Journal of Computer Applications
计算机应用,2014,34(9):2720 - 2724
ISSN 1001-9081
CODEN JYIIDU
2014-09-10
http://www.joca.cn

文章编号:1001-9081(2014)09-2720-05
doi:10.11772/j.issn.1001-9081.2014.09.2720

STL 格式文件的快速拓扑重建算法

王增波*
(衡阳师范学院 数学与计算科学系,湖南 衡阳 421002)
(*通信作者电子邮箱 wonbowonbo@163.com)

摘　要:针对立体光刻(STL)文件所表示的图形要素之间缺乏必要的拓扑关系,对 STL 格式文件进行分析和读取,以哈希表作为查找表快速建立三维模型各要素间的拓扑关系,建立能表示要素关系的点表和面表,利用基于哈希表的拓扑重建算法实现了拓扑结构的快速建立,算法时间复杂度仅为 $O(n)$,空间复杂度为 $O(3n+(4+m)f+m)$。最后,列举 5 个实例进行验证测试,实验结果显示,与直接算法和红黑树法相比,所提出的算法用时更少,在普通计算机上重建含有 65 万个三角面片模型的拓扑结构只需 2.3 s。

关键词:立体光刻格式文件;拓扑重建;哈希函数;三角网格;哈希地址
中图分类号:TP391　　**文献标志码**:A

Fast topological reconstruction algorithm for a STL file

WANG Zengbo*
(*Department of Mathematics and Computational Science, Hengyang Normal University, Hengyang Hunan* 421002, *China*)

Abstract: Because the lack of the necessary topological relation between graphic factors, through analyzing the STL (Stereolithographic) format file and reading it, and using the hash table as a lookup table, the topological relation among various elements of three-dimensional model is quickly created. Using the Hash table, this algorithm created the point table and the surface table for the elements, and realized the topological reconstruction. The time complexity of this algorithm is $O(n)$, while the space complexity of this algorithm is $O(3n+(4+m)f+m)$. Finally, the algorithm was compared with the direct algorithm and red-black tree algorithm through five examples, and the results show that the proposed algorithm costs less time and the model with 650 thousand triangular facets can be reconstructed within 2.3 seconds on PC.

Key words: Stereolithographic (STL) format file; topological reconstruction; hash function; triangular mesh; hash address

0　引言

立体光刻(Stereolithographic, STL)数据格式[1]由 3D Systems 公司发明,在逆向工程中普遍得到应用,是三维测量设备数据输出的主要文件格式。因其格式简单、数据处理方便,所以很快被广泛采用,目前大多数的计算机辅助设计(Computer-Aided Design, CAD)系统都提供 STL 文件的接口[2]。然而,通过对 STL 格式文件的读取无法直接得到三角形面片间的拓扑关系,因为它只是无序地列出几何面模型上三角形的坐标信息,无法直接获取三角形之间的拓扑连接关系,因此在进行区域分割和网格划分前,建立点、边和面的拓扑信息是一项必须的重要工作。同时,STL 格式文件中每个顶点大都被重用了 6 次以致数据的冗余太大,因此有必要建立 STL 格式文件的拓扑结构并设计合理的数据结构,对每个顶点坐标只存储一次以减少数据冗余节省模型的存储空间。所以对无序三角形建立其拓扑关系非常重要。

进行 STL 数字文件拓扑关系的构建过程中若采用直接遍历的方法来查询处理就需要进行排序,当面片的数量级非常大时数据处理的效率将会非常低[3]。针对如何快速实现 STL 文件的三角网格模型的冗余数据滤除和基本拓扑信息的重建,在文献[4 - 17]中国内外的许多学者专家进行过大量的研究。其中文献[8 - 10]采用平衡二叉树的数据结构进行顶点聚合以达到去除冗余顶点的目的,但当数据量较大时,建树过程中调整二叉树平衡所耗费的时间代价过大;文献[13]采用基于红黑树为基础的数据结构实现冗余数据的删除和拓扑结构的重建,基于树型结构的特点,算法在建树和检测平衡性上要耗费一定的时间代价;文献[14]通过新建点面存储结构去除冗余顶点,再利用虚平衡二叉树进行快速邻边搜索,算法效率较高,但是要生成点面存储结构的中间过渡结构,造成了运算效率下降;在翼边结构[17]里每条边使用多个指针分别链接两个顶点、两个邻接面和该边邻接的两个邻边,用它表示多面体模型是完备的,并且能够实现快速查找各元素间的邻接关系,但是不能够表示带有精确曲面边界的实体,过多的指针也为后续工作的编程增加了难度。

