AUTHORING TOOL DESIGN DOCUMENT OUTLINE

TITLE: Hydro Terrain

PROJECT SUMMARY

In a vast majority of visual media, be it games or movies, terrains play an integral part of the viewer's experience as it is oftentimes the largest asset that is seen. However, most assets of significant size are slow to create manually; therefore, the ability to quickly produce visually accurate terrain is highly desired. Our HydroTerrain tool aims to fulfill this need by generating believable terrain based on real hydrology principles and its effects on the surrounding landscape.

We are aiming to create a tool that takes simple user inputs such as contour maps and placement of river nodes which will then be used to generate a terrain. Since the inputs are at the user's discretion, we are designing this tool for level designers, artists, and game designers due to its ease of use as these users are most likely to use a terrain for their projects. Once an initial terrain is built, the user can change the original settings if need be to edit the terrain features to their liking.

Most terrain generation tools involve fractals or erosion; while they can provide excellent results, to a trained eye the terrains produced with such methods can be unrealistic at certain areas. With our hydrology based system, not only will the tool provide accurate geology that emulate real world terrain systems, but there is the added benefit of river systems being generated which can not easily be done in the methods mentioned earlier. The users will be able to have a better control of where such important terrain features are placed do to the inherent nature of producing a terrain based off rivers.

The terrain itself is created by a Voronoi partition of the initial points set out by the user. The river network is represented by a geometric graph which is created when the initial seeds expand via an algorithm that utilizes Horton-Strahler ordering. New nodes generated will be of a higher altitude to ensure consistent water flow. Once Voronoi cells are generated, river junctions and paths are created while each cell's area is filled in with terrain primitives. The structure we use to store the terrain is a tree structure where leaves are the base terrain primitives; we can combine these leaves in their parent nodes for feature blending.

Our development time schedule will see us deliver an alpha, two betas, and final version. The date of the alpha will be March 19, 2014. The betas will be completed April 2, 2014 and April 21, 2014 respectively. Our final release will be on May 9, 2014.

1. AUTHORING TOOL DESIGN

1.1. Significance of Problem or Production/Development Need

Virtual terrains have played an important role in computer graphics for the past twenty years, and their application ranges from movie productions to landscape design to flight simulators. While researchers have made considerable progress toward developing efficient methods for synthetic terrain generation, most existing technologies fall into two distinct categories, each with specific advantages and drawbacks.

One of the more popular techniques for terrain generation is through procedural modeling. This method offers a relatively simple implementation and a wide range of variation they when a few parameters are changed [GGGPB13]. However, these features come at the cost of accurate geomorphology , as most of these approaches are based on fractal subdivisions and noise function combinations.

Physics-based algorithms provide results that are correct from a geological standpoint, as they take into account the generation of terrain that are exposed to various morphological agents such as water and temperature changes. However, because of the amount of computation involved in erosion simulations, these methods cannot model large terrains with a high level of detail. Another main disadvantage of morphological algorithms is their low controllability.

HydroTerrain seeks to address the main disadvantages described above, and provide the user with an intuitive framework for procedural terrain generation using rivers as modeling features. Our tool aims to offer the user a considerable amount of control over the final terrain model, which is generated both rapidly and with geomorpholigical accuracy.

1.2. Technology

Our primary paper is “Terrain Generation Using Procedural Models Based on Hydrology” by Genevaux, Galin, Guerin, Peytavie, and Benes. This paper discusses how traditional methods of procedural terrain generation produced adequate terrains but had their own drawbacks. The method employed by Genevaux and his associates introduce a new way to create terrain based off of real principles of hydrology. Not only does this generate a believable environment but has the additional benefit of creating river systems, which can be difficult and tedious to do by hand.

The authors reference several other papers either for technical assistance or research on real world hydrology. The significant papers are as follows:

- “Terrain Simulation Using a Model of Stream Erosion” [KMN88]

- "A Fractal Model of Mountains with Rivers" [PH93]
- "Modeling Landscapes with Ridges and Rivers: Bottom Up Approach" [BA05]
- "River Networks for Instant Procedural Planets" [DGGK11]
- "The Synthesis and Rendering of Eroded Fractal Terrains" [MKM89]
- "An Erosion Model Based on Velocity Fields for the Visual Simulation of Mountain Scenery" [CMF98]
- "Terrain Synthesis from Digital Elevation Models" [ZSTR07]
- "Interactive Terrain Modeling Using Hydraulic Erosion" [SBBK08]

Our specific paper was chosen because of the need for a fast realistic terrain generator. Terrains can be created in Maya through the clever use of height maps but it is not as intuitive nor as interactive as our method.

1.3. Design Goals

Our tool will allow the user to procedurally create hydrologically accurate terrain while maintaining a high level of control over the generation process. There will be an emphasis on creating a visually compelling end model that can be manipulated by the user through various contour maps. The user will be able to interactively create a river network system, where river networks are either generated procedurally or through user-specified parameters. With the river network system as a foundation, the user can continue to interactively create terrain representations. We intend to use computationally inexpensive techniques to allow rapid generation of the geometries. This tool will be developed as a Maya plug-in using MEL scripting and the C++ API.

1.3.1 Target Audience.

Since our tool aims to generate detailed geometry via an intuitive interface, there is little technical knowledge required in using it. We expect our main audience to consist of technical artists in the fields of game development and CG for level design and environment creation. The tool can also be used by people with little experience with Maya, since all of its functionality will be encapsulated within a single user interface.

1.3.2 User goals and objectives

We envision our tool being used to rapidly prototype various types of terrain, since its procedural component enables quick generation of river network and terrain parameters not specified by the user. We also see artists using our tool to create complex and detailed geometry by interactively modifying input parameters to their specifications.

1.3.3 Tool features and functionality

HydroTerrain will comprise of two main functionalities: river network generation and terrain geometry generation.

The river generation component of the tool will allow the user to interactively provide the contour of the generated terrain and designate river mouth locations as well as set the river slope values.

The terrain generation component of the tool will allow users to generate a terrain mesh from the river network based on a set of primitives and blend/replace operators. The user will be able to control the nature of the generated terrain by refining river network parameters in the previous step.

1.3.4 Tool input and output

For input the user supplies an initial contour map as well as a starting set of candidate nodes along the contour as the basis for the river network generation.

