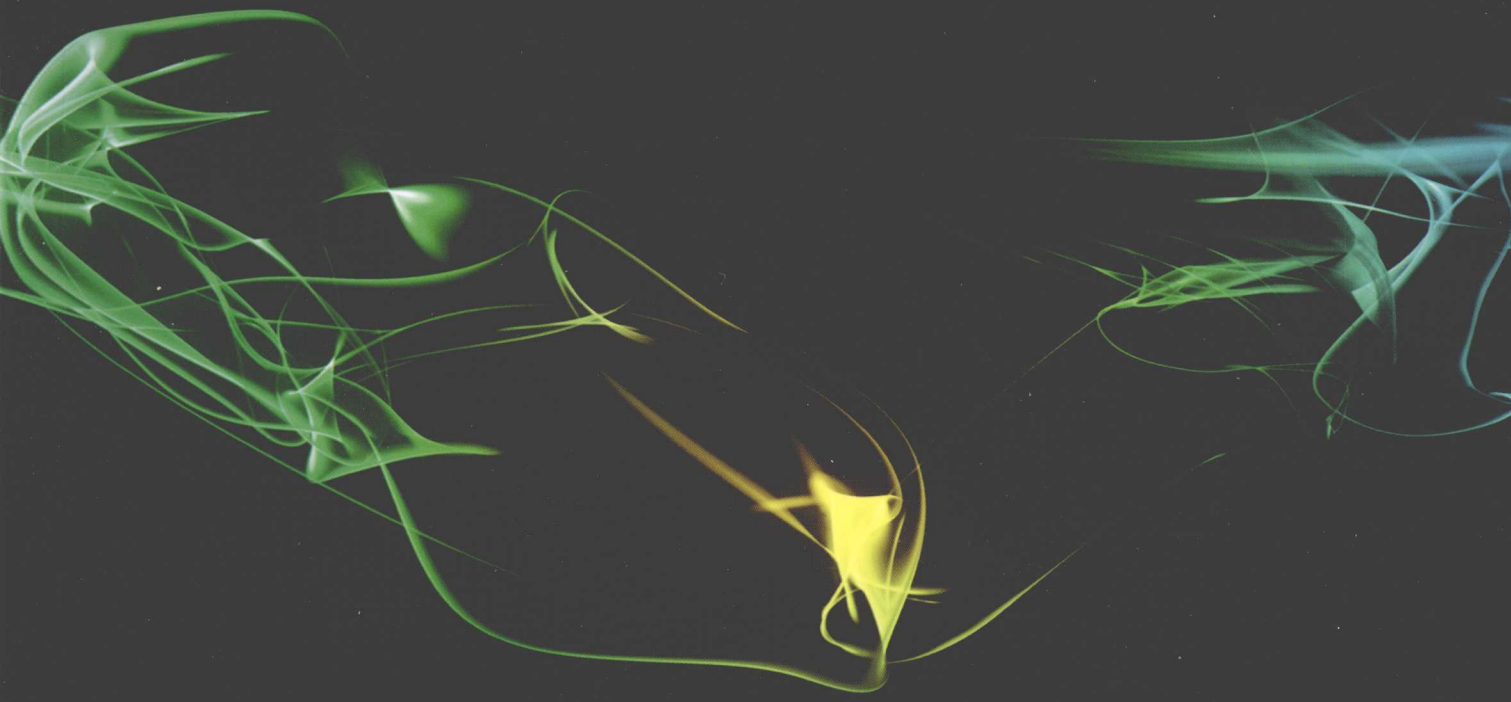
- 基于工业图形标准OpenGL的高层次三维渲染引擎。
- 深入讲解OpenSceneGraph渲染引擎的内部实现及其应用。
- 详细介绍OpenSceneGraph渲染引擎的组织架构及其实现流程。



OpenSceneGraph
三维渲染引擎设计与实践

王 锐 钱学雷 编著

- OpenGL架构评估编委会(ARB)独立撰稿人Paul Martz先生强力推荐。
- 拥有丰富的社区资源和强大的网络支持,以方便读者进一步的学习和交流。

清华大学出版社

OpenSceneGraph 三维渲染引擎设计与实践

王 锐 钱学雷 编著

清华大学出版社
北 京

内 容 简 介

OpenSceneGraph（OSG）是一个基于工业图形标准 OpenGL 的高层次图形开发 API 接口，一款开放源代码的、具备商业级别渲染能力的实时三维渲染引擎，在国内外均已得到广泛的应用，并且已经有越来越多的虚拟现实行业开发者加入到 OSG 开发的行列中来。

