

基于自适应延迟切割的三角网格布尔运算优化*

姜旭东^{1,2}, 盛斌^{1,3}, 马利庄¹, 申瑞民¹, 吴恩华³



¹(上海交通大学 计算机科学与工程系, 上海 200240)

²(欧特克(中国)软件研发有限公司, 上海 200122)

³(计算机科学国家重点实验室(中国科学院 软件研究所), 北京 100190)

通讯作者: 盛斌, E-mail: shengbin@cs.sjtu.edu.cn

摘要: 规则化的布尔运算被广泛应用在三维建模系统中. 近年来, 随着图形硬件的发展, 基于三角网格的规则化布尔算法由于输出结果能直接被图形硬件处理, 表现出了明显的优势. 但是传统的算法由于采用 CSG 树局部评估策略, 使得面片在相交测试中反复被切割, 并且由于面片分类在切割后的模型之间直接进行, 导致算法无法在保证鲁棒性的同时实现高性能. 为了避免这些问题, 提出了一种 CSG 树全局评估算法来统一执行单次和连续布尔运算. 算法由两部分组成: 自适应的延迟切割和全局化面片分类. 在自适应的延迟切割阶段, 算法通过仔细处理多个三角面片相交导致的各种情况扩展延迟切割到整个 CSG 树来避免由于面片的反复切割带来的数值误差累积, 并利用自适应的八叉树使得相交测试可在线性时间内完成. 在全局化面片分类阶段, 算法通过分治法使得分类始终在切割后的面片和原始输入模型之间进行来保证分类的精度; 通过结合组分类策略和自适应的八叉树来进一步优化分类性能. 实验结果表明, 所提算法无论是在执行单次还是在连续布尔运算时, 都能在保证鲁棒性的同时性能优于其他算法, 因此该算法可广泛应用于交互式建模系统中, 如数字雕刻、计算机辅助设计和制造(CAD/CAM)等.

关键词: 布尔运算; 三角网格; 构造实体几何; 延迟切割; 自适应八叉树

中图法分类号: TP391

中文引用格式: 姜旭东, 盛斌, 马利庄, 申瑞民, 吴恩华. 基于自适应延迟切割的三角网格布尔运算优化. 软件学报, 2016, 27(10): 2473-2487. <http://www.jos.org.cn/1000-9825/5086.htm>

英文引用格式: Jiang XD, Sheng B, Ma LZ, Shen RM, Wu EH. Optimization of set operations on triangulated polyhedrons using adaptive lazy splitting. Ruan Jian Xue Bao/Journal of Software, 2016, 27(10): 2473-2487 (in Chinese). <http://www.jos.org.cn/1000-9825/5086.htm>

Optimization of Set Operations on Triangulated Polyhedrons Using Adaptive Lazy Splitting

JIANG Xu-Dong^{1,2}, SHENG Bin^{1,3}, MA Li-Zhuang¹, SHEN Rui-Min¹, WU En-Hua³

¹(Department of Computer Science and Engineering, Shanghai Jiaotong University, Shanghai 200240, China)

²(Autodesk (China) Software Research and Development Co., Ltd., Shanghai 200122, China)

³(State Key Laboratory of Computer Science (Institute of Software, The Chinese Academy of Sciences), Beijing 100190, China)

Abstract: Regularized Boolean operations have been widely used in 3D modeling systems. In recent years, Boolean algorithms based on triangular polyhedron show the distinct advantages aligning with the development of graphic hardware, as their outputs can be processed by graphic hardware directly. But most existing methods rely on localized evaluation strategy over constructive solid geometry (CSG) tree perform regularized set operations. As a result, these methods cannot guarantee robustness while synchronously keeping high efficiency, because a facet may repeatedly split up in the splitting phase and the facets classification is carried out between the split polyhedrons by

* 基金项目: 国家自然科学基金(61572316, 61133009); 国家高技术研究发展计划(863)(2015AA015904)

Foundation item: National Natural Science Foundation of China (61572316, 61133009); National High Technology Research and Development Program of China (863) (2015AA015904)

收稿时间: 2016-01-20; 修改时间: 2016-03-25; 采用时间: 2016-05-09; jos 在线出版时间: 2016-08-08

CNKI 网络优先出版: 2016-08-09 15:37:58, <http://www.cnki.net/kcms/detail/11.2560.TP.20160809.1537.009.html>

triangulation. In this paper, a novel algorithm is presented to realize robust, exact and fast regularized Boolean operations through global evaluation of CSG tree. The algorithm is comprised of two steps: adaptive lazy splitting and globalized facets classification. The two steps aim to optimize splitting and facets classification phases of the regularized Boolean algorithms on triangulated polyhedrons respectively. In the adaptive lazy splitting phase, a lazy splitting strategy is applied to the whole CSG tree by coping with all intersection cases of triangular facets in order to eliminate the accumulation of number errors. In the meantime, an adaptive octree is employed to speed up the intersection test process. In the globalized facets classification phase, to ensure the accuracy of classification, the classification method is always executed between the split facet and the original input polyhedrons by divide and conquer algorithm. The performance of classification is further optimized by combining the grouping classification strategy and the octree. Experimental results demonstrate that the proposed approach cannot only guarantee the robustness of Boolean computations but also achieve better performance than existing approaches. Thus, the algorithm offers wide-ranging usage in for interactive modeling systems, such as digital sculpture, and CAD/CAM.

Key words: Boolean operations; triangular polyhedron; constructive solid geometry; lazy splitting; adaptive octree

1 引言

三维几何模型的规则化布尔运算^[1]定义为两个模型内部集合运算的闭包.通过对多个三维几何模型反复执行规则化的并(union)、差(subtract)或交(intersect)操作,可以生成新的任意复杂的三维几何模型.因此,三维几何模型的规则化布尔运算被广泛应用在许多领域,如数字雕刻、机械零件设计及加工、建筑工程中建筑量的估算等.

构造实体几何实体(CSG)树是一棵由基本图元,如球、圆柱、立方体等以及布尔运算符组成的树^[2].其中基本图元组成叶子结点,布尔运算符组成中间结点.在实际应用中,CSG 树被广泛用来评估连续布尔运算,并且叶子结点可以由任意复杂的模型组成.

近几十年来,针对鲁棒性或者性能问题,许多不同的布尔算法被提出来.这些算法或者侧重于鲁棒性,或者侧重于算法的效率,但都无法同时保证这两点.在这些方法中,文献[3]试图使用任意精度技术并仔细处理各种几何退化现象来提高算法的鲁棒性,但是实践证明,这些算法都太慢,无法用于实时交互系统中.比如,最新的CGAL^[4]库比采用非鲁棒布尔算法的商业化软件 Maya^[5]慢 50 倍.为了克服这些缺点,文献[6,7]使用平面表示的空间二分树(BSP)来表示三维模型,从而避免在布尔运算过程中引入新的顶点并简化了各种几何相交和退化现象的处理,因此提高了单次布尔运算的鲁棒性和效率,但却仍然无法避免连续布尔运算过程中数值误差的累积.另外,这些方法需要在边界表示法(B-Rep)^[8]与平面表示的空间二分树之间转换,转换过程不但需要额外的时间,而且也会引入数值误差.为了避免模型转换所引入的时间消耗和数值误差,Feito 等研究者^[9]实现了基于三角网格的规则化布尔运算的算法,他们通过使用自适应的八叉树来加速面片的相交测试和分类,进一步提高了算法的性能.然而,该算法在每次面片相交时都需要执行面片的切割,导致面片数不断增加,数值误差也不断累积,特别是在复杂模型的连续布尔运算时,问题将更严重.

