GENERATIONS / VANCOUVER
SIGGRAPH2018

The Challenges of Rendering an Open World in Far Cry 5

Stephen McAuley
Technical Lead
Ubisoft Montreal

Photography & Recording Encouraged

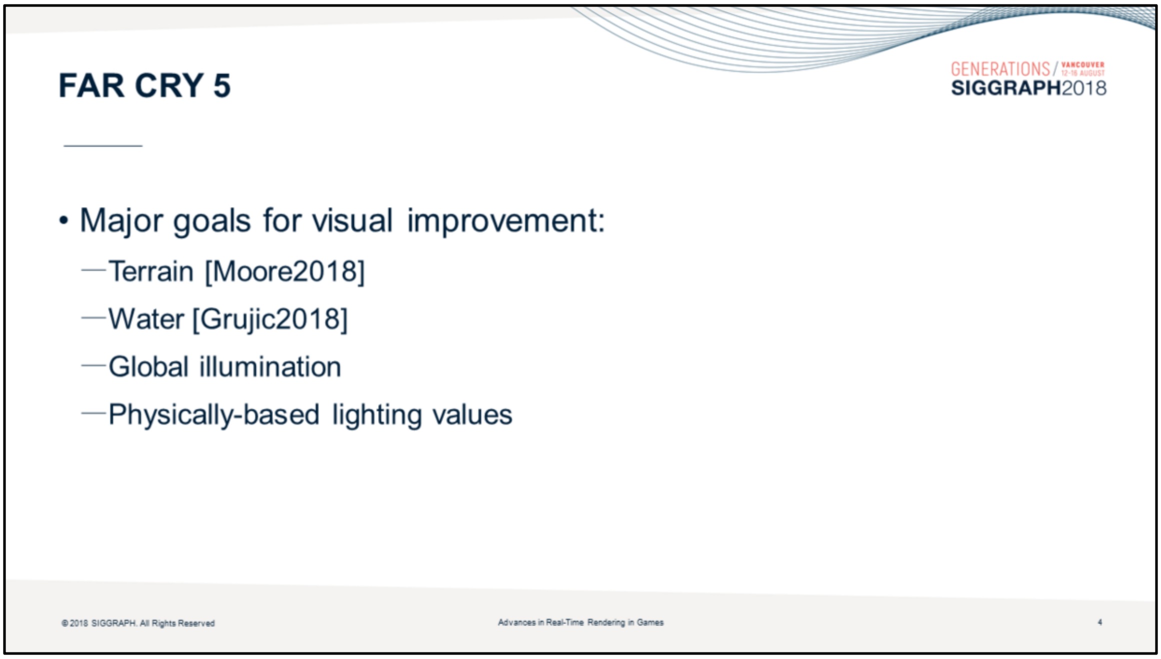
FAR CRY 5

- Open world first-person shooter set in Montana.
- Released in March 2018 on PC, Xbox One, PS4, Xbox One X and PS4 Pro.
- Engine evolved from Far Cry 4 and Far Cry Primal.
- Uses a deferred renderer.

