A BIOLOGICALLY INSPIRED METHODOLOGY FOR MULTI-DISCIPLINARY DESIGN OPTIMIZATION

A THESIS SUBMITTED TO THE
GRADUATE DIVISION OF THE
UNIVERSITY OF HAWAI'I AT MĀNOA
IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

IN

MECHANICAL ENGINEERING

DECEMBER 2010

By

MIGUEL ALEXANDRE NUNES

Thesis Committee:

Marcelo H. Kobayashi, Chairperson
Amit Sanyal
Trevor Sorensen
Luke Flynn

To my dear wife, Mara,

who offered me unconditional love and support
throughout the course of this thesis.

To my beloved family,

that has always supported and motivated my education.

ACKNOWLEDGMENTS

I want to thank my advisor Dr. Marcelo Kobayahsi for providing me an opportunity to pursue this research and for assisting me throughout my Master's in the Mechanical Engineering program. He was an outstanding advisor, always creative and challenging, that motivated my studies and my research efforts.

I want to thank my committee members in the order they were added to the list: Dr. Amit Sanyal for challenging me to pursue an education of excellence; Dr. Trevor Sorensen for motivating me to do more research in space missions and challenging orbital problems; Dr. Luke Flynn for giving me the opportunity to work and develop my research on the Hawaii Space Flight Laboratory as a research assistant. To all these members I want to thank for taking the time and effort to be on my thesis committee and support my education.

I also want to thank my dear wife, Mara Nunes, for her constant support, help and patience. This work would not have been possible without her. I am very grateful to Lance Yoneshige for the SolidWorks drawings and for his readiness to help. My sincere acknowledgements to Zachary Lee-Ho for his support since I have arrived to "the office", and to Elizabeth Gregory that so much helped me reviewing this document. Zach and Elizabeth helped making my days more enjoyable during this stressful section of my life - that is special. Last but not the least, I want to thank all the members of the Hawaii Space Flight Laboratory that I have been working with. They have had a very positive influence on my work and my life.

"If there occur some changes in nature, the amount of action necessary for this change must be as small as possible."

Pierre Louis Moreau de Maupertuis, 1698 - 1759;
French mathematician who formulated the principle of least action.

ABSTRACT

Optimization problems in engineering are of major importance for the development of new structures, new materials, and even for new ways of improving engineering that are so demanding in today's industry. The development of a biologically inspired methodology brings new ways for topology optimization to be applied in a multidisciplinary design approach. The process developed in this thesis extends the methodology proposed by Kobayashi for engineering designs to multiply connected regions. The methodology is based on a cellular division model for developing the design topology. The topology generated is then improved using a Genetic Algorithm. The methodology is demonstrated in the design of a structural panel from a satellite at launching conditions. Software was developed to illustrate the applicability of the proposed design approach. The results show how the method improves a given structural problem and compares it to a traditional engineering design.

Keywords: Multidisciplinary Design Optimization, Map *L*-system, Biologically Inspired Structures, Satellites.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

This work concerns the extension to multiply-connected domains of the biologically inspired methodology put forward by Kobayashi and collaborators (computational EvoDevo) [1][2][3][4][5][6][7][8][9][10]. The studies in these papers have shown that enormous gains can be attained by the computational EvoDevo. By extending the methodology to multiply-connected domains, we expect these gains to be enhanced by the inclusion of sub-system placement.

Current design paradigms are usually based on the experience of a designer or engineer on a specific task and frequently optimization is not considered as another approach to a problem design. Using a Multidisciplinary Design Optimization (MDO) approach to engineering problems can give a new understanding of a problem and create new and unexpected results revealing different ways of dealing with it. It brings many different sciences together and creates a synergic effort between them to produce a result that provides more insight onto the problem and the solution.

This work uses a MDO procedure as a population-based method that combines three main subjects:

1. Map *L*-system as a biologically-inspired method for topological modelling of systems,
2. Finite Element Method for structural analysis and
3. Genetic Algorithm for objective topology optimization as another biologically-inspired methodology.

This process develops an optimized topology for any given system that has clearly defined constraints. When topology is referred in this work it holds the connotation of a biological structure of cells as used by the Map *L*-system which represent mechanical structures of systems.

The bio-inspired methodology is developed into the design of a structural part of a satellite - a structural panel of the HawaiiSat-1 - scheduled for launch in late 2011. The main goal set for the System Design Optimization is to optimally minimize the mass of the panel while keeping constraints and safety factors given by the launch segment requirements. Another objective of this work is to optimally place one subsystem in the satellite deck given the same conditions. Based on this multidisciplinary procedure a generalization is made for multiply-connected topologies (e.g. placing holes or subsystems on the structure).

As previously described the methodology presented in this thesis is supported on two other methods inspired on natural processes. The first is for *topological modeling* and is based on the Map *L*-system and more precisely on the *Binary Propagating Map OL-system with markers*[11] or mBPOL-systems for short. This modelling of cellular layers based on the Map *L*-system permits the development of complex topologies that can be of interest to structural design. This modelling method consists of two stages. First is the creation of the topology with the cell division patterns and second, the cell geometry is modelled using some predefined constraints based on the design problem. The other biologically-inspired method is for *topological optimization* which is based on Genetic Algorithms. The combination of these methods added to the newly proposed elements make a structural optimization more efficient and robust for mechanical systems.

The presented methodologies were selected not only to extend the work on EvoDevo but also because it has been shown that algorithms inspired on natural processes have innovative and usually perform better than other traditional methods. These have also been widely applied to various fields ranging from engineering, mathematics and arts. The presented methodology uses two biologically inspired and validated methods: Map *L*-system and Genetic Algorithm, which combine to provide a powerful tool to optimize almost any system that can be represented topologically.

Topology optimization is a problem that has been tackled for more than a century since the first published paper on this subject in 1904 by Michell [12]. This work though important has no practical applications since it involves an infinite number of trusses. In the

1960s, the ground-structure methods were introduced. These methods allowed a practical and rapid topology optimization. Thought these methods are able to handle any kind of objective functions they are still computationally inefficient for large number of variables. From the 1980s, Bendsoe and Kikuchi [13] and others [14] proposed methods based on Finite Element Method as a competitive alternative to other optimization tools. Recently many other approaches have been used for topology optimization which also have been implemented in commercial software [15]. Two categories of the most popular methods are: Solid Isotropic Material with Penalization (SIMP) [16][17][18] and "Hard-kill" methods [19] [20] [21] [22] [23] [24].

The SIMP method assumes that the design variables vary continuously from 0 to 1 instead of being discrete. This continuous approach has the problem of material represented with a number between 0 and 1 may not exist or be very hard to manufacture. Penalization may also result in a solution that is a local minimum and no optimal solution is guaranteed. "Hard-kill" methods are based on rejecting elements on the ground-structure given some criteria. These methods are also heuristic and the best solution obtained may be far from optimal and they will hardly provide a simple but rather a complex structure.

Another category of algorithms is surging for topology optimization which are based on processes found in nature. These algorithms have been developed for solving single and multi-objective optimization problems like ant-colony optimization [25] and particle swarm optimization [26]. For topology optimization Genetic Algorithms [27][28][29], have been used to search for the best solution in the topology domain making them a very attractive method. Genetic Algorithms are also flexible and adaptable to very different problem sets because the search is done using a fitness value that globally represents any aspects to be optimized in the selected problem.

For these reasons Genetic Algorithm for topology optimization was used to support the search for the optimal structure in this thesis. More information and results about the selected biologically inspired methods can be obtained by reading the dissertation by Hugo T. C. Carreira [10] produced under the advisory of Dr. Kobayashi. Dr. Carreira thesis

provides one more resource for the validation and applicability of the proposed biologically-inspired methodology with results that compare it's performance with other methods and with more more gains.

The design procedures are a critical aspect of a satellite mission which may affect not only performance but cost as well. The cost of every kilogram launched into space may range from $5,000 to $50,000 depending on the launch vehicle and orbit. The minimization of the overall weight of any component sent to space then can be an important factor for a mission success and cost control.

This work provides a basis for spacecraft structural-topology design using biomimetic methods whose results obtained are of major interest for structural design optimization. More studies will be done on other satellite subsystems based on this bio-inspired procedure to refine and validate the proposed methodology.

Chapter 2 starts by presenting the methods used to support the research and work presented in this document, including a new methodology for topology generation for multiple-connected components. Chapter 3 introduces the structural and Finite Element Method models that transport the generated topology to a physical domain able to be analysed within the Finite Element Method toolbox selected for this task: COMSOL Multiphysics™ . Chapter 4 discusses the software developed that embeds all methods and tools discussed in chapter 2 and 3 in a single framework. Chapter 5 employs the proposed methodology for multiply connected components to a Satellite deck with one subsystem and presents the results obtained from three different optimization runs and compares with previously designed benchmarks. Chapter 6 summarizes the results and emphasises the practical and financial importance of the proposed methodology for space systems and finally presents ways to improve the present work and other problems where future research may be developed based on the presented methodology.

CHAPTER 2
METHODS

This chapter explains the most important components of the computational EvoDevo method developed by Kobayashi and collaborators that are used on the research presented in this work. It also presents the new tools and methods that support the presented methodology for the optimization of topologies with multiple-connected components. Section 2.1 will start with the Map *L*-system which is a method to generate topological maps using biological principles of cellular division. The second method introduced in section 2.2, Map *L*-system and Connected Components, is to show how to create the new topologies with multiple connected components in the original map. Section 2.2.1, Intersection of Connected Components, approaches the way to avoid intersections of connected components using a proper mathematical formulation. Finally, the method presented in section 2.3 is the Objective Optimization that is used to generate the populations of various topologies using the Genetic Algorithm and select the best individuals given a fitness value.

2.1 Map *L*-system

Aristid Lindenmayer was a Hungarian Biologist who in 1968 introduced a new type of formal language called *L*-system (or Lindenmayer-Systems). This is a parallel rewriting system [30, 31] that is able to model the morphology of various organisms. It is well known for modelling the growth processes of plants using an algorithmic and systematic approach. The rewriting of the *L*-systems differs significantly from the other formal languages' sequential replacement [32] because it allows for the generation of complex models from simple rules in a parallel sequentiation. This formal language was initially used for modelling the development of simple multicellular organisms and the existing relations between the surroundings of the plant cells but was later extended to model complex branching structures and various topological features such as fractals (see example in figure 2.1). *L*-systems have become an important research tool and a scientific theory in itself. *L*-systems are used today in various

fields like music [33], in mathematics of formal languages [32, 34], in computer graphics [35], in generating realistic images of trees and flowers, in artificial intelligence and arts.

String *L*-systems are suitable for the modelling of branching topologies (which is the case of many structures in nature) but this method cannot model structures that are best described by planar graphs with cycles [11]. Map *L*-systems [36, 37, 11] are an extension of the *L*-systems but where the topology must start as closed loop (and always remain bounded) - these are called maps [38]. The rules create branches facing each other inside the map that are connected if the criteria for division are met. This leads to an increasing subdivision of the original map and sequentially the maps are evolved similarly to cellular division processes' in simple organisms. The formalism of Map *L*-system allows the formation of cycles in a structure and the relations between every cell (or region) are well defined. One example of a Map *L*-system is shown in figure 2.2.

Figure 2.1: a fractal plant as an example of a L System

Map *L*-systems are planar graphs which can have a finite number of regions (also called faces in graph theory[1]) as it will be presented in section 2.2.2. These regions are bounded by a sequence of edges which connect the vertices in order. The boundary of a region is defined by a set of edges that are simply connected, these boundaries represent the cells, and the edges represent their walls. The maps represent the cellular layers.

There are different Map *L*-system formulations but this work focuses on the edge controlled formalism of *Binary Propagating Map OL-system with markers*, in short mBPMOL-systems [11]. This work uses a more powerful method originally proposed by Nakamura, Lindenmayer and Aizawa [37, 11] in 1986 as a refinement of the original context-free Map

[1] *regions* is the term preferred in the context of Map *L*-system and *faces* is the term preferred in the context of Graph Theory formulation, this notation is used later in this work.

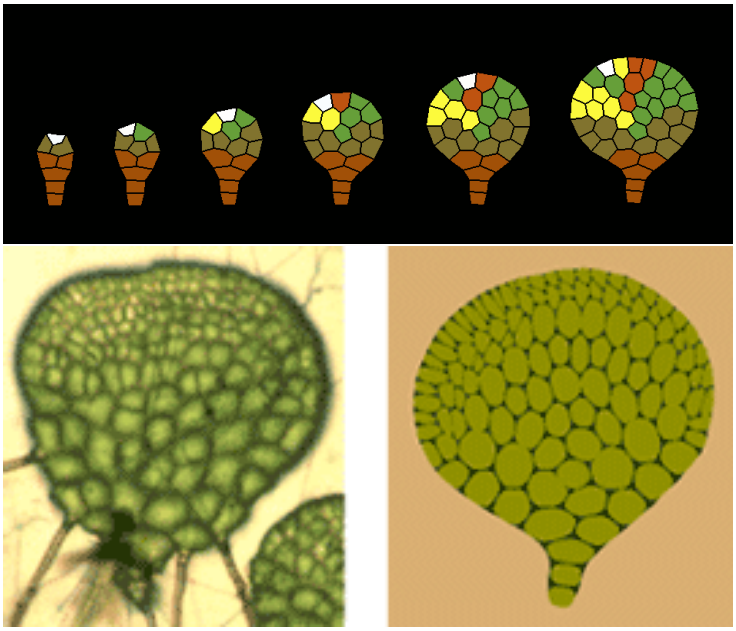


Figure 2.2: Left: Microphotograph of *Microsorium linguaeforme* at magnication 70x; on the right: simulated development of *Microsorium linguaeforme* using the Map *L*-system

L-system presented and introduced by Lindenmayer and Rozenberg in 1979 [36]. The name of the method explains well the process:

- It is **Binary** because during the cell division the cells divide at most in two daughter regions;
- It is **Propagating** because cells cannot merge or disappear;
- **0L systems** refers to context-free parallel rewriting systems that do not allow regions to interact;
- the **markers** specify juncture points on the edges where the cell may divide—these markers are an analog to the biological equivalent attachment sites for division walls during mitosis [11].

The mBPMOL-systems can be described mathematically as an alphabet Σ, an axiom ω and a finite set of rules P. The alphabet Σ is a finite and non-empty set whose elements may be any symbol. These symbols are called letters or tokens. Figures 2.3 and 2.5 will

serve as an example to illustrate the principles of operation for the mBPMOL-systems. An example of an alphabet would be $\Sigma = \{A, B, C..., [,], +, -\}$ and an example of an axiom using the letters from this alphabet is $\omega = ABAB$. The rules can be in the form:

$$\begin{aligned} A &\rightarrow B[-A]x[+A]B \\ B &\rightarrow A \end{aligned}$$

This set-up produces the result shown in figure 2.3. The alphabet Σ can use any symbol including numbers and letters. The method implemented in this work uses numbers. The axiom must have the same number of letters has the number of edges in the initial map, which in the given example is four.

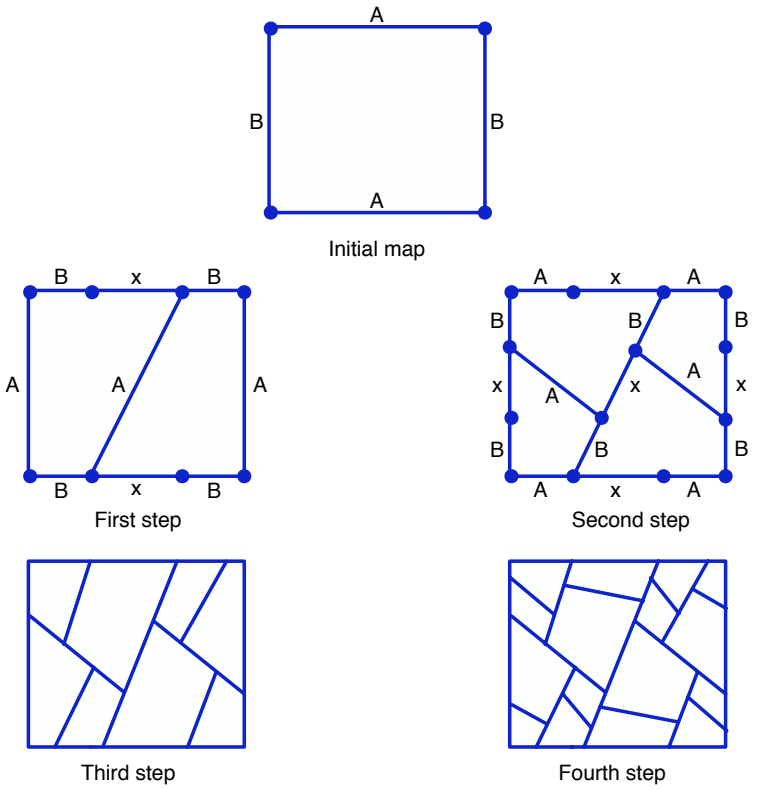


Figure 2.3: Example of the mBPMOL-systems process for the first four steps of the cellular division. This also shows the process of modeling the developmental stages of the topology of the structure using the rules $A \rightarrow B[-A]x[+A]B$ and $B \rightarrow A$.

Every rule in mBPMOL-systems must be in the form $A \to \alpha$ where the edge $A \in \Sigma$ is called the predecessor, and the string α is called the successor. This string is composed of any symbols from Σ including special symbols [,], + and −. To continue using the given example a rule could be of the form $B \to A[-C]$ so when applied to an edge with the label B it would be replaced by $A[-C]$.

To mimic the biological development of cellular division it is necessary to "mark" the edge that will create that division. This is done with the bracket symbols [and] which specify the makers for possible cell-dividing walls. Inside these brackets there are always two symbols. The first symbol is either a − or a + meaning that the marker is to be placed to the left or to the right of the predecessor edge, respectively. This is the notation used in this work but could easily be set differently. The second symbol is always a letter that may or may not carry an arrow over it to represent the local edge orientation of the successor edge relative to the predecessor edge. An example of such rule could be given in the form $A \to B[-\vec{B}]A$ which means the marker is to be placed to the left of the edge and is directed away from A. The remaining symbols (outside the brackets) specify the edge subdivisions where each subdivision will have the same length:

$$\text{length}_{\text{new edge i}} = \frac{\text{length parent edge}}{\text{number of letters outside the brackets}} \tag{2.1}$$

The previous example would produce two new edges with the dimension (lenght of A)/2 and a marker in the middle of the two new edges directed to the left.

The cellular division happens at the derivation phase where at each cell the scanning of markers reveals which markers are matching. This matching is done if two markers have the same label and exist in the same cell but not on the same edge. So, if there are two markers with the same label in a cell, and they lay on different edges and have the same orientation, then a cellular division can be carried connecting the two markers. If more than one pair of matching marker is found in the search then only one of these pairs is to be selected for dividing the region, this happens because of the definition of binary division of the mBPMOL-systems. The remaining markers are discarded.

Some criteria exist for the cellular division to happen after the selection and validation of the edge pair. These are described as follows:

1. Do not allow small angles (see figure 2.4a): the angles between the adjacent edges in the divided cells must be larger than a prescribed lower limit, this prevents the creation of cells with narrow edges angles and is useful for of the mesh generation in the finite element analysis.

2. Do not allow small edges (see figure 2.4b): if the children edges are smaller than a prescribed lower limit then the parent edge are kept.

3. Do not allow small areas (see figure 2.4c): the areas of the offspring cells must be larger than a predefined percentage of the original map, this precludes excessively slender or relatively minute offspring cells to form.

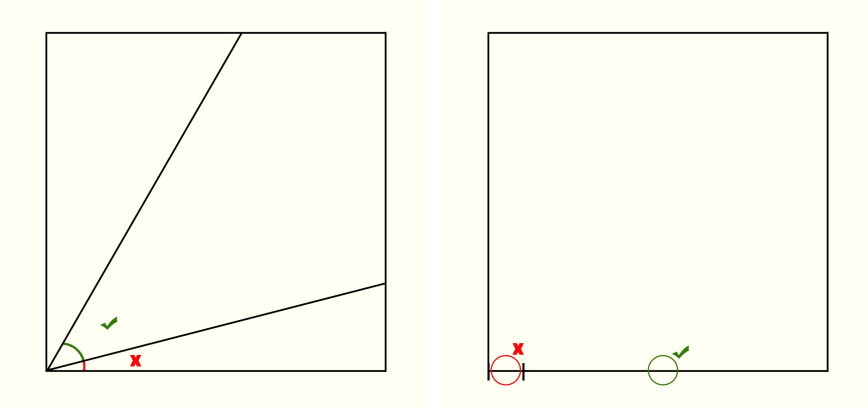
4. Only one division may happen (see figure 2.4d): when multiple markers are available, the first pair to pass the first three criteria described above, is selected and the remaining are dropped.

