Dedicated to Eva, Felix, and Elina
T. A-M.

Dedicated to Cathy, Ryan, and Evan
E. H.

Dedicated to Dorit, Karen, and Daniel
N. H.

Dedicated to Fei, Clelia, and Alberto
A. P.

Dedicated to Aneta and Weronika
M. I.

Dedicated to Stéphanie and Svea
S. H.
## Contents

### Preface

1 Introduction 1

1.1 Contents Overview 3

1.2 Notation and Definitions 5

2 The Graphics Rendering Pipeline 11

2.1 Architecture 12

2.2 The Application Stage 13

2.3 Geometry Processing 14

2.4 Rasterization 21

2.5 Pixel Processing 22

2.6 Through the Pipeline 25

3 The Graphics Processing Unit 29

3.1 Data-Parallel Architectures 30

3.2 GPU Pipeline Overview 34

3.3 The Programmable Shader Stage 35

3.4 The Evolution of Programmable Shading and APIs 37

3.5 The Vertex Shader 42

3.6 The Tessellation Stage 44

3.7 The Geometry Shader 47

3.8 The Pixel Shader 49

3.9 The Merging Stage 53

3.10 The Compute Shader 54

4 Transforms 57

4.1 Basic Transforms 58

4.2 Special Matrix Transforms and Operations 70

4.3 Quaternions 76

4.4 Vertex Blending 84

4.5 Morphing 87

4.6 Geometry Cache Playback 92

4.7 Projections 92
5 Shading Basics  103
5.1 Shading Models ............................................. 103
5.2 Light Sources .............................................. 106
5.3 Implementing Shading Models .............................. 117
5.4 Aliasing and Antialiasing ................................. 130
5.5 Transparency, Alpha, and Compositing .................... 148
5.6 Display Encoding ......................................... 160

6 Texturing  167
6.1 The Texturing Pipeline ..................................... 169
6.2 Image Texturing ............................................ 176
6.3 Procedural Texturing ...................................... 198
6.4 Texture Animation ......................................... 200
6.5 Material Mapping .......................................... 201
6.6 Alpha Mapping ............................................ 202
6.7 Bump Mapping ............................................. 208
6.8 Parallax Mapping .......................................... 214
6.9 Textured Lights ............................................ 221

7 Shadows  223
7.1 Planar Shadows ............................................. 225
7.2 Shadows on Curved Surfaces ............................... 229
7.3 Shadow Volumes ........................................... 230
7.4 Shadow Maps ................................................ 234
7.5 Percentage-Closer Filtering ............................... 247
7.6 Percentage-Closer Soft Shadows ......................... 250
7.7 Filtered Shadow Maps ..................................... 252
7.8 Volumetric Shadow Techniques ........................... 257
7.9 Irregular Z-Buffer Shadows ............................... 259
7.10 Other Applications ........................................ 262

8 Light and Color  267
8.1 Light Quantities ............................................ 267
8.2 Scene to Screen ............................................ 281

9 Physically Based Shading  293
9.1 Physics of Light ............................................ 293
9.2 The Camera ................................................ 307
9.3 The BRDF .................................................. 308
9.4 Illumination ............................................... 315
9.5 Fresnel Reflectance ....................................... 316
9.6 Microgeometry ............................................. 327
9.7 Microfacet Theory ......................................... 331
## Contents

9.8 BRDF Models for Surface Reflection ........................................ 336  
9.9 BRDF Models for Subsurface Scattering ............................... 347  
9.10 BRDF Models for Cloth .................................................. 356  
9.11 Wave Optics BRDF Models ............................................... 359  
9.12 Layered Materials .......................................................... 363  
9.13 Blending and Filtering Materials ....................................... 365

10 Local Illumination ................................................................. 375  
10.1 Area Light Sources .......................................................... 377  
10.2 Environment Lighting ....................................................... 391  
10.3 Spherical and Hemispherical Functions ............................... 392  
10.4 Environment Mapping ....................................................... 405  
10.5 Specular Image-Based Lighting .......................................... 415  
10.6 Irradiance Environment Mapping ....................................... 425  
10.7 Sources of Error .............................................................. 434

11 Global Illumination ................................................................. 437  
11.1 The Rendering Equation .................................................... 437  
11.2 General Global Illumination .......................................... 441  
11.3 Ambient Occlusion .......................................................... 446  
11.4 Directional Occlusion ....................................................... 465  
11.5 Diffuse Global Illumination .............................................. 472  
11.6 Specular Global Illumination ........................................... 497  
11.7 Unified Approaches ........................................................ 509

12 Image-Space Effects ................................................................. 513  
12.1 Image Processing ............................................................ 513  
12.2 Reprojection Techniques ................................................. 522  
12.3 Lens Flare and Bloom ....................................................... 524  
12.4 Depth of Field ............................................................... 527  
12.5 Motion Blur ................................................................. 536

13 Beyond Polygons ............................................................. 545  
13.1 The Rendering Spectrum ................................................ 545  
13.2 Fixed-View Effects ........................................................ 546  
13.3 Skyboxes ................................................................. 547  
13.4 Light Field Rendering .................................................... 549  
13.5 Sprites and Layers ......................................................... 550  
13.6 Billboardding .............................................................. 551  
13.7 Displacement Techniques .............................................. 564  
13.8 Particle Systems ............................................................ 567  
13.9 Point Rendering ........................................................... 572  
13.10 Voxels ................................................................. 578
## Contents

14 Volumetric and Translucency Rendering 589
   14.1 Light Scattering Theory .............................. 589
   14.2 Specialized Volumetric Rendering .................. 600
   14.3 General Volumetric Rendering ....................... 605
   14.4 Sky Rendering ....................................... 613
   14.5 Translucent Surfaces ................................. 623
   14.6 Subsurface Scattering ............................... 632
   14.7 Hair and Fur ......................................... 640
   14.8 Unified Approaches .................................. 648

15 Non-Photorealistic Rendering 651
   15.1 Toon Shading .......................................... 652
   15.2 Outline Rendering .................................... 654
   15.3 Stroke Surface Stylization ........................... 669
   15.4 Lines .................................................. 673
   15.5 Text Rendering ........................................ 675

16 Polygonal Techniques 681
   16.1 Sources of Three-Dimensional Data ................. 682
   16.2 Tessellation and Triangulation ..................... 683
   16.3 Consolidation ......................................... 690
   16.4 Triangle Fans, Strips, and Meshes ................ 696
   16.5 Simplification ........................................ 706
   16.6 Compression and Precision ........................... 712

17 Curves and Curved Surfaces 717
   17.1 Parametric Curves ..................................... 718
   17.2 Parametric Curved Surfaces ......................... 734
   17.3 Implicit Surfaces ..................................... 749
   17.4 Subdivision Curves .................................. 753
   17.5 Subdivision Surfaces ................................ 756
   17.6 Efficient Tessellation ................................ 767

18 Pipeline Optimization 783
   18.1 Profiling and Debugging Tools ....................... 784
   18.2 Locating the Bottleneck ............................. 786
   18.3 Performance Measurements ............................ 788
   18.4 Optimization .......................................... 790
   18.5 Multiprocessing ...................................... 805
19 Acceleration Algorithms 817
  19.1 Spatial Data Structures 818
  19.2 Culling Techniques 830
  19.3 Backface Culling 831
  19.4 View Frustum Culling 835
  19.5 Portal Culling 837
  19.6 Detail and Small Triangle Culling 839
  19.7 Occlusion Culling 840
  19.8 Culling Systems 850
  19.9 Level of Detail 852
  19.10 Rendering Large Scenes 866

20 Efficient Shading 881
  20.1 Deferred Shading 883
  20.2 Decal Rendering 888
  20.3 Tiled Shading 892
  20.4 Clustered Shading 898
  20.5 Deferred Texturing 905
  20.6 Object- and Texture-Space Shading 908

21 Virtual and Augmented Reality 915
  21.1 Equipment and Systems Overview 916
  21.2 Physical Elements 919
  21.3 APIs and Hardware 924
  21.4 Rendering Techniques 932

22 Intersection Test Methods 941
  22.1 GPU-Accelerated Picking 942
  22.2 Definitions and Tools 943
  22.3 Bounding Volume Creation 948
  22.4 Geometric Probability 953
  22.5 Rules of Thumb 954
  22.6 Ray/Sphere Intersection 955
  22.7 Ray/Box Intersection 959
  22.8 Ray/Triangle Intersection 962
  22.9 Ray/Polygon Intersection 966
  22.10 Plane/Box Intersection 970
  22.11 Triangle/Triangle Intersection 972
  22.12 Triangle/Box Intersection 974
  22.13 Bounding-Volume/Bounding-Volume Intersection 976
  22.14 View Frustum Intersection 981
  22.15 Line/Line Intersection 987
  22.16 Intersection between Three Planes 990
## Contents

23 Graphics Hardware

23.1 Rasterization ................................................. 993
23.2 Massive Compute and Scheduling .......................... 1002
23.3 Latency and Occupancy ....................................... 1004
23.4 Memory Architecture and Buses ............................ 1006
23.5 Caching and Compression .................................... 1007
23.6 Color Buffering ............................................... 1009
23.7 Depth Culling, Testing, and Buffering ..................... 1014
23.8 Texturing ..................................................... 1017
23.9 Architecture .................................................. 1019
23.10 Case Studies .................................................. 1024
23.11 Ray Tracing Architectures ................................. 1039

24 The Future

24.1 Everything Else ............................................... 1042
24.2 You .......................................................... 1046

Bibliography ......................................................... 1051

Index ............................................................... 1155
“Things have not changed *that* much in the past eight years,” was our thought entering into this fourth edition. “How hard could it be to update the book?” A year and a half later, and with three more experts recruited, our task is done. We could probably spend another year editing and elaborating, at which time there would be easily a hundred more articles and presentations to fold in. As a data point, we made a Google Doc of references that is more than 170 pages long, with about 20 references and related notes on each page. Some references we cite could and do each take up a full section in some other book. A few of our chapters, such as that on shadows, have entire books dedicated to their subjects. While creating more work for us, this wealth of information is good news for practitioners. We will often point to these primary sources, as they offer much more detail than appropriate here.

This book is about algorithms that create synthetic images fast enough that the viewer can interact with a virtual environment. We have focused on three-dimensional rendering and, to a limited extent, on the mechanics of user interaction. Modeling, animation, and many other areas are important to the process of making a real-time application, but these topics are beyond the scope of this book.

We expect you to have some basic understanding of computer graphics before reading this book, as well as knowledge of computer science and programming. We also focus on algorithms, not APIs. Many texts are available on these other subjects. If some section does lose you, skim on through or look at the references. We believe that the most valuable service we can provide you is a realization of what you yet do not know about—a basic kernel of an idea, a sense of what others have discovered about it, and ways to learn more, if you wish.

We make a point of referencing relevant material as possible, as well as providing a summary of further reading and resources at the end of most chapters. In prior editions we cited nearly everything we felt had relevant information. Here we are more a guidebook than an encyclopedia, as the field has far outgrown exhaustive (and exhausting) lists of all possible variations of a given technique. We believe you are better served by describing only a few representative schemes of many, by replacing original sources with newer, broader overviews, and by relying on you, the reader, to pursue more information from the references cited.

Most of these sources are but a mouse click away; see realtimerendering.com for the list of links to references in the bibliography. Even if you have only a passing interest in a topic, consider taking a little time to look at the related references, if for nothing else than to see some of the fantastic images presented. Our website also
Preface

contains links to resources, tutorials, demonstration programs, code samples, software libraries, book corrections, and more.

Our true goal and guiding light while writing this book was simple. We wanted to write a book that we wished we had owned when we had started out, a book that both was unified yet also included details and references not found in introductory texts. We hope that you will find this book, our view of the world, of use in your travels.

Acknowledgments for the Fourth Edition

We are not experts in everything, by any stretch of the imagination, nor perfect writers. Many, many people's responses and reviews improved this edition immeasurably, saving us from our own ignorance or inattention. As but one example, when we asked around for advice on what to cover in the area of virtual reality, Johannes Van Waferen (who did not know any of us) instantly responded with a wonderfully detailed outline of topics, which formed the basis for that chapter. These kind acts by computer graphics professionals were some of the great pleasures in writing this book. One person is of particular note: Patrick Cozzi did a yeoman's job, reviewing every chapter in the book. We are grateful to the many people who helped us along the way with this edition. We could write a sentence or three about everyone who helped us along the way, but this would push us further past our book-breaking page limit.

Nießner, Jim Nilsson, Reza Nourai, Chris Oat, Ola Olsson, Rafael Orozco, Bryan Pardilla, Steve Parker, Ankit Patel, Jasmin Patry, Jan Pechenik, Emil Persson, Marc Petit, Matt Pettineo, Agnieszka Piechnik, Jerome Platteaux, Aras Pranckevičius, Eleanor Quittner, Silvia Rasheva, Nathaniel Reed, Philip Rideout, Jon Rocatis, Robert Runesson, Marco Salvi, Nicolas Savva, Andrew Schneider, Michael Schneider, Markus Schuetz, Jeremy Selan, Tarek Sherif, Peter Shirley, Peter Sikachev, Peter-Pike Sloan, Ashley Vaughan Smith, Rys Sommefeldt, Edvard Sørgård, Tiago Sousa, Tomasz Staˇchowiak, Nick Stam, Lee Stemkoski, Jonathan Stone, Kier Storey, Jacob Ström, Filip Strugar, Pierre Terdiman, Aaron Thibault, Nicolas Thibieroz, Robert Toth, Thatcher Ulrich, Mauricio Vives, Alex Vlachos, Evan Wallace, Ian Webster, Nick Whiting, Brandon Whitley, Mattias Widmark, Graham Wihlidal, Michael Wimmer, Daniel Wright, Bart Wroński, Chris Wyman, Ke Xu, Cem Yuksel, and Egor Yusov. We thank you for your time and effort, selflessly offered and gratefully received.

Finally, we want to thank the people at Taylor & Francis for all their efforts, in particular Rick Adams, for getting us going and guiding us along the way, Jessica Vega and Michele Dimont, for their efficient editorial work, and Charlotte Byrnes, for her superb copyediting.

