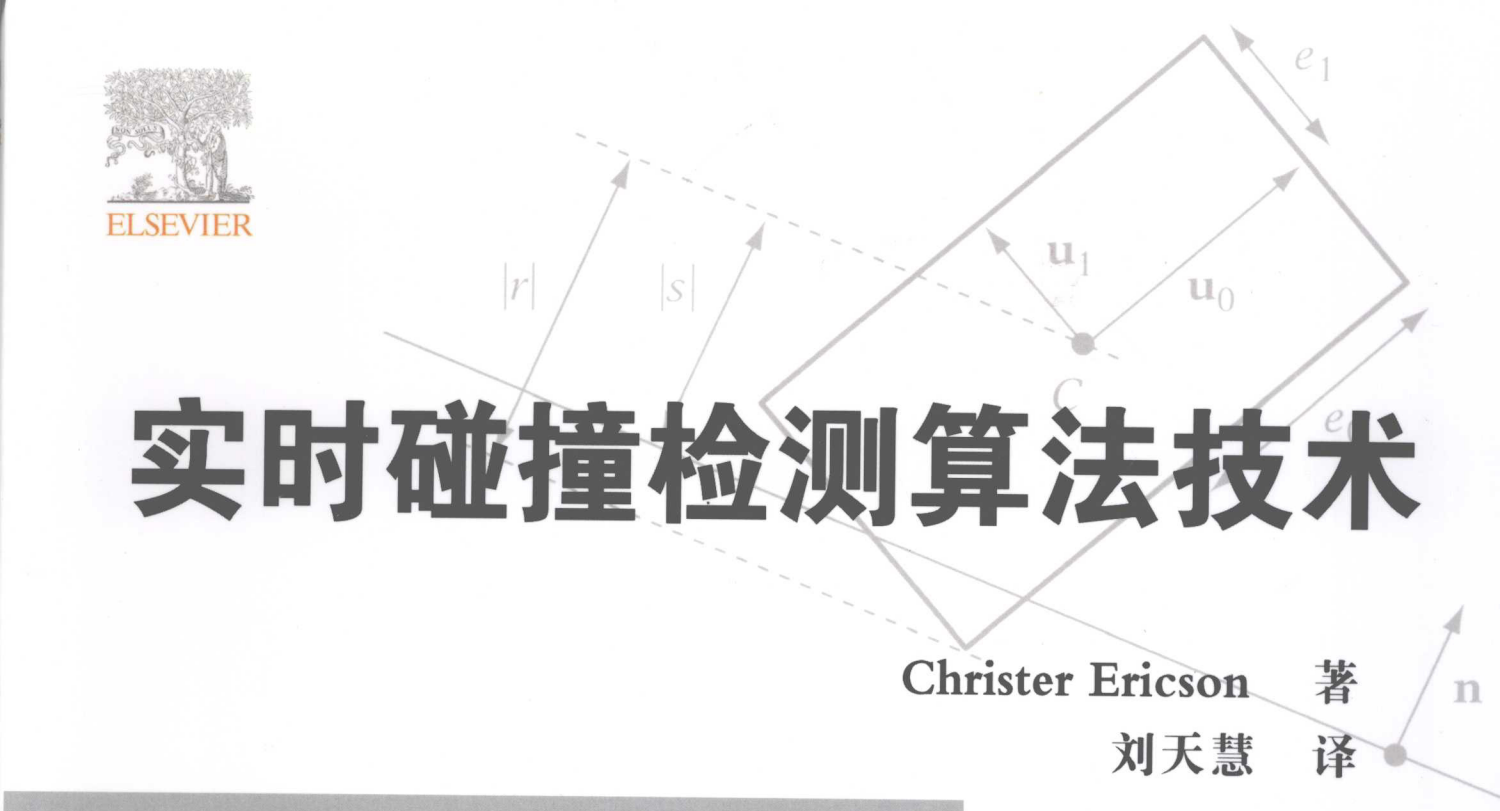

实时碰撞检测算法技术

Christer Ericson 著
刘天慧 译

Real-Time Collision Detection

清华大学出版社

实时碰撞检测算法技术

Christer Ericson 著

刘天慧 译

清华大学出版社

北 京

内 容 简 介

本书详细阐述了与碰撞检测问题相关的高效解决方案及相应的数据结构和算法，主要包括：碰撞检测系统中的设计问题、数学和几何学入门、包围体、基本图元测试、层次包围体技术、空间划分、BSP 树层次结构、凸体算法、基于 GPU 的碰撞检测、数值健壮性、几何健壮性以及优化操作。另外，本书还提供了相应的算法、代码以及伪代码，以帮助读者进一步理解计算方案的实现过程。

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

Real-Time Collision Detection, 1e
Christer Ericson
ISBN: 9789812724120

ISBN: 978-1-55860-732-3

图书在版编目（CIP）数据

实时碰撞检测算法技术/（美）埃里克森（Ericson，C.）著；刘天慧译. —北京：清华大学出版社，2010.6
书名原文：Real-Time Collision Detection

ISBN 978-7-302-22411-2

I. ①实… II. ①埃… ②刘… III. ①计算机仿真 IV. ①TP391.9

中国版本图书馆 CIP 数据核字（2010）第 065078 号

责任编辑：熊　健　郭　伟
封面设计：刘　超
版式设计：牛瑞瑞
责任校对：王　云
责任印制：李红英
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：北京鑫丰华彩印有限公司
装 订 者：三河市兴旺装订有限公司
经　　销：全国新华书店
开　　本：203×260　**印　张**：26.5　**字　数**：707 千字
版　　次：2010 年 6 月第 1 版　　**印　　次**：2010 年 6 月第1 次印刷
印　　数：1～4000