There is an intermediate output that displays visual feedback on the generated river network.

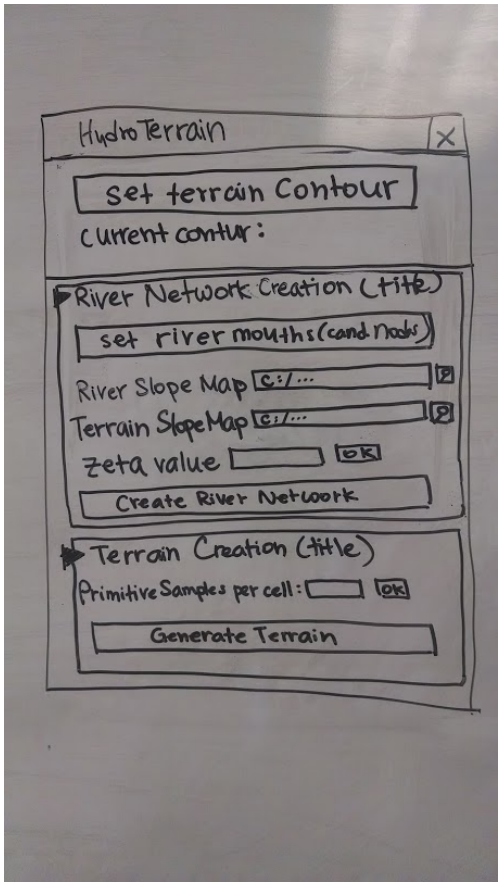
The final output is a 3D mesh of the generated terrain.

1.4. User Interface

1.4.1 GUI Components and Layout

The user interface will be simple and minimal. It will reflect the main workflow of the algorithm with one section outlining river network generation, and another for terrain generation.

The user will be able to set a curved, enclosed line as the initial contour of the terrain, as well as specify the initial candidate nodes for river network expansion. The user will also be able to control which nodes are chosen during network expansion by adjusting a constant (zeta) that weighs node elevation against priority. The tool will take in image maps as input for terrain slope and river slope values. After the initial river network is created, the user will be able to specify the number of primitive samples used to create each terrain patch. After this is set, the tool will be able to generate a complete terrain.



Preliminary GUI mockup

1.4.2 User Tasks

The user should have basic familiarity with the Maya interface and knowledge on how to create and manipulate basic geometry such as curves. For river network generation, the user will be able to set river nodes along the initial contour. For terrain generation, the user should be comfortable working with a polygonal output mesh.

1.4.3 Work Flow

The tool takes in as input contours from the user a curve to define the contour of the terrain and a set of locators that mark the placement of river mouths. The user will also submit river slope map and terrain slope map images, as well as a value for the constant zeta that weights node elevation against priority. The user will then input the number of primitives that will be used to generate each terrain patch. The tool generates as output a 3d polygonal terrain mesh.
A typical work flow will try to follow the steps listed:

1. Draw a closed Maya spline to outline the terrain contour or have one available.
2. Click on a contour spline and click “Set Terrain Contour”.
3. Click along contour spline to attach locators that represent candidate nodes.
4. Select the newly created locators and click “Set River Mouths”
5. Input a River Slope Map.
6. Input a Terrain Slope Map
7. Enter a Zeta value to determine whether elevation or priority is more influential.
8. Enter a number for primitive samples per cell.
9. Click “Generate Terrain”.

2. AUTHORING TOOL DEVELOPMENT

2.1. Technical Approach

2.1.1 Algorithm Details

From a high level, the algorithm for generating hydrologically accurate terrain can be broken down into two main sections—generating the initial river network followed by the generation of the actual terrain. We will describe the steps involved in each.

River Network Generation

In the first step, the user interactively provides the contour of the generated terrain, river mouths, and input parameters. From this input, the system will generate the drainage river network inside the domain formed by the contour, and the network will be represented as a geometric graph. The image maps for river slope and terrain slope will be fitted to the bounding box of the river contour, and the values for each node location will be determined by the pixel lightness at the corresponding point on each image.

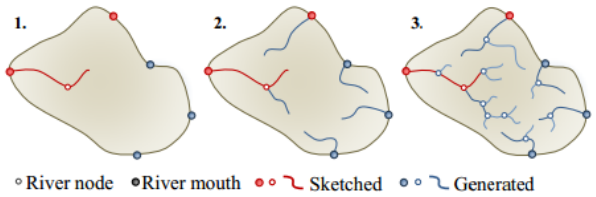


Figure 3: *Given an initial contour Γ and a partial river graph $\mathcal{G}$ (step 1), the river network is incrementally generated by growing the geometric graph over the domain Ω (steps 2 and 3).*

Initial Candidate Nodes: The river mouths from user input are considered seed

nodes, and are located on the boundary of the input contour. This set of initial candidate nodes will be expanded based on a set of rules to be explained further into this section. Each node will be assigned a priority index that defines its importance. In addition, each node will expand into its own river network tree, which collectively form the geometric graph covering the domain defined by the input contour. In this step, we have decided to deterministically set the input nodes and priority indices instead of generating them procedurally in favor of time.

River Network Generation: The river network is generated by incrementally growing and elevating the river network trees using a probabilistic approach. The set of candidate nodes is expanded using a selection algorithm and several rules. The following steps will be performed iteratively:

1. Node selection: choose a node from the set of candidate nodes that well be expanded
2. Node expansion: expand the candidate node and perform geometric tests to verify that the new nodes are compatible with the previously created ones. Any new node that is not compatible will be removed from the graph. The elevation of each new node will be computed according to a river slope magnitude value provided by the image map described earlier.
3. Node creation: update the list of candidate nodes.

