虚拟现实场景中基于八叉树的网格简化方法

余文勇　李亚军　陈幼平　周祖德
(华中科技大学机械科学与工程学院,武汉 430074)
E-mail:weyoung@mail.hust.edu.cn

摘　要　在基于顶点聚类的网格简化算法中，通常对网格模型进行八叉剖分来建立一棵顶点树以表示整个网格模型的数据结构;但顶点在空间上分布的不均匀导致了顶点树的不平衡,增加了顶点树的深度。针对上述问题,论文提出了基于紧凑八叉树的剖分方法。该方法在虚拟现实场景中表现了较好的效果。

关键词　虚拟现实　八叉树　包围盒　网格简化

文章编号 1002-8331-(2006)14-0034-03　**文献标识码** A　**中图分类号** TP301

A Mesh Simplification Method Based on Octree in Virtual Reality

Yu Wenyong　Li Yajun　Chen Youping　Zhou Zude
(School of Macharical Science and Engineering,Huazhong University of Science and Technology,Wuhan 430074)

Abstract: In the algorithm of mesh simplification based on vertex clustering,a vertex tree is built to represent the data structure of the models by octonary subdivision.But the spatial asymmetry of the vertexes leads to the vertex tree imbalance and increases its depth.The subdivision algorithm based on tight octree is proposed,whose efficiency is proved in the virtual reality.

Keywords: virtual reality,octree,envelopment-box,mesh simplification

空间八叉树可帮助对组成虚拟现实场景的三角片按从前到后的次序快速排序。因此,利用空间八叉树结构可帮助绘制复杂的多边形场景,通过从后往前或者从前往后的次序一一绘制场景对象,自动完成场景绘制中的消隐;同时利用八叉树结构也可帮助判断对象可见性。

由于八叉树方法充分利用了形体在空间上的相关性,与其他的物体表示方法相比较,有以下优点[1]:

(1)方便实现集合运算。由于物体的八叉树表示就是由它内部含有的大大小小的立方体(体元)组成,物体间的并、交、差等集合运算可转换成对形体体元的简单操作。

(2)有利于形体显示操作。八叉树方法具有结构的分层性与有序性的特点,使之对于物体显示精度与速度的平衡、隐线与隐面的清除等要求,都非常有效。特别在实现消隐的过程中,八叉树的数据结构大大简化了隐藏线和隐藏面的消除。众所周知，各类消隐算法的核心是排序，即将待显示的物体上的点、线、面按离视点的远近排定次序,离观察点近的面片遮挡较远的面片,从而有效地实现三维形体的显示。

由于具有以上的优势,空间八叉树已被广泛应用于图形处理及绘制中,以减少在场景中进行可视面确定、光线跟踪、碰撞检测等计算时需要处理的面的比较次数。八叉树算法可显著减少对场景中多边形进行排序的时间。对于静态场景(即场景中没有可以发生运动的对象，也没有可以和用户发生交互的对象),八叉树是非常有效的。

1　常见的八叉树剖分方法

八叉树是一种空间分割的层次数据结构,可看成是四叉树在三维空间的一种扩展,空间被递归地分成 8 个单元,其根表示了整个景物体,8 个子空间用 0~7 表示,写成二进制是 zyx 形式,即 000~111,传统的八叉树剖分是在每一个方向上进行等分。一般选择平行于坐标轴的分割平面,每次把一个空间均匀划分成八个子空间。

八叉树最早用于表示三维实体,也用于表示几何实体的空间关系,大部分应用是表示对象中单元间的布尔性质,采用八叉树实体表示,使图形学中的某些操作更简易快速,如确定领域、消隐和布尔运算等。

八叉树生成过程是由根结点生成整个树结构从而对场景进行剖分的过程。先取整个场景的最大长宽高构成一个场景的最大包络立方体作为根结点,然后在这个立方体的基础上对场景进行细分,最终得到八叉树结构。生成算法如图 1 所示。

由于所考虑的景物均被包含在一充分大的立方体(称为世界立方体)内,将世界立方体按八叉树方式进行递归剖分,八叉树的每一叶结点对应于空间网格单元,其中含有景物的叶结点称为非空叶结点,否则称为空叶结点。

为有效地存取八叉树中的结点,对树中的每一结点进行编码。实验表明,编码方案的优劣将直接影响八叉树结点的存取效率。这里采用八进制数编码方案,即对同一父结点的八个兄弟结点(立方体网格),其具有最小(x,y,z)值的结点编号为 0,

