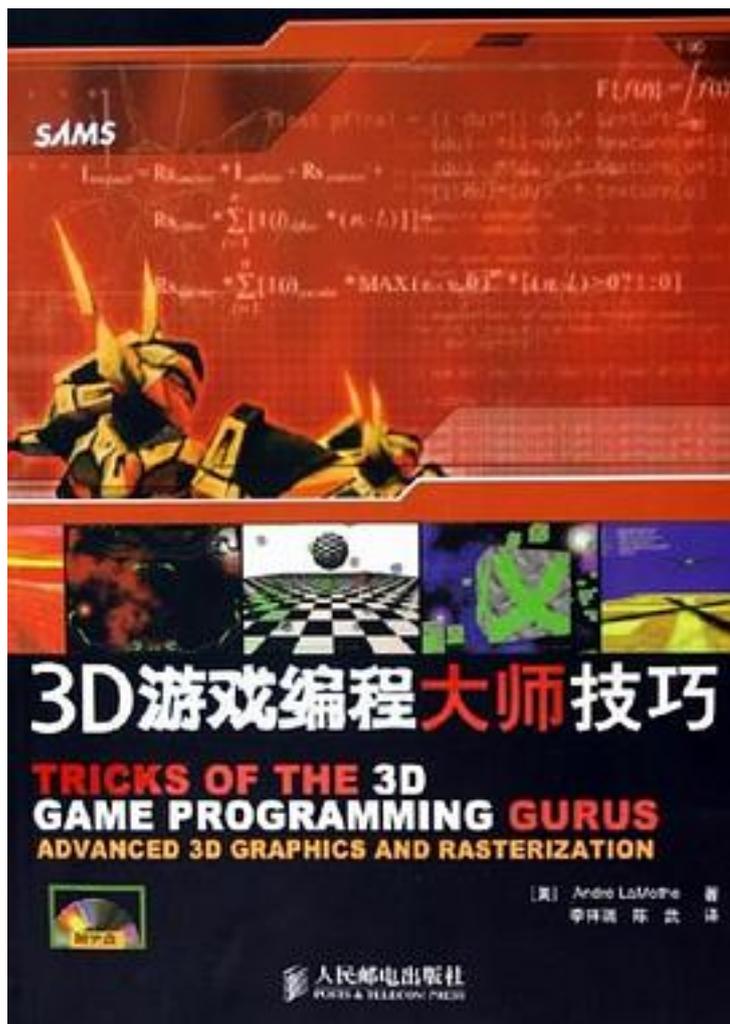


3D游戏编程大师技巧



[3D游戏编程大师技巧_下载链接1](#)

著者:[美] 拉莫泽

出版者:人民邮电出版社

出版时间:2005-6

装帧:平装

isbn:9787115133717

《3D 游戏编程大师技巧》是游戏编程畅销书作者André

LaMothe的扛鼎之作，从游戏编程和软件引擎的角度深入探讨了3D图形学的各个重要主题。全书共分5部分，包括16章的内容。第1~3章简要地介绍了Windows和DirectX编程，创建了一个Windows应用程序模板，让读者能够将精力放在游戏逻辑和图形实现中，而不用考虑Windows和DirectX方面的琐事；第4~5章简要地介绍了一些数学知识并实现了一个数学库，供以后编写演示程序时使用；第6章概述了3D图形学，让读者对《3D游戏编程大师技巧(附光盘)》将介绍的内容有大致地了解；第7~11章分别介绍了光照、明暗处理、仿射纹理映射、3D裁剪和深度缓存等内容；第12~14章讨论了高级3D渲染技术，包括透视修正纹理映射、Alpha混合、1/z缓存、纹理滤波、空间划分和可见性算法、阴影、光照映射等；第15~16章讨论了动画、运动碰撞检测和优化技术。

《3D游戏编程大师技巧》适合于有一定编程经验并想从事游戏编程工作或对3D图形学感兴趣的人员阅读。

作者简介:

目录: 第一部分 3D游戏编程简介

第1章 3D游戏编程入门 2

1.1 简介 2

1.2 2D/3D游戏的元素 3

1.2.1 初始化 3

1.2.2 进入游戏循环 3

1.2.3 读取玩家输入 4

1.2.4 执行AI和游戏逻辑 4

1.2.5 渲染下一帧 4

1.2.6 同步显示 4

1.2.7 循环 4

1.2.8 关闭 5

1.3 通用游戏编程指南 7

1.4 使用工具 9

1.4.1 3D关卡编辑器 12

1.4.2 使用编译器 13

1.5 一个3D游戏范例: Raiders 3D 15

1.5.1 事件循环 33

1.5.2 核心3D游戏逻辑 34

1.5.3 3D投影 35

1.5.4 星空 36

1.5.5 激光炮和碰撞检测 37

1.5.6 爆炸 37

1.5.7 玩Raiders3D 37

1.6 总结 37

第2章 Windows和DirectX简明教程 38

2.1 Win32编程模型 38

2.2 Windows程序的最小需求 39

2.3 一个基本的Windows应用程序 43

2.3.1 Windows类 43

2.3.2 注册Windows类 47

2.3.3 创建窗口 47

2.3.4 事件处理程序 48

2.3.5 主事件循环 52

2.3.6 构建实时事件循环 55

2.4 DirectX和COM简明教程 56

- 2.4.1 HEL和HAL 57
- 2.4.2 DirectX基本类 58
- 2.5 COM简介 59
 - 2.5.1 什么是COM对象 60
 - 2.5.2 创建和使用DirectX COM接口 61
 - 2.5.3 查询接口 62
- 2.6 总结 64
- 第3章 使用虚拟计算机进行3D游戏编程 65
 - 3.1 虚拟计算机接口简介 65
 - 3.2 建立虚拟计算机接口 66
 - 3.2.1 帧缓存和视频系统 66
 - 3.2.2 使用颜色 70
 - 3.2.3 缓存交换 71
 - 3.2.4 完整的虚拟图形系统 73
 - 3.2.5 I/O、声音和音乐 73
 - 3.3 T3DLIB游戏控制台 74
 - 3.3.1 T3DLIB系统概述 74
 - 3.3.2 基本游戏控制台 74
 - 3.4 T3DLIB1库 79
 - 3.4.1 DirectX图形引擎体系结构 79
 - 3.4.2 基本常量 79
 - 3.4.3 工作宏 81
 - 3.4.4 数据类型和结构 81
 - 3.4.5 函数原型 84
 - 3.4.6 全局变量 88
 - 3.4.7 DirectDraw接口 89
 - 3.4.8 2D多边形函数 92
 - 3.4.9 数学函数和错误函数 97
 - 3.4.10 位图函数 99
 - 3.4.11 8位调色板函数 102
 - 3.4.12 实用函数 104
 - 3.4.13 BOB(Blitter对象)引擎 106
 - 3.5 T3DLIB2 DirectX输入系统 112
 - 3.6 T3DLIB3声音和音乐库 116
 - 3.6.1 头文件 117
 - 3.6.2 类型 117
 - 3.6.3 全局变量 117
 - 3.6.4 DirectSound API封装函数 118
 - 3.6.5 DirectMusic API封装函数 121
 - 3.7 建立最终的T3D游戏控制台 124
 - 3.7.1 映射真实图形到虚拟接口的非真实图形 124
 - 3.7.2 最终的T3DLIB游戏控制台 126
 - 3.8 范例T3LIB应用程序 134
 - 3.8.1 窗口应用程序 134
 - 3.8.2 全屏应用程序 135
 - 3.8.3 声音和音乐 136
 - 3.8.4 处理输入 136
 - 3.9 总结 139
- 第二部分 3D数学和变换
- 第4章 三角学、向量、矩阵和四元数 142
 - 4.1 数学表示法 142
 - 4.2 2D坐标系 143
 - 4.2.1 2D笛卡尔坐标 143
 - 4.2.2 2D极坐标 144

