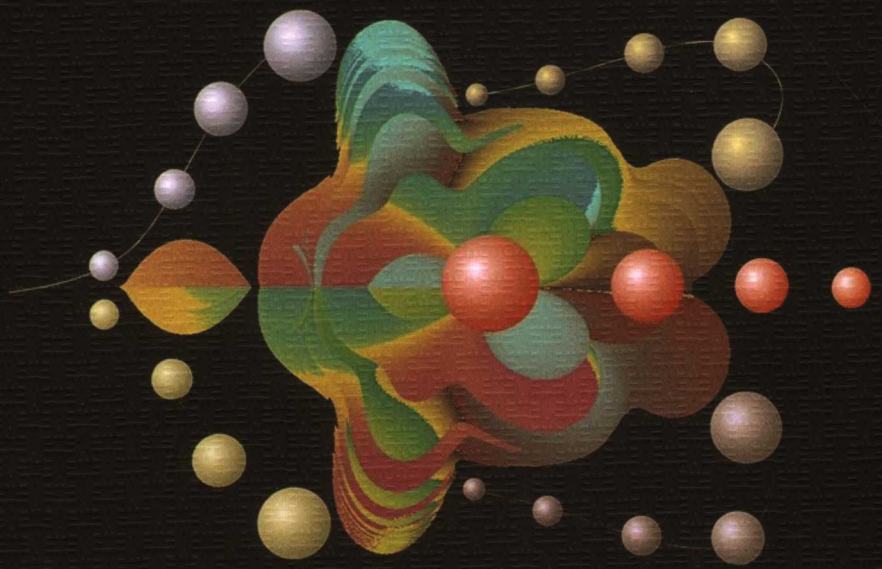
世界著名计算机教材精选

3D计算机图形学

（OpenGL版）

Samuel R. Buss 著

唐龙 等译



3D COMPUTER GRAPHICS

A Mathematical Introduction with OpenGL

清华大学出版社

世界著名计算机教材精选

3D 计算机图形学

(OpenGL 版)

Samuel R. Buss 著
唐　龙　等译

清华大学出版社
北　京

Originally published by Cambridge University Press in 2006.
This reprint edition is published with the permission of the Syndicate of the Press of the University of Cambridge, Cambridge, England.

THIS EDITION IS LICENSED FOR DISTRIBUTION AND SALE IN THE PEOPLE'S REPUBLIC OF CHINA ONLY, EXCLUDING HONG KONG, MACAO SAR AND TAIWAN.

本书中文翻译版由 Cambridge University Press 授权给清华大学出版社出版发行。

仅限于中华人民共和国境内(不包括中国香港、澳门特别行政区和中国台湾地区)销售发行。

北京市版权局著作权合同登记号　国字 01-2004-1915 号

图书在版编目(CIP)数据

3D 计算机图形学(OpenGL 版)/(美)巴斯(Buss, S. R.)著;唐龙等译. —北京: 清华大学出版社, 2006.10
(世界著名计算机教材精选)
书名原文: 3D Computer Graphics
ISBN 7-302-13604-1

Ⅰ. 3…　Ⅱ. ①巴… ②唐…　Ⅲ. 三维—动画—图形软件, OpenGL—教材　Ⅳ. TP391.41

中国版本图书馆 CIP 数据核字(2006)第 091693 号

出 版 者: 清华大学出版社　　**地　　址**: 北京清华大学学研大厦
http://www.tup.com.cn　　**邮　　编**: 100084
社 总 机: 010-62770175　　**客户服务**: 010-62776969
印 刷 者: 北京国马印刷厂
装 订 者: 三河市化甲屯小学装订二厂
发 行 者: 新华书店总店北京发行所
开　　本: 185×260　**印张**: 22　**字数**: 546 千字
版　　次: 2006 年 10 月第 1 版　2006 年 10 月第1 次印刷
书　　号: ISBN 7-302-13604-1/TP・8214
印　　数: 1～3000
定　　价: 45.00 元

序

最近 10 年，计算机图形学有着异常的发展，从二维(2D)图形进步到复杂的、高质量的三维(3D)的环境。在娱乐方面，电影和计算机游戏中广泛地使用计算机图形学。动画片发展到完全靠计算机制作。甚至非动画片也主要依靠计算机图形学去开发特定的效果。例如，20 世纪 70 年代中期电影“星球大战”的成功就是个证明。个人计算机图形和家庭游戏控制器的能力现在已经改进扩展到了低价系统也有能力每秒显示数百万个多边形。

