|  |
| --- |
| 电子科技大学  UNIVERSITY OF ELECTRONIC SCIENCE AND TECHNOLOGY OF CHINA  硕士学位论文  MASTER THESIS |

论文题目 **微波管仿真环境下实时碰撞检测技术的**

**研究及应用**

学科专业  **电子科学与技术**

学 号 **201821020617**

作者姓名 **聂 铨**

指导教师 **徐 立 副 教 授**

分类号 密级

UDC注1

学 位 论 文

**微波管仿真环境下实时碰撞检测**

**技术的研究及应用**

（题名和副题名）

**聂 铨**

（作者姓名）

指导教师  **徐 立 副 教 授**

**电子科技大学 成 都**

（姓名、职称、单位名称）

申请学位级别 **硕士**  学科专业 **电子科学与技术**

提交论文日期  论文答辩日期

学位授予单位和日期

答辩委员会主席

评阅人

注1：注明《国际十进分类法UDC》的类号。

|  |
| --- |
| **Research and application of real-time collision detection technology in microwave tube simulation environment** |

A Master Thesis Submitted to

University of Electronic Science and Technology of China

|  |  |
| --- | --- |
| Discipline: | **Electronic Science and Technology** |
| Author: | **Quan** **Nie** |
| Supervisor: | **Prof. Li Xu** |
| School: | **School of Electronic Engineering of UESTC** |

独创性声明

本人声明所呈交的学位论文是本人在导师指导下进行的研究工作及取得的研究成果。据我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得电子科技大学或其它教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示谢意。

作者签名： 日期： 年 月 日

论文使用授权

本学位论文作者完全了解电子科技大学有关保留、使用学位论文的规定，有权保留并向国家有关部门或机构送交论文的复印件和磁盘，允许论文被查阅和借阅。本人授权电子科技大学可以将学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存、汇编学位论文。

（保密的学位论文在解密后应遵守此规定）

作者签名： 导师签名：

日期： 年 月 日

摘 要

碰撞检测在CAD/CAM、运动规划和虚拟制造等领域具有广泛的应用，能精确检测到物体之间的碰撞和干涉现象并返回碰撞信息。随着计算机的性能、数值建模和模拟仿真技术的不断发展，计算机辅助设计技术已被广泛应用到微波管的设计和制造过程中，能够显著提高制管效率、降低返工风险和优化仿真的性能。在微波管仿真环境中，研究模型之间的碰撞检测问题对改进微波管的建模设计和装配细节具有重要的指导作用，能够确保微波管组件设计和装配过程的合理性、可行性和成品率。本论文对碰撞检测和微波管参数化建模技术进行了深入研究，在微软基础类库的基础上，集成ACIS和HOOPS引擎自主开发一款三维微波管参数化建模软件平台，并基于此平台设计了一种高效的混合层次包围体算法实现微波管组件之间的碰撞检测。具体的工作内容和创新点如下：

（1）本文首先综述和分析了碰撞检测算法的研究发展现状，总结了当前研究存在的问题与挑战，提出了本论文研究的思路和方向。

（2）通过对碰撞检测算法的研究和分析，针对单一种类包围体算法的实时性和精确度的性能不足问题，本文设计了一种AABB-OBB混合的层次包围体树算法。在不同的场景下对算法性能进行测试，验证了混合碰撞检测算法的性能和稳定性。

（3）微波管的仿真过程往往需要不断根据仿真结果调整模型尺寸、位置等参数以进行迭代仿真。基于ACIS几何建模引擎、HOOPS图形套件和VC++软件开发技术，本文自主开发了微波管参数化建模的软件平台，此软件包括响应用户操作的图形用户界面，进行几何特征造型与操作的参数化建模系统，以及支持xml和sat文件格式的读写功能的特征文件读写模块。此软件现已应用到课题组研发的MTSS软件中

（4）在本文所实现的微波管参数建模软件的基础上，应用混合碰撞检测算法自主开发实现了碰撞检测求解器模块。模块功能主要包括静态干涉检查、模型最近距离计算、基本干涉检查和间隙分析，证明了算法的实用性。

（5）最后，对行波管的各个组件进行干涉检查以验证算法的适用性和可靠性。在基本干涉检查实例中，检测了电子枪模型的零部件之间的面干涉情况；在对行波管的各个组件模型执行间隙分析的实例中，检测了零部件之间的间隙距离和干涉现象，可视化分析和评审了微波管零部件之间的四种类型的干涉结果。

**关键词：**碰撞检测，微波管，ACIS，HOOPS，干涉检查

**Abstract**

Collision detection is a key technology in the field of CAD/CAM, motion planning and virtual manufacturing. It can accurately detect collision and interference phenomena between objects and return relevant information. With the continuous development of computer performance, numerical modeling and simulation technology, computer-aided design technology in the field of microwave tubes has been widely used in the design and manufacturing process of microwave tubes, which can significantly improve manufacturing efficiency, reduce the risk of rework and optimize the performance of the simulation. In the microwave tube simulation environment, the collision detection problem between the solid models plays an important guiding role in improving the modeling design and assemble of the microwave tube, and ensure the rationality, feasibility and yield of the design and assemble of the parts of the microwave tube.

This paper has conducted an in-depth research on the collision detection technology and parametric modeling of the microwave tubes. On the basis of Microsoft Foundation Classes, the paper has integrated ACIS and HOOPS engines to independently develop a 3D microwave tube’s parametric modeling software platform. Based on this platform, the paper has designed an efficient hybrid bounding volume hierarchy structure to achieve collision detection between the components of microwave tube. The specific research content and innovation points are as follows:

(1) The paper reviews and analyzes the research and development status of collision detection algorithms, summarize the existing problems and challenges of current research, and put forward the ideas and directions of the research in this thesis.

(2) This paper research and analysis of the collision detection algorithm. In order to solve the limitation of real-time and accuracy of a single type of bounding volume algorithm, this paper introduces a hybrid bounding volume hierarchical algorithm which integrates AABB bounding volume and OBB bounding volume. Through carrying out collision detections between models under different scenarios, verifying the performance and stability of the algorithm.

(3) The simulation process of the microwave tube often requires iterative simulation by continuously adjusting the parameters such as size and position of the model according to the simulation results. Based on the ACIS geometric modeling engine, HOOPS graphics suite and VC++ software development technology, this paper independently developed a software platform for parametric modeling of microwave tubes. This software includes a graphical user interface that responds to user operations, parameters modeling system that performs geometric feature modeling and operating, and feature file read-write module that supports the read-write function of xml and sat file formats. This software has now been applied to the MTSS software developed by the research group.

(4) On the basis of the software platform, the hybrid collision detection algorithm is independently developed to realize the collision detection solver. The functions of this module include static collision checking, touch solution solving, closest distance calculation, interference checking and gap analysis, which verify the practicability and reliability of the algorithm in this paper.

(5) Finally, interference check is performed on each component of the traveling-wave tube to verify the applicability and reliability of the algorithm. In the basic interference inspection example, the interference between surface and surface of the electron gun models are detected; in the example of performing gap analysis on each component model of the traveling wave tube, the gap distance and interference phenomenon between the parts are detected, and the visualization Four types of interference results between microwave tube components are analyzed and reviewed.

**Keywords:** collision detection, microwave tube, ACIS, HOOPS, interference check

目 录

[第一章 绪论 1](#_Toc68772593)

[1.1 课题研究的背景及意义 1](#_Toc68772594)

[1.2 国内外研究现状 2](#_Toc68772595)

[1.2.1 基于时间域的碰撞检测算法 2](#_Toc68772596)

[1.2.2 基于空间域的碰撞检测算法 3](#_Toc68772597)

[1.2.3 存在的问题与挑战 5](#_Toc68772598)

[1.3 本文的主要贡献和创新 6](#_Toc68772599)

[1.4 本文的组织结构 7](#_Toc68772600)

[第二章 相关理论研究与分析 8](#_Toc68772601)

[2.1 HOOPS图形引擎的介绍 8](#_Toc68772602)

[2.2 ACIS建模引擎的介绍 11](#_Toc68772603)

[2.3 集成HOOPS和ACIS 13](#_Toc68772604)

[2.4 碰撞检测算法理论 14](#_Toc68772605)

[2.4.1 基本概念 14](#_Toc68772606)

[2.4.2 包围体技术 14](#_Toc68772607)

[2.4.3 性能评价函数 18](#_Toc68772608)

[2.4.4 GJK算法的基本概念 18](#_Toc68772609)

[2.5 本章小结 20](#_Toc68772610)

[第三章 混合碰撞检查算法的研究与设计 22](#_Toc68772611)

[3.1 混合层次包围体树的设计 22](#_Toc68772612)

[3.1.1 混合层次包围体树的结构 22](#_Toc68772613)

[3.1.2 算法流程 23](#_Toc68772614)

[3.2 层次包围体树的构造 24](#_Toc68772615)

[3.3 层次包围体树的遍历和更新 25](#_Toc68772616)

[3.3.1 层次包围体树的遍历 25](#_Toc68772617)

[3.3.2 层次包围体树的更新 26](#_Toc68772618)

[3.4 三角形图元之间的相交测试 28](#_Toc68772619)

[3.5 GJK算法计算分离距离和穿透距离 28](#_Toc68772620)

[3.5.1 GJK算法计算分离距离 28](#_Toc68772621)

[3.5.2 GJK算法计算穿透深度 31](#_Toc68772622)

[3.6 碰撞检测实验结果 33](#_Toc68772623)

[3.7 本章小结 35](#_Toc68772624)

[第四章 微波管参数化建模软件平台设计与实现 36](#_Toc68772625)

[4.1 系统总体架构设计 36](#_Toc68772626)

[4.1.1 系统的需求分析 36](#_Toc68772627)

[4.1.2 系统框架 37](#_Toc68772628)

[4.2 图形用户界面的实现 38](#_Toc68772629)

[4.2.1 组合菜单栏 38](#_Toc68772630)

[4.2.2 属性浏览器 39](#_Toc68772631)

[4.2.3 特征导航器 40](#_Toc68772632)

[4.2.4 对话框界面 42](#_Toc68772633)

[4.3 视图操作和模型渲染 42](#_Toc68772634)

[4.4 基于特征的参数化建模系统 44](#_Toc68772635)

[4.4.1 建模系统的设计 44](#_Toc68772636)

[4.4.2 建模系统的实现 46](#_Toc68772637)

[4.4.3 捕捉点功能的实现 48](#_Toc68772638)

[4.4.4 特征历史流的管理 49](#_Toc68772639)

[4.5 碰撞检测求解器模块 50](#_Toc68772640)

[4.6 特征文件的存储和读取 53](#_Toc68772641)

[4.7 本章小结 54](#_Toc68772642)

[第五章 微波管仿真环境下碰撞检测算法的应用实例 55](#_Toc68772643)

[5.1 概述 55](#_Toc68772644)

[5.2 基本干涉检查应用实例 55](#_Toc68772645)

[5.3 零部件间隙分析的应用实例 58](#_Toc68772646)

[5.3.1 软干涉检查结果 61](#_Toc68772647)

[5.3.2 接触干涉检查结果 62](#_Toc68772648)

[5.3.3 硬干涉检查结果 62](#_Toc68772649)

[5.3.4 包含干涉检查结果 64](#_Toc68772650)

[5.4 本章小结 65](#_Toc68772651)

[第六章 总结和展望 66](#_Toc68772652)

[致 谢 68](#_Toc68772653)

[参考文献 69](#_Toc68772654)

[攻读硕士学位期间取得的成果 73](#_Toc68772655)

第一章 绪论

1.1 课题研究的背景及意义

微波管是一类工作在微波波段的电真空器件，在电子对抗、卫星导航和民用的微波炉中是关键的功率放大器件，主要包括磁控管、行波管和速调管等。然而，微波管的生产和制造过程复杂、材料属性众多、不同应用场景中微波管的尺寸和结构有较大差异，种种因素对微波管器件的性能、稳定性和可靠性带来了巨大挑战[1]。在引入计算机辅助设计技术之前，主要依靠人工在实际的生产过程中检测出微波管零部件之间的干涉现象和影响仿真性能的不合理的部件间隙距离，这种方法的效率显然很低下。此外，工程师在参数设计和生产过程中的微小误差都可能导致零部件之间的干涉从而造成无法成功组装的问题，这使微波管的生产成本高昂、研发周期长且成品率低。随着计算机的硬件性能和软件开发效率的逐步提高，传统人工检测和设计的方法已经不能满足需求，人们对生产成本低、制作周期短、设计效率和性能高的微波管的需求越来越大，微波管的仿真模拟技术比如计算机辅助设计（Computer Aided Design，CAD）和计算机辅助工程（Computer Aided Engineering，CAE）等被不断应用在微波管的设计和生产过程中[2]。通过计算机模拟和仿真软件进行组件的参数化建模设计、结构划分和尺寸计算并不断测试、调整和优化，能够帮助工程师反复仿真评估微波管的功率、带宽、增益等性能，改进参数设计以避免干涉现象，能够大大降低微波管生产的返工风险、研发周期并大幅提高微波管的性能。电子科技大学计算机仿真实验室自主研发的一款名为MTSS（Microwave tube simulator suite）的微波管CAD软件，支持对微波管的各个组件进行仿真模拟，提供螺旋线行波管、速调管和回旋管等器件的仿真计算，已广泛应用在国内各大制管单位的微波管研发制造过程中[3]。目前，一些成熟的三维CAD软件平台基本都具备对模型进行干涉检测的功能，例如Pro/E、UG、CATIA、SolidWorks等主流CAD软件都实现了装配体的零部件之间的干涉检查功能[4]。因此，将碰撞检测算法应用于三维微波管仿真模拟器MTSS中以实现拥有自主知识产权的干涉检查功能，并帮助工程师不断优化微波管的参数设计和装配过程具有重要的现实意义。

碰撞检测是基于物理的虚拟仿真、CAD/CAM、机器人运动规划等领域的核心技术和不可避免的技术瓶颈之一。碰撞检测算法不仅要回答两个对象是否碰撞的问题，还要回答碰撞何时发生、碰撞区域在哪里以及接近碰撞的物体之间的最近距离等问题。基于应用领域的不同，相关研究人员针对各种类型的碰撞检测问题设计了多种碰撞检测算法。从时间角度看，主要可以分为静态碰撞检测算法和动态碰撞检测算法，其中的动态检测算法可细分为离散碰撞检测算法和连续碰撞检测算法两类[5]。离散碰撞检测算法是在对时间离散化采样后的时间点上检测物体之间的静态干涉情况，连续检测算法是根据轨迹参数方程对物体的运行轨迹采样，涉及模型结构的精确建模问题[6]。从空间的角度看，算法又可分为基于图形空间的检测算法和基于图像空间的碰撞检测算法。此外，根据不同的场景需求和解决方式，碰撞检测技术还有多种不同的分类方法：（1）根据算法执行流不同，分为串行化算法和并行化算法，（2）根据物体材料特性，分为针对刚性体的碰撞检测和针对变形体的碰撞检测，变形体间的碰撞检测又要考虑物体的自碰撞情况，（3）根据检测的模型个数，可分为两个模型系统的碰撞检测和多体系统的碰撞检测，（4）根据物体的形态不同，分为针对凸体的算法和针对凹体的算法等。高精确度和真实性的碰撞检测算法能确保微波管装配和设计过程的合理性和可靠性，能够大大降低微波管器件的生产成本并提高制管的成品率。然而，为了得到更高的仿真精度和对现实世界高度逼近的真实性，模型的计算机表示方法和数据结构将更复杂，组成模型的基本图元的数量将大大增加，这会造成碰撞检测计算复杂度和时间成本的成倍增长。因此，在确保微波管参数化建模的高度精确度和真实性的同时，又要保证碰撞检测的速度，是微波管仿真环境下的碰撞检测算法的一大挑战。

本论文基于以上背景，主要研究了碰撞检测算法和参数化建模技术，设计和实现了一套三维微波管参数化建模软件平台和碰撞检测求解器，现在已经应用于课题组的MTSS软件之中。通过对刚体模型之间的碰撞检测技术的研究与分析，设计和实现了一种基于混合包围体的碰撞检测算法，并将之应用到微波管仿真环境中，支持模型零部件之间的干涉检测和间隙分析，确保微波管设计和制作过程的合理性和准确性。

1.2 国内外研究现状

### 1.2.1 基于时间域的碰撞检测算法

从上世纪九十年代开始，研究人员在几何计算领域广泛和深入的研究了静态碰撞检测算法，其基本原理是只在场景中模型的位置和方向被固定的情况下，判断物体是否相交。静态检测算法对算法准确程度的要求高于对算法速度的要求，算法的复杂度高[7]。

随着对碰撞检测算法的性能要求的不断严格以及应用领域的拓宽，学者们陆续提出了处理动态场景下连续运动模型的算法。Larsen等[8]提出了一种离散碰撞检测算法（Discrete Collision Detection, DCD），通过计算扫掠球体之间的距离来判断模型之间的碰撞情况。Bridson等[9]提出了一种针对变形体碰撞问题的离散检测算法，该算法可以计算场景中物体碰撞的穿透深度和物体内部的自碰撞，并有较高的计算精度。

然而，离散检测算法存在两个主要问题。一方面，算法效率受时间采样的方式的影响较大。过于粗糙的采样方式会造成场景中物体之间发生穿透现象之后才检测到碰撞的问题[10]，这会严重降低虚拟场景中物体运动过程的真实感，毕竟谁都不想看到玩游戏时出现穿墙现象；但增加检测频率，又会增加检测计算的成本，使得时间开销变大，自适应采样或许是一个解决方案[11]。另一方面，假设有这样的场景，场景中有一个模型如子弹在高速运动过程中穿透了一个金属薄片如钢板。然后可能发生这样的情况：在连续的两个时间采样点上子弹分别位于钢板的两侧，算法没有检测出这两个连续时间点之间发生的碰撞，导致遗漏检测[12]。学者们因此提出了相关解决办法如自适应步长法等[13]能在一定程度上降低穿透和漏检的现象。但是这些处理方法都是以牺牲计算时间为代价来提高精度，而且对复杂模型的检测会占用过多CPU资源。

虚拟环境下可能存在具有大规模运动物体的场景，为了使检测算法满足此需求并解决离散碰撞检测的穿透和漏检问题。Redon等[14]提出一种连续碰撞检测算法（Continuous collision detection，CCD），对多边形物体的运行轨迹进行插值，在连续时间步长内检测物体与其他物体、物体与场景的碰撞以及物体的自碰撞的算法。此算法在四维时空和几何结构空间进行精确建模，优势在于减轻了穿透和漏检问题，但缺陷是算法的实现复杂，计算时间较长，在大规模场景中实时交互性能差。Tang M等人[15]提出了一种可变形物体之间连续碰撞检测的交互式算法。首先，构建一种基于K-DOP的两级动态层次结构，并进行高层剔除来排除模型自身的大部分非碰撞区域来减少自碰撞测试的三角形对；然后，设计一个孤立集（小部分特征对）来减少相邻三角形对之间的冗余检测。该算法适用于变形物体如布料、破碎物体和N-body仿真，但受到模型中网格连接度的限制。Wang等[16]基于伯恩斯坦符号分类法（BSC）改进了连续碰撞检测算法，在计算中进行详细的误差分析并利用贝塞尔曲线得出严格的误差范围。基于此，他们设计出一种能够显著降低碰撞检测误报率的精确算法。Peng Du等[17]为提高大规模的动态模型场景下连续碰撞检测的时间性能，提出了一种适用于分布式并行系统的DCCD算法。该算法在一个GPU集群中并行分配连续碰撞检测算法的负载，并采用动态分区方法来平衡系统中的负载。该算法实现了更均衡的负载和更高的多体刚性CCD算法剔除率。

**1.2.2** 基于空间域的碰撞检测算法

基于几何空间的碰撞检测算法主要分为空间分割法、扫掠和裁剪法以及层次包围体算法[18]。

（1）空间分割法。该算法是指将三维空间划分为多个子空间，为子空间中的所有几何物体分别创建对象链表。在进行干涉检查时，只用判断同一子空间或者邻接子空间的对象链表中的几何对象是否相交即可。如果相交的几何属于同一模型，则可以确定该模型存在自碰撞情况。鲍义东等[19]提出了基于自适应网格划分和编码优化的八叉树结构，应用于物体分布不均匀且密集的手术训练系统中。该算法在图元精确检测阶段采用自适应三角形细分方法，避免大量冗余的测试并提高检测效率。Francisco等[20]针对高分辨率的多边形模型的碰撞检测问题，提出了一种在单核CPU架构中使用EBP-Octree（Extended Bounding-Planes Octree）结构的碰撞检测算法，突出特点是在千万级物体运动并相互穿透的情况下实时性较好，并且能根据硬件性能调整内存消耗，适用于几乎任何环境下的三角形间的相交检测。

（2）扫掠和裁剪法（SAP）。该算法在处理大规模运动对象的场景方面是非常高效的，可以确保在线性时间内完成初步碰撞检测。近年来，结合动态数据结构与图形硬件、并行化技术等的扫掠与裁剪算法陆续被提出。王发麟等[21]应用SAP算法检测电缆内部的自碰撞情况。该算法首先构造了电缆的分段数学表达式。然后遍历所有电缆离散点，执行SAP计算电缆穿透深度。最后，根据穿透深度与电缆半径之和的比值来确定干涉。该算法的检测精度高，适用于电缆布线过程中的碰撞检测。Capannini等[22]提出了一种基于CPU并行和上下文感知的自适应SAP算法。此算法使用双轴扫描和简洁数据结构SuccTree降低范围查询复杂度。实验结果证明此方式能显著提高多核高密度分布的物体之间的碰撞检测性能。但缺陷是需要在图形设备和系统CPU之间频繁交换数据，会导致大量开销。Binbin Qi等[23]提出了一种基于动态SAP框架和事件驱动机制的连续碰撞检测策略。该策略自动生成事件预测对象之间的碰撞，如场景中对象的插入和移除，脆性刚体的压碎以及物体形状的动态变化等情景要发生阻塞的事件。该算法提出事件阻塞报警和阻塞缓解机制，可以自适应缓解多体系统中将要发生阻塞的事件，提高算法的性能和鲁棒性。

（3）层次包围体树（Bounding volume hierarchy，BVH）。该层次结构是最常使用的碰撞检测数据结构，而且效率较高[24]。在过去，由于硬件架构的限制，大多数算法依赖于运行在CPU上的串行算法来构建和更新BVH树，然后在GPU上进行实际的碰撞查询。近年来，研究人员开始将GPU渲染与BVH的构造相结合[25]。直接在GPU上构建BVH结构可以避免频繁在CPU和GPU内存空间之间传输数据造成的时间消耗，从而提高碰撞检测算法的效率，降低算法复杂度和时间消耗。Lauterbach [26]引入了一种线性BVH（LBVH），此结构利用空间莫顿码将BVH构建问题简化为节点排序问题以促进复杂对象的划分并显著提高构造速度。自提出以来，LBVH已经被多次扩展，并启发了许多构建算法。文献[27]扩展了此算法，提出一种表面启发式优化的HLBVH算法，此算法通过利用层次网格的粗粒度空间相关性减少了检测计算开销和内存使用。Karras[28]提出了一种深度优先的二元基数树技术，并在线性时间复杂度内构建了整颗树。Apetrei[29]提出了一种自底向上的构造策略，被认为是现今最快的构造方法之一，仅需要一个GPU内核构建层次结构并计算包围体边界。该方法相对高效但复杂，需要分析BVH树的内部节点的划分情况以在节点索引和莫顿码范围之间建立连接。

基于连续帧之间运动物体的时空相关性（即相邻时间点上物体的位置和状态变换较小），包围体遍历树（Bounding Volume Traverse Tree, BVTT）前沿被提出，定义为包围体遍历树（BVTT）的非重叠节点的遍历路径。跟踪BVTT前沿可以使碰撞检测剔除冗余的包围体测试。文献[30]首先提出了一种利用空间相干性来加速碰撞检测的增量方案，也称为广义前向跟踪，后来文献[31]将其应用于基于GPU的流算法中。之后，为解决场景中模型变形会降低剔除效率和查询速度以及BVTT任意布局导致的GPU缓存效率低下问题，Wang等人[32]提出了一种基于GPU的快速碰撞检测方案，该方案提出对BVH和BVTT前沿进行直方图排序和并行计算以及基于更新日志的BVH自适应调整功能。

此外，很多学者引入几何特性对传统算法进行改进，也取得了一定的成果。孙敬荣[33]提出了一种改进策略，根据三角形面积加权计算协方差的方式构造OBB包围体的局部坐标系，解决偏差问题；并在精确检测阶段引入新坐标系，将空间三角形投影到二维平面判断相交情况，以提高碰撞检测的效率。李玉虎[34]引入材料力学的形心来确定OBB包围体的方向和位置，并通过调整BVH树的节点包围体权重来提高性能。唐皓源等[35]提出一种基于混合OBB包围体的算法，根据模型拓扑结构结合质点转换进行粗检测和质点还原进行细检测，有效减少检测计算的时间。

### 1.2.3 存在的问题与挑战

碰撞检测算法已被不断应用于虚拟制造、CAD/CAM和触觉交互等各个领域，是虚拟仿真过程的可靠性、精确性和实际运行安全性的重要保证。但是碰撞检测研究过程中仍然存在一些问题，已有方法至少存在如下几个方面的挑战。

