MOSM – Design for Additive Manufacturing

Tutorial:

Free form & periodic structure creation via Rhino GH Applications

Version 1.0

Spring 2024

Dr. Zhiping WANG

Lecturer at SeaTech, École d'ingénieurs de l'université de Toulon

zhiping.wang@univ-tln.fr

1. Introduction

Objective:

In this TP, we will learn how to generate bio-inspired structures via Rabbit and Millipede plugins in GH. Rabbit is an open-source plug-in for Grasshopper that simulates biological and physical processes. Millipede is a structural analysis and optimization plugin. Additionally, it exposes functionality that is relevant to the solution of many numerical and geometric problems.

Demonstration tool: Rabbit, Millipede and Weaverbird.

Rabbit comes with a set of components allowing you to model cellular automata and L-systems directly inside Rhino and Grasshopper 3D. It also provides an easy way to explore natural phenomena such as pattern formation, self-organization and emergence.

Millipede is a structural analysis and optimization component for grasshopper. It allows for very fast linear elastic analysis of frame and shell elements in 3d, 2d plate elements for in plane forces, and 3d volumetric elements. All systems can be optimized using built in topology optimization methods and have their results extracted and visualized in a variety of ways.

Weaverbird is a topological modeler that contains many of the known subdivision and transformation operators, readily usable by designers. Instead of doing the work repeatedly, or sometimes using complicated scripts, this plug-in reconstructs the shape, subdivides any mesh, even made by polylines, and helps preparing for fabrication.

Rabbit: https://morphocode.com/rabbit/

Millipede: https://drive.google.com/file/d/1revQ4mcTGKkUwyvQCteimUWrffa2AyW-/view?usp=sharing

Weavebird: http://www.giuliopiacentino.com/weaverbird/.

2. Tutoring example

In this tutoring example, we will create bio-inspired free-form periodic structures using a set of GH plugins.

2.1. Introduction: L-System [1]

An L-System is a parallel string rewriting system. A string rewriting system consists of an initial string, called the seed, and a set of rules for specifying how the symbols in a string are rewritten as (replaced by) strings. Let's have a look at a simple L-System:

Seed: A

rules:

Rule #1: A = AB

Rule #2: B = BA

The L-System starts with the seed 'A' and iteratively rewrites that string using the production rules. On each iteration a new string/word is derived.

n is the derivation length = the number of iterations

n=0: A

n=1: AB (A becomes AB according to Rule #1)

n=2: ABBA (A becomes AB according to Rule #1, while B becomes BA according to Rule #2. In result we get ABBA)

n=3: ABBABAAB

n=4: ABBABAABBAABABBA

…

Turtle Graphics are often used for L-System interpretation

The following symbols drive the Turtle:

F *move forward at distance* ***L*** *(Step Length) and draw a line*

f *moves forward at distance* ***L*** *(Step Length) without drawing a line*

+ turn left ***A*** *(Default Angle) degrees*

–turn right ***A*** *(Default Angle) degrees*

\ roll left ***A*** *(Default Angle) degrees*

/ roll right ***A*** *(Default Angle) degrees*

^ pitch up ***A*** *(Default Angle) degrees*

& *pitch down* ***A*** *(Default Angle) degrees*

| turn around 180 degrees

J *insert point at this position*

*"multiply current length by d****L*** *(Length Scale)*

! *multiply current thickness by d****T*** *(Thickness Scale)*

[*start a branch (push turtle state)*

] *end a branch (pop turtle state)*

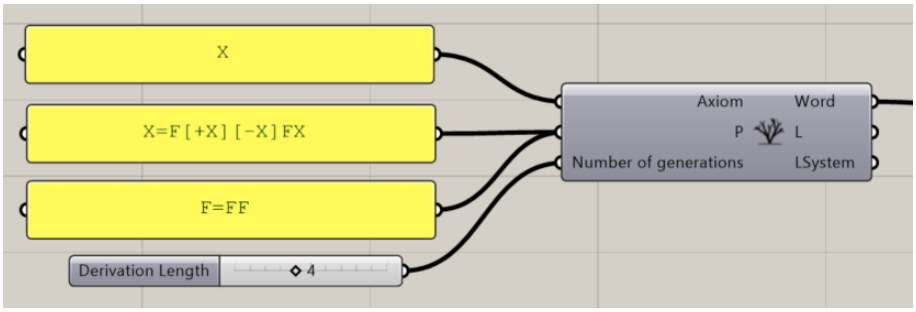
A/B/C/D… placeholders, used to nest other symbols

More details can be found in [1].

