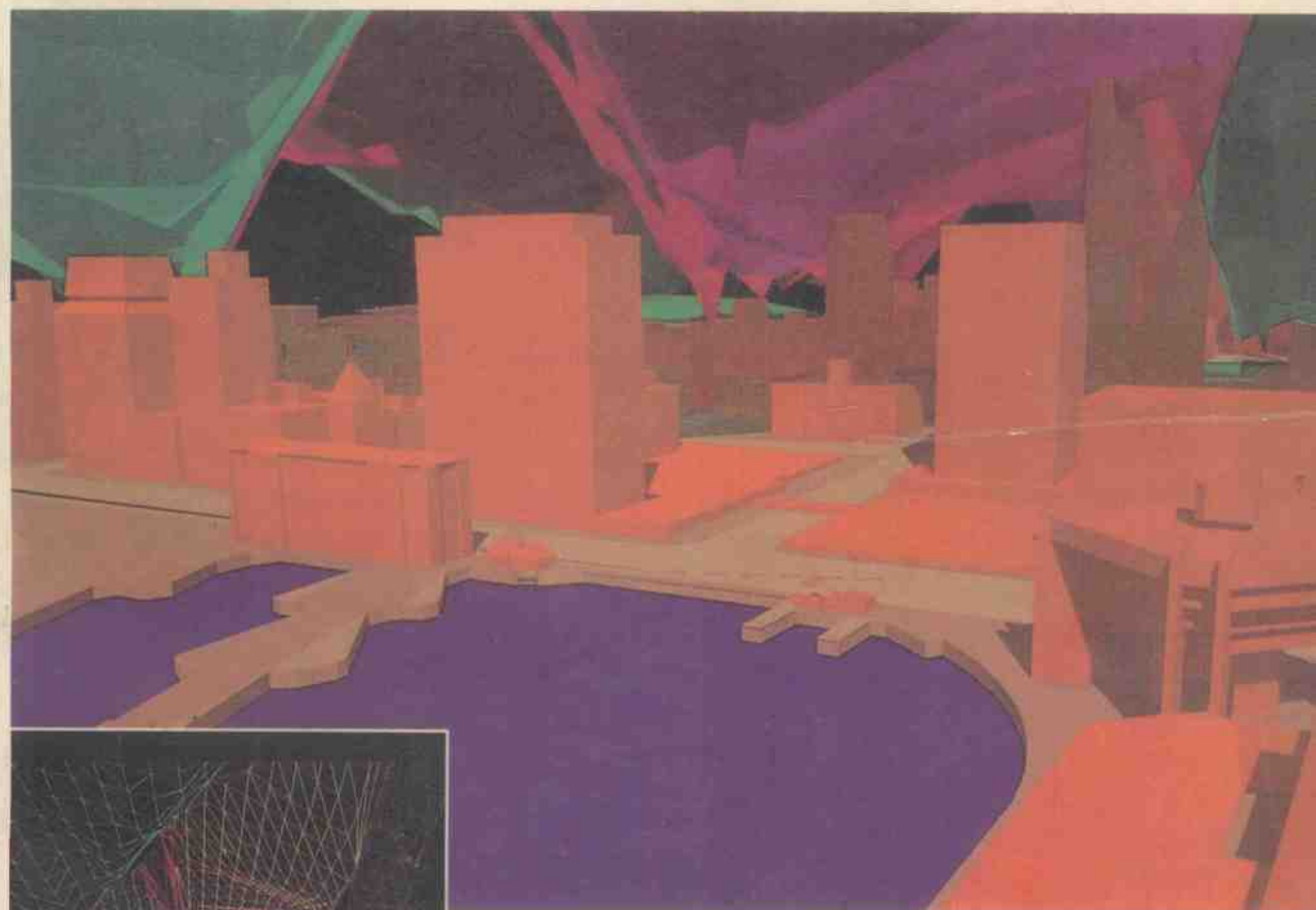
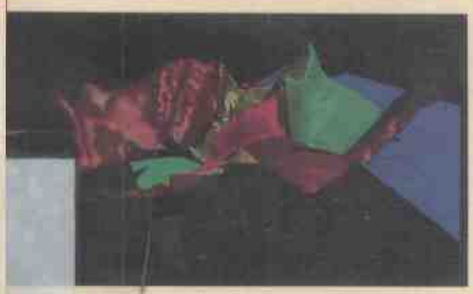
北京希望电脑公司计算机技术丛书

