文章编号：2096-1472(2018)-03-10-06
DOI:10.19644/j.cnki.issn2096-1472.2018.03.003

剖面数据场中的三维表面建模

王 铮
(太原工业学院计算机工程系，山西 太原 030008)

摘 要：在三维表面建模技术中，Marching Cubes算法是应用最为广泛的方法之一。该算法简单高效，但与此同时，研究人员也发现它存在一些不足。在构造等值面时，Marching Cubes算法要把所有体素全部检测一遍，即使有些体素没有和等值面相交，这影响了算法效率；此外在这个过程中，Marching Cubes算法还会忽略掉一些本来在等值面上的点，降低了表面重建的精度。针对这些问题，本文对算法进行了改进。在构造等值面时，不检测空的体素以提高算法的速度，并且把一些被忽略的等值点添加进来以提高算法的精度。

关键词：三维表面建模；Marching Cubes算法；算法效率；逼近精度
中图分类号：TP311 **文献标识码**：A

3D Surface Modeling in the Section Data Field

WANG Zheng
(*Department of Computer Engineering,Taiyuan Institute of Technology,Taiyuan* 030008,*China*)

Abstract: Marching Cubes algorithm is one of the most widely used methods in 3D surface modeling technologies. Although the algorithm is simple and efficient,researchers have found some shortcomings in it.When constructing the isosurface,the Marching Cubes algorithm detects all voxels,even if some voxels do not intersect with the isosurface,which affects the efficiency of the algorithm.In addition,the Marching Cubes algorithm ignores some points that are originally on the isosurface in this process,which reduces the accuracy of surface reconstruction.In order to solve these problems,this paper improves the algorithm.When constructing isosurface,it does not detect empty voxels to increase the efficiency of the algorithm,and adds some neglected equivalents to improve the accuracy of algorithm.

Keywords:3D surface modeling;Marching Cubes algorithm;algorithmic efficiency;approximation accuracy

1 引言(Introduction)

随着科学计算可视化和计算机硬件技术水平的发展，三维表面建模技术越来越多地应用于生活、工程和科研领域[1]。在三维表面建模技术中，等值面抽取方法的重建效果比较好[2]，而Marching Cubes是其中比较经典的算法[3]。在医学领域，医生需要在核磁共振图像(MRI)的帮助下，掌握患者身体内部的病变情况，对病人进行治疗[4,5]；在地质领域，地质专家需要通过地质勘测图像了解地层结构，指导矿藏开采或者预测地质灾害[6,7]。这些图像都可以由Marching Cubes算法重建获得。然而Marching Cubes算法也存在一些不足之处，并且不断有学者和研究人员进行研究和改进[8,9]。

本文以Marching Cubes算法为基础，对算法进行了一定的改进，提高了算法的效率和逼近精度。利用三维矿体剖面轮廓线数据，对三维矿体表面进行重建，还原了矿体原貌。在通过实验图像验证了算法可行性之后，对算法进行了时间复杂度分析，并且总结了算法的优缺点。

2 基本原理(The basic principle)

本文的基础数据是矿山剖面轮廓线数据，在进行三维表面建模之前，先要对此数据进行处理以生成体数据。生成体数据之后，用改进的Marching Cubes算法构造等值面，完成建模工作。主要步骤有两个：(1)构造体数据，这个步骤中排除了对空体素的检测；(2)生成等值面，这个过程中增加了等值点。

2.1 构造体数据

体数据是体素级表面建模的基础。本算法将每一层剖面都定义为一个$N_x \times N_y$的二维网格，所有剖面垂直叠加就形成一个规模为$N_x \times N_y \times N_z$的三维空间区域，算法过程就在这个区域中进行。

基金项目：2017年度太原工业学院青年科学基金项目《等值面提取算法研究》（项目编号：2017LQ05）.

