Auto CAD

计算机绘图基础教程

朱泽平 主编

机械工业出版社

Auto CAD 计算机绘图基础教程

主　编　朱泽平
副主编　朱清萍　李东生　赵　军
参　编　徐元龙　汪　颖　何汉阳
　　辛洪兵　王　谦　张　磊
主　审　敖泌云　陈　集

机械工业出版社

本书主要介绍了计算机绘图的常用设备，Turbo C 语言绘图函数，二、三维图形变换，窗口与裁剪技术和 Auto CAD R13 的使用方法。

在 Auto CAD R13 的使用方法中系统介绍了 Auto CAD 绘图、编辑、图层、尺寸标注与剖面线、辅助工具等各种命令，以及三维绘图、二次开发技术、Auto LISP 语言等。最后简单介绍了新近面世的 Auto CAD R14 的新增功能。

本书内容系统、全面，可供计算机绘图技术人员使用。

图书在版编目（CIP）数据

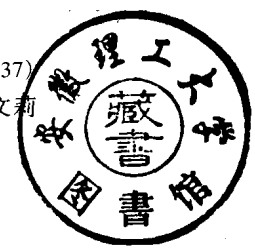
Auto CAD 计算机绘图基础教程/朱泽平主编．－北京：机械工业出版社，1998.9

ISBN 7-111-06657-X

Ⅰ.A… Ⅱ.朱… Ⅲ.自动绘图－教材 Ⅳ.TP391.72

中国版本图书馆 CIP 数据核字（98）第 22188 号

出 版 人：马九荣（北京市百万庄大街 22 号 邮政编码 100037）
责任编辑：曲彩云 版式设计：李 悦 责任校对：罗文新
封面设计：郭景云 责任印制：侯新民
大兴兴达印刷厂印刷·新华书店北京发行所发行
1998 年 10 月第 1 版·1998 年 10 月第 1 次印刷
787mm×1092mm 1/16·17.25 印张·435 千字
0 001－5 000 册
定价：29.00 元



前　言

随着计算机与图形设备的日益普及与发展，计算机辅助设计（CAD）及绘图（CG）技术已成为众多计算机应用技术领域的最具活力并发展最快的分支之一。“计算机绘图”作为计算机辅助设计的核心内容，其作用已日益为人们所重视。它将工程图形语言及方法与崭新的计算机技术相结合，给生产、科研、管理等领域提供了高速度、高效率和高精度的图形设计及输出方法。它已在航空航天、船舶、汽车、机械、建筑、电子、地质、气象、轻工、管理等各个部门得到了广泛的应用。

在计算机绘图的应用软件中，Auto CAD 绘图软件是当前最为流行的 CAD 软件之一，它具有交互功能强、绘图方便、便于二次开发等优点，成为实际应用中客户最多、最受工程技术人员欢迎的绘图软件。

为了普及计算机绘图技术，各大专院校相继开设了计算机绘图课程。但从国内出版的各类计算机绘图教材来看，以介绍计算机绘图原理与方法的书为多。而轻工、化工类各专业的学生，毕业后则主要使用绘图软件绘图。目前国内适合学生使用的，以介绍绘图软件为主、绘图理论为辅的教材尚不多见。为此，根据轻工、化工类专业学生的特点，在总结了多年教学经验的基础上，编写了这本《Auto CAD 计算机绘图基础教程》。

全书共分十四章。前两章为绪论和绘图基本原理，主要介绍计算机绘图的常用设备，Turbo C 语言绘图函数，二、三维图形变换，窗口与裁剪技术。后面十二章介绍 Auto CAD R13 的使用方法。其中，Auto CAD 绘图基础一章主要介绍系统要求、启动、操作、环境等基本知识。后面章节则较为系统地介绍了 Auto CAD 绘图、编辑、图层、尺寸标注与剖面线、辅助工具等各种命令，以及三维绘图、二次开发技术、Auto LISP 语言等。最后简单介绍了新近面世的 Auto CAD R14 的新增功能。

本书可在大专院校二年级以上班级讲授，大学一年级的基础教学（包括线性代数）及 C 语言为本书的先修课程。讲授内容可根据专业情况及其它要求进行取舍。由于该课是一门实践性很强的课程，应让学生有尽可能多的上机时间，自己上机操作，这是学好本课的基本保证。

参加本书编写的学校有山东轻工业学院、大连轻工业学院、齐齐哈尔轻工学院、北京轻工业学院、深圳大学、桂林航天工业高等专科学校、辽宁省农业工程学校、大连电视大学、沈阳电视大学。

