CDN OPTIMIZATIONS FOR VR STREAMING

Introducing a VR streaming latency model

Ray van Brandenburg, Tiledmedia
David Sztykman, Akamai
Rob Koenen, Tiledmedia

About Tiledmedia

Founded in October 2016

Spin-off from TNO, based on research since 2011

Focused exclusively on streaming of VR and panoramic video

The problem with streaming VR360

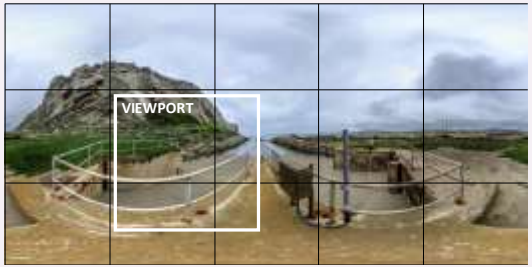
- Today's HMDs are still limited in resolution per eye
- Yet, to make optimal use of it, you still need 6K x 3K video (>30 Mbit)
- And headsets with 4K and 8K *per eye* are already being proposed

- Bandwidth is clearly an issue

Viewport-dependent streaming techniques exist to reduce the bandwidth problem: only stream what you see

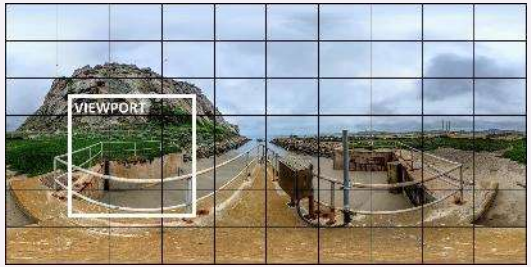
Create multiple representations of full sphere, each with focus area in high-res; rest in low-res

