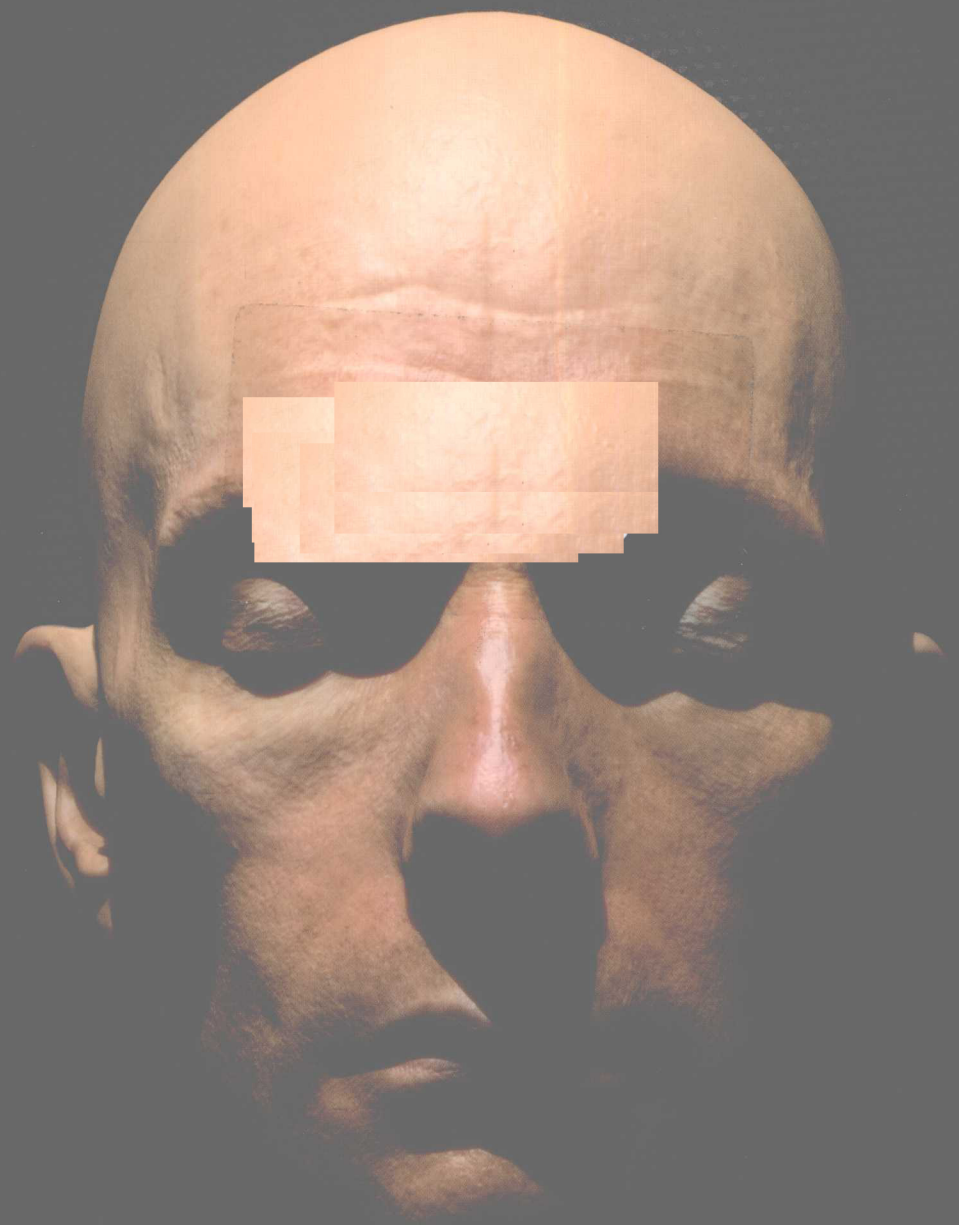
GPU Gems 3

GPU精粹3



(美) Hubert Nguyen 编著

杨柏林 陈根浪 王聪 译

NVIDIA

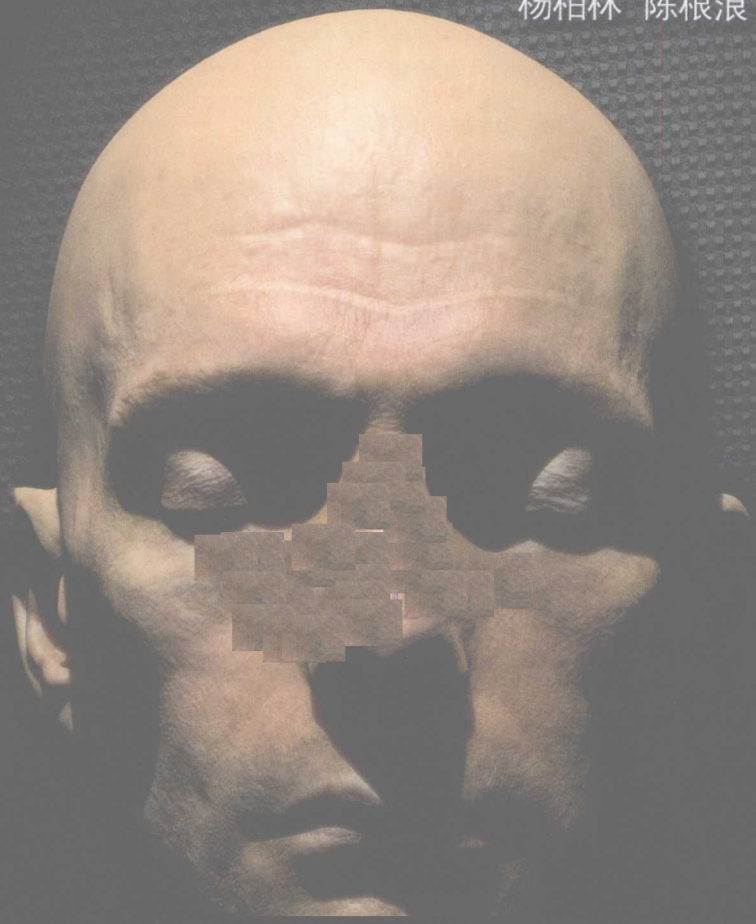
PEARSON Education

清华大学出版社

GPU精粹3

(美) Hubert Nguyen 编著

杨柏林 陈根浪 王聪 译



清华大学出版社

北 京

图书在版编目(CIP)数据

GPU 精粹 3/(美)阮亚(Nguyen, H.) 编著；杨柏林，陈根浪，王聪 译。—北京：清华大学出版社，2010.6
书名原文：GPU Gems 3
ISBN 978-7-302-22070-1

Ⅰ. ①G… Ⅱ. ①阮… ②杨… ③陈… ④王… Ⅲ. ①图像处理—程序设计 Ⅳ. ①TP391.41

中国版本图书馆 CIP 数据核字(2010)第 026816 号

责任编辑：王　军　李　阳
装帧设计：孔祥丰
责任校对：成凤进
责任印制：何　芊
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：北京嘉实印刷有限公司
装 订 者：三河市溧源装订厂
经　　销：全国新华书店
开　　本：185×260　**印　张**：44.5　**字　数**：1055 千字
附光盘 1 张
版　　次：2010 年 6 月第 1 版　　**印　　次**：2010 年 6 月第1 次印刷
印　　数：1～3000
定　　价：180.00 元

产品编号：027889-01

译者序

作为图形学领域近十年来最富有革命性和充满活力的图形处理技术，GPU 可谓包罗万象、博大精深。然而，尽管关于 GPU 技术的著作、论文、技术文档早已汗牛充栋、随处可见，但其中业界公认的最具价值的扛鼎之作却只有《GPU Gems》系列能算得上。自 NVIDIA 公司在成功推出《GPU Gems》系列的第一、二卷之后，经过主编精心挑选，内容更加全面、深入、准确、前沿的第三卷终于在 2007 年底出版了。

