基于图像的快速碰撞检测算法

范昭炜 万华根 高曙明
(浙江大学 CAD & CG 国家重点实验室 杭州 310027)

摘 要 基于图像的碰撞检测算法是一类较新的碰撞检测方法,它有效地利用图形硬件的加速功能,以减轻 CPU 的负担. 文中提出一种基于图像的快速碰撞检测算法,该算法在继承一般基于图像的碰撞检测算法优点的同时,不但能处理任意形状的多面体,而且具有更高效率. 该算法主要采用对物体表面进行自动凸分解,将凸分解结果合理地组织成层次二叉树结构,以及绘制加速等技术. 与相关算法的实验比较说明,该算法在性能上有较大的提高.

关键词 碰撞检测, 基于图像, 绘制加速, 凸分解
中图法分类号 TP391

A Fast Collision Detection Algorithm in Image Space

Fan Zhaowei Wan Huagen Gao Shuming
(*State Key Laboratory of CAD & CG, Zhejiang University, Hangzhou* 310027)

Abstract Image-based collision detection algorithms make efficient use of the graphics rendering hardware and reduce the computation overhead of CPU. It can process arbitrary polyhedra, while preserving the merits of image-based collision detection algorithms. This is achieved by decomposing the surfaces of the object into a list of convex pieces. High efficiency is gained by organizing the convex pieces into a hierarchical binary tree and adopting triangle strip compression to accelerate the rendering process. It has been verified by complex benchmarks, giving favorable results in comparison with some related algorithms.

Key words collision detection, image-based, rendering accelerating, convex decomposition

1 引 言

碰撞检测问题在计算机图形学、仿真、动画及虚拟现实等领域中都是至关重要的问题之一. 随着计算机软硬件及网络技术的高速发展,人们迫切要求对现实世界的真实模拟及与计算机的实时交互操作. 要真实地模拟现实世界中物体的运动,关键之一是必须对场景中的所有物体进行实时碰撞检测. 目前,虚拟环境中的三维物体模型越来越复杂,场景越来越巨大,而实时性的要求越来越高,这种苛刻的实时性要求迫使碰撞检测过程必须尽可能地快速完成. 国内外有许多专家和学者已经针对碰撞检测问题开展了不少有重要价值的研究工作,并且将其理论成果实际应用到虚拟现实、动画和网络交互等各个领域中.

碰撞检测问题的解决方法在时间域上可分为全局的和局部的两类. 在每一类中,又可按空间采样方式划分为两大类: 基于物体空间的碰撞检测和基于图像空间的碰撞检测.

对基于物体空间的碰撞检测已有了较为充分的研究[1-10],研究人员把各种技术如层次表示法、几何推理、代数范式、空间划分、解析方法和最优化方法等应用到相交检测中,其中 Gottschalk 等[5] 的 RAPID, Cohen 等[6] 的 I-COLLIDE, Klosowski 等[7] 的 QuickCD, Chung 等[9] 的 Q-COLLIDE 和 Ehmann 等[10] 的 SWIFT++ 等算法是典型代表. 但这些算法的效率极大地取决于物体模型的表示方法和物体所处场景的复杂程度,同时由于物体空间的碰撞检测只在物体几何空间进行,巨大的计算量往往使得系统不堪重负,特别是在复杂的大规模场景中要实现实时交互显得尤为困难.

基于图像空间的碰撞检测方法一般将三维几何物体通过投影绘制到图像平面上,降维得到一个二维的图像空间;然后分析该空间中保存在各类缓存的信息;进而检测出物体之间是否发生干涉. 基于图像空间的碰撞检测方法最初由于检测结果的不精确性和对图形硬件要求过高一直发展较慢,近年来,随着图形硬件卡加速技术的快速发展,研究人员结

收稿日期: 2002-07-01. 本课题得到国家自然科学基金(60103003)、国家创新研究群体科学基金(60021201)资助. 范昭炜, 男, 1975 年生, 博士研究生, 主要研究方向为计算机视觉、虚拟设计与装配、并行与分布式计算等. 万华根, 男, 1968 年生, 博士, 副研究员, 主要研究方向为虚拟现实及其在工程中的应用、计算机动画等. 高曙明, 男, 1964 年生, 博士, 教授, 博士生导师, 主要研究方向为先进产品建模技术、虚拟设计与装配、基于 Internet 的协同设计等.