基金项目:国家 863 高技术研究发展计划资助项目(编号:2001AA423230);武汉市青年科技晨光计划资助项目(编号:20045006071-26)

作者简介:余文勇,博士,讲师,研究方向为虚拟制造、图形图像学和机器视觉。李亚军,硕士生,研究方向为虚拟制造。陈幼平,教授,博导,研究方向为虚拟制造和智能控制。周祖德,教授,博导,研究方向为数字制造和智能控制。

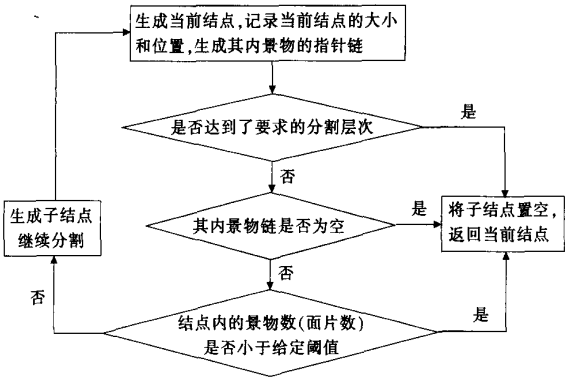


图1 八叉树的生成算法

相邻兄弟结点的编号沿 x 方向增加 1,沿 y 方向增加 2,沿 z 方向增加 4,并将父结点的编号前缀于其八个子结点的编号上,作为结点的编码。为保证八叉树中每一结点具有相同长度的编码,必要时可在结点编后增加一串字符"F"(F 为异于 0,1,…,7 的符号)使其码长均为 N(即顶点树的最大的深度)。这样,每一八叉树结点的编码均可表示为 $q_1q_2\cdots q_iFF\cdots F$,其中 $q_1,q_2,\cdots,q_i\in\{0,1,\cdots,7\},0\leqslant i\leqslant N$。显然,$q_1q_2\cdots q_N$ 表示了空间最低层次(第 N 层)立方体网格单元(边长为单位长度),$q_1q_2\cdots q_iFF\cdots F$ 表示了空间分割至第 i 层时立方体网格,而编码 $FF\cdots F(N$ 个)则表示了世界立方体。图 2 即为上述编码方式的一个例子。

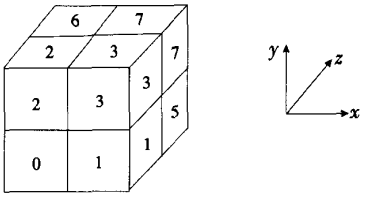


图2 八叉剖分编码

典型的八叉树数据结构在每一结点上保存指向其八个子结点的指针。若每一指针占用一个长机器字(四个字节),则对于具有十万个结点的场景来说,仅指针的内存占用就达 3 125K,这种数据结构将导致巨大的内存耗费。

采用八叉树的最大缺点是它占用存储空间很多。在每一个八叉树结点中,除去描述该结点性质的域以外,还需要存储它指向父结点及八个子结点地址的指针。定义物体的八叉树所需的众多的结点连同描述每个结点的存储域,使得物体的八叉树的表示花费了十分昂贵的存储空间。实际上八叉树表示以存储空间换取了算法的效率[2]。

2 紧凑八叉剖分算法

由于传统的八叉树对空间的划分是均匀的,即在八叉树建立过程中,通常对每个结点的剖分都是以结点的空间二分面作为剖分基准。除非网格模型中的顶点是均匀分布的,否则这种以空间二分面作为剖分基准的方法将导致最终生成的八叉树结构不平衡,从而增加整个八叉树的存储空间以及各结点的遍历时间。

用上一层次包围盒的中分面作为八叉剖分的依据通常会使建立的八叉树不平衡,这是因为:在网格模型中,通常,各个顶点在空间上的分布是不均匀的,如果对各子树平均分配空间,容易导致各子树中顶点数的急剧不均匀,而对于顶点数大大超过平均数的子树,又要不断地对其递归地进行八叉剖分,直至达到给定的标准,这必然导致八叉树层次数的增加。由于顶点在包围盒中的分布是不均匀的,且有些子空间对应的顶点很少或者没有,这样,很多叶子结点中的记录很少或为空,从而浪费了存储空间。因此,本文提出采用基于紧凑八叉树的网格剖分算法[3],如图 3 所示。

基于紧凑八叉树的剖分方法可用以下算法描述:

步骤 1 建立模型的最小包围盒。

步骤 2 以包围盒 X、Y、Z 向的中分面作为剖分基准,将包围盒平均剖分为八个子包围盒。

步骤 3 对于每个部分,若未达到停止剖分标准,则重新从步骤 1 开始进行剖分。

步骤 4 结束。

上述步骤由于对八叉剖分的每一子空间又重新建立其最小包围盒,从而避免了由于该部分的顶点在空间上分布的不均匀而增加了建立的顶点树的深度,进而减少存储空间,加快网格模型数据的读取速度。另外,由于是建立顶点的最小包围盒,在简化误差比较小时,只有空间距离比较近的顶点才会聚合在一起;而相距较远的顶点只有在更大幅度简化时才会聚合。这些特点在一定程度上保证了简化时网格模型的逼真度。

在网格简化过程中,顶点树如何建立直接影响着整个网格模型的存储空间和网格模型简化的效果。

建立八叉树,停止剖分的标准需要考虑两个问题:(1)在剖分过程中应遵循的原则,这可以通过规定一个阀值 M(M 表示子空间中顶点或三角片的个数)来解决,即只有当区域中顶点或三角片的个数多于 M 个时,该区域需要进一步划分;(2)分辨率,即分解时允许达到的最小子空间是多大的问题,这可以规定一个不再需要分割的正方体大小。

每一结点包含以下内容:结点编号、剖分标志、结点在顶点树中的深度以及它所含的景物面片表的入口指针等。下面给出相应的数据结构描述:

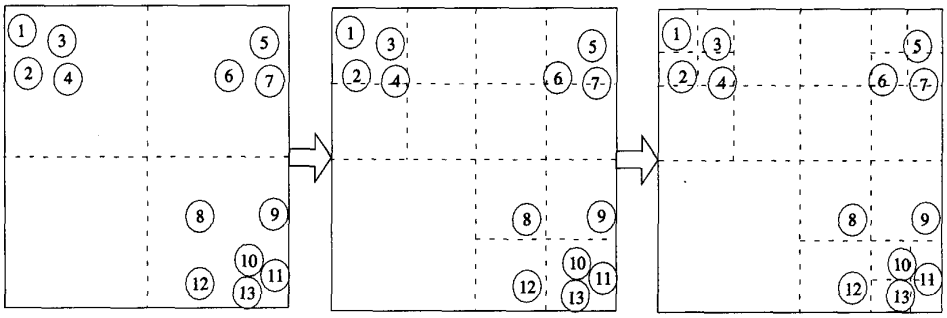
```
Struct Node {
  int currNodeID;//结点编号,用来标识从顶点到本结点的路径
  int depth;//本结点在顶点树中的深度
  NodeStatus label;//剖分标志,如果没有子结点,则为 FALSE
  Node* parent;//顶点树中本结点的父结点
  Node* children[ ];//指向本结点的子结点的指针数组
  Byte numchildren;//本结点所包含的子结点数目
  int faceList[ ];//存放当前结点对应面片集的索引
  int faceNum;//叶结点面片数,如果不是叶结点则为 0
};
```

当结点空间确定后,如果它是叶结点,可根据整个景物空间的三角片表来确定该结点空间的三角片表;如果不是,再查找它的子结点进行直至获得它的三角片表。

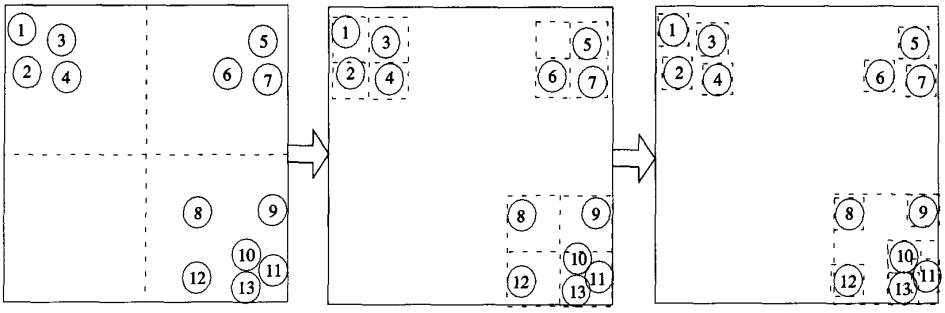
2.1 树生成算法

八叉树生成过程是由根结点生成整个树结构从而对网格模型进行剖分的过程。其算法采用递归调用伪代码如下:

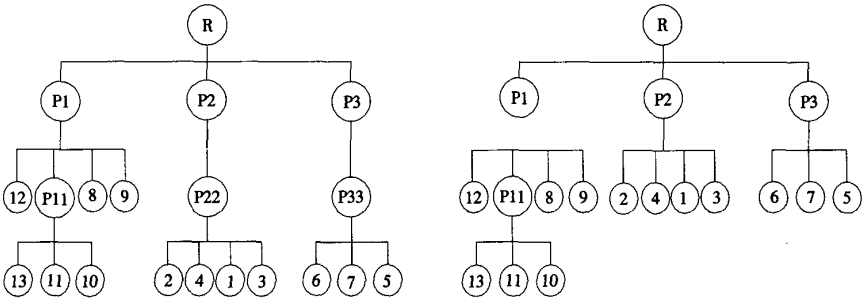
```
if(根结点非空 && 不满足停止剖分要求)
  {
    for(i= 0;i<8;i++)
      生成子结点;
    for(i=0;i<8;i++)
```



(a)传统八叉树的剖分过程



(b)紧凑八叉树的剖分过程



(c)基于传统八叉树剖分所得的顶点树　　(d)基于紧凑八叉树剖分所得的顶点树

图3　以二维空间为例,两种剖分过程比较

```
    生成子树;
  }
```

2.2　子结点生成算法

在找到叶子结点后,还必须为数值运算过程准备网格的顶点坐标、网格的大小、网格的类型及网格的顺序码。其算法的伪代码为:

```
for(i=0;i<8;i++)
  {
    申请一块 Struct Node 大小的内存空间;
    使父结点的指针指向此结点;
    填写 Node 结构各项内容;
  }
```

2.3　树遍历算法

八叉树生成后,其叶子结点才是实体的最终网格,因此要对八叉树进行遍历,找到这些叶子结点,其算法仍采用递归调用,伪代码如下:

```
if(结点不为空)
  {
    if(结点是叶子结点)
      输出结点信息;
    else
      for(i=0;i<8;i++)
        遍历子树;
  }
```

由上述的论述可知,在传统的八叉树剖分中,各叶子结点的包围盒的大小之比为 8^n(n 为 0 或整数);而在基于紧凑八叉树的网格剖分中,各叶结点包围盒大小之间没有固定的比例。由于在顶点树中的结点和树的遍历时间的关系是:树中结点越多,在绘制过程中所用的遍历时间越长。因此,对基于紧凑八叉树的网格剖分算法,分别选取适当的剖分原则以及最小的分辨率,使用户指定的阈值尽量调整到细节和遍历时间的折衷的最优化。如果叶子结点包围盒大小选择合适,其运算过程中所需机器内存较少,一般机器的内存容量足以满足要求,其运算速度相当高;若减小叶子结点,其运算过程所需的机器内存急剧增加,此时必须将一些中间结果存放在外部存储器上,这将使运算速度大大减小。

3　算法比较实验

在 Windows 2000 环境下(硬件配置:1.8GHz P4 CPU,内存

(下转 78 页)

情况 2:若块中含有二类以上的颜色,则块的类型只能是纹理块或边缘块(相邻不同区域边缘处的子块)。显然,要生长的块只能是纹理块,因此应先考察是否是纹理块。若块中含有三类以上的颜色,在颜色比较时,可考察主色调和次色调,以避免多种颜色比较的复杂性,试验表明在一般情况下是可行的。因此生长准则为以下 5 个条件的逻辑与:

(1)相邻块颜色数目应相同;

(2)纹理参数大于规定阈值,即为纹理块;

(3)相邻块纹理参数应近似;

(4)相邻块主色调应近似;

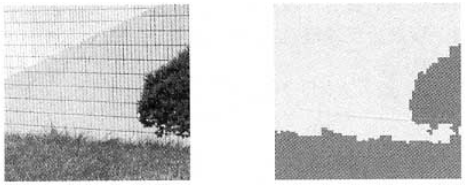
(5)相邻块次色调应近似。

若以上 5 个条件皆满足,则可以按邻域方式不断生长;否则,作为边缘块等待后处理。

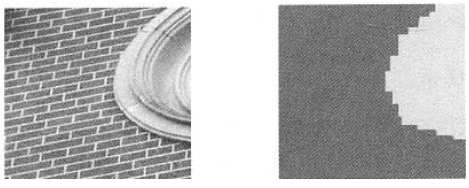
情况 3:若子块为边缘块,则等全部均匀块和纹理块生长完成后,再对每个边缘块进行后处理。处理方法是对边缘块按照 4 叉树规则不断细分,并按照以上生成规则把各个细分的块合并到邻近已经生长好的大区域中,直至所有边缘块处理完为止。