本书的编写目的是：详细剖析 OpenSceneGraph 引擎的实现流程，包括其场景图形结构，几何体绘制和渲染状态的封装机制，场景漫游、交互和动画的实现方式，以及最为重要的对于三维渲染引擎的内部裁减、数据动态调度和多线程渲染机制的深入分析。本书对虚拟现实行业的爱好者和从业者、对愿意了解最新图形学相关技术发展，以及有志于开发自主知识产权的三维引擎系统的读者，均会大有助益。

本书可以作为计算机图形学或虚拟现实专业的高年级本科生和研究生教材使用，也可供三维图形学领域的专门研发人员，尤其是使用 OpenSceneGraph 进行项目开发和科学研究的人员参考、学习。

图书在版编目（CIP）数据

OpenSceneGraph 三维渲染引擎设计与实践/王锐，钱学雷编著. —北京：清华大学出版社，2009.11

ISBN 978-7-302-21223-2

I. O…　II. ①王…　②钱…　III. 计算机图形学　IV. TP391.41

中国版本图书馆 CIP 数据核字（2009）第 173727 号

责任编辑：熊　健　纪文远
封面设计：刘　超
版式设计：杨　洋
责任校对：焦章英
责任印制：孟凡玉
出版发行：清华大学出版社　　**地　　址**：北京清华大学学研大厦 A 座
http://www.tup.com.cn　　**邮　　编**：100084
社　总　机：010-62770175　　**邮　购**：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质　量　反　馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 刷 者：清华大学印刷厂
装 订 者：北京市密云县京文制本装订厂
经　　销：全国新华书店
开　　本：203×260　**印　张**：24　**彩　插**：2　**字　数**：639 千字
版　　次：2009 年 11 月第 1 版　　**印　　次**：2009 年 11 月第1 次印刷
印　　数：1～4000
定　　价：45.00 元

Preface

OpenSceneGraph (OSG) is a high-level programming interface for 3D computer graphics, used in simulation, animation, and visualization applications. It's built on OpenGL, which ensures that OSG is both cross platform and high performance. But it goes beyond OpenGL to provide functionality common to many 3D applications, such as 2D and 3D file loaders, texture mapped font support, level of detail control, threaded database paging, and others. Because of its rich feature set and liberal open source license, OSG has become widely accepted by both academia and industry. It is used in hundreds of applications by thousands of developers worldwide.

OSG originated in the 1990's as a Linux-based scene graph support library. Much of what you see in core OSG today was written by Robert Osfield (Scotland, UK) about the year 2000. Over time, a large community of developers contributed many features to OSG. If you are new to OSG, you might be overwhelmed by OSG's size and capabilities, but you will also find many resources to assist you as you learn.

This book is one such resource. In 2006, I wrote and published the *OpenSceneGraph Quick Start Guide,* a short introduction to OSG, later translated into Chinese by Wang Rui with proofreading and editing by Dr. Qian Xuelei. In August 2009, I had the pleasure of meeting Wang Rui at the annual SIGGRAPH conference in the United States, where he presented on the growth and acceptance of OSG in China and discussed his progress with additional OSG documentation. You, the reader of this book, are about to benefit from their combined years of research and experience using OSG in China.

If OSG seems like unmapped territory, let this book be your navigational system, and you will certainly have a safe and successful journey.

September, 2009

序 言

Preface

OpenSceneGraph（OSG）是一个专为3D计算机图形开发而设计的高层次的可编程接口，广泛用于虚拟仿真、动画设计以及各种可视化程序中。OSG基于OpenGL进行构建，因此它同时具备了跨平台的特性和较高的渲染性能。此外，OSG还提供了一系列可供3D程序开发者使用的功能接口，包括2D和3D数据文件的加载、纹理字体支持、细节层次（LOD）控制、多线程数据分页处理等，而这些都是OpenGL本身所不具备的。凭借丰富的功能特性以及开放源代码的协议形式，OSG已经广泛地被学术界和产业界所认可。目前使用OSG进行开发的三维软件数以千计，而OSG的开发者团队也早已遍及世界各地。

