

## 基于分类遍历的碰撞检测优化算法

孙劲光<sup>1</sup>, 吴素红<sup>2\*</sup>

(1. 辽宁工程技术大学 电子与信息工程学院, 辽宁 葫芦岛 125105; 2. 辽宁工程技术大学 研究生院, 辽宁 葫芦岛 125105)

(\*通信作者电子邮箱 wusuhong126@126.com)

**摘要:**针对现有层次树遍历方法的低效率问题,提出了一种基于分类遍历的碰撞检测算法。首先根据两个物体树中节点的平衡因子差值来将所有的物体对进行分类:结构相似的,采用同步下降遍历方法;结构不相似的,采用交换下降遍历方法,这减少了相交测试的次数。然后加入时空相关性和优先级策略优化遍历过程。最后通过实验结果表明,相比基于统一遍历的碰撞检测算法,该算法缩短了相交测试的时间,物体数目越多,快速性优势越显著,大约可以缩减所需时间的1/5。

**关键词:**碰撞检测;层次包围盒;分类遍历;深度优先;物体结构;时空相关性

**中图分类号:** TP391.9; TP301.6 **文献标志码:** A

### Collision detection optimization algorithm based on classified traversal

SUN Jinguang<sup>1</sup>, WU Suhong<sup>2\*</sup>

(1. School of Electronic and Information Engineering, Liaoning Technical University, Huludao Liaoning 125105, China;

2. Graduate School, Liaoning Technical University, Huludao Liaoning 125105, China)

**Abstract:** To solve the problem that present traversal methods of hierarchical tree which lead to low efficiency, a new collision detection algorithm based on classified traversal was proposed. Firstly, these objects were classified according to the difference between the balance factors of two tree's nodes. The simultaneous depth-first traversal method was applied to the objects which have similar structure, and the commutative depth-first traversal method was applied to the other objects, which reduced the number of intersect tests. Then, the process of traversal was optimized by using the temporal spatial coherence and priority strategy. Finally, the experimental results show that, compared with the collision detection algorithm based on unified traversal, the proposed algorithm shortens the time of the intersection test. The bigger the number of objects, the more significant the advantage of quickness, it can reduce about 1/5 of the required time.

**Key words:** collision detection; hierarchical bounding box; classified traversal; depth-first; object structure; temporal spatial coherence

## 0 引言

在虚拟环境中,会有多个物体发生运动,它们不可能相互穿越。碰撞检测就是用来检测各物体在某一时刻是否占据了同一空间从而避免它们相互穿越的。这项技术多年来一直备受国内外专家学者的关注,现已形成了两类经典的算法:空间分割法和层次包围盒法。空间分割法<sup>[1]</sup>是将虚拟空间划分成一个个小区域,只对同处于一个区域的物体进行碰撞检测;而层次包围盒法实现起来更简单,并且检测更精准,所以更常用。它的核心思想是用体积略大而几何特性简单的包围盒来代替复杂的几何对象,通过构造层次树并对其进行遍历来确定物体的相交状态。现已形成了包围球(sphere)层次树<sup>[2]</sup>、轴对齐包围盒(Axis Aligned Bounding Box, AABB)层次树<sup>[3]</sup>、任意方向包围盒(Oriented Bounding Box, OBB)层次树<sup>[4-5]</sup>、离散有向多面体(K-Discrete Orientation Polytopes, K-DOP)层次树<sup>[6]</sup>等经典的层次包围盒树。

在层次包围盒算法中,需要按某一下降规则遍历来确定两个包围盒层次树的相交状态。下降规则是相交测试的核心

部分,它的好坏直接影响了相交测试的效率。早期,算法大多采用单一下降的规则<sup>[7]</sup>,虽然实现简单,但是相交测试的次数却很多;后期,出现了同时下降规则<sup>[8-9]</sup>,它一定程度上减少了节点间的相交测试次数,但有时却不能有效地缩减搜索空间。这两种遍历方法都没有考虑到物体对间结构的不同,而统一采用一种下降规则,这可能会降低相交测试的效率。

本文考虑待检测的物体对间结构的差异,提出了一种基于分类的深度优先遍历方法。算法将物体对分为两类:第一类为结构相似的,采用同步下降深度优先遍历方法;第二类为结构相差大的,采用交换下降深度优先遍历方法。此外,引入时间相关性的思想,使用优先级策略,对遍历过程进行优化,从而整体提高遍历的效率。

