中国科学: 物理学 力学 天文学 2025 年 第 55 卷 第 10 期: 104611
SCIENTIA SINICA Physica, Mechanica & Astronomica physcn.scichina.com
《中国科学》杂志社 SCIENCE CHINA PRESS
论 文

自适应笛卡儿网格并行生成加速优化方法

杨子璇[1], 齐昕宇[1], 王镇明[3], 田琳琳[2], 刘剑明[4], 赵宁[1,2*]

1. 南京航空航天大学, 航空航天结构力学及控制全国重点实验室, 南京 210016
2. 南京航空航天大学, 江苏省风力机设计高技术研究重点实验室, 南京 210016
3. 南京航空航天大学, 飞行器数学建模与高性能计算工信部重点实验室, 南京 210016
4. 江苏师范大学数学与统计学院, 徐州 221116
*联系人, E-mail: zhaoam@nuaa.edu.cn

收稿日期: 2024-06-04; 接受日期: 2025-01-06; 网络出版日期: 2025-06-13

摘要 计算网格的生成效率与质量直接决定了数值模拟结果的优劣, 笛卡儿网格因具备自适应适配性好、自动化程度高、复杂外形适应强等特点而受到广泛关注. 但对于复杂外形, 超大规模网格量使得网格生成效率难以令人满意, 因而有必要发展通用性好、效率高、可扩展性强的三维笛卡儿网格并行生成技术. 本文基于KD树(K-Dimensional Tree)提出了一种三维笛卡儿网格并行生成优化方法. 首先, 基于三维绕流问题中复杂外形的几何信息, 以表面三角形集合作为输入生成三角单元包围盒来构建KD树. 其次, 基于所构建的KD树发展了壁面距离计算和相交单元判断加速优化算法, 并通过在KD树中对投影点查找来加速内外单元判断. 最后, 以圆球、立方体、M6机翼以及Su27等外形为例, 验证本文自适应笛卡儿网格并行生成算法的有效性. 数值结果表明, 本文提出的相交判断和壁面距离计算优化方法相较于传统的遍历方法效率可以提升两个量级, 而内外单元判断效率可以提升三个量级, 充分验证了本文方法能够快速、鲁棒、自动地生成流场计算所需的自适应笛卡儿网格.

关键词 笛卡儿网格, 自适应加密, 并行生成, KD树, 复杂外形

PACS: 47.11.-j, 47.85.Gj, 87.55.de

1 引言

在数值模拟过程中, 高效且优质的计算网格是数值模拟的先决条件, 将会对计算结果的保真度产生直接影响. 2014年, NASA发布的《2030 CFD愿景》技术报告中指出: 自动化是未来CFD (Computational Fluid Dynamics)技术的一个重要发展方向[1]. 但是, 据不完全统计, 以网格生成为主的前处理过程占据整个CFD任务周期的60%–70%时间, 需要大量的人工干预或严重依赖于工程师的经验[2], 严重制约着CFD的自动计算. 正如商业软件Pointwise CEO Chawner等人在文献[3]中所言: 网格生成是CFD过程中的“首要瓶颈(Principal Bottleneck)”和“主要成本(Dominant Cost)”, 以至于大多数CFD分析过程变得“繁重(Onerous)”. 因此, 发展高效、鲁棒、自动的网格生成技术对实现CFD技术的自动计算具有重要意义.

引用格式: 杨子璇, 齐昕宇, 王镇明, 等. 自适应笛卡儿网格并行生成加速优化方法. 中国科学: 物理学 力学 天文学, 2025, 55: 104611
Yang Z X, Qi X Y, Wang Z M, et al. An accelerated optimization method for parallel generation of adaptive Cartesian grids (in Chinese). Sci Sin-Phys Mech Astron, 2025, 55: 104611, doi: 10.1360/SSPMA-2024-0211

相比于结构网格或非结构网格，笛卡儿网格具备自动化程度高、复杂外形适应好、网格生成鲁棒性强等优良特性，是降低CFD计算过程中人力成本的有效手段之一，因而成为快速自动计算的研究热点[4,5]. 同时，为了避免笛卡儿网格各向同性带来的“网格灾难”问题，往往需结合自适应技术来减少计算所需的网格量. 一般地，自适应笛卡儿网格的生成流程主要如下: (1) 输入STL (STereo Lithography)格式的外形文件并生成背景网格; (2) 根据背景网格与外形的几何关系对笛卡儿网格的类型进行判断并分类; (3) 计算网格到壁面的距离; (4) 对相交单元进行几何加密以及对外形附近区域进行光顺加密. 在整个网格生成流程中, 步骤2–4对于笛卡儿网格生成的效率和质量至关重要.

在步骤2中进行网格类型判断时, 计算域中的网格单元可以分为三种类型: 相交单元、内部单元和流场单元[6]. 由于空间笛卡儿网格单元都是标准的AABB (Axis-Aligbed Bounding Box), 所以相交单元的确定可以归结为AABB与外形表面三角形的相交判定. Voorhies[7]最早提出了该算法，之后Akenine-Möller[8]使用分离轴定理对算法进行了改进，进一步提升了相交判定效率. 同时，步骤2中，在确定了相交的网格单元之后，需要对其他网格进行内部单元和流场单元的判断，内部的网格为后续浸入边界法的实现提供基础[9–11]，外部的网格进行常规的流场模拟. 此过程中, 射线法[12]是处理复杂几何形状内外判断的常用技术, 简便直观且易于实现. 因此, 诸多学者将其用于笛卡儿网格生成研究. 然而, 对于射线与几何表面相切的情况下，射线法容易判断出错，出现需要额外验证的情况[13]. Aftosmis等人[14]提出了染色算法来进行非相交单元的内外判断，需要用到相邻单元的信息来进行判断[15,16]. Mittal等人[17]提出使用表面法向量与单元到壁面的方向向量来确定内外的方法. 李雪亮等人[18]提出了内积测试(IPT)的方法，根据内积的符号来进行内外判断，应用于复杂外形也可以很好地判断网格的类型. 另一方面，笛卡儿生成过程中的另一个关键技术是步骤3中的壁面距离计算，将直接影响后续的缓冲区加密、虚拟单元法物面边界处理[19,20]、湍流模型的耦合[21]等. 目前，壁面距离的计算方式主要分为求解偏微分方程和采用几何方法直接计算[22]两种. 第一类方法，将最小距离转换为关于波传播问题的偏微分方程数值求解[23]，需要额外的计算花费，壁面距离计算精度受数值离散精度的影响，且对于复杂外形适应性较差. 第二类方法是根据空间点与物面离散网格的几何关系计算距离，是一种基于比较搜索的直接计算方法. 对此，诸多学者基于笛卡儿网格开展了相应的研究[24–27], 在此不再赘述.