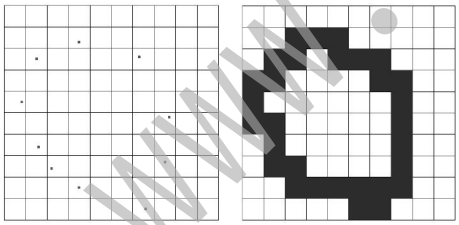
在一系列二维平面上，定义场函数为$f(x,y)$，对于平面上的某个坐标为(x,y)的网格点P的场函数：(1)如果该点在剖面轮廓线内部，其值为正；(2)如果该点位于剖面轮廓线的外部，其值为负；(3)如果该点落在剖面轮廓线上，其值为0。为了使得重建结果尽量平滑而且逼近物体真实表面，算法要选择合适的场函数，否则会出现明显的突变，造成绘制结果失真。本文采取欧氏距离函数作为场函数：

$$f(x,y)=\begin{cases} dis(x,y) & \text{如果}(x,y)\text{在全部轮廓线之内} \\ 0 & \text{如果}(x,y)\text{在轮廓线上} \\ -dis(x,y) & \text{如果}(x,y)\text{在全部轮廓线之外} \end{cases}$$

$dis(x,y)$表示在这个二维网格中，该网格点到该剖面上的轮廓线上点的最近距离。

在同样大小的三维空间区域中，体数据的密度会影响Marching Cubes算法重建结果的逼近精度[10]，因此在构造体数据的过程中，合理划分网格单元的大小对算法结果有直接的影响。在对算法处理时间和存储空间允许的情况下，尽量将网格划分的精细一些，可以得到更为精细的重建结果。

在Marching Cubes算法中，等值面与体素棱边的交点用线性插值求取。通常在计算场函数值时，要对所有体素端点都进行计算，但是实际上与等值面相交的体素比起没有与等值面相交的体素，数量要少很多[11]。本文是基于剖面数据进行表面重建，因此在剖面上考虑此问题，对原算法进行了改进。假设剖面是一个$N_x\times N_y$的二维网格，在某一个剖面上，与等值线相交的网格数量占整个网格的比例很小。



(a)等值点的分布　　(b)有等值线穿过的网格

图1 一个剖面中的等值点分布

Fig.1 Equivalent points distribution in a section

如图1所示，在一个剖面上，与等值线相交的网格数量所占比例很小，如果计算所有网格顶点的场函数值，无效计算很多，将会降低算法的效率。为了改善这一状况，提高算法的效率，需要尽量减少无效的计算。为此要设置一个标识符，一个表示各顶点状态值的数组，以及一个存放场函数值的数组，首先判断顶点与等值面的关系，若在等值线内(包括在等值线上)，则其状态值为1，反之状态值为0。当确定了一个网格四个顶点的状态值后，如果状态值全为0或者全为1，则说明该网格与等值面无交点，因此不需要计算其场函数值。步骤如下：

Step1：输入剖面数据；

Step2：将剖面数据网格化；

Step3：计算每个网格四个顶点的状态值，通过顶点状态值确定是否与等值面相交，计算与等值面相交的网格的四个顶点的场函数值；

Step4：逐个处理每个剖面，构造出体数据。

2.2 生成等值面

基于体素级的三维表面建模，即基于等值面生成的表面建模，需要从大量的体数据中把近似表示物体表面的等值面提取出来。在Marching Cubes算法中，在一个体素中等值面的提取过程如下：首先计算体素每个顶点的状态值，确定该体素是否是和等值面相交的体素；确定了体素和等值面的关系之后，对于和等值面相交的体素，计算其八个顶点的场函数值，然后根据已经给定的等值面阈值求取交点(即等值点)，求出体素内所有的交点后，把这些交点按照一定的方式连接起来，再进行三角剖分，就生成了该体素内的等值面。Marching Cubes算法的前提是数据场沿着体素棱边呈线性变化，求取等值点时根据其所在棱边两个端点的坐标进行插值即可求得。

