Chap1~2 导论 MFC基础

计算机图形学是一种使用图形生成原理和算法将二维或三维图形转化为计算机光栅化显示的学科。

图像处理是对基于点阵的图像进行增强、去噪、复原、分割、重建、编码、存储、压缩和恢复等不同处理方法的学科。

模式识别是把点阵图像进行特征抽取，然后用统计学方法给出图形描述的学科。

扫描线： 光栅显示器为了在能整个屏幕上显示出图形，电子束需要从屏幕的左上角开始，沿着水平方向从左至右匀速地扫描，一直扫描到屏幕的右下角，显示出一帧图像。

荫罩板：会拦下任何散乱的电子束，以免其轰击错误的荧光点

帧缓冲器（frame buffer）是显存内用于存储一帧图像的一片连续内存空间。光栅扫描显示器使用帧缓冲存储屏幕上每个像素的颜色信息，帧缓冲使用位面（bit plane）与屏幕像素一一对应。

显示控制器: 光栅扫描显示器的显示控制器反复扫描帧缓冲，读出像素的位置坐标和颜色值送给相应的地址寄存器，并经过数模转换后翻译为模拟信号。

图形软件标准最初是为提高软件的可移植性而提出的

CDC类：定义了设备上下文对象的基类，封装了绘图所需的所有函数

|  |  |
| --- | --- |
| CClientDC | 显示器客户去设备上下文类 只能在窗口的客户区（不包括边框、标题栏、菜单栏以及状态栏的空白区域）进行绘图。 |
| CMetaFileDC | Windows图元文件设备上下文类。 CMetaFileDC封装了在Windows中绘制图元文件的方法。Wmf格式 |
| CPaintDC | 只在响应WM\_PAINT消息时使用。 |
| CWindowDC | 整个屏幕区域的显示器设备上下文类。包括客户区（工具栏、状态栏和视图窗口的客户区）和非客户区（标题栏和菜单栏）。 |

绘图工具

|  |  |
| --- | --- |
| CGdiObject类 | GDI绘图工具的基类，一般不能直接使用。 |
| CBitmap | 封装了一个GDI位图，提供位图操作的接口。 |
| CBrush类 | 封装了GDI画刷，可以选作设备上下文的当前画刷。画刷用于填充图形内部 |
| CFont | 封装了GDI字体，可以选作设备上下文中的当前字体。 |
| CPallette | 封装了GDI调色板，提供应用程序和显示器之间的颜色接口。 |
| CPen | 封装了GDI画笔，可以选作设备上下文的当前画笔。画笔用于绘制图形边界线。 |

Chap3 基本图形的扫描转换

基本图形的光栅化就是在像素点阵中确定最佳逼近于理想图形的像素点集，并用指定颜色显示这些像素点集的过程

直线的中点Bresenham算法

根据P0，P1 求斜率k 判断向哪个方向迈步（从左往右绘制）

主方向每次+1，另一个方向求中点误差项d

F(x,y) = y - kx - d

写出初始值d0 F（主方向+1，此方向+0.5）

写出di 的计算公式

写出次方向迈步的条件，di<0不变 di>0加1 若d==0将其划为下，左的点

写出d<0 、d>0的递推公式，依次写出所有点

写出结束条件（主方向到P1点）

对于|k|=∞的直线 直接绘制

圆的扫描转换

C:\Users\15624\AppData\Local\Packages\Microsoft.Office.OneNote_8wekyb3d8bbwe\TempState\msohtmlclipclip_image001.png

当经过 x= y 或 y = -x 时，改变迈步的方向

椭圆扫描转换

F (x, y) 102x2 + a2y2 a2b2 

反走样技术

参数化表示 c = (1-t)c0 + tc1

0<k<1的示例 白色（255，255，255）

CPoint p0(-100,-50),p1(200,50),p;

int dx,dy;

dx=p1.x-p0.x;

dy=p1.y-p0.y;

double k=(double)dy/dx,e;

for(p=p0,e=0;p.x<p1.x;p.x++)//不包括终点p1

{

pDC->SetPixelV(p.x, p.y,RGB(e\*255,e\*255,e\*255));

pDC->SetPixelV(p.x, p.y+1,RGB((1-e)\*255,

(1-e)\*255,(1-e)\*255));

e+=k;

if(e>=1.0)

{

p.y++;

e--;

