基于 Unity3D 的仓储可视化编辑器①

杨迎春[1,2], 廉东本[2], 陈　月[2]

[1](中国科学院大学, 北京 100049)
[2](中国科学院 沈阳计算技术研究所, 沈阳 110168)

摘　要: 近年来, 随着仓储业的快速发展和仓库物资种类的不断增加, 传统的人工管理远远不能满足现代仓库管理的需要. 而能够提供实时、直观、沉浸式管理的三维可视化管理系统越来越受到仓库管理人员的青睐. 任何三维可视化系统的开发都需要一款无缝对接的编辑器. 本文提出的仓储可视化编辑器是针对仓储可视化系统的开发人员, 可帮助开发人员迅速搭建一个真实的三维场景, 缩短可视化系统的开发周期. 本文实现了基于 Unity3D 的仓储可视化编辑器, 首先, 实现了编辑器的总体框架. 然后, 实现了 3D 交互层、统一通信接口、模型操作面板等功能.

关键词: Unity3D; 仓储可视化; 可视化编辑器; 三维场景

Warehouse Visual Editor Based on Unity3D

YANG Ying-Chun[1,2], LIAN Dong-Ben[2], CHEN Yue[2]

[1](University of Chinese Academy of Sciences, Beijing 100049, China)
[2](Shenyang Institute of Computing Technology, Chinese Academy of Sciences, Shenyang 110168, China)

Abstract: In recent years, with the rapid development of the storage industry and the increasing variety of warehouse goods, the traditional manual management can not meet the needs of modern warehouse management. And 3D visualization management system that can manage the warehouse real-timely, intuitively and comprehensively, has become more attractive to warehouse managers. The development of any 3D visualization system requires a seamless editor. The warehouse visual editor presented in this article is developers-oriented. It can help developers to build a real 3D scene quickly and shorten the development cycle of visualization system. In this paper, it realizes the warehouse visual editor based on unity3D. Firstly, it realizes the overall framework of the editor. Secondly, it achieves the functions of 3D interface layer, uniform interface of communication, the operation panel of model.

Key words: Unity3D; warehouse visible; visual editor; 3D scene

仓储是物流与供应链系统中的重要节点和调控中心, 仓储业是国民经济中的一个重要产业, 现代仓储业在现代服务业中占有独特地位. 现代仓储业的发展对于优化物流与供应链系统、转变国民经济的增长方式、提高国民经济的运行质量都具有重大意义[1]. 随着仓储业的快速发展, 如何对仓储进行高效、精确的管理越来越成为人们关心的话题. 我国研究可视化管理系统最早从七十年代中期开始, 八十年代以来, 仓库可视化管理系统取得很大发展[2]. 近年来, 随着三维可视化技术的发展和日趋成熟, 仓储三维可视化管理系统越来越得到企业的青睐.

1　Unity3D引擎介绍

本文提出的仓储可视化编辑器即是仓储三维可视化管理系统的重要辅助工具. 它可以以所见即所得的方式帮助开发者快速搭建一个真实的三维场景, 极大地提高系统的开发效率. 仓储可视化编辑器是基于 Unity3D 引擎进行开发的. Unity3D 不仅仅是一款游戏

① 基金项目:国家水体污染控制与治理科技重大专项课题(2012ZX07505003)
收稿时间:2015-12-15;收到修改稿时间:2016-01-27　[doi:10.15888/j.cnki.csa.005308]

引擎, 它被广泛地应用在陆海空军事训练, 虚拟展馆、化工厂管理系统、安全监控系统等各行各业中[3]. 目前最前端的虚拟现实和诸多形形色色的穿戴式设备都在用 Unity 引擎作为其研发应用的主要技术手段. Unity3D 引擎的技术框架如图 1 所示.

