School of Mechanical and Manufacturing Engineering

Faculty of Engineering

Undergraduate Thesis

Automating Component Selection in Conceptual Design

Aidan O'Brien

June 2015

Thesis submitted as a requirement for the degree of Bachelor of Engineering in Mechatronic Engineering

Supervisor: Dr. Jason Held

Certificate of Originality

Abstract

When creating a new product, whether for sale or a specific purpose, a design process must occur. Conceptual design is often a creative and/or innovative process which establishes an initial set of design ideas. The ideas are compared against one another and a the preferred design is then refined for a final design. The creation of ideas and the comparisons between them are based upon customer requirements, which are often vague or arbitrarily assigned. New automation methods for the detailed design stage have shown benefits for the optimisation of the final product, however few automated tools exist for conceptualisation.

This thesis demonstrates the possibility of automating conceptual design. It utilises fuzzy logic to encode customer requirements to allow a comparison to be made between what the customer desires and the performance of a generated design. Conceptual designs are generated with a genetic algorithm which creates product specifications from components.

Two sets of experiments were conducted in this thesis, the first explored the effects caused by changing customer requirements. This resulted in showing that by changing customer requirements based upon one design can change design categories and have a greater similarity to designs in different categories. This allows comparison of a conceptual design against a greater range of past designs. The second set of experiments created conceptual designs with an evolutionary algorithm. These were conducted to demonstrate the potential for automating the conceptual design process. Results show that it is possible for evolutionary algorithms to develop conceptual designs.

Acknowledgements

Incredible thanks go to Jason Held, my supervisor and friend, whose boundless optimism in everything shines through, who had the confidence in me to believe that this topic was actually possible.

A big thanks goes towards my wonderful partner in crime, Christina for supporting me, the movies, the Star Wars and other sci-fi marathons, dragging me out to events and dealing with me randomly trailing off mid-sentence to write something down in my notebook.

To my parents, who have always been there, unconditionally supporting and encouraging me to improve myself.

Contents

List of Figures

List of Tables

Abbreviations

AI	**A**rtificial **I**ntelligence
CACD	**C**omputer **A**ided **C**onceptual **D**esign
MOPO	**M**ulti **O**bjective **P**areto **O**ptimisation
MOEA(s)	**M**ulti **O**bjective **E**volutionary **A**lgorithm(s)
CR	**C**ustomer **R**equirements
PS	**P**roduct **S**pecifications
NSGA-II	**N**ondominated **S**orted **G**enetic **A**lgorithm**-II**
ART	**A**daptive **R**esonance **T**heory
FAM	**F**uzzy **A**RT**M**AP
F-B-S	**F**unctional-**B**ehavioural-**S**tructural ontology
Be	**B**ehaviour **e**xpected
Bs	**B**ehaviour derived from **s**tructure
ADCS	**A**ttitude **D**etermination and **C**ontrol **S**ystem
INMS	**I**on and **N**eutral **M**ass **S**pectrometer
FIPEX	**F**lux-ϕ-**P**robe-**Ex**periment
bps	**b**aud **p**er **s**econd
VEGA	**V**ector **E**valuated **G**enetic **A**lgorithm
EPS	**E**lectrical **P**ower **S**ystem

Chapter 1

Introduction

"There is a popular cliche ... which says that you cannot get out of computers any more than you put in. Other versions are that computers only do exactly what you tell them to, and that therefore computers are never creative. The cliche is true only in the crashingly trivial sense, the same sense in which Shakespeare never wrote anything except what his first schoolteacher taught him to write–words." [1] - ***Richard Dawkins, The Blind Watchmaker***

Conceptual design is an important stage in the creation of a new product. Whether the new product is entirely original or an iterative improvement of a past design, conceptual design is a critical stage in the engineering process. Conceptual design is mostly a creative endeavour and creative processes are considered the hardest for a computer to replicate and replace. Conceptual design has formalised tools and methods such as morphological analysis and pugh selection matrices, however these require experience to implement and pugh selection matrices utilise arbitrarily assigned weights to compare performances [2]. These tools and methods are used to generate new designs, or evaluate multiple potential designs against one another. Most of the computerised tools however are used in the later stage of the design process, once a particular design has been chosen to be developed further and often optimise the part or product being designed. There are minimal automated tools for the early stages of conceptual design and there is little automated assistance given to engineers in generating new and creative designs. Conceptual designs have been created by computers in the past, but only for single purpose tasks, such as designing a component or generating the ideal proportions of an I-beam for weight bearing tasks [3].

This thesis investigates the possibility of an early stage conceptual design that can be generated autonomously from customer requirements. In recent years, machines have shown themselves capable of decision making based on learning rather than predetermined programming. This ranges from cars being able to drive themselves, identifying specific objects in images that the computer has not been exposed to before. The automation of decision making can be achieved once the specific algorithm relevant to the task is discovered. In machines, mathematical representations and probability allow for these actions to be learned and developed. People can be trained to understand the design process and where there is something that can be trained, a machine can be programmed for it.

1.1 Engineering design, a case study using small satellites

Studies have shown that creating a new design is time consuming and the majority of the design process is at the beginning, during conceptualisation [4]. Formalised methods of conceptual design involve techniques such as morphology matrices and design catalogues, which are combined in conceptual design and then analysed [5]. Later sections of the design process, for example detailed design where a design chosen during conceptual design is developed further, have been improved and streamlined through the use of computers, such as finite element analysis and 3D CAD programs. Evolutionary programming has been utilised to create designs for individual components of a satellite, structures for bridges and to solve a number of other engineering design problems, often resulting in designs that appear to be unintuitive, but analysis shows to be optimal [3, 6].

Every engineering discipline utilises conceptual design, because it is a critical task for creating new and innovative designs. Disciplines such as automative or naval engineering are established fields, with a great deal of maturity between designs. Space, however, is a frontier that has become a growth industry with companies such as SpaceX, crowd sourced satellites such as ArduSat, shown in Fig 1.1, and university projects. This thesis focuses upon the design of CubeSats because they have certain standards that are partially established, while being new enough to need innovative designs to fulfill different missions. Being a growth industry, CubeSats also have a growing number of sub-systems and components commercially available which makes them a perfect candidate for testing a range of conceptual designs.

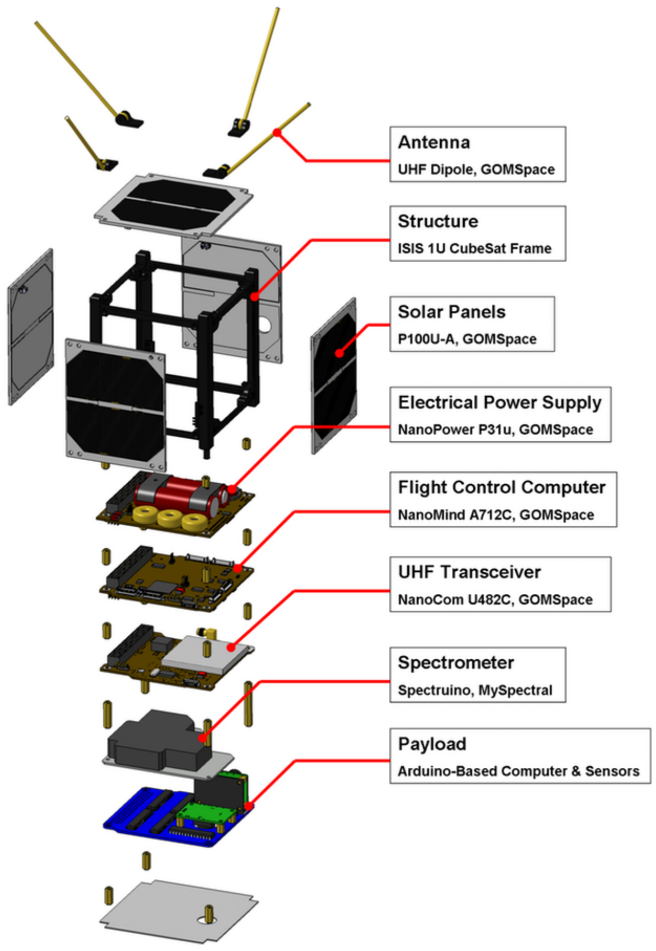


Figure 1.1: Exploded view of ArduSat CubeSat. From: ArduSat Kickstater. [7]

CubeSats are a small form factor satellite, which allow universities and private enterprises to directly access the space segment by reducing entry costs, this has resulted in over sixty universities and high schools becoming involved in the CubeSat program [8]. Partially because CubeSats provide cheap entry into space for remote sensing and other scientific missions [33]. The CubeSat standard has a range of sizes based upon the core '1U' standard. A 1U CubeSat is a 10cmx10cmx1cm cube, with guide rails to fit within the deployment mechanism. 2U and larger versions of CubeSats are created by stacking 1U dimensions, resulting in a 10cmx10cmx20cm cuboid [9]. The CubeSat market has been growing at over 38 percent and is estimated to continue growing by 28 percent annually, this would result in a $2 billion industry by 2020 [10].

This standardisation of a cheap satellite design has allowed for experimentation in both payloads and scientific missions with a large number of satellites in a constellation such as the QB50 project. The QB50 mission is to create a network of fifty satellites to perform "in-situ measurements in the lower thermosphere and re-entry research" [11]. This thesis is utilising CubeSats as a dataset to demonstrate the potential of artificial intelligence algorithms to provide assistance in the conceptual design phase. The final test of this thesis utilises the customer requirements for the QB50 project and creates a satellite from commercially available components.

1.2 Methods

Automating the process of conceptual design requires replicating the capabilities provided by conceptualisation methods, including morphological matrices and pugh selection. Morphological matrices work by mix and matching components, which can result in a large number of design parameters, while pugh selection is a method to compare different designs against one another by weighing different design objections, to aid in the selection of a design to optimise and develop for a final product. These capabilities have equivalents in evolutionary programming, which mutates solutions to introduce different components and have a fitness value to compare solutions against one another.

One characteristic of developing evolutionary programs is to establish rules for evolution. Depending on the intended purpose of an evolutionary program, different rule sets are used and within engineering design, a design methodology is required. This thesis uses the Function-Behaviour-Structure ontology as its design methodology, because it has a clear delineation between design levels and uses linkages between each level that can be quantified and automated.

Fuzzy logic is utilised to quantify customer requirements and allow a direct comparison with product specifications. To ensure that the fuzzy logic works as intended, a neural network is utilised to classify permutations of customer requirements and demonstrate the similarities between different designs and how changing customer requirements can change the categorisation that a satellite belongs to. The neural network will fail if modified customer requirements are dramatically different to any past satellite in the database which allows a secondary check that the fuzzification of natural language requirements to numerical values is a valid assumption.

The behaviour of a given design is quantified using metrics. These metrics are scaled to ensure that a particular value is matched with their respective customer requirement value. Then the similarity between the metrics and the fuzzified customer requirements is utilised within the fitness function of a genetic algorithm to evaluate the performance of generated satellite designs. Finally, the competing nature of different design objectives is considered within the genetic algorithm and is used to find a potential solution to the design problem. When multiple objectives compete against one another, it becomes a 'pareto-optimal' problem, which is discussed in depth in chapter 2.

1.2.1 Experimentation nomenclature

This thesis utilises two artificial intelligence methods in the design process. The first method is a neural network variation called Fuzzy ARTMAP and explained in depth in section 2.2.4. Neural networks are often utilised for pattern recognition [28]. The first step of any neural network is the 'training' process where a 'map' is created by the neural network algorithm. The data set utilised to create this map is the 'training set' and often created from historical data. The second stage of a neural network is the classification of test data. Often the test data, frequently referred to as a 'test set', is a subset of available historical data and is not utilised within the training set. In cases where small amounts of historical data are available, permutations of the training set are created for testing purposes. Both the training set and the test set will have a known correct answer in ideal circumstances. In cases where the test set is created by permutations, there is no guarantee that the assigned 'correct' answer is the correct result. The final stage of a neural network is utilising new data where the neural network attempts to classify new inputs into existing categories.

The second method of artificial intelligence used in this thesis is a genetic algorithm, which is part of a family of "stochastic search methods, inspired by natural evolution" [50]. Genetic algorithms all share certain traits, such as the population, mutation rate and generations. The population is the number of individual solutions that the algorithm is currently keeping track of. This population will ideally change and optimise over time, although a common flaw in genetic algorithms is becoming 'stuck' in local optima and are incapable of finding the global optimum. The mutation rate is a variable that determines the probability of a new variation in an individual becoming present. The generations of a genetic algorithm defines the number of times a population creates offspring and the highest performing individuals are selected for the next generation's population. The primary output of a genetic algorithm is the population that had evolved by the final generation.

1.2.2 Fuzzy logic classification

The customer requirement space is multi-dimensional as is normal in pareto-optimal problems, however, conceptually it can be visualised in two dimensions as shown in Fig. 1.2. Each satellite in the training set, can be treated as a category of satellite. In Fig. 1.2, a category is represented by one of the ovals, while individual customer requirements are represented by black X's.

Each of the categories shown in Fig. 1.2 shares some customer requirements with other categories, this is normal in similar designs such as CubeSats. Not all possible com-

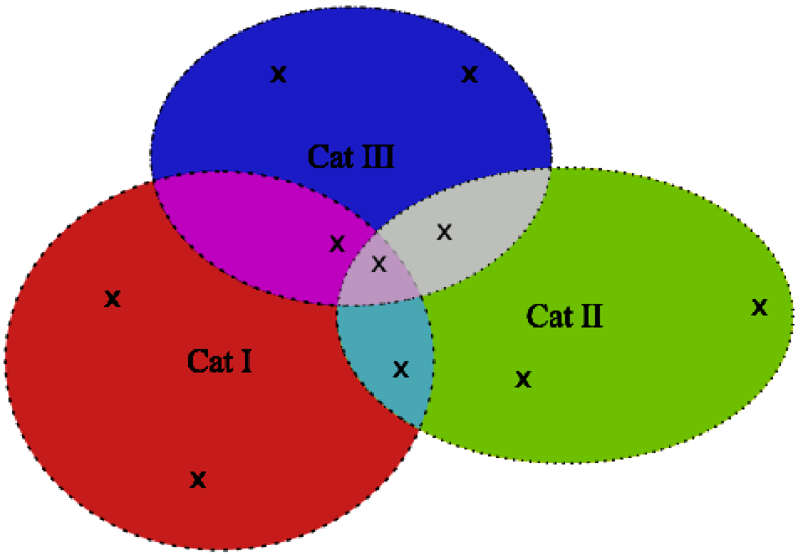


Figure 1.2: Visualisation of customer requirement space

binations are shown in the figure, nor are cases where there is no overlap of customer requirements. The fuzzy logic experiments are examining how changing customer requirements changes the categorisations of a satellite. The process of changing customer requirements is shown in Fig. 1.3, where a customer requirement in Cat III is modified through customer requirement space until it is located in the area dominated by Cat II satellites. This is represented by the white arrow and the final location is the white X.

The classification of what category of satellite a set of customer requirements belongs to shows the similarity to past designs, which allows a designer to compare designs with past solutions which may be for similar missions, or operate in a similar environment. The classification of customer requirements also allows for the identification of critical points, points where a set of customer requirements has equal similarity with two categories. The additional purpose of these tests is to examine whether classification occurs for the training set of satellites, a result where a satellite is unable to be classified would imply the customer requirements are so different to any of the satellites in the training set that a new category would have been found.

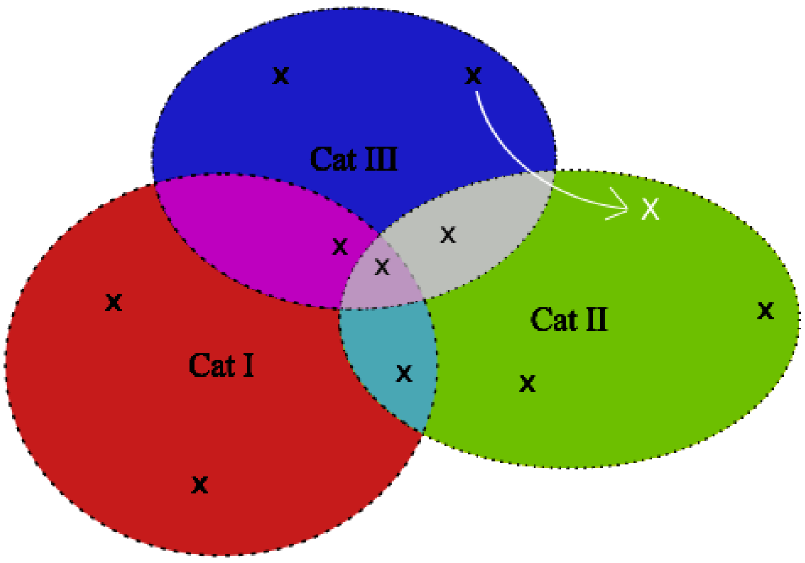


Figure 1.3: Visualisation of modifying CRs in customer requirement space

1.3 Thesis structure

- **Chapter 2** provides an overview of conceptual design, design theory and the background for the specific machine learning and artificial intelligence techniques being used. In particular, neural networks and genetic algorithms.
- **Chapter 3** explains the steps taken to quantify customer requirements, the creation of metrics for product specifications and explains the process to develop the genetic algorithm that utilises these processes to create potential solutions and returns a population of satellites.
- **Chapter 4** provides an explanation of the experimental method and the results of each experiment. The experiments prove the validity of assumptions about fuzzy logic and show the output of the genetic algorithm in comparison to human created designs.
- **Chapter 5** makes conclusions based out the output of the algorithms and experiments and discusses future work to be done in this thesis.

Chapter 2

Literature Review

2.1 Introduction

This chapter discusses conceptual design and tools to improve the design process. The Functional-Behavioural-Structural (FBS) ontology provides a framework for other design tools. The automation of the design process requires two major procedures to be conducted to fit within the FBS ontology. The first is the quantification of customer requirements and evaluating the quantification, the second is creating designs and associating them with the customer requirements. These processes form the foundation of automating the conceptual design stage.

2.2 Conceptual design

The current standard procedure in conceptual design relies upon the capability of people, either individually or in groups, to create new ideas in the process of design [12]. Conceptual design is one of an important series of steps that make up the early design process. This part of the design process is sometimes referred to as the "fuzzy front end", due to the way that the early design process is unpredictable in its nature. Studies have shown that this fuzzy front end takes up to eighty percent of the design process's duration [4], this large a time period enables opportunities to find methods and tools to improve the efficiency of the design process. It is common knowledge that there is a limit to possible designs for a given problem, bound by the laws of physics, the set of possible designs is often referred to as the 'design space'. This design space is further limited by reasonable limits in technology and resources that can be dedicated to the problem. Design space is large and can be time consuming to extensively explore [4]. Various methods for exploring design spaces have been proposed, including emergent algorithms and heuristic-based approaches [4, 13].

Often, "good enough" principles have to be utilised in design to balance constraints to the design. This makes it possible to convert design problems into multi-objective optimisation problems. Under certain conditions it is possible to solve the constraint problem by forming a Pareto front as explained in section 2.2.3, however, it can be difficult to solve a multi-modal problem due to rising complexity [14].

2.2.1 Forms of conceptual design

There are two main divisions in conceptual design. The first is routine conceptual design, which is the logical progression from an existing design to a later design. The second is non-routine and can be further subdivided. Two main sub-divisions for non-routine conceptual design are "innovative" design and "creative" design [12, 13]. Innovative and creative design methods differ in the way they generate conceptually new designs. Innovative design uses the same base knowledge of the designer, but removes constraints on previous designs to find new, successful designs. Creative designing produces novels designs by adding extra variables to the process, enabling new knowledge to be utilised [12]. These methods of looking at the design process enable us to mathematically make multiple attempts at designs and simulate human creativity through brute force over targeted knowledge.

2.2.2 Interaction in different levels of conceptual design

When looking at conceptual design, the type of design change can be divided into three levels. There are the functional, behavioural and structural design levels, the combination of which is called the Functional-Behavioural-Structural ontology [15]. The functional level of a design is the design's purpose, what the product is supposed to do once complete. The least abstract level of design is the structural level, which is defined by the components involved in the design. These components and their relationship to one another can be explained as how the design is put together. The intermediary level that links the function and structural levels is the behavioural level. The behavioural level acts in a way that is derived from the structural level. This behaviour is how the design acts. The behavioural level is able to be measured and used to provide comparisons between designs [16]. The different design levels are able to be understood intuitively by people, and trained designers know the levels are interlinked. An example of how these can be conceptually defined for a CubeSat's attitude with respect to Earth is shown in Table 2.1. The interlinking between the structural and functional levels must go through the behavioural level, due to there not existing a direct link between the structural and functional levels [17]. Designers can often take short-cuts from structure to function, however since there is no direct link, it is possible to form incorrect expectations of function from a structural component.

Table 2.1: Examples of different design levels in a CubeSat

Design Level	**Example**
Functional	Able to maintain camera alignment with ground below
Behavioural	Rotates about x-axis at a speed dependent upon orbital rate
Structural	Attitude control component (e.g. Flywheel, magnetorquer, etc.)

Within the functional-behavioural-structural (F-B-S) design ontology, it is possible to map the different levels of design to one another in a limited manner. The functional level of a design is able to be mapped directly to the behavioural level and the behavioural level to the structural level, but not directly from the functional level to the structural level [16]. Potentially, this means that it will be possible to work on a conceptual design at one or more levels in computer aided design. Another potential for computer aided conceptual design (CACD) is that it can be constructed on one level, then performance of the design analysed at a different level. The method to construct a design and analyse it computationally is explained in Section 2.2.3.

In addition to the named sections of the F-B-S ontology, there are some other mappings that occur. One major expansion on it is where the behavioural level is separated into two different forms in practice, to differentiate the expected behaviour (Be) and the behaviour derived from the structural level (Bs). These two forms can be compared against one another in analysis at a higher level [16]. The different forms of the behavioural level will be examined later in this thesis and whether reasonable judgments can be made autonomous, or will need to be manually evaluated later in the design process. Another is that since customer requirements are the inputs into the design process, they map directly to the functional level of design. The output is a description of the design, which is performed at the structural level [16].

