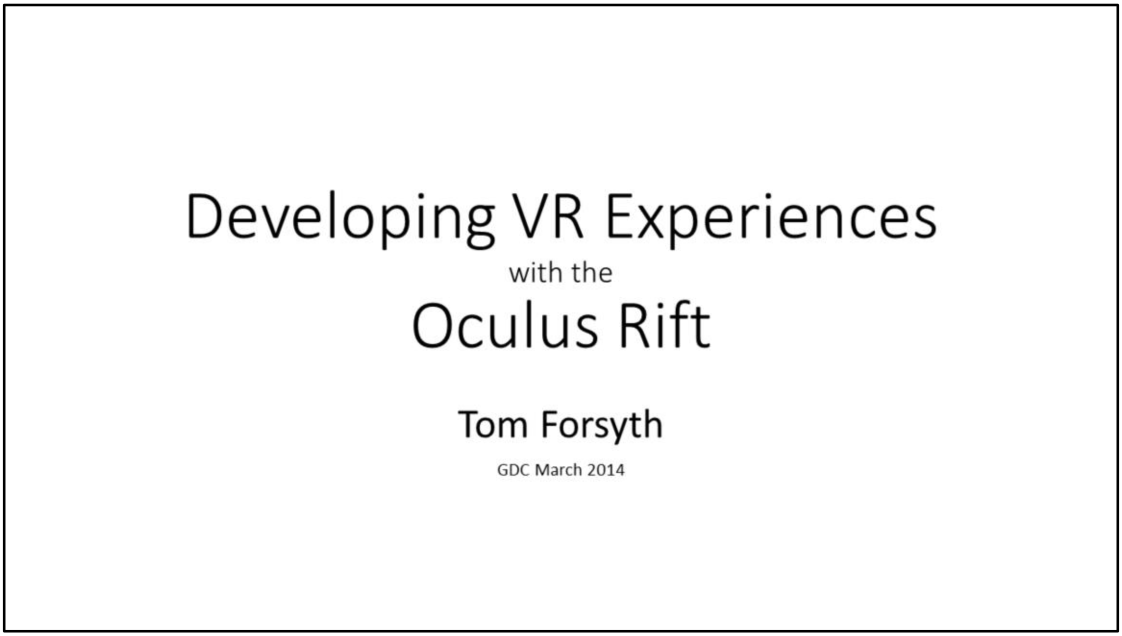


Developing VR Experiences

with the

Oculus Rift

Tom Forsyth

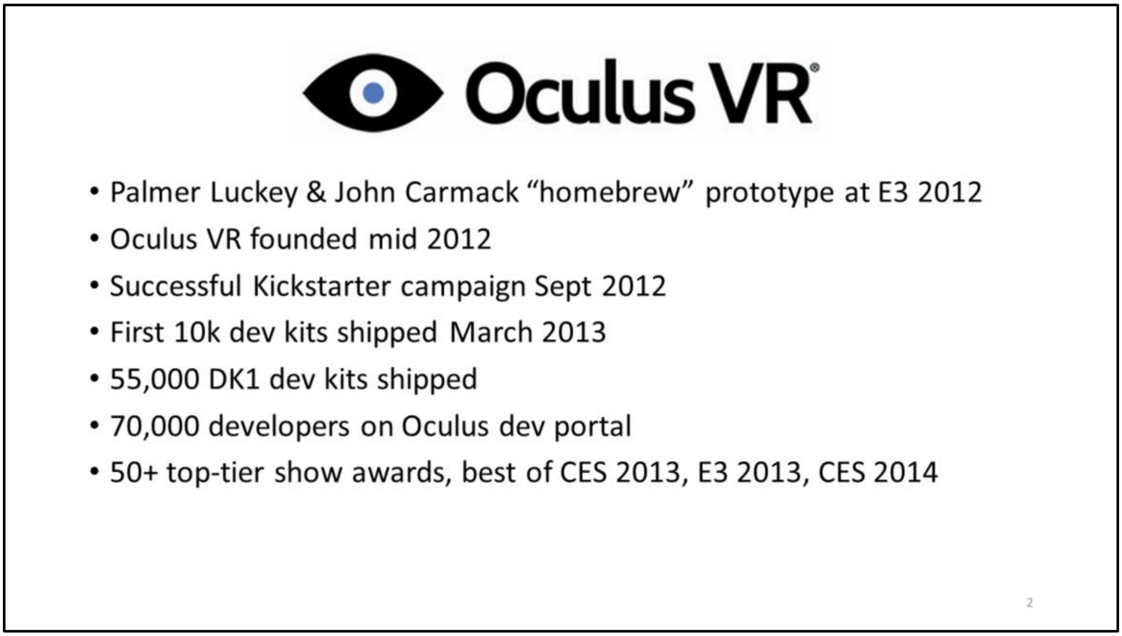
GDC March 2014

I don’t have traditional “speaker notes” because I tend to use the slides as my notes and then speak off-the-cuff. This session was videoed, and that will be up on the web. So I’m going to use these speaker notes to add further background or details that time didn’t allow, or that only a few will care about.

People ask about my background:
- Graphics coder most of my life
- Shipped three games at Muckyfoot Productions: Urban Chaos, Startopia, Blade 2
- Joined RAD Game Tools, worked on Granny3D animation middleware
- There I met Michael Abrash, and joined him on the Larrabee project for Intel
- Ended up designing much of the instruction set, and joined Intel to continue as Larrabee turned into Knights/XeonPhi
- Abrash seduced me back to games, joining his VR group at Valve
- With Joe Ludwig, converted Team Fortress 2 and Half Life 2 to VR
- Moved to Oculus to work on the rendering and SDK

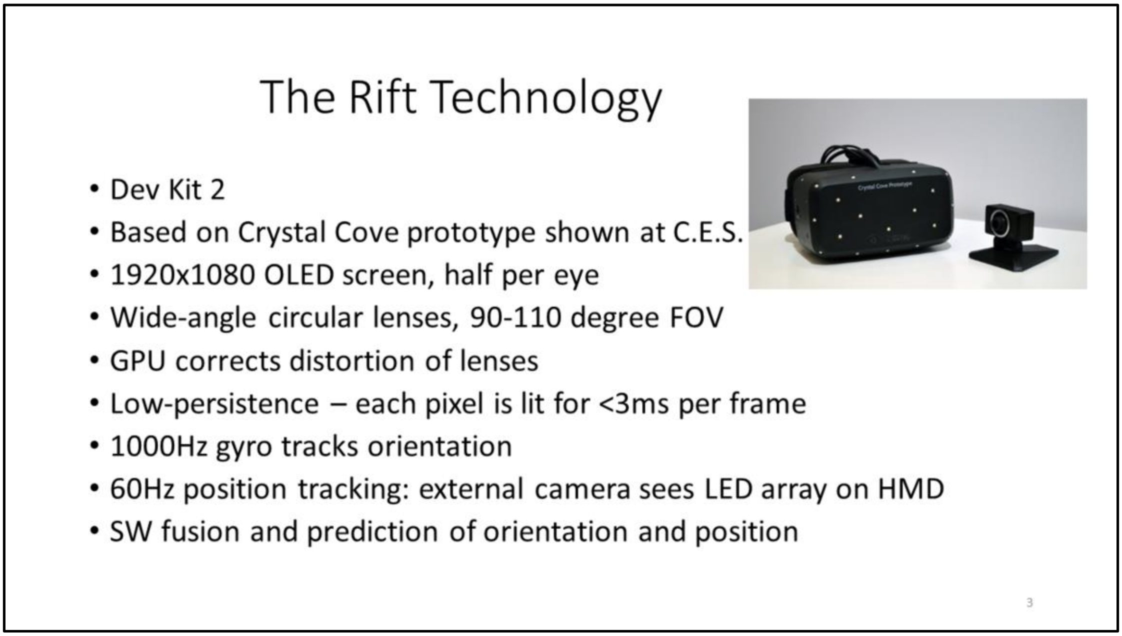
This talk is things that we have learned from shipping games, both ones I’ve done (TF2/HL2) and ones that our dev partners have done, such as EVE Valkyrie and

Hawken, and major kudos to our psychology department (yes, we have one!) for adding a lot more rigor to our hacky experiments.



Oculus VR®

- Palmer Luckey & John Carmack “homebrew” prototype at E3 2012
- Oculus VR founded mid 2012
- Successful Kickstarter campaign Sept 2012
- First 10k dev kits shipped March 2013
- 55,000 DK1 dev kits shipped
- 70,000 developers on Oculus dev portal
- 50+ top-tier show awards, best of CES 2013, E3 2013, CES 2014



The Rift Technology

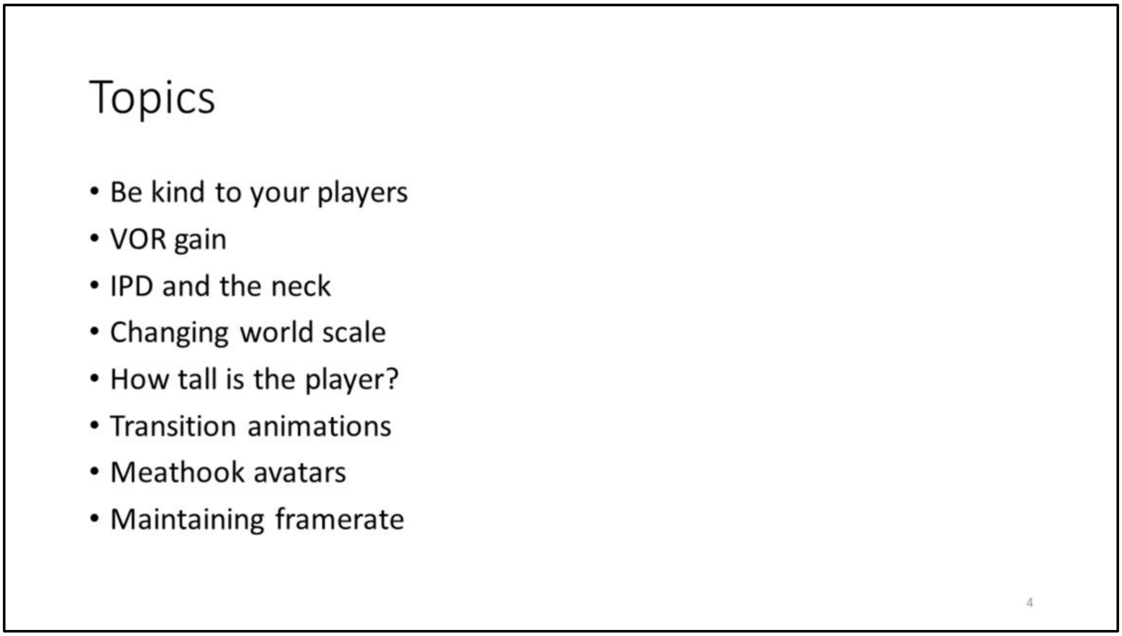
- Dev Kit 2
- Based on Crystal Cove prototype shown at C.E.S.
- 1920x1080 OLED screen, half per eye
- Wide-angle circular lenses, 90-110 degree FOV
- GPU corrects distortion of lenses
- Low-persistence – each pixel is lit for <3ms per frame
- 1000Hz gyro tracks orientation
- 60Hz position tracking: external camera sees LED array on HMD
- SW fusion and prediction of orientation and position

3

The picture here is of Crystal Cove, which is why the LEDs are visible. Note the LEDs on the top, side and bottom, which give us tracking over very wide angles. DK2 has almost the same pattern of LEDs, but they're covered by an IR-translucent shell, so they're not visible in visible light. The DK2 camera has a much better stand as well – it takes up less desk space, and can fold out to clip to the top of a monitor.

One thing people misunderstand about the camera is that it is NOT the primary tracking system. It's too slow (yes, 60Hz is slow in this context!) and relatively high-latency (20-30ms depending on USB bandwidth and CPU speed) to be used directly for tracking and prediction. The gyro and accelerometer are the main high-speed low-latency tracking inputs. The problem (as has been well-documented) is that they're unstable – because you have to integrate their data to produce absolute position and orientation, they can quickly drift into uselessness. In the case of the accelerometer, it's really only stable for about a second before it heads off towards Jupiter. What the camera does is constantly provide a "reality check" that sensor fusion uses to stop this drift. It doesn't matter that the camera data comes in late – we can retrospectively apply this correction to make sure we always have very stable low-latency data.

Fortunately, the SDK deals with all this for you and you really don’t need to care about it.



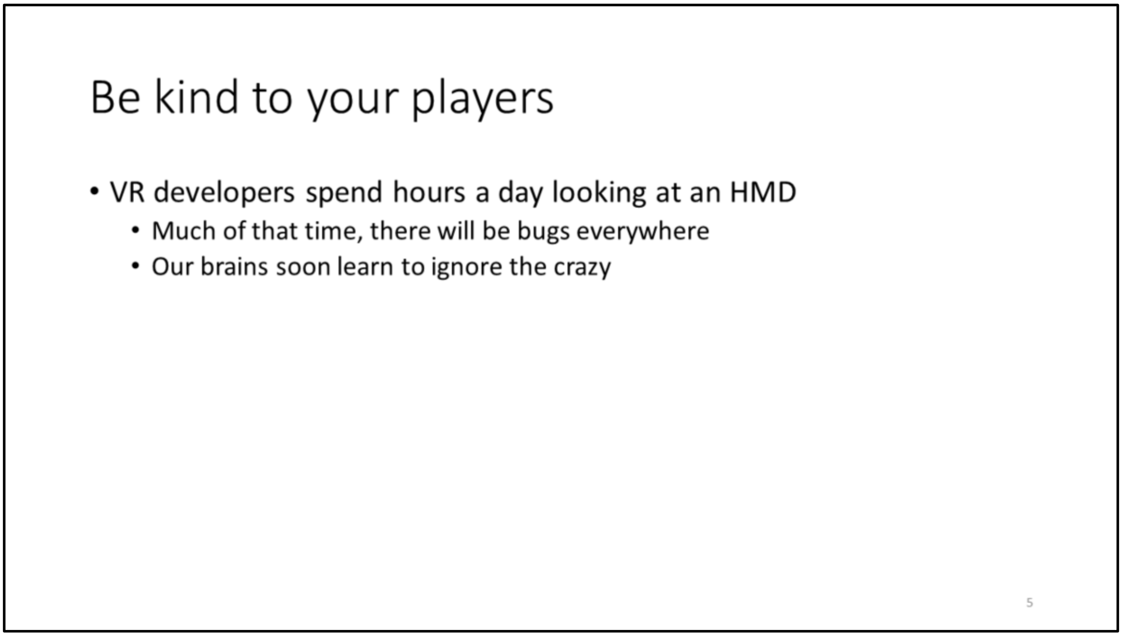
Topics

- Be kind to your players
- VOR gain
- IPD and the neck
- Changing world scale
- How tall is the player?
- Transition animations
- Meathook avatars
- Maintaining framerate

4

I’m not going to talk about all the low-level details of the Rift, because the SDK should do most of that stuff for you, or at least hold your hand through the process (e.g. distortion) enough that it’s not something that you need to worry about. If it doesn’t, then the SDK isn’t doing its job, I’ve screwed up, and it needs fixing – send me email and we’ll get on it.

What I am going to talk about is the decisions that a game developer does need to think about, because they are either difficult tradeoffs, unsolved problems, or where the solution is very dependent on the style or content of the game. I will also highlight a few more straightforward things that we’ve seen multiple people get wrong, just to focus peoples attention on getting it right.



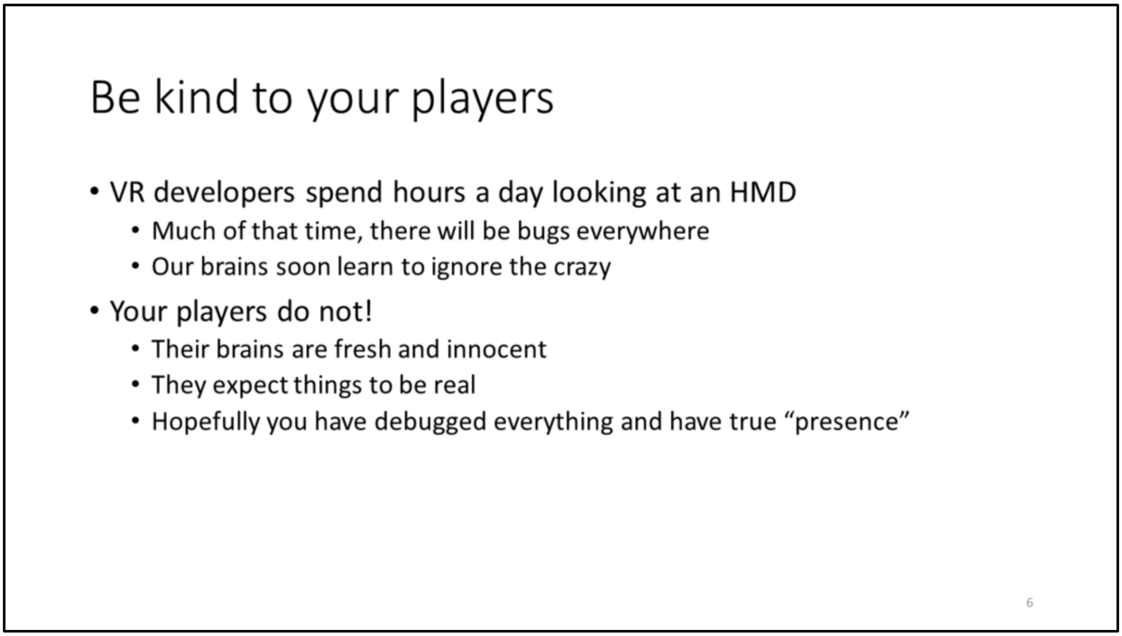
Be kind to your players

- 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

5

In fact as soon as we get rid of all the bugs, we ship it!

Our brains actually have multiple perceptual “modes” – similar to the way someone who wears glasses will get used to how the world looks with them on, and with them off, and can switch between those modes very quickly when they put on or take off their glasses. Developers spend enough time in (buggy) VR that we tend to develop these two modes of perception.



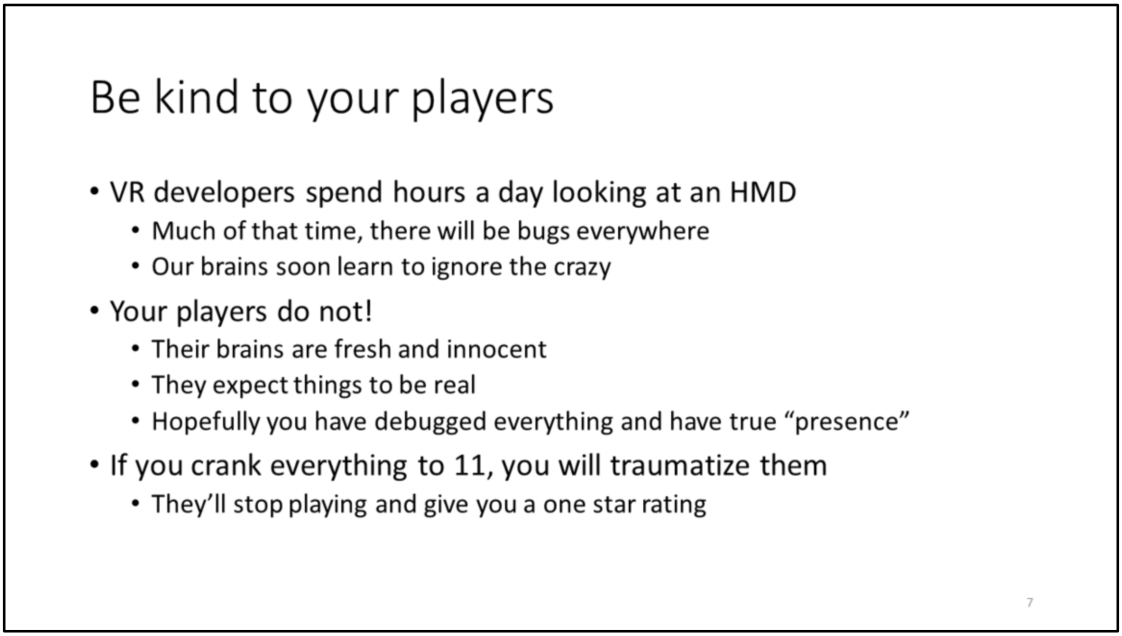
Be kind to your players

- 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”

We think there’s the VR equivalent of the “uncanny valley” – and it’s certainly a useful metaphor.

If you display something that is blatantly wrong, the brain will quickly spot that and ignore it, and you won’t cause people problems. On the other hand, it’s not much fun, and you lose presence.

But if you display something close enough to correct, the brain will really try to interpret it as correct. And the closer to correct it is, the more the brain will try to do this, and the less conscious you will be of those errors. But those errors will still be there, and they’ll cause confusion, especially if they disagree with the proprioceptive and vestibular senses. So you really need to get things bang on, not just close.



Be kind to your players

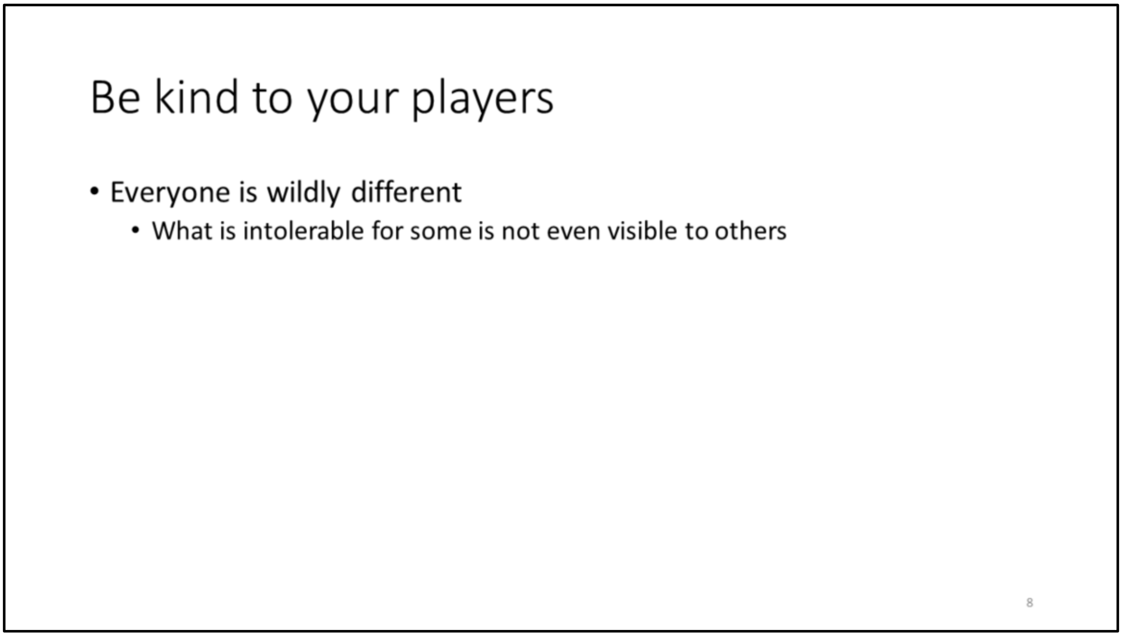
- VR developers spend hours a day looking at an HMD
  - Much of that time, there will be bugs everywhere
  - Our brains soon learn to ignore the crazy
- Your players do not!
  - Their brains are fresh and innocent
  - They expect things to be real
  - Hopefully you have debugged everything and have true “presence”
- If you crank everything to 11, you will traumatize them
  - They’ll stop playing and give you a one star rating

7

Here I’m not talking about errors, just about the experience. Assume a completely “perfect” VR experience – no bugs, no sensory conflicts – you’re really THERE. But now imagine being in this thoroughly realistic world, but with existing standard gameplay, where you’re shooting someone in the face every few seconds, double-jumping through the air at 30mph – it’s way too intense!

Team Fortress 2 is a good example. On a technical level, it’s a pretty decent port to VR (modesty is not my strong point). And although it’s non-stop-face-shooting action, when playing on a monitor I can “zen out” and play it for hours, even while thinking about other things. But in VR it’s so “invasive”, that I can’t easily do that – I can play it for about half an hour, and then even with the cartoony graphics and comedy hats, I need a break to let my heart rate return to normal.