译者在浙江大学 CAD&CG 国家重点实验室攻读博士学位期间，就开始应用该技术完成了多个 3D 实时渲染相关的项目，所以一直关注着 GPU 技术的最新进展。后来，有幸受清华大学出版社委托翻译本书。在翻译过程中，译者深感本卷在前两卷的基础上又有了质的飞跃，体现了当前工业界与研究机构在 GPU 应用领域的最新进展和最高水平。

本卷的目的是让“软件开发和科学研究人员充分挖掘 GPU 的并行运算潜力”，即充分发挥 GPU 的通用计算能力。所以，本书所描述的技术除了能为游戏开发、影视特效、虚拟现实与增强现实、仿真系统、图像处理等传统的图形应用领域“创造极其逼真的人物角色、更好的光照效果和渲染合成效果”之外，而且还增强了对“金融模拟、金融分析，甚至病毒检测，尤其是 NVIDIA CUDA 编程架构”等需求更为广阔的非图形类领域的应用。

本书主要分为六大部分，分别是：几何体、光照和阴影、绘制、图像效果、物理仿真和 GPU 计算，涵盖了图形学领域的几乎所有应用方向，给出了当前 GPU 发展的最新技术，汇集了下一代 3D 引擎所需要的最关键的运算法则。图形领域中几乎所有知名企业和研究机构都参与了此书的编写和技术提供工作，可谓阵容强大，其中包括 NVIDIA、苹果、微软研究院、世嘉、Crytek、EA、Infinity Ward、康乃尔大学、伊利诺斯大学、英国达特茅斯学院、伦敦帝国理工学院、东京大学等。

可以说，无论您是 GPU 硬件编程发烧友、3D 网络游戏的开发者、虚拟仿真的设计者还是和从事三维图形研究工作的研究人员，此书都是您学习、开发和研究的必备宝典！相信您通过拜读此书，必定会对 GPU 的强大计算能力有更多惊奇的发现！我们衷心希望本书中文版的出版，能够推动国内 GPU 技术的应用发展。

本书共 41 章，其中正文中第 1 章、第 6～15 章、第 17 章、第 18 章、第 20～29 章、第 32～38 章由浙江工商大学计算机与信息工程学院杨柏林副教授翻译(共计 30 章)；第 2、3、4、5、16、19、30、31 章和第 39 章至第 41 章由浙江大学宁波理工学院陈根浪老师翻译(共计 11 章)；其余部分由王聪翻译。