Tomas Akenine-Möller
Eric Haines
Naty Hoffman
Angelo Pesce
Michał Iwanicki
Sébastien Hillaire

February 2018

Acknowledgments for the Third Edition

Special thanks go out to a number of people who went out of their way to provide us with help. First, our graphics architecture case studies would not have been anywhere as good without the extensive and generous cooperation we received from the companies making the hardware. Many thanks to Edvard Sørgård, Borgar Ljosland, Dave Shreiner, and Jørn Nystad at ARM for providing details about their Mali 200 architecture. Thanks also to Michael Dougherty at Microsoft, who provided extremely valuable help with the Xbox 360 section. Masaaki Oka at Sony Computer Entertainment provided his own technical review of the PLAYSTATION® 3 system case study, while also serving as the liaison with the Cell Broadband Engine™ and RSX® developers for their reviews.

In answering a seemingly endless stream of questions, fact-checking numerous passages, and providing many screenshots, Natalya Tatarchuk of ATI/AMD went well beyond the call of duty in helping us out. In addition to responding to our usual requests for information and clarification, Wolfgang Engel was extremely helpful in providing us with articles from the upcoming ShaderX® book and copies of the difficult-to-
obtain ShaderX² books [427, 428], now available online for free. Ignacio Castaño at NVIDIA provided us with valuable support and contacts, going so far as to rework a refractory demo so we could get just the right screenshot.

The chapter reviewers provided an invaluable service to us. They suggested numerous improvements and helped providing additional insights, helping us immeasurably. In alphabetical order they are: Michael Ashikhmin, Dan Baker, Willem de Boer, Ben Diamand, Ben Discoc, Amir Ebrahimi, Christer Ericson, Michael Gleicher, Manny Ko, Wallace Lages, Thomas Larsson, Grégory Massal, Ville Miettinen, Mike Ramsey, Scott Schaefer, Vincent Scheib, Peter Shirley, K.R. Subramanian, Mauricio Vives, and Hector Yee.

We also had a number of reviewers help us on specific sections. Our thanks go out to Matt Bronder, Christine DeNezza, Frank Fox, Jon Hasselgren, Pete Isensee, Andrew Lauritzen, Morgan McGuire, Jacob Munkberg, Manuel M. Oliveira, Aurelio Reis, Peter-Pike Sloan, Jim Tilander, and Scott Whitman.

We particularly thank Rex Crowle, Kareem Ettouney, and Francis Pang from Media Molecule for their considerable help in providing fantastic imagery and layout concepts for the cover design.

Many people helped us out in other ways, such as answering questions and providing screenshots. Many gave significant amounts of time and effort, for which we thank you. Listed alphabetically: Paulo Abreu, Timo Aila, Johan Andersson, Andreas Bærentzen, Louis Bavoil, Jim Blinn, Jaime Borasi, Per Christensen, Patrick Conran, Rob Cook, Erwin Coumans, Leo Cubbin, Richard Daniels, Mark DeLoura, Tony DeRose, Andreas Dietrich, Michael Dougherty, Bryan Dudash, Alex Evans, Cass Everitt, Randy Fernando, Jim Ferwerda, Chris Ford, Tom Forsyth, Sam Glassenberg, Robin Green, Ned Greene, Larry Gritz, Joakim Grundwall, Mark Harris, Ted Himlan, Jack Hoxley, John “Spike” Hughes, Ladislav Kavan, Alicia Kim, Gary King, Chris Lambert, Jeff Lander, Daniel Leaver, Eric Lengyel, Jennifer Liu, Brandon Lloyd, Charles Loop, David Luebke, Jonathan Måin, Jason Mitchell, Martin Mittring, Nathan Monteleone, Gabe Newell, Hubert Nguyen, Petri Nordlund, Mike Pan, Ivan Pedersen, Matt Pharr, Fabio Policarpo, Aras Pranckevičius, Siobhan Reddy, Dirk Reiners, Christof Rezk-Salama, Eric Risser, Marcus Roth, Holly Rushmeier, Elan Ruskin, Marco Salvi, Daniel Scherzer, Kyle Shubel, Philipp Slusallek, Torbjörn Söderman, Tim Sweeney, Ben Trumbore, Michal Valient, Mark Valledor, Carsten Wenzel, Steve Westin, Chris Wyman, Cem Yuksel, Billy Zelnack, Fan Zhang, and Renaldas Zioma.

We also thank many others who responded to our queries on public forums such as GD Algorithms. Readers who took the time to send us corrections have also been a great help. It is this supportive attitude that is one of the pleasures of working in this field.

As we have come to expect, the cheerful competence of the people at A K Peters made the publishing part of the process much easier. For this wonderful support, we thank you all.
On a personal note, Tomas would like to thank his son Felix and daughter Elina for making him understand (again) just how fun it can be to play computer games (on the Wii), instead of just looking at the graphics, and needless to say, his beautiful wife Eva...

Eric would also like to thank his sons Ryan and Evan for their tireless efforts in finding cool game demos and screenshots, and his wife Cathy for helping him survive it all.

Naty would like to thank his daughter Karen and son Daniel for their forbearance when writing took precedence over piggyback rides, and his wife Dorit for her constant encouragement and support.

Tomas Akenine-Möller
Eric Haines
Naty Hoffman
March 2008

Acknowledgments for the Second Edition
One of the most agreeable aspects of writing this second edition has been working with people and receiving their help. Despite their own pressing deadlines and concerns, many people gave us significant amounts of their time to improve this book. We would particularly like to thank the major reviewers. They are, listed alphabetically: Michael Abrash, Ian Ashdown, Ulf Assarsson, Chris Brennan, Sébastien Dominé, David Eberly, Cass Everitt, Tommy Fortes, Evan Hart, Greg James, Jan Kautz, Alexander Keller, Mark Kilgard, Adam Lake, Paul Lalonde, Thomas Larsson, Dean Macri, Carl Marshall, Jason L. Mitchell, Kasper Høy Nielsen, Jon Paul Schelter, Jacob Ström, Nick Triantos, Joe Warren, Michael Wimmer, and Peter Wonka. Of these, we wish to single out Cass Everitt at NVIDIA and Jason L. Mitchell at ATI Technologies for spending large amounts of time and effort in getting us the resources we needed. Our thanks also go out to Wolfgang Engel for freely sharing the contents of his upcoming book, ShaderX [426], so that we could make this edition as current as possible.

From discussing their work with us, to providing images or other resources, to writing reviews of sections of the book, many others helped in creating this edition. They all have our gratitude. These people include: Jason Ang, Haim Barad, Jules Bloomenthal, Jonathan Blow, Chas. Boyd, John Brooks, Cem Cebenoyan, Per Christensen, Hamilton Chu, Michael Cohen, Daniel Cohen-Or, Matt Craighead, Paul Debevec, Joe Demers, Walt Donovan, Howard Dortch, Mark Duchaineau, Phil Dutré, Dave Eberle, Gerald Farin, Simon Fenney, Randy Fernando, Jim Ferwerda, Nickson Fong, Tom Forsyth, Piero Foscari, Laura Fryer, Markus Gieg, Peter Glaskowsky, Andrew Glassner, Amy Gooch, Bruce Gooch, Simon Green, Ned Greene, Larry Gritz, Joakim Grundwall, Juan Guardado, Pat Hanrahan, Mark Harris, Michael Herf, Carsten Hess, Rich Hilmer, Kenneth Hoff III, Naty Hoffman, Nick Holliman, Hugues Hoppe, Heather Horne, Tom Hubina, Richard Huddy, Adam James, Kaveh Kardan, Paul Keller, David
Preface

Kirk, Alex Klimovitski, Jason Knipe, Jeff Lander, Marc Levoy, J.P. Lewis, Ming Lin, Adrian Lopez, Michael McCool, Doug McNabb, Stan Melax, Ville Miettinen, Kenny Mitchell, Steve Morein, Henry Moreton, Jerris Mungai, Jim Napier, George Ngo, Hubert Nguyen, Tito Pagán, Jörg Peters, Tom Porter, Emil Praun, Kekoa Proudfoot, Bernd Raabe, Ravi Ramamoorthi, Ashutosh Rege, Szymon Rusinkiewicz, Chris Seitz, Carlo Séquin, Jonathan Shade, Brian Smits, John Spitzer, Wolfgang Straßer, Wolfgang Stürzlinger, Philip Taylor, Pierre Terdiman, Nicolas Thibieroz, Jack Tumblin, Fredrik Ulfves, Thatcher Ulrich, Steve Upstill, Alex Vlachos, Ingo Wald, Ben Watson, Steve Westin, Dan Wexler, Matthias Wloka, Peter Woytiuk, David Wu, Garrett Young, Borut Zalik, Harold Zatz, Hansong Zhang, and Denis Zorin. We also wish to thank the journal ACM Transactions on Graphics for providing a mirror website for this book.

Alice and Klaus Peters, our production manager Ariel Jaffee, our editor Heather Holcombe, our copyeditor Michelle M. Richards, and the rest of the staff at A K Peters have done a wonderful job making this book the best possible. Our thanks to all of you.

Finally, and most importantly, our deepest thanks go to our families for giving us the huge amounts of quiet time we have needed to complete this edition. Honestly, we never thought it would take this long!

Tomas Akenine-Möller
Eric Haines
May 2002

Acknowledgments for the First Edition

Many people helped in making this book. Some of the greatest contributions were made by those who reviewed parts of it. The reviewers willingly gave the benefit of their expertise, helping to significantly improve both content and style. We wish to thank (in alphabetical order) Thomas Barregren, Michael Cohen, Walt Donovan, Angus Dorbie, Michael Garland, Stefan Gottschalk, Neil Greene, Ming C. Lin, Jason L. Mitchell, Liang Peng, Keith Rule, Ken Shoemake, John Stone, Phil Taylor, Ben Trumbore, Jorrit Tyberghien, and Nick Wilt. We cannot thank you enough.

Many other people contributed their time and labor to this project. Some let us use images, others provided models, still others pointed out important resources or connected us with people who could help. In addition to the people listed above, we wish to acknowledge the help of Tony Barkans, Daniel Baum, Nelson Beebe, Curtis Bee- son, Tor Berg, David Blythe, Chas. Boyd, Don Brittain, Ian Bullard, Javier Castellar, Satyan Coorg, Jason Della Rocca, Paul Diefenbach, Alyssa Donovan, Dave Eberly, Kellis Elmquist, Stuart Feldman, Fred Fisher, Tom Forsyth, Marty Franz, Thomas Funkhouser, Andrew Glassner, Bruce Gooch, Larry Gritz, Robert Grzeszczuk, Paul Haebeler, Evan Hart, Paul Heckbert, Chris Hecker, Joachim Helenklaken, Hugues Hoppe, John Jack, Mark Kilgard, David Kirk, James Klosowski, Subodh Kumar, André LaMothe, Jeff Lander, Jens Larsson, Jed Lengyel, Fredrik Lillegren, David Luebke, Thomas Lundqvist, Tom McReynolds, Stan Melax, Don Mitchell, André Möller,
Steve Molnar, Scott R. Nelson, Hubert Nguyen, Doug Rogers, Holly Rushmeier, Gernot Schaufler, Jonas Skeppstedt, Stephen Spencer, Per Stenström, Jacob Ström, Filippo Tampieri, Gary Tarolli, Ken Turkowski, Turner Whitted, Agata and Andrzej Wojaczek, Andrew Woo, Steve Worley, Brian Yen, Hans-Philip Zachau, Gabriel Zachmann, and Al Zimmerman. We also wish to thank the journal ACM Transactions on Graphics for providing a stable website for this book.

Alice and Klaus Peters and the staff at AK Peters, particularly Carolyn Artin and Sarah Gillis, have been instrumental in making this book a reality. To all of you, thanks.

Finally, our deepest thanks go to our families and friends for providing support throughout this incredible, sometimes grueling, often exhilarating process.

Tomas Möller
Eric Haines
March 1999
Chapter 1
Introduction

Real-time rendering is concerned with rapidly making images on the computer. It is the most highly interactive area of computer graphics. An image appears on the screen, the viewer acts or reacts, and this feedback affects what is generated next. This cycle of reaction and rendering happens at a rapid enough rate that the viewer does not see individual images, but rather becomes immersed in a dynamic process.

The rate at which images are displayed is measured in frames per second (FPS) or Hertz (Hz). At one frame per second, there is little sense of interactivity; the user is painfully aware of the arrival of each new image. At around 6 FPS, a sense of interactivity starts to grow. Video games aim for 30, 60, 72, or higher FPS; at these speeds the user focuses on action and reaction.

Movie projectors show frames at 24 FPS but use a shutter system to display each frame two to four times to avoid flicker. This refresh rate is separate from the display rate and is expressed in Hertz (Hz). A shutter that illuminates the frame three times has a 72 Hz refresh rate. LCD monitors also separate refresh rate from display rate.

Watching images appear on a screen at 24 FPS might be acceptable, but a higher rate is important for minimizing response time. As little as 15 milliseconds of temporal delay can slow and interfere with interaction [1849]. As an example, head-mounted displays for virtual reality often require 90 FPS to minimize latency.

There is more to real-time rendering than interactivity. If speed was the only criterion, any application that rapidly responded to user commands and drew anything on the screen would qualify. Rendering in real time normally means producing three-dimensional images.

Interactivity and some sense of connection to three-dimensional space are sufficient conditions for real-time rendering, but a third element has become a part of its definition: graphics acceleration hardware. Many consider the introduction of the 3Dfx Voodoo 1 card in 1996 the real beginning of consumer-level three-dimensional graphics [408]. With the rapid advances in this market, every computer, tablet, and mobile phone now comes with a graphics processor built in. Some excellent examples of the results of real-time rendering made possible by hardware acceleration are shown in Figures 1.1 and 1.2.
Advances in graphics hardware have fueled an explosion of research in the field of interactive computer graphics. We will focus on providing methods to increase speed and improve image quality, while also describing the features and limitations of acceleration algorithms and graphics APIs. We will not be able to cover every topic in depth, so our goal is to present key concepts and terminology, explain the most robust and practical algorithms in the field, and provide pointers to the best places to go for more information. We hope our attempts to provide you with tools for understanding this field prove to be worth the time and effort you spend with our book.
1.1 Contents Overview

What follows is a brief overview of the chapters ahead.

Chapter 2, The Graphics Rendering Pipeline. The heart of real-time rendering is the set of steps that takes a scene description and converts it into something we can see.

