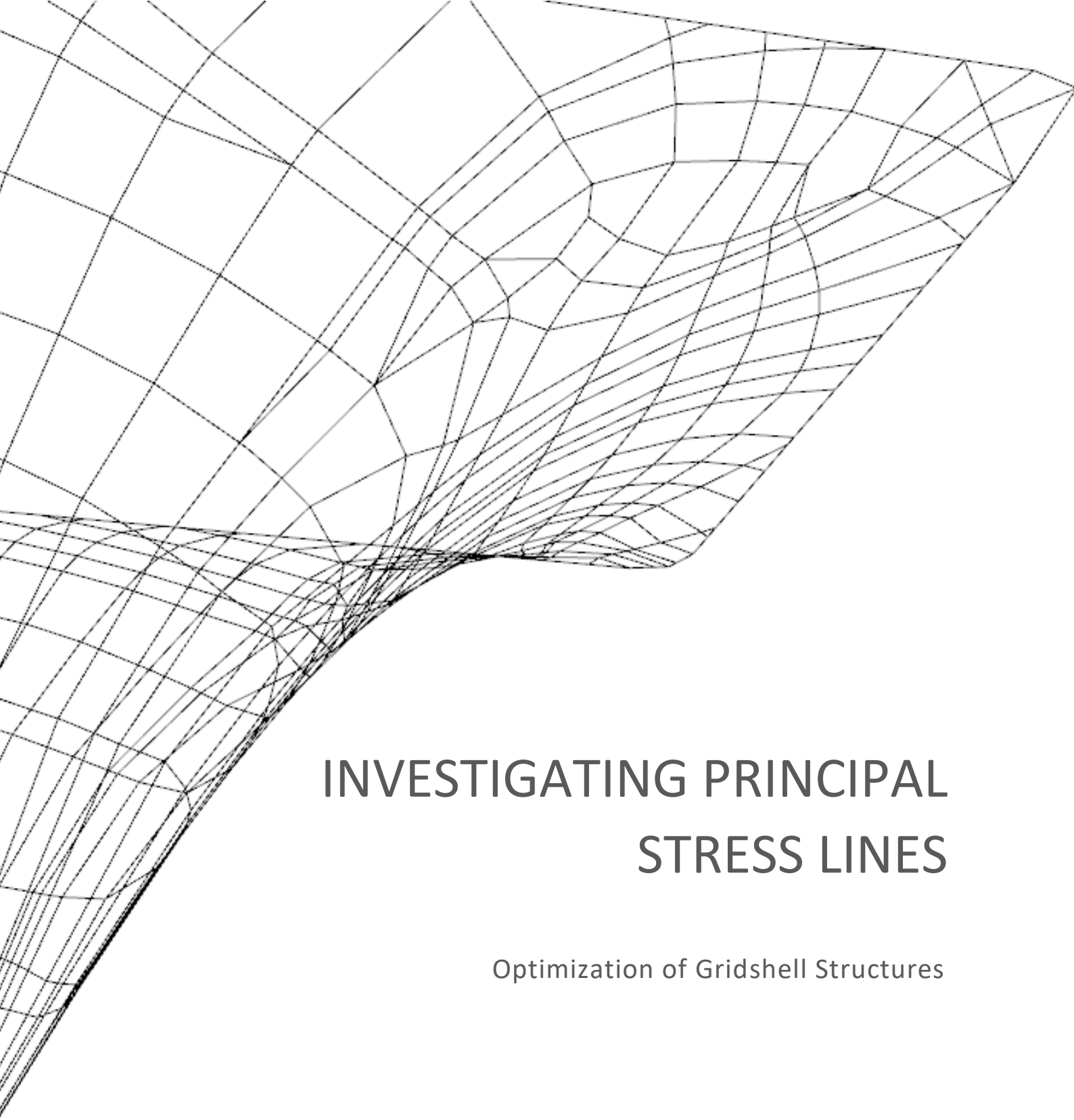


INVESTIGATING PRINCIPAL STRESS LINES

Optimization of Gridshell Structures

Michael Cobb (4510933)
m.a.t.cobb@student.tudelft.nl

INVESTIGATING PRINCIPAL STRESS LINES:

OPTIMIZATION OF GRIDSHELL STRUCTURES

MASTER THESIS FINAL REPORT

STUDENT:
Michael Cobb
Michael.a.t.cobb@gmail.com
4510933

First Mentor:
Ir. Andrew Borgart
A.Borgart@tudelft.nl

Second Mentor:
Dr. Ir. Pirouz Nourian
P.Nourian@tudelft.nl

External Examiner:
Dr. Ir. Sake Zijlstra
S.Zijlstra@tudelft.nl

31/10/2018

TU Delft

TABLE OF CONTENTS

TABLE OF SYMBOLS

SYMBOL	VARIABLE
VECTOR VARIABLES	
$\boldsymbol{v}$	General Vector
$\boldsymbol{f}$	Force
$\boldsymbol{n}$	Node Position
u	Velocity
$\boldsymbol{\sigma, \sigma_1, \sigma_2}$	Stress
$\boldsymbol{n_{xx,} n_{xy,} n_{yy}}$	In Plane Normal Force
d	Displacement
P	Load
x,y,z	Cartesian Coordinates
$\boldsymbol{c}$	Curvature
SCALAR VARIABLES	
$\boldsymbol{U}$	Energy
$\boldsymbol{E}$	Young's Modulus
$\boldsymbol{V}$	Volume
$\boldsymbol{N}$	Node Number
n	Count
R	Radius
$\boldsymbol{\epsilon}$	Strain
$\boldsymbol{m_{xx}, m_{xy}, m_{yy}}$	Moment
$\boldsymbol{f_s}$	Shear
$\boldsymbol{\phi}$	Plate Rotation
$\boldsymbol{\Delta}$	Change
B	Boolean
$\boldsymbol{\theta, \alpha, \beta}$	Angle
M	Mass
$\boldsymbol{\rho}$	Density
$\boldsymbol{v}$	Poisson's Ratio
i,j	Vertex Index
D	Plate Stiffness
MATRIX VARIABLES	
$\boldsymbol{K}$	Stiffness Matrix
A	Transformation Matrix
I	First Fundamental Form
V	Vector Matrix

All other symbols are described in the Thesis

ABSTRACT

Architecture within the last twenty years has developed in many ways. One such direction was reintroducing the physical world and physical phenomenon back to architectural language and design space. This is especially true as buildings have had to become more and more environmentally sustainable and developable with programs such as LEED, BREEAM, and Zero House popping up. While less pronounced, the same is true for other aspects of architecture too; such as pedestrian flow, constructability, and structural design. Each design in hi-tech architecture has become a compromise of these systems.

While significant funding has gone into understand how to design for energy efficient buildings, architects typically shy away from structural systems leaving these problems for structural engineers to solve. However, great design comes to fruition when the two systems work hand in hand. This method of design, integrated design, is starting to appear more and more in the design space with gridshell structures becoming classic icons of the beauty of this marriage.

Gridshell structures are becoming more widely known as architects such as Foster + Partners, Asymptote, and John McAslan & Partners have developed work that has become widely lauded and incredibly awe inspiring. These slender shapes still allow for large amounts of natural light as well as enable architects and designers to cover vast areas without the need of massive beams and a large amount of columns, instead finding rigidity in its own form.

This research project, attempts to create a system for architects and designers to play with in order to develop an efficient and useful gridshell concept structure and does this by approaching the perspective from two points. On one side a computational system is built based on Finite Element Analysis (FEA) data and signal processing while on the other, structural concepts are explored in order to determine a geometric relationship between load, form, and principal stresses.

Keywords: Gridshell, form finding, principal stress, topology, parametric design, performance based design, structural analysis

PREFACE

Investigating Principal Stress Lines is Michael Cobb's graduation thesis for the MSc. In Building Technology at the Faculty of Architecture at TU Delft. This research is a part of the Sustainable Graduation Studio graduation lab and specializes in Structural Design and Design Informatics.

I would like to acknowledge the help given to me by my two mentors Andrew Borgart and Pirouz Nourian. Their guidance in developing this thesis into its final paper has been vital and nothing short of miraculous. While this thesis took longer than either expected, their help and continued dedication was fundamental to the completion of this paper. I would also like to thank Peter Eigenraam for the Force Area Ratio Grasshopper3d code which was useful in examining proper loading methods for shell analysis.

I would like to thank Yu Feng Wong, Alex Kouwenhoven, Lia Tramontini, and Alvaro Garcia for their help with the visualization and model of TU Delft Bouwkunde along with the rest of the 2016-2018 MSc Building Technology Graduates for the companionship over the last two years.

I would like to thank Tamis van der Laan for his help with rationalizing and checking my Python code.

Lastly, I would like to dedicate this thesis to my mother (Sieuwke), father (Dave), brother (Niels), and sister (Janneke), as well as my Aunt (Els) and Uncle (Thijs). Without their stability and support I never would have been able to continue with my education. Their patience and love has meant the world to me.

1 INTRODUCTION

The goal of the project was to learn and develop a computational and parametric based methodology for designing non-standard gridshell structures. This was done through combining form finding and gridshell topology based on a base ground surface in order to develop a system that is structurally informed and intelligently optimized to reduce material and construction costs.

1.1 PROBLEM STATEMENT

'Form follows function', a quote first described by Louis Sullivan with the idea that the shape of the building should follow its intended uses, is a guiding theme seen throughout most of architectural history.

This ideal pops up much earlier in the Vitruvian values as utilitas, and is adored even later on in time by Le Corbusier as he marvels over the aesthetic of pure functional form in 'Towards a New Architecture'.

However, with shell structures and gridshells specifically, this method of design seems to have been forgotten when examining grid generation. Instead of developing grids that follow the flow of forces within the shell structures, the grids are generated using patterning and tessellation. This paper tries to repair this issue by defining other methods of developing a topology for gridshells and identify the issues that are related to these current systems.

1.2 GOAL

The goal of this thesis is to develop a quadrilaterally dominated gridshell layout that is created using straight bar elements. The goal is then to facilitate a potential design for the front entrance of the TU Delft Architecture Faculty. This structure of the gridshell should be laid out in a manner that optimizes mitigating the structural strain in gridshell structures. The thesis also compares the system to current more standard methods of developing gridshells. This thesis accomplishes a summary and review of methods that have been experimented with over the last several months.

1.3 RELEVANCE

The research undergone in this process should allow architects and engineers the ability to design lighter gridshell structures. This reduction in weight of gridshell structures is dependent upon the optimization occurring along the main load paths, to deliver the reduction in the amount of steel or concrete needed to create these architectural forms.

Steel and Portland cement production results in roughly 2 kg CO_2 eq / kg and 0.85 kg CO_2 eq / kg respectively (MPA Cement, 2015). By reducing the amount needed in each structure, large steps can be made to reduce the CO2 output of the construction industry.

This reduction in weight also leads to several other advantages. The demand for materials required to make concrete for example, especially quartz sand, is currently decimating local ecosystems as riverbeds and beaches are being stripped bare (NPR, 2017). Steel faces similar issues, with iron ore shortages already affecting many smaller steel plants in 2017(Bloomberg, 2017).

While gridshells are already efficient at spanning large distances and creating large open spaces, the popularity of this method is growing significantly with recent large scale projects including Google Mountainview Campus, Kings Cross Station, Macallan Visitors Center, and Yas Hotel all finishing up within the last 10 years. While many of these gridshells are highly efficient and beautiful, most of these systems rely on more traditional UV parameterization that is not informed by the flow of forces in the shell. This thesis attempts to tie the flow of forces to the geometry of the gridshells.

[1] MPA Cement. (2015, July 15). Fact Sheet 18 Embodied CO2e of UK cement, additions and cementitious material. Retrieved January 4, 2018, from http://cement.mineralproducts.org/documents/Factsheet_18.pdf
[2] NPR. (2017, July 21)World Faces Global Sand Shortage. Retrieved January 04, 2018, from https://www.npr.org/2017/07/21/538472671/world-faces-global-sand-shortage
[3] Ng, Williams & Engel. (2017, June 05). The Hard To Believe Steel Shortage That's Unfolding in China. Retrieved January 4, 2018 from https://www.bloomberg.com/news/articles/2017-06-05/the-hard-to-believe-steel-shortage-that-s-unfolding-across-china

1.4 RESEARCH QUESTION

How can the creation of a gridshell by using principal stress directions be used to provide a more efficient structure that follows principal stress lines?

1.4.1 Sub-questions

How can rod paths be plotted along principal stress streamlines on freeform surfaces?

What form finding methods are suitable for generating an efficient structural form with high percentage shell behavior (no out of plane forces) and low strain energy density (high stiffness)?

Is there a considerable advantage in optimizing a gridshell structure based on principal stress stream lines and an arbitrary generated tessellation?

1.5 METHODOLOGY

The thesis and research will be approached in the following way:

The gridshell (as a nurbs multi-surface) is developed in a set of stages of parametric interpretation of a set of parameters.

This project will be approached from two different perspectives. First, a streamline tracer method will be developed and compared to a standard 2 Dimensional UV mapping and a more advanced method known as Periodic Global Parameterization (PGP)(Ray, Li, Lévy, Sheffer, & Alliez, 2006). Secondly, structural mechanics research will be done to determine if a principal stress lines can be derived from the structure of the plate or shell rather than indirectly through the use of Finite Element Analysis vector sets, through the examination of Calladine's two shell theory and Beranek's plate theory.

The initial parametrization of the shape is completed using standard UV parameterization tools in Grasshopper3D and rationalized into a quadrilateral mesh surface with UV parameterization depending on its geometry. This is done via two scripts depending on the complexity of the geometry inserted.

The initial form finding is conducted using a linear solver/physics simulation tool known as Kangaroo (Piker, 2017) inside the Grasshopper3D (Rutten, 2014) environment. Here the form finding parameters are tested within 3 different structures of varying complexity and ranked based on the strain energy density and membrane vs moment behavior of the 3 different structures. From there, decisions are made based on the goal surface as to whether or not which parts of the form finding system will be useful in developing the final shape. Each shape will be directly and automatically exported to GSA (Oasys GSA, 2017) in order for FEA structural analysis to develop strain energy data and principal stress vectors.

The two methods of topology generation are compared. The first method is streamline fitting along a conformal mapping through a script written by myself in Python, while the second is using conformal mappings to generate a new map directed along principal stress direction. The attempts at the latter step failed and therefore a black box program within the Grasshopper3D environment was used so that one option could be generated using a program known as Millipede (Panagiotis & Sawako, 2014) for comparison.

Lastly, the structure is analyzed and sized using Karamba3D (Preisinger, 2016) and the strain energy of each base structure is compared to the man-made idealized solution and non-ideal solutions in order to ensure effectiveness.

2 LITERATURE STUDY

2.1 GRIDSHELL STRUCTURES

2.1.1 Introduction to Gridshells

Gridshells or lattice shells are rigid structural bodies spanning long distances using the form of the structure itself to carry the loads. There has been contention over whether a gridshell is defined by the structural concepts of the structure or whether it is determined by the method of construction.

The gridshell is according to Edmund Happold “a doubly curved surface formed from a lattice of timber bolted together [...]. Where the lattice is a mechanism with one degree of freedom." (Otto, 1974).

Fixed Jointed gridshells are commonly seen with steel gridshells being built today. These style of gridshells are typically created with pre-curved elements. This means that the structure uses more standard methods to generate its form and relies on a combination of its own stiffness and approximated deflection to predict the final form (Otto, 1974).

Pin jointed gridshells are typically generated from a series of flat members (most often wood) which are then lifted and fastened into position, thereby relying on the pre-stressing of the grid to also help maintain its structure. This method typically is developed through defining a base structure covering the goal surface and slowly flattened with the axial lathes warping out of planes rather than rotating (Otto, 1974).

However, as Pellegrino and Miura state in their book “Structural Concepts” (Forthcoming), besides the stresses developed in the system by the lifting of the pin-jointed version, both types of structures behave similarly as if the nodes themselves were pins much like most frame structures (Miura & Pellegrino, Forthcoming).

2.1.2 **Historical Background**

Despite the use of weaved and gridded structures in small scales, it was not until the end of the 1800s that Vladimir Shukhov created the first modern gridshell structures. Figure 2-1 shows the doubly curved grid structure under construction in Vyksa, Russia in 1897. However, the first known thoroughly conducted studies were developed by Frei Otto at the Institute of Lightweight Structures in the 1970's (Otto, 1974).

The work conducted at the institute focused on form finding through the use of hanging chain models through the construction of experimental small scale structures. Otto at this time named them as "gridshells" and defined them as the following:

"… a spatially curved framework of rods and rigid joints. The rod elements form a planar grid with rectangular meshes and constant spacing between the knots. The form of a gridshell is determined by inverting the form of a flexible hanging net." (Otto, 1974)

Therefore, in this paper a gridshell will be defined as:

A long span lattice structure emulating the behavior of a shell structure with the majority of load being transmitted axially through the lattice members with nodes that behave as if they were pinned.

This definition allows for the consideration of new steel based lattice structures while also not discounting the original lifted timber structures. This definition also defines the nodes as not being able to carry moment and therefore requires the structure to find a theoretically optimal solution.

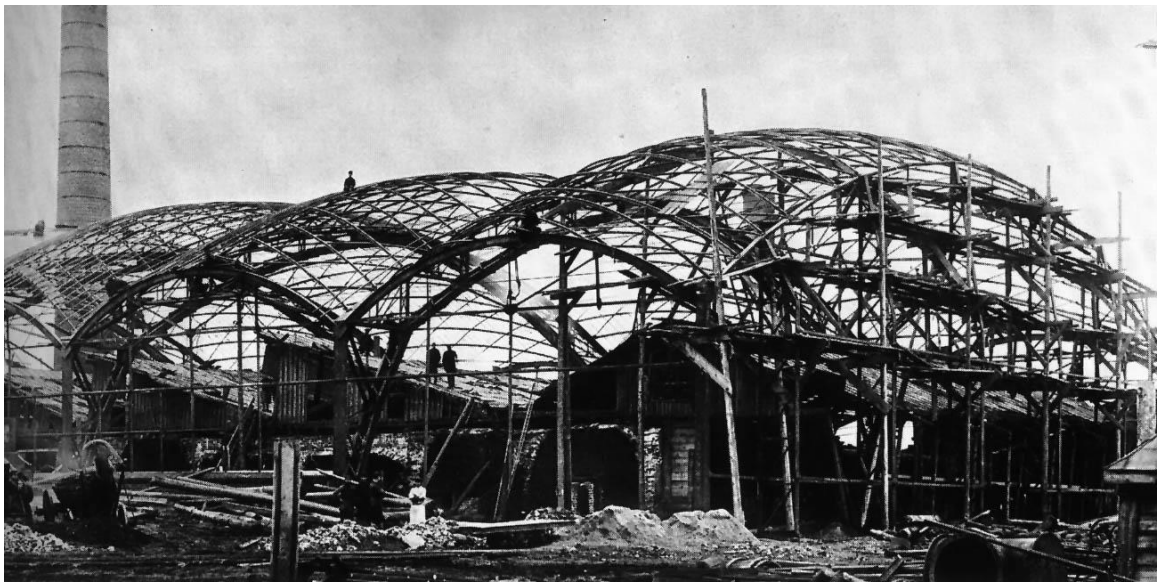


Figure 2-1: Shukov Shell, Russia (Source: Wikipedia Creative Commons)

The “hanging net” was an integral part of Otto’s 1970’s hanging chain models applied to construct the Multihalle. This type of technique had been used earlier by the likes of Gaudi for his form finding methods on non-gridshells in the 1880’s (Winslow, 2010).

Until recently, there have been very few of these structures; however, there are some non-free form structures that are also of interest, specifically the Gatti Wool Factory. This project completed by Nervi is a set of projects that examines the structural efficiency of intelligently placed concrete. This project specifically looked at the principal stresses of the floor plate and placed thicker ribs along the trajectories (Figure 2-2).

Figure 2-2: Gatti Wool Factory Roof (Winslow, 2010)

Typically a gridshell is developed by placing a preset pattern on a surface and relaxing the mesh until it fits properly. While this is a viable method, it lacks a feed forward approach to design with the focus on grid sizing rather than grid positions.

However, this has recently changed. In the last few years several papers such as Winslow’s Thesis (Winslow, 2010) as well as that of Mark Tim Kam (Kam, 2014) have approached the subject in two different ways, focusing on principal stress directions. While Winslow examines the mapping of stress vectors using a global parameterization, Kam prefers to directly trace the vector sets on the 3D shell.

2.1.3 **Generation of Gridshell**

Gridshell generation typically occurs along a set pattern, curvature, or network. These patterns in turn provide the stability and form to a gridshell. Typically tetrahedral and triangular elements are either generated from these patterns or are used to generate the gridshell patterns. Once the pattern is set, the tessellation is then relaxed until it fits continuously over the surface. However, recent topology analysis and generation of non-uniform gridshell tessellation has also been examined by Smidt (Smidt, 2014).

Regardless of the chosen pattern, all grid sizes are chosen initially by hand and developed by defining the distance of the nodes and creating a specific fineness of the mesh. From here a variety of options can be applied to either decrease or increase the node spacing as is sometimes done depending on the curvature of the gridshell.

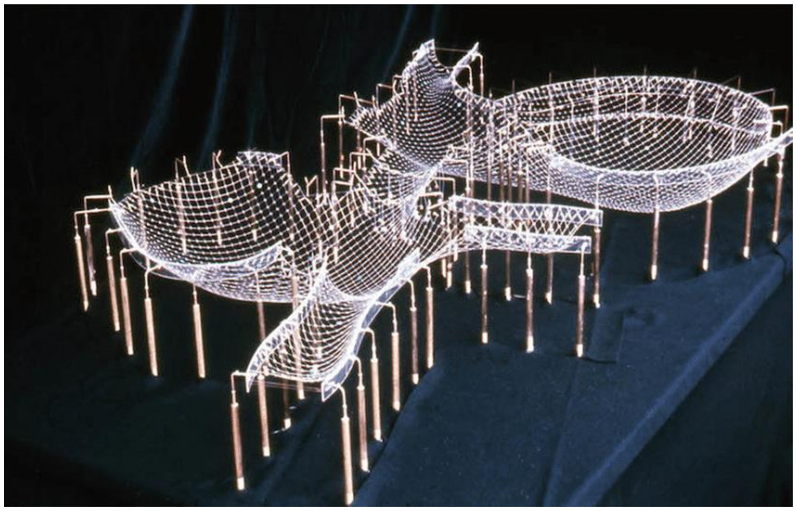


Figure 2-3: Hanging Net Structure of the Mannheim Multihalle (Otto, 1974)

2.2 FORM FINDING

While the thesis not only focuses on form finding of the shell, form finding analysis is still required to generate a proper shape for the gridshell bar structure to be generated on. Currently there are several methods used for the overall form finding for structural shells. These methods are outlined here for the development of shells and gridshells that rely on axial forces rather than bending moment in the shell to remain stiff. However, these form finding methods are not perfect.

Typically there are two types of form finding, empirical and computational. This thesis focuses on computational form finding methods by using a subset of defined elements and the forces loaded onto the shape. The form finding section of the thesis uses an iterative method until the geometry and forces in the lattice-spring structure efficiently counteract the loads applied to the structure.

2.2.1 Direct Stiffness Method

The essence of the direct stiffness method lies in establishing the relationship between applied forces at certain nodes of structure and the displacements occurring there. This relationship can be further denoted in the following equation

$$\boldsymbol{K}_{ij}\boldsymbol{d}_j = \boldsymbol{f}_i$$

Equation 2-1: Relationship between Stiffness, displacement, and Forces for a Node. (Samuelsson & Zienkiewicz, 2006)

Essentially, it is a matrix method that uses the relationship members' stiffness ($\boldsymbol{K}$) in order to determine either forces (**f**) or displacements (**d**) in structures with one or the other unknown. The direct stiffness method is the most common implementation of the finite element method, and therefore, is also inextricably linked with all structural optimization methods.

The entire structure is then modeled as a subset of simpler elements connected at nodes. The stiffness properties of these elements are compiled into a matrix which determines the behavior of the entire structure from which the displacements and forces can subsequently be determined. This stiffness method is similar to the method Pelligrino and Calladine (1986) used to develop the matrices for Singular Value Decomposition (SVD) of the equilibrium matrix, but when using the direct stiffness method for form finding, the method includes member stiffness as part of the matrix rather than just the connected network as needed for SVD.

2.2.1.1 Matrix Formulation

In the 1930's, Gabriel Kron developed a matrix formulation of these systems with each member creating a 12 x 12 matrix of the force-displacement relation and each of these matrices arranged diagonally in a large square matrix. This matrix was considered the tensor and the connections of frame were obtained by applying a transformation (Samuelsson & Zienkiewicz, 2006).

This formulation was later redeveloped by Argyris such that all component stiffnesses were arranged in diagonal matrix $\boldsymbol{K}$ and introduced to a transformation **A** to construct matrix $\boldsymbol{K}$**'** as shown here in Equation 2-2 (Samuelsson & Zienkiewicz, 2006).

$$\boldsymbol{K'} = \boldsymbol{A}^T \boldsymbol{K} \boldsymbol{A}$$

Equation 2-2: Matrix Formulation of the Stiffness Matrix (Samuelsson & Zienkiewicz, 2006)

Therefore by applying a set of forces as $\boldsymbol{A}$ to a stiffness matrix **K** the new matrix **K'** will be the matrix representation of the stiffness matrix for the new transformed geometry.

2.2.2 Dynamic Equilibrium Methods

2.2.2.1 Dynamic Relaxation Using Particle Springs

This method involves generating the structural elements as masses on springs and using the equilibrium position of these masses and the forces applied to them. In order for equilibrium to be reached damping of the oscillation is used.

Particle spring method is an updated subset of dynamic relaxation. In this method, springs and point masses are developed using a base mesh in order to simulate a gridshell structure. This method visualizes the mechanical energy through the use of a set of springs. Because these springs are modeled as ideal springs, they are unable to carry moment which along with the point masses creates a form found shape using the same principals as dynamic relaxation.

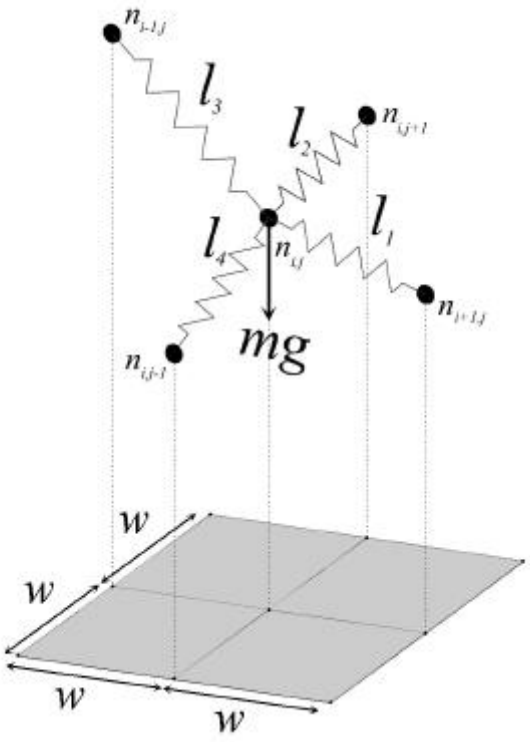


Figure 2-4: Forces on A 3-D Spring (Harding, 2011)

However, since none of the springs can carry moment, at equilibrium $\sum m_p = 0$ at all points in the form, therefore creating a catenary structure. The mass (m) nodes of the spring move on in time steps due to out of balance forces and rely on the spring constant (k_s) and the elongating tied springs to create static equilibrium such that the following holds true for a mass spring system with zero length springs.

$$\boldsymbol{f}_{ij,z} = k_s\left(\boldsymbol{z}_{i+1,j} + \boldsymbol{z}_{i,j+1} + \boldsymbol{z}_{i-1,j} + \boldsymbol{z}_{i,j-1} - 4\boldsymbol{z}_{i,j}\right) - m\boldsymbol{f}_g$$

Equation 2-3: Fore at Node ji in the Z direction (Harding, 2011)

For a spring system using non-zero length springs, the initial points are simply a non-zero value, therefore reducing the change in strain. In order to compensate for this, oftentimes a higher spring constant is chosen to ensure the development of a proper form.

$$\begin{aligned}\boldsymbol{f}_{\boldsymbol{ij,z}} = k_s \cdot \Bigg(&\frac{(\boldsymbol{z}^t_{i+1,j} - \boldsymbol{z}^t_{i,j}) - (\boldsymbol{z}^0_{i+1,j} - \boldsymbol{z}^0_{i,j})}{|\boldsymbol{z}^0_{i+1,j} - \boldsymbol{z}^0_{i,j}|} + \frac{\left(\boldsymbol{z}^t_{i,j+1} - \boldsymbol{z}^t_{i,j}\right) - (\boldsymbol{z}^0_{i,j+1} - \boldsymbol{z}^0_{i,j})}{|\boldsymbol{z}^0_{i,j+1} - \boldsymbol{z}^0_{i,j}|} \\ &+ \frac{(\boldsymbol{z}^t_{i,j-1} - \boldsymbol{z}^t_{i,j}) - (\boldsymbol{z}^0_{i,j-1} - \boldsymbol{z}^0_{i,j})}{|\boldsymbol{z}^0_{i,j-1} - \boldsymbol{z}^0_{i,j}|} + \frac{\left(\boldsymbol{z}^t_{i-1,j} - \boldsymbol{z}^t_{i,j}\right) - (\boldsymbol{z}^0_{i,j-1} - \boldsymbol{z}^0_{i,j})}{|\boldsymbol{z}^0_{i-1,j} - \boldsymbol{z}^0_{i,j}|}\Bigg) \\ &- m\boldsymbol{f}_{\boldsymbol{g}}\end{aligned}$$

$$\boldsymbol{f}_{\boldsymbol{ij,(x,y)}} = k_s \cdot \left(\frac{\left(\boldsymbol{n}^t_{i+1,j} - \boldsymbol{n}^t_{i,j}\right) - (\boldsymbol{n}^0_{i+1,j} - \boldsymbol{n}^0_{i,j})}{|\boldsymbol{n}^0_{i+1,j} - \boldsymbol{n}^0_{i,j}|} + \frac{(\boldsymbol{n}^t_{i,j+1} - \boldsymbol{n}^t_{i,j}) - (\boldsymbol{n}^0_{i,j+1} - \boldsymbol{n}^0_{i,j})}{|\boldsymbol{n}^0_{i,j+1} - \boldsymbol{n}^0_{i,j}|} + \frac{(\boldsymbol{n}^t_{i,j-1} - \boldsymbol{n}^t_{i,j}) - (\boldsymbol{n}^0_{i,j-1} - \boldsymbol{n}^0_{i,j})}{|\boldsymbol{n}^0_{i,j-1} - \boldsymbol{n}^0_{i,j}|} + \frac{\left(\boldsymbol{n}^t_{i-1,j} - \boldsymbol{n}^t_{i,j}\right) - (\boldsymbol{n}^0_{i-1,j} - \boldsymbol{n}^0_{i,j})}{|\boldsymbol{n}^0_{i-1,j} - \boldsymbol{n}^0_{i,j}|}\right)$$

Equation 2-4: Force at node ij at time t based on a 4 connected Node point with springs at rest at length during t=0. (Edited from: Harding, 2011)

In Equation 2-4 the force applied to node i in the z direction is different than that to the x and y directions as the additional force of gravity is applied to the point mass at each node. The length of the springs are written as the differences in their x,y,z coordinates where the position is determined at time (t) while the original spring length occurs at time (0). This, therefore, generates two sets of equations: one for in the z direction, where the forces include a gravity force, and a second set for the x and y direction forces where gravity plays no role. Since the strength of the force is equal to the change in length of the spring over its original length, this can be rewritten in each dimension as a change in the length of the spring in each Cartesian coordinate.

2.2.2.1.1 Kinetic Damping

The particles, however, will oscillate forever if the springs are not dampened. There are currently two methods of damping available, Kinetic Damping and Viscous Damping. The following two sections will describe the differences in the two. In a mechanical system, the total energy (U_{total}) is the sum of the kinetic energy ($U_{kinetic}$) and of the potential energy ($U_{potential}$).

$$U_{tot} = U_{kinetic} + U_{potential}$$

Equation 2-5: Summation of Energy in a System (Fritzsche, 2013)

Kinetic Damping works by resetting the velocity of the system to 0 at the point when it reaches maximum Kinetic Energy (or maximum average velocity) (Fritzsche, 2013). In order to use kinetic damping it must be assumed that that the

system is subject to conservative forces, and therefore the mechanical energy is conserved.

$$\frac{dU_{tot}}{dt} = \frac{dU_{kinetic}}{dt} + \frac{dU_{potential}}{dt} = 0$$

Equation 2-6: Energy Relationship between Total, Kinetic and Potential Energy (Fritzsche, 2013)

In a static approach, the equilibrium corresponds to a minimum of the potential energy, or maximum kinetic energy, as at that point, no forces are acting on the system that increase kinetic energy and velocity.

Since the mechanical energy is constant, it means that the kinetic energy reaches a maximum when the system is at the equilibrium position. The kinetic damping consists of looking for maxima of the kinetic energy to a structure subject to an artificial dynamic motion. When an approximate value for maximum is found, the structure is stopped and set to the approximate equilibrium position. From there, a new approximation of the maximum of the kinetic energy can be found. This iterative procedure converges to the equilibrium position. (Fritzsche, 2013)

2.2.2.1.2 Viscous Damping

In order for the particles to converge to their final positions, a damping factor can be used instead of the kinetic damping described in section 2.2.1.1. Kangaroo, according to the author of the program, Daniel Piker, utilizes a viscous damping factor depending on the particle's velocity. A damping factor generates a 'resistance' force that is antithetical to the current particle velocity ($\boldsymbol{u}_i$). Equation 2-7 shows the relationship as the damping force at time t is proportional to and acts opposite to the velocity of the current particle.

$$\boldsymbol{f}_{v,i}^{t} \propto -\boldsymbol{u}_{\boldsymbol{i}}^{\boldsymbol{t}}$$

Equation 2-7: Relationship between Viscous Damping Force and Velocity (own)

By utilizing this viscous damping system, particles will converge within the system as the damping coefficient directs the velocity of the particles to 0. The system will however never reach exactly 0 velocity, thus instead the solver stops once a specific threshold is reached (Adriaenssens, Block, Veenendaal, Williams, & Williams, 2014). This damping factor can be rewritten in the following method (Equation 2-8) to be utilized within form finding programs.

$$\boldsymbol{u}_{ix}^{t+\frac{\Delta t}{2}} = A \times \boldsymbol{u}_{ix}^{t-\frac{\Delta t}{2}} + B \times \left(\frac{\Delta t}{m_i}\right) \boldsymbol{f}_{ix}^{t}$$

$$where\ A = \frac{1-\frac{C}{2}}{1+\frac{C}{2}}, B = \frac{1+A}{2}, C = damping\ factor$$

Equation 2-8: Viscous Damping Equation (Adriaenssens, Block, Veenendaal, Williams, & Williams, 2014)

Equation 2-8 shows how a damping factor (C) would result in a scalar decrease of the change in velocity on each time step ($\boldsymbol{u}_{ix}$) but also the change in residual forces (**f**) (spring and gravity) acting to create acceleration in the i direction at time (t) on mass (m).

By using this method, a set constant can be determined and altered allowing for differing levels of damping while ensuring that consistent convergence, without analyzing the energy of the system (Adriaenssens et al., 2014).

2.2.3 **Choice of methodology**

In the end a choice was made to examine dynamic equilibrium methods in the following chapters more closely and to determine the viable levels of integration into the Grasshopper3D environment. This is further explained and explored in the form finding section of the thesis in Chapter 5. The choice for dynamic equilibrium was chosen over direct stiffness as there are several systems that work well within the Grasshopper3D environment for quickly and efficiently developing usable structures in the Grasshopper3D environment.

2.3 STRUCTURAL OPTIMIZATION

The main goal of the thesis is to develop an optimized gridshell structure, understanding the various forms of optimization and how these optimizations occur. This is vital to developing the ability to design and create the more streamlined method of developing the goal gridshell. All these processes use the total strain energy of the structure as described in Equation 2-9 as their ranking or optimization criterion. They all then apply as their optimization based on a minimization of strain energy where:

$$U_{strain} = \frac{1}{2} V\sigma\epsilon = \frac{1}{2} VE\epsilon^2 = \frac{1}{2}\frac{V}{E}\sigma^2$$

Equation 2-9: Various Methods to Calculate Strain Energy (Bendsoe & Sigmund, 2003)

Where:

U= strain energy
V= volume of structure
E= young's modulus
ϵ= strain
σ= stress

Here a tension is developed. Decreasing the volume of the structure would ideally also decrease the strain energy, however, as the volume decreases, the strain in each part of the structure increases. Due to the play between these two variables and the desire to minimize both, quite often structural optimization questions become multi-variable optimizations. For this reason minimizing strain energy is often used in both form finding and structural optimization as the objective function.

2.3.1 **Michell Trusses**

In 1904 Michell developed a series of analytically defined frame structures shown to be the most materially optimal for their support and load cases. These trusses are known as Michell trusses and are incredibly famous throughout structural computation as the main method of testing sizing, shape, and topology optimization. It is from these basic trusses that all topology optimization methods compare their initial runs to. An example of these is shown in Figure 2-5:

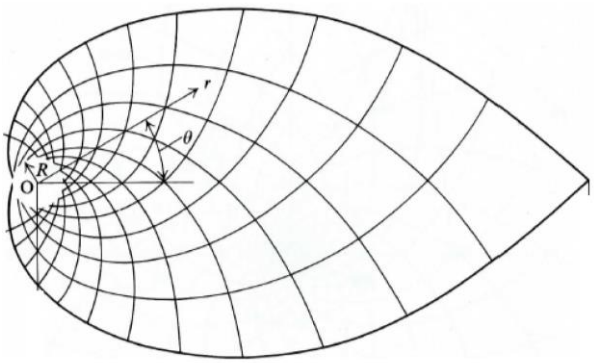


Figure 2-5: Optimized Cantilever (Michell, 1904)

In this example the lines for the Michell structure are generated by the minimum volume (V_{min}) of the frame such that the volume necessary is:

$$V_{min} = \boldsymbol{f}a \times log\frac{a}{r_o} \times (\frac{1}{P} + \frac{1}{Q})$$

Equation 2-10: Minimum Volume of Michell Cantilever (Michell, 1904)

Where (a) is the distance from the cantilever origin to the point of force at the tip, P and Q are the allowable stresses for tension and compression and r_o being the radius of the small bar from which the cantilever extends (Michell, 1904).

In all cases the minimum volume of the frame can also be rewritten for any generic frame in the following way:

$$V_{min} = \frac{\sum l_i|\boldsymbol{f}|}{P} = \frac{\delta W}{\epsilon P} = \frac{\sum \epsilon \boldsymbol{f} r cos\theta}{\epsilon P} = \frac{\sum \boldsymbol{f} r\ cos\theta}{P}$$

Equation 2-11: Minimum Volume of a Structure in the Generic Case (Michell, 1904)

In Equation 2-11, Michell (1904) states that the f is the applied force to the structure, l_i is the length of any bar in the frame structure, r is the distance of the applied force's application point to the fixed point, θ is the angle between between the line from the support to the load application point and the direction of the applied force, and P is the allowable stress in the material, ϵ is the strain in the structure, and δW is the virtual work.

This can be rewritten to state that the structure will minimize the virtual work, or maximize the stiffness of the structure based on the volume.

Michell structures minimize their volume by defining bar orientation along the shear slip lines also known as the Hencky-Prandtl Net. These slip lines describe the plastic flow of a 2D surface based on plastic deformation of the material along the principal shear directions (Strang and Kohn, 1983).

While Michell structures along Henky-Prandtl Nets are optimal structures under plastic yield, most structural design codes require the examination of structures under elastic yield. In order to optimize for elastic yield rather than plastic yield, bars are oriented along principal stresses directions as these directions have been analytically proven to be the optimum orientation (Brandmaier, 1970).

For this reason the thesis will focus on principal stresses rather than the Hencky-Prandtl Net.