When working with the F-B-S ontology, there are methods to change different variables and make judgments on the effects that the change will have at different levels of the design. How these will be implemented in an algorithm will be discussed in Section 2.2.3. It is possible to combine two or more attributes at the same design level to produce a new attribute [12]. In addition to design combination, at the structural level transformation of an attribute is also possible [16]. Transformation can be treated as a mutation within emergent computing, with two possible methods, the first is changing a variable to a different value of the same class, such as a change in mass. The second is a change in the variables to generate a new class of features. This second method of mutation can be utilised to generate new ideas and is able to be defined as creative design [12]. These mutations all occur at the structural level which propagates in the higher levels of the design.

Often design can be treated as an iterative process [18], which enables a design to improve in stages. This methodology has been formalised in the F-B-S system. Figure 2.1 displays a way that formal iteration can occur in the design process.

These eight processes and transformations are explained below [16]:

- 1: Formulation of the Customer Requirements (CRs) into requirements for the design process. And from these the expected behaviour that the design should have.
- 2: Synthesis of the expected behaviour into a structure.
- 3: An analysis of the behaviour as derived from the structure.
- 4: Evaluation is performed between the expected behaviour and the structurally derived behaviour.
- 5: Documentation of the structure to the final design.
- 6: Reformulation type 1 changes the structure to provide different derived behaviours to achieve the desired functions.
- 7: Reformulation type 2 changes the structure when a desired change in behaviour is required.
- 8: Reformulation type 3 changes the structure to account for changes in the customer requirements.

This thesis will be using reformulation types one and two, as they have a requirement for consistent customer requirements for each design. The simulation of the overall iteration process is one of the desired outcomes of this thesis. While the F-B-S ontology deals with the design process as a whole, multiple iterations of it can occur in the conceptual design stage alone [12]. The F-B-S ontology is a mature method of working with the design process and has had success in human driven design projects. This thesis investigates methods to apply this framework into an autonomous system.

2.2.3 Emergent computing in conceptual design

When attempting to create CACD, there is a requirement to link the algorithms with a specific purpose. This thesis is focusing upon utilising different algorithms in conjunction with one another, where each individually fulfills a different aspect of the

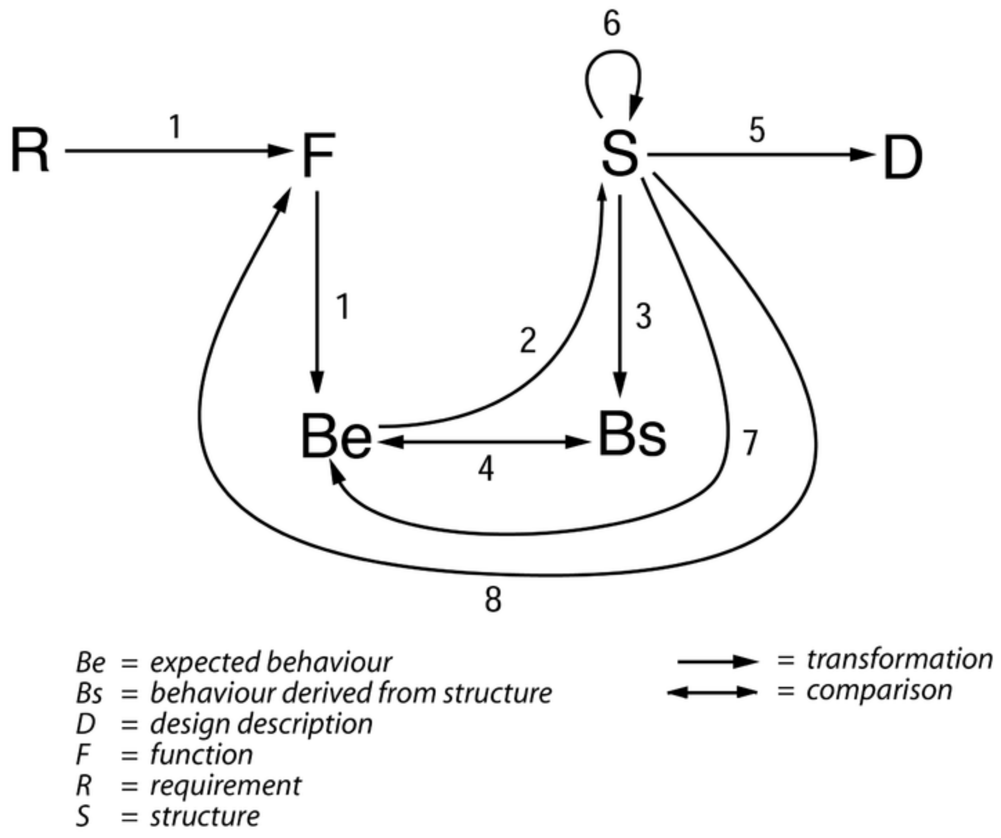


Figure 2.1: The Function-Behaviour-Structure Framework. From: [16]

functional-behaviour-structure ontology. This section of discusses using genetic algorithms and neural networks to allows computers to autonomously link different design levels to one another.

A major focus of emergent computing is genetic algorithms, which have been utilised in studies in conceptual design since the 1990s [3]. The strength of genetic algorithms is that they are good for working with discrete variables and large numbers of design parameters [19]. Early methods of conceptual design that have utilised genetic algorithms solved direct design issues, such as forms of bridges, from a limited number of shapes and sizes based upon input criteria. Genetic algorithms are often employed for solving non-linear fitness problems by randomly generating changes in discrete variables. Fitness scaling is applied each generation, however this does not give true autonomy to the algorithm and requires extensive expert knowledge to create the fitness algorithm.

More recently genetic algorithms have been utilised for searching design spaces, due to their ability to contain a wide variety of different variables. It is possible to design a variation on the basic algorithm that allows for conditional behaviours to occur. This is useful in situations where two contradictory components to a design solution may occur within a single generated chromosome [20]. Another advantage of genetic algorithms is their ability to find designs that people would not naturally search for due to the size of the design space and solutions that are are complex to analyse, which results in highly advanced designs such as the antenna for NASA's Space Technology 5 spacecraft [6]. Genetic algorithms return results based upon their search criteria and have the ability to sample the search space extensively.

Improved Multi-Objective Algorithms

Many genetic algorithms that were used previously in the design process had single objectives that they sought to optimise which limits their use in designs that had multiple requirements to balance against one another.

In engineering design, it's important that multiple design criteria are balanced against one another, which leads to multiple optimal solutions, which leads to Multi Objective Pareto Optimisation (MOPO). The solutions generated by MOPO algorithms form a pareto front, multiple solutions that can then be presented to a designer to be chosen between [21]. One of the problems with a standard MOPO algorithm is that is unable to deal with parameters that need to be categorised. For example, standard algorithms can easily compare distances traveled against fuel consumed, but don't compared whether a car could be considered a luxury vehicle or not. Fuzzy logic is an adaptation that has been utilised previously to improve pareto-optimality [22]. Fuzzy logic allows for classification of these parameters and would enable an algorithm to sort individual design features into groups, to assist in the classification stage of autonomous design.

The following subsections explain two algorithms that are utilised in this thesis to explore the design space through a pareto-optimal genetic algorithm and a neural network that establishes weighting for different designs and their adherence to the customer requirements. The Fuzzy ARTMAP algorithm is trained on prior designs and then is utilised to find past designs whose behaviour is similar to the the desired functions for the new design. The generation of new designs utilises a multi-objective genetic algorithm.

Multi-objective genetic algorithms

Genetic algorithms have been utilised to create the structure of engineering designs [3] and multi-objective engineering designs have utilised non-dominated algorithms [23]. The aim of a non-dominated algorithm is to ensure that one particular objective does not take over the population at the expense of other objectives. Usually this means that it will explore the pareto-front and points will have concessions between competing objectives. The pareto-front is the set of non-dominated solutions and multi-objective pareto-optimal genetic algorithms aim to sample the pareto-front [24]. There have been multiple methods designed to solve the multi-object pareto problem with genetic algorithms, including weighting different objectives and creating a summed total [25]. A genetic algorithm that has some success in finding extremes for individual objectives is the Vector Evaluated Genetic Algorithm (VEGA), which selects a portion of the next generation's population based upon success in an individual objective [26]. Another alternative to explore the pareto-front more extensively relies upon the principle of sharing, which finds 'peaks' in the search space and assigns members of the next population in relation to the height of each peak [24], this is used within niched pareto genetic algorithms. Another method of pareto-optimality in genetic algorithms is a sorting structure, that emphasises performance across a range of different objectives. This sorting structure is utilised in the NSGA-II algorithm to stretch across the entire front [27]. The NSGA-II algorithm is written in Appendix A and is an example of a non-dominated genetic algorithm.

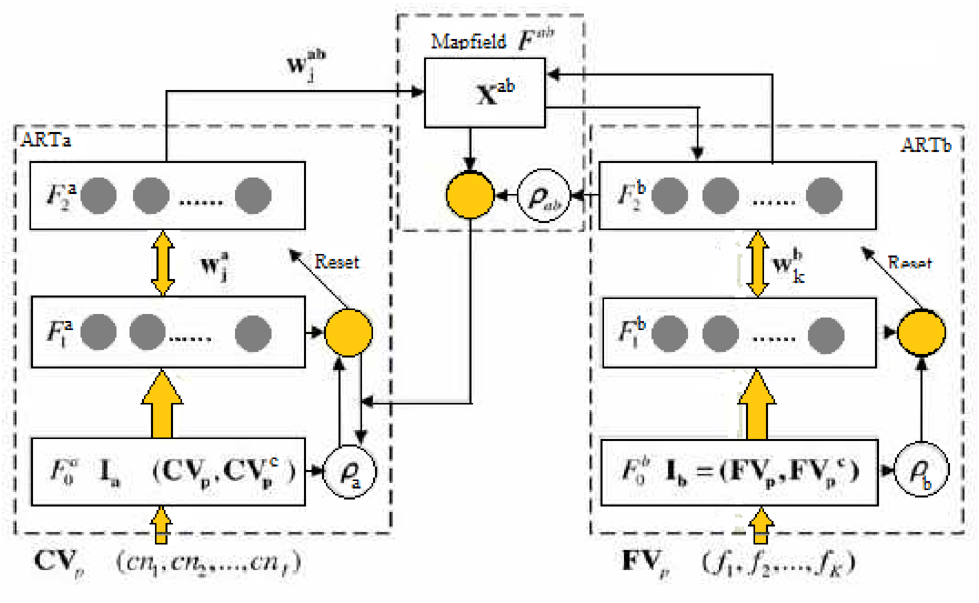
These genetic algorithms all have their advantages when seeking fully explore the pareto-front and engineering design is known for balancing objectives against one another, which allows for 'good enough' solutions across all the objects. What the above algorithms do not fully encompass is a principle of 'good enough' with regards to an individual objective. Where a front does not need to be fully explored, but can accept an objective being acceptable. An example of this in design is where the customer requirements will state that a product must be able to transmit at least 2 Megabits (Mb) of data per day. Anything above 2 Mb is an acceptable achievement but is not required, while the pareto-front method will find the exact 2 Mb point only if it requires a trade off with another objective if greater than 2 Mb is possible. A potential solution to this is proposed in section 3.5.

2.2.4 Fuzzy logic and identification of similar designs

A major use for neural networks is the classification of inputs into different categories [28]. Neural networks are capable of being trained on different types of input, including fuzzified values [29]. The classification process of a neural network allows for related but vague inputs to be utilised, which is useful for conceptual design since in the early stages of design, customer requirements can be vague, imprecise and potentially uncertain [5]. Fuzzy logic is utilised for the quantification of customer requirements due to its inherent property of imprecise values. A particular type of neural network, called Fuzzy ARTMAP (FAM), utilises incremental learning with fuzzy values and results in lower training times in comparison to other neural networks such as Multilayer Perceptron [29]. This allows extremely small training sets to be utilised and still achieve convergence of the network. The Fuzzy ARTMAP algorithm is explained in the following section.

Fuzzy ARTMAP algorithm

An improved algorithm that has been utilised in the search for prior similar designs is the Fuzzy ARTMAP (FAM) neural network [30]. The FAM algorithm is a neural network that is capable of unsupervised learning. This unsupervised learning is trained upon historical records, and compares new inputs to similar requirements. This algorithm has the capability to develop new clusters of similar designs when required, since it is capable of learning the underlying design logic [30]. The FAM algorithm learns these clusters by training the logic on past designs. A benefit of the FAM algorithm is that it trains extremely quickly, making it usable on small datasets [30] which can be utilised to link the functional level of design with categories of products. Customer requirements are often given as natural language requirements. Using a fuzzy logic algorithm, it is possible to give them a numerical value in the range of [0, 1], which allows them to be mapped to categories. The FAM algorithm shown in Fig. 2.2 can be implemented with the general following steps.



Mapfield F^{ab}
$\mathbf{w}_j^{ab}$
$\mathbf{x}^{ab}$
ARTa
ARTb
F_2^a ......
F_1^a ......
$\mathbf{w}_j^a$
Reset
ρ_{ab}
F_2^b ......
F_1^b ......
$\mathbf{w}_k^b$
Reset
F_0^a $\mathbf{I}_a$ $(\mathbf{CV}_p, \mathbf{CV}_p^c)$
ρ_a
F_0^b $\mathbf{I}_b = (\mathbf{FV}_p, \mathbf{FV}_p^c)$
ρ_b
$\mathbf{CV}_p$ $(cn_1, cn_2, \ldots, cn_I)$
$\mathbf{FV}_p$ $(f_1, f_2, \ldots, f_K)$

Figure 2.2: A schematic of the FAM network algorithm [30]

Two fuzzy ART modules, ART_a and ART_b are generated and joined together by an "inter-ART module", F^{ab}. This inter-ART module is more commonly known as a "*map-field*" [31]. The predictive core of the FAM algorithm is this map-field. Each ART module consists of three layers, labeled as F_0, F_1 and F_2. Each layer has individual nodes that interact with the other layers. The first layer, F_0 is the input vector, composed of normalised data and converted into complement code form. For example: ART_a, $\mathbf{I} = \mathbf{A} = (a, a^c)$. This complement coding is utilised to prevent the rise of categories as part of the normalisation process [29].

Nodes in F_1 are then linked with the output nodes in F_2 by a weight vector(w). The nodes in the F_2 layer are representations of a cluster of input patterns [30]. Each of the ART modules has a vector that leads to the map-field. The ART_a vector is also modified by a weighting value. There is a one-to-one alignment between the nodes in F^{ab} and F_2^b [30].

There is a working example of combining the FAM algorithm to retrieve designs based upon customer requirements. This example and technique, while not directly applicable is similar enough to have experiments to see if it can be co-opted for the desired design tasks by training the network upon the researched prior examples of engineering and making associations then utilising the generated vectors to judge designs generated by the genetic algorithm in section 2.2.3.

2.3 Summary

Based on past work, the design process is a mature field when it relates to human designers, while automated design is a relatively new field. There are well established forms of design that are being utilised to create products and satellites. The design process has been aided by computers extensively, and computer aided design is a large field, but has focused upon the single component level, or interfacing multiple components manually. Creation of new designs that are non-intuitive has been achieved at the sub-system level with genetic algorithms but not at the system level. System level often utilises more in depth design methodologies to be successful. These problems can be summarised with the following two questions.

1. **Can the levels of design be linked autonomously?** Retrieval systems for past designs utilise fuzzy logic and neural networks to identify designs with similar customer requirements. The challenge is to examine whether using fuzzy logic to link design levels can be applied to new designs.

2. **Can an evolutionary algorithm create sensible system level designs?** Ge-

netic algorithms have created new designs for individual sub-systems previously, the challenge here is to find a way to create a structure, derive the behaviour of the design from the structure and then judge whether the behaviour fulfills the desired function of the design. If this is possible, then it will complete the link between the requirements level and the design level according to the F-B-S ontology.

Chapter 3

Methodology

3.1 Introduction

The Functional-Behavioural-Structural ontology of design as explained in section 2.2.2 requires processes to provide 'linkages' between design levels. Fuzzy logic encoding is utilised to create the link between the functional and behavioural levels, represented as an arrow in Fig. 2.1. The metrics that are discussed in section 3.4 convert the product specifications of the behavioural level to values that align with the fuzzy logic values.

The first process for the experiments in this thesis is creating and ensuring the linkage between the customer requirements (CRs) and the product specifications (PSs) is possible. As Yu and Wang have demonstrated [30], it is possible to create a natural language expression of customer requirements through the use of Fuzzy logic encoding. This method has been shown to work with triangular fuzzy numbers and is utilised within this thesis as explained in section 3.3. The second process involves creating a population of designs that contain PSs and can be compared to existing designs, to match a new set of customer requirements. Methods to judge potential solutions, against the variety of criteria are tested to evaluate an optimal fitness function for the genetic algorithm, is discussed in section 3.5.1

3.2 Algorithm overview

The proposed overall algorithm utilised within this thesis is shown in Figure 3.1. Step 1 of the algorithm is the process of training a network map utilising a Fuzzy ARTMAP (FAM) algorithm as explained within Section 2.2.4. The the implementation of FAM, including the encoding of the customer requirements is discussed in detail within Section 3.3.

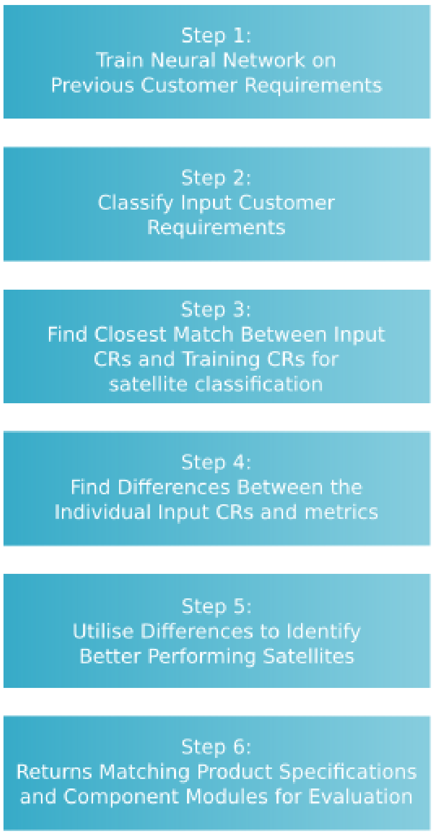

Step 1:
Train Neural Network on
Previous Customer Requirements
Step 2:
Classify Input Customer
Requirements
Step 3:
Find Closest Match Between Input
CRs and Training CRs for
satellite classification
Step 4:
Find Differences Between the
Individual Input CRs and metrics
Step 5:
Utilise Differences to Identify
Better Performing Satellites
Step 6:
Returns Matching Product Specifications
and Component Modules for Evaluation

Figure 3.1: Algorithm Overview

Step 2 is a human input stage, where the customer requirements for the proposed system are codified and put into the appropriate category, for example an attitude control accuracy of 0.1 degrees is rated as High, while an accuracy of 5 degrees is Medium-Low. Step 3 is a stage where the codified customer requirements are classified against the trained network, which categorises the satellite, allowing for comparison to past satellites.

This leads into the genetic algorithm (GA) fitness function. The differences between the input customer requirements and the metrics discussed in section 3.4 are identified in step 4 utilised in the fitness function in step 5 of the algorithm, which selects the satellites for the next generation's population as elaborated upon in Section 3.5.1. The final stage of the algorithm (step 6) is returning of the product specifications of the generated designs for evaluation and analysis, along with the CubeSat components that were utilised to generate the product specifications.

The advantages of utilising a high-level overview of the algorithm allowed for a modular approach within testing. This enabled multiple approaches to be tested for each step and allowing for optimisations in the linkage between customer requirements and product specifications to be made.

3.3 Quantifying and encoding customer requirements

Within the F-B-S ontology discussed in section 2.2.1, there are links between different design levels. To computationally evaluate customer requirements, the requirements must be converted into a numerical value so they can be compared between the functional and behavioural levels of design. Fuzzy logic is utilised to convert customer requirements from natural language variables into numerical values. As customer requirements are often vague or imprecise in the early stages of conceptual design [5], fuzzification is an ideal method to quantify them for the purposes of design.

3.3.1 Scope of customer requirements

During this thesis, a range of different categories of customer requirements were considered. However, the scope is focused upon core missions of the satellites, which limits the types of requirements to be considered. Limiting the scope involved four factors. First is the payload, the main equipment for the mission. There are two parts to the payload requirement, the type of measurement being taken, for example photographs in visible light or a measurement such as the current magnetic flux at a point above the earth, and the second aspect to this requirement is the accuracy of the measurement or detail in the image. Secondly, the satellite is a CubeSat and adheres to the standardised CubeSat specifications, which is assumed to be a constant for all satellites and is factored into the metrics discussed in section 3.4. Thirdly, the satellite's control system must be accounted for. This control is the avionics system and its purpose is to provide instructions to the satellite sub-systems and process the data collected. The fourth and final factor is the satellite's attitude control which was determined to be a major factor in the performance of a satellite. This is due to the fact that a satellite is unable to take an appropriate photo or measurement when it is facing the wrong direction.

Other possible categories were considered, however were ruled out for technical reasons, or a lack of components being available to provide a large sample of possible solutions. An example of the former was including an category for the desired orbit, which was dropped due to no component data sheets having any specifications that were influenced by different orbits. In addition to no available specifications, other categories were removed from experimentation due to a lack of options. One example of these categories that were examined was the satellite disposal category. There were only two components with publicly available datasheets that were designed for this task and listed performance specifications for it. These were the DragEn tether disposal method created by Saber Astronautics [34] and the AEOLDOS from Clyde Space which utilises a set of solar sails for aerobraking as its disposal method [35]. The lack of components resulted in only three possible conditions for a satellite, the first is that space disposal was not required and thus no disposal component would be selected for, the second was that the DragEn component was selected and the final condition was that the AEOLDOS was selected.

3.3.2 Quantifying customer requirements with fuzzy logic

Fuzzy logic algorithms were utilised in the encoding of high level customer requirements into numerical values. Once each of the customer requirements is defined as a product specification it has a numerical value. For example, a satellite must have a fast downlink speed and in comparison the uplink speed can be slow. The natural language is assigned a value due to a membership function to deal with the probable truth [30]. Triangular fuzzy membership functions have previously been utilised to represent customer requirements [36] and this approach has been adopted in this thesis. How these values are related to the high-level linguistic descriptors for each customer requirement is explained in the following sections. The following sections explain the encoding of the values as fuzzy logic for the specific linguistic variables and experiments are conducted in section 4.3 to ensure that quantifying the customer requirements with fuzzy logic is applicable to CubeSats.

