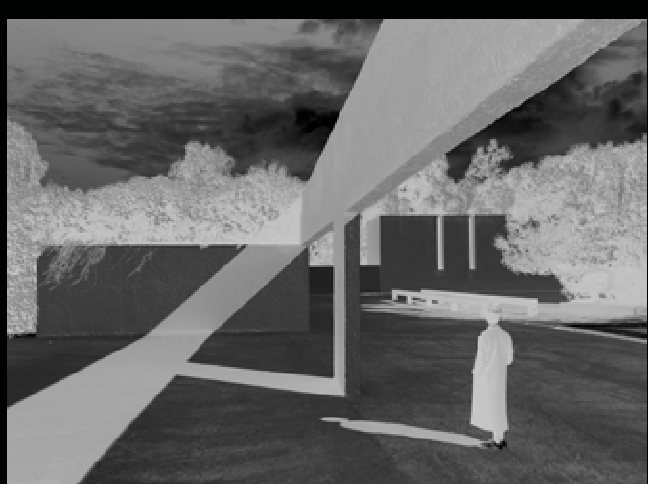
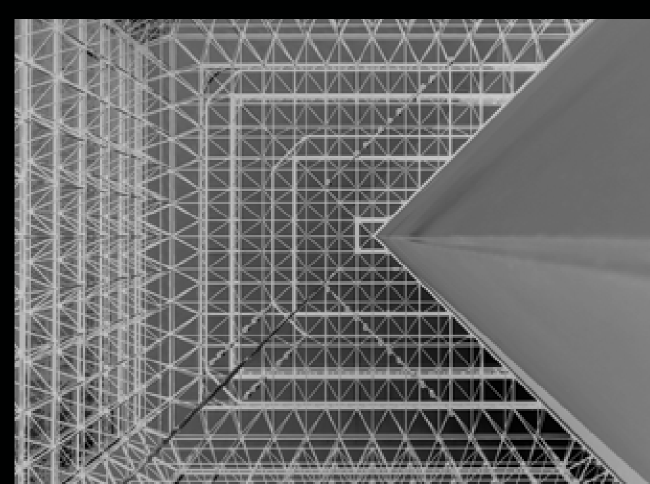



Porfolio | Computational Design | Building Simulation

QINGRU MIRAH XU

許 清 如

5 Selected Works | Academic | 2019 - 2020 Semester

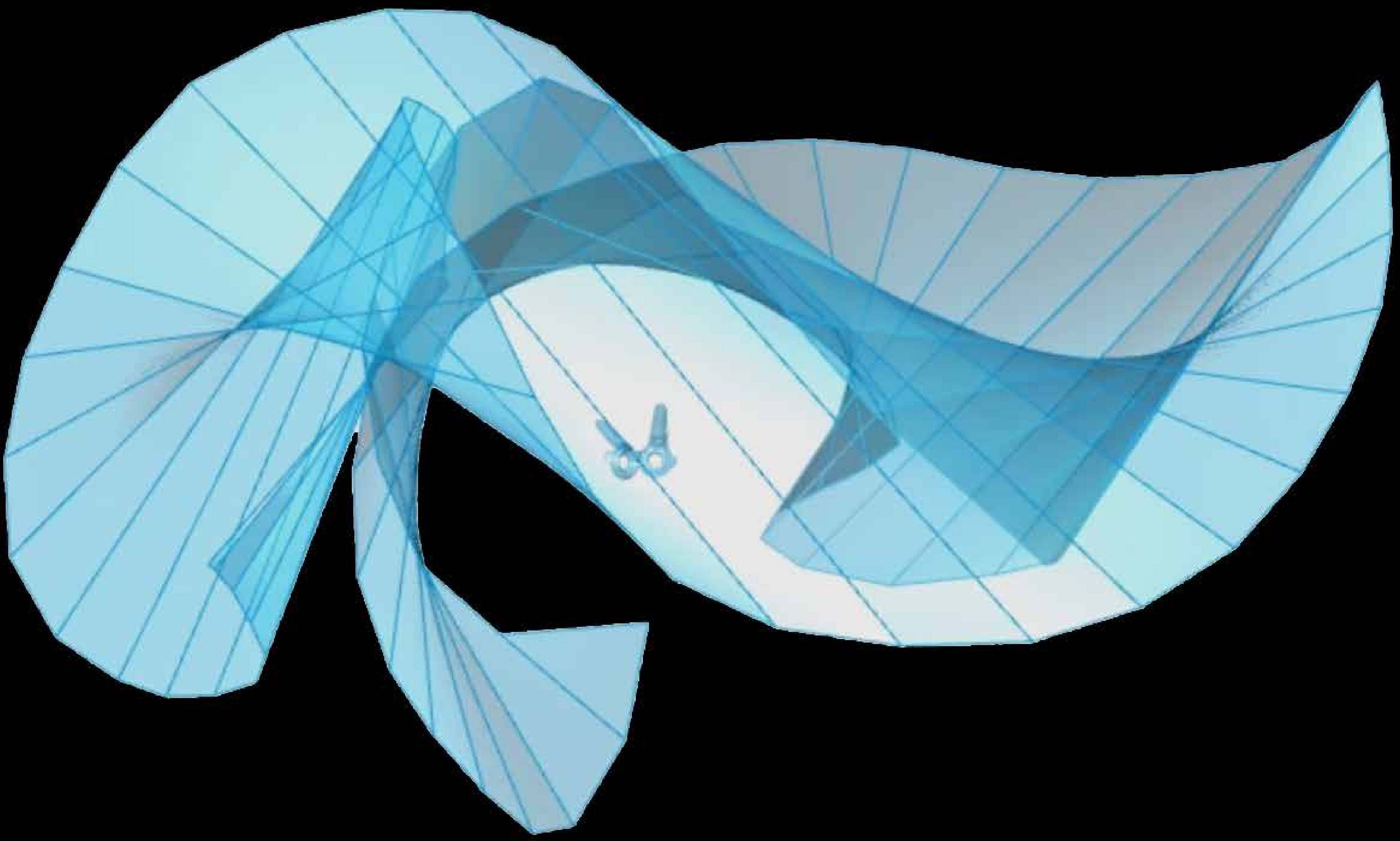



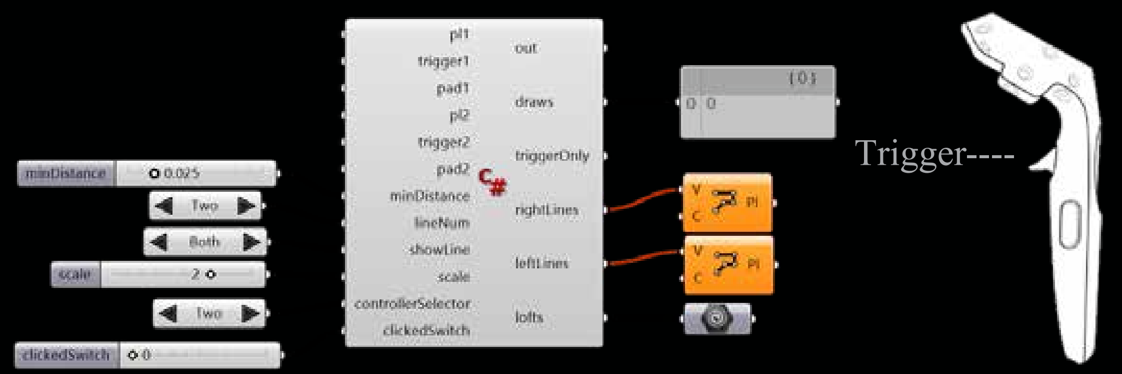

CONTENT

LIVE DESIGN
Real Time 3D Design in Virtual Reality

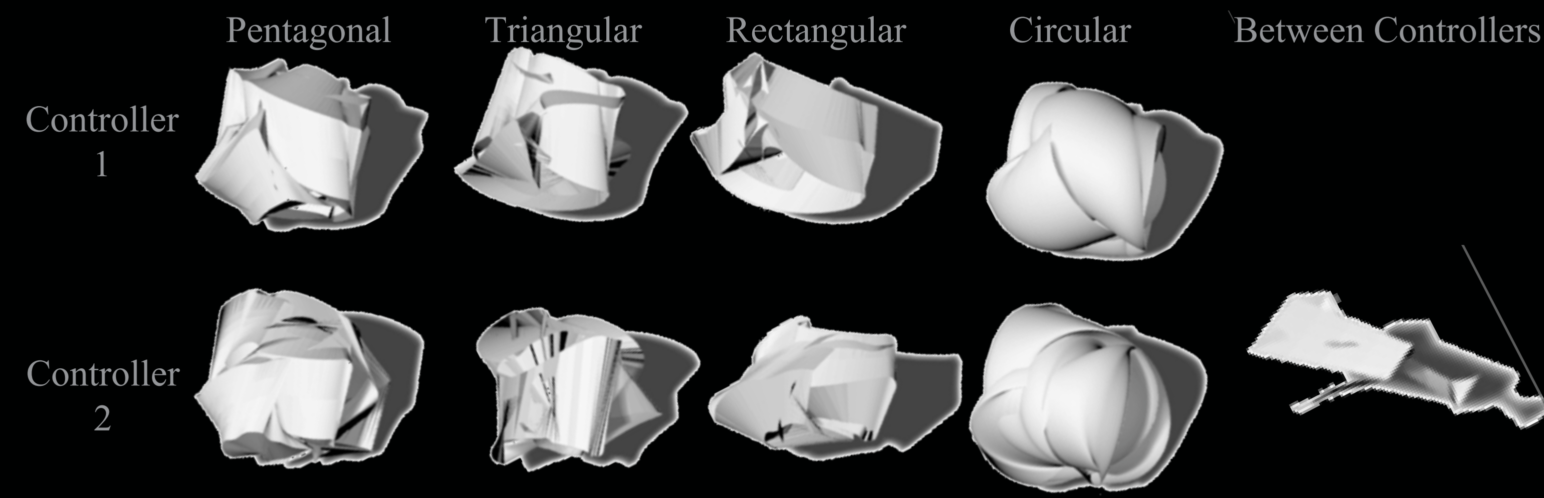
Introduction to Computational Design | GSD
Instructor: Jose Luis García del Castillo y López | September
Role: Concept, Code, and Visual Development
Collaborators: Sunghwan Lim, Moulshree Mittal

Implemented C# scripts in Grasshopper to interact with HTC Vive controllers. Allowing users to draw in the virtual space with the controllers and explore various geometry lofting options in real-time.





Script and Controler Inputs
Inputs to c# script component and Vive controller

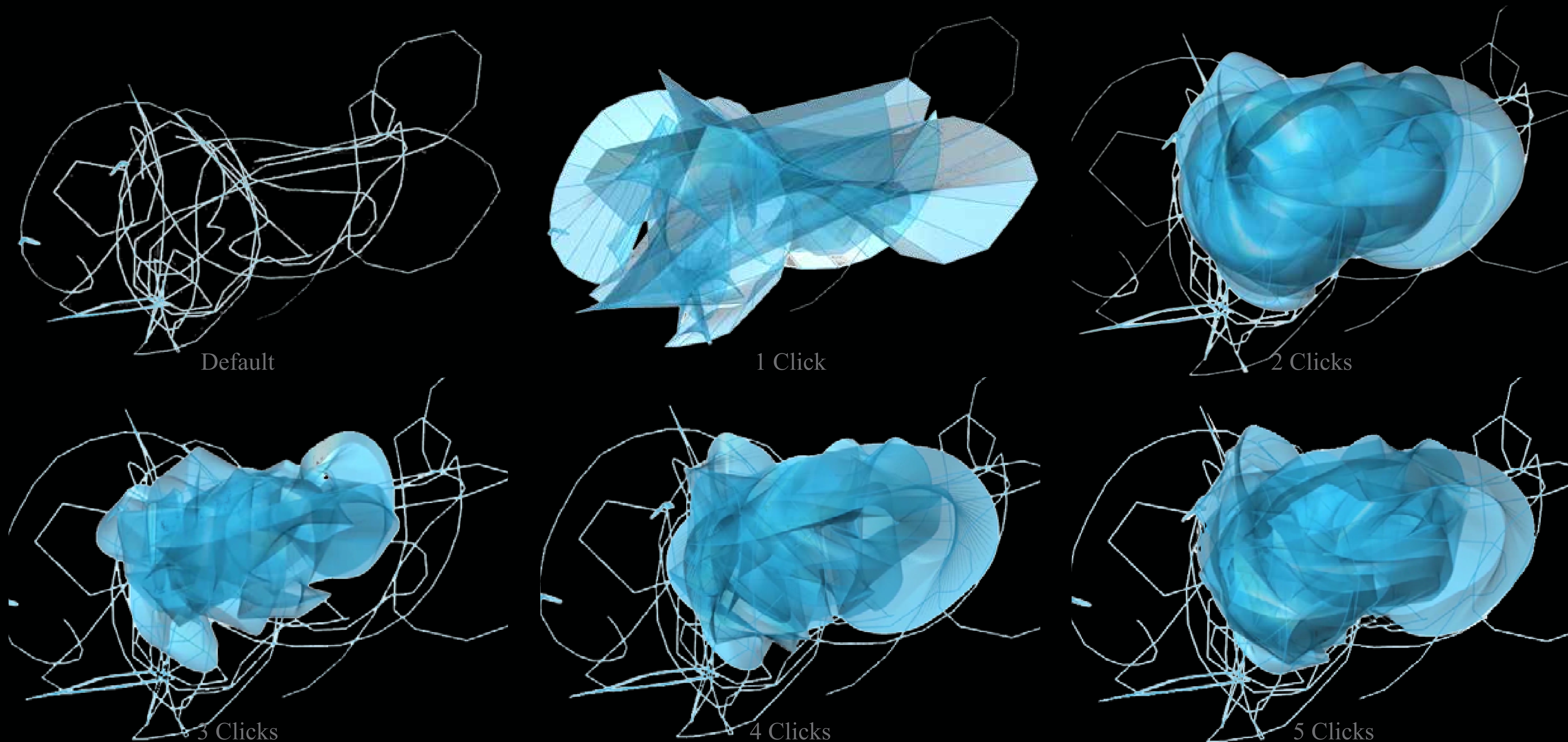


Output Options
Pipe output with different cross-section geometry

Image free-range geometries creation with Virtual Reality, using controllers how Jackson Pollock used his arms.

