INTERACTIVE COMPUTER GRAPHICS

8TH EDITION

INTERACTIVE COMPUTER GRAPHICS

8TH EDITION

EDWARD ANGEL

University of New Mexico

•

DAVE SHREINER

ARM, Inc.

Pearson

Senior Vice President Courseware Portfolio Management: Marcia J. Horton
Director, Portfolio Management: Engineering, Computer Science & Global Editions: Julian Partridge
Executive Portfolio Manager: Matt Goldstein
Portfolio Management Assistant: Meghan Jacoby
Managing Content Producer: Scott Disanno
Content Producer: Carole Snyder
Rights and Permissions Manager: Ben Ferrini
Manufacturing Buyer, Higher Ed, Lake Side Communications, Inc. (LSC): Maura Zaldivar-Garcia and Deidra Smith
Inventory Manager: Bruce Boundy
Product Marketing Manager: Yvonne Vannatta
Field Marketing Manager: Demetrius Hall
Marketing Assistant: Jon Bryant
Cover Designer: SPi Global, Inc.
Cover Credit: Emma Gould, Stephen Guerin, Kasra Manavi, Cody Smith and Joshua Thorp
Printer/Binder: LSC Communications, Inc.
Full-Service Project Management: Windfall Software, Paul C. Anagnostopoulos

Credits and acknowledgments borrowed from other sources and reproduced, with permission, in this textbook appear on the appropriate page within text.

About the Cover: The book cover shows an illustration from Simtable LLC (simtable.com) of the Santa Fe Railyard as a 3D point cloud and mesh, which is calculated and rendered in realtime in the browser using WebGL. Live imagery from mobile phones, public webcams, drones, and satellites is projection mapped onto the pointclouds. Realtime. Earth is currently used on wildfires and other emergency events to give realtime coordinated situational awareness during an unfolding incident. Real-time camera pose estimates allow for GIS points, polylines, and polygons to be both overlaid in augmented reality views on the phones as well as annotations on the imagery from the users.

Library of Congress Cataloging-in-Publication Data

Pearson

ISBN – 10: 0-13-525826-X
ISBN – 13: 978-0-13-525826-2

To Rose Mary —E.A.

To Vicki, Bonnie, Bob, Cookie, and Goatee —D.S.

CONTENTS

PREFACE

This book is an introduction to computer graphics with an emphasis on applications programming. The first edition, which was published in 1997, was somewhat revolutionary in using OpenGL and a top-down approach. Over the succeeding 22 years and seven editions, this approach has been adopted by most introductory classes in computer graphics and by virtually all the competing textbooks.

A Top-Down Approach

Recent advances and the success of the first seven editions continue to reinforce our belief in a top-down, programming-oriented approach to introductory computer graphics. Although many computer science and engineering departments now support more than one course in computer graphics, most students will take only a single course. Such a course usually is placed in the curriculum after students have already studied programming, data structures, algorithms, software engineering, and basic mathematics. Consequently, a class in computer graphics allows the instructor to build on these topics in a way that can be both informative and fun. We want these students to be programming three-dimensional applications as soon as possible. Low-level algorithms, such as those that draw lines or fill polygons, can be dealt with later, after students are creating graphics.

When asked "why teach programming?," John Kemeny, a pioneer in computer education, used a familiar automobile analogy: You don't have to know what's under the hood to be literate, but unless you know how to program, you'll be sitting in the back seat instead of driving. That same analogy applies to the way we teach computer graphics. One approach—the algorithmic approach—is to teach everything about what makes a car function: the engine, the transmission, the combustion process. A second approach—the survey approach—is to hire a chauffeur, sit back, and see the world as a spectator. The third approach—the programming approach that we have adopted here—is to teach you how to drive and how to take yourself wherever you want to go. As an old auto rental commercial used to say, "Let us put *you* in the driver's seat."