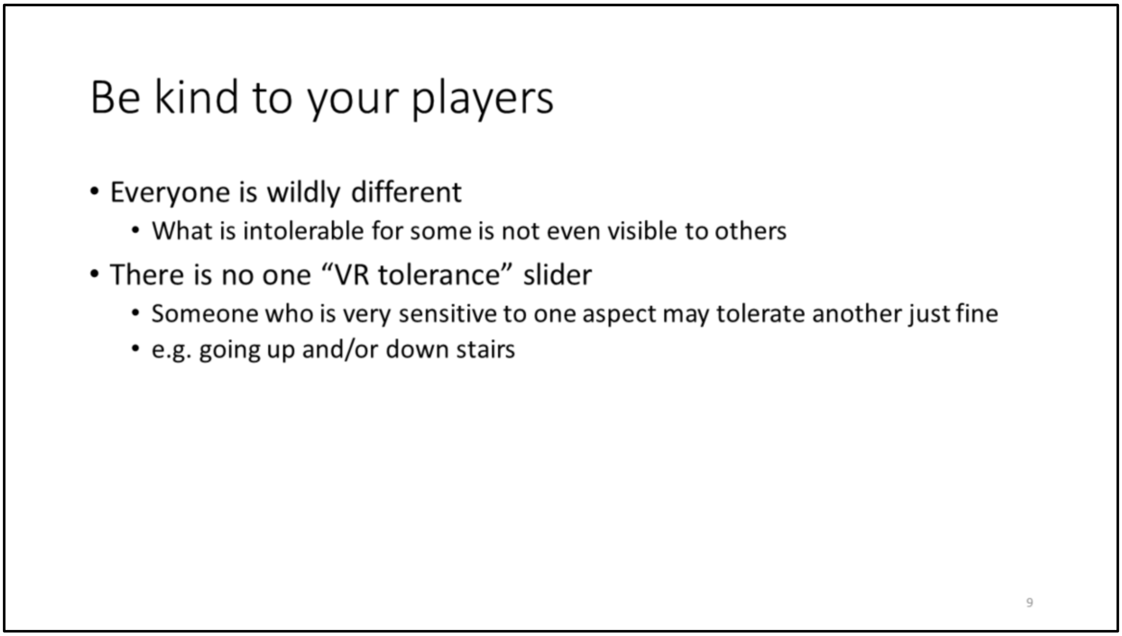
It is very difficult for people in VR to detach themselves from the game world. In some ways it’s like trying to sleep in a cinema – your only sensory inputs are from the film. Indeed that’s why we go to the cinema. So we need to deliberately give people space to relax. Pacing is important in games on all media, but that’s even more true in VR.



Be kind to your players

- Everyone is wildly different
  - What is intolerable for some is not even visible to others

This is one of the hardest things to learn about VR. You are the worst subject, and you are absolutely not representative of your audience.



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

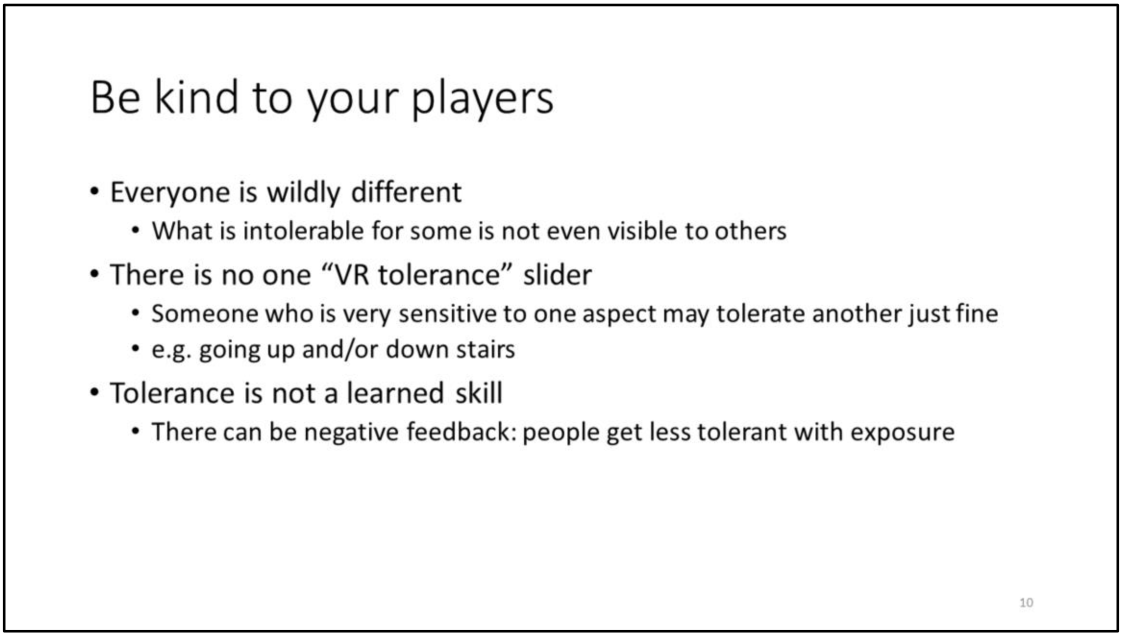
9

The stairs example was a revelation. In the VR group at Valve, we had a wide range of VR motion-tolerance (and I am one of the more sensitive), but for whatever reason none of us working in VR had problems with stairs. We didn’t even know it was a “thing” we needed to be careful of.

Once we started wider internal testing of TF2 on the Rift with other Valve devs, we found some people had trouble with one particular aspect of TF2 – going up and down stairs. People who otherwise had no problems running and jumping around the map at speeds that made me feel ill, would have real senses of vertigo when walking up or down stairs or steep slopes.

Even stranger, some people had no problem going down stairs, but going up unnerved them – it was something to do with the visual effect of the stairs scrolling in front of their eyes, because it wore off as they reached the top of the stairs and could see the next floor. And a few weeks ago I finally met the rarest example – someone who was fine going up stairs, but going down was a big problem.

In short, our lovely precision-built hardware and software is being completely ruined by the unpredictable squishy bag of meat that the HMD is mounted to.

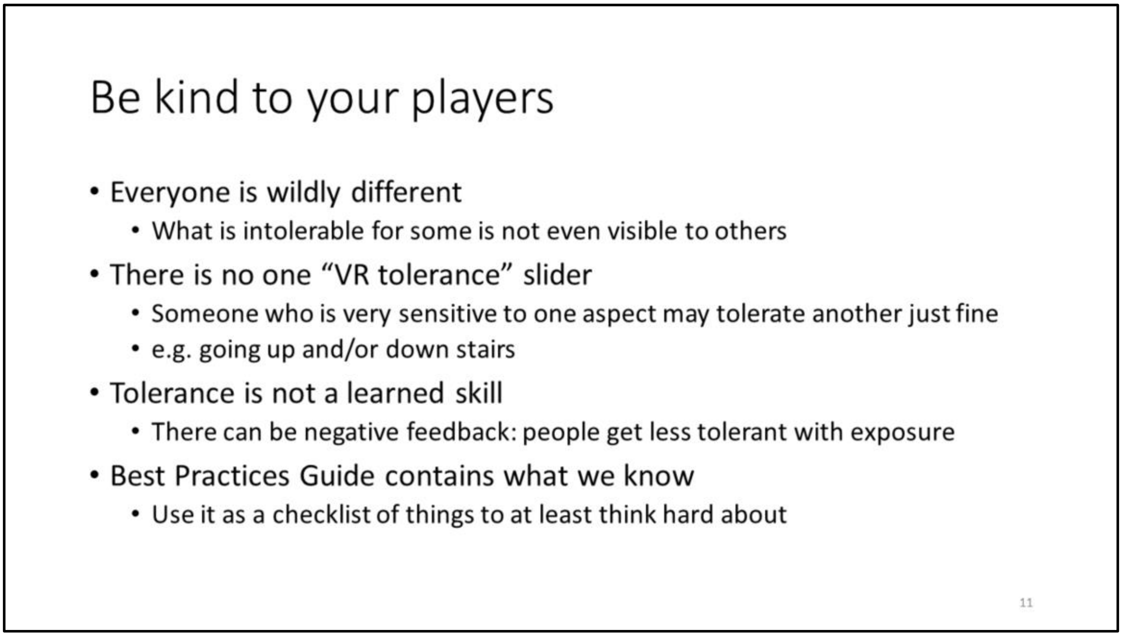


Be kind to your players

- 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 a learned skill
  - There can be negative feedback: people get less tolerant with exposure

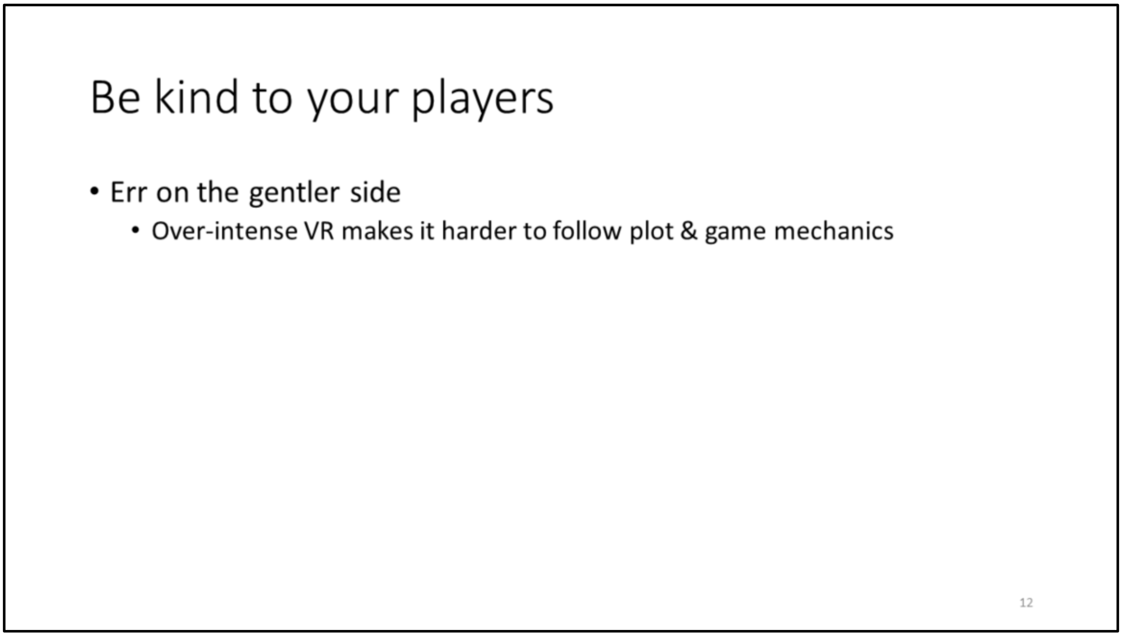
10

This negative feedback is somewhat like “aversion therapy” - except it actually works. Having discussed simulator sickness far too much over the past two years, I now actually GET nausea just by using the WORD nausea. So I’ll stop typing “nausea” and go to the next slide.



Be kind to your players

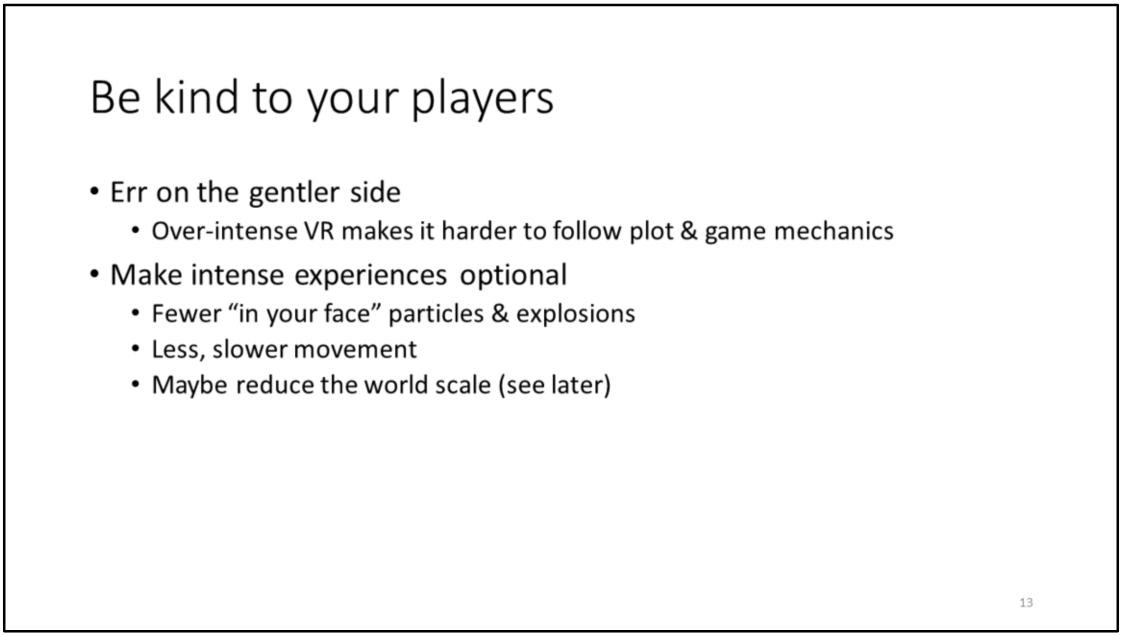
- Everyone is wildly different
  - What is intolerable for some is not even visible to others
- There is no one “VR tolerance” slider
  - Someone who is very sensitive to one aspect may tolerate another just fine
  - e.g. going up and/or down stairs
- Tolerance is not a learned skill
  - There can be 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

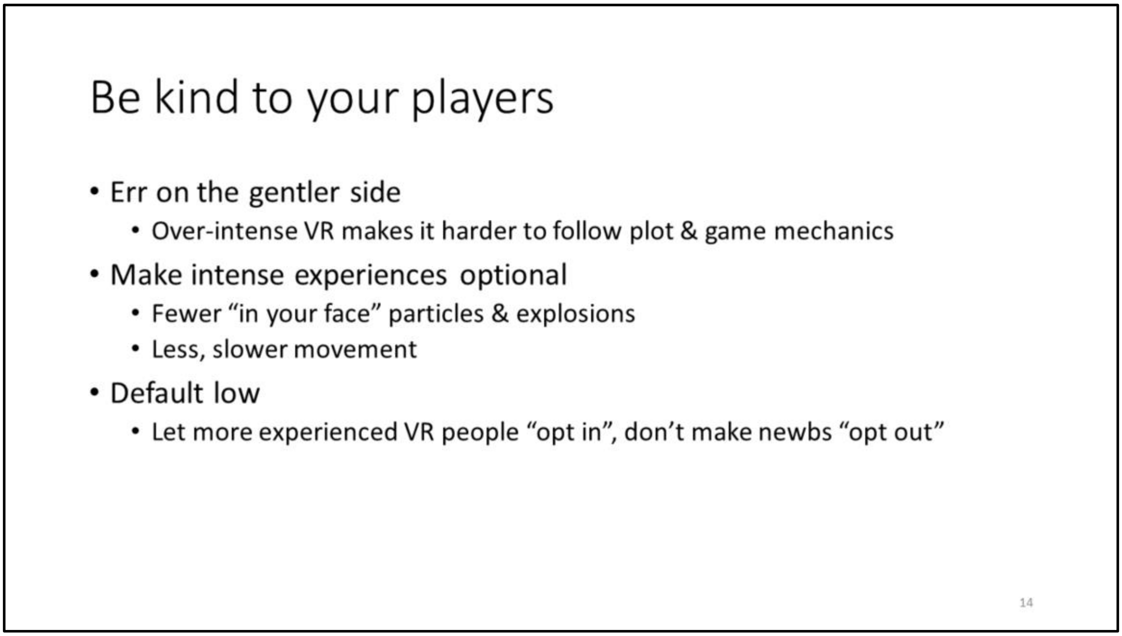
- Err on the gentler side
  - Over-intense VR makes it harder to follow plot & game mechanics

If the player is thinking “oh wow I’m in VR”, that’s not presence! We’re trying to make them FORGET they’re in VR. In fact, we’re trying to make it require a conscious effort to remember that they’re not actually in the world. We’re not there quite yet, but getting closer. You want them to be focusing on the experience your designers have spent so long crafting, not just looking around saying “it’s soooo 3D”.



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
  - Maybe reduce the world scale (see later)

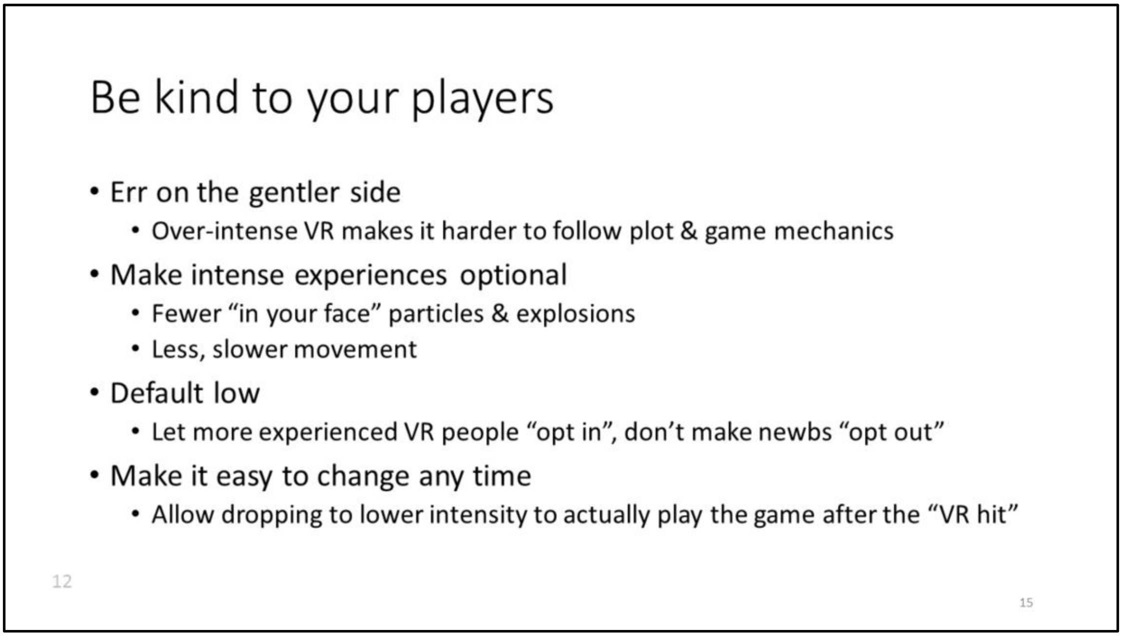


Be kind to your players

- 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”

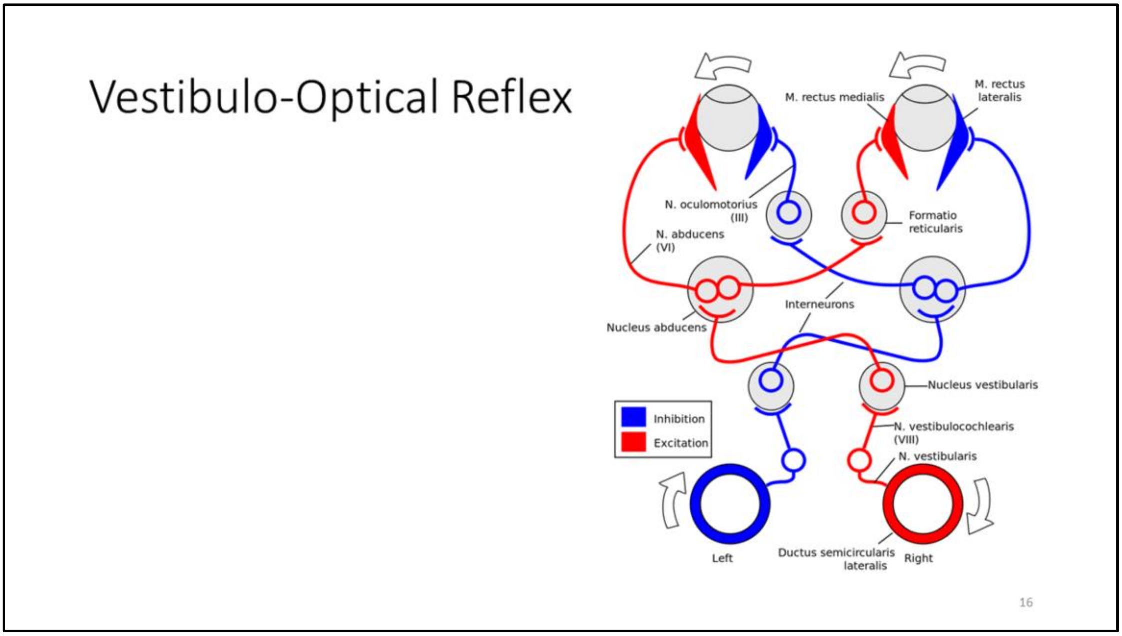
Our lovely hardcore VR fans enjoy hunting down graphics settings to make things more immersive, even if it then makes them ill. That’s fine because they understand it’s their choice, and they keep their GinGins handy.

But please don’t force this on people who haven’t been in VR much.



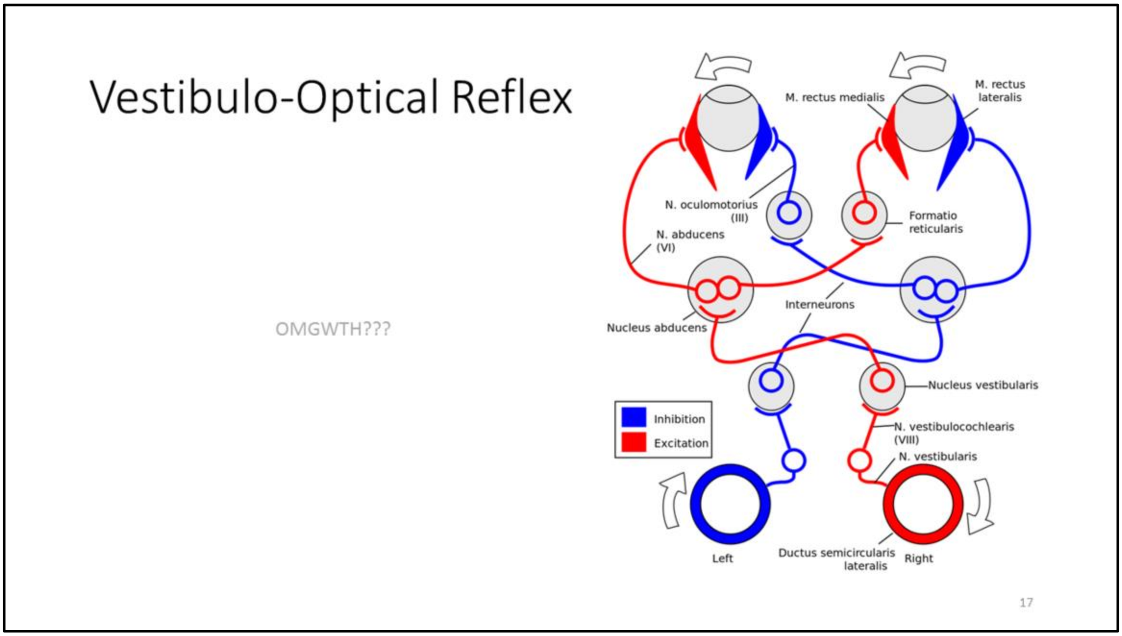
Be kind to your players

- 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

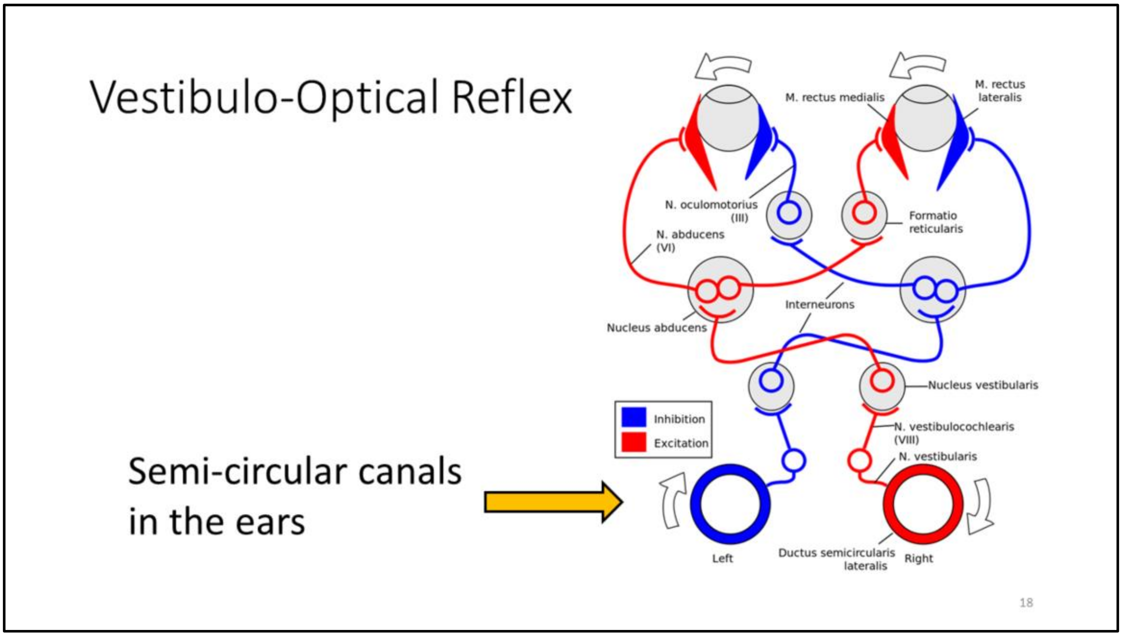
Voice: "this is a simple diagram I found on Wikipedia. I think it's pretty obvious what's going on here, it really doesn't need much explanation."



Vestibulo-Optical Reflex

OMGWTH???

I’m such a troll.