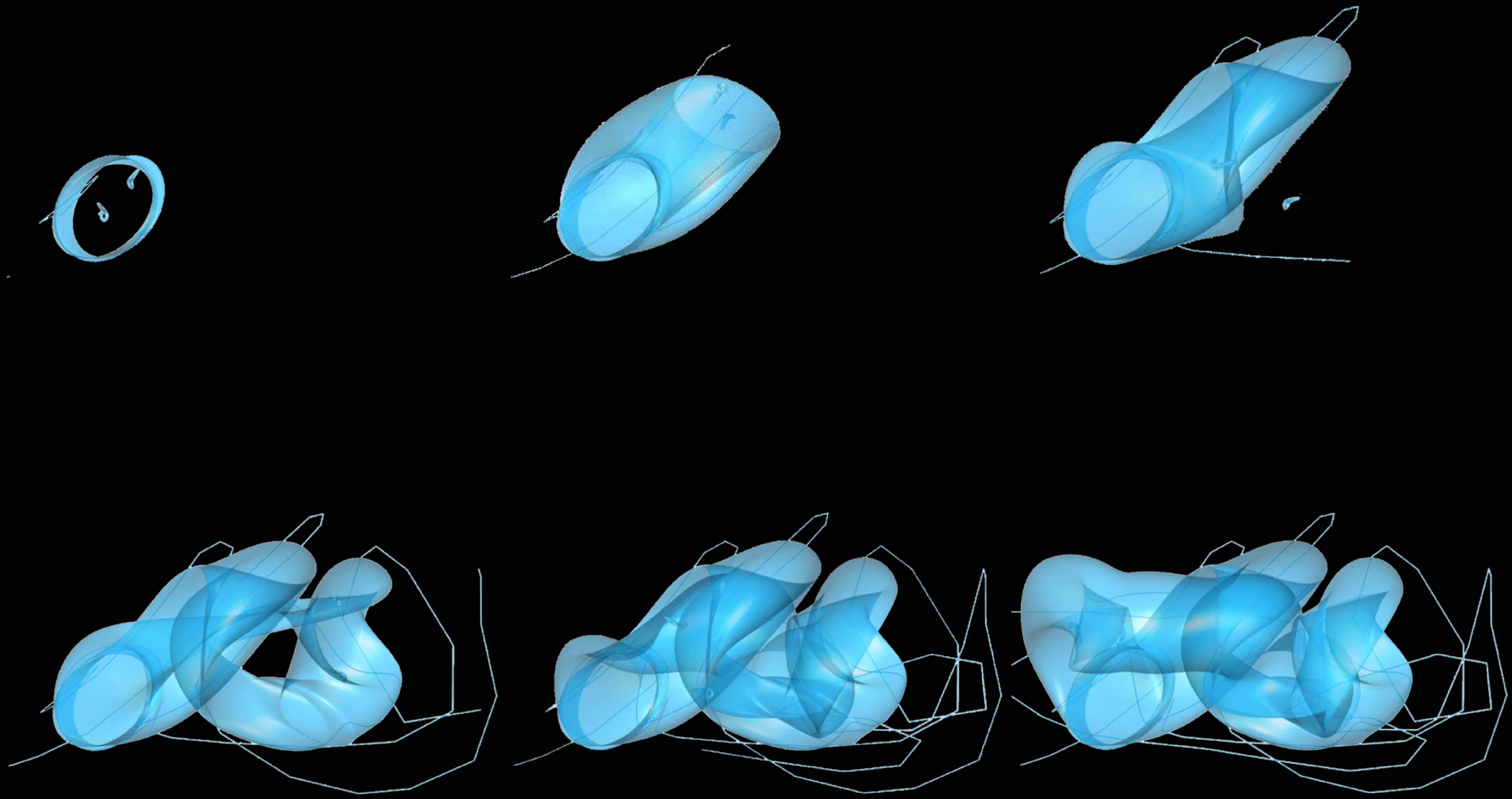
This is that reality.

Dance with VR controllers and render your geometries signature.



Script and Controler Inputs
Display options mapped to the number of times trigger is pulled

Create polyline using the controllers and stretch the limitless material between limbs. Click through to marval at the geometric volumns created out of thin air

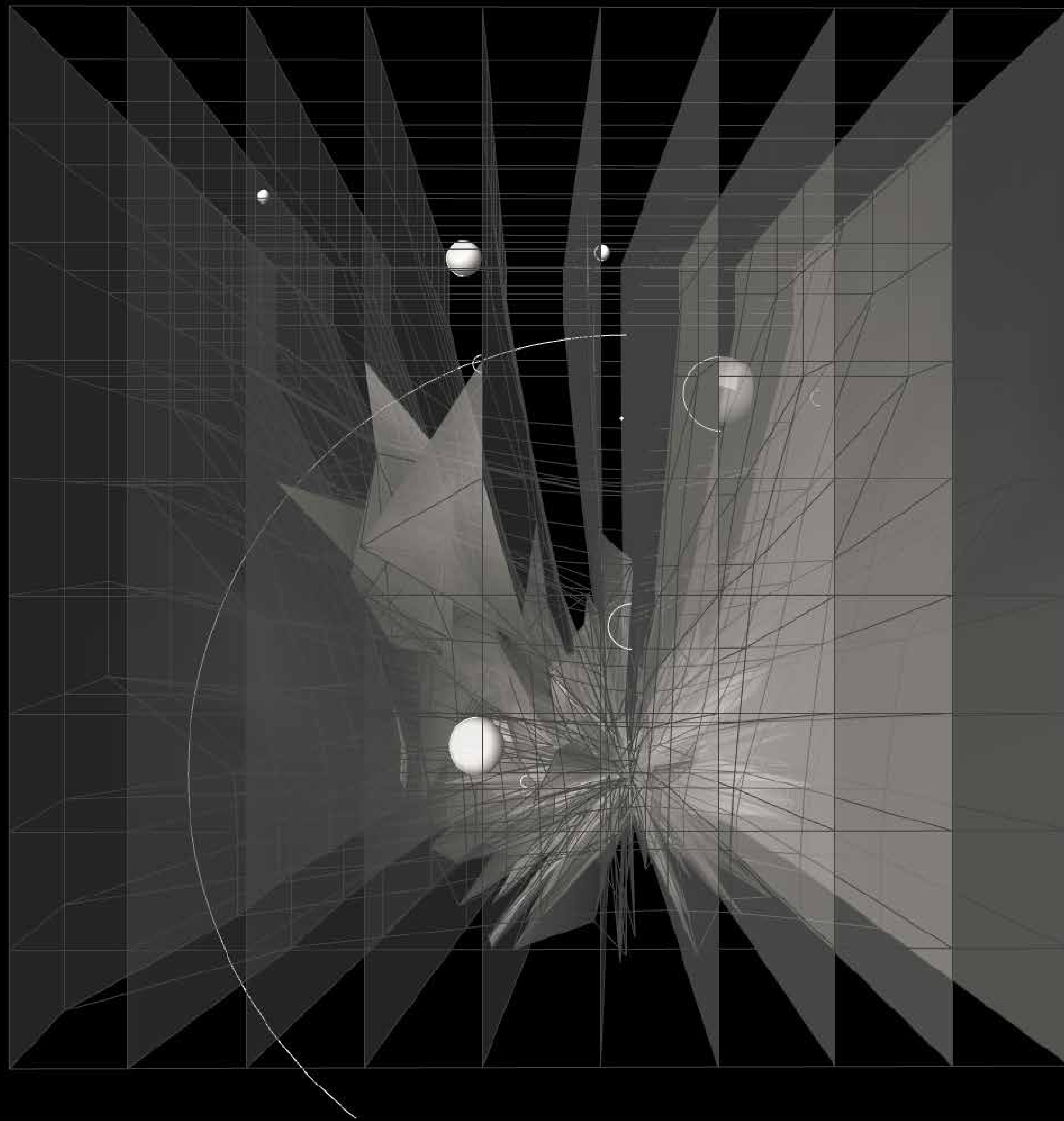


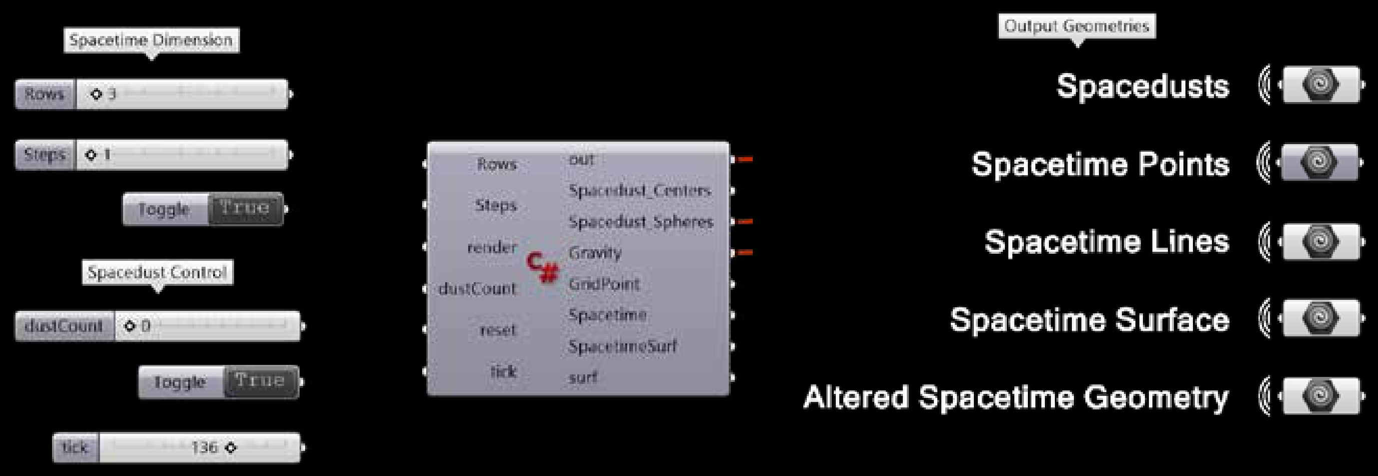
Live Result
Display of lofting with circular cross section geometry

SPACETIME
Generated University of Imaginary Space Dust

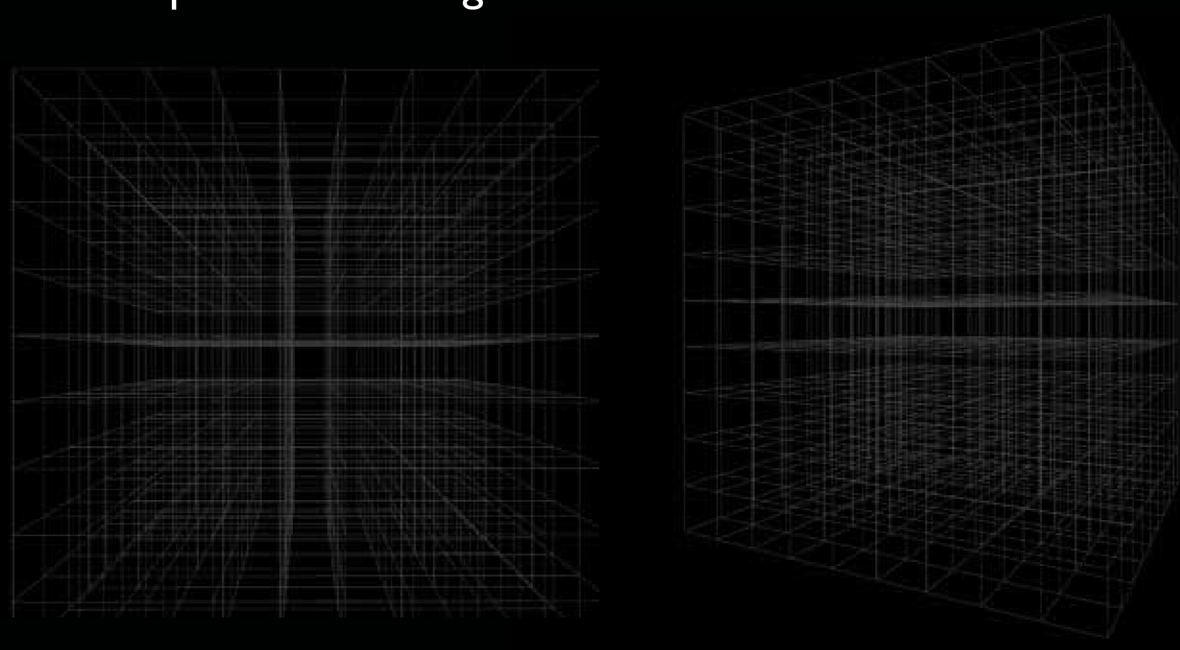
Introduction to Computational Design | GSD
Instructor: Jose Luis García del Castillo y López | October 2019
Role: Concept, Code, and Visual Development
Collaborators: Tamilyn Chen, Caleb Hawkins

Inspired by the physics of spacetime and existing art project shown on the right, we created a randomly generated grid universe with particle objects with fundamental physics attributes. As the particles expand and move around, the universe's grid structure is affected and attracted to the particle matters. Implemented with C# in Rhino Grasshopper.





Grasshopper User Inputs
Create "Spacetime" grid and Spacedusts

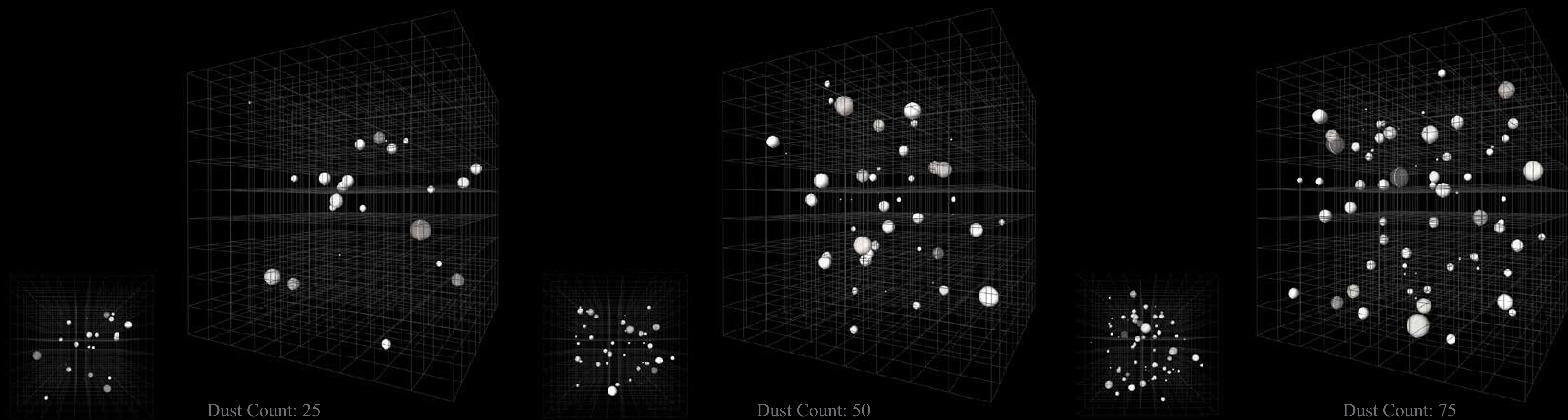


Spacetime Grid & Mesh
Graphic output of recorded user inputs

This was the creation of the universe... the Big Bang.

The first step was developing a parametric grid of points. The grid structure can be scaled by increasing or decreasing the size and steps of subdivisions.

These points were then used to create the lines that are representing the expanded surface of the Spacetime mesh.