针对这些问题,本文提出了一种基于自适应延迟切割优化的三角网格快速布尔运算算法.该算法利用 CSG 树全局评估策略实现了对单次和连续布尔运算的统一处理.算法由两部分组成:自适应的延迟切割技术和全局化面片分类技术.在自适应的延迟切割阶段,算法构建自适应的八叉树并将结点分为 3 类:相交结点、内部结点和外部结点.通过限定相交测试在每个相交结点内,算法实现了在线性时间内完成面片的相交测试;通过仔细处理多个三角面片相交导致的线段相交或重叠情况,算法扩展延迟切割至整个 CSG 树.在全局化面片分类阶段,算法一方面利用分治法使得分类始终在切割后面片和原始输入模型之间进行,从而保证了分类的精度;另一方面利用组策略和自适应的八叉树实现了在线性时间内完成面片的分类.因此,通过使用统一的策略处理单次和连续布尔运算,本文提出的算法避免了单次或连续布尔运算中新的面片的不断增加和数值误差的不断累积,从而在保证鲁棒性的同时实现了高性能.

本文第 2 节介绍与本文算法相关的研究工作.第 3 节呈现本文算法的理论基础和主要框架.第 4 节详细阐述本文算法的实现细节.第 5 节通过实验详细比较本文算法与其他相关算法的差异,并给出实验结果分析.最后是本文的结论部分.

2 相关研究

规则化布尔运算已被研究了很多年,这些方法可以分为 3 类:体素算法、近似算法、精确算法。

通过空间采样将三维几何模型转换成离散的体素表示,规则化布尔运算更容易实现并且更鲁棒,比如自适应的距离场方法^[10]或者基于水平集(level-set)的算法^[11]。然而,体素表示的模型精度受到体素大小的限制,采用更精细的体素又将导致存储空间和采样时间的上升。另外,在把模型从边界表示转换到体素表示的过程中,采样带宽的限制使得模型的几何细节被丢失掉。即使运用过采样技术,由于重构模型的表面向量偏离原来模型的表面向量,模型边缘的锯齿效果仍然不能完全消除^[12]。因此,文献[12,13]通过编码将模型的表面向量输入进采样过程实现了重构模型几何细节的能力。在文献[13]的基础上,一些研究者通过扩展双轮廓算法来达到保留最终模型的拓扑细节^[14-16]或流形细节^[17]的目的。然而,所有基于体素表示的布尔算法都需要将输入模型从边界表示转换成体素表示,在得到布尔运算的结果后,再从结果中抽取出表面。这个过程不可避免地导致模型几何细节以及模型精度的损失。

因为在精确的布尔算法中,处理面与面相交的各种情况是最复杂的问题,一些研究者试图通过近似布尔算法来避免此类问题。Simth 和 Dodgson^[18]提出了一种基于边界表示近似布尔算法,该方法由相互依赖的 6 个层次的几何操作集组成,算法利用随机扰动技术^[19]处理几何退化问题。但是,当输入模型存在着退化或接近退化的几何数据时,布尔运算的结果将破坏模型表面的拓扑一致性。另外,由于层次之间存在着相互依赖性,算法很难充分利用现代处理器的并行计算能力。直到最近,Wang^[20]将三角网格的相交区域转换成分层深度图像(LDI)^[21]进行相交处理,然后利用自适应轮廓生成算法产生近似的几何面片。该方法既达到了与基于体素的布尔算法同样的鲁棒性,又保持了模型在相交区域周围的几何细节。然而自适应的轮廓算法可能导致最终生成的三角网格出现自相交区域,从而破坏模型的拓扑一致性。另外,由于 GPU 硬件的限制,该算法无法处理 LDI 层数超过 256 的复杂模型。

在精确的布尔算法中,基于空间二分树(BSP)的算法具有独特的优势,这类算法能够处理非闭合的流形或非流形模型,并且能够避免处理面片相交的各种退化现象。Naylor 和 Thibault^[22-24]首先将输入模型表示成 BSP 树,然后通过合并生成的 BSP 树实现了规则化布尔运算。然而在 BSP 树合并过程中,由于面片被超平面反复切割使得数值误差被迅速累积,大大降低了算法的鲁棒性。因此,Berstein 和 Fussell^[6]在基于平面的 BSP 树算法的基础上利用自适应的几何断言技术^[25]实现了鲁棒、精确的规则化布尔算法。然而,在将模型从边界表示转换成平面表示时,算法需要使用网格修补算法^[26]保证拓扑一致性。这使得算法花费了大量的预处理时间,另外,由于两棵 BSP 树合并的时间复杂度接近 $O(n^2)$,从而使算法无法用于大规模场景中。为了避免这些问题的出现,Campen 和 Kobbelt 在 Berstein 等人的基础上通过引入自适应的八叉树实现了局部化的 BSP 树布尔算法^[7]。然而,该算法需要沿着包围盒切割平面表示的面片,当包围盒平面和八叉树结点中面片的平面接近平行时,误差迅速增大,可能导致最终结果出现破洞或裂缝。所有 BSP 树算法都需要额外的转换过程,这种转换不仅引入数值误差,也导致算法性能的下降,特别是在连续布尔运算的场景中,频繁的转换和 BSP 树的合并操作使得这类算法很难适用于实时应用。

在几何建模系统中,边界表示是最普遍使用的模型表示方法。因此,许多布尔算法的研究围绕如何直接在边界表示的基础上实现布尔运算而展开。在这些算法中,Hubbard^[27]引入了基于三角网格的全局化相交策略的布尔算法。利用全局化相交策略,面片仅在所有相交测试完成后才被切割,因此该算法有效地避免了单次布尔运算中数值误差的累积。然而,全局化相交策略无法应用到连续布尔运算中,因此无法解决连续布尔运算中数值误差的累积的问题。另外,该算法需要合并共面的三角形,此过程不仅引入了数值误差,也增加了处理时间。直到最近,Feito 等人在边界表示布尔算法^[27-30]的基础上实现了新的基于三角网格的规则化布尔运算^[9]。算法采用逐步构建自适应八叉树的策略并且利用多线程技术加速八叉树的查询,从而明显提高了性能,降低了系统内存消耗。然而,每当输入模型的一对面片相交时,算法需要重新三角化该面片对。这使得数值误差不断被累积,在执行复杂场景的连续布尔运算时,可能导致算法崩溃或三角化模型的拓扑不一致。由于该算法依赖三角化的模型进行分类,拓扑不一致将导致算法将无法产生正确的分类结果。另外,该算法在面片分类阶段需要利用面片的邻接信

息,因此,对于某些模型需要额外的预处理过程来建立这样的信息,从而增加了系统的处理时间.

3 系统概览

任意给定两个三角网格 P_i 和 P_j , 它们之间的规则化布尔运算被定义为根据下列公式对沿相交线切割后边界面片的选择过程^[31].

$$\begin{aligned}
 P_i \cup^* P_j &= \{F_i \text{ Out } P_j\} \cup \{F_j \text{ Out } P_i\} \cup \{F_i \text{ With } P_j\}, \\
 P_i \cap^* P_j &= \{F_i \text{ In } P_j\} \cup \{F_j \text{ In } P_i\} \cup \{F_i \text{ With } P_j\}, \\
 P_i -^* P_j &= \{F_i \text{ Out } P_j\} \cup \{F_j \text{ In } P_i\} \cup \{F_i \text{ Anti } P_j\},
 \end{aligned}$$

其中, F_x 是 P_x 切割后的边界表面集, 分类集 $F_x \text{ Out } P_y, F_x \text{ In } P_y, F_x \text{ With } P_y,$ and $F_x \text{ Anti } P_y,$ 分别表示在三角网格 P_y 外部、内部、共面并且法向量相同、共面但法向量相反的面片集. 进一步地, P_x 的边界表面可以被归并成 3 个不相交的子集, $P_x = \{S_{in}, S_{out}, \text{ and } S_{intersected}\}, S_{in} \cap S_{out} = S_{in} \cap S_{intersected} = S_{out} \cap S_{intersected} = \emptyset,$ 其中, $S_{in}, S_{out},$ and $S_{intersected}$ 分别为在 P_y 的最小轴对齐包围盒(AABB)^[32]内部、外部以及和 AABB 的面片的集合. 因此, F_x 能够从分类集 $S_{in}, S_{out}, S_{intersected}$ 中产生. 也就是说,

$$F_x = F_{x1} \cup F_{x2} \cup F_{x3}, F_{x1} \subset S_{in}, F_{x2} \subset S_{out}, F_{x3} \subset S_{intersected}.$$

比如, $F_x \text{ In } P_y$ 中的面片是对 S_{in} 分类的结果和 $S_{intersected}$ 中面片执行切割、分类后结果的并. 因此, P_i 和 P_j 之间规则化的布尔运算能够被分解为 3 个子过程.