Facebook, others



HEVC motion-constrained tiles with server-controlled sub-streams

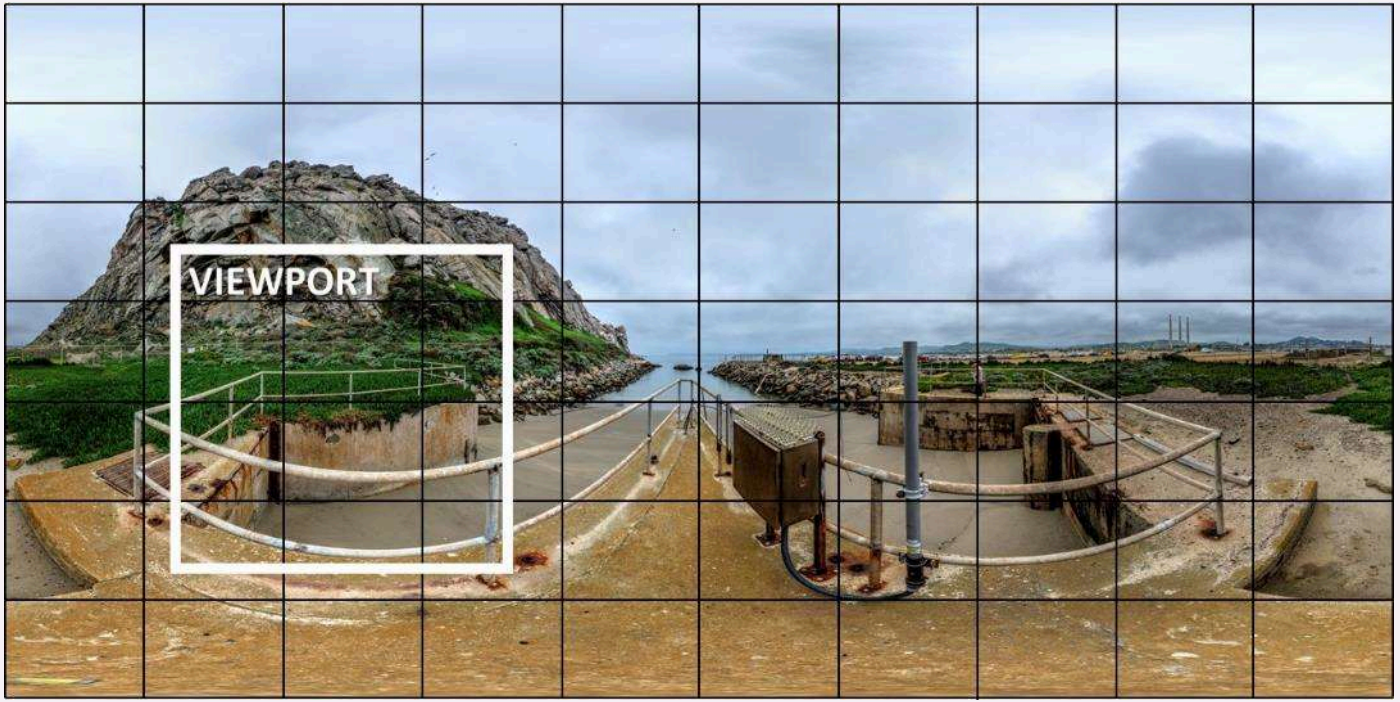
MPEG OMAF



HEVC tiles with client-controlled tile selection and recombination

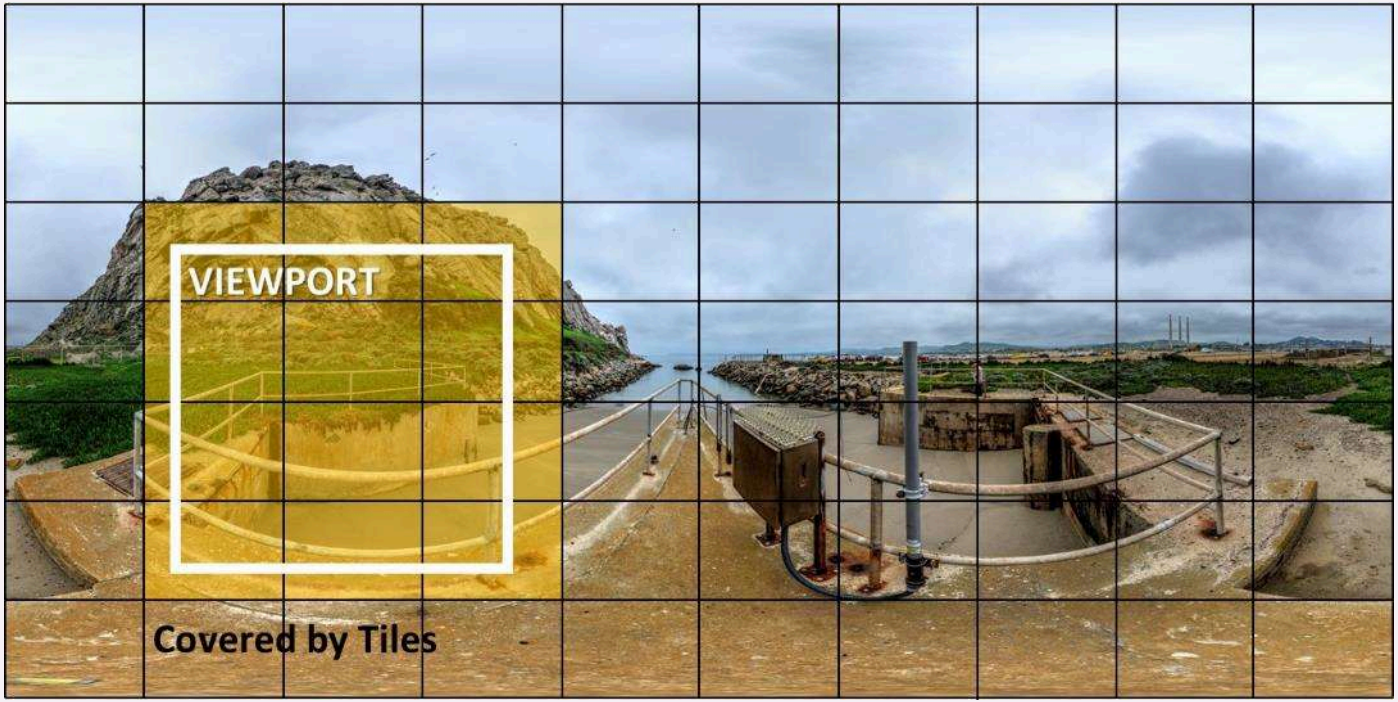
Tiledmedia ClearVR

Introduction to client-controlled tile-based streaming



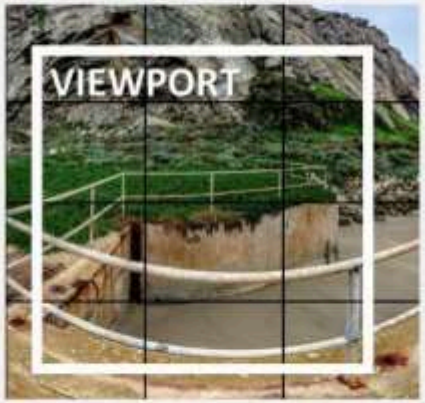
At encoding, split video up into multiple independently decodable ‘tiles’

Introduction to client-controlled tile-based streaming



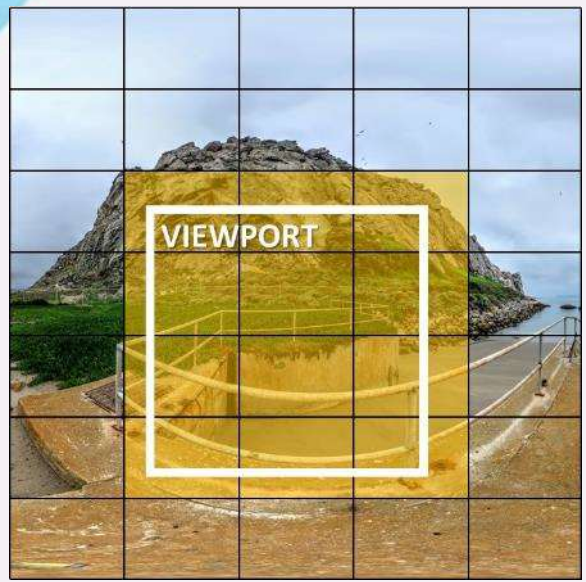
Fetch only tiles needed to reconstruct the viewport from network

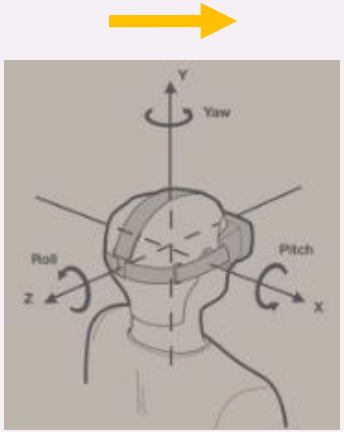
Introduction to client-controlled tile-based streaming