（1）复杂模型碰撞检测的实时性。微波管仿真环境要求对模型表示具有高度的精确度和几何保真度，处理的对象大都是复杂高分辨率的刚体模型，将这些刚体模型经过有限差分法和三角剖分等网格划分方法得到若干个三角面片或四面体图元。随着组成模型的图元数量的增加，碰撞检测算法的效率大大降低。因此，如何使算法满足大型模型场景下的交互要求并进行高效的碰撞检测成为一个难题。

（2）内存消耗问题。BVH是碰撞检测算法中最常用的数据结构，但作为一种典型的树状结构，在每一层的节点包围体处需要存储大量字节。特别是在场景中存在大量复杂模型的情况下，内存消耗会急剧增大。因此，减少数据的缓存并优化数据结构占据的内存是一个关键性的问题。

（3）微波管碰撞检测技术的发展滞后。国内在微波管的建模和装配仿真技术方面的研究起步相对比较晚、发展较慢、对国外依赖程度较高，近年来的相关研究主要是在国外的商业化三维建模和装配软件如UG NX、SolidWorks等的开放API基础上进行二次开发[36]。大量研究工作是将国外的先进的建模设计技术和国内理论知识进行结合与应用，缺乏独立自主的微波管CAD前处理（如建模和装配等）的工业软件。此外，国内基本没有微波管仿真环境下碰撞检测技术方面的研究成果。

1.3 本文的主要贡献和创新

本文的目标主要是研究与实现微波管仿真环境下的碰撞检测技术。在课题目标的指导下，开发了微波管领域的参数化建模软件平台和碰撞检测技术模块，从软件功能的实现方面研究了微波管的参数化建模和装配过程中的干涉检查技术。具体的贡献和创新点在于：

1. 研究了一种改进的AABB-OBB混合层次包围体碰撞检测算法，并采用相关设计模式封装算法求解的流程为不同的函数接口，可用于实现不同需求下的碰撞检测算法的应用。为提高检测效率，尽可能降低检测时间，本文的算法采用逐步求精的方式执行碰撞检测算法流程，首先采用简单的AABB包围体包围整个模型以加速排除距离较远的物体对，然后采用紧密性更好的OBB包围体层次树包围零部件以进行更精确的检测，并在基本图元检测阶段采用三角形之间的距离计算公式和Moller算法检测图元对之间的最接近距离和相交情况。

2. 经过系统的研究和理解HOOPS图形渲染套件，以及ACIS建模相关的API函数和数据结构，设计和开发了一款微波管参数化建模的软件平台，为本文碰撞检测算法的实现搭建了可视化环境。在软件平台中设计和实现了参数化建模系统，主要功能为基本几何模型的构建、复杂曲线/曲面模型构建、捕捉点功能和历史流管理等；应用碰撞检测算法实现了碰撞检测求解器模块，主要功能有静态碰撞检测、最接近距离计算和干涉检查等；在数据读写方面实现了xml特征文件和sat模型文件的序列化和解析的特征文件读写功能模块。在用户交互方面实现了图形用户界面模块，包括组合菜单栏、属性浏览器、特征导航器和各种建模与编辑操作的对话框，辅助相关工程师进行创建和编辑微波管模型、操作模型视图、实时渲染显示、查看几何属性、查看零部件创建过程等界面交互操作。

3. 实现了混合碰撞检测算法在微波管参数化建模软件平台上的应用，通过实例验证说明了碰撞检测在微波管仿真环境下的功能实现。在场景中导入包含多个零部件的行波管组件，在干涉检查和间隙分析后突出显示发生碰撞的部件对，并对四种干涉情况的仿真结果进行了分析和可视化评审。

1.4 本文的组织结构

微波管仿真环境下的碰撞检测技术是微波管设计、装配与仿真过程中的关键技术。本论文的组织结构如下：

第一章 绪论

主要介绍微波管仿真环境中碰撞检测算法研究的背景和意义以及算法研究的历史发展和研究现状，强调了碰撞检测对于微波管建模仿真的重要性，并指出了微波管仿真环境中碰撞检测技术存在的挑战与关键性的问题。

第二章 相关理论研究与分析

研究与分析了参数化建模软件和碰撞检测技术中用到的基本理论和开发工具。研究了HOOPS套件和ACIS引擎的功能特性以及集成两个引擎进行开发的框架。此外，重点研究了各种包围体算法的特点，对包围体算法做出了性能比较和分析。

第三章 混合碰撞检查算法的研究与设计

论述了对混合层次包围体碰撞检测算法的研究与设计方案。通过对AABB和OBB包围体的研究与分析，引入了一种结合这两种包围体的分层次的混合包围体树状结构。探讨了BVH树结构的构造、遍历和更新方法以及组成几何模型的图元对之间的距离计算和相交测试方法。最后，通过碰撞检测实验验证了算法的性能。

第四章 微波管参数化建模软件平台设计与实现

为实现微波管仿真环境下的可视化的碰撞检测技术，集成各类工具开发出了一款微波管参数化建模软件平台。从软件平台的需求分析和整体系统框架出发，详细论述了软件平台的设计和实现，包括图形用户界面、参数化建模系统、碰撞检测求解器和特征文件读写等功能模块。

第五章 基础干涉和间隙分析应用实例

简要概述了干涉检查和间隙分析的概念。从操作流程的演示、结果显示与分析等方面论述了应用碰撞检测算法实现的基础干涉和间隙分析功能。

第六章 总结和展望

对三年研究生阶段的工作做出一个总结，并展望了在微波管仿真环境下的碰撞检测技术未来的研究趋势和发展方向。

最后为论文的致谢、参考文献以及攻读硕士学位期间取得的研究成果。

第二章 相关理论研究与分析

2.1 HOOPS图形引擎的介绍

HOOPS是基于OpenGL等图形编程接口进行扩展开发的一种高度封装的应用程序框架，在CAD、CAM、CAE和PLM等数字化科技和娱乐领域等具有广泛的应用。HOOPS提供了高性能的图形编程功能，包括高质量的图形渲染、实时快速的用户交互、高级颜色插值、高级动画、图形数据流化处理以及图形光栅化和科学可视化等[37]。HOOPS图形引擎的组件库主要包括HOOPS/3dGS、HOOPS/MVO、HOOPS/GUI、HOOPS/GM和HOOPS/Stream。下图2-1说明了这些组件之间是如何相互连接的，以及用户生成的事件如何从GUI工具包传输到HOOPS中。GUI框架的事件队列收集用户生成的事件，HOOPS/GUI集成模块监控事件队列，并将事件发送到HOOPS/MVO对象、HOOPS/Net对象或直接发送到HOOPS/3dGS。HOOPS/MVO应用程序通过使用几何建模器或HOOPS自身的API来与存储在图形段中的数据进行交互。

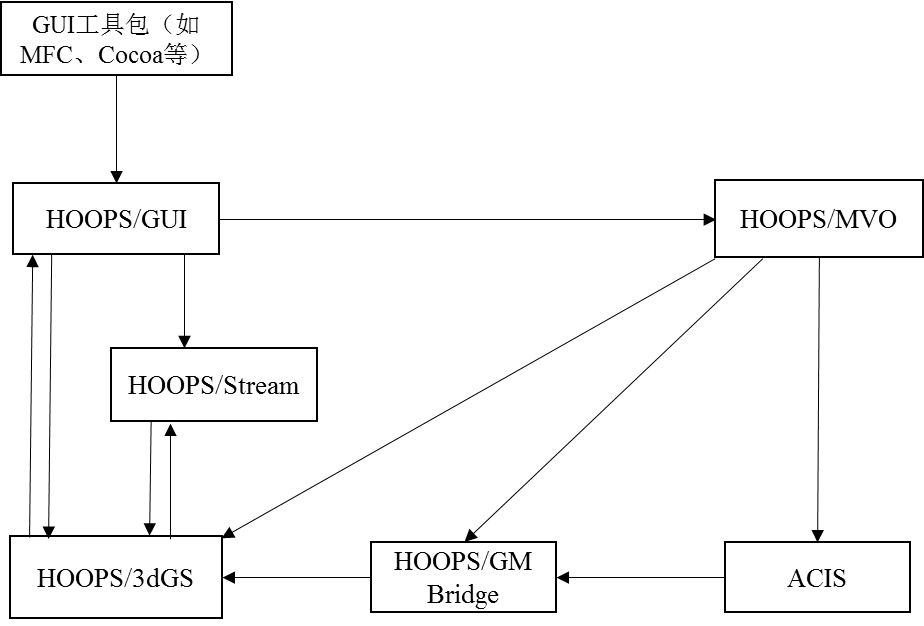


图2-1 HOOPS组件框架图

3dGS是HOOPS提供的一个高性能三维图形开发工具库，它通过高效的数据结构和算法实现了图形数据创建和管理、高级图形功能API和输出设备（包括OpenGL，X11，GDI，PostScript等图形系统），极大的提高了需要进行高效的图形图像后处理功能的CAD/CAM软件的开发。HOOPS/3dGS主要分为两个关键部分：①图形数据库，一种可以存储文本、图形、标记和几何元素等的内部数据库。②渲染管道，一种结构化和跨平台的高性能设备接口。

3dGS的图形数据库是一种树状结构，将段视为存储几何数据和图形绘制属性的容器。这种树状结构的段与段之间具有层次关系，像常见的二叉树和多叉树一样，可以是一对一或是一对多的“父子”节点关系，一个父段可以有一个或多个子段，但每个子段只能有唯一的父段[38]，其结构如下图2-2所示。此结构的属性是默认继承的，除非在子段中设置了局部的属性，否则子段具有与其父段相同的属性设置。在HOOPS/3dGS中，主要的属性有以下几种：渲染、显示、建模与查看、选中、驱动程序选项和启发式等属性。此外，3dGS中的所有几何都是无限薄的表面，也就是说HOOPS中没有实体。可以向图形数据库中插入网格、壳体、文本、图片、圆、NURBS曲面等几何，当几何插入到段树中之后，可以直接对几何信息进行相关操作。

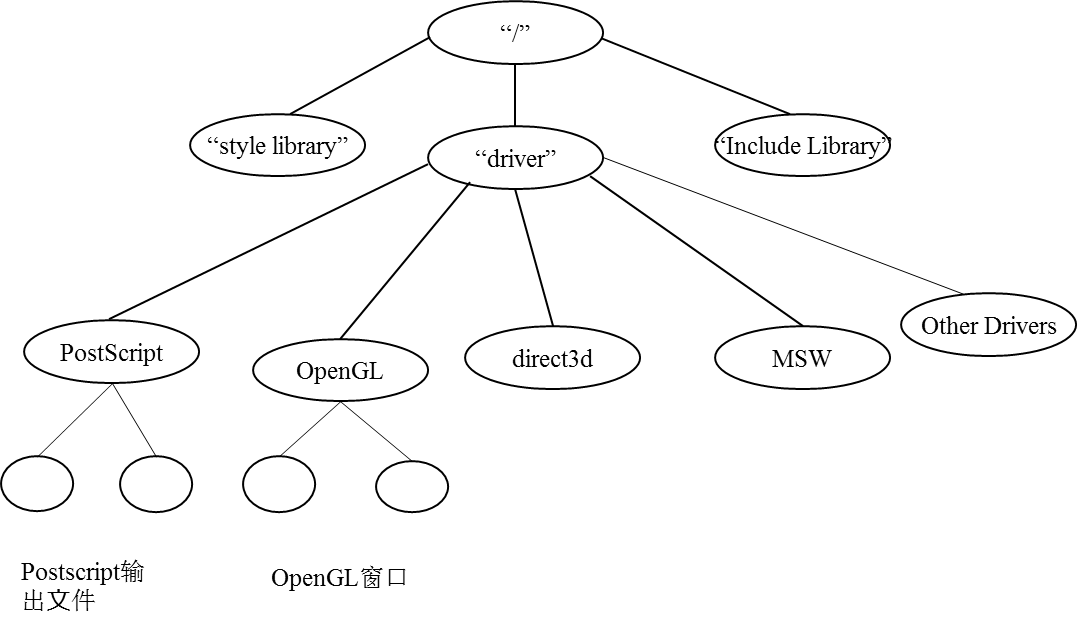


图2-2 HOOPS的图形段层次结构

HOOPS/3dGS进行高性能图形渲染得益于渲染管道（HDI），如图2-3为3dGS渲染的拖拉机模型图。渲染管道的主要功能是接收图形数据库中的图形数据，并将其逐层转换为输出设备可以读取的格式，然后将其发送到输出设备进行图形绘制[39]。如果输出设备的接口可以处理3D信息（例如DirectX），则HOOPS/3dGS可直接传输数据而不需做太多更改。但是，如果输出设备仅能处理2D数据（如Xlib或GDI），则必须经过数据的逐层解析。这样可确保在设备上最佳的渲染吞吐量，又可确保跨多个平台和设备的HOOPS/3dGS API的功能一致，渲染管道的工作过程如下图2-4所示。

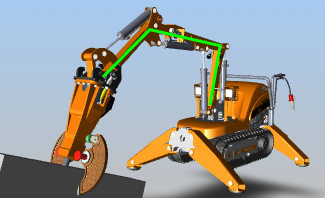


图2-3 HOOPS套件渲染的拖拉机模型

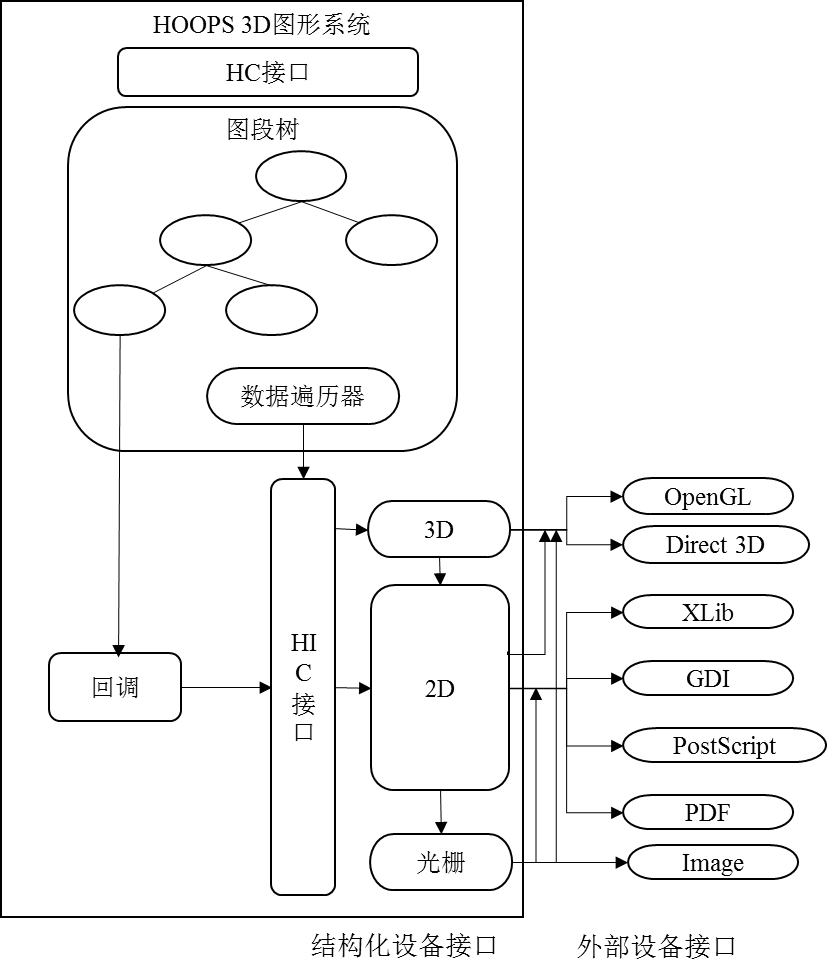


图2-4 HOOPS/3dGS的渲染管线组成结构图

HOOPS/MVO是一个面向对象的C++类库，它具有模型/视图/操作器的体系结构，是对3dGS模块的数据结构和API的封装。模型表示数据，而视图对应于此类数据信息的表示，操作器则是一组包括查询、创建、编辑和操纵图形对象数据的集合。在MVO的框架中，HBaseModel模型基类封装的是在“包含库”段节点之中的数据。HBaseView视图基类存储给定驱动段实例的各种信息和数据，管理几何段的渲染模式、颜色属性和相机位置等属性信息。HBaseOperator操作器基类及其派生类主要用于响应用户交互操作事件，完成用户在三维视图场景中的相机旋转、突出显示、选择对象以及移动对象等各种操作，如下图2-5所示为MVO结构中各个对象之间的关系。HOOPS/MVO的简洁和直观的API，能够满足CAD/CAM/CAE等大型应用程序高效开发的需求。本文对HOOPS的MVO框架进行了扩展开发，设计了一套基于特征的参数化建模系统，具体内容将在第四章详细阐述。

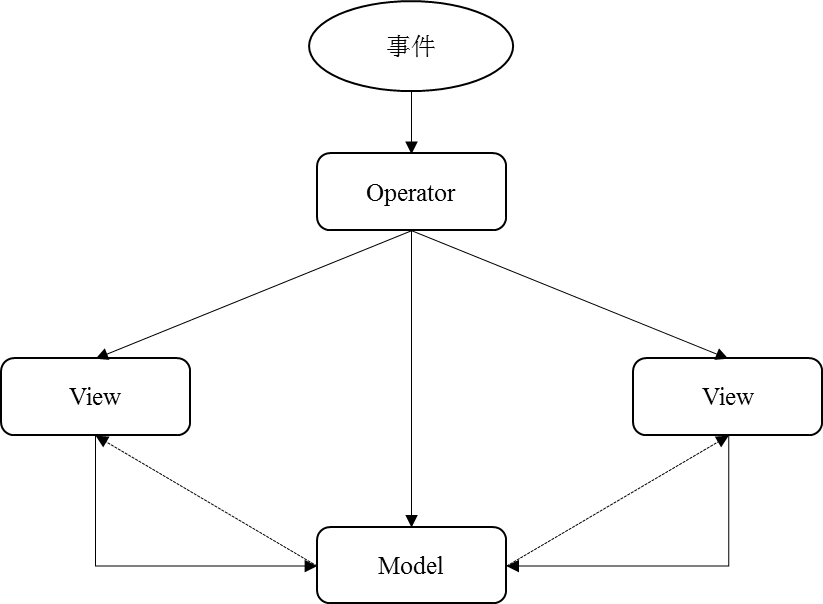


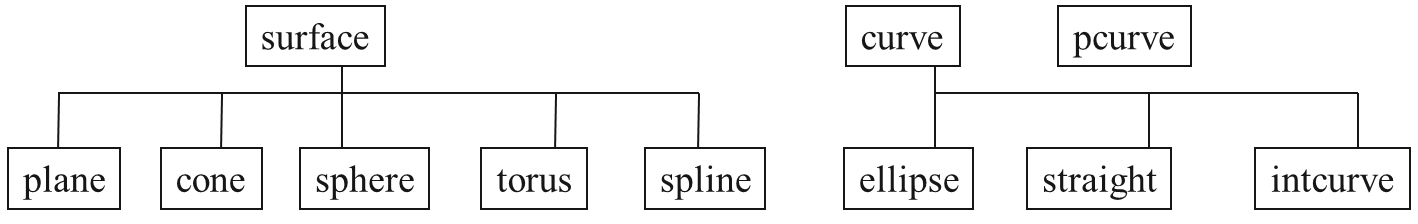
图2-5 MVO结构中的对象关系图

HOOPS/Stream是对HSF（Hoops Stream Files）文件进行高效的读写操作的工具包，HSF文件是一种高度压缩的二维、三维或自定义的图形数据文件。此外，HOOPS提供了持续更新迭代的帮助文档，较为全面的程序源代码文件和多个示例工程，对研究人员比较友好，能够大大缩短工业软件的开发周期。

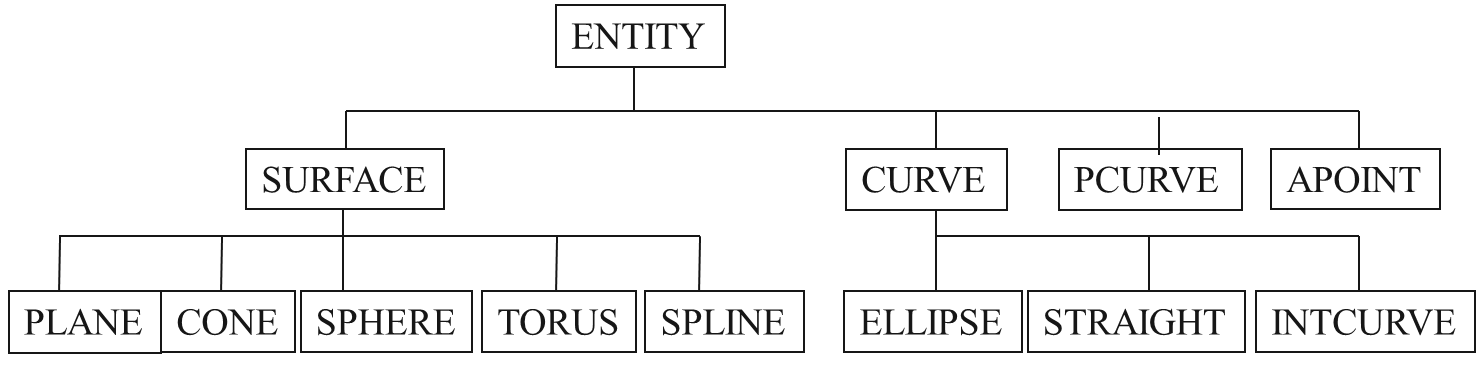
2.2 ACIS建模引擎的介绍

ACIS是一款精确的三维几何造型引擎，它采用高效和优化的数据结构并以组件形式封装起来。ACIS引擎在CAD/CAM领域具有广泛的应用，如MTSS、CATIA、CADKEY、HFSS等都是基于ACIS引擎进行建模和模型操作[40]。

ACIS造型引擎提供了集成线框、曲面、实体的几何造型功能，并设计了统一的数据结构描述这三种模型表示方式[41]。几何体（Geometry）是描述模型的基本几何元素如点、曲线和曲面等，几何元素不存在任何拓扑关系。ACIS几何实体的具体实现主要有两种方式：（1）构造几何体类，保存实体模型的几何定义信息，可以进行快速的求交操作，只能作为成员变量保存在相应的模型几何体类中。构造几何体类主要包括plane、cone、sphere、torus、spline和straight、intcurve等，它们之间的关系如下图2-6（a）所示。（2）模型几何体类，封装了几何信息、属性信息和对模型的操作如数据的读写、历史管理、变换操作等，并可以保存在sat模型数据文件中。模型几何体类主要包括PLANE、CONE、SPHERE、SPLINE和STRAIGHT、INTCURVE等。模型几何体类之间的关系图如下图2-6（b）所示。



（a）构造几何体类



（b）模型几何体类

图2-6 ACIS模型表示类关系图

拓扑体描述了模型中各个实体之间的空间关系和几何实体之间的连接关系[42]。ACIS内核在空间中建立了一个将实体区域和外部区域分离开的实体边界，从而可以判断空间中的一点是在实体的边界内、边界上或是边界外，其中，实体边界是由封闭的曲面组成。ACIS内核的模型表示法将组成实体模型的拓扑结构分成体、块、壳、子壳、面、环、线、有向边、边、顶点等元素，对应的类主要是BODY、LUMP、SHELL、SUBSHELL、FACE、LOOP、WIRE、COEDGE、EDGE和VERTEX类，这些类全部继承自ENTITY实体基类。模型拓扑类之间的关系如下图2-7中所示，下面给出一个说明，比如一个圆柱实体模型的拓扑组成为一个体、一个块、两个壳体、三个面、两个环、两条有向边是没有顶点。

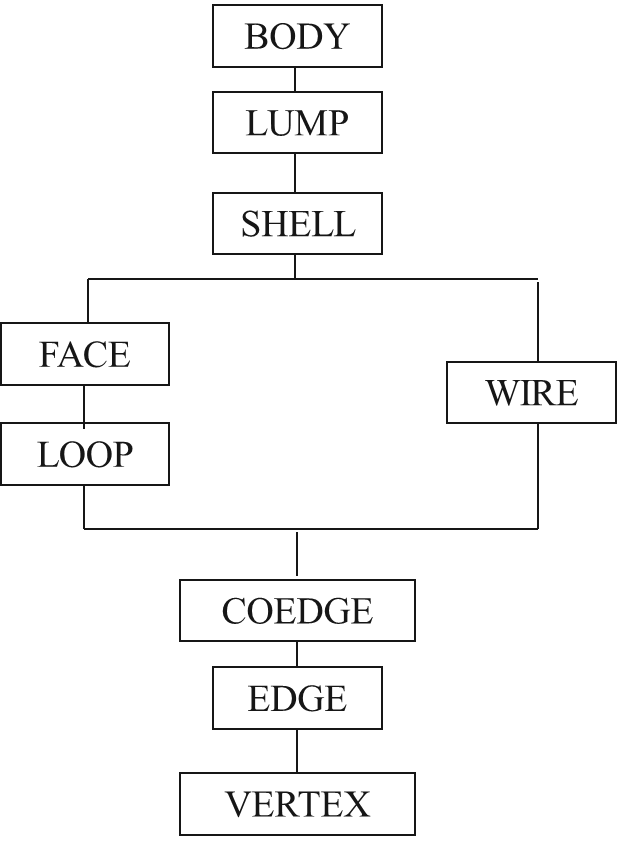


图2-7 拓扑类关联图

表2-1 ACIS的部分API函数表

|  |  |
| --- | --- |
| 尺寸驱动 | 位置驱动 |
| api\_make\_cuboid | api\_solid\_block |
| api\_make\_frustum | api\_solid\_cylinder\_cone |
| api\_make\_prism |  |
| api\_make\_pyramid |  |
| api\_make\_sphere | api\_solid\_sphere |
| api\_make\_torus | api\_solid\_torus |