计算机图形学在非娱乐方面也有重要应用。例如，在学习训练中经常使用虚拟现实系统。对于科学计算可视化和计算机辅助设计(CAD)，计算机图形学是一种不可缺少的工具。我们需要有很好的方法，以便直观地显示大量数据集和大规模的科学仿真结果。

自从计算机出现以来，20 世纪 60 年代初就开始热起来的计算机图形学的艺术和科学已经很有进展。计算机图形学发展成一个丰富多彩，深奥且有吸引力的领域。本书的目的瞄准计算机图形学的数学基础，并用 OpenGL 编程做具体介绍。我深信对于计算机图形学的任何一种高级应用，懂得数学的基础知识都是很重要的。为此，本书试图做到完全覆盖计算机图形学的基础数学知识。此原则指导着本书的选材，选用对计算机图形学的从业者，尤其是对软件开发有实际重要意义的话题。我希望本书对用在该领域的标准工具，尤其是对这些工具背后的数学理论，做全面的介绍。

关于本书

本书计划的形成是依据我作为一个数学学术研究者的个人经验和在多个应用计算机项目，包括计算机游戏和虚拟现实等项目的实践。当我在 San Digo 加州大学(UCSD)给一个数学班讲授计算机图形学和几何时，就开始写本书。构成该课程的内容包括介绍用 OpenGL 编写三维图形学程序和计算机图形学的数学基础知识。在教授这门课程中，我确信需要这样一本书，来介绍计算机图形学的基础数学理论。

写此书的另一动力是我涉及一些虚拟现实和计算机游戏的项目。本书中所包含的许多话题作为主要内容提出，其原因是我发现它们常用在计算机游戏应用程序中。当前计算机游戏和虚拟现实应用程序是技术上要求很多的软件项目：这些应用程序要求软件能够显示令人信服的三维环境。通常，应用程序必须保持跟上多个目标的运动；维持住主要目标的有关光照，色彩和纹理等信息；并在屏幕上以每秒 30 或 60 帧显示。此外，需要相当高的艺术性和创造性的技巧来做成为一种适宜的三维环境。毫不奇怪，这要求由大量程序员、艺术家和设计师进行成熟的软件开发。

3D 计算机图形学要求很宽广的数学，这或许有点令人感到惊讶。然而，这是客观存

在的事实。而且,数学趋向于高雅和跨学科。计算机图形学需用数学知识使来自各个领域的构造和方法组成在一起。这些领域包括:几何学、计算方法、线性代数、数值分析、抽象代数、数据结构和算法。实际上,计算机图形学是数学的组合性和优雅性应用的最好佐证。

本书呈现了应用和理论主题的结合。在更多的地方,我建议采用 OpenGL,它是一种对 3D 图形非常有用的、自由的、跨平台的编程环境。本书给出了可以免费因特网下载的 OpenGL 程序的 C 和C++ 代码,并介绍 OpenGL 如何实现本书所讨论的许多数学概念。本书还描述了一个光线追踪软件包;该软件包也可以从因特网络下载。在理论方面,本书强调计算机图形学的数学基础,远超过我所见到的任何其他教材。对于有可能使用的诸如 OpenGL 或 Direct3D 这类工具,或者稍微扩展,CAD 程序,我深信计算机图形学的数学基础知识都是相当重要的。

在本书中,所选择数学主题依据是它们对于图形学的重要性和相关性。当然,如果对计算机图形学很关键的概念,例如,等轴测坐标的投影几何表示,我会毫不犹豫地介绍。如果你要正确地使用计算机图形软件技术,良好的数学知识使无价的,如果你要开发新的或有创造性的使用计算机图形学,很好数学知识就更重要。

如何使用本书

本书侧重于作为教科书,作为自学的材料或是参考资料使用。特别强调建议你试运随本书提供的程序,写一些你自己的 OpenGL 程序。请注意,本书最好与一本介绍用 OpenGL 编程的书联系一起读。*OpenGL Programming Guide*(Woo et al., 1999)是学习 OpenGL 的好资料,有时也称该书为红皮书("red book")。如果你是初次学习OpenGL,*OpenGL Programming Guide* 也许是有点令人气馁。若是如此,*OpenGL SuperBible*(wright Jr., 1999)介绍 OpenGL 所需的数学知识更少些。另一本书 *OpenGL A Primer*(Angel, 1999)也对 OpenGL 进行了很好的综述。本书的概要如下所述。按覆盖一个课程所需材料多少来安排章节。当然不需要依次读这些材料。实际上,除了第 8 章依赖于第 7 章之外,较后面的章节可以大量地单独地阅读。