3.3.3 Communication bandwidth customer requirements

Communication with the satellite is an important part of operating a satellite, the uplink to a satellite is often used for command and control, while the downlink is utilised for transmission of the satellites mission. Other mission requirements, such as acting as a communication satellite can affect the uplink and downlink requirements. The communications customer requirements are based on the current commercially available communications components for CubeSats. These customer requirement values are used for both the uplink and downlink categories due to their similarity in usage. Table 3.1 shows the relationships between the natural language customer requirements, the baud rates and the fuzzy logic values.

The fuzzy logic values are created from matching the normalised values of the baud rates. The progression of baud rates chosen was due to the availability of hardware in communications equipment. To account for the progression, the normalisation was set with a median value of 4800 baud per second (bps), which was the minimum speed of

Table 3.1: Customer requirements for satellite communications

Cust. Reqs.	**Baud Rate (bps)**	**ln(bps)**	**Norm Val**	**Fuzzy Val**
Extremely Slow	<= 1	0	0	0
Very Slow	1200	7.09	0.1	0.1
Slow	2400	7.78	0.299	0.3
Average	4800	8.48	0.499	0.5
Fast	9600	9.17	0.699	0.7
Very Fast	19200	9.86	0.9	0.9
Extremely Fast	>= 38400	10.56	1	1

a downlink radio component, however uplink components ranged from as low as 1200 bps to a maximum of over 100 kbps for both uplink and downlink. The formula for normalising these values and bounding them within the range zero to one is explained fully in section 3.4.2.

3.3.4 Attitude control customer requirement

The purpose of attitude control in satellites is often maintaining the facing of the satellite with respect to the Earth. How precisely this position is kept is the aim of the attitude control customer requirement. Attitude control is made up of two aspects, the ability for the satellite to detect its orientation and the capability for the satellite to hold its position. However for the customer requirement, one is not effective without the other and so the customer requirement is the more precise of the two. Table 3.2 shows the relationship of the approximate acceptable accuracy for the attitude control along with the natural language customer requirement and the associated fuzzy logic values.

Table 3.2: Attitude Control Customer Requirements

Customer Req.	Control Accuracy (degrees)	Fuzzy Values
Free tumbling	N/A	0
Very Lenient	18	0.167
Lenient	6	0.334
Average	2	0.5
Precise	0.667	0.667
Very Precise	0.222	0.834
Extremely Precise	0.074	1

The values in table 3.2 were chosen based upon current CubeSat mission technical requirements, such as the QB50 constellation [37] and the needs of certain components to be able to perform their designed functions. The GOMX-1 satellite has a control accuracy and knowledge of ± 5 degrees [38] and would be classified as having a *Lenient* customer requirement. This requirement is relaxed in comparison industry standards for satellites due to lack of capability for a CubeSat to achieve industry standards. However, the Very Precise and Extremely Precise requirements allows for potential components to achieve the desired outcome, these values do not match the attitude control requirements for larger satellites, which require greater precision.

3.3.5 Remote sensing and satellite payloads customer requirements

Remote sensing is one of the many tasks that satellites can be required to perform and operate are a range of wavelengths including ultraviolet, visible, near infrared, thermal infrared and radar [39–41]. Other common usages for satellites include communication and scientific inquiry. Satellites such as GOMX-1, and the upcoming QB50 constellation are examples of CubeSats with mission objectives that are not remote sensing [11, 42]. The QB50 project has four scientific payloads that are distributed over the constellation. These payloads are the Ion and Neutral mass spectrometer (INMS), Multi-needle Langmuir probe, Flux-ϕ-Probe-Experiment (FIPEX) and thermistors/thermocouples [43]. The majority of available payloads are on the electromagnetic spectrum as shown in Fig. 3.2, so the decision was made to arrange the customer requirements in order of their average wavelength. However, the INMS and FIPEX payloads are not on the electromagnetic spectrum and as such do not have a wavelength value. The method to deal with the lack of an applicable wavelength is assigning the payloads at one end of the normalised values for the fuzzy logic. The Multi-needle Langmuir probe and thermistors/thermocouples, however, are entirely external components that takes up minimal volume. These are components that can be combined with a primary payload manually and so are left off the customer requirements list.

Due to the dramatic distance that the radio bands operate in, it was decided to add the

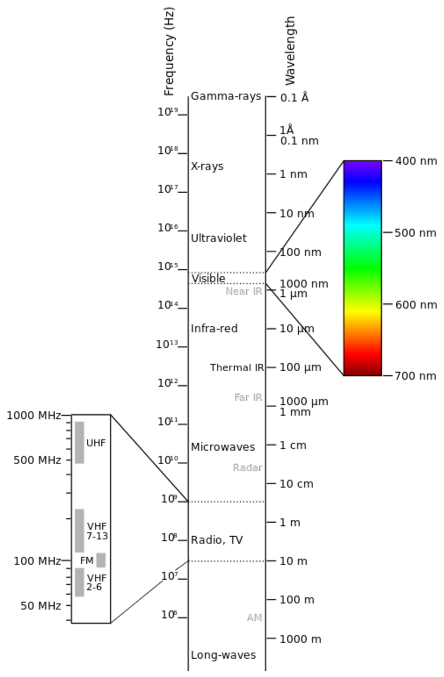


Figure 3.2: Electromagnetic Spectrum [44]

ion and magnetic flux customer requirements to the lower end of the list, having the minimum values. The values are evenly spaced as explained in section 3.3. Table 3.3 shows the assignment of fuzzy logic values and how they align with their average wavelength. With the addition of the ion and magnetic flux customer requirements this takes the total categories to seven. The average wavelength of each band is converted with a natural logarithmic function before being normalised into a value between zero and one, with the visual wavelength being the central value at 0.5.

Table 3.3: Customer requirements for satellite payloads and associated fuzzy logic values

CR	**Avg W/l**	**Ln(W/l)**	**Normed Value**	**Fuzzy Value**
Ion Measurement	N/A	N/A	N/A	0
Magnetic Flux	N/A	N/A	N/A	0.167
Ultraviolet	205 nm	5.32	0.344	0.334
Visual	550 nm	6.31	0.5	0.5
Near Infrared	1350 nm	7.21	0.625	0.667
Thermal Infrared	7500 nm	9.26	0.86	0.834
Radio	0.133 m	18.71	1	1

As table 3.3 shows, once the wavelengths are normalised their values line up relatively closely to the assigned fuzzy values. However, this is considered to be within acceptable margins and testing will be conducted to ensure that this a valid assumption.

The second customer requirement category for the payload was the performance specification of the component. There was no consistent standard utilised in component datasheets for different wavelengths and measurement types to judge accuracy against one another. So this customer requirement is considered to be judged relative to other payloads of its type. So a 1, 3, 10 ratio was chosen for the steps between the natural language options, with no accuracy at all associated with a zero value as shown in table 3.4.

Table 3.4: Customer requirements for payload detail and accuracy

CR	**Ratio**	**Fuzzy Logic**
No detail	0	0
Very Vague	0.1	0.167
Vague	0.3	0.334
Average	1	0.5
Detailed	3	0.667
Very Detailed	10	0.834
Extremely Detailed	30	1

The 1, 3, 10 ratio method has the average value for the accuracy or detail of the payload set as a baseline, anything three times more accurate is referred to as being detailed and anything a that has a third of the accuracy is considered to be vague. An example of this in practice is the Flock 1 satellite constellation which contains 28 'Dove' CubeSats, which have a resolution of approximately 3 meters from low earth orbit [45] which would have an 'average' detail value. This performance is compared to the KazEOSat-1 which has a resolution of 1 meter [46] and is associated with a value of 'detailed'.

3.4 Metric creation for a genetic algorithm and linkage to customer requirements

Metrics are often utilised to measure performance, with a common approach being to have the normalised maximum value representing a goal state [47]. For the performance based metrics, such as the ADCS, a value of one will be the optimum. However other metrics, such as the desired payload can be given values associated with the mission goals, for example the wavelength that the payload's camera operates in. These factors result in metrics being the quantification of the systems product specifications and as such, with proper normalisation can be aligned with the normalised fuzzy values of the customer requirements to provide a direct comparison between the two. The following sections explain in detail each individual metric that is utilised within this thesis and section 3.5.1 discusses how the metrics are utilised within the genetic algorithm to generate new conceptual designs. All metrics are bound within the range zero and one.

3.4.1 Metrics for all satellite operations

All CubeSats have certain requirements in common. These requirements are specified in the CubeSat design specification rev. 13 [9]. This specifies the maximum size of a CubeSat, the maximum mass for each size category and limitations on power sources. The sub-systems for each satellite also need to be controlled and thus an avionics metric was developed. Small penalties account for the possibility that optimising a design can minimise the issue, for example rearranging components to fit inside the allowed volume. However large penalties are often selected against.

Satellite volume metric

CubeSats are assigned categories based upon their size. One of the major requirements of being a CubeSat is that the base shape is that of a 10 cm cube [9]. Each 10 cm cube is considered to be 1U. Thus a CubeSat that is 20 x 10 x 10 cm is a 2U CubeSat.

This metric considers the maximum volume that all the components of a CubeSat is made up of and penalises a potential solution for exceeding the volume budget. The metric algorithm 3.1 shows the penalty related to the allowed volume. Breaching the maximum allowed volume results in a satellite being prohibited from launching.

This volume algorithm determines the satellite size based upon the base structure selected by the genetic algorithm and then calculates the available internal volume allowed. The V_{total} is the sum of the volumes of internal components. It treats anything that fits inside the satellite as acceptable and doesn't penalise these solutions, while exponentially penalises volumes that exceed the maximum allowed volume.

$$\begin{aligned}
&V_{max_{allowed}} \propto Sat_{size} \\
&volume = V_{max_{allowed}} - V_{total} \\
&if volume < 0 : \\
&\quad penalty = \exp^{-volume} - 1 \\
&else : \\
&\quad penalty = 0 \\
&metric = 1 - penalty \\
&metric \in [0, 1]
\end{aligned} \tag{3.1}$$

Satellite mass metric

As in section 3.4.1, CubeSats have assigned categories based upon their size. These size categories also have a weight limitation, which is 1.33 kilograms per 1U [9]. The satellite mass metric compares the total combined mass of components with the allowed mass for the size category of the CubeSat.

$$
\begin{aligned}
& Mass_{allowed} \propto Sat_{size} \\
& Mass_{sat} = Mass_{allowed} - \sum Mass_{sat_{components}} \\
& if Mass_{sat} < 0 : \\
& \quad penalty = \exp^{-Mass_{sat}} - 1 \\
& else : \\
& \quad penalty = 0 \\
& metric = 1 - penalty \\
& metric \in [0, 1]
\end{aligned}
\tag{3.2}
$$

Penalties are an exponential function that gets harsher the further over the mass limit the solution is. This is a metric that is bound in the range zero to one. It is assumed that refinements in later stages of design will ensure that the satellite will comply with the mass budget.

Avionics metric

All satellites need an avionics system for normal operation and this is often performed by the flight board/computer [9], however, it is outside the scope of this thesis to create a one size fits all metric for the software and the hardware that they run on. However, the database contains multiple different variations of flight boards that are commercially available. All flight boards in the database have similar size and power requirements. Within the scope of this thesis, this enables a binary metric, that ensures the presence of a flight board, while after the system is created, an engineer can decide on an appropriate flight board for the system requirements.

$$\begin{aligned}
&if Flightboard == present:\\
&\quad metric = 1\\
&else:\\
&\quad metric = 0
\end{aligned} \tag{3.3}$$

Electrical power system metric

Most satellites are designed with solar cells and rechargeable batteries to continue operation in orbit. The design of a satellite power system is a complicated task in itself that requires the engineer to account for the angle of the solar cell to the sun, the shade due to orbit and degradation of solar cells over time [48]. The complexities of designing a power system are outside the scope of this thesis and treats commercially available components as providing their advertised capabilities. The following algorithm is used to calculate the electrical power system metric.

$$\begin{aligned}
&Power = \sum Power_{generated} - \sum Power_{max_{used}}\\
&if Power < 0:\\
&\quad penalty = \exp^{-Power} - 1\\
&else:\\
&\quad penalty = 0\\
&metric = 1 - penalty\\
&metric \in [0, 1]
\end{aligned} \tag{3.4}$$

This metric is based upon a peak power usage and assumes that all components have a nominal amount of power draw at all times. The maximum power usage is the single largest power draw of any component at once. The reasoning behind only a single components maximum power being accounted for is that software can be utilised on-board a CubeSat to ensure that power is given to the current mission critical subsystem.

A further enhancement to the metric algorithm is to check whether the satellite both has a battery system and whether it requires a battery. Most orbits have periods where the satellite is in the shade, which can be up approximately fifty percent of the orbit.

These day/night cycles required the knowledge of the duration of the orbit, along with the orbital inclination to calculate how long each cycle the satellite is in the Earth's shadow. Without knowing the planned orbital elements, especially the altitude and inclination, this calculation can't be performed correctly. This customer requirement is outside the scope of this thesis. The end of life power output is substantially lower than at the beginning of the mission, however it can not be determined from the minimal information known [49]. For orbits that are not in a sun-synchronous orbit and are conducting missions that require permanent up-time, the approximation for the day/night cycle calculation is to divide the metric in half if a battery is not amongst the components. The end of life power generation will need to be calculated manually once all the variables are known.

3.4.2 Metrics linked to customer requirements

The following sections explain the metrics based on customer requirements used this thesis. These metrics map directly to the customer requirements given by the user. There are six customer requirement based metrics in use, combined with the four metrics common to all CubeSats previously discussed for ten metrics in total. The following six metrics are made by mapping directly to the communications and payload customer requirements, with an expansion of the attitude control requirement into two metrics. The attitude control metrics independently evaluate both the physical moment capable of being generated and the satellite's ability to estimate it's current attitude.

Communication bandwidth metrics

The communications metric equation is the same for both the uplink and downlink metrics, for, while the customer requirements may differ between the two, the hardware has standardised baud rates on their modems. Equation 3.5 demonstrates the method utilised to normalise the metric into the range zero to one.

$$\begin{aligned}
baud_{min} &= 1200 \\
baud_{max} &= 38400 \\
metric &= \frac{ln(baud_{down}) - ln(baud_{min})}{ln(baud_{max}) - ln(baud_{min})} + 0.1 \\
metric &\in [0, 1]
\end{aligned} \tag{3.5}$$

The logarithmic function for all the values enables the range to be easily normalised, while the minimum and maximum values are selected based on the customer requirements and available components. The purpose of the flat 0.1 modifier in this metric is to allow for the condition that a satellite has been selected without any communications capability in the relevant direction.

Attitude control metrics

The attitude control metrics are the only metrics to have an adaptable goal state based upon customer requirements that are not an exact one to one match to the customer requirements. The attitude control customer requirement was divided into two categories to achieve it based upon the different technologies required to achieve the desired result. A core principle of being able to maintain the correct attitude is to know the current orientation of the satellite with respect to an inertial reference frame and the other is to be able to manipulate the orientation to remain there. As such, this customer requirement is divided into the *knowledge* and *moment* metrics.

The first is the attitude knowledge metric, which relies on components ability to perceive the current orientation with respect to the Earth. In this case, it selects the minimum value available and then converts the raw knowledge value into the range [0, 1]. The algorithm to match the metric to the customer requirement is shown in algorithm 3.6.

This algorithm has the scale of the attitude knowledge repeatedly being divided by three. Each division is associated with a single tier of the customer requirement based around the average attitude knowledge of 2 degrees, and a knowledge of 6 degrees being equal to a 'lenient' value. While increasing the precision, a value of $\frac{2}{3}$ is considered to have a value of 'precise'.

```
$m = 18/k$
$ifm == 1 :$
  $r = 0$
  $i = 1$
$elseifm < 1 :$
  $ifm > 0.75 :$
    $i = m$
  $else :$
    $i = 0$
  $r = 0$
$else :$                                              (3.6)
  $i = 1$
  $whilem > 1 :$
    $i = i + 1$
    $m = m/3$
  $ifm \neq 1 :$
    $r = (1 - m) \bmod 3$
  $else :$
    $r = 0$
$division = 1/6$
$metric = (i * division - r * division)$
$metric \in [0, 1]$
```

The second metric for attitude control is the moment generated. There is no information for inertial properties in the datasheets for the majority of of components, nor does the genetic algorithm arrange the components in any specific order. Thus the simplified metric as shown in equation 3.7.

$$\begin{aligned} metric &= \frac{moment}{mass} \\ metric &\in [0,1] \end{aligned} \tag{3.7}$$

Payload and wavelengths metrics

The payload metrics include both the payload type and the performance of the component with respect to the average of that payload type. The scaling equation utilised for the payload itself is shown in equation 3.8. For the majority of payloads, they operated on the electromagnetic spectrum, which enabled the metric to calculate the median value of the wavelength range for a component and compare it directly against the other payloads. However, there were exceptions for both the ion and neutral mass spectrometry and magnetic flux payloads. Neither of these were directly on the em spectrum, so their components were given pseudo wavelength values that are equivalent to the fuzzy logic value assigned the respective payload in section 3.3.5.

$$\begin{aligned} Wavelength_{max} &= 2ln(\frac{Visual_{max} - Visual_{min}}{2}) \\ Wavelength &= \frac{Wavelength_{component_{max}} - Wavelength_{component_{min}}}{2} \\ Wavelength_{log} &= ln(Wavelength) \\ metric &= 1.75(\frac{Wavelength_{log}}{Wavelength_{max}} - 0.5) + 0.5 \\ metric &\in [0,1] \end{aligned} \tag{3.8}$$

The second metric is the accuracy of the payload. For example, payload accuracy can be the detail possible in a photograph, the nanometer precision in a thermal reading, the number of communications messages receivable per second, or other measures of effectiveness depending on the particular type of payload. Unfortunately, with the way the database was constructed, this metric was manually calculated and retrieved directly from the database as the satellite was created. It was based upon the 1, 3, 10 method explained in section 3.3.5.

3.5 Creating designs with a genetic algorithm

This thesis investigates autonomously creating conceptual designs based upon customer requirements. The database of components that was compiled has 60 unique components, along with six different structures that these components can be placed within. This results in a search space of 6 times 60 factorial possible solutions. As genetic algorithms are a stochastic search method, they take random samples of the search space and attempt to optimise the population over generations. Figure 3.3 shows an example of how a genetic algorithm's population often will be divided within the search space (not to scale).

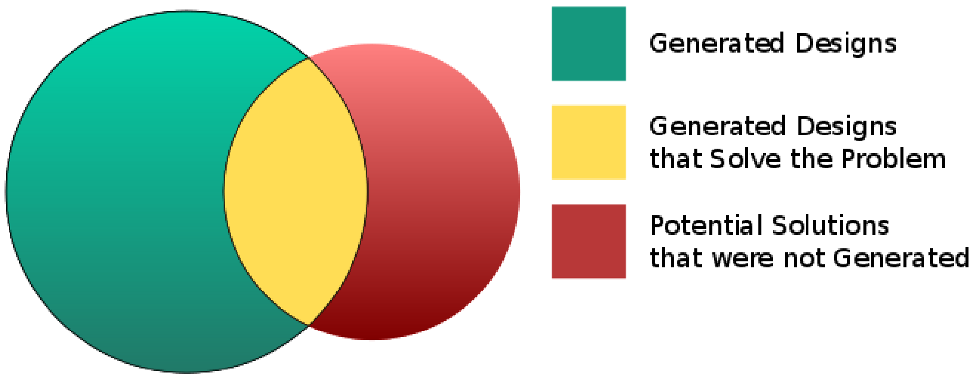

Generated Designs
Generated Designs that Solve the Problem
Potential Solutions that were not Generated

Figure 3.3: Comparison of solutions, potential solutions and unsuccessful designs

The Venn diagram shows a simplified process of a genetic algorithm that has generated workable solutions. Unfortunately, since genetic algorithms are stochastic, there is no guarantee that the section of the Venn diagram that shows the 'Generated designs that solve the problem' will be present within the population. However, genetic algorithms are commonly utilised for tasks similar to this, with unsuccessful attempts being repeated, because it is possible that due to the random nature of a genetic algorithm an appropriate solution was just not generated, or with a larger population to increase the solution space. The following sections explain the genetic algorithm that has been developed for this thesis.

3.5.1 Multi-object pareto optimal genetic algorithm

As was shown in section 3.4, there are multiple different metrics that have to be evaluated in the creation of a satellite. This thesis utilises a customised genetic algorithm based upon the NSGA-II and VEGA algorithms introduced in section 2.2.3. This change was made to create a linkage between the functional and behavioural levels of design. The customer requirements are used as "goal states" for the genetic algorithm, while the metrics for potential designs are a scaled version of the product specifications. This genetic algorithm uses MOPO concepts to provide the linkage between the functional and behavioural levels of the F-B-S ontology as explained in section 2.2.2.

Initial population creation rules

The initial creation of the population for the genetic algorithm is an important step in the algorithm, for it sets the rules that members of the population have to obey in future generations. These rules and the assumptions that guide them are checked against in the offspring population and the mutation stages of the algorithm. As this genetic algorithm utilises existing components as its base, an individual in the population contains more information than a single vector of numbers or bits. The assumptions made for an individual satellite are as follows:

- All CubeSats have a structural component
- A structural component may contain integrated components, in which case they are considered part of the structure
- Each structure has a number of available internal and external 'slots'
- Non-structural components take up a number of slots based on their size and required position

- The number of slots a component can require is allowed to be any positive number
- Non-structural components can use both internal and external slots
- A non-end external slot can only be filled by a solar panel
- An end external slot may be used by either a solar panel or by non-structural component
- All non-end external slots must be filled by the same type of solar panel

These assumptions do not include any provisions for the metric calculations, this process is performed in each generation of the algorithm. In addition to the assumptions, the rules for the creation of a satellite are composed of the following four steps:

- 1^{st}: Generate a random structural component for the satellite
- 2^{nd}: Calculate/retrieve the number of slots available for components
- 3^{rd}: Randomly generate non-structural components one at a time with a loop. If they can not fit within the slots remaining, generate a new component until all slots are utilised
- 4^{th}: Randomly select solar panels from the database for both end and non-end slots

Once each satellite is created, it is appended to the population and the algorithm continues until the population size reaches the maximum size. Once the initial population is created, the algorithm enters the generation loop which continues until the maximum generations is reached.