此外，译者要感谢霍建同、姚佳立等同学，他们也为本书的翻译做出了很多工作。由于译者经验和水平有限，译文难免有不妥之处，敬请读者指正并与我交流：programming.game@gmail.com。最后，希望这本令我在翻译过程中获益匪浅的书也能给您带来美妙而独一无二的阅读享受！

杨柏林

2010 年 3 月 26 日于浙江工商大学信息楼

序

组合，也就是把一些基本操作组织成一个不明显的整体，这对命令式编程非常重要。微处理器的指令集体系结构(ISA)是一个多功能的组合接口，开发软件渲染器的程序员们在不断追求真实感图像的过程中，会高效并富于创造性地使用它们。早期的硬件图形设备增强了渲染的性能，但是在组合方面经常会付出高额的成本，其结果造成在可编程性和应用的创新方面也需要付出很高的代价。类微处理器的可编程硬件一直都在发展(如 Ikonas 图形系统)。但图形硬件加速的主要形式一直围绕着固定渲染操作序列展开，通常称为图形流水线(graphic pipeline)。这些系统的早期接口，如 CORE 和后来的 PHIGS，允许程序员指定渲染的结果，但不允许进行组合。

我有幸参与了 OpenGL 从 20 世纪 90 年代初期 Silicon Graphics 定义的前身 IRIS GL，到通过指定相应的体系结构(非正式名称为 OpenGL Machine)来解决组合需要的过程，这些指令是通过命令式编程接口来访问的。许多特征——例如，严格规定的语义；表驱动的操作，如模板和深度缓冲功能；使用通用一维、二维和三维查找函数实现的纹理映射；可重复性的要求——确保程序员可以使用 OpenGL 的各种操作，生成强大可靠的结果。OpenGL 支持一些有用的技术，包括基于纹理的体积渲染，使用模板缓冲区的阴影体积和构造立方体几何算法，如封盖(在裁剪平面和由多边形构造的实体相交处计算表平面)。后来，Mark Peercy 和他的合作者在 SIGGRAPH 2000 上发表的论文“交互式多通道可编程着色”阐释了任意的 RenderMan 着色器均可以通过组合 OpenGL 渲染操作来实现加速。

在这 10 年中，集成电路技术原动力的增长使得 OpenGL 体系结构(以及后来的 Direct3D)扩展到了 ISA 接口。这些扩展在图形流水线中作为可编程顶点和片段着色器出现，现在，随着 CUDA 的引入，这些扩展已经成为了与微处理器同等重要的数据并行 ISA 了。尽管和完整的类微处理器多功能性相比，它的周期还不是那么完整，但图形硬件加速的巨大威力对程序员来讲已经变得比以前更易用了。

它的计算能力非常强大！在编写本书时，NVIDIA GeForce 8800 Ultra 每秒可以执行超过 400 亿次的浮点运算——比 10 年前最强大的超级计算机还要厉害，并且是现今最强大的微处理器功能的 5 倍。Ultra 所支持的数据并行编程模型可以充分发挥其计算能力，而不必关心究竟使用了多少个处理器。这是非常重要的，因为今天的 Ultra 已经包含了 100 个处理器，而明天的 Ultra 可能包含更多，以后会更多。如摩尔定律所言，我们看不到集成电路密度究竟会发展到何种程度，但是大规模并行系统无疑是计算的未来，而图形硬件将会引导这种趋势。

本书收集了大量最前沿的 GPU 编程实例。现在，已是将数据并行处理纳入工作中的

时候了。本书前 4 部分介绍的是 GPU 在几何体、光照、渲染和图像处理领域的具体应用。第Ⅴ、Ⅵ部分扩大了 GPU 的应用范围，通过一些具体的可被数据并行 GPU 技术解决的非图形应用程序来说明这一点。这些应用多种多样，从刚体模拟到流体模拟，从病毒签名匹配到加密和解密，从随机数生成到 Gaussian 计算。

最先进的计算设备究竟在哪里呢？本书的封面提醒我们，人类的大脑仍然是最强大的并行计算系统。计算机科学一个长期的目标是达到并最终超越人类的大脑。对计算机图形社区的成员来讲，这将是非常振奋人心的，因为他们了解、解决并掌控着大规模并行计算的挑战，他们最有可能实现这个梦想。

Kurt Akeley
微软研究院

前 言

自第一本 GPU Gems 出版后只有短短的三年时间，实时图形学就已经真正超越了现实。本书第 14 章“用于真实感实时皮肤渲染的高级技术”，将这种发展趋势展示得淋漓尽致，它描述了一种非常高级的皮肤渲染技术，使得未来几年数据获取和动画将会成为人体动画领域最具挑战性的问题。

所有这些进步都是由 GPU 持续的创新带来的。GPU 处理单元变得更快、更易用。今天的 GPU 能够处理海量数据，而不是仅仅用于渲染 3D 场景，同时也能够处理图像和执行大规模的并行计算，例如，金融统计或者用于地形分析以寻找新的油田。

无论是用于计算还是图形处理，GPU 都需要使用一个软件接口来驱动它们，我们现在正好处于这样一个重要的转变期。新一代的 API 带来了额外的不相关性并展现出了新的诸如生成可编程几何体的能力。在计算领域，CUDA 体系结构使得开发人员可以使用类 C 风格的编程语言来执行计算任务，而不是强迫程序员必须使用图形流水线。这种体系结构使得开发人员不必掌握图形学的知识就可以充分利用 GPU 强大的性能。