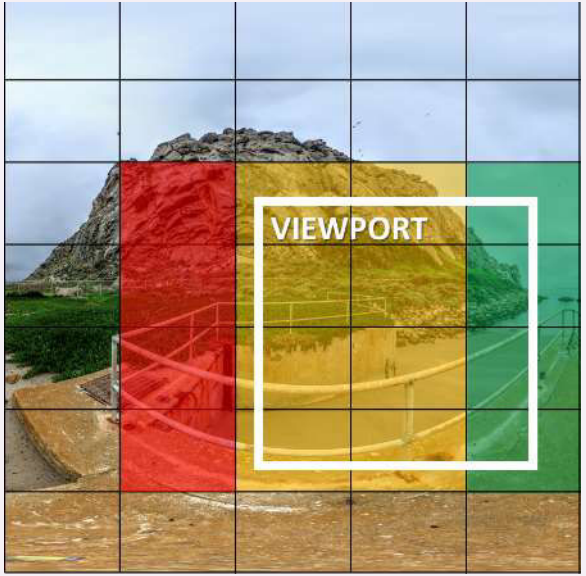


Allowing for a maximum bandwidth reduction of 85%

Introduction to client-controlled tile-based streaming







Tiles in viewport

Cancelled tiles

Newly requested tiles

During head motion, only small number of files need to be changed,
allowing for most of the buffer to be re-used

Viewport-dependent streaming introduces a new problem: latency

Motion-to-*photon* latency

The time it takes for the viewport to respond to head-movement

Motion-to-*high-resolution* latency

The time it takes for the *full* viewport to be available in *high quality* after head movement

To assist in the design, development and testing of viewport-dependent solutions, we propose a model that models the various elements impacting these latency characteristics

Introducing: VR streaming latency model

Trigger

Cache-miss delay
0.5*RTT (ca. 10-200ms)
Function of: CDN

Network request delay
0.5*RTT (5-100ms)
Function of: Network

Sensor delay
1-10ms

GET

GET

Sensor data

Origin

CDN Edge

3G/4G

Wifi/Lan

Client
(smartphone, PC)

Decoder

Renderer

VR
HMD

<data>

<data>

<bitstream>

<frames>

<textures>

Cache population delay
0.5*RTT + transmission delay (ca. 10-400ms)

Function of:
CDN, request size

Network response delay
0.5*RTT + transmission delay (ca. 10-1000ms)

Function of:
Network, request size, segment size

Buffering delay

Function of:
Buffer size, Packaging format, streaming protocol

Decoding delay
(6-500ms)

Function of:
b-frames, position in GOP, content FPS, HW/SW decoder

Rendering delay
(0-80ms)

Function of:
Single/Dual framebuffer (platform)
Refresh rate

Latency for traditional brute-force VR streaming

Trigger

Sensor delay
1-10ms

Sensor data

Cache-miss delay
0.5*RTT (ca. 10-200ms)
Function of: CDN

GET

Network request delay
0.5*RTT (5-100ms)
Function of: Network

GET

Origin

CDN Edge

3G/4G

Wifi/Lan

Client
(smartphone, PC)

Decoder

Renderer

VR
HMD

<data>

<data>

<bitstream>

<frames>

<textures>

Cache population delay
0.5*RTT + transmission delay (ca. 10-400ms)

Function of:
CDN, request size

Network response delay
0.5*RTT + transmission delay (ca. 10-1000ms)

Function of:
Network, request size, segment size

Buffering delay

Function of:
Buffer size,
Packaging format,
streaming protocol

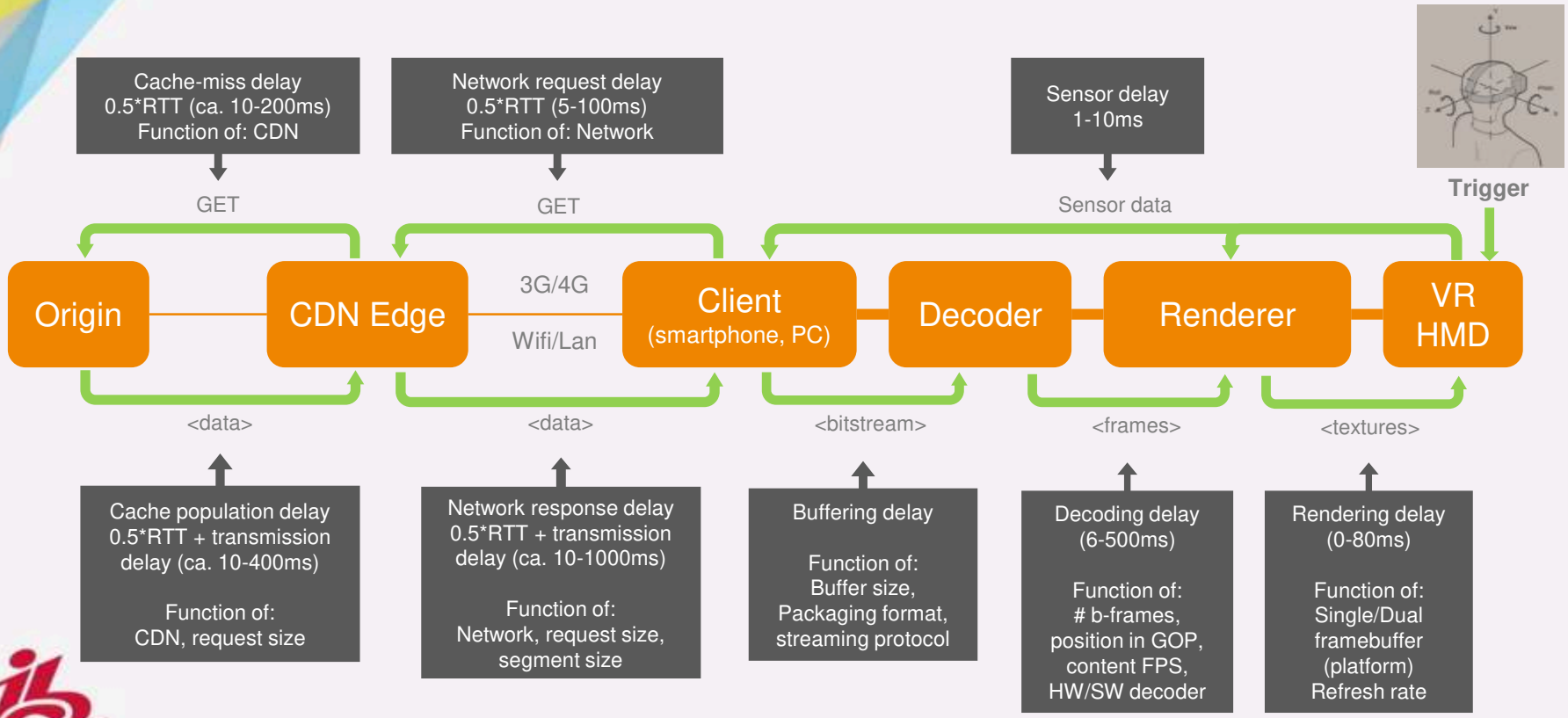
Decoding delay
(6-500ms)