目　录

第一章 绪 论

第一节 概 述

计算机图形学（Computer Graphics）是随着计算机的发展而形成的计算机应用领域的一个重要分支，它是传统图学、几何学与现代的计算机技术相结合而形成的一门新兴学科。

作为计算机图形学的重要内容之一，计算机绘图将传统的工程绘图技术与计算机有机地结合起来，使得工程图学这一传统的学科进入了近代技术的行列。

计算机绘图的优点是速度快、精度高，且能绘制复杂的曲线、曲面图形。尤其是在汽车、飞机和造船等行业中，有许多图形很复杂，手工绘制相当困难，近些年都逐步由计算机绘图所取代。图 1－1 所示是由计算机绘出的一大型船体曲面图，其绘图速度和精度是人工绘图无法相比的。

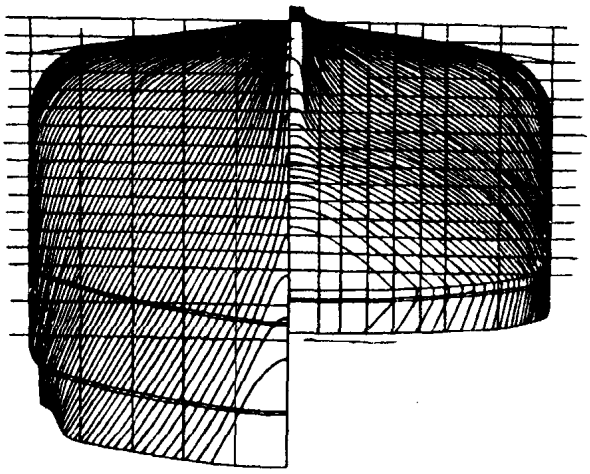


图 1－1

在现代化生产中，为了不断更新产品，提高生产率，降低成本，就必须缩短设计、绘图与制造周期，其有效途径是利用计算机辅助工程，主要是计算机绘图（CG）、计算机辅助设计（CAD）、计算机辅助制造（CAM），以实现设计、绘图和制造管理的全自动化。而计算机绘图在此有着重要的作用。

随着计算机与智能绘图机的迅速发展，计算机绘图越来越普及，不仅在工业生产有广泛的应用，在医学、气象、军事、教学、管理及影视、出版业中的作用也日益增加，并且计算机绘图还在向诸多方向迅速发展，如由静态绘图向动态绘图方向发展；由二维图形软件向三维实体造型方向发展；向 CAD、CAM、CG 三位一体化方向发展；向多媒体方向发展。

第二节 图形输入输出设备

一、计算机绘图系统

计算机绘图系统包括一台主机、图形输入设备和图形输出设备以及相应的图形支持软件。主要的图形输入设备包括键盘、数字化仪、鼠标器、扫描仪等。图形输出设备主要有显示器、绘图机、打印机等。计算机绘图的系统配置如图 1－2 所示。

二、输入设备

1. 键盘

键盘是计算机系统不可缺少的设备，也是常用的图形输入设备之一。通过键盘，可以输入字符、图形数据及命令等。

键盘由若干键、相应的开关元件、编码器及寄存器等组成。按键的作用分为有字符键、光标键、数字键及编辑功能键等。

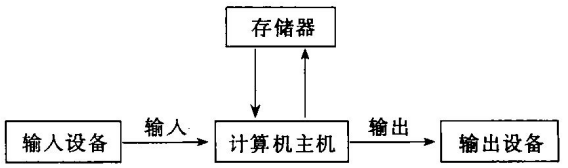


图 1-2

2. 数字化仪

它是一种图形数据坐标输入设备。利用数字化仪，可将图样上的点或线变成坐标数据输入计算机。数字化仪有机械式、超声波式及电磁式等几种，由图形输入设备将点的坐标数据送入计算机的功能称为定位功能。通过移动数字化仪面板上的指示器带动屏幕光标移动，以选取所操作的图形操作，这种功能称为拾取功能。另外，利用数字化仪还可选择菜单，从而执行该菜单所对应的功能，这称为选择功能。

3. 鼠标器

鼠标器有光电式和机械式两种。

光电式鼠标器利用光电管在特制的反射衬垫上检测鼠标器的移动。而机械式鼠标器利用鼠标器底面上的滚轮转动，使电位器移动，将位置信号送给系统。鼠标器主要用来实现图形的定位、拾取与选择功能，但它的定位功能是由屏幕光标位置提供的，其坐标精度不如数字化仪精确。但鼠标器结构简单，价格低廉，目前已广泛用于计算机系统。