Mutation and crossover rules

The creation of the child population is the stage of the genetic algorithm where the previous generation is changed and differences are inserted into the population for evaluation. This is the method by which the overall fitness of the population is improved from generation to generation. The first stage of creating the child generation is called 'crossover'. In this implementation crossover is conducted with the following steps:

- Divide parent population into two sub-populations named P_1 and P_2
- For uneven populations, remove the extra individual from the population and store it
- For 1 to n, where n is the number of satellites in a sub-population, conduct the following procedure
  - Take the i-th satellite from both sub-populations
  - Combine all internal components of both satellites into a single list
  - Randomly determine whether the first satellite (Sat_a) will use the structure from P_{1_i} or P_{2_i} with the second satellite (Sat_b) getting the remaining structure
  - Randomly select components from the combined list to add into Sat_a until it has filled the available slots according to the rules laid out in section 3.5.1. Once Sat_a is created, repeat for Sat_b.
  - If a satellite has remaining slots and the list of components is empty, randomly generate a new component from the database
  - Randomly assign solar panels from P_{1_i} and P_{2_i} for Sat_a and Sat_b in the same method as the structure
  - Append both Sat_a and Sat_b to the child population (C_t)
- Append the stored individual to the child population

Once this crossover process is complete, an initial child population has been created. This does not guarantee the possibility that new components will be added into the population over time. This requirement is performed by the mutation procedure, which has parts to it. The first option for mutation is changing the structure of the satellite, while the second is replacing internal components. This algorithm utilises both approaches, acting upon an individual in the population, with the limitation that if the satellite is selected to be mutated, only one option is chosen. The implementation is as follows:

- Determine if satellite is to be mutated by generating a random number and if it is less than the mutation rate, continuing.
- Randomly determine if the structure or the components are to be mutated
- If structure is mutated:
  - Place all components into a temporary list
  - Randomly generate a new structure from the database
  - Add components into the structure from the temporary list, following the rules from section 3.5.1
  - If all space is utilised, the mutation is complete and extra components are discarded
  - If there are remaining slots available, generate components until the slots are filled
- If the components are to be mutated:
  - Place all current components into a temporary list
  - Randomly generate a component from the database
  - Randomly select items from the list until all satellite slots are filled
  - Discard extra components if all slots are filled, or generate new components until slots are filled

The mutation stage finishes creating the child population in preparation for making the next generation for the algorithm.

The intermediate generation population creation rules

After the child population has been generated, the intermediate generation population is created for evaluation of the metrics and the creation of the rankings. This is the union of the parent population and the child population into a single population as shown in 3.9. This results in a temporary population size twice that of the designated population size.

$$R_t = P_t \cup C_t \tag{3.9}$$

Where R_t is the resultant population being created from the parent and child populations. One of the side effects of this method of creating the next generation is that it does not check whether any members of the child population are identical to the parent population. However, due to the mutation and crossover explained in section 3.5.1 there is always the chance that better solutions will be created in a population that may become dominated by a single solution.

Similarities and differences for use within a fitness function

The fitness function in a single parameter genetic algorithm is designed to minimise, or maximise, the value and achieve a desired goal, by providing a single scalar value [24]. This thesis treats each metric as an individual parameter and calculates the distance for each separately. There are multiple reasons to do this. The first is that there is not a continuous pareto-optimal wavefront identified by discrete points required for this algorithm, with multiple components having different tasks and discrete values. Secondly, each metric's calculation is independent of the other metrics, even if they are produced by the same component, the system treats it as a whole. It must be noted that the mass metric is a calculation over a maximum limit and not directly influencing other metrics such as the attitude moment metric in section 3.4.2. Finally, as the purpose of the genetic algorithm is to meet or exceed the customers requirements, as many metrics as possible must be satisfied simultaneously. As a human designer can balance these different objectives, the algorithm must be capable of making a reasonable attempt at achieving multiple goals as opposed to creating the wave front.

The chosen method for the fitness functions in this algorithm was a minimum distance from the goal, with two separate methods utilised. The first method is a 'good enough' method as shown in equation 3.10, for customer requirements that need to achieve a minimum performance to be satisfactory. This is utilised for the majority of metrics, such as the communications metrics, where achieving matching the 'average' customer requirement is acceptable, but a performance value of 'fast' would be more desirable.

$$\begin{aligned}
&if metric > goal : \\
&\quad fitness = 0 \\
&else : \\
&\quad fitness = goal - metric \\
&fitness \in [0, 1]
\end{aligned} \tag{3.10}$$

The good enough method utilises conditional logic to allow anything that exceeds the minimum required level to reach the optimal fitness, without giving any benefit to potentially over-engineering a solution. The second method is utilised for the payload metric, where overshooting the value given would negatively impact the desired performance of the generated satellites, this is shown in equation 3.11.

$$\begin{aligned}
&fitness = |goal - metric| - leeway \\
&fitness \in [0, 1]
\end{aligned} \tag{3.11}$$

The difference in the above fitness function is both in the absolute distance from the ideal solution and the leeway variable, which is utilised to allow for wavelengths that are not exactly centered upon the median values of their waveband, but are still viable options for the satellite and for the engineer to investigate.

Ranking method for multiple objectives

Influenced by one of the methods in the NSGA-II algorithm, a ranking structure was developed to ensure that more optimal designs would propagate through to the next generation. An alternative method for multiple metrics is the Vector Evaluated Genetic Algorithm (VEGA) which analyses the individual metrics for their performance and inputs an equivalent portion of the intermediate generation into the next generation [26]. VEGA is only partially used, for determining the next generation of the genetic algorithm. Optimisation across all metrics simultaneously is the goal in this genetic algorithm rather than determining the entire pareto-optimal front.

This thesis utilises a hybrid approach to create a ranking system for the selection of the next generation. The first satellite to be input into the next generation is the one with the absolute minimum distance selection method. Next, each metric has the minimum score inserted into the next generation, as is used in VEGA. In the case of a tie, the first found is selected. After this is performed for all ten metrics, the algorithm ranks all remaining satellites in the following order, where a zero is a metric that has achieved the desired goal value:

- Calculate the number of zeros belonging to a satellite
- Insert satellites with the maximum number of zeros into next generation
- Ties between individual satellites with the same number of satellites are determined by minimum total distance and if a tie still exists, the first found is selected
- Remove ranked satellites from the intermediate population
- Loop through process until all satellites are accounted for

The reasoning behind this hybrid methodology is that it covers multiple different methods of determining an ideal solution, it ensures that the best solution for every metric exists so that future generations can a chance to maintain that capability. Further, this algorithm is not required to explore the full extent of a pareto front, but to attempt to find a solution that has the highest chance of achieving local optimal solutions. Thus, the usual criticisms of dominated genetic algorithms are not a concern due to the potential domination of a single solution, or small number of solutions, being an acceptable outcome.

3.5.2 Differences to other multi-pareto genetic algorithms

The genetic algorithm used in this thesis takes inspiration from the VEGA and the summed total method for seeding satellites into the next population. However, the remainder of the population is likely to become dominated by a small number of solutions. The domination of the population is caused by the preferencing in the ranking system for success in multiple objectives. However in engineering conceptual design, a small number of optimised designs is preferred to a larger sample of less optimised solutions. Thus this is considered to be acceptable.

Chapter 4

Experiments and Results

4.1 Introduction

Experiments were conducted to assess whether an algorithm can provide information to create an early conceptual design for a CubeSat. The information required in this thesis is a list of components that act as subsystems of the satellite. The categories for these components are: payload, electrical power system, avionics system, communications and attitude control systems. The satellite system bus is the external structure and determines how much internal volume for the subsystems is available and the mass budget for the satellite in question. A valid satellite design for the purposes of this thesis has subsystems that match or exceed the customer requirements and contains the correct primary payload.

4.1.1 Experimental overview

The experimentation procedure for this thesis is divided into two parts. The first stage of the experiments is testing whether the fuzzy logic encoding in section 3.3 is a valid method to create the linkage between the functional and behaviour levels of design. The experiments based upon fuzzy logic utilise a Fuzzy ARTMAP (FAM) neural network to classify a new vector of customer requirements to the most similar satellite. The FAM algorithm is utilised for two reasons, it takes fuzzy logic values as input and due to it returning a result of 'unclassified' for inputs that are dissimilar to the training set. The fuzzy logic experiments commence with a sanity check that ensures that the FAM algorithm has achieved at least partial convergence and is shown in section 4.3. The fuzzy logic is tested in three experiments. The purpose of these experiments is to test that the permutations are similar enough to be recognised as valid inputs. Each data point in a test set is considered to be an individual test for the experiment.

The second set of experiments is based upon the genetic algorithm. The individual components form the structural level of the design process and the data sheets describe the performance and product specifications. The metrics developed in chapter 3 are designed to normalise these product specifications and align them with the fuzzy logic value for the respective customer requirement. Each experiment tests the convergence of the algorithm. Within each experiment are multiple data points utilised as inputs into the genetic algorithm which are treated as independent tests. The first experiment of the genetic algorithm is designed to recreate past operational satellites as a way to validate the algorithm. The second experiment attempts to create satellites by modifying the customer requirements from the training set, this experiment allows comparison between the generated satellite and an existing satellite that shares similar customer requirements and provides further validation of the algorithm. The final experiment for the genetic algorithm uses customer requirements generated from the QB50 project that have no relationship with the training set and attempts to create a design that could be utilised for a space mission. The third experiment demonstrates whether it is possible to generate new CubeSats from customer requirements.

4.2 Equipment and component data sets

The computer these experiments were performed upon was a Windows 7 Professional (64-bit) machine, with 12 GB of RAM and an AMD Phenom II X6 1055T processor running at 2.80 GHz. The programming language utilised for experimentation and evaluation was Python 3.4 with the Anaconda distribution. The data sets were compiled with components from commercial websites. These components were chosen because they are commercially available and gave data sheets for each component. Major software packages utilised were pandas and numpy.

The construction of each entry into the database of parts involved taking the datasheets claims at face value, in Appendix B, Table B.1 lists the complete source of all information utilised within the creation of the data set. Each component has x, y and z dimensions listed, along with a mass value. Depending upon the component, they are classified as either internal or external to the satellite structure. They also have a number of 'slots' attributed to them, based upon the size of the component. There are internal slots and external slots. Most components reduce either internal slots or external slots, with structure components determining how many slots each satellite can contain. Certain components, such as payloads can reduce a number of internal and external slots if they require an external viewpoint, such as a camera. All components are considered to be capable of surviving the thermal conditions caused by being in orbit and the operation of the satellites subsystems. Each component is classified according to the general type of component that it is. Depending upon component type, values are assigned to the database entry, such as a visual camera having resolution values, and wavelength range, while an attitude determination and control system (ADCS) will have valid values for attitude knowledge in degrees. If a component's data sheet does not have a valid entry for a category, then a relevant null value is input into the database.

4.3 Validation of representing customer requirements as fuzzy logic values

The following set of experiments and tests are designed to validate the assumption that natural language customer requirements can be converted into fuzzy logic values. The method of each of the fuzzy logic tests is that a small training set of data, created from past satellite missions will be input into a FAM neural network. Each data point is a vector of fuzzified customer requirements. Once the network has been trained, a test set of vectorised customer requirements will be input into the classification function. The output of this function is a vector which contains the number of the data point (in this case the satellite's customer requirements) the test set data point most closely resembles. The output is then compared to a vector of 'correct' classifications. This correct classification is the original satellite that the test data was created from.

As an additional check of the performance of the fuzzy logic, a summation of the normalised distance of each test data point from each training data point is created, this summed value represents the similarity between the test data point and each of the data points of the training data, a lower value shows a greater similarity than a higher value. The minimum value is checked against the assigned category and the result that the classification function finds. The classification function is deemed to be correct if the value for the category the training data point is classified as is less than or equal to that of the category that the training data point was based upon. In an edge case where the classification function does not identify either a minimum value or the predetermined category, a manual investigation into the cause will be conducted.

4.3.1 Validation test. Identifying existing CubeSats

This validation test was conducted with the purpose of identifying previous CubeSats with identical customer requirements. The training data and the test data sets were identical. The expected result is a 100 percent classification by the FAM algorithm, as a converged neural network algorithm should find matches for data points with identical values. Upon running the experiment, the expected outcome was achieved. For each data point in the test data, there was a distance of zero from the matching data point in the training data. The training data set is shown in table 4.1.

Table 4.1: Customer Requirements Natural Language Values

Satellite	Downlink	Uplink	Attitude	Wavelength	Detail
NanoEye	Average	Average	Lenient	Visual	Detailed
AAUSat	Average	Fast	V. Lenient	Visual	Average
COMPASS-1	Average	V. Slow	Lenient	Visual	Not Detailed
GOMX-1	Average	Ex. Fast	Lenient	Radio	Detailed
ARCTIC-1	Average	V. Slow	Average	Thermal IR	Average
COPPER	Average	Fast	Average	Thermal IR	Detailed

4.3.2 Fuzzy logic. Experiment 1. Changing wavelengths

This experiment explores the effect that changing a single customer requirement in the data point vector has upon the identification capabilities of the classification function. This ensures that there is at least a limited possibility to identify similar satellites based upon commonalities. This experiment utilises the same training set as the validation test. Each data point in the test is created from one of the data points in the training set. Each training data point is utilised to generate six test data points. To provide a distinct comparison, only the payload customer requirement was modified in the test set. An example of this is shown in table 4.2. The full list of permutations is shown in appendix D.

Table 4.2: Example modified customer requirements

Satellite	Downlink	Uplink	Attitude	Wavelength	Detail
NanoEye	*Average*	*Average*	*Lenient*	*Visual*	*Detailed*
Test 1	Average	Average	Lenient	Radio	Detailed
Test 2	Average	Average	Lenient	Thermal Infrared	Detailed
Test 3	Average	Average	Lenient	Near Infrared	Detailed
Test 4	Average	Average	Lenient	Ultraviolet	Detailed
Test 5	Average	Average	Lenient	Mag Flux	Detailed
Test 6	Average	Average	Lenient	Ion and Neutral Mass	Detailed

The experiment was performed with every other possible payload for each satellite, for a total of thirty six permutations. The total normalised distance for each permutation with respect to each item in the training set is shown in appendix E, table E.4. This experiment had a 88.89 percent classification of each permutation to the category of the satellite that it was based upon and a complete list of the classification categories can be found in table E.1. Analysis of each permutation against every satellite in the training set uses equation 4.1. Each permutation is given 'scores' which are generated by calculating the distance between the individual customer requirements for the permutation and the satellite in the training set, then a summation of all the distances. If a permutation shares a customer requirement with the satellite, the distance for the individual customer requirement will be zero. Lower scores show greater similarity between a permutation and a training satellite.

$$Score = \sum_{i}^{n} |CR_{Perm_i} - CR_{Sat_i}| \tag{4.1}$$

In depth examination of the results show that for certain permutations of the GOMX-1 satellite, the NanoEye satellite was extremely similar. An excerpt from table E.4 is shown in table 4.3 which shows the permutations of the GOMX-1 satellite. The italicised entries show the similarity value that was identified and the bolded entries show the value of the category that they were expected to be assigned. The values for tests 21 through 24 in the NanoEye and GOMX-1 columns are the same within floating point error. The remaining columns are scores for the other satellites in the training set, numbered in order that they were presented in table 4.1.

Table 4.3: Experiment 1. Scores for Permutations of the GOMX-1 Satellite.

	GOMX	**NanoEye**	**AAU**	**COMP.**	**ARC.**	**COP.**
Test 19 Thermal IR	0.166	0.834	0.967	1.568	1.234	0.467
Test 20 Near IR	0.333	0.667	0.800	1.401	1.401	0.634
Test 21 Visual	**0.500**	*0.500*	0.633	1.234	1.568	0.801
Test 22 Ultraviolet	**0.667**	*0.667*	0.800	1.401	1.735	0.968
Test 23 Mag. Flux	**0.833**	*0.833*	0.967	1.567	1.901	1.134
Test 24 Ion Spec.	**1.000**	*1.000*	1.133	1.734	2.068	1.301

The NanoEye and GOMX-1 satellites were both developed by GomSpace ApS in Denmark. As can be seen in table 4.4 the hardware between the two satellites is extremely similar. The major differences between the two is the addition of the Custom Software Defined Radio (SDR) payload in the GOMX-1 and the difference in unit size. The SDR was the primary mission requirement for the GOMX-1 and was utilised to track trans-Atlantic flights by receiving their broadcast signals [51]. This radio reception operated on a different frequency to both the control uplink and the downlink transmissions of its observations and operated in excess of 100 kB per second.

Table 4.4: Comparison of GomSpace CubeSat Components

	NanoEye	**GOMX-1**
Payload	NanoCam C1U	Custom SDR
Secondary Payload		NanoCam C1U
EPS	P31U Power Supply	P31U Power Supply
	P110 Solar Panels	P110 Solar Panels
Avionics	A712D Cubesat Computer	A712D Cubesat Computer
	NanoHub	NanoHub
Communications	U482C UHF Transceiver	U482C UHF Transceiver
	ANT430	ANT430
	Antenna Release Board	Antenna Release Board
Attitude Control	Attitude control system	Attitude control system
CubeSat Size	1U	2U

The customer requirements for these satellites is compared in Fig. 4.1. The two major differences between these satellites is the uplink speed and the payload requirements. The other requirements are unchanged. As the permutations for the experiment only change the payload requirement this is the most likely cause of the rising similarity between the permutations of the GOMX-1 satellite and the NanoEye satellite.

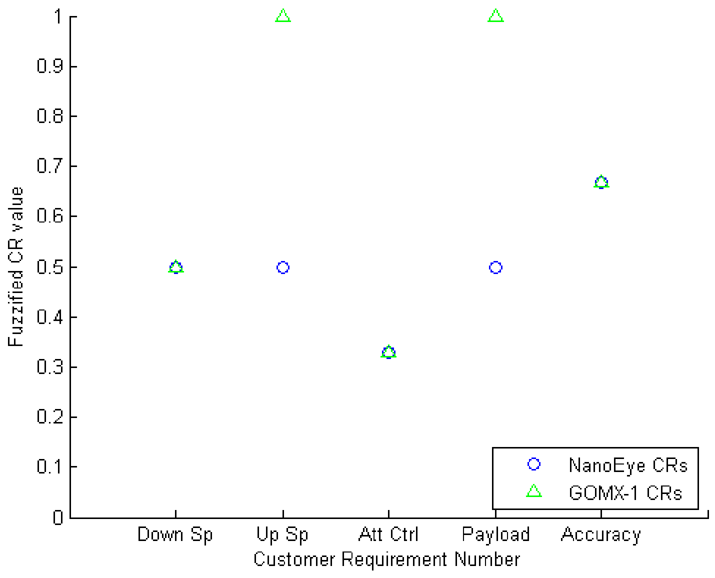


Figure 4.1: NanoEye and GOMX-1 Customer Requirements

The customer requirements for the permutations are shown in figure 4.2 and overlay the CRs for the NanoEye and GOMX-1 satellites. This graphical representation shows how the payload eventually has the exact same value as the NanoEye satellite's payload customer requirement.

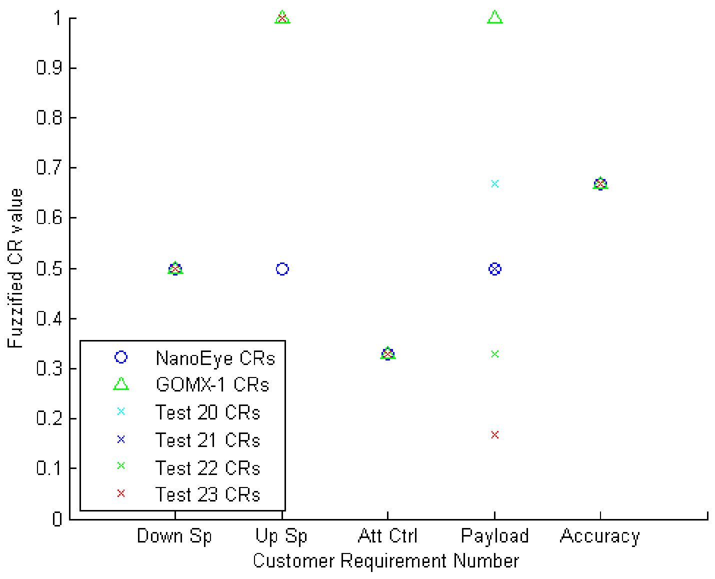


Figure 4.2: NanoCam, GOMX-1 and GOMX-1 Permutation's CRs

Examining the change in customer requirements, it can be determined that Test 21 reached a critical value, where the permutations reached a point where the similarity between category 1 (NanoEye) and category 4 (GOMX-1) was identical. The difference between the uplink speed had the same normalised value as the difference in the payload values. Tests 22, 23 and 24 all had lower normalised values for their payload and had a similarity score that was the same for both the NanoEye and GOMX-1 satellites. In this case, the FAM neural network returned the first category acceptable. From this experiment it can be deduced that while the neural network partially converged to be able to find satellites based upon overall similarity, but is unable to differentiate between different customer requirements.

4.3.3 Fuzzy logic. Experiment 2. Small changes in customer requirements

This experiment focuses on multiple small changes in customer requirements being identifiable with their original customer requirements. These changes are based upon data points in the training data. Rather than taking a single customer requirement for a data point and modifying it by all potential distances, this experiment takes two customer requirements for each training data point and shifts the customer requirements one or two steps either up or down. The purpose of this is to see what happens when a data point is heavily modified. This experiment also examines the effect when enough small changes are made to make the test data point closer to a different training data point than the one that it was based upon. Table 4.5 shows the customer requirements used, with the changed customer requirements italicised. The choice for the customer requirements being changed was based upon the customer requirements that had the greatest amount of diversity in the customer requirements while having a substantial affect upon the satellite's capability. For this reason, the uplink customer requirement was chosen in each satellite, with an uneven split between payload and attitude control being selected for changes.

