Survey Report

吴家行
2020213991
清华大学软件学院

文献：Cutting the Cord: Designing a High-quality Untethered VR System with Low Latency Remote Rendering (Mobisys 2018)

1 内容概述

这篇文章介绍了一个适用于端到端的 VR 系统，在无线网络条件下，它可以满足高质量 VR 的延迟和视频质量的需求，因此作者称之是“无所束缚的”，“Cutting the Cord”这个标题也是比较生动形象的。

文中主要用到了两种方法来优化无线 VR 系统，即：

（1） 并行渲染和流处理

（2） 远程垂直同步渲染技术

结果显示该 VR 系统可以在 60Ghz 无线网络、2160x1200 的分辨率、90Hz 刷新率的条件下达到 16ms 以内的延迟，4K 分辨率可以达到 20ms 以内，同时保证给用户展示出无损的画质。

2 背景介绍

VR（Virtual Reality）是我们常说的虚拟现实，利用计算机可以产生出三维空间的虚拟世界，给用户带来“身临其境”的体验。

VR 的设备主要分为两种，一种是结合式的，另一种是独立式的。结合式 VR 需要将头戴设备（Head Mounted Display, HMD）和 PC 连接起来，头戴设备通过 USB 线将传感器数据传输给 PC，然后 PC 通过 HDMI 线将渲染的画面传回给头戴设备，这种方式可以是 VR 的画面质量更高清保真。独立式 VR 只是在单独一个设备上对画面进行操作，这种方式摆脱了“绳子的束缚”。

这两种设备都有各自的缺点，结合式 VR 有了“绳子的束缚”，用户行动不方便，甚至有安全隐患，独立式 VR 计算资源有限，没法做到高质量的画面渲染，有的人尝试用无线传输的方式将一些计算任务迁移到 PC 上，然而延迟会对画面帧率造成影响，使 VR 用户产生眩晕的感觉，严重影响用户体验。

对于无线的 VR 设备来说，端到端的延迟主要有以下几个组成部分：

$$T_{e2e} = T_{sense} + T_{render} + T_{stream} + T_{display}$$

每个部分的含义如下：

T_{e2e}：端到端的延迟时间，也就是从 HMD 开始发送数据，到画面显示出来的过程所需要的时间。

T_{sense}：感知时间，即从 HMD 开始发送数据到服务器接收到数据的时间。

T_{render}：渲染时间，即服务器根据传感数据生成一个新视频帧的时间。

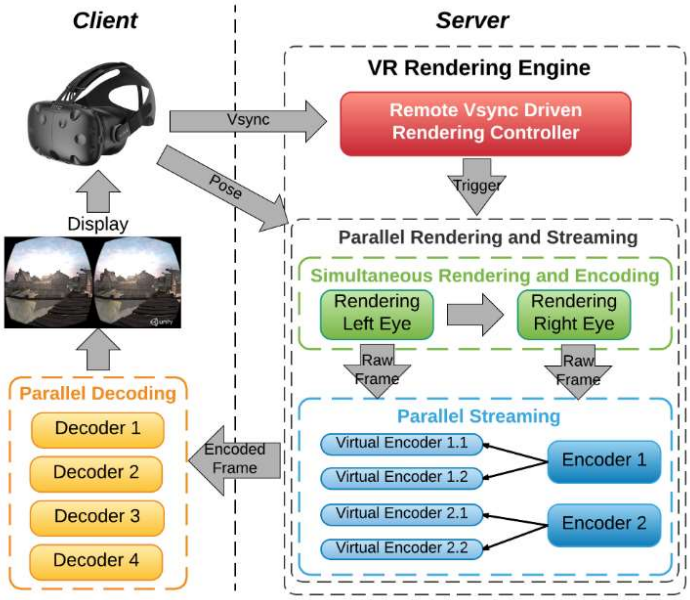
T_{stream}：流处理时间，即将新视频帧从服务器传输到 HMD 的时间，这包括对原视频的编码、传输和解码的过程。

$T_{display}$：显示时间，即 HMD 将新视频帧显示出来的时间，这一部分需要考虑进去是因为，在现在的图像系统中，一个视频帧的显示是通过 VSync 信号驱动的，VSync 信号是通过屏幕刷新率周期性生成的。如果一个帧错过了当前的 VSync 信号，那么它就需要在缓存队列中等待，直到下一次的 VSync 信号传来，才可以显示在屏幕上，对于 90Hz 的刷新率来说，平均等待时间是 5.5ms，这个等待时间会对整体的延迟造成一定影响。