2.2. Implement: L-System in Rabbit

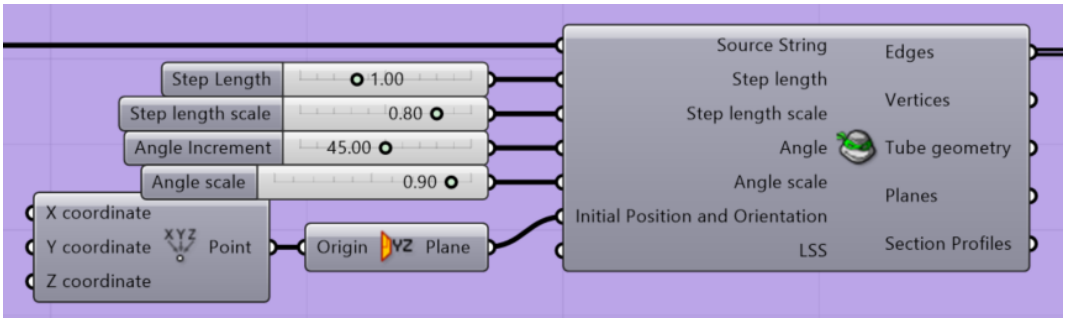
Step 1: Axiom and production rule creations

Drag **LSystem** component to the GH interface from Rabbit plugin. Set **Axiom** as X using **Panel** component. For production rules, two basic rules are used: $X = F[+X][-X]FX, F = FF$. The number of generations is set as 4.



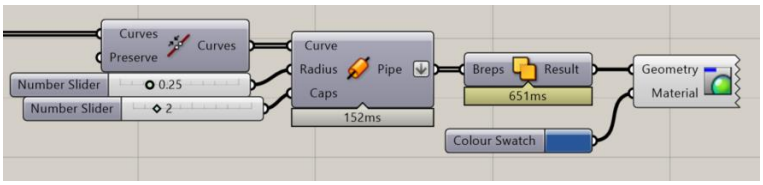
Step 2: 2D L-System skeleton generation

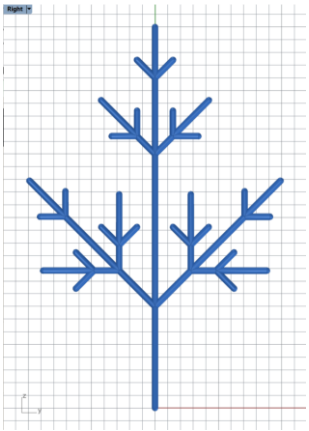
Use **Turtle** component to connect with **LSystem** component in Step 1. **Step length**, **step length scale**, **angle**, and **angle scale** are set as 1.00, 0.80, 45.00, and 0.90, respectively. The initial position and orientation is the original point at **YZ plane**.

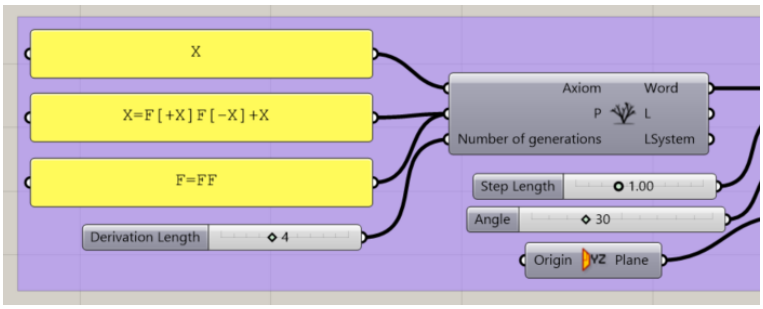


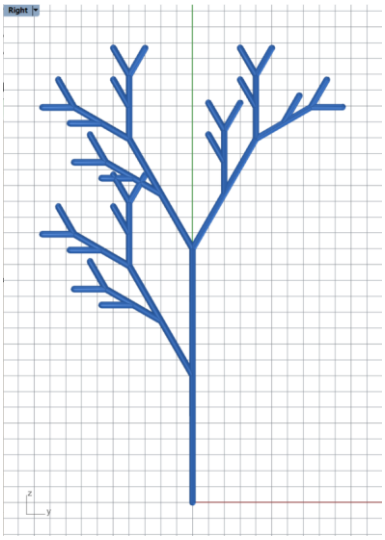
Step 3: 2D L-System pipe creation

Use **Join Curves** component to join as many curves as possible. Then, we use **Pipe** component to generate round pipe. At last, **Solid Union** component is applied to generate solid geometry.



