FRASER BARKER

1600196

1600196@uad.ac.uk

ASSESSMENT REPORT

CMP305

Contents

[Features 2](#_Toc6946867)

[Smoothing 2](#_Toc6946868)

[Faulting 2](#_Toc6946869)

[Circle Algorithm 2](#_Toc6946870)

[Midpoint Displacement 2](#_Toc6946871)

[Simplex Noise 2](#_Toc6946872)

[Fractal Brownian Motion 2](#_Toc6946873)

[Voronoi Regions 2](#_Toc6946874)

[Terrain Picking 2](#_Toc6946875)

[L-Systems 2](#_Toc6946876)

[Organisation 3](#_Toc6946877)

[Terrain 3](#_Toc6946878)

[L-System 3](#_Toc6946879)

[Sizeable Quad 3](#_Toc6946880)

[Simplex Noise 3](#_Toc6946881)

[Terrain Shader 3](#_Toc6946882)

[Manipulation Shader 3](#_Toc6946883)

[Critical Appraisal 4](#_Toc6946884)

[Terrain Class 4](#_Toc6946885)

[L-System 4](#_Toc6946886)

[Tessellation 4](#_Toc6946887)

[Overall 4](#_Toc6946888)

[Changes 4](#_Toc6946889)

[Reflection 5](#_Toc6946890)

[Topics 5](#_Toc6946891)

[Further/Alternative Applications 5](#_Toc6946892)

[References 6](#_Toc6946893)

# Features

1. An outline of the general features of your program
2. A description of features such as visual and sound effects or interactivity.

## Controls

## Smoothing

## Faulting

## Circle Algorithm

## Midpoint Displacement

## Simplex Noise

## Fractal Brownian Motion

## Voronoi Regions

## Terrain Picking

## L-Systems

# Organisation

1. A description of your code organization: classes, data structures, separate code files etc. Include diagrams as required.

## Terrain

## L-System

## Sizeable Quad

## Simplex Noise

## Terrain Shader

## Manipulation Shader

# Critical Appraisal

1. A critical appraisal of your program in relation to code organization, efficiency of strategies and the decisions regarding alternative strategies and a critical evaluation of the success of your procedural content.

## Terrain Class

Non-utilisation of the GPU for semi-expensive calculations.

## L-System

More central integration with the terrain.

## Tessellation

Should have been utilised for both terrain and l-system.

## Overall

Giant success!

## Changes

Better texturing, lighting, use of tessellation.

# Reflection

1. Reflection on what has been learned, technically in terms of programming and also about procedural methods from undertaking the work, concentrating on being informative to someone else who might take on a similar task.

## Topics

## Further/Alternative Applications

# References

Nic. 2013. Simple terrain smoothing. [ONLINE] Available at:

<http://nic-gamedev.blogspot.com/2013/02/simple-terrain-smoothing.html> [Accessed 23/04/2019]

Antonio Ramires Fernandes. 2019. The Fault Algorithm. [ONLINE] Available at:

<http://www.lighthouse3d.com/opengl/terrain/index.php?fault> [Accessed 23/04/2019]

Steve Losh. 2016. Terrain Generation with Midpoint Displacement. [ONLINE] Available at:

<http://stevelosh.com/blog/2016/02/midpoint-displacement/> [Accessed 23/04/2019]

Stefan Gustavson. 2005. Simplex noise demystified. [ONLINE] Available at:

<http://staffwww.itn.liu.se/~stegu/simplexnoise/simplexnoise.pdf> [Accessed 23/04/2019]

Patricio Gonzalez Vivo & Jen Lowe. 2015. Fractal Brownian Motion. [ONLINE] Available at:

<https://thebookofshaders.com/13/> [Accessed 23/04/2019]

Glare Technologies. 2019. Advanced Fractional Brownian Motion Noise. [ONLINE] Available at:

<https://www.indigorenderer.com/indigo-technical-reference/indigo-shader-language-reference/built-functions-%E2%80%93-procedural-noise-fun-0> [Accessed 23/04/2019]

Matej Zabsky. 2016. Procedural Terrain with Ridged Fractal Noise. [ONLINE] Available at: <https://stackoverflow.com/questions/36796829/procedural-terrain-with-ridged-fractal-noise> [Accessed 23/04/2019]

Red Blob Games. 2016. Making maps with noise functions. [ONLINE] Available at:

<https://www.redblobgames.com/maps/terrain-from-noise/> [Accessed 23/04/2019]

Paul Bourke. 1989. Efficient Triangulation Algorithm Suitable for Terrain Modelling. [ONLINE] Available at: <http://paulbourke.net/papers/triangulate/> [Accessed 23/04/2019]

Iedoc. 2015. Picking. [ONLINE] Available at: <https://www.braynzarsoft.net/viewtutorial/q16390-24-picking> [Accessed 23/04/2019]

Przemyslaw Prusinkiewicz & James Hanan. 2016. Lindenmayer Systems, Fractals, and Plants. [ONLINE] Available at: <http://algorithmicbotany.org/papers/lsfp.pdf> [Accessed 23/04/2019]

Richard Hawkes. 2016. Unity Algorithms – The Lindenmayer Series (L Series). [ONLINE] Available at: <https://www.youtube.com/watch?v=uBEA6VSUybk> [Accessed 23/04/2019]

Sher Minn Chong. 2018. L-Systems Renderer. [ONLINE] Available at: <http://piratefsh.github.io/p5js-art/public/lsystems/> [Accessed 23/04/2019]

Rastertek. Unknown. Slope Based Texturing. [ONLINE] Available at: <http://www.rastertek.com/tertut14.html> [Accessed 23/04/2019]

Xing Mei, Philippe Decaudin & Bao-Gang Hu. 2007. Fast Hydraulic Erosion Simulation and Visualization on GPU. [ONLINE] Available at: <http://www-evasion.imag.fr/Publications/2007/MDH07/FastErosion_PG07.pdf> [Accessed 23/04/2019]

Balazs Jako. 2011. Fast Hydraulic and Thermal Erosion on the GPU. [ONLINE] Available at: <http://old.cescg.org/CESCG-2011/papers/TUBudapest-Jako-Balazs.pdf> [Accessed 23/04/2019]

Florian Bosch. 2011. WebGL Landscaping and Erosion. [ONLINE] Available at: <http://codeflow.org/webgl/craftscape/> [Accessed 23/04/2019]