| | |
|-------------------|-----|
| 4.3 3D坐标系 | 147 |
| 4.3.1 3D笛卡尔坐标 | 147 |
| 4.3.2 3D柱面坐标 | 149 |
| 4.3.3 3D球面坐标 | 150 |
| 4.4 三角学 | 151 |
| 4.4.1 直角三角形 | 151 |
| 4.4.2 反三角函数 | 153 |
| 4.4.3 三角恒等式 | 153 |
| 4.5 向量 | 154 |
| 4.5.1 向量长度 | 155 |
| 4.5.2 归一化 | 155 |
| 4.5.3 向量和标量的乘法 | 155 |
| 4.5.4 向量加法 | 156 |
| 4.5.5 向量减法 | 157 |
| 4.5.6 点积 | 157 |
| 4.5.7 叉积 | 159 |
| 4.5.8 零向量 | 160 |
| 4.5.9 位置和位移向量 | 160 |
| 4.5.10 用线性组合表示的向量 | 161 |
| 4.6 矩阵和线性代数 | 161 |
| 4.6.1 单位矩阵 | 162 |
| 4.6.2 矩阵加法 | 163 |
| 4.6.3 矩阵的转置 | 163 |
| 4.6.4 矩阵乘法 | 164 |
| 4.6.5 矩阵运算满足的定律 | 165 |
| 4.7 逆矩阵和方程组求解 | 165 |
| 4.7.1 克来姆法则 | 167 |
| 4.7.2 使用矩阵进行变换 | 168 |
| 4.7.3 齐次坐标 | 169 |
| 4.7.4 应用矩阵变换 | 170 |
| 4.8 基本几何实体 | 176 |
| 4.8.1 点 | 176 |
| 4.8.2 直线 | 176 |
| 4.8.3 平面 | 179 |
| 4.9 使用参数化方程 | 182 |
| 4.9.1 2D参数化直线 | 182 |
| 4.9.2 3D参数化直线 | 184 |
| 4.10 四元数简介 | 189 |
| 4.10.1 复数理论 | 189 |
| 4.10.2 超复数 | 193 |
| 4.10.3 四元数的应用 | 197 |
| 4.11 总结 | 200 |
| 第5章 建立数学引擎 | 201 |
| 5.1 数学引擎概述 | 201 |
| 5.1.1 数学引擎的文件结构 | 201 |
| 5.1.2 命名规则 | 202 |
| 5.1.3 错误处理 | 203 |
| 5.1.4 关于C++的最后说明 | 203 |
| 5.2 数据结构和类型 | 203 |
| 5.2.1 向量和点 | 203 |
| 5.2.2 参数化直线 | 204 |
| 5.2.3 3D平面 | 206 |
| 5.2.4 矩阵 | 206 |
| 5.2.5 四元数 | 209 |

- 5.2.6 角坐标系支持 210
- 5.2.7 2D极坐标 210
- 5.2.8 3D柱面坐标 211
- 5.2.9 3D球面坐标 211
- 5.2.10 定点数 212
- 5.3 数学常量 213
- 5.4 宏和内联函数 214
 - 5.4.1 通用宏 218
 - 5.4.2 点和向量宏 218
 - 5.4.3 矩阵宏 219
 - 5.4.4 四元数 220
 - 5.4.5 定点数宏 221
- 5.5 函数原型 221
- 5.6 全局变量 224
- 5.7 数学引擎API清单 225
 - 5.7.1 三角函数 225
 - 5.7.2 坐标系支持函数 226
 - 5.7.3 向量支持函数 228
 - 5.7.4 矩阵支持函数 235
 - 5.7.5 2D和3D参数化直线支持函数 245
 - 5.7.6 3D平面支持函数 248
 - 5.7.7 四元数支持函数 252
 - 5.7.8 定点数支持函数 259
 - 5.7.9 方程求解支持函数 263
- 5.8 浮点单元运算初步 265
 - 5.8.1 FPU体系结构 266
 - 5.8.2 FPU堆栈 266
 - 5.8.3 FPU指令集 268
 - 5.8.4 经典指令格式 270
 - 5.8.5 内存指令格式 271
 - 5.8.6 寄存器指令格式 271
 - 5.8.7 寄存器弹出指令格式 271
 - 5.8.8 FPU范例 271
 - 5.8.9 FLD范例 272
 - 5.8.10 FST范例 272
 - 5.8.11 FADD范例 273
 - 5.8.12 FSUB范例 275
 - 5.8.13 FMUL范例 276
 - 5.8.14 FDIV范例 278
- 5.9 数学引擎使用说明 279
- 5.10 关于数学优化的说明 280
- 5.11 总结 280
- 第6章 3D图形学简介 282
 - 6.1 3D引擎原理 282
 - 6.2 3D游戏引擎的结构 282
 - 6.2.1 3D引擎 283
 - 6.2.2 游戏引擎 283
 - 6.2.3 输入系统和网络 284
 - 6.2.4 动画系统 284
 - 6.2.5 碰撞检测和导航系统 287
 - 6.2.6 物理引擎 288
 - 6.2.7 人工智能系统 289
 - 6.2.8 3D模型和图像数据库 289
 - 6.3 3D坐标系 291