在本书准备编写时，我们收到了来自 GPU 编程社区提交的 200 多个章节，覆盖了 GPU 使用领域的大部分内容，从 3D 渲染到非图形应用。每一章节的内容都经过了 NVIDIA 的工程师和外部评审家的严格审查。

我们最终在本书中收录了 41 章的内容，对每一章又经过了仔细评审，在这个过程中我们不断收到了编辑和同行们的反馈意见，并对相应内容进行了改进，大大提高了内容质量。遗憾的是，限于篇幅，我们无法将其他一些优秀的章节收录其中。对于我们而言，建立最终的内容目录是非常困难的，但是我们仍然要感谢所有提交作品的作者。

读者对象

对那些与图形学相关的章节，我们希望读者能够熟悉一些图形学的基础知识，包括图形 APIs(如 DirectX 和 OpenGL)，以及它们所关联的高级编程语言，如 HLSL、GLSL 或 Cg。任何交互式 3D 应用程序的开发者都会在这本书中找到适用于 GPU 的有价值的开发技术。

对计算和 CUDA 感兴趣的读者最好能了解一些并行计算方面的知识。C 语言编程的知识也是必备的。

使用示例代码

本书配有一张光盘，其中包含了部分示例代码、视频和其他解释本书提到技术的材料。您可以到本书的网站上查看最近的更新和补充材料，网址为 **developer.nvidia.com/gpugems**。

致谢

这本书凝聚了很多人的心血——尤其是大量将他们最近的工作成果提交给 GPU 社区的作者。毫无疑问，这些充满灵感和力量的章节将会帮助数以千计的开发者把这些技术应用到他们的程序中。

我们的责任编辑 Cyril Zeller、Evan Hart、Ignacio Castaño Aguado、Kevin Bjorke、Kevin Myers 和 Nolan Goodnight——在本书中起到了无价的作用，他们给作者提供反馈和指导，使得这些章节尽可能的精益求精。没有他们的专家经验和贡献，以及超负荷的工作，也就没有本书的出版。

为了使本书的讲解更加清楚，需要使用大量的图表、注释和屏幕截图。将大约 500 幅图的风格整理一致，需要大量辛勤的劳动，这里我们对 Michael Fornalski 和 Jim Reed 的出色表现表示感谢。我们对 Huey Nguyen 及其团队对我们项目的支持表示感激。我们同样对 Rory Loeb 令人赞叹的封面设计和本书许多其他的图形元素设计表示感谢。

我们也非常感谢 Catherine Kilkenny 和 Teresa Saffaie，他们在之前副本的编辑中给予了巨大帮助。

Randy Fernando，上一版 *GPU Gems* 的编辑，和我们分享了他编辑这类书籍的宝贵经验。

我们对 Kurt Akeley 表示感激，他对前沿技术富有洞察力和先见之明。

在 Addison-Wesley，Peter Gordon、John Fuller 和 Kim Boedigheimer 一直管理着这个项目，直到将其送给市场部的 Curt Johnson。Christopher Keane 在副本编辑和排版上出色的表现令人感激。

来自 NVIDIA 主管部门的支持对这本书的出版也至关重要：Tony Tamasi 和 Dan Vivoli 一直都高度重视培训教材的编写，同时提供了完成这个项目所需的必要资源。

