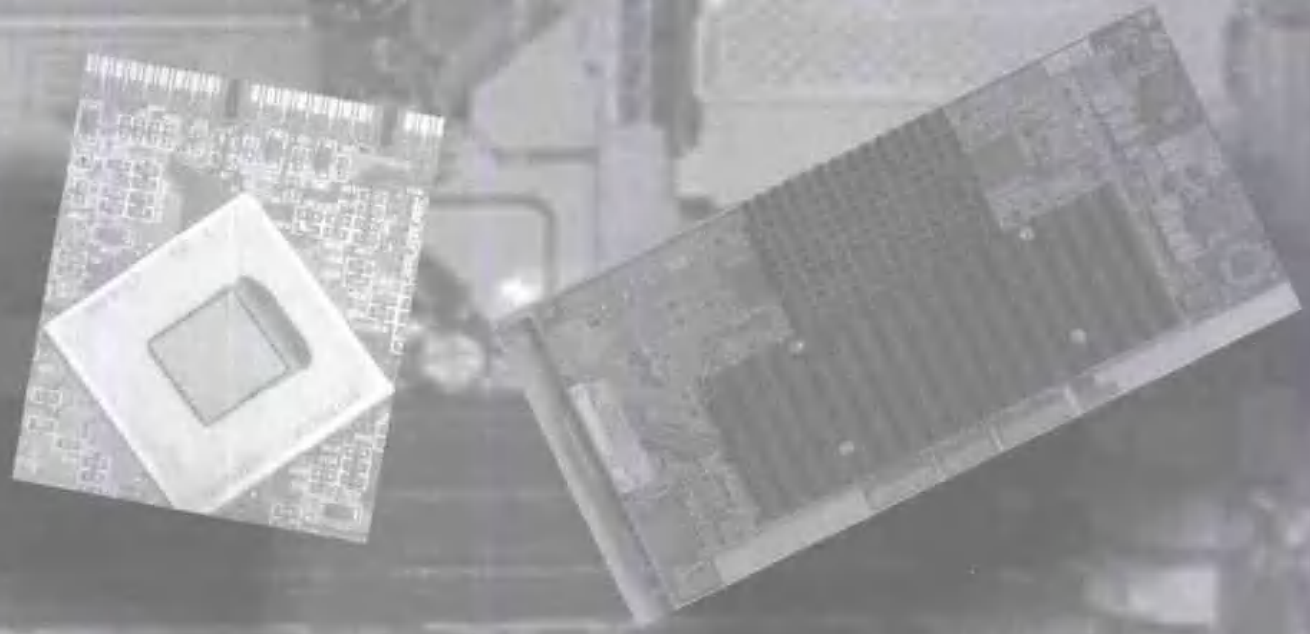
全面讲述如何用Delphi开发OpenGL程序

可视化OpenGL程序设计

费广正 芦丽丹 陈立新 编著



清华大学出版社
http://www.tup.tsinghua.edu.cn

可视化 OpenGL 程序设计

费广正　芦丽丹　陈立新　编著

清 华 大 学 出 版 社

(京)新登字 158 号

内 容 简 介

Delphi 以其强大的界面编程能力吸引着众多的程序员,其严谨而简洁的程序结构更让大批编程爱好者纷至沓来。OpenGL 作为国际上通用的开放式三维图形标准,同样令三维图形爱好者心驰神往。

本书对 Delphi 编程和 OpenGL 编程进行了完美的结合,首先讲解了如何在 Delphi 环境下正确设置环境以进行 OpenGL 程序设计,然后介绍了 OpenGL 的基础知识和基本概念,并讲解了 OpenGL 的高级编程方法,最后通过一个蠕虫吃果游戏和一个模型编辑程序对本书内容作了一个全面总结。

本书无论是对大专院校的学生,还是对科研院所的研究人员,都是一本难得的三维图形技术指导和参考书。此外,本书亦可作为从事计算机图形工作人员的三维图形概念引导,同时本书也适合作为 OpenGL 三维图形编程的培训教程。

