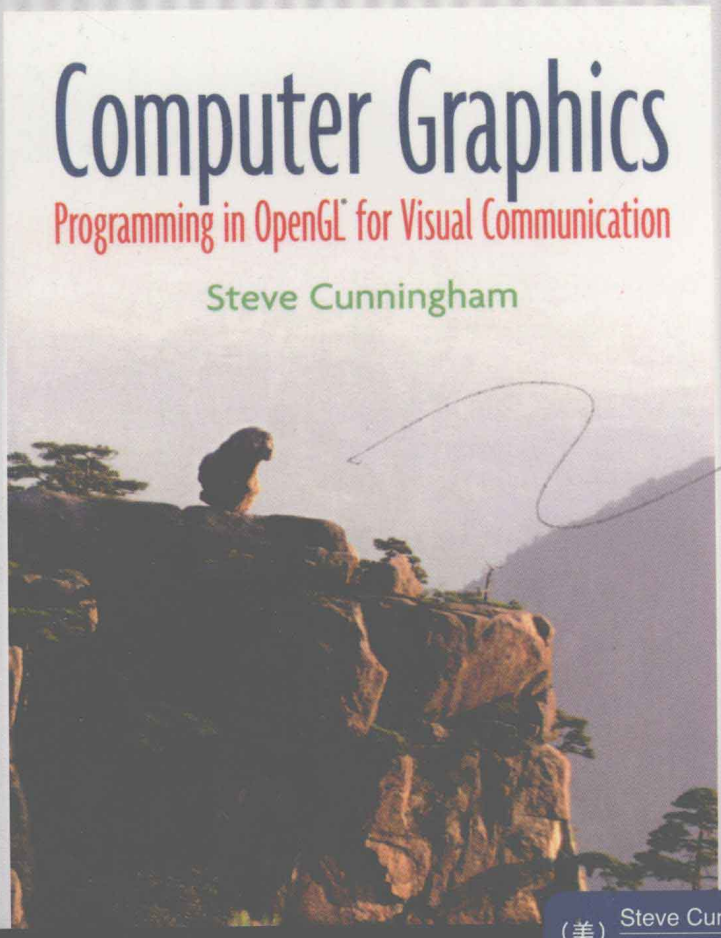
HZ BOOKS
华章教育
经 典 原 版 书 库
PRENTICE HALL

计算机图形学

（英文版）



（美）Steve Cunningham 著
加州大学斯坦尼斯洛斯分校

机械工业出版社
China Machine Press

经 典 原 版 书 库

计算机图形学

（英文版）

Computer Graphics
Programming in OpenGL for
Visual Communication

（美）Steve Cunningham
加州大学斯坦尼斯洛斯分校 著

机械工业出版社
China Machine Press

本书版权登记号：图字：01-2008-1782

图书在版编目（CIP）数据

计算机图形学（英文版）/（美）坎宁安（Cunningham, S.）著. —北京：机械工业出版社，2008.5

书名原文：Computer Graphics: Programming in OpenGL for Visual Communication

（经典原版书库）

ISBN 978-7-111-23916-1

Ⅰ.计…　Ⅱ.坎…　Ⅲ.计算机图形学—英文　Ⅳ.TP391.41

中国版本图书馆CIP数据核字（2008）第049530号

机械工业出版社（北京市西城区百万庄大街22号　邮政编码　100037）

责任编辑：迟振春

北京京北制版厂印刷　·　新华书店北京发行所发行

2008年5月第1版第1次印刷

145mm×210mm　·　18.125印张（彩插0.5印张）

标准书号：ISBN 978-7-111-23916-1

定价：36.00元

凡购本书，如有倒页、脱页、缺页，由本社发行部调换

本社购书热线：（010）68326294

出版者的话

文艺复兴以降，源远流长的科学精神和逐步形成的学术规范，使西方国家在自然科学的各个领域取得了垄断性的优势；也正是这样的传统，使美国在信息技术发展的六十多年间名家辈出、独领风骚。在商业化的进程中，美国的产业界与教育界越来越紧密地结合，计算机学科中的许多泰山北斗同时身处科研和教学的最前线，由此而产生的经典科学著作，不仅擘划了研究的范畴，还揭橥了学术的源变，既遵循学术规范，又自有学者个性，其价值并不会因年月的流逝而减退。