合基于物体空间的碰撞检测方法中的一些技术, 使得基于图像空间碰撞检测技术有了新的发展. Shinya 等[11] 和 Rossignac 等[12]做了这方面的开创性工作, 提出了简单有效的干涉检测算法, 但要求有特殊的图形硬件来支持; MyszKowski 等[13]等提出的算法更为实用, 能够处理较复杂的实体; Baciu 等[14-15]在 RECODE 算法中仅采用了深度缓存和模板缓存, 使算法可应用于常规的图形工作站, 甚至只具有低端图形卡的 PC 机. 值得指出的是, 上述算法均未能有效地处理非凸的复杂物体, 而且所采用的绘制方法也比较基础, 性能不够理想.

2 算法概述

一般地, 基于图像的碰撞检测算法由于受到图形硬件存储信息的限制只能处理凸体. 为了能处理任意形状的多面体, 算法首先采用一种新的表面凸分解技术将物体分解为一系列凸面片的集合; 随后, 采用自顶向下策略将这些凸表面片合理地组织成为一棵节点均为凸体的层次二叉树, 同时对该层次树的每个节点进行三角形带压缩编码, 以利于后续的绘制加速过程. 这些步骤属于预处理阶段. 在实时碰撞检测的阶段, 算法先采用掠扫和删减技术[6] 将场景中明显不发生碰撞的物体对快速排除, 然后再遍历那些可能发生碰撞的物体对的层次二叉树. 在遍历二叉树的同时对其节点进行绘制, 并对绘制所得的图像空间进行实时分析, 得出碰撞检测的准确结果. 在对二叉树节点进行绘制的过程中, 算法采用了适合于碰撞检测需求的绘制加速技术加速绘制过程. 算法的整体框架如图 1 所示.

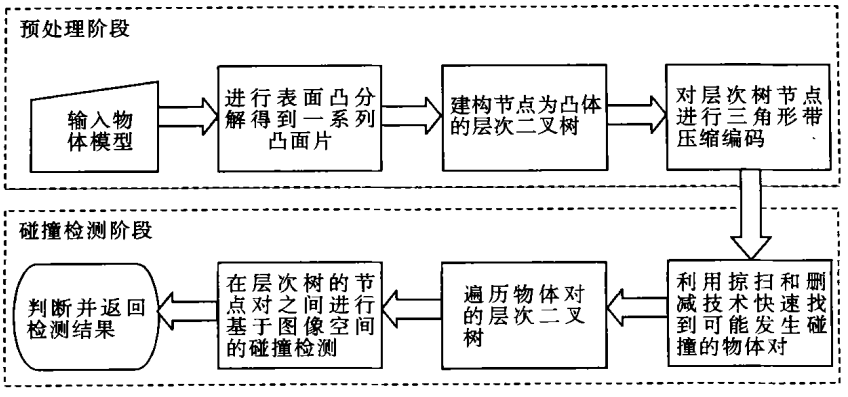


图 1 算法整体框架

3 算法的预处理

算法在预处理阶段的主要任务包括三个方面: (1) 对复杂的非凸物体进行凸分解; (2) 建构节点为凸体的层次二叉树; (3) 对层次二叉树上的所有节点进行三角形带压缩编码.

3.1 基于表面的凸分解

基于表面的凸分解即把非凸物体的表面分解成一些凸面片的集合, 可表示为

$$boundary(P) = \bigcup_i c_i \quad \forall i, j: i \neq j,$$

$$c_i \cap^* c_j = \cong, \ c_i \subseteq boundary(C_i).$$

其中, P 表示物体, c_i为凸的表面片, 即凸片; $C_i = CH(c_i)$, 表示凸片 c_i的凸包体, 本文称为凸块; $boundary(P)$ 表示物体 P 的边界面.