Chapter 3, The Graphics Processing Unit. The modern GPU implements the stages of the rendering pipeline using a combination of fixed-function and programmable units.

Chapter 4, Transforms. Transforms are the basic tools for manipulating the position, orientation, size, and shape of objects and the location and view of the camera.

Chapter 5, Shading Basics. Discussion begins on the definition of materials and lights and their use in achieving the desired surface appearance, whether realistic or stylized. Other appearance-related topics are introduced, such as providing higher image quality through the use of antialiasing, transparency, and gamma correction.

Chapter 6, Texturing. One of the most powerful tools for real-time rendering is the ability to rapidly access and display images on surfaces. This process is called texturing, and there are a wide variety of methods for applying it.

Chapter 7, Shadows. Adding shadows to a scene increases both realism and comprehension. The more popular algorithms for computing shadows rapidly are presented.

Chapter 8, Light and Color. Before we perform physically based rendering, we first need to understand how to quantify light and color. And after our physical rendering process is done, we need to transform the resulting quantities into values for the display, accounting for the properties of the screen and viewing environment. Both topics are covered in this chapter.

Chapter 9, Physically Based Shading. We build an understanding of physically based shading models from the ground up. The chapter starts with the underlying physical phenomena, covers models for a variety of rendered materials, and ends with methods for blending materials together and filtering them to avoid aliasing and preserve surface appearance.

Chapter 10, Local Illumination. Algorithms for portraying more elaborate light sources are explored. Surface shading takes into account that light is emitted by physical objects, which have characteristic shapes.

Chapter 11, Global Illumination. Algorithms that simulate multiple interactions between the light and the scene further increase the realism of an image. We discuss ambient and directional occlusion and methods for rendering global illumination effects on diffuse and specular surfaces, as well as some promising unified approaches.

Chapter 12, Image-Space Effects. Graphics hardware is adept at performing image processing at rapid speeds. Image filtering and reprojection techniques are discussed
first, then we survey several popular post-processing effects: lens flares, motion blur, and depth of field.

Chapter 13, Beyond Polygons. Triangles are not always the fastest or most realistic way to describe objects. Alternate representations based on using images, point clouds, voxels, and other sets of samples each have their advantages.

Chapter 14, Volumetric and Translucency Rendering. The focus here is the theory and practice of volumetric material representations and their interactions with light sources. The simulated phenomena range from large-scale atmospheric effects down to light scattering within thin hair fibers.

Chapter 15, Non-Photorealistic Rendering. Attempting to make a scene look realistic is only one way of rendering it. Other styles, such as cartoon shading and watercolor effects, are surveyed. Line and text generation techniques are also discussed.

Chapter 16, Polygonal Techniques. Geometric data comes from a wide range of sources, and sometimes requires modification to be rendered rapidly and well. The many facets of polygonal data representation and compression are presented.

Chapter 17, Curves and Curved Surfaces. More complex surface representations offer advantages such as being able to trade off between quality and rendering speed, more compact representation, and smooth surface generation.

Chapter 18, Pipeline Optimization. Once an application is running and uses efficient algorithms, it can be made even faster using various optimization techniques. Finding the bottleneck and deciding what to do about it is the theme here. Multiprocessing is also discussed.

Chapter 19, Acceleration Algorithms. After you make it go, make it go fast. Various forms of culling and level of detail rendering are covered.

Chapter 20, Efficient Shading. A large number of lights in a scene can slow performance considerably. Fully shading surface fragments before they are known to be visible is another source of wasted cycles. We explore a wide range of approaches to tackle these and other forms of inefficiency while shading.

Chapter 21, Virtual and Augmented Reality. These fields have particular challenges and techniques for efficiently producing realistic images at rapid and consistent rates.

Chapter 22, Intersection Test Methods. Intersection testing is important for rendering, user interaction, and collision detection. In-depth coverage is provided here for a wide range of the most efficient algorithms for common geometric intersection tests.

Chapter 23, Graphics Hardware. The focus here is on components such as color depth, framebuffers, and basic architecture types. A case study of representative GPUs is provided.

Chapter 24, The Future. Take a guess (we do).
Due to space constraints, we have made a chapter about Collision Detection free for download at realtime rendering.com, along with appendices on linear algebra and trigonometry.

### 1.2 Notation and Definitions

First, we shall explain the mathematical notation used in this book. For a more thorough explanation of many of the terms used in this section, and throughout this book, get our linear algebra appendix at realtime rendering.com.

#### 1.2.1 Mathematical Notation

Table 1.1 summarizes most of the mathematical notation we will use. Some of the concepts will be described at some length here.

Note that there are some exceptions to the rules in the table, primarily shading equations using notation that is extremely well established in the literature, e.g., $L$ for radiance, $E$ for irradiance, and $\sigma_s$ for scattering coefficient.

The angles and the scalars are taken from $\mathbb{R}$, i.e., they are real numbers. Vectors and points are denoted by bold lowercase letters, and the components are accessed as

$$
v = \begin{pmatrix} v_x \\ v_y \\ v_z \end{pmatrix},
$$

that is, in column vector format, which is commonly used in the computer graphics world. At some places in the text we use $(v_x, v_y, v_z)$ instead of the formally more correct $(v_x \ v_y \ v_z)^T$, since the former is easier to read.

<table>
<thead>
<tr>
<th>Type</th>
<th>Notation</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>angle</td>
<td>lowercase Greek</td>
<td>$\alpha, \phi, \rho, \eta, \gamma_2, 42, \theta$</td>
</tr>
<tr>
<td>scalar</td>
<td>lowercase italic</td>
<td>$a, b, t, u, v, w, i, j$</td>
</tr>
<tr>
<td>vector or point</td>
<td>lowercase bold</td>
<td>$a, u, v, h(\rho)$</td>
</tr>
<tr>
<td>matrix</td>
<td>capital bold</td>
<td>$T(t), X, R_x(\rho)$</td>
</tr>
<tr>
<td>plane</td>
<td>$\pi$: a vector and a scalar</td>
<td>$\pi: n \cdot x + d = 0$, $\pi_1: n_1 \cdot x + d_1 = 0$</td>
</tr>
<tr>
<td>triangle</td>
<td>$\triangle$ 3 points</td>
<td>$\triangle v_0 v_1 v_2, \triangle c b a$</td>
</tr>
<tr>
<td>line segment</td>
<td>two points</td>
<td>$uv, a, b_j$</td>
</tr>
<tr>
<td>geometric entity</td>
<td>capital italic</td>
<td>$A_{OBB}, T, B_{ABB}$</td>
</tr>
</tbody>
</table>

**Table 1.1.** Summary of the notation used in this book.
Using homogeneous notation, a coordinate is represented by four values \( \mathbf{v} = (v_x \ v_y \ v_z \ v_w)^T \), where a vector is \( \mathbf{v} = (v_x \ v_y \ v_z \ 0)^T \) and a point is \( \mathbf{v} = (v_x \ v_y \ v_z \ 1)^T \). Sometimes we use only three-element vectors and points, but we try to avoid any ambiguity as to which type is being used. For matrix manipulations, it is extremely advantageous to have the same notation for vectors as for points. For more information, see Chapter 4 on transforms. In some algorithms, it will be convenient to use numeric indices instead of \( x, y, \) and \( z \), for example \( \mathbf{v} = (v_0 \ v_1 \ v_2)^T \). All these rules for vectors and points also hold for two-element vectors; in that case, we simply skip the last component of a three-element vector.

The matrix deserves a bit more explanation. The common sizes that will be used are \( 2 \times 2, 3 \times 3, \) and \( 4 \times 4 \). We will review the manner of accessing a \( 3 \times 3 \) matrix \( \mathbf{M} \), and it is simple to extend this process to the other sizes. The (scalar) elements of \( \mathbf{M} \) are denoted \( m_{ij} \), \( 0 \leq (i,j) \leq 2 \), where \( i \) denotes the row and \( j \) the column, as in Equation 1.1:

\[
\mathbf{M} = \begin{pmatrix}
    m_{00} & m_{01} & m_{02} \\
    m_{10} & m_{11} & m_{12} \\
    m_{20} & m_{21} & m_{22}
\end{pmatrix}
\]  

(1.1)

The following notation, shown in Equation 1.2 for a \( 3 \times 3 \) matrix, is used to isolate vectors from the matrix \( \mathbf{M} \): \( \mathbf{m}_{ij} \) represents the \( j \)th column vector and \( \mathbf{m}_{i} \) represents the \( i \)th row vector (in column vector form). As with vectors and points, indexing the column vectors can also be done with \( x, y, \) and sometimes \( w \), if that is more convenient:

\[
\mathbf{M} = ( \begin{pmatrix} m_0 \\ m_1 \\ m_2 \end{pmatrix} ) = ( \begin{pmatrix} \mathbf{m}_x \\ \mathbf{m}_y \\ \mathbf{m}_z \end{pmatrix} ) = ( \begin{pmatrix} \mathbf{m}_0^T \\ \mathbf{m}_1^T \\ \mathbf{m}_2^T \end{pmatrix} ).
\]  

(1.2)

A plane is denoted \( \pi : \mathbf{n} \cdot \mathbf{x} + d = 0 \) and contains its mathematical formula, the plane normal \( \mathbf{n} \) and the scalar \( d \). The normal is a vector describing what direction the plane faces. More generally (e.g., for curved surfaces), a normal describes this direction for a particular point on the surface. For a plane the same normal happens to apply to all its points. \( \pi \) is the common mathematical notation for a plane. The plane \( \pi \) is said to divide the space into a positive half-space, where \( \mathbf{n} \cdot \mathbf{x} + d > 0 \), and a negative half-space, where \( \mathbf{n} \cdot \mathbf{x} + d < 0 \). All other points are said to lie in the plane.

A triangle can be defined by three points \( \mathbf{v}_0, \mathbf{v}_1, \) and \( \mathbf{v}_2 \) and is denoted by \( \triangle \mathbf{v}_0 \mathbf{v}_1 \mathbf{v}_2 \). Table 1.2 presents some additional mathematical operators and their notation. The dot, cross, determinant, and length operators are explained in our downloadable linear algebra appendix at realtime rendering.com. The transpose operator turns a column vector into a row vector and vice versa. Thus a column vector can be written in compressed form in a block of text as \( \mathbf{v} = (v_x \ v_y \ v_z)^T \). Operator 4, introduced in Graphics Gems IV [735], is a unary operator on a two-dimensional vector. Letting
1.2. Notation and Definitions

<table>
<thead>
<tr>
<th>Operator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1: ·</td>
<td>dot product</td>
</tr>
<tr>
<td>2: ×</td>
<td>cross product</td>
</tr>
<tr>
<td>3: ( \mathbf{v}^T )</td>
<td>transpose of the vector ( \mathbf{v} )</td>
</tr>
<tr>
<td>4: ⊥</td>
<td>the unary, perp dot product operator</td>
</tr>
<tr>
<td>5:</td>
<td>determinant of a matrix</td>
</tr>
<tr>
<td>6:</td>
<td>absolute value of a scalar</td>
</tr>
<tr>
<td>7: ( | \cdot | )</td>
<td>length (or norm) of argument</td>
</tr>
<tr>
<td>8: ( x^+ )</td>
<td>clamping ( x ) to 0</td>
</tr>
<tr>
<td>9: ( x^\pi )</td>
<td>clamping ( x ) between 0 and 1</td>
</tr>
<tr>
<td>10: ( n! )</td>
<td>factorial</td>
</tr>
<tr>
<td>11: ( \binom{n}{k} )</td>
<td>binomial coefficients</td>
</tr>
</tbody>
</table>

Table 1.2. Notation for some mathematical operators.

This operator work on a vector \( \mathbf{v} = (v_x \ v_y)^T \) gives a vector that is perpendicular to \( \mathbf{v} \), i.e., \( \mathbf{v}^\perp = (-v_y \ v_x)^T \). We use \( |a| \) to denote the absolute value of the scalar \( a \), while \( |A| \) means the determinant of the matrix \( A \). Sometimes, we also use \( |A| = |a \ b \ c| = \det(a \ b \ c) \), where \( a, b, \) and \( c \) are column vectors of the matrix \( A \).

Operators 8 and 9 are clamping operators, commonly used in shading calculations. Operator 8 clamps negative values to 0:

\[
x^+ = \begin{cases} 
    x, & \text{if } x > 0, \\
    0, & \text{otherwise,} 
\end{cases} \tag{1.3}
\]

and operator 9 clamps values between 0 and 1:

\[
    x^\pi = \begin{cases} 
        1, & \text{if } x \geq 1, \\
        x, & \text{if } 0 < x < 1, \\
        0, & \text{otherwise.} 
    \end{cases} \tag{1.4}
\]

The tenth operator, factorial, is defined as shown below, and note that \( 0! = 1 \):

\[
n! = n(n-1)(n-2) \cdots 3 \cdot 2 \cdot 1. \tag{1.5}
\]

The eleventh operator, the binomial factor, is defined as shown in Equation 1.6:

\[
    \binom{n}{k} = \frac{n!}{k!(n-k)!}. \tag{1.6}
\]
1. Introduction

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>atan2(y, x)</td>
<td>two-value arctangent</td>
</tr>
<tr>
<td>log(n)</td>
<td>natural logarithm of n</td>
</tr>
</tbody>
</table>

Table 1.3. Notation for some specialized mathematical functions.

Further on, we call the common planes $x = 0$, $y = 0$, and $z = 0$ the coordinate planes or axis-aligned planes. The axes $e_x = (1 0 0)^T$, $e_y = (0 1 0)^T$, and $e_z = (0 0 1)^T$ are called main axes or main directions and individually called the $x$-axis, $y$-axis, and $z$-axis. This set of axes is often called the standard basis. Unless otherwise noted, we will use orthonormal bases (consisting of mutually perpendicular unit vectors).

The notation for a range that includes both $a$ and $b$, and all numbers in between, is $[a, b]$. If we want all number between $a$ and $b$, but not $a$ and $b$ themselves, then we write $(a, b)$. Combinations of these can also be made, e.g., $[a, b)$ means all numbers between $a$ and $b$ including $a$ but not $b$.

The C-math function $\text{atan2}(y, x)$ is often used in this text, and so deserves some attention. It is an extension of the mathematical function $\arctan(x)$. The main differences between them are that $-\frac{\pi}{2} < \arctan(x) < \frac{\pi}{2}$, that $0 \leq \text{atan2}(y, x) < 2\pi$, and that an extra argument has been added to the latter function. A common use for $\arctan$ is to compute $\arctan(y/x)$, but when $x = 0$, division by zero results. The extra argument for $\text{atan2}(y, x)$ avoids this.