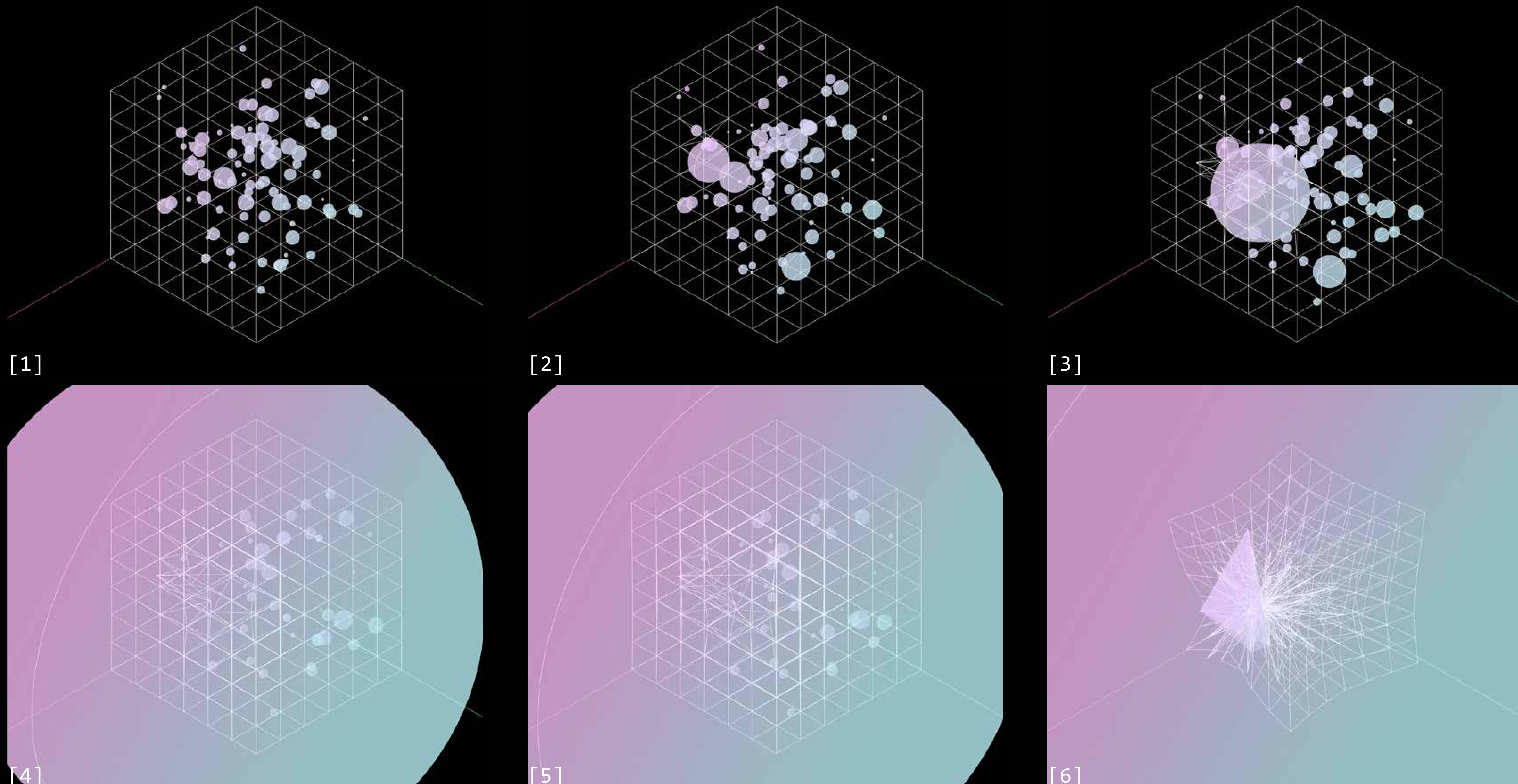


Spacedust with Gravitational Force
Random spacedust generation with different dust counts

To populate our universe, I started with the creation of mass particles. The script allows the particle count to be adjusted from 0 to 100 particles.
The size of the original particles are random between 0 and 1 and over time they grow.

The size of the particles are random at start but their size also determines the display of the particle.
The bigger the particles are, the more mass it has, and the more transparent it becomes in relation to the others.

You can see the transperency as well as the varying starting sizes of the particles.
Before the universe is sevt into motion the particles are still.

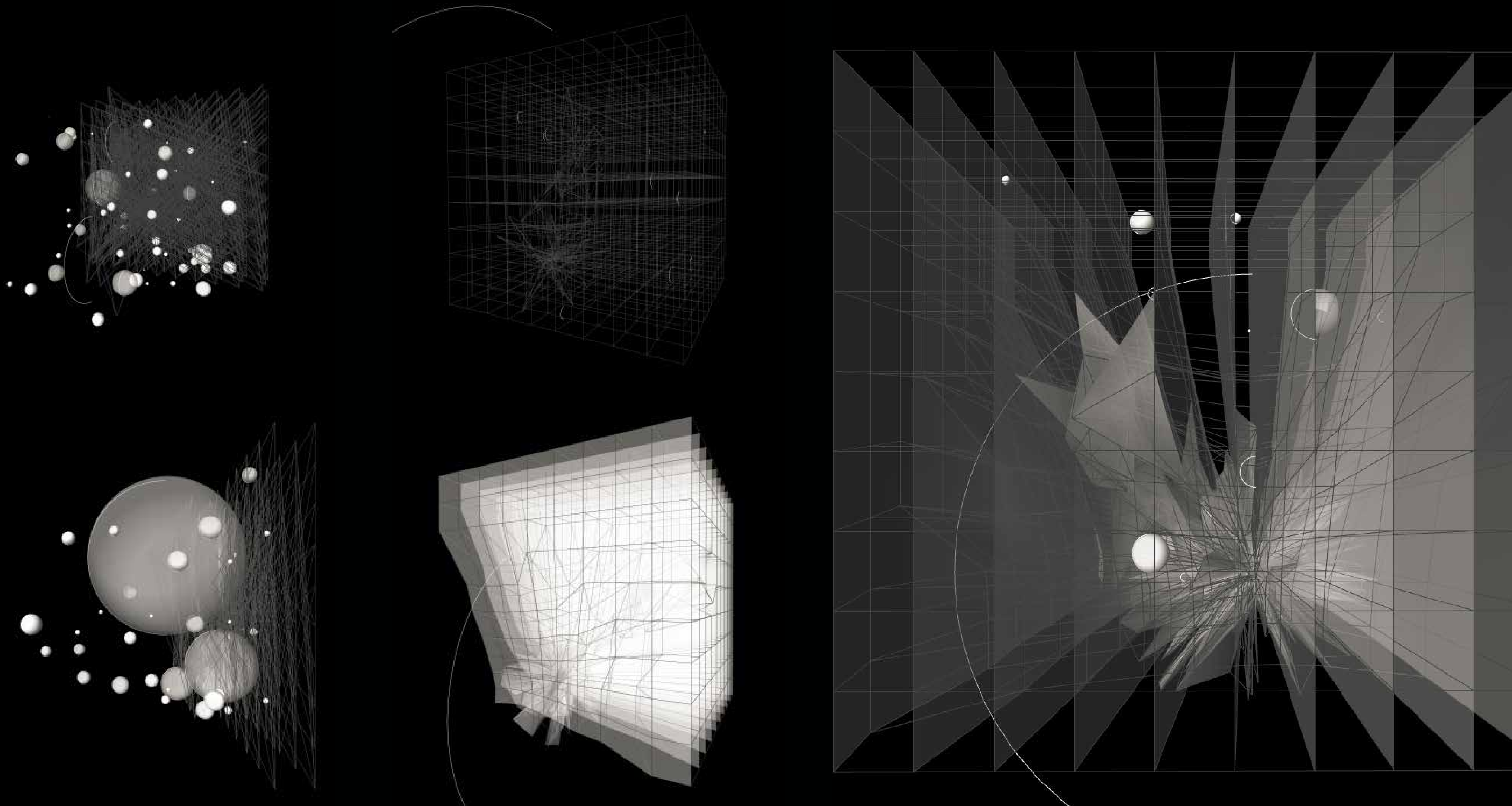


Spacetime
Spacedust and spacetime progression in time

The particles move in a random pattern within the spacetime grid. As the particles collide and absorb into one another their is one large particle thar gets the biggest, which also has the most force acting on the framework of the grid structure.

Colors were assign to the particles to echo reality's spacedust. As the universe runs its course you can see how the particles interact with each other as well as with the SPACETIME grid.

As the universe continues through time. All particles become one large gravitational force... A black hole!
The collasped spacetime grid create new surface geometries for potential new universe to populate on.



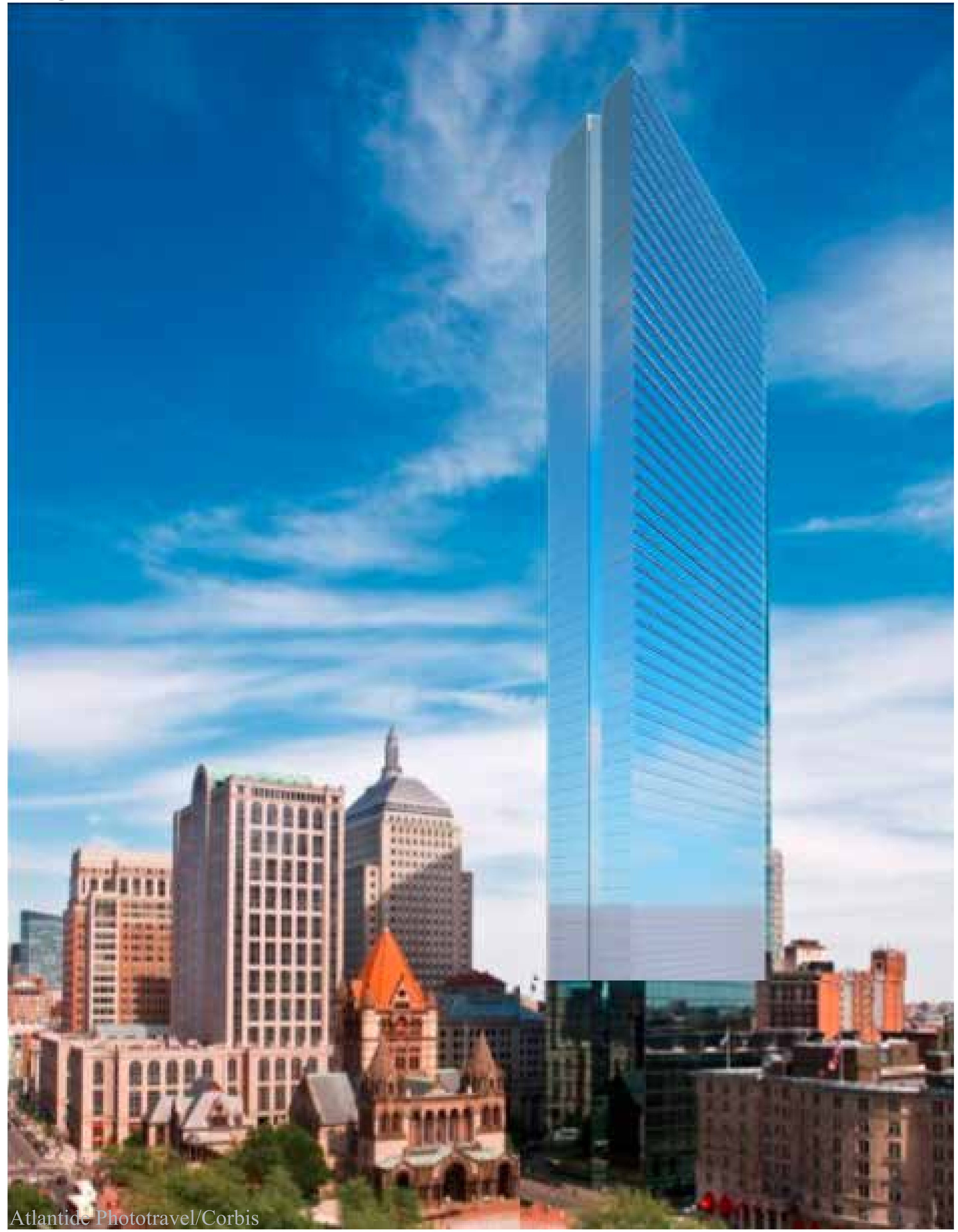
Broken Universe
Unexpected component behavior

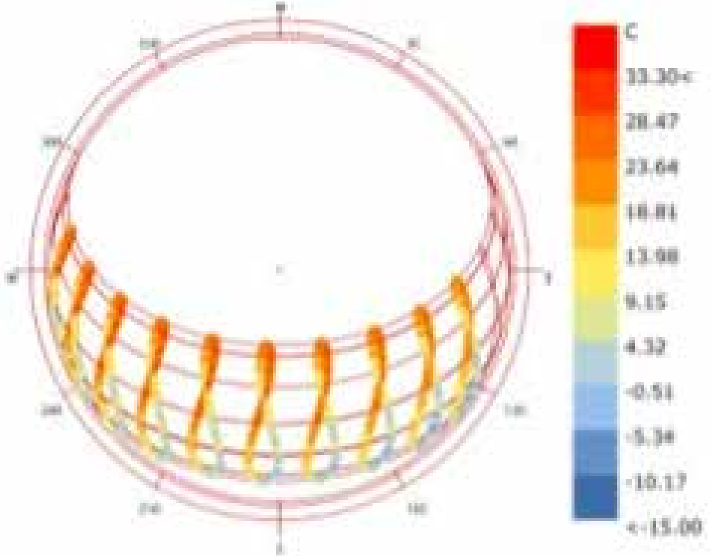
As time progress infinitely, the spacetime grid is seem to compress and collapse in unexpected ways. Once there is a particle grow to a gigantic size, its influence on the grid seems to grow tremendously and creates surprising geometries.

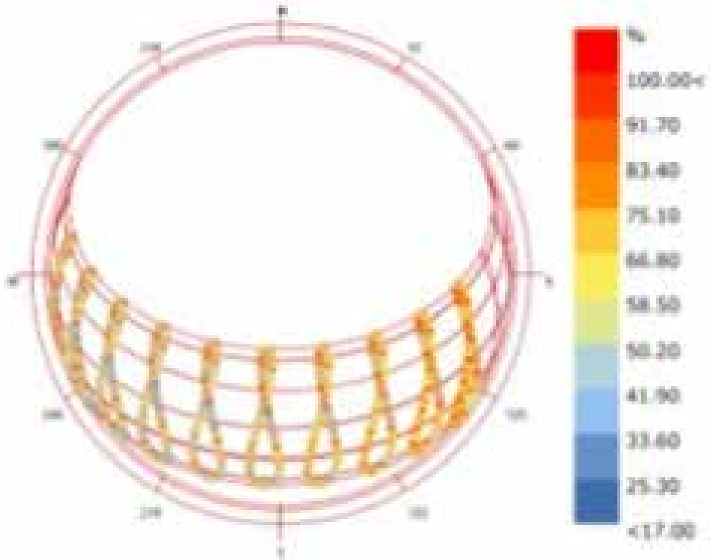
REDESIGNING FACADE
200 Claredon Street for better energy performance

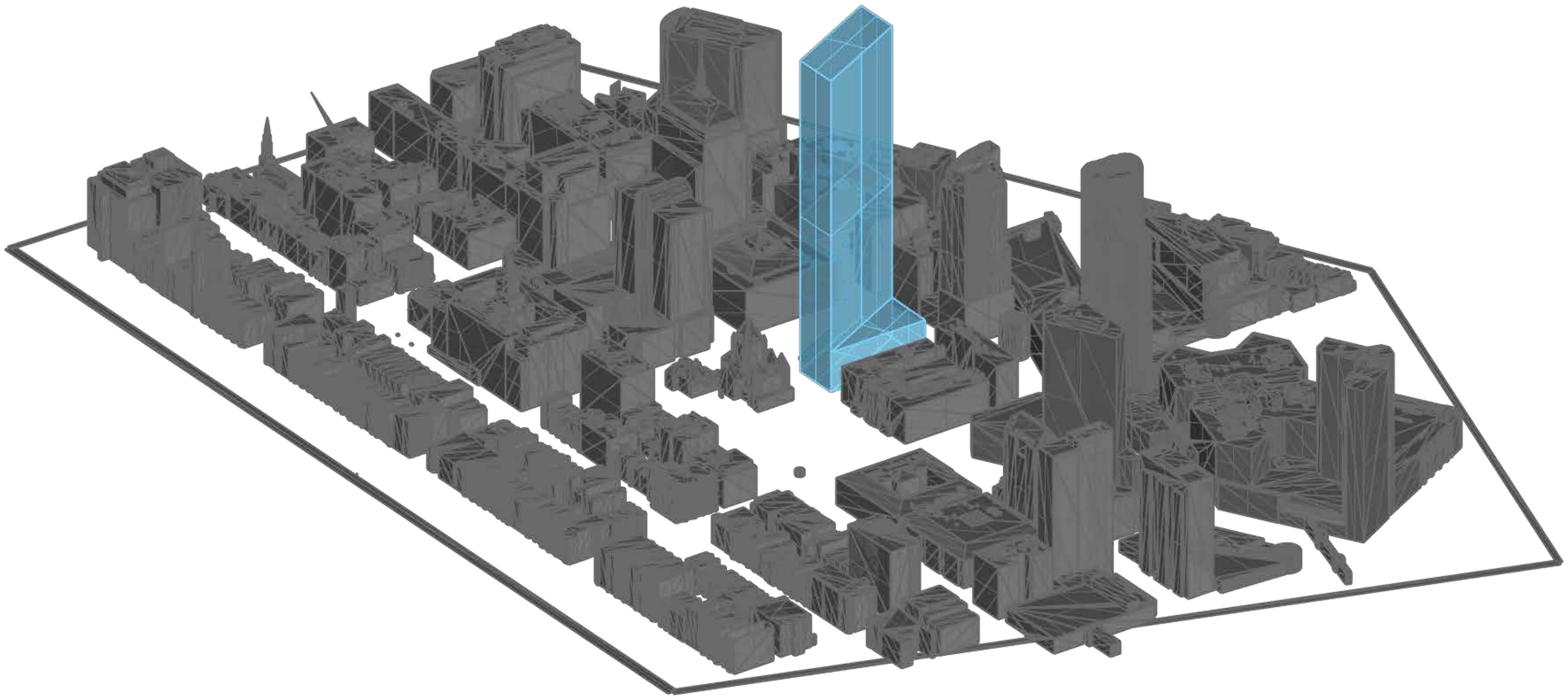
Optimazing Facade Performance | GSD
Instructor: Andrea Love | September 2019
Role: Climate, Simulation Analyst, Visual Development, and Design
Collaborators: Moulshree Mittal

