Course Introduction + Design With Parameters

DM II Week 1

DES 212| Winter Term 2 | 2026 | 3 Credits
Instructor: Xun Liu
GTAs: Lea Papillon, Wylie Moorman

Content

Course Introduction

Technological Evolution in Architecture

Concepts and Design Methods

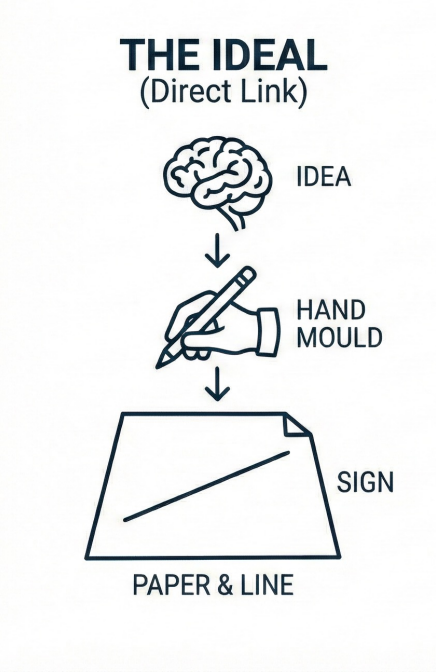
The Tool: Grasshopper- What and Why

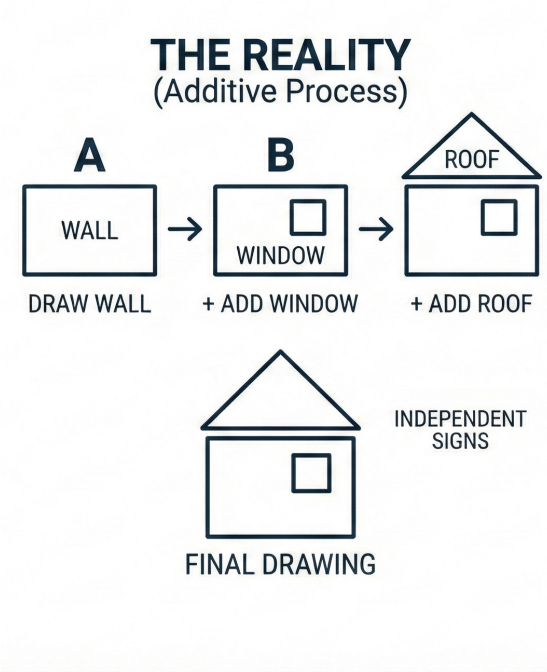
Course Introduction

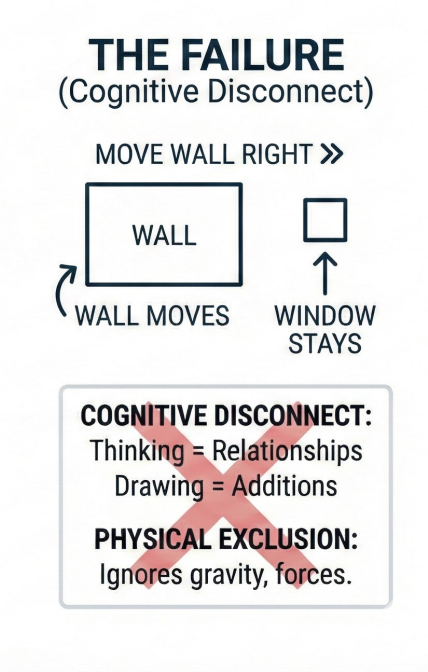
“Architects do not make buildings; they make drawings of buildings.”

-- Robin Evans

The Architects’ Medium







THE LIMITS OF ADDITIVE DRAWING LOGIC

Image generated by Gemini

The Architects’ Medium

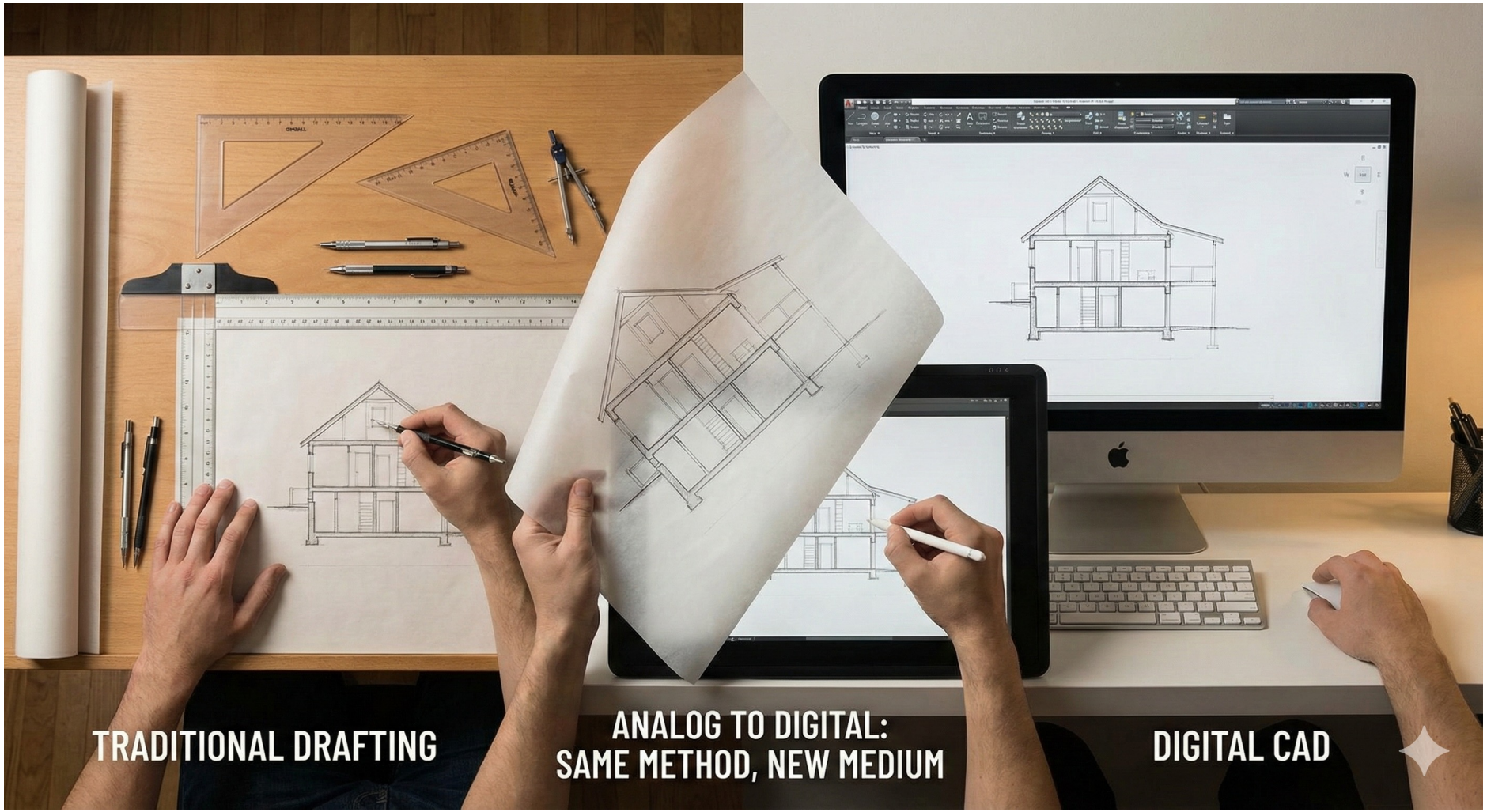


Image generated by Gemini

The Architects’ Medium

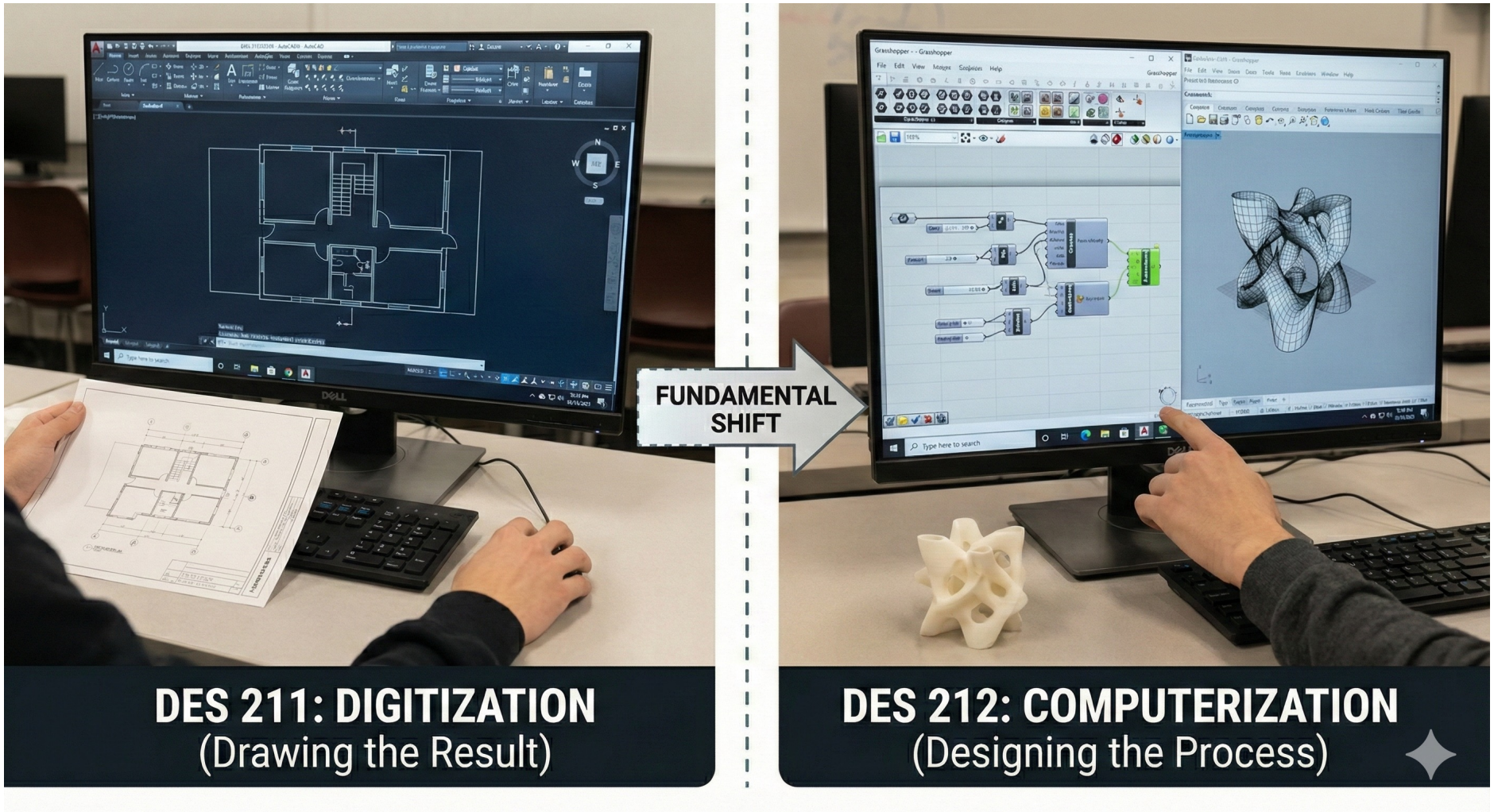


Image generated by Gemini

Course Introduction

Learning Objectives

Systemic Thinking

Transition from drawing static objects to defining generative rules.

Algorithmic Logic

Employ data hierarchies and conditional structures as drivers.

Co-creative Intelligence

Navigate collaboration between humans and AI.

Material Agency

Bridge the digital-physical divide (gravity, tolerance).

Performative Justification

Use environmental simulation to validate design decisions.

Representational Genealogy

Articulate the computational lineage of a design.

Course Structure: The Three Modules

Module 1

The Logic of Form

Module 2

The Intelligence of Matter

Module 3

Advanced Form-Finding

Course Structure: The Three Modules

	WEDNESDAY	FRIDAY
Week 1	**Lecture x.1**	**Async Tutorial x.1** *canvas submission on the following Friday*
Week 2	**Lecture x.2**	**Async Tutorial x.2** *canvas submission on the following Friday*
Week 3	**Lecture x.3**	**Async Tutorial x.3** *canvas submission on the following Friday*
Week 4	**WORK SESSION**	**Pin-ups** *WIP Miro pin-up one day before this Friday* *final canvas submission on this Friday*

Course Structure: The Three Modules

MODULE I: The Logic of Form

1	7 Jan	* **L1.1: COURSE OVERVIEW + DESIGN WITH PARAMETERS** *Algorithmic thinking history and theory* *The shift from digitization to computerization*	9 Jan	**T 1.1: HELLO, GRASSHOPPER** *Grasshopper interface, navigation, and component anatomy* Simple demos of creating geometries in GH *canvas submission due 1.16*	
2	14 Jan	* **L 1.2: RULE-BASED DESIGN** *Top-down design strategies* *Computational geometries and rule-based form generation*	16 Jan	**T 1.2: GH: ATTRACTORS** *2D patterns using attractors and image sampler* *Basic of data trees: graft vs flatten* *canvas submission due 1.23*	
3	21 Jan	* **L 1.3: EMERGENT DESIGN** *Bottom-up design strategies* *Field conditions and the nature of code*	23 Jan	**T 1.3: GH: FIELDS** *Vector Field, Transform*, Animation Deep into data trees *canvas submission due 1.30*	DDS Opening Workshop *Time TBD* *Attendance Optional*
4	28 Jan	**WORK SESSION I**	30 Jan	* **PIN-UP I** *Miro pin-up is due at 11:59 PM the day before the review.* *canvas submission due 1.30*	

Course Structure: The Three Modules

MODULE II: The Intelligence of Matter

5	4 Feb	* **L 2.1: AI-ASSISTED DESIGN** *Prompt engineering as design syntax* *Image-to-Geometry workflows*	6 Feb	**T 2.1: AI FOR VISUALIZATION** *Text-to-Image generation* *Image-to-Mesh generation* *canvas submission due 2.13*	
6	11 Feb	* **L 2.2: DIGITAL FABRICATION** *Mass customization, file-to-factory and material agency* *Subtractive (CNC) vs. additive (3D print)*	13 Feb	**T 2.2: 3D PRINT** *Mesh, Nurbs, SubD* *Mesh repair and slicing* *canvas submission due 3.6*	* **DDS LAB VISIT** Group A
7	18 Feb	**READING WEEK**	20 Feb	**READING WEEK**	
8	25 Feb	* **L 2.3: ROBOTIC INTELLIGENCE** *Robotic fabrication of architecture* *Human-machine collaboration*	27 Feb	**T 2.3: AI FOR CODING** *GH scripting components* *AI as co-pilot* *canvas submission due 3.6*	* **DDS LAB VISIT** Group B
9	4 Mar	**WORK SESSION II**	6 Mar	* **PIN-UP II** *Miro pin-up is due at 11:59 PM the day before the review.* *canvas submission due 3.6*	

Course Structure: The Three Modules

MODULE III: Advanced Form-Finding

10	11 Mar	* **L 3.1: DATA-DRIVEN DESIGN** *Form follows data (datascape)* *The Informational Context, and mapping the invisible*	13 Mar	**T 3.1: SPATIAL DATA INJECTION** *Navigating the package manager and Food4Rhino* *Working with site data and non-spatial data* *canvas submission due 3.20*
11	18 Mar	* **L 3.2: PERFORMANCE-DRIVEN DESIGN** *Form follows performance* *Simulation as generative: structural and environmental*	20 Mar	**T 3.2: ENVIRONMENTAL SIMULATION** *Ladybug basics* *canvas submission due 3.27*
12	25 Mar	* **L 3.3: EVOLUTIONARY DESIGN** *Theory of evolutionary architecture* *The iterative process and optimization strategies*	27 Mar	**T 3.3: EVOLUTIONARY SOLVERS** *Galapagos basics* *canvas submission due 4.3*
13	1 Apr	**WORK SESSION III**	3 Apr	**NO CLASS**
14	8 Apr	* **PIN-UP III** *Miro pin-up is due at 11:59 PM the day before the review.* *canvas submission due 4.8*		

Course Structure: The Three Modules

Assignment 1: 20%

Assignment 2: 20%

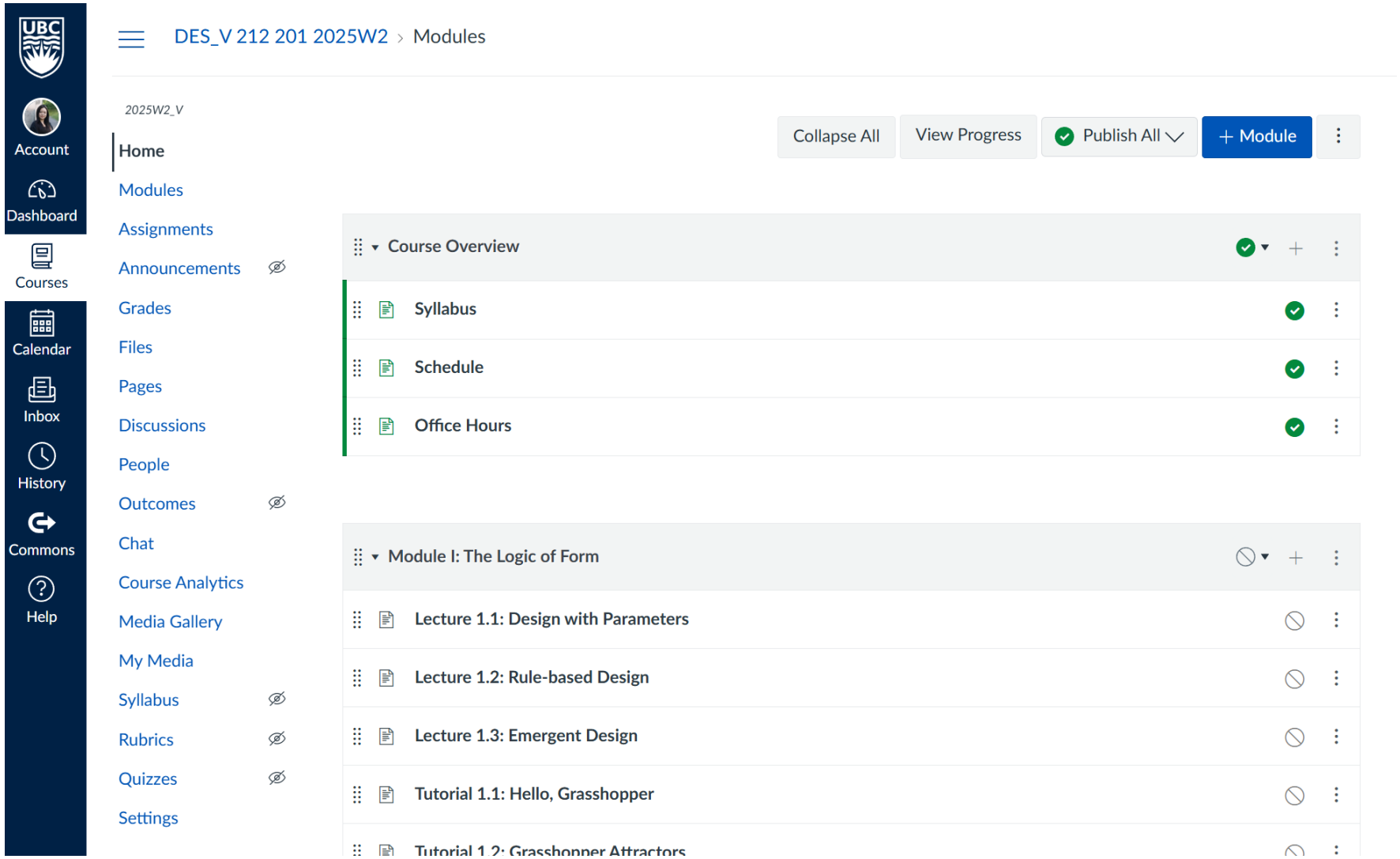
Assignment 3: 25%

Exercises (9 total): 27% (3% each)