第 1 章引言,介绍计算机图形学的基本概念;绘制点、线和多边形;以多边形来造型;动画;OpenGL 编程入门。

第 2 章变换与视图,讨论绘制流水线,线性和仿射变换,二维和三维矩阵,平移和旋转,homogeneous 坐标,在 OpenGL 中的变换,正交视图和透视变换,投影几何,像素化,Gouraud 和扫描线插值,Bresenham 算法。

第 3 章光照、亮度和着色,发表 Phong 光照模型;周围环境光,漫反射和镜面反射;在 OpenGL 中的光照和物质属性;Cook-Torrance 光照模型。

第 4 章均值与插值,提出线性插值;重心坐标,双线性插值,凸面,双曲线插值,球面的线性插值。这是较多数学的一章,带有在本书其他地方用到的许多工具。在初次读时,你也许希望跳过本章好多地方,当需要时再回来读。

第 5 章纹理映射,讨论纹理和纹理坐标,映射,超采样和颤动,撞击映射,环境映射,以及 OpenGL 中的纹理映射。

第 6 章色彩，色彩感知的发表，加和减色彩，RGB 和 HSL 色彩的表达。

第 7 章贝塞尔曲线，三阶和通用阶次的贝塞尔曲线；De Casteljau 方法；细分；分段贝塞尔曲线；Hermite 多项式；贝塞尔曲面片；OpenGL 中的贝塞尔曲线；有理贝塞尔曲线；圆锥曲线段，阶次的提高；Catmull-Rom，Bessel-Overhauser 插值，紧张-连续性-斜偏的样条；贝塞尔曲面插入。

第 8 章 B 样条曲线，描述规则的和不规则的 B 样条及其特性，OpenGL 中的 B 样条，de Boor 算法，Blossoms，平滑特性，有理的 B 样条(NURBS)，圆锥形截面，结点插入，贝塞尔曲线的关系，以及用 B 样条曲线插入。本章混合有介绍性话题和较专业化话题。我们给全部证明，但建议在初次读时跳过这些证明。

第 9 章光线跟踪，提出递归光线跟踪、反射和透射，分布式光线跟踪，逆向光线跟踪，还有逃离以及避免光线跟踪的技巧。

第 10 章相交测试，描述与球面、平面、三角形，多边行及其他面的相交测试，和体边界寻址，还有修枝相交测试。

第 11 章辐射度，给出路径，组成因子，以及辐射度公式；半立方体方法；Jacobi，Gauss-Seidel 和 Southwell 重复方法

第 12 章动画与运动学，讨论关键帧，灵活地移入和灵活地移出，定方位表达方式，公式，插入公式，有关节刚性多形体的向前和反向的运动学。

附录 A 数学背景，复习矢量，矩阵，线性代数和计算。

附录 B 射线追踪软件包，介绍光线跟踪软件包。该软件可免费下载。

练习题分散在书中，尤其在更多介绍性章节。常提供提示，不是非常困难。特别建议你做练习以便理解教材。书中少数章节，还有一些定理，证明和练习都给标有星号(*)。这表示这些内容是可选，不太重要或者可以安全地跳过去，不会影响你理解书中其他部分。定理、推论、插图和练习按每章分别编号。

获取有关的软件

书中讨论的所有软件实例都可以从网络下载，其网址为：

http://math.ucsd.edu/~sbuss/Mathcg/

该软件作为源文件和 PC 可执行的文件是可用的。此外，完全的 Microsoft Visual C++ 项目文件也是可用的。

该软件包含有一些小的 OpenGL 的程序和一个相关的大的光线跟踪软件包。除了在商业产品或实体化项目类使用必须得到许可之外，使用该软件不受任何限制。

以 OpenGL 开始

OpenGL 是绘制 3D 图形的 API(应用程序设计接口)独立平台。使用 OpenGL 的一个大优点在于它是得到广泛支持的工业标准。其它 3D 环境，例如著名的 Direct3D，有类似的能力；然而，Direet3D 是专门针对 Microsoft Windows 操作系统的。