凸片集的生成采用基于图的搜索方法[16-17]. 首先随机选取一种子面; 然后跨边遍历该面的相邻面, 并依据一定的判别准则来决定是否把这些相邻面加入到当前连通凸片上, 如此进行下去, 直到找到最大的连通凸片; 继续该过程直至得到所有凸片的集合. 对凸片求凸包即可得到凸块. 相邻面的遍历策略可基于广度优先, 也可以基于深度优先. 为了保证得到凸的连通面片, 我们采用了下述三个判别准则:

准则 **1**. 遍历经过的边不能是凹边.

准则 **2**. 在当前处理面的远顶点(即除邻边的两顶点之外的顶点) 处不能看到当前凸片中任一面的外表面.

准则 **3**. 当增加当前面后, 所形成的新凸块不能与任何非原凸片上的面相交.

图 2 所示为创建某一个凸片的遍历图例. 其中面 F_1由于违反了准则 1 不能被加入当前凸片, 面 F_2则由于违反了准则 2 而不能被加入. 彩图 1(见彩图页 i) 所示为一个对圆环体进行表面凸分解的实例.

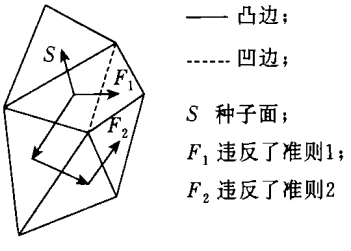


图 2 凸片创建例图

3.2 建构层次二叉树

通过表面凸分解和求凸包运算, 得到了一组凸块集. 显然这组凸块的排列是无序的, 不利于快速有效地进行碰撞检测. 为此, 算法将凸块集组织成一个层次二叉树的结构. 建构层次二叉树就是递归地对一组基本体素进行划分, 这里基本体素即凸块. 层次二叉树的建构有两种策略: 一是自顶向下;

二是自底向上. 我们采用自顶向下的策略. 先将一组凸块集划分为两组, 并创建它们的父节点, 即求这两组凸块集总和的凸包体; 然后递归地对划分出的两组子集进行处理. 凸块集的划分过程如图 3 所示. 其中把有序数值列表划分为两组的方法有: (1) 中值划分; (2) 中间点划分; (3) 平均值划分, 等. 本文采用第二种方法, 即从第一点和最后一点的中间处进行剖分. 如此建构的层次二叉树, 根节点为整个物体的凸包体, 叶子节点为凸块, 中间节点则为其两子节点合并的凸包体.

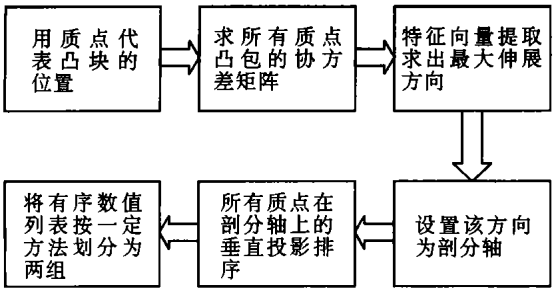


图 3　凸块集划分过程

3.3　凸块的三角形带压缩编码

在基于图像的碰撞检测算法中, 两物体间进行一次碰撞检测时, 往往要对物体进行多次绘制. 因此, 加快绘制速度是提高基于图像碰撞检测算法效率的关键. 物体绘制加速技术的常用方法有多种, 如 LOD、三角形带、基于图像的绘制等. 由于三角形带技术不改变多面体的几何结构, 只通过对三角形面片进行重新组织来减少绘制过程中的数据传输量, 能较好地满足基于图像的碰撞检测算法的要求, 故我们采用这种技术来加速碰撞检测中的绘制过程.

对于用三角形网格表示的物体, 绘制时若要将三角形的每个顶点都传送给图形硬件, 会有许多重复. 为减少重复数据传输的次数, 一般可利用三角形的相邻性, 形成三角形带[18-19]. 这样, 前一个三角形的两个顶点在其相邻三角形的绘制中就可以重用. 这种绘制加速技术获得的理想加速比是 3∶1, 具体加速比则取决于三角形带的质量, 即物体中三角形带的长短及数量等.

为了尽可能加快绘制, 要求优化三角形带, 使生成的三角形带尽可能长, 个数尽量少. 我们借鉴了文献[18]中采用启发式搜索法生成三角形带的思想, 在预处理阶段一次性地对物体以及凸块层次二叉树中每个节点凸块的三角形网格进行处理, 得到比较理想的三角形带, 并对这些三角形带压缩编码. 在基于图像的碰撞检测过程中和物体绘制时, 通过对三角形带进行解码绘制, 以提高绘制速度, 进而加速碰撞检测过程.