- 6.3.1 模型(局部)坐标 291
- 6.3.2 世界坐标 293
- 6.3.3 相机坐标 296
- 6.3.4 有关相机坐标的说明 302
- 6.3.5 隐藏物体(面)消除和裁剪 303
- 6.3.6 透视坐标 308
- 6.3.7 流水线终点: 屏幕坐标 315
- 6.4 基本的3D数据结构 321
 - 6.4.1 表示3D多边形数据时需要考虑的问题 322
 - 6.4.2 定义多边形 323
 - 6.4.3 定义物体 327
 - 6.4.4 表示世界 330
- 6.5 3D工具 331
- 6.6 从外部加载数据 332
 - 6.6.1 PLG文件 333
 - 6.6.2 NFF文件 335
 - 6.6.3 3D Studio文件 338
 - 6.6.4 Caligari COB文件 343
 - 6.6.5 Microsoft DirectX .X文件 345
 - 6.6.6 3D文件格式小结 345
- 6.7 基本刚性变换和动画 345
 - 6.7.1 3D平移 345
 - 6.7.2 3D旋转 346
 - 6.7.3 3D变形 347
- 6.8 再看观察流水线 348
- 6.9 3D引擎类型 349
 - 6.9.1 太空引擎 349
 - 6.9.2 地形引擎 350
 - 6.9.3 FPS室内引擎 351
 - 6.9.4 光线投射和体素引擎 352
 - 6.9.5 混合引擎 353
- 6.10 将各种功能集成到引擎中 353
- 6.11 总结 353
- 第7章 渲染3D线框世界 354
 - 7.1 线框引擎的总体体系结构 354
 - 7.1.1 数据结构和3D流水线 355
 - 7.1.2 主多边形列表 357
 - 7.1.3 新的软件模块 359
 - 7.2 编写3D文件加载器 359
 - 7.3 构建3D流水线 367
 - 7.3.1 通用变换函数 367
 - 7.3.2 局部坐标到世界坐标变换 372
 - 7.3.3 欧拉相机模型 375
 - 7.3.4 UVN相机模型 377
 - 7.3.5 世界坐标到相机坐标变换 387
 - 7.3.6 物体剔除 390
 - 7.3.7 背面消除 393
 - 7.3.8 相机坐标到透视坐标变换 395
 - 7.3.9 透视坐标到屏幕(视口)坐标变换 399
 - 7.3.10 合并透视变换和屏幕变换 403
 - 7.4 渲染3D世界 405
 - 7.5 3D演示程序 408
 - 7.5.1 单个3D三角形 408
 - 7.5.2 3D线框立方体 411

- 7.5.3 消除了背面的3D线框立方体 413
- 7.5.4 3D坦克演示程序 414
- 7.5.5 相机移动的3D坦克演示程序 416
- 7.5.6 战区漫步演示程序 418
- 7.6 总结 421
- 第三部分 基本3D渲染
- 第8章 基本光照和实体造型 424
- 8.1 计算机图形学的基本光照模型 424
- 8.1.1 颜色模型和材质 426
- 8.1.2 光源类型 432
- 8.2 三角形的光照计算和光栅化 437
- 8.2.1 为光照做准备 441
- 8.2.2 定义材质 442
- 8.2.3 定义光源 445
- 8.3 真实世界中的着色 449
- 8.3.1 16位着色 449
- 8.3.2 8位着色 450
- 8.3.3 一个健壮的用于8位模式的RGB模型 450
- 8.3.4 一个简化的用于8位模式的强度模型 453
- 8.3.5 固定着色 457
- 8.3.6 恒定着色 459
- 8.3.7 Gouraud着色概述 472
- 8.3.8 Phong着色概述 474
- 8.4 深度排序和画家算法 475
- 8.5 使用新的模型格式 479
- 8.5.1 分析器类 479
- 8.5.2 辅助函数 482
- 8.5.3 3D Studio MAX ASCII格式.ASC 484
- 8.5.4 TrueSpace ASCII.COB格式 486
- 8.5.5 Quake II二进制.MD2格式概述 494
- 8.6 3D建模工具简介 495
- 8.7 总结 497
- 第9章 插值着色技术和仿射纹理映射 498
- 9.1 新T3D引擎的特性 498
- 9.2 更新T3D数据结构和设计 499
- 9.2.1 新的#define 499
- 9.2.2 新增的数学结构 501
- 9.2.3 实用宏 502
- 9.2.4 添加表示3D网格数据的特性 503
- 9.2.5 更新物体结构和渲染列表结构 508
- 9.2.6 函数清单和原型 511
- 9.3 重新编写物体加载函数 517
- 9.3.1 更新.PLG/PLX加载函数 517
- 9.3.2 更新3D Studio .ASC加载函数 527
- 9.3.3 更新Caligari .COB加载函数 528
- 9.4 回顾多边形的光栅化 532
- 9.4.1 三角形的光栅化 532
- 9.4.2 填充规则 535
- 9.4.3 裁剪 537
- 9.4.4 新的三角形渲染函数 538
- 9.4.5 优化 542
- 9.5 实现Gouraud着色处理 543
- 9.5.1 没有光照时的Gouraud着色 544
- 9.5.2 对使用Gouraud Shader的多边形执行光照计算 553

