Video Streaming for Foveated High-resolution Rendering

Master's thesis, Marc Aurel Kastner
Supervised by M. Stengel, Prof. M. Magnor
Bandwidth limits

• Panoramic scenes create enormous FOV
• Video data continuously increases
  - Resolution: 2K → 4K → 8K → ...
  - Frame rate: 25 fps → 60 fps → ...
• Unable to process full high resolution frames

→ Need for high resolution with low bandwidth
Background: Visual acuity

• Simplified model:
  ▪ Aubert / Foerster (1857)
  ▪ Linear fall-off until 20°
  ▪ Then, strong drop

• Still commonly used
  ▪ Conservative
  ▪ Simplicity
Related Work

Foveated MPEG (Geisler et al. 1996)
• Static / Passive
  ▪ Saliency estimation
  ▪ Molding acuity into data
• e.g. sport & news clips
• Prone to error
  ▪ Users might look in different directions
  ▪ Not well optimized to one users’ view

Foveated 3D Graphics (Gunther et al. 2012)
• Dynamic
  ▪ Pupil Tracking
  ▪ No data modifications
• e.g. rasterization, raytracing
• Adapts to users’ eye
  → Make this for videos
Pupil tracking in HMDs

• Recent VR often use Head mounted Displays

• HMDs allow pupil tracking
  ▪ Active gaze estimation
  ▪ Knowledge about users’ field of view
Idea

• Decrease bandwidth
  ▪ Use gaze estimation
  ▪ Filter resolutions simulating acuity
  ▪ Optimize streaming

• Restraints
  ▪ Dynamic
  ▪ Performance
  ▪ Perception

Table of contents

• Motivation / Idea
• Related work
• Design
• First approach
• Second approach
• Evaluation
Design

• Three main pillars
  ▪ Pre-processing
  ▪ Video loading
  ▪ Texture streaming
Pre-processing

• Video formats don't allow region or pixel access

• Impractical to decode full frames
  ▪ Access smaller videos
  ▪ Have multiple quality versions

• No relation to eye position
  ▪ Dynamic
Background: Uniform grid

- Uniformly distributed
- Spatial sub-division
- Used for
  - Ray tracing
  - Video tiling
Background: Mip mapping

• Idea
  ▪ Pre-process scaled down variants of textures
  ▪ Save different resolution levels
  ▪ Reduce stress on GPU, simplify filtering, avoid moiré patterns, …

• Often used in video games
Pre-processing

- Idea
  - Use video tiling
  - Create multiple mip-map levels per video tile
- Allow region-based loading
First approach

- Wide-screen videos
  - Arbitrary codec

- Acuity distribution: Hybrid!
  - Grid-based loading
  - Sampling-based streaming
Grid-based video loading

- Depending on eye position
  - Select suitable resolution
  - Threshold mip-map levels with distance

- Grid 4x4, 3x Mip-map
  - 48 video files
  - Load frames as needed
Multi-threaded video loading

• Seeking is major bottleneck
  ▪ Every video file is open in parallel
  ▪ An eye movement triggers wall of seeking

• Solution
  ▪ Multi-threading model
  ▪ Sometimes use wrong mip-map levels
    • Didyk et al. (2010): Retina takes 60 ms to adapt
  ▪ Allow system to keep up
Sampling-based acuity

- Two masks
  1. Acuity mask
  2. Interpolation mask

- Allows minimum bandwidth
- Linear reconstruction with look up tables
Frame reconstruction

<table>
<thead>
<tr>
<th>ID</th>
<th>X</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>15</td>
<td>12</td>
</tr>
<tr>
<td>1</td>
<td>46</td>
<td>56</td>
</tr>
<tr>
<td>2</td>
<td>89</td>
<td>96</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Streamed to GPU every frame

