计算机图形学与动画技术

于万波　编著

清华大学出版社

计算机图形学与动画技术

于万波　编著

清华大学出版社
北京

内 容 简 介

本书是计算机图形学的入门教程，以 Visual C++ 与 OpenGL 为工具讲解计算机图形学以及动画制作的基本知识。本书主要内容包括 Visual C++ 绘图相关类及函数的使用、二维直线曲线的绘制方法以及区域填充的基本内容；通过一些典型实例介绍 OpenGL 和学习三维图形投影、消隐、光照等内容；分类讲解动画制作与实例分析。

本书可作为数字媒体技术专业、计算机科学与技术专业、信息与计算科学专业、软件工程专业、机械或建筑设计等专业的计算机图形学教材，也可以供对计算机图形动画制作感兴趣的程序设计人员使用。

图书在版编目（CIP）数据

计算机图形学与动画技术/于万波编著. —北京：清华大学出版社，2011.8
ISBN 978-7-302-26164-3

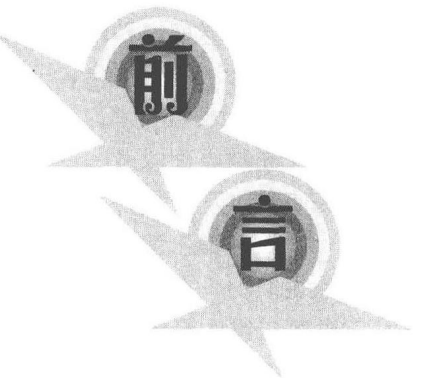
Ⅰ. ①计… Ⅱ. ①于… Ⅲ. ①计算机图形学 ②计算机图形学—应用—动画片—制作
Ⅳ. ①TP391.41 ②J954-39

中国版本图书馆 CIP 数据核字(2011)第 136889 号

责任编辑：龙啟铭　赵晓宁
责任校对：焦丽丽
责任印制：何　芊

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：010-62795954，jsjjc@tup.tsinghua.edu.cn
质　量　反　馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 装 者：三河市春园印刷有限公司
经　　销：全国新华书店
开　　本：185×260　　**印　张**：16.5　　**字　数**：385 千字
版　　次：2011 年 8 月第 1 版　　**印　次**：2011 年 8 月第 1 次印刷
印　　数：1～3000
定　　价：26.00 元

产品编号：038587-01



计算机图形学（Computer Graphics，CG）的研究与应用仍然是目前科学及工程领域的一个热点。计算机图形与动画的应用领域十分宽广，除了用来制作影视作品外，在科学研究、视觉模拟、电子游戏、工业设计、教学训练、写真仿真、过程控制、平面绘画、建筑设计等许多方面都有重要应用。一些高校的计算机科学与技术、信息与计算科学、机械设计、建筑设计、航空、汽车、地理信息、动画游戏、媒体信息、美术创作等相关专业都开设计算机图形类课程。在图形学这一学科分支领域中，既有理论深度，也有实际应用背景。进入该领域，一面是繁杂深刻而精细的算法程序，另外一面是漂亮优美而鲜活的图形动画，形成了鲜明的对照。也正因为如此，吸引着众多的爱好者。

计算机图形学是一种使用数学算法将二维或三维图形转化为计算机显示器的栅格形式的科学，就是研究如何在计算机中表示、计算、处理和显示图形的相关原理与算法。计算机图形学的研究内容非常广泛，如图形硬件、图形标准、图形交互技术、光栅图形生成算法、曲线曲面造型、实体造型、真实感图形计算与显示算法、非真实感绘制，以及科学计算可视化、计算机动画、自然景物仿真、虚拟现实等。