近年，在全球信息化大潮的推动下，我国的计算机产业发展迅猛，对专业人才的需求日益迫切。这对计算机教育界和出版界都既是机遇，也是挑战；而专业教材的建设在教育战略上显得举足轻重。在我国信息技术发展时间较短、从业人员较少的现状下，美国等发达国家在其计算机科学发展的几十年间积淀的经典教材仍有许多值得借鉴之处。因此，引进一批国外优秀计算机教材将对我国计算机教育事业的发展起积极的推动作用，也是与世界接轨、建设真正的世界一流大学的必由之路。

机械工业出版社华章图文信息有限公司较早意识到“出版要为教育服务”。自1998年开始，华章公司就将工作重点放在了遴选、移译国外优秀教材上。经过几年的不懈努力，我们与Prentice Hall，Addison-Wesley，McGraw-Hill，Morgan Kaufmann等世界著名出版公司建立了良好的合作关系，从它们现有的数百种教材中甄选出Tanenbaum，Stroustrup，Kernighan，Jim Gray等大师名家的一批经典作品，以“计算机科学丛书”为总称出版，供读者学习、研究及庋藏。大理石纹理的封面，也正体现了这套丛书的品位和格调。

“计算机科学丛书”的出版工作得到了国内外学者的鼎力襄助，国内的专家不仅提供了中肯的选题指导，还不辞劳苦地担任了翻译和审校的工作；而原书的作者也相当关注其作品在中国的传播，有的还专程为其书的中译本作序。迄今，“计算机科学丛书”已经出版了近260

个品种，这些书籍在读者中树立了良好的口碑，并被许多高校采用为正式教材和参考书籍，为进一步推广与发展打下了坚实的基础。

随着学科建设的初步完善和教材改革的逐渐深化，教育界对国外计算机教材的需求和应用都步入一个新的阶段。为此，华章公司将加大引进教材的力度，除“计算机科学丛书”之外，对影印版的教材，则单独开辟出“经典原版书库”。为了保证这两套丛书的权威性，同时也为了更好地为学校和老师们服务，华章公司聘请了中国科学院、北京大学、清华大学、国防科技大学、复旦大学、上海交通大学、南京大学、浙江大学、中国科技大学、哈尔滨工业大学、西安交通大学、中国人民大学、北京航空航天大学、北京邮电大学、中山大学、解放军理工大学、郑州大学、湖北工学院、中国国家信息安全测评认证中心等国内重点大学和科研机构在计算机的各个领域的著名学者组成“专家指导委员会”，为我们提供选题意见和出版监督。

这两套丛书是响应教育部提出的使用外版教材的号召，为国内高校的计算机及相关专业的教学度身订造的。其中许多教材均已为M. I. T.，Stanford，U.C. Berkeley，C. M. U. 等世界名牌大学所采用。不仅涵盖了程序设计、数据结构、操作系统、计算机体系结构、数据库、编译原理、软件工程、图形学、通信与网络、离散数学等国内大学计算机专业普遍开设的核心课程，而且各具特色——有的出自语言设计者之手、有的历经三十年而不衰、有的已被全世界的几百所高校采用。在这些圆熟通博的名师大作的指引之下，读者必将在计算机科学的宫殿中由登堂而入室。

权威的作者、经典的教材、一流的译者、严格的审校、精细的编辑，这些因素使我们的图书有了质量的保证，但我们的目标是尽善尽美，而反馈的意见正是我们达到这一终极目标的重要帮助。教材的出版只是我们的后续服务的起点。华章公司欢迎老师和读者对我们的工作提出建议或给予指正，我们的联系方法如下：

电子邮件：hzjsj@hzbook.com
联系电话：（010）68995264
联系地址：北京市西城区百万庄南街1号
邮政编码：100037

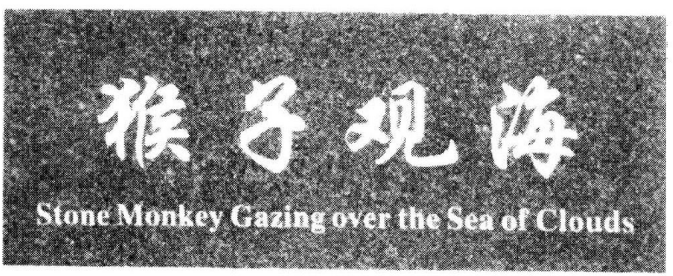
专家指导委员会

（按姓氏笔画顺序）

尤晋元　王　珊　冯博琴　史忠植　史美林
石教英　吕　建　孙玉芳　吴世忠　吴时霖
张立昂　李伟琴　李师贤　李建中　杨冬青
邵维忠　陆丽娜　陆鑫达　陈向群　周伯生
周克定　周傲英　孟小峰　岳丽华　范　明
郑国梁　施伯乐　钟玉琢　唐世渭　袁崇义
高传善　梅　宏　程　旭　程时端　谢希仁
裘宗燕　戴　葵

