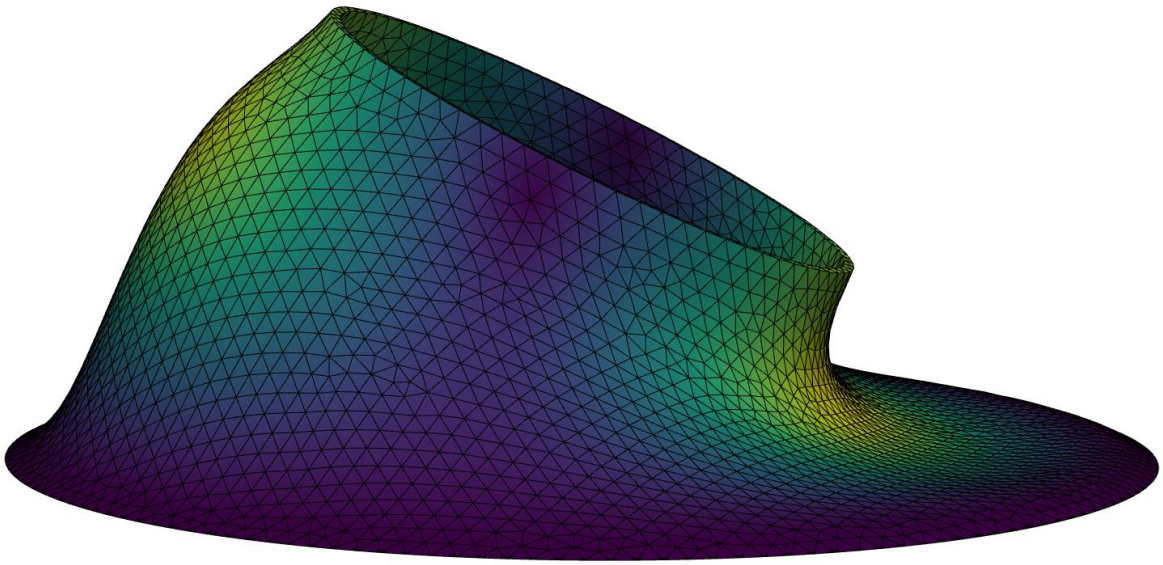


Concept exploration of spacecraft separation structures using evolutionary optimization

Master's thesis in Complex Adaptive Systems

ERIK ARNEBRO SÖDERBERG

DEPARTMENT OF MECHANICS AND
MARITIME SCIENCES

CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2022
www.chalmers.se

MASTER'S THESIS IN COMPLEX ADAPTIVE SYSTEMS

Concept exploration of spacecraft separation structures using evolutionary optimization

ERIK ARNEBRO SÖDERBERG

Department of Mechanics and Maritime Sciences
Applied Artificial Intelligence
CHALMERS UNIVERSITY OF TECHNOLOGY

Göteborg, Sweden 2022

Concept exploration of spacecraft separation structures using evolutionary optimization
ERIK ARNEBRO SÖDERBERG

© ERIK ARNEBRO SÖDERBERG, 2022

Master's thesis 2022:30
Department of Mechanics and Maritime Sciences
Applied Artificial Intelligence
Chalmers University of Technology
SE-412 96 Göteborg
Sweden
Telephone: +46 (0)31-772 1000

Cover:
First modal shape of a launch vehicle adapter. Displacement exaggerated by a factor of 150. This structure is one of the Pareto optimal solutions found using the method developed for this thesis.

Chalmers Reproservice
Göteborg, Sweden 2022

Concept exploration of spacecraft separation structures using evolutionary optimization
Master's thesis in Complex Adaptive Systems
ERIK ARNEBRO SÖDERBERG
Department of Mechanics and Maritime Sciences
Applied Artificial Intelligence
Chalmers University of Technology

Abstract

Conventional concept exploration tends to be narrowly focused around previous solutions, especially in an industry with as much emphasis on reliability as aerospace engineering. A drawback of this approach is the likelihood of high performing design concepts existing outside that narrow design domain. In this thesis project a tool is developed which aims to augment the concept exploration phase by autonomously searching a broad range of feasible solutions. The search converges to the non-dominated set by means of multi-objective evolutionary optimization, wherein the structures are optimized for lower mass and higher natural frequency. The candidate solutions are modelled, meshed, and structurally analysed using open-source software. The tool was tested on two different types of structures, namely *launch vehicle adapters* (LVA) and *satellite dispenser structures*. LVA design solutions with both higher natural frequency (up to 4% increase) and reduced mass were discovered. The results show that the developed method is able to converge to a set of high performing solutions, and present these to the user as input for subsequent design decisions.

Keywords: Multi-objective evolutionary optimization; automated finite element analysis; aerospace engineering

Acknowledgements

First and foremost I would like to thank my supervisor Michael Thuswaldner for his enthusiasm and encouragement. By hiring me as a summer intern in 2021 he opened the door for me to the fascinating world of space engineering which I had admired from the outside for so long. Following that, he enabled this thesis project and dedicated precious time and effort to support it from start to finish. As a result the trajectory of my career, and thus also my life, has changed for the better. In addition, I would like to thank the systems engineering team at Beyond Gravity for not only being exceptionally competent, but also friendly and welcoming. Secondly I would like to thank my examiner Mattias Wahde for showing great interest in the work and providing excellent guidance. Having someone with as much expertise in the field involved throughout the project made me confident at each step, and allowed me to move forward more effectively. Not least, I thank Mattias for teaching the course on stochastic optimization which introduced me to the subject matter at a university level. Without that course I doubt this thesis project would exist. Finally, I would like to thank my very close group of friends whom I have shared my time at Chalmers with.

Contents

Abstract i

Acknowledgements ii

Contents iii

1 Introduction 1
1.1 Background . . . 1
1.2 Problem . . . 1
1.3 Purpose . . . 1
1.4 Goals . . . 2
1.5 Limitations . . . 2

2 Theory 3
2.1 Launch vehicle structures . . . 3
2.1.1 Launch vehicle adapter . . . 3
2.1.2 Dispenser structure . . . 3
2.1.3 Fairing . . . 4
2.2 Evolutionary computation . . . 5
2.2.1 Genetic algorithms . . . 5
2.2.2 Interactive evolutionary computation . . . 6
2.2.3 Novelty search . . . 6
2.3 Multi-objective optimization . . . 7
2.3.1 Multi-objective evolutionary algorithm . . . 7
2.4 Finite element method . . . 8
2.4.1 Natural frequency analysis . . . 8

3 Method 9
3.1 Project specific parameters . . . 10
3.1.1 Dispenser . . . 10
3.1.2 Launch vehicle adapter . . . 11
3.2 Decoding . . . 11
3.2.1 Dispenser . . . 13
3.2.2 Launch vehicle adapter . . . 14
3.3 CAD modelling . . . 15
3.3.1 Dispenser . . . 15
3.3.2 Launch vehicle adapter . . . 16
3.4 Meshing . . . 16
3.4.1 Dispenser . . . 17
3.4.2 Launch vehicle adapter . . . 17
3.5 Finite element analysis . . . 17
3.6 Optimization . . . 18
3.7 User interface and interaction . . . 19
3.8 Parallel computation . . . 19

4 Results 20
4.1 Dispenser . . . 20
4.2 Launch vehicle adapter . . . 23

5 Discussion and conclusion **26**
5.1 Structure generation 26
5.2 Optimization 27
5.2.1 Spurious solutions 27
5.3 User interface 27
5.4 Concept exploration 28
5.5 Model discrepancies 28
5.6 Conclusions 29

References **30**

1 Introduction

Concept exploration refers to the process of searching a space of feasible solutions for new and useful design ideas. A phase of concept exploration is inherent to any engineering project which poses new questions or challenges. When performed by humans this process is limited in its scope, and for good reason. Building from previous solutions when faced with a new problem, as opposed to starting with a blank slate, is a sound engineering practise as it is both time and resource effective. Concept exploration therefore tends to originate from previous solutions. A thorough exploration of the space of all feasible solutions, if even possible, would be very expensive to perform manually. A tool which could search that space autonomously, or even intelligently, would therefore be of benefit.

1.1 Background

Beyond Gravity (previously RUAG Space) in Linköping designs and manufactures mechanical systems for spacecraft separation. These systems are generally composed of a structure, a separation mechanism, and accompanying electronics. The structures have to be as light as possible, as is with all hardware launched into space. The less mass spent on the structure of the vehicle, the more is left for the payload (generally satellites). Additionally, the structures need to be very stiff to endure the high loads and to avoid resonances. In addition to these inherently tough requirements the space industry tends to err heavily on the side of caution. This means exaggerated margins for error and derating of components, resulting is an even narrower design domain.

While many other steps in the process of product development have incorporated modern computation effectively, the initial design phase still relies on human intuition. There are however computational methods for generating solutions to design problems, and there are even precedent examples in the space industry. The most notable case is the evolved antenna flown on NASA's ST5 spacecraft in 2006 [14]. The geometry of the antenna was discovered by a genetic algorithm optimizing for increased antenna gain.

Extracting the optimization metrics from every candidate solution is a prerequisite for the optimization itself, this process varies in difficulty depending on the nature of the metrics. In this thesis project mass and natural frequency are the core optimization metrics, meaning a structural model must be generated and a structural analysis must be carried out. Previous work that combines automated structural analysis and optimization exists [36, 5, 11], but no formalized or standardized method has been established. Other fields such as shape optimization [25, 34] as well as generative design [16] have also inspired this work.

1.2 Problem

The conventional method of modifying previous solutions to meet new demands has proven effective, but it limits the search for new solutions substantially. This incremental approach is especially prevalent in the space industry where there is a strong hesitancy towards anything that has not yet flown [10].

In practise the narrow design scope may lead to certain geometries being overlooked completely or discarded without full consideration. Since each project entails a unique design space some solutions which has previously been correctly discarded may prove useful under new circumstances. However, such a solution is less likely to be considered due to learned bias. For these reasons it has been suggested that less conservative methods should at least be experimented with.

It is useful to view the problem from a business perspective in which time and resources are scarce. When a company receives a new project offer a design proposal is to be generated and the client needs to be convinced of the quality and reliability of the design. This situation does not incentivize spending more time and resources exploring new concepts. The problem that this thesis addresses is that conventional concept exploration is too narrow with respect to the likelihood of undiscovered high performing solutions.

1.3 Purpose

The purpose is to investigate the feasibility of using evolutionary computation to augment the concept exploration phase. The aim is to give the engineers a practically inexhaustible source of input to use at their discretion, in the form of a broad set of high performing solutions which conform to the relevant requirements.

The suggested solutions could influence design decisions and give the engineers a better understanding of the design domain in which they operate. Note that the purpose is not to replace the conventional practise, but to provide an auxiliary tool.

1.4 Goals

The central goal is to show that evolutionary computation is a viable method for concept exploration, since achieving this goal would entail the fulfilment of the purpose as stated above. Listed below are additional, more specific goals which reflect the different aspects of the project. The first four goals on the list are relatively concrete and central to the project's success. The last three goals are more abstract and it will be difficult to determine whether they have been achieved or not.

1. Develop a tool which autonomously generates feasible solutions.
2. Implement a genetic algorithm which iteratively optimizes the generated solutions.
3. Expose the user to a broad set of feasible solutions.
4. Propose novel design concepts which outperform the conventional design.
5. Give the user a better understanding of the design space.
6. Give the user higher confidence in design decisions.
7. Contribute to introducing the space industry to machine learning ideas and methods.

1.5 Limitations

As the word *concept* indicates, the solutions are meant to be perceived at a high level of abstraction. In other words the generated structures are meant to spark ideas, not to be manufactured. Optimization is used as a tool as opposed to being a goal in and of itself. The structures are thus not guaranteed to be optimal. While the developed method is generalizable the implementation only considers two types of structures: dispenser structures and launch vehicle adapters (LVA), see Section 2.1. There are numerous simplifications made to the modelling of the structures in comparison to standard practises. These simplifications are necessary and justified for the following three reasons:

1. Structure generation and analysis needs to be automated which makes certain details challenging to implement.
2. This process is time consuming as it needs to be performed many times (thousands).
3. Standard modelling software is not suited for automated and integrated input-output. Open source software is therefore used instead which means that some specific feature (e.g. shell elements for FEM) are not available.

Genetic algorithms (GA) were the only class of optimization algorithms considered. There are other methods which could be applied successfully, for example particle swarm optimization of simulated annealing [36]. There is a further delimitation in that only one specific genetic algorithm, NSGA-II [7], is implemented. There are of course several other genetic algorithms which could be implemented with comparable results. The graphical user interface (GUI) used to visualize the progress and result of the GA is a basic implementation lacking many features which could be useful.

2 Theory

This chapter introduces the theoretical foundation and relevant background for the method developed for this thesis.

2.1 Launch vehicle structures

It is common to think of a launch vehicle (a rocket) as a single unified entity ready to take a payload into space. However, this will often be the wrong level of abstraction as a launch vehicle includes many subsystems and modules, some of which need to be modified to accommodate specific missions. For example, a single launch vehicle, such as Ariane 5, can be configured to launch one large payload, two payloads stacked on top of each other, or several smaller payloads.

The substructures of a launch vehicle which are relevant to this thesis are introduced in this section. The aim is to put these structures, and thus also the thesis project, into the context of space flight.

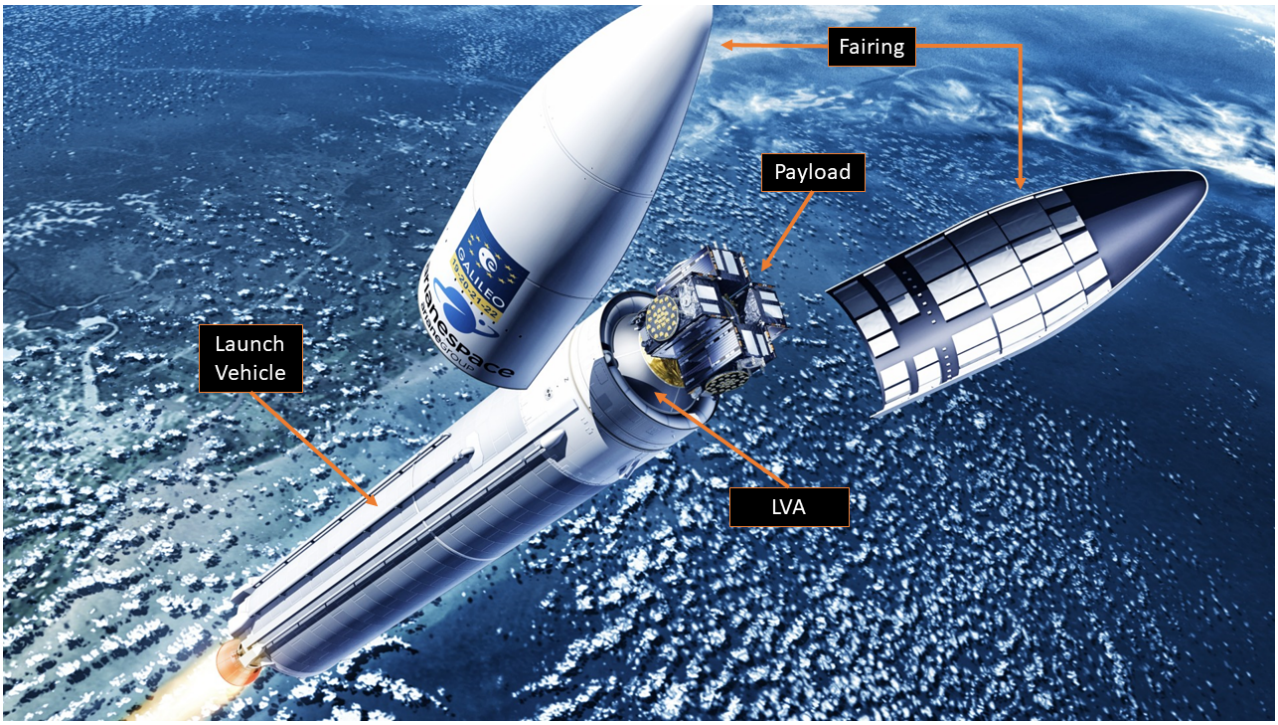


Figure 2.1: *Rendered visualization of Ariane 5 fairing separation. The components mentioned in this section are labelled. Source: https://www.beyondgravity.com/en/launchers, image reproduced with kind permission from Beyond Gravity.*

2.1.1 Launch vehicle adapter

Between the launch vehicle and the payload (e.g. a satellite) is an additional structure called the launch vehicle adapter (LVA), see Figure 2.1. The primary role of this structure is to connect the interface of the launch vehicle with that of the payload. Since these interfaces tend to have different dimensions the geometry of the LVA is typically a truncated cone. In addition to transitioning between cross-sections, the LVA is also responsible for the final separation of the payload.