- 9.6 基本采样理论 560
 - 9.6.1 一维空间中的采样 560
 - 9.6.2 双线性插值 561
 - 9.6.3 u和v的插值 563
 - 9.6.4 实现仿射纹理映射 564
- 9.7 更新光照/光栅化引擎以支持纹理 566
- 9.8 对8位和16位模式下优化策略的最后思考 571
 - 9.8.1 查找表 571
 - 9.8.2 网格的顶点结合性 572
 - 9.8.3 存储计算结果 572
 - 9.8.4 SIMD 573
- 9.9 最后的演示程序 573
- 9.10 总结 576
- 第10章 3D裁剪 577
 - 10.1 裁剪简介 577
 - 10.1.1 物体空间裁剪 577
 - 10.1.2 图像空间裁剪 580
 - 10.2 裁剪算法 581
 - 10.2.1 有关裁剪的基本知识 581
 - 10.2.2 Cohen-Sutherland裁剪算法 585
 - 10.2.3 Cyrus-Beck/梁友栋-Barsky裁剪算法 586
 - 10.2.4 Weiler-Atherton裁剪算法 588
 - 10.2.5 深入学习裁剪算法 590
 - 10.3 实现视景体裁剪 591
 - 10.3.1 几何流水线和数据结构 592
 - 10.3.2 在引擎中加入裁剪功能 593
 - 10.4 地形小议 611
 - 10.4.1 地形生成函数 612
 - 10.4.2 生成地形数据 619
 - 10.4.3 沙地汽车演示程序 619
 - 10.5 总结 623
- 第11章 深度缓存和可见性 624
 - 11.1 深度缓存和可见性简介 624
 - 11.2 z缓存基础 626
 - 11.2.1 z缓存存在的问题 627
 - 11.2.2 z缓存范例 627
 - 11.2.3 平面方程法 630
 - 11.2.4 z坐标插值 631
 - 11.2.5 z缓存中的问题和1/z缓存 632
 - 11.2.6 一个通过插值计算z和1/z的例子 633
 - 11.3 创建z缓存系统 635
 - 11.4 可能的z缓存优化 649
 - 11.4.1 使用更少的内存 649
 - 11.4.2 降低清空z缓存的频率 650
 - 11.4.3 混合z缓存 651
 - 11.5 z缓存存在的问题 651
 - 11.6 软件和z缓存演示程序 652
 - 11.6.1 演示程序I: z缓存可视化 652
 - 11.6.2 演示程序II: Wave Raider 653
 - 11.7 总结 658
- 第四部分 高级3D渲染
- 第12章 高级纹理映射技术 660
 - 12.1 纹理映射——第二波 660
 - 12.2 新的光栅化函数 667

- 12.2.1 最终决定使用定点数 667
- 12.2.2 不使用z缓存的新光栅化函数 668
- 12.2.3 支持z缓存的新光栅化函数 670
- 12.3 使用Gouraud着色的纹理映射 671
- 12.4 透明度和alpha混合 677
 - 12.4.1 使用查找表来进行alpha混合 678
 - 12.4.2 在物体级支持alpha混合功能 688
 - 12.4.3 在地形生成函数中加入alpha支持 694
- 12.5 透视修正纹理映射和1/z缓存 696
 - 12.5.1 透视纹理映射的数学基础 696
 - 12.5.2 在光栅化函数中加入1/z缓存功能 702
 - 12.5.3 实现完美透视修正纹理映射 707
 - 12.5.4 实现线性分段透视修正纹理映射 710
 - 12.5.5 透视修正纹理映射的二次近似 714
 - 12.5.6 使用混合方法优化纹理映射 718
- 12.6 双线性纹理滤波 719
- 12.7 Mipmapping和三线性纹理滤波 724
 - 12.7.1 傅立叶分析和走样简介 725
 - 12.7.2 创建Mip纹理链 727
 - 12.7.3 选择mip纹理 734
 - 12.7.4 三线性滤波 739
- 12.8 多次渲染和纹理映射 740
- 12.9 使用单个函数来完成渲染工作 741
 - 12.9.1 新的渲染场境 741
 - 12.9.2 设置渲染场境 743
 - 12.9.3 调用对渲染场境进行渲染的函数 745
- 12.10 总结 753
- 第13章 空间划分和可见性算法 754
 - 13.1 新的游戏引擎模块 754
 - 13.2 空间划分和可见性判定简介 754
 - 13.3 二元空间划分 757
 - 13.3.1 平行于坐标轴的二元空间划分 758
 - 13.3.2 任意平面空间划分 759
 - 13.3.3 使用多边形所在的平面来划分空间 760
 - 13.3.4 显示/访问BSP树中的每个节点 762
 - 13.3.5 BSP树数据结构和支撑函数 763
 - 13.3.6 创建BSP树 765
 - 13.3.7 分割策略 767
 - 13.3.8 遍历和显示BSP树 775
 - 13.3.9 将BSP树集成到图形流水线中 784
 - 13.3.10 BSP关卡编辑器 785
 - 13.3.11 BSP的局限性 793
 - 13.3.12 使用BSP树的零重绘策略 794
 - 13.3.13 将BSP树用于剔除 795
 - 13.3.14 将BSP树用于碰撞检测 802
 - 13.3.15 集成BSP树和标准渲染 802
 - 13.4 潜在可见集 807
 - 13.4.1 使用潜在可见集 808
 - 13.4.2 潜在可见集的其他编码方法 809
 - 13.4.3 流行的PVS计算方法 810
 - 13.5 入口 811
 - 13.6 包围体层次结构和八叉树 813
 - 13.6.1 使用BHV树 815

