Quarterly Report 2

by

Naudé Thomas Conradie

Thesis presented in partial fulfilment of the requirements for the degree of Master of Engineering (Mechatronic) in the Faculty of Engineering at Stellenbosch University

Supervisor: Dr. M. P. Venter

February 2020

Declaration

By submitting this thesis electronically, I declare that the entirety of the work contained therein is my own, original work, that I am the sole author thereof (save to the extent explicitly otherwise stated), that reproduction and publication thereof by Stellenbosch University will not infringe any third party rights and that I have not previously in its entirety or in part submitted it for obtaining any qualification.

Date: 2020/05/29 .....

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This report serves to describe the work done on my thesis project in the second quarter of the year 2020. This report was made using the Stellenbosch University Thesis LaTeX template, as a large portion of the material will be used in or adapted for the final report.

This quarterly report consists of updated introduction literature review, material testing and software sections.

1.1 Background

Manufacturing capabilities have greatly increased over the past decades with the advent of three dimensional (3D) printing and other advanced manufacturing technologies. Development of new products and systems may now be limited by design capabilities. Improving and creating new design methods may lead to innovative designs not yet seen before [4].

The field of soft robotics is particularly suited for creative and novel approaches to the design of components. Soft robotic geometries are often highly complex and free-form. Soft robotic actuators may move in imprecise and non-trivial manners [5]. Materials used in the construction of soft robotic components may behave with non-linear responses[6]. Novel soft robotic actuator designs with new and non-trivial behaviours are actively being created and implemented [2].

A computationally efficient generative design approach may be used to explore the design space of soft robotics. Generative design is a powerful automated approach to design that evaluates the performance of many different potential design solutions. Generative design is useful when performing manual design evaluations may become tedious or impossible within realistic time constraints [7].

1.2 Aim And Objectives

The aim of this research is to implement a generative design process to construct basic units and soft bodies constructed from these units. The design process shall be set up in order to accommodate implementation of multiple objective functions for soft bodies. The design process shall facilitate further development of the software.

Chapter 2

Literature Review

2.1 Soft Robotics

The field of soft robotics is relatively new and developmental. Soft robots inherently differ from traditional robots. They are constructed from compliant and pliable materials. Soft robots generally have fixed joints and locomotive actuators between joints, as opposed to traditional robots which usually have locomotive actuated joints connected by rigid sections. [5]

Soft robots offer many advantages over traditional robots. They provide a safer working environment around humans and with fragile materials, as they are much lighter, more pliable and move slower and with less force than traditional robots. They require lower tolerances for manufacture due to their inherent less precise nature. They are much more lightweight compared to traditional robots. [5]

2.1.1 Actuators

Actuators are components that cause controlled motion, generally used in robotics and machinery [8]. There are currently a few major types of soft robotic actuators in use [6].

2.1.1.1 Actuator Types

Shape Memory Alloys (SMA) are metallic alloys capable of being formed into a specific shape while above an inherent transformation temperature, as well as being formed into another shape below the transformation temperature. When the material is then heated or cooled above or below the transformation temperature, it reforms into those respective shapes. This property exists due to the transition between the martensite phase of the material below the transformation temperature, and the austenite phase above the transformation temperature. SMA actuators are heated by applying a current directly to the material. SMA actuators are small, lightweight, silent, and have a high force-

to-weight ratio. When shaped straight, they can exert high forces, but only achieve small displacements relative to their length. When coiled, they can extend more, but exert smaller forces. [9]

Shape Memory Polymers (SMP) are similar to SMAs, consisting of smart polymers with the same shape memory properties, instead of metallic alloys. The initial is shape is determined during the manufacturing process. The transformed shape is obtained by cooling the SMP and shaping it as desired. SMPs use electricity or light as a heat source for transformation [10]. They have a high deformation capacity and shape recovery. They are lighter, cheaper and easier to produce than SMAs. They are limited in size due to their low recovery stresses [11; 10].

Dieelectric/Electrically-Actuated Polymers (DEAP) consist of layers of polymers interspersed with conductive material. When the conductive material receives an electrical input, a chemical reaction occurs that causes a change in volume across the layers. This causes the layers to bend in a predetermined direction. DEAPs are lightweight, silent and use little energy. They are biocompatible and functional in water. They are well-suited to mimicking real muscles. Their reactions under high voltages are not fully understood yet and accurately modelling them is highly complicated. [12]

Electro-Magnetic Actuators (EMA) make use of magnetic microparticles within a polymer matrix. The particles are manipulated to cause motion by an external magnetic field from an electromagnet. This allows for a wide range of motion by varying the orientation and magnitude of the electromagnetic field. EMAs are small, require low voltages and are efficient. They have quick response times and high dynamic ranges. They are still an emerging technology in the early stages of development. [13]

Fluid Elastomeric Actuators (FEA) use soft polymeric structures with internal geometry designed for specific types of motion when driven by fluid pressure. Fluid pressure may be obtained from pressurized containers or chemical reactions. They are simple to design, manufacture and control, and are lightweight and usually inexpensive. They are scalable to different sizes and resistant to many types of damage. [14; 15]

2.1.1.2 Actuator Shapes

Multiple types of soft robotic actuators implement linear extension. This allows for an increase in reach and activation of levers or switches. Torsional extension is a variant of linear extension. Torsionally extending actuators twist as they extend, resulting in an angular difference between the ends of the actuator. [5]

FEAs can be built to curl while contracting and straighten out while expanding, similar to natural muscles. This has a range of applications, especially when multiple of these FEAs are used in conjunction with one another. One application is as a gripper, where a number of these FEAs are arranged sim-

ilarly to a hand or tentacles all curling inward. Grippers are well-suited to picking up and manipulating soft and/or irregularly shaped objects. [5]

Non-trivial actuators are manufacturable. A bimodal FEA was developed by Ellis in 2020. The actuator has a crimped paper strip acting as a strain limiter embedded within a stiff layer of Ecoflex 0030. The stiff layer is alongside multiple Mold-Star 15 cells. When pressurised by a linearly increasing single pressure source, the actuator first curls inwards, then back outwards while extending linearly. [2]

2.2 Soft Robot Modeling

To prevent having to physically construct every design of a soft body, a computationally efficient and accurate digital model can be constructed.

2.2.1 Modeling Approaches

The Finite Element Method (FEM) is an approach to numerically solving field problems. Field problems require the determination of one or more dependent variables' distribution in space. Field problems are mathematically described with differential equations or integral expressions. Finite elements can be expressed as small parts of a larger body. A field quantity within an element may only have a simple spatial variation, such as being described by polynomial terms no higher than the second order. FEM differs from calculus as calculus uses infinitesimal elements. FEM thus delivers approximate solutions. [3]

Voxels are three dimensional (3D) pixels. Voxels are usually cubical. Realistic physical properties may be applied to voxels in specific software packages, such as the free VoxCAD software. Bodies may be constructed and undergo deformation when constructed from voxels in these software packages. It is relatively simple to construct bodies from voxels. [16; 17]

The Gaussian Mixtures representation uses a density field analogy to a level-set method. The density field is initialised as having zero density. Points of density with Gaussian falloff are added within the field. The points may have positive or negative weights. Positive weights contribute to the density, while negative weights subtract from it. If a single point was used, a solid sphere would be the thresholded result.

2.2.2 Modeling Software

Several commercial software packages capable of realistically modeling soft bodies are available.

LSDyna is a FEM software package widely used in industry. It is owned by ANSYS and maintained by LST. The software's code is based on highly non-linear and transient dynamic FEM with explicit time integration [18].

Siemens NX 12 is an integrated software package capable of performing FEM analysis. The software package has a user-friendly interface for graphical design of components [19].

MSC.Marc Mentat is a pre- and postprocessing software for the MSC.Marc FEM solver. It is focused on nonlinear material modeling and analysis. It has an extensive set of options available for post-processing [20].

2.2.3 Material Modeling

The modeling of materials undergoing relatively large deformations is an active research field. Stored strain energy density may be used to compute stress in hyperelastic materials. The strain energy density is defined using invariants of strain. The three invariants are given as

$$I_1 = \lambda_1^2 + \lambda_2^2 + \lambda_3^2 \tag{2.1}$$

$$I_2 = \lambda_1^2\lambda_2^2 + \lambda_2^2\lambda_3^2 + \lambda_3^2\lambda_1^2 \tag{2.2}$$

and

$$I_3 = \lambda_1^2\lambda_2^2\lambda_3^2 \tag{2.3}$$

where λ_1^2, λ_2^2, and λ_3^2 are three eigenvalues. The undeformed state is used as the frame of reference. The three invariants will not change when using different coordinate systems. The three invariants must be positive for the deformation to be valid. The square root of I_3 measures the volume change of the material. $I_3 = 1$ if the material is incompressible [21].