The process described to model cellular division using the mBPMOL-systems is best explained with the following example. Using an alphabet with numbers (instead of letters) we have $\Sigma = \{1, 2, x, [,], +, -\}$, the initial map or axiom is $\omega = 1, 2, 1, 2$ and the production rules are:

$$\begin{aligned} 1 &\rightarrow 2[-1]x[+1]2 \\ 2 &\rightarrow 1 \end{aligned}$$

The symbols 1 and 2 are the non-terminal tokens and the remaining symbols are called terminal tokens. The rules for the terminal tokens are not listed because they are constants; for instance, a symbol x in a word is copied as x in the rewriting.

The process of cellular division is described as follows:



(a) Small angles are not allowed

(b) Small edges are not allowed

(c) Small areas are not allowed

(d) Only the first two markers are selected

Figure 2.4: Four different criteria for cellular division

1. Initialize with the labelling of each edge of the initial map according to the axiom - please refer to the initial map on top of figure 2.5 - each edge has a specific orientation that makes possible to define the left (-) and right (+) for each edge. The orientation of the edges is counter-clockwise in this example.

2. Each edge is then labelled and divided according to the corresponding rule. In the example, the edges with a label 2 are relabelled with the label 1 and no sub-division follows. The edges labelled 1 are sub-divided into three new and equal segments. Considering now, for example, the edge 1 in the initial map (bottom of map) in figure 2.5. The edge is replaced by three new segments and labelled 2 for the first, the second x and the third 2. The first marker $[-1]$ is placed after the first segment to

its left because the symbol "−" precedes the marker within the brackets. The second marker is placed to the right after the edge with the label "x". Because this marker lays outside the map it is discarded. This process holds true for the remaining edges where the relabelling process is done counter-clock wise.

3. After the rewriting process there are two markers left with the label 1 pointing inside the map. Because the two markers are of the same type and the other criteria are met (no small edges, no small area, no small angles) the cell may be divided by connecting these markers. The result is shown in figure 2.5 "First step".

4. The same process described in the previous items is done sequentially and in parallel for the next cells. This process generates a sequence of maps that models the development stages of cellular division which can be seen as the developmental stages of the structural topology.

Remark 1 *Note that the cell division would stop if, at a certain stage, all edges were labeled with a terminal token. However, stopping only when all letters are terminal tokens could lead to infinite iterations: if x, for example, would be absent from the previous rules. Therefore a maximum number of cell divisions is specified in the beginning of the program and the actual number of developmental stages for the topology generation is optimized in the genetic algorithm.*

Remark 2 *Reference [11] explains the cellular division in living organisms in two stages: a division stage and a dynamic stage. After each cellular division, in nature, the cells deform according to forces acting on their walls. This work does not apply these forces. Nevertheless the forces could be easily modelled for other purposes if interested in the structural topology of cells only. This dynamic stage could be modelled as Picard iterations for the equilibrium of two forces acting on the map vertices: an elastic, spring force on each edge of the topology and a "osmotic" pressure proportional to the inverse of the area of each region. The elasticity constant of each edge and the osmotic pressure constant are modified, and can, therefore, be evolved within the topology, using the state rules.*

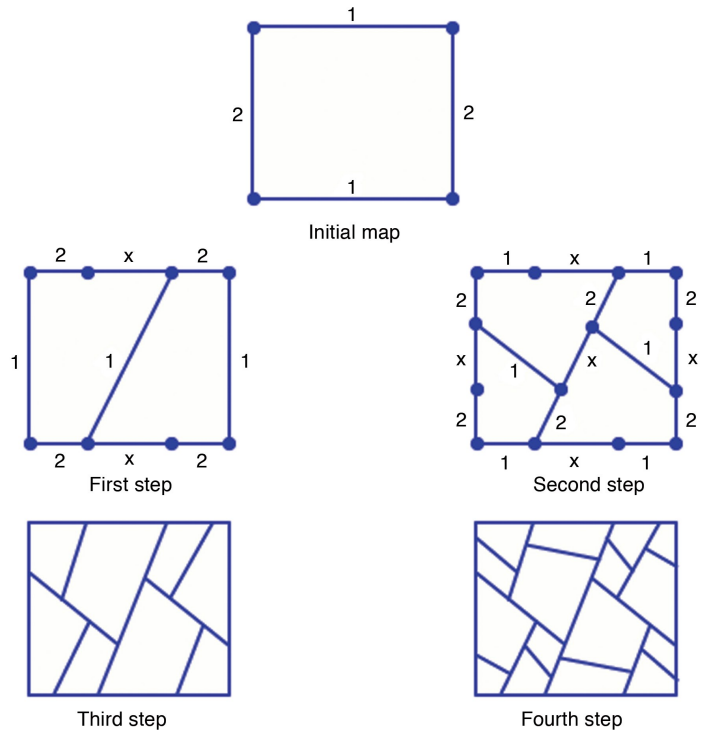


Figure 2.5: Same example of the mBPMOL-systems process as expressed before for the first four steps of the cellular division using numbers instead of letters. The rules for this example are: $1 \rightarrow 2[-1]x[+1]2$

Remark 3 *The original Map L-system deals with convex regions only [36, 37, 11]. In this work, we perform the topology optimization with regions that can be non-convex as well as with multiple connected regions. To extend Map L-system for maps with non-convex regions, we add another criterion for cellular division, where we require the new edge dividing the parent cell to remain completely inside the parent cell.*

This section described in part the development of cellular layers based on the mBPMOL-systems for defining a planar topology. For further information on Map L-system or L-system please refer to [11] and the references in that book. The next section will expand the methodology of mBPMOL-systems to include multiple connected components inside the map and have a general approach for developmental stages of topology generation.

2.2 Map *L*-system and Connected Components

Map *L*-systems are suitable to mimic the cellular development when no physical constraints exist inside the cells. In a topological sense it is as there are no holes or interior faces that cannot be crossed. Figure 2.6 shows a typical Map *L*-system division using the a simple example. Considering the alphabet $\Sigma = \{1, 2, x, [,], +, -\}$ and the following initial map defined by the axiom:

$$\omega : 1212 \tag{2.2}$$

and the production rules

$$r_1 : 1 \rightarrow 2[-1][+1]2 \tag{2.3}$$

$$r_2 : 2 \rightarrow 1 \tag{2.4}$$

This configuration will produce the map sequence shown in figure 2.6. Adding a connected

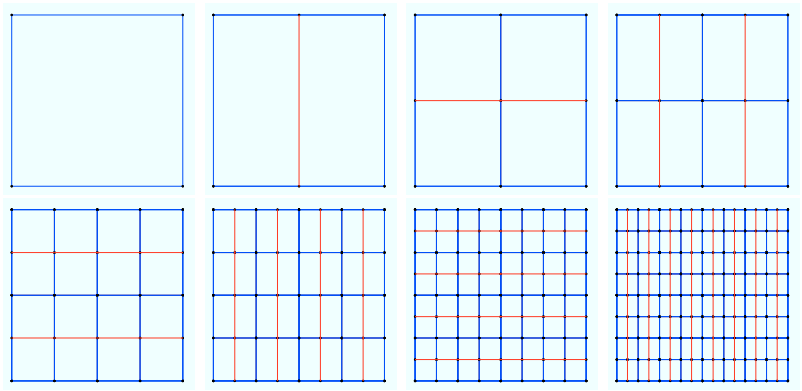


Figure 2.6: First 5 steps in the cellular division process using the Map *L*-system with **no** connected components inside the initial map. Map *L*-system iterations using the axiom 2.2

component to the map the cellular division process will behave differently as shown in figure 2.7. The most important aspect of this work is the development of a new method to create proper cellular divisions when the maps contain multiple connected components while keeping all the same principles of the Map *L*-system explained in section 2.1.

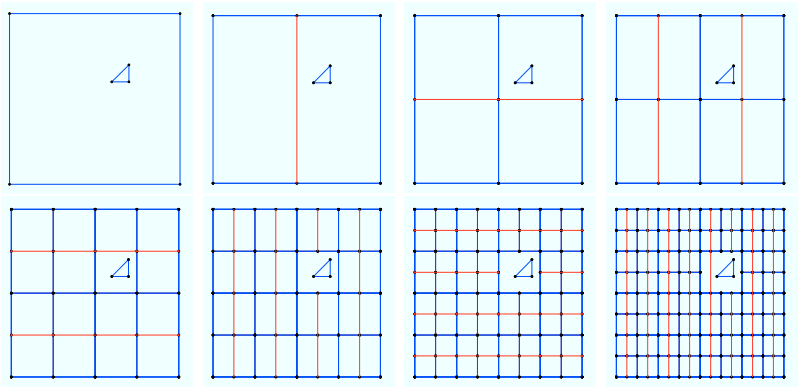


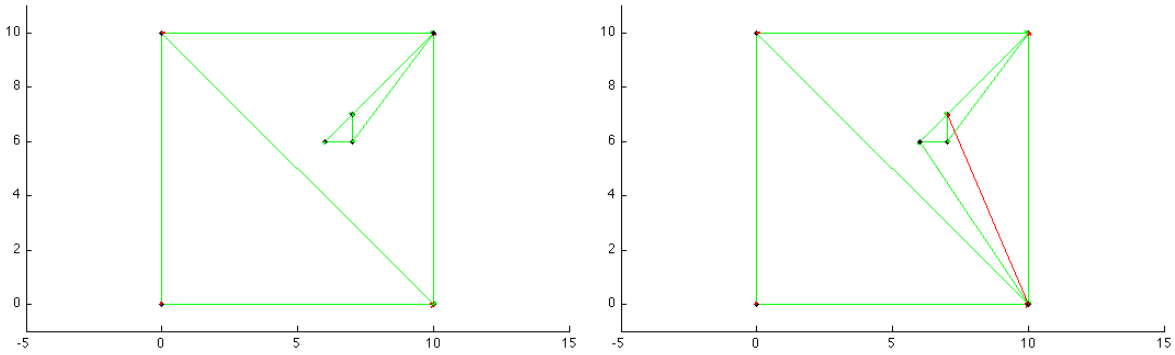
Figure 2.7: First 5 steps in the cellular division process using the Map L-systemwith **one** connected component inside the initial map. Map L-systemiterations using the axiom 2.2

The search for a systematic and general method to divide maps with connected components was the main driver for this research and also the most challenging and time consuming because the Map L-system methodology is not formulated to be used with multiple connected components inside of maps. The two main problems are 1) divide the cells without intersecting the connected components and 2) properly defining the new cells after the division has been done. The process is more challenging when a connection is made between different connected components. Mathematical tools and Graph Theory elements must be used to successfully solve this problem. The next section will set the framework to develop the process of decision about possible intersections with other connected components during the cellular division.

2.2.1 Intersection of Connected Components

The Map L-system methodology does not take into account the fact that there can be other connected components inside the initial map, so it does not take into account the fact that improper intersections done with compatible markers may happen. An example of an improper cellular division is shown in figure 2.8.

To have a successful division the process must not only be based on the compatibility criterion of the markers (as shown previously in section 2.1) but must add another criterion that is based on the fact that a new possible division must not intersect any edges other



(a) cellular divisions done properly (b) wrong cellular division

Figure 2.8: Problem with the intersection of a connected component

than the ones that it is being connected to.

This added criterion implies that all the edges of the connected components inside the cell being divided must be tested for intersection. Before explaining the intersection determination procedure some brief definitions will be introduced using figure 2.9 as a reference: 1) The angle θ is defined between the $x - axis$ and the new line segment L; 2) the angle θ_1 is defined between the $x - axis$ and the line segment $L_{z_1} = [p_1, z_1]$; 3) the angle θ_2 is defined between the $x - axis$ and the line segment $L_{z_2} = [p_1, z_2]$. L_I is the line segment defined between $p1$ and the intersection point I of the edge to be tested. Figure 2.9 exemplifies the angles and line segments for an arbitrary case.

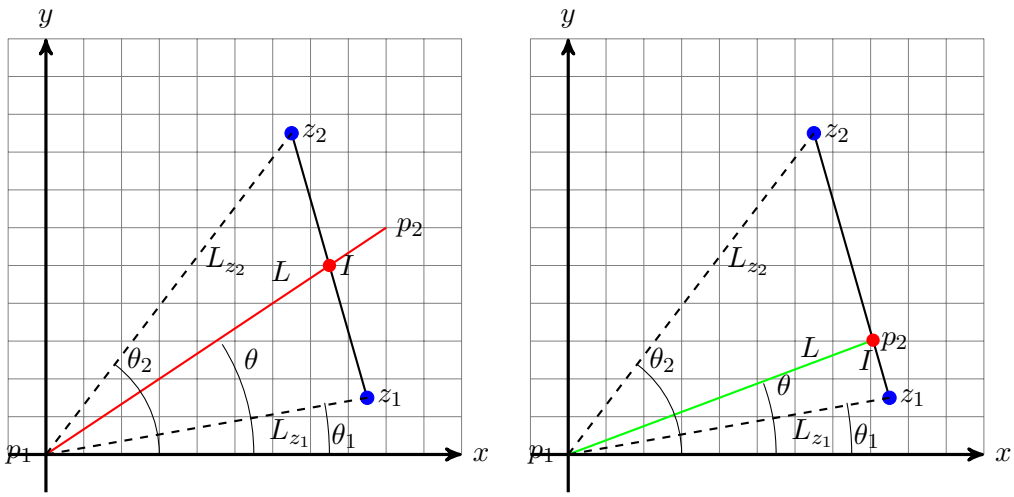
The equation 2.5 computes the length of the line segment L_I which is an important information for the intersection criterion.

$$|L_I| = r_{L_I}(z_1, z_2, \theta) = real\{\frac{z_1 - \overline{z_1} * \alpha}{e^{i\theta} - \alpha * e^{-i\theta}}\} \tag{2.5}$$

$$\text{where } \alpha = \frac{z_2 - z_1}{\overline{z_2} - \overline{z_1}} \tag{2.6}$$

The decision about the intersection is given after the testing of the two following rules is done:

- the angle θ must be defined in between the angles θ_1 and θ_2, so $\theta_1 \leq \theta \leq \theta_2$;
- the length of L must be longer than the length of the line segment L_I defined between



(a) intersection exists because $|L_I| \geq |L|$ and $\theta_1 \leq \theta \leq \theta_2$

(b) intersection possible because $|L_I| \geq |L|$ even if $\theta_1 \leq \theta \leq \theta_2$

Figure 2.9: Intersection criterion

p_1 and I, so $|L| > |L_I|$

If, and only if, the two rules hold true then there is an intersection which is not acceptable, otherwise the edge can be traced. Figure 2.9a shows one example of an edge intersection that is not allowed while figure 2.9b shows one acceptable edge connection.

Applying this criterion to every edge that belongs to the cell being tested (including all the connected components of that cell) will determine if the new division is possible. To illustrate the method of eliminating intersections within a topology with multiple connected components, a test was made with two connected components on the topology used for this work. The sequence of cellular division is shown in the following figure 2.10; it illustrates the method of properly dividing cells without intersecting any connected components.

The next section will set the framework with some Graph Theory elements that are necessary to solve the problem of properly defining the new cells after the division has been made.

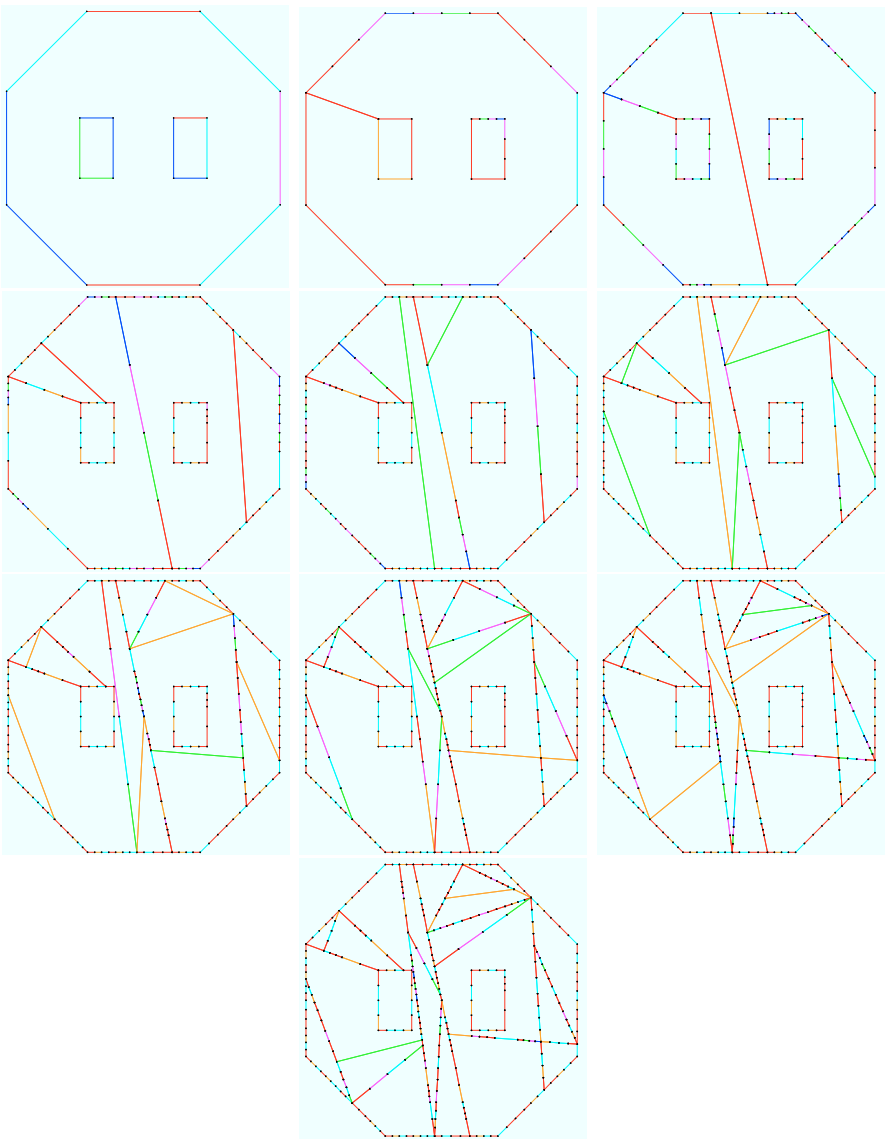


Figure 2.10: Sequence of cellular division without intersecting the two connected components

2.2.2 Elements of Graph Theory

Graph theory will help defining some principles used in the methodology of connecting maps with multiple connected components. A very brief overview of Graph Theory and some definitions are presented to set the concepts uses in the following sections.

Graphs are mathematical structures that are used to model the relations between objects of a specific group. In general a graph $G =< V, E >$ represents the connections between

groups of vertices V using a set of edges E that connect the vertices pairwise. One edge can only connect two vertices a and b, where b is said to be adjacent of a.

Definition # 1 *Two vertices a and b are called connected if an undirected graph G contains a path from a to b. Otherwise, they are called disconnected.*

Graphs are categorized by the way they are connected. In an *undirected graph* the edges do not have information about their orientation, meaning that they are 2-multisets of vertices a, b. In a *directed graph*, also called digraph, the vertices are connected pairwise using directed edges. Each edge has the information of the direction from a to b and is specified as an ordered pair $< a, b >$ meaning "a goes to b", also refered as $a \rightarrow b$.

A *simple graph* is an undirected graph that has no more than one edge connecting two different vertices. In this way the edges form a single set where every edge is defined as pair of distinct vertices. On the other hand a complex graph, also known as a multigraph or pseudo-graph, may have multiple edges connecting the same vertices (also called parallel edges). Using this formulation a multigraph can be an undirected or directed graph [39].

To define new cells inside a map with multiple connected components it is necessary to use a multigraph and undirected graphs mixed with directed graphs. This helps the purpose of connecting different vertices using the n connected components inside the graphs (or maps). The reason for using the mix of both graphs is given by definition 2 adding the fact that the initial map has to be properly directed before the division starts.

Definition # 2 *an undirected graph can be represented by a directed graph if every undirected edge a,b is represented by two directed edges $< a, b >$ and $< b, a >$ [40].*

Using the formulation presented it is still necessary to add a new definition about the *the half-edges*. Figure 2.11 is useful to understand this concept of the half-edges. The concept works well for planar maps, polyhedra and other two-dimensional surfaces that are orientable and are embedded in an arbitrary dimension. The concept is simply constructed

as taking one edge and decomposing it in two separate ones (called the half edges) with opposite orientations.