正式的 OpenGL 网址是 http://www.opengl.org。这网址有大量的资料，但若你正开始学习 OpenGL，更多有用的资料也许是教材和代码实例，可用到：

http://www.opengl.org/developers/code/tutorials.html

按此文本用 OpenGL Utility Toolkit 子程序支持 OpenGL 程序,简称为 GLUT,它被广泛地使用,提供一个使用简便的接口,适于控制 OpenGL 窗口和处理简单的用户输入。你通常需要从其余的 OpenGL 文件安装 GLUT 文件

如果你正用 Microsoft Visual C++,那么,OpenGL 头文件和库文件由 Visual C++ 包含。当然,你自己必需下载 GLUT 文件。OpenGL 也可用其他的开发环境,例如,Borland C++ 的编译器。

下载来自 Nate Robin 的,适于 Windows 操作系统的 GLUT 最近版本的正式的 OpenGL 可用网址是:

http://www.xmission.com/~nate/glut.html

为了在 Windows 机器上安装需要的 GLUT,你应该像其他 OpenGL 头文件,如 glu.h 一样,将头文件 glut.h 放在该同一目录中。同样还应该将 glut32.dll 文件和 glut32.lib 放在与 OpenGL 相关文件 glu32.dll 文件和 glu32.lib 同一目录中。

OpenGL 和 GLUT 也工作在其它各种不同类型操作系统下。我并未试过所有这些操作系统,但列出其中特别突出的一些,作为读者试图在其他环境中运行 OpenGL 的目标(当然,在软件开发领域,经常发生变化,因此,这些连接也许很快变得过时)。

对于 Macintosh 计算机,你可以在如下 Apple 计算机网站找到有关 OpenGL 和 GLUT 库的信息:

http://developer.apple.com/opengl/

OpenGL 和 GLUT 也工作在 Cygwin 系统下。它实现一个 Windows 下类似 Unix 开发环境。Cygwin 方面可用信息在

http://cygwin.com/ 或 http://sources.redhat.com/cygwin/

适于 Sun Solaris 系统的 OpenGL 可从如下网站得到:

http://www.sun.com/software/graphics/OpenGL/

有一个 OpenGL 兼容的系统 Mesa3D 可从 http://www.mesa3d.sourceforge.net/ 得到。在各种操作系统中,包括 Linux,并且支持一图形包变种。

其他计算机图形资源

你也许希望以计算机图形学方面其他信息原始资料来补充本书。一本更好理解的教科书是 Foley 等写的书(1997)。另一本非常出色的书是 Möller 和 Haines 的书(1999)。Blinn(1996, 1998)和 Glassner(1999)写的文章也很有趣。

最后,在 internet 网络上庞大的有关计算机图形学的理论和实践所有信息是很有用的。你可以在这儿找到 OpenGL 程序实例和有关图形硬件以及理论和数学的发展。通过你所喜爱的搜索引擎可找到许多这类东西,但你也许可以用 ACM Transactions on Graphics 网站 http://www.acm.org/tog/ 作为起点。

关于指导书

此书的本意是用于先进的低年级或高年级大学生课程或引导性的研究生课程。它是

基于我在提高班和研究生班所讲授的计算机图形学的大部分内容。在两季度大学生课程中，包括本书中的大部分材料，或多或少依次出现于此。某些更高级话题应跳过去，然而，更著名的 Cook-Torrance 光学模型和双曲线插补，某些 Bézier 和 B 样条曲线和面片最好从大学生课程中略去。我在大学生课程中也不包含更难的证明。

明确地推荐学生学习本书得到有关用 OpenGL 编程。虽然本书含有大量 OpenGL 大纲式资料，学生会还需有一个另加的原材料以便学习用 OpenGL 编程的细节。编程必需有些 C，C++ 或 Java 方面经验（当我们写到此，对于 Java 没有标准化的 OpenGL API，然而，Java 非常接近 C 或C++，学生确实可以做到为精通包含这文本的简单程序所需转换）。我自己课程的第一季度包含有编程功课，首先在于二维图形；其次是基于 solar 系统三维图形变换练习；第三是多边形造型（要学生画出图 1.11 中那样的圆环）；第四另加材料和对场景的光照；最后，自选的内容，学生们从中选择一个他们自己的设计作业。课程的第二季度包含有用 Bézier 面片造型物体功课（利用 Blinn 的文章（1987）如何构造 Utah 茶壶对此是有帮助的），写一个程序，用鼠标拾取插补点画 Catmull-Rom 和 Overhauser 样条曲线；用计算机辅助设计程序 *3D Studio Max*（本书并不包含任何有关如何使用 CAD 程序的材料）；用由本书所补充的光线跟踪软件；实现分布式光线跟踪的某些外观；然后，以他们所选择另一个期末考试作业作为结束。通过课程的资料可在网络上从我的主页 http://math.ucsd.edu/~sbuss/中找到。

