Realistic Terrain Generation Based on River Space Colonisation

Reuben Speirs
University of Auckland

November 2020

Abstract—**Procedural terrain generation for river watersheds is computationally expensive. To achieve realistic terrain output the standard approach is to perform erosion simulations on random noise maps to shape terrain landscape and discover river paths. This paper presents a novel approach to procedurally generating realistic watershed terrain using the tree space colonisation algorithm. River networks are generated by growing tributaries toward empty space in the terrain represented by attraction point primitives within a terrain space. During growth, ridges formed by Catmull-Rom splines cut through the terrain representing mountainous features that tributaries can not grow across. Multiple rivers can grow in the terrain competing for attraction points to grow towards. Tributaries will increase in height as they grow further away from the mouth of the river. At each point along the river a disk primitive is used to encapsulate a small area terrain whose height is calculated and blended with other terrain. This produces realistic river watershed landscapes based on the effect of river hydrological processes on surrounding terrain heights while removing the need to run algorithm steps that simulate river erosion processes.**

1. Introduction

Traditionally landscapes in games and movies are drawn and modelled by 3D artists which requires a large time investment to create scenes that are realistic and detailed. A movie may require scenery of some terrain that is not possible to be filmed due to its location or that it is not similar to earth like terrain. Certain games also have large scale open worlds for players to explore requiring very large scale terrain to be produced that doesn't lack details the player would notice. Procedural generation has long been an area of focus as algorithmic approaches to generating landscapes can create large and unique areas of detailed terrain that require much less time investment for artists. Generation can be highly customisable allowing artists to still keep a strong degree of control over how terrain is formed, for instance setting a constraint such as the maximum height of the terrain can be used to generate terrain with very flat or very mountainous profiles and apply this affect over large areas without the need for any extra user input.

Generating terrain that includes physically realistic river paths with realistic terrain is challenging. Terrain heights are greatly impacted by river hydrological processes; without implementing terrain erosion that simulates this process, terrain generated will not be physically realistic. This paper covers an algorithmic approach to generating a terrain height map based off the river path ways created by the tree space colonisation algorithm. Terrain heights are influenced by the height of the rivers within local proximity simulating the affect of the river erosion process on nearby terrain.

3D terrain can be generated from a height map in which different height values are assigned different shades of colours depending on the height of the terrain. For example, an increase in the intensity of a grey pixel may represent an increase in the pixels terrain height, or a map of features may use different colours such as blue to represent a specific terrain feature such as a body of water. 3D terrain can then be rendered with each vertex having a height represented by these colour values [17].

Early procedural terrain generation methods such as fractal generation could be used to create a height map that can be used to render 3D terrain [6]. Though a more recent approach is generating an underlying noise map which is used as a source of random clusters whose values are interpreted as differences in height. The height maps formed from these methods are rendered to create realistic complete 3D terrains with very little artist input [5].

Using a noise function to generate terrain height maps and 3D terrain results in unrealistic generation due to noise functions not creating height maps that account for the erosion processes that shape natural geography. These functions also produce terrain that is homogenous at scale and require random erosion algorithms to shape the terrain surface otherwise generation will appear to follow repetitive patterns. Perlin noise is commonly used to generate height maps which create smooth and visually unnatural terrain. One method that tries to solve this involves generating a complete terrain then simulating erosion processes by dropping many small water droplets over the terrain surface [3]. Droplets fall onto the terrain surface and aided by gravity, follow the steepest paths of descent while carving and reshaping terrain until slopes flatten and movement stops simulating the affect of river erosion. This can also be done using software agents which move over the landscape and apply erosion algorithms to shape the local terrain they pass over [4]. Another approach is to generate a watershed and fit it onto a smooth terrain then use geomorphology models to interpolate how the terrain would change with the presence of a river mapped onto its surface [9], [21].

These methods are very computationally expensive and take an unreasonable amount of time or resources to compute large landscapes [3], [9], [11], [17]. Striking a balance between realistic terrain and efficient generation is difficult [9]. One way to help solve this issue is creating a watershed populated with river networks then generating the terrain around the river features. This helps drastically reduce the amount computation needed while also factoring in the effect of river erosion on terrain [5].

We propose a novel approach to generating the river terrain of a watershed using a modified tree space colonisation algorithm [20]. A blank height map is created that contains a set of random attraction points and impassible ridge lines that the shape of river generation, the algorithm generates the skeleton of a river by growing river tributaries between points. River networks are labelled and used to interpolate the heights of local terrain. This approach produces a greyscale pixel map that contains height information of our generated terrain and a 3D rendering of the terrain.

By removing the need to generate terrain then erode the landscape we aim to decrease the computation time for large scale terrain generation and create realistic landscapes built around the erosion caused by geomorphic actions in a watershed populated with river networks.

2. Related work

This paper introduces a novel approach to producing procedural watershed terrains. River skeletons are generated using a space colonisation algorithm where rivers grow towards attraction points within a user defined terrain. Terrain height maps are produced in which terrain elevation is influenced by the height of surrounding river skeletons. By having river networks shape local terrain height removes the need to simulate erosion processes to find locations for river flow when creating noise map based terrain models.

2.1. Introductory work

Opening chapters of the book "Texturing modelling: a procedural approach" written by Ebert et al. introduces and discusses the history and basics of procedural generation. This introduces how early implementations of terrain generation was inspired and developed through the use of procedural texture generation [5]. Examples of texture generation were used to help convey the process of procedural generation and how it can be adapted to include algorithmic processes to simulate the affects of erosion during initial height map generation. The book goes into depth about the use of Perlin noise for terrain height map generation and discusses the impact and use of procedural terrain generation in the movies/game industry. Rose et al. goes on to compare the advantages and disadvantages of using different noise generation methods such as Perlin noise for generating terrain. The paper further outlines the significance of different user requirements and their effect on terrain generation. For example, depending on the generation outcome the user wants, it may be more desirable to use software agents which allow for more user control over generation parameters but create more algorithmic complexity and terrain is difficult to keep random decreasing generation performance [17].

2.2. Early terrain generation techniques

Early terrain generation techniques used Fractal Brownian motion to create terrain efficiently [6]. The model uses a path finding algorithm to map out paths of motion with different step sizes changing line resolution; larger resolutions created a more detailed line path. This was useful as line data can be written to a height map and used to generate smooth terrain; the map heights differed depending on the line motion. This is an early representation of how terrain could be procedurally generated and modelled using a height map. Later work by K Perlin opted to not use Fractal Brownian motion and instead the paper details the implementation of a pixel steam editor filter used to change the properties of pixels. Each individual pixel has its location and colour altered depending on patterns created by a random noise function [14]. This is used for texture mapping with convincing detail while the underlying algorithm is simple and efficient. Generated textures and their texture maps don't need to be the same shape while still being able successfully mapping together. This independence of each other meant that different textures could be mapped to any variety of shapes as long as they met certain criteria. The paper also details how to use different generated noise patterns to create a variety of different textures. Textures could then be implemented so they map onto an objects surface to display a variety of shapes and patterns. For example, if a texture was generated areas of low elevation then it is mapped to the surface of an object there would be visible dips in objects surface.

2.3. First implementations of terrain shaping using simulated erosion processes

Kelley et al. simulates stream erosion on un-eroded terrain surface to create an approximation generation of natural terrain [9]. Implemented by creating a surface under tension which maps a watershed terrain to an un-eroded terrain surface. This is used to alter the terrain surfaces heights dependent on the levels of tension between the local areas. Prusinkiewicz et al. improves on this work by creating two terrain models using fractal generation [16]. A watershed and a mountain terrain are mapped together and the terrain close to river paths is altered dependent on an erosion function. Roudier et al. instead uses noise map generation to create un-eroded terrain; a river is created on the surface of the terrain depending on several erosion factors [19]. The height map is then altered depending on the location of river generating across its surface. This method also discusses the use and involvement of several erosion techniques used to generate the final terrain output such as taking into account the geography of the terrain and how that will affect the erosion taking place. It also shows that there are many erosion factors that need to be accounted for when trying to create terrain that is as realistic as possible, this demonstrates that there is a trade off between an increase in realism, computation time and resources when generating terrain. Certain erosion factors can be ignored if there is no noticeable impact on the terrain shape to save on computation time while terrain still appears visually realistic.