我们对 Jen-Hsun Huang 在本书中的不断支持表示感谢，同时他也创建了一个鼓励创新和团队合作的环境。

我们也要感谢 NVIDIA 公司每个人对本书的支持，是他们持续不断的创新改变了人们对于计算的认识。

Hubert Nguyen
NVIDIA 公司

目　录

第 I 部分　几何体……2

第 1 章　使用 GPU 生成复杂的程序化地形……7

1.1　介绍……7

1.2　Marching Cubes 算法和密度函数……7

1.2.1　在单元内生成多边形……8

1.2.2　查找表……9

1.3　地形生成系统概述……10

1.3.1　在地形块内部生成多边形……10

1.3.2　生成密度值……11

1.3.3　写一个有趣的密度函数……11

1.3.4　定制地形……14

1.4　在地形块中生成多边形……16

1.4.1　边缘数据……17

1.4.2　生成地形块：方法 1……19

1.4.3　生成地形块：方法 2……19

1.4.4　生成地形块：方法 3……21

1.5　纹理和光影……23

1.6　对实际应用的考虑……27

1.6.1　细节层次……27

1.6.2　外部对象的碰撞和光照问题……28

1.7　结论……29

1.8　参考资料……29

第 2 章　群体动画渲染……31

2.1　目的……32

2.2　实例化的简单回顾……32

2.3　技术细节……33

2.3.1　基于常量的实例化……34

2.3.2　使用动画纹理的调色板蒙皮……35

2.3.3　几何变化……38

2.3.4　LOD 系统……39

2.4　其他考虑因素……39

2.4.1　颜色变化……39

2.4.2　性能……39

2.4.3　整合……40

2.5　结论……40

2.6　参考资料……40

第 3 章　DirectX 10 混合形状：打破限制……41

3.1　介绍……41

3.2　Dawn 例子的实现……43

3.2.1　DirectX 10 的特性……43

3.2.2　定义网格……43

3.2.3　流输出的方法……44

3.2.4　缓冲区-模板方法……47

3.3　运行例子……51

3.4　性能……51

3.5　参考资料……52

第 4 章　下一代 SpeedTree 渲染……53

4.1　介绍……53

4.2　轮廓裁减……53

4.2.1　轮廓鳍挤压……54

4.2.2　高度追踪……56

4.2.3 轮廓细节层次 …… 58
4.3 阴影 …… 59
4.3.1 树叶自遮挡 …… 59
4.3.2 级联阴影贴图 …… 61
4.4 树叶光照 …… 62
4.4.1 两边光照 …… 62
4.4.2 镜面光照 …… 64
4.5 高动态范围和反锯齿 …… 64
4.6 半透明覆盖 …… 65
4.6.1 将半透明覆盖应用于 SpeedTree …… 65
4.6.2 细节层次的交叉衰减 …… 65
4.6.3 轮廓边反锯齿 …… 66
4.7 结论 …… 68
4.8 参考资料 …… 69

第 5 章 普遍自适应的网格优化 …… 71
5.1 介绍 …… 71
5.2 总览 …… 72
5.3 自适应优化模式 …… 73
5.4 渲染工作流 …… 74
5.4.1 深度标签计算 …… 74
5.4.2 CPU 阶段的渲染循环 …… 75
5.4.3 GPU 阶段的优化处理 …… 76
5.5 结果 …… 76
5.6 结论和改进 …… 79
5.7 参考资料 …… 79

第 6 章 GPU 生成的树的过程式风动画 …… 81
6.1 介绍 …… 81
6.2 GPU 上的过程式动画 …… 81
6.3 现象学方法 …… 82
6.3.1 风场 …… 82
6.3.2 树的概念结构 …… 82
6.3.3 模拟的两种分类 …… 83
6.4 模拟步骤 …… 86
6.5 渲染树 …… 89
6.6 分析和比较 …… 90
6.6.1 优势 …… 91
6.6.2 劣势 …… 91
6.6.3 性能评估 …… 91
6.7 结论 …… 92
6.8 参考资料 …… 92

第 7 章 GPU 上基于点的变形球可视化 …… 93
7.1 变形球、光滑粒子流体力学和表面粒子 …… 93
7.1.1 各种方法的对比 …… 94
7.1.2 在 GPU 上基于点的表面可视化 …… 95
7.2 限制粒子 …… 96
7.2.1 定义显式表面 …… 96
7.2.2 速度限制等式 …… 97
7.2.3 在 GPU 中计算密度场 …… 99
7.2.4 选择散列函数 …… 99
7.2.5 创建并查询散列表 …… 100
7.3 局部粒子斥力 …… 102
7.3.1 斥力等式 …… 102
7.3.2 GPU 端最近的邻居 …… 104
7.4 全局粒子传播 …… 106
7.5 性能 …… 109
7.6 渲染 …… 110
7.7 结论 …… 111
7.8 参考资料 …… 111

第 II 部分 光照和阴影 …… 114

第 8 章 区域求和的差值阴影贴图 …… 119
8.1 介绍 …… 119
8.2 相关工作 …… 120
8.3 percentage-closer 过滤 …… 120
8.4 差值阴影贴图 …… 122
8.4.1 过滤差值阴影贴图 …… 123
8.4.2 偏离 …… 124
8.4.3 光渗色 …… 126
8.4.4 数值稳定性 …… 128
8.4.5 实现注释 …… 129
8.4.6 差值阴影贴图和软阴影 …… 130

8.5 区域求和差值阴影贴图……131
8.5.1 生成区域求和表……132
8.5.2 数字固定性重访问……132
8.5.3 结果……133
8.6 percentage-closer 软阴影……135
8.6.1 遮挡体查找……135
8.6.2 渐变区域尺寸估计……135
8.6.3 阴影过滤……136
8.6.4 结果……136
8.7 结论……136
8.8 参考资料……137

第 9 章 使用全局照明实现互动的电影级重光照……139
9.1 介绍……139
9.2 算法总览……140
9.3 聚集样本……141
9.4 一次反射的间接照明……143
9.5 用于压缩的小波……144
9.6 增加多次反射……146
9.7 对稀疏矩阵数据进行压缩……147
9.8 基于 GPU 的重光照引擎……148
9.8.1 直接照明……149
9.8.2 小波变换……149
9.8.3 稀疏矩阵乘法……151
9.9 结果……152
9.10 结论……153
9.11 参考资料……153

第 10 章 在可编程 GPU 中实现并行分割的阴影贴图……155
10.1 介绍……155
10.2 算法……156
10.2.1 步骤 1：分割视锥体……157
10.2.2 步骤 2：计算光的变换矩阵……160
10.2.3 步骤 3 和 4：产生 PSSM 和综合阴影……163
10.3 基于硬件的实现方法……164
10.3.1 多步方法……164
10.3.2 DirectX 9 级别的加速……166
10.3.3 DirectX 10 级别的加速……168
10.4 进一步的优化……177
10.5 结果……177
10.6 结论……180
10.7 参考资料……180

第 11 章 使用层次化的遮挡剔除和几何体着色器得到高效鲁棒的阴影体……183
11.1 介绍……183
11.2 阴影体综述……183
11.2.1 Z-Pass 和 Z-Fail……184
11.2.2 阴影体生成……185
11.2.3 性能和优化方法……186
11.3 实现方法……186
11.3.1 针对低质量网络的鲁棒阴影……186
11.3.2 使用几何体着色器动态生成阴影体……188
11.3.3 使用层次化遮挡裁剪提高性能……192
11.4 结论……194
11.5 参考资料……195