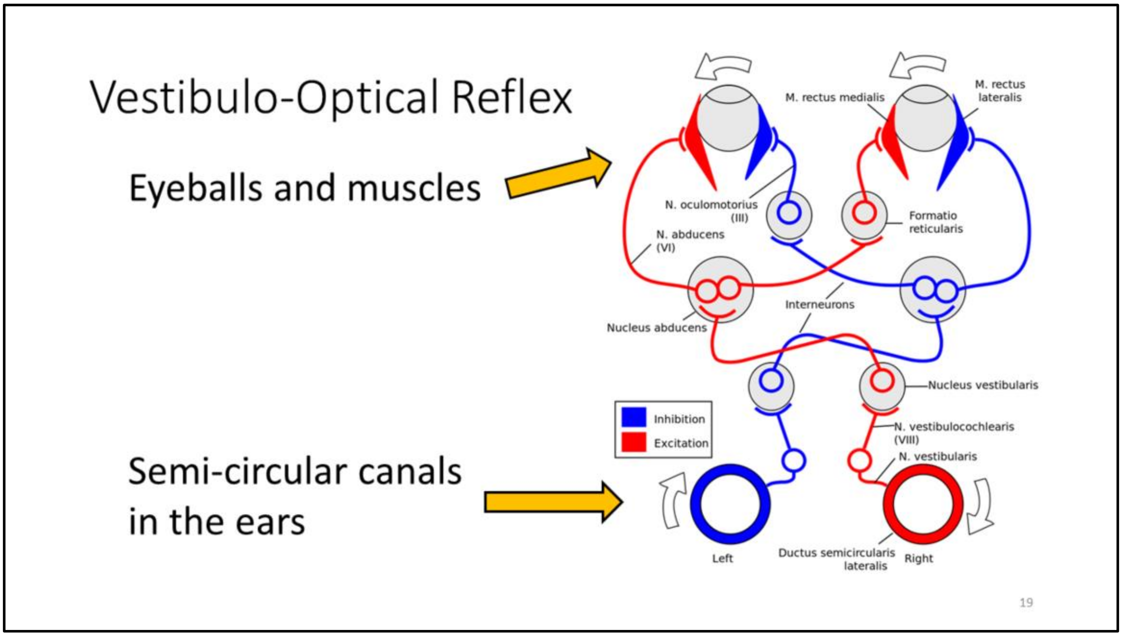


Vestibulo-Optical Reflex

Semi-circular canals in the ears

18

This is a horizontal slice through the head at ear/eye level. For simplicity it only shows the components relating to horizontal rotation of the head. There are similar systems for vertical rotation, and also rolling of the head.

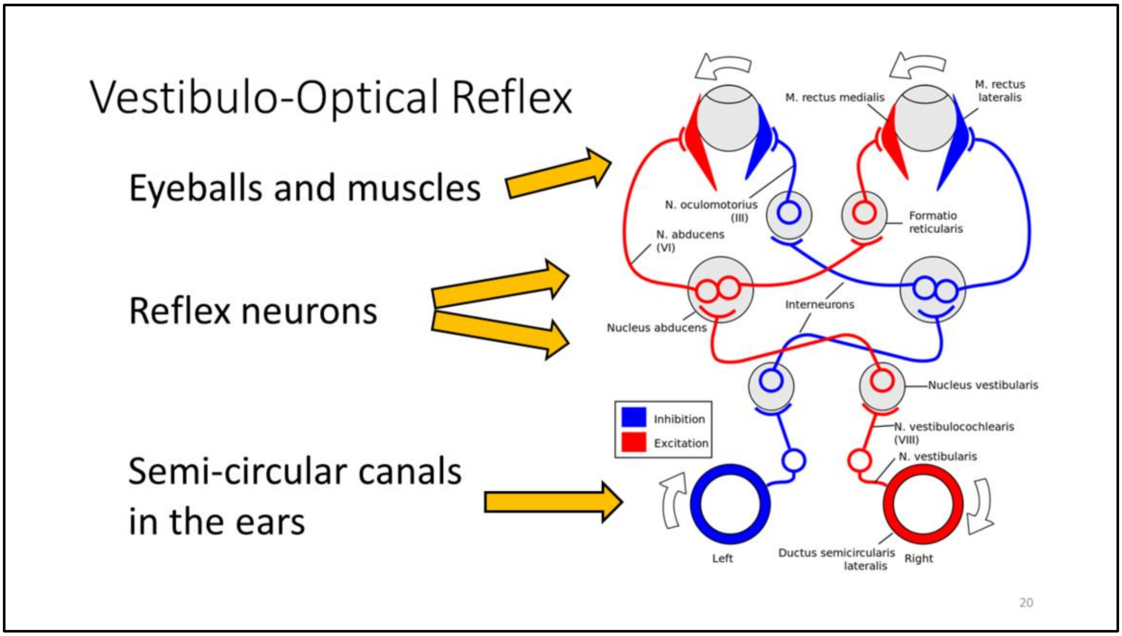


Vestibulo-Optical Reflex

Eyeballs and muscles

Semi-circular canals in the ears

19

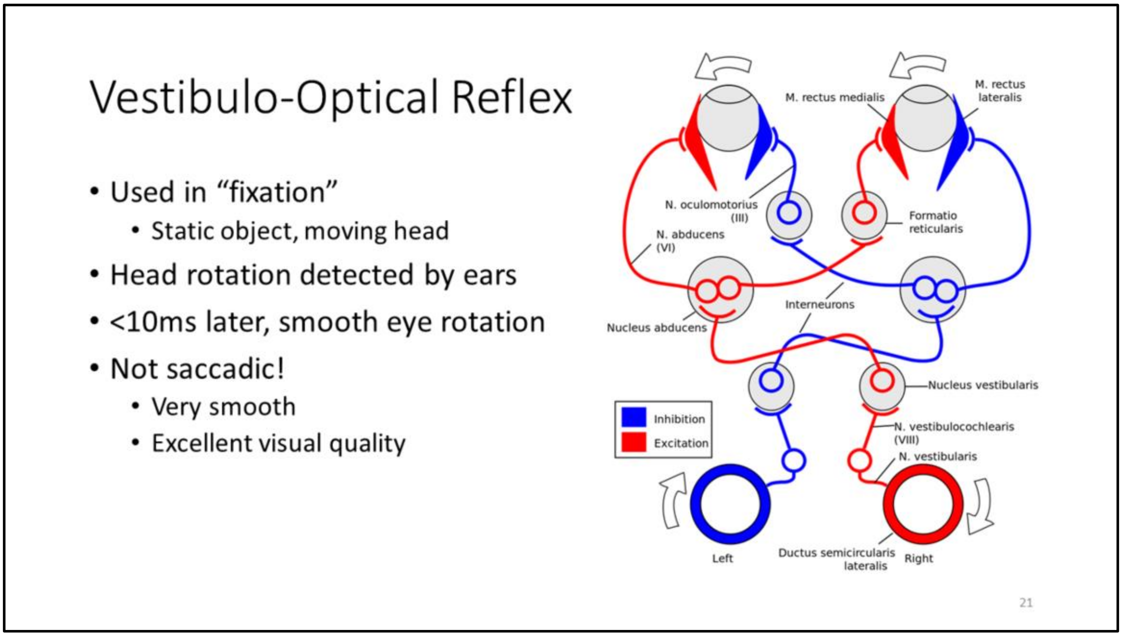


Vestibulo-Optical Reflex

Eyeballs and muscles

Reflex neurons

Semi-circular canals in the ears



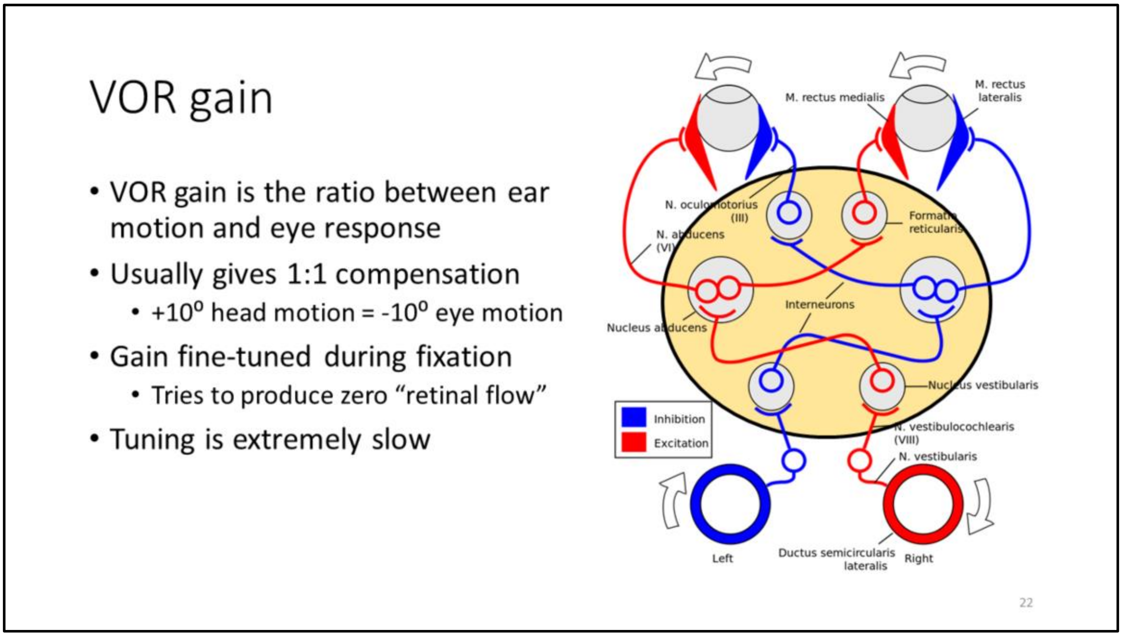
Vestibulo-Optical Reflex

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

Fixation is any case where your eyes are looking at a fixed object but the head is moving. Because we're humans balancing our heads on a tall, unstable bipedal platform, this happens almost all the time, so this system needs to be really good, or we wouldn't be able to see anything clearly. It's so good you can read a stationary book while aggressively turning your head in all sorts of directions.

This is a very fundamental reflex. You cannot turn it off, and it happens even in the dark or when your eyes are closed – if you rotate your head, your eyes WILL counter-rotate.

Interestingly, not only is this not motion not saccadic, the VOR reflex actually causes saccades. If you try to follow a moving object with your head, or sweep your eyes smoothly around a room, the VOR reflex is constantly trying to keep your eyes pointed at a fixed point in space. So to keep following the moving object, or scanning the room, your eyes have to jump "past" the VOR reflex to catch up – saccades! If the object moves slower than 30 degrees/second, we can do non-saccadic smooth pursuit – but that's a really slow speed!



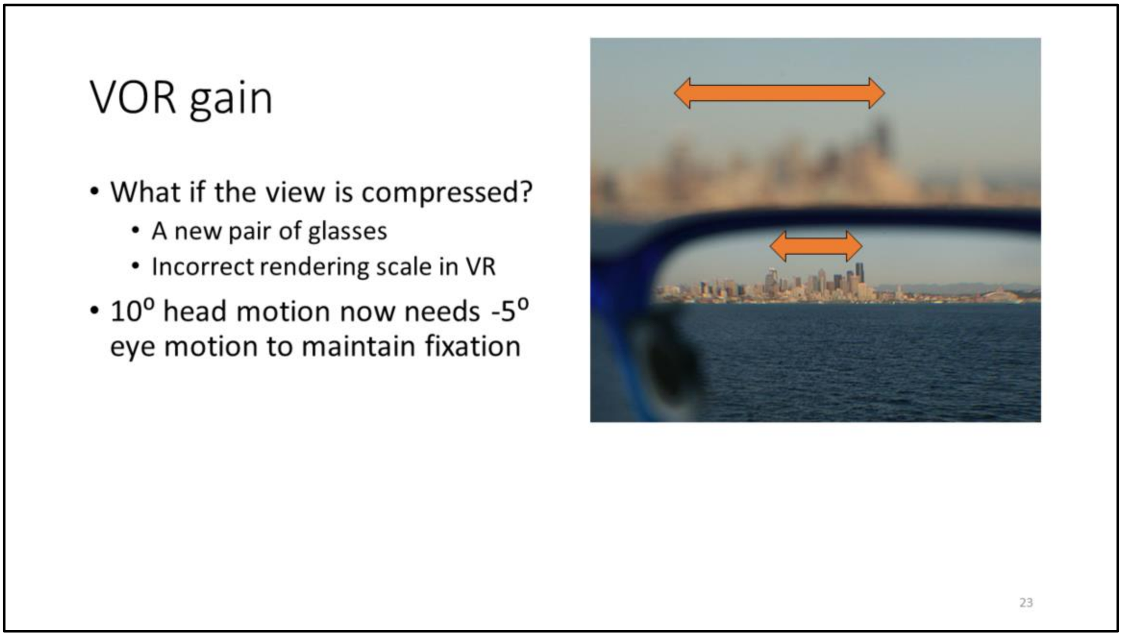
VOR gain

- VOR gain is the ratio between ear motion and eye response
- Usually gives 1:1 compensation
  - $+10^0$ head motion = -10^0 eye motion
- Gain fine-tuned during fixation
  - Tries to produce zero “retinal flow”
- Tuning is extremely slow

22

This is a simplification – the gain is not quite as hardwired as shown here. For example, people who wear glasses will usually have two settings of VOR gain – one with glasses and one without – and they can switch between them very quickly. However, both those modes take long-term training, and when they get a new set of glasses with a slightly different VOR gain, it takes 1-2 weeks to retrain one of those “slots”.

In fact this is one of the dangers of developing in VR. If you get the settings wrong and don’t correct them, in a few weeks you can retrain one of the “slots” of your VOR gain to cope with the error. And then you simply WON’T see the error any more. In fact if you correct the error, now it will look wrong! This is why we as developers need to be very aware that “looks fine to me” is not only not useful, it’s often actively misleading data.

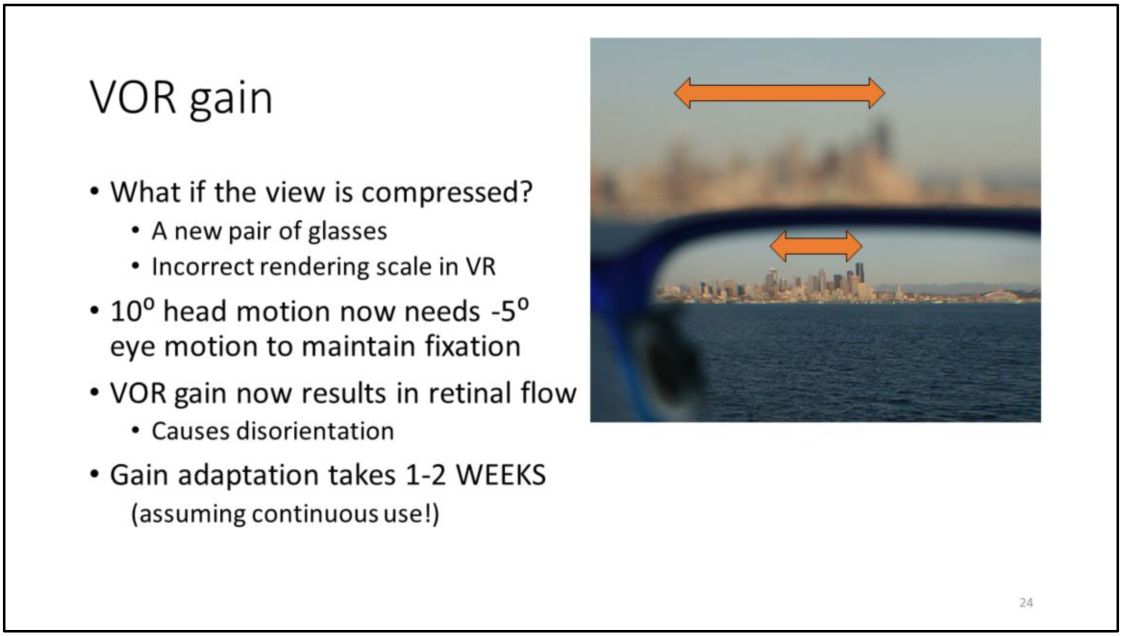


VOR gain

- What if the view is compressed?
  - A new pair of glasses
  - Incorrect rendering scale in VR
- 10° head motion now needs -5° eye motion to maintain fixation

23

Image off Wikipedia again – thanks! A halving in size means these are really strong glasses, and you can see the chromatic aberration at the right hand edge.

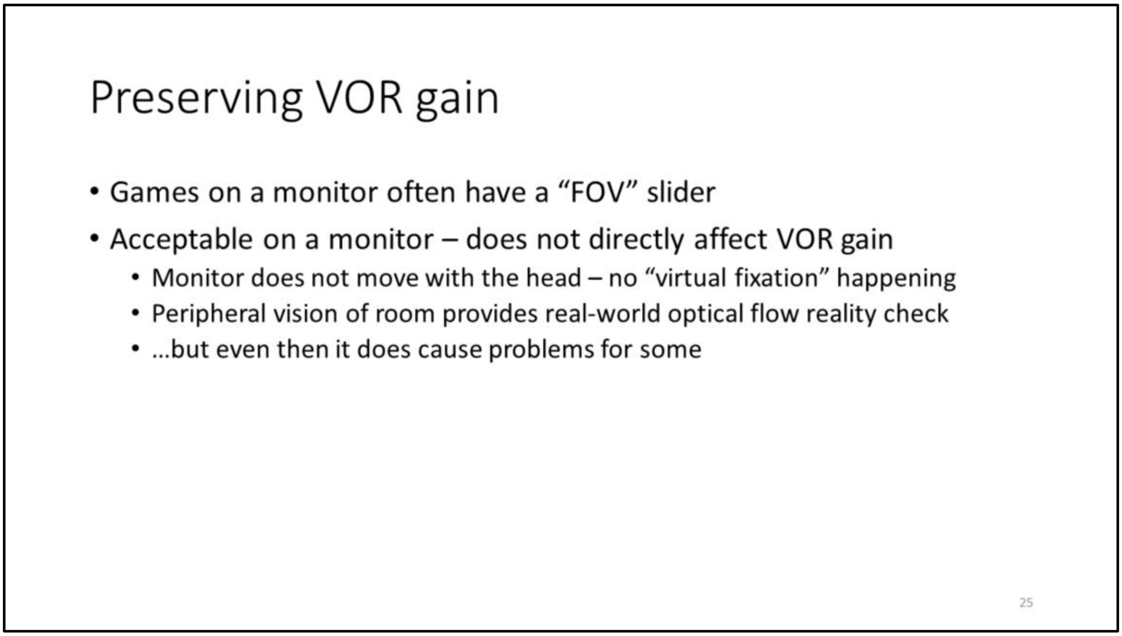


VOR gain

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

24

Abbreviated for time. There are lots of other training feedback mechanisms for VOR gain that don’t require fixation. But the principle still stands – the brain is expecting a certain amount of retinal flow for any given head/eye motion, and the image on the HMD needs to match that to within a few percent to avoid simulator sickness and other problems.

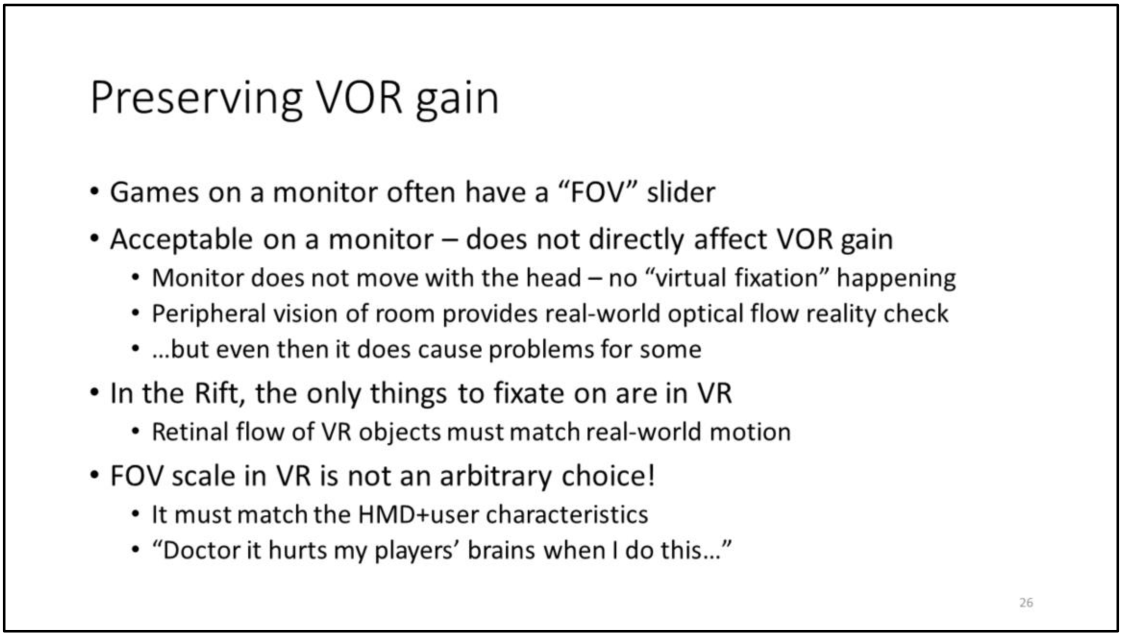


Preserving VOR gain

- 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

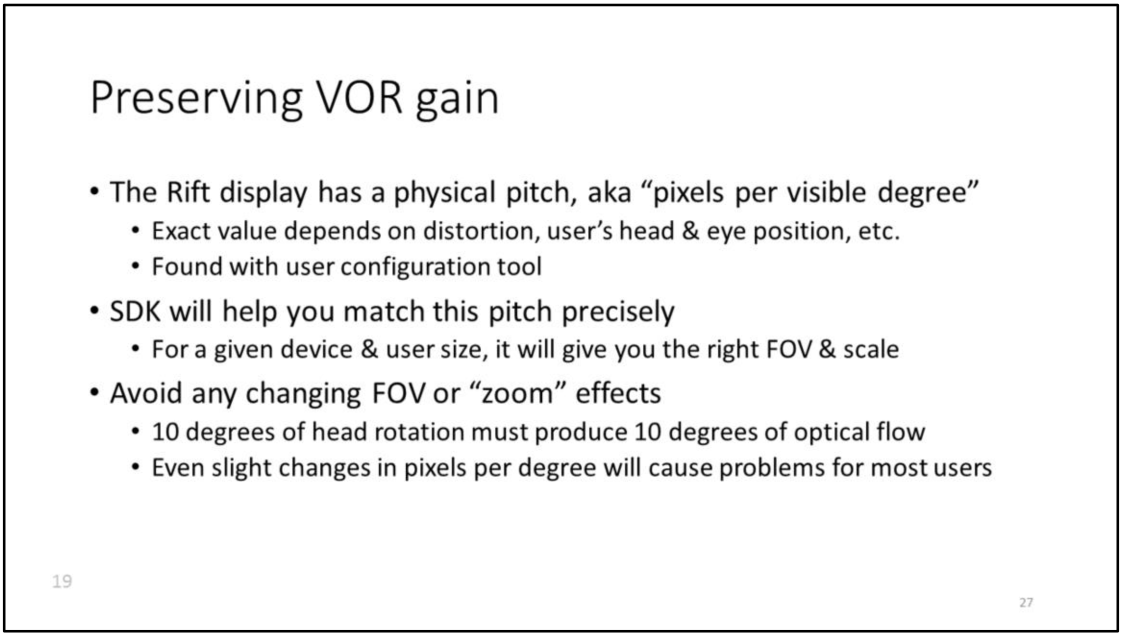
Michael Abrash is an example of someone sensitive to this problem even on a monitor. He has spoken on his blog about how developing Quake was a problem because it made him queasy if he played for too long. He was an incredibly useful guinea-pig in the Valve VR group – if Michael could play your game for more than a few minutes, you knew you were doing something right!

Another interesting side-effect I have noticed is that some people playing games on a monitor naturally move their heads into a spot so that the view looks right. I find myself leaning closer to the monitor with some FPSes than others, and I realized that this correlated with their FOV – the wider the FOV, the closer I’d move. I went around the office watching people playing TF2 fullscreen during a playtest, then measured where their eyes were relative to the size of the screen, and what their FOV settings were (default is 75 degrees, but it can be changed), and sure enough a significant proportion of them placed their eyes pretty much exactly at the realistic spot so that the monitor filled 75 degrees of their vision, even it that meant they hunched uncomfortably close to the screen – far closer than when they were just coding/designing/etc.



