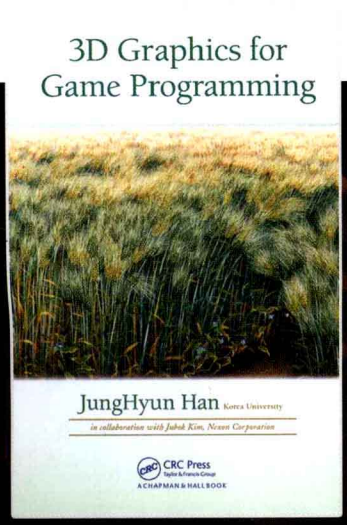
CRC Press
Taylor & Francis Group

3D Graphics for
Game Programming

# 计算机图形学
## —— 基于3D图形开发技术

JungHyun Han 著
刘鹏 译




清华大学出版社

# 计算机图形学

## ——基于3D图形开发技术

JungHyun Han 著

刘 鹏 译


清华大学出版社

北 京

## 内 容 简 介

本书详细阐述了与计算机游戏设计相关的基本解决方案及相应的数据结构和算法，主要包括游戏模型、顶点处理机制、光栅化操作、片元处理和输出合并、光照和着色、参数曲线和表面、着色器模型、图像纹理、凹凸贴图、高级纹理操作、角色动画以及物理模拟等内容。此外，本书还提供了相应代码以及伪代码，以帮助读者进一步理解相关方案的实现过程。

本书适合作为高等院校计算机及相关专业的教材，也可作为相关开发人员的自学教材和参考手册。



**图书在版编目（CIP）数据**

计算机图形学——基于3D图形开发技术/（韩）韩正贤著；刘鹏译. —北京：清华大学出版社，2013.1
（书名原文：3D Graphics for Game Programming）

ISBN 978-7-302-30183-7

I. ①计… II. ①韩… ②刘… III. ①三维-计算机图形学 IV. ①TP391.41

中国版本图书馆CIP数据核字（2012）第223321号

**责任编辑**：赵洛育
**封面设计**：刘　超
**版式设计**：文森时代
**责任校对**：柴　燕
**责任印制**：何　芊
**出版发行**：清华大学出版社
**网　　址**：http://www.tup.com.cn，http://www.wqbook.com
**地　　址**：北京清华大学学研大厦A座　　**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 装 者**：北京亿浓世纪彩色印刷有限公司
**经　　销**：全国新华书店
**开　　本**：185mm×230mm　　**印　　张**：19　　**字　　数**：378千字
**版　　次**：2013年1月第1版　　**印　　次**：2013年1月第1次印刷
**印　　数**：1～4000
**定　　价**：59.00元

产品编号：045527-01

# 前　言

大学的计算机图形学课程主要集中于动画和实时渲染，然而很少有教材可完美地体现交互式图形学特征，并在理论与实践方面做到较好的平衡且篇幅适中。本书涵盖了计算机图形学中的基础知识，可在一定程度上满足上述要求。本书适用于计算机图形学或游戏程序设计专业高年级本科课程以及初级研究生课程。

除此之外，本书还适用于游戏开发人员。此类人员具备图形API以及着色器编程经验，但缺乏一定的3D图形学背景知识。目前，市场上充斥着大量手册式的程序设计书籍。对于游戏开发人员而言，这一类书籍并未提供应有的数学知识。本书假设读者了解诸如向量和矩阵等较为基础的数学内容，进而将计算机图形学的基础理论和实践经验相结合。

作为交互式图形学的核心内容，GPU始终在不断地发展。为此，本书将围绕GPU程序设计进行组织。GPU可划分为可编程阶段和硬件阶段。针对可编程阶段，本书将阐述各种算法；对于硬件阶段，本书也将对必要知识加以分析。

本书的组织方式和讲述内容均经过精心设计，以使读者深入理解交互式图形学的核心内容。相关章节提供了大量的3D知识，以帮助读者快速地掌握相对复杂的知识点。同时，本书还在提示部分讲述了理论或技术细节，并涵盖了相应的可选章节（采用星号标记）以供读者阅读。当然，读者也可忽略这一部分内容，且不会对后续章节的内容产生任何影响。

