Journal of System Simulation

Volume 31 | Issue 11 Article 24

12-13-2019

Dynamic Collision Optimization Algorithm Based on Ray Detection

Li Xing
1. School of Computer Science and Engineering, Xi'an Technological University, Xi'an 710025, China; ;

Yanfang Fu
1. School of Computer Science and Engineering, Xi'an Technological University, Xi'an 710025, China; ;

Wang Liang
2. Key Laboratory of UAVs, Northwestern Polytechnical University, Xi'an 710025, China;

Chengtao Lu
1. School of Computer Science and Engineering, Xi'an Technological University, Xi'an 710025, China; ;

Dynamic Collision Optimization Algorithm Based on Ray Detection

Abstract

Abstract: Aiming at the inaccurate collision detection problem in Unity 3D-based visual simulation systems, a dynamic collision optimization algorithm based on ray detection is designed and implemented. The 3D virtual scene is segmented by octree, which simplifies the detection range of obstacles during the operation of simulation systems. At the same time, based on ray detection, according to the quantization coefficient and distance of the threat degree of obstacles, *an appropriate collision collider is dynamically added* to the obstacles to complete collision detection. The results show that the algorithm not only improves the fluency of visual simulation systems but also increases the accuracy of collision detection in visual simulation systems, and therefore solves the problem of inconsistency between the real environment and the visual simulation system.

Keywords

visual simulation, ray detection, collision detection, Unity 3D

Recommended Citation

Li Xing, Fu Yanfang, Wang Liang, Lu Chengtao. Dynamic Collision Optimization Algorithm Based on Ray Detection[J]. Journal of System Simulation, 2019, 31(11): 2393-2401.

第 31 卷第 11 期 系统仿真学报© Vol. 31 No. 11
2019 年 11 月 Journal of System Simulation Nov., 2019

基于射线检测的动态碰撞优化算法

李星[1]，傅妍芳[1]，王亮[2]，陆承涛[1]
(1. 西安工业大学计算机科学与工程学院，陕西 西安 710025；
2. 西北工业大学无人机重点实验室，陕西 西安 710025)

摘要：针对基于 Unity 3D 的视景仿真系统中的碰撞检测不准确问题，设计并实现了一种基于射线检测的动态碰撞优化算法。通过八叉树分割三维虚拟场景，简化并缩小了仿真系统运行时对障碍物的检测范围；同时*基于射线检测，根据障碍物的威胁程度量化系数和距离动态地为障碍物添加合适的碰撞器*，完成碰撞检测。结果表明：在提升视景仿真系统画面流畅度的同时，该算法提高了视景仿真系统中碰撞检测的准确性，解决了真实环境与视景仿真系统碰撞效果不一致的问题。

关键词：视景仿真；射线检测；碰撞检测；Unity 3D
中图分类号：TP391 文献标识码：A 文章编号：1004-731X (2019) 11-2393-09
DOI: 10.16182/j.issn1004731x.joss.19-FZ0353

Dynamic Collision Optimization Algorithm Based on Ray Detection

Li Xing[1], *Fu Yanfang*[1], *Wang Liang*[2], *Lu Chengtao*[1]
(1. School of Computer Science and Engineering, Xi'an Technological University, Xi'an 710025, China;
2. Key Laboratory of UAVs, Northwestern Polytechnical University, Xi'an 710025, China)

Abstract: Aiming at the inaccurate collision detection problem in Unity 3D-based visual simulation systems, a dynamic collision optimization algorithm based on ray detection is designed and implemented. The 3D virtual scene is segmented by octree, which simplifies the detection range of obstacles during the operation of simulation systems. At the same time, based on ray detection, according to the quantization coefficient and distance of the threat degree of obstacles, *an appropriate collision collider is dynamically added* to the obstacles to complete collision detection. The results show that the algorithm not only improves the fluency of visual simulation systems but also increases the accuracy of collision detection in visual simulation systems, and therefore solves the problem of inconsistency between the real environment and the visual simulation system.

Keywords: visual simulation; ray detection; collision detection; Unity 3D

引言

碰撞检测技术是虚拟现实技术中三维视景仿真的关键技术，也是保障虚拟仿真真实性与准确性的必要条件。精确的碰撞检测可以很大程度地提高虚拟仿真系统的真实程度，增强虚拟视景的沉浸感；合理地进行碰撞检测可以避免出现虚拟场景中物体穿透或重叠等现象，因此碰撞检测具有广泛的应用背景和研究意义[1]。

针对无人机感知与规避视景仿真系统中无人机在虚拟城市以及虚拟山地场景中出现的碰撞检测不准确、高速飞行物体穿透场景模型等问题，设计并实现了一种基于射线检测的动态碰撞优化算

收稿日期：2019-05-13 修回日期：2019-07-19；
基金项目：民机专项(MJ-2016-F-18)，西安市科技计划高校院所人才服务企业工程类项目(2017075CG/RC038(XAGY004))；
作者简介：李星(1995-)，女，陕西宝鸡，硕士生，研究方向为网络仿真与建模。