笔者这几年在讲授图形学的过程中进行摸索、思考、积累，考虑写一本不同风格的图形学教材，以实现教学方式的改革，供教师、同学以及其他读者使用，这样开始了这本教材的编写。笔者曾经编写《基于 MATLAB 的计算机图形与动画技术》教材，该书也是在清华大学出版社袁勤勇老师的策划下完成的。本书参考了其部分内容，延续了其思想方法，这两本书可以互相补充。喜欢 Visual C++的读者，或准备直接投入到开发产品工作中的读者可以选择这本《计算机图形学与动画技术》。

回顾近几年的积累以及几个月来的编写过程，编写这本书的指导思想是减轻学生入门图形学的难度，引导学生去思考一些图形学问题，通过实际操作激发学生的兴趣；以实例制作为主线，化解图形学的一些难点，把算法与理论公式等有机地穿插在例题习题之间，实现一种以归纳为主导的层嵌式递进学习模式。

在慎重的思考后，选择了 OpenGL 作为辅助工作，也许有的读者

通过学习会把这种辅助工具变为自己的主流开发工具。至少 OpenGL 可以作为一个阶梯、一个平台，借此平台，一方面去了解图形算法原理；另一方面直达图形动画产品设计现场。通过本书的学习，可以让读者掌握计算机绘图及动画制作的一般性知识，了解一些图形绘制原理以及一些经典算法，希望读者在了解与学习的基础上，根据具体情况对工具使用以及产品开发进行再深入的研究。

本书第 1 章介绍如何使用 Visual C++编写简单而有趣的图形动画程序，使读者掌握 Visual C++的图形绘制功能，了解计算机绘图的基本知识与概念。第 2 章通过多个实例讲解二维图形绘制与填充算法，包括直线与圆的绘制算法、二维插值拟合曲线绘制方法、平面多边形填充等，同时也介绍二维分形图的绘制。第 3 章讲解 OpenGL，主要有绘制多面体、交互操作、曲线曲面绘制、图像操作等内容，通过这些内容的学习，使读者从二维图形绘制开始逐步过渡到三维图形，从使用 C++ Source File 进行 OpenGL 程序设计过渡到使用单文档进行 OpenGL 程序设计。第 4 章是本书的重点，主要讲述了空间曲线曲面、几何造型、三维数据的二维投影、隐藏面计算、光照效果等内容，通过一些可操作的例题与习题，使读者理解并掌握三维图形学的基本内容与方法。在第 5 章中，首先介绍了动画制作的基本概念以及三维变换等内容，然后给出了一些动画制作实例，如飘动的图像、爆炸效果、转动的地球、飞机飞行与发射子弹等，通过这些实例逐步使读者了解并掌握动画制作技术。

本书主要涉及在语言层次上绘制图形与制作动画，以及图形动画函数库 OpenGL 的使用。实际上 OpenGL 中的函数对应图形动画软件的各个按钮及面板参数设置功能，所以通过 OpenGL 的学习，可以辅助图形绘制算法原理的学习；另一方面可以更好地领悟与学习图形动画软件。鉴于 Visual C++软件目前拥有众多的使用者以及它具有的一些优点，选择了 Visual C++作为平台工具。学习时应该多上机操作、多思考、多归纳总结，查阅经典的计算机图形学教材，了解常用的计算机图形动画制作软件，如 Flash、3ds Max 等。

本书的一些例题的源代码已经放在清华大学出版社的网站上，使用这些代码可以提高学习效率。在编写的过程中，参考了廖朵朵、张华军的《OpenGL 三维图形设计》，Nehe 教程的电子稿，这些电子书稿是在 CSDN 网站下载的，在此一并表示真心的感谢与诚挚的敬意！

杨灵芝参加了第 1 章的编写，臧超参加了第 2 和第 3 章的编写，杨雪松参加了第 4 章的编写，于硕参加了第 5 章的编写。于万波负责设计指导与统稿工作，并参与了全书各个章节的编写。学生李文杰、董佼靖、王琼瑶、李宁等的作业中提供的一些程序写在了各章习题中，在此表示感谢。