Participation: 8%

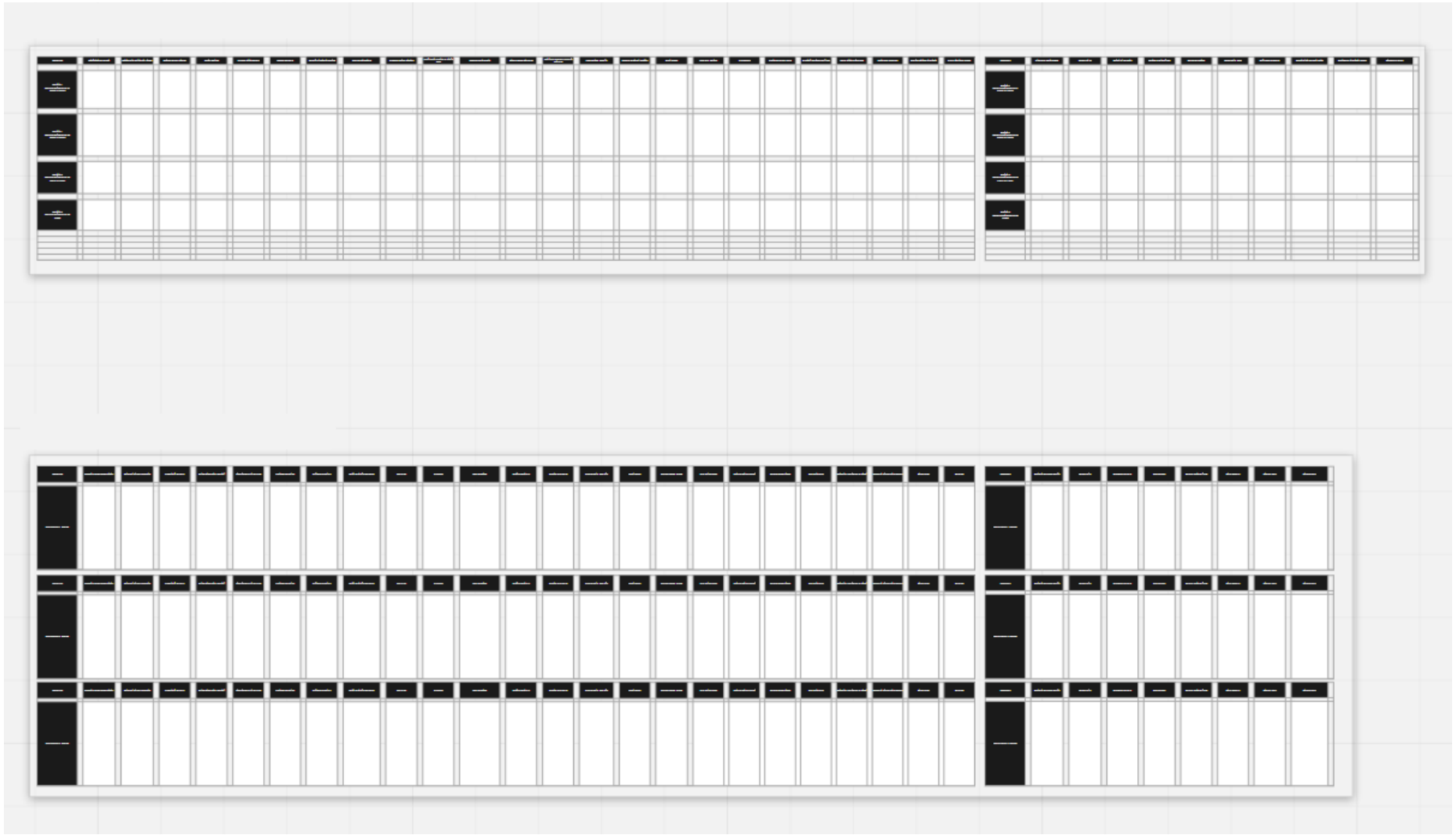
The Learning Ecosystem

Canvas



The Learning Ecosystem

Miro



Course Introduction

The Learning Ecosystem

Tools:

Rhino 8 and Grasshopper: Latest version is required.

Adobe: Photoshop and Illustrator.

AI Tools: Cursors. Free access will be provided to Nano Banana and other online platforms.

Grasshopper Plugins: Ladybug Tools and others as announced in class.

Digital Fabrication: Access to 3D printers, CNC machines and robotic arm.

Digital Design Studio (DDS) Visits: Scheduled visits to the DDS are required for training and fabrication labs.

Technological Evolution in Architecture

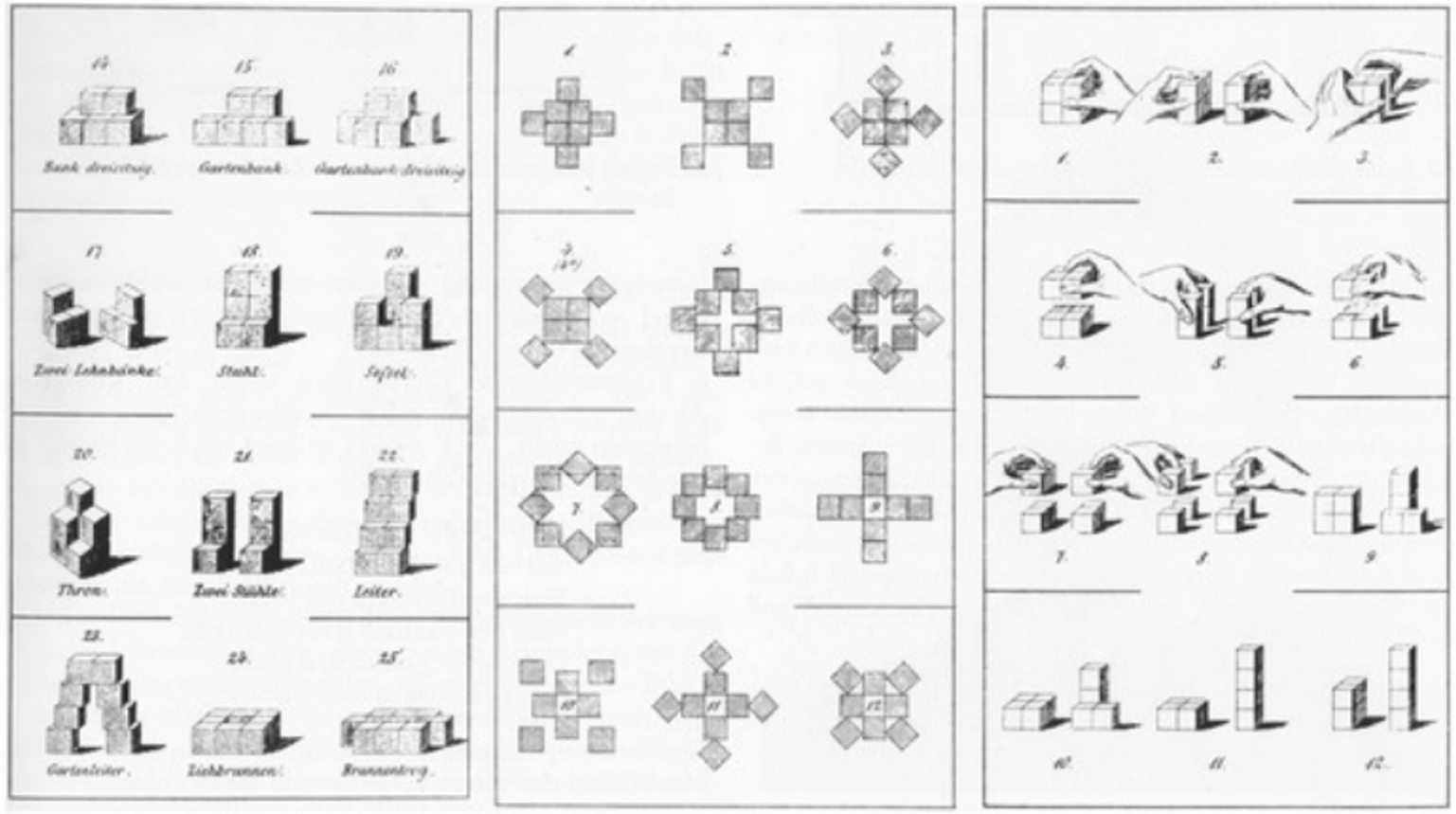
Before Computer: Modularity

Construction Kit

Before code, we had the grid. Architects sought to escape bespoke craft by designing systems of standardized assembly.

‘Baukasten’

Walter Gropious, 1923



Before Computer: Modularity

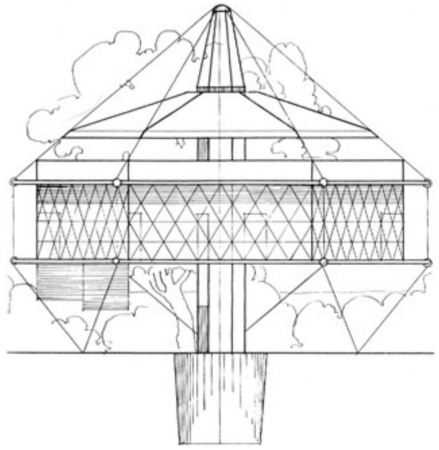
Object-Oriented Design

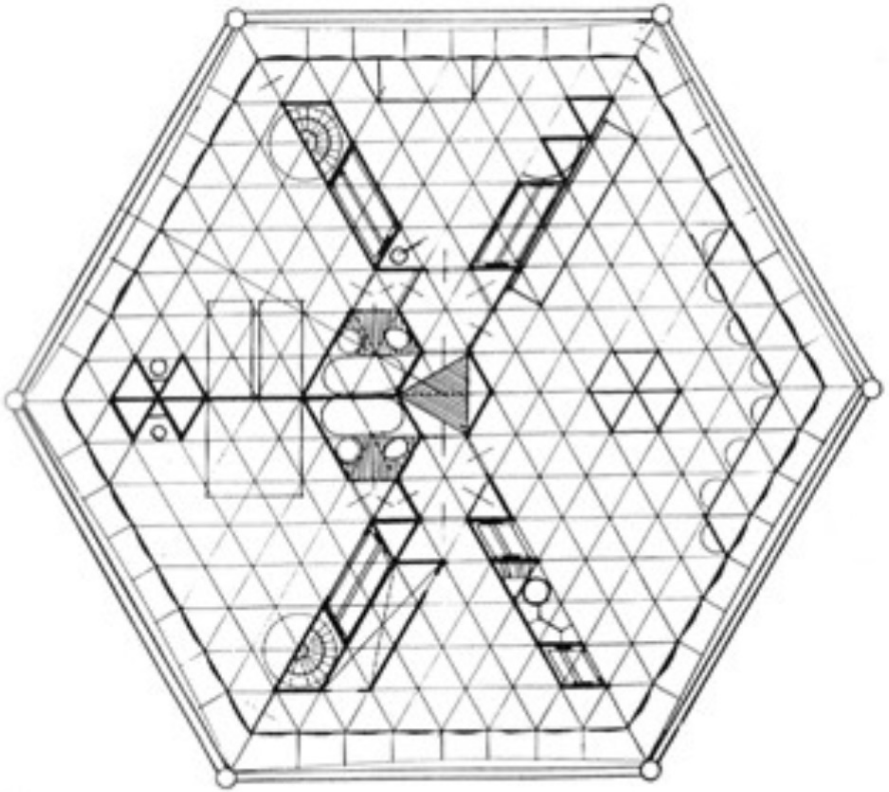
Embedded mechanical systems directly into the module. The house became a machine for living.

Dymaxion House

Buckminster Fuller, 1930







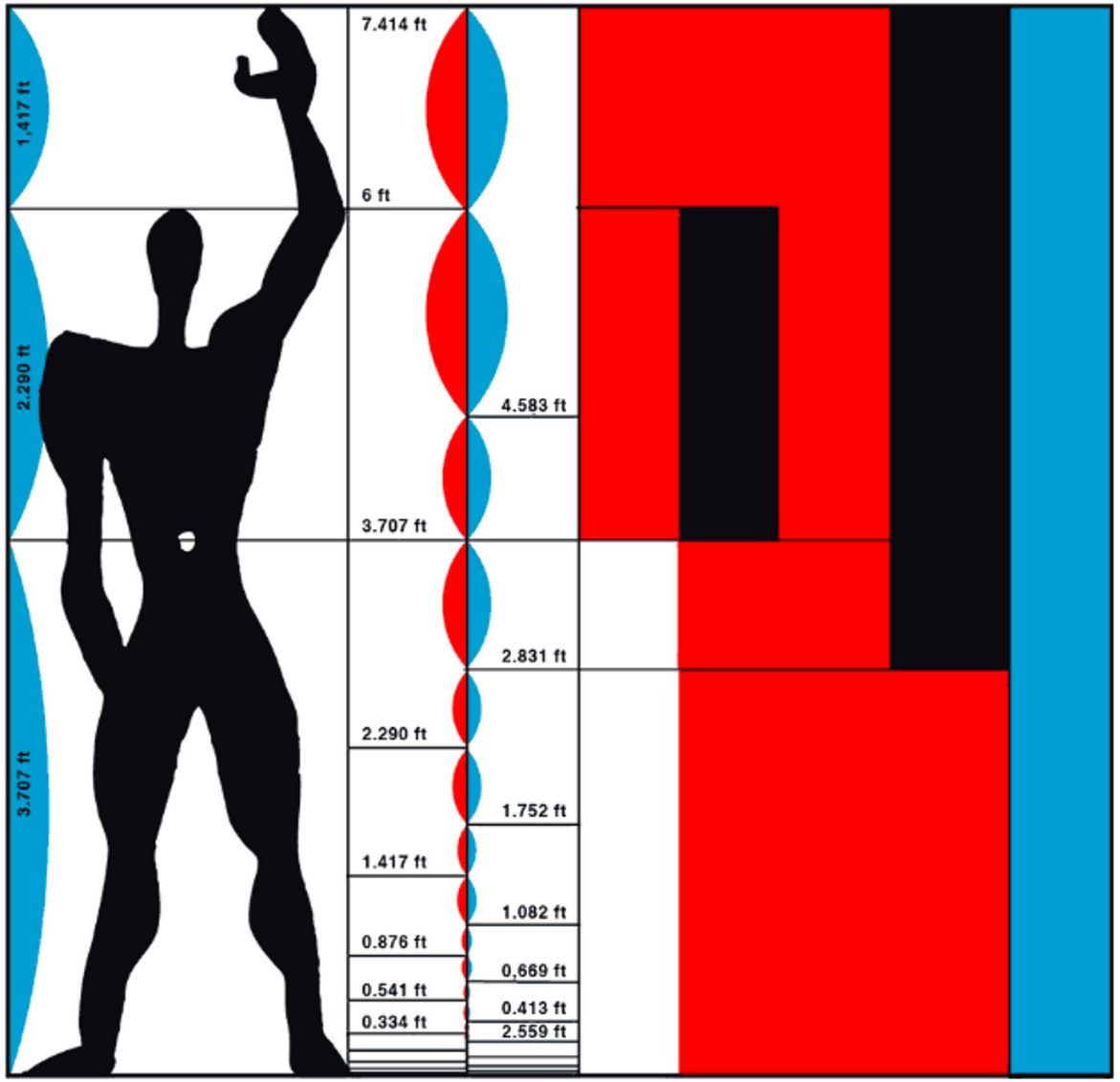
Before Computer: Modularity

Universal Code

An analog algorithm for dimensioning, regulating the built environment through human scale.

The Modulor

Le Corbusier, 1946

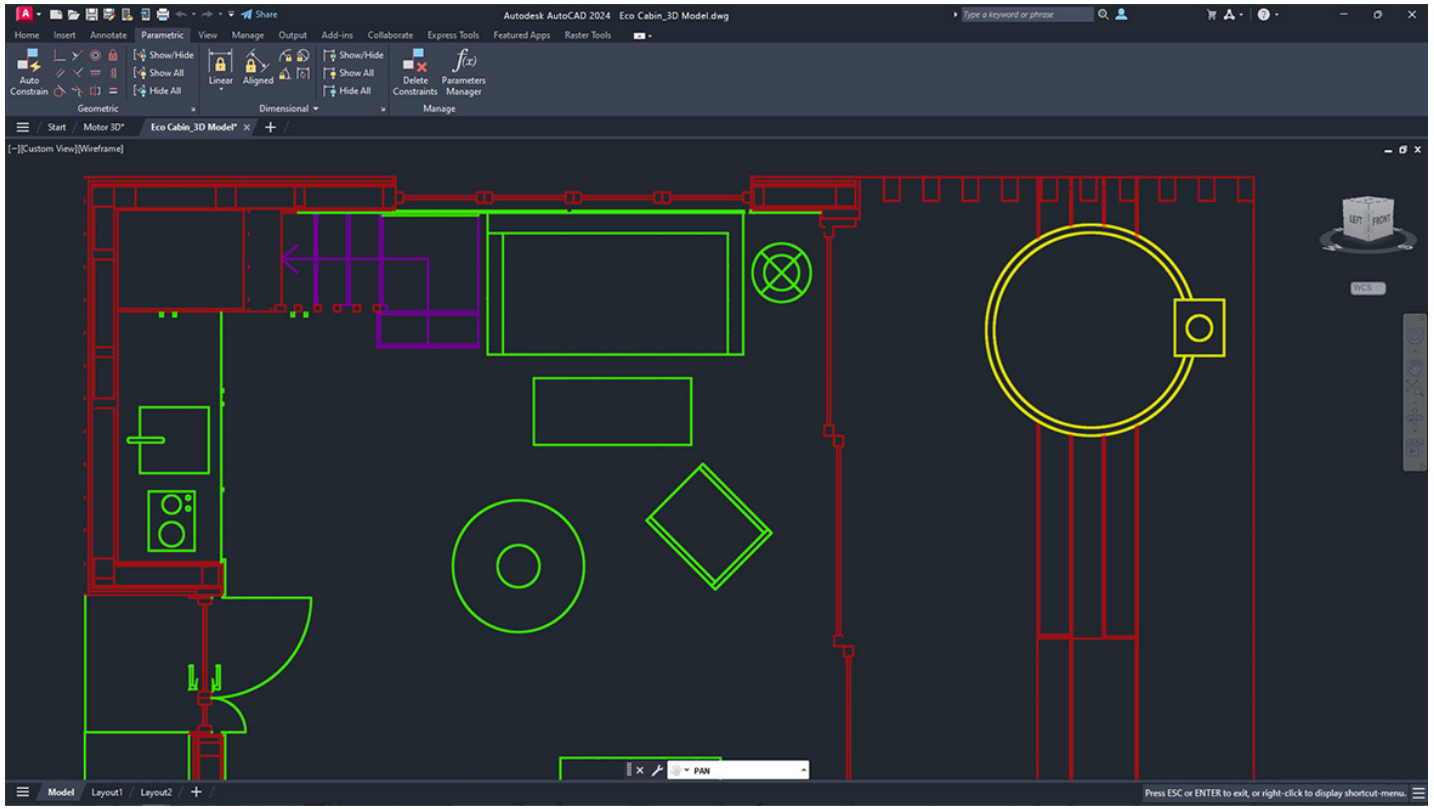


Digitization

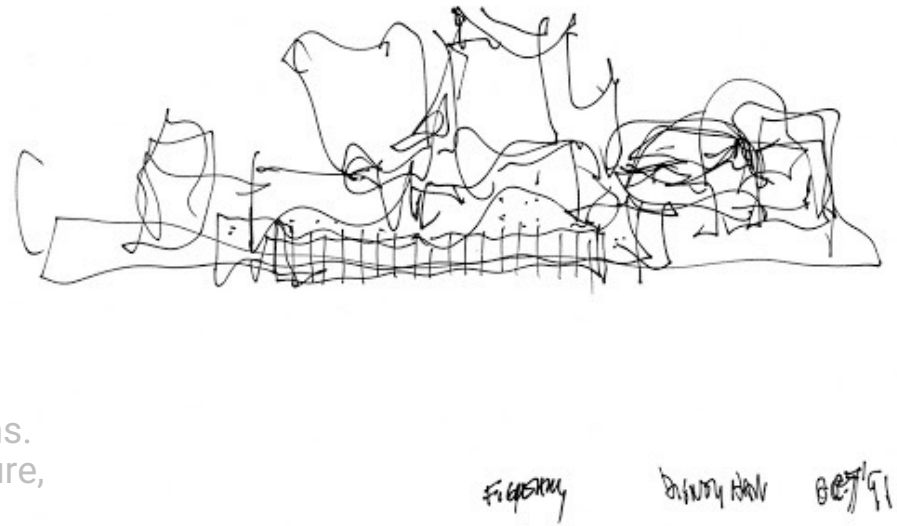
The Explicit Era

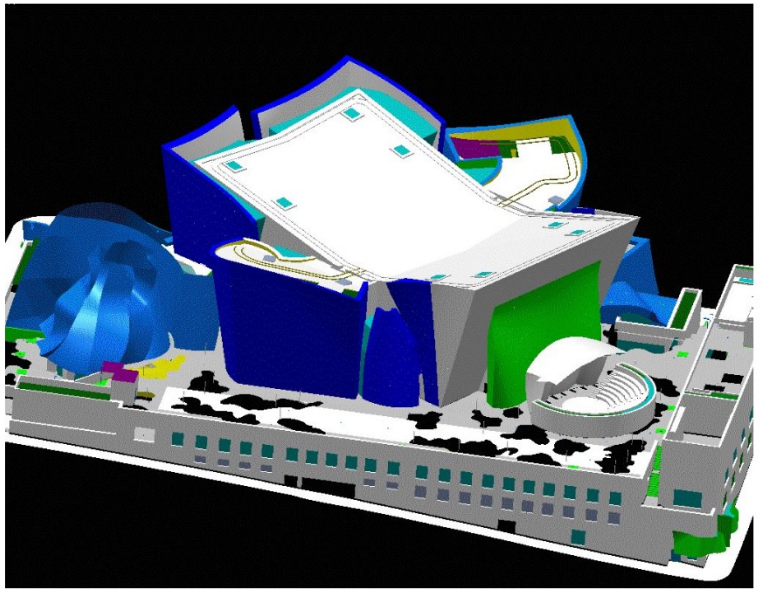
Moving from the paper grid to the screen. Traditional CAD was an additive process—digitalizing the drafting board.