4. 扫描仪

扫描仪是一种将各种形式的图像信息输入计算机的设备。按扫描图像幅面的大小可分为小幅面的手持式扫描仪、中等幅面的台式扫描仪和大幅面的工程图样扫描仪。

扫描仪是利用其光源照射到图像表面后的漫反射光线，经过 A/D（模/数）转换和适当的处理，使图像数据存储到计算机中并使之重新显示在屏幕上的。

三、输出设备

1. 图形显示器

图形显示器是一种用于输出字符或图形的设备。现在，绝大多数计算机系统均配有带图形显示功能的显示器。由于可通过图形、字符实现操作者与计算机对话，且可实现修改和编辑，目前已成为图形系统中的关键设备。显示器分单显和彩显两种，其主要指标为屏幕的分辨率。其分辨率从 320×200 到 4096×4096 不等，目前微机系统较常用的为 VGA640×480。

2. 绘图机

绘图机是计算机绘图系统中重要的输出设备，它可以高速度绘制高精度的单色或彩色图形。绘图机主要有以下几种类型。

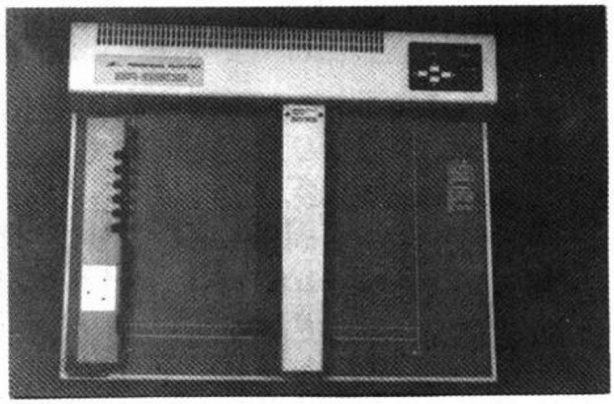


图 1-3

(1) 平台式

图 1-3 是典型的平台式绘图机。其特点是工作台面大（幅面可达 A1 以上），绘图精度高。绘图时，图纸固定在台面上。通过绘图笔在 X、Y 方向上运动而画

出图形。这种绘图机要求制造精度较高，因而价格比较昂贵。

(2) 滚筒式

图 1－4 是滚筒式绘图机的外形图。滚筒式绘图机是利用装在滚筒上的绘图纸随滚筒的来回移动形成 X 方向的运动，再由电动机带动笔架移动实现 Y 方向移动，由 X、Y 方向的组合运动来完成绘图的。这种绘图机结构简单，绘图速度较高，价格较平台式绘图机低，但绘图精度较低。

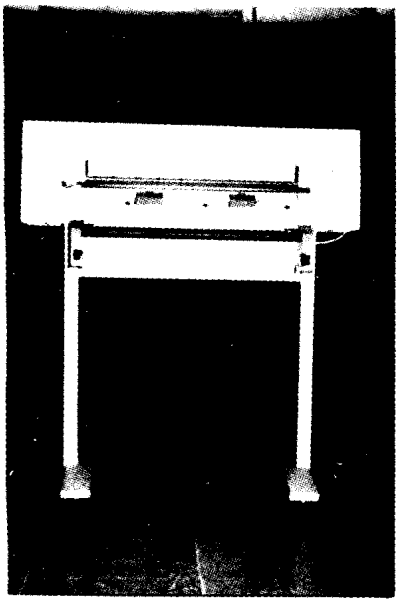


图 1－4

3. 打印机

由于打印机能够迅速打印图形，因而可利用它来检查所画的图形。

打印机有点阵式打印机、喷墨打印机和激光打印机三种。一个 9 针打印机的标准输出是每英寸 120×120 点，通过使用加强型打印机软件，可增加到每英寸 240×180 点。24 针打印机还可有更高的分辨率。

激光打印机能打印出比点阵式打印机质量更高的图线，且打印速度更快。但激光打印机的打印区域通常限制在 8½in×11in 的范围。

图 1－5 是一套典型的微型计算机绘图系统的配置示意图。

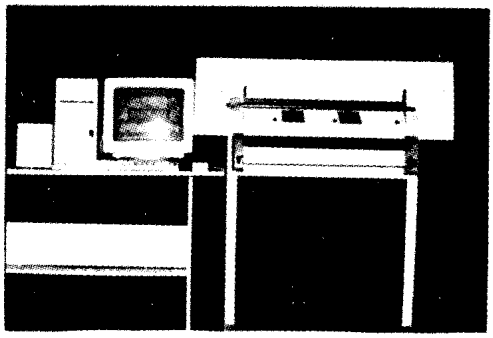


图 1－5

第三节　Auto CAD 图形软件简介

CAD 是 Computer Aided Design 的缩写，意为计算机辅助设计，它是利用计算机绘制和生成工程图纸的一种现代高新技术。目前这项技术已广泛应用在建筑、电子、机械等领域中。随着计算机性价比的进一步提高和 CAD 技术的飞跃发展，各种 CAD 软件都由原始的二维绘图转变成三维造型，实现了二维与三维混合，线框和曲面及实体混合等新技术。Auto CAD 是由美国 Autodesk 专门为微机绘图而设计的应用软件，它是在众多的 CAD 软件中功能比较完善、深受用户欢迎、市场占有率很高的一种软件。所以，本书介绍 Auto CAD 的使用方法。