In this volume the notation $\log(n)$ always means the natural logarithm, $\log_e(n)$, not the base-10 logarithm, $\log_{10}(n)$.

We use a right-hand coordinate system since this is the standard system for three-dimensional geometry in the field of computer graphics.

Colors are represented by a three-element vector, such as $(\text{red}, \text{green}, \text{blue})$, where each element has the range $[0, 1]$.

1.2.2 Geometrical Definitions

The basic rendering primitives (also called drawing primitives) used by almost all graphics hardware are points, lines, and triangles.\(^1\)

Throughout this book, we will refer to a collection of geometric entities as either a model or an object. A scene is a collection of models comprising everything that is included in the environment to be rendered. A scene can also include material descriptions, lighting, and viewing specifications.

Examples of objects are a car, a building, and even a line. In practice, an object often consists of a set of drawing primitives, but this may not always be the case; an object may have a higher kind of geometrical representation, such as Bézier curves or

\(^1\)The only exceptions we know of are Pixel-Planes [502], which could draw spheres, and the NVIDIA NV1 chip, which could draw ellipsoids.
surfaces, or subdivision surfaces. Also, objects can consist of other objects, e.g., a car object includes four door objects, four wheel objects, and so on.

1.2.3 Shading

Following well-established computer graphics usage, in this book terms derived from “shading,” “shader,” and related words are used to refer to two distinct but related concepts: computer-generated visual appearance (e.g., “shading model,” “shading equation,” “toon shading”) or a programmable component of a rendering system (e.g., “vertex shader,” “shading language”). In both cases, the intended meaning should be clear from the context.

Further Reading and Resources

The most important resource we can refer you to is the website for this book: realtimerendering.com. It contains links to the latest information and websites relevant to each chapter. The field of real-time rendering is changing with real-time speed. In the book we have attempted to focus on concepts that are fundamental and techniques that are unlikely to go out of style. On the website we have the opportunity to present information that is relevant to today’s software developer, and we have the ability to keep it up-to-date.
Bibliography


1056

Bibliography


Bibliography


Bibliography


Bibliography


1076

Bibliography


Cited on p. 234, 241, 242

Cited on p. 701, 705


p. 795, 796, 802, 803

p. 917

p. 208

p. 161, 162, 163


[493] Franklin, Dustin, “Hardware-Based Ambient Occlusion,” in Wolfgang Engel, ed., ShaderX4,
Charles River Media, pp. 91–100, 2005. Cited on p. 452

[494] Frey, Ivo Zoltan, “Spherical Skinning with Dual-Quaternions and QTangents,” in ACM SIG-

[495] Frisken, Sarah, Ronald N. Perry, Alyn P. Rockwood, and Thouis R. Jones, “Adaptively Sam-
pled Distance Fields: A General Representation of Shape for Computer Graphics,” in SIG-
GRAPH ’00: Proceedings of the 27th Annual Conference on Computer Graphics and Interac-
p. 677, 751

[496] Frisvad, Jeppe Revall, “Building an Orthonormal Basis from a 3D Unit Vector Without Normal-

[497] Fry, Alex, “High Dynamic Range Color Grading and Display in Frostbite,” Game Developers

[498] Frykholm, Niklas, “The BitSquid Low Level Animation System,” Autodesk Stingray blog,

[499] Frykholm, Niklas, “What Is Gimbal Lock and Why Do We Still Have to Worry about It?”
Autodesk Stingray blog, Mar. 15, 2013. Cited on p. 73


on p. 823


Bibliography


Bibliography


Bibliography


Bibliography


Cited on p. 185


Bibliography


Bibliography


Bibliography


Bibliography


Bibliography


Bibliography


Bibliography


Bibliography


Bibliography


Bibliography


Bibliography


Bibliography


Bibliography


Bibliography


Bibliography


Bibliography


Bibliography


Index

For indexed terms with more than one page reference, the italicized page (or range) indicates the most significant reference.

1-ring, 759, 760
2.5 dimensions, 531
3D printing, 578, 683, 693
3dfx Interactive, 37
A-buffer, see under buffer
AABB, 822, 944, 946, 974, 976
creation, 949
orthographic projection, 93
AABB/object intersection, see intersection testing, AABB/AABB
Academy Color Encoding System, see ACES
acceleration algorithms, 14, 682, 817–879
see also optimization
accessibility shading, 449
accommodation, 923
accumulation buffer, see buffer, accumulation
ACE, 1036
ACES, 287
adaptation, 285
adaptive refinement, 547
adjacency graph, 692
affinity mask, 928
Afro Samurai, 658
AHD basis, see basis, AHD
albedo, 314
directional, 313
texture, 885
aliasing, 130, 131
crawlies, 130
fireflies, 132, 801
jaggies, 130, 132, 537
perspective, 239
projective, 240
self-shadow, 236
shadow map, 236
temporal, 132, 182
texture, 182, 186
alpha, 149, 159–160, 202–208
blending, 149–150, 203
channel, 24, 159, 160, 203, 1010
LOD, see level of detail, alpha
mapping, see texturing
premultiplied, 159–160
testing, 24, 204
unmultiplied, 160
alpha to coverage, 149, 207
alternate frame rendering, 1013
ALU, 1002–1003, 1029, 1036
ambient
aperture lighting, 466
color, 392
cube, 394–395, 432, 478, 488
dice, 395, 478, 488
light, see light, ambient
ambient occlusion, 446–451
dynamic, 453–457
field, 452
ground-truth, 461
horizon-based, 460
precomputed, 451–453
screen-space, 457–463
shading, 463–465
temporal supersampling, 462
volume, 452
volumetric, 459
ambient/highlight/direction basis, 476
Amdahl’s law, 1020
animation, 81, 85, 200, 829
cel, 652
impostor, 562
particle system, 567
sprite, 550
subdivision, 781
texture, see texturing, animation
vertex blending, see under transform
anisotropic filtering, see under texturing, minification
anisotropic reflection, 314
anisotropic scaling, 62
anti-shadow, 227
anisotropic filtering, 130–148
anti-shadow, 227
coverage sampling, 141
custom filter, 142
directionally localized, 147
distance-to-edge, 147
enhanced quality, 141
fast approximate, 146, 148
FLIPQUAD, 146
full-scene, 138
gyere buffer, 147
hybrid reconstruction, 146
image-based, 147, 148
jittering, 144, 909
morphological, 146–148
multisampling, 139–142, 144–148, 155
N-rooks, 143
Quincunx, 145–146
rotated grid, 143, 145, 146
screen based, 137–148, 204–207
subpixel morphological, 148
subpixel reconstruction, 147
supersampling, 138–139
rotated grid, 1028
temporal, 142–143, 144–146, 148
texture, see texturing, minification
aperture, 307
apex point map, 837, 980
API, 15
application stage, see under pipeline
arithmetic logic unit, see ALU
artistic rendering, see non-photorealistic rendering
Ashes of the Singularity, 883, 911, 913
Ashikhmin model, 357
aspect graph, see under spatial data structure
Assassin’s Creed, 453, 539
Assassin’s Creed 4: Black Flag, 478, 481
Assassin’s Creed Unity, 834, 851, 905
Assimp, 716
ASTC, see under texturing, compression
asynchronous compute engine, see ACE
atan2, 8, 72
atmosphere, 595, 596, 601, 613–616, 622–623
atmospheric index, 268
B-spline, see under curves and surfaces
back buffer, see buffer, back
back plane, 93n
backface culling, see culling, backface
backprojection, 252
backward mapping, 532
baking, 451, 473, 853
least-squares, 452
balancing the pipeline, see pipeline
band-limited signal, 133
banding, 161, 1010
banding artifacts, 279, 1010
bandwidth, 1006
Bartleson-Breneman effect, 285
barycentric coordinates, 45, 46, 489, 673, 740
perspective correct, 998–1001
basis, 209
AHD, 402, 467, 471, 484, 488, 498
functions
orthogonal, 398
orthonormal, 399
hemispherical, 402–404
projection, 393
spherical, 395–402
harmonics, see spherical, harmonics
radial, 396
standard, 8, 400
tangent space, 209–210, 433, 403, 766
batch, 796
batching, 795
Battlefield 1, 601
Battlefield 4, 514
BC, see under texturing, compression
bell curve, 515
benchmarking, 1012
bent cone, screen-space, 467
bent normal, 448, 465
Bernstein
form, 835
Bézier curve, 722
Bézier patch, 737
Bézier triangle, 740
polynomial, 723, 737
Bézier triangle, 741
Bézier basis function, 723
Bézier curves, see curves, Bézier
Bézier patch, see under surfaces
Bézier triangle, see under surfaces
BGR color order, 1010
bias, 226
cone, 249
normal offset, 238, 250
receiver plane depth, 250
slope scale, 236, 249
bidirectional reflectance distribution function, see BRDF
bidirectional scattering distribution function, see BSDF
bidirectional surface scattering distribution functions, see BSSRDF
bilinear interpolation, 735
billboard, 551–564, see also impostor
axial, 559–560, 568
clouds, 563–564
particle, 567
screen-aligned, 553
spherical, 559
world-oriented, 554–559
binary space partitioning tree, see spatial data structure, BSP tree
binary tree, 820
bindless texture, 192
binormal vector, 209
biquadratic surface, 736
bitangent vector, 209, 343
blend shapes, see transform, morph targets
blending, 25
additive, 151, 527
function, 723, 729
implicit surface, 752
multi-layer alpha, 156
operations, see texturing
surfaces, see surfaces, implicit
Blinn lighting equation, 314
blocking, 809
bloom, 524, 604
blue-screening, 160
blur, 515–518
bokeh, 531, 536
Boost, 793
border, 174
Borderlands, 662, 679
bottleneck, 12, 783, 786–788, 1023
boundary representation, 581
bounded Bézier curve, see curves, bounded Bézier
bounding volume, 819, 976
creation, 948–953
hierarchy, see under spatial data structure
temporal, 821
bounding volume/object intersection, see intersection testing
bowtie, 684
box/object intersection, see specific objects under intersection testing
BRDF, 308–315
anisotropic, 314
Ashikhmin, 357
Banks, 359
Blinn-Phong, 314
clear coat, 364
cloth, 356–359
Cook-Torrance, 314
Disney diffuse, 354, 357
Disney principled, 324, 340, 345, 353, 364
Hapke model, 314
isotropic, 310
Kajiya-Kay, 359
Lambertian, 313, 314
Lommel-Seeliger model, 314
lunar, 314
Oren-Nayar, 354
Phong, 314, 340
reflectance lobe, 315, 416
specular lobe, 315, 338, 416, 418
Torrance-Sparrow, 334
Ward, 314
wave optics model, 359–363
Brütal Legend, 572
BSDF, 641–648
BSP tree, see under spatial data structure
BSSRDF, 634
buffer
A-buffer, 155
accumulation
antialiasing, 139
depth of field, 529
motion blur, 537
soft shadow, 228
back, 25, 1012
cache, 1007
color, 24, 1009–1010
compression, 1007–1009, 1032–1033
deep, 884
double, 25, 1012
dynamic, 793
framebuffer, 25
front, 25, 1012
G-buffer, 661, 884
identification, 668, 942
interleaved, 702
pending, 1013
single, 790, 1012
static, 794
stencil, 24, 53
projection shadow, 227
shadow volume, 230–233
swap, 25, 1012, 1013
triple, 1013
buffer (continued)
  velocity, 143, 540–541
  visibility, 906–908, 912
  z-buffer, 24, 53, 152, 1014–1016, 1048
  hierarchical, see under culling
bump mapping, 167, 208–214
  filtering
  CLEAN, 370
  LEAN, 370
  Tokevig, 369
  heightfield, 211–212
  normal map, 195, 211–214, 366, 710
  offset vector, 211
bus bandwidth, 1006
BV, see bounding volume
BVH, see spatial data structure, bounding volume hierarchy
C0-continuity, see continuity
C1-continuity, see continuity

cache
  hierarchy, 1038
  memory, 792
  post-transform, 700, 705
  pre-transform, 705
  texture, 1017–1018
  vertex, 700, 701, 703
cache-oblivious mesh, see mesh, cache-oblivious
CAD, 546
Call of Duty, 402, 476, 478
Call of Duty: Advanced Warfare, 286, 542, 718
Call of Duty: Black Ops, 340, 370
Call of Duty: Infinite Warfare, 325, 363–365, 420, 902
Call of Duty: WWII, 476
camera, 307–308
camera space, 15
candela, 271
canonical view volume, 16, 94
capsule, 946
cartoon rendering, see shading, toon
cathode-ray tube, see CRT
Catmull-Clark subdivision, see surfaces, subdivision, Catmull-Clark
Catmull-Rom spline, 731
caustics, 630–632
Cel Damage, 659, 660
cel rendering, see shading, toon
cell, see culling, portal
cell-based visibility, 842
CFAA, see antialiasing, custom filter
character animation, see transform, vertex
  blending
charcoal, 652
chart, 909
checkerboard rendering, 146, 930
chroma subsampling, 804
chroma-keying, 160
chromatic aberration, 521, 628, 921
chromaticity, 274
chrominance, 197
Chromium, 1020
CIE, 272, 273
CIE chromaticity diagram, 274–278
CIE XYZ, 273–276
CIECAM02, 278
CIELAB, 276
CIELUV, 276
ciliary corona, 524
  circle of confusion, 531
Civilization V, 879
clamp, 174
Claybook, 1045
CLEAN mapping, 370
ClearType, 675
clip coordinates, 18
clipmap, 867
clipping, 19, 997–998
  guard-band, 998
  plane, 19
clock gating, 1028
clock rate, 789
CLOD, see level of detail, continuous
closed model, 693
clouds, 257, 556, 563–564, 598, 613, 616–620, 622–623
clustered deferred shading, 904
clustered forward shading, 904, 907, 908, 914
clustered shading, 898–905
Cn-continuity, 728
code optimization, see optimization, code
CodeAnalyst, 792
calibration
  frame-to-frame, 866
  length, 362
  spatial, 837
temporal, 866
collision detection, 14
color, 8, 272–290
  ambient, 392
  buffer, see buffer, color
  grading, 289–290
  matching, 272–274
  mode, 1009
  deep color, 1010
  high color, 1009–1010
  true color, 1009–1010
  perception, 278
### Index