Optimizing based on Michell structures also requires analysis of plastic deformation and defining the yield functions in the plastic region of Prandtl-Reuss flow (Strang & Kohn, 1983) which is costly and difficult to generate. Instead, principal stresses, provide an elastic analysis which is quicker to generate and requires less material analysis (Li & Chen, 2010). The comparison between the principal stress lines and the idealized Michell structure can be seen in Figure 2-6 below:

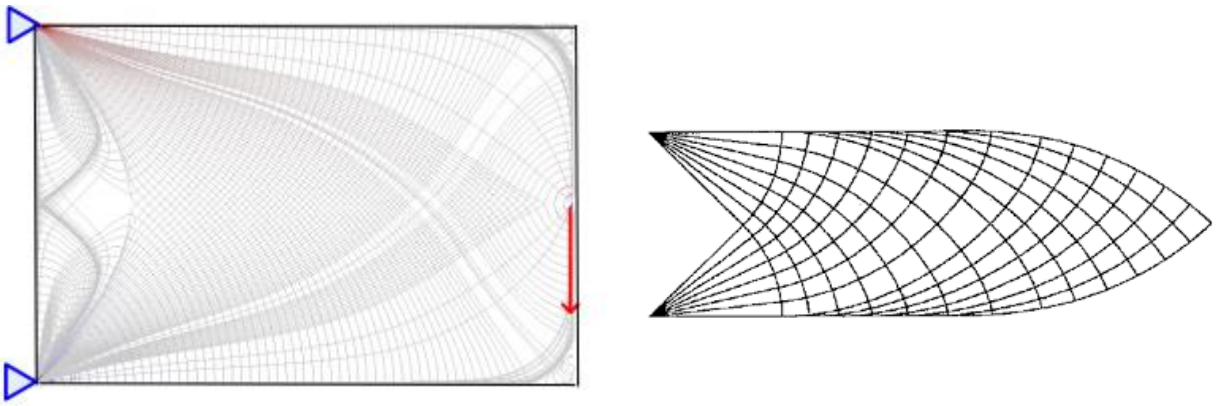


Figure 2-6: Comparison of Principal Stress Orientation (Left) with Michell Structure (Right) of a 2 Point Supported Cantilever with Point Load (Li & Chen, 2010)

By utilizing principal stress orientations, the optimization requires only the elastic analysis and allows for residual strength should the material begin to yield before total failure occurs while creating a highly stiff geometry.

The issue that arose was, how would a Michell structure develop for loads distributed across the entire domain and what are the other methods of developing these optimized systems? Considering Michell structures focus solely on supporting a stated load, what happens if each node itself becomes loaded in addition to providing support? Instead of trying to directly implement a Michell structure can we take the lessons learned about quadrilateral paneling along principal stress directions and apply it to other structures?

2.3.2 **Optimization Types**

There are three distinct types of optimization, sizing optimization, shape optimization, and topology optimization. These types are based upon how they fundamentally approach optimization. Each has a different set of boundary conditions, and optimize discretely based upon a variety of constraints.

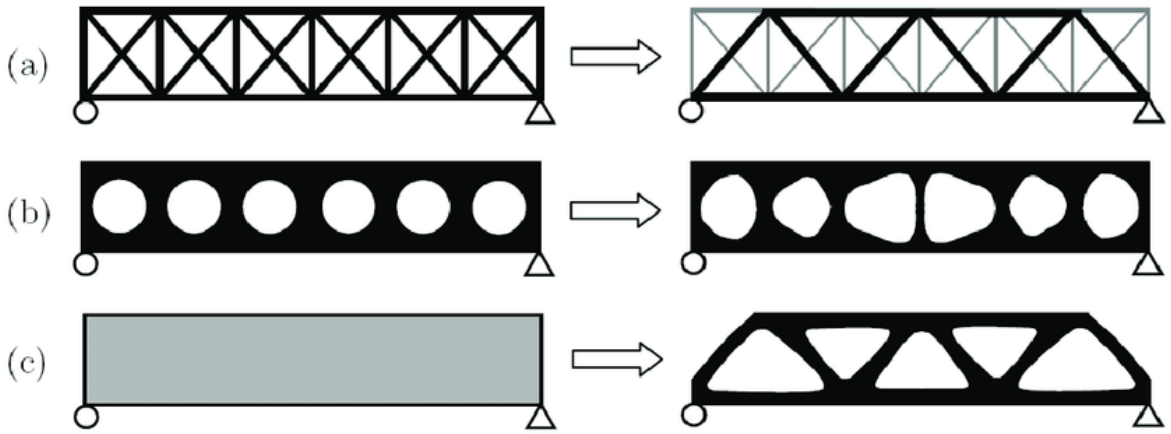


Figure 2-7: Types of Optimization: (a) Sizing (b) Shape (c) Topology (Bendsoe & Sigmund, 2003)

2.3.2.1 Sizing optimization

The first and most simple type of optimization is known as sizing optimization. Here the design domain is fixed to prechosen members. These members are then given different thicknesses based on a subset of variables, typically stress and deflection of the structure. By using these maxima and minima, the algorithm forms the new optimized structure. The stiffness method provides optimum criteria using a minimization of strain energy. This then allows for a reduction in either member size or complete removal of members until an optimum is reached such that elements do not fail.

2.3.2.2 Shape Optimization

The second method, known as shape optimization, allows for the members to move in a bounded space to position themselves in an optimal shape. From this boundary shape (domain), an optimum shape is found within the design domain such that minimal mass is used while examining the same variables in order to find an optimum shape for the member. This now rather than examining full members examines what are the elements and mesh in the FEA system, however, only around the specified area. A method of understanding this is that the nodes of the mesh defining the shape are essentially translated in 3 dimensions while the connectivity of those nodes remains the same.

2.3.2.3 Topology Optimization

Topology optimization, on the other hand is fundamentally distinct. The design domain limits the maximum volume that the object can exist in but does not provide a full constraint. Instead, the constants are the locations of loads, supports, and the bounding volume. By using this method, an entirely new form is generated based on minimizing the strain energy of the overall form. As shown in Figure 2-7, this results in shapes that provide optimum structural efficiency and generate these forms without the initial topology guesswork required for shape and sizing optimization.

Thus, topology optimization is a determination of how each element is connected and whether that element is required to exist within the form or not. This allows the routine to remove unnecessary elements and ensure that only optimum elements remain. Due to the power and simplicity of topology optimization, it has become vogue to use within a variety of projects. Current systems have been analyzing the use of topology optimization in shells, beams, and 3D objected (Winslow, 2010).

Topology optimization is currently done through a variety of methods. Each method approaches the minimum compliance of minimizing strain energy, shown in the equations at the beginning of 2.3. The most used methods are the following.

2.3.2.3.1 SIMP

The Solid Isotropic Material Penalization method or SIMP method uses the density of the material as a way of controlling for topology. By using Finite Element Analysis to generate individual elements, these elements are given a density based upon minimizing the strain energy of a structure given a specific volume fraction. This is shown in Figure 2-8 from Bendsoe and Sigmund.

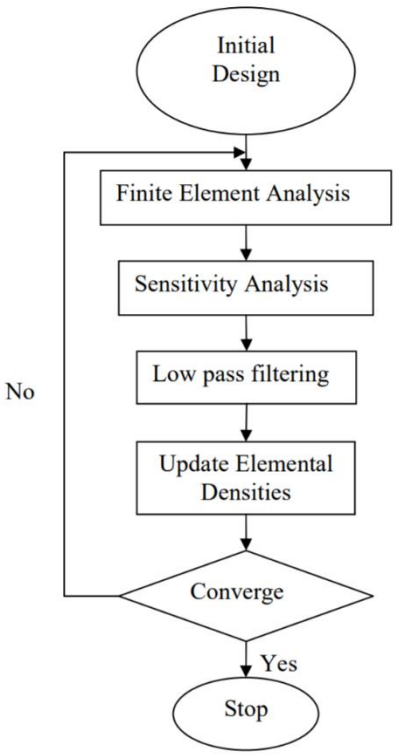


Figure 2-8: SIMP Process (Bendsoe & Sigmund, 2003)

Strain energy is the goal function, and is calculated as a summation of the deformation times the Young's Modulus of the material times a density factor determined by the program. Using this summation subject to a volume fraction, elements are then given density values ranging from 0 to 1 which is then penalized at an amount

dependent on the desired discretion. Using a discretion of 1 will in turn yield a full solid and fully void system where other systems will generate different values of grey. The volume was recalculated giving less dense elements a lower volume fraction such that $V^* = \rho \times V$ (Bendsoe & Sigmund, 2003) and was repeated until the strain energy stopped reducing or reached its convergence limit.

This method assumes that each element contains only isotropic material and, therefore, does not assume any directionality to the material (Bendsoe & Sigmund, 2003).

2.3.2.3.2 ESO

ESO or Evolutionary Structural Optimization works by removing inefficient material from a structure by examining the stress or strain values of the structure at that specific point. This is because low strain material is considered to not be acting under the load and therefore underutilized. This cycle continues until a steady state is reached or the rejection ratio is reached. (Huang & Xie, 2010)

2.3.2.3.3 BESO

The BESO or Bi-Directional Evolutionary Structural Optimization algorithm uses a different analysis logic and is non-gradient based. Due to using a variety of user subset functions such as inclusion and rejection ratios as well as target volume, the structural algorithm, in turn, leads to volumes not optimized based on materials and instead requires filtering a subset of sensitivities as shown in the logic chart in Figure 2-9. This in combination with the issues that BESO and ESO break down if the sensitivity to element density rapidly changes results in issues such as the algorithm only approaching local minima rather than a fully optimized structure. (Rozvany, 2008).

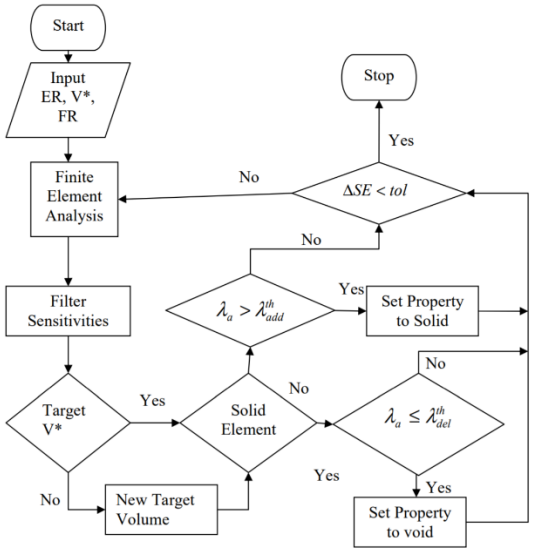


Figure 2-9: BESO Decision Process (Rozvany, 2008)

2.3.2.3.4 Why Topology Optimization for Gridshell Structures Does not Work

The design of shell structure is typically a function of tessellation. This tessellation is used as a method of tiling a preset, single layered pattern, often triangles, along the surface. This is done because a tessellation typically creates a distinct one layer form that is fully connected.

Discrete Topology Optimization on the other-hand does not use a standard tiled element but instead uses a subset of pre-set lattices or connections. Because only

connections previously defined can be considered, a sub-optimal grid may be selected, as the optimal topology was not included in the base structure. This allows for structure to exist where structure should not be required, thereby reducing the optimization usefulness.

The method works very well in continuum mechanics because a prescribed volume can be used to generate a ground structure. However, for discrete topology optimization, this is not possible and, therefore, either a dense ground structure must be created or limitations in the allowable structural beam directions must be defined by the user. If too dense a ground structure is used, this also means that for a 2.5D surface, bars have the possibility of overlapping at places that are not nodes or existing at multiple heights which cannot be allowed in a single layer gridshell as shown in Figure 2-10.

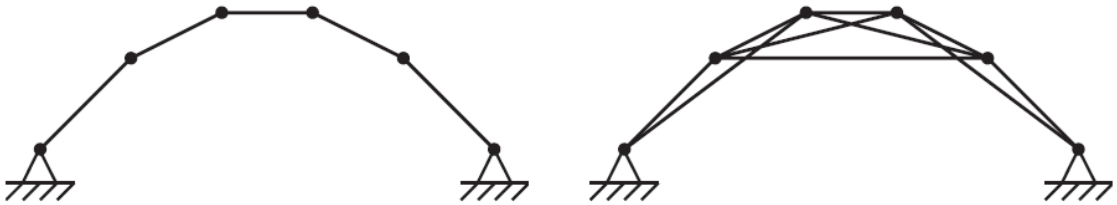


Figure 2-10: Issues with Developing Dense Ground Structures for Gridshell Topology Optimization (Richardson, Adriaenssens, Filomeno Coelho, & Bouillard, 2013)

Currently, topology optimization and sizing optimization have been investigated in gridshell lattice structures (Richardson, Adriaenssens, Filomeno Coelho, & Bouillard, 2013). However, the structures as shown in Figure 2-11, leave much to be architecturally desired. While these lattice structures reduced the weight of the initial structure by as much as 50 percent while still remaining kinematically stable, they do not visualize formal structural logic as would be expected. The structural check for kinematic stability results in a system that re-inserts kinematically necessary elements and, therefore, results in bar orientations with low stresses and finds a local minimum solution rather than a global minimum solution (Adriaenssens et al., 2014).

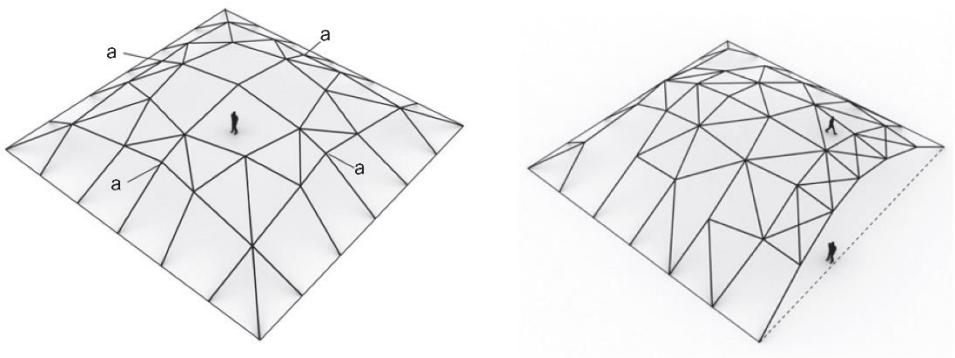


Figure 2-11: Topology and Shape Optimized Lattice Frames (Richardson, 2013)

2.4 DISCRETE DIFFERENTIAL GEOMETRY

Since the problem of a gridshell topology is defined as a subset of discrete bar elements and panels, a significant part of the issue is creating a developable divided surfaces. Discrete geometry must be applied to ensure that the bar and node structure maintain their system rather than continuous geometry typically used in shells. Here the main factor that is being examined is a primarily quadrilateral mesh that is determined via a series of vectors or tensors.

2.4.1 Conjugate vectors

Conjugate vectors are vector sets where at any point, the two vectors are always orthogonal. These vector sets are typically named within association of each other such that the dot product of the interior angles is always zero.

$$\boldsymbol{v_1} \cdot \boldsymbol{v_2} = 0$$

Equation 2-12: Definition of Perpendicular Vector Sets

These sets of conjugate vectors are common throughout mechanics, physics, and geometry. They occur often times in wave phenomenon with the wave crest always being conjugate to the direction it travels, they occur in the relationship between electrical and magnetic forces and fields, and they also occur often in curvature and stresses.

2.4.1.1 Stress Tensors

In 2D elements, principal stresses (σ_1, σ_2) are the maximum and minimum axial stresses that affect a finite element at any orientation. These are fundamentally determinable through a trigonometric calculations based off of the current axial stresses and the shear stress of each finite element. The principal stresses themselves are a rotation of this element to the point of zero shear with the directions of the force in compression and tension being the vector directions of the principal stress. Most commonly this can be seen in the equations shown in equations (2-13) and Figure (2-12)

However, as the loads change throughout the structure the principal stress values and directions change too. Yet, these principal values are what are useful for developing gridshells. This is because if the bars are oriented along a principal stress field, there is theoretically no shear in the quadrilateral bars, creating only axial forces, thereby reducing the strain energy on the structure and reducing the amount of shear in each node.

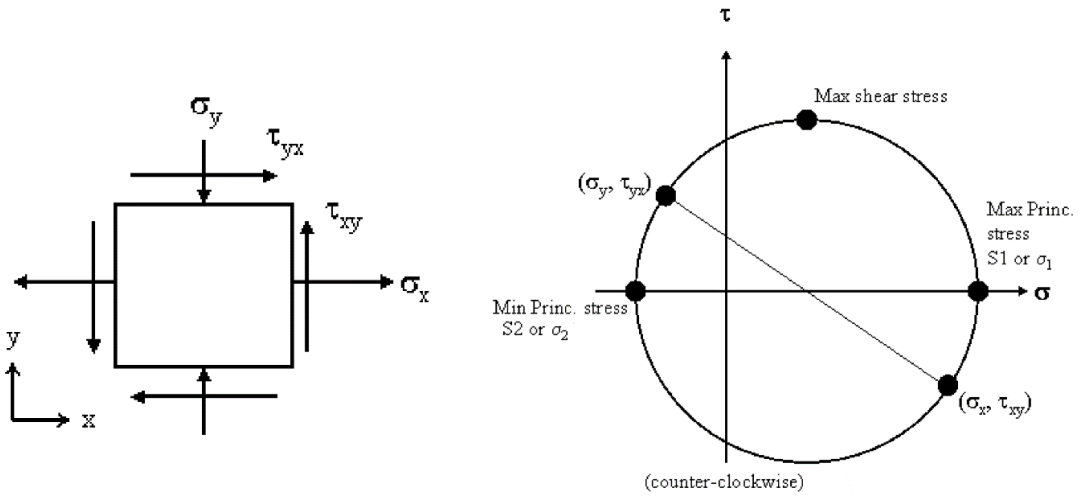


Figure 2-12: Mohr's Circle and Principal Stresses (Own)

$$\sigma_{1,2} = \frac{\sigma_x - \sigma_y}{2} \pm \sqrt{(\frac{\sigma_x - \sigma_y}{2} + \tau_{xy}^2)}$$

$$\theta = \left(\frac{1}{2}\right) \tan^{-1}(0.5 \times \frac{\tau_{xy}}{\sigma_x - \sigma_y})$$

Equation 2-13: Derivation of Principal Stress and Standard Method of Determining Stress Rotation (Li & Chen, 2010)

However, because of the way that these vector sets are determined, the directions are 2nd order tensors and eigenvectors as shown here on Mohr's Circle. This means that currently the only method to develop these directions and lines is through using discrete vectors on a surface and interpolating along them. While this is feasible, it becomes very messy and very difficult on more complicated surfaces.

2.4.1.2 Curvature

While curvature analysis can occur over a continuum if the shape has a mathematically defined surface, as with a nurbs surface, many times such curvature analysis is not possible. In this case the analysis occurs at discrete points. Oftentimes these points exist so close to one another, across such finite elements that it can appear as a continuum, but in principle the analysis works very similar to that of the finite element method. From this curvature analysis, just as with the structural analysis, a subset of tensors can be determined much like the stresses. However, unlike continuum geometry, discrete differential geometry

must deal with meshes and therefore a standard directional derivative of a functional surface is not possible; instead the value must be taken based on angles pertaining to a vertex.

2.4.1.2.1 Principal Curvature

Principal curvature (c_1, c_2) are the two extremes of curvature at a specific position. The two are typically orthogonally related to each other and help define the shape. Principal curvature lines are useful in gridshell structures as beyond just an analysis tool. They are also used often as the beginning method for defining grid directions and location on a surface as well as determining whether the surface can be created. This is useful as it gives radial surfaces a radial UV pattern and surfaces that were highly one dimensional a linear UV pattern. In continuous geometry the principal curvatures are derivatives of the slope $(d\phi)$ of the surface in 3D space. This is then used to develop a radius value (R) which is equivalent to the radius of a circle if the arc were to continue, and is written as follows in Equation 2-14:

$$c_1 = \frac{d\phi_{max}}{ds} = \frac{1}{R_1}; c_2 = \frac{d\phi_{min}}{ds} = \frac{1}{R_2}$$

Equation 2-14: Definition of Principal Curvatures in a Continuum

However, these values are only viable with continuous surfaces. Unfortunately most gridshells are made from meshes and thus the principal curvatures are defined based on the mean (c_H) and Gaussian (c_g) curvatures (Meyer, Desbrun, Schröder, & Barr, 2003).

$$c_1 = c_H + \sqrt{{c_H}^2 - c_g} \; and \; c_2 = c_H - \sqrt{{c_H}^2 - c_g}$$

Equation 2-15: Definition of Principal Curvatures using Mean Curvature and Gaussian Curvature (Meyer et al., 2003)

When compared to the principal stress equations in Equation 2-13, one notices a distinct similarity between Equations 2-13 and 2-15. This similarity arises in the way these values are determined. Both principal stress and principal curvature are 2nd order directional derivatives of the original functions (forces for principal stress and the shape of the surface for principal curvatures). For this reason both sets are difficult to get direct results for and instead are often found implicitly or discretely through orthogonal vector sets.

2.4.1.2.2 Mean Curvature

Mean curvature (c_H) is defined as the average curvature of a surface, and is often written in continuous form as shown in Equation 2-16, where c_1 and c_2 are the maximum and minimum principal curvatures respectively:

$$c_H = \frac{c_1 + c_2}{2}$$

Equation 2-16: Mean Curvature in the Continuous Form

However, these are not known directly in discrete curvature analysis.

This method is not very useful for defining the look of a surface as everything from flat plates to minimal surfaces have a 0 mean curvature. However, it has use in discrete differential geometry and is therefore considered as part of an analytical tool. While Gaussian curvature is defined by the difference in the summation of interior angles from 2π, divided by the sum of the distributed area pertaining to the vertex, the mean curvature is directly derivable from the Laplacian Cotangent Operator.

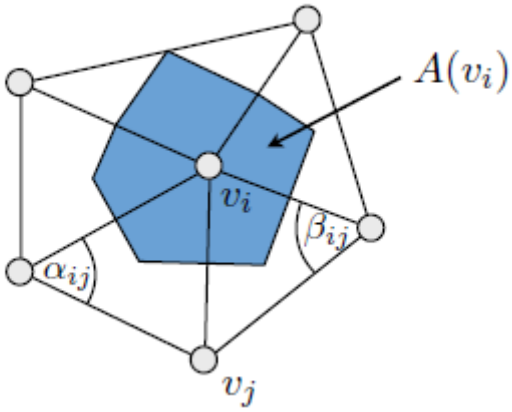


Figure 2-13: Cotangent Method for Developing the Laplace Beltrami Operator (source: computergraphics.stackexchange.com)

$$\nabla^2 f(v_i) = \frac{1}{2A(v_i)} \sum_{v_j \in N_i(v_i)} \left(cot(\alpha_{ij}) + cot(\beta_{ij})\right)\left(f(v_j) - f(v_i)\right)$$

Equation 2-17: Laplace Beltrami Operator via Cotangent Method (Meyer et al., 2003)

Where A is 1/3 the area of each triangle containing the vertex v_i with neighboring vertices and $v_j \in N_i(v_i)$ is every vertex connected to v_i.

Once the Laplace Beltrami cotangent operator is determined, the mean curvature at the vertex can also be easily found as the mean curvature is equal to ½ the magnitude of the Laplace Beltrami cotangent operator. Note this Laplace Beltrami cotangent operator is not the same as the Laplace adjacency matrix.

$$c_H = 0.5 \| \nabla^2 \mathbf{f}(v_i) \|$$

Equation 2-18: Mean Curvature Based on the Laplace Beltrami Operator (Modified from: (Meyer et al., 2003)

2.4.1.2.3 Gaussian Curvature

Most curvature analysis is conducted through the use of Gaussian curvature, which in continuous analysis is often defined as the multiplication of the two principal curvatures at a point (maximum and minimum). This analysis gives information on the typology of the curvature of the surface:

- If the Gaussian curvature is 0, the curvature defines a flat plate or parabolic surface

- If the Gaussian curvature is negative, the point at which the curvature analysis is being computed will be a saddle point
- If the Gaussian curvature that is positive defines a bowl shape in the shell surface or a sphere

These can all be seen as follows in Figure 2-14 where the Gaussian Curvature in the product of the two principal curvatures.

$$c_g = c_1 \times c_2$$

The change in Gaussian curvature is also a useful tool for form finding of shell structures as too great a change in curvature will result in significant moment at the point of change, therefore reducing the shell-like behaviors of the structures and is also useful when comparing twin shell theory from Calladine (Calladine, 1983).

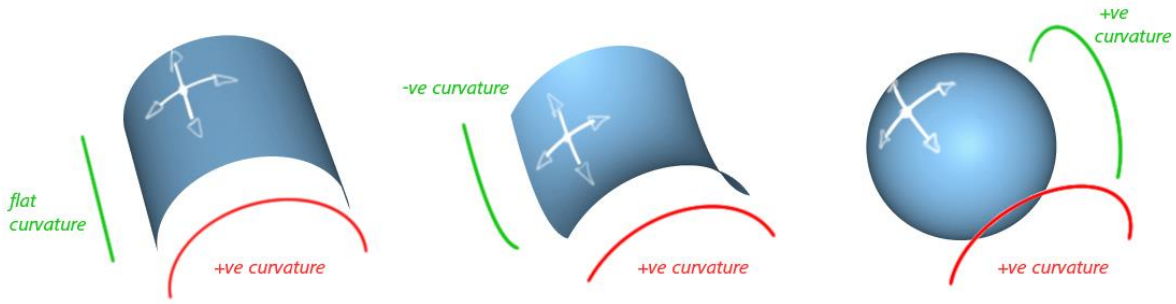


Figure 2-14: Curvature Typologies With Red and Green Lines showing c_1 and c_2 (Fritzsche, 2013)

Gaussian curvature can also be explicitly defined in discrete differential geometry based on the Laplace Beltrami operator mentioned above in Equation 2-18. Here the Gaussian curvature can be defined as the following:

$$c_g = \left(2\pi - \sum_j \theta_j \right) / A$$

Equation 2-19: Interior Angle Summation for Discrete Gaussian Curvature (Meyer et al., 2003)

Where c_g is the Gaussian Curvature, θ is the internal angle of the vertex, and A is 1/3 the area of the corresponding mesh face triangle. Since gridshell roof forms are developed in action against gravity, there should always exist positive Gaussian curvature throughout the whole surface. Any point at which the Gaussian curvature shifts from positive to negative or zero Gaussian

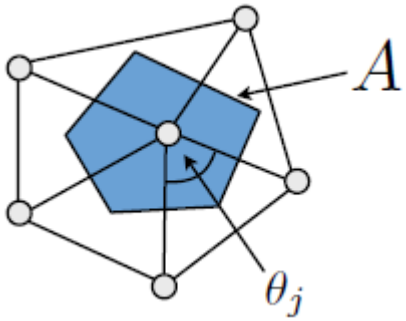


Figure 2-15: Summation of Internal Angles (source: computergraphics.stackexchange.com)

curvature defines an inflection point that would result in significant stress density.

2.4.1.3 Vector Fields

Vector fields are sets of values typically in 3D space giving direction and magnitude to a function at all points. Vector fields can be broken down into 2 parts, their direction and their magnitude.

In order to understand the different relationships that exist within a vector field, it is often best to break it down into the gradient (or direction) and the scalar field (or magnitude) which can be written as $\boldsymbol{\nabla} f = \boldsymbol{V}$, where f is the scalar field and $\boldsymbol{\nabla}$ is the gradient operator (or the vector of all partial derivatives) that exists on the scalar field to generate the gradient (Dawber, 1987). Such that:

$$\boldsymbol{\nabla} = \sum_{i=1}^{n} \boldsymbol{e_i} \left(\frac{\partial}{\partial i}\right)$$

Equation 2-20: Gradient Operator (Dawber, 1987)

Equation 2-20 shows that this operator is the summation of all directional derivatives of the unit vector (**e**) in each direction (i) for a vector field.

2.4.1.3.1 Divergence

Divergence of a vector field is the amount at which vectors point either towards or away from each other. A vector field with 0 divergence will have all vectors be parallel in that field.

The divergence of the field is then $\boldsymbol{\nabla} \cdot \boldsymbol{v}$**,** where if $v(x, y, z) = v_x \boldsymbol{i} + v_y \boldsymbol{j} + v_z \boldsymbol{k}$, then the divergence is equal to $\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} + \frac{\partial v_z}{\partial z}$ where **i,j,k** are unit vectors in 3 dimensional reals space (Dawber, 1987).

Convergence and divergence of a vector set from a line or point are what are known as singularities. Singularities become a major problem for parameterization as this means that the size of the parameterized elements as you approach the point become closer and closer to 0, resulting in overly dense meshes close to the singularity.

2.4.1.3.2 Curl

The curl of a field is also known as the rotation of a field such that in 2D space it is typically seen as the rotation of the vectors or the cross product of the gradient operator and the field. $\nabla \times \boldsymbol{V} = curl\, \boldsymbol{V}$ (Dawber, 1987)**.** This is often visualized as a rotation in the field and is often seen orthogonal to the field leading to singularities. As rotation around a specific point means that the orthogonal vector set will either converge or diverge on that point.

2.4.2 **Mesh Embedding**

Mapping and exploring the relationships between points through lines has been around since the time of Euler, where he famously tried to map out whether or not one could cross the seven bridges of Koningsburg only one time each and still cross every single one. However, before Tutte presented his paper in 1963 known as “How to Draw a Graph”, mapping and graph theory was not necessarily focused purely on straight lines and direct linear relationship. In his process Tutte algorithmically determines a proper projection of a complex surface based on network connectivity. This initial foray started the work into graph topology without using any curves. In this paper, Tutte (1963), goes on to describe an embedding known as barycentric mapping, or also known as spring theorem, in which the edges of a mesh could be modeled as springs that so long as the boundary was set and planar, with no holes (Kobourov, 2012). A standard example is that of a cube. By fixing the initial boundaries as a unit square, the position of the remaining vertices are calculated by linearly moving each position towards the barycenter of the connected points.

This works as follows:

Once the exterior is set to a convex boundary, the points are moved linearly towards each other as if they are zero length springs. This is defined the by average some of the neighbor’s positions or more succinctly as the barycenter or centroidal position. As long as the graph is at least 3 connected (containing 3 or more connections), the resultant graph will be a planar embedding. The iterative method shown below shows how this method is developed iteratively (Kobourov, 2012).

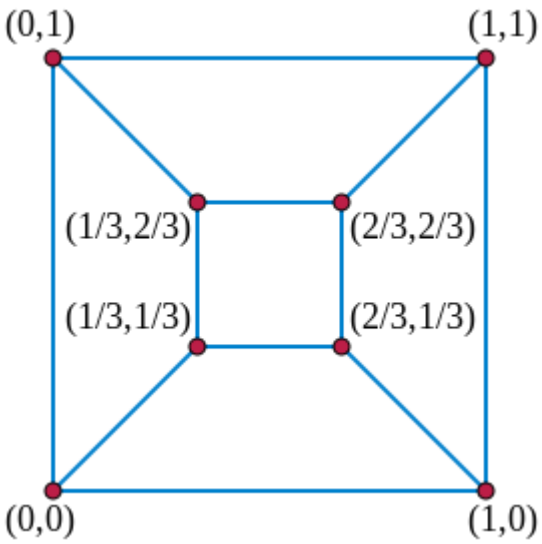


Figure 2-16: Tutte Embedding of a Cube (Source: Wikipedia Creative Commons)

$$Input: G = (V, E) \; where \; V = V_0 \cap V_1 \; with \; fixed \; V_0 \; \& \; free \; V_1;$$

$$and \; a \; convex \; polygon \; P \; with \; |V_0| \; vertices$$

$$Output: position \; p_v \; for \; every \; vertex$$

$$Initalize \; V_0: Place \; fixed \; verties \; at \; corners \; of \; p$$

$$Initialize \; V_1: place \; free \; vertices \; inside \; polygon \; at \; origin$$

$$Repeat \quad for\, i \in [0, |\, v_i \,|] do:$$

$$x_v \leftarrow \left(\frac{1}{\deg(v)}\right) \sum_{(u,v)\epsilon E} \left(p_u - p_v\right)_x$$

$$y_v \leftarrow \left(\frac{1}{\deg(v)}\right) \sum_{(u,v)\epsilon E} \left(p_u - p_v\right)_y \; until\, x_v and\, y_v converge\, for all\, free vertices$$

2.4.2.1 Planar Projections

Planar projections are linear mappings of a 3D surface onto a 2D plane via a direct translation or orthographic projection. This method of embedding is useful for gridshells. Since no two points can occupy the same (x,y) coordinate, a mapping to the plane of z=0 always provides a unique crossing-free solution. Since these projections are entirely linear, there is a much simpler remapping to occur in order to retransform from R^2 to R^3 space. This system matches closest to the Tutte Emedding system, except that here there is no energy minimization that occurs. For the thesis, planar embedding will be used to to examine the system, as this provides the simplest method for also modifying the stress direction vectors. Since for a gridshell, no points have the exact same X and Y values in real space a proper projection and embedding for most gridshells is to the XY plane. This results in Equation 2-21

$$n(x, y, x) \rightarrow n(x, y)$$

Equation 2-21: Vertical Projection Embedding for Each Point of the Gridshell to the XY Plane (Own)

This method does lose information though, and therefore angles are not perfectly preserved and mesh faces that are mostly vertical will become much smaller than mesh faces that are horizontal. However, it does allow for the easy transformation of the principal stress vectors as the stress vector location also mirrors Equation 2-21.

This also allows for an easy mapping of the principal stress vectors. As the principal stress vectors are tangent to the mesh face, this means that the principal stress vector directions can also follow the transformation such that Equation 2-22 also holds true.

$$\sigma_1(x, y, x) \rightarrow \sigma_1(x, y)$$

Equation 2-22: Transformation of Principal Stress Direction from Real 3D space to the XY Plane (Own)

2.4.3 Mesh Parameterizations

Since a comparison structure is generated using Periodic Global Parameterization, this section gives rough background information on the working of the mesh system to help understand why this method is effective and used in current quadrilateral meshing algorithms including Winslow's 2010 thesis.

In order to get surfaces that are used every day out of real space and into a 2D plane, meshes would have to be represented as the following

$$\mathbf{x}(u,v) = [x(u,v), y(u,v), z(u,v)]^T$$

Equation 2-23: $S \subset \mathbb{R}^3 \rightarrow \mathbb{R}^2$(Floater & Hormann, 2005)

$$\boldsymbol{x}_1 = \frac{\partial \boldsymbol{x}}{\partial u}, \boldsymbol{x}_2 = \frac{\partial \boldsymbol{x}}{\partial v}$$

The first fundamental form is an important characteristic of differential geometry in which the square of the element of arc of a curve on the surface creates the following equation.

$$ds^2 = x_1 \bullet x_1 \left(du^1\right)^2 + 2x_1 \bullet x_2 du^1 du^2 + x_2 \bullet x_2^2 \left(du^2\right)^2$$

Equation 2-24: Derivative Form of the First Fundamental Form (Floater and Hormann, 2005)

Or since these can be written in a symmetric matrix (**I**) by taking the transform Jacobian $\boldsymbol{x}(u,v)$ and multiplying it by the Jacobian of $\boldsymbol{x}(u,v)$ as shown in Equation 2-25.

$$\mathbf{J}^T\mathbf{J} = \begin{pmatrix} \frac{\partial x}{\partial v} & \frac{\partial y}{\partial v} & \frac{\partial z}{\partial v} \\ \frac{\partial x}{\partial v} & \frac{\partial y}{\partial v} & \frac{\partial z}{\partial v} \end{pmatrix} \times \begin{pmatrix} \frac{\partial x}{\partial u} & \frac{\partial x}{\partial v} \\ \frac{\partial y}{\partial u} & \frac{\partial y}{\partial v} \\ \frac{\partial z}{\partial u} & \frac{\partial z}{\partial v} \end{pmatrix} = \mathbf{I} = \begin{pmatrix} g11 & g12 \\ g12 & g22 \end{pmatrix}$$

Equation 2-25: Identity Matrix of the First Fundamental Form (Floater and Hormann, 2005)

This identity is then used to ensure that the transformation of the mapping is Isometric, Conformal, or another type of mapping.

2.4.3.1 Types of Useful Mappings

2.4.3.1.1 Isometric Maps

Isometric Maps are length preserving maps. This means that any line or dimension on S* is the same length as on the main surface S. This is can only be defined in this way if the two have the same fundamental form which also means that they will have the same Gaussian curvature at corresponding points on the mesh.

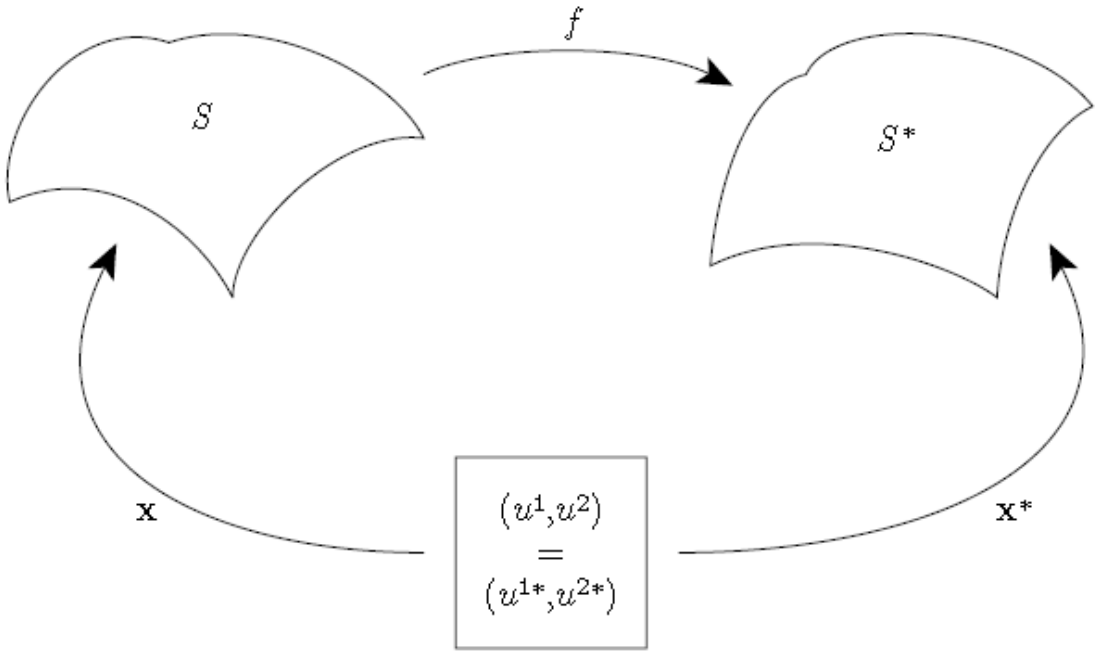


Figure 2-17: Mapping between S and S (Floater and Hormann, 2005)*

This means that for this to occur the mapping between the two surfaces must follow the following form:

$$\mathbf{I} = \mathbf{I}^*$$

Equation 2-26: First Fundamental forms I and I are Equal (Floater & Hormann, 2005)*

2.4.3.1.2 Conformal Maps

Conformal maps are angle preserving. This means that every angle on the surface S* is equal at the same point on S. One example of these is a Mercator projection.

In order for this to occur, the first fundamental forms must be proportional. This is typically written as a function of the U and V coordinates as in Equation 2-27, where the first fundamental form of the mapped surface is equivalent to the first fundamental form of the original surface times a constant.

$$\mathbf{I} = f(u,v)\mathbf{I}^*$$

Equation 2-27: Scaling of the First Fundamental Forms Between I and I (Modified From Floater & Hormann, 2005)*

This means for a surface to be conformal it must have a scalar function of the first fundamental form that holds true at all points. This does not mean the value has

to be constant, but rather that the value is scalar and is of some function the UV parameters $f(u, v)$.

2.4.3.1.3 Harmonic Maps

Harmonic maps are conformal maps where any mapping of (u(x, y), v(x, y)) can be represented based on the follow equations:

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0 \; and \; \frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} = 0$$

Equation 2-28: Requirements for Harmonic Maps

Harmonic maps also have the advantage to be quickly computed. As long as a decent boundary condition is chosen, a non-linear solver will try to solve the partial differential equation through energy minimization where the energy is the value of the difference in area and angle through the following equation.

$$U_D(f) = \frac{1}{2}\int_s (\left|\left|\nabla u\right|\right|^2 + \left|\left|\nabla v\right|\right|^2)$$

Equation 2-29: Dirichlet Energy Equation

Therefore, in theory, these mappings should be significantly more powerful than a planar projection that will occur during the thesis and will provide a solid comparison for between the two sets.

2.4.3.2 Periodic Global Parameterization (PGP)

Periodic Global Parameterization is a method developed by Nicholas Ray that creates a smooth mesh parameterization based on two orthogonal vectors (Ray et al., 2006). The vectors used are typically principal curvature, but can also be principal stresses (Winslow, 2010). This method creates a globally smooth conformal map aligning by taking a triangular mesh and overlaying it with a quadrilateral mesh that minimizes the angle difference between the mesh edges and the principal stress vector field.

The inputs required are a triangular mesh and the two vector sets that are required to be orthogonal to each other ($\boldsymbol{V}, \boldsymbol{V}^{\perp}$). Periodic Global parameterization is a quasi-isometric mapping that utilizes several deep functions to create globally smooth charts. To begin, the parameterization starts by smoothing out the vector field and eliminating curl. This reduces the number of singularities in the field thereby creating a simpler parameterization, however, this curl reduction introduces inaccuracy to the field in order to create a more uniform parameterization (Ray et al., 2006).