定　　价：52.00 元

产品编号：033422-01

译 者 序

目前，3D 游戏已经成为计算机游戏领域的主流，但不可否认的是，其复杂度也随之上升。相应地，在处理这类较大的数据集时，与碰撞检测相关的数据结构和算法也变得日趋复杂。无论是游戏，还是其他类型的模拟仿真应用程序，碰撞检测始终是程序开发的核心之一。有时，甚至可以用碰撞检测作为衡量游戏引擎是否完善的标准。因此，碰撞检测是让很多开发者棘手的一个难点，不仅算法复杂，而且容易出错。

针对这一问题，本书详细阐述了与碰撞检测问题相关的高效解决方案及其相应的数据结构和算法。碰撞检测问题涵盖了丰富的内容，主要包括：包围体、基本图元测试、层次包围体技术、空间划分、BSP 树层次结构、凸体算法、基于 GPU 的碰撞检测、数值健壮性、几何健壮性以及优化操作。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。

本书深入碰撞检测这一核心问题的内部，内容涉及数学、物理、数据结构、算法、计算机图形学、计算机体系结构以及优化操作等知识。这里是我们具体应用所学知识的地方，不再有 API 调用，一切都是真实的——这的确令人激动！正如本书作者所言：作为一名读者，此刻，你值得拥有！

在本书的翻译过程中，除刘天慧外，刘鹏、张博、刘璋、刘祎、周建娟、李莉、马琳琳、祁宏亮、付钰、李素梅等也参与了部分翻译和校对工作。

限于译者的水平，译文中难免有错误和不妥之处，恳请广大读者批评指正。

译 者

前 言

Preface

1978 年，孩提时代的我就和朋友编写开发了第一款属于我们自己的计算机游戏——我仍然清楚地记得，那是一款采用 BASIC 语言编写的猜谜类游戏，玩家将被问及与非洲动物相关的一些问题。恰巧，“太空入侵者”游戏也于同年发布。诚然，与“太空入侵者”相比，我们的作品多少有些黯然失色，但大家仍沉湎于其中，乐此不疲。不久，我们即在自己的家用计算机上开始模仿“太空入侵者”以及当年流行的街机游戏，并设计自己的原创游戏。时光荏苒，岁月蹉跎，昔日对游戏设计的爱好引领我成为一名业界中人，而游戏领域业已发展壮大为一项价值达数十亿美元的产业，并一直在图形硬件以及 CPU 的发展过程中扮演着催化剂的角色。

