Web 环境下产品虚拟展示系统的场景优化①

张瑞秋, 褚原峰, 李 晨, 乔莎莎

(华南理工大学 设计学院, 广州 510006)

摘 要: 在 Web 环境下, 由于较慢的网络传输速度和过大的虚拟展示系统场景文件问题, 导致展示场景首次加载时间长, 场景漫游时出现卡顿和模型闪烁. 针对以上情况, 提出了双层次细节渐进优化(Gradually Optimize Two Level of Detail)和模型-贴图过渡显示优化方法. GOTLOD 技术通过分割场景和异步加载场景文件, 使用低精度模型快速搭建展示场景, 利用视距远近动态切换不同精度的模型. 模型-贴图过渡显示优化技术利用视距改变的差值大小, 在模型和贴图间进行多层级切换. 实验结果表明, 采用 GOTLOD 和模型-贴图优化技术, 场景初次加载时间减少 71%, 场景漫游时的 FPS 值提高了 3.9 倍. 该优化技术有效减少了场景加载时间, 有效解决了场景高真实度运行时漫游的流畅性问题.

关键词: 场景优化; GOTLOD; 虚拟展示; unity3D; 展示场景

Product Virtual Exhibition System for Scene Optimization on Web Environment

ZHANG Rui-Qiu, CHU Yuan-Feng, LI Chen, QIAO Sha-Sha

(School of Design, South China University of Technology, Guangzhou 510006, China)

Abstract: Because of the slow network transmission speed and the large amount of data of the virtual exhibition system scene file on web environment, exhibition scene takes a long time to load for the first time, and it has standstill and models flashing during scene roaming. To solve these problems, this paper proposed new methods, which are Gradually Optimize Two Level of Detail and model-mapping transition display optimization technology. GOTLOD technology loads the scene file by splitting scenes and asynchronously, it uses low accuracy of the model show to build scene quickly, and it uses stadia and dynamic to switch different precision of the model. Model-mapping transition display optimization technology uses the difference size of Stadia to switch between models and textures. Experimental results shows that the initial load time reduces 71% and FPS of scene roaming value increase 3.9 times by using the GOTLOD and model-mapping transition display optimization technology. The optimization technology reduces the scene loading time effectively, and solves the high-fidelity runtime scene roaming fluency problems.

Key words: scene optimization; GOTLOD; virtual exhibition; unity3D; exhibition scene

随着虚拟现实应用技术的不断成熟, 桌面虚拟现实系统开始出现在电子商务平台中用于产品的展示. 常见的产品展示形式为图片展示、视频展示以及交互式虚拟展示. 图片展示虽然能简单、直观的展示产品信息, 但图片由于二维的限制, 无法了解到更多产品的细节, 视频展示在三维空间展示上有所突破, 但用户无法无产进行实时的交互, 只能按商家提供的角度去观察和了解产品. 与传统的展示相比, 交互式虚拟展示给用户更大的自由度和更好的体验, 较高的沉浸感和实时的互动, 使得用户对产品有更为直观的体验感受[1]. 常用的虚拟展示技术包括 360 度全景图像、三维模型虚拟展示、全息投影虚拟展示等. 虚拟展示具有经济、方便、对时间和空闲的限制少等优势, 但为了追求虚拟场景高质量的真实度, 搭建场景的模型变

① 基金项目:广东省教育部产学研结合项目(2012B091100327)
收稿时间:2014-06-26;收到修改稿时间:2014-07-24

的越来越复杂, 场景数据量变大, 给硬件设备带来运行负担, 在网络端下载大数据量的虚拟展示场景文件下载时间长, 浏览时不顺畅, 给用户带来不好的体验[2-3]. 因此在较高场景真实度的前提下, 研究如何提高虚拟展示场景加载速度、实时浏览时的流畅性具有重要的意义.

现有的场景优化方法主要包括利用 LOD 技术来简化复杂模型、场景分块、可见消隐等, 但它们只是针对场景的运行速度、计算量进行了优化, 而且存在一定的缺陷. 如利用 LOD 技术简化模型时可能出现突跳现象, 场景分块将大场景划分成多块相互之间不可见的子场景, 通过减少多边形的显示来降低场景的复杂度, 但该方法对于开放的空间无法完成合适的子场景分割. 可见消隐则利用用户视点的可见范围来实时显示场景, 这样能有效降低场景实时渲染时模型的数量, 但当用户视点范围内的场景很复杂时依旧会导致场景运行速度变慢[4]. 而且传统的场景优化方法主要是针对非网页端运行的大场景进行优化, 该类优化虽对场景运行速度有较高的提升, 但场景文件依旧很大, 并未对 Web 环境下的场景加载等问题进行有效优化.