法。通过对遇到问题产生原因的研究与分析，提出利用八叉树对场景进行空间分割，简化空间障碍物检测个数，并用射线检测实时判断当前无人机周围的障碍物类型及距离，通过计算障碍物的威胁系数，为障碍物动态地添加合适的包围盒进行碰撞检测，解决了无人机感知与规避视景仿真系统中遇到的相关问题。

本文做了多组验证实验来检验算法的效率及有效性。通过对实验中多个信息数据的对比表明，基于射线检测的动态碰撞优化算法具有良好的仿真碰撞检测效果，对当前复杂度较高的三维场景具有较好的适用性。

1 相关研究综述

无人机感知与规避视景仿真系统的场景中含有大量几何模型，特别是城市场景中有大量三维建筑物模型。这些模型由大量三角面片构成，几何关系复杂，大大提高了碰撞检测的计算复杂度[2]。而无人机与各场景之间的碰撞检测需要实时完成，因此快速且精准的碰撞检测算法对于提高感知与规避仿真场景的真实性，增强虚拟场景的沉浸感有着至关重要的作用。

目前研究中多用空间分割法和层次包围盒法来进行碰撞检测。空间分割法主要包括均匀分割法、二叉空间分割树法、八叉树法等，应用于环境稀疏且物体分布均匀的场景。均匀分割法是将整个虚拟空间及场景中的物体沿坐标轴划分成等体积的规则单元格，该算法在三维物体间进行碰撞检测时都需要确定其所在的空间单元，只对同一个空间节点或邻域节点单元中的模型进行精确的碰撞检测，避免了大量不必要的计算[3]。层次包围盒法适用于复杂环境中的碰撞检测，典型的包围盒有轴对齐包围盒、方向包围盒、离散方向多面体包围盒和球体包围盒[4]。

上述已有的算法并不完全适用于无人机感知与规避仿真场景，原因如下：(1) 无人机仿真飞行时，与场景中大多数物体模型都不发生碰撞，若每个时间采样点都完全更新其层次包围盒树，将会产生不必要的时间损耗；(2) 城市场景中所含建筑物数量庞大，粗略检测阶段若选用相交测试、更新计算复杂的包围盒，会消耗大量时间及内存资源；(3) 场景中静态障碍物多且结构复杂，若对障碍物全部构造复杂的网格碰撞器，检测精度高但会极大降低系统性能；若对场景中全部障碍物构造简单的盒碰撞器，检测精度将不能满足仿真要求；(4) 无人机感知避障过程中，还需要实时检测场景中的其它无人机、热气球等动态障碍物，对于检测实时性及检测精度要求较高。

针对以上问题，利用八叉树分割空间，简化复杂场景中静态障碍物的粗略检测，利用射线检测提前预测近距离范围内障碍物类型及个数，通过计算并判断障碍物威胁系数动态地选择适当的碰撞器进行碰撞检测，在保证碰撞检测准确性的同时避免耗费大量不必要的资源。

2 基于射线检测的动态碰撞优化算法

2.1 问题描述

本文研究的问题描述如下：无人机感知与规避视景仿真系统是一个由真实无人机感知与规避数据驱动的视景仿真模拟系统。真实的无人机在真实环境中飞行，实时地传回真实飞行过程中的数据进行感知与规避计算，同时将数据转发给视景仿真系统，驱动仿真系统运行并准确地模拟显示出当前无人机飞行过程中的具体感知与规避过程，方便研究人员近距离地观测无人机感知与规避过程，研究无人机的感知与规避问题。

2.2 算法整体框架

在视景仿真的过程中，由于无人机速度较快以及真实环境与模拟环境存在的误差，导致视景仿真系统不能准确地模拟显示真实飞行过程，存在真实环境未碰撞到障碍物而视景系统已经发生碰撞，或真实环境发生碰撞而视景系统发生物体穿透等问题，极大地影响了视景仿真系统的实时性和准确

DOI: 10.16182/j.issn1004731x.joss.19-FZ0353

性，为此本文提出一种基于射线检测的动态碰撞检测算法。

本文算法主要包含以下几个步骤：

(1) 城市和山地场景中的大部分障碍物为静态障碍物，如建筑物、山脉等，这些障碍物在软件运行的过程中，其位置、大小等基本参数不会发生改变。因此在程序开始时通过八叉树空间分割法将整个大场景中的所有对象进行预处理分割，且在之后的整个感知与规避视景仿真过程中，不需要实时地对该八叉树进行动态更新。

(2) 判断当前无人机所处的空间节点，只需进一步检测该模块中无人机是否与该节点中的静态障碍物发生碰撞。对这些静态障碍物统一添加盒碰撞器(Box Collider)，方便后续的射线检测。

(3) 由于感知与规避视景仿真系统中存在一些动态障碍物，在程序运行过程中随时可能与无人机发生碰撞，如果一直检测场景中所有动态障碍物的运行状态是不必要且耗费资源的，因此通过射线检测可以提前对碰撞做出预测。通过射线检测，获得无人机前、后、左、右、上、下的障碍物及其基本信息。