在自适应笛卡儿网格生成过程中，关键在于准确确定相关的物面单元. 常用的方法是将物面网格按顺序存入, 并通过遍历搜索确定相关单元. 然而, 这种方法的算法复杂度较高，取决于空间网格数目和物面单元数目. 例如, 设空间网格数目为N, 物面三角形单元数目为M, 在不使用优化算法的情况下, 网格单元的相交、内外判断及壁面距离计算的时间复杂度为$O(M\cdot N)$. 同时，在保证模拟精度相同的前提下，笛卡儿网格的各向同性使得其网格数量大幅多于贴体网格[28]. 尽管可以通过大规模并行技术[29–31]来缓解这一问题, 但在几何形状复杂或网格规模庞大时, 传统遍历方法依然会带来严重的计算负担，严重降低计算网格的生成效率. 对此, 为快速定位到相应的壁面网格, Boger[32]提出采用ADT (Alternating Digital Tree)二叉树数据结构存储物面网格来代替传统的顺序存储，大幅提高计算效率. 但对于复杂外形, ADT的平衡性下降, 依然会对计算效率产生一定的影响. 为此，改进版本的KD树[33](K-Dimensional Tree)理论被提出来用于构造平衡性更优的二叉树数据结构，可以实现效率的进一步提升. 相比而言，KD树是一种易于实现点查找和距离查找的数据存储结构[34]，其在同等条件下可将几何外形表面单元存储为一深度为$\log_2 M$的平衡二叉树，同样情况下的时间复杂度降为$O\left(N\cdot\log_2 M\right)$[35]. 因此，许多学者利用KD树分别对物面相交判定[4,36–38]及壁面距离计算[22,27]进行了加速，取得了显著的效果. 然而，这两步骤间存在较多的重复操作和计算，可以进行更进一步的整合优化. 而对于使用射线法来判断网格类型的情况，射线法容易存在特殊情况判断出错的问题[13]. 此外, 在进行内外单元的判断时, 染色法的实现效率要高于射线法[16,39]，但与射线法仅需网格中点坐标即可完成判断不同，染色法需要更多的辅助信息，因而难以在并行框架下有效实施.

针对上述问题，有必要发展一种更为鲁棒且便捷快速的网格生成优化算法，进而形成统一的自适应笛卡儿网格生成体系. 对此，本文结合KD树的优良特性，发展了一种多核并行框架下的自适应笛卡儿网格自动

生成加速方法, 具体如下: 首先, 使用KD树存储物面网格; 其次, 将壁面距离计算与网格相交判断合并为一个统一的问题, 减少该过程中的冗余计算, 并利用KD树进行距离计算和单元查找以提升计算效率; 最后, 在KD树中通过坐标轴投影点进行点查找, 加快了内外判断的速度. 同时, 投影点的多选策略避免了射线法中可能出现的拐点或边界穿越导致的判断错误, 从而提高了算法的鲁棒性. 基于上述改进, 形成了一种三维自适应笛卡儿网格的并行生成方法体系, 实现了自适应笛卡儿网格的快速、鲁棒和自动生成.

2　自适应笛卡儿网格生成流程

自适应笛卡儿网格的几何自适应方法根据几何外形的输入形式分为二维和三维情形, 其中二维外形由线段集合构成, 三维外形由三角形集合构成. 由于二维的网格生成较为简单, 本文主要阐述三维适应笛卡儿网格的几何自适应方法. 如图1(a)所示, 自适应笛卡儿网格生成流程一般如下:

步骤1: 输入STL格式的外形数据文件;

步骤2: 根据外形尺寸设定相应的计算域和网格步长生成初始背景网格;

步骤3: 通过基于轴对齐边框的快速相交测试[12]判断笛卡儿网格与物面的相交关系;

步骤4: 使用射线法[35]对笛卡儿网格进行内外判定;

步骤5: 使用几何方法计算壁面距离;

步骤6: 基于2:1原则对相交单元及临近区域进行光顺加密;

步骤7: 重复步骤3到步骤6直至网格大小达到预设的最小尺寸.

以DLR-F6模型为例, 整个流程中网格的变化如图1(b)所示, 各个步骤所占用的时间如图2所示. 可以看出, 在网格生成的全过程中, 相交判断、内外判断、距离计算的时间占比分别为4.7%, 2.0%, 55.5%, 共约60%的时间用来进行网格类型的确定和壁面距离的计算. 因此, 此部分的计算时间直接决定了自适应笛卡儿网格的生成效率. 对此, 本文将在下一节分别从相交判断、内外判断以及距离计算三个方面进行优化, 以减少自适应笛卡儿网格的生成时间.

3　自适应笛卡儿网格生成优化方法

传统方法按照读入顺序来存储物面单元, 是制约生成效率的关键. 为提高网格生成效率, 本文引入了平衡k维二叉树, 即使用KD树来存储物面网格, 代替传统的顺序存储, 发展了一种三维笛卡儿网格并行生成优化方法, 具体如图3所示. 下面将详细介绍本文方法.

3.1　KD树算法概述

KD树作为一种特殊的二叉树数据存储结构, 可以实现数据的快捷访问和调用, 应用于几何搜索、几何相交问题时可以显著提高计算效率. 设空间网格数目为N, 物面三角形单元数目为M, 在不采用任何优化算法的情况下, 定位相关三角单元的时间复杂度

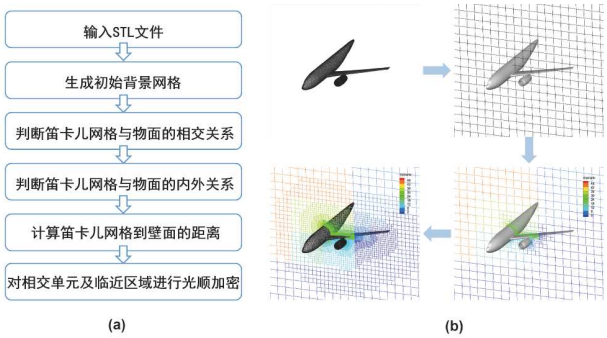

图 1　(网络版彩图)自适应笛卡儿网格生成流程. (a) 流程图. (b) DLR-F6示例

Figure 1　(Color online) Adaptive Cartesian grid generation process (a) Process chart. (b) DLR-F6 example.