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 hurts my players’ brains when I do this…”



Preserving VOR gain

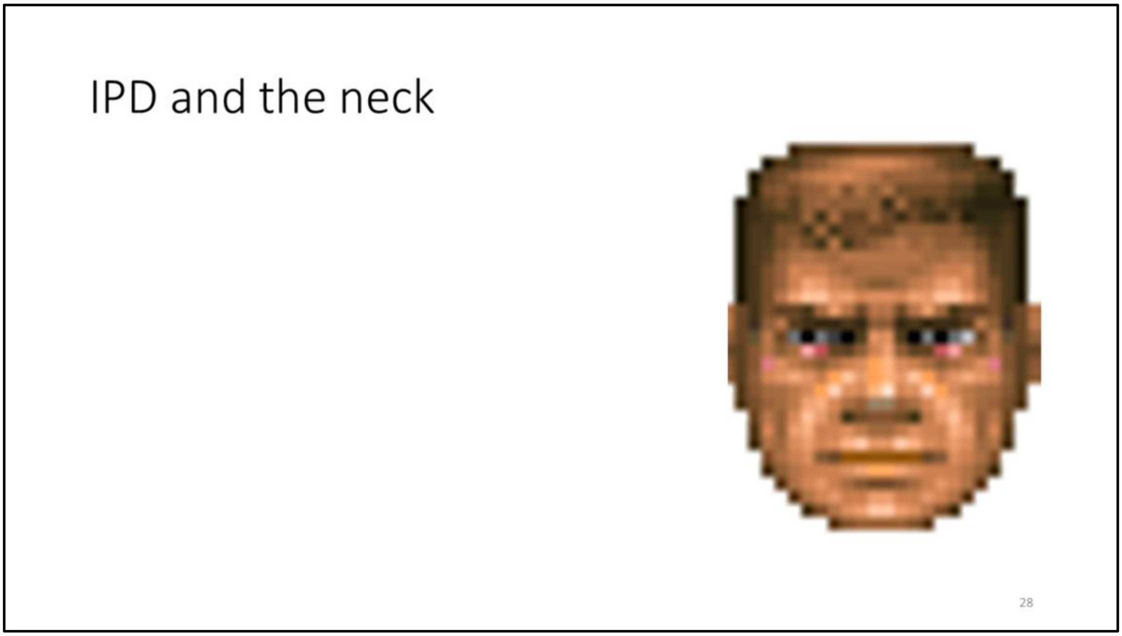
- The Rift display has a physical pitch, aka “pixels per visible degree”
  - Exact value depends on distortion, user’s head & eye position, etc.
  - Found with user configuration tool
- SDK will help you match this pitch precisely
  - For a given device & user size, it will give you the right FOV & scale
- 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

19

27

In summary, trust the SDK. We give you these values for a very good reason – don’t play with them! This isn’t actually at all difficult to get right, it’s just that on monitors we’ve become so accustomed to being able to manipulate these values freely.

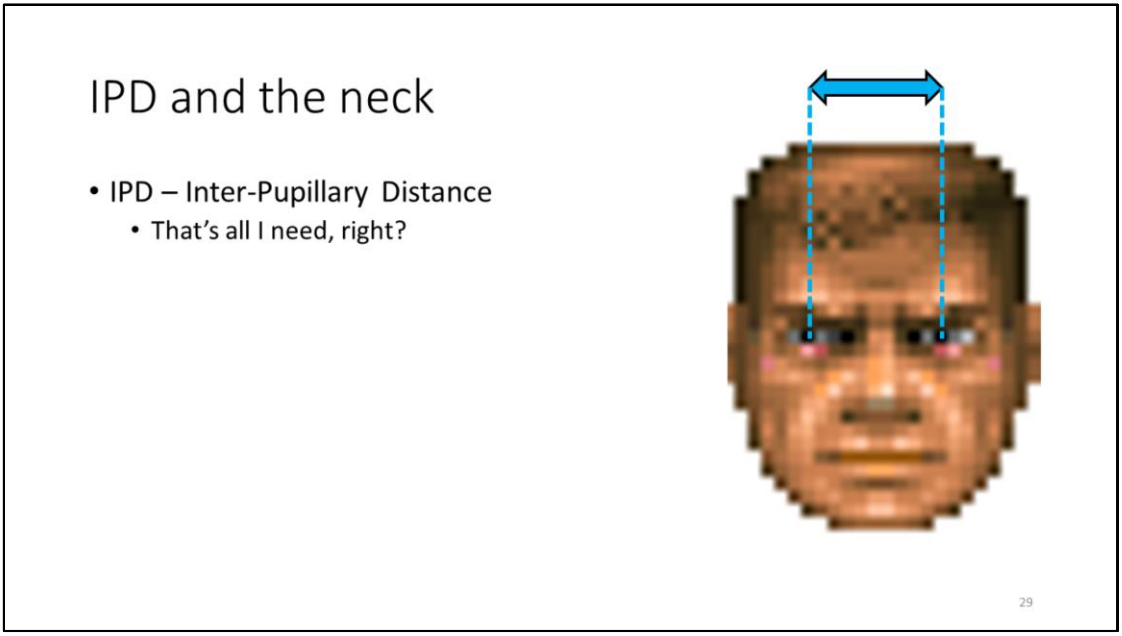
To say it again – as developers, we get used to all sorts of crazy stuff, and changing the FOV might not seem that bad to us. But I guarantee you, you’re making life unpleasant for a significant chunk of your audience. Worse still – they won’t know why. Even when you know what you’re looking for, diagnosing incorrect FOV scale is amazingly difficult.



IPD and the neck

Here is your typical VR user. Those of you under 25 may need to ask your parents who this chap is.

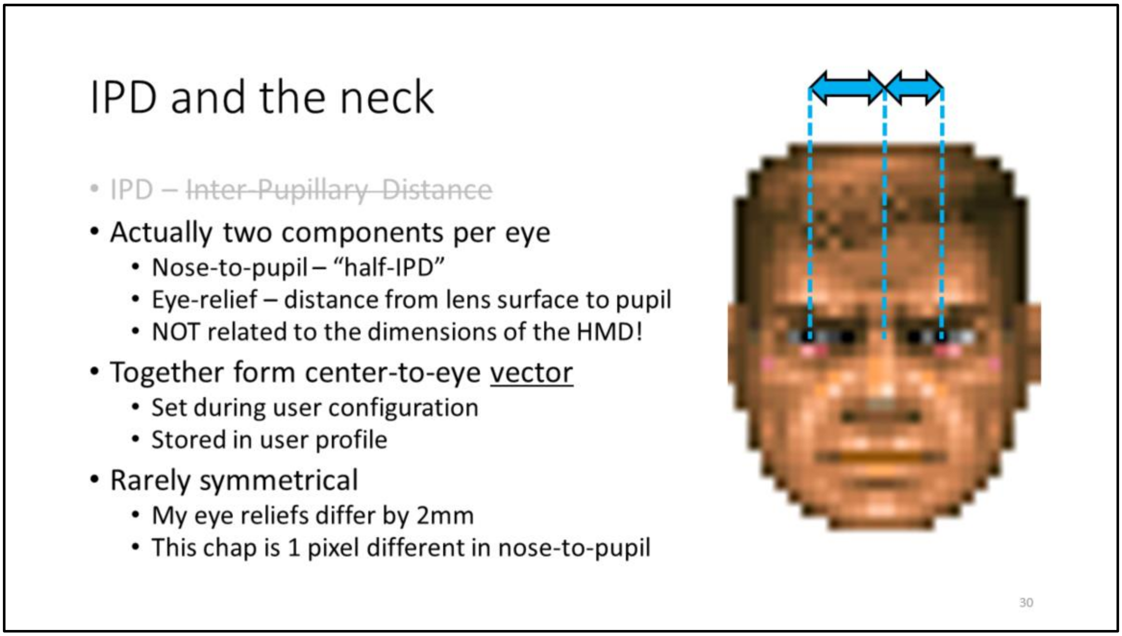
(it always makes me happy when my audience laughs at my terrible jokes. I know it’s a tech-heavy GDC talk, but that doesn’t mean it has to be boring!)



IPD and the neck

- IPD – Inter-Pupillary Distance
  - That’s all I need, right?

This is what we used to think – that all you needed was IPD and you’re good to go.

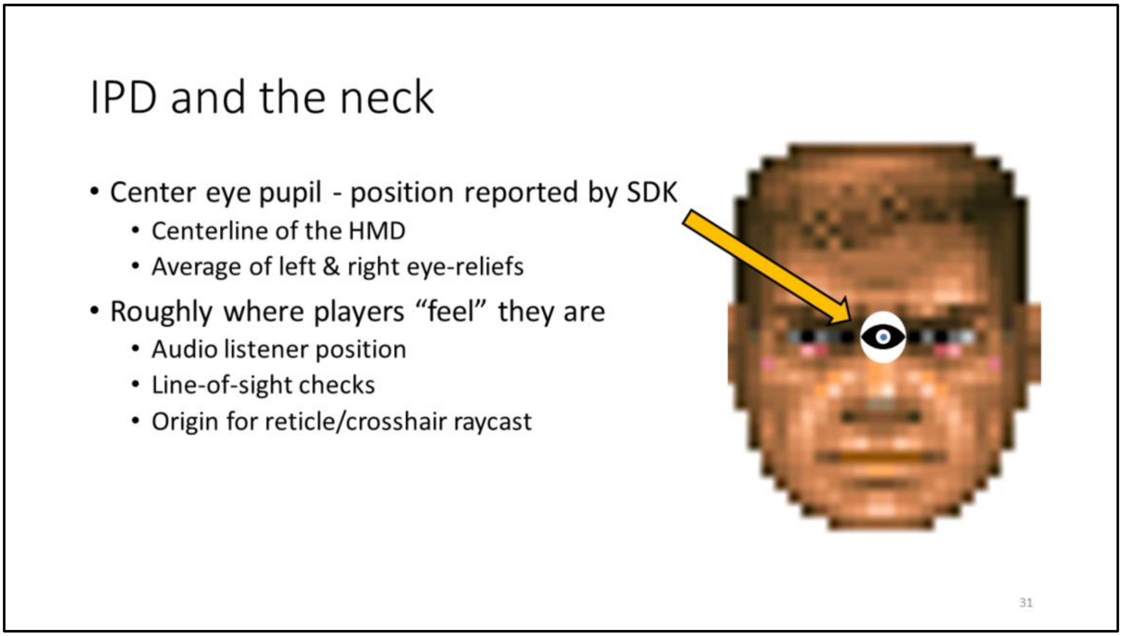


IPD and the neck

- 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
  - This chap is 1 pixel different in nose-to-pupil

30

Nose-to-eye and eye relief determine the translation of the virtual cameras for rendering. They also determine the edges of the FOV (e.g. a larger eye relief means a narrower FOV, because of the edge of the lenses), and the particular shape of the distortion function. Yup – we change the distortion function per-user, and in fact we do it per-eye as well. The SDK will take care of this for you, but just be aware that these values do all matter.



IPD and the neck

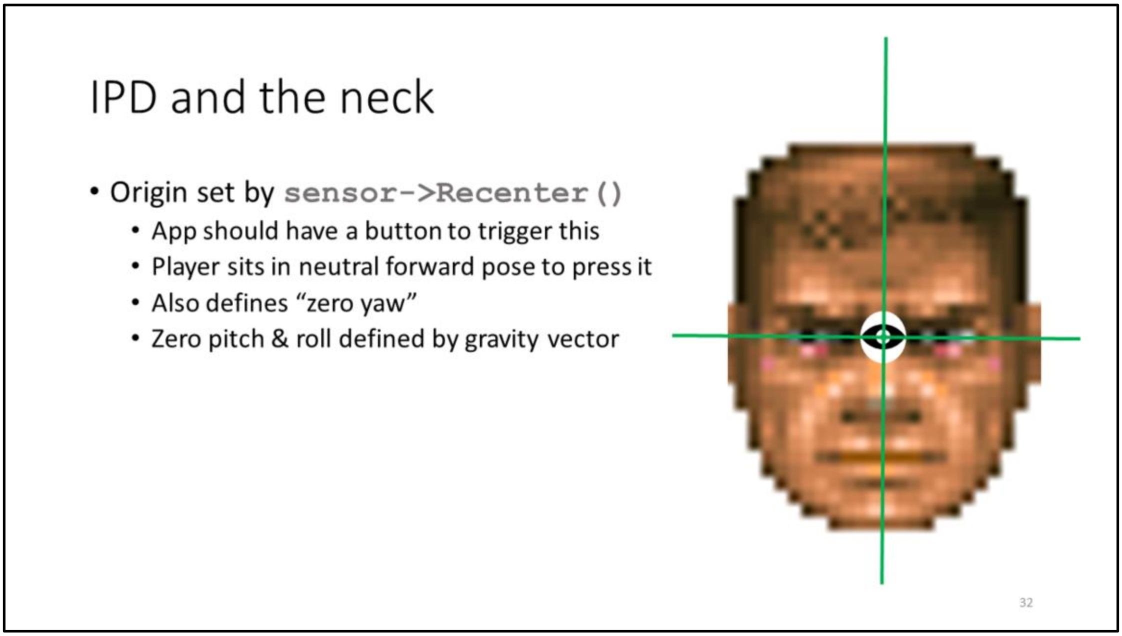
- Center eye pupil - position reported by SDK
  - Centerline of the HMD
  - Average of left & right eye-reliefs
- Roughly where players “feel” they are
  - Audio listener position
  - Line-of-sight checks
  - Origin for reticle/crosshair raycast

31

The audio question is a bit larger than I had time for in the talk, but three reasonable places to put the audio listener are:

1. Between the eyes.
2. Between the ears, i.e. 3-4 inches further back.
3. One listener per ear, i.e. 3-4 inches back, 4-6 inches outwards. Not many audio libraries support this option though.

It is a good idea to have options for both speakers and headphones in your game. The crucial difference in VR is that headphones move with the head, while speakers do not. So if the player rotates their head left, the environmental sounds that speakers emit should not change, but with headphones the sounds should all rotate right. Treat audio listener position and orientation with just as much care as you do the cameras for rendering – mismatches can quickly ruin the effect of presence.

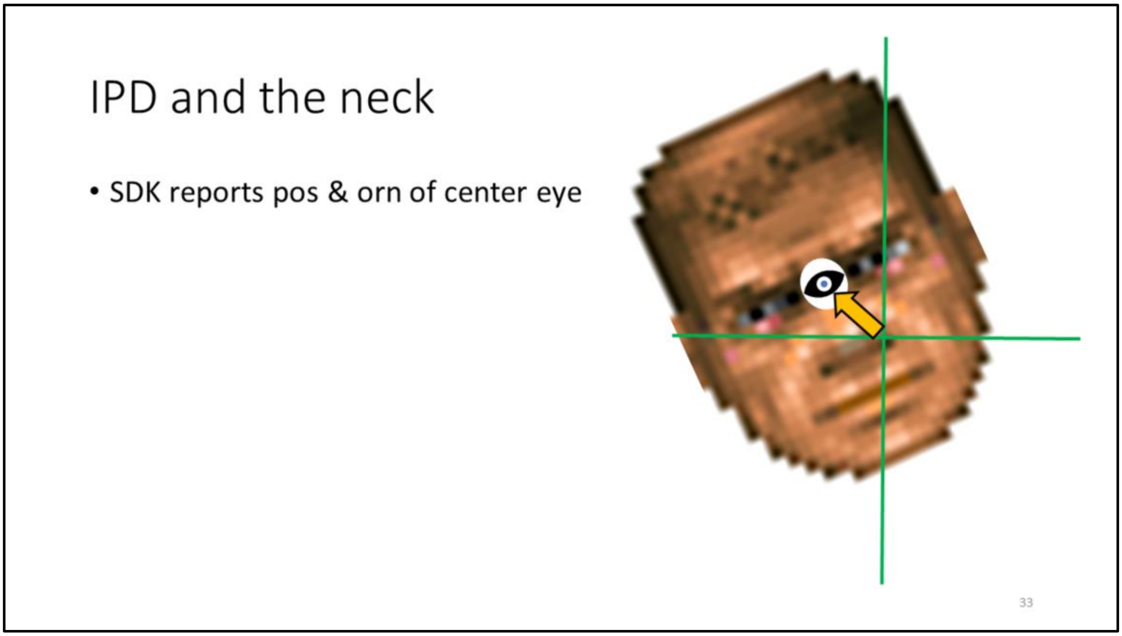


IPD and the neck

- 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

32

The user’s “center” position will change slightly every time they sit down (e.g. on a couch), and so will their “zero yaw” direction (e.g. on a swivel chair), so it’s good to have a way to reset it in-game without having to exit to the desktop, run a separate app, etc. For the same reason, you want this option on the pause menu, not just at start of day. Also, DK1 doesn’t have a camera, and some locations don’t have enough field for the magnetometer to work, so they can still drift and need resetting.

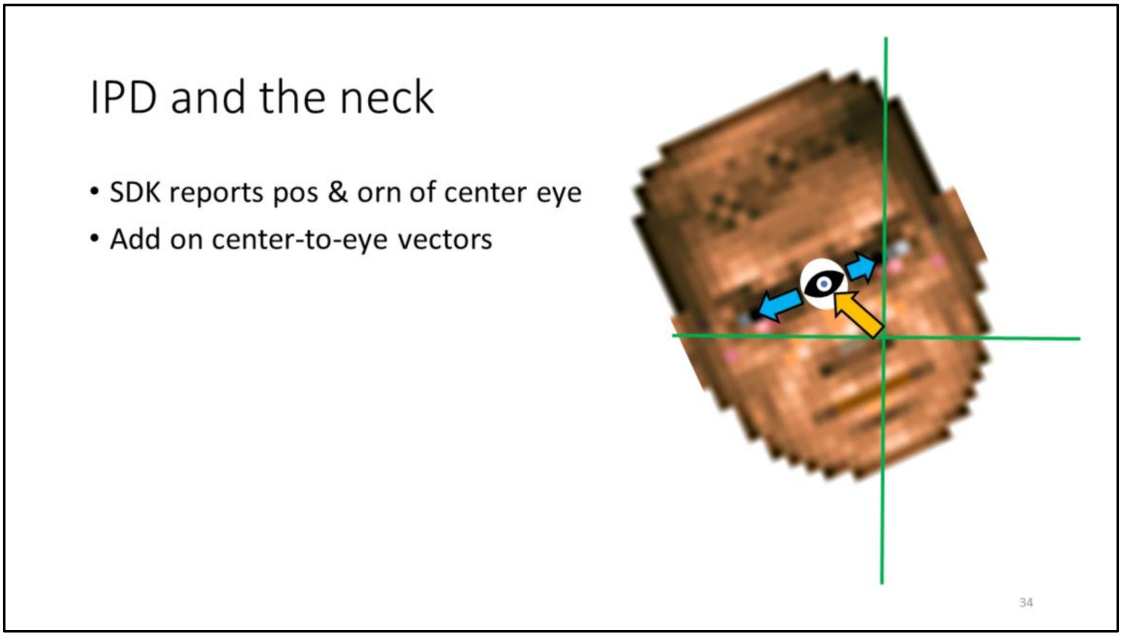


IPD and the neck

- SDK reports pos & orn of center eye

33

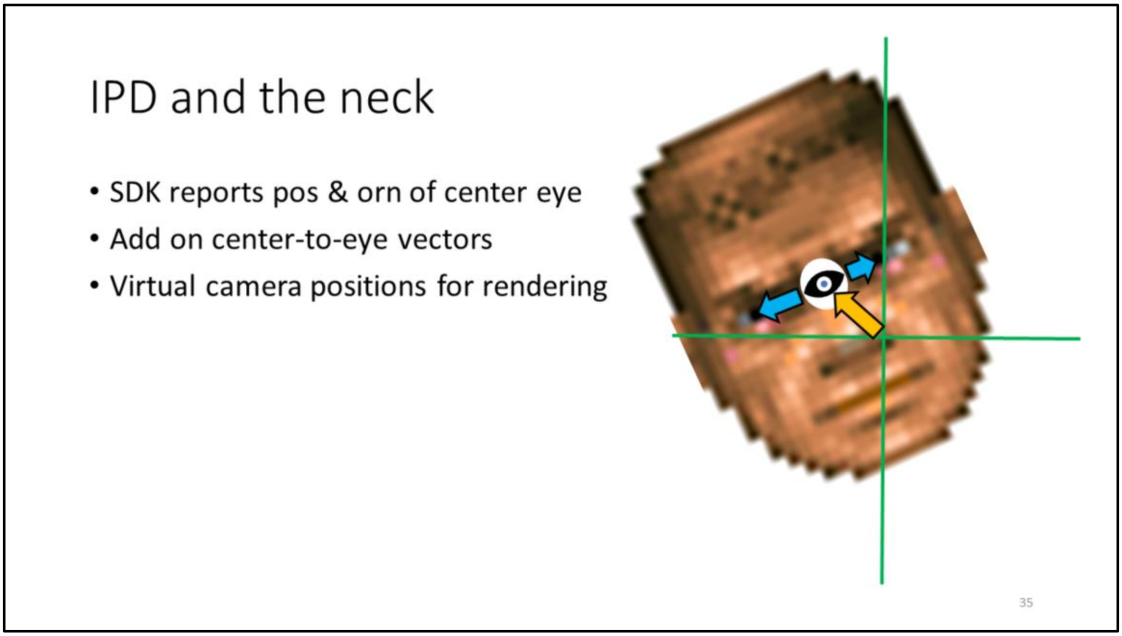
Note that even without a camera, e.g. the user has DK1, or using DK2 without a camera, the SDK has a kinetic “head model” that does a reasonable guess at the position of the head as long as the user doesn’t move their shoulders. The API is identical whichever mode is being used, and unless you explicitly query for it, you don’t really know or care which mode is in use at any one time.



IPD and the neck

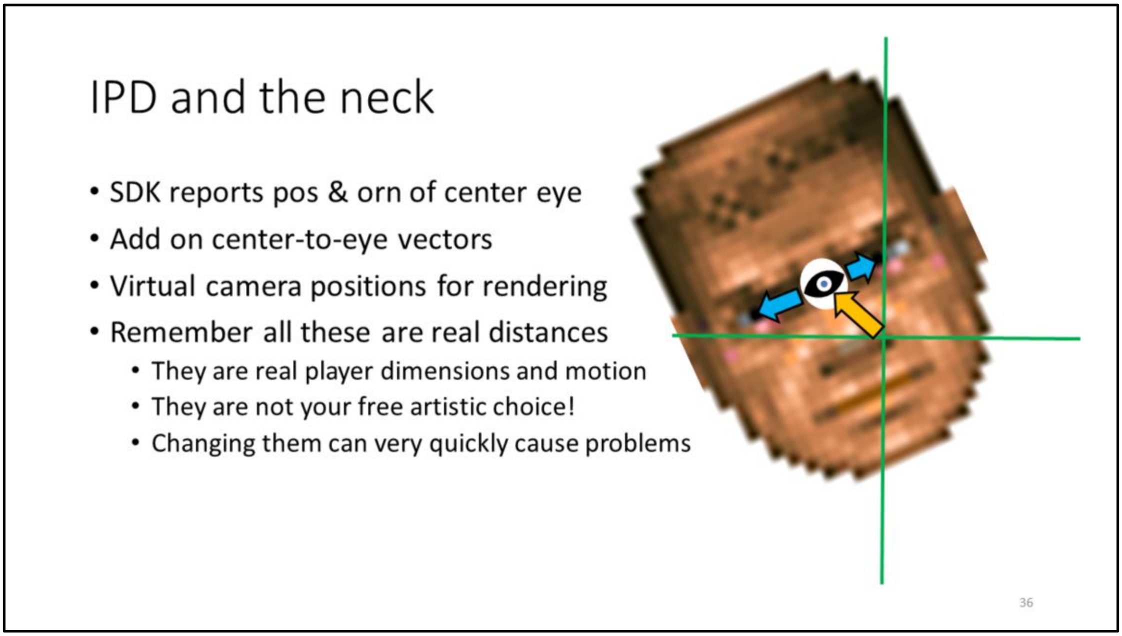
- SDK reports pos & orn of center eye
- Add on center-to-eye vectors

34



IPD and the neck

- SDK reports pos & orn of center eye
- Add on center-to-eye vectors
- Virtual camera positions for rendering

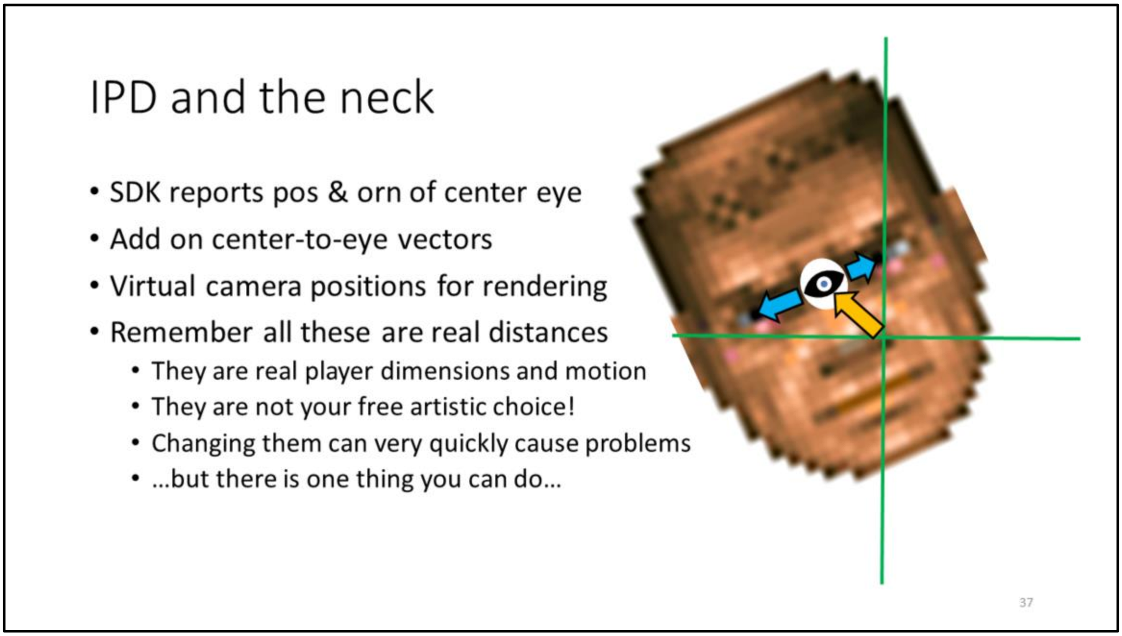


IPD and the neck

- 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

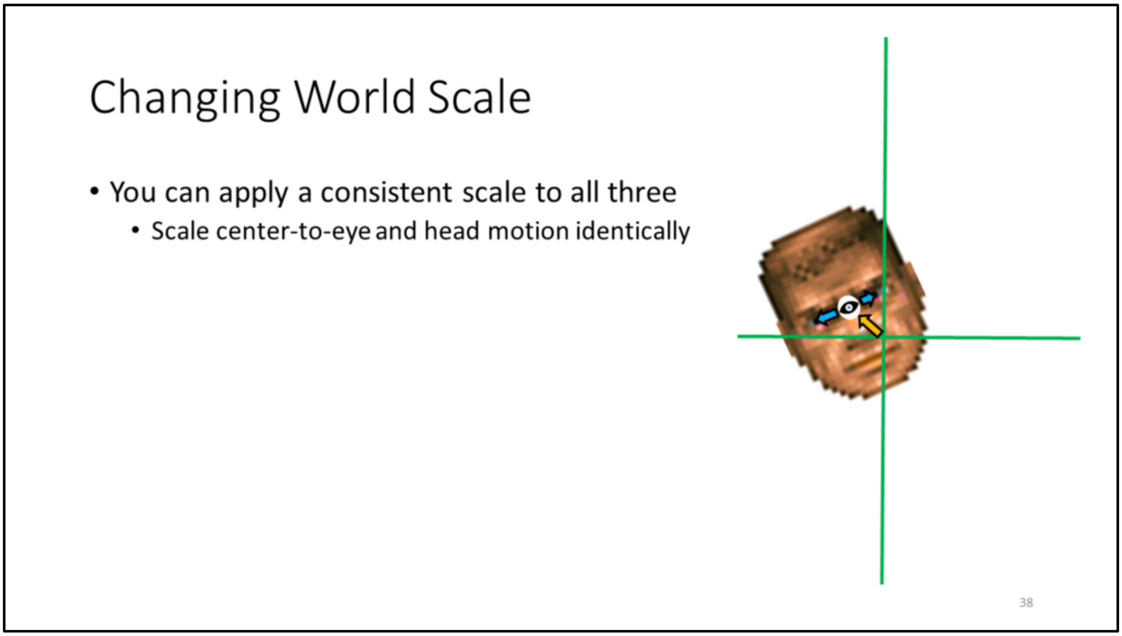
36

None of this is difficult, and the SDK feeds you all these numbers. This is just me banging on about how important it is to do the right thing and not cut corners or use fudge factors.