2.4. Emphasis on more user control over generation

Olsen and Jacob introduce a traditional approach to terrain generation by creating noise maps then simulating erosion affects on the un-eroded noise [12]. A height map is created using a pink noise function and the generation process attempts to create terrain that is heavily impacted by user set parameters to limit the need for erosion simulation as it is computationally expensive. This was illustrated as algorithms to simulate erosion factors such as hydrophilic erosion were optimised and adapted to be run in real time but they were still computationally expensive and slow; a new algorithm to perform mid point displacement to generate terrain was suggested. This demonstrated a reduction in erosion simulation needed after terrain generation reducing the reliance on erosion to create a realistic looking terrain. Doran et al. adds further user control by implementing a method to generate terrain with software agents controlled by user constraints [4]. These agents can be run in parallel and move over the terrain surface to smooth elevation in different sections of the terrain and add landscape features depending on their attributes and location. This method still requires agents to move across previously generated terrain.

2.5. Terrain shaped by water flow

N Chiba introduces a method of terrain generation without producing an underlying noise map and instead uses ridge lines to influence terrain heights [2]. The algorithm first creates a 2D array height map containing a set of lines that represent ridges. Another algorithm is then used to assign elevations to the end points of the ridge lines. During this process, another algorithm simulates raindrops over the terrain surface which carve out the terrain to create more natural looking ridges. Lastly the method uses a final algorithm to find the heights of terrain which surround ridge. Later work by Chiba et al. extends the use of rain droplet erosion by creating a quasi-physically based model to generate terrain in which eroded mountains are shaped by the velocity of water flow [3]. Simulation of water droplets falling down a mountain side is used to simulate the erosion taking place. Erosion is based on the velocity of the water flow off mountains into surrounding rivers through surface run-off. The model then uses the water flow to calculate the transportation of sediment and build up surrounding terrain around areas of high water flow.

2.6. Terrain elevation influenced by local river features

Research by ST Teoh discusses how erosion greatly impacts the appearance of a landscape, however the accurate generation of rivers and other coastal features were not available using current generation techniques [21]. ST Teoh implements algorithms to generate terrain involving river features such as meanders, deltas, coastal cliffs and beaches. Without these features terrain looks unnatural; highlighting the importance of the erosion processes which shape landscapes and create these features. If realistic procedural generation is to look believable it must includes river features. Génevaux et al. introduces a framework to generate terrain based on river hydrology [7]. Using a sketch and some user parameters to control generation, it is possible to build a realistic drainage network over an input domain. While this is promising, it requires user effort to sketch out a copy of plausible river locations, the algorithm then uses the sketch to map river positions and builds estimated surrounding terrain. Terrain elevation is influenced by the structure of sketched river so terrain will naturally form important river features. Further research by Soon Tee Teoh creates a fractal based procedural terrain model named RiverLand that performs fractal based terrain generation while creating a consistent river terrain representation and allowing the user a high degree of customisation over the terrain shape and features [22].

2.7. Space colonisation algorithm used to create tree structures

To generate realistic terrain based on the location of river features, first a physically accurate river skeleton must be created. Tree space colonisation introduced by Runions et al. creates a tree based on the space colonisation algorithm where tree branches compete for available leaves [20]. Starting at a root node, a branch is created and attracted to randomly located leaves in the surrounding area. When a branch grows and reaches a leaves position, the leaf is removed from the growth area so other branches can not grow towards an already reached leaf; branches growth will continue by growing towards the next nearest leaf. Each unreached leaf in the growth area attracts the growth of a branch producing a gravitational like pull causing branches to grow towards leaves in an arch rather than a straight line. This creates a smooth branch curve which appears more natural and realistic. Patrick et al. extends this implementation by generating trees that are sensitive to obstacles in their environment [13]. Genus type as well as natural and artificial influences such as obstacles in the way of branch grow affect; implementation to account for these factors is heavily focused on. For example, in enclosed spaces where a tree is surrounded by walls, trees will grow narrower to fit into their surroundings. Feedback from modellers via a study showed that users felt tree growth was visually realistic and gave the desired generation and environment interaction.

2.8. Bottom up approaches to generating terrain

Belhadj et al. uses a bottom up approach to building landscapes where ridges are drawn based on points given by the user and then river starting points are spread randomly along these ridges [1]. Starting points are given a beginning mass which determines the width and depth of the river. Subject to the force of gravity, these points fall following Fractal Brownian Motion while the force of friction is applied and slows point movement to an eventual halt. The river stops growing once the point stops moving or grows over the designated landscape boundary. This creates river skeletons which use midpoint displacement to interpret surrounding terrain heights based on local height of rivers. This process generates the final terrain heights without running an extra erosion simulations as terrain heights are solely based on the erosion caused by river generation. In contrast Lau et al. implements a bottom up approach to terrain generation by reconstructing river networks from broken river segments [10]. Individual river segments are used to interpolate surrounding terrain heights while areas around the tips of these rivers have reduced affects on local heights. The height differences between river segment tips allow for physically accurate river paths to be created between disjoint segments to complete river terrain.

2.9. Alternative to terrain height maps

Work by Génevaux et al. introduces a construction tree data structure as a way to store terrain generation information [8]. Internal nodes of the construction tree represent operators such as blend and replace that are applied to terrain primitives to form complex landscapes. Leaf nodes store the terrain primitives and these may represent areas of terrain such as mountains or valleys. This provides an alternative and efficient approach to storing terrain information and does not require rendering terrain from a height map. Further work by Peytavie et al. extends this approach by introducing a method of generating a large scale river scape with high levels of graphical detail using the construction tree implementation [15]. River beds are carved into terrain primitives using inner node blending and replacement operators.

3. Space colonisation

3.1. Overview

To create the layout of a river skeleton in an area of terrain, an adaption of a tree space colonisation is used to generate tree skeletons that share a similar structure to real life river networks [20].

Repurposing of the tree space colonisation algorithm and how it's used to generate and grow river skeletons is covered in section 4. Section 4.1 details the use and effect of ridge line impact on the shape of river growth. Terrain elevations are defined by their distance from local rivers, detailed in section 4.2.

4. River space colonisation

River skeletons are generated through the extension and repurposing of the tree Space Colonisation algorithm [20]. Figure 1 provides an overview of the initial stages of river growth.

Before the river generation begins a set of attraction points are created inside a terrain boundary. The terrain boundary is a space which encompasses a square area in which the river skeleton is allowed to grow. Attraction points are randomly dispersed throughout the terrain boundary based on a random seed, this seed is also used for all random generation in the model so it can be used as a way to produce deterministic terrain results. The final set of attraction points represent empty space and are locations that the river tributaries grow towards.

Once attraction points populate the terrain space, river mouths that represent the starting position for river growth are found along the terrain boundary. From the set of all attraction points, the closest point terrain boundary is a valid river mouth position. For example if two river mouths are meant to be created then the second closest point to the terrain boundary will become the second river mouth; this pattern continues as the number of rivers increases. Once a river mouth is set the point it occupies is removed from the set of attraction points. The reason for generating river mouths as close as possible to the terrain boundary is so that no attraction points that rivers grow towards will be behind river mouths. The final shape of the terrain space intersects all river mouths and contains all attraction points. Each of the river mouths has its height set to zero which is deemed to be sea level.

The mouth of a river is the starting position from which the first tributary is created and it will start growing towards attraction points. A tributary starts with a direction of growth equal to the direction of its parent growth. A parent is the tributary a current tributary has been set to grow off. As tributaries grow out from the river mouth their direction of growth is altered so it does not grow directly towards an attraction point which produces a realistic curved river path. To alter the direction of tributary growth, for each attraction point, the closest river tributary is found and set to be attracted to the points position. The more attraction points a tributary is attracted to the less direct the growth will be towards a point.