The global surface is a manifold, which is used to define the globally smooth parameterization and combination of multiple parameterizations of overlapping charts or sections by linking them with transition functions (Equation 2-30). In this case the whole surface or manifold shall be (S) and the sections shall be (C) with the second mapped section being (C')(Ray et al., 2006). Each chart has its own

mapping function to map them all to a different 2D domain that will be the global parameterization. The advantage is, as long as the 2D regions that intersect between each of the mappings is a topological disk, the sections are linked by a geometric transformation.

$$C \cap C', \varphi'(p) = \tau_{\phi \to \phi'}(\phi(p))$$

Equation 2-30: Transition Function between Mappings (Ray et al., 2006)

This scaling function ($\tau_{\phi \to \phi'}$) can be any combination of rotation, translation, or scaling in order for the transition from one surface section to the other to fit, so that in the end the gradient along the parameter space $(\nabla\theta, \nabla\phi)$ matches the vector directions as shown in Equation 2-31 (Ray et al., 2006). In other words, the parameter space is being warped to match the vector direction.

These sets can then be theoretically mapped as parameters in the two directions using the equation

$$\nabla\theta^T = \omega\mathbf{V} = 0; \nabla\phi^T = \omega\mathbf{V}^\perp = 0$$

Equation 2-31: Divergence of Parameterization in Comparison to the Vector Sets (Ray, et al, 2006)

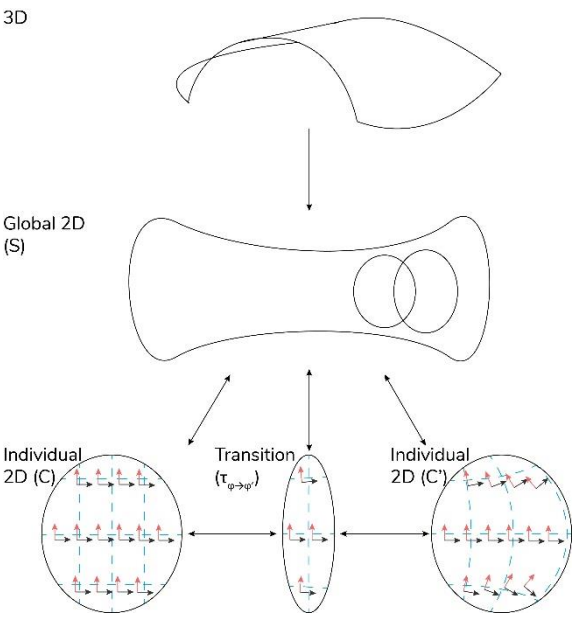


Figure 2-18: Process Used in Periodic Global Parameterization

In other terms, the parameterization lines match up perfectly with the vector field so that the tangent of the parameterization lines matches exactly with each frame of the structure.

However, since the vector fields are not without curl and do diverge, an energy function is instead introduced as written in Equation 2-32. This energy function is trigonometric function of the real parameterization and is used to minimize the difference between the UV parameterization on the Individual 2D chart and the vector map (Figure 2-18).

$$F = \min \int_S \left(\left\| \nabla\theta^T - \omega\mathbf{V} \right\|^2 + \left\| \nabla\phi^T - \omega\mathbf{V}^\perp \right\|^2 \right) dS$$

Equation 2-32: Modification of the Dirichlet Energy Function for PGP (edited from: Ray et al., 2006)

2.5 PRECEDENT AND EXAMPLES

2.5.1 Gridshells

2.5.1.1 British Museum Courtyard

Architect: Foster & Partners

Engineer: Buro Happold

The off-center reading room and tower provided a unique challenge to the design of the roof. In order to find the proper shape, the engineers used dynamic relaxation to ensure minimal structural thickness could be used before generating a smooth gridshell pattern across the surface.

(www.fosterandpartners.com, n.d.)

Figure 2-19: Interior of British Museum Courtyard (Source: Buro Happold)

Figure 2-20: Exterior of British Museum Roof (Source: Buro Happold)

2.5.1.2 *Smithsonian Courtyard*

Architect: Foster & Partners

Engineer: Buro Happold

A triple vaulted gridshell supported by 8 columns sits just above the rest of the Smithsonian portrait building creating a free standing and free flowing gridshell structure. Due to the orthogonal shape of the courtyard, a diagrid was used to develop a lightweight and complex roof. (www.fosterandpartners.com, n.d.)

Figure 2-22: Interior of Smithsonian Courtyard (Source: Foster + Partners)

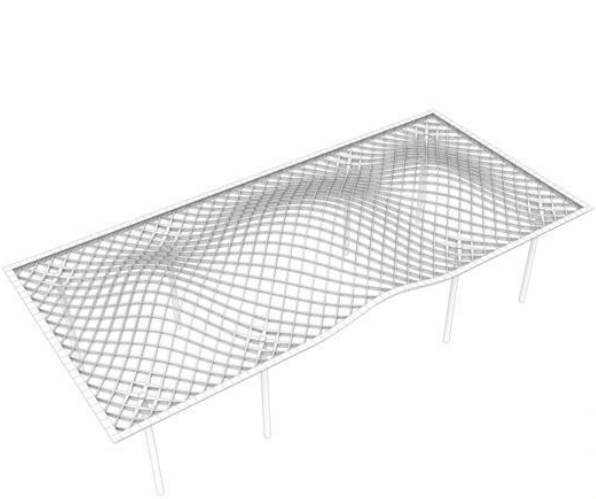


Figure 2-21: Interior and Structure of Smithsonian Roof (Source: Foster + Partners)

2.5.1.3 *Kings Cross Western Concourse*

Architect: John McAslan & Partners

Engineer: ARUP

"The diagrid shell structure of the new concourse roof spans to and is supported by perimeter tree columns and a central funnel structure – making it structurally independent of the sensitive Grade I-listed Western Range building.

The envelope and structure of the roof are fully integrated. This gives it both an elegant, natural form and also a modular, repetitive construction that helps fabrication and erection."("Redevelopment of King's Cross station - Arup," n.d.)

Figure 2-23: Western Concourse Top (Source: John McAslan & Partners)

Figure 2-24: Western Concourse Interior (Source: ARUP)

2.5.1.4 *Weald and Downland Gridshell*

Architect: Edward Cullinan Architects

Engineer: Buro Happold

Figure 2-25: Downland Gridshell Interior (Source: Buro Happold)

"In order to correctly map the structure and ensure each joint in the lattice was positioned in a way that would allow the wood the appropriate degree of curvature to form the gridshell, our engineers developed customized software based on the 'dynamic relaxation' technique. This interactive process of computer analysis aims to solve a set of nonlinear equations, a technique that is also used for the modelling and analysis of tensile structures.

Shaped into a giant peanut shell form, the undulating structure is an impressive 50 meters long, 12 meters wide and 10 meters high. The gridshell is based on straight lines that are crisscrossed and twisted to form a rotating grid plane. To achieve this unique outcome the team applied new techniques developed especially for this project." ("The Weald and Downland Gridshell - BuroHappold Engineering," n.d.)

Figure 2-26: Construction of Gridshell (Source: Buro Happold)

2.5.1.5 *Yas Hotel, Abu Dhabi*

Architect: Asymptote Architecture

Engineers: Schlaich Bergermann und Partner (SBP)

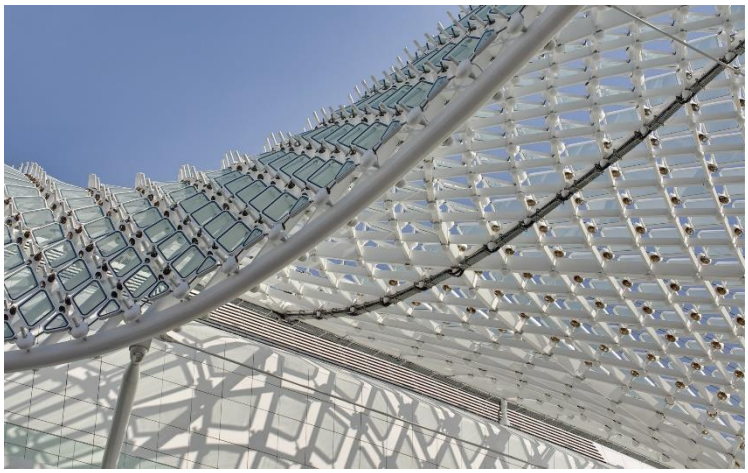


Figure 2-27: Yas Facade (SBP)

"Yas Hotel is a five star signature hotel next to the Formula 1 racetrack on Yas Island. The iconic glazed veil visually connects two buildings located on both sides of the racetrack. Its structural system is a gridshell that consists of the triangulated mega structure and the quadrangular grid. The free flowing gridshell is structurally one piece without any expansion joints and is supported vertically by 10 V-shaped columns. Due to its size and extreme temperature changes, the gridshell must be able to slide under temperature movements. Therefore eight out of ten supports are able to slide in one direction, with the other two acting as fixed supports. Wind loads are transferred to the concrete hotel structure by horizontal struts. It is covered by 5.800 pivoting diamond-shaped glass panels and features LED lighting at each node that allow for programmed lighting and image sequences over the whole surface." ("Yas Hotel," n.d.)

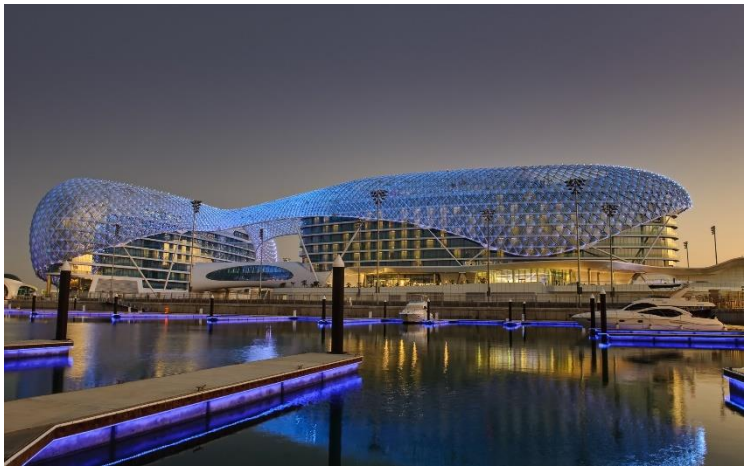


Figure 2-28: Yas Island Hotel (SBP)

2.5.1.6 Mannheim Multihalle

Architect: Frei Otto

Engineer: Ove Arup & Partners (Edmund Happold & Ian Liddell)

Figure 2-29: Mannheim Interior (Otto, 1974)

Originally a temporary pavilion, the Mannheim Multihalle was a timber gridshell spanning 60 meters by 60 meters. The form was found by using hanging chains and used a tetrahedral grid supported by cables to provide rigidity (Otto,1974). The frames themselves are four layers of timber connected with pins. This strained timber structure like the Weald and Downland Gridshell was developed from a lattice of straight timber that was bent into shape by lifting the gridshell and locking it into the exterior tension ring.

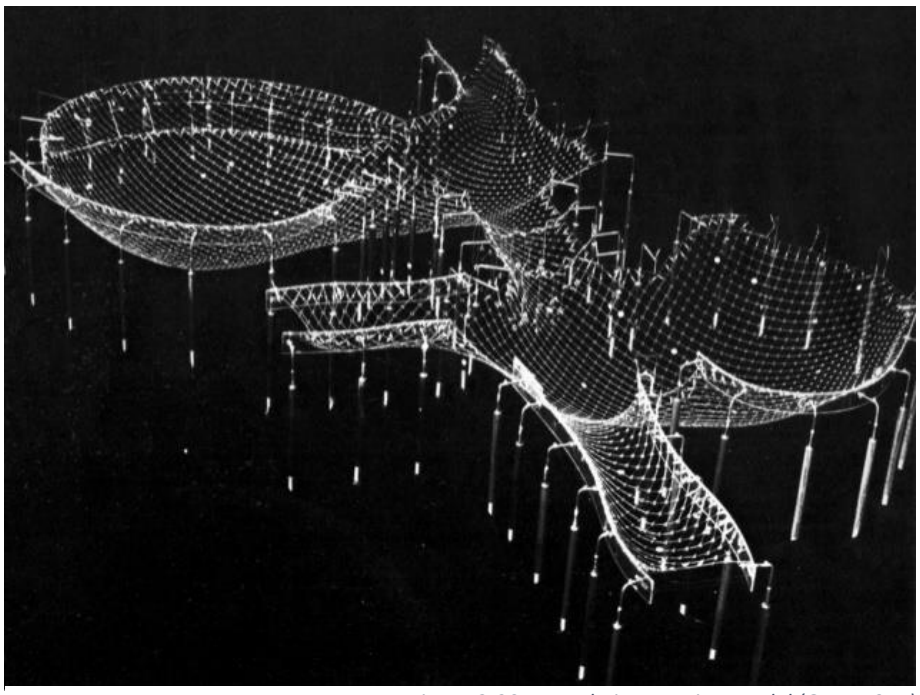


Figure 2-30: Mannheim Hanging Model (Otto, 1974)

2.5.1.7 Het Scheepvaartmuseum, Amsterdam

Architect: Dok Architecten

Engineer: Ney + Partners

"The roof of the Dutch Marine Museum courtyard in Amsterdam NL-had to offer an intrinsic added value to this historically preserved building. Based on wind roses of ancient marine maps of the museum collection, a steel structure is developed of 30m by 30m. This basic geometry was curved to a lightly bent dome that only transmits vertical loads onto the museum's existing walls" ("NEY & Partners | Projects | Glass roof Dutch Maritime Museum | 10915 | Amsterdam," n.d.)

The roof while based upon old navigation maps was also defined by a reciprocal Maxwell reciprocal diagram (Figure 2-33).

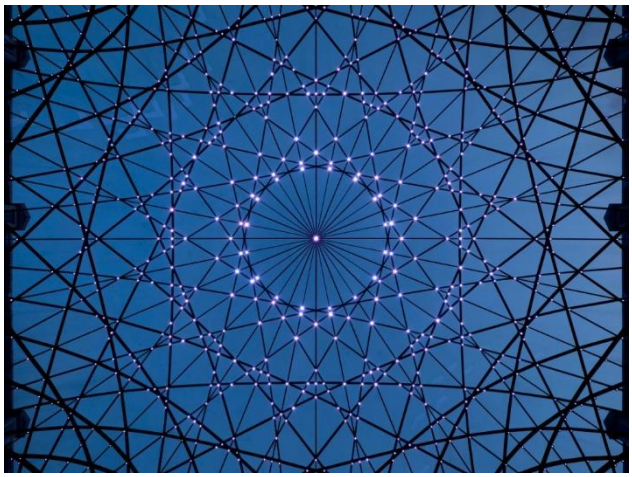


Figure 2-31:Roof Grid of Dutch Maritime Museum (NEY & Partners)

Figure 2-33: Dutch Navigational Chart (NEY & Partners)

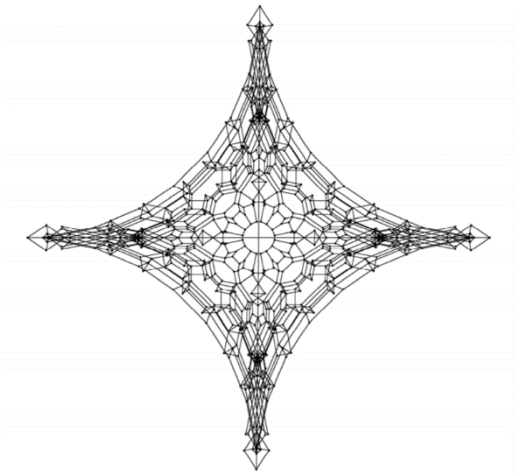


Figure 2-32: Maxwell Reciprocal Frame of the Dutch Maritime Museum Roof (Adriaenssens et al., 2014)

2.6 SOFTWARE AND CHOSEN TOOLS

In order to run the computation required, a set of software was chosen that facilitated the proper development of the form while also containing robust protocols for allowing the systems to communicate with each other. The main visual computing closet is developed in Grasshopper3D (Rutten, 2014). This is a visual modeling tool allowing for a set of plugins allowing for parametric geometry modification. This system is used to build generative algorithms and generate the geometry. Rhino5 (*Rhino* 5, 2017) runs as the 3D modeler for visualization.

There are several components and plugins used within Grasshopper3D for this script, the three most utilized are GeometryGym (Mirtschin, 2015), GSA (Oasys GSA, 2017) and Kangaroo (Piker, 2017). These systems are each slightly different but are used for the structural form finding and analysis.

Karamba3D (Preisinger, 2016), is used for final sizing optimization and final analysis of the structure. With GSA (*Oasys GSA*, 2017) being a more potent analysis program, it is used to develop the principal stress directions of the triangulated shell structures. While it seems odd to complete the setup in the method, GSA is a more robust tool that allows for proper analysis of the shell functions and generates a more robust vector field.

Kangaroo is a physics engine that allows for creating a spring system that is vital to generating a particle spring solution for optimizing the initial shell structure. It is also used to develop a proper Tutte Spring Mapping Algorithm for the purposes of this report.

Custom software is generated within Python. In order to perform discrete integration steps and apply some geometry processing such as support checks and the Euler Tracer itself.

3 THE DESIGN

Roof Design

With Building Technology and the School of Architecture not being only a research institution, but also a design institution, this paper will also implement a design scenario in order to showcase the method developed. The goal of the study is to verify the power of the tool across multiple boundary representation surfaces (breps) with varied support conditions.

To complete this task and integrate the space in a known location, the area in front of the main entrance to TU Delft Bouwkunde was used. At this point in time the current space is fully open to the elements and contains only a small covered front porch. This, in turn, results in a large open space that is rarely used due to the lack of protection against the elements despite its prime location. By developing a roofing structure in this method, current bike storage would be covered allowing for reduced weathering and erosion on the bikes and a covered space for students and faculty to use for outdoor functions regardless of the meteorological conditions. The site is shown in Figure 3-1 in plan with the location for the gridshell developed in a light blue.

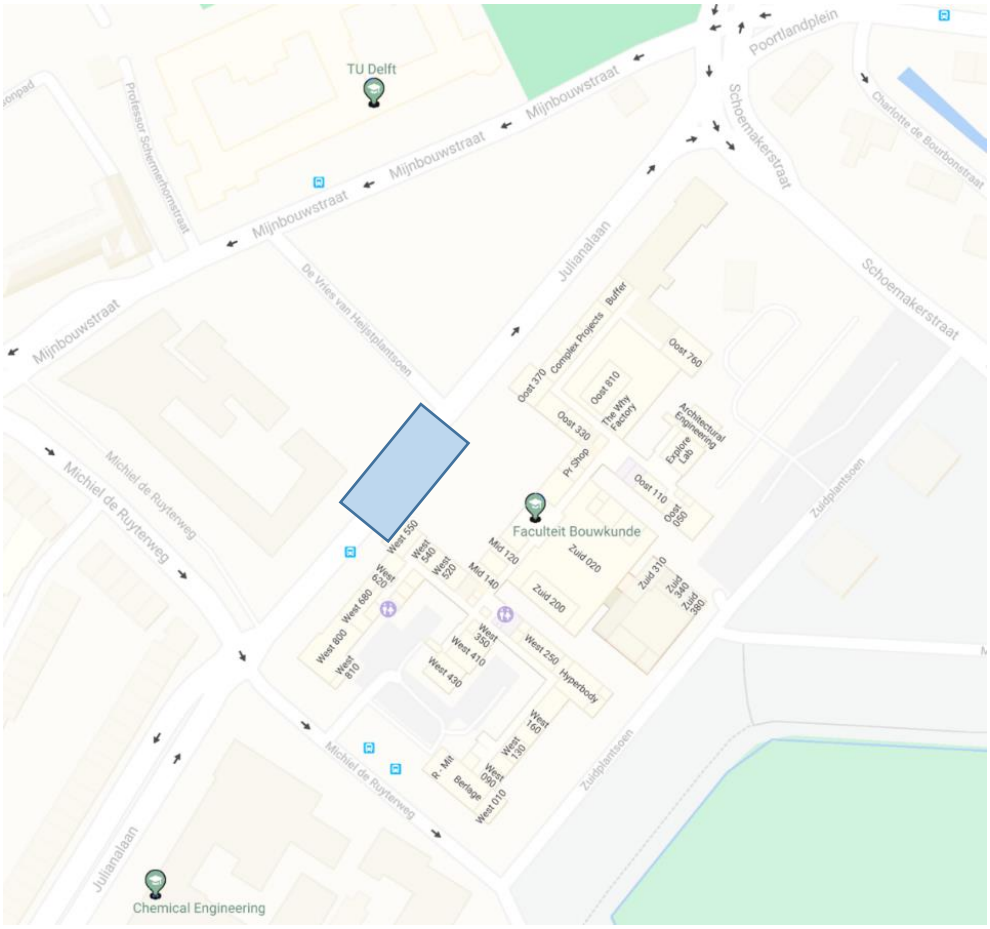


Figure 3-1: Map of TU Delft Bouwkunde and Location of Gridshell (Source: Google Maps)

The design of the Gridshell Roof for the main entrance cannot be defined as one surface. This is because, unlike a standard roof or gridshell, this structure is designed to mirror the entrance of the faculty and the ingress points.

This site must also contain few to no columns; not only do columns disrupt the flow of the gridshell, but they also break up the space further than necessary. Instead, the gridshell itself will connect to the ground at various positions in order to ensure that the system functions with diverse support criteria while also ensuring the minimal need for columns.

The gridshell must not have a predetermined topology as the topology is instead developed through the use of the script. This should result in a bar topology that follows the principal stress and force flows of the gridshell as much as possible while still being developable without relying on predetermined guesswork.

This site has several advantages for testing the validity of the script. By using two different main heights for the gridshell supports, the gridshell will have highly vertical sections creating distortions in most maps. The initial form is also complex enough of a shape to need to be developed by two polysurfaces. Since the gridshell will rest on the TU Delft Bouwkunde Building, there are several requirements that exist which simultaneously provide unique possibilities to develop irregular forms, for example, the brick walls will not be able to take moment and therefore the supports must be treated as a series of pins.

The initial area and space where the gridshell will be implemented is shown in Figure 3-2. The base plan allows for the Architecture sign still be visible from approach while ensuring students and faculty arriving through the front have clear ingress and egress points. The space that the gridshell would be placed in has one line of symmetry and therefore this symmetry will be brought into the design of the gridshell as well.

Figure 3-2: Photos of TU Delft BK Entrance (Own Photos)

4 THE GENERATIVE MODEL

4.1 MODEL AND DESIGN PATH: FULL MODEL SUMMARY

The parametric form finding, analysis, gridshell topology, and cleanup can be considered as one full loop containing several smaller loops within Figure 4-1. This section of the paper will discuss the full loop and the roles of each set with the form finding, gridshell topology, and other sections being discussed in each subsection. This section will also identify locations where the loop can be improved for future work and how this method could be further optimized.

This layout shows the four main sections that exist in this thesis. The start is the initial UV parameterization and Form Finding. This section is described in Part 1 and is completed on the quadrangular parameterized initial surface.

The second section is the FEA analysis and generation of the principal vector directions. This is completed on a mesh that is further divided into triangles using the loop subdivision.

The third part is the streamline generation. This method is completed on the two dimensionally parameterized surface. This includes the generation, discretization and cleanup of the gridshell.

Finally, the rationalization and final FEA analysis in order to determine the viability of this method in comparison to other methods is completed again in real 3D space. Here the final scores are determined from the volume, strain energy, and shell behavior of the final gridshell surfaces.

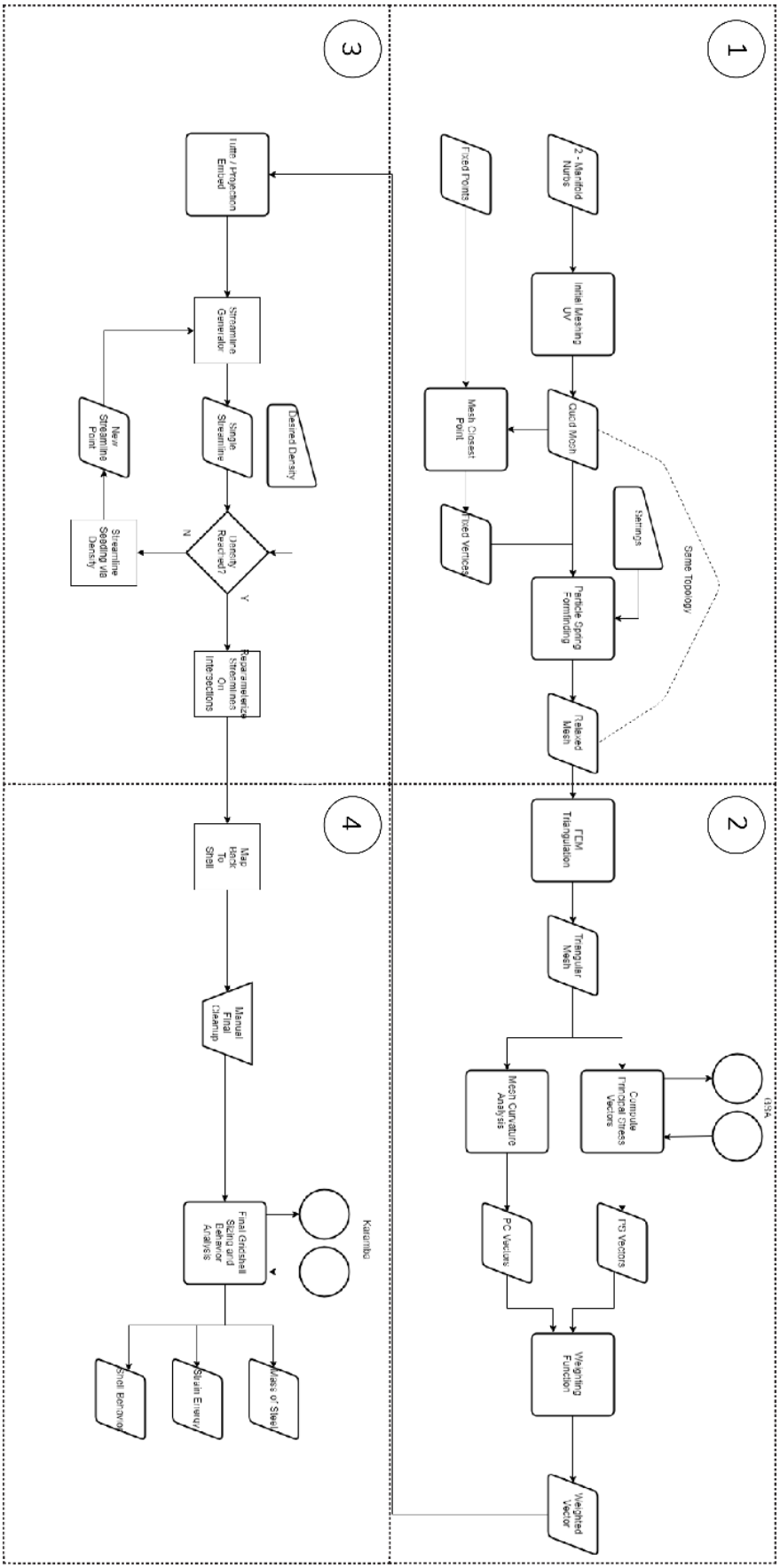


Figure 4-1: Layout of Generative Model

5 CONSTRUCTING A BASE STRUCTURE

5.1 INPUTS

5.1.1 The Supports

The support structure also had to be well understood from the perspective of designers. While in a physical gridshell structure every support is a point support, the goal was to make the script understandable from the perspective of a designer. In this case the designer has the option of setting line supports or point supports wherever the designer wishes to support the structure. By utilizing this setup, the script will then divide the line supports accordingly as to generate the proper support positions for the form finding.

5.1.2 The Shell

The initial chosen technical basis for the entire system was determined by what were the most usable inputs in order to develop a proper grid system. Within Rhino there are a variety of possible different methods to model a 2.5D structure like gridshells. The most common method is through the development of surfaces. Freeform surfaces are a useful input as they are derived from either boundary curves or a subset of curves used to create a sweep either through or along them. Unlike the meshing algorithms in Rhino, this method of surface development contains an inherent 2D structure known as a U,V coordinate that parameterizes the entire surface across a UV of {0:1},{0:1}. This inherent data set allows for a quick and efficient method of parameterizing the surfaces for the needs in form finding and mesh generation. This method of parameterizing the initial shape of the surface allows for architects also to readily enter multiple connected surfaces and allows the script to generate a continuous UV system over the length of the entire design.

With the supports and rough initial shape entered in by hand, the script generates a targeted UV meshing structure over the system and welds these UV meshes together before sending the output to the form finding algorithm.

5.2 HOUSEKEEPING SCRIPTS

5.2.1 Basic Mechanics Check

The script runs a quick check on the location and the number of supports. This is done by ensuring that all supports do not sit inline as this would guarantee the whole shell is statically unstable. The script is written in Python and is available in Appendix 2.

This instability is because any linear system will result in the ability for rotation around said line, and linear support systems have no way of dealing with off axis loads as they create unsolvable moment around the supports. While it is theoretical that if the area centroid of the shell sit exactly along the line of action that the form would be possible, any wind load would shift the line of action outside of the line of supports, therefore a minimum of any area must be given. The pseudocode is written as below:

```
Input SupportPos
Output Boolean
Define checkequal (pt)
    iterator = iter(pt)
    try:
        first = next(iterator)
     except StopIteration:
        return True
    return all (first == rest for rest in iteration)

If supportcount <3:
     print "Not enough supports”
     output = none
If supportPos is None:
     print "Not enough supports”
     output = none
If supportPos.count()>=3:
     print “You have enough supports”
Decompose Support Positions
     For I in range(len(supportpositions)):
          X.supportloc = supportpositions[i][0]
          Y.supportloc = supportpositions[i][1]
     If checkequal(X.supportloc) | Print (“Your supports are in a line”) a=None
     If checkequal(Y.supportloc) | Print (“Your supports are in a line”) a=None
     Else Print (“Your supports are well placed”) a=x
```

5.2.2 **Proper Initial Shell Form:**

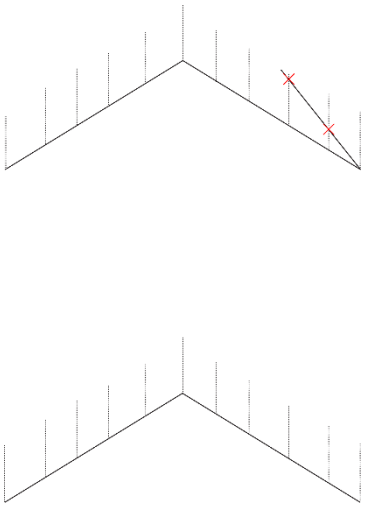


Figure 5-1: Method of Checking for Shell Folding in Initial Surface

The second check determines whether or not the initial form is a reasonable starting point. A shell structure is efficient through its geometry, in this case the forces throughout the shell flow directionally towards the edges and supports of the gridshell. As the form approaches the thrust network of the shell, the moment in the shell significantly reduces. However, if a structure were to be desired that loops back on itself, the structure would no longer be valid for shell analysis and form finding as significant moment would be generated in the curves exceeding verticality. Therefore, this subroutine checks to ensure that the desired surface has no points on the surface that are coincidental in the Z direction. This script was developed in Grasshopper3D. A quick example of how this works can be seen in Figure 5-1 where lines are drawn from the UV coordinate points upwards to check for any curve – mesh intersections.

6 FORM FINDING

Before the topology of a valid gridshell can occur, the form of the covered shell structure must be developed. The system chosen was a mass-spring model that models a digital system of a cable net structure. By developing this system, a dynamic catenary structure is developed that results in a near optimal shell configuration. As proper form finding is more important than that of the topology of the gridshell for minimizing the strain energy in a system and generating proper structural gridshells, significant time and focus was given to this subsection of the thesis.

6.1 CHOICE OF METHODOLOGY

Throughout the thesis several methods of form finding experimentation were conducted. These systems were then ranked based on a variety of requirements. The two main options were Dynamic Relaxation and Particle Spring methods. Both Dynamic Relaxation and Particle Spring Methodologies fall under what is known as dynamic equilibrium methods. Dynamic Equilibrium methods update velocities of mass spring particles with a defined bending resistance. Results from these systems were determined based on simplicity of development, accuracy, and level of customization for form finding.

6.1.1 Choosing Particle Spring Method

This system is a methodology that utilizes the same interactions as dynamic relaxation. Static analysis is conducted on a set of springs modeling the space that defines the area where a shell would like to be developed. Particle spring method was implemented using a physics simulator known as Kangaroo which linearly solves the position of particles under load until such a point that the particles have stopped moving and their position has converged on a final form.

	DYNAMIC RELAXATION	**PARTICLE SPRING METHOD**
SIMPLICITY OF DEVELOPMENT	Requires Bending Stiffness, Axial Stiffness & Ground Structure	Requires Ground Structure & Axial Stiffness
PREVIOUS IMPLEMENTATION IN IDE	Karamba	Kangaroo
LEVEL OF CUSTOMIZATION	Set Material Properties	Easily Modifiable Stiffness with Ability to Modify Stiffness Sets
WORKING ORDER	Works as Discrete Lines	Works as A Mesh

Table 6.1: Comparison of Kangaroo and Karamba

In the end, while dynamic relaxation is more realistic for predefined forms, since this system is needing to work with meshes throughout the entire process, particle springs with Kangaroo was determined to be the most optimal strategy. Particle spring method also works much more fluidly with the goal of generating a tool for architectural form finding as the user define setup can be limited to as little as a spring stiffness.

6.2 IMPLEMENTATION

The implementation of the form finding structure was developed based on the inherent UV structure of surfaces and polysurfaces that exist in Rhino. Depending on how the surface is developed, a standard UV system will be developed. Rails develop the system with U values parallel to the rails with V based values perpendicular to the rails. A lofted structure works with similar parameterization.

If a polysurface is desired with varying surface lengths and height, the V values are determined not through the length of the swept or lofted surface, but by bounding the surface and placing V values based on the Z-coordinate. This ensures that the mesh generates a set of values that are at equivalent locations to ensure that the mesh can be welded together.

When the springs are released from their position, $\sum F_{iz} \neq 0$ and the system starts to move in the z direction. This unbalanced force will then generate acceleration in the particles in correspondence with the particle spring system laid out in section 2.2.2, utilizing the Kangaroo simulation plugin with viscous damping. A convergence value of average movement within the time step function is less than 1e-10 meters was used and a spring stiffness of 750kN/m was used for the entire section.

6.3 SHELL SMOOTHING

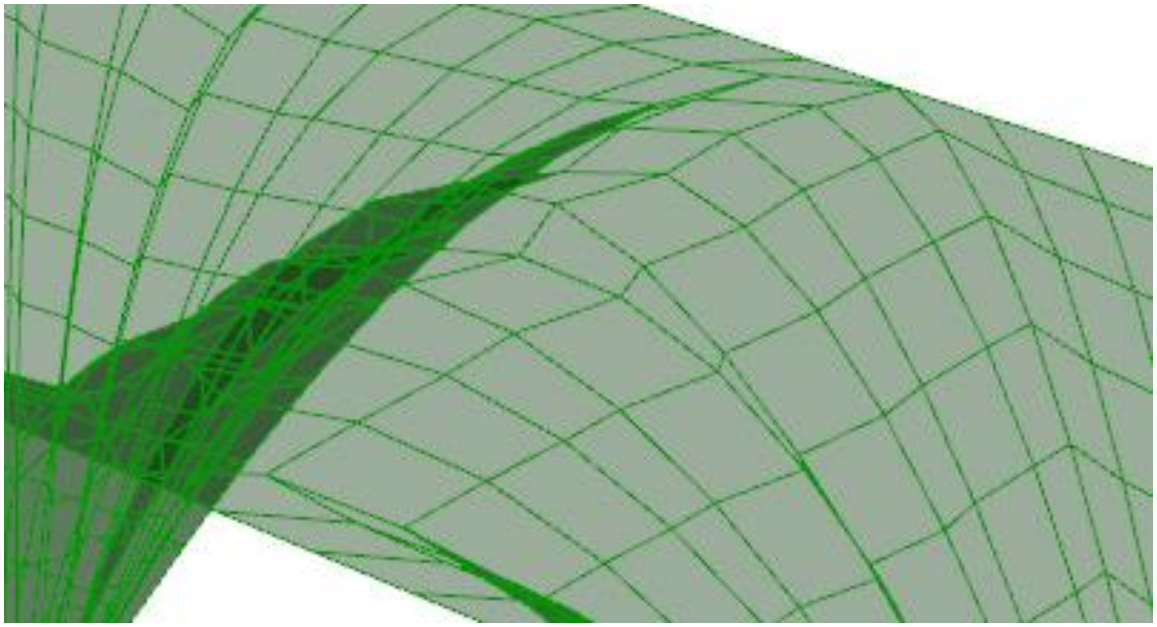


Figure 6-1: Form Found Mesh before Laplacian Smoothing

One major issue the method of form finding is that the shell form finding will generate kinks if the spring structure reduces in span. In order for these issues to be mitigated, some post processing was conducted on the shell system. Laplacian smoothing (Equation 6-1) was completed on the shell. Laplacian smoothing is a method of normalizing point locations based on the values of their neighbors such that each nodal point becomes the mean value of the neighboring points in 3D space.

$$\bar{\mathbf{x}}_i = \left(\frac{1}{N}\right)\sum_{j=1}^{N}\mathbf{x}_j$$

Equation 6-1: Laplacian Smoothing (Rework of Tutte Embedding Equation) [Hansen, 2005]

Where:

N = number of neighboring vertices

$\bar{x}_i$ = the new mean nodal location

x_j = the position of the neighboring vertices

This creates an average of the nodal positions and removes the creasing that occurs in shells with interior boundary conditions. With shells containing no interior boundaries, the effect is essentially minimal. Notice that the structure for position locations is similar to that of Tutte Embedding. It is also the same structure as the dynamic relaxation engine with zero length springs.

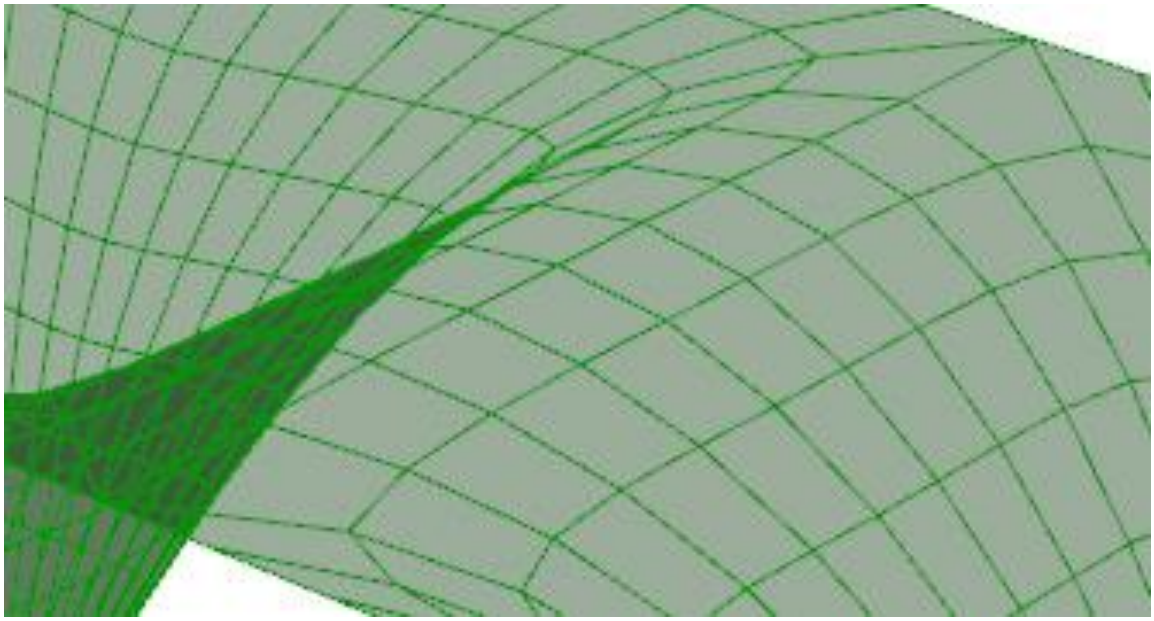


Figure 6-2: Form Found Mesh after Laplacian Smoothing

6.4 SHAPE SET FOR ANALYSIS

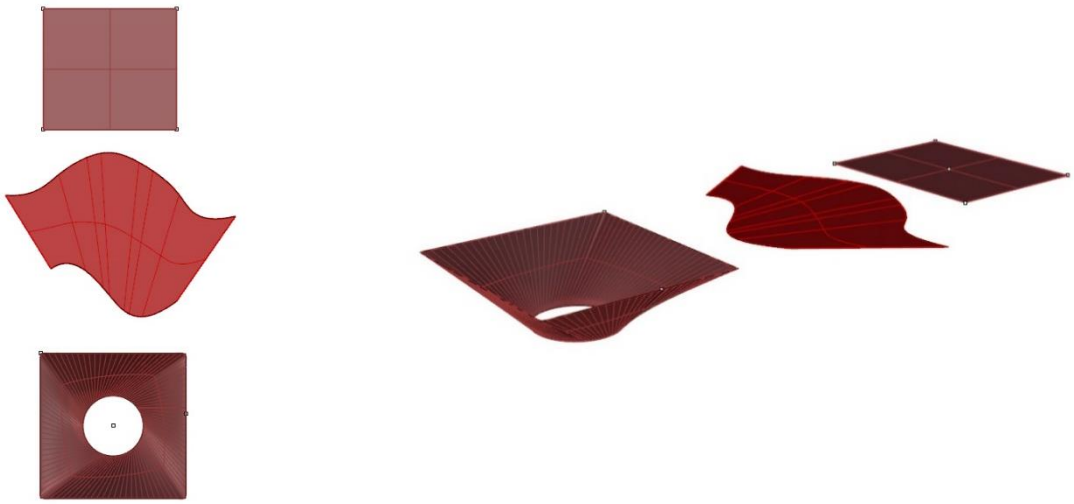


Figure 6-3: Visualization of Base Test Shells

The shapes seen in Figure 6-3 are used for the applied structural analysis. The first shape, which will be known as the Square Shell, show at the top is used to estimate the form finding with a structural system with 4 set points. The following structure in the middle is the structure used to develop a lofted surface with a full supported boundary, and will be called Shell with Fixed Boundary. Finally, the last structure developed uses a radial UV condition containing 4 different polysurfaces that are welded together as meshes, and will be called Conical Shell throughout the form finding section of the thesis. This system also contains different height support conditions in order to ensure that all variable methods of the system are tested.

