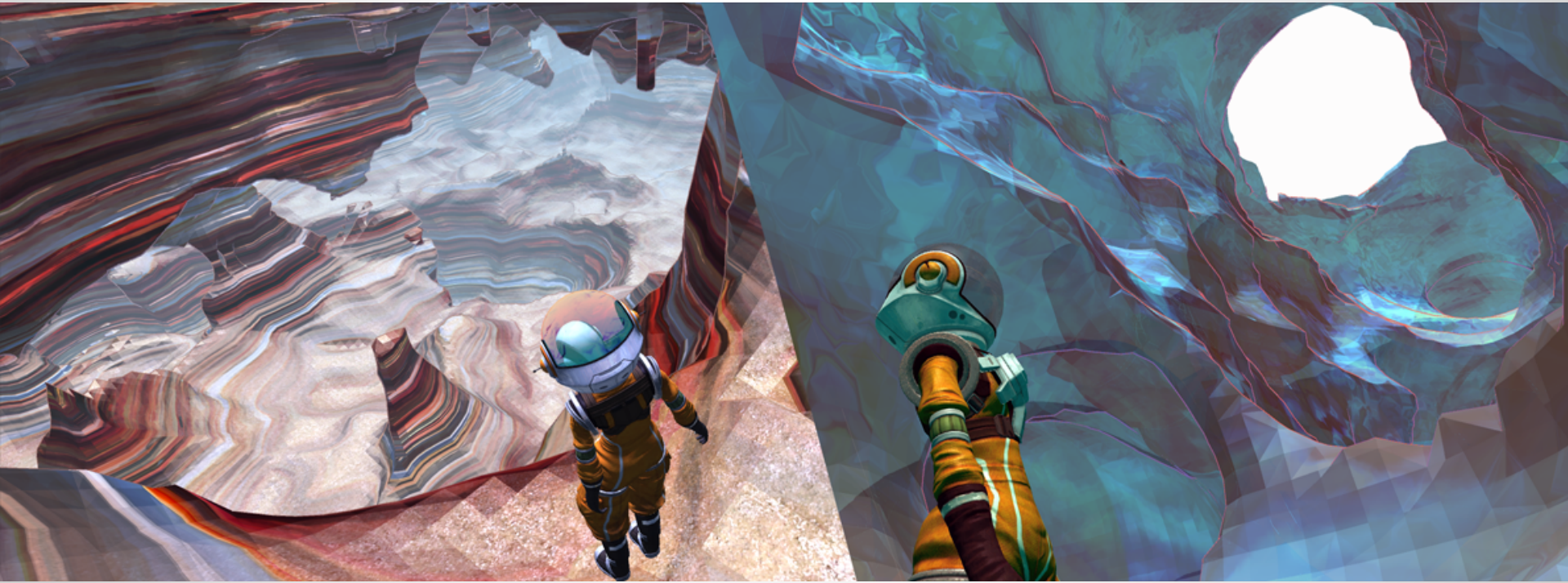
Procedural 3D Cave Generation



Our Goals

- Compelling organic cave terrain.
  + believability and expressivity (emergence)
  - reliability
- Gameplay viability.
- Customizability.

Challenge:

Emergence is unreliable; find balance to maintain immersion.
Constrain expressivity to natural patterns.

Our Research

- Existing terrain generation methods

  Mesh distortion:

  - Heightmaps, Vectors, Midpoint Displacement.

  Voxel modelling:

  - Noise functions, Cellular Automata, L-systems, Meta-primitives.

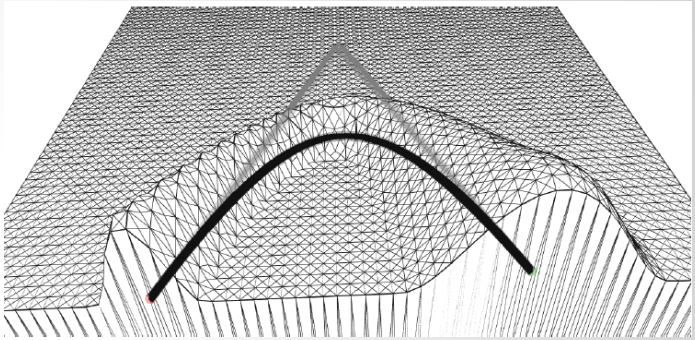
  Rendering:

  - Isosurface extraction, Procedural Solid Texturing.

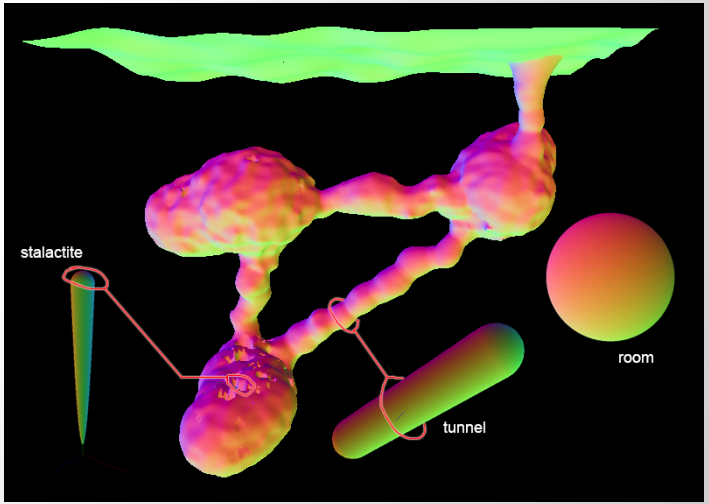
- Natural Geologic Phenomena

Notable Approaches

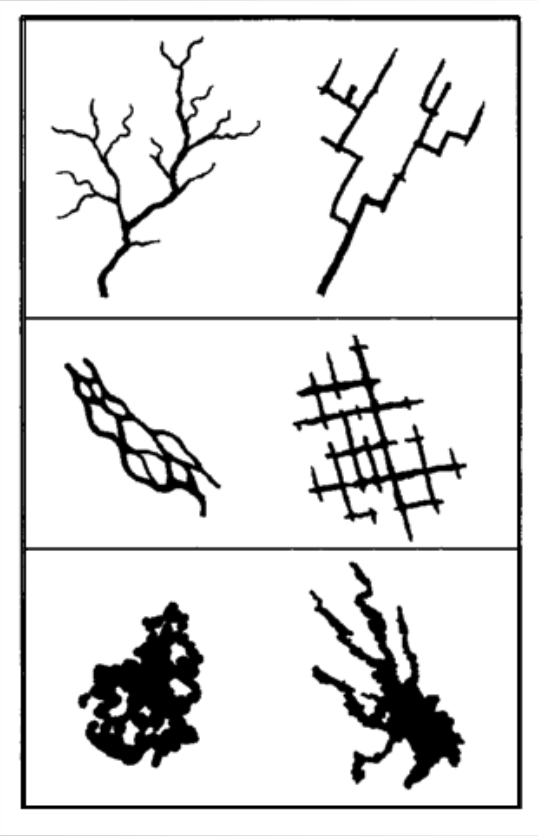
-Wires and radiuses
(Greeff 2009)



- Graph of Meta-primitives
(Yurovchak 2009)



Natural phenomena



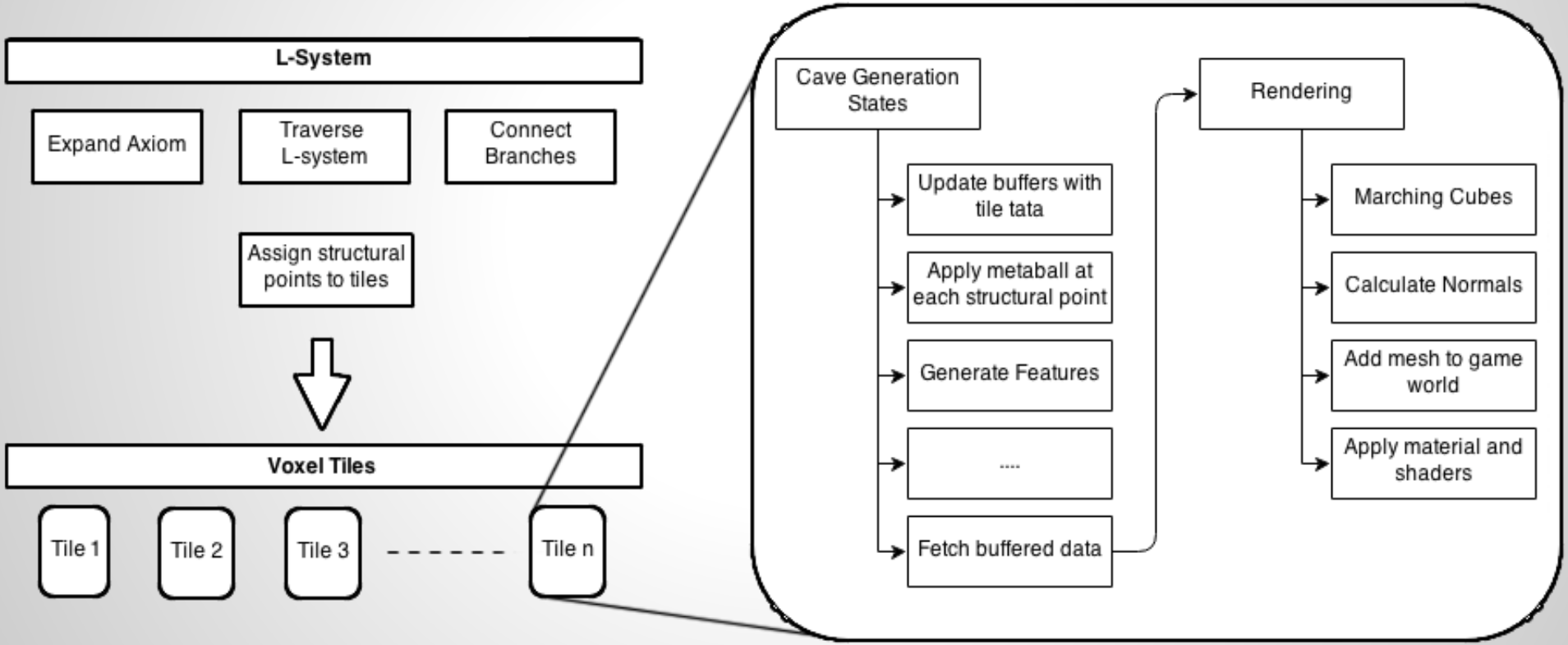
GPGPU Voxel Cave Modeling

1) **L-System**: emulate the expanded cracks and passages which form cave structures in nature.