- 13.6.2 运行性能 816
- 13.6.3 选择策略 817
- 13.6.4 实现BHV 818
- 13.6.5 八叉树 825
- 13.7 遮掩剔除 825
 - 13.7.1 遮掩体 826
 - 13.7.2 选择遮掩物 826
 - 13.7.3 混合型遮掩物选择方法 827
- 13.8 总结 827
- 第14章 阴影和光照映射 828
 - 14.1 新的游戏引擎模块 828
 - 14.2 概述 828
 - 14.3 简化的阴影物理学 829
 - 14.4 使用透视图像和广告牌来模拟阴影 832
 - 14.4.1 编写支持透明功能的光栅化函数 833
 - 14.4.2 新的库模块 835
 - 14.4.3 简单阴影 837
 - 14.4.4 缩放阴影 839
 - 14.4.5 跟踪光源 841
 - 14.4.6 有关模拟阴影的最后思考 844
 - 14.5 平面网格阴影映射 845
 - 14.5.1 计算投影变换 845
 - 14.5.2 优化平面阴影 848
 - 14.6 光照映射和面缓存技术简介 848
 - 14.6.1 面缓存技术 850
 - 14.6.2 生成光照图 850
 - 14.6.3 实现光照映射函数 851
 - 14.6.4 暗映射(dark mapping) 853
 - 14.6.5 光照图特效 854
 - 14.6.6 优化光照映射代码 854
 - 14.7 整理思路 854
 - 14.8 总结 854
- 第五部分 高级动画、物理建模和优化
- 第15章 3D角色动画、运动和碰撞检测 858
 - 15.1 新的游戏引擎模块 858
 - 15.2 3D动画简介 858
 - 15.3 Quake II .MD2文件格式 859
 - 15.3.1 .MD2文件头 861
 - 15.3.2 加载Quake II .MD2文件 868
 - 15.3.3 使用.MD2文件实现动画 874
 - 15.3.4 .MD2演示程序 882
 - 15.4 不基于角色的简单动画 883
 - 15.4.1 旋转运动和平移运动 883
 - 15.4.2 复杂的参数化曲线移动 885
 - 15.4.3 使用脚本来实现运动 885
 - 15.5 3D碰撞检测 887
 - 15.5.1 包围球和包围圆柱 887
 - 15.5.2 使用数据结构来提高碰撞检测的速度 888
 - 15.5.3 地形跟踪技术 889
 - 15.6 总结 890
- 第16章 优化技术 891
 - 16.1 优化技术简介 891
 - 16.2 使用Microsoft Visual C++和Intel VTune剖析代码 892
 - 16.2.1 使用Visual C++进行剖析 892

