

高等学校教材

计算机图形学

徐长青 许志闻 郭晓新 杨赢涛 冯月萍 编著
王钺旋 主审

机械工业出版社

本书系统地介绍了计算机图形学的有关原理、算法及实现，其主要内容包括：计算机图形硬件、基本图形的生成、图形几何变换、曲线和曲面造型、几何造型、基本的图形运算、典型的隐藏线与隐藏面的消除算法、生成真实感图形的相关技术。本书可作为大专院校本科生、研究生学习计算机图形学的教材，也可作为从事 CAD 和计算机图形学技术研究的广大科技人员的参考书。

图书在版编目 (CIP) 数据

计算机图形学/徐长青等编著. —北京：机械工业出版社，2004.2

高等学校教材

ISBN 7-111-13920-8

I. 计... II. 徐... III. 计算机图形学 IV. TP391.41

中国版本图书馆 CIP 数据核字 (2004) 第 006707 号

机械工业出版社 (北京市百万庄大街 22 号 邮政编码 100037)

策划编辑：王保家

责任编辑：苏颖杰 王保家 贡克勤 版式设计：冉晓华 责任校对：程俊巧

封面设计：陈沛 责任印制：闫焱

北京京丰印刷厂印刷·新华书店北京发行所发行

2004 年 2 月第 1 版·第 1 次印刷

787mm×1092mm $1/16$ ·12.5 印张·304 千字

定价：19.00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换

本社购书热线电话 (010) 68993821、88379646

封面无防伪标均为盗版

前 言

21 世纪是经济全球化、信息社会化、产业知识化、市场开放化高速发展的新时代。伴随着微电子和计算机科学技术日益渗透到经济、生活、学习、工作以及生产加工等社会活动的各个领域，人类正跨步迈进一个全新的知识经济时代。在实际工作环境中，计算机图形的应用范围非常广泛，相关知识的更新、发展亦非常快，及时学习和掌握新的研究成果以及提高实际应用能力，将为以后工作与研究奠定基础。自最初在吉林大学计算机系开设“计算机图形学基础”这门课，至今已有 20 多年，其间曾编撰出版《计算机图形学》一书。经过近几年的教学实践，并同时参考国内外一些专著，我们对原书又进行了必要的改动和补充，形成了目前的新版本。

本书主要介绍计算机图形学的原理、算法及实现。参考和总结计算机图形学近年来的新成果，力图全面、准确地介绍这些内容。本书首先概括计算机图形学学科的研究内容，然后介绍基本图形的生成算法，采取循序渐进的内容安排，由简单到复杂，由二维到三维，最后再介绍真实感图形显示的有关内容。计算机图形学是一门正在发展、形成中的新兴学科，还有许多方法正在研究探索中，本学科的研究可以说是大有可为、前景广阔。

由于编者的水平和能力所限，书中可能存在错误和不足，衷心希望读者给予批评与指正。

徐长青、许志闻、郭晓新、杨赢涛、冯月萍参加了本书的编撰工作，全书由王钺旋教授主审。

作者

2004 年 1 月于吉林大学

目 录

前 言	
第一章 计算机图形学简介	1
第一节 计算机图形学	1
第二节 计算机图形学 的起源	2
第三节 计算机图形学的 应用及发展动向	3
第四节 图形系统的硬件	4
第五节 计算机图形标准	6
习 题	7
第二章 图形基元的显示	8
第一节 直线扫描转换算法	8
第二节 圆的扫描转换算法	13
第三节 区域填充	18
习 题	26
第三章 图形变换	28
第一节 变换的数学基础	29
第二节 二维图形变换	32
第三节 二维视见变换	37
第四节 三维图形变换	39
第五节 投影	44
第六节 裁剪	51
习 题	60
第四章 曲线和曲面	64
第一节 曲线和曲面表示 的基础知识	64
第二节 Hermite 多项式	67
第三节 Coons 曲面	73
第四节 Bezier 曲线和曲面	79
第五节 B 样条曲线和曲面	91
习 题	99
第五章 图形运算	102
第一节 线段的交点计算	102
第二节 多边形的 交线计算	105
第三节 平面中的凸壳算法	108
第四节 包含与重叠	110
第五节 简单多边形的 三角剖分	115
习 题	120
第六章 形体的表示 及其数据结构	122
第一节 二维形体的表示	122
第二节 三维几何模型	128
第三节 分形	137
习 题	145
第七章 消除隐藏线和隐藏面 的算法	148
第一节 线面比较法消除 隐藏线	148
第二节 曲面隐藏线消除的 浮动水平线算法	151
第三节 深度排序算法	153
第四节 画家算法	155
第五节 z-缓冲算法	158
第六节 扫描线算法	159
第七节 区域分割算法	162
第八节 BSP 树算法	165

第九节	八叉树算法	166	第三节	阴影	176
第十节	光线投射算法	167	第四节	纹理	177
习 题		167	第五节	整体光照模型	178
第八章	真实感图形的绘制	169	第六节	光线跟踪	180
第一节	漫反射及具体光源 的照明	170	第七节	加速光线跟踪算法	182
第二节	多边形网的 明暗处理	173	第八节	辐射度方法	184
			第九节	色彩模型	187
			习 题		190
			参考文献		192