Direct3D和OpenGL是两种较为知名的图形API，本书将会弱化API的归属特征，并在多处提供对应的示例代码，进而使读者深入理解交互式图形算法的实现方式。

本书辅助网站地址为http://media.korea.ac.kr/book，其中包含PowerPoint文件格式的教学备案以及视频剪辑在内的其他辅助资源。特别地，教学备案中包含了本书的全部图像。

## 致谢

本书源自Nexon Corporation和Korea Creative Content Agency所开发的项目，来自Nexon Corporation的众多人员均对本书的出版有所贡献。Jubok Kim在本书的筹备阶段即提出了许多中肯建议，并分别对本书的Alpha和Beta版本作了校对。Seungwon Han提供了

本书所需的3D模型以及其他艺术设计工作。另外，Hyunwoo Ki、Joongwon Gouk、Minsun Song也提供了颇具价值的图像。

同时，来自Korea University的相关人员也对本书的编写工作给予了大力的支持。例如，本书的视觉内容均来自Seungjik Lee，他在艺术设计和程序设计方面颇具天分。感谢Nguyen Trung Kien博士，在他的帮助之下，与纹理相关的3章内容得以顺利完成。Hanyoung Jang博士对着色模型以及物理模拟等相关章节也提出了宝贵的意见。在Kiwon Um的帮助下，笔者对全书章节进行了重新整理。同时，还要感谢来自3D Interactive Media Lab的学生们，他们是Dong-young Kim、Hyun Ju Shin、YoungBeom Kim、EunSeok Han、GwangHyun Park、Seungho Baek和Junhoo Park。本书内容源自Korea University和Carnegie Mellon University的授课内容，在此也要对学生们的反馈信息表示感谢。

Ewha Womans University的Young J. Kim教授、Soongsil University的Kyoung-Su教授、Kwangwoon University的Kang Hoon Lee教授、University of Florida的Jorg Peters教授以及Crytek的Dongwook Ha教授阅读了本书的手稿并提出了诸多宝贵的建议和意见，在此深表谢意。此外，还要感谢本书编辑——来自CRC Press的Li-Ming Leong，在他的大力支持下，本书方得以顺利出版。

最后，还要感谢Kyung-Ok、Jeehee和Jihoon，感谢你们的陪伴，我爱你们。

# 译 者 序

计算机图形技术以及计算机动画在视觉效果和动画制作方面曾引发了重大的技术变革，如今其前进的步伐仍未停止，数字技术进一步拓展了发展空间。

有人曾将游戏称为继绘画、雕刻、建筑、音乐、诗歌（文学）、舞蹈、戏剧、电影之后的人类历史上的第9种艺术，其发展前景由此可见一斑。对于当前的游戏来说，计算机图形技术和计算机动画在其中饰演了重要的角色。随着观众对视觉效果的不断追求，计算机游戏程序设计人员面临的挑战也越来越大，不仅需要掌握程序设计技巧和软件工程方法，还需要有坚实的专业领域知识。

针对这一问题，本书将详细阐述计算机游戏设计的基本方案及相应的数据结构和算法，主要包括游戏模型、顶点处理机制、光栅化操作、片元处理和输出合并、光照和着色、参数曲线和表面、着色器模型、图像纹理、凹凸贴图、高级纹理操作、角色动画以及物理模拟等内容。值得一提的是，本书并非是一本纯理论书籍，除了对相关内容进行了全面、系统的讲解以外，其设计思想、数据结构和算法均辅以对应的代码示例，以帮助读者进一步理解计算方案的实现过程。

在本书的翻译过程中，除刘鹏外，翁桂荣、康照丽、孙颖、万立梅、陈丽丽、龙清伟、郜利萍、高影、安娇娇、李霞、刘笑彤、周宏丽、魏晶、李芳芳、张凤欣、高阳、胡艳娇、商婵婵、武变琴、顾庆娟、蒋丽丽、张超、赵雅利、吕莹莹、贾亚飞和古庆梅等人也参与了部分翻译工作，在此一并表示感谢。

限于译者的水平，译文中难免出现错误和不妥之处，敬请广大读者批评指正。

译 者

# 目　录

# 第1章 游戏模型

游戏开发过程通常划分为3个阶段，即前期制作过程、制作过程以及后期制作过程。同时，各个阶段的特定步骤也会根据游戏的类型以及相关工作室不尽相同。总体而言，前期阶段包含游戏角色的绘制、编写故事大纲、创建故事板（故事大纲的可视化表现形式）以及撰写设计文档。游戏设计文档常被视为“设计图纸”并依此制作游戏、标识游戏目标和游戏规则、定义地图和关卡的设计规则以及屏幕和菜单的组织方式。