2.1.2 Dispenser structure

A satellite dispenser is a system designed to carry several smaller payloads as opposed to one large payload. In recent years the demand for communication satellite constellations together with technological advances

allowing for smaller satellites (e.g. denser batteries, computers, and solar cells) has encouraged the development of methods for launching many satellites at once. One example of such a constellation project is the OneWeb constellation, for which Beyond Gravity developed and manufacture the dispenser system. Each launch carries up to 36 satellites meant to provide internet access in places which are otherwise hard to reach. The dispenser structure typically has a cylindrical geometry with satellites attached along its circumference at several tiers.

Between the launch vehicle and the dispenser structure is an LVA. However, the payload separation occurs at the level of the individual satellites, and therefore the LVA does not perform any separation in the case of a dispenser system.

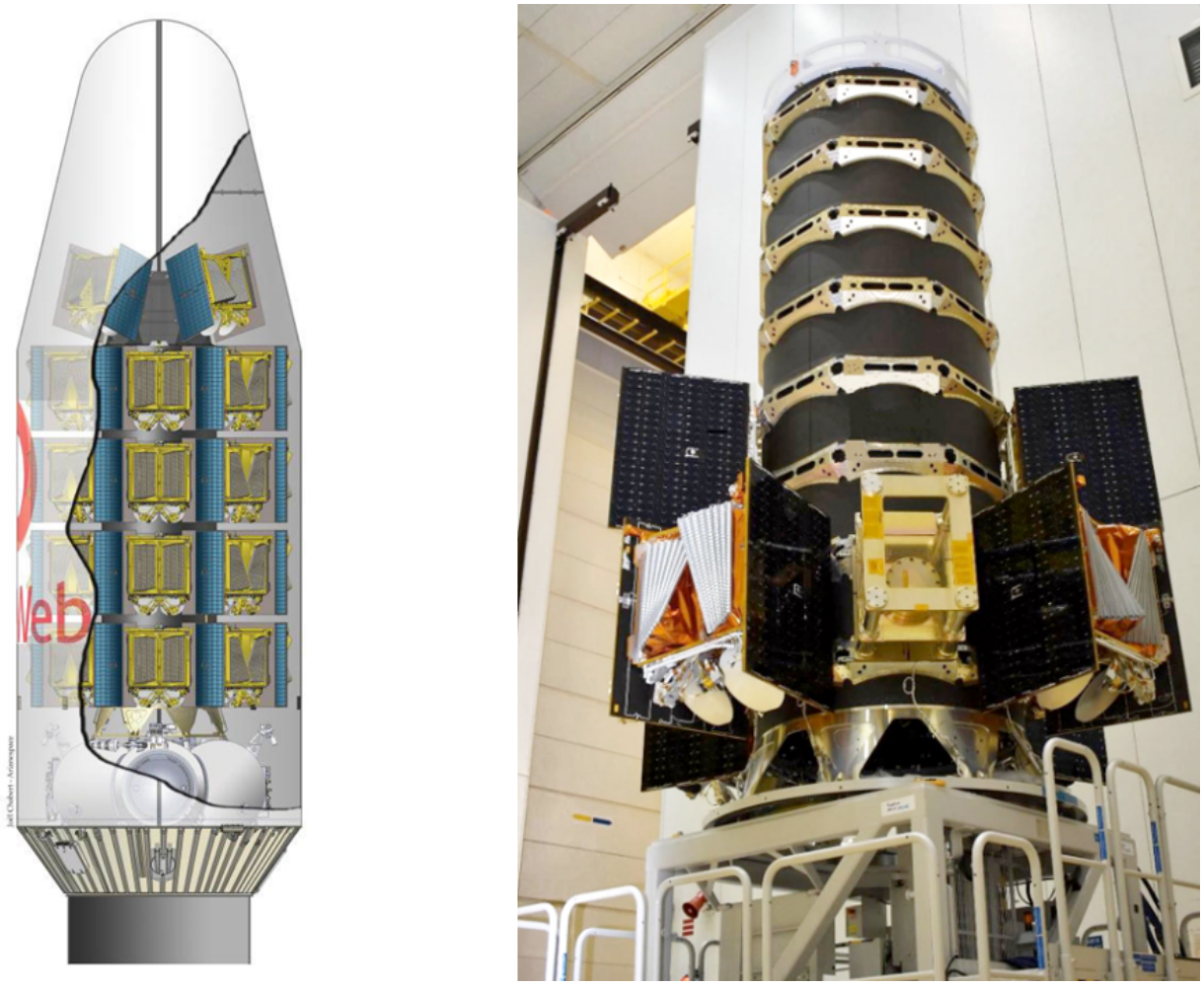


Figure 2.2: *OneWeb dispenser system which is developed and manufactured at BeyondGravity. It is able to carry 36 satellites to orbit per launch. Left: Visualization of the dispenser system with satellites attached inside its payload fairing [3], image reproduced with kind permission from Arianespace. Right: Partially assembled dispenser system included to show the underlying cylindrical structure (source: OneWeb Twitter), image reproduced with kind permission from OneWeb.*

2.1.3 Fairing

In order to protect the payload from pressure and heat during launch it is enclosed in a nose cone known as the payload fairing, see Figure 2.2 (left). Once the spacecraft has reached above the atmosphere the fairing is discarded and the payload is exposed, see Figure 2.1. The fairing is not an object of any optimization efforts in this project, but it does represent the physical boundary of the structures which are considered.

2.2 Evolutionary computation

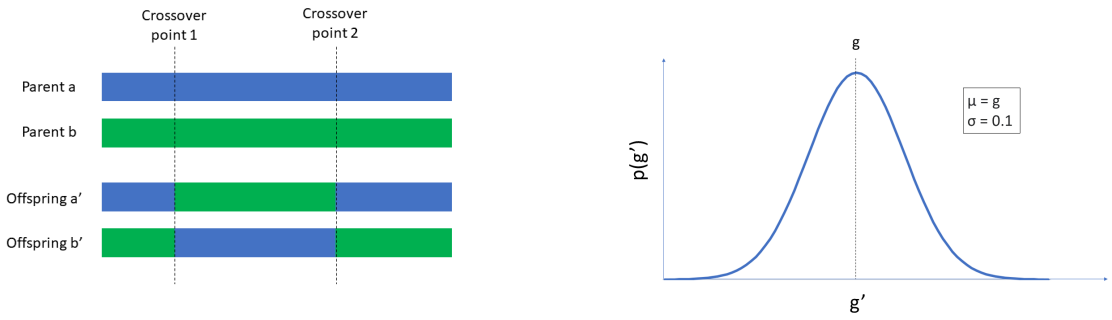
Evolutionary computation (EC) is a class of algorithms taking inspiration from biological evolution [13, 26]. The same governing principles that drive biological evolution can be applied in computer programs to similar effect. Both biological evolution and EC functions through the iterative adaptation of a population with regard to the applied selection pressures. The three necessary and sufficient components are: replication, variation, and differential success. If these components are present it is expected that evolution occurs [8]. While EC is considered a type of artificial intelligence it is perhaps best characterized as a stochastic optimization method. It should be noted that EC never guarantees optimality.

2.2.1 Genetic algorithms

A genetic algorithm (GA) is a type of evolutionary computation. GAs are iterative, population based algorithms wherein the members of the population (the candidate solutions) are called individuals. The individuals are represented by a set of genes forming a chromosome. A given chromosome with its particular settings constitute the genotype (the genetic information) of an individual, which is a representation of the phenotype (candidate solution to a problem). By encoding the phenotype in the representative genotype it is possible to apply evolutionary operators and modify the solution [9]. A standard genetic algorithm is presented in Algorithm 1.

While replication and variation (mutation) occur at the genotypic level, the differential success of the population is evaluated at the phenotypic level. This requires that the chromosome is decoded to form the corresponding individual (the phenotype). The individual is evaluated against a set of selection pressures, most commonly represented by a fitness function which is formulated by the programmer beforehand. Individuals are selected for reproduction based on their fitness such that more fit individuals have their genes overrepresented in the next generation. The genes of selected parents are combined in a process called crossover (or recombination) before random mutations occur. The process then repeats for a predetermined number of generations or until a stopping criterion is satisfied.

There are different methods for performing each evolutionary operator (i.e. crossover, mutation, and selection)[32]. A common crossover method is two-point crossover, wherein each chromosome (assuming two parents have been selected) is cut at two random points, splitting each chromosome into three parts. The middle parts are then swapped, producing two offspring as shown in Figure 2.3 (a). Crossover occurs with probability p_{cross}.



(a) (b)

Figure 2.3: *(a) Two point crossover, which is the method for recombination used in this project. (b) Gaussian creep mutation, as implemented for this project, where g is the value of the gene before mutation, g' (x axis) is the value of the gene after mutation, and $p(g')$ is the probability density. A standard deviation of $\sigma = 0.1$ is used.*

The choice of mutation method depends on the gene representation. If the genes are binary then mutation most commonly means flipping a bit. However, if the genes are represented by real numbers then a creep mutation is common, wherein a mutation results in the gene value being shifted randomly a small amount. Genes mutate with a certain probability p_{mut}, which is commonly chosen such that one gene per offspring is expected to mutate.

The two most common selection methods are tournament selection and roulette wheel selection. In

tournament selection a random subset of the population compete. The individual with highest fitness is chosen with a probability $p_{\text{tour}} > 0.5$. If that individual is not selected, it is removed from the tournament, whereupon the competition is repeated until either an individual has been selected or there remains only one individual, which is then selected. Roulette wheel selection selects individuals probabilistically in proportion to their relative fitness. Another common feature of GAs is elitism which ensures that one or more of the most fit individuals is preserved by placing them in the next generation directly, bypassing the selection step. This is done to avoid losing these high fitness solutions due to stochasticity.

Algorithm 1 Standard genetic algorithm

```
Initialize counter t ← 0
Initialize population P_t
Fitness ← Evaluate(P_t)
while t < n_generations do
    Parents ← Selection(P_t, Fitness)
    Offspring ← Recombination(Parents)
    P_{t+1} ← Offspring
    P_{t+1} ← Mutation(P_{t+1})
    Fitness ← Evaluate(P_{t+1})
    t ← t + 1
end while
return argmax(Fitness)
```

2.2.2 Interactive evolutionary computation

In interactive evolutionary computation (IEC) the evaluation of individuals is carried out at least in part by a human user as opposed to a fitness function. The central idea is that the fitness of a given solution can be too complex (or too subjective) to evaluate with numerical analysis alone, and human input can therefore be of benefit. In practise the human evaluation is done through a visual interface. Typically an array of candidate solutions is presented on a screen and the user assigns ratings according to some scheme.

A problem specific to IEC is user fatigue. Unlike computers, human evaluators generally become tired after a fairly small number of evaluations. It has been shown that the quality of the evaluation deteriorates after some time [27, 22]. In practise this limits the number of generations and the number of individuals in the population, which in turn requires faster convergence to a solution. Another problem with having a human involved is that the evaluation changes over time in contrast to a static fitness function. This means that fitness is only relative to the individuals in the same generation.

IEC algorithms have been shown to outperform conventional algorithms in some domains. Deceptive problems which tend to get algorithms trapped at local optima can often use human input effectively [33]. As mentioned above, IEC performs particularly well where the problem is very complex or subjective (e.g. art or music).

2.2.3 Novelty search

The emergence of this method stems from a critique of objective functions in general. Researchers have argued that the weakness of evolutionary algorithms compared to natural evolution suggests that something is missing [18]. With traditional fitness functions the algorithms strive in a certain direction for higher fitness values, but in nature there is no such goal. Nature finds niches by randomly exploring some configuration space. This is the idea with novelty search (NS), to swap out the fitness function with a novelty score which only cares about discovering new regions of the design space. The counter-intuitive result is that NS in many cases has better results reaching the objective despite not looking for it. An extension of NS algorithms is NS with local competition, mimicking the concept of niches from biological evolution [23, 17].

The biggest challenge when implementing NS is to define what is called the behavioural space, that is, the space in which novelty is quantified based on distance to other solutions. One must determine what characteristics define the behaviour of an individual.

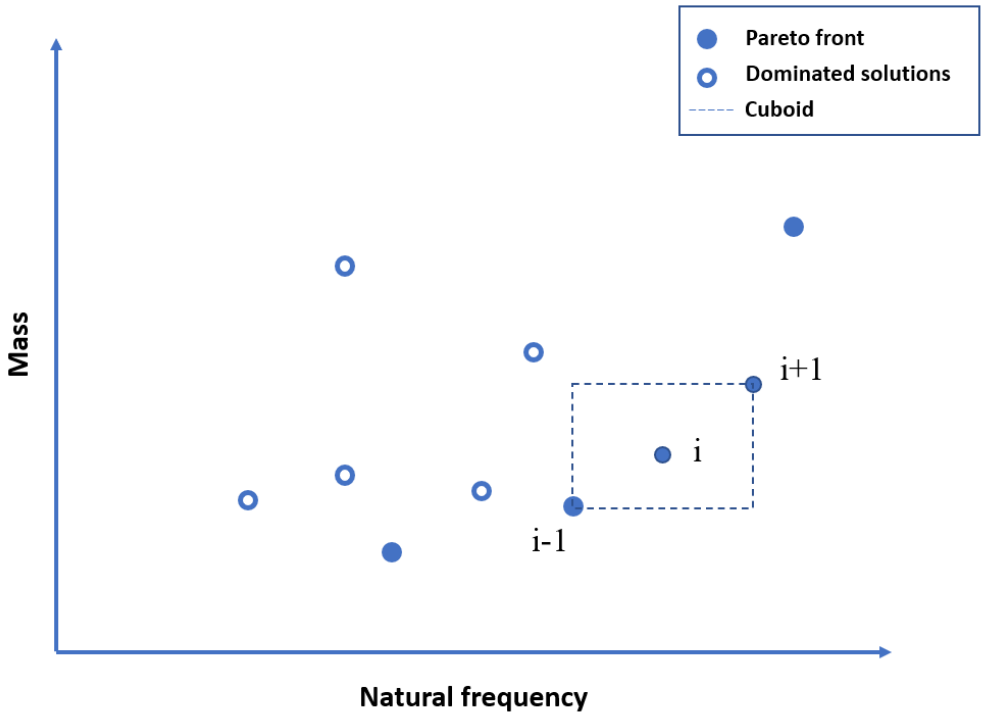


Figure 2.4: *Visualization of a Pareto front in two dimensions, inspired by Figure 1. from the original NSGA-II paper [7]. The filled dots are non-dominated and constitute the Pareto front. The cuboid is used to calculate crowding distance, which is a sparsity metric used to break ties among solutions that belong to the same front.*

2.3 Multi-objective optimization

When optimizing for a single objective it is self-evident which solution is most optimal. However, that is no longer the case when optimizing for two or more objectives. A solution is considered *non-dominated* if and only if no other solution performs better (or equally well) at *all* objectives [29], which in the context of multiple optimization objectives often results in several solutions being mutually non-dominated. Conversely, the solutions for which there exists another solution that performs better at all objectives are considered dominated. The output of multi-objective optimization is thus, rather than a single solution, the set of all non-dominated solutions. This set approximates a hypersurface known as the *Pareto front* [30]. To further evaluate the dominated solutions one can denote the true Pareto front as the *first* front, and define the *second* front as the set of solutions dominated only by solutions in the first front, and so on.