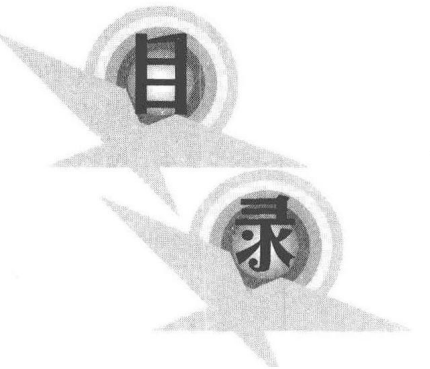
最后感谢龙启铭编辑的信任、支持与指导，感谢为本书编辑出版辛苦工作的编辑及其他工作人员。

期待着大家喜欢这本书，也期待得到批评与建议。

作者邮箱：yu_wb@sohu.com,yu_sb@126.com。

编 者

2010 年 11 月



第1章 Visual C++绘图程序设计

计算机绘图可以使用专门的绘图软件，如 Photoshop 等；可以使用专门语言，如 OpenGL 等。除了专门软件以及专门语言外，很多语言（软件）本身也提供了简单的绘图功能，Visual C++也有很多绘图函数。在 1.1 节中使用 Visual C++中 MFC 的 CDC 类封装的绘图函数绘制一些图形，除了学习 MFC 单文档中绘图功能外，还可以了解语言绘图的一些基本方式。在 1.2 节中通过实例深入研究 CDC 类封装的一些常用的绘图函数。在 1.3 节介绍画笔类与画刷类，以便实现更精细更丰富的绘制功能。在 1.4 节介绍位图图像操作。在 1.5 节给出一些绘图实例。最后在 1.6 节介绍使用 Win32 应用程序绘制图形与制作动画。

1.1 使用 CDC 类函数绘制图形

Visual C++功能强大，系统庞杂，把各种功能都分散到各个类中。其中绘图功能可以使用 CDC 类中封装的函数实现。如果建立单文档（或多文档）程序，那么在其 OnDraw 函数中写入 CDC 类函数构成的绘图语句就可以进行绘图。

1.1.1 使用单文档程序绘图

【例 1-1】 在 MFC 单文档程序的 OnDraw 函数中加入语句绘制直线。

该例题分以下几步完成：

1. 建立单文档工程

进入 Visual C++工作界面，执行 File→New 命令，在弹出的对话框中选择 Project 选项卡，如图 1-1 所示。

在该选项卡列出的各条目中，选择MFC AppWizard (exe)，在右面 Project name 栏写上工程名字（该例题把工程命名为 Huatu1），单击 OK 按钮，出现图 1-2 所示界面。

在该界面上选择 Single document 单选按钮，然后单击 Finish 按钮。在随后弹出的对话（框）窗口中单击 Finish 按钮，出现了 Visual C++的工作界面，如图 1-3 所示。