(4) 当检测到无人机周围有障碍物时，检测障碍物类型、速度和障碍物与无人机航向夹角，计算障碍物威胁系数，通过威胁系数和障碍物与无人机距离动态地选择是否需要将盒碰撞器转换为网格碰撞器(Mesh Collider)。

(5) 当无人机与障碍物距离大于某个值时，将障碍物上的网格碰撞器再转换回盒碰撞器；当无人机进入其它空间节点时，删除上一个空间节点里的所有碰撞器。

3 动态碰撞检测优化算法设计

3.1 八叉树空间分割法

八叉树(octree)是一种用于描述三维空间的树状数据结构，每个节点有 8 个子节点，一般中心点作为节点的分叉中心。如图 1 所示，把三维空间中的一个立方体均匀地分割为 8 个同样大小的小立方体，然后递归地分割出更小的立方体。这种分割方式可以得到比较规则的树状结构，从而使得空间中三维物体的查询变得高效[5]。

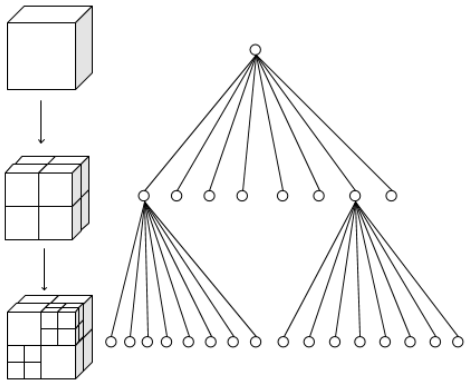

图 1 八叉树示意图
Fig. 1 Diagram of the octree

如果一个对象与某个节点处的立方体不相交，那么该对象与这个节点的节点子树下所有物体都不相交。从根节点开始，依次遍历八叉树的子节点，如遇到节点相交就继续往下遍历，如不相交就放弃遍历该子树，最后在叶节点进行形状与点的相交测试[6]。采用八叉树对场景进行初始化分割见图 2。

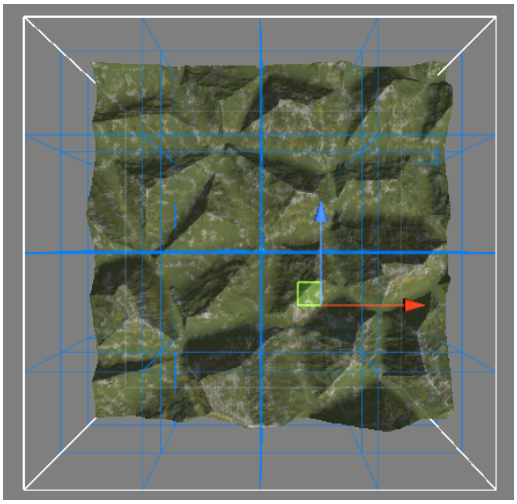

图 2 采用八叉树对场景进行分割
Fig. 2 Split the scene with an octree

具体实现时先初始化八叉树节点类，主要包括节点包围盒的尺寸和信息、节点当前深度、节点包含对象列表以及节点的子节点，其次写出八叉树的构造函数，再判断场景中对象是否与节点相交，递

归地将场景中的对象依次分割到相应子节点中，最后根据场景的大小决定八叉树的深度。

场景的分割完成后，图 3 所示为系统运行时，实时地检测无人机的当前坐标(图中中心店黄线上的绿色圆点为无人机)，判断其所处的八叉树节点空间，仅对该节点空间内的障碍物添加盒碰撞器以方便后续的射线检测，图中绿色区域为场景中不在当前无人机所处八叉树节点的障碍物，对这些障碍物会删除其上的碰撞器，减少场景中三角面片数，减少视景仿真系统的资源消耗。

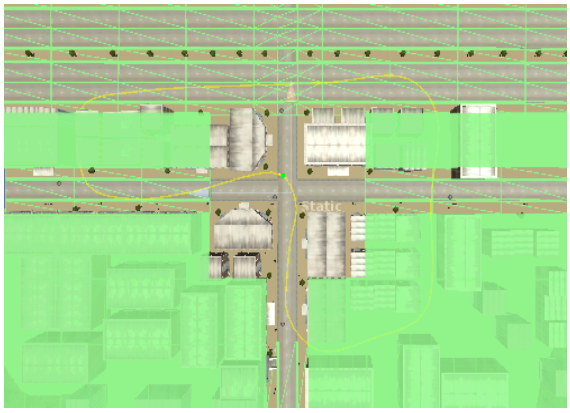

图 3　当前无人机所在的八叉树节点
Fig. 3　Octree node where currently the UAV is located

3.2 Unity3D 中的碰撞器

Unity3D 中的碰撞检测主要是通过给对象添加各种不同的碰撞器来实现，碰撞器是在碰撞包围盒的基础上，描述了物体可被碰撞的边界，并通过物理引擎模拟产生类似现实中的碰撞效果，达到碰撞检测的作用。