在游戏的制作阶段，由关卡设计师[1]、美工师、程序员以及音效师构成开发团队，并对制作总监负责。相应地，设计文档常在游戏制作阶段加以更新。例如，可添加或去除相关关卡。制作阶段的输出结果通常为游戏程序及其所需的大量数据（包括3D模型、图像以及动画数据）。

后期制作则为最终阶段。其中，制作阶段的输出内容交予游戏测试人员，测试人员返回产品缺陷以及错误报告。随后，程序员以及美工师对此进行必要的修复——该过程持续进行，直至游戏作品日臻完美。

本书主要讨论制作阶段中的3D图形学内容。3D游戏制作阶段的首要步骤便是建模操作，本章将对此加以讨论。

## 1.1 游戏制作流程

与游戏制作前期、制作后期相比，3D游戏的制作过程通常包含更为丰富的内容，如图1.1所示。此处，对应流程可称之为“管线”，即某一步骤的输出内容可作为下一步骤的输入内容。图1.1中的管线根据图形视口加以定义，因此图像美工师和程序员在此阶段内饰演了重要的角色。其中，美工师负责创建图形数据并管理建模操作以及部分动画操作；而程序员则负责管理另一部分动画操作以及渲染过程。大致而言，动画阶段可划分为离线任务和运行期任务，且分别由美工师和程序员加以掌控。

在建模步骤中，美工师将创建游戏场景内容。例如，考察一款室外设计游戏，游戏场景中常包含士兵、枪械以及地图等内容，如图1.2所示。相关对象多采用多边形进行建模，因而此类表现方式常被称为多边形网格，这也是最常见的一种游戏建模方案。

[1] 关卡设计的主要任务包括游戏地图的布局、设置游戏角色和对象（如敌方角色和障碍物）并定义其行为，进而与玩家或游戏事件实现互动。


图1.1　游戏制作流程中的3个主要步骤

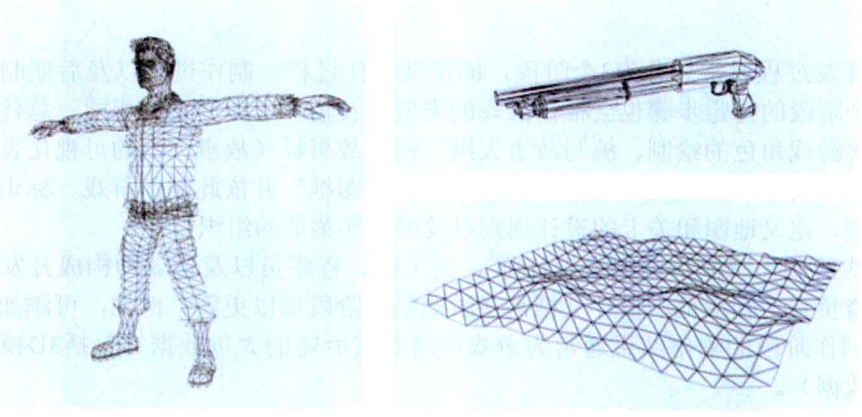


图1.2　以多边形网格显示的全3D游戏模型

需要说明的是，建模过程并不仅局限于3D模型的构造过程，通常还包括纹理创建操作（添加至3D模型之上），进而增加其视觉真实感。最简单的纹理形式为位图，可将其直接置于对象表面之上。例如，图1.3（a）展示了士兵模型对象的图像纹理构造过程，该纹理在运行期内应用于士兵对象的表面之上，对应结果如图1.3（b）所示。

（a）该纹理可视为小型图像集合，各幅图像覆盖士兵身体某一部分。初看之下，该纹理稍显奇特

（b）该纹理在运行期添加至士兵模型的多边形网格之上

图1.3　应用于多边形网格表面上的图像纹理

除此之外，士兵对象还应具备行走、奔跑以及匍匐前行等能力，因而需实现对应的动画效果。据此，应定义士兵对象的骨骼结构并于随后定义骨骼运动所导致的士兵对象多边形网格的变形效果。例如，股骨运动时大腿也将随之运动。该过程常被称为运动控制操作。图1.4显示了多边形模型中的骨骼结构。另外，运动模型常由美工师加以操控（对应的动画效果随后将在运行期内加以展现）。图1.5显示了线框模式下士兵对象的动画截图。