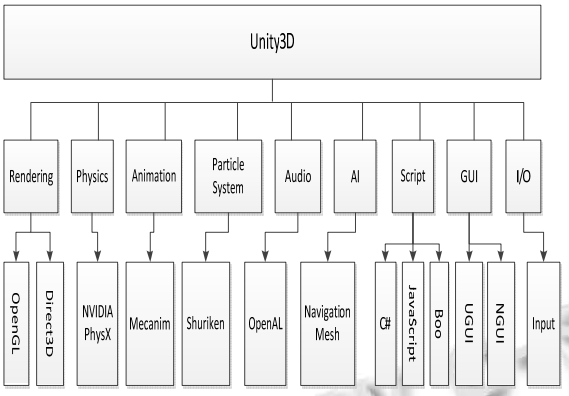

图 1 Unity3D 技术框架图

在图形渲染方面, Unity3D 不仅支持 DirectX 而且还支持 OpenGL. 这为 Unity3D 能够跨平台发布产品提供了支持. 同时, Unity 内置了一个强大的光照贴图烘焙工具 Beast, 开发者可以直接在 Unity 中烘焙出非常逼真、漂亮的光照贴图, 节省在计算光照效果方面的开销. 另外, Unity 内置了 NVIDIA 的 PhysX 物理引擎. PhysX 是目前使用最为广泛的物理引擎, 被很多游戏大作所采用. 开发者可以通过物理引擎高效、逼真地模拟刚体碰撞、布料、重力等物理效果, 使得三维场景更加真实而生动. 在动画系统方面, Unity 从 4.0 版本开始启用了名为 Mecanim 的动画系统[3]. Mecanim 具有功能强大而灵活的特点, 可以创作出令人难以置信的自然流畅动作, 让角色栩栩如生, 能直接在编辑器中编辑和设置角色蒙皮、混合树、状态机和控制器, 还支持动画重定向、IK 骨骼等. 在快速开发方面, Unity3D 支持 C#、JavaScript 和 Boo 三种脚本语言, 其中 C#和 JavaScript 现在在网络开发上使用非常广泛, Boo 和 Phython 很类似, 因此 Unity3D 对于大多数开发者具有亲和力.

使用 Unity3D 的众多类库开发仓储可视化编辑器是具有一定的难度, 只有熟练掌握各种支持的类库, 才能实现具体的功能. 下面讨论了在编辑器的开发过程中一些关键模块和技术的实现方式.

2 编辑器框架的实现

传统的三维可视化编辑器都是 C/S 模式, 而且客户端软件十分庞大, 这给软件的安装和系统的升级带来了很大的不便. 尽管 Unity 公司提供了一个三维场景编辑器, 但是该编辑器不仅需要安装, 而且不具有针对性, 不适用于快速搭建仓库三维场景. 本文实现的 B/S 模式的仓储可视化编辑器不仅降低了系统升级和维护的费用[4], 而且极大地方便了用户的使用. 用户只需要通过浏览器登陆系统就可以快速搭建真实的三维场景, 这极大地提高了用户的工作效率. 同时, 本文实现的编辑器内置了大量的基础三维模型, 在编辑模型的同时, 用户还可以给模型配置基础的模型算法, 使模型具有一定的动态性. 这些功能都最大限度地缩短了三维可视化系统的开发周期.

本文实现的 B/S 模式的编辑器架构如图 2 所示. 系统整体架构采用开源而且成熟的 SSH 框架, Web 容器采用开源而流行的 tomcat 容器, 数据库选择开源的 MySQL 数据库. 3D 交互层使用 html+css+JavaScript 技术, 同时将基础三维场景打包成.unity3d 文件, 嵌入到 web 页面. 基础三维场景通过 unity 编辑器搭建, 三维模型都存放在基础三维场景的固定位置中, 同时为不同的模型编写不同算法, 以方便后期的开发. 数据库存放了模版信息, 模型库信息和用户的场景信息, 数据处理层使用 Java 语言实现, 主要是为用户的操作提供数据, 同时将用户的操作结果保存到数据库. 分层的结构不仅使系统结构清晰、逻辑清楚, 而且也降低了后期维护的成本.