The sixth and seventh editions reflected the major changes in graphics application development due to advances in graphics hardware. In particular, the sixth edition was fully shader-based, enabling readers to create applications that could fully exploit the capabilities of modern GPUs. We noted that these changes were part of OpenGL ES 2.0, which was being used to develop applications for embedded systems and handheld devices, such as cell phones and tablets, and of WebGL, its JavaScript implementation. At the time, we did not anticipate the extraordinary

interest in WebGL that began as soon as web browsers became available that support WebGL through HTML5. For the seventh edition, we switched from desktop OpenGL to WebGL.

As we noted then, WebGL applications were running everywhere, including on some of the latest smart phones, and even though WebGL lacks some of the advanced features of the latest versions of OpenGL, the ability to integrate it with HTML5 opened up a wealth of new application areas. As an added benefit, we found it much better suited than desktop OpenGL for teaching computer graphics. Our hopes for the seventh edition were more than fulfilled. WebGL has proven to be excellent API for both teaching and developing real applications that run on all platforms.

In particular, features of the seventh edition included:

- WebGL 1.0 was used for all examples and programs.
- All code was written in JavaScript.
- All code runs in recent web browsers.
- A new chapter on interaction was added.
- Additional material on render-to-texture was added.
- Additional material on displaying meshes also was added.
- An efficient matrix–vector package was included.
- An introduction to agent-based modeling was added.

For the eighth edition, building on the success of using WebGL, we have:

- Updated all examples and programs to WebGL 2.0.
- Added many additional examples.
- Switched to a fully electronic version that includes links to examples so they can be viewed with the code while reading.
- Expanded coverage of render-to-texture in a separate chapter that includes new topics, including shadow maps and projective textures.
- Added coverage of three-dimensional texture mapping.
- Updated the chapter on modeling to include an introduction to three.js, a popular higher-level JavaScript scene-graph API.
- Added coverage of point sprites for simulation.
- Expanded and updated our coverage of rendering.
- Enhanced discussions of hardware implementation and GPU architectures.

Programming with WebGL and JavaScript

When Ed began teaching computer graphics 35 years ago, the greatest impediment to implementing a programming-oriented course, and to writing a textbook for that course, was the lack of a widely accepted graphics library or application programming interface (API). Difficulties included high cost, limited availability, lack of generality, and high complexity. The development of OpenGL resolved most of the difficulties

many of us had experienced with other APIs and with the alternative of using home-brewed software. OpenGL today is supported on all platforms and is widely accepted as the cross-platform standard. WebGL builds on OpenGL's wide acceptance (it's effectively the same API), but provides a more accessible development platform using web technologies.

A graphics class teaches far more than the use of a particular API, but a good API makes it easier to teach key graphics topics, including three-dimensional transformations, lighting and shading, client–server graphics, modeling, and implementation algorithms. We believe that OpenGL's extensive capabilities and well-defined architecture lead to a stronger foundation for teaching both theoretical and practical aspects of the field and for teaching advanced concepts, including texture mapping, compositing, and programmable shaders.

Ed switched his classes to OpenGL about 24 years ago and the results astounded him. By the middle of the semester, *every* student was able to write a moderately complex three-dimensional application that required understanding of three-dimensional viewing and event-driven input. In the previous years of teaching computer graphics, he had never come even close to this result. That class led to the first edition of this book.

This book is a textbook on computer graphics; it is not an OpenGL or WebGL manual. Consequently, it does not cover all aspects of the WebGL API but rather explains only what is necessary for mastering this book's contents. It presents WebGL at a level that should permit users of other APIs to have little difficulty with the material.