1 问题分析

Web 环境下的产品虚拟展示系统实现需要良好的计算机硬件和网络传输速度的支持, 表 1 为体验目前网络平台下各虚拟展示系统加载时间的统计表, 浏览环境为电信 4M 宽带. 其中网站 A 为上海交通大学钱学森数字图书馆, 加载时间为 87s, 网站 B 为国家会议中心, 加载时间超过 15min 且加载失败. 网站 C 为上海大众汽车产品展示页面, 加载页面时间为 69s.

由表 1 可以看出, 在网络上浏览一个大中型的虚拟展示场景, 其加载时间一般超过 2min, 场景加载速度慢、加载时间长是网络环境下虚拟展示面临的主要问题之一. 造成场景加载速度慢、加载时间长的主要原因是网络传输速度慢和虚拟展示场景文件数据庞大.

表 1 各网站虚拟展示场景加载时间

网站	网站A	网站B	网站C	网站D	网站E	平均
加载时间	87s	--	69s	158s	205s	130s

此外, 为了追求场景的真实感和细节, 大量图片、音频、视频的加载, 导致网络环境下虚拟展示平台进行交互时出现停滞、卡顿等现象, 场景精美度和场景运行流畅度平衡失调. 场景的真实度和浏览时的流畅度是影响场景显示效果的主要因素, 高精度的纹理贴图能让展示场景更加真实, 但加载高精度的纹理贴图时会严重占用下载时的网络宽带, 常见的优化方式为压缩纹理贴图[5-8]. 快速的场景加载速度和良好的场景显示效果往往很难得兼, 因此在快速完成场景初始化文件的加载后, 需要对场景的显示效果进行优化.

2 场景优化策略

针对场景中模型制作和后期的渲染场景优化大致可以分为三类, 模型优化、场景显示优化以及光照优化[9]. 实现模型优化主要有两种方式, 一种是在建模时注意减少模型不必要的分段数, 完成模型后删除隐藏的面, 以减少模型网格的面数. 另外一种是基于 LOD 相关算法, 根据视点裁剪网格选择合适的细节层次输出[10-12]. 场景的显示优化主要有分割场景、可见消隐等方式, 在光照优化方面主要通过烘焙静态场景光影贴图来替换场景实时光照计算效果, 以减轻场景模拟光照计算量[13,14].

结合五金产品虚拟展示服务平台项目的研究和开发经验总结, 文章主要从场景的加载速度和显示效果两方面对场景优化进行研究和分析并给出可行的解决方案.

2.1 场景加载速度优化

在 Web 环境下, 影响场景加载速度的因素主要有两个, 一是场景文件的大小, 二是网络传输速度. 国内网络传输速度普遍不高, 大场景的完整加载需要耗费很长的时间, 在五金产品虚拟展示平台的实际项目中, 结合场景 Web 运行环境提出了一种双层次细节渐进优化技术(Gradually Optimize Two Level of Detail).

双层次细节渐进优化(GOTLOD)技术实现的基本原理为: 首先使用 LOD 建模技术, 对各产品组建一组精度不同的模型 M_0 并存放在数据库服务器中, 当用户在虚拟展厅中要求查看某一产品时, 服务器以保证用户浏览速度为前提, 先将精度最低的模型传送到客户机端, 以供搭建虚拟展示的场景. 在用户对产品进行交互浏览的过程中, 网络实际上是空闲的, 此时服务器根据当前展厅的模型与更高一级精细度的模型求取差值 δN, 并自动将高一级精度的模型传送到 IE 临时文件夹, 当差值达到设定的阈值 T_0 时, 对客户机产品模型进行优化替换. 使用何种细节级别的模型组组建场景的依据主要是预先设定的阈值 T_0 和视距 L 的远近

以及视距变化值 δL. 整个过程不仅保证了客户能最快速的在虚拟展厅中浏览产品模型, 并且在浏览和交互的过程中, 模型会逐渐精细. 这一过程正好符合了人们对一个新事物由粗到细的认知过程. 其实现的原理框架图如图 1.

