B-Rep 模型高效布尔运算算法

梁思立[1)], 黄浩勉[2)], 梁好[2)], 唐为然[3)], 鲍桉畅[2)], 沈恩亚[2,4)*], 王建民[2)], 杨义军[3)]

[1)] (南开大学软件学院　天津　300457)
[2)] (清华大学软件学院　北京　100084)
[3)] (西安交通大学计算机科学与技术学院　西安　710049)
[4)] (先进计算与关键软件(信创)海河实验室　天津　300459)
(shenenya@tsinghua.edu.cn)

摘　要: 布尔运算是三维建模中一项基础操作, 其技术水平对建模的鲁棒性、效率有着重要的影响. 针对目前公开的 B-Rep 模型的布尔运算研究中难以兼顾鲁棒性和效率的问题, 提出一种高效的 B-Rep 模型的布尔运算算法. 首先采用层次化包围盒干涉检查、面分组和交线局部信息判断等方法, 极大地减少几何求交的次数; 然后引入构建交线图、多次成环等技术提高了算法的稳定性. 基于实际三维 CAD 模型, 与几何建模引擎 Parasolid, ACIS 和 OCCT 进行对比实验的结果表明, 与 OCCT 的布尔运算相比, 所提算法在效率上具有一定优势; 与 Parasolid 和 ACIS 的布尔运算相比, 该算法在一些场景下的效率相当, 为 B-Rep 模型的布尔运算提供了一种有效的解决方案.

关键词: B-Rep; 布尔运算; 三维几何建模; CAD
中图法分类号: TP391.41　　**DOI:** 10.3724/SP.J.1089.2024-00367

Efficient Boolean Operations for B-Rep Models

Liang Sili[1)], Huang Haomian[2)], Liang Hao[2)], Tang Weiran[3)], Bao Anchang[2)], Shen Enya[2,4)*], Wang Jianmin[2)], and Yang Yijun[3)]

[1)] (*College of Software, Nankai University, Tianjin*　300457)
[2)] (*School of Software, Tsinghua University, Beijing*　100084)
[3)] (*School of Computer Science and Technology, Xi'an Jiaotong University, Xi'an*　710049)
[4)] (*Haihe Lab of ITAI, Tianjin*　300459)

Abstract: Boolean operations are fundamental in 3D modeling, as their performance significantly impacts the robustness and efficiency of the modeling process. Current research on Boolean operations for B-Rep models often struggle to achieve a balance between robustness and efficiency. This paper introduces an efficient algorithm for B-Rep Boolean operations. The algorithm leverages a Bounding Volume Hierarchy (BVH), face clustering, and local information near intersection edges to reduce the number of geometric intersection calculations. Additionally, techniques such as intersection graph construction and progressive imprinting are employed to enhance the algorithm's stability. Experimental evaluations using CAD models from real-world scenarios demonstrate that the proposed algorithm offers significant efficiency improvements over OCCT's Boolean operations and achieves performance comparable to that of Parasolid and ACIS in specific scenarios. These results underscore the proposed algorithm as a highly effective solution for Boolean operations on B-Rep models.

收稿日期: 2024-07-04; 修回日期: 2024-11-28. **基金项目**: 国家重点研发计划(2021YFB1715900); 先进计算与关键软件海河实验室重点项目(XCHK20210102); 深圳市创新创业计划(KJZD20230923114114028). **梁思立**(1999—), 男, 博士研究生, CCF 学生会员, 主要研究方向为计算机图形学; **黄浩勉**(1998—), 男, 硕士研究生, 主要研究方向为计算机图形学; **梁好**(2000—), 男, 硕士研究生, 主要研究方向为计算机图形学; **唐为然**(1998—), 男, 硕士研究生, 主要研究方向为计算机图形学; **鲍桉畅**(2002—), 男, 博士研究生, 主要研究方向为计算机图形学; **沈恩亚**(1985—), 男, 博士, 助理研究员, 论文通信作者, 主要研究方向为几何建模; **王建民**(1968—), 男, 博士, 教授, 博士生导师, 主要研究方向为大数据与知识工程; **杨义军**(1979—), 男, 博士, 教授, 博士生导师, CCF 会员, 主要研究方向为 CAD&CG.

Key words: B-Rep; Boolean operation; 3D Geometric Modeling; CAD

在边界表示(boundary representation, B-Rep)的三维几何建模领域中，布尔运算是一种通过布尔算子(交、并和差)将 2 个形体组合成一个形体的操作[1]. 在 CAD 软件中，布尔运算技术水平对建模的鲁棒性和效率有着重要的影响[2].

B-Rep 模型的布尔运算研究中面临着鲁棒性和效率的挑战，当前公开的布尔运算研究很难在这 2 个方面同时兼顾. 由于 B-Rep 模型通常被视为共享边界的裁剪面片的集合，因此各个面片的边界只是近似匹配，难以确保不同参数域中的 2 条裁剪曲线在空间上相同. 针对该问题, Urick 等[3]提出水密布尔运算框架，建立裁剪面片模型及其对应水密模型的映射关系，获得由具有显著连续性的未裁剪曲面片组成的水密模型，但这种操作带来了额外的时间开销. 布尔运算作为几何建模引擎基础的建模操作，其耗时集中在涉及求交运算的步骤. 对于一些不需要精确结果的应用场景，如动画或者游戏建模，可以通过针对网格模型的重构与缝合[4]等方法规避求交操作，得到布尔运算的近似结果[5]，使得算法的效率比传统方法有了很大的提高. 但是，对于需要精确布尔运算结果的场景，如工业设计，仍然需要基于参数化曲线曲面求交的布尔运算[6].

当前，兼顾鲁棒性和效率的布尔运算算法主要存在于以 Parasolid 和 ACIS 为代表的商用几何建模内核中.

Parasolid 是由 Siemens PLM Software 开发的商用几何建模内核，广泛应用于 CAD/CAM/CAE 等多个领域. 作为一个成熟的几何建模引擎，Parasolid 提供了强大的布尔运算功能，能够处理复杂的几何体之间的交、并和差操作. Parasolid 的布尔运算算法以其高效性和鲁棒性著称，经过多年的优化与验证，能够在处理大型和复杂的模型时保持稳定的性能表现.

ACIS 是 Spatial 公司开发的业界领先的三维几何建模内核，被用于 14 个领域的数百款软件，包括 CAD/CAM/CAE，建筑、工程与施工(architecture, engineering and construction, AEC)，动画和造船业等，提供了包含布尔运算在内的强大的几何建模功能. ACIS 的布尔运算模块经过三十多年真实工业场景的测试和验证，具有较高的稳定性和可靠性，可以处理各种情况下的几何相交和重合，确保生成准确的布尔运算结果；同时，其优化的算法和数据结构设计，使得处理大型模型时能够保持较高的运算速度和稳定性.

OCCT(open CASCADE technology)是一个开源的几何建模引擎，拥有完备的几何建模功能，包括对多种文件格式的支持以及高级建模功能；但是，与其他商业内核相比，OCCT 存在计算效率较低的问题，尤其是在面对复杂模型时，其布尔运算效率不尽如人意.

基于此，本文提出一种高效的 B-Rep 模型的布尔运算算法. 首先使用层次化包围盒(bounding volume hierarchy, BVH)干涉检查、面分组和交线局部信息进行相对位置关系判断等方法，极大地减少了几何求交的次数；然后通过构建交线图、多次成环等方法提高算法的稳定性.