The distortional strain energy density is defined as

$$W_1(I_1, I_2) = \sum_{m+n=1}^{\infty} A_{mn} (I_1 - 3)^m (I_2 - 3)^n \tag{2.4}$$

The Ogden model uses the principal stretches to define the distortional strain energy density as

$$W_1(\lambda_1, \lambda_2, \lambda_3) = \sum_{i=1}^{N} \frac{\mu_i}{\alpha_i} (\lambda_1^{\alpha_i} + \lambda_2^{\alpha_i} + \lambda_3^{\alpha_i}) \tag{2.5}$$

where N, μ_i, and α_i are material parameters. N is usually three. The principal stretches are the three eigenvalues of the deformation gradient. If the material is incompressible, the three principal stresses are not independent, meaning $\lambda_1\lambda_2\lambda_3 = 1$. The shear modulus is

$$\mu = \frac{1}{2} \sum_{i=1}^{N} \alpha_i \mu_i \tag{2.6}$$

The Ogden model correlates well with simple tension test data that is elongated up to 700%. The model accommodates for slightly compressible behaviour and a nonconstant shear modulus [21].

2.3 Evolved Virtual Bodies

Optimization of discretely represented mathematical model problems, such as virtual bodies, is an active area of research.

2.3.1 Generative Design

An early approach to generative design used a model that encompassed the domain of all components and their possible interactions to design physical bodies with desired behaviours. This approach divided the behaviours and constructed appropriate design fragments for them. These fragments were incrementally composed until a design that obeyed the desired behaviours was obtained. [7]

A generative encoding is a type of encoding that specifies the construction of a phenotype. A genotype is a programmed representation of a potential individual or problem solution. A phenotype is a set of characteristics of an individual resulting from the composite of its genotypes [22]. It may scale well because of its inherent self-similar and hierarchical structure. [23]

Data base amplification is the generation of seemingly complex objects from very concise descriptions [24].

2.3.1.1 Evolutionary Algorithms

Evolutionary algorithms, also known as genetic algorithms, are a robust approach to solving optimization problems. They derive their name from their similarity to the evolution of biological organisms. They typically start with a randomly generated population of solutions to a given problem. Each individual solution's suitability is checked. The more suitable solutions are used to generate a new population, by "breeding" existing members of the population, and introducing some random variations. The new population is checked again, and this process is repeated for some number of generations [25]. Evolutionary algorithms are versatile. They can be applied to the optimization of functions and the evolution of complex behaviours and bodily morphologies. They have been specifically applied to the evolution of robotic bodies for many years [22; 26].

Evolutionary algorithms require a measure of the population's performance. Within the context of evolving simulated physical bodies, a realistic physically simulated goal is usually set. Examples include traversing the greatest distance within a set amount of time, jumping or climbing over an obstacle, or drawing

another object closer to it. Fitness measures may also be implemented as survival criteria in testing, such as energy requirements, size, and complexity of the respective bodies. [22; 26]

Evolutionary algorithms typically use direct encodings of solutions. They may struggle to successfully design highly complex systems using direct encodings. [23]

2.3.1.2 Lindenmayer Systems (L-systems)

L-systems were originally conceived as a mathematical theory of plant development. They are used in theoretical biology to describe and simulate natural growth processes. They did not originally include enough detail to completely model higher-level plants. They focused on plant topology and not geometry. [27; 24]

The main component of L-systems is rewriting. Rewriting is used to define complex objects by successively replacing parts (letters) of an initial, simple object (word) according to a set of rules (grammar). Grammars are applied in parallel and simultaneously replace all letters in a given word [24]. This makes L-systems suitable for describing and generating fractal structures. Words generated by L-systems can be used as genotypes for virtual bodies [27].

Growth functions describe the number of letters in a word in terms of its derivation length. Growth functions of DOL-systems are independent of the order of the letters in a word and its derived words. [24]

There are many variations of L-systems. Deterministic and context-free L-systems (DOL-systems) use edge rewriting to replace polygon edges with figures and node rewriting to operate on polygon vertices. Stochastic L-systems implement randomization to obtain variation in productions. A context-sensitive L-system's productions' expression may depend on its predecessors' context. [24]

Partial L-systems use the notation of non-deterministic context-free L-systems (OL-systems) to define the different possible structures of a given type that can develop. They capture the main traits that characterise a structural type and provide a formal basis for their classification. [24]

Original L-systems are discrete in time and space. Model states are known only at specific time intervals and only a finite number exist. Parametric L-systems allow for infinite model states due to the assignments of continuous attributes to model components. Parametric L-systems are not limited by all values being reduced to integer multiples of a unit segment. [24]

Map L-systems allow for the formation of cycles in a production. Maps are finite sets of regions. Regions are surrounded by boundaries consisting of finite, circular sequences of edges meeting at vertices. Each edge has one or two vertices associated with it (only one if the edge forms a loop). Edges cannot cross without forming a vertex. There are no vertices not associated

with an edge. Every edge is part of the boundary of a region. The set of all edges is connected. [24]

Some simulations of the branching patterns often achieved by L-systems consider the interactions among the growing features, structures and environment. This makes models more realistic and introduces some complexity. [24]

2.3.1.3 Compositional Pattern Producing Networks (CPPN)

define and discuss CPPNs

2.3.1.4 The Monte Carlo Method

The Monte Carlo method may be applied to physical problems described by systems with moderate numbers of parts. The method involves producing large numbers of results of a given problem. The relative number of successful iterations may be analysed. The method will never yield results with complete certainty. The method may yield results within certain limits with great probability. The method is well suited to generative design problems of which individual iterations are not too computationally expensive [28].

2.3.1.5 Emergent Properties

Emergent properties occur when not all components of a given property satisfy that property. An emergent property is not satisfied by the constituent components of a system, but is satisfied by the overall system. If the required condition for a specific property to exist can be determined, it is possible to construct a system satisfying that property from components that do not satisfy that property. If the target property of a system and the property of a component is known, it can be determined if the other component can have a property that will result in the system satisfying the target property. If that property exists, it can be found. [29]

Reactive systems consist of interconnected sub-components that are a part of structural links defining communication methods. These systems may exhibit emergent properties that are unpredictable even when complete knowledge of the systems is accessible. This implies the systems are complex in such a manner that they cannot be simplified to rules based on inferences from their properties. Knowledge of the rules of interactions between the sub-components is also necessary. [30]

Emergent properties are sometimes encountered with generative design. Emergent properties may be complex behaviours that are difficult to predict [30] and challenging to understand initially, that arise from the combination of the simple elements and rules used to construct the generative design algorithms. For example, virtually evolved bodies may end up being complexly constructed in such a way that the methods of completing their objectives are

not initially obvious [31]. These emergent properties are desirable, as one advantage of generative design processes is that they may arrive at original and unique designs that may be extremely difficult for a human to arrive at [22].

2.3.2 Previous Work

In 1994, Karl Sims introduced the concept of evolving three dimensional virtual bodies with the aid of evolutionary algorithms. Four simulations with different fitness evaluation functions were run. The four functions were swimming as far as possible within a limited time period, moving across a flat surface as far as possible within a limited time period, jumping as high as possible from a stationary position, and following a light source. The bodies were simply modelled. Nodes describe the rigid parts of the body, the joints between parts and their parent parts, and the set of connections they have to other nodes. Rigid parts have specific dimension. Joints have different types and limits of motion. Node connections describe a part's relationship to its parent. The bodies are described as a set of nodes. These bodies are evolved in three dimensions according to their performance against one of the simulation fitness functions using evolutionary algorithms for a number of generations. The best-performing models were inspected at the end [22]. Further tests were done where two bodies were evolved in the same environment. The bodies had to compete with each other to keep a cube as close to themselves and as far away from the other body as possible. This study investigated the effect of competition between organisms on evolution and optimisation [26].

