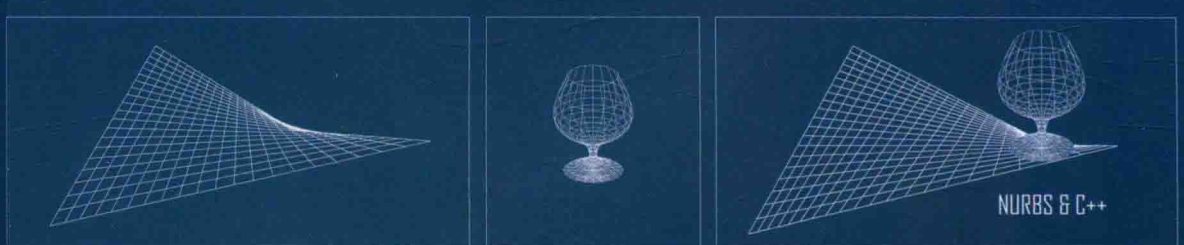
普通高等教育“十三五”规划教材

计算几何算法与实现

（Visual C++版）

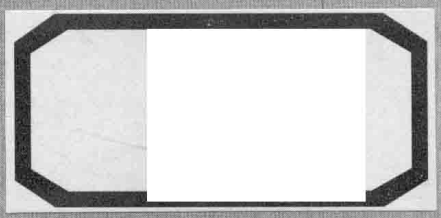
◆ 孔令德　等编著



中国工信出版集团
電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

018
64

普通高等教育“十三五”规划教材



计算几何算法与实现

（Visual C++版）

孔令德　等编著

阜阳师范学院图书馆
藏书

電子工業出版社
Publishing House of Electronics Industry
北京 · BEIJING

内 容 简 介

本书系统介绍 Bezier 曲线曲面、B 样条曲线曲面和 NURBS 曲线曲面的理论与算法。第 1 章介绍曲线曲面的基本概念及表示形式；第 2 章介绍二维图形和三维图形的程序设计方法，示范直线绘图函数的使用方法，重点讲解制作网格模型动画的双缓冲技术；第 3 章讲解三次样条曲线、三次参数样条曲线、Hermite 样条曲线和 Cardinal 曲线的原理与算法；第 4 章介绍三次 Bezier 曲线的定义算法、de Casteljau 递推算法，重点讲解基于双三次 Bezier 曲面片制作 Utah 茶壶的算法，并在课程设计部分给出完整的代码；第 5 章介绍 B 样条的 de Boor-Cox 递推定义算法、二次和三次均匀 B 样条算法、非均匀 B 样条曲线计算节点矢量的 Hartley-Judd 算法；第 6 章在曲线部分介绍 NURBS 精确表示圆弧的方法，在曲面部分重点讲解 NURBS 构建三维曲面如球、圆环、酒杯的原理和算法。为了改变计算几何以数学公式推导为主的单调学习方法，增强曲线曲面的可视化效果，本书提供所有与原理配套的 Visual C++源程序。这些源程序用模块化方法编写，注释简单易懂。为了降低程序的理解难度，旋转曲面投影以最简单的正交投影为主。对于计算机专业教师，可以深入理解原理与代码的对应关系；对于非计算机专业教师，可以直接运行程序。

本书不追求数学上的严密性与完整性，而注重于根据曲线曲面的数学公式的编程实现。本书的所有插图全部使用程序绘制。从数学角度的理解转换为图形方面的观察，可有效提高读者的学习兴趣，实现将数学公式借助于编程技术表示为图形效果的设计初衷。本书附录部分给出了 6 个实验项目及 2 个课程设计项目，并给出了犹他茶壶和花瓶的 Visual C++源代码。

本书可作为高等院校计算机科学与技术、数字媒体技术、信息与计算科学、机械设计等专业本科生、硕士生、博士生的教材与参考书，也可供从事游戏开发、计算机建模、计算机图形学等领域的科学工作者参考使用。