This approach to multiple optimization objectives, known as Pareto optimization, is very general as it makes no assumptions about the weighting of different objectives. One *could* define a weighting scheme beforehand, but this would in essence reduce a multi-objective problem to a single-objective problem. In practical applications wherein a single solution is to be selected from the non-dominated set a weighting of the objectives occurs implicitly when that choice is made.

2.3.1 Multi-objective evolutionary algorithm

Evolutionary algorithms can be extended to optimize for multiple objectives, and there exists a large literature of efficient methods [31, 19, 35]. One popular algorithm is NSGA-II [7] which utilizes Pareto fronts for evaluation and selection. In NSGA-II each individual is assigned a rank corresponding to the Pareto front to which the individual belongs. Rank is used in place of fitness and determines the individuals likelihood of being selected. In contrast to fitness, a lower rank is better such that individuals on the true (first) Pareto front have a rank of 0.

Tournament selection is used to select individuals based on their rank. Since the ranks are discrete, there is need for a tiebreaker. In the NSGA-II algorithm a metric called *crowding distance* is used to break ties. A

cuboid is defined around the i^{th} solution such that the sides extend to the solutions in the same front adjacent to it along each dimension. The crowding distance is then calculated as the average length of the sides of the cuboid. Individuals in lower density regions thus have a higher crowding distance (which is preferable), making it a measure of sparsity in optimization space. In addition, it ensures the prioritisation of the solutions at the endpoints of the non-dominated front which is desirable since these represent the highest performance along at least one dimension, which is an advantage of crowding distance over other sparsity metrics based on average Euclidian distance to nearest neighbours. The main benefit of using crowding distance is however a more even distribution of solutions along the Pareto front. In other words the crowding distance promotes diversity in optimization space which in turn decreases the likelihood of premature convergence.

Since the non-dominated solutions are equally optimal in the Pareto sense it is not possible to choose one (or a few) solutions to preserve through elitism. Preserving the entire first front is generally not viable since a large fraction of the population often belong to the first front once the algorithm has started to converge, meaning evolution would essentially be halted.

2.4 Finite element method

The finite element method (FEM) is a widely used numerical technique for modelling and analysing complex physical objects. It revolves around splitting a complex shape into smaller parts (i.e. finite elements) which can be evaluated numerically. FEM has proven to be well suited for structural analysis.

2.4.1 Natural frequency analysis

A structure's natural frequencies are the frequencies at which the structure will oscillate when perturbed. If there exists an external driving force oscillating the structure at a certain frequency, that oscillation will be amplified if that frequency is near one of the structure's natural frequencies. This phenomena is called resonance and can occur in the context of a rocket launch if the natural frequencies (typically the first and lowest frequency) coincides with the frequency of the vibrations from the rocket engine, likely leading to catastrophic failure. Determining the natural frequencies of a system requires solving the following eigenvalue problem:

$$(M^{-1}K - \lambda I)X = 0, \tag{2.1}$$

where M is the mass matrix, K is the stiffness matrix, λ are the eigenvalues, I is the identity matrix, and X are the eigenvectors. There are as many eigenvalues and eigenvectors as there are degrees of freedom in the system. The eigenvalues λ correspond to the natural frequencies of the system, and the eigenvectors correspond to the modal shapes (see Figure 2.5). The i^{th} natural frequency f_i is determined from the i^{th} eigenvalue as

$$\lambda_i = \omega_i^2 \text{ [rad/s]}, \tag{2.2}$$

$$f_i = \frac{\omega_i}{2\pi} = \frac{\sqrt{\lambda_i}}{2\pi} \text{ [Hz]}. \tag{2.3}$$

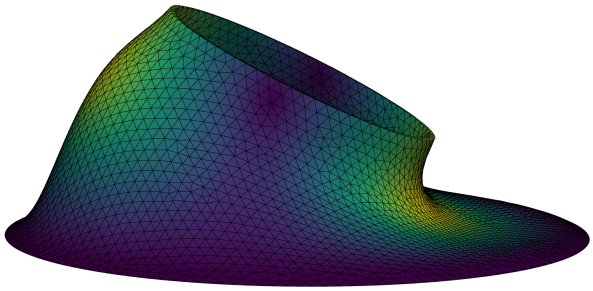


Figure 2.5: *First modal shape of an LVA structure. The displacement of the structure, as described by the first eigenvector X_0, is multiplied by 150 for the purposes of visual demonstration. Observe that the modal shape is that of lateral (side-to-side) vibration.*

3 Method

The method developed for this project consists of three parts: (1) a solution generator, (2) an evolutionary optimizer, and (3) a graphical user interface (GUI). The architecture of the code is primarily dictated by the iterative nature of a GA, as can be seen in Figure 3.1. While it is true that the three parts are connected they are largely modular, in the sense that they can be modified or replaced independently. Different methods for defining and synthesising structures is necessary for any level of generalizability.

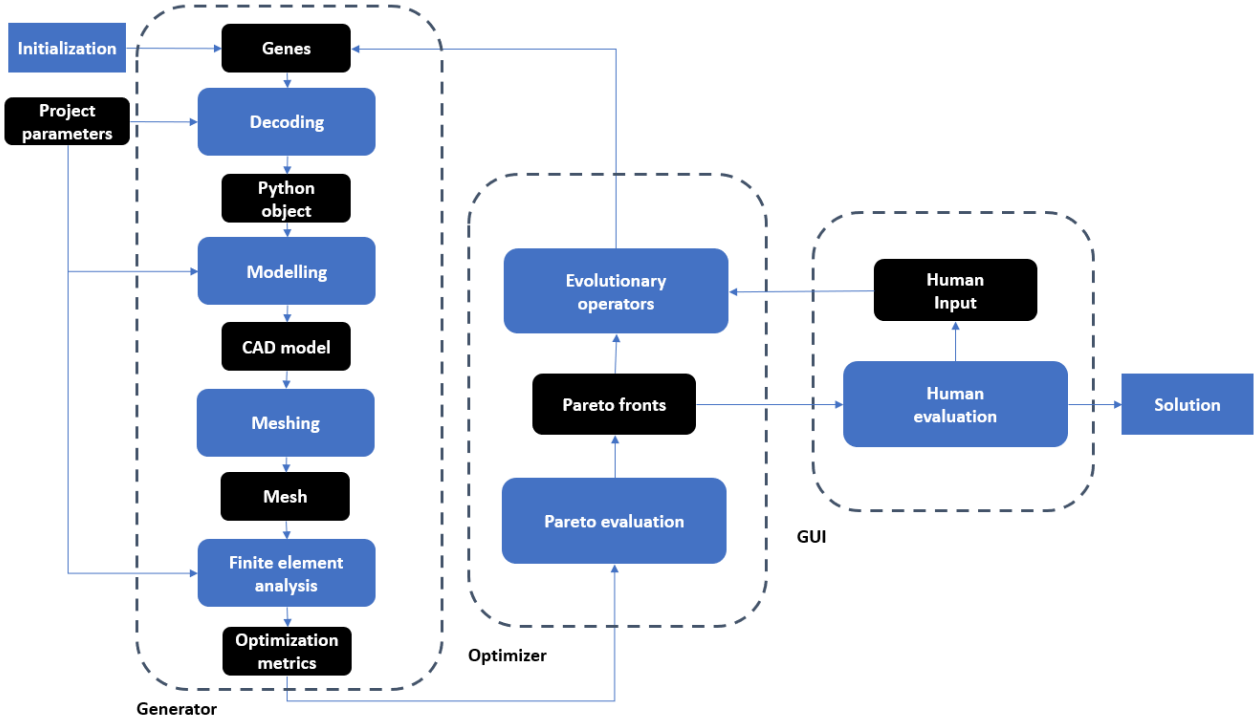


Figure 3.1: *Schematic representation of the method developed for this thesis. Each column represents one of the three modules. From left to right: (1) solution generator, (2) evolutionary optimizer, and (3) graphical user interface. The shapes colored blue within these modules represent functions performed, and those colored black represent information passed between the functions.*

The genotype must be translated to the phenotype before the individual can be evaluated, which is the function of the solution generator. The complexity of this process can vary significantly. In a simple textbook example of a GA the task could be fitting a polynomial function. The genes would then represent the coefficients of an n^{th} order polynomial, the phenotype would be the resulting function, and the fitness would be some measure of the discrepancy between the target and the generated solution. The conversion from genotype to phenotype in this example is straightforward and takes no significant computation. In contrast, the corresponding process is often ambiguous and computationally expensive in practical applications such as this thesis project. Much of this chapter focuses on the sequence of operations required for this specific genotype to phenotype translation.

There are certainly alternative methods that could have been developed and improvements could undoubtedly be made. However, this method demonstrates the feasibility of implementing an automated sequence of operations, using open-source software, that can generate and analyse a structure starting only with an array of real valued numbers (the genotype). By achieving this, the solution generator provides the translation between genotype and phenotype necessary for the evolutionary optimization.

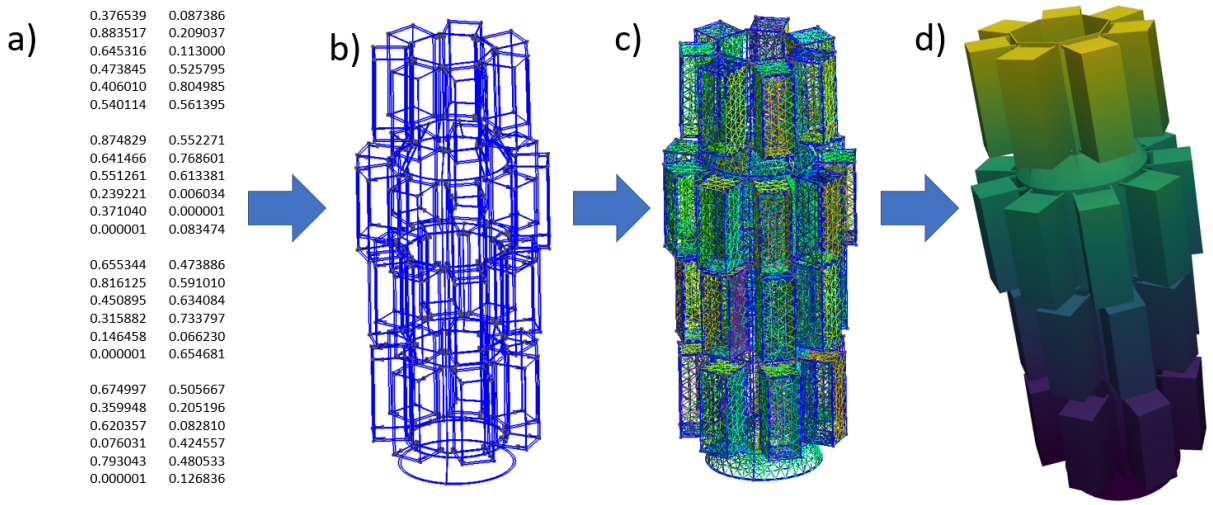


Figure 3.2: *Overview of the solution generation sequence for a dispenser structure. a) The genotype split into chunks corresponding to the satellite tier segments. b) CAD model representation of the structure, generated by decoding the genotype according to the structure definition. c) Mesh generated from the CAD model. d) First modal shape of the structure as determined through finite element analysis, with colors indicating magnitude of displacement.*

3.1 Project specific parameters

Each project comes with a unique set of parameters. The word *parameters* is used in a broad sense and includes fundamental things such as the type of structure being considered (since the method is not restricted to any specific structure). Geometrical constraints, industry standards, and requirements posed by the client are all parameters that contribute to shaping the project and thus also shaping the concept exploration space. It is necessary and desirable to constrain the concept exploration since any solution must comply with the constraints of the project, otherwise the solution is meaningless.

The geometrical constraints of a structure depend largely on physical boundaries and interfaces. Tolerances or safety margins can either exacerbate pre-existing constraints or introduce new ones. Conforming to the many requirements associated with a project is to be considered a feature as opposed to a limitation since, as stated in the previous paragraph, a solution is meaningless if it does not satisfy the boundary conditions no matter how well it performs in isolation.

It is however possible to allow the algorithm to treat hard constraints as soft penalties, allowing for the possibility of breaking some constraint for a significant improvement in performance. The result would then include (and be dependent upon) a suggested change in the constraints. This approach is outside the scope of this thesis.

3.1.1 Dispenser

The physical design space of a dispenser structure is limited by the geometry of the payload fairing and the satellites. There are additional constraints stemming from tolerances, such as the required space between the inner fairing wall and the satellites. The values of many parameters, including the aforementioned buffer between satellites and the fairing, are determined by industry standard practises and are accepted as is in the context of this thesis.

The fairing used as a test case is the *Atlas V 5-m Long* fairing, see Figure 3.5. The fairing is modelled for visualization purposes but is not included in the analysis of the structure. This particular fairing was chosen because the dimensions were suitable for the payload, publicly available, and trivial to model.

The satellite used as a test case is not a real satellite but a realistic approximation of typical satellites launched with this type of dispenser system. In addition to having reasonable physical dimensions and mass, the satellites are modelled to give an accurate stiffness contributions to the dispenser structures.

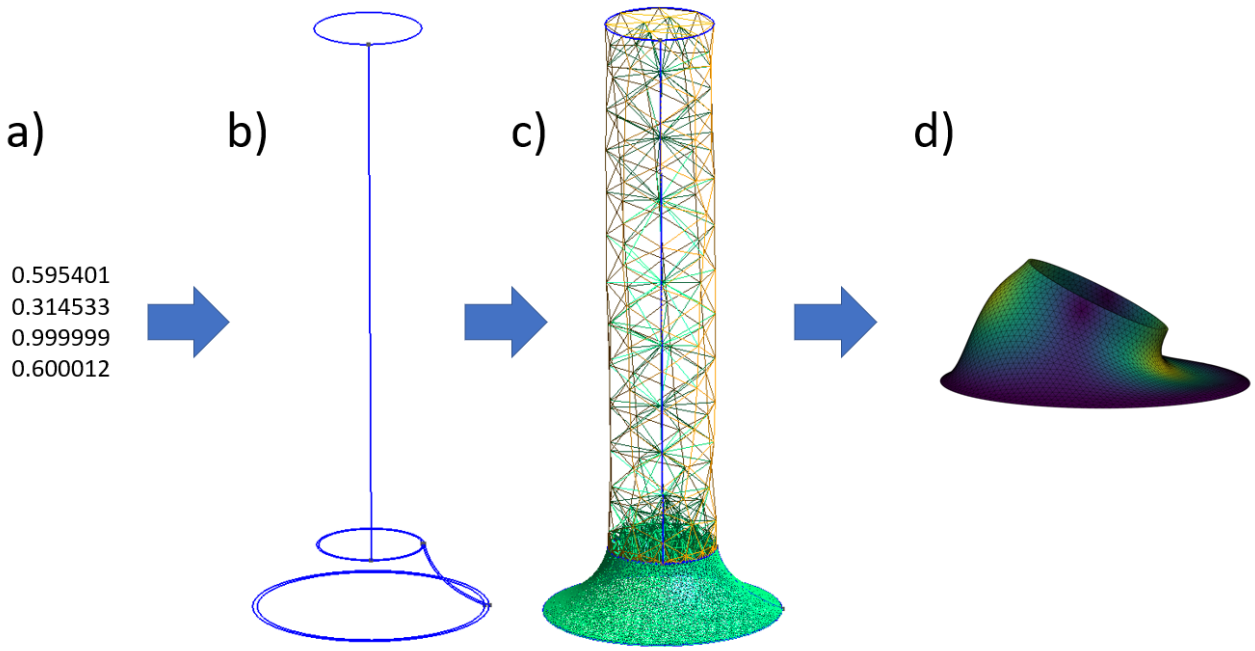


Figure 3.3: *Overview of the solution generation sequence for an LVA. a) The genotype. b) CAD model representation of the structure, generated by decoding the genotype according to the structure definition. c) Mesh generated from the CAD model. d) First modal shape of the structure as determined through finite element analysis, with colors indicating magnitude of displacement.*

The dispenser structure itself is made of carbon-fibre-reinforced-polymer (CFRP). In reality this material is highly anisotropic but a simplified isotropic model is used in this thesis, see Table 3.1 for material properties.