IPD and the neck

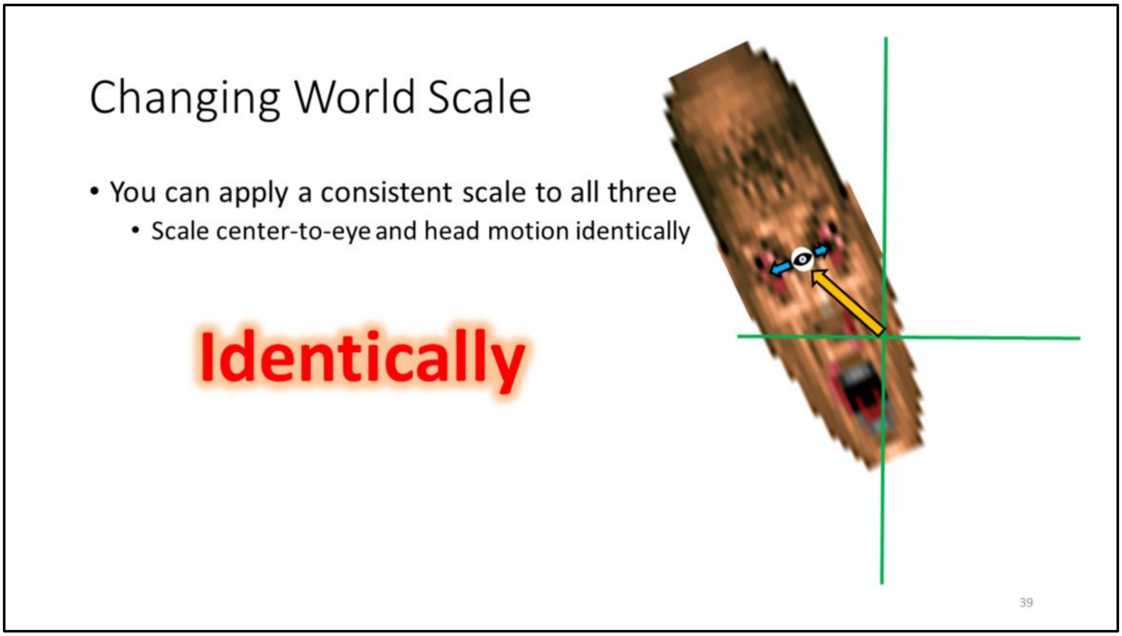
- SDK reports pos & orn of center eye
- Add on center-to-eye vectors
- Virtual camera positions for rendering
- Remember all these are real distances
  - They are real player dimensions and motion
  - They are not your free artistic choice!
  - Changing them can very quickly cause problems
  - …but there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



Changing World Scale

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

“Three” = world-origin-to-center; center-to-left-eye; center-to-right-eye.



Changing World Scale

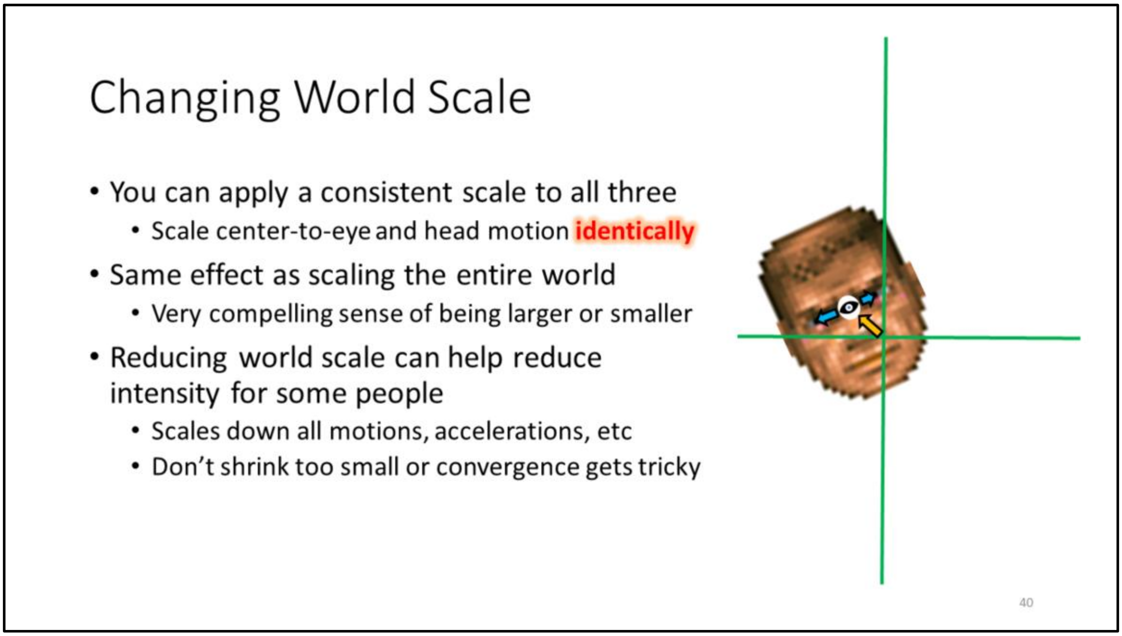
- You can apply a consistent scale to all three
  - Scale center-to-eye and head motion identically

Identically

39

Here the motion of the player’s head has not been scaled, but their center-to-eye vectors have been. This will cause simulator sickness – in fact we believe it’s one of the leading causes.

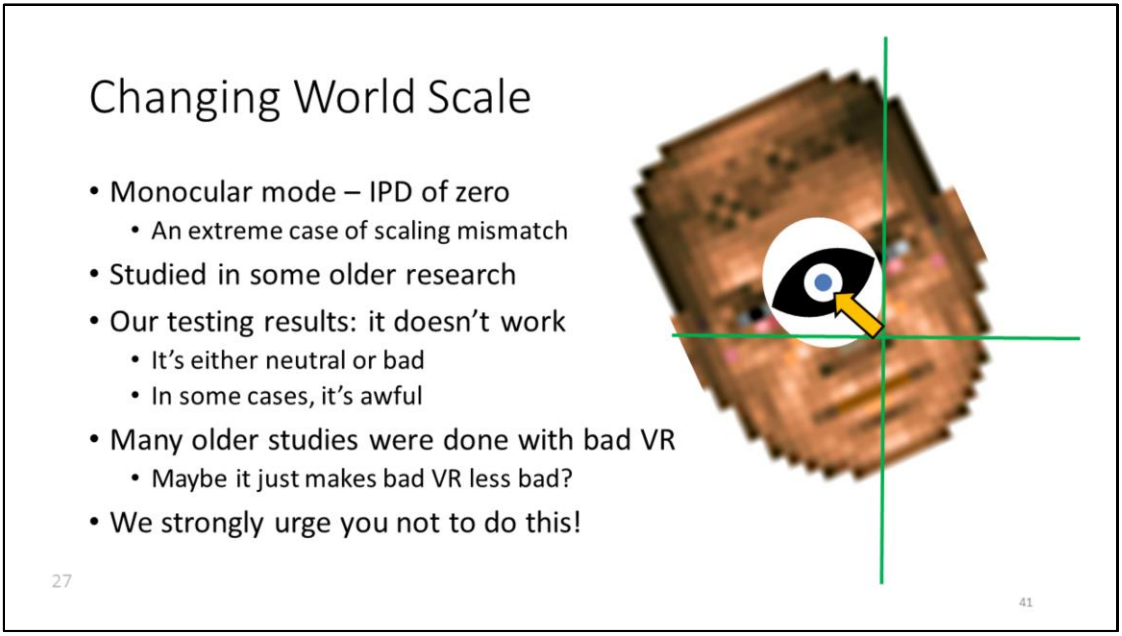
It is very easy to forget to scale all three, or to have the center-to-eye vectors not match the players actual distances. It is also extremely difficult to diagnose – our conscious brains are incredibly bad at spotting that the center-to-eye vectors are wrong, even by multiple centimeters. But our lizard-brains are still sensitive, and errors of as little as 5% can make people ill. And they won’t know why! VR is annoying that way.



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

“Convergence gets tricky” – what I mean is this: A normal distance to talk to someone is around 3 feet away. If you shrink the world by a factor of 12x, it’s really amusing, because everyone looks like Barbie and Ken. The problem is they now stand 3 inches away to talk to you – this is almost impossible to point your eyes at! But moderate scales like 2x or 3x are usually fine, and reduce intensity significantly for people who are sensitive.



Changing World Scale

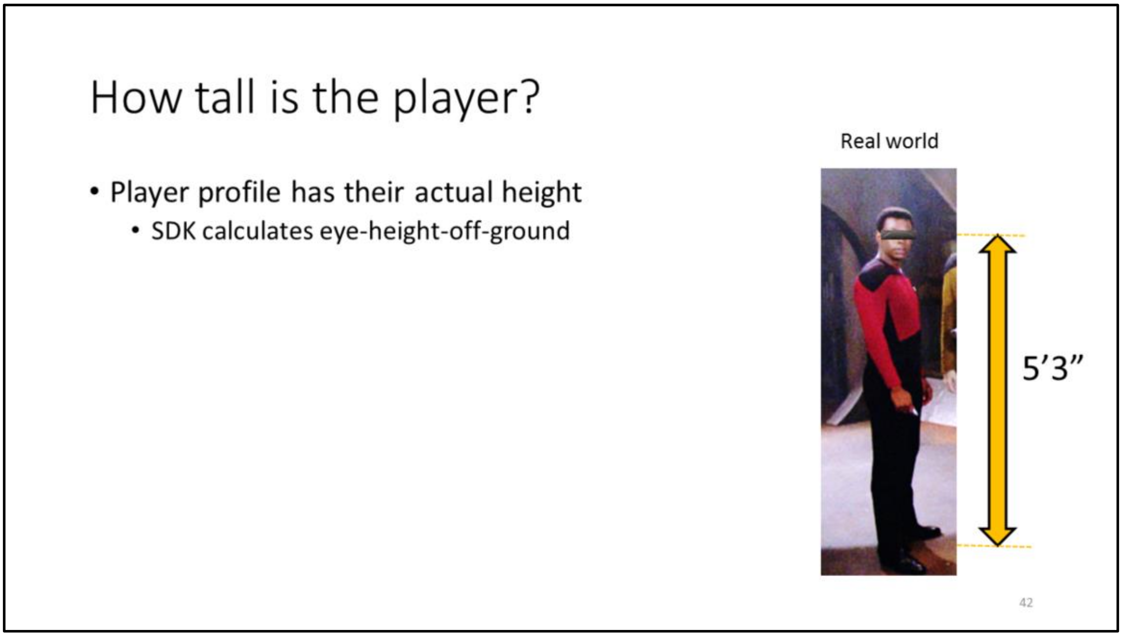
- Monocular mode – IPD of zero
  - An extreme case of scaling mismatch
- Studied in some older research
- Our testing results: it doesn't work
  - It's either neutral or bad
  - In some cases, it's awful
- Many older studies were done with bad VR
  - Maybe it just makes bad VR less bad?
- We strongly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

27

41

This is an example of where older research can be misleading. We're not sure why they recommended monocular mode, but we believe it's because their HW was what we'd call "terrible" by today's standards – latencies of 100s of milliseconds, limited FOV and calibration, etc. Not their fault – it was the 80s, and the tech just wasn't ready. But still bad. In that scenario, it may be that monocular made it slightly less bad.

What we do know is that with modern tech, good calibration, and latencies below 40ms, monocular never seems to help, and in some people (e.g. myself) it is significantly worse than proper stereo. However, this does still need more thorough research, and it may turn out that different people simply prefer different things. We'll keep the Best Practices Guide up to date on our findings.



How tall is the player?

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

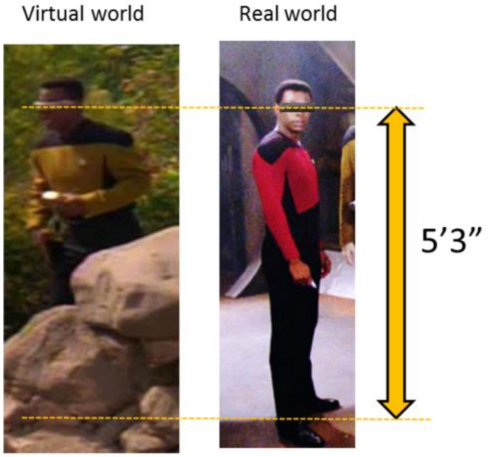
Real world
5’3”

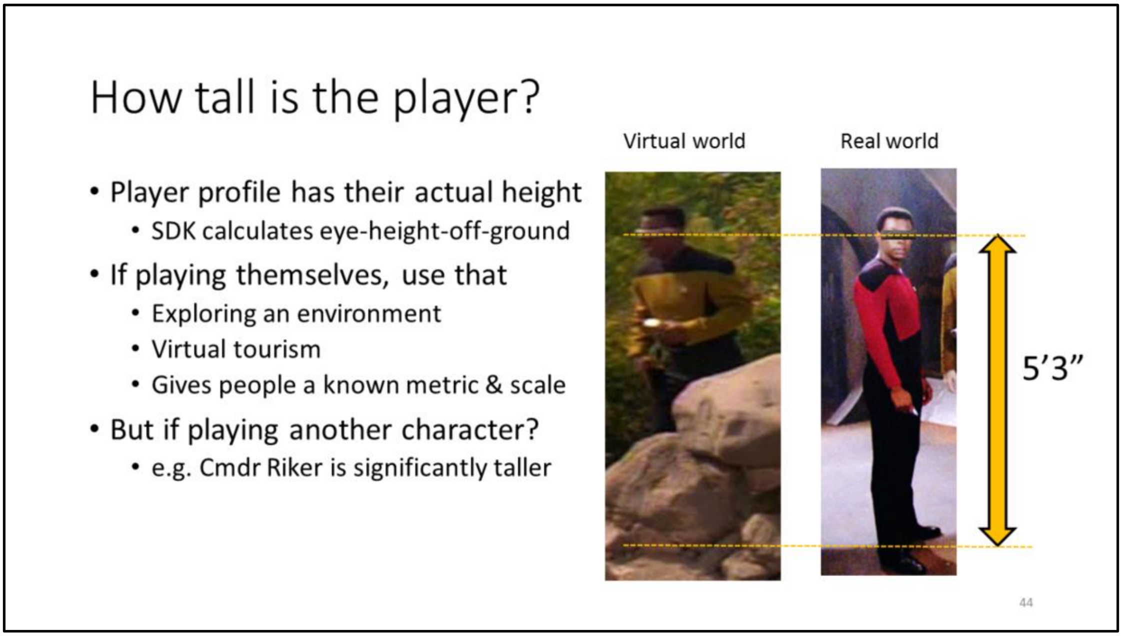
Again, our younger viewers may need to ask their parents who Geordi La Forge is. He is shown here wearing the Oculus Rift Mk1701

Geordie is 5’7”, so I estimated his eye height at 5’3”. The SDK will do this for your players as well. Fortunately (as you will see), it doesn’t need to be all that precise.

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





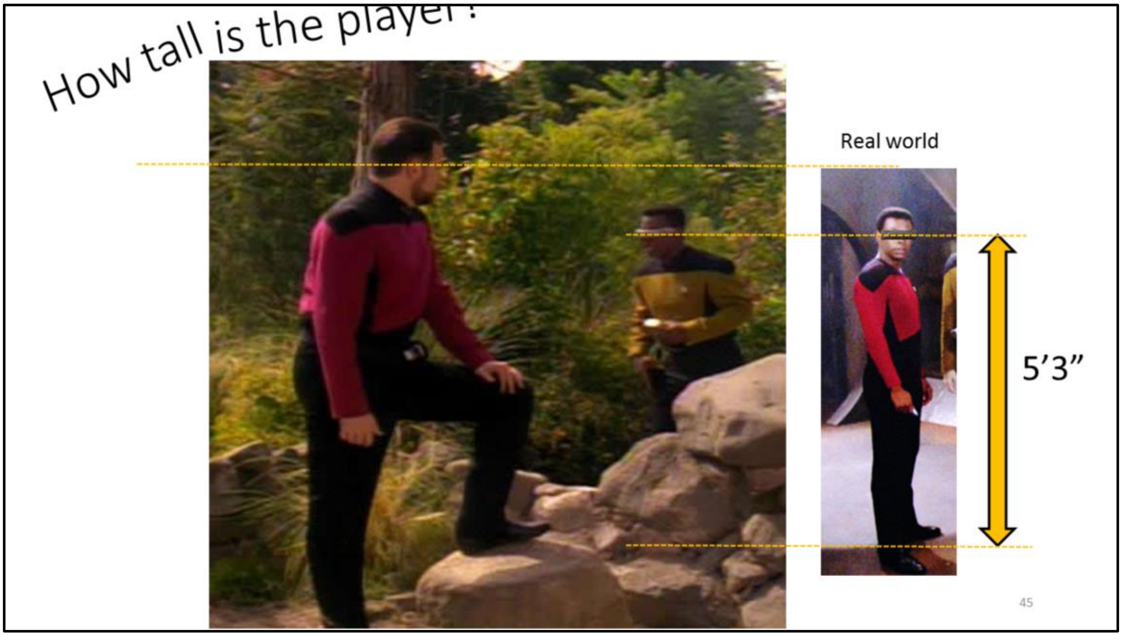
How tall is the player?

- 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. Cmdr Riker is significantly taller

Virtual world
Real world
5’3”

44

I chose Geordi because of the VISOR of course, but it turns out he is one of the shorter members of the crew. The tallest? Well, if you ignore Worf’s forehead (which makes his head taller, but doesn’t affect his eye height), it’s Riker…



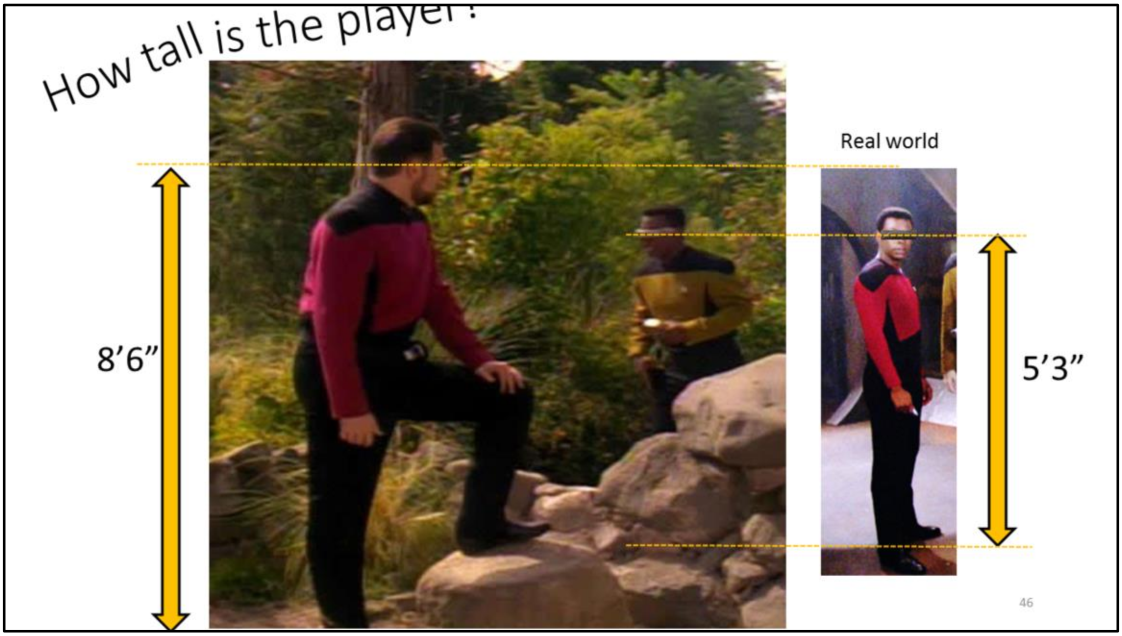
How tall is the player?

Real world

5’3”

…and the camera never lies - Riker really is over nine feet tall.

(Thankyou Google Image – so perfect, I just couldn’t resist using it)



How tall is the player?

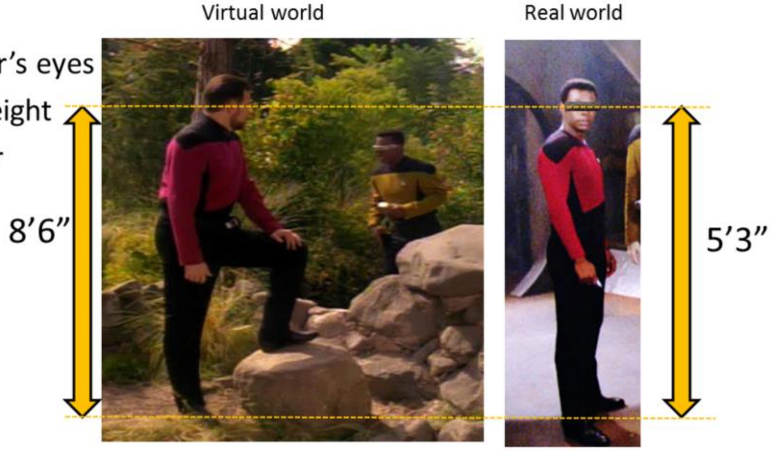
8’6”

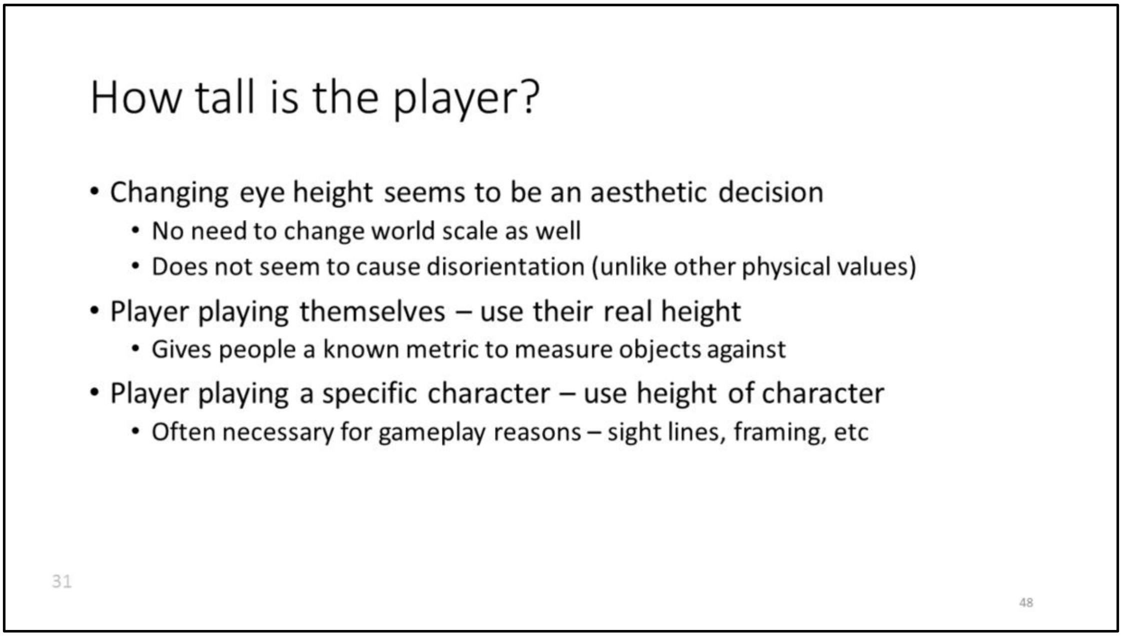
Real world

5’3”

How tall is the player?