1 相关工作

1.1 研究工作中的 B-Rep 布尔运算

作为几何建模引擎中基础的建模操作，布尔运算的技术水平直接影响建模的鲁棒性和效率；特别是在涉及复杂形体的工业设计中，布尔运算的精度和效率显得尤为重要. 作为布尔运算中最耗时的部分，求交运算成为优化的关键. 根据不同应用场景的需求，布尔运算的优化方向和策略有所不同.

对于动画和游戏建模等不要求高精度的场景中，优化的重点在于通过近似方法提高效率. 网格重构与缝合[4]可以规避烦琐的求交运算，快速生成近似结果[5].

在工业设计等需要精确结果的领域，求交运算仍然是必不可少的[6]. 鉴于现有的复杂曲线曲面的求交算法已经比较成熟，再进一步优化这些算法本身非常困难，因此研究的重点转向了如何通过优化空间划分结构来减少求交运算的频率. 如通过构建八叉树或 BVH 可以有效地缩小运算范围，仅在有可能发生相交的局部区域内执行求交计算，该方法不仅可减少计算量，还提升了算法的并行处理能力，大幅提高了整体效率.

近年来，B-Rep 布尔运算的相关研究主要集中在网格模型的布尔运算优化上，网格模型因其灵活性和广泛应用在渲染和工业设计中占据了重要地位. Feito 等[7]利用八叉树结构进行面片的快速相交检测，通过将面片划分到交线区域并进行聚类，

大大减少了点-实体位置关系[8]的判断次数；同时，利用此前构建的八叉树，减少射线与三角面片的求交判断，并通过光线与三角面片的快速求交算法[9]，使得求交能够并行执行．这种方法虽然有效，但也暴露了一些局限性，如其主要针对三角网格，并不适用于其他 B-Rep 模型类型．此外，如果网格面片的大小和形状差异较大，可能会影响八叉树的分割效果，进而降低干涉检查的效率．

为了解决这些问题，Nehring-Wirxel 等[10]提出结合二叉树空间划分(binary space partiton, BSP)树八叉树的方法，通过将 BSP 树嵌入到八叉树的叶子节点中[11]，并将 2 个网格实体的布尔运算转换为 2 个 BSP 树的布尔运算[12]，显著地提高了布尔运算的效率；然而，因为该方法要求实体及其面片必须是凸的，限制了其在更复杂的模型上的应用．

还有一些研究通过转换计算形式来优化布尔运算．Trettner 等[13]提出卷绕数向量(winding number vector, WNV)方法，将布尔运算转换为空间中的拓扑计算[14]，利用八叉树结构进一步简化计算．这种创新的思路展示了将复杂几何问题转换为简单拓扑问题的潜力，开辟了新的优化途径．

然而，当前大多数优化策略主要针对离散网格模型，而对于参数化曲线曲面模型的研究较少．与网格模型相比，参数化曲线曲面模型具有更强的表达能力，但也更加复杂．传统的空间划分策略难以直接应用于这类模型，尤其是当面片的大小和形状差异较大时，如何有效地组织和划分这些面片成为一大挑战．考虑工业设计中广泛使用的参数化曲线曲面模型，研究如何提升这类模型的布尔运算效率显得尤为重要．

1.2　几何内核引擎中的布尔运算

除了上文提到的研究工作外，市场上几乎所有的内核都有自己的一套布尔运算流程，但由于该领域的特殊性，大部分成果都没有发表，而是作为算法集成到了内核中．

Parasolid 是一个广泛应用的商业级几何建模内核，由 Siemens PLM Software 开发和维护，已在 CAD/CAM/CAE 中得到了广泛应用，如 SolidWorks 和 NX 等知名软件都是基于 Parasolid 内核．Parasolid 以其稳健的几何建模能力和高度优化的算法著称，在布尔运算的处理上表现卓越，能够快速且精确地处理复杂的几何变换和布尔运算，满足高精度的建模需求．

ACIS 是一个应用非常广泛的商业级的几何建模内核，包括久负盛名的 CAD 软件 AutoCAD 和有限元计算软件 Abaqus，均是基于此内核开发的．ACIS 采用面向对象的数据结构，非常有利于开发者对其进行灵活的使用或者拓展，且支持复杂的几何造型需求；经过几十年的实践积累，其可以处理各种情况下的几何相交和重合，确保生成准确的布尔运算结果，同时，优化的算法和数据结构设计使得其处理大型模型时能够保持较高的运算速度和稳定性．

OCCT 是一个开源的几何建模内核，广泛应用于 CAD/CAE 等软件的开发，由于其开源且免费，因此极大地降低了在其基础上进行二次开发的成本．作为一个成熟的内核，OCCT 也拥有完备的几何建模功能，包括对多种文件格式的支持以及高级建模功能．但是，与其他商业内核相比，OCCT 存在计算效率较低的问题；尤其是在大场景、面对复杂模型时，其布尔运算效率不尽如人意．

2　本文算法

B-Rep 模型由线框区域、面片区域和实体区域组成．布尔运算涉及的 2 个模型称为执行操作的模型(tool)和被执行操作的模型(blank)．通常，布尔运算可以分为生成交线图、交线成环、线面分类和拓扑合并 4 个步骤[15]．本文算法的流程如图 1 所示．

Step1．生成交线图．在 2 个模型中，将模型的各个区域与另一个模型的各个区域进行拓扑求交，得到交线或者交点．

Step2．交线成环．根据交线和交点对 2 个模型进行面拆分和边打断．经过该步骤后，模型的线面元素只会完全在另一个模型的外部，或另一个模型的边界上，或另一个模型的内部．

Step3．线面分类．对于划分后的 2 个模型的线面，根据与另一个模型的相对位置关系进行分类[16]，模型的面和线框相对位置关系类别如表 1 所示．考虑模型存在线框区域和面片区域，同时避免重合情况下保留的不确定性，参考 ACIS，并对拓扑分类[6]进行扩展．

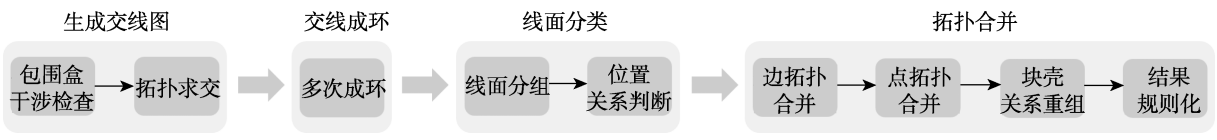


图 1　本文算法流程

表 1 模型的面和线框相对位置关系类别

类型	位置关系
wire_inside	线框在另一个模型的实体区域的内部
wire_outside	线框在另一个模型的实体区域的外部
wire_boundary	线框在另一个模型的边界
face_inside	面在另一个模型的内部
face_outside	面在另一个模型的外部
face_boundary_double_double	面在另一个模型的边界，自身是双边面，重合面是双边面
face_boundary_double_single	面在另一个模型的边界，自身是双边面，重合面是单边面
face_boundary_single_double	面在另一个模型的边界，自身是单边面，重合面是双边面
face_boundary_single_single_same_normal	面在另一个模型的边界，自身是单边面，重合面是单边面，法向同向
face_boundary_single_single_oppo_normal	面在另一个模型的边界，自身是单边面，重合面是单边面，法向反向

Step4. 拓扑合并. 根据布尔运算的类型对分类后的边、面进行舍弃, 如表 2 所示. 将 2 个模型剩余的拓扑结构合并到一个拓扑结构树中, 并且对各级拓扑结构的连接关系进行维护, 对需要合并的面与边进行合并, 最后形成布尔运算结果的边界表示形式.