Function of:
b-frames,
position in GOP,
content FPS,
HW/SW decoder

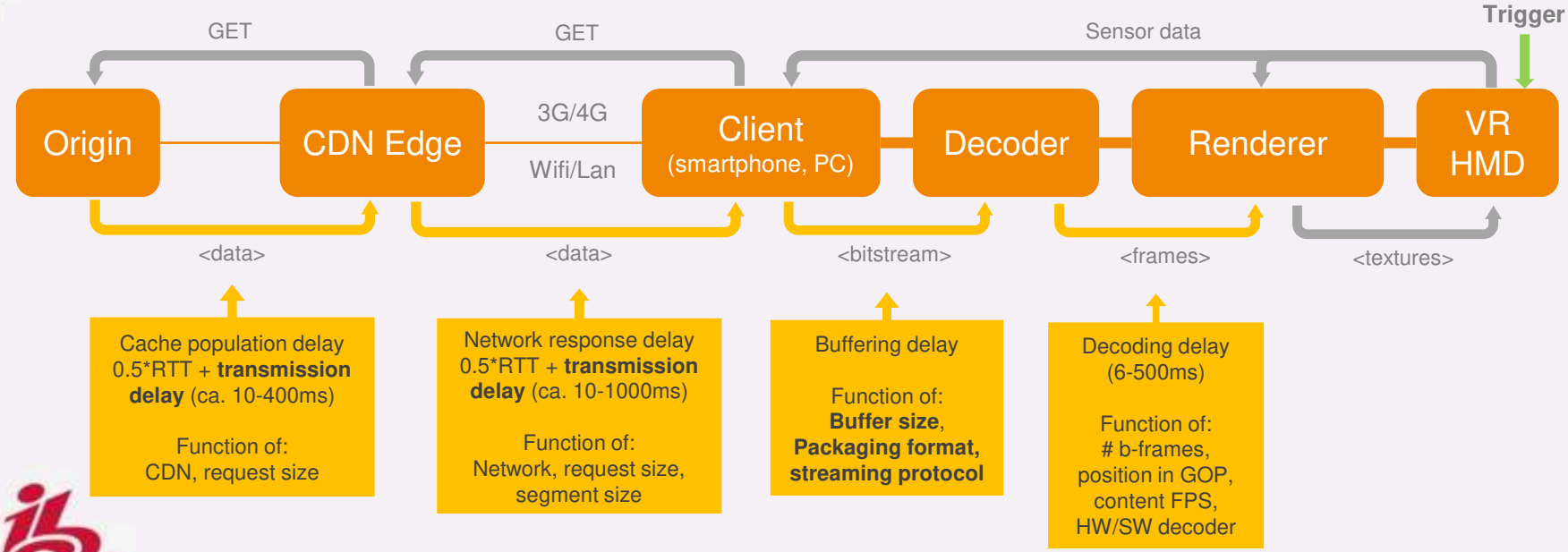
Rendering delay
(0-80ms)