真实感三维图形程序设计实例

朱 聆 鞠玉兰 徐 曦 著





学苑出版社

北京希望电脑公司计算机

真实感三维图形程序设计实例

朱 聆
鞠玉兰 著
徐 曦
熊可宜 校

学苑出版社
1993·10月

(京)新登字 151 号

内 容 提 要

本书分六章围绕着几千行 C 语言的源程序，介绍了分数维、三维实体、光照模型、高级动画、图形减色等几种在三维图形设计中的核心知识，具有很强的实用性和启发性，即可作为学习《计算机图形学》一课的教材及参考资料，也可作为各种 CAD 应用软件的开发指南。

欲购本书的用户，请与北京 8721 信箱联系，电话 2562328，邮编 100080。

真实感三维图形程序设计实例

著　　者：朱　聆　鞠玉兰　徐　曦
校　　对：熊可宜
责任编辑：徐建军
出版发行：学苑出版社　　邮政编码：100032
社　　址：北京市西城区成方街 33 号
印　　刷：北京艺辉胶印厂印刷
开　　本：787×1092 1/16
印　　张：33.75　　字数：815 千字
印　　数：1—5000 册
版　　次：1993 年 10 月北京第 1 版第 1 次
ISBN 7-5077-0760—1/TP・7
定　　价：49.00 元(含盘)

学苑版图书印、装错误可随时退换

前 言

近几年来，随着三维实体造型设计，三维动画设计及多媒体技术的发展，计算机图形学进入了一个空前繁荣的崭新阶段。当国内用户在 3DSTUDIO，AUTOCAD 等国外三维设计软件的强大功能面前惊诧不已时，国内对于这一计算机技术前沿领域的研究还处于理论探讨阶段，有关的技术资料也只是零散地分布在各专业期刊中，尤其是不能以一个具体的程序实例的形式形象地说明抽象的原理，往往使人感到可望而不可即。

本书正是针对这方面的不足，围绕着几千行 C 语言源程序，介绍了分数维，三维实体，光强模型，高级动画，图形减色等几种在三维图形设计中的核心内容，具有很强的实用性和启发性；既可以作为大学高年级本科生或研究生学习《计算机图形学》一课的教材或参考资料，也可以作为各类 CAD 应用软件的开发指南。

全书共分六章，由朱聆，徐曦执笔。书中程序均经过上机调试，并存储在随书发行的软盘中（5 寸高密盘一张）

本书在出版过程中得到了希望公司资料部各位老师的大力帮助，这里谨向秦人华经理和其他老师表示深深的谢意。

目　录

第一章　基本库函数

1.1　全局头文件

1.1.1　头文件 DEF.H

本书所有程序实例的基本类型定义。

(1) 几种常用的数据类型

▲ 字节

```
typedef unsigned char       Byte;
```

▲ 字

```
typedef unsigned int        Word;
```

▲ 双字

```
typedef unsigned long       DWord;
```

▲ 布尔型

```
typedef enum {false, true} Boolean;
```

(2) 调色板

▲ RGB 模型结构

```
typedef struct
{
  Byte Red;
  Byte Grn;
  Byte Blu;
}
RGB;
```

▲ 256 色调色板

```
typedef RGB Palette Register[256];
```

(3) 三维坐标

▲ 浮点坐标

```
typedef float TDA[3];
```

▲ 整数坐标

```
typedef int   TDIA[3];
```

▲ 浮点向量坐标

```
typedef float FDA[4];
```

▲ 浮点矩阵坐标

```
typedef float Matx4x4[4][4];
```

(4) 常数

▲ 坐标

```
#define MaxCol    7
#define MaxInten  35
```

▲ 数学

```
#define Ln10 2.30258509299405E+000
#define OneOverLn10 0.43429448190325E+000
#define Pi 3.1415927
#define PiOver180 1.74532925199433E-002
#define PiUnder180 5.72957795130823E+001
```