Shading

0 0 1 1
0 2 3 1
4 5 6 7
4 4 7 7

0 0 1 1
0 2 3 1
4 5 6 7
4 4 7 7
Second approach

- 360 degree panorama
  - Pictures per frame
  - JPEG/PNG

- Sampling may introduce visible flickering

- Acuity distribution
  - Fully grid-based
  - Radial blur in post-process reduces peripheral frequencies
Background: Cube mapping

- Six single textures for 360° surface
- GPU: One cube map per mip-map
  - One Grid per cube side

Source: David J. Eck
Source: Emil Persson
For every frame:
- Calculate view vector
- Decide mip-map
Update only relevant region per frame
Acuity data as look up table
Evaluation: Demo video 1
Evaluation: Demo video 2
Evaluation Codecs

Foveated Video
Approach 1

Loading performance for different codecs

<table>
<thead>
<tr>
<th>Image format (8K video)</th>
<th>VP 8</th>
<th>VP 9</th>
<th>H.264</th>
<th>H.265</th>
</tr>
</thead>
<tbody>
<tr>
<td>frames per second</td>
<td>60</td>
<td>62</td>
<td>40</td>
<td>45</td>
</tr>
</tbody>
</table>

Foveated Video
Approach 2

Loading performance for different image formats

<table>
<thead>
<tr>
<th>Codecs (8K video)</th>
<th>PNG</th>
<th>JPEG q=90</th>
<th>JPEG q=100</th>
</tr>
</thead>
<tbody>
<tr>
<td>frames per second</td>
<td>15</td>
<td>45</td>
<td>30</td>
</tr>
</tbody>
</table>
Evaluation Performance

Foveated Video
Approach 1

Performance with different resolution settings

<table>
<thead>
<tr>
<th>Resolution Settings</th>
<th>Scene (8K)</th>
<th>Scene (16K)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Simulated acuity</td>
<td>60</td>
<td>70</td>
</tr>
<tr>
<td>Minimum</td>
<td>40</td>
<td>60</td>
</tr>
</tbody>
</table>

Foveated Video
Approach 2

Performance with different resolution settings

<table>
<thead>
<tr>
<th>Resolution Settings</th>
<th>Scene 1: Inside the Gl tract (16K)</th>
<th>Scene 2: Weihnachtsmarkt (12K)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Simulated acuity</td>
<td>30</td>
<td>20</td>
</tr>
<tr>
<td>Minimum</td>
<td>10</td>
<td>20</td>
</tr>
</tbody>
</table>
## Evaluation Bandwidth

<table>
<thead>
<tr>
<th>Wide-screen videos</th>
<th>Transferred pixels</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sampling-based acuity (1080p acuity mask)</td>
<td>690.926</td>
<td>2.08%</td>
</tr>
<tr>
<td>Minimum resolution (smallest mip-map 240p)</td>
<td>129.600</td>
<td>0.39%</td>
</tr>
<tr>
<td>Full resolution (8K wide-screen)</td>
<td>3.177.600</td>
<td>100.00%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Panoramic videos</th>
<th>Transferred pixels</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grid-based acuity (best case)</td>
<td>1.773.229</td>
<td>2.89%</td>
</tr>
<tr>
<td>Grid-based acuity (worst case)</td>
<td>2.789.232</td>
<td>7.74%</td>
</tr>
<tr>
<td>Minimum resolution (smallest mip-map 120p)</td>
<td>212.064</td>
<td>0.59%</td>
</tr>
<tr>
<td>Full resolution (12K panoramic)</td>
<td>36.000.000</td>
<td>100.00%</td>
</tr>
</tbody>
</table>
Conclusion

- Exploit human visual system
- Replace bandwidth with storage
  - Pre-processing is cheaper than bandwidth
- No alternative: Common video codecs very slow >4K
- Potential for various future apps (VR, mobile, internet)
Thank you for your attention.
Appendix
• Slide 5

• Slide 16

• Full bibliography in thesis p. 69-76
Sources / Licenses

• Slide 2
  - Grzegorz Rogala, Creative Commons
    https://www.flickr.com/photos/grzegorz_rogala/5097827282/