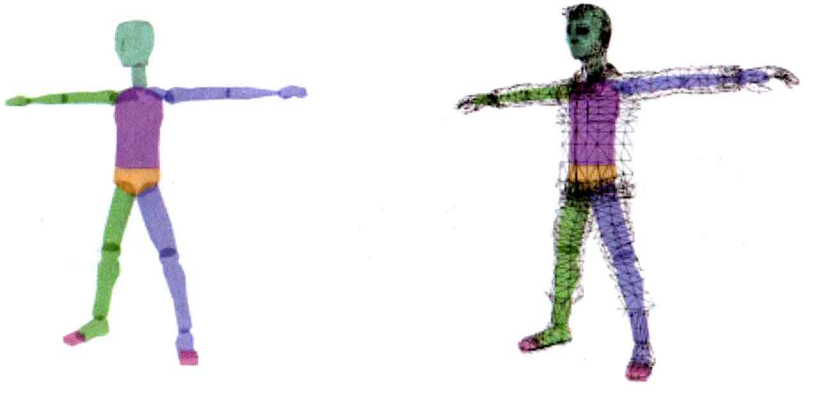


图1.4 躯体结构由骨骼构成并内置于多边形网格中。该图显示了类似于表面的骨骼表现方式，但骨骼并不存在显式的几何体表现方式，并采用矩阵体现其概念性实体结构（第11章将详细讨论这一问题）

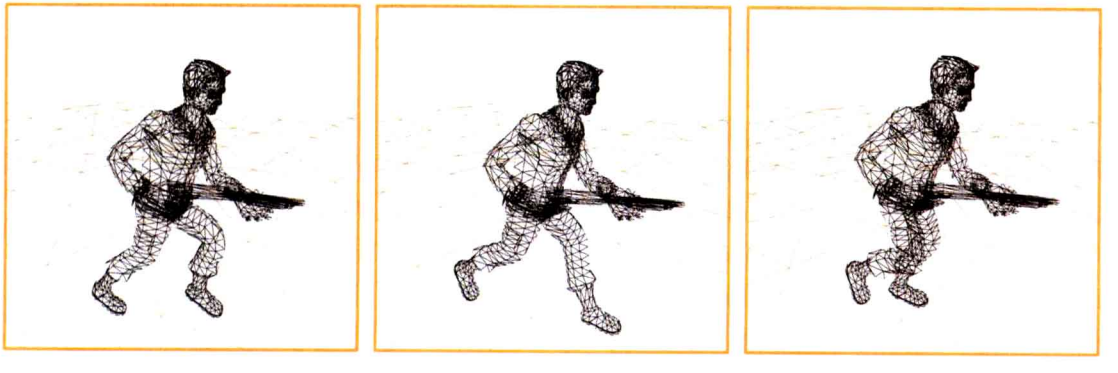


图1.5 通过控制骨骼结构可实现多边形网格的动画效果

在离线模式下，美工师负责执行建模以及动画制作任务，并常会使用Autodesk 3ds Max和Autodesk Maya等软件工具。针对美工师的作品，本书将采用3ds Max予以展示。

当快速显示处于变化状态的图像序列（称为帧）时，计算机游戏将在屏幕上生成处于

运动状态的视觉效果。当演示某一对象的动画效果时，将针对该对象所独有的形状和姿态（位置和方向）实施逐帧计算。除此之外，还将会进一步解决游戏对象间的外部作用力以及碰撞行为所导致的、运行期内的动力学问题。同时，涉及对象的状态将得到更新（该处理过程称为物理模拟，将在第12章中加以讨论）。进一步讲，光照条件以及视见条件也可在每一帧中发生变化。当上述参数确认无误后，即可调用渲染模块。渲染模块将根据3D场景生成2D图像且该图像将构成一帧。图1.6显示了动画场景的渲染结果。

图1.6　动画场景的渲染结果

与美工师所实现的建模和离线动画不同，运行期内的动画以及渲染过程则通过游戏程序得以执行。通常情况下，渲染操作依赖于图形API（应用程序接口），如Direct3D或OpenGL（参见“参考文献”中的第1、2项）。Direct3D是由微软公司推出的一种基于微软平台的DirectX API；OpenGL则由非盈利组织Khronos Group所掌控，是一种跨平台标准API。