1.1.2 头文件 GLOBAL.H

本书所有程序实例的全局变量定义。

(1) 坐标参数

```
int  XRes, YRes;
Word MaxXRes, MaxYRes;
Word MaxX, MaxY;
```

(2) 屏幕横纵比率

```
float Asp;
```

(3) 正交坐标标志位

```
Boolean PerspectivePlot;
```

(4) 三维坐标

```
float   Mx, My, Mz, ds;
```

(5) 坐标轴及调色板标志位

```
Boolean Draw Axis And Palette;
```

(6) 视角和观察面倾角

```
int   Angl, Tilt;
```

1.1.3 头文件 MATHB.H

数学函数库的函数原型。

(1) 基本数值函数

▲ 取整数部分（四舍五入法）

```
extern int Round(double x);
```

▲ 取小数部分

```
extern float Frac(double x);
```

▲ 取整数部分（截断法）

```
extern int Trunc(double x);
```

▲ 取实数平方

```
extern float SqrFP(float x);
```

▲ 取整数平方

```
extern int Sqr(int x);
```

(2) 角度,弧度函数

▲ 角度变弧度

extern float Radians(float Angle);

▲ 弧度变角度

extern float Degrees(float Angle);

(3) 算术函数

▲ 取余弦

extern float CosD(float Angle);

▲ 取正弦

extern float SinD(float Angle);

▲ 取 A 的 N 次幂 (实数)

extern float Power(float Base, int Exponent);

▲ 取常用对数

extern float Log(float x);

▲ 取 10 的 N 次幂

extern float Exp10(float x);

▲ 符号判别 (实数)

extern float Sign(float x);

▲ 符号判别 (整数)

extern int IntSign(int x);

▲ 取平方根

extern int IntSqrt(int x);

▲ 取 A 的 N 次幂 (整数)

extern int IntPower(int Base, int Exponent);

(4) 比较函数

▲ 取两数小者

extern float MIN(float a, float b);

▲ 取两数大者

extern float MAX(float a, float b);

▲ 取三数小者

extern float MIN3(float a, float b, float c);

▲ 取三数大者

extern float MAX3(float a, float b, float c);

▲ 取四数小者

extern float MIN4(float a, float b, float c, float d);

▲ 取四数大者

extern float MAX4(float a, float b, float c, float d);

(5) 向量函数

▲ 生成向量结构 TDA (实数)

```
extern void Vec(float r, float s, float t, TDA A);
```
▲ 生成向量结构 TDA (整数)
```
extern void VecInt(int r, int s, int t, TDIA A);
```
▲ 取向量元素 (实数)
```
extern void UnVec(TDA A, float *r, float *s, float *t);
```
▲ 取向量元素 (整数)
```
extern void UnVecInt(TDIA A, int *r, int *s, int *t);
```
▲ 向量点乘
```
extern float VecDot(TDA A, TDA B);
```
▲ 向量叉乘
```
extern void VecCross(TDA A, TDA B, TDA C);
```
▲ 向量长度
```
extern float VecLen(TDA A);
```
▲ 向量正规化
```
extern void VecNormalize(TDA A);
```
▲ 向量乘矩阵
```
extern void VecMatxMult(FDA A, Matx4x4 Matrix, FDA B);
```
▲ 向量相减 (实数)
```
extern void VecSub(TDA A, TDA B, TDA C);
```
▲ 向量相减 (整数)
```
extern void VecSubInt(TDIA A, TDIA B, TDA C);
```
▲ 向量相加 (两个)
```
extern void VecAdd(TDA A, TDA B, TDA C);
```
▲ 向量相加 (三个)
```
extern void VecAdd3(TDA A, TDA B, TDA C, TDA D);
```
▲ 向量拷贝 (实数)
```
extern void VecCopy(TDA A, TDA B);
```
▲ 向量拷贝 (整数)
```
extern void VecCopyInt(TDIA A, TDIA B);
```
▲ 向量线性组合
```
extern void VecLinComb(float r, TDA A, float s, TDA B, TDA C);
```
▲ 向量相乘 (实数)
```
extern void VecScalMult(float r, TDA A, TDA B);
```
▲ 向量相乘 (整数)
```
extern void VecScalMultI(float r, TDIA A, TDA B);
```
▲ 向量相乘 (整数,四舍五入)
```
extern void VecScalMultInt(float r, TDA A, TDIA B);
```
▲ 向量加乘
```
extern void VecAddScalMult(float r, TDA A, TDA B, TDA C);
```