In 2012, Rieffel et al applied evolutionary algorithms to soft bodies. They used NVidia's PhysX engine to model soft tetrahedral meshes. The physics engine allows for the manipulation of a material's stiffness and damping coefficient. Three properties of the bodies were used to control the evolution. The properties were the body shape, body motion, and material properties. Three different simulations were run. For each one, one of the three properties were fixed, while the other two were allowed to evolve. The mesh resolution was also manipulated. Higher resolutions allowed for smoother surfaces at the cost of computing power. The final bodies are 3D printable, but lack a simple actuation mechanism. [32]

Also in 2012, Hiller and Lipson modelled soft amorphous bodies using the Gaussian mixtures representation. Gaussian points are listed in a 3D workspace. Densities and falloff ranges are associated with the points. This results in smooth, free-form shapes. Material properties are stored in the points and distributed accordingly between points. The Gaussian mixtures representation is a computationally efficient method of storing and representing complex smooth shapes. Shapes were evolved using evolutionary algorithms. The Gaussian points' coordinates, densities, falloff ranges and material indices were used as parameters of interest during evolution.[33]

In 2013, Cheney et al used voxels to model soft bodies in VoxCAD. VoxCAD is free voxel-modelling software. Soft bodies were evolved using CPPN-NEAT. CPPN morphologies were found to appear natural and produce interesting and varying results. Three materials were used for the construction of the soft bodies. The three materials differed in hardness, being compliant, partially compliant and stiff. Compliant and partially compliant voxels acted as the actuators. Soft bodies were tasked either with traversing along a linear path or squeezing through a tight space. Simulations were run multiple times with different penalty functions. Penalty functions included costs for actuated voxels, voxel connections and the total number of voxels. It was found that with differing penalty functions, different bodies evolved and performed better. Final bodies are 3D printable. They are actuated by placing them within a pressure chamber where the pressure is sinusoidally varied. This limits their practical application. [16; 17]

Chapter 3

Material Testing

3.1 Motivation

3.1.1 Material

Mold Star 15 SLOW is selected as the material to be utilised in this project. Mold Star 15 is selected because of its availability and properties. Mold Star 15 is deliverable to the premises where testing is to be done and available from a registered supplier. Additional properties are listed in Table 3.1 below.

Material	Cost/kg	Pot life (min)	Cure time (hr)	Tensile strength (MPa)
Mold Star 15 SLOW	R332.00	50	4	2.7579

Table 3.1: Mold Star 15 Given Properties [1]

Mold Star 15's long pot life allows for adequate time to prepare specimens thoroughly.

3.1.2 Testing

The necessary material properties to accurately model Mold Star 15 are not available from the supplier. Testing is done according to ISO 37 and ISO 7743 standards for tensile and compression testing respectively to obtain data to construct an accurate Ogden model for the material [34; 35].

3.2 Testing Procedure

3.2.1 Specimen Preparation

Mold Star 15 requires a specific preparation process to ensure the consistency of the material properties and the usability of the product.

A clean workspace and tools are required for the preparation of the material. Dirt and other materials may cause impurities in the material. Impurities will result in deviations from standard material behaviour. The workspace and tools are wiped down thoroughly with tissue paper. Moulds are sprayed with a non-stick spray. Nitrile gloves are worn. Latex gloves are not appropriate as they may react with the material.

Mold Star 15 consists of two components, Mold Star 15 A and Mold Star 15 B. The two components need to be mixed according to a 1:1 ratio. A clean, empty mixing bowl is placed on an electronic scale. The scale is zeroed. An appropriate mass of Mold Star 15 A is poured into the mixing bowl. The mass added is noted. The scale is zeroed again. A matching mass of Mold Star 15 B is poured into the mixing bowl. Once both materials have been added together, the pot life of 45 minutes starts.

The materials are mixed using a wooden tongue depressor. The materials are mixed until the colour is consistent throughout and no streaks are visible.

The mixed material is degassed until 1 minute after no more bubbles appear. The degassed material is poured into selected moulds. The material is degassed until 1 minute after no more bubbles appear again. Excess material is carefully and slowly removed from the moulds using scrapers or other flat materials. This should be completed before the pot life of 45 minutes has ended.

The material takes 4 hours to set. The material may be baked as well for shorter periods of time. Baking was not used in order to keep material properties consistent and because time constraints were not relevant to material preparation. Set materials may be removed from moulds.

3.2.2 Specimen Testing

Chapter 4

Software

4.1 Motivation

4.1.1 Software

MSC.Marc Mentat 2019 is used to construct and define the templates, units and bodies. MSC.Marc Mentat 2019 is used to model the materials and their behaviour. Marc is capable of advanced non-linear analysis.

Python 3.6 is used to construct a modelling pipeline. Packages are available that allow for the integration of Python and Marc. Many advanced numerical analysis packages are also available for Python.

4.1.2 Materials

Mold Star 15 SLOW is selected as the material to be digitally modelled for the purposes of the project. Mold Star 15 has a hyper-elastic non-linear response. It is suitable for inflation while being capable of supporting itself at the relevant scale of construction.

4.1.3 Assumptions

A 2D modeling approach is followed. 2D modeling reduces complexity and computational requirements. 2D modeling is analogous to real-world behaviour under appropriate circumstances [2; 36].

A complete unit boundary of varying thickness is always present. External elements and their neighbours for a specified range inwards are always excluded from being removed during unit generation. A complete unit boundary prevents any unwanted gaps when units are used to construct bodies. A complete unit boundary simplifies the process of applying boundary conditions.

Mesh refinement during a simulation is not possible with the current software architecture. The construction of the units also defines the element and node indices and coordinates. The indices and coordinates are referenced and

used throughout the simulation and data retrieval. Mesh refinement would change the indices and coordinates, rendering the software incapable of interpreting the simulations. More accurate results may be achieved by running simulations at higher element resolutions. Higher element resolutions will result in different unit configurations.

Free-floating elements are eliminated during unit generation. Therefore fewer unique combinations exist than an *nChooser* calculation would arrive at. It is not possible to determine exactly how many unique possible combinations exist without generating all of them. This unnecessary and computationally expensive. The result of the *nChooser* calculation is used as an estimate of the number of possible unique combinations.

Material properties for Mold Star 15 are initially obtained from previous work [2]. Due to the COVID-19 pandemic, testing facilities were not available at an appropriate time to perform material testing before software implementation. Material testing is intended to occur before the completion of the project to verify the material properties used.

4.2 Method

Figure 4.1 roughly illustrates the current software pipeline. Elaborative figures are found in Appendix A.

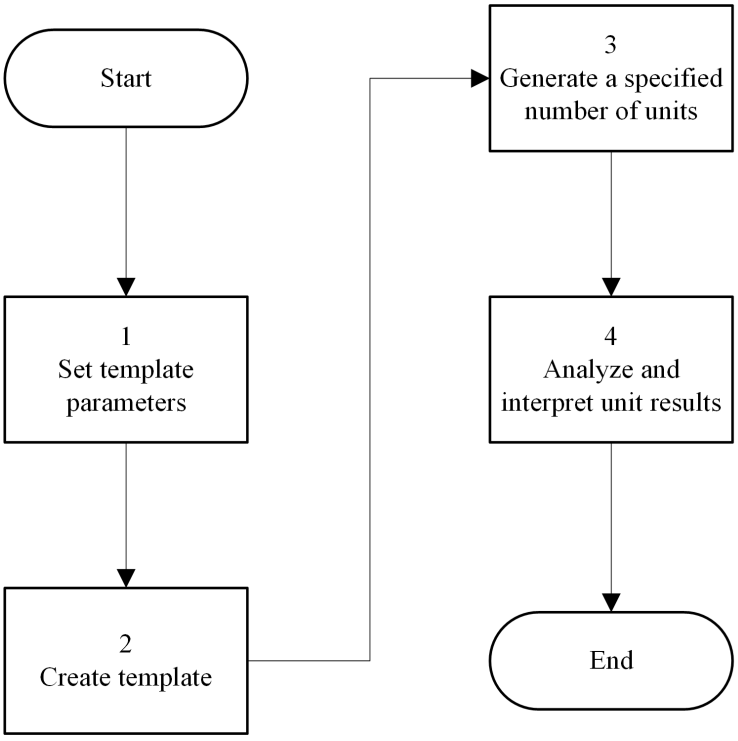


Figure 4.1: Software Pipeline

4.2.1 File Management

File management is an important component of the software structure. A large number of files are generated during simulations. Each model file has a number of files associated with it. Many model files may be generated during a simulation. Every simulation with new parameters requires new addresses and forms of identification. Figure 4.2 illustrates the file hierarchy implemented in the software. Dotted lines indicate components that are still to be implemented. Table 4.1 serves as a key for Figure 4.2.