<table>
<thead>
<tr>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1159</td>
<td>color appearance model, 278</td>
</tr>
<tr>
<td>277</td>
<td>color space, 278</td>
</tr>
<tr>
<td>Adobe 1998, 277</td>
<td>ACEScg, 278</td>
</tr>
<tr>
<td>DCI-P3, 277</td>
<td>Adobe 1998, 277</td>
</tr>
<tr>
<td>IcyCp, 276, 287</td>
<td>DCI-P3, 276, 287</td>
</tr>
<tr>
<td>Rec. 2020, 277, 281</td>
<td>ICYCp, 277, 281</td>
</tr>
<tr>
<td>Rec. 709, 277, 281</td>
<td>sRGB, 277, 281</td>
</tr>
<tr>
<td>working, 278</td>
<td>color-matching functions, 272–273</td>
</tr>
<tr>
<td>colorimetry, 272–279</td>
<td>colorimetry, 272–279</td>
</tr>
<tr>
<td>command buffer, 812–814</td>
<td>command buffer, 812–814</td>
</tr>
<tr>
<td>common-shader core, 35</td>
<td>common-shader core, 35</td>
</tr>
<tr>
<td>communication, 1023</td>
<td>communication, 1023</td>
</tr>
<tr>
<td>compositing, 159</td>
<td>compositing, 159</td>
</tr>
<tr>
<td>compression</td>
<td>compression</td>
</tr>
<tr>
<td>asymmetry, 196</td>
<td>asymmetry, 196</td>
</tr>
<tr>
<td>buffer, see buffer, compression</td>
<td>buffer, see buffer, compression</td>
</tr>
<tr>
<td>texture, see texturing, compression</td>
<td>texture, see texturing, compression</td>
</tr>
<tr>
<td>vertex, see vertex, compression</td>
<td>vertex, see vertex, compression</td>
</tr>
<tr>
<td>computational photography, 549, 573</td>
<td>computational photography, 549, 573</td>
</tr>
<tr>
<td>compute unit, 1003, 1035</td>
<td>compute unit, 1003, 1035</td>
</tr>
<tr>
<td>concatenation, see transform, concatenation of</td>
<td>concatenation, see transform, concatenation of</td>
</tr>
<tr>
<td>cone tracing, 455, 467, 584</td>
<td>cone tracing, 455, 467, 584</td>
</tr>
<tr>
<td>voxel, 495, 504</td>
<td>voxel, 495, 504</td>
</tr>
<tr>
<td>conservation of energy, 312</td>
<td>conservation of energy, 312</td>
</tr>
<tr>
<td>conservative depth, 1016</td>
<td>conservative depth, 1016</td>
</tr>
<tr>
<td>conservative rasterization, see rasterization, conservative</td>
<td>conservative rasterization, see rasterization, conservative</td>
</tr>
<tr>
<td>constructive interference, see light interference, constructive</td>
<td>constructive interference, see light interference, constructive</td>
</tr>
<tr>
<td>constructive solid geometry, 750</td>
<td>constructive solid geometry, 750</td>
</tr>
<tr>
<td>continuity, see also curves and surfaces</td>
<td>continuity, see also curves and surfaces</td>
</tr>
<tr>
<td>C⁰, 728, 741, 745</td>
<td>C⁰, 728, 741, 745</td>
</tr>
<tr>
<td>C¹, 728, 742</td>
<td>C¹, 728, 742</td>
</tr>
<tr>
<td>Cⁿ, 728</td>
<td>Cⁿ, 728</td>
</tr>
<tr>
<td>G¹, 728, 742</td>
<td>G¹, 728, 742</td>
</tr>
<tr>
<td>Gⁿ, 728</td>
<td>Gⁿ, 728</td>
</tr>
<tr>
<td>continuous signal, 131</td>
<td>continuous signal, 131</td>
</tr>
<tr>
<td>contour, 686</td>
<td>contour, 686</td>
</tr>
<tr>
<td>edge detection, 665–669</td>
<td>edge detection, 665–669</td>
</tr>
<tr>
<td>halo, 659</td>
<td>halo, 659</td>
</tr>
<tr>
<td>image, 660–665</td>
<td>image, 660–665</td>
</tr>
<tr>
<td>line, 655</td>
<td>line, 655</td>
</tr>
<tr>
<td>loop, 667</td>
<td>loop, 667</td>
</tr>
<tr>
<td>procedural geometry, 657–660</td>
<td>procedural geometry, 657–660</td>
</tr>
<tr>
<td>shading normal, 656–657</td>
<td>shading normal, 656–657</td>
</tr>
<tr>
<td>shell, 658–659</td>
<td>shell, 658–659</td>
</tr>
<tr>
<td>contouring artifacts, see banding artifacts</td>
<td>contouring artifacts, see banding artifacts</td>
</tr>
<tr>
<td>control cage, 756</td>
<td>control cage, 756</td>
</tr>
<tr>
<td>control mesh, 756</td>
<td>control mesh, 756</td>
</tr>
<tr>
<td>control points, 720</td>
<td>control points, 720</td>
</tr>
<tr>
<td>control polygon, 754</td>
<td>control polygon, 754</td>
</tr>
<tr>
<td>convex hull, 950</td>
<td>convex hull, 950</td>
</tr>
<tr>
<td>Bézier curve, 723</td>
<td>Bézier curve, 723</td>
</tr>
<tr>
<td>Bézier patch, 738</td>
<td>Bézier patch, 738</td>
</tr>
<tr>
<td>Bézier triangle, 741</td>
<td>Bézier triangle, 741</td>
</tr>
<tr>
<td>Loop, 761</td>
<td>Loop, 761</td>
</tr>
<tr>
<td>convex partitioning, 684</td>
<td>convex partitioning, 684</td>
</tr>
<tr>
<td>convex polyhedron, 946, 950</td>
<td>convex polyhedron, 946, 950</td>
</tr>
<tr>
<td>convex region, 685</td>
<td>convex region, 685</td>
</tr>
<tr>
<td>convolution, 135</td>
<td>convolution, 135</td>
</tr>
<tr>
<td>Cook-Torrance model, 314</td>
<td>Cook-Torrance model, 314</td>
</tr>
<tr>
<td>cookie, 221, 230, 434</td>
<td>cookie, 221, 230, 434</td>
</tr>
<tr>
<td>coordinate system</td>
<td>coordinate system</td>
</tr>
<tr>
<td>left-handed, 92, 95</td>
<td>left-handed, 92, 95</td>
</tr>
<tr>
<td>right-handed, 92</td>
<td>right-handed, 92</td>
</tr>
<tr>
<td>corner cutting, 753</td>
<td>corner cutting, 753</td>
</tr>
<tr>
<td>counterclockwise vertex order, 63, 692</td>
<td>counterclockwise vertex order, 63, 692</td>
</tr>
<tr>
<td>coverage, 995</td>
<td>coverage, 995</td>
</tr>
<tr>
<td>coverage mask, A-buffer, 155</td>
<td>coverage mask, A-buffer, 155</td>
</tr>
<tr>
<td>CPU-limited, 786</td>
<td>CPU-limited, 786</td>
</tr>
<tr>
<td>cracking, 689, 769</td>
<td>cracking, 689, 769</td>
</tr>
<tr>
<td>Bézier triangle, 747</td>
<td>Bézier triangle, 747</td>
</tr>
<tr>
<td>fractional tessellation, 769</td>
<td>fractional tessellation, 769</td>
</tr>
<tr>
<td>polygon edge, 689, 714</td>
<td>polygon edge, 689, 714</td>
</tr>
<tr>
<td>quadtree, 774</td>
<td>quadtree, 774</td>
</tr>
<tr>
<td>tessellation, 771</td>
<td>tessellation, 771</td>
</tr>
<tr>
<td>crawlies, 130</td>
<td>crawlies, 130</td>
</tr>
<tr>
<td>crease, 763</td>
<td>crease, 763</td>
</tr>
<tr>
<td>critical angle, 326</td>
<td>critical angle, 326</td>
</tr>
<tr>
<td>cross product, 7</td>
<td>cross product, 7</td>
</tr>
<tr>
<td>CrossFire X, 1013</td>
<td>CrossFire X, 1013</td>
</tr>
<tr>
<td>CRT, 161</td>
<td>CRT, 161</td>
</tr>
<tr>
<td>Crysis, 220, 457, 458, 559</td>
<td>Crysis, 220, 457, 458, 559</td>
</tr>
<tr>
<td>Crysis 3, 631</td>
<td>Crysis 3, 631</td>
</tr>
<tr>
<td>CSA, see antialiasing, coverage sampling</td>
<td>CSA, see antialiasing, coverage sampling</td>
</tr>
<tr>
<td>CSG, 750</td>
<td>CSG, 750</td>
</tr>
<tr>
<td>CSM, see shadow, map, cascaded</td>
<td>CSM, see shadow, map, cascaded</td>
</tr>
<tr>
<td>cube map, 173, 190</td>
<td>cube map, 173, 190</td>
</tr>
<tr>
<td>cube mapping, see environment mapping, cubic</td>
<td>cube mapping, see environment mapping, cubic</td>
</tr>
<tr>
<td>cube texture, 190</td>
<td>cube texture, 190</td>
</tr>
<tr>
<td>CubeMapGen, 415</td>
<td>CubeMapGen, 415</td>
</tr>
<tr>
<td>cubic convolution, 178</td>
<td>cubic convolution, 178</td>
</tr>
<tr>
<td>cubic curve, see curves, cubic</td>
<td>cubic curve, see curves, cubic</td>
</tr>
<tr>
<td>cuculoris, 434</td>
<td>cuculoris, 434</td>
</tr>
<tr>
<td>CUDA, 54, 1040</td>
<td>CUDA, 54, 1040</td>
</tr>
<tr>
<td>culling, 830–851</td>
<td>culling, 830–851</td>
</tr>
<tr>
<td>backface, 800, 831–835</td>
<td>backface, 800, 831–835</td>
</tr>
<tr>
<td>orientation consistency, 63</td>
<td>orientation consistency, 63</td>
</tr>
<tr>
<td>clustered backface, 833–835</td>
<td>clustered backface, 833–835</td>
</tr>
<tr>
<td>detail, 839–840</td>
<td>detail, 839–840</td>
</tr>
</tbody>
</table>
Index

Culling (continued)
culling, 53, 801, 849, 851, 1016
frontface, 832
dependent texture read, see texture, dependent
depth
buffer, see buffer, z-buffer
reversed, 100
complexity, 801, 841
peeling, 152, 154–155, 252, 625, 893
sprite, see under impostor
deferred lighting, 892
deferred shading, 883–890, 1022, 1028
deferred texturing, 905–908
denoising, 519
depth of field, 523, 525, 527–536
Derivatives, see gradient of pixel
destiny, 129, 130, 453, 815
destiny 2, 571, 572, 1041
destiny: the taken king, 128
destructive interference, see light, interference,
destructive
determinant of a matrix, 7
dead end, 586, 829
decimal differential analyzer, 506
digital visual interface, 1011
diffracted color, 314, 348
diffuse term, 306
diffusion, 634
depth
buffer, see buffer, z-buffer
reversed, 100
dependent texture read, see texture, dependent
depth of field, 523, 525, 527–536
diffraction, 303, 360–361
diffuse color, 314, 348
diffuse term, 306
diffusion, 634
depth
buffer, see buffer, z-buffer
reversed, 100
complexity, 801, 841
peeling, 152, 154–155, 252, 625, 893
sprite, see under impostor
depth of field, 523, 525, 527–536, 835
derivative, see gradient of pixel
Destiny, 129, 130, 453, 815
Destiny 2, 571, 572, 1041
Destiny: The Taken King, 128
destructive interference, see light, interference,
destructive
determinant of a matrix, 7
Deus Ex: Mankind Divided, 908
dielectric, 321
difference of Gaussians, 665
diffraction, 303, 360–361
diffuse color, 314, 348
diffuse term, 306
diffusion, 634
depth
buffer, see buffer, z-buffer
reversed, 100
dependent texture read, see texture, dependent
depth of field, 523, 525, 527–536
diffraction, 303, 360–361
diffuse color, 314, 348
diffuse term, 306
diffusion, 634
depth
buffer, see buffer, z-buffer
reversed, 100
dependent texture read, see texture, dependent
depth of field, 523, 525, 527–536
diffraction, 303, 360–361
diffuse color, 314, 348
diffuse term, 306
diffusion, 634
normal-map, 635
screen-space, 636–638
texture-space, 635
digital differential analyzer, 506
digital visual interface, 1011
dihedral angle, 654, 660, 695
dimension reduction, 955
direct memory access, see DMA
Direct3D, 21n
DirectCompute, 40
directed acyclic graph, 586, 829
direction, principal, 672
directional occlusion, 465
dynamic, 467–468
precomputed, 466
shading, 468–472
DirectX, 38–41
DirectX 11, 813
DirectX 12, 814
discrete geometry LOD, see level of detail,
discrete geometry
discrete oriented methods, 493
discrete oriented polytope, see k-DOP
discretized signal, 131
Disney Infinity 3.0, 372
Displaced subdivision, see surfaces, subdivision,
displaced
displacement mapping, 167, 219, 765, 770

display
  encoding, 160–165
  engine, 1011
  flare, 285
  head-mounted, 916
  interface, 1011
  list, 812
  primary, 276
  varifocal, 923
display rate, 1
display-referred, 283
DisplayPort, 1011
distance field, 677
distortion, lens, 921
distribution of normals, see NDF
distribution of visible normals, 333
dithering, 1010
DLAA, see antialiasing, directionally localized
DNA, 1034
Dolby Vision, 282
DOM, see discrete ordinate methods
domain, 719
  rectangular, 736
  triangular, 740
domain shader, 44
DOOM (2016), 246, 540, 629, 823, 869, 883, 901
dot product, 7
dots per inch, 817
double buffer, see buffer, double
downsampling, 136, 518, 525
DRAM, 791
draw call, 35
Dreams, 577, 1045
driver, see graphics driver
dual paraboloid mapping, see environment
  mapping, parabolic
dual source-color blending, 53
dueling frusta, 242
Dust 514, 493
DYI, 1011
DXR, 1044
DXTC, see under texturing, compression
dynamic buffer, 793
dynamic super resolution, 139

