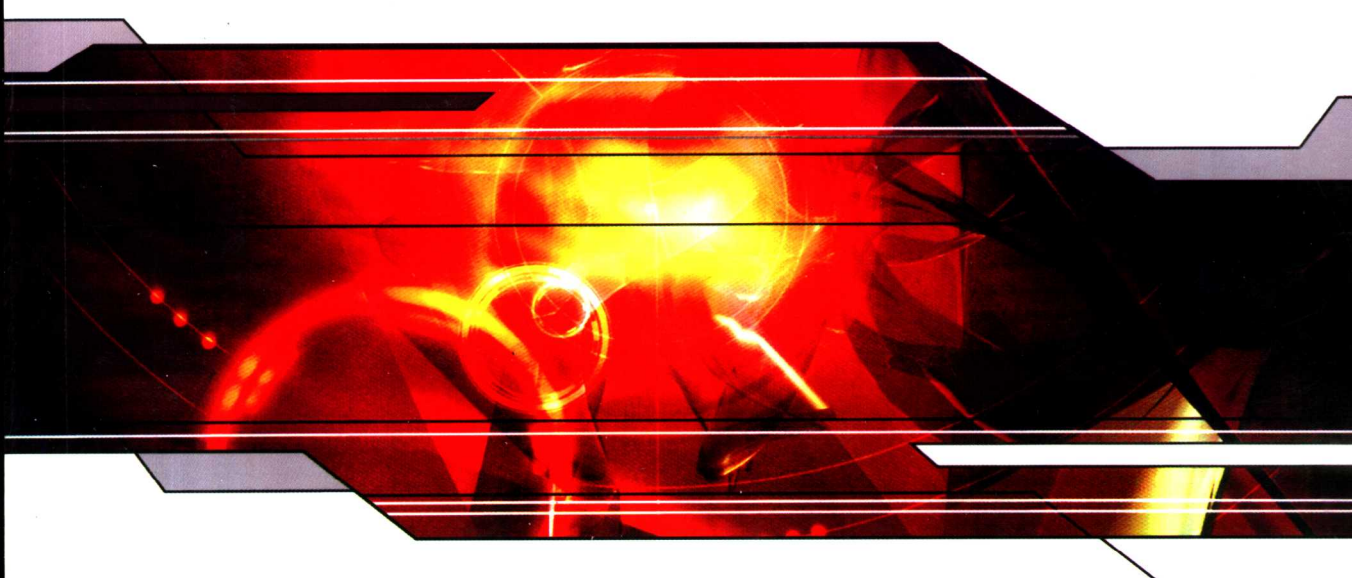




游戏开发与编程系列

Core Techniques and Algorithms in Game Programming

游戏核心算法编程内幕



北京希望电子出版社 总策划

[西班牙] Daniel Sánchez-Crespo Dalmau 编 著

邱仲潘 译

爱亿尔(北京)国际游戏开发院 审 校



中国环境科学出版社
China Environmental Science Press
www.cesp.com.cn



北京希望电子出版社
Beijing Hope Electronic Press
www.bhep.com.cn

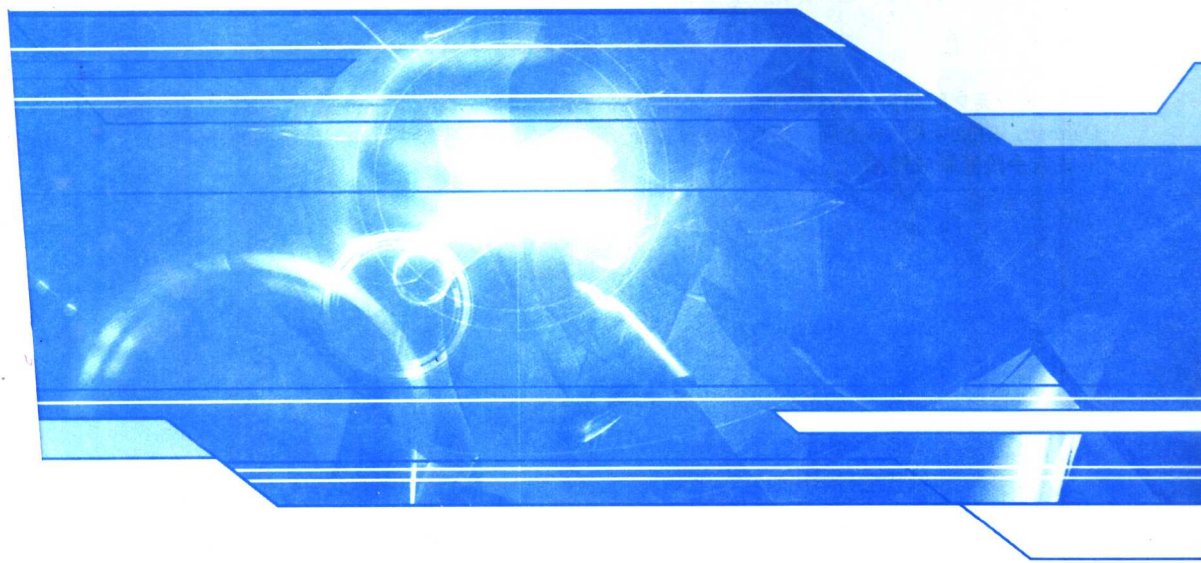
NRG



游戏开发与编程系列

Core Techniques and Algorithms in Game Programming

游戏核心算法编程内幕



北京希望电子出版社 总策划

[西班牙] Daniel Sánchez-Crespo Dalmau 编 著

邱仲潘 译

爱亿尔（北京）国际游戏开发院 审 校

中国环境科学出版社
China Environmental Science Press
www.cesp.com.cn



北京希望电子出版社
Beijing Hope Electronic Press
www.bhp.com.cn

NR

图书在版编目(CIP)数据

游戏核心算法编程内幕 / (西班牙)多尔曼(Dalmau, D.S.)著;邱仲潘译. —北京:中国环境科学出版社, 2004.12

原出版社:培生教育集团分公司

ISBN 7-80163-889-1

I. 游… II. ①多… ②邱… III. 游戏—软件设计—教材 IV. TP311.5

中国版本图书馆CIP数据核字(2004)第067402号

内容提要

本书主要介绍市面上大多数计算机与视频游戏编程的基本核心算法与技术,以及游戏编程理论和许多PC与控制台上的AAA级产品的实现细节,使之成为游戏编程导论课程教材。书中大多数相关方法都有全面解释、框图和必要的代码样本,使读者可以了解幕后工作原理、工作方法和本质。

全书由三部分组成。其中第一部分主要介绍游戏编程的基本概念;第二部分详细介绍游戏编程中的各种技术和算法;第三部分是附录,介绍游戏编程中相关技术和知识以及其他相关读物。

版权声明

本书英文版名为:Core Techniques and Algorithms in Game Programming,作者是Daniel Sánchez-Crespo Dalmau。由New Riders出版,版权归New Riders所有。未经出版者书面许可,本书的任何部分不得以任何形式或手段复制或传播。

书 名 游戏核心算法编程内幕
著 者 [西班牙]Daniel Sánchez-Crespo Dalmau
译 者 邱仲潘
总 策 划 北京希望电子出版社