本文以查找能达到常量时间的哈希表作为查找表为基础快速建立三维模型各要素间的拓扑关系,为今后的三维模型分割等工作打下基础,因为图形分割都需要找到图形元素间的邻接关系,所以要求事先建立好面、点和边这些元素之间的邻接关系才能实现。所以,本文使用基于哈希表的快速匹配分类方法清除数据冗余,并同时建立面片间的拓扑关系,因为没有建立树型结构及树的平衡性问题,算法实现简单且效率较高。选取了 5 个典型实例用本文算法分别与直接算法和文

收稿日期:2014-04-02;**修回日期**:2014-06-16。
作者简介:王增波(1975 -),男,湖南衡阳人,讲师,硕士,主要研究方向:计算机图形学、智能算法。

献[10]的红黑树法进行效率比较,结果验证了本文算法的有效性和高效性。

1　STL 格式文件分析

STL 格式文件以三角形面为单位记录了每个三角形的简单的几何信息,包含每个三角形的三个顶点坐标及该三角形所在平面的法向矢量。在具体格式表现上它分为 ASCII 码文本和二进制文本两种格式。

ASCII 码格式按一个面片一个面片地给出每个三角面片的几何信息,该格式文件并未给出三角形之间的拓扑关系。

二进制格式的文件是一个固定的格式文件,从文件开始第一部分是文件头部分共 80 B 的字符串,用来存储模型的文字说明信息;紧接着的第二部分是 4 B 的整型数据,记录了该文件总共的面片数目,可以据此可控制读取面片的循环语句的次数;第三部分就是连续记录所有面片的法矢量和三个顶点坐标的信息了,每个面片占用 50 B,其中按顺序分别是 12 B 的法矢量坐标(每个三维分量占 4 B 的浮点数据),12 B 的三角形面片顶点 1 ~ 3 的三维坐标和 2 B 的属性信息。

STL 格式文件中每个三角面片记录了该三角面片本身的法矢和按照右手螺旋法则排列的 3 个顶点的坐标值。这种存储格式可以保证文件阅读十分方便,便于理解,但缺点也很明显:其中的各个三角面片是相对独立、随机存储的,无法体现出面片间的邻接关系,在文件中上下相邻的三角面片在模型中的实际位置可能相距很远;而且同一顶点通常会被不同的三角片多次记录,造成文件中包含了大量的冗余数据。这就需要设计一种数据结构,以降低数据的冗余和建立图形要素之间的邻接关系,即通过对 STL 数字文件的拓扑关系重建,快速检索到指定的图形要素及其相邻接的图形要素。

2　拓扑重建的数据结构设计

图形学中的拓扑关系是指网结构元素(结点、弧段、面域)间的邻接、包含、关联等关系,即要素(图元)之间的连通性或相邻的关系。建立完整的 STL 数据拓扑信息的过程相当费时,特别是对于三角面片很多的情况[7]。与许多基于散乱数据拓扑重建算法[8]不同的是,STL 格式文件还是有一些点与面的相关信息,所以对它们的算法要相对简单一些。

STL 格式文件的拓扑结构重建主要包含两个方面的工作:一是合并重复的顶点减少冗余,也就是将 STL 格式文件中重复列出的顶点坐标只存储一次;二是建立邻接关系,也就是通过某一面片能够找到三个顶点坐标以及与其相邻接的面片,从而建立三角面片与点、面与面之间的毗邻关系。

STL 格式文件的拓扑重建过程就是从 STL 格式文件中依次读取每个三角形的顶点,通过查找并去除重复的顶点,建立一个包含所有不重复顶点的点表(存储每个顶点的坐标),同时建立包含所有三角形在点表中索引值的面表(只存储每个三角形顶点在点表中的索引)。