The node that will be expanded is selected from the list of candidate nodes by using a heuristic that takes into account the node elevation and its priority index, as well as the user specified constant zeta. The method proceeds as follows:

1. Find the elevation z of the lowest located candidate
2. Consider the subset of admissible nodes whose elevations are within the range [z, z+zeta]
3. Choose the node with the highest priority from the set of candidate nodes

The river nodes are expanded using the rules described in Table 1 below:

0.			$\alpha_1(s_1)\ldots\alpha_n(s_n)$		{ *Axiom* }
1.	$\alpha(1)$	$\longrightarrow$	$\tau(1)\,\beta^p(1)$ with $p \in [1;5]$		{ *Filling* }
2.1	$\alpha(n)$	$\longrightarrow$	$\tau(n)\,\beta(n)$	$:\mathcal{P}_c$	{ *River growth* }
2.2		$\longrightarrow$	$\tau(n)\,\beta(n-1)\,\beta(n-1)$	$:\mathcal{P}_s$	{ *Symmetric Horton-Strahler junction* }
2.3		$\longrightarrow$	$\tau(n)\,\beta(n)\,\beta(m)$ where $m < n$	$:\mathcal{P}_a$	{ *Asymmetric Horton-Strahler junction* }
3.1	$\beta(n)$	$\overset{\text{if valid}}{\longrightarrow}$	$\alpha(n)$		{ *Instantiation:* $\mathcal{X} \leftarrow (\mathcal{X} \setminus \{N_\mathcal{X}\}) \cup \{N\}$ }
3.2		$\overset{\text{otherwise}}{\longrightarrow}$	ε		{ *Rejection:* $\mathcal{X} \leftarrow \mathcal{X} \setminus \{N_\mathcal{X}\}$ }

Table 1: *Expansion rules for the hierarchical drainage network growth.*

The non-terminal symbols in the expansion rules are parameterized by a priority index. This is the Horton-Strahler number, a measure of the branching complexity of the graph. The rules code the Horton-Strahler number into symmetric and asymmetric branching, and growth stops when it is no longer possible to add new nodes. The set of nodes and their corresponding edges are

then classified and passed to the terrain generation portion of the algorithm. We plan on simplifying the classification of the rivers from the 9 listed in the paper to a symbolic 2-3.

Terrain Generation

The terrain generation requires a Voronoi partition of the points created through the river generation. A simple 2D Voronoi function will suit our uses as we can procedurally raise or lower the points through height values determined when running the node expansion algorithm when generating the rivers. Once the Voronoi cells have been generated, we proceed to create junctions for cells that require them. These cells will have exactly one outlet and can have any positive number of entries. The junction or confluence will take in two neighboring water entries, and may combine with another inlet to create another junction. If the flow of water at these junctions are radically different, then the connection angle will be almost perpendicular.

Figure 13: *Different trajectories of rivers corresponding to their Rosgen type. Light blue is type* ***B****, blue type* ***G****, and dark blue* ***D****.*

The terrain features themselves are a blend of the various primitives in our terrain tree leaf nodes. Primitives must have a point, segment, and curve, together known as a geometric skeleton; they must also have elevation and weight functions. The equation for weighting of primitives is as follows:

$$w(\mathbf{p}) = \frac{\left(1 - d(\mathbf{p})^2\right)^2}{r^4} \text{ if } d(\mathbf{p})^2 < r^2 \text{ else } w(\mathbf{p}) = 0.$$

d(p) in this case is just the distance from the elevation of a point to the skeleton of a primitive. Each primitive has a radius r for its area of influence around its center c.

Each patch of terrain is populated with points determined by Poisson sampling which we then use to determine the primitives that correspond with a point.

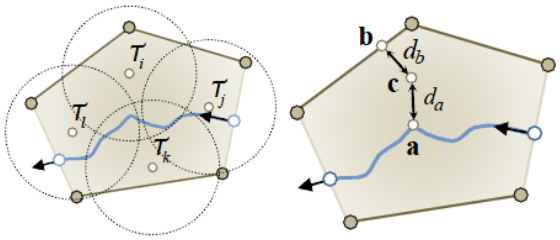


Figure 14: *Terrain primitive distribution and elevation.*

Once the terrain has been populated with blend points, the elevation of an area around a point is determined by a distance-weighted combination of elevations. This combination is the projection of the point on the set of rivers, and the projection of the point on the ridges of the Voronoi cell.

The shape of the terrain is further refined through the addition of noise which is calculated through distance check with a nearby river system and the elevation difference between that river and a ridge.

Blending involves the following equations:

$$h_{\mathcal{B}(\mathrm{A},\mathrm{B})} = \frac{w_\mathrm{A} h_\mathrm{A} + w_\mathrm{B} h_\mathrm{B}}{w_\mathrm{A} + w_\mathrm{B}} \qquad w_{\mathcal{B}(\mathrm{A},\mathrm{B})} = (w_\mathrm{A} + w_\mathrm{B})/2$$

$$h_{\mathcal{C}(\mathrm{A},\mathrm{B})} = (1-w_\mathrm{B})\, h_\mathrm{A} + w_\mathrm{B}\, h_\mathrm{B} \qquad w_{\mathcal{C}(\mathrm{A},\mathrm{B})} = (1-w_\mathrm{B})\, w_\mathrm{A} + w_\mathrm{B}^2$$

Simply put, to blend A and B, we take the average of the their respective weights to form the combined weight of A and B. The height of the A/B blend is the sum of each weighted height divided by the total weight of the two primitives.

After creating the primitives and blending them according to the selected operators, we output the final terrain mesh.

2.1.2 Maya Interface and Integration

The MEL code will be used for user interface and input mechanisms. The user will define a curve for the terrain contour, set locators along the contour as river mouths, define priority index vs. elevation importance for node expansion via a constant value, upload river and terrain slope image maps, an set the number of terrain primitives used for each cell.

The actual handling of the algorithm and creation of river network and terrain mesh will be done within the C++ API. It will receive curve data and other parameters via the MEL script, generate the appropriate data structures, and perform the necessary steps of the algorithm.

After generating the river system, the plug-in will utilize the points to create Voronoi cells and patches. The refinement of the terrain is also handled in C++, such as the terrain blending and the creation of junctions. In the end, the information passed back to Maya will be the locations and points of the vertices that comprises the terrain mesh, and the API will return the mesh into a new Maya object.

2.1.3 Software Design and Development

Spec out the objects, custom nodes, data structures and class hierarchies you plan to develop as part of the authoring tool implementation. Provide descriptions of each. List any third party software you plan to use as part of your development effort.

This procedural terrain generation tool will be implemented as a C++ plug-in. The core data structures for the river and terrain models are given below:

RiverNode
A basic node data structure will be created to hold the following river node parameters: position, elevation, priority index, river type. It will contain references to its ancestor and children nodes for traversal along a river tree network.

RiverGraph
A basic graph comprised of all the river network trees in the scene. The graph will hold references to the initial set of user-defined nodes (river mouths), and will allow the running of the river network expansion algorithms mentioned above.