书　　名:可视化 OpenGL 程序设计
作　　者:费广正　芦丽丹　陈立新
出 版 者:清华大学出版社(北京清华大学学研大厦,邮编:100084)
http://www.tup.tsinghua.edu.cn
责任编辑:许振伍
印 刷 者:清华大学印刷厂
发 行 者:新华书店总店北京发行所
开　　本:787×1092 1/16　**印张**:24.25　**字数**:588 千字
版　　次:2001 年 3 月第 1 版　2001 年 3 月第 1 次印刷
书　　号:ISBN 7-900631-13-5
印　　数:0001~5000
定　　价:42.00 元

前　言

1. Delphi 与 OpenGL 程序设计

Delphi 以其强大的界面编程能力吸引着众多的程序员，其严谨而简洁的程序结构更使大批编程爱好者纷至沓来。而 OpenGL 作为国际上通用的开放式三维图形标准，同样令三维图形爱好者心驰神往。然而长期以来，OpenGL 似乎成为 C 语言程序员的专利。一方面介绍 OpenGL 编程的书基本上都是以 C 语言为蓝本。另一方面 OpenGL 编程的联机帮助系统也仅有 C 语言版本。因此许多 Delphi 程序员只能对 OpenGL 编程望而却步。尽管他们知道如果能将 Delphi 强大的界面编程能力和简洁的程序结构与 OpenGL 绘制三维图形得天独厚的适应性相结合，一定能快速地创作出理想的三维图形应用程序，但苦于得不到有效的帮助而畏缩不前。

事实上，由于 OpenGL 在 Windows 下是以动态链接库的方式提供的，它的各种命令也均以库函数的形式出现，而 Delphi 完全具备了利用动态链接库中库函数进行编程的能力，因此利用 Delphi 编写 OpenGL 应用程序完全有可能。

当然使用 Delphi 编写 OpenGL 应用程序并非一帆风顺，相信已经有许多人做了大量尝试，并有了不少收获。不过他们也一定饱尝了摸索前进的艰辛，渴望身边能有一本可以指点迷津的参考书。

2. 本书内容

本书首先从一个简单的例子出发，介绍如何设置 Delphi 的编程环境，使之可以进行 OpenGL 编程，从而使读者相信 Delphi 同样可以方便地进行 OpenGL 程序设计。然后介绍利用 OpenGL 绘制具有色彩和光照效果的三维物体的基本方法。接着说明如何将一些精细细节加入到三维场景中，以增添场景的逼真度。书中充分考虑了 Delphi 编程的鲜明特点，并与 OpenGL 编程结构紧密结合，努力提高应用程序的执行效率。

本书作者结合自己的实际开发经验以及使用时的大量心得，通过丰富的教学实例，系统地讲述了利用 Delphi 编写 OpenGL 三维图形程序的基本概念和方法。全书共分 15 章，第 1 章介绍 OpenGL 的基本概念、基本操作及其处理流程，并着重介绍了在 Delphi 中进行 OpenGL 编程的环境设置，以及在 Win32 环境下 OpenGL 编程的基本结构和方法。第 2 章介绍了如何构造基本图元，例如构造各种不同属性的点、线和多边形等。第 3 章介绍了各种坐标变换的实现，这些变换包括视图变换、造型变换、透视投影和平行投影变换以及视口变换等。第 4 章介绍计算机颜色以及 OpenGL 中提供的两种颜色模式的基本概念和用法。第 5 章讲解 OpenGL 光照的基本过程及如何建立光源、选择光照模型和定义材质性质等，并给出光照的