本文算法可以处理线框区域、面片区域和实体区域, 图 2 所示为一个包含线框、面片和实体区域模型的布尔并/交/差示例.

表 2 基本布尔运算保留的相对位置关系类型

运算类型	tool 保留类型	blank 保留类型
布尔并	wire_outside, face_outside, face_boundary_single_double	wire_outside, wire_boundary, face_outside, face_boundary_double_double, face_boundary_single_double, face_boundary_single_single_same_normal
布尔交	wire_inside, wire_boundary, face_inside	wire_boundary, wire_inside, face_inside, face_boundary_double_double, face_boundary_double_single, face_boundary_single_double, face_boundary_single_single_same_normal
布尔差	wire_inside, face_inside	wire_outside, face_outside, face_boundary_single_double, face_boundary_single_single_oppo_normal

图 2 包含线框、面片和实体区域模型的布尔并/交/差示例

2.1 生成交线图

与几何求交只需要考虑边或者面的抽象数学描述不同, 2 个模型的求交需要考虑模型本身具有的复杂的边界信息. 直观上, 拓扑求交是在几何求交的基础上引入了边界信息的进阶操作, 生成交线图的整体流程如图 3 所示. 拓扑实体的求交结果将以交线图的形式传递给布尔运算的后续步骤.

2.1.1 交线图

交线图是一种由拓扑求交得到的仅由互相连接的交线边和它们的顶点组成的拓扑结构, 描述了 2 个模型的求交结果. 交线图的构造是简单和高效的, 可以在拓扑求交时逐步构造, 一条新的交线边根据其顶点是否与图中顶点在容差内重叠被合适地插入图中.

因为单条交线边受到各种误差的影响, 可能是不准确的, 所以将分离的交线边组合成一个图. 如果孤立地保存每条交线边, 则不能有效地利用其拓扑和几何信息提高准确性. 虽然交线图的拓扑结构非常简单, 但其携带了很多关于结果模型的几何和拓扑信息, 通过使用交线图, 可以很容易地检测出模型中可能断开和错位的区域.

2.1.2 基于 BVH 树的干涉检查

由于几何求交相关的步骤比较耗时, 因此需要在该步骤之前对模型进行干涉检查, 排除模型

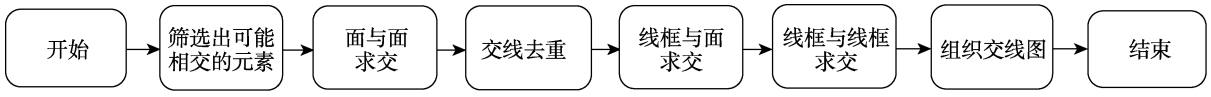


图 3 生成交线图的流程图

之间不会相交的部分，降低布尔运算的整体耗时.

一种比较常见的干涉检查方法是包围盒碰撞检查，通过检查 2 个元素的包围盒是否有重叠区域，筛选出可能出现相交的元素. 虽然该方法能够取得一定的优化效果，但对于拥有大量拓扑实体的模型或者场景，包围盒的检测也会有一定的时间消耗. 例如，使用布尔运算建模一些复杂设备时，场景中可能会同时有成百上千个面，如果进行两两之间的包围盒干涉检查会特别耗时. 此时，可以使用类似 BVH 树的层次化数据结构，重新组织位于拓扑树中同一层的元素，将一个元素与其附近的元素聚合起来形成拓扑树中的一个新节点，增加拓扑树的层级，从而在干涉检查中起到快速筛选的作用.

BVH 是一种计算机图形学中常用的用于快速碰撞检测的树状数据结构. 仿照 BVH 的原理为模型构造 BVH 树时，除了为面和边这种较底层的拓扑元素计算包围盒之外，也可以为壳、块等上层拓扑元素构建包围盒. 为了进一步增加包围盒的层级，起到对模型的各部分进行空间划分以加速判断的作用，在壳的拓扑层级下引入子壳的概念. 壳是由面和线框组成的一个连通分量，而子壳则由壳中部分连通的面和线框组成. 壳中的不同子壳互不重叠，子壳也可以进一步划分出不同的下级子壳，由此形成 BVH 结构.

子壳的划分策略是在参数化曲线曲面中应用 BVH 的一大难点. 由于模型中各个面的表面积差异可能很大，BVH 常用的表面积启发式划分方法使得子壳中的面过于集中或者分散，从而降低优化效果，因此需要重新考虑划分策略. 本文根据当前层级包围盒中点与上层包围盒中点的相对位置，以及子壳内面的数量来决定是否应该继续细分. 这种方法可以很好地将复杂的结构根据空间局部性的原理，划分成复杂程度显著减小的子结构.

在拓扑结构中引入 BVH，可以快速地判断出模型中参与布尔运算的部分，将后续的所有计算均限制在此部分，减少资源占用，提高运行效率；另一方面，能够快速地判断直线是否与物体相交，在分类步骤中由射线法判断点与模型位置关系时也能够起到加速判断的作用.

2.1.3　拓扑求交

拓扑求交由下至上，由边边求交、边面求交、面面求交和体体求交(即 2 个拓扑实体的求交)这 4 个层级组成. 层级较低的拓扑求交由于边界信息较少，因此实现较为简单；而层级较高的拓扑求交因为包含的边界信息更多，并且存在对低层级拓扑求交的调用，所以相对来说步骤要更烦琐. 整个拓扑求交采用自底向上的顺序实现，即先实现边边求交，再以此为基础实现边面求交，以此类推. 实际调用过程则是自顶向下的，即在体体求交中会调用面面求交等步骤，面面求交中会调用层级更低的边面求交等步骤.

拓扑求交的总体思路是通过边界信息打断几何求交结果. 需要注意的是，交线之间可能存在相交的情况，此时应该将交线在相交处打断. 如图 4 所示，2 个圆锥面相交时一共有 3 条交线，但由于这 3 条交线彼此有交点，因此经过打断之后，面面求交输出 7 条交线. 若不进行打断，后续步骤中会导致根据交线划分面出现拓扑错误.

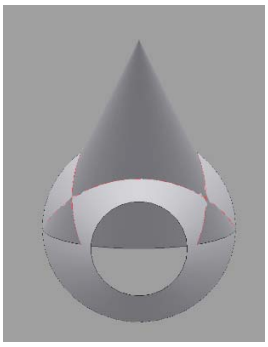


图 4　2 个圆锥面相交时交线之间出现相交

2.2　交线成环

获取 2 个模型的交线图之后，需要将交线印记到模型上并形成拓扑环.

2.2.1　多次成环

现有的交线成环算法均通过在交线的交点处进行寻路，将交线图中位于同一个面的交线统一地进行成环. 该方法的缺点是：交线生成阶段需要生成包含拓扑面原有边的拓展交线图版本，并对输入的 2 个体生成不同的结果；搜索成环算法复杂，搜索时可能需要满足一定的顺序，依赖于输入交线的顺序，面临的拓扑情况复杂；面分割情况非常复杂，一个几何面上的所有环需要统一地进行面分割，无法进行枚举分割.