The final direction of growth is denoted by $\delta(x)$, the direction of growth of the parent tributary is denoted $\delta(y)$ and the number of attraction points is denoted n:

$$\delta(x) = \left\| \frac{\delta(y)}{n+1} \right\|$$

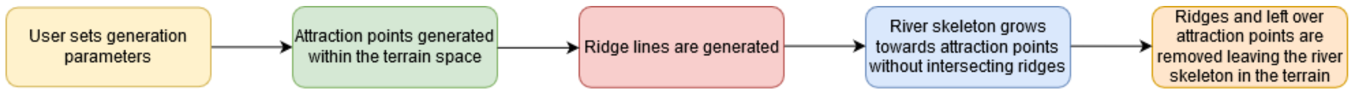
User sets generation parameters
Attraction points generated within the terrain space
Ridge lines are generated
River skeleton grows towards attraction points without intersecting ridges
Ridges and left over attraction points are removed leaving the river skeleton in the terrain

Figure 1. Flow chart detailing the initial stages of river skeleton growth.

Each tributary also has a height which is dependant on the distance to the mouth of the river it grew from. This height is equal to the distance travelled from the end position of the tributary to the position of the river mouth. The height of the tributary is denoted $h(x)$, the distance from the start position of the tributary to its end position is referred to as $d(x)$ and the height of the parent tributary is $h(y)$. Each tributary remembers its height meaning that parent height plus the new tributaries length will equal the distance from the root to the end of the new growth.

$$h(x) = h(y) + d(x)$$

From the mouth of a river, tributaries grow towards attraction points in their local area creating a tree structure used to represent a river skeleton. Tributaries from multiple rivers will compete for attraction points and once a point is reached by a tributary it is removed from the terrain space and no further tributaries can grow towards that point. Once all the attraction points have been reached and removed then the river growth finishes and the tributaries form the complete skeleton of the river.

4.1. Ridge lines

The user is able to enable ridges that randomly generate within the terrain boundary. Ridges are treated as sections of mountainous terrain which may contain river sources so it would not be physically accurate for rivers to grow over these areas. During initial river growth, ridge lines represent additional boundaries which a tributary cannot intersect when growing towards an attraction point.

While a river skeletons is growing, for each attraction point within the terrain boundary, only the closest tributary to a point is eligible for growth towards the points position. The competition for attraction points occur because there may be multiple rivers and their tributaries trying to grow towards a finite amount of points. For example if a tributary reaches a certain point, the growth of the new tributary may result in it being closer to a point that another tributary attempting growing towards. The new tributary will be defined as the current candidate for growth towards the attraction point and the other tributary will have to start growing towards a different attraction point. During this growth phase each tributary has its starting position and the position of the attraction point it is wanting to grow towards checked to see if a line drawn between the two positions does not intersect with a ridge line. If a new tributary created between the two positions does intersect with a ridge then growth towards the attraction point is halted and the point will search the remaining set of tributaries to check for others that may have valid growth paths.

Each ridge is defined as a section of terrain that is near the maximum height of the terrain. Ridges are are shaped with a centripetal Catmull-Rom spline between two points to create a single smooth curve.

Figure 2 illustrates an example of the effect of four ridges on the growth of a river skeleton. Since river tributaries are not allowed to intersect with ridge lines, this causes some attraction points inaccessible and not reached. Once river growth finishes, left over attraction points and ridges are removed from the terrain surface leaving small white area of terrain without river growth. This are will generate terrain with large elevations due the lower influence of rivers impacting terrain height.

4.2. Finding terrain heights

As rivers increase in height so does the surrounding terrain. For each section of the river skeleton, a disk primitive encompasses an area of terrain which is influenced by the height of the river. Terrain within the disk that lies further away from the river is less effected by river erosion processes and therefore will have a greater height. Terrain heights and weights are found using equations produced by Génevaux et al. [8].

The height of a point in the terrain within the disk primitives area is denoted by $h(x)$. The distance between the terrain point and the river is denoted $d(x)$. Cross section height of the terrain point is represented by c. The height of the river is denoted by $h(z)$

$$h(x) = h(z) + c \circ d(x)$$

4.2.1. Weighting. Terrain within the disk primitive has its height effected by branching tributaries or other rivers that have grown close by. While finding terrain heights, disk primitives are likely to overlap, so to blend heights a weight is associated with each terrain point which decreases the further terrain is from the river. This is because the further the terrain is from the river the less the effect of river erosion processes shaping terrain.

$d(x)$ denotes the distance the point of terrain is from the river. The weight $\alpha(x)$ is the composition of the terrain distance with a smooth fall-off filtering function g

$$\alpha(x) = g \circ d(x)$$

The smooth fall-off filter introduced by Wyille et al. and implemented by Génevaux et al. is used to determine the influence a river height in the centre of a disk primitive has on the height of surrounding terrain [23], [8]. The distance from a terrain point to the river point x is divided by the

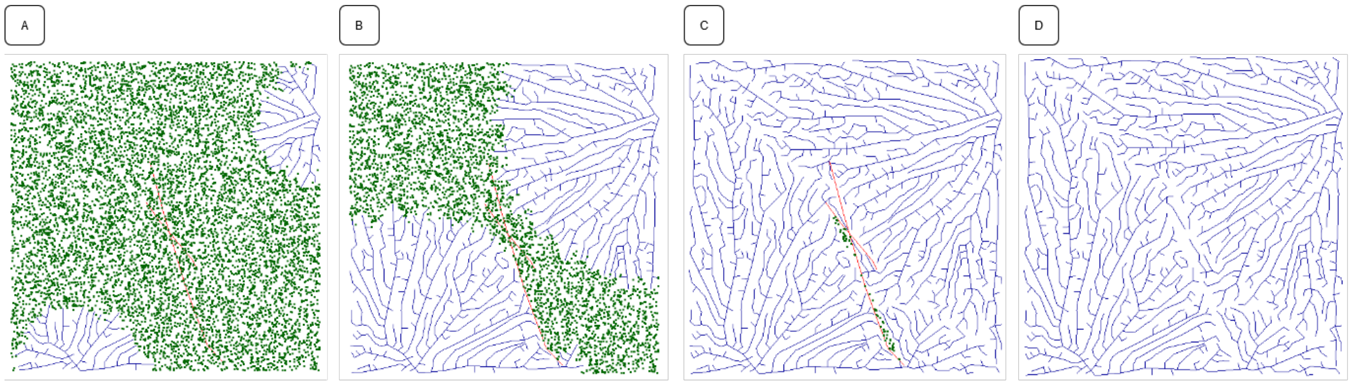

Figure 2. Green dots represent attraction points, red curves represent ridges and blue lines represent the tributary segments of the rive skeleton. A) Shows the beginning stage of river growth, B/C) Illustrate the affect of ridges on the growing shape of both rivers. A group attraction points cannot be reached due to tributary growth

maximum distance between the river point and the edge of the disk primitive r.

$$g(x) = \begin{cases} \left(1 - \left(\frac{x}{r}\right)^2\right)^3 & \text{if } x < r. \\ 0 & \text{otherwise.} \end{cases}$$

4.2.2. Blending. When terrain heights are being calculated an overlap between disk primitives can occur resulting in terrain heights being recalculated. To account for this a terrain point has its new weight and new height calculated which is then blended with its previous weight and previous height. A new weight is defined by α_A and new height by f_A. Old weight α_B and old height f_B. The terrain point then has its final height f and final weight α set.

$$f = \frac{\alpha_A f_A + \alpha_B f_B}{\alpha_A + \alpha_B}, \quad \alpha = \alpha_A + \alpha_B$$

4.3. Initialising the river skeleton

Before the initial river is built, attraction points are generated and set as root nodes if the length of the root node list is less than the number of rivers specified by the user. Then for each new attraction point, if the list of root nodes is full then there is a possibility that the new point is closer to the terrain boundary than one of the points in the root set. Each root node must be a point that is closer to the terrain boundary than any attraction point and if not then it will be replaced. To make the check easier, every point in the root set is checked to find the root with the closest x or y position to the centre of the terrain. Every time an attraction point is generated, it is checked against the closest root to the terrain centre to see if its x or y position closer to the terrain boundary than the root. If a new attraction point is found further from the terrain centre than a root node it will be set to become a root, now the new root point might not be the root closest to the centre of the terrain so the complete set of root positions must be checked again. Once a new root, closest to the terrain centre is found, it is recorded for future checks against other randomly generated attraction points. This makes it is easy to check whether new attraction points may be eligible to become a new root node. Once a root is set to be removed from the root list, it is then added to the attraction point list.

