三维面心立方网格下的直线生成算法

何丽君[1),2)] 刘勇奎[2)] 孙世昶[2)]

[1)](大连理工大学数学科学学院 大连 116024)

[2)](大连民族学院计算机科学与工程学院 大连 116600)

摘 要 以菱形十二面体为体素构成的三维面心立方(Face-Centered Cubic,FCC)网格是六角网格在三维的一种推广,直线生成算法在三维图形和图像应用中是一个非常重要和基础的算法.文中首先研究了二维六角网格下基于附属菱形空间的直线生成算法,然后将其推广至三维 FCC 网格,得到了一种 FCC 网格下的直线生成算法,该算法在三维方形网格下的 Bresenham 算法的基础上,利用附属平行六面体空间的平行六面体与 FCC 网格空间的体素之间的一一对应关系生成直线.该算法应用简单的判断公式,一步最多可生成 3 个体素,且只涉及到整数运算,因而没有累计误差.

关键词 菱形十二面体;面心立方网格;算法;体素;直线生成

中图法分类号 TP391 **DOI 号**: 10.3724/SP.J.1016.2010.02407

A Line Generation Algorithm on 3D Face-Centered Cubic Grid

HE Li-Jun[1),2)] LIU Yong-Kui[2)] SUN Shi-Chang[2)]

[1)](*School of Mathematics Sciences, Dalian University of Technology, Dalian* 116024)

[2)](*College of Computer Science and Engineering, Dalian Nationalities University, Dalian* 116600)

Abstract Three-dimensional grid where the voxels are rhombic dodecahedra is called FCC (face-centered cubic) grid which is one of the three-dimensional equivalents of the two-dimensional hexagonal grid. 3D Line generation algorithm is an important and fundamental algorithm in applications for 3D graphics and images. An integer line generation algorithm on the FCC grid is presented in this paper. Firstly, a line generation algorithm is observed on the 2D hexagonal grid based on the adjunct rhomb space, and then is extended to the 3D FCC grid. The method is a modification of the 3D cubic Bresenham algorithm based on an adjunct parallelepiped space with the same center and basis vectors. The 3D FCC line is generated employing the one-to-one correspondence between the parallelogram cells of parallelepiped space and the voxels of the FCC space. The procedure is characterized by a simple discriminator with up to 3 voxels being processed in one step and is implemented in integer form without accumulated error. In this way, the accumulation of rounding errors is eliminated completely.

Keywords rhombic dodecahedron; face-centered cubic (FCC) grid; algorithm; voxel; line generation

---

收稿日期:2009-05-06;最终修改稿收到日期:2010-10-18.本课题得到国家自然科学基金(60675008)、辽宁省教育厅科学技术研究项目(L2010096)、中央高校基本科研业务费专项资金(DC10010114)和大连民族学院青年基金(2007A204)资助.**何丽君**,女,1974 年生,博士研究生,副教授,主要研究方向为计算机图形学、计算几何.E-mail: helijun2002@hotmail.com. **刘勇奎**,男,1961 年生,博士,教授,主要研究领域为计算机图形学、图像处理. **孙世昶**,男,1979 年生,博士研究生,讲师,主要研究方向为计算机图形学、机器学习.

1 引 言

三维直线作为计算机图形学中基本的几何元素,其生成算法是计算机图形学的重要基础,是计算机图形学及 CAD 最基本的算法之一,是数控机床和快速成型机控制系统中直线插补算法的数学基础.三维直线生成算法的应用领域包括物体的真实感显示、光线追踪、医学图像的三维重建、体元素绘制、计算机视觉以及立体几何造型.直线生成算法的好坏不仅直接影响图形生成与显示的效率,而且与数控加工和快速原型制造的速度和精度有直接的关系,因此,研究高效、高精度的直线生成算法,对于开发高性能的计算机数控系统具有十分重要的意义.

目前,已有较多文献讨论了三维方形网格下的直线生成算法,分别将二维直线生成中有代表性的算法等推广到三维.文献[1-2]在二维直线 Bresenham 生成算法的基础上,利用三维直线在坐标平面内的投影,将二维直线 Bresenham 算法推广到三维.文献[3]通过对三维直线特性的分析,充分利用直线的方向性和对称性,结合投影原理,提出了一种基于直线特性和投影原理的三维直线生成算法.文献[4]利用直线在两个坐标平面内的投影,将二维中点画线算法推广到三维,用于三维空间直线的生成.文献[5]利用整数运算来生成面连接体素序列,实现了顶点连接到面连接之间的转换.

除了常见的方形网格外,其它的网格也受到人们的关注.体素为六棱柱构成的无通道(tunnel-free)三维网格有两种[6],体素为菱形十二面体和截八面体构成的三维网格分别称为面心立方(Face-Centered Cubic,FCC)网格和体心立方(Body-Centered Cubic,BCC)网格,FCC 网格和 BCC 网格是二维六角网格在三维的推广.与六角网格类似,FCC 网格和 BCC 网格上的体素分布比立方网格更加紧凑和合理,在三维空间具有更好的采样性质.在不影响体数据质量的前提下,FCC 网格将采样数据密度变为立方网格的 76.98%,即 FCC 网格能比立方网格减少 23.02%的体数据量[7].因此,最近很多研究人员关注 FCC 网格[8-11],总结了 FCC 网格的性质和优点[12];提出了类似于步进立方体(Marching Cubes)算法的 FCC 网格表面绘制重建算法[13],来进行 FCC 网格体数据场等值面生成和绘制;研究了 FCC 网格的一般情形(平行十二面体网格)的快速离散傅立叶变换及其并行算法[14];构造了 FCC 网格的箱样条[15].这些研究说明 FCC 网格因其良好的采样性质将在三维计算机图形学及 CAD 中具有较好的应用前景.

然而,很少有人研究作为 FCC 网格的基础算法之一的直线生成算法.本文则提出了 FCC 网格下的直线生成算法.该算法建立在三维 Bresenham 算法和附属平行六面体的基础之上,只用整数运算,从而消除累积误差并提高算法速度.该算法为基于 FCC 网格的真实感显示、体绘制、光线追踪、消除隐藏线面及三维重建等技术提供了基础.