OSG最早诞生于1990年，当时只是一个基于Linux开发的场景图形（Scene Graph）支持库。2000年以后，OSG的核心功能代码由来自苏格兰（Scotland，UK）的Robert Osfield先生主持开发。与此同时，一个庞大的开发者社区也在不断地为这款开源图形引擎做出贡献。如果您还是一名新近入门的OSG开发者，可能会迷失于OSG本身的繁多功能和过于复杂的结构中，但是您也会从中收获颇丰。

这本书就是其中的一例。早在2006年，由我负责编写和出版了《*OpenSceneGraph Quick Start Guide*》一书（中文译为《OpenSceneGraph快速入门指导》），而当时主持其中文版翻译和校对工作的，就是本书的作者王锐和钱学雷博士。2009年8月，在美国举办的SIGGRAPH大会上，我很荣幸地与王锐先生会面，聆听了其有关OSG在中国的发展与展望的演讲，并与他进一步讨论了OSG相关文档的编排和完善工作。在中国，他们有着多年研究和使用OSG的丰富经验。而现在，本书的读者，也就是您，将有机会受益于此。

如果把OSG比喻为一片尚未完全开拓的广阔疆域，那么这本书就是您的导航系统，盼望您能够即刻开始一段安心而完满的旅程。

Paul E. Martz
2009年9月

前　言

Preface

OpenSceneGraph（OSG）是一个基于工业图形标准 OpenGL 的高层次图形开发 API 接口，即三维渲染引擎。通过它，程序员能够更加快速、便捷地创建高性能、跨平台的交互式图形程序。其优点显而易见：除了开源和平台无关性以外，它封装并提供了数量众多的提升程序运行时性能的算法、动态数据分页机制，针对几乎所有主流数据格式的可扩展的读写接口，以及对其他语言系统（包括 Python、C#、Java）的封装支持。作为一款功能强大、具备工业应用水准的场景图形中间件（Middleware），OpenSceneGraph 已经成为虚拟仿真行业软件开发的首选之一，并且已在三维地理信息系统（GIS）、计算机辅助设计、科学和工程数据可视化、游戏和娱乐等多个行业中得到了广泛应用。

然而，作为一个开发者社区异常活跃、由来自全球各地的软件工程师参与贡献的开源引擎，OpenSceneGraph 相关参考文献过少的问题一直为人所诟病，也使得很多对虚拟现实开发感兴趣的爱好者和研究者不免望而却步，转而选择一些资料更为全面，甚至是“傻瓜化”的开发工具，放弃了深入研究三维渲染理论，深入了解现代图形引擎设计结构和思想的大好机会。

本书编写的目的，即是为了向广大三维行业的从业人员，以及有志于虚拟现实软件开发的学生和爱好者们，推荐 OpenSceneGraph——这个广受好评的、在全球图形引擎中永远占据一席之地的实时开源三维图形渲染引擎系统。同时，本书还将深入 OpenSceneGraph 引擎的内部实现，介绍场景图形理论及其应用、多种场景管理和渲染技术的实现，剖析现代多线程渲染的理论与编码机制，为广大读者了解和学习三维图形引擎，甚至开发出属于自己的图形引擎，提供第一手的技术资料。

目标读者群

本书专为准备开始学习 OSG 的三维图形学爱好者、计划或已经使用 OSG 进行程序开发的开发者或者有志于开发自主知识产权的三维渲染引擎的朋友而编写。

开始学习本书前，读者需要具备相当的图形学基础理论和 OpenGL 程序开发的知识储备，对于 OpenGL 开发的基础概念，例如顶点和顶点数组、渲染状态、纹理映射、显示列表、VBO 等，均应十分了解并可编写相应的 OpenGL 程序；同时，本书的读者还应当是一名熟练的 C++程序开发人员，对于 C++的各种特性，包括公有和私有成员、类的继承与重载、构造与析构、标准模板库 STL 等，都需要有相当的运用水平。

本书还将介绍多种现代图形学理论的概念和实现方法，包括多线程渲染技术、动态数据调度技术、纹理烘焙技术、场景裁剪技术、动画技术、模块化的插件设计技术等，对这些方面感兴趣的读者可能会因此受益。

本书的读者还应当具备一定的线性代数基础，对于空间坐标系的概念较为熟悉，了解向量、矩阵、四元数的基本概念，并可运用数学方法从本质上解决各种坐标空间变换的问题，例如在 OpenGL 中使用空间矩阵变换向量的方法等。