Each face of the graph is to be identified according to a specific orientation. One incident face and one incident vertex can then be stored in each half-edge. Figure 2.11 shows every edge of an arbitrary initial map used as an example and the corresponding half-edges properly set. The faces are defined as a "walking path along the wall" according to the half-edges orientation. In this way the faces of the graph are not lost in further divisions - care must be taken to always properly orient the half edges in coherence with the previous half edges. As an example, let's say that edge #1 is defined by the pair $< 1, 2 >$ (it goes from 1 to 2) then if a person walks from 1 to 2 the half edge is along that same direction on the left side of the edge but flows in the other direction on the right side of that same edge.

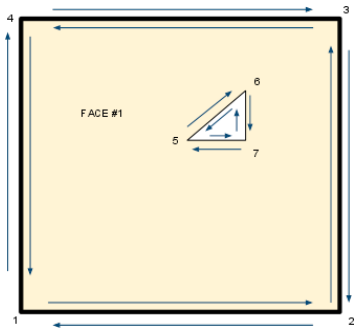


Figure 2.11: Map half-edges orientation

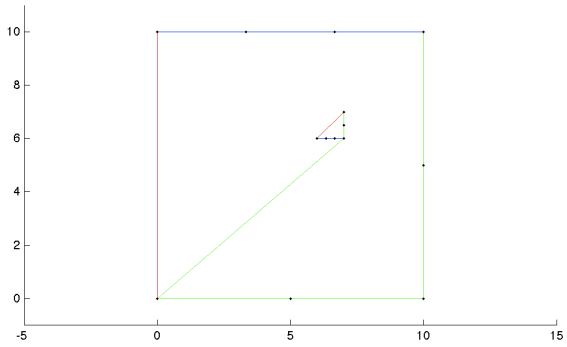
These elements of planar graphs will help us on the next section to divide the maps for creating the topology in a recursive way.

2.2.3 Connection of Connected Components in a Graph

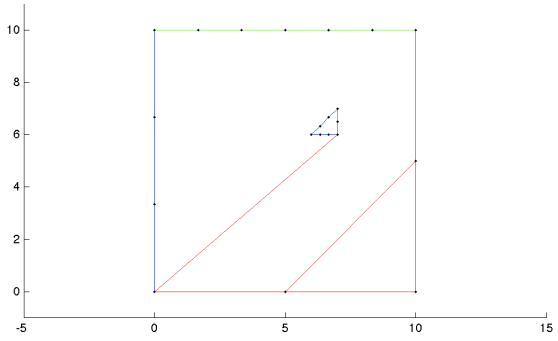
The connection of multiple connected components that define new regions inside a graph has no trivial solution. Nevertheless using a proper formulation that was defined in section 2.2.2 it eventually becomes simple and systematic to solve the problem of defining regions.

There are two different types of connections that can be made inside a graph: 1) connecting the same connected component and 2) connecting different connected components. This is exemplified in figure 2.12 where the picture on the left shows the connection between two different connected components which then become the same connected component. The picture on the right the connection is made in the same connected component creating two

separate connected regions, so two new cells are created.



(a) connection between two different connected components → becomes the same connected component, no cell is created



(b) connection in the same connected component → becomes two separate connected components, two new cells are created

Figure 2.12: Possible connections inside a graph

The procedure to connect two different connected components may create multi-branches in the graph, to successfully decide the right branch that must be selected a rigorous methodology must be used. The following description illustrates one example of the division method:

1. The initial map may have any orientation: in the case of figure 2.13 the orientation on the left of the edge is counterclockwise. The external part of the cell (or face) is defined by the edges [1, 2, 3, 4] and the connected component has the edges [5, 6, 7]. These edges and corresponding half-edges are oriented in the way presented in the figure. It is of extreme importance that the initial map be well oriented because otherwise the method to connect multiple connected components will fail.

2. A new connection is made and is represented by the edge [1 5]. It must keep the same orientation based on the initial map: counter-clockwise on the left and clockwise on the right of the edge.

3. Another edge is created [2 5] and the orientation is assigned coherently as described before. This edge divided the cell and created a new face. The criterion to determine the new division is to look for the vertices that are intersected from the previous face: in the case of this edge it is exactly at the vertices that it connects: 2 and 5. To define the new

Initial Map with proper orientation

- Face #1 vertices: [1 2 3 4 1; 5 6 7 5]
- Face #1 edges: [1 2 3 4; 5 6 7]
- Face #1 edges dir: [1 1 1 1; 1 1 1]

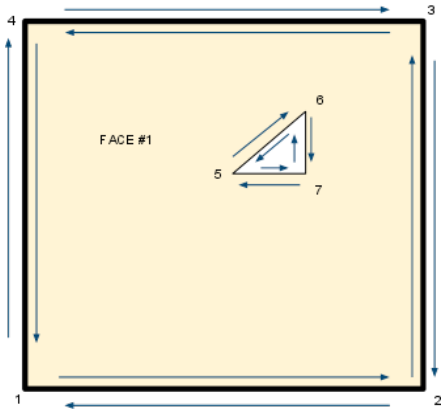


Figure 2.13: Step 1 :- Initial Map Orientation

New connection between two different connected components

- Face #1 verts: [1 2 3 4 1 5 6 7 5 1]
- Face #1 edges: [1 2 3 4 **8** 5 6 7 **8**]
- Face #1 edges dir: [1 1 1 1 **1** 1 1 1 **2**]

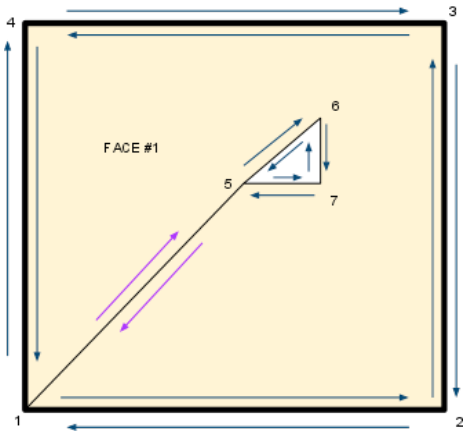


Figure 2.14: Step 2 :- New edge created between two different connected components, the connected components are merged

cells one starts from the new edge created on the left side (counter-clockwise direction) → [2 5]. Using the vertices of face #1: [1 2 3 4 1 **5 6** 7 **5 1**] (the "cut" is done on the edges that have vertex 5: that is [5 6] and [5 1]). At this stage two edges have been selected to connect. To choose which of these two edges to connect, the smaller open angle criterion is used. This means ([5 1] has a smaller angle than [5 6] with respect to [2 5]). So the new face will have this vertices sequence: [2 5 1 2] and the other face will have: [5 2 3 4 1 5 6 7 5 2]. This defines with generality the new regions, or cells, with the proper orientations.

The previous two sections formulated a new methodology to not intersect the connected components and to define the new cells properly when connecting different connected com-

New connection within the same connected component

- Face #1 verts: [5 2 3 4 1 5 6 7 5 2]
- Face #1 edges: [9 2 3 4 8 5 6 7 9]
- Face #1 edges dir: [1 1 1 1 1 1 1 1 2]
- Face #2 verts: [2 5 1 2]
- Face #2 edges: [9 8 1]
- Face #2 edges dir: [1 2 1]

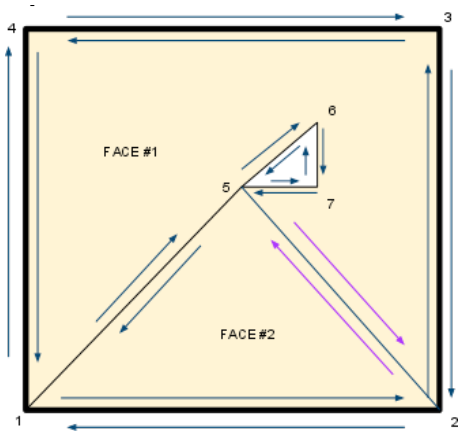


Figure 2.15: Step 3 :- New edge created within the same connected component, two new cells are formed

ponents inside a map. This methodology is implemented in the software for automatic topology optimization. The next section will explain how to select the best topology, which corresponds to a structural and physically equivalent object.

2.3 Single Objective Optimization

Optimization problems often have a single candidate metric for defining optimality. Optimization is searching for and choosing the best element from a set of evaluated candidates. Very often this is done by minimizing or maximizing a real function that is well defined. A well defined function must transport every element of its own domain to a paired element of its co-domain. Nevertheless, most of the problems to be solved are not modelled by well-defined functions. That is why a generalization of the optimization theory encompasses a large domain of techniques and applied formulations to find the best solution available in the solution space. Because this work has a single objective optimization - minimize the mass of a given structure - the choice of the optimization search algorithm is based on the Genetic Algorithm for the search of the solution space of the fitness of a given structure.

The optimization for this type of minimization problems can be described as follows: *let a function* $f(x) : A \rightarrow \mathbf{R}$ *where* x *is the vector of design variables ("the genes" for the Genetic Algorithm),* A *is the set of* x*'s and* $\mathbf{R}$ *represents the set of real values of the fitness*

function for the topology generated after the x's. The minimal value is given by the search for an element x_0 in A such that $f(x_0) \leq f(x)$ $\forall x$ in A.

The elements of A, the candidate solution, are a subset of the Euclidean space $\mathbf{R^n}$ where n is the dimension of the gene that is produced on the Genetic Algorithm.

2.3.1 Genetic Algorithm

This section will present the Genetic Algorithm as a search tool for topology optimization. This process encodes the topology grammar presented before (see section 2.1) and searches for the individual with the best fitness.

A Genetic Algorithm is a search heuristic that generates an insightful and more useful solution at each iteration. The Genetic Algorithms are keen to find the optimal solution of problems that are not well defined or difficult to model like discontinuous sets, highly non-linear functions, stochastic or even with undefined variables. The Genetic Algorithm is used to find solutions for problems that are difficult to solve with traditional optimization algorithms.

The Genetic Algorithm is a biological metaphor from genetics applied to computer science. The algorithm starts with a population of strings (also called chromosomes or the genotype of the genome) each one carrying a genotypic content. These encode the candidate solutions (also called individuals or phenotypes). The genotype has the primitive parameters (the genes) that determine the individuals' layout and topology in the context of the Map L-system methodology. It also refers to the geometric and physical parameters for the application object. Table 2.1 shows one example of the de-codification process.

It is the phenotype that translates the raw information in the genotype to the actual structural model in physical terms (see table 2.1). The evaluation of the phenotype is carried over using the finite element analysis done on the structural model (the phenotype). The final step of the evaluation is to compute the fitness of the individual. The genetic algorithm then uses the all the fitness values for the current population and selects the best individuals and it then advances to the next generation by selection, mutation and crossover

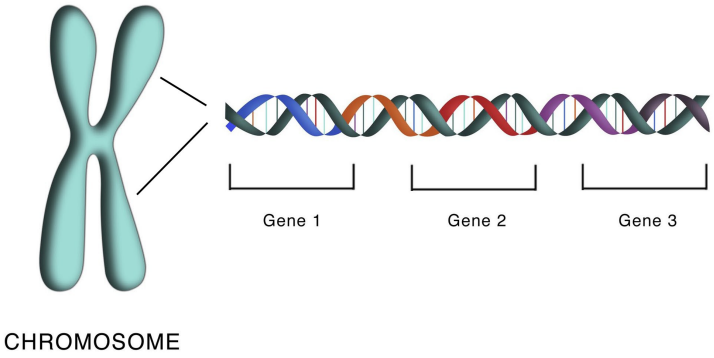


Figure 2.16: Example of a chromosome and its genes

	Gene 1	**Gene 2**	**Gene 3**
genotype	[19.3, . . . , 53.6] Axiom	[53.6, . . . , 17.8] Rules	[39.9, . . . ,46.7] Topology input
phenotype	con. comp. 1: 2 1 2 1 2 1 4 2 con. comp. 2: 4 5 3 4	1 → [+5];[-3];[+6];5;5 2 → [+1];[-6];1;2;[+5] 3 → [-5];6;[-2];[-3];[-4] 4 → [-4];2;1;[-2];6 5 → 3;3;[-2];5;6 6 → 6;5;4;1;[-5]	number of iterations = 6 global shell thickness = 1.8 [mm] subsystem shell thickness = 0.5 [mm] external beam feature size = 9.4 [mm] internal beam feature size = 3.1 [mm] subsystem position x = -57.3 [mm] subsystem position y = -46.8 [mm] subsystem angle = 168.1 [deg]

Table 2.1: Example of a chromosome and its de-codification

operators.

Genome definition

The genome is the entirety of an individual's hereditary information which includes the genes that define it. In this work the genome is composed of three main classes of genes. These three genes affect directly the topology defined by the Map L-system in the variant of the mBPOML-system using a fixed alphabet Σ (for more information about the method refer to section 2.1). Table 2.1 has summarized the following description:

- The ***first gene*** has the information to define the axiom word ω that affects the original map division and dynamics of the developmental stages. This axiom is encoded using

the same entries of Σ.

- The ***second gene*** has the information defining the edge production rules P. This gene may be seen as the regulator for the complex process of cellular division by parallel interpretation of the DNA. Every production rule is encoded according to a master rule which has the form $Y \rightarrow X_1 X_2 \ldots X_{n-1} X_n$ where Y is a non-terminal token and X_i represent terminal, non-terminal or special tokens. The number of the X slots (n), also described as the rule length, is the same for all rules and is predefined by the user as an input parameter. If the slot is a terminal token then $X_i = A_i$, if the slot is a marker then it includes bracket defined as $X_i = [\pm A_i]$, finally if the slot is a non-terminal token it is then represented by $X_i = x$

- The ***third gene*** has the information about the geometry and physical properties of the object. In this application some of these parameters are the shell or the beam thickness.

In summary, the genome is a combination of three genes that compose the axiom, the production rules and the geometric and physical parameters to define the structure to be analysed.

In the next chapter we define the steps for creating the structural model from the topology generated after the methods presented in this chapter.

CHAPTER 3
STRUCTURAL AND FINITE ELEMENT MODELS

The Map *L*-system generates a planar topology that has no canonical structural meaning attached to it - this topology is called the structural skeleton. To create the structural model with a physical equivalence we select the following basic structural elements:

- Skin shell panel, for uniform support of the entire panel
- Euler Beams, attached to the shell

The panel is a structural element that in this work is modelled as an isotropic shell with Euler beams attached to the shell. The topology generated from the Map *L*-system corresponds to the beams attached to the panel. The shell is separated in two different sections each with two different thickness parameters, one for the subsystem and the other for the remaining part of the panel. Figure 3.1 shows the section view to illustrate the different thickness for the shells and the beams.

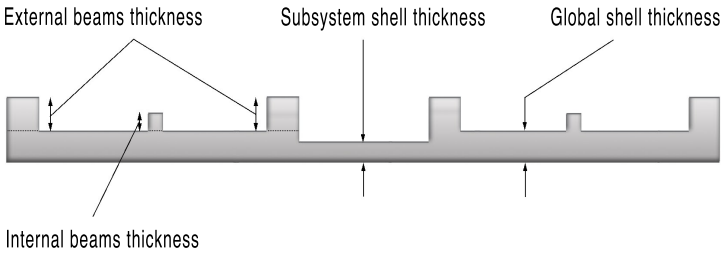
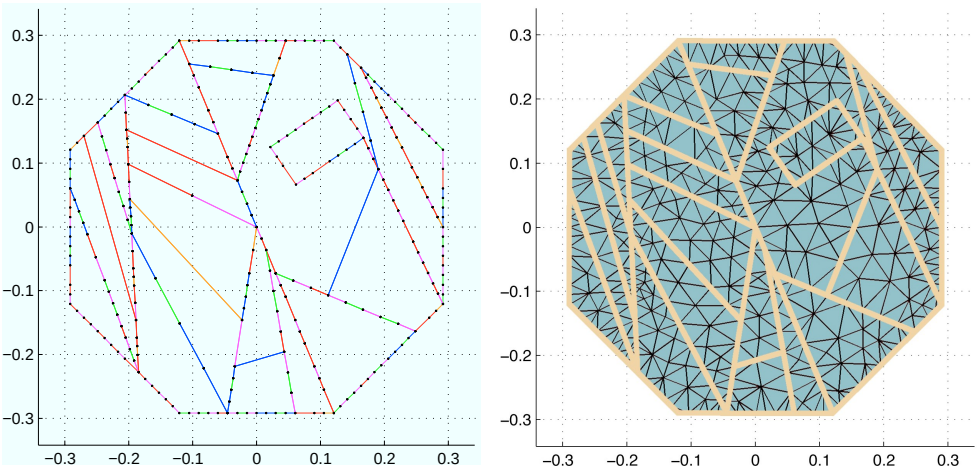


Figure 3.1: Section view of structural panel

The structural elements are combined into a complete structural model for the satellite panel which is simulated using the Finite Element Method [41, 42, 43]. The resulting Finite Element Method mesh is illustrated in figure 3.2 and is explained in the description for the satellite panel that follows.



(a) Skeleton (or topology) of the panel structure

(b) Generated mesh from the panel structural skeleton

Figure 3.2: Finite Element Method applied to the topology generated from the Map *L*-system

There are various implementations of the Finite Element Method available in the industry and academia. This work was developed using the Finite Element Method toolbox COMSOL Multiphysics™ (formerly FEMLAB). This is a Finite Element Method analysis software that is capable of coupling different physical phenomena making it appropriate to simulate the shell and the beams structural behaviour within the same analysis. This software interfaces with MATLAB® and its toolboxes giving it a wide variety of programming, pre and post processing capabilities.

The procedure to generate the Finite Element Method model for the satellite panel is essentially based on three steps:

1. The **mesh generation**: this is to generate a two-dimensional mesh that conforms to the planar topology. Figure 3.2b shows one example of a bidimensional mesh. The default method for generating free triangle meshes in COMSOL Multiphysics™ is based on an advancing front algorithm [44][41] resulting of a mesh object derived from the analysed geometry. This also returns a set of two-dimensional grid points, the adjacency matrices for the nodes, the edges and faces of the discretized domain. The command to generate the mesh in COMSOL Multiphysics™ is meshinit(fem)".

2. The **multiphysics setup**: after the mesh has been generated it is necessary to set the physical properties like the material properties, the physical loads, the thickness, the beams size, etc. This is done using internal COMSOL Multiphysics™ commands to set each specific physical problem.

3. The **Finite Element Method analysis**: after creating the structural model, the analysis can be performed using other COMSOL Multiphysics™ commands to solve the Partial Differential Equations to obtain the desired information on deformation and stress of the structure.

Figure 3.3 shows one of the simulation results for the static structural mechanics problem applied to the satellite panel.

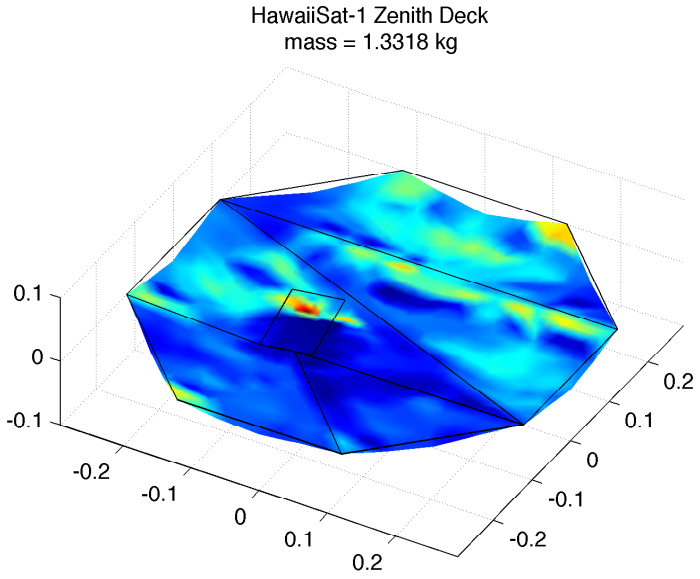


Figure 3.3: Example of von Mises stress distribution for the satellite structural panel

CHAPTER 4
SOFTWARE DEVELOPMENT

This chapter will describe the software development for the optimization of structures in this work. The software development was the main driver of the research done for this thesis and the most time consuming process. Its development brought interesting insight into the process of automatic topology generation moving from traditional Map L-system algorithmic processes to graph theory processes. This software is divided in three main sections: 1) the Automatic Topology Generation 4.1, 2) the Structural Analysis 4.2 and 3) the Search for the Optimal Structure 4.3. The previous chapters described the foundations for the software integration, starting with the Map L-system method and ending with the search for the best individuals using the Genetic Algorithm. The following sections will briefly describe the three main software processes.

4.1 Automatic Topology Generation

The Automatic Topology Generation process essentially uses the Map L-system method with n-connected components to generate iteratively a new topology. The methods that support this part of the software are described in sections 2.1, 2.2 and 2.2.1.