在图 1-3 中，左面有文件组织列表窗口，在这个窗口中有各个文件的组织关系。

Source Files 称为源文件。

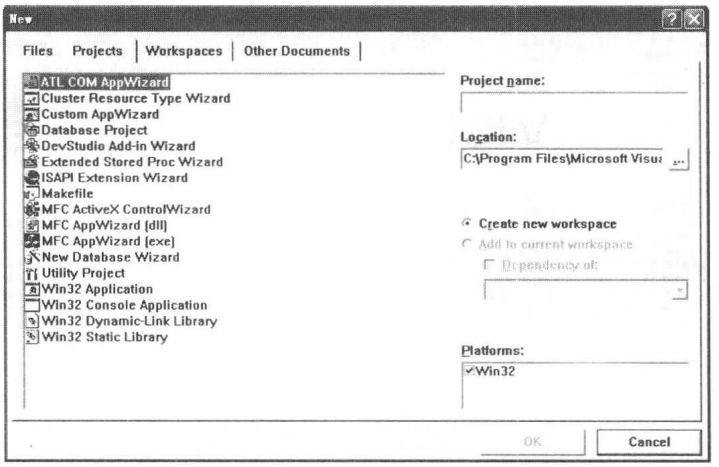


图 1-1　New 对话框窗口（Projects 选项卡）

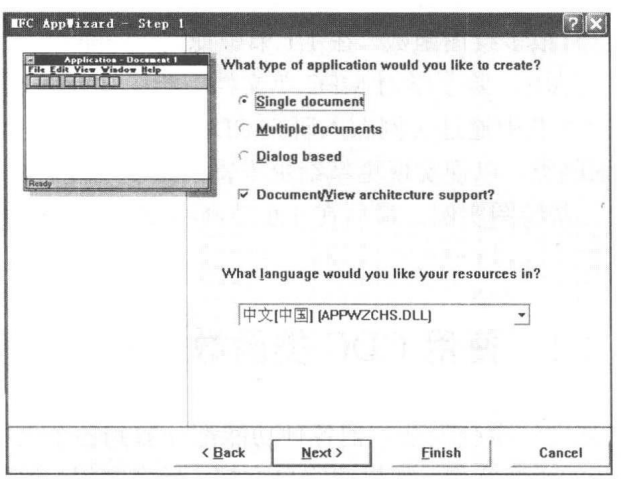


图 1-2　确定单文档程序类型对话框

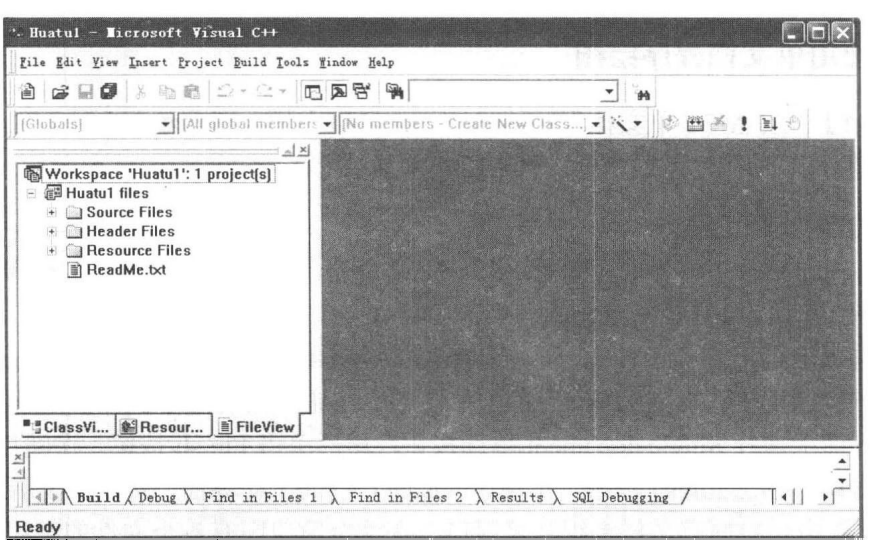


图 1-3　单文档程序设计界面

Header Files 称为头文件。

Resource Files 称为资源文件。

上面这 3 个文件夹都可以单击其左面的⊞展开，也可以在文件夹上双击展开。

2. 在 OnDraw 函数中加入语句

展开 Source Files 文件夹，双击 Huatu1View.cpp，在右面工作区中显示出该文件的内容，拉动滚动条滑块，找到函数 void CHuatu1View::OnDraw(CDC* pDC)，如图 1-4 所示。