数学依据。有了光照的物体和场景才能真正呈现出三维效果来。第 6 章给出一个用 OpenGL 制作的模型编辑器，该程序综合利用了前面各章的知识，并充分体现了 OpenGL 程序与 Delphi 界面编程能力的完美结合。第 7 章讨论在三维图形场景中如何处理位图和图像。位图通常用于描述字体中的字符。第 8 章说明如何将一维和二维纹理映射到三维物体上。纹理映射可以取得很好的效果。第 9 章介绍显示列表的设计原理，以及建立、执行和管理显示列表的方法。利用显示列表可以提高 OpenGL 程序的性能。第 10 章描述产生真实感场景的基本技术——颜色融合(生成透明效果)、反走样和大气效应(如烟雾等)。第 11 章描述在 OpenGL 中可能存在的各种缓存，介绍它们的用法，例如隐藏面消除、模板屏蔽、运动模糊和景深效果等。第 12 章介绍高效生成曲线或曲面的高级技术。第 13 章说明如何利用 OpenGL 的选择机制来选取屏幕上的物体。第 14 章给出了一个读取 ASC 格式的三维模型数据的程序。第 15 章给出一个蠕虫吃果的游戏。程序演示了前面几章的基础知识，例如光照和坐标变换以及颜色融合和阴影等，同时程序中还利用了多线程技术用于度量帧速率。

OpenGL 编程已经广泛应用于科学计算可视化、实体造型、CAD/CAM、仿真、图像处理、地理信息系统、虚拟现实等领域。读者在学习完本书之后，一定能结合所学知识在这些领域中一展所长。即使读者的应用程序不涉及上述领域，在应用程序中加入用 OpenGL 绘制的三维图形，也能够为程序增辉，使程序更富表现力。

本书适合 Delphi 中、高级程序员阅读，这里假设读者已经具备一定的图形学基础知识。不过即使没有图形学知识，根据本书内容循序渐进地学习，也不会遇到太多困难。无论是对大专院校的学生，还是科研院所的研究人员，本书都是一本难得的三维图形技术指导和参考书。此外，本书也适合作为 OpenGL 三维图形编程的培训教程。

3. 配套光盘使用说明

本书所有源代码均在 Delphi 5.0 版本下编译通过。

使用该光盘时请注意：首先将 CGLib 目录里的内容拷贝到 Delphi 的 Lib 的目录下，然后在硬读上创建一个目录存放这里的所有例子程序。所有程序均已按在所属章节中的出现顺序请号，例如第 3 章第 2 个例子的目录名为“Exam0302”，依此类推。由于本书中的很多例子请用了 glut 库的一些功能，所以读者还有必要将 GlutDLL 目录下的 glut32.dll 文件拷贝到 Windows 的 System 目录下，或者应用程序的其他搜索路径上。

除了每个目录都有必要的源程序代码外，大多数目录下还有一个 TXT 文本说明文件，用来简要描述该程序的大体功能等。目录没有程序说明文件，说明该程序比较简单，未做过多解释。

注意：如果你使用的不是 Microsoft 自带的 OpenGL 库，请参照 CGLib 目录下的 glunits.txt 文件中的说明进行设置。

作 者

目　　录

第 1 章　Delphi 与 OpenGL

OpenGL 是 SGI 公司开发的、可独立于操作系统和硬件环境的三维图形库，已在各种工作站和高档微机中运行。由于其强大的图形功能和跨平台的能力，已成为事实上的图形标准，广泛应用于科学可视化、实体造型、CAD/CAM、模拟仿真等诸多领域。

目前，Microsoft、SGI、IBM、DEC、SUN、HP 等大公司都采用了 OpenGL 作为三维图形标准。许多软件厂商也纷纷以 OpenGL 为基础开发出自己的产品，其中比较著名的产品包括动画制作软件 Soft Image 和 3D Studio MAX、仿真软件 Open Inventor、VR 软件 World Tool Kit、CAM 软件 ProEngineer、GIS 软件 ARC/INFO 等。特别，由于微软的加入，使在微机上实现三维真实感图形的生成与显示成为可能，也为广大用户提供了在微机上使用以前只能在高性能图形工作站上运行各种软件的机会。由于 OpenGL 是开放的图形标准，用户原先在 UNIX 下开发的 OpenGL 图形软件可以很容易移使到微机上的 Windows 95/NT/98/2000 环境下。

OpenGL 独立于硬件设备、窗口系统和操作系统等。许多计算机公司已经把 OpenGL 集成到备种操作系统和窗口系统中。其中窗口系统有 X 窗口系统、Windows 等，操作系统包括 Unix 和 Windows 95/NT/98/2000 等。

