Journal of System Simulation

Volume 32 | Issue 7 Article 17

7-15-2020

Mobile-phone-oriented Stereoscopic Display and Interaction Framework for Cloud-based Virtual Reality 3D Scenes

Jiazhe Liu
School of Computer Science and Technology, Changchun University of Science and Technology, Changchun 130022, China;

Chunyi Chen
School of Computer Science and Technology, Changchun University of Science and Technology, Changchun 130022, China;

Xiaojuan Hu
School of Computer Science and Technology, Changchun University of Science and Technology, Changchun 130022, China;

Weidong Liang
School of Computer Science and Technology, Changchun University of Science and Technology, Changchun 130022, China;

See next page for additional authors

Mobile-phone-oriented Stereoscopic Display and Interaction Framework for Cloud-based Virtual Reality 3D Scenes

Abstract

Abstract: A VR application framework based on cloud computing is proposed to overcome the difficulty of interactively generating high quality 3D scene images on mobile phones. *We combine the cloud-based rendering and the VR technology. Our framework uses a mobile browser as a client, transmits the image data by the video block stream, processes the interactive data by the anti-jitter algorithm, and has accomplished the web-based lightweight rendering. The framework reduces the cost of data visualization on mobile phone and avoids the need of installing a plug-in package.* The experimental results show that our display rate could be improved to 30 frames per second(fps), and the bandwidth could be limited under 80 kb/s. With the proposed framework, we can obtain much better VR Experience than the general mobile applications.

Keywords

cloud-based rendering, VR technology, web-based lightweight rendering, video block stream, anti-jitter algorithm, mobile plug-in

Authors

Jiazhe Liu, Chunyi Chen, Xiaojuan Hu, Weidong Liang, Qiwei Xing, and Huamin Yang

Recommended Citation

Liu Jiazhe, Chen Chunyi, Hu Xiaojuan, Liang Weidong, Xing Qiwei, Yang Huamin. Mobile-phone-oriented Stereoscopic Display and Interaction Framework for Cloud-based Virtual Reality 3D Scenes[J]. Journal of System Simulation, 2020, 32(7): 1360-1374.

第 32 卷第 7 期 系统仿真学报© Vol. 32 No. 7
2020 年 7 月 Journal of System Simulation Jul., 2020

面向手机的云端 VR 三维场景立体显示与交互框架

刘佳哲，陈纯毅*，胡小娟，梁伟东，邢琦玮，杨华民
(长春理工大学计算机科学技术学院，吉林 长春 130022)

摘要：针对手机无法渲染可交互的高质量三维场景这一弊端，提出一种面向手机的云端 VR 应用框架。*此框架结合云渲染和 VR 技术在手机端实现可交互高质量三维场景的立体显示，并以手机浏览器作为客户端，利用视频块流传输图像数据、消抖算法(AJA)处理交互数据，实现 Web 端轻量级渲染，不仅降低了移动端数据可视化运算成本，还避免使用插件和下载安装包。*实验表明本文框架的显示帧率能达到 30 帧每秒，带宽控制在 80 kb/s 以内，能稳定地在手机上实现传统 App 无法实现的逼真 VR 体验。

关键词：云渲染；VR 技术；Web 端轻量级渲染；视频块流；消抖算法；手机插件
中图分类号：TP391.41 文献标识码：A 文章编号：1004-731X (2020) 07-1360-15
DOI: 10.16182/j.issn1004731x.joss.19-VR0477

Mobile-phone-oriented Stereoscopic Display and Interaction Framework for Cloud-based Virtual Reality 3D Scenes

Liu Jiazhe, *Chen Chunyi**, *Hu Xiaojuan*, *Liang Weidong*, *Xing Qiwei*, *Yang Huamin*
(School of Computer Science and Technology, Changchun University of Science and Technology, Changchun 130022, China)

Abstract: A VR application framework based on cloud computing is proposed to overcome the difficulty of interactively generating high quality 3D scene images on mobile phones. *We combine the cloud-based rendering and the VR technology. Our framework uses a mobile browser as a client, transmits the image data by the video block stream, processes the interactive data by the anti-jitter algorithm, and has accomplished the web-based lightweight rendering. The framework reduces the cost of data visualization on mobile phone and avoids the need of installing a plug-in package.* The experimental results show that our display rate could be improved to 30 frames per second(fps), and the bandwidth could be limited under 80 kb/s. With the proposed framework, we can obtain much better VR Experience than the general mobile applications.

Keywords: cloud-based rendering; VR technology; web-based lightweight rendering; video block stream; anti-jitter algorithm; mobile plug-in

引言

现如今，随着智能手机的普及，各种基于手机的 App 也随之被开发出来。如：基于移动端的多人网络游戏[1-2]、移动增强现实(Augmented Reality， AR)应用[3-4]、移动虚拟现实(Virtual Reality，VR)应用。移动 VR 应用利用手机的普及性和便携性，在保证 VR 可视化质量的前提下，将 VR 技术与智能手机相结合。相比需要价格高昂的 VR 硬件设备和特殊使用条件的传统 VR 应用，其

收稿日期：2019-08-31 修回日期：2020-01-13;
基金项目：国家自然科学基金(U19A2063), 吉林省科技发展计划(20170101005JC, 20180519012JH, 20190302113GX);
作者简介：刘佳哲(1994-)，男，湖南，硕士生，研究方向为真实感三维图形绘制；陈纯毅(通讯作者 1981-)，男，重庆，博士，教授，博导，研究方向为真实感三维图形绘制、计算机仿真。

更易被大众普遍接受[5]。

移动 VR 技术能降低设备成本、提高设备便携性、提升用户体验质量，但是在手机上进行 3D 渲染需要很高的计算成本，特别是针对手机游戏这种需要实时交互的应用。刘小军等[6]提出的 WebBIM 场景轻量级实时漫游算法和刘镇等[7]提出的面向移动终端的分布并行化渲染都是为了改善这个问题，但是都没有改变移动端本地渲染的模式。研究表明[8]，尽管目前移动设备的 CPU 与 GPU 的处理能力已经大幅度提高，但与使用最新图形渲染技术而创新开发的高质量互联网/控制台 3D 游戏对设备的要求相比，仍有较大的差距。针对这个问题，云游戏[9]是一种很好的替代模式，它将高计算成本的渲染部分放到了云服务器上，移动端只需要显示容量较小的图像。其降低了移动端的运行成本，对移动设备的使用寿命也有很好的保护，以一种全新的方式在低端设备上向用户提供高端的游戏体验[10]。

本文引入云渲染，嫁接云游戏模式，面向手机 Web 端进行开发。并结合光线跟踪技术、VR 技术、消抖算法、HTML5 标准、智能手机和云服务，将带有镜面反射和全局光照的高质量三维场景，在客户端实现无插件立体显示。本框架降低了移动设备的发热和负载，最终在低配置移动设备上显示出高计算成本的可交互 VR 三维场景。

相比现有的移动 VR 应用，本框架创新如下：

(1) 本框架可以使用户在手机上体验传统移动 VR 应用无法体验的高质量 VR 三维场景。由于硬件限制，某些可交互的高质量逼真 VR 三维场景无法被手机渲染。而本框架引用云渲染，渲染成本皆消耗在服务器端，手机端只需要显示小容量的图像，运行成本低、效果上限高。

(2) 本框架不需要加载任何移动 VR 应用插件。手机体积小、集成度高，运行高资源需求的 VR 应用通常会伴随着很多插件。而本框架面向手机 Web 端进行开发，客户端不需要加载任何插件，包括浏览器常用的 Flash 插件；同时本框架借助手机自带的传感器进行 VR 三维场景交互，降低了交互硬件成本，使 VR 体验变得自然。

(3) WebVR 将传统移动 VR 应用带入手机 Web，解决了移动 VR 应用插件繁多等问题。本框架进一步改进了 WebVR，用云渲染代替 WebGL 本地渲染，用手机传感器事件代替 WebVR API。本框架不仅具有 WebVR 开发成本低、跨平台、易集成、易维护、无安装包等优点，还具备开发效率高、渲染场景逼真、内容丰富、VR 交互成本低等优势。

1 相关工作

1.1 移动端云渲染应用

远程渲染技术的思想最早出现于 20 世纪 90 年代[11]。其通过专有的、高性能的集成设备(称为服务器端)负责渲染工作，把显示和用户交互的程序放在网络远端设备(称为客户端)执行。现如今，远程渲染技术已经变得成熟，并且被应用在很多 App 中。

尽管未来移动设备的性能会有所提高，但人们预测，基于 3D 渲染的新兴应用程序对移动设备的需求与移动设备能力之间的差距只会越来越大[12-13]。在所有移动应用中网络游戏是最考验设备和带宽的应用，特别是高配置需求的网络游戏，它会大大的降低移动设备的使用寿命。为了解决这些问题，很早之前就有人提出了用云游戏代替传统互联网游戏的方法[2,9,14]，Lampe 等[15]对能量、延迟和成本做了一些研究，发现这种体系是可行的。现在云游戏平台也在 IOS、安卓谷歌平台开始普及。