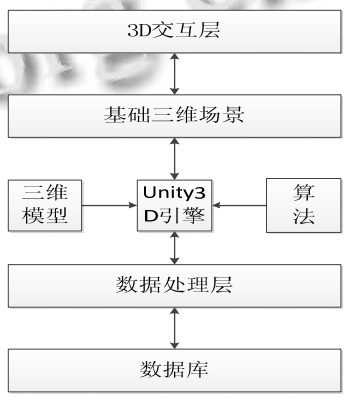

图 2 编辑器架构图

3 3D交互层的实现

3D 交互层主要包括基础三维场景和 web 页面. 基础三维场景是由 Unity 编辑器来实现. 其主要包括模型资源、模型算法和场景基础环境两部分组成. 模型资源的实现过程如下: 首先是三维模型的制作, 使用

Maya、3ds Max 等三维建模软件来对仓库内常用物体进行建模. 然后, 将这些三维模型导入 Unity 中 , 根据实际需要将一个模型或者多个模型组合制作为一个 Prefab. Prefab 为预设体, 可以理解为是一个游戏对象极其组件的集合, 同一个 Prefab 可以生成多个对象, 目的使游戏对象及资源能够被重复使用. 模型算法主要包括用户对模型的基础操作, 如: 拖拽模型, 框选模型等. 这些功能都是需要模型对不同的鼠标事件有不同的响应. Unity3D 引擎并没有提供鼠标的单击、双击和拖拽等事件, 以下是根据鼠标两次单击之间的时间差来区分单击和双击事件.

```
void OnMouseDown()//鼠标单击某物体时响应
{    i++;//鼠标点击次数的统计
     if (1 == i) //当第一次点击的时候
     {    //记下第一次点击的时间
          first = Time.time;
          //是否为单击事件的计时开始
          isBeginSingle = true;
     }
     if (2 == i)//当鼠标第二次点击的时候
{    //若两次点击之间的时间差小于 0.3s, 则为双击事件
          if (Time.time - first < 0.3f)
          {    //不再进行是否为单击事件的计时
               isBeginSingle = false;
               OnMouseDoubleDown ();//双击
               i = 0;//重置计数器
          }
          else //若两次点击的时间差过大
          {    i--;//则这次点击有可能称为单击事件, 或者双击事件. 需要重新计数
               first = Time.time;//需要重新计时
               isBeginSingle = true;
          }
     }
}
void Update () //系统函数, 每帧调用一次
{    if (isBeginSingle) //第一次点击计时开始
     {
          if(Time.time - first > 0.3f)
          {
               if(Input.GetMouseButton(0))// 若 第 一 个点击后过了 0.3s, 鼠标还处于左键按下状态, 则此次点击不是单击事件
               {
                    isBeginSingle = false;
               }
               else//否则的话, 是单击事件
               {    OnMouseSingleDown();//单击
                    isBeginSingle = false;
               }
          }
     }
}
```

场景基础环境的实现主要包括添加天空盒、实现摄像机漫游脚本、实现单元网格等功能. 其中单元网格的单位是 0.5m.

3D 交互层通过将 web 页面和基础三维场景打包后形成的.unity3d 文件融合形成. 3D 交互层的实现效果如图 3 所示.

图 3　3D 交互层效果图

4　统一通信接口的实现

基础三维场景通过 Unity 公司提供的 UnityWebPlayer 插件加载, 网页的 html 代码通过 UnityObject2.js 文件来和该插件进行通信[5]. UnityObject2.js 是 Unity 公司提供的一个 js 文件, 它主要是简化了 Unity 内容嵌入到 html 页面的过程.

在用户搭建三维场景的过程中, 三维场景中的对象需要频繁地和服务器交互. 对象需要将自身的信息发送至服务器进行保存, 同时对象还要向服务器请求自身的固有属性信息. 如果每个物体直接给服务器发送消息, 这将使系统内的消息十分混乱, 不利于消息的管理和系统后期的维护. 因此, 设计一个统一的、易用的交互接口对于系统的稳定性来说是至关重要的.

由于三维场景中的游戏对象和服务器的交互需要借助 JavaScript 脚本. 我们设计和实现了一个如图 4 所示的游戏对象和 JavaScript 交互的接口.