本文主要通过优化T_{stream}和$T_{display}$来减小总体的端到端的延迟。

3 系统架构

本文的系统架构如下图所示。首先，VR 设备作为客户端向服务器发送垂直同步信号，以及用户的动作位姿信息，服务器端根据垂直同步信号时间决定渲染图像的具体时间，之后就是并行编码与传输环节，服务器同时进行渲染以及图像的编码，之后将压缩的视频帧传输到客户端，经过解码之后显示出来。



4 方法介绍

为了减少上文所说的T_{stream}延迟，文中提出**并行渲染和流处理**的方法，主要包含两部分：同时渲染与编码、并行流处理。针对$T_{display}$延迟，文中提出了**远程垂直同步渲染**的方法。

4.1 同时渲染编码

渲染一个高质量的视频帧需要比较长的时间，甚至在一个高性能的显卡上也需要超过 5ms 的时间，我们又不能降低图像质量，因此需要寻找其他方法去减少这一部分的耗时，提出了一种同时渲染和编码的方法。

可行性主要有两点：

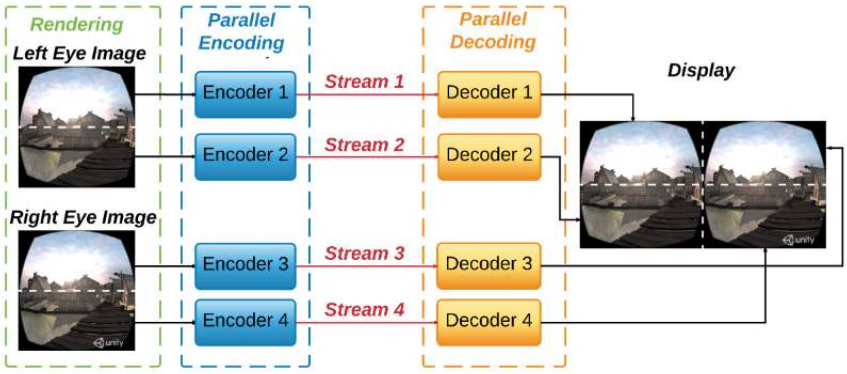
1、 VR 上的渲染通常分三步骤进行：（1）渲染左眼图像（2）渲染右眼图像（3）在整个视频帧加上镜头模糊，因此这种顺序渲染提供了一个机会，让我们可以在渲染右眼图像的同时开始对渲染好的左眼图像进行编码。

2、 现在的好多 GPU 有专用的硬件编解码部分，他们和渲染部分是相互独立的，因此即使利用 GPU 的编解码部分，也不会对渲染产生影响。

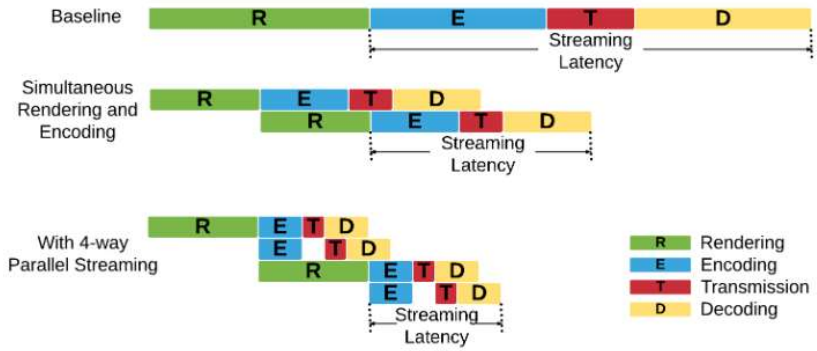
4.2 并行流处理

为了进一步减小视频流处理的时间，文中提出了一种多线程视频流处理的技术，利用多线程对每只眼睛的图像进行编码。

文中将每只眼的图像有分成两个 slide，分别对应图像的上半部分和下半部分，因此整个视频帧对应 4 个 slide，对应有四个编码线程和解码线程，结合前面提到的同时渲染与编码，过程如下图所示。



文中也解释了为什么这种分 slide 处理可以节约视频流处理的时间，相对于串行而言，如果将视频流分为 2 路，那么流处理的时间就会缩短一半；如果将视频流分为 4 路，那么流处理的时间就会变为四分之一，如下图所示。



