Journal of System Simulation

Volume 30 | Issue 7 Article 31

1-8-2019

Real-time Collision Detection of Coastline in Navigation Simulator

Qianfeng Jing
Key Lab. Of Marine Simulation and Control, Dalian Maritime University, Dalian 116026, China;

Helong Shen
Key Lab. Of Marine Simulation and Control, Dalian Maritime University, Dalian 116026, China;

Yin Yong
Key Lab. Of Marine Simulation and Control, Dalian Maritime University, Dalian 116026, China;

Xiuwen Liu
Key Lab. Of Marine Simulation and Control, Dalian Maritime University, Dalian 116026, China;

Real-time Collision Detection of Coastline in Navigation Simulator

Abstract

Abstract: In order to develop real-time collision detection module of coastline in navigation simulator, a mesh preprocessing method for coastline data is applied. *The principle of meshing is established and the coastline data is meshed based on the clipping algorithm. The mesh data is stored into an appropriate data structure.* The collision detection algorithm and test program are developed based on Microsoft Foundation Classes (MFC). *The ship contour points and the mesh expansion method are applied to avoid error in special cases which could also improve the accuracy of collision detection.* The results show that the method reduces much time compared with the method of traversing the coastline directly, which ensures the real-time performance of the program. The accuracy of the detection algorithm is high, which could enhance the behavior and physical realism of the navigation simulator as a virtual reality system.

Keywords

navigation simulator, virtual reality, collision detection, real-time

Recommended Citation

Jing Qianfeng, Shen Helong, Yin Yong, Liu Xiuwen. Real-time Collision Detection of Coastline in Navigation Simulator[J]. Journal of System Simulation, 2018, 30(7): 2682-2688.

第 30 卷第 7 期 系统仿真学报© Vol. 30 No. 7
2018 年 7 月 Journal of System Simulation Jul., 2018

航海模拟器中的岸线实时碰撞检测

景乾峰，神和龙，尹勇，刘秀文
(大连海事大学 航海动态仿真和控制实验室，大连 116026)

摘要：为实现航海模拟器中岸线的实时碰撞检测，对岸线数据采取网格化的预处理方式。*建立网格划分的原则；基于裁剪算法进行划分，将网格数据存储至合适的数据结构*；基于 MFC 实现碰撞检测算法并开发了测试程序。为增强碰撞检测的准确性，*采用船舶轮廓点进行计算并应用网格拓展方法来避免特殊情况下的判断错误*。结果表明，采用预处理后的岸线数据较直接遍历岸线大幅降低了运算量，保证了程序的实时性；所采取的碰撞检测算法精确度较高，增强了航海模拟器作为虚拟现实系统的行为和物理真实感。

关键词：航海模拟器；虚拟现实；碰撞检测；实时性
中图分类号：TP391.9 文献标识码：A 文章编号：1004-731X (2018) 07-2682-08
DOI: 10.16182/j.issn1004731x.joss.201807031

Real-time Collision Detection of Coastline in Navigation Simulator

Jing Qianfeng, Shen Helong, Yin Yong, Liu Xiuwen
(Key Lab. Of Marine Simulation and Control, Dalian Maritime University, Dalian 116026, China)

Abstract: In order to develop real-time collision detection module of coastline in navigation simulator, a mesh preprocessing method for coastline data is applied. *The principle of meshing is established and the coastline data is meshed based on the clipping algorithm. The mesh data is stored into an appropriate data structure.* The collision detection algorithm and test program are developed based on Microsoft Foundation Classes (MFC). *The ship contour points and the mesh expansion method are applied to avoid error in special cases which could also improve the accuracy of collision detection.* The results show that the method reduces much time compared with the method of traversing the coastline directly, which ensures the real-time performance of the program. The accuracy of the detection algorithm is high, which could enhance the behavior and physical realism of the navigation simulator as a virtual reality system.