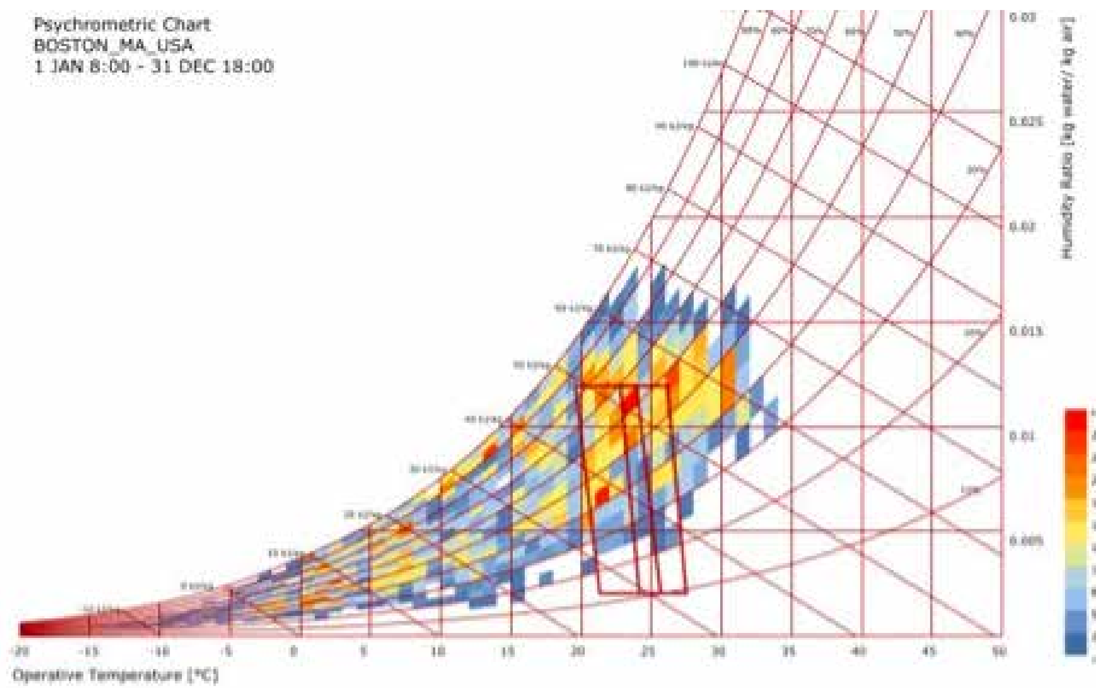
Analyzed the current facade performance in Boston Climate and identified target issues: glare issue, intensive cooling & heating load, lighting energy, and hostile street environment. Redesigned to improve targe issues based on simulation feedback without changing overall aesthetic or disrupting views.

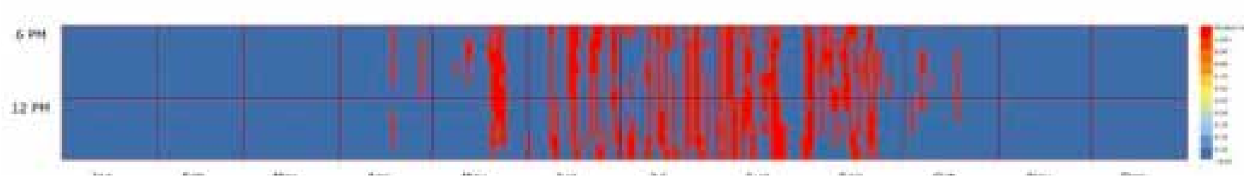

Atlantide Phototravel/Corbis

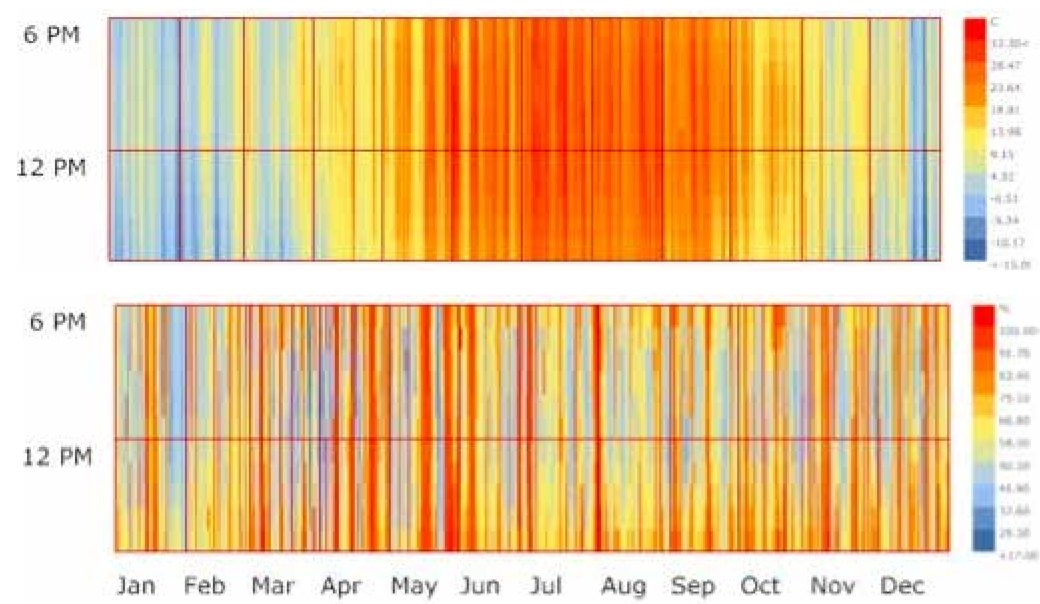


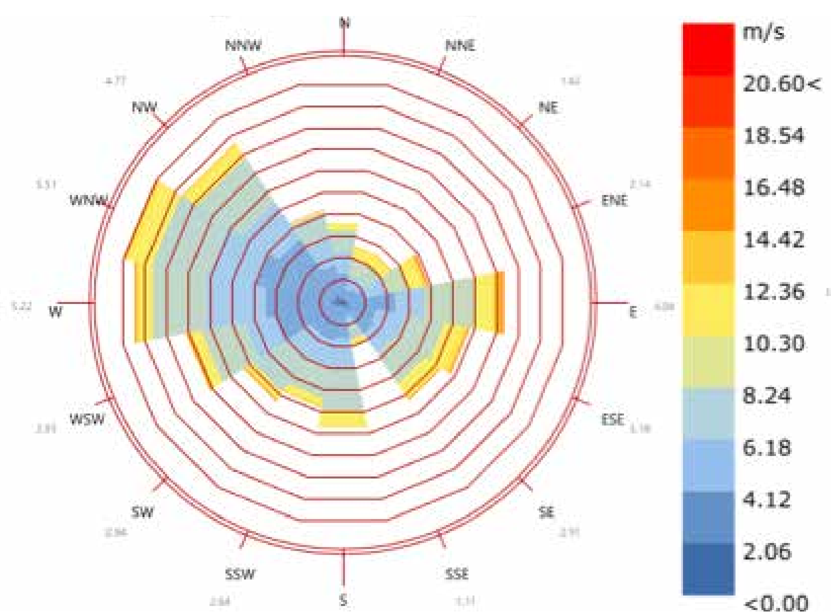






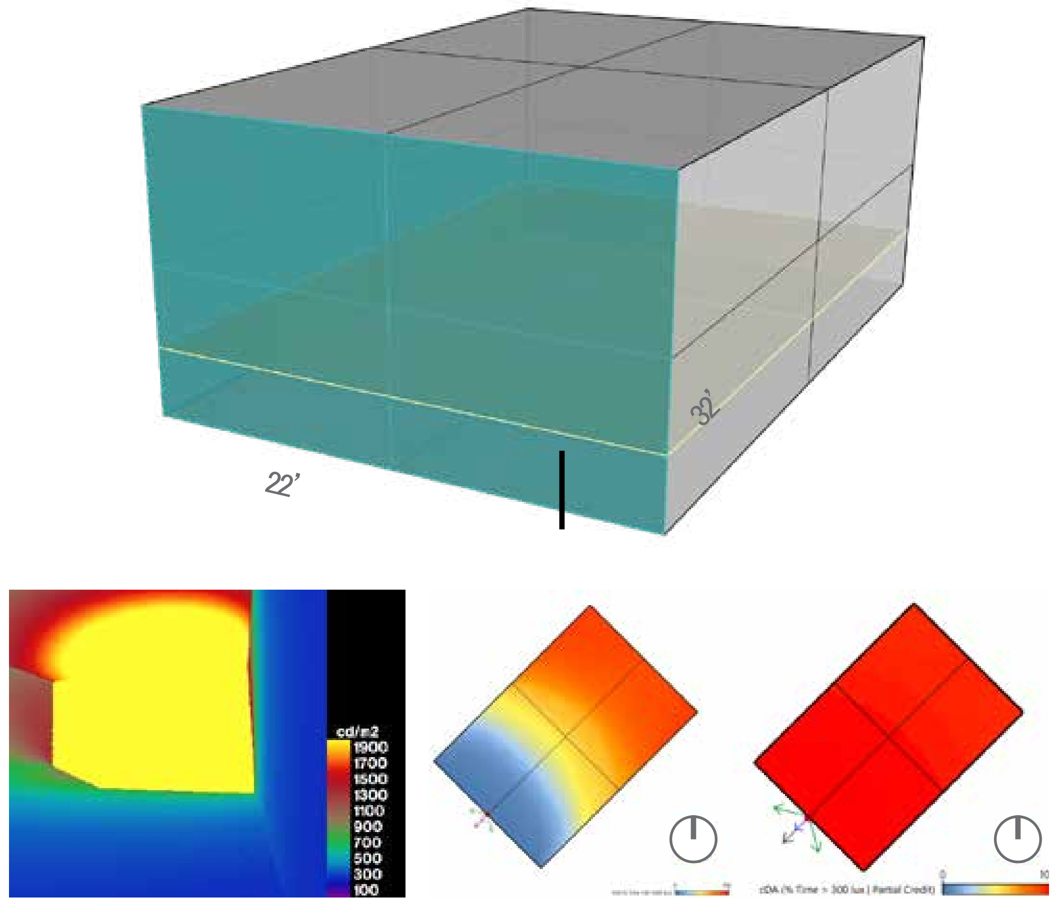




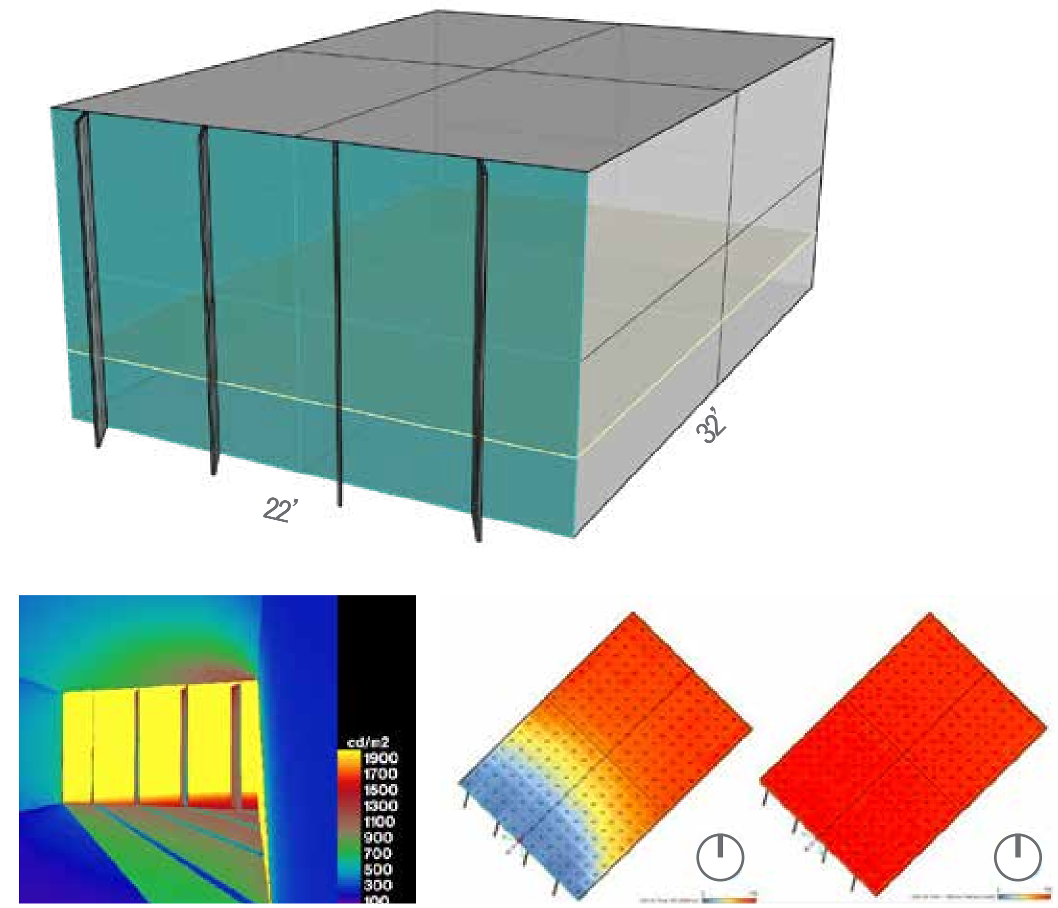


The previous named John Hancock Tower is located at Copley Square, Boston, where the climate includes hot & humid summer days with cold & dry winters. Boston also observes a great deal of wind, calm for roughly 23 hours per year. As seen in our adaptive psychrometric chart adjusted to working hours and activities, occupants are only comfortable roughly 15% of the time.