Unlike earlier editions, this one uses WebGL and JavaScript for all the examples. WebGL 2.0 is a JavaScript implementation of OpenGL ES 3.0 and runs in most recent browsers. Because it is supported by HTML5, not only does it provide compatibility with other applications but also there are no platform dependences; WebGL runs within the browser and makes use of the local graphics hardware. Although JavaScript is not the usual programming language with which we teach most programming courses, it is the language of the Web. Over the past few years, JavaScript has become increasingly more powerful, and our experience is that students who are comfortable with Java, Python, C, or C++ will have little trouble programming in JavaScript. So that we can continue to support a wide audience with varied backgrounds, we have kept with a very basic JavaScript requiring only ES5. Students who have more experience with JavaScript should have little trouble updating our examples and libraries to exploit powerful new JavaScript features contained in ES6.

All the modern versions of OpenGL, including WebGL, require every application to provide two shaders written in the OpenGL Shading Language (GLSL). GLSL is similar to C but adds vectors and matrices as basic types, along with some C++ features, such as operator overloading. We provide a JavaScript library, **MV.js**, that supports both our presentation of graphics functions and the types and operations in GLSL. It also contains many functions that perform operations equivalent to deprecated functions from the earlier fixed-function versions of OpenGL.

Intended Audience

This book is suitable for advanced undergraduates and first-year graduate students in computer science and engineering and for students in other disciplines who have good programming skills. The book also will be useful to many professionals. Between us, we have taught well over 100 short courses for professionals (including many courses presented at the annual SIGGRAPH conferences, two of which are available on YouTube within the SIGGRAPH University channel), and even a Massive Online Course (MOOC) with Coursera. Our experiences with these nontraditional students have had a great influence on what we chose to include in the book.

Prerequisites for the book are good programming skills in JavaScript, Python, C, C++, or Java; an understanding of basic data structures (arrays, linked lists, trees); and a rudimentary knowledge of linear algebra and trigonometry. We have found that the mathematical backgrounds of computer science students, whether undergraduates or graduates, vary considerably. Hence, we have chosen to integrate into the text much of the linear algebra and geometry that is required for fundamental computer graphics.

Organization of the Book

The book is organized as follows:

- Chapter 1 provides an overview of the field and introduces image formation by optical devices; thus, we start with three-dimensional concepts immediately.
- Chapter 2 introduces programming using WebGL. Although the first example program that we develop (each chapter has one or more complete programming examples) is two-dimensional, it is embedded in a three-dimensional setting and leads to a three-dimensional extension.
- Chapter 3 introduces interactive graphics and develops event-driven graphics within the browser environment.
- Chapters 4 and 5 concentrate on three-dimensional concepts. Chapter 4 is concerned with defining and manipulating three-dimensional objects, whereas Chapter 5 is concerned with viewing them.
- Chapter 6 introduces light–material interactions and shading.
- Chapters 7 and 8 introduce many of the new discrete capabilities that are now supported in graphics hardware and by WebGL. All these techniques involve working with various buffers. Chapter 7 focuses on classical texture mapping with a single texture, whereas Chapter 8 concentrates on texture mapping using off-screen buffers.

These chapters should be covered in order and can be taught in about 10 weeks of a 15-week semester.

The last five chapters can be read in almost any order. All five are somewhat open-ended and can be covered at a survey level, or individual topics can be pursued in depth.

- Chapter 9 includes a number of topics that fit loosely under the heading of hierarchical modeling. The topics range from building models that encapsulate the relationships between the parts of a model, to high-level approaches to graphics over the Internet. Chapter 9 also includes an introduction to scene graphs.
- Chapter 10 introduces a number of procedural methods, including particle systems, fractals, and procedural noise.
- Chapter 11 discussed curves and surfaces, including subdivision surfaces.
- Chapter 12 surveys implementations. It gives one or two major algorithms for each of the basic steps, including clipping, line generation, and polygon fill.
- Finally, Chapter 13 surveys alternate approaches to rendering. It includes expanded discussions of ray tracing and radiosity, and an introduction to image-based rendering, parallel rendering, and concepts of virtual and augmented reality.