The reason we take the closest x, y position to the boundary instead of the distance of the point from the centre of the terrain is because the distance from the centre to the corners of the terrain will be higher than the distance to a position that is on a parallel axis to the terrain centre. This will mean that the roots of the rivers will more likely randomly generate and clump in the corners of the terrain.

After the attraction points and roots are found, ridges are randomly generated as obstacles that the future tributaries will not be allowed to intersect when growing towards points. A number of ridges will be generated depending on how many were defined by the user. A resolution value is also set which defines how many lines make up the complete ridge curve. The higher the resolution of the spline curve, the greater the number of points that define the shape of the curve. If the resolution is low the line will be more angular in shape.

Once the attraction points, roots and ridges are found then the initial tributaries are created. For every root in the terrain area a tributary is created at the root. This tributary contains various information such as its start and end position, direction of growth, starting height of and end height. In the case for the first tributary, its start and end position is the roots position and its direction defaults to be moving up the y axis. The root is the mouth of a river deemed to be located at sea level so the start and end heights of the starting tributary are also zero. The final boundary of the terrain is the area which intersects all river mouths and encompasses all attraction points.

During the initial build phase of the river, each of the randomly generated roots is checked to see if its possible to grow a tributary towards the nearest attraction point. Each new tributary is checked to make sure that there is an attraction point within a user defined maximum growth distance. For the first tributary extending from the river mouth, if it cannot find an attraction point to grow towards due to all points being further away from the tributary then the max distance limit, to make sure it will start growing, a new tributary is found that extends towards the centre of the terrain. The length of this tributary is defined by the user set branch length, its new position found and set as the end position, its height found which is equal to the height of the start position plus the distance travelled to the end position. This new tributary is then added to the list of root branches so it can be checked to see if it has a valid attraction point to grow towards. Once all tributaries have been checked and are able to grow towards an attraction point then the growth phase of tree space colonisation begins.

4.4. Growing the river skeleton

For each attraction point in the terrain, each tributary is checked to see if the point is in a valid location for growth. If growth is possible then a line drawn between the end position of the tributary and the position of the attraction point is checked to see if it will intersect with a ridge line. If there is an intersection then tributary growth is not allowed and other tributaries are checked for valid growth paths. If there is no intersection then the distance from the tributaries end position to the attraction point is checked to see if it's smaller than all previous tributaries growth distance. Once all tributaries have been checked the one with the smallest growth distance will have a new tributary line drawn from its end position towards the attraction point. If the new line drawn ends within the user set kill distance the attraction point is marked for deletion and removed from the set of points that tributaries are able to grow towards.

A counter is incremented every time a tributary is associated as being within the closest distance of an attraction point. Once all attraction points have been searched through and all closest tributaries set, for every tributary a new tributary is created if its counter is greater than one. The current tributary is labelled as the parent and a new tributary has its starting position as the end position of its parent and its start direction the same as the parent direction. Each new tributary also has an ending direction set that is the direction of its parent divided by its parents counter. The greater parent counter the higher number of attraction points in the local area that affect the direction of growth of a new tributary. A high counter will cause new growth to be pulled towards many attraction points simultaneously making its new growth direction less direct towards any attraction point. This produces arching growth paths.

Tributaries use their positions to find their river heights. The end position of a newly formed tributary p_e is found using its starting position p_s plus direction of growth $\delta(x)$ multiplied by the user defined branch length l.

$$p_e = p_s + (\delta(x) * l)$$

This position is then used to calculate the new branch height. The new height is set as the height of the end position of the tributary. This creates a change in height from the start position to the end position used to portray an increase in river height as you move upstream. The end height h_e of the tributary is equal to the starting point height h_s plus the distance between the start and end position of the tributary $d(x)$.

$$h_e = h_s + d(x)$$

Once a new tributary is successfully created it is then added to a list of remain tributaries that are eligible for future growth. The parent tributary has its counter reset so future growth from its end position is not affected by previously reached attraction points.

5. Implementation

5.1. Overview

There is a large amount of influence a user can have on the scale and shape of the terrain generation by adjusting various growth parameters discussed in section 5.2. Section 4.1 details the use and affect of ridge line impact the shape of river growth. Terrain elevations are defined by their distance from local rivers, detailed in section 4.2.

5.2. User parameters

There are several defined parameters that control and affect the initial river skeleton growth. First the user must choose how many attraction points are generated within the terrain boundary. The amount of points generated will increase the amount of river tributaries and the complexity of the river shape. A maximum growth distance is set which controls the maximum distance a tributary is able to grow towards an attraction point. If a tributary is unable to grow towards any point in the terrain boundary due to the distance between the nearest point and the starting position of the new tributary being greater than the maximum distance, then that tributaries growth is stopped indefinitely. Minimum distance defines the size of the kill radius around an attraction point; when a tributary grows towards its nearest point, if the end position of the tributary stops within the kill radius of a point then the point is marked as reached and removed from the set of attraction points. The user has a strong degree of control over the terrain generation as the increase in kill distance can cause river skeletons to be

The initial terrain boundary the river can grow within is set by a minimum x and y position representing the bottom left hand corner of the terrain space and a maximum x and y position representing the top right hand corner of the space. This gives a rectangle area in which all attraction points have their random positions restricted to being generated within these terrain boundaries; the growth of rivers and the effect they have on terrain is not allowed to continue outside these boundaries. The number of rivers control how many river roots will be generated, a new river is created from each river mouth and the mouth represents the end position of the river before water flows out to sea.

The user defined ridge number defines how many ridges will be randomly generated within the terrain space. Ridges are defined by Catmull-Rom splines between a pair of points in the terrain space and a ridge definition is defined that controls the definition of the Catmull-Rom spline. Ridge definition controls the smoothness of the ridge curve, for example if the definition is low then the amount of lines that make up the spline are low. The shape of the ridge is determined by the final growth of the river skeleton, as areas where ridge lines occurred during the growth phase will have a lower density of river tributaries in the area resulting in higher terrain elevation.

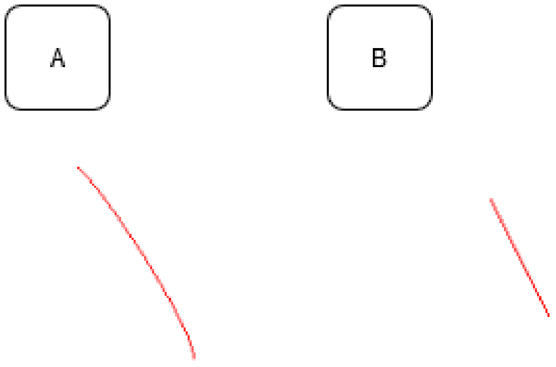

Figure 3. Shows the affect the user has on ridge line smoothness by increase ridge resolution. A illustrates a ridge with a high resolution spline representation resulting in a slight curve. B shows the same ridge with a very long resolution resulting in the ridge appearing as a straight line.

Lastly the user has the ability to alter the terrain seed. This value is used for all random generation components in the project making the terrain generation deterministic. If a seed is saved from a desirable terrain generation then that seed can be used to perfectly generate the same terrain on a different machine.

5.3. Drawing the river skeleton

While river growth occurs attraction points and the growing river skeleton are drawn to an OpenGL window and the rendering is used to interpolate terrain heights.

Tributaries are drawn as lines between their starting position and end position with each tributary limited to being within 80% of the maximum height of the terrain. The reason for this limit is that terrain will be generated around the river based on local river heights. Terrain further from the river will have a greater height than terrain close to the river so a limit is used so that the terrain height does not grow over the maximum set terrain height.

The start and end positions have their rendered colours adjusted based on their heights. Each point has a colour set in the RGBA colour space and the alpha component is used to represent the height of the river. When OpenGl draws a line between the two positions, the alpha value of all pixels between the start and end position also change based on the linear interpolation of the alpha component between the two points.