一、Auto CAD 的发展概况

CAD 的开发最初是 20 世纪 60 年代，开始时由于 CAD 程序价格昂贵，使用困难，再加

上运行 CAD 的计算机是庞大的、昂贵的，所以 CAD 的用户并不多。随着计算机硬件和软件的发展，CAD 的功能越来越强大，界面越来越友好，使用越来越容易。到现在，Auto CAD 已经是计算机辅助设计的主流。

最先采用 Auto CAD 技术是在 1982 年，它在装有 640KB 内存、DOS 环境、IBM XT 系统下就可以运行。Auto CAD 的早期版本只是生成二维图形的简单工具，但后来，Auto desk 公司投入了大量的人力和物力，由最初的 1.0 版本逐步升级到 2.0、2.01、2.17、2.18、2.5、2.62、R9、R10、R11、R12、R13，现在已发展到了 R14。从 R12 版后，又有 for DOS 和 for Windows 两种版本。由于 R14 版是新发行的版本，国内还没有普及，所以本书以 Auto CAD R13 for Windows 版为蓝本，介绍 Auto CAD 的使用方法。实际上，高版本只是在低版本作了部分改进，学会了一种版本的使用，学习使用其它版本就容易多了。

二、Auto CAD R13 的基本功能

Auto CAD 是具有强大功能的绘图软件包，主要功能表现在下面几点：

1. 具有完善的绘图功能

Auto CAD R13 有丰富的二维绘图命令，比如绘制点、直线、曲线、多义线、圆、弧、椭圆、多边形、剖面线等，并且线型、颜色等由用户定义。Auto CAD R13 还可以生成三维图形，如 3D 平面、曲面、三维实体模型等，并且新版本的模型渲染和效果展示功能逐步加强。利用 Auto CAD 提供的实体绘图命令可以创造任何复杂的图形。

2. 具有较强的图形编辑功能

传统的手工绘图，如果发现绘制有错误或布局不合理，轻则用橡皮擦除，然后重新绘制，重则作废已画好的图纸，重新构图。很多设计师都饱尝过其中的辛苦和乏味。使用 CAD 系统，可以随心所欲地编辑，包括删除、移动、旋转、复制、修剪、变形、断开、倒角、缩放等，并且图纸上不留任何编辑的痕迹。

3. 标注功能

使用传统的绘图方法，图形尺寸标注是令设计者很苦恼的事情，但自从有了 CAD 后，这件工作却变得非常容易和直观。对于各种类型的标注，只要用鼠标点两下，计算机就可以自动完成标注。例如：标注长度时，只要用鼠标点击要标注的长度的两个边界，计算机便自动将尺寸界线、尺寸线、尺寸箭头、尺寸文本自动标出，并且界线的长度、箭头的形状、文本的高度都可由用户自由定义。

4. 具有直观的三维图形功能

过去 CAD 主要是一个二维软件，但 R13 版具有很强的三维功能。它有 X，Y，Z 三个坐标值，不但可以画出立体图，也可以改变视线角度，从三维空间任意点来观察物体，还可以画出轴测投影图或透视投影图。如果加上灯光和材质，便可把三维物体渲染成真实的三维物体。

5. 向用户提供了多种开发手段

从 2.5 版开始，Auto CAD 中嵌入了 AutoLISP 语言。使用这种语言，可以把 Auto CAD 的基本命令同 Auto LISP 函数结合起来，既能进行计算又能调用 Auto CAD 命令。通过图形交换文件等可以与其它高级语言或系统连接。通过二次开发，很容易扩展其功能，方便用户的使用。

Auto CAD R13 不仅有上述功能，还有文本功能、阴影修饰功能、输入输出等功能。

第二章　计算机绘图原理

第一节　Turbo C语言绘图函数

一、图形模式

在IBM个人计算机上，屏幕输出的能力决定于两项因素，一项是显示适配器(adapter)，另一项是显示监视器（moniter)。适配器是硬件的适配器，插在扩充槽上，有的称为图形显示适配器，而在IBM PS/2个人计算机主机板里就有内建图形的适配器。而监视器是个软件程序，它负责监视将实际的字符和图形输出显示。因此市面上有各种标准的显示器可供选用，IBM个人计算机上可用的显示适配器就有四种，分别是：