云移动渲染(Cloud Mobile Rendering，CMR)为移动设备提供了丰富的多媒体 App，但是想要获得用户体验却是一项挑战[12]。为了加强用户体验，云移动 VR(Cloud-Mobile Convergence for Virtual Reality，CMCVR)[16]在 CMR 的基础上增加 VR 技术，为手机用户带来逼真的 VR 体验。

本文将 CMCVR 带入手机浏览器，且不需要加载任何插件，避免了手机应用的下载和更新环节，是一种将 VR 体验带入手机浏览器的轻量级应

DOI: 10.16182/j.issn1004731x.joss.19-VR0477

第 32 卷第 7 期 系统仿真学报 Vol. 32 No. 7
2020 年 7 月 Journal of System Simulation Jul., 2020

用框架，为 VR 技术在 Web 中的开发做出了贡献。

1.2 移动 VR 应用

随着移动 VR 应用的发展，利用有限资源的移动设备进行 3D 视频游戏的 App 也随之增多，用户需求也越来越复杂[17]。由于 3D 视频游戏在移动端图像渲染的过程中会消耗大量的计算资源[18-21]，Kang 等[22]提出了一种基于 Docker 的视频游戏计算卸载的 VR 移动框架。它在云渲染的基础上添加了一种单对单的 Off-loading 传输方式，加快了传输速度、降低了移动端计算成本，但是图像流在传输多组高质量连续图像时压缩质量和压缩效率低下，并且只能在特定的移动设备上下载 Off-loading 数据接收插件进行单对单传输。

Cao 等[5]提出了一种将头戴式 VR 设备和智能手机相结合的框架。它将高硬件成本的 VR 体验带入了低硬件成本的移动设备，但是移动端渲染对手机造成了过重的负载，一些对设备有过高硬件需求的应用无法用此框架运行。同时移动端需要下载相应的 App 插件，造成了手机资源的堆积与损耗。

Nguyen 等[23]提出了一种基于 SHVC Tile 的 360°移动 VR 视频流传输方式。它将 Kang 等[22]提出的基于图像流的 Off-loading 传输方式改进成了基于 SHVC (H265)视频流的 Off-loading 传输方式，并结合 Cao 等[5]的移动 VR 框架实现了三维场景立体显示与交互，弥补了 Kang 等[22]在普适性和便携性上的不足。然而它的客户端依然要下载 Off-loading 数据接收插件，并且 SHVC (H265)压缩率尽管很高，减小了传输带宽占用，但是相比 H264，它的压缩和解压时间更长，是 H264 的 5 倍左右，降低了移动 VR 应用在客户端的刷新频率和交互实时性。

林定等[24]以 WebGL 为基础，集成 JavaScript 中的开源图数据库 Neo4j 和图形库 ThreeJS，采用浏览器-服务器(B/S)模式，实现了一种基于 WebVR 的网络数据三维树形可视化，解决了传统 VR 应用插件繁多问题，为利用虚拟现实技术探索网络数据的潜在价值提供有效的技术手段。然而其开发所用的硬件设备依然停留在 Oculus 设备上，尽管近期 Google 将 WebVR 应用于安卓手机的 Chrome 浏览器，但是基于 WebGL 的本地渲染依然有着一定的硬件成本和限制，无法在手机上实时渲染出有过高资源需求的可交互 VR 三维场景。

本框架结合云渲染、手机浏览器和 VR 技术，以手机 Web 作为客户端，传输光线跟踪渲染的 H264 视频块流(兼容图像流)，利用消抖算法处理交互数据，通过头戴式 VR 设备实现三维场景立体显示与交互。移动端无需下载插件和安装包，是一种集便携性、兼容性、高效性、实时性和真实感于一体的低成本移动 VR 网页应用框架。

2 总体框架

云渲染将渲染端设置在云服务器上，降低了客户端资源损耗，让客户端能显示本地无法渲染的高质量场景。而云游戏模式则是在云渲染的基础上增加了交互功能。本文扩展了这种模式，如图 1 所示，在服务器图像处理阶段和客户端显示与交互阶段采用了本文提出的面向手机的云端 VR 三维场景立体显示与交互框架。

首先，在云服务器端用光线跟踪渲染出高质量三维场景图像，然后将多张图像通过 H264 视频压缩算法进行压缩。由于客户端设立在手机 Web 上，无法直接显示 H264 格式的视频流，所以在服务器端需要结合 FFMEPG 将 H264 视频流封装成 MP4 (MPEG-4)视频块。最后将 MP4 视频块流传到手机 Web 上排队显示。同时在手机网页上将手机陀螺仪和加速度传感器获取的数据作消抖处理，传至渲染端完成交互。本框架算法步骤分为如下 2 个部分：

(1) 服务器部分

step 1：服务器监听到连接请求后，将客户端发送过来的密匙 Sec-WebSocket-Key 编码成 SHA-1 哈希值，并返回哈希值的 base64 编码，客户端验证编码后确认双方握手。

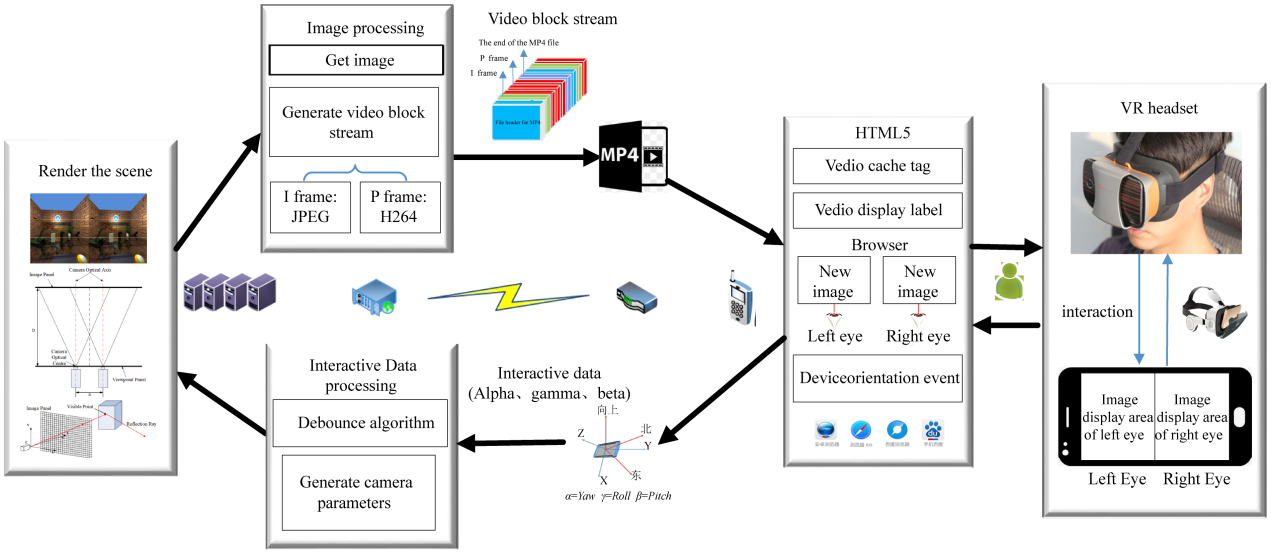


图 1 本文技术路线

Fig. 1 Technology overview

step 2：服务器端创建多线程并行执行。主线程作为渲染端，设置渲染完成标记 *A*，视频块封装完成标记 *B*，使 *A*=false、*B*=false。根据握手后接收到的手机屏幕像素宽和高确定渲染图像大小和 MP4 视频头文件信息。创建 MP4 视频容器 Buffer，将 MP4 视频头文件信息写入 Buffer，同时开启发送线程和交互线程，并行执行 step 3~5。

step 3：根据 *A* 判断是否开始渲染，若 *A*=false 则进行三维场景渲染，完成渲染 *A*=true；根据 *A* 和 *B* 判断是否进行压缩，若 *A*=true、*B*=false，利用 FFMPEG(一个开源免费跨平台的视频和音频流方案)将渲染的图像压缩成 H264 视频流的帧，放入 MP4 视频容器 Buffer，*A*=false；循环 step 3 直至渲染线程结束。

step 4：发送线程判断当前 H264 视频块内包含压缩帧数是否达到 FPS (frame per second)的六分之一。若达到则 *B*=true，为 MP4 视频容器 Buffer 添加尾文件并调整头文件。之后将 Buffer 按 WebSocket 协议进行封装，并传输给手机客户端。同时初始化 MP4 视频容器和压缩编码器参数，*B*=false；循环 step 4 直至发送线程结束。

step 5：在交互线程中如果接收到交互数据，并且 *A*=true，则将接收到的交互数据传至渲染端，渲染端根据获取到的交互数据确定场景中摄像机参数，然后以 step 3 中修改过的摄像机参数为准进行图像渲染；循环 step 5 直至交互线程结束。