Function of:
Single/Dual framebuffer
(platform)
Refresh rate

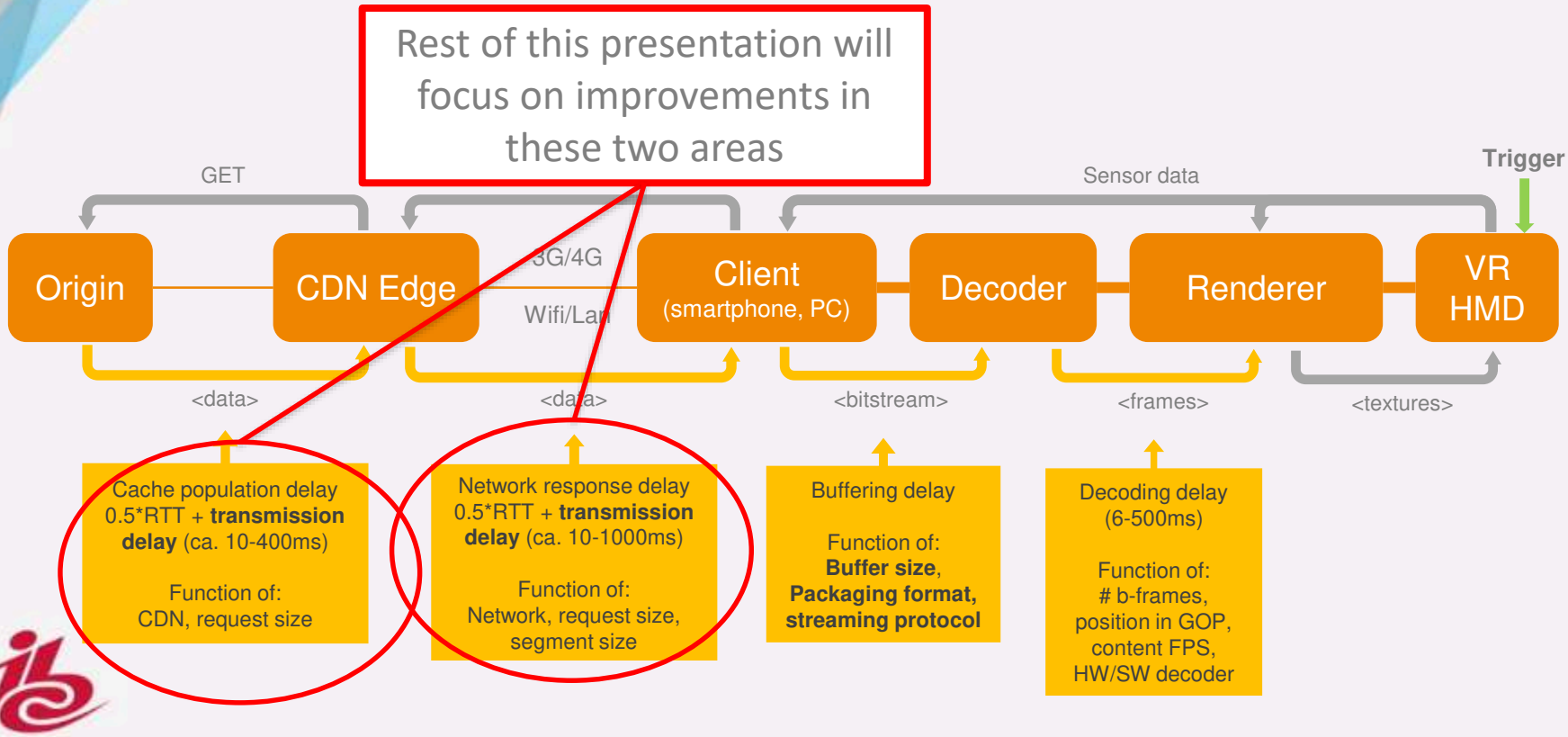
Applying latency model to client-controlled tile-based streaming



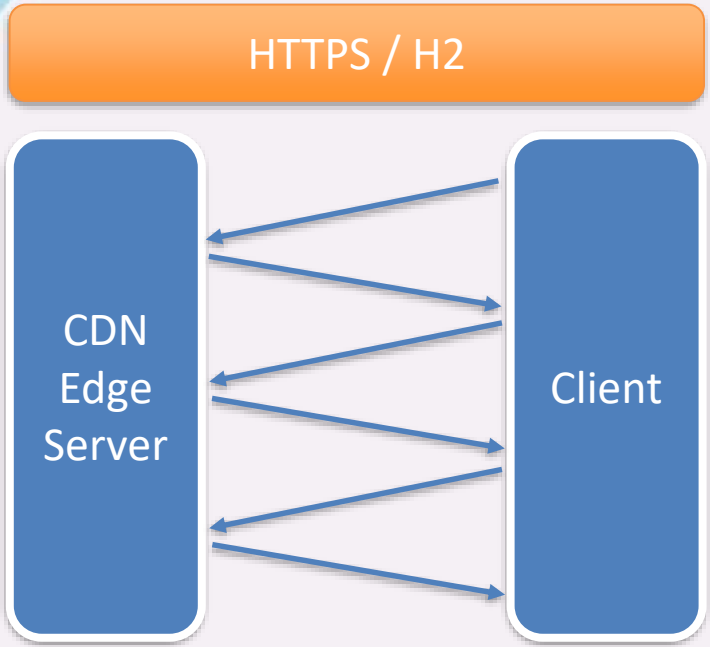
Applying latency model to client-controlled tile-based streaming