Property	Isotropic CFRP	Aluminium	LVA dummy load
Density ρ $[kg/m^3]$	1750	2700	1042
Young's modulus E $[GPa]$	100	70	100
Poisson's ratio ν	0.2	0.35	0.2

Table 3.1: Material properties.

3.1.2 Launch vehicle adapter

Due to the satellites and the confined space of the fairing, the design space of the dispenser structure is very restricted. The LVA does not have any satellites and therefore allows for more complex and interesting shapes, likely leading to a more interesting result of the concept exploration. In addition, a different structure definition scheme is used in which the shape of the structure is defined by its profile as opposed to several intermediate cross-sections. The third justification for focusing on the LVA in addition to the dispenser structure as a whole is to demonstrate the generalizability of the method.

The test case for the LVA is based on industry standard interfaces. The diameter of the (lower) interface with the launch vehicle is 2624 mm and the (upper) interface with the payload is 1194 mm. The net slope is 45° resulting in a total height of 715 mm. The value of the net slope was chosen as a challenging but realistic test case. The material of the LVA is the same isotropic CFRP used for the dispenser structure.

3.2 Decoding

Each individual in the GA population is represented by a chromosome which contains a predetermined constant number of genes. The number of genes, and how that number is determined, varies with the structure being

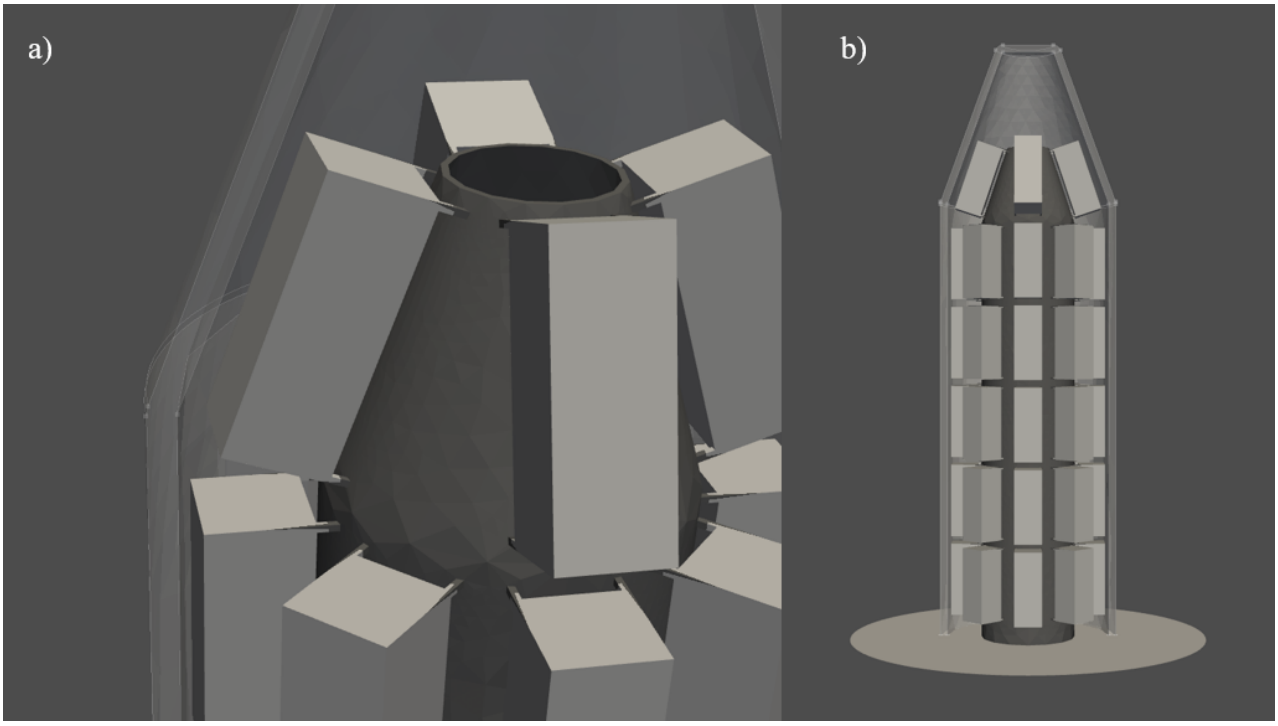


Figure 3.4: *a) Close-up of satellites. Each satellite interface with the dispenser at four points. The colors indicate different materials. The uneven transition (in color) is due to the resolution of the mesh. b) dispenser system placed in the fairing. The particular dispenser shown here was manually created in the framework of the structure definition used in this method, meaning it is able to be expressed but has not been evolutionarily generated. Note: only having 4 satellites in a tier, as is the case in this figure, is not possible with the settings used to produce the results in this thesis.*

considered. Decoding means interpreting the genes and connecting them to some characteristic of the structure.

Different structures will in general have different decoding schemes, as is the case here. The decoding schemes for both structures does however use the same method for interpreting the values of genes, even though these values are connected to fundamentally different characteristics of the respective structures. In both cases the genes are in the range (0,1) and are used for interpolation between a minimum and a maximum value (see Eq. 3.1). The upper and lower limits for the parameters are determined by project specific constraints, for example the maximum radius being determined by the inner radius of the fairing.

$$f(f_{\min}, f_{\max}, g) = f_{\min} + (f_{\max} - f_{\min}) \cdot g, \quad g \in (0, 1) \tag{3.1}$$

The way in which the structures are synthesized from the chromosomes explicitly determines what structures can and cannot be generated. Since only structures definable by the decoding scheme can be expressed, much care and thought should be put into which features are implemented and which are not. The tailor made structure definition step is thus the primary defining factor of the subsequent search space.

Formulating an appropriate structure definition can thus be interpreted as the process of designing the system that then designs the structure, meaning human influence is shifted and abstracted, but not removed. Human bias is thus reintroduced which can be considered a drawback of the method. However, in practise the design of the decoding scheme allows the engineer to set necessary limitations on the solution space, for example based on geometrical requirements or restrictions stemming from manufacturing capabilities. If performed properly this step improves the search by narrowing the set of all solutions to the set of all feasible solutions. If performed improperly, the problem which this thesis aims to address is reintroduced by implementing a decoding scheme which excessively limits variability. Allowing the algorithm to be free and to explore as widely as possible, while also producing results that are both recognizable and implementable are the two competing values, and it is certainly possible to go too far in either direction. In order to strike a balance it is important to be in dialogue with experts as they can determine which variables would be interesting to incorporate and which can be excluded with certainty.

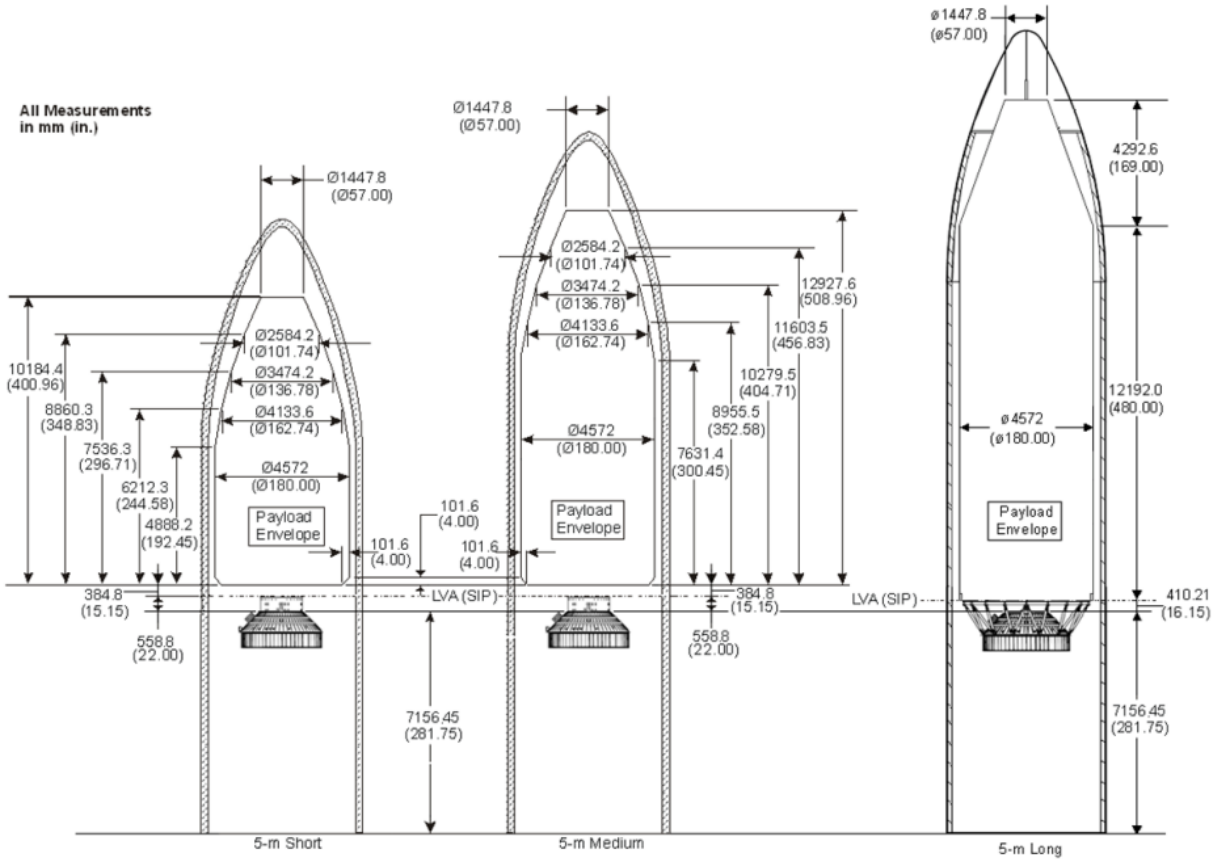


Figure 3.5: *Atlas V 5-m Simplified Static Payload Envelopes [2]. The model 5-m Long (rightmost) is used as a test case in this thesis. Figure reproduced with kind permission from United Launch Alliance.*

3.2.1 Dispenser

A dispenser system is typically divided into several tiers, which is the basis for the implemented structure definition. The generated dispenser structures are made up of several segments, stacking upwards from the bottom. The chromosome is split into chunks corresponding to the segments. The first gene in each chunk determines whether or not the corresponding segment is activated or not by comparing the gene's value to a threshold value (by default 0.5), meaning the dispenser structures are made up of a variable number of segments. This feature results in the structure not being strictly parametrized and allows for varying levels of complexity. Exploring new concepts requires an increase in complexity assuming the conventional design is of low geometric complexity. However, when implementing a decoding scheme which allows for more complex shapes it is important that the conventional (i.e. less complex) shape also can be expressed in that same scheme, otherwise it is difficult to make any comparisons between any newly discovered designs and the conventional design, or even determine whether or not the higher complexity is warranted. Variable chromosome length could be an alternative to encoding activation of segments. The latter was chosen mainly for ease of implementation.

The satellites affects the geometrical requirements of the segments, since they must fit between the dispenser structure and the fairing wall. The uppermost satellite tier segment is allowed to be tapered while the rest have a vertical profile to ensure proper satellite separation. This means that the upper and lower radii of the satellite tier segments are the same, and only one gene is required to determine the radius of the entire segment. Since different segments can have different radii a transition is required between the satellite tiers. This is achieved by introducing special transitions segments which are not encoded in the genotype but adapt to the segments above and below. The transition segments do not carry satellites.

The total number of genes in a chromosome is the product of the number of genes per segment and the maximum number of segments. The segments themselves are parametrized and defined by the genes. As mentioned the first gene determines whether or not the segment is activated, the second gene determines the cross sectional shape of the segment (e.g. circular or polygonal), the third gene determines the number of

satellites to place on that segment (if the segment is polygonal this parameter also determines the number of sides), the fourth gene determines the offset from the previous segment (i.e. the height of the transition segment between the two), the fifth gene determines the radius of the segment (if polygonal this is the inscribed radius), and the sixth gene determines the thickness of the segment at its top (see Table 3.2). The result of the decoding is a list containing segment objects, where each such object contains the information needed to model the segment.

Some genes may be dormant depending on their position in the chromosome. If a segment is deactivated all genes of that segment (except the first gene) are dormant. Dormant genes still take part in the evolutionary operations, but they do not get expressed in the phenotype and so they do not influence the fitness of the individual. Dormant genes can be activated by mutations or relocation to an active part of the chromosome through crossover (analogous to a feature of biological genes known as atavism [28]).

Gene	Variable	Range or options
0	Activation	on/off
1	Shape	circular/polygonal
2	Array number	7/8/9/10
3	Offset [mm]	(100, 1000)
4	Radius [mm]	(750, 2286)
5	Top thickness [mm]	(30, 50)

Table 3.2: Function of each gene in a chromosome chunk representing a satellite tier segment. Note that there are several such chunks in a chromosome. All options are determined by thresholds proportional to the number of options. Activation: whether or not the segment is active (threshold 0.5). Shape: circular or polygonal cross-section. Array number: number of satellites attached to the tier, if the shape is polygonal then this number determines the number of sides. Offset: vertical distance from the top of the previous tier to the bottom of this tier, corresponding to the height of the transition segment connecting the two. Radius: radius of the segment if circular, inscribed radius of the segment if polygonal. Top thickness: thickness of the structure at the top of the segment, thickness is interpolated linearly.

3.2.2 Launch vehicle adapter

Unlike the dispenser structure which is built bottom-up one segment at a time the LVA is a defined by a profile which is revolved around the z-axis. The LVA structures are therefore always symmetric about the z-axis and have a circular cross-section.

The outer profile of the LVA is defined by a spline. Splines are piecewise polynomial functions used to create smooth shapes which are easily controlled and parametrized. The position of the endpoints of the spline is fixed and determined by the diameter of the upper and lower interfaces as well as the total height of the LVA. The tangential direction (derivative) of the spline at the endpoints is not fixed, but a variable encoded in the genome. In addition to the endpoints, an intermediate point is used to define the spline. The position of this point, which the spline must pass through, is also encoded in the genome. Two genes determine the position of the intermediate point, corresponding to the two dimensions of the profile plane. The tangent of the spline at the intermediate point is neither fixed nor determined by the genes, but left as a degree of freedom. This results in four genetically encoded parameters per structure: lower tangential angle α_1, upper tangential angle α_2, radial distance from the z-axis to the intermediate point r, and height of the intermediate point z. See Figure 3.6 and Table 3.3.