SlopeMap
A 2D vector of slope values extracted from the river slope and terrain slope image inputs from the user. The class will contain a value look-up function that would take a river node's position and return to the corresponding slope value on the map.

River Node L-System
L-system used to expand a candidate node using the set of rules described above.

Terrain Construction Tree
The terrain is stored in a hierarchical representation where its surface is defined procedurally as a continuous function h(p) over the input domain. We define h using a construction tree whose leaves are primitives describing terrain fragments, and the inner nodes combine subtrees together. In this way, the elevation of a point can be defined as a combination of hierarchy of primitives.

PrimitiveNode
These hold terrain and river primitives, as well as static functions for blending and replace operators.

3rd Party Software: Boost.Polygon Voronoi Library

2.2. Target Platforms

2.2.1 Hardware

Minimum hardware configuration (i.e. processor speed, memory requirements, graphics card, etc.) required to run the tool and/or implement the effect.

Since our plug-in is built for Maya 2013, our minimum hardware configuration matches those of Maya 2013:
For 32-bit Maya:

- Windows: Intel® Pentium® 4, AMD Athlon™ processor with SSE3 instruction set support (or higher)
- 2 GB RAM
- 10 GB free hard drive space
- Certified hardware-accelerated OpenGL® graphics card

For 64-bit Maya:

- Windows and Linux: Intel Pentium 4, AMD Athlon processor with SSE3 instruction set support (or higher)
- Macintosh® computer: Macintosh computer with a Intel-based 64 bit processor
- 4 GB RAM
- 10 GB free hard drive space

2.2.2 Software

Version of Windows, Maya, OpenGL, Direct3D, etc. required to implement the tool and/or effect.

- (64-bit) Windows 7

- Maya 2012
- OpenGL 1.2+
- Microsoft Visual Studio 2010 or higher
- 3delight

2.3. Software Versions

2.3.1 Alpha Version Features (first prototype)

Tool will have working UI with some basic functionality as listed below:

- Creation of a rudimentary river network based on an input contour and river mouths
- Create child nodes by expanding initial set of river nodes
- Height calculation for every generated river node.
- Load source images for terrain and river slope maps
- Creation of a rudimentary terrain mesh with Voronoi partitioning and placeholder elevation values.

2.3.2 Beta Version Features

List the complete set of features to be included in the beta version. Describe the important development milestones. Describe the demo/test app you plan to create to show off the beta features.

For our beta version, we intend to update the mesh generated in the alpha version by processing the river and terrain slope data to calculate terrain elevation. Our river system will incorporate probabilistic expansion rules and continue to refine its classifications. For terrain generation, we will implement a data structure based on the principles of constructive solid geometry to store terrain information. We will also create a set of terrain primitives, as well as implement the blend on carve operators.

2.3.3 Description of any demos or tutorials

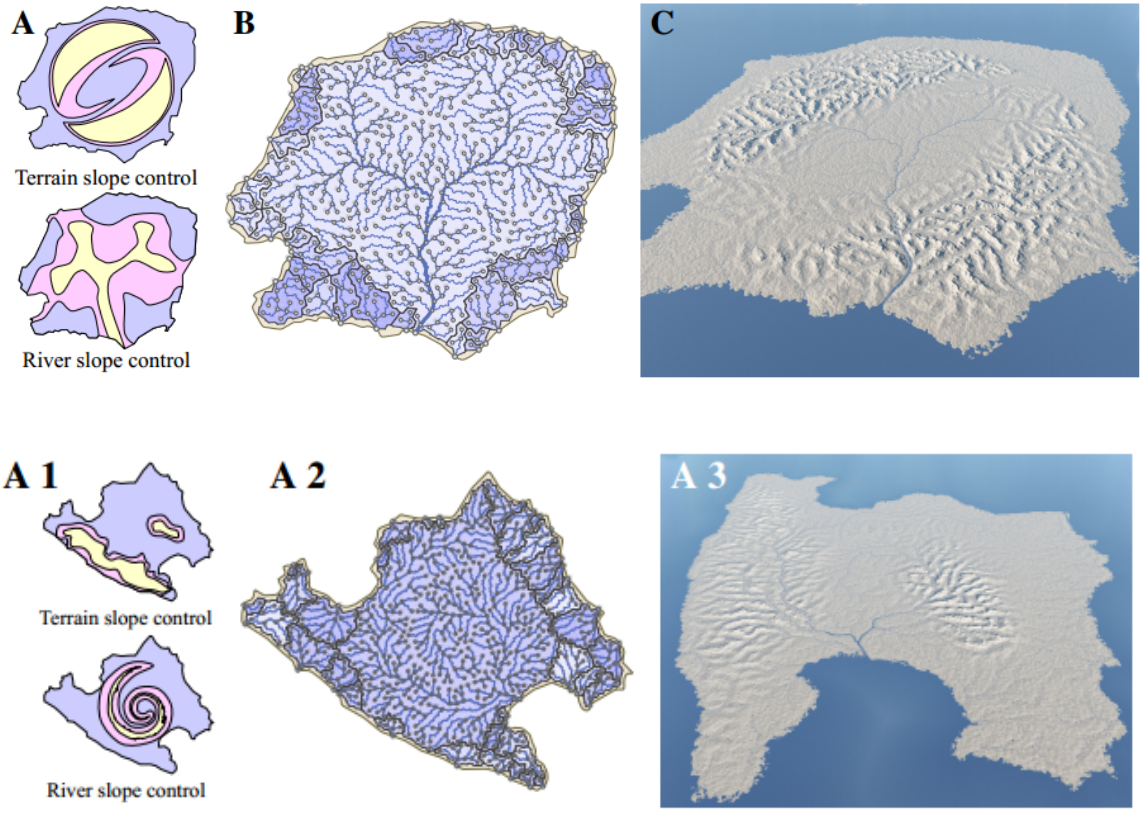
Alpha Demo: The tool will display UI and show a basic river network constructed from an input counter and river mouth nodes. The terrain will be subdivided into patches and elevated using placeholder height values. The alpha demo will demonstrate the tool workflow and give a preliminary look of what the final terrain generation will output.