(2) 客户端部分

step 1：在手机 Web 上创建 Video 标签作为视频播放容器，创建 div 标签，将属性设为可见，并将 Video 标签插入 div 标签。将 WebSocket 接收到的第一个视频块生成 URL 赋值给 Video 标签的 src 属性，标签铺满全屏在手机浏览器上显示。执行下一步。

step 2：通过触发式接收函数创建 Video 标签作为视频缓存容器，创建 div 标签，属性设为隐藏，并将 Video 标签插入 div 标签。将 WebSocket 接收到的视频块生成 URL 赋值给 Video 标签的 src 属性进行预缓存。执行下一步。

step 3：为正在播放视频的 Video 标签添加 ended 事件，当视频播放结束时触发。事件内容为：将事件对象 Video 标签移除，step 2 中 Video 标签的父标签 div 标签属性由隐藏变为显示，并开始播放视频。转至 step 2；同时执行下一步。

step 4：将手机放入头戴式 VR 设备，用户通过佩戴此设备观看三维场景。当用户头部运动时，执行下一步。

https://dc-china-simulation.researchcommons.org/journal/vol32/iss7/17
DOI: 10.16182/j.issn1004731x.joss.19-VR0477

step 5：在手机 Web 上调用设备方向事件 deviceorientation，通过此事件实时获取陀螺仪设备方向数据，利用消抖算法处理方向数据，之后将数据传输到服务器。转至 step 4。

2.1 本文传输框架

本文借助云渲染，将 VR 技术应用于手机网页，不需要下载任何插件。这得益于 HTML5 强大的功能和云渲染的优越性，它不仅改善了手机发热和寿命缩短问题，也使很多手机渲染引擎无法渲染的效果在手机上得以呈现。同时本框架只通过视频块流传输当前可见场景，相比传统移动 VR 应用的 360°视频流，本框架极大地减少了传输带宽占用以及视频压缩和解压时间。而本框架通过头部追踪交互，也可以观看传统移动 VR 应用的 360°全景视频，为全景视频提供了更佳的渲染和呈现方式。本文实验部分所使用的 4 种场景皆实现了 360°全景立体效果。

2.1.1 服务器端

本文使用计算能力强大的云集群作为服务器，同时为了使应用程序的互动逼真度上升到全新高度，渲染端使用手机渲染引擎无法使用的 Optix 光线跟踪技术进行渲染。

服务器端 MP4 视频块流生成流程如图 2 所示。先通过光线跟踪渲染技术在渲染端渲染出一组高质量立体图像。再将生成的 RGB 图像发送至图像处理端，利用 FFMPEG 进行 H264 编码，生成 YUV 颜色编码的视频流。最后将一定数量的 H264 视频流封装成 MP4 视频块，将视频块按 WebSocket 协议编码后发送至客户端。同时接收客户端交互信息，调整渲染时摄像机的视点位置。

2.1.2 MP4 视频块流传输

图像的压缩算法中，兼容性、压缩效率和压缩效果等方面都比较好的是 JPEG 算法。但是当多组图像进行压缩时，JPEG 算法的压缩效率和速度远远比不上视频压缩，而且图像流会因为网络波动出现卡屏现象。然而另一方面，视频压缩形成的视频流实时性差。如 HLS 协议，由于要将原视频进行 ts 文件切块，导致延迟通常在 30 s。而实时性相对较好的视频流，如 RTMP 协议和 HTTP-FLV 协议都需要下载对应的 App 插件，兼容性差、移动端内存资源损耗高，延迟也有 3~4 s。所以本框架结合视频流和图像流的优点提出了一种视频块流的传输方式。它同时拥有视频流的流畅性和图像流的实时性，并且直接应用 HTML5 标准，不需要借助其它任何插件。

MP4 视频块流利用 H264 视频压缩算法对多组图像进行压缩，并将一定数量的图像封装成视频块。通过限制 MP4 视频块长度使延迟控制在 1 s 以内，实时性比 HLS 协议、RTMP 协议和 HTTP-FLV 协议都更强。并且应用 HTML5 标准在 Web 上直接显示，比需要 Flash 支持的 RTMP 协议和 HTTP-FLV 协议兼容性更好、开发成本更低、开发效率更高。

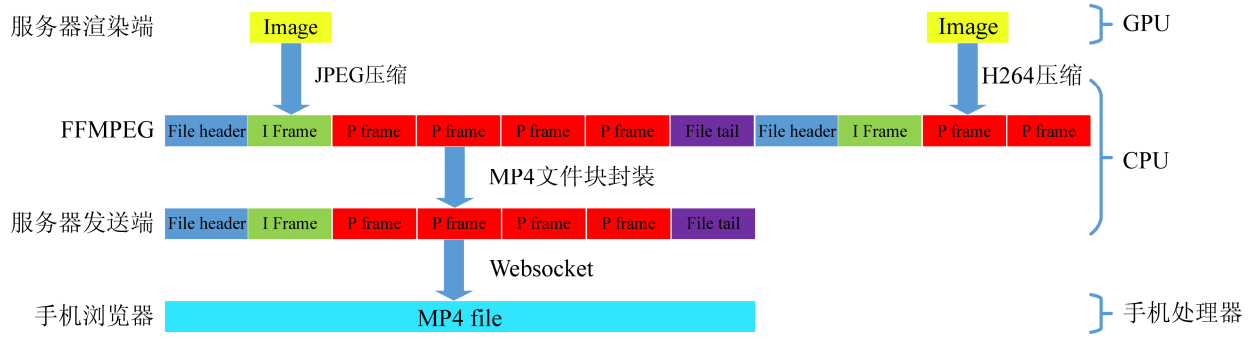


图 2 MP4 视频块流生成流程
Fig. 2 Generation of MP4 stream

MP4 视频块流主要采用了以下几种方法对视频数据进行压缩。包括：

帧内预测压缩：解决空域数据冗余问题。

帧间预测压缩：(运动估计与补偿)，解决时域数据冗余问题。

整数离散余弦变换(DCT)：将空间上的相关性变为频域上无关的数据，然后进行量化。

CABAC 压缩：一种高效的无损压缩技术。

MP4 视频块封装：将一定帧数的 H264 视频流封装成 MP4 文件块。

RGB 图像经过压缩后可生成 I 帧、P 帧和 B 帧：

I 帧：关键帧，采用帧内压缩技术(类似于 JPEG 算法)。I 帧生成流程如图 3 所示。

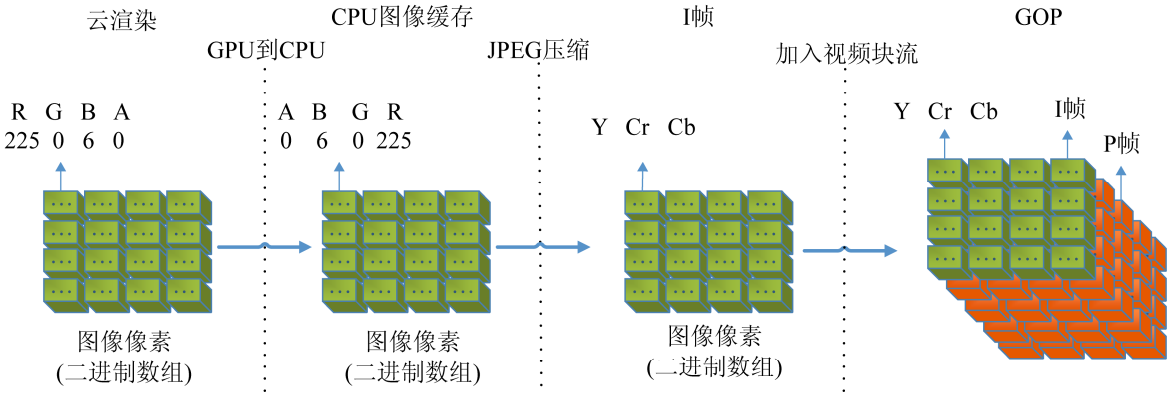
P 帧：向前参考帧，采用帧间压缩技术，在压缩时，只参考前面已经处理的帧。P 帧生成流程如图 4 所示。

B 帧：双向参考帧，采用帧间压缩技术，在压缩时，它既参考前面的帧，又参考后面的帧。由于 B 帧计算成本过高，所以本框架不使用 B 帧。

2.1.3 客户端

本框架客户端面向手机 Web 进行开发，与 WebVR 一样将高资源需求的 VR 体验带入低成本开发的手机网页。本框架用云渲染代替 WebGL 渲染，解决了 WebVR 本地渲染的弊端；以手机自带传感器实现 VR 交互，降低了硬件成本，使交互更加自然。

