# 计算机动画的算法与技术——基于 GPU 的碰撞检测算法

田正祺　2020080095　软件 01

## 1　碰撞检测的加速算法设计

### 1.1　背景

碰撞检测通常分为宽相位碰撞检测（broad phase collision detection）和窄相位碰撞检测（narrow phase collision detection）[1]。宽相位碰撞检测用于快速粗略地将完全没法碰撞的物体去除掉，然后对可能碰撞的物体进行更精确的但通常更慢窄相位碰撞检测。由于预期仿真的物体（球，正/长方体，四面体）比较简单，此项目会使用简单的窄相位碰撞检测算法，而聚焦在宽相位碰撞检测。

### 1.2　算法概述

朴素的算法会将 $n$ 个物体中的每一个物体与其他 $n-1$ 个物体进行碰撞检测，其时间复杂度为 $O(n^2)$。

此项目使用的数据结构是层级包围体树。一个包围体必需包含它的物体的所有点。为了加快计算，通常会使用最小轴对齐包围盒（minimum axis-aligned bounding box），即最小的，各个边与三个坐标轴平行的长方体。包围体可以形成树结构，约束条件是叶节点必须是物体本身，而所有其他节点都是包围体，并且任意一个内部节点必须是它的子节点的包围盒。在便利层级包围体树时，若一个物体与某一个节点表示的包围体没有碰撞，则物体不会与此包围体的子包围体/物体碰撞，因此可以跳过子节点。对于理想构建的树而言，时间复杂度可以降低到 $O(n \log n)$。

### 1.3　树的构建

此项目的碰撞检测对象是动态的物体，即需要　每次物体的位置更新后做一次碰撞检测，因此需要更新层级包围体树。目前存在 高效的算法，可以更新重复使用同一个树，但是考虑到此在此场景中，物体的位置可能会有巨大的变化，因此选择每次进行碰撞检测时重新构建全新的树。

我们采用 线性层级包围体算法，将所有物体排序，并划分区间，而在同一个区间的物体会在同一个包围体中。由于更小的包围体能够更精确地拟合物体，则目标是将位置类似的物体放在同一个包围体中。将三唯位置信息映射到一唯空间并且保留位置的局部性，可以使用 Z 阶曲线（Z-order curve，也称 Morton order/code）。一个三唯位置的 Z 值由位置的三个值的二进制表示中每个比特的交错而计算的。比如：

```
x = 0.00101010 -> 0.0  0  1  0  1  0  1  0
y = 0.10100101 -> 0. 1  0  1  0  0  1  0  1
z = 0.10101010 -> 0.  1  0  1  0  1  0  1  0
                  = 0.011000111000101010101010
```

### 1.4 树的遍历

#### 1.4.1 其他算法

### 1.5 窄相位碰撞检测

## 2 GPU 实现的思路设计

here[2] f[3]

## 参考文献


[1] Thinking Parallel, Part I: Collision Detection on the GPU | NVIDIA Technical Blog — developer.nvidia.com[EB/OL]. https://developer.nvidia.com/blog/thinking-parallel-part-i-collision-detection-gpu/.

[2] Thinking Parallel, Part II: Tree Traversal on the GPU | NVIDIA Technical Blog — developer.nvidia.com [EB/OL]. https://developer.nvidia.com/blog/thinking-parallel-part-ii-tree-traversal-gpu/.

[3] Thinking Parallel, Part III: Tree Construction on the GPU | NVIDIA Technical Blog — developer.nvidia.com[EB/OL]. https://developer.nvidia.com/blog/thinking-parallel-part-iii-tree-construction-gpu/.