CAD



Digitization





Rationalization

Moving Using aerospace software (CATIA) to rationalize complex forms. The computer described the sculpture, it did not generate it.



Walt Disney Concert Hall

Frank Gehry, 2003

Technological Evolution in Architecture

Digitization

Atomic Constraint

Allowing designers to define relationships

Sketchpad

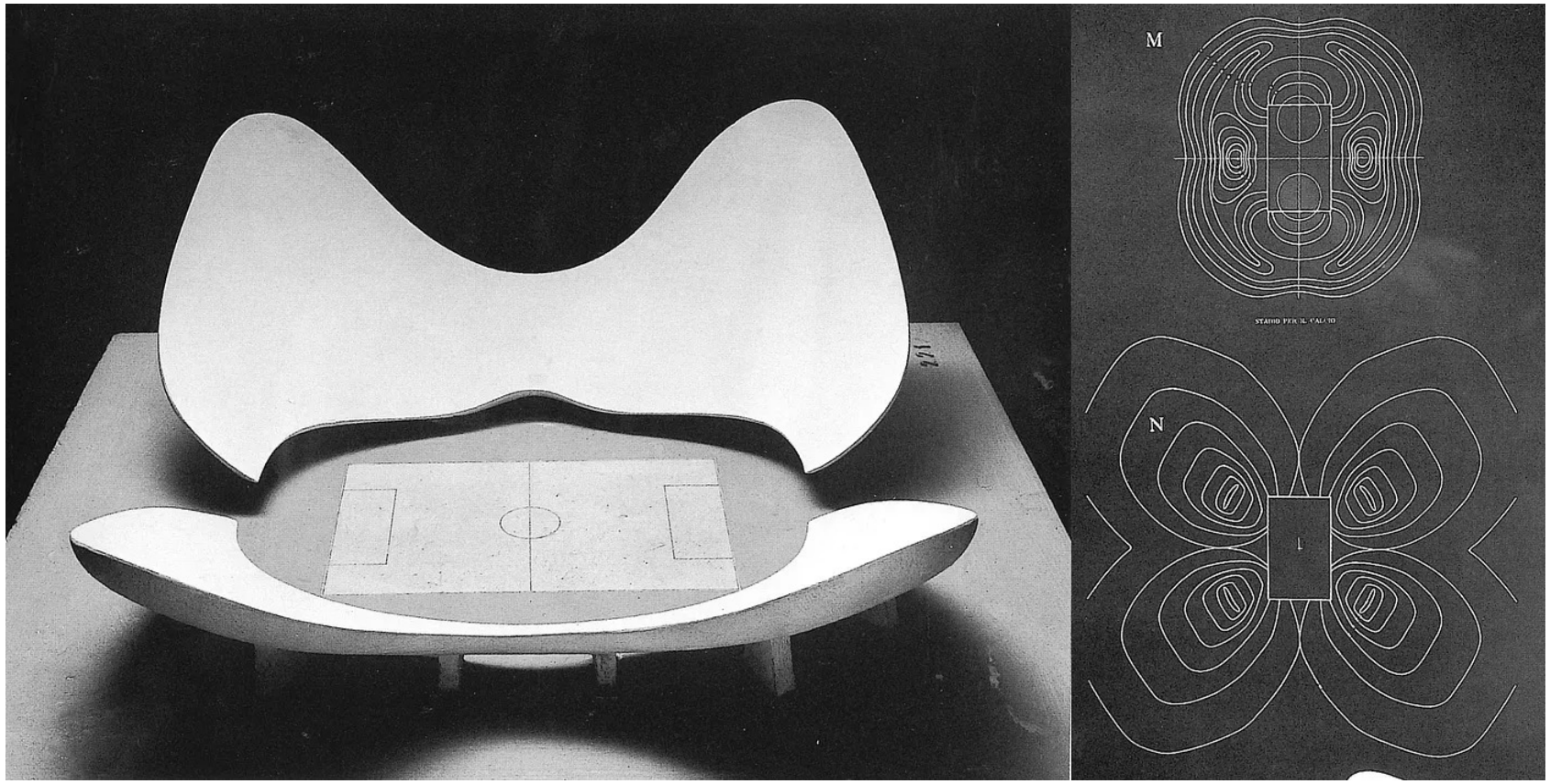
Ivan Sutherland, 1963



Computation

The Implicit Era

The shift from drawing geometry to scripting logic. Defining the equation that generates the wall.



Stadium N

Luigi Moretti, 1960

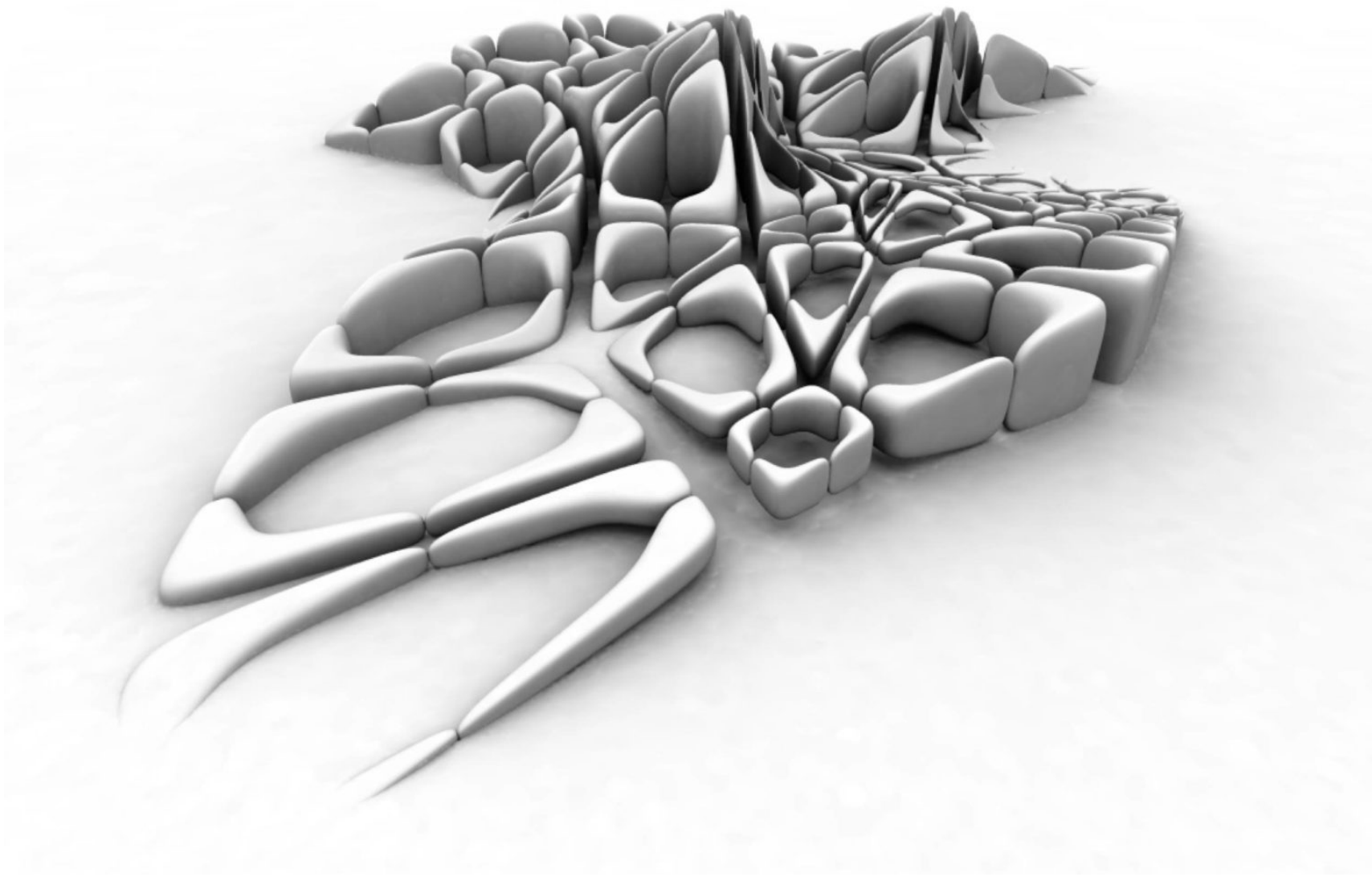
Computation

Parametricism

Kartal-Pendik Masterplan

Zaha Hadid Archiects,

Patrik Schumacher



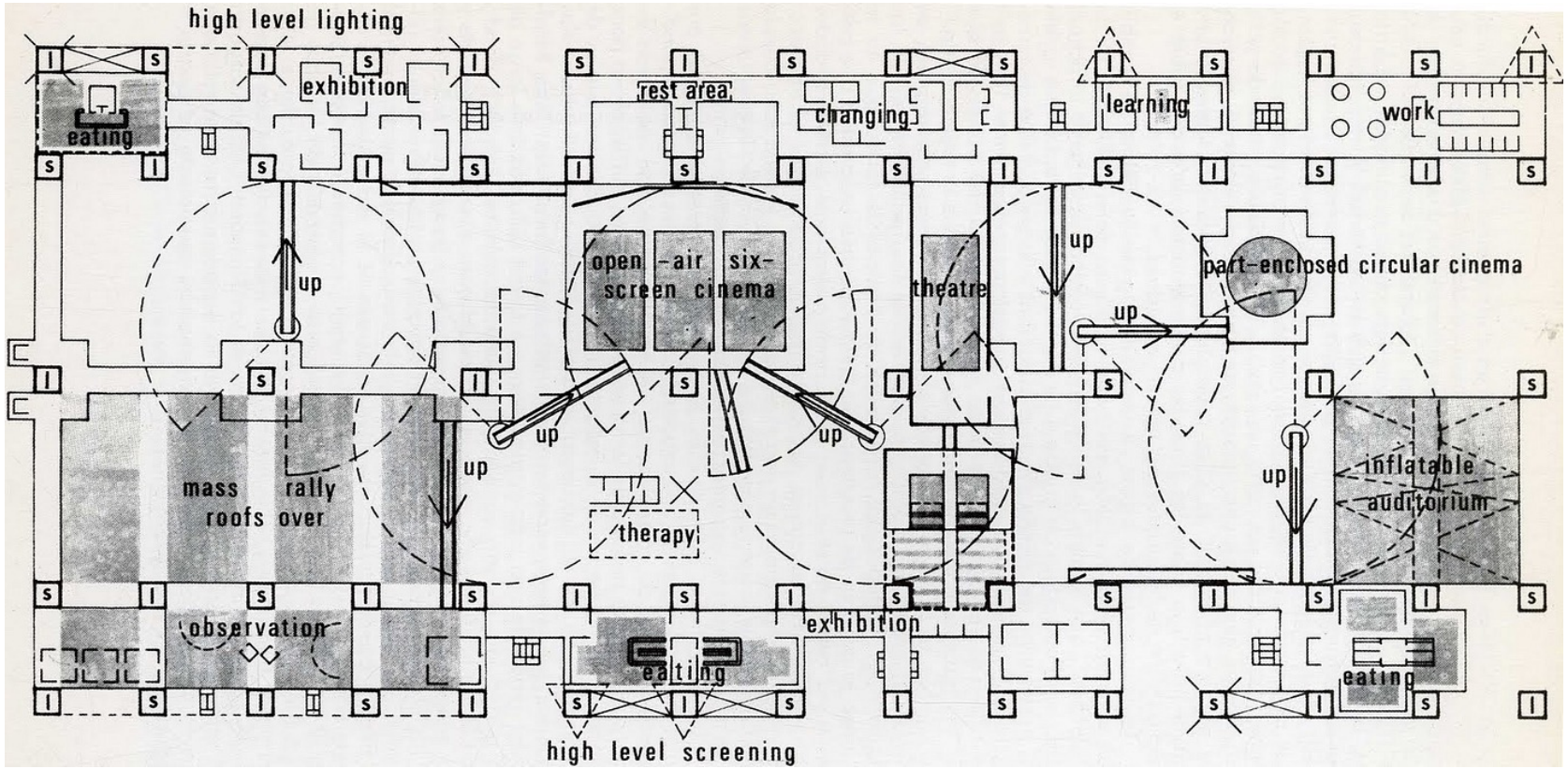
Artificial Intelligence

Cybernetic

A building is not a static object, but an "intelligent agent.” Architecture should be "unstable, indeterminate, and unknown in advance" to allow for self-organization.

Fun Palace

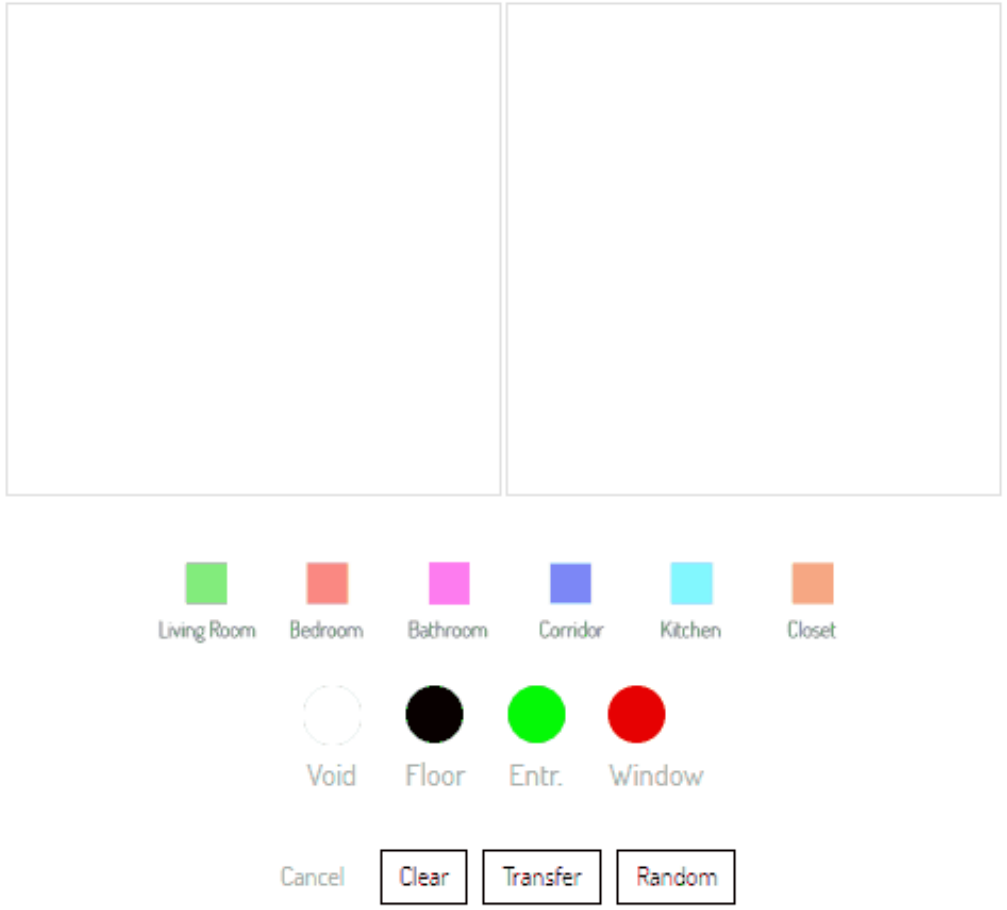
Cedric Price, 1964



Artificial Intelligence

Hallucination

From optimization to hallucination.
The machine becomes an intelligent agent capable of suggestion.