言归正传，动作类游戏设计的挑战之一便是如何正确地处理碰撞检测问题，即判断两个对象间以及对象与场景间的相交问题。鉴于游戏最初源于 2D 场景，因而碰撞检测主要集中于如何在屏幕空间内高效地计算对象间的相交状态。时至今日，虽然硬件的计算速度已提高了不止 1000 倍，但碰撞检测仍被视为一类核心问题，这一点确实令人深思。目前，3D 游戏已经成为主流，但其复杂度也随之上升——动辄千万个多边形的计算量已屡见不鲜。相应地，在处理这一类较大的数据集时，与碰撞检测相关的数据结构和算法也变得日趋复杂，皆因这一类计算具有显著的实时特征。这里需要指出的是，游戏设计只是复杂实时碰撞检测应用的一个分支，在诸如 CAD/CAM 系统以及 3D 建模设计中也面临着同样的问题。

针对游戏设计以及其他实时应用程序，本书详细阐述了与碰撞检测问题相关的高效解决方案，包括相应的数据结构和算法。阅读本书需要读者具备良好的数学知识，相关内容也会在书中进行深入的探讨。同时，为了帮助读者进一步理解计算方案的实现过程，本书还提供了相应的算法、代码以及伪代码。

综上所述，碰撞检测问题涵盖了丰富的内容，甚至书中的各章均可独立成书。鉴于此，本书内容力争做到有的放矢，以便为读者今后的学习打下坚实的基础。

关于作者

本书作者 Christer Ericson 毕业于瑞典 Umeå 大学并获得计算机科学硕士学位。随后，在该校任教多年并于 1996 年移居美国。1999 年，他曾在索尼电脑娱乐公司美国分公司担任高级主程序员、高级软件配置管理工程师和首席软件工程师。此前，他还曾在 Neversoft 娱乐公司担任主程序员。2002 年以来，他一直在“Full Sail 游戏设计”和项目开发部担任顾问委员会委员。Christer Ericson 涉猎广泛，目前他将主要精力集中在程序优化管理方面，并在游戏开发者大会上就此问题进行了演讲。

致谢

下列读者对本书内容提出了相关的指正及修改建议，在此表示衷心的感谢，他们是：Ian Ashdown,Gino van den Bergen,David Eberly,George Innis,Neil Kirby,Eric Larsen,Thomas Larsson, Amit Patel, Jamie Siglar,Steven Woodcock 以及一位低调的匿名读者。

另外，我的感谢名单还包括：Ville Miettinen，感谢他慷慨地提供了源代码，这使得与包围体构造相关的章节看起来更加出色；感谢 Matt Pharr，他的建议令我受益匪浅并极大地丰富了第 13 章中的内容；当然，还要感谢 Tim Moss（因为他，我可以按时下班回家并致力于本书的创作）和 Bob Soper（感谢他对初稿提出的建议并就本书的构思与我进行愉快的交流）。

在此，还要感谢编辑 Tim Cox，他对本书的创作赋予了极大的耐心；感谢助理编辑 Richard Camp 和 Stacie Pierce，以及为本书出版付出艰辛劳动的 Morgan Kaufmann 出版社的全体工作人员。

最后，请允许我向 Kim,Ellinor,Tekla 和 Maja 献上我最真挚的谢意，感谢他们 4 年来对我的支持，并在本书的创作过程中陪伴我度过无数个不眠之夜。