- 16.2.2 分析剖析数据 893
- 16.2.3 使用VTune进行优化 894
- 16.3 使用Intel C++编译器 899
 - 16.3.1 下载Intel的优化编译器 900
 - 16.3.2 使用Intel编译器 900
 - 16.3.3 使用编译器选项 901
 - 16.3.4 手工为源文件选择编译器 901
 - 16.3.5 优化策略 902
- 16.4 SIMD编程初步 902
 - 16.4.1 SIMD基本体系结构 903
 - 16.4.2 使用SIMD 903
 - 16.4.3 一个SIMD 3D向量类 912
- 16.5 通用优化技巧 918
 - 16.5.1 技巧1: 消除 `ftol()` 918
 - 16.5.2 技巧2: 设置FPU控制字 918
 - 16.5.3 技巧3: 快速将浮点变量设置为零 919
 - 16.5.4 技巧4: 快速计算平方根 919
 - 16.5.5 技巧5: 分段线性反正切 920
 - 16.5.6 技巧6: 指针递增运算 920
 - 16.5.7 技巧7: 尽可能将if语句放在循环外面 921
 - 16.5.8 技巧8: 支化(branching)流水线 921
 - 16.5.9 技巧9: 数据对齐 921
 - 16.5.10 技巧10: 将所有简短函数都声明为内联的 922
 - 16.5.11 参考文献 922
- 16.6 总结 922
- 第六部分 附录
 - 附录A 光盘内容简介 CD: 924
 - 附录B 安装DirectX和使用Visual C/C++ CD: 925
 - B.1 安装DirectX CD: 925
 - B.2 使用Visual C/C++编译器 CD: 925
 - B.3 编译提示 CD: 926
 - 附录C 三角学和向量参考 CD: 927
 - C.1 三角学 CD: 927
 - C.2 向量 CD: 929
 - C.2.1 向量长度 CD: 930
 - C.2.2 归一化 CD: 930
 - C.2.3 标量乘法 CD: 930
 - C.2.4 向量加法 CD: 931
 - C.2.5 向量减法 CD: 931
 - C.2.6 点积 CD: 932
 - C.2.7 叉积 CD: 933
 - C.2.8 零向量 CD: 934
 - C.2.9 位置向量 CD: 934
 - C.2.10 向量的线性组合 CD: 934
 - 附录D C++入门 CD: 935
 - D.1 C++是什么 CD: 935
 - D.2 必须掌握的C++知识 CD: 937
 - D.3 新的类型、关键字和约定 CD: 937
 - D.3.1 注释符 CD: 937
 - D.3.2 常量 CD: 937
 - D.3.3 引用型变量 CD: 938
 - D.3.4 即时创建变量 CD: 938
 - D.4 内存管理 CD: 939
 - D.5 流式输入/输出 CD: 939

D.6 类 CD: 941
D.6.1 新结构 CD: 941
D.6.2 一个简单的类 CD: 942
D.6.3 公有和私有 CD: 942
D.6.4 类的成员函数(方法) CD: 943
D.6.5 构造函数和析构函数 CD: 944
D.6.6 编写构造函数 CD: 945
D.6.7 编写析构函数 CD: 946
D.7 域运算符 CD: 947
D.8 函数和运算符重载 CD: 948
D.9 基本模板 CD: 950
D.10 异常处理简介 CD: 951
D.11 总结 CD: 954
附录E 游戏编程资源 CD: 955
E.1 游戏编程和新闻网站 CD: 955
E.2 下载站点 CD: 955
E.3 2D/3D引擎 CD: 956
E.4 游戏编程书籍 CD: 956
E.5 微软公司的Direct X 多媒体展示 CD: 956
E.6 新闻组 CD: 957
E.7 跟上行业的步伐 CD: 957
E.8 游戏开发杂志 CD: 957
E.9 Quake资料 CD: 957
E.10 免费模型和纹理 CD: 957
E.11 游戏网站开发者 CD: 957
附录F ASCII码表 CD: 959
• • • • • ([收起](#))

[3D游戏编程大师技巧_下载链接1](#)

标签

游戏开发

3d

游戏编程

图形学

游戏

3D游戏编程大师技巧

计算机

编程

评论

看目录我就兴奋了。

好书！因为要还了，没能全部读完；数学基础方面建议阅读《3D数学基础》

组会轮讲，每周每人一章。感觉东西有点旧，所以读着读着就乏味了，也有可能是因为电子版的书影印质量太差，所以没什么感觉。不过讲得挺细致的，什么流水线啊剪裁Z缓冲之类，也不知道哪些技术过时了哪些是保留下来的，总之我也记得的不多了。

本书的核心就是讲3D绘制的流水线。最爽的就是一步步将软光栅给实现出来。这样很多概念就具体化了。边看原理边看代码，是工程师绝佳的学习方式！写软光栅是图形学入门的第一步。当然，不推荐入门者直接写，这里面知识点太多，不好把握分寸。这也是本书存在的意义！900多页的书，看了一个多月，要有点耐心。

书是好书,但只是有点旧

书绝对是经典，不过内容略老了点。最近正好做个类似的开源项目，就趁没事的时候翻完了。

我一定要找个时间认真读完这本书~~以前看了1/3左右，但是感觉已经受益匪浅了

介绍详细，示例丰富，翻译质量还算凑合。

重读第二遍

太厚了....

就是内容太老了

最好的3d软渲染书籍，没有之一

全。

详细讲述3D软引擎开发，非常优秀的教材。

2009年在读过《Windows游戏编程大师技巧》后看的。对于理解固定渲染管线很有好处。自己模仿写了个软件渲染器...

写得非常好，读了过后可以轻松写一个软件光栅器。作者是用C写的，代码比较啰嗦，文字也比较啰嗦，可能对新人友好点吧，其实书的信息量不大，关键就只是个光栅化算法。

how to explanation my feeling about this book...