• Slide 3
  - JeffPerry, Public domain
    https://en.wikipedia.org/wiki/Foveated_imaging

• Slide 12
  - Mulad, Creative Commons
    https://en.wikipedia.org/wiki/Mipmap

• Slide 20
  - Emil Persson, Creative Commons
  - David J. Eck, Creative Commons
    http://math.hws.edu/graphicsbook/c5/s3.html

• Evaluated scenes
  - Appendix Slide 34

• Everything else created by myself or ICG
Evaluation

• Hardware
  ▪ MacBook Pro
  ▪ 2.5 GHz Intel Core i7 Haswell CPU
  ▪ NVIDIA GeForce GT 750M 2048 MB
  ▪ 16 GB DDR3 RAM
  ▪ SSD

• Software
  ▪ SDL 2.0
  ▪ OpenCV 3.1
  ▪ FFmpeg 2.8.6
  ▪ STB_Image 2.8
  ▪ PIL 3.0
  ▪ Mac OS X 10.11.3
  ▪ C++ '14 (Main)
  ▪ Python 3.5.1 (Pre-Proc)
Evaluation Scenes

• Scene 1: Caminandes
  ▪ Wide Screen
  ▪ Original 4K, upscaled 8K/16K
  ▪ Blender Foundation (Creative Com.)
  ▪ http://www.caminandes.com

• Scene 2: Inside the GI tract
  ▪ Panoramic
  ▪ Original 4K, upscaled 16K
  ▪ HybridMedical (Creative Commons)
  ▪ https://www.youtube.com/watch?v=upSnH436ya8

• Scene 3: Christmas market
  ▪ Panoramic
  ▪ Original 12K
  ▪ Captured with GoPro (Matthias)
Evaluation

• Grid
  - 8x8 for widescreen 16K
  - 4x4 everything else

• Mip-map levels: 3
  - Full resolution
  - Mid resolution = 1 / 4 times full
  - Low resolution = 1 / 16 times full
### Evaluation Video RAM

<table>
<thead>
<tr>
<th>Approach</th>
<th>Data</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acuity mask</td>
<td>2.76 mb</td>
<td>0.69</td>
</tr>
<tr>
<td>Interpolation mask</td>
<td>44.00 mb</td>
<td>11.06</td>
</tr>
<tr>
<td>Total used by approach</td>
<td>46.76 mb</td>
<td>11.75</td>
</tr>
<tr>
<td>Total used by classic video</td>
<td>398.00 mb</td>
<td>100.00</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Approach</th>
<th>Data</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cube map (high resolution)</td>
<td>1152.0 mb</td>
<td>0.69</td>
</tr>
<tr>
<td>Cube map (mid resolution)</td>
<td>72.0 mb</td>
<td>11.06</td>
</tr>
<tr>
<td>Cube map (low resolution)</td>
<td>4.5 mb</td>
<td>11.75</td>
</tr>
<tr>
<td>Total used by approach</td>
<td>1228.5 mb</td>
<td>106.64</td>
</tr>
<tr>
<td>Total used by classic video</td>
<td>1152.0 mb</td>
<td>100.00</td>
</tr>
</tbody>
</table>
## Evaluation Pre-processing