2) **Metaball** carving and noise: intersecting virtual 3D brush strokes.

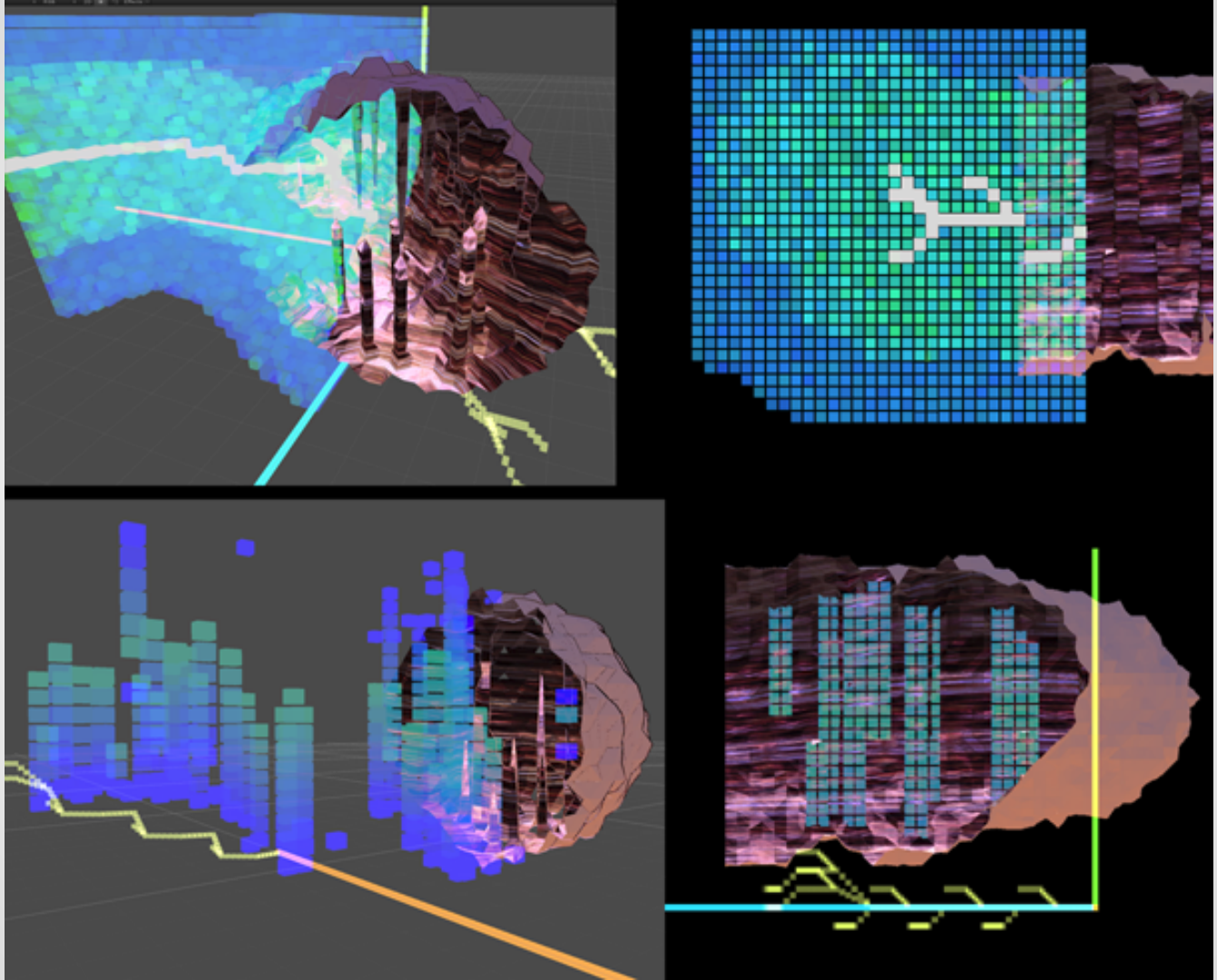
3) **Rendering**: isosurface extraction, and further mesh enhancement through shader programming.

Structure of the Pipeline



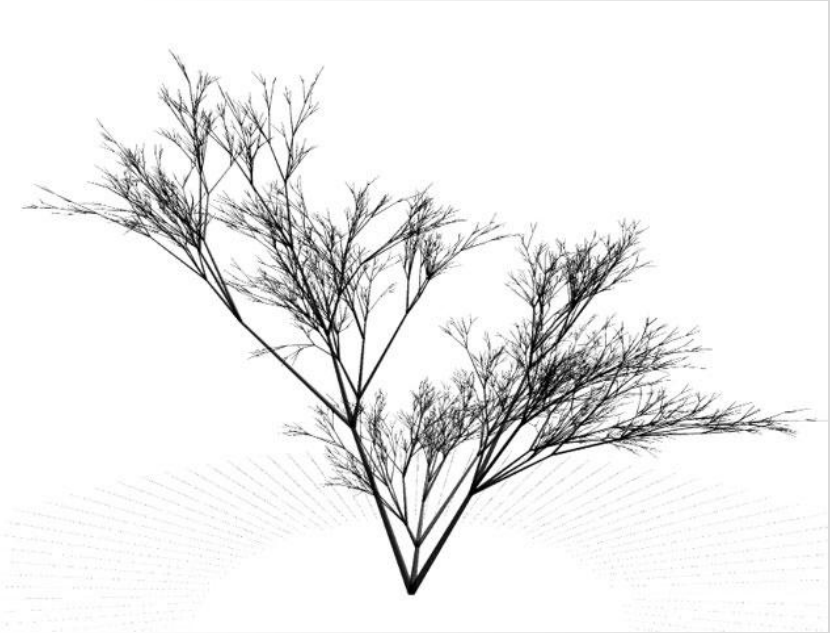
Why These Choices?

- Splitting the generation into structure and detail.
  - Discrete and clear outer structure. Doubles as guiding for detailing brush.
- L-system: customizability, control and speed (as opposed to agents or perlin worms).
- Smooth voxel manipulation with Metaball and procedural Noise (as opposed to Cellular Automata).



Options for Generating Structure

- Perlin Worms
- Constructive Methods
- L-Systems

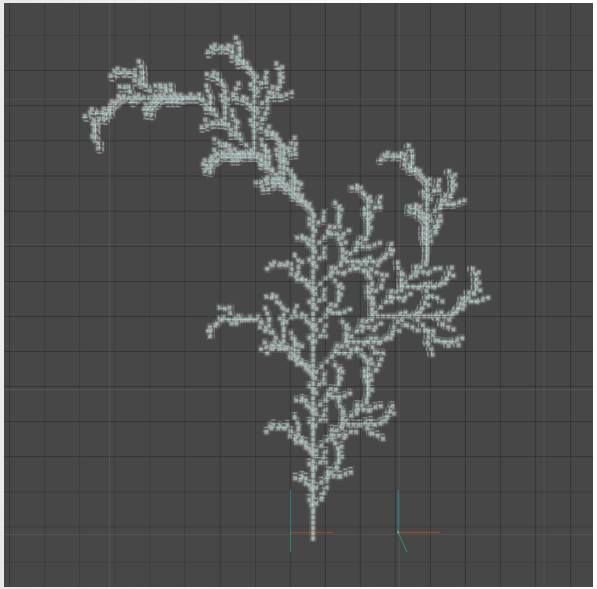


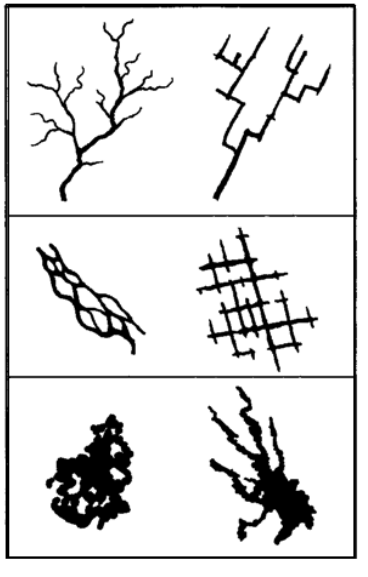
L-systems

- A type of formal grammar
- Consists of production rules and an alphabet
- Originally designed to model simple plants
- Generally interpreted as instructions for a turtle
- Can be extended with brackets or parameters to expand functionality

Why L-systems?