ArchiGan

Stanislas Chaillou, 2019

Concepts and Design Methods

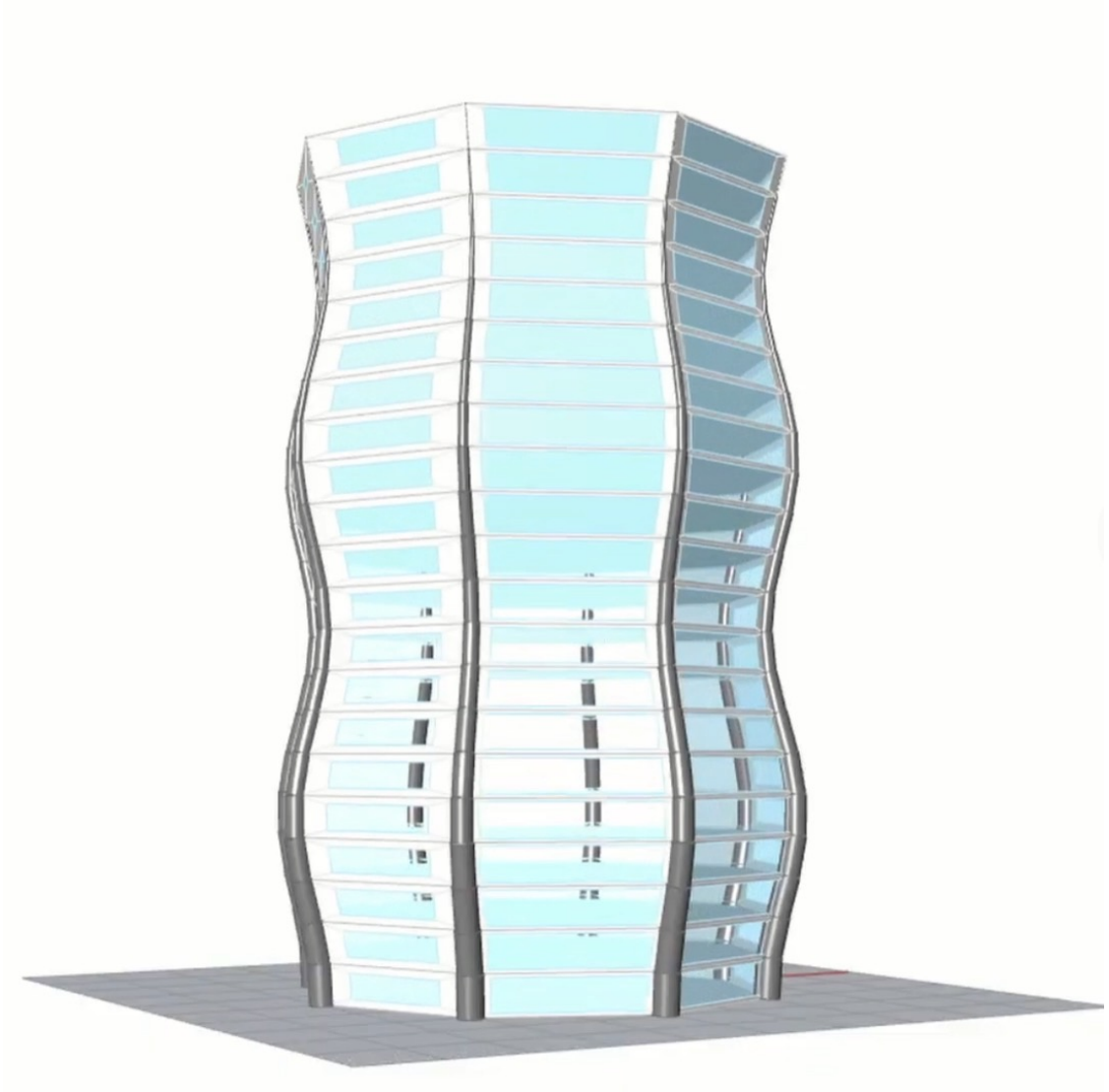
Module 1: The Logic of Form

- Parametric
- Rule-Based Design
- Emergent Design

1: The Logic of Form

Parametric Design

Continuous Differentiation



Concepts and Design Methods

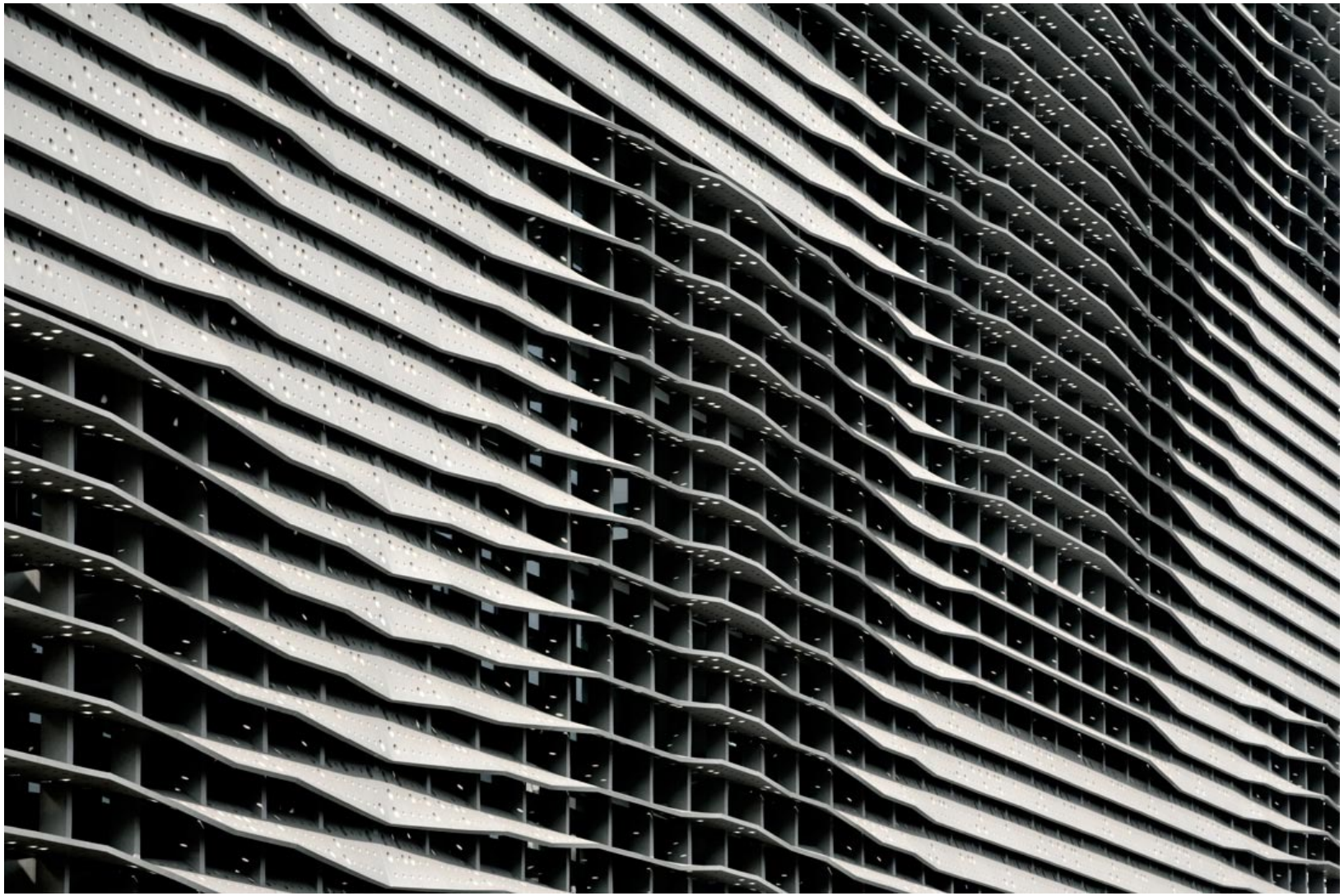
1: The Logic of Form

Parametric Design

Continuous Differentiation

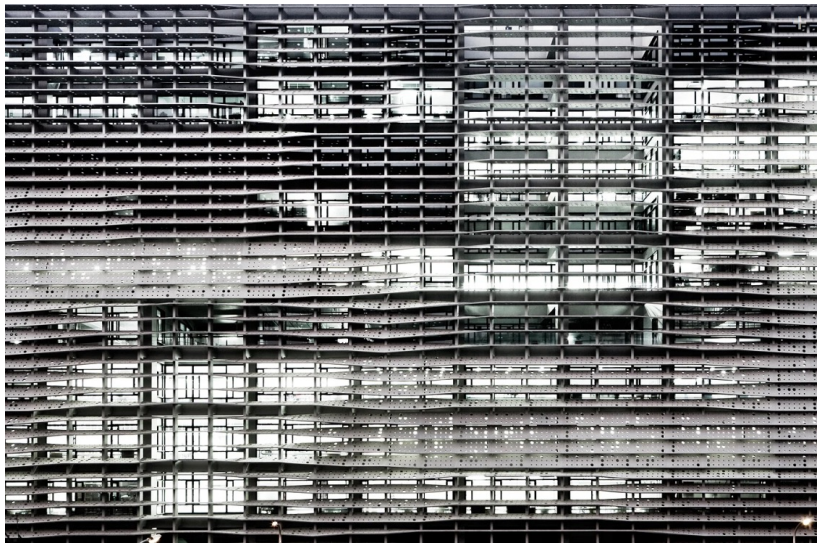
Lafayette 148 Global Headquarters

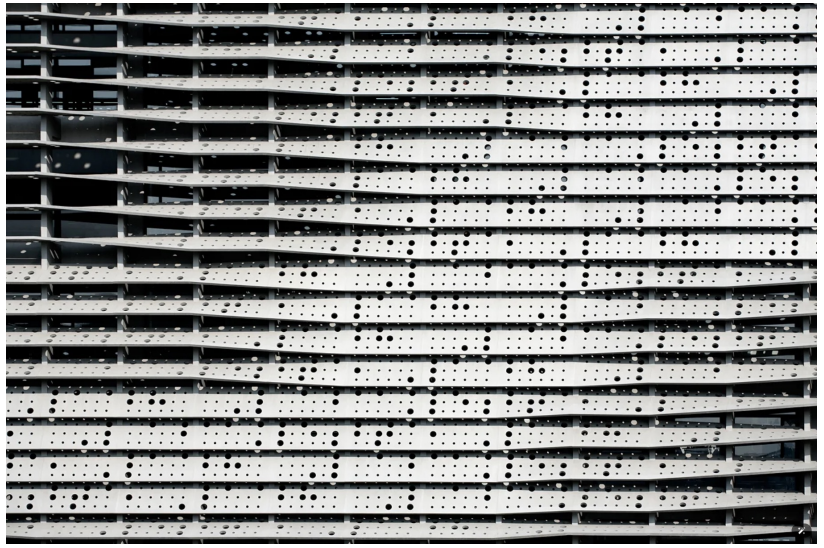
Studio Gang



Concepts and Design Methods

1: The Logic of Form



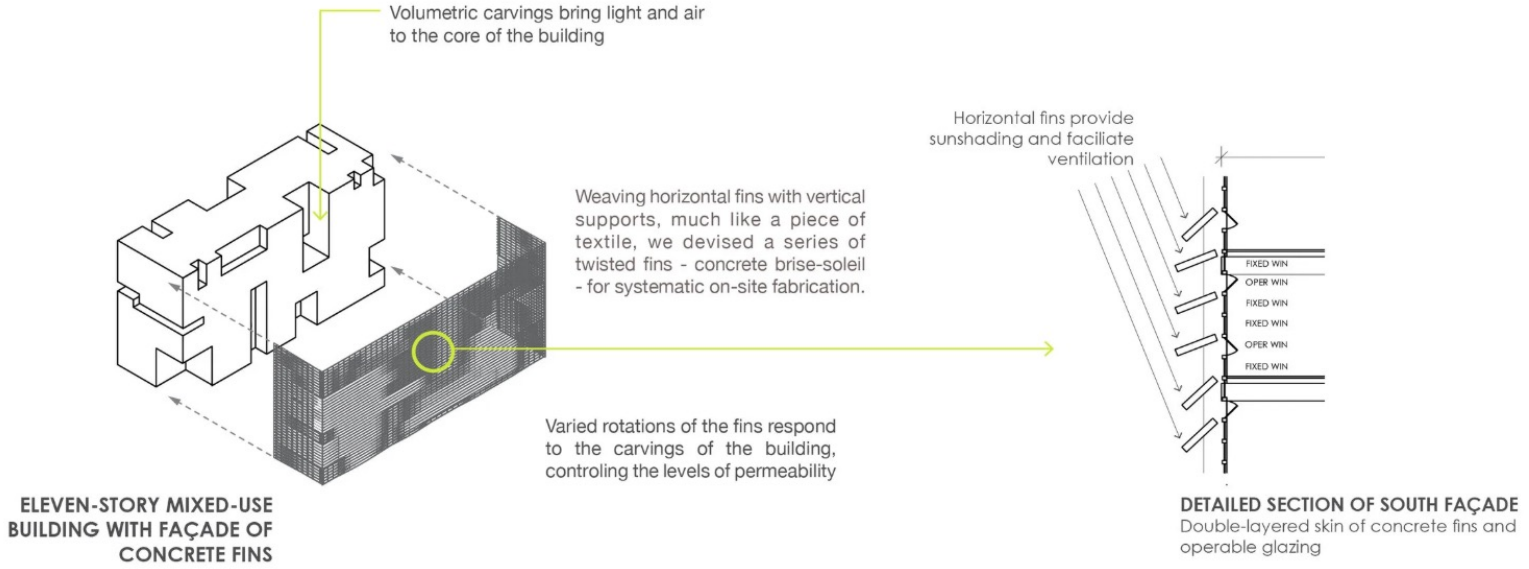


Parametric Design

Continuous Differentiation

Lafayette 148 Global Headquarters

Tsz Yan Ng

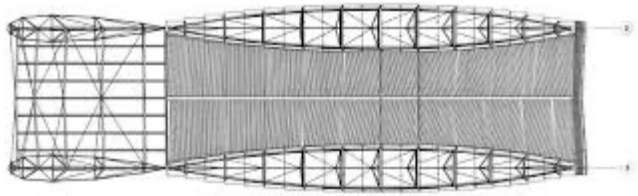


1: The Logic of Form

Parametric Design

Structural Logic







Waterloo International Terminal

Grimshaw

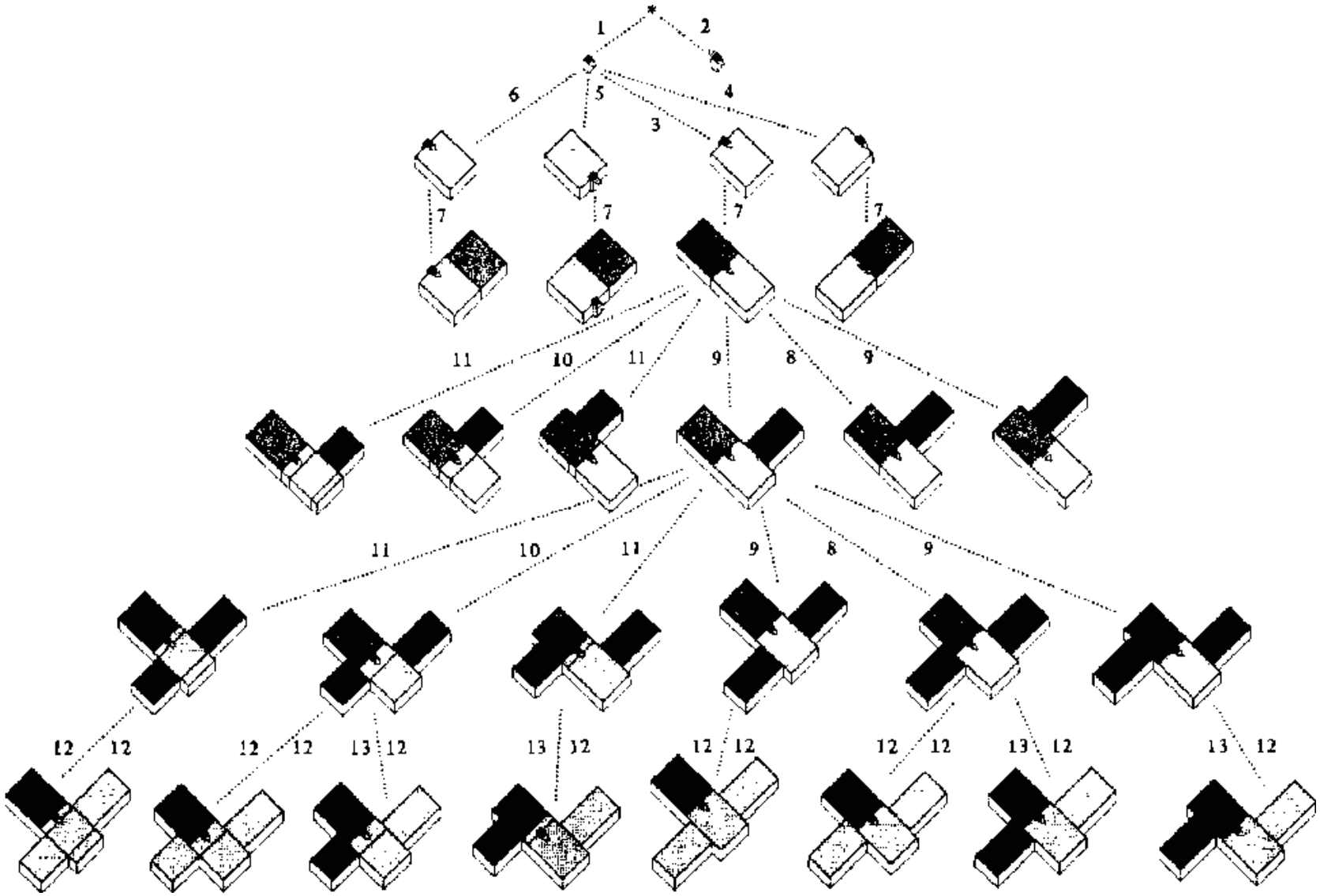
1: The Logic of Form

Rule-based Design

Imposing Order

Shape Gramma

George Stiny and James Gips



Concepts and Design Methods

1: The Logic of Form

Rule-based Design

Imposing Order

Habitat 67

Moshe Safdie



Concepts and Design Methods

1: The Logic of Form

Rule-based Design

Top-down Strategy

Chi She

Archi Union



Concepts and Design Methods

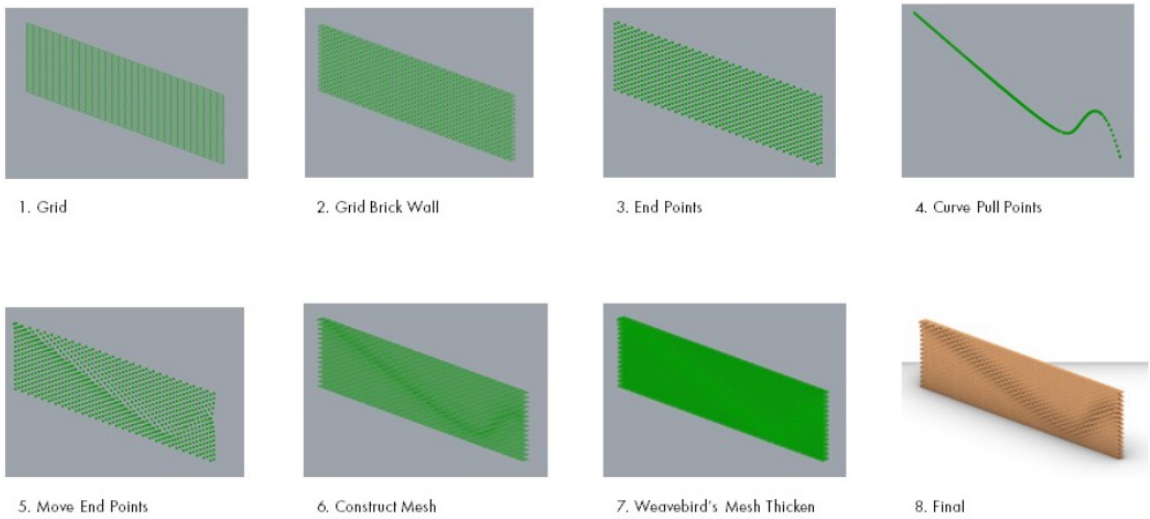
1: The Logic of Form

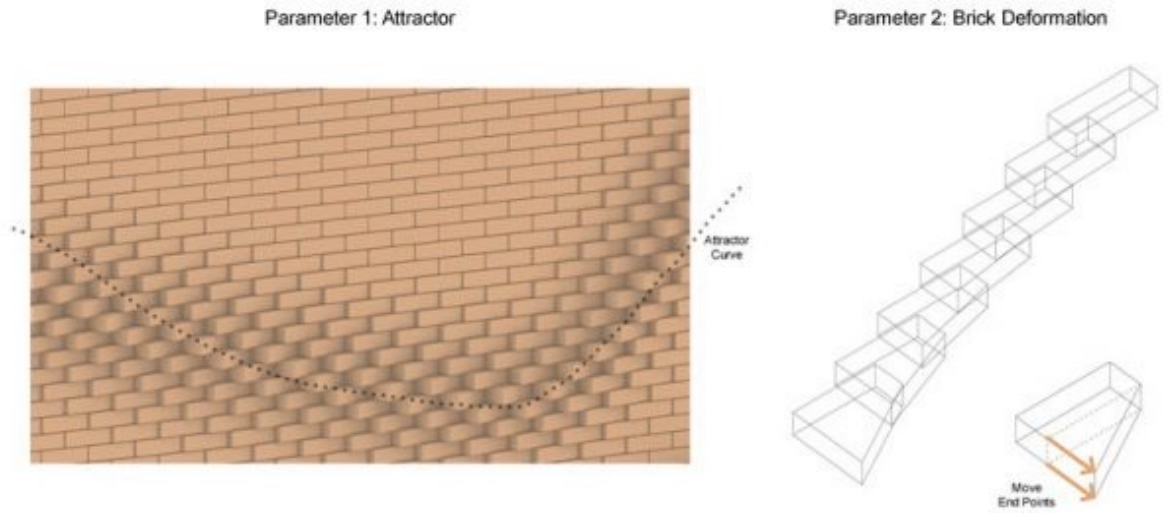
Rule-based Design

Top-down Strategy

Administrative Building Textilverband

Behet Bondzio Lin Architekten





1: The Logic of Form

Emergent Design

Bottom-Up Strategy

Fractal Patterns in Nature



1: The Logic of Form

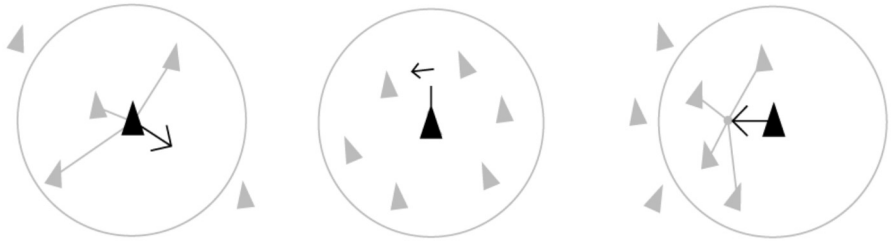
Emergent Design

Bottom-Up Strategy

Swarm Intelligence



1: The Logic of Form



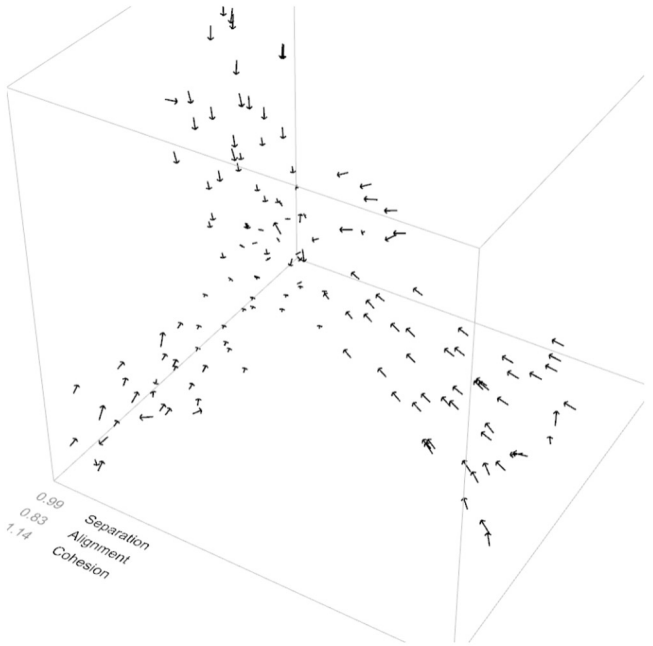
Separation:
Steer to avoid crowding local flockmates