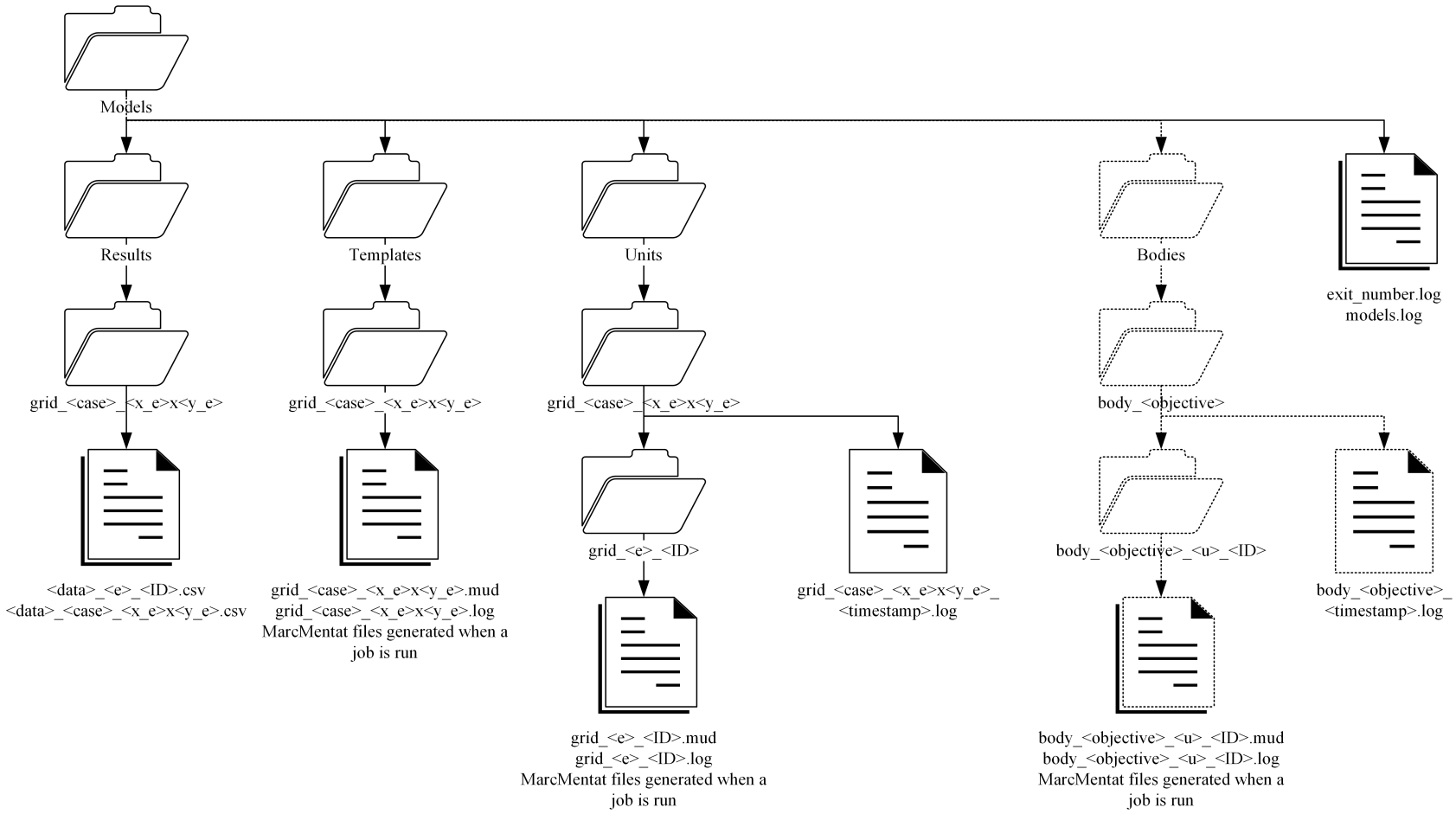


Figure 4.2: File Hierarchy

Name	Description	Example
case	The unit case identifier	1
x_e	The number of elements in the x-direction	5
y_e	The number of elements in the y-direction	5
data	The type of data stored	Reaction Force
e	The number of elements removed	5
ID	The unique unit ID	4121ab03c09f703b eb534ab74aa3c079
timestamp	The timestamp of the start of the simulation	2020-05-20 –15-20-55
objective	The body objective identifier	A
u	The number of units used to construct the body	10

Table 4.1: File Hierarchy Key

The software code is structured as a Python library. Python libraries allow for intuitive organisation and hierarchical construction of code. Relevant functions are grouped together. Sub-grouping is possible where required. Figure 4.3 illustrates the Python library structure implemented for this project's code. Table 4.2 describes the relevant function libraries.

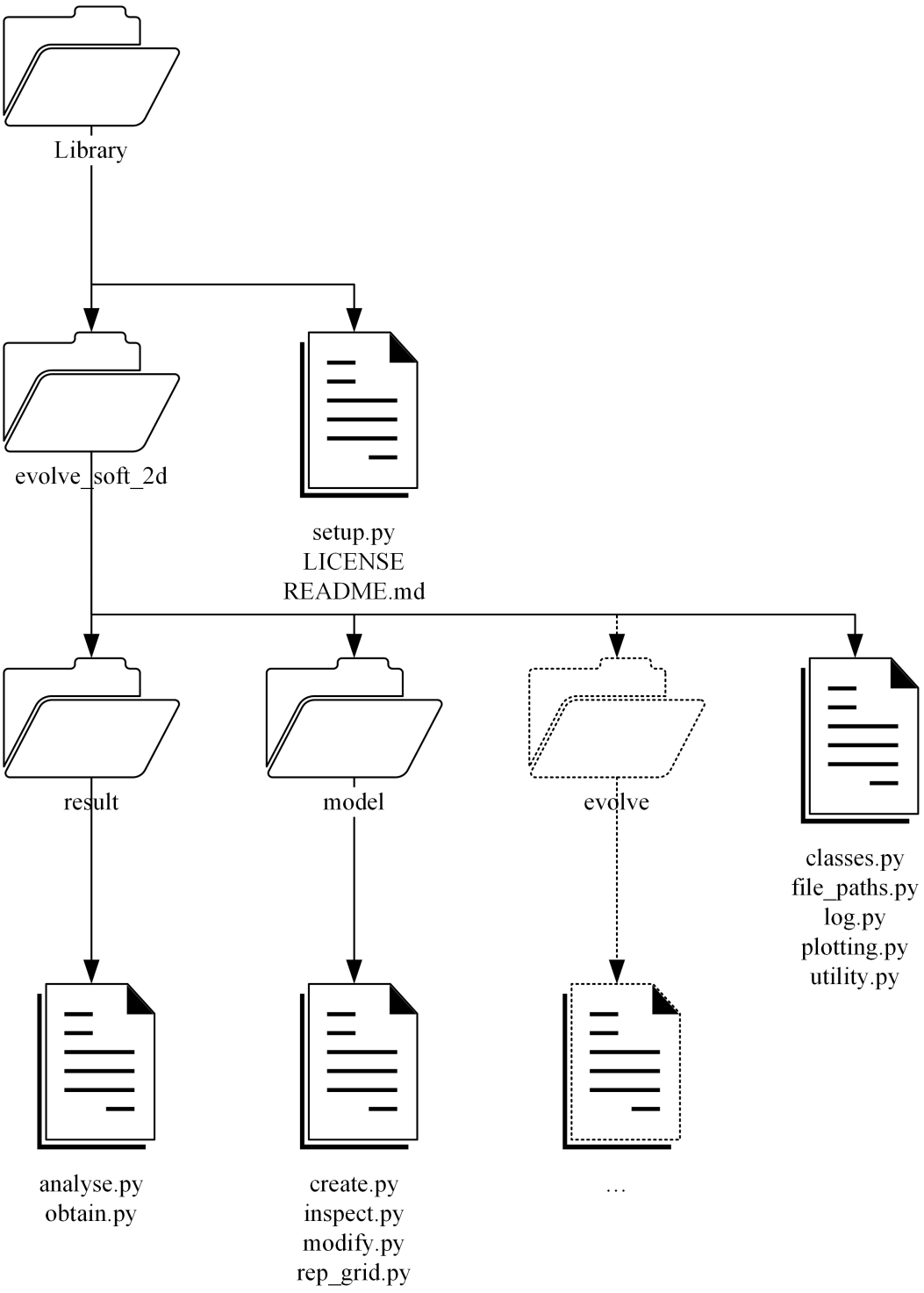


Figure 4.3: Python Library

Name	Description
evolve_soft_2d	The main library folder
result	The folder containing functions related to obtaining and analysing results
model	The folder containing functions related to creating and modifying models
evolve	The folder containing functions related to evolving models
classes.py	The object classes, their related functions, and example material models
file_paths.py	The file path generative functions and base file paths
log.py	The logging formatting and functions
plotting.py	The plotting functions
utility.py	Utility functions for multiple purposes
analyse.py	Result analysis functions
obtain.py	Result obtainment functions
create.py	Simulation and template creation functions
inspect.py	Model inspection functions
modify.py	Model creation and modification functions
rep_grid.py	The representative grid creation and modification functions

Table 4.2: Python Library Description

4.2.2 Material Properties

An Ogden material model of Mold Star 15 is used. Material model parameters are obtained from [2]. Material model parameters are shown in Table 4.3 and applied to Equation 2.5.