此外，如果读者对设计模式（Design Pattern）能有较深入的理解的话，对于本书内容以及 OSG 源代码的阅读将会大有好处。

本书的组织结构

本书共分 13 章，其主要内容如下。

第 1 章：介绍场景图形概念、OpenSceneGraph 引擎的诞生与发展及其主要组成结构。

第 2 章：介绍开源引擎 OpenSceneGraph 的源代码获取方法，并详细解释使用 CMake 工具编译生成引擎功能模块的过程。

第 3 章：介绍 OSG 开发的一些基础数据结构和机制实现。

第 4 章：介绍场景图形理论的实际实现，即场景组织和管理的方法，并剖析 OSG 引擎中的场景树结构实现流程。对于 OSG 引擎已有一定了解的朋友可以直接从本章开始阅读。

第 5 章：介绍几何体和文字绘制的基本方法，并详细解析 OSG 引擎中对于它们的封装和实现。

第 6 章：剖析 OpenGL 状态机的实现机制，以及 OSG 对于所有 OpenGL 渲染状态和模式开关的封装方式。

第 7 章：介绍三维场景到二维窗口的投影和映射过程，并专门解释 OSG 引擎中对这一流程的封装机制、渲染工作循环的建立，以及渲染到纹理（Render To Texture，纹理烘焙）技术的实现。

第 8 章：介绍三维人机交互的概念，以及 OSG 中图形设备抽象层的实现方法和应用。

第 9 章：介绍各种场景动画的实现机制和具体流程，详细解析关键帧动画、变形体动画、纹理动画、渐进动画和角色动画在 OSG 引擎中的实现和应用。

第 10 章：剖析 OSG 引擎中强大的模块化插件设计机制，以及它在多种文件格式的读写和转换方面的应用。

第 11 章：详细介绍 OSG 中的场景更新和裁减技术实现，以及成熟的大规模数据动态分页调度技术的实现，剖析 OSG 引擎内部“按状态排序渲染”技术的实现流程。

第 12 章：介绍场景多线程渲染的基本理论和概念，详细剖析 OSG 渲染后台的数据交换流程、OSG 目前支持的 4 种多线程渲染方法及其优劣。

第 13 章：提供大量基于 OSG 开发的、极具参考价值的第三方工程，以及开发者资源网站和部分案例图片，并对即将开始 OSG 学习征程的初学者提出一些善意的忠告。

格式约定

本书使用表格的方式列出 OSG 中最常用的功能类及其成员函数，重要的成员函数使用“★”符号进行标识。

本书中使用加粗字体来表达一些简短的示意命令或代码，它们往往是不能直接运行的，但是可以运用在用户自己编写的程序代码中。

本书中提供的范例完整程序代码使用带有灰色底纹的文本框与正文区分开来，并且标识其行号，便于读者的阅读。

本书中有关引擎内部实现的介绍均使用灰色底纹的文本框来表述，并使用伪代码介绍其执行流程。

本书中出现的成员函数和变量，以及全局枚举类型和常量，均可以直接在程序开发时使用。它们在文中均使用常规方式表达。

本书的资源站点

本书的网上资源站点为 http://code.google.com/p/osgenginebook/，您可以在这里免费浏览本书中介绍的所有示例程序的源代码，下载其最新版本。此外，由于本书在编写过程中难免会产生各种格式和概念上的错误，欢迎您随时与本书的作者进行交流，作者也会在这个站点上维护一个不断更新的勘误列表。

此外，您还可以注册并登录 OSG 中国论坛（网址：http://bbs.osgchina.org），发表您的意见与建议，并与更多的 OSG 爱好者进行深入的交流和讨论。

致谢

感谢 Don Burns 和 Robert Osfield，没有他们富有创造性的努力，就没有 OpenSceneGraph 这个出色引擎的诞生，也就不会有本书的面世。

感谢 372 名 OSG 引擎核心贡献者（包括笔者在内），感谢超过 1600 名来自全球各地的 OSG 开发者，以及超过 2000 名来自中国的 OSG 开发者和爱好者的不懈努力。没有你们，OSG 引擎的发展将停滞不前。

感谢 Skew Matrix 公司主席 Paul Martz 先生，作为全世界第一本 OSG 读物《OpenSceneGraph 快速入门指导》的作者，同时也是 OpenGL 架构评估编委会（ARB）的独立撰稿人，他在本书的编写过程中为笔者提供了大量有价值的参考意见，支持笔者在 SIGRGAPH 2009 OSG BOF 大会上发言介绍自己的微薄成绩，并且为本书作序。