- 利用表 1 的表达式简化规则从每个网格的面片集 S_{out} 得到 F_{x2} , 这是因为 S_{out} 中的所有面片完全在其他模型外部.
- 通过对每个网格的面片集 S_{in} 中面片分类结果的收集得到 F_{x1} , 这是因为该集中面片或者完全在其他模型外部, 或者完全在其他模型内部.
- 通过对每个网格的面片集 $S_{intersected}$ 中面片切割、分类后得到 F_{x3} .

这种策略可扩展到对多个模型的连续布尔运算评估中. 任意给定一棵 CSG 树 T 和一系列三角网格 $P_{i=(1, \dots, n)}, P_i$ 构成了 T 的叶子结点. T 能够分解成 3 棵不相交的子树的并, 也就是说: $T = T_{in} \cup T_{out} \cup T_{intersected}$. 其中, $T_{in}, T_{out}, T_{intersected}$ 是分别由来自输入网格的面片集合 $S_{in}, S_{out}, S_{intersected}$ 构成的 CSG 子树. 当集合 S 为空集时, 对应子树的一个叶结点包含一个空的面片集合. 因此, 多模型的连续规则化布尔运算的评估能够表示为对所有子树评估结果的收集. 图 1 显示了一棵 CSG 树分解的例子.

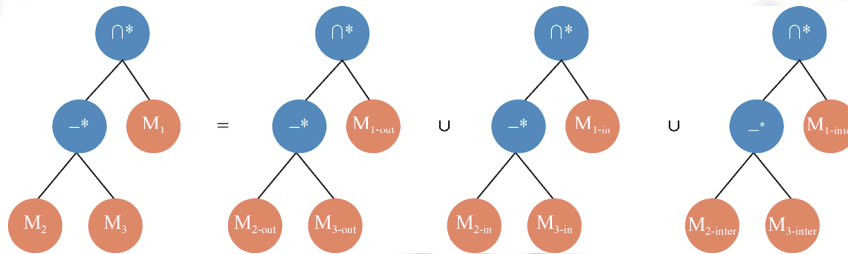


Fig.1 Shows an example of CSG decomposition

图 1 一个 CSG 树分解例子

通过部署自适应的八叉树, CSG 树 T 能够有效分解为一系列子树. 因此, 我们的算法由 4 步组成.

- CSG 树构建

算法首先转换输入布尔表达式为 CSG 树, 其中叶子结点是输入的三角网格, 中间结点具体的布尔操作符. 在单次布尔运算时, CSG 树仍被构建, 但此时仅有 3 个结点.

- 自适应的延迟切割

算法利用自适应的八叉树对所有 CSG 树叶子结点网格进行空间划分, 使得最终生成的八叉树包含 3 类叶

结点:相交结点、内部结点、外部结点(见第 4.1.1 节).这 3 类结点中的面片构成 3 类不同的子树,从而可以应用不同的策略处理以加速算法的执行.比如:相交区域内的面片构成的子树将参与相交测试和面片分类;内部区域的面片构成的子树仅参与面片分类以便精确确定和其他模型的关系.传统的延迟切割技术局限于两个模型之间^[30],而本文将延迟切割策略应用于由相交结点中面片构成的 CSG 子树.算法对每个相交结点内的面片按照所属模型分组,然后对属于不同组的三角面片两两执行相交测试,和每个三角面片相交的线段被加入到该面片对应的列表中.由于一个面片可能和多个面片相交,相交线段加入列表前需要和列表中已有的线段进行比较和处理以保证列表中始终不存在相交或相互重叠的线段,这个过程一直被执行直到所有相交结点被处理完为止.此时,每个三角形相交线段列表中的线段集合定义了不同模型之间分段线性的相交曲线,因此需要沿着这些相交线段定义的边界一次性切割面片.在计算几何中,面片的切割是一个带约束的三角化问题,其中相交线段定义了三角化问题中的约束边.

- 全局化面片分类

当八叉树所有相交结点中的面片完成切割后,这些面片构成一棵 CSG 子树,而所有内部结点中的面片构成另一棵 CSG 子树.此时所有面片与其他模型不相交,它们或者完全在其他模型内、或者完全在其他模型外.因此,点和多面体关系的测试算法可被应用以精确确定每个面片相对于其他模型的关系.然而,传统的分类算法是在切割后的模型之间直接进行的,不但严重影响分类的精度,而且错误的分类结果将向上传播导致算法失败.为了克服这些缺点,我们利用分治法使得分类始终在切割后面片和原始输入模型之间进行,从而保证了分类的鲁棒性.本文将在第 4.2 节详细叙述全局化面片分类技术.同时,本文从两方面进一步优化了分类算法的性能:(1) 利用组策略加速分类.由于八叉树中同一内部结点中的面片来自同一模型并且具有相同的分类关系(本文将在第 4.1.1 节阐述详细的证明),每个内部结点仅需任选一个面片参与测试;(2) 利用基于八叉树的光线投射和参数化遍历方法优化点和多面体关系的判定.这些措施使得面片分类算法可以在线性时间内完成,大大提高了算法的效率.

- 布尔运算结果生成

这一步的目的是通过收集每棵子树的评估结果生成最终布尔运算的结果.此时,八叉树相交结点面片构成的子树和内部结点面片构成的子树已被评估完成,这一阶段仅需评估由外部结点面片构成的 CSG 子树.给定该 CSG 子树的任意一个中间结点及其左右子结点,由其组成的布尔表达式 $A \text{ op } B$ 的评估结果可以利用表 1 的评估规则快速得到.其中, A 和 B 表示分别对左右子结点评估得到的面片集合, op 表示规则化布尔运算符.例如, $A \cup B$ 的结果为 A 和 B 中面片构成的集合,这是因为, A 或 B 中面片完全在其他模型外部.

Table 1 Evaluation rules for a CSG tree that only contains facets from all external nodes of an octree

表 1 八叉树外部结点中面片构成的 CSG 子树评估规则

运算符	左操作数	右操作数	结果
\cup^*	A^*	B	$A \cup B$
\cap^*	A	B	\emptyset
$-^*$	A	B	A

注: A 和 B 表示对 CSG 树任一中间结点的左右子结点评估得到的面片集合

4 CSG 树全局评估算法

4.1 自适应的延迟切割

延迟切割算法首先被 Hubbard^[27]用于改进三角网格布尔运算的鲁棒性和性能.然而该算法仅限于处理两个模型之间的布尔运算,无法应用于更多模型的连续布尔运算中.为了克服这一问题,本文实现了一种新的自适应延迟切割算法.该算法通过对三角面片之间相交线段的仔细处理,扩展延迟切割策略至整个 CSG 树,从而对单次和连续布尔运算提供了统一的处理方法.利用自适应的八叉树,算法保证了在线性时间内完成面片的相交测试和切割.该算法由 3 个阶段组成:自适应八叉树构建、相交信息记录、面片切割.

4.1.1 自适应八叉树构建

为了提高布尔运算性能,近年来许多研究者通过部署自适应八叉树快速确定输入模型的相交区域^[7,9,20].不同于这些算法中提出的自适应八叉树构建方法,本文构建的自适应八叉树允许同一面片被加入多个结点,因而避免了在构建过程中由于切割面片导致的数值误差累积.同时,本文在构建过程中将八叉树的每个结点分类为以下3种类型之一.