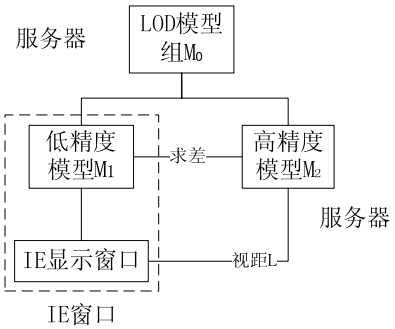

图 1　双层次细节渐进优化技术实现框架图

与 Web 环境下的 3D 游戏场景模型加载不一样, 虚拟展示平台场景加载具有一定的特殊性, 一般虚拟展示交互页面位于网站的二级或者三级页面. 用户在浏览产品虚拟交互展厅前会在公司企业主页停留, 在了解产品的过程中也会在产品介绍页面大量停留, 以便查找和筛选. 将虚拟展厅的场景配置文件和产品资源文件分开存储, 在用户进入首页浏览的网络空闲期, 在后台异步加载场景配置文件到客户机缓存区, 如 IE 临时文件夹. 以便用户在浏览虚拟展示时能快速搭建展厅的场景, 然后根据用户的选择动态加载产品展示模型.

应用 GOTLOD 技术优化原理, 结合 unity3D 平台在项目开发中实现不同细节级别模型动态加载的主要流程包括三个步骤. 首先将 3ds Max 中完成的高精度产品模型导入到 unity3D 中进行编辑, 利用 uniLOD 插件对高精度模型进行网格简化生成一组包含不同细节层次的产品模型 M, 然后将这组模型以流的形式导出成.unity3d 格式的资源文件, 在导出资源文件的同时生成一个.xml 格式的场景配置文件, 打包资源文件和场景配置文件并存至服务器. 用户在浏览网页时, 先异步加载场景配置文件到缓存区, 以此来提高场景的加载速度. 用户在虚拟展厅浏览产品时, 快速加载缓存区的场景配置文件, 再动态加载产品展示模型. 实现产品模型动态加载替换的算法关键代码如下, 实现语言为 C#.

```
public void Start()
{
    ResourceManager.getInstance().LoadSence("Scenes 1/xml.txt");
    if(StartEvent != null)
            {
    StartEvent(this.gameObject);}
        }
    public void Update()
    {
        if (UpdateEvent != null)
        {
         UpdateEvent(this.gameObject); }
        }
```

最后将阈值 T_0 和视距 L 作为不同细节层级模型间替换的标准, 不断优化替换场景中的低精度模型文件, 对于阈值 T_0 的设定, 需要考虑场景复杂度、LOD 模型组层级数. 在实际项目开发中, 通过反复实验各种不同面数的五金产品以及设置不同层次数的 LOD 模型组来最终获得较为理想的阈值 T_0.

2.2 场景显示效果优化

场景模型和纹理贴图的精细程度决定了场景的真实度, 而场景的停顿感以及模型切换时的闪烁会导致流畅度的下降, 模型闪烁现象的出现是由于高精度模型切换到低精度模型之后视差所造成的. 此外环境光照、模型的抗锯齿性等都会对场景的显示效果产生影响[15-17]. 由于制作虚拟展厅使用的 unity3D 平台对模型抗锯齿优化具有良好的支持, 因此在开发过程中主要对在保证以场景快速加载为前提下如何提高场景显示的真实度和模型闪烁问题进行了分析研究.

LOD 和 Mipmap 技术都是基于视点远近的不同层级模型或者贴图的切换, 利用视点远近切换显示原理, 提出一种模型-贴图过渡显示优化方法. 模型-贴图过渡显示以动态加载为基础, 在不同层级模型间切换时, 根据视距 L 的变化差值 δL 作为判断条件, 如果差值 δL 大于预先设置的阈值, 则在模型切换前先进行不同精细层级的贴图切换, 再进行不同层级模型间的切换, 起到切换时平滑过渡的效果. 此外在模型与贴图切换时, 设置一定的渐隐时间以增强平滑过渡效果. 为了解决像 Mipmap 贴图占用过大内存的而导致场景运行出现流畅性问题, 此优化方法经过多次场景 FPS 数据测试后, 发现 1024×1024 的贴图选取 3 至 4 个低层级

的贴图, 512×512 的贴图选取 2 至 3 个低层级的贴图用于模型-贴图切换能达到最好的优化效果. 模型-贴图过渡显示优化算法的流程图如图 2 所示, 其中 L_1、L_2 和 L_3 分别为判断是否进行切换的阈值, 且 $L_1<L_2<L_3$.