As attraction points are removed during the river growth phase they are no longer rendered to the OpenGL window. Eventually all attraction points will have been removed leaving just the river skeleton. To signal the end of the river drawing and to move on to generating a height map, if the tributaries have not drawn to any attraction points over five draw calls or there are no more attraction points to grow towards then river growth is assumed to be completed. Attraction points that have not been reached by tributaries are removed from the 2D representation and a final drawing of the river skeleton will be shown. An example of the final

5.4. Creating terrain height map

Once the final river skeleton has been displayed the frame buffer is read to find each pixel in the terrain boundaries RGBA value. Based on the pixels RGB colour portion, each pixel will be grouped into either a river feature or a general terrain feature. For example, all pixels with an RGB competent value (0, 0, 1) corresponding to the colour blue will be set as river features and white pixels (1, 1, 1) will be set as general terrain.

For each pixel in the river skeleton a disk primitive encompasses a local area of terrain and the general terrain in the area has its height influenced by the river pixels height. Each pixel is checked to see what feature it belongs to so if a pixel within the disk is a river pixel then its height is not altered.

For each general terrain pixel, the height of the pixel is only adjusted if its position in the OpenGL window is within the terrain space. This is to stop the heights of terrain being altered if they lie outside the boundary which is considered sea level.

Each terrain pixel then has its height calculated which increases the further the distance it is from the river pixel. Its height weighting is defined so that the further away it is from a river pixel the lower the river pixels height affects the terrain pixel.

Disk primitives encompassing terrain will overlap so each terrain pixel that has had its height set previously will recalculate a new height and a new weight based on the new disk primitive it is in then blend with the its previous height and weight. This creates a smooth transition between the heights of every single terrain pixel.

Figure 4 illustrates the differences in height maps depending on the user set slope of the terrain around the river.

6. Evaluation

River space colonisation, height map generation and 3D terrain representation was implemented in Visual Studio using C++ and the OpenGL API.

All terrain generated in this project is done with a Nvidia 1080TI GPU, Intel 4790k CPU and 16GB of RAM. The Visual Studio performance profiler was used to times and

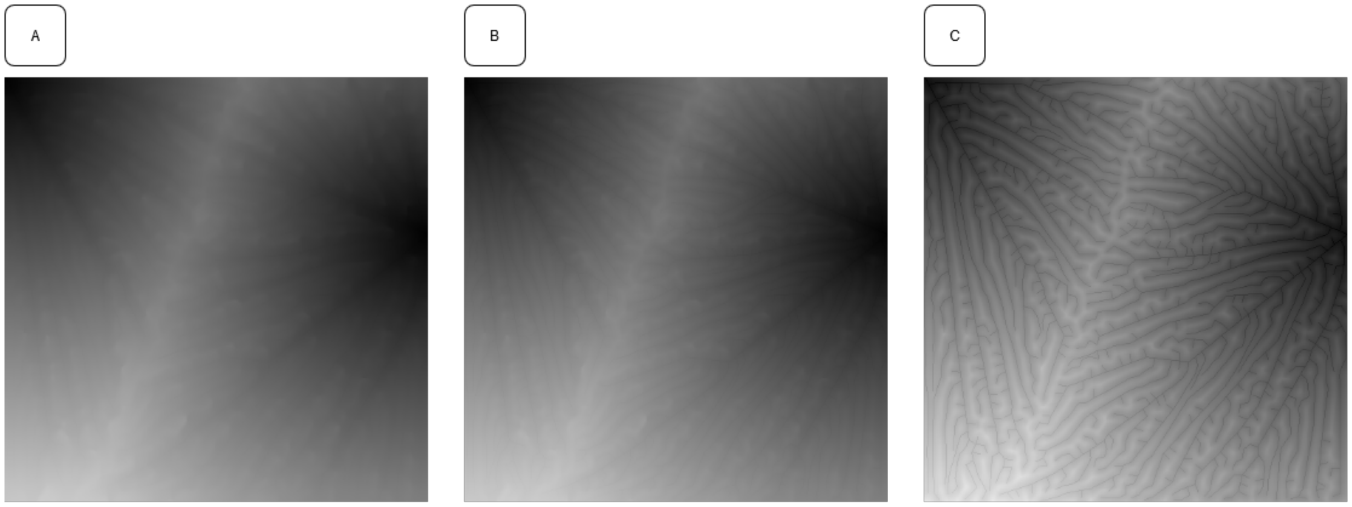

Figure 4. A defines a height map where the terrain grows at a 20 degree angle from the river. B shows a 40 degree angle. C illustrates a 80 degree slope.

memory use during terrain generation. Each generation phase must wait for previous phase to complete before execution. For example the time taken to generate 3D terrain is the time between generating the terrain height map and then generating terrain not the total generation time from the start of the program to the end of phase execution.

Table 1 illustrates the effect of how increasing terrain size, rivers created and attraction points generated has large impacts on performance.

While the amount of the attraction points is kept to a minimum for example 2,000 points, the increase in generation time for terrain sizes 512 x 512 pixels is 0.47 seconds when increasing the number of rivers created from one to four. On the other hand for a larger terrain generation of 2048 x 2048 we see a more noticeable increase in run time, for example execution time is 8.73 seconds when the number of rivers generated increases from one to four (6.47 - 15.20).

When the amount of attraction points is increased this creates the most significant impact on performance. For all three terrain sizes, increase the points by 25 times from 2,000 to 50,000 would be expected to increase the run time by 25 times. In reality the run time of the smallest terrain size with minimal attraction points compare to maximal points is 6.70 times longer whereas for the largest terrain the run time takes 41.88 times longer. As the number of rivers increases the difference between the time taken for the smallest and largest amount of attraction points at each terrain size only slightly increases due to the fact the amount of rivers has a small impact on performance compared to the number of attraction points.

The reason for performance being impacted more when increasing the number of attraction points compared to increase the number of rivers is due to how the river growth occurs. Each attraction point during the growth of the river skeleton checks every single river tributary to find the shortest valid path of growth. Even with a high density of river mouths in a terrain space, there will still be a small amount of rivers relative to how many attraction points can be used to better define the terrain and river shapes. An increase in the number of rivers doesn't guarantee a large increase in the number of river tributaries. However, the number of attraction points is more likely to generate a large number of tributaries as many will branch off the same point to points that require different directions of growth to reach. There is still a performance penalty for increasing the number of rivers which is especially noticeable when the number of attraction points also increases but the main loss in performance is due to attraction points.

Table 2 illustrates the large performance penalty for adding ridges to the terrain generation. Another reason for such a large increase in terrain height generation time is because every time an attraction point checks if a tributary has a valid path of growth it must also check that for every ridge line in the terrain boundary, does a new tributary line between the attraction point and the old tributary intersect. Every ridge line is also made up of several smaller lines to create a smooth Catmull-Rom spline. If the user decides to increase the amount of lines that make up the ridge spline, more line intersection checks will need to be completed.

Both Table 1 and Table 2 illustrate when terrain sizes increase the time taken to generate the height map and the 3D terrain slightly increases as more information has to be written and rendered. This would suggest most of the programs running time in these phases is spent on executing various overhead processes. Even when both the number of attraction points and rivers generated increases, there is only small fluctuating increase in time taken to perform the both generation tasks.

Table 3 illustrates memory usage during generation phases. The table displays the mean, median and standard deviation for generation with and without ridges. Memory use showed little or no significant change during each generation phase even when there was an increase in the number of attraction points, rivers or the with the addition of ridges.