Parameter	1	2	3
μ	-6.50266e-06	0.216863	0.00137158
α	-21.322	1.1797	4.88396

Table 4.3: Mold Star 15 Ogden Parameters [2]

4.2.3 Units

Certain template parameters are required to be provided by the user. Other template parameters are calculated and/or generated when the template class object is defined. Relevant template parameters and example parameters are outlined in Table 4.4.

Parameter	**Data Type**	**Description**	**Source**	**Example**
case	int	The unit case identifier	User-defined	1
x0	int	The initial x-coordinate	User-defined	0
y0	int	The initial y-coordinate	User-defined	0
x_n	int	The number of nodes in the x-direction	User-defined	6
y_n	int	The number of nodes in the y-direction	User-defined	6
ogd_mat	ogd_mat	The Ogden material model	User-defined	mold_star_15
n_steps	int	The number of steps in the second of the simulation	User-defined	4
tab_nam	str	The name of the table containing the function of the load to be applied	User-defined	ramp_input
apply	float	The conditions to be applied to the unit template	User-defined	$\frac{y_n-1}{2}$
run_success	bool	The success of the unit template's run	Obtained	True
c_e	float	The constraint energy of the unit template	Obtained	6462952.804663594
i_e	float	The internal energy of the unit template	Obtained	1505641.7999267578
n_n	int	The total number of nodes	$x_n \times y_n$	36
x_e	int	The number of elements in the x-direction	$x_n - 1$	5
y_e	int	The number of elements in the y-direction	$y_n - 1$	5
n_e	int	The total number of elements	$x_e \times y_e$	25
n_e_l	str	The total number of elements as a string label	Obtained	5x5
e_internal	list	The list of internal elements	Obtained	[7, 8, 9, 12, 13, 14, 17, 18, 19]
n_external	list	The list of external nodes	Obtained	[1, 2, 3, 4, 5, 6, 7, 12, 13, 18, 19, 24, 25, 30, 31, 32, 33, 34, 35, 36]
grid	list	The representative grid of ones	Obtained	[[1, 1, 1, 1, 1], [1, 1, 1, 1, 1], [1, 1, 1, 1, 1], [1, 1, 1, 1, 1], [1, 1, 1, 1, 1]]
fp_t_...	str	The various file paths associated with the template	Obtained	C:...\Templates \grid_1_5x5 \grid_1_5x5.mud

Table 4.4: Template Class Object Parameters

A unit template is generated in Marc Mentat according to the template parameters. Units are 2D meshes constructed from square elements. A template is a solid grid with all boundary conditions and other properties defined. Figure 4.4 illustrates an example template.

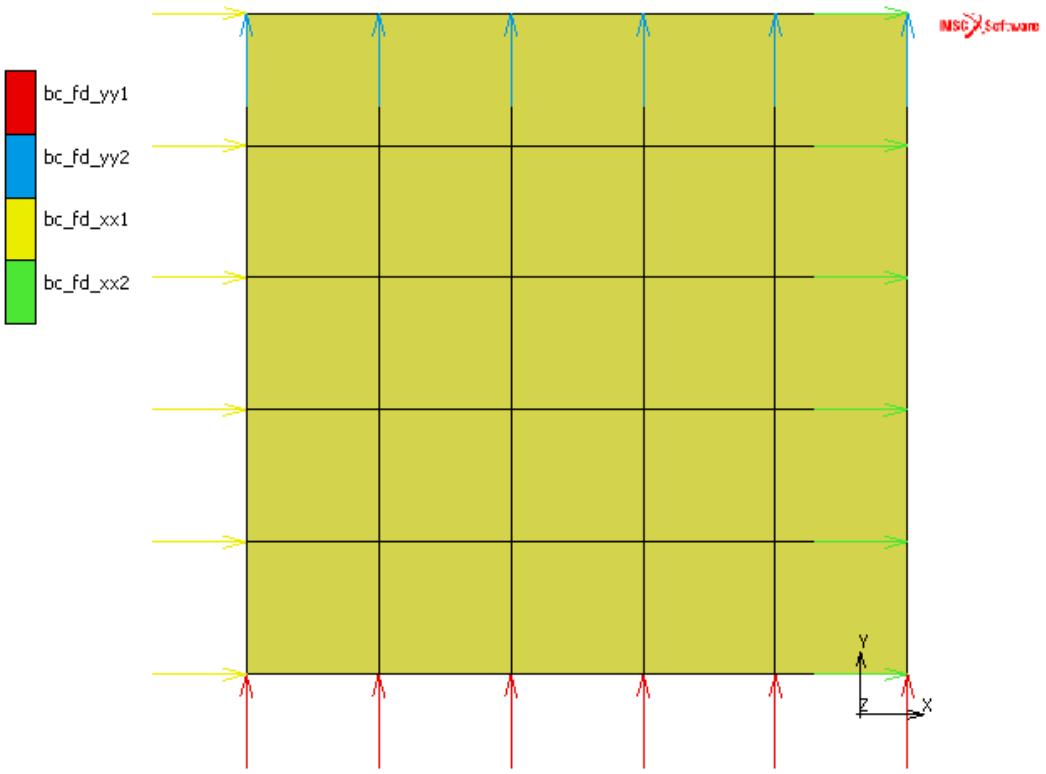
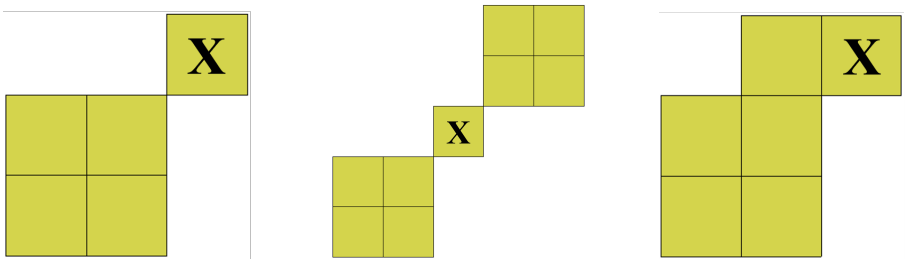


Figure 4.4: Unit Template For Case 1

A specified number of unique units are generated, run and analysed. A check is performed to determine that no more unique units are requested than theoretically possible. Units are defined by removing random internal elements. Free-floating elements are identified and removed. Free-floating elements are defined as any elements that do not share at least two node connections with a single other node. Figure 4.5 illustrates some examples.



(a) Element X is recognised as free-floating and will be removed.

(b) Element X is recognised as free-floating and will be removed.

(c) Element X is not recognised as free-floating and will remain.

Figure 4.5: Free-Floating Element Examples

The number of elements removed and the element IDs of the removed elements are used to generate a unique ID for each unit. Unit parameters are saved. Relevant unit parameters and example parameters are outlined in Table 4.5. Units are run and evaluated. Displacement, reaction force and strain energy density data are extracted from each unit and used to evaluate them.

Parameter	**Data Type**	**Description**	**Source**	**Example**
template	template	The unit template parameters	User-defined	temp_1
rem	list	The list of elements removed from the unit	Obtained	[7, 8, 13, 18, 19]
grid	list	The representative grid with the elements removed	Obtained	[[1, 1, 1, 1, 1], [1, 1, 0, 0, 1], [1, 1, 0, 1, 1], [1, 0, 0, 1, 1], [1, 1, 1, 1, 1]]
run_success	bool	The success of the unit's run	Obtained	True
c_e	float	The constraint energy of the unit	Obtained	2951.728374235662
i_e	float	The internal energy of the unit	Obtained	1530.943825491704
u_id	str	The unique unit ID	Obtained	5_4121ab03c09f703 beb534ab74aa3c079
fp_u_...	str	The various file paths associated with the template	Obtained	C:..\Units\grid_1_5x5 \grid_5_4121ab03c09f703 beb534ab74aa3c079 \grid_5_4121ab03c09f703 beb534ab74aa3c079.mud

Table 4.5: Unit Class Object Parameters

Displacement and reaction force nodal values are used to calculate constraint energy. Constraint energy is defined as

$$E = \sum_{i=1}^{n} F_i d_i \tag{4.1}$$

where

- i is any node on the external boundary
- n is the total number of nodes
- F is the reaction force at the node
- d is the displacement at the node

Different results are desired for different cases. Specific cases are discussed in Section 4.3.