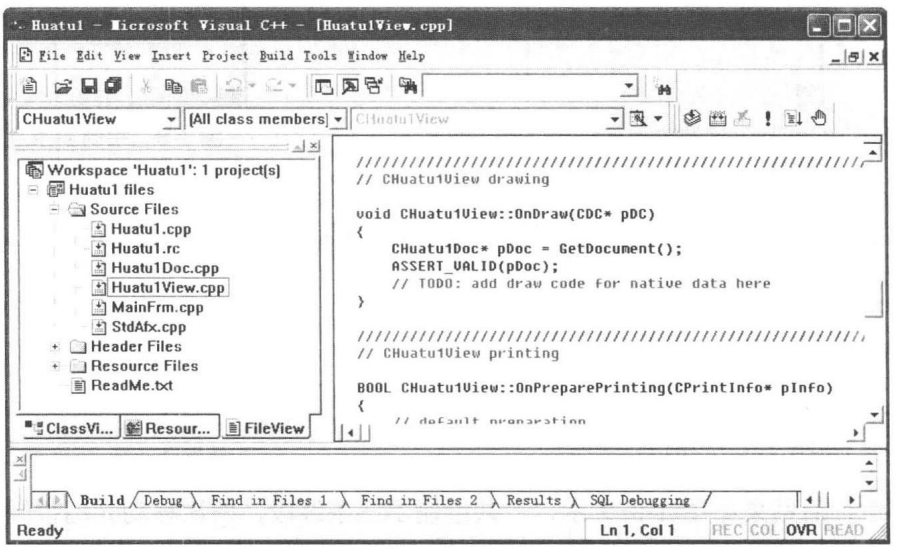


图 1-4　单文档程序设计界面

观察图 1-5 所示的函数 OnDraw(CDC* pDC)。

在这个函数的后面加上语句 pDC->LineTo(200,100);，如图 1-6 所示。

```
void CHuatu1View::OnDraw(CDC* pDC)
{
    CHuatu1Doc* pDoc = GetDocument();
    ASSERT_VALID(pDoc);
    // TODO: add draw code for native data here

}
```

图 1-5　OnDraw 函数

```
void CHuatu1View::OnDraw(CDC* pDC)
{
    CHuatu1Doc* pDoc = GetDocument();
    ASSERT_VALID(pDoc);
    pDC->LineTo(200,100);
}
```

图 1-6　在 OnDraw 函数中添加绘制直线语句

注意：语句// TODO: add draw code for native data here 是注释，可以删掉，不影响语句的执行。

3. 编译运行

单击工具栏上的“项目编译”按钮，或单击“编译”菜单 Build 中的 Build Huitu1.exe　F7 编译项目；然后单击工具栏上的“运行”按钮！，或“编译”菜单中的选项 ！执行 Huatu1.exe　Ctrl+F5，运行程序，运行结果如图 1-7 所示。

语句 pDC->LineTo(200,100)是从当前位置到绘制直线到点(200,100)，该例的（默认的）当前位置为（0，0），点(200,100)在距离左边界 200，上边界 100 处。

注意：文档中的白色窗口默认坐标是左上角为原点（0，0），从原点出发横轴正方向向右，纵轴正方向向下。

图 1-7　在 OnDraw 函数中添加语句绘制一条直线

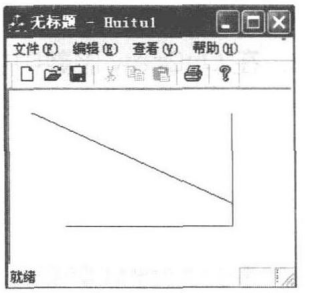


图 1-8　绘制多条直线

【例 1-2】　绘制多条直线。

修改例 1-1 程序，在 OnDraw 函数中加入下面语句（其他都不改变），

```
pDC->MoveTo(20,20);
pDC->LineTo(200,100);
pDC->MoveTo(50,120);
pDC->LineTo(200,120);
pDC->LineTo(200,20);
```

那么运行后结果如图 1-8 所示，一共绘制了三条线段，两次移动光标位置，第三条线段绘制时没有移动光标，是从点（200，120）开始绘制。

MoveTo(20,20)是把当前焦点移到点(20,20)，所以语句 pDC->LineTo(200,100)是从点(20,20)画线到点(200,100)，其他类推。