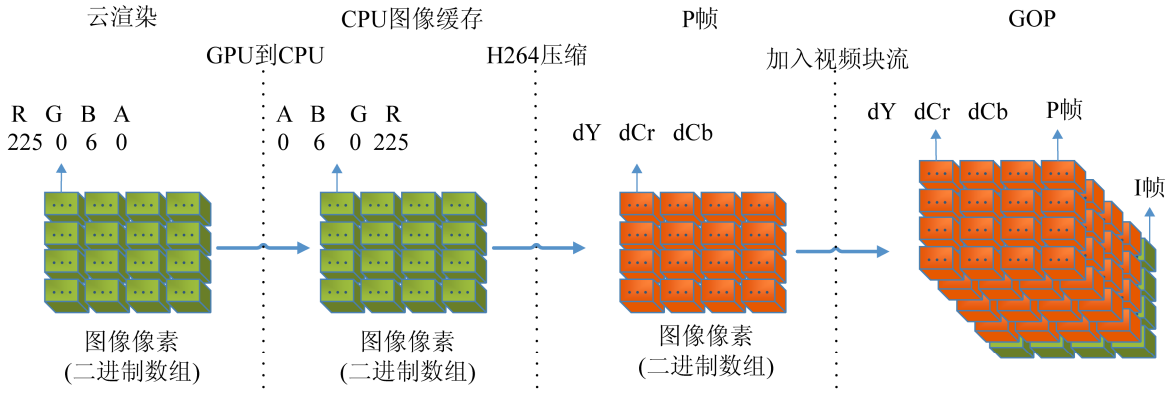
浏览器所使用的操作指令是 HTML5 标准，常用视频格式对浏览器的适用情况如表 1 所示。



GOP: 两个I帧之间的一个图像序列，一个图像序列中只有一个I帧。

图 3 I 帧生成过程

Fig. 3 Generation of frame I



GOP: 两个I帧之间的一个图像序列，一个图像序列中只有一个I帧。

图 4 P 帧生成过程

Fig. 4 Generation of frame P

第 32 卷第 7 期 系统仿真学报 Vol. 32 No. 7
2020 年 7 月 Journal of System Simulation Jul., 2020

表 1 常用视频格式对浏览器的适用情况
Tab. 1 Application of Common Video formats to browsers

格式	文件	描述
MPEG	.mpg .mpeg	MPEG 是电影专家组开发的，第一个在网络上流行的视频格式，所有浏览器都支持，但是在 HTLM5 里面不直接支持(Look MP4)。
AVI	.avi	AVI(Audio Video Interleave)是 Microsoft Inc.开发的，通常用于摄像机和电视机，在 Windows 上能很好的播放，但是浏览器不支持。
WMV	.wmv	WMV(Windows Media Video)是 Microsoft Inc.开发的，通常用于摄像机和电视机，在 Windows 上能很好的播放，但是浏览器不支持。
QuickTime	.mov	QuickTime 是 Apple Inc.开发的，通常用于摄像机和电视机，在 IOS 上能很好的播放，但是浏览器不支持(Look MP4)。
RealVideo	.rm .rvm	RealVideo 是 Real Media 公司开发的，允许在低宽带下的视频流，它用于在线视频和网络电视，但是浏览器不支持。
Flash	.swf .flv	Flash 是 Macromedia 公司开发的，通常需要额外的组建(插件)才能在浏览器上播放。
Ogg	.ogg	Theora Ogg 是 Xiph.Org 基金会开发的。HTML5 支持。
WebM	.webm	WebM 由 web giants，Mozilla，Opera，Adobe 和 Google 开发。HTML5 支持。
MP4 (MPEG-4)	.MP4	MP4 由电影专家组基于 QuickTime 开发，通常用于比较新的视频摄像机和电视机。HTML5 的所有浏览器都支持。YouTube 建议使用这个格式。

本框架客户端使用 Video 标签作为显示载体，手机 Web 将接收的 MP4 视频块流缓存生成 URL，并嵌入 Video 标签进行显示。此时会出现一个问题，Video 标签在预缓存时视频窗口并不会显示视频，在不同的手机系统上会有白屏、黑屏或者加载圈等情况，这导致了两个视频块交替时播放不连续。因此本框架在客户端使用了一种队列显示模式：首先，在手机 Web 上定义一个 Video 标签，显示当前视频块并播放。之后，在手机 Web 接收到新视频块时，新建一个隐藏的 Video 标签，将接收到的视频块嵌入并进行预加载。最后，当正在播放的视频块播放完成后，释放当前视频块，由后一个加载完成的视频块继续播放。由于进行了预加载，所以两个视频块可以连续播放。客户端显示效果如图 5 所示。

图 5 在客户端手机浏览器上的显示效果
Fig. 5 Final result in mobile phone browser

2.2 移动端三维立体显示与交互

2.2.1 三维场景的立体显示

由于要实现三维场景的立体显示，所以需要将场景中两个不同视角的图像显示出来。本文在场景中设置了一组立体摄像机，关于立体摄像机的设置采用了 Chen 等[25]提出的虚拟场景中立体摄像机参数的设置方法。应用平行光轴模型作为立体摄像机的模型，并在其工作的基础上对摄像机的设置进行了改进。

平行光轴模型如图 6 所示，点 $\boldsymbol{T}$ 为注视点坐标点；左摄像机的坐标点 $\boldsymbol{C}$ 相对摄像机系统坐标点 $\boldsymbol{O}$ 沿向量 $\boldsymbol{V}$ 的负方向平移了二分之一个 a；右摄像机的坐标点 $\boldsymbol{C}'$相对摄像机系统坐标点 $\boldsymbol{O}$ 沿向量 $\boldsymbol{V}$ 的正方向平移了二分之一个 a。由此可确定 $\boldsymbol{C}$ 点和 $\boldsymbol{C}'$点的坐标，计算公式如下：

$$\boldsymbol{C}=\boldsymbol{O}-\frac{a}{2}\frac{\boldsymbol{V}}{\|\boldsymbol{V}\|} \tag{1}$$

$$\boldsymbol{C}'=\boldsymbol{O}+\frac{a}{2}\frac{\boldsymbol{V}}{\|\boldsymbol{V}\|} \tag{2}$$

式中：点 $\boldsymbol{O}$ 为摄像机系统在虚拟世界坐标系下的坐标点；a 为左摄像机和右摄像机的距离；向量 $\boldsymbol{V}$ 为摄像机系统观察方向向量和向上方向向量外积；点

C 为左摄像机的坐标点；点 ***C'*** 为右摄像机的坐标点。

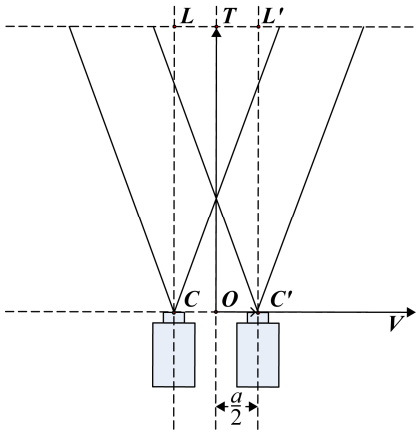


图 6　立体摄像机系统示意图
Fig. 6　Schematic diagram of Stereo camera

最后，应用 OptiX 光线跟踪引擎根据立体摄像机参数进行场景的渲染，得到一组立体图像。通过上文所提到的框架将图像传输到移动端，完成三维场景的立体显示。

2.2.2 三维场景的交互

(1) 手机设备方向事件

现在的智能手机普遍配备了重力传感器和方向传感器(陀螺仪)。HTML5 标准提供了两个事件，获取手机的设备方向数据，它们分别是 orientationchange 事件和 deviceorientation 事件。

智能手机有自己的设备坐标系，并且与手机固定在一起，无论手机平移还是转动，手机设备坐标系都不会相对手机发生改变。如图 7 所示，当用户正对手机屏幕时，手机设备坐标系的 Z 轴垂直屏幕指向用户；Y 轴处于手机屏幕平面内指向手机顶部；X 轴处于手机屏幕平面内指向手机右侧。同时手机设备坐标系是笛卡尔坐标系，三条坐标轴两两垂直。

本文使用 deviceorientation 事件获取手机 VR 头部交互数据，分别是手机陀螺仪和重力传感器捕捉的 3 个欧拉角。当手机旋转到设备坐标系与世界坐标系重合时，3 个旋转角均等于 0°。由图 7 可见，这 3 个手机设备方向角的运动变化规律为：α 等于手机偏航角 *Yaw*，范围是 0°~360°，当 X 轴偏向 Y 轴时，α 值增加；β 等于手机俯仰角 *Pitch*，范围是 –180°~180°，当 Y 轴偏向 Z 轴时，β 值增加；γ 等于手机翻滚角 *Roll*，范围是–90°~90°，当 Z 轴偏向 X 轴时，γ 值增加。