Keywords: navigation simulator; virtual reality; collision detection; real-time

引言

根据航海模拟器标准的要求[1]，航海模拟器应尽可能地符合真实环境中的情况。由于船舶在岸线附近航行时，与岸线的交互主要体现在碰撞，所以开发高效的碰撞检测功能可以增强航海模拟器作为 VR 系统的行为真实感[2]。碰撞检测作为虚拟现实系统中的关键技术已有诸多研究成果[3-4]。

在船舶与岸线的碰撞检测过程中，与大量数据进行交互势必耗费一定时间，因此岸线实时碰撞检测作为航海模拟器的功能模块主要存在以下两方面研究难点：一是平衡碰撞检测模块与主程序的性能开销，保证程序实时性；二是碰撞发生后，准确地传递碰撞点信息，方便下一步碰撞力的计算。

收稿日期：2017-08-06 修回日期：2017-11-28；
基金项目：交通青年科技英才项目(36260401)，海洋公益性行业科研专项(201505017-4)，中央高校基本科研业务费(3132016310)；
作者简介：景乾峰(1993-)，男，陕西，博士生，研究方向为航海模拟器、船舶运动仿真。

传统检测方式通常为岸线和船舶建立包围盒，并为船舶外形建立包围盒，在碰撞检测模块中，通过计算岸线包围盒与船舶包围盒的关系来判断船舶与岸线的碰撞情况[5]。由于包围盒是为了简化计算而对物体外形的近似描述，所以这种处理方式精度较低[6]。在岸线数据量较小时该方法能保证较少的运算时间，但遇到数据量较大的岸线时，实时性通常难以保证。文献[7]采取包围盒法与相交测试法结合的碰撞检测方法进行航海模拟器中的船舶碰撞检测，文献[8]采取轴向包围盒技术，分别对船舶和桥体构建碰撞检测包围盒，并且根据桥墩实际情况来选取不同的包围盒进行建模来实现碰撞检测。

对于航海模拟器而言，主程序生成视景等功能模块需要大量性能开销，为保证程序的实时性，其他功能应尽可能地减少运算时间，即使在毫秒级降低运算时间也十分有意义。本文对这一问题进行研究，在该基础上进行优化，首先对岸线数据进行预处理，通过图形学裁减算法对岸线进行二维网格的划分，将处理后的网格数据存储至 map 映射关系。在碰撞检测程序中，通过船位获得船舶所在网格的坐标，通过网格拓展，按需求对其周围的网格也进行索引，利用 map 映射找到所有网格内的岸线数据，对这些网格内的数据进行遍历，利用船舶外轮廓数据点替换以往的包围盒方法，对数据和轮廓点进行计算来判断碰撞情况。

1 船、岸碰撞检测的方法分析

1.1 岸线的特点分析

1.1.1 连续性

岸线数据由连续的点组成，两点之间有前后顺序，碰撞检测算法中需要利用点之间的顺序关系判断船舶在岸线的哪一侧。从岸线中读取和处理这些数据时都不应改变数据点的存放顺序。

1.1.2 区域分布性

在一张海图中岸线主要分布在港口附近，在集中分布的区域外数据量十分稀少，划分网格时应当考虑到这一点，选取高效率的划分方式有助于降低程序运算时间。

1.2 岸线检测方法比较

1.2.1 包围盒法

经典方法主要有 AABB (Aligned Axis Bounding Box)、Oriented Bounding Box(OBB)、包围球(Sphere)、K-DOPs (Discrete Orientation Polytope)，文献[9]对以上方法进行了对比研究，对比结果如表 1 所示(1 代表最优，4 代表最劣)。

表 1 不同包围盒比较
Tab. 1 Different bounding box comparison

包围盒类别	构造难度	存储量	相交测试难度	紧密性	更新计算量
包围球	2	1	1	4	1
AABB	1	2	2	3	3
OBB	4	3	4	2	2
K-DOPs	3	4	3	1	4

一般情况下岸线跨度较大，其尺度远大于船长，如图 1 所示，需要对岸线进一步细分后建立包围盒，该步骤需要耗费一定时间，而且包围盒本身就是一种近似方法，因此不论采取何种包围盒，都无法精确地描述岸线外形，导致碰撞检测的精确度受到包围盒大小和形状的制约[10]。