- 相交结点:结点包含两个以上模型的面片.
- 内部结点:结点内所有面片都属于同一模型并且在其他模型的轴对齐包围盒内.
- 外部结点:结点内所有面片都属于同一模型并且在其他模型的轴对齐包围盒外.

所有输入模型的面片按空间关系分布在八叉树的叶结点中,每个叶结点中的面片或者完全包含在该结点对应的轴对齐包围盒中,或者与该包围盒相交,因此,一个面片可能被多个叶结点共享.为了管理八叉树中面片的共享,算法定义了不同结点类型之间的优先级:相交结点的优先级高于内部结点,而内部结点的优先级高于外部结点.这意味着,当一个面片同时被不同类型的结点共享时,该面片将被当作属于高优先级结点并在遍历该结点时被处理,比如:如果一个面片被相交结点和外部结点共享,该面片将在相交结点内处理.为了确定每个共享的面片的归属,这些面片同样被设置为以下3种标志之一.

- 相交:面片至少被一个相交结点共享.
- 内部:面片被至少一个内部结点共享但不被相交结点共享.
- 外部:面片仅与多个外部结点相交.

这些标志定义了八叉树中面片的遍历规则:在遍历外部结点时,所有含有相交或内部标志的面片将被跳过,这些面片将分别在被共享的相交结点或内部结点中被遍历;在遍历内部结点时将跳过所有含有相交标志的面片,这些面片将在被共享的相交结点中被遍历到.图2显示了在二维空间对两个多边形利用自适应的八叉树进行空间剖分的例子.

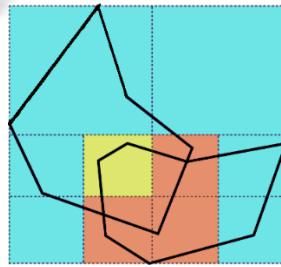


Fig.2 2D illustration of subdividing two polygons by adaptive octree.

Cyan denotes exclusive cells; orange denotes intersected cells; and yellow denotes inclusive cell

图2 利用自适应的八叉树对两个多边形进行空间剖分的例子.

其中,青色单元是外部结点、橙色是相交结点、黄色是内部结点

以这种方式构建的自适应八叉树具有以下性质.

- 1) 自适应八叉树的内部结点和外部结点只可能是叶结点,仅相交结点可能含有子结点.

证明:在自适应八叉树的构建过程中,仅当一个结点是相交结点并且结点中的面片数超过指定值时,才对该结点递归分割,因此内部结点和外部结点的父结点只可能是相交结点并且该结点不可能再被划分. □

- 2) 自适应八叉树内部结点中的所有面片具有完全相同的关系,它们或者完全在其他模型外部、或者完全在其他模型内部、或者和其他模型共面.

证明:根据定义和性质1,一个内部结点是叶子结点并且结点内所有面片属于同一模型,假设结点内至少有一个面片在其他模型外部而其余面片在该模型内部,则这些面片将穿越相交模型的边界,因此相交模型中必定有一些面片在该结点中,这与内部结点的定义不符,因此性质2成立. □

3) 自适应八叉树中仅相交结点内的面片可能和其他模型相交.

证明:假设八叉树中内部结点和外部结点存在一个面片和其他模型相交,则或者该结点包含来自其他模型的面片,或者该面片是被一个相交结点所共享.第 1 种情况下,该结点属于相交结点,与假设不符;第 2 种情况下,该面片将被相交结点处理,因此性质 3 成立. \square

4) 输入模型的相交测试被限定在每个相交结点内.

证明:根据性质 3,仅需考虑相交结点.任意给定一个相交结点 C_i ,如果 C_i 中不存在共享的面片,则结论成立;设 C_i 存在一个面片 a ,若 a 仅与 C_i 中的面片相交但被至少一个非相交结点共享,当构建八叉树时, a 已被设置为相交标志,因此 a 仅在 C_i 中被处理;若只有 a 不仅与 C_i 中的面片相交而且与 C_j 中面片相交,如图 3 所示,则当遍历 C_i 时, a 将与 C_i 中面片执行相交测试,当算法遍历 C_j 时, a 将再次和 C_j 中面片执行相交测试,因此结论仍然是成立的.证毕. \square

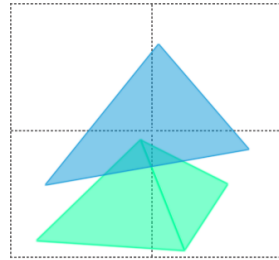


Fig.3 An example of intersection between a triangle and two triangles from two different intersected cells

图 3 一个面片分别和两个相交结点内三角面片相交的例子

以这种方式构建的八叉树隐含地将每个输入模型的表面划分为 3 种不同类型的区域:相交区域、内部区域和外部区域,分别对应所有相交结点、内部结点和外部结点中属于该模型的面片集合,图 4 显示了用不同颜色表示不同区域的模型例子.属于不同区域的面片构成不同类型的 CSG 子树,因而算法可以应用不同的策略评估这些子树以加速算法的执行.概要地说,算法仅需对相交区域的面片进行相交测试及切割,对切割后的面片构建 CSG 子树进行分类,而对由内部区域面片构成的 CSG 子树直接分类,对由外部区域的面片构成的 CSG 子树直接评估,本节余下部分将详细介绍这些内容.

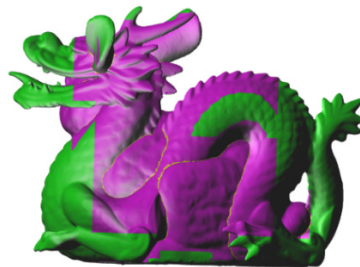


Fig.4 An example of different regions on each polyhedron constructed by an adaptive octree.

Green, pink and yellow denote the external, internal, intersected region respectively

图 4 利用自适应八叉树对两个模型划分生成的不同区域的例子.

其中,绿色、粉色和黄色分别表示外部、内部和相交区域

4.1.2 相交信息记录

规则化布尔运算的相交处理需要改变输入模型的拓扑结构,这个过程的关键问题是如何在线性时间内实现面片的相交测试;如何避免面片的反复切割引起的鲁棒性问题.根据八叉树性质 4,算法将相交测试限定在八叉树每个相交结点内,从而实现了在线性时间内完成所有面片的相交测试和切割.第 2 个问题是通过扩展延迟

切割策略至整个 CSG 树来解决的.

文献[27]提出的延迟切割策略仅限于处理两个模型的相交,这是因为输入模型的三角面片是可定向的二维流形表面,同一模型的任意两个面片不存在相互重叠的情况,因此,当一个面片和另一个模型的多个面片相交时,多个相交线段之间不可能重叠或交叉.而当多于两个模型相交时,相交线段之间存在着多种相交或重叠的可能性,算法必须处理所有这些情况.这也是本文算法能扩展至整个 CSG 树从而保证单次和连续布尔运算鲁棒性的关键.另外,该方法将相邻的共面三角形合并为多边形,在每个多边形内部属于原来三角形的边不被作为切割的约束条件,从而减少在切割阶段生成三角形的数量.然而,我们的研究发现,在大量复杂的模型中,相邻面片的法向量经常是接近平行但不完全平行,如果合并共面的三角形,一方面将引入新的数值误差从而可能导致切割算法失败;另一方面,合并后的多边形可能是凹多边形,从而进一步增加切割算法的复杂性.因此,本文的算法并不合并共面的三角形,而是通过直接切割原来的三角形保证了算法的鲁棒性和性能.