Alignment:
Steer toward the average heading of local flockmates

Cohesion:
Steer to move toward the average position of local flockmates

Emergent Design

Bottom-Up Strategy





Swarm Intelligence

Craig Reynold

Concepts and Design Methods

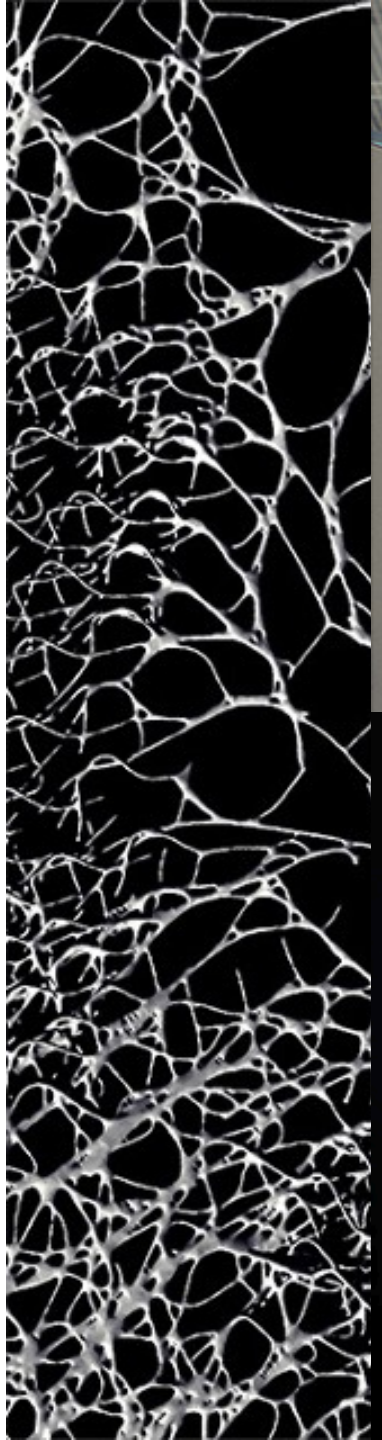
1: The Logic of Form

Emergent Design

Bottom-Up Strategy

Swarm Intelligence

Roland Snooks







Module 2: The Intelligence of Matter

- AI-Assisted Design
- Digital Fabrication
- Robotic Intelligence

2: The Intelligence of Matter

AI-Assisted Design

Visualiztion

Nano Banana

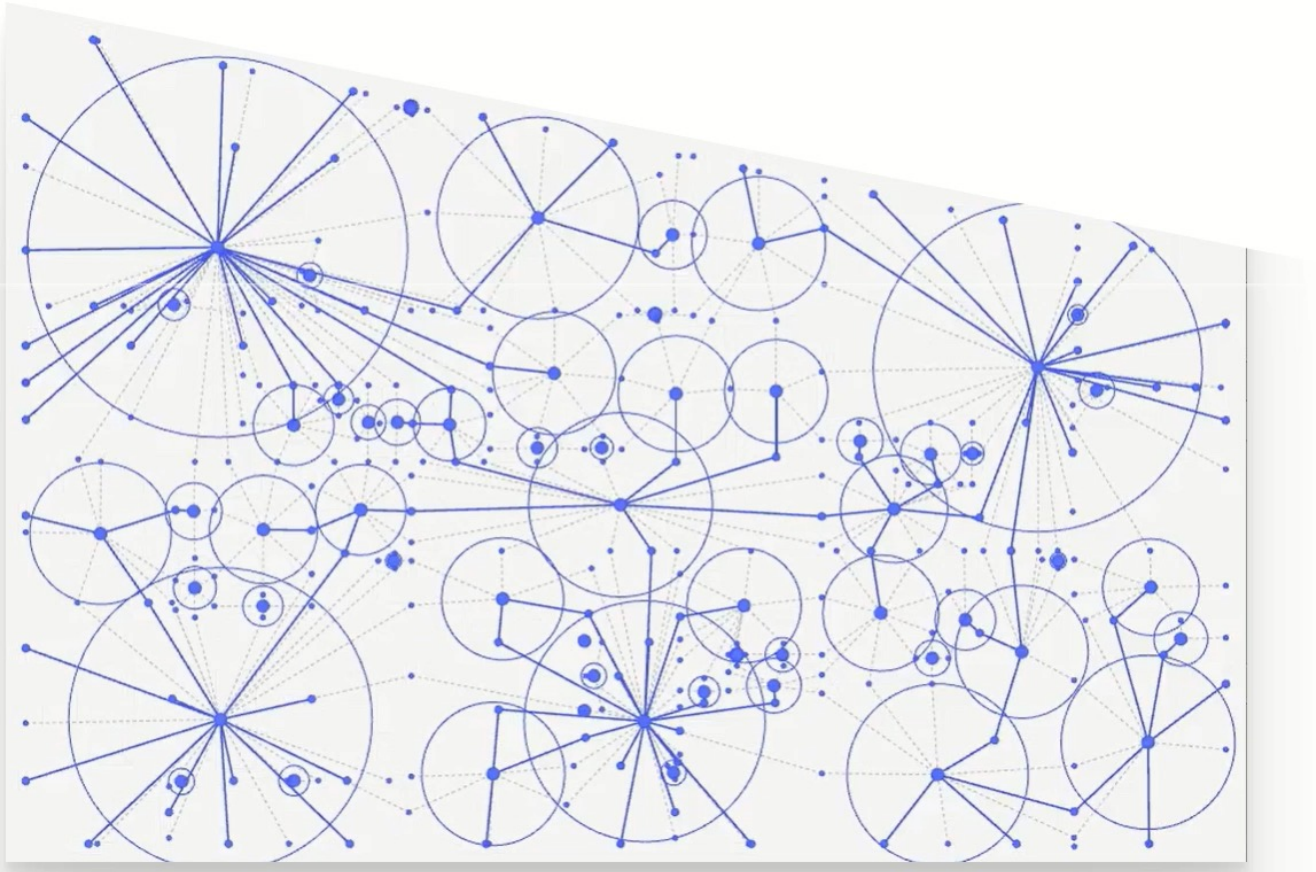


2: The Intelligence of Matter

AI-Assisted Design

Layout Generation

Finch

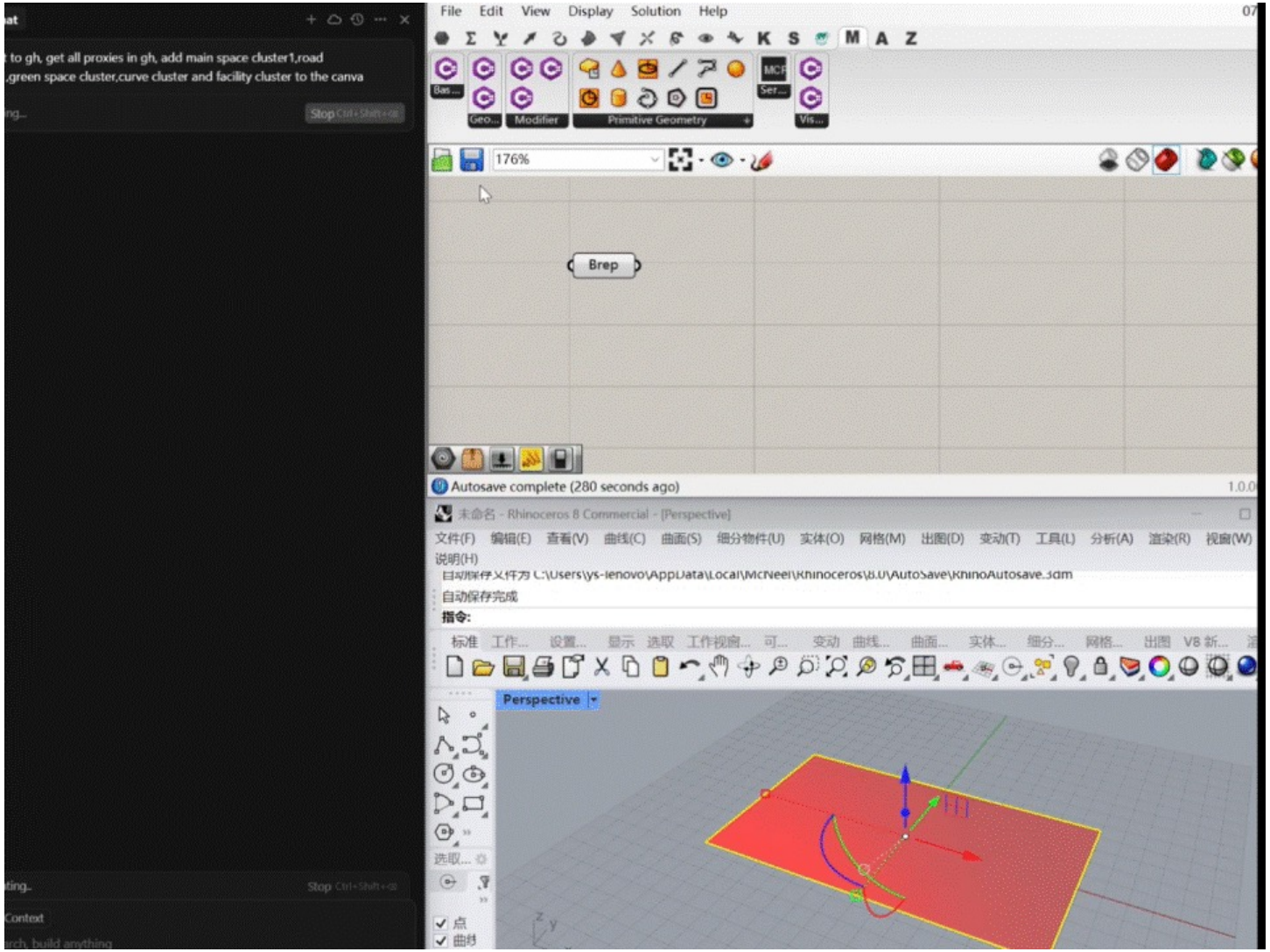


2: The Intelligence of Matter

AI-Assisted Design

Tool Use

Grasshopper MCP

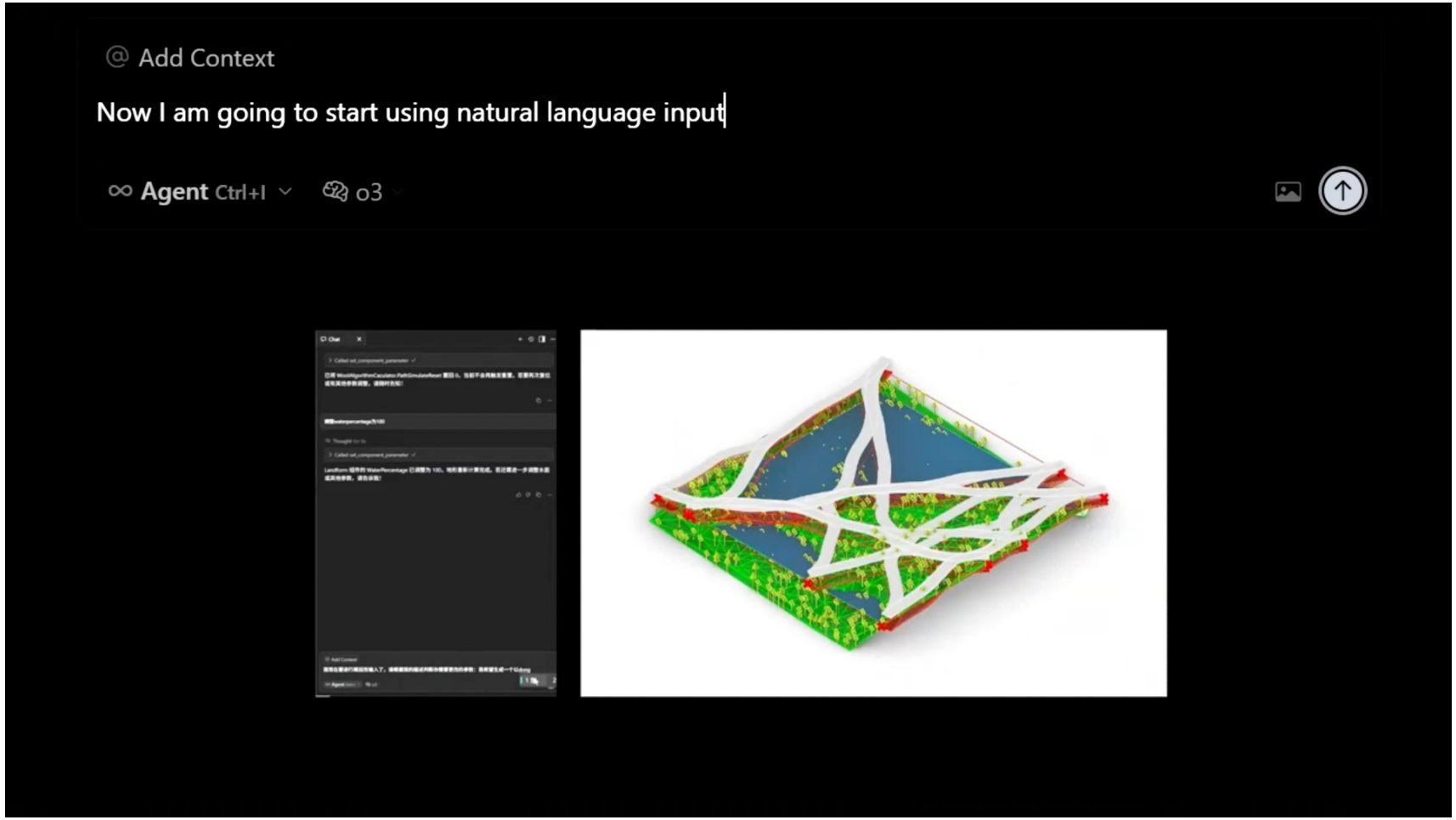


2: The Intelligence of Matter

AI-Assisted Design

Tool Use

Grasshopper MCP



2: The Intelligence of Matter

Digital Fabrication

3D Print

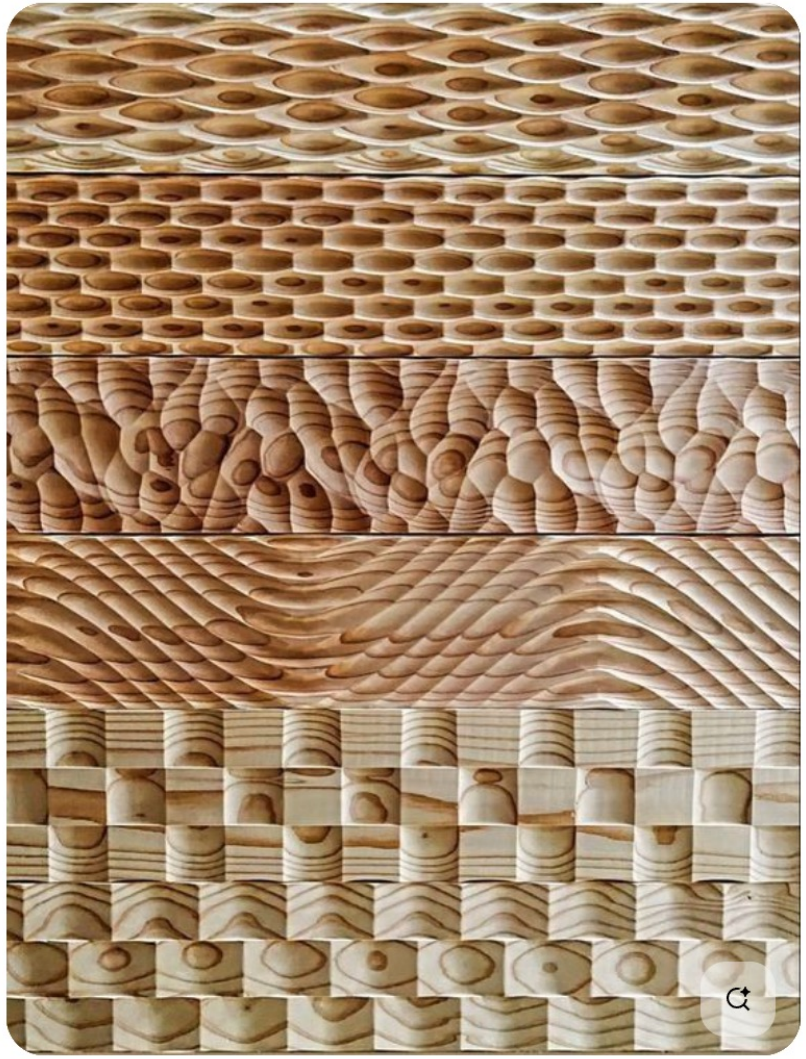


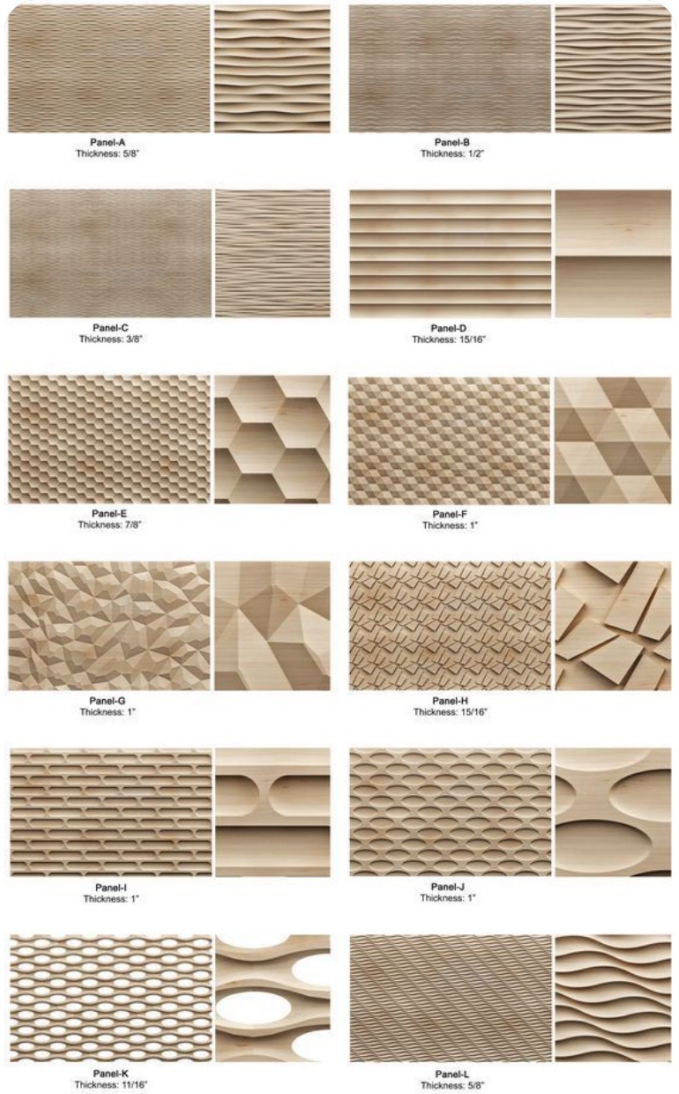


2: The Intelligence of Matter

Digital Fabrication

CNC



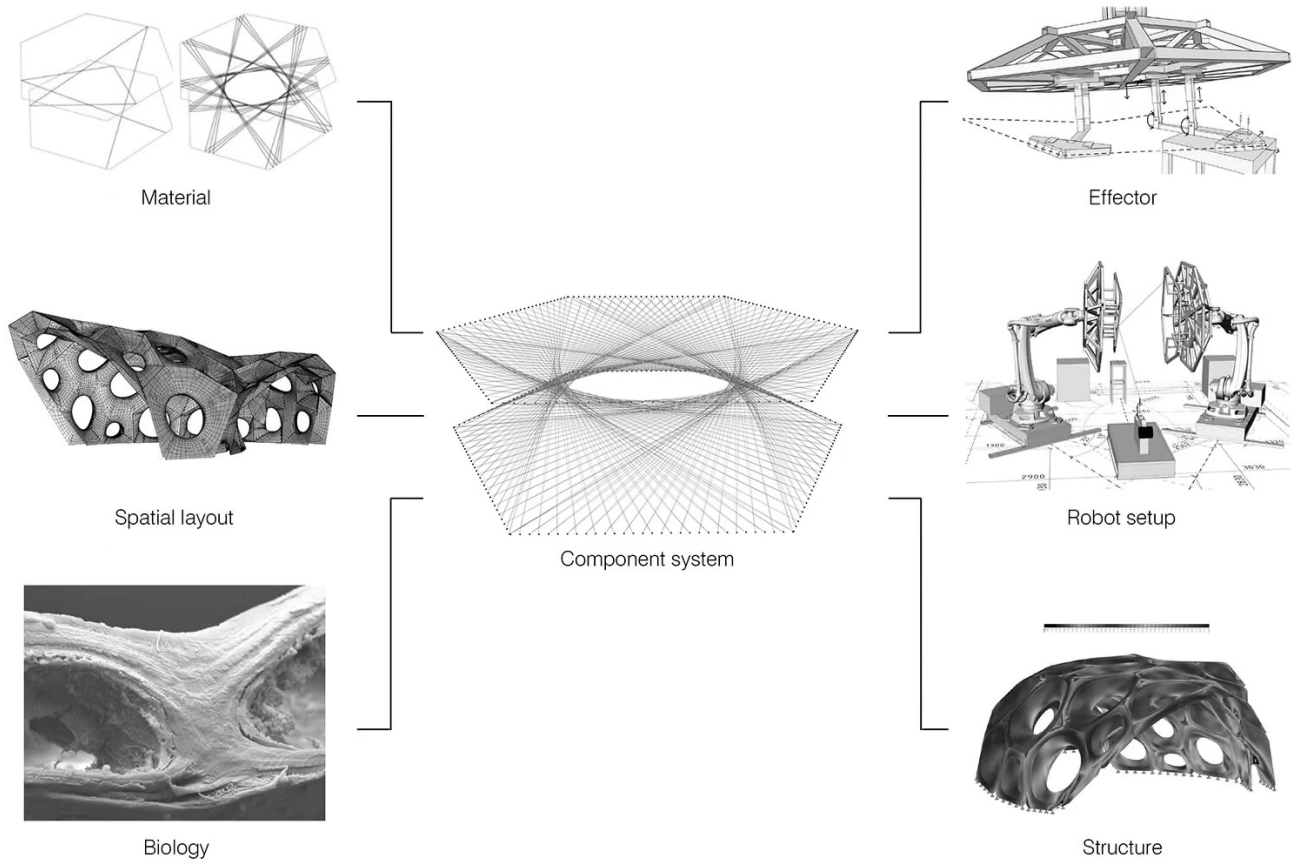


2: The Intelligence of Matter

Robotic Intelligence

ICD Pavilion

Achim Menges



Module 3: Advanced Form-Finding

- Data-driven Design
- Performance-driven Design
- Evolutionary Design

Concepts and Design Methods

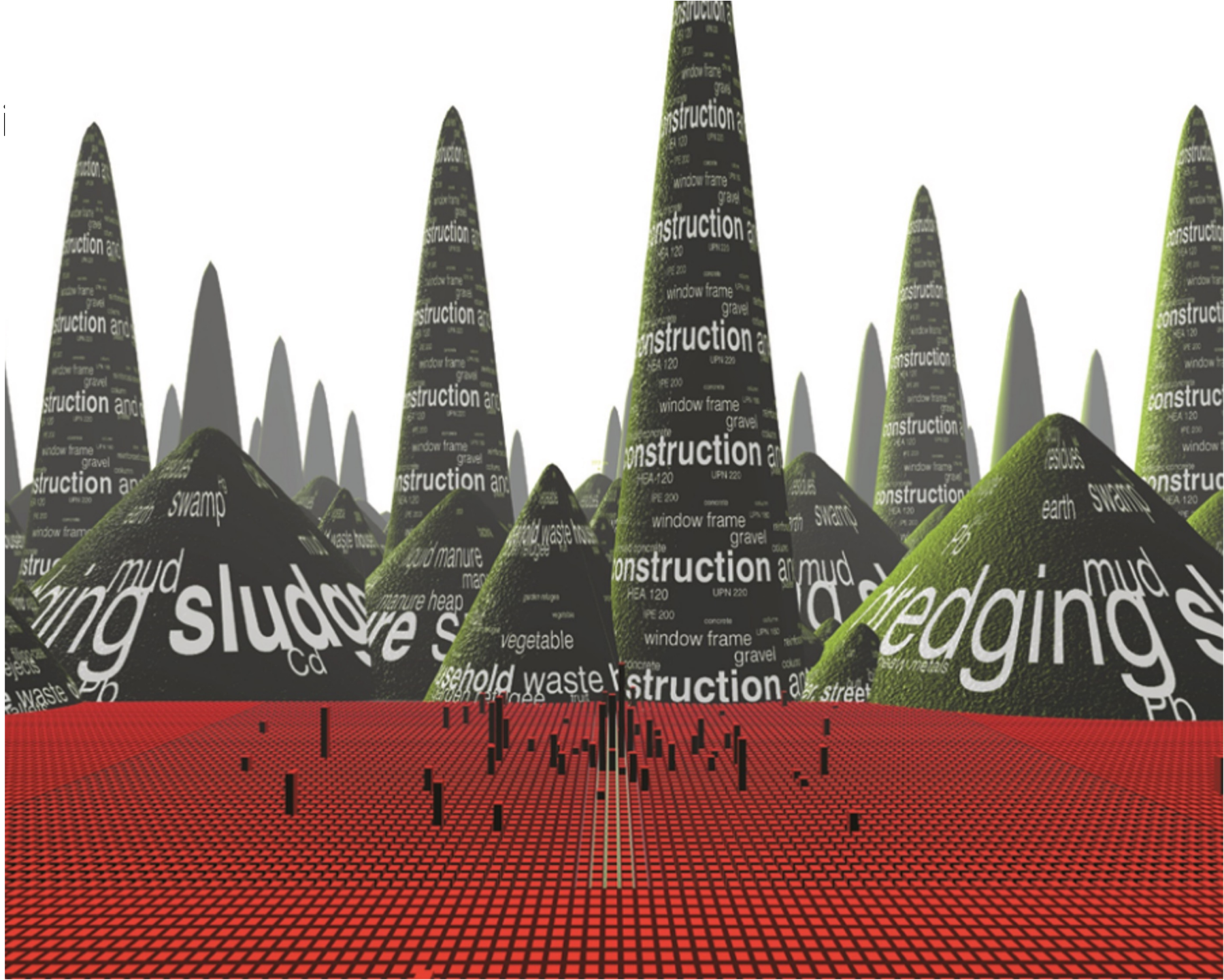
3: Advanced Form-Findi

Data-driven Design

Context as Data

Metacity/Datatown

MVRDV, 1999



3: Advanced Form-Finding

Data-driven Design

Context as Data

City that Never Sleeps

KPF UI, 2017

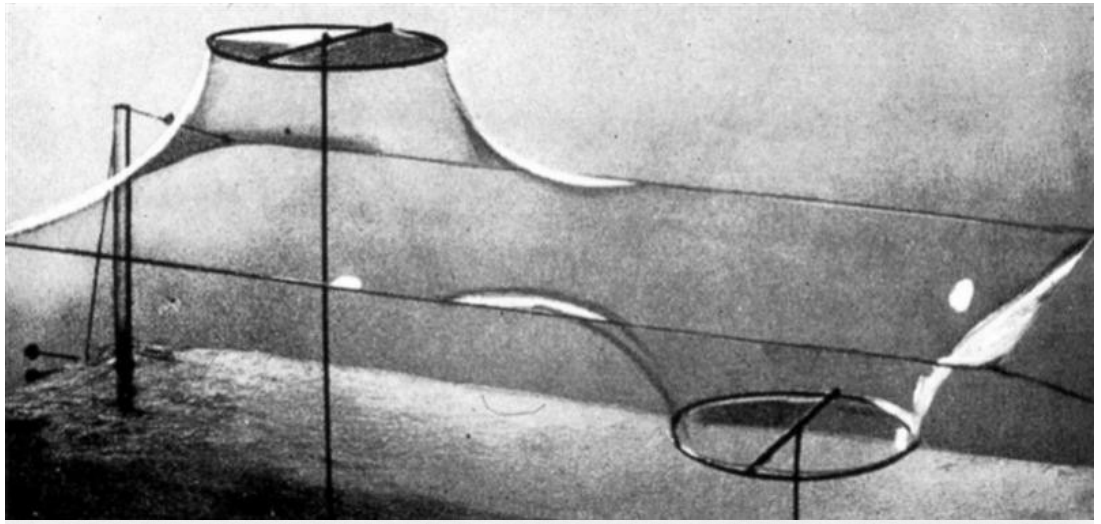


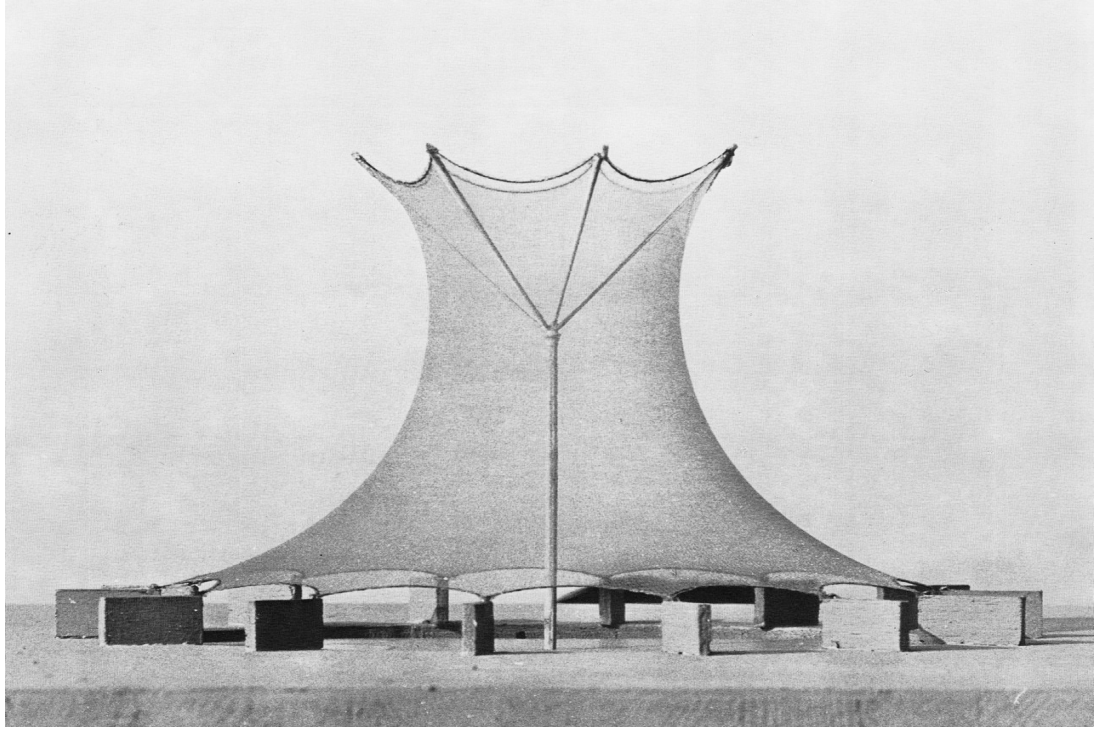
3: Advanced Form-Finding

Performance-driven Design

Physical Simulation

Frei Otto





Concepts and Design Methods

3: Advanced Form-Finding

Performance-driven Design