}

}

Chap4 多边形填充

多边形定义：由折线段组成的封闭图形

使多边形的内部区域在左右边

多边形表示

顶点表示法：直观、占内存少，易于几何变换

点阵表示法：运用与帧缓冲

多边形着色模式

平面着色模式：单一色

光滑着色模式：各顶点颜色不同，内部颜色有双线性差值得到

填充多边形

左闭右开，上闭下开

有效边表填充算法

通过维护边表和有效边表，避免直线求交运算，按照扫描线从小到大的移动顺序，计算当前扫描线与有效边的交点进行排列、配对，最后完成填充

特殊点处理原则

起点保留、终点忽略，扫面先与多边形交点先四舍五入，之后留左舍右

有效边： 多边形与当前扫描线相交的边

有效边表AET：把有效边按照与扫描线交点x坐标递增的顺序存放到一个链表中

x 
next 

桶表与边表ET

ymax 
next 

边 表 
12 
0 
． 02 

边缘填充算法

先求出交点，交点右侧颜色全部取反，按任意顺序处理完所有边，则填充完毕

0 的 0 思 到 00 的 到 00 ， 
0 0 ` 过 0000000 
000 到 思 烈 000 0007 ， ， 000 0 过 0 ， ， ， 蟲 000 
的 到 的 到 的 生 河 蟲 ， 
过 过 生 生 过 生 # 0 饪 ` 
0 00000 到 000 0 ` 
00 的 0000 影 00 地 00 ` 
0 的 0 跹 000 思 麯 00 000 ` 
00 到 00 的 00 的 思 麯 0000 
00 的 0 一 0 的 0 到 的 00 到 
00 的 0 能 0 跷 0 地 00 到 
0 的 到 葫 到 到 0 到 生 00 生 
00 就 俺 能 0 生 0 生 的 0 生 0 
的 到 到 的 到 的 
能 00 地 00 的 00 
的 ， ` 的 0 的 0 的 0 的 
0 
0 地 00 影 00 能 00 ` 
" 生 生 过 总 射 总 生 
/ 生 00 的 00 的 
生 生 到 总 生 
思 0 的 0 的 0 的 0 的 
00000000 0 
0 000 00 过 0 
就 00 就 0 总 就 0 就 
能 以 0 能 能 訕 生 能 0 能 
00 ` 000 的 的 的 的 0 
0 ` 0000000 
` 000000 
0 
思 00 00 
0 00 00 
0 鼽 00 思 000 
00 石 000 000 
00 ` 00000000 
0000 慮 0000 
的 ， ` 0 地 到 地 到 地 地 
边 缘 填 充 算 法 原 理 
， 到 的 

区域填充算法

种子填充算法

四连通边界-----边界为直角

八连通边界-----可以斜着

四连通域-----左上右下

八连通域-----左、左上、上、右上、右、右下、下、左下

使用堆栈原理，从任一种子像素位置开始，按照（四连通---左上右下， 八连通……）的顺序依次扩散

扫描线种子填充算法

（1）栈顶像素出栈。

（2）沿扫描线对出栈像素的左右像素进行填充，直至遇到边界像素为止。即每出栈一个像素，就对区域内包含该像素的整个连续区间进行填充。

（3）同时记录该区间，将区间最左端像素记为xleft，最右端像素记为xright。

（4）在区间〔xleft，xright〕中检查与当前扫描线相邻的上下两条扫描线的有关像素是否全为边界像素或已填充像素，若存在非边界且未填充的像素，则把未填充区间的最右端像素取作种子像素入栈。

Chap5

齐次坐标：n+1维表示n维矢量

二维几何变换

口 co / 
对 图 形 进 行 比 例 、 旋 转 、 反 射 和 错 切 变 换 ； 
对 图 形 进 行 半 移 变 换 ； 
丆 2 一 [ 0 司 
对 图 形 进 行 投 影 变 换 ； 
对 图 形 进 行 整 体 比 例 变 换 。 

平移变换 lm

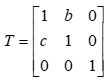
比例变换 ad

旋转变换（逆时针方向为正）

ı 
0 
0 
0 
Ü soo 
Ü uys 
0 
Ü uys 
Ü soo 

反射变换 a d 基于x轴，x不变y轴取反

错切变换



习题p137-140

二维图形剪裁

常用坐标系概念

世界坐标系指现实世界中场景固定的坐标系，既实数坐标系系统。

用户坐标系是可移动坐标系，用户坐标系的原点可以放在物体的任意位置上，坐标系也可以旋转任意角度

观察坐标系是在世界坐标系中定义的坐标系，观察坐标系原点位于视点，z轴垂直于屏幕，正向为视线方向。二维观察坐标系主要用于指定图形的输出范围。三维观察坐标系是左手系，用于生成物体的旋转动画。