第一章 计算机图形学简介

客观世界中的事物是多姿多彩的，而呈现在我们眼前的往往是它们的外观，通过外观人们进一步地认识及研究它们。以图画为表现形式的图形信息在人类的社会生活中起着非常重要的作用。与其它的信息表现形式相比，图形信息具有容易理解、容易记忆、直观等特点。随着现代科学技术的发展，用计算机来处理图形的信息，完成图形的构造、显示与分析很自然地就成为人们研究与探索的领域。

第一节 计算机图形学

随着计算机技术的快速发展，涉及到图形方面的应用也越来越深入。诸如零件的构造与显示、卫星照片的处理及手写文字的识别等。经过多年的研究与发展，逐渐形成了多个与图形相关的分支，计算机图形学（Computer Graphics）、图像处理（Image Processing）和模式识别（Pattern Recognition）就是其中的典型代表。

简单来说，计算机图形学是指用计算机产生对象图形的输出的技术。更确切地说，计算机图形学是研究通过计算机将数据转换为图形，并在专门显示设备上显示的原理、方法和技术的学科。它综合了应用数学、计算机科学等多方面的知识。

图形是对象的一种外在表现形式，它是对象有关信息的具体体现。所谓对象，可以是各种具体的、实在的物体，如家具、机械零件、房屋建筑等，也可以是抽象的、假想的事物，如天气形势、人口分布、经济增长趋势等。能够正确地表达出一个对象性质、结构和行为的描述信息，称为这个对象的模型。计算机图形学中产生图形的方法是建立对象的模型，即对该对象作出所需的正确的信息描述，然后利用计算机对这个模型进行各种必要的处理，从无到有地产生出能正确反映对象的某种性质的图形输出。可以说，计算机产生图形的过程就是将数据（对象的模型表示）转化为图形的过程。

图像处理是指用计算机来改善图像质量的数字技术。可见或不可见的图像经过量化后输入到计算机中（扫描仪扫描输入、数码相机拍照），由计算机按应用的需要对已有的图像进行增强、复原、分割、重建、编码、存储、传输等种种不同的处理，再把加工后的图像进行输出。在太空探索中分析宇宙飞船发回的各种照片，在生物医学工程中发展起来的计算机的X射线断层摄影技术（Computer Tomography，简称CT）是计算机图像处理技术的典型例子。

模式识别是指用计算机对输入图形进行识别的技术。图形信息输入计算机后，先进行特征抽取等预处理，然后用统计判定方法或语法分析方法对图形作出识别，最后按照使用的要求给出图形的分类或描述。各种中西文字符及工程图样的自动阅读装置，就是模式识别技术的应用实例。

与计算机图形处理有关的上述三门学科是独立发展起来的。当前由于光栅扫描显示器的广泛使用及解决复杂的实际问题的需要，它们已经相互渗透，也使人们对这三门学科的相互

关系和共同技术产生了越来越大兴趣。

图 1-1 是计算机图形学、图像处理和模式识别相互关系的示意图。

此外，与计算机图形学关系密切的学科还有计算几何学。计算几何学是研究几何模型和数据处理的数据集合。显然，寻找对复杂形体的描述方法和在计算机中存放的适当的数据结构不是很容易的。通常认为，二维、三维物体及曲线、曲面的描述，几何问题算法的设计和分析，都是计算几何学研究的内容。

交互式计算机图形学是指用计算机交互式地产生图形的技术。交互式绘图允许操作人员以对话方式控制和操纵图形的生成。图形可以边生成、边显示、边修改，直到产生符合使用要求的图形为止。交互式绘图可以使人的逻辑思维能力、分析能力和计算机准确快速的计算能力结合起来，从而发挥更大的威力，使人们运用起来更加方便。其中，交互设备是实现交互技术，完成交互任务的基础。一般来说，交互设备有定位、键盘、选择、取值和拾取。交互式技术是用户使用交互设备把信息输入计算机的不同方式，交互任务是用户输入到计算机的一个单元信息，基本任务有四种：定位、字符串、选择、取数。如用鼠标选择菜单项或定位坐标点作图等。开发软件系统的人员需要了解相应的概念与原理，以便选择适宜的交互设备实现计算机系统和用户的沟通，而用户只需了解交互设备的作用和种类即可。交互技术是完成交互任务的手段，它的实现有赖于交互设备及其支撑环境。

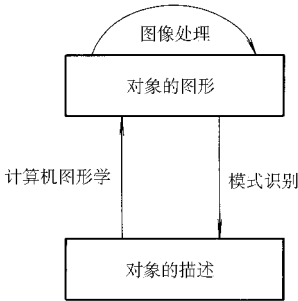


图 1-1 图像处理、模式识别和计算机图形学的相互关系

第二节 计算机图形学的起源

1950 年，第一台和麻省理工学院的 Whirlwind 计算机相连的显示器产生了简单的图形。这台显示器使用了类似电视机使用的阴极射线管（CRT，Cathode Ray Tube）。在这之前，已故的 F. Williams 曾使用 CRT 作为信息存储器件，这项技术后来被用来以存储式 CRT 的形式，组成价格低廉的交互式图形终端。