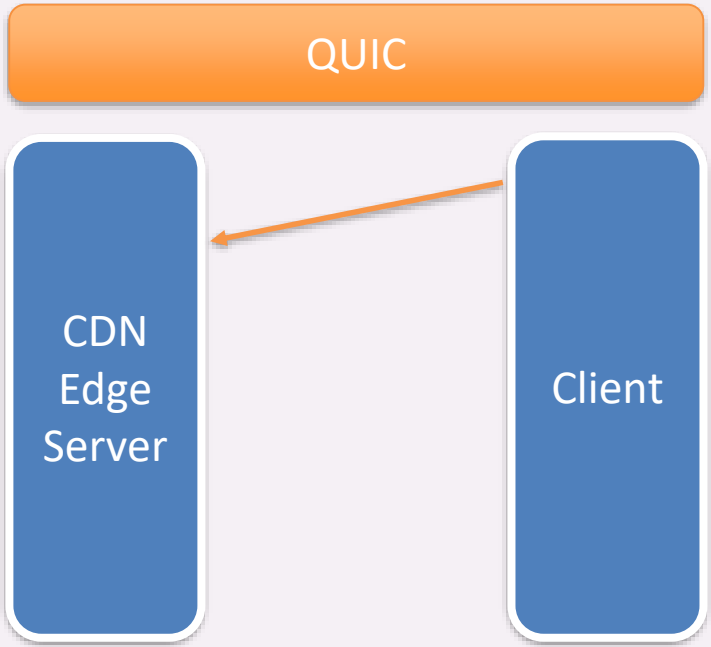
Applying latency model to client-controlled tile-based streaming



Optimization 1: Reducing network response delay by switching to QUIC



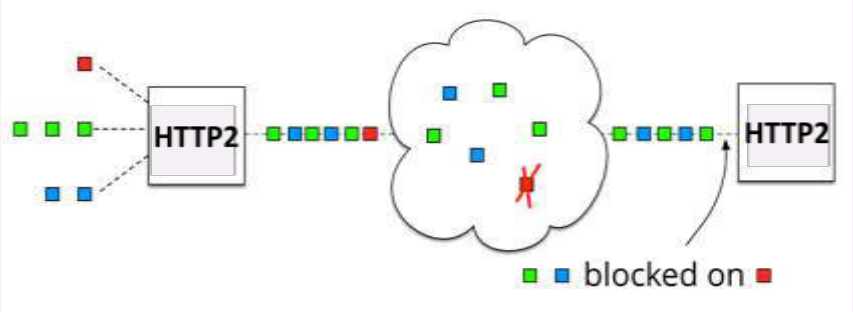
Connection setup latency: 3xRTT
(120ms, assuming 40ms RTT)



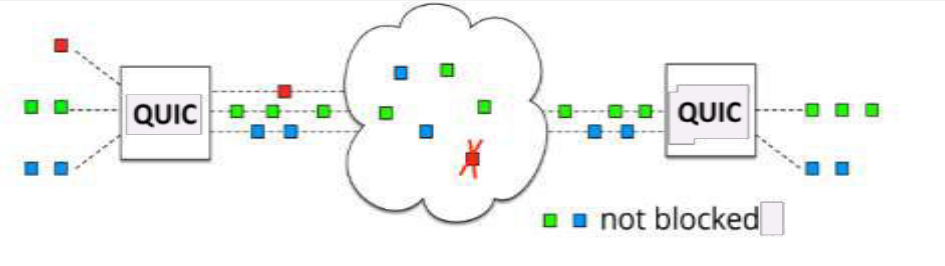
Connection setup latency: 1xRTT (initial)
(40ms, assuming 40ms RTT)

Theoretical QUIC benefits over TCP

HTTP2/TCP requests are multiplexed but can suffer from “head-of-line blocking”

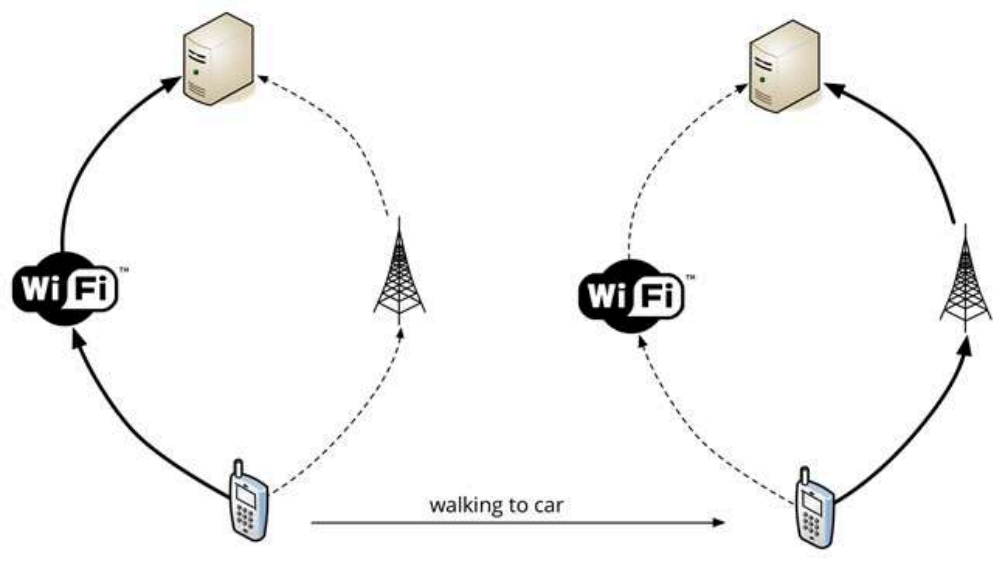


QUIC minimizes issue as only one stream would be impacted



Theoretical QUIC benefits over TCP

We keep the same session with the server even if the underlying network connection source IP changes



Measurement setup

Streaming of 8K sphere (8k x 4k source)

Three types of network conditions simulated:

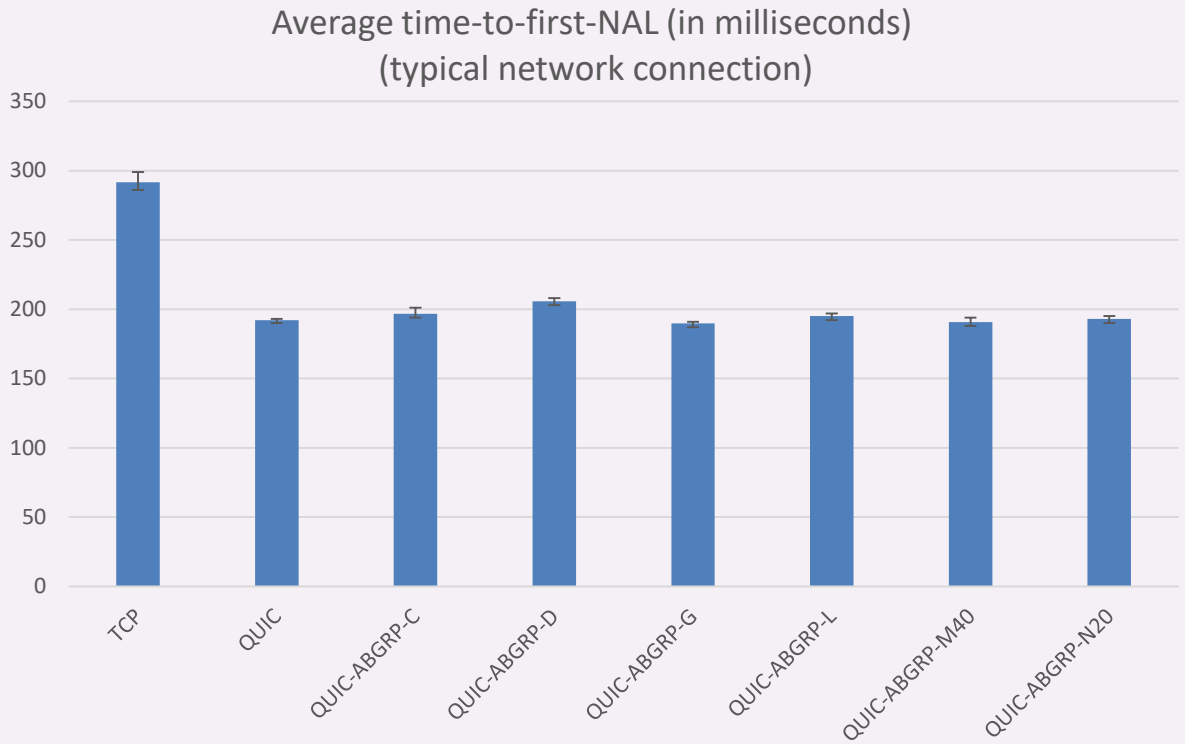
- Very good connection – 80 Mbps, 0% packet loss, 20ms RTT
- Average internet connection *) 20 Mbps, 0.1% packet loss, 65ms RTT
- Bad internet connection – 15 Mbps, 2% packet loss, 90ms RTT

Compare performance using two metrics

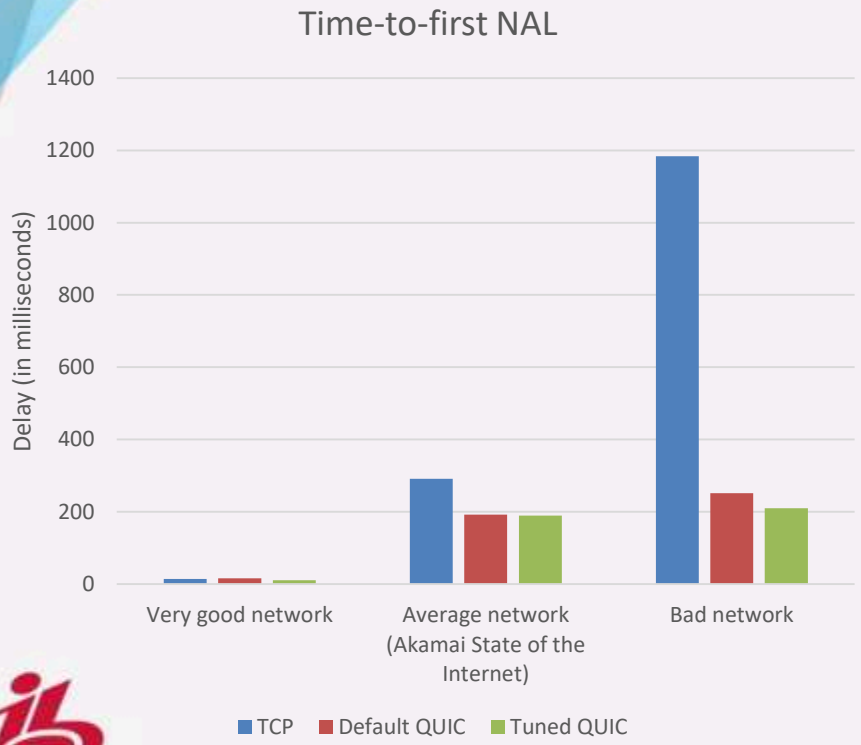
- Time-to-first-NAL (time between head movement and receiving first complete tile)
- Average percentage of low-resolution pixels in viewport

