

# OSG 的视景漫游技术研究

祝新霞<sup>1</sup>, 马明栋<sup>2</sup>

(1. 南京邮电大学 通信与信息工程学院, 江苏 南京 210003;  
2. 南京邮电大学 地理与生物信息学院, 江苏 南京 210023)

**摘要:**虚拟现实技术的发展使得大量开源和商业图形图像引擎相继出现,而 OpenSceneGraph(开源三维图形引擎)越发得到国内外爱好和研究仿真学者的青睐。虚拟场景是虚拟现实技术应用中必不可少的部分,而场景漫游和碰撞检测又是虚拟场景的核心技术,其效果的好坏直接影响整个虚拟场景的真实感。文中阐述了 OSG 的基本特点、核心库及其结构,结合 Visual C++ 编程语言,实现了一个虚拟校园漫游模型,分析了实现此模型的漫游设计,通过从不同角度、距离、方位来观察和操作场景中的某一物体,从而给使用者以沉浸式的效果。同时,采用直线求交器的碰撞检测方法,模拟了学生在遇到障碍物时,发生碰撞停止运动防止穿过物体的过程。利用 OSG 三维图形引擎,明显改善了三维场景渲染对硬件的需求以及开发难度。

**关键词:**虚拟现实技术;开源三维图形引擎;场景漫游;碰撞检测

**中图分类号:** TP393

**文献标识码:** A

**文章编号:** 1673-629X(2018)05-0201-04

**doi:** 10.3969/j.issn.1673-629X.2018.05.045

## Research on OSG Scene Roaming Technology

ZHU Xin-xia<sup>1</sup>, MA Ming-dong<sup>2</sup>

(1. School of Telecommunications & Information Engineering, Nanjing University of  
Posts and Telecommunications, Nanjing 210003, China;

2. School of Geographical and Biological Information, Nanjing University of Posts and Telecommunications,  
Nanjing 210023, China)

**Abstract:** With the development of virtual reality technology, a large number of open source and commercial graphics engine are launched. OpenSceneGraph (open source 3d graphics engine) is paid more and more attention by simulation users both at home and abroad. Virtual scene is a crucial part of the application of virtual reality technology, and the scene roaming and collision detection are the core technology of virtual scene, which affect the authenticity of the whole virtual scene. We discuss the basic characteristics of OSG, the core library and its structure in the paper. Combining Visual C++ programming language, we realize a virtual campus model and analyze its roaming design. The observation and operation of an object from different angles, distances and azimuths gives the user the effect of immersion. At the same time, by the line of intersection collision detection method, we simulate the process of students stopping collision to prevent passing through an object when they encounter obstacles. By OSG technology, the requirements of 3D scene rendering to hardware and the developing difficulty are obviously improved.

**Key words:** virtual reality technology; OpenSceneGraph; scene roaming; collision detection

## 0 引言

现如今,虚拟现实已渗透到人们生活的方方面面<sup>[1]</sup>。它被认为是一种先进的科学技术,相较于传统的人机界面以及流行的视窗操作。人们采用一种全新的方式,即通过计算机对复杂数据进行可视化操作与

交互,极大地促进了虚拟现实技术的发展<sup>[2]</sup>,广泛应用于科学计算可视化、城市规划、建筑设计、广告宣传、医疗、场景仿真等领域<sup>[3]</sup>。场景漫游是虚拟现实的一个重要应用部分,具有人机实时交互特性,可以在虚拟场景中自由行走,不受时间和空间的限制,实现真正意义

收稿日期:2017-04-26

修回日期:2017-08-23

网络出版时间:2018-02-08

基金项目:江苏省青年科学基金项目(BK20140868)

作者简介:祝新霞(1992-),女,硕士研究生,研究方向为图像处理与图像通信;马明栋,博士,教授,研究方向为地理信息系统平台软件设计与开发等。

网络出版地址: <http://cnki.net/kcms/detail/61.1450.TP.20180207.1809.026.html>

上的交互,给人以身临其境的感觉。

文中采用 OpenSceneGraph 三维图形引擎实现了一个虚拟校园场景漫游系统,同时研究了漫游过程中的碰撞检测技术。该系统的实现和设计充分体现了 OSG 在主流三维仿真平台的优势和潜力,其开源、面向对象等优点,可以加快开发过程,提高渲染效率。