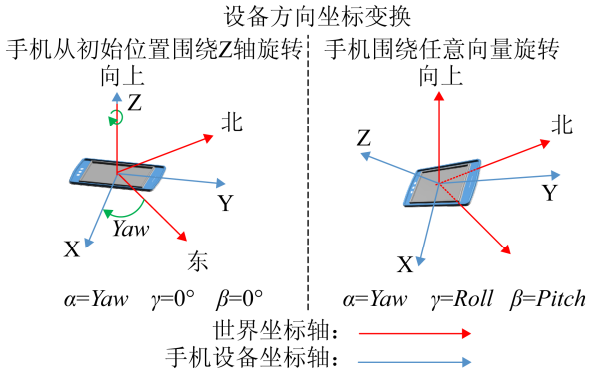


图 7　设备坐标系旋转示意图
Fig. 7　Rotation of device coordinate

根据欧拉旋转定理(Euler's rotation theorem)，由世界坐标系和设备坐标系可以定向出一组 4 个数字(四元数)的旋转矢量 $\boldsymbol{q}=[q_0, q_1, q_2, q_3]^{\mathrm{T}}$，由 Rodrigues 旋转公式求得旋转矩阵 ***Q***，结合世界坐标可以得到手机设备坐标系(两坐标系取同一原点)。手机传感器捕捉的 3 个设备方向角就是四元数表达式中的后 3 个数，四元数表达式为：

$$\cos\left(\frac{\alpha}{2}\right),\ x\sin\left(\frac{\alpha}{2}\right),\ y\sin\left(\frac{\alpha}{2}\right),\ z\sin\left(\frac{\alpha}{2}\right) \quad (3)$$

式中：α 为等效旋转角，等效旋转轴方向向量 $\boldsymbol{K}=[x, y, z]^{\mathrm{T}}$。四元数存储了旋转轴和旋转角的信息，它能方便的描述刚体绕任意轴的旋转。

(2) 渲染端头部追踪交互

要实现与三维场景的交互，就要通过客户端的交互数据确定服务器的渲染图像。本文实现了与虚拟场景的头部追踪交互，可让用户 360°全景观看立体画面。先将现实中的世界坐标系和虚拟场景中的世界坐标系对齐，再结合欧拉旋转定理和手机端消抖处理后的 3 个设备方向角求出手机设备坐标系方位，最后确定与手机设备坐标系对齐的虚拟摄像机系统方向参数，完成对场景中立体摄像机的旋转。本框架还拥有虚拟场景漫游、虚拟场景互动等其它交互功能，本文仅实现了头部追踪交互。

DOI: 10.16182/j.issn1004731x.joss.19-VR0477

渲染场景中的摄像机拥有向上方向向量和向前观察向量。本文计算了从物理世界坐标系到 VR 场景虚拟世界坐标系的变换矩阵 $\boldsymbol{M}$。根据 $\boldsymbol{M}$ 的变换，计算物理世界坐标系 Z 轴负方向在 VR 场景虚拟世界坐标系上对应的单位向量 $\boldsymbol{W}_0$ 和物理世界坐标系 X 轴负方向在 VR 场景虚拟世界坐标系上对应的单位向量 $\boldsymbol{U}_0$，之后将摄像机向上方向向量初始化为 $\boldsymbol{U}_0$，把摄像机向前观察向量初始化为 $\boldsymbol{W}_0$。

完成 VR 场景中摄像机位置的初始化后，由(1)中提到的旋转矩阵 $\boldsymbol{Q}$ 确定摄像机旋转位置。根据手机旋转传感器捕捉的 3 个设备方向角计算出旋转矩阵 $\boldsymbol{Q}$。根据 $\boldsymbol{Q}$ 的变换，计算手机设备坐标系 X 轴方向在物理世界坐标系中对应的单位向量 $\boldsymbol{U}_1$ 和手机设备坐标系 Z 轴负方向在物理世界坐标系中对应的单位向量 $\boldsymbol{W}_1$。之后通过变换矩阵 $\boldsymbol{M}$，计算出向量 $\boldsymbol{U}_1$ 和 $\boldsymbol{W}_1$ 在 VR 场景虚拟世界坐标系中对应的向量 $\boldsymbol{U}_2$ 和 $\boldsymbol{W}_2$。并把场景中摄像机的向上方向向量更新为 $\boldsymbol{U}_2$，把场景中摄像机的向前观察向量更新为 $\boldsymbol{W}_2$，完成摄像机旋转位置的更新。

如图 8 所示，通过确定好的摄像机旋转位置，结合 2.2.1 所述的平行光轴模型，可以将 VR 场景立体摄像机中左右两个摄像机的参数计算出来，并且与手机设备坐标系和手机屏幕上的左右眼画面显示区域形成对应。

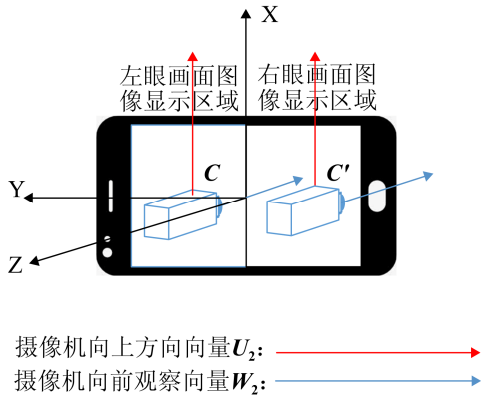


图 8 VR 场景中左右摄像机方向向量与手机设备坐标系对齐示意图

Fig. 8 Relationship between device coordinate and camera direction in VR scene

在图 8 中，以 $\boldsymbol{C}$ 点作为向量 $\boldsymbol{U}_2$ 和 $\boldsymbol{W}_2$ 的起点，生成左摄像机的方向向量参数；以 $\boldsymbol{C}'$点作为向量 $\boldsymbol{U}_2$ 和 $\boldsymbol{W}_2$ 的起点，生成右摄像机的方向向量参数。最终通过这些参数进行三维场景的渲染，实现手机 VR 三维场景的显示与交互。

(3) 抖动问题处理

由于外界各种干扰和手机自身设备缺陷等因素，手机通过陀螺仪和重力传感器获取到的设备方向角会有一定误差，这个误差在 0.2°以内，并且静止时 20 s 内误差值偏移不超过 0.1°，误差的变化幅度小，处于人眼视觉无法察觉的范围之内。但是，手机陀螺仪非常敏感、外界干扰因素复杂，各种干扰变化累积后，会导致设备方向角的测量值变化频率变高，高频率的测量值抖动尽管幅度小，却能被人眼察觉，从而在手机静止时造成手机画面抖动。为了解决这个问题，本文对设备方向数据进行了消抖处理。

在手机静止时设备方向角整数位是不变的，主要是十分位以后的数值发生不规则高频率变化，并带动可交互场景抖动。本文将手机设备方向角取到测量值的十分位，后面位数忽略不计，以此消除抖动。但是，如果设备方向角恒取到测量值十分位，则当测量值十分位以后的数带动十分位发生变化时，会消抖失败，因此此时设备方向角需要再次进行消抖处理，如公式(4)：

$$A_k = B_k - A_{k-1} + C_k + A_{k-1} \tag{4}$$

式中：k 为当前时刻；k–1 为前一时刻；A_n 为 n 时刻手机设备方向角消抖处理后的值；B_n 为 n 时取到小数点后第一位的设备方向角测量值；C_n 为 b_n–B_n 之后四舍五入的值；b_n 为 n 时刻取到小数点后第二位的设备方向角测量值。

由于在手机跟着头部不断运动时，手机设备方向角变化幅度远大于测量值的误差，因此设备方向角度值在变化过程中覆盖了干扰值，手机画面不会出现人眼可见的抖动。

3 实验结果与分析

本实验在服务器端利用高性能台式电脑替代云集群，配合小米智能手机和小米路由器实现本文框架。同时对比图像流和视频块流两种框架的优劣，并根据得到的实验数据分析本框架相比现有移动 VR 应用所具备的优势以及未来需要弥补的不足。

本实验用来替代云集群作服务器的高性能电脑配置为：双核处理器；内存(RMA) 128 GB；64 位操作系统；CPU 型号为 Intel(R) Xeon(R) E5-2620 v4；GPU 型号为 NVIDIA Quadro P5000。本实验以 1 920×1 080 屏幕的小米 Note 标配版智能手机作为移动端；小米路由器 3C 作为网络节点。实验步骤如下：(1) 用高性能电脑代替云集群渲染，并对实时渲染出的 1 920×1 080 图像进行处理；(2) 在实验室内用小米路由器单独搭建内网，用任意电脑通过 IIS 发布网页，由小米手机浏览器打开框架客户端，并运行框架；(3) 将手机放入头戴式 VR 设备，分别向 4 位使用者展示三维场景的 360°全景立体画面。在他们进行头部追踪交互时，记录客户端刷新频率、传输带宽占用和交互延迟时间。

3.1 图像流和视频块流对比分析

本实验以 4 种光线跟踪渲染的场景为样本，分别测试了图像流和视频块两种传输模式下的客户端刷新频率、传输带宽占用和交互延迟时间。同时测试服务器仅绘制时 4 种场景的渲染刷新频率，以此进行对比分析。