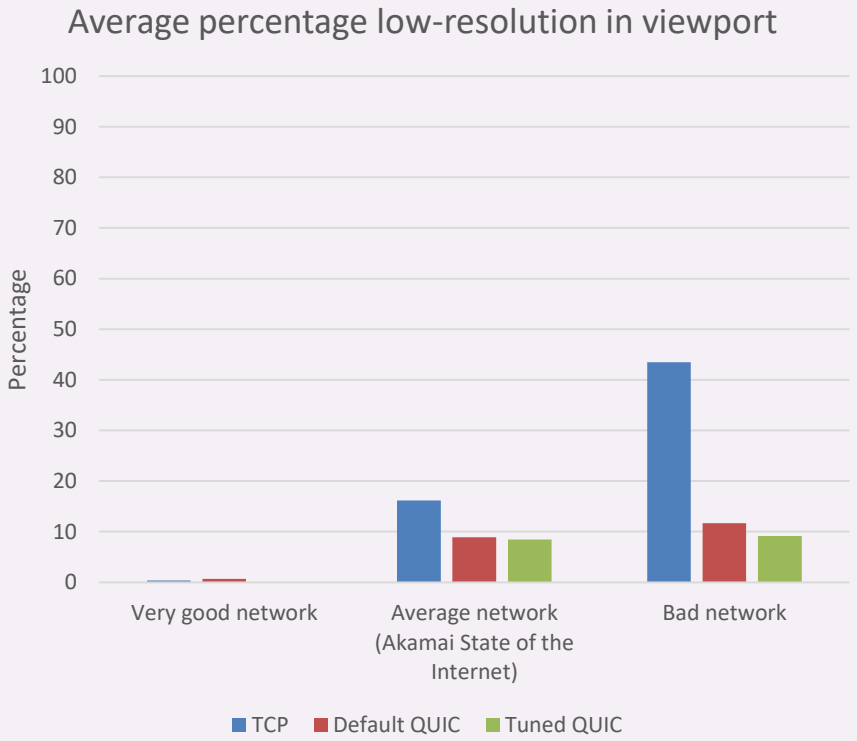
*) Akamai State of the Internet

Performance of TCP-based HTTP/2 vs. various QUIC algorithms

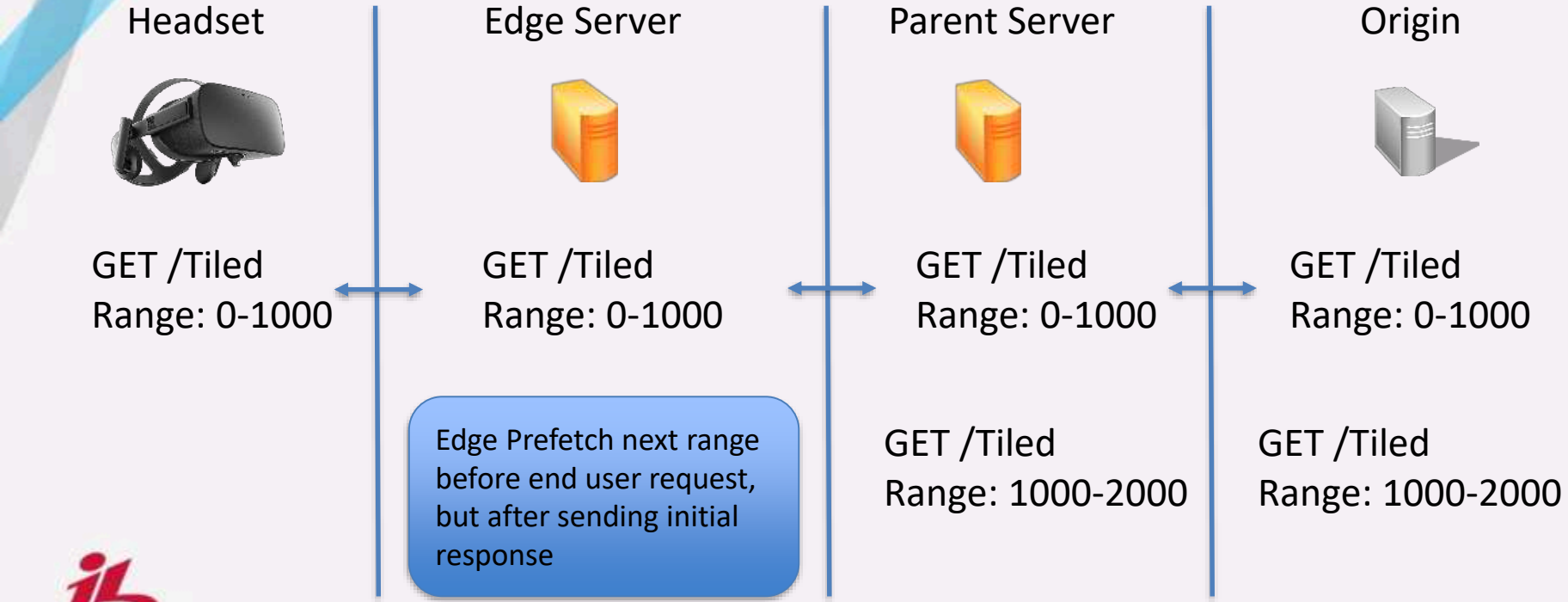


QUIC improvements depend on amount of packet loss
(no head-of-line blocking as in TCP)





Optimization 2: Reducing CDN cache population delay: byte-range requests



Summary

- After optimizations, total end-to-end latency between head movement and high-res tiles in HMD is only 20-40ms using Tiledmedia's ClearVR client – virtually invisible to user
- Demo at Akamai booth, showing real 8K ClearVR streaming from Akamai's CDN!
- Other Tiledmedia ClearVR demos at Ericsson, Viaccess-Orca, DTS and Harmonic booths