然而，并不是线程数越多越好，GPU 上编解码部分并行的资源是有限的，而且分的 slide 太多会影响 H.264 编码的性能，对延迟造成影响，系统的实现也会变得更加复杂，因此本文仅将视频流分为 4 个。

4.3 远程垂直同步渲染

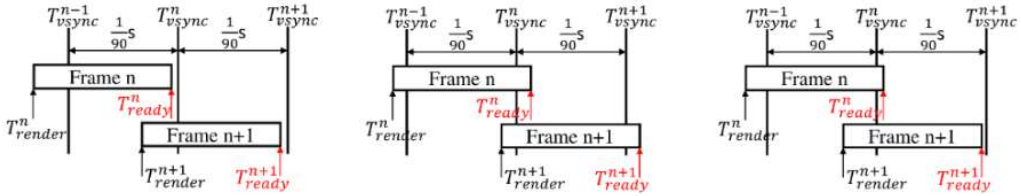
为了保证流畅的用户体验，避免画面撕裂现象，通常采用双缓冲的机制，分为前缓冲和

后缓冲。前缓冲用来存放在屏幕上显示的帧，后缓冲用来存放渲染的帧，一旦接收到 Vsync 信号，系统将会交换两个缓存，将新渲染的视频帧显示出来。

如果有一帧系统渲染的很快，那么这个帧必须在后缓冲中等待下一次的 VSync 信号到来才能显示出来；如果有一帧系统渲染的很慢，以至于错过了下一次的 VSync 信号，那么它必须再继续等待后面的 VSync 信号，才能显示出来。

上面说的 VSync 驱动的渲染机制适用于本地的视频渲染画面展示，但是对于远程渲染会出现新的问题，因为远程渲染中，显示是在 HMD 上进行的，渲染是在远程服务器上进行的。

那么，对于远程渲染来说，丢帧现象会变得明显。如下图所示，假设T^n_{render}代表开始渲染的时间，T^n_{ready}代表的是渲染结束并在 HMD 上解码完成等待显示的时间，$T^n_{ready}-T^n_{render}$代表一个视频帧的是生成时间，(a)展示的是理想情况，(b)展示的是第 n 帧错过了 VSync 信号的情况，这会增加端到端的延迟，(c)展示的是第 n 帧错过了 VSync 信号，第 n+1 帧和第 n 帧在等待同一个 VSync 的情况，这时就不得不丢掉第 n 帧，这意味着对第 n 帧的渲染、编码、传输、解码的操作全都白费了。



(a) Ideal case　　(b) Long waiting time　　(c) Missing a frame

当然，本地也会出现(b)和(c)这样的问题，但是并不会像远程那样出现的很频繁。

为了解决这个问题，作者利用 HMD 的 VSync 信号驱动远程服务器的渲染，核心思想就是，利用 HMD 传来的反馈，服务器决定渲染的时间，反馈信息包括当前帧的显示状态、等待时间以及 HMD 的移动。

第 n 帧渲染显示后，可以用下面的等式来表示第 n+1 帧的渲染时间：

$$T^{n+1}_{render}=T^n_{render}+\frac{1}{90}s+T_{shif}$$

在计算T^{n+1}_{render}和T^n_{render}的时间间隔时，除了考虑帧率 90Hz，还引入了T_{shift}这个动态的偏移量，表示如下：

$$T_{shift}=\left(T^n_{vsync}-T^n_{ready}-T_{conf}-T_{motion}\right)*cc$$

cc作为一个放缩因子，相当于一个低通滤波器，T_{shift}受多个因素影响，第一个因素是第 n 帧的等待时间$T^n_{vsyn}\ -T^n_{ready}$，如果将T^{n+1}_{render}向后推迟$T^n_{vsync}-T^n_{ready}$，假设每帧有相同的生成时间，那么$T^{n+1}_{ready}=T^{n+1}_{vsync}$，这样的话等待时间是最短的，但是一旦错过了 VSync 信号，那么很可能出现长等待或丢帧的现象，为了减少这种情况的发生，作者又引入了一个变量T_{conf}，使T^{n+1}_{ready}可以提前一些，避免错过 VSync 信号。