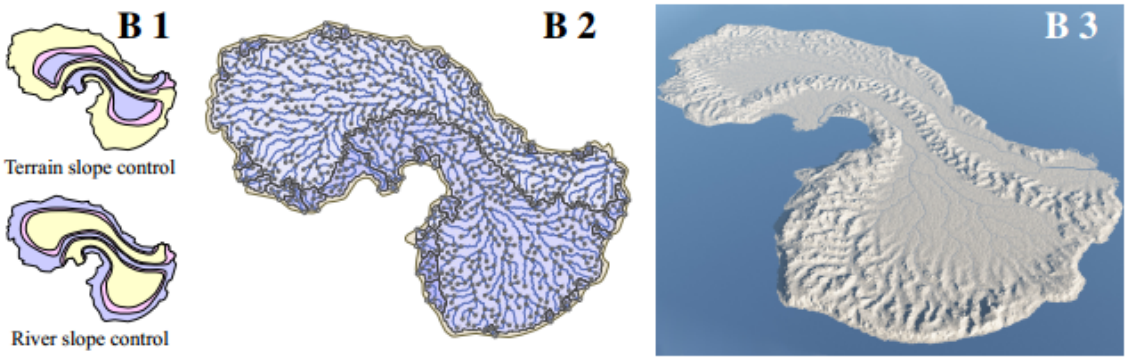
Beta Demo: The river system will be further refined with additional classifications. The terrain model will have elevations that will more accurately reflect what was input in the slope maps. The polygon mesh will incorporate blending of terrain primitives to better match the expected output.

Final Demo: This version will have bug fixes and changes to address any comments from the previous two demos. The output from this demo should have varied ridges and crests, and display a convincing river network with corresponding terrain geometry. Time permitting, we will add additional input parameters to the UI (e.g. river node priority index) to allow further user control over the generated mesh, as well as export a generated terrain mesh into an external rendering software such as 3delight to create a visually compelling scene.

2.3.4 Test Scenarios

In addition to basic testing of our tool, we will try to match our terrain map similar to ones used by the paper. The aforementioned test cases are showcased as follows:





We will input a terrain slope control and river slope control similar to the above reference diagrams and expect our tool to output a river network and terrain that resemble the resulting output.

3. WORK PLAN

3.1. Tasks

List and number all the tasks and subtasks which are necessary to develop your authoring tool. Provide separate descriptions of each task/subtask, which members of the group are assigned to it, and the expected task duration. Be as detailed as you can.

1. Build Framework (Alice)
   - 1.1. Implement user interface in MEL [1d]
   - 1.2. Create source images (river slope, terrain slope) [1d]
   - 1.3. Implement command plug-in framework in C++ [1d]
     - 1.3.1. Create code stubs for plug-in initialization, compute, etc.

2. River network generation (Alice)
   - 2.1. Create data structures
     - 2.1.1. Individual river nodes with priority index (Horton-Strahler's number), position and elevation [3d]
     - 2.1.2. River node graph [1d]
   - 2.2. Convert source images to usable data [2d]
   - 2.3. Implement L-system expansion with initial candidate nodes for hierarchical drainage network growth. [6d]
   - 2.4. Implement probabilistic rule application to determine river branch types [4d]
   - 2.5. Add compatibility checks for newly created nodes [3d]
   - 2.6. Integrate voronoi partitioning library [3d]
   - 2.7. Classify river type for each river node and edge (dependant on 3.1) [3d]

3. Terrain system generation (Cheng)

3.1. Take expanded nodes and implement voronoi partitioning (Cheng & Alice) [4d]
  3.1.1. Connect the points for each voronoi cell to get a patch.
3.2. Assign height values to voronoi cells [1d]
3.3. Create River primitives (Alice) [3d]
  3.3.1. Define river junction algorithm
  3.3.2. Connect river junctions
3.4. Implement Poisson distribution to cover terrain patch with terrain primitives. [4d]

4. CSG-Tree like datastructure (Cheng) [7d]
  4.1. Implement tree structure. [1-2 d]
  4.2. Define different terrain features (hills, valleys, etc.) [1-2 d]
  4.3. Implement blending, adding, subtracting, carving. [1-3 d]

5. Integration (Alice & Cheng)
  5.1. Test final design [3d]
  5.2. Fix bugs, add optional features [5d]

6. Optional features, time allowing (Alice & Cheng)
  6.1. User defined priority index [2d]
  6.2. Add fluid simulation for river flow [5d]
  6.3. Add additional environment geometry such as trees and grass [3d]
  6.4. Add textures and lighting [2d]
  6.5. Export to 3delight for final rendering [4d]

7. Presentations (Alice & Cheng)
  7.1. Alpha (3/19/14)
  7.2. Beta 1 (4/2/14)
  7.3. Beta 2 (4/21/14)
  7.4. Final (5/9/14)

3.1.1 Alpha Version

The following tasks will be completed for the alpha version:

- All of task 1.
- Tasks 2.1, 2.3, 2.6
- Tasks 3.1, 3.2

3.1.2 Beta Version

The following tasks will be completed in time for the beta version:

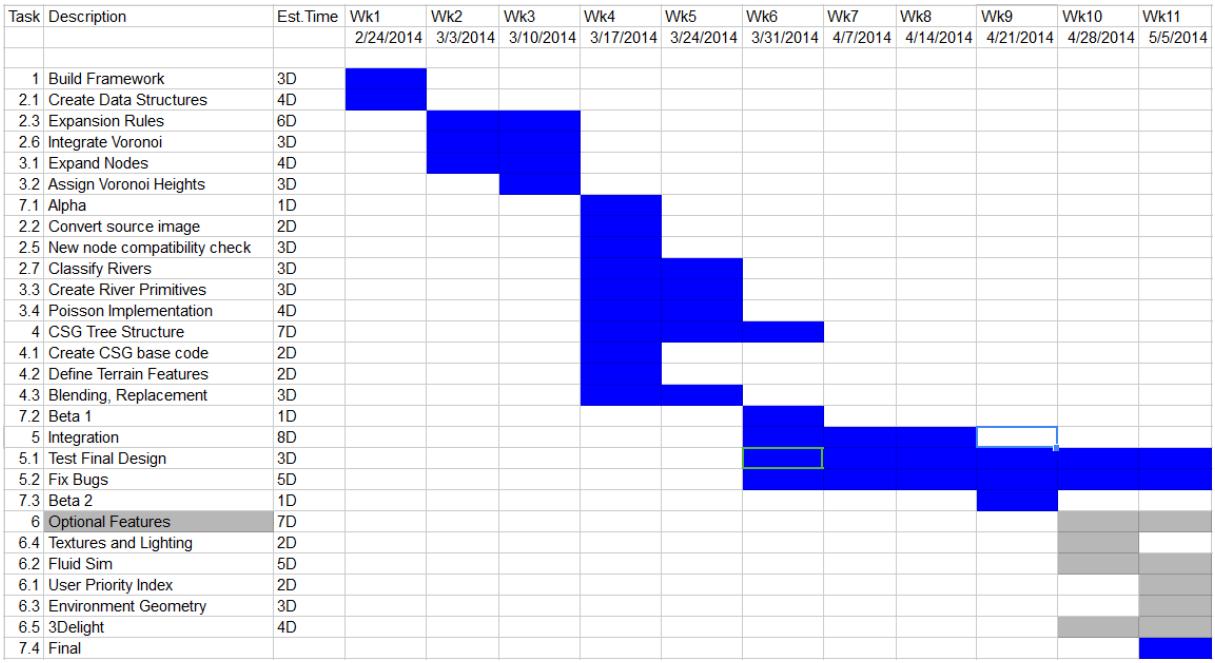
- Tasks 2.2, 2.5, 2.7
- Tasks 3.3, 3.4
- All of task 4
- All of task 5

3.1.4 Final Version

All tasks listed from sections 1-5 will be completed. We will attempt additional tasks from section 6 if time allows.

3.2. Schedule

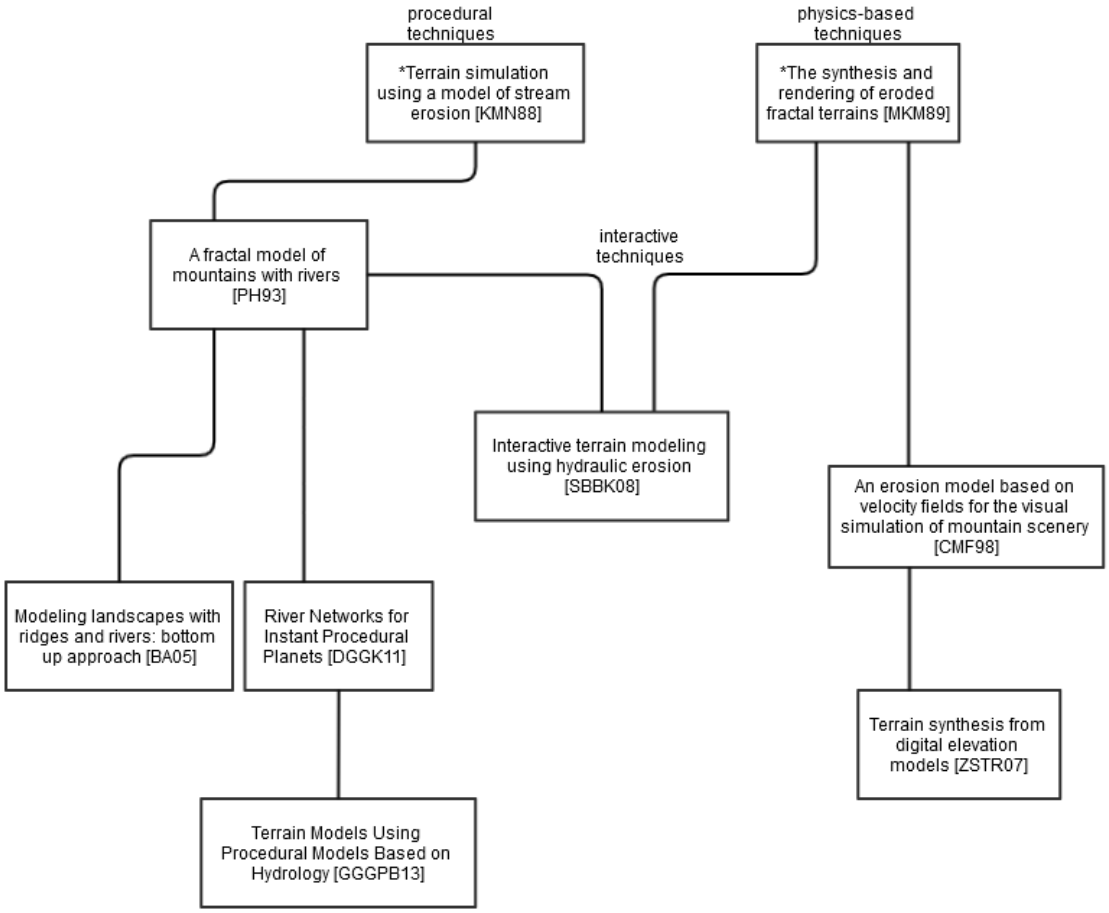
Using Microsoft Project, organize your task schedule in the form of a Gantt chart. List all the tasks (including durations) and the dates of each major milestone (e.g. alpha, beta, final).



Task	Description	Est.Time	Wk1	Wk2	Wk3	Wk4	Wk5	Wk6	Wk7	Wk8	Wk9	Wk10	Wk11
			2/24/2014	3/3/2014	3/10/2014	3/17/2014	3/24/2014	3/31/2014	4/7/2014	4/14/2014	4/21/2014	4/28/2014	5/5/2014
1	Build Framework	3D											
2.1	Create Data Structures	4D											
2.3	Expansion Rules	6D											
2.6	Integrate Voronoi	3D											
3.1	Expand Nodes	4D											
3.2	Assign Voronoi Heights	3D											
7.1	Alpha	1D											
2.2	Convert source image	2D											
2.5	New node compatibility check	3D											
2.7	Classify Rivers	3D											
3.3	Create River Primitives	3D											
3.4	Poisson Implementation	4D											
4	CSG Tree Structure	7D											
4.1	Create CSG base code	2D											
4.2	Define Terrain Features	2D											
4.3	Blending, Replacement	3D											
7.2	Beta 1	1D											
5	Integration	8D											
5.1	Test Final Design	3D											
5.2	Fix Bugs	5D											
7.3	Beta 2	1D											
6	Optional Features	7D											
6.4	Textures and Lighting	2D											
6.2	Fluid Sim	5D											
6.1	User Priority Index	2D											
6.3	Environment Geometry	3D											
6.5	3Delight	4D											
7.4	Final												

4. RELATED RESEARCH

In order to prepare the Related Research section you will need to read the papers referenced in the paper you have chosen to implement, then chain backwards through a number of levels of references until you identify/reach the seminal work. Once the seminal work has been identified, you can then chain forward to trace the evolution of the research. In this section you need to include a tree-like graph showing the evolution of the research from the seminal work to the paper you have chosen to implement along with descriptions of the enhancements/improvements/contributions made by each of the subsequent works.