Several appendices are included to provide additional reference:

- Appendix A presents the details of the WebGL functions needed to read, compile, and link the application and shaders.
- Appendices B and C contain a review of the background mathematics.
- Appendix D discusses sampling and aliasing starting with Nyquist's theorem and applying these results to computer graphics.

Changes from the Seventh Edition

The reaction of readers to the first seven editions of this book was overwhelmingly positive, especially to the use of OpenGL/WebGL and the top-down approach. In the sixth edition, we abandoned the fixed-function pipeline and went to full shader-based OpenGL. In the seventh edition, we moved to WebGL, which is not only fully shader based—each application must provide a vertex shader and a fragment shader—but also a version that works within the latest web browsers.

Moving to an only online format for this edition allowed us make some interesting, and, we think, positive changes. First, all the figures are now in full color. We have moved the images from the color insert into regular figures in the chapters, which places them where they should be. We were also able to add screen shots from our examples as regular figures. The URLs for these examples are in the figure captions. Thus, students will be able to use the URL to open another browser window to run the example. Most browsers will also let the student examine the full code in another window. We believe these features will greatly enhance the examples. We have also added many new examples to the book's website (www.interactivecomputergraphics.com/Code), which are accessible through links at the end of each chapter.

For this edition, we updated all our code to use WebGL 2.0. Although we introduce few additional WebGL 2.0 features—three-dimensional texture mapping being

an exception—the newer version of GLSL results in much clearer shader code. For examples that do not use new features, we will keep WebGL 1.0 versions on the book's website.

Applications are written in JavaScript. Although JavaScript has its own idiosyncrasies and may not be the language used in students' programming courses, we have not observed any problems with students using JavaScript. JavaScript has many variants, which students and instructors may prefer to use. In addition, there are many ways to develop code in other languages and use a transpiler to produce JavaScript. Because many of these options are new and perhaps transient, we have stuck with a very basic JavaScript that should run everywhere.

We have added additional material on off-screen rendering and render-to-texture, including techniques such as projective textures and shadow maps. We have also added material on using GPUs for a variety of compute-intensive applications, such as image processing and simulation. We have also added coverage of three-dimensional texture, an added feature in WebGL 2.0, and its use for volume visualization. Because of the large amount of added material, we split the former Chapter 7 into two chapters: 7 and 8.

Given the positive feedback we have received on the core material from Chapters 1–6 in previous editions, other than updating the code to WebGL 2.0, we have tried to keep the changes to those chapters to a minimum. We see Chapters 1–8 as the core of any introductory course in computer graphics. Chapters 9–13 can be used in almost any order, either as a survey in a one-semester course or as the basis of a two-semester sequence.

Chapter 9 has been updated to be more consistent with three.js scene graphs, and includes a short introduction to the three.js API. Chapter 11 adds material on using point sprites in particle simulations.

Chapter 11 is largely unchanged. Material on implementation that was formerly in Chapter 8 in the seventh edition has been moved to Chapter 12. As interesting as we find many of the classic algorithms for tasks such as line generation and clipping, this material is no longer a core part of most first courses on Computer Graphics. We have kept the parts of the chapter that are still highly relevant and taken out parts that are no longer used in modern GPUs. Chapter 13 (formerly Chapter 12) has been updated to cover some additional approaches to rendering such as deferred shading.

Acknowledgments

Ed has been fortunate over the past few years to have worked with wonderful students at the University of New Mexico. They were the first to get him interested in OpenGL, and he has learned much from them. They include Ye Cong, Pat Crossno, Tommie Daniel, Chris Davis, Lisa Desjarlais, Kim Edlund, Lee Ann Fisk, Maria Gallegos, Brian Jones, Christopher Jordan, Takeshi Hakamata, Max Hazelrigg, Sheryl Hurley, Thomas Keller, Ge Li, Pat McCormick, Al McPherson, Ken Moreland, Martin Muller, David Munich, Jim Pinkerton, Jim Prewett, Dave Rogers, Hal Smyer, Dave Vick, Hue (Bumgarner-Kirby) Walker, Brian Wylie, and Jin Xiong.