Table 4.5: Fuzzy logic values for small changes in customer requirements

Satellite	Downlink	Uplink	Attitude	Wavelength	Detail
Test 1	Average	*Fast*	*Average*	Visual	Detailed
Test 2	Average	Fast	*Lenient*	*Near Infrared*	Average
Test 3	Average	*Extremely Slow*	Lenient	*Ultraviolet*	Vague
Test 4	Average	*Fast*	Lenient	*Thermal Infrared*	Detailed
Test 5	Average	*Slow*	Average	*Near Infrared*	Average
Test 6	Average	*Average*	*Precise*	Thermal IR	Detailed

The classification function returned only 66.67 percent of satellites matching their base satellite for the customer requirements in table 4.5 and the output is fully listed in table E.2. A misclassification is an indication that a set of customer requirements has been changed sufficiently to be identified with a different category of CubeSat. Test 1 and Test 4 were both misclassified, so the following is an examination of these two permutations. Figure 4.3 displays the customer requirements for the NanoEye satellite, which this permutation is based off, the COPPER satellite that the permutation was classified as and the customer requirements of the permutation itself.

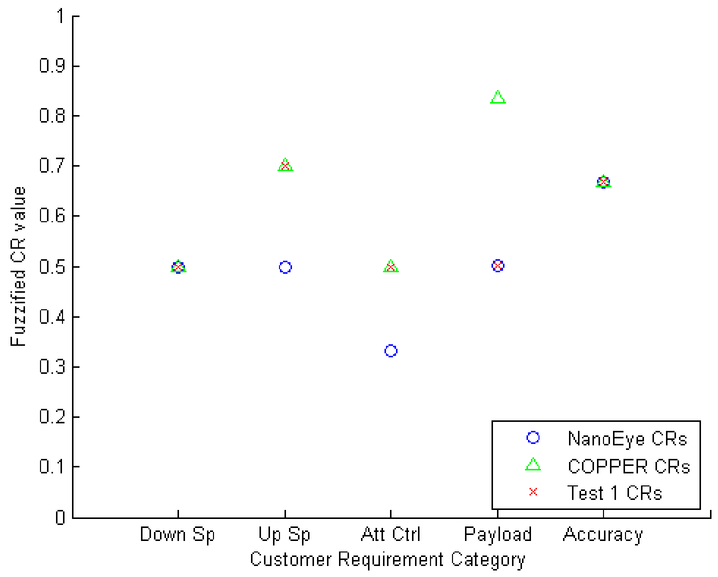


Figure 4.3: NanoEye, COPPER and Test 1 CRs

An examination of the customer requirements of both the NanoEye and COPPER satellites shows that they share the same value for the Downlink and Detail requirements. The permutation of the NanoEye satellite shares the same values for the Uplink and Attitude control requirements as the COPPER satellite, while the remaining category, the payload is unchanged from the NanoEye satellite. This results in the data point sharing four of six requirements with the COPPER satellite in comparison to only three requirements with the original NanoEye satellite. In this particular instance, the classification is correct, and the suggested classification as the satellite the permutation was incorrect due to the permutation sharing greater similarity with the COPPER satellite.

The second potentially misclassified permutation was the fourth test in this experiment and it was based upon the GOMX-1 satellite. The satellite that the permutation was classified as was the COPPER satellite. The customer requirements for the GOMX-1 and COPPER satellites are shown in Fig. 4.4 and are overlaid with the customer requirements for the fourth test.

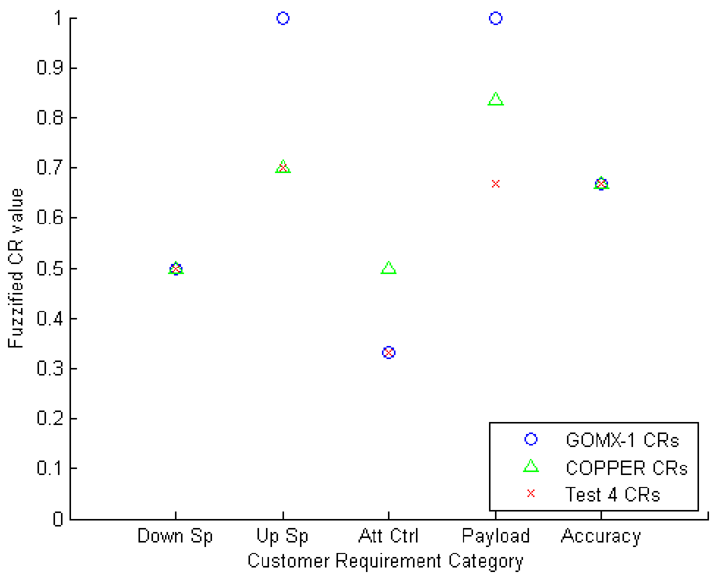


Figure 4.4: NanoEye, COPPER and Test 4 CRs

As was mentioned in section 4.3.2, the NanoEye and GOMX-1 satellites are extremely similar and share similar customer requirements. These shared similarities are also evident in the customer requirements with the COPPER satellite which results in the GOMX-1 and COPPER satellites having the same value for their Downlink and Detail requirements. The test four customer requirements match three requirements for both the GOMX-1 and COPPER satellites. However, this test case is different to test 1 where the customer requirements all matched a customer requirement value of one or the other training satellite. The payload value for test 4 has a different value to either training data point. As can be seen in figure 4.4 there is a greater similarity between the permutation and the COPPER satellite. This is another case of the classification function identifying the correct category.

4.3.4 Fuzzy logic. Experiment 3. Multiple consistent changes to customer requirements

The final fuzzy logic experiment has the largest number of customer requirements changed when creating the test data points. The data points were created by shifting every customer requirement one step or 'tier' in one direction. An example of shifting the downlink customer requirement up one tier would be changing a value from *Average* to *Fast*. These changes are only applied in one direction for each data point. So a test data point that is based upon the NanoEye satellite will have all its customer requirements shifted one tier higher. This experiment further tests the classification function and sees whether or not major changes have a substantial effect on the category a satellite is assigned to. Table D.1 has the complete list of permutations, however an extract is shown in table 4.6. This extract contains the two permutations based upon the NanoEye satellite customer requirements.

Table 4.6: Extract of fuzzy logic requirements for permutations of the NanoEye satellite

Satellite	Downlink	Uplink	Attitude	Wavelength	Detail
NanoEye +1 Tier	Fast	Fast	Average	Near IR	V. Det.
NanoEye -1 Tier	Slow	Slow	V. Lenient	Ultraviolet	Avg.

A graph of the NanoEye satellite's permutations is shown in Fig. 4.5. The two permutations maintain the shape of the original satellites customer requirements, however do not have a single requirement that is the same tier as the NanoEye satellite. Each permutation in the test set is created this way with the exception of those training data points that contain a customer requirement that is a minimum normalised value. The maximum value remains the same as the original satellite. This capped value occurs in the permutations based upon the GOMX-1 satellite.

The classification function only identified 33.3 percent of permutations as the satellite they were based upon, which results in nearly 70 percent of permutations having changed category. This result was not unexpected when the previous experiments were taken into consideration. The results of the classification function can be found in table E.3. The sixth test for this experiment was a permutation of the COMPASS-1 satellite created by lowering the customer requirements by a 'tier'. This test is a typical example of a misclassified permutation and is shown in figure 4.6. Each of the misclassified permutations in this experiment all had more customer requirements in common with the training set category they were identified with in comparison to the satellite they were based upon.

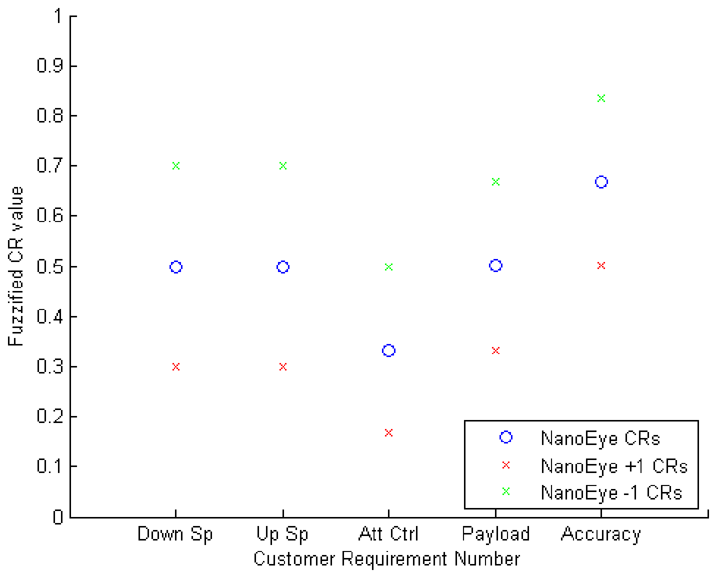


Figure 4.5: NanoEye, COPPER and Test 4 CRs

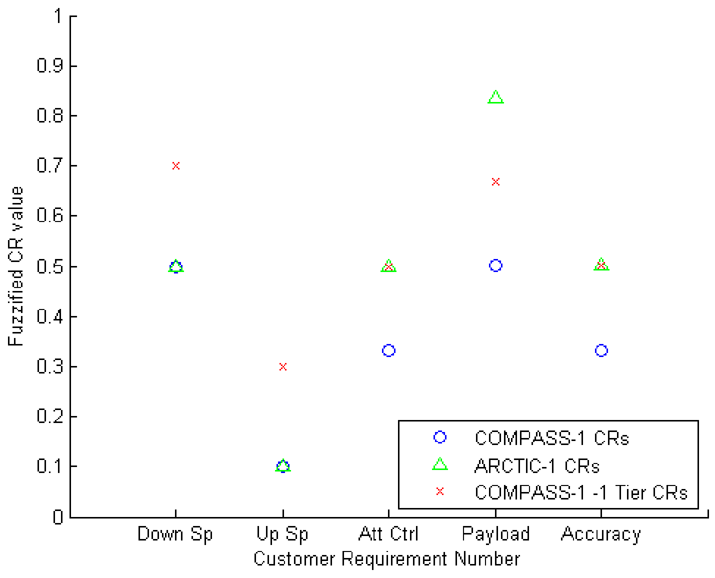


Figure 4.6: NanoEye, COPPER and Test 11 CRs

4.3.5 Results of fuzzy logic analysis

The experiments conducted above have shown that the Fuzzy ARTMAP algorithm did not fully converge to a solution, however a partial convergence was achieved from the training data set. A match was found for every test data point, which shows that the algorithm did not fail. These findings lead to the conclusion that FAM algorithm as implemented is unable to be utilised within the genetic algorithm. However, it shows that utilising fuzzy logic to encode natural language customer requirements is possible. These quantified customer requirements can be used to create a linkage between the functional and behavioural levels of design as is required according to the FBS ontology discussed in section 2.2.2.

4.4 Testing the creation of valid satellites

The genetic algorithm utilised for the following experiments is explained in section 3.5. This algorithm utilises a single data point vector of customer requirements as the goal states for each metric. The other input values are the population size, the number of generations and the mutation rate. Each data point has a single run of the genetic algorithm. The output of the algorithm is the final population generated. The first eleven ranked satellites in the population are seeded into the population, the first is the overall minimum score for the population. The next ten satellites are each the minimum of their respective metrics. The remaining satellites are ranked in order of the number of metrics that have achieved a goal state and then ties between satellites are broken by the minimum distance found.

Manual analysis of the results will be performed, along with an evaluation of performance goals per generation. The ideal state will be a 'distance' score of zero, the worst possible score will be ten. This worst possible score is generated by every metric in a satellite being the maximum distance from the goal state. Each distance is in the range [0, 1]. An indicator that a genetic algorithm is working correctly is that the performance of the best solution per generation improves, or the in the case of multi-objective algorithms, that the fitness trend improves across the whole population. Bumps in the overall fitness are expected when certain metrics improve, but at the cost of other metrics in the individual.

All of the following experiments use these additional settings : Population of 100. 100 generations and a mutation rate of 30%.

4.4.1 Genetic algorithm. Experiment 1. Recreating a past satellite

The purpose of this experiment is to take a data point from the test set, and see if the genetic algorithm is able to be recreated with the same components when given the same customer requirements. This acts as a validation of the metrics. The available satellites that can be used as a goal in this test are limited to satellites that use only commercially available components. This leaves the options from the training data set as the two GOMspace offerings [38]. These are the GOMspace NanoEye and the GOMX-1 satellite. Both are made with parts available and have their performance specifications published online. This makes them reasonable test subjects. The customer requirements for these two satellites are shown in table 4.7.

Table 4.7: Customer requirements for GOMspace satellites

Satellite	Downlink	Uplink	Attitude	Wavelength	Detail
NanoEye	Average	Average	Lenient	Visual	Detailed
GOMX-1	Average	Ex. Fast	Lenient	Radio	Detailed

These linguistic variables are converted into their numerical values before being applied to the genetic algorithm utilising the methods in section 3.3. The first satellite recreation to be attempted was the NanoEye satellite. The overall performance of the satellite with the minimum fitness score is shown in figure 4.7 for each generation. Initial analysis of the performance curve shows the expected shape for a well performing genetic algorithm. The ideal curve will approach a desired state, which for this implementation is zero.

The initial indication is that the genetic algorithm is working as expected, however the graphs in figure 4.7 only show the overall average and the performance of a single satellite each generation. As this genetic algorithm is designed to be pareto optimal, analysis of each metric needs to be performed. Figure 4.8 shows the performance of each generation for the four essential metrics, 'Volume', 'Mass', 'Avionics' and 'EPS'.

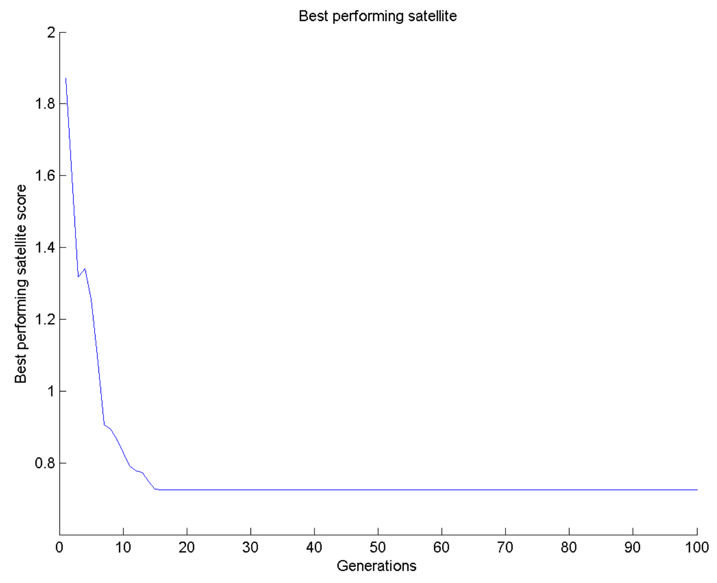


Figure 4.7: Minimum scored satellite over generations in recreating the NanoEye Satellite

As expected, the graph shows a decrease or steady state for all essential metrics in each generation. A notable feature of the graph is that the volume metric was zero for all generations.

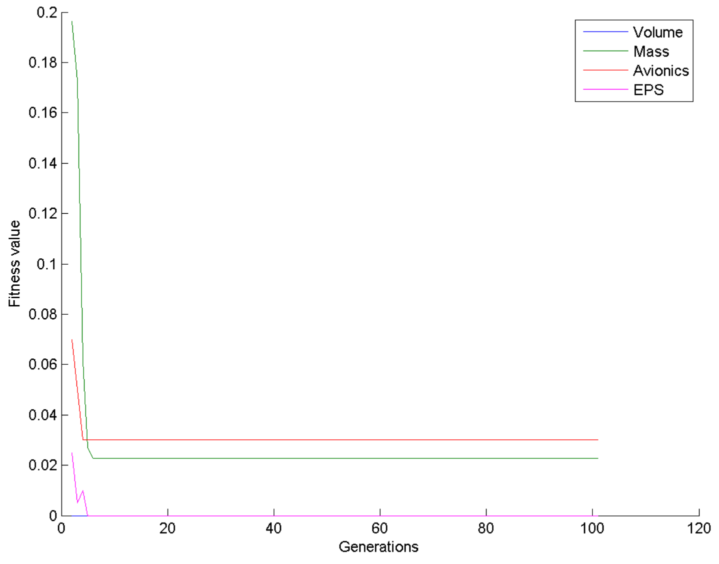


Figure 4.8: Essential metric performance, in recreating the NanoEye Satellite

The metrics aligned with the customer requirements are shown in figure 4.9. All bar one metric trend towards an ideal state. The Payload detail metric trends upwards as the Payload metric improves. Comparing this to the components database, there is only one payload option that can fulfill the customer requirement for the desired payload, the 'NanoCam C1U'. The detail that this camera shows is not as high quality as the customer requirement that it was assigned, thus this shows an error in the formulation of the customer requirements in the NanoEye satellite.

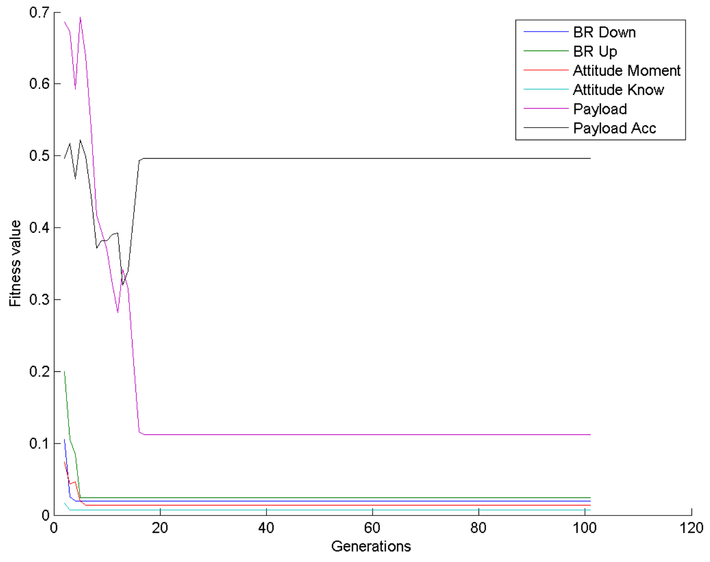


Figure 4.9: Customer specific performance, in recreating the NanoEye Satellite

As the metrics performed as expected within the genetic algorithm, the judgment is reserved for whether a satellite has been generated by the genetic algorithm that is a reasonable conceptual design. Table 4.8 shows the components in the highest ranked satellite in the population that was not seeded in for an individual metric.

The differences between the two satellites are minor. They include different manufacturers for the systems but in general it successfully creates an initial conceptual design. This satellite however would not work as implemented due to the following reasons. The SX450G requires an additional modem and there is no antenna release board. This shows a lack of a comprehensive metric for the communications system, at no point does the metric consider the need for a modem and assumes that one is included.

Table 4.8: Comparison of Generated CubeSat vs NanoEye

	NanoEye	**Generated Satellite**
Payload	NanoCam C1U	NanoCam C1U
EPS	P31U Power Supply	CS-1U5EPS2-10
	P110 Solar Panels	P110 Solar Panels
Avionics	A712D Cubesat Computer	Q6 Processor Board
	NanoHub	
Communications	U482C UHF Transceiver	SX450G
	ANT430 - Release Board	SX450G
Attitude Control	GOMSpace Att. ctrl. system	CubeSat Sun Sensor
		2xCubeSat Magnetorquer Rod
CubeSat Size	1U	1.5U

Further to this, no antenna release board, or antenna was included. However, for early in the conceptual design stage, this is not a critical failure and can be corrected by an engineer. The sizes are also slightly different between the two satellites, internal space in a CubeSat is at a premium and the method of generating components for a satellite was via filling slots, rather than making the slots as close together as practicable, thus it's expected that correct placement of components would save space and shrink the required structure size.

The second test for this experiment was attempting to recreate the GOMX-1 satellite by utilising the customer requirements listed in table 4.7. The performance for the minimum scoring satellite over generations is shown in figure 4.10. As expected the performance trended towards a minimum as in the previous test.

The essential metrics for the GOMX-1 recreation test are shown in figure 4.11. There are some unexpected anomalies in this graph. While the metrics trend towards their optimum value there are upward spikes in the metrics. However these are averages in a multi-pareto population. It is possible for an individual metric to suffer, or in this case a group of metrics if a different metric is being improved.

Analysing the remaining metrics in figure 4.12 shows that there are dramatic average drops in multiple metrics where the majority of spikes in the essential metrics occurs. The most likely cause of this is optimisation of one metric over another that has an impact over the short term. However, there is a dramatic spike in the payload accuracy

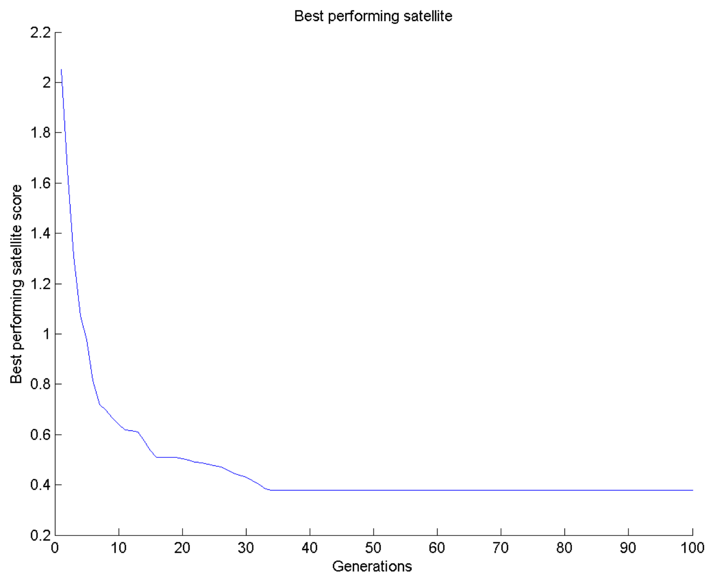


Figure 4.10: Minimum scored satellite over generations in recreating the GOMX-1 Satellite

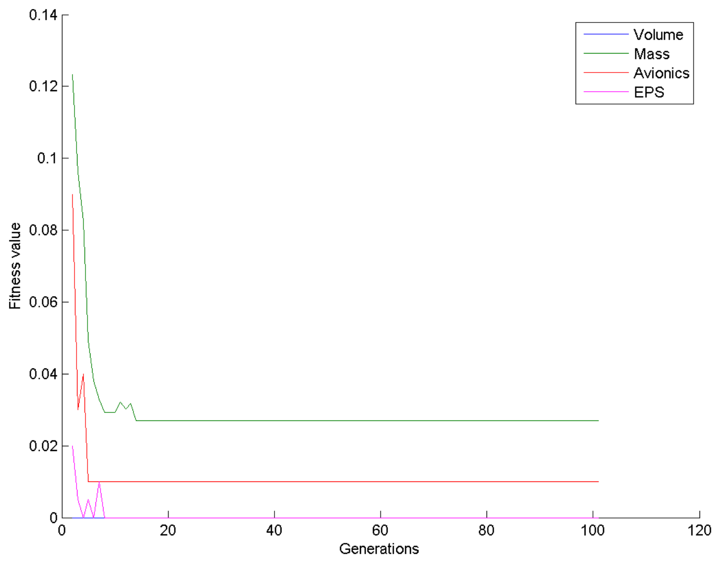


Figure 4.11: Essential metric performance, in recreating the GOMX-1 Satellite

metric.