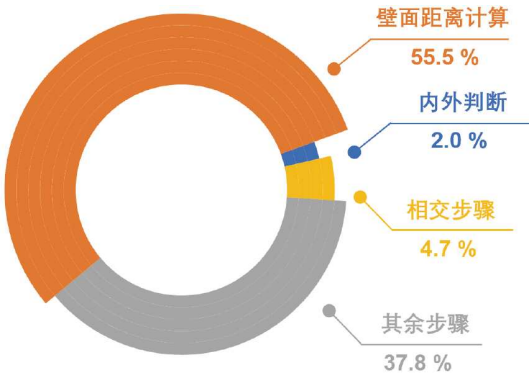

图 2　(网络版彩图)自适应笛卡儿网格生成过程中各步骤时间占比(DLR-F6模型)

Figure 2　(Color online) The time proportion of each step in the adaptive Cartesian grid generation process (DLR-F6 model).

图 3　(网络版彩图)自适应笛卡儿网格生成优化前(a)和优化后(b)流程比较

Figure 3　(Color online) Comparison of processes before (a) and after (b) optimization for adaptive Cartesian grid generation.

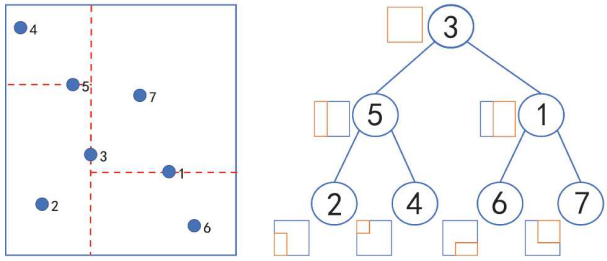

图 4　(网络版彩图) KD树示意图

Figure 4　(Color online) Illustration of KD tree.

```
FUNC KDTree(BoxList, depth, PreNode)
Split ← depth mod 3
Range ← [min(BoxList),max(BoxList)]
Node.data ← FindMedian(BoxList,Split)
Parent ← PreNode
Left ← KDTree(Boxes in BoxList before median,depth+1,Node)
Right ← KDTree(Boxes in BoxList after median,depth+1,Node)
return Node
```

图 5　KD树构建流程

Figure 5　The process of KD tree construction.

为$O(M\cdot N)$. 而使用KD树结构后可以将表面元素存储为一深度为$\log_2 M$的平衡二叉树, 进而将时间复杂度降为$O(N\cdot\log_2 M)$, 可以大幅度提升搜索效率[35]. 首先, 读入物面单元的STL几何外形文件, 其中KD树的各叉树节点对应各物面三角单元, 且各节点代表实际查找区域中的一片方盒区域[36]. 以二维数据为例, 给定随机分布的7个点, 根据这7个点的几何位置来创建KD树, 其KD树结构如图4所示. 本文构造的KD树数据结构包括:

Node-data, 数据集中的某个数据点, 是k维矢量;

Range, 该节点所代表的空间范围;

Split, 垂直于分割超平面的方向轴序号;

Left, 由位于该节点分割超平面左子空间内所有数据点所构成的KD树;

Right, 由位于该节点分割超平面右子空间内所有数据点所构成的KD树;

Parent, 父节点.

其次, 根据物面离散三角网格生成三角单元包围盒来构建KD树, 其算法如图5所示, 下面将介绍主要流程.

步骤1: 初始化分割轴.

步骤2: 得到数据集的空间范围, 初始化节点的空间范围.

步骤3: 确定节点, 对当前数据按分割轴维度进行检索, 找到中位数数据, 并将其放入到当前节点上.

步骤4: 划分双支——(1) 划分左支, 在当前分割轴维度, 所有小于中位数的值划分到左支中; (2) 划分右支, 在当前分割轴维度, 所有大于等于中位数的值划分到右支中.

步骤5: 根据左右支的数据, 更新左右支的空间范围.

步骤6: 按照节点的层数, 更新左右支的分割轴.

步骤7: 确定子节点——(1) 确定左节点, 在左支的数据中进行步骤3到6; (2) 确定右节点, 在右支的数据中进行步骤3到6.

3.2　KD树加速相交判断及壁面距离计算

笛卡儿网格与物面的相交判断是生成笛卡儿网格的关键所在. 而根据几何性质, 该问题可归结为正方体集合与物面三角形集合相交关系的求解. 分离轴理论[8]是该过程的常用方法, 其在进行几何判断时需要将立方体和空间三角形分别在特定的13条轴上进行投影. 若两者的投影结果均发生重叠, 则说明立方体与三角形相交; 反之, 则说明两者并不相交.

然而, 对于与网格相距较远的三角单元, 可以通过几何距离进行更为简单的判断. 因此, 可以将相交判断与壁面距离的计算合并为一个过程. 在笛卡儿网格对三角单元进行最短距离的比较搜索时, 当距离小于某一尺度时, 才使用分离轴理论进行更为精确的相交判断, 从而进一步减少冗余的计算开销.

在上述过程中, 使用KD树代替遍历搜索来查询相关的三角单元, 可以将时间复杂度由为$O(M\cdot N)$降低至$O(N\cdot\log_2 M)$. 对此, 算法设计如图6所示, 具体流程如下:

(1) 针对要判断的笛卡儿网格单元, 选取相应的物面三角单元. 其中, 初始笛卡儿网格单元进行判断时, 随机选取一物面三角单元, 如根节点对应的三角单元. 其余由加密产生的新笛卡儿网格单元进行判断时, 选取上一级粗笛卡儿网格单元距离最近的物面三角单元K_{nearest}.

(2) 计算该单元到选取的三角单元距离L.

(3) 进入KD树开始查找.

(4) 以该网格单元中心坐标为原点, 向x, y, z三个维度的两个方向均伸长L长度, 形成相应的网格单元包围盒.

(5) 比较该单元包围盒与当前节点存储的空间范围是否有重合. 若没有重合区域, 则该节点停止向下查找, 执行回溯操作; 若有重合, 则比较L与H的大小, 若$L<H$, 则使用基于分离轴理论的网格相交判定算法判断是否与物面相交. 其中, H是一与网格尺寸有关的量, 可取H为$2h+\max h$, 其中h为当前网格单元尺寸, $\max h$为所有物面三角单元包围盒的最大尺寸, 以避免不必要的相交判断, 提高查找速度. 若判断为相交, 则将该网格单元标记为相交单元.

(6) 若步骤(5)中有重合且当前节点不为叶子单元, 则进入子树, 计算单元与当前节点存储的物面三角单元的距离new$h$. 若new$h<L$, 则更新L和K_{nearest}.