When increasing the size of the terrain by a factor of 16 from 512 x 512 to 2048 x 2048 we see only a small

Terrain description	Terrain size	Generate terrain heights	Generate height map	Generate 3D terrain
1 river, 2,000 attraction points	512 x 512	1.40	0.67	1.94
	1024 x 1024	2.67	4.73	7.62
	2048 x 2048	6.47	34.52	30.36
1 river, 10,000 attraction points	512 x 512	2.54	0.73	1.89
	1024 x 1024	7.76	5.17	7.75
	2048 x 2048	34.60	35.80	30.61
1 river, 50,000 attraction points	512 x 512	9.37	0.77	1.91
	1024 x 1024	46.61	5.48	7.71
	2048 x 2048	270.94	37.51	30.83
2 rivers, 2,000 attraction points	512 x 512	1.57	0.70	1.95
	1024 x 1024	2.94	4.84	7.63
	2048 x 2048	8.12	34.39	30.48
2 rivers, 10,000 attraction points	512 x 512	3.44	0.72	1.88
	1024 x 1024	11.37	5.13	7.63
	2048 x 2048	54.10	35.99	30.37
2 rivers, 50,000 attraction points	512 x 512	21.84	0.74	1.90
	1024 x 1024	96.52	5.62	7.84
	2048 x 2048	618.76	44.26	35.83
4 rivers, 2,000 attraction points	512 x 512	1.87	0.70	1.78
	1024 x 1024	4.19	4.79	7.66
	2048 x 2048	15.20	34.78	30.49
4 rivers, 10,000 attraction points	512 x 512	6.08	0.72	1.94
	1024 x 1024	26.28	5.14	7.79
	2048 x 2048	111.25	35.62	30.40
4 rivers, 50,000 attraction points	512 x 512	59.86	0.74	1.90
	1024 x 1024	211.75	5.53	7.66
	2048 x 2048	1183.12	38.79	32.05

TABLE 1. Run time statistics without ridges during generation phase. Terrain description defines how many rivers and attraction points were generated in the terrain. Terrain size defines the pixel by pixel area in which terrain generation took place. Generate terrain heights, height map and 3D terrain define the time in seconds (s) it took to complete each generation step.

Terrain description	Terrain size	Generate terrain heights	Generate height map	Generate 3D terrain
1 river, 10,000 attraction points, 1 ridge	512 x 512	5.71	0.71	1.95
	1024 x 1024	27.25	5.19	7.75
	2048 x 2048	134.21	35.67	30.42
1 river, 10,000 attraction points, 2 ridges	512 x 512	8.99	0.71	1.93
	1024 x 1024	47.35	5.19	7.63
	2048 x 2048	232.75	35.91	30.45
1 river, 10,000 attraction points, 4 ridges	512 x 512	16.29	0.72	1.88
	1024 x 1024	90.39	5.19	7.21
	2048 x 2048	475.69	36.57	31.12
2 rivers, 10,000 attraction points, 1 ridge	512 x 512	11.74	0.72	1.92
	1024 x 1024	61.37	5.15	7.71
	2048 x 2048	295.96	35.64	30.33
2 rivers, 10,000 attraction points, 2 ridges	512 x 512	26.89	0.72	2.02
	1024 x 1024	146.98	5.30	7.83
	2048 x 2048	701.65	35.73	34.59
2 rivers, 10,000 attraction points, 4 ridges	512 x 512	51.10	0.74	1.95
	1024 x 1024	282.85	5.14	7.88
	2048 x 2048	1347.39	35.83	30.61
4 rivers, 10,000 attraction points, 1 ridge	512 x 512	25.06	0.72	1.89
	1024 x 1024	139.19	5.16	7.69
	2048 x 2048	681.29	35.41	29.93
4 rivers, 10,000 attraction points, 2 ridges	512 x 512	60.50	0.73	1.91
	1024 x 1024	335.41	5.07	7.68
	2048 x 2048	1645.39	38.16	33.76
4 rivers, 10,000 attraction points, 4 ridges	512 x 512	110.65	0.73	1.98
	1024 x 1024	619.23	5.12	7.89
	2048 x 2048	2950.83	38.64	32.78

TABLE 2. Run time statistics with ridges during generation phases. Each ridge is made up of 10 spline curves to make a more realistic smooth ridge line. For example two ridges would be comprised on 20 splines, 4 ridges contains 40 splines. Terrain generation times measured in seconds (s)

Memory usage measurements	Terrain size	Generate terrain heights	Generate height map	Generate 3D terrain
Mean	512 x 512	16.67	27.02	55.95
	1024 x 1024	17.57	54.21	160.77
	2048 x 2048	19.14	192.40	671.03
Standard deviation	512 x 512	2.17	2.17	2.16
	1024 x 1024	2.46	2.32	2.32
	2048 x 2048	2.86	2.88	2.97

TABLE 3. MEMORY USAGE DURING GENERATION PHASES. MEMORY WHILE ALTERING THE NUMBER OF RIVERS, ATTRACTIONS POINTS AND RIDGES HAD A NEGLIGIBLE IMPACT ON EACH PHASES MEMORY USAGE SO THE MEAN AND STANDARD DEVIATION WAS FOUND. MEMORY USE MEASURED IN MEGABYTES (MB).

increase in the mean and median memory usage by 2.17MB and 2.43MB respectively which is only a 14 - 15% increase in memory usage.

Memory usage during height map generation is due to the overhead of writing each of the terrain pixel height values to the height map. As the size of the terrain increases we see a large jump in memory usage as terrain size increases.

Generating the 3D terrain representation of the height map is the most memory intensive task. As terrain size grows from 512 x 512 to 1024 x 1024 there is a 2.87 times increase in mean memory usage and between 512 x 512 and 2048 x 2048 there is a 12.00 times increase in mean memory usage. We expect to see a 4 times increase in memory usage when generating a terrain of size 512 x 512 compared to a terrain 1024 x 1024 with another 4 times increase usage when compared to 2048 x 2048. While memory usage is not as expected we do see a large jump in usage relative to an increase in terrain size so some of the memory costs can be attributed to this. It is likely though that large overhead introduced during the rendering stages is the reason for unexpected 3D generation memory usage.

Standard deviations over all three generation stages grow as the size of the terrain being generated increases however between generation phases they are relatively similar.

7. Discussion

Creating watershed terrain that has landscape features matching real life terrain is difficult. Many user parameters affect the final shape of the terrain and several random elements such as attraction point generation, ridge position and river mouth locations make emulating real height maps a tedious task. This project does however create terrain based around realistic river skeletons producing similar terrain height to real life counterparts.

A height map representation of a naturally formed watershed in New Zealand is shown in figures 5 and figure 7 as image A. A similar river skeleton shape is construction and illustrated shown in image B. C shows the terrain height maps generated from the river skeletons. Both height maps produce similar terrain heights as their real life counterparts. Most notably the height of the river and terrain local to the river show a large decrease in height in both the real height maps and generated ones. The smooth increase in the colour gradient of surrounding terrain is also represented in both height maps showing the gradual increase in terrain height as the river networks increase their distance from their river mouths. Noticeable terrain boundaries also occur in both real height maps and generated height maps, figure 5 A shows a subtle divide between the top and bottom river also apparent on the generated height map figure 5 C.

Terrain heights set in river primitives are generated by using a single user defined value for the slope. Terrain slopes are therefore uniformly used to generate terrain heights around every point in the river; this ignores the fact that the slope of a river at a high terrain height will have steep banks while also resulting in steep increases in surrounding terrain height the further you move away from the river. This can be addressed by having the height of a river affect the cross sectional area of the river, a smaller cross section would correspond to an increase in the slope of the terrain generated in the surrounding disk primitive [18].

While trees share similar growth structures to rivers, the impact of river erosion processes is not relevant to tree grow and so is absent space colonisation algorithm. Figure 5 image A illustrates meandering river sections which are not present in the procedurally generated river in image C. While the heights of terrain surrounding rivers appears to emulate natural terrain heights, the lack of meandering causes river tributaries to appear unnaturally straight.

Different sections of rivers have different widths with wider parts of the river having large amounts of small tributary growth. The current generation implementation does not adjust for changes in river width, such as a very wide river section near the river mouth or wide main channels that have many smaller tributaries extruding from them. As the length of a river channel grows it appears to increase in width and have many tributaries connecting with it. Figure 5 image B has long tributaries extending off the initial river channel with very little small tributary growth along its path.