3 实验结果

用本文提出的算法,在 P4 1.7G 的机器上,通过 Visual C++6.0 平台上对 40 幅自然景物图像进行了基于内容的分割,图 2 是两幅典型试验结果。



(a)原始图像 1　　(b)分割结果



(c)原始图像 2　　(d)分割结果

图 2　分割实验结果

原始图像均为 256×256 的 24 位真彩色图像,处理时间一般不到 1s,实时性较好。从两幅图像的分割结果可看出,对于较容易产生过分割的纹理区域,本算法能准确地将其分割出来,边界比较清晰,满足一般基于内容的图像检索或识别处理。

4 结论

本文将聚类和区域生长方法有机结合在一起,并将彩色图像的颜色特征和纹理特征,采用符合人类视觉特征的规则进行融合,实现了彩色图像基于内容的快速、有效分割,本研究为基于内容的彩色图像检索奠定了基础。(收稿日期:2005 年 12 月)

参考文献

1.林开颜,吴军辉,徐立鸿.彩色图像分割方法综述[J].中国图象图形学报,2005;10(1):1~10

2.Miyahala M,Yasuhida Y.Mathematical transform of(R,G,B)color data to Munsell(H,S,V)color data[J].SPIE,1988;1001:650~675

3.Cinque L,Ciocca G,Levialdi S et al.Color-based image retrieval using spatial-chromatic histograms[J].Image And Vision Computing,2001;19(13):979~986

4.叶齐祥,高文,王伟强等.一种融合颜色和空间信息的彩色图像分割算法[J].软件学报,2004;15(4):522~530

5.Ya Chun Cheng,Shu Yuan Chen.Image classification using color,text and regions[J].Image and Vision Computing,2003;21:759~776

6.Julien Fauqueur,NozhaBoujemaa.Region-based image retrieval:fast coarse segementation an fine color description[J].Journal of Visual Languages and Computing,2004;15:69~95

7.Tie Qi Chen,Yi Lu.Color image segmentation-an innovative approach[J].Pattern Recognition,2002;35:395~405

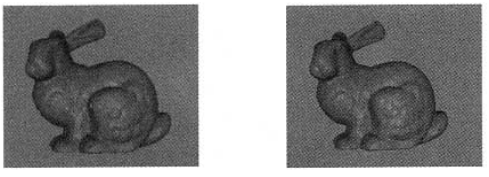
8.Alian Tremeau,Nathalie Borel.A region growing and merging algorithm to color segmentation[J].Pattern Regcognition,1997;30(7):1191~1203

9.蒋苏荣,王松,冯刚.IRGA:一种改进的区域生长彩色图像分割方法[J].计算机工程与应用,2003;39(7):96~97

10.H D Cheng,X H jiang,Y sun et al.Color image segmentation:advance and prospect[J].Pattern Recognition,2001;34:2259~2281

11.H D Cheng,X H jiang,Y sun et al.Color image segmentation based on homogram thresholding and region merging[J].Pattern Recognition,2002;35:373~393

(上接 36 页)



(a)传统八叉树剖分后的简化效果　　(b)基于紧凑八叉树剖分后的简化效果

图 4　两种剖分方法后简化效果的比较

256M)和 Visual C++ 6.0 开发环境下,底层图形绘制和操作采用 OpenGL,在网格简化算法中,分别采用传统的八叉树剖分方法和基于紧凑八叉树的剖分方法。将剖分原则和分辨率设定后,对于两者的简化效果进行比较。图 4(a)为以传统的八叉树剖分方法建立网格模型的顶点树,再简化后得到的效果,含有 9 652 个三角片;图 4(b)为基于紧凑八叉树剖分方法建立网格模型的顶点树,再简化后得到的效果,含有 6 324 个三角片。实验表明,对网格模型进行基于紧凑八叉树的剖分有效地减小树的深度,提高了顶点树的遍历速度,加大了网格模型的简化幅度。(收稿日期:2005 年 12 月)

参考文献

1.耿国华,周明全.一种从空间物体大八叉树转换的简捷算法[J].西北大学学报(自然科学版),1996;26(4):289~292

2.唐荣锡,汪嘉业,彭群生.计算机图形学教程[M].北京:科学出版社,1990-04

3.David P Luebke.View-Dependent Simplification of Arbitrary Polygonal Environments[D].University of North Carolina,1998