## 1 基于统一遍历的包围盒碰撞检测算法

基于统一遍历的包围盒碰撞检测算法,就是对所有待检测的物体对采用同一种遍历方法。常用的遍历方法有单一下降深度优先遍历和同步下降深度优先遍历方法。前者是先下降其中一棵树至叶节点,然后再下降另外一棵树;后者是同时

收稿日期:2014-08-15;修回日期:2014-09-12。 基金项目:国家科技支撑计划项目(2013BAH120f00)。

作者简介:孙劲光(1962-),女,辽宁阜新,教授,博士,主要研究方向:数据挖掘、图形和图像处理、人脸识别; 吴素红(1989-),女,辽宁沈阳人,硕士研究生,主要研究方向:计算机图形学。

下降两棵树。

现假设有两棵包含  $n$  个叶节点的层次树  $a$  和  $b$ , 它们的叶节点相交了, 但是 2 个物体却未真正地碰撞。那么采用单一下降深度优先遍历方法将会执行  $2^{n-1} - 1$  次内部节点与节点间的相交测试和  $(2n - 1)2^{n-1}$  次叶子节点与节点、节点与叶节点、叶节点与叶节点间的相交测试, 累计共  $2^{2n-1} - 1$  次相交测试; 而同步下降深度优先遍历将会执行  $(2^{2(n-1)} - 1)/3$  次内部节点间的相交测试和  $2^{2(n-1)}$  次叶节点与节点、节点与叶节点、叶节点与叶节点间的相交测试, 累计共  $(2^{2n} - 1)/3$  次相交测试。当  $n$  趋近于无穷大, 后者的测试量仅为前者的  $2/3$ ; 并且单一下降规则并不太好, 若先下降  $a$  (或  $b$ ) 的叶子节点几乎都与  $b$  (或  $a$ ) 相交, 将导致  $b$  (或  $a$ ) 的递归次数为  $a$  (或  $b$ ) 的叶子节点数, 那么结构  $a$  (或  $b$ ) 的设计就没有太大意义了。同步下降规则也有其局限性, 它虽然能够快速地下降到叶节点; 但是对于尺寸和结构相差大的  $a$  和  $b$  而言, 同步下降遍历却不能有效地缩减搜索空间。

综上所述, 物体的结构对遍历方法的有效性是有一定影响的。所以本文根据物体对间结构的差别, 采用不同的深度优先遍历方法, 避免上述下降规则所产生的问题, 使得遍历更高效, 碰撞检测的实时性更高。

## 2 基于分类遍历的包围盒碰撞检测算法

### 2.1 树的度数

在包围盒碰撞检测算法中, 为了确定物体间的相交状态就会构造物体的层次树结构, 这就会涉及树度数的选择问题。树的深度和度数是成反比的。较高度数的树的深度一般较小, 从树根遍历到叶节点的步数会少, 而深度大的树的度数一般较高, 当前访问节点的子节点数就会增多。考虑两者之间的关系, 本文选择二叉树层次结构。因为二叉树优于  $d$  叉树<sup>[10-11]</sup>。当给定  $n$  个叶子节点的平衡  $d$  叉树, 采用单一下降与同步下降遍历方法的消耗分别与  $f_1(d) = d \log_d(n)$  和  $f_2(d) = d^2 \log_d(n)$  成正比, 当  $d = 2.71$  时, 前者有最小值; 当  $d = 2$  时, 后者有最小值, 优先选择  $d = 2$ 。除此之外, 二叉树也易于构造和遍历。

### 2.2 物体对在结构上的分类

碰撞检测时需要将两两为一组的物体对进行碰撞检测。因为每对物体都有各自的特点, 所以若将待检测的所有物体对在结构上进行分类, 然后分别对其采用不同的遍历方法, 将会减少遍历的时间, 提高检测效率。

二叉树节点上的平衡因子定义为该节点的左子树的深度减去它的右子树的深度<sup>[12]</sup>。因为在构造包围盒树的时候, 会把具有对齐性质的轴作为分裂轴, 物体会相对等体积地被分为左右两份, 若左右平衡程度相差大, 这就表示某一侧的三角面片数偏多, 这一侧的细节程度更大。这样, 可以通过两棵树相对应节点的平衡因子的差的绝对值来衡量这两棵树在结构上的差别。若所有对应上的节点的平衡因子差值的绝对值都不大于 1, 就认为两物体的细节程度相似, 在结构上相差不大; 否则两物体的细节程度相差大, 在结构上不相似。