EAC, 194
ear clipping, 685
early-z culling, see under culling edge, 654–656, see also line
  border, 654
  boundary, 654, 661, 692, 709
  bridge, 686
collapse, see under simplification
  contour, 654–656
  crease, 654, 695, 709, 747
  detection, 661, 663
  feature, 654
  function, 994–996
  hard, 654
  join, 686
  keyholed, 686
  material, 654
  preservation, 695
  ridge, 654, 660
  silhouette, 654–655
  stitching, 689
  suggestive contour, 655
  valley, 654, 660
effective surface, 350
electrical optical transfer function, 161, 283
EM, see environment mapping
energy efficiency, 1024
Enlighten, 482
enveloping, see transform, vertex blending
environment mapping, 404–433
  cubic, 410–412, 425
  irradiance, 424–433
  latitude-longitude, 406–408
  localized, 499–502
  octahedral, 413
  parabolic, 413
  prefilted, 415–420, 471, 502, 503
  sphere, 408–410
EOTF, see electrical optical transfer function
EQAA, see antialiasing, enhanced quality
Ericsson texture compression, see texturing,
  compression, ETC
ESM, see shadow, map, exponential
ETC, see under texturing, compression
Euler angles, 59, 70, 73, 82
Euler transform, see transform, Euler
Euler-Mascheroni constant, 802
Euler-Poincaré formula, 699, 706
EVSM, see exact visible set
EVS, see exact visible set
EWS, see shadow, map, exponential
EWA, 189
exact visible set, 831
execution unit, 1003
exitance, 442, 474
explicit surface, see surfaces, explicit
exposure, 285, 288–289
extraordinary vertex, 758
eye space, 15
eye-dome lighting, 575
  faceter, 682
  fairness, 761
falloff function, 114, 381
fan, see triangle, fan
Far Cry, 453, 476
Far Cry 3, 478, 481
Far Cry 4, 420, 481
far plane, 93, 99, 981
Feline, 189
fence, 938

FIFA, 616
FIFO, 808, 809, 1023
fill rate, 788
film frame rate, 536
filter, 130–137
bilateral, 462, 518–520
box, 134, 165, 517, 518
bright-pass, 527
cross bilateral, see filter, joint bilateral
disk, 518
de edge-preserving, 520
Gaussian, 136, 189, 515, 517, 572, 665
joint bilateral, 249, 519
kernel, 517
low-pass, 135, 136
nearest neighbor, 134
rotation-invariant, 515
running-average, 523
separable, 516, 517, 520, 532
sinc, 135–136, 515
steerable, 525
support, 517
tent, 134
triangle, 134
fin, 646, 668
Final Fantasy XV, 620
fireflies, see under aliasing
Firewatch, 104
first principal direction, 672
fixed-function pipeline, 27
fixed-view effects, 546–547
flat shading, 120
FLIPQUAD, 146
floor, 769
gain control, 36
dynamic, 36
static, 36
flush, 1005
FMA, 1026, 1033
log, 598, 600–602, 608
force feedback, 14
form factor, 442
forward mapping, 531
forward shading, 883
forward+ shading, see tiled, forward shading
Forza Horizon 2, 141, 899
Forza Motorsport 7, 2, 412
foveated rendering, 931–932
FPS, 1, 13, 789, 817
fragment, 22, 49
fragment shader, 23, 49, 125, see also pixel shader
frame rate, 1, 808
costant, 865
frame-to-frame coherence, 866
frame buffer, 25
frames per second, 13
FreeSync, 1011
FreeType, 676
Fresnel effect, 319
Fresnel equations, 316
Fresnel reflectance, 316–327, 330, 331, 348, 351, 405, 420, 421, 426, 498, 626, 631, 632, 643, 892
Schlick approximation, 320, 321, 326, 347, 351, 598
front buffer, see buffer, front
front plane, 93n
Frostbite game engine, 111, 113, 115, 116, 287, 290, 312, 325, 616, 804, 811, 851, 878, 890, 893, 903
froxtel, 611
frustum, 11, 17–18, 981
plane extraction, 983–984
tracing, 261
frustum/object intersection, see specific objects under intersection testing
FSAA, see antialiasing, full-scene
full screen pass, 514
fur, 640–641, 646–649
FX Composer, 44
FXAA, see antialiasing, fast approximate
G-sync, 1011
G1-continuity, see continuity
gamma correction, 160–165, 184
gamut, 276, 323
sRGB, 323
gas, ideal, 297
gather operation, 532
Gauss map, 667
Gaussian, anisotropic spherical, 398, 498
GBAA, see antialiasing, geometry buffer
GCN, see under hardware
genus, 699
goodesic curve, 81
geometric mean, 864
geometry
clipmap, 872–873
patch, 775
Index

Index

processing, see under pipeline shader, 18–19, 47–48, 647, 668, 677, 702, 786, 798
stage, see pipeline, geometry processing geomorph LOD, see level of detail, geomorph GigaThread engine, 1032
 Gimbal lock, 73
 glare effects, 524
 global illumination, 315, 438
 glPolygonOffset, 236, 657, 673
 GLSL, 35, 39
 gluLookAt, 67
 gluPerspective, 99
 G-n-continuity, 728
 gobo, 173, 221, 230, 434
 golden thread, 547
 Gooch shading, 103, 663
 Gouraud shading, 118
 GPA, 785
 GPU, 13, 29, see also hardware computing, 54
 GPU Boost, 789
 GPU PerfStudio, 785
 GPUView, 785
 gradient of pixel, 51, 185
 graffiti, 672
 Grand Theft Auto V, 525
 graphics driver, 786, 793, 1012
 graphics processing unit, see GPU grayscale, conversion to, 278
 great arc, 81
 great circle, 81
 green-screening, 160
 GRID2, 258
 GTX 1080, see under hardware guard-band clipping, 998
 H-basis, 404
 hair, 257, 640–646, 649
 half vector, 336
 half-edge, 692
 Half-Life 2, 402, 403, 476, 499
 Half-Life 2 basis, 403
 half-space, 6, 946
 halo, 524
 Halo 3, 475
 haloing, 675
 Halton sequence, 144
 hard real time, 865
 hardware GameCube, 867
 GCN, 1035–1039
 GeForce 256, 29
 GeForce3, 38
 GTX 1080, 1029–1035
 Mali architecture, 1020, 1024–1029
 NVIDIA Pascal, 1029–1035
 Pixel-Planes, 8n, 1026
 PixelFlow, 1022
 PLAYSTATION, 936
 PLAYSTATION 3, 39, 700
 PLAYSTATION 4, 867, 1007, 1035
 Pomegranate, 1022
 Talisman, 189, 551
 Vega, 1035–1039
 Voodoo 1, 1
 Wii, 27, 39
 Xbox, 1035
 Xbox 360, 39
 Xbox One, 867
 harmonic series, 802
Hausdorff distance, 708, 875
 H-basis, 475
 HBM2, 1034, 1038
 HDMI, 1011
 HDR, 193, 271, 281–283, 405
 display, 1011
 HDR10, 281
 head, 70, 72
 heads-up display, 561, 917, 932, 933
 heat diffusion, 535
 heightfield, 564–566, see also bump mapping terrain, 877
 Hellgate: London, 609
 Helmholtz reciprocity, 312, 351
 hemispherical basis, see basis, hemispherical hemispherical harmonics, 404
 Henyect-Greenstein phase function, 598–599, 620
 Hermite curves, see curves, Hermite Hermite interpolation, see interpolation, Hermite Hertz, 13
 hidden line removal, 668–669
 hidden line rendering, see line, hidden hierarchical image caching, see impostor hierarchical spatial data structure, see spatial data structure hierarchical view frustum culling, see culling, hierarchical view frustum hierarchical z-buffering, see under culling high color mode, see color, mode, high color high dynamic range, see HDR high-definition multimedia interface, 1011
 High-Level Shading Language, see HLSL highlight, 119
 highlight selection, 673
 histogram, 245
Index

histogram renormalization, 196
hither, 93n
HiZ, 252, 1015, 1038
HLG, 281
HLSL, 35, 39
homogeneous notation, 6, 58, 62, 173
homogenization, 62, 92
horizon angle, 460
horizon mapping, 460, 466
hourglass, 684
HRAA, see antialiasing, hybrid reconstruction
HTC Vive, see Vive
HTILE, 1038
HUD, 561, 917, 932, 933
hue, 276
hull shader, 44
Hunt effect, 285
Huygens-Fresnel principle, 360
Hybrid Log-Gamma, see HLG
hysteresis, 861
HZB culling, see culling, hierarchical z-buffering
IBR, see image-based rendering
illuminant D65, 270, 274
image
gometry, 566, 876
processing, 513–522, 665
pyramid, 846, 847
state, 283
image-based lighting, 406, 414–424, 435
image-based rendering, 269, 545
immediate context, 813
implicit surface, see surfaces, implicit
importance sampling, 385, 445, 451, 503
impostor, 561–564, 866
depth sprite, 564–565
layered depth image, 565
index buffer, 702–705
index of refraction, 298
complex, 298
indirect draw command, 851
inFAMOUS Second Son, 91, 572
inflection, 728
inner product, 398
input assembler, 42
inside test, 996
instance, 15, 829
instanting, 42, 797
instruction set architecture, 35
instruction-level parallelism, 1003
Instruments, 785, 792
integral
double product, 464, 470
triple product, 470
intensity, 269
interactivity, 1
interface, see hardware
interference, see under light
interleaved sampling, 145
intermediate language, 35
interpolation, 781, 998–1001
barycentric, 963
bicubic, 178
bilinear, 178–180, 182, 735–736
centroid, 141
Hermite, 729–732
linear, 720
perspective-correct, 22, 49, 1000
quadrilinear, 189
repeated, 740
bilinear, 736
linear, 720–722
trilinear, 186
interpupillary distance, 923
intersection testing, 941–991
AABB/AABB, 978–979
box/plane, 970–972
box/ray, 959–962
ray slope, 961–962
slabs method, 959–961
BV/BV, 976–981
convex polyhedron/ray, 961
crossings test, 967–970
dimension reduction, 955
frustum, 981–987
frustum/box, 980–987
frustum/ray, 961
frustum/sphere, 984–986
hardware-accelerated, 942–943
interval overlap method, 972–974
k-DOP/k-DOP, 979–980
k-DOP/ray, 961
line/line, 987–990
OBB/OBB, 980–981
picking, 942
plane/box, 970–972
plane/ray, 966
plane/sphere, 970
polygon/ray, 966–970
polyhedron/polyhedron, 987
ray/box, 959–961
rejection test, 948
rules of thumb, 954–955
separating axis, 946
separating axis test, 947, 974, 979, 980, 986–987
sphere/box, 977–978
sphere/ray, 955–959
sphere/sphere, 976–977
three planes, 990
triangle/box, 974–975
triangle/ray, 962–966
triangle/triangle, 972–974
interval overlap method, see under intersection testing
intrinsic functions, 36
inverse displacement mapping, see texturing, parallax occlusion mapping
inverse z, 100
IOR, see index of refraction
irradiance, 268, 294, 425
precomputed, 474
spherical harmonics, 475
volume, 487
irradiance mapping, see environment mapping, irradiance
irregular vertex, 758
isocube, 412
isosurface, 584, 682, 753
isotropic scaling, 62
J<sub>aba</sub>b<sub>b</sub>, 276
jaggies, see under aliasing
jittering, see under antialiasing
joint, 720, 728, 731
Jordan curve theorem, 967
judder, 935
Just Cause 2, 114, 882
Just Cause 3, 883, 899, 900
k-ary tree, 820
k-d tree, see under spatial data structure
k-DOP, 945–946, 961, 970, 990
creation, 949
Kentucky Route Zero, 121
Killzone: Shadow Fall, 116, 523
Killzone 2, 885
Kite, 493
Kochanek-Bartels curves, see curves, Kochanek-Bartels
LAB, 276
Lambertian shading, see BRDF, Lambertian
The Last of Us, 476
late depth test, 1016
late latching, 938
latency, 1, 30, 791, 807–810, 920–921, 935, 1004–1006, 1013
occlusion query, 845
Latin hypercube sampling, 143
latitude, 407, 944
layered depth image, see under impostor
LCD, 676
LDI, see impostor, layered depth image
LEAN mapping, 370
left-handed, see under coordinate system
lens flare, 524–526
level of detail, 44, 580, 706, 717, 807, 852–866
alpha, 857–858
bias, 186, see also texturing
blend, 856
continuous, 706, 859, 860
discrete geometry, 854–856
fractional tessellation, 768
generation, 853
gemorph, 859–860
hysteresis, 861
PN triangle, 747
popping, 710, 854, 856, 858
projected area-based, 861–864
range-based, 860–861
selection, 853, 860–864
simplification, 710
subdivision surface, 756
switching, 853, 854–860
time-critical, 865–866
level set, 583
LIDAR, 573
light
ambient, 391–392
attenuation mask, 230
baking, 798
bandwidth, 362
bleeding, 255
field, 269
interference
constructive, 296, 298
destructive, 296
thin-film, 361–363
inverse-square attenuation, 111
leak, 238, 255, 256
map, 484
meter, 271
monochromatic, 293
polarized, linearly, 293
polychromatic, 293
prepass, 892
probe, 414, 490, 901
propagation volumes, 493
cascaded, 494
scattering, see scattering
shafts, 602, 604, 608, 631
source, 106–117, 798
area, 116–117, 224, 228, 377–391
card, 387, 388, 427
directional, 109–110
light (continued)
source (continued)
disk, 379, 381, 388, 430, 435
fill, 431
omni, see light, source, point
point, 111–114
polygonal, 389
punctual, 110–116
spherical, 381–384, 386, 387, 430
spot, 114–115
tube, 387
volume, 224
transport
linearity, 438, 479
meshless, 484
modular, 484
notation, 439–440
unpolarized, 294
velocity, phase, 294
visible, 268
light map, 227
light-field rendering, 549
lightcuts, 431
lighting probe, 490
limit
curve, 754, 756
surface, 760
line, 19, 673–675, see also edge
haloing, 675
hidden, 674–675
integral convolution, 538
triangle edge rendering, 673–674
line/line intersection, see intersection testing,
line/line
linear blend skinning, 84
linear interpolation, 720
linear speedup, 810
linear transform, see transform, linear
linearly transformed cosines, 390
LiSPSM, see shadow, map, light space
perspective
LittleBigPlanet, 488
load balance, 1023
lobe
anisotropic, 422–424
asymmetric, 422–424
local frame, 343
local illumination, 315
local lighting model, 438
LOD, see level of detail
log, 8
longitude, 407, 944
lookup table, 173
loop, 686
Loop subdivision, see surfaces, subdivision,
Loop
loose octree, see under spatial data structure
lossy compression, 194
Lost Planet, 647
lozenge, 946
LPV, see light, propagation volumes
Lumberyard, 740, 1044
Lumigraph, 549
luminance, 197, 271, 273, 278
LUT, see lookup table
LUV, 276
Möbius strips, 693
Mach banding, 1010
macroscale, 208, 367
magnification, see under texturing
main axes, 8
Mali, see hardware, Mali architecture
manifold, 694
Mantle, 40
marching cubes, 583, 683, 753
marching tetrahedra, 753
mask, 759
masked hierarchical depth buffer, 849
masking
function, 333
perceptual, 278
masking-shadowing function, 334, 335
material, 125
glossy, 382–386
instance, 126
template, 126
matrix, see also transform
adjoint, 68
change of basis, 63, 67, 75
column-major, 60
determinant, 63
orientation, 60, 70
orthogonal, 69, 72, 80
rotation, 70
row-major, 60, 95
trace, 61, 80
transpose, 7, 63
matte, 159
mean width, 954
media, 310
mediated reality, 917
medium
absorptive, 298
homogeneous, 298
megatexture, 867
memory
allocation, 793
architecture, 1006–1007
Index