4　算法核心

经过预处理, 物体被表示为一个有组织的凸块层次二叉树. 于是在检测两物体是否相交时, 我们通过同时递归遍历它们的层次二叉树来加速检测过程. 由于两物体层次二叉树的节点均为凸体, 因此采用下述基于图像的碰撞检测算法进行它们之间的碰撞检测.

假设 $R_{xy}(X)$ 是物体 X 在 xoy 平面上的垂直投影区域; $I_z(X)$ 是物体 X 在 z 轴上所占的区间; 不妨设 z 轴为深度方向, 于是凸体 A 与凸体 B 相交发生碰撞, 当且仅当下述条件成立:

$$R_{xy}(A) \cap R_{xy}(B) \neq \cong \quad \text{且} \quad I_z(A) \cap I_z(B) \neq \cong .$$

注意, 对于两凸体而言, 在某象素点上 z 轴区间的重叠情况仅有 8 种, 如图 4 所示.

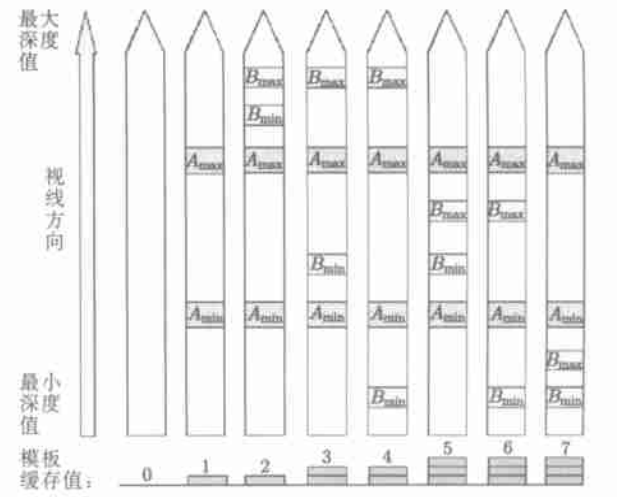


图 4　深度区间重叠情况与模板缓存值的关系图

基于上述结论, 我们就可以将三维物体的碰撞检测问题降维到二维图像空间, 并最终简化到深度方向上的一维区间进行重叠检测.

算法伪代码如下.

算法**1**. Boolean *Image Based Collision Detection*(A, B);

```
Input: 凸体 A 和凸体 B
{ 求出 A, B 在投影面上的最小重叠区域(MOR);
  禁止写帧缓存;
  清深度缓存与模板缓存;
  调用 myRenderSetStencilByZ(A);
  调用 myRenderTestStencilByZ(B);
  SecondRender = false;
  for (MOR 上所有象素点){
      if (模板缓存值为 2)
        return(true); //发现碰撞
      if (模板缓存值为 3)
        SecondRender = true;
  }
  if(SecondRender == false)
    return(false);   //没有碰撞发生
  //互换 A 和 B 的地位
  清深度缓存与模板缓存;
  调用 myRenderSetStencilByZ(B);
  调用 myRenderTestStencilByZ(A);
  for (MOR 上所有象素点){
      if (模板缓存值为 2)
        return(true);   //发现碰撞
      //end for
```

```
    return(false);   //没有碰撞发生
    } //主函数结束

myRenderSetStencilByZ(X)
{
   剔除物体 X 的正面;
   用模板测试操作给模板缓存置 1;
   允许写深度缓存;
   三角形带解码绘制 X;
}
myRenderTestStencilByZ(X)
{
   用模板测试只处理值为 1 的象素点;
   深度测试是否满足条件 Z(X) ≤ Zmax;
   上面两个测试通过,则模板缓存值加 1;
   禁止写深度缓存;
   三角形带解码绘制 X; //正面和背面
}
```

其中 *myRenderSetStencilByZ*(*X*), *myRenderTestStencilByZ*(*X*) 使用了 OpenGL 的多次绘制技术,在进行绘制的同时进行深度测试和模板测试; *myRenderSetStencilByZ*(*X*) 通过模板缓存测试,设置物体 X 投影区域内象素点的模板缓存值为 1,同时使这些象素点的深度缓存值为 X 的对应点的最大深度值;而 *myRenderTestStencilByZ*(*X*) 则在绘制 X 的过程中,测试所有模板缓存为 1 的象素点,若满足条件 $Z(X) \leq Z_{max}$,则将该象素点处模板缓存值加 1.