感谢北京四维远见信息技术有限公司的全体同仁，尤其是三维软件项目组的魏占营老师、关艳玲老师、陈学霞老师和苏玉扬老师，他们为笔者的写作创造了巨大的便利和一切可能的支持。

感谢西安虹影科技有限公司（3DVRI）的朱幼虹老师、神州普惠公司的高峰老师、北京林业大学图形图像实验室，以及 osgChina 团队的杨石兴、肖鹏、贺思聪等朋友，为本书提供了第一手的技术资料。

感谢我的合作者钱学雷博士，在自身工作十分繁忙的情况下，依然抽出大量时间协助笔者完成了本书的第 1 章和第 13 章，并对全书的格式与用语规范进行了审校和修订。

感谢冷琴，在本书一度遇到创作瓶颈的时候，给我以巨大的支持和鼓励，并愿意静静地倾听我的满腹苦水。

最后还要感谢我的父母，没有你们的支持，我无法想象自己能够完成这本内容浩繁、创作难度极大的专业著作。

编　者

目 录

Contents

第 1 章

初识 OpenSceneGraph（OSG）

那些没有受过未知物折磨的人，
不知道什么是发现的快乐。
——克劳德·贝尔纳

本章将介绍虚拟现实和场景图形的概念，进而以此为基础，深入讲解三维渲染引擎 OpenSceneGraph（以下简称 OSG）的历史和组成，了解其性能优势和体系结构。通过对本章的学习，将有助于快速掌握虚拟现实技术、场景图形等基本概念，以及对日渐广受关注的三维实时渲染引擎 OSG 能够有一个初步的认识和了解。

1.1 场景图形初步

1.1.1 场景图形的概念

场景图形（Scene Graph）是一种经常用于计算机游戏和图形学相关软件的数据结构设计方法。比较典型的应用了场景图形概念的软件系统包括 AutoCAD、Java3D、CorelDRAW 以及本书将要详细讲解的 OpenSceneGraph 等。

场景图形的实现并没有一定之规，它可以用于二维或三维图形场景的逻辑关系和空间表达。最简单的场景图形实现是使用数组或者链表的结构，并按照固定的顺序依次绘制或操作这些节点。当然，有些操作可能会因此变得十分低效，例如判断鼠标是否选中了某个节点，此时应用程序同样不得不按照数组或链表的访问顺序依次对各个节点进行检查。

因此，适用于大规模场景管理的场景图形往往使用图结构（Graph Structure）或树结构（Tree Structure）来组织成一组节点集。目前多数渲染引擎均采用一种自顶向下的、分层的树状数据结构来组织空间数据集，以提升渲染的效率。

在场景树结构中，多个场景图形中的子节点往往只有一个父节点，并且可以继承父节点的多重特性；同时场景图形还提供了遍历所有子节点的访问方式。在有些系统中，子节点也可以拥有不止一个父节点，甚至可以与其他不属于同一结构的节点进行关联。与之相关的设计模式称为“组合”（Composite）模式，主要用于表达不同层次结构的分支节点（Group Node）和叶子节点（Leaf Node）。

1.1.2 具体实现：三维渲染引擎

所谓三维渲染引擎，简单地说，就是为了实现三维场景图形的结构管理和绘制而提供的一系列 API 的集合。它应当包含至少两个层次——构建层（Construction Layer）和交互层（Interaction Layer），前者提供了在三维空间中设计和完成所需模型的工具集，或者从外部加载复杂模型的数据接口；后者则提供对三维空间及所含模型的装配、渲染、优化和控制功能。

三维渲染引擎可以分为低阶引擎和高阶引擎两种。

（1）低阶引擎

OpenGL 是跨平台的、可用于图形软件系统开发的工业标准，也是一个典型的低阶引擎；另一个典型的例子是由 Microsoft 开发和维护的非跨平台的 DirectX 引擎。OpenGL 并没有为如何实现一个图形软件系统提供一个标准模型，而仅仅是一系列底层图形操作的接口，它具有跨平台的特性，并且提供了建立基本模型和渲染环境的工具集。

（2）高阶引擎

为了寻找一种基于 OpenGL/DirectX 且无须关注底层的实现细节的、面向对象的图形系统，一大批高层次的三维渲染引擎应运而生。其中大多数不仅支持面向对象的开发方式，并且以场景图形为基础实现了几何图形与动态行为的描述。