因此，本文提出一种交线成环算法——多次成环，即将每条交线逐个地进行成环操作，聚焦于一条交线成环时所需面临的拓扑情况. 图 5 所示为一条交线成环时的整体算法流程图.

2.2.2　拓扑位置类型

由于交线一定位于所处理的拓扑面内部或边界，因此单一交线与该拓扑面只有 4 种位置关系:

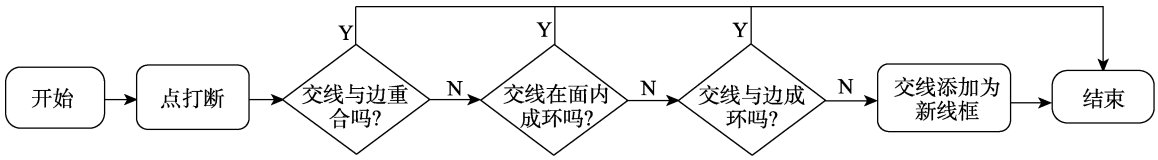
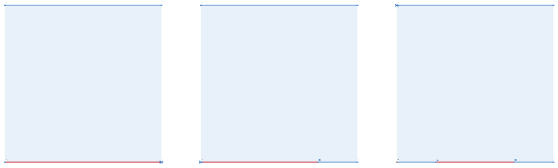


图 5　交线成环整体流程图

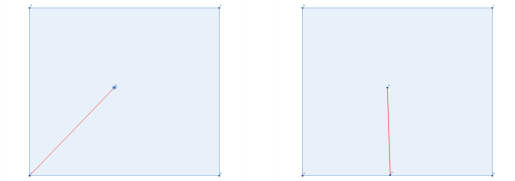
(1) 与边界重合. 包括交线与一条拓扑边完全重合、与一条拓扑边共一个端点的部分重合, 以及不共端点的部分重合, 每当有 1 个不共享的端点, 就需要对拓扑面上的拓扑边进行一次打断操作. 图 6 所示为重合的各种拓扑位置关系的示意图, 其中, 红色为交线位置.



a. 完全重合　b. 共一个端点　c. 无公共端点

图 6　重合的拓扑情况示意图

(2) 交线有一个端点在拓扑面的边上且不闭合. 包括交线与拓扑面上的边共端点与不共端点 2 种情况, 当不共端点时需要进行打断. 图 7 所示为点线成环中各种拓扑位置关系的示意图.



a. 共一个端点　b. 无公共端点

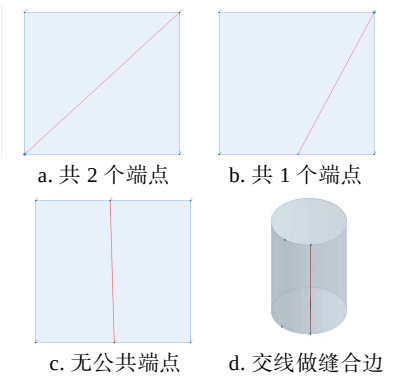
图 7　点线成环的拓扑情况示意图

(3) 交线不封闭且 2 个端点均在边界上. 除不共端点时需要进行交线打断外, 该情形下需要处理的拓扑情况较为复杂, 当 2 条交线的 2 个端点位于同一个环时, 则需要进行拆环, 之后还需要处理可能的拆面. 当 2 个端点位于不同的环时, 交线就类似于圆柱的缝合边, 需要处理环的合并. 图 8 所示为线线成环中各种可能的拓扑位置关系的示意图.

(4) 交线封闭. 交线一旦封闭, 则必产生新的环与环, 因此需要单独讨论. 图 9 所示为封闭线成环中各种可能的拓扑位置关系的示意图, 其中, 交线端点在缝合边上较为特殊, 需要先进行拆环处理.

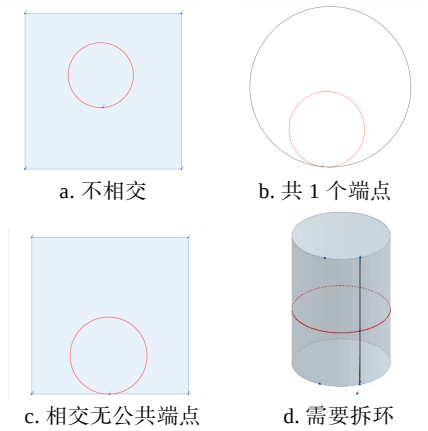
2.2.3　成环后的处理

区分出不同的拓扑情况后, 根据是否相交以及相交端点处的拓扑情况, 可以比较快速地完成成环操作. 成环后, 若此次成环操作生成了新的环, 则需要根据被操作的环以及新环的类型处理可能的拆面问题. 环的类型参考 ACIS 的设计, 分为 5 种外环、内环、u 环、v 环和 uv 环. 不同环类型所做的拓扑操作如表 3 所示. 当有新面生成之



a. 共 2 个端点　b. 共 1 个端点　c. 无公共端点　d. 交线做缝合边

图 8　交线不封闭的拓扑情况示意图



a. 不相交　b. 共 1 个端点　c. 相交无公共端点　d. 需要拆环

图 9　交线封闭的拓扑情况示意图

表 3　不同环类型所做的拓扑操作

被操作环	新环		
	外环	内环	u/v/uv 环
外环	任一分配到新面	外环分配到新面	外环分配到新面
内环	外环分配到新面	不会出现	为 u/v/uv 环找到配对环后一起分配到新面
u/v/uv 环	外环分配到新面	为 u/v/uv 环找到配对环后一起分配到新面	为 u/v/uv 环找到配对环后一起分配到新面

后，还需要遍历原面上的所有环，将位于新面上的内环重新分配到新面中.

2.3　线面分类

使用交线和交点拆分模型的线框和面后，需要根据操作类型确定模型的线框和面是否需要被保留. 由于拆分后的线框和面不会与另一个模型出现部分重合的情况，因此可以按照相对位置关系对线框和面进行划分(如表 1 所示)；然后根据不同的操作类型保留不同相对位置关系类型的线框和面(如表 2 所示)，并删除不需要的线框和面；对于差运算，还需将 tool 的所有面进行反向. 由于进行其他类型的布尔运算时，如保留 2 个模型的非交集等，只需要设定对应需要保留的相对位置类型，因此问题归结为对线框和面的位置关系判断.

线框的相对位置关系可以分成 2 部分：线框与另一个模型的线框的位置关系，以及线框与另一个模型的实体的位置关系. 其中，前者可以通过比较当前线框的中点与另一个模型的线框的中点是否在容差范围内重合，确定其位置关系；后者可以取线框的中点与另一个模型进行点与模型相对位置关系判断，推出线与模型相对位置关系.

面分类算法的流程如图 10 所示. 受到文献[14]的启发，在判断面的位置关系前进行面分组，可以大大削减问题规模.