5 算法实现与实验结果

算法采用 C++, OpenGL/GLUT 在 SGI ONYX2 工作站(CPU 为 R10000)上实现,其中凸包计算采用了共享软件包 QHULL(http://www.geom.umn.edu/software/qhull/).

我们先后建立了两个测试场景,如彩图 2～3(见彩图页 i)所示.其中,红色显示的物体表示与其它物体发生碰撞. 彩图 2 中有 130 个物体做随机运动,彩图 3 中是一只人手和托盘之间做仿真运动.测试在每个场景中使用不同的碰撞检测算法(RAPID[5], RECODE[14]和本文算法)分别进行,并记录每一步碰撞检测所需的时间 t_i 和运行 1 000 步的平均时间 $\bar{t}$.

图 5,6 显示了在两个测试场景中随着场景复杂度的增加平均时间 $\bar{t}$ 的变化情况.可以看出,本文算法比基于图像的碰撞检测算法 RECODE 在速度上有较大提高.

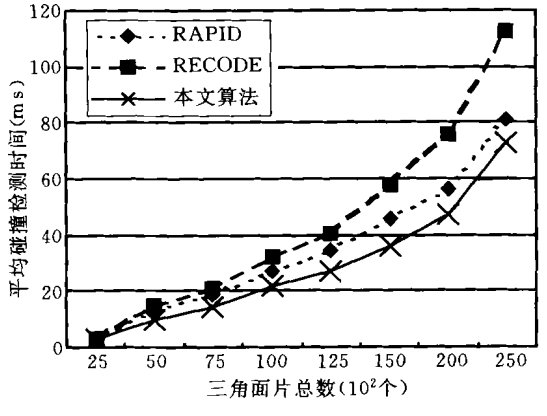


图 5 测试场景一的实验结果

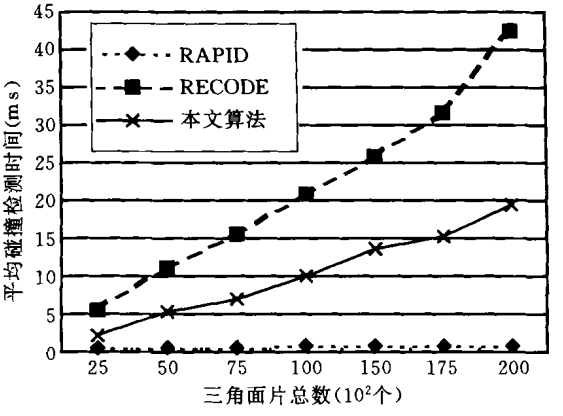


图 6 测试场景二的实验结果

图 7,8 所示为针对同一复杂度场景的碰撞检测时间的变化情况.其中,水平轴是碰撞检测的步数,垂直轴是碰撞检测时间的变化率 η.