The watershed is generated within a square terrain boundary. This is slightly altered once the river mouths are found as terrain will only generate within the terrain whose shape intersects with all river mouths. The issue is

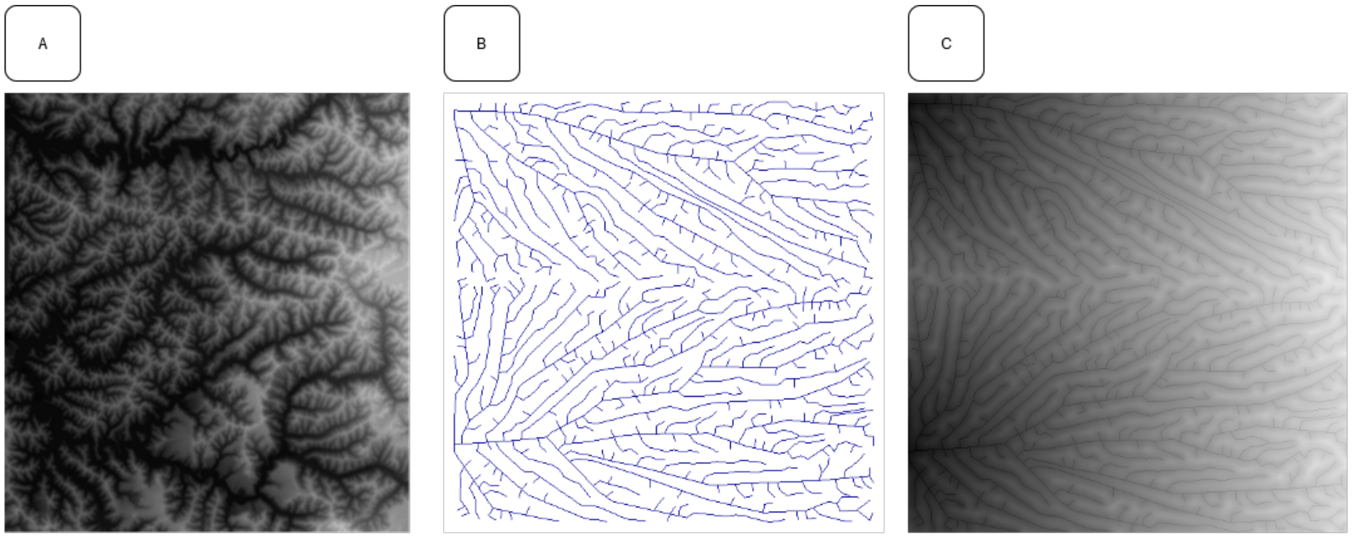

Figure 5. First evaluation image contrasting a real life terrain height map to a generated height map. A shows a height map of a New Zealand watershed, geographical location = -38.8210/534.6974. B illustrates a generated river skeleton with a similar shape to the one in the real life river height map, random generation seed = 2345678902346789. C shows the generated height map whose heights were based on the generated river skeleton.

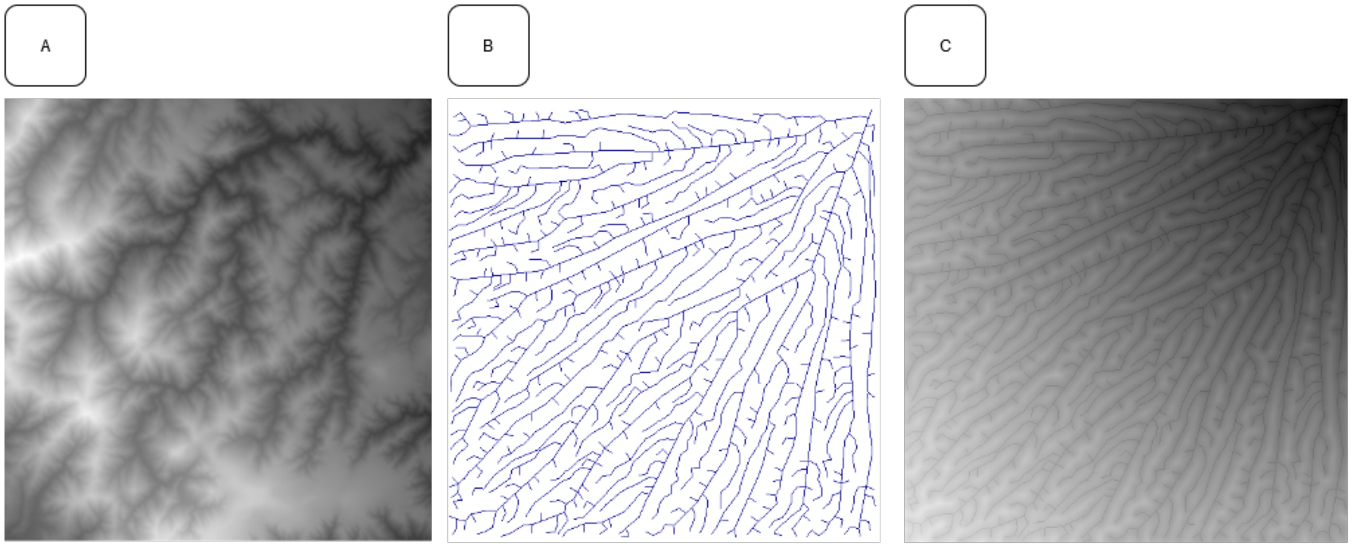

Figure 6. Second evaluation image contrasting a real life terrain height map to a generated height map. A Geographical location = -45.0953/170.7398. B Random generation seed = 4830534897258945.

attraction point generation is restricted within the boundaries of a square and as the number of points increase the terrain space is made to consistently resemble a square shape. To fix this issue, the shape of the initial terrain should be randomly assigned to create more realistic boundaries as shown as the subtle white terrain borders in figure 7 image A.

8. Future work

The implementation of the ability to have varying river slopes based on the height of a river section and varying river widths would improve the realism of the generated 3D terrain. As a river increase in height the slope of the river rapidly increases in real life resulting in much steeper and high surrounding terrains. Also river width varies in size through different sections of a river impacting how local terrain in impacted by river erosion processes for example, build up of sediment and fast water in tight tributaries. As shown in real life height map example, this is missing from this projects implementation but is possible to add as an initial improvement to improve the realism of generated terrain.

Many erosion processes occur over long periods of time that play a large role in the shaping terrain. Rivers slowly meander and cut away at the terrain lying on river banks while storms reshape terrain by moving loose rock and soil

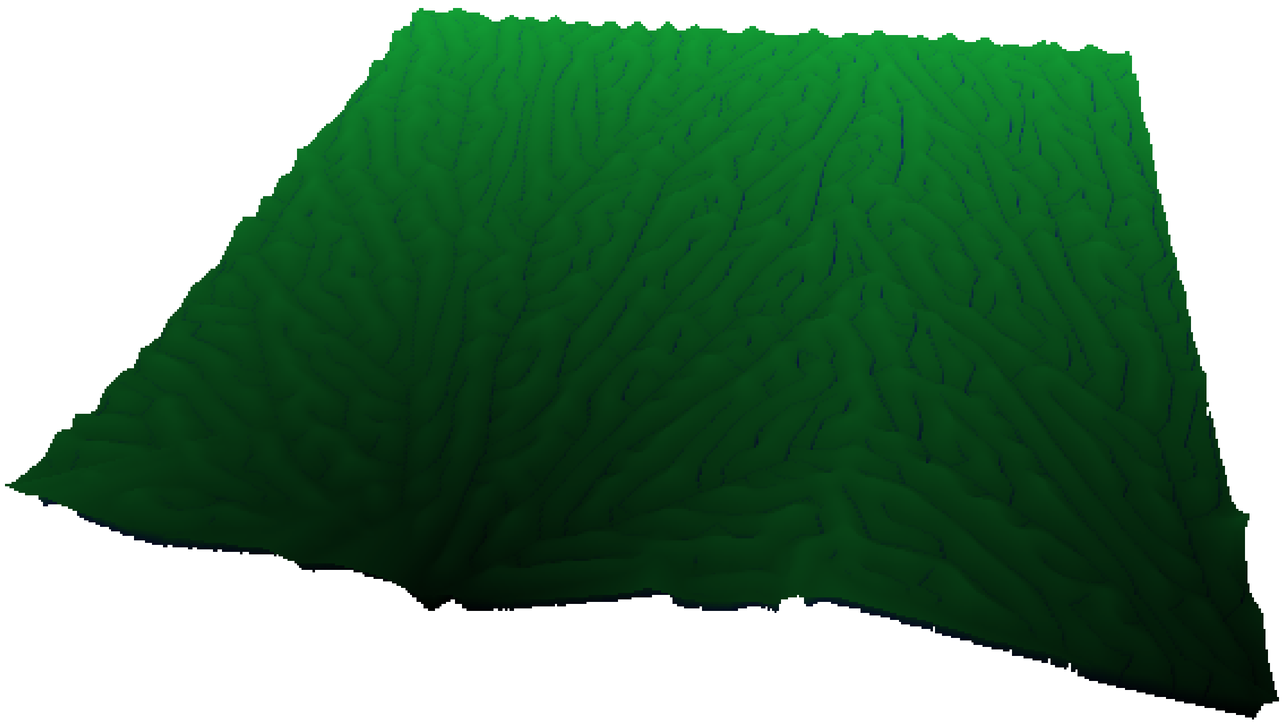

Figure 7. 3D generation of the terrain from Figure 5 to represent possible output. Rivers have been inverted.