第 12 章 高质量的环境遮挡……197
12.1 回顾……197
12.2 问题……198
12.2.1 圆盘形状的失真……199
12.2.2 高频的尖点失真……199
12.3 一个鲁棒的解决方法……200
12.3.1 对不连续性进行光滑处理……200
12.3.2 移除尖点并加入细节……201
12.4 结果……204
12.5 性能……205
12.6 一些注意事项……206
12.6.1 强制收敛……206
12.6.2 可调整的参数……207
12.7 后续工作……209

12.8 参考资料……210

第 13 章 作为后置处理的体积光照散射……211
13.1 介绍……211
13.2 云隙光……212
13.3 体积光照散射……212
13.4 后置处理像素着色器……214
13.5 屏幕空间遮挡方法……215
13.5.1 遮挡 pre-pass 方法……215
13.5.2 遮挡模板方法……216
13.5.3 遮挡对比方法……216
13.6 一些注意事项……216
13.7 演示……216
13.8 扩展……217
13.9 结论……217
13.10 参考资料……218

第Ⅲ部分 渲染……220

第 14 章 用于真实感实时皮肤渲染的高级技术……225
14.1 皮肤外观……225
14.1.1 皮肤表面反射……226
14.1.2 皮肤子表面反射……227
14.2 皮肤渲染系统总述……228
14.3 镜面表面反射……229
14.4 散射理论……233
14.4.1 漫反射剖面……233
14.4.2 使用漫反射剖面渲染……234
14.4.3 漫反射剖面的形状……235
14.4.4 一个高斯和漫反射剖面……236
14.4.5 拟合预测的或测量的剖面……237
14.4.6 配置漫反射剖面……238
14.4.7 对皮肤的高斯和拟合……238
14.5 高级子表面散射……239
14.5.1 纹理空间漫反射……240
14.5.2 改进的纹理空间漫反射……241
14.5.3 经过修改的半透明阴影贴图……256
14.6 一个快速的布隆过滤器……260
14.7 结论……261
14.8 后续工作……261
14.9 参考资料……262

第 15 章 可播放的全方位捕捉……265
15.1 介绍……265
15.2 数据捕捉流水线……266
15.3 动态纹理的压缩和解压……268
15.3.1 主要分量分析……268
15.3.2 压缩……270
15.3.3 解压缩……271
15.3.4 可变 PCA……272
15.3.5 实际使用中的考虑……273
15.4 串连特性……274
15.5 结论……275
15.6 参考资料……280

第 16 章 Crysis 中植被的过程化动画和着色……283
16.1 过程化动画……283
16.2 植被着色……287
16.2.1 环境光照……288
16.2.2 边的光滑化……288
16.2.3 整合……289
16.2.4 实现细节……289
16.3 结论……291
16.4 参考资料……292

第 17 章 鲁棒的多镜面反射和折射……293
17.1 介绍……293
17.2 跟踪辅助光线……294
17.2.1 层次化距离图的生成……295
17.2.2 层次化距离图的光线跟踪……295
17.3 反射和折射……300
17.4 结果……302

17.5 结论……306
17.6 参考资料……306

第 18 章 用于浮雕映射的松散式锥形步进……309
18.1 介绍……309
18.2 浮雕映射总览……310
18.3 锥形步进映射……313
18.4 松散式锥形步进……313
18.4.1 计算松散式锥形映射图……314
18.4.2 使用松散式锥形映射图进行渲染……317
18.5 结论……320
18.6 补充读物……321
18.7 参考资料……321

第 19 章 Tabula Rasa 中的延迟着色……323
19.1 介绍……323
19.2 一些背景知识……324
19.3 所支持的向前着色……324
19.3.1 一个受限的特效集……325
19.3.2 单效果，多技术……325
19.3.3 光照优先度……326
19.4 先进的光照特性……326
19.4.1 双向光照……327
19.4.2 球体映射……327
19.4.3 盒式光照……328
19.4.4 阴影贴图……328
19.4.5 进一步的扩展……330
19.5 可读取深度和法线缓存的优点……331
19.5.1 高级水体和折射……331
19.5.2 与分辨率无关的边检测……332
19.6 一些注意事项……334
19.6.1 材质属性……334
19.6.2 精确度……336
19.7 优化……336
19.7.1 高效的光照体积……337
19.7.2 模板掩码……337
19.7.3 动态分支……338
19.8 问题……338
19.8.1 混合透明度几何体……338
19.8.2 内存带宽……339
19.8.3 内存管理……340
19.9 结果……340
19.10 结论……342
19.11 参考资料……343

第 20 章 基于 GPU 的重要性采样……345
20.1 介绍……345
20.2 渲染公式……346
20.2.1 蒙特卡洛积分……346
20.2.2 重要性采样……346
20.2.3 对材质函数进行采样……347
20.3 拟随机低差异序列……349
20.4 使用 mipmap 过滤的样本……350
20.5 性能……353
20.6 结论……355
20.7 补充读物及参考资料……355