3.1.1 图像流和视频块流实验数据

不同场景在服务器仅绘制、视频块流和图像流 3 种模式下的刷新频率(FPS)和网络带宽占用如图 9 所示。图 9(a)和(b)选取了 2 个简单场景，图 9(c)和(d)选取了 2 个复杂场景。为了凸显实验的严谨性，这 4 个场景的数据都是在用户不断进行头部追踪交互时获取的。由于存在网络波动，网络带宽占用取出现频率最高的值。

图 10 是 4 种场景分别在 PC 端渲染和手机 Web 端显示的效果对比图。图 10(a)、(c)、(e)和(g)为 PC 端基于光线跟踪的渲染效果图；图 10 (b)、(d)、(f)和(h)为有框架时手机 Web 端基于云渲染的显示效果图。可以看出手机端网页显示效果跟 PC 端渲染出来的效果几乎差不多。

图 10 说明本框架可以显示传统移动 VR 应用无法渲染的基于光线跟踪的 VR 三维场景。同时，在手机端多个浏览器上无插件使用，表明本框架具有跨平台、低成本、使用效率高的优势。

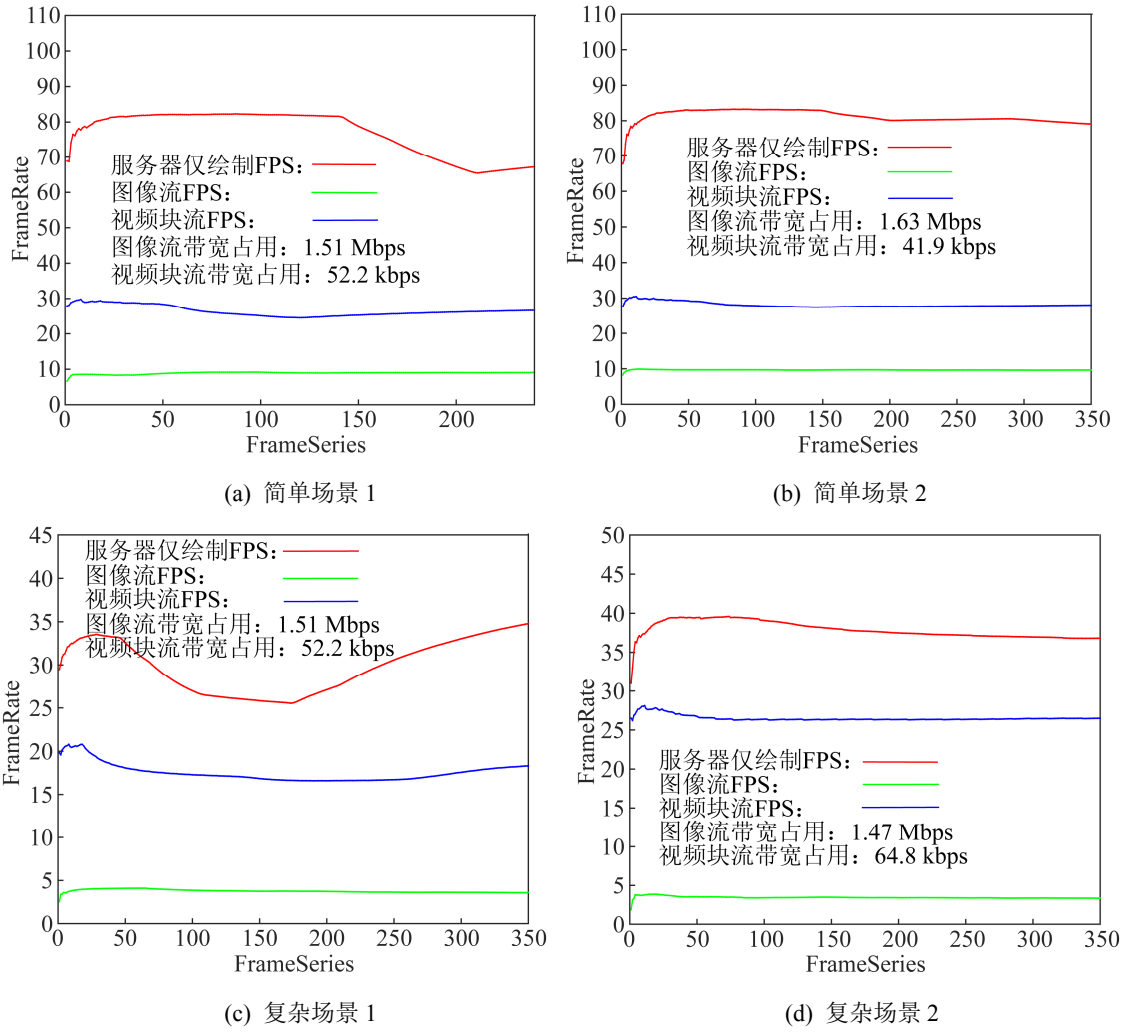
3.1.2 客户端刷新频率分析

从图 9 可以看出，图像流在传输复杂场景和简单场景时，FPS 值都低于 10 帧/秒，而视频块流的 FPS 值最高可达 30 帧/秒；对比图 9(b)和图 9(d)，从复杂场景变换到简单场景，服务器仅绘制时的 FPS 值提升了 2 倍左右，视频块流只提升了 1.2 倍左右，图像流尽管提升了 2.5 倍左右，但是图 9 中图像流的 FPS 值全部不超过 10 帧/秒，所以两种传输方式的 FPS 值都提升的很少。因此得出结论：服务器渲染速度不是限制这两种传输方式在移动端刷新频率的主要原因，并且视频块流的客户端刷新频率优于图像流。

尽管视频块流能达到 30 FPS，但也远比服务器仅绘制时的 FPS 值低，且根据中国信通院的虚拟现实白皮书中通常认为渲染计算的频率达到 60 FPS 才能保证初步沉浸的效果。而影响客户端刷新频率的因素有 3 个：网络延时、场景渲染、视频压制。本框架使用的光线跟踪渲染技术、H264 视频压制算法和 WebSocket 网络传输协议，都是现今非常热门和高效的算法技术。本实验旨在验证框架的可行性，并寻找其性能影响因素，所以服务器未用高性能集群设备，传输部分也仅使用低性能路由器代替高性能信号发射塔。

https://dc-china-simulation.researchcommons.org/journal/vol32/iss7/17
DOI: 10.16182/j.issn1004731x.joss.19-VR0477

第 32 卷第 7 期 系统仿真学报 Vol. 32 No. 7
2020 年 7 月 Journal of System Simulation Jul., 2020

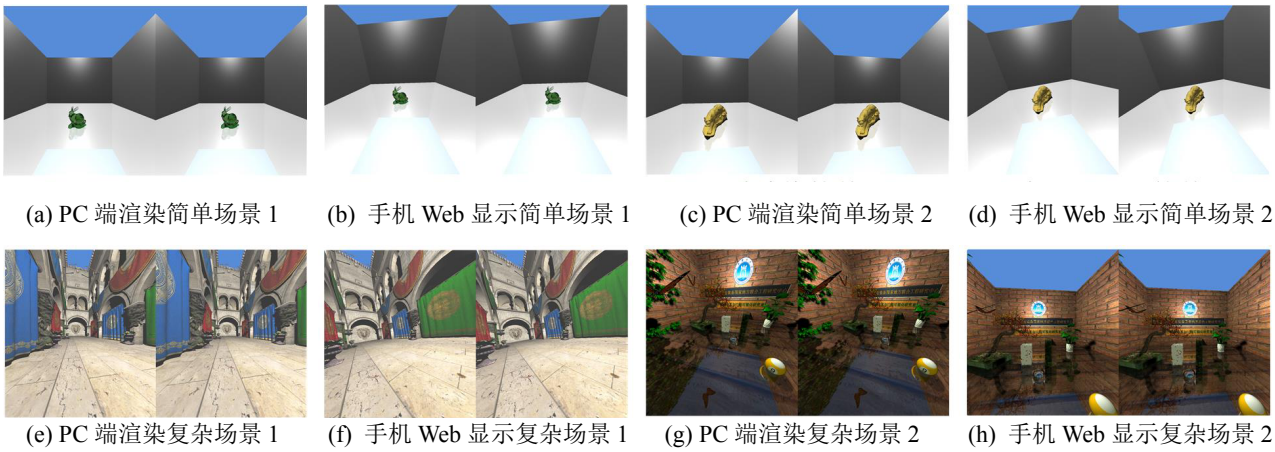


(a) 简单场景 1 (b) 简单场景 2

(c) 复杂场景 1 (d) 复杂场景 2

图 9 FPS 值变化曲线(附网络带宽占用情况)

Fig. 9 Curve of FPS value depended on time (occupation of network bandwidth)



(a) PC 端渲染简单场景 1 (b) 手机 Web 显示简单场景 1 (c) PC 端渲染简单场景 2 (d) 手机 Web 显示简单场景 2

(e) PC 端渲染复杂场景 1 (f) 手机 Web 显示复杂场景 1 (g) PC 端渲染复杂场景 2 (h) 手机 Web 显示复杂场景 2

图 10 4 种场景的效果对比图

Fig. 10 Comparison of effects of four scenarios

Google 在 2019 年发布了云游戏平台 Google Stadia，其在 25 Mbps 宽带下支持 HDR、4K 分辨率和 60 FPS，未来可实现对 8 K、120 FPS 的支持。本框架与 Stadia 平台一样主要基于云渲染而设计，