The software process starts with the setup of the configuration parameters. Some of the most important configuration parameters are the number of edges of the initial map ($numEdges_0$), the number of rules ($numRules$) to be utilized, the length of the rules ($lenRules$) and the number of control variables of the structure ($numControlVariables$). Some of the other configuration parameters are described in table 4.1.

As described in section 2.3.1 the chromosome contains all the genotype information. This information is created as a random array by the Genetic Algorithm. This array contains the three genes also described in that section: the axiom, the rules and the topological and physical parameters of the structure. The length of this array is given by the sum of: the length of the axiom ($numEdges_0$), the length of all the entries for the rules ($numRules \times$

Configuration Parameters	Description
directory paths	initial paths for the software directories
automatic flow variables	variables for normal software operation with the genetic algorithm (draw [on, off]; debug mode [on, off]; create directories [on, off])
plot parameters	plot size, docked [on, off], etc.
colours	colours for edges identification (edges with same label will have the same color), edge of type 1 is always red, type 2 is green, type 3 is blue, the remaining colours are given randomly
tolerance	if the number computed is too small then trim it to the configured tolerance
length and number of rules	the number of rules gives the number of different labels possible as well as the number of rules to be applied to the map, the length of rules is the number of terminal, non-terminal or special tokens in the rule (see sub section 2.3.1).
log files	create the names and folders to store the log files during the run

Table 4.1: Table with configuration parameters

lenRules) and the number of control variables (*numControlVariables*). The chromosome length is a critical information for the production of the chromosome and for a coherent phenotype transformation of the genes.

$$\begin{aligned} \text{Chromosome Length} = & \; numEdges_0 \\ & + (numRules \times lenRules) \\ & + numControlVariables \end{aligned} \tag{4.1}$$

Other parameters needing to be set at the initial stage of the software run are the physical parameters and constraints. These are defined as the input parameters like the material properties, dimensions, physical loads, etc. Table 4.2 has a brief description of the input parameters.

After setting the configuration and input parameters and having generated the Chromosome, the topology generation process may begin. The first step is to read and separate

Input Parameters	Description
physical dimensions	area for planar structure, area for subsystems, length beams, etc.
material properties	density, Poisson's ratio, Young's modulus, etc.
physical loads	axial loads
physical constraints	minimum edge length, minimum cell area, etc.

Table 4.2: Table with input parameters

the chromosome and translate it to the corresponding genes and their phenotype properties. The genes are given in a random array and their conversion to meaningful terms is done inside the process "Translate Genes". This process starts by normalizing the genes by setting every number inside the range $[0, 100]$. If a gene as a number outside this domain it will then normalize it. Table 4.3 shows one example of this normalization.

	Gene 1
raw gene	[**132.4**, **158.9**, 62.2, **105.7**, 33.1, **120.4**, 52.6, **130.8**, **137.8**, **149.6**, 90.1, 16.8]
normalized gene	[**32.4**, **58.9**, 62.2, **5.7**, 33.1, **20.4**, 52.6, **30.8**, **37.8**, **49.6**, 90.1, 16.8]
	Gene 2
raw gene	[45.8, **182.7**, 30.5, **165.2**, **107.7**, **199.2**, 15.6, 88.5, 21.3, **192.4**, 0.9, **155.0**, **163.5**, **173.7**, 16.9, 80.0, 52.0, **160.0**, 86.3, **182.1**, 36.4, 52.8, 29.1, 27.2, **173.9**, **115.9**, **110.0**, 29.0, **170.6**, **124.4**]
normalized gene	[45.8, **82.7**, 30.5, **65.2**, **7.7**, **99.2**, 15.6, 88.5, 21.3, **92.4**, 0.9, **55.0**, **63.5**, **73.7**, 16.9, 80.0, 52.0, **60.0**, 86.3, **82.1**, 36.4, 52.8, 29.1, 27.2, **73.9**, **15.9**, **10.0**, 29.0, **70.6**, **24.4**]
	Gene 3
raw gene	[70.2, **102.6**, 80.4, 15.2, 48.0, 24.7, 36.8, 48.0]
normalized gene	[70.2, **2.6**, 80.4, 15.2, 48.0, 24.7, 36.8, 48.0]

Table 4.3: Table with the genes and the normalized co-gene

After the normalization is complete the process of converting the genes starts by generating the axiom and the production rules for the Map L-system. The axiom is the phenotype expression of the first gene so it reads the first entries of the chromosome array. This gene is decoded by dividing the maximum range of the normalized domain (in this case it is 100) by the number of rules set in the configuration parameters. For the selected example the axiom then becomes as expressed in table 4.4b. It is important to note that the axiom is split for each of the connected components - two for this particular case. The first connected component is the satellite panel and the second connected component is the subsystem.

The production rules conversion follows the same procedure as indicated for the axiom.

Axiom Mapping				
0	...	16.6(6)	→	1
16.6(6)	...	33.3(3)	→	2
33.3(3)	...	50.0	→	3
50.0	...	66.6(6)	→	4
66.6(6)	...	83.3(3)	→	5
83.3(3)	...	100.0	→	6

(a) Conversion table for the Axiom

Normalized Gene	[32.4, 58.9, 62.2, 5.7, 33.1, 20.4, 52.6, 30.8;	37.8, 49.6, 90.1, 16.8]
gene for connected component #1	[32.4, 58.9, 62.2, 5.7, 33.1, 20.4, 52.6, 30.8]	
gene for connected component #2		[37.8, 49.6, 90.1, 16.8]
	↓	↓
axiom for connected component #1	[2 4 4 1 2 2 4 2]	
axiom for connected component #2		[3 3 6 2]

(b) Genotype and Phenotype for the axiom

Table 4.4: Translation of the first gene to the axiom

The productions rules are the phenotype expression of the second gene. This gene is decoded by mapping the raw data into three categories, the negative token, the positive token and the non-terminal token. All may range from 1 to the number of existing rules. This is done by subdividing equally the mapped domain into the number of existing rules. As an example, the number 45.8 (the first number of the example gene) is to be mapped into the first production rule sequence, since the first five digits belong to the first production rule. Because this number is between the range [25,50] it is a positive token. This range must be divided by the total number of rules in equal parts, each one corresponding to a number from 1 to $numRules$. The number 45.8 corresponds to the label 5 in this mapping. Combining the fact that this is a positive terminal token with a label 5 then the rule entry becomes: [+5]. Table 4.5 expands this example for the complete gene.

Finally, the last and third gene is transformed in a slightly different manner than the first two. The third gene has the information about the topology and the physical structure. The only requirement is that these properties are within a boundary. As an example, in the case of the *global shell* thickness it is imperative that it is bounded by a minimum and maximum amount physically plausible. The conversion of this gene is then taken with the

Rules Mapping					
0	...	25	→	[−1 ... 6]	(negative token)
25	...	50	→	[+1 ... 6]	(positive token)
50	...	100	→	1 ... 6	(non terminal token)

(a) Conversion table for the Production Rules

Normalized Gene

[45.8, 182.7, 30.5, 165.2, 107.7; 199.2, 15.6, 88.5, 21.3, 192.4; 0.9, 155.0, 163.5, 173.7, 16.9; 80.0, 52.0, 160.0, 86.3, 182.1; 36.4, 52.8, 29.1, 27.2, 173.9; 115.9, 110.0, 29.0, 170.6, 124.4]

↓

Production Rules

1 → [+5];4;[+2];2;[-2]
2 → 6;[-4];5;[-6];6
3 → [-1];1;2;3;[-5]
4 → 4;1;2;5;4
5 → [+3];1;[+1];[+1];3
6 → [-4];[-3];[+1];3;[-6]

(b) Genotype and Phenotype for the Production Rules

Table 4.5: Translation of the second gene to the production rules

expression 4.2 where min and max define the desired physical boundaries and raw is the normalized entry of the gene that corresponds to the property to be converted:

$$\text{physical property value} = min + \frac{(max - min) \times raw}{100} \tag{4.2}$$

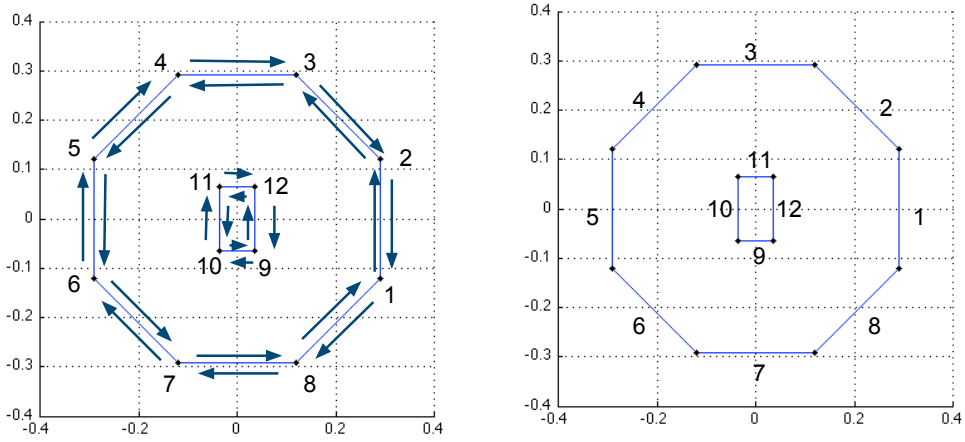
From the example that is being presented in this chapter (benchmark example) the conversion of the third gene is given in table 4.6. This ends the process for the conversion of the genotype into the phenotype.

The next process in the software is the construction of the initial map given the configuration and topology parameters for the desired object to be analysed. The benchmark example in this case is given for the structural panel shown in figure 5.2. The topology for this panel is shown in figure 4.1.

The initial map is generated using the information of the edges: the edges unique

genotype/gene 3		phenotype input	range
70.2 →	8	number of iterations for topology division	3 . . . 10
2.6 →	0.4 mm	global shell thickness	0.1 . . . 12.5 mm
80.4 →	10.1 mm	subsystem shell thickness	0.1 . . . 12.5 mm
15.2 →	1.6 mm	external beam feature size	0.1 . . . 10.0 mm
48.0 →	4.9 mm	internal beam feature size	0.1 . . . 10.0 mm
24.7 →	-70.9 mm	subsystem position x	-140 . . . 140 mm
36.8 →	-37.0 mm	subsystem position y	-140 . . . 140 mm
48.0→	172.7 deg	subsystem angle	0.0 . . . 360.0 deg

Table 4.6: Example of the genes de-codification process for the benchmark gene



(a) initial map with vertices numbered and orientation of the edges

(b) initial map with edges numbered

Figure 4.1: Initial map definition

identification number "edgeID" and the edges direction "edgeDir". This information is stored in a structure "map" which is a structure that has other nested structures in itself:

1. the *cells structure* which contains the information of all connected components
2. the *edges structure* contains the ordered vertices that form the edges
3. the *vertices structure* contains the coordinates for every vertex used in the map
4. the *faces structure* which contains the information of the two faces that each edge touches (in coherent order with the edge orientation)

For the initial map in this benchmark example there exists one cell only with two connected components. The information is then stored in the following way:

- map.cell(1).connectedComponent(1).edgesIDs = [1 2 3 4 5 6 7 8]

- map.cell(1).connectedComponent(1).edgesDir = [1 1 1 1 1 1 1 1]

The edges are stored in the edges structure - map.edges, the vertices are stored in map.vertices and the faces are stored in the map.faces structure. This assignment is expressed in table 4.7. The process for the initial map returns a clean map structure with all the relevant information.

Edge	vertices	faces
1	[1 2]	[1 0]
2	[2 3]	[1 0]
3	[3 4]	[1 0]
4	[4 5]	[1 0]
5	[5 6]	[1 0]
6	[6 7]	[1 0]
7	[7 8]	[1 0]
8	[8 1]	[1 0]
9	[9 10]	[1 -1]
10	[10 11]	[1 -1]
11	[11 12]	[1 -1]
12	[12 9]	[1 -1]

(a) Edges connections

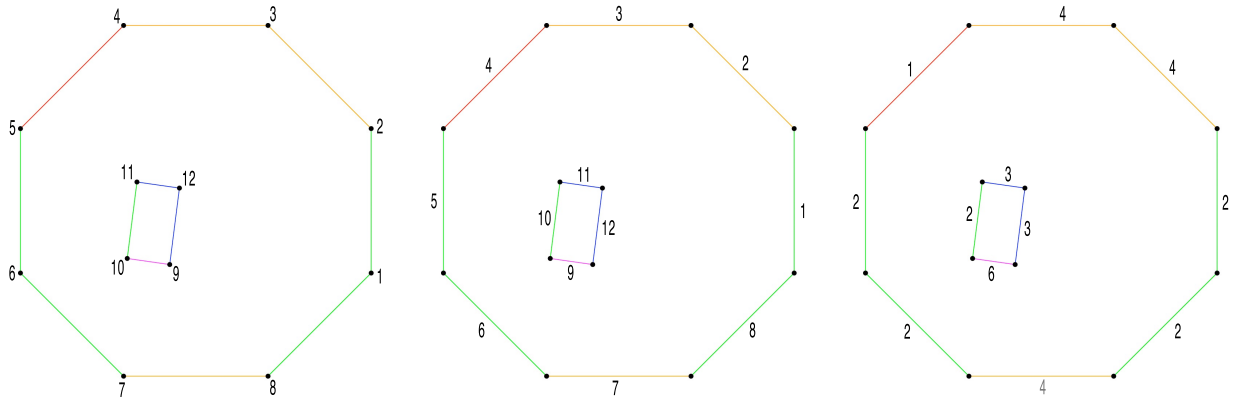
Vertex	X coord. [mm]	Y coord. [mm]
1	291.52	-120.75
2	291.52	120.75
3	120.75	291.52
4	-120.75	291.52
5	-291.52	120.75
6	-291.52	-120.75
7	-120.75	-291.52
8	120.75	-291.52
9	35.68	-64.18
10	-35.68	-64.18
11	-35.68	64.18
12	35.68	64.18

(b) Coordinates of the vertices for the initial map

Table 4.7: Initial map definition

With the initial map properly defined it is then possible to apply the axiom to this map which is done on the “Map Labelling” process. This returns the map structure with a new nested structure with the assigned labels for each edge. The assignment is revealed in figure 4.2.

Given the complete map structure with all the relevant information, the process of cellular division can start. This is done inside a “for-loop” that counts up to the number of iterations for the topology division, this number is set from the third gene phenotype (see table 4.6), and is set to 8 in this example. Inside the loop the “Remapping” process reads the information of the previous map, the axiom and the rules and from these it returns a



(a) vertices identification (b) edges identification (c) labels from the axiom

Figure 4.2: Initial map definition

new map structure. The complex functionality of this process is explained from section 2.1 to 2.2.1. What it does is essentially divide the existing cells if the rules apply and decide whether the divisions are possible or not while keeping the topology generation constraints. Figure 4.3 shows the sequence for the 8 iterations.

As an overview of the Automatic Topology Generation process a high level flow diagram is presented in figure 4.4.

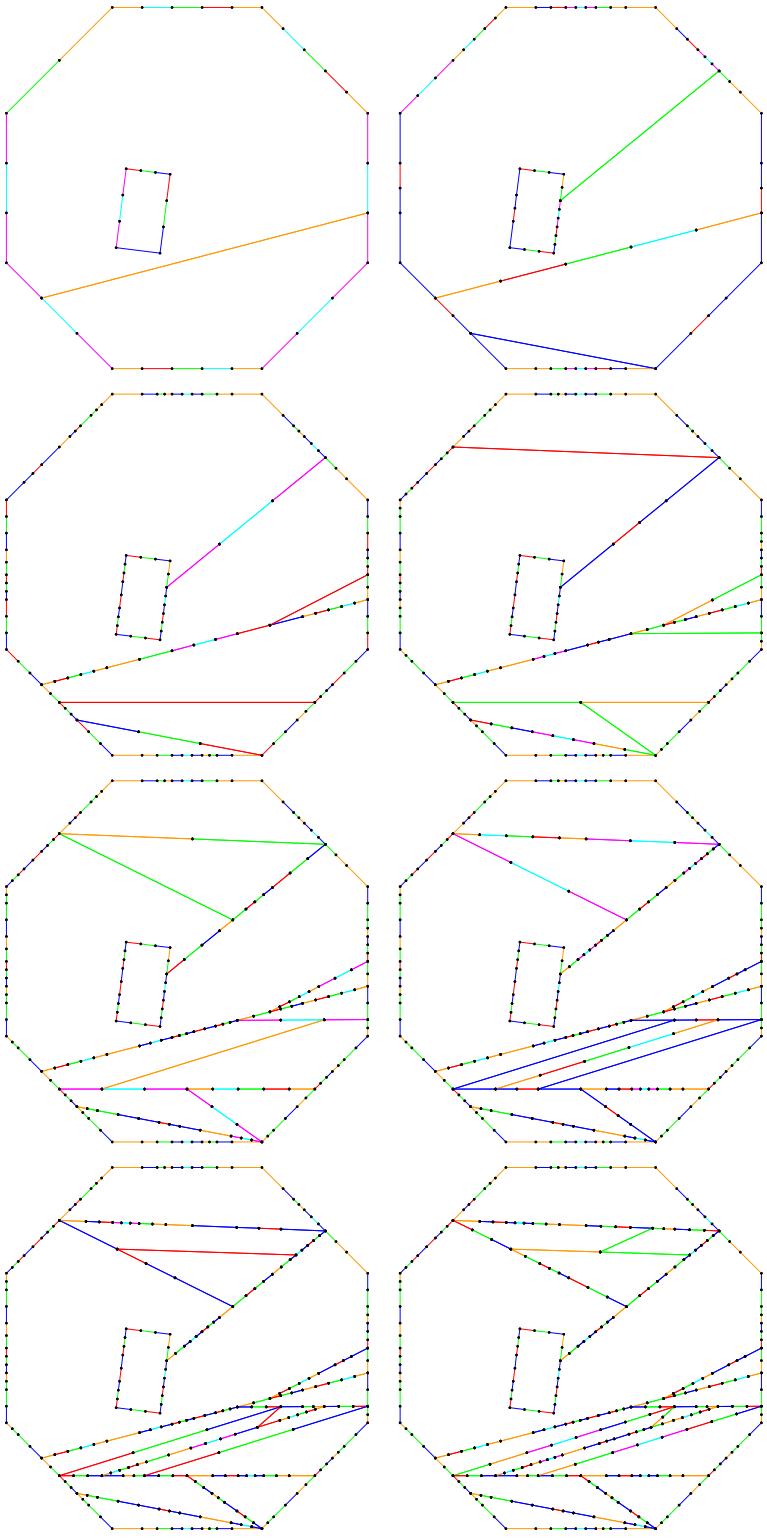


Figure 4.3: 8 steps in the cellular division process using the “Remapping” process for the benchmark example