未经许可，不得以任何方式复制或抄袭本书之部分或全部内容。
版权所有，侵权必究。

图书在版编目 (CIP) 数据

计算几何算法与实现：Visual C++版 / 孔令德等编著. —北京：电子工业出版社，2017.8
ISBN 978-7-121-31569-5

I. ①计… II. ①孔… III. ①计算几何－高等学校－教材②计算机算法－高等学校－教材
IV. ①O18 ②TP301.6

中国版本图书馆 CIP 数据核字（2017）第 121344 号

策划编辑：谭海平
责任编辑：谭海平　　特约编辑：王崧
印　　刷：北京京科印刷有限公司
装　　订：北京京科印刷有限公司
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　　邮编：100036
开　　本：787×1 092　1/16　印张：19　字数：486.4 千字
版　　次：2017 年 8 月第 1 版
印　　次：2017 年 8 月第 1 次印刷
定　　价：49.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888，88258888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

本书咨询联系方式：（010）88254552，tan02@phei.com.cn。

前　言

数学的研究对象是“数”与“形”。几何学是研究“形”的一门数学学科。最早的几何学是欧氏几何学。阿基米德创立的欧氏几何学，用刚体运动研究几何不变量，这是最原始的几何学。笛卡儿发现坐标系后，形成了解析几何，通过坐标系将几何问题转化为代数问题。随着微积分的诞生，出现了微分几何。20 世纪 40 年代发明计算机后，为计算几何的出现提供了物质基础，很多几何问题都可使用计算机来解决。计算几何是以计算机为核心的、在信息环境下新产生的一门几何学。计算几何是对物体的形状信息进行计算机表示、分析与综合。计算几何的理论只有借助于计算机的编程实现，才能被更好地理解和应用。

计算几何的研究方面，国外有皮格尔（Piegl）与蒂勒（Tiller）合著的经典教材 *The NURBS Book*，书中重点介绍了关于 NURBS 的理论和算法。作为 NURBS 的主要研究者，皮格尔与蒂勒提出“要想从事 CAD，必须了解 NURBS”。NURBS 是计算几何的集大成者，已成为形状的表示、设计和数据交换的工业标准。国内苏步青与刘鼎元先生于 1981 年合著且影响深远的《计算几何》，开启了我国计算几何研究的先河。北京航空航天大学的施法中先生出版的《计算机辅助几何设计与非均匀有理 B 样条》已成为计算机几何领域的翘楚，内容涵盖了国内外近年来的最新研究进展及施法中先生的创新。

本书的研究内容涉及 Bezier、B 样条、NURBS 曲线曲面。其中，“数”是指 Bezier、B 样条、NURBS 等曲线曲面理论中数学公式的严格推理，“形”是指借助于计算机的强大计算能力，将曲线曲面的数学公式表达为图形。分形几何的创始人曼德尔布罗特（Mandelbrot）曾说过，“看到数学公式，我首先想到的是图形，图形的问题解决了，数学的问题也就解决了”。为了降低理解曲线曲面理论的难度，本书编写的出发点是基于计算几何的基本理论，使用 Visual C++编程生成曲线曲面图形。这些图形包括二维曲线图形和三维曲面图形。三维曲面图形主要采用最简单的平行投影讲解，这也有助于降低理解编程的难度。

本书不追求数学上的严密性与完整性，而注重于使用现有的曲线曲面的理论研究结论，编程绘制二维曲线及三维曲面。对于三维曲面，作者借助于三维图形学技术，基于双缓冲技术建立物体的线框模型，支持键盘方向键旋转图形，可从各个角度对曲面进行观察。作者在大学中主要讲授计算机图形学课程，出版有“十二五”本科普通高等教育国家级规划教材——计算机图形学系列教材（见 www.klingde.com）。计算机图形学研究的内容是模型与渲染问题。作者在讲授计算机图形学模型时，发现仅使用立方体、球体、圆环等模型太过乏味，进而研究犹他茶壶，并使用 Visual C++编程绘制了包含 306 个控制点的 32 片双三次曲面。调用 Visual C++编写的不同程序模块，学生可以选择绘制壶盖、壶嘴、壶体、壶柄、壶嘴等。这与直接调用 OpenGL 或 3DS 中的茶壶模型，只能绘制茶壶的整体效果是不同的。在此基础上，指导博创研究所的学生基于 B 样条方法与 NURBS 方法，使用 MFC 框架建立不同旋转体的曲面模型。博创研究所开发了一套“平转立”系统，只要给出一段 NURBS 二维曲线，就可直接生成其三