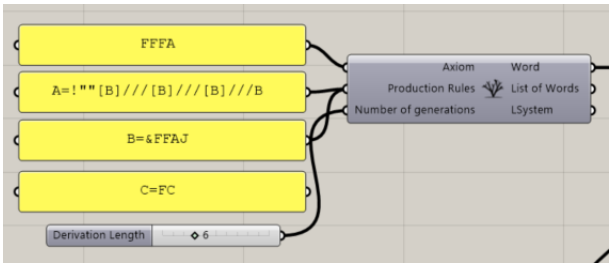


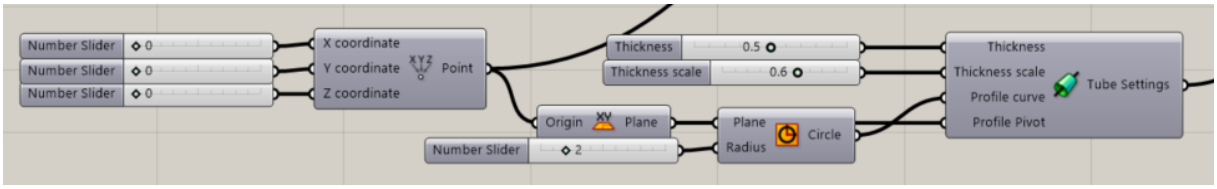


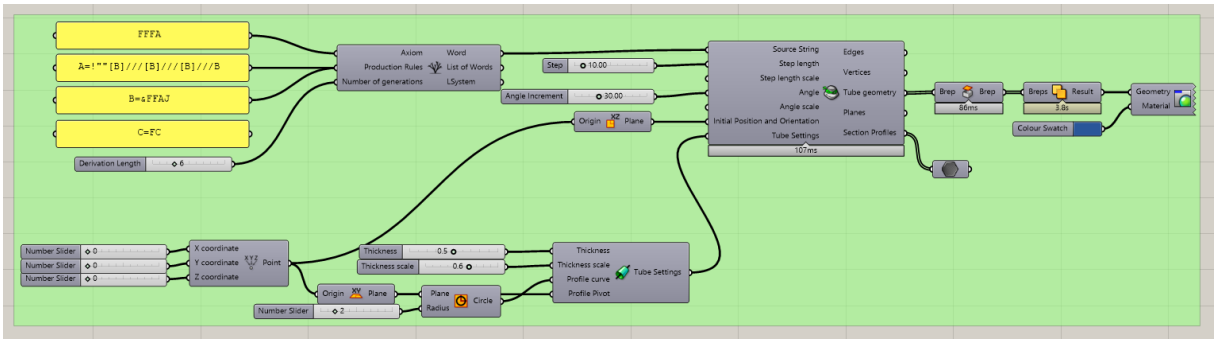


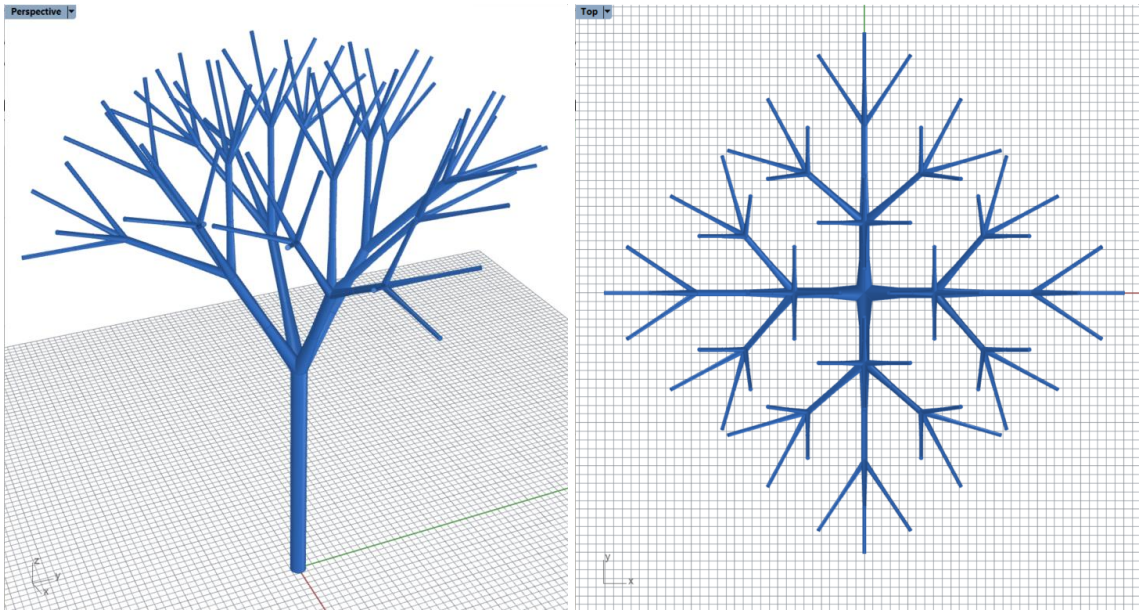
Step 4: 3D L-System tube generation

Use **Circle** component to make a circle on the XY plane with an original point (0,0,0). Drag **Tube settings** from Rabbit plugin to connect with the circle. The **Tube geometries** are capped as closed geometries using **Cap Holes** components.



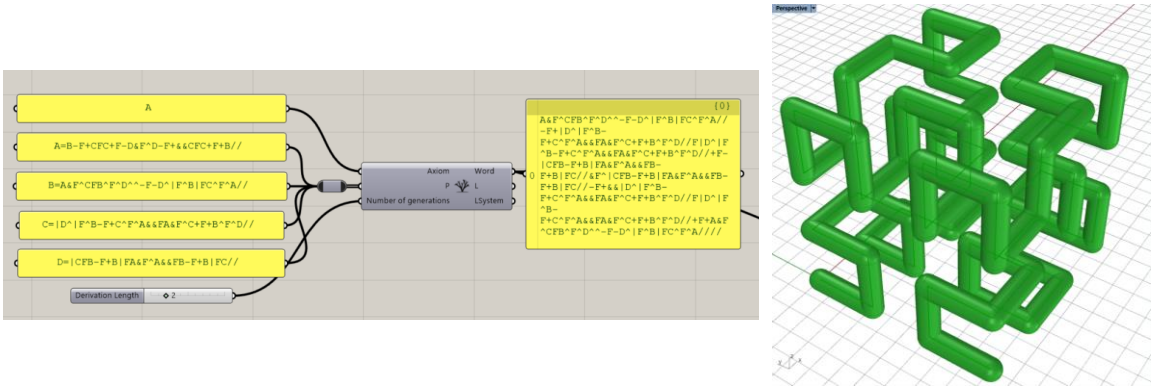