责 任 编 辑 苏金河 陈光辉 李兴旺 贾卫列
出 版 中国环境科学出版社 北京希望电子出版社
发 行 北京希望电子出版社
地 址 中国环境科学出版社 北京市崇文区广渠门内大街16号(100062)
电话:010-67113409
网址: <http://www.cesp.com.cn> E-mail: cesp@sohu.com
北京希望电子出版社 北京市海淀区上地三街9号金隅嘉华大厦C座610
电话:010-82702660, 010-82702658, 010-62978181 转 103 或者 238 (发行部)
010-82702698 (传真)
经 销 E-mail: yanmc@bhp.com.cn
各地新华书店 软件连锁店
排 版 希望图书输出中心 马君
印 刷 北京媛明印刷厂
版次/印次 2004年12月第1版 2004年12月第1次印刷
开本/印张 787×1092 1/16 33.125 印张
字 数 762 千字
印 数 1~5000 册
书 号 ISBN 7-80163-889-1/TP·028
定 价 58.00 元

作者介绍



Daniel Sánchez-Crespo Dalmau 是西班牙巴塞罗那市 Pompeu Fabra 大学教授，是西班牙第一个视频游戏生成专业硕士点的联合创办者和现任主任，他创办的 Novarama 是巴塞罗那市的独立游戏工作室，为 PC/Xbox 平台生成富有创意的游戏。他是《Byte Magazine Spain》、《Game Developer Magazine》和 Gamasutra web 站点的专栏作家，发表的文章有四十多篇。他的 E-mail 地址为 dsanchez@novarama.com。

技术审校介绍

本书技术审校把丰富的实践经验适用到《游戏核心算法编程内幕》一书的整个开发过程中。本书编写过程中，技术审校审阅了所有材料的技术内容、组织与流程，他的意见保证了《游戏核心算法编程内幕》一书满足读者需求，提供最高质量的技术信息。



Wolfgang Engel 是作家和程序员，著有《Beginning Direct3D Game Programming》(Premier 出版)，编写和编辑了《Direct3D ShaderX: Vertex and Pixel Shader Programming Tips and Tricks》一书 (Wordware 出版)。他还在德文杂志和 gamedev.net 之类联机教程中发表过游戏编程方面的文章。他从事游戏行业期间，从头开始建立了两个游戏开发单位，为欧洲最大的电视节目“Wetten das...?” 开发联机游戏；他还是几家公司的董事会成员或 CEO，是游戏娱乐技术学院的顾问和教员。

致 谢

写书不是独行天下的事，而需要许多人的参与，因此我要感谢对本书作出贡献的每个人。

首先感谢 New Riders 出版公司的 Stephanie Wall 与 Chris Zahn，和 Prentice Hall 出版公司的 John Neidhart，他们的努力与奉献是至为宝贵的。