盒碰撞器是一种轴对称碰撞器，它以待测对象的重心为碰撞器的中心，以待测对象的最大边长为碰撞器边长，紧贴三维对象外边缘构造最小立方体。盒碰撞器能够很好地应用于如建筑物墙体、桥梁平面等规则立方体的碰撞检测，对于不规则对象的检测不够准确，只能用于粗糙碰撞检测。图 4 中绿色线框即为无人机的盒碰撞器，其形状规则但不能准确地覆盖在三维模型表面，易出现物体并未碰撞却已检测到物体碰撞的情况。

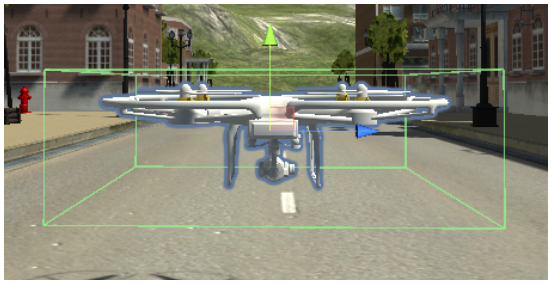

图 4　大疆无人机的盒碰撞器
Fig. 4　Box collider of Dajiang UAV

网格碰撞器是在三维模型的网格资源上构建的碰撞器，它严格按照目标三维对象的 Transform 属性来设定碰撞器自身的位置和大小比例。对于复杂模型的碰撞检测，网格碰撞器比盒碰撞器精确得多，但同时资源消耗也大得多[7]。碰撞网格使用背面剔除来删掉物体背面被遮挡住的多边形，以此来达到提高效率的目的[8]。大疆无人机添加网格碰撞器如图 5，根据大疆模型的网格信息添加的网格碰撞器可以准确地贴在模型表面，碰撞精度高，但是碰撞器线面多，碰撞计算量大。

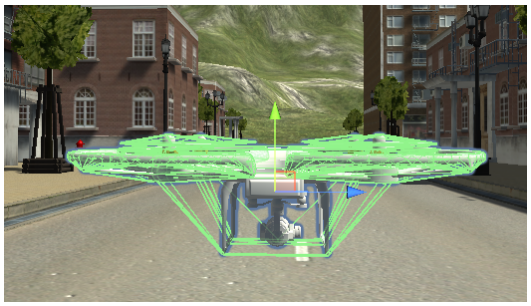

图 5　大疆无人机的网格碰撞器
Fig. 5　Mesh Collider of Dajiang UAV

通过比较可以看出，网格碰撞器较盒碰撞器而言更加贴近对象的实际边缘大小，因而可以获得更加精确的碰撞检测效果；但是网格碰撞器却多了很多节点，增加了计算，降低了性能，因此需要动态地决定为障碍物添加哪种碰撞器，在保证碰撞准确度的同时减少性能损耗。

3.3 射线检测

3.3.1 射线检测基本原理

射线检测，即利用射线来检测在该射线的路径

DOI: 10.16182/j.issn1004731x.joss.19-FZ0353

上是否和场景中的物体发生了碰撞。无论多么复杂的三维模型，都可以由若干个三角形组合而成。因此三维场景中的射线检测实际上就是判断该射线上的点是否与某个三角形相交[9]，只要检测到它们相交，则该点所在的射线与该三角形所在的三维模型相交。

最简单的方法是先判断射线是否与三角形所在平面相交，再判断射线与平面的交点是否在三角形内。但这种方法需要额外计算出三角形所在的平面[10]，效率并不很高，可以利用另一种方法来计算。主要求解过程如下。

射线的参数方程为：

$$\boldsymbol{O}+\boldsymbol{D}t \tag{1}$$

式中：$\boldsymbol{O}$ 为射线的起点；$\boldsymbol{D}$ 为射线的方向。

三角形的参数方程为：

$$(1-\boldsymbol{u}-\boldsymbol{v})\boldsymbol{V}_0+\boldsymbol{u}\boldsymbol{V}_1+\boldsymbol{v}\boldsymbol{V}_2$$

式中：$\boldsymbol{V}_0$，$\boldsymbol{V}_1$ 和 $\boldsymbol{V}_2$ 为三角形的 3 个顶点；$\boldsymbol{u}$，$\boldsymbol{v}$ 为 $\boldsymbol{V}_1$ 和 $\boldsymbol{V}_2$ 的权重；$1-\boldsymbol{u}-\boldsymbol{v}$ 为 $\boldsymbol{V}_0$ 的权重，并且满足 $\boldsymbol{u}\geqslant 0$，$\boldsymbol{v}\geqslant 0$，$\boldsymbol{u}+\boldsymbol{v}\leqslant 1$。

三角形内任意一点都可以理解为从顶点 $\boldsymbol{V}_0$ 开始，沿着边 $\boldsymbol{V}_0\boldsymbol{V}_1$ 移动一段距离，然后再沿着边 $\boldsymbol{V}_0\boldsymbol{V}_2$ 移动一段距离，如图 6 所示，移动的距离由参数 $\boldsymbol{u}$ 和 $\boldsymbol{v}$ 控制，然后求它们的和向量[11]。