观 察 坐 标 系 
三 维 观 察 坐 标 系 
屏 幕 坐 标 系 
在 
YW 
世 界 坐 标 系 

屏幕坐标系为实数域二维直角坐标系。原点位于屏幕中心，x轴水平向右为正，y轴垂直向上为正。

显示器等图形输出设备自身都带有一个二维直角坐标系称为设备坐标系。设备坐标系是整数域二维坐标系，原点位于屏幕左上角，x轴水平向右，y轴垂直向下，基本单位为像素。格化到〔0.0，0.0〕到〔1.0，1.0〕的范围内的设备坐标系称为规格化设备坐标系。

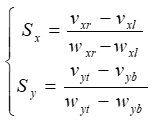
窗口与视区

窗口是在观察坐标系中定义的确定显示内容的矩阵区域，是剪裁图形的标准参照物。

视区为在屏幕坐标系中定义的输出图形的矩形区域。图形输出需要进行从窗口到视区的变换，只有窗口内的图形才能在视区中输出，且输出的形状要根据视区的大小进行调整，这称为视窗变换。

视窗变换（p145）

平移至原点 比例变换 反平移



l] 
XI 

xl 
xl 
xl 
w Nia 
W ybC 

x 
axıı, + b 

Ciohen-Sutherland直线段裁剪算法p146 or 习题解答p106

解题步骤

写出RC编码4位所代表的意义,并对9个区域进行编码

对两端点进行编码（边界、端点）

RC0|RC1=0 端点都在窗口内，简取

RC0&RC1≠0 两端点位于窗口外同一侧，简弃

若不满足以上两个条件，对窗口进行“求交”判断

按顺序（左右上下）计算窗口边界延长线与直线段的交点、

У = К(х — хо) + уо 

у- УО 
К 

选出在窗口边界的交点，剪裁

输出剪裁后线段的两端点

多边形裁剪算法

四种裁剪情况

(c) 内 一 外 ， 保 存 
PI 
(d) 外 一 外 ， 不 保 存 

每次裁剪的时候，写出输入、输出的顶点

PO 
So 
输 入 ： 
PoPIP2P3 
输 出 ： S S P P P 
0 1 1 2 3 
（ a ） 用 窗 口 左 边 界 裁 剪 
输 入 ： 
S S P P P 
0 1 1 2 3 
输 出 ： S S P S S P 
0 1 1 2 3 3 
(b) 用 窗 口 右 边 界 裁 剪 

4fiA•. 
SS PS SP 
011233 
S5S0SIPIS2S3S4 
4fiA•. 
S5SOSlPlS2S3S4 

Chap6 三维变换与投影

三维图形几何变换

f 
g 
I 
m 

平移变换

o 
0 
0 
I 
o 
0 
0 
I 
o 
0 
0 
ı 

比例变换

I 
0 
0 
0 
S 
0 
0 
0 
S 
0 
0 
0 

旋转变换（右手螺旋---逆时针为正）

0 d soo d 115— 0 
0 dup d soo 0 

反射变换

I 
0 
0 
0 
I 
0 
0 
0 
0 
I 
o 
0 
0 
0 
1 

关于xoy面的反射（xoz、yoz类似）

I 
o 
0 
0 
0 
I 
0 
0 
0 
0 
I 
o 
0 
0 
0 
1 

沿x方向错切 （y、z方向错切类似）

0 一 0 一 0 一 
0 一 0 一 0 一 
0 

三维图像复合变换 P162

平移、两次旋转与某一坐标轴重合、绕该轴旋转、一系列逆变换

坐标系变换

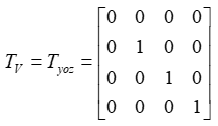
二维：坐标轴的平移旋转，但方向与点的变换相反

三维：同二维类似

投影变换

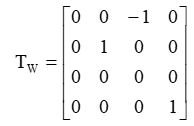
疣 视 视 等 二 三 
正 
交 
俯 侧 正 正 正 
图 图 测 测 测 
正 投 影 
正 
影 
平 行 投 影 
了 斜 等 测 
斜 投 影 乁 斜 二 测 
一 点 透 视 
二 点 透 视 
透 视 投 影 
三 点 透 视 