致谢

本书中非常少材料是原创的。原创的最主要考虑在于组织和陈述方面；有些地方，我试图提出新的，比那些以往已知的更为简单的证明。提出的材料往往缺乏缘由和可信度的，但在大多数情况下，这些材料来自其他人的。我已包含了我通过请教原创作者所学习过的项目的参考文献和原创性资源很容易探知的题目的参考文献。当然，在课程可信度方面，我并不试图过多理会。

我从一些原材料学习计算机图形学。首先，我在 SAIC 与一些人，包括 Tom Yonkman 及其夫人，Teresa Buss，Subsequently 等一起做计算机图形学课题。我曾在 Angel Studics 从事多年的计算机游戏应用程序工作，在那儿从 Steve Rotenberg，Brad Hunt，Dave Etherton，Santi Sacerra，Nathhan Brown，Ted Carson，Daniel Slumenthaly 及其他人学习到大量东西，受益良多。我尤其从 Steve Rotenberg 受惠，他是我在计算机图形学方面高级课题及当前研究工作中的导师。

我在 UCSD 教了几个学期计算机图形学，在各个学期分别使用了由 Watt 和 Watt (1992)，Watt(1993)，以及 Hill(2001)写的教科书。本书写作来自教这些班时所做的笔记。

本书早期草稿的通篇校对，我大大地受惠于 Frank Chang 和 Malachi Pust。此外，我得感谢 Michaei Bailey，Stephanic Buss（我的导师），Chris Calabro，Joseph Chow，Daniel Curtis，Tamsen Dunu，Rosalie Iemhoff，Cyrus Jam，Jin-Su Kim，Vivek Manpuria，Jang-

Won Oh, Horng Bin Ou, Chris Pollett, John Rapp, Don Quach, Daryl Sterlion, Aubin Whitley,以及订正本书初步草稿的不知名审阅者。我还得感谢 Cambridge University Prees 拷贝编辑和排版。我多么希望避免所有留下的错误都是我自己的责任。

本书中的插图是借助于一些软件系统精制的。这些主要插图使用 Zandt 的 patrick 宏软件包 LATEX 来生成。一些插图使用 Geuzaine 的程序修改版 GL2PS 来生成,以便将 OpenGL 图像转换成 PostScript 文件。少量插图从屏幕截取的位图来生成,并用 Adobe Photoshop 转换成 PostScript 图像。

目　录

第1章 引 言

本章讨论在计算机图形学背后的一些基本概念，特别强调如何从用 OpenGL 画简单的图形起步。本章主要部分解释用 OpenGL 画图形的最简单的方法和各种绘制方法。如果这是您首次接触 OpenGL，读本章时，强烈建议您了解一些 OpenGL 命令的代码和实验。

本章所包含的第一个话题是图形显示的不同模型。在本书稍后所包含的此话题特别重要的思想是任意的三维几何形体可用一个多边形集合来近似值，尤其是作为一个三角形集合。其次，我们讨论用 OpenGL 编程来显示由点、线、三角形及其他多边形构成的简单的二维和三维模型的一些基本方法。我们还描述如何设置颜色和多边形的方向、如何才能消除隐藏面，以及如何借助于双缓冲制作动画片。所包含的简单的 OpenGL 代码实例阐明所有这些能力。本章后部分将讨论如何使用变换、如何设置视点、如何加上光照和明暗法处理、如何加上纹理，以及其他话题。

1.1 显示的模型

我们从描述适于图形显示模型的三种模式开始(1)画点；(2)画线；(3)画三角形以及其他多边形面片。这三种模式适合于不同的图形显示的硬件结构。画点适合于粗糙地将一个图形图像模型作为一个矩形的像素矩阵。画线适合于矢量图形显示。画三角形以及其他多边形适用现代图形显示硬件以显示三维图像。