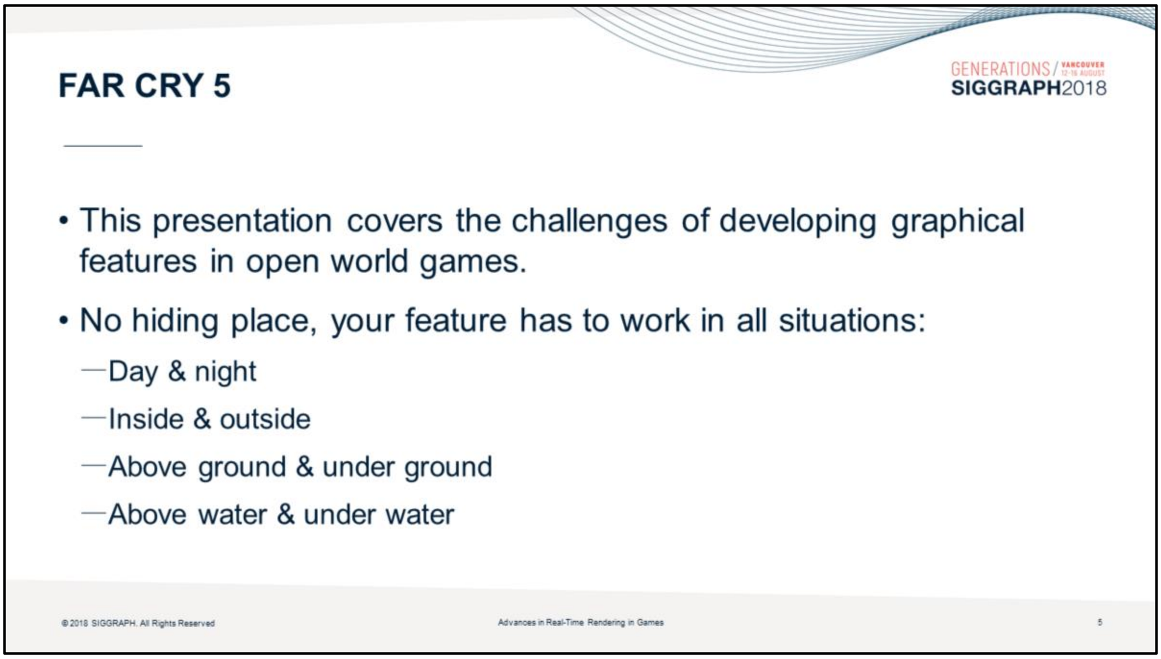
FAR CRY 5

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

- At Ubisoft, we have a strong co-development mandate.
- Graphics programming team:
  - Nine programmers in Montreal
  - Three programmers in Toronto
  - Three programmers in Kiev



We've already spoken about a couple of features we developed at GDC earlier this year…



…but this presentation we'll focus less on features but more on challenges, particularly the challenges of rendering open world games where there is no hiding place for many of your graphics developments.

FAR CRY 5

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

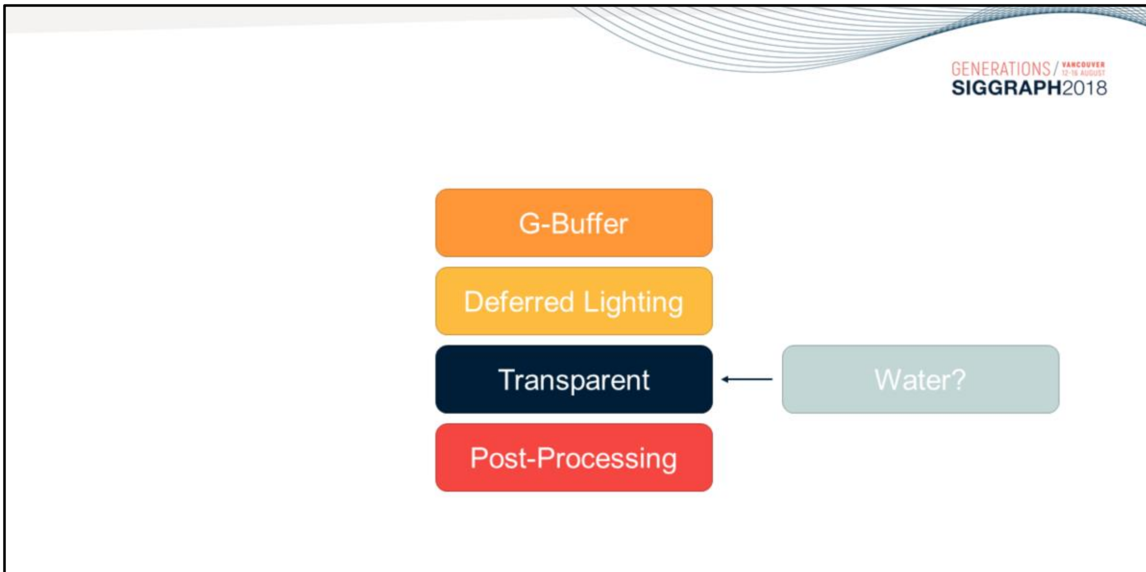
- Covering challenges in four main areas:
  - Water
  - A physically-based time of day cycle
  - Local tone mapping
  - Art production

WATER

WATER IS EASY, RIGHT?