<table>
<thead>
<tr>
<th>Approach</th>
<th>Time spent</th>
<th>Storage used</th>
</tr>
</thead>
<tbody>
<tr>
<td>Framework 1: VP 8</td>
<td>3 h 58 min</td>
<td>3237 mb</td>
</tr>
<tr>
<td>Framework 1: VP 9</td>
<td>8 h 17 min</td>
<td>2294 mb</td>
</tr>
<tr>
<td>Framework 1: h.284</td>
<td>0 h 36 min</td>
<td>1923 mb</td>
</tr>
<tr>
<td>Framework 1: h.285</td>
<td>1 h 30 min</td>
<td>104 mb</td>
</tr>
<tr>
<td>Framework 2: PNG</td>
<td>50 min</td>
<td>1608 mb</td>
</tr>
<tr>
<td>Framework 2: JPEG q=90</td>
<td>17 min</td>
<td>816 mb</td>
</tr>
<tr>
<td>Framework 2: JPEG q=100</td>
<td>21 min</td>
<td>2500 mb</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Approach</th>
<th>Mip-map 0</th>
<th>Mip-map 1</th>
<th>Mip-map 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Framework 1 (Scene 1)</td>
<td>1885 MB</td>
<td>365 mb</td>
<td>46 mb</td>
</tr>
<tr>
<td>Framework 2 (Scene 2)</td>
<td>721 MB</td>
<td>81 mb</td>
<td>14 mb</td>
</tr>
<tr>
<td>Framework 2 (Scene 3)</td>
<td>3520 MB</td>
<td>463 mb</td>
<td>92 mb</td>
</tr>
</tbody>
</table>
Perception optimization

• Saliency-based approaches
• Post-processing
  ▪ Radial blur filter / Gauss blur
• Data
  ▪ Bigger Grid size
  ▪ Higher mip-map levels
  ▪ Less distance between mip-map levels
Flowchart Multithread Model

1. Acuity mask
   - Eye vector
   - Acuity sampling determination
     - selecting mip-map level for each tile as necessary for sampling
2. Video Handling (for frame n)
   - if this video is behind
   - Video tiles
     - Seek to frame n−1
       - 1 thread per tile
       - inform about tile
3. Acquire next frame
   - if video is at frame n−1
     - While waiting, use low mip-maps
     - Send sampled data to GPU if full frame n was sampled
4. Sampling
   - Interpolation mask
Acuity calculations

```python
1 @side, axis calculateCubeSide(avec3 eye):
2     switch eye:
3         eye.x > 0.5: return +X, X
4         eye.x <= 0.5: return -X, X
5         eye.y > 0.5: return +Y, Y
6         eye.y <= 0.5: return -Y, Y
7         eye.z > 0.5: return +Z, Z
8         eye.z <= 0.5: return -Z, Z

1 vec2 calculateGridPosition(avec3):
2     side, axis = calculateCubeSide(eye)
3     otheraxis1 = (getAxisUnequal(eye, axis).a + 0.75) * 2
4     otheraxis2 = (getAxisUnequal(eye, axis).b + 0.75) * 2
5     return (otheraxis1 / 4.0, otheraxis2 / 4.0)
```
void selectMipmap(vec3 eye):
  setquality(ALL, lowest)
  // Grid-based acuity
  vec2 fovealCell = calculateGridPosition(eye)
  setquality(fovealCell, maximum)
  foreach(neighborCell : adjacent to fovealCell):
    setquality(neighborCell, halfway)
  // Variance-based acuity
  foreach(noiseCell : cell near maximum(color variance)):
    setquality(noiseCell, halfway)
Saliency-based approach

• Idea
  ▪ Add analysis to pre-processing
    • Color variance (RGB/HSV/LAB)
    • Hard edges (FFT)
    • Temporal differences
  ▪ Set mip-map level not only based on visual acuity but also above analysis
  ▪ For example:
    • use mid resolution instead of low resolution if salient feature in tile
HMD System

• Current implementation: "Debug"
  ▪ Eye position = Mouse pointer
  ▪ No stereoscopic rendering

• Port to HMD: No obstacles
  ▪ Code is standard C++ '14
  ▪ All cross-platform libraries
  ▪ Stereoscopic rendering presumably no major influence on performance
Limitations

• Hardware
  - GPU least busy component
  - Due to parallel video access SSD necessary

• Pre-processing necessary
  - Mostly time consuming
  - Overhead in storage

• Grid
Future work

• Perceptual user study to verify results
• Bandwidth allows new applications
  ▪ Mobile devices
  ▪ Remote streaming
    • Internet/Cloud?

• Approach
  ▪ Other video codecs
  ▪ Container format, storage optimizations
  ▪ Add saliency-based approaches (variances, edges)