分类公式  $C_k$  如下所示:

$$C_k = |B_{ai} - B_{bj}|; k = 1, 2, \dots, n$$

其中:  $B_{ai}$  表示树  $a$  的第  $i$  个节点的平衡因子,  $B_{bj}$  表示树  $b$  上与节点  $ai$  对应的节点  $bj$  的平衡因子,  $n$  为树  $a$  与树  $b$  对应上的节点个数。若  $C_1$  至  $C_n$  都不大于 1, 该物体对被分为第一类: 结构相似; 否则,  $C_1$  至  $C_n$  有其一大于 1, 该物体对就被分为第二类: 结构相差大。

如图 1 ~ 2 所示, 图 1(a)、(b)、(c) 分别是  $a$ 、 $b$ 、 $c$  三个物体的树结构, 它们的节点值为各节点的平衡因子; 图 2(a)、(b) 分别是树  $a$  与树  $b$ 、树  $a$  与树  $c$  平衡因子差值树, 各节点值为两棵树相对应节点的平衡因子差的绝对值。通过图 1 ~ 2 可知, 树  $a$  与树  $b$  结构上不相似, 树  $a$  与树  $c$  结构上相似。也就是说树  $a$  相对于树  $b$  有细节部分。

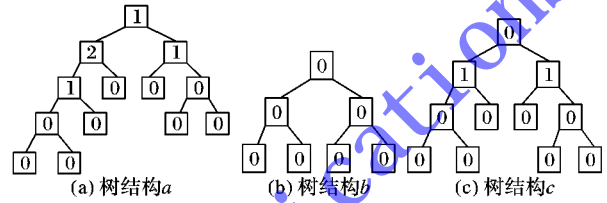


图1 树结构  $a$ 、 $b$ 、 $c$

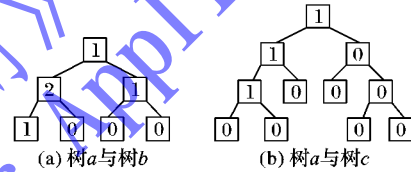


图2 各树间平衡因子差值树

待检测物体对的分类结论如下:

- 1) 结构上相似的, 采用同步下降深度优先遍历方法。
- 2) 结构上不相似, 采用交换下降深度优先遍历方法。

交换下降深度优先遍历方法的具体说明如下:

首先下降尺寸较大的树结构  $a$ , 这是因为树  $a$  可能仅有一小部分与尺寸小的树  $b$  接触, 测试过程中能尽快对大树  $a$  进行剪枝。在这里, 以树根的包围盒的体积作为物体的尺寸。

然后下降到树  $a$  的细节部分时, 交换下降对象, 对尺寸较小的树  $b$  执行下降操作。同样地, 下降到树  $b$  的细节部分大于树  $a$  的细节部分时, 再次交换下降对象, 对树  $a$  进行下降, 如此反复, 直到遍历过程结束。也就是说, 当下降到平衡因子差值大于 1 的节点时, 转而下降另一棵树。

针对结构上相似的物体对, 同步下降深度优先遍历方法能够更快地遍历到叶节点, 节省遍历的时间, 而结构上不相似的物体对, 采用一种改进的单一下降 (交换下降) 深度优先遍历方法。它能够避免单一下降深度优先遍历中递归次数多的问题。

## 3 算法的优化

### 3.1 时空相关性

在虚拟环境中进行碰撞检测过程中, 时间采样点往往很密集, 这就导致相邻采样点的物体状态和位置变化很小。也就是说在上一时间采样点, 如果两个物体碰撞了, 那么在下个时间采样点物体也很有可能是碰撞的, 并且碰撞的位置是相近的; 如果两个物体没碰撞, 那么在下个时间采样点也很可能不发生碰撞<sup>[13]</sup>, 这就是时空相关性在碰撞过程上的体现。