It spikes approximately when the average payload metric achieves its minimum value. After the spike in the payload accuracy metric it continues to trend downwards which

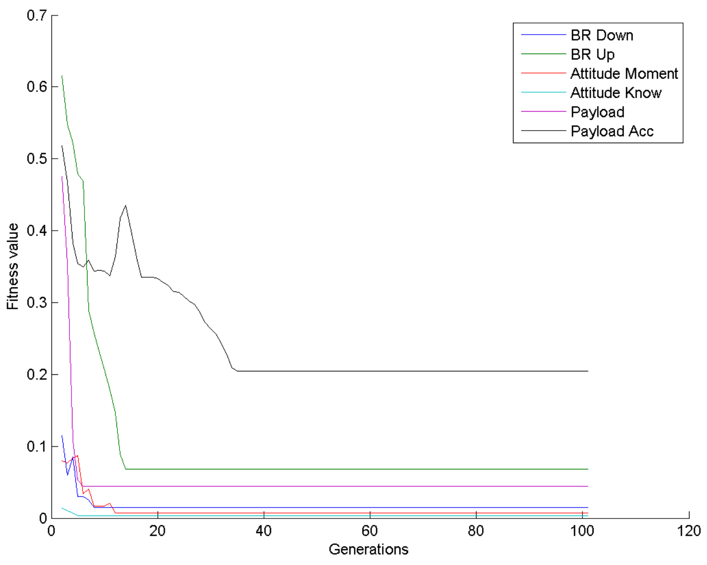


Figure 4.12: Customer specific performance, in recreating the GOMX-1 Satellite

leads towards further investigation. Considering the components for the highest ranked satellite in the generated population in comparison to the original GOMX-1 satellite as shown in table 4.9 there is the addition of a different secondary payload.

Table 4.9: Comparison of Generated CubeSat vs GOMX-1

	NanoEye	**Generated Satellite**
Payload	GOMX SDR	GOMX SDR
Secondary Payload	NanoCam C1U	Flux-O-Probe QB50
EPS	P31U Power Supply	CS-1UEPS2-10
	P110 Solar Panels	P110 Solar Panels
Avionics	A712D Cubesat Computer	A712D Cubesat Computer
	NanoHub	ISIS On-board Computer
Communications	U482C UHF Transceiver	SX450G
	ANT430 - Release Board	ISIS VHF down UHF up
		ISIS TXS S-band down
Attitude Control	GOMSpace Att. ctrl. system	CubeSat Sun Sensor
		ISIS Magnetorquer Board
		2xMagnometer
		Magnetorquer
CubeSat Size	2U	3U

The secondary payload was randomly selected to be the Flux-O-Probe, a component utilised in the QB50 project. Overall, the differences between the generated satellite and the GOMX-1 satellite is in the secondary payload and the communications system. It also has an additional flight computer as opposed to the NanoHub that GOMSpace utilises. Considering that a secondary payload was not specified, the most logical explanation for the presence of the Flux-O-Probe is to improve the payload accuracy metric. The genetic algorithm does not consider the source of the accuracy in the payload, thus it utilises the extra space available in the larger structure to improve the metric, which explains the spike in the accuracy metric and the subsequent trend towards zero. The upward spike occurs as the correct payload is selected for in the population, at the expense of the payload accuracy before the Flux-O-Probe is selected for in addition to the GOMSpace SDR payload.

The result of these two tests is that while they were not perfect recreations of the original satellites, they are decent conceptual designs. The time taken to run both of these tests was approximately half an hour and each population included an additional 99 satellite designs to investigate.

4.4.2 Genetic algorithm. Experiment 2. Modification of a past satellite for a new purpose

This experiment is conducted by changing a single customer requirement from a past satellite mission. This experiment allows an evaluation of the effect the change of both a single customer requirement and three customer requirements can have on the design of the satellite. The chosen satellite to be changed was the GOMspace NanoEye satellite from the experiment in section 4.4.1 to allow for a direct comparison. The chosen customer requirement was the payload. Each then had the uplink and downlink speeds changed to reflect the possibility of having data transmission speeds altered, this allows for a comparison with only a single change to customer requirements. The differences for this secondary change was increasing the downlink speed to reflect a higher information download speed and reducing the uplink speed to a level that handles only commands. The linguistic input into the algorithm is shown in table 4.10.

These satellites investigate changes on the electromagnetic wavelength and how the

Table 4.10: Customer requirements for modified satellites

Satellite	Downlink	Uplink	Attitude	Wavelength	Detail
A1	Average	Average	Lenient	Near Infrared	Detailed
A2	Fast	Very Slow	Lenient	Near Infrared	Detailed
B1	Average	Average	Lenient	Thermal Infrared	Detailed
B2	Fast	Very Slow	Lenient	Thermal Infrared	Detailed

algorithm copes with the differences in payloads. The individual permutations tested are based upon the NanoEye and COMPASS-1 satellites. Permutations A1 and B1 are derivations of the NanoEye satellite and the remaining two permutations are based off the COMPASS-1 satellite. The first test of this experiment alters the required payload from the Visual requirement to Near Infrared. The best performing satellite is shown in figure 4.13. As expected, the best performing individual for each generation trends towards a local minimum.

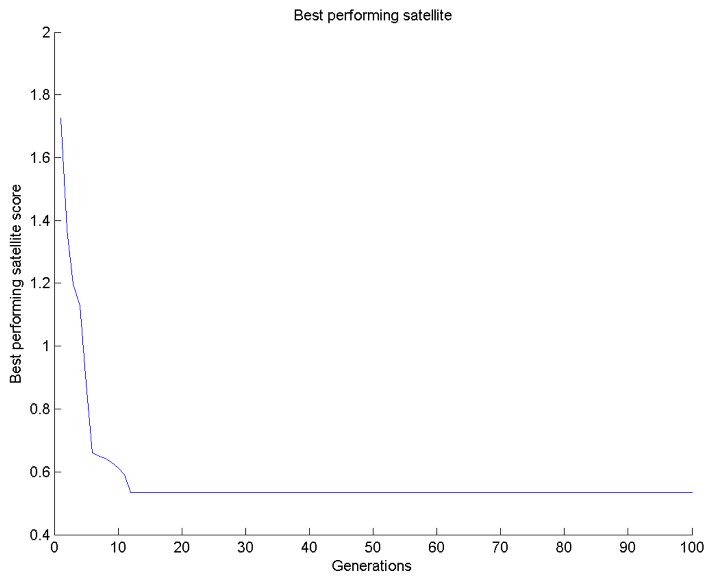


Figure 4.13: Best performing individual satellite for permutation A1

The essential metrics had early trending towards the steady state value with the exception of the electrical power system (EPS) as shown in figure 4.14. These performed exactly as expected, with the spikes in the EPS settling once new arrangements had been discovered by the algorithm and optimised.

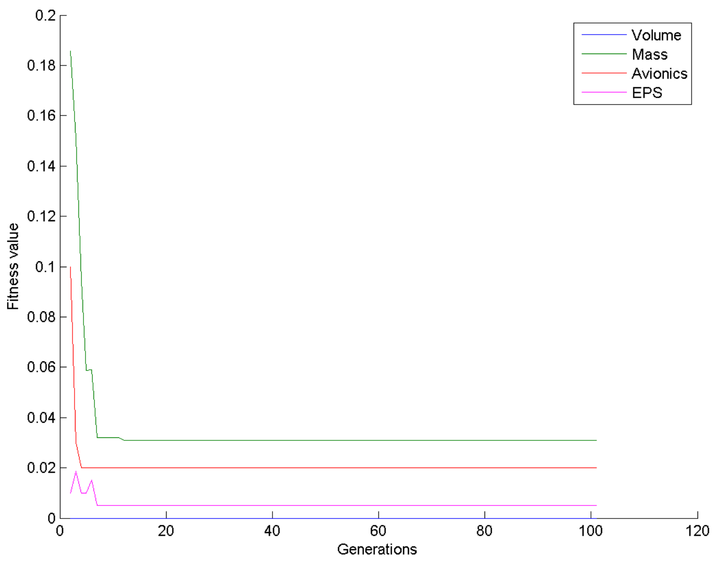


Figure 4.14: Essential metric performance for permutation A1

The graph in figure 4.15 shows a difference to the results in both tests of experiment one. The payload is the only metric to have a poor performance. Unlike the recreation experiment, where the payload metric dominated the payload accuracy metric, the reverse is true for this permutation.

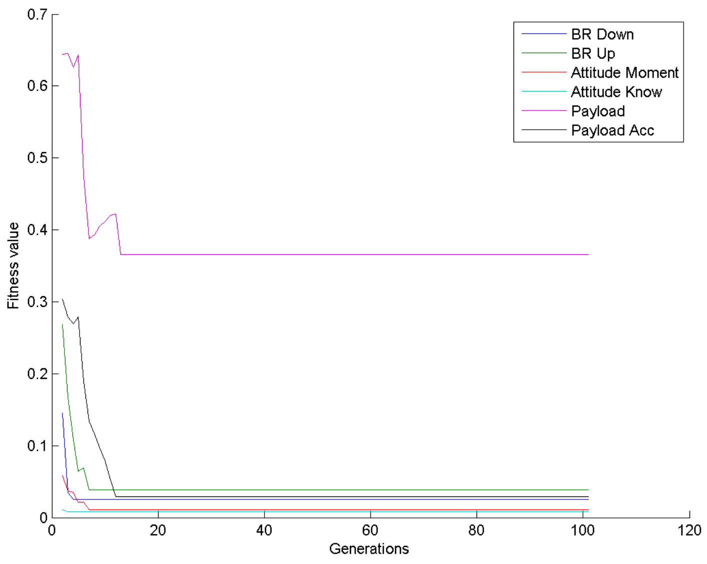


Figure 4.15: Customer specific performance for permutation A1

A search of the final population shows that not a single design contained a near infrared payload. Due to the way that a satellite in the population is seeded into the next generation for each metric, it can be assumed that a Near Infrared compatible component was not generated. As there are 76 components in the database, each component selection has an independent probability of 1.32% of being generated. This leads to the conclusion that further attempts could potentially generate a successful design.

The behaviour of the minimum scoring satellite and the essential metrics for permutation A2 trended towards a minimum steady state value as expected and are shown in figures F.1 and F.2 in appendix F. An examination of the plot in figure 4.16 shows that similar to permutation A1, the payload failed to dominate the population. There was a spike slightly before the twentieth generation and then continued to trend toward the steady state score.

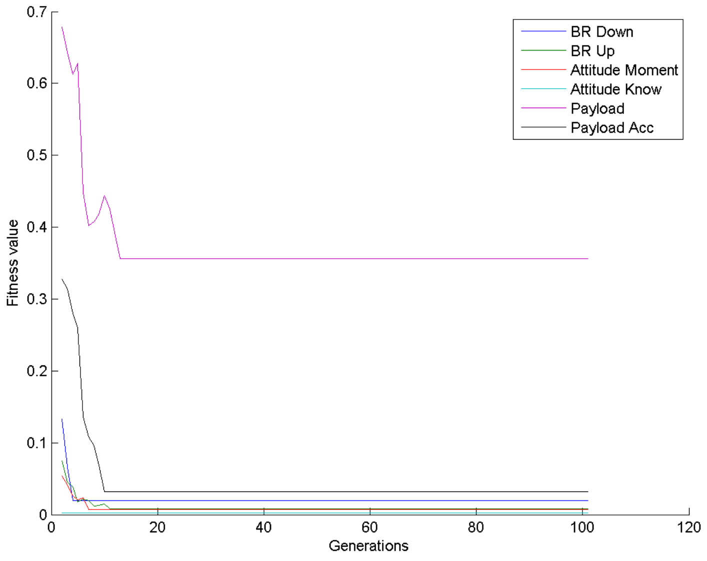


Figure 4.16: Customer specific performance for permutation A2

Unlike the test of permutation A1, this test generated a satellite that had a payload in the near infrared range. The two generated satellites were a seeded satellite for a minimum score on a single metric and a single satellite that existed within the population at the end of the one hundred generations and the components for both satellites are shown in table 4.11.

Table 4.11: Minimum scoring satellite for permutation A2

	Generated Satellite Seed	**Generated Satellite Pop.**
Payload	Argus 1000 Infrared	Argus 1000 Infrared
Secondary Payload		GOMX SDR
EPS	P31US Power Supply	CS-1U5EPS2-10
	CN-SWT-0035-CS	CN-SWT-0035-CS
	CS-1U5EPS2-20	
	CS-SBAT2-10	
	CS-SBAT2-30	
	P110 Solar Panels	
Avionics	Cube Computer	Cube Computer
Communications	Null	NanoCom U482C UHF
		ISIS VHF down UHF up
		ANT430 - Release Board
Attitude Control	Magnometer	2x SSOC A-60 Sun Sensors
	1-axis Propulsion System	Magnometer
	MAI-201 Reaction Wheel	CubeSat Sun Sensor
	CubeSat Sun Sensor	
CubeSat Size	3U	3U

In the table, this satellite has generated a secondary payload much like the recreation of the GOMX-1 satellite. This, combined with the poor performance of the payload metric indicates that the GOMX SDR is being utilised to lower the accuracy so that the satellite has the ability to compete with other satellites. However, success in a single metric is not enough to dominate the entire population. This generated design contains all essential components for a conceptual design and can be considered a successful test.

The latter two tests for the permutation experiment both seek to generate satellites with payloads that take readings of thermal infrared wavelengths. The first, permutation B1, is based upon the NanoEye satellite. The customer requirement metrics are shown in figure 4.17 and the payload metric did not dominate the other metrics. There is a spike away from optimal before it settles into a steady value.

An examination of the population shows that two thermal infrared capable satellites were generated. However, both were in the seeded results. The first utilised the ARCTIC-1 thermal camera, and was the seeded result for the mass metric. The second possessed the Tau FLIR camera. The second satellites components are listed in table 4.12.

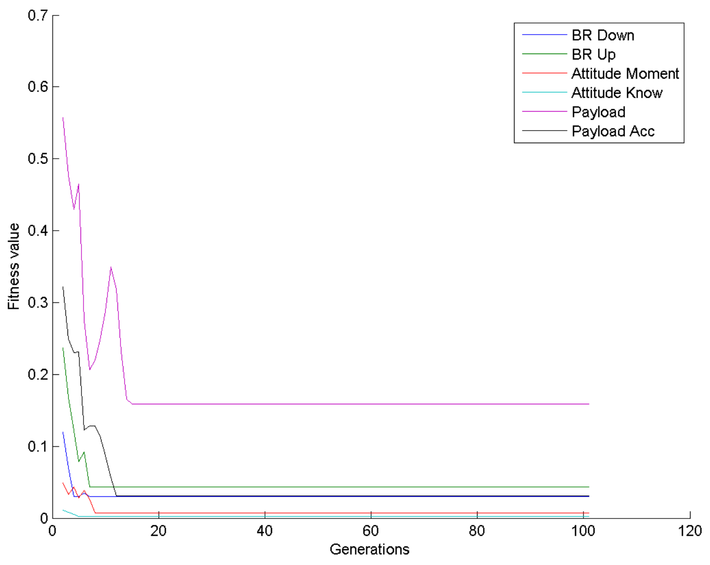


Figure 4.17: Customer specific performance for permutation B1

Table 4.12: Generated satellite for permutation B1

	Generated Satellite
Payload	Tau FLIR
Secondary Payload	None
EPS	CS-1U5EPS2-NB
	CS-2UEPS2-20
	CS-SBAT2-30
	P110 Solar Panels
Avionics	160 High Perform Flight Comp
Communications	ANT430 - Ant. release board
	No transceiver
Attitude Control	1-axis Propulsion System
	SSOC D-60 Sun Sensor
	Magnometer
CubeSat Size	2U

The only thing missing from the satellite displayed in table 4.12 is a communications subsystem. However, there are two EPS subsystems present, one of which could potentially be swapped for a transceiver.

The final test for this experiment was a permutation based upon the COMPASS-1 satellite with an altered payload. The customer requirement for the payload was in the thermal infrared wavelength. The metrics for the specified customer requirements are shown in figure 4.18. The metrics converge within thirty generations. In comparison to earlier tests, the payload metric performs much more effectively. The performance of the payload metric across the entire population was less than 0.1 on average, however in the attempt to generate a satellite with permutation B1, the payload metric performed poorly and has a score over twice that of permutation B2.

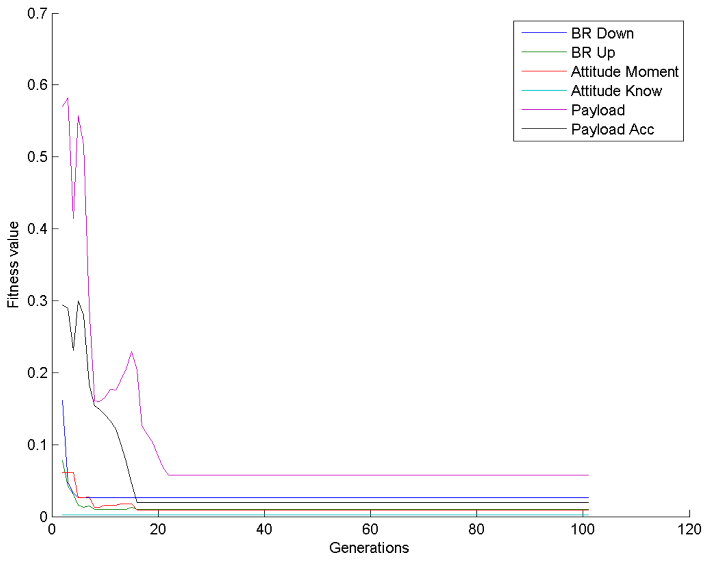


Figure 4.18: Customer specific performance for permutation B2

The metric performance when compared against test B1, which had the same payload requirement, shows that it is possible for the algorithm to converge. Examining generated population, it was clear that the Tau FLIR component had dominated the population. The only generated satellites in the population that did not contain the Tau FLIR component were seeded results for competing objectives. Table 4.13 contains a sampled satellite.

This satellite shows a complete list of components for a single purpose satellite. Every sub-system has a component designed to fulfill the system purpose. This test has shown that for a variation on past customer requirements, it can autonomously generate a working satellite. This test can be classified as a success.

Table 4.13: Generated satellite for permutation B2

	Generated Satellite
Payload	Tau FLIR
Secondary Payload	None
EPS	CS-2UEPS2-20
	P110 Solar Panels
Avionics	Q6 Processor Board
	Cube Computer
Communications	NanoCom U482C UHF Half-Dup
	ISIS TXS S-band down
	ANT430 - Ant. release board
Attitude Control	2x SSOC A-60 2 axis Sun Sensor
	Digital Fine Sun Sensor
CubeSat Size	2U

4.4.3 Genetic algorithm. Experiment 3. Creation of a satellite from new customer requirements.

The final experiment utilised for the evaluation of the genetic algorithm was to create a new set of customer requirements for a new purpose. These customer requirements were generated from the QB50 project's mission requirements [52]. These mission requirements involve upper atmospheric measurements of the magnetosphere and the ionisation of particles in low earth orbit and are shown in table 4.14. This experiment attempts to generate a starting point for a satellite to perform these missions. An individual run of the genetic algorithm is performed for each set of customer requirements and is then compared against the real world solution to this problem provided by GOMspace [38].

Table 4.14: Customer requirements for a new satellite

Satellite	Downlink	Uplink	Attitude	Wavelength	Detail
QB50 Mag Flux	Slow	Very Slow	Lenient	Mag. Flux	Average
QB50 Ion	Slow	Very Slow	Lenient	INMS	Average

The first test of this experiment was conducted by attempting to create a workable satellite for the QB50's magnetic flux mission. The metrics shown in figure 4.19 converge in slightly under twenty generations. There is a very quick convergence for the population, the value for the payload metric is the worst performing and has a value less than 0.1, which previous tests have shown to generally have good overall results. As this is a test of real world customer requirements, the comparison with a real world design is to be done. The components for each design are laid out next to one another in table 4.15.

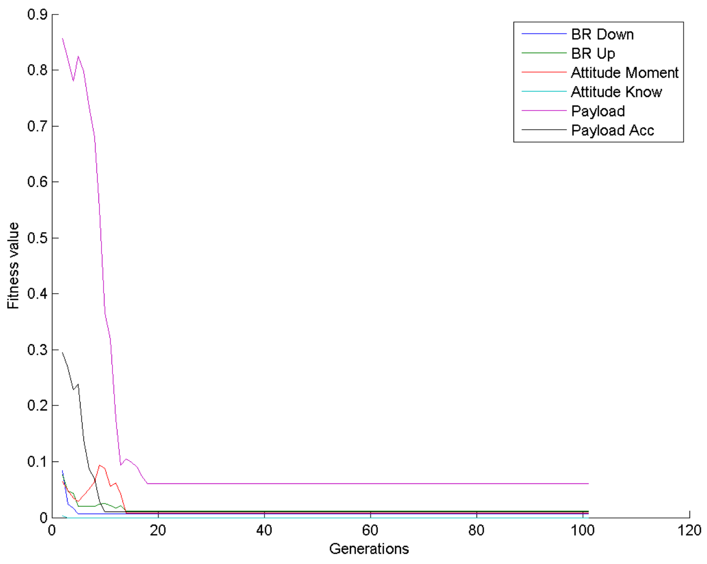


Figure 4.19: QB50 Mag Flux customer specific metric performance

Examining the generated satellite, all the major sub-systems have one or more relevant components. This appears to superficially fulfill the customer requirements set for the QB50 mission. A critical failure would have been if the payload had not been correctly selected. The Flux-O-Probe is the QB50 specific payload for this mission and was specifically required. The remaining sub-systems are essential to the successful operation of the satellite.