如果以商业级高性能集群和高质量传输设备为基础，其客户端刷新频率能达到 60 FPS，甚至更高；分辨率也能达到 4 K，甚至更高。同时，随着 5G 时代的到来，以其高带宽、低延时、高并发的特点，可以大大缓解网络延时问题，保证用户能体验到高质量的沉浸式 VR 效果。因此得出结论：本实验未达到沉浸效果的主要原因是硬件层面的不足。

3.1.3 带宽占用分析

图 9(c)中图像流带宽占用 1.51 Mbps，是视频块流带宽占用的 29.6 倍，极大地消耗了客户端网络资源。图 9 其他场景曲线图中图像流带宽占用最低也在 1.47 Mbps，而视频块流带宽占用最高只有 64.8 kbps。结合两种传输方式的 FPS 值相差 3~4 倍可以得出结论：带宽占用的多少会影响框架在客户端的刷新频率，并且视频块流的带宽占用情况远优于图像流。

3.1.4 压缩算法分析

图像流使用的压缩算法是 JPEG 算法，视频块流使用的压缩算法是 H264 格式的视频压缩算法。在压缩速度和压缩程度方面，H264 视频压缩算法优于 JPEG 压缩算法。从图 9 可以看出，无论是简单场景还是复杂场景，视频块流的 FPS 值都比图像流的 FPS 值高 2~4 倍，并且场景越复杂，相差越明显。但是，H264 视频压缩算法是有损压缩算法，JPEG 算法是无损压缩算法，从图 10 可以看出，图像流的手机网页端显示效果跟服务器渲染端渲染效果在视觉上无差别；并且图像流在排除网络波动因素下近乎零延迟，而视频块流由于要封装视频块故存在极短的延迟。因此得出结论：压缩算法会影响框架在客户端的刷新频率和显示效果，并且视频压缩算法动态显示效果好、存在极短延迟，图像压缩算法静态显示效果好、近乎零延迟。

综上所述，本框架客户端刷新频率与传输方式、带宽占用、压缩算法和服务器 CPU 运算能力有关，显示效果与服务器渲染内容和压缩算法有关。而本实验在手机上实现了基于光线跟踪的 VR 三维场景立体显示与交互，这种高资源需求的三维场景在移动端是无法渲染的，因此本框架对智能手机的运算能力、内存大小等硬件要求很低。

3.2 手机设备方向角抖动分析

本实验以手机设备俯仰角作为测试角，分别测试了在静止和运动两种状态下，设备俯仰角的抖动数据和消抖数据，并对其进行抖动分析。

3.2.1 手机静止状态时设备俯仰角实验数据

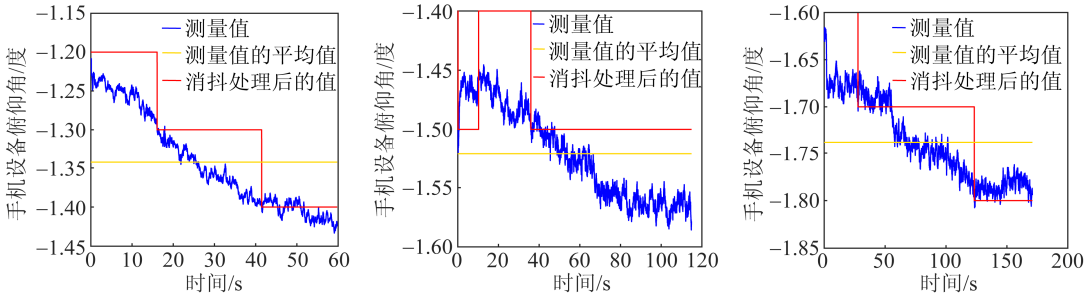
图 11 是在手机平放桌面静止时，分别获取的 1、2、3 分钟 3 组设备俯仰角测量值和消抖处理后的值，并对 3 组测量值分别计算平均值和标准差。

3.2.2 手机设备俯仰角抖动分析

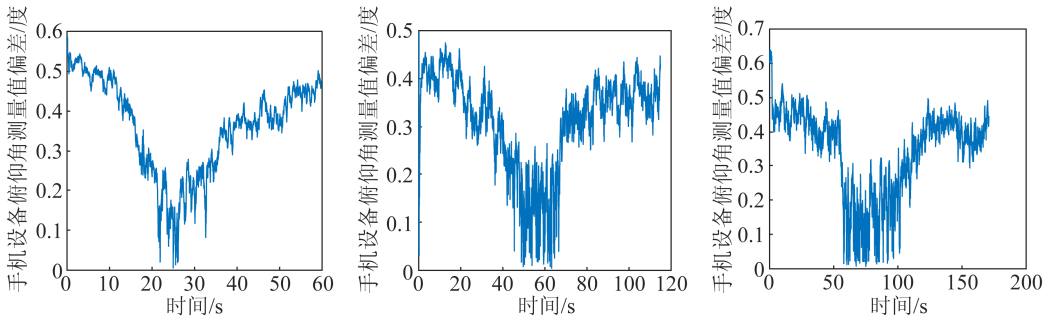
从图 11(a)中可以看出，在手机平放桌面静止状态下，手机传感器测量的设备方向数据会在抖动中呈下降趋势，整体趋势变化在 0.2°左右，相邻测量值相差在 0.01°以内，而消抖处理后的值升降阶梯是 0.1°，整体趋势呈阶梯下降，趋势变化为 0.2°。从图 11(b)中可以看出，方向角测量值标准差基本控制在 0.5°以内，其受干扰造成的误差幅度很小，说明消抖处理后干扰对角度造成的误差幅度也一样不大，都属于人眼无法感知的范围；消抖处理后的角度值整体趋势的变化幅度小于测量值整体趋势的变化幅度，说明方向角整体波动得到了轻微的改善。从图 11(a)可以看出两曲线的最大区别在于消抖处理后的数据没有高频率抖动，而在手机平放不动时，未消抖处理的测量数据会带来画面抖动，使用消抖处理后，人眼可察觉的手机画面抖动消失了。因此得出结论：当手机静止不动时，设备测量的方向数据会使手机显示的可交互场景发生抖动，其原因是测量值的高频率变化。同时，测量值误差变化幅度对用户视觉体验没有影响。

从图 12(a)和(c)可以看出，当手机不断运动时，设备俯仰角测量值与消抖处理后的值几乎重合；图 12(b)中某随机区间内重合部分多达 97.5%，图 12(d)中为 100%。因此得出结论：在手机运动时，干扰值被设备方向角变化值覆盖，不会出现可见抖动。

第 32 卷第 7 期 系统仿真学报 Vol. 32 No. 7
2020 年 7 月 Journal of System Simulation Jul., 2020