体素的每个顶点都有两种状态，要么在等值面内，要么在等值面外，而8个顶点，则共有28种状态，也就是说根据顶点状态的不同，可以把体素类型分成256种，记录256种状态比较复杂，而且这些状态中很多是可以相互转换的，根据旋转对称性和互补对称性进行简化，只需要记录15种等值面连接构型[12]。在求出体素与等值面的交点后，再根据这15种等值面构型连接三角形组成最终的等值多边形。

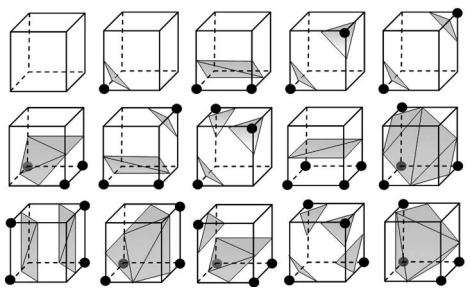


图2 15种基本体素构型

Fig.2 15 basic voxel configurations

将数据场内所有和等值面相交的体素处理完后，这些体素内部的等值面组合起来就构成了所要重现物体的表面网格模型，而可视化最终要将抽象数据转变为容易理解的图像信号，因此为了更好地逼真物体真实表面，需要对网格模型进行光照处理。处理方法为在体素内构造等值面的同时，也要计算每个剖分三角形三个顶点的法向量，然后求取平均值作

为该三角面片的法向量，利用此法向量就可以计算该面片上的光照强度，最终得到逼近真实物体的绘制结果[13]。

Marching Cubes算法在提取等值面时，通过插值计算出体素棱边与等值面的交点，然后再通过一定的方式把插值点连接起来形成多边形近似表示物体表面。在实际情况中，等值面和体素表面的交线是双曲线，Marching Cubes算法以连接插值点的方法，用直线段近似表示曲线段生成物体表面。在这个过程中，原算法并不考虑是否有原轮廓线上的点落入体素之中，因此插值后形成的等值轮廓线和原轮廓线之间存在一定的误差，而原轮廓线上的点是物体表面上的点，因此使用等值轮廓线提取出的等值面与物体原表面也存在一定的误差。当体数据场密度较大的时候，利用直线段来近似表示曲线段是可行的，因为当线段长度非常小时，直线段和曲线段的逼近程度会很高，小到一定程度的时候肉眼是无法分辨的。但是如果体数据场密度较小，插值求取交点后连接形成的直线段长度较大，此时其和实际的曲线段交线相比较，逼近程度就会差很多，这个逼近程度会随着体数据场密度的减小而降低。

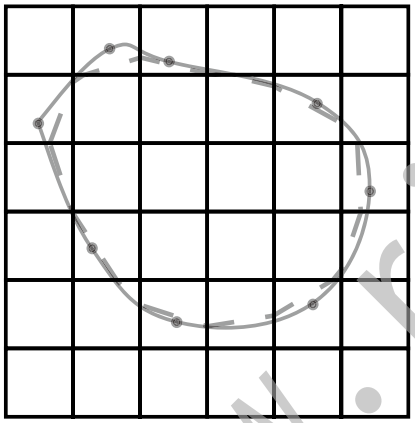


图3 插值轮廓线和原轮廓线的误差

Fig.3 The error between the interpolation contour and the original contour

如图3所示，网格中的闭合实曲线是原轮廓线，虚线是经过插值计算后连接插值点形成的等值轮廓线，实心黑点是原轮廓线上的数据点。从图中可以看到原轮廓线上的点不全在等值线上，这样就造成连接插值点形成的等值线和原轮廓线吻合程度较低。