4.3 Cases

Generic properties are applied to each template. Generic properties include a ramp table representative of a simple $y = x$ equation used to apply displacements over time. Plane strain geometric properties are added. Solid state contact body properties are added. A static loadcase is added. A structural job is added requesting total strain energy density outputs.

4.3.1 Case 1

Case 1 is reflective of pure stress in a single direction. The y-direction is arbitrarily chosen. If extension in the x-direction is required, a unit may be rotated by 90°. Case 1 is obtained from [3]. Case 1 is selected because of the usefulness of simple linear extension and the lack of rigid body modes. Boundary conditions are defined in Table 4.6 and illustrated in Figure 4.6.

AB	**BC**	**CD**	**DA**
$u = 0$	$v = a$	$u = 0$	$v = 0$

Table 4.6: Case 1 Boundary Conditions [3]

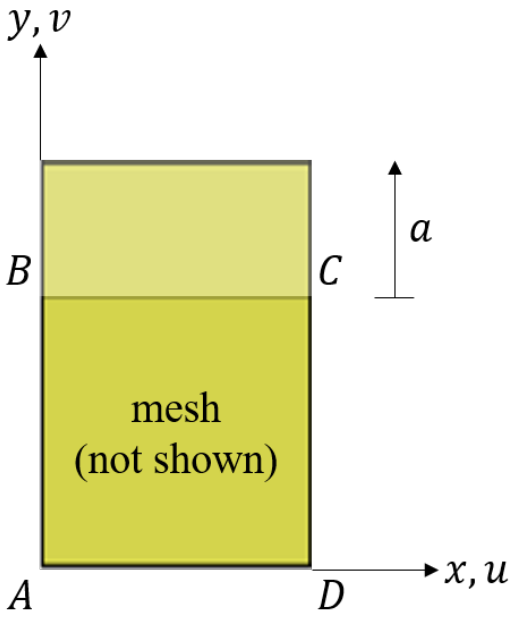


Figure 4.6: Case 1

Successful units have comparatively low energy in the y-direction and near zero energy in the x-direction. Comparatively low energy in the y-direction for a constant displacement boundary condition implies a lower reaction force. A comparatively lower reaction force implies a smaller force required to obtain the desired displacement. Near zero energy in the x-direction for a fixed zero displacement boundary condition implies a low reaction force. A low reaction force implies a low resistance to being constrained in the x-direction.

Figure 4.7 shows the distribution of 100 units' constraint energy in the x- and y-directions. Elements in the lower left area of the graph may be suitable candidates for case 1.

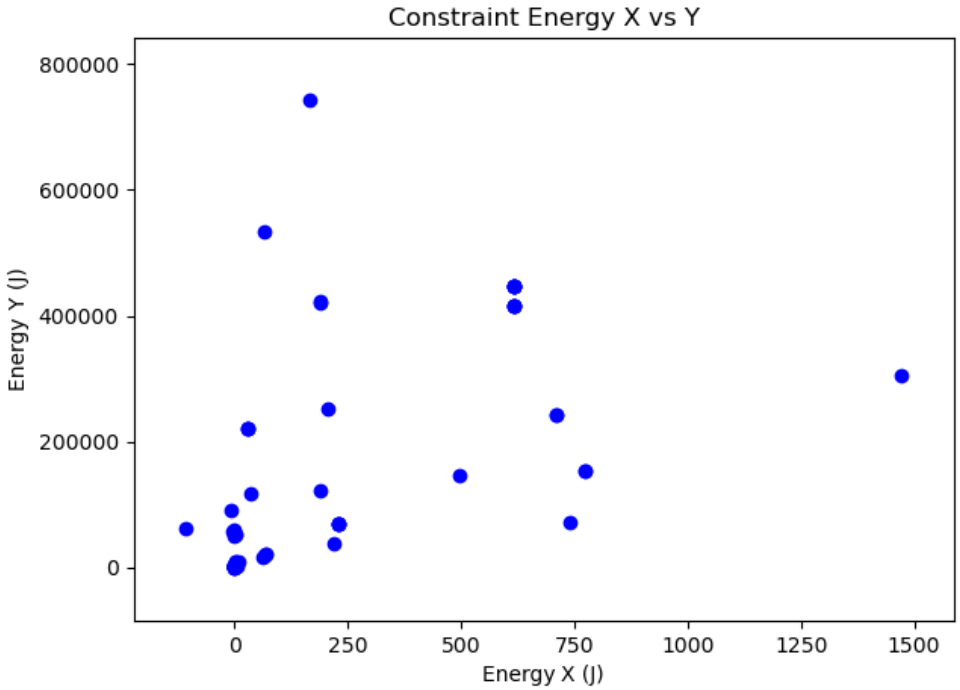


Figure 4.7: Constraint Energy Scatter Plot

4.3.2 Case 2

Case 2 is reflective of pure shear strain. Case 2 is obtained from [3]. Case 1 is selected because of the usefulness of shear distortion and the lack of rigid body modes. Boundary conditions are defined in Table 4.7 and illustrated in Figure 4.8.

AB	BC	CD	DA
$v = 0$	$u = a$	$v = 0$	$u = 0$

Table 4.7: Case 2 Boundary Conditions [3]

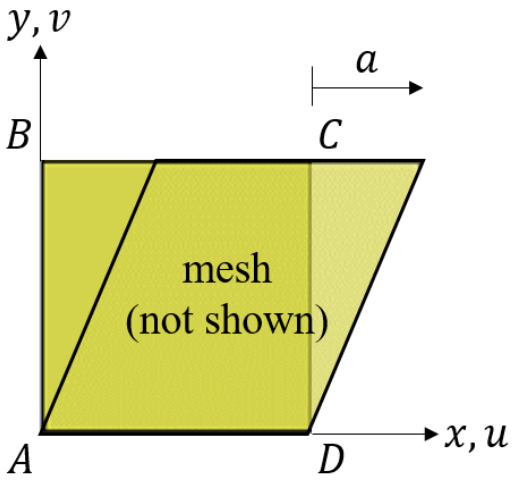


Figure 4.8: Case 2

Chapter 5

To-Do List

5.1 Introduction

- Scope And Assumptions
  - 2D
  - Square elements
  - Hyper-elastic material
- Project Layout

5.2 Literature Review

- L-systems
  - General definition
  - Illustrative diagrams
- CPPNs
- Diagrams

5.3 Material Testing

- Materials
  - Smooth Sil 950
  - Ecoflex 0030
- Testing
  - ISO standards

  - Testing procedure and equipment
    * Instron machine with 100kN load cell
    * Clamp grips or roller grips
    * Long travel extensometer or DIC

5.4 Software

- Calculations
  - More details
  - Software calculations
- Cases
  - Case 1
    * Example units
  - Case 2
    * Desired results
    * Example units
  - More cases