The UV coordinates are then set as springs. These springs are set with an initial lengths in the given form being considered their rest length. Any forces acting on the points at the end of the springs will cause them to move, putting tension on the spring system in accordance to Hookes law, following the Particle Spring System.

Once this has been set, the springs are released from their initial positions and the points move until the sum of forces are equal. This equilibrium positions allows for the mesh to be consistently in tension with respect to the desired geometry.

Diagonal Springs or Quadrilaterals

Once the selected method has been chosen for developing the form, another method must also be examined for developing proper form. When finding form for shells, a hanging chain or spring method containing quadrilaterals is typically used. However, this results in forms that ignore the Poisson's ratio of the shell material. This results in two different issues, namely the difference in form and the introduction of tension into a structure. The goal of form finding is to fundamentally reduce these issues to create an efficient shell structure. The equations 6-2, show how the difference between the two subsets and how the diagonals are implemented. This method implements diagonal springs in data sets designated to mimic a Poisson's Ratio. The implementation of the UV springs (upper) as well as the diagonals (lower) are shown in Figure 6-4. For a more detailed understanding of the script, please reference Appendix 1.

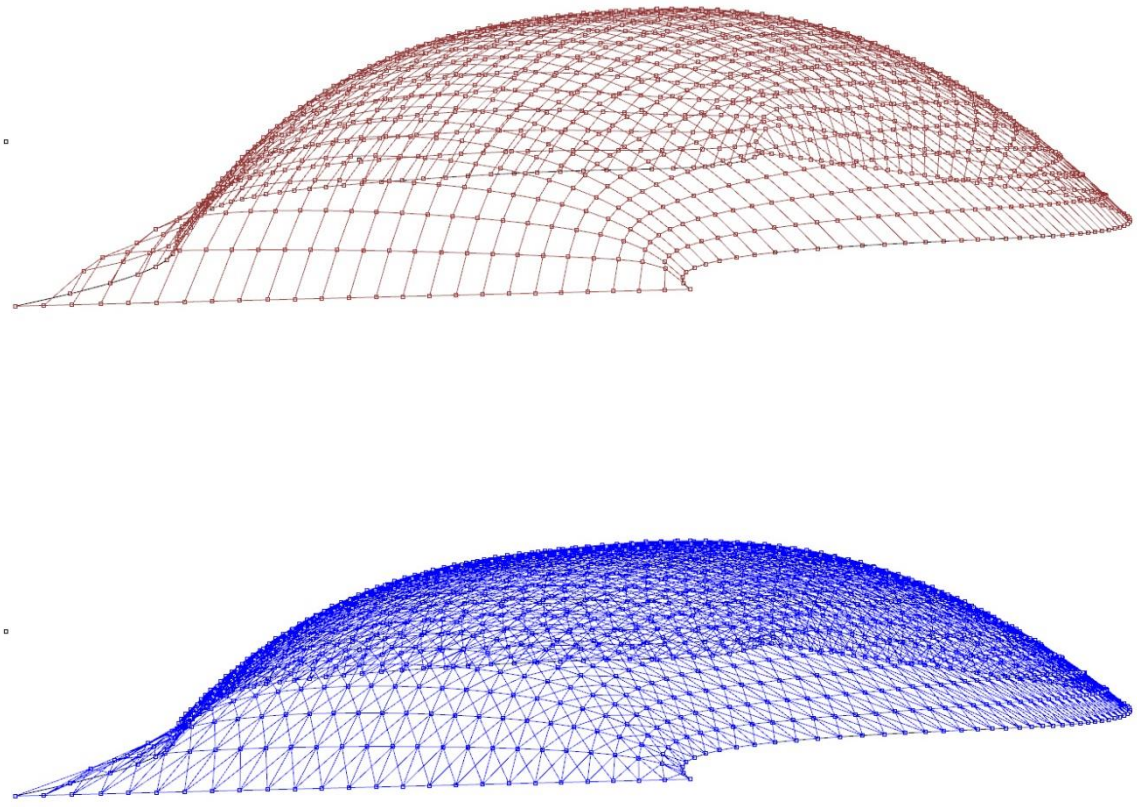


Figure 6-4: Differing Layouts of Springs on Fully Bounded Mesh Quadrangular (top) vs Included Diagonals (bottom)(own)

An introduction of additional springs brings new force sets into the equation as well. The introduction of additional springs now exist in the diagonals as well. Therefore additional nodal connections needed to be determined. This was done

by modifying the spring equations to include a diagonal forces ($\boldsymbol{f}_{i,d}\ in\ the\ x, y, and\ z\ diretions$) and including a separate spring strength ($k_{s,d}$).

$$\begin{aligned}\boldsymbol{f}_{\boldsymbol{i,d,(x,y,z)}} = k_{s,d} \\ \cdot\Bigg(&\frac{\left(\boldsymbol{n}^{t}_{i+1,j+1} - \boldsymbol{n}^{t}_{i,j}\right) - \left(\boldsymbol{n}^{0}_{i+1,j+1} - \boldsymbol{n}^{0}_{i,j}\right)}{\left|\boldsymbol{n}^{0}_{i+1,j+1} - \boldsymbol{n}^{0}_{i,j}\right|} \\ &+ \frac{(\boldsymbol{n}^{t}_{i-1,j+1} - \boldsymbol{n}^{t}_{i,j}) - (\boldsymbol{n}^{0}_{i-1,j+1} - \boldsymbol{n}^{0}_{i,j})}{\left|\boldsymbol{n}^{0}_{i-1,j+1} - \boldsymbol{n}^{0}_{i,j}\right|} \\ &+ \frac{(\boldsymbol{n}^{t}_{i+1,j-1} - \boldsymbol{n}^{t}_{i,j}) - (\boldsymbol{n}^{0}_{i+1,j-1} - \boldsymbol{n}^{0}_{i,j})}{\left|\boldsymbol{n}^{0}_{i+1,j-1} - \boldsymbol{n}^{0}_{i,j}\right|} \\ &+ \frac{\left(\boldsymbol{n}^{t}_{i-1,j-1} - \boldsymbol{n}^{t}_{i,j}\right) - \left(\boldsymbol{n}^{0}_{i-1,j-1} - \boldsymbol{n}^{0}_{i,j}\right)}{\left|\boldsymbol{n}^{0}_{i-1,j-1} - \boldsymbol{n}^{0}_{i,j}\right|}\Bigg)\end{aligned}$$

Equation 6-2: Additional Spring Forces Due to Diagonals (Modified from: (Harding, 2011)

Four versions of the form finding algorithm were used and then analyzed based on the strain energy per unit volume of the shell structure. Each system was analyzed using a consistent shell cross section and categorized based on the strain energy per cubic meter of shell and the average ratio of bending moment to in plane stress. This method was chosen as this averaging would allow for variations in shell area that are caused by the differing effects of the geometry of the springs. Since the flexural rigidity of plates is defined as:

$$D = \frac{Eh^3}{12(1-\boldsymbol{v}^2)}$$

Equation 6-3: Flexural Rigidity of A Plate (Beranek, 1972)

This means that the rigidity of a 0.1m thick steel plate would be 0.017GPa in comparison to a Young's Modulus or Tensile Modulus of 200GPa. Most deformations and strain created in a plate will come from bending stress and therefore using strain energy density of a plate structure is a viable method of measuring shell efficiency.

A second ranking (R) is performed based on the mean shell behavior ratio. The ratio is defined as the average of the maximum normal forces (n_{max}) to total forces. Where total forces are the moment (m_{max}) over the thickness of the shell (h) plus the normal forces in the element and therefore will be a ratio of the principal normal and principal bending moment as shown in Equation 6-4.

$$R_{mean} = \frac{\sum_{i=1}^{N} \frac{n_{\max,i}}{\frac{m_{\max,i}}{h} + n_{\max,i}}}{N} * 100$$

Equation 6-4: Shell Scoring Percentage (Oosterhuis, 2010)

Each system was then meshed using the Weaverbird's Loop subdivision such that the largest triangle was less than 0.2 square meters (Piacentino, 2015). The reasoning behind this was twofold: to ensure the validity of linear solvers and to allow for curvature analysis. In structural FEA programs, triangular meshes are solved linearly. This linear assumption is used later on in order to develop the principal stress direction. This triangulation is also used to develop principal curvature of the mesh so that future research can optimize based on a weighting function between the two orthogonal vector sets.

The first method was a standard quadrilateral based structure with the form developed from traditional hanging chain models. These models are similar to those used by Frei Otto and Antoni Gaudi and used for form finding such as the Multihalle and La Sagrada Familia. This setup is significantly simpler and is generated upon any untrimmed mesh by using UV systems based on the dimensions of the gridshell area.

The second method develops a standard diagonalized structure. This structure is developed using the same springs as developed in the previous method, but also with an additional set of springs with their own spring stiffness ($k_{s,d}$). In this case the shell is form found with the diagonals having the same strength as the main springs. With the introduction of diagonal springs, there is a noticeable visual difference: the stiffening "wings" that are visible in most concrete thin shell structures with point supports are now also visible in the model. However, these 'wings' are overly exaggerated, and therefore create moment, introducing an increase in strain energy per unit volume of the shell.

The third method introduces springs at 45 degrees with a strength proportional to the main springs. This method utilizes a set of triangulating springs in two directions in order to provide stability and introduce both the effect of the Poisson's Ratio as well as shear stiffness. This can be seen as introducing a relationship constraint between the U and V springs, essentially acting as a low information Poisson's ratio by applying a force of $k_{s,d} \times \frac{\Delta l}{l_0}$ as accordance with Hooke's Law where $k_{s,d} = \frac{k_s\sqrt{2}}{2}$ ..

The fourth and final method utilizes the same diagonal springs, but in this method, the springs are given a different strength based on the geometry and Poisson's ratio of the material. Since the springs are drawn as diagonals, they must become a proportion of the Poisson's Ratio. Baudet et al. (2007) modeled these parameters for 2D representations in the following Lagrangian equations starting with the shear modulus.

$$k_{sd} = \frac{E}{2(1+\upsilon)} = G$$

Equation 6-5: Diagonal Spring Strength Based on Shear Modulus (Baudet et al, 2007)

In equation 6-5, (G) is the shear modulus of a plate, (E) is the Young's Modulus, $\boldsymbol{\upsilon}$ is the Poisson's Ratio, and k_{sd} is the diagonal spring constant strength.

When modified for non-square meshes, the shear modulus can be rewritten for a subset of any quadrilateral shear modulus as:

$$k_{sd} = \frac{E\left(l_0^2 + l^{\perp 2}{}_0\right)}{4 l_0 l^{\perp}{}_0 (1+\upsilon)}$$

Equation 6-6: Non-Square Mesh Stiffness Based on Shear Modulus (Baudet, et al. 2007)

Where:

l is the length of the springs in direction 1

$l^{\perp}$ is the length of the spring that is perpendicular to the first spring as each set of springs run along UV coordinates perpendicular to each other

By using this value, any strain on the quadrilateral elements thus results in a force perpendicular to the load plane with a load value determined by the Poisson's Ratio. This allows for the generation of forces acting perpendicular to the strain. However, this also adds resistance to the strain developed at a rate similar to that of standard materials. By utilizing this system, a more accurate depiction of the behavior of the material in a specific shape is generated rather than that of either far too high a Poisson's ratio a non-existent ratio.

	SQUARE SHELL STRAIN ENERGY DENSITY (KJ/M^3)	SHELL WITH FIXED BOUNDARY STRAIN ENERGY DENSITY (KJ/M^3)	CONICAL SHELL STRAIN ENERGY DENSITY (KJ/M^3)
QUADRILATERAL STRUCTURE	5.77E-2	1.33E-3	1.06e-3
STANDARD DIAGONAL SPRINGS	2.10E-2	2.16E-3	1.22e-3
GEOMETRICAL DIAGONAL SPRINGS	1.81E-2	1.94E-3	1.26e-3
SHEAR MODULUS DIAGONAL SPRINGS	1.58E-2	1.74E-3	1.29e-3
% DIFFERENCE BEST VS 2ND	**12.71%**	**23.56%**	**13.11%**

Table 6.2: Strain Energy Density of Quad vs Diagonal Spring Modeling Method

	SQUARE SHELL BEHAVIOR RATIO	SHELL WITH FIXED BOUNDARY BEHAVIOR RATIO	CONICAL SHELL BEHAVIOR RATIO	MEAN SCORE
QUADRILATERAL STRUCTURE	93.0	96.8	91.5	**93.78**
STANDARD DIAGONAL SPRINGS	87.2	96.7	92.2	**92.03**
GEOMETRICAL DIAGONAL SPRINGS	88.4	96.8	91.8	**92.33**
SHEAR MODULUS DIAGONALS	90.0	96.9	91.9	**92.93**

Table 6.3: Mean Shell Behavior Scores for Shell Modeling Methods

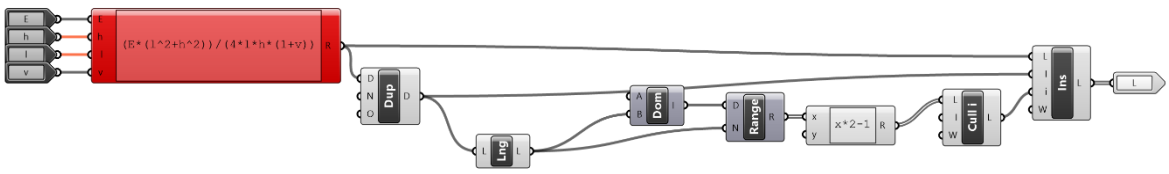


Figure 6-5: Shear Modulus Diagonals in Grasshopper3D

In the end it was determined that the most efficient method was dependent on support criteria. A surface that was supported with line supports around its entire perimeter modeled using the quadrilateral method was most efficient in reducing strain energy 23% over the closest comparison. However, when examining point supports, the most efficient method was the method of shear modulus diagonal springs by as much as 12.7% over the second place. The quadrilateral spring structure was also considered to be the most efficient method of modeling based on the shell behavior ratios. Based on this data, the rest of the thesis will continue to be based on the modeling from this setup. Since the goal shape is primarily supported along line supports, the model will be developed with a quadrilateral mesh.

Figure 6-6 shows difference in form created between the Quadrilateral structure (top) and a structure with a modeled Poisson's Ratio (bottom). Here it is clear that the introduction of the Poisson's Ratio has resulted in a flattening of the structure and that the edges of the shell are being lifted to stiffen against moment.

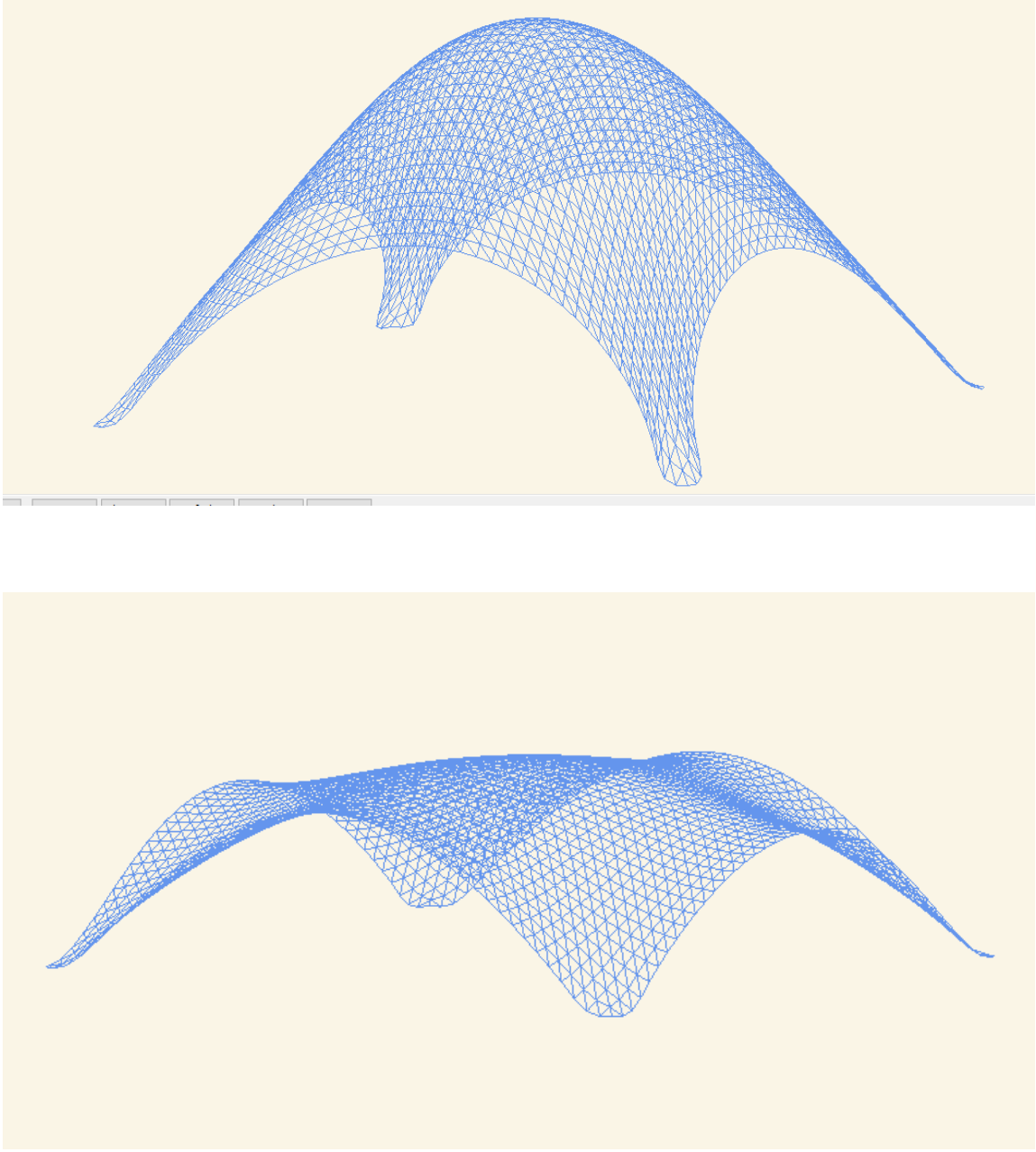


Figure 6-6: Difference in Form Finding Shape: Quadrilateral Springs (above) vs Shear Modulus Diagonal Springs (below) for a 4 Point Shell

6.5 Force Area Ratio

The force area ratio system is a script initially developed by Peter Eigenraam to generate a form where the load is based on the mesh area around each point. This system is developed in response to the initial method of calculating loading conditions for the form finding algorithm. The initial methodology and loading system determined loading based on area of the mesh structure projected onto a flat plane. By using this system, quick analysis can be made for determining rough initial forms. This, however, introduces inaccuracy as the shell structure becomes more vertical. Therefore, it becomes quite inaccurate for loading conditions of shell with dissimilar vertical support locations. In order to solve this issue, Peter Eigenraam created an updated method that rebuilds the mesh and calculates the area of loading based on the mesh face areas. This in turn provides a more accurate approximation of the loading behavior of the shell. The script as seen in Figure 6-7 is the process that develops this system.

In other terms, the original method of Kangaroo uses $A_{mesh} \perp$ rather than A_{mesh}. This results in unequal loading characteristics on parts of the shell which are more vertical. This can be corrected however, by factoring the loads being applied to the structure. This, however, does change the area being loaded and means that the previous results are not comparable to the new Force Area Ratio results. Therefore instead, the Force Area Ratio values are averaged and the mean value of the Force Area Ratio is applied to each node to establish the difference purely due to loading criteria. The version implemented in the Grasshopper3D script also determines the force area ratio a second time, after the form finding has started, in order to ensure that the system is not acting on a flat plate, but rather determining the mesh loading criteria itself.

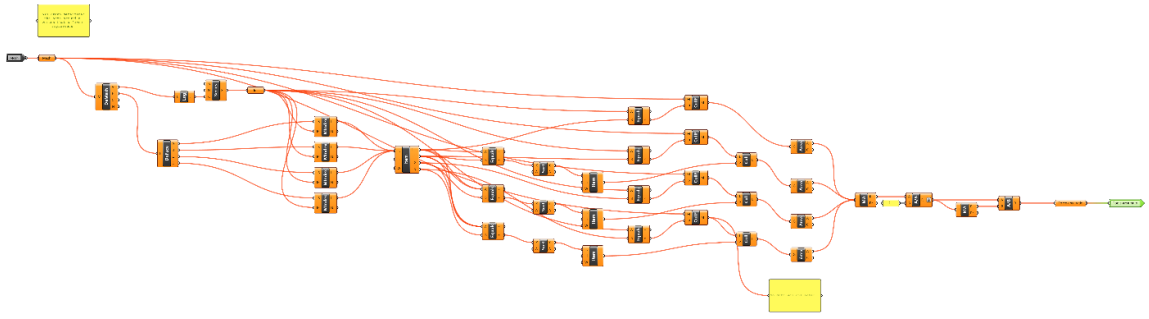


Figure 6-7: Script of Force Area Ratio

Table 6.4 shows that by utilizing this updated method of applying loading characteristics, the form found shells increase their efficiency by as much as 18.5% compared to uniform loading characteristics. This proves that just as structurally correct springs are required to create the proper expected forms, the Force Area Ratio method significantly increases the accuracy of form finding. However, with large datasets this module can become quite slow.

This is because the module divides each mesh face into 4 components, with each mesh edge midpoint is determined and connected to the opposite midpoint. The result is then 4 different faces all one quarter the size of the original mesh face. Each new face only has one of the original vertices. This face is assigned to the new vertex. This is done for each mesh face so that each vertex will have 4 new mesh faces. The area of each mesh face is then summed and assigned to each original vertex for loading.

	SQUARE SHELL STRAIN ENERGY DENSITY (KJ/M^3)	**SHELL WITH ADDED POINT SUPPORT STRAIN ENERGY DENSITY (KJ/M^3)**	**CONICAL SHELL STRAIN ENERGY DENSITY (KJ/M^3)**
BASE STRUCTURE	2.32E-02	1.26E-03	1.39E-03
FORCE AREA RATIO	2.34E-02	1.17E-03	1.13E-03
PERCENTAGE IMPROVEMENT	**-0.82%**	**7.18%**	**18.5 %**

Table 6.4: Shell Strain Energy Density of Normalized Shell Structures with Force Area Ration Implemented

These results show that the force area ratio method does indeed increase the validity of the simulation tool as a software package. The reduction in strain energy density is significant for the fully supported systems. As the mean shell behavior of the initial shell structures was already above 90%, further valuation using this methodology would gain little further information. As a result, mean shell behavior was not considered for the rest of the form finding methods. Instead, the strain energy density of the structure was deemed a more useful determiner.

6.6 LOOPING OF LOADING RATIOS

The looping algorithm updates the mesh area values of the force area ratio. This is done because the force area ratio method developed by Peter Eigenraam only applies loads based upon the initial form. While this is accurate for shells that start with an initial shape similar to that of the final shape, it does create significant distortions with large deformation systems. With the goal of the form finding system in and of itself being useful for large scale form finding applications and with architects often giving incorrect initial parameters, large scale deformation iteration must be both viable and accurate. The looping subroutine generated in this thesis updates this environment by applying a looping iteration as well as a termination condition. The script examines the updated mesh and compares the current area ratios to the previous iterations area ratios. The difference of each area ratio is taken and the absolute value of each set is calculated. This gives a normalized change in loading values. This normalized change is then delineated such that the script will loop up until this maximum change in loading area ratio for all mesh faces becomes less than one percent of the face area. By using this method, the final form found shape has significantly more accurate loading parameters based upon the final geometry rather than the initial geometry or generic loading requirements.

This algorithm works such that:

$$B < 0.01 = \frac{\sum \left| A^{t}_{i} - A^{t-1}_{i} \right|}{\sum A^{t}_{i}}$$

Equation 6-7: Convergence Determined by the Percentage Change in Mean Area (own)

Where:

A = area ratio of each mesh face

t= time step

i = mesh face index

B = convergence Boolean

This was considered converged once a one percent difference in the maximum change in force area ratio between each iteration was determined.

Due to the calculation time required to conduct the force area ratio analysis in Grasshopper3D, the looping analysis was conducted using a spring density of 0.5 times that of the other form finding experiments. This, therefore works as an independent experiment, and the results can be extrapolated based on the initial structural efficiency of the Force Area Ratio, but they cannot be directly compared.

	SQUARE SHELL STRAIN ENERGY DENSITY (KJ/M³)	**SHELL WITH ADDED POINT SUPPORT STRAIN ENERGY DENSITY (KJ/M³)**	**CONICAL SHELL STRAIN ENERGY DENSITY (KJ/M³)**
FORCE AREA RATIO	2.34E-02	1.17E-3	1.13E-3
LOOPED FORCE AREA RATIO	2.35E-2	1.19E-3	1.14E-3
% IMPROVEMENT	**-0.42%**	**-1.7%**	**-0.88%**

Table 6.5: Strain Energy Density (kJ/m^3) of Volume Normalized Shell Structures with Updated Loading Characteristics

Interestingly, the results show that the looping of the force area ratio calculations to update loading conditions to the updated geometry seem to create a worse performing shell than that of the shells generated with initial force area ration calculations. The value should at worst exhibit no change in the structural efficiency, so this result is an area for further study. Due to this and the fact that looping can take up to an hour to form find, the looping will be left out of the form finding for the final design.

6.7 Final Shape for the Structure

When examining the previous methods, the final form found structure was generated using quadrilaterals, with the force area ratio applied, and the looping structure not taken into account. This resulted in the final shape of the structure being developed as shown in Figure 6-11.

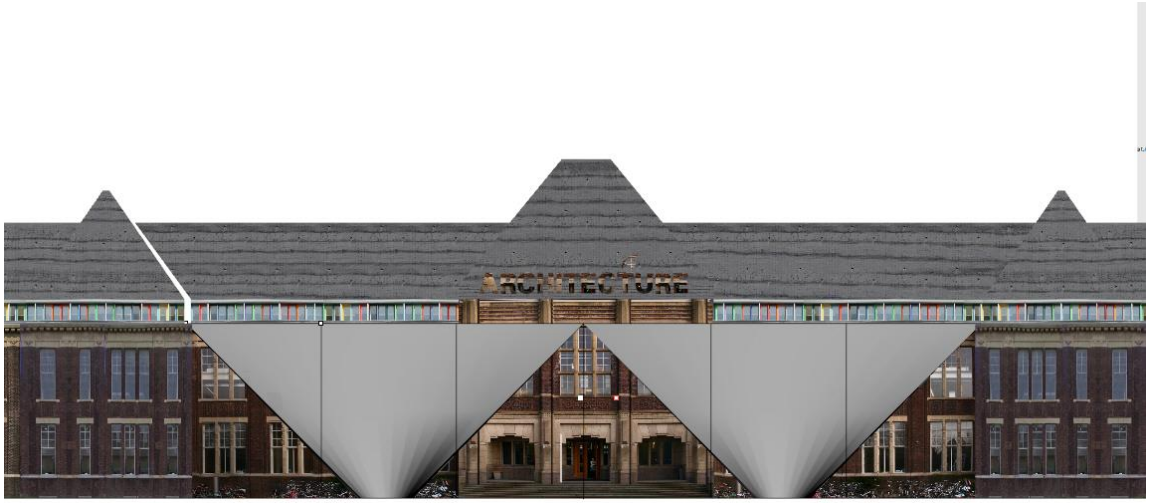


Figure 6-8: Base Polysurfaces for Gridshell

This system was creating using two sets of UV coordinates welded together along the center of the two polysurfaces. These two polysurfaces can be seen here in Figure 6-9 and show the initial rough shape.

This shape therefore is as the initial shape structure for the generation of the gridshell. This, in turn, allows for the creation of the structure in the following chapters.

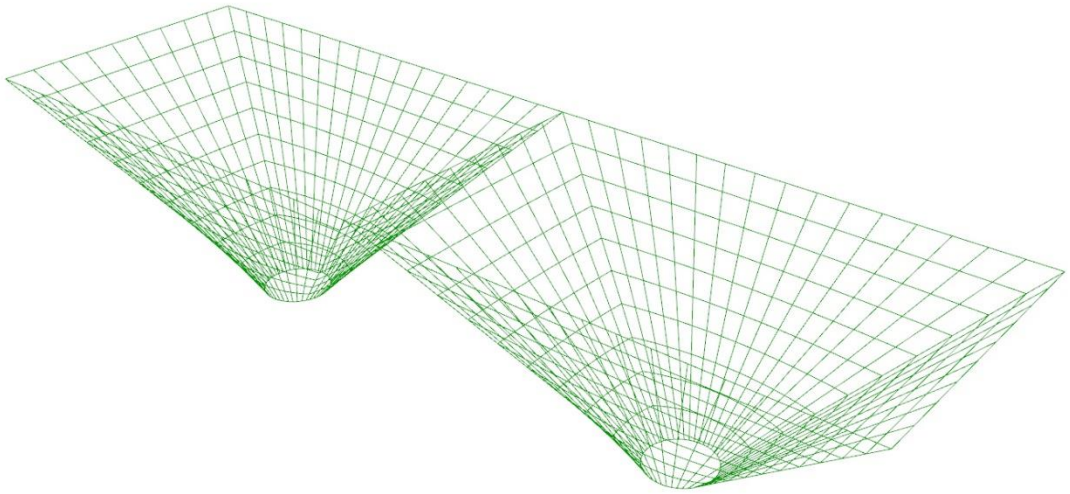


Figure 6-9: Base Discretized Layout

This shape was chosen to specifically test the limits of the streamline iterator with planar projection. Unlike Winslow's and Panagiotis' parameterized meshing

system, the streamline tracer does not efficiently transpose to highly curved shapes. The system works best on a relatively flat shell with supports all at the same vertical position, but by testing the edges of this system's boundaries, useful information can be determined about where significant improvements need to be made in the future. The initial form found shape can be seen in Figure 6-10 with the smoothened form shown in Figure 6-11.

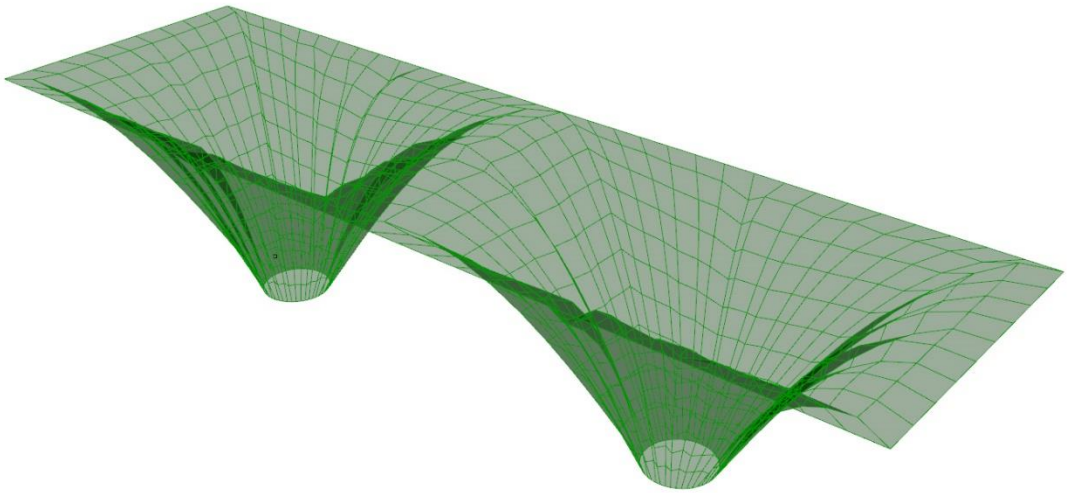


Figure 6-10: Shell Shape Post Form Finding

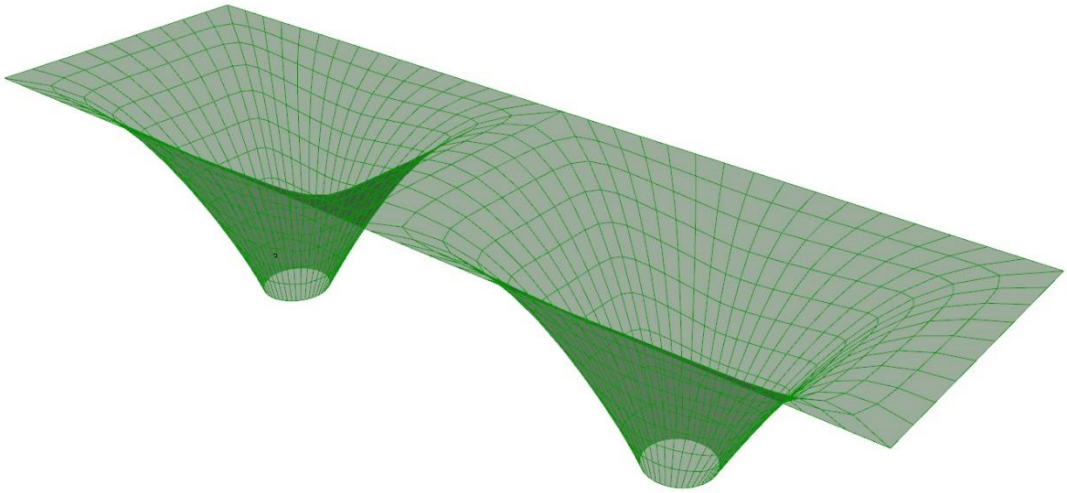


Figure 6-11: Shell Shape Post Smoothing

6.8 FEA Analysis

In order to develop a proper subset of principal stress directions, a Finite Element Analysis Engine was required. In this case GSA was chosen as the analysis system for shell principal stress analysis. Communication protocols are handled through what is known as IFC, or industry foundation classes, a standard communications package that many building industry programs use. In order to handle this part of the algorithm, the plugin GeometryGym (Mirtschin, 2015) was used with the model shown in Figure 6-13. In order to develop a proper gridshell model, the simulation needed to be run to develop the 3D principal stress vector sets.

However, the current mesh was far too coarse to develop a proper sampling of data, so the mesh had to be refined. The initial method was to utilize a method of rebuilding a surface through a set of patches. This works by defining the UV points used from the form finding to generate a smooth surface and allows for the use of continuum analysis on the surface to determine principal curvatures. Once the system was back as a surface, GeometryGym exported the brep to GSA which generated its own meshing for the structure.

This however, led to major issues when two different surfaces were too close to each other. Therefore instead the form found mesh itself was used and mesh refinement algorithms were examined instead.

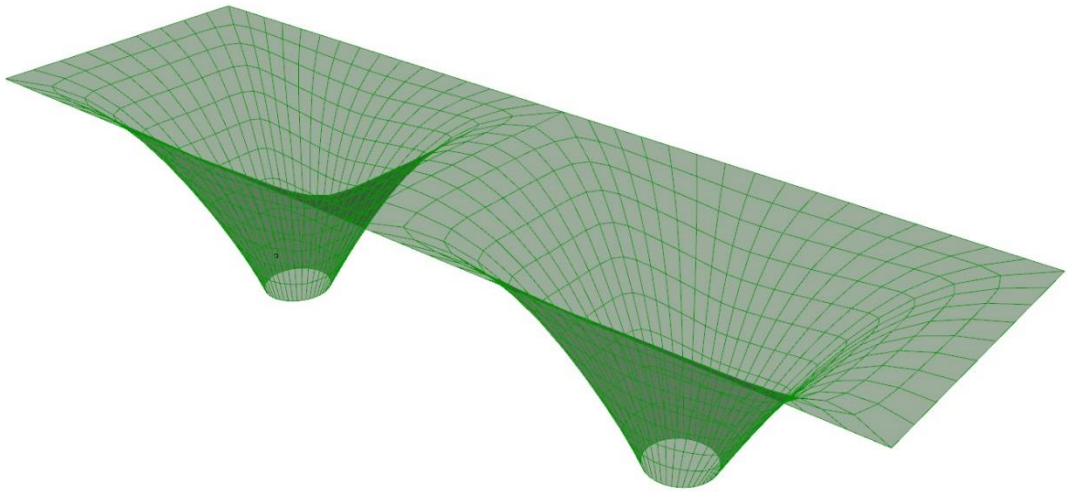


Figure 6-12: Mesh Before Loop Subdivision

In this case the mesh was divided into triangles by connecting the mesh faces edges from one set of diagonal points. Then each midpoint of each edge is divided and a set of four new smaller triangles area created. This was completed through the use of the algorithm developed by Charles Loop and implemented by Weaverbird (Piacentino, 2015). This produced a more refined mesh with many more vector points, allowing for a finer field that is easier to integrate over.

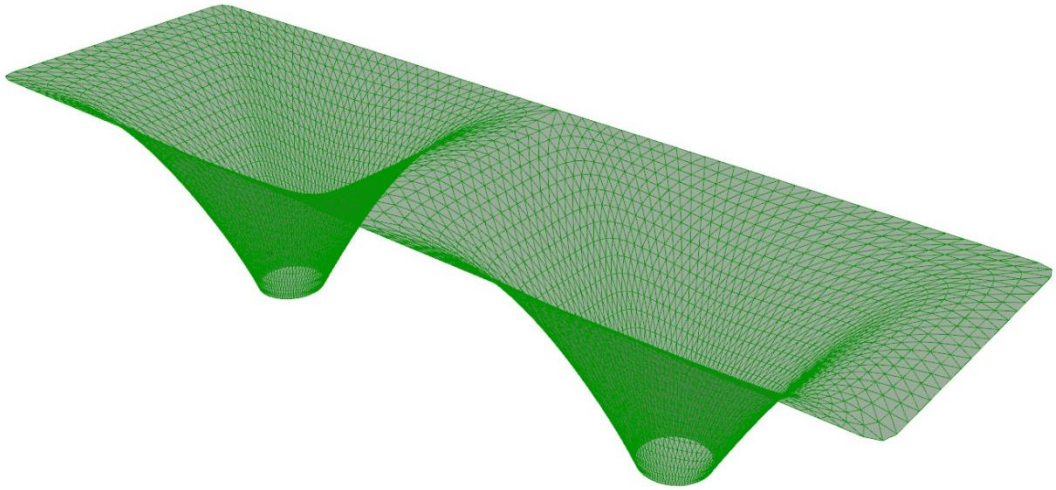


Figure 6-13: Mesh after Loop Subdivision

This was then exported to GSA via GeometryGym to query principal stresses based on a global gravity load and subsequently the data was reimported to Grasshopper3D as shown in Figures 6-14 and 6-15:

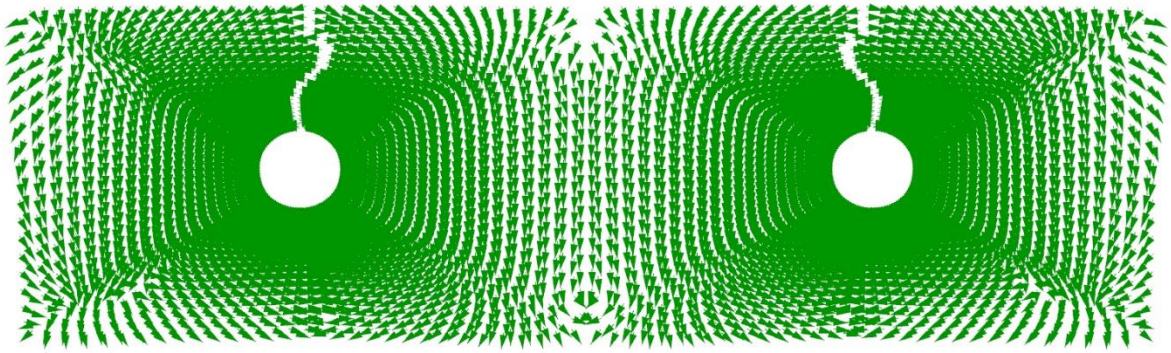


Figure 6-14: Principal Stress Directions for σ_1 Retrieved from Oasys GSA

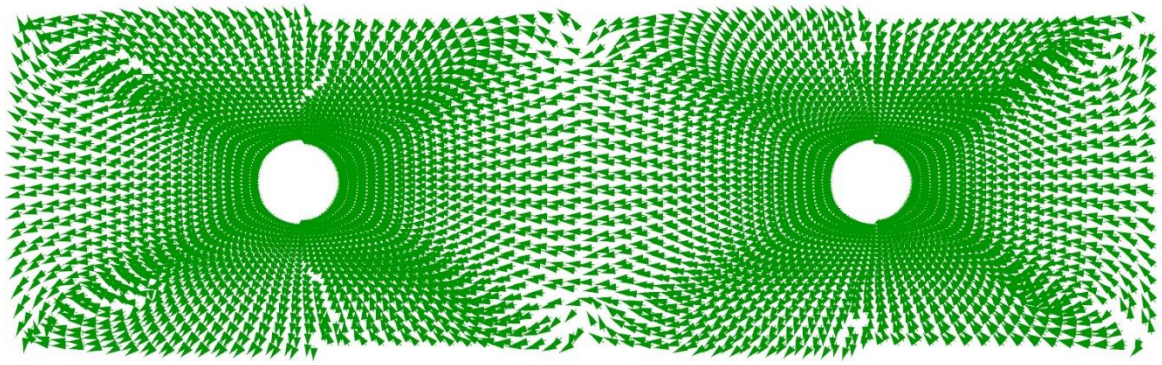


Figure 6-15: Principal Stress Directions for σ_2 Retrieved from Oasys GSA

The vectors ($\sigma_1 and\ \sigma_2$) and locations shown in Figures 6-14 and 6-15 were then used in the following steps in order to develop the streamline generation and Periodic Global Parameterization in the following chapters.