(7) 重复步骤(3)至步骤(6)直到KD树查找结束.

若是相交单元, 则查找结束时, 当前单元标记为相交单元. 同时, 对于所有笛卡儿网格单元, 查找结束时均能得到单元与物面的最短距离L和对应的最近物面三角单元K_{nearest}.

```
FUNC RangeSearch(K, L, Node)
index ← Node.index
newh ← GetDistance(cell,index)
if newh < L then
  | L ← newh  K ← index
end
if newh < H then
  | cell.flag ← IntersectJudge(cell,index)
end
if 当前节点不为叶子单元 then
  | cellBox ← [cell.midpoint-L,cell.midpoint+L]
  | if Left.Range与cellBox有重合 then
  |   | RangeSearch(K, L, LeftNode)
  | end
  | if Right.Range与cellBox有重合 then
  |   | RangeSearch(K, L, RightNode)
  | end
end
```

图 6　相交判断及壁面距离计算优化算法流程

Figure 6　Process of the optimization algorithm for intersection detection and wall distance calculation.

3.3　KD树加速内外判断

由于相交单元已在上一步骤中确定, 因此剩余单元与物面的相对位置关系仅有内外两种. 当前的内外判定可以简化为立方体单元中心与某点连线与表面三角形集合的相交关系. 传统射线法使用物体内部点作为参考点进行内外判断, 但网格单元中点与参考点的连线穿过的物面三角单元需要准确定位. 对此, 最直观的方法是遍历所有物面单元, 进而判断是否被线段穿过, 但该方法极为耗时, 其时间复杂度为$O(M\cdot N)$.

为解决上述问题, 本文提出了一种改进方法. 首先, 选择网格单元的中点, 并将其沿坐标方向投影到流场边界得到一个外部点, 作为判断的参考点. 然后, 连接参考点与网格中点构成一条平行于坐标轴方向的线段, 这条线段在投影面上可以视为一个二维点. 此时, 线段穿过的三角单元范围可大致确定. 接下来, 结合KD树对二维投影点进行查找, 以提高搜索的效率. 最后, 使用重心坐标法[40]来判断线段是否穿过查找到的三角单元. 此外, 可以沿不同坐标轴的正负方向进行投影, 当重心坐标法检测到线段穿过三角单元的边界或顶点时, 自动切换方向投影, 以避免由于重复计数[13]导致的判断错误. 算法设计如图7所示, 具体流程如下:

(1) 选择投影方向, 如x轴正方向进入KD树开始查找.

(2) 将需要判断的网格单元中心点O坐标向投影方向投影到流场边界, 得到一个边界投影点$O1$和一个二维点$O2$. 对于投影方向为x轴正方向的情况, $O1$坐标为($x$pos, y, z), $O2$坐标为(y, z). 其中$x$pos为流场在x轴正方向的边界位置, (x, y, z)为网格中点坐标, 投影过程如图8所示.

(3) 使用二维点$O2$在KD树中查找. 首先判断根节点存储的整体空间范围在投影方向上是否包含$O2$, 若不包含, 则直接判断为流场单元, 判断结束. 若包含$O2$,

```
FUNC PointSearch(direction, Node)
index ← Node.index
O1,O2 ← cell.midpoint向投影方向投影
if O2 ∈ NodeBox then
    LinearIntersectTriangleTest(cell.midpoint,O1,index)
    if 线段从拐点或者边界穿过 then
        return //更换投影方向
    end
    if 线段穿过该三角单元 then
        count++
    end
end
if 当前节点不为叶子单元 then
    if O2 ∈ Left.Range then
        PointSearch(direction, LeftNode)
    end
    if O2 ∈ Right.Range then
        PointSearch(direction, RightNode)
    end
end
```

图 7　内外判断优化算法流程

Figure 7　Process of the optimization algorithm for determining interior and exterior cells.

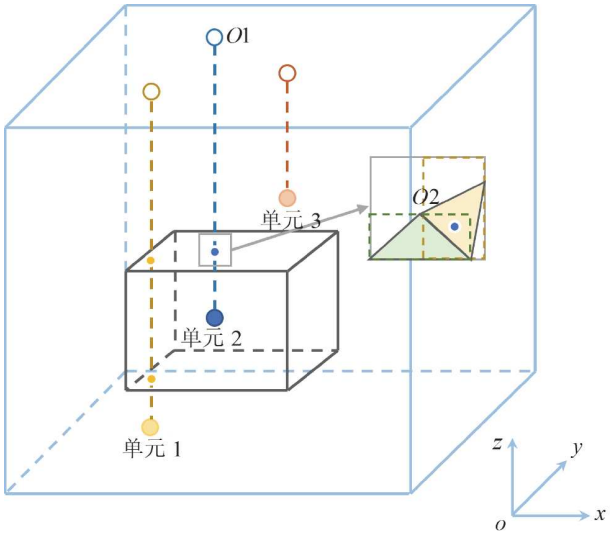

图 8　(网络版彩图)投影示意图

Figure 8　(Color online) Illustration of projection.

则沿着树向下查找，在判断点是否落在左右子树区域范围的同时，判断点是否落在节点存储的三角单元包围盒中，落在包围盒的情况下使用重心坐标法判断是否穿过该三角单元并计数.

(4) 在KD树中查找结束后，因流场边界点$O1$为一外部点，若上述线段与三角形集合的相交次数为偶数，则该正方体网格处于物体外部，为流场单元；反之则为内部单元.

(5) 在步骤(3)中，若判断线段是从三角单元的边界或顶点穿过，则此次KD树查找结束，更换投影方向为x，y，z轴正负方向中的其余方向，重复(2)到(4)步以避免错误的内外判断.

4　算例测试

本节对本文发展的使用KD树加速网格类型判断及壁面距离计算方法效率进行了测试和评估. 为了保证测试结果的合理性，本文所有工况均在CPU为Intel (R) Xeon(R) CPU Gold 6248 @ 2.50 GHz的处理器上测试，共48核进行计算. 首先，以圆球模型和立方体模型为对象，测试本文优化算法对简单但特殊的几何外形的正确性及适用性. 其次，以ONERA-M6机翼、带短舱的DLR-F6半机模型、带有进气道及导弹的SU-27模型等复杂外形为对象，验证本文算法对实际工程外形的自适应笛卡儿网格生成能力.