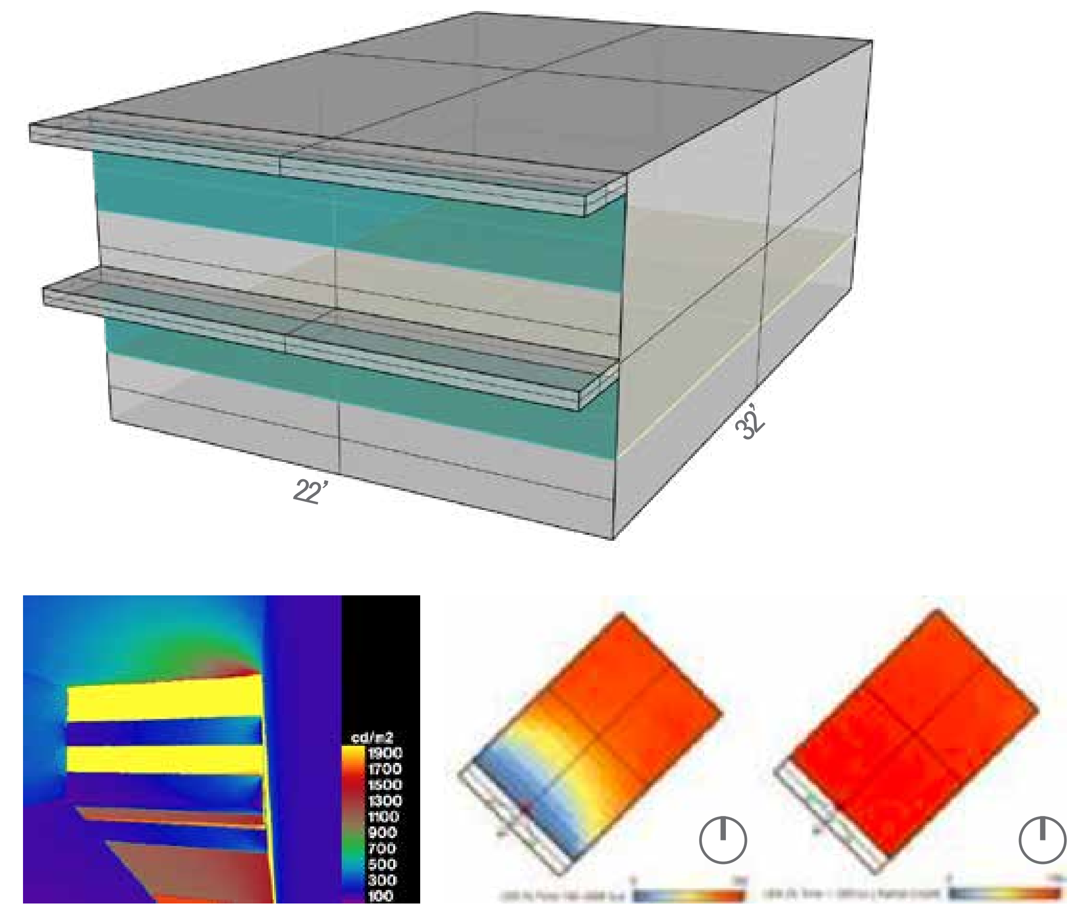
The entire plot size is roughly 247 x 340 feet. The tower is a 62-story class A office building with lower levels dedicated to lobby and cafe/restaurants.