2 二维六角网格

六角网格理论上优于方形网格,主要原因在于方形网格上除了 4 个边相邻像素外,还有 4 个点相邻像素,而在六角网格上,每个像素的 6 个相邻像素都是边相邻的,因而六角网格直线或曲线更平滑;同时,六角网格上的像素分布更加紧凑和合理,因而可以更好地表示图形与图像的细节.文献[16]证明了方形网格和六角网格在几何意义上是相似的,并且比较了两种网格上的绘制直线和圆的算法.文献[17]提出了一种在六角网格上的直线生成算法.

本节提出了一种六角网格上的直线生成算法.首先,介绍了六角网格坐标系统附属的菱形空间并分析了它们之间的邻接关系,利用它们之间的一一对应关系和 Bresenham 算法,建立了一个有效的六角网格直线生成算法.

2.1 六角网格空间及其附属菱形空间

正六边形可以覆盖整个二维空间(即平面),在六角网格上每个像素对应于一个正六边形,而将正六边形的中心称为网格点.所有像素构成了一个数字六角网格空间.记 $h(x,y)$ 为中心在点 (x,y) 的像素.Brimkov 和 Barneva[6] 创建了六角网格空间的附属菱形空间.对每个六角网格像素 $h(x,y)$,其相应的附属菱形 $r(x,y)$ 按如下方法生成:菱形的中心和像素 $h(x,y)$ 的中心相同,菱形边与六角网格空间的单位基向量相等(图 1).这样一来,附属菱形空间和六角网格空间具有相同的中心和基向量.

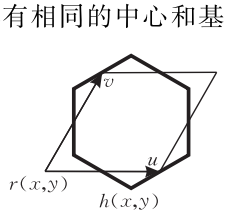


图 1 六角网格像素 $h(x,y)$ 及其相应的菱形 $r(x,y)$

按上述方法建立的菱形空间中,每个菱形 $r(i,j)$ 有8个相邻菱形,其中4个菱形($r(i+1,j)$,$r(i,j+1)$,$r(i-1,j)$和 $r(i,j-1)$)与 $r(i,j)$边相邻,相应的六角网格像素($h(i+1,j)$,$h(i,j+1)$,$h(i-1,j)$和 $h(i,j-1)$)也与 $h(i,j)$边相邻.另外4个菱形($r(i+1,j+1)$,$r(i-1,j+1)$,$r(i-1,j-1)$和 $r(i+1,j-1)$)与 $r(i,j)$点相邻,此时相应的六角网格像素有两种情形:$h(i+1,j+1)$,$h(i-1,j-1)$与 $h(i,j)$边相邻;$h(i+1,j-1)$,$h(i-1,j+1)$与 $h(i,j)$不相邻.表1中列出了它们的相邻关系.

表 1　六角网格像素及其对应的菱形空间菱形之间的邻接关系

相邻的坐标对	邻接关系		需要增加的像素个数
	菱形	像素(六角网格)	
(i,j)和$(i+1,j)$	边相邻	边相邻	0
(i,j)和$(i,j+1)$	边相邻	边相邻	0
(i,j)和$(i+1,j-1)$	点相邻	边相邻	0
(i,j)和$(i+1,j+1)$	点相邻	不相邻	1

2.2　六角网格上的直线生成算法

假设直线起点和终点的六角网格坐标分别为 (x_1,y_1)和(x_2,y_2),则(x_1,y_1)和(x_2,y_2)同时也是两点的相应菱形坐标.记 $\Delta x=|x_2-x_1|$,$\Delta y=|y_2-y_1|$,$xsign=\mathrm{SIGN}(x_2-x_1)$,$ysign=\mathrm{SIGN}(y_2-y_1)$.为简便起见,我们这里假设 $\Delta x>\Delta y$,即直线的斜率的绝对值在0与1之间,此时 x 为主坐标轴,x 值每一步都递增(或递减),y 值是否递增(或递减)由 Bresenham 误差变量 $e_{yx}=2\Delta x(d_{xy}-0.5)$决定(图2),变量 e_{yx} 的初始化公式为 $e_{yx}=2\Delta y-\Delta x$.假设当前已选择的离直线最近的菱形记为 $r(x,y)$,则下一步可选的菱形有两种选择:$r(x+xsign,y)$或 $r(x+xsign,y+ysign)$(图2中为 $xsign=ysign=1$),由 e_{yx} 的符号决定究竟选择上述两个菱形中的哪一个.

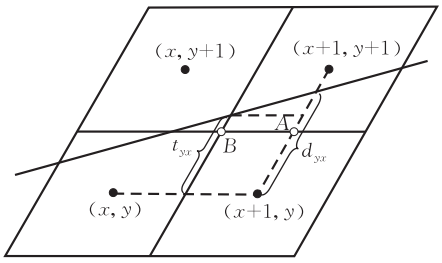


图 2　平行四边形网格下的 Bresenham 算法示意图

若 $e_{yx}<0$,则右方菱形 $r(x+xsign,y)$与直线上理想位置更接近,应取右方菱形,同时变量 e_{yx} 递增为 $e_{yx}=e_{yx}+2\Delta y$.由于相应的六角网格像素 $h(x+xsign,y)$与 $h(x,y)$为边相邻,故六角网格上的下一像素即为 $h(x+xsign,y)$.

若 $e_{yx}\geq 0$,则右上方菱形 $r(x+xsign,y+ysign)$与直线上理想位置更接近,应取右上方菱形,同时变量 e_{yx} 递增为 $e_{yx}=e_{yx}+2(\Delta y-\Delta x)$.此时需要分情况处理:

(1)若相邻的坐标对形如(i,j)和$(i+1,j-1)$,相应的六角网格像素 $h(x+xsign,y+ysign)$与 $h(x,y)$边相邻,故六角网格上的下一像素即为 $h(x+xsign,y+ysign)$.

(2)若相邻的坐标对形如(i,j)和$(i+1,j+1)$,相应的六角网格像素 $h(x+xsign,y+ysign)$与 $h(x,y)$不相邻,需要增加一个像素来连接这两个断开的像素.从 $h(x,y)$到 $h(x+xsign,y+ysign)$有2条可能路径:$h(x,y)$,$h(x,y+ysign)$到 $h(x+xsign,y+ysign)$或 $h(x,y)$,$h(x+xsign,y)$到 $h(x+xsign,y+ysign)$.究竟如何选择呢?从图2中可以看出,只要判断直线在点 B 上方或下方即可.也就是判断 t_{yx} 是否大于1/2.注意到 $t_{yx}-\frac{1}{2}=d_{yx}-h-\frac{1}{2}=\left(\frac{1}{2}+\frac{e_{yx}}{2\Delta x}\right)-\frac{\Delta y}{2\Delta x}-\frac{1}{2}=\frac{e_{yx}-\Delta y}{2\Delta x}$,并且 Δx 大于0,则问题归结为比较 e_{yx} 和 Δy 的大小,若 $e_{yx}<\Delta y$,则接下来的两个像素分别为 $h(x+xsign,y)$和 $h(x+xsign,y+ysign)$,见图3(a);否则 $e_{yx}\geq\Delta y$,接下来的两个像素分别为 $h(x,y+ysign)$和 $h(x+xsign,y+ysign)$,见图3(b).