使用 CDC 类除了可以绘制直线以及移动焦点外，还可以绘制椭圆等，参考下面例题 1-3。

【例 1-3】　在 OnDraw 函数中加入语句绘制椭圆。

其他步骤不需要改变，修改 OnDraw 如下：

```
void CFile1View::OnDraw(CDC* pDC)
{
    CFile1Doc* pDoc = GetDocument();
    ASSERT_VALID(pDoc);
    pDC->Ellipse(30,20,300,200);
}
```

在 OnDraw 函数中加入了绘制椭圆的语句，运行结果如图 1-9 所示。

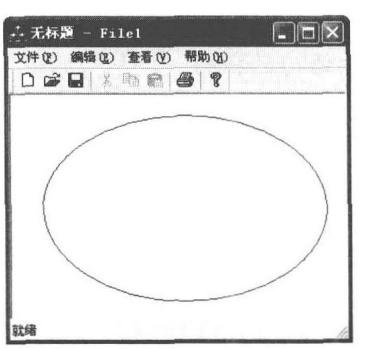


图 1-9　绘制椭圆

语句 pDC->Ellipse(30,20,300,200)中，“30,20”为椭圆的外接矩形左上顶点坐标，300 为横轴长，200 为纵轴长。

【例 1-4】　使用画点函数绘制图形。

在例 1-3 的基础上，在 OnDraw 函数中加入语句，如

下所示。

```
void CHuitu1View::OnDraw(CDC* pDC)
{
    CHuitu1Doc* pDoc = GetDocument();
    ASSERT_VALID(pDoc);
    for(int i=30;i<250;i++)
        for(int j=20;j<180;j++)
        {
            pDC->SetPixel(i,j,RGB(i,j,0));
        }
    pDC->Ellipse(30,20,250,180);
}
```

该程序绘制的图形如图 1-10 所示。先用画点函数绘制一个彩色矩形，然后又绘制一个线条椭圆，虽然是线条椭圆，但是也把彩色矩形的中间覆盖了。

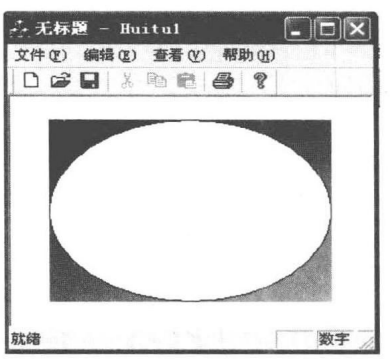


图 1-10　使用画点函数绘制图形

函数 SetPixel(i,j,RGB(i,j,0))是在(i,j)位置绘制一个点，颜色是 RGB(i,j,0)。颜色函数 RGB(r,g,b)是用红、绿、蓝三种分量成分组合方式表示彩色，每个分量的取值范围是 0～255，例如，r 是 255 表示最红的红色成分，随着 r 的减少红色成分依次减少。绿色与蓝色成分也是如此。

RGB(255,255,255)是白色，RGB(0,0,0)是黑色。

画点函数是最基本的绘图函数，实际上，使用画点函数可以在计算机上绘制任何复杂的图形。

【例 1-5】　在绘制正弦曲线的基础上制作动画。

在 OnDraw 函数中加入下面语句，就能够绘制正弦曲线。

```
for(int i=0;i<628;i++)
{
    int y=100*sin(float(i)/100);
    pDC->SetPixel(i,y+120,0);
}
```

在 OnDraw 函数中加入下面语句，运行后就可以出现动画效果，每隔 10 毫秒绘制一点，所以出现动画效果。一共绘制 628 个点，逐渐构成了一段（离散的）曲线。

```
for(int i=0;i<628;i++)
{
    int y=100*sin(float(i)/100);
    pDC->SetPixel(i,y+120,0);
    Sleep(10);                          //休眠 10 毫秒,继续执行程序
}
```