引入时空相关性的思想, 可以优化遍历过程。根据上一

时间采样点的遍历情况,记录一些关键节点,在下一个时间采样点就可以不必从树根开始向下遍历,而是从这些关键节点开始向下遍历,减少重复的遍历路径。关键节点<sup>[14]</sup>包括未发生碰撞的内部节点和遍历到的叶节点。关键节点所包含的子树的交集是空集,并且并集是全集,这就保证了整个过程是完整、正确的,并且没有冗余的遍历路径。

下面通过一个实例来说明时空相关性对遍历的优化效果。一个用二叉树表示的对象在两个相邻时间采样点( $t, t+1$ 时刻)与其他对象发生碰撞的情况如图3~4所示。其中:灰色节点表示与其他对象发生了碰撞,白色节点表示与其他对象未发生碰撞,右侧 list 表存储当前采样点的关键节点。

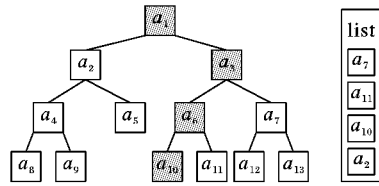


图3  $t$ 时刻物体的碰撞情况以及对应的关键点表

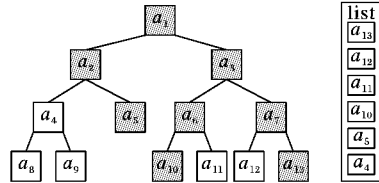


图4  $t+1$ 时刻物体的碰撞情况以及对应的关键点表

根据图4所示的 $t+1$ 时刻物体的碰撞情况,如果按传统的遍历过程( $a_1, a_2, a_4, a_5, a_3, a_6, a_{10}, a_{11}, a_7, a_{12}, a_{13}$ ),将执行11次相交测试。如果有了 $t$ 时刻的list表,就可以从关键节点开始向下遍历( $a_2, a_4, a_5, a_{10}, a_{11}, a_7, a_{12}, a_{13}$ ),将简化执行8次相交测试。包围盒树的层数越高,简化越明显,时空相关性对遍历过程的优化效果越好。

### 3.2 优先级策略

虚拟环境中存在有 $n$ 个物体,将所有物体两两组合进行碰撞检测,称为一轮碰撞检测,若1s要进行 $m$ 轮碰撞检测,那么每秒检测总次数为 $C_n^2 \times m$ 。随着物体数量的增多,系统每秒钟的测试任务量将会越来越大,这难免会影响碰撞检测的实时性。

为了及时检测出物体间的相交状态,本文制定一个物体对间的检测优先级策略,将两两组合的物体对进行优先级的排序,按次序对它们进行检测,而超低优先级的物体对可不进行检测。

物体对的优先级大小通过优先级函数 $F$ 来确定:

$$F = \begin{cases} 2, & b = 1 \\ 1 + v/s, & b = 0, 2s/(l_1 + l_2) \leq 1 \\ v/s, & b = 0, 2s/(l_1 + l_2) > 1 \end{cases}$$

其中: $b$ 表示上一个时间采样点物体是否碰撞, $b=1$ 表示碰撞上了, $b=0$ 表示没碰撞上; $v$ 表示上一个时间采样点两物体的速度,等于大上次与上次两物体最近距离的差值; $s$ 表示上一个时间采样点两物体最近点间的距离; $l_1$ 和 $l_2$ 分别为两物体的平均宽度。

函数 $F$ 的取值范围为 $[2, 0]$ 。 $F=2$ 时,表示两个物体在上个时间采样点碰撞了,这个时间采样点必须进行检测,所以优先级最高; $1 \leq F < 2$ 表示上个时间采样点没碰撞上,但它们

相距很近; $F < 1$ 表示上个时间采样点没碰撞上,并且它们相距还很远,所以优先级最低。在这里,可以设定一个阈值,低于一定优先级的物体不进行检测,为了避免漏检,如果一对物体超过一定次数未检测,就必须对它们进行检测。

### 3.3 三节点的二叉树

在计算机体系结构中,执行时间往往受限于与内存访问数据相关的缓存机制<sup>[15]</sup>。本文采用缓存友好的三节点二叉树结构来取代单节点二叉树结构,这将会减少相应的内存占有量。