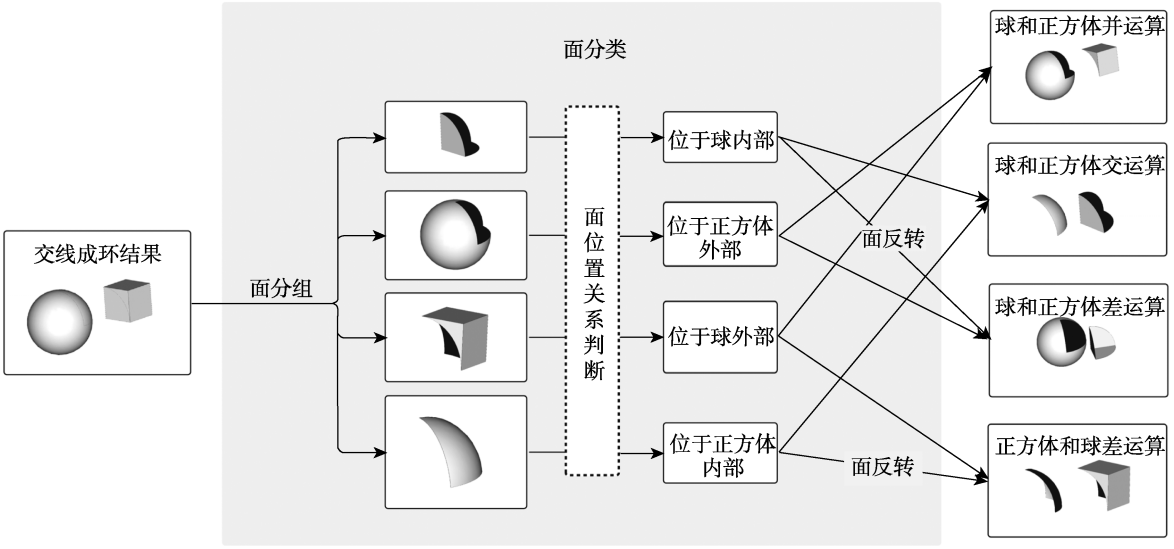


图 10　面分类算法流程图

定义一个无向图 $G=(F,E)$，其中，F 表示形体的拓扑面集合，E 表示 $\{\langle f_1,f_2\rangle \mid f_1,f_2\in F$ 且 f_1 和 f_2 共享非交线的边$\}$，则 G 中同一个连通分量内的面具有相同的相对位置关系. 通过任取连通分量内的任一面进行相对位置关系判断，可以得到同一连通分量内其他面的相对位置关系，有效地削减了问题的规模，将判断次数从 $|F|$ 减少为 G 的连通分量的个数.

判断面的相对位置关系的一种直观的方法是，取面内部的一个点，用点体的相对位置关系推出面体的相对位置关系. 但这种方法存在 2 个问题：(1) 该方法的正确性依赖于该点是否正确地在面内部，而这不是一个容易的问题，特别当该点在交线附近时；(2) 之后步骤的点体相对位置关系判断依赖于几何求交，耗时较长.

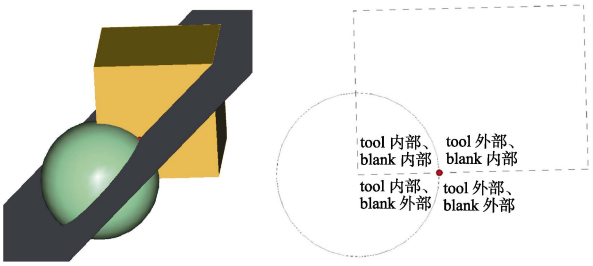
基于以上原因，本文利用交线局部信息进行面的相对位置关系判断.

进行位置关系判断时，为了规避奇异情况，先进行重合关系判断，再进行内外关系判断. 2 个面重合当且仅当二者的几何面重合，同时，二者所有边均为交线且其几何和拓扑一一对应. 重合关系判断可以直接使用前面步骤的几何面重合信息和交线信息.

对于不存在包含交线的面的连通分量，可以取该连通分量类中任一面内的一点，用点体的相对位置关系判断推出面体相对位置关系. 注意，该情况下进行取点时，不会遇到诸如靠近交线等问题导致位置关系判断困难.

对于存在包含交线的面的连通分量，使用交线的局部信息进行判断包含交线的面的相对位置关系. 如图 11a 所示，以球(tool)与正方体(blank)进行布尔运算为例，假设一个与交线在交线一点 Q (如图中的红点所示)处垂直的平面 P，其与交

线相连的 2 个模型上的所有面会产生相应的交线(如图 11b 所示), 这些交线在点 Q 处会将平面 P 划分为多个区域. 根据点 Q 在这些面的法向, 可以将划分区域标记为 tool 外部、tool 内部、blank 外部和 blank 内部, 从而获得面的位置关系.



a. 球与正方体布尔运算 b. 假想平面内划分内外示意

图 11 交线局部信息划分内外部分示意图

要注意以下 4 点:

(1) 该方法一次即可获取多个与交线相连的面(通常为 4 个面, 在非流形的情况可能会更多或更少)的位置关系.

(2) 点 Q 的选取应避开奇异点, 而当交线正好为曲面参数域的边界时, 可以使用其附近一点的局部信息代替, 无需对近似点使用点面位置关系判断是否在面内部.

(3) 平面 P 在实际实现时并不存在. 算法需要计算点 Q 处各个面对应的有向交线边方向 $\boldsymbol{n}_{\text{coedge}}$、法向 $\boldsymbol{n}_{\text{face}}$、面内方向 $\boldsymbol{n}_{\text{inside}}=\boldsymbol{n}_{\text{face}}\times\boldsymbol{n}_{\text{coedge}}$ 和曲率 K. 任取点 Q 处一面的 $\boldsymbol{n}_{\text{coedge}}$ 为法向 $\boldsymbol{N}$, 对各个面根据面内方向 $\boldsymbol{n}_{\text{inside}}$ 进行逆时针排序, 当 $\boldsymbol{n}_{\text{inside}}$ 的差值在容差范围内时, 需要考虑 K. 遍历排序后的面, 若当前面的 $\boldsymbol{n}_{\text{coedge}}$ 与 $\boldsymbol{N}$ 同向时, 则认为其到其之后的面之间的区域为当前面所在模型的外部; 否则, 情况相反.

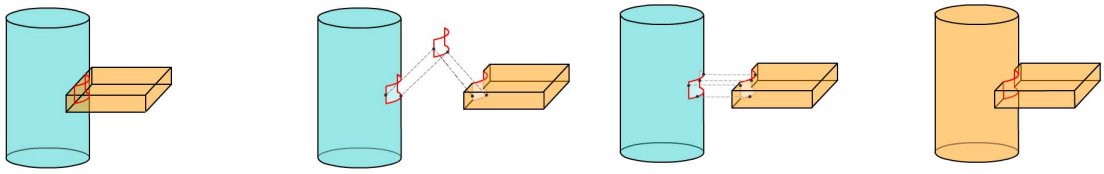
(4) 在理想情况下, 因为算法不会出现不一致内外关系判断结果, 所以该方法在一定程度上可以检查出非法输入. 此外, 还有可能是容差导致同一个面在不同的交线处计算得到的内外关系不一致, 正如文献[16]指出, 该方法的一个缺陷就是可能需要更高阶的信息. 处理这种不一致时, 一个折中的方法是采用多数原则的方法, 在多数正确的情况下忽略异常值.