表1列出了五种不同外形的物面三角形单元数量、几何自适应次数以及最终生成的空间笛卡儿网格数目. 基于不同外形的STL文件，图9–图11展示了使用本文方法生成的相应笛卡儿网格. 图示中，flag为2标记为相交网格单元，−1标记为流场网格单元，−2标记为内部网格. 图9展示的简单特殊外形网格生成，验证了基于KD树的内外判断投影算法在处理此类特殊外形时的有效性. 通过图1、图10和图11中的实际工程外形的自适应笛卡儿网格示意图，进一步验证了本文方法在全自动、准确和鲁棒性方面的自适应笛卡儿网格并行生成能力. 特别地，图11(c)展示了生成的自适应网格的并行分区情况，表明了本文方法在几何连续性方面的优越性.

表1展示了未优化、使用ADT方法和KD树优化的网格生成总时间对比，其中T_1和T_2分别表示KD树方法相比于遍历法和ADT方法的加速效率. 结果表明，在相同条件下，本文算法的整体网格生成效率相比于传统遍历法提高了59.5，198.2，48.2，107.7，192.2倍，相比于ADT方法提高了1.8，1.3，3.0，3.1，3.6倍，且几何形状越复杂或网格量越大，速度提升越明显.

为了进一步说明本文3.2–3.3节算法的效率提升效果，表2和表3分别展示了使用KD树加速的相交判断、壁面距离计算及内外判断所需的计算时间. 结果显示，优化后的相交判断和壁面距离计算效率分别提升了

表 1　优化前后网格生成总时间对比(CPU时间/s)

Table 1　Comparison of total mesh generation times before and after optimization (CPU time/s)

算例	三角单元数目	几何自适应次数	空间网格数目(万)	遍历法	ADT方法	KD树方法	T_1	T_2
圆球	3918	8	2638	244	7.5	4.1	59.5	1.8
立方体	28160	7	2708	2200	14.1	11.1	198.2	1.3
ONERA-M6	16280	9	2442	829	50.9	17.2	48.2	3.0
DLR-F6	35532	8	2507	2834	81.5	26.3	107.7	3.1
SU-27	548042	8	2386	38903	737.0	202.4	192.2	3.6

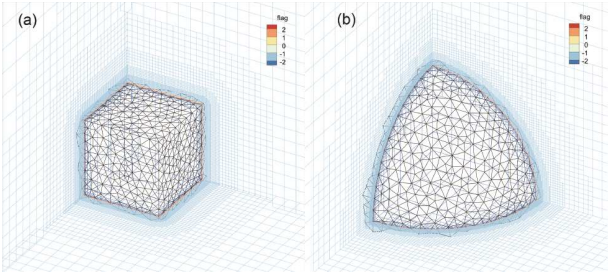

图 9　(网络版彩图)简单外形几何自适应网格. (a) 立方体. (b) 圆球

Figure 9　(Color online) Adaptive meshes of simple geometries. (a) Cube. (b) Sphere.

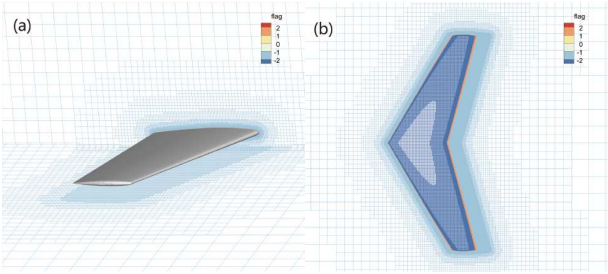

图 10　(网络版彩图) ONERA-M6几何自适应网格. (a) 侧视图. (b) 横切面

Figure 10　(Color online) Adaptive mesh of ONERA-M6 model. (a) Side view. (b) Cross-section.

92, 278, 116, 266, 477倍, 整体效率提升约两个量级. 表3显示, 优化前后内外判断时间的对比中, 本文提出的内外判断算法使网格生成效率分别提高了70, 367, 114, 431和2430倍, 相较于传统方法效率提升了最高三个量级, 与ADT方法相比也有明显提升. 从数值结果来看, 基于KD树的优化算法在处理复杂物面网格时效果优于简单外形, 几何表面三角形单元越多效果越明显. 原因在于KD树将物面元素存储为一深度为$\log_2 M$的平衡二叉树, 在笛卡儿网格数目相近的情况下, 复杂物面外形的性能提升将优于简单外形.

为了测试本文方法在大规模网格计算中的效率, 以SU-27模型为例, 几何自适应次数设置为8, 9, 10次, 最终生成的自适应笛卡儿网格数量分别为2386万、9168万、3.5亿. 表4显示, 对于复杂的三维SU-27模型, 生成三亿网格的时间约为0.3 h. 随着网格数量增加, 相交判断和壁面距离计算的计算占比由23%增至35%, 内外判断的计算占比保持在0.2%. 图12显示, 本文方法的计算时间基本呈线性增长, 表现出较好的超大规模网格生成能力. 值得注意的是, 距离查找加速壁面计算和相交单元判断的方法中, 距离查找使用的空间包围盒与多个子树重叠, 导致时间占比较大[39]. 而点查找实现的内外判断不会出现与多个子树重叠的现象, 显著减少了回溯过程, 可以快速的实现判断.

特别地, 负载平衡是自适应笛卡儿网格大规模并行生成可扩展性的一个重要因素. 以SU-27模型为例, 图13展示了不同几何加密次数下本文网格生成优化方法的负载平衡情况. 横轴为进程序号, 纵轴Ψ为当前进程的实际网格数与理想平均网格数的比值. 结果表明, 不同核内的网格数量差异在0.1%以内, 验证了本文方法在自适应笛卡儿网格并行分区动态负载平衡方面的优异性能.

最后, 为检验本文发展的方法在大规模网格下的并行生成性能, 分别对并行扩展性以及大规模网格的生成效率进行了测试. 如图14所示, 以SU-27模型为对象, 单核负载10万网格在64–1024核进行了计算. 结果表明, 使用KD树优化的两个算法以及总网格生成过程在千核并行下分别能保持在78%, 73%, 63%的并行效率, 表现出良好的并行扩展性. 同时, 我们注意到, 相交判断及壁面距离计算过程造成本文自适应笛卡儿网格生成方法呈现非一致下降的不规律趋势. 其内在原因是: 虽然基于物面三角形单元构造的KD树是平衡的, 但是不同三角单元在KD树中存储的位置不一致造成