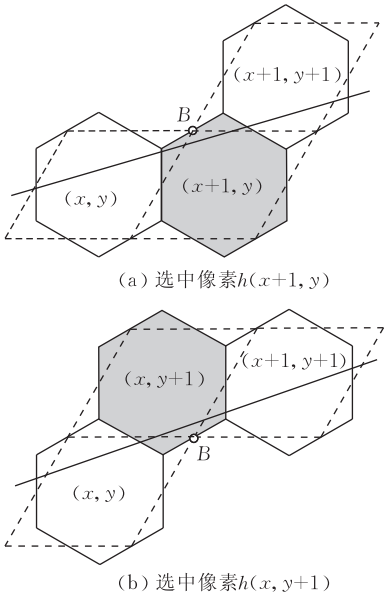


图 3　连接两个不相邻像素 $h(x,y)$和 $h(x+1,y+1)$的像素

若直线斜率绝对值大于1(即 $\Delta y>\Delta x$),则 y 值

每一步总是递增(或递减),x 值是否递增(或递减)由 Bresenham 误差变量 e_{yx} 决定.

2.3 算法分析

与 Bresenham 算法一样,本算法只涉及到整数运算,只用到整数的加法、减法、移位(所有乘 2 用左移运算完成)和比较运算.

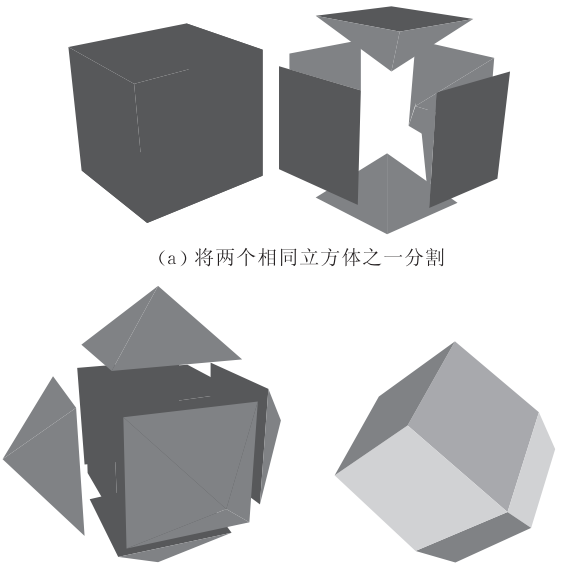
本算法在一步循环中可能生成 1 或 2 个像素,具体的操作次数见表 2. 该表中的计算量不包括 x 和 y 坐标值的操作,因为这些操作在任何直线生成算法中都是一样的,但其中的计算量包括循环语句本身的一次比较和一次加减法操作. 同时,我们只考虑了算法循环体内的操作,因为循环体外的操作只执行一次,对算法速度影响不大.

表 2 本算法循环一次的操作次数

操作类型	一步形成 1 个像素	一步形成 2 个像素
加法	2	2
比较	2	3

3 FCC(Face-Centered Cubic)网格及其坐标系统

由菱形十二面体为体素构成的三维网格称为面心立方(Face-Centered Cubic,FCC)网格. 菱形十二面体由 12 个全等的菱形构成,共有 14 个顶点,其中 8 个顶点的度为 3,6 个顶点的度为 4. 菱形十二面体可由下述方法构造:取两个相同的立方体,立方体表面为深灰色,内部为浅灰色,将其中一个立方体分割成 6 个如图 4(a)所示相同的四棱锥,然后将这 6 个四棱锥的四边形粘帖到另一个立方体的 6 个面上,见图 4(b),即形成如图 4(c)所示的菱形十二面体,此菱形十二面体的 6 个度为 4 的顶点恰好是四棱锥的顶点,度为 3 的 8 个顶点则是原立方体的 8 个顶点. 对于 6 个相同的菱形十二面体,各取其 1 个度为 4 的顶点如切割时相对,则正好可以无缝连接,于是可以铺满整个三维空间.



(a) 将两个相同立方体之一分割

(b) 粘帖 (c) 形成的菱形十二面体

图 4 菱形十二面体的一种构造方法

FCC 网格是二维六角网格在三维的一种推广,它与方形网格相比具有如下优点:

(1) 在 FCC 网格上,每个体素只有点相邻和面相邻两种相邻关系,而方形网格除了点相邻和面相邻外,还有边相邻,因此,在 FCC 网格上,为许多图像处理算法提供了更加简便的实现途径,并提高了算法的效率.

(2) FCC 网格的误差小于方形网格[12]. 从而具有更好的近似效果.

(3) FCC 网格的体素点分布更加合理和紧凑[18]. FCC 网格在 3D 空间比方形网格具有更好的采样性质. 也就是说,对于三维信号,需要较少的采样点就可以完全重构信号. 从而可以节约内存空间,提高体绘制速度.

(4) FCC 网格的体素更接近球面,从而 FCC 网格上曲线或曲面更平滑[9].

(5) FCC 网格更经济. 即在体积相同的条件下,FCC 网格的表面积要小于方形网格.

与传统的方形网格不同,菱形十二面体构成的离散空间中只存在点相邻和面相邻,不存在边相邻的情形. 我们按如下方法构造 FCC 坐标系统:选取一个体素中心定义为坐标原点 O,坐标原点 O 的所有坐标值为 0,即 $O=(0,0,0)$. 然后确定 3 个坐标轴,考虑坐标原点体素(菱形十二面体)的一个有 3 个相邻面的顶点(即度为 3 的顶点),过原点且分别垂直于这 3 个相邻面的直线为坐标轴 Ox,Oy 和 Oz. 基向量 $\boldsymbol{u}$,$\boldsymbol{v}$ 和 $\boldsymbol{w}$ 方向分别与坐标轴 Ox,Oy 和 Oz 相同,长度为从坐标原点到相邻体素中心.