Many of the examples in the color plates were created by these students.

The first edition of this book was written during Ed's sabbatical; various parts were written in five different countries. The task would not have been accomplished without the help of a number of people and institutions that made their facilities available to him. He is greatly indebted to Jonas Montilva and Chris Birkbeck of the Universidad de los Andes (Venezuela), to Rodrigo Gallegos and Aristides Novoa of the Universidad Tecnologica Equinoccial (Ecuador), to Long Wen Chang of the National Tsing Hua University (Taiwan), and to Kim Hong Wong and Pheng Ann Heng of the Chinese University of Hong Kong. Ramiro Jordan of ISTEC and the University of New Mexico made possible many of these visits. John Brayer and Jason Stewart at the University of New Mexico and Helen Goldstein at Addison-Wesley somehow managed to get a variety of items to him wherever he happened to be. His website contains a description of his adventures writing the first edition.

David Kirk and Mark Kilgard at NVIDIA were kind enough to provide graphics cards for testing many of the algorithms. A number of other people provided significant help. Ed thanks Ben Bederson, Gonzalo Cartagenova, Tom Caudell, Kathi Collins, Kathleen Danielson, Roger Ehrich, Robert Geist, Chuck Hansen, Mark Henne, Bernard Moret, Dick Nordhaus, Helena Saona, Vicki Shreiner, Gwen Sylvan, and Mason Woo. Mark Kilgard, Brian Paul, and Nate Robins are owed a great debt by the OpenGL community for creating software that enables OpenGL code to be developed over a variety of platforms. Our many SIGGRAPH courses have contributed to our ideas and presentations. Eric Haines (NVIDIA) and Patrick Cozzi (Analytic Graphics) have been especially helpful as reviewers, sounding boards, and co-presenters.

At the University of New Mexico, the Art, Research, Technology, and Science Laboratory (ARTS Lab) and the Center for High Performance Computing have provided support for many of Ed's projects. The Computer Science Department, the Arts Technology Center in the College of Fine Arts, the National Science Foundation, Sandia National Laboratories, and Los Alamos National Laboratory have supported many of Ed's students and research projects that led to parts of this book. David Beining, formerly with the Lodestar Astronomy Center and now at the ARTS Lab, has provided tremendous support for the Fulldome Project. Sheryl Hurley, Christopher Jordan, Laurel Ladwig, Jon Strawn, and Hue (Bumgarner-Kirby) Walker provided some of the images in the color plates through Fulldome projects. Hue Walker has done the wonderful covers for previous editions and some of the examples.

Ed would also like to acknowledge the informal group that started at the Santa Fe Complex and has continued with Redfish and Simtable, including Jeff Bowles, Ruth Chabay, Emma Gould, Stephen Guerin, Kaz Manavi, Bruce Sherwood, Scott Wittenberg, and especially JavaScript evangelist Owen Densmore, who convinced him to teach a graphics course in Santa Fe in exchange for getting him involved with JavaScript. We have all gained by the experience.

Dave would like first to thank Ed for asking him to participate in this project. We have exchanged ideas on OpenGL and how to teach it for many years, and it's exciting to advance those concepts to new audiences. Dave would also like to thank those who created OpenGL, and who worked at Silicon Graphics Computer Systems, leading the way in their day. He would like to recognize the various Khronos working groups who continue to evolve the API and bring graphics to unexpected places.

Finally, as Ed mentioned, SIGGRAPH has featured prominently in the development of these materials, and is definitely owed a debt of gratitude for providing access to enthusiastic test subjects for exploring our ideas.