1）单色显示适配器（Mono Chromc Display Adapter，简称MDA）

2）彩色图形适配器（Color Graphics Adapter，简称CGA）

3）高强度图形适配器（Enhanced Graphics Adapter，简称EGA）

4）赫氏图形卡（Hercules Graphics Card，简称HGC）

MDA卡有80×25的文本单色显示字幕，没有图形功能，HGC卡能显示文本，像MDA卡，同时也有单色显示720×348的绘图能力。CGA卡有显示文本和图形的彩色显示能力。前景颜色有16色，背景颜色有8色，图形能有4色显示且有320×200的分辨率，若选择2色（黑和白色）时，则其分辨率为640×200。EGA卡1984年由IBM公司推出，有更多颜色，分辨率更高，它共有16种颜色，640×350分辨率。

Turbo C提供相当多的库函数来支持图形上的显示能力，只是用户的计算机配备须有图形适配器，同时为了考虑不同接口图形显示能力不同，譬如分辨率、颜色等各种因素，Tubro C的开发公司——Borland公司，开发了图形设备驱动程序并使驱动程序能兼容于各种适配器，因为当图形模式引导时，调用一个initgraph函数，它将会检测系统上所拥有的设备，而装入该设备的驱动程序，使图形输出例程在不同的设备下也照常运行。而不同适配器上的驱动程序，Turbo C系统分别收集归纳在BGI文件（Borland Graphics Interface）里，所有有关图形的例程全储存在头文件graphics.h里。本节为使初学者容易进入Turbo C的领域，采用渐进式逐步介绍常用的函数及应用例子。

1. 图形模式引导、设置所需使用的相关函数与常量

(1) int graphdriver=DETECT;

在Turbo C共提供11种图形驱动程序，见表2-1。若graphdriver变量设置为DETECT常量，则引导函数initgraph会自动调用detectgraph函数来检查系统的设备，选择适当的驱动程序和适合适配器的高分辨率的相关信息给系统，以便进入图形模式。

(2) Void far initgraph (int far * graphdriver, int far * graphmode, char far * pathtodriver);

使用该函数可初始化Turbo C的绘图系统。

int far *garphdriver 是整数指针，其拥有的数值代表所属的图形驱动程序，见表 2－1。通常不晓得何种设备时，就指定 DETECT 常量，则此函数会自动调用 detectgraph 函数来检查。

表 2－1　Turbo C 图形驱动程序

图形驱动程序的列举常量	数值	说　　明
DETECT	0	要求自动测试
CGA	1	彩色图形适配器
MCGA	2	多色图形数组
EGA	3	128K 或以上的内存空间的高强度图形适配器和彩色显示监视器
EGA64	4	64K 内存空间的高强度图形适配器和彩色显示监视器
EGAMONO	5	高强度图形适配器和单色监视器
IBM8514	6	IBM8514 图形卡 808514 仿真监视器
HERCMONO	7	赫氏图形卡和单色监视器
ATT400	8	at&T400 行图形适配器
VGA	9	图形图符数组和模拟监视器
PC3270	10	IBM3270 个人计算机图形适配器

int far *graphmode 是整数指针，代表图形模式，见表 2－2。

表 2－2　图形模式

模 式 常 量	说　　明			图形驱动程序
CGAC0	320×200	palette	0，1 page	CGA
CGAC1	320×200	palette	1，1 page	CGA
CGAC2	320×200	palette	2，1 page	CGA
CGAC3	320×200	palette	3，1 page	CGA
CGAHI	640×200	2-color	，1 page	CGA
MCGAC0	320×200	palette	0，1 page	MCGA
MCGAC1	320×200	palette	1，1 page	MCGA
MCGAC2	320×200	palette	2，1 page	MCGA
MCGAC3	320×200	palette	3，1 page	MCGA
MCGAMED	640×200	2-color	1 page	MCGA
MCGAHI	640×480	2-color	1 pages	MCGA
EGAL0	640×200	16-color	4 pages	EGA
EGAHI	640×350	16-color	2 page	EGA
EGA64L0	640×200	4-color	1 page	EGA64
EGA64HI	640×350	4-color	page	EGA64
EGAMONOHI	720×348	2-pages		EGAMONO
HERCMONOHI	720×348	2-pages		HERCOMONO
ATT400C0	320×200	palette	0，1 page	ATT400
ATT400C1	320×200	palette	1，1 page	ATT400
ATT400C2	320×200	palette	2，1 page	ATT400
ATT400C3	320×200	palette	3，1 page	ATT400
ATT400MED	640×200	1 page		ATT400
ATT400HI	640×400	1 page		ATT400