To Mike Bailey for his outstanding example in computer graphics education and his help with many of the ideas that led to this book

The cover is a photograph of the formation "Stone Monkey Gazing over the Sea of Clouds" in the Yellow Mountains in Anhui Province, China. The photograph was taken by the author shortly before sunrise, the traditional time to view the sea of clouds in these mountains, in June, 2006. Below is a photograph (in black and white) of a plaque marking the best view of this formation.



The author chose this cover to honor Prof. Jiaoying Shi of Zhejiang University, a longtime leader in computer graphics in China, and because it looks like a student who is deeply contemplating a project (in computer graphics, of course!)

Preface

Computer graphics is one of the most exciting ways that computing has made an impact on the world. From the simple ways that spreadsheets allow you to create charts to see data, to the ways graphics has enhanced entertainment by providing new kinds of cartoons and special effects, to the ways graphics has enabled us to see and understand scientific principles, computer graphics is everywhere we turn. This important presence has come from the greatly improved graphics hardware and software that is found in current computing systems. With these advances, computer graphics has emerged from being a highly technical field needing very expensive computers and frame buffers and requiring programmers to master all the mathematics and algorithms needed to create an image. It has become a field that allows the graphics programmer to think and work at a much higher level of modeling and to create effective images that communicate effectively with the user. We believe that the beginning computer graphics course should focus on how the student can learn to create effective communications with computer graphics, including motion and interaction, and that the more technical details of algorithms and mathematics for graphics should be saved for more advanced courses.

What Is Computer Graphics?

Computer graphics is involved in any work that uses computation to create images, whether those images are still or moving; interactive or fixed; on film, video, screen, or print. This makes it a very broad field, encompassing many kinds of uses in the creative, commercial, and scientific worlds. The breadth of the field has led to the development of many kinds of tools to create and manipulate images for all these different areas. And finally, the large number of tools and applications means that one could learn many different things about computer graphics.

Most of the books on computer graphics fall into one of two groups. The first is the traditional academic textbook on graphics, emphasizing the algorithms and techniques used for modeling, rendering, and viewing. These are important concepts, but an emphasis on how images are made can take time away from considering what images mean. The second kind of book focuses on the applications used to create images, particularly in the commercial and entertainment areas. Here the capabilities and limitations of the applications shape the coursework, and you can do little that is not designed into the application. Basic concepts can be included in studies of applications, but they often take second place to a focus on learning how the application works.

This book takes something of a middle road between algorithms and applications. We do not give a major emphasis to the details of the algorithms and techniques involved in computer graphics, and we do not deal with applications for creating images. Rather, we view computer graphics as the art and science of creating synthetic images by programming the geometry, appearance, and presentation of the contents of the images, and by displaying the results of that programming on appropriate devices that support graphical output and interaction. This focus on creating images by programming means that we must understand something of basic concepts as we learn about representing graphical and interaction concepts in ways that can be used by the

computer. This focus on programming to create images both empowers and limits the graphics student, because it places the student in control of the whole process of image making.

But the process of creating images is not the end in itself; the images are important because they communicate information to the viewer, and so our study must also include a consideration of effective visual communication. So the work of the graphics professional in creating images is to understand the content the images are to communicate, to develop appropriate representations for the geometric objects that are to make up the images, to assemble these objects into an appropriate geometric space where they can have the proper relationships with one as needed for the image, to define and present the look of each of the objects as part of that scene, to specify how the scene is to be viewed, and to specify how the scene as viewed is to be displayed on the graphic device. The programming may be done in many ways, but in current practice it typically uses a graphics API that supports the necessary modeling and does most of the detailed work of rendering the scene that is defined through the programming. A number of graphics APIs are available, but the OpenGL API is probably most commonly used currently and gives us a good platform to learn how to create effective images.

