Contents

Preface **ix**

1 Modeling in Game Production **1**
1.1 Game Production Pipeline 1
1.2 Polygon Mesh 5
1.2.1 Polygon Mesh Creation* 7
1.2.2 Polygon Mesh Representation 11
1.2.3 Surface Normal 15
1.3 Model Export and Import 17
1.4 Coordinate Systems 19

2 Vertex Processing **23**
2.1 World Transform 24
2.1.1 Affine Transforms and Homogeneous Coordinates . . . 25
2.1.2 World Matrix 27
2.1.3 Euler Transform 31
2.1.4 Transform of Surface Normals 32
2.2 View Transform 34
2.2.1 Camera Space 35
2.2.2 Space Change and View Matrix 36
2.3 Per-vertex Lighting 41
2.4 Projection Transform 41
2.4.1 View Frustum 42
2.4.2 Projection Matrix 43
2.4.3 Derivation of Projection Matrix* 48

3 Rasterization **53**
3.1 Clipping 53
3.2 Perspective Division 54
3.3 Back-face Culling 56
3.4 Coordinate Systems - Revisited* 59
3.4.1 3ds Max to OpenGL - Axis Flipping 60
3.4.2 OpenGL to Direct3D - Reflection 60
3.4.3 OpenGL to Direct3D - Vertex Reordering 64
3.5 Viewport Transform 66
3.6 Scan Conversion 70
3.7 Application: Object Picking 75

3.7.1 Computing World-space Ray 76
3.7.2 Ray-object Intersection Test 81

4 Fragment Processing and Output Merging 89
4.1 Texturing 89
4.1.1 Texture Coordinates 89
4.1.2 Surface Parameterization 92
4.1.3 Texture Coordinates to Texel Address 94
4.2 Output Merging 95
4.2.1 Z-buffering 95
4.2.2 Alpha Blending 97
4.3 Z-culling 99
4.3.1 Tile-based Culling 99
4.3.2 Application: Pre-Z Pass 101

5 Illumination and Shaders 105
5.1 Phong Lighting Model 105
5.1.1 Diffuse Reflection 107
5.1.2 Specular Reflection 108
5.1.3 Ambient Reflection 111
5.1.4 Emissive Light 111
5.2 Shaders and Shading Languages 112
5.2.1 Vertex and Fragment Shaders 112
5.2.2 High-Level Shading Language* 113
5.3 Lighting in the Pipeline 115
5.3.1 Per-vertex Lighting in HLSL* 115
5.3.2 Per-vertex Lighting vs. Per-fragment Lighting 117
5.3.3 Per-fragment Lighting in HLSL* 120
5.4 Global Illumination 121
5.4.1 Ray Tracing 122
5.4.2 Radiosity 125

6 Parametric Curves and Surfaces 131
6.1 Parametric Curves 131
6.1.1 Bézier Curves 132
6.1.2 Hermite Curve and Catmull-Rom Spline 136
6.2 Application: Camera Path 138
6.3 Bézier Surfaces 141
6.3.1 Bilinear Patch 141
6.3.2 Biquadratic Bézier Patch 144
6.3.3 Bicubic Bézier Patch 148
6.3.4 Bézier Triangle 151

7 Shader Models **157**
7.1 Shader Model 4 and Geometry Shader 157
7.2 Application: Dynamic Particle System 159
7.2.1 Physics-based Simulation of Particles 159
7.2.2 Fireworks Simulation 162
7.2.3 Fireworks Rendering 164
7.3 Shader Model 5 and Tessellation 166
7.4 Application: PN-triangles 167
7.4.1 Computing Control Points 168
7.4.2 Computing Control Normals 171
7.4.3 PN-triangle Tessellation 174

8 Image Texturing **177**
8.1 Texture Addressing Mode 177
8.2 Texture Filtering 180
8.2.1 Magnification 180
8.2.2 Minification 182
8.3 Mipmapping 183
8.3.1 Mipmap Construction 184
8.3.2 Mipmap Filtering 184
8.3.3 Options for Mipmap Filtering 187
8.4 Anisotropic Filtering 190