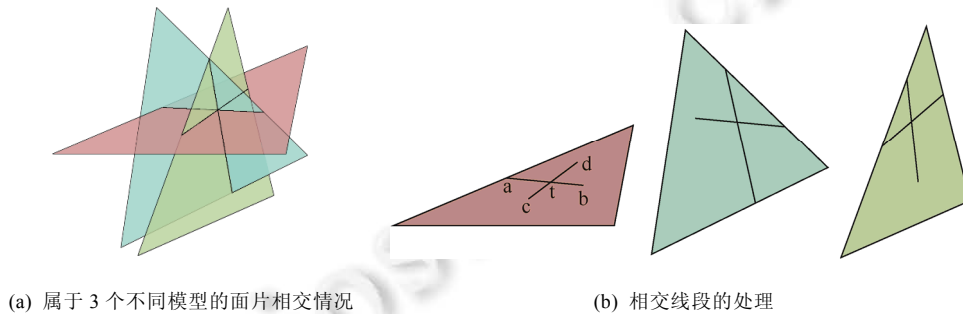


Fig.5 An example of triangular facets intersection and handling

图 5 多个面片相交及其相交线段处理的例子

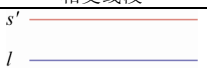






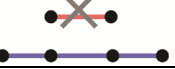
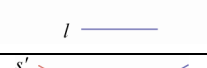



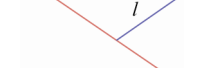



任意给定一个相交结点 C_i , 算法首先对 C_i 中每个面片按照所属网格两两分组形成若干二元面片集的关系对. 对于每个二元关系对 (T_i, T_j) 中的两个三角面片 a 和 b , 其中 $a \in T_i, b \in T_j$, 算法测试 a 和 b 的轴对齐包围盒是否相交, 若不相交, 则测试其他面片; 若相交, 则利用文献[33]提出的方法执行三角形和三角形之间测试. 当 a 和 b 相交时, 算法将首先检测 a 和 b 是否存在相关联的用于保存二维相交线段的列表 L_a 和 L_b , 如果不存在, 则创建相关的列表, 并将 a 和 b 的边转换到二维模型空间后分别加入到对应的列表中. 然后算法处理 a 和 b 相交线段集中的每根线段 s (相交于一个点的情况被忽略以避免产生非规则的运算结果). 由于 L_a 或 L_b 中线段将作为切割阶段的约束边, 这些约束边不能交叉或重叠. 而当评估连续布尔运算时, a 可能和多个模型的面片相交, 此时 s 可能和列表中的已有线段相交或重叠. 因此, s 被分别转换到 a 和 b 的模型空间后必须与 L_a 和 L_b 中所有线段进行相交测试并消除相交或重叠情况才能保证算法的正确性.

表 2 显示了两根线段相交或重叠的各种情况以及处理方法, 其中, s' 为 s 在 a 或 b 模型空间的表示, l 为对应列表中线段. 图 5 则显示了 a 分别和属于不同模型的面片 b, c 相交的例子. 在该例中, a 和 b 的相交线段 l_{ab} 已经存在于列表 L_a 中, 当 a 和 c 相交测试时, 由于线段 l_{cd} 和 l_{ab} 相交于非端点 t , 根据表 2 必须将 l_{ac} 和 l_{cb} 分别替换为 $l_{at}, l_{tb}, l_{ct}, l_{td}$. 算法对每个相交结点执行类似的操作, 直到完成八叉树所有相交结点的处理.

由于采用延迟切割策略, 相交结点内的面片数量在布尔运算过程中保持不变, 因此相交测试的时间降低到 $O(\sqrt{n \times k})$, 其中, n 是输入模型的数量, k 是一个网格中的平均三角面片数. 这是由于, 每个相交结点中面片 A 的数量不超过指定值且不随模型数量的增加而增加, 因而结点内相交测试的时间复杂度为 $O(1)$, 当输入模型的面片是均匀分布时, 构建的八叉树平均有 $\sqrt{n \times k}$ 个相交结点.

Table 2 An example of two segments intersection or overlapping processing

表 2 两根线段相交或重叠的各种情况及处理

相交线段	处理结果	说明
		s' 和 l 完全重合,线段 s' 被丢弃
		s' 和 l 部分重叠, s' 中与 l 重叠部分被丢弃, l 被分为两段
		s' 和 l 部分重叠, s' 中与 l 重叠部分被丢弃, l 被分为两段
		s' 完全包含在 l 中, s' 被丢弃, l 被分为3段
		l 完全包含在 s' 中, l 被丢弃, s' 被分为3段
		s' 和 l 相交于非端点, s' 和 l 根据相交点分别分为两段
		l 的端点和 s' 相交, s' 被分为两段
		s' 的端点和 l 相交, l 被分为两段

4.1.3 面片切割

在这个阶段,所有相交结点内的面片已经完成相交测试,相应的相交线段集也已经生成并加入相应面片的列表中,这些相交线段的集合定义了三角网格表面分段线性的相交线及包围每段相交线的外轮廓.因此,面片的切割被转换为带约束的三角化问题,通过对相交组中每个面片 a 关联的相交线段集 L_a 执行带约束的三角剖分并用剖分生成的新三角形集合替换 a ,最终输入模型所有面片都不存在与其他模型部分相交的情况.由于在 L_a 中所有线段都位于二维空间,因此面片切割被进一步简化为二维带约束的三角剖分问题.在实际实现中常使用二维 Delaunay 三角剖分^[34],文献[35]中所提算法被用于执行这样的三角化过程,其中, L_a 中的线段定义了所有的约束边.三角化的结果在替换原始三角形 a 前被转换回世界坐标系.

4.2 全局化面片分类

二叉树所有相交结点内面片利用延迟切割算法切割后彼此不再相交,这些切割后的面片构成一棵 CSG 子树,类似地,所有内部结点的面片构成另一棵 CSG 子树,因此面片分类仅需确定这两棵子树中面片相对于其他模型的关系以便决定是否将面片保留在最终结果中.利用性质 2,算法在构建由内部结点中面片构成的子树时,对面片按所属结点分组,分类时仅从每组中任选一个面片进行测试从而得到整组中面片的关系,这种组分类策略大大优化了分类的性能.

由于三角面片不再彼此相交,面片和一个网格的关系测试被转换为确定三角形的质心和多面体之间关系的基本几何问题.在点和多面体关系的测试中,算法要求被测模型是基于边界表示的拓扑多面体^[36].然而,传统分类算法不能保证被测模型总是拓扑多面体,这是由于传统的方法基于局部评估策略,分类在 CSG 树任一中间结点的两个子结点表示的模型之间直接进行.当子结点是叶结点时,该结点表示的模型是切割后的模型;当子结点是中间结点时,该结点表示的模型是对其子树分类的结果.错误的分类结果将使得结点表示的模型产生破洞

或裂缝,并且该结果将往上传播,最终导致算法无法生成正确的结果.由于切割后面片数多于原来的模型,这种方法也导致算法性能有所下降.

为了避免这些问题的出现,本文利用分治法使得分类总是在切割后的面片和原始输入模型之间进行.因此,在构建 CSG 子树时,每个叶结点包含对输入 CSG 树对应叶结点中原始输入模型的引用.给定一个被分类的面片 t ,基于分治法的分类算法首先判断 t 所在结点的兄弟结点 n 是否是叶结点,如果条件成立,则测试 t 的质心和 n 对应原始输入模型之间的关系.否则,算法分别测试 t 和 n 的左右子树表示模型之间的关系,这个递归过程一直向下进行直到叶结点,此时, t 分别和叶结点引用的原始输入模型进行分类,分类的结果再往上传递到父结点,在该结点处根据定义的布尔运算符从预定义的规则表中(见表 3~表 5)得到分类结果,这个结果一直往上传递最终得到 t 和 n 的分类结果.具体的过程描述在算法 1 中给出,其中函数 Combine 以 n 上定义的布尔运算符以及左右子树的分类结果为输入,根据表 3~表 5 中定义的规则得到 t 和 n 的分类结果.