Step 5: Hilbert curve generation (optional)

Production rule of Hilbert curve:

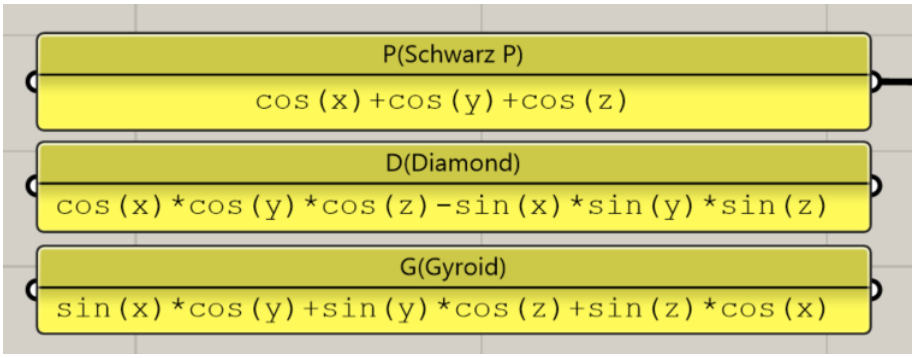


2.3. TPSM lattice structure creation

Triply periodic minimal surfaces (TPMS) are a non-intersecting 3D surface characterized by a zero value of mean curvature at each point. TPMS are frequently considered to generate regular structures with crystalline symmetry. TPMS-based materials can be defined with a simple mathematical function.