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

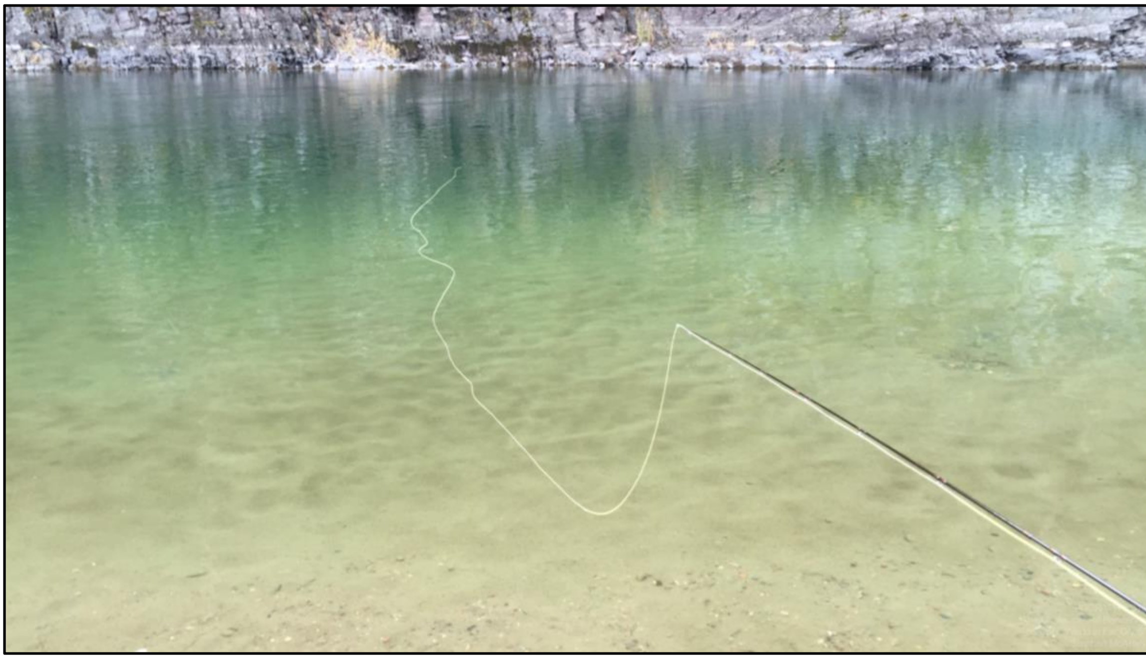

G-Buffer
Deferred Lighting
Transparent
Post-Processing


GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018
G-Buffer
Deferred Lighting
Transparent
Water?
Post-Processing