4 附属平行六面体空间

我们可以为上一节的离散 FCC 空间定义相应的附属平行六面体空间. FCC 空间中的每个体素 $h(x,y,z)$ 都对应一个平行六面体 $p(x,y,z)$. $p(x,y,z)$ 称为 $h(x,y,z)$ 的附属平行六面体,它们具有相同的中心,并且平行六面体 $p(x,y,z)$ 的边为 FCC 空间的基向量 $\boldsymbol{u}$,$\boldsymbol{v}$ 和 $\boldsymbol{w}$(见图 5). 按这种方法

定义的附属平行六面体空间和 FCC 空间具有相同的中心和基向量.

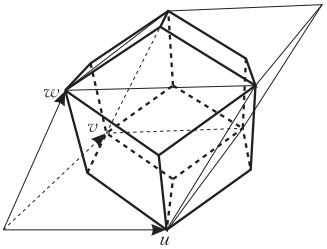
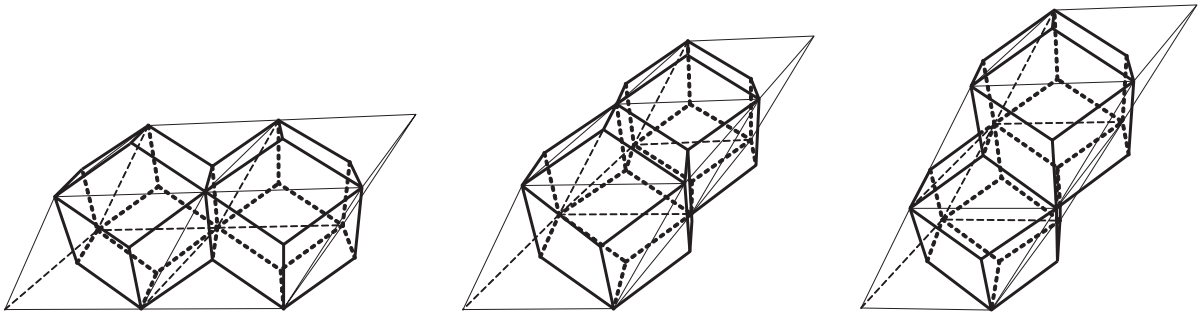


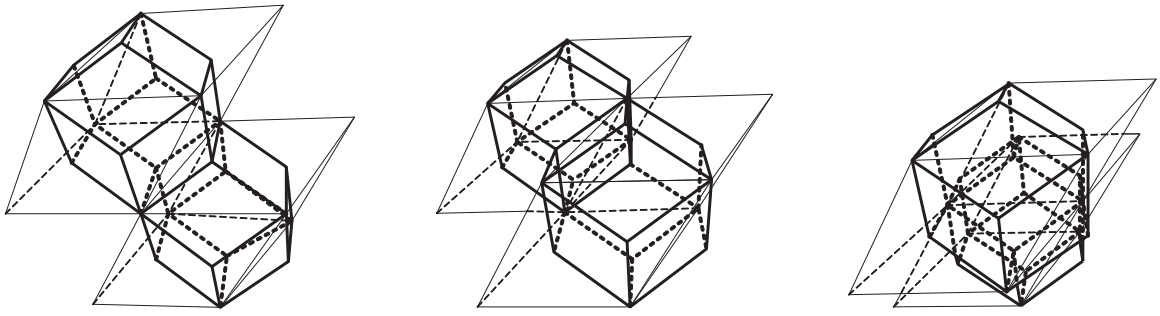
图 5　菱形十二面体和其相应的平行六面体

下面分析构成 FCC 空间和其附属平行六面体空间的体素之间的关系. 记 $h(x,y,z)$ 为 FCC 空间的体素且 $p(x,y,z)$ 为与之对应的平行六面体空间中的平行六面体. 由图 6～10 我们可以得出如下结论:平行六面体 $p(i,j,k)$ 有 26 个相邻平行六面体,其中 6 个与 $p(i,j,k)$ 面相邻,12 个与 $p(i,j,k)$ 边相邻,其余 8 个与 $p(i,j,k)$ 点相邻. 根据对称性,相邻的两个平行六面体有 13 种不同的坐标对,分别对应于表 3 的 13 行,由表中可以看出,面相邻的体素坐标对有 6 种(由于对称性,对应于 12 个面相邻体素),点相邻体素坐标对有 3 中(由于对称性,对应于 6 个点相邻体素),不相邻的体素坐标对有 4 种,其中对应于边相邻的平行六面体 3 种(由于对称性,对应于 6 个边相邻平行六面体),点相邻的平行六面体 1 种(由于对称性,对应于 2 个点相邻平行六面体).



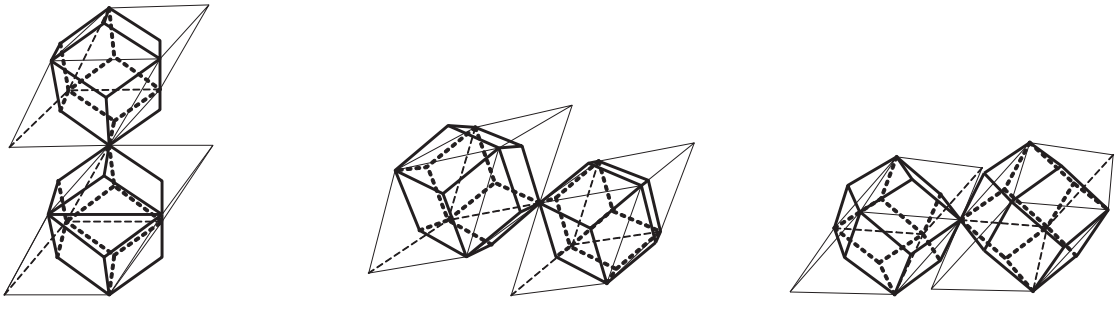
(a) 形如(i,j,k)和$(i+1,j,k)$的体素对　(b) 形如(i,j,k)和$(i,j+1,k)$的体素对　(c) 形如(i,j,k)和$(i,j,k+1)$的体素对