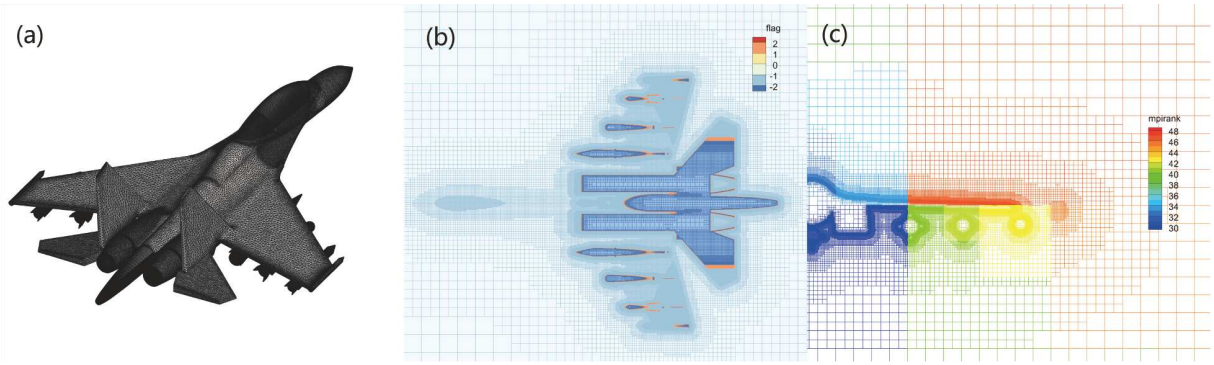

图 11　(网络版彩图) SU-27几何自适应网格. (a) 表面网格. (b) 横切面. (c) 并行分区情况

Figure 11　(Color online) Adaptive mesh of SU-27 model. (a) Surface mesh. (b) Cross-section. (c) Parallel partitioning situation.

表 2　相交判断及壁面距离计算优化前后时间对比(CPU时间/s)

Table 2　Comparison of intersection detection and wall distance calculation before and after optimization (CPU time/s)

算例	遍历法	ADT方法	KD树优化法
圆球	202	4.5	2.2
立方体	1194	6.1	4.3
ONERA-M6	743	20.6	6.4
DLR-F6	1706	18.6	6.4
SU-27	22689	207.3	47.6

表 3　内外判断优化前后时间对比(CPU时间/s)

Table 3　Comparison of interior and exterior cells judgment time before and after optimization (CPU time/s)

算例	遍历法	ADT方法	KD树优化法
圆球	7	0.2	0.10
立方体	55	0.5	0.15
ONERA-M6	16	1.6	0.14
DLR-F6	56	0.6	0.13
SU-27	559	2.3	0.23

了各网格单元判断时进入KD树所需的查询递归次数不同, 使得各核心在此过程的计算量不平衡, 从而导致不同核心并行效率具有一定的随机性和不规律性. 基于这一发现, 后续将对此进行更加深入的研究以避免这种不规律现象, 进而优化整体并行效率.

另一方面, 表4中以SU-27模型为对象, 进行11次几何自适应, 共1024核进行计算. 结果表明, 使用千核并行, 可以在7.2 min内实现14亿网格的高效生成, 单个网格生成时间约为3.1×10^{-4} s. 与法国ONERA纯笛卡儿网格解算器FastS[41]在18 min内336核生成15亿网格相比较(折算成单个网格生成时间为2.4×10^{-4} s), 生成效率基本相当. 而如前文所述, 本文方法在48核生成三

表 4　SU-27模型网格类型判断及壁面距离计算优化后时间(CPU时间/s)

Table 4　Time for mesh type judgment and wall distance calculation after optimization for SU-27 model (CPU time/s)

核数	空间网格数目(万)	内外判断	相交判断及壁面距离计算	网格生成总时间
48	2386	0.23	47.60	202.46
	9168	0.87	131.29	403.31
	35983	3.36	412.52	1166.03
1024	142451	0.58	63.68	431.83

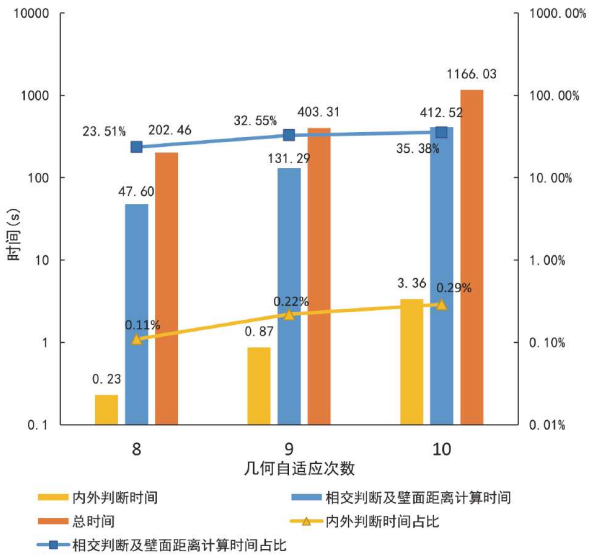

图 12　(网络版彩图)不同几何自适应次数下各部分时间占比(SU-27模型)

Figure 12　(Color online) Time proportion of each part under different geometric AMR times (SU-27 model).

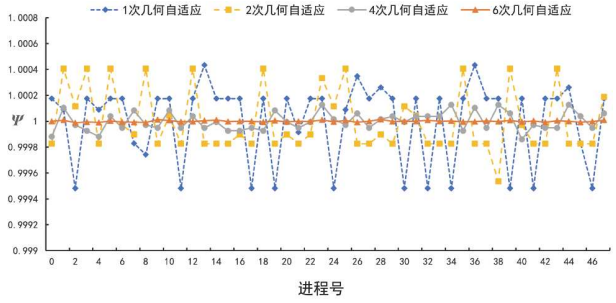

图 13　(网络版彩图)不同几何自适应次数下的负载平衡情况

Figure 13　(Color online) Dynamic partition load balancing with different geometric AMR times.

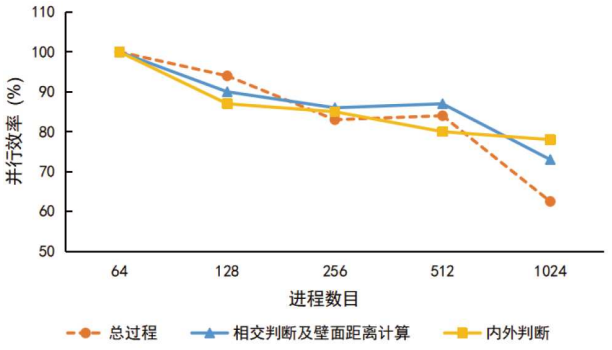

图 14　(网络版彩图)总过程及优化算法的并行扩展性(SU-27模型)

Figure 14　(Color online) Parallel scaling performance of the total process and the optimized algorithms (SU-27 model).