## 1 OSG(OpenSceneGraph)简介

OSG 具有支持大规模场景的分页、粒子系统与阴影、多线程、多显示的渲染,以及支持各种文件格式等特性<sup>[4]</sup>。OSG 是一个高性能的开源的跨平台的图形开发三维引擎,诸如飞行器仿真、虚拟现实、科学计算可视化等高性能图形应用程序的开发,就是利用 OSG 设计实现的<sup>[5]</sup>。OSG 结合 Visual C++ 开发,其基于场景图的概念,为图形程序的开发提供场景管理和渲染优化功能,同时 OSG 是一个基于 OpenGL 函数库的面向对象框架,促使开发者从实现和优化底层图形的调用功能的实现中解放出来,为快速开发图形应用程序提供了许多附加的实用工具。

### 1.1 OSG 特点

(1) 开源。

OSG 跟目前市面上一些商业化的三维图形引擎不同,它的所有源码都是遵循 OSGPL 开源协议开发,基于 OSG 的程序和软件的使用与发布无需额外费用。

(2) 源码质量高。

基于 OSG 的开源特点,它的源码历经了无数优秀开发人员的测试和修改,其程序架构和执行效率都有较好的提升。

(3) 跨平台操作。

OSG 支持的操作平台包括 Windows、UNIX、Linux、MacOSX、IRIX、Solaris 等多种操作系统,可以跨平台操作,具有较强的可移植性。

(4) 可扩展性强。

OSG 提升了强大的可扩展能力,其支持节点扩展、属性渲染、回调和交互事件,并支持第三方插件。

### 1.2 OSG 核心库

OSG 核心库(core library)是 OSG 存在的基础,核心场景数据库和操作场景图形的组织和管理都是由 OSG 核心库完成的,并且为导入外部数据库提供接口<sup>[6]</sup>。OSG 核心库主要由以下五个库构成:

osg 库:作为最基本数据库,提供基本场景图形构建场景图节点类、几何体类,用作向量和矩阵运算类。

osgUtil 库:又叫工具库,提供了通用的公用类,用于场景图及内容操作,统计和优化场景图形数据,以及渲染器的创建。

osgDB 库:读写数据库,也是一个文件工具类,提

供场景中读写数据的工作。通过遍历场景图层次结构以完成数据处理工作,实现场景图管理。

osgViewer 库:管理视窗库,提供场景中视口及可视化内容的管理,有利于 OSG 与各种 GUI 结合。

osgGA 库:用于改写界面事件。

## 2 OSG 的基本理论

### 2.1 场景图

场景图是用于游戏和计算机图形学等软件的数据结构设计方法,即在抽象的场景中绘制元素组织结构、相互关系<sup>[7]</sup>。实际上大规模管理场景图形通常使用树结构或图结构组成一组节点集。当前大多数渲染引擎组织其空间数据集皆采用自顶向下且分层的树状结构,以提高渲染效率。例如渲染场景中包含一辆卡车和一条路况顺畅的马路,由于场景图的节点不代表几何关系,可申请一个节点用来表示移动,如图 1 所示。

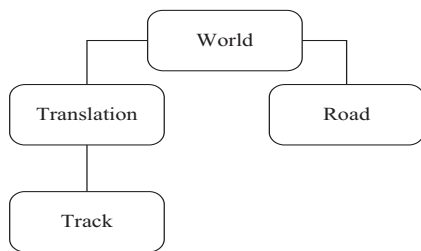


图 1 场景图

根节点一般处于场景图结构的顶部,由根节点延伸至各组节点,各自的几何信息以及用来控制其外观的渲染状态信息均包含在组节点中。这些节点皆可没有或含有多个子成员。场景图的底端展示了组成场景中各个叶节点的物体实际信息。图 2 表示典型的场景图树型结构的组织方式。