API函数是ACIS内核和应用程序之间的接口，基本实现了对三维实体模型的所有操作，包括几何造型、属性更改和模型信息提取等。应用程序通过调用API函数来创建、修改或恢复模型数据，在应用程序调用API函数的过程中内核记录了模型操作的历史记录，提供在函数调用出错后自动恢复的强大功能。API函数将几何造型功能封装为统一的可维护接口，并隐藏了实现细节，支持开发人员进行集成化开发和快速建模。ACIS中的部分几何造型API函数如上表2-1所示。

2.3 集成HOOPS和ACIS

采用ACIS内核进行三维几何建模，并利用HOOPS套件高效的渲染和显示所创建的几何模型，可以搭建出高性能的CAD/CAM/CAE工业软件平台。一般来说，集成了ACIS造型引擎和HOOPS图形套件的工业软件的系统框架如图2-8所示，用户通过图形用户界面和应用程序进行交互，主要的用户交互操作为视图操作、相机操作、创建模型、特征操作以及历史的回溯和恢复等等。其中，ACIS内核提供了几何造型的API函数和历史记录的功能，并通过HOOPS\ACIS Bridge接口与HOOPS组件通信，HOOPS通过渲染管道的接口调用底层图形库（如OpenGL、DirectX等）渲染和显示插入到其图形段树上的ACIS几何实体[43]。此外，HOOPS支持对hsf、hmf等模型和数据文件的读写，ACIS支持对sat模型文件的读写功能。

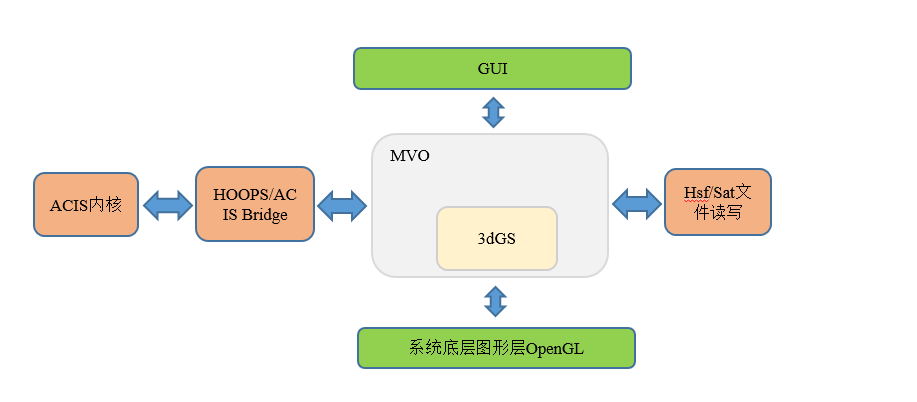


图2-8 HOOPS和ACIS间的交互框架图

2.4 碰撞检测算法理论

**2.4.1** 基本概念

碰撞检测的定义为：设三维空间为*R*，空间三维坐标系使用表示。*A*和*B*是三维空间*R*中的两个几何模型，用表示几何模型A所占的集合，表示几何模型*B*所占的集合。表示离散时间集。*A*和*B*在虚拟场景中运动变化形成四维空间坐标系，其中模型*A*和*B*沿时间*t*运动的轨迹构成的子集和，若成立，则两个物体相交，否则不相交。如若场景中模型个数为*n*个，判断第个模型和第个模型之间的相交情况的时间代价非常高，暴力法检测的时间复杂度为。

**2.4.2** 包围体技术

包围体方法是应用最为广泛的碰撞检测方法之一。基本思想是通过使用一个体积比模型物体略大、更易于计算的包围体来包围复杂模型并代替模型进行碰撞检测，从而简化模型间的检测计算。几种类型的包围体的构造方式、数据结构表示和相交测试方法等特性的描述如下：

（1）轴向包围体(Axis-Aligned Bounding Box, AABB)是一种包含模型对象且各边平行于世界坐标系方向的立方体，构造时只用将模型的基本图元在三个坐标轴上进行投影并取最大和最小值。AABB包围体的数据结构表示有多种，如下代码是给定AABB包围体的两点，总共24字节浮点数的内存占用：

struct AABB{

Point minPoint;//包含3个浮点数

Point maxPoint;//包含3个浮点数

};

其中，Point是一个表示顶点的结构体。AABB包围体之间的碰撞测试是判断二者的顶点的三个坐标值是否有交叉，最多比较6次。优点是构造较为简单，更新和重构快速，碰撞测试简单。但缺点是紧密性差，且在动态场景下的更新时间过长，实时性较差[44]。

（2）球包围体的定义是完全包围三维模型的半径最小的球体[45]。球包围体的构造首先要计算组成模型的所有基本图元的顶点的坐标均值，得到球心，然后计算模型的顶点到球心的最大距离值，得到半径。一个球包围体的数据结构代码描述如下。

struct Sphere{

Point Center;//包含3个浮点数

float radius;//1个浮点数

};

球包围体间的碰撞测试非常容易，假设有两个球心分别为(，)和(，)的包围球，如果球心距离小于它们的半径和，那么它们所包围的模型之间存在干涉，否则不干涉。见下式（2-1）所述。

 （2-1）

球包围体的优点在于当模型旋转时，不需要对它进行同样的旋转变换；当模型平移时，只用对球心执行同样的平移变换即可。但缺点是紧密性差，且对于变形体之间的测试并不适用。

（3）k-DOPs(K-Discrete orientation polytopes)包围体是一类使用对互相平行的平面包围模型的离散方向多面体[46]。K-DOPs中每两个平行的平面共享同一条法线，表示方式和AABB包围体类似，仅需存储模型顶点在条法线上投影的最大值和最小值，一种8-Dops包围体的数据结构表示如下。

struct k\_Dop8{

float min[4];//记录模型顶点在4个方向轴上投影的最小值

float max[4];//记录模型顶点在4个方向轴上投影的最大值

};

k-DOPs包围体之间的碰撞测试相对简易，只需要比较在二者在各法线上的投影范围是否有重叠，若全部重叠则判定相交。通过调整*k*的取值可平衡包围体的紧密性和简单性，从而提高检测效率。

（4）方向包围体（Oriented Bounding Box, OBB）[47]是指包围几何物体且在任意方向轴上的最小立方体。构造OBB包围体的关键是三个局部坐标轴向量的计算，假设组成某一个物体中所有几何图元（如三角面片）的集合为，其中组成模型的第个三角形顶点的向量为、、，组成物体的图元特征总数为*n*，则集合的均值和协方差矩阵的计算公式见下式（2-2）和式（2-3）。

 (2-2)

  (2-3)

其中， ,  和 ，都是一维向量，矩阵C是对称矩阵，由此矩阵得到的三个单位向量彼此正交，因此将向量正规化为单位向量之后可作为OBB包围体局部坐标系的三个方向轴[48]。然后将物体的顶点投影到这三个轴向上，并计算投影区域的最大和最小的值，即可确定包围物体的最小OBB包围体的长、宽、高尺寸。本文OBB包围体采用中心点、三个方向轴和三条边长的一半描述，内存占用为15个浮点数，数据结构的代码表示如下。

struct OBB{

Point Center;//OBB包围体的中心点（3个float数）

Point Vx, Vy, Vz;//三个方向轴（9个float数）

float x, y, z;//三条边长的一半值（3个float数）

}

运用分离轴定理能够有效的检测出两个OBB包围体之间是否有碰撞或干涉的情况。分离轴和分离轴定理的定义为：（a）分离轴：对于空间中的一条直线*L*，将多面体*E*和多面体*F*投影到*L*上，如果两个投影区域未重叠，那么*L*就是体*E*和体*F*之间的一根分离轴。（b）分离轴定理：当且仅当多面体*E*和多面体*F*之间存在分离轴的情况下，它们所包围的物体之间才不会碰撞或干涉。由定义不难推知，两个多面体的分离轴肯定符合以下的三个情况之一:

1. 分离轴平行于多面体*E*中某个面的法向量。
2. 分离轴平行于多面体*F*中某个面的法向量。
3. 分离轴平行于多面体*E*中某个面和多面体*F*中某个面之间的公共垂面的法向量。

通过分离轴定理可知，只要确定两个凸多面体之间是否存在分离轴，就可以确定两个物体之间是否存在干涉。两个OBB包围体之间的潜在分离轴有两个OBB局部坐标系上的六个坐标轴向量，以及轴向向量之间叉乘得到的九个向量[49]。在相交测试时，将两个OBB包围体的顶点分别向这些向量方向上投影，当且仅当在所有向量上的投影区间都重叠时，才能确定两个包围体发生了碰撞。若在其中一个向量方向上的投影不重叠，则确定两个物体不相交，该向量方向就是一条分离轴。

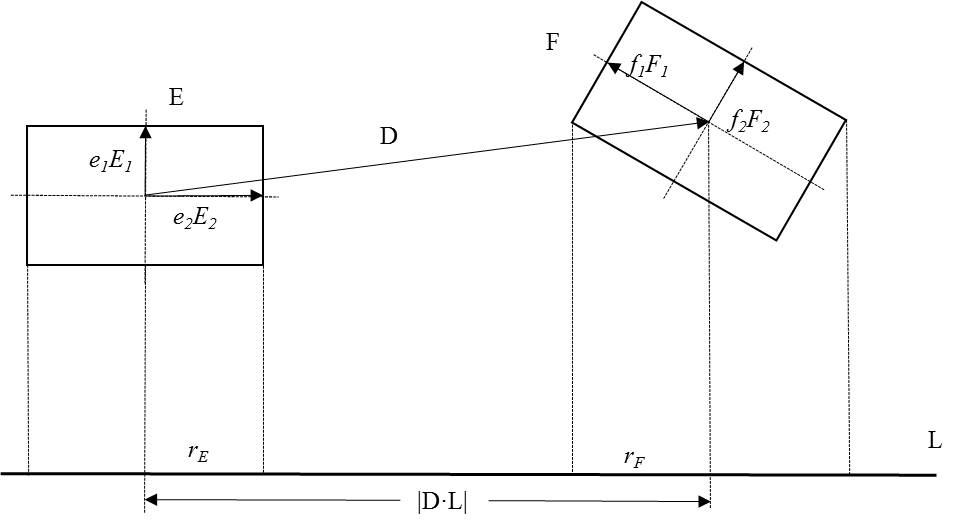


图2-9 OBB包围体间的相交测试

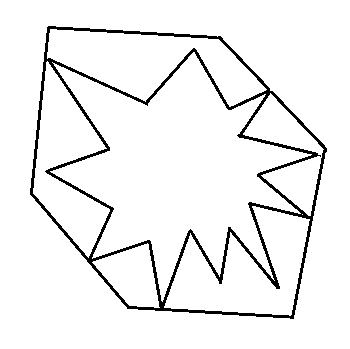
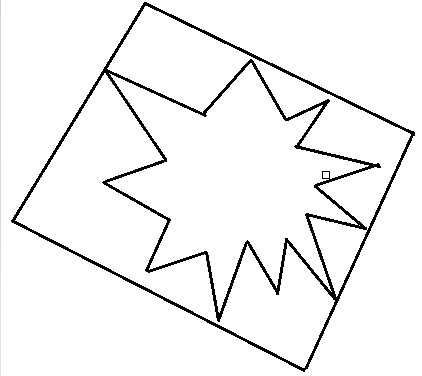
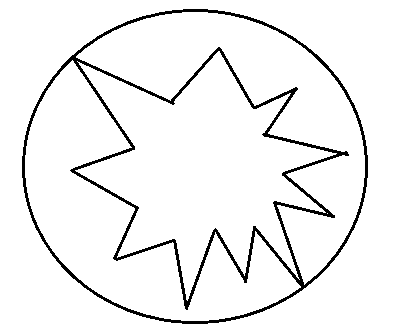
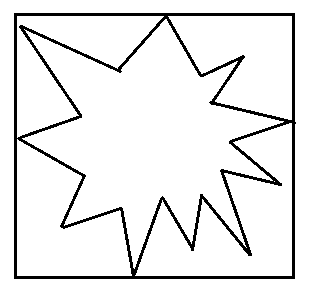
如上图2-9所示，其中多面体*E*和多面体*F*是空间中几何物体的OBB包围体。是包围体的质心到的质心的距离，是空间中的一条直线，是多面体*E*和*F*的质心在直线上的投影距离。*、*分别是体*E*与*F*在上的投影区间的半径。是*E*的边长值的一半，是*E*的轴线向量，*fi*是*F*的边长值的一半，*Fi*是*F*的轴线向量。若下式（2-4）成立，则两个包围体之间不会干涉或碰撞。

 (2-4)

此外，其他各种形式的包围体也陆续被提出如椭球体、圆柱体等，在不同的碰撞场景下都有所应用。下表2-2是一般情况下的几种典型的包围体结构的性能比较与分析的结果[50]，如图2-10所示为这几种包围体在二维平面上的构造形式。

表2-2 部分包围体算法性能的比较(1为最优)[50]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 包围体 | 构造简单性 | 内存占用 | 相交测试复杂性 | 紧密性 | 更新计算量 |
| AABB | 1 | 2 | 2 | 3 | 3 |
| Spheres | 2 | 1 | 1 | 4 | 1 |
| OBB | 4 | 3 | 4 | 2 | 2 |
| K-DOPs | 3 | 4 | 3 | 1 | 4 |



（a）AABB包围体 （b）包围球 （c）OBB包围体 （d）6-DOPs包围体

图2-10 几种包围体的构造形式

**2.4.3** 性能评价函数

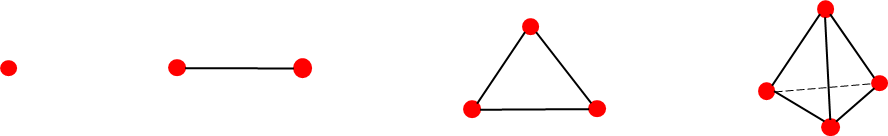
层次包围体树构建和遍历效率的关键点在于包围体的性能。通常由代价函数式（2-5）来评价基于层次包围体树的碰撞检查算法的性能。

** (2-5)

其中，各个参数含义为：*T*是算法的总时间消耗，是执行检查的包围体对的数量，是一对包围体测试的时间消耗，是图元测试的三角形对数，是一对基本图元对测试的时间消耗，是物体运动后BVH树结构中要更新的包围体节点数，是更新包围体节点的时间消耗[51]。从上式可看出，构造的包围体紧凑性越好，则要执行检查的包围体对的数量越少。此外，要实现快速的碰撞检测算法，则必须降低和的值，但这些变量总是相互矛盾。包围体的紧凑程度和简易性是研究人员追求的终极目标，但包围体越紧凑则它的构建就越困难，特征之间的干涉测试就越耗时。对包围体的优化必须对包围体的紧凑性和简单性进行平衡。

2.4.4 GJK算法的基本概念

定义1：单纯形的定义是在*x*维空间中，由包含*x*+1个点的点集组成的凸包，点集中的点都与仿射无关。*x*的取值范围为0-3，当*x*=0时，0-单纯形表示一个点；当*x*=1，则两个点组成的2-单纯形是一条线段；当*x*=2时，2-单纯形表示一个三角形；当*x*=3时，3-单纯形表示一个四面体。如下图2-11所示为四种单纯形。

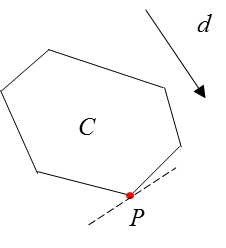
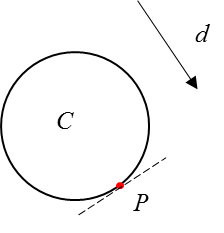


（a）0-单纯形 （b）1-单纯形 （c）2-单纯形 （d）3-单纯形

图2-11 四种类型的单纯形

定义2：凸包是指包含一个几何物体的最小的凸多面体。在实数空间中，对于给定的集合*T*，所有包含集合*T*的凸集的交集组成了凸包。凸包可以采用集合*T*中所有点的线性组合表示。

定义3：对于凸集*C*，将沿给定方向并且距离该凸集最远的一点定义为凸集*C*的支撑点。若对于空间中给定的一个方向*d*，满足，则将顶点*P*称作为凸集*C*的支撑点，即顶点*P*使得存在最大值，如下图2-12所示为方向*d*在多边形和圆上的两个支撑点。

（a）六边形在d方向的支撑点 （b）圆在d方向的支撑点

图2-12 凸体C在方向d上的支撑点

定义4：支撑映射，又称为支撑函数（support function）。支撑函数与一个凸集关联，把方向*d*映射到凸集*C*的支撑点。根据支撑函数可以得到凸集*C*上沿给定方向*d*的一个支撑点*P*，此点就是沿方向*d*且距离凸集最远的点。此外，因为支撑函数的计算是求取线性函数的最大值，所以凸体*A*和*B*的支撑函数可以表示为。

定义5：Minkowski和为欧几里得空间中的点集之和。假设*A*和*B*是空间中的两个凸集，*a*和*b*分别是由原点指向凸集*A*和凸集*B*中任一顶点的位置矢量，两个凸集的Minkowski和的定义见式（2-6）。

 （2-6）

其中的为位置矢量之和，Minkowski和的计算过程如下图2-13所示。

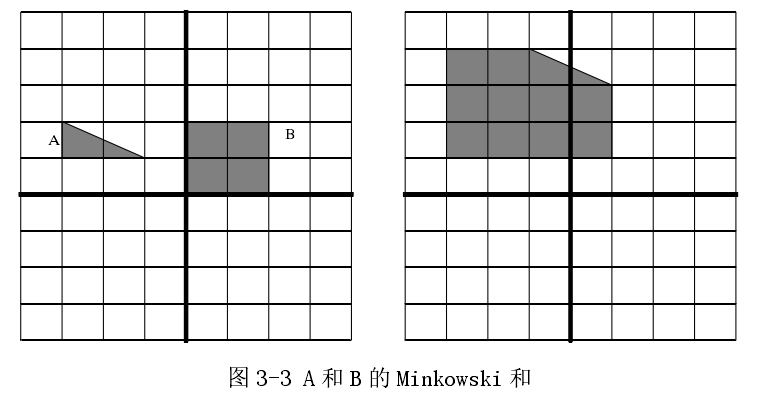


图2-13 三角形A和正方形B的Minkowski和

此外，凸集A、B的Minkowski差的定义见下式（2-7）。

 （2-7）

其中，两个凸集的Minkowski差也被称为位形空间障碍（Configuration Space Obstacle，CSO），Minkowski差的计算过程如下图2-14所示。

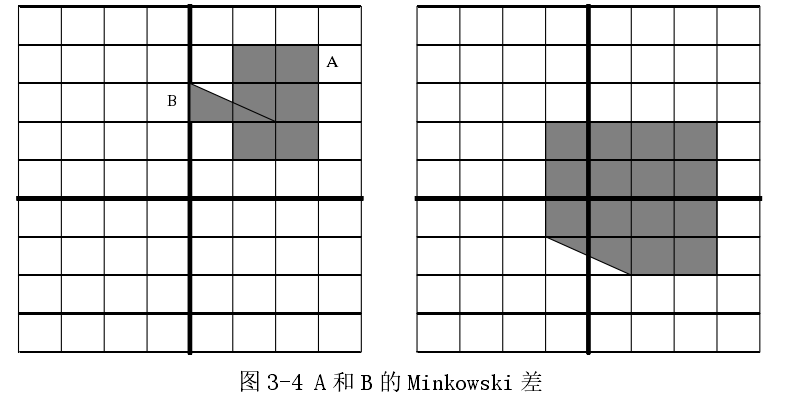


图2-14 三角形*A*和正方形*B*的Minkowski差

两个凸体集*A*、*B*的Minkowski和与Minkowski差均为凸体。在碰撞检测计算中，检测出凸多面体*A*和*B*相交的充要条件是原点位于凸体集*A*、*B*的Minkowski差所构造的凸多面体之内。两个凸体之间的最接近距离等价于CSO到原点*O*之间的最接近距离，因此，两个凸体的最近距离转换为点到一个凸体间的最近距离，凸体*A*、*B*之间距离的定义见下式（2-8）。

 （2-8）

2.5 本章小结

本章主要是对本文研究中涉及到的理论和算法技术的介绍。首先，介绍了HOOPS图形渲染引擎及其各个组件的特性，然后介绍了ACIS几何造型引擎的几何与拓扑表示方式及相关的类，研究和分析了ACIS的API接口功能和部分函数。之后，研究和分析了几种包围体的表示方式、构造方式以及相交测试方法，并对各类包围体进行性能的比较与分析。最后引入一种评估算法性能的性能评价公式。

第三章 混合碰撞检查算法的研究与设计

上一章中介绍了本论文所采用的一些理论的基本原理，对碰撞检测算法中最常用的包围体算法进行了详细的研究与比较。本章将详细阐述混合碰撞检查算法的研究与设计，主要分析和研究了一种混合层次包围体树的设计、构建、遍历和更新算法以及计算分离距离和穿透距离的GJK算法原理和流程等。

3.1 碰撞检测算法的设计

**3.1.1** 混合层次包围体树的结构

经过上一章的研究与分析，我们知道所选包围体的性能极大的影响了使用包围体代替模型进行碰撞检测的算法性能。有的包围体的精度更高，但会有计算复杂且不易于更新的问题；而有的包围体构造简单方便，且更新计算快，但是精度低。此外，使用一个包围体检测物体是否干涉是非常不精确的，也不能给出关于碰撞的更加详细的相关信息，因此一般将包围体和树状数据结构结合构成层次包围体结构[52]（Bounding Volume Hierarchy，BVH）来进行逐步求精的检测。这种做法基于现实的情况，将模型组件及其零部件的包围体组成一种分层次的树状结构，根结点是整个模型的包围体，中间的枝节点是逐步细分的零部件对象的集合，叶子结点是实体的基本特征（如顶点、三角面片等）。然而，要想得到一种精确度好、计算复杂度低、内存空间消耗少的层次包围体树结构，仅使用单一种类的包围体难以有所突破。经过综合的考虑和分析，本论文决定采取一种基于AABB包围体和OBB包围体的混合BVH结构，如下图3-1所示。混合BVH结构结合了AABB包围体与OBB包围体的优势，顶层采用构造方便、测试计算更为简单的AABB包围体包含复杂模型，下层采用OBB包围体更加紧凑的包围实体。本文算法的主要优势为：

（1）AABB的内存消耗小，构造计算简单，而且相交测试复杂度要远远低于一对OBB间的相交测试复杂度，能够降低使用单一OBB包围体的BVH树的算法的内存消耗和测试计算复杂度，并快速的排除距离较远的物体对。

（2）在BVH树的下层节点中采用OBB包围体可以更加紧凑的包围实体，提高检测精度，获得更详细的图元信息。OBB包围体根据组成物体的几何元素特征构建的局部坐标系可以使它更紧凑地包围几何物体，能够大大减少需要测试的包围体数目和图元特征的数量，从而降低时间代价，可以满足微波管仿真环境下碰撞检测的实时性要求。

（3）当物体发生平移和旋转变换等几何变换操作时，只用对AABB包围体的世界坐标轴或OBB包围体的局部坐标轴执行同样的变换矩阵就可以重新构造AABB或OBB包围体。

