



Preface

This book is designed for junior- or senior-level computer science, science, or engineering students with a basic background in linear algebra and experience with an object-oriented programming language. Working professionals with comparable background should be able to follow this book and learn the essential computer graphics concepts by themselves.

The technical content and organization of this book represent our answer to the question:

If students can only schedule one elective computer graphics course in their undergraduate education, what should we teach in such a course?

When answering this question, we strive to achieve two major objectives:

- **To provide practical information.** The knowledge should be practical with potential applicability in the students' chosen field of profession.
- **To provide essential information.** The knowledge should be essential concepts that support students' future self-learning in the field of computer graphics.



Tutorials

We approach these objectives by focusing on topics that are relevant to popular computer graphics applications. To address the “one elective course” time restriction, only concepts that are relevant to interactive applications are covered. To maximize potential applicability, source code for example implementations (the tutorials) of the concepts is provided. These concept-demonstration tutorials are implemented using popular APIs. To ensure students learn the associated concepts together with the APIs, each tutorial has an OpenGL version and a DirectX version. We are in the process of porting the tutorials to the XNA Framework, a third graphics API.

The UWBGL Libraries

This book uses “the development of comprehensive software libraries suitable for building popular interactive computer graphics applications” as the central theme to connect and relate all the tutorials. All tutorials are implemented based on the UWBGL library system. Initially this library is trivial. As new concepts are introduced, corresponding sample implementation will be discussed and integrated into the UWBGL. In this way, the complexity of the library builds gradually as more concepts are presented in the book. There are a total of 18 different versions of the UWBGL library. This library system increases in complexity throughout the book and eventually enables fast prototyping of moderately complex interactive computer graphics applications.

Organization

We present the concepts associated with interactive graphics applications in three distinct parts.

1. Part I discusses the issues surrounding designing and implementing interactive applications based on a simple graphical user interface (GUI). To ensure the focus is on user interaction and software design topics, coverage of computer graphics is minimal in Part I.
2. Part II discusses the fundamental concepts for building graphics applications: graphical primitives, transformation, and (simple) applied linear algebra. To ensure the context is simple for understanding these concepts, the presentation is carried out in two dimensions.



3. Part III extends the discussion to the third dimension by introducing the computer graphics camera model and discussing techniques in software implementation.

Throughout the presentation, the independence of concepts from implementation is emphasized. The tutorials serve to reinforce this theme, where the same concepts are demonstrated in multiple versions based on competing APIs. Following this theme, we have chosen to omit the discussion of hardware shading and illumination. This is because the field is currently undergoing rapid changes and most of the materials involved are highly hardware- and API-dependent. Instead, implementation with the simple traditional Phong model is demonstrated in Appendix A. We believe that this field will continue to undergo rapid changes, and before it settles, the best source of up-to-date information will be online documentation and manuals.

Using This Book in a Course

The book organizes the topics above to support the gradual building of the UWGL library system. It is important that the first four chapters of the book are covered in their given order. These chapters present the foundation and describe the requirements of the application/library that students will be building. Beyond the first four chapters, the topics can be covered in almost any order. In all cases, as soon as sufficient computer graphics topics are covered (by around Chapter 10 or 11), students should commence working on a final project based on the library system. In these final projects, students should demonstrate their knowledge and understanding by designing and developing an original and significantly complex application.

At the University of Washington, Bothell, two 10-week quarter courses are taught based on the contents of this book. The first course is a 2D graphics course concentrating on software infrastructure, user interaction, coordinate transformation, and 2D hierarchical modeling. This 2D course covers Parts I and II of the book. The first half of the second course covers the rest of the book. The second half of the second course covers topics in hardware shading and illumination based on up-to-date online documentation.

For more advanced students with a background in event-driven GUI programming, it is possible to review the topics in Part I with an initial programming assignment. In this case, Chapters 3 and 4 still need to be covered for an introduction to programming with graphics APIs.



This book can be used to teach building graphics applications to computer science (CS) major students, as well as non-major students with the appropriate programming background (experience with object-oriented programming). For CS majors, the course should concentrate on building and improving the UWGL library system. For non-CS majors, the course should focus on using the UWGL library system in developing applications (instead of developing the libraries).