▲ 向量清零 (实数)

```
extern void VecNull(TDA A);
```

▲ 向量清零 (整数)

```
extern void VecNullInt(TDIA A);
```

▲ 向量数乘

```
extern void VecElemMult(float r, TDA A, TDA B, TDA C);
```

▲ 向量取负

```
extern void VecNegate(TDA A);
```

▲ 向量取小

```
extern void VecMin(TDA a, TDA b, TDA c);
```

▲ 向量取大

```
extern void VecMax(TDA a, TDA b, TDA c);
```

(6) 仿射变换函数

▲ 矩阵清零

```
extern void ZeroMatrix(Matx4x4 A);
```

▲ 平移变换

```
extern void Translate3D(float tx, float ty, float tz, Matx4x4 A);
```

▲ 放缩变换

```
extern void Scale3D(float sx, float sy, float sz, Matx4x4 A);
```

▲ 旋转变换

```
extern void Rotate3D(int m, float Theta, Matx4x4 A);
```

▲ 矩阵相乘

```
extern void Multiply3DMatricies(Matx4x4 A, Matx4x4 B, Matx4x4 C);
```

▲ 矩阵拷贝

```
extern void MatCopy(Matx4x4 a, Matx4x4 b);
```

▲ 仿射变换

```
extern void PrepareMatrix(float Tx, float Ty, float Tz,
                          float Sx, float Sy, float Sz,
                          float Rx, float Ry, float Rz,
                          Matx4x4 XForm);
```

▲ 仿射逆变换

```
extern void PrepareInvMatrix(float Tx, float Ty, float Tz,
                             float Sx, float Sy, float Sz,
                             float Rx, float Ry, float Rz,
                             Matx4x4 XForm);
```

▲ 向量乘矩阵

```
extern void Transform(TDA A, Matx4x4 M, TDA B);
```

(7) 随机数函数

▲ 随机数置初值

```
extern void InitRand(float Seed);
```
▲ 产生随机数 (整数)
```
extern int RandInt(Word Range);
```
▲ 产生随机数 (实数)
```
extern float Rand();
```

1.1.4 头文件 GRAPHB.H

图形函数库的函数原型。

(1) 图形模式设置

▲ 设置图形模式 (TVGA)
```
extern void Set_Mode(int Mode);
```
▲ 初始化地址数组缓冲区
```
extern void Pre_Calc();
```
▲ 初始化图形模块
```
extern void Init_Graphics();
```
▲ 设置图形窗口大小
```
extern void Set_Graphics_Mode(Word xRes, Word yRes);
```
▲ 等待按键
```
extern void Wait_For_Key();
```
▲ 退出图形模块
```
extern void Exit_Graphics();
```
▲ 文本方式设置标题
```
extern void Title();
```
▲ 数据交换
```
extern void Swap(int * first, int * second);
```
(2) 调色板设置

▲ 设置调色板
```
extern void Set_Palette(Palette Register Hue);
```
▲ 初始化调色板 (64 级灰度, RGB 3 色)
```
extern void Init_Palette(Palette Register Color);
```
▲ 初始化调色板 (7 色, 35 种层次)
```
extern void Init_Palette_2(Palette Register Color);
```
▲ 循环设置 256 种调色板
```
extern void Cycle_Palette(Palette Register Hue);
```
(3) 基本图元

▲ 写象素
```
extern void Plot(Word x, Word y, Byte color);
```
▲ 画圆
```
extern void Circle(Word x, Word y, Word radius, Byte color);
```