- Move the player’s eyes to match the height of the character





How tall is the player?

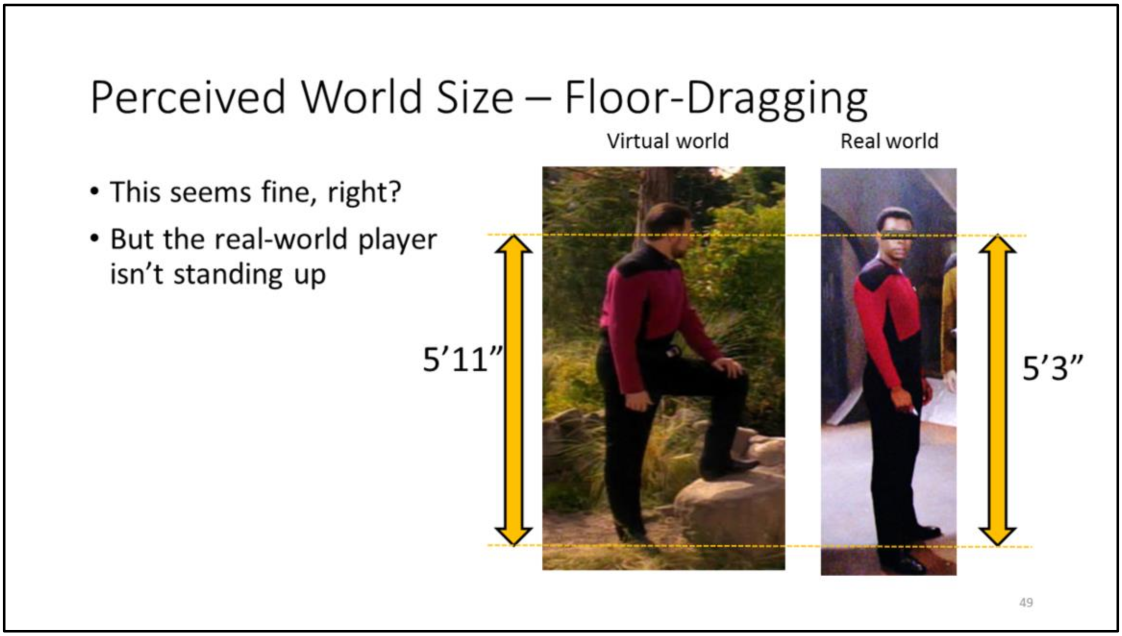
- Changing eye height seems to be an aesthetic decision
  - No need to change world scale as well
  - Does not seem to cause disorientation (unlike other physical values)
- Player playing themselves – use their real height
  - Gives people a known metric to measure objects against
- Player playing a specific character – use height of character
  - Often necessary for gameplay reasons – sight lines, framing, etc

31

48

It seems to be totally fine to just move the eye height up, but leave the world scale as it is.

This is fortunate because changing the eye height of a specific character in a game (James Bond, Gordon Freeman, etc) can cause all sorts of pain for level designers.



Perceived World Size – Floor-Dragging

- This seems fine, right?
- But the real-world player isn't standing up

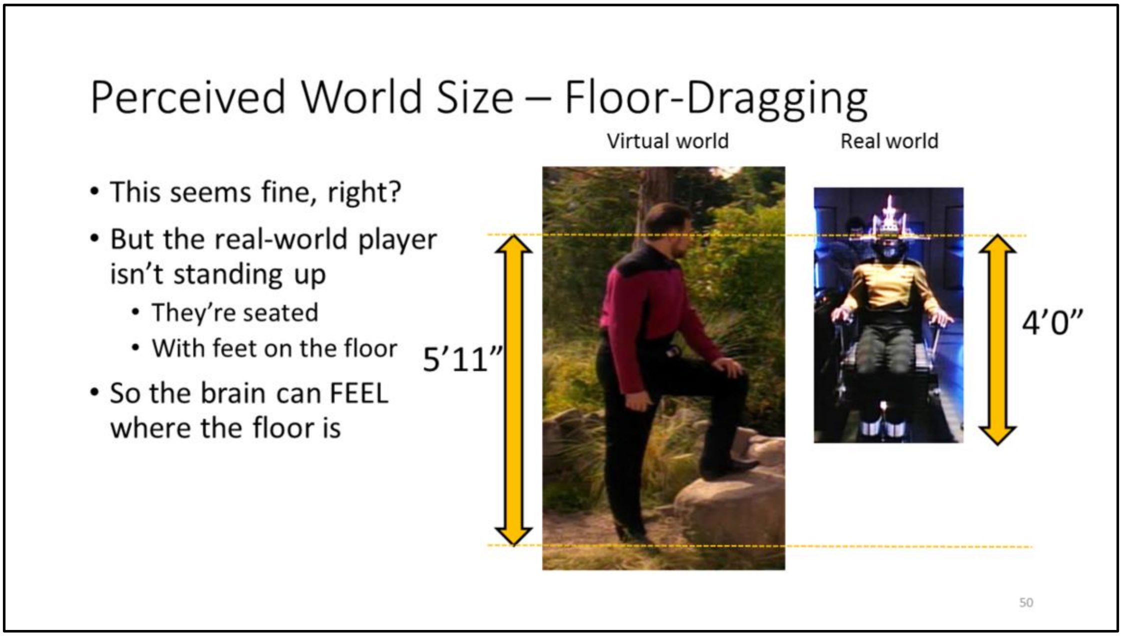
Virtual world

Real world

5'11"

5'3"

…except for the problem of Floor-Dragging. I'm not sure if this has a formal name, but "floor-dragging" is what I call it.

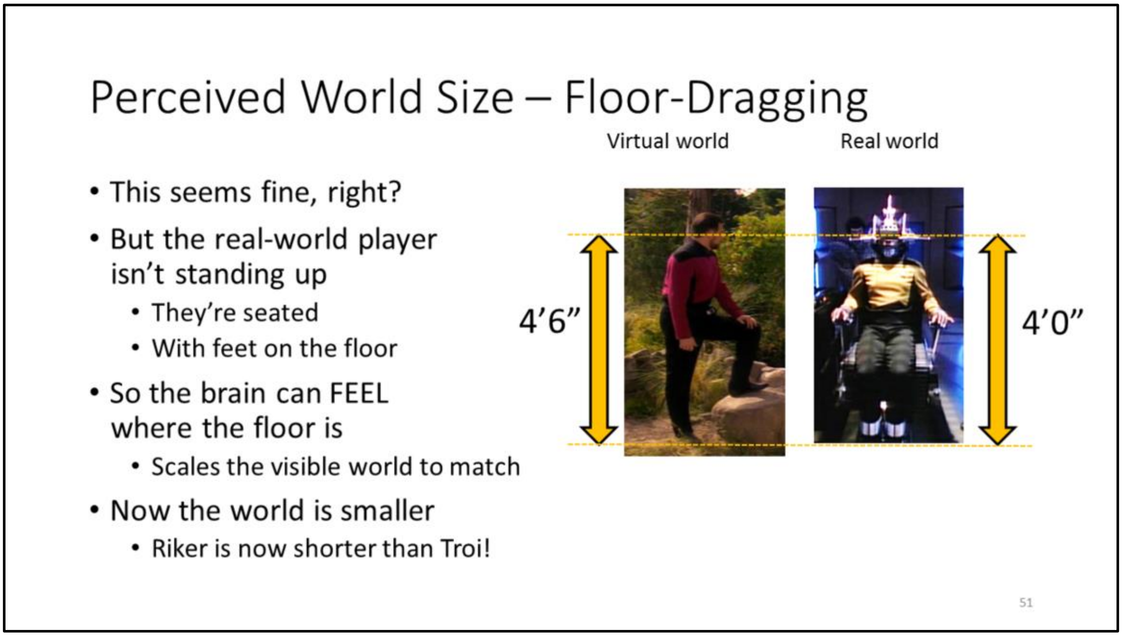


Perceived World Size – Floor-Dragging

- This seems fine, 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

(finding pictures of Geordi sitting down where you can see his feet is very difficult – in this one he's actually being tortured by Romulans)

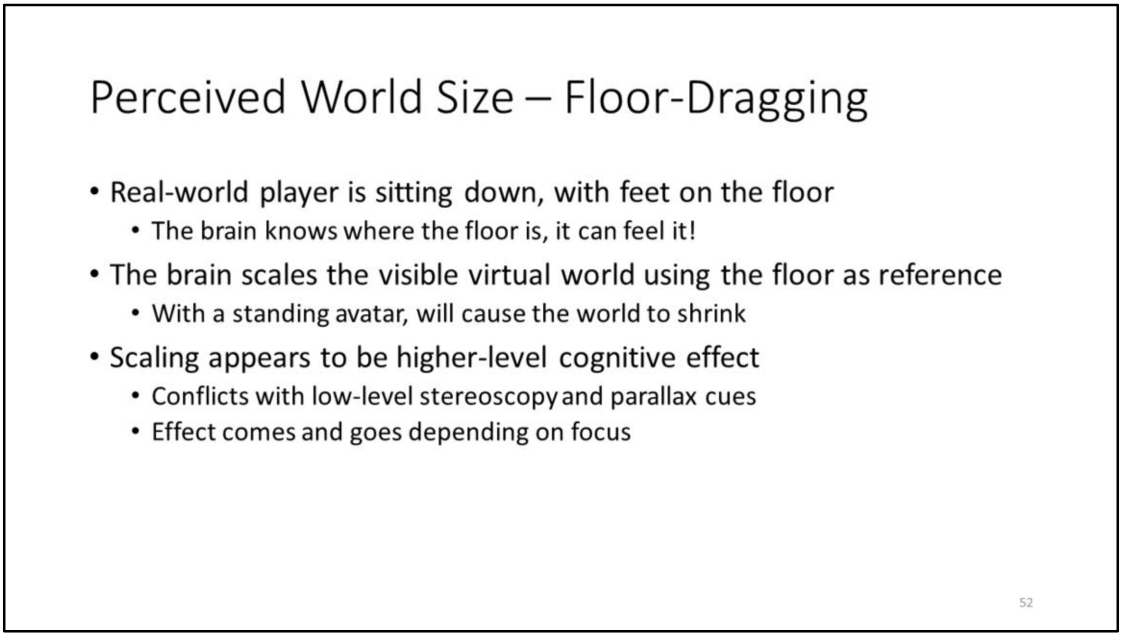
The exact eye-height of the player when sitting down will of course depend on the chair, but a normal office chair puts eyes at around 4 feet off the floor.



Perceived World Size – Floor-Dragging

- This seems fine, 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
  - Scales the visible world to match
- Now the world is smaller
  - Riker is now shorter than Troi!

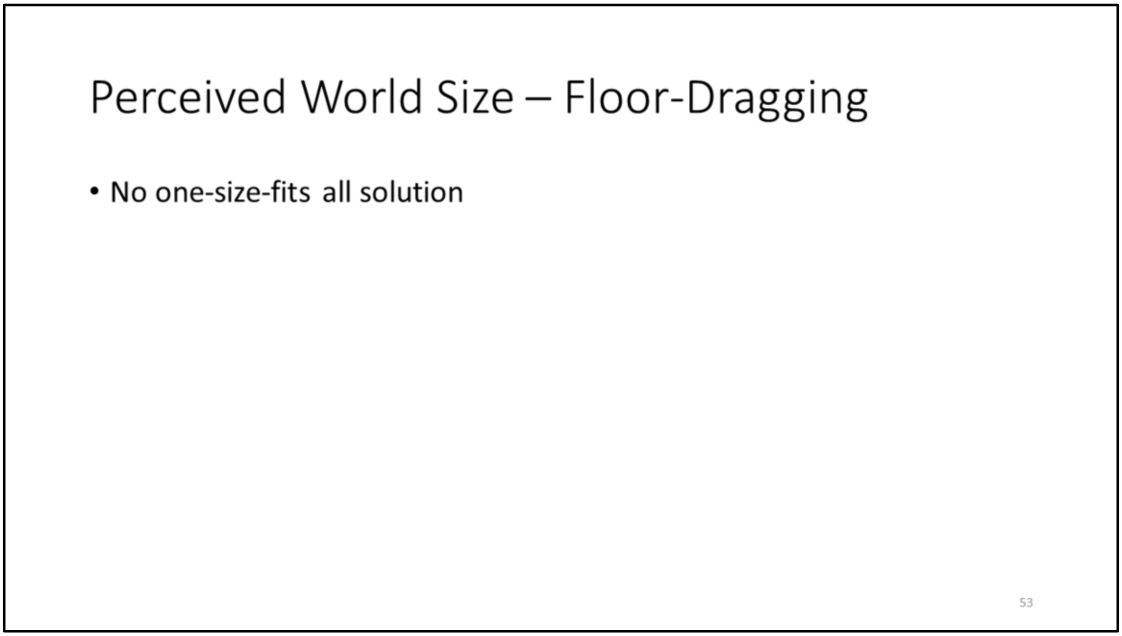
This effect took a while to find. We couldn’t figure out why we felt so short. I spent a week looking for math errors. Then someone said “oh, you fixed it!” – err… no I didn’t. Then they realized they were standing up. Sit down, effect comes back, stand up, it goes away. Yet again, the weakest link in our perfect VR simulation is the wetware!



Perceived World Size – Floor-Dragging

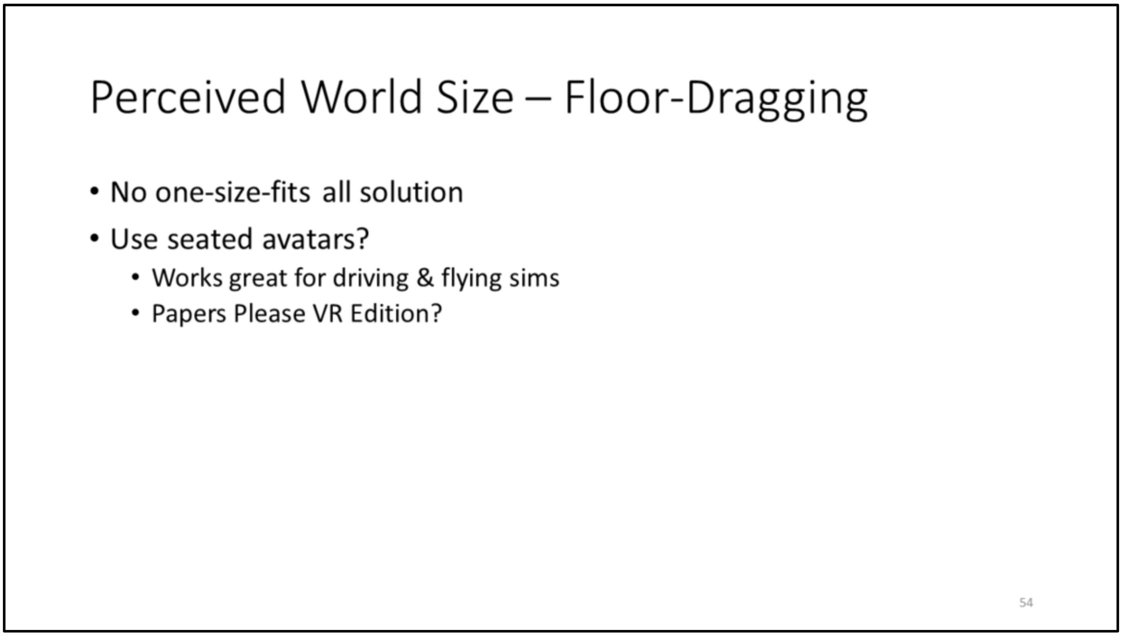
- Real-world player is sitting down, with feet on the floor
  - The brain knows where the floor is, it can feel it!
- The brain scales the visible virtual world using the floor as reference
  - With a standing avatar, will cause the world to shrink
- Scaling appears to be higher-level cognitive effect
  - Conflicts with low-level stereoscopy and parallax cues
  - Effect comes and goes depending on focus

The good news is this effect does not seem to cause any disorientation or sickness, and this is one reason we think it is a higher-level effect.



Perceived World Size – Floor-Dragging

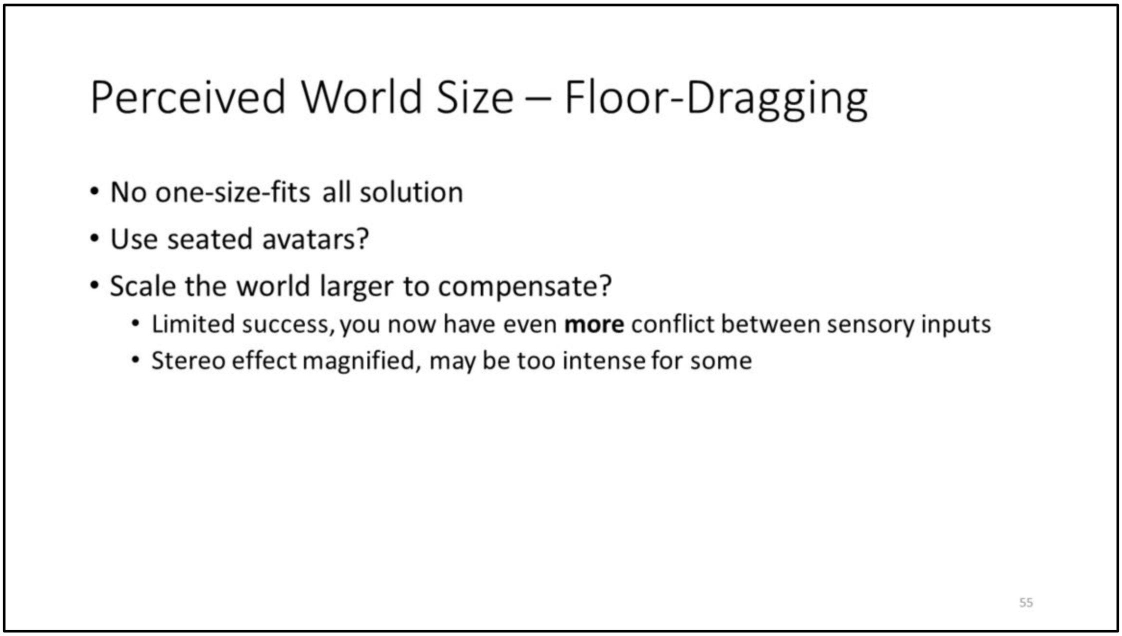
- No one-size-fits all solution



Perceived World Size – Floor-Dragging

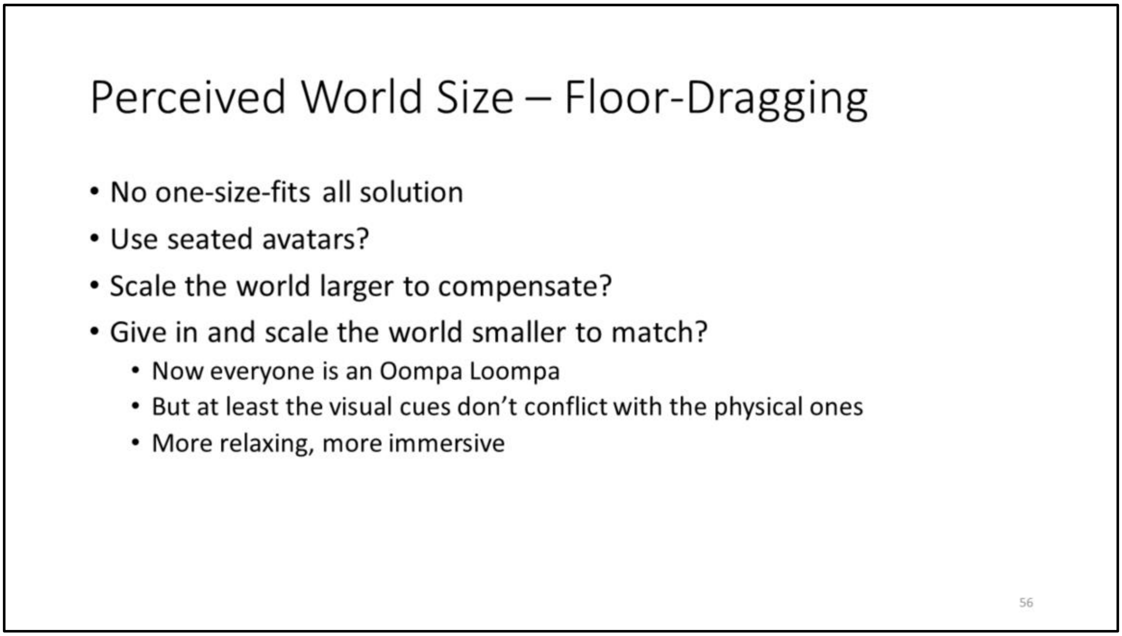
- No one-size-fits all solution
- Use seated avatars?
  - Works great for driving & flying sims
  - Papers Please VR Edition?

Lucas Pope is going to hate me…



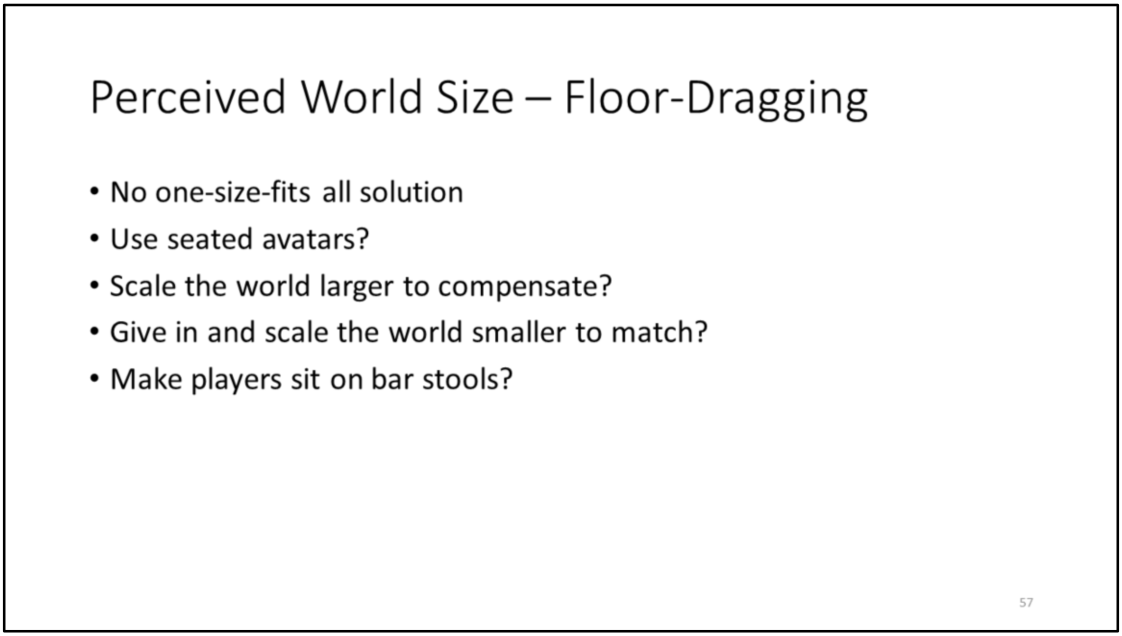
Perceived World Size – Floor-Dragging

- 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

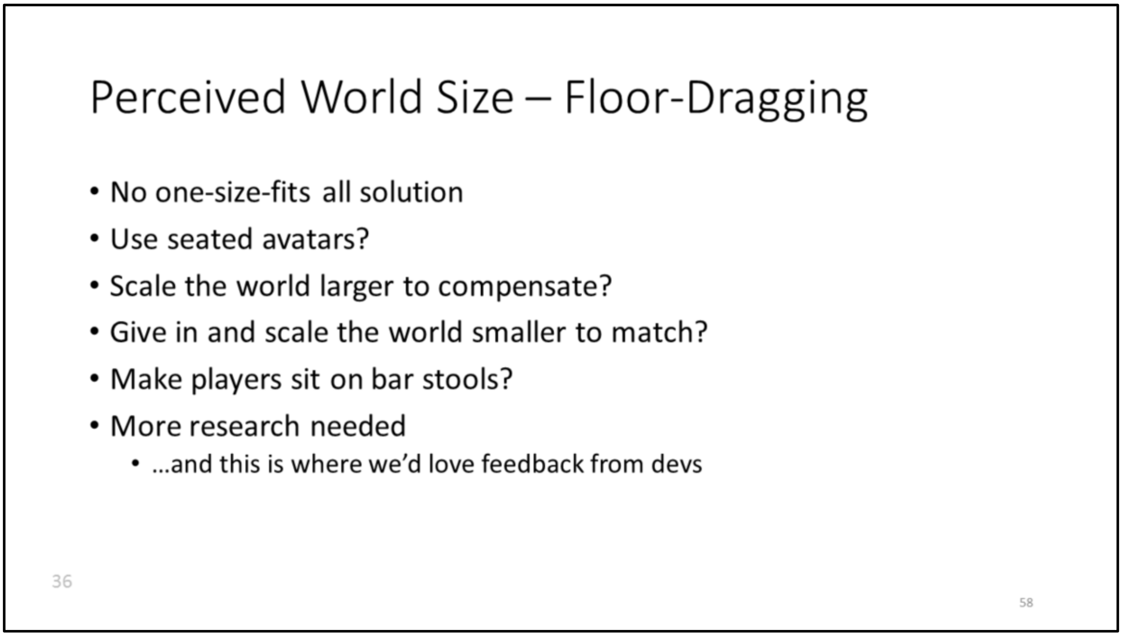
- No one-size-fits all solution
- Use seated avatars?
- Scale the world larger to compensate?
- Give in and scale the world smaller to match?
  - Now everyone is an Oompa Loompa
  - But at least the visual cues don't conflict with the physical ones
  - More relaxing, more immersive



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?