对于T_{conf}的设定，作者使用了一种统计的方法，首先将T_{conf}初始化为 0，然后每 1000 帧记录前 1000 帧的生成时间，计算出置信度为 99%的置信区间，将T_{conf}设置为置信区间的中点值。

第二个因素是 HMD 的移动，快速的移动会产生较大的画面变化，从而对应的渲染时间也比较长，因此需要进行提前渲染，以防错过 VSync 信号，HMD 的移动快慢主要受平移和旋转变化的影响，由于帧率很高，平移不会带来很大变化，而旋转角度的影响比较大，因此T_{motion}延迟部分将旋转角度考虑进去，计算公式如下：

$$T_{motion}=k*\Delta\theta^n$$

可以看出，这部分主要做的事情就是推迟下一帧的渲染时间，达到减小等待时间的目的，

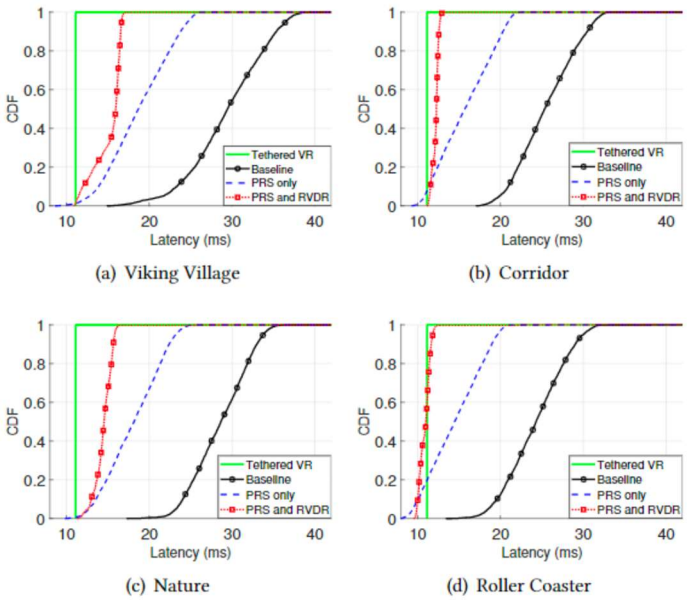
有人可能会觉得推迟渲染时间会不会让用户感受到不适，其实并不会，因为即使推迟了渲染时间，HMD 的位置信息用的是最新的，而 HMD 位置信息更新频率达到了 1000Hz，远比帧率大，因此这种方法是可行的。这样便尽可能的使垂直同步渲染处于一个理想的状态，既不产生较长的等待时间，也不丢帧，让视频帧的待显示时间一直处于下一次 VSync 信号到来之前，最小化用户的感知延迟，从而增强用户的实际体验。

5 实验结果

文中的实验采用了服务端和客户端的架构，远程服务器性能较高，配有 Titan 显卡，有线连接到一个 WiGig AP 上，HMD 连接一台普通的笔记本电脑作为客户端，无线连接到 AP 上。

作者选取 4 个不同场景的 VR 视频，提前录制了 VR 体验者的动作进行实验，将自己的方法分别和有线 VR、无线 VR（不包含任何优化）、无线 VR（只有 PRS 优化）进行对比。

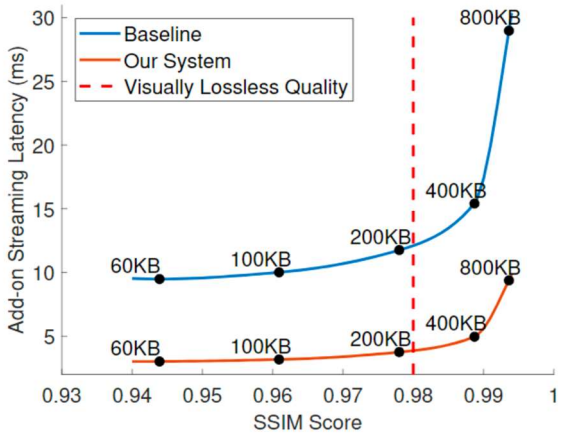
可以看出有的场景，本文的方法性能和有线 VR 一样出色，而有的场景不如有线 VR，这其实是渲染时间造成的。