感谢 Pompeu Fabra 大学的 Josep Blat 富有远见地建立了游戏专业课程，支持创作这本书和建立了欧洲第一个视频游戏生成专业硕士点。

许多人贡献了书中涉及的思想、源代码和其他内容，在此一并致谢（排名不分先后）：

Dave Pottinger, Ensemble Studios 公司

Don Hopkins, Maxis 公司

Ismael Noche, PyroStudios 公司

Andrew Worsey, Codeplay 公司

Cass Everitt, NVIDIA 公司

Juan Guardado, NVIDIA 公司

Alex Dunne, Game Developer/Gamasutra 公司

Aleks Jakulin, Ljubljana 大学人工智能实验室

Ken Perlin, NYU大学

Toni Susín, UPC大学

Anna Puig, UPC大学习

Carla Brossa, La Salle多媒体工程学院

Adam Valdez, WETA Digital公司

Ernest Adams

Guillaume Werle

Martijn van Welie

Pere Fort, UPF大学

Hector Geffner, ICREA/ICREA

最后，感谢Pompeu Fabra大学的每个人，特别是我的学生和同事们。感谢Novarama工作室的同事们提供了与这个项目有关的丰富信息，还要感谢CIDEM/Generalitat de Catalunya支持Novarama游戏工作室的活动。最后，感谢我的所有朋友和亲人，他们给了我很大的支持；还要感谢美丽的Cecilia，感谢她的无条件支持。

2002年3月到2003年6月于巴塞罗那

从游戏中学会生存（总序）

——献给游戏创造者

游戏为什么受到重视

今天，越来越多的人不再把游戏看作小儿科，也不仅仅视其为一门艺术了。这大概要归功于全球每天数亿美元的市场业绩，抑或在金融危机中使一个国家经济转危为安的力挽狂澜作用。作为一种生存途径，日本、韩国深得游戏产业振兴经济的非凡要领。无数名不见经传的小人物靠游戏制作成为百万富翁，游戏经典之作的原创者则受到了超级明星般的待遇而被全世界的游戏玩家追捧着。

当游戏被视为艺术时，人们常常拿他与传统艺术形式作比较。认为游戏作为一门娱乐艺术形式，不同于诗歌、小说、戏剧、电影、电视等等传统艺术形式的根本点是“交互”，是“参与”。不错，游戏作为一个产业勃然兴起，确实离不开计算机和互联网。但就其对人类的根本作用而言，要比交互、参与这类技术层面的问题远为原始得多。

游戏比艺术起源更早

诗歌、小说、戏剧、电影、电视等等艺术的形成，依赖于语言文字的产生，因此其历史见于壁画与甲骨等人类考古学证据的发现，大约可以上溯到距今 5000 年前。

游戏却不仅仅属于人类。小猫玩线球，人们看到的只是小猫的兴趣盎然和乐此不疲，却不见在这种练习中小猫捕捉能力的提高：通过追逐线球，小猫强壮了自己的利爪，敏捷了全身的器官和肌腱。这种对后代的训练形式普遍存在于动物界，其历史至少可以追溯到对哺乳动物的考古学发现，并非最早的证据距今也已经有 6000 万年了。

从生存描述到生存模拟

作为对生存过程的一种描述，诗歌、小说、戏剧、电影、电视等等艺术，是把受体放在媒体之外，通过媒体的艺术感染力，使受体产生共鸣，令其产生沉浸感。

作为对生存过程的一种模拟，游戏则是创造一种虚拟环境，把受体放在媒体之中，使受体自己判断、决策和动作，通过反复操作媒体获得某种能力。数千万年以来，游戏使用着更直接的沉浸手段，让后代在沉浸中得到更实际的生存锻炼。

假如游戏不仅以娱乐为目的

由此可见，作为功能的界定，与游戏更接近的社会部类是教育，而不仅仅或不主要是娱乐。无怪乎国际游戏开发者协会（IGDA）活动负责人杰森·德拉·罗卡（Jason Della Rocca）在 2003 CHINAJOY 大会上的主题发言是《“严肃”游戏：游戏对社会经济的潜在影响》。他把“严肃”游戏（Serious Game）定义为“不以娱乐为主要目的的游戏”，例举了训练市长的《模拟城市》、训练董事长的《虚拟领导》、训练员工的《直言者》、训练海军陆战队员的《DOOM》等等经典游戏作品。在这一方面，美国确实走在了全世界的前面。早在 1994 年，美国海军陆战队就成立了世界上第一个 Game 军事训练机构；1995 年，美国空军和陆军紧随其后，把游戏作为军队训练的有效辅助手段。总结这些经验，罗卡先生列出了游戏的适用领域依次为：

- 教育
- 国家政策
- 卫生保健
- 企业培训
- 其他

游戏可以创造未来

随着计算机的更加移动化、个性化，虚拟社会与现实社会的界限愈加模糊了。如今网游交易市场上的商业链，与现实市场上的买卖规则何其一致！试想，当我们的手机、腰带、手表甚至纽扣全都可以无线上网之后，谁还能够分清现实世界与虚拟世界的截然界限呢？

在虚拟世界与现实世界的直接联系中，虚拟世界的创造者可以把游戏里发展起来的生存新规则，通过游戏渠道传播到更大的范围，影响更多的人。换言之，游戏的创造者可以创造更加理想的未来。这就是：