Research Evolution Graph

[KMN88] KELLEY, A., M ALIN, M., AND NIELSON , G. 1988. Terrain simulation using a model of stream erosion. In *Computer Graphics 22*, 4, 263–268.

In this seminal paper, Kelley et al. describes an algorithm for generating geomorphologically correct terrain using river networks. The paper asserts that images of realistic looking terrain may be produced "while retaining a degree of database amplification as present in fractal models." [KMN88] The underlying tenet is that greater realism can be achieved from somewhat more deterministic approximations of the relief of natural landscapes, as opposed to the purely fractal methods that have been widely used previous to this paper.

The method observes that relief is primarily developed through water erosion; therefore the model developed creates topographical structure by tracking the negative space formed by a drainage system comprised of a stream and its tributaries. Due to the property of self-similarity of fractals, increasing or decreasing the number of tributaries allow the terrain to be modeled at various degrees of detail.

Because water flows along edges that are lower than surrounding landscape, the algorithm begins by creating a river network, then calculating river vertex altitudes, and finally assigning higher altitudes to surrounding mountains. All tributaries in the terrain lead to a single, main river.

Since Kelley et al's pioneering work was proposed, there have been a number of innovative methods that furthered their initial fractal-based approach. In fact, it resembles the approach of our authoring tool [GGGPB13] in that the river network is generated first and the terrain second. However, the difference is that our algorithm creates large terrains represented by a continuous procedural model from a hydrographically and geomorphologically consistent river drainage network. There is also an additional functionality of generating terrains from partial input sketches.

[MKM89] Musgrave, F., Kolb, C., Mace, R. *The Synthesis and Rendering of Eroded Fractal Terrains*. 1989

Terrain generation started out with fractals and [MKM89] expands upon fractal terrain generation by introducing erosion. The approach introduces noise synthesis which allows for local control of fractal dimensions and also attempts to introduce height fields as a new type of primitive; these two techniques are designed to deal with some issues that arise through fractal generation. An initial concern with fractals is that the bases of a fractal terrain is as sporadic as the peaks which may not be the case in most realistic situations.

Noise synthesis builds upon work done by Miller, Gardner, and Saupe, and is described as adding noise as a displacement map through tightly band-limited frequencies. Because each point is computed independently, there is no need for a global computation. The erosion is then caused by hydrology or thermal weathering.

The hydraulic method involved requires each vertex to have its own amount of water and sediment, which it can pass to a neighbor; their process then allows for water and sediment from higher regions to flow downward towards the base of the terrain, allowing for a degree of smoothness and resemblance of natural erosion. The thermal weathering accounts for the effect of loose material being knocked down and its accumulation. This is related to the material slippage technique that was later employed in [SBBK08].

The influence of [MKM89] on further research is substantial as the techniques described have been expanded upon in various works while also being central to the advancement of techniques that addresses its flaws. [GGGPB13] incorporates some procedural methods but focuses on rivers as the defining characteristic of terrain generation rather than the fractal mountains described in [MKM89], all the while attempting to solve the same issues such as erosion and smoothing.

[PH93] PRUSINKIEWICZ, P., AND HAMMEL, M. 1993. A fractal model of mountains with rivers. In *Graphics Interface*, 174–180.

This paper presents a partial solution for generating fractal mountains with rivers that incorporates a squig-curve model of a river's course into the midpoint-displacement model for mountains. While the paper shares the premise of fractal generation for mountain terrains as pioneered by [KMN88], it diverges from the original method by creating the mountain and the river system simultaneously. The novel approach combines the midpoint-displacement method for mountain generation with the squig-curve model of a non-branching river originated by Mandelbrot. The algorithm then employs a context-sensitive L-system operating on geometric objects, which then imprints the rivers into fractal terrains.

[WGG99] Wyvill, B., Guy A., and Galin, E. *Extending the csg tree - warping, blending, and boolean operations in an implicit surface modeling system.* 1999

This paper is used as reference for the data structure that holds the generated terrain data. In the study, Wyvill addresses recent developments in implicit modeling systems, which include the addition of Boolean operations reminiscent of Constructive Solid Geometry systems and space warping which provides a method of implementing deformations. He goes on to describe a novel implicit modeling system, the *BlobTree,* that can organize blending, warping and Boolean operations in a manner that will enable global and local operations to be exploited in a general and intuitive fashion.

[DGGK11] Derzapf, E., Ganster, B., Guthe, M., Klein, R. *River Networks for Instant Procedural Planets. 2011*

While procedural terrain generation has been a standard for many years, many instances lack realistic rivers. Erosion simulations can provide accurate depictions of river formation but are usually expensive in terms of computation. Therefore Derzapf et al. propose an alternative solution by following a set of rules to emulate realistic river geology. This allows for real time generation of river systems and terrain on a planetary scale. Several key points of this approach include an adaptive level of detail to allow for easy refinement, and reproducibility so that a duplicate terrain can be generated when such needs are required. The method provided by [FFC82] provides a good basis for terrain generation but rivers must be handled separately. [KMN88] features river networks but is not a real-time approach. Derzapf's technique starts with formation of the planet and then moves on to continents and river systems, this environment is also refined in real time dependant on user movements.

[SBBK08] St'ava, O., Benes, B., Brisbin, M., Krivanek, J. *Interactive Terrain Modeling Using Hydraulic Erosion*. 2008

St'ava and his associates introduce a physically-based approach to terrain modeling that allows for interactivity by the user. [SBBK08] builds off of a layered approach to terrain modeling introduced in [BF01], as it is a more appropriate approach considering the layer revealing effects of erosion. The water simulation required for the erosion is a shallow water model due to the inherent interactivity between shallow water and terrain. Erosion is handled by several algorithms: force-based, dissolution-based, and sediment slippage. Force-based erosion is applied by the naturally running water, St'ava improves upon this technique by including sediment transportation. This running water also affects the underlying soil, which forms the basis for dissolution-based erosion, in that the geology under the water is influenced in some fashion. The above forces may culminate in material slippage as the soil or sediment is now loosened and is pulled by gravity.