主视图

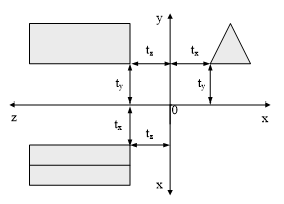


俯视图（yoz平面表示俯视图 将xoz面绕z顺时针旋转90度）

侧视图（将xoy面绕y轴逆时针旋转90度）



得到三视图后 需要将其平移到特定位置



正交投影

0 一 0 一 0 一 
0 一 0 一 0 一 0 一 

斜投影：将三维物体向投影面内作平行投影，但投影方向不垂直于投影面得到的投影称为斜投影（具有较好立体感，可测量）

斜等测：投影方向与投影面成45度，x y z 三个方向的长度相等

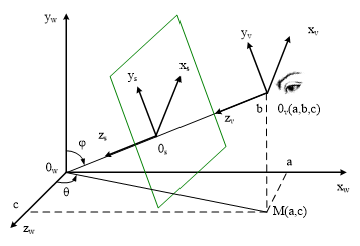
斜二测：投影方向与投影面成63.4度，z方向的长度为原长度的一半

透视投影

与平行投影相比，透视投影的特点是所有投影线都从空间一点（称为视点或投影中心）投射，离视点近的物体投影大，离视点远的物体投影小，小到极点消失，称为灭点（vanishing point）。

|  |  |
| --- | --- |
| 灭点 | 不与屏幕平行的平行投影汇聚到的点为灭点，灭点是无线远点在屏幕上的投影 |
| 主灭点 | 三维图形中有无数个灭点， 坐标轴上的灭点，主灭点最多有三个 |

|  |  |
| --- | --- |
| 一点透视 | 当φ＝90°，θ＝0°时，屏幕平行于xOz面 ，形成一个灭点，透视投影图为一点透视图。  C:\Users\15624\AppData\Local\Packages\Microsoft.Office.OneNote_8wekyb3d8bbwe\TempState\msohtmlclipclip_image032.png |
| 二点透视 | 当φ＝90°，0°＜θ＜90°时，屏幕与x轴和z轴相交，但平行于y轴， 得到二点透视图，形成两个灭点  C:\Users\15624\AppData\Local\Packages\Microsoft.Office.OneNote_8wekyb3d8bbwe\TempState\msohtmlclipclip_image033.png |
| 三点透视 | 三点透视图是屏幕与三个坐标轴都相交时的透视投影图。当φ≠0°、90°、180°；且θ≠0°、90°、180°、270°时，屏幕与x轴、y轴和z轴都相交，得到三点透视图。  C:\Users\15624\AppData\Local\Packages\Microsoft.Office.OneNote_8wekyb3d8bbwe\TempState\msohtmlclipclip_image034.png |



世界坐标系

世界坐标系OwXwYwZw采用右手直角坐标系。坐标原点位于Ow点，视点的直角坐标为Ov（a,b,c）。

观察坐标系

观察坐标系Ovxvyvzv为左手直角坐标系，坐标原点取在视点Ov上。zv轴沿着视线方向OvOw指向Ow点，视线的正右方为xv轴，视线的正上方为yv轴。

屏幕坐标系

屏幕坐标系OsXsYsZs也是左手直角坐标系，坐标原点Os位于视心。屏幕坐标系的Xs和Ys轴与观察坐标系的Xv轴和Yv轴方向一致，也就是说屏幕垂直于视线，Zs轴自然与Zv轴重合。

世界坐标系到观察坐标系的变换

将物体上点的坐标表示，从世界坐标系变换为观察坐标系。

Ow平移至Ov, 右手系变左手系（）

观察坐标系到屏幕坐标系

两坐标系z轴同向，只需按比例缩放xoy平面(d为 Ov与Os 两坐标徐原点的距离)，去掉z轴上的信息

I 
o 
0 
0 
0 
I 
0 
0 
0 
0 
o 
0 
0 
0 
Id 
0 

Chap7 自由曲线与曲面（概念）

|  |  |
| --- | --- |
| 插值 | 当用一组数据点来指定曲线的形状时，曲线精确地通过给定的数据点且形成光滑的曲线 |
| 逼近 | 当用一组控制点来指定曲线的形状时，曲线被每个控制点所吸引，但实际上并不经过这些控制点 |
| 连续性条件 | 为了保证在链接点处光滑过渡，需要满足连续性条件  参数连续性：  C0，指相邻两段曲线在结合点处具有相同的坐标。  C1，指相邻两段曲线在结合点处具有相同的一阶导数。  C2，指相邻两段曲线在结合点处具有相同的一阶导数和二阶导数。  几何连续性：  G0，指相邻两段曲线在结合点处具有相同的坐标。  G1，指相邻两段曲线在结合点处的一阶导数成比例 。  G2，指相邻两段曲线在结合点处的一阶导数和二阶导数成比例。 |