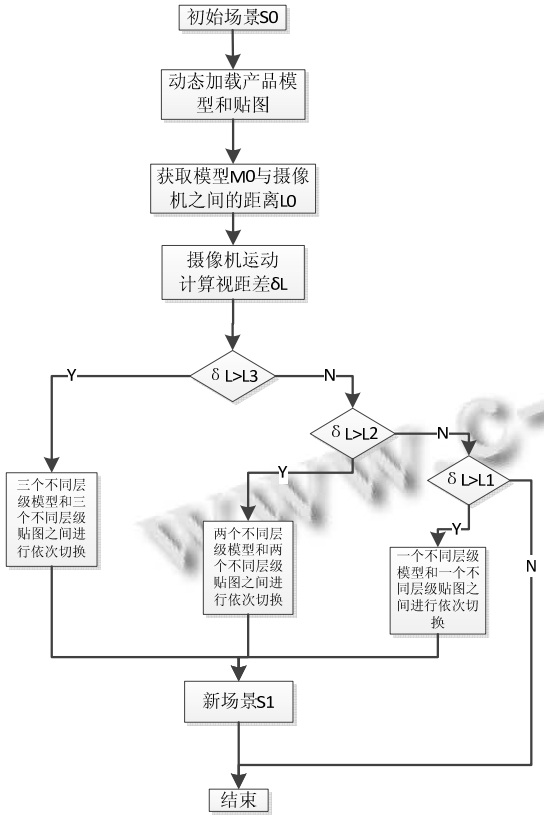

图 2 模型—贴图过渡显示优化算法流程图

以视距差作为模型替换的判断条件时, 在较短的时间内用户调整视距差的范围可能较大, 较大层级差之间的模型切换时便会出现模型闪烁问题. 模型-贴图过渡显示方法能有效的解决模型闪烁问题, 而且能够有效的提高场景的真实度. 当视距差 δL 大于预先设定的最大阈值 L_3 时, 模型从较高精度的 M_0 层级切换到降低精度的M_4层级, 但在切换过程中并不是像传统的 LOD 技术那样直接切换到 M_4 层级. 其切换过程为先将 M_0 模型对应的高分辨率 T_0 贴图切换到分辨率相对较低的 T_1 贴图, 然后切换模型到 M_2 模型, 再将贴图切换到 T_2 贴图, 最后将模型切换到 M_4 模型, 贴图切换到 T_3 贴图. 当视距差 δL 在阈值 L_3 和 L_2 之间时, 进行两次的贴图切换和两次的模型切换, 而当在阈值 L_2 和 L_1 之间时则只进行一次的贴图切换. 通过后期的场景性能测试, 证明模型-贴图过渡显示优化方法, 能有效的降低模型之间切换出现的闪烁现象, 而且场景能一直保持较高的真实度.

针对 unity3D 平台在场景制作过程的优化, 主要包括 LightMap 光照贴图优化、LOD 技术以及选择剔除. 利用 unity3D 内置的 Lightmapping 组件对虚拟展厅进行光影烘焙, 得到 LightMap 光照贴图. 使用 LightMap 光照贴图可以有效的减少场景中灯光的计算量, 利用光照贴图来模拟场景中静态物体的光照效果, 而场景运行时这类模型物体不参与光照计算, 从而降低场景的计算量提高场景流畅度. 在虚拟展示场景中 draw calls 的多少会直接影响到场景运行的流畅度, draw calls 越少场景运行的越流畅. 所谓的 draw call 是指引擎准备数据并通知 GPU 的过程, 每次 draw call 都会逐个物体进行, 对于每一个物体, 不仅 GPU 渲染, 在 unity3D 中重置材质也会非常耗时. 在 unity3D 平台中, 将使用相同材质的模型在 Inspector 面板中标记为静态物体, 然后使用 Static Batching 功能将所有模型组合在一起, 使得在一次 draw call 中批量处理了多个模型, 从而降低了场景中的 draw calls. 而 unity3D 内置的 Draw Batch 功能实际上就是通过将相同材质的物体分为一组, 然后组合成一个物体, 从而在一次 draw call 中处理多个模型, 降低了 GPU 渲染所耗费的时间, 提高了场景显示速度.