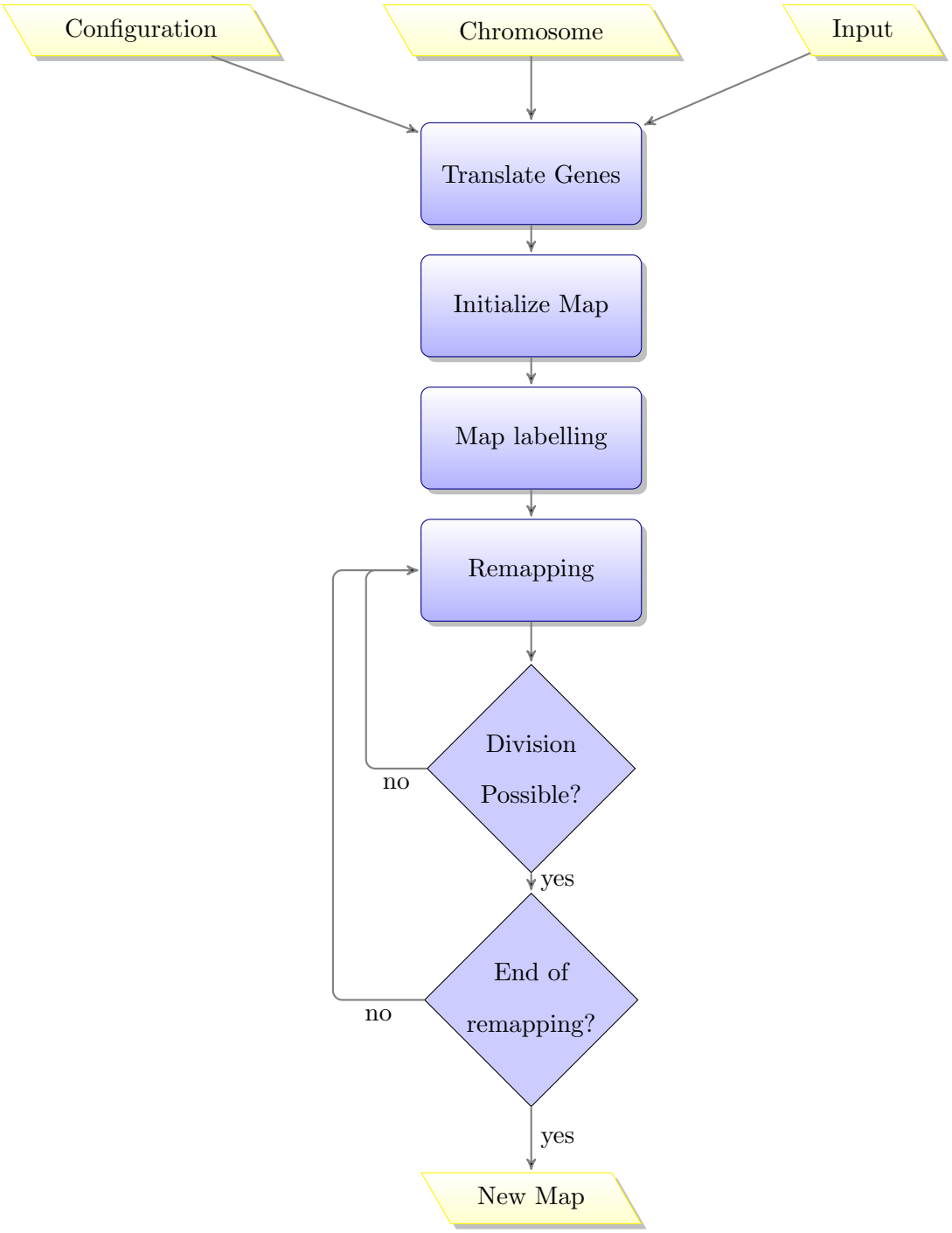
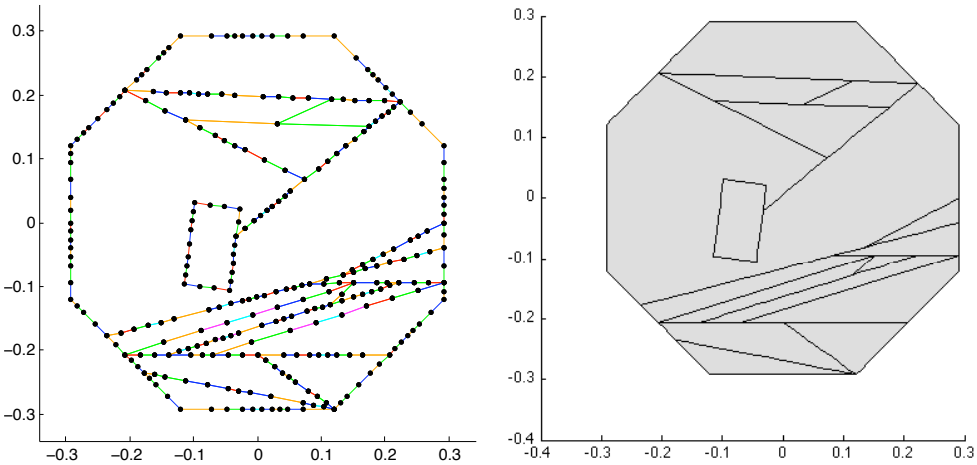


Figure 4.4: Flow diagram of the automatic topology generation software

4.2 Structural Analysis

The Structural Analysis process is applied to the resulting map from the Automatic Topology Generation process converting the map to a structural equivalent form and then analyzing its mechanical behaviour using a Finite Element Method in COMSOL Multiphysics™ . The methods that support this section of the software are described in chapter 3.

The software process starts with the set-up of the physical geometry for the Finite Element Method. This reads the information provided in the map structure and for every cell it populates the vertices, the edges and finally the faces for the structural object. The "geomcoerce" command in COMSOL Multiphysics™ forms the union of the geometry objects given into solids so they are not more abstract entities. The next step is to embed the just created planar geometry in a three-dimensional working plane, this is done with the command "embed" in COMSOL Multiphysics™ . Finally, to define the geometry, the "geomcsg" command is requested to analyse the geometry model for possible errors in the geometry and to generate a specific format to be used for the mesh generation process. Figure 4.5 shows the geometry generated at this stage by the COMSOL Multiphysics™ commands.

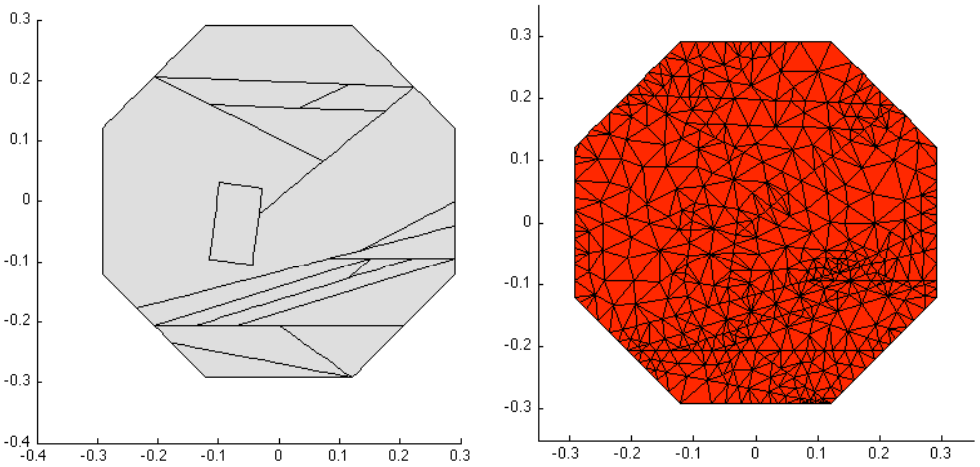


(a) Topology given from the Automatic Topology Generation process

(b) Geometry generated in COMSOL Multiphysics™ from the given topology

Figure 4.5: Geometry created in COMSOL Multiphysics™ from the topology map

To continue the preparation of the structural analysis using Finite Element Method it is necessary to generate a free mesh derived from the analysed geometry. This is achived with the commmand “meshinit”. The mesh size can be controlled using this command. Figure 4.6 shows the mesh generated after the geometry was set in COMSOL Multiphysics™ .



(a) Geometry generated in COMSOL Multiphysics™ from the given topology

(b) Mesh generated in COMSOL Multiphysics™ from the given geometry

Figure 4.6: Mesh generation in COMSOL Multiphysics™

Two more modes are required to start the Finite Element Method , that are the physical definitions of the structure using the application mode of COMSOL Multiphysics™ . The first application mode is the “Shell Application” from the Structural Mechanics Module in COMSOL Multiphysics™ and the second is the “3D Euler Beams Application”.

To obtain a physical equivalence for the these two modes it is necessary to divide the panel in the different conceptual sections as presented before:

1. the global shell
2. the subsystem shell (IMU)
3. the external beams
4. the internal beams

The shells are analyzed in COMSOL Multiphysics™ using the Structural Mechanics Module for Shells while the beams are analysed using the Structural Mechanics Module for 3D Euler Beams.

To be computed in COMSOL Multiphysics™ the shells must have the boundary conditions defined. This is done assigning the material properties, the thickness, and the loads applied to the shell. The material properties are defined according to the selected alloy (see table 5.1). The thickness is selected from the Genetic Algorithm and represented by the genotype. This information is carried by the 3$^{\text{rd}}$ gene of the chromosome which has the information that directly affects the physical properties of the structure. Table 4.6 shows in detail the gene that is used has benchmark and its physical interpretation. The thickness is then assigned to each shell accordingly. Finally, to complete the setup for the physical analysis of the shells it is necessary to compute the separate forces that the loads exert on the shell. This is done by applying the face load in the x and z directions with equations 4.3 and 4.4.

$$F_{x_{shell}} = \frac{mass_{\text{global shell}} \times load_x \times \mathrm{g}}{area_{shell}} \tag{4.3}$$

$$F_{z_{shell}} = \frac{mass_{\text{global shell}} \times load_z \times \mathrm{g}}{area_{shell}} \tag{4.4}$$

The next step is to setup the physical properties for the beams in COMSOL Multiphysics™ . As done previously for the shells, the material properties must be defined using the same procedure. It follows the vertices physical definition with the indication that the eight vertices of the octagon are fixed and that the four vertices of the subsystem (IMU) have an applied inertial load that is given by equations 4.5 and 4.6, these equations refer to a unique vertex.

$$F_{x_{IMU_i}} = \frac{mass_{IMU}}{4} \times load_x \times \mathrm{g} \tag{4.5}$$

$$F_{z_{IMU_i}} = \frac{mass_{IMU}}{4} \times load_z \times \mathrm{g} \tag{4.6}$$

After the physical properties of the vertices have been configured it is necessary to set the physical properties for the beams (or edges). Using the information from the genotype about the external and internal beams feature size it is possible to assign the desired size, the height z and height y are set as well as the cross sectional area. It is also necessary to apply the force per unit length of each beam according to equations 4.7 and 4.8.

$$F_{x_{beam}} = \frac{mass_{beam} \times load_x \times \mathrm{g}}{\text{lenght beam}} \tag{4.7}$$

$$F_{z_{beam}} = \frac{mass_{beam} \times load_z \times \mathrm{g}}{\text{lenght beam}} \tag{4.8}$$

Finally, the torsional constant J and the area moments of inertia I_x and I_y for the squared beams are included. These are set according to equations 4.9 and 4.10

$$J \simeq \frac{a^4}{7.10} \tag{4.9}$$

$$I = \frac{a^4}{12} \tag{4.10}$$

With this configuration set it is then possible to analyse the shells and the beam using the Finite Element Method within COMSOL Multiphysics™ and continue to the post processing environment. COMSOL Multiphysics™ has some functions that select the maximum like "*postmax*" or minimum "*postmin*" of a data set. The necessary information to complete this process is the maximum displacement of the panel and the maximum von Mises stress. The post processing functions accept input arguments to change the default units for the results so the maximum displacement is given in μm and the max von Mises stress is given in MPa.

The final step is to compute the fitness value given the parameters for the current panel. This information is used in the Genetic Algorithm to select the best individuals from the population generated. The fitness value takes into account "how good" the given structure is. Because this is a single objective optimization, where the single objective is to minimize the mass while keeping the constraints, the fitness could be merely the mass

value. Nevertheless, this would not take into account the other constraints, the maximum displacement and the maximum stress. It is not necessary to penalize the fitness if the requirements are kept so an exponential mapping function is used to convert the maximum displacement and stress into a neutral domain if the values are within the constraints, and to a penalization domain if they are found outside the boundaries. The penalization equation is given in 4.11 and figure 4.7 shows the behaviour of this function for an unitary constraint.

$$penalization = e^{max[0,x_{In}-x_{Lim}]} - 1 - max[0, x_{In} - x_{Lim}]; \tag{4.11}$$

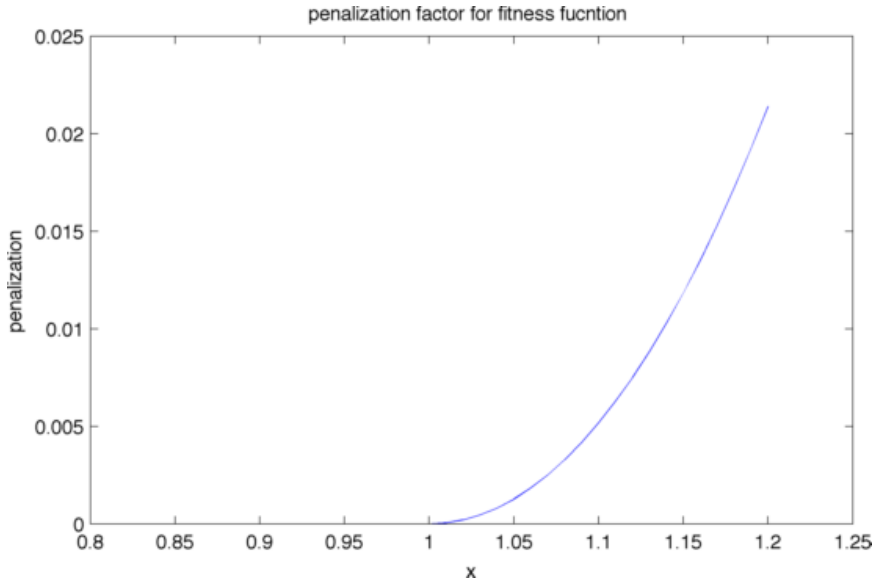


Figure 4.7: Example of the penalization function to be used in the fitness value computation

Finally the fitness value is computed as given by equation 4.12

$$\begin{aligned} fitness = & \frac{\text{mass of current map}}{\text{reference mass}} \\ & + \lambda_{disp} \times \text{penalization(displacement)} \\ & + \lambda_{vMises} \times \text{penalization(von Mises)}; \end{aligned} \tag{4.12}$$

The lower this value is the fitter the individual is.

Next section will show how the Automatic Topology Generation plus the Structural Analysis code are integrated into the search heuristic based on the Genetic Algorithm to obtain an optimized structure.

4.3 Search for the Optimal Structure

Searching for the optimal structure using a systematic approach is the main objective for this thesis. This search process uses the Genetic Algorithm as the search algorithm because the fitness function is not known and is highly dependent on a myriad of variables. The methods that support this section of the software are described in section 2.3. This search process is implemented using the Genetic Algorithm and Direct Search Toolbox in MATLAB® .

To start the process it is necessary to set the Genetic Algorithm options using the "gaoptimset" command in MATLAB® . The most important options that need to be set are the number of generations the Genetic Algorithm will run, the population of individuals for each generation and the number of selected individuals that will pass to the next generations. All these parameters are defined in the configuration parameters.

Because this run is for a single objective optimization (only the mass is to be optimized) the "ga" command in MATLAB® is used. If instead a multi-objective optimization was required the "gamultiobj" would be used. This command calls the fitness function handle for every new individual to be tested during the run, it is the fitness function that calls the procedures previously discussed: Automatic Topology Generation and Structural Analysis, returning the fitness value for every individual. For each generation the individual with lower fitness value is kept since its the most optimized structure.

Figure 4.8 shows the overall process in a flow diagram for the structural optimization process that encompasses the software developed for this thesis.

The next chapter will present the results obtained from this process development.

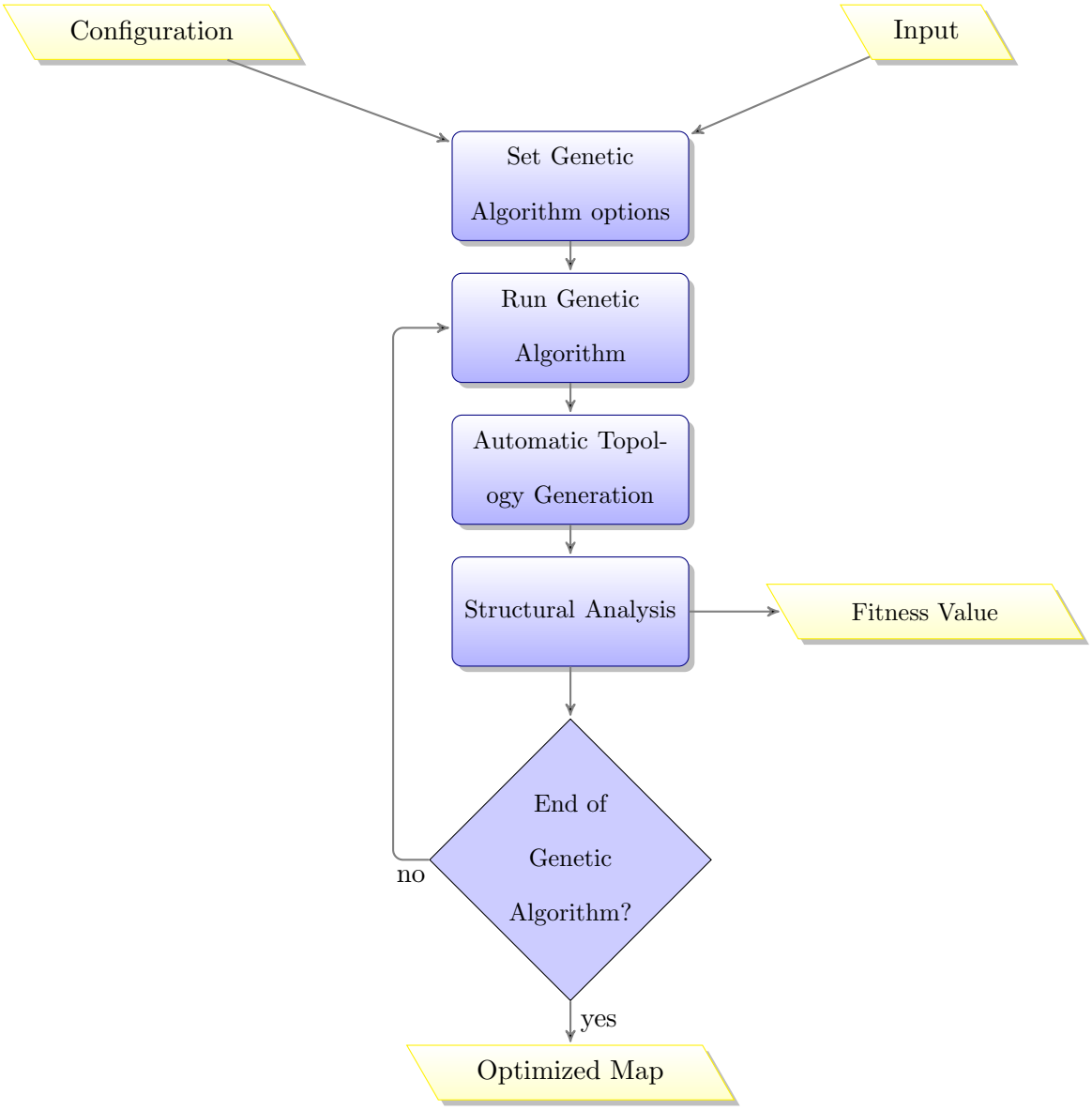
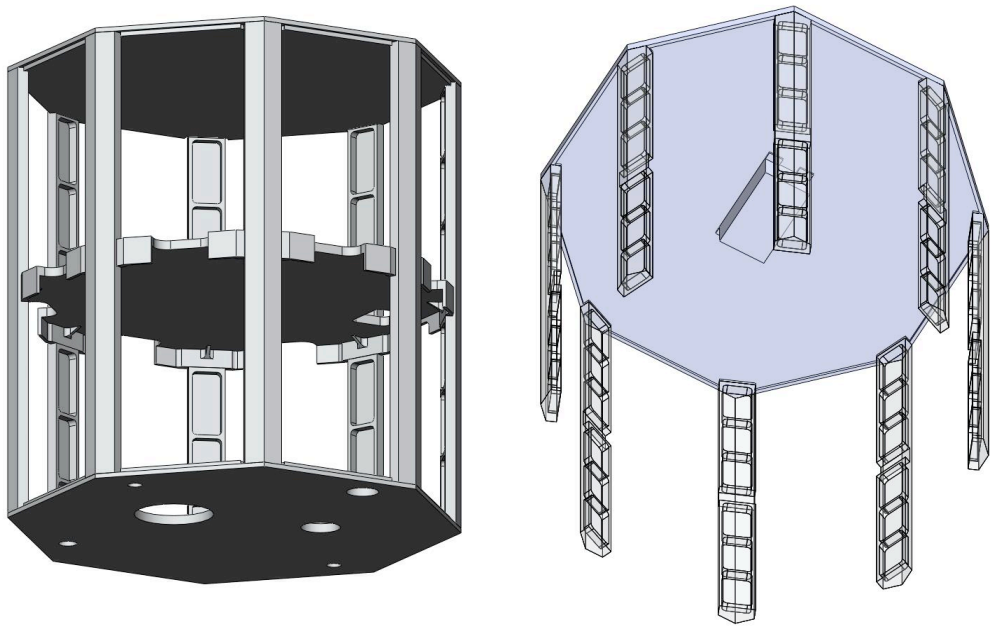


Figure 4.8: Flow diagram of the structural optimization procedure

CHAPTER 5
RESULTS AND ANALYSIS

The previous sections presented the building blocks for the method developed in this work: the topology optimization of mechanical structures. This section shows the results of the methodology applied to optimize the mass of a structural component for the HawaiiSat-1 satellite under the launch conditions.