Bezier曲线

定义（公式）n表示控制点，B为Bernstein基函数

p(t) 
PB (t) 
te (0, 1) 

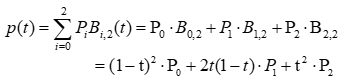
Bernstein基函数

ti (1 —ty 
Cinti (l — ty 

一次Bezier曲线（连接P0 P1的直线）

p(t) 
= PO BOI(t) + Pl • Bll(t) 
(1 -t).P0+t.Pl 

二次Bezier曲线（起点在P0，终点在P1的抛物线）



三次bizier曲线 (自由曲线 P193)

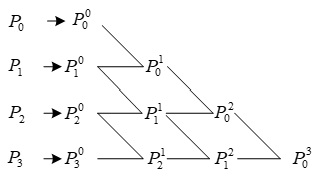
Bernstein基函数的性质

|  |  |
| --- | --- |
| 非负性 | C:\Users\15624\AppData\Local\Packages\Microsoft.Office.OneNote_8wekyb3d8bbwe\TempState\msohtmlclipclip_image041.png |
| 端点性质 | O, İ 17 |
| 权性 | te  o,  1) |
| 对称性 | cnlfl(l ty |
| 导函数 | B7At) l(t)] , (i 012 |

Bezier曲线的性质

|  |  |
| --- | --- |
| 端点性质 | t=0、1时，与P0、Pn点重合 |
| 一阶导数 | 在起点和终点的切线方向位于控制多边形的起始边和终止边的切线方向  (l üd- üd)u (ı) d  (Od- Id)u (O) d |
| 二阶导数 | C:\Users\15624\AppData\Local\Packages\Microsoft.Office.OneNote_8wekyb3d8bbwe\TempState\msohtmlclipclip_image047.png  起始点和终止点的二阶导数分别取决于最开始的3个控制点和最后的3个控制点。（r阶导数只与r+1个控制点有关） |
| 对称性 | 控制点相反，能构造出一样的曲线  C:\Users\15624\AppData\Local\Packages\Microsoft.Office.OneNote_8wekyb3d8bbwe\TempState\msohtmlclipclip_image048.png  Bezier曲线在控制多边形的起点和终点具有相同的性质 |
| 凸包性 | 由Bernstein基函数的正性和权性可知  Bezier曲线位于控制多边形构成的凸包之内，而且永远不会超出凸包的范围 |

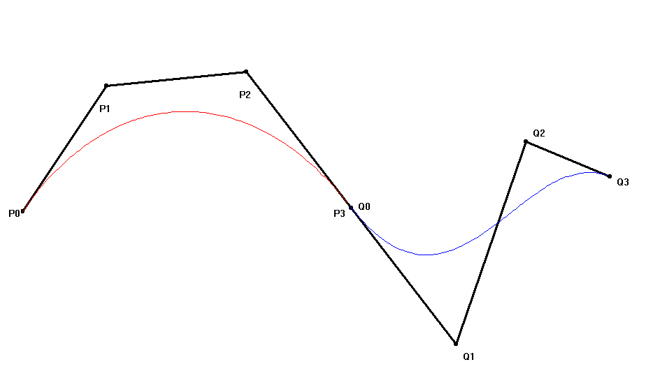
de Casteljau算法



Bezier曲线的拼接

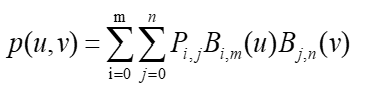
两段三次Bezier曲线达到G0连续性的条件是：P3＝Q0。

达到G1连续性的条件是：P2、P3（Q0）和Q1三点共线，且P2和Q1位于P3（Q0）的两侧。



Bezier曲面

定义 两组正交的Bezier曲线控制点构造空间网格来生成曲面。m×n次张量积形式的 Bezier曲面的定义如下：（u，v）∈〔0，1〕×〔0，1〕



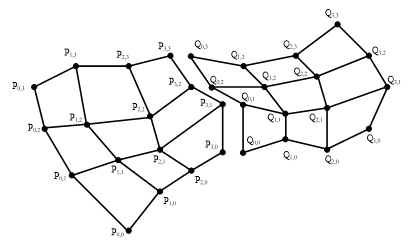
双三次Bezier曲面的拼接

达到G0连续性的条件是（两点重合）

C:\Users\15624\AppData\Local\Packages\Microsoft.Office.OneNote_8wekyb3d8bbwe\TempState\msohtmlclipclip_image052.png