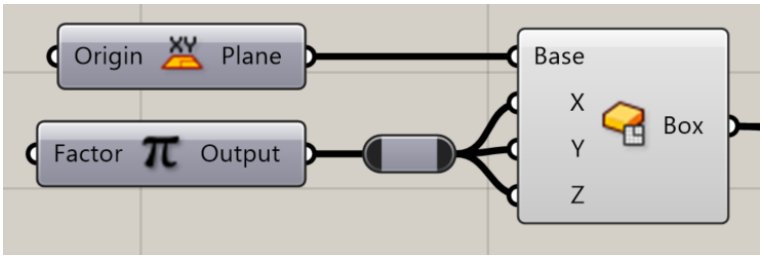
Step 1: Surface equation definition

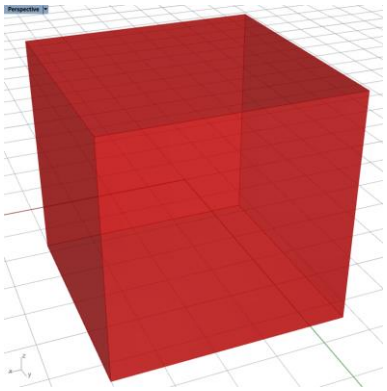
Use **Panel** component to define three common TPMS equations, Schwarz P, Diamond and Gyroid surfaces.



Step 2: Design domain definition

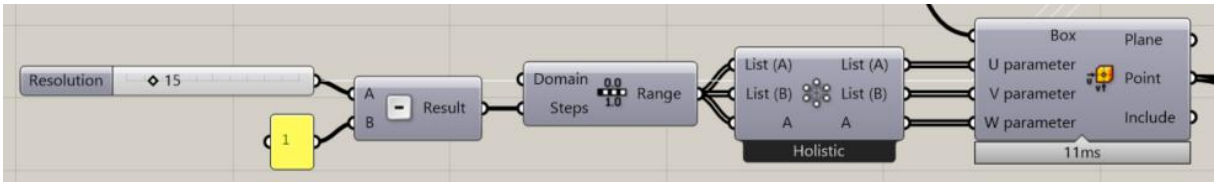
Use **Centre Box** component to generate design domain. The base is **XY plane** and the size of X, Y, and Z is 2*pi.

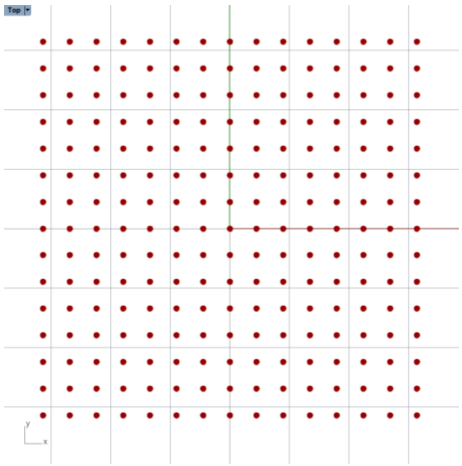


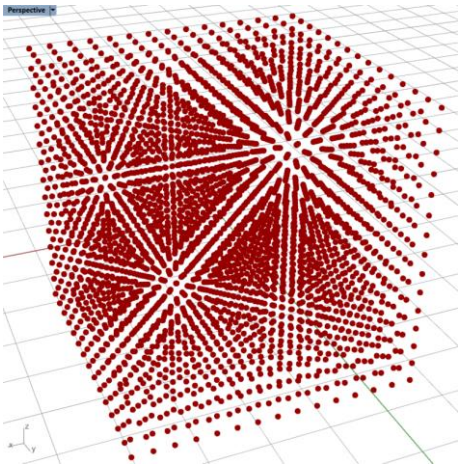


Step 3: Parametric point generation

Use **Evaluate Box** component to create parametric points in the design domain. The resolution is defined as 15.