Appendices

Appendix A

Software Pipeline

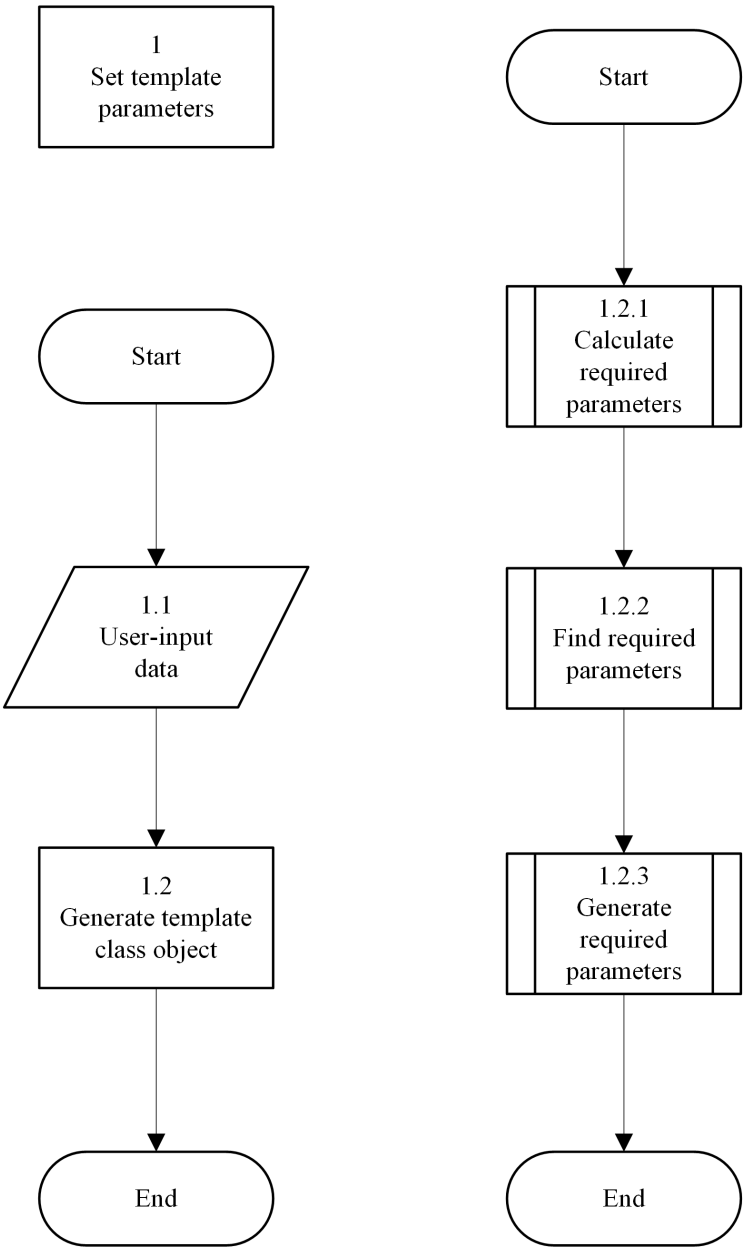


Figure A.1: Template Parameters

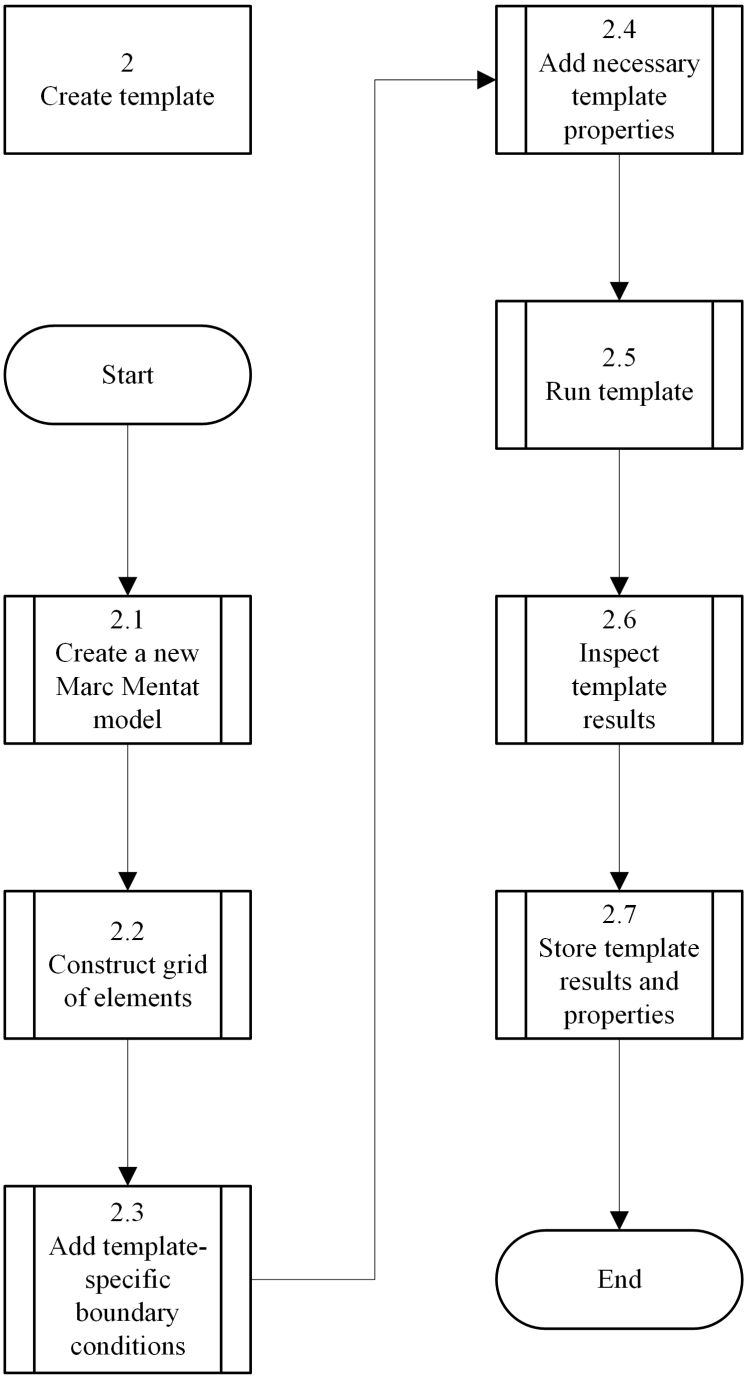


Figure A.2: Template Creation

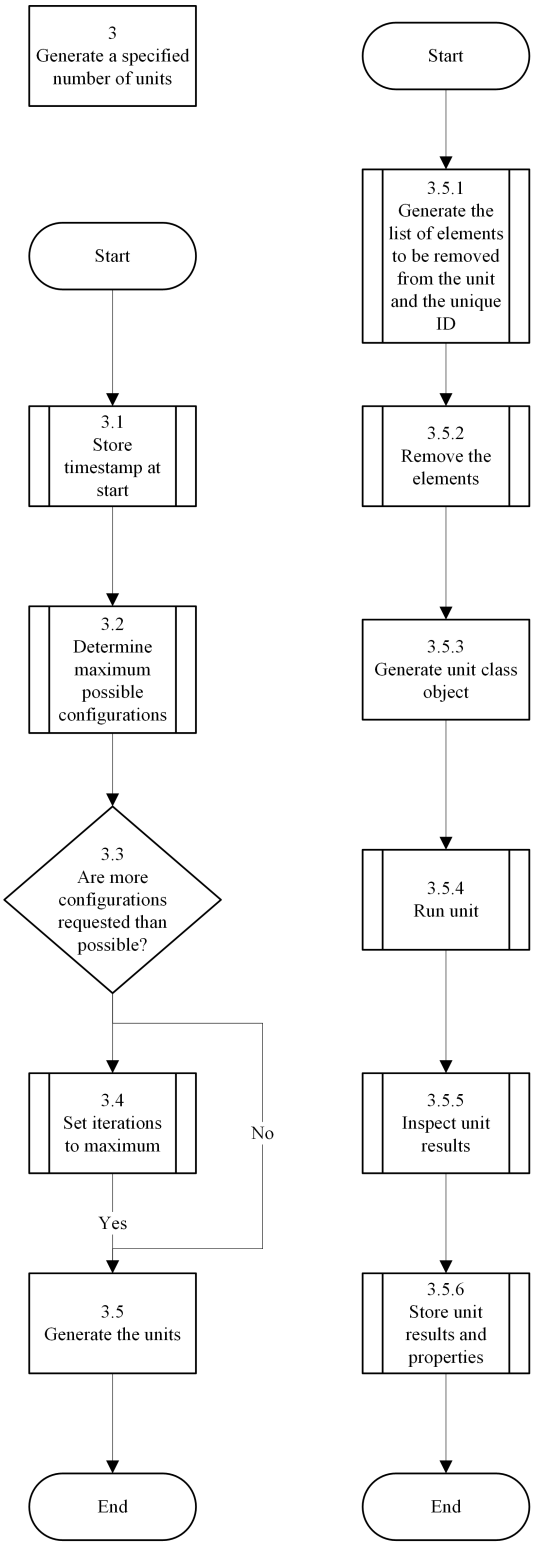


Figure A.3: Unit Generation

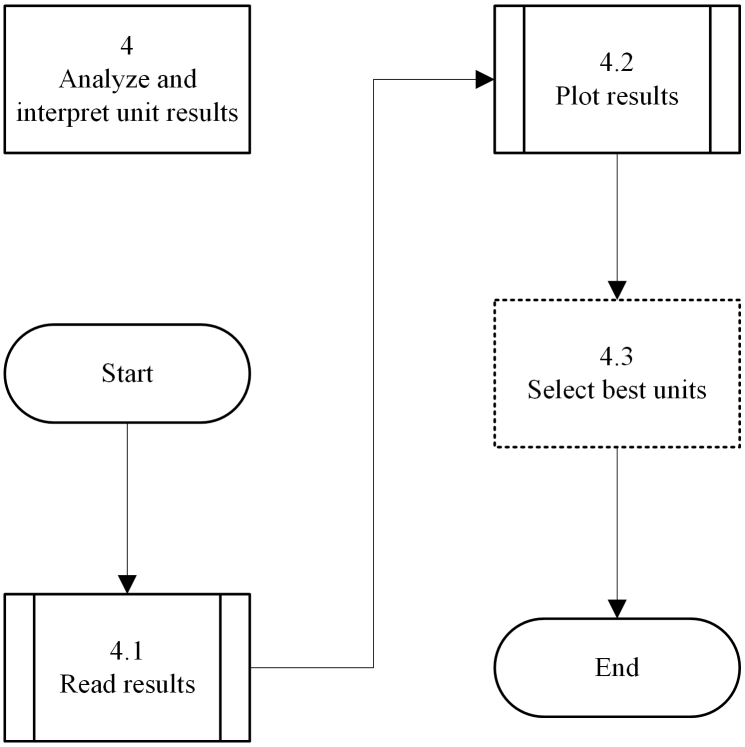


Figure A.4: Unit Selection

Bibliography

[1] Mold Star 15, 16 and 30.
Available at: http://www.smooth-on.com/tb/files/MOLD{_}STAR{_}15{_}16{_}30{_}TB.pdf

[2] Ellis, D.R.: *Generative Design Procedure for Embedding Complex Behaviour in Pneumatic Soft Robots.* Ph.D. thesis, Stellenbosch University, 2020.