To avoid issues with the subsequent meshing analysis of the structure the ranges for these parameters are adjusted. The range of the height z is truncated 20 mm in both ends. The tangential angles α_1 and α_2 are similarly truncated to the range $(\frac{\pi}{8}, \frac{7\pi}{8})$. In contrast, the minimum value of r is lower than the upper interface. The reason for this is simply to allow greater variability when possible, even though high performing solutions are unlikely to be found in that region of the search space. The constraints that are introduced are done so to improve reliability of the automated procedure, not to narrow the search.

Gene	Variable	Range
0	r	$(397, 1312)$ [mm]
1	z	$(20, 695)$ [mm]
2	α_1	$(\frac{\pi}{8}, \frac{7\pi}{8})$ [rad]
3	α_2	$(\frac{\pi}{8}, \frac{7\pi}{8})$ [rad]

Table 3.3: Each gene and its corresponding role in defining the LVA structure. r: radius of intermediate point. z: height of intermediate point. α_1: angle of tangent at lower endpoint. α_2: angle of tangent at upper endpoint.

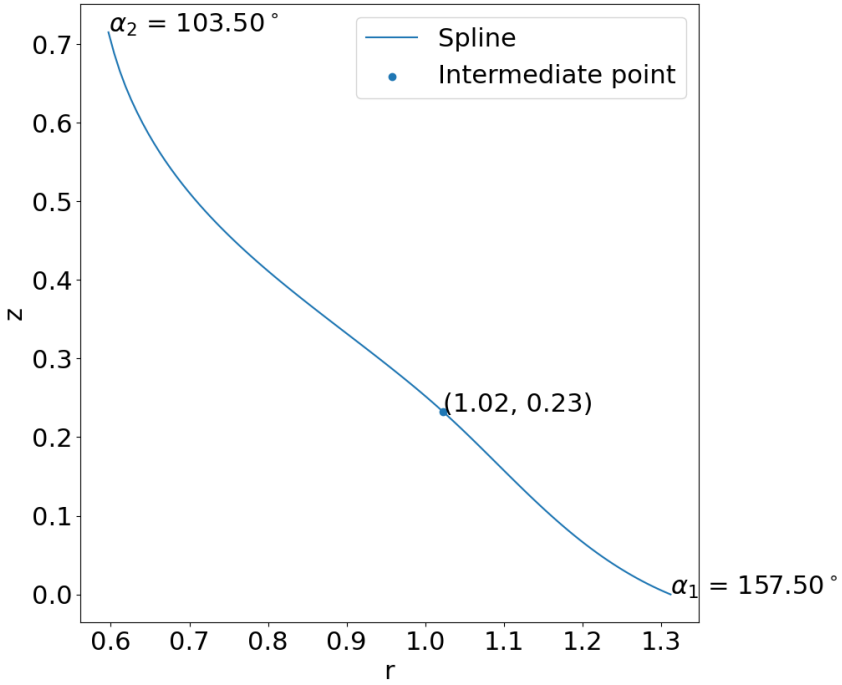


Figure 3.6: *Spline defining the outer profile of an LVA structure. The coordinates of the intermediate point are* (r, z) *as described by the genes (see Table 3.3).* r*: distance to intermediate spline point from z-axis.* z*: height of intermediate spline point.* α_1 *and* α_2*: tangential angle at bottom and top endpoints respectively (reference direction* $r+$*). Values of* r *and* z *given in meters.*

3.3 CAD modelling

The result of the decoding step is a representation of the structure as an instance of a Python object which contains the defining parameters of the structure. The next step is to produce a CAD model from this representation using the Python library CadQuery [6] which builds on the open-source 3D-modelling kernel Open CASCADE Technology (OCCT) [21]. CadQuery provides an interface to the underlying kernel and enables the development of an automated process with Python.

The decoding and modelling steps are interdependent in the sense that the modelling determines what information needs to be encoded in, and extracted from, the genes. Each type of structure (e.g. dispenser or LVA) thus require both the decoding and modelling to be custom made.

3.3.1 Dispenser

A list of segment objects is parsed in order to build the dispenser structure from the bottom up. Note that the list of segments only contain satellite tier segments, with the exception of the first segment in the list which is the LVA segment (not to be confused with the standalone LVA structure). In addition to not hosting any satellites the LVA is special in that it includes the boundary condition as the bottom radius, thickness, and shape are predetermined input parameters.

Between the satellite tiers are special transition segments which are not explicitly encoded in the genome,

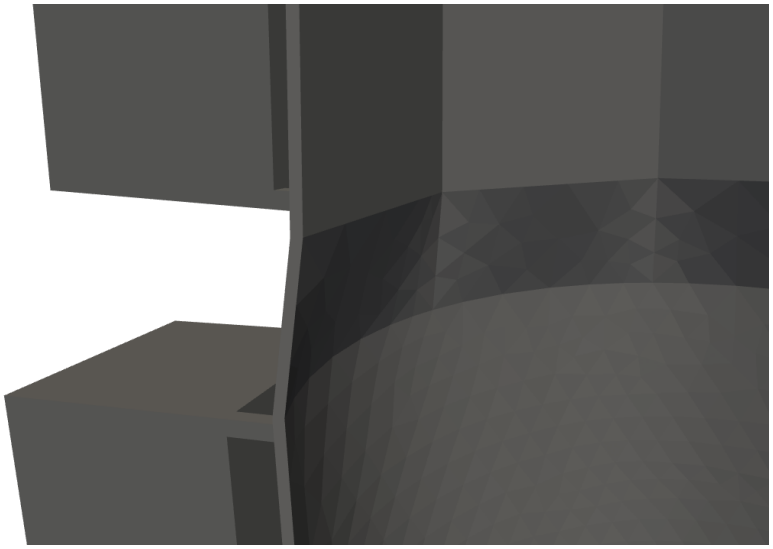


Figure 3.7: *Example of transition from a circular to a polygonal satellite tier.*

but are defined by the segments above and below it. The modelling of the transition segments utilize the *loft* function to enable the connection of a polygonal and a circular cross-section, see Figure 3.7. The top radius of the uppermost satellite tier is decreased as much as possible to taper the segment. The minimum radius of a satellite tier segment is determined by the width of the satellites (including buffer) and the number of satellites to be arrayed on that segment.

Each satellite interface the dispenser structure at four points, see Figure 3.4 a). While this modelling choice results in a more complex model requiring more computation, it is justified by a more accurate stiffness contribution to the dispenser structure from the satellites. Note that the modelling of the interface is still simplified in comparison to a real satellite attachment solution, which can be seen in Figure 2.2. The dimensions of the satellites are variable input parameters. By default the satellites are modelled as boxes with the dimensions 750x750x2000 mm.

3.3.2 Launch vehicle adapter

The CadQuery function *spline* generates a curve given a set of knots (points) and optional tangents. As described in Section 3.2.2 each spline is generated from three knots (both endpoints and one intermediate point) and the tangents at the endpoints. In the underlying OCCT-kernel the spline is represented as a B-spline, and is generated with the *GeomAPI_Interpolate* class.

The spline represents the outer profile of the LVA structure. A 3D-shape is generated by revolving the spline 360° about the z-axis. The CadQuery function *shell* is used to make the shape hollow with a given wall thickness, and to remove the upper and lower faces.

Examining variable wall thickness would be interesting (and is done for the dispenser structure), but not supported for the function used to hollow out the shapes. For this reason all LVA structures have constant thickness. It is certainly possible to achieve variable thickness with a different implementation, but this is left for future work.

A simplified payload is attached atop the LVA to simulate a typical dispenser system. The payload is modelled as a solid cylindrical structure with a diameter matching the upper interface of the LVA (1194 mm). It has a mass of 7000 kg and a centre of mass 3000 mm above the interface. The material density is adjusted to match these parameters.

3.4 Meshing

In order to enable analysis of the structure, a mesh is generated. A mesh is a finite element approximation of the geometry. The CAD model, which is stored as a *.step* file, is automatically meshed using the open source software Gmsh and its Python API.

Different mesh elements can be used to generate a mesh. In this thesis tetrahedral elements are used. This is however not the optimal element type for analysing the relevant structures. Thin walled structures are best analysed using shell elements [1], but this feature was not available in the software used. Tetrahedral elements was therefore considered sufficient as they allow for automatic meshing of most geometries and are widely used. Note, however, that this simplification in choice of mesh element results in inaccuracies in the analysis of the structures, see Section 3.5.

It is crucial to configure the mesh element size properly. If the mesh is too refined the computational cost of generating and subsequently analysing the mesh is too large. In contrast, if the mesh is too coarse the automatic meshing might fail completely or cause excessive inaccuracies in the finite element analysis.

Structures which are both thin and curved pose a challenge. If the mesh element size is too large the edges and surfaces of the inner and outer wall of the structure can overlap, causing the meshing procedure to fail. To prevent this the mesh must either be refined further, or the thickness and curvature of the structure must be limited. Although it is undesirable to further constrain the search space due to the meshing procedure, it may be necessary in order to decrease the probability of failure. Due to the randomness associated with the geometry of the structures it is a challenge to achieve high reliability and robustness of the automated procedure.

Although the meshing software Gmsh includes an interface to the OCCT modelling kernel, it lacks features which were necessary to produce more complex and interesting geometries (e.g. the *loft* feature used to interpolate between cross sections). For this reason Gmsh is used only for meshing, not modelling.

3.4.1 Dispenser

The mesh element size is determined by the curvature of the surfaces to avoid the problem of overlapping elements which occurs due to curved, thin walled structures. The element size is thus adaptive, and varies significantly depending on the characteristics of the individual structure. Circular segments are thus meshed more finely than polygonal segments.

3.4.2 Launch vehicle adapter

There are two different mesh element sizes, one for the LVA itself and one for the payload. The mesh for the payload is much coarser in order to conserve computational efforts. The interface between the LVA and the payload, however, has the same high mesh refinement as the LVA in order to avoid introducing additional inaccuracies.

The approach of determining mesh size from curvature, as is done for the dispenser structure, was tried but resulted in a coarser mesh overall while requiring comparable computation. It is possible to allow a higher level of mesh refinement for the LVA than for the dispenser structure. The dispenser structure is more complex overall but any given segment of the dispenser is less complex when compared to the standalone LVA.

3.5 Finite element analysis

After the mesh has been generated the finite element method (FEM) can be applied. The ultimate goal of the structure generation process is to extract the optimization metrics which are used to determine the fitness of the individual. The final step is thus to evaluate the generated structure through finite element analysis (FEA). Note that the optimization metrics in question are the mass and the first natural frequency (as indicator of stiffness). While the mass can be determined by considering the geometry of the structure and the properties of the material, FEA is required to determine a complex structure's natural frequencies.

Based on the experience of the engineers at Beyond Gravity, the first natural frequency of the structures considered typically corresponds to the lateral vibration mode of the system. This has been confirmed for both the dispenser and LVA structures as modelled in this thesis. Further, if the requirements on the lateral vibrations are achieved, then the secondary vibration requirements are typically achieved as well. This is the justification for the simplified approach of only analysing the first natural frequency and its corresponding modal shape.

The Python library *scikit-fem* [12] is used to assemble and solve the eigenvalue problem formulated in Eq. 2.1. As with previous steps of the structure generation this open-source software is used to enable the automation of the process. The mesh, which is stored in the Gmsh native format *.msh*, is loaded using the Python library *meshio*. The mass and stiffness matrices, M and K, are assembled using the scikit-fem function

linear_elasticity given the Lamé parameters, which are calculated from the relevant material properties. In order to model the structure being firmly secured atop the launch vehicle, the degrees of freedom corresponding to the bottom nodes are eliminated. This fixates their position and defines the boundary condition for the system. The eigenvalue problem is solved using SciPy's eigenvalue solver.

The open-source software can be considered lightweight in comparison to commercial structural analysis software (e.g. ABAQUS) used in industry, and some inaccuracies are likely. One source of discrepancy between the structural analysis performed in this thesis and the corresponding analysis performed in the industry is the mesh element type. For thin structures it is typically desirable to use shell elements [4, 1] whereas tetrahedral elements are used in this method because shell element were not available in the software used. This simplification in terms of mesh element type leads to inaccuracies in the analysis since tetrahedral elements can provide unintended and non-physical stiffness contributions. This effect is largely dependent on the quality of the mesh, which in turn depends on the mesh size and the complexity of the geometry.

For the purpose of the evolutionary optimization the relative accuracy is more important than the absolute accuracy. Relative accuracy means that a structure which in reality is stiffer than another should be recognized as such in the simulation, and vice versa. The absolute accuracy of the extracted frequency is not as crucial, and some margin for error is therefore allowed in the interest of minimizing complexity and computation. Even relative accuracy could suffer from poor meshing and overly simplified analysis. While this is difficult to verify there are no indications of such fundamental inaccuracies in this thesis. Note however that one ought to be careful when comparing results from different analysis methods of varying complexity and sophistication.

To assess relative accuracy one can compare frequency and modal results from one simulation method with another, or optimally with experimental results on physical structures. This was not done in any systematic or comprehensive manner, and relative inaccuracies can therefore not be ruled out. However, it has been confirmed that the first modal shape, which corresponds to the first natural frequency, is the lateral vibration mode of the structure. This matches the expected behaviour of the type of structures modelled.

3.6 Optimization

In order to generate structures with increasingly high performance a GA has been implemented. Since the performance of the structures cannot be determined from a single parameter, a multi-objective approach is necessary. To this end the Pareto front based evaluation procedure of NSGA-II [7] is used. Note that other evolutionary operators, such as mutation and recombination, are not effected by the multi-objective fitness evaluation. There are many choices of methods and hyperparameters associated with implementing a GA, a thorough and systematic hyperparameter optimization is however outside the scope of this thesis. The method choices and parameter settings are presented below. The same GA is used for both the dispenser structure and LVA structure optimization, although the objectives differ. The dispenser structure is optimized for three objectives: higher natural frequency, lower mass (excluding mass of satellites), and larger number of satellites. The LVA structure is optimized for two objectives: higher natural frequency and lower mass. The corresponding optimization spaces are thus three- and two dimensional, respectively. Note that the dimensionality of the hypersurface Pareto front therefore also differs.

Parameter	Value
Population size	50
Mutation rate, p_{mut}	0.2
Crossover rate, p_{cross}	0.75
Tournament probability, p_{tour}	0.8

Table 3.4: Hyperparameter settings for the GA.

As mentioned above the evaluation step is different from a standard GA in order to accommodate multi-objective optimization. To determine which individuals belong to which Pareto front the individuals are compared to establish whether or not one dominates the other. As a result each individual has a list of other individuals it dominates as well as a count of how many individuals it is dominated by. The individuals that are non-dominated (has a domination count of 0) form the first Pareto front. The individuals of the first front are then ignored when the procedure repeats to establish the second front, and so on. During the procedure each individual is assigned a rank corresponding to the front it belongs to, lower being better. A standard tournament selection procedure is then carried out using this rank as fitness, and a sparsity metric (crowding

distance) as tiebreaker.

For the dispenser structure the rank is shifted by penalties representing minimal requirements (e.g. minimum total height or minimal number of satellites). Each failure to fulfil a requirements results in a rank penalty (increase in rank). This is an indirect way of limiting the solution space and is done instead of modifying the decoding and modelling. Without these penalties (or analogous caveats of the structure definition) the optimization algorithms will, for example, produce dispenser structures with very few or no satellite tiers as smaller structures tend to have less mass and higher stiffness. The LVA structure has no such penalties applied.

Mutation is done using a Gaussian creep function. Each gene is mutated with the probability p_{mut}. If a gene mutates, it is shifted with a value drawn from a Gaussian distribution, see Figure 2.3 (b). Crossover is done with probability p_{cross} using the method of two-point crossover as visualized in Figure 2.3 (a).

3.7 User interface and interaction

A rudimentary GUI application was developed using PyQT5. The purposes of the GUI are: (1) visualization of both final and intermediate results of the concept exploration, and (2) enabling of user interaction.