Physical Simulation

Munich Olympic Stadium

Frei Otto, 1972



Concepts and Design Methods

3: Advanced Form-Finding

Performance-driven Design

Physical Simulation

KnitCandela

ZHA/ETH Zurich, 2018



Concepts and Design Methods

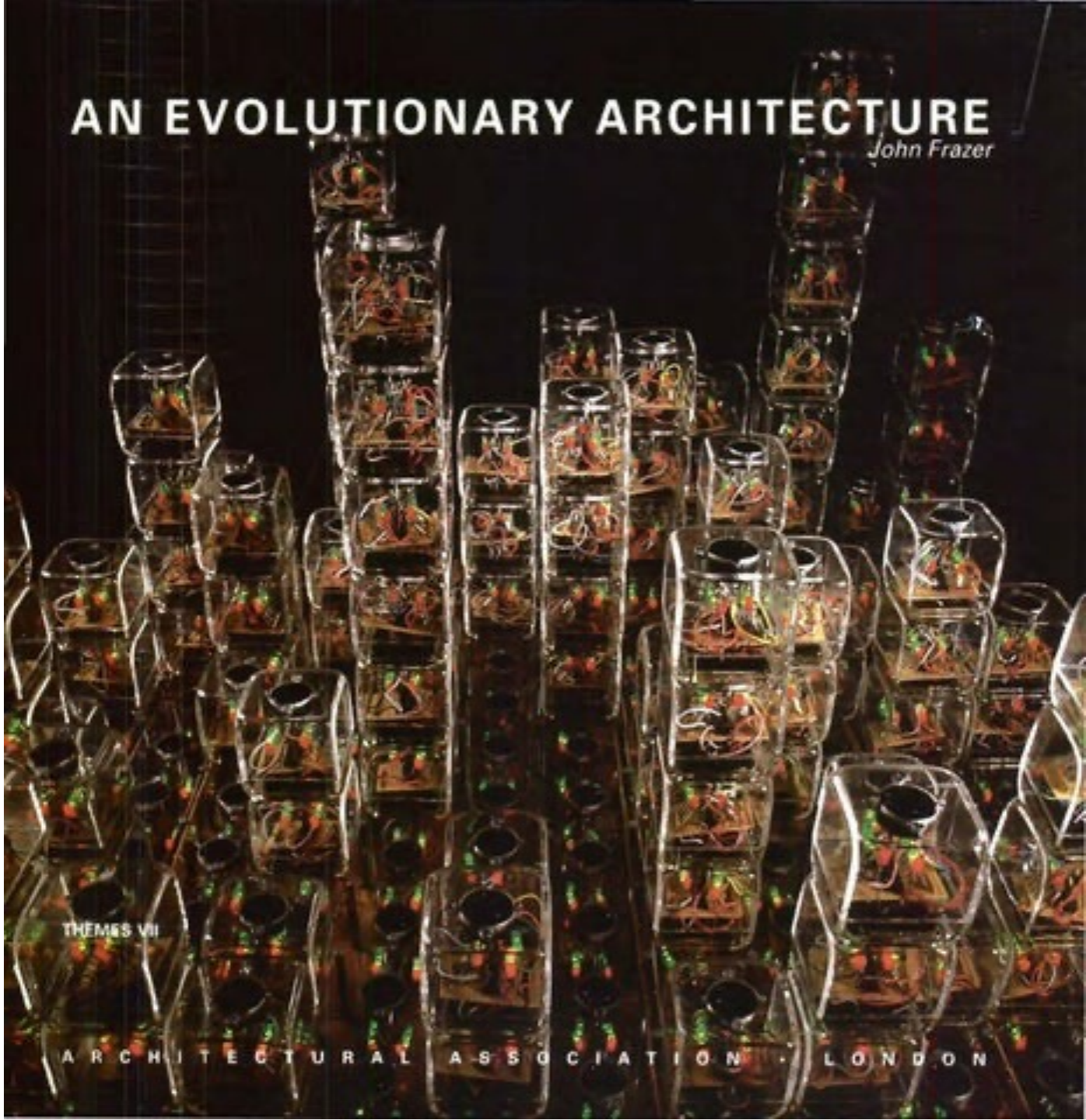
3: Advanced Form-Finding

Evolutionary Design

Genetic Code

An Evolutionary Architecture

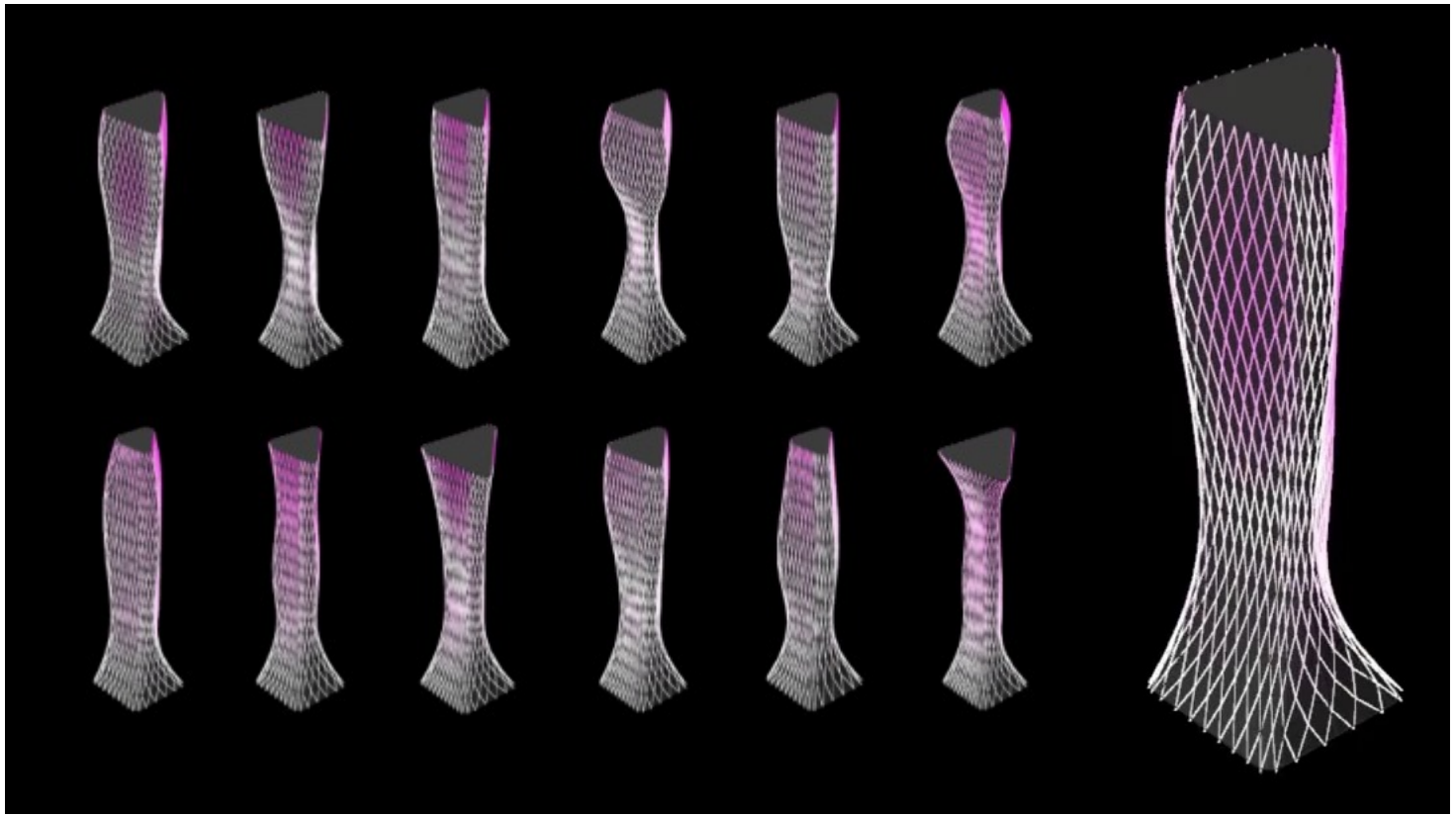
John Frazer



3: Advanced Form-Finding

Evolutionary Design

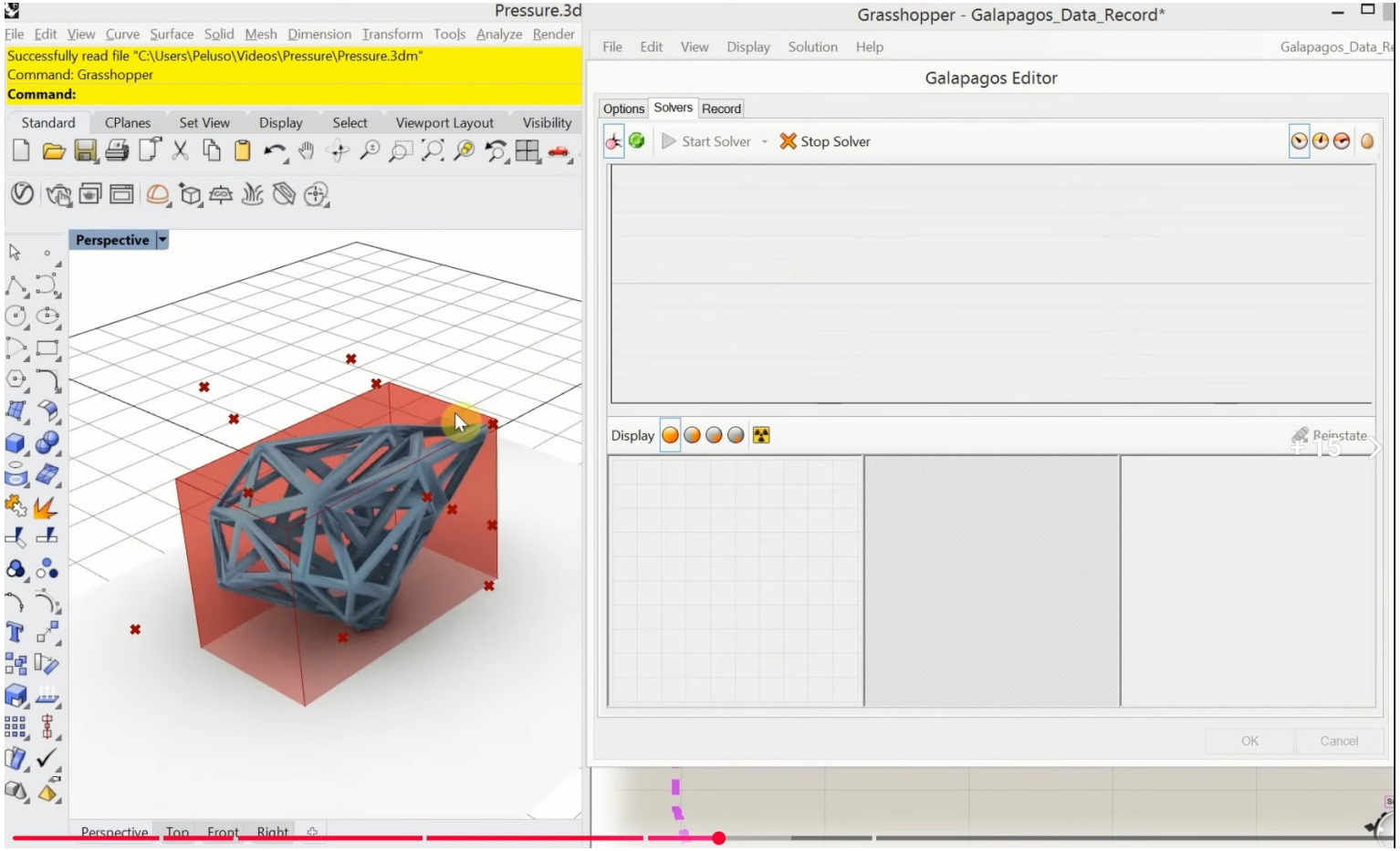
Optimization



3: Advanced Form-Finding

Evolutionary Design

Optimization

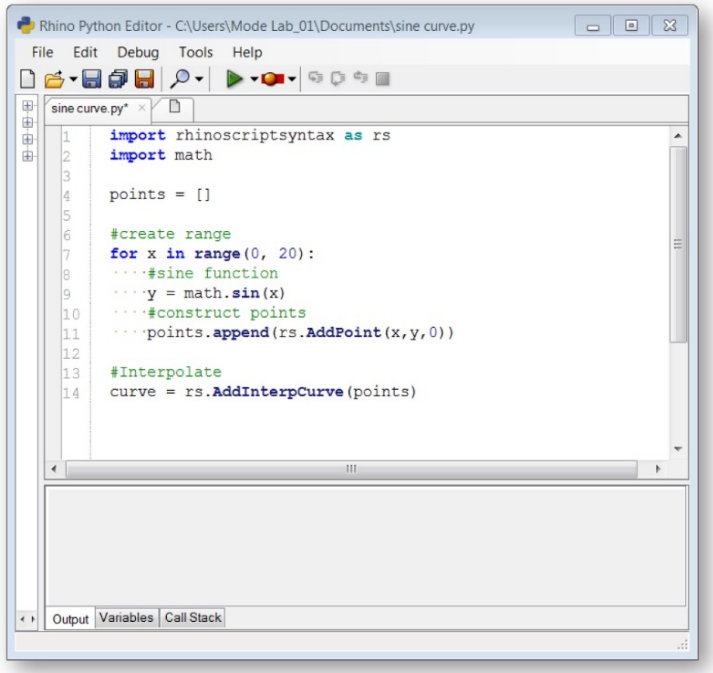


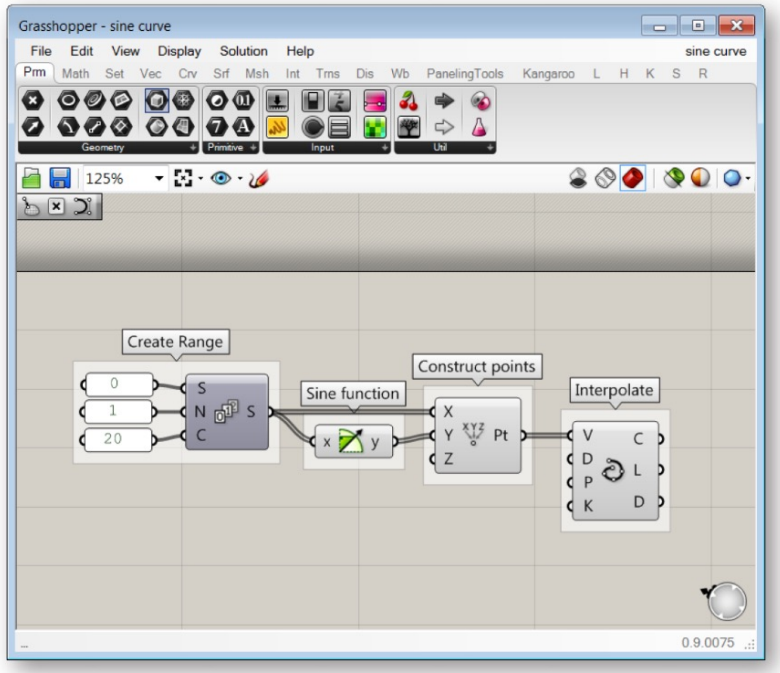
Grasshopper – What and Why

A Visual Programming Tool

Grasshopper is a plugin for Rhino that serves as a Visual Programming Language (VPL).

It transitions the design process from manual clicking to explicit logical construction.





Textual Programming

Traditional coding (Python, C#) uses abstract text. Syntax errors like a missing semicolon can break the entire system, making it unintuitive for visual thinkers.

Visual Programming

Grasshopper uses a graphical interface. Instead of typing, you drag "Components" onto a canvas and connect them with "Wires" to build logic.

This image show the process for drawing a sine curve in python and in Grasshopper.

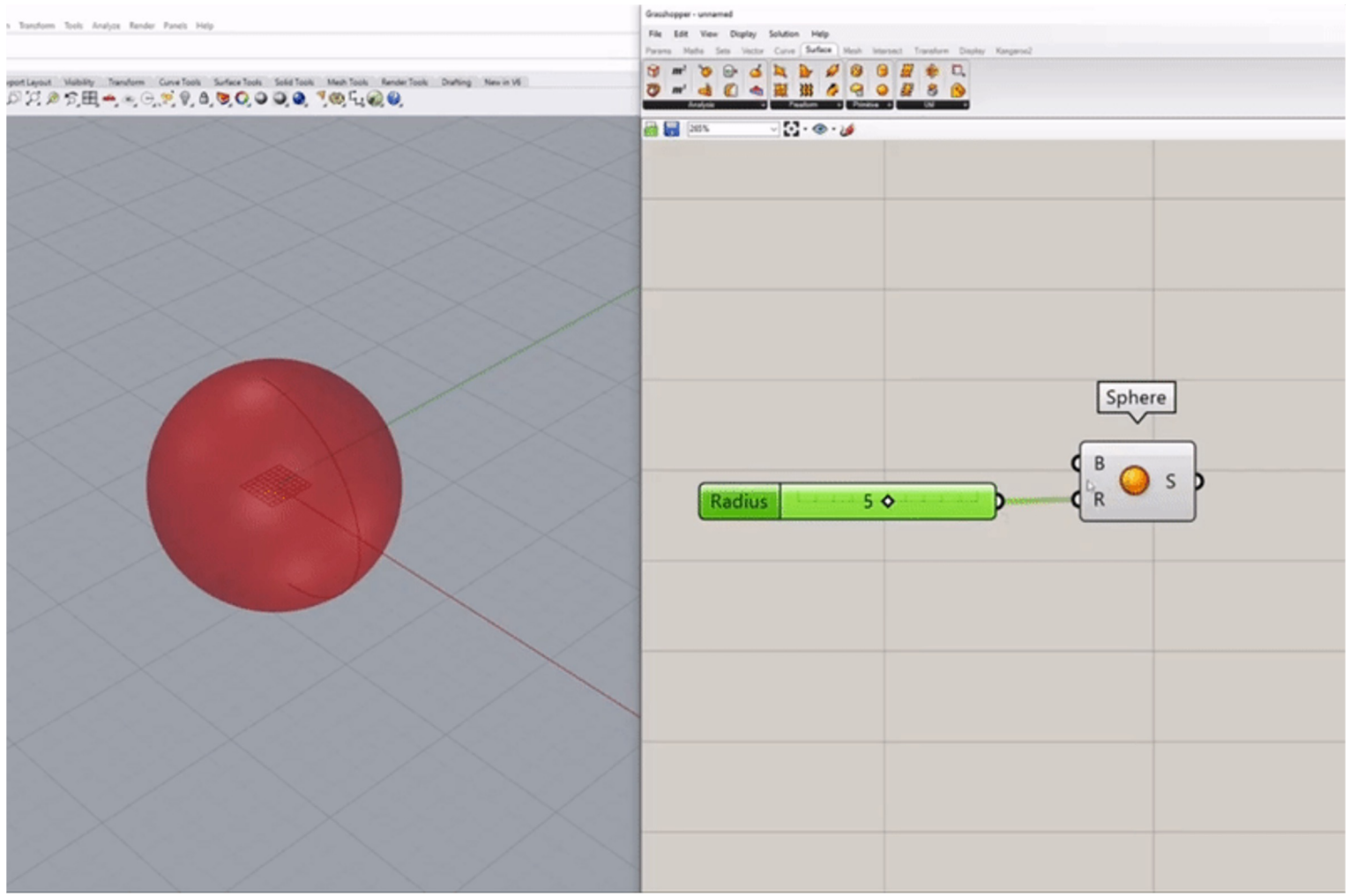
What is Grasshopper?

A Visual Programming Tool

Inputs: Data values or geometry fed into the left side of the component.

Function: The component box itself representing an operation (e.g., Draw Line).

Outputs: The resulting data emerging from the right side..

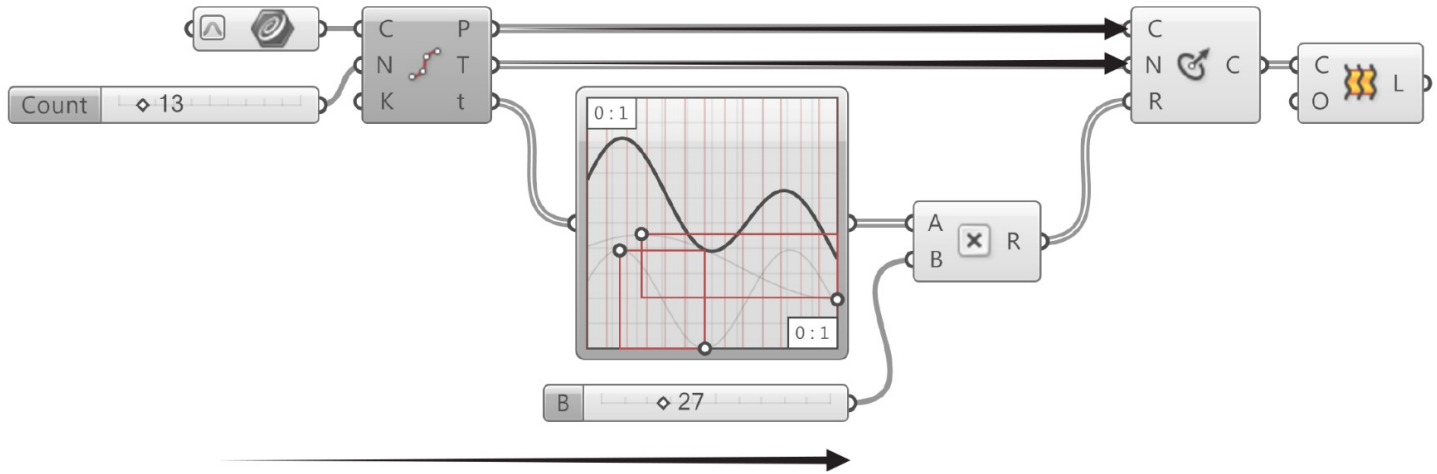


A Visual Programming Tool

Inputs: Data values or geometry fed into the left side of the component.

Function: The component box itself representing an operation (e.g., Draw Line).

Outputs: The resulting data emerging from the right side..