为了降低逼近误差，本文在构造等值面时，除了计算体素棱边和等值面的交点之外，还考虑了原轮廓线上的点，对于有原轮廓线上的点落入的体素，将插值点和原轮廓线上的点以一定的方式连接起来，然后再通过三角剖分生成等值面。通过这一方法，可以使得剖面数据中原轮廓线上的点包含在等值线上，而且在连接的时候加入原轮廓线上的点，那么生成的等值线段就不再是是线段，而是由两条直线段组成的折线段。如图4为一个有原轮廓线点落入的网格中等值线的连接方式，曲线段P_1P_2是等值面和该体素表面的实际交线，直线段P_1P_2是传统Marching Cubes算法经过插值计算出交点坐标后连接形成的等值线段，Q是落入该表面上的原轮廓线上的点，连接的时候如果将点Q考虑在内，那么最终以折线段P_1QP_2作为等值线，很明显可以看到折线段P_1QP_2和点O围城的多边形面积更接近曲线段P_1P_2和点O围城的扇形面积。

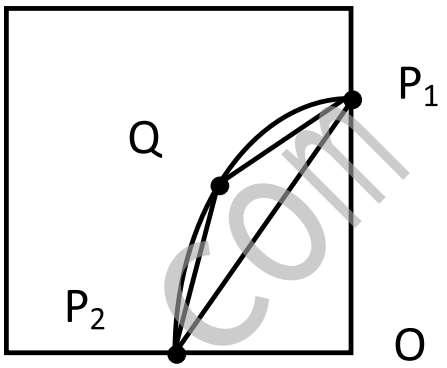
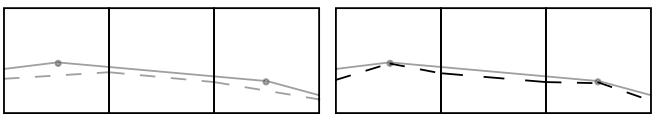


图4 不同连接方式中的逼近程度

Fig.4 Approximate degree in different connection methods

连接插值点形成等值线时，如果加入原轮廓线上的点，可能会改变原来的连接方式。对于体素的某个面来说，如果原轮廓线上的点落在该面的某条边上，那么计算插值点的时候，插值点刚好就是这个原轮廓线上的点，此时即使将该原轮廓线上的点考虑在内，也不会改变该面上插值点的连接方式。但是如果原轮廓线上的点完全落入该面内，那么就会改变插值点的连接方式。如图5所示，图a和图b分别是调整前后的等值点连接方式，调整后左边和右边两个网格内等值点的连接方式都发生了改变，中间的网格由于没有原轮廓线上的点落入，因此连接方式没有改变。



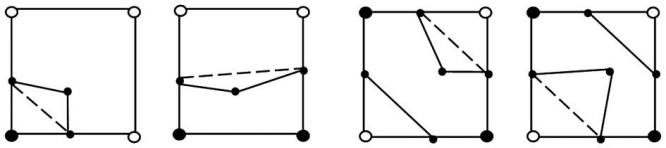
(a)不考虑轮廓线点的连接方式 (b)考虑轮廓线点的连接方式

图5 调整前后的连接方式(局部放大后)

Fig.5 Adjust the connection before and after (after partial enlargement)

根据算法前提假设，场函数沿着体素棱边呈线性变化，因此体素的一个表面四条边与等值面的交点数量有0、2和4这三种情况。当体素某个表面四条边与等值面的交点数量为0的时候，必定没有原轮廓线上的点落入；当交点数量为2的时候，如果有原轮廓线上的点落入，那么改变连接方式比较简单，只需要依次连接三个点即可。如果有四个交点，该面是二义性面，在连接的同时也要考虑解决面二义性问题，不同

的连接方式可能会生成不同的等值面构型，甚至可能会出现拓扑错误的情况[14]，比较经典的解决方法有四面体剖分法[15]和双曲线渐近线法[16]，不同的方法各有特点，本文采取了一种基于插值点连线交点的方法解决面二义性问题[17]。