bandwidth, 1006
dynamic random access, 791
controller, 1038
hierarchy, 791
optimization, see optimization, memory
UMA, 1007
wall, 791
merging of pixels, 24–25
merging stage, 24–25, 53
mesh
cache-oblivious, 700–701
parameterization, 173
segmentation, 683
smoothing, 694–696
solidity, 693–694
triangle, 691, 699–701
universal, 700–701
Meshlab, 695, 716
mesoscale, 208–209, 367
message-passing architecture, see multiprocessing, message-passing
metaball, 48, 683, 751
Metal, 40, 814
texturing, see
morph targets, see under transform
morphing, see under transform
Morton sequence, 1018
mosaicing, see texturing, tiling
motion blur, 536–542, 835
MPEG-4, 712
MRT, 50
MSAA, see antialiasing, multisampling
multi-view, 928
multicore, 806
multiprocessing, 805–814, 1023
dynamic assignment, 810
message-passing, 806
parallel, 809–810
pipeline, 806–809
static assignment, 810
symmetric, 806
task, 811–812
task-based, 806
multiprocessor, 1003
shared memory, 806, 1003
streaming, see streaming, multiprocessor
multisampling, see under antialiasing
multitexturing, see texturing
multum in parvo, 183
N-patch, see surfaces, PN triangle
N-rooks sampling, 143
nailboard, see impostor, depth sprite
nanogeometry, 359
NDF, 332, 337–346, 367, 498
anisotropic, 343–346
Beckmann, 338
Blinn-Phong, 339, 340
filtering, 367–372
generalized Trowbridge-Reitz, see NDF, GTR
GGX, 340–342, 369
GTR, 342
isotropic, 338–343
shape-invariance, 339
trowbridge-Reitz, see NDF, GGX
near plane, 93, 99, 862, 981
nearest neighbor, see under filter and texturing, magnification and texturing, minification
Need for Speed, 616
Newell’s formula, 685
node, 819–821
internal, 819
leaf, 819
root, 819
node hierarchy, 828
noise, 872
noise function, see texturing, noise
non-photorealistic rendering, 651–673
noncommutativity, 65, 77
normal
  cone, 833–835
  incidence, 317
  map, see under bump mapping
  transform, see transform, normal
normal distribution function, see NDF
normal-masking independence, 334
normalized device coordinates, 19, 94, 98, 100
NPR, see non-photorealistic rendering
NSight, 785
NURBS, 781
NVIDIA Pascal, see under hardware
Nyquist limit, see under sampling
OBB, 945, 946, 976
OBB/object intersection, see specific objects under intersection testing
object-based shading, 908–912
obscurance, 449, 450, 454, 457
  volumetric, 459
occluder, 844
occluding power, 844
occlusion culling, see culling, occlusion
occupancy, 32, 127, 801, 886, 898, 1005
occupancy function, 459
octahedral mapping, 413
octave, 198
octree, see under spatial data structure
octree texture, 190
Oculus Rift, 915, 916, 923, 935
OETF, see optical electric transfer function
Okami, 653
opacity, 149
Open3DGC, 712
OpenCL, 54
OpenCTM, 712
OpenGL, 39–41
  extensions, 40
  ES, 41, 194
  Shading Language, 35
OpenSubdiv, 777–779
optical electric transfer function, 161
optics
  geometrical, 303
  physical, 359
  wave, 359
optimization
  application stage, 790–793
  code, 790–793
  geometry processing, 798–800
  lighting, 798–800
  memory, 791–793
  merging, 805
  mobile, 814
  pipeline, 783–815
  pixel processing, 800–804
  pixel shader, 803
  rasterization, 800
The Orange Box, 288
The Order: 1886, 91, 357, 365, 370, 477, 498, 896
ordinary vertex, 758
Oren and Nayar model, 354
orientation, see under polygon
oriented bounding box, see OBB
orienting the camera, 67
over operator, 150–151
overblurring, 186
overclock, 787
overdraw, see under pixel
packed pixel format, 1010
padding, 792
painter’s algorithm, 551, 824
painterly rendering, 652
pan, 538
parabola, 721
parabolic mapping, 413
parallax, 548
  mapping, 167, 214–220
  occlusion mapping, see under texturing
parallel
  architectures, 1020
  graphics, 1019
  processing, see multiprocessing, parallel
  projection, see projection, orthographic
  parallelism, 810
parametric curves, see curves, parametric
parametric surfaces, see surfaces, parametric
participating media, 310
  absorption, 590
  extinction, 590, 593, 595, 610, 616, 624, 639, 643
  optical depth, 593, 595
  phase function, 590, 623, 626, 638, 644
  geometric, see scattering, geometric
  Mie, see scattering, Mie
  Rayleigh, see scattering, Rayleigh
particle
  soft, 558–559
  system, 567–572
Pascal, see hardware, NVIDIA Pascal
patch, 736
path tracing, 26, 444, 510, 1043, 1044
Index

PCF, see percentage-closer filtering
PCI Express, 1006
Pearl Harbor, 446
pen and ink, 652
pending buffer, 1013
penumbra, see under shadow
per-triangle operations, 14
per-vertex operations, 14
percentage-closer filtering, 247–250, 849
perceptual quantizer, see PQ
performance measurement, 788–790
perp dot product, 6, 987, 989
persistence, 935
perspective division, 19
projection, see projection, perspective warping, 241
perspective-correct interpolation, see interpolation
Peter Panning, 238
Phong lighting equation, see BRDF, Phong
Phong shading, 118
Phong tessellation, see under surfaces
photogrammetry, 573, 682
photometric curve, 271, 273, 278
photometry, 271
photopic, 271
photorealistic rendering, 545, 651
PhyreEngine, 893
pick window, see intersection testing, picking
picking, 942, 943, 957
piecewise Bézier curves, see curves, piecewise
pixelation, 178
pixels per inch, 817
pixels per second, 788
plane, 6
- axis-aligned, 8
- coordinate, 8
plane masking, 836
plane/object intersection, see specific objects under intersection testing
PLAYSTATION, see under hardware
point cloud, 572–578, 683
point rendering, 572–578
point-based visibility, 842
pointer indirection, 792
Poisson disk, 249
Pokémon GO, 917
polycube maps, 171
polygon
- bowtie, 684
- consolidation, 691
- contour, 686
- convex, 685
- edge cracking, see cracking, polygon edge
- edge stitching, 689
- hourglass, 684
- loop, 686
- merging, 691
- mesh, 691
- orientation, 691–693
- sorting, 824
- soup, 691
- star-shaped, 686
- T-vertex, 689–690
polygon-aligned BSP tree, see spatial data structure, BSP tree
polyhedral techniques, 853
polygonalization, 583, 683
polymorph engine, 1031
polyphostor, 562
POM, 217
popping, see under level of detail
port, 1006
portal culling, see culling, portal
pose, 921, 924, 938
post-processing, 514
posterization, 652, 1010
potentially visible set, 831
power form, 724
power gating, 1028
PowerTune, 789
PowerVR, 196
PQ, 281
pre-lighting, 892
pre-order traversal, 835
precision, 712–715
  color, 186, 1010
  depth, 236
  floating point, 713
  mobile, 814
  subpixel, 689
precomputed radiance transfer, 471, 478, 479, 481
  local deformable, 481
predictive rendering, 280
prefilter, 414
primitive generator, 44
primitive shader, 1037
Prince of Persia, 658
principal component analysis, 480, 484
probability, geometric, 953–954
procedural modeling, 222, 672, 682
procedural texturing, see texturing, procedural
  processor
pixel, see pixel, shader
vertex, see vertex, shader
progressive refinement, 510, 547
projection, 16–18, 92–102
  3D polygon to 2D, 966
  3D triangle to 2D, 962
  bounding volume, 861–864
  cylindrical, 172
  orthographic, 17–18, 59, 93–95
  parallel, see projection, orthographic perspective, 17, 59, 96–102, 1014
  planar, 172
  spherical, 172
projective texturing, see texturing, projective
proxy object, 819
PRT, see precomputed radiance transfer
PSM, see shadow, map, perspective
Ptex, 191
purple fringing, 628
purple line, 274
PVRTC, see under texturing, compression
PVS, see potentially visible set
PxrSurface, 343, 359, 363, 364
QEM, 708
quad, 51, 801, 994
quad overshading, 787, 853, 863, 910, 994
quadratic curve, see curves, quadratic
quadratic equation, 957
quadric error metric, 708
quadtree, see under spatial data structure
Quake, 37, 474
Quake II, 474
Quake III, 37, 496
quantization, scalar, 714
Quantum Break, 496
quartic curve, see curves, quartic
quaternion, 72, 76–84
  addition, 77
  conjugate, 77, 78
  definition, 76
  dual, 87
  identity, 77
  imaginary units, 76
  inverse, 77
  laws of multiplication, 78
  logarithm, 78
  matrix conversion, 79–81
  multiplication, 77
  norm, 77, 78
  power, 78
  spherical, 81–83
  spherical linear interpolation, 81–82
  spline interpolation, 82–83
  transforms, 79–84
  unit, 78, 79
Quidditch, 950
Quincunx, see antialiasing, Quincunx
quintic curve, 181
radiance, 269–270, 273, 425
  distribution, 269
  incoming, 315
radiant
  exitance, see exitance
  flux, 268
  intensity, 269
radiometry, 267
radiosity, 442–443
  normal mapping, 402–404
  progressive, 483
RAGE, 867
Rainbow Six Siege, 887
range-based fog, see fog
raster engine, 1031
raster operation, see ROP
rasterization, see under pipeline
  conservative, 22, 139, 259, 582, 1001
    inner, 1001
    outer, 1001
    overestimated, 1001
    underestimated, 1001
rasterizer order view, 52, 139, 156
rasterizer stage, see pipeline, rasterization
Ratatouille, 638
rational linear interpolation, 720
Index