2.4 拓扑合并

对参与运算的模型中边面进行分类后, 根据表 2 对相应类别的数据进行舍弃, 可以获得所有被包含在布尔运算结果中的数据和几何数据; 但是, 由于这些实体在拓扑层面是相互分离的, 因此需要对相互分离的数据进行合并, 并完成拓扑信息重建, 使得这些实体部分最终能够属于同一个实体, 且保证拓扑信息的正确性. 本文算法中, 该过程被分为边拓扑合并、顶点拓扑合并、块壳关系重组和运算结果的正规化 4 个步骤.

2.4.1 边拓扑合并

边的拓扑合并时, 最直观的方法是对 tool 体和 blank 体的边进行逐一检索, 匹配几何和端点都相同的边, 然后进行合并. 然而在布尔运算中, 物体面类别的变化只会发生在与另一个物体的交线两侧且 2 个物体有交的部分; 另外, 布尔运算的结果只会依据沿着交线进行合并. 因此在边拓扑合并之前, 在交线分割面过程中, 可以将每条交线与 2 个模型的边的对应关系记录下来, 并结合后续面的分类与移除得到 tool 体和 blank 体中需要进行合并的边对, 直接指定 2 条边进行合并, 节省了相同边的匹配时间, 该过程如图 12 所示.



a. 交线计算 b. 交线关联 c. 合并边匹配 d. 边合并

图 12 利用交线信息直接进行边合并匹配

2.4.2 顶点拓扑合并

当顶点的拓扑合并时, 需要配对坐标位置相同的顶点, 且合并其上层拓扑. 考虑顶点数目较大时, 对所有点的坐标进行逐一比对的时间开销较大, 因此需要对点的匹配过程进行一定处理. 如图 13 所示, 使用基于排序的方法, 根据点的坐标对所有点进行排序, 选取一个主要的比较维度(如按照 x 坐标排序), 如果该维度在容差范围内相等, 则按照次要维度(如 y 坐标)排序, 以此类推. 由于排序使得接近的点在列表中也位于接近的位置,

因此在排序后，只需要逐一检查相邻点的距离是否在容差范围内重合，即可找出在空间上重合的点.

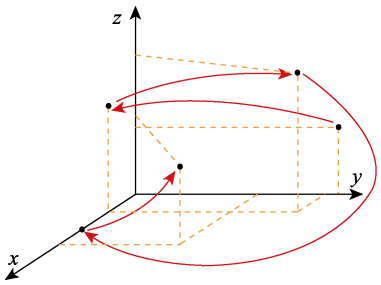
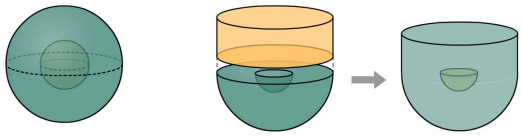


图 13　基于坐标的点排序示意图

2.4.3　块壳关系重组

通常，部分布尔运算会导致运算结果产生新的壳，且这些壳在物理空间上会有新的包含关系，将导致块与壳的关系发生变化，如图 14 所示，此时需要对块壳关系进行重组.



a. 差运算导致内表面产生　　b. 并运算导致新的内表面产生

图 14　布尔运算导致新壳产生

块壳关系重组最直观的方法是在完成所有合并操作之后，对所有的块两两进行内外包含判断，并合并在物理空间上相互包含的块；然而，对于有多个块的实体，这种方法显然会产生大量的时间开销. 对于这部分的优化，其核心思想是利用相交信息减少块之间干涉检查的次数，提升布尔运算的整体效率.

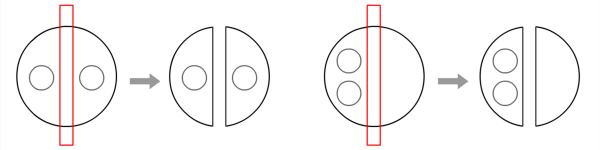
完成了线面分类之后，可以通过相交信息，根据一个块是否与另一个块产生相交，分为块有交和块无交进行讨论. 由于块无交无法通过交线确定出与其关联的块，因此对其需要与其余块进行逐一内外判断；而对于块有交的情况，则可以利用交线信息减少干涉检查的次数.

可以通过块的交线信息将 2 个块关联起来，利用这个性质，块中所有壳只需要和与其相交的块进行干涉检查，大大减少了干涉检查所需要的次数. 根据不同类型的布尔运算的不同性质，块有交又可以分为布尔并运算及布尔交和差运算 2 个方面进行讨论.

(1) 布尔并运算. 对于并运算中块上有交线的情况，块可能包含多个相对位置关系的等价类，可能会因为线面删除操作而被打断成多个新壳. 与布尔交和差运算不同，布尔并是一种增量运算，即原本属于模型空间的部分在运算结束以后也必然属于该实体，因此，原本连通的区域也不会因为并运算而变得不连通，只会导致原本不连通的区域可能变得连通. 所以对于上述的新壳，在并运算结束后其必然依旧属于同一个块.

针对这种性质，可以在线面分类操作中单独处理并运算的情况，即对于并运算中的删除操作，如果该删除导致原有块的壳连通性被破坏，则此时删除后产生的新壳仍属于原先的块. 因此直接在块有交的情况下省略干涉检查的过程，极大地提升了并运算的效率.

(2) 布尔交和差运算. 与并运算不同，交运算和差运算都是减量运算，即结果中可能只保存部分的原始模型，从而改变空间的连通性，将一个块变成多个块. 图 15 所示为内部存在 2 个球形空腔的球体与长方体薄板布尔差的示意图，球体被长方体薄板截断. 其中，图 15a 球体内部 2 个空腔在薄板异侧，而图 15b 球体内部 2 个空腔在薄板同侧. 这 2 种情况具有相同的交线，但由于其内部空腔处在薄板的不同侧，单纯根据交线信息无法判断 2 个内壳在布尔运算结束后分别属于哪个块，因为单从交线信息来看，图 15a 和图 15b 在拓扑上等价，所以内外包含判断也是无法避免的.



a. 结果内表面位于 2 个不同块　　b. 结果内表面位于同一块中

图 15　相同交线下发生不同的块壳关系重组

尽管如此，通过利用交线信息也能减少干涉检查的次数. 从图 15 可以看出，无论一个块有多少个壳，只需要对一个块中无交的壳和该块有交的壳所形成的新块进行干涉检查. 设一个块有 m 个无交的壳，有交的壳上需要被保留的相对位置关系的等价类数量为 n，则内外包含判断的次数为 $O(mn)$.

图 16 所示为利用交线信息后的内外判断处理过程. 该过程的步骤如下:

Step1. 对于包含多个壳的块，找出这些壳中没有交线的壳 S_1 和有交线的壳 S_2.

Step2. 在 S_2 中找出所有需要被保留的相对位置关

系的等价类，并从每个等价类中取出该等价类中任意一个元素，将这些等价类放到一个数组 L 中.

Step3. 待删除操作之后，将删除前属于 S_1 的元素所属的块和 L 中的元素所处的块进行内外包含判断.