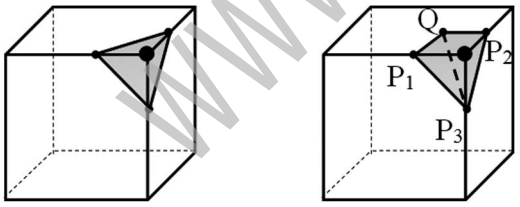
(a)两个交点的连接方式　　(b)四个交点(二义性面)的连接方式

图6 插值点连接方式

Fig.6 Interpolation point connection

体素级表面建模在提取等值面的过程中，是以直线段近似代替曲线，而此处将原轮廓线上的点考虑进来，实际上是以两段折线段近似代替曲线，逼近程度要比之前高一些。在图6(a)中可以看到，有两个交点的情况有两种，在每种情况中，考虑原轮廓线上的点之后，连接方式是唯一的，将原轮廓线上的点分别和两个交点连接起来即可。但是在二义性面上，因为内部等值直线段有两条，因此可能的连接方式有两种，如图6(b)所示，两种不同的连接方式会导致体素内的三角面片拓扑关系不同。在这种情况下，结合轮廓线和交点的关系，计算该轮廓线上点到两个直线段的距离，取距离较近的那个直线段为需要调整的直线段，因为距离近的直线段其上面点的场函数值更接近等值面阈值，这样更符合实际情况。

在调整了体素表面上的等值点连接方式之后，体素内部三角剖分形成等值面的方式也随着会发生改变，不能再完全按照之前介绍的15种基本构型来构造等值面了。但是如果全部重新调整等值面构型会增加较大的计算量，而调整插值点连接方式的面上，调整方式都是相对简单的，因此可以在原来15种构型的基础上进行改进。



(a)原等值面构型　　(b)调整后的等值面构型

图7 调整前后的等值面构型

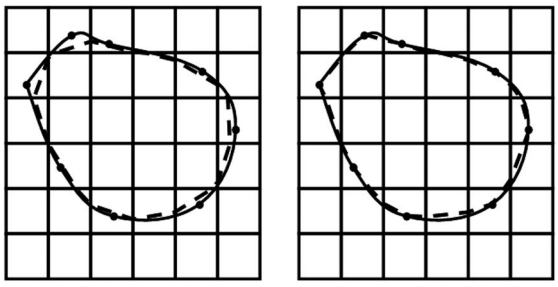
Fig.7 Adjust the isosurface configuration before and after

如图7所示，以Marching Cubes算法中的一种等值面构型为例。图(a)是原来的等值面构型，该体素中一个顶点在等值面内(外)，其余七个顶点都在等值面外(内)。按照经典Marching Cubes算法原理，在该体素中有三条棱边和等值面相交，插值计算出交点坐标后，连接这三个交点构成一个三角形面片，最终以该三角形面片作为该体素内的逼近等值面。但是当加入原轮廓线上的点之后，上表面的连接方式发生了改变，此时就不能再按照图(a)的方式构造等值面了。此时要在原来构型的基础上，如图7(b)所示，点P_1、P_2和P_3分别是插值计算出的交点，Q是原轮廓线上的点，做如下调整：首先确定该体素中插值点的连接方式，也就是说要按照传统Marching Cubes算法的方法确定等值面构型，然后通过点P_1和P_2找到原构型中和这两个点处于同一个三角形面片中的另外一个顶点P_3，之后分别连接QP_1、QP_2和QP_3，最后去掉P_1和P_2之间的连线，这样就完成了调整。由图可知加入原轮廓线上的点Q后，通过调整等值面构型，使用两个三角形面片

$\triangle QP_1P_3$和$\triangle QP_2P_3$代替了原来的三角面片$\triangle P_1P_2P_3$，这样就使得重建后的逼近精度有了一定的提高。