其中点表是一个顺序表,顺序表中的每个元素有两个域,第一个域存储了每个顶点的坐标信息,第二个域是指针域指向一个链表,该链表记录了该顶点所属于的多个三角形在面表中的索引(由于进行了顶点的合并,每个顶点可能有多个邻接面,需要进行冲突处理)。其中的面表是每个元素包含了三个域的顺序表,分别记录了每个面所包含的 3 个顶点在点表中的编号、法向量信息以及指向存储邻接面片的编号的链表指针。其中每个面记录指向三个顶点的指针,即建立了由面获得包含点的拓扑信息。每一个点记录与之连接的面,即建立了由点获得邻接面片的拓扑信息。其他的拓扑关系都可以根据这两种拓扑信息派生出来。

因此,这里的主要工作就是点的合并以及点表和面表的建立,数据结构和合并算法的设计就显得相当重要,如果数据结构和合并算法设计不好,会导致程序的效率不高。

通过前面的说明构造 STL 格式文件的拓扑关系的关键在于如何去除重复的顶点,而这个过程就是一个顶点的查找过程。如何能够快速查找到重复顶点并把它去除,直接关系到拓扑重建的效率。由于用哈希表存储顶点的坐标检索运算平均可能达到常数的时间,又能通过哈希函数的运算快速找到冗余的顶点坐标和顶点间的拓扑关系,因此,在算法设计过程中,设计了一个顶点哈希表,因此有了顶点哈希表、点表和面表三个表结构。

1)顶点哈希表的结构。顶点哈希表是通过哈希函数实现顶点快速定位的数据结构,主要目的就是用来实现查找,它是一个顺序表,且表中的每个元素连接一个链表。通过给定的哈希函数计算出顶点坐标对应的哈希地址(哈希地址就是顺序表的下标),从该哈希地址所在行的所在连链表中进行查找(如图 1 所示),能够快速地判断出该顶点是否已经存在。链表中的每个结点有两个域,第一个域存储该顶点在点表中的点索引号(顺序表的下标),第二个域存储下一个具有相同哈希地址的不同点的索引号的结点指针。

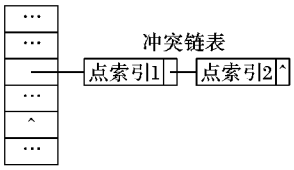


图 1　顶点哈希表

2)点表的结构。点表是一个存储无重复顶点的顺序表,表中每个元素有一个数据域和一个指针域,在相应哈希地址对应的冲突链表中查找不到的顶点(这就保证了不会重复)将被加入到该点表的数据域,并且该顶点所邻接的面片索引会被保存在该行元素的指针域所指向的链表中(如图 2 所示),最终通过这个链表可找到与该顶点邻接的所有面片。

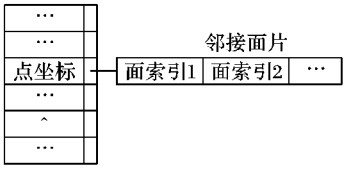


图 2　点表

3)面表的结构。面表是存储所有三角面片点信息的顺序表,表中每个元素有两个数据域和一个指针域,分别存储了面片法向量、三顶点索引和邻接面片索引的指针三类信息。存储结构示意图如图 3 所示。

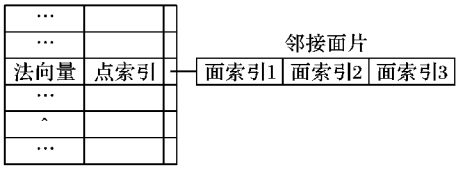


图 3　面表

在实现时创建一个 STL 数字模型的拓扑结构类 Face,保存了数字模型的顶点信息、法向量信息以及面片间的拓扑关系。类 Face 的主要数据结构的定义如下所示:

```
Class Face
{ typedef struct NodeHash                    //一个顶点的哈希结构
  { unsigned long index;
    NodeHash * next;
  };
  struct FaceStruct                          //面表结构体定义
  { sVector Normal;                          //法向量
    unsigned long v1, v2, v3;                //面片三顶点的索引
    vector <unsigned long> AdjFace;          //邻接面片
  };
  struct NodeStruct //点表中顶点的结构体定义
  { sVector Node;                            //顶点坐标
    vector <unsigned long> AdjFace;  //与该顶点邻接的面索引
  };
  vector <struct NodeHash> NodeHashList;     //顶点哈希表
  vector <struct NodeStruct> NodeList;       //点表
  vector <struct FaceStruct> FacetList;      //面片列表
  …
}
```