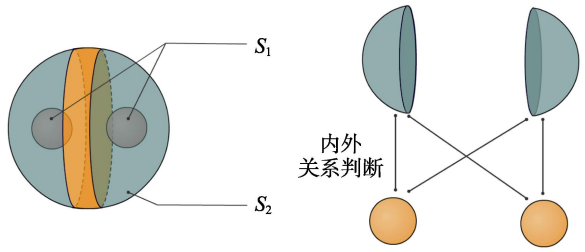
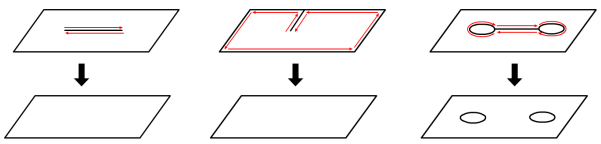


图 16 利用交线信息的块壳关系重组方法示意图

2.4.4 运算结果规则化

在布尔运算的结束阶段，通常还需要去除运算过程中产生的不必要的边和顶点，以实现布尔运算结果的规则化，包含点移除和边移除 2 个过程. 如果一条边是流形的，并且在该边关联的面的几何表达是等价的，那么这条边是可合并的；如果一个顶点被 2 条边共享，这 2 条边有等价的几何形状，并且顶点不位于边凸度变化的点上，那么这个顶点是可合并的. 当可合并的边所关联的 2 个面不为同一个面时，移除该边，并修改对应拓扑使得将这 2 个面合并为一个面. 而当可合并的边的 2 条共边属于同一个面环的情况时，则结果更加复杂：首先，若共边所在的环仅包含这 2 条共边的情况，则这个环构成了一个只有一段边的缝合线，此时直接将该面环从面的环列表中进行移除即可；其次，如果 2 条共边互为前驱或者后继，则这 2 条共边在面环中构成了刺边，此时需要将这 2 条边从环的共边链表环中移除；最后，如果 2 条边之间没有前驱后继关系，清除这 2 条共边所在的边会使得原有的环变成 2 个环，需要对环的拓扑进行处理，并更新面的环列表，如图 17 所示.



a. 缝合线清理 b. 刺边清理 c. 环重合边清理

图 17 共边所在的面相同时的边清理

3 实验及结果分析

基于 ACIS 的求交模块实现本文算法，并与 Parasolid, ACIS 和 OCCT 的布尔运算算法在开孔场景和真实船舶设计场景进行实验.

3.1 程序实现与实验环境

本文实验环境如下：CPU 为 Intel Core i7-10700K，RAM 为 32 GB. 所有程序由现代 C++实现，通过 MSVC 编译器编译，并在 Windows11 操作系统上运行.

3.2 实验评估

3.2.1 开孔操作实验

在实际设计场景中，一个极其常见且重要的设计操作是在局部建模时对模型进行开孔. 常见的做法是使用布尔运算减去一个圆柱体，如图 18 所示. 开孔的效率和稳定性能在一定程度上反映布尔运算效率和稳定性. 本文分别在长方体薄板和摆动体薄板(其中一面为 NURBS 曲面)上进行开孔实验，结果如图 19 所示.

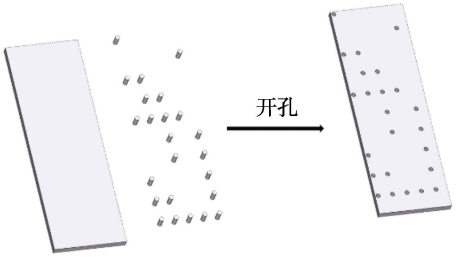
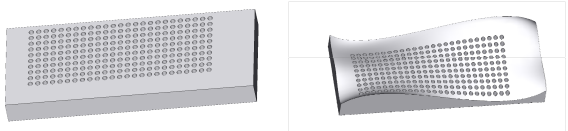


图 18 开孔操作示意图



a. 长方体薄板 b. 摆动体薄板

图 19 开孔结果图

在长方体薄板和摆动体薄板上进行开孔实验，4 种算法的耗时结果如图 20 所示. 可以看出，商用内核 Parasolid 和 ACIS 算法的效率极高，随着开孔总数量的增加，耗时增加较缓，说明其线性时间复杂度的常数更小；同时，本文算法耗时随开孔数量约呈线性增长，明显优于 OCCT 的表现.

3.2.2 真实船舶设计模型实验

本节使用真实船舶设计场景下的 CAD 模型进行布尔运算实验，包含小样装配体的组装和组装后复杂形体的布尔运算，以及复杂船体模型组装. 其中，所有模型均涉及 NURBS 曲线曲面.

(1) 小样装配体

船舶设计中存在组装小样装配体的应用场景，主要通过布尔并运算进行组装. 本文选取简单、中等和复杂 3 种模型进行实验. 其中, BOOLXY 系列模型为简单模型，组装后的可视化结果如图 21 所

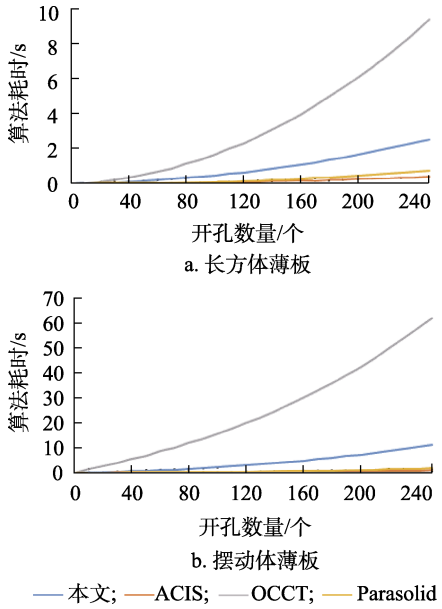


图 20　在 2 种薄板上进行开孔实验时 4 种算法耗时对比

示；M 系列模型为中等模型，C 系列模型为复杂模型，组装后的可视化结果如图 22 所示. 装配体组装的耗时结果如表 4 所示.

本文还对组装后的模型进行布尔运算实验，进一步评估本文算法的正确性与效率，4 种算法耗时结果如表 5 所示.

(2) 复杂船体模型组装实验

在船舶设计中，涉及使用布尔运算进行复杂船体模型组装. 本文选取的复杂船体模型如图 23 所示，其由 68 个模型组成，构成的曲线曲面均为 NURBS 曲线曲面，其中，曲线数量 2 734 条，曲面数量 1 115 个. 4 种算法耗时如表 6 所示.

(3) 真实船舶设计模型实验总结

从图 21~图 23，以及表 4~表 6 可以看出，商业级几何内核 Parasolid 和 ACIS 效率展现出极高的水平；与 OCCT 这一开源几何引擎相比，本文算法的效率

