Optimizing Motion-to-Photon Latency
on DAQRI Smart Glasses

Heinrich Fink
heinrich.fink@daqri.com
XDC 2018
System Overview

Platform Software $\text{Linux 4.14.x}$

Motion

Camera + Sensors
Visual Inertial Odometry (VIO)

Samples

Tracking

Pose

Rendering
User App

Frame

Display

Photons

Hardware

Intel SKL Gen9
$m7-6Y75$

Display-Port

Himax Ctrl + LCOS

Ideal $\text{motion-to-photon}$ latency:

$\text{AR} < 5 \text{ ms}$ [1,3]
$\text{VR} < 20 \text{ ms}$ [2]
Demo :: Basic Mode

Motion-to-photon latency: ~26 ms
different for each color → rainbow effect
Demo :: Optimized Mode

M2P Latency: ~8 ms
Challenges of AR Compositor Architecture

- Custom interface between App and Compositor
  - Need to attach pose / time to app render buffer (for downstream)
  - Let compositor pace app render cycles
  - Decouple render-rate from display-rate
  - No prevalent standard exists for that (yet?)

- Keep end-to-end pipeline short
  - No triple buffering, no intermediates
  - Updating poses as late as possible w/o stalling the full pipeline

- Compositor/App compete for GPU resources
  - Pre-emption likely needed
The Path Ahead

- Remove the desktop render path
  - Direct use of KMS to flip / access timing info
- Use DRM format modifiers for optimal end-to-end buffer formats
  - Can import into EGL / Vulkan for applications
- Use dma_fence for down/up stream synch and traceability
- Use KMS-exposed hardware planes for simple compositing
  - Although varying timing requirements might be tricky
- Observability through standard tools (GPUView, GPUTop, ...?)
References