- 改变工作生活方式
- 影响现有科学技术
- 增强综合国力
- 造福于社会

创造者在游戏中永生

回首 12 年前，中国最早的一批拓荒者进入了游戏开发领域。那时他们相信凭借坚定的信念和努力付出就可以确保成功。然而，游戏也是一种商业，他像其他行业一样受到财务、市场和管理等等条件的制约。在软件盗版和决策失误、管理失措的多重夹击下，许多人付出了惨重代价。然而这些付出换来的是生存经验的积累和生存能力的提高。在经过最初的游戏狂热到现在国际游戏市场的供不应求之后，更多的游戏开发从业者学会了在激烈竞争和快速变化的市场中生存。在游戏产业中，似乎并不遵守“胜者王侯败者寇”的法则，美酒的甘甜和血汗的腥涩都可以收入博物馆，正因为他们是历史的拓荒者。

今天，互联网给游戏带来了蓬勃发展的春天。大量资金裹挟着人才和技术进入游戏开发领域。然而，仿效、重复也跟着多起来了，类同题材和近似玩法的游戏产品充斥市场。巨大的风险潜藏随之，二八铁律决定了绝大多数仿效者的时空有限。因此，要想在游戏中获得永生，唯有创造，别无他途。

希望这套出自世界游戏创造高手的丛书能对梦想创造者有所帮助。

杨南征

爱亿尔（北京）国际游戏软件开发院院长
工商管理博士、作战模拟研究员

2004 年 6 月 15 日

前 言

多年来,关于游戏编程的许多奇思妙想都是在便条上写成的,这方面的知识是非正式交流的,开发人员在开会或聚会时相互切磋,或与其他开发人员口头交流。这一切都是相当缺乏组织的。许多年过去了,如今,编制游戏代码比过去复杂了许多。逼真和沉浸感的增加使开发小组的规模、预算和时间表不断增大。这是游戏开发人员值得激动的时代。

但是,具有这种规模和重要性的行业不能再缺乏组织。如今,必须有高效的专业游戏开发和可靠的开发人员,而不能因为一个特性无法如期工作或不知道某个效果如何编码而使整个项目失败。

游戏行业要用结构化的方式培训新的专业人员。我们的地位正在改变。游戏开发专业会议之类的事件、《游戏开发者》(Game Developer)之类的杂志、国际游戏开发者协会(IGDA)之类的组织和遍布全球的游戏开发学校使人们看到,游戏开发是个切实可行的职业,需要自己的学科和培训。但是,仅仅有这些是不够的。每年推出的大量游戏类文档令人震惊,学习游戏开发的人很难找到可靠的参考资料。许多书籍介绍API编程,通常与游戏开发关系不大。有些书籍是来自其他学科的,如图形学与人工智能。有些书籍有很好的学术价值与深度,但缺乏针对游戏开发的材料。结果,我们要从许多不同地方收集信息,而没有一个中心信息源。

本书正是在这种情况下产生的。作者在游戏编程教学方面有6年经验,是欧洲第一个视频游戏生成专业硕士点的创办人和现任主任,参与了培育游戏开发人员的重要课程。我向学生讲授任何游戏开发人员都应掌握的核心课题与技术,使他们能在很短的适应期内尽快进入游戏工作室并取得成功。本书的目的正在于此——以内聚和结构化的方式介绍视频游戏编程的基本课题。我的宗旨是提供一本优秀的参考书,使人们能够了解游戏的编码与设计方法。

因此,本书的重点不是介绍一个个案例,因为这方面的书很多;不是介绍特定类型的游戏,因为这些方法通常适用于不同类型的游戏;不是一本理论书籍,而是介绍现成的广为接受的方法,是当今许多重磅游戏的核心。本书可以帮助你奠定基础,使你了解大多数游戏项目中的所有基本算法(稍作变化)。

本书内容

本书介绍游戏编程,目的是介绍市面上大多数计算机与视频游戏的基本算法与技术。介绍游戏编程理论和许多PC与控制台上的AAA级产品实现细节。本书的重点不是图形、网络等特定领域,而是介绍每个领域的核心算法,使本书成为游戏编程导论课程教材。

我不准备把本书写成简单的算法集合或长长的技术评审报告,而是要使这些技术更容易理解和编码,使你更容易运用这些技术。因此,大多数相关方法都会有全面解释、框图和必要的代码样本,使读者可以了解幕后工作原理、工作方法和本质。如果你能够实现书中介绍的思想,由此创建下一代游戏,则本书的创作意图已经成功实现。

也许读完本书之后,你的脑海中会出现更多问题。游戏编程是个大课题,而且是不断演变的。如果一本书要介绍从过去到现在的所有技术,则需要好几千页。因此,本书末尾在附录E“更多读物”中推荐更多读物,读者可以在感兴趣的领域中进一步钻研。我把更多读物按章归类,使读者可以方便地找到相关信息。

预备知识

游戏编程是巨大的跨学科领域，涉及数学、一般编程、人工智能等学科。尽管书中会介绍其中许多知识，但为了更好地掌握书中的知识，读者还是需要一定的预备知识。

首先，本书假设读者能够用C/C++编程语言熟练编程，能够在代码中使用包括语句、指针、类、数据结构、继承等。本书不是介绍一般编程的，而是介绍程序的特定用法。第3章“数据结构与算法”将介绍一些常用数据结构，但仅此而已。另一方面，你不需要知道如何进行游戏编程，因为本书会介绍游戏编程。

另一个要求是一些数学知识，相当于许多大学的线性代数和微积分。读者要熟悉向量运算、微分与积分、矩阵、三角学等。如果你已经忘记这些知识，可以阅读附录D“数学知识”，其中列出了书中用到的大多数公式与数学方法，一定要浏览一遍。