图 6　面相邻平行六面体和面相邻体素



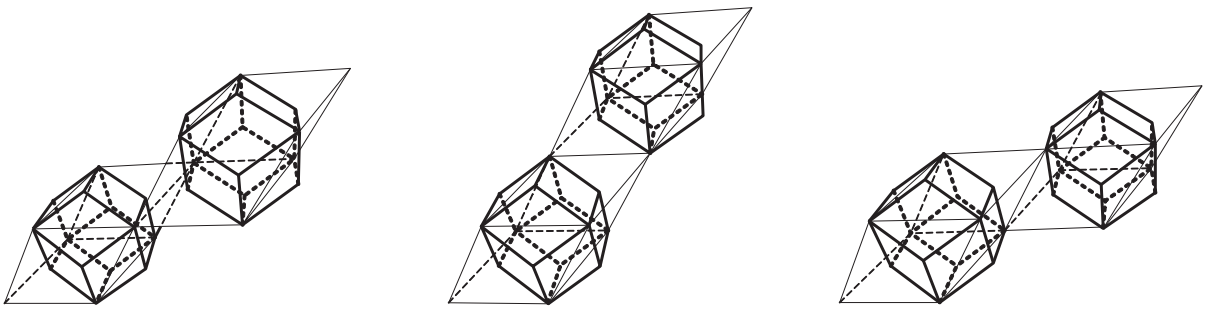
(a) 形如(i,j,k)和$(i+1,j,k-1)$的体素对　(b) 形如(i,j,k)和$(i+1,j-1,k)$的体素对　(c) 形如(i,j,k)和$(i,j+1,k-1)$的体素对

图 7　边相邻平行六面体和面相邻体素



(a) 形如(i,j,k)和$(i+1,j-1,k-1)$的体素对　(b) 形如(i,j,k)和$(i+1,j+1,k-1)$的体素对　(c) 形如(i,j,k)和$(i+1,j-1,k+1)$的体素对

图 8　点相邻平行六面体和点相邻体素



(a) 形如(i,j,k)和$(i+1,j,k+1)$的体素对 (b) 形如(i,j,k)和$(i,j+1,k+1)$的体素对 (c) 形如(i,j,k)和$(i+1,j+1,k)$的体素对

图 9 边相邻平行六面体和不相邻体素

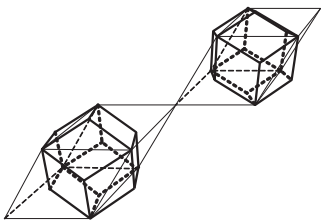


图 10 点相邻平行六面体和不相邻体素,形如(i,j,k)和$(i+1,j+1,k+1)$的体素对

表 3 FCC 网格体素及其对应的平行六面体空间平行六面体之间的邻接关系

相邻的坐标对	邻接关系		图示编号	需要增加的体素个数
	平行六面体	体素(菱形十二面体)		
(i,j,k)和$(i+1,j,k)$	面相邻	面相邻	图 6(a)	0
(i,j,k)和$(i,j+1,k)$	面相邻	面相邻	图 6(b)	0
(i,j,k)和$(i,j,k+1)$	面相邻	面相邻	图 6(c)	0
(i,j,k)和$(i+1,j,k-1)$	边相邻	面相邻	图 7(a)	0
(i,j,k)和$(i+1,j-1,k)$	边相邻	面相邻	图 7(b)	0
(i,j,k)和$(i,j+1,k-1)$	边相邻	面相邻	图 7(c)	0
(i,j,k)和$(i+1,j-1,k-1)$	点相邻	点相邻	图 8(a)	0
(i,j,k)和$(i+1,j+1,k-1)$	点相邻	点相邻	图 8(b)	0
(i,j,k)和$(i+1,j-1,k+1)$	点相邻	点相邻	图 8(c)	0
(i,j,k)和$(i+1,j,k+1)$	边相邻	不相邻	图 9(a)	1
(i,j,k)和$(i,j+1,k+1)$	边相邻	不相邻	图 9(b)	1
(i,j,k)和$(i+1,j+1,k)$	边相邻	不相邻	图 9(c)	1
(i,j,k)和$(i+1,j+1,k+1)$	点相邻	不相邻	图 10	2

对于体素不相邻的情形,需要增加体素来连接这两个不相邻的体素.若相应的两个平行六面体为边相邻的情形,需要增加 1 个体素来连接体素,则由 2.2 节的二维算法的简单三维扩展就可解决 FCC 网格下体素的选择问题,即由 xy 平面、xz 平面或 yz 平面的二维算法扩展成三维算法.若相应的两个平行六面体为点相邻的情形,需要增加 2 个体素来连接体素,此时坐标对为(i,j,k)和$(i+1,j+1,k+1)$.

注意到有 3 种选择:$h(x+1,y,z)$,$h(x,y+1,z)$和 $h(x,y,z+1)$,分别对应 3 个平行六面体:$p(x+1,y,z)$,$p(x,y+1,z)$和 $p(x,y,z+1)$.要在这 3 个平行六面体中选取一个使其与直线的距离最小,我们根据判断变量 e_{yx} 和 e_{zx} 的大小来选择(e_{yx} 和 e_{zx} 的定义和递增与 2.2 节的二维算法类似),首先在 xy 平面由 $e_{yx}<\Delta y$ 是否成立决定 y 是否先变化,然后在 xz 平面由 $e_{zx}<\Delta z$ 是否成立决定 z 是否先变化.根据 e_{yx} 和 e_{zx} 的大小有以下 4 种情形:

(1) 若 $e_{yx}<\Delta y$ 且 $e_{zx}<\Delta z$,则选取平行六面体 $p(x+1,y,z)$(体素 $h(x+1,y,z)$).

(2) 若 $e_{yx}<\Delta y$ 且 $e_{zx}\geq\Delta z$,则选取平行六面体 $p(x,y,z+1)$(体素 $h(x,y,z+1)$).

(3) 若 $e_{yx}\geq\Delta y$ 且 $e_{zx}<\Delta z$,则选取平行六面体 $p(x,y+1,z)$(体素 $h(x,y+1,z)$).

(4) 若 $e_{yx}\geq\Delta y$ 且 $e_{zx}\geq\Delta z$,则选取平行六面体 $p(x,y+1,z)$或 $p(x,y,z+1)$(体素 $h(x,y+1,z)$或 $h(x,y,z+1)$).