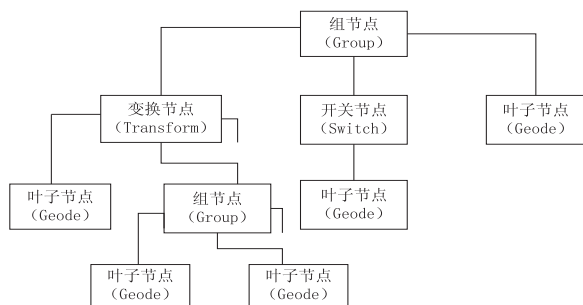


图 2 典型场景图树型结构

整个三维场景结构可以由根节点表示,物体状态切换、矩阵变换、细节层次等属性信息可由根节点延伸的组节点表示;管理属性信息的开关节点(osg::Switch)、变换节点(osg::Transform)、细节层次节点(osg::LOD)等是由组节点派生出来的;叶子节点(osg::Geode)反映场景空间结构和对象状态,数据节点之间共有行为和属性一般通过叶子节点提取<sup>[8]</sup>。

场景图形结构是建在底层 API 函数之上的,它为

3D 程序提供了所需的空 间数据组织能 力及其他特性。OSG 程序层次结构如图 3 所示。

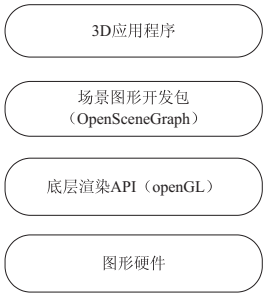


图 3 典型 OSG 程序结构层次模型

2.2 场景图的渲染机制

三维场景处于漫游系统中时被认为是静止的。在实际设计时,使用相机从场景的不同位置 和不同角度去观察,然后把相机从不同位置 和不同角度拍摄到的“场景”输出到屏 幕,就完成了虚拟场景漫游效果<sup>[9]</sup>。三维图 像显示过程如图 4 所示。

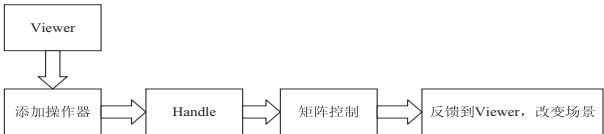


图 4 三维图像显示流程

3 视景漫游实现

3.1 漫游器概述

人们通常是通过眼睛来观察事物的,当视点发生移动时,周围的景物相对于视点在做反向移动,该过程反应到大脑,在大脑中形成场景漫游效果。漫游功能实现在于漫游器的选择。在 OpenSceneGraph 中,既可用默认漫游器,也可使用自定义的漫游器。

Viewer 是场景的核心管理器,即漫游器。在漫游过程中必须根据响应事件的类 osgGA::GUIEvent-Handler 来响应操作。Viewer 通过实时修正场景相机 (Camera 类)的观察矩阵 (代表观察者位置和姿态的矩阵)方法来实现平滑的场景浏览效果<sup>[10]</sup>,因此自定义漫游器就是设置相机观察矩阵的过程。osgGA::CameraManipulator 类由 osgGA::GUIEventHandler 类派生而来,Viewer 可以通过设置矩阵的公共接口和响应得到有效控制。一般的场景操作如图 5 所示。

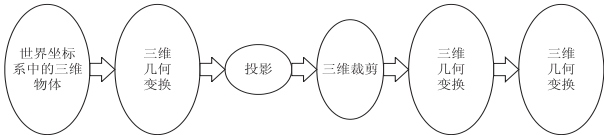


图 5 一般的场景操作

3.2 照相机控制

矩阵为 OSG 提供了一个通用模型。在 OSG 中,不仅所有视图矩阵操作是通过矩阵完成,而且不同相

机间的交互也是通过矩阵完成<sup>[11]</sup>。OSG osgGA::CameraManipulator 的派生类用来实现 OSG 的相机操作功能,而控制相机的关键是实现 osgGA::CameraManipulator 类的 4 个纯虚函数:

```
virtual void setByMatrix ( const osg::Matrixd & matrix ) {} /* 设置相机的观察矩阵 */
virtual void setByInverseMatrix ( const osg::Matrixd& matrix ) {} /* 设置相机的视图矩阵 */
virtual osg::Matrixd getMatrix ( ) const {} /* 获取相机的观察矩阵 */
virtual osg::Matrixd getInverseMatrix ( ) const {} /* 获取相机的视图矩阵 */
```