作为一名读者，此刻，你值得拥有！

Christer Ericson

目 录

Contents

第1章 概 述

本书将探讨碰撞检测这一主题，即处理如下看似简单的问题：检测两个（或多个）物体是否相交。特别地，碰撞检测将确定两个物体是否、何时以及在何处形成碰撞。“是否碰撞”将产生一个布尔值，并计算两个物体是否相交。“何时碰撞”将额外测定碰撞发生的时刻。“何处碰撞”将解释物体如何形成碰撞。一般来讲，以上述给定顺序分析这3类问题是一个比较复杂的过程。

“何时碰撞”和“何处碰撞”（以及布尔碰撞结果值）常被解释为碰撞检测。术语“相交检测”和“冲突检测”有时也与“碰撞检测”具有相同的含义。

碰撞检测计算往往是各种相关应用程序的基本原理，包括计算机游戏、物理仿真（如计算机动画）、机器人技术、虚拟样机技术以及工程仿真。

在计算机游戏中，碰撞检测将保证真实世界的正确虚拟化（例如，禁止人物穿越墙壁，或者防止人物坠落至地板之下）。同时，碰撞检测还提供了某种视线查询，通知敌人是否发现玩家并发动相应的攻击。

在计算机动画（例如布料的实现）中，碰撞检测常用于约束布料的物理模拟，确保布料以更加真实的方式运动。同时，还可以防止人物运动时产生“脱离”现象。在机器人模拟程序中，碰撞检测还可以用于路径规划，从而使机器人能够避开障碍物。在虚拟样机技术中，物理模型进行批量生产之前，碰撞检测常用于修正计算。另外，碰撞检测还可用于撞击测试和其他类型的工程仿真。

某些应用程序，例如路径规划和动画渲染，并不需要其碰撞检测系统的实时特性；而有些应用程序，特别是计算机游戏，则十分强调碰撞检测系统的实时性能。一些基于计算机或游戏机平台的动作类游戏所涉及的模拟计算数据量，则要求以每秒30帧～60帧的帧速率进行计算。在这种实时状态以及游戏和物理引擎中碰撞检测占优的状态下，碰撞检测系统将占用游戏一帧中的大量时间消耗。因此，在计算机游戏中，拙劣的碰撞系统设计将导致其瓶颈的产生。

本书内容并非只是泛泛阐述碰撞检测这一概念，同时，还将强调实时应用程序中与碰撞检测相关的数据结构和算法的高效实现方法。碰撞检测常被用于游戏领域并加以演示，然而，其他一些非游戏类应用程序也存在类似（或更加苛刻）的性能需求，包括：触觉系统（力反馈）、粒子系统的模拟、医疗外科模拟系统，以及其他一些仿真系统。本书所阐述的相关方案同样适用于这一类应用程序。

这里所讨论的解决方案不仅仅适用于碰撞检测系统。例如，在第6章和第8章中所讨论的内容，同样适用于光线跟踪计算和光线投射计算（如计算场景光照）以及地理信息系统。一些计算机图形学领域中的问题，也可以利用碰撞检测加以解决。例如，视见体的剔除问题就可以采用第6章和第7章所描述的方法实现。

1.1 内容概览

本节将描述本书中各个章节的内容提要。

1.1.1 第 2 章：碰撞检测系统中的设计问题

本章阐述构造碰撞检测系统时应考虑的各种问题，以及影响系统设计的各种因素。包括：物体如何显示、物体的数量、物体的移动方式，以及用户所触发的碰撞查询类型。本章还将介绍贯穿本书的一些相关术语。

1.1.2 第 3 章：数学和几何学入门

任何一种复杂的碰撞检测系统都需要大量的、基于几何定位的数学知识，用于解决碰撞计算的 if、when 和 where 等问题。本章介绍了后续章节中将要用到的数学和几何学概念。