- Caves have an organic structure similar to vegetation
- An L-system is a simple way to represent a complex shape
- Can be easily controlled





Similarities between an L-System and natural caves

The L-system Alphabet

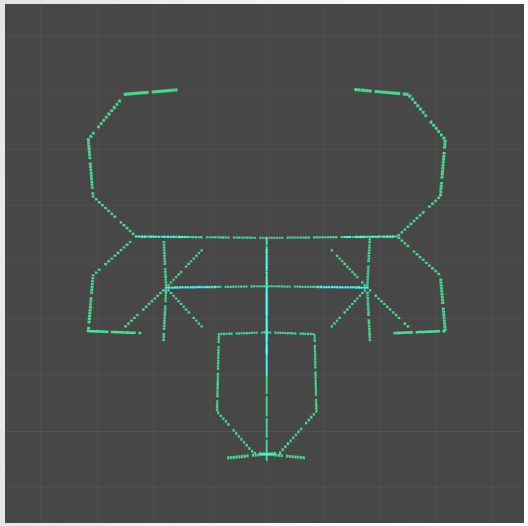
Axioms are interpreted as instructions for a turtle

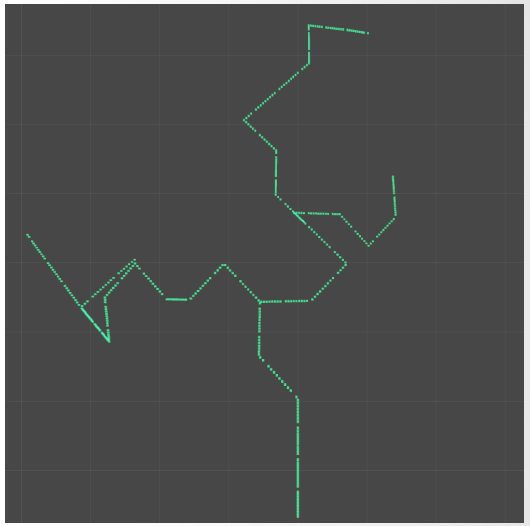
F	Move forward
L/R	Yaw left or right
U/D	Pitch up or down
O/A	Increase or decrease the turning angle
B/S	Increase or decrease the step size
Z	Symbol representing the tip of a branch
0	Stops this branch connecting to other branches

L-System Macros

- Simplifies complex sequences
- Are substituted before the rewriting step
- Introduces a stochastic element

C	A curve
H	A vertical curve that returns to horizontal
Q	A branching structure, representing a room
T	Similar to H, but splits into two curves
I	Represents a straight line

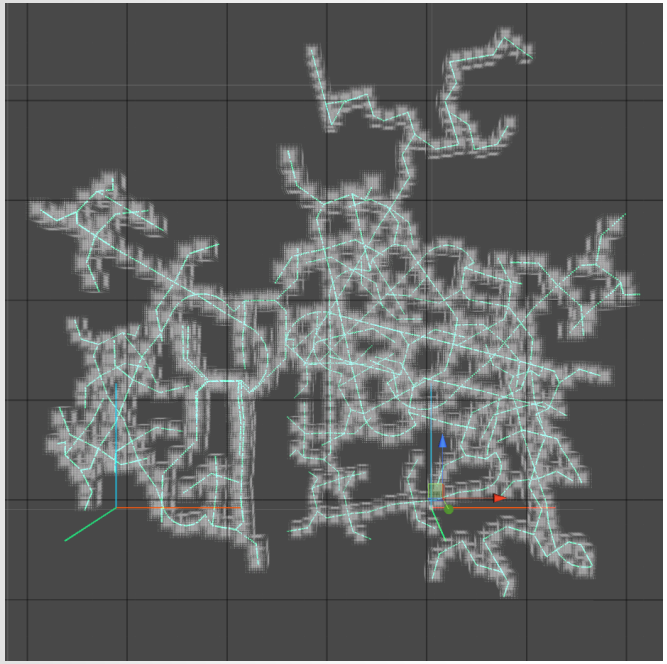


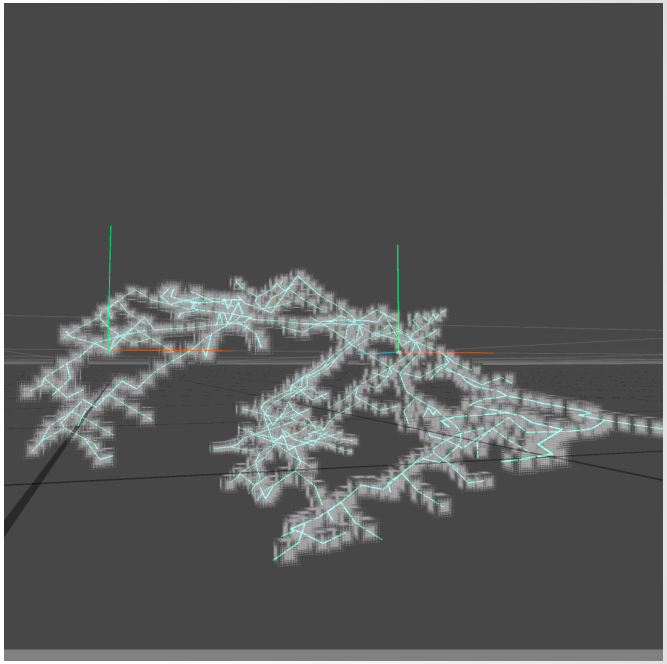


Two different versions of Q

Random Generation of Rules

- Main production rules can be generated
- Generated rules consists purely of macro symbols and brackets
- The frequency of symbols and brackets is modified by weights





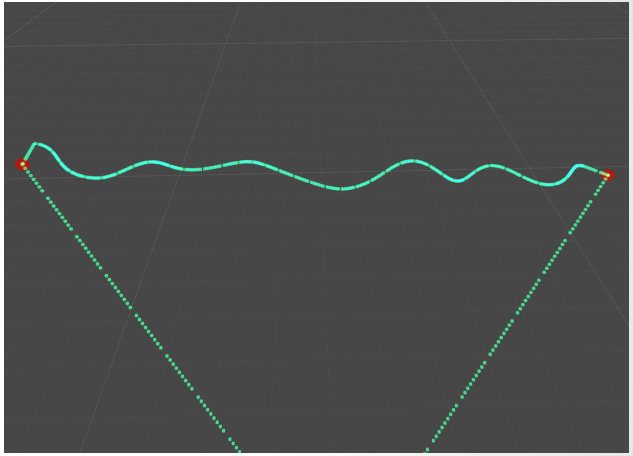
The final product of the L-system used with a randomly generated rule

Ensuring Connectivity

- L-systems are inherently tree-like
  - Dead ends could be a gameplay concern
- Can be avoided by simply connecting branches
- The level of connectivity can be controlled

Ensuring Connectivity

- Branches are chosen based on probability
- The connection is a straight line distorted by curl noise



Controlling the Structure

The structure can be controlled by:

1. Customizing the rules/macros
2. Specifying parameters

Customizing Rules and Macros

- Rules and macros are specified in a rule file
- Strings can be defined for each macro
- New macro symbols can only be added in code

Syntax: `Symbol;SubstituteString;Type;Weight`

Customizing Rules and Macros

- Main rules can contain both macros and alphabet symbols
- Any number of main rules can be added
- Each macro can have any number of strings
- The strings associated with each macro and main rule can be assigned a weight

Specifying Parameters

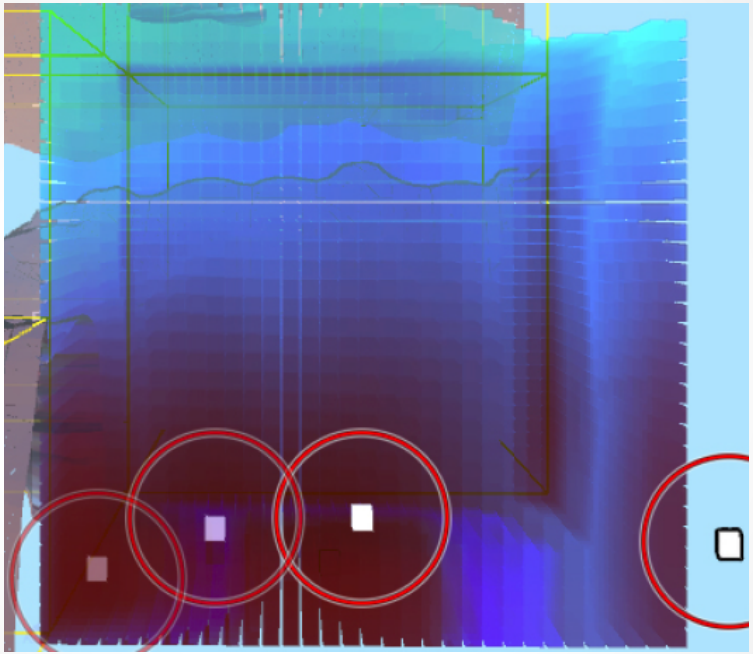
Parameters control every aspect of the structure:

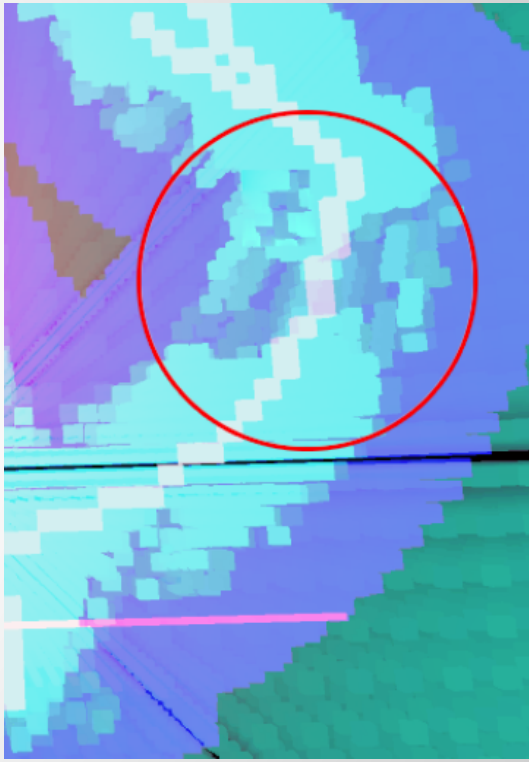
- The interpretation of the axiom by the turtle
- The generation of rules
- The connection of endpoints
- Constraining the L-system

Options For Modeling Detail

- Voxel erosion (physics or agents)
- Cellular Automata
- Noise function based distortions
    - Vertex offset
    - Energy field perturbation (Metaballs)

Voxel modeling: Metaballs vs CA





Metaball Benefits:

- Smooth voxel manipulation.
- No wasted frames.
- Empty centre + full edge = floating geometry can be controlled.

