Developing VR Experiences with the Oculus Rift

Tom Forsyth | Software Architect, Oculus
What this talk is *not* about

- **Hardware specs**
  - Constantly evolving
  - Documented elsewhere

- **Use of the SDK and distortion**
  - Read the docs, follow the examples
  - Some unintuitive things there, particularly the lens config – trust us!

- **User calibration**
  - Config utility does this for you
  - Use the profiles, they’re important
  - Discourage users from overriding them (usually a sign of hidden bugs!)
# Topics

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Be kind to your players</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>VOR gain</td>
<td>6</td>
</tr>
<tr>
<td>3</td>
<td>IPD and the neck</td>
<td>7</td>
</tr>
<tr>
<td>4</td>
<td>Changing world scale</td>
<td>8</td>
</tr>
</tbody>
</table>
Be kind to your players

- VR developers spend hours a day looking at an HMD
  - Much of that time, there will be bugs everywhere
  - Our brains soon learn to ignore the crazy

- **Your players do not!**
  - Their brains are fresh and innocent
  - They expect things to be real
  - Hopefully you have debugged everything and have true “presence”

- **If you crank everything to 11, you will traumatize them**
  - They’ll stop playing and give you a one star review 😞 😞 😞
Be kind to your players

- Everyone is wildly different
  - What is intolerable for some is not even visible to others
- There is no one “VR tolerance” slider
  - Someone who is very sensitive to one aspect may tolerate another just fine
  - e.g. going up and/or down stairs
- Tolerance is not simply a skill you can learn
  - There can be negative feedback: people get less tolerant with exposure
- Best Practices Guide contains what we know
  - Use it as a checklist of things to at least think hard about
Be kind to your players

• **Err on the gentler side**
  - Over-intense VR makes it harder to follow plot & game mechanics

• **Make intense experiences optional**
  - Fewer “in your face” particles & explosions
  - Less, slower movement

• **Default low**
  - Let more experienced VR people “opt in”, don’t make newbs “opt out”

• **Make it easy to change any time**
  - Allow dropping to lower intensity to actually play the game after the “VR hit”
Vestibulo-Optical Reflex

OMGWTH?!
• Horizontal slice through the head
• (just showing systems for yaw control)
2 Vestibulo-Optical Reflex

Semi-circular canals in the ears →
Vestibulo-Optical Reflex

Eyeballs and muscles →

Semi-circular canals in the ears
2 Vestibulo-Optical Reflex

Eyeballs and muscles

Reflex neurons ➔

Semi-circular canals in the ears
Vestibulo-Optical Reflex

- Used in “fixation”
  - Static object, moving head
- Head rotation detected by ears
- <10ms later, smooth eye rotation
- Not saccadic!
  - Very smooth
  - Excellent visual quality
VOR gain

- VOR gain is the ratio between ear motion and eye response
- Usually gives 1:1 compensation
  - $+10^\circ$ head motion = $-10^\circ$ eye motion
- Gain fine-tuned during fixation
  - Tries to produce zero “retinal flow”
- Tuning is extremely slow
VOR gain

- What if the view is compressed?
  - A new pair of glasses
  - Incorrect rendering scale in VR
- $10^\circ$ head motion now needs $-5^\circ$ eye motion to maintain fixation
- VOR gain now results in retinal flow
  - Causes disorientation
- Gain adaptation takes 1-2 WEEKS
  - (assuming continuous use!)
Preserving VOR gain

- Games on a monitor often have a “FOV” slider
- Acceptable on a monitor – does not directly affect VOR gain
  - Monitor does not move with the head – no “virtual fixation” happening
  - Peripheral vision of room provides real-world optical flow reality check
  - ...but even then it does cause problems for some
- In the Rift, the only things to fixate on are in VR
  - Retinal flow of VR objects must match real-world motion
- FOV scale in VR is not an arbitrary choice!
  - It must match the HMD+user characteristics
  - Doctor it [redacted by legal] my player’s brains when I do this
Preserving VOR gain