亿网格下的单个网格生成时间约为1.6×10^{-4} s, 相对更快. 同时需要特别说明的是, 本文统计的时间包含了壁面距离这一重要参数计算所带来的消耗. 因此, 除去网格类型、测试环境、并行效率带来的影响, 总体来说, 本文方法在大规模网格下仍然表现出较高效率.

5　结论

针对三维复杂外形, 本文发展了一种自适应笛卡儿网格并行生成加速优化方法. 一方面, 使用KD树存储代替传统的顺序存储, 借助其能够快速高效进行距离查找和点查找的性能, 设计了一种通过距离查找加速壁面计算和相交单元判断的算法. 另一方面, 对于笛卡儿网格生成中的内外判断问题, 利用所构建的KD树发展了一种坐标轴投影方法用于提升效率. 通过多个三维简单/复杂几何构型的数值结果表明, 相比于传统的遍历方法, 本文所发展的相交单元判断和壁面距离计算方法效率能够提升上百倍, 内外判断优化算法效率可提升2–3个量级. 此外, 在千核并行框架下生成14亿自适应笛卡儿网格仅需7.2 min (含壁面距离计算), 而不同核之间的网格数差异在0.1%以内, 充分体现了本文方法具有优异的大规模CPU并行可扩展性. 在未来工作中, 计划在网格负载平衡的基础上进一步实现计算负载的平衡. 同时, 将结合团队前期发展的并行框架浸入边界方法实现复杂外形绕流问题的计算, 进而实现自适应笛卡儿网格全自动生成与更加高效计算的一体化流程.

参考文献

1　Slotnick J P, Khodadoust A, Alonso J, et al. CFD vision 2030 study: A path to revolutionary computational aerosciences. Technical Report, NASA/CR-2014-218178, NASA, 2014

2　Cottrell J A, Hughes T J, Bazilevs Y. Isogeometric Analysis: Toward Integration of CAD and FEA. Hoboken: John Wiley & Sons, 2009, 1–3

3　Chawner J R, Dannenhoffer J, Taylor N J. Geometry, mesh generation, and the CFD 2030 vision. In: Proceedings of the 46th AIAA Fluid Dynamics Conference. Reston: AIAA, 2016. 3485

4　Capizzano F. Automatic generation of locally refined Cartesian meshes: Data management and algorithms. Numer Meth Eng, 2018, 113: 789–813

5　Delanaye M, Aftosmis M, Berger M, et al. Automatic hybrid-Cartesian grid generation for high-Reynolds number flows around complex geometries. In: Proceedings of the 37th Aerospace Sciences Meeting and Exhibit. Reston: AIAA, 1999. 777

6　Lu J Y, Chen J, Liu J, et al. Characteristics analysis of automated CFD based on Cartesian grid and a new strategy (in Chinese). Aero Sci Meet Exhib, 2023, 3: 70–81 [卢俊宇, 陈洁, 刘君, 等. 基于笛卡儿网格的自动化 CFD 特点分析及一种新策略. 气动研究与实验, 2023, 3: 70–81]

7　Voorhies D. Triangle-cube intersection. Graphics Gems III (IBM Version). Amsterdam: Elsevier, 1992. 236–239

8　Akenine-Möller T. Fast 3D triangle-box overlap testing. J Graph Tools, 2001, 6: 29–33

9 Qi X, Yang Y, Tian L, et al. A parallel methodology of adaptive Cartesian grid for compressible flow simulations. Adv Aerodyn, 2022, 4: 21

10 Yang Y, Qi X, Wang Z, et al. An immersed boundary method based on parallel adaptive Cartesian grids for high reynolds number turbulent flow. Int J Comput Fluid Dyn, 2022, 36: 319–341

11 Wang L, Tian F B. Recent progress of immersed boundary method and its applications in compressible fluid flow (in Chinese). Sci Sin-Phys Mech Astron, 2018, 48: 094703 [王力, 田方宝. 浸入边界法及其在可压缩流动中的应用和进展. 中国科学: 物理学 力学 天文学, 2018, 48: 094703]

12 Mahovsky J, Wyvill B. Fast ray-axis aligned bounding box overlap tests with Plucker coordinates. J Graph Tools, 2004, 9: 35–46

13 Chen H, Bi L, Hua R H, et al. Efficient Cartesian mesh generation method based on fully threaded tree data structure (in Chinese). Acta Aeronaut Astronaut Sin, 2022, 43: 125170 [陈浩, 毕林, 华如豪, 等. 基于全线程树数据结构的笛卡尔网格高效生成技术. 航空学报, 2022, 43: 125170]

14 Melton J, Berger M, Aftosmis M, et al. 3D applications of a Cartesian grid Euler method. In: Proceedings of the 33rd Aerospace Sciences Meeting and Exhibit. Reston: AIAA, 1995. 853

15 Chen H, Yuan X X, Wang T T, et al. Advances in viscous adaptive Cartesian grid methodology of NNW Project (in Chinese). Acta Aeronaut Astronaut Sin, 2021, 42: 118–136 [陈浩, 袁先旭, 王田天, 等. 国家数值风洞(NNW)工程中的黏性自适应笛卡尔网格方法研究进展. 航空学报, 2021, 42: 118–136]

16 Ishida T, Takahashi S, Nakahashi K. Efficient and robust Cartesian mesh generation for building-cube method. J Comput Sci Technol, 2008, 2: 435–446

17 Mittal R, Dong H, Bozkurttas M, et al. A versatile sharp interface immersed boundary method for incompressible flows with complex boundaries. J Comput Phys, 2008, 227: 4825–4852

18 Li X, Yang M, Bi L, et al. An efficient Cartesian mesh generation strategy for complex geometries. Comput Methods Appl Mech Eng, 2024, 418: 116564

19 Shen Z W, Zhao N, Hu O. Numerical research of Cartesian based ghost cell method for compressible viscous flows (in Chinese). Acta Aerodyn Sin, 2014, 32: 748–754 [沈志伟, 赵宁, 胡偶. 可压缩粘性流动笛卡尔网格虚拟单元方法研究. 空气动力学学报, 2014, 32: 748–754]

20 Luo C Y, Zhou D, Du H, et al. A novel high-fidelity ghost-cell immersed boundary method for the Cartesian grid and its applications (in Chinese). Sci Sin-Phys Mech Astron, 2024, 54: 234611 [罗灿炎, 周丹, 杜昊, 等. 笛卡儿网格高保真虚拟单元浸入边界法研究与应用. 中国科学: 物理学 力学 天文学, 2024, 54: 234611]