图形API向程序员提供了必要的图形函数。当前，此类函数多在硬件中（即GPU，图形处理单元）加以实现，该硬件可视为一类专注于图形内容的处理器。图形API则定义为GPU的软件接口。其中，API可将图形命令转换为可执行于GPU之上的指令。

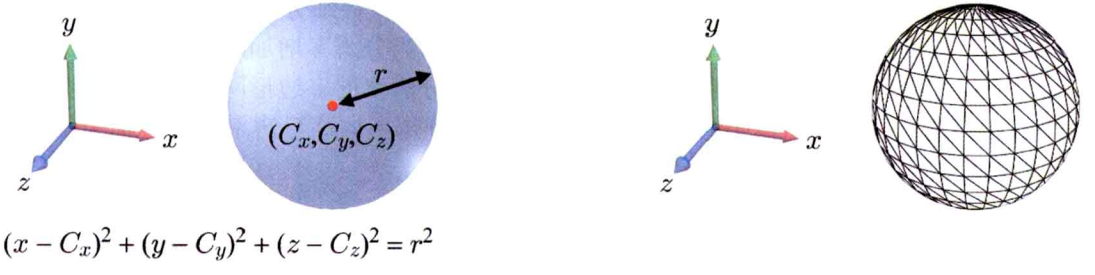
本书并非是Direct3D或OpenGL手册，但会包含某些与此相关的示例程序，以便帮助读者理解实时图形算法的实现过程。当采用API进行程序设计时，读者应翻阅相关SDK文档。例如，Direct3D SDK（参见“参考文献”中的第3项）针对初学者提供了相应的文档以及示例程序。

## 1.2　多边形网格

在3D计算机图形学中，可采用各种建模技术方案。例如，考察半径为$r$且位于

$(C_x,C_y,C_z)$处的某一球体对象，其相对简单的表达方式是使用球体方程，如图1.7（a）所示。该示例对象表示为基于隐式函数$f(x,y,z)=0$的隐式表面。当点$(x,y,z)$代入隐式函数中时，对应结果为0，则该点位于当前隐式表面之上。

当然，也可根据拓扑实体（如顶点）采用显式方式表达球体对象，如图1.7（b）中的多边形网格。考虑到GPU对多边形网格采取了优化处理，因而推荐使用该表现方式。需要注意的是，网格顶点可视为平滑表面的顶点采样操作，因而多边形网格仅为一类近似表达方案而非精准方案。




（a）隐式表面　　（b）多边形网格

图1.7　两种不同的球体表达方式

OpenGL支持具有任意数量顶点的通用多边形，但该多边形应定义为凸多边形。图1.8显示了凸多边形和凹多边形之间的对比结果。鉴于凹多边形处理算法的运算速度较慢，因而应对其进行必要的限制。除此之外，OpenGL还限定多边形应呈现为共面状态，即多边形顶点应位于同一平面之上。三角形可视为最简单的多边形，并包含凸性和共面性特征。Direct3D仅支持三角形，也就是说，Direct3D中的多边形网格为三角形网格。图1.9显示了三角形网格和四边形网格（简单地采用了正方形网格）之间的比较结果。显然，三角形网格更为常见。然而，正方形网格也存在一定的用武之地，尤其是在建模过程中，具体内容将在1.2.1小节中讨论。

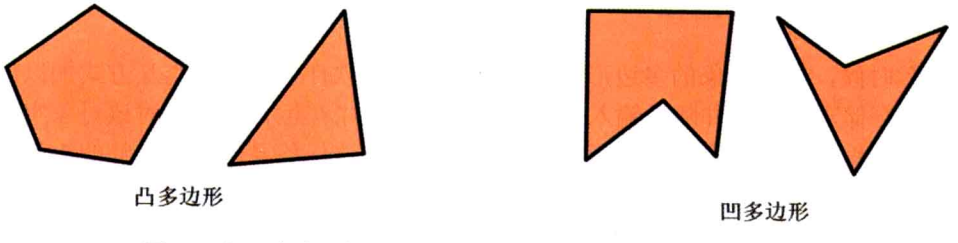



图1.8　与凸多边形相比，凹多边形难以处理，因而较少使用