在 20 世纪 50 年代，由于计算机还不适应交互式使用，交互式计算机图形学几乎没有什么发展。那时的计算机主要是被物理学家及导弹设计工程师用来进行冗长的数字计算。到了 20 世纪 50 年代末期，由于麻省理工学院的 TX-0 和 TX-2 机器的研制成功，交互式计算成为可能，人们对计算机图形显示的兴趣迅速增长。

1962 年，麻省理工学院的 Ivan E. Sutherland 发表了一篇光辉的博士论文，这极大地推进了交互式计算机图形显示成为一个重要的崭新的领域。这篇题为“Sketchpad：人一机图形通信系统”的论文，向许多读者证明了交互式计算机图形显示是一个有生命力的、有前途的和令人振奋的研究领域。到了 20 世纪 60 年代中期，麻省理工学院、通用汽车公司、贝尔电话实验室和洛克希德飞机公司等都在进行大规模的计算机图形显示研究，从此开始了计算机图形学的黄金年代。

如果说 20 世纪 60 年代代表了计算机图形学研究工作令人兴奋的岁月，那么 20 世纪 70 年代就是这些研究工作结出硕果的十年。很多国家应用了交互式图形显示，这些技术广泛地

应用在计算机辅助设计、事物管理、过程控制、教育等诸多领域。20 世纪 80 年代, 计算机图形学不仅在工业、管理、艺术领域发挥巨大作用, 而且已进入家庭。20 世纪 90 年代以来, 科学计算的可视化、虚拟现实环境的应用又向计算机图形学提出了许多更高更新的要求, 使得三维乃至高维计算机图形学在真实性和实时性方面将有飞速发展。

计算机图形学是人们和计算机通信的最通用的和最有力的手段。计算机图形显示对不同年龄的用户都具有强大的吸引力, 图形直观清晰的特性也大大推广了它的应用范围。大规模集成电路技术以及新型廉价硬件的诞生使计算机图形学得到飞速的发展。当前由于各种图形设备性能不断提高, 价格不断下降, 使计算机图形学的应用领域正在不断增加。毋庸置疑, 这一技术将继续发展、普及。

第三节 计算机图形学的应用及发展动向

计算机图形学几乎在一切领域, 诸如工业、商业、政府部门、教育、科研、医学、娱乐和广告方面, 都有着广泛的应用。

在科学技术事业中, 可以使用计算机来绘制表示数值计算或数据处理结果的图形, 如各种函数的图形、统计用的直方图、百分比图等。

在制图学方面, 可以利用计算机来绘制精确的地形图、天气图、海洋图、石油开发图、人口密度图等。

计算机辅助设计 (CAD: Computer Aided Design) 和计算机辅助制造 (CAM: Computer Aided Manufacturing) 是计算机图形学的重要应用领域。CAD 和 CAM 技术已经相当广泛地应用到电子、机械、建筑、汽车、飞机、船舶等方面的设计和制造工作中。

在计算机仿真与动画方面, 可以用计算机制作动画电影来表现真实物体或模拟物体的运动或变化, 例如训练飞行员可以使用一种用计算机控制的可以产生需要的动态效果的飞行模拟器。

在过程控制中, 可用以显示被控制对象有关环节在操作过程中的状态, 使操作人员可以进行必要的调节和处理。

在办公室自动化方面, 图形显示技术有助于数据及其相互关系的有效表达, 因而有利于人们进行正确的决策,

计算机艺术的出现, 使计算机的应用进入到艺术领域。利用计算机已经创造出具有一定水平和独特风格的艺术作品, 例如传统的中国绘画、书法、油画、美术图案以及动画电影和广告等。

随着计算机的推广和普及, 计算机图形学的应用领域还将继续深入和扩大。

计算机图形学的研究内容是十分丰富的。虽然有些部分的研究工作已经进行了多年, 取得了不少成果, 但随着图形显示技术应用领域的扩大和深入, 不断有新的研究课题涌现出来。从计算机图形学发展和应用的现状看, 这门学科的主要研究内容可以概括为以下几个方面:

1) 图形的生成和表示技术。例如线段、圆弧、曲线和曲面的生成算法, 区域填充算法, 基本几何体的截交、相贯及展开算法以及投影、隐藏线和隐藏面消除, 浓淡处理, 灰度与色彩等各种表示技术。

2) 图形的操作与处理方法。例如图形的剪裁、开窗、平移、旋转、放大、缩小等各种操作的方法及软件或硬件的实现技术。

3) 图形输出设备与输出技术的研究。

4) 图形输入设备、交互技术及用户接口技术的研究。

5) 图形信息的数据结构及存储、检索方法。例如图形信息的各种机内表示方法、组织形式、存取技术、图形数据库的管理、图形信息的通信等。

6) 几何模型构造技术。刻画被处理对象几何性质的描述信息就构成它的几何模型。图形生成和操作的基础就是对象的几何模型，所以要研究几何模型的构造方法及性能分析等。

7) 动画技术。研究实现各种高速动画生成的各种软硬件方法、开发工具、动画语言等。

8) 图形软硬件的系列化、模块化和标准化的研究。

9) 科学计算的可视化。科学计算可视化是指应用计算机图形生成技术将科学及工程计算的结果以及测量数据等以图像的形式在计算机屏幕上显示出来，使人们能观察到用常规手段难以观察到的自然现象和规律，实现计算环境和工具进一步现代化。