首先考虑情形(1),$e_{yx}<\Delta y$ 且 $e_{zx}<\Delta z$,则选取的体素 $h(x+1,y,z)$与 $h(x+1,y+1,z+1)$还不相邻,连接它们的体素可能为 $h(x+1,y,z+1)$或 $h(x+1,y+1,z)$.通过分析,我们发现变量 $e_{zy}=e_{zx}\Delta y-e_{yx}\Delta z$ 的符号决定了该选取哪个连接体素:若 e_{zy} 为正,则应选取 $h(x+1,y,z+1)$,即体素 $h(x,y,z)$的紧接着的 3 个体素为 $h(x+1,y,z)$,$h(x+1,y,z+1)$,$h(x+1,y+1,z+1)$;否则体素 $h(x,y,z)$的紧接着的 3 个体素为 $h(x+1,y,z)$,$h(x+1,y+1,z)$,$h(x+1,y+1,z+1)$.

下面说明为什么变量 e_{zy} 能决定体素 $h(x+1,y,z+1)$或 $h(x+1,y+1,z)$的选取.图 11 画出了 yz 平面在 $x=x+1$ 处的情况,可以看出,若直线在点 E 上方,即 $d_{zx}+d>1/2$,则应选取体素 $h(x+1,y,z+1)$,否则应选取体素 $h(x+1,y+1,z)$.由图 11 看出,d 可由下式计算:

$$d=\left(\frac{1}{2}-d_{yx}\right)\frac{\Delta z}{\Delta y},$$

这样公式 $d_{zx}+d>1/2$ 可写为

$$d_{zx}+\left(\frac{1}{2}-d_{yx}\right)\frac{\Delta z}{\Delta y}>\frac{1}{2} \tag{1}$$

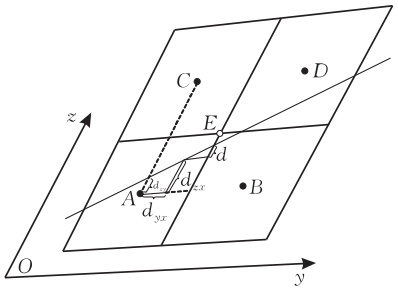


图 11　yz 平面的判断变量的计算示意图

为消除浮点数运算,下式将 d_{yx}(或 d_{zx})为转换为 Bresenham 误差变量 e_{yx}(或 e_{zx}):

$$d_{yx}=\frac{1}{2}+\frac{e_{yx}}{2\Delta x},\quad d_{zx}=\frac{1}{2}+\frac{e_{zx}}{2\Delta x} \tag{2}$$

将式(2)中的 d_{yx} 和 d_{zx} 代入式(1)得

$$\frac{1}{2}+\frac{e_{zx}}{2\Delta x}+\left[\frac{1}{2}-\left(\frac{1}{2}+\frac{e_{yx}}{2\Delta x}\right)\right]\frac{\Delta z}{\Delta y}>\frac{1}{2},$$

即

$$\frac{e_{zx}\Delta y-e_{yx}\Delta z}{2\Delta x\Delta y}>0 \tag{3}$$

由于 Δx 和 Δy 为正数(绝对值),则式(3)等价于

$$e_{zy}=e_{zx}\Delta y-e_{yx}\Delta z>0.$$

于是变量 e_{zy} 能决定体素 $h(x+1,y,z+1)$ 或 $h(x+1,y+1,z)$ 的选取.

现在考虑情形(2),$e_{yx}<\Delta y$ 且 $e_{zx}\geq\Delta z$,则选取的体素 $h(x,y,z+1)$ 与 $h(x+1,y+1,z+1)$ 还不相邻,连接它们的体素可能为 $h(x+1,y,z+1)$ 或 $h(x,y+1,z+1)$,但由于 $e_{yx}<\Delta y$,只能选取 $h(x+1,y,z+1)$. 于是体素 $h(x,y,z)$ 的紧接着的 3 个体素为 $h(x,y,z+1)$,$h(x+1,y,z+1)$,$h(x+1,y+1,z+1)$.

现在考虑情形(3),$e_{yx}\geq\Delta y$ 且 $e_{zx}<\Delta z$,则选取的体素 $h(x,y+1,z)$ 与 $h(x+1,y+1,z+1)$ 还不相邻,连接它们的体素可能为 $h(x+1,y+1,z)$ 或 $h(x,y+1,z+1)$,但由于 $e_{zx}<\Delta z$,只能选取 $h(x+1,y+1,z)$. 于是体素 $h(x,y,z)$ 的紧接着的 3 个体素为 $h(x,y+1,z)$,$h(x+1,y+1,z)$,$h(x+1,y+1,z+1)$.

最后考虑情形(4),$e_{yx}\geq\Delta y$ 且 $e_{zx}\geq\Delta z$,则可能选取的体素为 $h(x,y+1,z)$ 或 $h(x,y,z+1)$,与情形(1)的分析类似,我们可以在 yz 平面由 $e_{zy}=e_{zx}\Delta y-e_{yx}\Delta z$ 的符号决定应该选取哪个体素. 若 $e_{zy}<0$,则应该选取 $h(x,y+1,z)$ 和 $h(x,y+1,z+1)$,于是体素 $h(x,y,z)$ 的紧接着的 3 个体素为 $h(x,y+1,z)$,$h(x,y+1,z+1)$,$h(x+1,y+1,z+1)$. 否则若 $e_{yx}\geq 0$,体素 $h(x,y,z)$ 的紧接着的 3 个体素为 $h(x,y,z+1)$,$h(x,y+1,z+1)$,$h(x+1,y+1,z+1)$.

5　FCC 网格上的直线生成算法

假设直线的起点和终点的 FCC 坐标分别为 (x_1,y_1,z_1) 和 (x_2,y_2,z_2),则 (x_1,y_1,z_1) 和 (x_2,y_2,z_2) 同时也是相应点的平行六面体坐标. 令 $\Delta x=|x_2-x_1|$,$\Delta y=|y_2-y_1|$ 和 $\Delta z=|z_2-z_1|$,$xsign=\mathrm{SIGN}(x_2-x_1)$,$ysign=\mathrm{SIGN}(y_2-y_1)$ 和 $ysign=\mathrm{SIGN}(z_2-z_1)$. 为简便起见,我们这里假设 $\Delta x>\Delta y$ 且 $\Delta x>\Delta z$(即 x 轴为主坐标轴),则 x 值每一步都递增(或递减). FCC 网格上,两点 (x_1,y_1,z_1) 和 (x_2,y_2,z_2) 之间的直线生成算法包括如下步骤:

1. 选取起点 (x_1,y_1,z_1) 为当前体素(也是当前平行六面体).