1.1.1 矩形的像素矩阵

最常见的低层次模型是将一个图形图像处理成一个矩形的像素矩阵，其中，每个像素可以独立地设置成一个不同的颜色和亮度。这种显示模型适用阴极射线管(CRT)和电视。若像素足够小，它们不能被肉眼分别地观察到，因而，图像即使由点组成的，也能呈现为一个单一平滑的图像。这技术使用在艺术方面，就像非常有名的用马赛克方法。而且，用点画法的那些图由小块彩色面片组成，但当从足够远处观测时，呈现为由一个连续图像组成。

无论如何请注意，作为一个矩形的像素矩阵的图形图像模式只是一个合适的抽象，而不是完全准确的。例如，在一个 CRT 或电视屏幕上，每个像素实际上是由三个不同的点组成的(或荧光的点)：每个点对应于三基色(红、蓝和绿)之一，并且可独立地设定一个亮度值。这样，每个像素点实际上是由三个彩色点构成的。用一个放大镜，你可以看到在像素上分离的颜色(见图 1.1)。(最好试一试用一个低分辨率的设备，例如，电视机。取决

于屏幕的物理设计,您也许能看到在各个点或条带上分离的颜色)。

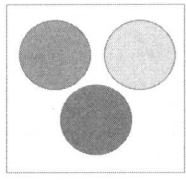


图 1.1 一个像素由三个子区域或子像素构成,每个像素点显示三种彩色中之一种

矩形矩阵模式的第二方面不足是偶尔使用子像素图像寻址。例如,激光打印机和喷墨打印机减轻走样问题,诸如在线条和符号上,由微小的色剂或墨点形成的锯齿状的边。最近,某些手持计算机(也就是,掌上机)可以更高分辨率显示文本。甚至高到只有以每个像素作为三个独立的可寻址子像素来处理才可能做到水平。在这方面,设备能定位文本在子像素级,并做到细节更高级,更好字符构造。

然而,在本书中,并不讨论子像素的问题,我们总是将一个像素作为一个单一的矩形点,可将它设置成所要的颜色和亮度。计算机图形学图像的像素基础,对我们有时侯是很重要的。在第 2.4 节,我们讨论用像素近似一条笔直的倾斜线问题。此外,当使用纹理映射和光线跟踪时,必须注意消除走样的问题,采样一条连续或高分辨率的图像成为一组像素时会出现这问题。

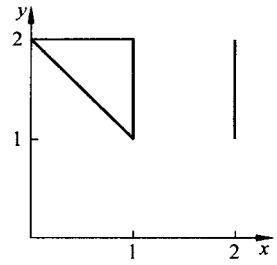
总体上我们一般不考虑像素,而是在更高级的基于多边形造型上做工作。原则上,通过直接设置图像中每个像素亮度级别,可以画任何图画。实际上,这是很困难的,也很费时间的。然而,关于可以用矩形阵列像素绘制图形的图像,在更高级的图形编程应用中,我们并没有太多想到此事。画线或尤其画多边形时,图形硬件处理大量转换工作,将结果转换成像素亮度级。已有各种成熟的技术,将多边形(或三角形)在计算机屏幕上成像素矩阵,包括明暗法处理和平滑方法,以及纹理映射应用。这些都包含在本书稍后部分。

1.1.2 矢量图形

在传统的矢量图形中的办法,将图像作为直线的集合。这样一来,不可能做出实体的物体,而是画出二维的形状,基本的图形,或者三维物体的线框图像。矢量图形系统的典型例子是笔绘图机,这包括“turtle geometry”系统。笔绘图机有一支画笔在平展的纸页面上移动。有用命令包括:(a)抬笔,笔从纸面上升高抬起;(b)落笔,笔尖落低到纸上;(c)移动到(x,y),笔在一直线上移动,从当前位置到达坐标(x,y)那一点。当笔抬起时,笔不画而移动;当笔落下时,笔移动时就画(见图 1.2)。另外,也许还有转换不同颜色的笔的命令,以及为了更容易画图像的方便的命令。

矢量图形设备的另一例子是矢量图形显示终端,它照例是单色监视器,可以画任意的线。在这些矢量图形显示终端上,屏幕是一个很宽阔的磷光区域,并不存在像素。一种传统示波器也是一种矢量图形显示设备的实例。