[3] Cook, R.D., Malkus, D.S., Plesha, M.E. and Witt, R.J.: *Concepts and Applications of Finite Element Analysis.* 4th edn. Wiley, Madison, 2002. ISBN 978-0-471-35605-9.

[4] Shea, K., Aish, R. and Gourtovaia, M.: Towards integrated performance-driven generative design tools. *Automation in Construction*, vol. 14, no. 2 SPEC. ISS., pp. 253–264, 2005. ISSN 09265805.

[5] Whitesides, G.M.: Soft Robotics. *Angewandte Chemie - International Edition*, vol. 57, no. 16, pp. 4258–4273, 2018. ISSN 15213773.

[6] Boyraz, P., Runge, G. and Raatz, A.: An Overview of Novel Actuators for Soft Robotics. *Actuators*, vol. 7, no. 3, p. 48, 2018. ISSN 2076-0825.

[7] Brose, P.: *Compositional Model-Based Design: A Generative Approach To The Conceptual Design Of Physical Systems.* Ph.D. thesis, University Of Southern California, 1993.

[8] Sekhar, P. and Uwizeye, V.: Review of sensor and actuator mechanisms for bioMEMS. *MEMS for Biomedical Applications*, pp. 46–77, jan 2012.
Available at: https://www.sciencedirect.com/science/article/pii/B978085709129150002X

[9] Villoslada, A., Flores, A., Copaci, D., Blanco, D. and Moreno, L.: High-displacement flexible Shape Memory Alloy actuator for soft wearable robots. *Robotics and Autonomous Systems*, vol. 73, no. October 2017, pp. 91–101, 2015. ISSN 09218890.

[10] Behl, M. and Lendlein, A.: Shape-memory polymers. *Materials Today*, vol. 10, no. 4, pp. 20–28, 2007. ISSN 13697021.
Available at: http://dx.doi.org/10.1016/S1369-7021(07)70047-0

[11] Rodriguez, J.N., Zhu, C., Duoss, E.B., Wilson, T.S., Spadaccini, C.M. and Lewicki, J.P.: Shape-morphing composites with designed micro-architectures. *Scientific Reports*, vol. 6, no. June, pp. 1–10, 2016. ISSN 20452322.
Available at: `www.nature.com/scientificreports/http://dx.doi.org/10.1038/srep27933`

[12] Mutlu, R., Alici, G., Xiang, X. and Li, W.: Electro-mechanical modelling and identification of electroactive polymer actuators as smart robotic manipulators. *Mechatronics*, vol. 24, no. 3, pp. 241–251, 2014. ISSN 09574158.

[13] Do, T.N., Phan, H., Nguyen, T.-Q.Q. and Visell, Y.: Miniature Soft Electromagnetic Actuators for Robotic Applications. *Advanced Functional Materials*, vol. 28, no. 18, p. 1870116, 2018. ISSN 16163028.

[14] Shepherd, R.F., Ilievski, F., Choi, W., Morin, S.A., Stokes, A.A., Mazzeo, A.D., Chen, X., Wang, M. and Whitesides, G.M.: Multigait soft robot. *Proceedings of the National Academy of Sciences of the United States of America*, vol. 108, no. 51, pp. 20400–20403, 2011. ISSN 00278424.

[15] Onal, C.D., Chen, X., Whitesides, G.M. and Rus, D.: Soft mobile robots with on-board chemical pressure generation. *Springer Tracts in Advanced Robotics*, vol. 100, pp. 525–540, 2017. ISSN 1610742X.

[16] Cheney, N., MacCurdy, R., Clune, J. and Lipson, H.: Unshackling evolution: Evolving soft robots with multiple materials and a powerful generative encoding. *GECCO 2013 - Proceedings of the 2013 Genetic and Evolutionary Computation Conference*, pp. 167–174, 2013.

[17] Cheney, N., Bongard, J. and Lipson, H.: Evolving Soft Robots in Tight Spaces. pp. 935–942, 2015.

[18] LS-DYNA | Livermore Software Technology Corp.
Available at: `https://www.lstc.com/products/ls-dyna`

[19] Announcing the Next Generation Design Platform: NX 12 | NX Design.
Available at: `https://blogs.sw.siemens.com/nx-design/announcing-the-next-generation-design-platform-nx-12/`

[20] MSC.Marc Mentat Datasheet. 2003.

[21] Kim, N.H.: *Introduction to nonlinear finite element analysis*. 2015. ISBN 9781441917461.

[22] Sims, K.: Evolving virtual creatures. *ACM*, pp. 15–22, 1994.
Available at: `http://www.karlsims.com/papers/siggraph94.pdf`

[23] Hornby, G.S. and Pollack, J.B.: The advantages of generative grammatical encodings for physical design. *Proceedings of the IEEE Conference on Evolutionary Computation, ICEC*, vol. 1, no. December, pp. 600–607, 2001.

[24] Prusinkiewicz, P., Lindenmayer, A., Hanan, J.S., Fracchia, F.D., Fowler, D., de Boer, M.J.M. and Mercer, L.: *The algorithmic beauty of plants*. 1. 2004.

[25] Groenwold, A.A., Stander, N. and Snyman, J.A.: A regional genetic algorithm for the discrete optimal design of truss structures. *International Journal for Numerical Methods in Engineering*, vol. 44, no. 6, pp. 749–766, 1999. ISSN 00295981.

[26] Sims, K.: Evolving 3D Morphology and Behavior by Competition. *Artificial Life IV Proceedings*, pp. 28—-39, 1994.

[27] Kolodziej, J.: Modeling hierarchical genetic strategy as a Lindenmayer system. *Proceedings. International Conference on Parallel Computing in Electrical Engineering*, pp. 409–414, 2002.
Available at: `http://ieeexplore.ieee.org/document/1115312/`

[28] Metropolis, N. and Ulam, S.: The Monte Carlo Method. Tech. Rep. 247, 1949.

[29] Zakinthinos, A. and Lee, E.S.: Composing secure systems that have emergent properties. *Proceedings of the Computer Security Foundations Workshop*, pp. 117–122, 1998. ISSN 10636900.

[30] Aiguier, M., Gall, P.L. and Mabrouki, M.: Emergent properties in reactive systems. *Neonatal, Paediatric and Child Health Nursing*, , no. January 2008, pp. 273–280, 2008. ISSN 14416638.

[31] Damper, R.I.: Emergence and levels of abstraction. *International Journal of Systems Science*, vol. 31, no. 7, pp. 811–818, 2000. ISSN 14645319.

[32] Rieffel, J., Knox, D., Smith, S. and Trimmer, B.: Growing and Evolving Soft Robots. *Artificial Life*, vol. 20, no. 1, pp. 143–162, 2013. ISSN 1064-5462.

[33] Hiller, J. and Lipson, H.: Automatic design and manufacture of soft robots. *IEEE Transactions on Robotics*, vol. 28, no. 2, pp. 457–466, 2012. ISSN 15523098.

[34] Rubber, vulcanized or thermoplastic - Determination of tensile stress-strain properties. 2017.

[35] Rubber, vulcanized or thermoplastic - Determination of compression stress-strain properties. 2017.

[36] Joubert, I.J.: *Generative design using Lindenmayer-systems and numerical optimisation*. Ph.D. thesis, Stellenbosch University, 2020. `arXiv:1011.1669v3`.