维模型，并可使用键盘方向键旋转进行交互观察。作为教学相长的范例，学生的快速进步，促使我产生了编写一本适合于应用型本科院校使用的计算几何教材的想法，而该教材应该是图文并茂的。为了编好本书，作者开发了近 100 多个源程序，并详细编写了程序注释。博创研究所的石长盛、李振林、陶作柠等学生全程参与了程序的设计与开发，付出了辛苦的努力，在此一并致谢。本书不包含任何手绘插图，每幅插图全部由作者提供的 Visual C++算法生成。插图准确、图例丰富、立体感强，易于理解，更便于教与学。

本书的特点是理论与代码一一对应。代码方面模块性强、注释规范、容易理解。这些代码不仅可以从实践上印证理论的正确性，也可以从图形可视化角度展示理论应用的效果。本书附录部分给出了 6 个实验项目及 2 个课程设计项目，并给出了犹他茶壶和花瓶的源代码。读者可以通过编程提高对理论的认识。读者学习本书的先行课程包括高等数学、C++程序设计等。

本书理论部分由孔令德编写，上机实验及课程设计部分由康凤娥编写。全书由孔令德提出编写提纲并统稿、校对。康凤娥上机调试了全部程序并绘制了插图。

作者提供计算几何算法与实现 QQ 群，群号为 229275085。群内的“共享文件”包括与教学配套的电子课件及相关的案例源程序等教学资源，欢迎读者下载使用。请扫描微信二维码或通过 QQ（307622194）与作者联系。就书中理论和编程方面的问题，作者将为读者提供在线答疑。

本书适合于计算机科学与技术专业、数字媒体技术专业、数学与应用数学专业以及机械设计与制造专业的本科生与研究生使用。建议本科生的学习重点为 Bezier 曲线曲面，研究生的学习重点为 B 样条与 NURBS 曲线曲面。

王阳明说过：“知者行之始，行者知之成”，让我们打开计算机，开始学习基于动画的计算几何学，以物体的几何形状来加深对枯燥数学公式的理解吧。

孔令德说计算几何课

作　者

于太原万达龙湖广场

2017 年 7 月 7 日

试读结束：需要全本请在线购买：　www.ertongbook.com

目　录

第1章 绪 论

计算几何（Computational Geometry）由 Minsky 和 Papert 于 1969 年提出，但直到 Forrest 才给出正式的定义：对几何外形信息的计算机表示、分析与综合。几何外形信息是指几何体的曲线曲面信息。按照这些信息建立数学模型，使用计算机进行计算，求得曲线曲面上的更多信息，这就是所谓的计算机表示，然后对计算机内的几何模型进行分析和综合，这些内容形成了计算几何。计算几何借助于程序设计算法，可以方便地研究“数”与“形”的问题。1981 年苏步青教授在上海科学技术出版社出版的专著《计算几何》，开启了我国研究计算几何的先河。该书是对 1980 年前国际上关于计算几何的理论、方法和应用的总结。1982 年，该书荣获“全国优秀科技图书奖”。需要说明的是，国际上常把计算几何称为计算机辅助几何设计（Computer Aided Geometric Design，CAGD）。施法中编著的《计算机辅助几何设计与非均匀有理 B 样条》是国内这一领域的经典教材；国际上著名的教材是 Piegl 与 Tiller 合著的权威之作 *The NURBS Book*。

1.1 计算几何的研究内容

世界是由形状各异的物体组成的，物体的几何形状大致可分为两类：一类由初等解析曲面如平面、圆柱面、圆锥面、球面和圆环面等组成，使用初等函数可完全且清楚地表达这些形状；另一类由没有数学表达式的自由曲面（Free form Surface）组成，如汽车车身、飞机机翼和轮船船体等的曲面，如图 1.1 所示。不能用初等函数完全且清楚地表达全部形状，需要构造新的函数来表示。