7 GRIDSHELL TOPOLOGY GENERATION

According to Pilkington Glass, the maximum span of 6mm tempered glass is 1.1 meters when a square panel is supported at its corners. Therefore, the spacing that is desired will be set to 1 meter. Panels could be longer with a one meter spacing in the primary direction; however, in order to develop evenly spaced grids, each quadrilateral panel will be developed as close to a square as possible. This will remain mostly true with some area and angle distortion on the projected gridshell tracing, with more curved gridshells having significantly more warp. This is one of the weak points of this methodology, hence a highly curved gridshell was chosen for the Euler Tracer to examine the effects of this warping.

7.1 METHODOLOGY 1: BASE GRID

The base grid is developed from the form found mesh which was based on the initial UV surface parameterization. Once the mesh is form found, the standard UV parameters inherent to the initial surface are then utilized in order to develop a maximum of 1m spacing of the grid in both the U and V directions. This base grid was developed to ensure that an initial structure could be used to examine the strength of the process and ensure the validity of the results. This method provides the layout shown in Figure 7-1 for the gridshell roof.

The three systems shown in Figures 7-1, 7-2, and 7-3 are developed from the base UV model and diagonals. These were then used as the baseline valuations to compare with the streamline and parameterization methods.

The simplest system was the quadrilateral system shown in Figure 7-1. This quadrilateral system is directly generated from the UV grid of the original surfaces and develops a simple quadrilateral based analysis. The quadrilateral structure is the same UV grid layout as used for form finding and therefore allows for the analysis of this grid without any further iteration work. However, this UV mapping for this specific gridshell should closely follow the principal stress vectors as it generates relatively direct lines between the supports with a stabilizing hoop system in the V direction. Therefore while this was the simplest structure to set up in this system, it should be also one of the most efficient.

One of the base grid forms is generated with the rationality of two point charges relating to one another. This results in a form that closely mirrors the principal stress orientations without any direct computational requirements. This makes this option incredibly interesting as it has the ability to be used as a ranking system and a solid efficiency criterion on which to examine the differences between a pure form finding algorithm with human estimation of the principal stresses and the other methodologies.

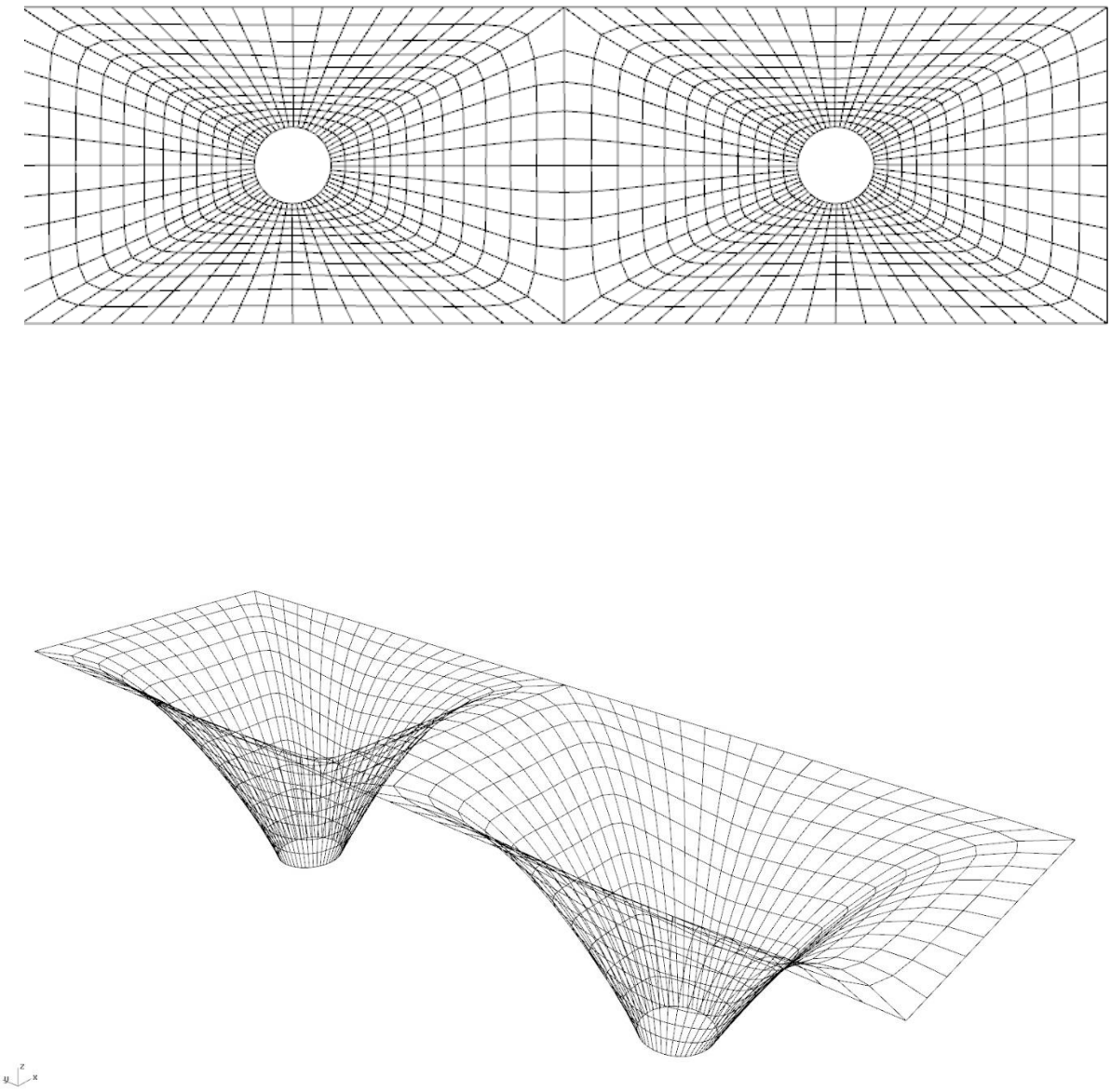


Figure 7-1: Base UV Subdivided Gridshell

The second quadrilateral panel section is shown in Figure 7-2. This is developed by taking the diagonals of the UV structure. This method results in the traditional “ribbon” wrapping which is often seen in gridshell columns such as the one at the Western Concourse. This “ribbon” version combines both primary and secondary systems, creating a more even distribution of force throughout the entire system, rather than creating a specific discrepancy between primary and secondary structure.

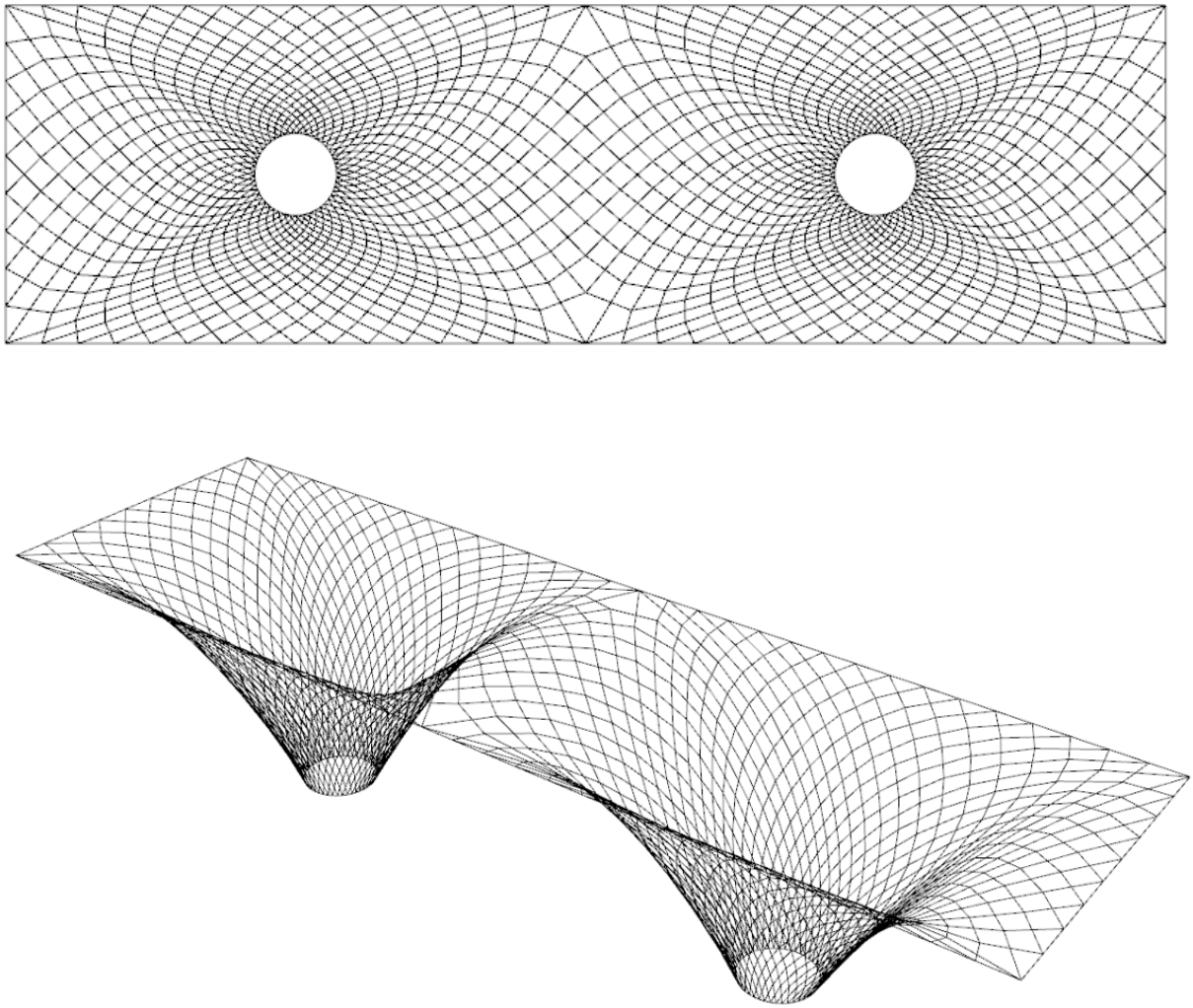


Figure 7-2: Base Diagonalized Structure

In a standard quadrilateral as the one in Figure 7-1, the mesh is defined as a set of radial and a set of ringed UV parameterization. In the diagonal method, this is no longer viable as both are just as radial. Therefore the U and V parameterization is done based on the direction the beams curve, with clockwise being U and counter clockwise being V.

The last set is a triangular system and therefore only uses flat panels, as a combination of the structures it should be able to leverage the strength of each. However, since it's such a significant combination of the two structures, it in turn uses vastly more structural steel, this can be seen in in the following base:

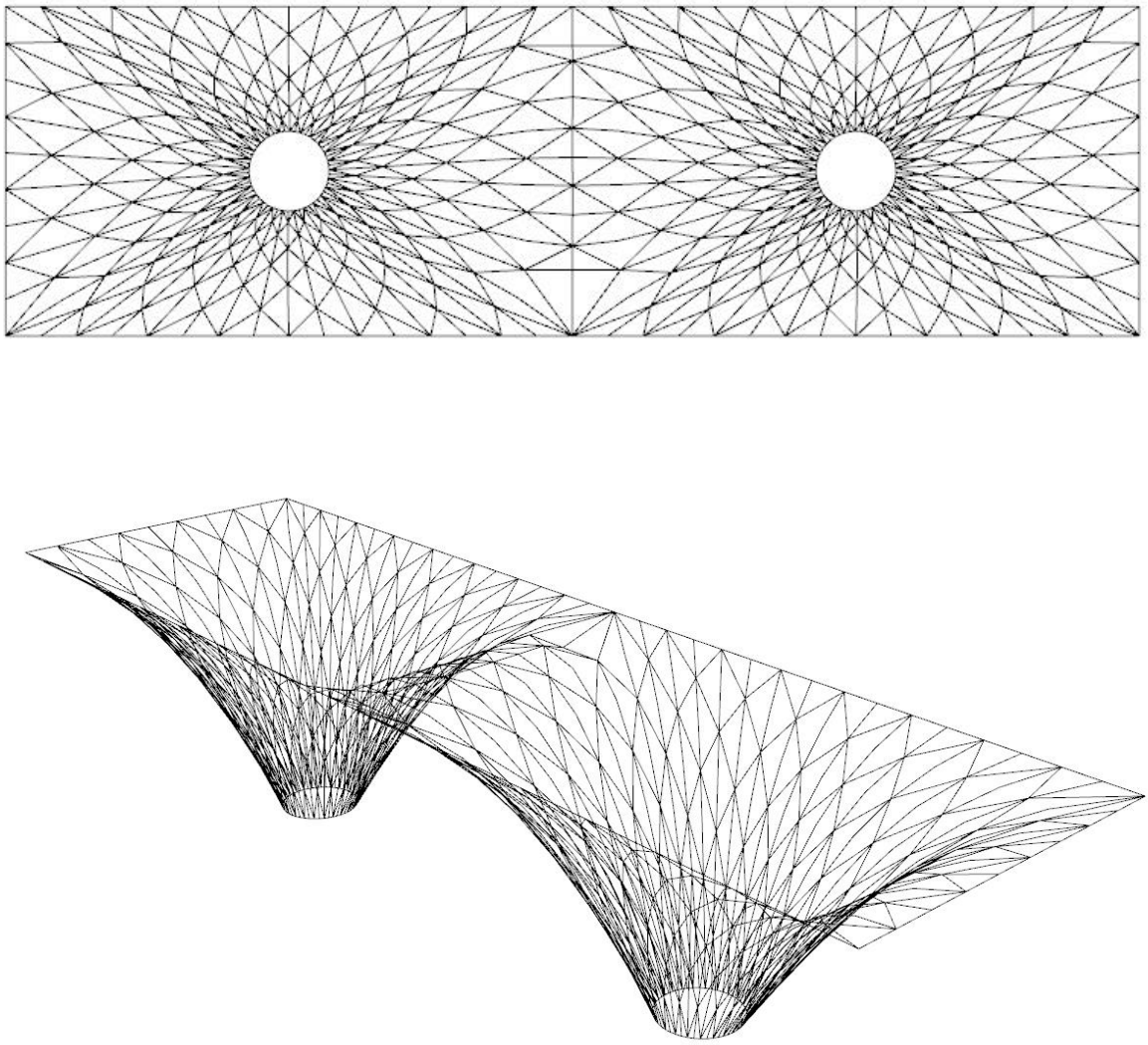


Figure 7-3: Base Diagonalized and Radial Structure

7.2 METHODOLOGY 2: STREAMLINE GENERATION WITH MESH MAPPING

7.2.1 Bar Generation

To create the optimized gridshell, a combination of Winslow's (2010) and Tam's (2014) methodologies were adapted to generate an appropriate gridshell. In the end, both systems generate a mapping function over a surface. While Winslow's paper utilizes periodic global parameterization to develop a globally conformal map with minimal area distortion, Tam's paper (2014) utilizes the shell surface directly, trying to implement stream lines across an FEA surface in 3D and removing set streamlines based on their strain energy.

Spacing stream lines in and of itself is not a simple task, and doing so in 3D becomes impossible as distance measurements between the streamline sets become more convoluted and complex. By maintaining a 2D mapping representation, effective and efficient seeding strategies can be used to optimize streamline placement.

7.2.2 Mesh Flattening

In the case of this system, since polylines with very small iteration points are used, the primary goal will be focusing on generating a primarily quadrilateral grid all existing in a single layer. One way of ensuring that the entire grid is developed on the same layer is to map the layout in 2D and the remap that back to the 3D surface.

The first iteration of the design meshed the gridshell into small set triangles, with each center point check being given a principal stress direction by the FEA analysis. This iteration then mapped these flat along the X-Y plane. This was done as an initial mapping system as Winslow's paper mentions that this systems is somewhat easy to implement and derives a relatively low area distortion for flatter shells. This is because the mapping to the XY Plane of most gridshell surfaces creates a semi-isometric semi-conformal map.

This flattening and mapping can been seen here in Figure 7-4:

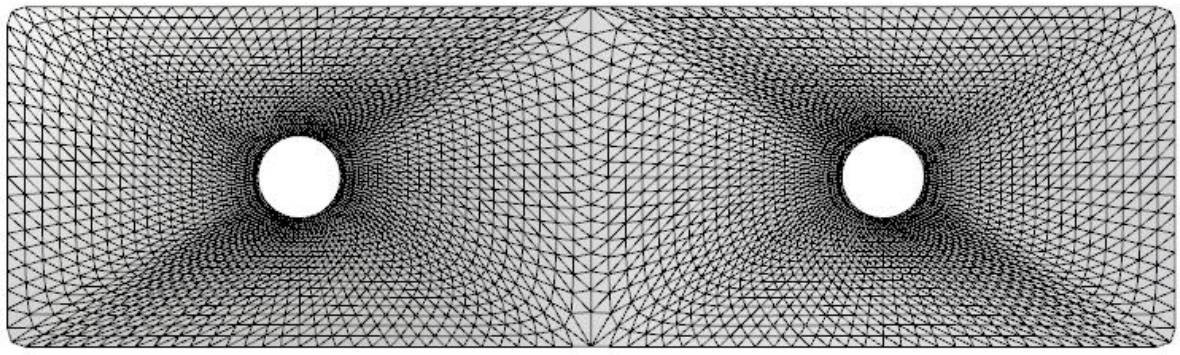


Figure 7-4: Planar Projection Mesh

While this flattened mesh is viable for low curvature gridshells, it provides significant issues with gridshells that are more vertical. This can be seen on the Figure 7.4 as the faces closest to the support points, i.e., the most vertical faces, appear smaller than the more horizontal faces, when in fact they are the same area in 3D space. In order to reduce this distortion, a different embedding was examined, Tutte Embedding.

Mesh parameterization and embedding is the act of balancing length distortion of each mesh edge while pulling the mesh to a 2D surface. If all springs or mesh edges retain the same spring stiffness, the resulted embedding is known as a Tutte Embedding (Tutte, 1963). Tutte's Spring Embedding theorem for planar graphs can be developed as shown in the pseudocode in Section 2.4.2 and again below, through the use of a Kangaroo script. The script pulls all points to the XY plane just as the previous flattening did, but now also uses the changing lengths of the springs to redevelop the mesh faces to a conformal global parameterization. This, in turn, reduces the angle distortion that was noticeable in the pure flattening algorithm. However, this method only works on convex polygons with a genus of 0. This provides a reasonable improvement in the mapping between the Tutte meshing and the mesh flattening on highly distorted gridshells. Nevertheless, the area distortion is quite clear when looking at the size of the hole in this mesh in comparison to that of the original mesh shown in Table 7.1 and visible in the outputs based on Figures 7-4 and 7-5.

The Tutte meshing is found using an energy minimization of all points such that:

$$v_{i,j} = \frac{\sum v_n}{n}$$

Equation 7-1: Vertex locations

Where $v_{i,j}$ is the mesh vertex index location, v_n are the neighboring vertex positions, and n is the number of neighboring vertex positions (Equation 7.1). This means that each vertex position becomes the average location of its neighbors (Equation 7.2) with the boundaries set as shown in section 2.4.2.

$$v_n = v_{i+1,j}, v_{i,j+1}, v_{i-1,j}, v_{i,j-1}$$

Equation 7-2: Definition of neighboring vertex locations

While most planar mapping functions cannot support non-topological disks, by giving the option of either a standard projection or an embedding, most gridshell structures should be mappable within the Grasshopper3d environment. By examining the area distortions between a Tutte and planar projection, one can choose which system most accurately maps the original shell.

Pseudocode for Tutte Barycentric Embedding

Import MeshPoints, MeshEdges, Boundary, Threshold
Output Mesh

Project Boundary
Construct Circumcircle
Check Interior Angles
    if Angle > 180: **Pull** PtsBoundary to Circle
    else Do Nothing

Place Interior Points
Replace MeshEdge with spring

Set SpringL_0 to 0
Define PtPositions
While SumDelta > Threshold
  For I in Points
    NewPtLocation = sum (
    Delta = (NewPtLocation[i] – PtPostion[i])
    SumDelta = Sum (Delta)

This was implemented using the same system as listed in the literature review in section 2.4.2 through Kangaroo with the use of 0 length springs, which in turn pull each vertex point that is not set to the barycenter of the neighboring vertices.

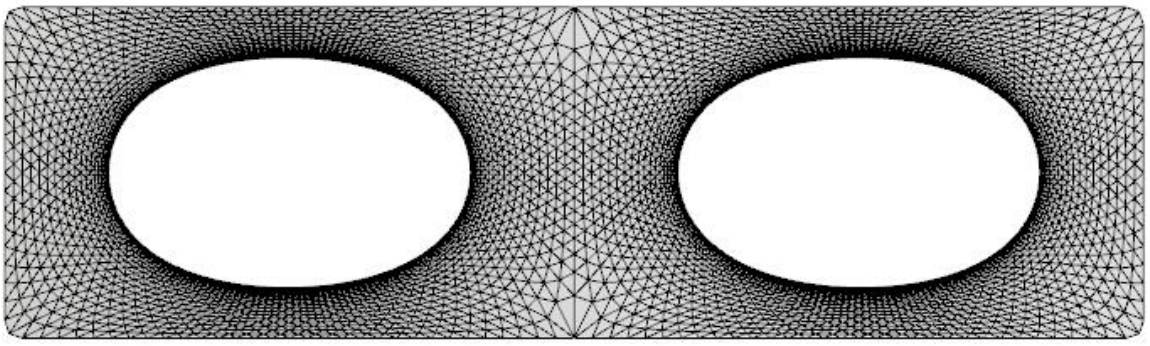


Figure 7-5: Tutte Barycentric Mapping

	MAXIMUM DIFFERENCE IN AREA	MINIMUM DIFFERENCE IN AREA	MEAN DIFFERENCE IN AREA	MEDIAN DIFFERENCE IN AREA
PROJECTION	0.197	1.65E-7	0.018	0.003
TUTTE EMBEDDING	0.205	3.5E-5	0.101	0.099

Table 7.1: Difference In Mesh Face Areas Between Original Mesh and Flattened Meshes

While Tutte Embedding is specifically useful for Genus 0, convex graphs, it cannot be guaranteed that every system will be so, just like this system is not. For this reason a planar projection will be used in this case. It can be seen here that the mean and median difference in area is much larger with the Tutte Embedding than with the planar projection. Therefore, the decision was made to continue with using the Planar Projection.

7.2.3 CUSTOM TRACER FOR DISCRETE VECTOR FIELD

Once the mesh flattening occurred, a custom Python script was developed (Appendix 2). This script utilizes streamlines generated via the Euler method with a step size of 0.2 meters and a weighted average vector movement by weighting the unit vector of the nearest point with their previous vector. This weighting allows for reduced movement throughout the field and allows for area normalization.

The streamline generator uses a Gaussian filter function (g(x)) across the entire vector domain in order to smooth the vector field towards mean values and remove noise in the data (Shapiro & Stockman, 2000). This is used to create the weighted values as shown in Equations 7-5.

$$g(x) = ce^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Equation 7-3: Gaussian distribution for Individual Value (Shapiro & Stockman, 2000)

This weighted average by distance allows for distinct discrete streamlines to be generated over the dense field using a direct Euler integration rather than a more complex tracer while still retaining high levels of accuracy. The equation for which is:

$$d_{i,j} = |\boldsymbol{n_i} - \boldsymbol{n_j}|$$

Equation 7-4: Distance Formula (Own)

$Where$:
$d_{i,j} = distance\ from\ node\ i\ to\ node\ j$
$\boldsymbol{n_i} = node\ location\ vector\ of\ current\ node$
$\boldsymbol{n_j} = node\ location\ vector\ of\ node\ to\ calculate\ distance\ to$

By finding the Gaussian distribution of each vector based on the distance from the current position to each vector, a smoothed field can be generated where:

$$w_j = e^{\frac{-(d_{i,j}-\mu)^2}{2\sigma^2}}$$

$$w_{sum} = \sum_{j=1}^{n} e^{\frac{-(d_{i,j}-\mu)^2}{2\sigma^2}}$$

$$\boldsymbol{\sigma_{1,i}} = \sum_{j=1}^{n} \boldsymbol{\sigma_{1,j}} * \frac{w_j}{w_{sum}}$$

Equation 7-5: Creating the Gaussian Weighted Stress Direction at Point I (Modified from (Shapiro & Stockman, 2000)

In this equation, w is the weight applied to the stress vectors from the FEA analysis, with w_j being an individual weight from the weighting function of vector at position j on $\sigma_{1,j}$ and w_{sum} being the total weights of all positions on the stress vector $\sigma_{1,i}$.

The section of the code relating to generating this streamline weighting is shown here:

```
#Gaussian function
def gauss_fun(values,sig=1,mu=0):
    return [ math.exp(-(x-mu)**2/(2*sig**2)) for x in values]

#Compute stress at location
def stress(loc):
    dist = rs.Distance(loc,field_loc)
    weights = gauss_fun(dist,sigma)
    weights_sum = sum(weights)
   return functools.reduce(lambda x,y: rs.PointAdd(x,y), [ w*s/weights_sum for s,w in zip(field_dir,weights)])
```

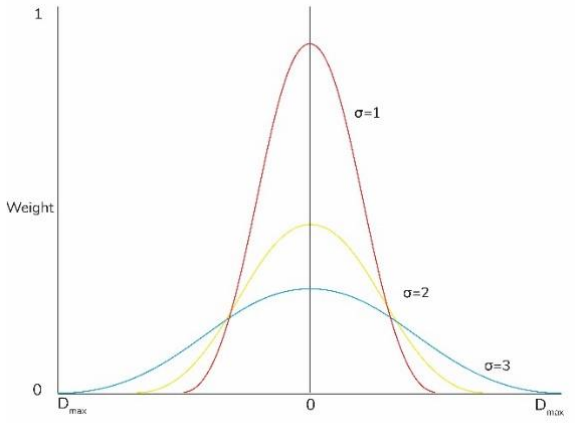


Figure 7-6: Lower Value Sigma's Increase the Weighting of Closer Streamlines (Own)

Unlike Kam's (2014) streamline tracer, this tracer works specifically in 2D to ensure proper mesh mapping and proper weighting of the Gaussian function. Kam's paper uses what is known as an n+1 vector averaging set, where all vectors within the streamline step radius are averaged in order to develop the step direction. While this method is viable with either a very dense vector field or a

large step size, it does not work well with non-smooth systems. The method presented in this paper uses the summation of Gaussian weighted vectors on a 2D surface to smoothen and process the vector field. This means that a weighted average of all vectors in the field is based on the distance they are from the current step point as shown in Equation 7-5. Here, the sigma value was set at 0.1 for this streamline tracer to ensure that the nearest subset of vector directions have a greater effect on the direction of the streamline tracer as shown in Figure 7-6.

7.2.3.1 Boundary Detection

The streamline tracer itself ends based on two different conditions: leaving the boundary domain or looping. If the streamline leaves the domain, the streamline will end and select the boundary curve intersection point as the final point in the streamline. This is a vital end condition that allows for the streamline to terminate properly and is standard for on surface streamline tracers. This must be checked in both directions and ensures proper termination of the main boundary condition.

This method is developed by ensuring that the streamline point is "on mesh" or has a distance to the mesh of 0 before each step function. By using this system the streamline will terminate the first step function after leaving the mesh. While this does not give the exact position of the last vertex, this is determined afterwards when the boundary curve is reintroduced during cleanup, and the end points are defined as the intersection between the boundary curves and the streamlines.

7.2.3.2 Ensuring Proper Step Direction

Since the principal stress vectors are 2nd order tensors, their direction can be defined either in the positive or negative direction. This means that at times the principal stress vectors do not flow smoothly but, instead, suddenly switch directions. This can be seen in Figure 7-7 where the vectors suddenly rotate 180 degrees. In order for the streamline not to oscillate at these points, a directional check was developed. Figure 7-8 shows the logic that built the system. Using the previous step direction, the following principal stress direction is checked. If the dot product of the two vectors is less than 0, then the angle between the vectors is greater than 90 degrees and the

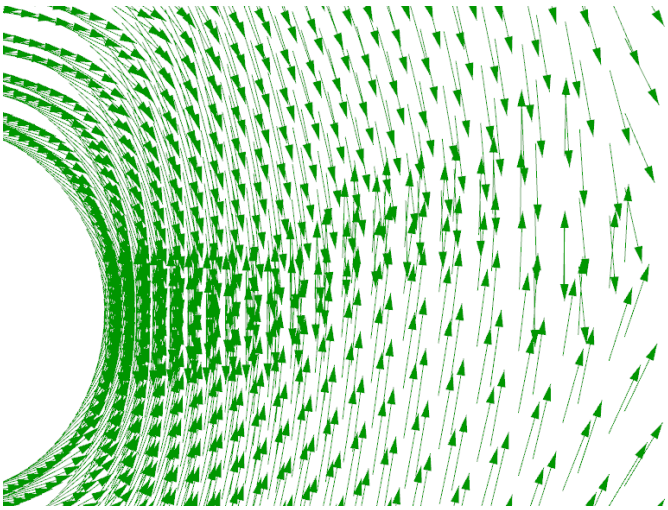


Figure 7-7: Converging Vector Direction in Principal Stress 1 from GSA Output

step would cause inconsistencies and oscillations in the stream line. Therefore, if the dot product is less than 0, the following principal stress direction is rotated 180 degrees, in turn creating a new angle with a dot product greater than 0. At this point the streamline steps to the following iteration.

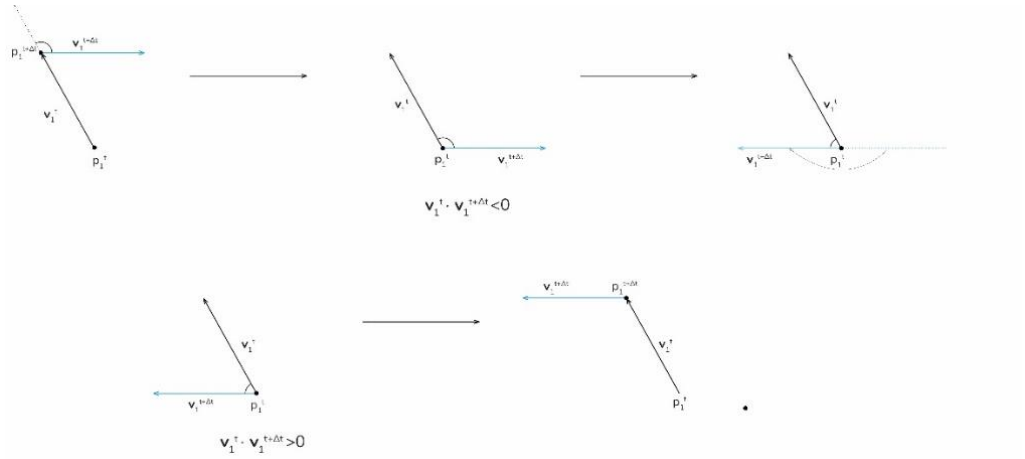


Figure *7-8: Process for Determining Proper Step Direction to Avoid Oscillation*

7.2.3.3 Looping

The second requirement for streamline tracing was defining a looped line. This was completed in the end by having each point during the iteration check for the other line at a distance of 99% of one step size. This method was integrated into the script via the use of a call distance function between the start and current step of the process. This part was vital to the function of the script as ring stresses often occur in gridshells. When trying to ensure that the streamlines do maintain a proper hoop, each step searches for another streamline step point within a small radius. If the two points are close together, the streamline will complete the ring and stop the integrator. By having the streamline move in both directions simultaneously, it also means that the expected error that occurs during the streamline tracing is halved. This makes the looping search much more effective than attempting to search for the start point after a full loop. Figure 7-9 shows the differences that these methods make in developing discrete loops with Karamba3D: the figure on the left shows continual steps and no termination

through their streamline tracer, whilst the figure on the right shows my script detailing several fully terminated streamlines.

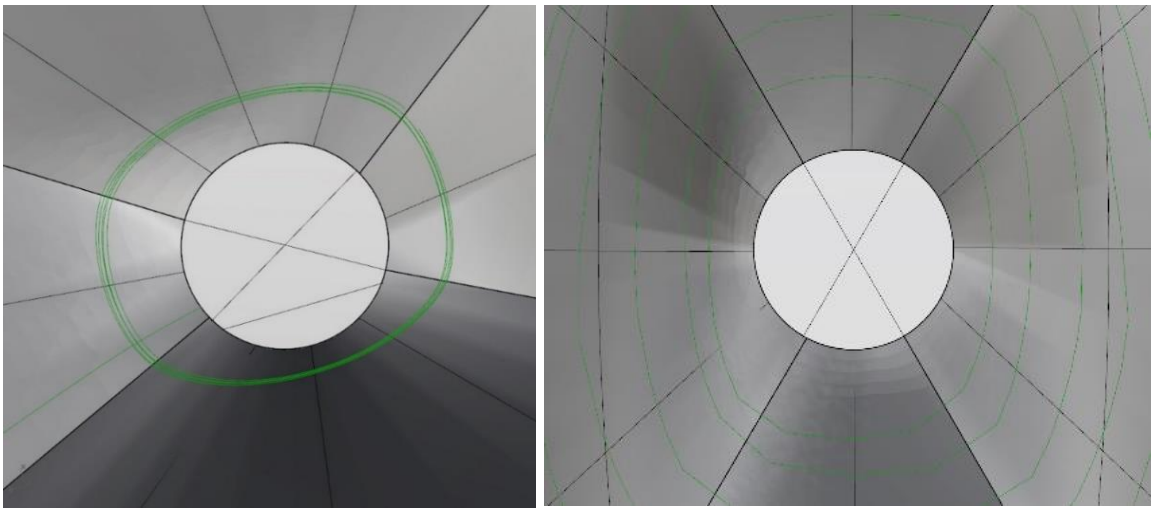


Figure 7-9: Comparison of Karamba3D (Left) to Looping of Own Script Right (Own Images)

7.2.3.4 Singularities

Finally, there is the condition of the streamlines when they approach singularities. In these positions streamlines will start to misbehave. The streamline will either circle back around on and in turn cross over itself, or the streamline will begin to switch back after circling the singularity. This typically leaves two more boundary conditions that are required to be satisfied. Either way it is in these points that the streamline will no longer continue to the boundary or loop fully around on itself.

One other advantage of using a Gaussian function is that the system smooths the vector field enough to ensure that the streamline is able to move through the singularity without too much distortion. However, the singularity does implement some issues as seen in Figure 7-10. Specifically, there is a noticeably higher density of streamlines in and around the singularity, resulting in a highly dense structure.

These singularity points have also been a major point of trouble within the discrete differential geometry field, with many times the solution being to either leave them out or apply a “quick fix” to smoothing them out of the mapping (see Figure 7-16). Nicholas Ray for instance uses a vector field smoothing and a curl correction

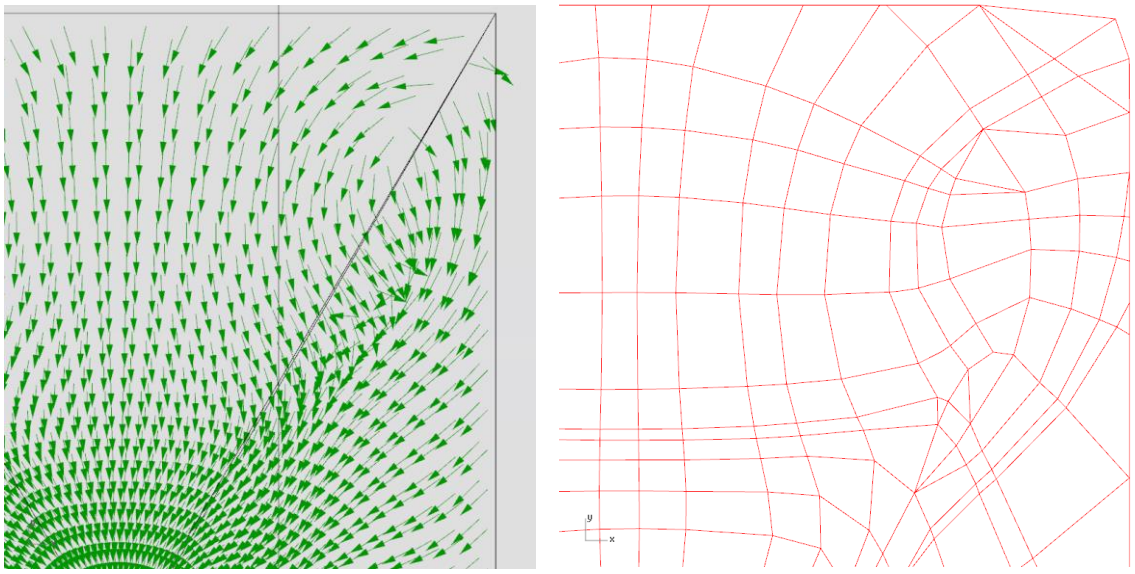


Figure 7-10: Singularity in P1 Stresses (Own Image)

algorithm in order to minimize the number of singularities in the data at the expense of accuracy (Ray et al., 2006). Winslow (2010) takes this a step further by integrating an area function into the energy minimization, such that each U and V isocurve maintain similar spacing throughout the entire parameterization, essentially removing them entirely. For the main issues experienced here, a pruning system was set up. This system ranks the importance of loops ahead of single line streamlines, thereby cleaning up any formed loops in the system. Regardless of the location of the loop in the streamline, any intersection in the streamline with itself will signal to the program to stop and cull the ends off of the loop.

The pseudocode below shows how these complications are dealt with by using the pruning functions within the script to ensure for closed looped streamlines even when the singularity begins to distort the streamline tracer.