第四节 图形系统的硬件

一个简单的计算机图形系统的硬件包括五个部分：计算机、显示处理器、图形显示器、输入设备（键盘、图形输入板等）和硬拷贝设备（绘图机、打印机等）

显示处理器（DPU）是专用于显示图形的处理器，是一种具有专门用途的 CPU。

图形显示器是最终产生图形显示效果的部件。多年来，阴极射线管显示器（CRT）由于其分辨率好、可靠性高、速度快、成本低等优点，始终是图形显示器中使用最多的。通过对阴极加负电压产生电子束，然后进行加速、聚焦，再经过偏转系统控制，使被聚焦的一定数量的电子束轰击到荧光屏发出亮光，产生图形显示。

CRT 图形显示器工作方式有两种：随机扫描方式和光栅扫描方式。随机扫描方式的图形显示器通过画出一系列线段来画出图形。图 1-2 是随机扫描图形显示的示意图。CRT 的电子束像一支快速移动的画笔，沿箭头指出的路线点亮各线段，就显示出一个多边形的轮廓。随机扫描方式画线速度快，产生图形分辨率高，但是难以产生具有多种灰度、色调能连续变化的图形。随机扫描又称为向量扫描或轨迹扫描，随机扫描显示器又称为画线显示器。

在光栅扫描方式的图形显示器中，CRT 的电子束被控制按图 1-3 方式在显示屏上扫描。电子束从屏的左上角开始由左向右进行正向扫描，当扫到最右位置时迅速回到最左边位置。电子束同时还进行自上而下的扫描，所以行扫描线略向下倾斜，回扫后电子束到达下一行的左边位置。当电子束如此逐行扫描到达屏面右下角时，即完成整个屏幕的扫描。扫描过程所产生的图像称为一帧，然后电子束立即又回到屏的左上角扫描的起始位置，然后开始重复下一个扫描过程。在电子束从左向右和从上至下的扫描过程中按要显示的图形点亮或熄灭屏上的光点，就在屏幕上画出了要显示的图形。图 1-4 是光栅扫描方式显示多边形轮廓的示意图。当电子束扫描到屏幕上的每一点时，若其能量不同，则光点的明暗程度也不同，因此光栅扫描图形显示器能产生与电视图像类似的、相当逼真的图形显示，已经成为当前最重要的显示方式。光栅扫描又称为电视扫描。

在光栅扫描图形显示器中，屏幕上可以点亮或熄灭的最小单位称为像素。显示屏上像素

的总数称为分辨率，常用每行的像素数与行数的乘积表示。显示分辨率是图形显示器最重要的性能参数，分辨率越大，像素的总数越多，画面的质量也就越好，与此同时必须在单位时间内完成对图形的多帧显示，以使得图形不闪烁，因而分辨率的大小对刷新速度有着直接的影响。另一个重要的性能参数是颜色或亮度等级数目。亮度等级或称灰度等级数目是指单色显示器像素的亮度可以有多少种不同的变化。彩色显示器的颜色数目分成两个指标，一个是显示器可以显示所有不同的颜色的总数，另一个是同一帧画面允许显示的不同颜色数目。例如，好的微机系统已经发展到有 1600×1200 像素和 256 种彩色的显示能力。

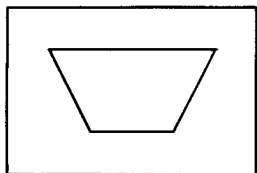


图 1-2 随机扫描

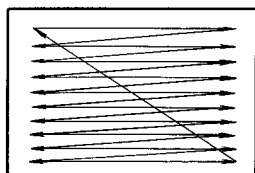


图 1-3 光栅扫描

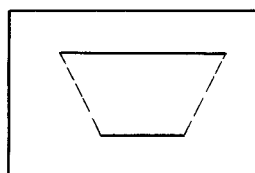


图 1-4 光栅扫描

光栅扫描图形显示的基本的结构如图 1-5 所示。屏幕上每个像素对应的颜色或亮度值要存储在帧存储器中。帧存储器可以理解为一个二维矩阵，矩阵第 i 行 j 列的元素的值，决定了显示屏第 i 行 j 列位置上像素的颜色或亮度。如果像素有 256 中不同颜色，则帧存储器的每一单元至少应为 8bit，对于分辨率为 1024×768 的显示器，帧存储器的大小约为 1M，若为真彩色，则所需的存储单元就更多。为了实现图形的绘制，颜色系统有多种构造形式。RGB（红、绿、蓝）、颜色查询表（Color Lookup Table）、CMY（青、品红、黄）、HSV（色彩、饱和度、亮度）。图形显示时，光栅扫描发生器中的 x 地址和 y 地址计数器不断计数。其数值一方面通过光栅扫描电路使 CRT 电子束产生扫描偏转，另一方面又作为地址值读出帧存储器中的内容以控制显示屏上对应像素的颜色或亮度。电子束在显示屏上进行大约每秒 25 帧的扫描，帧存储器的内容被反复读出，在显示屏上便可以看到稳定的图形显示。