2. 与三维方形网格直线生成算法 Bresenham 算法类似,在平行六面体空间决定下一步所选取的平行六面体(我们将在下一节详细说明其选取方法).

3. 根据当前平行六面体和选取的下一个平行六面体的相邻关系和对应体素的相邻关系,采用上一节的方法决定是否需要增加体素.

4. 若还没到达终点 (x_2,y_2,z_2),则转向步 2,否则算法结束.

6　在平行六面体空间选择下一个平行六面体

为了选择下一个平行六面体,我们采用与三维方形网格下 Bresenham 算法类似的方法,下面将说明其详细过程. 由于算法在平行六面体空间 x 值每一步都递增(或递减),因此只要决定下一平行六面体的 y 和 z 坐标是否变化即可. 假设当前平行六面体为 $E=p(x,y,z)$,则下一平行六面体的取法只有四种可能的选择,即 $A=p(x+xsign,y,z)$,$B=p(x+xsign,y+ysign,z)$,$C=p(x+xsign,y,z+zsign)$ 和 $D=p(x+xsign,y+ysign,z+zsign)$(图 12). 要在这 4 个平行六面体中选取一个使其与直线的距离最小,我们根据判断变量 e_{yx} 和 e_{zx} 的符号来选择. 我们分两步进行,首先在 xy 平面由 e_{yx} 的符号决定 y 是否变化,然后在 xz 平面由 e_{zx} 的符号决定 z 是否变化. 根据 e_{yx} 和 e_{zx} 的符号有以下 4 种情形:

(1)若 $e_{yx}<0$ 且 $e_{zx}<0$,则下一平行六面体为 $p(x+xsign,y,z)$.

(2)若 $e_{yx}<0$ 且 $e_{zx}\geq 0$,则下一平行六面体为 $p(x+xsign,y,z+zsign)$.

(3)若 $e_{yx}\geq 0$ 且 $e_{zx}<0$,则下一平行六面体为 $p(x+xsign,y+ysign,z)$.

(4)若 $e_{yx}\geq 0$ 且 $e_{zx}\geq 0$,则下一平行六面体为 $p(x+xsign,y+ysign,z+zsign)$.

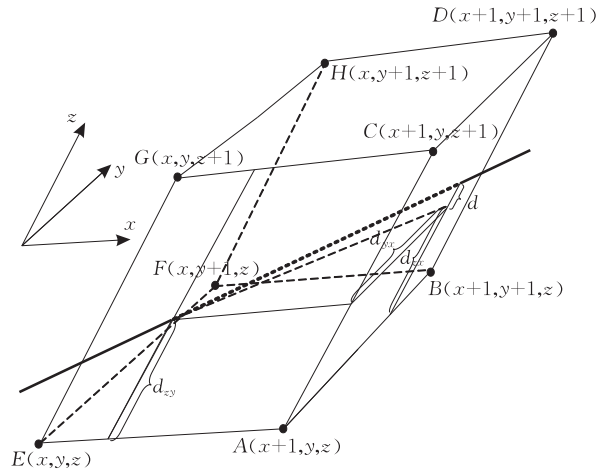


图 12 三维平行六面体空间的 Bresenham 算法示意图

7 算法分析与实验

本算法只用到整数运算,而且除了在一步生成 3 个体素的时候需要 2 次整数乘法运算外,其余的只用到整数加法、减法、移位、比较运算.算法主要集中在第 2 步和第 3 步,这两步需要重复 $N=\mathrm{Max}(\Delta x,\Delta y,\Delta z)$次,在算法第 2 步,需要 2 次比较和 2 次加法操作,在算法第 3 步,若不需要增加体素,即一次仅生成一个体素则没有任何操作;若需要增加 1 个体素,即一步生成 2 个体素,则第 3 步需要 1 次比较操作;若需要增加 2 个体素,即一步生成 3 个体素,则第 3 步最多需要 2 次比较、2 次乘法和 1 次加法运算共 5 次操作.因此,本算法的时间复杂度为 $O(N)$.

我们用 VC6.0 在 Intel 2.53GHz 微机上编程实现了本文算法.为了测试算法的速度,我们在面心立方网格中嵌入一个球面,随机取球面上的点,连接其和球心得到一条直线,则该直线即为球面的半径,通过生成这些直线得到算法的执行时间.我们通过改变球面半径大小(直线长度)和直线数目得到不同组数据.表 4 列出了对于不同长度和数目的直线,本文算法的执行时间比较,表中所列的时间是执行完整的算法所需的时间,即包括主循环体外对判断变量等的初值和增量的计算.从表 4 可以看出,随着直线长度的加大和直线数目的增加,算法执行时间逐步增加.

表 4 本文算法生成不同长度直线所需要的时间

长度	时间/s		
	10 万条	100 万条	1000 万条
200	0.323	3.265	32.788
400	0.422	4.580	43.285
600	0.542	5.397	53.219
800	0.656	6.382	63.455
1000	0.765	7.429	75.726
1200	0.859	8.513	85.888

8 结 论

本文提出了一个 FCC 网格下的三维直线生成算法,该算法基于三维 Bresenham 算法和附属平行六面体空间,在每一步利用判断变量找出离直线最近的体素(菱形十二面体).算法的每一步最多可生成 3 个体素.本文算法和已有的三维直线生成算法相比,优点在于生成的直线是基于 FCC 网格的,由于 FCC 网格比方形网格具有更好的采样性质,并且具有较小的误差,因此,本文算法生成为 FCC 网格直线,在相同质量的条件下,体素数量更少,生成的直线更平滑.但本文算法要较方形网格直线生成算法稍为复杂.