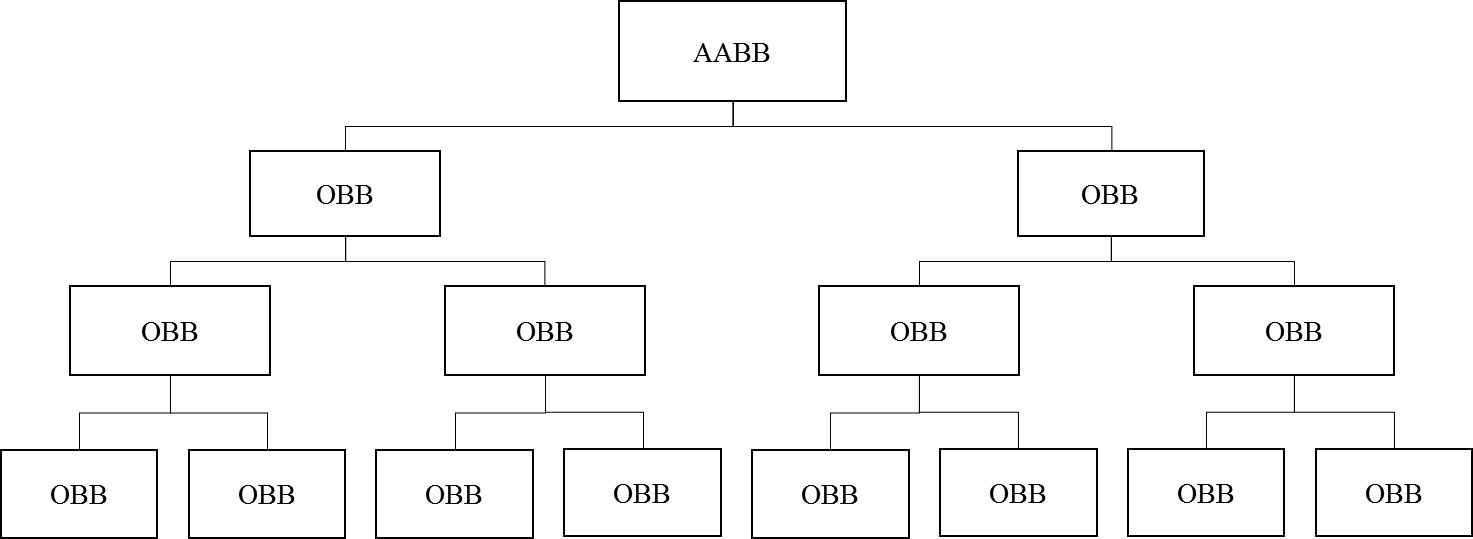


图3-1 AABB-OBB混合层次包围体树结构

这种混合层次组织结构的目标是（1）在构造包围体的简单性和紧凑型之间平衡，使层次包围体树紧凑的逼近几何模型，（2）快速的初步检测，排除距离较远的物体，并将碰撞检测的时间复杂度由尽可能降低到，（3）实现运动模型的BVH树的快速更新或重构。

**3.1.2** 算法流程

本文的基于混合BVH树结构和GJK算法的碰撞检测算法的流程如下图3-2所示。算法流程的执行步骤如下所述：

（1）启动微波管参数化建模软件环境，初始化配置碰撞检测环境。

（2）使用参数化建模模块创建模型零部件或直接导入微波管的组件模型，对需要执行碰撞检测的刚体模型建立AABB包围体。

（3）通过初步的碰撞测试阶段快速排除相距较远而不会产生碰撞的零部件模型，如果检测到两个AABB包围体不相交，则返回并等待下次检测，若检测到相交，则继续下一步。

（4）如果步骤（3）中的两个AABB包围体相交，此时若启用“计算穿透深度”选项，则启用GJK算法计算物体对的最近距离，并继续执行第（5）步；否则，为相交部位的潜在干涉的零部件对构造能够更紧凑的包围物体的OBB包围体层次树，并继续执行（6）和（7）步。

（5）获取潜在碰撞的特征实体模型的顶点集合作为GJK算法的输入，然后采用GJK算法计算物体之间的最近距离，若检测到物体对未发生碰撞，则算法结果为分离距离值，若碰撞，则输出穿透距离值。

（6）采用层次树的遍历算法检索两颗OBB包围体层次树直到叶子节点，完成针对树中不同层次节点的检测，以实现逐步求精的检测，若在此过程中没有检测到干涉，则返回并等待下次的检测，否则，进行下一步。

（7）若遍历检测到叶子节点之间的碰撞情况，则进入图元特征的相交测试阶段，若未检测到相交，则没有碰撞发生；若检测到相交，则零部件之间一定发生了碰撞，返回碰撞结果信息以进行分析和处理。

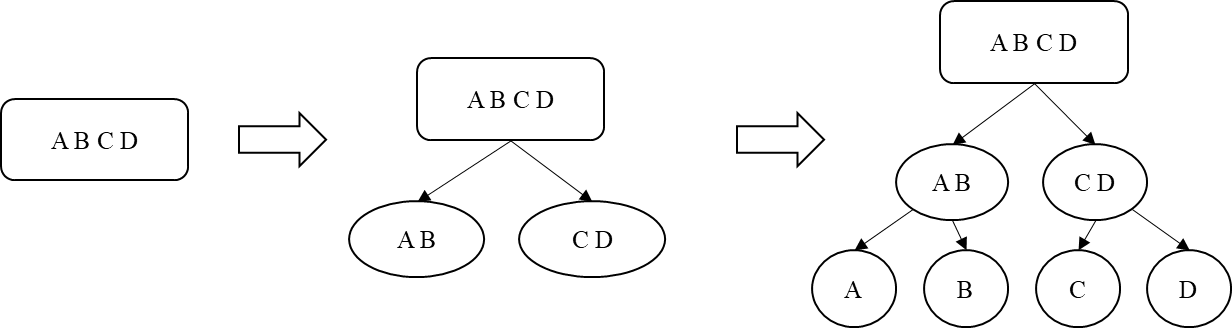
C:\Users\Andrew\Desktop\碰撞检测算法流程.tif

图3-2 混合碰撞检测算法流程

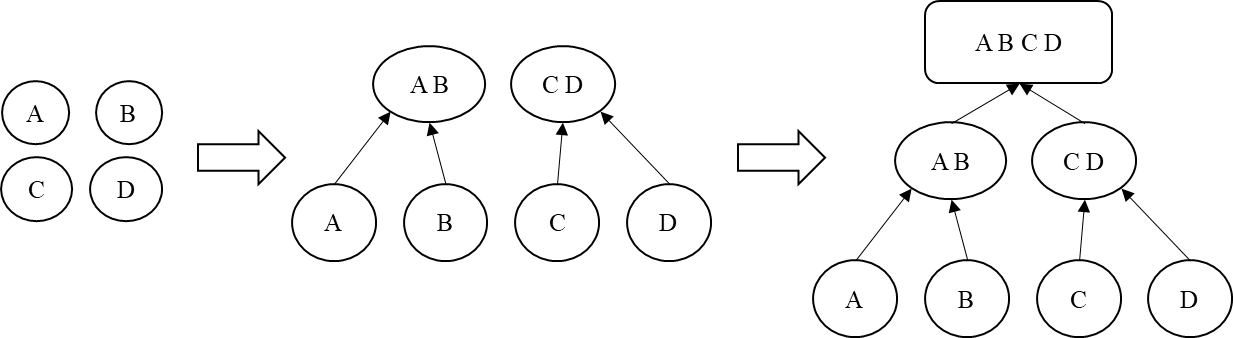
3.2 层次包围体树的构造

自顶向下的方式构建层次包围体树的过程如下图3-3（a）所示，首先以整个几何模型的所有对象作为整体建立根节点的包围体；然后，按照一定的规则（如计算分割超平面）将几何模型的基本图元集合递归划分为两个子集，并建立子集的包围体作为根节点的子节点；之后按上述方法递归地进行逐步分割和构造包围体，直到树的叶子节点只包含基本图元或达到结束条件，完成层次包围体树的构建。自顶向下的构造方式的优点在于实现简单、构造快速且应用最为广泛，技术成熟度高，因此本文决定采用自顶向下的层次包围体树构造方式。

与自顶向下的构造方式相反，自底向上的构造方法如下图3-3（b），首先计算和构建组成几何模型的各个基本图元的包围体，并令其作为层次包围体树的叶子结点，采用一种组合优化算法组合叶子节点并生成父节点，然后递归的组合子节点并生成父节点，直到生成最后一个包围根节点的包围体时终止，根节点包围体是包含整个模型的包围体[53]。其中，包围体合并的时间复杂度是整个算法性能的关键点，根据所选择的包围体和合并算法的不同而有所差异。



（a）自顶向下构造法



（b）自底向上构造法

图3-3 层次包围体树的构造方法示意图

3.3 层次包围体树的遍历和更新

**3.3.1** 层次包围体树的遍历

在BVH树中，每个节点上的包围体都是组成物体的部分组件集合的一个子集，根节点的包围体是包围整个模型的包围体。BVH树的深度优先遍历算法是从树的根节点开始逐个访问子节点，当遍历到叶子节点后回溯，此算法逻辑清晰、代码量较少、实现相对容易。本节介绍的遍历算法核心思想是深度优先遍历两棵树的所有节点，确定相交的包围体对。算法首先是判断两棵树的根节点的包围体对是否相交，如果不相交则两个模型肯定不会碰撞，就不用再判断子节点；否则，深度优先遍历两棵树，这时有三种情况需要考虑：

1. 两节点均是非叶子节点。检测两节点的包围体是否存在干涉，如果干涉则递归检测两个节点的子节点的包围体。
2. 两节点中有一个是分支节点（非叶子）。检测节点的包围体之间是否存在干涉，如果干涉，则递归检测非叶子节点的子节点和叶子节点的包围体。
3. 两节点均是叶子节点（即基本图元的包围体）。调用图元特征对之间的相交测试算法进行精确的求交，判断两个节点包围体所包含的基本图元之间是否发生干涉或相交情况。

以上三种情况下，只要检测出两个包围体之间不相交，则当前节点及其子节点包围体的检测终止并返回。通过这种自顶向下的逐步求精的遍历算法，可以快速检测的发生碰撞的物体对，算法流程的描述如下图3-4所示。

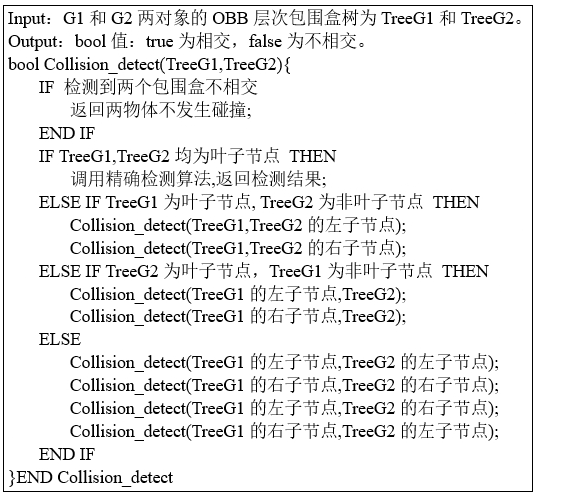


图3-4 混合BVH树的遍历算法

**3.3.2** 层次包围体树的更新

虚拟环境中，物体的运动将使得物体的位置和状态都会发生变化，初始位置和初始状态下的层次包围体树将不能反映运动物体的实时位置和方向。一旦物体发生运动和变化时都对层次包围体树进行更新，这样操作的时间开销会严重影响碰撞检测的速度和实时性能。一般来说，实时高效的碰撞检测算法的关键在于更新过程的速度和时间消耗[54]。

 （3-1）

本文讨论的物体主要是刚体对象，刚体对象的几何变换一般而言可以分解为旋转和平移两种变换类型。因此，假定世界坐标系下沿三个坐标轴方向的平移向量为，以及绕三个坐标轴方向旋转的比例参数为、、，可以得到对应的几何变换矩阵见上式（3-1）所示。假设几何变换操作的顺序依次为绕*x*轴旋转、绕*y*轴旋转、绕*z*轴旋转，则总旋转矩阵见下式（3-2）所示[55]。

 （3-2）

根据以上分析可知，通过对初始包围体的位置坐标应用矩阵变换可以实现BVH树的更新，其中*x*为几何变换前的包围体位置，*R*为旋转矩阵，*T*为平移矩阵。将模型平移之后，对包围体进行同样的平移变换即可得到移动后的新包围体。但是旋转模型之后，对包围体的更新不能直接应用旋转矩阵，而是要根据包围体的类型选择不同的更新策略[56]。比如，包围球的旋转更新操作就无需应用旋转矩阵，而OBB包围体的旋转更新操作就必须为三个局部坐标轴应用旋转变换操作。

3.4 三角形图元之间的相交测试

在BVH树的遍历过程中，若深度优先遍历检测的输入参数为两个叶子节点的包围体时，接下来就需要进行叶子节点包围体所包围的基本图元间的相交测试，组成微波管零部件的基本图元特征通常是简单的四面体或者是三角形面片。本文所讨论的情况是由三角形面片构成的刚体模型，基于图3-7所示的两种三角形之间的相交情况，主要探究三角形间相交测试的三种标量型判别法。

本文采用Moller提出的区间相交算法[60]，实现步骤是：首先建立三角形DEF所处平面的点法线式方程，并计算三角形ABC的顶点到该平面的距离，若三个距离值均不等于0且有同样的正负符号，则可确定不相交并结束计算；否则，计算两个三角形所在平面的相交直线方程，且，其中和分别为两个三角形所在面的法线，并计算三角形ABC和DEF的顶点在相交直线*L*上的投影区间是否存在交叉或重叠，若不重叠则可确定二者不相交；如果距离都为0，即两个三角形共面，则问题转换为二维相交测试问题；如果有一个顶点到DEF所在平面的距离为0，则只需判断该顶点是否在三角形DEF内。

Held[61]算法对Moller算法进行了简化和改进，主要是省略Moller算法中投影计算过程，直接检测穿过三角形ABC的相交直线*L*是否同样穿过三角形DEF，若直线*L*同时穿过三角形ABC和DEF则显然二者是相交的。Tropp[62]算法主要使用数学方法对相交测试算法过程进行简化，具体实现过程中通过降低冗余的图元测试计算量来提高三角形相交测试的效率。三种算法的计算量比较如下表3-1所示。

表3-1 三种相交测试算法的比较[63]

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 相交测试算法 | 加/减 | 乘法 | 除法 | 比较 | 绝对值 | = |
| Moller[60] | 57 | 63/65 | - | 12/28 | 3/9 | 69 |
| Held[61] | 74/94 | 35/45 | 1 | 33/50 | 3 | 51 |
| Tropp[62] | 42/45 | 51/57 | - | 10/19 | 0 | 46/52 |

## **3.5** GJK算法计算分离距离和穿透距离

**3.5.1** GJK算法计算分离距离

GJK算法是针对凸体模型进行距离计算的碰撞检测算法，输入任意两个凸体的顶点集，就能计算并返回凸体间的最近距离值。基本原理是将凸多面体*A*和*B*之间的最接近距离计算转换为求解*A*和*B*的Minkowski差与原点之间的最接近距离。GJK使用Minkowski差中的支撑点来渐进的接近原点，它将支撑点存储在集合*Q*中，如果集合*Q*包围了原点，则表示两个凸体发生了碰撞。此外，对于未发生碰撞的凸体对，GJK算法可以返回两个物体上距离最近的两点。

空间中两个未相交的凸多面体*A*和*B*之间的最接近距离又被称为的分离距离，它是连接两个物体之间的最短线段的长度，可以用下式（3-3）来表示。

 （3-3）

假设两个凸多面体*A*与*B*的Minkowski差为凸体集*S*，则*A*与*B*之间的最接近距离公式可以等价于下式（3-4）所示

 （3-4）

其中，是凸集S内或S边缘上的点。

综上，两个凸体的最近距离转换为点到一个凸体间的最近距离，如下图3-5所示。本文采用GJK算法计算物体对之间最接近距离和穿透深度，若算法得到的物体对之间的距离值小于或等于0，则两个物体发生了碰撞，距离为负值说明物体之间存在穿透。

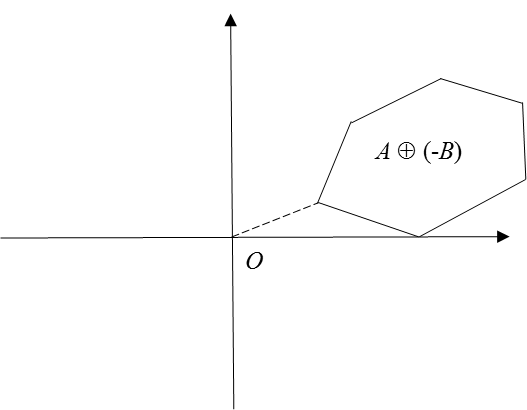
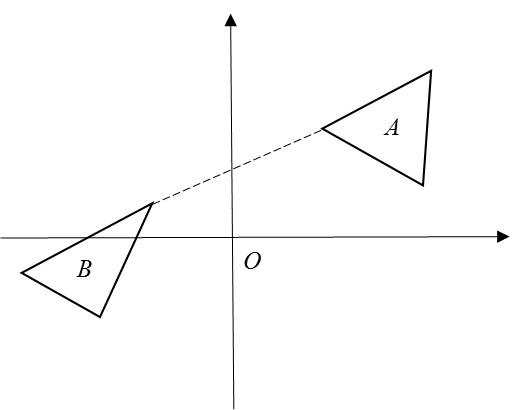


图3-5 GJK算法的距离计算原理

采用GJK算法计算两个凸体*A*和*B*之间的最接近距离，可以通过凸体*A*和*B*的CSO的顶点集降序迭代建立最多由四个顶点构成的单纯形集合，如果当前的单纯形包围原点，则停止迭代并返回碰撞。否则更新单纯形，并保证新的单纯形比当前单纯形到原点的距离更近。GJK算法的迭代流程如下：

（1）使用CSO中的一个或者多个顶点（顶点数最多4个）构造集合*Q*

（2）计算集合*Q*中最小范数顶点*P*。

（3）如果*P*为坐标原点，则原点在CSO占据的空间范围内，表示两个凸体*A*和*B*碰撞。

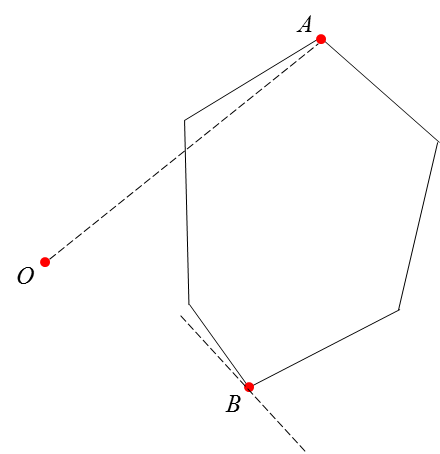
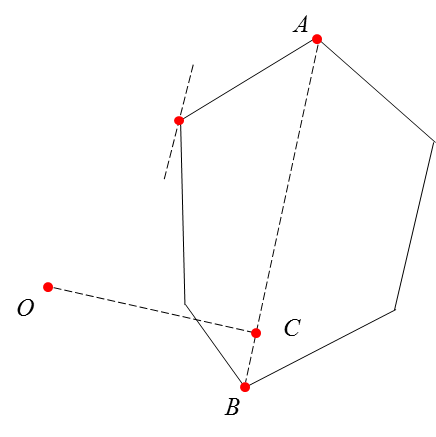
（4）删除*Q*中不包括点*P*的子单纯形中的顶点，得到子集。

（5）计算凸体CSO在-*P*方向上的支撑点。

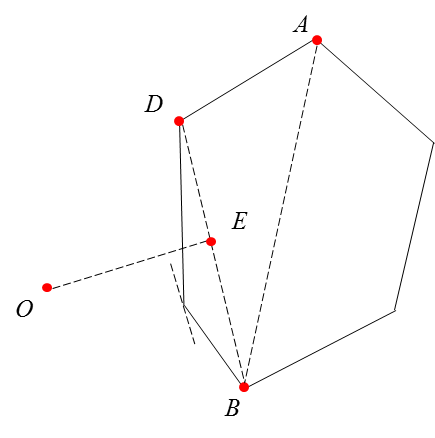
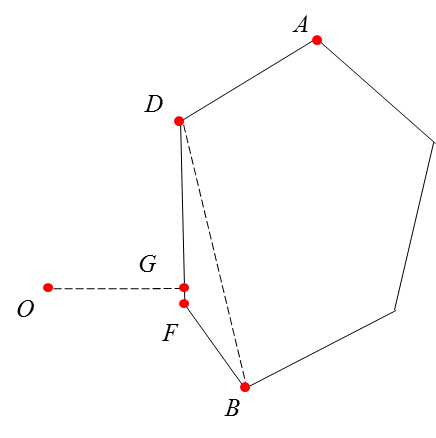
（6）如果点*v*在-*P*方向上不是极值点，则表示两个凸体不相交，且它们之间的距离等于从原点到顶点*P*的距离值。

（7）否则，将点*v*加入至单纯形集合*Q*中，并返回执行步骤（2）。

GJK算法适用于任意凸体类型，通过迭代收敛于一个距离值，必须采取相应的阈值作为算法的结束条件，以使计算非多面体对象时，算法可以正常结束，避免出现死循环。

（a）单纯形集合*Q*={*A*} （b）单纯形集合*Q*={*A*，*B*}

（c）单纯形集合*Q*={ *B*，*D*} （d）单纯形集合*Q*={ *D*，*F*}

图3-6 GJK算法计算多边形到原点的最近点的迭代过程

如图3-6所示为GJK算法在二维空间中计算多边形上距原点*O*最近点的计算过程。该算法以顶点*A*作为初始单纯形组成集合*Q*，算法开始时的*Q* ={*A*}。之后，沿-*A*方向搜索多边形内的支撑点并得到点*B*，将*B*添加到单纯形集合*Q*中得到Q={*A*，*B*}。计算此时的凸包*Q*中最接近原点的点为*C*，由于需要*A*和*B*作为凸组合表示*C*，因此两者都保留在*Q*中。再次计算-*C*方向的支撑点为点*D*，将之添加到集合*Q*中，此时的凸包*Q*中最接近原点的点是边BD上的点*E*，因为只需要*B*和*D*即可作为表示*E*为的顶点组合，所以集合*Q*更新为{*B*，*D*}。计算方向-E上的支撑点为点*F*，该点添加到*Q*中，此时凸包*Q*中最接近原点的点是边DF上的点*G*，*D*和*F*是表示*G*的凸组合的最小顶点集，故*Q*更新为{*D*，*F*}。此时，由于在-*G*方向，多边形中没有比*G*点更靠近原点的点，因此点*G*是多边形中最接近原点的点，算法终止，返回没有发生碰撞。

**3.5.2** GJK算法计算穿透深度

碰撞物体对之间的穿透深度是使得两个物体从相交状态转变为接触状态所需要平移的最小距离，假设点集*A*和*B*是空间中的两个凸集，穿刺距离可以用表示。如下图3-7所示为二维平面上的长方形和三角形之间的穿刺深度说明图。

C:\Users\Andrew\Desktop\图片3-7.tif

图3-7 长方形和三角形之间的穿透深度

两个凸体的穿刺深度本质上是凸体的CSO的边上离原点最近的点到原点的距离值。算法步骤为：（1）首先，基于CSO凸体创建一个单纯形如一个2-单纯形（三角形）或3-单纯形（四面体），确保单纯形包含原点且顶点位于CSO的边界上。（2）然后，迭代查找离原点最近的线段或者面，使用支撑点不断细分单纯形为新的单纯形以进行新一轮的查找。

在空间中，先在两个凸体的CSO上构造一个2-单纯形（三角形），确保其包含原点且顶点位于CSO上。对于三角形的三条边的仿射集，求解离原点*O*最近的顶点*v*。然后，计算向量*Ov*方向上凸体CSO的支撑点*w*，并使用该点细分三角形的对应边。此外，计算得到的*v*点应落在边上，而不能在边的延长线上。在每次迭代过程中，*v*点对应的边被不断细分为两条新边并加入单纯形集合中，重复执行计算单纯形中距离原点最近的点*v，*直到满足迭代终止条件。如下图4-4所示为在二维空间中求解穿刺深度的迭代过程。

C:\Users\Andrew\Desktop\图片3-8（a）.tif C:\Users\Andrew\Desktop\图片3-8（b）.tif

（a）第一次迭代计算 （b）第二次迭代计算

C:\Users\Andrew\Desktop\图片3-8（c）.tif C:\Users\Andrew\Desktop\图片3-8（d）.tif

（c）第三次迭代计算 （d）第四次迭代计算

图3-8 GJK算法计算穿透深度的迭代过程（箭头为最近向量v，空心点为支撑点，过支撑点的虚线为支撑平面）

在空间中，计算穿透深度的算法和在空间中类似，不同点是在二维空间中的单纯形是三角形，被细分的是三角形的边，而在三维空间中的单纯形是四面体，被细分的是四面体的（三角形）面。如下图3-9所示为GJK算法对三角形面的划分方法，图中的向量*OV*是三角形的法线，是O*V*方向在CSO上的支撑点，用*w*点将三角形划分为三个新的三角形。*y0*

C:\Users\Andrew\Desktop\图片3-9.tif

图3-9 三维空间中GJK算法对三角形面的细分

3.6 碰撞检测实验结果

在本节中，为了验证算法的性能，对本章所设计的碰撞检测算法进行测试。第一个实验是在不同规模的场景下，测试混合碰撞检测算法的性能，各场景中模型数量分别为2、10、30、60、90、120、150个，模型为自建的立方体、球、圆柱体等。第二个实验是在同一规模的不同相交程度场景下测试，不同相交程度是指场景中发生碰撞的物体或三角形的对数。测试的硬件条件为Intel(R) Xeon(R) CPU E5-2697 @ 2.30GHz，内存为8GB，win10系统。测试过程中的场景图如下图3-10所示。