3 优化测试

五金产品虚拟展示系统测试运行的计算机硬件参数如下:

CPU: AMD A8-4500M APU with Radeon(tm) HD Graphics 1.90 GHZ

内存: 4.00 GB

显卡: ATI Radeon HD 7640G + 7470M 1G

硬盘: WDC WD50 00BPVT-80HXZT3

虚拟展厅的面积为 220 平米, 制作比例 1:1, 仿真测试最优加载速度和显示效果的阈值 T_0、 L_1、L_2、L_3 的值分别为 3、1.5m、3m、5m. 对于阈值 L_1、L_2、L_3 的设定, 需要根据展厅的大小以及制作模型的比例来设定, 如果场景中摄像机视角、可见距离等参数与人眼的可视范围接近, 且场景按照 1:1 的比例构建, 则上述值为阈值 L_1、L_2、L_3 为较优值, 但阈值小于上述值时会频繁发生模型与贴图之间的切换, 反而增大了运算量, 其它比例下的场景, 则需要按照相应的比例缩放阈值大小. 阈值 T_0 的确定则通过了一系列对比测试

试验, 测试数据如图 3 所示, 其中所有模型组的层级数均为 10, 过低的层级数时在模型-贴图过渡显示优化算法时效果不明显, 但过多的层级数会导致场景文件过大, 反而影响了场景的加载.

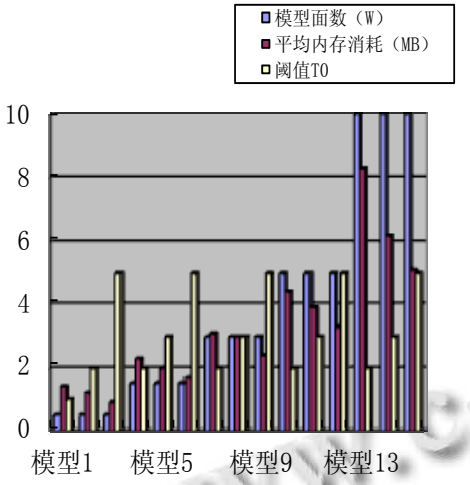

图 3　阈值 T_0 设定测试实验数据

服务器端差值 δN 的计算公式为:

$$\delta N=(N_x-N_x\text{-}1)/_N$$

其中 N_x、$N_{x\text{-}1}$ 分别代表服务器端更高一级精度模型的多边形数量和当前客户机端展示模型的多边形数量, 通过实验测试当 δN=3 时, 展示系统在不同精度模型替换时内存消耗适中, 当 δN<3 时, 模型切换频率过高, 导致消耗内存过大, 而 δN>3 时, 较近视距内的模型替换时会出现跳跃现象, 因此仿真实验选取的阈值 T_0 为 3. 但阈值 T_0 的设定还与视距有关, 视距较远的模型阈值 T_0 可以相应的调大一些, 以减少内存消耗.

测试的网络传输条件为 4M 宽带, 对图 4 所示的虚拟展厅场景优化前和优化后场景加载时间和内存消耗测试数据对比如表 2.

表 2　场景加载时间和内存消耗数据对比

虚拟展厅	加载时间	消耗内存	
		最大内存	平均内存
优化前	24.2s	395 796K	233 852K
优化后	6.9s	228 620K	132 348K

对图 5 所示的单件产品交互展示模型进行优化测试的 FPS 数据如表 3 所示, 对图 4 所示的虚拟展厅场景进行优化测试的 FPS 数据如表 4 所示.

图 4　五金产品虚拟展厅场景

表 3　单件产品模型 FPS 数据对比

单件展品	加载稳定后	发生交互时
优化前平均 FPS(帧/s)	28.8	21.6
优化后平均 FPS(帧/s)	47.4	40.7

表 4　虚拟展厅优化前后 FPS 数据比较

虚拟展厅	载入时	载入稳定后	漫游时
优化前平均 FPS(帧/s)	14.7	16.8	9.4
优化后平均 FPS(帧/s)	42.6	37.5	35.3

测试数据表明, 通过对场景模型、场景加载速度和场景显示效果的一系列优化, 达到了场景漫游时 15 帧/s 及以上的要求. 通过表 2 的数据可以看到, 在未使用优化算法前场景加载时间为 24.2s, 利用 GOTLOD 优化后, 场景在 6.9s 内加载完成, 场景的加载速度提高了 3.5 倍. 表 3 显示的单件产品展示场景在加载稳定后的 FPS 测试数据为 47.4 帧/s, 高出优化前近 20 帧/s, 而发生交互时, 优化后的 FPS 是优化前的 1.9 倍, 有效提高了场景交互时运行的流畅度. 表 4 所示的虚拟展厅场景 FPS 测试数据显示, 虚拟展厅场景经过优化后 FPS 提高了 3.9 倍, 而且对于越复杂的场景, 优化效果越明显, 能有效提高浏览速度. 在实验测试时发现该优化存在一点不足之处, 当展厅初次以低精度模型加载时, 虽然加载速度有明显提高, 但由于初次搭建场景的模型都为低精度模型, 视距很近的物体会显得精度不高, 场景真实度不高.