此外，还需要时间，这是最重要的。许多人以为游戏编程是很难的，代码要用到一些古怪的语法，只有专家才能明白。尽管游戏编程的有些部分的确很复杂，但大多数源代码很容易理解，复杂之处是很容易确定的某些特定部分。那么，游戏编程为什么这么难呢？不是它的难度大，而是要花很多时间才能达到理想结果。只要用功努力，可以生成许多游戏，通常不会特别难，但游戏编程需要时间，因此一定要有充裕的时间。

本书组织

从纯技术角度看，游戏编程可以分为两大部分：大量代码涉及玩游戏的问题：控制交互、人工智能、脚本等，这是游戏的逻辑段，放在本书第一部分介绍；然后我们介绍图形、音频和其他各种技术，使游戏成为丰富的多媒体体验。本书第一部分主要介绍普通的C/C++代码，第二部分要用到各种库和工具，如OpenGL与DirectX。下面简要介绍本书的内容：

第一部分：游戏编程

第1章“游戏编程时间表”介绍历史背景，简要介绍游戏开发行业的演变过程，并提供平台例子。

第2章“游戏体系结构”、第3章“数据结构与算法”和第4章“设计模式”介绍宏观代码分析，介绍的课题包括游戏代码结构、常用数据结构、设计模式等。

第5章“用户输入”介绍如何处理玩家交互。

第6章“基本人工智能技术”、第7章“面向动作的人工智能”、第8章“战术性人工智能”和第9章“脚本”介绍人工智能课题，包括基本算法、动作与战略人工智能、人工生命和脚本技术等。

第10章“网络编程”详细介绍联网游戏的问题，从客户机和服务器角度介绍多方编程和大量用户的多方游戏。

第二部分：引擎编程

第11章到第15章“二维游戏编程”、“三维管道概述”、“室内绘制”、“户外算法”和“角色动画”介绍图形引擎课题，如绘制管道，首先介绍一般性设计，然后介绍室内与户外绘制算法。

第16章到第18章“制片术”、“色档”和“纹理贴图”介绍外观，包括设置镜头与灯光

和对几何形体采用不同纹理技术。

第 19 章到第 21 章“微粒系统”、“有机构绘制”和“过程性技术”处理复杂绘制情形，介绍三个重要课题：微粒系统、自然模拟的有机体绘制和过程性/色档方法。

第 22 章“几何算法”介绍几何测试、碰撞探测、几何简化等。

第三部分：附录

附录 A “性能调整”介绍优化，介绍性能调整技术，保证游戏达到最佳性能。

附录 B “OpenGL”与附录 C “Direct3D”介绍 API，介绍 OpenGL 1.4 与 DirectX 9。

附录 D “数学知识”介绍游戏编程中用到的一些数学知识。

附录 E “更多读物”提供更多读物，使读者可以对特定课题进一步深化、钻研。

本书翻译过程中得到了刘文红、周阳生、邹能东、彭振庆、黄志坚、李耀平、郭王旋等同志的大力帮助，刘文琼、黄素平、李富招等同志完成了本书的录入工作，刘云昌、刘昌和兄弟帮助进行了书稿与打印稿的校对，在此深表感谢。

目 录

第一部分 游戏编程

第1章 游戏编程时间表	2
1.1 第一阶段：“星球大战”之前	2
1.2 第二阶段：从“星球大战”到 Atari	5
1.3 第三阶段：游戏控制台与个人计算机	6
1.3.1 游戏控制台与游戏开发人员	6
1.3.2 个人计算机	8
1.4 第四阶段：调整与汇总	9
1.5 第五阶段：游戏引擎的出现	11
1.6 第六阶段：手指式革命	14
1.7 第七阶段：移动现象	15
1.8 第八阶段：多方游戏	16
1.9 结束语	17
第2章 游戏体系结构	19
2.1 实时软件	19
2.2 游戏逻辑段	24
2.2.1 玩家更新	24
2.2.2 世界更新	25
2.3 显示段	26
2.3.1 世界绘制	27
2.3.2 NPC 绘制	27
2.3.3 玩家	28
2.3.4 问题：网络游戏	29
2.4 编程过程	30
关卡	30
2.5 结束语	36
第3章 数据结构与算法	37
3.1 类型、结构和类	37
3.2 数据结构	39
3.2.1 静态数组	39
3.2.2 链表	40
3.2.3 双向链表	41
3.2.4 队列	42
3.2.5 堆栈	43

3.2.6 双头队列	43
3.2.7 表	44
3.2.8 树	47
3.2.9 优先队列	51
3.2.10 图	52
3.3 标准模板库	54
3.3.1 容器	55
3.3.2 迭代器	58
3.4 结束语	59
第4章 设计模式	60
4.1 设计模式定义	60
4.2 一些重要编程模式	61
4.2.1 单体模式	61
4.2.2 策略模式	62
4.2.3 工厂模式	64
4.2.4 空间索引	66
4.2.5 复合模式	69
4.2.6 轻量级模式	72
4.3 可用性模式	74
4.3.1 屏蔽模式	74
4.3.2 状态模式	75
4.3.3 自动方式取消	75
4.3.4 磁化	75
4.3.5 焦点模式	75
4.3.6 进程模式	76
4.4 结束语	76
第5章 用户输入	77
5.1 键盘	77
5.2 鼠标	82
5.3 游戏杆	85
5.4 硬件抽象	90
5.5 力量反馈	92
5.6 结束语	93
第6章 基本人工智能技术	94
6.1 情境	94