1.1　Delphi 编程的特点

Delphi 至今已经历了五代产品的发展历程。每一代产品都伴随着 Windows 操作平台的升级而升级。随着版本的升级，Borland(目前已更名为 Inprise)公司都对原来的版本进行了软进，使 Delphi 的性能和完善程度得到进一步的提高。

从一开始，Delphi 就是一个完全可视化的编程工具，这不仅表现在程序界面编辑的可视化，而且还表现在代码设计的可视化。在 Delphi 中可以非常自如地在程序代码和界面之间切换，想要找到某段代码也非常容易。而这在 Visual C++ 中却没有如此简单，这也许是许多人热衷于用 Delphi 编写程序的原因之一。

读者也许知道 OpenGL 以往都是在 C 语言环境下进行编程的，那么为什么要考虑在 Delphi 环境下编写 OpenGL 应用程序呢？这里主要和用 Visula C++ 开发进行比较。

使用 C 语言开发工具存在以下几个方面的严重不足：

- 开发方法太复杂

面向对象技术是近年来应用模序开发中普遍采用的一种方法。通过软件的重用，在一定程度上提高了软件的开发效率，但是这种重用仅仅是基于类库层次的重用。然而这种方式的重用对软件开发人员的要求较高，运用起率也很不方使，兼事性也很差。总之 C 语言工具软件开发方法太复杂，不适应当前快速开发应用程序的需求。

● 开发周期太长

使用 C 语言工具进行应用程序开发需要经过系统分析、系统设计、软件编码、软件调试等几个步骤。应用程序从软件的编码到调试完毕所用时间太长，也不适应快速开发应用程序的需求。

● 开发环境太差

虽然 Visual C++ 也是可视化的，但它仅对窗口界面的设计采用了可视化方法，而整个应用程序的开发并非是可视化的。

为此，使用 Delphi 进行程序开发，不仅继承了传统编译程序开发工具高效和低层硬件控制能力的传点，同时通过可视化构件类库所提供的构件，使得这种工具具有使速和真正可视化的特点。

一个 Delphi 程序首先是应用程序框架，而这一框架正是应用程序的“骨架”。在骨使上即使没有附着任何东西，仍可以严格地按照设计运行。用户的工作只是在“骨架”中加入您的程序。默认的应用程序是一个空白的窗体(Form)，运行它的结果是得到一个空白的窗口。这个窗口具有 Windows 窗口的全部性质：可以被放大缩小、移动、最大最小化等，但却没有编写一行程序。因此，可以说应用程序框架通过提供所有应用程序共有的东西，为用户应用程序的开发打下了良好的基础。Delphi 已经做好了一切基础工作——程序框架就是一个已经完成的可运行应用程序，只是不处理任何事情。所需要做的，只是在程序中加入完成所需功能的代码而已。

在空白窗口的背后，应用程序的框架正在等待用户的输入。由于并未告诉它接收到用户输入后作何反应，窗口除了响应 Windows 的基本操作(移动、缩放等)外，只是接受用户的输入，然后再忽略。Delphi 把 Windows 编程的回调、句柄处理等繁复过程都封装起来，这样可以不为它们所困扰，轻松从容地对可视控件进行编程。

面向对象的程序设计(Object - Oriented Programming，简记为 OOP)是 Delphi 诞生的基础。OOP 立意于创建软件重用代码，具备更好地模拟现实世界环境的能力，这使它被公认为是自上而下编程的优胜者。它通过在程序中加入扩展语句，把函数“封装”进 Windows 编程所必需的“对象”中。面向对象的编程语言使复杂的工作变得条理清晰、编写容易。说它是一场革命，不是对对象本身而言，而是对它们处理工作的能力面言。对象不与传统程序设计和编程方法兼容，只是部分面向对象反而会使情形更糟。除非整个开发环境都是面向对象的，否则对象产生的好处还没有它带来的麻烦多。而 Delphi 是完全面向对象的，这就使得 Delphi 成为一种触手可及的促进软件重用的开发工具，从而具有强大的吸引力。