1.1.3　主流渲染引擎介绍

经过多年的研究和发展，基于 OpenGL 或 DirectX 实现的各种渲染引擎已经逐渐趋于成熟，一批优秀的开源和商业引擎也逐渐在市场上占据了主导地位，并越来越为企业和开发者、爱好者们所熟悉。下面介绍一些主流的三维渲染引擎及其主要特色。

（1）PHIGS

PHIGS（Programmer's Hierarchical Interactive Graphics System）是一个起源于 20 世纪 80 年代末的三维场景图形标准，其思想为大多数后来者所继承和借鉴，但是这个工具本身已经不再被使用。

（2）Open Inventor

1992 年，由 Silicon Graphics（SGI）开发的 IRIS Inventor（一个构架于当时的 IRIS GL 3D API 底层接口的场景图形系统）面世。1994 年，它的继任者，也就是现在仍然广泛使用的 Open Inventor 诞生了，这是一个基于 OpenGL 开发的、跨平台的、面向对象的场景图形系统。2000 年，Open Inventor 宣布遵循开源协议发布，至今仍然广泛运用于各种工程和科学可视化程序中。

相关网址：http://oss.sgi.com/projects/inventor/。

（3）OpenGL Performer

OpenGL Performer（早期称之为 IRIS Performer，或者简称为 Performer）是一种使用商业协议、基于 OpenGL 开发的实时虚拟仿真系统，同样由 SGI 负责开发和维护，并可运行于 IRIX、Linux 以及 Windows 系统中。和 Open Inventor 有所区别的是，1991 年诞生的 Performer 更专注于场景图形系统的重构和性能改善。

相关网址：http://www.sgi.com/products/software/performer/。

（4）Crystal Space

Crystal Space 是一个使用 C++开发的三维渲染引擎，最早发布于 1997 年。它主要用于游戏引擎的开发，但是也可用于其他类型的三维仿真程序设计。它可以运行于 Windows、UNIX/Linux 以及 Mac OS X 系统上，基于 LGPL 协议发布，并可选择多种类型的底层支持，包括 OpenGL、SDL、X11 接口或者 SVGALib。

相关网址：http://www.crystalspace3d.org/。

（5）Java3D

Java3D 于 1996 年第一次提出，并于 1998 年完成第一个版本；2008 年开始，Java3D 系统均基于 GPL 开源协议发布。它是一种基于场景图形概念的、多线程的、用于 Java 平台的三维场景渲染引擎，其底层基于 OpenGL 或 DirectX 进行构建。

相关地址：https://java3d.dev.java.net/。

（6）Unreal

Unreal（虚幻）商业引擎最早诞生于 Epic Games 在 1998 年开发的游戏 Unreal 中，并广泛应用于“虚幻竞技场”、“彩虹六号”等著名三维游戏中。它基于 C++进行开发，可以运行于 Windows、Linux、Mac OS X 等操作系统以及 Xbox、PS2、PS3 等游戏平台，底层基于 OpenGL 或 DirectX 进行构建。

相关网址：http://udn.epicgames.com/。

（7）OpenSG

OpenSG 是一个著名的实时场景图形系统，它遵循 LGPL 开源协议发布，基于 OpenGL 底层，并可运行于 Windows、Linux、Solaris 和 Mac OS X 等操作系统中。其主要特性包括多线程和排序渲染算法的设计。它诞生于 1999 年，多少继承了 Fahrenheit graphics API（Microsoft 与 SGI 合作的产物，但是已经终止）的特性。它与本书将要介绍的 OpenSceneGraph 是基于完全不同的设计思想开发的。

相关网址：http://opensg.vrsource.org/。

（8）OGRE 3D

OGRE（Object-Oriented Graphics Rendering Engine）是一个面向场景的、灵活的三维渲染引擎，诞生于1999年，至今其开发团队仍然十分活跃。OGRE 3D主要针对游戏开发，可以使用OpenGL或DirectX作为底层支持，并支持多种高级特性。OGRE 基于有改动的 LGPL 协议发布，并实现了场景八叉树、BSP 树、CLOD 以及分页调度的机制，可以运行于 Linux、Mac OS X 和 Windows 系统中。