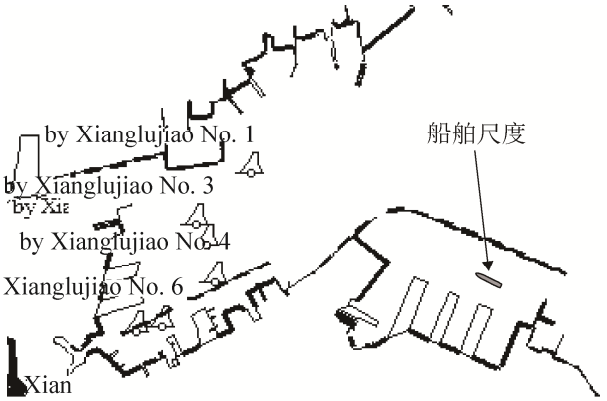

图 1 海图中岸线示例
Fig. 1 Coastline in ECDIS

第 30 卷第 7 期 系统仿真学报 Vol. 30 No. 7
2018 年 7 月 Journal of System Simulation Jul., 2018

1.2.2 二维网格法

本文提出一种针对岸线优化的二维网格法，该方法通过海图长宽和船舶尺度来确定单元网格的长和宽，对于不同的岸线和船舶网格大小和数量不同，通过图形学裁剪算法对岸线进行网格划分，该方法具有流程清晰，网格划分原理简单，易于实现，数据存储结构清晰，查找便捷等特点，在预处理过程中对每个网格是否存有数据进行标记，大幅减少了运算过程，以此保证程序的实时性。

1.3 船舶外边界比较

1.3.1 包围盒法

对船舶建立包围盒也可采用上文提到的包围盒方法，其效果如图 2 所示。

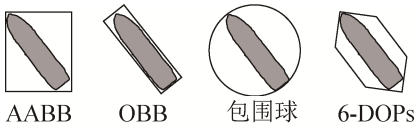

图 2 不同船舶包围盒示例
Fig. 2 Different ship bounding box

1.3.2 轮廓点法

船舶轮廓点一般由 30~40 个型值点组成，如图 3 所示，该方法能够较为准确的描述船舶外边界形状，型值点相对于船舶中心的位置固定不变，船舶旋转后的型值点由坐标变换得到，计算速度优于 K-DOPs 方法，有利于快速给出碰撞力的受力方向。

图 3 船舶轮廓点示例
Fig. 3 Ship contour points

2 网格划分

2.1 网格划分原则

2.1.1 岸线跨过网格

在岸线变化趋势很小的部分，两个相邻的数据点距离较远。在网格划分过程中，直接将数据点划分至对应范围的网格中会出现如图 4 所示情形。

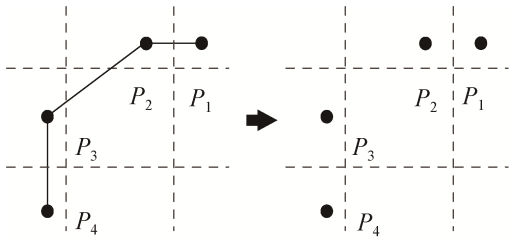

图 4 岸线点跨过网格
Fig. 4 Coastline points across mesh

被横跨的网格中未存储任何数据，这将导致碰撞检测出现错误，正确的网格划分应当如图 5 所示。

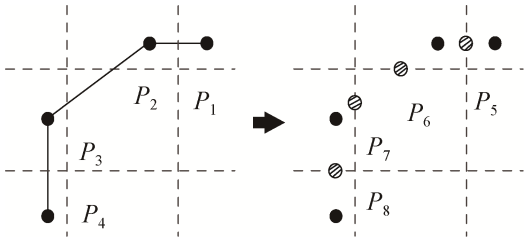

图 5 岸线点横跨网格产生点
Fig. 5 New points generated across the coastline

图 5 中产生了 4 个新的数据点，每次跨过网格时都应该产生位于网格线边界上的数据点，对比图 4 中间网格，图 5 中网格包含新产生的 P6 和 P7 数据点，这两点可以产生被网格分隔的岸线，且不改变原有数据点的顺序，因此网格划分应采用基于网格大小窗口对岸线进行裁剪的方式，而不能简单地将数据划分至对应网格。

2.1.2 多边形封闭问题

本文基于开源图形裁剪工具包 clipper 完成网格划分中的裁剪工作[11]，在 clipper 工具包中有两类图形格式，封闭和不封闭，对岸线进行裁剪的过程中，需要考虑多边形的封闭问题。由于裁剪窗口是网格大小的封闭矩形，岸线是否为封闭图形会得到不同的裁剪结果，如图 6 和图 7 所示。