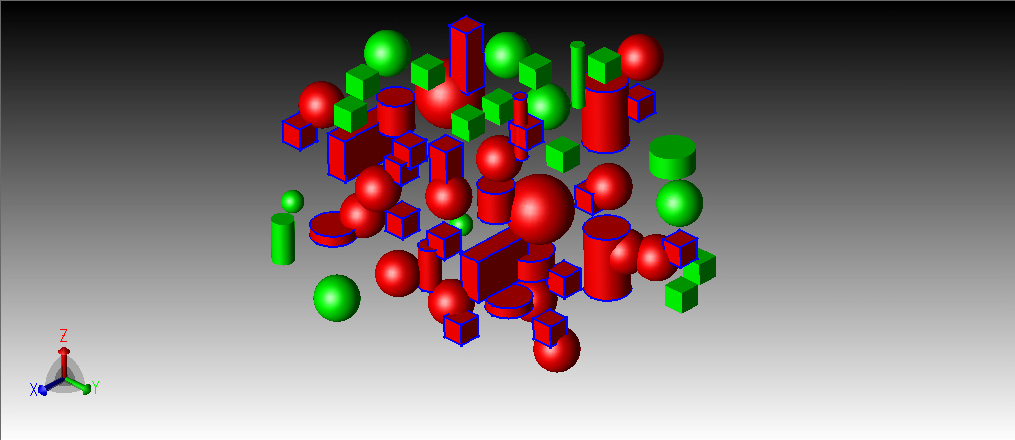


图3-10 碰撞检测测试场景图

（1）不同规模的场景下碰撞检测性能测试

在不同规模的场景下，碰撞检测算法的性能测试实验数据如下表3-2所示，检测时间单位为毫秒。不难看出，碰撞检测算法的实时性能受到场景中物体的个数和碰撞物体对数的影响较大。在实验测试中，随着场景中的物体数和发生碰撞的物体对数的增加，检测时间消耗不断增长，可以推测场景中物体的数量和相交程度对碰撞检测算法的性能有很大的影响。

表3-2 不同规模场景下碰撞检测数据

|  |  |  |  |
| --- | --- | --- | --- |
| 测试场景 | 总物体个数 | 碰撞物体对数 | 检测时间/ms |
| 场景1 | 2 | 1 | 4 |
| 场景2 | 10 | 5 | 11 |
| 场景3 | 30 | 20 | 67 |
| 场景4 | 60 | 28 | 169 |
| 场景5 | 90 | 40 | 364 |
| 场景6 | 120 | 53 | 586 |
| 场景7 | 150 | 55 | 648 |

（2）同一规模不同相交程度的场景下碰撞检测性能测试

在本实验中运用控制变量法，探究在同一规模不同相交程度情况下碰撞检测的效率。对场景3中的30个模型执行不同碰撞程度的性能测试，检测本文算法的性能，测试结果见下表3-3和图3-11所示。从表和图中可明显看出，场景中物体之间的相交程度对碰撞检测效率影响较大，随着碰撞物体对数量的不断增加，检测时间曲线不断上升。可以得到结论：稀疏场景和稠密场景下碰撞检测效率相差很大，碰撞检测的时间消耗基本在200ms以内，在几次检测过程中均未产生误报情况，基本能够满足微波管仿真环境下的碰撞检测实时性和精确度的需求。

表3-3 同一规模场景下碰撞检测数据

|  |  |  |
| --- | --- | --- |
| 碰撞物体对数 | 检测时间/ms | 误报率 |
| 7 | 31 | 0 |
| 10 | 37 | 0 |
| 19 | 73 | 0 |
| 26 | 116 | 0 |
| 43 | 142 | 0 |
| 74 | 162 | 0 |
| 81 | 199 | 0 |

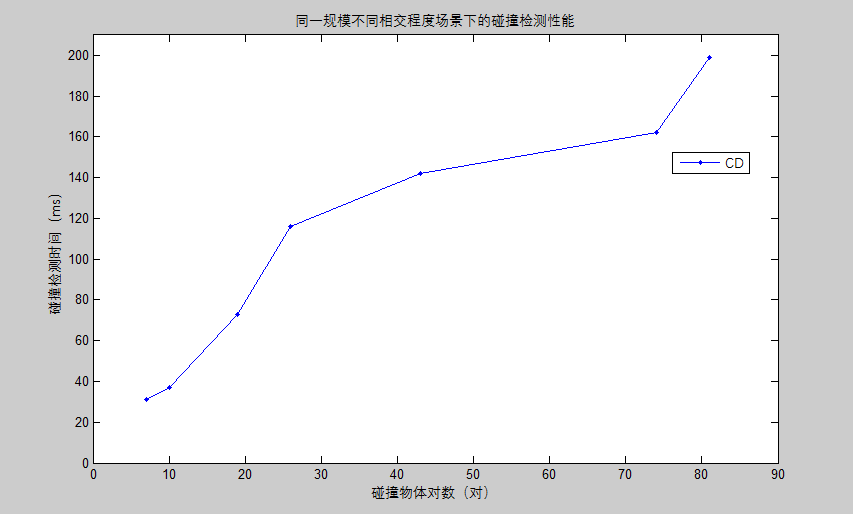


图3-11 不同相交程度下的碰撞检测数据

3.7 本章小结

本章主要介绍了基于混合层次包围体的碰撞检测算法的基本原理。首先，结合AABB包围体和OBB包围体在混合BVH树的不同层次进行布局，提出了一种混合BVH树结构，并分析了这种结构的结构优势，阐述了算法的实现流程。然后，研究和分析了BVH树的遍历和更新方式，提出遍历BVH树的各个节点包围体的算法伪代码。之后，研究和分析三角形图元之间的相交测试算法，并比较了几种相交测试算法的计算量。接着详细阐述了GJK算法计算凸体之间的分离距离和穿透距离的算法原理和求解流程。最后，通过不同场景下的碰撞检测实验结果验证了算法的可靠性和稳定性。

第四章 微波管参数化建模软件平台设计与实现

4.1 系统总体架构设计

**4.1.1** 系统的需求分析

随着计算机图形学和几何建模的迅速发展，运用三维几何建模技术搭建的虚拟场景已经广泛存在于人们的生产和生活中，如CAD/CAM软件、动画制作和游戏场景等等。由于内存、处理器等的限制以及实时性要求，计算机游戏和动画等场景中对三维几何模型的需求仅在于显示模型的主要特征，并不要求过于精细的模型细节。但是，微波管如行波管、速调管等电真空器件属于高度精密的仪器，核心组件及其零部件的关键尺寸的微小误差都可能会在后续的仿真计算中被扩大，从而引起该微波管整体性能的巨大变化，影响微波管设计的合理性和可靠性。因此，微波管的设计、装配和仿真过程要求精确显示模型特征和不断的调整模型参数。为实现高效、快捷、方便的微波管的参数化建模功能，微波管参数化建模软件系统设计要考虑到以下的具体功能需求：

（1）搭建图形用户界面以实现与设计人员的人机交互操作和对用户操作的实时响应。微波管零部件属性和参数的实时显示和迭代修改对于设计过程至关重要，任何的延迟都会严重影响行波管设计的效率。显示与管理零部件模型的创建和装配记录的功能，能够使用户清晰的看到零部件组成，并控制各个部件在组件里的参数显示，是参数化建模和设计时必须实现的。

（2）为了实现对物理世界的更逼真的模拟，必须使用碰撞检测算法对虚拟的微波管模型进行物理仿真，使之符合真实的微波管生产环境的特性。碰撞检测功能对于微波管设计和制造的合理性和可行性具有至关重要的作用，同时也是零部件的设计和装配过程中较大的时间资源消耗点。微波管的仿真环境要求碰撞检测算法能够高精度和快速的检测到零部件之间的碰撞和干涉现象，并直观的显示碰撞和干涉的检测报告，为相关设计人员及时调整关键尺寸参数提供参考。

（3）微波管是结构复杂，加工要求苛刻，生产周期长，高度精密的电真空器件，其三维电磁仿真、力学仿真、热学仿真等都对所构建的模型有较高的精确度要求[64]。微波管的仿真过程经常需要不断根据仿真结果调整模型的尺寸、位置等参数以进行迭代仿真。因此，参数化建模系统必须提供高精确度的几何造型功能，并支持对模型的参数调整和不断迭代修改。

**4.1.2** 系统框架

为满足上述系统功能需求，本章采用ACIS引擎进行三维几何的精确建模并利用HOOPS套件进行高性能的模型渲染，自主开发了一款面向微波管设计和开发流程的微波管参数化建模软件平台，该平台现已经成功应用于课题组研发的微波管仿真模拟套装MTSS中。该软件平台的软件环境是Windows 10操作系统，采用Visual C++为编程语言，微软基础类库（Microsoft Foundation Classes, MFC）、HOOPS套件和ACIS引擎为开发工具，提供了基于特征的参数化建模等功能。硬件环境为Intel(R) Core(TM) i7 CPU 860@ 2.80GHz处理器和8GB的RAM。

微波管参数化建模软件平台的总体框架图如下图4-1所示，主要包括图形用户界面、参数化建模模块和特征文件读写等模块。图形用户界面基于MFC的对话框和相关组件开发，包括组合菜单栏、特征导航器、属性浏览器、特征实体创建和编辑对话框等。参数化建模模块继承与扩展了HOOPS/MVO中的三大基类，并封装了ACIS的实体模型类为几何特征类，支持基本几何模型的创建、曲线/曲面的特征造型设计、捕捉点辅助设计、对模型的编辑修改和历史流管理（撤销和重做）的特征操作。碰撞检测求解器模块将混合碰撞检测算法封装为专有的类，应用程序通过调用相关函数将数据传入求解器模块，经过碰撞检测算法的处理后得到碰撞检测结果信息，支持静态碰撞检测、干涉检查和间隙分析操作。此外，特征文件的读写模块使用自定义的xml数据格式存储创建好的特征数据和历史记录，以支持对模型数据和历史的读写、修改和分析操作。接下来，将详细介绍各个模块功能的具体设计、实现与操作流程。

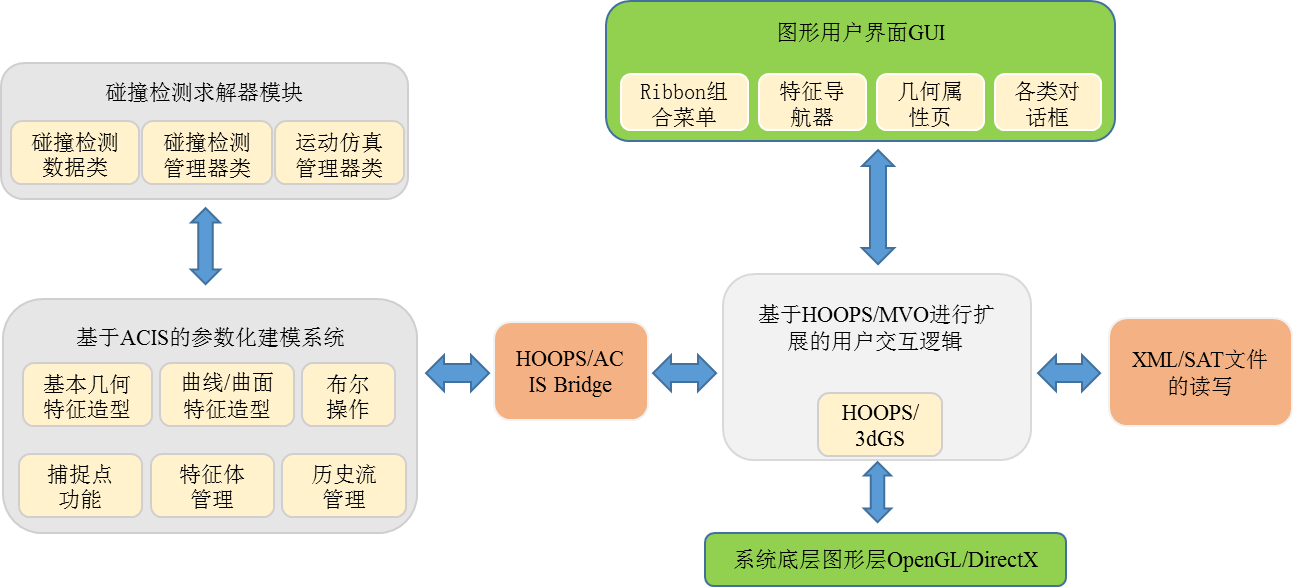


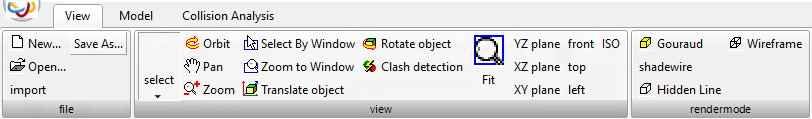
图4-1 微波管参数化建模软件平台的系统框架

4.2 图形用户界面的实现

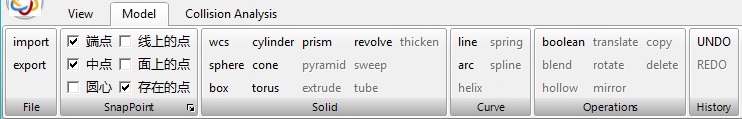
图形用户界面（Graphical User Interface, GUI）提供用户与应用程序的实时交互，主要包括视图操作、建模操作、碰撞检测求解器的菜单组，设置特征参数并实时渲染显示模型的对话框，记录场景中模型创建记录的特征导航器以及显示几何属性的属性浏览器界面等，具体实现如下。

**4.2.1** 组合菜单栏

组合式Ribbon菜单栏使用一个简单的接口系统取代传统的菜单、工具栏和任务窗格，使控件窗体中的相关命令聚合成分组，并使相关的分组聚合到标签中。本软件平台中实现了三个菜单分组，分别是基于HOOPS的视图显示（View）菜单组、基于ACIS的建模菜单组（Model）菜单组和碰撞检测（Collision Analysis）菜单组。界面显示如下图4-2所示。



(a) Start菜单组



(b) 建模菜单组



(c) 碰撞检测模块菜单

图4-2 Ribbon操作菜单界面

其中，图4-2（a）显示的是视图显示菜单组，主要实现了文件读写、对视图相机的管理、模型的渲染模式和模型变换操作。最左侧的file区是新建文件、导入文件等文件操作功能按钮，中间view区包括选择器（包括选择点、线、面、体）命令组合按钮、视图的相机控制操作如旋转视图（Orbit）和视图变换操作如模型的标准视图（ISO）显示等命令按钮。此外，还提供了平移对象（Translate object）、旋转对象（Rotate object）、选择对象（Select by window）等几何变换操作的功能命令。在渲染模式区是着色（Gouraud）、背面隐藏的线框着色（Hidden Line）、静态线框（Wireframe）等渲染命令按钮。

图4-2（b）显示的是建模操作菜单组，最左侧的File区为模型文件导入（import）和文件导出（export）操作按钮；捕捉点功能Snap Point区主要包括对线的端点、线的中点、圆心点、线上的点、面上的点和存在的点的捕捉，用户可以通过6个复选框选择开启/关闭应用程序捕捉特征顶点的功能；实体特征建模Solid区包括创建WCS、球体（sphere）、方块（box）、圆柱体（cylinder）、圆锥体（cone）、圆环体（torus）和回转体（revolve）操作按钮；曲线特征Curve区包括直线（line）、圆弧线（arc）的创建操作按钮；特征操作Operators区包括布尔操作、平移、旋转模型等操作按钮。在4.3节中，将详细介绍特征建模操作模块的设计和实现过程。

图4-2（c）显示的是碰撞检测菜单组，在下文4.4节中将详细说明本文的混合碰撞检测算法应用到此软件平台的功能实现。

**4.2.2** 属性浏览器

属性浏览器显示几何模型的点、线、面和体的各种参数信息，能够帮助用户更高效和快速的设计与分析微波管特征模型的属性。属性浏览器可以通过Ribbon菜单的选择器（图4-2（a）视图菜单栏的Select组合按钮），然后在场景中选中相应的几何元素，在界面右侧的属性浏览器中显示几何属性。在属性浏览器中显示几何属性的操作流程如下图4-3所示。

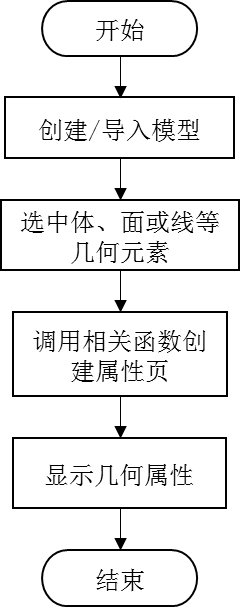


图4-3 属性显示的操作流程

曲面属性浏览器界面如下图4-4所示，其中显示了曲面的参数信息如曲面的类型、锥顶点、轴线、底面圆心、曲面材料特性等属性。体属性界面暂时仅显示模型的材料属性；曲线属性界面显示曲线的类型、其他相关参数（如端点、圆心点等）；顶点属性界面显示顶点的坐标值；WCS属性界面显示坐标系的原点和三个坐标轴的参数信息并提供WCS的修改操作。

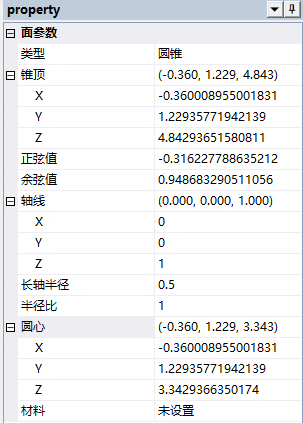


图4-4 几何属性浏览器界面

**4.2.3** 特征导航器

特征导航器是一个记录特征模型创建历史的树状结构，也可称为特征树，支持对特征体的高亮显示、修改等功能。

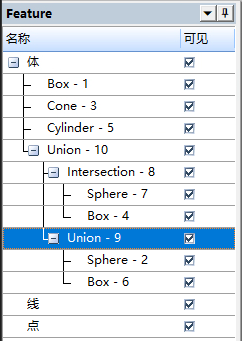


图4-5 特征导航器界面

在上图4-5所示的特征树结构中，每一个模型组件作为特征树结构的一个节点显示，每个节点都存储了到一个特征模型的映射。这样，不仅能清晰的看到模型创建的历史记录，而且能提供给用户在必要时快速选取和修改各个特征模型的功能。此外，用户可以根据不同的操作要求在特征导航器中选择相应的特征实体，完成其他特征操作的功能；还可以编辑参数修改特征实体、改变显示部件和隐藏部件等。特征导航器界面的用户交互操作主要包括弹出编辑对话框实时显示与更改模型、调用HOOPS/3dGS的API渲染或显示模型、建立特征树节点与特征模型对象的映射表，具体实现如下：

（1）双击特征树的节点后，弹出节点所映射特征模型的编辑对话框。如下图4-6所示为立方体的编辑对话框，在对话框中显示了立方体特征模型的参数信息提供给用户参考，并支持用户编辑参数以修改立方体模型。

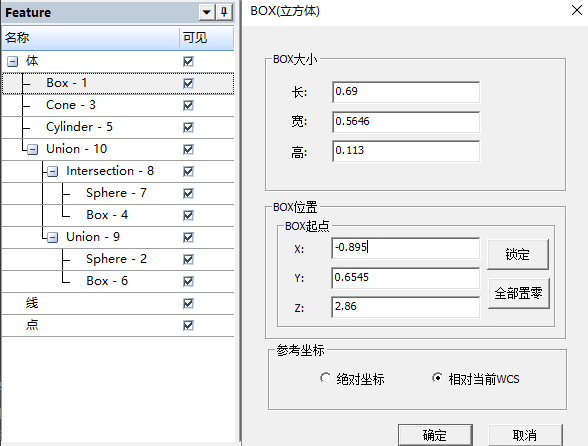
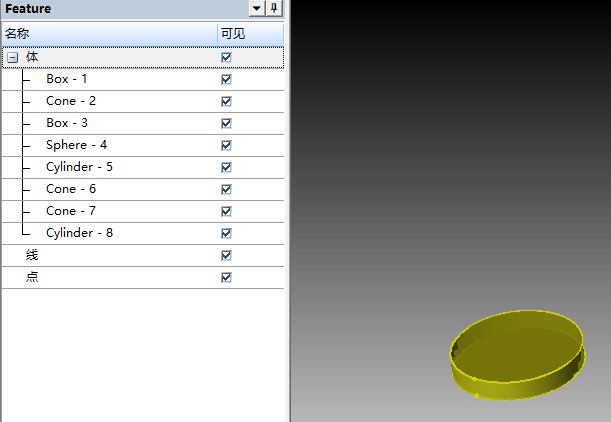
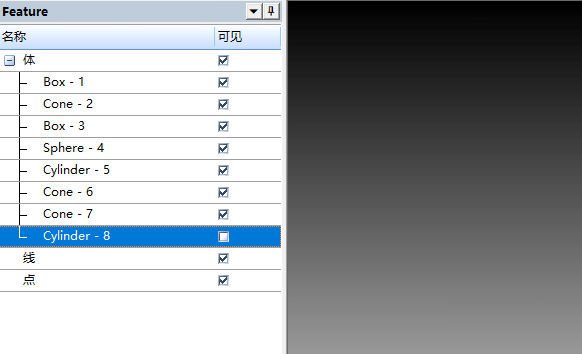


图4-6 双击节点显示编辑对话框

（2）特征导航器调用HOOPS渲染和显示模型的主要操作是选中特征树的模型节点后，将节点映射到的特征模型进行高亮显示，如下图4-7（a）所示，以及节点右侧的“可见”复选框的勾选/不勾选决定模型的显示/隐藏，如图4-7（b）所示显示的是取消勾选特征树节点的“可见性”选项后隐藏模型。

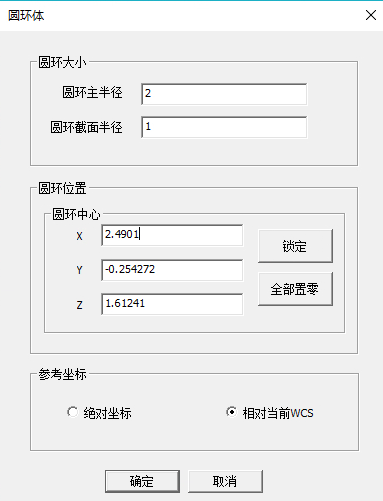
 

图（a）高亮模型图 （b）隐藏模型

图4-7 特征导航器的相关操作

**4.2.4** 对话框界面

对话框是微软基础类库（MFC）提供的一类重要的用户界面，能够接受应用程序处理用户输入的处理结果并输出到屏幕上。实际上，对话框是一种窗口，它不仅能接收来自外部设备的系统消息（如鼠标和键盘操作的消息），而且能够非常直观的显示应用程序的数据，并支持移动和关闭。此外，开发人员可以在对话框上设计和使用各类用于用户交互的控件如静态文本、编辑框、组合下拉框等。

（a）圆环体创建对话框 （b）WCS创建对话框

图4-8 建模对话框

对话框广泛存在于参数化建模的过程之中，比如鼠标选中Model菜单栏的建模操作按钮后，会弹出建模对话框，用户可在建模对话框的辅助下在视图场景中进行建模。如图4-8所示为创建圆环体（图4-8（a））和创建WCS工作坐标系（图4-8（b））时弹出的建模对话框，通过修改对话框中的圆环体半径、圆环中心或WCS的原点位置等相关参数，可以修改实体模型或WCS并在场景中实时更新显示。

4.3 视图操作和模型渲染

视图显示操作模块中，模型的相机操作和视图变换是在HOOPS的基础上实现的，相机控制操作和视图变换操作仅改变了位于屏幕处的相机的位置，二者均未修改底层的ACIS实体数据。以缩放操作的代码为例，执行相机缩放操作的流程如下：

void CNQView::OnZoom() //zoom按钮命令的消息响应函数

{

LocalSetOperator(new HOpCameraZoom(m\_pHView)); //设置当前视图的操作器为视图缩放操作器

}

void CNQView::OnUpdateZoom(CCmdUI \*pCmdUI)

{

if (streq(m\_pHView->GetOperator()->GetName(), “HOpCameraZoom”)) //判断当前操作器是否为缩放操作器

pCmdUI->SetCheck(1); //设置Zoom按钮状态为按下

else

pCmdUI-> SetCheck(0);//还原Zoom按钮状态

}

上述操作流程主要涉及菜单栏按钮的响应函数和HOOPS/MVO中基于HBaseOperator的操作器类库，其中消息响应函数是通过MFC的消息循环和消息映射机制实现的，操作器类将在下一节中介绍。如下图4-9所示为场景中视图的相机执行缩放操作后显示的方块模型。