类 vector 是一个能够存放任意类型的动态数组类，类 sVector 为三维向量类，保存点的三个分量坐标及实现向量的基本运算。

3 拓扑信息快速重建算法

3.1 三角网格读取

通过前面的分析知道 STL 文件有 ASCII 码和二进制两种格式，必须能够区分它们并根据各自不同的格式特点读取文件。在实现文件的打开和关闭时用到了 C ++ 的 CFile 类的 Open( ) 和 Close( ) 函数，然后使用指定当前的读写位置函数 Seek( ) 和 Read( ) 来读取文件头信息，判断当前的 STL 文件是否二进制文件，以确定以何种方式读取数据。这部分程序的主要代码为：

```
CFile * myfile = new CFile( ); int FileLen, Nmu;
char Path[255]; bool stb = false;                    //变量定义
myfile -> Open( Path, CFile: : modeRead);    //打开指定路径文件
FileLen = myfile -> GetLength( );                    //得到文件长度
myfile -> Seek( 80, CFile: : begin);
myfile -> Read( &Num, 4);                    //读取面片个数到变量 Num
if ( ( 80 + 4 + Num * 50) == FileLen)        //测试是否二进制文件
  stb = true;                                        //是二进制文件
myfile -> Close( );
delete myfile;
```

这段程序完成后，将会通过判断变量 stb 的取值确定文件的类型，为 true 就是二进制文件。对于文本文件类型使用 fgets( ) 函数循环读取每一行数据，再利用 sscanf( ) 函数从每行数据中截取出数字的部分存储到相应的变量中。二进制文件格式非常固定，只要使用 Read( ) 函数读取相应字节的数据。

由于文本文件类型的文件中没有说明面片的个数，所以利用动态数组来存取面片的数据将非常方便，本文选择了 C ++ 标准库中的容器类 vector，可实现数据的动态高效存取。对于每一个面片信息使用自己定义的一个向量类 sVector，在该类中不仅可以存取三维顶点或法向量的 x、y、z 分量值，还可以对运算符进行重载，方便实现向量间的赋值、等于、加减、点乘和叉乘等运算。最后，把读取的数据根据文件的格式，按顺序把每一个面片的法矢向量和三个顶点的坐标值存储到动态数组 p 中，p 的类型为 vector < sVector > 。这样后面进行的拓扑重建就以 p 中存储的数据为对象。

3.2 哈希函数的建立

本文利用哈希表的目的是实现快速查找顶点，把顶点的坐标作为关键字，通过哈希函数计算出该顶点在点表中的位置。

通过反复实验，设置哈希表的理想长度 m 为：

$$m = s * 3/10 + 1$$

其中 s 为 STL 模型中顶点的数量。顶点关键值 k 及哈希函数 h 的计算如下：

$$\begin{cases} k = |v.x| \times 10^4/23 + |v.y| \times 10^4/19 + \\ \qquad |v.z| \times 10^4/17 \\ h(k) = k \bmod m \end{cases} \tag{1}$$

其中：v 为顶点坐标，$v.x$，$v.y$，$v.z$ 分别为各坐标轴分量；而且由于点坐标的精度较高，为了使哈希表中元素更好地散列分布，在计算 k 值时放大了一定的倍数。顶点哈希表中初始值都为空指针，在处理地址冲突时，具有相同哈希地址的顶点在点表中的索引放在后面的单链表中。

3.3 拓扑重建算法

算法在读取面片的过程中利用顶点哈希表区别是新插入点还是重复的旧点来消除顶点的冗余，因此在点表中不会出现重复的顶点；同时在点表中存储了点与面的邻接关系，在面表中存储了法向量以及面与面的邻接关系，有了这两种邻接关系还可以根据自己的需要非常方便地得出点与边、边与边之间的邻接关系。拓扑结构建立的算法步骤如下所示。