Benefits of Metaballs

Intersecting virtual brush strokes: More control over landmark/feature creation with the L-system “wires”.

Can be distorted with a cocktail of noise to achieve:

- A better defined complex result (match cave wall shapes).
- Different detail across multiple scales.

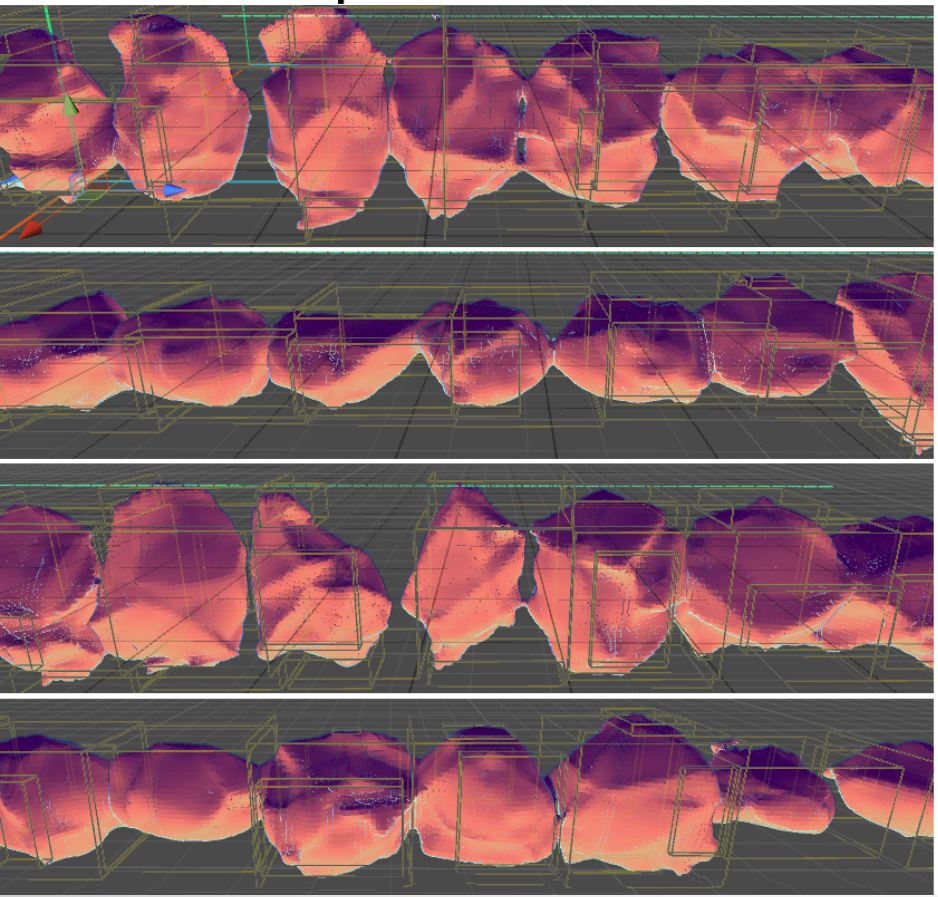
3D Noise Field Distortion Equation

- f(structural point, current voxel point);

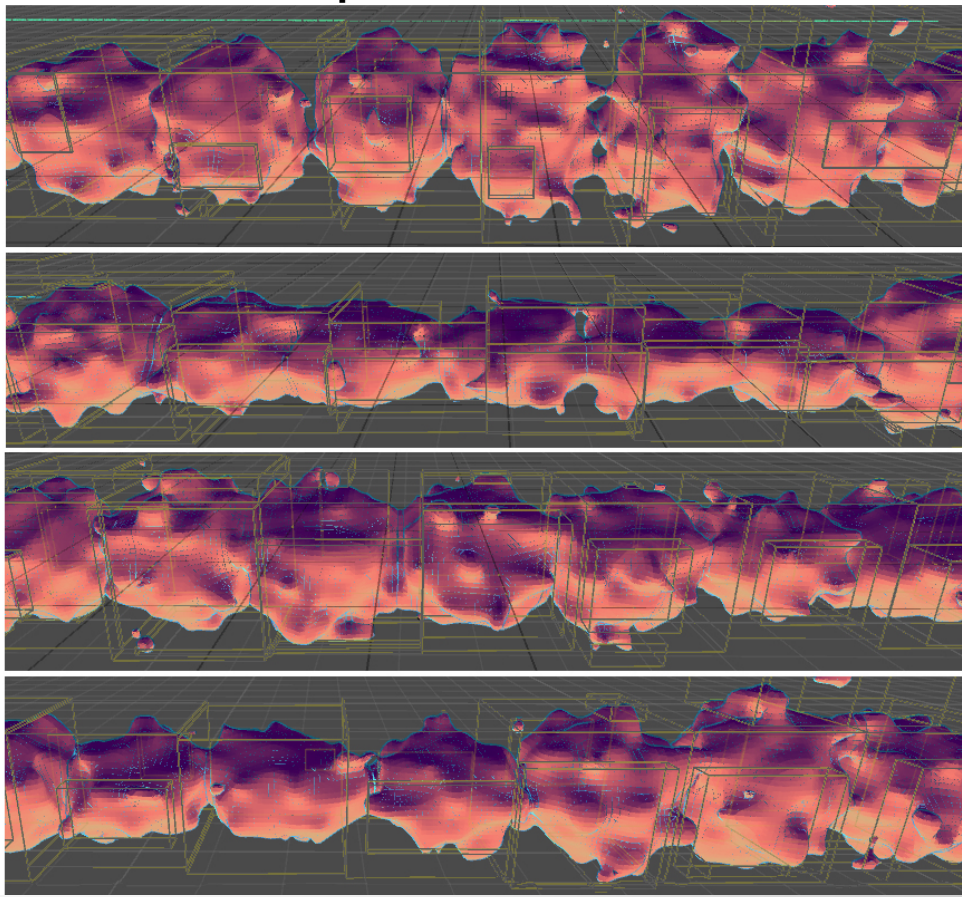
1. Offset structural point by Curl(Simplex) noise value for few but abrupt alternations between tall and short tunnels;
2. Compute distance;
3. Warp distance by Simplex noise to get scallops ;
4. Warp distance by 3 Voronoi noise scales to get sharper angles on cave walls.

* alternate each distortion by low frequency Simplex noise.

Simplex + Voronoi



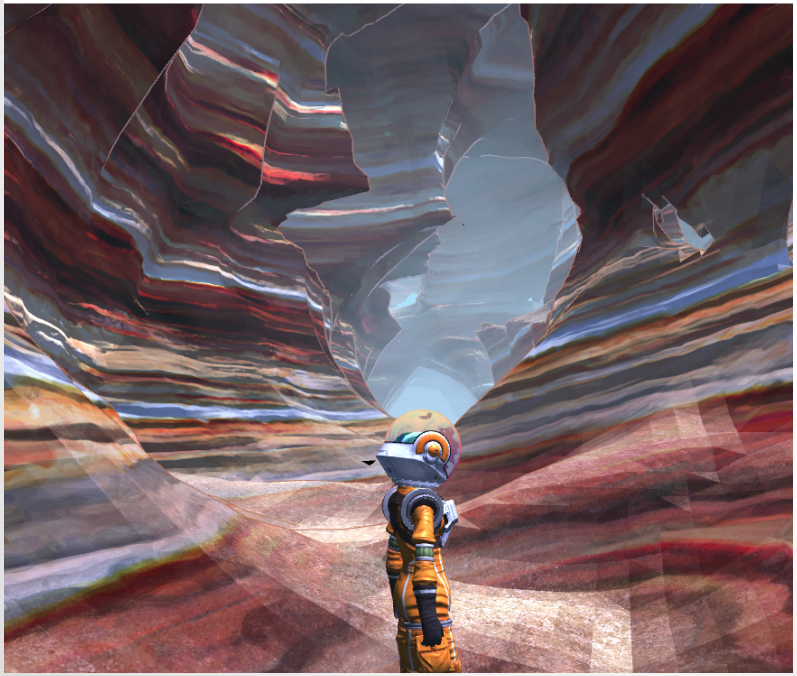
Simplex + Curl

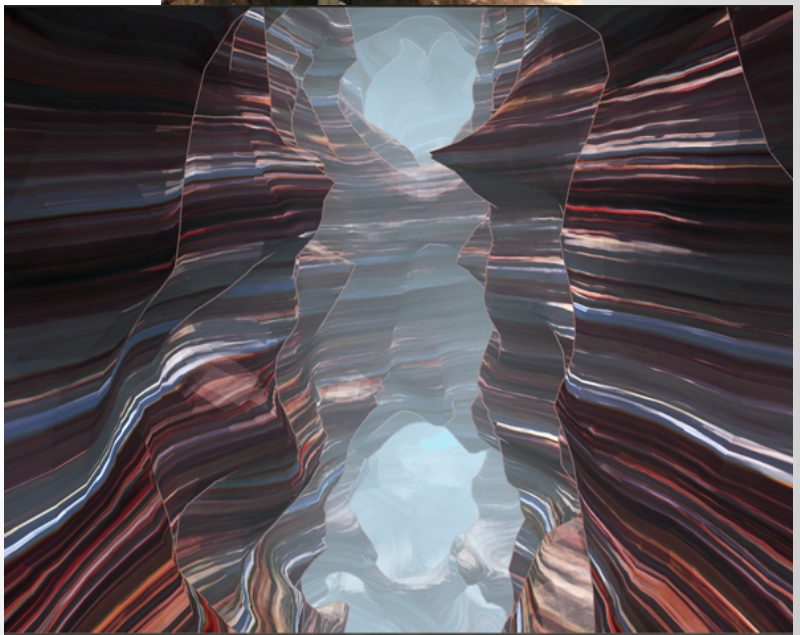


Metaball

- Frequency and amplitude can be tweaked.