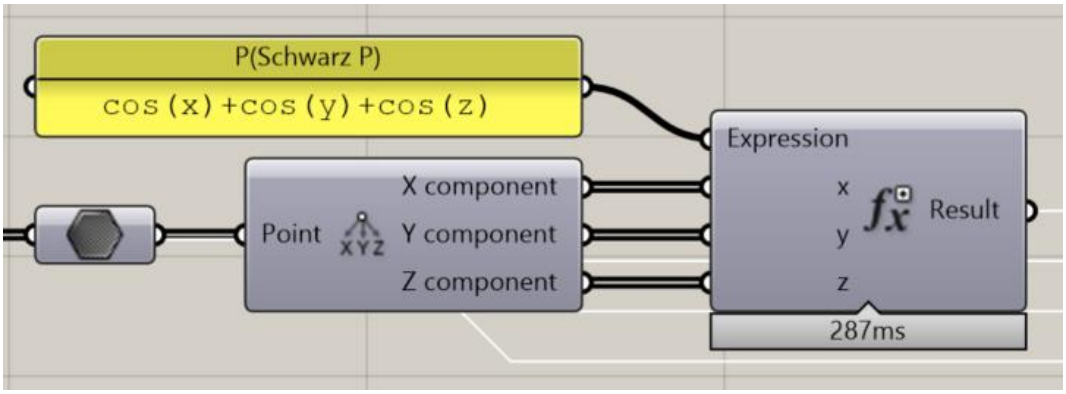






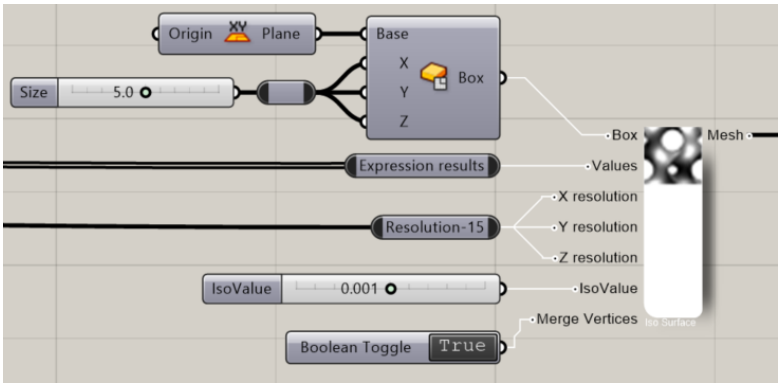
Step 4: Evaluate the expression results for all points

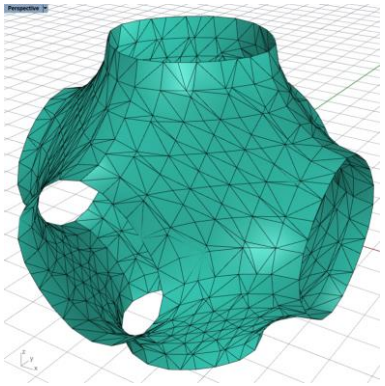
Use **Deconstruct** component to deconstruct the defined points as X, Y, and Z values. Calculate the value of each point using the Schwarz P mathematical function via **Evaluate** component.



Step 5: Iso Mesh generation

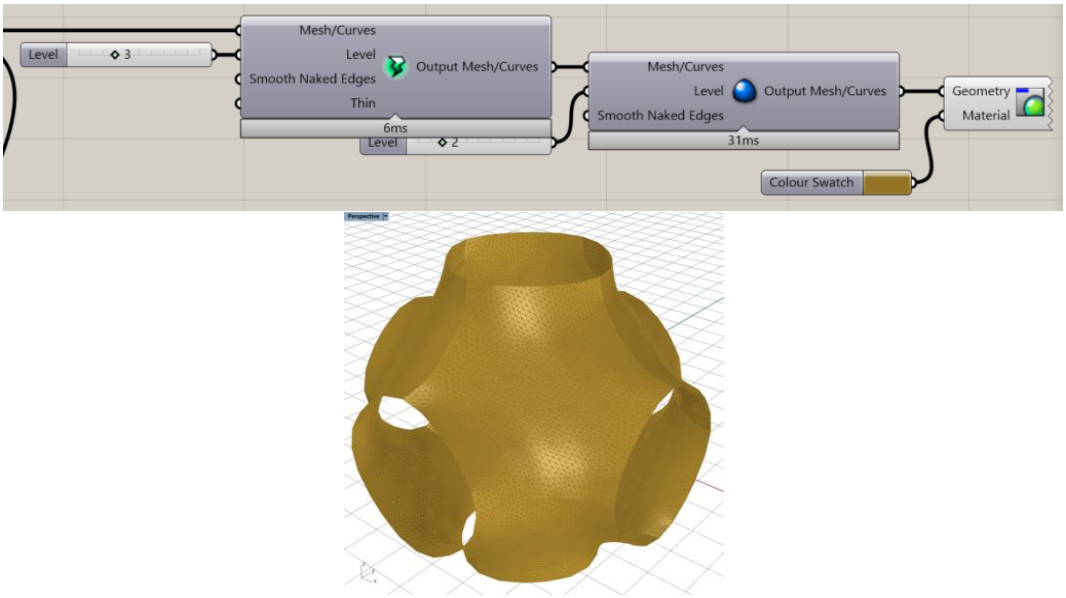
Use **Iso Surface** component to generate 0-iso-surface mesh in **Millipede** plugin. Create a design box using **Center Box** component.