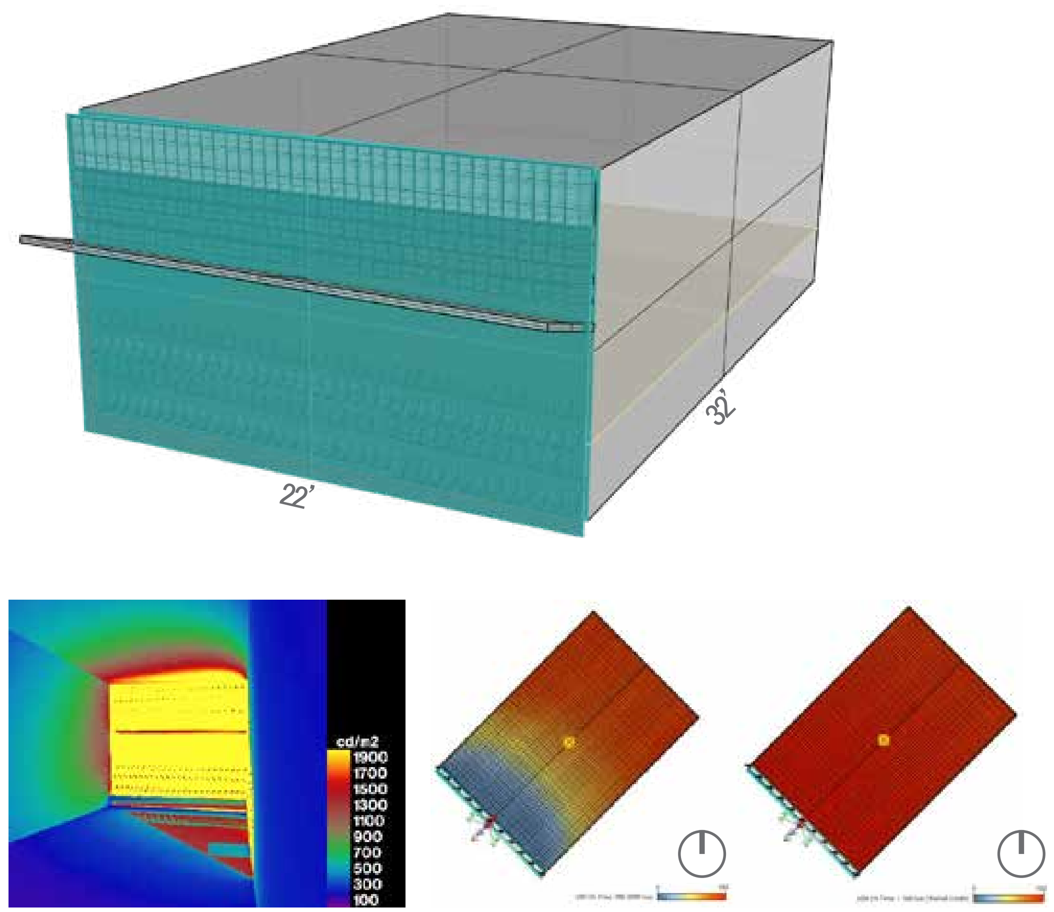
Unaltered



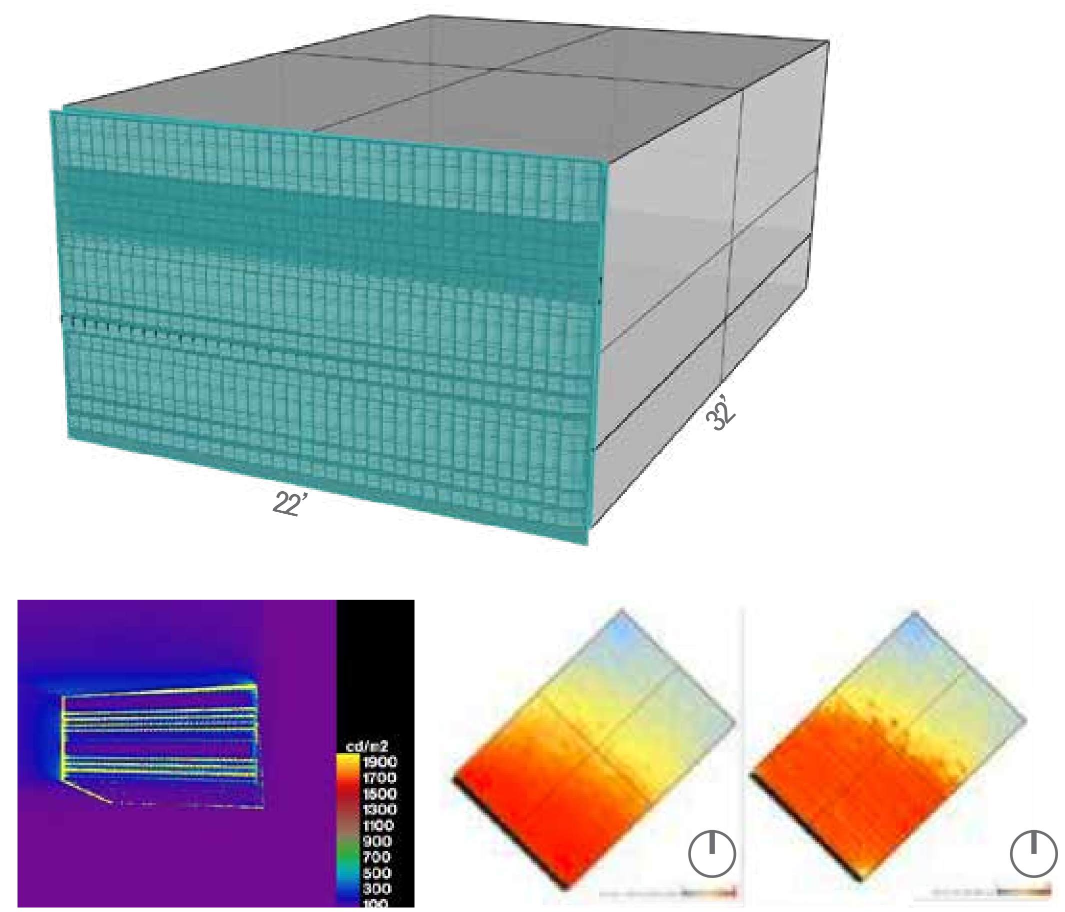
Iteration 1



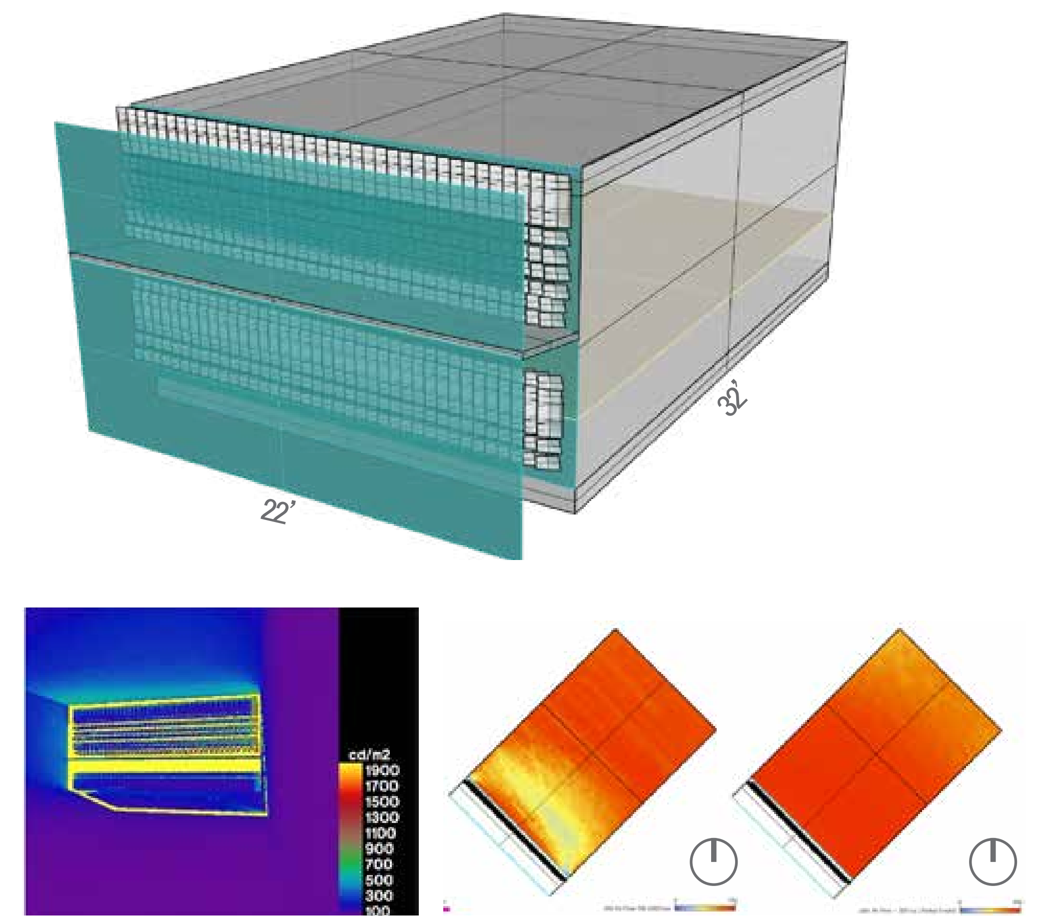
Iteration 2



Iteration 3

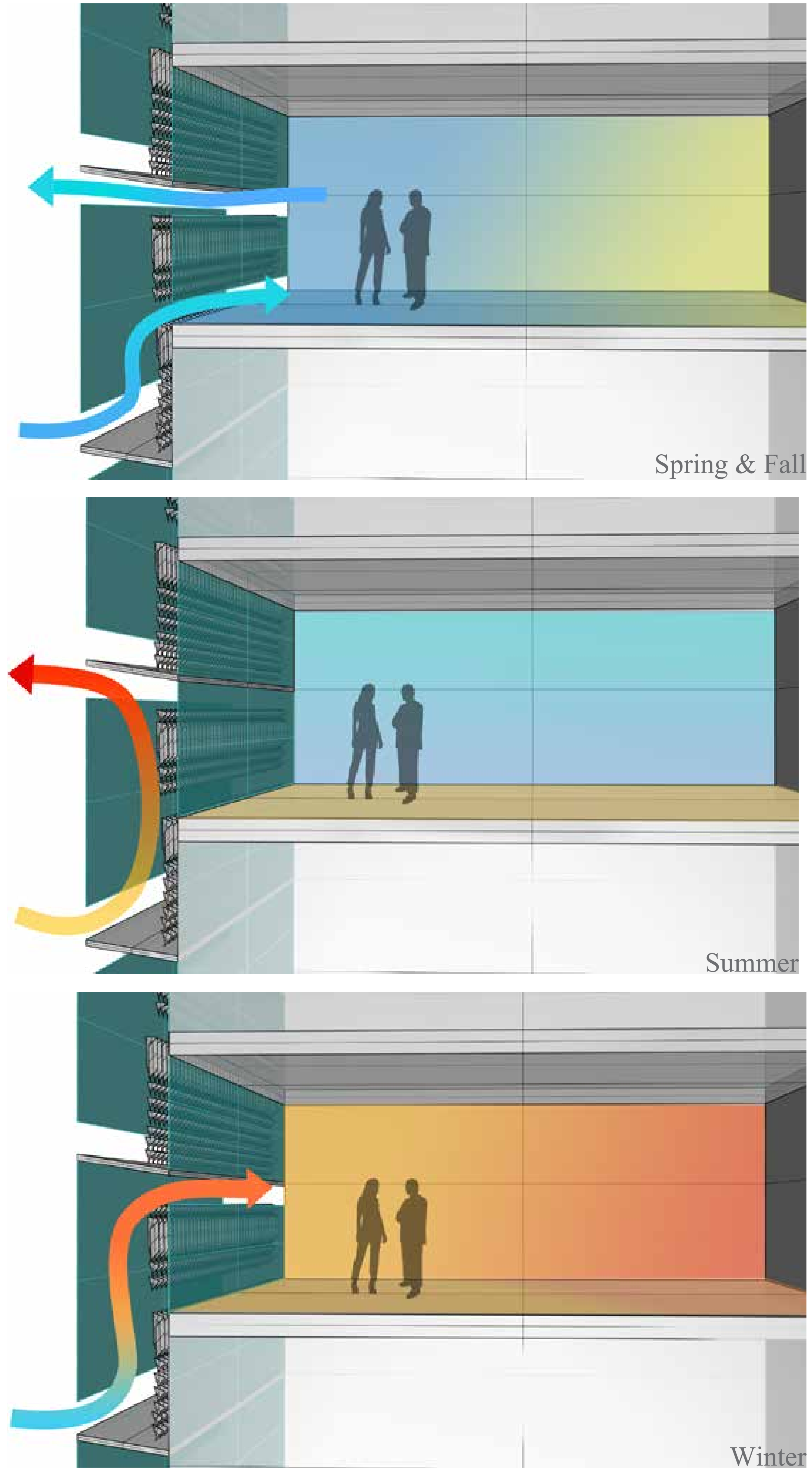


Iteration 4

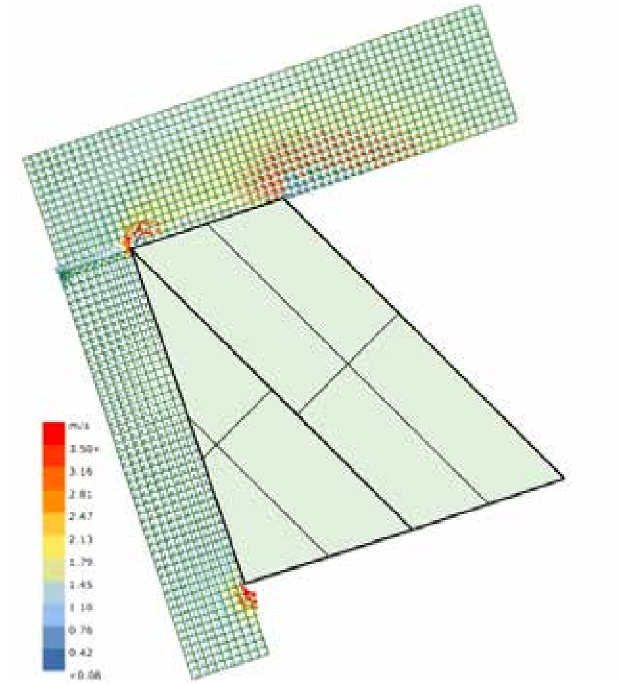


Iteration 5

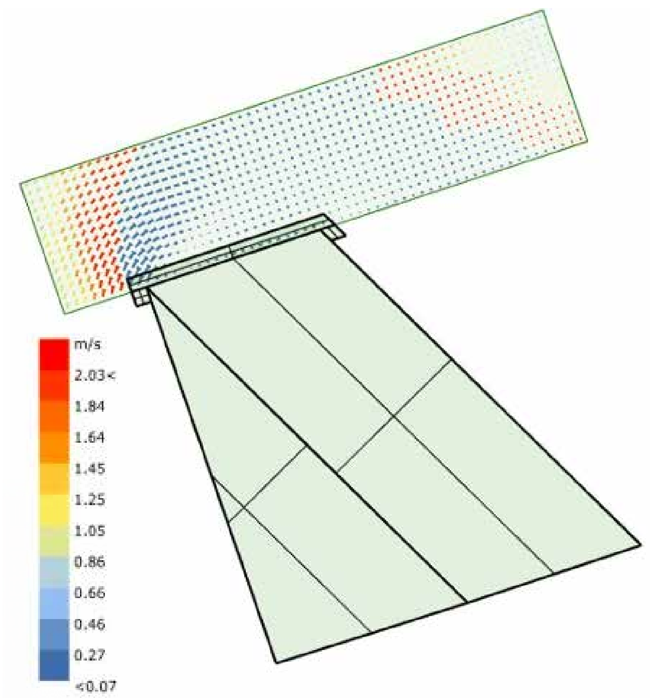
Daylight & Glare Simulation Analysis
Shoebox models design iteration to



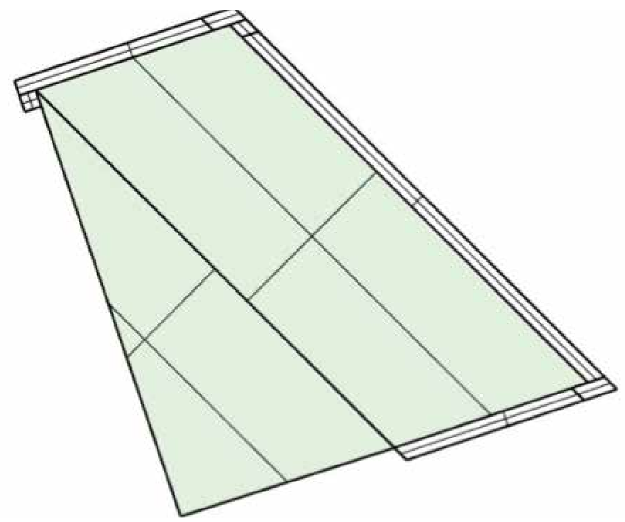
Natural Ventilation Window Schedule
Double skin design to leverage benefit of natural ventilation



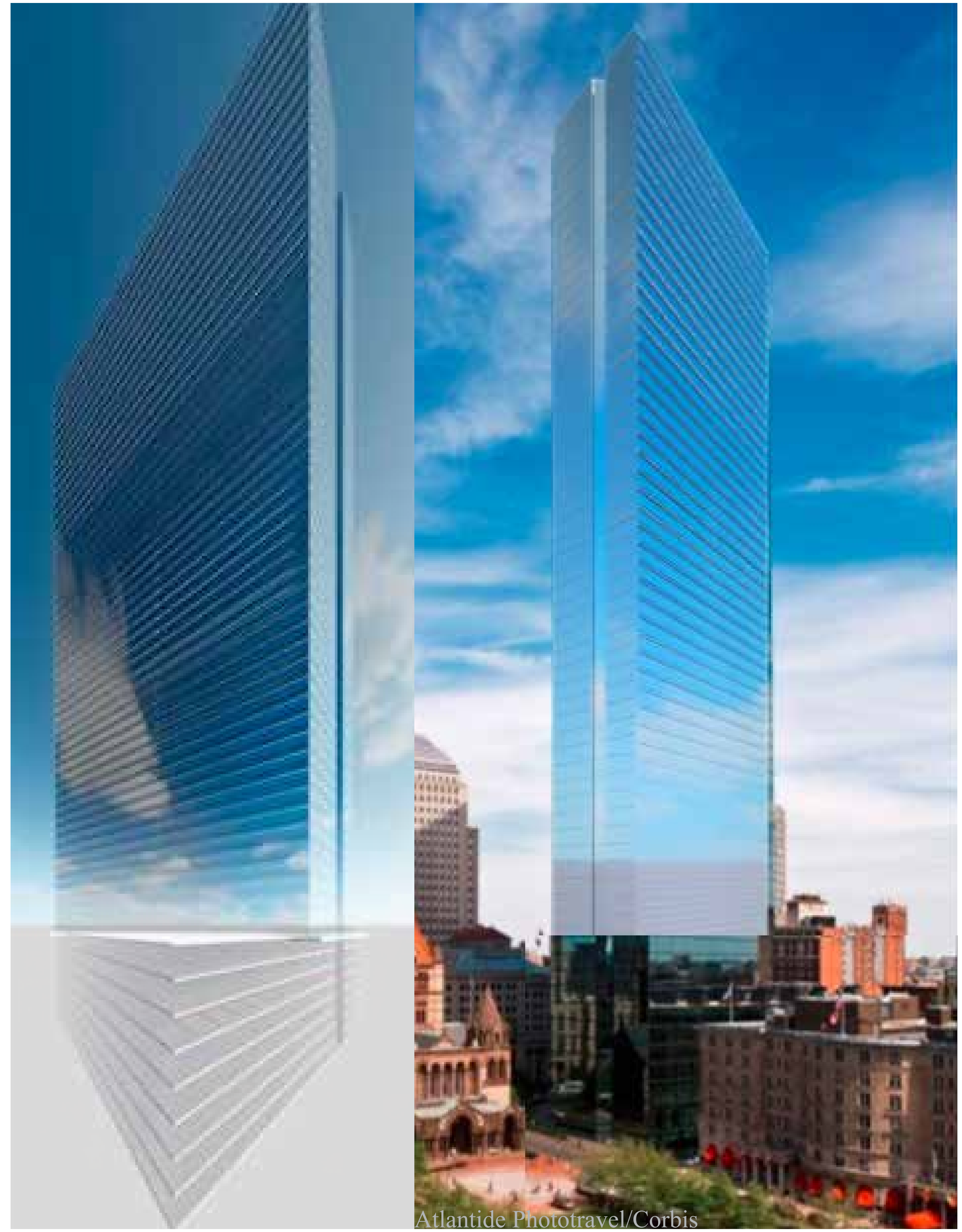
Unaltered design, 3m/s west wind



Canopy design, 3m/s west wind



Proposed canopy



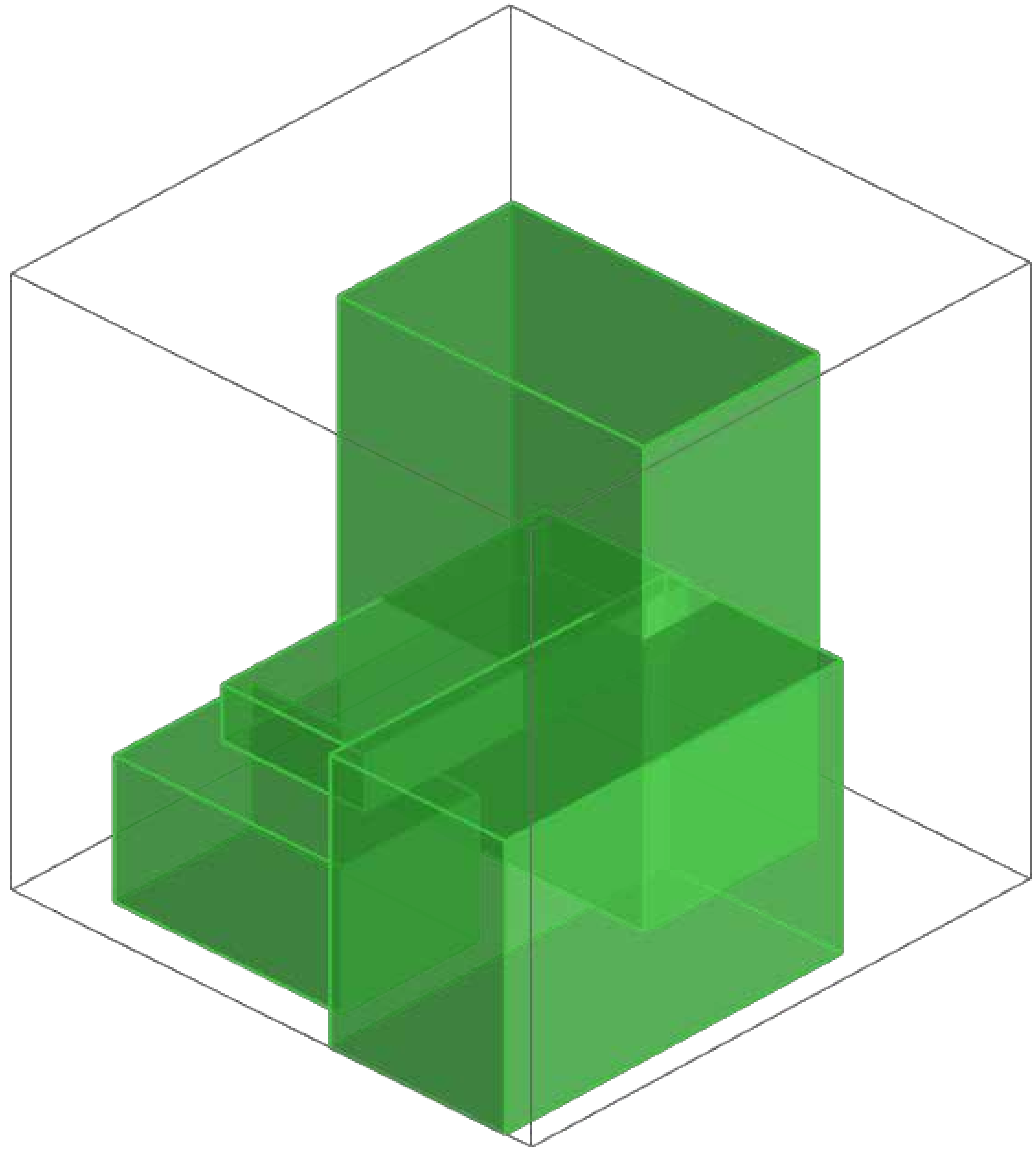
Final Rendering with Open Windows
Natural ventilation, daylight, glare prevention, and canopy

GREENFORM

Accelerated Massing Interation for Sustainable Designs

Introduction to Computational Design | GSD
Instructor: Jose Luis García del Castillo y López | November 2019
Role: Concept, Code, and Visual Development
Collaborators: Kezi Cheng, Moulshree Mittal

Iteratively generating massing blocks and evaluating environmental scores associated. User inputs include site outlines, latitude, hight boundary, ground coverage, wall number, rankings of environmental factors, and iteration number. Implemented with C# in Rhino Grasshopper.

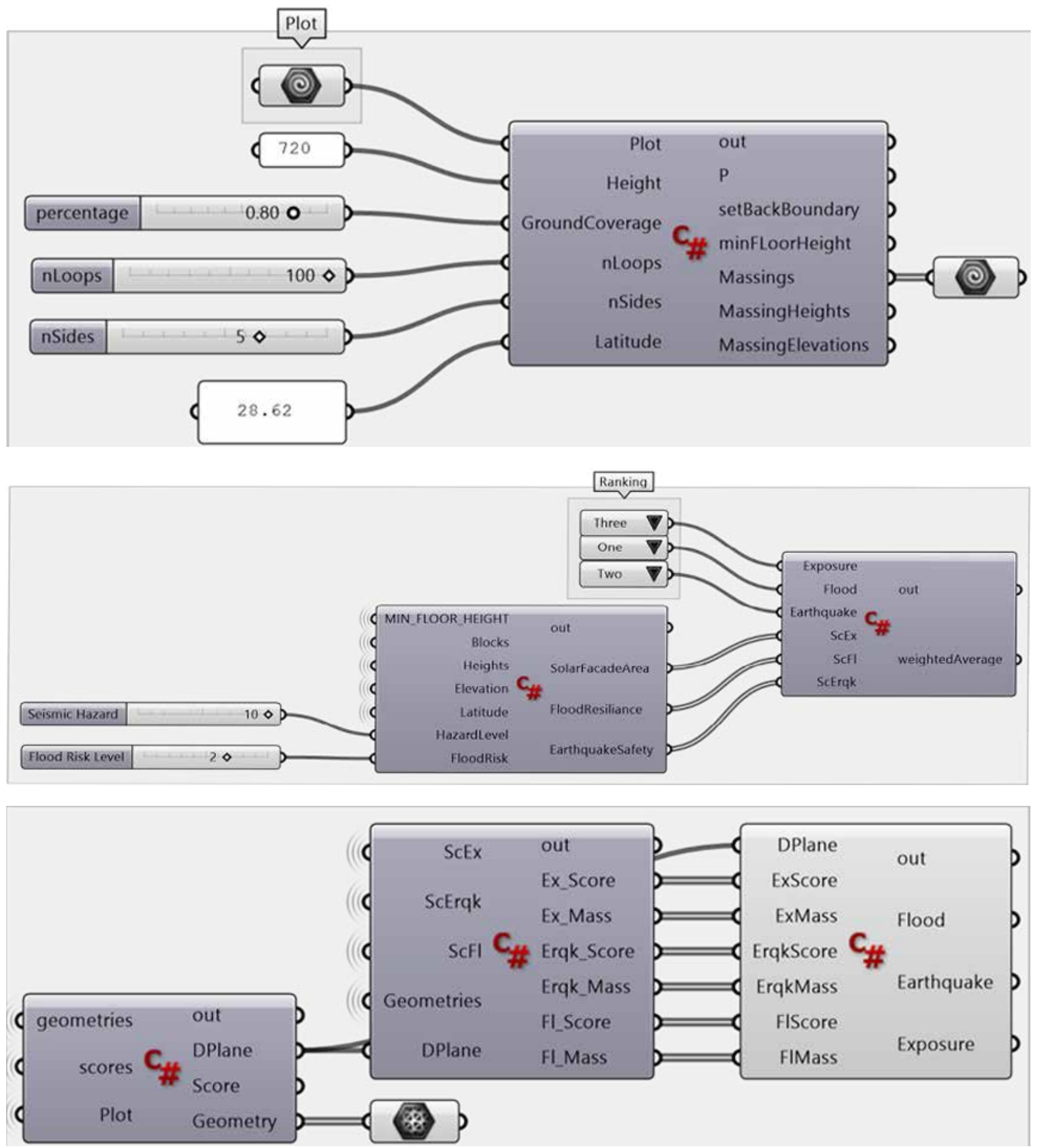


Inspired by the difficulties to generate sustainable design and iteration process with simulations, GreenForm incorporates environmental factors such as solar exposure per surface of the building as an essential factor of design concept, makes it easier and quicker for designers to get through the initial massing phase.