PROBLEMS

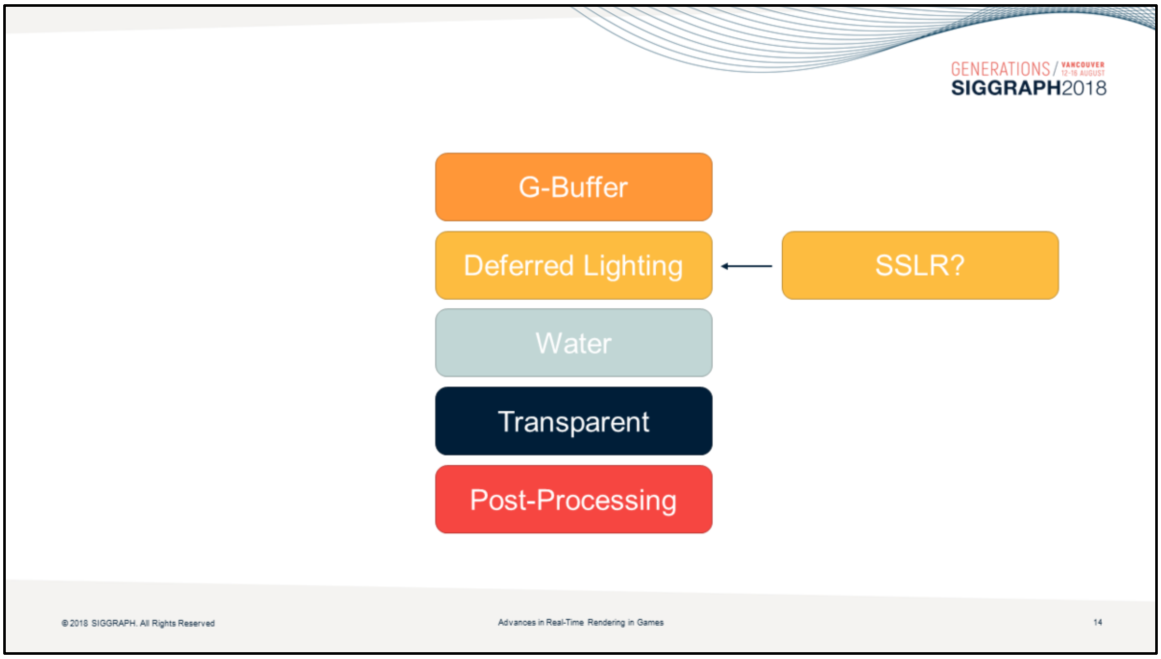
GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

- Sorting with other transparent objects will be a nightmare…
- What about refraction and underwater light transport?

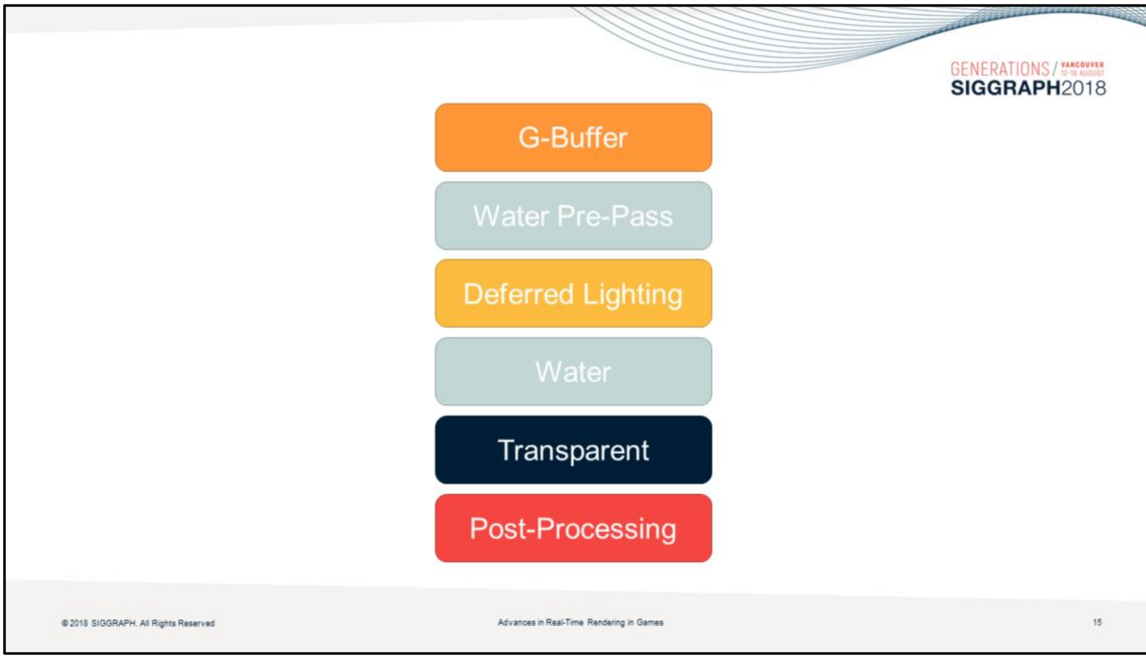


This is an image taken during a reference trip to Montana. Notice how vertically down the image you can see the colour of the water shift and change, and then the refraction present close to the camera at the bottom of the image. We'd like some of that!