▲ 画直线

extern void Draw(int xx1, int yy1, int xx2, int yy2, Byte color);

▲ 画三维直线

extern void Draw_Line_3D(TDA Pnt1, TDA Pnt2, Byte Color);

▲ 设置象素灰度

extern void Put_Pixel(int x, int y, Byte Color, Byte Intensity);

▲ 读象素（64 级灰度，RGB 3 色）

extern Byte Get_Pixel(Word x, Word y);

▲ 读象素（7 色，35 种层次）

extern Byte Get_Pixel_2(Word x, Word y);

▲ 设置坐标系和调色板

extern void Put_Axis_And_Palette(Boolean PlaceOnScreen);

▲ 显示坐标系

extern void Display_Axis();

▲ 显示调色板

extern void Display_Palette();

▲ 显示坐标系和调色板

extern void Axis_And_Palette();

(4) 设置坐标系

▲ 初始化三维坐标系

extern void Init_Plotting(int Ang, int Tlt);

▲ 初始化三维用户视窗

extern void Init_Perspective(Boolean Perspective, float x, float y, float z, float m);

▲ 三维坐标影射二维平面

extern void Map_Coordinates(float X, float Y, float Z, int *Xp, int *Yp);

▲ 笛卡尔坐标系画点

extern void Cartesian_Plot_3D(float X, float Y, float Z, Byte Color);

▲ 柱面坐标系画点

extern void Cylindrical_Plot_3D(float Rho, float Theta, float Z, Byte Color);

▲ 球面坐标系画点

extern void Spherical_Plot_3D(float R, float Theta, float Phi, Byte Color);

1.2 模块编译

头文件模块可以单独编译，这样可不必在修改主程序时每次都重新编译一次，节省了不少时间. 下面介绍在 TURBO C 编译器环境下使用 PROJECT 功能的具体方法：

(1) ALT - C 键编译头文件模块，生成 .OBJ 文件

(2) ALT - P 键进入 PROJECT 功能，定义 PROJECT 文件名

(3) ALT - A 键加入相应的 .OBJ 目标代码

(4) ALT – C L 键连接所有 .OBJ 目标文件执行.

注意: 所有源程序必须在大模式下编译.

1.3 数学模块库

1.3.1 基本概念和算法

(1) 向量正规化

又称向量的单位化，即求出与原向量方向相同，但长度为 1 的向量。VecNormalize 实现向量正规化，在三维光线相交时计算单位向量。

(2) 向量线性组合

公式为 C = rA+sB 其中 A，B 为向量，r, s 为系数，这是一个通用向量运算公式，如:

向量和 VecAdd = VecLinComb(1.0,A,1.0,B,C)

向量差 VecSub = VecLinComb(1.0,A,–1.0,B,C)

(3) 矩阵变换一般形式

$$\begin{bmatrix} b0 \\ b1 \\ b2 \\ b3 \end{bmatrix} = \begin{bmatrix} m00 & m01 & m02 & m03 \\ m10 & m11 & m12 & m13 \\ m20 & m21 & m22 & m23 \\ m30 & m31 & m32 & m33 \end{bmatrix} \quad \begin{bmatrix} a0 \\ a1 \\ a2 \\ a3 \end{bmatrix}$$

其中(mij)为变换矩阵，(az)为输入向量，(bk)为输出向量。

(4) 仿射变换

▲ 平移变换

矩阵:

$$T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -tx & 1 & 0 & 0 \\ -ty & 0 & 1 & 0 \\ -tz & 0 & 0 & 1 \end{bmatrix}$$

其中(tx,ty,tz)的平移坐标

▲ 放缩变换

矩阵:

$$S = \begin{bmatrix} sx & 0 & 0 & 0 \\ 0 & sy & 0 & 0 \\ 0 & 0 & sz & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