The electronic power system on the generated satellite is made up of two EPS boards, and a battery pack. Both of the EPS boards are made by alternate manufacturers, while the NanoPower BP4 is a battery pack made by GOMspace. These are matched up against the P31U Power Module from GOMspace with integrated batteries. A downside to this generated design is the additional EPS board, but otherwise fulfills the requirements for the EPS sub-system.

Table 4.15: Algorithm generated satellite vs GOMspace satellite for the QB50 Magnetic Flux mission

	Generated Satellite	**GOMspace QB50 satellite**
Payload	Flux-O-Probe	Flux-O-Probe
EPS	CS-3UEPS2-NB	P31U Power Module
	NanoPower BP4	
	CN-SWT-0035-CS	
	P110 Solar Panels	P110 Solar Panels
Avionics	A712D Cubesat Comp.	A712D Cubesat Comp.
	2xQ6 Processor Board	NanoHub
Communications	2xU482C UHF Half-Duplex	U482C UHF Half-Duplex
		ANT430 Ant. Release Board
Attitude Control	2x SSOC A-60 Sun Sensor	GOMSpace Att. ctrl. system
	2x 1-axis Propulsion System	
	Digital Fine Sun Sensor	
	Z-axis Magnetorquer	
	Magnometer	
CubeSat Size	3U	2U

The avionics sub-system for both satellites utilises the same flight computer, however, the GOMspace offering has the NanoHub in addition to the computer. The NanoHub is designed to assist in integrating the flight computer with the other sub-systems, including the QB50 payload. However, the metric does not take compatibility between components into consideration. The Q6 processor boards can easily be exchanged for a lower power NanoHub or equivalent. These changes are outside the scope of the algorithm in this thesis and are considered to be a successful selection.

The communications system contains the same transceiver in both satellites. The GOMspace satellite also contains the ANT430 antenna release board, which is utilised after the satellite is launched but is not accounted for in the communications metrics. The generated satellite has an additional transceiver of the exact same variety, which could be replaced with the ANT430 and would lower power usage. This sub-system is considered to be successfully generated for this satellite.

The final sub-system for the satellite is the attitude control system. The standardised GOMspace attitude control system is magnetically actuated and utilises both an inertial measurement unit (IMU) built into the flight computer and sun sensors to detect current attitude. The solar panels contain magnetorquers for the actuation. In comparison, the generated satellite contains the same IMU and magnetic actuators in the solar panels. There are three sun sensors that have been generated, in addition to a magnetometer that can be utilised to detect the orientation of the Earth's magnetic field. It also contains small thrusters that have the potential to conduct orbital manoeuvres. Overall, with the correct optimisations performed by an engineer, the generated satellite has all the capabilities of the GOMspace satellite. This sub-system was successfully generated.

Each of the sub-systems have been compared with the GOMspace satellite and in general fulfill an early conceptual design for the satellite. While the algorithm generated a solution that isn't as optimal as the solution that a human engineer would have designed, it provides a starting point for refinement into an improved final design. The size difference, while a significant factor does not disqualify the satellite from consideration.

The final test of the genetic algorithm was conducted upon a set of customer requirements for another QB50 scientific experiment. The only difference between the two sets of customer requirements is the payload required. The performance of the customer specified metrics is in figure 4.20. All of the metrics for the population converge to their final average value converge before the twentieth generation. The performance of the payload metric was worse than in the previous test, however was still had a mean value under 0.2.

Examining the final population showed that the QB50's Ion and Neutral Mass Spectrometer (INMS) was the dominant payload in all non-seeded satellites, it was present in every individual satellite. A sample satellite was selected for analysis and comparison to the Gomspace satellite, shown side-by-side together in table 4.16. The first requirement checked is the payload. As the INMS dominated the solutions in the population, the likelihood that it would appear in the sample was extremely high.

The EPS for the generated satellite is formed by the solar panels, a Clyde Space EPS board and additional battery pack also created by Clyde Space. There is only a single power management system, with extra battery capacity. This sub-system achieves all

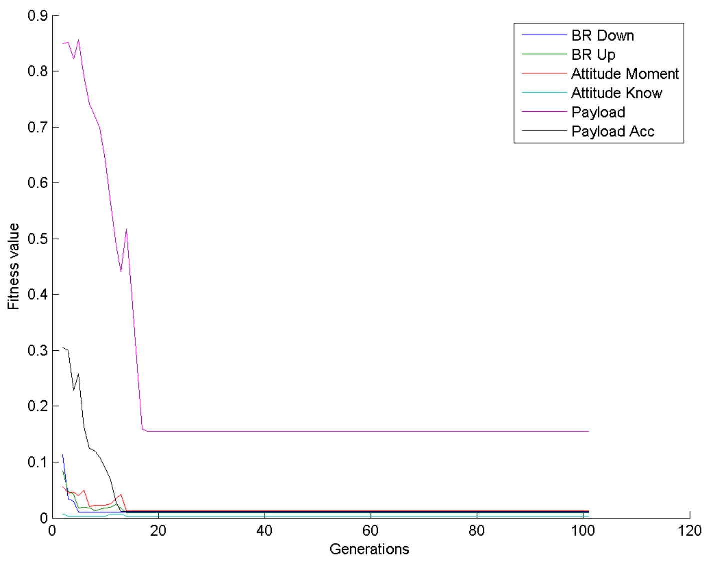


Figure 4.20: QB50 Ion and Neutral Mass spectrometry customer specific metric performance

Table 4.16: Algorithm generated satellite vs GOMspace satellite for the QB50 Ion and Neutral Mass Spectrometry mission

	Generated Satellite	**GOMspace QB50 satellite**
Payload	QB50 INMS	QB50 INMS
EPS	CS-2UEPS2-20	P31U Power Module
	CS-SBAT2-10	
	P110 Solar Panels	P110 Solar Panels
Avionics	A712D Cubesat Comp.	A712D Cubesat Comp.
	160 Flight Computer	NanoHub
	Q6 Processor Board	
Communications	U482C UHF Half-Duplex	U482C UHF Half-Duplex
	ANT430 Release Board	ANT430 Release Board
Attitude Control	Digital Fine Sun Sensor	GOMSpace Att. ctrl. system
	SSOC D-60 Sun Sensor	
	CubeSat Sun Sensor	
CubeSat Size	2U	2U

the goals required and is entirely compatible with itself. From the limited information contained in the datasheet for these components, the EPS could be utilised without modification in a final design.

The avionics sub-system for the generated satellite utilises the same flight computer as the GOMspace solution. However, much like the previous test, the limited metric results in multiple components being selected for the sub-system. Any one of the listed options would be sufficient, with the power requirements and space freed up for an interfacing board.

A comparison of the communications sub-system between the generated satellite and the GOMspace satellite shows that the option selected by the satellite was identical to the communication sub-system developed by GOMspace. The final sub-system is the attitude control sub-system that is also extremely similar to the GOMspace option, utilising the IMU on-board the flight computer for attitude detection and the solar panels to generate torque. The knowledge of the satellite's attitude in relation to the earth is provided in the generated satellite by the three sun sensors. This solution is similar to the GOMspace option, however utilises components created by competitors.

The satellite generated in the final test of the algorithm achieves the same size as the real world satellite that it is being compared against and fulfills the customer requirements that were input into the algorithm. This test is considered to be extremely successful.

4.5 Summary

The experiments in this chapter were designed around answering the two questions posed in section 2.3. They were "Can the levels of design be linked autonomously by a computer?" and "Can a evolutionary algorithm create sensible system level designs?" To answer the former question, fuzzification was utilised to quantify the customer requirements at the functional level of design and create an association with metrics generated from product specifications which are within the behavioural level of design. A neural network was utilised to ensure that when customer requirements were changed that the input remained value and was able to be associated with previous designs based on similarity. This similarity to previous designs was to check confirm that the fuzzification process worked. The randomised creation of satellites by the genetic algorithm ensured that the structural level of design was present. The results from the genetic algorithm experiments show that it is possible for a satellite to be created that has structural components, with a derived behaviour which was represented by the metrics and the metrics were compared to the customer requirements. This completes the linkage between all design levels.

The latter question was answered with the experiments, where the generated satellite designs were reasonable approximations of past satellites. There were tests that failed, however, with a stochastic search algorithm such as a genetic algorithm, there is no guarantee that the optimum design will be found. The second test in the third experiment for the genetic algorithm answered the question emphatically, with it possessing only minor differences to a human designed satellite.

Chapter 5

Conclusion

5.1 Summary of contributions

This thesis utilises a design framework for use within computer assisted conceptual design, methods to create linkages between design levels and how to quantify customer requirements. An evolutionary algorithm is used within the design framework to generate potential conceptual designs and select the highest performing individuals in the population to propagate to the next generation. Limitations and future work considers the drawbacks of this implementation, the improvements that could be made to improve performance and achieve a goal state. The primary contribution of this thesis is establishing that it is possible for a computer to create early stage conceptual designs at the system level rather than just individual components.

5.1.1 Artificial intelligence in the design framework

The Functional-Behavioural-Structural ontology was explained in chapter 2 and formed a framework for the design process. This thesis explored methods to automate processes within the FBS structure and utilised tools such as fuzzy logic to create linkages between design levels. Once the customer requirements were quantified and able to be compared to product specifications via metrics, a genetic algorithm was capable of generating designs and judging each design's fitness in comparison to the customer requirements. Between the fuzzy logic and genetic algorithm, the structure of the FBS ontology is achieved and links the customer requirements to a design.

5.1.2 Benefits of computer aided conceptual design

The genetic algorithm experiments that were conducted in this thesis took approximately half an hour to create a population of one hundred potential designs. Component selection in conceptual design is a process that often has a duration in excess of a week. This improvement is orders of magnitude faster than current methods. These designs often had a similar score but contained different combinations of components, unless one component was significantly superior for a purpose. This is a significant time saving measure, in addition to providing a list of components to investigate. Each component's entry into the database was created from the vendor's data sheet and allows a designer to quickly and efficiently start evaluating potential designs.

5.1.3 The genetic algorithm

Chapter 3 explained the metrics involved in evaluating the components for a satellite, the generation of individual satellites from the components and how to generate child populations by selecting the best performing satellites from each generation and allowing them to propagate through generations. The experiments chapter 4 showed that not only was this possible, but the genetic algorithm was capable of creating satellite designs similar to that of a human designer. The algorithm was able to recreate past satellites, variations on past satellites and in a real world application created a CubeSats to fulfill the requirements set for the QB50 mission. These tests were in the majority of cases successful. The following section will elaborate upon flaws in the current algorithm and potential improvements.

5.2 Limitations and future work

Much of the error in the genetic algorithm experiments was caused by vague metrics. An example of this is the avionics metric, which was purely binary based upon whether there was a flight computer present or not. Fortunately, due to the standardised nature of the components for CubeSats, this did not cause extensive failures that couldn't be corrected. This thesis treated the satellite as a whole system and left the arrangement of the components to a human designer for the finalised design, however this could be improved in future versions.

5.2.1 Improving satellite generation

The creation of individual satellites for the genetic algorithms population did not consider the placement of components, only considering the amount of 'slots' that a component required and the remaining slots in the generated satellite's structure. Nor did the algorithm consider alternate arrangements, such as placing component boards against the edge of the satellite's walls around a central component.

A system for accurately modeling a satellite to ensure every component fits correctly would allow greater flexibility within the design and improve the metric analysis. The accurate modeling would consider placements and individual sizes of components, which in turn would provide more knowledge for being able to consider moments of inertia, allowing for precise attitude calculations. The modeling would also ensure that a designer would require less time to create a full conceptual design from the finalised solution, being supplied with the exact placement of components.

5.2.2 Improving satellite metrics

The metric analysis within this thesis was necessarily vague, due to lack of detail in both the modeling of the satellite and the insufficient detail for most components. Few component data sheets included inertial data, nor a minimum clearance from other components. However, methods exist for evaluating the theoretical values for the combination once all the information is known. This would make it possible to improve the mass, volume and attitude control metrics.

The avionics metric was a binary metric that only checked for the presence of a flight computer. Research and development into modeling different forms of flight computers, such as boards with a full operating system versus a programmed microprocessor versus a dedicated microcontroller and creating an accurate metric to reflect their capability would enable the full analysis of the satellite's abilities. This was well outside the scope of this thesis, but was a significant factor in the duplication of components in the avionics sub-system.

The communication metrics did not consider anything other than baud rate for the comparison between components, where wavelength and the ability to transmit simultaneously are important considerations. Also ignored was the capability to deploy the satellite's antennas, or whether the transceivers had modems to properly modulate a signal for transmission. The modeling of a communications system and performance evaluation would ensure that a fully working communications sub-system would eventually be created.

The payload metrics require refinement, however their major issue was the payload accuracy dominating the actual payload's capability. Evaluating the accuracy against the desired payload and interlinking the two more effectively would allow the algorithm to select the correct payload and optimise payloads if they are selected within the population.

5.3 Improving customer requirements

The customer requirements utilised within this thesis were extremely limited. Each addition customer requirement for a design needs to be quantified and metrics made to match them. Examples of additional customer requirements that were considered included: orbital types, preferred size constraints and thermal behaviour. However, the more customer requirements that are able to be quantified and matched to a satellite, the more capable the algorithm will become.

5.3.1 Real world implementation

With the algorithm and methods described in this thesis, it requires a human designer to take the suggested design as a suggestion and convert it into a conceptual design. However, it is expected that with enough development of the model of a satellite, improvements in metrics and increased customer requirements that the algorithm could be utilised to create a complete satellite.

5.4 Summary

This thesis is investigates the possibility that computers could autonomously generate a CubeSat and provides an algorithm with the potential develop a working conceptual design. For creating a design, a design methodology was evaluated and utilised to provide a method to link the customer's requirements to the final design. The design methodology provides a consistent structure to evaluate the computers design process against and enables future improvements to the algorithm and metrics. The automation of component selection in conceptual design also potentially provides a substantial time saving in the front end of the design process.

By utilising a genetic algorithm, random sampling of a large search space is possible, and the overall population improves or maintains design choices with each generation. In short, this thesis answers two questions, the first is whether it is possible to autonomously link design levels and the second is whether or not a computer can create a conceptual design at the system level. This thesis indicates that both are possible and that future development of models and metrics will improve performance in the design process. This potentially leads to time savings in engineering design, allowing a designer to reduce the time spent in the conceptual design stage.

Bibliography

[1] Richard Dawkins. *The blind watchmaker: Why the evidence of evolution reveals a universe without design.* WW Norton & Company, 1996.

[2] Gul E Okudan and Shafin Tauhid. Concept selection methods–a literature review from 1980 to 2008. *International Journal of Design Engineering*, 1(3):243–277, 2008.

[3] Donald E. Grierson and Prabhat Hajela. *Emergent Computing Methods in Engineering Design*, volume 149 of *F: Computer and Systems Sciences*. Springer, 1994.

[4] Y.T. Chong, C.H. Chen, and K.F. Leong. A heuristic-based approach to conceptual design. *Research in Engineering Design*, 20(2):97–116, July 2009.

[5] Hong-Zhong Huang, Ruifeng Bo, and Wei Chen. An integrated computational intelligence approach to product concept generation and evaluation. *Mechanism and Machine Theory*, 41(5):567–583, 2006.

[6] Jason D Lohn, Derek S Linden, Gregory S Hornby, William F Kraus, and Adaan Rodriguez-Arroyo. Evolutionary design of an x-band antenna for nasa's space technology 5 mission. In *Evolvable Hardware, NASA/DoD Conference on*, pages 155–155. IEEE Computer Society, 2003.

[7] ArduSat. Ardusat - your arduino experiment in space, 2012. URL `https://www.kickstarter.com/projects/575960623/ardusat-your-arduino-experiment-in-space/description`.

[8] CubeSat. Mission statement, 2015. URL `http://www.cubesat.org/index.php/about-us/mission-statement`.

[9] Arash Mehrparvar et al. Cubesat design specification rev. 13. *The CubeSat Program, California Polytechnic State University*, 1, 2009.

[10] Elizabeth Buchen and Dominic DePasquale. 2014 nano / microsatellite market assessment, 2014. URL `http://www.sei.aero/eng/papers/uploads/archive/SpaceWorks_Nano_Microsatellite_Market_Assessment_January_2014.pdf`.

[11] E Gill, P Sundaramoorthy, J Bouwmeester, B Zandbergen, and R Reinhard. Formation flying within a constellation of nano-satellites: The qb50 mission. *Acta Astronautica*, 82(1):110–117, 2013.

[12] John S. Gero. Computational models of innovative and creative design processes. *Technological Forecasting and Social Change*, 64:183–196, 2000.

[13] Donald E. Grierson. Conceptual design using emergent computing techniques. In *Emergent Computing Methods in Engineering Design*, volume 149 of *Computer and System Sciences*, pages 150–161. NATO Advanced Research Workshop, Springer, August 1994.

[14] Miguel Martínez-Iranzo, Juan M Herrero, Javier Sanchis, Xavier Blasco, and Sergio García-Nieto. Applied pareto multi-objective optimization by stochastic solvers. *Engineering applications of artificial intelligence*, 22(3):455–465, 2009.

[15] John S. Gero and Udo Kannengiesser. A function-behavior-structure ontology of processes. *AI EDAM: Artificial Intelligence for Engineering Design, Analysis, and Manufacturing*, 21:379–391, 11 2007. ISSN 1469-1760. doi: 10.1017/S0890060407000340. URL `http://journals.cambridge.org/article_S0890060407000340`.

[16] John S. Gero and Udo Kannengiesser. The function-behaviour-structure ontology of design. In A. Chakrabarti and L.T.M Blessing, editors, *An Anthology of Theories and Models of Design*, pages 263–283. Springer, 2014.

[17] Johan De Kleer and John Seely Brown. A qualitative physics based on confluences. *Artificial intelligence*, 24(1):7–83, 1984.

[18] Norbert FM Roozenburg and Johannes Eekels. *Product design: fundamentals and methods*, volume 2. Wiley Chichester, 1995.

[19] WM Jenkins. A genetic algorithm for structural design optimization. *Proc. NATO Advanced Science Institutes. Series F: Computer and Systems Sciences. Emergent Computing Methods in Engineering Design: Applications of Genetic Algorithms and Neural Networks*, 149:30–52, 1996.

[20] Jeng-Jong Lin. Constructing an intelligent conceptual design system using genetic algorithm. *Journal of materials processing technology*, 140(1):95–99, 2003.

[21] Patrick Ngatchou, Anahita Zarei, and MA El-Sharkawi. Pareto multi objective optimization. In *Intelligent Systems Application to Power Systems, 2005. Proceedings of the 13th International Conference on*, pages 84–91. IEEE, 2005.

[22] Masatoshi Sakawa and Hitoshi Yano. Feasibility and pareto optimality for multi-objective nonlinear programming problems with fuzzy parameters. *Fuzzy Sets and Systems*, 43(1):1–15, 1991.

[23] Hao Jiang, Jing Chen, Tundong Liu, and Hongyan Fu. Design of an fbg sensor network based on pareto multi-objective optimization. *Photonics Technology Letters, IEEE*, 25(15):1450–1453, 2013.

[24] Jeffrey Horn, Nicholas Nafpliotis, and David E Goldberg. A niched pareto genetic algorithm for multiobjective optimization. In *Evolutionary Computation, 1994.*

IEEE World Congress on Computational Intelligence., Proceedings of the First IEEE Conference on, pages 82–87. Ieee, 1994.

[25] Jon T Richardson, Mark R Palmer, Gunar E Liepins, and Mike Hilliard. Some guidelines for genetic algorithms with penalty functions. In *Proceedings of the third international conference on Genetic algorithms*, pages 191–197. Morgan Kaufmann Publishers Inc., 1989.

[26] J David Schaffer. Multiple objective optimization with vector evaluated genetic algorithms. In *Proceedings of the 1st International Conference on Genetic Algorithms, Pittsburgh, PA, USA, July 1985*, pages 93–100, 1985.

[27] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *Evolutionary Computation, IEEE Transactions on*, 6(2):182–197, 2002.

[28] George F Hepner, Thomas Logan, Niles Ritter, and Nevin Bryant. Artificial neural network classification using a minimal training set-comparison to conventional supervised classification. *Photogrammetric Engineering and Remote Sensing*, 1990.

[29] Emina Alickovic and Abdulhamit Subasi. Usage of simplified fuzzy artmap for improvement of classification performances. *SouthEast Europe Journal of Soft Computing*, 2(2), 2013.

[30] Li Yu and Liya Wang. A fuzzy intelligent design retrieving system for customer requirements. *International Journal of Computer Applications in Technology*, 33 (2):247–254, 2008.

[31] Gail A Carpenter, Stephen Grossberg, Natalya Markuzon, John H Reynolds, and David B Rosen. Fuzzy artmap: A neural network architecture for incremental supervised learning of analog multidimensional maps. *Neural Networks, IEEE Transactions on*, 3(5):698–713, 1992.

[32] Gregory S Parnell, Patrick J Driscoll, and Dale L Henderson. *Decision making in systems engineering and management*, volume 81, chapter Decision Making, pages 395–446. John Wiley & Sons, 2011.

[33] Herbert J Kramer and Arthur P Cracknell. An overview of small satellites in remote sensing*. *International journal of remote Sensing*, 29(15):4285–4337, 2008.

[34] Jason Held. Dragen. Technical report, Saber Astronautics, 2011. URL `http://saberastro.com/home/company/ProductBriefDragEN2011.pdf`.

[35] Clyde Space. Clyde space, 2015. URL `http://www.clyde-space.com/cubesat_shop`.

[36] X-Y Shao, Z-H Wang, P-G Li, and C-X J Feng. Integrating data mining and rough set for customer group-based discovery of product configuration rules. *International Journal of Production Research*, 44(14):2789–2811, 2006.

[37] D. Kataria and A. Smith. Qb50 science units. In *3rd QB50 Workshop, VKI*, Brussels, Feb 2012.

[38] GomSpace. Gomspace, 2015. URL `http://www.gomspace.dk`.