1）分别创建顶点哈希表 NodeHashList，点表 NodeList 和面表 FacetList。

2）按顺序读取一个面片 f_i。

3）利用式（1）计算出每个面片 f_i 的 3 个顶点的哈希地址。

4）依次在顶点哈希表 NodeHashList 中查找，若未找到（该哈希地址存储为空指针）表示是一个新点，转到 5）；若找到（该哈希地址存储为非空地址）表示是一个已存在的旧点，转到 6）。

5）把新点加入到点表 NodeList 中（点坐标和邻接面片索引号），且把新点的索引号加入到哈希表 NodeHashLis 中，转到 7）。

6）若是一个旧点时，把当前面片的索引号加入到点表中该顶点的邻接面片链中，转到 7）。

7）不管是新旧点，都把法向量、三顶点在点表中的索引号加入面表 FacetList 中，同时对邻接面片的邻接面片链中加入该面片的索引。

8）转到 2）直到所有面片都读完为止。

4 算法分析与实例测试

该算法实现了 STL 模型的拓扑重建，不仅消除了顶点的冗余，即使一个顶点可能属于多个面也保证每个顶点坐标只保存了一次，具有较低的空间代价，而且每个面每个顶点也只读取一次，具有很低的时间代价。

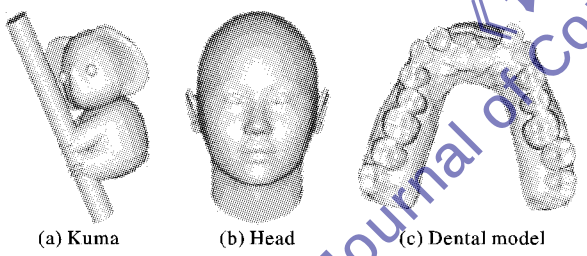
设计顶点哈希表的主要目的是消除原 STL 格式文件中顶点数据的冗余，使得每个顶点的三维坐标只在点表中出现

一次,如果每个顶点平均邻接 m 个三角形面片,则顶点坐标的内存占用将减少至未消除冗余顶点时的 $1/m$;并且顶点哈希表在完成点表和面表的建立后空间被释放,以后将不再占用内存。

假设一个三维模型有 n 个顶点,f 个面片,且每个顶点平均邻接 m 个三角形面片,则点表的空间代价为 $O(m+mf)$;面表中保存了所有的面片的法向量共 f 个,关联的 3 个顶点的索引号共 $3n$ 个,邻接面片的索引号平均共 $3f$ 个,则空间代价为 $O(3n+4f)$;表示拓扑结构所占用的空间代价为 $O(3n+(4+m)f+m)$,其中 m 的取值一般都是 10 以内的整数,总的空间代价是分别相对于顶点数和面片数的线性关系,因此该数据结构所占用的内存空间是完全可以接受的。

直接重建算法是建立在对 STL 模型的点表和边表进行顺序查找算法的基础上,完成拓扑关系重建的时间复杂度为 $O(n^2+e^2)$,其中:n 为 STL 模型中顶点的个数,e 为 STL 模型中边的个数。本文算法只需要对 STL 中的所有三角面片访问一次即可完成点表、面表等拓扑关系的建立,算法的时间复杂度为 $O(n)$,可以有效地降低计算总量。

本文在 1 台 Pentium Dual-Core 3.00 GHz CPU,1.87 GB 内存的计算机上分别用直接算法、文献[10]的红黑树法和本文算法进行了拓扑重建的实现,并测试 5 个面片数目比较典型的 STL 文件的实例模型(如图 4 所示),测试结果如表 1 所示。从表 1 可以看出本文所采用的算法对实现拓扑结构的重建效率非常高,对于比较少的面片模型不到 1 s 就能完成拓扑重建(其中 Kuma 模型是 ASCII 码格式的),对于 60 多万个面片的模型也只需要 2 s 多的时间。