(a) deck is represented at the top of this figure

(b) top deck, the ribs and the subsystem with transparency seen from "bellow"

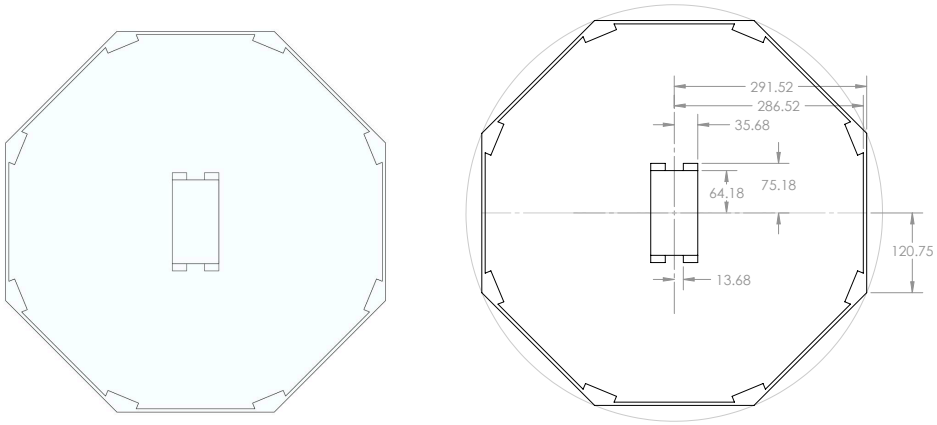
Figure 5.1: Structural Frame of the HawaiiSat-1

5.1 Satellite Panel Design

The structural component to be optimized is shown in figure 5.1 as well as the subsystem in place. This structural component is designated as the *nominal zenith deck* or simply the *top deck*. The panel (or deck) is part of the satellite structure and is connected to 8 ribs for structural support. The subsystem in this case is an Inertial Measurement Unit (IMU).

5.1.1 Panel Geometry

The geometry of the deck is shown in figure 5.2. This geometry consists of an octagon with an internal rectangular area whose structural elements cannot cross.



(a) panel topology (b) panel dimensions in mm

Figure 5.2: Geometry of the panel

Figure 4.1 shows the geometry drawing with the vertices and edges identified. To complement this figure please refer to the table 4.7 that lays the initial map in terms of edges connections and coordinates.

5.1.2 Panel Optimization Features

The top deck will be structurally optimized to minimize its mass while keeping the structural constraints which are given from the structural requirements for the satellite. The maximum displacement of the shell is to be 1 mm. Using a safety factor of 1.5 this requirement is changed to 0.5 mm (or 500 μm) of maximum displacement. The stresses should be within the allowable range, that is, bellow the yield stress of the material to be used. In this case the material used is the Aluminum Alloy 6061-T6 that is known to have high strength and good workability. This alloy has an yield strength of at least 241 MPa and an ultimate tensile strength of 290 MPa[1]. Using the same safety factor of 1.5 we get a yield of 120.5 MPa. The

[1] other typical values are 275 MPa for the yield strength and 310 MPa for tensile strength but these were not used in the optimization runs in COMSOL Multiphysics™

main reason for using this material in this work is to conform with the HawaiiSat-1 project materials selection. Table 5.1 has the material properties for this alloy.

	Aluminum 6061-T6
Young Modulus [GPa]	68.9
Poisson Ratio	0.33
Density [kgm^{-3}]	2700
Shear Modulus [MPa]	25.84
Yield Strength [MPa]	241

Table 5.1: Material Properties of the Satellite Structure

The design parameters for this work are: Map *L*-system parameters (topology); plate thickness for the subsystem region; plate thickness for the main region of the panel; the side length of the external beams; and the side length of the internal beams and the sub-system placement. The boundary conditions are defined to have the eight vertices of the initial map (the octagon) fixed and the boundary edges are free.

There are essentially two components used in the panel: the shell and the beams. The shell can be divided in two segments, the one called the *global shell* that is the main structural support of the panel excluding the subsystem component, and the other is the subsystem component itself - also referred as the *subsystem shell* whose thickness can be different from the *global shell*. All components of the panel are built with the same material.

The beams are placed on top of the *global shell* and their placement is dependent on the map generation algorithm (generated in MATLAB®), which in turn is dependent on the original gene submitted. Figure 5.3 shows an example of the development stages for the map generation algorithm for this panel. The beams are divided in two categories, the internal beams and the external beams. The internal beams are all the newly created beams during the topology development process and the external beams are the ones that define the original map. All beams will have a square cross section but this section may differ according to the side length of the beams.

The analysis is done using the software for topology optimization developed for this work and is written in MATLAB® and COMSOL Multiphysics™ scripting language. MATLAB®

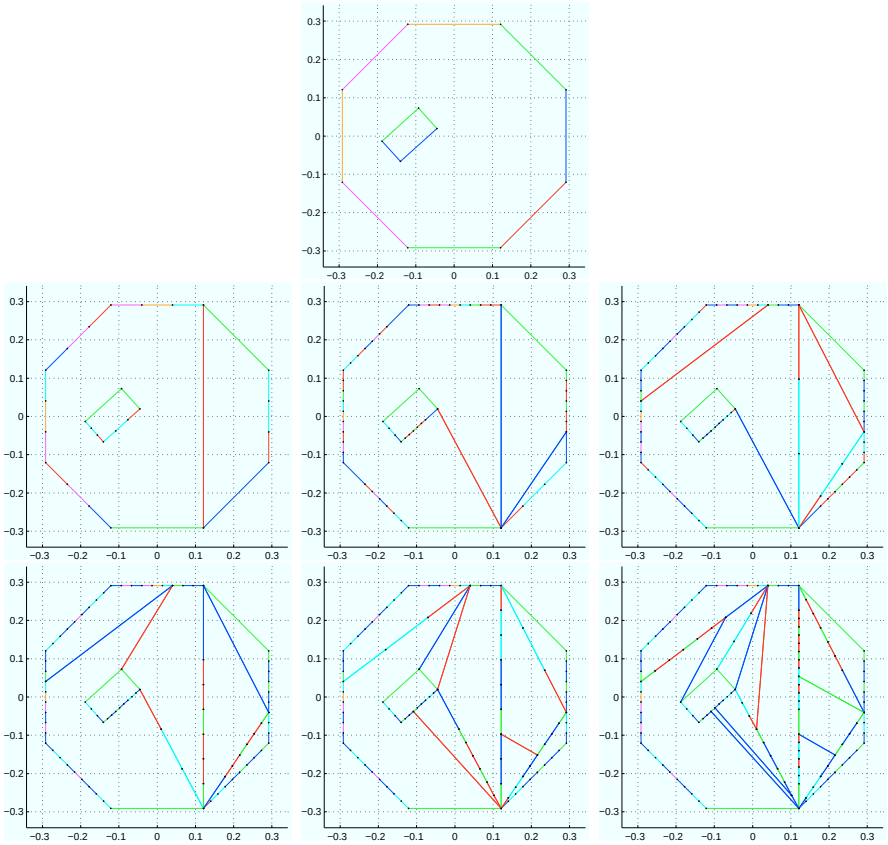


Figure 5.3: Example of the first 6 steps of the cellular division process using the Map L System for the zenith deck of the HawaiiSat-1

is a mathematical and numerical computing software and COMSOL Multiphysics™ is a multiphysics Finite Element Method toolbox. These two software packages can communicate using a local server passing the data from one software package to the other allowing powerful simulations and analysis. All the analysis code is run in MATLAB® which in turn calls the COMSOL Multiphysics™ prototypes required in the script.

The script uses a set of global physical constants that are presented in table 5.2. It is important to note that these constants are used as the physical setting of this specific problem. There is another set if important global constants that determine the division criteria as explained in section 2.1. These are the minimum length of the edges, defined to be 2% of the characteristic length of the panel (that is approximately 0.6 m) and the minimum area possible for a cell that is defined to be 2% of the characteristic area of the

panel (that is approximately 0.28 m^2) and the minimum angle is set to be 10 deg to avoid sharp angles when creating the mesh in the Finite Element Method.

A change in these values might lead to very different results than those presented in this work.

constants	value	description
g	9.80665	gravitic acceleration in m^2/s
$area_{panel}$	0.281605	static area of the planar panel in m^2
$area_{IMU}$	0.009158	static area of the planar IMU in m^2
$length_{edge}$	0.2415	length of the side edges in m^2
$mass_{IMU}$	0.299	mass of the IMU in kg
$load_z$	-10.00	vertical load in g
$load_x$	8.75	lateral load in g

Table 5.2: global constants used in the program that represent physical terms

5.2 Mesh Independency study

A test was done to determine the resolution necessary to be used on the mesh generation process. This procedure guarantees accuracy for the Finite Element Method results on the variables that go into the fitness equation. In COMSOL Multiphysics™ there are nine different levels of mesh resolution, one is the finer mesh with more elements and nine is the coarser mesh with less elements. One can see the difference of the different meshes sizes from figure 5.8.

Different tests were conducted to determine the level of refinement necessary to use in the the runs for the Genetic Algorithm. It was determined that level 1 is necessary to have an accurate result. Figure 5.4 shows the test results for the different selected meshes. It can be seen that level 5 and lower do not affect significantly the displacement but the stress still changes considerably.

After the optimization run more tests were done to confirm the selection of the mesh size in the COMSOL Multiphysics™ setup. This test is shown in figure 5.5 which confirms the previous selection of the mesh level in COMSOL Multiphysics™ . Though the stress results are affected if using a coarser mesh, the displacement results are approximately steady if

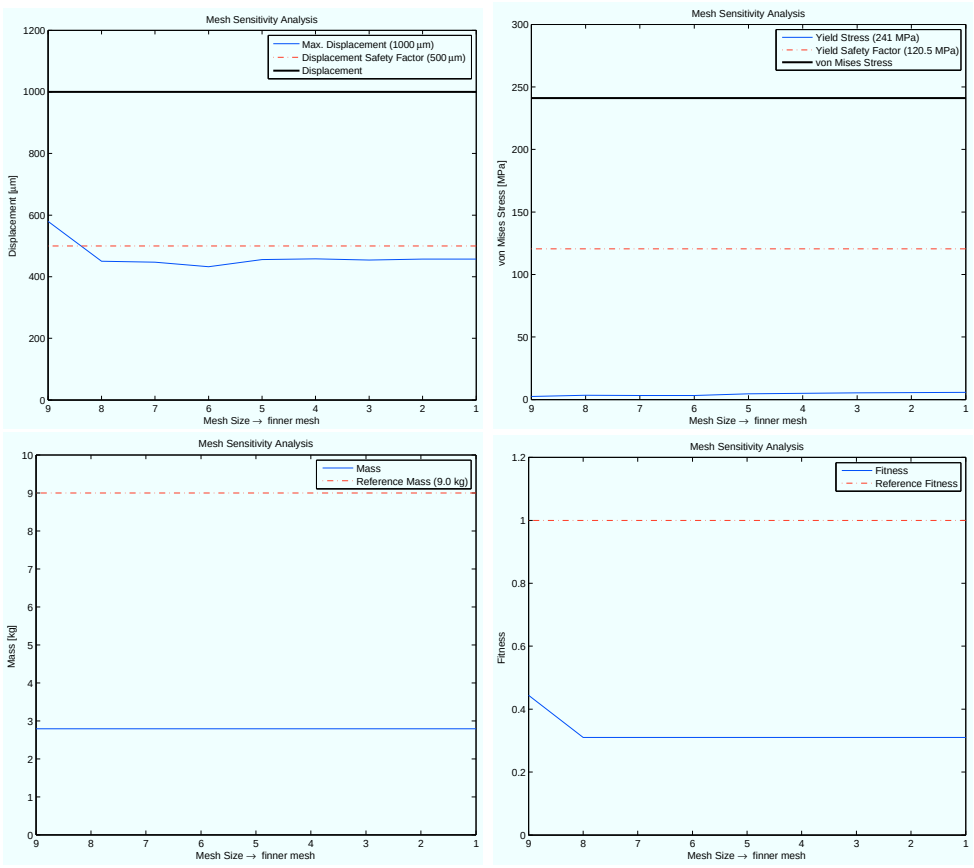


Figure 5.4: Mesh independency study made with a generic structure (non-optimized). Displacement requires level 5 or lower and Stress measures requires mesh level 1 for accurate results.

level 5 or lower are used. Because the yield strength of the selected material (AL 6061-T6) is 241 MPa (or 120.5 MPa considering the safety factor) and the given results are within a much lower range (0 - 40 MPa) the fitness value is not be affected. This concludes that the use of the level 5 or lower in the COMSOL Multiphysics™ analysis is acceptable.

To complete the mesh independency study the SolidWorks model for the most optimized structure was also evaluated according to the different mesh sizes available in the software. The different mesh sizes were discretized in the same way done in COMSOL Multiphysics™ to make the results comparable. This test is represented in figure 5.6 which shows that it is necessary to have a very fine mesh to get an accurate displacement equivalently done for COMSOL Multiphysics™ .

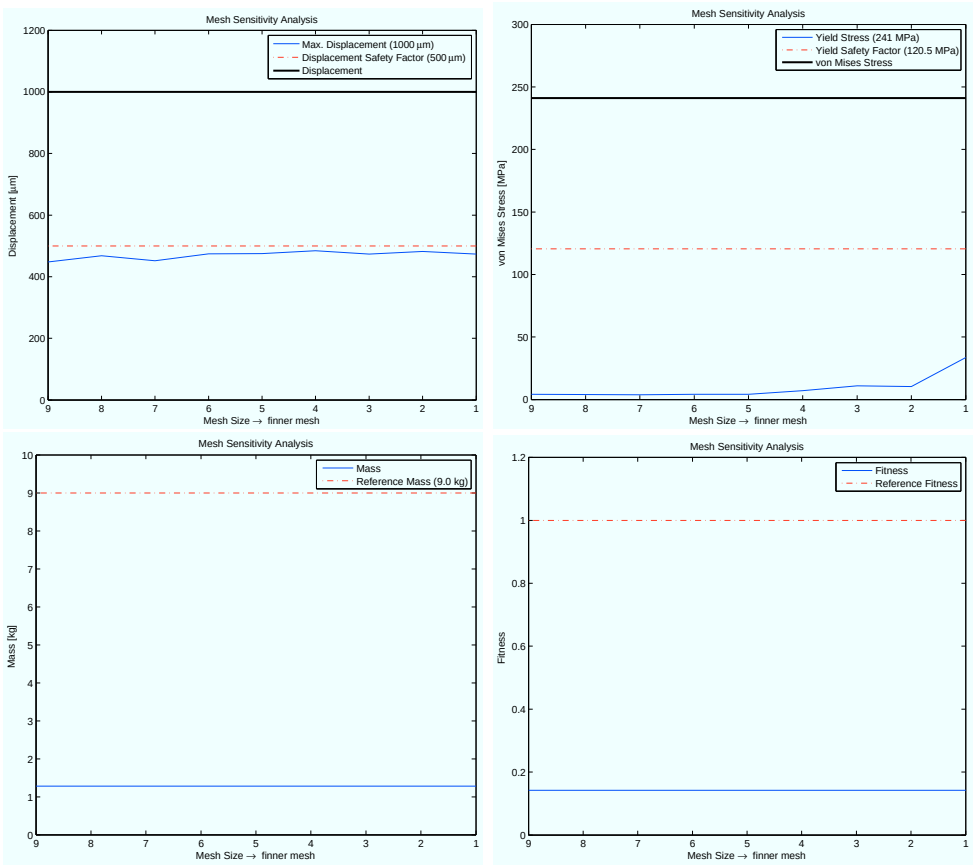


Figure 5.5: Mesh independency study made after the optimization run in COMSOL Multiphysics™ for the most optimized structure. Displacement and Stress measures require mesh level 1 for accurate results.

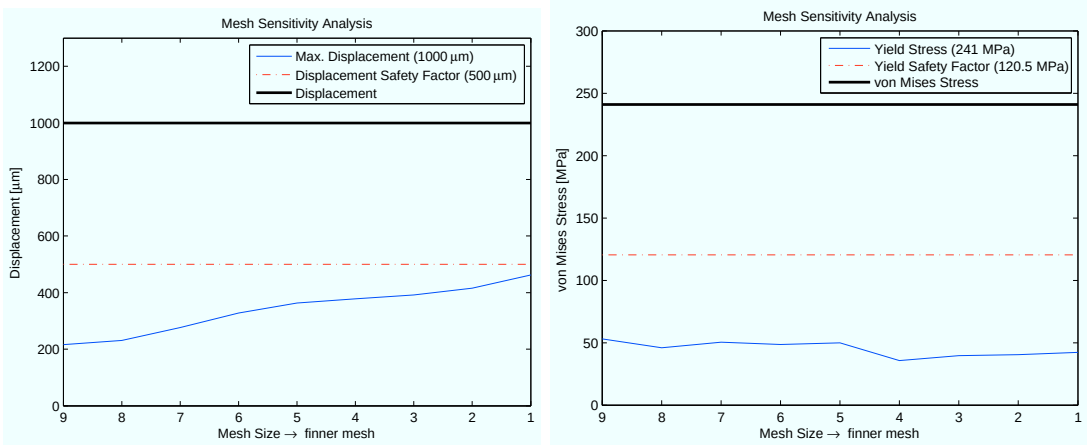


Figure 5.6: Mesh independency study for the most optimized structure using SolidWorks. Displacement and Stress measures require mesh level 1 for accurate results.

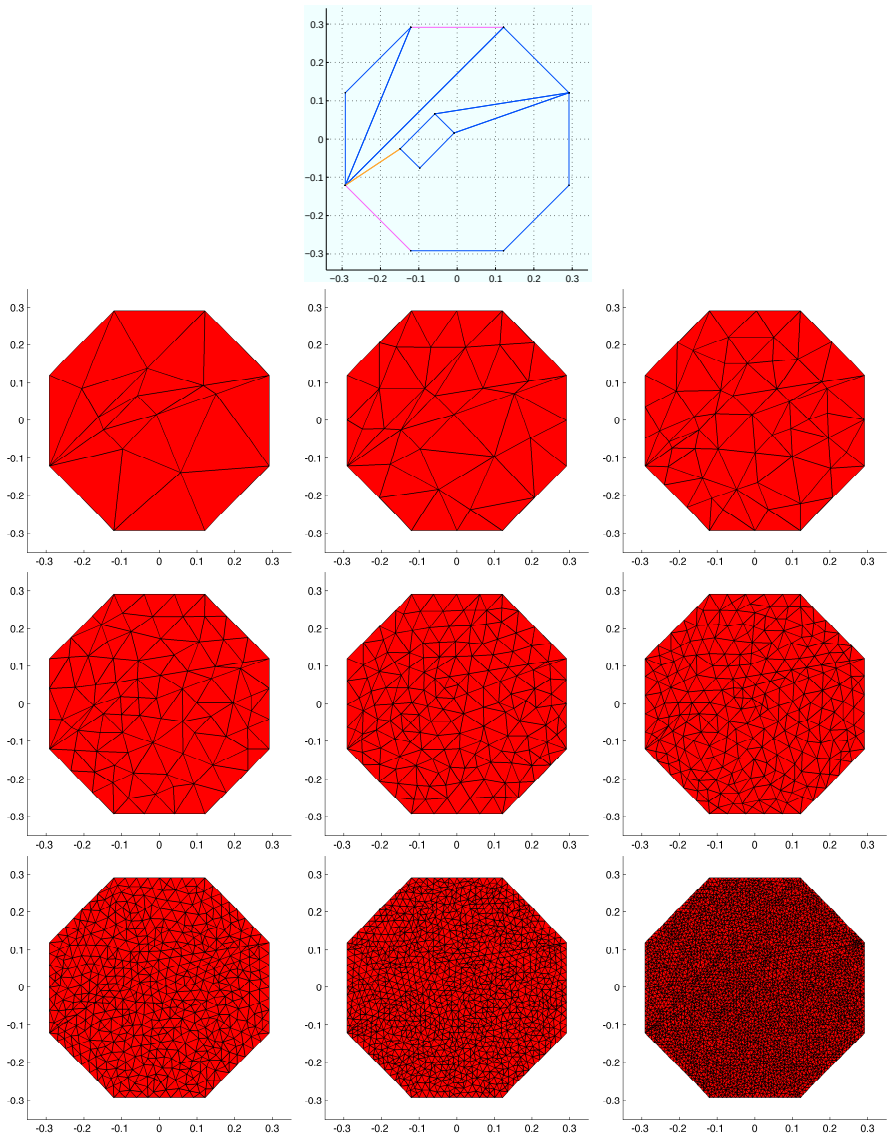


Figure 5.7: Top figure shows the original map to which the mesh independency study is made, the remaining figures show the refinement levels applied to this map. The more coarser mesh is level 9 on top left, the finer mesh is in the bottom right - level 1.