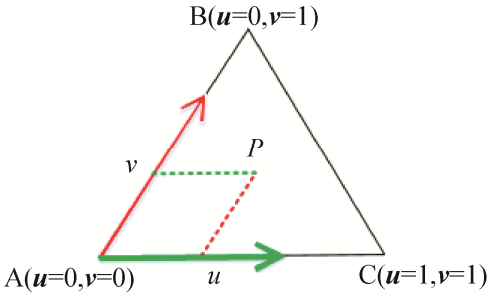

图 6 三角形内任意点表示图示

Fig. 6 Representation of a point in the triangle

于是，求射线与三角形的交点也就变成求解如下方程，其中 $\boldsymbol{t}$，$\boldsymbol{u}$，$\boldsymbol{v}$ 是未知数，其它都是已知的。

$$\boldsymbol{O}+\boldsymbol{D}t=(1-\boldsymbol{u}-\boldsymbol{v})\boldsymbol{V}_0+\boldsymbol{u}\boldsymbol{V}_1+\boldsymbol{v}\boldsymbol{V}_2 \tag{2}$$

对式(2)进行整理，将 $\boldsymbol{t}$，$\boldsymbol{u}$，$\boldsymbol{v}$ 提取出来作为未知数，得到下面的线性方程组：

$$\begin{bmatrix}-\boldsymbol{D} & \boldsymbol{V}_1-\boldsymbol{V}_0 & \boldsymbol{V}_2-\boldsymbol{V}_0\end{bmatrix}\begin{bmatrix}\boldsymbol{t}\\ \boldsymbol{u}\\ \boldsymbol{v}\end{bmatrix}=\boldsymbol{O}-\boldsymbol{V}_0 \tag{3}$$

利用克莱姆法则，令 $\boldsymbol{E}_1=\boldsymbol{V}_1-\boldsymbol{V}_0$，$\boldsymbol{E}_2=\boldsymbol{V}_2-\boldsymbol{V}_0$，$\boldsymbol{T}=\boldsymbol{O}-\boldsymbol{V}_0$，可得到 $\boldsymbol{t}$，$\boldsymbol{u}$，$\boldsymbol{v}$ 的解分别是：

$$\boldsymbol{t}=\frac{1}{\left|-\boldsymbol{D}\quad \boldsymbol{E}_1\quad \boldsymbol{E}_2\right|}\left|\boldsymbol{T}\quad \boldsymbol{E}_1\quad \boldsymbol{E}_2\right|;$$

$$\boldsymbol{u}=\frac{1}{\left|-\boldsymbol{D}\quad \boldsymbol{E}_1\quad \boldsymbol{E}_2\right|}\left|-\boldsymbol{D}\quad \boldsymbol{T}\quad \boldsymbol{E}_2\right|;$$

$$\boldsymbol{v}=\frac{1}{\left|-\boldsymbol{D}\quad \boldsymbol{E}_1\quad \boldsymbol{E}_2\right|}\left|-\boldsymbol{D}\quad \boldsymbol{E}_1\quad \boldsymbol{T}\right|。$$

将这 3 个解联合起来是：

$$\begin{bmatrix}\boldsymbol{t}\\ \boldsymbol{u}\\ \boldsymbol{v}\end{bmatrix}=\frac{1}{\left|-\boldsymbol{D}\quad \boldsymbol{E}_1\quad \boldsymbol{E}_2\right|}\begin{vmatrix}\boldsymbol{T} & \boldsymbol{E}_1 & \boldsymbol{E}_2\\ -\boldsymbol{D} & \boldsymbol{T} & \boldsymbol{E}_2\\ -\boldsymbol{D} & \boldsymbol{E}_1 & \boldsymbol{T}\end{vmatrix} \tag{4}$$

根据混合积公式：

$$\left|a\quad b\quad c\right|=a\times b\bullet c=-a\times c\bullet b \tag{5}$$

式(5)可以改写成：

$$\begin{bmatrix}\boldsymbol{t}\\ \boldsymbol{u}\\ \boldsymbol{v}\end{bmatrix}=\frac{1}{\left[\boldsymbol{D}\times\boldsymbol{E}_2\bullet\boldsymbol{E}_1\right]}\begin{bmatrix}\boldsymbol{T}\times\boldsymbol{E}_1\bullet\boldsymbol{E}_2\\ \boldsymbol{D}\times\boldsymbol{E}_2\bullet\boldsymbol{T}\\ \boldsymbol{T}\times\boldsymbol{E}_1\bullet\boldsymbol{D}\end{bmatrix}$$