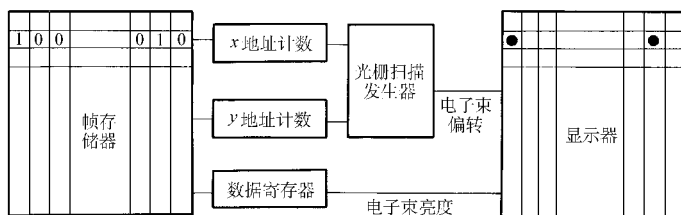


图 1-5 光栅扫描显示结构

图形就它对人们的关系来说，需要的是识别它、生成它、改变它、欣赏它。计算机图形学正是人们想利用计算机来达到这些目的而产生和发展的。今天，人们在利用计算机来生成和操作图形方面已经取得了很大的成功，然而在利用计算机来识别复杂图形方面，在产生可以美化生活的各种艺术品以及计算机动画和电影方面，以及其它许多方面，还有很多要做的工作。计算机图形学正在快速地发展，其前景是无限美好的。

第五节 计算机图形标准

随着计算机应用的日益深入，人们对图形方面的要求也越来越普遍，而且要求也越来越高，例如提供简单、清晰、易用的图形界面，配合相应的交互式输入输出设备，将使用户能更好地工作。要达到这样的目的就需要开发具有高性能的图形系统。由于计算机及输入输出设备种类繁多，且不断推陈出新，这就要求图形系统具有较好的适应性，可应用在多种不同类型计算机上，同时也可使用种类繁多的输入输出设备，也就是要求应用程序与图形设备无关。为了使应用程序在不同系统之间或不同程序之间可以移植，使不同系统之间或不同程序之间相互交换图形数据成为可能，制定图形软件的标准是非常重要的。早在 1974 年，在美国国家标准化局（ANSI）举行的“与机器无关的图形技术”的会议上，就提出了计算机图形标准化和制定有关标准的规则。1977 年，美国计算机协会（ACM）下属的图形标准化委员会提出了“核心图形系统”（Core Graphics System）的规范。与此同时，国际标准化组织（ISO）先后批准了与计算机图形有关的标准有：计算机图形核心系统（GKS）及其语言联编、三维图形核心系统（GKS-3D）及其语言联编、程序员层次交互式图形系统（PHIGS）及其语言联编、计算机图形元文件（CGM）、计算机图形接口（CGI）、基本图形转换规范（IGES）、产品数据转换规范（STEP）等。

计算机图形的标准是指图形系统及其相关应用系统中各界面之间进行数据传送和通信的接口标准，以及供图形应用程序调用的子程序功能及其格式标准，前者为数据及文件格式标准，后者为子程序界面标准。

CGI（Computer Graphics Interface）为用户提供控制图形硬件的一种与设备无关的方法，使得用户能够灵活方便地直接控制图形设备，它是面向图形设备的接口标准。CGI 以一种不依赖于具体设备的方式提供图形信息的描述和通信，其控制功能集支持虚拟设备管理、虚拟设备坐标系和设备坐标系及两坐标系间的映射、画面的裁剪以及系统控制、出错检测和有关上述信息的访问；输出功能集描述图元、属性、对象的构成及有关的控制和查询；此外，提供输入和应答功能集支持逻辑输入设备（定位、拾取、光栅等）的处理以及产生、修改、检索和显示以像素数据形式存储的光栅功能集。

CGM（Computer Graphics Metafile）是一套与设备无关的语义词法定义的图形文件格式。CGM 提供的图形元文件规定了生成、存储、传送图形信息的格式。其作用如图 1-6 所示。

GKS（Graphics Kernel System）提供应用程序和图形输入输出设备之间的功能接口，它是一个独立于具体语言的图形核心系统，在应用中将 GKS 嵌入到相应的语言中。GKS 提供有交互式与非交互式的图形处理功能。其包括控制、图形输入输出、坐标转换（用户、设备、规范化设备坐标系）、图段、GKS 文件接口（元文件）。

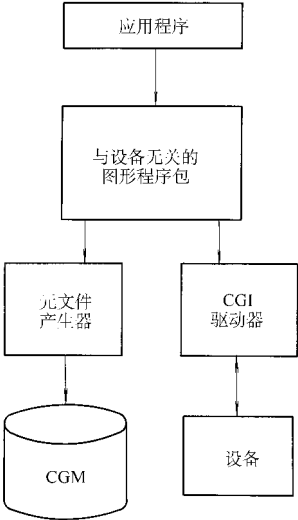


图 1-6 生成 CGM 的方式

PHIGS (Programmer's Hierarchical Interactive Graphics System) 是为三维图形应用而设计的图形软件工具库。其主要特点是在系统中高效地描述应用模型，迅速修改图形模型的数据，并能绘制显示修改后的图形模型，易于对图形作动态的处理；在图形操作上，拥有二维和三维图形操作能力，满足向量和光栅图形设备的特点。

GL (Graphics Library) 是在工作站上广泛应用的一个标准的图形程序库。它具有如下功能：基本图素（线、多边形等）、坐标变换、设置属性和显示方式、输入输出处理、真实图形显示。其提供了丰富的图元；支持 RGB 和颜色索引方式，提供 Gouraud 和 Phong 光照模型；采用 Z 缓冲技术处理消隐；提供的光照模型充分地考虑了各种因素对光强及颜色的影响，大大地提高了显示的图形的真实程度。