第Ⅳ部分 图像效果……358

第 21 章 真正的 Impostor……363
21.1 介绍……363
21.2 算法和实现细节……364
21.3 结果……367
21.4 结论……368
21.5 参考资料……369

第 22 章 在 GPU 上处理法线贴图……371
22.1 传统实现……371
22.1.1 投影……371
22.1.2 包围盒……372
22.2 加速结构……373
22.2.1 一致网格……373
22.2.2 3D 数值微分分析……373
22.3 GPU 的处理……374
22.3.1 序号限制……375
22.3.2 内存和结构限制……375
22.4 实现……376

22.4.1 设置以及预处理……376
22.4.2 单路径实现……378
22.4.3 多路径实现……383
22.4.4 反走样……383
22.5 结果……383
22.6 结论……385
22.7 参考资料……386

第 23 章 高速的离屏粒子……387
23.1 动机……387
23.2 离屏渲染……388
23.2.1 离屏深度测试……388
23.2.2 获取深度……388
23.3 向下采样深度……389
23.3.1 点采样深度……390
23.3.2 最大深度采样……391
23.4 深度测试与柔和粒子……391
23.5 alpha 混合……392
23.6 混合分辨率渲染……393
23.6.1 边检测……394
23.6.2 与模板进行合成……394
23.7 结果……396
23.7.1 图像质量……396
23.7.2 性能……396
23.8 结论……398
23.9 参考资料……398

第 24 章 保持线性的重要性……399
24.1 介绍……399
24.2 光照、显示以及颜色空间……399
24.2.1 数字图像捕捉、生成及显示中的问题……399
24.2.2 题外话：何为线性……400
24.2.3 显示器并非线性，渲染器为线性……400
24.3 症状……402
24.3.1 非线性输入纹理……402
24.3.2 mipmap……402
24.3.3 光照……403
24.3.4 负负不得正……403
24.4 解决方法……405
24.4.1 输入图像(扫描、绘画以及数字图片)……406
24.4.2 输出图像(最终渲染)……407
24.4.3 中间颜色缓存……408
24.5 结论……408
24.6 补充读物……408

第 25 章 在 GPU 上渲染向量图……411
25.1 介绍……411
25.2 二次曲线……411
25.3 三次样条……413
25.3.1 弯形曲线……417
25.3.2 环形曲线……418
25.3.3 尖形曲线……419
25.3.4 二次曲线……419
25.4 三角化……420
25.5 抗锯齿……421
25.6 代码……422
25.7 结论……423
25.8 参考资料……424

第 26 章 通过颜色进行对象探测：使用 GPU 进行实时视频图像处理……425
26.1 图像处理概况……425
26.2 根据颜色进行对象检测……427
26.2.1 创建标记……428
26.2.2 查找质心……430
26.2.3 将一幅图像整合到输入信号上……432
26.3 结论……433
26.4 补充读物……433

第 27 章 作为后置处理效果的运动模糊……435
27.1 介绍……435
27.2 从深度缓存提取对象位置……436
27.3 执行运动模糊……438
27.4 处理动态对象……439

27.5 屏蔽对象……439
27.6 额外的工作……439
27.7 结论……439
27.8 参考资料……439

第 28 章 实用景深后期处理……441
28.1 介绍……441
28.2 相关工作……441
28.2.1 总览……441
28.2.2 特定技术……442
28.3 景深……442
28.4 算法的改进……443
28.4.1 最初的随机方法 (Stochastic Approach)……444
28.4.2 作为聚集的散射方法……444
28.4.3 模糊方法……445
28.5 完整的算法……448
28.5.1 深度信息……448
28.5.2 可变宽度的模糊……448
28.5.3 散光圈半径……449
28.5.4 关于第一人称视角武器的考虑……450
28.5.5 完整的着色器代码……450
28.6 结论……455
28.7 局限和后续工作……456
28.8 参考资料……458

第Ⅴ部分 物理仿真……460

第 29 章 GPU 上实时刚体仿真……465
29.1 介绍……466
29.1.1 平移……466
29.1.2 旋转……467
29.1.3 形状表示……468
29.1.4 碰撞检测……469
29.1.5 碰撞交互……470
29.2 GPU 上刚体仿真……470
29.2.1 概述……470
29.2.2 数据结构……472
29.2.3 步骤 1：计算粒子值……473
29.2.4 步骤 2：网格生成……474
29.2.5 步骤 3：碰撞检测与交互……475
29.2.6 步骤 4：动量计算……476
29.2.7 步骤 5：计算位置和四元组……476
29.2.8 渲染……476
29.2.9 性能……477
29.3 应用……477
29.3.1 颗粒状材料……478
29.3.2 液体……478
29.3.3 耦合……479
29.4 结论……480
29.5 附录……480
29.6 参考资料……480

第 30 章 实时仿真与 3D 流体渲染……483
30.1 介绍……483
30.2 仿真……484
30.2.1 背景……484
30.2.2 流体运动方程……485
30.2.3 速度求解……485
30.2.4 固-流体交互……490
30.2.5 烟雾……501
30.2.6 火……502
30.2.7 水……502
30.2.8 性能考虑……503
30.2.9 存储……504
30.2.10 数值问题……504
30.3 渲染……506
30.3.1 体积渲染……506
30.3.2 液体渲染……511
30.4 结论……512
30.5 参考资料……512

