, K., Nakata, K., Kanno, T., Aoyama, H., and Brown, M. (2012). Cognitive process modelling of controllers in en route air traffic control. *Ergonomics*, **55**(4), 450–464.

Issenberg, S., Petrusa, E., McGaghie, W., Felner, J., Waugh, R., Nash, I., and Hart, I. (1999). Effectiveness of a computer-based system to teach bedside cardiology. *Academic Medicine*, **74**(10), S93–5.

Jacobs, J. W. and Dempsey, J. V. (1993). Simulation and gaming: Fidelity, feedback and motivation. *Interactive instruction and feedback*, pages 197–227.

Jordan, M. I. and Rumelhart, D. E. (1992). Forward models: Supervised learning with a distal teacher. *Cognitive science*, **16**(3), 307–354.

Justesen, P. D. (2009). Multi-objective optimization using evolutionary algorithms. *University of Aarhus, Department of Computer Science, Denmark.*

Kaufman, L. and Rousseeuw, P. J. (2009). *Finding groups in data: an introduction to cluster analysis*, volume 344. John Wiley & Sons.

Kicinger, R., Arciszewski, T., and Jong, K. (2005). Evolutionary computation and structural design: A survey of the state-of-the-art. *Computers & Structures*, **83**(23), 1943–1978.

Kim, Y. J., Pinon-Fischer, O. J., and Mavris, D. N. (2015). Parallel simulation of agent-based model for air traffic network. In *AIAA Modeling and Simulation Technologies Conference*, page 2799.

Kopardekar, P. H., Schwartz, A., Magyarits, S., and Rhodes, J. (2009). Airspace complexity measurement: An air traffic control simulation analysis. *International Journal of Industrial Engineering: Theory, Applications and Practice*, **16**(1), 61–70.

Kotsiantis, S. B., Zaharakis, I., and Pintelas, P. (2007). Supervised machine learning: A review of classification techniques.

Kuchar, J. K. and Yang, L. C. (2000). A review of conflict detection and resolution modeling methods. *Intelligent Transportation Systems, IEEE Transactions on*, **1**(4), 179–189.

Laudeman, I. V., Shelden, S., Branstrom, R., and Brasil, C. (1998). Dynamic density: An air traffic management metric. Technical report, National Aeronautics and Space Administration, Ames Research Center.

Lee, K., Feron, E., and Pritchett, A. (2009). Describing airspace complexity: Airspace response to disturbances. *Journal of Guidance, Control, and Dynamics*, **32**(1), 210–222.

Lee, L. H., Huang, H. C., Lee, C., Chew, E. P., Jaruphongsa, W., Yong, Y. Y., Liang, Z., Leong, C. H., Tan, Y. P., Namburi, K., *et al.* (2003). Simulation of airports/aviation systems: discrete event simulation model for airline operations: Simair. In *Proceedings of the 35th conference on Winter simulation: driving innovation*, pages 1656–1662. Winter Simulation Conference.

Lee, S. M. *et al.* (1972). *Goal programming for decision analysis*. Auerbach Philadelphia.

Loft, S., Sanderson, P., Neal, A., and Mooij, M. (2007). Modeling and predicting mental workload in en route air traffic control: Critical review and broader implications. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, **49**(3), 376–399.

Majumdar, A. and Ochieng, W. (2002). Factors affecting air traffic controller workload: Multivariate analysis based on simulation modeling of controller workload. *Transportation Research Record: Journal of the Transportation Research Board*, (1788), 58–69.

Mallipeddi, R., Suganthan, P., Pan, Q., and Tasgetiren, M. (2011). Differential evolution algorithm with ensemble of parameters and mutation strategies. *Applied Soft Computing*, **11**(2), 1679 – 1696. The Impact of Soft Computing for the Progress of Artificial Intelligence.

Maria, A. (1997). Introduction to modeling and simulation. In *Proceedings of the 29th conference on Winter simulation*, pages 7–13. IEEE Computer Society.

MathWorks (2015a). Convex hull. `http://au.mathworks.com/help/matlab/ref/convhull.html`.

MathWorks (2015b). k-medoids clustering. `http://au.mathworks.com/help/stats/kmedoids.html`.

MathWorks (2015c). Multiple linear regression. `http://au.mathworks.com/help/stats/regress.html`.

MathWorks (2015d). Train neural network. `http://au.mathworks.com/help/nnet/ref/train.html`.

Menard, S. (2000). Coefficients of determination for multiple logistic regression analysis. *The American Statistician*, **54**(1), 17–24.

Menon, P. K., Sweriduk, G. D., and Bilimoria, K. D. (2004). New approach for modeling, analysis, and control of air traffic flow. *Journal of guidance, control, and dynamics*, **27**(5), 737–744.