本书的目标是为从事计算机图形学方面有关工作提供必要的基本知识，因此重点介绍与图形处理有关的基本原理和方法。

习 题

1. 试比较计算机图形学与图像处理、模式识别的共同点和不同点。
2. 试举出几个计算机图形学的应用实例。

第二章 图形基元的显示

光栅扫描图形显示系统的出现极大地推动了计算机图形学的发展。在光栅扫描显示器中，帧缓冲存储器中每个地址都必须有正确的图形亮度值，图形描述模型中的点、线和面的表示都必须转换成存储图形的像素矩阵表示。将图形描述转换成用像素矩阵表示的过程称为扫描转换。每次图形改变都要进行扫描转换，因此高效快速的扫描转换算法是非常重要的。

直接采用像素矩阵描述图形显然是不方便的。实际图形往往由一些基本图形组合产生。图形基元，或称输出图形元素，是图形系统能产生的最基本图形。图形基元的选择可以不同，但通常都把线段、圆、多边形选为图形基元。本章将讨论线段、圆、多边形的扫描转换算法。

第一节 直线扫描转换算法

一、DDA 直线扫描转换算法

现在我们考查线段的扫描转换算法。

较高级图形系统提出画一条线段的要求通常是给出线段的两个端点，线段扫描转换算法的任务是根据端点位置求出构成该线段的所有像素位置的坐标，如图 2-1 所示。在光栅扫描显示方式中像素坐标是行和列的位置值，只能取整数，是理想线段上点坐标的近似值。显然这使得当光栅扫描图形显示器的显示分辨率较低时，画出的线段会呈现阶梯状。好的线段扫描转换算法画出的线段应尽可能逼近原直线，线的亮度应均匀且与线的方向无关，线段两端截断要精确，要能够尽可能快速地产生整条线段。

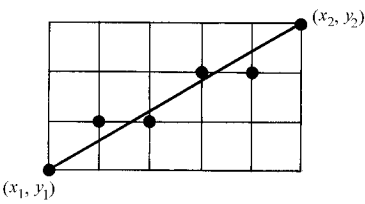


图 2-1 光栅扫描方式显示的直线

设待画线段两端点的坐标值是 (x_1, y_1) 和 (x_2, y_2) ，不妨假定 $x_1 < x_2$ ，要画线段所在直线方程是：

$$y = mx + b \tag{2-1}$$

则有：

$$m = \frac{y_2 - y_1}{x_2 - x_1}, b = \frac{x_2 y_1 - x_1 y_2}{x_2 - x_1} \tag{2-2}$$

于是得到线段扫描转换最直接的算法是：给出 (x_1, y_1) 和 (x_2, y_2) ，利用式 (2-2) 求出 m 和 b 。当 $|m| \leq 1$ 时，对 x 每增 1 取允许的各整数值，用式 (2-1) 进行乘法和加法运算求 y 后再取整。当 $|m| \geq 1$ 时，应先对 y 取各整数值再计算 x 。这个算法计算量大，画线慢，不是好的扫描转换算法。

注意到对直线，其数值微分是

$$m = \frac{\Delta y}{\Delta x} = \frac{y_{i+1} - y_i}{x_{i+1} - x_i} \quad (2-3)$$

由式 (2-3) 得, $y_{i+1} = y_i + m (x_{i+1} - x_i)$, 于是知, 使 x_i 增 1, 即 $x_{i+1} = x_i + 1$ 时, $y_{i+1} = y_i + m$, 为画线精确, 应使相邻的画出点的坐标值相差最大为 1, 这样可以得到画线段的数值微分分析器 (Digital Differential Analyzer, 简称 DDA) 算法如下:

```
void DDALine (int x1, int y1, int x2, int y2)
{
    double dx, dy, e, x, y;
    dx = x2 - x1;
    dy = y2 - y1;
    e = (fabs(dx) > fabs(dy)) ? fabs(dx) : fabs(dy);
    dx /= e;
    dy /= e;
    x = x1;
    y = y1;
    for (int i = 1; i <= e; i++)
    {
        SetPixel ((int) (x + 0.5), (int) (y + 0.5));
        x += dx;
        y += dy;
    }
}
```

算法中 $(int) (x + 0.5)$, $(int) (y + 0.5)$ 是四舍五入后取得的整数, 注意到算法在逐点计算的循环中已经没有了乘除运算, 但计算下一点位置的加法是实数运算。

二、中点画线法

为了讨论方便, 本小节假定直线斜率在 0、1 之间。其它情况可参照下述讨论进行处理。如图 2-2 所示, 若直线在 x 方向上增加一个单位, 则在 y 方向上的增量只能在 0、1 之间。假设 x 坐标为 x_i 的各像素点中, 与直线最近者已确定, 为 (x_i, y_i) 。那么, 下一个与直线最近的像素只能是正右方的 $P_1 (x_i + 1, y_i)$ 或右上方的 $P_2 (x_i + 1, y_i + 1)$ 两者之一。再以 M 表示 P_1 与 P_2 的中点, 即 $M = (x_i + 1, y_i + 0.5)$ 。又设 Q 是理想直线与垂直线 $x = x_i + 1$ 的交点。显然, 若 M 在 Q 的下方, 则 P_2 离直线近, 应取为下一个像素; 否则应取 P_1 。这就是中点画线法的基本原理。