从图中可以看出，如果岸线采取封闭图形的格式存储，裁剪后取交集得到的结果是一块完整的图形；当岸线为不封闭图形时，裁剪后取交集得到一

小段岸线，图 6 中裁剪后得到的阴影部分在碰撞检测过程中没有贡献，属于冗余数据，故岸线在裁剪之前应当以不封闭图形的格式存储。

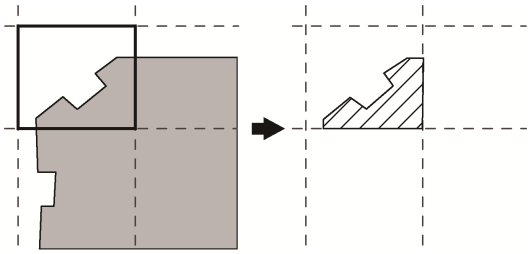

图 6 岸线为封闭图形
Fig. 6 Closed graphics

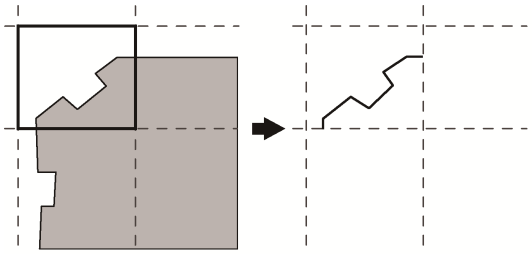

图 7 岸线为不封闭图形
Fig. 7 Open Graphics

2.2 岸线数据预处理

2.2.1 岸线与船舶轮廓数据读取

从海图中抽取岸线数据，存放至文本文档，该部分工作基于实验室基础程序完成。下一步读取文档中的岸线数据存至 clipper 库中提供的用于保存不封闭图形的数据结构。船舶轮廓点数据由船舶文件读取，保存至封闭图形结构。

2.2.2 网格划分及存储

本文建立了一个网格类，用于实现网格划分和网格数据存储。类中包含一个表示网格内是否存在数据的逻辑变量，在划分网格的过程中对其进行赋值。该变量能够在检测过程中降低数据处理的时间。网格划分基于 clipper 库中的求交函数，网格大小由岸线尺度和船长共同决定，一般网格的长和宽为 2 到 3 倍船长，确定完网格大小后，逐行逐列对岸线进行裁剪，将当前裁剪的网格坐标与裁剪后得到的网格内的数据点存放至 map 映射关系。其中 key 是网格的二维坐标，value 是网格中的数据。

网格大小的设置考虑到船舶的船长、船速、船舶领域、船舶是否航行在狭窄水域等因素，网格不应过小，通常船舶在航行中 2~3 倍船长范围内很少有其他船舶进入[12]，为了避免大型船舶的船长带来误差，本文将网格大小简化选取为大型船舶船长的 2.5 倍。

2.2.3 预处理效率分析

利用本文所开发的划分工具对模拟器中已有的所有港口的岸线数据进行划分预处理，处理完的数据统一存放在模拟器程序中。该处理过程需要耗费一定时间，但由于预处理工作是独立于模拟器程序的，并不占用程序的初始化时间。

本文考虑到可能存在一些新的港口从未进行过预处理，针对这一情况，将岸线划分程序集成至模拟器程序中，在初始化工作中首先检测该港口的预处理数据是否存在，如果不存在则进行预处理并保存数据到文件中，下一次该港口数据不用重复处理。本文对不同网格数量下的预处理时间进行了测试，如表 2 所示。

表 2 预处理时间测试
Tab. 2 Preprocessing time cost

岸线数据量/kb	行数	列数	网格数量	划分时间/s
41	1	2	2	1.05
82	9	13	117	1.69
122	9	13	117	1.66
166	9	13	117	1.71
232	9	13	117	1.68
317	26	35	910	5.76
544	41	48	1 968	19.43

从表 2 可看出，随着网格数量的增多，划分处理的耗时也在增长，从中间 4 行可看出，对于边界相同的岸线数据来说，划分的网格数量一致，仅增加数据量对划分处理的时间影响不大，总体来看，在网格数量达到近 2 000 个时划分时间约为 20 s，对于模拟器程序的初始化来说，该耗时是可以接受的，且这种处理是一次性的，仅当出现全新的港口数据时才进行。

https://dc-china-simulation.researchcommons.org/journal/vol30/iss7/31
DOI: 10.16182/j.issn1004731x.joss.201807031