- The Rift display has a physical pitch, aka “pixels per visible degree”
  - Exact value depends on distortion, user’s head & eye position, etc.
  - Found with user configuration tool

- **SDK will help you match this pitch precisely**
  - For a given device & user size, it will give you the right FOV & scale

- **Avoid any changing FOV or “zoom” effects**
  - 10 degrees of head rotation must produce 10 degrees of optical flow
  - Even slight changes in pixels per degree will cause problems for most users
IPD, eye-relief, and the neck
IPD, eye-relief, and the neck

- IPD – Inter-Pupillary Distance
  - That’s all I need, right?
IPD, eye-relief, and the neck

- **IPD** - Inter-Pupillary Distance
  - Actually two components per eye
    - Nose-to-pupil – “half-IPD”
    - Eye-relief – distance from lens surface to pupil
  - NOT related to the dimensions of the HMD!
- **Together form center-to-eye vector**
  - Set during user configuration
  - Stored in user profile
- **Rarely symmetrical**
  - My eye reliefs differ by 2mm
IPD, eye-relief, and the neck

- Center eye pupil - position reported by SDK
  - Centerline of the HMD
  - Average of left & right eye-reliefs
- Roughly where players “feel” they are
  - Audio listener position
  - Line-of-sight checks
  - Origin for reticle/crosshair raycast
IPD, eye-relief, and the neck

- **Origin set by** `sensor->Recenter()`
  - App should have a button to trigger this
  - Player sits in neutral forward pose to press it
  - Also defines “zero yaw”
  - Zero pitch & roll defined by gravity vector
IPD, eye-relief, and the neck

- SDK reports pos & orn of center eye
IPD, eye-relief, and the neck

- SDK reports pos & orn of center eye
- Add on center-to-eye vectors
IPD, eye-relief, and the neck

- SDK reports pos & orn of center eye
- Add on center-to-eye vectors
- Virtual camera positions for rendering
IPD, eye-relief, and the neck

- SDK reports pos & orn of center eye
- Add on center-to-eye vectors
- Virtual camera positions for rendering
- Remember all these are real distances
  - They are real player dimensions and motion
  - They are not your free artistic choice!
  - Changing them can very quickly cause problems
  - ...but there is one thing you can do...
Changing World Scale
Changing World Scale

- You can apply a consistent scale to all three
- Scale center-to-eye and head motion identically
Changing World Scale

- You can apply a consistent scale to all three
- Scale center-to-eye and head motion identically

Identically

NOPE
Changing World Scale

- You can apply a consistent scale to all three
- Scale center-to-eye and head motion identically

No VR giraffes
Changing World Scale

- You can apply a consistent scale to all three
  - Scale center-to-eye and head motion identically
- Same effect as scaling the entire world
  - Very compelling sense of being larger or smaller
- Reducing world scale can help reduce intensity for some people
  - Scales down all motions, accelerations, etc
  - Don’t shrink too small or convergence gets tricky
Changing World Scale

• Monocular mode – IPD of zero
  • An extreme case of scaling mismatch
• Studied in some older research
• Our testing results: it doesn’t work
  • It’s either neutral or bad
  • In some cases, it’s awful
• Many older studies were done with bad VR
  • Maybe it just makes bad VR less bad?
• We strongly urge you not to do this!
How tall is the player?

- Player profile has their actual height
- SDK calculates eye-height-off-ground

Real World

6'0"
How tall is the player?

- Player profile has their actual height
  - SDK calculates eye-height-off-ground
- If playing themselves, use that
  - Exploring an environment
  - Virtual tourism
  - Gives people a known metric & scale
- But if playing another character?
  - e.g. Heavy is much taller
How tall is the player?

Virtual World: 7'3"
Real World: 6'0"
How tall is the player?

- Move the player's eyes to match the height of the character
- But do not change their virtual IPD!
How tall is the player?