在原Marching Cubes算法15种基本构型中，有些体素构型中可能包含多个三角形等值面片，0号构型中没有三角形面片，1号构型中有1个三角形面片，2、3、4、8号构型中都有2个三角形面片，5、6、7号构型中包含三个三角形面片，9—14号构型中分别包含四个三角形面片。当体素中三角形面片数量大于等于2的时候，就可能出现有公共边的三角面片。但是通过观察这些构型，可以发现在所有存在公共边面片的体素中，公共边只在体素内部，而在体素表面上不存在包含于两个或两个以上三角形面片中的等值直线段。本文是基于剖面轮廓线数据进行三维表面重建，在构造体数据的时候，要将相邻两层剖面进行体素化，因此剖面上的原轮廓线数据只分布在体素的上下表面上，由于体素表面上的等值直线段不存在被几个三角形面片共用的情况，因此在连接插值点和原轮廓线上的点时，不需要考虑图7(b)中P_1P_2被几个三角形面片共用的情况。

经过上面的处理，在构造等值面的时候既考虑了体素棱边与等值面的交点，也考虑了原轮廓线上点的分布，因此在一个剖面上，最终形成的等值轮廓线与原轮廓线之间的误差相比之前有了一定的减小，一定程度上提高了算法的逼近精度，图8是对图3进行处理前后的对比，可以明显看到二者的区别。



(a)调整前的逼近程度　　(b)调整后的逼近程度

图8 逼近程度

Fig.8 Approximation

综合上述步骤，得出构造等值面的步骤如下：

Step1：读入一个体素数据；

Step2：根据体素八个顶点的场函数值插值计算棱边上的交点；

Step3：根据顶点状态索引值查找对应的边状态索引值，再根据边状态索引值查找该体素内的等值面构型，确定等值面连接方式；

Step4：确定该体素表面是否包含原轮廓线上的点，如果包含原轮廓线上的点，则执行Step5，否则执行Step8；

Step5：查找包含原轮廓线点Q的面，计算该面上插值求出的等值点个数n，如果n=4，则执行Step6，如果n=2则执行Step7；

Step6：按照Step3中已经确定的连接方式，计算点Q到该面上两条等值线段的距离，取距离较近的等值线段进行调整；

Step7：查找等值线段P_1P_2所在三角形面片的另外一个顶点P_3，分别连接QP_1、QP_2和QP_3，并且去掉等值线段P_1P_2；

Step8：算法结束。

2.3 基于剖面数据进行三维表面重建的过程

根据之前对算法的介绍，基于剖面数据的改进MC表面建模流程如下：

Step1：读入剖面数据；

Step2：将相邻两层剖面数据网格化，得到体素；

Step3：选中体素，求取体素顶点状态值，判断体素是否与等值面相交；

Step4：对于和等值面相交的体素，求取每个顶点的场函数值；

Step5：检查体素是否还有二义性面，对二义性面进行处理；

Step6：根据顶点状态值找到相应的边状态值，根据边状态值查找对应的等值面连接方式，对于有原轮廓线上的点落入的体素，对其中的等值面连接方式进行调整；

Step7：计算剖分三角形的法向量，计算光照，绘制该三角形对应的等值面片；

Step8：选择新的体素转入Step3继续处理，直到所有体素都处理完毕；

Step9：结束。

3 实验结果(Experimental result)

本文在对剖面数据进行体素化构造体数据的过程中，将剖面网格划分为规模大小为$N_x \times N_y$的网格，假设每个剖面上平均有m条矿体轮廓线剖面，每条轮廓线上有n个点。下面对算法效率和结果进行分析说明。

(1)算法的时间复杂度

算法主要时间用于构造体数据和生成等值面两部分，这两个部分中有大量的计算和比较。体数据的构造由判断体素顶点与等值面关系、计算顶点场函数值两部分组成。生成等值面时需要查询边状态表和等值面结构表。