（续）

模式常量	说明			图形驱动程序
VGAL0	640×200	16-color	4 pages	VGA
VGAMED	640×350	16-color	2 pages	VGA
VGAHI	640×480	16-color	1 pages	VGA
PC3270HI	720×350	1 page		PC3270
IBM8514L0	640×480	256 colors		IBM8514
IBM8514HI	1024×768	256 colors		IBM8514

char far * pathtodriver 是字符串指针，表示图形驱动程序所在处的路径名称，譬如是在C驱动器的 TURBO 目录下，则此参数应为“C: \\ TURBO”。假使驱动程序不在指定的目录下，此函数会自动到当前任务目录下查找 BGI 文件。换句话说，若 * pathtodriver 参数若是空字符串（NULL）时，则会在当前的任务目录下查找。

此函数 initgraph 初始化时，若有错误发生，会有错误代码设置，同时拷贝此错误代码到整数指针 graphdriver，可使用 graphresult 函数来获得其错误代码，见表 2－3。

（3）Void far detectgraph（int far * graphdriver，int far * graphmode）；

使用此函数可用来检测系统的绘图设备，而获得其图形模式与分辨率等图形适配器与监视器的相关信息，调用此函数后通常再调用 graphresult 函数，以测试是否有绘图能力。

（4）int far graphresult（void）；

使用此函数可获取最近一次图形操作的错误代码，各种可能的错误代码见表 2－3。

综合前述四项函数的用法，Turbo 绘图的开始，应有类似如表 2－4 的定义指令。

表 2－3　图形错误常量

图形错误常量	value	错误讯息
grok	0	没有错误
grNoInitGraph	－1	(BGI) 图形驱动程序没有安装
grNotDetected	－2	图形硬件配备没有检测
grFileNoFound	－3	驱动程序文件没有找到
grinvalidDriver	－4	无效的驱动程序文件
grNoloadMem	－5	没有足够空间装入驱动程序
grNoFloodMem	－6	在扫描填图时空间不够
grNoFloodMem	－7	在填充图形时空间不够
grFontNotFound	－8	图形码文件找不到
grNoFontMem	－9	没有足够空间装入图形码
grInvalidmode	－10	无效图形模式
grError	－11	图形错误
grIoError	－12	图形输入输出错误
grInvalidFont	－13	无效图形码文件
grInvalidFontNum	－14	无效图形码代号
grInvalidDeviceNum	－15	无效的设备代号
grInvalidversion	－18	无效的版本

表 2－4　绘图程序起始时应有定义与指令

```
1       #include  <graphics.h>
2       #include  <stdlib.h>
3       main ()
4       {
5           int graphdriver = DETECT;
6           int graphdriver = DETECT;
7           initgraph (&graphdriver, &graphmode,"");
8           if (graphresult ()! = grok)
9           {
10          printf (" Can not graphics \ n")
11          exit (1);
12      };
```

(5) void far closegraph (void)

使用此函数可释放掉 Turbo C 绘图系统所分配的内存空间，关闭图形系统，使屏幕回到引导绘图系统前的原来模式。即一旦采用图形模式，则在程序结束前最后一条语句，应该就是调用此函数来结束图形模式，并重新设置回到进入绘图系统前的屏幕模式。

(6) void far restorectrmode (void);

使用此函数是将屏幕重新设置回到调用 initgraph 函数前的原来屏幕模式。一般来说，原来屏幕模式即文本模式，也就是此函数允许由图形模式回到文本模式。

(7) char * far grapherrormsg (int error code);

使用此函数可将获得的错误代号转换成错误信息。

(8) void far graphdefaults (void);

使用此函数可重新设置绘图系统的初设值。

(9) void far cleardevice (void);

使用此函数可清除绘图屏幕。

(10) void far setgraphmode (int mode);

使用此函数可转移到另一个不同的图形模式并清除屏幕。有关图形模式，请见表 2－2。

(11) void far setpalette (int pixel-value, int color);

使用此函数可更改调色板的某一颜色。此函数可用于更改 EGA 或 VGA 调色板颜色内容。譬如：setpalette (0, GREEN); 表示将当前的第一个颜色（背景颜色）改为绿色。在 CGA 驱动程序下，调用此函数只能更改背景颜色。在 IBM8514 驱动程序中不能调用此函数，应改调用 Setrgbpalette。表 2－5 是 graphics.h 头文件已定义的各驱动程序的实际调色板。

(12) void far setallpalette (struct palettetype far * palette);

使用此函数可根据指针结构 palette 参数内所设置的颜色，来改变调色板内的所有颜色。Struct palettetype 的结构在 graphics.h 文件中定义如下：

```
#define MAXCOLORS 15
Struct palettetype
```