单节点二叉树结构和三节点二叉树结构如图5~6所示。

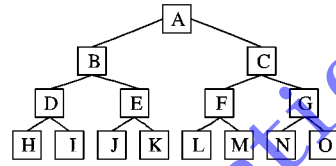


图5 4层单节点的二叉树结构

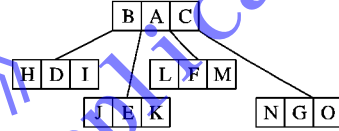


图6 2层三节点的二叉树结构

对于前者,需要存储14个指针和15个节点;而后者只需存储4个指针和4个节点。当物体越复杂,后者对内存的缩减量也就更明显,并且后者可以对前层数据实现缓存。

## 4 实验结果与分析

本实验程序是在具有Windows 7系统,CPU 2.2 GHz,四核处理器的PC机上应用VS2010和OpenGL进行开发的。为了说明本算法的有效性和优越性,本文做了两个实验。

实验1 同一场景下不同遍历方法的比较。

本实验是在同一场景下,统一采用轴对齐包围盒(Axis Aligned Bounding Box, AABB),分别采用单一下降遍历方法、同步下降遍历方法和分类下降遍历方法来实现碰撞检测,统计了不同遍历方法下运行1000步的平均碰撞检测时间(ms)和帧频(单位为fps(frame per second)),并且验证时空相关性和优先级策略对基于分类下降遍历碰撞检测算法的优化效果。场景如图7所示,环境中有30个大小不一、形状各异的物体进行随机运动,三角面片多达5000个。实验结果如表1所示。

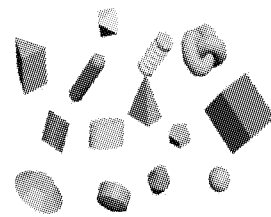


图7 碰撞场景

从表1可以看出,采用分类遍历方法的碰撞检测算法的平均碰撞检测时间比采用单一下降遍历方法和同步下降遍历方法的碰撞检测时间要小,并且每秒的帧数也略有提高,在加入时间相关性和优先级策略后,算法又得到了进一步的优化,



平均碰撞检测时间缩小到 40.314 ms,场景绘制速度达到了 39.51 frame/s。

表1 各种算法在碰撞检测时间和帧频上的对比

碰撞检测算法	平均碰撞检测时间/ms	帧频/fps
单一下降遍历方法	56.347	25.13
同步下降遍历方法	54.283	30.36
分类遍历方法	47.641	34.47
分类遍历优化方法(本文)	40.314	39.51

实验2 不同场景下不同分类结果的比较。

本实验是在是在4个不同场景下,分别采用分类遍历的碰撞检测算法和最常用的同步遍历碰撞检测算法,通过统计它们的平均碰撞检测时间(ms)来说明不同分类结果(有几对物体是相似的,有几对物体是不相似的)对实验效果的影响。实验结果如表2所示,其中: $N_{\text{like}}$ 为结构相似的物体对个数, $N_{\text{unlike}}$ 为结构不相似的物体对个数。

表2 不同场景下的算法比较

不同场景下的物体个数	基于同步遍历的碰撞检测算法	本文算法		
	平均碰撞检测时间/ms	$N_{\text{like}}$	$N_{\text{unlike}}$	平均碰撞检测时间/ms
10	38.748	29	16	35.274
15	40.239	40	65	36.108
20	44.837	58	132	38.231
30	54.472	122	313	40.314

从表2可知,本文算法的平均碰撞检测时间比基于同步遍历的碰撞检测算法的时间短很多。随着物体个数的增加,缩减的程度就更大,当物体个数为30时,平均碰撞检测的时间可以缩减14 ms。这证实了对物体对进行分类优化有效提高了碰撞检测的效率。

## 5 结语

本文对现有的层次树遍历方法进行了对比研究,提出了一种基于分类遍历的包围盒碰撞检测优化算法。算法将待检测的物体对按照结构的不同进行了分类,对不同类别的物体对采用不同的遍历方法,避免了不适合同步下降遍历方法的情况,并改进了单一下降遍历方法的缺陷。除此之外,使用时空相关性和优先级策略,并且改进二叉树的存储结构,从而优化了碰撞检测过程。最后,通过对比实验,证明了相对于其他基于统一遍历的包围盒碰撞检测算法,该算法减少了碰撞检测的时间。由于碰撞场景越来越复杂,检测的实时性将迎来更大的挑战,提高超大场景下的检测效率将是下一步研究的重点。