In addition to the creation of the modeling, viewing, and look of the scene, the graphics professional has two other important tasks. Because a static image does not present as much information as a moving image, it may be important to design some motion into the scene—that is, to define some animation for the image. And because the viewer may want or need to be able to control the content of the image, the way the image is seen, or the kind of computation that is done for the image, it may be important to design ways for the user to interact with the scene as it is presented. These additional tasks are also supported by the graphics API.

What Is a Graphics API?

An API is an *Application Programming Interface*—a set of tools that allow a programmer to work in an application area. The API's tools are oriented to the tasks of the application area and allow a programmer to design applications using the concepts of the area without having to deal with the details of the computer system. Among the advantages of an API is that it hides the details of any one computer system and allows the programmer to develop applications that will work on any of a wide range of systems. So a **graphics** API is a set of tools that allow a programmer to write applications that include interactive computer graphics operations without dealing with the details of graphics operations or with system details for tasks such as window handling and interactions.

Besides covering the basic ideas of interactive computer graphics, this book will introduce you to the OpenGL graphics API and to give you a number of examples that will help you understand the capabilities that OpenGL provides and will allow you to learn how to integrate graphics programming into your other work. Like most APIs, OpenGL supports quite advanced work that goes beyond the level we will use in a text for a beginning course; resources at `http://www.opengl.org/` will be useful if you want to look at the API in more depth.

Why Do Computer Graphics?

Computer graphics has many faces, so there are many reasons why one might want to use computer graphics in his or her work. Many of the most prominent uses of computer graphics are

to create images for the sciences (scientific visualization, explanations to the public) and entertainment (movies, video games, special effects), for creative or aesthetic work (art, interactive installations), for commercial purposes (advertising, communication, product design), or for general communication (animated weather displays, information graphics). The processes described in this book are all fundamental to each of these applications, although some of the applications will require the kinds of sophistication or realism in images that are not possible through simple API programming.

In all of these application areas, and more, there is a fundamental role for computer graphics in solving problems. Problem solving is a basic process in all human activity, so computer graphics can play a fundamental role in almost any area, as shown in Figure 1. This figure describes what occurs as someone:

- identifies a problem
- addresses the problem by building a model that represents it and allows it to be considered more abstractly
- identifies a way to represent the problem geometrically
- creates an image from that geometry so that the problem can be seen
- uses the image to understand the problem or the model and to try to understand a possible solution

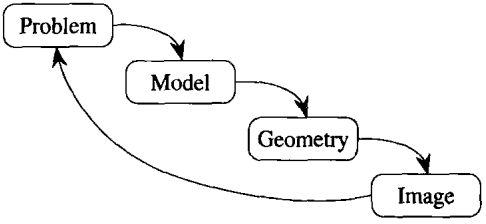


Figure 1 Computer graphics in the problem-solving process.

The image that represents a problem can be made in many ways. One of the classic uses of images in problem solving is simply to sketch an image—a diagram or picture—to communicate the problem to a colleague so it can be discussed informally. (It is folk wisdom that restaurants are not happy to see a group of scientists or mathematicians come to dinner because they write diagrams on the tablecloth!) But an image can also be made with computer graphics, and this is especially useful when it is important to share the idea with a larger audience. If the model permits it, this image may be an animation or an interactive display so that the problem can be examined more generally than a single image would permit. That image, then, can be used by the problem solver or the audience to gain a deeper understanding of the model and hence of the problem, and the problem can be refined iteratively and a more sophisticated model created, and the process can continue.

This process is the basis for all of the discussions in Chapter 9 on graphical problem solving in the sciences, but it may be applied to more general application areas. In allowing us to bring the visual parts of our brain and our intelligence to a problem, computer graphics gives us a powerful tool to think about the world. In the words of Mike Bailey of Oregon State University, computer graphics gives us a "brain wrench" that magnifies the power of our mind, just as a physical wrench magnifies the power of our hands.

Overview of the Book

This book is a textbook for a beginning computer graphics course for students who have a good programming background, equivalent to a full year of programming courses. We use C as the programming language in our examples because it is the most common language for developing applications with OpenGL, though the course could use C++ and extend some of the processes here into an object-oriented approach. The book can be used by students with no previous computer graphics experience and with less mathematics experience and advanced computer science studies than the traditional computer graphics course. Because we focus on graphics programming rather than on algorithms and techniques, we have fewer instances of data structures and other computer science techniques. This means that this text can be used for a computer graphics course that can be taken earlier in a student's computer science studies than the traditional graphics course, or for self-study by anyone with a sound programming background. In particular, this book can be used as a text for a computer graphics course at the community college level.