If the streamline were to start exactly on the singularity, the streamline would generate an oscillation, leading to the streamline culling itself to a length of 0. Since the seeding strategy is based on the farthest point, this would result in the seeding strategy reselecting this point. In order to solve this issue a command was inserted for the script to generate a circle with a radius of 0.1m. This was implemented to ensure that the point would not be reused in the seeding selection algorithm while being small enough that it would eliminate itself during cleanup. This would occur as the polyline around this point would only have 1 intersection point and thus not create a line.

```
INPUTS(Field_Location,Field_Direction,StartPt,StepSize,Sigma,Mesh)
OUTPUTS(Streamline)
Coerce Inputs
Define Gaussian_Function(values,sigma,mu)
   return (e^(-(x-μ)^2/(2σ^2))) for x in values

Define stress(location)
   dist = distance(location to each field location)
   weights = Gaussian_Function(distance, sigma)
   weights_sum = sum(weights)
   stressvec = functools.reduce(lambda x,y: rs.PointAdd(x,y),[w*s/weights_sum
for s,w in zip(field_direction, weights)
   return stressvec

Define step(loc,dir,stress)
   stress = stress.normalized()
   dir = dir.normalized()
   if dotproduct(stress,dir)<0:
                stress = reverse(stress)
   delta = stress*stepsize
   return = loc+delta, stress

Define check_boundary(mesh,loc)
   return IsPointOnMesh(mesh,loc)

Define check_looping(loc, pointlist)
   dist = distance(loc,pointlist)
   min_dist = min(dist)
   return point_list[(dist.index(min_dist)] if min_dist<0.99*step_size else None

Define prune_head (point, line)
   try:
      return line[0: line.index(point)+1]
   except:
      return []

Define prune_back (point, line)
   try:
      return line[line.index(point)-1::]
   except:
```

```
    return []

Define iterr (loc, dir):
  line1, line2 = [loc],[loc]
  loc1, loc2 = loc, loc
  dir1, dir2 = dir, dir.reverse()
  run1, run2 = true, true
  for I in range (iterations):
      run1 = run1 and check_boundary(loc1)
      if run1:
          loc1, dir1 = step(loc1,dir1,stress(loc1)
          point = check_looping(loc1, line1+line2)
          if point is not None:
            line2 = prune_head(point, line2)
            if check_looping(loc, line1) is not None:
              line1 = prune_back(point,line1)
            line1.append(point)
            break
          else
            line1.append(loc1)
      run2 = run2 and check_boundary(loc2)
      if run2:
          repeat run1 (loc1 = loc2, dir 1 = dir2, line1 = line2)
      if not run1 and not run2
          break
return list(reversed(line2[1::]))+line1

line=iterr(start,stress(start)/vectorlength(stress(start)))
try:
      line = addpolyline(line)
except:  #If Line length is 0 or None#
      line = addcircle(start, 0.1)
```

7.2.4 **EVEN SEEDING OF STREAMLINES**

As mentioned earlier, the bar structures need to be spaced evenly across the surface with a distance of roughly 1 meter.

While Kam's system is also based on developing streamlines, his system utilizes alternating between the σ_1 and σ_2 sets of Vectors (Kam, 2014). His algorithm uses the midpoint of the previous streamline and creates a new polyline in both directions until such polyline reaches the boundaries. Once this is completed, the polyline finds all intersection points and creates straight bar structures between them. While this script is simple and useful, it does run into some issues. The script then does large amounts of cleanup using a variety requirements such as clumping of grouped streamlines and defining each set by an average strain energy. According to the author the streamline cleanup took up to 3 months before it was presentable. This system was only tested on standard shell shapes: primarily a 3 point, 4 point and 5 point supported shell. These specifically do not contain principal stress singularities and thus fundamentally avoids one of the major problems (Kam 2014).

It was therefore determined in the end that this method of combining parameterized maps and streamline generation had the highest chance to be viable. The only way to solve the major issues that come with streamline generation for gridshell structures would be to be able to maintain singular streamlines that are generated on a 2D surface where each streamline is purposefully placed.

Once a streamline has been developed, the script creates a Delaunay mesh across the boundary of the flattened mesh and streamline. Based on the Delaunay mesh, the largest span distance is determined by circumscribing each interior triangle. From this, each circle's diameter was determined and the center point of the circle was used as the new point to start the Euler Tracer for Streamline. This can be seen here in Figure 7-11. This method is a modification on Abdelkrim Mebarki's (2006) farthest point seeding strategy in which their system utilizes a kicking of the mesh faces affected, followed by a re-computation of the mesh in the area. My method redevelops the mesh anew each time and creates a new triangulation scheme.

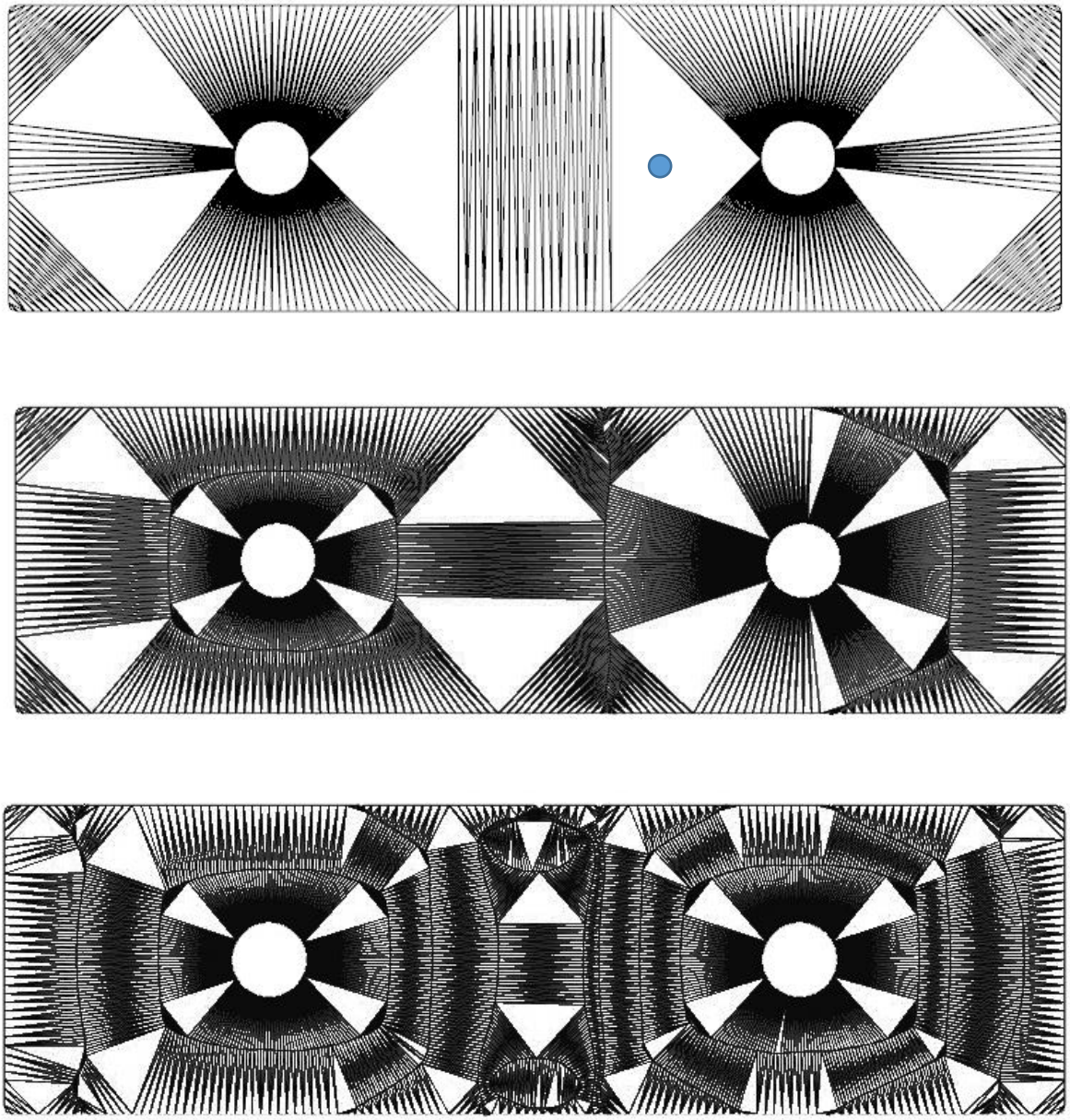


Figure 7-11: 0th, 7th, and 20th Iteration Delaunay Meshes

However, the base system does not work fully with the holes that exist in the mesh. Therefore, a constraint had to be placed on the Delaunay triangulation. The reason it does not work is that many meshes were being generated within the mesh holes or outside of the mesh. Since the scheme is a 2D scheme, the center of each circumcircle must exist on base mesh that the streamlines are being mapped onto itself in order for the streamline to be developed.

This allows for all circumcircles with a center point that exist outside of the mesh area to be culled and removed from the possible analysis start points.

The streamline tracer is run until the desired density is reached as shown in Figure 7-12.

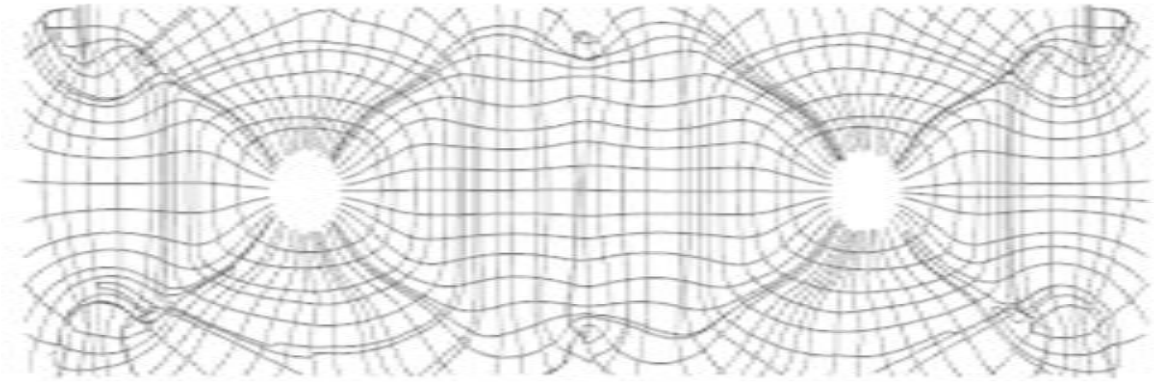


Figure 7-12: Raw Streamline Output

As seen, this system is far too messy to generate a proper gridshell, despite that the system is better than the Karamba streamline functionality. In order to use this system as a developable structure, more cleanup is needed.

7.2.5 **Cleanup of streamlines:**

The cleanup of the system is developed through ideas brought through in Kam (2014) and a personal parametrization method. The main issues here are the convergence of streamlines at singularities and the set looping for hoop stresses.

The standard streamline cleanup was completed after the streamlines have been generated. There are 3 data trees, one for each set of lines: the U parameters, V parameter, and boundaries. Each polyline has its own data in each list within the tree. Any change to a point will only occur to the individual poly line that needs to be modified. From here, the network of intersections between the U and V parameters create the new points for the polylines. This approach creates straight lines between the intersection points which become the straight bar structure for the gridshell.

From here, the system is developed by reintroducing the boundaries and defining intersection points between U or V lines and the boundaries. Now that the discretized boundaries are developed, the full structure is developed with the straightened bar version is shown here in Figure 7-13.

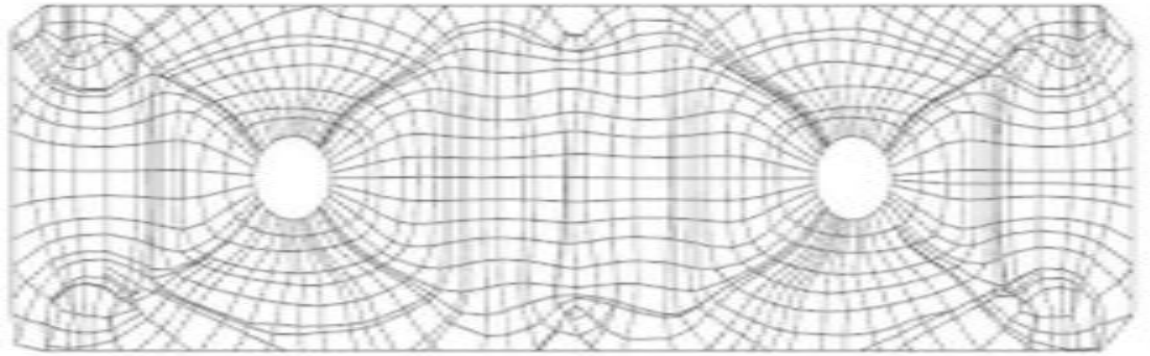


Figure 7-13: Cleaned Up and Discretized Streamlined Structure

When this is completed, the polylines are simplified to the intersection points becoming the control points. This generates closed quadrilateral-panels, however it does not control for planarity or convergence.

7.2.6 **Mapping Back**

Mesh stream lines in 3D have the problem of diverging from the shell shape. This typically occurs as the streamline steps over a mesh boundary utilizing data from the previous step and therefore diverge from the surface. In order to remove this possibility, meshes can be flattened and portrayed in 2 dimensions and, as long as a 1:1 mapping exists back to the original mesh, the optimized streamline can be brought back to the original shell by sending an ordered structure of points through the mapping and by rebuilding the polylines on the other end.

It is at this point that parts of the shell can be seen to have significant area distortion, especially in the spacing of the rings around the more vertical sections of the shell. In order to ensure that the solution is viable, the parametric process is broken and the gridshell is finalized by hand.

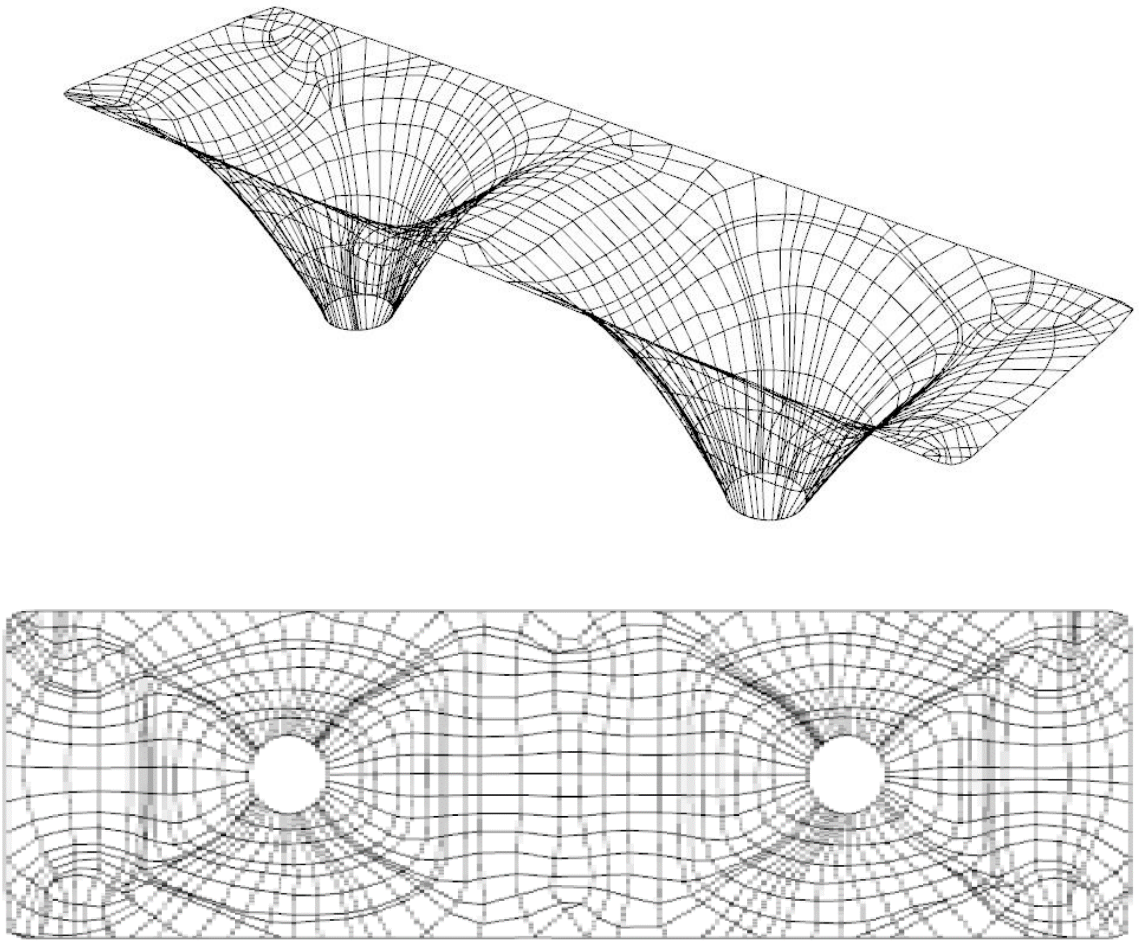


Figure 7-14: Final Streamline Structure when Finished By Hand

During the last weeks of work, an algorithm was developed where points that existed within 0.2m of each other were grouped and given a mean nodal location. This resulted in triangulating the converging streamlines to single points at these locations resulting in the result shown in Figure 7-15. This was done to ensure that bars would not run parallel to one another at a distance that would result in the steel bars overlapping. This in turn provides the beginning basis for a method of further rationalizing the gridshell under constructability requirements and removes much of the hand cleanup that was required in the initial gridshell streamline development.

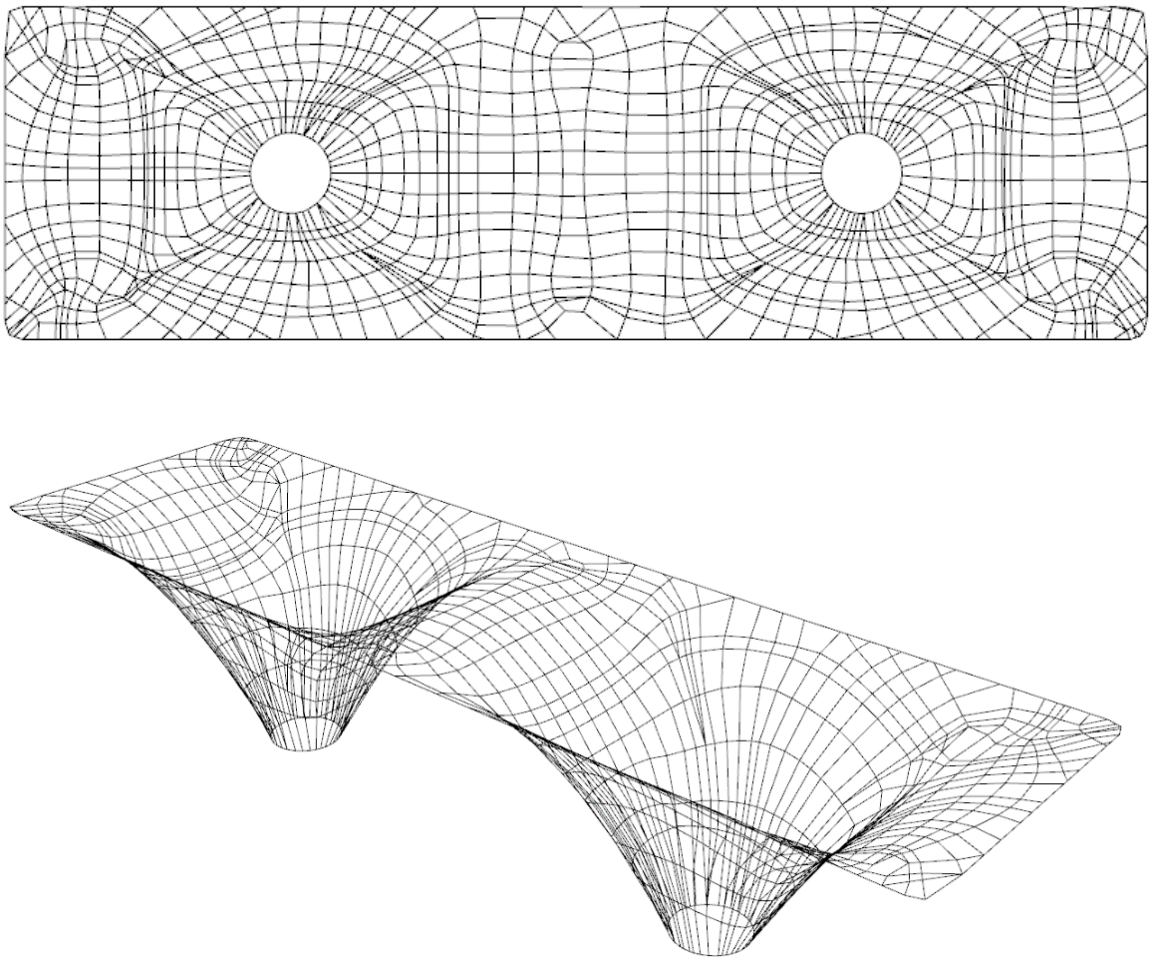


Figure 7-15 Nodal Averaging of Converging Bar Structures without Hand Work

7.3 METHOD 3: GRIDSHELL GENERATION THROUGH PGP

In order to develop another comparison, a new method of grid generation was analyzed for sake of structural efficiency. The mapping must be pseudo-isometric and conformal as any other mapping will distort the gridshell spacing too much or distort the angles of the gridshell too far from 90 degrees. One useful mapping that fits this criteria is that from Nicholas Ray called Periodic Global Parameterization (Ray et al., 2006). This is a complex algorithm that uses a set of conjugate vectors to reparametrize a pseudo-conformal and pseudo-isometric mapping of a grid along two discrete orthogonal vector sets.

This mapping method is the same as what Winslow used to develop his gridshell in his PhD thesis (2010). This method works as outlined earlier in the thesis and by minimizing the energy differential between the vector direction and the UV map, it is an incredibly powerful method of generating very efficient meshes along a cross-vector field.

The UV parameterization requires complex sub-maps linked via transition functions. The computer system to generate this analysis would require significant modifications, the CGAL system is developed in C++ rather than .NET interface, hence development of this modification was not practical within the scope of this project.

Efforts were made to try to develop this system in Grasshopper3D. However, since the UV parameterization requires complex sub-maps linked via transition functions, it was determined to be out of the scope of the thesis, especially when further examined in other papers. When examining Winslow's paper and N. Ray's base parameterization it was noted that access to CGAL was required in order to generate a working example. Unfortunately, the current CGAL system is written explicitly for C++ which means there is no .NET interface making it infeasible to develop given the time constraints. Therefore another option was examined. The parameterization also needs principal stress vectors at each node point, rather than at each mesh face which is also not an output GSA or Karamba have available.

The main algorithm used in Grasshopper3D was developed by Panagiotis and Sawako and is defined under the Millipede component (Panagiotis & Sawako, 2014). This system allows for a fundamental base of Nicholas Ray's algorithm in the Grasshopper3D environment. This method requires an input of the desired orthogonal vector parameter fields and a desired mesh density. From this data a new UV parameterization is developed that minimizes the difference between the direction in angle between the UV parameterization and the vectors.

When the chosen design was applied to the principal stress vector field, the following shape was developed as shown in Figure 7-16.

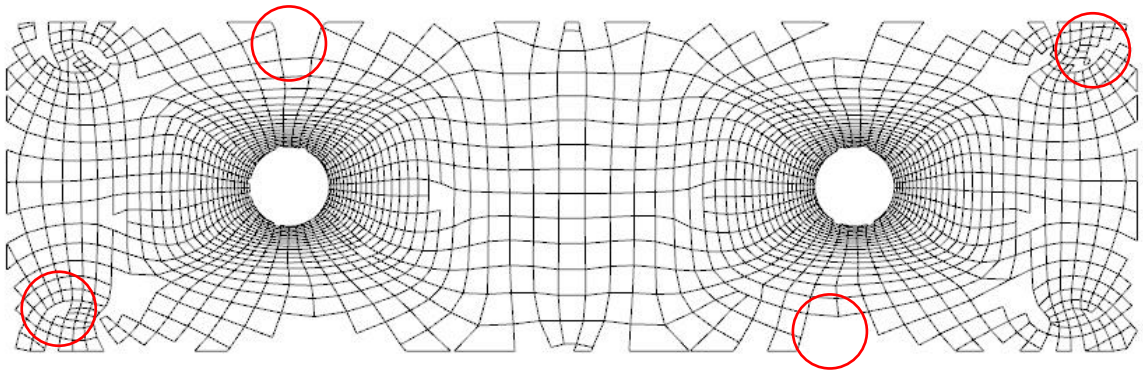


Figure 7-16: Initial Output of the Parametrized Grid Including Gaps and Unstable Joints

As shown in Figure 7-16, this method of parameterization also leads to significant discontinuities in the developed grid. Due to the singularities in the corners and problems with the parameterization at edges of a surface, the parameterization cannot fully complete. This is because singularities cause null points in the mesh, causing mesh edges to be zero length, and thus collapse. Instead, in this case parts of the grid are cut away in order to ensure a smooth global mesh while leaving local discrepancies (Figure 7-17). The system also struggles to parameterize near edges, and therefore, has to have the boundaries reinserted and cleaned up by some parametric processing and manual hand cleanup. This is visible in Figure 7-17. Several points along the mesh edge, two curves will nearly intersect, creating an unstable joint as visible in Figure 7-16. These joints were fixed using the algorithmic grid refinement.

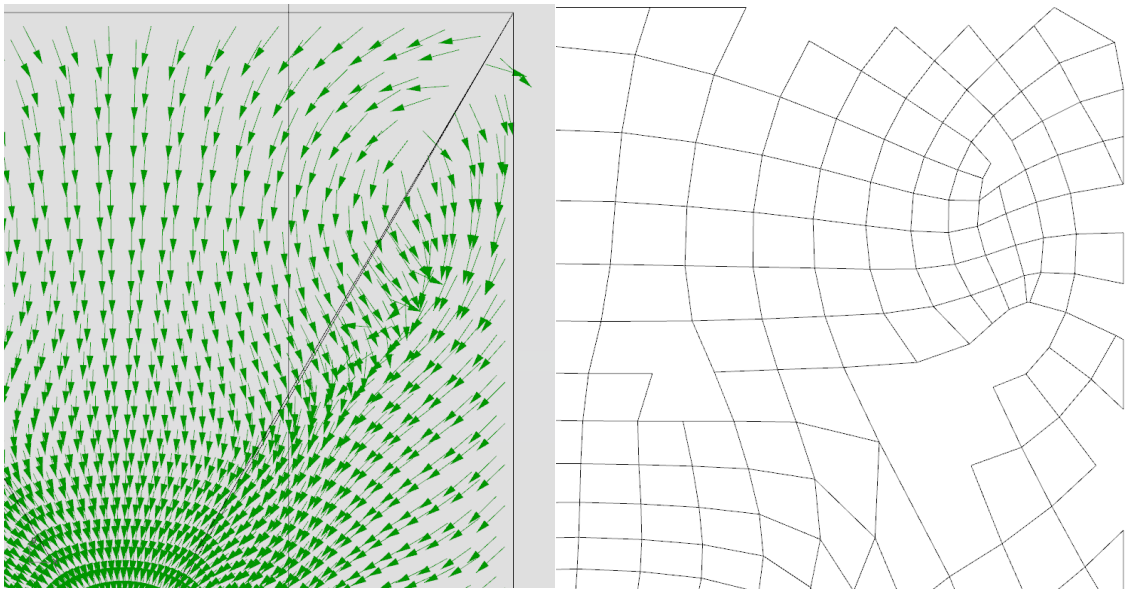


Figure 7-17: Results of Periodic Global Parameterization Near Singularities

7.3.1 **Refining the Grid**

The Grid in both methods defines essentially different UV subsets of lines. In order for the parameterization to be viable across the entire gridshell, the output had to be cleaned up and completed.

The first option is to develop the grid by hand this method leads to a quicker method of synthesizing the gridshell structure as lines are followed by hand. However, this does break the flow of data meaning that optimization based on structural or planarity criteria is impossible.

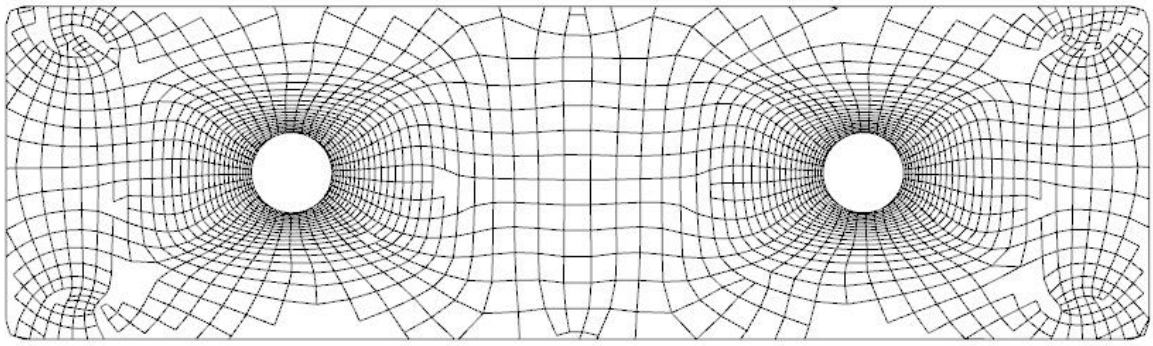


Figure 7-18: Script Cleanup of PGP Mesh

First, a gridshell cleanup was developed, where any points that existed within 0.05 m from each other were unified. This means that there are significantly less small bar structures than in the original gridshell, it also turns some quadrilaterals into triangles, but this is not an issue as triangles are far more stable and these triangles therefore give stability to the structure. It however, does not fix the gaps in the structure. Those were then required to be fixed by hand.

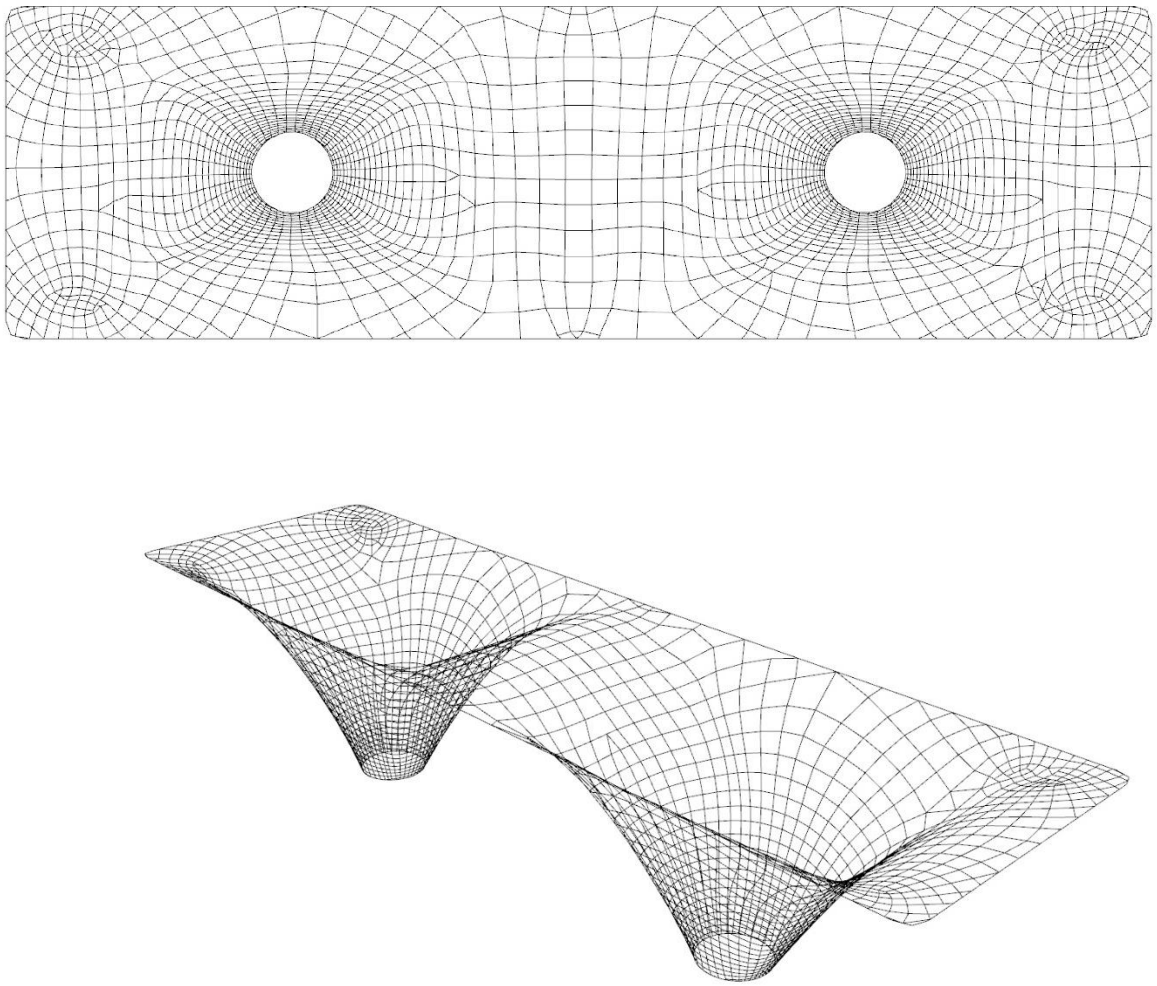


Figure 7-19: Final Cleaned-Up Grid from Parametrization When Finished By Hand

7.4 SHELL THEORY

Since the efficiency of the form of a gridshell comes directly from the same methods as a shell structure, it seemed logical that perhaps utilizing the method in which shell structures are analyzed could yield some results.

If a circular plate is loaded uniformly with the area $A = \pi r^2$, a cross section of the plate can be modeled and defined as in Figure 7-20. Taking a slice of the plate as θ means that we can develop an infinitesimally small cross section as $d\theta$. However, since $d\theta$ still contains a depth difference between the plate at the full radius and at the center. The load changes linearly over the full length as shown in the top diagram for Figure 7-21.

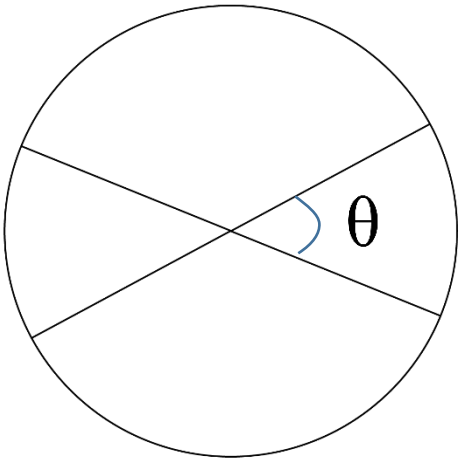


Figure 7-20: Subdivision of Circular Plate

This generates a loading, shear and moment condition depicted in Figure 7-22:

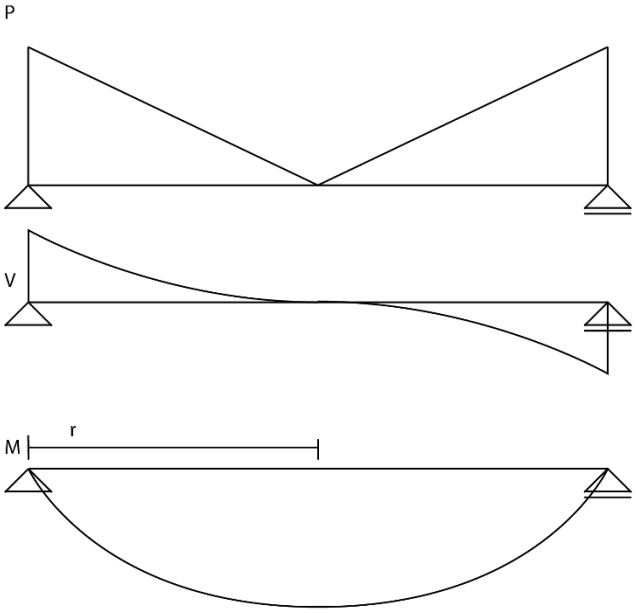


Figure 7-21: Loads and Beam Behavior of Shell Section ($d\theta$)

The moment in the beam can therefore be modeled as:

$$m = -\frac{1}{6r}x^3 + \frac{1}{6}r^2$$

Equation 7-6: Moment for a Section ($d\theta$) of a circular plate

Where:

r = circular radius

x = radial point on the shell

m = moment

For a circular plate this moment line can become a moment hill for the plate by rotating it around the position $\frac{r}{2}$, giving the full moment circle for the plate. Provided the plate is twist-less, this results in a moment hill that when, used as a shell, provides a shape containing in plane stresses of as much as $1.11E-6$ N/m^2 and maximum out of plane moments up to 443 N as shown in Figure 7-22:

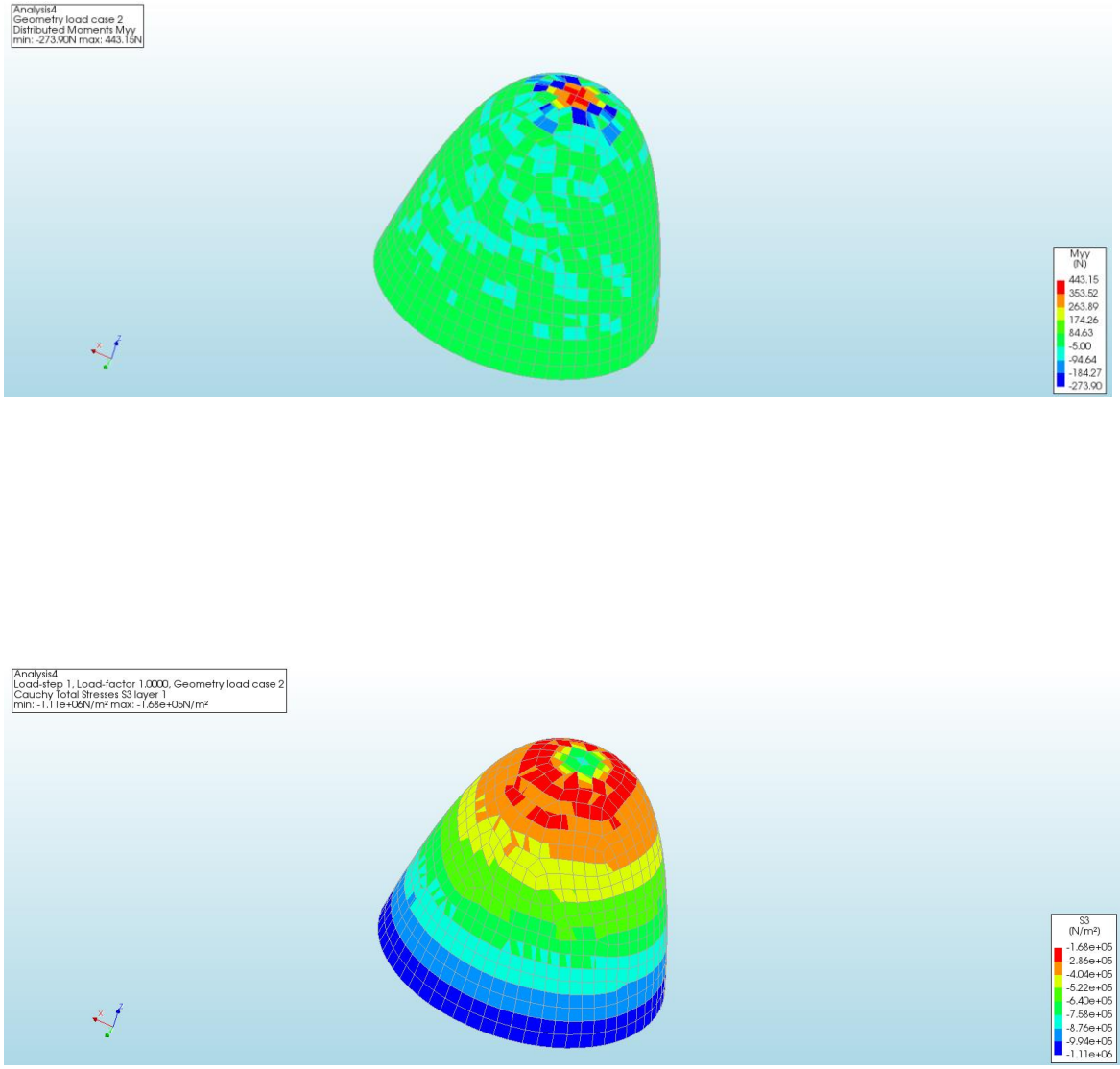


Figure 7-22: Moment Hill Out of Plane Moment (m_{yy}) vs Moment Hill In Plane Stress (S3)

Based on this idea, we can apply Calladine's (1983) twin shell theories to examine the relationships between the bending and stretching surface of this shell. Calladine (1983) proposed splitting a shell structure into two discrete surfaces, the B-surface or bending surface and the S-surface or stretching surface as shown in Figure 7-23. This allows for the divorcing of the in plane interactions from out of plane forces. This in turn means that the S-surface has no bending stiffness and cannot twist; instead it takes all normal forces and in plane shear $(n_{xx}, n_{yy}, n_{xy}, n_{xy})$. Whereas the B-surface only takes bending, out of plane shear and twisting into account $(m_{xx}, m_{yy}, m_{xy}, m_{yx}, v_x, v_y)$ when a force (P) is applied to the shell. In light of this, all loads and divide them between the shells. Despite divorcing these behaviors, the shell must maintain its wholeness and thus any deformations that occur in one shell must occur in the other. This oneness is set by forcing the Gaussian curvatures (c_g) of both shells to be equal (Equations 7-7) and by ensuring that the sum of forces acting on the two shells is equal to the total force (Equation 7-8). Therefore, to maintain this interaction the following requirements and coupling are set.