$$\eta = \frac{t_i - \bar{t}}{\bar{t}} \times 100\% .$$

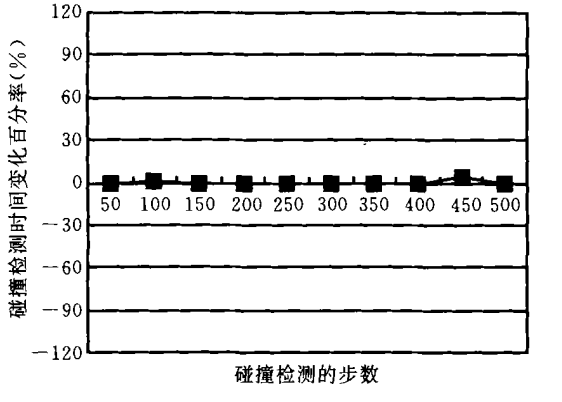


图 7 本文算法在测试场景二中的碰撞检测时间变化率

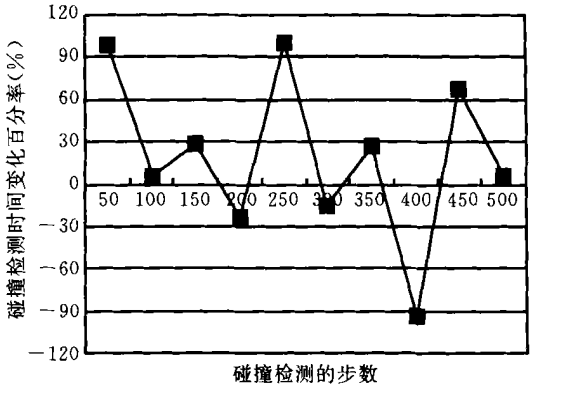


图 8 RAPID 算法在测试场景二中的碰撞检测时间变化率

从图中不难看出,本文算法继承了基于图像的碰撞检测算法的优点,具有基于物体空间的碰撞检测算法所不具备的平稳性.即对同一复杂度的场景,不同步数中的碰撞检测时间变化不大,有利于预测碰撞检测时间.

6 结　论

本文提出了一种新的基于图像的快速碰撞检测算法.通过自动对物体进行表面凸分解,并合理地组织成层次二叉树的结构,该算法能够处理任意形状的多面体,并且具有较高的效率.本文算法继承了基于图像的碰撞检测算法的特点,有效地利用了图形硬件的加速功能,减轻了 CPU 的负担.与其它相关算法相比,本文算法具有下述特点:

(1) 高效率. 通过建构结点为凸体的层次二叉树以及采用基于三角形带的绘制加速技术,有效提高了算法的效率;

(2) 通用性. 可处理任意形状的物体;

(3) 平稳性. 继承了基于图像碰撞检测算法的优点,对同一复杂度的场景而言,碰撞检测时间变化不大,具有较高的平稳性,有利于预测碰撞检测过程;

(4) 可发展性. 由于充分利用了图形硬件,解放了 CPU,相信随着图形硬件的高速发展,算法将有更好的发展前景.

今后的工作将着重于如何结合物体几何空间碰撞检测和图像空间碰撞检测的优势,自适应优化图形加速卡和 CPU 之间的负荷,进一步提高算法的效率.

参 考 文 献

[1] Wang Zhaoqi, Zhao Qinping, Wang Chengwei. An object-oriented collision detection method and its application on distributed virtual environment [J]. Chinese Journal of Computers, 1998, 21(10): 990～994(in Chinese)
(王兆其,赵沁平,汪成为.面向对象碰撞检测方法及其在分布式虚拟环境中的应用[J].计算机学报,1998,21(10):990～994)

[2] Wu Minghua, Yu Yongxiang, Zhou Ji. An octree algorithm for collision detection using space partition [J]. Chinese Journal of Computers, 1997, 20(9): 849～854(in Chinese)
(吴明华,余永翔,周　济.采用空间分割技术的八叉树干涉检测算法[J].计算机学报,1997,20(9):849～854)

[3] Cameron S. Enhancing GJK: Computing minimum and penetration distance between convex polyhedra [A]. In: Proceedings of International Conference on Robotics and Automation, San Francisco, CA, 1997. 3112～3117

[4] Lin M C, Gottschalk S. Collision detection between geometric models: A survey[A]. In: Proceedings of the 8th IMA Conference on Mathematics of Surfaces, Birmingham, UK, 1998. 37～56

[5] Gottschalk S, Lin M C, Manocha D. Obb-tree: A hierarchical structure for rapid interference detection[A]. In: Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH, New Orleans, Louisiana, 1996. 171～180

[6] Cohen J, Lin M C, Manocha D, *et al*. I-COLLIDE: An interactive and exact collision detection system for large-scale environments[A]. In: Proceedings of ACM Interactive 3D Graphics Conference, Monterey, CA, 1995. 189～196

[7] Klosowski J T, *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

[8] Hubbard P M. Real-time collision detection and time-critical computing [A]. In: Proceedings of the 1st Workshop on Simulation and Interaction in Virtual Environments (SIVE), Iowa, 1995. 92～96