## 参考文献:

- [1] CHANG J, WANG W, KIM M. Efficient collision detection using a dual OBB-sphere bounding volume hierarchy [J]. *Computer-Aided Design*, 2010, 42(1): 50–57.
- [2] JIMÉNEZ P, THOMAS F, TORRAS C. 3D collision detection: a survey [J]. *Computers & Graphics*, 2001, 25(2): 269–285.
- [3] ZHANG L. The research and realization of collision detection in virtual reality [J]. *Journal of Communication and Computer*, 2011, 8: 693–696.
- [4] STÜVEL S A, MAGNENAT-THALMANN N, THALMANN D, *et al.* Hierarchical structures for collision checking between virtual characters [J]. *Journal of Visualization and Computer Animation*, 2014, 25(3/4): 333–342.
- [5] SEILER C, PENNEC X, REYES M. Geometry-aware multiscale image registration via OBBTree-based polyaffine log-demons [C]// MICCAI'11: Proceedings of the 14th International Conference on Medical Image Computing and Computer-Assisted Intervention. Berlin: Springer, 2011: 631–638.
- [6] KLOSOWSKI J T, HELD M, MITCHELL J S B, *et al.* Efficient collision detection using bounding volume hierarchies of  $k$ -DOPs [J]. *IEEE Transactions on Visualization and Computer Graphics*, 1998, 4(1): 21–36.
- [7] XAVIER P G. A generic algorithm for constructing hierarchical representations of geometric objects [C]// Proceedings of the 1996 IEEE International Conference on Robotics and Automation. Piscataway: IEEE, 1996: 3644–3651.
- [8] GREB A, GUTHE M, KLEIN R. GPU-based collision detection for deformable parameterized surfaces [J]. *Computer Graphics Forum*, 2006, 25: 497–506.
- [9] ZHAO Y, GUO Q, WANG Y, *et al.* Interactive augmented reality method based on collision detection [J]. *Journal of Wuhan University of Technology*, 2010, 32(20): 160–163. (赵元杰, 郭禾, 王伟, 等. 增强现实技术中基于碰撞检测的交互性方法[J]. 武汉理工大学学报, 2010, 32(20): 160–163.)
- [10] DU P. Research on some key technologies for continuous collision detection between deformable models [D]. Hangzhou: Zhejiang University, 2013. (杜鹏. 柔性物体连续碰撞检测若干关键技术研究[D]. 杭州: 浙江大学, 2013.)
- [11] LU R, LIU H. A self-collision detection algorithm based on BVH of complete binary tree [J]. *Computer Applications and Software*, 2012, 29(12): 282–294. (陆睿, 刘卉. 基于完全二叉树 BVH 的自碰撞检测算法[J]. 计算机应用与软件, 2012, 29(12): 282–294.)
- [12] ZHU H. The teaching and research of balanced binary tree in data structure [J]. *Journal of Chifeng College: Natural Science Edition*, 2012, 28(3): 19–21. (朱洪浩. 数据结构中平衡二叉树的教学探讨与研究[J]. 赤峰学院学报: 自然科学版, 2012, 28(3): 19–21.)
- [13] LI H, ZHOU D, WU Y. Collision detection algorithm based on mixed bounding box [J]. *Journal of Computer Applications*, 2010, 30(12): 3304–3307. (李红波, 周东谕, 吴渝. 基于混合包围盒的碰撞检测算法[J]. 计算机应用, 2010, 30(12): 3304–3307.)
- [14] JIANG J, FANG Z, XU J, *et al.* Improved collision detection on dual Sphere-OBB bounding volume hierarchy and its application [J]. *Computer Engineering and Applications*, 2011, 47(17): 172–174. (蒋健勋, 方志刚, 徐洁, 等. 基于 Sphere-OBB 的改进碰撞检测算法及其应用[J]. 计算机工程与应用, 2011, 47(17): 172–174.)
- [15] TANG M, LIN J, TONG R. Graphics hardware accelerated continuous collision detection between deformable objects [J]. *Chinese Journal of Computers*, 2010, 33(10): 2023–2027. (唐敏, 林江, 童若锋. 图形硬件加速的柔性物体连续碰撞检测[J]. 计算机学报, 2010, 33(10): 2023–2027.)