其中(sx,sy,sz)为放缩因子，

▲ 旋转变换

矩阵 (又分为沿 X，Y，Z 轴旋转三种情况)。

X 轴:

$$Rx = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos\Theta & sin\Theta & 0 \\ 0 & -sin\Theta & cos\Theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Y 轴:

$$Ry = \begin{bmatrix} cos\Theta & sin\Theta & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -sin\Theta & cos\Theta & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Z 轴:

$$Rz = \begin{bmatrix} cos\Theta & sin\Theta & 0 & 0 \\ -sin\Theta & cos\Theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

其中 Θ 为旋转角度。

(5) 三维坐标系

▲ TDA

三维浮点数组，其内容为(X，Y，Z)三维坐标，即以三维坐标系三个单位向量 i,j,k,i=(1,0,0),j=(0,1,0),z=(0,0,1)为单位的向量表示: V=xi+yj+zr。

▲ FDA

四维浮点数组，与 TDA 类似，其中第四个元素为常数 0.0 或 1.0，用于向量转置运算。

▲ Matx4×4

二维浮点数组，表示 4×4 的变换矩阵。

(6) 向量点乘 (结果为实数)

公式为 VectDot (A,B)=Veclen(A) * Veclen(B) * cos(theta)

其中 Veclen(A)为向量A的长度，

Veclen(B)为向量B的长度，

theta 为A，B之间夹角。

性质:

▲ 若 VecDot = 0，则 A，B 两向量相互垂直。

▲ 若 VecDot = Veclen(A) * Veclen(B)，
则 Cos(theta) = 1,表示 A，B 向量同向。

▲ 若 VecDot = Veclen(A) * Veclen(B)，
则 cos(theta) = −1，表示 A；B 向量反向。

用途：判断物体切平面之间的位置关系。

(7) 向量叉乘 (结果为向量)

公式：VecCross(A,B) = ABsin (theta)

方向：与 A，B 两向量都垂直。

用途：确定物体的法向量平面。

一般公式：

设向量 A，B 在 X，Y，Z 三轴上的分量分别为 A1，A2，A3 和 B1，B2，B3。

则：VecCross(A,B)的长度为 = |A2B3−A3B2，A3B1−B1B3,A1B2−A2B1|。

(8) 向量长度 (模)

公式：Veclen(V) = sqrt(x * x+y * y+z * z)

用途：求三维空间两点间距离。

(9) 伪随机数

伪随机数返回一个值在 0.0 到 1.0 之间的无规律的实数序列。首先要给出一个种子数，再乘上一个常数 Sigma，代入函数运算，得到一个新随机数，这个数又做为种子数产生下一个随机数，……，这样可通过递归运算取得一串无规律的数。

1.3.2 数学库函数

源代码清单如下：

```
( MATHB.C )
/ * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
  *                                                                   *
  *          Mathematical Functions                                   *
  *                                                                   *
  * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

   Radians      - converts degrees to radians
   Degrees      - converts radians to degrees
   CosD         - cosine in degrees
   SinD         - sine in degrees
   Power        - power a^n
   Log          - log base 10
   Exp10        - exp base 10
```

```
    Sign        - negative=-1  positive=1  null=0
    IntSign     - negative=-1  positive=1  null=0
    IntSqrt     - integer square root
    IntPower    - integer power a/\n
 * /

#include "stdio.h"
#include "math.h"
#include "defs.h"
#include "mathb.h"

int Round(double x)
{
  return((int)(x+0.5));
}

int Trunc(double x)
{
  return((int)(x));
}

float Frac(double x)
{
  int y;

  y=((int)(x));
  return(x-(float)y);
}

float SqrFP(float x)
{
  return(x * x);
}

int Sqr(int x)
{
  return(x * x);
}
```

```
float Radians(float Angle)
{
  return(Angle * PiOver180);
}

float Degrees(float Angle)
{
  return(Angle * PiUnder180);
}

float CosD(float Angle)
{
  return(cos(Radians(Angle)));
}

float SinD(float Angle)
{
  return(sin(Radians(Angle)));
}

float Power(float Base, int Exponent)
{
  float BPower;
  int   t;

  if(Exponent==0)
    return(1);
  else
  {
    BPower=1.0;
    for(t=1; t<=Exponent; t++)
    {
      BPower *= Base;
    }
    return(BPower);
  }
}

float Log(float x)
```