[9] Chung K, Wang W. Quick collision detection of polytopes in virtual environments [A]. In: Proceedings of the ACM Symposium on Virtual Reality Software and Technology, Hong Kong, 1996. 125～132

[10] Ehmann S, Lin M C. Accurate and fast proximity queries between polyhedra using convex surface decomposition[A]. In: Proceedings of the Eurographics Conference, Manchester, 2001. 500～510

[11] Shinya M, Forgue M. Interference detection through rasterization[J]. The Journal of Visualization and Computer Animation, 1991, 4(2): 131～134

[12] Rossignac J, Megahed A, Schneider B O. Interactive inspection of solids: Cross-section and interferences[A]. In: Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH, Chicago, Illinois, 1992. 353～360

[13] MyszKowski K, *et al*. Fast collision detection between computer solids using rasterizing graphics hardware[J]. The Visual Computer, 1995, 11(9): 497～511

[14] Baciu G, Wong W S-K. Rendering in object interference detection on conventional graphics workstations [A]. In: Proceedings of the Pacific Graphics, Korea, 1997. 51～58

[15] Baciu G, Wong W S-K, Sun H. RECODE: An image-based collision detection algorithm[J]. Journal of Visualization and Computer Animation, 1999, 10(4): 181～192

[16] Chazelle B, *et al*. Strategies for polyhedral surface decomposition: An experimental study[A]. In: Proceedings of ACM Symposium on Computational Geometry, Vancouver, Canada, 1995. 297～305

[17] Chazelle B, Palios L. Decomposing the boundary of a nonconvex polyhedron[J]. Algorithmica, 1997, 17(3): 245～265

[18] Isenburg M. Triangle strip compression[A]. In: Graphics Interface'00 Conference Proceedings, Montr al, Qu bec, 2000. 197～204

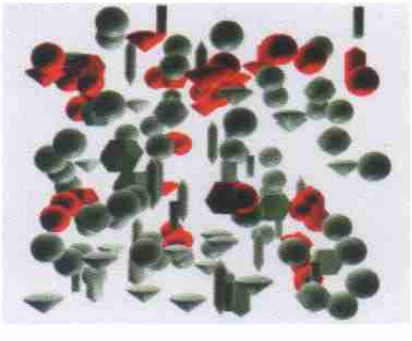
[19] Evans F, Skiena S S, Varshney A. Optimizing triangle strips for fast rendering [A]. In: Proceedings of IEEE Visualization '96, San Francisco, CA, 1996. 319～326

范昭炜等：基于图像的快速碰撞检测算法

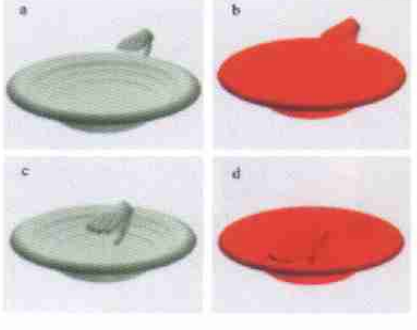
a 圆环体

b 表面凸分解后的凸片集

彩图 1 （范昭炜等） 物体表面凸分解实例

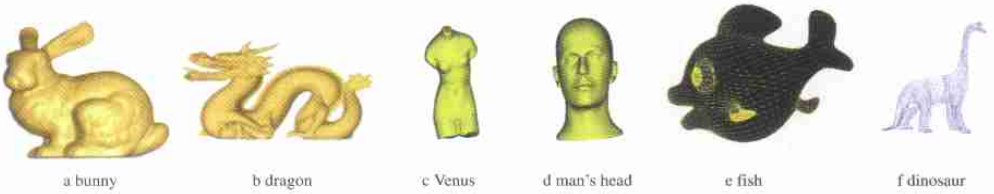


彩图 2 （范昭炜等）测试场景一



彩图 3 （范昭炜等）测试场景二

金哲凡等：分布式并行绘制系统中几何指令流压缩的研究与实现



a bunny b dragon c Venus d man's head e fish f dinosaur

g n=5 h n=4 i n=3 j bunny, N=8 k bunny, N=8

彩图 4 几何模型质量