{

unsigned char size;

signed char colors〔MAXCOLORS+1〕;

};

栏项 size 是指颜色数组所拥有的实际颜色的个数。

栏项 colors 数组是将欲更改的新调色板的值放在此数组中。而如果数组中的值放 -1，代表不更改其颜色。此函数适用于 EGA 和 VGA 环境，而有关调色板见表 2-5。

(13) void far setbkcolor (int color);

使用此函数可用来设置新的背景颜色，颜色常量见表 2-5。

(14) void far setcolor (int color);

使用此函数可用来设置当前的图形颜色，颜色常量见表 2-5。

表 2-5 颜色对照表

适配器	颜色常量	数值	适配器	颜色常量	数值
CGA	BLACK	0	EGA/VGA	EGA-BLACK	0
	BLUE	1		EGA-BLUE	1
	GREEN	2		EGA-GREEN	2
	CYAN	3		EGA-CYAN	3
	RED	4		EGA-RED	4
	MAGENTA	5		EGA-MAGENTA	5
	BROWN	6		EGA-BROWN	20
	LIGHTGRAY	7		EGA-LIGHTGRAY	7
	DARKGRAY	8		EGA-DARKGRAY	56
	LIGHTBLUE	9		EGA-LIGHTBLUE	57
	LIGHTGREEN	10		EGA-LIGHTGREEN	58
	LIGHTCYAN	11		EGA-LIGHTCYAN	59
	LIGHTRED	12		EGA-LIGHTMAGENTA	61
	LIGHTMAGENTA	13		EGA-YELLOW	62
	YELLOW	14		EGA-WHITE	63
	WHITE	15			

(15) unsigned far setgraphbufsize (unsigned bufsize);

使用此函数可用来设置图形缓冲区的大小，初值是 4096 个字节 (4KB)。

(16) int far graphmode (void);

使用此函数可返回当前的图形模式。

(17) struct palettetype * far getdefaultpalette (void);

使用此函数要返回当前图形模式下的调色板的定义结构，其结构类型见第 12 项的介绍。

(18) int far getbkcolor (void);

使用此函数可返回当前图形模式下的背景颜色值。

(19) int far getcolor (void);

使用此函数可返回当前图形模式下的前景颜色值。

(20) int far getmaxcolor (void);

使用此函数可返回当前图形模式下该绘图驱动程序所能使用的最大颜色编号，该编号可给 setcolor 函数使用。譬如：HERMONO 适配器的 getmaxcolor 返回值为 1，因为是单色，所以只有 0 和 1 两种颜色。

(21) void far getmoderange (int graphdriver, int far *lomode, int far *himode);

使用此函数可获得指定的绘图驱动程序的模式范围，见示例 2-1。

2. 与点有关的例程

(1) int far getx (void);

使用此函数可返回绘图屏幕当前横坐标值。

(2) int far gety (void);

使用此函数可返回绘图屏幕当前纵坐标值。

(3) int far getmaxx (void);

使用此函数可返回绘图屏幕的最大行数。

(4) int far getmaxy (void);

使用此函数可返回绘图屏幕的最大列数。

(5) unsigned far getpixel (void);

使用此函数可返回绘图屏幕指定位置点的颜色值。

(6) void far putpixel (int x, int y, int color);

此函数可按照指定横坐标 x 与纵坐标 y 的位置，按指定的颜色画出一点。

示例 2-1:

```
#include<graphics.h>
#include<stdio.h>
#include<math.h>
#include<stdlib.h>
main()
{int graphdriver=DETECT,graphmode=DETECT;
 initgraph(&graphdriver,&graphmode,"");
 setlinestyle(0,0,1);
 setcolor(4);
ellipse(fx(60),fy(40),0,180,25,25);
line(fx(85),fy(40),fx(85),fy(-60));
line(fx(35),fy(40),fx(35),fy(-60));
ellipse(fx(60),fy(-60),180,0,25,25);
line(fx(85),fy(40),fx(35),fy(40));
line(fx(85),fy(-60),fx(35),fy(-60));
line(fx(85),fy(40),fx(35),fy(-60));
line(fx(85),fy(-60),fx(35),fy(40));
line(fx(150),fy(100),fx(150),fy(150));

 setlinestyle(0,0,3);
 setcolor(2);
```

```
line(fx(60),fy(65),fx(60),fy(100));
line(fx(60),fy(100),fx(215),fy(100));
line(fx(215),fy(100),fx(215),fy(60));
line(fx(215),fy(30),fx(215),fy(-40));
line(fx(215),fy(-70),fx(215),fy(-120));
line(fx(60),fy(-85),fx(60),fy(-140));
line(fx(60),fy(-140),fx(-60),fy(-140));
line(fx(-60),fy(-140),fx(-60),fy(-120));
line(fx(-60),fy(-20),fx(-60),fy(5));
line(fx(-60),fy(5),fx(-180),fy(5));
line(fx(-127),fy(-100),fx(-127),fy(-40));
line(fx(-200),fy(70),fx(-200),fy(120));
line(fx(-200),fy(120),fx(-280),fy(120));
 setlinestyle(0,0,1);
 setcolor(4);
line(fx(240),fy(60),fx(190),fy(60));
ellipse(fx(190),fy(45),270,90,15,15);
line(fx(240),fy(30),fx(190),fy(30));
```