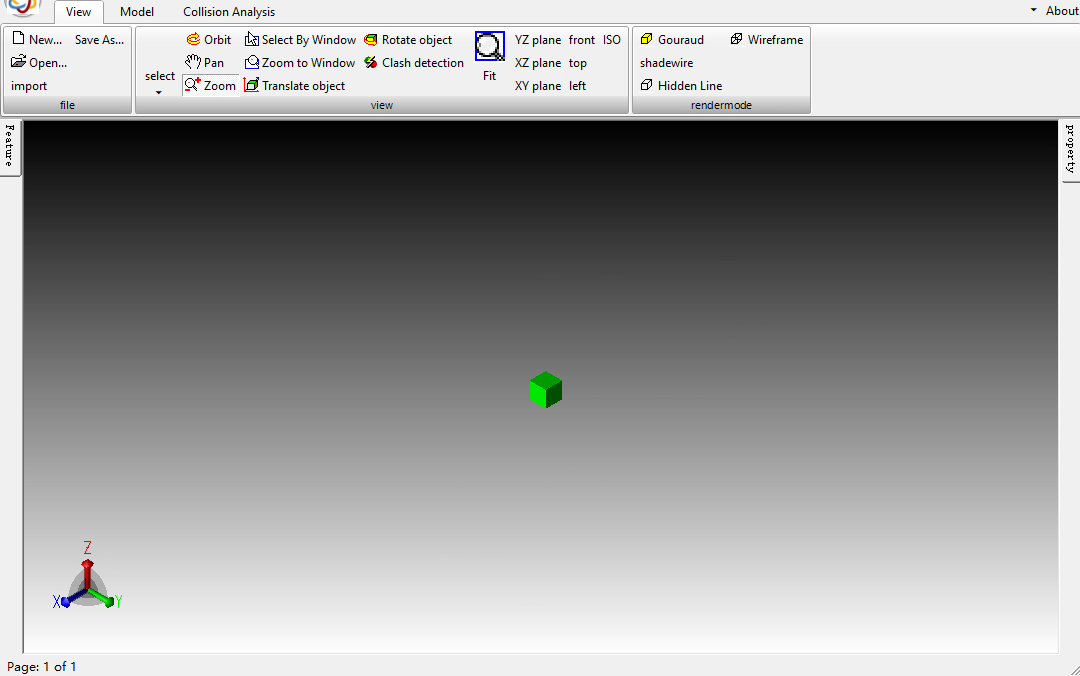


图4-9 缩放后的方块

模型渲染操作是基于ACIS和HOOPS之间的数据通信完成的，本平台的渲染流程实现如下图4-10所示。应用程序采用ACIS的API创建实体之后，打开HOOPS的图形段结构，通过HOOPS/ACIS桥的HA\_Render\_Entity()函数将ACIS实体渲染到场景中，实质上是将ACIS实体映射为HOOPS的键值并插入到HOOPS的图形数据库之中，然后通过HOOPS/3dGS的渲染管道调用底层的图形库渲染和绘制ACIS实体模型。此外，可以通过HOOPS/ACIS桥的HA\_Compute\_Geometry\_Key()函数将ACIS的实体映射成HOOPS的键值，然后打开键值对应的段并调用HOOPS的API函数进行模型渲染选项设置如高亮实体、设置线框宽度等。



图4-10 模型渲染流程图

4.4 基于特征的参数化建模系统

本节设计的参数化建模系统在ACIS和HOOPS/MVO类的基础上扩展了几何造型和模型操作的功能并支持模型参数的实时调整。

**4.4.1** 建模系统的设计

特征建模方法能更为详尽的表达几何模型功能和形状信息，通过维护相关信息加速三维模型的建立过程并有利于设计和制造的信息传递与数据交换。特征建模模块是整个软件平台的主体部分，采用的设计模式是基于HOOPS的模型、视图和控制器（MVO）的设计模式，在此基础上，本文构建了特征造型框架体系。在模型（Model）方面，基于ACIS造型引擎的建模体系，实现了对ACIS的几何造型类的封装和扩展，形成了一套特征模型类的框架。基于面向对象的设计思想，将ACIS的点、线、面、体等几何实体类封装为参数化的特征类。以AHFeatureENTITY类作为公共的抽象基类，顶点特征类以一组三维空间中坐标点集合的形式列出，存储的参数为顶点的空间坐标值；曲线特征类是对ACIS的基本曲线如椭圆线、直线的参数化管理，存储实体类指针和端点、半径比等曲线相关参数；实体特征类是对ACIS的几何实体如立方体、球体、圆锥体、回转体等进行参数化管理。表征特征类之间关系的UML类图的部分如下图4-11所示。

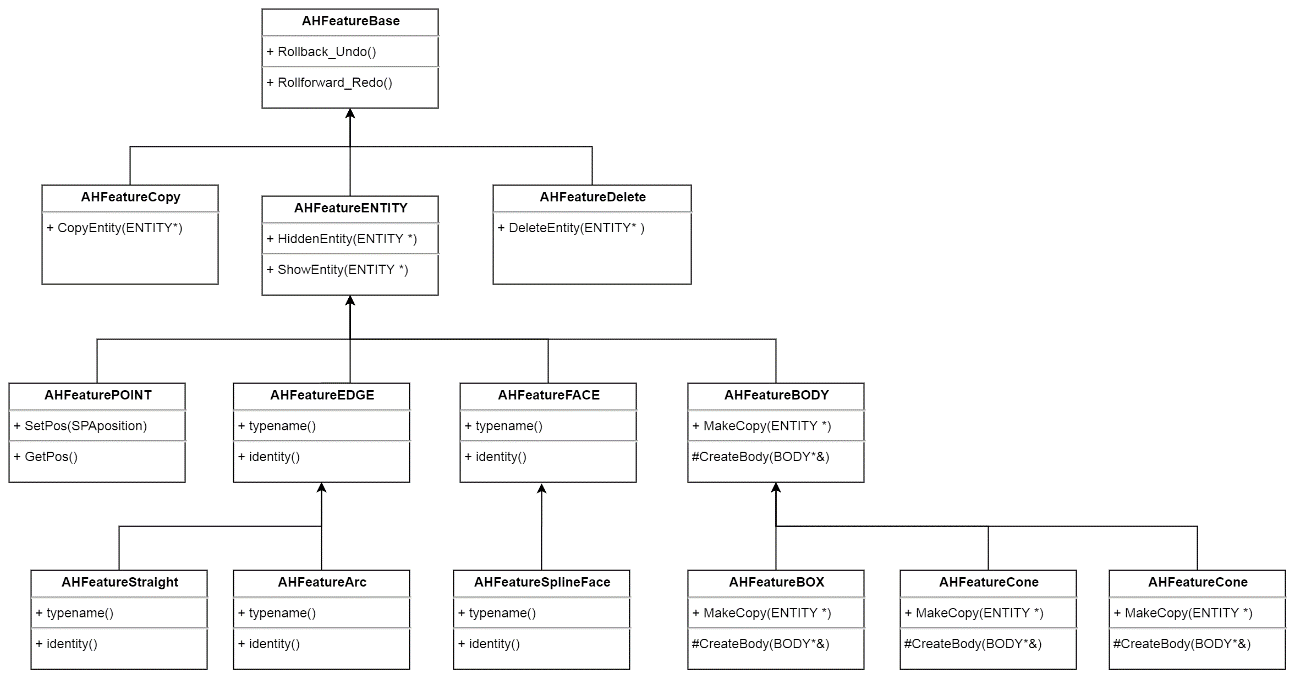


图4-11 特征类的UML类图

视图（View）方面，基于MFC的对话框为每个实体特征和造型操作建立了一个对话框视图。在前文4.2节中我们简要介绍了对话框，对话框可以将应用程序的数据输出到图形界面上供用户分析和设计。在此，建模对话框视图可以将几何体的参数信息输出到用户界面上，通过在对话框中设置模型的参数来实现对模型的编辑和更改，并实时显示参数更改后的几何模型，以达到实时的用户交互功能，主要包括立方体对话框、圆锥体对话框和布尔操作对话框等。

实体操作器类提供了对用户交互操作的支持，具体实现是以HOOPS/MVO中的基础操作器类（HBaseOperator）作为基类，设计了一套特征创建操作器的类库。主要功能是响应鼠标按键消息，操纵应用程序的数据实现特征建模和特征操作的功能。在组合菜单栏中的建模菜单组中，选定模型创建按钮后激活相应的模型特征创建操作器。接下来，用户可以操作鼠标在视图场景中建模和操作。本软件平台中的操作器类与实体特征类一一对应，即每一个实体特征对应于一个创建实体的操作器，以满足用户交互操作需求。操作器主要包括立方体、圆锥体、圆柱体、圆环体的特征操作器以及布尔操作、回转操作等的操作器。其中圆锥体类操作器的头文件代码如下图，操作器类和底层的消息循环紧密关联，可以看到每个鼠标按键响应函数的参数均为消息事件类，可以说操作器是根据鼠标事件消息做出相应的动作。

class HOpCreateSolidCone : public HBaseOperator

{

public:

HOpCreateSolidCone(HBaseView\* view, int DoRepeat = 0, int DoCapture= 1);

~HOpCreateSolidCone();

virtual const char \* GetName();//获取操作器名称

HBaseOperator \* Clone(); //返回一个操作器拷贝的指针

virtual int OnLButtonDown(HEventInfo &event);// 鼠标左键按下消息响应函数

virtual int OnNoButtonDownAndMove(HEventInfo &event);//鼠标悬浮消息响应函数

virtual int OnLButtonUp(HEventInfo &event); //鼠标左键抬起的消息响应函数

private:

void CreateSolidCone(const SPAposition &pt1); //创建圆锥实体的函数

DLGSolidCone\* dlgCone;//圆锥实体的建模对话框指针

}

经过上述设计，特征体类、模型创建的操作器类和对话框类组成了特征建模的MVO架构。这种设计模式实现了模块化编程，提高了代码的可维护性和扩展性。

**4.4.2** 建模系统的实现

1. 基本几何模型的设计与创建

基本几何模型主要包括立方体、球体、圆锥体、圆柱体和圆环体等基本几何模型的建模。接下来，通过下图4-12为例简要说明圆锥操作器类创建圆锥实体的过程，其它基本几何造型操作均与此类似，具体操作流程如下：

（1）首先，鼠标悬浮在空间时，在对话框的圆锥体位置编辑框中实时显示当前鼠标所在空间点位置，其中，圆锥的其他参数具有默认值，而底面法线可以通过事先创建的工作坐标系WCS的相对坐标或者是世界坐标系下的绝对坐标确定。

（2）鼠标左键点击场景空间，选定空间中的一个点作为圆锥底部的椭圆中心。锁定时，无法再通过鼠标选中点，解除锁定时，才可以再次选中；若选中“全部置零”按钮，那么底部椭圆中心坐标值将全部归零。

（3）通过对话框可以进行几何模型的迭代设计和修改。修改对话框中圆锥的长轴半径、圆锥高度等参数信息，修改后的实体会实时显示在场景视图中。

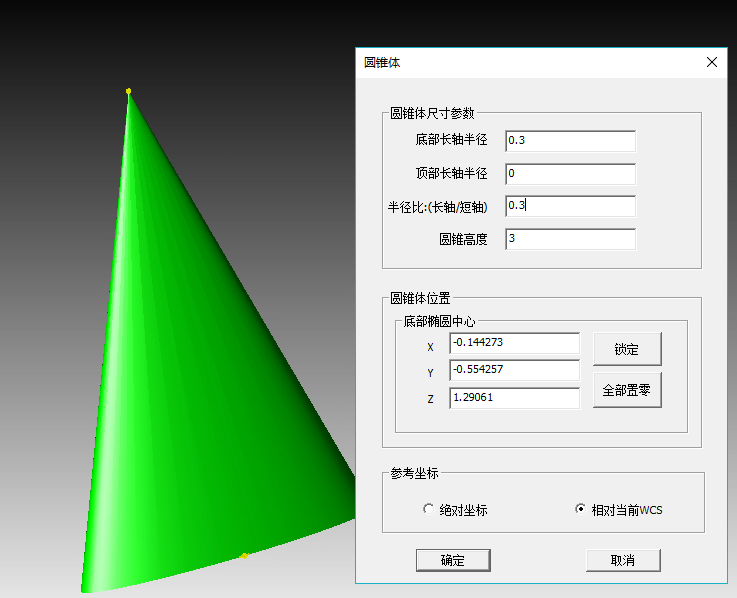


图4-12 圆锥体的创建结果

2. 高级几何造型

特征建模模块支持复杂实体的几何造型设计与创建。曲线或曲面可通过扫掠、拉伸、回转等操作，生成复杂的自由曲面实体造型；三维实体可以通过布尔运算操作生成复杂的几何模型。

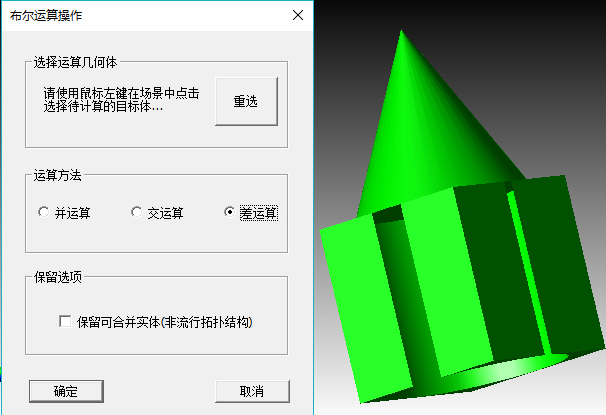


图4-13 布尔差运算建模结果

几何造型设计中的集合运算又称为布尔运算，该运算最初出现在数学家George Boole提出的逻辑代数学中。几何模型的布尔运算如交、并、差等操作可以当作是点集合的集合运算。布尔并运算指主体中的点集合和客体中的点集合求并集，该操作将两个实体合并为一个体。布尔交运算指将参与运算的主体中的点集和客体中的点集求交得到公共点集，结果是得到两个实体的公共部分。布尔差运算是指主体中的点集和客体中的点集求差运算，得到主体中去掉在客体中的部分。如图4-13所示为使用布尔运算功能创建的实体造型，布尔运算功能的操作流程如下：

（1）选择第一个体作为目标体；

（2）选择第二个体作为工具体；

（3）选择保留/不保留工具体；

（4）在两个体之间应用布尔运算，产生一个新的布尔体。

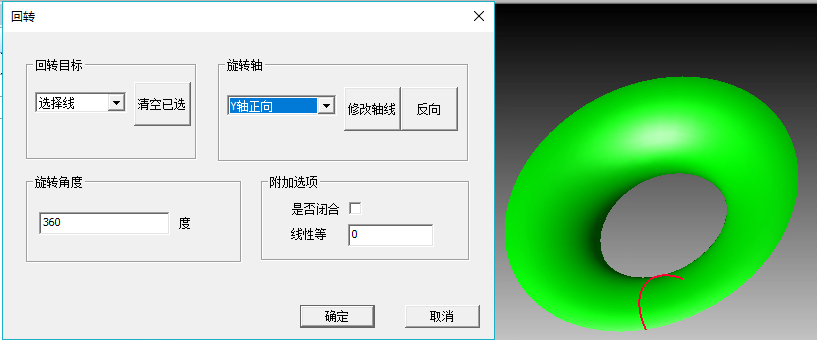


图4-14 回转实体造型

通过回转特征操作创建的自由曲面实体如图4-14所示，回转造型功能的操作流程为：

（1）选定一个旋转的基本几何称为回转体，该回转体可以是曲线或曲面；

（2）给出回转体的运动路径或旋转轴，可以是一条曲线、向量或者坐标轴；

（3）设置旋转角度，回转特征操作将回转指定的角度；选择闭合，则回转操作会将回转形成的曲面闭合成为一个实体。

**4.4.3** 捕捉点功能的实现

在用户进行建模或者绘图操作时，经常会有从直线的中心点、端点等特征顶点开始创建模型的需求，使用人眼找到这些特征顶点当然是非常低效和不准确的，尤其对于建模精度要求较高的微波管参数化建模，目标点的捕捉功能能够高效、精确的辅助用户确定点。软件平台支持捕捉几何元素（如线、面、体）上的特征点的功能。捕捉点功能的操作流程为：（1）首先选中Ribbon菜单栏图上的端点、中点、圆心点和线上的点、面上的点、存在的点等复选框，以启用对相关类型的顶点的捕捉操作；（2）然后，移动鼠标至视图场景中，当鼠标悬浮在模型上时，会高亮显示模型上对应位置的端点、中点和圆心等顶点；（3）鼠标的移动会带动如线上的点、面上的点等捕捉到的顶点的移动，点击鼠标左键后会在场景中创建一个顶点标记。

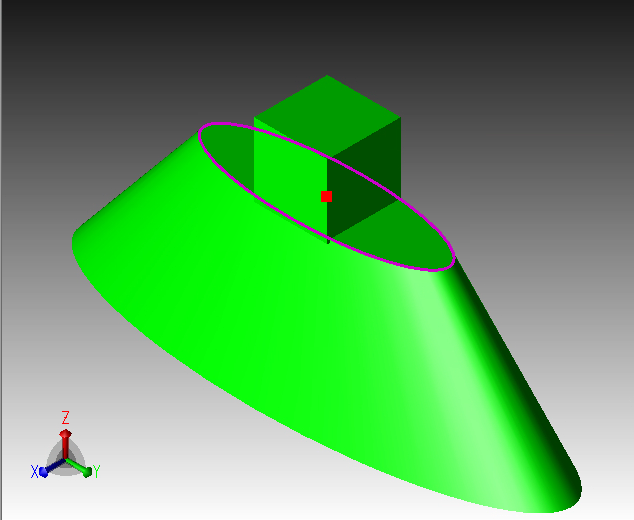


图4-15 捕捉点功能展示

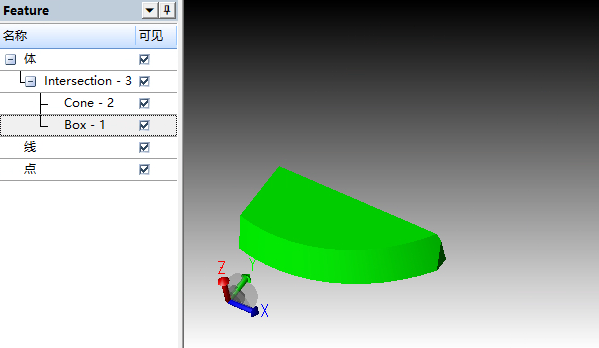
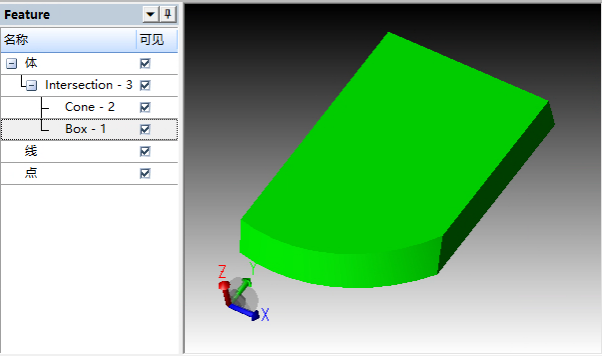
捕捉点功能实例如图4-15所示，此实例启用了捕捉圆心点的选项。当鼠标悬浮到椭圆柱体顶部的椭圆附近（一定范围内）时，捕捉器会拾取椭圆的圆心点并高亮显示。若在建模时启用捕捉点功能，则可以利用捕捉到的点辅助建模，图中将椭圆心作为立方体的底部中心点，进而创建立方体特征。

**4.4.4** 特征历史流的管理

本软件平台支持建模过程的反悔机制，这个机制支持用户进行回溯操作。正如在其他软件中处理数据时常用的撤回和恢复操作一样，回溯机制提供给设计人员在任何时候撤销（Undo）上一次的建模操作和重做（Redo）之前被撤销的一次建模操作。这个功能对微波管的仿真系统来说很重要，设计人员可以测试某个设计思想并对现有的模型进行撤销修改，在测试完成之后再恢复到模型修改之前的状态。

特征历史流管理功能基于ACIS的回溯机制，具体而言，每个特征模型在被创建或修改后，记录此次操作的修改状态，并插入到历史流管理器类的历史记录链表中，同时，建立历史记录与被创建或修改的模型的映射表。在用户执行撤销和重做指令操作时，由历史流管理器对模型操作的历史进行统一管理；Undo时，将当前历史记录下的最后一次操作执行回退；Redo时，通过历史记录链表获取被撤销的操作历史并恢复。接下来，通过以下实例来讲解历史管理的工作流程：

（1）创建一个立方体Box-1和一个圆锥体Cone-1模型，并对两个实体执行布尔交运算，得到如图4-16（a）所示的结果体Intersection-3；（2）双击下图特征导航树中的立方体特征Box-1节点后，弹出立方体特征模型的编辑对话框，修改尺寸参数，以立方体特征作为工具体的布尔体随之修改，如图4-16（b）所示；

（a）布尔交运算后的模型 （b）修改立方体参数后的布尔运算模型

图4-16 特征历史流操作示例图

（3）点击Undo按钮，立方体修改操作被撤销，但作为布尔操作的工具体，立方体并不会显示出来，执行撤销回退操作后布尔交运算结果体还原为图4-16（a）所示的模型。

（4）点击Redo按钮，重新对Box-1执行之前被撤销的尺寸修改操作，此时，以Box-1为工具体的布尔运算结果体同步更改并渲染显示，如图4-16（b）所示。

综上所述，本文软件平台的历史管理不仅实现了对单一模型操作的几何造型的回溯功能，而且实现了对布尔操作的回溯功能。布尔操作的建模过程可以重复和迭代执行，参与布尔操作的实体构成的树状结构将是一颗任意的二叉树，在本软件平台中实现了对布尔操作的工具体树中任意节点上的实体（即参与布尔操作的任意工具体）的更改与回溯功能，以辅助用户利用布尔操作设计和创建特征造型。

4.5 碰撞检测求解器模块

在参数化建模软件平台搭建的可视化虚拟仿真环境的基础上，应用第三章所开发的碰撞检测算法实现了碰撞检测求解器模块，并将该算法求解器嵌入到可视化的微波管参数化建模软件中。使用碰撞检测求解器进行算法求解的程序流程如图4-17所示。在初始化碰撞检测环境和建立/导入模型之后，可以通过函数调用开启碰撞检测算法。通过选择检测对象或者以场景中所有可见模型，将待检测的特征实体组合成一个集合数组并设置为活跃集，然后将特征实体的属性信息转换为碰撞检测模块的模型数据，完成检测的预处理操作，并进入到碰撞检测流程。这里主要介绍第三章提出的结合AABB和OBB包围体的混合层次包围体树进行检测的流程，碰撞检测流程分为三个阶段：初步碰撞检测阶段，精确碰撞检测阶段和相交测试阶段。初步检测阶段通过构造AABB包围体以实现快速排除距离较远的物体对，并得到存在干涉可能的潜在碰撞对；在精确碰撞检测阶段，输入所有潜在的碰撞对并为其构造OBB包围体树结构，递归检测两颗BVH树中的OBB包围体节点，若检测到不碰撞，则返回，否则得到发生干涉的基本图元特征对；最后，在相交测试阶段对两个基本图元（三角形面片）进行相交测试，返回发生碰撞的模型信息、发生干涉的面对信息以及模型之间的最接近距离等碰撞数据。

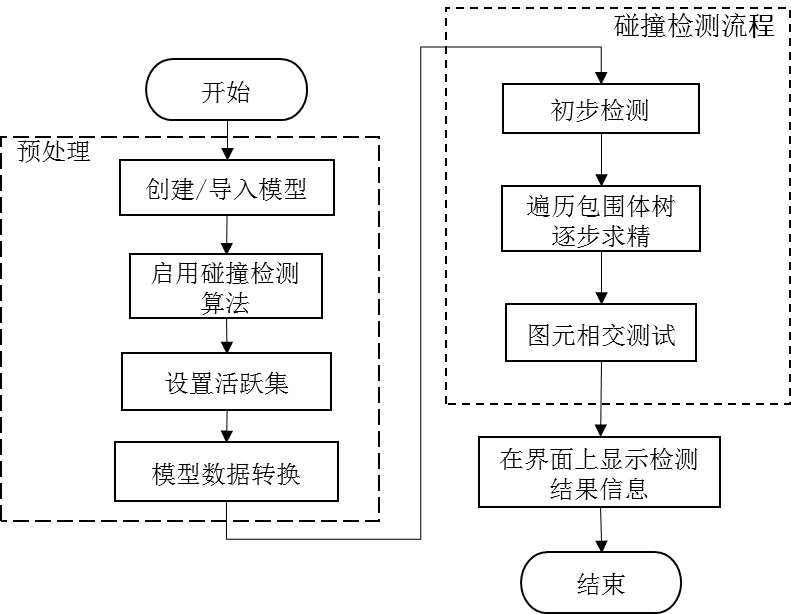


图4-17 碰撞检测求解器的操作流程

该求解器模块的功能是将场景空间中的部分或全部几何模型按需加载到模型数据类中，经过碰撞检测算法求解器的处理，输出详细的碰撞信息。碰撞检测子模块采用独立的类将求解计算过程封装起来，以函数接口的形式供应用程序直接调用，在图形界面窗口中实时显示检测结果信息。如下代码描述的是GpcCDManager碰撞检测管理器类中获取基本干涉检查结果的函数接口，vector和pair这两种数据结构是碰撞检测管理器类中最常用的容器，干涉实体对在函数的容器参数中返回。

CD\_OUTCOME GetALLCollidingSetPairs (

vector<pair<BODY\*, BODY\*>>& interSetPairs//检测出干涉的实体对列表

);

碰撞检测求解器模块嵌入到可视化软件平台后的部分功能实现如下：

（1）静态干涉检查是一种检测场景中所有实体模型的碰撞情况的功能，可以快速判断出场景中的所有模型之间是否存在碰撞，一旦检测到模型之间的碰撞现象则立刻返回结果，因此该操作时间消耗较少，功能展示如下图4-18所示。