2.2.4 预处理流程

模拟器程序运行后，当选定港口之后，会进行该港口的数据初始化工作，首先检测是否存在已处理好的岸线划分数据，若数据不存在则对岸线进行预处理，并将处理好的数据存放至 map 映射结构，所有实时碰撞检测中的岸线数据来源于此结构，具体工作流程如图 8 所示。

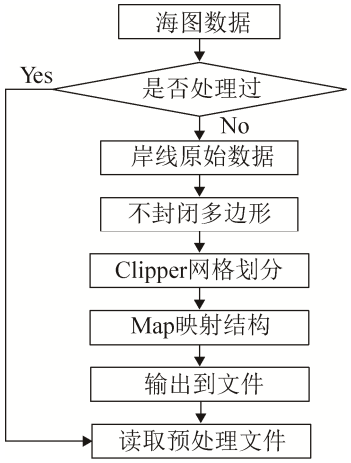

图 8 岸线预处理流程图
Fig. 8 Coastline preprocessing flow chart

3 实时碰撞检测

3.1 碰撞检测流程

实时碰撞检测流程如图 9 所示。

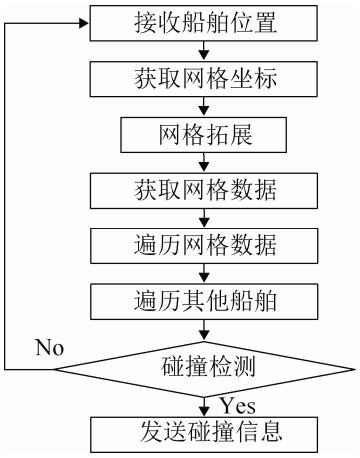

图 9 实时碰撞检测流程图
Fig. 9 Collision detection flow chart

在每一次仿真循环中，模拟器向碰撞检测模块发送所有我船和其他船的船位、航向信息，首先对我船与岸线进行碰撞检测，然后对我船与其他船进行碰撞检测。

其中船舶之间的碰撞检测需要遍历所有船的位置和轮廓点信息进行运算，这种处理方式显然相比于包围盒算法效率低，但为获得更准确的碰撞信息，采取这一方法是有必要的。本文在遍历其他船舶进行碰撞检测时设置了一个阈值，并对程序代码进行了优化，当其他船舶进入该阈值时再进行碰撞检测运算，通过这样的方式大幅减少了需要遍历的数据，同时又能获得较为准确的碰撞点信息。由于该阈值的考虑因素与网格大小相同，故阈值设置为与网格大小相等。对不同船舶数量下的碰撞检测效率进行了测试，随机设置进入阈值的船舶数量，对不同数量下的单次碰撞检测时间进行多次记录取平均值，测试结果如表 3 所示。

表 3 不同船舶数量时间测试
Tab. 3 Time cost of different number of ships

船舶数量/艘	阈值/m	单次碰撞检测时间/ms
10	300*2.5	1.43
50	300*2.5	1.52
200	300*2.5	1.53
500	300*2.5	1.65

从表 3 中可以看出在阈值内船舶数量达到增加到 500 艘时对碰撞检测的效率影响较小，而这在海上情况中已经不具有实际意义，通常在该阈值内的船舶数量很少，尤其是开阔水域中，可见优化后的算法代码完全能满足程序实时性的要求。

3.2 碰撞检测算法

3.2.1 网格拓展

在碰撞检测的过程中可能遇到网格边界十分贴近岸线的特殊情况，如图 10 所示。

其中船舶所在网格坐标为(i，j)，当前网格并无岸线数据，但由于岸线十分贴近边界，实际上船舶马上将发生碰撞，由于岸线数据多且形状复杂，难以在划分时避免所有特殊情况，故本文应用一种网格拓展的方式，当获得船舶所在网格后，通过平