- Changing eye height seems to be an aesthetic decision
  - No need to change world scale as well
  - Does not seem to cause disorientation (unlike other physical values)
- Player playing themselves – use their real height
  - Gives people a known metric to measure objects against
- Player playing a specific character – use height of character
  - Often necessary for gameplay reasons – sight lines, framing, etc
• This should work, right?
• But the real-world player isn’t standing up
• This should work, right?
• But the real-world player isn’t standing up
  • They’re seated
  • With feet on the floor
• So the brain can FEEL where the floor is
Perceived World Size – Floor-Dragging

- This should work, right?
- But the real-world player isn’t standing up
  - They’re seated
  - With feet on the floor
- So the brain can FEEL where the floor is
- Now the world is smaller
  - Heavy is tiny baby man!
Perceived World Size – Floor-Dragging

• **Real-world player is sitting down, with feet on the floor**
  • The brain knows where the floor is, it can feel it!

• **The brain scales the visible virtual world using the floor as reference**
  • With a standing avatar, will cause the world to shrink

• **Scaling appears to be higher-level cognitive effect**
  • Conflicts with low-level stereoscopy and parallax cues
  • Effect comes and goes depending on focus
Perceived World Size – Floor-Dragging

- No one-size-fits all solution
Perceived World Size – Floor-Dragging

- No one-size-fits all solution
- Use seated avatars?
  - Works great for driving & flying sims
  - Papers Please VR Edition?
Perceived World Size – Floor-Dragging

- No one-size-fits all solution
- Use seated avatars?
- Scale the world larger to compensate?
  - Limited success, you now have even more conflict between sensory inputs
  - Stereo effect magnified, may be too intense for some
Perceived World Size – Floor-Dragging

- No one-size-fits all solution
- Use seated avatars?
- Scale the world larger to compensate?
- Give in and scale the world smaller to match?
  - Now everyone is an Oompa Loompa
  - But at least the visual cues don’t conflict with the physical ones
  - More relaxing, more immersive
Perceived World Size – Floor-Dragging

- No one-size-fits all solution
- Use seated avatars?
- Scale the world larger to compensate?
- Give in and scale the world smaller to match?
- Make players sit on bar stools?
  - Or any other way of getting feet off the floor
  - Heel rests, “ankle stirrups”, sit cross-legged
Perceived World Size – Floor-Dragging

- No one-size-fits all solution
- Use seated avatars?
- Scale the world larger to compensate?
- Give in and scale the world smaller to match?
- Make players sit on bar stools?
- More research needed
  - ...and this is where we'd love feedback from devs
Transition animations

- In general, never take control of the camera
  - Always keep head-tracking on & faithful
  - Causes significant problems for many users
  - If you must do it, do it FAST – teleport rather than fly
Transition animations

• In general, never take control of the camera
• But sometimes, transitions need to happen for story/world
  • Getting into/out of vehicles
  • Getting into/out of bed
  • Standing up after knockdown
  • Picking an object off the floor
Transition animations

- In general, never take control of the camera
- But sometimes, transitions need to happen for story/world
- Most people find these too intense in VR
  - Especially orientation changes
Transition animations

- In general, never take control of the camera
- But sometimes, transitions need to happen for story/world
- Most people find these too intense in VR
- Option: show your avatar doing the action
  - Be careful of the 1st to 3rd person transition
  - Try a ghostly/transparent avatar
Transition animations

- In general, never take control of the camera
- But sometimes, transitions need to happen for story/world
- Most people find these too intense in VR
- Option: show your avatar doing the action
- Option: use a dissolve or fade-through-black
  - Needs to be live rendering, not a screenshot
  - Maintain head-tracking all the time - let the player look around
Transition animations

- In general, never take control of the camera
- But sometimes, transitions need to happen for story/world
- Most people find these too intense in VR
- Option: show your avatar doing the action
- Option: use a dissolve or fade-through-black
- Option: use a “blink”
  - Fuzzy-edged black borders top & bottom
  - Close, teleport, open
  - If ~300ms, some players don’t even “see” them!
Animated avatars