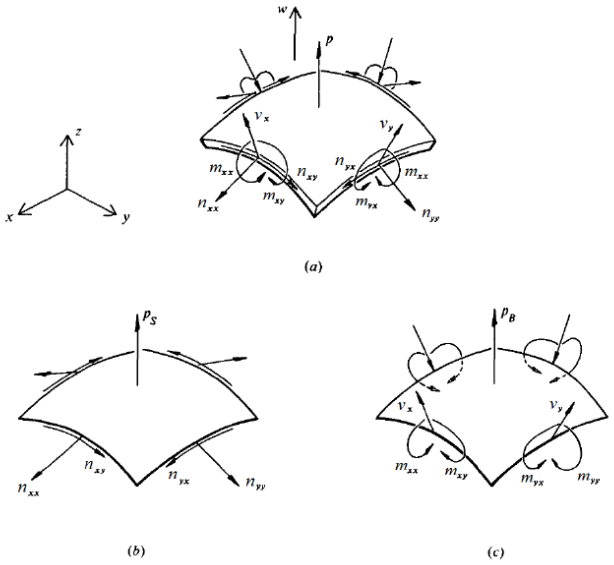


Figure 7-23: The Twin Shells (Blauwendraad & Hoefakker, 2014)

$$c_g^s = c_g^B$$

Equation 7-7: Gaussian Curvature Relationship Between Bending and Stretching Shells

$$\boldsymbol{P_{tot} = P_s + P_B}$$

Equation 7-8: Loading Relationship between Bending and Stretching Shells

Where

$$\eta_{xx} c_1 + \eta_{yy} c_2 = \mathbf{p}_s$$

$$\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} = \frac{\partial m_{xx}}{\partial x^2} + 2\left(\frac{\partial^2 m_{xy}}{\partial x \partial y}\right) + \frac{\partial^2 m_{yy}}{\partial y^2} = -\mathbf{p}_B$$

Equation 7-9: Stretching and Bending Loading Values

This means that, since the curvature is one over the radius of curvature, the total load can be rewritten as the following:

$$\frac{\eta_{xx}}{R_1} + \frac{\eta_{yy}}{R_2} - \frac{\partial v_x}{\partial x} - \frac{\partial v_y}{\partial y} = \boldsymbol{p}_{tot}$$

Equation 7-10: Unified Loading

Now, knowing that the relationship exists around the Gaussian curvatures of the shells and the loads applied to each case, one can state that the following relationships exist:

$$m_{xx} = D\kappa_{xx}, m_{yy} = D\kappa_{yy}, -m_{xy} = -D\kappa_{xy}$$

Equation 7-11: Relationship between Moment and curvature in the moment shell

And:

$$\epsilon_y = \frac{1}{Et} n_{yy}, \epsilon_x = \frac{1}{Et} n_{xx}, \frac{1}{2}\gamma_{xy} = \frac{1}{Et} n_{xy}$$

Equation 7-12: Relationship between strain and in plane stresses

From these sets of equations we can determine that the following ratios exist:

$$n_{xx} \leftrightarrow \kappa_{yy}, n_{yy} \leftrightarrow \kappa_{xx}, n_{xy} \leftrightarrow -\kappa_{xy}$$

Equation 7-13: Relationship Between Curvature of the Moment Shell and the In Plane Forces (Oosterhuis, 2010)

Where:

$\kappa = curvature\ created\ by\ moment$

$n = normal\ force$

This means that the curvature of the moment shell should be proportional to the normal forces, therefore principal curvatures should be related to the principal normal forces.

Theory states that for a twistless case, the absolute moment values of a plate should be the idealized shell form when projected into 3D space. This idea was first introduced by Beranek (1972) and was later explored through analysis from Oosterhuis (2010). This can be rewritten as the following:

$$\overline{m} = -D\left(\frac{d^2w}{dx^2} + \frac{d^2w}{dy^2}\right)$$

$$\overline{m} = \frac{m_{xx} + m_{yy}}{1-\upsilon} = \frac{m_1 + m_2}{1-\upsilon}$$

Equation 7-14: Moment Hill Definition

Since principal shear force is the directional derivative of this moment hill, and shear is a first order tensor, this meant that the isolines of the moment hill of a flat plate are the minimum shear, with the maximum shear being the steepest gradient of the moment hill. From this idea a steepest ascent calculator was created by Oosterhuis (2010) using finite difference methods as shown in Figure 7-24. This gave rise to the idea that the principal normal force should also be plottable along a steepest gradient function

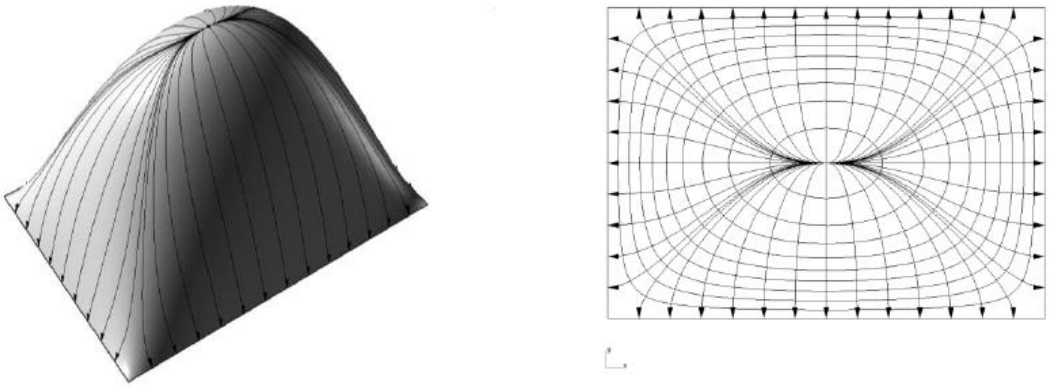


Figure 7-24: Moment Hill and Principal Shear Force Trajectories (Oosterhuis, 2010)

The first idea was to examine the moment hill itself, however, it became clear that the only derivation possible from the moment hill was that of utilizing the principal shear and generating sets of orthogonal vectors at 45 degrees from the steepest descent. While this information is useful, it unfortunately remains a continued issue of integrating with a discrete vector step.

Therefore, the main issue of deriving principal normal force without relying on Mohr's circle proves challenging and continuously a problem. Since both moment and normal stress remain 2nd order tensors affined by 1st order tensors as shown in Equation 7-15.

$$\varphi_y \rightarrow \begin{matrix} m_{yy} \\ n_{yy} \end{matrix} \rightarrow v_y$$

Equation 7-15: Relationship between Tensors Rotation(ϕ_y), Moment (m_{yy}), Normal Force (n_{yy}), and Shear (v_y)

If perhaps the principal moment trajectories can be mapped, a relationship between the moment in a plate and the normal forces in a shell can created.

The next idea came from Beranek's paper on Moire lines and flat plates (Beranek, 1972). If the principal moment hill can lead to an idealized shell, which means that the moment trajectories in a twistless flat plate translate to the membrane forces

in a shell structure. This would mean that the principal moment trajectories in a flat plate are in turn the principal axial stresses in the shell. However, principal moment trajectories are typically defined as a conjugate vector set such as principal stresses.

We know that the first order derivative of the rotation gives the same form as the second order derivative of the moment; the question becomes which partial derivatives are needed.

This is shown constitutively in the twin shell theory where the equilibrium equations rewrite the principal stresses as

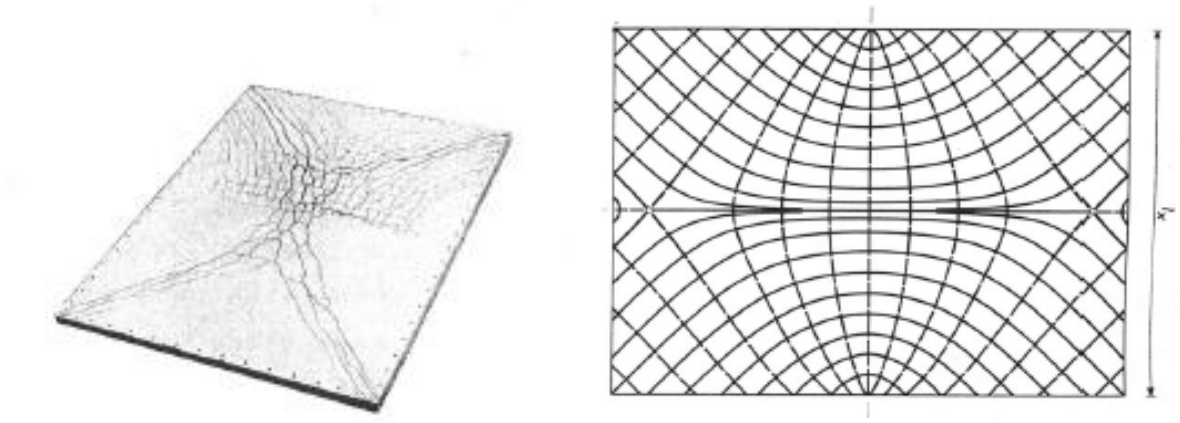


Figure 7-25: Failure of a Plate Due to Principal Moments and Mapping of Principal Moment Curves (Beranek, 1972)

For this reason flat plates were examined in order to see if a proper mapping could be developed. In 1972, Prof Beranek created a paper based on the work of Ligtenberg's examination of moiré patterns of flat plates. For his experiments he uses a 1.4:1 aspect ratio plate. In these experiments curved mirrors and screens containing line filters were used to examine what relationships could be determined from plate curvature.

In the meantime other options were examined. This system itself was the principal slope of the angle of the plate. Beranek in his lecture notes (1972) mentions using a second order Moire photo set (2 Moire line patterns over each other) to generate a principal moment diagram.

This work led to the development of contour plots of moment on the surface of the plate, where Beranek was able to model failure locations of a flat plate based on the lines generated from the trajectory projections of principal moment. From this the idea arose that the possibility may exist to map these principal moments to a shell.

Since the moment hill of a flat plate is the optimal shell configuration for said floorplan, we know that the principal moment trajectories and isocurves become principal stress curves and isocurves. However, the goal is to map this requirement to that of a geometric property of the plate itself such that the use of vector directions are no longer needed. In order to explore relationships between shell normal forces and plate bending, a script written by Oosterhuis (2010) based on Bernarek's (1976) concept of rain flow analogy was used. This system uses steepest descent in order to determine the principal gradient of the surface. Isolines are then drawn by taking planar cuts of the surface every 1/20th the depth of the surface. By using this method we could examine the principal gradient of a variety of surfaces.

From Beranek (1972) for a plate the principal moments in a twistless case would be:

$$m_1 = \frac{1}{2}(m_{xx} + m_{yy}) + \sqrt{\frac{1}{4}(m_{xx} + m_{yy})^2}$$

$$m_2 = \frac{1}{2}(m_{xx} + m_{yy}) - \sqrt{\frac{1}{4}(m_{xx} + m_{yy})^2}$$

Equation 7-16: Calculation of principal moments ($m_1 and\ m_2$) (Modified from: Beranek, 1972)

As τ_{xy} is zero. Therefore a logical method of examining the patterning of these values would be to overlay the $m_{yy}\ and\ m_{xx}$ Moire patterns over one another.

If the relationship for principal moment trajectories can be related to moment, deflection, or rotation of the plate surface, a mapping of the moment surface as the ideal shell surface should allow for the mapping of these principal moment lines to principal stress lines.

An examination tested the relationship between the flat shell displacements due to vertical loads in comparison to the total displacement of a shell. The idea came from the fact that if the plate moment hill is equivalent to the idealized shell shape then the plate surface stresses caused by moment (m) should be the shell principal stresses in membrane action. Therefore, since stresses (σ) are directly proportional to strain (ϵ) and strain is the first order derivative of the displacement of the plate, this should mirror quite closely to the stresses in a closed shell. In a flat plate these values would be the principal moment as the lines would be the fiber stresses on the surfaces of the plate, but since the moment shell is based on the idealized moment surface, these values are translated to in plane stress, specifically the principal stress lines.

$$\epsilon_{1,2\ bending} \in \sigma_{1,2\ membrane} \in m_{1,2\ plate}$$

Equation 7-17: Relationship Between Bending Strain, Membrane Stress, and Principal Moment (Own Work)

This however, also does not match expected results and instead appears to be very similar to the principal shear as visible in Figure 7-26.

In a twistless flat plate all rotation and deflection are the result of moments. Therefore on a twistless flat plate Equation 7-13 can be rewritten as in Equation 7-18:

$$n_{xx} \leftrightarrow \kappa_{yy} \leftrightarrow c_{yy}, n_{yy} \leftrightarrow \kappa_{xx} \leftrightarrow c_{xx}, n_{xy} \leftrightarrow -\kappa_{xy} \leftrightarrow -c_{xy}$$

Equation 7-18: Relationships Between In Plane Forces, Curvature from the moment shell, and full shell curvature (modified from Oosterhuis, 2010)

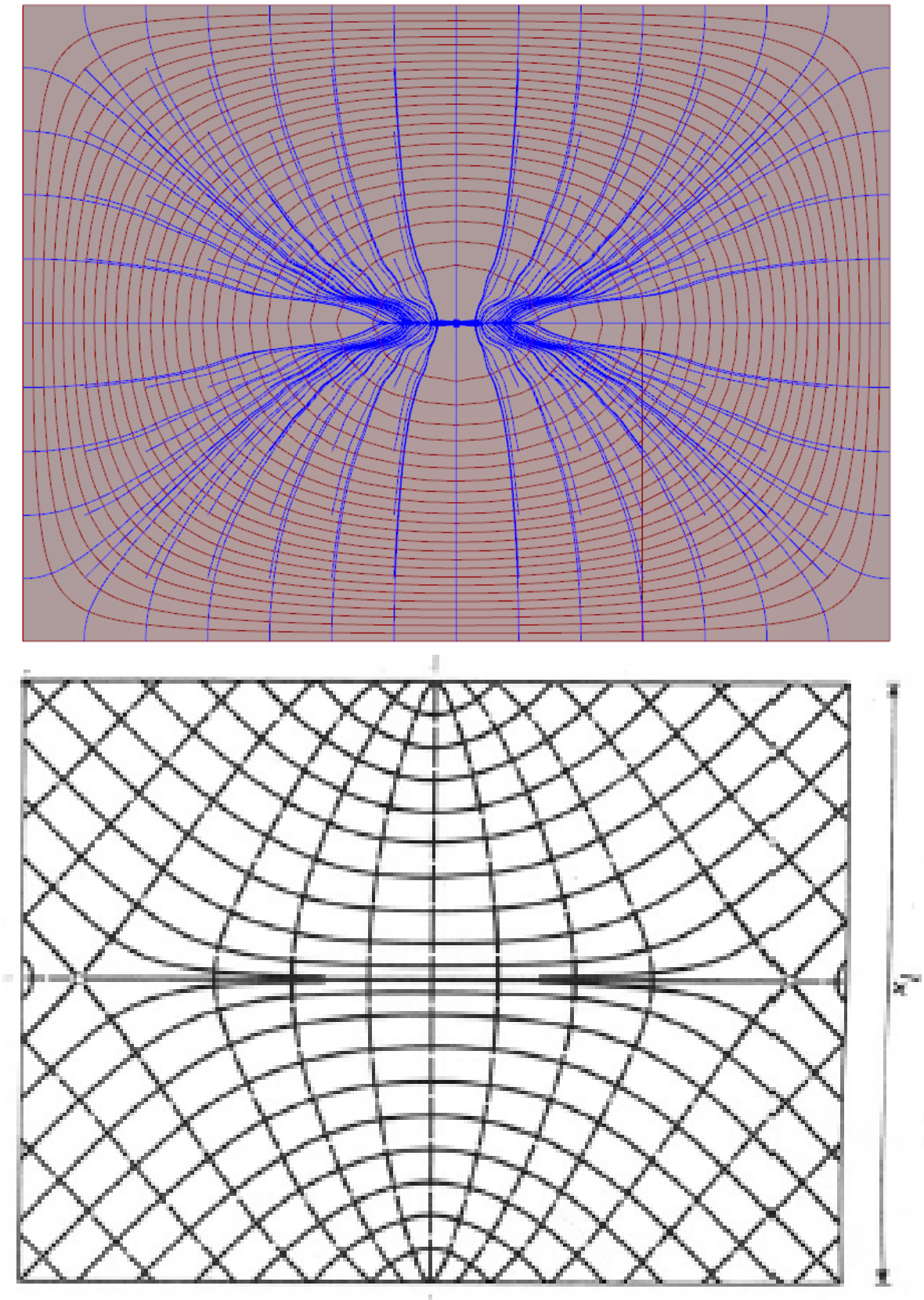


Figure 7-26: Comparison of Principal Gradient of the Deflected Form with Principal Moment (Own Image) and (Beranek, 1972)

Since we know that curvature is the derivative of the plate rotation, the principal curvature of the plate and thus the principal stresses should be equal to the steepest ascent of the rotation hill. Here in Equation 7-19, the principal descent of the angle of a flat plate, or principal curvature of the plate where the φ -hill (rotation hill) can be expressed as follows, where t is the steepest direction along the rotation hill:

$$\left(\frac{d\varphi}{dt}\right) = n_{tt}$$

$$\kappa_{xx} = \frac{d\varphi_x}{dx} = c_{xx}$$

$$\varphi_{\max} = \left(\varphi_x^2 + \varphi_y^2\right)^{\frac{1}{2}}$$

$$\text{Where: } n_1 = \frac{\partial \varphi_{\max}}{\partial n} = c_{\max} = c_1$$

Equation 7-19: Links between Rotation, moment and normal forces in a flat plate *(Modified From: Beranek, 1972)*

These equations state that by defining the curvature as the change in rotation, the maximum curvature (c_{max}) should be the maximum change in rotation or, in other words, the principal curvature (c_1) should be along the trajectory of the largest normal forces in a shell (n_{tt}), which in turn is the principal moment in the plate. Since the curvature in a plate is created entirely from moment, the curvature (c) should be equal to the curvature in the moment surface (κ) and therefore the principal normal force (n_1)should be equal in direction to the principal moment (m_1).

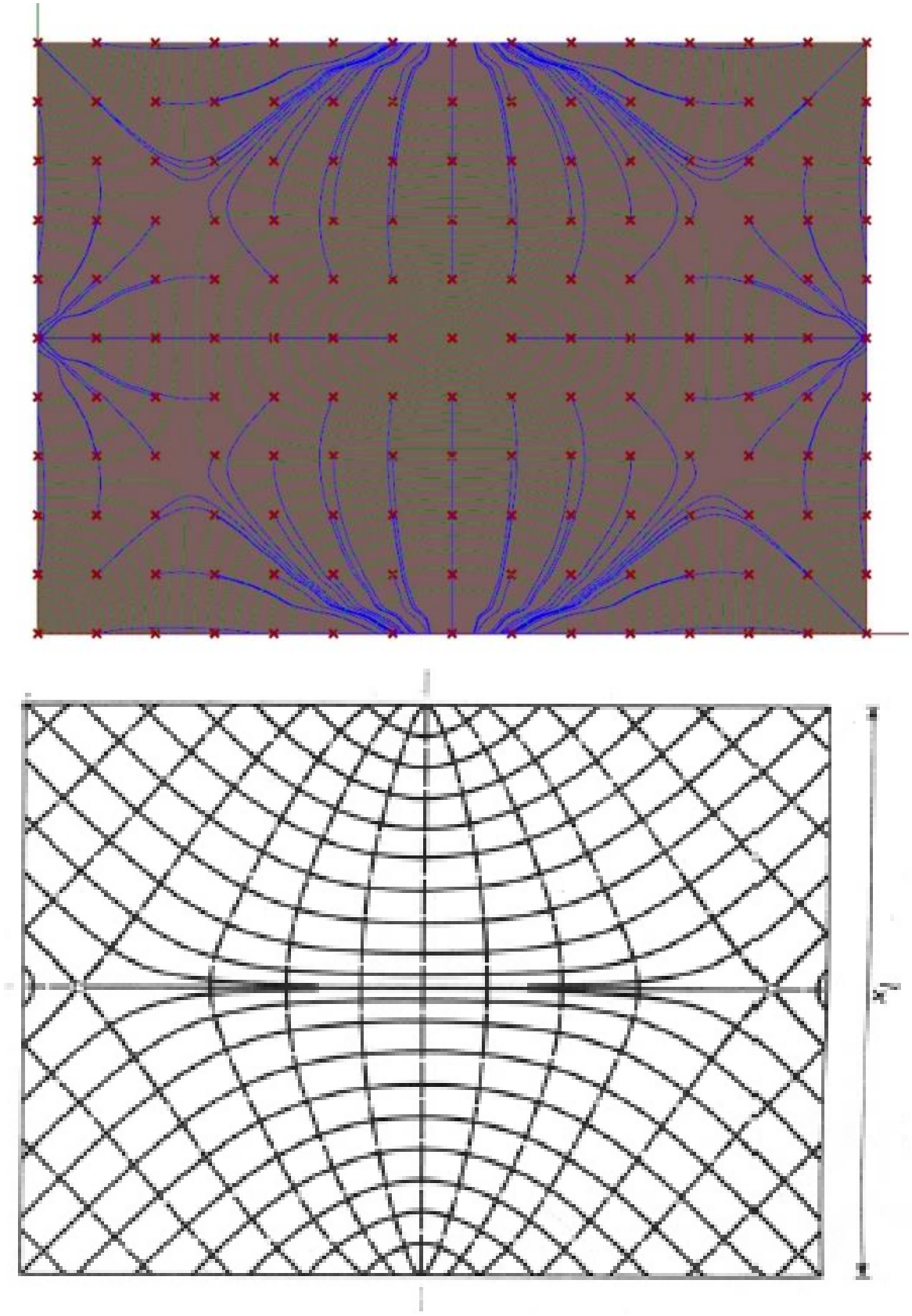


Figure 7-27: Isolevels and Streamlines of Total Rotation in an Edge Supported Plate (Own Image) and (Beranek, 1972)

One can see that while Figure 7-27 resembles the principal moments, it is not entirely correct. Therefore, the question became why does this not work? Is there perhaps twisting that needs to be accounted for? The current hypothesis is that twist would result in the streamlines experiencing convergence and curl. This is likely the culprit as convergence can be seen in the middle of the plate and in the corners where none is to be expected.

There also appears to be a combination occurring of the principal directions as the vector set does not mention which set contains the highest stresses at that point. Therefore it could be that some streamlines are shifting from σ_1 to σ_2.

If a method can be developed that utilizes the relationships between moment in a plate and normal forces in a thin shell, the pattern developed by this method could be easily mapped using set moment values and isolines to create the trajectories and map set values using UV mapping onto the shell surface. However, research in this area was limited and unfortunately did not yield a full result within the last several months of search.

8 FINAL SIZING AND LOADING

8.1 LOADING CONDITIONS:

Eurocode EN 1990 was used to develop the correct load conditions on the structure as laid out in the following paragraphs. This load was calculated as a value for the whole roof per unit area on which a load ratio script was used to distribute the load values to the proper joint locations. This assumes the worst case scenario for construction in which the glass is only structurally connected to the gridshell at the joints rather than evenly along the edge of each panel.

Self-Weight is calculated based on the weight of the glass rather than the steel sections as most of the weight of the structure comes from the glass mesh. The glass has depth of 6mm. According to CES Edupack 2016, the average weight of annealed glass is 2750 kg/m3. Assuming that standard 6mm panes of glass are used to span the roofing structure the self-weight load is measured out to be 161.85 kN / m2

Snow load according to Eurocode EN-1990 is defined by location. The Netherlands is located within Zone 3. Based on this location, the required snow load calculation is (0.614*Z-0.082+ (A/966)). Substituting in a Z of 0.4 for the zone parameter for the Netherlands and a A for the average height of the shell to be roughly 10m, the snow load is 0.17 kN/m2. The thermal coefficient is 1. While it would normally be higher for a primarily glass structure, due to the fact that the space underneath is not air conditioned unlike other roof structures, a normalized value must be used. The final coefficient is the exposure coefficient, this value is listed at 1.2 as accordance to Eurocode EN-1990-3. With all these values combined the structural load used per square meter on the structure is 163.94 kN/m2. Due to uncertainty, the load is multiplied by 1.4, the steel safety factor, giving a final loading condition of 229.51 kN/m2.

Since the full length of the courtyard is approximately 63 meters a serviceability requirement would be the $\frac{Max\ Span}{250}$. Therefore, required maximum displacement was 25cm.

8.2 SIZING OF STRUCTURAL STEEL AND STRUCTURAL PERFORMANCE:

The sizing of structural steel is typically conducted across three different sets of data for this parameterization, structural volume, strain energy, and mean shell behavior.

The first is structural volume; this is determined primarily by the axial stress and the bending moment in the beams. The bending moment in the beams will develop the structural depth of the beams and the axial stress in the system will generate a required area and thus thickness of the system required. This setup will therefore generate a minimum size CHS for each beam in order to generate a minimum theoretical structural volume The structure will be developed using a subset of Circular Hollow Sections (CHS) in order to reduce issues with angle implementation.

The second structural ranking will be developed based on strain energy. This valuation examines the amount of strain placed on the structure per unit volume and develops a consistent method of examining which system more accurately resolves the forces in the gridshell. Based on these two values, a proper assessment can be made about the validity of the results of this structural system.

The third ranking is the mean ratio of beam axial force to beam axial moment of the gridshells. Theoretically a perfect gridshell would have no moment, however, due to the nature of straight bar structures being impossible to follow the moment hill perfectly, there will never be 100% shell structure.

The final ranking is an average of the previous 3. With the highest scoring design receiving a score of 1 and each following, scoring percentage of that score.

$$Score = \frac{\frac{mass_{lowest}}{mass} + \frac{strainenergy_{lowest}}{strainenergy} + \frac{shellbehavior}{shellbehavior_{highest}}}{3}$$

Equation 8-1: Scoring Methodology

TYPE	MASS (KG)	STRAIN ENERGY (KJ)	MEAN SHELL BEHAVIOR	SCORE
BASE 1	368794	2527	0.8083	**0.911**
BASE 2 (DIAGONALS)	719527	2043	0.8736	**0.838**
BASE 3 (COMBINED)	669424	2535	0.7515	**0.739**
STREAMLINE	509424	2135	0.7894	**0.861**
STREAMLINE WITH MEAN CONVERGED NODES	509488	2139	0.7892	**0.860**
PGP	517174	2281	0.8615	**0.865**

Table 8.1: Scoring of Structural Options

The initial base that was in this case highly aligned with the principal stresses showed to be the most efficient method for this case. However, the second and third most efficient methods were in fact the Periodic Global Parameterization and the Streamline Generated method. This final score provides further validity of this system when compared to structures of similar density. This can be seen in Table 8.1 where the score for the streamline method generated in this thesis is 5.48% less efficient than optimized base method and less than 1% less efficient than the periodic global parameterized method. Considering the complexity of the parameterization scheme, this analysis is surprising.

The optimization is further validated based on the summation of the shear at nodes in the gridshell. For the streamline method the shear of the final gridshell is 2.40967E5kN. This compares well to the 1.33278E5 kN of the base UV design that follows the principal stresses and is an order of magnitude less in comparison to the 1.687E6 kN of the diagonalized gridshell.

Table 8.1 also shows that the expected further cleanup of the structure for constructability does begin to affect the shell behavior as the lines are forced to deviate from exactly along principal stress directions to mean nodal coordinates.

Figures 8-1 and 8-2 show the gridshell with mean converged coordinates in location spanning across the courtyard.

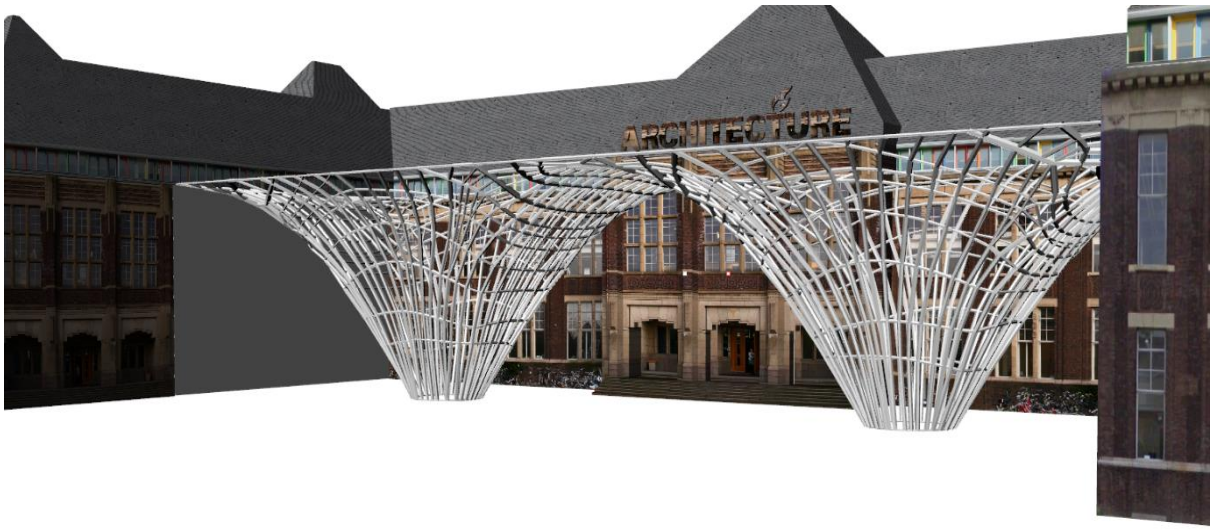


Figure 8-1: Streamline Gridshell In Courtyard

Figure 8-2: Elevation of Streamline Gridshell In Courtyard

8.3 DETAIL:

The detail developed in Figure 8-3 shows what sort of structural member would be best suited and how the structure would be best positioned for capping and developing a quick to construct system that has minimal issues with beam twist.

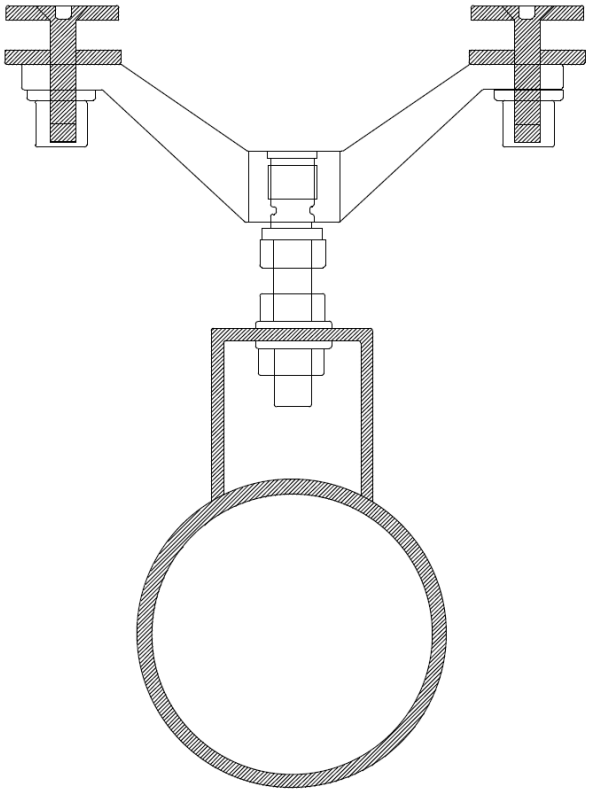


Figure 8-3: Possible Beam and Glass Connection Detail (Own)

Since many methods of beam structure rely on orienting flat surfaces outwards, significant variations in bar orientations can occur at nodes. This makes the development of nodes for gridshells a very time consuming and costly process. Typically there are two solutions to this problem:

The first solution is to parametrically model every node and examine nodes that are developed from the parametric meshing to see if there are any major issues. This method allows for the use of multiples sizes and shapes of profiles. However, it does create costly nodes and front ends the work to parametric design.

A simpler solution is to use CHS profiles. By using a CHS profiles any orientation of the profile is exactly the same. This means that as long as the glass for the structure is either set after the construction of the node and beam structure, which is almost guaranteed, and allows for a more rapid assembly of joints. This system also allows for the plates to be either attached only at the nodes, or fully along the bar structure without having to worry about the rotation of the surface, as a circle is always tangent to a plane at a single point. From here the glass direction is projected to the center of the beam allowing for the alignment of the connectors between the beam and the pieces of glass. This point connection also allows for significant discrepancies between the two panes and still have a decent connection.

9 CONCLUSION AND REFLECTION:

9.1 RESEARCH QUESTION

How can the creation of a gridshell by using principal stress directions be used to provide a more efficient structure that follows principal stress paths?

The principal stress paths can be traced using a streamline operator on a 2D flat mapping. This methodology generates a solution that is nearly as efficient as the Periodic Global Parameterization of similar density and is more efficient than an arbitrarily generated gridshell. However, the spacing on the structure is not tightly controlled as the system measures mapping distances on the 2D parameterized plane rather than in 3D which results in significant area distortion in some parts of the shell.

The streamline generator, however, was more efficient than most generic structural tessellations as shown in Results Table 8.1. A better mapping such as with Periodic Global Parameterization provides a slightly more efficient structure although not a considerable improvement.

9.1.1 Sub-questions

How can rod paths be plotted along principal stress streamlines on freeform surfaces?

Rod paths can be plotted using principal stress vectors through the use of a streamline generator. This generator is most viable on a 2D plane and therefore requires the mapping to and from a 3 dimensional surface in order to create proper rod pathing.

What form finding methods are suitable for generating an efficient structural form with high percentage shell behavior (no out of plane forces) and low strain energy density(high stiffness)?

The particle spring system was determined to be the most suitable method for implementation into the form finding section of the thesis due to the level of customization and integration facilitated by the Grasshopper3D environment via the Kangaroo plugin. This allowed for shells to be generated with highly customizable and definable parameters with mean shell behavior scores of over 90%, as shown in Table 6.3.

Is there a considerable advantage in optimizing a gridshell structure based on principal stress stream lines and an arbitrary generated tessellation?

Compared to arbitrary gridshell structures of similar density, there is an advantage to developing gridshell structures using both streamline mapping and parameterization. If the principal stress directions of the structure are difficult to

determine, then this method provides a useful solution to generating a concept structure for future development.

However, if the structure has easily determinable principal stress directions, creating a UV structure via intuition would be a more efficient process. Of note, the system does not integrate all required construction limitations that would be necessary for this system to create a viable panelization scheme. The most important check would be the curvature of the panels. Since both the stream and periodic global parameterization systems do not area normalize, there is a significant probability that many of the glass panels for the roofing would need to be pre-formed off site via hot bending techniques.

9.2 POSSIBLE FUTURE STEPS:

9.2.1 Computational

While the current tracer does account for some vector field smoothing, there are still significant issues that occur around the singularities. The large amounts of curl that occur in these areas distort and warp both streamline and parameterizations and thus creates discontinuities in the bar paths and orientation.

There are a significant number of possible steps to be taken in future investigations.

9.2.1.1 Better Tracing and Streamline Analysis

One method of advancing this system would be to integrate constructability requirements into the data set so that beam spacing, too close together or too far apart, signal the necessity for beam convergence or insertion. However, to do this and maintain a proper UV mapping, the main streamline tracer would have to be rewritten to allow for parallel input points and allow for a continual distance check between the points as the streamlines are generated. Another approach would be to cleanup all points by checking their relative spacing in the same grid direction and all points within a specified distance, for example all points within 0.2m, are consolidated to a mean point. This would bring converged streamlines together and create a more even spread of streamlines on the vector field.

If this system is properly implemented and a bar structure can be created without manual input, full parametric design can occur with fluid data flow. This data flow would allow for much more analysis to occur such as optimizing for constructability by checking warping or area conformance and structural sizing.

9.2.1.2 Generating a Curvature Field

Based on current technology glass panels can be cold bent to provide some geometric flexibility. Since the current grid is based on polyline coordinates and thus generated from straight bar elements, there cannot be any single curvature in the panels, only skewness, generating double curvature. Schober in his 2015 book “Transparent Gridshells” states that current limits as follows.

$$W = D/175$$

Equation 9-1: Glass Panel Warping Limitations (Schober, 2015)

Where the warping factor (W) is based on the average diagonal length of the quadrilateral panel (D) where d_1 and d_2 are the two diagonal lengths.

$$D = \frac{d_1 + d_2}{2}$$

Equation 9-2: Average Diagonal Length (Schober, 2015)

Quadrilateral planarization can be completed in several methods. Typically Planar Quad (PQ) meshes are generated using the principal curvature as a guiding vector set for generating a quadrilateral system in geometry processing libraries. Using principal curvatures is the simplest method to generate a planar quadrilateral approximation of the surface; this is because by aligning the principal curvature with the planar quadratic meshes, the resultant mesh will have as little skew as possible (Ray et al., 2006).

Winslow (2010) opted instead for a formula utilizing an area normalization coefficient and ignored the curvature vectors of the gridshell. He therefore ignored the skewness, arguing that most panels would be developable as most gridshells exhibit low curvature and therefore low skewness. This is not necessarily robust across all designs and leaves the designer exposed to some non-conformance.

A curvature analysis can be used in order to create the principal curvatures c_1 and c_2 directions at each node point would allow for the weighting of these vectors in comparison to the principal stress vectors (σ_1 and σ_2).

However, for a gridshell not every panel has to be perfectly planar. Therefore in order to develop a planar surface, the principal curvature of a patched surface is determined at the same point of analysis. Once each principal curvature vector is matched to a principal stress vector (based on which angle in each orthogonal set is the smallest) a weighting scheme is developed to generate a new vector grid. Where:

$$\hat{\boldsymbol{r}} = W\hat{\boldsymbol{s}} + \left(1 - W\right)\hat{\boldsymbol{c}}$$

Equation 9-3: Vector Weighting Function for Defining Principal Stress vs Curvature Lines (Own Work)

In which**:**

$$\hat{\boldsymbol{r}} = unitized\ rationalized\ vector\ set$$

$$W = scalar\ weighting$$

$$\hat{\boldsymbol{s}} = unitized\ stress\ vector\ set$$

$$\hat{\boldsymbol{c}} = unitized\ curvature\ vector\ set$$

While this set was not used in the end for computation, it is hopefully useful in further investigations.

9.2.1.3 Using Conformal Mapping / Periodic Global Parameterization

In comparison to Panagiotis and Sawako's Millipede(Panagiotis & Sawako, 2014) algorithm the other main shortcoming appears to be the regularity of the grid. This can be explained by the parameterization choice: if instead a pseudo-isometric parameterization could be generated, then the streamline spacing algorithm would have less spacing variability.

Therefore, it is probable that a proper pseudo-isometric or conformal map would generate a closer representation of this parameter. However, for this to work the translation function would need to be known for each mesh face in order to ensure that the transformation also occurs to the corresponding vector set, which was unfortunately not possible within the mapping available while using Kangaroo.

9.2.2 **Analytical Analysis**

An attempt was made at the analytical derivation of the principal stresses in a shell so that significant computational work and complexity of the geometric process could be simplified. By approaching the system from a fundamentals perspective, there is a chance that these methods may be derivable from the loading characteristics of a flat plate. It is already known that the moment hill ($\overline{m}$) of a flat plate with identical load conditions will create the idealized funicular shape when Calladine's two surface theory is used.

From here it is also possible to develop the principal shear direction which happens to be the principal curvature of the moment hill shell. There appear to be several usable relationships that could possibly generate the same relationship for principal stresses, specifically analyzing the rotation values of a plate in comparison to the mapped shell form. If this is the case, a proper relationship can be determined between principal moments and in plate surface geometry, then the ability to generate usable isocurves is much more stable and results in fewer issues such as vector field singularities. Instead, these will be low points in the surface and, therefore, not induce the same amount of uncertainty and rotation that is created in the parameterization and streamline method.

9.3 REFLECTION

The form finding methods outlined in this thesis were a success, and the application of a variety of form finding algorithms did yield more efficient structural solutions within the standard context of particle spring loading methodologies. These methods more accurately model real world loading characteristics to a digital form finding mechanism. Form finding via analytical solutions through determining the moment surface is also a viable option for most shells. The system yields results with average shell behaviors often high in the 90%+ range, as shown in Table 6.3, while dealing with complications that analytical structures have difficulties with, including mixed height supports. This form finding system was also shown to be viable for gridshells developed from this structure reaching scores averaging 81.7% from Table 8.1

The second part of the thesis, however, was less successful. There were several issues that occurred within the thesis. Firstly, streamlines even when mapped to a 2D mesh of the original shape to reduce integration step errors are not usable as a method of optimization. The streamline generation along principal stress vectors for gridshell development requires the result to become a discrete grid unlike most other engineering applications of streamlines. Typically streamlines are used for visualization of flow analysis rather than generating a connected network. Even through the use of mapping to a 2D surface and only working with orthogonal vectors, the system still did not efficiently work. The result yielded a messy and unrefined structural flow that in turn resulted in major manual cleanup.