6.2 人工智能系统结构	95
6.2.1 感知世界	96
6.2.2 记忆	96
6.2.3 分析/推理核心	96
6.2.4 动作/输出系统	97
6.3 特定技术	97
6.3.1 有限状态机	97
6.3.2 规则系统	106
6.3.3 规划与解题	113
6.3.4 生物灵感人工智能	115
6.4 结束语	119
第7章 面向动作的人工智能	120
7.1 动作游戏	120
7.2 动作设计人工智能	120
7.3 对象跟踪	122
7.3.1 眼睛接触:二维半平面测试	122
7.3.2 三维版本:半球	124
7.4 追赶	125
7.4.1 二维追赶:固定速度	125
7.4.2 预测性追赶	126
7.5 逃避	127
7.6 巡逻	128
7.7 隐藏与遮盖	128
7.8 射击	129
7.8.1 无限速度瞄准	129
7.8.2 实际瞄准	130
7.8.3 机关枪	131
7.9 综合起来	132
7.9.1 并行自动化	132
7.9.2 AI 同步	133
7.10 结束语	134
7.10.1 平台游戏	134
7.10.2 射击游戏	135
7.10.3 格斗游戏	136
7.10.4 赛车游戏	136
第8章 战术性人工智能	137
8.1 战术性思维	137
8.1.1 路径寻找	138
8.1.2 团队活动	147

8.2 军事分析:影响图	149
8.2.1 数据结构	149
8.2.2 一些重要测试	150
8.3 表示战术	151
8.4 结束语	154
第9章 脚本	155
9.1 建立脚本语言	156
9.1.1 简单分析语言	156
9.1.2 分析结构化语言	162
9.2 嵌入语言	165
9.2.1 学习 Lua	165
9.2.2 Java 脚本	171
9.3 基于套接的脚本	173
9.4 结束语	175
第10章 网络编程	176
10.1 Internet 的实际工作	176
10.2 从编程人员角度看套接	177
10.3 客户机	178
10.3.1 简单 TCP 客户机	178
10.3.2 简单 UDP 客户机	181
10.4 简单 TCP 服务器	183
10.5 多客户机服务器	185
10.5.1 并发的面向连接服务器	185
10.5.2 迭代的面向连接服务器	187
10.6 UDP 服务器	188
10.7 防止套接受阻	189
10.8 设计客户机/服务器游戏	190
10.9 大量用户的多方游戏	191
10.9.1 数据外推	191
10.9.2 层次消息	193
10.9.3 空间分割	193
10.9.4 只发送状态改变	194
10.9.5 使用服务器群集	195
10.9.6 动态服务器与勇士综合症	195
10.10 结束语	196