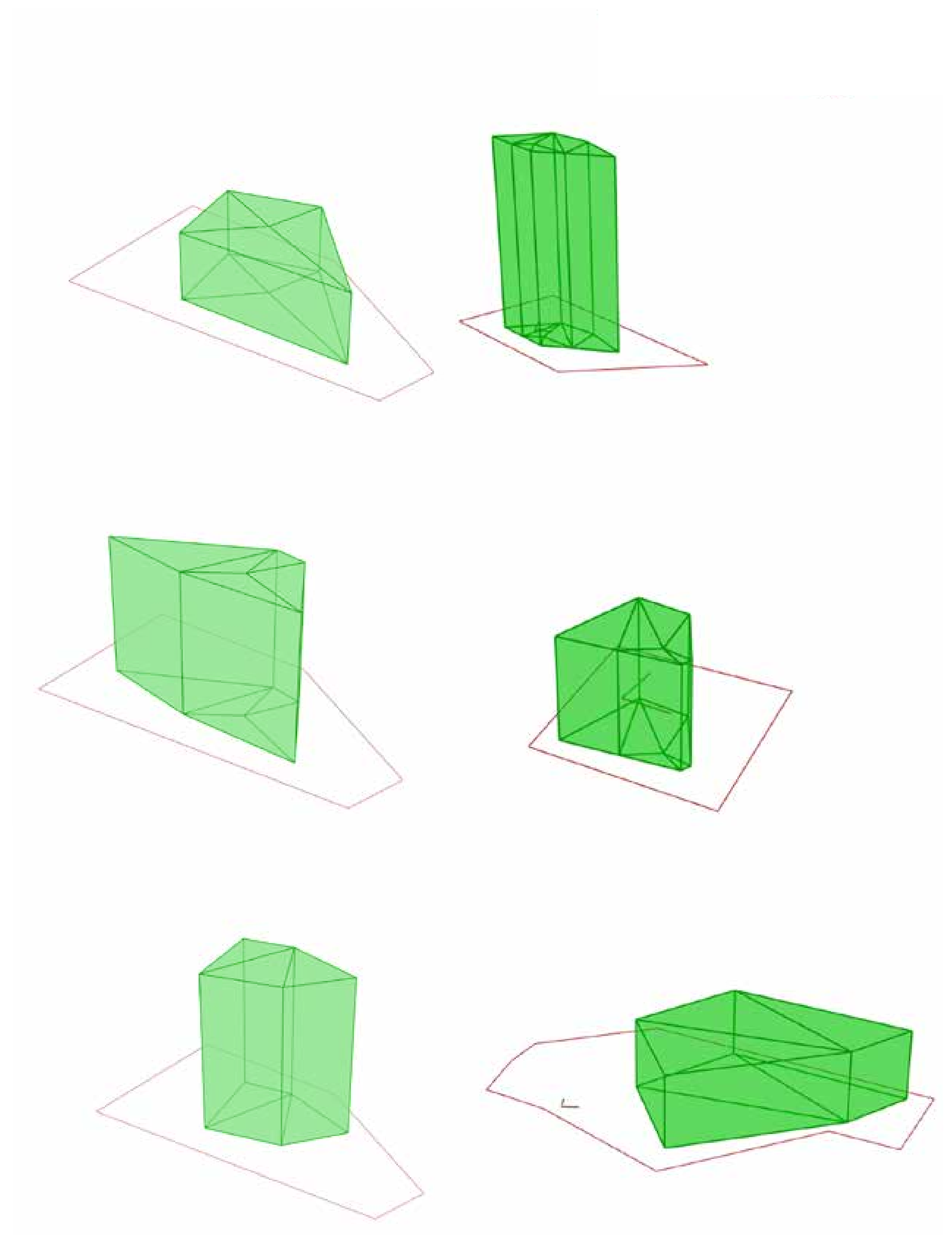
The grasshopper component generate randomly geometries based on user inputs, such as plot geometries, height limitation, by-laws, and ground coverage. They can also choose how many massing geometries generated. Here are six of the randomly geometries generated on the same plot.

The script anaylze the inputs and output a list of randomly generated massing geometries. It generates random mesh for ground level, then create solid mesh by extruding upwards to random height with random elevation height.

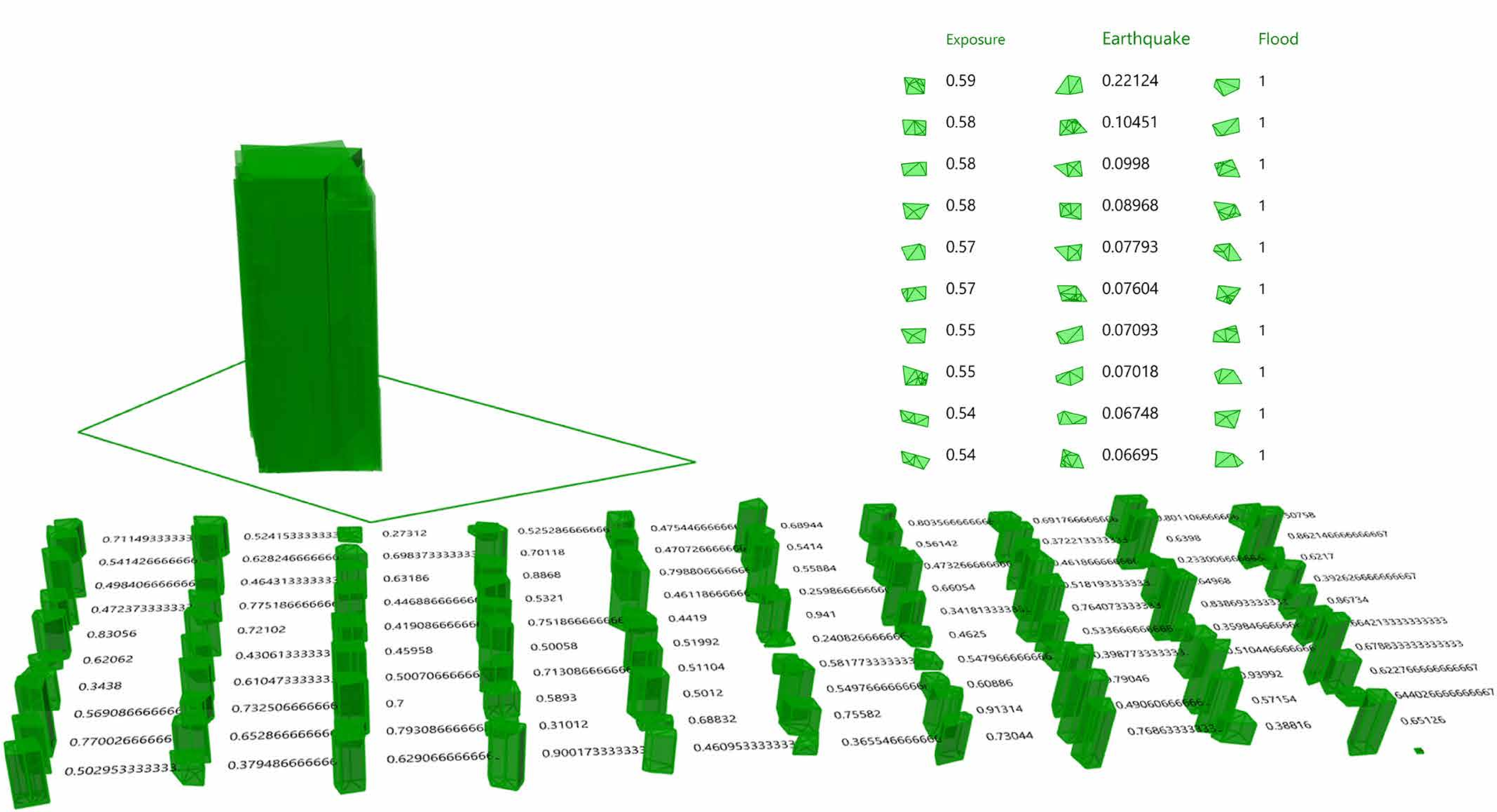
The final comprehensive score of each geometry is based on the user-ranked environmental factors, such as seismic hazard level and flood risk level of the location.



Grasshopper Script
Custom scripts for generating massings, evulating massing, and displaying scores



Massing Output
Random massing on the same plot & different plots



Final Display

Display with random custom massing overlay, top ten massings in three catagories, and total weighted scores

After the massings are generated, users then can input the seismic hazard level and flood risk level of the location. They can also rank the importance of each environmental factors for generating a comprehensive score. The script evulates each of the massings base on three environment factors: solar exposure, flood resiliancy, and earthquake resiliancy. This script outputs a list of weighted total score as well as lists of individual scoring catagories. The result is then displayed on Rhino canvas in a table-like manner. All the total weighted score with associated geometries will be placed together, as well as top-ten best geometries for each individual catagories.

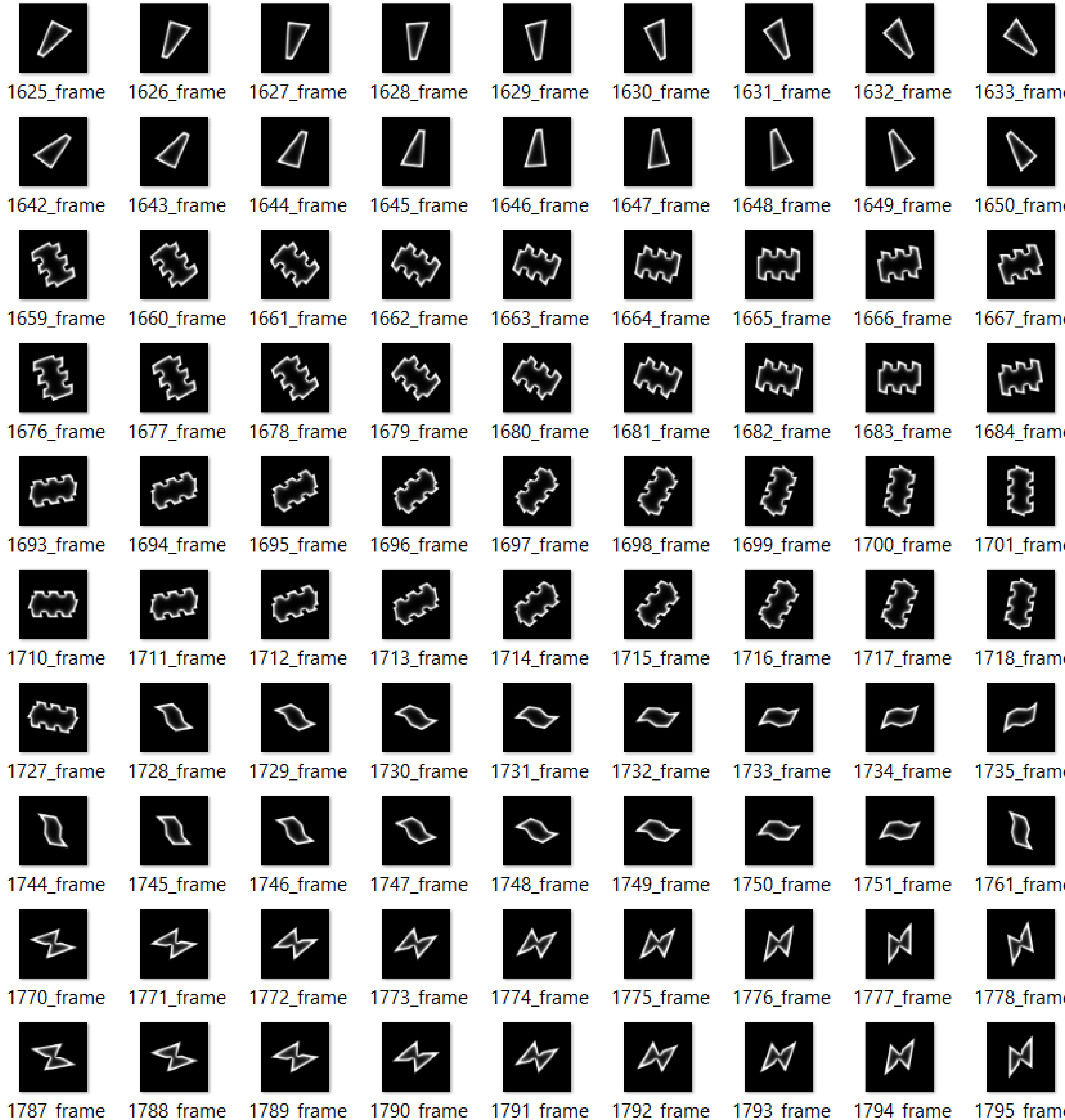
SMARTBEE

Daylight Factor in Office Space with Machine Learning

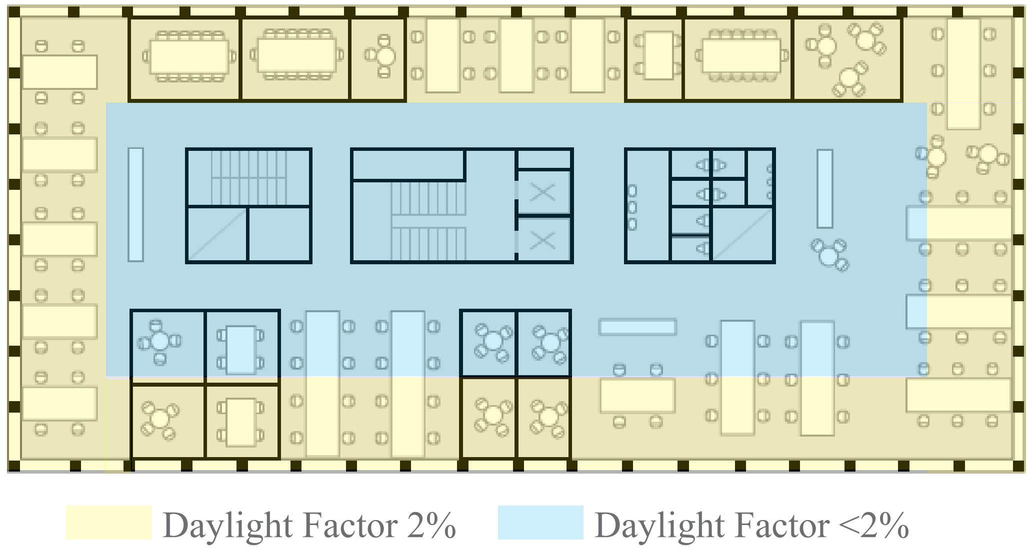
Creative Machine Learning for Design | MIT