ray, 943–944
  casting, 443
  function, 437
marching, 199, 216–220, 262, 566, 570, 594, 607, 608, 614, 616, 618, 620–622, 639, 642, 648, 752, 753, 1048
tracing, 26, 259, 261, 443–445, 530, 586, 802, 953, 1006, 1044–1047
architecture, 1039
isosurface, 584
voxel, 580
ray/object intersection, see specific objects under intersection testing
Rayleigh scattering, see scattering, Rayleigh reciprocity, 312
reconstruction, 131, 133–136
reduce, 245, 896
reflectance
  anisotropic, 328
directional-hemispherical, 313
equation, 311, 437
hemispherical-directional, 313
isotropic, 328
spectral, 279
reflectance lobe, see under BRDF
reflection, 314, 315, 623, 626, 630
environment mapping, 413
equation, see reflectance, equation
external, 317
internal, 317, 325
total, 326
law of, 504
mapping, 405
planar, 504–505, 839
probe, 499
  localized, 500
proxy, 500
screen-space, 505–509
transform, see transform, reflection
refraction, 149, 302, 626–630, 631–633, 638, 639
image-space, 630
refractive index, 298
refresh rate, 1
vertical, 1011
register combiners, 38
register pressure, 127, 801, 904, 1005
regular vertex, 758
relief texture mapping, see texturing, relief relighting, 547
render target, 50
RenderDoc, 785
rendering
  equation, 437–438
  spectrum, 545–546
  state, 794
RenderMan, 37, 39
repeated linear interpolation, see interpolation, repeated, linear
reprojection, 143, 522–523, 936
resampling, 136–137
resolve, 142
retopology, 712
retrace, vertical, 25, 1012
retroreflection, 330
reverse mapping, 532
reversed z, 100
Reyes, 908–912
RGB, 176
color cube, 275
color mode, see color, mode, true color to grayscale, 278
RGBA, 150, 159, 1010
texture, 176
RGSS, see antialiasing, rotated grid
right-hand rule, 692
right-handed, 92
rigid-body transform, see transform, rigid-body
ringing, 256, 401, 428, 570
roll, 70, 72
ROP, 24, 25, 1010, 1032–1033
roping, 165
rotation, see under transform
roughness, 304
ROV, see rasterizer order view
RSM, see shadow, map, reflective
S3TC, 192
saccade, 931
SAH, see surface area heuristic
sample, 22
sampling, 130–137, 143, see also antialiasing
hand-limited signal, 133
centroid, 141
continuous signal, 131
discretized signal, 131
Nyquist limit, 133, 182, 186
pattern, 143
stochastic, 145, 149
stratified, 144
theorem, 133
SAT, see intersection testing, separating axis test
saturation, 276
SBRDF, 310
scalable link interface, 1013
scaling, see under transform
scan conversion, 21
scanline interleave, 1013
scatter operation, 531
scattering, 297, 589–599
  backward, 597, 598, 599
  forward, 597, 598, 607, 638
  geometric, 596, 599
  Mie, 298, 596, 597–599, 614, 620
  multiple, 607, 615, 616, 621–622, 633, 643–646
  Rayleigh, 298, 596–597, 613, 614
  single, 589, 592, 610, 614, 618, 633, 638
  subsurface, see subsurface scattering
  Tyndall, 298
scene graph, see under spatial data structure
scene-referred, 283
Schlick phase function, 599
scoreboard, 1031
scotopic, 271
screen
  coordinates, 20
  mapping, 20
  space coverage, 772, 862
scRGB, 282
SDR, 281
SDSM, see shadow, map, sample distribution
second-order equation, 957
sectioning, 19
segmentation, 683
semiconductor, 324
separating axis test, see under intersection testing
separating hyperplane theorem, 946
SGI algorithm, see triangle, strip
shade tree, 37
shader
  cores, 30
  storage buffer object, see unordered access view
  unified, see unified shader architecture
Shader Model, 38
Shadertoy, 199, 222, 753, 1048
shading, 16
  clustered, see clustered shading deferred, see deferred shading equation, 16
  flat, 120
  forward, 883
  Gouraud, 118
  hard, 652
  language, 35
  model, 103–106
  Lambertian, 109
  Phong, 118
  pixel, 23, see also pixel shader tiled, see tiled, shading toon, 652–654
  vertex, see vertex, shader
shadow, 223–265
  acne, 236
  anti-shadow, 227
  buffer, 234
  contact hardening, 251
  on curved surfaces, 229–230
  depth map, 234
  hard, 223
  map, 230, 234–252, 594, 604
  adaptive volumetric, 258
  bias, 236–239
  cascaded, 242–247
  convolution, 255
  deep, 257–259, 638
  dual, 238
  exponential, 256–257
  filtered, 252–257
  imperfect, 492
  irregular, 259–264
  light space perspective, 241
  minmax, 252
  moment, 256
  omnidirectional, 234
  opacity, 257, 612
  parallel-split, 242
  perspective, 241
  reflective, 491, 493
  sample distribution, 245
  second-depth, 238
  sparse, 246, 263
  translucent, 639
  trapezoidal, 241
  variance, 252–255
  volumetric, 644
  penumbra, 224, 228
  percentage-closer soft, 250–252
  planar, 225–229
  soft, 228–229
  projection, 225–227
  screen-space, 262
  soft, 224–225, 227–229, 247–252, 442
  umbra, 224
  volume, 230–233
shadowing-masking function, see
  masking-shadowing function
shape blending, see transform, morph targets
shared memory multiprocessor, see
  multiprocessor, shared memory
shear, see under transform
shell, 646
shell mapping, 220, 659
shortest arc, 81
shower door effect, 670
Shrek 2, 491
signed distance field, 454, 579, 677
stalling, 809
standard dynamic range, see SDR
Star Ocean 4, 286
Star Wars Battlefront, 647
star-shaped polygon, 686
Starcraft II, 459
starving, see under stage
state
  changes, 794
  sorting, 807
static buffer, 794
stationary subdivision, see surfaces, subdivision, stationary
stencil, 759
stencil buffer, see buffer, stencil
steradian, 268, 269
stereo rendering, 927–931
stereo vision, 922–924
stereopsis, 922
Stevens effect, 285
stitching, 689
stream output, 19, 48–49, 571, 705
streaming, 871–872
  multiprocessor, 1003, 1029
  texture, see texturing, streaming
stride, 702
strip, see triangle, strip
stroke, 672
stylized rendering, see non-photorealistic rendering
subdivision curves, see curves, subdivision
subdivision surfaces, see surfaces, subdivision
subpixel addressing, 689
subsurface albedo, 348–349
subsurface scattering, 305–307, 445, 607
  global, 306, 632–640
  local, 306, 347–355
subtexture, see texturing
summed-area table, see under texturing, minification
superscalar, 14
supershader, 128
surface area heuristic, 953
surface extraction, 583
surfaces
  acne, 236
  B-spline, 749, 762
  Bézier patch, 735–738
  Bézier triangle, 740–741, 745
  biquadratic, 736
  continuity, 741–742
  explicit, 944
  sphere, 944
  triangle, 944, 963
  implicit, 749–753, 944
  blending, 751
  derivatives, 751
  sphere, 956
NURBS, 781
parametric, 171, 734–747
Phong tessellation, 735, 740, 748–749
PN triangle, 46, 735, 740, 744–747, 748, 749
spline, 689, 761
subdivision, 756–767
  adaptive quadtree, 718, 779–780
  approximating, 758
  Catmull-Clark, 761–763
  displaced, 765–766
  feature adaptive, 777–779
  limit position, 760
  limit surface, 760
  limit tangents, 760
  Loop, 758–761
  mask, 759
  modified butterfly, 761
  stationary, 756
  stencil, 759
  tensor product, 735
  tessellation, 735
surfel, 573
surround, 285
SVBRDF, 310
swap buffer, see buffer, swap
swizzling, 1018
synchronization with monitor, 790, 1012, 1013
TAM, see tonal art map
tangent
  frame, 209
  map, 344
  patch, 775
  space, see under basis
  vector, 209, 729
TBN, 209
Team Fortress 2, 654, 677, 678, 940
tearing, 1012
technical illustration, 651, 673
temporal
  aliasing, see aliasing, temporal coherence, 866
delay, 1
  locality, 791
temporary register, 36
tensor product surfaces, 735
terrain chunked LOD, 874–877
tessellation, 683–690, 767–780, 853
  adaptive, 770–775
  control shader, 44
domain shader, 44
evaluation shader, 44
factors, 45
fractional, 768–770, 860
hull shader, 44
levels, 45
stage, 18, 44–46, 677
surface, 735
tessellator, 44
uniform, 767
tetrahedralization, 489
texel, 169
Texram, 189
text, 675–677, 725
texture
array, 191
atlas, 190
bandwidth, 1006
cache, see cache
coordinates, 169
cube map, 190
dependent read, 38, 177, 220, 406
matrix, 174n, 410
periodicity, 175
space, 169
volume, 189–190
volumetric, 646
texture processing cluster, 1031
texture-space shading, 910
texturing, 23, 167–222
albedo color map, 201
alpha mapping, 176, 202–208, 551
animation, 200, 203
bindless, 192
border, 174
cellular, 199
charts, 485
clamp, 174
clipmap, 867
compression, 192–198, 486, 503
ASTC, 196, 1029
BC, 192–193
DXTC, 192–193
EAC, 194
ETC, 194–195, 1029
lossy, 194
normal, 195
PVRTC, 195–196
SSTC, 192
correspondence function, 169, 174–175
decaling, 202
detail, 180
diffuse color map, 201
distortion, 687–688
image, 176–198
image size, 177
level of detail bias, 186
light mapping, 484
magnification, 177, 178–181
bilinear interpolation, 178
cubic convolution, 178
nearest neighbor, 178
minification, 177, 182–189
anisotropic filtering, 187–188
bilinear interpolation, 182
Elliptical Weighted Average, 189
level of detail, 185
mipmapping, 183–186
nearest neighbor, 182
quadrilinear interpolation, 189
summed-area table, 186–188
trilinear interpolation, 186
mipmapping, 485
mirror, 174
mirror once, 175
noise, 198, 549
one-dimensional, 173
parallax occlusion mapping, 167, 216–220
parameterization, 485, 486
pipeline, 169–176
procedural, 198–200
projective, 221, 688
projector function, 169–174
relief, 216–220, 222, 565–566, 630, 646,
853, 854
repeat, 174
seams, 486
shells, 485
sparse, 246, 263, 867–871
streaming, 870–871
subtexture, 184
swizzling, 1018
texture coordinates, 169
tiling, 795
transcoding, 870–871
value transform function, 169
vertex, 43, 186
virtual, 867–871
wrap, 174
TFAN, 712
That Dragon, Cancer, 121
thin-film interference, see light, interference,
thin-film
thread
divergence, 32, 260
group, 54, 518
shader, 31
thread-level parallelism, 1003
Index

Threading Building Blocks, 812
three plane intersection, see intersection testing, three planes
three-dimensional printing, see 3D printing
three.js, 41, 50, 189, 407, 485, 568, 628, 1048
thresholding, 656
throughput, 59, 783, 808
tile, 995
  local storage, 156
  screen, 1007, 1021
  table, 1008
  texture, 795
tiled caching, 1033
defered shading, 894, 896, 904, 914
forward shading, 895–896, 903, 904, 914
rasterization, see pipeline, rasterization shading, 893–898
triangle traversal, 996

tiling, 795
time-critical rendering, 865
timer query, 785
timewarp, 935–937
timing, 955
toroidal mapping, 369
Tom Clancy's The Division, 478
Tomb Raider (2013), 114, 116
Tomorrow Children, The, 496, 497
tonal art map, 671
tone mapping, 283–289
  global, 285
  local, 285
toon rendering, see shading, toon
top-left rule, 995
topology, 712
Torrance-Sparrow model, 334
tracking, 916, 921
transaction elimination, 1028
transcoding, see under texturing transfer function, 161, 478
  volume, 605
transform, 57, see also matrix
  affine, 58, 68
  angle-preserving, 66
  concatenation of, 65–66
  constraining, 73
  decomposition, 73–74
  Euler, 70–73
  extracting parameters, 72–73
  gimbal lock, 73
  feedback, 49
  inverse, 59, 61–64, 66, 69, 75
  adjoint method, 69
  Cramer's rule, 69, 964
Gaussian elimination, 69
LU decomposition, 69
length-preserving, 66
linear, 57–58
mirror, see transform, reflection
morph targets, 89–91
morphing, 87–91
normal, 68–69
orthographic, see under projection perspective, see under projection quaternion, 80
reflection, 63, 692, 832
rigid-body, 60, 66–67, 74, 84
rotation, 60–61
  about an arbitrary axis, 74–76
  from one vector to another, 83–84
  around a point, 61
scaling, 62–63
  anisotropic, 62
  isotropic, 62
  nonuniform, 62
  uniform, 62
translation, 59
vertex blending, 84–87, 90, 102, 1006
view, 15–16
volume-preserving, 64
translation, 59
transparency, 148–160
  order-independent, 154–159
  screen-door, 149, 858
  sorting, 152, 823
  stochastic, 149
  weighted average, 156–158
  weighted sum, 157
transparency adaptive antialiasing, 207
tree
  balanced, 820
  binary, 820
  k-ary tree, 820
trees (forest), 202, 559–560
triangle
  fan, 686, 696–697
  formula, 944, 963
  list, 696
  indexed, 703
  setup, 22, 997–998
  sorting, 152–153, 802–803
  soup, 691
  strip, 697–699
  indexed, 703
  sequential, 698
traversal, 22, 996–997
tiled, 996
triangle/object intersection, see specific objects
under intersection testing
triangulated irregular network, 705, 877
triangulation, 683–686
Delaunay, 684
trilight, 432
trilinear interpolation, 186
triple buffer, 1013
tristimulus values, 273
ture color mode, see color, mode, true color
TSM, see shadow, map, trapezoidal
turbulence, 198
T-vertex, see under polygon
TXAA, 142
Tyndall scattering, 298
UAV, see unordered access view
ubershader, 128
UBO, 795
UMA, see unified memory architecture
umbra, 224
Uncharted 2, 286, 357
Uncharted 3, 879
Uncharted 4, 290, 356–359, 492
Uncharted: Drake’s Fortune, 893
under operator, 153
underclock, 787
unified memory architecture, 1007
unified shader architecture, 35, 786
uniform buffer object, 795
uniform tessellation, 707
Unity engine, 128, 287, 482, 489, 740, 930
unordered access view, 51–52, 87, 155, 192, 896, 1016
up direction, 70
upsampling, 136

valence, 699, 758
Valgrind, 792
van Emde Boas layout, 827–828
VAO, 703
variance mapping, 370
VDC, see video display controller
vector irradiance, 379–380, 389
vector norm, 7
Vega, see under hardware
vergence, 923, 932
vertex
array, see vertex, buffer
array object, 703
blending, see under transform
buffer, 701–705, 793

cache, see cache, vertex
clustering, 709
compression, 712–715
correspondence, 87
pulling, 703
shader, 15–16, 42–43
animation, 43
effects, 43
skinning, 87
stream, 702
vertical refresh rate, 1011
vertical retrace, see retrace, vertical
vertical synchronization, see synchronization
with monitor
vertices per second, 788
VGA, 1011
video display controller, 1011
video graphics array, 1011
video memory, 1006, 1011
view frustum culling, see culling, view frustum
view space, 15, 26
view transform, see transform, view
view-independent progressive meshing, 706
VIPM, 706
virtual point light, 491
virtual reality, 523, 912, 915–940
compositor, 924
optics, 921–922
visibility
buffer, see buffer, visibility
cone, 470, 471
function, 446
test, 843
visual appearance, 103
Vive, 915, 916, 917, 922, 925, 934
von Mises-Fisher distribution, 397
Von Neumann bottleneck, 791
voxel, 578–586
voxelization, 580–582, 610–612, 974
VPL, see virtual point light
VSM, see shadow, map, variance
vsync, see synchronization with monitor
V’Tune, 792
Vulkan, 40, 814

Wang tiles, 175
Ward model, 314
warp, 31
watercolor, 652, 665
watertight model, 693
watt, 268
wave
electromagnetic, 293
transverse, 293
wavefront, 31, 1035
wavelength, 267, 293
wavellets, 199
WebGL, 41, 50, 122, 125, 129, 189, 201, 208, 407, 485, 568, 628, 631, 713, 796, 805, 829, 1048
welding vertices, 691
white point, 274
Wii, see under hardware
winding direction, 692
winding number, 968
window coordinates, 20
wireframe, 674, 675
*The Witcher 3*, 2, 263, 420, 526, 534, 873, 1049
world space, 15
wrap, see texturing, repeat
wrap lighting, 382, 633
Xbox, see under hardware
XR, 915
Y’CbCr, 892
yaw, 70n
YCoCg, 197–198, 804–805
yom, 93n
z-buffer, see under buffer
z-fighting, 1014
z-prepass, 803, 881, 882, 901, 1016
z-pyramid, 846
Zaxxon, 17
zmax-culling, see culling, zmax
zmin-culling, see culling, zmin
zonal harmonics, 401, 428, 490, 470