函数 SetPixel(i,y+120,0)绘制一点，颜色为黑色。函数 SetPixel 中，也可以使用一个长整型数表示颜色，0 为黑色。为了分析用一个无符号整数表示颜色的方法，设计了例 1-6。

【例 1-6】 利用给定的颜色绘点，分析使用一个整数表示颜色的方法。

填写 OnDraw 函数如下：

```
void CA1View::OnDraw(CDC* pDC)
{
    CA1Doc* pDoc = GetDocument();
    ASSERT_VALID(pDoc);

    for(int i=1;i<100;i++)
        for(int j=1;j<100;j++)
        {
            pDC->SetPixel(i,j,255);
        }
}
```

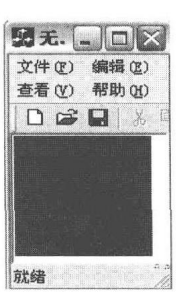
该项目（工程）运行后绘制的图形如图 1-11（a）所示。

把下面程序段嵌入到 OnDraw 函数中，绘制的图形如图 1-11（b）所示。

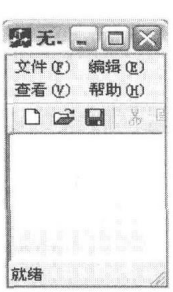
```
for(int i=1;i<100;i++)
        for(int j=1;j<100;j++)
        {
            pDC->SetPixel(i,j,16777215-j);//颜色随着 j 变化，与 i 无关
        }
```

把下面程序段嵌入到 OnDraw 函数中，绘制的图形如图 1-11（c）所示。

```
for(int i=1;i<100;i++)
        for(int j=1;j<100;j++)
        {
            pDC->SetPixel(i,j,16770000+i*j);
        }
```



(a) 使用红色绘图



(b) 上下颜色渐变

(c) 从左上到右下颜色渐变

图 1-11 使用给定的值绘制各种颜色

16777215 变为 16777216 后则颜色变黑。16777215 是 2 的 24 次方减 1，即 $2^{24}-1$。

1.1.2　绘制具有真实感的三维图形

【例 1-7】　绘制一个具有真实感的球。

首先，像例 1-1 一样，使用MFC AppWizard (exe)建立一个单文档程序 My，然后打开 Source Files 文件夹，打开 View 文件（名字为项目名加上 View，即 MyView.cpp），然后在函数 void CMyView::OnDraw(CDC* pDC)的前面自定义一个函数如下。

```
void paintBall(int x,int y,int r,CDC *pDC)
{
    double w,u,d; int R=0,G=0,B=0;
    w=r;d=w/255;
    while(w>=0)      //循环嵌套,里层循环是绘制一个圆,外层循环是绘制半径不同的圆
    {
        for(u=0;u<=628;u=u+1)          //使用参数方程绘圆
        {                              //x 与 y 是球心(圆心)位置,u/100 约等于 2π
    pDC->SetPixel((int)(x+w*cos(u/100)),(int)(y+(w*sin(u/100))),
    RGB(R,G,B));
        }
        R++;G++;B++;                   //每次颜色值分量增加,点变亮
        w=w-d;                         //每次半径变小,圆逐渐向里面绘制
    }
}
```

之后在 OnDraw 函数中加入调用语句，如下所示：

```
void CMyView::OnDraw(CDC* pDC)
{
    CMyDoc* pDoc = GetDocument();
    ASSERT_VALID(pDoc);
    paintBall(255,255,90,pDC);
}
```