Many, if not most, of the examples in this book are taken from sources in the sciences, and we include a chapter that discusses several kinds of scientific and mathematical applications of computer graphics. This emphasis makes this book appropriate for courses in computational science or for computer science programs that want to develop ties with other science programs on campus, particularly programs that want to provide science students with a background that will support development of computational science or scientific visualization work. It has been tempting to use the word *visualization* somewhere in the title of this book, but we would reserve that word for material that is primarily focused on the science with only a sidelight on the graphics; because we reverse that emphasis, the role of scientific visualization is in the application of the computer graphics.

The book is organized along fairly traditional lines, treating projection, viewing, modeling, rendering, lighting, shading, and many other aspects of the field. These are all structured into the scene graph, a construct that is extremely useful to help the programmer organize scenes. There is also an emphasis on the graphics pipeline to produce images. Besides the basic techniques of creating images, the book has an emphasis on using computer graphics to address real problems and to communicate results effectively to the viewer. As we move through this material, we describe a full range of general concepts and techniques in computer graphics and show how the OpenGL API provides the graphics programming tools that implement these principles and techniques. Our goal is to give the student both an understanding of graphics concepts and skill in using a graphics API (application programming interface) to implement graphics operations and create effective images for communicating with a viewer. We are mindful of the limitations of a single course, so while we also outline or sketch the algorithms and principles that lie behind the way the techniques are implemented, we do not develop these as fully as a more theoretical text would. Your instructor will provide these in more detail if he or she finds it useful to do so.

We have tried to match the sequence of chapters in the book to the sequence we would expect to be used in a beginning computer graphics course, and in some cases the presentation of one chapter will depend on your knowing the content of an earlier chapter. However, in other cases it will not be critical that earlier chapters have been covered. It should be pretty obvious if other chapters are assumed, and we may make that assumption explicit in some chapters. Assuming that the course will want to cover interactive graphics, we believe that all the chapters through Chapter 7 on interaction should be covered (though the mathematics chapter, Chapter 4, may be treated as a reference), as well as Chapter 10 on the graphics pipeline. Beyond that, the chapters are relatively independent, and you may choose them as you will.

The book focuses on computer graphics programming with a graphics API and in particular uses the OpenGL API to implement the basic concepts that it presents. Each chapter includes a general discussion of a topic in graphics as well as a discussion of the way the topic is handled in OpenGL. However, another graphics API might also be used, with the OpenGL discussion serving as an example of the way an API could work. Many of the fundamental algorithms and techniques that are at the root of computer graphics are covered only at the level they are needed to understand questions of graphics programming. This differs from most computer graphics textbooks that place a great deal of emphasis on understanding these algorithms and techniques. We recognize the importance of these for persons who want to develop a deep knowledge of the subject and suggest that a second graphics course can provide that knowledge. We believe that the experience provided by API-based graphics programming will help you understand the importance of these algorithms and techniques as they are developed and will equip you to work with them more fluently than if you met them with no previous background.

This book includes several features that are not found in most beginning textbooks. These features support a course that fits the current programming practice in computer graphics. The discussions in this book focus on 3D graphics and almost completely omit uniquely 2D techniques. It has been traditional for computer graphics courses to start with 2D graphics and move up to 3D because some of the algorithms and techniques have been easier to grasp at the 2D level, but without that concern it is easier to begin by covering 3D concepts and discuss 2D graphics as the special case where all the modeling happens in the X-Y plane.

Modeling is a very fundamental topic in computer graphics, and there are many different ways that one can model objects for graphical display. This book uses the standard beginning approach of focusing on polygon-based modeling because that approach is supported by OpenGL and most other graphics APIs. The discussion on modeling in this book places an important emphasis on the scene graph as a fundamental tool in organizing the work needed to create a graphics scene. The concept of the scene graph allows the student to design the transformations, geometry, and appearance of a number of complex components such that they can be implemented quite readily in code, even if the graphics API itself does not support the scene graph directly. This is particularly important for hierarchical modeling, but it also provides a unified design approach to modeling and has some very useful applications for placing the eyepoint in the scene and for managing motion and animation.