Mingyong, L. and Erbao, C. (2010). An improved differential evolution algorithm for vehicle routing problem with simultaneous pickups and deliveries and time windows. *Engineering Applications of Artificial Intelligence*, **23**(2), 188–195.

Mogford, R. H., Guttman, J., Morrow, S., and Kopardekar, P. (1995). The complexity construct in air traffic control: A review and synthesis of the literature. Technical report, DTIC Document.

Moon, I.-C. and Hong, J. H. (2013). Theoretic interplay between abstraction, resolution, and fidelity in model information. In *Proceedings of the 2013 Winter Simulation Conference: Simulation: Making Decisions in a Complex World*, pages 1283–1291. IEEE Press.

Moré, J. J. (1978). The levenberg-marquardt algorithm: implementation and theory. In *Numerical analysis*, pages 105–116. Springer.

Nikoletseas, S., of Electrical, I., Engineers, E., Chlebus, B., Johnson, D., and Krishnamachari, B. (2008). *Distributed Computing in Sensor Systems: 4th IEEE International Conference, DCOSS 2008 Santorini Island, Greece, June 11-14, 2008, Proceedings*. LNCS sublibrary: Computer communication networks and telecommunications. Springer.

Nolan, M. (2010). *Fundamentals of air traffic control*. Cengage Learning.

Norman, G., Dore, K., and Grierson, L. (2012). The minimal relationship between simulation fidelity and transfer of learning. *Medical Education*, **46**(7), 636–647.

Ozeki, S. (2014). *Air Traffic Management and Systems: Selected Papers of the 3rd ENRI International Workshop on ATM/CNS (EIWAC2013)*. Lecture Notes in Electrical Engineering. Springer.

Pachepsky, Y. (2006). *Model abstraction techniques for soil-water flow and transport*. Division of Fuel, Engineering & Radiological Research, Office of Nuclear Regulatory Research, US Nuclear Regulatory Commission.

Paielli, R. A., Erzberger, H., Chiu, D., and Heere, K. R. (2009). Tactical conflict alerting aid for air traffic controllers. *Journal of Guidance, Control, and Dynamics*, **32**(1), 184–193.

Phillips, M. and Marsh, D. (2000). The validation of fast-time air traffic simulations in practice. *Journal of the Operational Research Society*, pages 457–464.

Prandini, M., Piroddi, L., Puechmorel, S., and Brázdilová, S. L. (2011). Toward air traffic complexity assessment in new generation air traffic management systems. *Intelligent Transportation Systems, IEEE Transactions on*, **12**(3), 809–818.

Qin, A., Huang, V., and Suganthan, P. (2009). Differential evolution algorithm with strategy adaptation for global numerical optimization. *Evolutionary Computation, IEEE Transactions on*, **13**(2), 398 –417.

Quist, R. (2011). Common map projections. `http://www.gislounge.com/common-map-projections/`.

Rao, L., Owen, L., and Goldsman, D. (1998). Development and application of a validation framework for traffic simulation models. In *Proceedings of the 30th conference on Winter simulation*, pages 1079–1086. IEEE Computer Society Press.

Rawlings, J. B. and Muske, K. R. (1993). The stability of constrained receding horizon control. *Automatic Control, IEEE Transactions on*, **38**(10), 1512–1516.

Reshetin, V. P. and Regens, J. L. (2003). Simulation modeling of anthrax spore dispersion in a bioterrorism incident. *Risk Analysis*, **23**(6), 1135–1145.

Rey, D., Rapine, C., Dixit, V. V., and Waller, S. T. (2015). Equity-oriented aircraft collision avoidance model. *Intelligent Transportation Systems, IEEE Transactions on*, **16**(1), 172–183.

Rodriguez, J. F. (2008). Metamodeling techniques to aid in the aggregation process of large hierarchical simulation models. Technical report, DTIC Document.

Rogalski, S. (2012). Factory design and process optimisation with flexibility measurements in industrial production. *International Journal of Production Research*, **50**(21), 6060–6071.

Rönkkönen, J., Kukkonen, S., and Price, K. (2005). Real-parameter optimization with differential evolution. volume 1, pages 506–513. cited By 154.

Sadek, A. W. (2007). Artificial intelligence applications in transportation. *Transportation Research Circular E-C113*, pages 1–6.

Santosa, B., Sunarto, A., and Rahman, A. (2010). Using differential evolution method to solve crew rostering problem. *Applied Mathematics*, **1**(4), 316–325.

Sargent, R. G. (2005). Verification and validation of simulation models. In *Proceedings of the 37th conference on Winter simulation*, pages 130–143. Winter Simulation Conference.

Sarker, R., Elsayed, S. M., Ray, T., *et al.* (2014). Differential evolution with dynamic parameters selection for optimization problems. *Evolutionary Computation, IEEE Transactions on*, **18**(5), 689–707.