通过实验数据表明该优化方法能有效减少虚拟展示场景初次搭建时间, 给用户带来较好浏览体验. 在后期场景漫游时能在保证场景高真实度的情况下, 有效降低系统内存消耗、提高场景运行时的 FPS 值. 但优化算法在场景初次搭建时由于模型精度较低, 在视距较近的场景会出现短暂的真实度偏低的情况. 这将

是后续研究需要解决的问题, 可以考虑在首次场景加载时将视距远近作为判断条件, 综合考虑加载时间和场景真实度, 有选择性的加载不同精度的场景模型, 而非全以最低精度模型加载模型来搭建初始场景.

图 5 单件五金产品展示

4 结语

计算机硬件的不断发展带来了产品虚拟展示应用的普及, 但由于网络传输速度的限制产品虚拟展示系统较难在网络平台中快速的加载和流畅的运行. 结合五金产品虚拟展示平台的实际项目开发, 研究了大规模复杂虚拟展示系统的场景优化方法, 并结合 Web 运行环境, 在场景快速加载和场景显示效果方面提出了可行的优化方法, 并通过简单和复杂的产品展示场景进行了验证. 在保持较高的场景真实度的情况下, 有效的缩短了场景初次加载的时间, 在场景漫游时有效提高了场景实时渲染的帧率.

参考文献

1 Tian W. The analysis of product design research based on virtual reality technology. Applied Mechanics and Materials, 2013, 347: 3123–3127.

2 孟凡超.浅析虚拟现实技术在现代展示艺术中的应用.天津美术学院学报,2012,(2):76.

3 Li C, Ke Y, Wang Y. Research on the interaction of car online modification based on virtual display. 2013 International Conference on the Modern Development of Humanities and Social Science. Atlantis Press, 2013.

4 李长春,何荣,王宝山.LOD 技术在大范围复杂场景简化中的应用.河南理工大学学报(自然科学版),2007,26(2):181–186.

5 赵玲,程其超.基于 EON 的虚拟化工场景优化技术的研究.微型机与应用,2010,29(15):32–37.

6 谢阳,张燕,李改红.基于 Virtools 的虚拟现实系统建模与优化.微处理机,2013,(1):92–95.

7 Chen SY, Shi GS, Li G. The research of large–scale 3D scenes rendering optimization. 2009 WRI World Congress on Software Engineering(WCSE 2009). 2009, 3. 276–285.

8 花蕾.具有较高真实感的网上虚拟校园构建技术研究[学位论文].西安:西安理工大学,2010.

9 俞静.三维复杂场景优化处理的研究与实现[学位论文].成都:西南交通大学,2011.

10 程堃.LOD 模型简化算法的研究与实现[学位论文].沈阳:东北大学,2011.

11 Le MH, Vavilin A, Jo KH. Enhancing 3D scene models based on automatic context analysis and optimization algorithm. Emerging Intelligent Computing Technology and Applications. Springer Berlin Heidelberg. 2012. 435–440.

12 祝清鲁.大规模场景中 LOD 技术的研究与应用[学位论文].北京:北京邮电大学,2009.

13 况扬,江婕.桌面型虚拟现实系统场景优化问题研究.科技广场,2011,(5):42–44.

14 孙恒宇,尤超,王洋.基于 Unity3D 的数字站区管理应用系统优化研究.地理信息世界,2013,20(1):103–106.

15 李佳蓓,杜宝江,刘佳.虚拟场景实时渲染软件优化方法.计算机应用,2009,29(S1):299–301.

16 宋歌,杨红雨.基于纹理集的大规模场景模型优化算法.计算机工程与设计,2011,32(9):3119–3122.

17 冯良波.虚拟现实中三维场景构建的优化研究[学位论文].长沙:中南大学,2010.