令 $\boldsymbol{P}=\boldsymbol{D}*\boldsymbol{E}_2$，$\boldsymbol{Q}=\boldsymbol{T}*\boldsymbol{E}_1$，得到最终的计算公式为：

$$\begin{bmatrix}\boldsymbol{t}\\ \boldsymbol{u}\\ \boldsymbol{v}\end{bmatrix}=\frac{1}{\left|\boldsymbol{P}\bullet\boldsymbol{E}_1\right|}\begin{vmatrix}\boldsymbol{Q}\bullet\boldsymbol{E}_2\\ \boldsymbol{P}\bullet\boldsymbol{T}\\ \boldsymbol{Q}\bullet\boldsymbol{D}\end{vmatrix}$$

在 Unity 3D 中，与射线检测相关的类方法主要有射线类(Ray)和射线投射碰撞信息类(RaycastHit)。其中射线类用于生成一条确定世界坐标起点，确定矢量方向的射线；射线投射碰撞信息类用于存储发射射线后产生的碰撞信息，这些信息主要包括：与射线发生碰撞的碰撞器信息、从射线起点到射线与碰撞器的交点的距离、射线射入平面的法向量、射线与碰撞器交点的坐标(Vector3 对象)。通过方法 boolRaycast 从无人机的中心发射上、下、左、右、前、后 6 个方向上的射线，检测相应射线路径上是否存在障碍物。

3.3.2 目标威胁评估模型

本文主要根据现代战场上无人机的空防态势来建立目标威胁评估模型，采用包括障碍物类型、障碍物速度和障碍物与无人机航向夹角在内的 3 个主要评价指标建立目标威胁评估模型。这 3 个指标的具体含义为：

(1) 障碍物类型分两大类：动态障碍物和静态障碍物。其中动态障碍物包括空客 A320、空客 B737、大疆、飞艇、热气球等，静态障碍物分为建筑物、山脉、树木等。根据威胁程度 T 对其进行量化，量化结果如表 1 所示。

表 1 威胁程度量化表

Tab. 1 Threat level quantization table

障碍物类型		威胁程度 T
动态障碍物	A320	75
	B737	75
	大疆	45
	飞艇	30
	热气球	25
静态障碍物	建筑物	60
	山脉	65
	树木	20

(2) 障碍物速度 V：速度越快，威胁程度越大；速度越小，威胁度越小。

(3) 障碍物与无人机航向夹角：即为无人机的航向 forward 与无人机到障碍物位置连线的夹角 angle，该夹角的范围在 0~45°内时才会威胁到无人机运动。将其进行必要的转化，则夹角 A 越大，威胁程度越大；反之越小。

A=180–angle

由于视景系统中并不需要计算出精确的威胁系数，只需根据障碍物的威胁因子给出相应的计算，添加相应的碰撞器，参考文献[12]中给出的比例尺度的意义见表 2，定义威胁因子比例尺的标度值。

由表 2 得出各影响因子判断矩阵 $\boldsymbol{A}$-$\boldsymbol{B}$，如表 3 所示[12]。其中 B_1 表示障碍物威胁程度 T，B_2 表示障碍物速度 V，B_3 表示障碍物与无人机航向夹角 A。

根据表 3 计算出各影响因子的权重步骤如下：

step 1：$\boldsymbol{B}$ 的元素按行相乘：

$$U_{ij} = \prod_{j=1}^{n} b_{ij}$$

step 2：所得的乘积分别开 n 次方：

$$U_i = \sqrt[n]{U_{ij}}$$

step 3：将方根向量正规化，得到排序权向量 $\boldsymbol{W}$：

$$W_i = U_i \Big/ \left[\sum_{i=1}^{n} U_i\right]$$

表 2 比例尺度的意义[12]

Tab. 2 Significance of scale

标度值	定义
1	对 A 而言，B_i 与 B_j 同样重要
3	对 A 而言，B_i 比 B_j 稍为重要
5	对 A 而言，B_i 比 B_j 重要
7	对 A 而言，B_i 比 B_j 重要得多
9	对 A 而言，B_i 比 B_j 绝对重要
2，4，6，8	其重要程度介于上述两邻近比例尺度之间
倒数	因素 i 与 j 判断比较得 b_{ij}，因素 j 与 i 比较判断 $b_{ji}=1/b_{ij}$

表 3 各影响因子判断矩阵 A-B

Tab. 3 Judgment matrix A-B of each impact factor

A	B_1	B_2	B_3
B_1	1	7/3	3/2
B_2	3/7	1	5/4
B_3	2/3	4/5	1

根据上述 3 个步骤的计算，得到障碍物类型、障碍物速度、障碍物与无人机之间的距离、以及障碍物与无人机航向夹角的权向量分别为：$\boldsymbol{W}_1$=0.48，$\boldsymbol{W}_2$=0.26，$\boldsymbol{W}_3$=0.26。根据算出的权向量，得到目标威胁系数：

$$\beta = \boldsymbol{W}_1\mathrm{T} + \boldsymbol{W}_2\mathrm{V} + \boldsymbol{W}_3\mathrm{A}$$