移计算获得周围网格的坐标，进而获取所有网格的数据，对这些网格进行遍历计算，防止出现与实际情况不符的错误，具体向外拓展范围视网格大小决定。

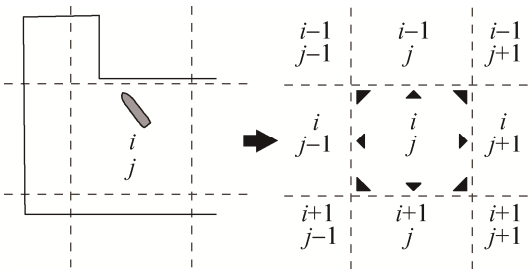

图 10 网格拓展
Fig. 10 Mesh expansion

3.2.2 点与线段碰撞检测

实际碰撞检测算法中，通过计算船舶轮廓点与岸线数据每两点构成的线段的关系进行判断，主要包括以下两部分算法：

(1) 点在线段的两侧

本文通过计算矢量三角形的面积来确定点在直线的哪一侧。如图 11 所示。

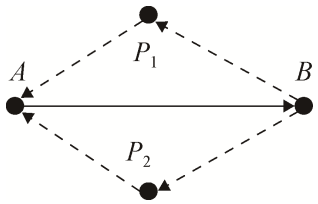

图 11 点在线段两侧
Fig. 11 Points at different side

两个相邻的岸线数据点构成有向线段 AB，船舶轮廓点之中一点为 P_1 或 P_2，令矢量起点为 $A(x_1, y_1)$终点为 $B(x_2, y_2)$判断点为 P_1 或 $P_2(x_p, y_p)$通过式(1)计算三点构成的矢量三角形面积的正负可以判断点位于线段的哪一侧。

$$S(A,B,C)=\frac{(x_1-x_3)(y_2-y_3)-(y_1-y_3)(x_2-x_3)}{2} \tag{1}$$

(2) 点到线段最短距离

不同于点到直线的距离，点到线段有一定的特殊性，如图 12 所示。

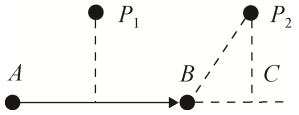

图 12 点到线段最短距离
Fig. 12 Minimum distance from point to segment

图中 P_1 位于线段之间，P_2 位于线段之外，两种情况下到线段的最短距离不同。可以根据向量的投影关系快速得到点到线段的距离。通过式(2)计算得到 r。

$$r=\frac{(\overrightarrow{AP}\cdot\overrightarrow{AB})}{(\overrightarrow{AB})^2} \tag{2}$$

通过 r 的值，利用式(3)进一步确定最短距离。

$$d=\begin{cases}\left|\overrightarrow{AP}\right|, r\leqslant 0\\ \left|\overrightarrow{BP}\right|, r\geqslant 1\\ \left|\overrightarrow{CP}\right|, other\end{cases} \tag{3}$$

4 测试程序

4.1 测试程序功能

测试程序基于 MFC 开发，主要包含碰撞检测提示，坐标显示，网格及岸线显示，视图缩放，船舶拖拽、复位等功能，其界面如图 13 所示。

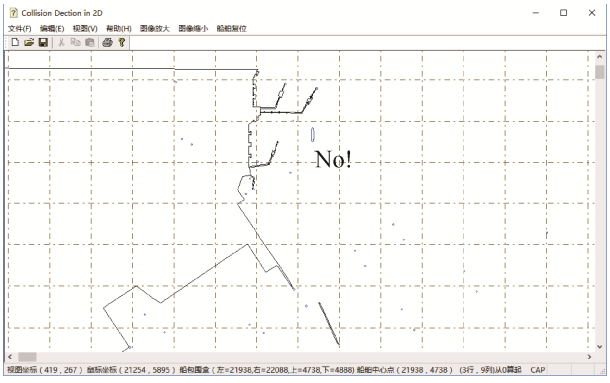

图 13 测试程序界面
Fig. 13 Test program interface

4.2 程序测试结果

本文对直接遍历法和网格划分法进行了简要对比，岸线直接遍历法指数据不进行网格化处理，直接遍历全部数据进行碰撞检测。首先在相同的边

DOI: 10.16182/j.issn1004731x.joss.201807031

第 30 卷第 7 期 系统仿真学报 Vol. 30 No. 7
2018 年 7 月 Journal of System Simulation Jul., 2018