图 21　BOOLXY 系列模型结果

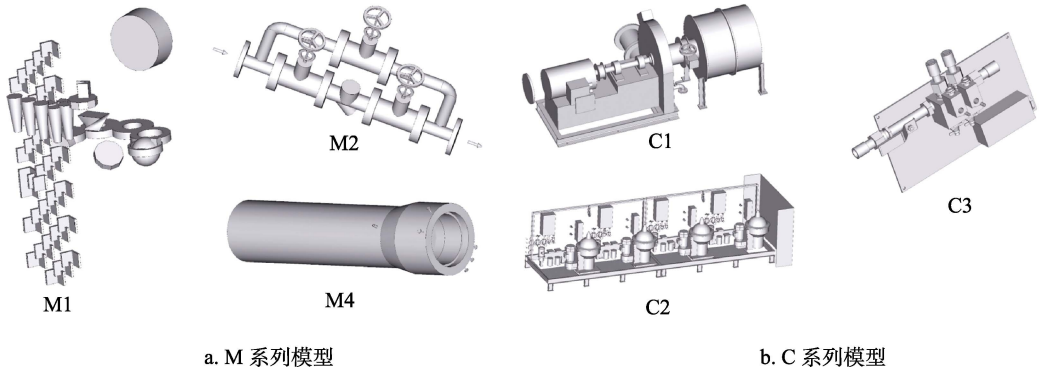


图 22　M 系列和 C 系列模型组装结果

表 4　4 种算法小样装配体组装耗时对比

模型名	组装次数	耗时/ms			
		Parasolid	ACIS	本文	OCCT
BOOLXY001_1	6	8.998	5.225	12.114	54.866
BOOLXY001_2	9	1.407	7.396	14.352	91.878
BOOLXY002_1	10	20.760	16.160	26.216	145.249
BOOLXY002_2	2	0.511	0.593	1.198	5.437
BOOLXY003_1	3	1.642	1.164	1.863	22.031
BOOLXY003_2	2	6.931	3.971	12.974	112.728
M1	57	1 099.206	1 092.760	1 945.640	3 304.218
M2	50	18.823	332.717	513.586	1 952.282
M4	19	14.275	31.274	83.018	515.319
C1	59	26.549	65.749	191.292	2 663.570
C2	454	534.877	1 340.648	3 357.303	144 051.130
C3	85	46.859	127.312	402.606	4 158.149

表 5　不同算法小样装配体组装结果间布尔运算实验对比

模型名	涉及边数	涉及面数	耗时/ms			
			Parasolid	ACIS	本文	OCCT
BOOLXY001	33	32	0.323	3.469	14.857	143.595
BOOLXY002	96	49	0.024	4.355	17.108	121.696
BOOLXY003	48	27	0.621	0.335	1.200	28.363
M1-M2	1722	646	0.033	12.325	46.800	421.009
M1-M4	1277	485	0.152	31.477	153.047	1145.089
M2-M4	545	263	0.030	1.087	3.558	73.298
C1-C2	4069	1998	0.073	40.854	407.442	1565.943
C1-C3	1674	730	0.050	9.664	45.973	351.916
C2-C3	4347	2098	0.072	13.276	171.799	1407.815

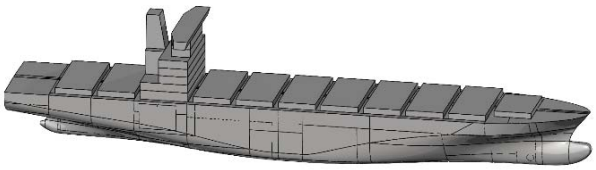


图 23　复杂船体模型组装结果

表 6　4 种算法复杂船体模型组装总耗时对比

算法	组装总耗时/s
Parasolid	6.1
ACIS	21.8
本文	23.7
OCCT	597.8

较高; 尽管在一些场景下本文算法能与 Parasolid 和 ACIS 性能相当, 但在复杂场景下仍有一定的差距.

4　结　语

本文提出一种高效的 B-Rep 模型的布尔运算算法. 使用层次化包围盒干涉检查、面分组、交线局部信息进行相对位置关系判断等, 极大地减少了几何求交的耗时; 同时, 通过构建交线图、多次成环等提高了算法的稳定性. 本文基于实际三维 CAD 模型, 与几何建模引擎 Parasolid, ACIS 和 OCCT 进行对比实验. 实验结果表明, 与 OCCT 的布尔运算相比, 本文算法在效率上具有一定的优势; 尽管在一些场景下其性能与 Parasolid 和 ACIS 相当, 但在复杂场景下仍有一定的差距.

本文算法也存在一些不足之处: 在生成交线图阶段, 可能会出现容差导致的与实际情况不符的交线, 需要更恰当的处理; 在交线成环阶段, 可

能在一些情形中可以利用交线信息等直接进行面分割，而不需要判别环类型；在线面分类阶段，可以考虑利用交线信息处理线框分类；在拓扑合并阶段，可以用更高效的壳包含关系进行判断.

参考文献(References):

[1] Stroud I. Boundary representation modelling techniques[M]. London: Springer, 2006

[2] Cheng Jin, Ye Huqiang, Tan Jianrong, *et al*. Review of research progress and development trends of 3D CAD technology[J]. Journal of Mechanical Engineering, 2023, 59(23): 158-185(in Chinese)
(程锦, 叶虎强, 谭建荣, 等. 三维 CAD 技术研究进展及其发展趋势综述[J]. 机械工程学报, 2023, 59(23): 158-185)

[3] Urick B, Marussig B, Cohen E, *et al*. Watertight boolean operations: a framework for creating CAD-compatible gap-free editable solid models[J]. Computer-Aided Design, 2019, 115: 147-160

[4] Cherchi G, Pellacini F, Attene M, *et al*. Interactive and robust mesh booleans[J]. ACM Transactions on Graphics, 2022, 41(6): Article No.248

[5] Biermann H, Kristjansson D, Zorin D. Approximate Boolean operations on free-form solids[C] //Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques. New York: ACM Press, 2001: 185-194

[6] Requicha A A G, Voelcker H B. Boolean operations in solid modeling: boundary evaluation and merging algorithms[J]. Proceedings of the IEEE, 1985, 73(1): 30-44

[7] Feito F R, Ogáyar C J, Segura R J, *et al*. Fast and accurate evaluation of regularized Boolean operations on triangulated solids[J]. Computer-Aided Design, 2013, 45(3): 705-716

[8] Ogayar C J, Segura R J, Feito F R. Point in solid strategies[J]. Computers & Graphics, 2005, 29(4): 616-624

[9] Segura R J, Feito F R. Algorithms to test ray-triangle intersection. Comparative study[J]. Journal of WSCG, 2001, 9(3): 76-81

[10] Nehring-Wirxel J, Trettner P, Kobbelt L. Fast exact Booleans for iterated CSG using octree-embedded BSPs[J]. Computer-Aided Design, 2021, 135: Article No.103015

[11] Naylor B, Amanatides J, Thibault W. Merging BSP trees yields polyhedral set operations[J]. ACM Siggraph Computer Graphics, 1990, 24(4): 115-124

[12] Campen M, Kobbelt L. Exact and robust (self-)intersections for polygonal meshes[J]. Computer Graphics Forum, 2010, 29(2): 397-406

[13] Trettner P, Nehring-Wirxel J, Kobbelt L. EMBER: exact mesh Booleans via efficient & robust local arrangements[J]. ACM Transactions on Graphics, 2022, 41(4): Article No.39

[14] Zhou Q N, Grinspun E, Zorin D, *et al*. Mesh arrangements for solid geometry[J]. ACM Transactions on Graphics, 2016, 35(4): Article No.39

[15] Zuo Zheng. Intersection, classification and application of geometric modeling[D]. Beijing: Tsinghua University, 1998(in Chinese)
(左征. 几何造型中的求交分类及其应用[D]. 北京: 清华大学, 1989)

[16] Yang Sheng. Classification methods for Boolean operations between solids[D]. Beijing: Tsinghua University, 2010(in Chinese)
(杨胜. 实体布尔运算中的分类方法研究[D]. 北京: 清华大学, 2010)