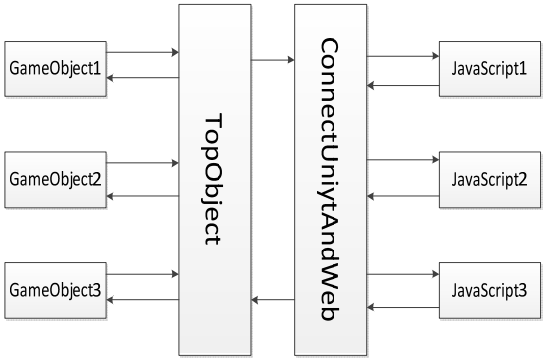

图 4 GameObject 和 JavaScript 交互接口图

在三维场景中, 我们指定了一个 TopObject 物体负责三维场景中所有消息的进出. 即三维场景中每个需要和服务器进行通信的物体都需要先将消息发送给 TopObject, 然后由此物体将消息发送到服务器端. TopObject 发送消息和接收消息的函数如下:

```
//接收三维场景中对象的消息并转发到服务器
void SendMessageToServer(String message)
{     //将消息发送到服务器端
      Application.ExternalCall (“handle3DMes”, params);
}
//接收服务器的消息并转发给特定的对象
void RecMessageFormServer(String message)
{     //获得对象实例名
      string GOname = parseGOName(message);
      //获得对象接收消息的函数接口
      string funName = parseFunName(message);
      //获得游戏对象需要接收的参数
      string params = parseParams (message);
      //将消息转发到特定对象
GameObject.Find(GOname).SendMessage(funName,
params);
}
```

同样地, 我们指定 ConnectUniytAndWeb.js 来实现从服务器端到三维场景的消息的统一管理. 这样的实现不仅利于系统以后的维护, 而且统一管理了系统内的消息, 提高了系统的稳定性.

5 模型操作面板的实现

模型属性面板是编辑器最主要的基础功能. 它不仅可以帮助用户精确地操作模型, 而且也给用户提供了编辑模型业务数据的接口. 本文实现的模型属性面板效果如图 5 所示. 属性面板的实现是通过 Unity3D 引擎的 GUI 实现, 系统提供的 OnGUI()函数可以绘制各种属性框. 其中: 图 5 中左边是对模型进行各种操作的功能按钮, 主要有: 旋转、复制、删除、属性等. 右边是模型的属性面板, 通过属性面板不仅可以控制模型的物理属性: 三维坐标、旋转角度和缩放比例等. 同时还可以录入模型的业务属性等. 这些数据都为后期开发三维可视化系统提供数据源, 方便了后者的开发.

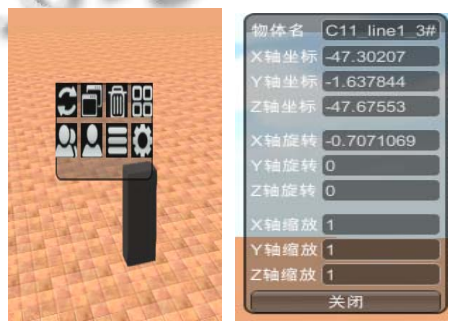

图 5 模型属性面板效果图

6 结语

本文基于功能强大的 Unity3D 引擎, 开发了一个 B/S 模式的仓储可视化编辑器. 首先介绍了编辑器总体框架的实现, 然后对编辑器的核心模块: 3D 交互层的实现、统一通信接口的实现和模型操作面板的实现进行介绍. 该编辑器达到了预期的目标, 极大地方便了实验室人员开发三维可视化管理系统, 具有很高的应用价值.

参考文献

1 林昶,黄庆,帅斌.我国现代仓储业发展现状与对策.西南交通大学学报(社会科学版),2007,8(2):122–126.

2 吕佳乐.仓库可视化管理系统的设计与实现.北京:中国科学院研究生院,2011.

3 Unity Technologies. unity 4.x 从入门到精通.北京:中国铁道出版社,2013.

4 侯淑英. B/S 模式和 C/S 模式优势比较.沈阳教育学院学报, 2007,9(2):98–100.

5 Katz N, Cook T, Smart R. Extending web browsers with a unity 3D-based virtual worlds viewer. IEEE Computer Society, 2011, 15: 15–21.