SESARJU (2015). Sesar solutions. `http://www.sesarju.eu/solutions`.

Sewall, J., Wilkie, D., Merrell, P., and Lin, M. C. (2010). Continuum traffic simulation. In *Computer Graphics Forum*, volume 29, pages 439–448. Wiley Online Library.

Sewall, J., Wilkie, D., and Lin, M. C. (2011). Interactive hybrid simulation of large-scale traffic. In *ACM Transactions on Graphics (TOG)*, volume 30, page 135. ACM.

Simić, T. K. and Babić, O. (2015). Airport traffic complexity and environment efficiency metrics for evaluation of atm measures. *Journal of Air Transport Management*, **42**, 260–271.

Sisti, A. and Farr, S. (1998). Model abstraction techniques: an intuitive overview. In *Aerospace and Electronics Conference, 1998. NAECON 1998. Proceedings of the IEEE 1998 National*, pages 447–450.

Sokolowski, J. and Banks, C. (2010). *Modeling and Simulation Fundamentals: Theoretical Underpinnings and Practical Domains*. Wiley.

Sokolowski, J. and Banks, C. (2011). *Principles of Modeling and Simulation: A Multidisciplinary Approach*. Wiley.

Song, Z., Chen, Y., Li, Z., Zhang, D., and Bi, H. (2012). Measurement of controller workloads based on air traffic complexity factors. In *CICTP 2012@ sMultimodal Transportation SystemsConvenient, Safe, Cost-Effective, Efficient*, pages 1903–1914. ASCE.

Spišák, M. and Šperka, R. (2011). Financial market simulation based on intelligent agents–case study. *Journal of Applied Economic Sciences (JAES)*, (3 (17), 249–256.

Stanica, R., Chaput, E., and Beylot, A.-L. (2011). Simulation of vehicular ad-hoc networks: Challenges, review of tools and recommendations. *Computer Networks*, **55**(14), 3179–3188.

Stein, E. S., Della Rocco, P. S., and Sollenberger, R. L. (2006). Dynamic resectorization in air traffic control: A human factors perspective. Technical Report DOT/FAA/TC-TN06/19, Atlantic City International Airport, NJ: Federal Aviation Administration William J. Hughes Technical Center.

Storn, R. and Price, K. (1997). Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*, **11**(4), 341–359.

Sun, D., Strub, I. S., and Bayen, A. M. (2007). Comparison of the performance of four eulerian network flow models for strategic air traffic management. *Networks and Heterogeneous Media*, **2**(4), 569.

Tang, J. (2012). *Dynamic Sectorization of air traffic using multi-objective clustering*. Ph.D. thesis, School of Engineering and Information Technologu, University of New South Wales, Canberra.

Teyssier, R. (2015). Cosmological radiative transfer+ hydro+ n-body simulation of galaxy formation and the intergalactic medium: The radiation-hydro frontier. *Bulletin of the American Physical Society*, **60**.

Tumer, K. and Agogino, A. (2007). Distributed agent-based air traffic flow management. In *Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems*, page 255. ACM.

Villiers, J. (2004). Automatisation du contrôle de la circulation aérienne-projet erasmus une voie originale pour mieux utiliser lespace aérien. Technical report, Technical report, Institut de Transport Aérien.

Vogel, M., Schelbert, K., Fricke, H., and Kistan, T. (2013). Analysis of airspace complexity factors capability to predict work-load and safety levels in the tma. In *10th USA/Europe ATM R&D Seminar, Chicago, USA*.

Volf, P., Šišlák, D., and Pěchouček, M. (2011). Large-scale high-fidelity agent-based simulation in air traffic domain. *Cybernetics and Systems*, **42**(7), 502–525.

Wang, F.-Y. (2005). Agent-based control for networked traffic management systems. *Intelligent Systems, IEEE*, **20**(5), 92–96.

Weiß, G. (1999). *Mutiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. Intelligent Robotics and Autonomous Agents Series. Mit Press.

Wise, J., Hopkin, V., and Smith, M. (2012). *Automation and Systems Issues in Air Traffic Control*. Nato ASI Subseries F:. Springer Berlin Heidelberg.

Yilmaz, L. and Ören, T. (2009). *Agent-Directed Simulation and Systems Engineering*. Wiley Series in Systems Engineering and Management. Wiley.

Zaporozhets, O. I. and Tokarev, V. I. (1998). Aircraft noise modelling for environmental assessment around airports. *Applied Acoustics*, **55**(2), 99–127.

Zhang, W., Kamgarpour, M., Sun, D., and Tomlin, C. J. (2012). A hierarchical flight planning framework for air traffic management. *Proceedings of the IEEE*, **100**(1), 179–194.