9 Bump Mapping **197**
9.1 Height Field 198
9.2 Normal Mapping 199
9.2.1 Normal Map 200
9.2.2 Algorithm for Normal Mapping 202
9.3 Tangent-space Normal Mapping 206
9.3.1 Algorithm for Tangent-space Normal Mapping 206
9.3.2 Tangent Space Computation* 210
9.4 Authoring of Normal Map 212
9.5 Parallax Mapping 215
9.6 Displacement Mapping 219

10 Advanced Texturing **225**
10.1 Environment Mapping 225
10.1.1 Cube Mapping 226
10.1.2 Cube Map Access* 227
10.1.3 Dynamic Cube Mapping 229
10.2 Light Mapping 230
10.2.1 Diffuse Light Mapping 231
10.2.2 Radiosity Normal Mapping* 232
10.3 Shadow Mapping 236
10.3.1 Algorithm for Shadow Mapping 237

10.3.2 Shader Codes for Shadow Mapping* 242
10.3.3 Shadow Map Filtering 245
10.4 Ambient Occlusion 248
10.5 Deferred Shading 253

11 Character Animation 257
11.1 Keyframe Animation 257
11.2 Rotation 260
11.2.1 Interpolation of Euler Angles 261
11.2.2 Quaternion Representation 262
11.2.3 Rotation Using Quaternion 263
11.2.4 Interpolation of Quaternions 267
11.3 Hierarchical Modeling and Space Change 269
11.3.1 Hierarchical Model 270
11.3.2 Space Change between Bones 272
11.3.3 World Space to Bone Space 274
11.4 Forward Kinematics 278
11.5 Skinning and Keyframe Animation 280
11.5.1 Skinning 280
11.5.2 Skinning in Keyframe Animation 283
11.6 Inverse Kinematics 286
11.6.1 Analytic Solution 286
11.6.2 Cyclic Coordinate Descent 288

12 Physics-based Simulation* 293
12.1 Penalty Method 293
12.2 Impulse Method 296
12.2.1 Impulse 296
12.2.2 Impulse-based Collision Resolution 297
12.3 Collision Detection 301
12.3.1 Bounding Volumes and Their Hierarchy 303
12.3.2 Triangle-triangle Intersection 306

References 311

Preface

Many of computer graphics classes in colleges are focused on real-time rendering and animation. However, it is not easy to find an appropriate textbook, which presents the state of the art in interactive graphics, is balanced between theory and practicality, and is of a proper length to be covered in a semester. This book is written for answering the need and presents the *must-know* in interactive graphics. This book fits the advanced undergraduate or beginning graduate classes for 'Computer Graphics' and 'Game Programming.'

Another primary reader group of this book may be composed of game developers, who have experience in graphics APIs and shader programming but have felt lack of theoretical background in 3D graphics. A lot of programming manual-like books can be found in the bookstore, but they do not provide a sufficient level of mathematical background for the game developers. Assuming that the readers have minimal understanding of vectors and matrices, this book provides an opportunity to combine their experiences with the background theory of computer graphics.

At the core of contemporary interactive graphics is the evolution of GPU. The content of this book is organized around GPU programming. The GPU is partitioned into programmable stages and hard-wired stages. This book presents a variety of algorithms for the programmable stages, and the indispensable knowledge required to configure the hard-wired stages.

The organization and presentation of this book have been carefully designed so as to enable the readers to easily understand the key aspects of interactive graphics. Over the chapters, a lot of 3D presentations are provided in order to help the readers quickly grasp the complicated topics. An important organizational feature of this book is that theoretical or technical details are presented in separate notes (in shaded boxes) and in optional sections (marked by asterisk). They can be safely skipped without any difficulty in understanding the main stream of the book.

Two well-known graphics APIs are Direct3D and OpenGL. This book is API-neutral but presents the sample programs bit by bit in many places. They help the readers understand how the interactive graphics algorithms are implemented. However, the sample programs are located at the optional part and can be safely skipped.

If the optional parts are not taught, the content of this book would be appropriate for being covered in a semester for the undergraduate class. If needed, additional choices may be made among the required parts. For example, Sections 6.3 and 7.4 may be skipped.