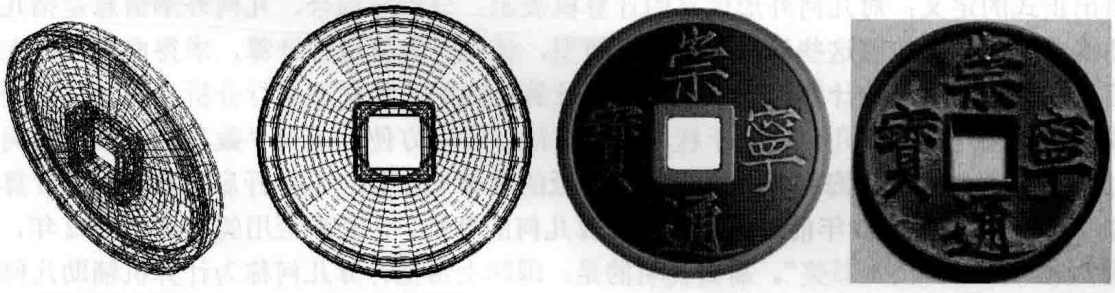
图 1.1 汽车曲面

在制造飞机与船舶的工厂里，传统上采用模线样板法来表示自由曲线（Free form Curve）的形状。模线员通过在型值点处固定均匀且带弹性的细木条、有机玻璃条或金属条来绘制所需要的曲线即模线，以此制成样板作为生产与检验的依据。曲面上没有模线的部分取成光滑过渡，这是采用模拟量传递信息的设计制造方法。人们一直在寻求用数学方法来唯一地定义自由曲线曲面的

试读结束：需要全本请在线购买： www.ertongbook.com

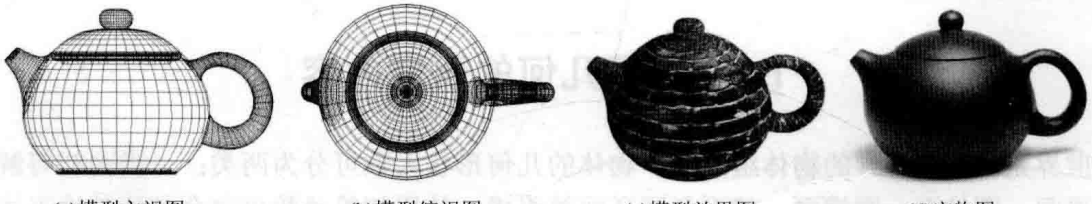
形状，将形状信息从模拟量改为数字量，这称为形状数学。形状信息的表示中，大量计算工作手工无法完成，只能依赖于计算机。随着计算机应用的普及，采用数学方法定义的自由曲线曲面才得到实际应用，这就促成了计算几何学科的产生与发展。目前，人们普遍认为计算几何是综合了微分几何、代数几何、数值计算、逼近论、拓扑学等的一门边缘性学科。计算几何与微分几何不同，它的生命力在于应用。国际上重要的计算几何会议，总有 IBM、波音和通用汽车等大企业的代表参加。计算几何的研究内容是计算机环境中曲线曲面的表示与逼近。依据定义形状的几何信息可建立相应的曲线曲面方程，即几何模型，并借助于程序设计方法绘制这些曲线曲面。

实体造型（Solid Modeling）技术是采用描述几何模型的曲线曲面理论，由计算机生成具有真实感的三维图形的技术。虽然近年来实体造型技术已经取得很大的进展并进入实际应用，但实体造型理论的发展落仍后于曲线曲面，不像曲线曲面理论那样成熟。学习计算几何的目的之一是为了建模。本书将从计算机学科的角度讲解曲线曲面造型理论，不注重理论的证明，而侧重于编程实现。书中讲述的每个理论，都采用案例化的方法给出使用二维、三维建模技术绘制的图形。图 1.2 为采用 B 样条曲面制作的“崇寧通寶”铜钱的实体模型，图 1.3 为使用 Bezier 曲面制作的“西施壶”紫砂壶的实体模型。



(a)模型任意向视图　(b)模型主视图　(c)模型效果图　(d)实物图

图 1.2　铜钱实体造型



(a)模型主视图　(b)模型俯视图　(c)模型效果图　(d)实物图

图 1.3　西施壶实体造型

1.2　曲线曲面描述数学的发展

自从 1946 年世界上首台电子计算机问世以来，计算机应用的一个重要里程碑是 1962 年美国麻省理工学院发明了世界上第一台图形显示器。自此以后，计算机就可以通过图形显示器直接输入、输出图形，并可以在显示屏上通过光标的移动而修改图形。而在这之前，工程师是通过一厚叠纸上密密麻麻的数字来表达工程图形的。