around the landscape. This project only finds the height of terrain based on the location of river networks ignoring other erosion factors. Also terrain height is solely based on the distance terrain lies from river networks so there is no adjustment for the natural effect of river meandering which cuts away river banks dependent on the path and speed of a rivers water flow. The speed of water flow could be implemented so that the height of the river influences the speed of water flow; this impacts the erosion processes on surrounding landscape causing smaller terrain heights in areas with greater river height.

Creating a river skeleton in 2D and rendering it with OpenGL meant the frame buffer could be read to get pixel accurate locations of the river skeleton and ridge features. Each river tributary was created by drawing a 2D line between a start and end point, this meant the pixel locations between the start and end points of a tributary were unknown. The ability to read the frame buffer gave individual pixel locations of the river skeleton speeding up the development process and reduced the amount of extra algorithmic work. The issue is that terrain generation is limited by the size of OpenGL window the river skeleton is drawn in. The size of the OpenGL window can't be larger than the size of the monitor the river skeleton is being rendered on as parts of the river will render off screen and will not display in the frame buffer. The locations of the river pixels are important as each pixel has an affect on the surrounding terrain heights. However, it is possible to mathematically calculate exact river pixel locations between the start and end points of the tributaries which will remove the need for OpenGL frame buffers and allow terrains to be generated that are larger than the size of the users computer monitor.

9. Conclusion

This paper introduces a novel approach to generating a realistic watershed by deriving terrain heights based on the height of rivers. The user is given a high degree of control over the size and shape of terrain allowing for unique generation results. River space colonisation shows promising results as a terrain height map can be produced while avoiding the use of erosion simulations by interpolating the heights of the surrounding landscape to produce a terrain height map. The space colonisation algorithm generates rivers that have a similar structure to naturally occurring ones but they lack subtle terrain features unique to rivers. The addition of varying river slopes and widths would likely improve the realism of generated terrain. This is because there are many river erosion process that impact the shape of surrounding terrain so basing terrain height solely on the heights of a river ignores many of these processes resulting in sub optimal terrain realism.

10. Acknowledgments

This research project was approved by Dr Alex Shaw and Dr Sathiamoorthy Manoharan.

11. Appendix

GitHub repository for the implementation of Realistic Terrain Generation Based on River Space Colonisation: https://github.com/RSpe/River-Space-Colonization

References

[1] Farès Belhadj and Pierre Audibert. Modeling landscapes with ridges and rivers: bottom up approach. In *Proceedings of the 3rd international conference on Computer graphics and interactive techniques in Australasia and South East Asia*, pages 447–450, 2005.

[2] N Chiba. Terrain simulation based on the recursive refinement of ridge-lines. In *Proc. of CAD/Graphics' 91*, 1991.

[3] Norishige Chiba, Kazunobu Muraoka, and Kunihiko Fujita. An erosion model based on velocity fields for the visual simulation of mountain scenery. *The Journal of Visualization and Computer Animation*, 9(4):185–194, 1998.

[4] Jonathon Doran and Ian Parberry. Controlled procedural terrain generation using software agents. *IEEE Transactions on Computational Intelligence and AI in Games*, 2(2):111–119, 2010.

[5] David S Ebert, F Kenton Musgrave, Darwyn Peachey, Ken Perlin, and Steven Worley. *Texturing & modeling: a procedural approach*. Morgan Kaufmann, 2003.

[6] Alain Fournier, Don Fussell, and Loren Carpenter. Computer rendering of stochastic models. *Communications of the ACM*, 25(6):371–384, 1982.

[7] Jean-David Génevaux, Éric Galin, Eric Guérin, Adrien Peytavie, and Bedrich Benes. Terrain generation using procedural models based on hydrology. *ACM Transactions on Graphics (TOG)*, 32(4):1–13, 2013.

[8] Jean-David Génevaux, Eric Galin, Adrien Peytavie, Eric Guérin, Cyril Briquet, François Grosbellet, and Bedrich Benes. Terrain modelling from feature primitives. In *Computer Graphics Forum*, volume 34, pages 198–210. Wiley Online Library, 2015.

[9] Alex D Kelley, Michael C Malin, and Gregory M Nielson. Terrain simulation using a model of stream erosion. In *Proceedings of the 15th annual conference on Computer graphics and interactive techniques*, pages 263–268, 1988.

[10] Tsz-Yam Lau and W Randolph Franklin. River network completion without height samples using geometry-based induced terrain. *Cartography and Geographic Information Science*, 40(4):316–325, 2013.

[11] Huailiang Li, Xianguo Tuo, Yao Liu, and Xin Jiang. A parallel algorithm using perlin noise superposition method for terrain generation based on cuda architecture. In *International Conference on Materials Engineering and Information Technology Applications (MEITA 2015)*. Atlantis Press, 2015.

[12] Jacob Olsen. Realtime procedural terrain generation. 2004.

[13] Owen Patrick, Manjeet Rege, and Reynold Bailey. Extending space colonization tree modeling for artistic control and environmental interactions. In *2014 International Conference on Computer Graphics Theory and Applications (GRAPP)*, pages 1–8. IEEE, 2014.

[14] Ken Perlin. An image synthesizer. *ACM Siggraph Computer Graphics*, 19(3):287–296, 1985.

[15] Adrien Peytavie, Thibault Dupont, Eric Guérin, Yann Cortial, Bedrich Benes, James Gain, and Eric Galin. Procedural riverscapes. In *Computer Graphics Forum*, volume 38, pages 35–46. Wiley Online Library, 2019.

[16] Przemyslaw Prusinkiewicz and Mark Hammel. A fractal model of mountains and rivers. In *Graphics Interface*, volume 93, pages 174–180. Canadian Information Processing Society, 1993.

[17] Thomas J Rose and Anastasios G Bakaoukas. Algorithms and approaches for procedural terrain generation-a brief review of current techniques. In *2016 8th International Conference on Games and Virtual Worlds for Serious Applications (VS-GAMES)*, pages 1–2. IEEE, 2016.

[18] David L Rosgen. A classification of natural rivers. *Catena*, 22(3):169–199, 1994.

[19] Pascale Roudier, Bernard Peroche, and M Perrin. Landscapes synthesis achieved through erosion and deposition process simulation. In *Computer Graphics Forum*, volume 12, pages 375–383. Wiley Online Library, 1993.

[20] Adam Runions, Brendan Lane, and Przemyslaw Prusinkiewicz. Modeling trees with a space colonization algorithm. *NPH*, 7:63–70, 2007.

[21] Soon Tee Teoh. River and coastal action in automatic terrain generation. In *CGVR*, pages 3–9. Citeseer, 2008.

[22] Soon Tee Teoh. Riverland: An efficient procedural modeling system for creating realistic-looking terrains. In *International Symposium on Visual Computing*, pages 468–479. Springer, 2009.

[23] Brian Wyvill, Andrew Guy, and Eric Galin. Extending the csg tree. warping, blending and boolean operations in an implicit surface modeling system. In *Computer Graphics Forum*, volume 18, pages 149–158. Wiley Online Library, 1999.