参 考 文 献

[1] Liu Yong-Kui, Shen Hong, Shi Jiao-Ying. An efficient integer algorithm for traversing voxels along a 3D line. Chinese Journal of Computers, 2002, 25(11): 1257-1262(in Chinese)
(刘勇奎,沈红,石教英. 一个有效的沿三维直线的体素遍历整数算法. 计算机学报, 2002, 25(11): 1257-1262)

[2] Tu Xiao-Ming, Liu Xiong-Wei. Generalizing Bresenham's algorithm to 3D straight-line. Journal of Computer-Aided Design & Computer Graphics, 2001, 13(9): 779-782 (in Chinese)
(屠晓明,刘雄伟. 直线 Bresenham 生成算法的三维推广. 计算机辅助设计与图形学学报, 2001, 13(9): 779-782)

[3] Deng Wei-Yan, Lu Guo-Dong, Chen Long. New 3D line-generating algorithm based on line property and projecting. Journal of Zhejiang University (Engineering Science), 2007, 41(4): 626-629(in Chinese)
(邓卫燕,陆国栋,陈龙. 基于直线特性和投影原理的三维直线生成算法. 浙江大学学报(工学版), 2007, 41(4): 626-629)

[4] Wang Zhi-Xi, Wang Run-Yun. Generalizing midpoint algorithm to 3D straight-line. Computer Simulation, 2004, 21(4): 40-42, 79(in Chinese)
(王志喜, 王润云. 中点画线算法的三维推广. 计算机仿真, 2004, 21(4): 40-42, 79)

[5] Cohen-Or D, Kaufman A. 3D line voxelization and connectivity control. IEEE Computer Graphics and Applications, 1997, 17(6): 80-87

[6] Brimkov V E, Barneva R P. Analytical honeycomb geometry for raster and volume graphics. The Computer Journal, 2005, 48(2): 180-199

[7] Ibáñez L, Hamitouche C, Roux C. Determination of discrete sampling grids with optimal topological and spectral properties. Lecture Notes in Computer Science, 1996, 1176: 181-192

[8] Matej S, Lewitt R M. Efficient 3D grids for image reconstruction using spherically-symmetric volume elements//Proceedings of the Nuclear Science Symposium and Medical Imaging Conference. Norfolk, Virginia, 1994: 1177-1181

[9] Strand R, Borgefors G. Distance transforms for three-dimensional grids with non-cubic voxels. Computer Vision and Image Understanding, 2005, 100(3): 294-311

[10] Strand R. Surface skeletons in grids with non-cubic voxels//Proceedings of the Pattern Recognition'04. Cambridge, UK, 2004: 548-551

[11] Linh T K, Imiya A, Strand R, Borgefors G. Supercover of Non-square and Non-cubic grids//Proceedings of the Combinatorial Image Analysis: 10th International Workshop. Auckland, New Zealand, 2004: 88-97

[12] Miller E G. Alternative tilings for improved surface area estimates by local counting algorithms. Computer Vision and Image Understanding, 1999, 74(3): 193-211

[13] Strand R, Stelldinger P. Topology preserving marching cubes-like algorithms on the face-centered cubic grid//Proceedings of the 14th International Conference on Image Analysis and Processing. Modena, Italy, 2007: 781-788

[14] Yao Ji-Feng, Sun Jia-Chang. HFFT on parallel dodecahedron domains and its parallel implementation. Journal of Numerical Methods and Computer Applications, 2004, (4): 303-314 (in Chinese)
(姚继锋, 孙家昶. 平行十二面体区域上的快速离散傅立叶变换及其并行实现. 数值计算与计算机应用, 2004, (4): 303-314)

[15] Kim M, Entezari A, Peters J. Box spline reconstruction on the face-centered cubic lattice. IEEE Transactions on Visualization and Computer Graphics, 2008, 14(6): 1523-1530

[16] Wuthrich C A, Stucki P. An algorithm comparison between square- and hexagonal-based grids. Graphical Models and Image Processing, 1991, 53(4): 324-339

[17] Liu Y K. The generation of straight lines on hexagonal grids. Computer Graphics Forum, 1993, 12(1): 27-31

[18] Theußl T, Möller T, Gröller M E. Optimal regular volume sampling//Proceedings of the Conference on Visualization. San Diego, USA, 2001: 91-98

HE Li-Jun, born in 1974, Ph. D. candidate, associate professor. Her research interests include computer graphics, and computational geometry.

LIU Yong-Kui, born in 1961, Ph. D., professor. His research interests include computer graphics, and image processing.

SUN Shi-Chang, born in 1979, Ph. D. candidate, lecturer. His research interests include computer graphics, and machine learning.

Background

In two dimensions, the hexagonal grid is theoretically better than the square grid due to its denser packing than the standard square grid. Furthermore, the hexagonal grid only has edge-adjacent neighbors.

In three dimensions, the digital three dimensional objects are usually generated on cubic grid on which the cubes are called voxels. In all of the rich algorithms on the cubic grid, line generation algorithms that convert a 3D continuous line representation into a discrete line representation are basic but very important in volume graphics. First, these algorithms are used for synthesizing voxel-based objects in volume graphics. The 3D line itself is also used to build block for voxelizing more complex objects. The second application of the 3D line voxelization algorithms is for ray traversal in voxel space. Ray tracing techniques that cast rays (lines) through a volume of voxels are based on algorithms that generate the set of voxels visited by the continuous ray.

Alternatives to the cubic grid in 3D have also been considered. Two honeycomb graphical models in which the voxels are hexagonal prisms are introduced. Three-dimensional grids where the voxels are rhombic dodecahedra and truncated octahedron are called BCC (Body-Centered Cubic) grid

and FCC(Face-Centered Cubic) grid respectively. The BCC grid and FCC grid are the three-dimensional equivalents of the two-dimensional hexagonal grid. Many recent studies have focus on BCC grid and FCC grid. Some properties and advantages of grids based on BCC grid and FCC grid have been studied.

In 2002, the group proposed a multi-step 6-connectivity voxel traversing algorithm, which uses only integer operation. The algorithm is capable of traversing only voxels having the form of unit cubes, i. e., having length, width and height equal to one. In 2006, the algorithm was extended to uniformly divided voxel space. The new algorithm is a more practical and general one traversing voxels having the form of a block. It uses only integer arithmetic, while the start and end points of the line segments can have any position in a voxel.

This paper gives an efficient line generation algorithm on the FCC grid. The algorithm is based on the adjunct parallelepiped grid and the 3D cubic Bresenham's line drawing algorithm. The algorithm can be implemented in integer form. In this way, it is faster, and the accumulation of rounding errors is eliminated completely.

This work was partly supported by the National Natural Science Foundation of China (60675008), the Research Project of Science and Technology of Education Department of Liaoning Province (L2010096), the Fundamental Research Funds for the Central Universities (DC10040114), and the Youth Science Foundation of Dalian Nationalities University (2007A204).