下面来讨论中点画线法的实现。假设直线的起点和终点分别为 (x_0, y_0) 和 (x_1, y_1) 。则直线方程为:

$$F(x, y) = ax + by + c = 0$$

其中, $a = y_0 - y_1$, $b = x_1 - x_0$, $c = x_0 y_1 - x_1 y_0$ 。对于直线上的点, $F(x, y) = 0$; 对于直线上方的点, $F(x, y) > 0$; 而对于直线下方的点, $F(x, y) < 0$ 。因此, 欲判断前述 Q 在 M 的上方还是下方, 只要把 M 代入

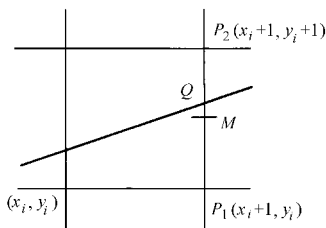


图 2-2 中点画线法每步迭代涉及的像素和中点示意图

$F(x, y)$, 并判断它的符号。构造判别式：

$$d = F(M) = F(x_i + 1, y_i + 0.5) = a(x_i + 1) + b(y_i + 0.5) + c$$

当 $d < 0$ 时, M 在直线下方 (即在 Q 的下方), 故应取右上方的 P_2 作为下一个像素。而当 $d > 0$, 则应取正右方的 P_1 。当 $d = 0$ 时, 二者一样合适, 可以随便取一个。我们约定取正右方的 P_1 。

对每一个像素计算判别式 d , 根据它的符号确定下一像素。至此已经可以写出完整的算法。但是注意到 d 是 x_i 和 y_i 的线性函数, 可采用增量计算, 提高运算效率。在 $d \geq 0$ 的情况下, 取正右方像素 P_1 , 欲判断再下一个像素应取那个, 应计算：

$$d_1 = F(x_i + 2, y_i + 0.5) = a(x_i + 2) + b(y_i + 0.5) + c = d + a$$

故 d 的增量为 a 。而若 $d < 0$, 则取右上方像素 P_2 。要判断再下一个像素, 则要计算：

$$d_2 = F(x_i + 2, y_i + 1.5) = a(x_i + 2) + b(y_i + 1.5) + c = d + a + b$$

故在第二种情况, d 的增量为 $a + b$ 。

再看 d 的初始值。显然, 第一个像素应取左端点 (x_0, y_0) , 相应的判别式值为：

$$\begin{aligned} d_0 &= F(x_0 + 1, y_0 + 0.5) = (x_0 + 1) + b(y_0 + 0.5) + c \\ &= ax_0 + by_0 + c + a + 0.5b \\ &= F(x_0, y_0) + a + 0.5b \end{aligned}$$

但由于 (x_0, y_0) 在直线上, 故 $F(x_0, y_0) = 0$ 。因此, d 的初始值为 $d_0 = a + 0.5b$ 。

由于我们使用的只是 d 的符号, 而且 d 的增量都是整数, 只是其初始值包含小数。因此, 我们可以用 $2d$ 代替, 来摆脱小数, 写出仅包含整数运算的算法：

```
void MidpointLine (int x0, int y0, int x1, int y1)
```

```
{
    int a, b, delta1, delta2, d, x, y ;
    a = y0 - y1 ;
    b = x1 - x0 ;
    d = 2 * a + b ;
    delta1 = 2 * a ;
    delta2 = 2 * (a + b) ;
    x = x0 ;
    y = y0 ;
    SetPixel (x, y) ;
    while (x < x1)
    {
        if (d < 0)
        {
            x ++ ;
            y ++ ;
            d += delta1 ;
        }
        else
        {
            x ++ ;
            d += delta2 ;
        }
    }
}
```

```
else
{
    x++ ;
    d+=delta1 ;
}
SetPixel (x,y) ;
}/* while */
}/* MidpointLine */
```

上述就是中点画线算法程序。如果进一步把算法中 $2 * a$ 改为 $a + a$ 等等，那么，这个算法不仅只包含整数变量，而且不包含乘除法，适合硬件实现。

作为一个例子，我们来看中点画线法如何光栅化一条连接两点 $(0,0)$ 和 $(5,2)$ 的直线段。由于 $(x_0,y_0) = (0,0)$ ，且 $(x_1,y_1) = (5,2)$ ，直线斜率 $k = 2/5$ ，满足 $0 \leq k \leq 1$ ，所以，可以应用上述算法。

第一个像素应取线段左端点 $(0,0)$ 。判别式 d 的初始值为： $d_0 = 2 * a + b = 1$ (因为 $a = y_0 - y_1 = -2, b = x_1 - x_0 = 5$)。 d 往正右方向的增量 $\Delta_1 = 2a = -4$ ； d 往右上方的增量 $\Delta_2 = 2(a + b) = 6$ 。

由于 $d_0 > 0$ ，所以迭代循环的第一步取初始点的正右方像素 $(1, 0)$ ， x 递增 1，并将 d 更新为： $d = d_0 + \Delta_1 = 1 - 4 = -3$ 。