图4-18 静态全局碰撞检测

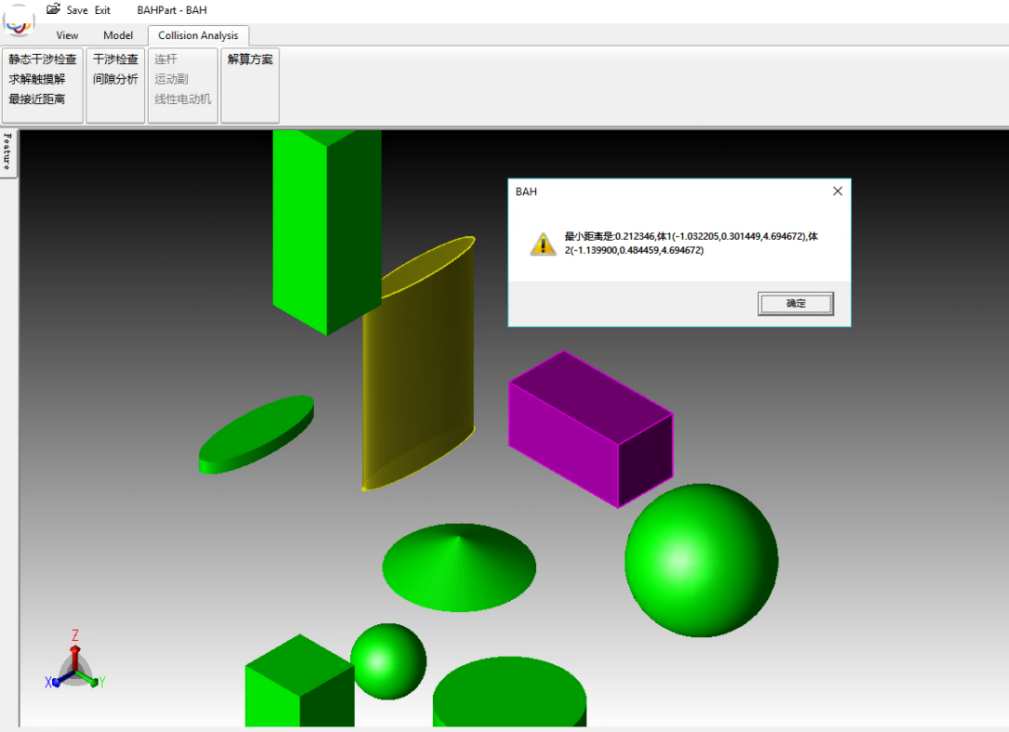


图4-19 椭圆柱和立方体间的最接近距离计算

（2）最接近距离计算功能可计算出两个几何集合之间或一个几何集合与一个点之间的最近距离。通过GJK算法计算物体对之间的Minkowski差到原点的最近距离值，即可得到物体之间的最接近距离值，如上图4-19所示为椭圆柱和立方体之间的最近距离的计算结果。此外，干涉检查和间隙分析功能将在第五章中详细介绍。计算最近距离的函数声明如下：

CD\_OUTCOME Compute\_closest\_approach(

ENTITY\* body1,//实体1的指针

ENTITY\* body2,//实体2的指针

double\* dist,//输出的实体之间的最近距离

double secureDist,//实体间的合理间隙距离

PAIR\_STATUS colStatus,//碰撞状态

CLAP\_OUTCOME\* colRes);//碰撞结果

4.6 特征文件的存储和读取

标准ACIS文本文件(Standard ACIS Text, SAT)是ACIS造型引擎的标准模型文件存储格式。SAT文件采用ASCII码的形式保存ACIS内核中的实体数据，ACIS支持将用户定义的属性数据和实体数据一起保存在SAT文件中，此外，ACIS提供了专门的API函数对SAT格式的文件进行数据的读入和写出操作。XML是一种通用性极强的可扩展性标记语言，可以应用于任何应用程序中，主要优点是数据读写方便，结构清晰，显示直观。

本软件平台支持XML和SAT两种文件格式的读写操作。特征文件的存储使用XML文件格式，文件内容严格按照格式规范，主要包括版本信息、特征体个数、模型参数、特征体的编号和特征体类型等信息，其中，模型参数包括WCS工作坐标系的参数（原点坐标、x轴向量和y轴向量）、变换矩阵（transform matrix）参数（如平移方向、旋转角度等）和其他几何参数（如球心等）。此外，本文软件平台支持对从xml文件格式读入的模型的历史操作功能。存储特征模型文件时，应用程序依次读取特征管理器的特征对象链表和特征历史链表，基于面向对象编程的多态性原理，通过特征基类对象调用每个具体特征实体的序列化回调函数，将几何模型的属性信息和操作历史按照一定的格式规范标准存储为xml特征文件。读取特征模型文件操作的步骤为：

（1）点击建模菜单栏的import按钮，应用程序根据消息映射表调用OnFileImport函数，弹出打开文件对话框，在用户打开sat/xml特征文件后，应用程序将调用如下函数读取文件：HFileInputResult Read(const char \*Filename//文件名);

（2）判断文件类型，若文件是sat格式，则调用HA\_Read\_Sat\_File函数读取sat文件，并得到ACIS的实体列表，HOOPS/ACIS bridge的内核将所有读取到的实体通过HOOPS渲染到视图中；若文件是xml格式，则标签类型确定特征实体的类型，并调用与特征实体相对应的XMLCallback回调函数解析xml文件数据；

logical HA\_Read\_Sat\_File(const char \*fname,//文件名

ENTITY\_LIST& entityList//读取到的几何实体列表

);

void XMLCallback(HUtilityXMLTag \*xt,//xml标签信息压缩类

bool open,//文件是否打开

void \*m\_pExtraData//额外的数据

);

（3）根据读取到的几何实体数据以及相关参数，创建对应的特征实体对象，然后将创建后的特征实体插入到特征实体管理器中。若导入的文件为sat格式，则到此结束读取，若导入文件为xml格式，则继续以下两步；

（4）记录XMLCallback回调函数中的建模过程的历史记录，并插入到历史流管理器中（如果特征是布尔操作体，则执行布尔操作后再插入）；

（5）为读入的特征模型依次做好id编号，在特征导航器界面中实时显示所创建模型的创建记录。同时，在视图中对创建的模型进行渲染和显示。

4.7 本章小结

本章详细介绍了微波管参数化建模软件平台的设计与实现流程，包括图形用户界面、基于特征的参数化建模模块、特征文件的读写操作以及碰撞检测模块。首先，提出了软件平台的系统框架，简单概述了各个模块的功能和耦合关系。然后，详细讲解了软件平台的图形用户界面的视图和部分实现。之后着重描述了特征建模模块的设计与实现过程，此模块基于HOOPS和ACIS设计和实现了模型、视图和操作器的建模架构，并在此架构之上实现了创建基本几何体和复杂曲线/曲面造型的功能、捕捉点功能以及特征体和特征历史流的管理功能。之后，阐述了碰撞检测算法在所搭建的微波管参数化建模软件平台中的应用，说明了嵌入的碰撞检测求解器模块的程序执行流程以及部分功能实例。最后，软件平台实现了特征文件的读写功能，支持对xml文件和sat模型文件格式的存储和读取。

第五章 微波管仿真环境下碰撞检测算法的应用实例

碰撞检测求解器模块是微波管参数化建模软件平台的主要组成部分，在前文中已经简要介绍了该模块的功能，接下来，本章将详细介绍在微波管的仿真环境下碰撞检测算法的应用实例。

5.1 概述

通过软件平台中的布尔交运算操作，可以判断两个模型是否存在相交和干涉区域，但对于微波管的设计和仿真，仅知道是否干涉远远不够，还需要更多的干涉信息。由于几何模型相交的任意性，获取布尔交运算结果体的数据相当困难，要使微波管仿真系统中零部件的设计和装配过程具备真实性并保证精确度，必须要对场景中的零部件进行干涉检查和间隙分析，判断微波管模型的组件及其零部件之间是否存在碰撞现象，并计算部件之间的分离距离以优化仿真性能。在微波管CAD的仿真环境中，本文的碰撞检测技术的应用主要是干涉检查和间隙分析。

本章应用实例的硬件环境是Intel(R) Xeon(R) CPU E5-2697 v4 @ 2.30GHz的处理器和8GB的RAM内存；软件环境是Windows 10系统，Microsoft Visual Studio 2013开发平台，采用ACIS、HOOPS、微软基础类库MFC和C++语言开发。此外，上一章基于ACIS和HOOPS引擎搭建的微波管参数化建模软件平台提供了可视化的微波管仿真环境，构建的几何模型精确度较高，操作方便，界面友好。

5.2 基本干涉检查应用实例

场景中物体之间的干涉现象是指不同物体在同一时刻占据同一块空间区域或者不同物体包围的空间有重叠的现象[65]。由于微波管制造尺寸与设计尺寸之间存在偏差，因此在设计过程中不存在干涉现象的零部件，实际生产中也有可能产生因干涉从而无法组装的问题。此外，人工检查微波管零部件间隙的效率和可靠性非常低。基本干涉检查能够快速、直观的识别微波管零部件之间的体-体干涉和面-面干涉的现象，并帮助用户检查和评估微波管装配和设计的合理性。基本干涉检查针对的对象是场景中所选的两个几何体，可以提供发生干涉的几何体之间的干涉体信息，以及发生干涉的面对信息等。本文碰撞检测求解器模块的基本干涉检查功能的操作流程图如下图5-1所示。

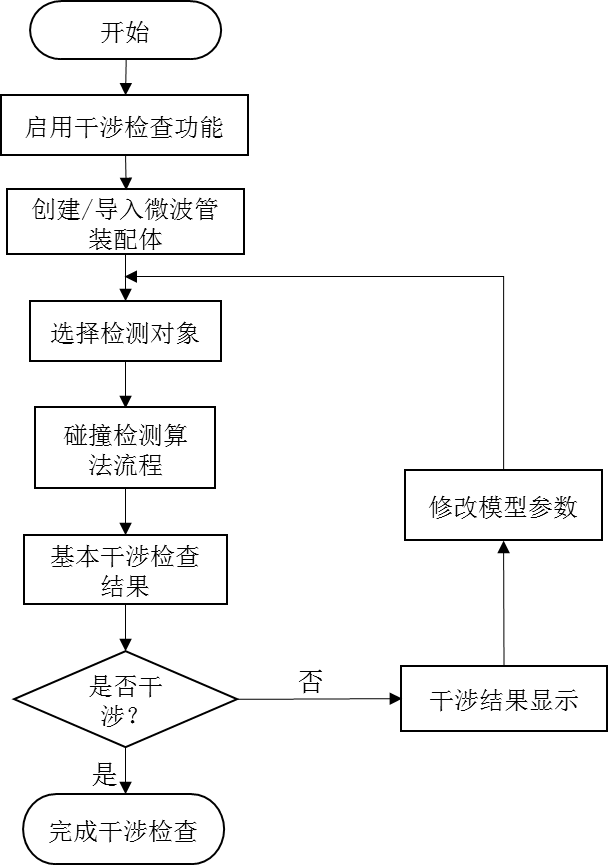


图5-1 基本干涉检查操作流程

（1）工程实例的初始化。首先，注册与运行本软件平台的应用程序并进行软件工程的初始化配置，包括打开软件和创建工作坐标系。然后，启用基本干涉检查功能。如下图5-2为软件的整体图形用户界面，正如在前文4.2节中所描述的。

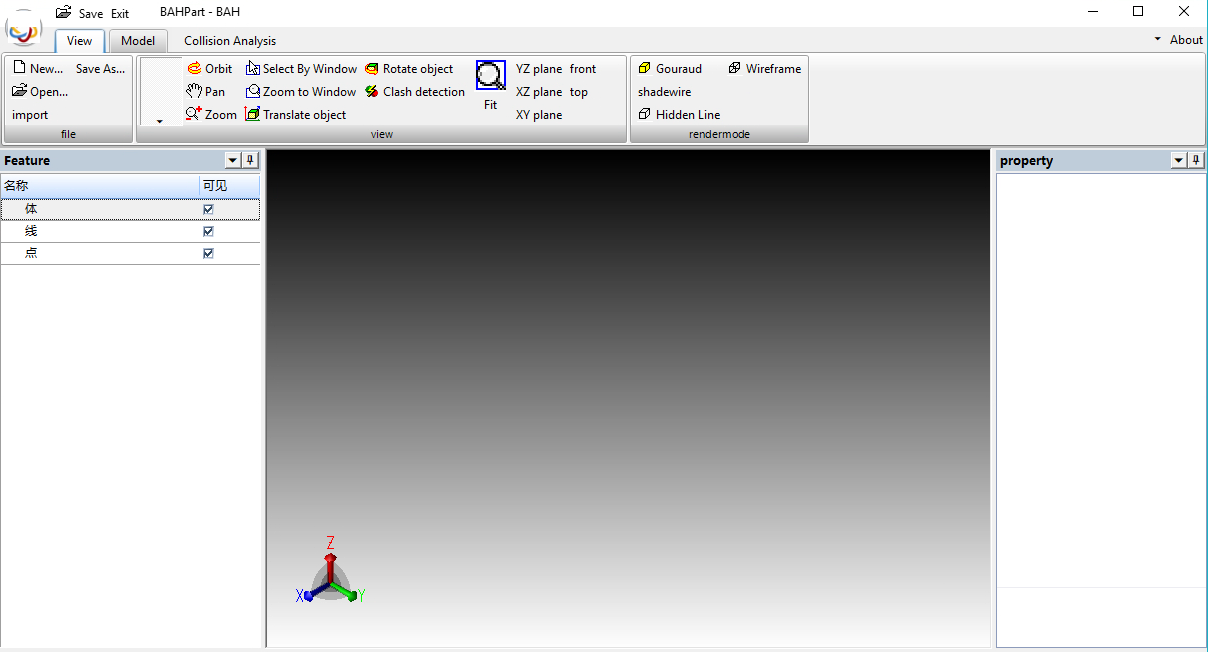


图5-2 软件的图形用户界面

（2）创建模型或导入sat/xml模型文件。向场景中导入模型的方式有两种，一种是在场景中直接创建模型；另一种是直接将微波管组件通过sat模型文件格式导入到工程中。本实例导入如下图5-3所示的行波管的电子枪模型进行基本干涉检查计算，重点关注组成模型的零部件之间的干涉现象。

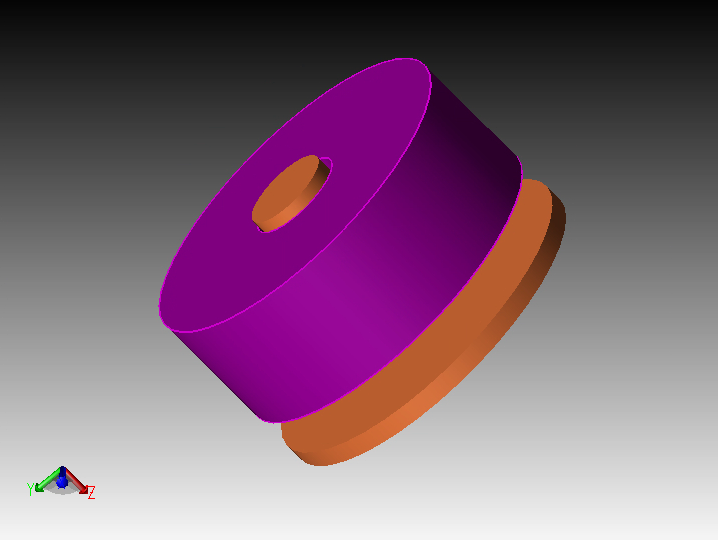


图5-3 行波管的电子枪模型

（3）选择待检测的目标对象。基本干涉检查功能仅涉及组成模型的两个零部件实体，如图5-3所示为在场景中选择零部件对象（淡蓝色高亮的模型为已选中的零部件，紫色高亮的模型为鼠标悬浮待选中的零部件）。设置干涉检查的显示选项。如图5-4所示为干涉检查选项的设置界面，本实例设置“循环显示干涉的面对”选项，在执行碰撞检测算法后，将循环高亮显示两个零部件之间发生干涉的所有面对。

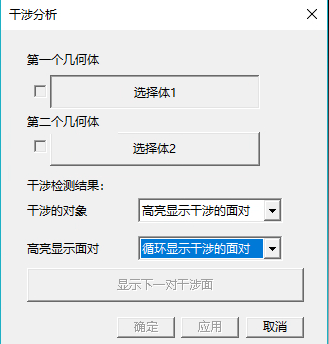


图5-4 干涉检查选项的设置对话框

（5）执行干涉检查，通过碰撞检测求解器模块调用两个实体之间的碰撞检测函数，如果返回的碰撞信息为未发生干涉，则完成干涉检查并退出；若检测出发生干涉，则高亮显示干涉结果。

（6）基本干涉检测结果分析。本实例选择零部件①和零部件②两个零部件作为检查对象，如图5-5所示为基本干涉检查的仿真结果。从图中可以看出，两个零部件存在干涉现象，第一对干涉面对为零部件①的干涉面①和零部件②的干涉面②。其中，对发生干涉的面对采用了黄色高亮显示处理。接下来，需要对模型进行参数调整以避免实际生产装配过程的不匹配。

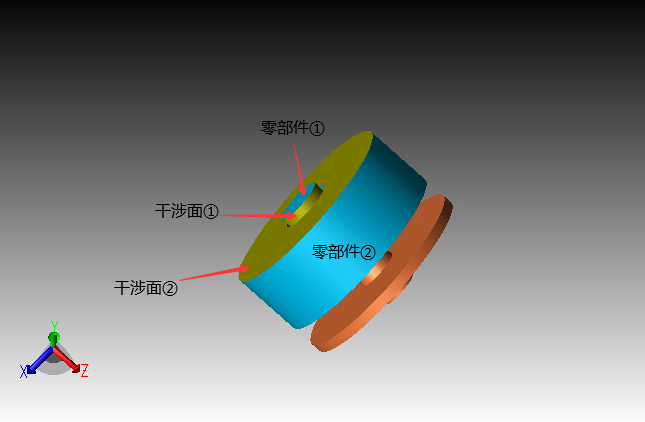


图5-5 干涉检查的仿真结果图

5.3 零部件间隙分析的应用实例

装配间隙是指零部件之间的最接近的距离及相关信息。在微波管模型的设计与仿真过程中，在装配空间的所有零部件之间都有一定的间隙要求，以满足微波管模型的功能、精确度、真实性需求。零部件之间的间隙距离对微波管的组装和仿真有非常大的影响，如果间隙太小，生产、组装和摩擦等误差的积累可能造成部件干涉现象[66]；如果间隙太大，又会大大降低微波管的性能和效率，干扰微波管的仿真分析。因此，对组成微波管的零部件之间进行间隙分析可优化设计参数，减小返工风险和生产成本，进而改善微波管的仿真效率和性能。根据零部件的空间相对位置，可以分为如下五种干涉类型[67]：

（1）不干涉：两个零部件间的距离大于所设置的间隙距离；

（2）接触干涉：两个零部件恰好相互接触但是没有发生穿透或干涉现象；

（3）硬干涉：两个零部件发生干涉和穿透现象且在空间中存在重叠区域；

（4）软干涉：零部件间的最小距离小于设置的间隙距离，但并未接触或干涉；

（5）包含干涉：一个零部件完全将另一个零部件包含在其所占的实体空间之内的情况。对微波管的零部件之间执行装配间隙分析操作的流程如下图5-6所示。

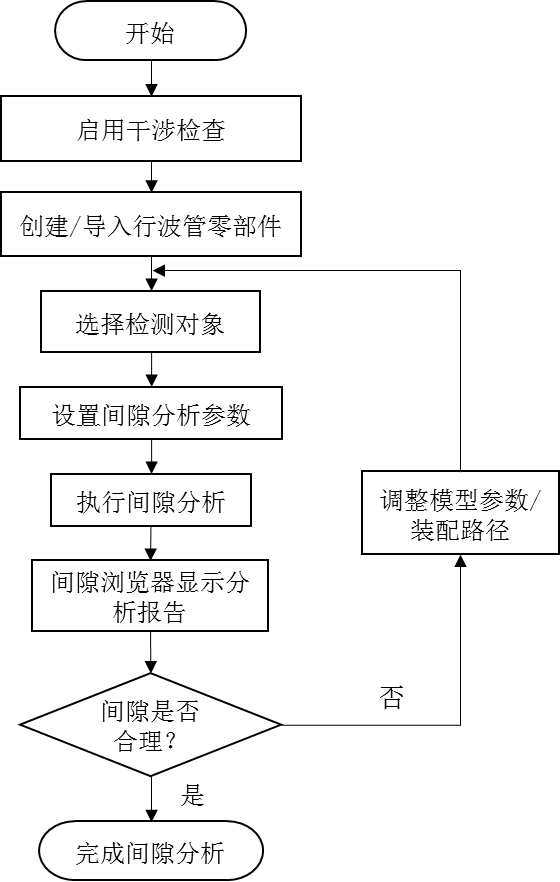


图5-6 零部件的间隙分析流程图

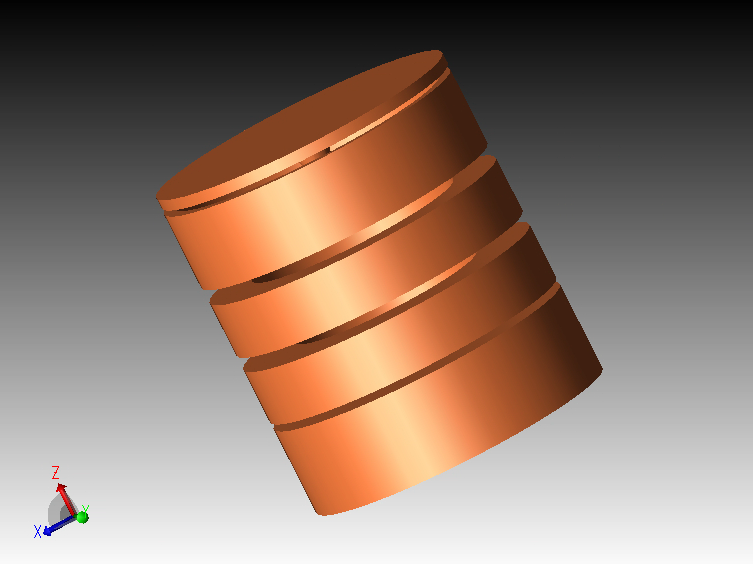


图5-7 行波管的降压收集极模型

（1）首先，操作流程的前两步与上一节基本干涉检查操作流程的前两步类似，在此不再赘述。在本实例中，导入了行波管的多级降压收集极模型如图5-7所示。

（2）选择要执行间隙分析的检测对象。如下图5-8所示为间隙分析的设置界面，在“要分析的对象”分类下选择对“一个集合中”的所有零部件间执行间隙分析，选中“选定的对象”后，在场景中使用矩形框选择零部件作为一个零部件集合。

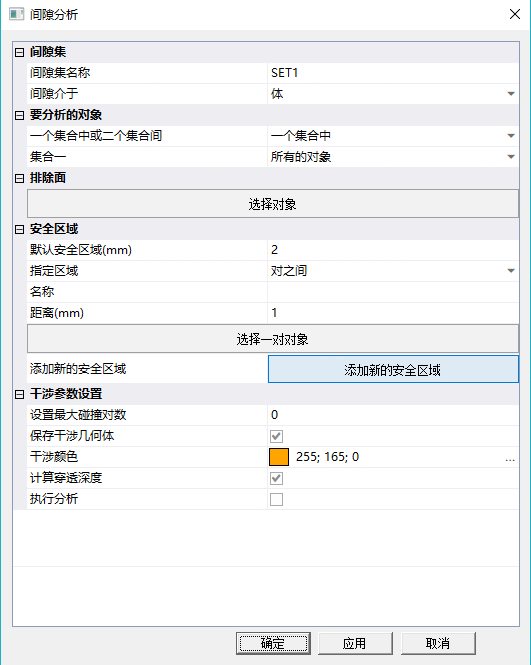


图5-8 间隙分析设置界面

（3）设置间隙分析的选项和参数，如下表5-1所示为间隙分析参数表。

表5-1 间隙分析参数说明表

|  |  |
| --- | --- |
| 参数名称 | 参数说明 |
| 排除的面 | 选择指定的面，设置执行分析时忽略物体之间在指定面上的干涉 |
| 默认安全区域 | 满足零部件之间误差需求的默认间隙距离 |
| 距离 | 设置选择对象之间的合理间隙距离 |
| 最大碰撞对数 | 设置允许检测到的最大碰撞对个数 |
| 保存干涉几何体 | 确定是否保存物体之间发生干涉的部分 |
| 干涉颜色 | 设置检测出干涉的体、面的突出显示颜色 |
| 计算穿刺深度 | 设置是否计算已碰撞物体之间的穿透深度 |

（4）执行间隙分析的碰撞检测算法流程，在执行间隙分析阶段，应用程序将调用碰撞检测求解器模块中封装好的碰撞检测算法函数，对所选的零部件集合实施碰撞检测并计算物体之间的最接近距离。

（5）间隙结果分析和可视化评审

间隙分析检测结果为下图5-9所示的间隙浏览器界面，其中包含了干涉信息、排除的体或面信息以及所选中的检测对象集合信息。其中，干涉信息主要包括所选中的实体、与实体干涉的体、干涉的类型、干涉点位置、实体间的最接近距离、所设置的合理间隙以及标识符。若物体之间存在干涉现象或最近距离不符合所设置的间隙距离，需要对模型的参数进行调整，接下来将详细介绍微波管的零部件之间可能检测到的几种干涉情况。

