Copyright is owned by the Author of the thesis. Permission is given for a copy to be downloaded by an individual for the purpose of research and private study only. The thesis may not be reproduced elsewhere without the permission of the Author.

Novel Technologies for the Manipulation of Meshes on the CPU and GPU

A thesis presented in partial fulfilment of the requirements for the degree of

Masters of Science
in
Computer Science

at Massey University, Palmerston North,
New Zealand.

Richard John Rountree

2007

Abstract

This thesis relates to research and development in the field of 3D mesh data for computer graphics. A review of existing storage and manipulation techniques for mesh data is given followed by a framework for mesh editing. The proposed framework combines complex mesh editing techniques, automatic level of detail generation and mesh compression for storage. These methods work coherently due to the underlying data structure. The problem of storing and manipulating data for 3D models is a highly researched field. Models are usually represented by sparse mesh data which consists of vertex position information, the connectivity information to generate faces from those vertices, surface normal data and texture coordinate information. This sparse data is sent to the graphics hardware for rendering but must be manipulated on the CPU.

The proposed framework is based upon geometry images and is designed to store and manipulate the mesh data entirely on the graphics hardware. By utilizing the highly parallel nature of current graphics hardware and new hardware features, new levels of interactivity with large meshes can be gained. Automatic level of detail rendering can be used to allow models upwards of 2 million polygons to be manipulated in real time while viewing a lower level of detail. Through the use of pixels shaders the high detail is preserved in the surface normals while geometric detail is reduced. A compression scheme is then introduced which utilizes the regular structure of the geometry image to compress the floating point data. A number of existing compression schemes are compared as well as custom bit packing.

This is a TIF funded project which is partnered with Unlimited Realities, a Palmerston North software development company. The project was to design a system to create, manipulate and store 3D meshes in a compressed and easy to manipulate manner. The goal is to create the underlying technologies to allow for a 3D modelling system to become integrated into the Umajin engine, not to create a user interface/stand alone modelling program. The Umajin engine is a 3D engine created by Unlimited Realities which has a strong focus on multimedia. More information on the Umajin engine can be found at www.umajin.com.

In this project we propose a method which gives the user the ability to model with the high level of detail found in packages aimed at creating offline renders but create models which are designed for real time rendering.

Acknowledgements

I would like to thank Dr Ramesh Rayudu, my supervisor for this project for keeping things on track and providing support.

David Brebner for his knowledge of all things 3D, compression and for creating the project in the first place.

Robert Grapes for his patience and consistent help with solving bugs.

Russell Brebner for his management and financial organisation.

I would like to thank my family and friends for their support throughout the project.

Finally I would like to thank The Foundation for Research, Science and Technology (FRST) for funding this Technology for Industry Fellowships (TIF) project.