1962 年被认为是美国和欧洲计算机辅助设计（Computer Aided Design，CAD）开始发展

的一年。最早的应用领域是汽车、飞机和造船工业。在这三个行业，由于产品外形特别复杂，要求特别严格，因此成为 CAD 首先应用的领域。与此同时，发展出了一门新兴学科 CAGD，专门研究“几何图形信息（曲面和三维实体）的计算机表示、分析和综合”。1974 年在美国举行的 CAGD 第一次国际会议，标志着计算几何学科的形成。

1963 年，美国波音公司的 Ferguson 首先提出将曲线曲面表示为参数的矢函数方法。他最早引入了参数三次曲线，构造了组合曲线和由四角点的位置矢量及两个方向的切矢定义的 Ferguson 双三次曲面片。在这之前，曲线的描述一直采用显式的标量函数 $y=y(x)$ 或隐函数方程 $F(x,y)=0$ 的形式，曲面相应采用 $z=z(x,y)$ 或 $F(x,y,z)=0$ 的形式。Ferguson 所采用的曲线曲面参数形式从此成为自由曲线曲面数学描述的标准形式。

1964 年美国麻省理工学院（MIT）的 Coons 发表了一种具有一般性的曲面描述方法，即给定围成封闭曲线的四条边界就可定义一片曲面。1967 年，Coons 进一步推广了他的这一思想。在 CAGD 实践中应用广泛的只是其特殊形式：Coons 双三次曲面片。它与 Ferguson 双三次曲面片的区别，仅在于将角点扭矢由零矢量改为非零矢量。但这两种方法都存在形状控制与连接问题。1967 年，由 Schoenberg 提出的样条函数提供了解决连接问题的一种技术。用于形状描述的样条方法是样条函数的参数形式，即参数样条曲线曲面。样条方法用于解决插值问题，在构造整体上达到某种参数连续性的插值曲线曲面时很方便。但该方法不存在局部调整性，而且样条曲线和曲面的形式难以预测。

1962 年，法国雷诺汽车公司的 Bezier 提出了一种由控制多边形定义曲线的方法，并成功地运用于 UNISURF 汽车造型系统中。设计人员只需要移动控制顶点，就可方便地修改曲线的形状，而且曲线形状的变化完全在可控范围之内。Bezier 方法不仅简单易用，而且出色地解决了整体形状控制问题，把曲线曲面的设计向前推进了一大步，为曲面造型的进一步发展奠定了坚实的基础。但 Bezier 方法仍然存在连接问题和局部修改问题。

1972 年，de Boor 给出了关于 B 样条的一套标准算法。1974 年，美国通用汽车公司的 Gordon 和 Riesenfeld 又将 B 样条理论应用于形状描述，提出了 B 样条曲线曲面。B 样条方法几乎继承了 Bezier 方法的所有优点，克服了 Bezier 方法存在的缺点，既较为成功地解决了局部控制问题，又轻而易举地在参数连续性基础上解决了连接问题，从而使得自由曲线曲面形状的描述问题得到了较好解决。但随着技术的进步，B 样条方法显示出明显不足：不能精确地表示圆锥截线及初等解析曲面，这就造成了产品几何定义的不唯一，使曲线曲面没有统一的数学描述形式，容易造成生产管理混乱。

人们希望找到一种统一的数学方法。1975 年，美国 Syracuse 大学的 Versprille 在其博士论文中首次提出了有理 B 样条方法。此后，主要由于 Piegl 和 Tiller 等人的功绩，到 20 世纪 80 年代后期，非均匀有理 B 样条（Non-Uniform Rational B-Spline，NURBS）方法成为现代曲面造型中广泛流行的数学方法。国际标准化组织（International Standardization Organization，ISO）于 1991 年颁布了关于工业产品数据交换（Standard for Exchange of Product model data，STEP）的国际标准，将 NURBS 方法作为定义工业产品几何形状的唯一数学方法，从而使得 NURBS 方法成为曲面造型技术发展中最重要的基础。目前，国外几乎所有的商品化软件都使用了 NURBS 方法。