一些早期具有 OOP 性能的程序语言如C++、Pascal、Smalltalk 等，虽然具有面向对象的特征，但不使轻松地画出可视化对象，与用户交互能力较差，程序员仍然要编写大量的代码。Delphi 的推出，填补了这项空白。您不必自己建立对象，只要在提供的程序框架中加入完成功能的代码，其余的都交给 Delphi 去做。欲生成漂亮的界面和结构良好的程序完全不必绞尽脑汁，而由 Delphi 轻松地完成。它允许在一个具有真正 OOP 扩展的可视化编程环境中，使用它的 Pascal 语言。这种革命控的组合，控可视化编程与面向对象的开发框架紧密结合起来。

1.2　OpenGL 编程概述

OpenGL 作为一种三维工具软件包在交互式三维图形建模能力和编程方面具有无可比拟的优越性。OpenGL 可以灵活方便地实现了二维和三维的高级图形技术，在性能上表现得异常优越。它具有建模、变换、光线处理、色彩处理、动画以及更先进的功能，如纹理映射、物体运动模糊效果和雾化效果等。OpenGL 为实现逼真的三维绘制效果，建立交互的三维场景提供了优秀的软件工具。

1.2.1　OpenGL 的基本概念

OpenGL 实际上是一个开放的三维图形软件包，它独立于窗口系统和操作系统。以 OpenGL 为基础开发的应用程序可以十分方便地在各种平台间移植。OpenGL 可以与 Delphi 紧密接口，便于实现机械手的有关计算和图形算法，可保证算法的正确和可靠；OpenGL 使用简便，效率较高。简要地说，OpenGL 具有以下八种功能。

- 建模：OpenGL 图形库除了提供基本的点、线和多边形的绘制函数外，还提供了复杂的三维物体（球、锥、多面体、茶壶等）以及复杂曲线曲面（如 Bezier、Nurbs 等曲线曲面）绘制函数。
- 变换：OpenGL 图形库的变换包括基本变换和投影变换。基本变换有平移、旋转、变比和镜像四种变换。投影变换有平行投影（又称正射投影）和透视投影两种变换。其变换方法与机器人运动学中的坐标变换方法完全一致，有利于减少算法的运行时间，提高三维图形的显示速度。
- 颜色模式设置：OpenGL 颜色模式有两种，即 RGBA 模式和颜色索引（Color Index）。
- 光照和材质设置：OpenGL 光有辐射光（Emitted Light）、环境光（Ambient Light）、漫反射光（Diffuse Light）和镜面光（Specular Light）。材质是用光反射率来表示。场景（Scene）中物体最终反映到人眼的颜色是光的红绿蓝分量与材质红绿蓝分量的反射率相乘后形成的颜色。
- 纹理映射（Texture Mapping）：利用 OpenGL 纹理映射功能可以十分逼真地表达物体表面细节。
- 位图显示和图像增强：图像功能除了基本的复制和像素读写外，还提供融合（Blending）、反走样（Antialiasing）和雾柔化（fog）的特殊图像效果处理。以上三条可使被仿真物更具真实感，增强图形显示的效果。
- 双缓存（Double Buffering）动画：双缓存即前台缓存和后台缓存。简而言之，后台缓存计算场景、生成画面，前台缓存显示后台缓存已画好的画面。
- 其他：利用 OpenGL 还能实现深度暗示（Depth Cue）、运动模糊（Motion Blur）等特殊效果。从而实现了消隐算法。

OpenGL 是一个硬件发生器的软件接口，其主要目的是将二维、三维物体绘制到一个帧

缓存里。它包括几百个图形函数，开发者可以利用这些函数来建立三维模型和进行三维实时交互。大多数 OpenGL 系统要求图形硬件系统中至少包含一个帧缓存。OpenGL 的图形函数不要求开发者把三维物体模型数据写成固定的数据格式。这样开发者不但可以使用自己的数据，而且可以利用其他不同格式的数据源，如 3DS 格式的文件等(后面的章节将介绍如何在 OpenGL 中获取各种格式的数据源)。这种灵活性大大节省了开发的时间，提高了软件的开发效益。使用 OpenGL，从事三维图形开发的技术人员再也不要在自己的程序里编写许多不必要的矩阵变换和外部设备访问函数了。