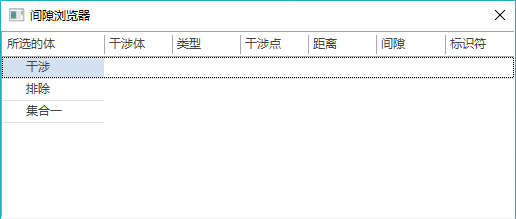


图5-9 间隙浏览器显示检测报告

**5.3.1** 软干涉检查结果

对图5-7中导入的行波管降压收集极模型执行间隙分析，设置“默认安全区域”为2mm，未设置“排除的面”，未设置“最大碰撞对数”，设置“保存干涉几何体”为是，设置的“干涉颜色”为红色。从图5-10中可以看出，间隙分析的集合一中的零部件实体包括“Cylinder-1”、“Cylinder-2”、“Cylinder-3”、“Cylinder-4”和“Cylinder-5”。由于未设置排除面，因此排除节点下没有显示分支。执行碰撞检测算法之后，发现的干涉位置有四处，干涉类型均为“软干涉”，启用GJK算法得到的间隙距离分别为0.696053mm、1.303947mm、1.405264mm、0.594736mm，这样的间隙符合收集极的安全间隙要求，但是对微波管性能的影响还有待进一步的仿真分析。



图5-10 降压收集极间的软干涉检查结果

**5.3.2** 接触干涉检查结果

组成行波管输入输出窗的零部件相对少，但是设计过程中的性能优化较为复杂，主要原因在于电磁波的各个频带宽度对应的结构尺寸差异比较大，导入如下图5-11所示的行波管输入输出窗模型执行间隙分析。设置对一个集合中的所有零部件分析，设置“默认安全区域”为2mm，未设置“排除的面”，设置“保存干涉几何体”为否，设置的“干涉颜色”为黄色。执行间隙分析后，得到如图5-12所示的间隙分析报告，可以看到零部件之间存在接触干涉现象，而且发生接触干涉的一对零部件之间的距离为0mm。

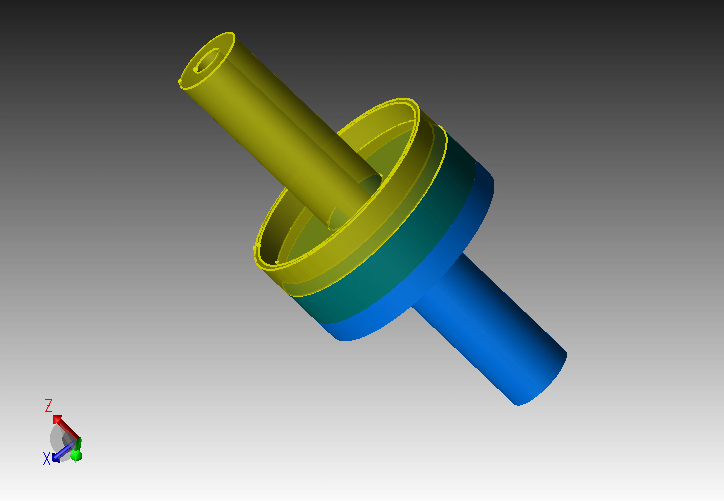


图5-11 行波管的输入输出窗



图5-12 行波管输入输出窗的接触干涉检查结果

**5.3.3** 硬干涉检查结果

对下图5-13所示的行波管周期永磁系统模型执行间隙分析，对一个集合中的所有零部件执行分析，设置“默认安全区域”为1mm，设置“保存干涉几何体”为否，设置“计算穿透深度”为否。执行间隙分析后，得到周期永磁系统的可视化分析报告如图5-14所示，可以看到零部件之间存在硬干涉现象，发生干涉的零部件对有五对。此外，通过三角形之间的精确相交检测得到了模型之间的硬干涉区域，下图5-13中的蓝色线条即为实体之间相交产生的线框。接下来需要对模型进行参数调整以解除零部件的碰撞穿透状态，减小实际生产过程中的返工风险。

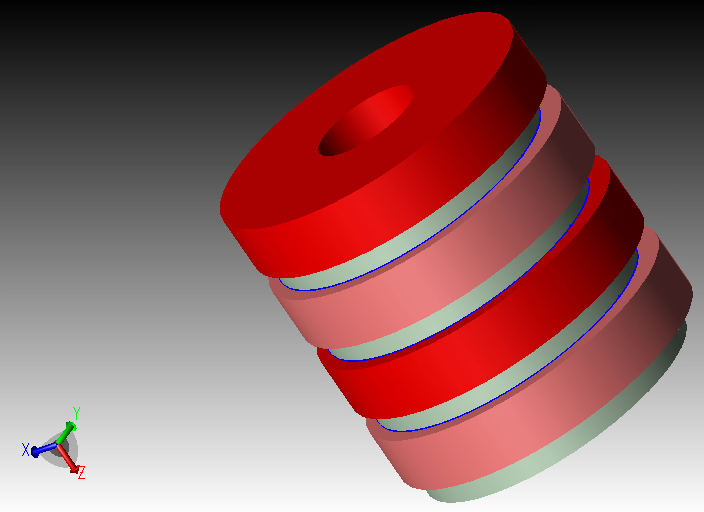


图5-13 行波管的周期永磁系统模型

此外，若将间隙参数中的“保存干涉几何体”选项设置为“是”，那么硬干涉检查的结果中将显示精确的干涉几何体，并隐藏检测对象，如下图5-14所示。

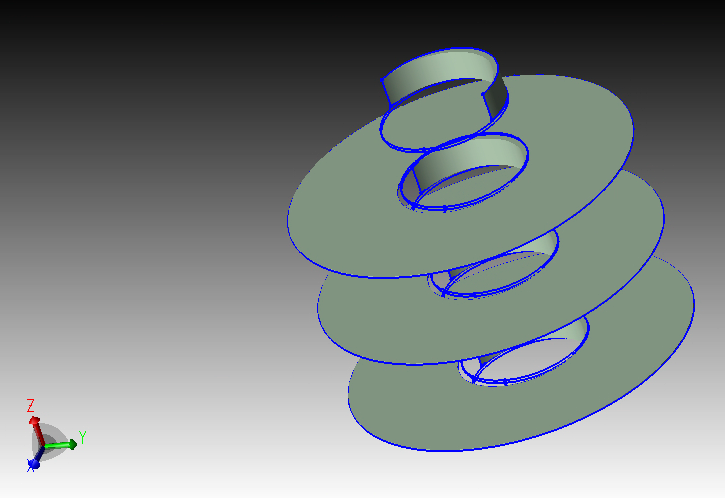


图5-14 周期永磁系统模型零部件之间的干涉区域



图5-15 周期永磁系统的硬干涉检查报告

**5.3.4** 包含干涉检查结果

微波管的同轴窗组件尺寸较小，通常采用圆柱体之间进行布尔差运算的形式建模，在使用参数化建模系统创建行波管同轴窗模型的过程中，笔者创建了如下图5-16所示的两个圆柱体模型。对模型进行间隙分析，设置“默认安全区域”为1mm，设置“保存干涉几何体”为否，设置“计算穿透深度”为否。进行间隙分析之后，得到如图5-17所示的间隙分析报告，可以看到零部件“Cylinder-1”和“Cylinder-2”之间存在包含干涉现象，即存在一个零部件将另一个零部件完全包含在其实体空间内。

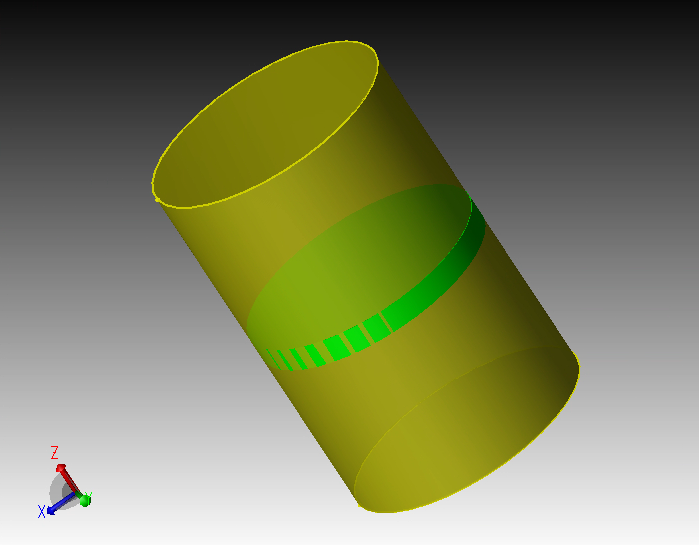


图5-16 行波管同轴窗模型创建过程



图5-17 行波管同轴窗模型的包含干涉检查结果

5.4 本章小结

虚拟仿真环境与真实物理世界中微波管的零部件之间的尺寸差距较大，一个微小的尺寸误差都很可能造成零部件之间的干涉现象或过大的间隙距离，导致微波管的实际生产过程中无法装配或影响微波管仿真的性能。本章首先介绍了应用实例的软硬件环境，然后分析了三维微波管零部件之间的基本干涉检查操作实例以及间隙分析操作的实例和四种类型的干涉结果，验证了在微波管的仿真环境下的碰撞检测算法的实用性和可靠性。

第六章 总结和展望

微波管是一类广泛应用于信息对抗、卫星导航和军事通讯等领域中关键的功率放大器件，工作的可靠性和稳定性在很大的程度上决定了整个雷达或通信设备的性能。通过碰撞检测技术辅助求解分析组成行波管的零部件之间的接触和干涉的情况，如果零部件之间发生碰撞或干涉现象，说明建模和设计过程存在问题，需要对零部件模型进行参数调整，重复上述过程直至微波管的零部件之间的间隙符合设计要求或不再干涉。

本文的目标是研究微波管建模与装配仿真过程中所必须要考虑的碰撞和接触的问题，并对这些问题提出自己的解决方案。主要工作如下：

（1）参阅了大量碰撞检测相关的文献资料，调研了碰撞检测技术的国内外发展现状和研究热点，横向比较了几种典型的包围体算法在构造难度、内存消耗、紧密性、更新计算复杂度等方面的性能。

（2）设计了混合层次包围体算法的结构和算法流程，采用自顶向下的方式构造层次包围体树结构，采用深度优先遍历方法遍历包围体树结构，并在基本图元的相交测试阶段采用区间相交算法判断三角形之间的重叠情况。

（3）研究和分析ACIS几何建模引擎、HOOPS图形渲染套件以及MFC软件开发技术，设计和开发了一款可视化的微波管参数化建模软件平台，实现了图形用户交互界面、视图显示操作、参数化建模、文件读写等功能模块。

（4）设计了几何模型的特征体类的框架，并采用特征体管理器和历史流管理器在整个应用程序全局管理所创建的特征模型和特征操作的历史记录。

（5）研究了xml文件格式，设计了一种特征文件存储的规范，支持模型的几何参数的存储以及对模型的创建和操作过程的历史流的读写功能。

（6）利用面向接口编程的思想，将碰撞检测算法封装为碰撞检测模块。该模块根据从ACIS引擎获取的几何属性信息构造AABB-OBB混合层次包围体树，采用深度优先遍历的方式递归检测包围体树中零部件之间的碰撞情况，最终返回发生碰撞体、面以及最近距离参数等碰撞信息。

由于本人的能力有限和疫情等原因导致的科研时间的紧缺，本文的工作还有很多待完善和解决之处，具体不足之处和对未来工作的展望在于以下几个方面：

1）本论文未完成模拟零部件在碰撞后的物理响应的运动仿真分析。运动仿真需要运用物理学中的力学理论，包括理论力学、分析力学和弹性力学等。以往的研究经常将碰撞检测和碰撞响应作为两个独立的研究领域，但要实现微波管仿真过程的虚拟装配，必须要结合物体的动力学分析和动态干涉检查。因此，后续的工作可围绕连续碰撞检测算法和碰撞响应算法方面展开。

2）本文在AABB和OBB包围体结构的基础上设计了混合BVH树结构，基本满足的微波管仿真检测的精度要求，并在实时性和精确性之间达到了一定的平衡。但是在实际的干涉检查和间隙分析应用中，还是存在检测速度较慢的问题，接下来可选用更快速的包围体或其他优化的碰撞检测算法进行研究。

3）此外，如何提高计算分离距离的穿刺距离通常采用的GJK算法的效率，这也是未来值得深入研究的方向。

致 谢

行文至此，已近尾声。春花秋月何时了，往事知多少，从三年前考研进入成电至今，已将近三载。犹记当年考研期间通宵达旦的努力，观看视频学习考研英语时的认真与坚持，思考数学和电磁场专业课难题时的专注，背诵政治习题时的朗朗书声，碎片式的画面一帧接着一帧在我的脑海中回放，幸而上榜，感谢当年一起并肩的同学，虽然渐渐没了通讯，但回忆和感情珍贵。往后，从南京到成都，跨越千里的求学之路，初心是提升能力和弥补高考落榜的遗憾，同时也是想换一个环境弥补略显荒废的大学四年时光。时光荏苒，三年时光如流水般，转瞬即逝。回顾这三年，我成长了很多，也收获了许多，通过健身和跑步使身体强健，通过努力学习在学业上取得不错的绩点，通过刻苦科研取得过老师的认可。

首先，我要感谢我的导师徐立老师。徐老师严谨的科研作风，积极进取的态度和敏锐的思维给我留下了深刻的印象，您对学生的关心和负责使我终身难忘。我要特别感谢赵瑛峰老师，您渊博的学识、谦逊的品德和敏锐的洞察力，为我的科研道路指明了方向，使我受益匪浅。此外，在科研项目和本篇论文撰写过程中，赵老师都给予了我关心和指导，在此，谨向您表示诚挚的感谢。感谢黄桃老师和胡玉禄老师在科研上给予我的指导和帮助。感谢贺老师和廖老师在生活上的照顾。感谢教研室的同窗：宋浩、曹炎颢、杨德云、刘晓东、范家晖、林冠杰、张晓妍、罗广、陈源、王泓涵、朱世龙以及陈斌师弟、赵晓鸽师妹、韦晓晨师妹使我得以在和谐的环境中进行科研工作，和你们在一起的时光真的很开心。我要特别感谢吴兴师兄、邓文凯博士和朱世龙博士，在软件开发和三维建模方面对我的大力帮助。我还要感谢朝夕相处的室友杨鹏同学，曹炎颢同学和郭宏阳博士，谢谢你们的支持和包容。感谢一起健身和跑步的黄杰博士，许星星博士同学。感谢一起旅游和相互鼓励的徐宜亮同学，感谢一起跑步和打台球的罗广同学。

当然，我最应该感谢的是我的父亲和母亲，我能取得今日的知识和能力，离不开他们对我的爱护和默默支持。我还要特别感谢我的女朋友杨静同学的陪伴和支持，给了我前进的动力，希望我们能一直走下去！

最后，感谢各位专家在百忙之中抽出时间批阅我们的论文！

参考文献

1. 陈浩.行波管零件设计加工以及装配的数字化实现方法研究[D]. 成都:电子科技大学,2019.
2. 刘敏玉.行波管电子光学系统仿真计算软件设计与实现[D]. 成都:电子科技大学,2014.
3. 吕骄阳.行波管多物理场协同仿真技术研究[D]. 成都:电子科技大学,2019.
4. 刘卫明,刘恩锦,郑第称,等.基于Pro/E的通用机枪干涉检查与验证[J].兵工自动化,2017,36(02):30-33.
5. 王祎.虚拟现实中碰撞检测关键技术研究[D]. 吉林:吉林大学,2009.
6. 刘丽.虚拟现实中碰撞检测算法研究[D]. 天津:天津大学,2014.
7. M. M. Shinde, A.S. Rao. Static collision detection using visualization tool[J]. Materials Today: Proceedings, 2018, 5(13):27220-27224.
8. E. Larsen, S. Gottschalk, M. C. Lin, et al. Fast distance queries with rectangular swept sphere volumes[C]. IEEE International Conference on Robotics and Automation, 2000. Proceedings. San Francisco, CA, USA, 2000, 3719-3726
9. R. Bridson, R. Fedkiw, J. Anderson. Robust treatment of collisions, contact and friction for cloth animation[J]. ACM Transactions on Graphics (TOG), 2002, 21(3): 594-603.
10. 潘仁宇,孙长乐,熊伟,王海涛.虚拟装配环境中碰撞检测算法的研究综述与展望[J].计算机科学,2016,43(S2):136-139.
11. P. Jiménez, F. Thomas, C. Torras. 3D collision detection: a survey[J]. Computers & Graphics, 2001, 25(2):269-285.
12. D. John, C O'Sullivan. Graceful degradation of collision handling in physically based animation [J]. Computer Graphics Forum, 2000, 19(3):239-247.
13. S. A. Cameron, A study of the clash detection problem in robotics[C]. IEEE International Conference on Robotics & Automation, St. Louis, MO, USA, 1985, 488-493.
14. Redon Stephane, Lin Ming C., M. Dinesh, et al. Fast Continuous Collision Detection for Articulated Models[J]. Journal of Computing and Information Science in Engineering, 2005, 5(2):126-137.
15. Min Tang, Curtis Sean, Yoon Sung-Eui, et al. ICCD: interactive continuous collision detection between deformable models using connectivity-based culling[J]. IEEE transactions on visualization and computer graphics, 2009, 15(4):544-557.
16. Zhendong Wang, Min Tang, Ruofeng Tong, et al. TightCCD: Efficient and Robust Continuous Collision Detection using Tight Error Bounds[J]. Computer Graphics Forum, 2015, 34(7):289-298.
17. Peng Du, Jieyi Zhao, Weijuan Cao, et al. DCCD: Distributed N-Body Rigid Continuous Collision Detection for Large-Scale Virtual Environments[J]. Arabian Journal for Science and Engineering, 2017, 42(8):3141-3147.
18. R. Man, D. Zhou, Q. Zhang. A Survey of Collision Detection[J]. Applied Mechanics and Materials, 2014, 538:360-363.
19. 鲍义东,吴冬梅.自适应细分及优化编码八叉树碰撞检测算法[J].上海交通大学学报,2015,49(08):1114-1122.
20. F. J. Melero, A. Aguilera, F. R. Feito. Fast collision detection between high resolution polygonal models[J]. Computers & Graphics, 2019, 83:97-106.
21. 王发麟,郭宇,廖文和,等. 基于距离场和扫掠剪除算法的线缆碰撞检测技术[J].计算机工程与应用,2017,53(10):27-34+42.
22. Capannini Gabriele, Larsson Thomas. Adaptive Collision Culling for Massive Simulations by a Parallel and Context-Aware Sweep and Prune Algorithm[J]. IEEE transactions on visualization and computer graphics, 2018, 24(7):2064-2077.
23. Binbin Qi, Mingyong Pang. An enhanced sweep and prune algorithm for multi-body continuous collision detection[J]. The Visual Computer, 2019, 35(11):1503-1515.
24. 王嘉,李孔清.碰撞检测算法研究综述[J].电脑知识与技术,2017,13(20):202-205.
25. 曲慧雁. 复杂虚拟环境下的快速碰撞检测技术研究[D]. 吉林:吉林大学,2020.
26. C. Lauterbach, M. Garland, S. Sengupta, et al. Fast BVH Construction on GPUs[J]. Computer Graphics Forum, 2009, 28(2):375-384.
27. J. Pantaleoni, D. Luebke. HLBVH: hierarchical LBVH construction for real-time ray tracing of dynamic geometry[C]. Proceedings of the ACM SIGGRAPH/EUROGRAPHICS Conference on High Performance Graphics 2010, Saarbrücken, Germany, 2010, 25-27.
28. Tero Karras. Maximizing Parallelism in the Construction of BVHs, Octrees, and k-d Trees[P]. High-Performance Graphics, 2012.
29. C. Apetrei. Fast and Simple Agglomerative LBVH Construction[J].Computer Graphics & Visual Computing, 2014, 12:1-4.
30. T. Y. Li, J. S. Chen. Incremental 3D collision detection with hierarchical data structures[J].Virtual reality software and technology, 1999, 98:139-144.
31. Peng Du, Jie Yi Zhao, Wan Bin Pan, et al. GPU Accelerated Real-Time Collision Handling in Virtual Disassembly[J]. Journal of Computer Science and Technology, 2015, 30(3):511-518.
32. X. Wang, M. Tang, M. Dinesh, et al. Efficient BVH-based Collision Detection Scheme with Ordering and Restructuring[J]. Computer Graphics Forum, 2018, 37(2):227-237.
33. 孙敬荣,卢新明.基于混合包围盒与三角形相交的碰撞检测优化算法[J].计算机工程与应用,2018,54(19):198-203.
34. 李玉虎,王宗彦.基于混合层次包围盒碰撞算法的改进[J].华东交通大学学报,2019,36(06):112-118.
35. 唐源皓,侯进,吴婷婷,等.基于质点转换和包围盒的混合碰撞检测算法[J].哈尔滨工程大学学报,2018,39(10):1695-1701.
36. 陈梁.基于UG的行波管数字化装配过程分析[D]. 成都:电子科技大学,2017.
37. 吴敏,黄峻森.3D应用程序开发的强大动力——可视化组件HOOPS[J].CAD/CAM与制造业信息化,2006(07):50-52.
38. 范超.基于HOOPS产品数字建模的三维图形平台研究[D].大连:大连理工大学,2012.
39. 李阳.基于HOOPS三维实体尺寸及公差的交互标注[D].大连:大连理工大学,2015.
40. 王牌.基于ACIS-HOOPS三维电磁仿真中的模型构建和网格剖分研究与实现[D].苏州:苏州大学,2015.
41. 马登富.基于ACIS变形造型系统的关键技术研究[D].广州:广东工业大学,2012.
42. 刘婷婷.基于Google Earth和ACIS的三维可视化关键技术研究与实现[D].济南:山东师范大学,2016.
43. 黄敏. 基于ACIS和HOOPS的快速成形空间布局系统的研究与开发[D].天津:河北工业大学,2007.
44. 王晓荣,王萌,李春贵.基于AABB包围盒的碰撞检测算法的研究[J].计算机工程与科学,2010,32(04):59-61.
45. 熊玉梅. 虚拟环境中物体碰撞检测技术的研究[D].上海:上海大学,2011.
46. 张冲. 基于K-DOPS的快速碰撞检测算法研究[D].阜新:辽宁工程技术大学,2008.
47. 丁佳. 大型复杂场景中快速碰撞检测技术的研究[D].成都:电子科技大学,2007.
48. 姜光焱. 基于包围盒的碰撞检测算法的研究及应用[D].成都:电子科技大学,2012.
49. 于凯. 层次包围盒碰撞检测算法的优化及其并行化[D].镇江:江苏科技大学,2016.
50. 邹益胜,丁国富,许明恒,何邕.实时碰撞检测算法综述[J].计算机应用研究,2008(01):8-12.
51. 水泳. 虚拟现实中连续碰撞检测算法研究[D].合肥:中国科学技术大学,2013.
52. 乔梁. 超大规模动态场景的实时碰撞检测研究[D].成都:电子科技大学,2011.
53. 杜群. 基于蚁群算法的碰撞检测在虚拟装配中的应用[D].北京:华北电力大学,2019.
54. 刘海燕. 数控机床的碰撞检测算法的研究[D].天津:天津大学,2009.
55. Donald Hearn, M. Pauline, W. R. Carithers. 计算机图形学[M]. 北京:电子工业出版社, 2014.
56. 戴杰. 虚拟装配系统中碰撞检测技术的研究与应用[D].大连:大连交通大学,2010.
57. 王怀瑜. 防碰撞系统中体素相交检测算法的研究[D].天津:天津大学,2014.
58. 胡俊超. 面向虚拟仿真的碰撞检测研究与应用[D].杭州:浙江大学,2019.
59. 刘超. 虚拟环境中碰撞检测算法的研究和实现[D].南京:南京航空航天大学,2018.
60. Tomas Möller. A Fast Triangle-Triangle Intersection Test[J]. Journal of Graphics Tools, 1997, 2(2):25-30.
61. Martin Held. ERIT-A Collection of Efficient and Reliable Intersection Tests[J]. Journal of Graphics Tools, 1997, 2(4):25-44.
62. O. Tropp, A. Tal, I. Shimshoni. A fast triangle to triangle intersection test for collision detection[J]. Computer Animation and Virtual Worlds, 2006, 17(5):527-535.
63. 林小彬. 基于混合式包围盒的碰撞检测算法的研究与实现[D].广州:华南理工大学,2015.
64. 车婧媛. 行波管多物理场仿真技术研究[D].成都:电子科技大学,2020.
65. 刘国伟. 虚拟装配系统中模型信息库的构建及干涉检查技术研究[D].北京:北京邮电大学,2009.
66. 柯爱心. 基于Pro/E的装配间隙优化算法研究及软件开发[D].广州:华南理工大学,2014.
67. 朱亮. 汽车覆盖件模具软干涉检查算法研究及系统开发[D].武汉:华中科技大学,2015.

攻读硕士学位期间取得的成果

[1] Quan Nie, Yingfeng Zhao, Bin Li. A Survey of Continuous Collision Detection Algorithm[C]. 2020 2nd International Conference on Information Technology and Computer Application, Guangzhou, 2020, Accepted.