达到G1连续性的条件是（4点在一条直线）

lô)p = l'Zd 



B样条曲线

提出的原因

Bezier曲线虽然有许多优点，但也存在不足之处：

其一、确定了控制多边形的顶点个数为n+1个，也就确定了曲线的次数为n次；

其二、控制多边形与曲线的逼近程度较差，次数越高，逼进程度越差；

其三、曲线不能局部修改，调整某一控制点将影响到整条曲线，原因是Bernstein基函数在整个区间[0，1]内有支撑，所以曲线在区间内任何一点的值都将受到全部顶点的影响，调整任何控制点的位置，将会引起整条曲线的改变；

其四、Bezier曲线的拼接比较复杂。

优点

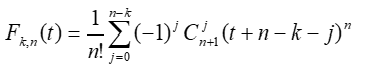
B样条曲线比Bezier曲线更贴近控制多边形，曲线更光滑（很容易达到C2连续性），其多项式的次数可根据需要指定，而不像Bezier曲线多项式的次数是由控制点的个数来确定。除此之外B样条曲线的突出优点是增加了对曲线的局部修改功能，因为B样条曲线是分段组成的，所以控制多边形的顶点对曲线的控制灵活而直观。

定义（均匀B样条曲线）

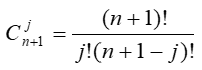
给定m+n+1个控制点Pi（i＝0，1，2，…，m+n ），可以定义m+1段n次的参数曲线

PI n (t) Pl+kFk (t) 
te [0,1] 

F k,n(t)为n次B样条基函数，其形式为



n) 



性质

|  |  |
| --- | --- |
| 连续性 | B样条曲线各段之间自然连接 |
| 局部性质 | 在B样条曲线中，n次每段B样条曲线受n+1个控制点影响，改变一个控制点的位置，最多影响n+1个曲线段，其它部分曲线形状保持不变。 |

Chap8 几何分形

分形的特征

自相似性

无标度性

区分什么是分形

分形集具有任意尺度下的比例细节，或者说具有精细结构。

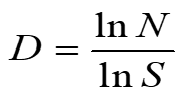
分形集是不规则的，以至于不能用传统的几何语言来描述。

分形集通常具有某种自相似性，或许是近似的或许是统计意义下的自相似。

分形集在某种方式下定义的“分维数”一般大于它的拓扑维数。

分形集的定义常常是非常简单的，或许是递归的。

分形维度的定义



D代表分维

N为与整体自相似性的局部形体的个数

S为相似比（整体/ 局部形体的长度）

Cantor集

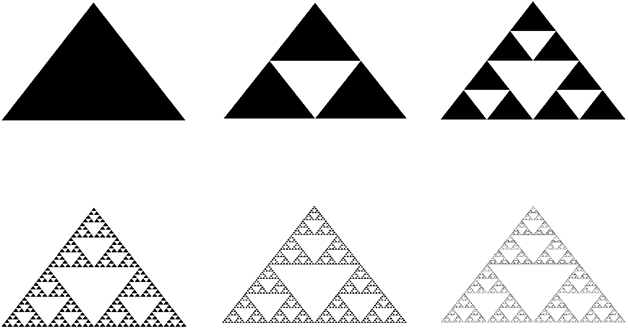


分 形 维 数 ： D 一 1n20n3 ～ 0 ． 6309 。 

程序实现

void CTestView: :Cantor(CP2 PO,CP2 PI,int 
if(0==n) 
pDC->MoveTo(Round(P0.x),Round(P0.y)); 
pDC->LineTo(Round(P1.x),Round(P1.y)); 
return; 
CP2 TO,TI•, 

Sierpinski垫片和地毯



分形维数：D=ln3/ln2≈1.585。

У20) 
Р2(Х2, У2) 
Р01(хо1, У01) 
Р12 (хп, УП) 
Ро(хо, 

分形时，求出个线段终点，使之重构成三个三角形

01 
12 
20 

void CTestView: :Gasket(int n,CP2 p0,CP2 pl,CP2 p2) 
if(0==n) 
FillTriangle(p0,p1,p2); 
return; 
CP2 
Gasket(n-1,p0,p01,p20); 
Gasket(n-1,p01,p1,p12); 
Gasket(n-1,p20,p12,p2); 