通过目标威胁系数以及障碍物与无人机之间的距离选择是否将无人机周围障碍物的盒碰撞器转换为网格碰撞器。经过对无人机航速及仿真要求进行计算，当威胁系数大于 50 并且无人机与障碍物距离小于 300 m 时，将障碍物的盒碰撞器换为网格碰撞器；当两者距离大于 800 m 时，将障碍物的网格碰撞器转换为盒碰撞器。

4 实验结果及分析

对无人机和场景中障碍物添加不同的碰撞器，分为 5 种情况进行对比实验验证。这 5 种情况分别是：无人机和障碍物均无碰撞器；无人机添加网格碰撞器，障碍物无碰撞器；无人机添加网格碰撞器，障碍物添加盒碰撞器；无人机、障碍物均有网格碰撞器；根据本文优化算法动态添加碰撞器。图 7 所示为无人机和城市场景中障碍物均添加复杂的网格碰撞器的情况。图 8 为本文算法优化后的城市场景中碰撞器添加情况，无人机周围场景障碍物均动态添加了网格碰撞器，无人机所在八叉树节点中障碍物均添加了盒碰撞器。可以看出，本文的优化算法可以很好地检测无人机当前所处位置，并且动态地选择是否为无人机周围的障碍物更换碰撞检测效果更好的网格碰撞器。

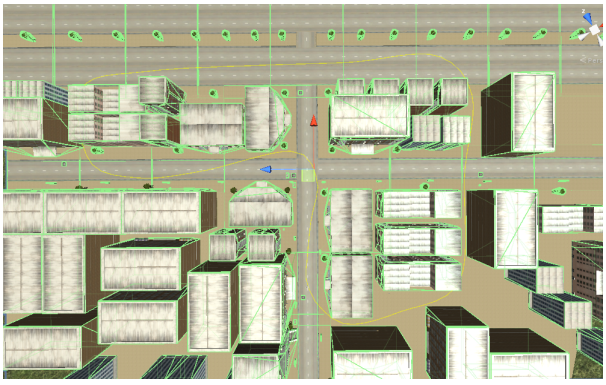

图 7　第 4 种实验情况下城市场景的碰撞器添加情况
Fig. 7　Collider addition in urban scenes in the fourth experimental situation

图 8　本文算法优化后的城市场景中碰撞器添加情况
Fig. 8　Collider addition in the urban scene optimized by the algorithm

可以通过 Unity 3D 的渲染数据统计窗口(Rendering Statistics Window)，来随时观测仿真系统运行过程中的渲染、声音、网络状况等多种统计信息，检测碰撞检测优化算法是否有效。本文用于实验对比的信息数据主要包括以下 4 个：

(1) FPS(Time per frame and Frame per second)：表示 Unity 引擎处理和渲染一帧所花费的时间，主要受到场景中要渲染的物体数量和 GPU 性能的影响。FPS 数值越高，虚拟场景显示、运行会更加平滑和流畅。一般来说，由于光在视网膜上停止作用后人眼还会将其保持在视网膜上 40 ms 左右的时间，因此视景仿真画面每秒帧数至少保证在 0 FPS 以上，才会让人感觉到该模拟视景是流畅的。

(2) Render thread：GPU 渲染线程处理图像所花费的时间，具体数值由 GPU 的性能来决定。

(3) Verts：摄像机视野(field of view)内渲染的三维模型的顶点总数。

(4) Tris：摄像机视野(field of view)内渲染的的三维模型的三角面总数量。

上述 5 种实验情况分别统计运行时的信息数据，结果如表 4 所示。

表 4　不同碰撞器添加情况下的信息数据
Tab. 4　Information data in the case of different collider additions

碰撞器添加情况	场景类型	FPS	Render thread/ms	Verts	Tris
对象均无碰撞器	城市	63.3	3.7	626.6 K	792.2 K
	山地	42.2	20.7	2.5 M	3.1 M
无人机有网格碰撞器，障碍物无碰撞器	城市	44.6	8.1	2.1 M	2.5 M
	山地	41.3	4.2	3.3 M	4.4 M
无人机有网格碰撞器，障碍物均有盒碰撞器	城市	22.0	13.1	6.1 M	8.0 M
	山地	75.9	0.9	134.1 K	185.2 K
无人机、障碍物均有网格碰撞器	城市	42.5	19.9	2.6 M	3.3 M
	山地	26.6	25.5	3.9 M	5.2 M
本文优化算法	城市	49.7	18.8	2.4 M	3.0 M
	山地	71.8	1.2	207.2 K	301.8 K

由表 4 可知，对于同一个场景，添加不同的碰撞器，对于视景的影响差别很大。当为所有无人机及障碍物均添加碰撞器时，视景内顶点个数和三角形面片数最高达到 8 M，FPS 变为 22.0，视景画面已经开始卡顿，极大地影响了视景体验。通过优化算法，场景中每个对象仍然都有碰撞器，并且可以保证较为精确的碰撞检测，同时可以保持平均在 60 FPS，视景画面流畅，可以达到视景要求。

总之，本文的基于射线检测的动态碰撞优化算法较传统的碰撞检测方式而言，较大地减少了场景中模型的顶点数和三角形面片数，在提高视景仿真的平滑性与流畅性的同时，解决了视景仿真系统中遇到的碰撞检测不准确和高速飞行器穿透障碍物等问题，保证了仿真系统碰撞检测的准确性。