1.1.3 第 4 章：包围体

为了加速碰撞查询的过程，一些简单的几何体对象（如球体和包围盒）常用于表达具有复杂特征的物体。只有当这些“简单的”包围体（该包围体足够大，用于包围具有复杂几何特征的物体）发生了碰撞，复杂几何体的测试计算才得以执行。本章将介绍一些常用的包围体类型、作用于包围体之上的相交测试，以及如何确定一个具有复杂几何特征的物体的包围体。

1.1.4 第 5 章：基本图元测试

在前述章节中介绍相交测试之后，本章将详细阐述不同类型物体之间的相交状态及其空间距离的计算，包括：直线、光线、线段、平面、三角形、多边形、球体、包围盒、圆柱体和多面体。同时，本章还将分别针对静态和动态物体加以测试。

1.1.5 第 6 章：层次包围体技术

对于包含众多物体对象的对象集而言，可以通过构建相应的包围体层次结构来获取高效的查询性能。在碰撞检测过程中，这种层次结构提供了碰撞对象，抑或单一对象中未碰撞部分的快速确认方案，并允许查询限定在少量对象或者对象的某一部分中。本章将讨论包围体的层次结构特征、构造方案，以及作用于其上的查询方法。同时，本章还将探讨实现这一层次结构的高效方案。

1.1.6 第 7 章：空间划分

当碰撞测试计算涉及大量的物体对象时，多个物体需要划分为子集并用于快速测试（避免测试时出现 n^2 级别的运算）。在第 6 章中讨论的包围体层次结构给出了实现该划分的方法。

1.1.7　第 8 章：BSP 树层次结构

一种比较通用的、用于表达碰撞检测系统数据结构的方法是二分空间划分树，即 BSP 树。BSP 树可以从空间对象中单独划分出子空间，也可以用于从对象所处空间中划分其边界线，因此能够高效地实现相应物体对象的体空间表示方法。本章将讨论如何构造健壮的 BSP 树系统，以及如何在该系统中执行相关的测试。

1.1.8　第 9 章：凸体算法

本章将考查凸体对象碰撞测试时的其他高级解决方案，并探讨凸体的特征、层次表达方式、V-Clip 最近特征算法、线性和二次方程程序设计的数学优化方法、高效的 GJK 算法，以及 Chung-Wang 分离向量算法。

1.1.9　第 10 章：基于 GPU 的碰撞检测

与个人计算机的核心之一 CPU 相比，当前民用级别的图形显示卡已经整合了更强大的计算能力，这种改进激发了图形显示卡的外围计算能力。本章将考查如何利用图形硬件来执行碰撞检测测试。

1.1.10　第 11 章：数值健壮性

碰撞检测系统中很小的误差都将导致灾难的发生，例如，物体未曾与几何场景发生正确的碰撞，从而脱离场景。本章将讨论浮点算术的健壮性问题，以及解决这些问题的相关建议。

1.1.11　第 12 章：几何健壮性

第 11 章考查了计算系统的健壮性问题，本章则分析如何将任意的多边形集合转换为具有良好格式的几何元素，并将其正确地提供给碰撞检测系统，包括：顶点的焊接、缝隙的去除、共面合并，以及将物体分解为凸面（三角形）。

1.1.12　第 13 章：优化操作

针对特殊的硬件平台以及相关的代码调试，本章将考查如何利用前述高效的数据结构和算法，并在此基础上获取更优的性能。可以通过优化代码并利用存储结构（缓存），以及代码和数据的并行计算方式，从而获取巨大的性能提升。

1.2 关于本书的代码

本书并非是一本纯理论书籍，其相关的设计思想均辅以对应的代码示例。不同于其他书籍所采用的伪代码算法描述，本书中的绝大多数代码均采用 C++语言加以实现。首先，代码提供了理解（以及实现）算法问题所必需的细节；其次，读者可以运行相关代码并查看运行过程中变量值的状态以加深对该问题的理解，这对于不熟悉算法数学思想的读者尤为重要。本书仅有少量代码采用了伪代码实现，皆因其整体实现方案缺乏相应的可行性。