[39] Adam C Watts, Vincent G Ambrosia, and Everett A Hinkley. Unmanned aircraft systems in remote sensing and scientific research: Classification and considerations of use. *Remote Sensing*, 4(6):1671–1692, 2012.

[40] Zhongping Lee, Chuanmin Hu, Shaoling Shang, Keping Du, Marlon Lewis, Robert Arnone, and Robert Brewin. Penetration of uv-visible solar radiation in the global oceans: Insights from ocean color remote sensing. *Journal of Geophysical Research: Oceans*, 118(9):4241–4255, 2013.

[41] William Gareth Rees. *Physical principles of remote sensing*. Cambridge University Press, 2013.

[42] Lars Alminde, Karl Kaas, Morten Bisgaard, Johan Christiansen, and David Gerhardt. Gomx-1 flight experience and air traffic monitoring results. In *Small Satellite Conference*, 2014.

[43] Alan Smith. Qb50 sensor selection working group final report. Technical report, QB50, March 2012.

[44] Victor Blacus. Electromagnetic spectrum, 2012. URL `http://en.wikipedia.org/wiki/File:Electromagnetic-Spectrum.svg`.

[45] Planet Labs. Flock 1. Website, 2015. URL `https://www.planet.com/flock1/`.

[46] Jonathan McDowell. Jonathan's space report, no. 697. *Harvard-Smithsonian Center for Astrophysics*, 2014.

[47] Jason M. Held. *The Modelling of Systems of Systems*. PhD thesis, University of Sydney, School of Aerospace, Mechanical and Mechatronic Engineering, Sydney, Australia, February 2008.

[48] Jinkyu Lee, Eugene Kim, and Kang G Shin. Design and management of satellite power systems. In *Real-Time Systems Symposium (RTSS), 2013 IEEE 34th*, pages 97–106. IEEE, 2013.

[49] Charles D Brown. *Elements of spacecraft design*. Aiaa, 2002.

[50] Gregory S Hornby, Al Globus, Derek S Linden, and Jason D Lohn. Automated antenna design with evolutionary algorithms. In *AIAA Space*, pages 19–21, 2006.

[51] Gunter Krebs. Gomx1 - gatoss, 2012. URL `http://space.skyrocket.de/doc_sdat/gomx-1.htm`.

[52] Dhiren Kataria, Craig Leff, Alan Smith, Rahil Chaudery, Peter Coker, Anasuya Aruliah, Ruedeger Reinhard, and Cem Ozan Asma. Sensors and science. In *QB50 5th Workshop, VKI*, Jan 2013.

[53] ISIS. Cubesat shop, 2015. URL http://www.cubesatshop.com.

[54] Aalborg Universitys Studentsatellite. Aau cubesat, 2015. URL http://www.space.aau.dk/cubesat/.

[55] Lars Alminde, Morten Bisgaard, Dennis Vinther, Tor Viscor, and Kasper Z Østergard. The aaucubesat student satellite project: architectural overview and lessons learned. In *16th IFAC Symposium on Automatic Control in Aerospace,(Russia)*, 2004.

[56] Wood & Douglas. Spec sheet, 2015. URL http://ultra-woodanddouglas.com/wireless-telemetry-downloads/datasheet-archive.

[57] Steve Massey. Copper: Ir imaging and radiation studies, 2011. URL http://mstl.atl.calpoly.edu/~bklofas/Presentations/SummerWorkshop2011/Massey_COPPER.pdf.

Non-dominated Sorting Genetic Algorithm - II

NSGA-II algorithm as developed by Deb, et. al [27].

```
$R_t = P_t \cup Q_t$                                  combine parent and offspring population
$F =$ fast-non-dominated-sort$(R_t)$                  $F = (F_1, F_2, ..., )$, all nondominated fronts of $R_t$
$P_{t+1} = \emptyset$ and $i = 1$
until $|P_{t+1}| + |F_i| \leq N$                      until the parent population is filled
  crowding-dist-assignment$(F_i)$                     calculate crowding-distance in$F_i$
  $P_{t+1} = P_{t+1} \cup F_i$                        include $i$ the nondom front in the parent pop
  $i = i + 1$                                         check the next front for inclusion
Sort$(F_i, \prec_n)$                                  sort in descending order using $\prec_n$
$P_{t+1} = P_{t+1} \cup F_i[1 : (N - |P_{t+1}|)]$     choose the first$(N - |P_{t+1}|)$elements of$F_i$
$Q_{t+1} =$ make-new-pop$(P_{t+1})$                   use selection, crossover and mutation
                                                      to create a new population$Q_{t+1}$
$t = t + 1$                                           increment the generation counter
```

Dataset Sources

Table B.1: Sources of Data utilised in creation of datasets

Company/Uni	**Category**	**Notes**
CubeSat Shop [53]	Power Systems	This section contains the datasheets for the Nanopower range of power supplies and battery packs.
	Communication Systems	The Nanocom, ISIS and S-band transceiver datasheets were retrieved from here.
	CubeSat Structures	The datasheets for all the ISIS brand structures were retrieved from this section of the website.
	Solar Panels	The ISIS and NanoPower solar panels datasheets were retrieved from this section.
	Attitude Control Systems	The MAI, ISIS, CubeSense and CubeSat ADCS and sub-system components datasheets were retrieved from this part of the website.
	Antenna Systems	The ISIS deployable antenna and the S-band patch antenna datasheets were retrieved from this part of the website.
	Command & Data Handling	The NanoMind, ISIS, Cube, Q-series and Andrews Model onboard computer datasheets were recovered from this section.
	Propulsion & Pressurisation	The Nanosatellite subsystem data was retrieved from here.
	Cameras & Payloads	The NanoCam and Argus Spectrometer datasheets were located in this section of the website

Table B.2: Sources of Data utilised in creation of datasets, continued

Company/Uni	**Category**	**Notes**
GomSpace [38]	Computers	The A712D and NanoHub subsystems were retrieved from this section of the website.
	Spacelink	The ANT430 and U482C communication and antenna subsystem data was compiled from this part of the website.
	Platforms	The NanoEye satellite component arrangement and GOMX structure datasheets were retrieved from this section of the website.
AAU CubeSat [54]	Documentation	The components for each section were compared with off the shelf (OTS) components and where possible analogs were drawn. The payload had no analog with commerical OTS components, so was constructed from the given architectural overview document [55].
Clyde Space [35]	CubeSat Lab	The Clyde Space EPS, battery, magnetorquer and switching components datsheets were gathered from this part of the commercial website.
Wood & Douglas [56]	Spec Sheet	The SX450G transceiver datasheet was gathered from the website of the manufacturer.
FLIR [57]	PDF	The FLIR Tau infrared camera data specifications was retrieved from this website.
QB50 [37, 43, 52]	PDF	The details for the INMS and Magenetic Flux equipment were retrieved from here. Along with the QB50 project customer requirements

Customer Requirements Training Data

Table C.1: Customer Requirements Natural Language Values

Satellite	Downlink	Uplink	Attitude	Wavelength	Detail
NanoEye	Average	Average	Lenient	Visual	Detailed
AAU Cube	Average	Fast	V. Lenient	Visual	Average
COMPASS-1	Average	V. Slow	Lenient	Visual	Not Detailed
GOMX-1	Average	Ex. Fast	Lenient	Radio	Detailed
ARCTIC-1	Average	V. Slow	Average	Thermal IR	Average
COPPER	Average	Fast	Average	Thermal IR	Detailed

Table C.2: Customer Requirements Finalised Fuzzy Logic Values

Satellite	Downlink	Uplink	Attitude	Wavelength	Detail
NanoEye	0.4999	0.4998	0.3329	0.5001	0.6672
AAU Cube	0.4999	0.6998	0.1667	0.5001	0.5002
COMPASS-1	0.4999	0.0998	0.3329	0.5001	0.3331
GOMX-1	0.4999	0.9998	0.3329	1.0000	0.6672
ARCTIC-1	0.4999	0.0998	0.5	0.834	0.5002
COPPER	0.4999	0.6998	0.5	0.834	0.6672

Experimental customer requirements

Table D.1: Total fuzzy distance from training data satellites

Satellite CRs	Downlink	Uplink	Attitude	Wavelength	Detail
NanoEye +1	Fast	Fast	Average	Near IR	V. Det.
NanoEye -1	Slow	Slow	V. Lenient	Ultraviolet	Average
AAU Cube +1	Fast	E. Fast	Lenient	Near IR	Det.
AAU Cube -1	Slow	Average	E. Lenient	Ultraviolet	Vague
COMPASS-1 +1	Fast	Slow	Average	Near IR	Average
COMPASS-1 -1	Slow	E. Slow	V. Lenient	Ultraviolet	V. Vague
GOMX-1 +1	Fast	E. Fast	Average	Radio	V. Det.
GOMX-1 -1	Slow	V. Fast	V. Lenient	Thermal IR	Average
ARCTIC-1 +1	Fast	Slow	Precise	Radio	Det.
ARCTIC-1 -1	Slow	E. Slow	Lenient	Near IR	Vague
COPPER +1	Fast	V. Fast	Precise	Radio	V. Det.
COPPER -1	Slow	Average	Lenient	Near IR	Average

Table D.2: Customer Requirements Natural Language Experiment 1

Satellite	Downlink	Uplink	Attitude	Wavelength	Detail
NanoEye	Average	Average	Lenient	Visual	Detailed
Test 1	Average	Average	Lenient	Radio	Detailed
Test 2	Average	Average	Lenient	Thermal Infrared	Detailed
Test 3	Average	Average	Lenient	Near Infrared	Detailed
Test 4	Average	Average	Lenient	Ultraviolet	Detailed
Test 5	Average	Average	Lenient	Mag Flux	Detailed
Test 6	Average	Average	Lenient	INMS	Detailed
AAU Cube	Average	Fast	V. Lenient	Visual	Average
Test 7	Average	Fast	V. Lenient	Radio	Average
Test 8	Average	Fast	V. Lenient	Thermal Infrared	Average
Test 9	Average	Fast	V. Lenient	Near Infrared	Average
Test 10	Average	Fast	V. Lenient	Ultraviolet	Average
Test 11	Average	Fast	V. Lenient	Mag Flux	Average
Test 12	Average	Fast	V. Lenient	INMS	Average
COMPASS-1	Average	V. Slow	Lenient	Visual	Not Det.
Test 13	Average	V. Slow	Lenient	Radio	Not Det.
Test 14	Average	V. Slow	Lenient	Thermal Infrared	Not Det.
Test 15	Average	V. Slow	Lenient	Near Infrared	Not Det.
Test 16	Average	V. Slow	Lenient	Ultraviolet	Not Det.
Test 17	Average	V. Slow	Lenient	Mag Flux	Not Det.
Test 18	Average	V. Slow	Lenient	INMS	Not Det.
GOMX-1	Average	Ex. Fast	Lenient	Radio	Detailed
Test 19	Average	Ex. Fast	Lenient	Thermal Infrared	Detailed
Test 21	Average	Ex. Fast	Lenient	Near Infrared	Detailed
Test 22	Average	Ex. Fast	Lenient	Visual	Detailed
Test 23	Average	Ex. Fast	Lenient	Ultraviolet	Detailed
Test 24	Average	Ex. Fast	Lenient	Mag Flux	Detailed
Test 25	Average	Ex. Fast	Lenient	INMS	Detailed
ARCTIC-1	Average	V. Slow	Average	Thermal IR	Average
Test 26	Average	V. Slow	Average	Radio	Average
Test 27	Average	V. Slow	Average	Near Infrared	Average
Test 28	Average	V. Slow	Average	Visual	Average
Test 29	Average	V. Slow	Average	Ultraviolet IR	Average
Test 30	Average	V. Slow	Average	Mag Flux	Average
Test 31	Average	V. Slow	Average	INMS	Average
COPPER	Average	Fast	Average	Thermal IR	Detailed
Test 32	Average	Fast	Average	Radio	Detailed
Test 33	Average	Fast	Average	Near Infrared	Detailed
Test 34	Average	Fast	Average	Visual	Detailed
Test 35	Average	Fast	Average	Ultraviolet	Detailed
Test 36	Average	Fast	Average	Mag Flux	Detailed
Test 37	Average	Fast	Average	INMS	Detailed

Experimental permutation distances

Table E.1: Fuzzy ARTMAP Classification Outputs, Experiment 1

NanoEye Perms.	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6
Class. Category	1	1	1	1	1	1
AAUSat Perms.	Test 7	Test 8	Test 9	Test 10	Test 11	Test 12
Class. Category	2	2	2	2	2	2
COMPASS Perms.	Test 13	Test 14	Test 15	Test 16	Test 17	Test 18
Class. Category	3	3	3	3	3	3
GOMX-1 Perms.	Test 19	Test 20	Test 21	Test 22	Test 23	Test 24
Class. Category	4	4	1	1	1	1
ARCTIC-1 Perms.	Test 25	Test 26	Test 27	Test 28	Test 29	Test 30
Class. Category	5	5	5	5	5	5
COPPER Perms.	Test 31	Test 32	Test 33	Test 34	Test 35	Test 36
Class. Category	6	6	6	6	6	6

Table E.2: Fuzzy ARTMAP Classification Outputs, Experiment 2

Original Satellite	1	2	3	4	5	6
Classification Category	6	2	3	6	5	6

Table E.3: Fuzzy ARTMAP Classification Outputs, Experiment 3

Original Satellite	1	1	2	2	3	3
Classification Category	2	6	2	4	3	5
Original Satellite	4	4	5	5	6	6
Classification Category	2	4	3	5	1	4

Table E.4: Total fuzzy distance between training data and test data, experiment 1

	GOMX	**NanoEye**	**AAU**	**COMP.**	**ARC.**	**COP.**
Sat 1 Radio	0.500	1.033	1.234	0.500	0.900	0.533
Sat 1 Thermal	0.334	0.867	1.068	0.666	0.734	0.367
Sat 1 IR	0.167	0.700	0.901	0.833	0.901	0.534
Sat 1 UV	0.167	0.700	0.901	1.167	1.235	0.868
Sat 1 Flux	0.333	0.867	1.067	1.333	1.401	1.034
Sat 1 Ions	0.500	1.033	1.234	1.500	1.568	1.201
Sat 2 Radio	1.033	0.500	1.433	0.633	1.099	0.666
Sat 2 Thermal	0.867	0.334	1.267	0.799	0.933	0.500
Sat 2 IR	0.700	0.167	1.100	0.966	1.100	0.667
Sat 2 UV	0.700	0.167	1.100	1.300	1.434	1.001
Sat 2 Flux	0.867	0.333	1.267	1.466	1.601	1.168
Sat 2 Ions	1.033	0.500	1.433	1.633	1.767	1.334
Sat 3 Radio	1.234	1.433	0.500	1.234	0.500	1.267
Sat 3 Thermal	1.068	1.267	0.334	1.400	0.334	1.101
Sat 3 IR	0.901	1.100	0.167	1.567	0.501	1.268
Sat 3 UV	0.901	1.100	0.167	1.901	0.835	1.602
Sat 3 Flux	1.067	1.267	0.333	2.067	1.001	1.768
Sat 3 Ions	1.234	1.433	0.500	2.234	1.168	1.935
Sat 4 Thermal	0.834	0.967	1.568	0.166	1.234	0.467
Sat 4 IR	0.667	0.800	1.401	0.333	1.401	0.634
Sat 4 Visual	0.500	0.633	1.234	0.500	1.568	0.801
Sat 4 UV	0.667	0.800	1.401	0.667	1.735	0.968
Sat 4 Flux	0.833	0.967	1.567	0.833	1.901	1.134
Sat 4 Ions	1.000	1.133	1.734	1.000	2.068	1.301
Sat 5 Radio	1.234	1.433	0.834	1.234	0.166	0.933
Sat 5 IR	0.901	1.100	0.501	1.567	0.167	0.934
Sat 5 Visual	0.734	0.933	0.334	1.734	0.334	1.101
Sat 5 UV	0.901	1.100	0.501	1.901	0.501	1.268
Sat 5 Flux	1.067	1.267	0.668	2.067	0.667	1.434
Sat 5 Ions	1.234	1.433	0.834	2.234	0.834	1.601
Sat 6 Radio	0.867	1.000	1.601	0.467	0.933	0.166
Sat 6 IR	0.534	0.667	1.268	0.800	0.934	0.167
Sat 6 Visual	0.367	0.500	1.101	0.967	1.101	0.334
Sat 6 UV	0.534	0.667	1.268	1.134	1.268	0.501
Sat 6 Flux	0.700	0.834	1.435	1.300	1.434	0.667
Sat 6 Ions	0.867	1.000	1.601	1.467	1.601	0.834

Table E.5: Total fuzzy distance between training data and test data, experiment 2

	GOMX	**NanoEye**	**AAU**	**COMP.**	**ARC.**	**COP.**
Test 1	0.367	0.500	1.101	0.967	1.101	0.334
Test 2	0.534	0.333	0.934	0.800	0.934	0.501
Test 3	1.001	1.200	0.267	2.001	0.935	1.702
Test 4	0.367	0.500	1.101	0.633	1.101	0.334
Test 5	0.701	0.900	0.701	1.367	0.367	0.734
Test 6	0.668	1.201	1.402	1.000	0.734	0.367

Table E.6: Total fuzzy distance between training data and test data, experiment 3

	GOMX	**NanoEye**	**AAU**	**COMP.**	**ARC.**	**COP.**
Sat 1 -1 Tier	0.900	0.767	0.900	1.900	1.234	1.601
Sat 1 +1 Tier	0.901	1.034	1.635	1.167	1.301	0.534
Sat 2 -1 Tier	1.034	0.901	1.100	2.034	1.768	1.735
Sat 2 +1 Tier	0.867	1.000	1.601	0.533	1.601	0.834
Sat 3 -1 Tier	1.533	1.400	0.799	2.533	1.467	2.234
Sat 3 +1 Tier	0.901	1.100	0.901	1.567	0.567	0.934
Sat 4 -1 Tier	1.267	0.734	1.667	0.799	1.333	0.900
Sat 4 +1 Tier	1.534	1.667	2.268	0.534	1.600	0.833
Sat 5 -1 Tier	1.201	1.400	0.467	1.867	0.801	1.568
Sat 5 +1 Tier	1.234	1.767	1.568	1.234	0.900	0.933
Sat 6 -1 Tier	0.534	0.733	0.934	1.200	0.934	0.901
Sat 6 +1 Tier	1.601	1.734	2.335	0.801	1.667	0.900

Satellite population metric fitness

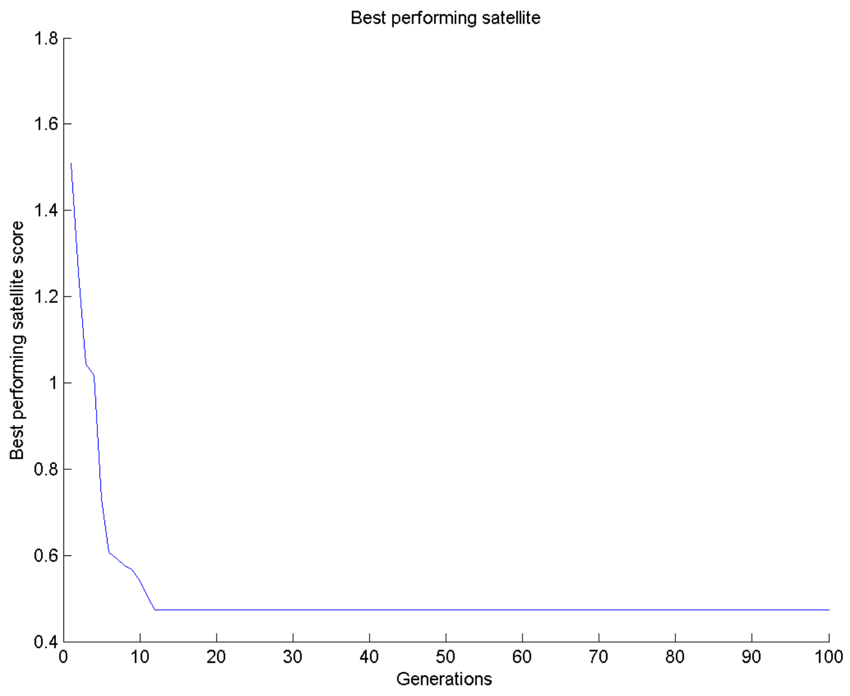


Figure F.1: Best performing individual satellite for permutation A2

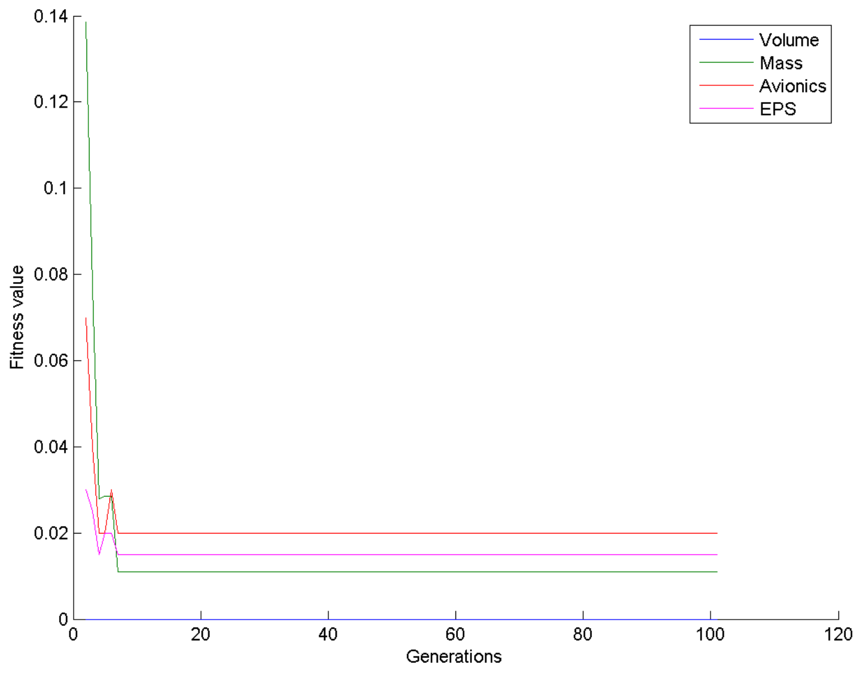


Figure F.2: Essential metric performance for permutation A2