- Future work: expose warping equation as pluggable elements for designers
  - Change noise types
  - Change the way they interact
  - Live “brush” preview



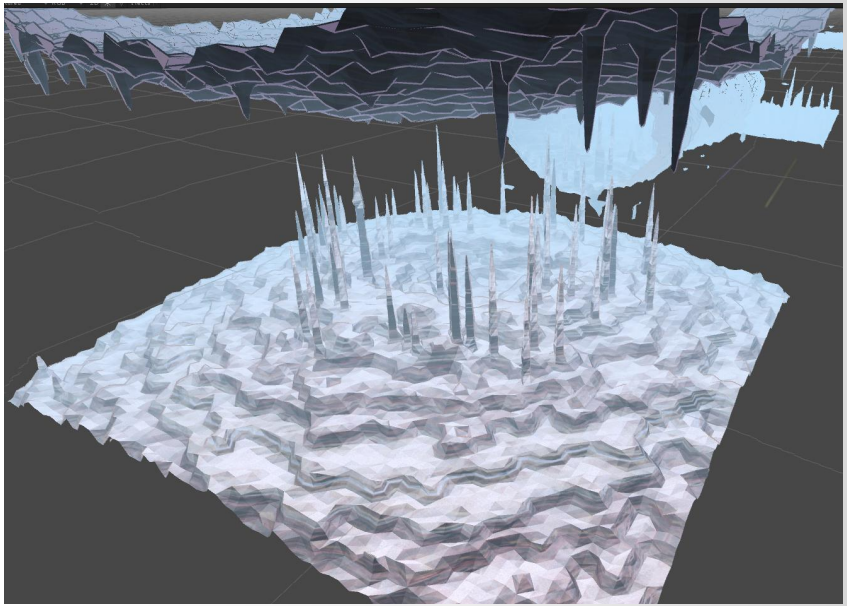


Adding Features

- Features that are found in caves are simulated
- Positioned with noise and grown with Cellular Automata
- Feature frequency and density can be tweaked.

Adding Features

- Clumped together with low frequency noise
- Positioned with high frequency noise
- Can be used for placement of any class of objects.

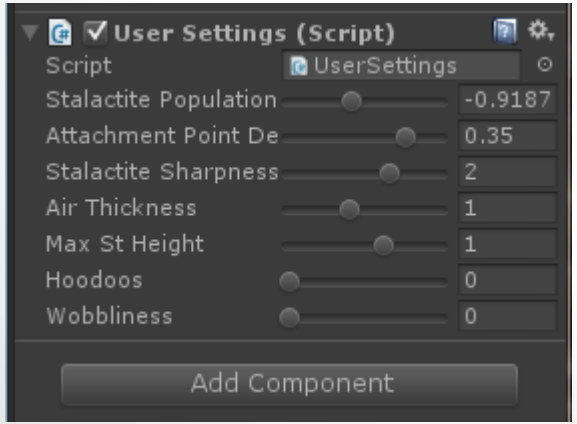


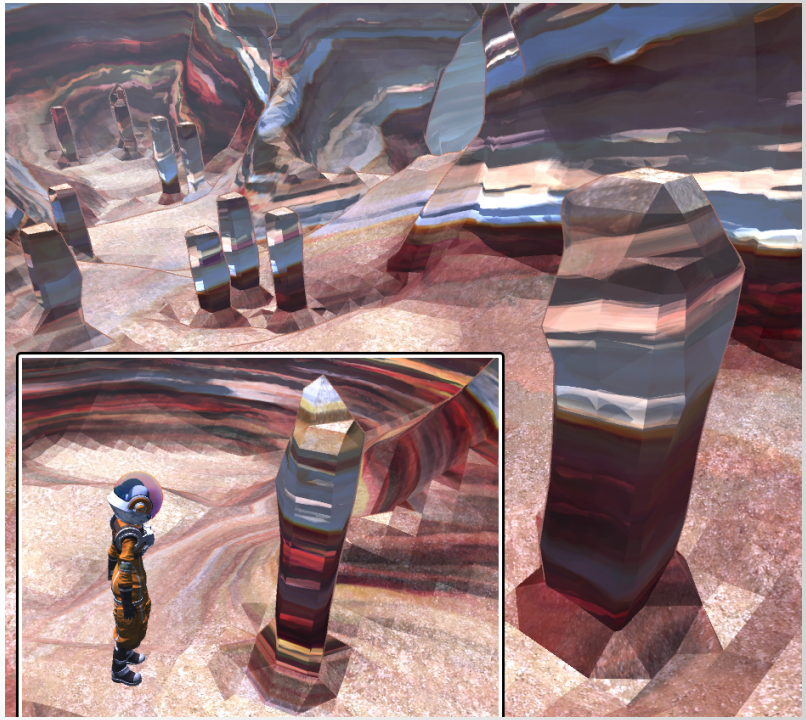
Adding Features

- The features are built with cellular automata from the floor and ceiling
- Length is based on the initial noise value and the distance from the spawning point
- It is possible to form a complete column

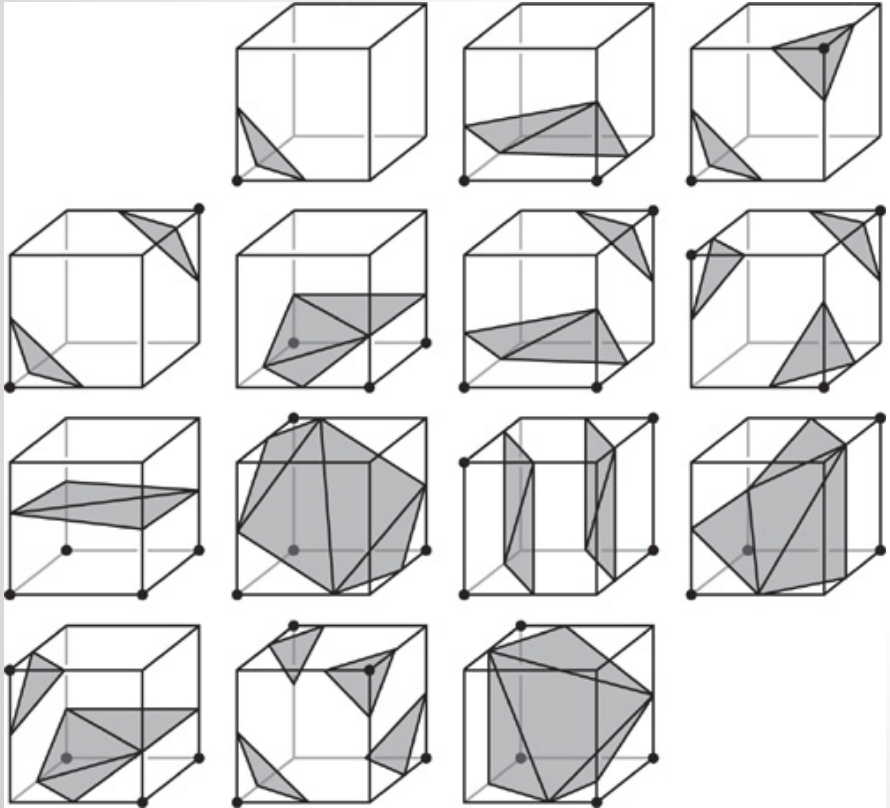
Shaping Features

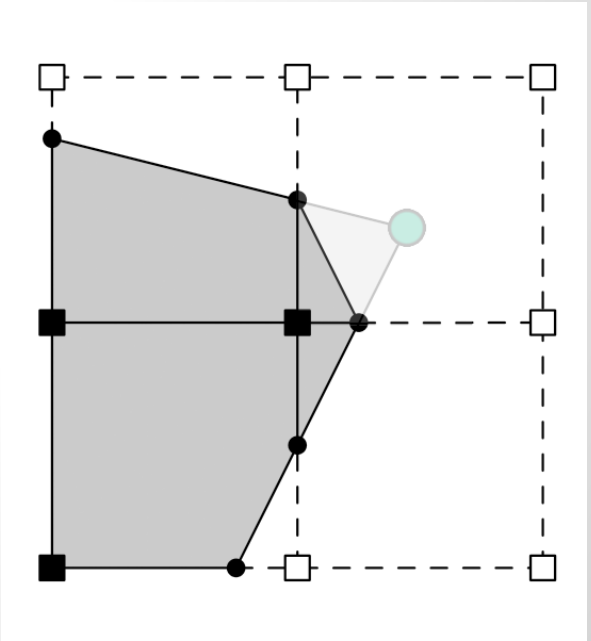
Stalactite/hoodoo shapes can be tweaked by sliders attached to their noise values.