在构造体数据时，计算量主要集中网格顶点状态值和场函数值的计算，计算顶点状态值的次数为$N_x \times N_y$，所以时间复杂度为$O(N_xN_y)$；计算场函数的次数为$4 \times m \times n$，由于$m<<n$，因此算法的复杂度为线性时间复杂度$O(N)$，二者相加得到构造体数据的总体时间复杂度：

$$O(N_xN_y)+O(N)=O(N_xN_y)$$

生成等值面的过程中，需要在每个体素内提取等值面，计算次数$N_x \times N_y$；而查询边状态表和等值面结构表的时间复杂度都是是线性的$O(N)$，因此二者相加得到生成等值面等值面的总体时间复杂度：

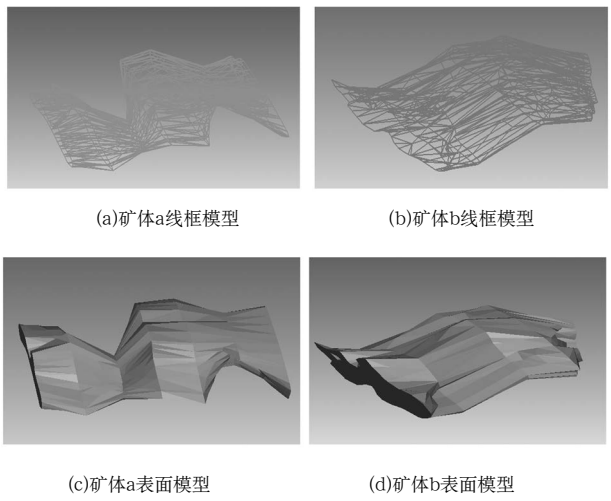
$$O(N_xN_y)+O(N)+O(N)=O(N_xN_y)$$

算法的总体时间复杂度由构造体数据和生成等值面两部分的时间复杂度相加所得，近似为$O(N_xN_y)$。

(2)算法实验效果

本文以矿体剖面轮廓线数据为实验数据，通过实验对算法进行了仿真和验证，编程环境为VS2010，结果如下。

使用本文算法对矿体剖面数据进行处理，正确重建了矿体的三维表面模型，消除了二义性带来的孔洞现象，效果如图9所示为两种矿体剖面数据重建后的线框模型和表面模型。



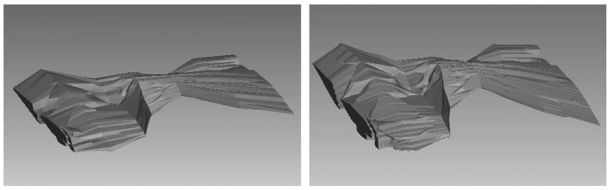
(a)矿体a线框模型 (b)矿体b线框模型

(c)矿体a表面模型 (d)矿体b表面模型

图9 矿体模型

Fig.9 Orebody model

在重建过程中，随着网格规模的增加，图形的逼近程度也会增加，但是由此产生的计算量增大，因此所耗时间也会增加。图10是对同一种矿体的剖面数据采取不同规模的网格重建后的结果。



(a)网格规模20×20　　(b)网格规模40×40

图10 不同规模的重建结果

Fig.10 Different scale reconstruction results

表1 结果比较

Tab.1 Comparison of results

图号	网格规模	时间消耗(s)
a	20×20	0.823
b	40×40	1.452

如表1所示，同样的剖面数据，采用不同的网格规模进行重建，当网格规模为20×20时所需时间为0.823s，当网格规模为40×40时所需时间为1.452s，因此要得到效果较好的重建图形并且尽量节省时间，需要合理选用网格规模。

4 结论(Conclusion)

本文在Marching Cubes算法的基础上，改进了原算法，不再对空体素顶点场函数值进行计算，提升了算法效率，并且通过增加交点的方式，提高了算法的逼近精度，此外也存在一些不足，归纳如下：