[ZSTR07] Zhou, H., Sun, J., Turk, G., Rehg, J. *Terrain synthesis from digital elevation models*. 2007.

With the the many advancements towards procedural terrain generation, a noticeable problem occurs when a desired style of terrain or specific terrain feature needs to be created. The parameters generally used for terrain generation do not consider singular details. [ZSTR07] improves on fractal and physical erosion techniques by introducing sketch-based specification of synthesized terrain features for more individual control and the ability to transition between different terrain types and synthesized features. By using a technique called Profile recognition and Polygon breaking Algorithm (PPA), Zhou and his associates can connect the segments produced by PPA into consecutive geology through valleys or ridges.

[R94] Rosgen, David. *A Classification of Natural Rivers*. 1994.

This paper by David Rosgen forms the scientific basis for a hydrology based terrain system as it tackles the fundamental issue of defining and understanding the processes of river system creation. It is important to distinguish between the various types of river formations as the geology and attributes of the affected terrain are influenced by a river's characteristics, thus the defining of these formations will help generate proper geology and accurate terrain. Some characteristics that are important to understand are the longitudinal profiles, the entrenchment, and the width/depth ratio.

[CMF98] Chiba, N., Muraoka, K., Fujita, K. *An Erosion Model Based on Velocity Fields for the Visual Simulation of Mountain Scenery*. 1998.

[CMF98] extends the approach of terrain generation presented by [MKM89] by determining the motion of individual water particles when calculating water flow. In previous works, this calculation was based on water flow being calculated at only one point at the slope. Because of the effect on the entire topography, a more accurate erosion is produced. The erosion is also fairly simple in that a velocity field is prepared by two-dimensional arrays, provide an initial topography, obtaining that field for the terrain, and then performing the erosion process and repeating. During this calculation, the velocity has been determined, and so can be applied to the quantity of water to get the quantity of moved sediment. While this model is accurate in terms of erosion effects, it focuses more on erosion rather than actual terrain and river generation.

[BA05] BELHADJ, F., AND AUDIBERT, P. 2005. Modeling landscapes with ridges and rivers: bottom up approach. In *GRAPHITE*, 447–450.

This paper presents a fractal based method of generating realistic models of natural landscape. As opposed to previous papers that model geometry based on erosion [SBBK08], this approach utilizes a terrain model called Digital Elevation Map that is generated by a precomputed set of ridge lines and river network. The algorithm comprises of three main steps: First, a simple and rapid process that generates the skeleton of river networks is used to initialize the terrain digital elevation map. The elevations data set is then enriched using an extension of the basic midpoint displacement model. The remaining elevation coordinates are then interpolated. The midpoint displacement extension (Midpoint Displacement's Inverse) is able to compute new elevation coordinates in order to avoid discontinuities that appear while a single midpoint displacement process is applied on a pre-filled digital elevation map. Thus, this novel algorithm allows the generation of artifact-less landscape models around a pre-filled digital elevation map.

[JEEAB13] GENEVEAUX J., GALIN, E., GUERIN, E., PEYTAVIE, A., BENES, B.

Terrain Models Using Procedural Models Based on Hydrology

This paper represents the forefront of current fractal-based procedural terrain generation technologies. It is observed that when looking at real terrains, their morphologies are structured around river networks, which divide the terrain into visual and clearly defined areas. The novel procedural approach allows users to optionally define the river mouths and sketch the most important rivers on the terrain. From that input, the system first generates the drainage river network, which is created inside the domain formed by the contour and is represented as a geometric graph. The output of this river network generator is a set of 3D polylines with increasing elevation from the outlet to the spring. The algorithm then extracts the graph topology and geometry that is used for terrain generation, and decomposes the terrain into a set of patches using Voronoi cells corresponding to the nodes of the river graph. Two user-defined maps— the river slope map and terrain slope map control the shape of the rivers as well as the locations of mountains and flatlands. Finally, the algorithm compiles the generated data and outputs the continuous-terrain model. This paper exceeds its predecessors in that not only does it generate geologically accurate river-defined terrains, it also eliminates the need for specialised river input data by enabling the procedural generation of a river network given optional user input and a set of adjustable parameters.

REFERENCES

[JEEAB13] GENEVEAUX J., GALIN, E., GUERIN, E., PEYTAVIE, A., BENES, B. Terrain Models Using Procedural Models Based on Hydrology

[DGGK11] DERZAPF, E., G ANSTER, B., G UTHE, M., AND KLEIN, R. 2011. River networks for instant procedural planets. *Comput. Graph. Forum 30,* 7, 2031–2040.

[SBBK08] STAVA, O., BENES, B., BRISBIN, M., AND KRIVANEK, J. 2008. Interactive terrain modeling using hydraulic erosion. In *Symposium on Computer Animation*, 201–210.

[ZSTR07] ZHOU, H., S UN, J., T URK, G., AND REHG, J. M. 2007. Terrain synthesis from digital elevation models. *IEEE Transactions on Visualization and Computer Graphics 13*, 4, 834–848.

[BA05] BELHADJ, F., AND AUDIBERT, P. 2005. Modeling landscapes with ridges and rivers: bottom up approach. In *GRAPHITE*, 447–450.

[CMF98] CHIBA, N., M URAOKA, K., AND FUJITA, K. 1998. An erosion model based on velocity fields for the visual simulation of mountain scenery. *Journal of Vis. and Comp. Anim. 9,* 4, 185–194.

[PH93] PRUSINKIEWICZ, P., AND HAMMEL, M. 1993. A fractal model of mountains with rivers. In *Graphics Interface*, 174–180.

[MKM89] MUSGRAVE, F. K., KOLB, C. E., AND MACE, R. S. 1989. The synthesis and rendering of eroded fractal terrains. In *Computer Graphics 23*, 3, 41–50.

[KMN88] KELLEY, A., M ALIN, M., AND NIELSON , G. 1988. Terrain simulation using a model of stream erosion. In *Computer Graphics 22*, 4, 263–268.