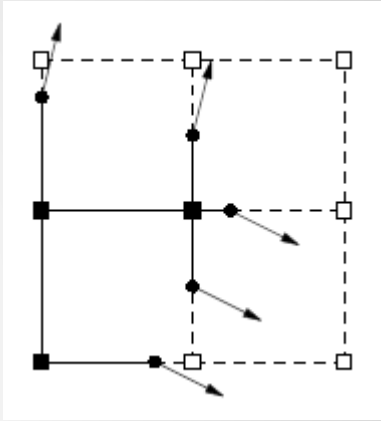


Isosurface Extraction: Marching Cubes

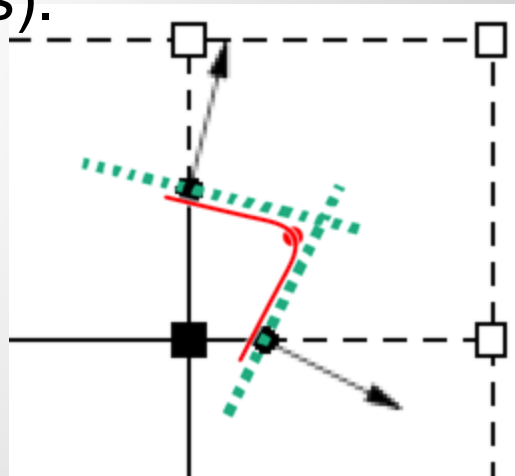




Dual Grid methods - “Better”



- Goal: Create a second grid of “feature nodes” to approximate detail.
- Feature node is inside the primal cube (red).
- Requires Hermite data (depends on normals).



Problem:

- Carve with Constructive Solid Geometry.

or

- Quickly compute gradient function.

Mesh Extraction

- For now, we use a primal grid method (Marching Cubes).
    - Allows experimentation (not bound to Hermite data or partial derivatives)
- Smooth voxels -> Smooth Marching Cubes.
- Getting smooth normals from voxel topology.
    - Large cluster Normal smoothing.

Rendering

- Triplanar projection + Solid texturing

  Write full pixel-lit Lambert shader from scratch:

  - Tile the 3 base textures to work as infinite planes.

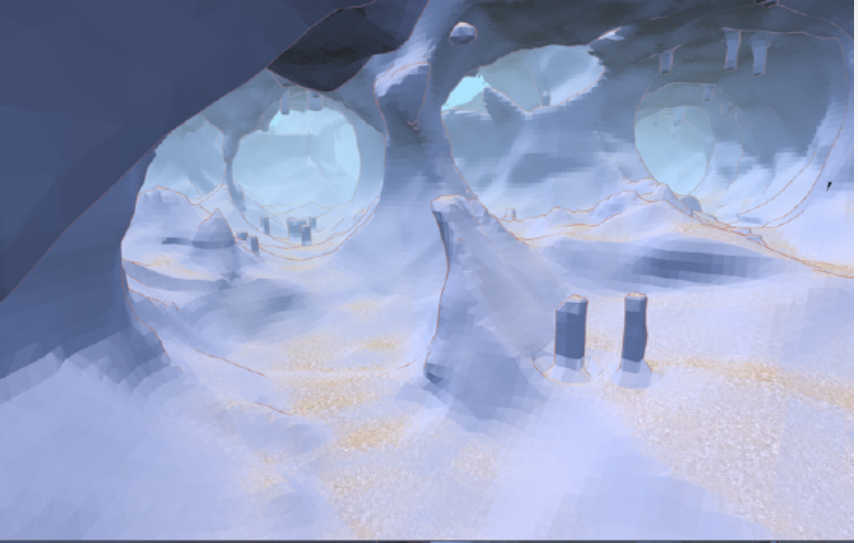
  - `float3 s = CurlNoise(worldspace(fragment.xyz));`

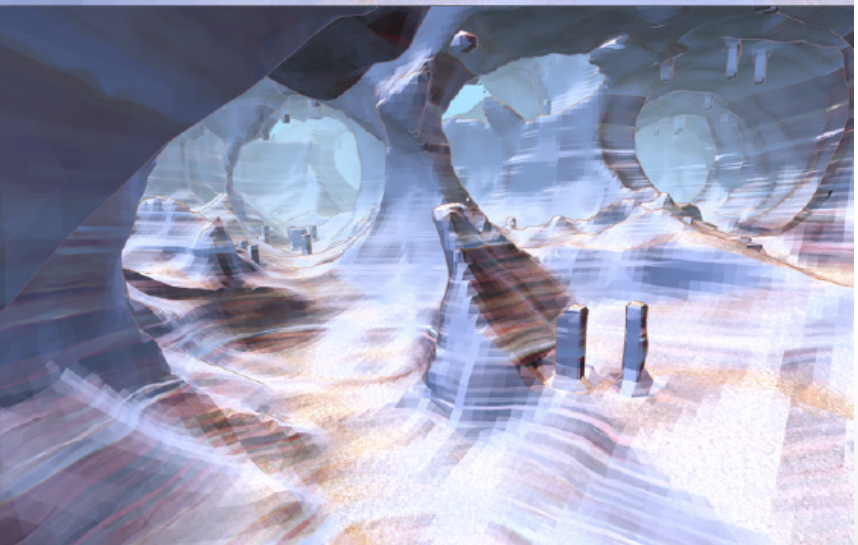
  - `Sample2D(Tex1, s.yz);Sample2D(Tex2, s.xy);Sample2D(Tex3, s.zx);`

  - Blend the 3 texels by a weight based on the Normal.

  - Blend in bumpmaps and other filter shenanigans/sparkles.

  - Calculate lighting equation.





sharedMeshMaterial

Shader: Custom/OutlinedDiffus | Edit...

Main Color	
Lambert Contrast (2)	1
Bump Contrast (1)	1.75
Texture Contrast (1)	0.77
Curl Octaves (1)	500
Curl Octaves (2)	3
Outline Color	
Outline Color 2	
Outline width	
OutlineTop width	
Marble Texture Scale	
Marble Contrast	1.25
Marble Saturation	1.27
Projected Texture Scale	0.036
Bumpmap Texture Scale	0.8

Noise Seed

X -4484.2 Y -5230.7 Z 991.125 W 0

Base 1 (RGB) — Select

	Tiling	Offset
x	1	0
y	1	0

Base 2 (RGB) — Select

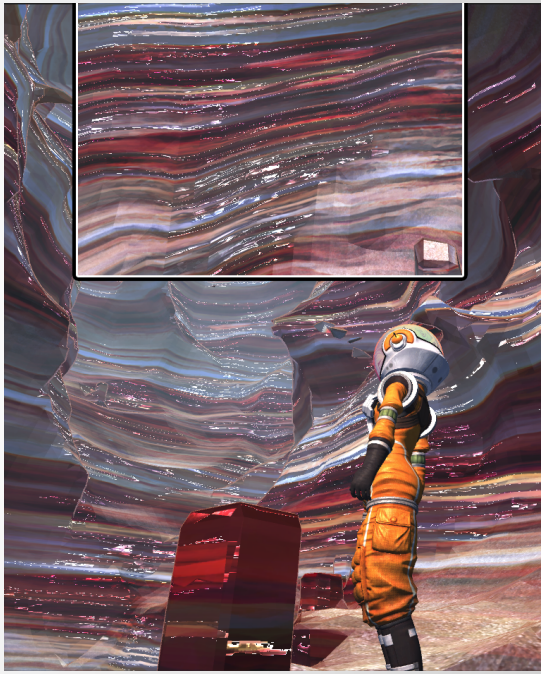
	Tiling	Offset
x	1	0
y	1	0

Base 3 (RGB) — Select

	Tiling	Offset
x	1	0
y	1	0

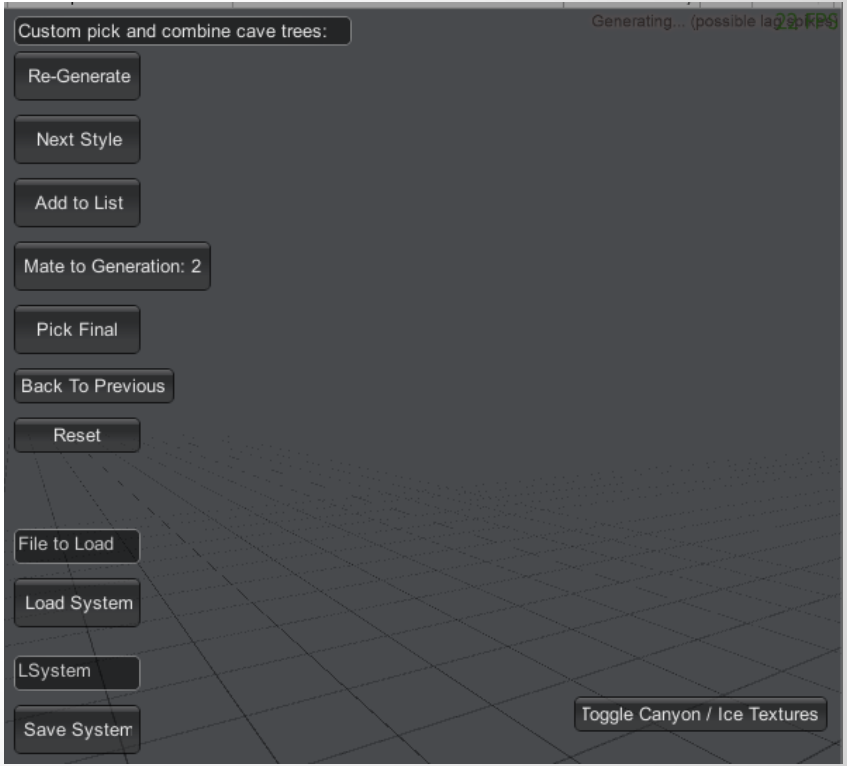
Bump 1 — Select

	Tiling	Offset
x	1	0
y	1	0



Designer Interface

- A tool for designers to easily select and save caves
- Facilitates interactive evolution