虽然本书的代码示例采用了 C++语言，但需要强调的是，本书的重点并非是 C++语言。这里，C++语言仅提供了相关概念的一种具体的可行性描述，其他计算机语言均可完成这一任务。本书选择 C++作为描述语言包括下列因素：该语言应用范围较广并可对几何实体的底层计算提供简洁的抽象操作。例如，可使用类和（重载后的）中缀操作符对顶点和向量实施有效的计算。考虑到代码的受众范围（如仅通晓 C 语言或 Java 语言的程序员），本书将会弱化 C++语言中的某些特性操作，例如模板和 STL（标准模板库）。因而，C++纯粹主义者请保持淡定。

同样，本书也并非是一本介绍软件工程的专业性书籍。为了更好地引入相关基础知识点，书中代码尽可能地保持短小精悍。另外，为了避免读者陷入 C++语法的繁文缛节中，书中代码也有相应的调整。例如，尽量保持简约风格的类定义（或不使用类）；将局部变量替换为全局变量；避免将指针声明为 const 类型（或 restrict 类型）；采取固定尺寸的数组而非动态数组。最后，代码还将对变量名的长度加以限制，以使其符合排版要求。

需注意的是，本书中的示例代码并非最终版本。例如，代码中一般不包含“除 0 错”测试，以使读者不至于过早地陷入细节问题中，从而干扰对整体算法的理解。同样，某些代码测试需要添加相应的容值以满足算法的健壮性要求。同时，第 11 章还将对健壮性问题进行详细讨论，并就实现健壮的代码这一问题彰显其中的不同之处。

另外，输入变量通常采取值传递，输出变量则一般通过引用返回，从而进一步区分二者间的差别。在某些情况下，按引用传递出入变量则更加高效，这一部分内容将作为练习留给读者完成。

在本书中，注释将采用手写体表示，代码则使用黑体标注。另外，函数名、类名、结构名以及用户自定义复合类型均以大写字母开头，变量则以小写字母开头。同时，变量的命名尽量与文中所提及的内容一致。但在某些情况下，上述规则也并非一成不变。例如，文中所描述的顶点一般采用大写字母，代码中则使用小写字母。

本书中相关示例代码已上传至 www.tup.com.cn，请读者自行下载。

第 2 章　碰撞检测系统中的设计问题

高效的碰撞检测系统设计过程有一些类似于拼图游戏：在见识一幅完整图画的庐山真面目之前，需要将每一个零散部件拼接起来。类似地，本书的主要工作即是将碰撞检测系统中的不同解决方案所形成的各个组件加以整合。通过本书的学习，读者将逐步地了解碰撞检测系统的全貌。本章将给出在选择诸多解决方案并考虑其相互关系时所涉及的相关问题。同时，本章还将介绍一些相关的术语，其定义和详细的解释将在后续章节中加以说明，其更深层次的内容解析亦将在后续章节中进行阐述。

2.1　碰撞算法的设计因素

在设计碰撞检测系统时，多种因素将会影响设计者的选择，包括以下几类：

（1）应用程序中对象的表达方式。场景以及物体的几何表达方式将直接影响到相关算法的选择，表达方式的约束条件越少，则碰撞检测的解决方案也就越通用，同时，也能够满足既定的运行性能。

（2）查询类型的多样化。通常，查询及其相应的结果越详细，需要的计算量就越大。另外，也许还需要额外的数据结构对某一类特定的查询加以支持。最后，某些对象的几何表达方式还可能不存在相应的查询类型与之对应。

（3）环境模拟参数。模拟环境自身亦包含了某些参数，并对碰撞检测系统产生直接的影响。例如，对象的数量、尺寸和位置、是否移动以及如何移动、物体间是否允许相互穿透、物体是刚体还是可变形物体。