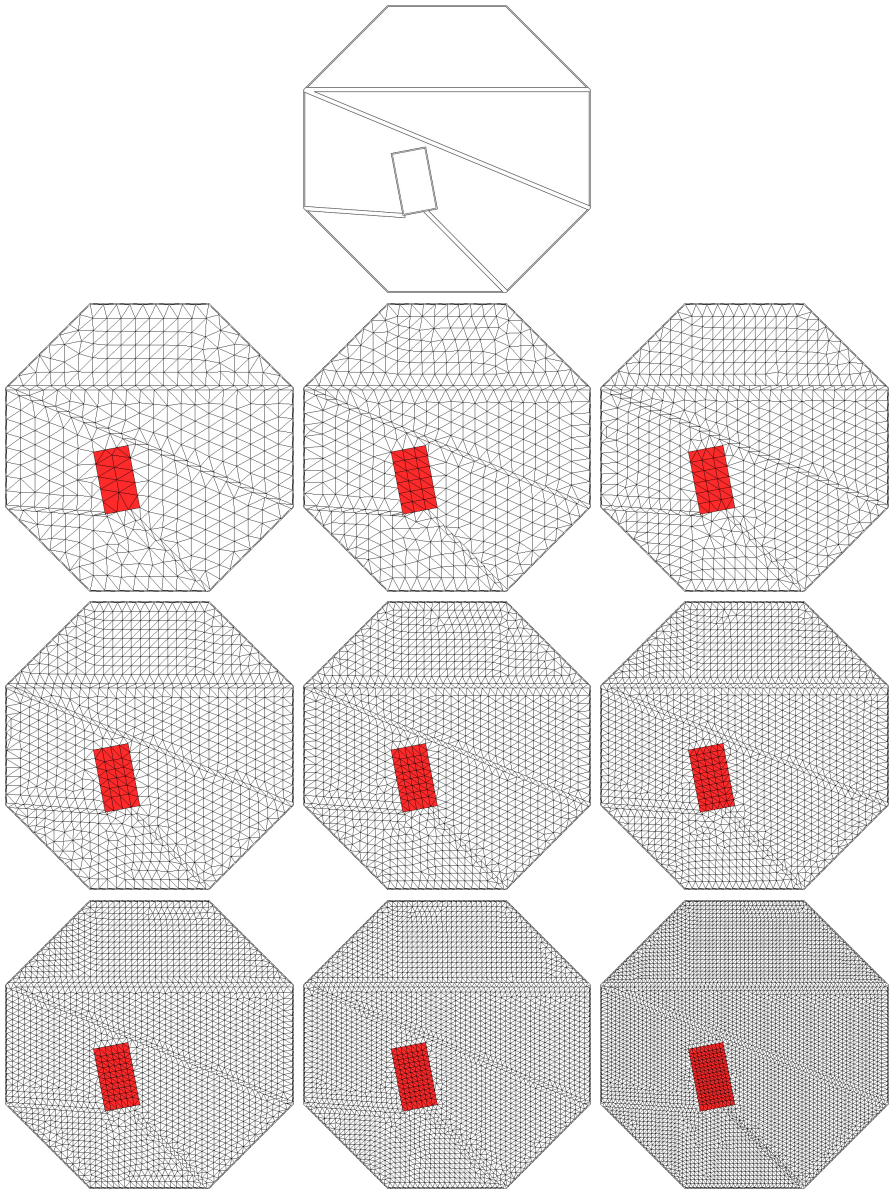
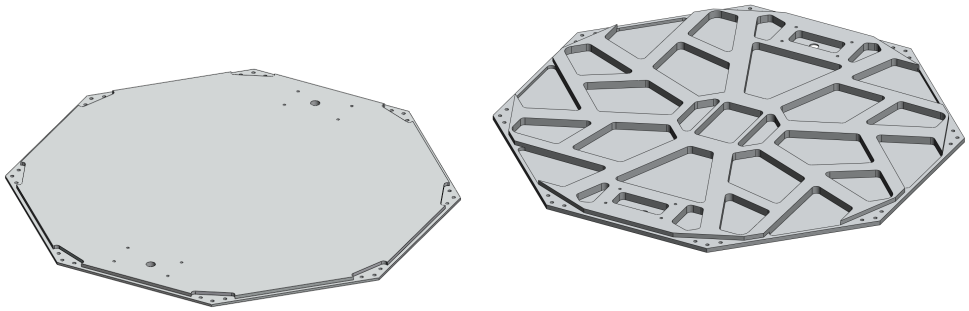


Figure 5.8: Top figure shows the top view for the SolidWorksmodel, the remaining figures show the refinement levels of the mesh applied to this model. The more coarser mesh is level 9 on top left, the finer mesh is in the bottom right - level 1.

5.3 Benchmarks

This section will show some benchmarks used to compare the optimization results. The first benchmark is set after the baseline given at the Preliminary Design Review for the HawaiiSat-1 structural elements. This panel had a uniform thickness of 12.5 mm and a

weight of 9.37 kg. Figure 5.9a shows the panel as presented in the review.



(a) Unoptimized Panel, mass = 9.37 kg

(b) Optimized panel by an experienced engineer, mass = 3.8 kg

Figure 5.9: Two possible benchmarks for the structural panel of the Satellite.

The objective is to minimize the mass of the panel and still meet the requirements as presented in the subsection 5.1.2.

Another benchmark of 3.8 kg is presented after the optimization done by an experienced mechanical engineer. Figure 5.9b shows this design.

benchmark	mass $[kg]$	Max. Displacement $[\mu m]$
$1 \rightarrow$	9.37	13
$2 \rightarrow$	3.80	81

Table 5.3: Benchmarks to compare with the optimization runs

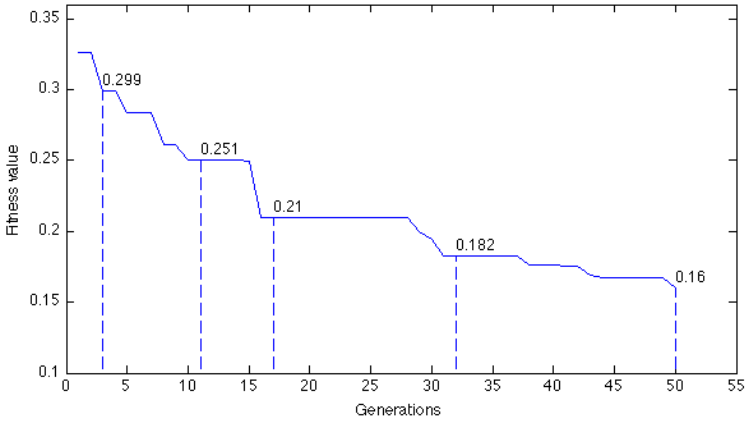
The two benchmarks do not include the weight of the subsystem which in this case is 0.299 kg. The next section will show the results of the optimization process applied to the problem that has been presented in these previous sections.

5.4 Optimization Run

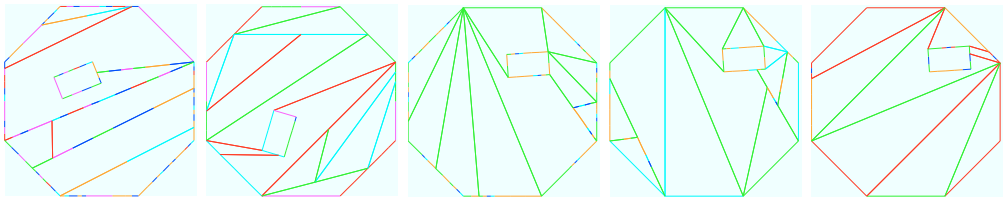
Various optimizations runs were made to compare results. The longest Genetic Algorithm run had a population of two hundred individuals and was run for one hundred generations, the equivalent to 20,000 individuals were evaluated. This run was broken in two sections (run 3.1 and 3.2) because of computer memory problems with pure virtual function calls.

The Genetic Algorithm can resume from any population so it was possible to continue the run without loosing a significant amount of information.

Table 5.4 shows, in brief, the results of the optimization runs using a reference mass of 9.0 kg for the fitness calculation. The Lagrangian multipliers are $\lambda_{disp} = 10$ and $\lambda_{vMises} = 5$, which penalize the displacement more than the stress. The stress levels obtained are much smaller than the yield stress for this alloy which makes it a non-critical criteria so the displacement becomes a more important criteria to follow. For more details refer to the figures 5.10, 5.11 and 5.12. These figures show the sequentiation of the various topologies for the different optimization runs.

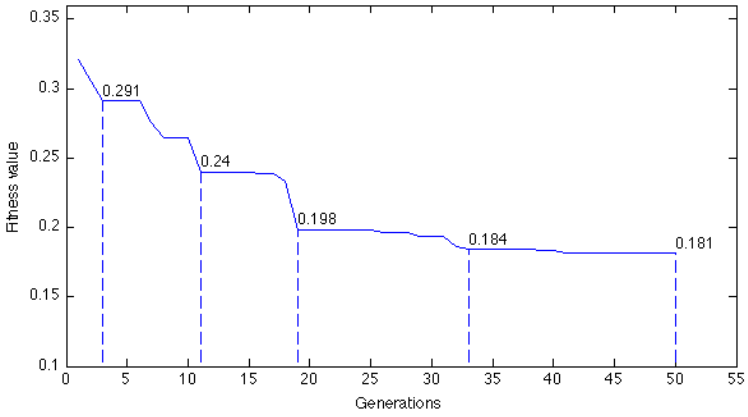


(a) Plot with fitness values for the different generations in the run #1.

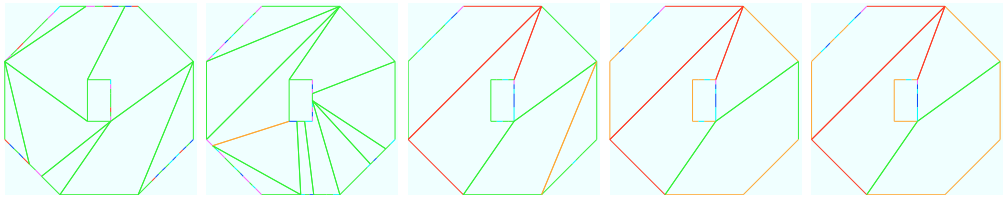


(b) Topologies that correspond to the selected fitness values in the fitness plot above.

Figure 5.10: Topology selection sequencing for run #1.



(a) Plot with fitness values for the different generations in the run #2.



(b) Topologies that correspond to the selected fitness values in the fitness plot above.

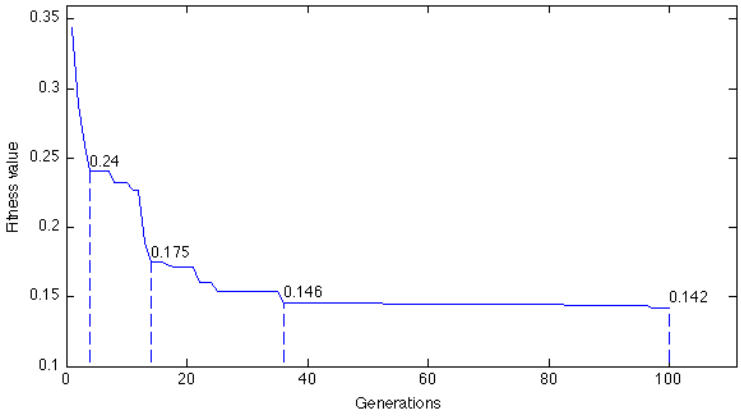
Figure 5.11: Topology selection sequencing for run #2.

Run #	Indiv.	Gen.	Run Time	Fitness	Mass [kg]	Mass Reduction (Bench #1)
1 (free)	100	50	43h 51m	0.1604	1.443	85%
2 (fixed)	200	50	38h 17m	0.1813	1.632	83%
3.1 (free)	200	50	33h 03m	0.1459	1.308	86%
3.2 (free)	200	50	25h 15m	0.1422	1.280	87%

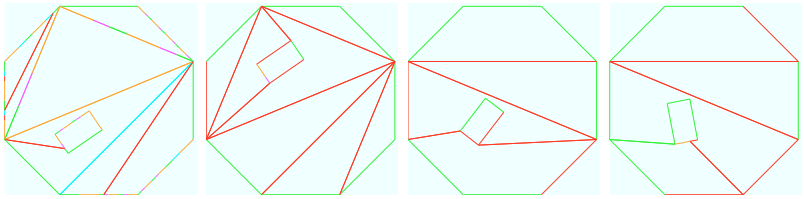
Table 5.4: Different optimization runs with the Genetic Algorithm based on the biologically inspired methodology for topology generation

The results show that the longer the Genetic Algorithm runs the most likely is to find a better structure topology with lower mass. The mass for the run 3.2 is the lowest: 1.280 kg. This is a significant improvement when compared to the benchmarks.

Figures 5.14 to 5.17 show the results after the finite element analysis and optimization run was done. The SolidWorks model for the most optimized structure is shown in figure 5.13. This SolidWorks model has a mass of 1.202 kg which is very close (within 6%) to



(a) Plot with fitness values for the different generations in the run #3.



(b) Topologies that correspond to the selected fitness values in the fitness plot above.

Figure 5.12: Topology selection sequencing for run #3.

the estimated mass in the developed software. The displacement and the stress values are also compared in table 5.5 and shows how close the results are in percentages. The mass and the displacement results between SolidWorks and COMSOL Multiphysics™ are very close which confirms the accuracy of the model implemented in the software developed. The stress values are not so close but because they are well bellow the yield strength of the selected material and also because the mesh grid has different sizes in both models which affects these results more than the displacement.

	mass [kg]	Max. Displacement [μm]	Max. Stress [MPa]
COMSOL Multiphysics™ →	1.280	473.28	33.6
SolidWorks→	1.202	461.91	42.2
absolute difference	6%	2%	26%

Table 5.5: Comparison between results from COMSOL Multiphysics™ and SolidWorks for the most optimized structure.

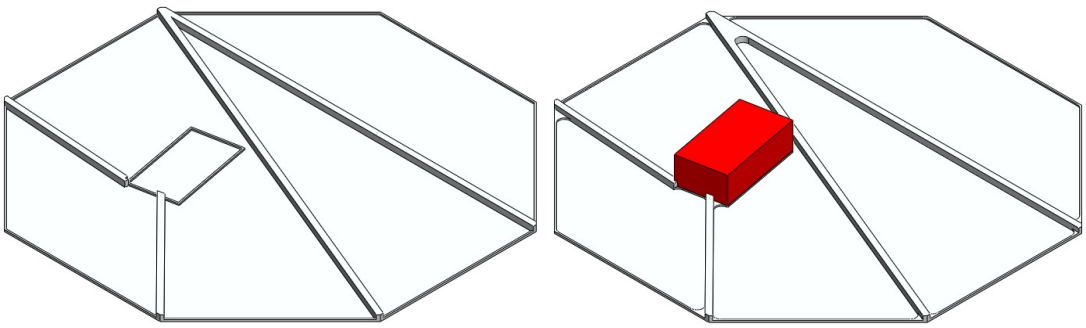


Figure 5.13: SolidWorks model for the best individual. Raw model on the left and finalized model with chamfers on the right.

To better understand the changes for the optimization run that led to the most optimized structure in these tests, with mass of 1.28 kg, some key topology changes were selected after the plot that expresses the fitness value for every generation of the Genetic Algorithm. This plot is represented in figure 5.12.

It is interesting to note that after generation number 40, in the optimization run #3, there is not much improvement in the fitness value and correspondingly the mass of the structure (see figure 5.12a). Another important remark is the fact that the final selected structure becomes topologically simpler the longer the Genetic Algorithm runs to end up with a very simple geometry after all (see 5.12b).

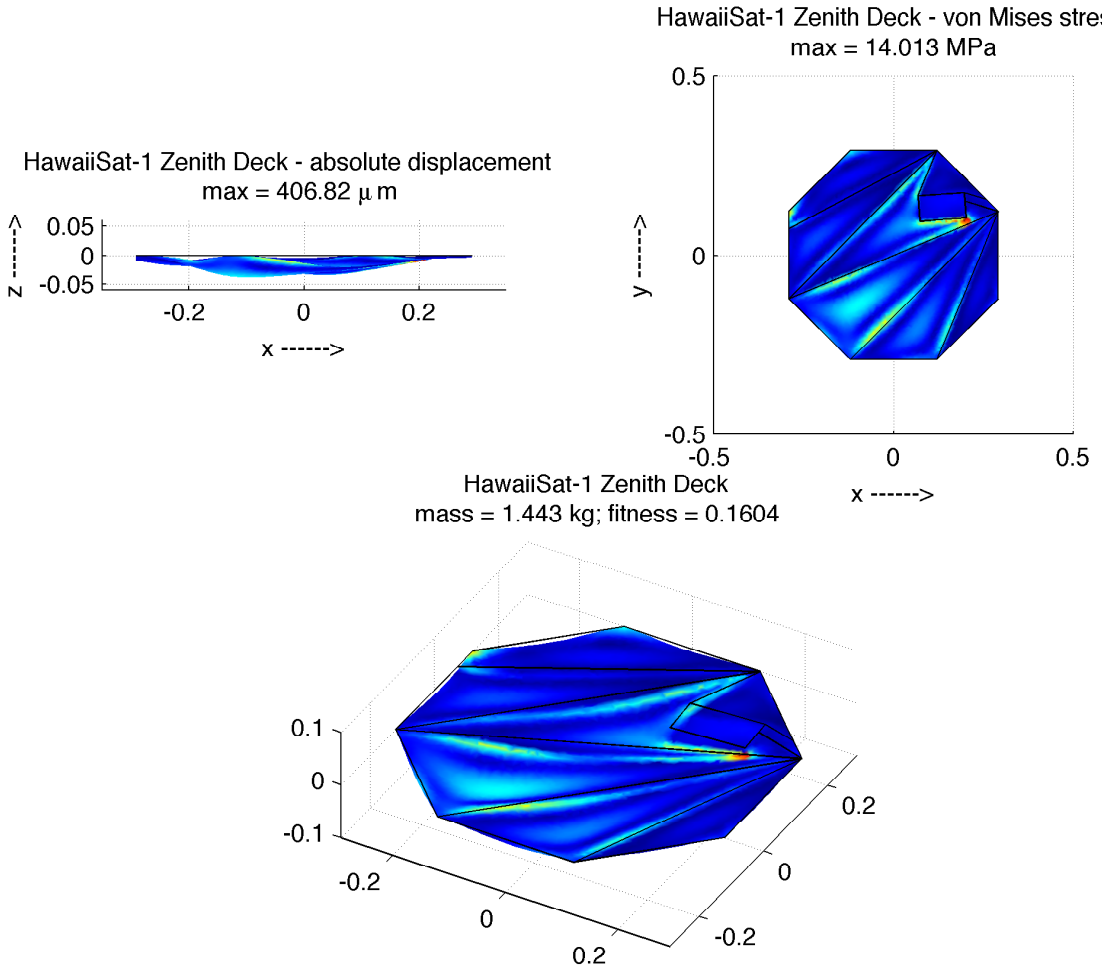


Figure 5.14: Optimized structure after run #1 with 50 generations and 100 individuals. Final Mass = 1.443 kg, Fitness = 0.1604, and subsystem was free to move.