调用 getMatrix 来获得 SetByMatrix 函数所需的矩阵,实现向下一个照相机传递矩阵。当从一个相机切换到另一个相机时调用 setByMatrix 函数,将上一个相机的视图矩阵传过来,据此设置相机的初始位置。

getInverseMatrix 是变换矩阵的逆矩阵,它会在更新遍历中被场景相机调用,返回当前的视图矩阵。利用这个方法对时间进行处理,可以改变场景状态,进而在调用 getInverseMatrix 矩阵时,改变场景内相机的位置情况<sup>[12]</sup>。这个函数在 void Viewer::updateTraversal () 中被调用。

```
_camera->setViewMatrix ( _cameraManipulator->getInverseMatrix ( ) );
```

外部调用 Viewer 的 setViewByMatrix 矩阵,通过 setByInverseMatrix 矩阵将设置的相机矩阵传递过来,实现了相机记录当前最新的位置的功能。

另外,handle 作为主要控制器函数也是必须的。

```
virtual bool handle ( const osgGA::GUIEventAdapter & ea, osgGA::GUIActionAdapter& us );
```

handle 方法含有两个参数,分别是 GUI 事件的提供者和对 GUI 进行反馈,可以让 GUIEventHandler 根据输入事件操作 GUI。若要对事件进行处理,可由 GUIEventHandler 继承得到自己的类,然后将 handle 方法覆盖,在继承的类中对事件进行处理。osgProducer::Viewer 类维护一个 GUIEventHandler 队列,在该队列里事件依次传递,handle 函数返回值决定此事件是否继续让 GUIEventHandler 处理,若返回 true,停止事件处理;否则 GUIEventHandler 继续对该事件进行处理<sup>[13]</sup>。图 6 为实现方向键控制相机视口的效果图。

4 碰撞检测研究

4.1 基本理论

碰撞问题是由用户的交互和物体的运动产生的。实时检测物体的碰撞并计算出相应的碰撞反应,更新显示结果来避免发生与现实情景不符的穿透现象<sup>[14]</sup>。





图 6 方向键控制相机视口的效果图

处理碰撞检测的方法有两种:一是求交器检测法,二是包围盒检测法。求交器检测法是将遍历器绑上求交器,以实现整个场景的求交过程。常用的有直线求交器和多边形求交器。因为直线求交器只检测直线与多边形的碰撞事件,故多用于判断鼠标点击事件。该系统要实现既定功能,应使用多边形求交器。设置初始值时需同时设置被检测的多边形,若此多边形每帧发生变化,则每帧都需重新设定多边形求交器的参数,就是 polytope 类型变量,以指定多边形网格。这一过程消耗时间较长,又因其默认位置为坐标原点,所以需要获取真实的世界坐标。

OSG 中包围盒检测方法也有两种,分别是用 `getBound()` 函数返回的球形包围盒和用 `getBoundingBox()` 函数返回的立方体包围盒。`getBoundingBox()` 的函数唯独 `Drawable` 节点含有,但所有 `Node` 节点却皆含 `getBound()` 函数,故 `MatrixTransform` 节点的 `getBound()` 函数包含位置旋转信息。因为 `MatrixTransform` 节点挂载实体节点,所以只能使用球体包围盒方法。但球体包围盒的缺点是范围不准确,因此得到的数据可能比实际物体稍微大一点儿。