Program flow from left to right

A Visual Programming Tool

Algorithmic:

Designing a step-by-step procedure or "recipe" for form generation.

Parametric:

Defining relationships between variables rather than fixed, static objects.

Generative:

Setting up rule systems where complex results emerge from simple constraints.

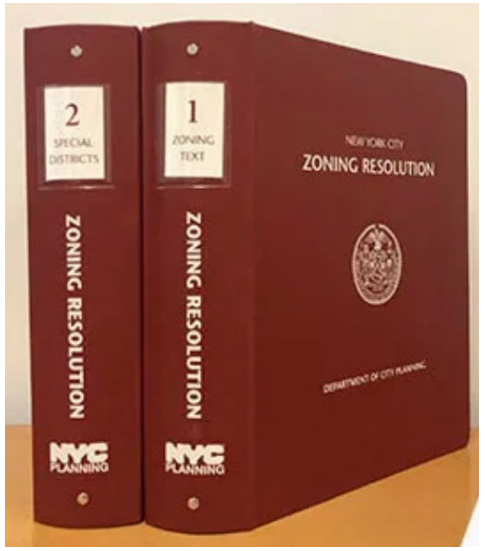
.

Why Use Grasshopper?

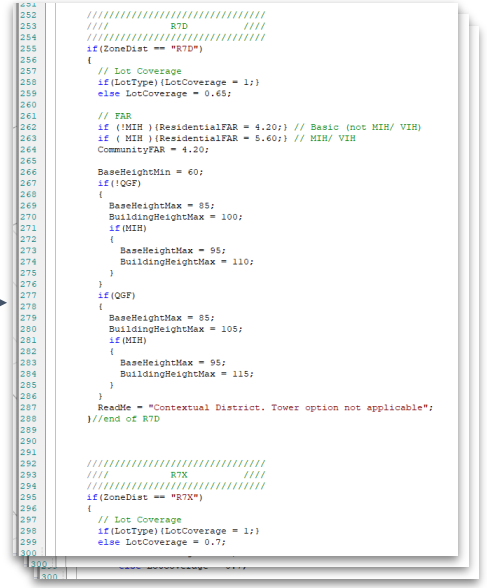
Automation

The "Boring Stuff“

Eliminate tedious tasks like renumbering parking stalls or unrolling facade panels. Write the script once, execute instantly.



Zoning Code



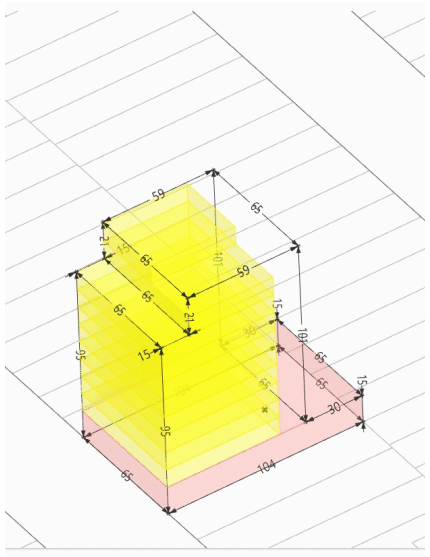
```
    ////////////////////////////////
    ////          R7D           ////
    ////////////////////////////////
    if(ZoneDist == "R7D")
    {
      // Lot Coverage
      if(LotType){LotCoverage = 1;}
      else LotCoverage = 0.65;

      // FAR
      if (!MIH ){ResidentialFAR = 4.20;} // Basic (not MIH/ VIH)
      if ( MIH ){ResidentialFAR = 5.60;} // MIH/ VIH
      CommunityFAR = 4.20;

      BaseHeightMin = 60;
      if(!QGF)
      {
        BaseHeightMax = 85;
        BuildingHeightMax = 100;
        if(MIH)
        {
          BaseHeightMax = 95;
          BuildingHeightMax = 110;
        }
      }
      if(QGF)
      {
        BaseHeightMax = 85;
        BuildingHeightMax = 105;
        if(MIH)
        {
          BaseHeightMax = 95;
          BuildingHeightMax = 115;
        }
      }
      ReadMe = "Contextual District. Tower option not applicable";
    }//end of R7D


    ////////////////////////////////
    ////          R7X           ////
    ////////////////////////////////
    if(ZoneDist == "R7X")
    {
      // Lot Coverage
      if(LotType){LotCoverage = 1;}
      else LotCoverage = 0.7;
```

Computational Code



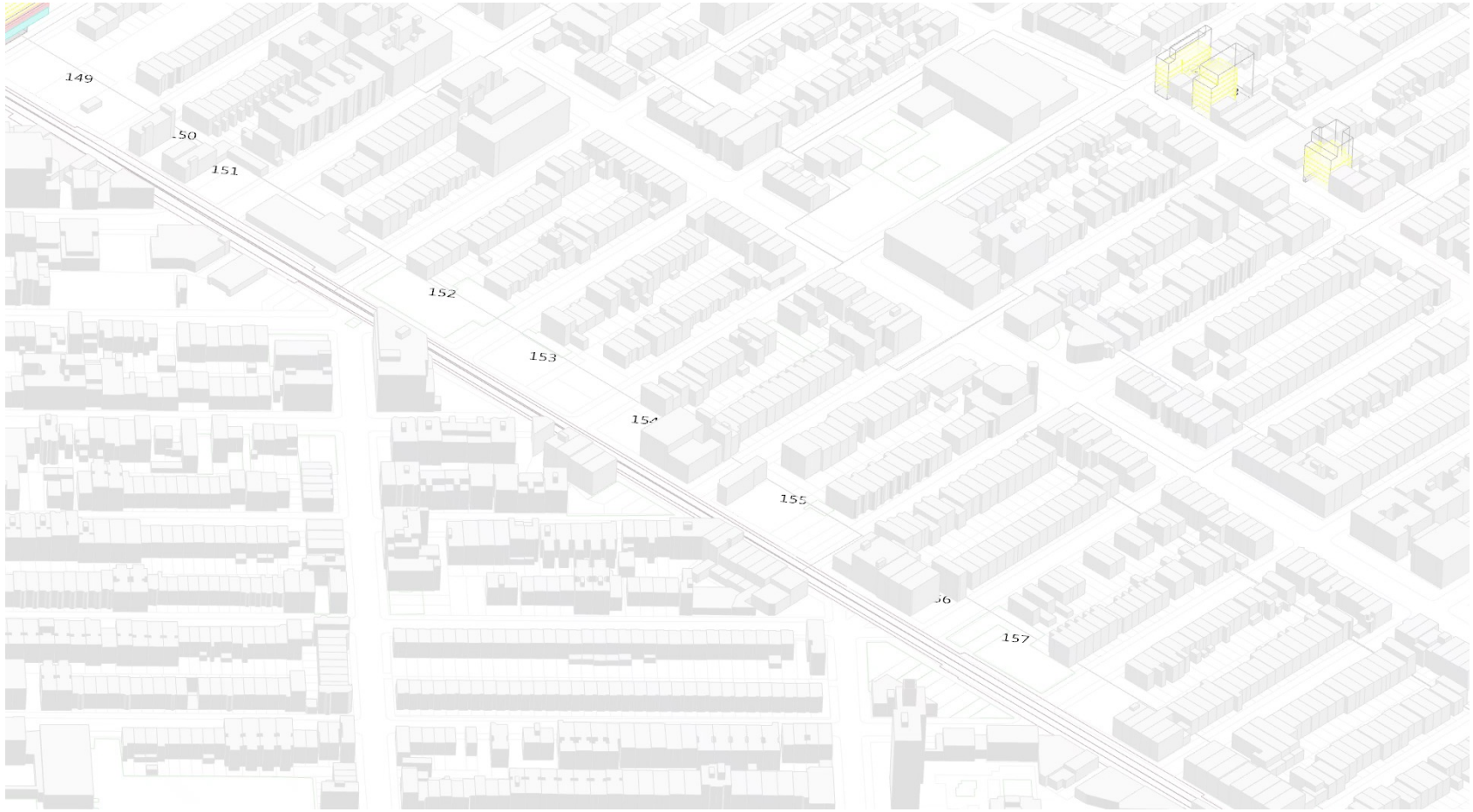
3D Visualization

Why Use Grasshopper?