矢量图形显示器和基于像素显示器使用完全不同的图像表达方式。在基于像素的系统中,将屏幕图像存储成一个位图,也就是一个含有所有像素颜色的表。另一方面,一个矢量图形系统,将图像存储为一个命令列表,也就是抬笔、落笔和移动命令的一张列表。称这样一张命令列表为一张显示列表。




```
penup( );
moveto(2, 2);
pendown( );
moveto(2, 1);
penup( );
moveto(1, 2);
pendown( );
moveto(0, 2);
moveto(1, 1);
moveto(1, 2);
```

图 1.2　矢量图形命令的实例

现在，基于像素的硬件已非常流行，因此，逻辑上矢量为基础的图形系统，通常显示在基于像素的硬件上。缺点是，基于像素的硬件不能直接画任意线，必须用像素近似表示线。另一方面，优点是，更复杂的图，例如填充区域，都可以画。

现代矢量图形系统超越过直线，合并在一起，还包括画曲线、文本、多边形，以及其他形状，例如，圆和椭圆等的能力。这些系统还有能力用一种颜色或一种图案对一个区域填充或做明暗处理。它们通常限于画二维图。Adobe 的 PostScript 语言是一个卓越的现代矢量图形系统例子。

1.1.3　多边形的造型

再经提炼或成熟，提升一步就是图形图像的多边形造型。对于所造型的三维几何形体最常见的，首先是作为一组多边形集合，然后，映射成在一个二维显示器上的多边形的形体。基本的显示硬件通常是基于像素的。但是，现在大多数计算机有处理多边形或者至少是三角形的专用图形硬件。绘制三角形的图形硬件也用于现代计算机游戏系统；以至于，对图形硬件性能的通常评测是每秒钟能绘制三角形数量。写这本书的时候，相对便宜的硬件的名义峰值执行速率高于每秒百万个多边形。

几乎在每个三维计算机图形系统都使用基于多边形的造型。对于交互的三维图形的生成，它是一种重要的工具，它用于具有照片一样真实感的绘制，包括电影动画片的绘制。

一个多边形的造型系统中的基本操作是画一单个三角形。另外，还提供三角形着色和明暗法处理。其中，“明暗法处理”意味着改变交迭在三角形上的颜色。另一种重要的工具是采用纹理映射。它可以用于将图像或其他纹理涂画在多边形中。由诸如 PC 机上廉价的图形板这类专用图形硬件，都是非常普遍提供着色、明暗处理和纹理映射。

这些技术的效果是使多边形造型的对象看起来更有真实感。参见图 3.1。你将看到一个茶壶的六种造型。图中(a)展示一线框的茶壶，可以在矢量图形设备上造型。(b)展示同样形体，但用实体颜色填充，结果仅显现出不带有三维效果的轮廓。(c)—(f)展示用光照效果绘制的茶壶：(c)和(e)展示扁平明暗法处理(即无影)的多边形，便于清楚显出茶壶的多边形的本性。(d)和(f)完整的明暗法处理，其中，用不同的颜色交迭在多边形上。这明暗法处理做了一件相当好的事情，掩蔽茶壶的多边形的本性，并且大大地增强了图像的真实感。

1.2 坐标、点、线和多边形

下一节讨论坐标系统和画点、线及多边形的一些基本的约定。我们所强调的将在于由 OpenGL 所使用的约定和命令。现在,只讨论在 xy 平面或 xyz 空间确定的位置画顶点。第二章将围绕旋转、平移及其他变换,解释如何移动顶点和几何形体。

1.2.1 坐标系统

当画几何形体图形时,由特定的一组顶点位置决定形体的位置。例如,一个三角形的位置和几何形态由其三个顶点位置关系来指定。图形编程语言,包括 OpenGL,允许你为特定的点位置设置你自己的坐标系;在 OpenGL 中,这是由特定的函数从你的坐标系转入屏幕坐标系。这就允许无论在二维空间($\mathbb{R}^2$)或三维空间($\mathbb{R}^3$)定位一些点,并让 OpenGL 自动地映射这些点到图形图像中适当的位置。

在二维的 xy 平面(又称为$\mathbb{R}^2$)由指定它的 x 和 y 坐标设置一个位置。通常的约定 x 轴是水平的并指向右,y 轴是垂直的并指向上(见图 1.3)。