#### 4.2 系统中实现方法

该系统采用直线求交器检测方法。`osg::LineSegment` 为线段类,表示一个起点和终点构成一条线段,如:`osg::ref_ptr<osg::LineSegment> line = new osg::LineSegment( newPos, m_vPosition)`。与节点的交集接受线段的类是通过 `osgUtil::IntersectVisitor` 进行判别的,添加一条线段到列表可以通过语句 `iv. addLineSegment( line. get())` 实现。若要将线段队列加入到场景中需要借助语句 `node->accept( iv)`。`iv. hits()` 函数可对物体是否发生碰撞进行辨别,返回 `TRUE` 代表物体发生碰撞,返回 `FALSE` 代表物体未发生碰撞。计算距离需要确定相交节点的具体位置,获得相交节点的具体位置通过语句 `osgUtil::IntersectVisitor::HitList hlist` 实现。验证碰撞检测算法的前后效果如图 7 所示。

## 5 结束语

在虚拟场景中,加快显示画面的速度可通过有效  
万方数据

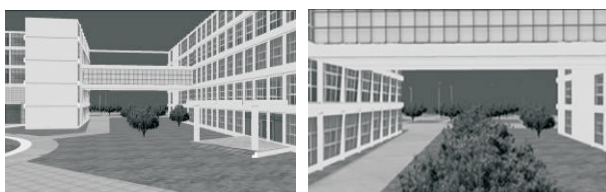


图 7 碰撞检测前后效果图

管理和组织虚拟场景,提高场景漫游效率来实现。OSG 可以非常简单方便地对虚拟场景进行管理和漫游,是目前技术较先进和性能较好的虚拟现实开发包。文中通过实践场景漫游的过程,验证了 OSG 对虚拟场景的漫游原理和发生碰撞时的检测方法,为以后在工程实践的应用提供了重要的参考依据。

#### 参考文献:

- [1] 王 锐,钱学雷. OpenSceneGraph 三维渲染引擎设计与实践[M]. 北京:清华大学出版社,2009.
- [2] 肖 鹏. OpenSceneGraph 三维渲染引擎编程指南[M]. 北京:清华大学出版社,2010.
- [3] MARTZ P. OpenSceneGraph 快速入门指导[M]. 王 锐,钱学雷,译. 北京:清华大学出版社,2006.
- [4] 杨石兴,曹明亮. Step Into OpenSceneGraph[M]. 郑州:郑州大学虚拟实现实验室,2014.
- [5] 徐 凌. 基于 OpenSceneGraph 引擎的漫游系统的研究与实现[D]. 武汉:武汉理工大学,2008.
- [6] 温转萍,申闫春. 基于 OSG 的虚拟校园漫游系统的设计与实现[J]. 计算机技术与发展,2009,19(1):217-220.
- [7] SHREINER D, WOO M, NEIDER J, et al. OpenGL 编程指南[M]. 邓郑详,译. 第 4 版. 北京:人民邮电出版社,2005.
- [8] 申闫春,朱幼虹,曹 莉,等. 基于 OSG 的三维仿真平台的设计与实现[J]. 计算机仿真,2007,24(6):207-211.
- [9] 张艳丽,谭同德,赵新灿,等. 基于 OSG 的虚拟化学实验平台的构建设计[J]. 计算机工程与设计,2010,31(12):2909-2913.
- [10] 杨 晓,廉静静,张新宇. 基于 OSG 的虚拟场景中包围盒碰撞检测的研究[J]. 计算机技术与发展,2011,21(9):32-34.
- [11] YUAN P, WANG S, ZHANG J, et al. Virtual reality platform based on open sourced graphics toolkit OpenSceneGraph [C]//International conference on computer-aided design and computer graphics. [s. l.]:IEEE,2007:361-364.
- [12] HIGHT J, NOVAK J. Game development essentials: game project management[M]. [s. l.]:Gengage Learning,2010.
- [13] ZLATANOVA S, RAHMAN A, SHI W. Topological models and frameworks for 3D spatial objects[J]. Computers & Geosciences,2004,30(4):419-428.
- [14] 孙 倩,刘晶晶. OSG 视景仿真技术应用[J]. 中国科技信息,2017(3):18-20.