界而岸线数据量不同的情况下进行测试，然后在不同边界下岸线数据量也不同的情况下进行测试，测试主要比对单次碰撞检测的运行时间，通过进行大量单次碰撞检测工作流程并取均值得到结果，测试机器配置为 2.6 GHz Intel Core i5 处理器，8g 内存，比较结果如表 4 所示。

表 4 不同网格数量下单次运行时间比较
Tab. 4 Time cost of different number of mesh

岸线数据量/kb	网格数量	全遍历法/ms	网格化法/ms
41	9*13	50.1	1.2
82	9*13	76.3	1.2
122	9*13	120.1	1.3
166	9*13	135.9	1.2
232	9*13	188.1	1.3
317	26*35	251.8	1.3
544	41*48	447.3	1.4

根据表 4 绘制不同岸线数据量下的运行时间折线图如图 14 所示。

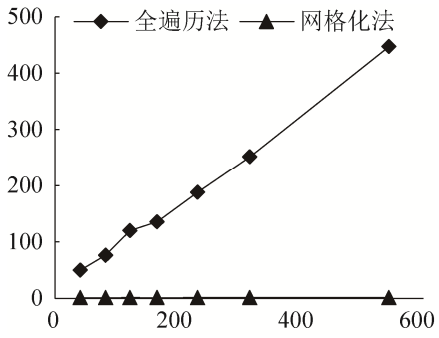

图 14 运行时间变化趋势
Fig. 14 Time cost tendency

从图 14 中可以清晰地看出在同样的图幅范围下，网格数量相同，但岸线数量不同时，全遍历法耗时随着岸线数据量的增加而稳定增长，在扩大图幅的情况下，全遍历法耗时与岸线数据量增长仍保持线性增长关系。经过数据预处理后的划分网格法耗时关于岸线数据量的增长几乎没有变化，这是由于该算法的原理决定的，海图数据经划分处理后，每次进行碰撞检测需要遍历的数据量很少，即使海图数据量很大也能高效地进行碰撞检测。

对于航海模拟器的而言，视景画面达到 30fps (Frames Per Second)才能满足实时性，即程序在一个仿真周期内仅有约 33.3 ms 的时间完成所有任务。从实时性角度分析，在处理器及内存性能约束下，直接遍历法在不同网格数量下耗费的时间与数据量成线性增长关系，且耗时较多，而网格化后的处理时间随着岸线数据量的增长几乎保持不变，证明该方法对性能的提升十分显著，节省了运行时间。

5 结论

本文对航海模拟器中的岸线实时检测进行了研究，分析了岸线特点，对不同检测方法进行了简要比较，最终选取了适合岸线数据的二维网格加船舶轮廓点的方法进行碰撞检测模块的开发，基于 MFC 实现了该模块并开发了测试程序。在测试程序中对岸线直接遍历法和网格化后的岸线遍历法进行了运行时间的比较，结果表明本文采取的方法相对于直接遍历法能够大幅降低运行时间，而且网格数量增加对其影响较小，从而保证了程序的实时性，应用网格拓展的方式也解决了网格划分中可能出现的特殊情况，提高了碰撞检测的精确度，采用船舶轮廓点替换包围盒能够迅速的给出发生碰撞的点信息，从而为下一步计算碰撞力提供便利，本文初步解决了碰撞检测中精确度和实时性矛盾的问题，作为航海模拟器的子模块仍有一定不足，下一步将会研究三维情况下的碰撞检测，完善这一功能模块。

参考文献:

[1] 金一丞, 尹勇. 航海模拟器[M]. 北京: 科学出版社, 2013. MARITIME SIMULATOR[M]. 2013

[2] 金一丞, 尹勇. STCW 公约马尼拉修正案下的航海模拟器发展战略[J]. 中国航海, 2012, 35(3): 5-10.
Jin Yicheng, Yin Yong. Development strategy of marine simulator in light of the manila amendments to STCW convention[J]. Navigation of China, 2012, 35(3): 5-10.

[3] Zhang L, Teng J, Feng T, et al. The Study on the Technology of Collision Detection in Virtual Reality Environment[C]//Wavelet Analysis and ITS Applications, and Active Media Technology-the International Computer Congress. 2004: 905-910.

(下转第 2699 页)