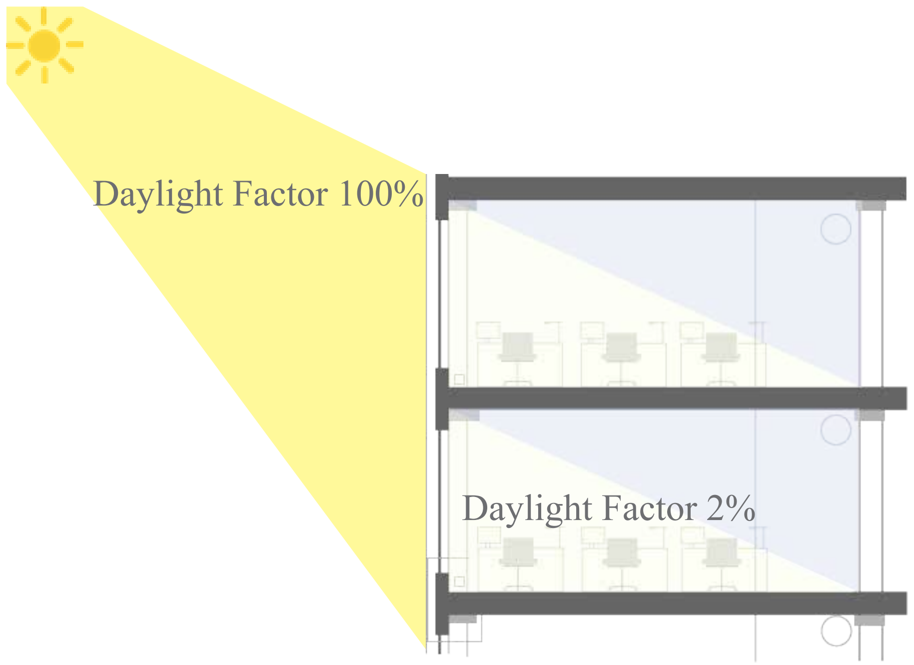
Instructor: Renaud Danhaive, Caitlin Mueller | May 2020

Estimating daylígh factor in office space by providing floorplans to the machine learning algorithm pix2pix. Using Python, Honeybee, and Ladybug.

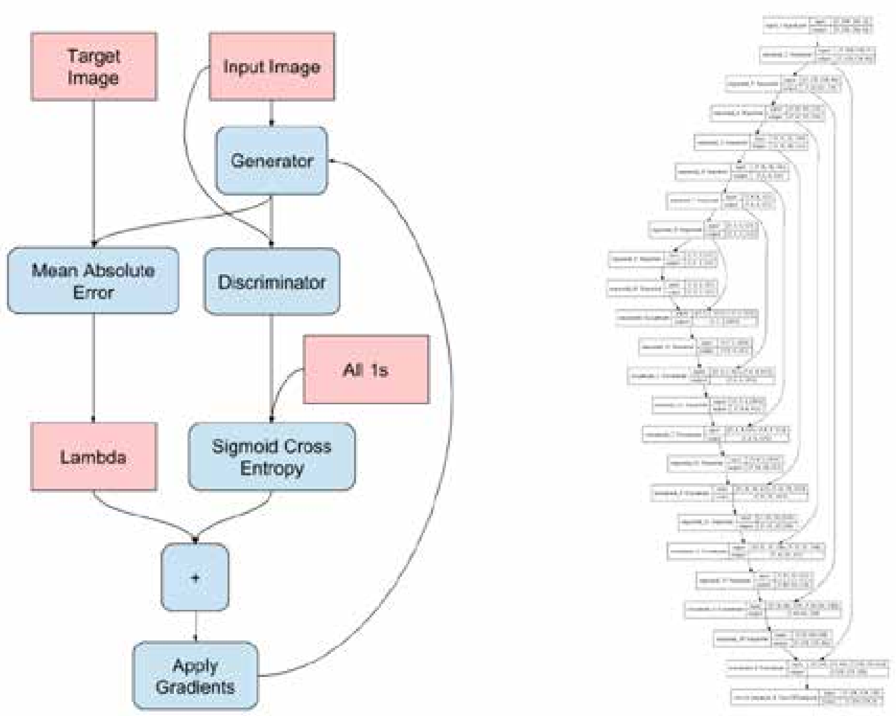


Smartbee | Creative Machine Learning for Design

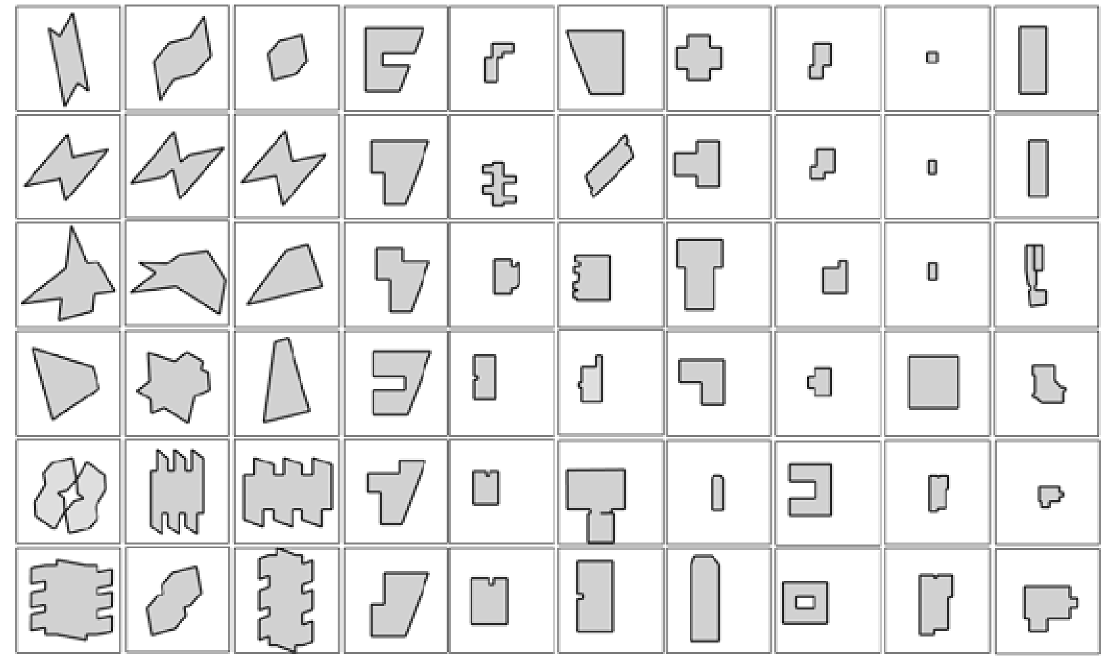


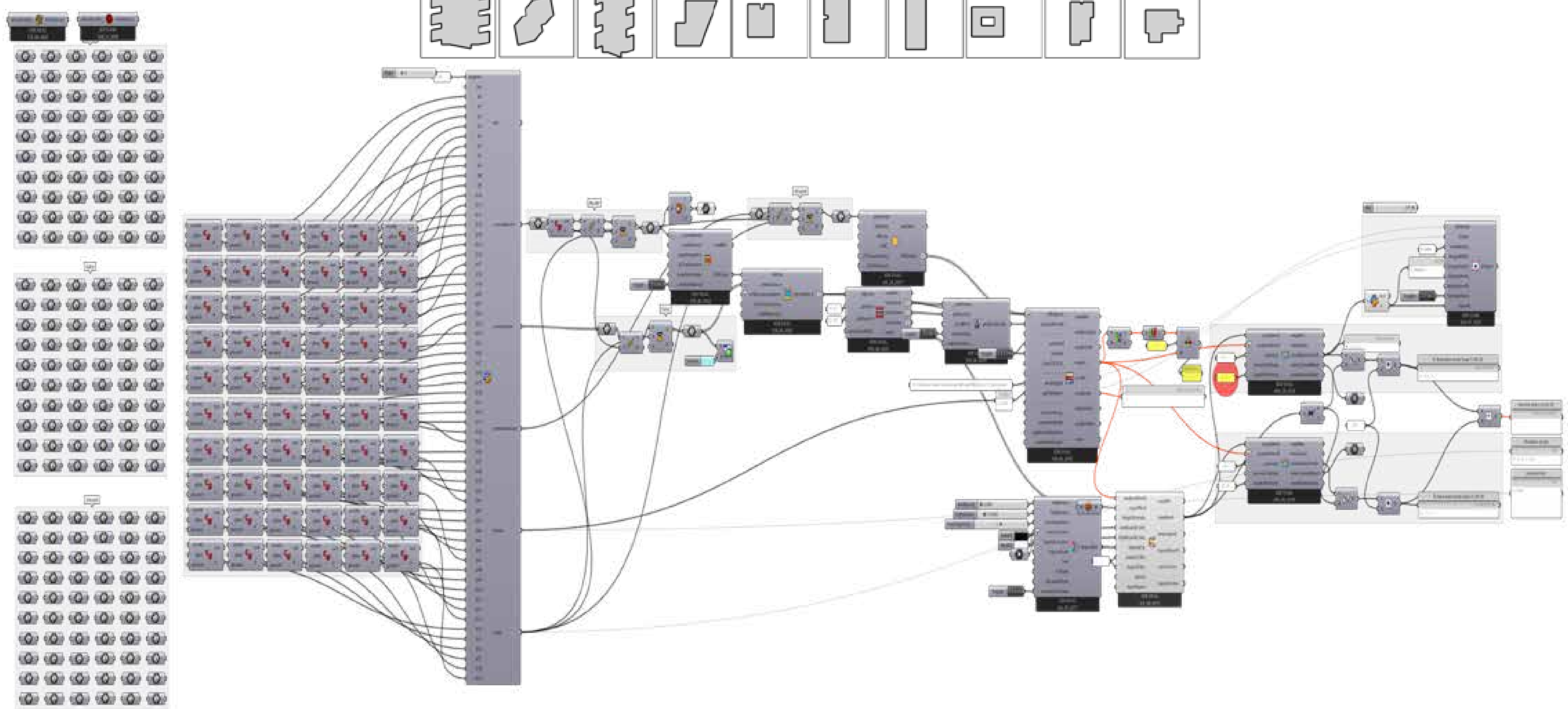


Typical daylight factor in a office layout



Pix2Pix Algorithm Architecture (Isola)



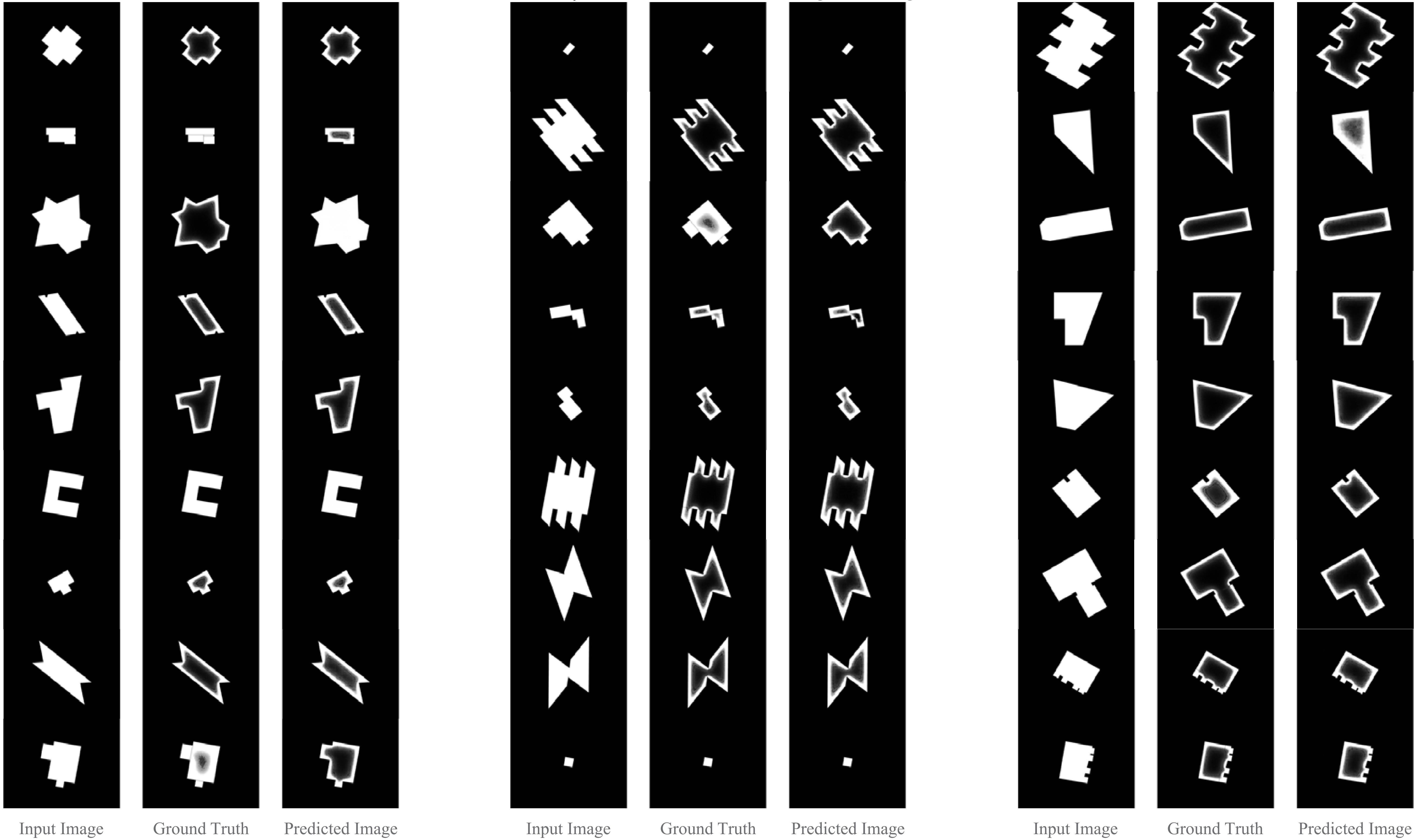


Training Data Generation Workflow

60 base geometries rotated and simulated 2160 Daylight Factor using Honeybee

Smartbee the algorithm takes floorplans or building footprints as inputs and predicts the space's daylight factor without any simulation downtime. The daylight factor is used as a metric in Europe to evaluate daylight quality in buildings - DF 2% for office space. And machine learning algorithms, such as pix2pix, can help negate the simulation time that often creates a barrier for design firms.

To generate the daylight factor simulations, I created 60 individual office models in Rhinoceros. Some of them are existing office building floor plans. For the algorithm to have consistent learning, all models are set to have the same floor height, WWR ratio, and without shading devices. Each of those base models is passed through a grasshopper script to rotate for 10 degrees, making a total 2160 models including the base models. Then Honeybee analyzed the daylight factor of each of those models and results are saved in images.



Final Results

Comparison layout with input floorplan, ground truth simulated results, and predicted result by machine learning algorithm

This project aims to investigate the possibility of implementing the algorithm for building performance prediction. Using daylight factor has the initial metric with a 2160 dataset, currently this particular implementation still does not output reliable prediction with the average MSE below 1. However, this result could be stemmed from the limitation of the small dataset. All 60 building models are drawn individually to possibly have predictions close to the buildings in reality and provide more design diversity. Using parametric design could be a possibility to create more building models. The model can also be trained with a different training and testing percentage distribution as well as with different epoch size.