(a) Kuma　(b) Head　(c) Dental model

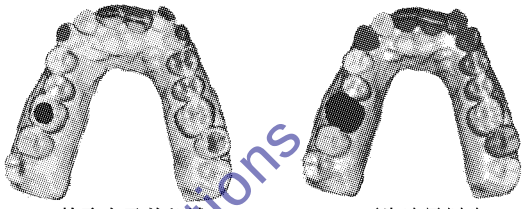
(d) Bird's nest　(e) Hand

图 4　测试用 STL 模型

表 1　三种算法对五个测试实例的运行时间对比

STL 模型	三角面片数	时间/s		
		直接算法	红黑树法	本文算法
Kuma	8 222	3.302	2.486	0.031
Head	12 280	20.875	3.145	0.032
Dental model	51 675	229.047	16.452	0.172
Bird's nest	102 058	1 020.047	20.378	0.344
Hand	654 666	7 590.797	104.604	2.344

建立 STL 模型的拓扑结构是对模型所包含的点、边和面几何信息进行快速查找,以及对模型进行变形调整、区域划分和曲面重建等后续处理的前提。利用本文建立的数据结构和拓扑重建算法,可以快速获得拾取点的一环、二环乃至 N 环邻域的信息,以及模型的边界等特征。本文所提供的方法已应用于作者自主开发的基于隐形牙套技术的牙齿矫正可视化系统,可帮助用户事先了解牙齿的纠正过程并且通过后续开发可利用该系统生成纠正过程中所需的多副牙套数据,据此数据可生产医用硅胶牙套产品用于患者治疗,对于图 4(c)中的牙颌模型,该系统采用本文的数据结构和拓扑重建算法去除重复顶点并建立拓扑结构,在此基础上计算拾取点的邻域信息以及模型的边界信息,结果如图 5 所示。其中,图 5(a)为鼠标在每颗牙齿上的拾取点并利用拓扑信息扩展分割邻域的过程图,图 5(b)为利用拓扑信息寻找到每颗牙齿边界并从牙颌模型中分离的结果图,这里采用不同的渲染颜色表现分离后的每颗牙齿和牙龈。



(a) 拾取点及其邻域　(b) 牙齿边界划分

图 5　拓扑重建算法应用实例

5　结语

快速重建三角形网格模型点边面的拓扑关系是对三角形网格模型进行分割和区域划分等后续处理的前提。STL 格式文件是记录三角网格模型的常用格式文件,但其中不含三角网格模型的拓扑信息,而且三角面片的顶点及边的信息被大量重复记录。本文针对 STL 格式文件记录三角网格模型信息的特点,设计了快速建立 STL 文件拓扑关系的数据结构,充分利用顶点与其邻接点和邻接面的关系,通过高效的搜索策略实现了拓扑信息的快速重建。根据该方法利用 Visual VC++ 6.0 和 OpenGL 三维平台开发了相关的软件功能,并成功应用于笔者自主开发的牙齿矫正可视化系统中。

参考文献:

[1] BÉCHET E, CUILLIERE J-C, TROCHU F. Generation of a finite element mesh from stereolithography (STL) files [J]. Computer-Aided Design, 2002, 34(1): 1-17

[2] STROUD J, XIROUCHAKIS P C. STL and extensions [J]. Advance in Engineering Software, 2000, 31(2): 83-95.

[3] JI F, LI Z. A novel method for the topological reconstruction based on STL files [J]. Journal of Hunan University of Science and Engineering, 2006, 27(5): 154-157. (纪峰, 李占利. 基于 STL 文件的拓扑重构新方法[J]. 湖南科技学院学报, 2006, 27(5): 154-157.)

[4] DUAN L, LIN H, WU C, *et al*. Method of automatic generation of STL files from industrial CT slicing image [J]. Computer Science, 2007, 34(1): 201-202. (段黎明, 林海, 吴朝明, 等. 工业 CT 断层图像自动生成 STL 文件的方法[J]. 计算机科学, 2007, 34(1): 201-202.)