The method of penalising (or promoting) solutions by shifting their Pareto front rank can be used to introduce human interaction into the optimization [15], thus extending the concept exploration to include elements of IEC (see Section 2.2.2). In doing so the user can represent an additional optimization criterion not easily quantified, for example cost or difficulty of manufacturing the structure. Alternatively the role of the user could be identifying structures that, while in the context of the optimization algorithm perform well, would be likely to fail in reality for whatever reason [15].

In the scenarios described above the human user aids the algorithm by, in some sense, covering for its shortcomings. It would likely be beneficial to have the human user strive for objective evaluation in these cases. In contrast, one could imagine a scenario in which the user input is used to steer the exploration in a certain direction, based on the subjective interest of the human user. In the latter case the algorithm aids the human by exploring regions of design space surrounding those solutions favoured by the user.

Incorporating human interaction was intended in this thesis work, and while the capability was successfully implemented it was not able to be properly tested or evaluated. An additional challenge is the relatively long run time between generations (several minutes) which would make real time interaction difficult.

3.8 Parallel computation

The solution generation sequence is executed in parallel using multiprocessing in order to reduce runtime. In addition, multithreading is used to avoid the GUI freezing during runtime. Support for GPU-utilization or distributed computing is however left for future work.

4 Results

The following chapter presents the results of two GA runs, one for each type of structure. The shared hyperparameter settings for the evolutionary operators are presented in Table 3.4. For the dispenser structure the GA ran for 54 generations, taking approximately 25 minutes per generation, or 22.5 hours in total. For the LVA the GA ran for 101 generations, taking approximately 3 minutes per generation, or 5 hours in total. The GA was run for 10 generation increments, with disruptions caused by occasional failures of the meshing procedure (discussed in Section 5.1) leading to the irregular generation counts. Evolution was stopped once the algorithm converged, meaning no significant improvements were observed for 5-10 generations.

4.1 Dispenser

In this section the results of the GA run for the dispenser structure are presented. Note that several runs were made during the course of development, but that comparisons cannot be made to these since the circumstance of the optimization changed with further development.

In Figure 4.1 the final generation is presented as seen through the GUI. A selection of six solutions are presented, with the color of the floor corresponding to the color of the datum in the scatter plot in the bottom right. Solutions belonging to the first Pareto front with the largest crowding distance are prioritized for visualization. Some *archived* solutions are included in the scatter plot. These are solutions from previous generations which are visualized to indicate change over the course of evolution.

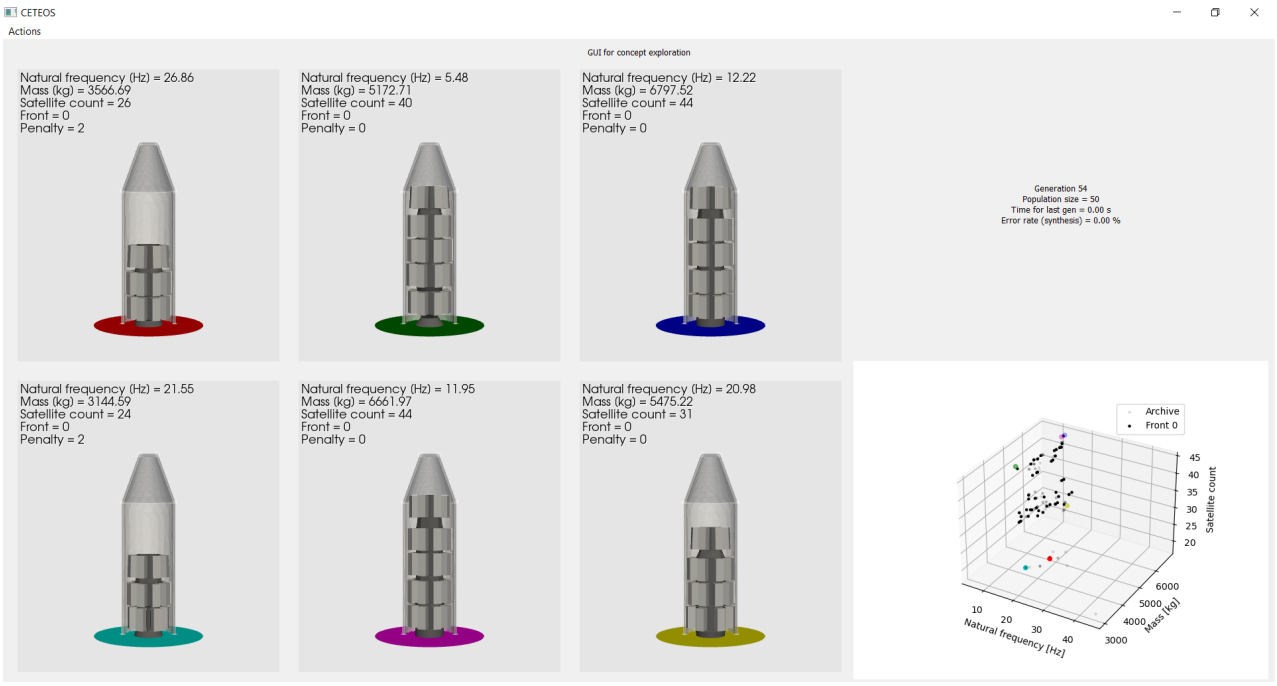
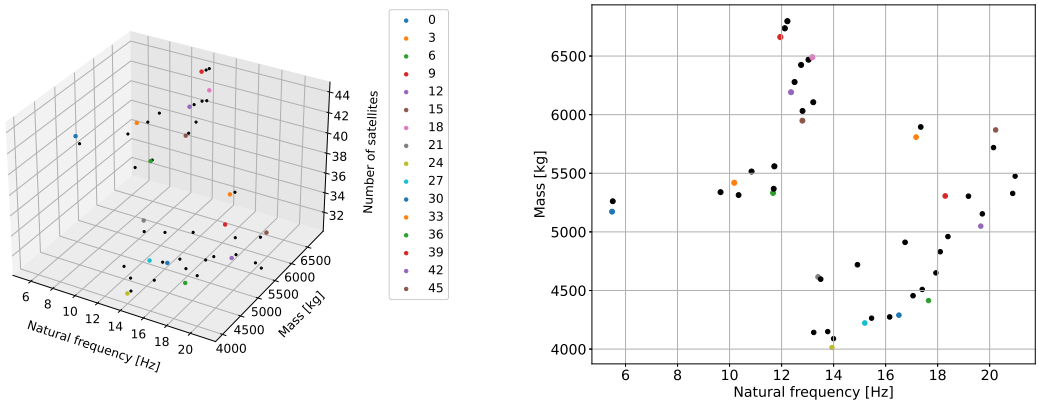


Figure 4.1: *GUI after completion of GA run. Population size: 50. Number of generation: 54. Approximately 25 minutes per generation (total 22.5 hours). Time for synthesis of the last generation (and the corresponding error rate) is displayed in the top right field, along with the current generation index and the population size. When reloading a population for visualization purposes the synthesis time and error rate are not recovered and thus incorrectly shown to be 0.*

Only solutions of the last generation on the first Pareto front are included in Figure 4.2. Archived solutions and solutions with penalties applied are excluded which is the reason that Figure 4.2a differs from the scatter plot included in the GUI in Figure 4.1. A subset of solutions are numbered in order to visualize a representative set in following figures. The numbering is sorted by increasing natural frequency.

The GA does converge to a Pareto front as demonstrated in Figure 4.2a. The solutions approximate a 2D-surface in the 3D optimization space. These solutions are quite clearly divided into two clusters, and are



(a) (b)

Figure 4.2: *(a) 3D-scatter plot in optimization space, showing the Pareto front in the last generation of the GA. Two linearly separable clusters are observed. Every third solution is numbered and assigned a color. This is done to enable visualization of a representative set in later figures. (b) Projection of the 3D-scatter on the frequency-mass plane. Colour coding consistent with left figure.*

indeed linearly separable, which can be seen in both Figure 4.2a and 4.2b. The latter figure is a projection of the former on the frequency-mass plane. In addition to showing the linear separability of the clusters, Figure 4.2b shows the frequency-mass Pareto front more clearly (compared to the 3D-scatter plot). Note that all solutions included here are non-dominated with regard to the three optimization metrics, meaning all solutions which are dominated in the frequency-mass plane necessarily have an advantage in terms of number of satellites (i.e. the third optimization dimension). These linearly separable clusters correspond to having either 4 or 5 active satellite tier segments, which can be confirmed by studying the structures in Figure 4.3. Adding another satellite tier means adding more mass, all else being equal. It also means a taller structure, which tends to result in lower natural frequency. For these reasons the clustering appears in the frequency-mass plane as well, which is visible in Figure 4.2b. The two clusters can be said to represent two different types of satellite dispenser designs, or two different *design concepts*.

While the algorithm does find a variety of mutually non-dominated solutions, and certain design concepts can be identified, the results are quite difficult to interpret further. There is of course variation within the clusters, not least in terms of number of satellites, since the number of satellites per tier varies (and is encoded genetically). Structures that look quite similar visually (e.g. 6 and 18, or 27 and 39) exist in different regions of the Pareto front. Part of this variance within groups is no doubt explained by the varying wall thickness, which is not visualized. But it is difficult to determine with any certainty which characteristics (e.g. wall thickness) has what effects, and to what degree. No good method has been found to analyse the differences within the clusters systematically and coherently. It is therefore difficult to explain, and draw conclusions from, the details of the results. Despite the dispenser structure being the original test case, and therefore influencing the method development significantly, the difficulty in interpreting the results ultimately suggests that the method is not well suited to the structure, or vice versa. This realization led to the adaptation of the method to the second test case, launch vehicle adapters.

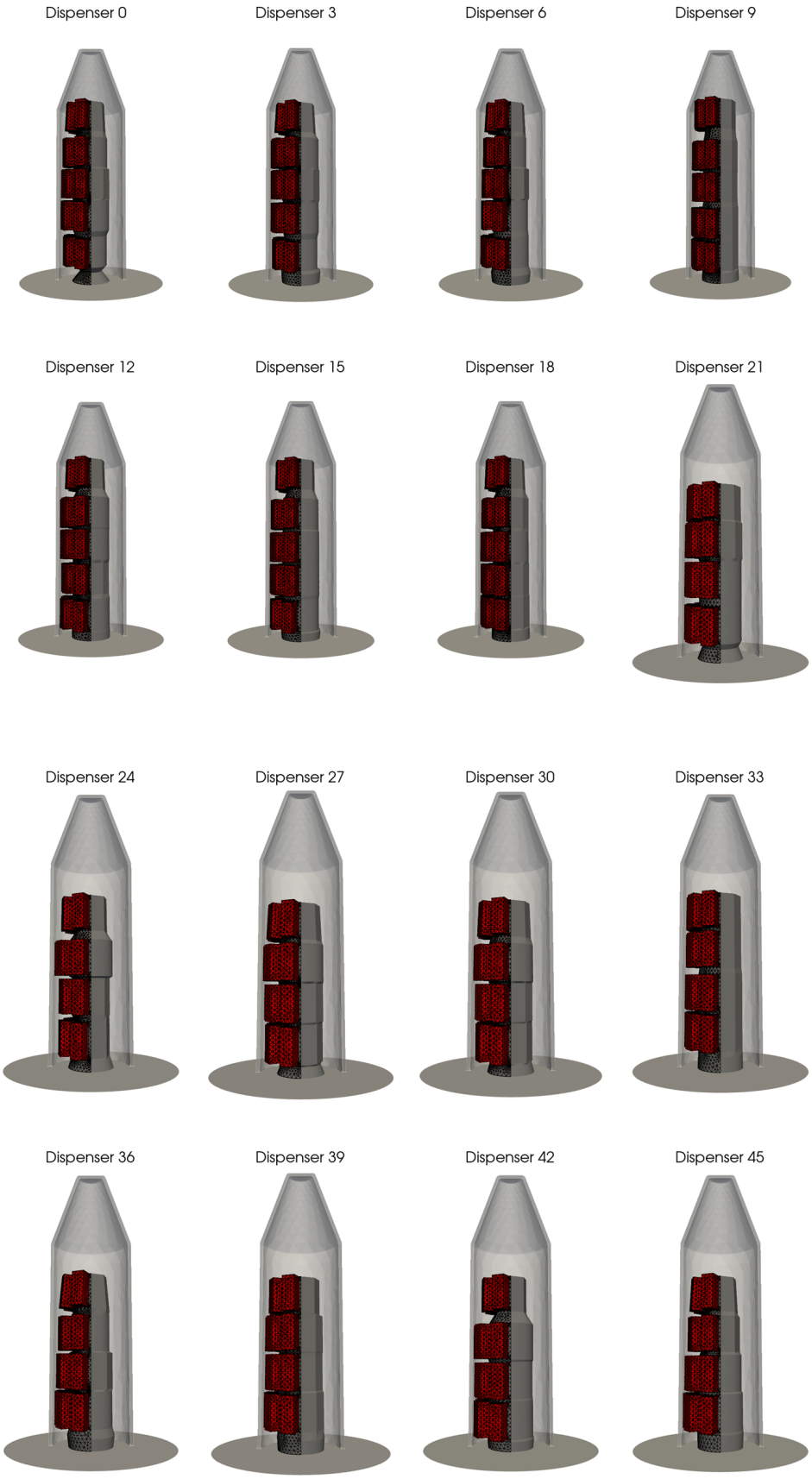


Figure 4.3: *Visualization of the numbered subset of solutions from the final generation. Numbering consistent with Figure 4.2a, sorted by increasing natural frequency. Two meshes are combined to show the dispenser structure both with and without satellites attached.*

4.2 Launch vehicle adapter

In this section the results of the GA run for the LVA are presented. Note that several runs were made during the course of development, but that comparisons cannot be made to these since the circumstance of the optimization changed with further development.

In Figure 4.4 the final generation is presented as seen through the GUI. A selection of six solutions are presented, with the color of the floor corresponding to the color of the datum in the scatter plot in the bottom right. Solutions belonging to the first Pareto front with the largest crowding distance are prioritized for visualization. Some *archived* solutions are included in the scatter plot. These are solutions from previous generations which are visualized to indicate change over the course of evolution.