Reviewers of the manuscript drafts and instructors who have used previous editions provided a variety of viewpoints on what we should include and what level of presentation we should use. These reviewers and instructors include Gur Saran Adhar (University of North Carolina at Wilmington), Mario Agrular (Jacksonville State University), Michael Anderson (University of Hartford), Norman I. Badler (University of Pennsylvania), Mike Bailey (Oregon State University), Marty Barrett (East Tennessee State University), C. S. Bauer (University of Central Florida), Bedrich Benes (Purdue University), Kabekode V. Bhat (The Pennsylvania State University), Isabelle Bichindaritz (University of Washington, Tacoma), Cory D. Boatright (University of Pennsylvania), Eric Brown, Robert P. Burton (Brigham Young University), Sam Buss (University of California, San Diego), Kai H. Chang (Auburn University), Patrick Cozzi (University of Pennsylvania and Analytic Graphics, Inc), James Cremer (University of Iowa), Ron DiNapoli (Cornell University), John David N. Dionisio (Loyola Marymount University), Eric Alan Durant (Milwaukee School of Engineering), David S. Ebert (Purdue University), Richard R. Eckert (Binghamton University), W. Randolph Franklin (Rensselaer Polytechnic Institute), Natacha Gueorguieva (City University of New York/College of Staten Island), Jianchao (Jack) Han (California State University, Dominguez Hills), Chenyi Hu (University of Central Arkansas), George Kamberov (Stevens Institute of Technology), Mark Kilgard (NVIDIA Corporation), Lisa B. Lancor (Southern Connecticut State University), Chung Lee (California State Polytechnic University, Pomona), John L. Lowther (Michigan Technological University), R. Marshall (Boston University and Bridgewater State College), Hugh C. Masterman (University of Massachusetts, Lowell), Bruce A. Maxwell (Swathmore College), Tim McGraw (West Virginia University), James R. Miller (University of Kansas), Rodrigo Obando (Columbus State University), Jeff Parker (Harvard University), Jon A. Preston (Southern Polytechnic State University), Harald Saleim (Bergen University College), Andrea Salgian (The College of New Jersey), Lori L. Scarlatos (Brooklyn College, CUNY), Han-Wei Shen (The Ohio State University), Oliver Staadt (University of California, Davis), Stephen L. Stepoway (Southern Methodist University), Bill Toll (Taylor University), Michael Wainer (Southern Illinois University, Carbondale), Yang Wang (Southern Methodist State University), Steve Warren (Kansas State University), Mike Way (Florida Southern College), George Wolberg (City College of New York), Xiaoyu Zhang (California State University San Marcos), Ye Zhao (Kent State University), and Ying Zhu (Georgia State University). Although the final decisions may not reflect their views—which often differed considerably from one another—each one forced us to reflect on every page of the manuscript.

We would also like to acknowledge the entire production team at Addison-Wesley. Ed's editors, Peter Gordon, Maite Suarez-Rivas, and Matt Goldstein, have been a pleasure to work with through eight editions of this book and the OpenGL primer. For this edition, Carole Snyder at Pearson has provided considerable help. Starting with the second edition, Paul Anagnostopoulos at Windfall Software has always been more than helpful in assisting with TEX problems. Ed is especially grateful

to Lyn Dupré. If the readers could see the original draft of the first edition, they would understand the wonders that Lyn does with a manuscript.

Ed wants to particularly recognize his wife, Rose Mary Molnar, who did the figures for his first graphics book, many of which form the basis for the figures in this book. Probably only other authors can fully appreciate the effort that goes into the book production process and the many contributions and sacrifices our partners make to that effort. The dedication to this book is a sincere but inadequate recognition of all of Rose Mary's contributions to Ed's work.

Dave would like to recognize the support and encouragement of Vicki, his wife, without whom creating works like this would never occur. Not only does she provide warmth and companionship but also invaluable feedback on our presentation and materials. She's been a valuable, unrecognized partner in all of Dave's OpenGL endeavors.

Ed Angel
Dave Shreiner

INTERACTIVE COMPUTER GRAPHICS

8TH EDITION