减少了无效的场函数值计算，传统Marching Cubes算法在寻找等值点时要计算所有的体素顶点，包括很多本来和等值面不相交的体素也进行了计算，本算法通过设置标志位只对和等值面相交的体素的顶点计算了场函数值，减少了无效计算，提升了算法效率。

在构造等值面的过程中，传统Marching Cubes算法只考虑插值计算求出的交点，将这些交点按照体素对应的等值面构型连接起来表示等值面。为了提高重建结果的逼近精度，本文在构造等值面时，将原轮廓线上的点和插值点一起连接起来，使得原轮廓线上的点包含于生成的等值面之中，并且由于使用折线段代替直线段近似表示等值线，使得算法重建的逼近精度有了一定的提高。

当数据场数据密度特别小时，没有解决逼近精度低的问题，传统Marching Cubes算法在较高密度的数据场中可以得到很好的表面绘制结果，但是随着数据场数据密度的降低，算法重建结果的逼近程度也随着降低，本文在数据场数据密度特别小时没有做相应的处理，将作为后续问题继续研究。

参考文献(References)

[1] Minho Chang,et al.Interactive marching cubes algorithm for intraoral scanners[J].The International Journal of Advanced Manufacturing Technology,2017,89(5-8):2053-2062.

[2] 黄伟.三维表面建模方法的研究与实现[D].长沙:中南大学,2010.

[3] William E.Lorensen,Harvey E.Cline.Marching cubes:A high resolution 3D surface construction algorithm[J].ACM SIGGRAPH Computer Graphics,1987,21(4):163-169.

[4] 于洋.肺部CT血管分割及三维重建[D].哈尔滨工业大学,2016.

[5] 姚翠萍,周湘连,王晶,等.纳米金标记细胞的荧光寿命成像及其三维重建[J]. 西安交通大学学报,2016,50(04):153-158.

[6] 刘致宁,宋承云,李志勇,等.基于多地震属性融合分割的地质异常体三维模型构建[J].Applied Geophysics,2016(3):519-528.

[7] 邹艳红,何建春.移动立方体算法的地质体三维空间形态模拟[J].测绘学报,2012,41(06):910-917.

[8] Seungki Kim.A Novel Interpolation Scheme for Dual Marching Cubes on Octree Volume Fraction Data[J].Computers & Graphics,2017,66(8):169-178.

[9] 毕硕本,陆源,曹晓文,等.Marching Cubes改进算法及其气象三维模拟[J]系统仿真学报,2017,29(07):1405-1410;1418.

[10] 孙伟,张彩明,杨兴强.Marching Cubes算法研究现状[J].计算机辅助设计与图形学学报,2007,19(7):947-952.

[11] 钱峰,马秀丽,杨胜齐,等.移动立方体算法的研究和改进[J].计算机工程与应用,2010,46(34):177-180.

[12] Adriano Lopes and Ken Brodlie.Improving the Robustness and Accuracy of the Marching Cubes Algorithm for Isosurface[J].IEEE Transactions on Visualization and Computer Graphics,2003,9(1):l6-29.

[13] 顾耀林,吕理伟.移动立方体算法中的三角剖分[J].计算机工程与设计,2006,27(1):120-123.

[14] 梁秀霞,张彩明.拓扑结构正确的三线性插值曲面的三角片逼近[J].计算机研究与发展,2006,43(3):528-535.

[15] Hummel R. Exploiting Triangulated Surface Extraction Using Tetrahedral Decomposition[M].IEEE Educational Activities Department,1995,1(4):328-342.

[16] Nielson G,Hamann B.The asymptotic decider:resolving the ambiguity in marching cubes[C].Proceedings of Visualization'91,Los Alamitos CA,1991:83-91;413.

[17] 王铮,李瑞明.移动立方体算法面二义性问题研究[J].软件工程,2017,20(09):6-8;5.

作者简介：

王　铮(1985-)，男，硕士，助教.研究领域：计算机图形学.