第 31 章 使用 CUDA 进行快速 N-body 仿真……515
31.1 介绍……515
31.2 全对 N-body 仿真……516

31.3 全对 N-body 算法的 CUDA 实现……518
31.3.1 体 - 体之间力的计算……518
31.3.2 块计算……519
31.3.3 把块聚集到线程块中……520
31.3.4 定义线程块的网格……521
31.4 性能评测……522
31.4.1 优化……523
31.4.2 性能结果分析……525
31.5 以前使用 GPU 进行 N-body 仿真的方法……526
31.6 分层的 N-body 方法……526
31.7 结论……527
31.8 参考资料……527

第 32 章 使用 CUDA 进行宽阶段碰撞检测……531
32.1 宽阶段算法……531
32.1.1 排序和搜索……531
32.1.2 空间划分……532
32.1.3 并行空间划分……534
32.2 空间划分的一种 CUDA 实现……535
32.2.1 初始化……536
32.2.2 构建单元 ID 数组……537
32.2.3 单元 ID 数组的排序……538
32.2.4 创建碰撞单元列表……547
32.2.5 遍历碰撞单元列表……548
32.3 性能结果……548
32.4 结论……549
32.5 参考资料……549

第 33 章 用于碰撞检测的 LCP 算法的 CUDA 实现……551
33.1 并行处理……551
33.2 物理流程……552
33.3 确定接触点……553
33.3.1 连续方法……553
33.3.2 离散方法(基于一致性)…554
33.3.3 解析接触点……554
33.4 数学优化……554
33.4.1 线性规划……554
33.4.2 线性互补问题……555
33.4.3 二次规划……556
33.5 凸面距离计算……556
33.6 使用 CUDA 的 LCP 并行解决方案……558
33.7 结果……561
33.8 参考资料……563

第 34 章 使用单过程 GPU 扫描和四面体转换的有向距离场…565
34.1 介绍……565
34.1.1 有向距离场总览……565
34.1.2 方法的概览……566
34.2 扫描方法中的泄露瑕疵……566
34.2.1 平面测试……566
34.2.2 如何构建包围体……567
34.2.3 多边形模型的折叠……569
34.3 四边形 GPU 扫描方法……570
34.3.1 计算壳……571
34.3.2 计算四边形的交叉区域…571
34.3.3 使用角度权重的伪法线计算有向距离……572
34.4 结果……575
34.5 结论……578
34.6 后续工作……578
34.6.1 算法的改进……578
34.6.2 对实现方法进行改进……578
34.7 补充读物……578
34.8 参考资料……580

第VI部分 GPU 计算……582

第 35 章 使用 GPU 进行病毒特征的快速匹配……587
35.1 介绍……587
35.2 模式匹配……589
35.3 GPU 实现……590
35.4 结果……593

35.5 结论和后续工作……595
35.6 参考资料……596

第 36 章 用 GPU 进行 AES 加密和解密……597
36.1 用于整数流处理的新功能……597
36.1.1 转换反馈模式……597
36.1.2 GPU 编程扩展……598
36.2 AES 算法概述……599
36.3 AES 在 GPU 上的实现……601
36.3.1 输入/输出和状态……602
36.3.2 初始化……603
36.3.3 轮操作……603
36.3.4 最后一轮……607
36.4 性能……607
36.4.1 顶点程序与片段程序……607
36.4.2 与基于 CPU 的加密的比较……608
36.5 对并行的考虑……608
36.5.1 操作的块密码模式……608
36.5.2 用于并行处理的模式……609
36.6 结论和后续工作……610
36.7 参考资料……611

第 37 章 使用 CUDA 进行高效的随机数生成及应用……613
37.1 Monte Carlo 仿真……614
37.2 随机数生成器……616
37.2.1 简介……616
37.2.2 均匀到高斯的转换生成器……617
37.2.3 高斯转换的类型……620
37.2.4 Wallace 高斯生成器……621
37.2.5 把 Wallace 高斯生成器集成到仿真中……624
37.3 示例程序……625
37.3.1 亚式期权……626
37.3.2 回顾期权的变体……627
37.3.3 结果……629
37.4 结论……630
37.5 参考资料……630

第 38 章 使用 CUDA 进行地球内部成像……633
38.1 介绍……633
38.2 地震数据……634
38.3 地震数据处理……635
38.3.1 震波传播……637
38.3.2 使用 SRMIP 算法进行震波偏移……638
38.4 GPU 实现……640
38.4.1 GPU/CPU 通信……641
38.4.2 CUDA 的实现……642
38.4.3 震波传播内核……643
38.5 性能……646
38.6 结论……646
38.7 参考资料……647

第 39 章 使用 CUDA 的并行前缀和(扫描方法)……649
39.1 介绍……649
39.2 实现……650
39.2.1 并行扫描的雏形……650
39.2.2 一种高效的并行扫描算法……652
39.2.3 避免 bank 冲突……654
39.2.4 随机长度的数组……656
39.2.5 进一步优化及性能结果……657
39.2.6 CUDA 对 OpenGL 实现的优势……659
39.3 扫描的应用……660
39.3.1 流压缩……660
39.3.2 区域求和表……661
39.3.3 基数排序……663
39.3.4 前人的贡献……665
39.4 结论……666
39.5 参考资料……666

第 40 章 高斯函数的增量计算……669
40.1 介绍和相关工作……669