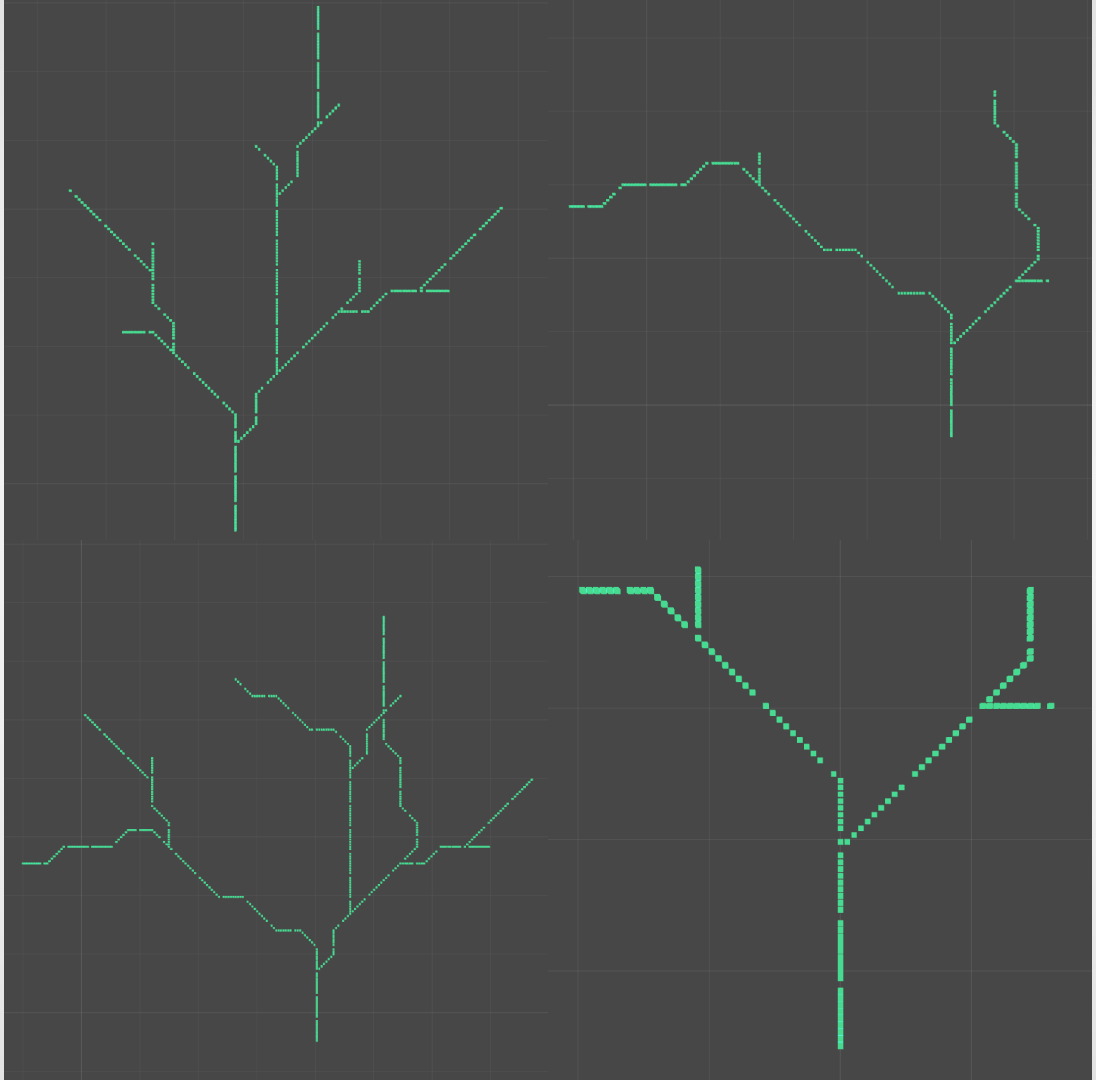
Interactive Evolution

Problem:

Caves can be unsuited for the purpose of a designer

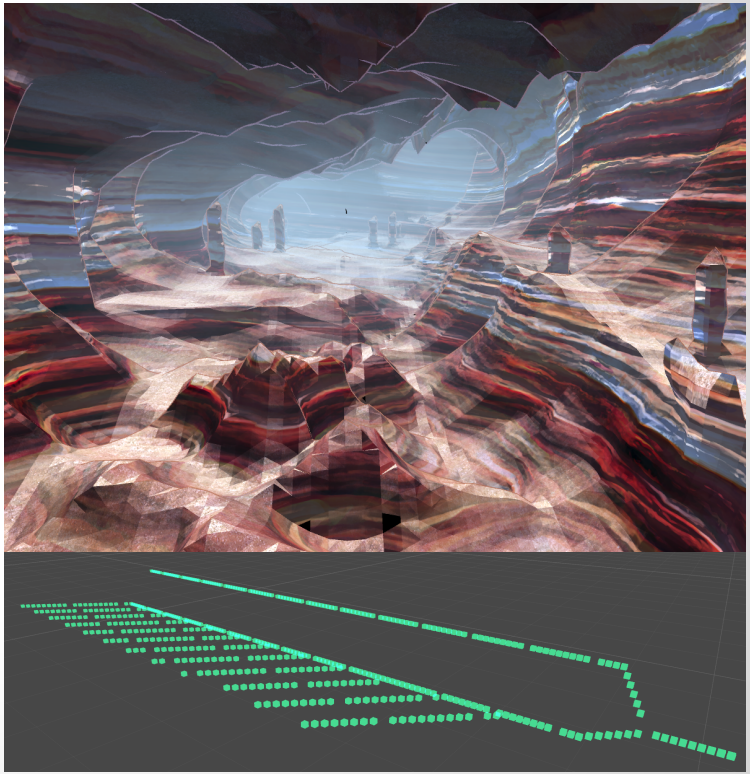
Solution:

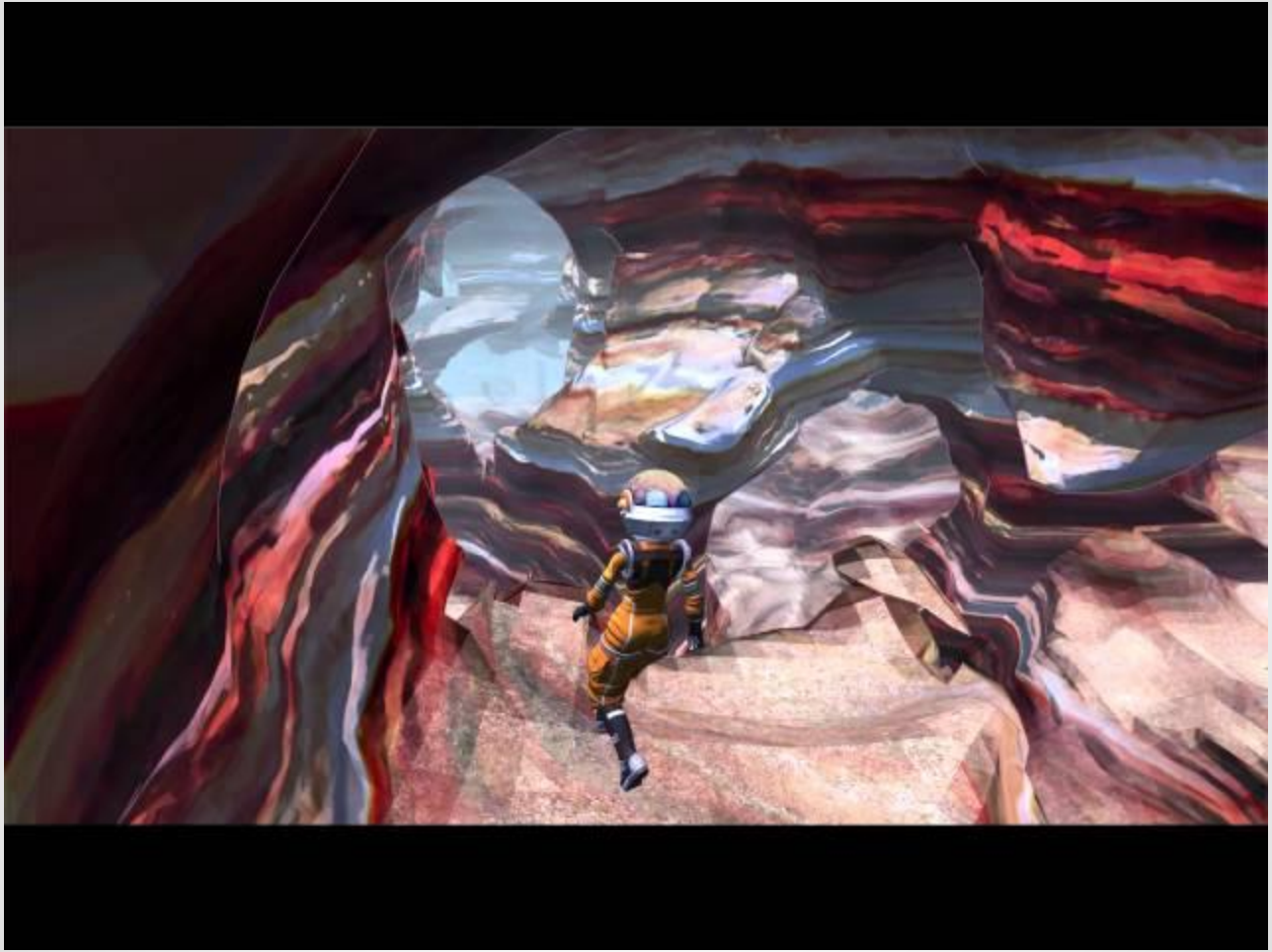
Allow the designer to rapidly view several caves and combine the useful ones