A key feature of this book is an emphasis on using computer graphics to create effective visual communication. This recognizes the key role that computer graphics has taken in developing an understanding of complex problems and in communicating this understanding to others, from small groups of working scientists to the general public. This emphasis is usually missing

from computer graphics textbooks, although we expect that most instructors include this somehow in their courses. The discussion of effective communication is integrated throughout several of the basic chapters in the book, because it is an important consideration in graphics modeling, viewing, color, and interaction. We believe that a systematic discussion of this subject will help prepare students for more effective use of computer graphics in their future professional lives, whether this is in technical areas in computing or is in areas where there are significant applications of computer graphics.

This book also places a good deal of emphasis on creating interactive displays. Most computer graphics textbooks cover interaction and the creation of interactive graphics. Historically this was a difficult area to implement because it involved writing or using specialized device drivers, but with the growing importance of OpenGL and other graphics APIs this area has become much more common. Because we are concerned with effective communication, we believe it is critically important to understand the role of interaction in communicating information with graphics. Our discussion of interaction includes a general treatment of event-driven programming and covers the events and callbacks used in OpenGL, but it also discusses the role of interaction in creating effective communications. This views interaction in the context of the task that is to be supported, not just the technology being studied, and thus integrates it into the overall context of the book.

You will probably notice that the figures in this book are not as sophisticated as those in most computer graphics texts. This is deliberate. We believe that if we are talking about creating graphical communications with the level of OpenGL that the book presents, we should make the figures with this same level. We hope that you will find these "homemade" figures effective and that perhaps you may be challenged to create illustrations of your own that may well be better than ours.

This book's approach, discussing computer graphics principles without covering the details of the algorithms and mathematics that implement them, differs from those of most computer graphics textbooks that place a much larger emphasis on understanding these graphics algorithms and techniques. We recognize the importance of these ideas for persons who want to develop a deep knowledge of the subject and suggest that a second graphics course can provide that knowledge. We believe that the experience provided by API-based graphics programming will help the student understand the importance of these algorithms and techniques as they are developed and will equip someone to work with them more fluently than if they were covered with no computer graphics background.

Outcomes

When you have finished with a computer graphics course based on this book, we believe that you should have the following knowledge or skills:

- An understanding of how graphical information is represented to a graphics system and encoded by the system to create images
- An understanding of how to organize graphical information in a program in order to create images with a graphics API
- An understanding of how to use events in a graphics system to create interactive graphics displays

- An understanding of the issues involved in making images that communicate effectively to a viewer
- Skill at using the OpenGL graphics API to create effective images

We hope that you will have applied the principles and skills you are learning to an area where computer graphics makes an important contribution, and that you will understand what that contribution is. We support this for the sciences but recognize that there are many other areas where this is also the case. If a course uses this text with an application area other than the sciences, the author would like to hear of it.

Credits

The initial development of this project was supported by National Science Foundation grant DUE-9950121. All opinions, findings, conclusions, and recommendations in this work are those of the author and do not necessarily reflect the views of the National Science Foundation. The author also gratefully acknowledges sabbatical support from California State University Stanislaus and thanks the San Diego Supercomputer Center (SDSC), most particularly Mike Bailey (now at Oregon State University) for hosting the sabbatical where this work was started and for providing significant assistance with both visualization and science content. Kris Stewart of San Diego State University, Angela Shiflet of Wofford College, Rozeanne Steckler and Kim Baldridge of SDSC, and Ian Littlewood of CSU Stanislaus provided helpful contributions to the chapter on graphics in the sciences. Sam Rebelsky of Grinnell College provided helpful comments. Sampson Asare of the University of Botswana, Petros Mashwama of the University of Swaziland, and Marcelo Zuffo of the University of São Paolo gave the author opportunities to use early versions of the manuscript in workshops. Several of the author's students at CSU Stanislaus and San Diego State University provided important examples and insight for this project. Ken Brown was particulary helpful with several figures and concepts in this manuscript. The author also thanks his students Mike Dibley, Ben Eadington, Jordan Maynard, and Virginia Muncy for their contributions through examples.

The author also wishes to thank the many reviewers who have looked at early stages of the book. They gave help at many levels, from general concepts to very detailed suggestions and corrections. To the extent that this book is successful, my students and colleagues and the reviewers get much of the credit; if anything in the book is not successful, that comes back to the author.