Table 3 Classification relation between model C and $A \cup B$

表 3 模型 C 中面片相对 $A \cup B$ 的分类关系

	$C \text{ in } B$	$C \text{ out } B$	$C \text{ with } B$	$C \text{ anti } B$
$C \text{ in } A$	In	In	In	In
$C \text{ out } A$	In	Out	With	Anti
$C \text{ with } A$	In	With	With	In
$C \text{ anti } A$	In	Anti	In	Out

In 是指面片在被测模型内;Out 是指面片在被测模型外;With 是指面片和被测模型共面且法向量一致;Anti 是指面片和被测模型共面但法向量相反

Table 4 Classification relation between model C and $A \cap B$

表 4 模型 C 中面片相对 $A \cap B$ 的分类关系

	$C \text{ in } B$	$C \text{ out } B$	$C \text{ with } B$	$C \text{ anti } B$
$C \text{ in } A$	In	Out	With	Anti
$C \text{ out } A$	Out	Out	Out	Out
$C \text{ with } A$	With	Out	With	Out
$C \text{ anti } A$	Anti	Out	Out	Anti

Table 5 Classification relation between model C and $A - B$

表 5 模型 C 中面片相对 $A - B$ 的分类

	$C \text{ in } B$	$C \text{ out } B$	$C \text{ with } B$	$C \text{ anti } B$
$C \text{ in } A$	Out	In	Anti	With
$C \text{ out } A$	Out	Out	Out	Out
$C \text{ with } A$	Out	With	Out	With
$C \text{ anti } A$	Out	Anti	Anti	Out

点和多面体的关系测试是基本的几何问题,许多研究者提出了不同的方法来解决这个问题^[37-40].这些算法按照是否利用空间结构来加速测试过程被分为两类:非空间结构算法和空间结构算法.非空间结构算法需要测试模型的所有面片,而空间结构算法仅需模型的部分面片信息.因此基于空间结构的点和多面体关系测试性能大大优于基于非空间结构的算法.利用算法开始阶段构建的八叉树,算法采用基于 Joran 曲线理论^[41]的点和多面体分类方法.利用文献[42]提出的参数化方法,分类算法的性能被进一步优化.

算法 2 描述了八叉树内部结点面片构成的 CSG 子树的评估过程,对于由相交结点内面片切割后生成的 CSG 子树,当假设结点中每个面片属于不同的组时,该子树的评估与此类似.算法以 CSG 子树的根结点为输入,在每个递归阶段收集来自左右子结点的面片组,由于每组中的面片具有相同的分类关系,每组中仅第 1 个面片测试和兄弟结点表示模型之间的关系.测试结果与表 3 和表 5 定义的规则一起决定是否将该组中所有面片保留在当前结点中.这个过程一直持续到根结点,从而得到最终的分类后结果.

算法 1. Classify.

输入:三角面片 t ; CSG 树结点 n .

输出:面片 t 的分类结果.

1: **if** n 是叶结点 **then**

```

2:   return Point-in-polyhedron( $t$ .barycenter, $n$ .mesh); //点和多面体关系测试
3:   else
4:   return Combine(Classify( $t$ , $n$ .left),Classify( $t$ , $n$ .right), $n$ .operator);
5: endif

```

算法 2. EvaluateCSGTree.

输入:CSG 树结点 n ;

输出:以 n 为根结点的 CSG 树布尔运算评估结果.

```

1: if  $n$ .left 是叶结点 then
2:   保存  $n$  的左子结点中所有面片组进面片组集合  $G_l$ ;
3:   else
4:    $G_l$ =EvaluateCSGTree( $n$ .left);
5:   end if
6: if  $n$ .right 是叶结点 then
7:   保存  $n$  的右子结点中所有面片组进面片组集合  $G_r$ ;
8:   else
9:    $G_r$ =EvaluateCSGTree( $n$ .right);
10:  end if
11: foreach  $G_l$  中面片组  $g$  do
12:   选择  $g$  中第 1 个面片  $f$ ;
13:   if IsAcceptable(Classify( $f$ , $n$ .right), $n$ .operator)) then //对  $f$  和  $n$ .right 表示的模型分类并判断可否接受
14:     如果可接受,则保存  $g$  至 node  $n$ ;
15:   end if
16: end for
17: foreach  $G_r$  中面片组  $g$  do
18:   选择  $g$  中第 1 个面片  $f$ ;
19:   if IsAcceptable(Classify( $f$ , $n$ .left), $n$ .operator)) then //对  $f$  和  $n$ .right 表示的模型分类并判断可否接受
20:     如果可接受,则保存  $g$  至 node  $n$ ;
21:   end if
22: end for
23: return 结点  $n$  中所有面片组.

```

5 实验

5.1 实验配置

整个系统使用 C++作为编程语言,免费的 Fade2D^[43]用于执行带约束的三角剖分.同时,英特尔 TBB 多线程库也被用于并行化点到多面体关系的判断算法.在构建自适应的八叉树时,每个相交类型的叶子结点中最大三角形的数量被设置为 17.这个数据是经过对多个模型进行划分,综合比较算法的运行时间后而得出的最优值.

算法的测试与比较在一台配置为 CPU i7 2.2 GHz,内存 16GB 的消费级计算机上进行.为了进行性能对比,我们实现了最新的基于平面 BSP 表示 Campen^[7]的布尔算法.同时,商业化软件 Maya 2015 也被选择作为比较对象.

5.2 单次布尔测试

单次布尔运算被广泛运用在实时交互系统中,这样的应用要求算法鲁棒及响应速度快.因此本文从两个维度来评估算法的鲁棒性和性能,一是算法的覆盖度,本文通过对图 6 每组模型用不同的算法分别执行交、并、差,然后计算每种算法的平均时间,结果显示在表 6 中;二是算法的时间复杂度,通过对同一模型不断的细分来评价算法的运行时间和三角面片之间的关系,结果显示在表 7 中.

从表 6 可以看出,不同的算法在相同测试集下性能差异明显.商业化软件 Maya 2015 和 Campen 算法对于简单的模型表现良好,但当模型的面片数超过 200 000 时,性能急剧下降.由于存在鲁棒性问题,当模型的面片数超过 4 000 000 时,Maya 2015 将无法产生正确的结果,而 Campen 算法则无法分配到足够的内存而导致程序崩溃.相比较而言,本文提出的算法对图 6 的各种模型保持了良好的鲁棒性,同时测试结果显示出更好的性能.比如:当

模型的面片数超过 150 000 时,本文算法分别比 Maya 2015 快 4 倍,比 Campen 算法快 10 倍.这是由于 Campen 算法需要额外的预处理和后处理过程,随着模型数的增加,额外的处理过程成为系统的性能瓶颈.而本文算法直接处理三角网格,避免了这些额外的开销.另外,对输入模型而言,相交线附近的面片仅占整个模型的面片数的很小比例.本文构建的自适应八叉树将这些面片划分到相交结点中,通过在每个相交结点内执行相交测试并利用延迟切割技术大大提高了算法的性能.在表 6 中,我们还可以发现恐龙-怪兽的面片数(2 317k)和龙-兔子面片数(2 314k)接近,但是和龙-兔子模型相比,其布尔运算时间却大了 15%,这是由于算法的性能不仅取决于相交测试的性能,也依赖分类的性能.在龙-兔子模型中,龙和兔子相交角度接近 0° ,这使得构建的八叉树中含有大量的内部结点,这也影响了分类的性能.一般说来,当两个模型相互垂直时,八叉树中的内部结点数最少.