此示例是示范由计算机任意产生 0 到 99 的随机数，当做画图的横、纵坐标点，在屏幕的左上角任意地画点，直到按任意键才结束。

3. 与画线有关的例程

(1) void far line (int x1, int y1, int x2, int y2);

此函数可在两个指定的坐标点 (x1, y1) 和 (x2, y2) 间，以系统当前的颜色和线型画一条线。

(2) void far lineral (int dx, int dy);

此函数可以从当前屏幕的位置，水平位移 dx，垂直位移 dy 的位置，以系统当前的颜色和线条形式画一条线。

(3) void far lineto (int x, int y);

此函数可从当前屏幕的位置到指定坐标点(x,y)间,以系统当前的颜色和线型画一条线。

(4) void far setlinestyle (int linestyle, unsigned upattern, int tickness);

此函数可用来设置绘出线条的宽度与形式，在 Turbo C 系统中有关线型已定义如表 2-6，宽度见表 2-7。

表 2-6 Turbo C 图形线条样式

线条样式常量	值	说　明
SOLID-LINE	0	实心线
DOTTED-LINE	1	点　线
CENTER-LINE	2	中心线
DASHED-LINE	3	长虚线
USERBIT-LINE	4	用户定义线条形式

表 2-7 Turbo C 图形线条宽度

线条宽度常量	值	说　明
NORM-WIDTH	1	正常宽度线条（一个像素）
THICK-WIDTH	3	粗宽度线条（三个像素）

(5) void far getlinesettings (struct linesettingstype far * lineinfo);

使用此函数是获取当前的线条的形式 (style), 样式 (pattern) 和宽度。结构类型 linesettingstype 定义在 graphics.h 文件里, 其定义格式如下:

```
Struct linesettingstype
{
  int linestyle;
  unsigned upattern;
  int thickness;
};
```

其中项 upattern 只有在 linestyle 为 4 时 (即 USERBIT _ LINE 常量), 其数值才有意义。譬如: 在设置线型与宽度时, 函数调用情况如下:

```
Setlinestyle (USERBIT _ LINE, OXOFOF, NORM _ WIDTH);
```

在 16 位的样式 upattern 里, 该位为 1 时对应该位的像素就画出当前的颜色, 因此 OXOFOF 事实上是画出波折线。如果要画出实线, 则 upattern 值应为 OXFFFF。故当线型不是用户定义线型时, 则设置时 upattern 参数也要传递, 但值会被忽略掉。

(6) void far moveto (int x, int y);

使用此函数是移动屏幕坐标点到指定 (x, y) 坐标点。

(7) void far moverel (int dx, int dy);

使用此函数可从当前坐标点水平位移 dx, 垂直位移 dy, 将其图形坐标点移到位移后的新坐标点。

(8) void far arc (int x, int y, int stangle, int endangle, int radius);

使用此函数是根据提供的中心点 (x, y), 半径 radius 值, 起始角度 stangle 和结束角度 endangle 画出一条弧线。

(9) void far getarccoords (struct arccoordstype far * arccoords);

使用此函数可返回最近一次调用 arc 函数的弧度起始与结束的两个坐标点及中心点的坐标点。结构类型 arccoordstype 在 graphics.h 文件里已定义, 其格式定义如下:

```
Struct arccoordstype
{
  int x, y;
  int xstract, ystrart, xend, yend;
};
```

(10) void far getaspectratio (int far * xasp, int far * yasp);

此函数返回图形模式的水平垂直坐标化值, 分别存于整数指针 xasp 和 yasp 所指的位置。此函数的主要目的是画图形、弧形、拼图形时, 根据其纵横比来修正水平与垂直的偏移使其弧度更正确。

示例 2-2:

```
line(fx(240),fy(60),fx(240),fy(30));
line(fx(235),fy(30),fx(235),fy(60));
line(fx(240),fy(-40),fx(190),fy(-40));
```