Steve Cunningham
Coralville, Iowa

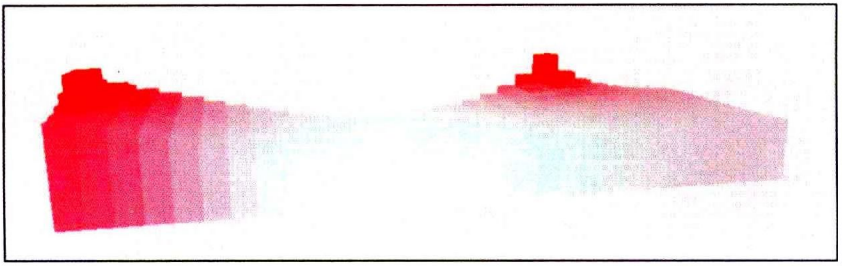


Figure 0.7 and Figure 9.2 Heat distribution in a bar.

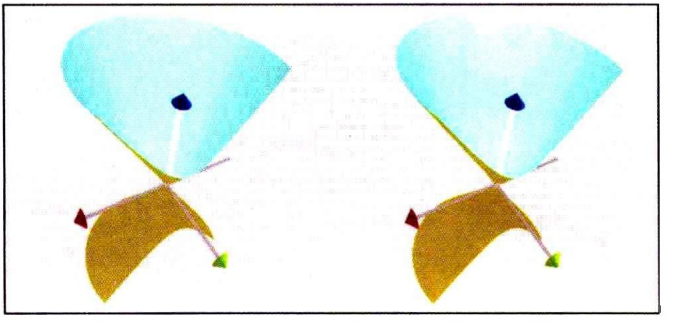


Figure 1.15 A stereo pair that can be viewed by merging individual eye images.

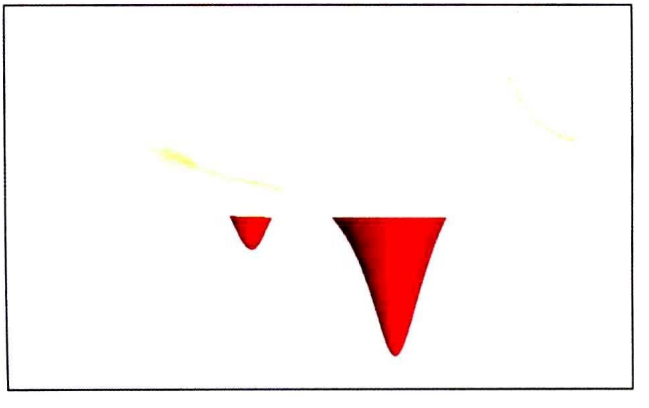


Figure 2.18 Traditional surface graph presented with three lights to show its shape.

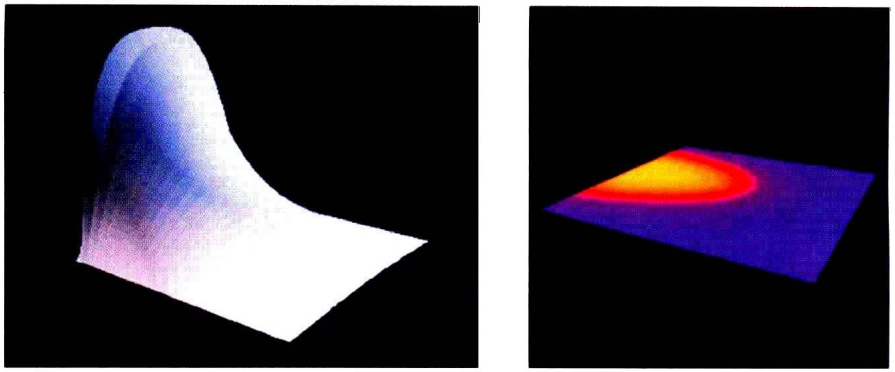


Figure 2.20 Change in temperature over time from solutions 1 and 3.

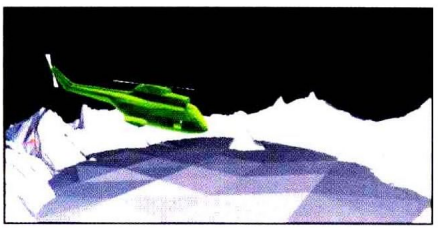


Figure 2.29 A scene that we will describe with a scene graph.

Figure 2.33 The same scene as in Figure 2.29 but with the eyepoint following directly behind the helicopter.