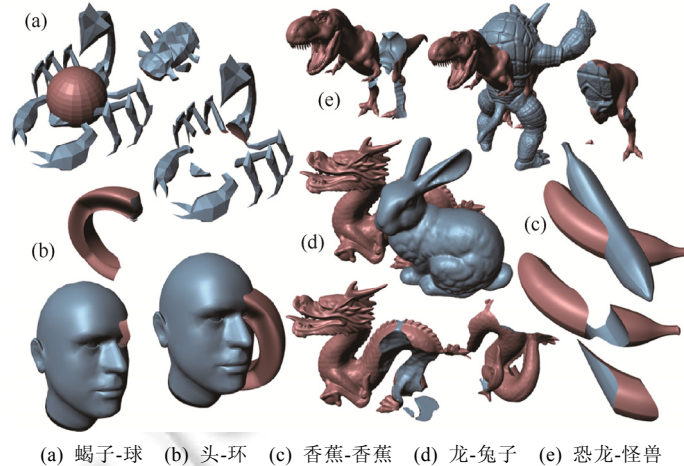


Fig.6 Models and its Boolean results

图 6 模型及布尔运算结果

Table 6 Average times in milliseconds for intersection, union and difference operations of different meshes in Fig.6 (Unit: ms)

表 6 对图 6 中每组模型交、并、差平均运算时间的统计 (单位:毫秒)

模型	面片数(k)	Maya 2015	Campen	本文方法
蝎子-球	2.4	62	113	31
	9.6	219	272	79
	38.2	920	800	254
头-环	10	281	349	97
	40	1 217	1 192	258
	160	5 679	5 481	934
香蕉-香蕉	48	983	1 725	368
	192	4 946	11 576	1 303
	768	24 960	70 792	6 757
龙-兔子	170	4 968	12 774	1 544
	680	23 594	42 707	5 060
	2 314	128 203	内存不足	30 426
恐龙-怪兽	2 317	95 610	内存不足	26 550

“内存不足”是指由于堆栈溢出导致程序崩溃

为了评估算法的复杂度,我们通过不断细分图 6(c)的两个香蕉模型,每次细分使得模型的面片数量增加到原来的 4 倍,从而得到相对关系不变、但面片数不同的一系列模型.然后统计不同算法对这些模型分别执行交、并、差的平均时间.从表 7 可以看出,即使对于简单的模型,当模型的面片数不断增加时,本文算法仍然明显优于其他算法.这是由于,当输入模型的相对关系保持不变时,本文构建的八叉树仅相交结点数随着面片数的增加而增加,外部结点和内部结点数基本保持不变,从而算法的性能取决于相交测试的时间.而在本文算法中,相交测试的时间复杂度是线性的,因此表 7 显示本文算法的时间复杂度近似 $O(n)$, n 为模型的三角面片数.

Table 7 Average time for Boolean operations with increasing facet count (ms)
表 7 对图 6(c)模型在不同面片数量下交、并、差平均运算时间的统计(毫秒)

面片数 (k)	Maya 2015	Campen	本文方法
8	250	319	139
32	797	971	313
128	3 406	4 516	771
512	16 438	34 113	2 554
2 048	87 453	32 790	12 371

5.3 连续布尔测试

连续布尔运算有着广泛的应用,比如数字雕刻、计算辅助设计等.这些应用要求算法零失败、高性能,而这正是本文算法针对的问题之一.我们通过对一系列小球和环执行连续的交操作,同时不断增加小球的数量来测试算法在连续布尔运算下的性能和鲁棒性.测试模型和部分的求交结果显示如图 7 所示,不同算法的测试结果比较如图 8 所示.

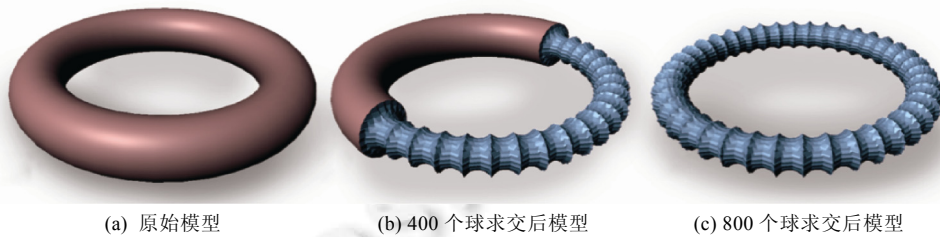


Fig.7 Sculpting a ring

图 7 交操作雕刻环

从图 8 可以清楚地看到,随着求交次数的不断增加,Maya 的处理时间呈指数上升,而 Campen 算法在求交次数超过 400 后性能快速下降.这是因为这些方法把连续的布尔运算操作分解成一系列的单次布尔运算,前一次的运算结果作为后一次的输入.由于布尔运算需要对相交区进行切割,每次运算都会产生新的面片,这些新的面片也参与分类,从而导致连续布尔运算的精度和性能不断下降.特别地,由于 Campen 算法采用局部化相交处理技术,性能优于 Maya.但是 Campen 算法在预处理阶段需要沿着关键结点包围盒边界切割面片;在后处理阶段需要执行三角化,随着运算次数的不断增加,一方面,BSP 树合并操作花费时间越来越长,另一方面,数值误差不断累积,这使得算法性能迅速下降.相比较而言,本文算法利用 CSG 树全局评估策略,通过自适应地延迟切割技术使得相交测试限定在八叉树相交结点内并保证整个相交过程中不产生新的面片,直到所有模型的相交测试完成后才进行一次切割,从而避免了数值误差的累积并提高了算法性能.面片的分类也利用全局化分类技术基于 CSG 树一次完成,所以本文算法对连续布尔运算在维持健壮性的同时保持了高性能,其处理时间远远小于其他算法.

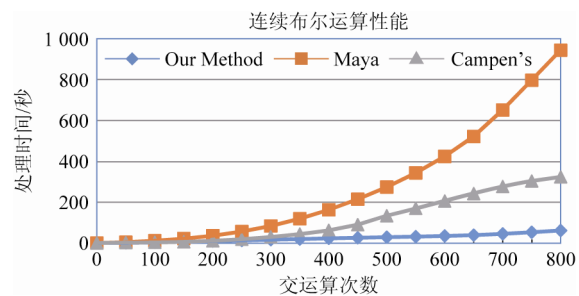


Fig.8 Time in seconds for continuous Boolean operation

图 8 不同算法连续交运算性能比较图

6 结 论

本文呈现了一种基于三角网格的高效、精确、鲁棒的布尔运算评估算法.算法利用 CSG 树全局评估策略实现了对单次和连续布尔运算的统一处理.通过限定相交测试在自适应八叉树的每个相交结点内,算法保证了在线性时间内完成面片的相交测试;通过扩展延迟切割技术至整个 CSG 树,算法实现了单次和连续布尔运算下相交测试和切割的鲁棒性和精度;通过使用分治法使得分类始终在切割后面片和原始输入模型之间进行,算法

保证了分类的鲁棒性和精度;通过组策略优化八叉树内部结点中面片的分类性能,并利用八叉树加速点和多面体关系测试的性能,算法实现了在线性时间内完成面片的分类.因此,该算法对三角网格构成的复杂场景执行布尔运算时,可在保证鲁棒性的同时实现高性能.

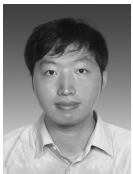
References:

- [1] Ertl T. *Computer Graphics—Principles and Practice*. Springer-Verlag, 1996. [doi: 10.1007/978-3-7091-2684-4_31]
- [2] Requicha A. *Mathematical models of rigid solid objects*. Technical Memorandum 28, University of Rochester, 1977.
- [3] Benouamer M, Michelucci D, Peroche B. Error-Free boundary evaluation using lazy rational arithmetic: A detailed implementation. In: *Proc. of the 2nd ACM Symp. on Solid Modeling and Applications*. ACM, 1993. [doi: 10.1145/164360.164403]
- [4] CGAL. *Computational Geometry*, 2014. <http://www.cgal.org>
- [5] Autodesk. *Maya*, 2014. <http://www.autodesk.com>
- [6] Bernstein G, Fussell D. Fast, exact, linear Booleans. *Computer Graphics Forum*, 2009,28(5):1269–1278. [doi: 10.1111/j.1467-8659.2009.01504.x]
- [7] Campen M, Kobbelt L. Exact and robust (self-) intersections for polygonal meshes. *Computer Graphics Forum*, 2010,29(2):397–406. [doi: 10.1111/j.1467-8659.2009.01609.x]
- [8] Hoffmann CM. *Geometric and Solid Modeling*. Palo Alto: Morgan Kaufmann Publishers, 1989.
- [9] Feito FR, Ogayar CJ, Segura RJ, Rivero M. Fast and accurate evaluation of regularized Boolean operations on triangulated solids. *Computer-Aided Design*, 2013,40(3):705–716. [doi: 10.1016/j.cad.2012.11.004]
- [10] Frisken SF, Perry RN, Rockwood AP, Jones TR. Adaptively sampled distance fields: A general representation of shape for computer graphics. In: *Proc. of the 27th Annual Conf. on Computer Graphics and Interactive Techniques*. 2000. [doi: 10.1145/344779.344899]
- [11] Museth K, Breen DE, Whitaker RT, Barr AH. Level set surface editing operators. *ACM Trans. on Graphics (TOG)*, 2002,21(3):330–338. [doi: 10.1145/566654.566585]
- [12] Kobbelt LP, Botsch M, Schwanecke U, Seidel HP. Feature sensitive surface extraction from volume data. In: *Proc. of the 28th Annual Conf. on Computer Graphics and Interactive Techniques*. 2001. [doi: 10.1145/383259.383265]
- [13] Ju T, Losasso F, Schaefer S, Warren J. Dual contouring of hermite data. *ACM Trans. on Graphics (TOG)*, 2002,21(3):339–346. [doi: 10.1145/566654.566586]
- [14] Varadhan G, Krishnan S, Kim YJ, Manocha D. Feature-Sensitive subdivision and isosurface reconstruction. In: *Proc. of the Visualization. IEEE*, 2003. 99–106. [doi: 10.1109/VISUAL.2003.1250360]
- [15] Varadhan G, Krishnan S, Sriram TVN, Manocha D. Topology preserving surface extraction using adaptive subdivision. In: *Proc. of the 2004 Eurographics/ACM SIGGRAPH Symp. on Geometry Processing*. 2004. 235–244. [doi: 10.1145/1057432.1057464]
- [16] Zhang N, Hong W, Kaufman A. Dual contouring with topology-preserving simplification using enhanced cell representation. In: *Proc. of the Visualization. IEEE*, 2004. 505–512. [doi: 10.1109/VISUAL.2004.27]
- [17] Schaefer S, Ju T, Warren J. Manifold dual contouring. *IEEE Trans. on Visualization and Computer Graphics*, 2007,13(3):610–619. [doi: 10.1109/TVCG.2007.1012]
- [18] Smith JM, Dodgson NA. A topologically robust algorithm for Boolean operations on polyhedral shapes using approximate arithmetic. *Computer-Aided Design*, 2007,39(2):149–163. [doi: 10.1016/j.cad.2006.11.003]
- [19] Edelsbrunner H, Mücke EP. Simulation of simplicity: A technique to cope with degenerate cases in geometric algorithms. *ACM Trans. on Graphics (TOG)*, 1990,9(1):66–104. [doi: 10.1145/77635.77639]
- [20] Wang CC. Approximate Boolean operations on large polyhedral solids with partial mesh reconstruction. *IEEE Trans. on Visualization and Computer Graphics*, 2011,17(6):836–849. [doi: 10.1109/TVCG.2010.106]
- [21] Shade J, Gortler S, He LW, Szeliski R. Layered depth images. In: *Proc. of the 25th Annual Conf. on Computer Graphics and Interactive Techniques*. 1998. [doi: 10.1145/280814.280882]
- [22] Thibault WC, Naylor BF. Set operations on polyhedra using binary space partitioning trees. *ACM SIGGRAPH Computer Graphics*, 1987,21(4):153–162. [doi: 10.1145/37402.37421]
- [23] Naylor B, Amanatides J, Thibault W. Merging BSP trees yields polyhedral set operations. *ACM SIGGRAPH Computer Graphics*, 1990,24(4). [doi: 10.1145/97880.97892]
- [24] Thibault WC. *Application of binary space partitioning trees to geometric modeling and ray-tracing* [Ph.D. Thesis]. Georgia Institute of Technology, 1987.
- [25] Shewchuk JR. Adaptive precision floating-point arithmetic and fast robust geometric predicates. *Discrete & Computational Geometry*, 1997,18(3):305–363. [doi: 10.1007/PL00009321]

- [26] Murali TM, Funkhouser TA. Consistent solid and boundary representations from arbitrary polygonal data. In: Proc. of the 1997 Symp. on Interactive 3D Graphics. ACM, 1997. [doi: 10.1145/253284.253326]
- [27] Hubbard PM. Constructive solid geometry for triangulated polyhedral [Ph.D. Thesis]. Department of Computer Science, Brown University, 1990.
- [28] Yamaguchi F, Tokieda T. A unified algorithm for Boolean shape operations. Computer Graphics and Applications, 1987,4(6):24–37. [doi: 10.1109/MCG.1984.275959]
- [29] Pilz M, Kamel HA. Creation and boundary evaluation of CSG-models. Engineering with Computers, 1987,5(2):105–118. [doi: 10.1007/BF01199073]
- [30] Mäntylä M, Tamminen M. Localized set operations for solid modeling. ACM SIGGRAPH Computer Graphics, 1983,17(3):279–288. [doi: 10.1145/964967.801159]
- [31] Kuratowski K, Mostowski A. Set Theory. Elsevier: Academic Press, 1968.
- [32] Ericson C. Real-Time Collision Detection. Boca Raton: CRC Press, 2004.
- [33] Möller T. A fast triangle-triangle intersection test. Journal of Graphics Tools, 1997,2(2):25–31. [doi: 10.1080/10867651.1997.10487472]
- [34] Lee DT, Schachter BJ. Two algorithms for constructing a Delaunay triangulation. Int'l Journal of Computer & Information Sciences, 1980,9(3):219–242. [doi: 10.1007/BF00977785]
- [35] Chew LP. Constrained delaunay triangulations. Algorithmica, 1989,4(1/4):97–108. [doi: 10.1007/BF01553876]
- [36] Horn WP, Taylor DL. A theorem to determine the spatial containment of a point in a planar polyhedron. Computer Vision, Graphics, and Image Processing, 1989,45(1):106–116. [doi: 10.1016/0734-189X(89)90073-X]
- [37] Feito FR, Torres JC. Inclusion test for general polyhedral. Computers & Graphics, 1997,21(1):23–30. [doi: 10.1016/S0097-8493(96)00067-2]
- [38] Ogayar CJ, Segura RJ, Feito FR. Point in solid strategies. Computers & Graphics, 2005,29(4):616–624. [doi: 10.1016/j.cag.2005.05.012]
- [39] Liu J, Chen YQ, Maisog JM, Luta G. A new point containment test algorithm based on preprocessing and determining triangles. Computer-Aided Design, 2010,42(12):1143–1150. [doi: 10.1016/j.cad.2010.08.002]
- [40] Wang W, Li J, Sun H, Wu E. Layer-Based representation of polyhedrons for point containment tests. IEEE Trans. on Visualization and Computer Graphics, 2008,14(1):73–83. [doi: 10.1109/TVCG.2007.70407]
- [41] Veblen O. Theory on plane curves in non-metrical analysis situs. Trans. of the American Mathematical Society, 1905,6(1):83–98. [doi: 10.1090/S0002-9947-1905-1500697-4]
- [42] Revelles J, Urena C, Lastra M. An efficient parametric algorithm for octree traversal. In: Proc. of the WSCG. 2000. 212–219.
- [43] Kornberger B. Fade2D. <http://www.geom.at>



姜旭东(1975—),男,江苏镇江人,硕士,主要研究领域为计算机图形学。



盛斌(1981—),男,博士,副教授,博士生导师,CCF 会员,主要研究领域为虚拟现实,计算机图形学。



马利庄(1963—),男,博士,教授,博士生导师,CCF 会员,主要研究领域为计算机图形,数字多媒体,计算机辅助设计。



申瑞民(1967—),男,博士,教授,博士生导师,CCF 会员,主要研究领域为计算机图形学。



吴恩华(1947—),男,博士,教授,博士生导师,CCF 会士,主要研究领域为计算机图形学,虚拟现实。