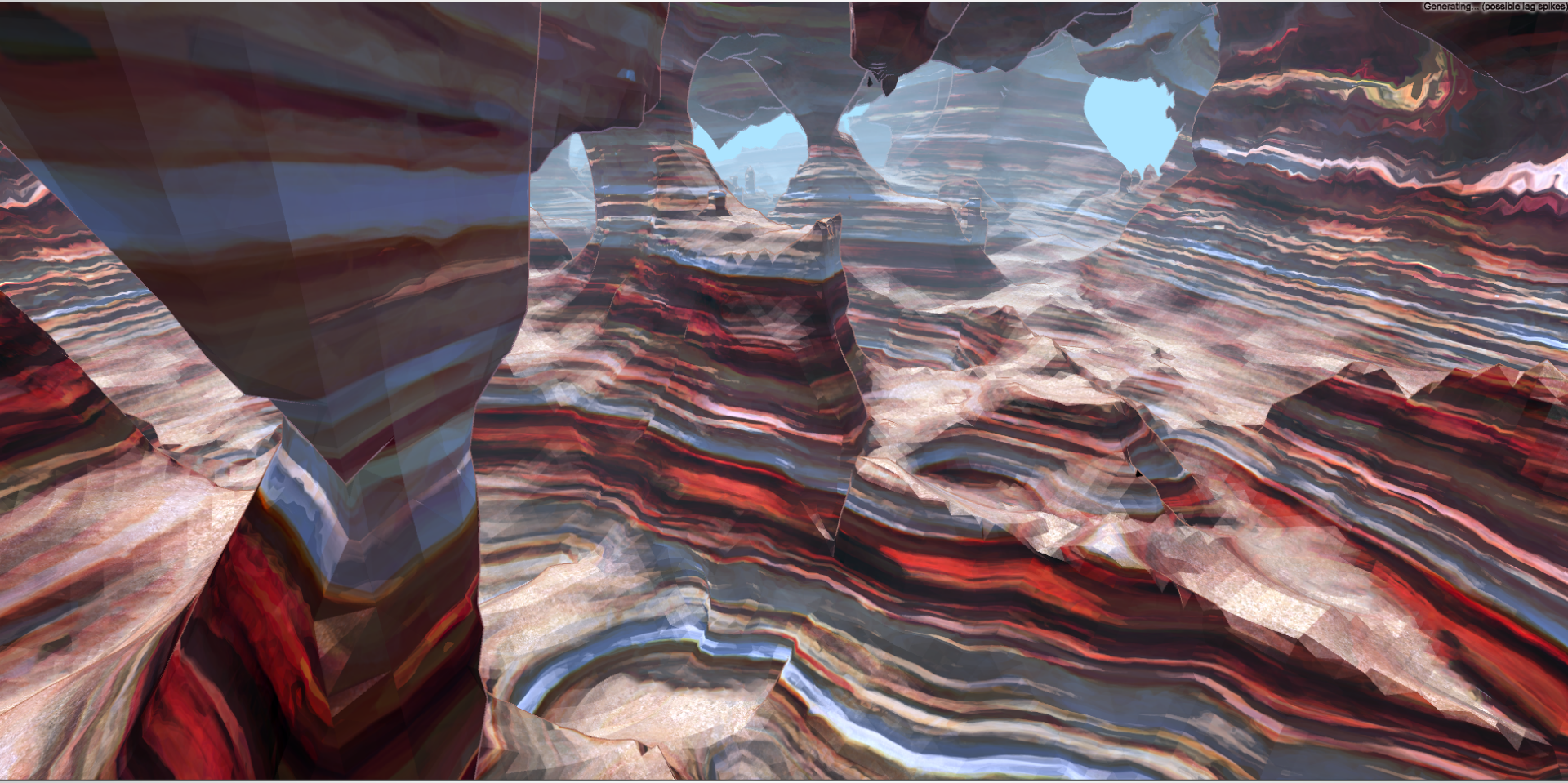


Structure and Detail Interactions

The structure and detail interact to create complex features, due to overlapping metaballs






Generating... (possible lag spikes)

User Feedback

- User testing was performed on random users via the internet
- The users were asked to explore the generated caves in a web build
- The users were presented with a questionnaire

User Feedback

- 30 users answered the questionnaire
- Feedback was generally positive

Overall Feedback

Negative	Neutral	Positive
20	43	143

User Feedback - Major Points

- ➔ Caves matched most natural phenomena very well.
- ➔ Art style and general visuals were very good.
- ➔ Generation speed was a problem.
- ➔ Most users reported spending 5-15 minutes.
- ➔ Most users only tested a single cave.
- ➔ Some users perceived a lack of variety.
- ➔ Most users found the caves well suited for an exploration style game.

User Feedback - Conclusions

- Stalactites need to be improved as they look unnatural
- More variety should be added to textures, and tunnel size
- Needs improvements to generation order and speed

Our Goals

- Compelling organic cave terrain.
  + believability and expressivity (emergence)
  - reliability
- Gameplay viability.
- Customizability.

Challenge:

- Emergence is unreliable; we can control it in code, but designers should also be able to.

Weaknesses of the Solution

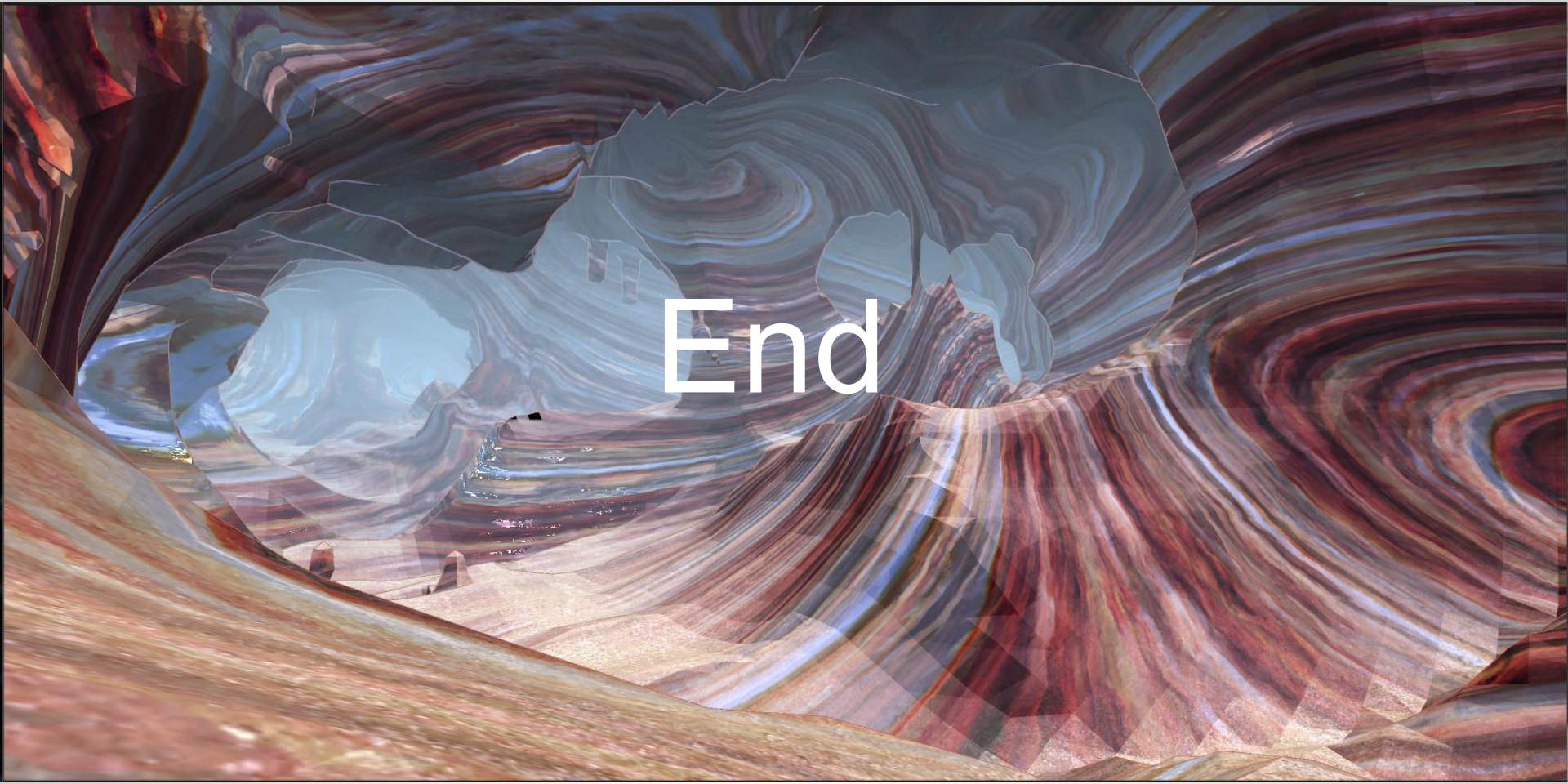
- Lack of LOD
- Generation is too slow for real time, due to limitations of Unity Free
- Tile based solution creates problems with floating and missing geometry

Possible Improvements

- Octree
  - LOD
  - One large sparse tile instead of many
- Offload noise calculations to 3D texture(s)
- Better isosurface: Dual grid
  - Hermite: gradient calculation or CSG Metaballs

Possible Improvements

- Finer control with parametric L-systems
  - Context sensitive decisions
  - Variations of tunnel shape and size on a large scale
  - Various biomes for different parts of a cave
- Designer tools for:
  - Custom distortion function
  - Manual control over structure with graph drawing


End