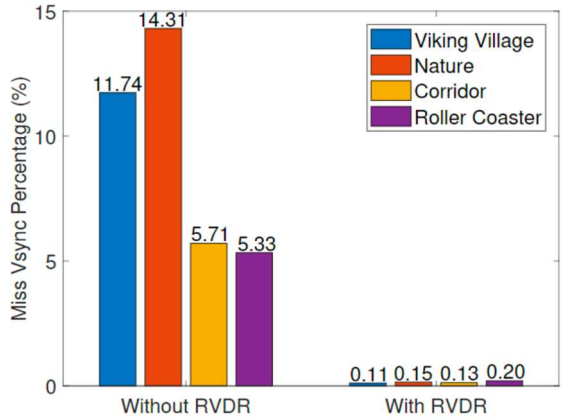
(a) Viking Village (b) Corridor

(c) Nature (d) Roller Coaster

作者使用 SSIM（Structural Similarity，结构相似度）作为评价视频质量的指标，SSIM 指标如果大于 0.98，就可以认为视频质量是无损压缩的。为了分析视频流延迟和视频质量的关系，作者做了一张不同的压缩质量对应视频流延迟的关系图，0.98 的 SSIM 对应的视频流延迟仅有 4ms 左右，相对于 baseline 已经很小了。



为了证明远程垂直同步渲染的有效性，文中使用 RVDR 技术和不使用 RVDR 技术的丢帧率对比，下图可见 RVDR 技术有效降低了丢帧的概率。



6 总结

本文利用了并行流处理和远程垂直同步渲染的方法实现了高质量、低延迟的无线 VR 系统，利用并行的方式，将原来流处理的时间进行成倍数的缩减，大大减少视频流处理的延迟；利用渲染过程中 VSync 信号的产生规律和显示规则，有效减少每个渲染帧的等待时间，降低了丢帧的概率。

这篇文章我认为他的亮点在于他很好的利用了硬件软件自身具备的一些性质，比如，目前一些 GPU 的编码模块是独立的，利用这个性质，可以充分利用 GPU 的资源，并行的处理视频流，大大减少了端到端的延迟。

另外我觉得这种并行流处理的方法可以用在很多的场景里，比如移动设备上的目标检测，AR 场景中的各种渲染等等，这类需要强大算力、却又在移动端部署的应用，都可以利用服务端强大的计算性能，实现一些仅靠移动端不能做到的功能。

Proposal

1 题目

基于并行流传输的移动端目标检测系统的研究

2 研究背景

随着计算机技术的高速发展，增强现实（AR）和虚拟现实（VR）等应用在日常生活中越来越受人们欢迎，AR 技术通过将计算机产生的虚拟图像叠加在实时捕捉的视频画面上，增强人们对现实世界的理解，由于移动设备的计算能力有限，将目标检测这种计算密集的任务部署在 AR 设备上是困难的，于是需要将计算密集的任务交给边缘服务器处理，这种边端协作模式成为 AR 场景下主流的目标检测方式。然而，目前已有的边端目标检测系统并不适合处理高分辨率且目标物体较多的视频，原因在于高清图像的文件较大，影响网络传输，而且对于多目标的跟踪准确度和实时性均较差。

在移动端计算资源有限的情况下，通过设计合理的边端协同策略，并行化视频流处理，减小端到端的延迟时间，从而提升移动端高清视频目标检测的性能，使用户体验大大提高。

3 研究目标

本课题主要研究一下几个问题：

1、 不同编码方式的压缩速度以及压缩率
2、 不同压缩率对应的视频质量
3、 通过并行视频流处理的方式优化移动端目标检测性能
4、 分析优化后不同分辨率下目标检测的延迟和精度

4 研究计划与时间安排

10.6-10.13	测量不同编码方式的压缩效率，测量不同压缩率对应的视频质量
10.13-10.20	基于并行流处理的目标检测系统的实现
10.20-10.27	测量不同分辨率下系统的检测精度和延迟
10.27-11.3	总结实验结果，形成报告

6 参考文献

[1] Chen, Tiffany Yu-Han, et al. "Glimpse: Continuous, real-time object recognition on mobile devices." Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems. 2015.
[2] Liu, Luyang, et al. "Cutting the cord: Designing a high-quality untethered vr system with low latency remote rendering." Proceedings of the 16th Annual International Conference on Mobile

Systems, Applications, and Services. 2018.
[3] Liu, Luyang, Hongyu Li, and Marco Gruteser. "Edge assisted real-time object detection for mobile augmented reality." The 25th Annual International Conference on Mobile Computing and Networking. 2019.