20 世纪 70 年代中期，CAD 在我国的造船和飞机制造业中开始发展。在苏步青的组织和推动下，我国于 1982 年举办了第一届计算几何和 CAD 学术会议。直到现在，我国每年都会举办计算机辅助设计与图形学（CG/CAD）大会。

1.3 矢量代数基础

1.3.1 矢量表示

矢量是既有大小又有方向的量，也称为向量，如位移、速度、加速度等。与之对应，只有大小而没有方向的量称为标量。

矢量依其始端是否位于原点分为绝对矢量与相对矢量。

曲线上的点用绝对矢量末端即矢端表示。绝对矢量也称为该点的位置矢量（Position Vector）。相对矢量是表示点与点之间相对位置关系的矢量。

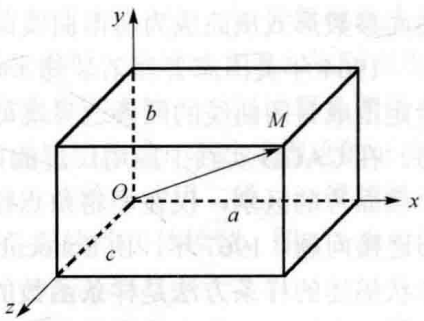


图 1.4 位置矢量

在图 1.4 中，$\overrightarrow{OM}$ 表示位置矢量，$\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{k}$ 分别表示三个坐标轴的单位矢量。矢量 $\overrightarrow{OM}$ 表示为

$$\overrightarrow{OM} = a\boldsymbol{i} + b\boldsymbol{j} + c\boldsymbol{k} \quad \text{或} \quad (a, b, c)$$

矢量的模：矢量的大小称为矢量的模，即 $\left|\overrightarrow{OM}\right| = \sqrt{a^2 + b^2 + c^2}$。

单位矢量：大小为 1 的矢量。

1.3.2 矢量的运算

矢量的加减法：设有两个矢量 $\boldsymbol{a} = (a_x, a_y, a_z)$ 和 $\boldsymbol{b} = (b_x, b_y, b_z)$，则

$$\boldsymbol{a} + \boldsymbol{b} = (a_x + b_x, a_y + b_y, a_z + b_z)$$

矢量与数的乘法：设有实数 λ，则 $\lambda\boldsymbol{a} = (\lambda a_x, \lambda a_y, \lambda a_z)$。

矢量的点积：两个矢量的点积是一个数，它等于两个矢量的模与夹角余弦的乘积。

设有两个矢量 $\boldsymbol{a} = (a_x, a_y, a_z)$ 和 $\boldsymbol{b} = (b_x, b_y, b_z)$，$\theta$ 为 $\boldsymbol{a}$ 和 $\boldsymbol{b}$ 间的夹角，则

$$\boldsymbol{a} \cdot \boldsymbol{b} = |\boldsymbol{a}| \cdot |\boldsymbol{b}| \cos\theta = a_x b_x + a_y b_y + a_z b_z \tag{1.1}$$

矢量的叉积：两个矢量的叉积是一个矢量，使用右手螺旋法可确定矢量的方向，如图 1.5 所示。矢量的模为两个矢量的模与夹角正弦的乘积。

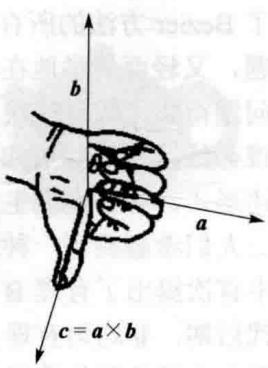


图 1.5 矢量的叉积

设有两个矢量 $\boldsymbol{a} = (a_x, a_y, a_z)$ 和 $\boldsymbol{b} = (b_x, b_y, b_z)$，$\theta$ 为 $\boldsymbol{a}$ 和 $\boldsymbol{b}$ 间的夹角，则 $|\boldsymbol{c}| = |\boldsymbol{a}| \cdot |\boldsymbol{b}| \sin\theta$。