5 结论

针对无人机感知与规避视景仿真系统存在的真实环境未碰撞到障碍物而视景系统已经发生碰撞，以及真实环境发生碰撞而视景系统发生物体穿透等问题，本文提出了一种基于射线检测的动态碰撞检测优化算法。首先利用八叉树剔除复杂环境中不必要检测的障碍物，其次对各障碍物进行威胁等级量化评估，根据障碍物的威胁等级动态地为其添加合适的碰撞器进行碰撞检测。通过进一步分析无人机与障碍物距离在 300 m 和 800 m 时对碰撞检测产生的影响，得到本文的算法可以有效地解决在视景仿真系统中遇到的问题，使得视景仿真系统可更加完美地配合真实数据实现仿真，极大地方便了研究人员对于无人机感知与规避算法的观测与研究。

参考文献：

[1] 唐源皓. 基于质点转换和包围盒的混合碰撞检测算法的研究与应用[D]. 成都: 西南交通大学, 2018.
Tang Yuanhao. Research and application of hybrid collision detection algorithm based on particle transformation and bounding box[D]. Chengdu: Southwest Jiaotong University, 2018.

[2] 于凌涛, 王涛, 宋华建, 等. 面向虚拟手术的碰撞检测优化算法[J]. 哈尔滨工程大学学报, 2014, 35(9): 1164-1170.
Yu Lingtao, Wang Tao, Song Huajian, et al. Collision Detection Optimization Algorithm for Virtual Surgery[J]. Journal of Harbin Engineering University, 2014, 35(9): 1164-1170.

[3] 景乾峰, 神和龙, 尹勇, 等. 航海模拟器中的岸线实时碰撞检测[J]. 系统仿真学报, 2018, 30(7): 2682-2699.
Jing Qianfeng, Shen Helong, Yin Yong, et al. Shoreline real-time collision detection in navigation simulator[J]. Journal of System Simulation, 2018, 30(7): 2682-2699.

[4] 郭娇娇. 基于动捕信息的虚实交互技术研究[D]. 北京: 北方工业大学, 2017.
Guo Jiaojiao. Research on virtual and real interaction technology based on dynamic capture information [D]. Beijing: North China University of Technology, 2017.

[5] 王振文. 复杂场景中不规则物体的碰撞检测算法设计与实现[D]. 北京: 北京化工大学, 2017.
Wang Zhenwen. Design and implementation of collision detection algorithm for irregular objects in complex scenes[D]. Beijing: Beijing University of Chemical Technology, 2017.

[6] 李辰龙. 虚拟现实系统中人机交互技术研究[D]. 杭州: 浙江大学, 2017.
Li Chenlong. Research on human-computer interaction technology in virtual reality system[D]. Hangzhou: Zhejiang University, 2017.

[7] 张晓曦, 尹勇, 梁民仓. 蛟龙号下潜及水下作业过程的交互仿真开发[J]. 系统仿真学报, 2018, 30(7): 2715-2721.
Zhang Xiaoxi, Yin Yong, Liang Mincang. Interactive Simulation of the Dive and Underwater Operation Process of the Jiaolong[J]. Journal of System Simulation, 2018, 30(7): 2715-2721.

[8] 尚华强. 基于 Kinect 的虚拟人物动作仿真研究[D]. 杭州: 杭州电子科技大学, 2013.
Shang Huaqiang. Research on virtual character motion simulation based on Kinect[D]. Hangzhou: Hangzhou University of Electronic Science and Technology, 2013.

[9] 张培承. 基于 KD-tree 的并行光线跟踪算法在 CPU/GPU 异构平台中的研究[D]. 上海: 上海大学, 2017.
Zhang Peicheng. Research on parallel ray tracing algorithm based on KD-tree in CPU/GPU heterogeneous platform[D]. Shanghai: Shanghai University, 2017.

[10] 石敏, 王俊铮, 毛天露, 等. 基于物理的可交互性虚拟服装动画模拟[J]. 系统仿真学报, 2018, 30(7): 2583-2592.

Shi Min, Wang Junzheng, Mao Tianlu, et al. Virtually animated animation simulation based on physics[J]. Journal of System Simulation, 2018, 30(7): 2583-2592.

[11] 郭晨光. 基于光线跟踪的高真实感红外三维场景仿真方法研究[D]. 西安: 西安电子科技大学, 2013.
Guo Chenguang. Research on high-realistic infrared three-dimensional scene simulation method based on ray tracing[D]. Xi'an: Xidian University, 2013.

[12] 李季，孙秀霞，马强. 无人机对空威胁算法与仿真[J]. 系统仿真学报, 2008, 20(16): 4237-4243.
Li Ji, Sun Xiuxia, Ma Qiang. Algorithm and Simulation of UAV's Airborne Threat[J]. Journal of System Simulation, 2008, 20(16): 4237-4243.