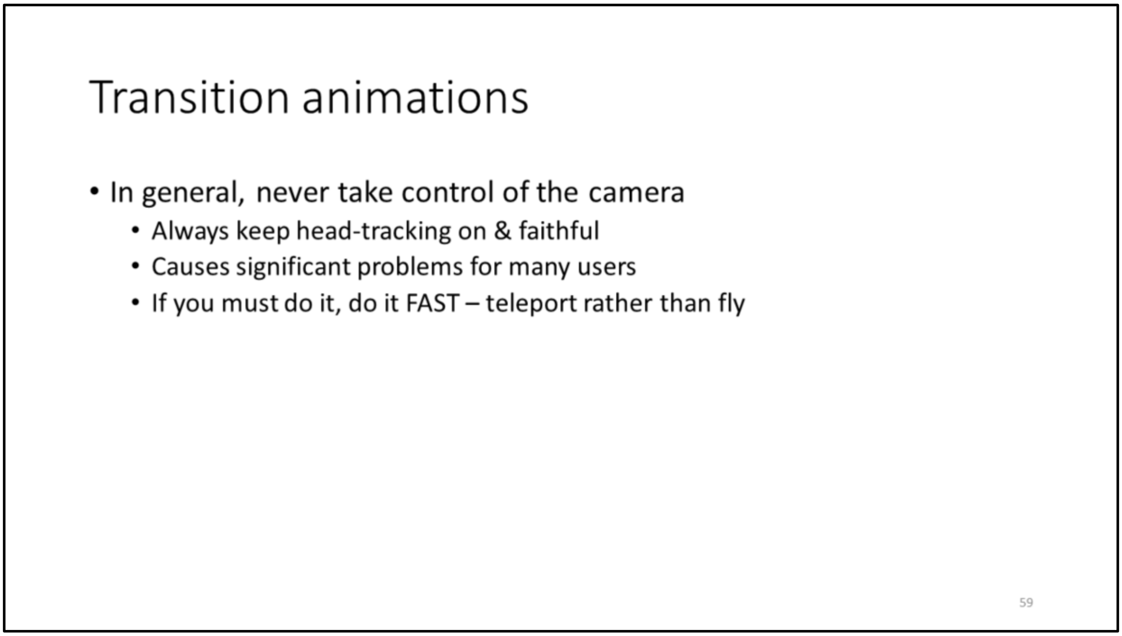
The whole point of a bar stool is that people sitting on them are at eye-level, and can talk, with people standing up. So your eyes are at the same height as when you’re standing, and the world is perceived as being at the correct scale.



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

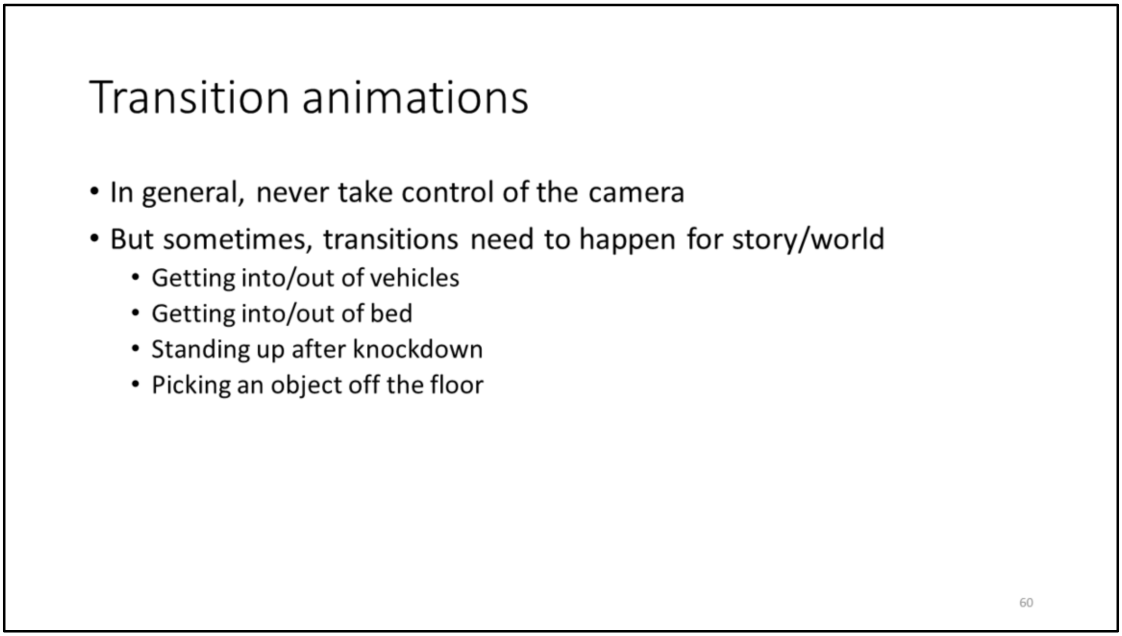
After my talk, I had a chat with Brad Herman from Dreamworks, and he mentioned they have chairs that are just high enough off the ground so that peoples' feet don't touch. And this works, even though they aren't as tall as bar stools. So my idea was somewhat flippant, but it turns out that any method of removing the pressure on the feet does indeed work – you don't actually need the eye height to match standing. He also described a sort of "saddle chair" with stirrups to do the same thing and stop the feet touching the floor. Thanks Brad!



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

This is the zeroth rule of VR.

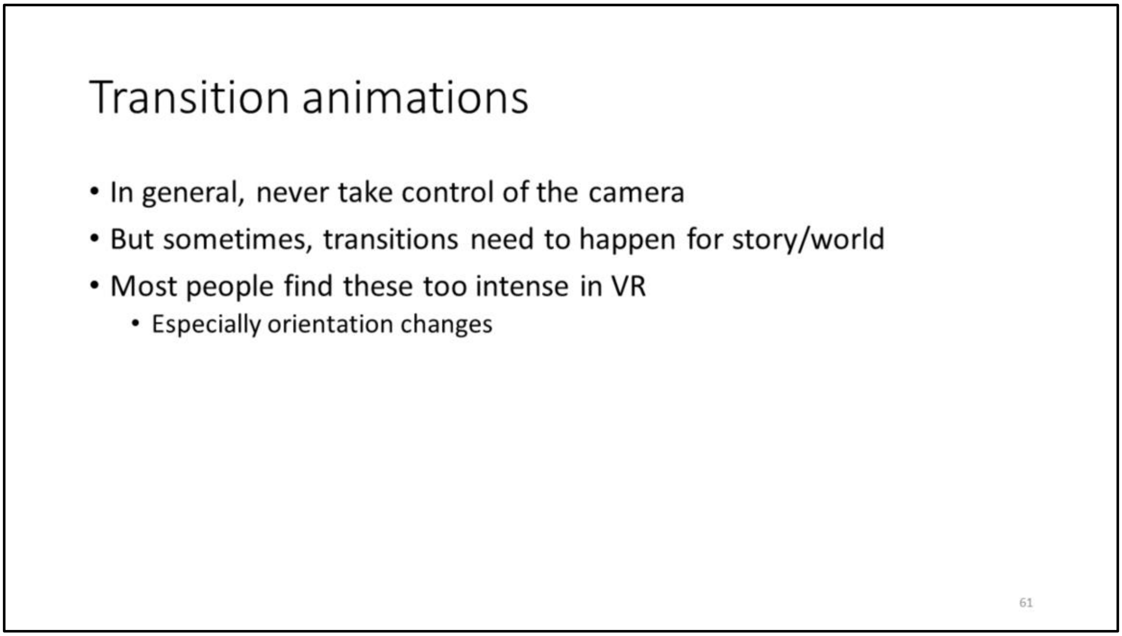


Transition animations

- 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

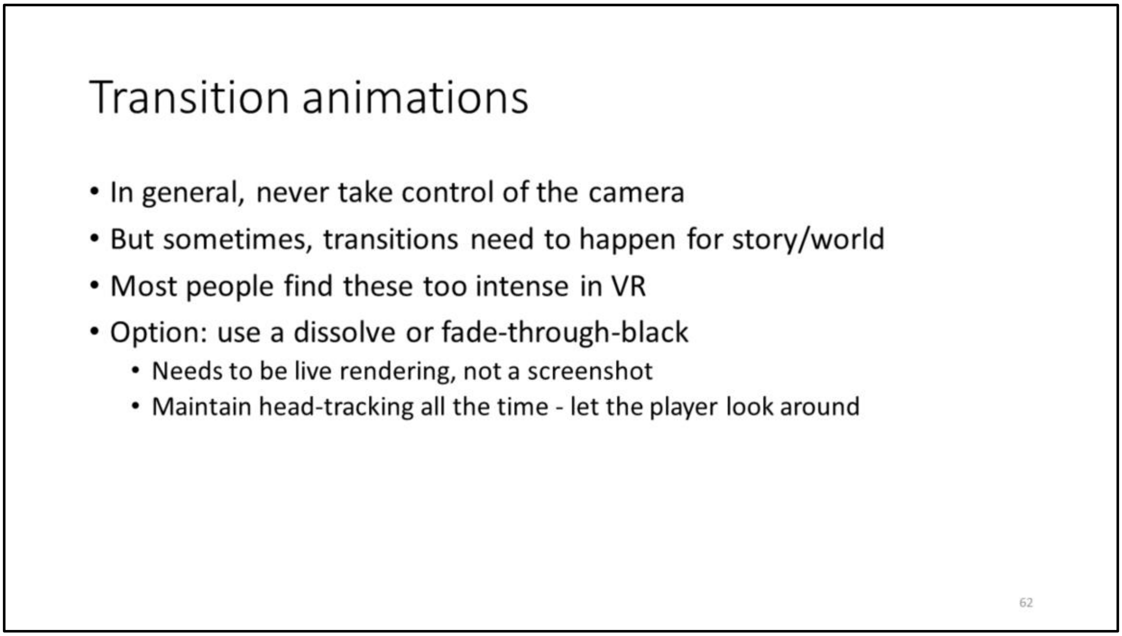
The knockdown itself is usually fast enough not to cause simulator sickness – it’s nearly a teleport. But getting up afterwards is five seconds of agony.

Even though picking an object off the floor causes no overall translation, think about the head motion involved!



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



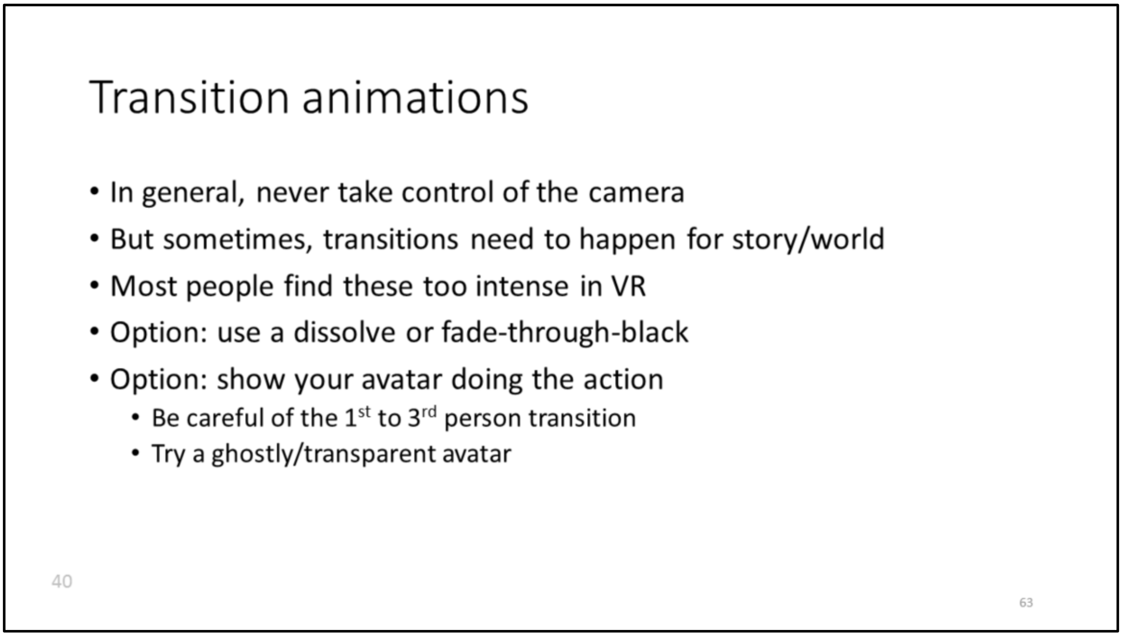
Transition animations

- 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: 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

62

Brad Herman from Dreamworks suggested using a “blink” animation rather than a simple fade-to-black. He says even really slow blinks seem “natural” to people.

Remember to keep head-tracking – during the fade or the blink, make sure if the player moves their head, the view will still move.



Transition animations

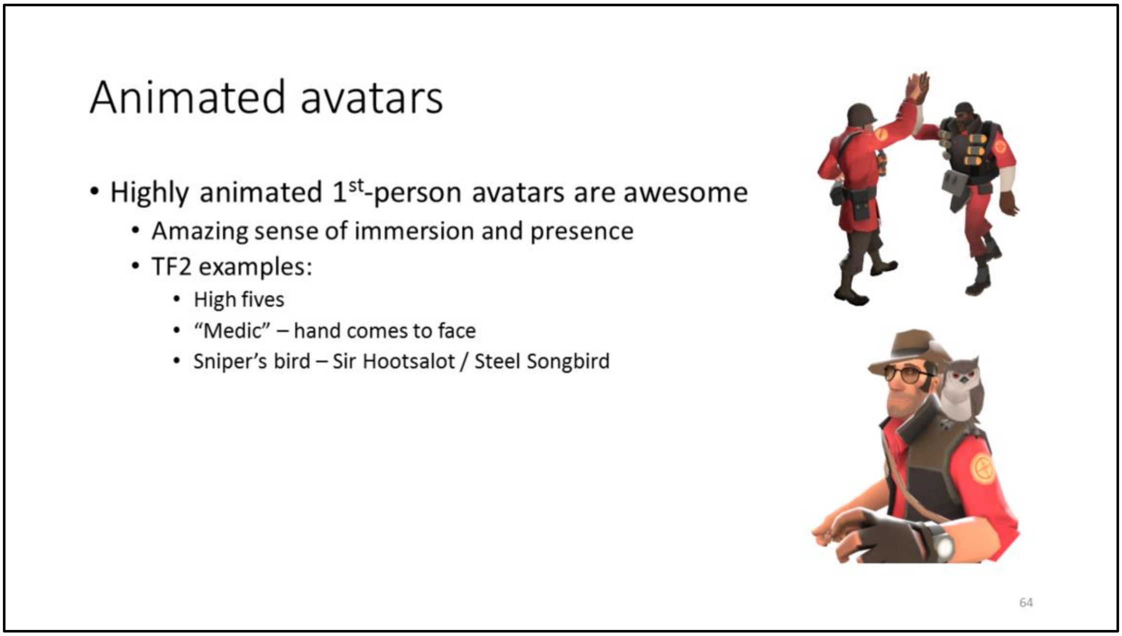
- In general, never take control of the camera
- But sometimes, transitions need to happen for story/world
- Most people find these too intense in VR
- Option: use a dissolve or fade-through-black
- Option: show your avatar doing the action
  - Be careful of the 1st to 3rd person transition
  - Try a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

40

63

The ghost option can be tough to retrofit nicely to an engine.

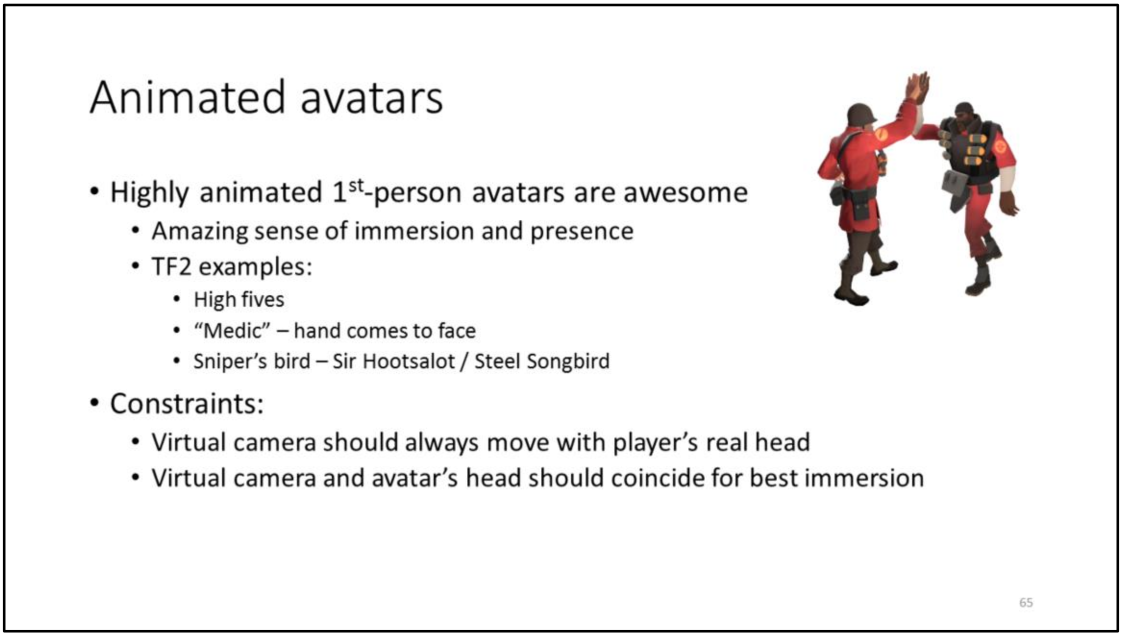
None of these solutions are great. Hopefully people will experiment with others and find out what works. More importantly, tell each other what DOESN’T work – saves everyone a lot of time (and since you’re not going to use it, because it doesn’t work, it’s not any sort of trade secret)



Animated avatars

- 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

Team Fortress 2 is not a good game to port to VR in many ways – things move too fast, and it’s much too intense. But the avatars and the animations and the ways you can customize them are fantastic for immersion. When rendering on a monitor, there are a bunch of custom-made 1st-person avatar models, but they are usually just a pair of arms, and they’re rendered with an odd FOV. In the VR version, we stopped using those (because we needed you to have an entire body) and used the existing 3rd-person models.

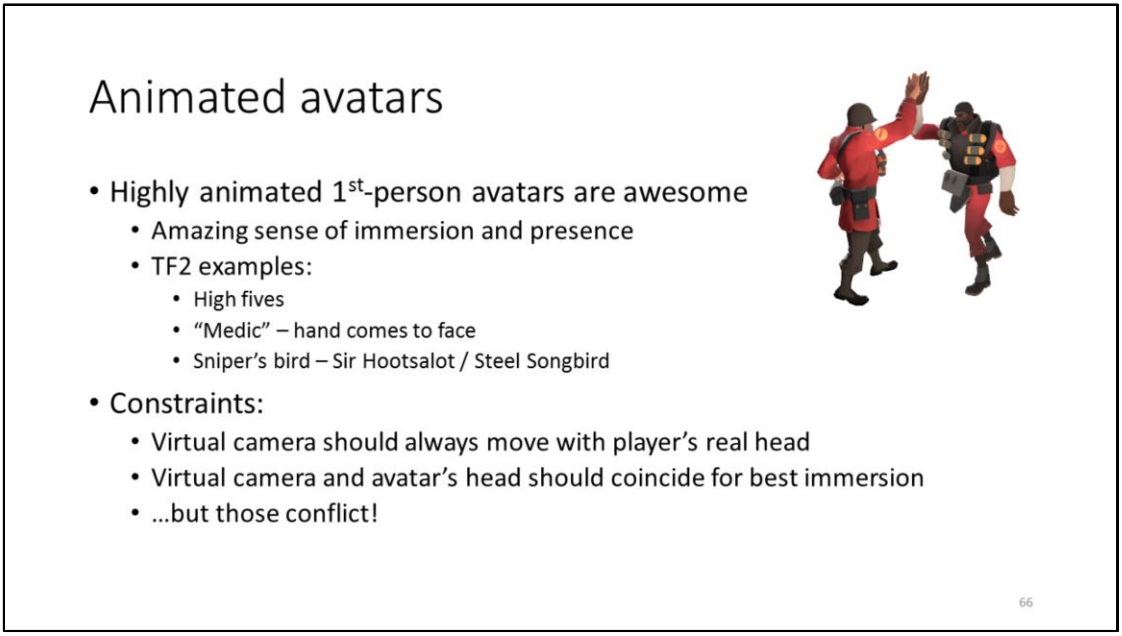


Animated avatars

- Highly animated 1^{st}-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

65

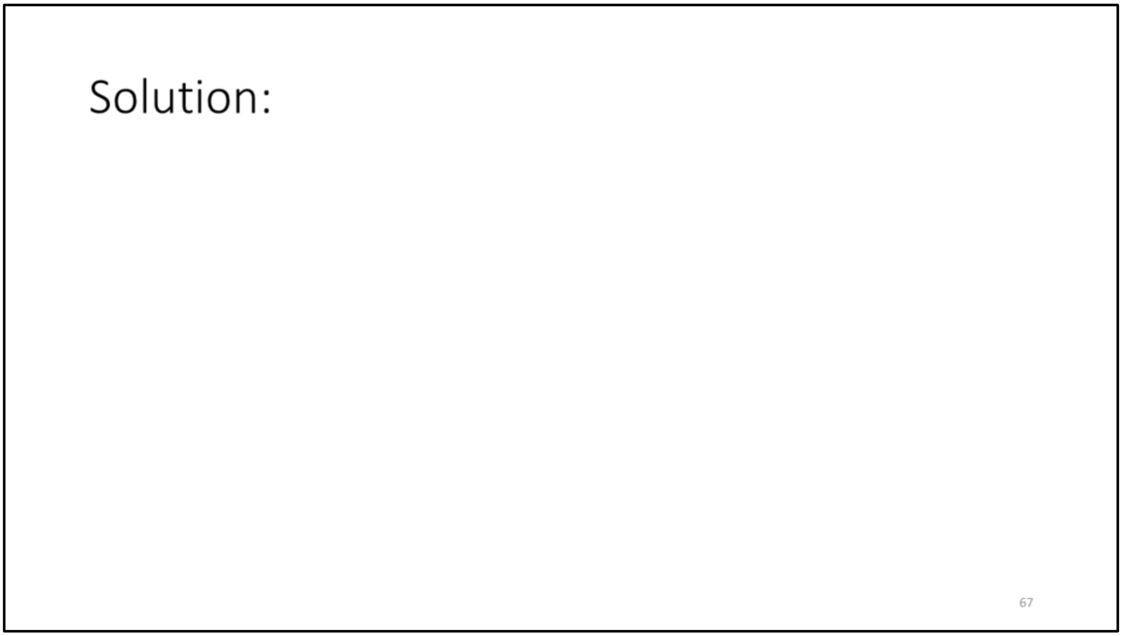
Again, first constraint is VR Rule Zero.