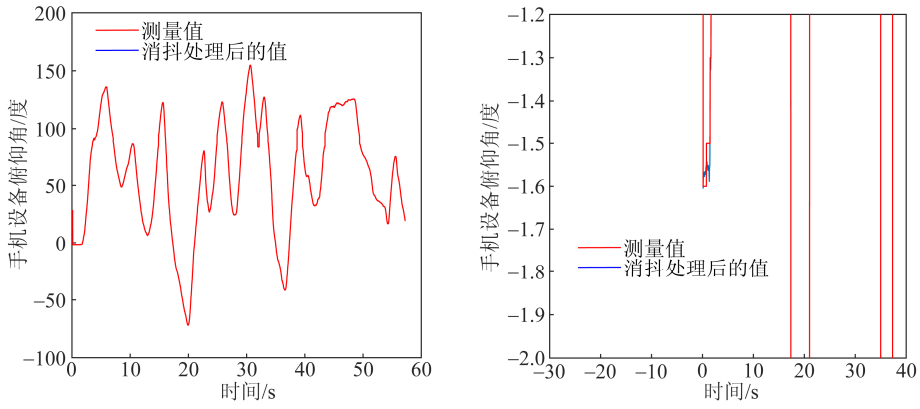
(a) 设备俯仰角曲线



(b) 测量值标准差

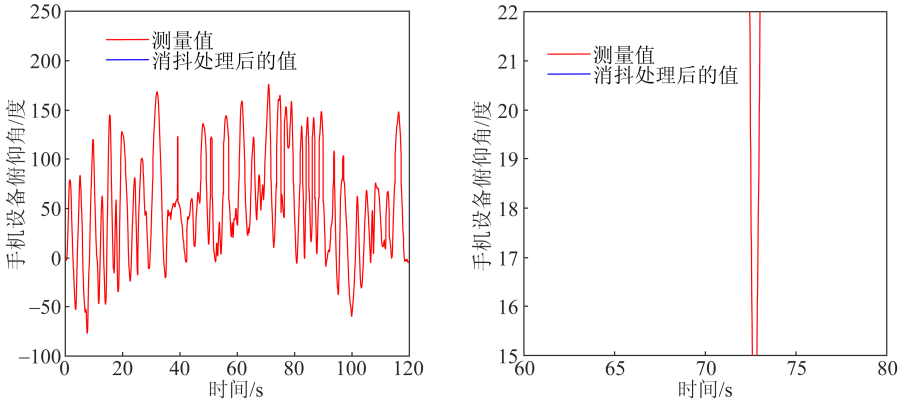
图 11 手机静止时设备俯仰角曲线图

Fig. 11 Change of pitch Angle when mobile phone is at rest



(a) 一分钟整体实验曲线

(b) 一分钟某随机区间实验曲线



(c) 两分钟整体实验曲线

(d) 两分钟某随机区间实验曲线

图 12 手机运动时设备俯仰角曲线图

Fig. 12 Change of pitch Angle when phone is in motion

4 结论

本文提出了一种面向手机的云端 VR 三维场景立体显示与交互框架。本框架结合光线跟踪渲染技术、VR 技术、视频块流传输、云渲染、移动端 Web 开发技术实现了 VR 三维场景在移动端 Web 上无插件立体显示与交互。既提供了较自然的交互方式，又能跨平台、低成本、高质量地在手机 Web 上立体显示 VR 三维场景；同时，本框架利用云渲染代替手机本地渲染，渲染效果上限高；且以手机 Web 作为客户端，开发成本低、开发效率高、易集成、易维护。为 VR 技术在手机 Web 中的应用提供了有效的技术途径。

然而，本框架在传输方式和手机 Web 显示与交互上也存在着诸多不足。在传输视频块流时，H264 视频压缩算法解压后图像会失真，并带有一定的显示延迟，而传输图像流的网络带宽占用太大，JPEG 压缩算法效率过低，因此框架在传输协议和压缩算法上还有改进空间；手机 Web 显示端由于手机系统和浏览器内核的局限性，仅能播放固定格式的视频和图像，播放效率低，需要借助未来手机系统和浏览器内核的更新提高框架性能；手机 VR 交互功能实现太少，仅实现了头部追踪交互，后续研究中将结合图像智能识别技术、移动端定位技术、智能声控技术和云计算等来丰富和完善手机 VR 交互功能，旨在为用户提供更逼真的 VR 体验。

随着云服务的发展和 5G 网络时代的到来，本框架将会有更广阔的发展前景，能够同时为 Web2D 领域和 Web3D 领域作出贡献。

参考文献:

[1] Suselbeck R, Schiele G, Becker C, et al. Peer-to-peer support for low-latency Massively Multiplayer Online Games in the cloud[C]. Paris: 2009 8th Annual Workshop on Network and Systems Support for Games (NetGames), 2009: 1-2.

[2] Ross P E. Cloud Computing's Killer App: Gaming[J]. IEEE Spectrum (S0018-9235), 2009, 46(3): 14.

[3] Ha J, Jung J, Han B, et al. Mobile Augmented Reality using scalable recognition and tracking[C]. Singapore: IEEE Virtual Reality Conference, 2011: 211-212.

[4] Huang B, Lin C H, Lee C, et al. Mobile augmented reality based on cloud computing[C]. Chinese Taipei: International Conference on Anti-Counterfeiting, Security and Identification (ASID), 2012: 1-5.

[5] Cao Y Z. Application Development of Virtual Reality Based on Smart Phone and VR Glasses[C]. Institute of Management Science and Industrial Engineering. Proceedings of 2018 5th International Conference on Electrical & Electronics Engineering and Computer Science (ICEEECS 2018), 2018: 497-502.

[6] 刘小军, 贾金原. 面向手机网页的大规模 WebBIM 场景轻量级实时漫游算法[J]. 中国科学: 信息科学, 2018, 48(3): 274-292.
Liu Xiaojun, Jia Jinyuan. Mobile web-based lightweight and real-time roaming algorithm for large-scale WebBIM scenes[J]. Scientia Sinica (Informationis), 2018, 48(3): 274-292.

[7] 刘镇, 刘晓, 梅向东. 面向移动终端的分布并行化渲染[J]. 中国图象图形学报, 2015, 20(9): 1247-1252.
Liu Zhen, Liu Xiao, Mei Xiangdong. Distributed parallel rendering method for mobile terminals[J]. Journal of Image and Graphics, 2015, 20(9): 1247-1252.

[8] Wang S, Dey S. Rendering Adaptation to Address Communication and Computation Constraints in Cloud Mobile Gaming[C]. Miami, Florida, USA: Global Communications Conference, 2010: 1-6.

[9] Cai W, Shea R, Huang C, et al. A Survey on Cloud Gaming: Future of Computer Games[J]. IEEE Access (S2169-3536), 2016, 4(99): 7605-7620.

[10] Hong H, Chen D, Huang C, et al. Placing Virtual Machines to Optimize Cloud Gaming Experience[J]. IEEE Transactions on Cloud Computing (S2168-7161), 2015, 3(1): 42-53.

[11] Schmalstieg D. The remote rendering pipeline-managing geometry and bandwidth in distributed virtual environments[D]. Vienna: Vienna University of Technology, 1997.

[12] Dey S. Cloud Mobile Media: Opportunities, challenges, and directions[C]. Maui, HI: 2012 International Conference on Computing, Networking and Communications (ICNC), 2012: 929-933.

[13] Wang S, Dey S. Adaptive Mobile Cloud Computing to Enable Rich Mobile Multimedia Applications[J]. IEEE Transactions on Multimedia (S1520-9210), 2013, 15(4): 870-883.

https://dc-china-simulation.researchcommons.org/journal/vol32/iss7/17
DOI: 10.16182/j.issn1004731x.joss.19-VR0477

第 32 卷第 7 期 系统仿真学报 Vol. 32 No. 7
2020 年 7 月 Journal of System Simulation Jul., 2020

[14] Ojala A, Tyrvainen P. Developing Cloud Business Models: A Case Study on Cloud Gaming[J]. IEEE Software (S0740-7459), 2011, 28(4): 42-47.

[15] Lampe U, Hans R, Steinmetz R. Will mobile cloud gaming work? findings on latency, energy, and cost[C]. Santa Clara, California, USA: Proceedings of the 2013 IEEE Second International Conference on Mobile Services, 2013: 960-961.

[16] Xun L, Woo W. The emerging Cloud-Mobile convergence paradigm for next-generation VR systems: Message from the CMCVR 2010 organizers[C]. Boston: In 2010 Cloud-Mobile Convergence for Virtual Reality Workshop (CMCVR 2010), 2010: 1-4.

[17] Satyanarayanan M. Pervasive computing: vision and challenges[J]. IEEE Personal Communications (S1070-9916), 2001, 8(4): 10-17

[18] Li Y, Su G, Hui P, et al. Multiple mobile data offloading through delay tolerant networks[C]. New York, NY, USA: Proceedings of the 6th ACM Workshop on Challenged Networks, 2011: 43-48.

[19] Flores H, Su X, Kostakos V, et al. Large-scale offloading in the Internet of Things[C]. Seattle, Washington, USA: 2011 IEEE International Conference on Pervasive Computing and Communications Workshops, 2017: 479-484.

[20] Sermpezis P, Spyropoulos T. Offloading on the edge: Performance and cost analysis of local data storage and offloading in HetNets[C]. Jackson Hole, Wyoming, USA: IEEE/IFIP Wireless On-demand Network systems and Services Conference, 2017: 49-56.

[21] Shi C, Habak K, Pandurangan P, et al. COSMOS: computation offloading as a service for mobile devices[C]. Philadelphia, Pennsylvania, USA: Proceedings of the 15th ACM international symposium on Mobile ad hoc networking and computing, 2014: 287-296.

[22] Kang Y, Kim H, Kang J. Docker based Computation Off-loading for Video Game based Mobile VR Framework[C]. The Institute of Electrical and Electronics Engineers、IEEE Beijing Section. Proceedings of 2017 IEEE 8th International Conference on Software Engineering and Service Science. The Institute of Electrical and Electronics Engineers, 2017: 143-145.

[23] Nguyen D, Le T, Lee S, et al. SHVC Tile-Based 360-Degree Video Streaming for Mobile VR: PC Offloading Over mmWave[J]. Sensors (S1424-8220), 2018, 18(11): 3728.

[24] 林定, 黄国新, 徐颖. 一种基于 WebVR 的网络数据三维树形可视化[J]. 系统仿真学报, 2018, 30(7): 2736-2743, 2752.
Lin Ding, Huang Guoxin, Xu Ying. A 3D Tree Visualization of Network Data Based on WebVR[J]. Journal of System Simulation, 2018, 30(7): 2736-2743, 2752.

[25] Chen C, Liang W, Yang C, et al. Generating stereoscopic videos of realistic 3D scenes with ray tracing[C]. Hefei, China: 5th Conference on Frontiers in Optical Imaging Technology and Applications, 2018(10832): 108320E.