大致翻了一遍，对3d引擎有了一定了解，没敲代码，还要读第二遍，跟着代码读

介绍的部分技术有些过时了

虽然时间很久远，但是里面的优化方法，和管线流程，万变不离其宗，看过之后受益匪浅。本文目的是搭建软渲染器，对gpu部分讲解较少

[3D游戏编程大师技巧 下载链接1](#)

书评

书的切入点还是不错的，毕竟市面上很少有专门教人写soft render的书，但是如果你确实想自己动手写一个图形管线，不建议完全照着书上的方法和代码做。因为这本书的年代实在是有点久远，书中居然在用8 bit的颜色模式..这玩意在我的机器上都无法正常运行。而且书非常厚(1000多页)，...

这本书已经绝版很久了,在学校的图书馆里觅得.
作为一个想向3D游戏发展的IT业新人,我觉得这本书十分值得一看,
对基础原理介绍还算十分详细,
例如网上基本搜不到关于1/zbuffer在屏幕空间中呈线性的证明,这本书12章有两种证法,
并且对各种高级渲染技术从算法的角度给读者解释,...

人民邮电出版社即将重新出版《3D游戏编程大师技巧》，预计2012年6、7月份上市。
最新信息请关注新浪微博@人民邮电出版社-信息技术分社
人民邮电出版社即将重新出版《3D游戏编程大师技巧》，预计2012年6、7月份上市。
最新信息请关注新浪微博@人民邮电出版社-信息技术分社

自己是做AS3的，2011年初据说Flash可以支持3D了，蛋疼，想要了解一下最基本的3D理论知识。
各方搜索后得出这本书比较适合自身水平阅读，还有大量的代码可以参考。于是乎花了大概1个月的时间翻了一遍。感触良多。
首先3D引擎果然不是我们这些低智商的写AS3的人可以写出来的 其...

几年前读了这本书，对于一个想要弄懂影视和游戏那些炫目的3D世界背后的原理这本书可以堪称始祖级教材，首先这本书厚的出奇，肯定会吓读者一跳，也考验读者的意志力，这可不是什么人可以随便看看的书哦，除了兴趣还是兴趣才能支撑你看完这本书并读完每一行代码，看完后你...

读完这本书，通过软件渲染原理，理解硬件渲染管线都做了什么。更深入的可以多学习各种优化知识理论。不过因为书本年代久远，不推荐新手阅读，至少需要学习过计算机图形学。并且对现代渲染管线有一定的了解，再去阅读本书效果更佳。对于程序员来说，按照个人对内容理解的不同对...

对每个对3D程序有兴趣的人都应该读一下。里面讲解的东西，基本上是把目前流行的图形API实现了一次，而且解释非常详细，十分值得一读。（读完后，你会更加理解图形管线的，呵呵） ...

几乎是从像素级别介绍如果构建一套软件实现的3D引擎，在市面中强调如何应用OpenGL或DirectX等现成的库的书相比算是个极为厚道的异类。偶啃了一年了，刚刚啃到1/z缓存的那一章。可以说，如果真理解了本书所教授的内容，对于3D引擎的内部机制的图像渲染主题的各方便应该就没...

虽然本书介绍的是软引擎，但是相关道理可以通用的。
优点：简单、知识点基础并且较全、有源代码
虽然本书介绍的是软引擎，但是相关道理可以通用的。
优点：简单、知识点基础并且较全、有源代码

André
LaMothe的这本书立足于3D软件图形加速，说白了就是用CPU而非显卡进行3D图形计算。在3D显示加速卡出现以前，都是利用的软件加速的方式开发。可时过境迁，目前有很多相当成熟的图形API集可供选择如DirectX和OpenGL等。这样的接口利用的是显卡资源进行加速，即硬件加速，这...

关于在xp和win7下运行出现的问题，书上第七章的demo运行后都被压缩成半屏的，wh

y?发现全屏demo就没有问题，why?是不是2d引擎的问题呢？求解答？求解答？求解答？
求解答？求解答？求解答？求解答？求解答？

有想卖的Q我 124740523,
绝版了很难买啊，到处都买不到，谁有闲置的不用卖给我吧，求 OR2... T_T

我看过了 我看过了 我看过了 我看过了 我看过了 我看过了 我看过了 我看过了 我看过了
我看过了 我看过了 我看过了 我看过了 我看过了 我看过了 我看过了 我看过了 我看过了
我看过了 我看过了 我看过了 我看过了 我看过了 我看过了 我看过了 我看过了 我看过了
我看过了 我看...

[3D游戏编程大师技巧_下载链接1](#)