Improving The Bandwidth Usage For VR Video Transmission

2018.01
Regular Video Streaming Method

Sphere 2D Mapping → Encoder → Single Video Stream → Decoder → Sphere 2D Mapping

VR Video Pre-processing → Traditional Video Streaming Infrastructure → VR Video Post-processing
Example-1: Intel TrueVR

- 360 degree panoramic video
- Average Bandwidth at ~12Mbps

Real network traffic captured from playing back 2018 Pyeong Chang Winter Olympic Opening Ceremony with Intel TrueVR player
Example-2: NextVR

- Video is 180 degree Stereo
- Average Bandwidth at ~10Mbps

Real network traffic captured from playing back NextVR NBA sports scene with NextVR App
Issues With Regular Video Streaming

1. Video content is treated equally in all viewing direction
2. Video content outside current Field of View (FOV) causes bandwidth waste
3. Not easy to support higher resolution (6K, 8K)
View Adaptive Video Streaming Method
Adaptive Video Streaming (1)

View Orientation

Original VR Content

Multiple Streams
One for each view port

Single Stream to be streamed each time

View Adaptation
Initiated by Viewer
Example-3: View Port Method

- Video is 360 degree Mono
- Without head movement
  - Average Bandwidth at 8Mbps
  - Peak Bandwidth at ~12Mbps
- With head movement
  - Average Bandwidth at 40Mbps!
  - Peak Bandwidth at ~80Mbps!

Real traffic captured from playing back a simulated view port encoded stream @ 8Mbps
Adaptive Video Streaming (2)

Tile Based Method: (Visbits, TiledMedia)

* Content copied from Bitmovin Website
Example-4: Visbits DemoApp

- Video is 360 degree Mono
- **Without head movement**
  - Average Bandwidth at 6Mbps
  - Peak Bandwidth at ~9Mbps
- **With head movement**
  - Average Bandwidth at 18Mbps!
  - Peak Bandwidth at ~30Mbps!

Real traffic captured from playing back demo sequence on Visbits DemoApp
Example-5: TiledMedia ClearVR DemoApp

- Video is 360 degree Mono
- Without head movement
  - Average Bandwidth at 12Mbps
  - Peak Bandwidth at ~20Mbps
- With head movement
  - Average Bandwidth at 20Mbps!
  - Peak Bandwidth at ~40Mbps!

Real traffic captured from playing back demo sequence on TiledMedia DemoApp
What happens when head moves?

• Head movement triggers view change
  - View change causes the pre-buffered content to be invalid
  - Empty decoder buffer triggers re-buffering!
  - Improper re-buffering strategy could cause:
    - Extreme volatility in network traffic
    - Significant bandwidth waste
    - Extended view switch latency
Re-Buffering Behavior During View Changes

When view is changed:
- The video inside decoder buffer view#1 is no longer useful
- The decoder buffer for the new view#2 is empty so decoder can’t start decoding
- The player must request the server to fill up the decoder buffer for view#2 ASAP
- This causes the spike in network traffic as well as waste of the pre-buffered content in view#1
Our Solution: Foveated Encoding (FE)

• We invented a multi-layer based view adaptation scheme based on foveat vision
• In our scheme, the VR video are encoded in multiple layers
  • At least one view independent layer, and several view dependent layers
  • View independent layers are encoded with large chunk
  • View dependent layers are encoded with small chunk
• When view changes
  • For view independent layer, no need to be rebuffered
  • For view dependent layers, only small chunk of video data need to be rebuffered
Major Benefits:

• More Efficient Compression
  • View independent layers are encoded in large chunks for better compression efficiency

• Faster View Switch
  • View dependent layers are encoded in small chunk to achieve faster view switch

• Less Bandwidth Waste
  • View independent layers have larger buffers without any waste
  • View dependent layers encoded in small chunk reduce bandwidth waste during view switch

• Smoother Network Traffic
  • Rebuffering is only needed for view dependent layers (with much smaller chunks)
Our Re-Buffering Strategy
Improved Network Bandwidth Usage

Regardless if there is head movement or not

- The bandwidth stays fairly smooth around average bandwidth
- With peak bandwidth < 1.5 average bandwidth
运行性能
——
推流端手机的CPU、GPU的使用率平均增加10%左右

<table>
<thead>
<tr>
<th>时间</th>
<th>CPU使用率</th>
<th>GPU使用率</th>
</tr>
</thead>
<tbody>
<tr>
<td>0:00</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>0:02</td>
<td>5%</td>
<td>5%</td>
</tr>
<tr>
<td>0:04</td>
<td>10%</td>
<td>10%</td>
</tr>
<tr>
<td>0:06</td>
<td>15%</td>
<td>15%</td>
</tr>
<tr>
<td>0:08</td>
<td>20%</td>
<td>20%</td>
</tr>
<tr>
<td>0:10</td>
<td>25%</td>
<td>25%</td>
</tr>
<tr>
<td>0:12</td>
<td>30%</td>
<td>30%</td>
</tr>
</tbody>
</table>

以某直播平台视频参数为准测试

深圳威尔视觉传媒有限公司