第二部分 引擎编程

第11章 二维游戏编程	198
11.1 旧式硬件	198

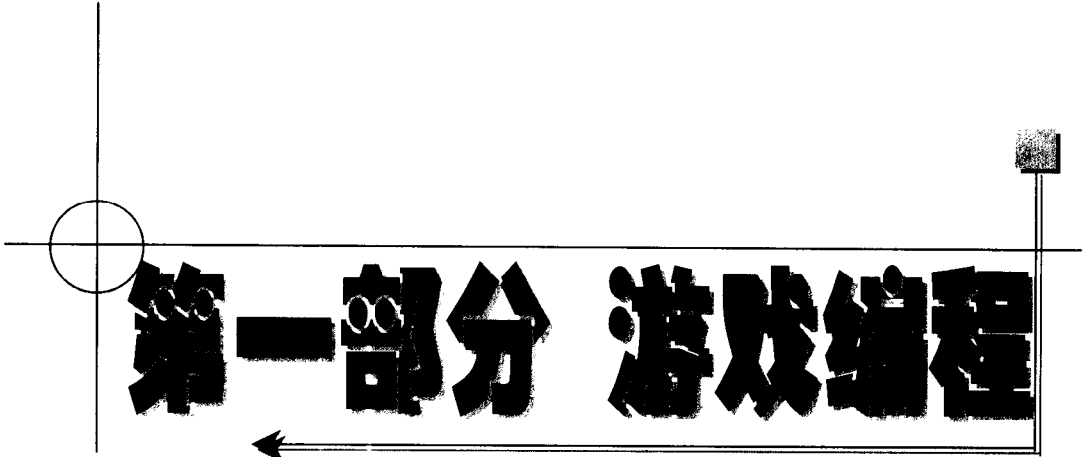
11.2	二维游戏的数据结构.....	199	13.3.4	闭合探测.....	252
11.3	贴图矩阵.....	201	13.3.5	绘制.....	253
	地砖表.....	202	13.4	门户绘制.....	254
11.4	二维游戏算法.....	203	13.5	层次式闭合图.....	258
11.4.1	屏幕游戏.....	203	13.6	混合方法.....	259
11.4.2	双向与四向卷轴.....	204	13.6.1	门户/八叉树混合.....	259
11.4.3	多方引擎.....	206	13.6.2	四叉树/BSP 混合.....	259
11.4.4	视差卷轴.....	207	13.7	硬件辅助闭合测试.....	260
11.4.5	各向异性引擎.....	208	13.8	结束语.....	262
11.4.6	翻页卷轴.....	210	第 14 章	户外算法.....	263
11.5	特殊效果.....	211	14.1	概述.....	263
11.5.1	调色板效果.....	211	14.2	户外绘制的数据结构.....	264
11.5.2	点刻效果.....	215	14.2.1	高度域.....	264
11.5.3	灰贴.....	216	14.2.2	四叉树.....	265
11.5.4	火.....	216	14.2.3	二叉三角形树.....	266
11.6	结束语.....	217	14.3	几何贴图.....	267
第 12 章	三维管道概述.....	218	14.4	ROAM.....	269
12.1	简介.....	218	14.4.1	第一遍:构造变差树.....	269
12.2	基本数据类型.....	219	14.4.2	第二遍:网格重构.....	270
12.2.1	顶点.....	219	14.4.3	优化.....	272
12.2.2	索引图元.....	219	14.5	块 LOD.....	273
12.2.3	颜色.....	223	14.6	以 GPU 为中心的方法.....	275
12.2.4	纹理贴图.....	223	14.7	户外视景图.....	277
12.3	几何格式.....	224	14.8	结束语.....	278
12.4	一般性图形管道.....	225	第 15 章	角色动画.....	279
12.4.1	剪取.....	226	15.1	分析.....	279
12.4.2	挑选.....	230	15.2	显式与隐式方法.....	279
12.4.3	闭合测试.....	232	15.3	显式动画技术.....	280
12.4.4	确定分辨率.....	234	15.3.1	帧动画.....	281
12.4.5	变换与灯光.....	237	15.3.2	关键帧动画.....	281
12.4.6	光栅化.....	238	15.3.3	标记插值.....	283
12.5	结束语.....	240	15.4	隐式动画概述.....	286
第 13 章	室内绘制.....	241	15.4.1	正向动力学.....	287
13.1	一般分析.....	241	15.4.2	框架动画的数学.....	288
13.2	基于闭合的算法.....	242	15.4.3	硬件辅助框架动画.....	289
13.3	二叉空间划分算法.....	243	15.5	道具处理.....	292
13.3.1	构造.....	243	15.6	关于机车.....	293
13.3.2	视图相关排序.....	250	15.7	四肢分解.....	293
13.3.3	层次式剪取.....	251	15.8	面部动画.....	294

15.9 逆向动力学	295	18.1 纹理类型	351
15.9.1 分析逆向动力学	296	18.1.1 纹理贴图	352
15.9.2 循环坐标派生	298	18.1.2 XYZ 贴图	353
15.10 混合正向与逆向动力学	299	18.1.3 圆柱贴图	353
15.11 结束语	299	18.1.4 球形贴图	353
第 16 章 制片术	300	18.1.5 三角形纹理贴图	354
16.1 第一人称视角射击游戏	300	18.2 平铺与贴花	355
16.2 处理惯性	301	18.3 过滤	355
16.3 飞行模拟器与四元数	303	18.4 Mip 贴图	356
16.4 第三人称视角镜头	307	18.5 纹理优化	357
16.5 制片镜头: 镜头样式	311	18.5.1 纹理压缩	359
16.6 制片镜头: 位置算法	314	18.5.2 纹理缓存与分页	359
16.6.1 选择镜头目标	315	18.6 多遍方法	360
16.6.2 选择相关信息	315	18.7 多纹理	363
16.6.3 选择视图角度	317	18.8 纹理运算与组合	364
16.7 基于 agent 的方法	317	18.9 细节纹理	370
16.8 结束语	318	18.10 环境贴图	370
第 17 章 色档	319	18.11 鼓包贴图	371
17.1 实际照明	319	18.11.1 浮雕鼓包贴图	371
17.1.1 简单绘制方程	320	18.11.2 Dot3 鼓包贴图	372
17.1.2 按顶点与按像素照明	322	18.12 上光贴图	373
17.2 灯光贴图	323	18.13 结束语	374
17.2.1 扩散贴图	323	第 19 章 微粒系统	375
17.2.2 镜面贴图	323	19.1 微粒系统剖析	375
17.2.3 使用灯光图的全局照明	324	19.2 微粒数据结束	376
17.2.4 实现光贴图: DirectX	325	19.2.1 一般性微粒系统	377
17.2.5 用 DirectX 实现灯光贴图	326	19.2.2 派生微粒	377
17.2.6 生成灯光贴图	332	19.2.3 微粒行为	379
17.3 BRDF	335	19.2.4 微粒消除	382
17.3.1 平均向量	339	19.2.5 绘制微粒	382
17.3.2 阴影	339	19.3 关于体系结构	384
17.4 非照片真实绘制	347	19.4 加速技术	386
17.4.1 铅笔绘制	347	19.4.1 避免内存分配与释放	386
17.4.2 外形线绘制	348	19.4.2 空间索引	387
17.4.3 划线外形	348	19.4.3 LOD 微粒系统	387
17.4.4 单帧色档	349	19.4.4 基于色档的微粒系统	388
17.4.5 油画绘制	349	19.5 结束语	388
17.5 结束语	350	第 20 章 有机构绘制	389
第 18 章 纹理贴图	351	20.1 自然与细节	389