[5] GONG Q, ZHANG L, MO J. The algorithm and application of automatic hollowing STL model [J]. Journal of Computer-Aided Design & Computer Graphics, 2007, 19(1): 54-58. (龚奇伟, 张李超, 莫健华. STL 模型自动镂空的算法与应用[J]. 计算机辅助设计与图形学学报, 2007, 19(1): 54-58.)

[6] LIANG Y, CHEN J, CHEN L, *et al*. Fast surface mesh generation

for STL models [J]. China Mechanical Engineering, 2010, 21(17): 2123 - 2127. (梁义, 陈建军, 陈立岗, 等. 基于 STL 模型的快速曲面网格生成[J]. 中国机械工程, 2010, 21(17): 2123 - 2127.)

[7] HOU B, CUI H, LIU X. Fast topological reconstruction algorithm for 3D mesh model [J]. Journal of Computer Applications, 2010, 30(11): 3002 - 3004. (侯宝明, 崔红霞, 刘雪娜. 三维网格模型的快速拓扑重建算法[J]. 计算机应用, 2010, 30(11): 3002 - 3004.)

[8] YANG S, DU Y, SHU S. Reconstruction algorithm of surface meshes based on STL files [J]. Computer Engineering, 2011, 37(4): 10 - 11. (杨晟院, 杜亚娟, 舒适. 基于 STL 文件的曲面网格重建算法[J]. 计算机工程, 2011, 37(4): 10 - 11.)

[9] LIU J, HOU B. Efficient topological reconstruction of solids in STL format [J]. Journal of Engineering Graphics, 2003, 24(4): 34 - 39. (刘金义, 侯宝明. STL 格式实体的快速拓扑重建[J]. 工程图学学报, 2003, 24(4): 34 - 39.)

[10] SHANGGUAN N, LIU B. Efficient topological reconstruction algorithm of STL model based on fast AVL tree [J]. Fujian Computer, 2008, 24(9): 10 - 11. (上官宁, 刘斌. 基于 AVL 树的 STL 模型快速拓扑重建算法[J]. 福建电脑, 2008, 24(9): 10 - 11.)

[11] TAN J, LI L. An algorithm for topology reconstruction from unorganized points based on local flatness of surface [J]. Journal of Software, 2002, 13(11): 2121 - 2126. (谭建荣, 李立新. 基于曲面局平特性的散乱数据拓扑重建算法[J]. 软件学报, 2002, 13(11): 2121 - 2126.)

[12] YANG L, SUN W, ZHAO Q, *et al.* 3D reconstruction of CT images of jaw and teeth's bones based on rapid prototyping [J]. Modular Machine Tool & Automatic Manufacturing Technique, 2007(11): 22 - 24. (杨莉玲, 孙文磊, 赵群, 等. 面向快速成型的颌骨及牙列 CT 图像的三维重建[J]. 组合机床与自动化加工技术, 2007(11): 22 - 24.)

[13] AN T, DAI N, LIAO W, *et al.* An efficient algorithm for topological reconstruction of STL data [J]. Mechanical Science and Technology, 2008, 27(8): 1031 - 1034. (安涛, 戴宁, 廖文和, 等. 基于红黑树的 STL 数据快速拓扑重建算法[J]. 机械科学与技术, 2008, 27(8): 1031 - 1034.)

[14] DAI N, LIAO W, CHEN C. Research on efficient algorithm of topological reconstruction for STL data [J]. Journal of Computer-Aided Design & Computer Graphics, 2005, 17(11): 2447 - 2452. (戴宁, 廖文和, 陈春美. STL 数据快速拓扑重建关键算法[J]. 计算机辅助设计与图形学学报, 2005, 17(11): 2447 - 2452.)

[15] WANG W, CHEN J, WANG Y. Research on the technology of three-dimensional surface topological reconstruction of salt caverns supported by 3D GIS [J]. Land and Resources Informatization, 2012(2): 62 - 67. (王文, 陈建忠, 王永志. 3D GIS 支持下的盐腔三维表面拓扑重建技术研究[J]. 国土资源信息化, 2012(2): 62 - 67.)