（4）系统的性能。实时碰撞检测系统对时间要求苛刻，并受到物体尺寸的约束。由于时间和空间的可交换性，为了满足既定的性能需求，某些特性通常应均衡加以考虑。

（5）系统的健壮性。应用程序的物理仿真级别是存在差异的。例如，考查一个堆砌的砖垛，与一个球场上跳动的篮球相比，其碰撞检测系统则要复杂得多。篮球弹射初期或以较大角度弹射时，其状态可以忽略不计。但是，在计算砖块之间的接触点时，即使是很小的误差，都将导致砖块逐渐地彼此嵌入或脱离。

（6）系统实现与应用的简易性。多数项目设计都存在其自身的开发周期，若系统不能按时完成并交付使用，那么项目的时间规划将是一句空谈。因此，在考查采用何种方案时，“系统实现的简洁性”这一决策将扮演重要的角色。

上述内容所涉及的问题将在后续章节中详细讨论。

2.2　应用程序中对象的表达方式

在选择合适的碰撞检测算法时，要考虑到场景和物体的几何表达方式，这一点是十分重要的。本

小节将简要地讨论各种物体对象的几何表达方式，并探讨如何使用简化后的几何体，以及如何使特定的方法应用于通用的解决方案之中。

2.2.1 对象的表达方式

当前多数计算机图形硬件均使用三角形作为基本的渲染图元。因此，多边形这一表达方式对于场景和场景中的物体，及其相应的碰撞几何体，将是一个自然的选择。而最为通用的对象表达方式则是“多边形汤”：一组无序的多边形构成的集合，并且不包含确定多边形相互关系的连接信息。由于没有过多的表达限制，对于美术师和关卡设计者来讲，多边形汤往往是一种较具吸引力的对象表达方式。多边形汤的相关算法虽然可以适用于任意多边形集合，但是与那些具备额外信息的多边形集合相比较而言，其执行效率低下并且算法的健壮性也较差。例如，多边形汤并没有考虑到物体对象的“内部”信息，因此，这将较难确定两个对象是否错误地重叠在一起。而所谓额外的数据往往包含一条边上的两个顶点的信息、两个面的连接信息、物体对象是否为一个封闭实体，以及物体对象是否为凸体抑或凹体。

多边形通过公共边可以连接成一个大型的多边形平面，称作多边形网格。如图 2.1 所示，当构造集合模型时，采用多边形网格集合是最常用的方法。

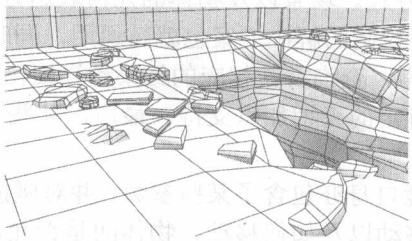

图 2.1　采用多边形网格集合构建的集合模型

多边形物体按照顶点、边和面加以定义。依据该方法构造的物体对象常被称为“具备显式的表达方式”。而“隐式表达方式”的物体对象一般指球体、圆锥体、椭圆、圆环以及其他非显式定义的、但可以间接地通过数学表达式表示的几何图元。隐式物体对象常描述为 3D 空间至实数的函数映射，$f:\mathbb{R}^3 \to \mathbb{R}$。其中，$f(x,y,z)<0$ 代表物体的内部点集；$f(x,y,z)=0$ 代表物体边界上的点集；$f(x,y,z)>0$ 代表物体的外部点集，如图 2.2 所示。通过隐式函数定义的物体对象边界常称作隐式表面。隐式物体对象常用于对场景中的物体进行粗略的快速剔除；在光线跟踪计算中，也可用于对直线和光线执行快速相交测试。隐式测试的一些具体实例将在第 5 章中加以分析。

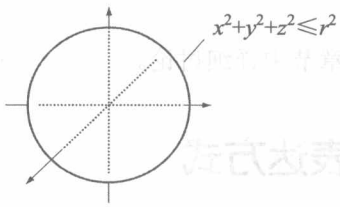

图 2.2　一个显式定义的球体（包括球体边界和内部区域）