- Highly animated 1st-person avatars are awesome
  - Amazing sense of immersion and presence
  - TF2 examples:
    - High fives
    - Yelling "medic" – hand comes to face
    - Sniper’s bird – Sir Hootsalot / Steel Songbird
Transition animations

- **Highly animated 1st-person avatars are awesome**
  - Amazing sense of immersion and presence
  - TF2 examples:
    - High fives
    - Yelling "medic" – hand comes to face
    - Sniper’s bird – Sir Hootsalot / Steel Songbird

- **Constraints:**
  - Virtual camera should always move with player’s real head
  - Virtual camera and avatar’s head should coincide for best immersion
Transition animations

- **Highly animated 1st-person avatars are awesome**
  - Amazing sense of immersion and presence
  - TF2 examples:
    - High fives
    - "Medic" – hand comes to face
    - Sniper's bird – Sir Hootsalot / Steel Songbird

- **Constraints:**
  - Virtual camera should always move with player's real head
  - Virtual camera and avatar's head should coincide for best immersion
  - ...but those conflict!
Solution:
Meathook avatars
Meathook avatars

- Play animation on the avatar
Meathook avatars

- Play animation on the avatar
- Find avatar’s animated head position
  - May need to add a “center eye” bone to the skeleton
Meathook avatars

- Play animation on the avatar
- Find avatar’s animated head position
- Decapitate
  - e.g. shrink the head bone to zero size
  - Otherwise you see teeth & eyeballs from the inside
Meathook avatars

- Play animation on the avatar
- Find avatar’s animated head position
- Decapitate
- Find player’s virtual camera position
  - Standard head-tracking data from the SDK
Meathook avatars

- Play animation on the avatar
- Find avatar’s animated head position
- Decapitate
- Find player’s virtual camera position
- Hang the avatar on the hook
  - Fix head position to player’s position
  - Retain existing orientation
Meathook avatars

- Play animation on the avatar
- Find avatar’s animated head position
- Decapitate
- Find player’s virtual camera position
- Hang the avatar on the hook
  - Fix head position to player’s position
  - Retain existing orientation
  - Body thrashes around underneath with animations
  - Result in external debug camera is really quite gruesome
  - But it works great in VR!
Framerate & timewarp

- Rock-solid, high FPS is vital to sense of presence in VR
- DK2 is 75Hz, Crescent Bay is 90Hz
- Image update at less than native speed is nauseating
- Stereo rendering at these speeds is challenging
  - Aggressively drop details and effects to maintain framerate and low latency
  - Maintaining presence gives far more player enjoyment than extra effects
- Main costs are draw calls and fillrate
Framerate & timewarp – draw calls

- **Twice as many eyes, so twice as many calls**
  - New APIs (Mantle, DX12, etc) should make multi-submission cheaper

- **Some things only need doing once**
  - Culling – use a conservative frustum that includes both eyes
  - Animation
  - Shadow buffer rendering
  - Some distant reflections/gloss maps/AO renders – but not all!
  - Some deferred lighting techniques
Framerate & timewarp – draw calls

- **Render once, submit each object twice?**
  - Pack both eyes on the same RT
  - Set viewport & view (left eye)
  - Render object
  - Set viewport & view (right eye)
  - Render object
  - No extra shader & texture changes compared to mono

- **But some cards (especially tilers) hate changing viewport**

- **Both eyes on same texture can reduce parallelism**
Framerate & timewarp – fill rate

• Change size of the virtual camera renders, NOT the framebuffer size
  • e.g. DK2 framebuffer is always 1920x1080 – don’t change this!
• But camera-eye renders typically 1150x1450 per eye
  • Depends on shape of user’s face & eye position – set by profile & SDK
Framerate & timewarp – fill rate

- Change size of the virtual camera renders, NOT the framebuffer size
  - e.g. DK2 framebuffer is always 1920x1080 – don’t change this!
- But camera-eye renders typically 1150x1450 per eye
  - Depends on shape of user’s face & eye position – set by profile & SDK
- Scaling this render is absolutely fine
  - Same concept as choosing rez on a normal monitor
  - Distortion correction pass will resample & filter it anyway
Framerate & timewarp – fill rate