[16] SHI Y, SUN D, LI Y, *et al.* The topology reconstruction of unorganized point cloud based on the topology neighbors of sampling point [J]. Journal of Shandong University of Technology: Science and Technology, 2012, 26(2): 5 - 10. (史阳, 孙殿柱, 李延瑞, 等. 基于样点拓扑近邻的散乱点云曲面拓扑重建[J]. 山东理工大学学报: 自然科学版, 2012, 26(2): 5 - 10.)

[17] BAUMGART B G. A polyhedron representation for computer vision [C]// AFIPS '75: Proceedings of the 1975 National Computer Conference and Exposition. New York: ACM, 1975: 589 - 596.

(上接第 2710 页)

另外在收敛性方面,本文算法更易收敛,且收敛过程更稳定。

目前,由于本文算法处理的都是空间不变的模糊。对于空间变化的模糊,诸如旋转模糊等牵涉到坐标变换,因此其复原过程更复杂,并且坐标变换中会用到大量的近似,使得复原效果也会受到一定影响。所以本文下一步工作是将该算法进行拓展,使其能适用于空间变化的模糊。

参考文献:

[1] BISHOP T E, BABACAN S D, AMIZIC B, *et al.* Blind image deconvolution: problem formulation and existing approaches [M]// CAMPISI P, EGIAZARIAN K. Blind image deconvolution: theory and applications. Boca Raton: CRC Press, 2007.

[2] FERGUS R, SINGH B, HERTZMANN A, *et al.* Removing camera shake from a single photograph [J]. ACM Transactions on Graphics, 2006, 25(3): 787 - 794.

[3] WEN B, ZHANG Q, ZHANG J. Realization of iterative blind image restoration by self deconvolution and increment Wiener filter [J]. Optics and Precision Engineering, 2011, 19(12): 3049 - 3055. (温博, 张启衡, 张建林. 应用自解卷积和增量 Wiener 滤波实现迭代盲图像复原[J]. 光学精密工程, 2011, 19(12): 3049 - 3055.)

[4] MONEY J H, KANG S H. Total variation minimizing blind deconvolution with shock filter reference [J]. Image and Vision Computing, 2008, 26(2): 302 - 314.

[5] LIAO H, NG M K. Blind deconvolution using generalized cross-validation approach to regularization parameter estimation [J]. IEEE Transactions on Image Processing, 2011, 20(3): 670 - 680.

[6] LIKAS C L, GALATSANOS N P. A variational approach for Bayesian blind image deconvolution [J]. IEEE Transactions on Signal Processing, 2004, 52(8): 2222 - 2233.

[7] CHANTAS G, GALATSANOS N, LIKAS A, *et al.* Variational Bayesian image restoration based on a product of *t*-distributions image prior [J]. IEEE Transactions on Image Processing, 2008, 17(10): 1795 - 1805.

[8] BABACAN S D, MOLINA R, KATSAGGELOS A K. Variational Bayesian blind deconvolution using a total variation prior [J]. IEEE Transactions on Image Processing, 2009, 18(1): 12 - 26.

[9] XIAO S, HAN G. Image restoration algorithm based on sparse regularized optimization [J]. Journal of Computer Applications, 2012, 32(1): 261 - 263. (肖宿, 韩国强. 基于稀疏正则优化的图像复原算法[J]. 计算机应用, 2012, 32(1): 261 - 263.)

[10] OLIVEIRA J P, BIOUCAS-DIAS J M, FIGUEIREDO M A T. Adaptive total variation image deblurring: a majorization-minimization approach [J]. Signal Processing, 2009, 89(9): 1683 - 1693.

[11] FARAMARZI E, RAJAN D, CHRISTENSEN M. A unified blind method for multi-image super-resolution and single/multi-image blur deconvolution [J]. IEEE Transactions on Image Processing, 2013, 22(6): 2101 - 2114.

[12] BOX G E P, TIAO G C. Bayesian inference in statistical analysis [M]. Hoboken: John Wiley and Sons, 2011.

[13] XU Z. Research on the practical technique of image blind restoration [D]. Changsha: National University of Defense Technology, 2006: 45 - 49. (徐综琦. 图像盲复原实用技术研究[D]. 长沙: 国防科学技术大学, 2006: 45 - 49.)