Animated avatars

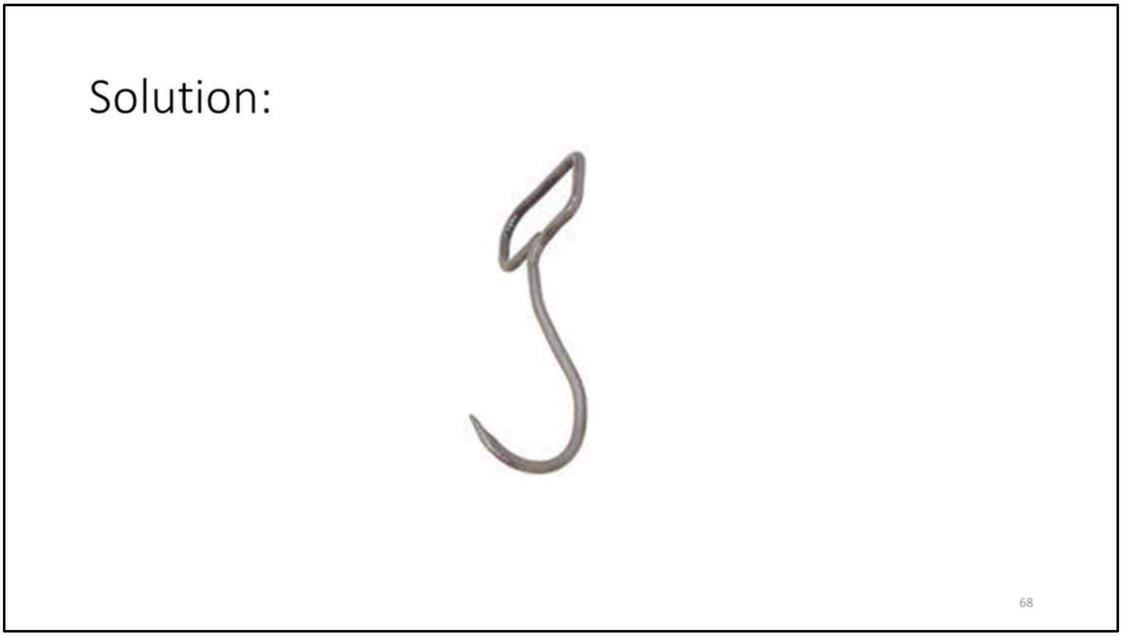
- Highly animated 1st-person avatars are awesome
  - Amazing sense of immersion and presence
  - TF2 examples:
    - High fives
    - “Medic” – hand comes to face
    - Sniper’s bird – Sir Hootsalot / Steel Songbird
- Constraints:
  - Virtual camera should always move with player’s real head
  - Virtual camera and avatar’s head should coincide for best immersion
  - …but those conflict!

Especially animations like the Soldier’s rocket-launcher reload anim – his head moves around a LOT.



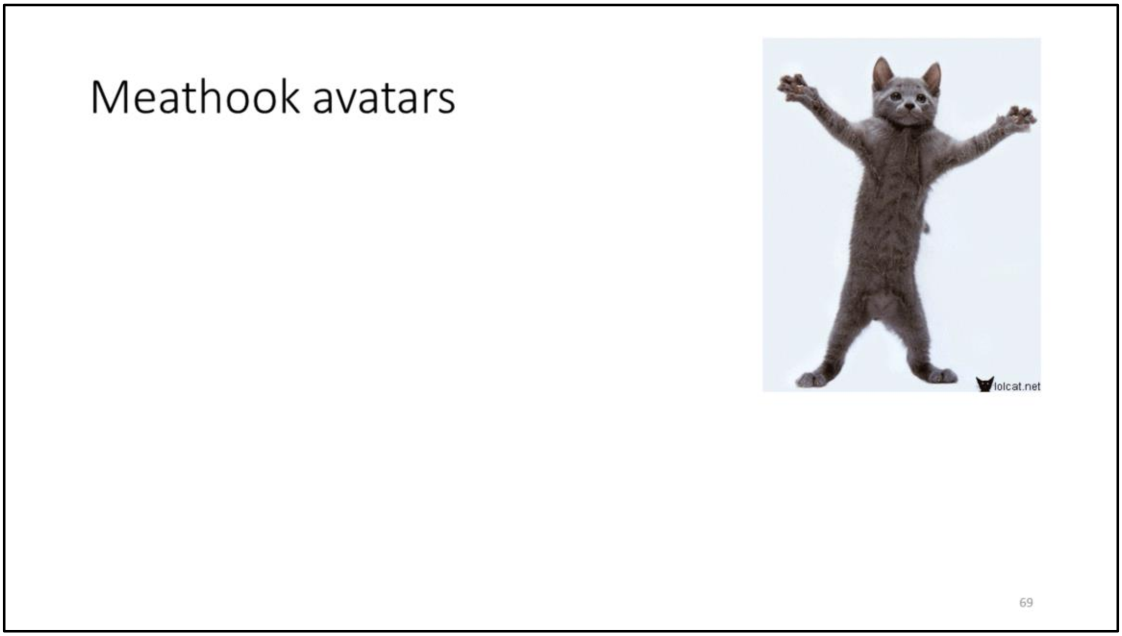
Solution:

Antici…



Solution:

EW!

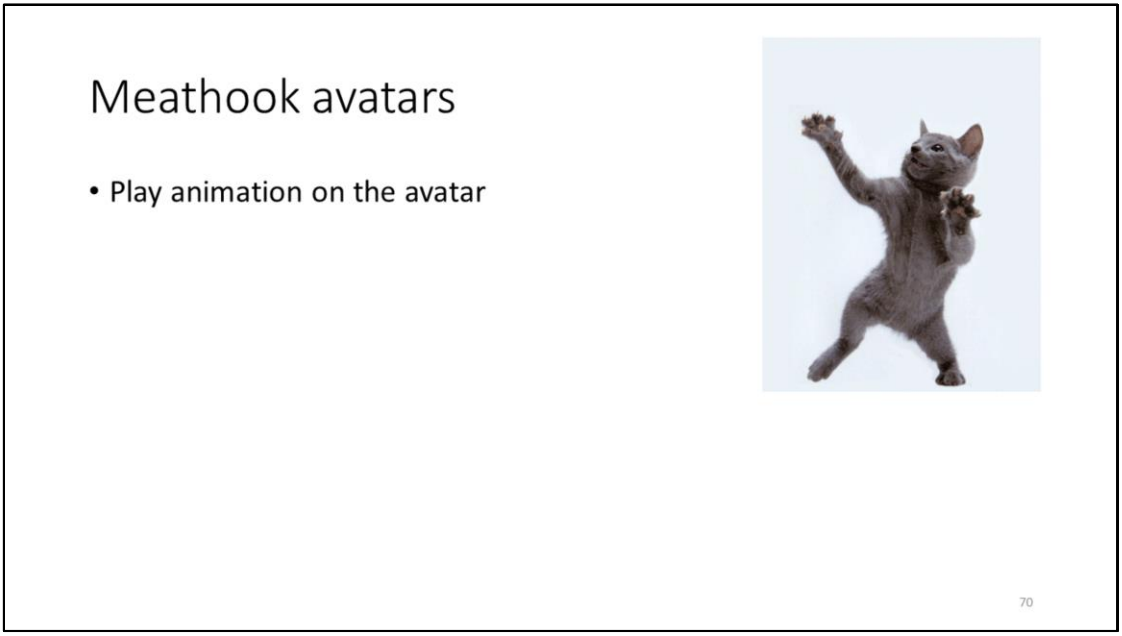


Meathook avatars

69

“Meathook Avatar” is the name of my death-metal band.

(this is an animated GIF and the cat is “dancing” – it’s an old internet meme I’m sure you’ve seen before)



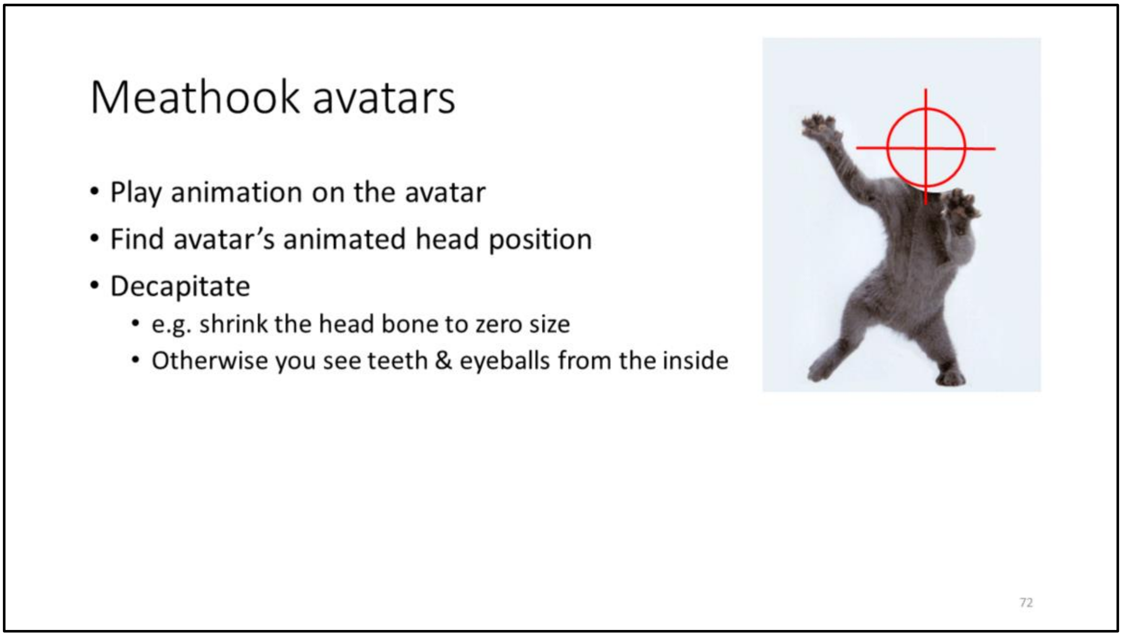
Meathook avatars

- Play animation on the avatar

70

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

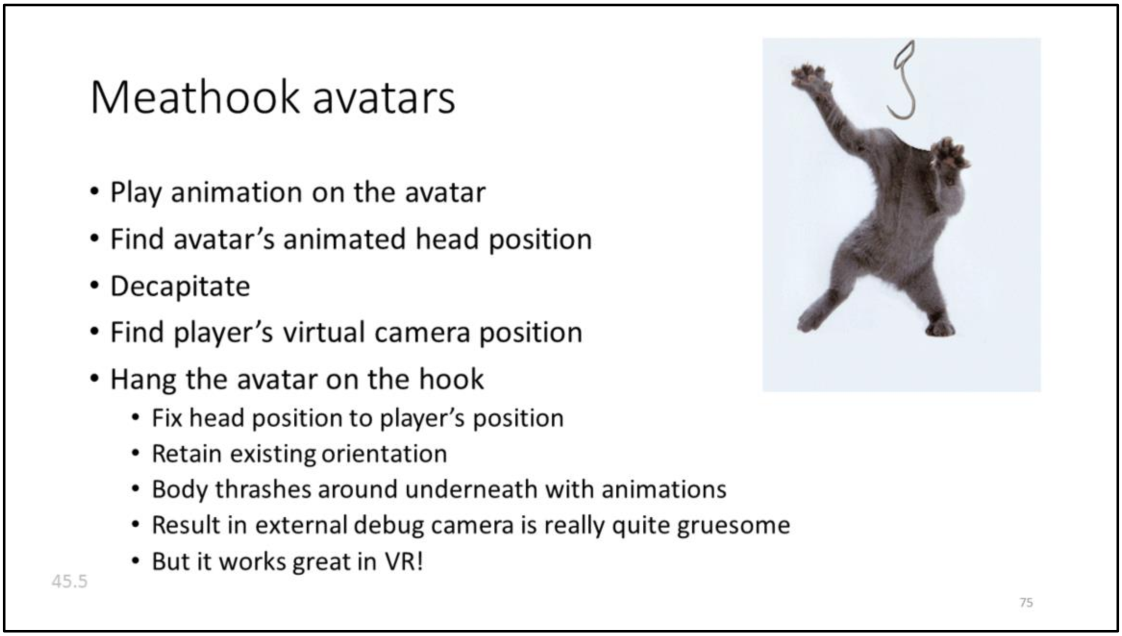
The War-Themed Hat Simulator team at Valve were devastated to discover we also had to turn off hats. The problem is that "hat" in TF2 includes things like glasses, masks & beards that don't render correctly when the camera is in inside them.

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!

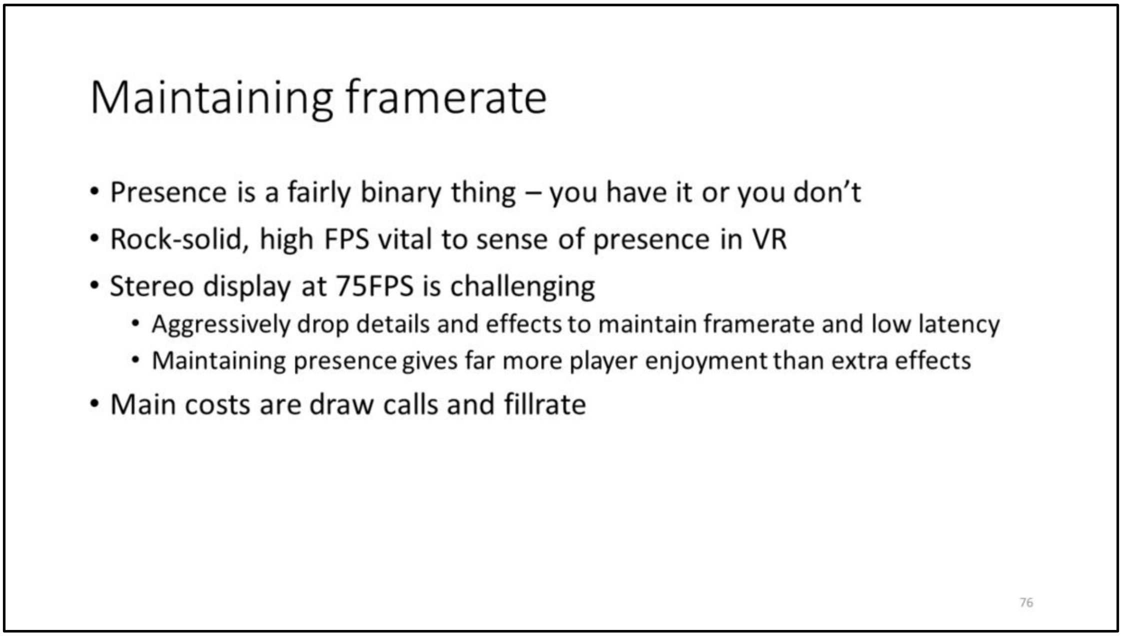
45.5

75

This picture is also animated – the position where the head would be is now fixed in a single place (i.e. where the player’s VR head is), while the body trashes around underneath it. This is where the “meathook” name comes from.

Obviously only do this for the 1st-person view of the current player. Don’t do it for all the other characters in the game! Also, don’t do it for the player’s shadow, or any reflections. Yes, this means your feet and your shadow’s feet don’t quite match up – but you really don’t notice that much. Ditto foot-slip – you really don’t notice that they’re sliding across the floor, because you have to look a long way down to even see them.

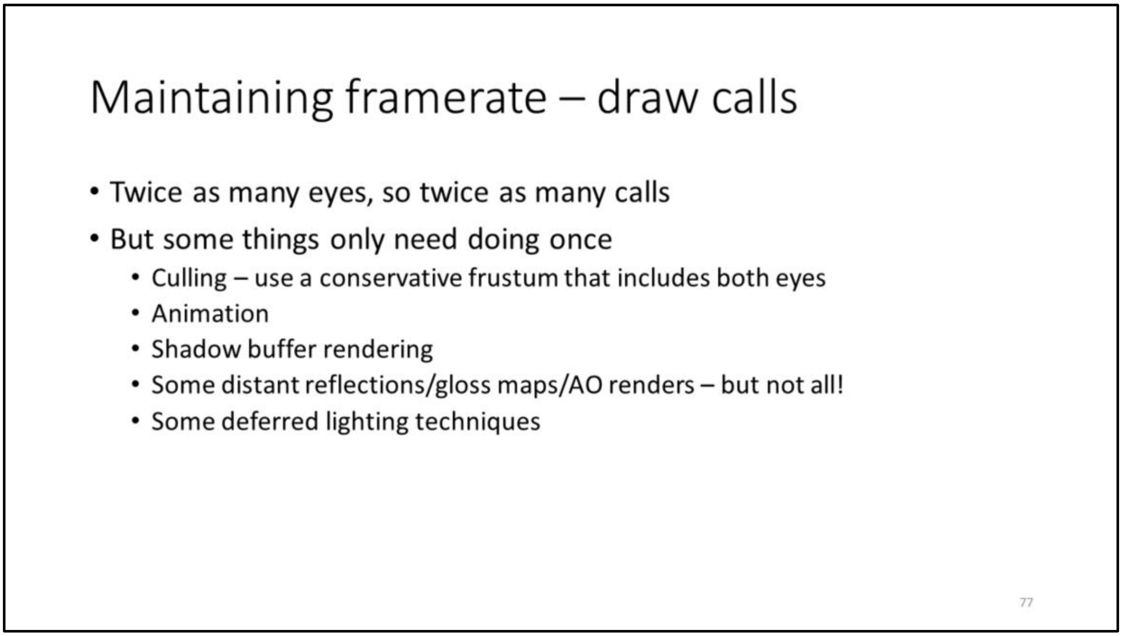
If you have a full-body IK system in your engine, you could possibly do this in a more elegant way – constrain feet and maybe hands to the animated positions, while moving the neck to the HMD-determined position, and let the waist sort itself out. But watch out for hand/face interactions e.g. the player holds a weapon up to their face, answers a telephone, lights a cigarette, etc. Now you need to start marking up animations to decide how the hands should IK – it can suddenly be a big increase in workload. Whereas this simple hack works fine – we didn’t need to modify any of the TF2 animations or meshes.



Maintaining framerate

- Presence is a fairly binary thing – you have it or you don’t
- Rock-solid, high FPS vital to sense of presence in VR
- Stereo display at 75FPS 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

Exaggerating for brevity – presence does actually have shades of grey, but they’re along other dimensions. In terms of framerate, there does seem to be a critical framerate at about 50fps, though for fast-moving scenes it can be higher. Note this is a different value from that required for “flicker fusion” which is significantly higher (varies widely, but around 90Hz seems to cover 90% of the population) – that’s something the HW designers need to worry about.

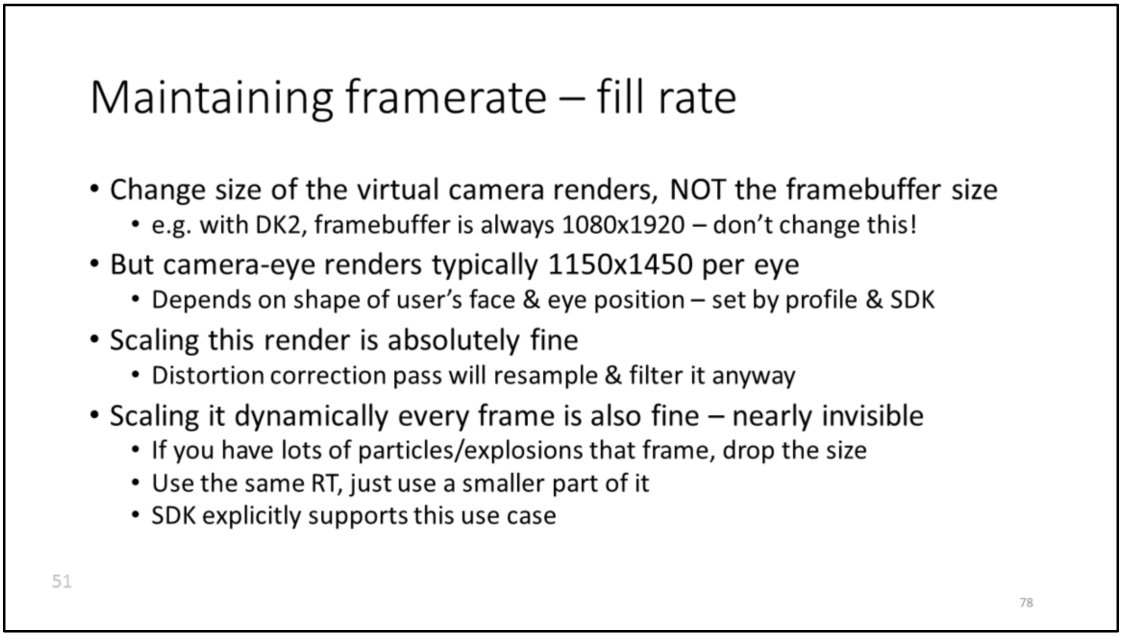


Maintaining framerate – draw calls

- Twice as many eyes, so twice as many calls
- But 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

77

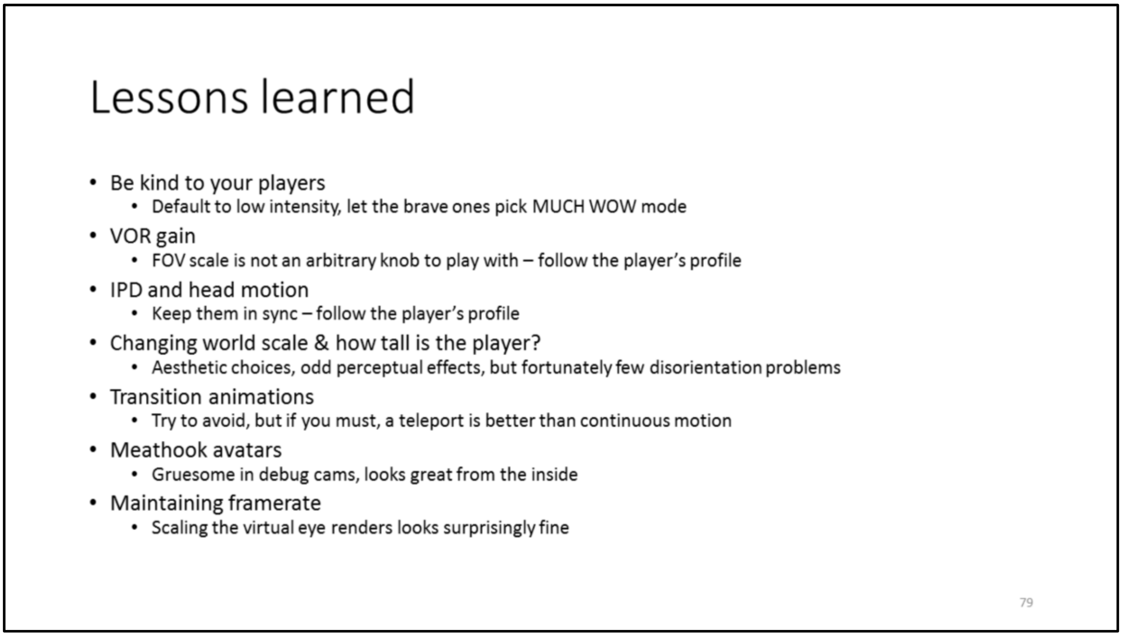
It looks like some of the upcoming DX12 and Mantle features should help reduce the cost of rendering objects multiple times, as well as reducing overall draw call costs.



Maintaining framerate – fill rate

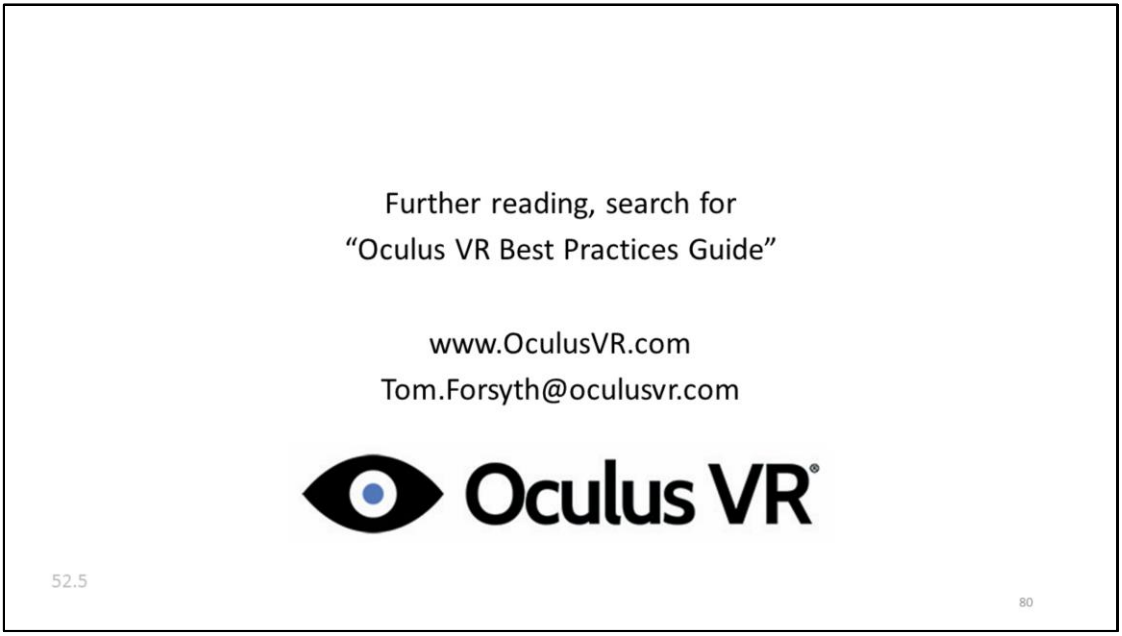
- Change size of the virtual camera renders, NOT the framebuffer size
  - e.g. with DK2, framebuffer is always 1080x1920 – don't change this!
- But camera-eye renders typically 1150x1450 per eye
  - Depends on shape of user's face & eye position – set by profile & SDK
- Scaling this render is absolutely fine
  - Distortion correction pass will resample & filter it anyway
- 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

The SDK has functions to help you decide the optimal-quality size of the eye-render targets, but of course it has no idea how fast or slow your engine is. If you have to choose between quality and sustained performance, performance should win every time. Maintaining framerate is so much more important than extra pixel detail.



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
- Maintaining framerate
  - Scaling the virtual eye renders looks surprisingly fine



Further reading, search for
“Oculus VR Best Practices Guide”

www.OculusVR.com
Tom.Forsyth@oculusvr.com

Oculus VR®

52.5