因为 $x = 1 < x_1$ ，所以进入第二步迭代运算。这时由于 $d < 0$ ，故取右上方像素 $(2,1)$ ， $x、y$ 同时递增 1，并将 d 更新为： $d = -3 + \Delta_2 = 3$ ，这样继续分析下去知 $x、y、d$ 的初值和循环迭代过程中每一步的值依序如下：

x	y	d	x	y	d
0	0	1	3	1	-1
1	0	-3	4	2	5
2	1	3	5	2	1

三、Bresenham 画线算法

Bresenham 提出了一个更好的算法，算法中可以只用整数运算，自然也不必有四舍五入。为了说明简便，假定直线斜率 m 在 0 到 1 之间，且 $x_2 > x_1$ 。设在第 i 步已经确定第 i 个像素点是 $(\dot{x}_i, \dot{y}_i)$ ，它是直线上点 (x_i, y_i) 的最接近位置， $\dot{x}_i = x_i$ ，现在看第 $i + 1$ 步如何确定第 $i + 1$ 个像素点的位置。

如图 2-3 所示，第 $i + 1$ 个像素点是 $(\dot{x}_i + 1, \dot{y}_i)$ 和 $(\dot{x}_i + 1, \dot{y}_i + 1)$ 两者中的一个。在 $x = \dot{x}_i + 1$ 处直线上的 y 值是 $y = m(\dot{x} + 1) + b$ ，该点到点 $(\dot{x}_i + 1, \dot{y}_i)$ 和点 $(\dot{x}_i + 1, \dot{y}_i + 1)$ 的距离分别是 d_1 和 d_2 ：

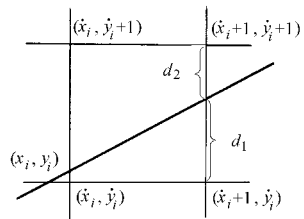


图 2-3 第 $i + 1$ 个像素点的位置计算

$$d_1 = y - \dot{y}_i = m(\dot{x} + 1) + b - \dot{y}_i \quad (2-4)$$

$$d_2 = (\dot{y}_i + 1) - y = (\dot{y} + 1) - m(\dot{x} + 1) - b$$

这两个距离的差是：

$$d_1 - d_2 = 2m(\dot{x}_i + 1) - 2\dot{y}_i + 2b - 1 \quad (2-5)$$

若此差值为正，则 $d_1 > d_2$ ，下一个像素点应取 $(\dot{x}_i + 1, \dot{y}_i + 1)$ ；若此差值为负，则 $d_1 < d_2$ ，下一个像素点应取 $(\dot{x}_i + 1, \dot{y}_i)$ ；若此差值为零，则 $d_1 = d_2$ ，下一个像素点可取上述两个像素点中的任意一个。但 $d_1 - d_2$ 的符号计算不够简便，为此引入一个新的判别量 p_i ：

$$p_i = \Delta x (d_1 - d_2) = 2\Delta y \cdot \dot{x}_i - 2\Delta x \cdot \dot{y}_i + c \quad (2-6)$$

式中， $\Delta x = x_2 - x_1$ ； $\Delta y = y_2 - y_1$ ； $c = 2 \cdot \Delta y + \Delta x (2b - 1)$ 。因为这里 $\Delta x > 0$ ，故 p_i 与 $d_1 - d_2$ 同号，可以作判别量。下面看怎样从 p_i 计算 p_{i+1} 。将式 (2-6) 中 i 换成 $i + 1$ ，有：

$$p_{i+1} = 2\Delta y \cdot \dot{x}_{i+1} - 2\Delta x \cdot \dot{y}_{i+1} + c \quad (2-7)$$

将式 (2-7) 两边减式 (2-6) 两边，并利用 $\dot{x}_{i+1} = \dot{x}_i + 1$ ，可得：

$$p_{i+1} - p_i = 2\Delta y - 2\Delta x (\dot{y}_{i+1} - \dot{y}_i) \quad (2-8)$$

注意到 $p_i \geq 0$ 时，应取 $\dot{y}_{i+1} = \dot{y}_i + 1$ ，此时式 (2-8) 给出 $p_{i+1} = p_i + 2(\Delta y - \Delta x)$ ；而 $p_i < 0$ 时，应取 $\dot{y}_{i+1} = \dot{y}_i$ ，此时式 (2-8) 给出 $p_{i+1} = p_i + 2\Delta y$ 。为形成算法，还需要说明初始判别量 p_1 如何确定。因为线段上第一个像素点可取起点，即 $\dot{x}_1 = x_1$ ， $\dot{y}_1 = y_1 = \frac{\Delta y}{\Delta x} x_1 + b$ ，将其代入式 (2-6)，令 $i = 1$ ，可计算求出：

$$p_1 = 2\Delta y - \Delta x$$

现在将上述说明整理为如下算法：

```
void BresenhamLine (int x1,int y1,int x2,int y2)
```

```
{
```

```
    int x,y,dx,dy,p;
```

```
    x=x1;
```

```
    y=y1;
```

```
    dx=x2-x1;
```

```
    dy=y2-y1;
```

```
    p=2 * dy - dx;
```

```
    for (x<=x2; x++)
```

```
    {
```

```
        SetPixel (x,y);
```

```
        if (p>=0)
```

```
        {
```

```
            y++;
```

```
            p+=2 * (dy-dx);
```

```
        }
```

```
    }
```

此为试读，需要完整PDF请访问：www.ertongbook.com