21 Lai Y G, So R M C. On near-wall turbulent flow modelling. J Fluid Mech, 1990, 221: 641–673

22 Meng S, Zhou D, Li X L, et al. Accurate and efficient wall distance calculation method for Cartesian grids (in Chinese). Acta Aerodyn Sin, 2023, 41: 93–101 [孟爽, 周丹, 李雪亮, 等. 笛卡尔网格下精确高效的壁面距离计算方法. 空气动力学学报, 2023, 41: 93–101]

23 Li G N, Li F W, Zhou Z H. An efficient method for Calculating wall distance (in Chinese). Adv Aeronaut Sci Eng, 2010, 1: 137–142 [李广宁, 李凤蔚, 周志宏. 一种高效的壁面距离计算方法. 航空工程进展, 2010, 1: 137–142]

24 Tucker P G. Differential equation-based wall distance computation for DES and RANS. J Comput Phys, 2003, 190: 229–248

25 Xu J, Yan C, Fan J. Computations of wall distances by solving a transport equation. Appl Math Mech-Engl Ed, 2011, 32: 141–150

26 Wang G, Zeng Z, Ye Z Y. An efficient search algorithm for calculating minimum wall distance of unstructured mesh (in Chinese). J Northwestern Polytechnical Univ, 2014, 32: 511–516 [王刚, 曾铮, 叶正寅. 混合非结构网格下壁面最短距离的快速计算方法. 西北工业大学学报, 2014, 32: 511–516]

27 Guo Z Z, He Z Q, Xia C C, et al. KD tree method for efficient wall distance computation of mesh (in Chinese). J National Univ Defense Technol, 2017, 39: 21–25 [郭中州, 何志强, 夏陈超, 等. 高效计算网格壁面距离的KD树方法. 国防科技大学学报, 2017, 39: 21–25]

28 Zhang X Q, Lai K L. Simulation of transonic aeroservoelasticity using Cartesian-grid based flow solver. In: Proceedings of the 42nd AIAA Fluid Dynamics Conference and Exhibit, Reston: AIAA, 2012. 3266

29 Ma T, Li P, Ma T. A three-dimensional Cartesian mesh generation algorithm based on the GPU parallel ray casting method. Appl Sci, 2019, 10: 58

30 Park S, Shin H. Efficient generation of adaptive Cartesian mesh for computational fluid dynamics using GPU. Numer Methods Fluids, 2012, 70: 1393–1404

31 Burstedde C, Wilcox L C, Ghattas O. p4est: Scalable algorithms for parallel adaptive mesh refinement on forests of octrees. SIAM J Sci Comput, 2011, 33: 1103–1133

32 Boger D A. Efficient method for calculating wall proximity. AIAA J, 2001, 39: 2404–2406

33 Bentley J L. Multidimensional binary search trees used for associative searching. Commun ACM, 1975, 18: 509–517
34 Lee D T, Wong C K. Worst-case analysis for region and partial region searches in multidimensional binary search trees and balanced quad trees. Acta Inform, 1977, 9: 23–29
35 Hasbestan J J, Senocak I. Binarized-octree generation for Cartesian adaptive mesh refinement around immersed geometries. J Comput Phys, 2018, 368: 179–195
36 Ning J, Ma T, Lin G. A grid generator for 3-D explosion simulations using the staircase boundary approach in Cartesian coordinates based on STL models. Adv Eng Software, 2014, 67: 148–155
37 Shevtsov M, Soupikov A, Kapustin A. Highly parallel fast KD-tree construction for interactive ray tracing of dynamic scenes. Comput Graph Forum, 2010, 26: 395–404
38 Iaccarino G, Ham F. Automatic mesh generation for LES in complex geometries. CTR Annu Briefs, 2005, 20: 19–30
39 Meng S, Zhou D, Yuan X, et al. Enhanced strategy for adaptive Cartesian grid generation with arbitrarily complex 3D geometry. Adv Eng Software, 2022, 174: 103304
40 Möller T, Trumbore B. Fast, minimum storage ray-triangle intersection. J Graph Tools, 1997, 2: 21–28
41 Péron S, Benoit C, Renaud T, et al. An immersed boundary method on Cartesian adaptive grids for the simulation of compressible flows around arbitrary geometries. Eng Comput, 2021, 37: 2419–2437

An accelerated optimization method for parallel generation of adaptive Cartesian grids

YANG ZiXuan[1], QI XinYu[1], WANG ZhenMing[3], TIAN LinLin[2], LIU JianMing[4] & ZHAO Ning[1,2*]

[1] *State Key Laboratory of Mechanics and Control for Aerospace Structures, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China*
[2] *Jiangsu Key Laboratory of Hi-Tech Research for Wind Turbine Design, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China*
[3] *Key Laboratory of Mathematical Modelling and High Performance Computing of Air Vehicles, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China*
[4] *School of Mathematics and Statistics, Jiangsu Normal University, Xuzhou 221116, China*
**Corresponding author (email: zhaoam@nuaa.edu.cn)*

The efficiency and quality of computational mesh generation directly determine the quality of numerical simulation results. The Cartesian mesh has received widespread attention due to its good adaptability, high degree of automation, and strong ability to handle complex shapes. However, for complex shapes, the extremely large mesh volume makes it difficult to achieve satisfactory mesh generation efficiency. Therefore, it is necessary to develop 3D Cartesian mesh parallel generation technology that is versatile, efficient, and highly scalable. In this paper, a 3D Cartesian mesh parallel generation optimization method is proposed based on the KD tree (K-Dimensional Tree). First, based on the geometric information of complex shapes in 3D flow problems, a set of surface triangles is used as input to generate a bounding box enclosing the shape, which is then used to construct the KD tree. Second, an accelerated optimization algorithm for wall distance calculation and intersecting cell judgment is developed based on the constructed KD tree, along with an algorithm that accelerates the classification of inner and outer cells by performing a lookup on the projected points in the KD tree. Finally, the effectiveness of the adaptive Cartesian mesh parallel generation algorithm proposed in this paper is verified by using models of a sphere, cube, M6 wing, and Su-27 aircraft. The numerical results show that the optimization method for intersection judgment and wall distance calculation proposed in this paper can improve efficiency by two orders of magnitude compared to the traditional method, while the efficiency of inner and outer cell classification can be improved by three orders of magnitude. This fully demonstrates that the proposed optimization method can generate adaptive Cartesian meshes required for flow field calculations in a fast, robust, and automatic manner.

Cartesian grids, AMR, parallel generation, KD tree, complex geometry

PACS: 47.11.-j, 47.85.Gj, 87.55.de

doi: 10.1360/SSPMA-2024-0211