矢量的叉积还可表示为

$$\begin{aligned}
\boldsymbol{a} \times \boldsymbol{b} &= (a_x\boldsymbol{i} + a_y\boldsymbol{j} + a_z\boldsymbol{k}) \times (b_x\boldsymbol{i} + b_y\boldsymbol{j} + b_z\boldsymbol{k}) \\
&= a_x\boldsymbol{i} \times (b_x\boldsymbol{i} + b_y\boldsymbol{j} + b_z\boldsymbol{k}) + a_y\boldsymbol{j} \times (b_x\boldsymbol{i} + b_y\boldsymbol{j} + b_z\boldsymbol{k}) + a_z\boldsymbol{k} \times (b_x\boldsymbol{i} + b_y\boldsymbol{j} + b_z\boldsymbol{k}) \\
&= a_x b_x(\boldsymbol{i} \times \boldsymbol{i}) + a_x b_y(\boldsymbol{i} \times \boldsymbol{j}) + a_x b_z(\boldsymbol{i} \times \boldsymbol{k}) + a_y b_x(\boldsymbol{j} \times \boldsymbol{i}) + a_y b_y(\boldsymbol{j} \times \boldsymbol{j}) + a_y b_z(\boldsymbol{j} \times \boldsymbol{k}) + \\
&\quad a_z b_x(\boldsymbol{k} \times \boldsymbol{i}) + a_z b_y(\boldsymbol{k} \times \boldsymbol{j}) + a_z b_z(\boldsymbol{k} \times \boldsymbol{k})
\end{aligned}$$

由于 $\boldsymbol{i}\times\boldsymbol{i}=\boldsymbol{j}\times\boldsymbol{j}=\boldsymbol{k}\times\boldsymbol{k}=0$，$\boldsymbol{i}\times\boldsymbol{j}=\boldsymbol{k}$，$\boldsymbol{j}\times\boldsymbol{k}=\boldsymbol{i}$，$\boldsymbol{k}\times\boldsymbol{i}=\boldsymbol{j}$，$\boldsymbol{j}\times\boldsymbol{i}=-\boldsymbol{k}$，$\boldsymbol{k}\times\boldsymbol{j}=-\boldsymbol{i}$，$\boldsymbol{i}\times\boldsymbol{k}=-\boldsymbol{j}$，所以

$$\boldsymbol{a}\times\boldsymbol{b}=(a_yb_z-a_zb_y,a_zb_x-a_xb_z,a_xb_y-a_yb_x) \tag{1.2}$$