相关网址：http://www.ogre3d.org/。

（9）Irrlicht

Irrlicht 源于一个德语词汇，诞生于 2003 年。它遵循 Irrlicht Engine License 协议，使用 C++语言进行开发、可以运行于 Linux，Mac OS X、Windows 和 Windows CE 等操作系统以及 Xbox，Symbian OS 等平台，并且支持与多种语言的绑定（包括.NET、Java、Perl 等）底层可以选择 OpenGL、DirectX 或者 OpenGL ES 接口。

相关网址：http://irrlicht.sourceforge.net/。

（10）Vega Prime

Vega Prime 是一个跨平台的、可扩展的三维渲染环境，包含了大量针对最终用户的解决方案和效果组件，可以大大降低三维系统开发的时间复杂度。

相关网址：http://www.presagis.com/products/visualization/vegaprime/。

（11）OpenSceneGraph

OpenSceneGraph 是一个高性能的开源三维图形引擎，基于修改的 LGPL 协议（OSGPL）免费发布，广泛应用于虚拟仿真、虚拟现实、科学和工程可视化等领域。它以 OpenGL 为底层平台，使用 C++编写而成，可运行于 Windows、UNIX/Linux、Mac OS X、IRIX、Solaris、HP-UX、AIX 和 FreeBSD 等操作系统。它诞生于 1998 年，系统架构思想起源于 OpenGL Performer；发展至今，其功能特性涵盖了大规模场景的分页支持，多线程、多显示的渲染，粒子系统与阴影，各种文件格式的支持，以及对于 Java、Perl、Python 等语言的封装等。

本书下面的内容将针对三维渲染引擎 OpenSceneGraph 的方方面面进行讲解。

1.2 OpenSceneGraph 概述

1.2.1 诞生与发展

1997 年，一个名叫 Don Burns 的软件工程师受雇于当时的 Silicon Graphics（SGI）公司，负责针对滑翔机飞行的虚拟仿真工作进行研究。他使用当时的 OpenGL Performer 系统，设计了一套广受好评

的滑翔仿真软件，并开始尝试在 Linux 中使用 Mesa3D 和 3dfx Voodoo 显示卡设备继续完善自己的仿真软件。

随着开发的深入，这一软件逐步壮大并开始支持 OpenGL。与此同时，Don 编写了一套简单的、类似于 Performer 的场景图形系统，命名为 SG。SG 强调朴素易用，并力求满足人们对场景图形设计的需求。

1998 年，Don 在一个滑翔爱好者的邮件组中遇到了 Robert Osfield，也就是目前 OpenSceneGraph 项目的主要负责人。当时 Robert 在苏格兰的油气公司工作，但对计算机图形学和可视化技术有着浓厚的兴趣。志趣相投的两人走到了一起，开始合作对 Don 的仿真软件进行改善。Robert 提议将 SG 作为独立的开源场景图形项目继续开发，并由自己担任项目主导；项目的名称改为 OpenSceneGraph，简称为 OSG；当时共有 9 人加入了 OSG 的用户邮件列表。

2000 年，Robert 离开了原来的工作单位，转而成为 OpenSceneGraph 的专业服务商，全职负责开发工作。那段时期，他设计并实现了 OSG 的许多核心功能，并且是在完全没有客户和薪酬的情况下完成了这项卓绝的工作。而 Don 来到了 Keyhole 数字地图公司（也就是后来的 Google Earth 部门），并于 2001 年辞职组建了自己的公司，继续从事 OSG 的开发工作。

第一届 OpenSceneGraph 爱好者会议（bird-of-a-feature）于 SIGGRAPH 2001 世界图形学大会期间举行，当时只有 12 人参加。与会者中包括了来自 Magic Earth 的代表，他们与 Don 和 Robert 讨论了开发的事宜，并成为 OSG 的第一个收费用户。

之后，OSG 开始以令人瞠目的速度飞快地发展，异常活跃的贡献者们为 OSG 提交了各种功能代码和插件，使得这个原本简单的渲染引擎逐步变得更加完善和无所不包，而其原有的精悍和结构清晰的特色依然得以保留。

如今，相当一部分高性能的软件已经使用了 OSG 来完成复杂场景的渲染工作。大部分基于 OSG 的软件开发更适用于可视化设计和工业仿真，这其中包括了地理信息系统（GIS）、计算机辅助设计（CAD）、建模和数字媒体创作（DCC），以及数据库开发、虚拟现实、动画、游戏和娱乐业等。