Other methods such as a periodic global parameterization through the Millipede black box parameterization engine did also not work. These two systems, streamline tracing and periodic global parameterization, both ran into significant issues around the singularities that may be resolved with more focus utilizing discrete differential mathematics.

The primary issues with the current methodology are singularities in the vector field and area distortion. These singularities cause discrepancies in the flow data and thus result in a point of convergence or divergence of the vector field due to curl. Despite efforts in using evenly spaced streamline algorithms, this approach was still unsuccessful.

The area distortion also yields some panels larger than the 1 meter desired spacing, especially around the areas where the gridshell is highly vertical. This can clearly be seen when comparing the periodic global parameterization methodology with that of the streamline algorithm.

The mathematics and mappings of parameterization are still unfortunately complex and unwieldy with some simple gridshells requiring multiple sub-maps and joining functions which was beyond the scope of this thesis.

While this paper does eventually develop a gridshell, it requires significant manual editing and development that is unsatisfactory for a fully parametric model for optimization. Some of the reduction was developed later on in the thesis, but would need to be more rigorously tested before implemented fully in an optimization algorithm. One of the designs chosen to compare against the base system was also in the end developed via black box methods. This was because the periodic global parameterization engine was not viable to run within the Grasshopper3D system. This would have needed to be fully rewritten from base discrete differential geometry in order to be implemented, which was unfortunately beyond my current capabilities and the time scope of this thesis.

Instead, this paper in turn explored multiple current methodologies with none being quite satisfactory from a parametric perspective. While this is in no small part due to my failure at understanding the complex mathematics of discrete differential geometry, it shows that this method is relatively impractical for the architecture student. This is confirmed by the primary papers in the field being a PhD thesis from civil engineering and other conformal mapping papers being co-written by mathematicians focusing in discrete differential geometry.

Instead, the research presented at the end of the paper that focuses on the plate behavior and utilizing structural theory may provide a better solution.

My belief is that this method of examining principal grid layout appears to be correct on the surface in order to generate grids for all forms, but is inherently flawed. Since principal stresses are a second order tensor and a directional derivative of derived stresses, these are often non-smooth. In order for the data to be viable either intensive smoothing processing or curl correction would need to be applied. Secondly, often times, principal stresses can be plotted far more quickly by hand than by utilizing this system, while slightly more difficult on more complex surfaces, often time engineering intuition will be faster than that of utilizing parametric systems.

The thesis relates to Building Technology by attempting to map 2nd order tensor sets using computation to generate more efficient lightweight and thin structures. Thin, lightweight and open structures such as gridshells are becoming more common around the world by the year because of their high span to structural depth ratio due to their form. This therefore attempts to further discretize the issues of the structure and also optimize their topology.

Discrete mapping of principal stresses would be useful in a variety of structures, however, there are more efficient ways of developing such structure if loads are concentrated at a point. But for structural systems under continuous load conditions made of bar elements, further work is required and my belief is this thesis has highlighted directional areas for future investigation in order to achieve these objectives.

9.4 Research Path

The research method was unfortunately not well defined. The examination began at wanting to optimize along principal stress directions which took a turn into examining mappings and mesh parameterization, while also diverging to examine plate mechanics and shell mechanics at the same time. I believe that members of the academic community with more advanced mathematical knowledge will be able to derive a more useful surface, and in fact in some cases have done so as seen with Winslow's PhD Thesis. Had this thesis focused instead on only one singular direction and track, a clearer and more developed thesis would likely have been developed. The main issue is that there were far too many parameters to examine in the scope of one master's thesis and would make a much better doctoral thesis, as evidenced by Winslow's thesis from 2010.

There were no conflicts of interest during the thesis.

During the research some extra scripts were also generated in expectancy of a successful generation of a grid generation. These are available in the final appendix.

10 BIBLIOGRAPHY

Adriaenssens, S., Block, P., Veenendaal, D., Williams, C., & Williams, C. A. (2014). *Shell Structures for Architecture: Form Finding and Optimization*. London, UNITED KINGDOM: Routledge. Retrieved from http://ebookcentral.proquest.com/lib/delft/detail.action?docID=1656820

Baudet, V., Beuve, M., Jaillet, F., Shariat, B., Zara, F., Beuve, M., … Zara, F. (2007). *New Mass-Spring System Integrating Elasticity Parameters*. LIRIS.

Beranek, Ir. W.J. (1972). *College: Draagkonstrukties Onderdeel: Platen en Schijven*, Technische Hogenschool Delft, Delft

Bendsoe M., and Sigmund, O. (2003). Topology Optimization, Theory, Methods and Applications. Springer, Berlin.

Brandmaier, H. Optimum Fillament orientation criteria. Journal of Composite Materials 4, (1970), pp. 422-425.

Dawber, P. G. (1987). Vectors and vector operators. Bristol: Hilger.

Floater, M. S., & Hormann, K. (2005). Surface Parameterization: a Tutorial and Survey. In N. A. Dodgson, M. S. Floater, & M. A. Sabin (Eds.), *Advances in Multiresolution for Geometric Modelling* (pp. 157–186). Berlin/Heidelberg: Springer-Verlag. https://doi.org/10.1007/3-540-26808-1_9

Fritzsche, J. Gridshell efficiency optimization; optimizing efficiency form- & grid-configuration through iterative approximation and minimization strain energy, Master thesis, TU Eindhoven, 2013.

Frei Otto. (1974). *IL 10: Gitterschalen / Gridshells*. Institute for Lightweight Structures. Retrieved from https://www.amazon.com/10-Gitterschalen-Grid-Shells/dp/B001SBUX20?SubscriptionId=AKIAIOBINVZYXZQZ2U3A&tag=chimbori05-20&linkCode=xm2&camp=2025&creative=165953&creativeASIN=B001SBUX20

Gea, H. C., & Luo, J. H. (2004). On the stress-based and strain-based methods for predicting optimal orientation of orthotropic materials. *Structural and Multidisciplinary Optimization*, *26*(3–4), 229–234. https://doi.org/10.1007/s00158-003-0348-x

Gortler, S. J., Gotsman, C., & Thurston, D. (2006). Discrete one-forms on meshes and applications to 3D mesh parameterization. *Computer Aided Geometric Design*, *23*(2), 83–112. https://doi.org/10.1016/j.cagd.2005.05.002

Hansen, Glen A.; Douglass, R. W; Zardecki, Andrew (2005). Mesh enhancement. Imperial College Press. p. 404.

Harding, J., Shepherd, P. (2011). Structural form finding using zero-length springs with dynamic mass. In *2011 IASS Annual Symposium: IABSE-IASS 2011: Taller, Longer, Lighter*. University of Bath. Retrieved from http://opus.bath.ac.uk/25185/

Huang, X, Xie, Y.M., & Wiley InterScience (Online service). (2010) Evolutionary topology optimization of continuum structures: Methods and applications. Chichester, West Sussex:Wiley.

Kobourov, S. G. (2012). Spring Embedders and Force Directed Graph Drawing Algorithms. *ArXiv:1201.3011 [Cs]*. Retrieved from http://arxiv.org/abs/1201.3011

Li, Y., & Chen, Y. (2010). Beam Structure Optimization for Additive Manufacturing based on Principal Stress Lines, 13.

Mebarki, A., Alliez, P., & Devillers, O. (1992). Farthet Point Seeding for Placement of Streamlines. *Bulletin of Sociological Methodology/Bulletin de Méthodologie Sociologique*, *37*(1), 55–57. https://doi.org/10.1177/075910639203700105

Meyer, M., Desbrun, M., Schröder, P., & Barr, A. H. (2003). Discrete Differential-Geometry Operators for Triangulated 2-Manifolds. In H.-C. Hege & K. Polthier (Eds.), *Visualization and Mathematics III* (pp. 35–57). Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-662-05105-4_2

Michell, A. G. M. (1904). The limits of economy of material in frame-structures. Philosophical Magazine Series 6, 8(47), 589–597. https://doi.org/10.1080/14786440409463229

Miura K. & Pellegrino S.(Forthcoming), Structural Concepts. Cambridge, UK: Cambridge University Press

NEY & Partners | Projects | Glass roof Dutch Maritime Museum | 10915 | Amsterdam. (n.d.). Retrieved October 26, 2018, from http://www.ney.be/project/glass-roof-dutch-maritime-museum.html

Oosterhuis, M. R. (2010). A parametric structural design tool for plate structures. Retrieved from http://resolver.tudelft.nl/uuid:09e3b9ad-4b22-4ccb-88ef-39792d6a5e54

Pellegrino, S. & Calladine, C.R. (1986). Matrix Analysis of Statically and Kinematically Indeterminate Frameworks. International Journal of Solids

and Structures, 22(4), 409-428. https://doi.org/10.1016/0020-7683(86)90014-4

Pilkington. (n.d.). Glass and Mechanical Strength, 17.

Ray, N., Li, W. C., Lévy, B., Sheffer, A., & Alliez, P. (2006). Periodic Global Parameterization. *ACM Trans. Graph.*, 25(4), 1460–1485. https://doi.org/10.1145/1183287.1183297

Redevelopment of King's Cross station - Arup. (n.d.). Retrieved October 26, 2018, from https://www.arup.com/en/projects/kings-cross-station

Richardson, J. N., Adriaenssens, S., Filomeno Coelho, R., & Bouillard, P. (2013). Coupled form-finding and grid optimization approach for single layer gridshells. Engineering Structures, 52, 230–239. https://doi.org/10.1016/j.engstruct.2013.02.017

Rozvany, G. I. N. (2008). A critical review of established methods of structural topology optimization. Structural and Multidisciplinary Optimization, 37(3), 217–237. https://doi.org/10.1007/s00158-007-0217-0

Samuelsson, A., & Zienkiewicz, O. C. (2006). History of the stiffness method. International Journal for Numerical Methods in Engineering, 67(2), 149-157. https://doi.org/10.1002/nme.1510

Schober, H. & Schaffert, C. (2016). *Transparent shells: form, topology, structure*. Berlin: Ernst & Sohn.

Shapiro, L., & Stockman, G. (2000). *Computer Vision*. The University of Washington and Michigan State University. Retrieved from http://nana.lecturer.pens.ac.id/index_files/referensi/computer_vision/Computer%20Vision.pdf

Smidt, D. Freeform follow functions, Master thesis, TU Delft, 2014.

Strang, G., & Kohn, R. V. (1983). Hencky-Prandtl nets and constrained Michell trusses. Computer Methods in Applied Mechanics and Engineering, 36(2), 207–222. https://doi.org/10.1016/0045-7825(83)90113-5

Tutte, W. T. (1963). How to Draw a Graph. *Proceedings of the London Mathematical Society*, *s3-13*, 743–767. https://doi.org/10.1112/plms/s3-13.1.743

Winslow P, Pellegrino S, Sharma S.B. Multi-objective optimization of free-form grid structures. Structural and multidisciplinary optimization, 4(1-6), p257-269, 2010.

The Weald and Downland Gridshell - BuroHappold Engineering. (n.d.). Retrieved October 26, 2018, from https://www.burohappold.com/projects/weald-downland-gridshell/

www.fosterandpartners.com, F. + P. /. (n.d.). Projects | Foster + Partners. Retrieved October 26, 2018, from https://www.fosterandpartners.com/projects/

Yas Hotel. (n.d.). Retrieved October 26, 2018, from https://www.sbp.de/en/project/yas-hotel/

10.1 SOFTWARE

Mirtschin, J. (2015). GeometryGym (Version 8.6). Australia: Geometry Gym Pty Ltd. Retrieved from https://geometrygym.wordpress.com/downloads-windows/

Oasys GSA. (2017). (Version 8.7.0.80). Oasys-Software / ARUP. Retrieved from https://www.oasys-software.com

Panagiotis, M., & Sawako, K. (2014). *Millipede*. Sawapan. Retrieved from http://www.sawapan.eu/

Piacentino, G. (2015). Weaverbird. Retrieved from http://www.giuliopiacentino.com/get-wb/

Piker, D. (2017). Kangaroo (Version 2.42). Kangaroo3D. Retrieved from http://kangaroo3D.com/

Preisinger, C. (2016). Karamba3D (Version 1.2.2). Bollinger und Grohmann ZT GmbH. Retrieved from https://www.karamba3D.com/download/

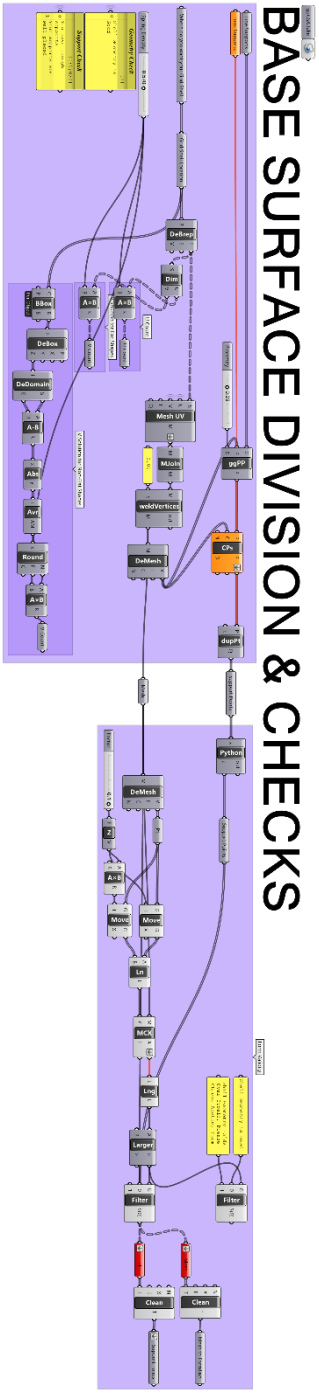
Rhino 5. (2017). Robert McNeel & Associates. Retrieved from rhino3D.com/download

Rutten, D. (2014). Grasshopper3D (Version 0.9.0.0076). Robert McNeel & Associates. Retrieved from rhino3D.com/download

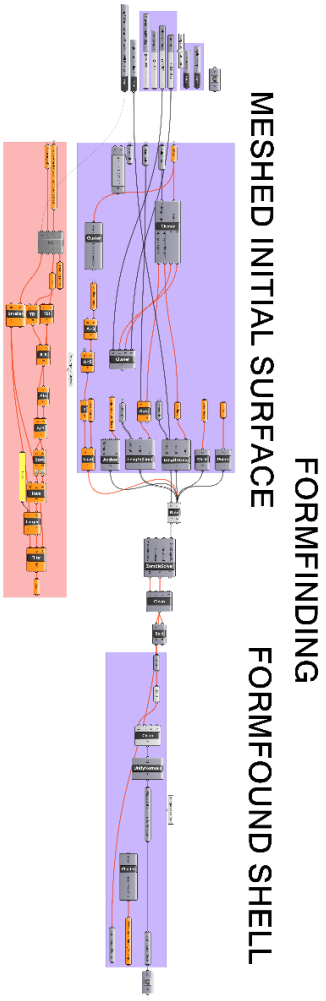
11 APPENDICES

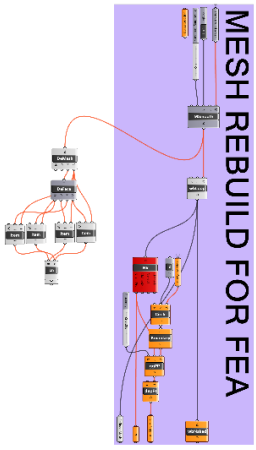
11.1 APPENDIX 1: GRASSHOPPER3D SCRIPT

11.1.1 Appendix 2: UV Generation

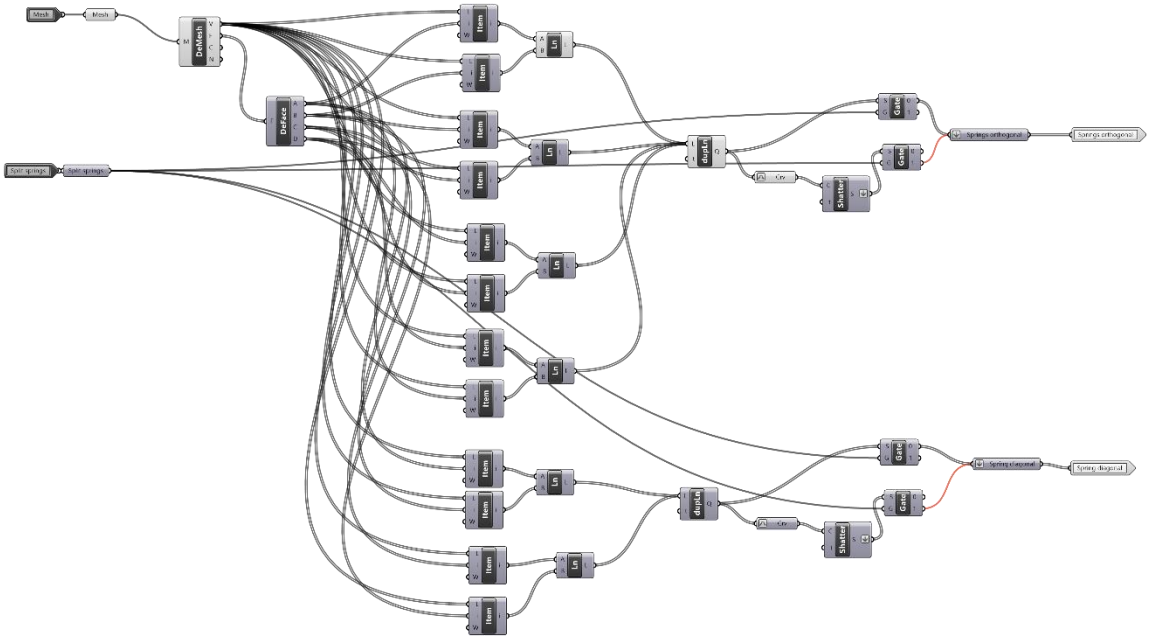


11.1.2 **Appendix 3: Form Finding and Remeshing**

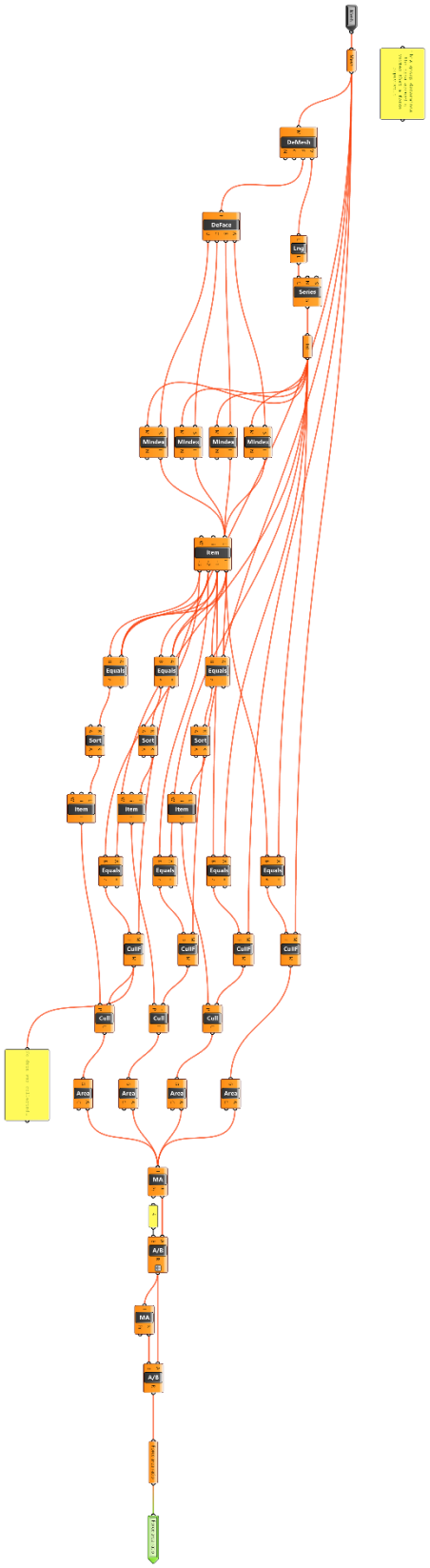




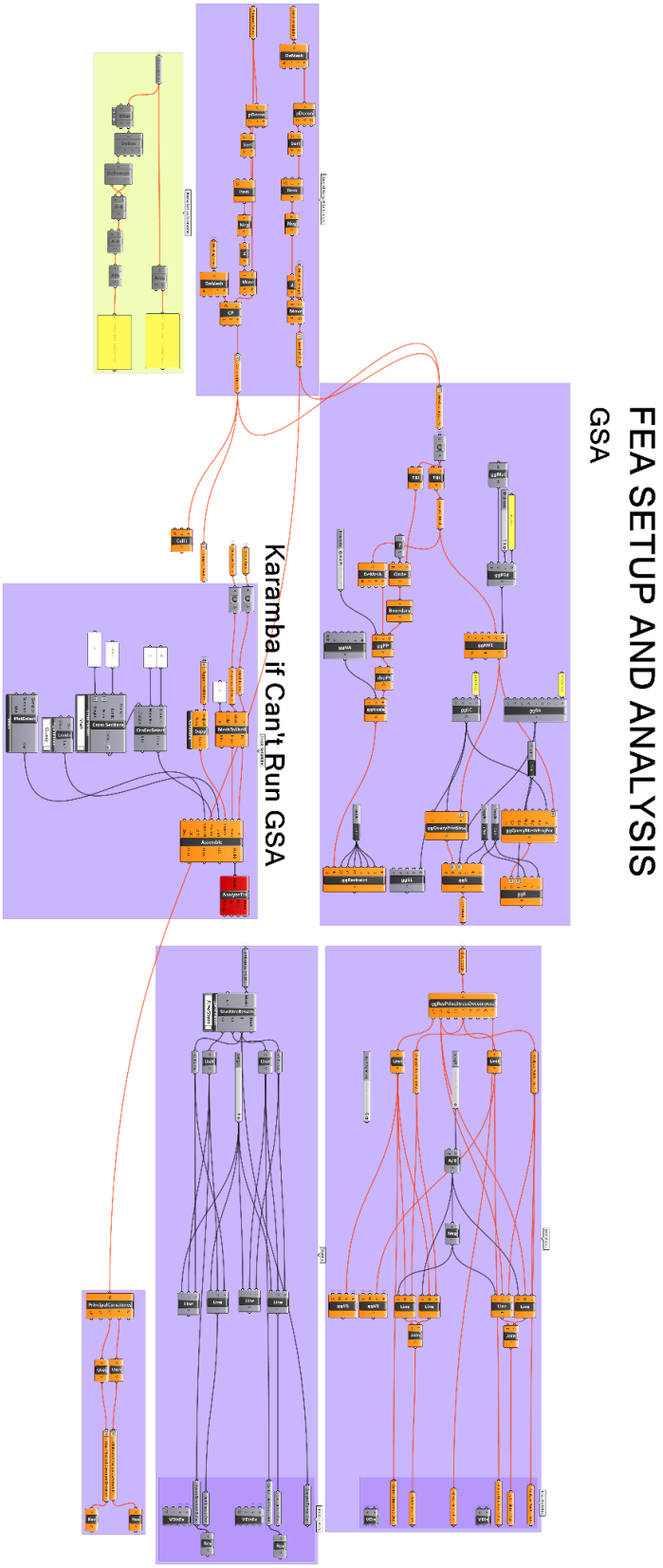
11.1.3 **Orthogonal and Diagonal Lines of a Quad Mesh**



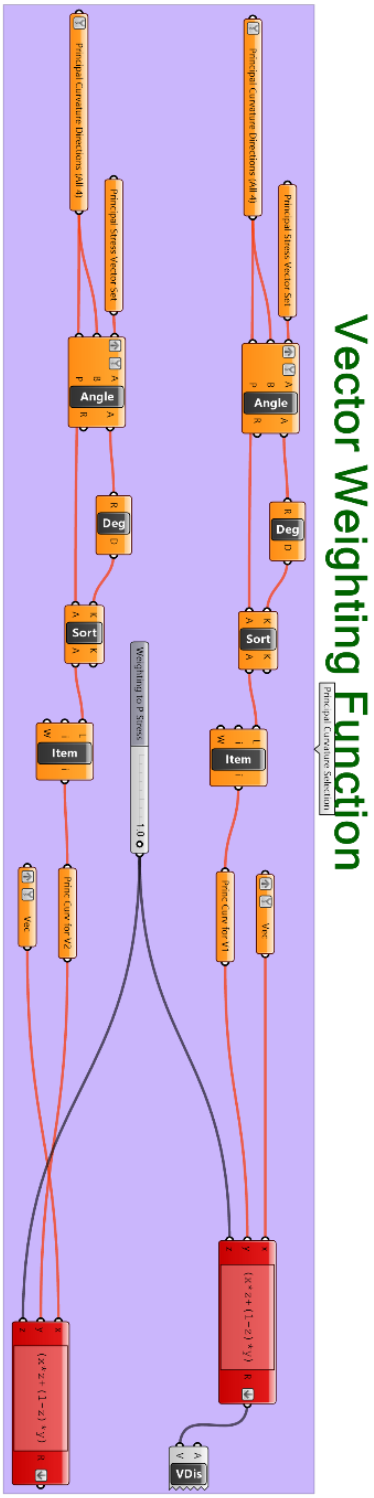
11.1.4 **Force Area Ratio**



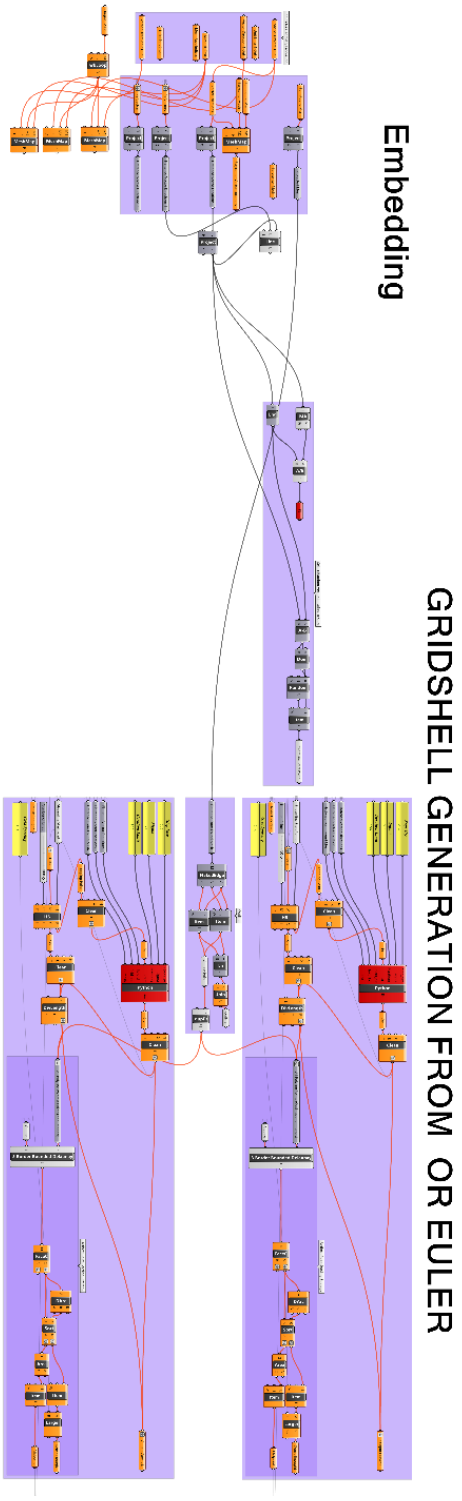
11.1.5 **Appendix 4: Principal Stress FEA**



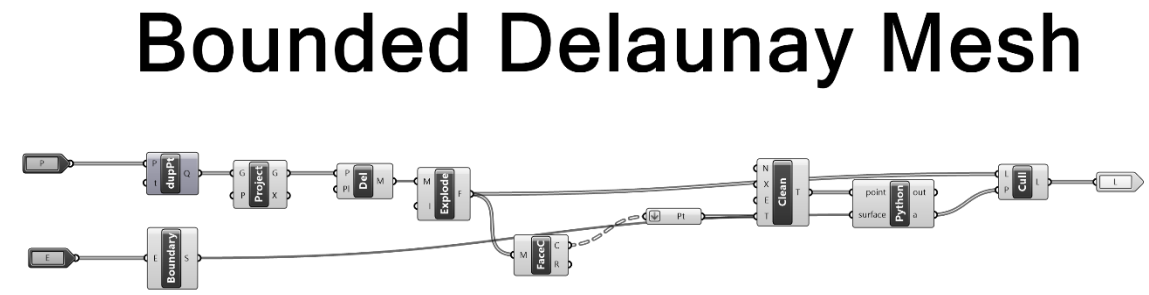
11.1.6 **Vector Weighting (Unused)**



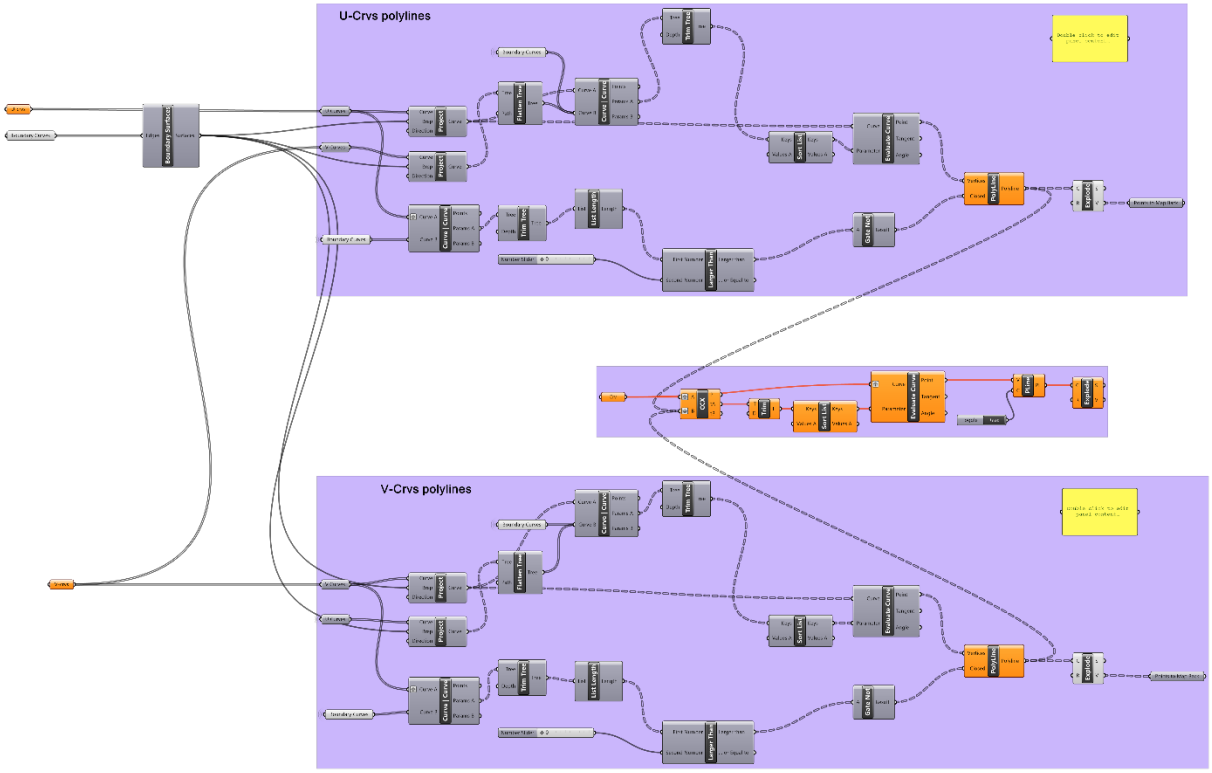
11.1.7 Mapping to 2D and Tracer with Evenly Seeded Streamline Algorithm



11.1.8 **Bounded Delaunay Mesh**

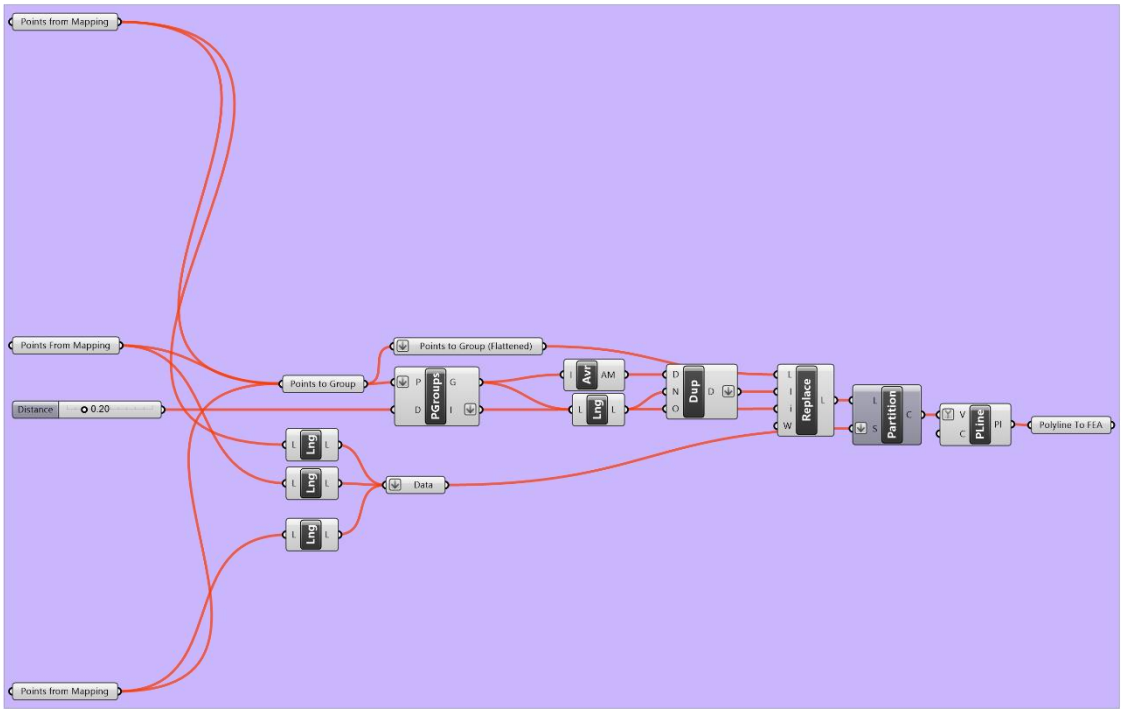


11.1.9 **UV Discretization**

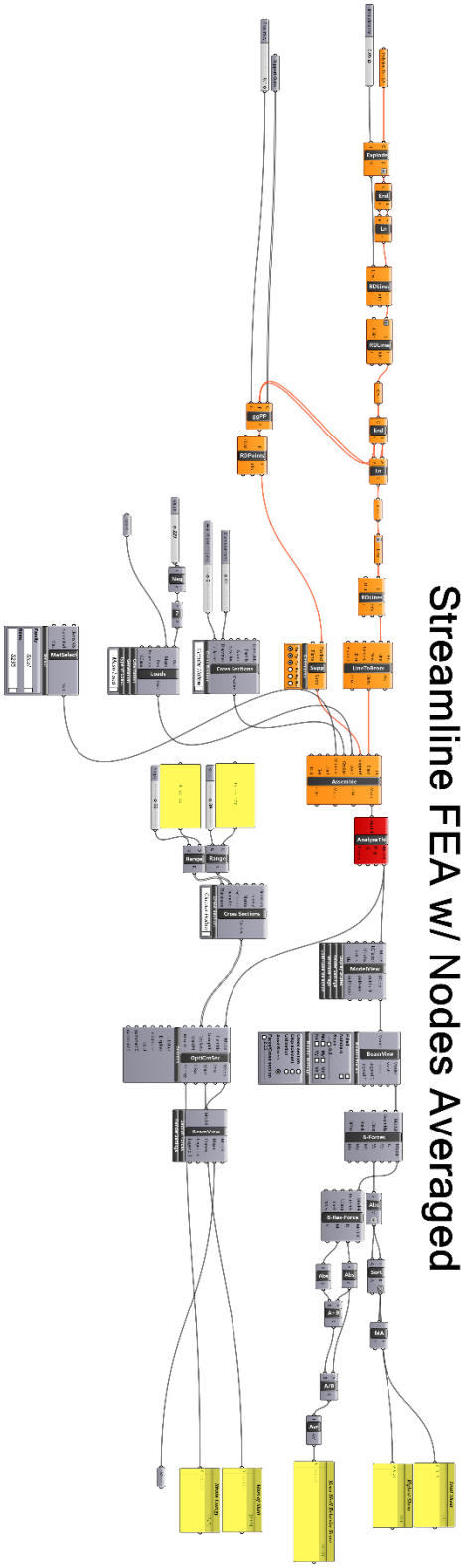


11.1.10 : **Point Grouping and UV Simplification**

Mean Locations of Converged Points



11.1.11 **Final Structural Analysis**



11.2 Appendix 2: Python Code

11.2.1 Streamline Generator

```
import math
import functools
import rhinoscriptsyntax as rs
import Rhino.Geometry as rg

#Coerce
field_loc = rs.coerce3Dpointlist(field_loc)
field_dir = rs.coerce3Dpointlist(field_dir)
start     = rs.coerce3Dpoint(start)

#Gaussian funciton
def gauss_fun(values,sig=1,mu=0):
    return [ math.exp(-(x-mu)**2/(2*sig**2)) for x in values]

#Compute stress at location
def stress(loc):
    dist = rs.Distance(loc,field_loc)
    weights = gauss_fun(dist,sigma)
    weights_sum = sum(weights)
    return functools.reduce(lambda x,y: rs.PointAdd(x,y), [ w*s/weights_sum for s,w in zip(field_dir, weights)])

#Compute one step for Euler integration
def step(loc,dir,stress):
    # Norm stress vector
    stress = stress/rs.VectorLength(stress)
    # Norm direction
    dir = dir/rs.VectorLength(dir)
    # Follow tangent
    if rs.VectorDotProduct(stress,dir)<0:
        stress = rs.VectorReverse(stress)
    # Compute step delta
    delta  = stress*step_size
    # Return
    return loc+delta,stress

#Check boundary condition
def check_boundary(loc):
    return rs.IsPointOnMesh(mesh,loc)

#Check for looping
def check_looping(loc,locations):
    #return None
    dist = rs.Distance(loc,locations)
    min_dist = min(dist)
    return locations[dist.index(min_dist)] if min_dist<0.99*step_size else None

#Prune line from loop
def prune_head(point,line):
    try:
        return line[0:line.index(point)+1]
    except:
```

```
        return []

def prune_back(point,line):
    print(len(line))
    print(line.index(point))
    try:
        return line[line.index(point)-1::]
    except:
        return []

############### Main ################
def iterr(loc, dir):
    line1,line2 = [loc],[loc]
    loc1,loc2 = loc,loc
    dir1,dir2 = dir,rs.VectorReverse(dir)
    run1,run2 = True,True
    for i in range(iterations):
        run1 = run1 and check_boundary(loc1)
        if run1:
            loc1,dir1 = step(loc1,dir1,stress(loc1))
            point = check_looping(loc1,line1+line2)
            if point is not None:
                line2 = prune_head(point,line2)
                if check_looping(loc1,line1) is not None:
                    line1 = prune_back(point,line1)
                line1.append(point)
                break
            else:
                line1.append(loc1)
        run2 = run2 and check_boundary(loc2)
        if run2:
            loc2,dir2 = step(loc2,dir2,stress(loc2))
            point = check_looping(loc2,line1+line2)
            if point is not None:
                line1 = prune_head(point,line1)
                if check_looping(loc2,line2) is not None:
                    line2 = prune_back(point,line2)
                line2.append(point)
                break
            else:
                line2.append(loc2)
        if not run1 and not run2:
            break
    return list(reversed(line2[1::]))+line1



            line = iterr(start,stress(start)/rs.VectorLength(stress(start)))
            try:
            #Normal Polyline Generation
                line = rs.AddPolyline(line)
            except:
            #Circle Developed if Point Starts Directly in Singularity
                line = rs.AddCircle(start, 0.1)
```

11.2.2 Support Checks

```
import math
import rhinoscriptsyntax as rs

xValues = []
yValues = []
equalX = None
equalY = None
x = rs.coerce3Dpointlist(x)

def checkEqual1(iterator):
    iterator = iter(iterator)
    try:
        first = next(iterator)
    except StopIteration:
        return True
    return all(first == rest for rest in iterator)


### Main Condition Check ###
if x is None:
    print "Not enough supports"
    a = None
else:
    if len(x)<3:
        print "Not enough supports"
        a = None

    if len(x)>=3:
        print "You have enough supports"

        for i in x:
            xValues.append(i[0])
            yValues.append(i[1])

        equalX = checkEqual1(xValues)
        equalY = checkEqual1(yValues)
        if equalX == True:
            print "Your supports are in a line"
            a = None
        if equalY == True:
            print "Your supports are in a line"
            a = None
        else:
            print "Your supports are well placed"
            a=x
```