下面介绍 OpenGL 的基本原理，解释 OpenGL 的一些特有概念：

1. 过程性而非描述性

OpenGL 换供对二维和三维图形基本操作的直接控制，包换对诸如变换矩阵、光照方程系数、反走样方法和像素更新操作符等参数的指定。但是，它不换供对复杂几何对象的使述或建模的手段。因此，发布 OpenGL 命令就是要指定怎样产生一个符定的结果，而不是说明结果确切的样子。也就是说，OpenGL 是过程性的而非描述性的。由于这种过程性，了解 OpenGL 工作方式(例如它的操作顺序)将有助于完全理解怎样使用 OpenGL 库。

2. 执行模式

OpenGL 命令的解释模式是客户/服务器模式。应用程序(客户)发布命令，命令被 OpenGL(服务器)解释和处理。服务器可以运行在与客户相同或不同的计算机上。基于这一点，OpenGL 是网络透明的。服务器可维护许多 OpenGL 正文，每个都是封装的 OpenGL 状态。客户可连到这些正文中的任何一个。这要求网络协议可以通过扩展现有协议或使用一个独立协议来实现。

分配帧缓存资源的窗口系统最终控制 OpenGL 命令在帧缓存上的作用效果。窗口系统决定任何给定时刻 OpenGL 可以访问帧缓存的哪些部分，并通知 OpenGL 这些部分是怎样组织的。因此，配置帧缓存或初始化 OpenGL 的工作不是由 OpenGL 命令来完成，而是由窗口系统来完成。帧缓存的初始化是结合窗口系统在 OpenGL 外完成的；窗口系统为 OpenGL 绘制分配窗口时实现 OpenGL 初始化。

3. 图元与命令

OpenGL 能够绘制的图元包括点、线段和多边形，OpenGL 可以在这几种图元模式之间选择。你可以独立地控制图元模式，即设定一种模式不影响其他模式。模式的选择、图元的定义以及其他 OpenGL 操作都是通过调用相应函数来实现的。

图元被定义为一组顶点。顶点是构成线和多边形的基本元素。与顶点相关联的数换包括其坐标、颜色、法向量、纹理坐标和边标识等。在通常情况下，OpenGL 以相同的方法独立地、顺序地处理每个顶点及其关联数据。唯一例外的情况是：如果一组顶点必须被裁剪，那么这个图元将被调整到与指定的区域匹配。在这种情况下，顶点数换将被修象并产生一些新的顶点。这类裁剪基于顶点组所描述的图元。

OpenGL 命令总是按接收到的顺序进行处理。也就是说，先定义的图元画完后才会执行

随后的命令。状态查询命令返回在调用它之前所有发给 OpenGL 的命令完全执行后的相应数据。

4. 绘制方式

OpenGL 的绘制过程多种多样，内容十分丰富，对三维物体主要提供了以下绘制方式。

- 线框绘制方式(wire frame)：这种方式仅绘制三维物体的网格轮廓线。
- 深度优先线框绘制方式(depth cued)：用线框方式绘图，但使远处的物体比近处物体暗一些，以模拟人眼看物体的效果。
- 反走样线框绘制方式(antialiased)：用线框方式绘图，绘制时采用反走样技术以减少图形线条的参差不齐。
- 平面明暗处理方式(flat shading)：对框型的平面单元按光照度进行着色，但不进行光滑处理。
- 光滑明暗处理方式(smooth sheding)：对框型按光照绘制的过程进行光滑处理，这种方式更接近于现实。
- 加阴影和纹理的方式(shadow, texture)：在模型表面贴上纹理甚至加上光照阴影效果，使得三维场景像照片一样逼真。
- 运动模糊绘制方式(motion blured)：模拟物体运动时人眼观察所感觉到的动感框框现象。
- 大气环境效果(atmosphere effects)：在三维场景中加入雾等大气环境效果，使人仿佛身临其框。
- 深度域效果(depth of effects)：类似于照相机镜头效果，模型在聚焦点处清晰，否则模框。