Table of Contents

Chapter 1 – Background 1

1.1 Scope and Objectives 5

1.2 Significance 6

1.3 Original Contributions 7

1.4 Thesis Organization 8

Chapter 2 – Representation and Literature Review 9

2.1 Voxels 9

2.2 Octree 11

2.3 Adaptively Sampled Distance Fields 12

2.4 Normal Meshes 13

2.5 Polygonal Meshes 14

2.6 Geometry Images 15

2.7 Subdivision 18

2.8 Constructive Solid Geometry 18

2.9 Existing modelling packages 19

2.10 Editing Techniques 20

2.11 Compression 21

2.12 Level of Detail 22

2.13 Summary 25

Chapter 3 - A Mesh Editing Framework for the GPU using Geometry Images 26

3.1 User Interaction 27

3.2 Geometry Image 28

3.3 Rendering 29

3.4 Editing 32

3.5 Improved Detail Version 34

3.6 Editing techniques 36

3.7 Seaming 39

3.8 Smooth 41

3.9 Push/Pull 42

3.10 Splatting 43

3.11 Dimple/CSG 45

3.12 Summary 47

Chapter 4 - Polygon Count Reduction for Automatic Level of Detail 48

4.1 Mesh Based 48

4.2 Geometry Shader 52

4.3 Geometry Image Based 54

4.4 Final Method 57

4.5 Summary 57

Chapter 5 - Compression of Geometry Images for storage 58

5.1 Existing file formats 58

5.2 NVIDIA and ATI compression tools 62

5.3 DCT/ floating point jpeg 62

5.4 Bezier curve compression 65

5.5 Custom byte packing 66

5.6 Summary 71

Chapter 6 – Results 72

6.1 Visual Results 72

6.2 Numeric Results 75

6.3 Other Implementations 76

6.4 Converting arbitrary meshes to Geometry Images 78

Chapter 7 - Conclusions and Future Reference 79

7.1 Limitations of Geometry Images 80

7.2 Future Directions 82

References 85

Bibliography 89

Table of Figures

Figure 1: Shader Model 3 Pipeline 2

Figure 2: Shader Model 4.0 pipeline 3

Figure 3: Vertex Adjacency Information 3

Figure 4: Common Shader Core – Courtesy DirectX SDK 5

Figure 5: 2562 Voxel Data set 9

Figure 6: Quadtree Diagram 11

Figure 7: Different LOD on a Normal Mesh 13

Figure 8: Polygon mesh data structure 14

Figure 9: Geometry Image and associated mesh 16

Figure 10: CSG Example courtesy Wikipedia 19

Figure 11: Comparing Different Levels of Detail 23

Figure 12: Polygon Reduction 23

Figure 13: Square based Polygon Reduction 24

Figure 14: Vertex Collapse 24

Figure 15: Framework Overview 27

Figure 16: Comparison of Special case % 28

Figure 17: Mesh Topology 29

Figure 18: 8 bit vs. 32 bit 29

Figure 19: Overview of editing 33

Figure 20: Overview of editing optimized framework 35

Figure 21: Comparison of geometry image sizes 36

Figure 22: Showing how the seams join 39

Figure 23: Wrapping 40

Figure 24: Multi attribute splatting 45

Figure 25: Different resolution for CSG operation 46

Figure 26: Square based reduction 50

Figure 27: Mesh before Vertex Collapse 51

Figure 28: Mesh after Vertex Collapse 51

Figure 29: Polygon Culling Suitability 52

Figure 30: Overlap and Tear 53

Figure 31: Geometry Shader Decimation 54
Figure 32: Seam carving approach 55
Figure 33: Comparison of single pixel removal 56
Figure 34: Effect of 16bit compression on Normals 60
Figure 35: Comparison of HDR image format 60
Figure 36: Lossless Photo HD compression 61
Figure 37: Lossy Photo HD compression 61
Figure 38: ATI compressinator results 62
Figure 39: Floating point JPEG compression 64
Figure 40: Bezier Curve Compression 65
Figure 41: Bezier Curve against Original Data 66
Figure 42: Custom mip map based storage format 67
Figure 43: Comparison of mip map resolution with file size 67
Figure 44: Compression on ball of bricks 68
Figure 45: Close up of lossy Compression 69
Figure 46: Compression on Smooth Surface 70
Figure 47: Worst case lighting for different surface 71
Figure 48: Before and After using Pull/Push brush 73
Figure 49: Face manipulation 73
Figure 50: High detail splatting 74
Figure 51: Automatic LOD Comparison 74
Figure 52: Rendering and Editing Frame rates 76
Figure 52: Torus geometry image mapping 80
Figure 53: Effect of Stretch on Diffuse Map 81
Figure 54: Shows Effect of Large Stretch 81

Table of Code Listings

Code Listing 1: Vertex & Geometry Shader to stream out Geometry Image data 30
Code Listing 2: Code to generate an object space normal from a Geometry Image...................31
Code Listing 3: Function to calculate normal based on texture coordinates 34
Code Listing 4: Structure of Brush information 37
Code Listing 5: Texture Coordinate based brush size calculation..37
Code Listing 6: Volume based brush size calculation 38
Code Listing 7: Volume and surface angle based brush size calculation 38
Code Listing 8: Soft Brush linear interpolation 39
Code Listing 9: Custom Texture wrapping function..41
Code Listing 10: Surface Smoothing Brush 42
Code Listing 11: Normal Based Shifting Brush..43
Code Listing 12: Projective splatting Brush...44
Code Listing 13: CSG Brush 46
Code Listing 14: Mesh Structure 49