Automation

The "Boring Stuff“

Eliminate tedious tasks like renumbering parking stalls or unrolling facade panels. Write the script once, execute instantly.

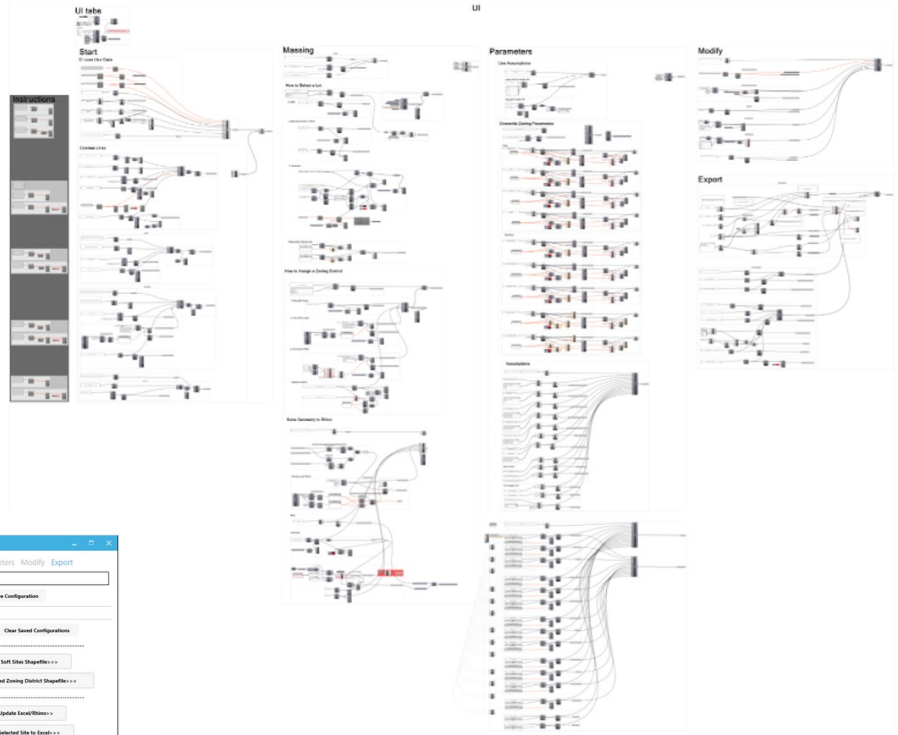


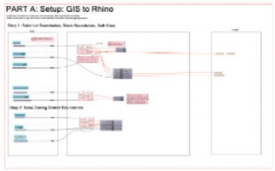
Why Use Grasshopper?

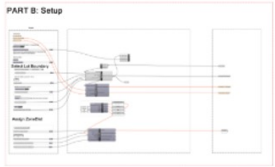
Automation

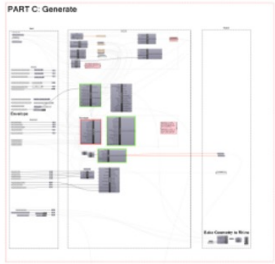
The "Boring Stuff“

Eliminate tedious tasks like renumbering parking stalls or unrolling facade panels. Write the script once, execute instantly.

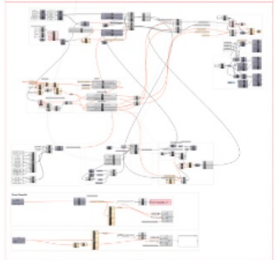


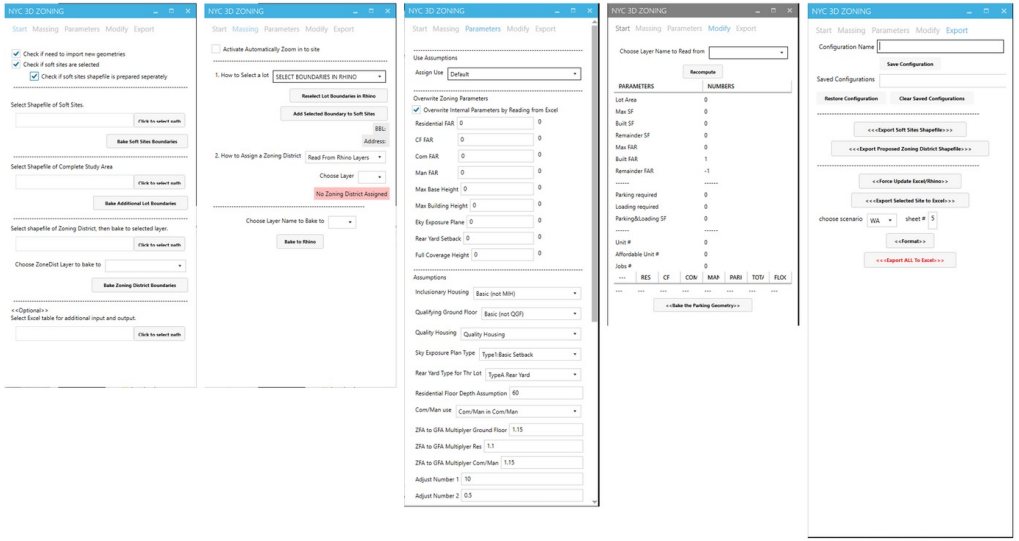








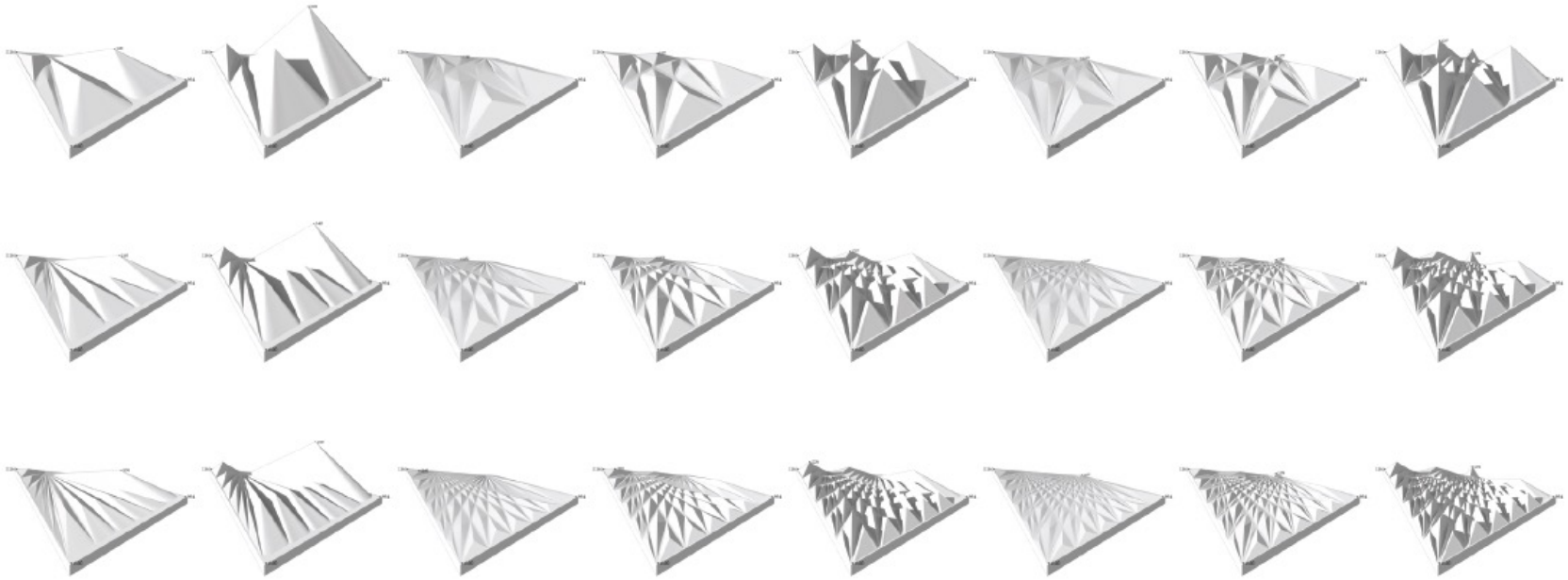




Iteration

Design Space Exploration

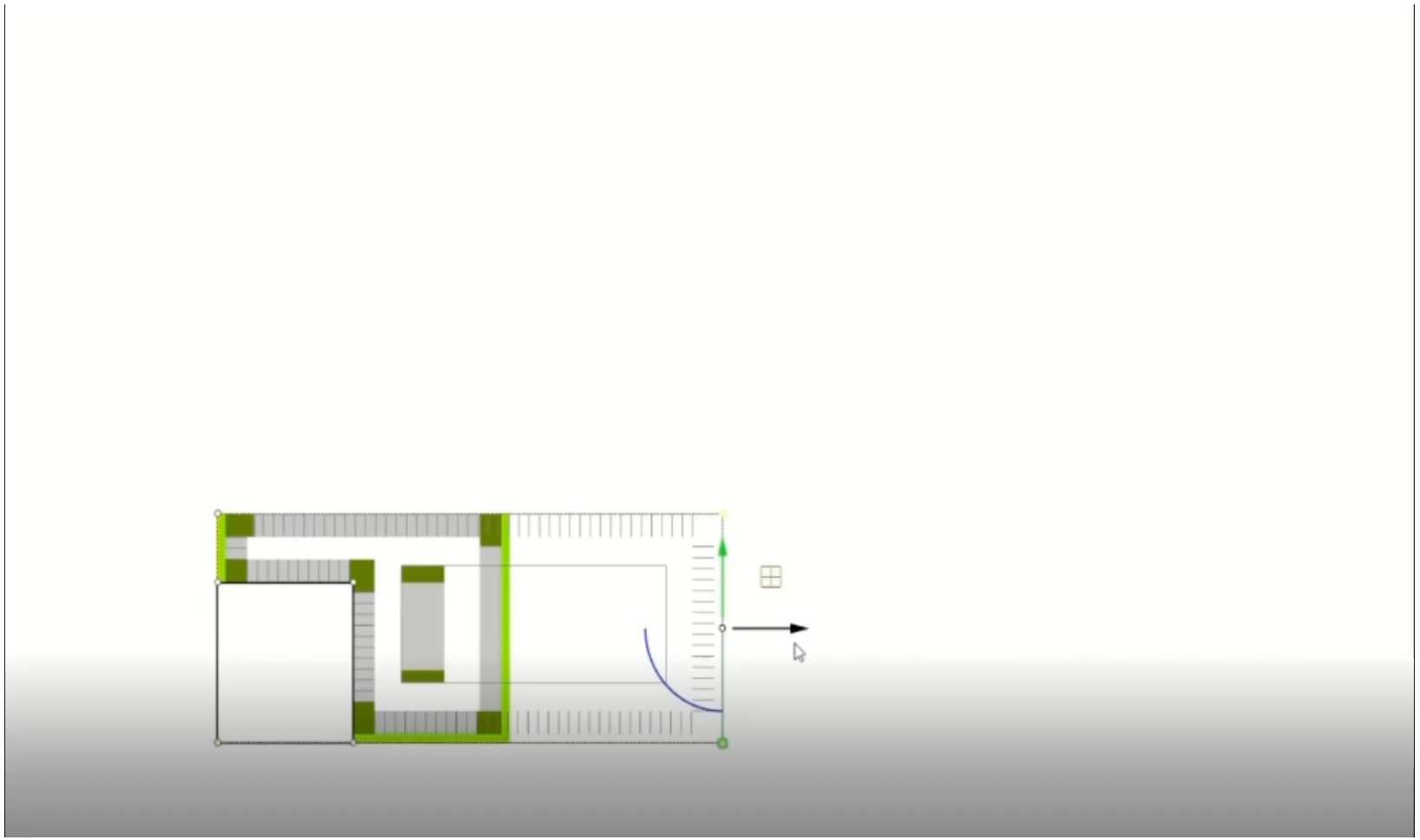
Design Space Exploration. Exploring thousands of options by sliding a single parameter (e.g., Tower Twist).



Encapsulation

Reusing Logic

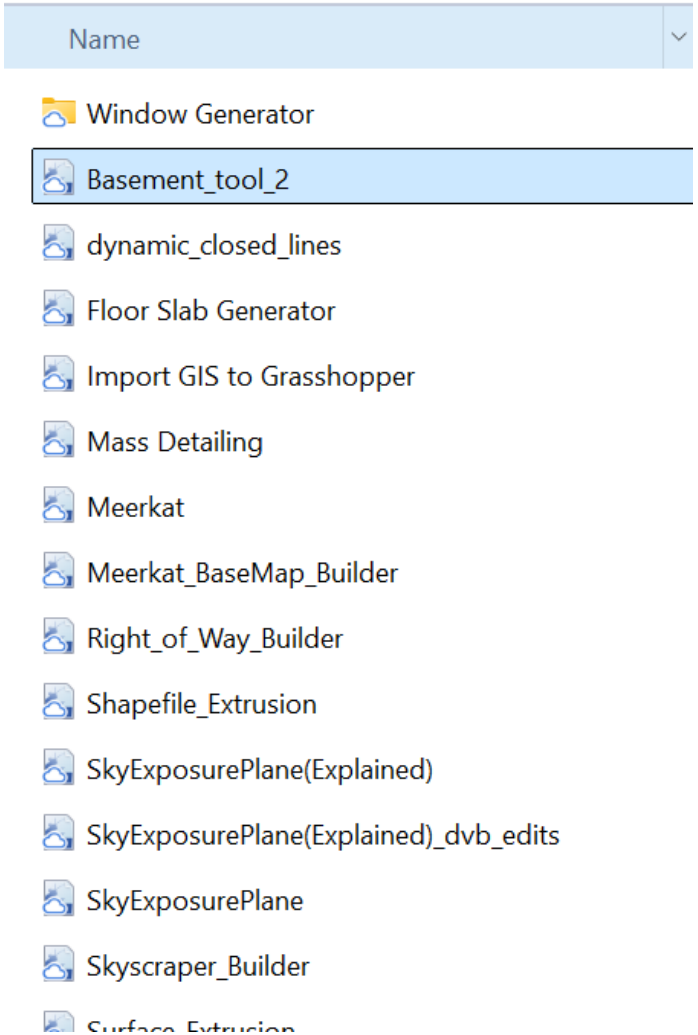
Saving complex logic to reuse later. Building your own library of architectural intelligence.



Encapsulation

Reusing Logic

Saving complex logic to reuse later. Building your own library of architectural intelligence.

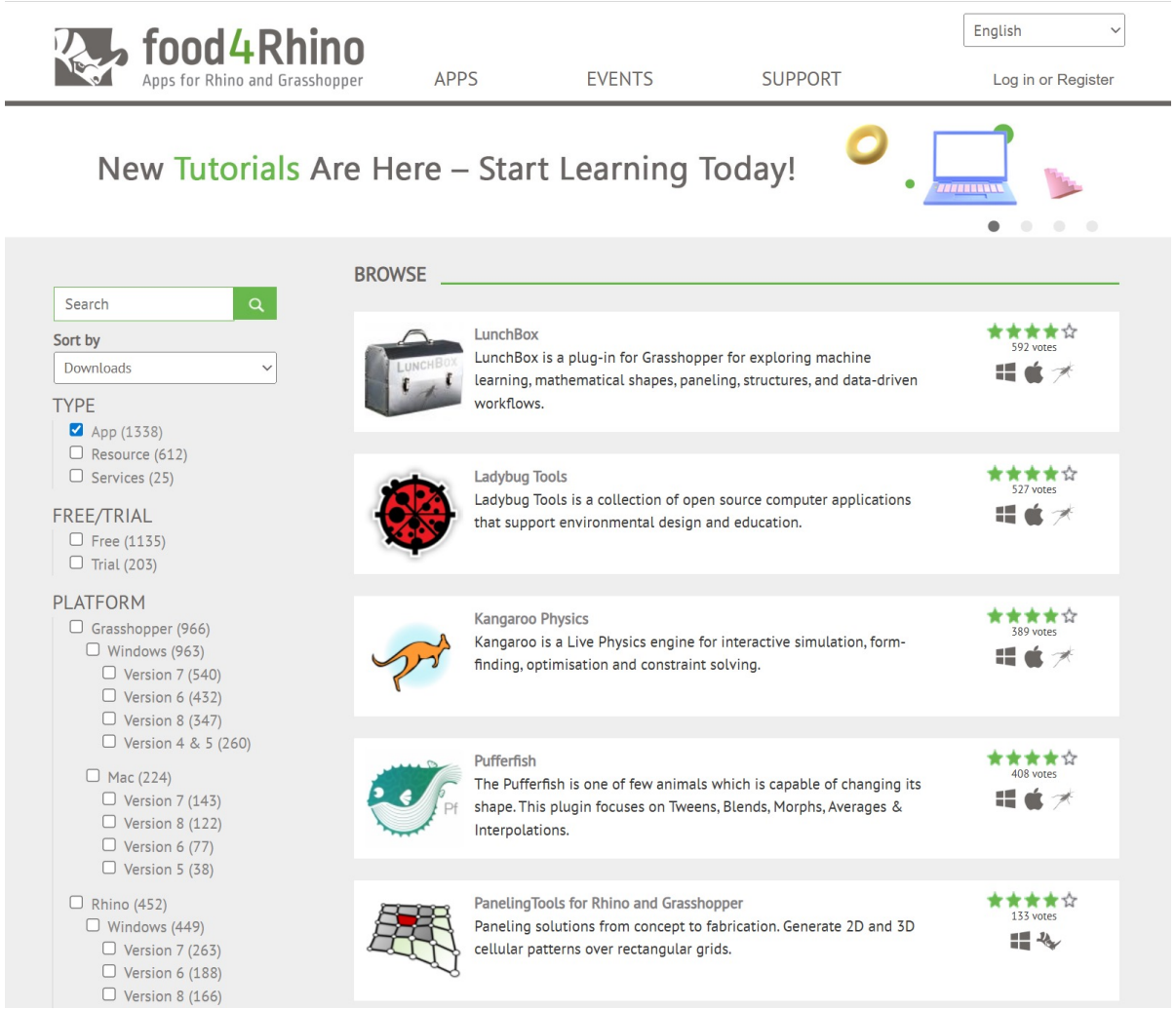


Why Use Grasshopper?

Plugins' Ecosystem

"The Zoo"

It is not just a tool; it is a platform. Thousands of plugins for Physics, Environmental Analysis, and Robotics.



Technology is the answer, but what was the question?