- Change size of the virtual camera renders, NOT the framebuffer size
  - e.g. DK2 framebuffer is always 1920x1080 – don’t change this!
- But camera-eye renders typically 1150x1450 per eye
  - Depends on shape of user’s face & eye position – set by profile & SDK
- Scaling this render is absolutely fine
  - Same concept as choosing rez on a normal monitor
- Scaling it dynamically every frame is also fine – nearly invisible
  - If you have lots of particles/explosions that frame, drop the size
  - Use the same RT, just use a smaller part of it
  - SDK explicitly supports this use case
Framerate & timewarp – fill rate

- Conceptually timewarp does three basically separate things at once:
  
  1. Correct for a rolling shutter
  
  2. Late re-prediction of head orientation and correction

  3. Magically fix up low framerates
Framerate & timewarp – fill rate

- Conceptually timewarp does three basically separate things at once:

  1. Correct for a rolling shutter
     - This is always on, and you don’t need to worry about it
  2. Late re-prediction of head orientation and correction
  3. Magically fix up low framerates
Framerate & timewarp – fill rate

- Conceptually timewarp does three basically separate things at once:

1. Correct for a rolling shutter
   - This is always on, and you don’t need to worry about it
2. Late re-prediction of head orientation and correction
   - Mostly automatic, some nuances about how late we can re-read
3. Magically fix up low framerates
Framerate & timewarp – fill rate

- Conceptually timewarp does three basically separate things at once:
  1. Correct for a rolling shutter
     - This is always on, and you don’t need to worry about it
  2. Late re-prediction of head orientation and correction
     - Mostly automatic, some nuances about how late we can re-read
  3. Magically fix up low framerates
Framerate & timewarp – fill rate

- Conceptually timewarp does three basically separate things at once:

1. Correct for a rolling shutter
   - This is always on, and you don’t need to worry about it

2. Late re-prediction of head orientation and correction
   - Mostly automatic, some nuances about how late we can re-read
Framerate & timewarp – fill rate

- Conceptually timewarp does three basically separate things at once:
  1. Correct for a rolling shutter
     - This is always on, and you don’t need to worry about it
  2. Late re-prediction of head orientation and correction
     - Mostly automatic, some nuances about how late we can re-read
Framerate & timewarp – fill rate

- Conceptually timewarp does three basically separate things at once:

1. Correct for a rolling shutter
   - This is always on, and you don’t need to worry about it

2. Late re-prediction of head orientation and correction
   - Mostly automatic, some nuances about how late we can re-read

3. Prevents illness from dropped frames
Framerate & timewarp – fill rate

- Conceptually timewarp does three basically separate things at once:

1. Correct for a rolling shutter
   - This is always on, and you don’t need to worry about it

2. Late re-prediction of head orientation and correction
   - Mostly automatic, some nuances about how late we can re-read

3. Prevents illness from dropped frames
   - With TW, occasional “PC stutter” isn’t like being slapped in the face
   - Prevents damage to HMD from being hurled at wall
   - Slow framerates still look bad – there’s no magic
Lessons learned

- **Be kind to your players**
  - Default to low intensity, let the brave ones pick MUCH WOW mode

- **VOR gain**
  - FOV scale is not an arbitrary knob to play with – follow the player’s profile

- **IPD and head motion**
  - Keep them in sync – follow the player’s profile

- **Changing world scale & how tall is the player?**
  - Aesthetic choices, odd perceptual effects, but fortunately few disorientation problems

- **Transition animations**
  - Try to avoid, but if you must, a teleport is better than continuous motion

- **Meathook avatars**
  - Gruesome in debug cams, looks great from the inside

- **Framerate & timewarp**
  - Timewarp isn’t magic, framerate is still #1 priority
Further reading, search for
“Oculus VR Best Practices Guide”

www.OculusVR.com
Tom.Forsyth@oculus.com
Thank You

Tom Forsyth