最后，不要忘记在该 MyView.cpp 文件的前面加入数学函数头文件，即把语句#include "math.h"写在该文件的前部（那里还引入了一些头文件）。

项目运行结果如图 1-12 所示。

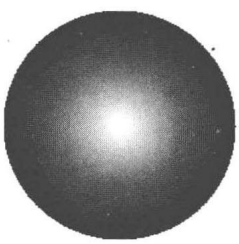


图 1-12　使用画点函数绘制真实感的球

图 1-12 中的球具有真实感，是因为其具有明暗效果（和人工素描类似），模拟光线是从正前方照射过来。

如果光线是从正上方照射下来，如何编写程序绘制一个具有真实感的球呢？

【思考题】　如果光线是从其他方向照射过来，如何绘制真实感的球？

真实感图形绘制是计算机图形学的重点，模拟真实的世

界，是计算机图形学追求的主要目标之一。

三维空间中图形的存储、表示以及显示在二维平面上，既是计算机图形学的重点也是难点。除了使用图 1-12 使用亮度显示真实感三维图形外，少数时候也使用如图 1-13 所示的线框图显示三维物体。

【例 1-8】 绘制三维螺旋线在三个坐标平面上的投影。

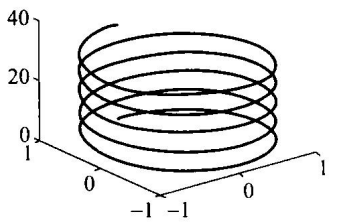


图 1-13 一个三维空间中的螺旋线

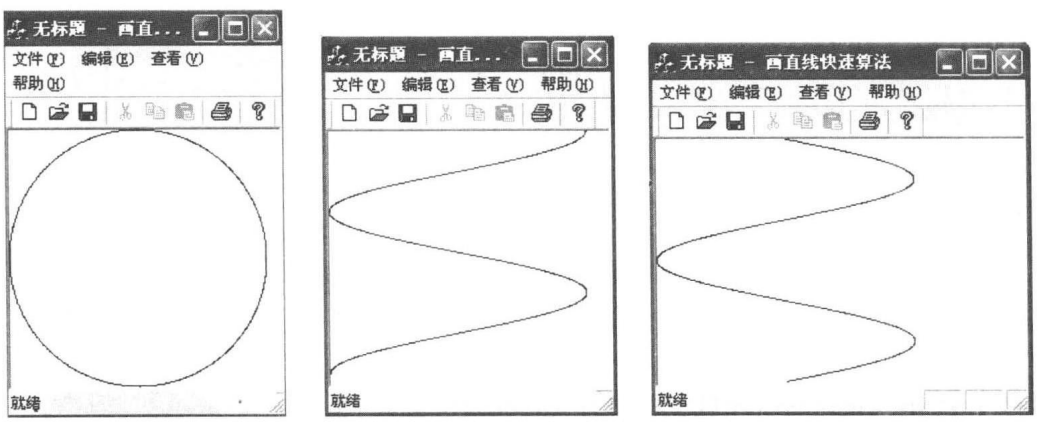
图 1-13 所示图形的代数方程如下：

$$\begin{cases} x = \sin t \\ y = \cos t \quad 0 \leqslant t \leqslant \pi \\ z = t \end{cases} \tag{1-1}$$

下面程序能够显示出该螺旋线在三个坐标平面上的投影图（这种投影称为正投影）。

```
double x,y,z,t;
for(t=0;t<=9.42;t=t+0.01)
{
    x=sin(t)*100+100;
    y=cos(t)*100+100;
    z=t*20;
    pDC->SetPixel((int)x,(int)(y),0);
    //pDC->SetPixel((int)x,z,0);
    //pDC->SetPixel((int)y,z,0);
}
```

该程序运行结果如图 1-14 所示。



(a) XOY面投影　(b) XOZ面投影　(c) YOZ面投影

图 1-14 螺旋线在三个坐标平面上的投影图

函数 SetPixel((int)x,(int)(y),0)的参数 0 是表示颜色为黑色。该参数用一个 $0 \sim 2^{24}-1$ 的整数表示各种颜色。

【思考题】 这个三维空间的螺旋线，投影在三个坐标平面上，没有真实感，那么如何绘制投影在其他斜平面上的投影图呢？即如何使用 Visual C++的画点函数绘制一个具