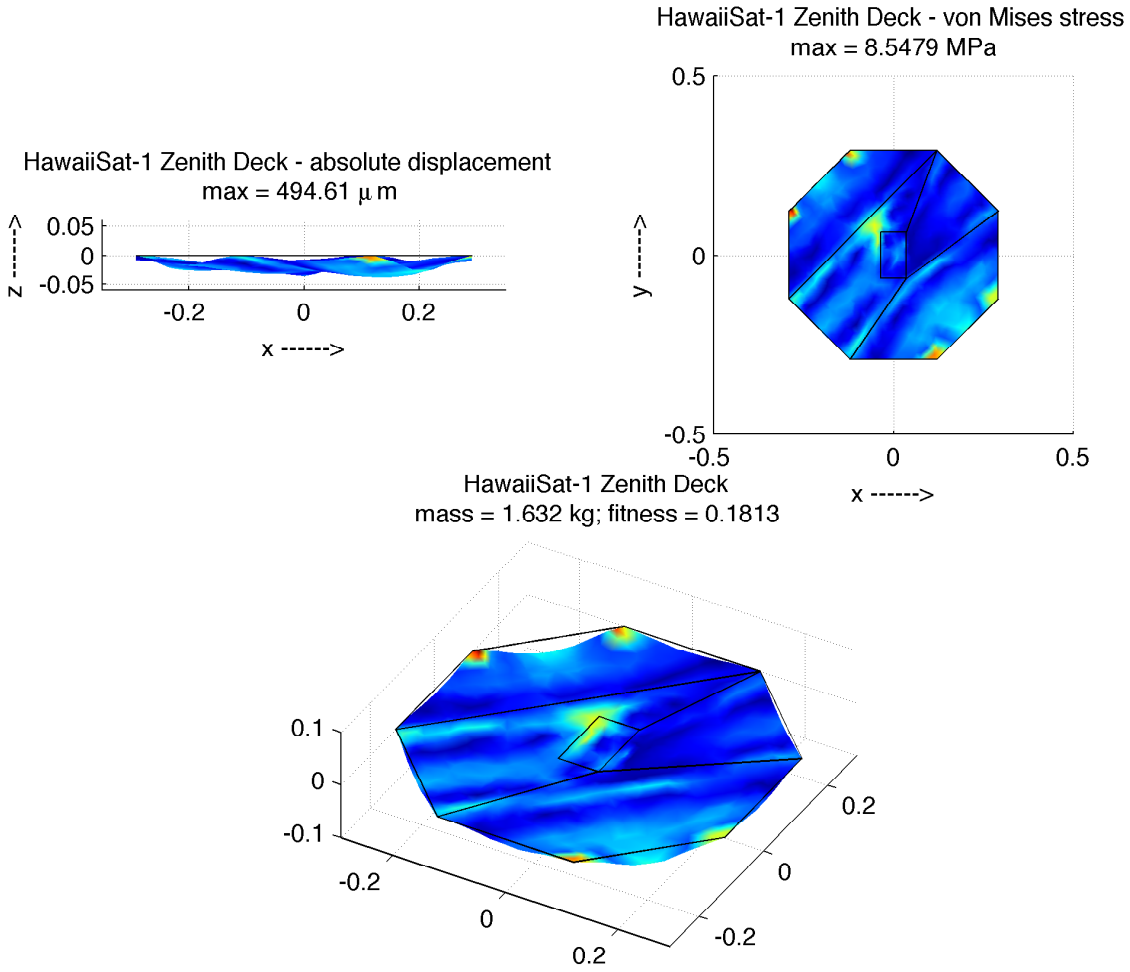


Figure 5.15: Optimized structure after run #2 with 50 generations and 200 individuals. Final Mass = 1.632 kg, Fitness = 0.1813, and subsystem was fixed.

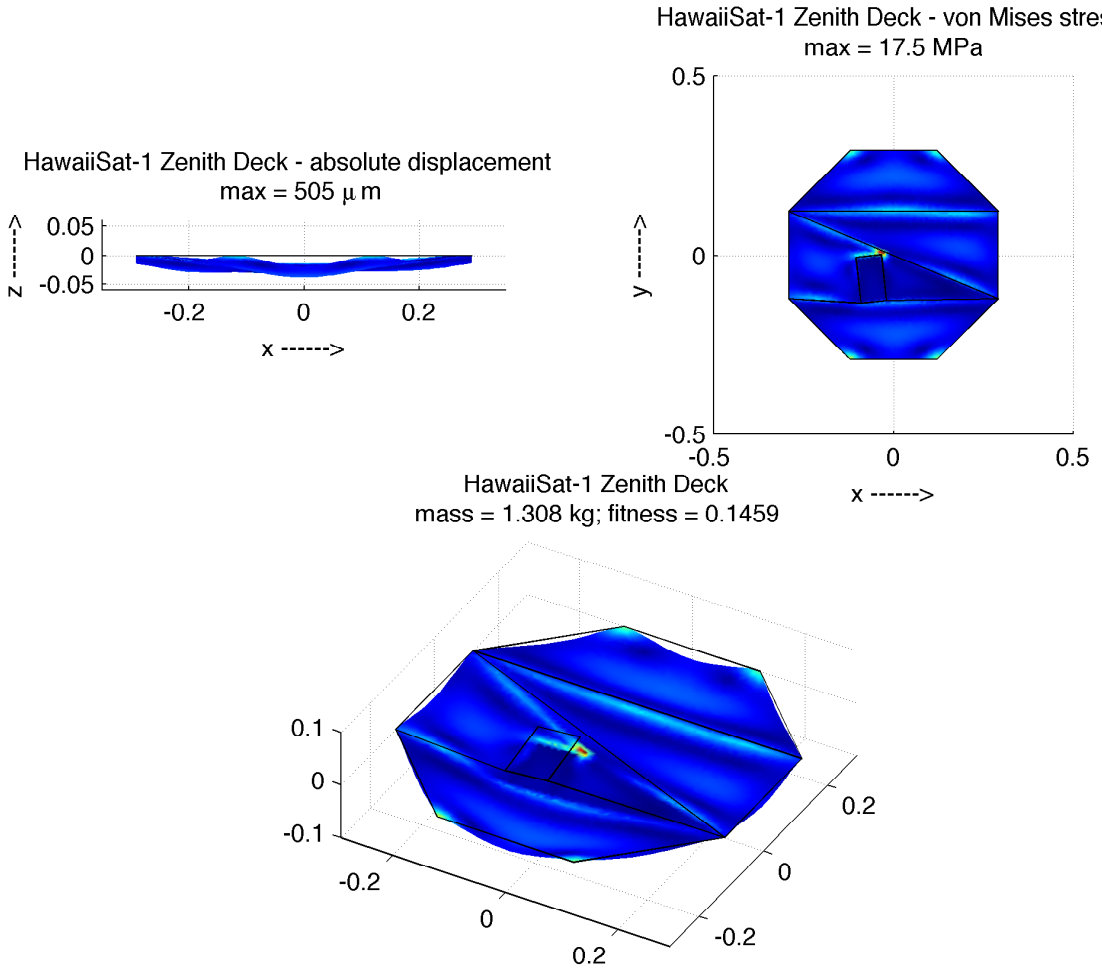


Figure 5.16: Optimized structure after run #3.1 with 50 generations and 200 individuals. Final Mass = 1.308 kg, Fitness = 0.1459, and subsystem was free to move.

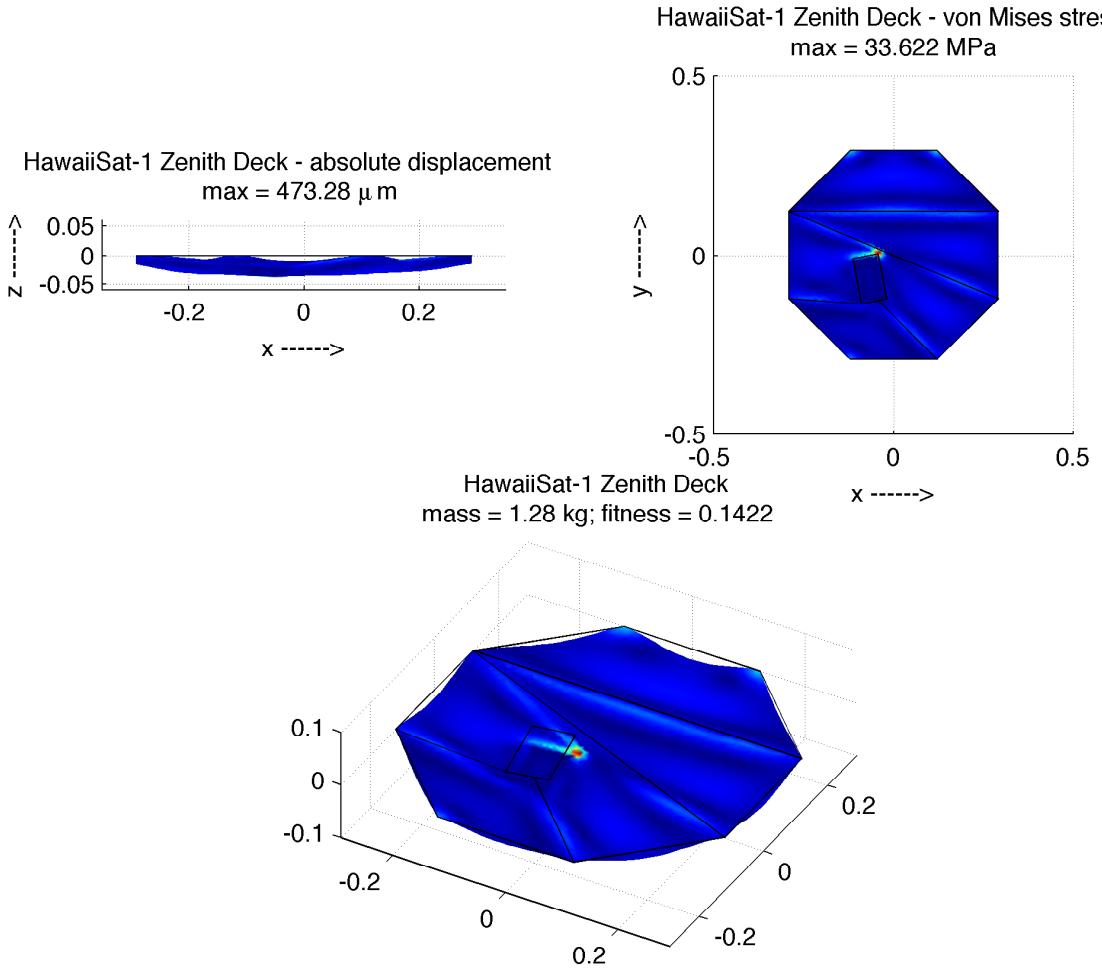


Figure 5.17: Optimized structure after run #3.2 with 50 generations and 200 individuals starting from best individual in run#3.1. Final Mass = 1.280 kg, Fitness = 0.1422, and subsystem was free to move. This is the best structural topology found.

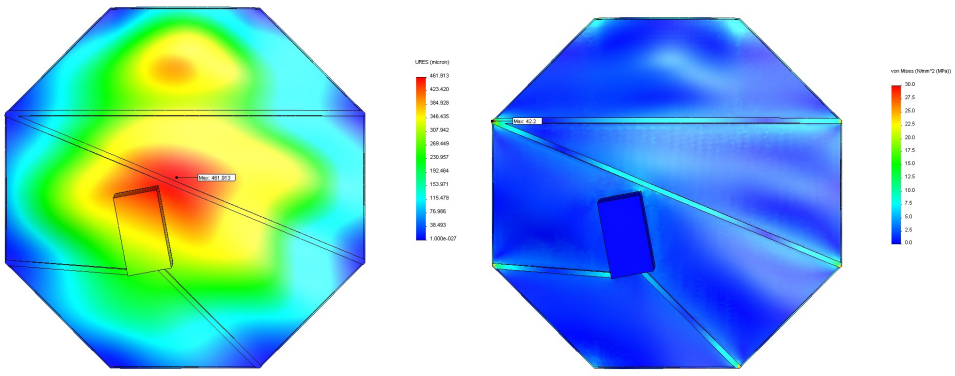


Figure 5.18: Best optimized structure after run #3 modelled in SolidWorks. Mass = 1.202 kg, Disp = 461.9μm, von Mises Stress = 42.2 MPa.

CHAPTER 6
CONCLUSION

This work introduces a new approach of optimal design for structures. The new paradigm is inspired in natural processes of cellular division existing in living organisms that naturally search for optimal layouts.

The multidisciplinary method used for structural optimization is based on the Map *L*-system and was successfully extended using Mathematical and Graph Theory elements to mimic cellular division in complex topologies, specifically topologies that have multiple connected components. These connected components represent physical boundaries on the structure like holes or subsystems.

The newly proposed method to generate topologies with multiple connected components was tested and validated making it possible to create a systematic approach to search for the best topologies (or individuals) using a Genetic Algorithm. This method is also extended to have movable connected components to search for the best placement. Three searches using the Genetic Algorithm were performed and the results revealed surprising and interesting structures.

Given the proposed unoptimized benchmark the mass was reduced by at least 83% (considering the least optimized run with the fixed subsystem). Regarding the engineered design the mass was then reduced at least 60% while keeping the required constraints. Assuming that the price per kilogram in space is $10,000 the savings regarding the unoptimized benchmark are of at least $78,680. The satellite has three of these panels, so considering that each of the other panels can be optimized by a similar amount the savings maybe up to $236,040.

These figures show that the proposed method is competitive, worthy to be implemented, and tested in real world applications.

The proposed method for structural optimization may be utilized for further development and research in vast areas such as Engineering and Biology. In a more immediate

time-frame the code of the software developed will be optimized and improved to be used in a parallel processing environment so the topology generation and structural analysis may be done much faster. Also a systematic check on the analysis results given by COMSOL Multiphysics™ with other software in the market will be done to guarantee the accuracy on the results obtained. A desirable milestone would be the development of an integrated Finite Element Method into the current software. Other important aspects that are worthy to investigate are the following:

- Extend the method as a systematic approach to a complete set of structural elements in a satellite (or any other engineering piece) to reduce the overall structure weight while keeping the requirements. This systematization is a challenge for defining 3D sets of structures instead of 2D as used in this work.

- Extend the method for a a multi-objective optimization with subsystem placement and mass optimization.

- Extend the proposed methodology to other fields of engineering and science, like topology optimization in Field Programmable Gate Arrays for faster performance in electronic circuits.

BIBLIOGRAPHY

[1] M. H. Kobayashi; H-T. C. Pedro; R. M. Kolonay and G. W. Reich. On a cellular division method for aircraft structural design. *The Aeronautical Journal of the Royal Aeronautical Society*, 113(1150):821–831, 2009.

[2] M. H. Kobayashi. On a biologically inspired topology optimization method. *Communications in Nonlinear Sciences and Numerical Simulations*, 787802(15), 2010.

[3] M. H. Kobayashi; H-T. C. Pedro; C. F. M. Coimbra and A. K. da Silva. Formal evolutionary development of low-entropy dendritic thermal systems. *AIAA Journal of Thermophysics and Heat Transfer*, (23):822–827, 2009.

[4] R. M. Kolonay and M. H. Kobayashi. Shape and topology optimization of aircraft lifting surfaces using a cellular division method. In *13th AIAA/ISSMO Multidisciplinary Analysis Optimization (MAO) Conference, Fort Worth, Texas*, 2010.

[5] M. H. Kobayashi; R. M. Kolonay; G. W. Reich; A. LeBon; H-T.C. Pedro. On a cellular division model for multi-disciplinary optimization. In *51st AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, Orlando, Florida*, 2010.

[6] M. H. Kobayashi; H-T. C. Pedro; R. M. Kolonay and G.W. Reich. On a cellular division method for topology optimization. In *50th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, Palm Springs, California*, 2009.

[7] H-T. C. Pedro and M. H. Kobayashi. Optimization of cellular structures using map l-systems. In *47th AIAA Aerospace Sciences Meeting and Exhibit Orlando, Florida*, 2009.

[8] E. Sabbatinii; G.-M. Revel; M. H. Kobayashi. On a biologically inspired topology optimization method for vibration suppression. In *47th AIAA Aerospace Sciences Meeting and Exhibit Orlando, Florida*, 2009.

[9] M. H. Kobayashi; H. T. C. Pedro; C. F. M. Coimbra and A. K. da Silva. The formal evolutionary development of low entropy dendritic thermal systems. In *11th IEEE International Conference on Computational Science and Engineering (CSE-08), paper SEC08-194, S. Paulo, Brazil*, 2008.

[10] Hugo T. C. Pedro. *On Biologically Inspired Designs and Methods.* PhD thesis, University of Hawaii at Manoa, August 2010.

[11] Przemyslav Prusinkiewicz and Aristid Lindenmayer. *The Algorithmic Beauty of Plants.* Springer Verlag, New York, 2004.

[12] A. G. M. Michell. The limits of economy of material in frame-structures. *Philosophical Magazine*, 8:589–597, 1904.

[13] Martin P. Bendsøe and Noboru Kikuchi. Generating optimal topologies in optimal design using a homogenization method. *Computer Methods in Applied Mechanics and Engineering*, 17:197—224, 1988.

[14] Katsuyuki Suzuki and Noboru Kikuchi. A homogenization method for shape and topology optimization. *Computer Methods in Applied Mechanics and Engineering*, 93:291—318, 1991.

[15] U. Schramm and M. Zhou. Recent developments in the commercial implementation of topology optimization. *IUTAM Symposium on Topological Design Optimization of Structures, Machines and Materials*, 137:239–247, 2006.

[16] M. P. Bendse. Optimal shape design as a material distribution problem. *Structural and multidisciplinary optimization*, 1:193–202, 1989.

[17] Martin P. Bendsøe and Ole Sigmund. *Topology Optimization: Theory, Methods and Applications*. Springer, New York, second edition, 2003.

[18] G. I. N. Rozvany. Aims, scope, methods, history and unified terminology of computer-aided topology optimization in structural mechanics. *Structural and Multidisciplinary Optimization*, 21, 2001.

[19] G. Rozvany. A critical review of established methods of structural topology optimization. *Structural and Multidisciplinary Optimization*, 37(3):217–239, 2009.

[20] Zhou M. and Rozvany G. On the validity of eso type methods in topology optimization. *Structural and Multidisciplinary Optimization*, 21:80–83, 2001.

[21] Q. Q. Ling and Steven G. P. A performance-based optimization method for topology design of continuum structures with mean compliance. *Computer methods in applied mechanics and engineering*, 191:1471–1489, 2002.

[22] C. S. Edwards, H. A. Kim, and C. J. Budd. An evaluative study on eso and simp for optimising a cantilever tie-beam. *Structural and Multidisciplinary Optimization*, 34:403–414, 2007.

[23] Q. Q. Liang, Y. M. Xie, and G. P. Steven. Optimal selection of topologies for the minimum-weight design of continuum structures with stress constraints. *Proceedings of the Institution of Mechanical Engineers*, 213(8):755–762, 1999.

[24] O. Sigmund. On the design of compliant mechanisms using topology optimization. *Mechanics of structures and machines*, 25(4):493–524, 1997.

[25] M. Dorigo; C. Blum. Ant colony optimization theory: A survey. *Theoretical Computer Science*, 344:243–278, 2005.

[26] J. Kennedy; R Eberhart. Particle swarm optimization. *In Proceedings of the IEEE International Conference on Neural Networks*, 1995.

[27] J. H. Holland. *Adaptation in Natural and Artificial Systems.* University of Michigan Press, 1975.

[28] Kalyanmoy Deb. *Multi-Objective Optimization using Evolutionary Algorithms.* John Wiley & Sons, Ltd, Chichester, UK, 2001.

[29] C. D. Chapman; K. Saitou; M. J. Jakiela. Genetic algorithms as an approach to configuration and topology design. *Journal of mechanical design*, 116(4):1005–1012, 1994.

[30] Aristid Lindenmayer. Mathematical models for cellular interactions in development i. filaments with one-sided inputs. *Journal of Theoretical Biology*, 18:280—299, 1968.

[31] Aristid Lindenmayer. Mathematical models for cellular interactions in development ii. simple and branching flaments with two-sided inputs. *Journal of Theoretical Biology*, 18:300—315, 1968.

[32] Grzegorz Rozenberg and Arto Salomaa. *Handbook of Formal Languages.* Springer Verlag, 1997.

[33] Stelios Manousakis. Musical l-systems. Master's thesis - sonology, The Royal Conservatory, The Hague, June 2006.

[34] Grzegorz Rozenberg and Arto Salomaa. *The mathematical theory of L systems.* Academic Press, New York, 1980.

[35] Przemyslav Prusinkiewicz. Graphical applications of l systems. In *Proceedings of Graphics Interface '86 — Vision Interface '86*, pages 247—253, 1986.

[36] A. Lindenmayer and G. Rozenberg. *Graph Grammars and Their Applications to Computer Science*, chapter Parallel Generation of Maps: Developmental Systems for Cell Layers, pages 301—316. Lecture Notes in Computer Science. Springer, Berln, 1979.

[37] A. Nakamura, A. Lindenmayer, and K. Aizawa. *The Book of L*, chapter Some Systems for Map Generation, pages 323—332. Springer, Berln, 1986.

[38] W. T. Tutte. *Graph Theory.* Addison-Wesly, Reading, Massachusetts, 1982.

[39] V. K. Balakrishnan. *Graph Theory.* McGraw-Hill, 1 edition, February 1997.

[40] Mark de Berg; Otfried Cheong; Marc van Kreveld; Mark Overmars. *Computational Geometry - Algorithms and Applications.* Springer, third edition, 2008.

[41] Alexandre Ern and Jean-Luc Guermond. *Theory and Practice of Finite Elements.* Springer Verlag, Berlin, D, 2004.

[42] O. C. Zienkiewicz, R. L. Taylor, and J. Z. Zhu. *The Finite Element Method: Its Basis and Fundamentals.* Elsevier Butterworth-Heinemann, sixth edition, 2005.

[43] O. C. Zienkiewicz and R. L. Taylor. *The Finite Element Method: For Solid and Structural Mechanics.* Elsevier Butterworth-Heinemann, sixth edition, 2005.

[44] Comsol. Comsol multiphysics scripting guide v3.5, September 2008.