1.3.3 设计矢量类

定义三维矢量类 CVector3，计算三维矢量的点积与叉积。其中 CP3 是三维点类，包括 x, y, z 三个数据成员。

```
#include "P3.h"
class CVector3
{
public:
    CVector3(void);
    virtual ~CVector3(void);
    CVector3(double x, double y, double z);                    //分量构造矢量
    CVector3(const CP3 &p);                                    //构造绝对矢量
    CVector3(const CP3 &p0, const CP3 &p1);                    //构造相对矢量
    double Magnitude(void);                                    //矢量的模
    CVector3 Normalize(void);                                  //单位矢量
    friend CVector3 operator+ (const CVector3 &v0, const CVector3 &v1);
                                                               //运算符重载
    friend CVector3 operator- (const CVector3 &v0, const CVector3 &v1);
    friend CVector3 operator* (const CVector3 &v, double scalar);
    friend CVector3 operator* (double scalar, const CVector3 &v);
    friend double   DotProduct (const CVector3 &v0, const CVector3 &v1);
                                                               //矢量点积
    friend CVector3 CrossProduct (const CVector3 &v0, const CVector3 &v1);
                                                               //矢量叉积
public:
    double x, y, z;
};
CVector3::CVector3(void)
{
    x = 0.0;
    y = 0.0;
    z = 1.0;
}
CVector3::~CVector3(void)
{
}
CVector3::CVector3(double x, double y, double z)
{
    this->x = x;
    this->y = y;
    this->z = z;
}
CVector3::CVector3(const CP3 &p)
```

```
{
    x = p.x;
    y = p.y;
    z = p.z;
}
CVector3::CVector3(const CP3 &p0, const CP3 &p1)
{
    x = p1.x – p0.x;
    y = p1.y – p0.y;
    z = p1.z – p0.z;
}
double CVector3::Magnitude(void)
{
    return sqrt(x * x + y * y + z * z);
}
CVector3 CVector3::Normalize(void)
{
    CVector3 vector;
    double Mag = sqrt(x * x + y * y + z * z);
    if(fabs(Mag) < 1e-6)
        Mag = 1.0;
    vector.x = x / Mag;
    vector.y = y / Mag;
    vector.z = z / Mag;
    return vector;
}
CVector3 operator +(const CVector3 &v0, const CVector3 &v1)
{
    CVector3 vector;
    vector.x = v0.x + v1.x;
    vector.y = v0.y + v1.y;
    vector.z = v0.z + v1.z;
    return vector;
}
CVector3 operator –(const CVector3 &v0, const CVector3 &v1)
{
    CVector3 vector;
    vector.x = v0.x – v1.x;
    vector.y = v0.y – v1.y;
    vector.z = v0.z – v1.z;
    return vector;
}
CVector3 operator *(const CVector3 &v, double scalar)
                                                    //矢量与常量的积
{
    CVector3 vector;
    vector.x = v.x * scalar;
```

```
        vector.y = v.y * scalar;
        vector.z = v.z * scalar;
        return vector;
    }
    CVector3 operator *(double scalar, const CVector3 &v)
                                                    //常量与矢量的积
    {
        CVector3 vector;
        vector.x = v.x * scalar;
        vector.y = v.y * scalar;
        vector.z = v.z * scalar;
        return vector;
    }
    double DotProduct(const CVector3 &v0, const CVector3 &v1)
                                                    //矢量的点积
    {
        return(v0.x * v1.x + v0.y * v1.y + v0.z * v1.z);
    }
    CVector3 CrossProduct(const CVector3 &v0, const CVector3 &v1)
                                                    //矢量的叉积
    {
        CVector3 vector;
        vector.x = v0.y * v1.z – v0.z * v1.y;
        vector.y = v0.z * v1.x – v0.x * v1.z;
        vector.z = v0.x * v1.y – v0.y * v1.x;
        return vector;
    }
```

例 1.1 图 1.6 所示正四棱锥的 4 个顶点为 P_0，P_1，P_2 和 P_3，试写出平面 $P_0P_1P_2$ 的方程。

平面的一般方程为

$$ax+by+cz+d=0 \tag{1.3}$$

平面的法矢量为 $\boldsymbol{n}=(a,b,c)$。

取 P_0 点为参考点，则有边矢量

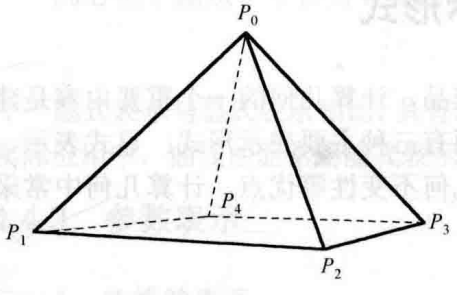


图 1.6 正四棱锥

$$\overrightarrow{P_1P_0}=(x_1-x_0,y_1-y_0,z_1-z_0)$$
$$\overrightarrow{P_2P_0}=(x_2-x_0,y_2-y_0,z_2-z_0)$$

平面的法矢量 $\boldsymbol{n}$ 为

$$\begin{aligned}\boldsymbol{n}&=\overrightarrow{P_1P_0}\times\overrightarrow{P_2P_0}\\&=((y_1-y_0)(z_2-z_0)-(z_1-z_0)(y_2-y_0),\\&\quad(z_1-z_0)(x_2-x_0)-(x_1-x_0)(z_2-z_0),\\&\quad(a_1-a_0)(y_2-y_0)-(y_1-y_0)(x_2-x_0))\end{aligned}$$

所以

$$a=(y_1-y_0)(z_2-z_0)-(z_1-z_0)(y_2-y_0)$$
$$b=(z_1-z_0)(x_2-x_0)-(x_1-x_0)(z_2-z_0)$$

计算曲面方程系数

试读结束：需要全本请在线购买： www.ertongbook.com