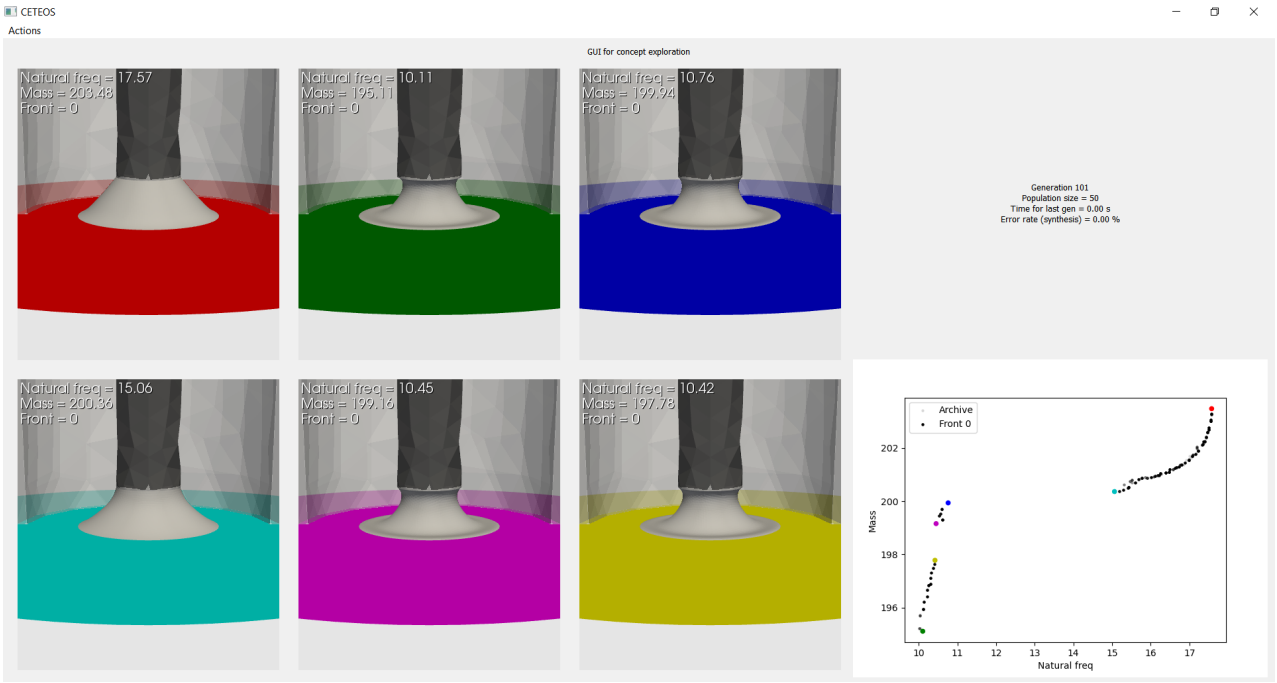


Figure 4.4: *GUI after completion of GA run. Population size: 50. Number of generation: 101. Approximately 180 seconds per generation (total 5 hours). To the left a selection of solutions are presented, with the color of the floor corresponding to the color of the datum in the figure to the right. Solutions of the first front with the largest crowding distance are prioritized for visualization. Time for synthesis of the last generation (and the corresponding error rate) is displayed in the top right field, along with the current generation index and the population size. When reloading a population for visualization purposes the synthesis time and error rate are not recovered and thus incorrectly shown to be 0.*

Two different classes of solutions emerge. In Figure 4.4 one can observe that the red and teal solutions are of a certain type, and belong to the same continuous part of the Pareto front. The four other solutions presented in the GUI are of a different type and belong to the other part of the Pareto front. While there is variation within the classes, two fundamentally different design concepts can be identified. This is further visualized in Figure 4.5. As is discussed in detail in Section 5.2.1, the solutions of class A are found through exploitation of the solution generation procedure, and are not viable solutions. These solutions are therefore ignored for the remainder of this section. The solutions of class B are however viable and will be analysed further.

In Figure 4.6 the part of the Pareto front corresponding to class B is enhanced. As these are the viable solutions, this part of the front is to be considered the result of the LVA concept exploration. A subset of the solutions in Figure 4.6 are numbered and assigned a color in order to enable the visualization of a representative subset, which is done in Figure 4.7. Further, the splines defining the structures are presented in Figure 4.8 to more clearly visualize the differences within the class of solutions. The splines indicate a continuous range (of shape profiles) corresponding to the Pareto front in optimization space. Whether or not one could draw conclusions from interpolating within this range (i.e. define the relation between mass and natural frequency) is not investigated further in this thesis project, but an interesting topic for future work.

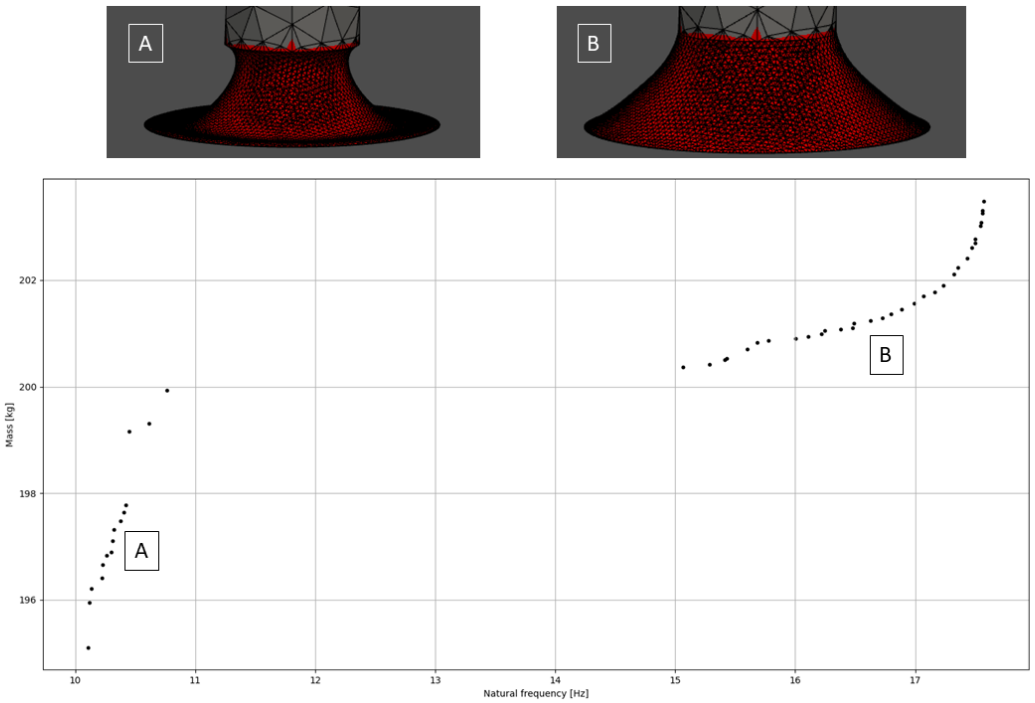


Figure 4.5: *Pareto front of the final generation. Two classes of solutions are identified by their visual appearance which corresponds to two regions in optimization space. These are denoted A and B respectively. Class A is a spurious solution type exploiting certain mechanics of the solution generation, this is discussed in Section 5.2. Class B is a viable set of solutions.*

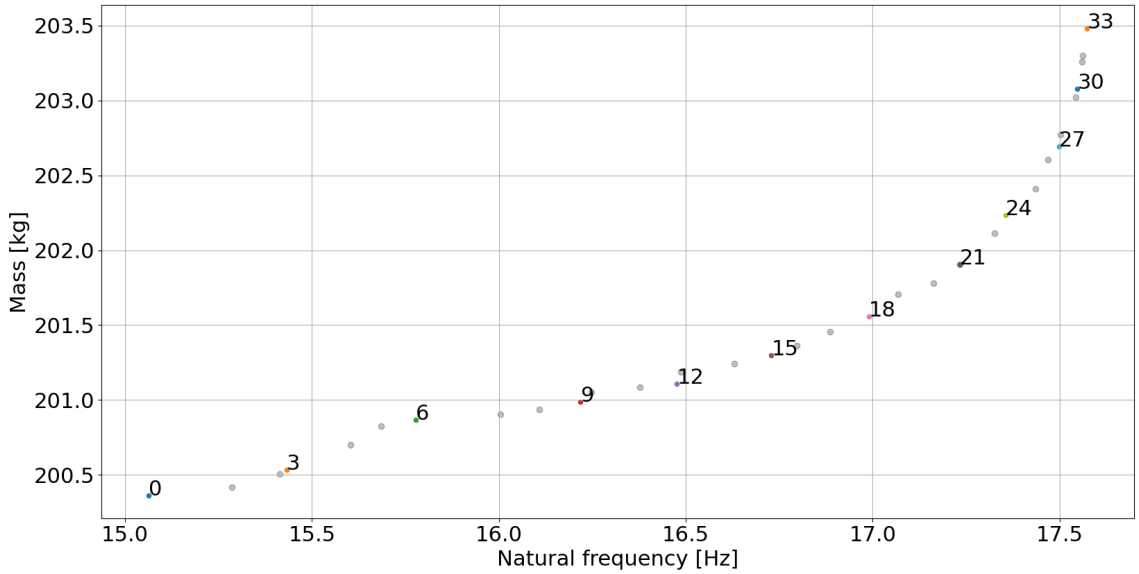


Figure 4.6: *The part of the Pareto front denoted B in Figure 4.5. Every third solution is numbered and assigned a color. This is done to enable visualizing a representative subset of these solutions in subsequent figures.*

The performance of the solutions can be compared to the conventional design solution, a regular cone. The analysis of this control case is presented in Figure 4.9. The natural frequency and mass of the regular cone are 16.88 Hz and 209.15 kg respectively. The solution with highest natural frequency, LVA 33, thus has 4.1% higher

natural frequency and 2.9% lower mass than the control case. The reason for the cone being heavier relates to the topic of Section 5.2.1, namely that the solutions of class B (just as those of class A) have a very flat tangent at the lower interface, meaning more volume is removed compared to the cone (which has a tangential angle of 45°). The tangential angle of the viable solutions all approach the maximum allowed angle, $\frac{7\pi}{8}$ radians.

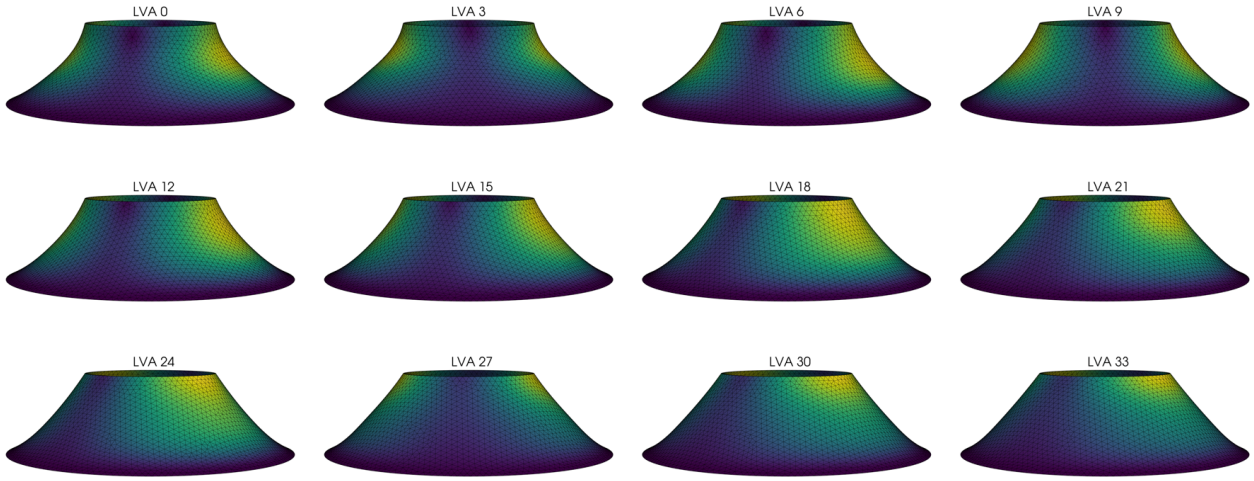


Figure 4.7: *Rendered LVA structures. Numbering is consistent with previous figures. A gradual transition in shape can be observed, especially by comparing LVA 0 with LVA 33. The color scheme represents the magnitude of the displacement associated with the first modal shape. The direction of the mode is stochastic.*

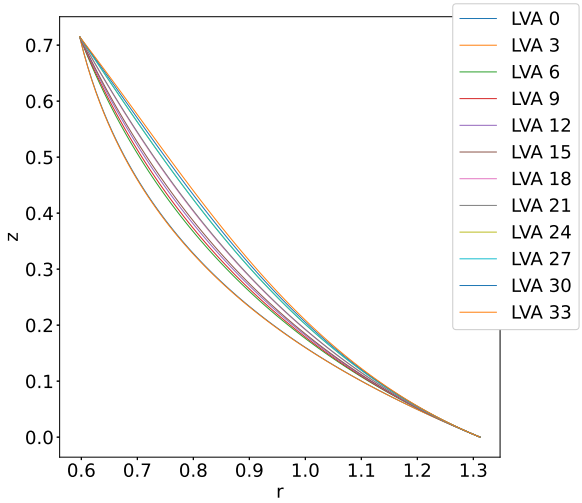


Figure 4.8: *The splines for each numbered solution. The spline corresponds to the outer profile of the revolved LVA structure, where the endpoints are at upper and lower interfaces. The aggregate of the splines suggest a range of viable splines corresponding to the Pareto front shown in Figure 4.6.*

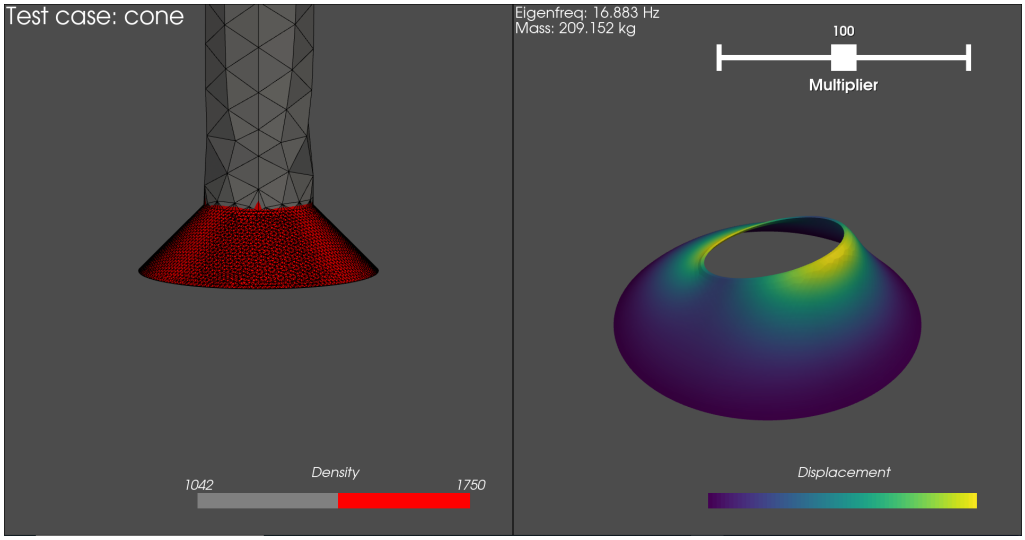


Figure 4.9: *A regular cone LVA expressed by the structure definition used in this thesis. The chromosome was made manually for the purpose of comparison. All relevant parameters are the same as for the LVAs presented in this section.*

5 Discussion and conclusion

This thesis project has studied whether it is feasible to use machine learning tools to augment concept exploration in the design of structures such as satellite dispenser structures, or launch vehicle adapters. The result of the thesis is thus not only the output of the optimization of any specific test case (presented in the previous chapter), but also the developed method as a whole. The goals of the project as specified in Section 1.4 reflect this, and will be referenced when assessing and discussing the project.

5.1 Structure generation

The first goal, to develop a tool which autonomously generates feasible solutions, has demonstrably been achieved. The procedure referred to in this report as the *solution generator* has proven successful in that the module is able to generate a solution given a representation in the form of an artificial genome and a structure definition based on project specific parameters. An arbitrary number of solutions can be generated once the user has defined the permitted design space.

The structure generation procedure is the most complex and computationally expensive part of the project. Unsurprisingly, it was therefore also the primary source of challenges. High reliability and fast execution are crucial when the solution generator is seen in the context of the optimization process, since thousands of solutions need to be generated autonomously. The frequency of failure is very much dependent on the type of structure being generated. The meshing, especially of the dispenser structure, was the most common point of failure. It is difficult to identify the reasons for failures of that kind with certainty, but they are likely to involve the comparatively high complexity of the dispenser structure and the incorporation of adaptive modelling (loft features) and meshing (mesh size from curvature).

Any failure of a structure generation instance risks interrupting the GA. In an attempt to increase robustness, any individual for which the solution generation failed was replaced with a new, randomly generated individual. This approach was successful for the most part, but some failures in the meshing procedure led to the algorithm entering a functionally infinite loop instead of throwing appropriate exceptions. In these instances the GA also halts, and has to be restarted. While the progress is saved after each completed generation, this problem nevertheless made running the GA for a significant number of generations challenging, since some level of supervision was necessary. This problem occurs less often if the mesh is refined further, but the trade off is longer run times. An appropriate solution to the problem would be to implement timeout callbacks for the solution generator function, such that an upper time limit can be specified for the procedure (perhaps adaptively in proportion to the successful generation of viable structures).