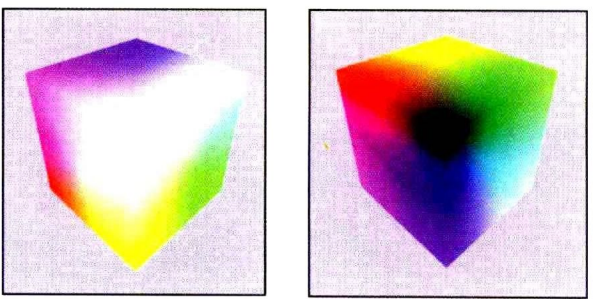


Figure 5.1 Two views of the RGB cube—from the white (*left*) and black (*right*) corners.

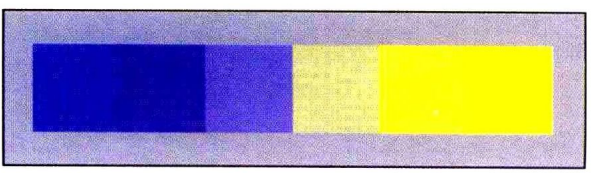


Figure 5.2 A sequence of six colors between yellow and blue.

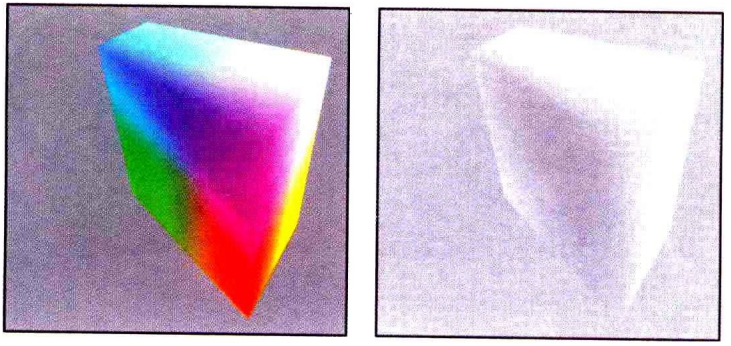


Figure 5.3 A plane of constant luminance in the RGB cube in both color (*left*) and grayscale (*right*).

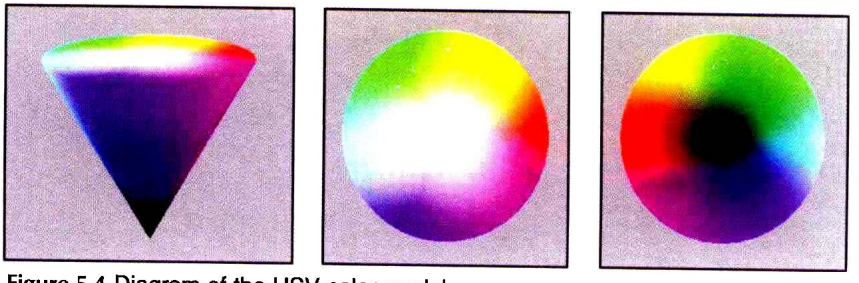


Figure 5.4 Diagram of the HSV color model.

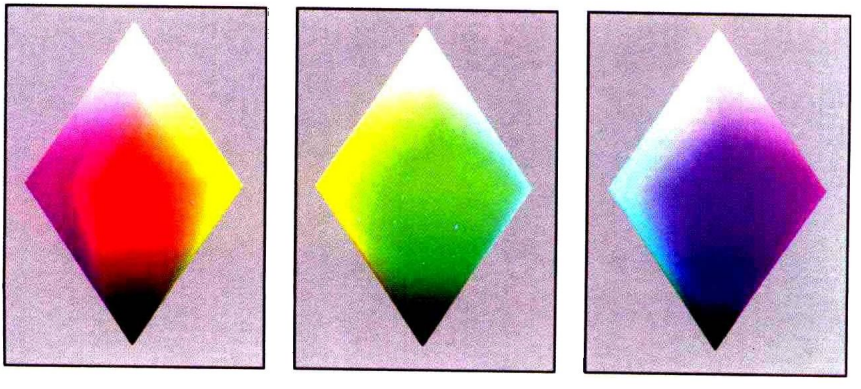


Figure 5.5 Approximately 120 degree views of the HLS double cone from the directions of red (*left*) green (*middle*), and blue (*right*).