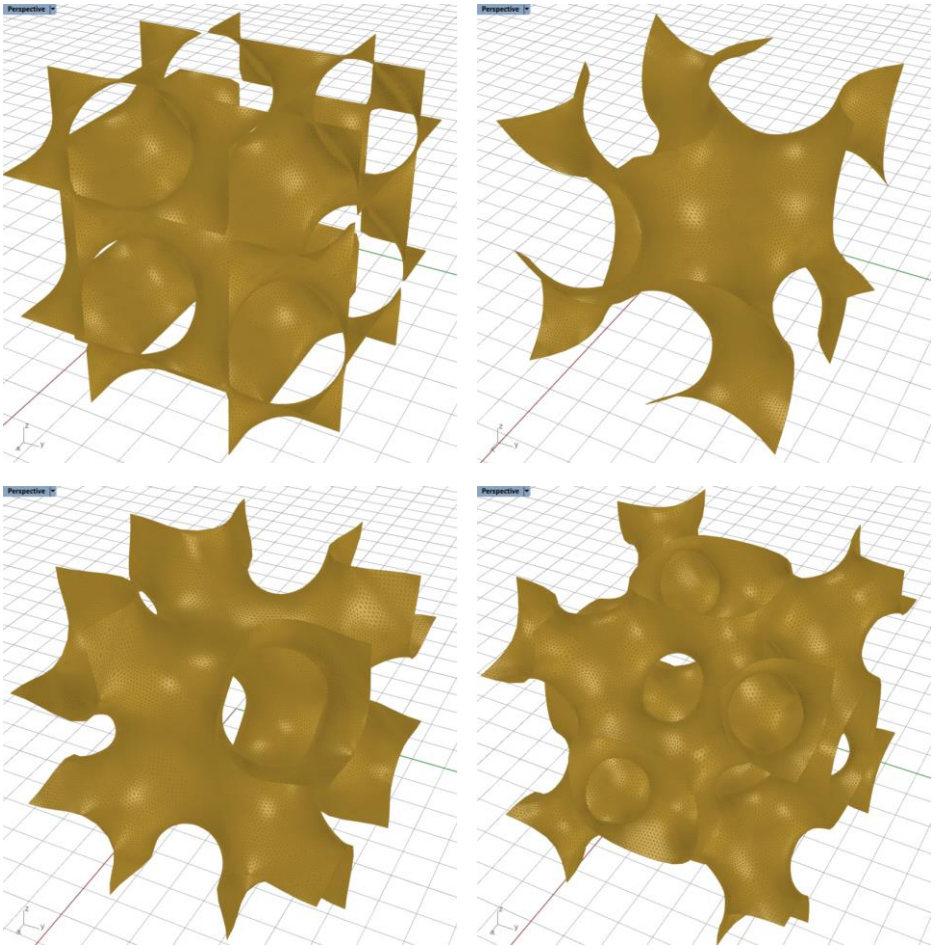


Step 5: Smooth mesh

Use **Weaverbird's Laplacian Smoothing** component to calculate a smoothened representation of the original mesh. It will not increase the number of mesh. Then, use **Weaverbird's Loop Subdivision** component to have a new mesh.



Step 6: Other TPMS structures



2.4.: Use **Crystallon** and TPMS to design lattice structures.

3. Reference

[1] https://morphocode.com/intro-to-l-systems/