Advantages of This Book

The major strength of this book is in the presentation of conceptual and practical insights required for the building of many popular interactive graphics applications. Essential concepts in interactive graphics application development are covered, including topics in software engineering, software design patterns, graphical user interfaces, real-time simulation, application programming interface (API) models, graphical primitives, coordinate transformations, scene nodes and hierarchies, blending, file texturing, and camera models and interaction. These issues are extremely important issues, especially for students who wish to pursue careers in computer graphics-related fields. This book offers readers a substantive learning experience designed to further enhance their computer graphics knowledge.

Software Update

We plan to continue improving and updating the tutorials and to include the support of additional APIs (e.g., XNA Framework and potentially Java and OpenGL) in the future. Please refer to <http://depts.washington.edu/cmmr/bigal/> for up-to-date source code of the tutorials.

Acknowledgments

First and foremost, thanks to all CSS 450 and CSS 451 students from the University of Washington, Bothell. Over the years, many brilliant students helped shape the design of the materials presented in this book. In 1999, after the raster-algorithm-based CSS 450 class, Jeff Wartes was the first to point out: “This course has been a disappointment. Drawing lines are only so interesting. I still have no idea how computer graphics relates to the games I play.”



The development of the materials that eventually became this book was started in part as a reaction to these sincere comments. During the experimentation in the earlier years, the 2001 batch of CSS 450 students, especially Steve Baer, Jennifer Beers, Kazuko Hass, Gabriel Holmes, and Derek Gerstmann, critically evaluated the design of the library. Having done such a wonderful job, Steve has become an author of this book. Other significant contributions from students include Chris Dannunzio on working with DirectInput, Adam Smith and Sean Brogan on working with swap chains, Jack Nichols and Alexei Yatskov on programming in C#, and Peter Yiap and Robert Stone on programming with XNA. Jordan Phillips worked on the initial pass of the library organization; Jason Pursell, William Frankhouser, and Ethan Verrall developed detailed guides for the tutorials in Chapter 2; and Adrian Bonar built the documentation support system. A thank you also goes to colleagues in the community: Steve Cunningham for his guidance and support for this project, Edward Angel for the enlightening discussions on different approaches to teaching computer graphics, Charles Jackels for his encouragements, and Frank Cioch for the challenging pedagogical discussions. The supporting letters from Gladimir Baranoski, Peng-Ann Heng, Cary Laxer, Bob Lewis, Tiow-Seng Tan, Hung-Chuan Teh, and Herbert Yang on the initial ideas for this textbook helped secure a grant from the National Science Foundation that partially funded this project. Rebecca Reed-Rosenberg worked closely with Kelvin Sung on the evaluation of the materials. Ivan Lumala and John Nordlinger provided critical support toward the end of the writing of this book. We would like to thank our editors, Alice Peters and Kevin Jackson-Mead; thank you for the hard work, diligence, and timely feedback.

Kelvin Sung would like to thank the University of Washington, Bothell, for the sabbatical support for working on this book. The materials presented are based upon results from projects supported by the University of Washington, Bothell, Scholar Award 2004 and the National Science Foundation Grant No. DUE-0442420.¹ Toward the end, many of the tutorials were refined with support from Microsoft Research under the Computer Gaming Curriculum in Computer Science RFP, Award Number 15871.

Part I of this book is based liberally on Chapter 18 of the second edition of *Fundamentals of Computer Graphics*. Although the contents are reorganized with additional detail and extensive code examples that integrate smoothly with the rest of this book, the basic concepts on building interactive applications are the same.

¹ Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.



This book was written using the *LaTeX* document preparation software in the *WinEdt*²/*MiKTeX*³ editing environment. The figures were made by the authors using *Autodesk Maya*, *Adobe Illustrator*, and *Jasc Paint Shop Pro*. The program source code documentation is built based on *Doxygen*.⁴ The authors would like to thank the creators of these wonderful programs.

Kelvin Sung
Peter Shirley
Steven Baer

²<http://www.winedt.com/>

³<http://www.miktex.org/>

⁴<http://www.doxygen.org/>