1.2.2　优势与不足

OSG 引擎由一系列图形学相关的功能模块组成，主要为图形图像应用程序的开发提供场景管理和图形渲染优化的功能。它使用可移植的 ANSI C++编写，并使用已成为工业标准的 OpenGL 底层渲染 API。OSG 具备跨平台的特性，可以运行在大多数类型的操作系统上，并使用抽象层的概念，使 OSG 的函数接口可以独立于用户的本地操作系统使用；但是 OSG 也包含了针对某些平台相关的支持代码。

OSG 遵循开源协议发布，其用户许可方式是一种修改过的 GNU 宽通用公共许可证（GNU Lesser General Public License，LGPL），称为 OSGPL。

OSG 主要具备以下一些优势：

- ☑ 快速开发。OSG 场景图形内核封装了几乎全部的 OpenGL 底层接口，并随时支持最新的扩展特性。应用程序的开发者可以将重心放在三维程序开发的实质性内容以及与各种场景对象交互的方法上，而不再过多关注底层的代码。
- ☑ 高品质。OSG 经历了许多开发成员的反复检查、测试和改善，直接参与 OSG 核心代码开发并有所贡献的开发人员目前已经接近 400 人，其中也包括笔者。

- ☑ 高性能。OSG 的核心代码支持多种场景裁剪技术（Culling）、细节层次节点（LOD）、渲染状态排序（State Sort）、顶点数组、显示列表、VBO、PBO、FBO、OpenGL 着色语言等；以及文字显示，粒子系统，阴影系统，雨、雪、火焰、烟雾等特效模拟，场景的动态调度，多线程渲染等各种机制。它们共同使 OSG 逐渐成为一个高性能的三维图形引擎。
- ☑ 高质量代码。要编写高质量的程序，开发者必须了解自己所用的引擎结构。如果引擎不开放源代码，那么与它相关的开发信息就被封闭起来，用户只能借助开发商的文档和客户支持来获得开发信息。开放源代码使得程序员可以检查和调试所用开发包的源代码，充分了解代码内部信息。
- ☑ 可扩展性。基于场景图形的扩展思想，OSG 提供了强大的可扩展能力，包括各种类型的扩展节点（NodeKits，节点工具箱）、扩展渲染属性、扩展回调、扩展交互事件处理器等，为用户的程序开发提供了灵活的支持能力。
- ☑ 可移植性。OSG 提供了 Windows、UNIX、Linux、Mac OS X、IRIX、Solaris、HP-UX、AIX 和 FreeBSD 系统的移植能力，基于 OSG 开发的程序只要经过一次编写，就可以编译并运行在这些平台之上，不需要关心更多的代码移植的细节。
- ☑ 低费用。开源意味着免费，使用和发布基于 OSG 开发的程序和软件是不需要额外许可费用的；基于 OSG 开发的程序也不需要再次开放自身的源代码。
- ☑ 没有知识产权问题。对于开源且易于所有人阅读的代码而言，不存在侵犯软件专利的可能性。

但是，OSG 目前也存在着诸多不足，例如参考文档较少、代码风格不统一、部分功能的实现过于臃肿、无法应用于实践等，这些都有待更多的开发者和贡献者去发现和完善。

1.3 OpenSceneGraph 的组成结构

1.3.1 核心结构

OSG 的功能类采用“命名空间+类名称”的形式来命名。命名空间（name space）的命名方式为：第一个单词小写，后继单词的首字母大写，例如 osg、osg Util、osgViewer 等；类的名称则采用每个单词首字母大写的组合形式，例如 MatrixTransform、NodeVisitor 等。功能类的成员函数使用小写字母开头，之后每个单词的首字母大写，例如 setMatrix()、setAttributeAndModes()等。如果试图描述某个功能类的成员函数则通常使用如下方式书写：osg::MatrixTransform::setMatrix()。

OSG 引擎的组成部分包括一系列动态链接库（.dll/.so）、插件（.dll/.so）、供开发者使用的静态链接库（.lib/.a）、头文件，以及可执行的工具程序和示例。按照其作用来划分，可以大致分为以下 5 种类型。

- ☑ OSG 核心库：提供了基本的场景图形和渲染功能，以及 3D 图形程序所需的某些特定功能实现。OSG 的核心库包括：
  - ➢ osg 库：其中包括构建场景图形的场景图形节点类、用作向量和矩阵运算的类；可绘制体和几何体类；用于描述和管理渲染状态的类；以及图形程序所需的典型功能类，例如命令行参数解析、简单动画路径以及错误和警告输出等。