20.2 树	390	22.1.1 球上的点	424
20.2.1 告示牌	390	22.1.2 AABB 中的点	425
20.2.2 基于图像方法	391	22.1.3 凸多边形的点	426
20.2.3 平行 IBR 方法	391	22.1.4 多边形的点 (凹与凸): 约当法	427
20.2.4 正交 IBR 方法	392	22.1.5 曲线定理	427
20.3 草	394	22.1.6 凸实体的点	428
20.3.1 分层草	395	22.1.7 实体的点 (约当曲线定理)	428
20.3.2 统计分布算法	395	22.2 光束相交测试	429
20.4 云	397	22.2.1 光束平面	429
20.4.1 天框与园顶	397	22.2.2 光三角形	430
20.4.2 招贴板云	397	22.2.3 光 AABB 测试	430
20.4.3 容积云	397	22.2.4 光球测试	431
20.5 海洋	398	22.2.5 光凸形外壳	432
20.5.1 逼真海洋几何	398	22.2.6 光一般实体 (三维 DDA)	432
20.5.2 海洋外观	399	22.3 运动测试	432
20.5.3 焦散性	399	22.4 点与三角形集合碰撞 (基于 BSP)	434
20.6 结束语	402	22.5 网格与网格 (清扫与修剪方法)	434
第 21 章 过程性技术	403	22.6 计算凸形外壳	435
21.1 过程性列表	403	22.6.1 二维方案	435
21.2 Renderman	404	22.6.2 三维方案	436
21.3 实时色档语言	407	22.7 三角形简化	436
21.3.1 当前语言	408	22.7.1 顶点重合	437
21.3.2 Cg	408	22.7.2 边重合	437
21.3.3 HLSL	408	22.7.3 递进网格	437
21.3.4 GL2 色档语言	408	22.7.4 非保守式三角形简化	438
21.4 色档程序类型	408	22.8 结束语	440
21.4.1 色档程序集合	409		
21.4.2 几何效果	409		
21.4.3 灯光	410		
21.5 纹理贴图	413		
21.6 微粒系统	415		
21.7 动画	416		
21.8 特殊效果	421		
21.9 结束语	423		
第 22 章 几何算法	424		
22.1 点包括测试	424		

第三部分 附录

附录 A 性能调整	442
附录 B OpenGL	458
附录 C Direct3D	484
附录 D 数学知识	500
附录 E 更多读物	510



第一部分 游戏编程

- 第 1 章 游戏编程时间表
- 第 2 章 游戏体系结构
- 第 3 章 数据结构与算法
- 第 4 章 设计模式
- 第 5 章 用户输入
- 第 6 章 基本人工智能技术
- 第 7 章 面向动作的人工智能
- 第 8 章 战术性人工智能
- 第 9 章 脚本
- 第 10 章 网络编程

第 1 章 游戏编程时间表

关键主题

- 第一阶段：“星球大战”之前
- 第二阶段：从“星球大战”到 Atari
- 第三阶段：游戏控制台与个人计算机
- 第四阶段：调整与汇总
- 第五阶段：游戏引擎的出现
- 第六阶段：手指式革命
- 第七阶段：移动现象
- 第八阶段：多方游戏
- 结束语

“最初创造了宇宙，使许多人非常愤怒，认为是个坏事。”

——道格拉斯·亚当斯

游戏开发的艺术和科学已经比早期大有进步。硬件已经提高了几个数量级，游戏复杂度和丰富程度也大大增加。要了解当前的游戏编程方法及其原因，就应回顾一下游戏编程的历史。我把游戏编程的历史分成 8 个阶段，下面一个阶段一个阶段地介绍，看看是怎么到达今天这个样子的，从而更好地理解游戏编程的现状是怎么形成的。我的叙述主要集中在编程方面。没有对比的原始数据是毫无用处的，因此一定要提供某种语境来解释数据，了解当前的游戏编程方法是如何建立的。

1.1 第一阶段：“星球大战”之前

视频游戏行业不是按现成规划建立的，而是不同国家的独立人群在几乎同一时间产生了相似的念头。这些传统游戏生成人员想让业务多样化，技术公司利用新发现的固态技术，一些独立的先见之士想象出新的娱乐形式，这一切在多年以后演变成今天的视频游戏行业。

稍后将会介绍，视频游戏行业萌芽于 70 年代初期，但这一时期起重要作用的大部分公司都是在更早的时候创办的。这些公司来自于不同行业和部门，但最终都认识到电子设备上玩游戏可以赚钱，从而形式了新的业务部门。

例如日本的任天堂公司是游戏行业的巨头。1889 年，Fusajiro Yamauchi 建立了这个传统游戏公司，最初注册为 Marufuku 公司，核心业务是生产和销售一种称为 Hanafuda 的日本牌，如图 1.1 所示。1951 年，Marufuku 公司更名为任天堂牌业公司。任天堂的意思就是“把好运留在天堂”。后来，随着第一个电子游戏和设备的出现，任天堂牌业建立了电子游戏部。随着时间的推移，传统游戏业务不断衰退，任天堂公司慢慢成为电子游戏行业的巨头。因此，任天堂牌业公司是根据新兴技术改变经营范围的一个范例。