在三维空间$\mathbb{R}^3$,由给定一个点的 x、y 和 z 坐标的三元组(a,b,c)指定其位置。当然,关于如何定位三个坐标轴的约定,对于计算机图形学,则与通常在教学中所用的不同。在计算机图形学中,x 轴指向右,y 轴指向上,z 轴指向观察者。这是不同于我们习惯性的期望。例如,在数学中,x 轴、y 轴和 z 轴通常分别指向前、指向右和指向上。计算机图形学中的采用约定,推测其原因是它保持 x 轴和 y 轴与 xy 平面相同的位置,但它有缺点。图 1.4显示坐标轴的方向。

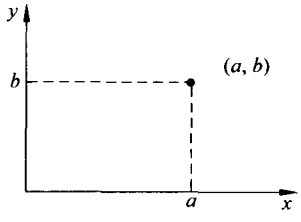


图 1.3 xy 平面(R^2)和点(a,b)

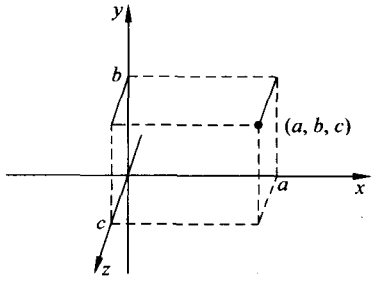


图 1.4 在 R^3的坐标轴和点(a,b,c),z 轴指向观察者

要注意到很重要的是,用在计算机图形学中的坐标轴做成右手坐标系形式。这意味着如果你摆布你的右手,使你的拇指和食指展开成 L 形状,并将你的右手放在这样位置:你的右拇指指向 x 轴的正向,食指指向 y 轴的正向,那么你的手掌向 z 轴的正向。实际上,这就意味着右手法则应用在$\mathbb{R}^3$空间做矢量叉积的结果。

1.2.2 在 OpenGL 中几何形体

下面我们讨论用 OpenGL 画点、线及多边形。我们只给出一些 OpenGL 中有效命令的通用版本。更完整的信息,你应查阅 OpenGL 编程手册(Woo et al.,1999)。

用 OpenGL 画点

OpenGL 有一些命令定义一个点的位置。用这些命令的两种常见方法是①：

```
glVertex3f(float x,float y,float z);
```

或

```
float v[3]={x,y,z};
glVertex3fv(&v[0]);
```

第一种格式命令 glVertex3f，以一组它的 x，y 和 z 坐标直接指定该点。第二种格式命令 glVertex3fv，需要一个指针指向一个含有坐标的数组。位于函数名结尾的“v”为表示“vector”。还有多种其他可以替代使用 glVertex * 命令格式②。例如，为表示“float”的“f”，可由为表示“short integer”的“s”，为表示“integer”的“i”，或为表示“double”的“d”来替代③。

对于二维的应用，OpenGL 也允许以一组它的 x 和 y 坐标指定该点，按如下命令用法：

```
glVertex2f(float x,float y);
```

或

```
float v[2]={x,y};
glVertex2fv(&v[0]);
```

glVertex2f 等同 glVertex3f 但 z=0。

对 glVertex * 的所有调用都必须被夹在对 OpenGL 命令 glBegin 和 glEnd 的调用之间，例如，画出图 1.5 所示的三点，您应当用下列命令：

```
glBegin(GL_POINTS);
glVertex2f(1.0,1.0);
glVertex2f(2.0,1.0);
glVertex2f(2.0,2.0);
glEnd();
```

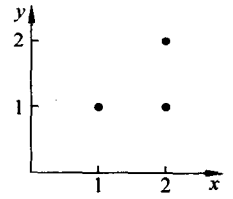


图 1.5　在二维中画三个点

函数 glBegin 和 glEnd 的调用作为画图开始和结束的信号。

一个简单的 OpenGL 程序 SimpleDraw 由此文本支持，文本含有为了画三个点的预

① 我们按简单约定描述 OpenGL 命令（通常不给正式正确的约定），在此情况下，标识符“float”说明 glVertex2f() 参数的类型，但在您的 C 或 C++ 代码应忽略之。

② 并没有函数名 glVertex *，我们用记号法共同地表示许多不同的 glVertex 命令。

③ 为了完全正确，我们将重新标志，有助于便捷性和今后兼容性。OpenGL 使用类型 GLfloat，GLshort，GLint 和 GLdouble，它们通常默认与 float，short，int double 是一样的。采用 OpenGL 的数据类型确实是更好的编程习惯。当然，额外的努力对于偶尔编程并没有实在价值。