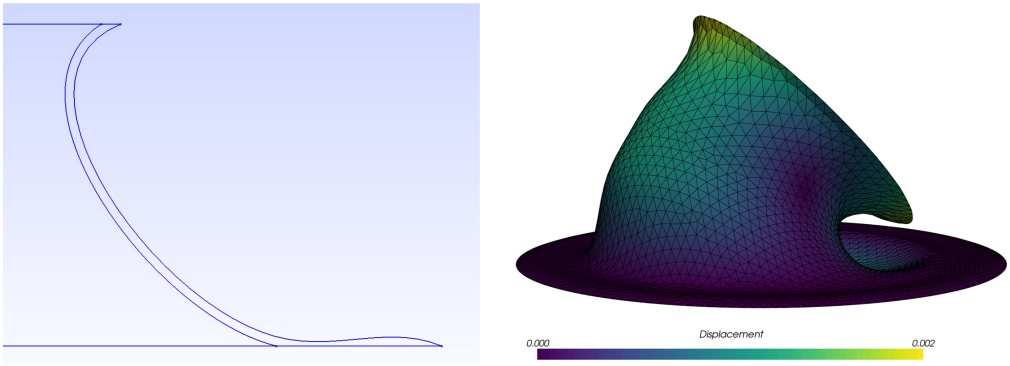


Figure 5.1: *An illustration of how the LVA solution Class A exploits the structure generation by creating a thinner wall than intended and fixating more nodes than intended. Displacement exaggerated 200 times.*

5.2 Optimization

The second goal, to implement a genetic algorithm which iteratively optimizes the generated solutions, has also been achieved. The NSGA-II algorithm was successfully implemented in Python, and it successfully converges to a Pareto front that looks much as expected. Tweaking and benchmarking of the algorithm was not prioritized because the computation time of the optimization is negligible in comparison to that of the solution generation. Moreover, the performance of the optimization was considered satisfactory.

Note that both optimization runs presented in Chapter 4 converged, meaning negligible additional benefit was observer during the final 5-10 generations of the run. Nevertheless, it is impossible to conclude how close the Pareto fronts found are to the true optimal front. That is to say, there may be even better solutions which, for whatever reason, were not discovered. To gain higher confidence it would be desirable to make several independent runs in order to aggregate and compare results.

5.2.1 Spurious solutions

Two different classes of solutions emerge from the LVA optimization, one of which is a spurious set of solutions found through exploiting features of the solution generator. By converging to a spline that follows the base plane ($z = 0$) the structure becomes thinner than intended. While the modelling does not allow the spline (outer wall) to extend below that plane, the inner wall is able to do so once the spline has been given its thickness. Part of the volume is thus removed, resulting in a thinner structure and thus a lower mass. The same class of splines take advantage of a second oversight in the FEA step. The nodes interfacing with the launch vehicle are fixated through elimination of the corresponding nodal degrees of freedom. These nodes are identified based on their z coordinate, i.e. all nodes for which $z = 0$ are fixated. Due to the same strategy of following the base plane, more nodes than intended get fixated as if they are connected to the launch vehicle, leading to a higher stiffness.

While the first part of the exploit could be seen as drawing outside the lines of the intended structure definition, the second part is essentially cheating. These exploits are unintended consequences of choices made during development of the method. A new run could be carried out with the exploits remedied, but since the viable class of solutions is well represented this was decided against in order to demonstrate the phenomenon of spurious solutions. The ability of the algorithm to find and use exploits is important to be aware of, both as a developer and as a user. In this case the spurious class of solutions formed a distinct part of the Pareto front which made it easy to identify. In a less fortunate circumstance the spurious solutions might share a front with the viable solutions making them hard to distinguish. In the worst possible case the viable front is dominated by a spurious front, leading to the elimination of the viable class over the course of evolution.

5.3 User interface

The third goal, to expose the user to a broad set of feasible solutions, is addressed by the development of the GUI, which can display several solutions in a given generation, in combination with the multi-objective optimization

approach. The formulation of the goal leaves room for interpretation, but by displaying a representative subset of the Pareto front in the GUI the goal is at least partially achieved.

A related goal, to give the user a better understanding of the design space, is addressed by achieving the aforementioned goal (exposing the user to a broad set of feasible solutions). In addition to displaying the Pareto optimal solutions, the optimization space itself is visualized in the GUI with the intention of enabling the user to form an intuition of the mapping between types of solutions (as visualized by the GUI) and their respective position in the optimization space.

Yet another goal, to give the user higher confidence in design decisions, relates to the same mechanisms. The aim of the GUI is largely to give the user insight into what concepts (type of solutions) are being evaluated throughout the optimization process, and subsequently which of these concepts perform well. Resulting from this is hopefully a better understanding of the solution space, leading to more informed design decisions moving forward. By showing the user what is ruled out (and why), the user presumably gains increased confidence in what is left. This approach does however run the risk of being naive since the algorithm is far from flawless, as can be seen in Section 5.2.1.

5.4 Concept exploration

The fourth goal, to propose novel design concepts which outperform the conventional design, is perhaps the most important goal of all concept exploration and structural optimization. It is easier to argue for this goal having been achieved through the LVA results, as opposed to the dispenser results. The dispenser structure definition is in some sense too complex, leading to results which are difficult to analyse coherently. The results of the LVA are much more interpretable, which is likely due to the simpler structure definition. The feature which was introduced for the LVA concept exploration was essentially profile curvature. By defining the structure with a spline a large range of shapes can be expressed, which are almost exclusively of higher complexity than the conventional design (truncated cone).

The results of the LVA concept exploration, perhaps best visualized in Figures 4.7 and 4.8, indicate a gradual transition between two different design concepts. This gradual transition is also evident by studying the Pareto front in Figure 4.6. By comparing the two extremes, LVA 0 and LVA 33, the differences can most clearly be examined. LVA 0 has a concave shape, with a very steep tangent at the upper interface. LVA 33, while concave at the bottom, transitions to an almost perfectly straight profile with a flatter tangential angle, seemingly converging to a regular truncated cone in its upper half. Further, the location of the maximum displacement associated with the modal shape differs. For LVA 0 the maximum is closer to the middle of the structure, while for LVA 33 the maximum is at the top.

Novelty search was experimented with as an alternative method of concept exploration, with the intention of further promoting variance and niching of the solutions. In addition, substituting the optimization method would demonstrate the modularity of the concept exploration tool. However, it proved difficult to define the behaviour space within which novelty is to be evaluated. Because of this, a novelty search variant of the algorithm was left for future work.

5.5 Model discrepancies

As mentioned throughout the report, several simplifications have been made to the modelling, meshing, and analysis of the structures. These simplifications have been made out of necessity with regard to either time for development, computation, or method availability. The discrepancies, which are discussed below, have been identified by comparing the method developed in this thesis to standard industry practices.

The materials used for simulation are modelled as isotropic. In the case of CFRP this is a simplification since the material in reality is highly anisotropic. It is not clear what type of inaccuracies this simplification causes in the analysis of the structures. The simplification was made due to the author's lack of experience with modelling of anisotropic materials and uncertainty of availability of such modelling techniques in the open-source software used. Note that the material modelling is secondary to the shape of the structure, and while some additional inaccuracy is introduced the overall result (in terms of shape) may still hold. There are also cases where structures (such as an LVA) are made of isotropic materials, in which case the modelling method is suitable. Modelling anisotropic materials does however present an interesting avenue for future work, since the anisotropies can be controlled in the case of CFRP and incorporated into the optimization framework.

In other words, one could encode the direction of the carbon fibres in the genome, examples of which exist in the relevant literature [20, 24].

The mesh element chosen in this project, tetrahedral elements, is another source of discrepancy. The tetrahedral elements can introduce unintended and non-physical stiffness contributions when they are ill-shaped, which is often the case with thin structures. This means that the mesh element size limits how thin a structure can be modelled without excessive inaccuracy. The problem can be addressed by refining the mesh further but this once again poses computational problems. A better approach would be to incorporate an implementation of the appropriate element type, shell elements.

The boundary condition of the finite element analysis fixates the nodes concurrent with the $z = 0$ plane. This removes all degrees of freedom (which are the three translational dimensions). In reality the interface is not a perfectly fixed boundary.

5.6 Conclusions

The final goal, to contribute to introducing the space industry to machine learning ideas and methods, is difficult to assess. This project has, at the very least, proposed auxiliary methods for finding alternative solutions. Further, the generation of Pareto optimal fronts, especially in combination with visual and interactive tools, shows promise as an addition to the design process.

The central goal of the project, to show that evolutionary computation is a viable method for concept exploration, has been addressed by the project in its entirety. Whether the goal has been achieved or not is a matter of opinion and degree. If the resulting Pareto optimal front of LVA structures can be said to represent one or more design concepts, and if other design concepts have been considered and discarded along the way, then it is reasonable to conclude that concept exploration has successfully been carried out. It is the hope of the author that the design concepts discovered using the method developed in this project be modelled and scrutinized using the techniques and tools available in industry.

References

[1] *About shell elements*. URL: https://abaqus-docs.mit.edu/2017/English/SIMACAEELMRefMap/simaelm-c-shelloverview.htm (visited on 2022).

[2] United Launch Alliance. "Atlas V Launch services user's guide". In: *Lockheed Martin Commercial Launch Services* (2010).

[3] Arianespace.com. *OneWeb cut-away image*. URL: https://www.arianespace.com/media/the-36-oneweb-satellites-are-shown-on-their-dispenser-system-in-this-cut-away-image-flight-st30-36-oneweb-satellites/ (visited on 2022).

[4] L Börgesson. "Abaqus". In: *Developments in geotechnical engineering*. Vol. 79. Elsevier, 1996, pp. 565–570.

[5] Jonathan Byrne et al. "Combining structural analysis and multi-objective criteria for evolutionary architectural design". In: *European Conference on the Applications of Evolutionary Computation*. Springer. 2011, pp. 204–213.

[6] *CadQuery*. URL: https://github.com/CadQuery/cadquery (visited on 2022).

[7] Kalyanmoy Deb et al. "A fast and elitist multiobjective genetic algorithm: NSGA-II". In: *IEEE transactions on evolutionary computation* 6.2 (2002), pp. 182–197.

[8] Daniel Dennet. *Darwin's dangerous idea: Evolution and the meanings of life*. Simon and Schuster, 2014.

[9] Agoston E Eiben and Jim Smith. "From evolutionary computation to the evolution of things". In: *Nature* 521.7553 (2015), pp. 476–482.

[10] Peter Fortescue, Graham Swinerd, and John Stark. *Spacecraft systems engineering*. John Wiley & Sons, 2011.

[11] Juarez Moara Santos Franco et al. "Shape Grammar of steel cold-formed sections based on manufacturing rules". In: *Thin-Walled Structures* 79 (2014), pp. 218–232.

[12] Tom Gustafsson and Geordie Mcbain. "scikit-fem: A Python package for finite element assembly". In: *Journal of Open Source Software* 5.52 (2020), p. 2369.

[13] John H Holland. "Genetic algorithms". In: *Scientific american* 267.1 (1992), pp. 66–73.

[14] Gregory Hornby et al. "Automated antenna design with evolutionary algorithms". In: *Space 2006*. 2006, p. 7242.

[15] Raffi Kamalian, Hideyuki Takagi, and Alice M Agogino. "Optimized design of MEMS by evolutionary multi-objective optimization with interactive evolutionary computation". In: *Genetic and Evolutionary Computation Conference*. Springer. 2004, pp. 1030–1041.

[16] Sivam Krish. "A practical generative design method". In: *Computer-Aided Design* 43.1 (2011), pp. 88–100.

[17] Joel Lehman and Kenneth O Stanley. "Evolving a diversity of virtual creatures through novelty search and local competition". In: *Proceedings of the 13th annual conference on Genetic and evolutionary computation*. 2011, pp. 211–218.

[18] Joel Lehman and Kenneth O Stanley. "Novelty search and the problem with objectives". In: *Genetic programming theory and practice IX*. Springer, 2011, pp. 37–56.

[19] Hui Li and Qingfu Zhang. "Multiobjective optimization problems with complicated Pareto sets, MOEA/D and NSGA-II". In: *IEEE transactions on evolutionary computation* 13.2 (2008), pp. 284–302.

[20] A Muc and W Gurba. "Genetic algorithms and finite element analysis in optimization of composite structures". In: *Composite Structures* 54.2-3 (2001), pp. 275–281.

[21] *Open CASCADE Technology*. URL: https://www.opencascade.com/open-cascade-technology (visited on 2022).

[22] Yan Pei and Hideyuki Takagi. "Research progress survey on interactive evolutionary computation". In: *Journal of Ambient Intelligence and Humanized Computing* (2018), pp. 1–14.

[23] Justin K Pugh, Lisa B Soros, and Kenneth O Stanley. "Quality diversity: A new frontier for evolutionary computation". In: *Frontiers in Robotics and AI* 3 (2016), p. 40.

[24] Rodolphe Le Riche and Raphael T Haftka. "Optimization of laminate stacking sequence for buckling load maximization by genetic algorithm". In: *AIAA journal* 31.5 (1993), pp. 951–956.

[25] Jan Sokolowski and Jean-Paul Zolésio. "Introduction to shape optimization". In: *Introduction to shape optimization*. Springer, 1992, pp. 5–12.

[26] William M Spears et al. "An overview of evolutionary computation". In: *European conference on machine learning*. Springer. 1993, pp. 442–459.

[27] Hideyuki Takagi. "Interactive evolutionary computation: Fusion of the capabilities of EC optimization and human evaluation". In: *Proceedings of the IEEE* 89.9 (2001), pp. 1275–1296.

[28] Nenad Tomić and Victor Benno Meyer-Rochow. "Atavisms: medical, genetic, and evolutionary implications". In: *Perspectives in Biology and Medicine* 54.3 (2011), pp. 332–353.
[29] Bogdan Tomoiagă et al. "Pareto optimal reconfiguration of power distribution systems using a genetic algorithm based on NSGA-II". In: *Energies* 6.3 (2013), pp. 1439–1455.
[30] David A Van Veldhuizen, Gary B Lamont, et al. "Evolutionary computation and convergence to a pareto front". In: *Late breaking papers at the genetic programming 1998 conference*. Citeseer. 1998, pp. 221–228.
[31] David A Van Veldhuizen and Gary B Lamont. "Multiobjective evolutionary algorithms: Analyzing the state-of-the-art". In: *Evolutionary computation* 8.2 (2000), pp. 125–147.
[32] Mattias Wahde. *Biologically inspired optimization methods: an introduction*. WIT press, 2008.
[33] Brian G Woolley and Kenneth O Stanley. "Exploring promising stepping stones by combining novelty search with interactive evolution". In: *arXiv preprint arXiv:1207.6682* (2012).
[34] Yin Yu et al. "On the influence of optimization algorithm and initial design on wing aerodynamic shape optimization". In: *Aerospace Science and Technology* 75 (2018), pp. 183–199.
[35] Aimin Zhou et al. "Multiobjective evolutionary algorithms: A survey of the state of the art". In: *Swarm and evolutionary computation* 1.1 (2011), pp. 32–49.
[36] Luca Zimmermann, Tian Chen, and Kristina Shea. "A 3D, performance-driven generative design framework: automating the link from a 3D spatial grammar interpreter to structural finite element analysis and stochastic optimization". In: *AI EDAM* 32.2 (2018), pp. 189–199.

DEPARTMENT OF MECHANICS AND MARITIME SCIENCES
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2022
www.chalmers.se



CHALMERS
UNIVERSITY OF TECHNOLOGY