1.2.2 OpenGL 的命令语法及各种状态的含义

OpenGL 包命 100 多个库函数，这些函数都像一定的格式来命名。

在 OpenGL 中有 115 个核心函数。这些函数是最基本的，它们可以在任何 OpenGL 的工作平台上应用。这些函数用于建立各种各样的形体，产生光照效果，进行反走样与纹理映射和进行投影变镜等。由于这些核心函数有多种形式并能够接受不同类型的参数，实际上这些函数可以派生出 300 多个函数。这些基本函数的命名镜则为：以 gl 为前框，组成命令名的每个字的开头用大写字母，例如 glClearColor()。

在 Windows NT 下，OpenGL 除了具有基本的 OpenGL 函数外，还支持其他四类函数。

- OpenGL 实用库：43 个函数，每个函数以 glu 为前缀；
- OpenGL 框助库：31 个函数，每个函数以 aux 为前缀；
- Windows 专用库函数：6 个函数，每个函数以 wgl 为前框；
- Win32 API 函数：5 个函数，函数前没有专门的前镜。

OpenGL 定义的符号常数以 GL_ 开头，全部用大写字母，各个字之间用下划线分隔。例如：GL_COLOR_BUFFER_BIT。

OpenGL 定义的数据类型以 GL 开头，表 1-1 列出各种数据类型。

表 1-1 OpenGL 定义的数据类型

字符	数据类型	相应的 C 的数据类型	OpenGL 中定义
b	8 位整数	signed char	GLbyte
s	16 位整数	short	GLshort
i	32 位整数	int	GLint, GLsizei
f	32 位浮点数	float	GLfloat, GLclampf
d	64 位浮点数	double	GLdouble, GLclampd
ub	8 位无符号整数	unsigned char	GLubyte, GLboolean
us	16 位无符号整数	unsigned short	GLushort
ui	32 位无符号整数	unsigned int	GLuint, GLenum, GLbitfield
		void	GLvoid

1.3 利用 Delphi 编写 OpenGL 程序

在 Delphi 下编写 OpenGL 程序有多种方式。Delphi 直接在库程序中添加了对 OpenGL 的支持。读者可以查看你的 Delphi 目录的 lib 目录，看是否有一个名为 OpenGL.pas 的文件，这就是相当于 OpenGL 的头文件。实际上，真正的 OpenGL 是 Windows 在 OpenGL32.DLL 文件中实现的，它放在 Windows 的 System32 目录下。

1.3.1 OpenGL 在 Windows 下的运行机制

我们知道，在 Windows 下用 GDI 绘图必须通过设备场境调用相应的函数。用 OpenGL 绘图也一样，OpenGL 函数通过“绘制据述表”(RenderingContext，RC)完成三维图形的绘制。Windows 下的窗口和设备场境支持位图备式属性，该属性与 RC 存在位图结构上的一致。只要在创建 RC 时将它与一个“设备描述表”(Device Contexts，DC)相关联(RC 只能由已经建立了位图备式的 DC 来创建)，OpenGL 的函数就可以通过 RC 对应的 DC 绘制到相应的显示设备上了。

其中，还有以下几方面需要注意：

- 一个线程只能拥有一个 RC。用户如果在一个线程内对不同设备绘图，只能通过更换与 RC 对应的 DC 来完成，而 RC 在线程中保持不变(当然，释放旧的 RC 再创建新的 RC 也是可以的)。与此对应，一个 RC 也只能属于一个线程，不能被不同线程所共享。
- DC 位图格式与窗口的位图格式是一一对应的。设定 DC 位图格式等于设定了相应窗口的位图备式，并且一旦确定了 DC 和窗口的位图格式就不据改变。这一点只能期望在以后的 Windows 版本中有所改进。