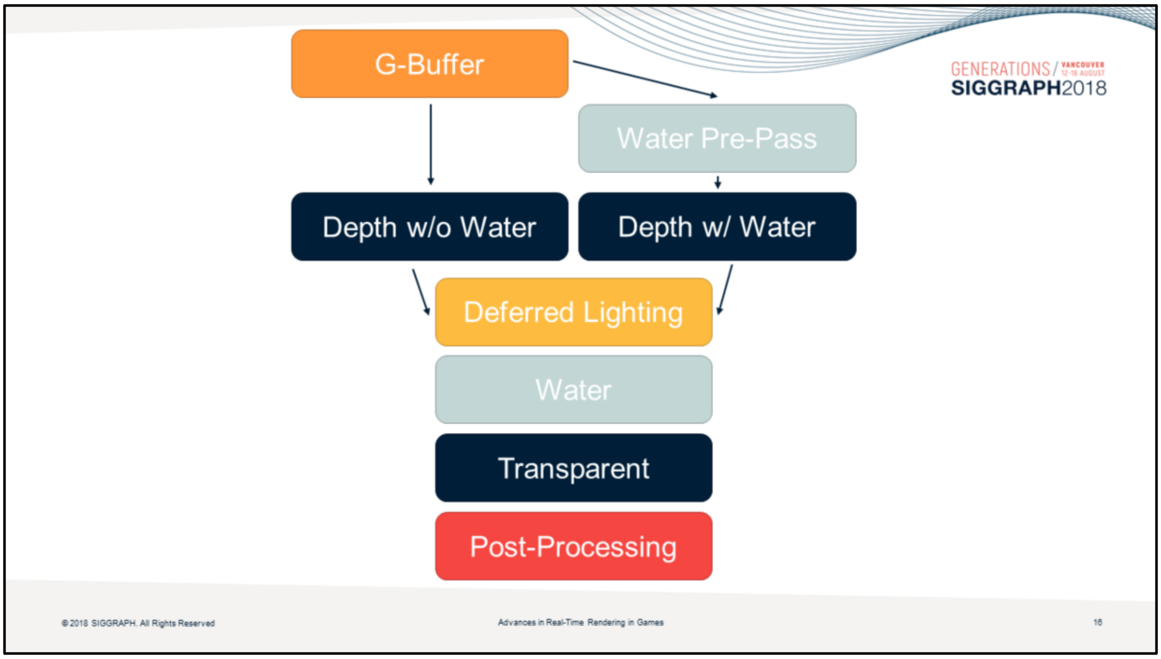
So now we have something that probably looks familiar to a lot of people here. You render your water before your transparent objects, having resolved the frame buffer beforehand to do some cool refraction effects. Water writes depth, so transparent objects don't appear underwater.



But we still have another problem. What about SSLR? We'd really like SSLR on water, but it's applied in the deferred lighting using G-Buffer data. Previously on Far Cry games we'd used a planar reflection, but it was difficult to maintain a forward rendering pipeline (and often it didn't match up with what was rendered in the main view), and ensuring we only had one water height to generate reflections at was always a pain for our art and world building team. Moreover, for Far Cry 5 we wanted sloped water for river rapids, so planar reflections would no longer work. Plus, if you already have SSLR for your world, why not re-use it for water?



So we render a water pre-pass with depth buffer and G-Buffer data BEFORE we do our deferred lighting passes. This can then be used by the SSLR.



Now for the deferred lighting, we need two depth buffers, one with water and one without. Some effects need with water (SSLR) and others need it without (shadows, SSAO, lighting).

WHAT DEPTH BUFFERS DO WE NEED?

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

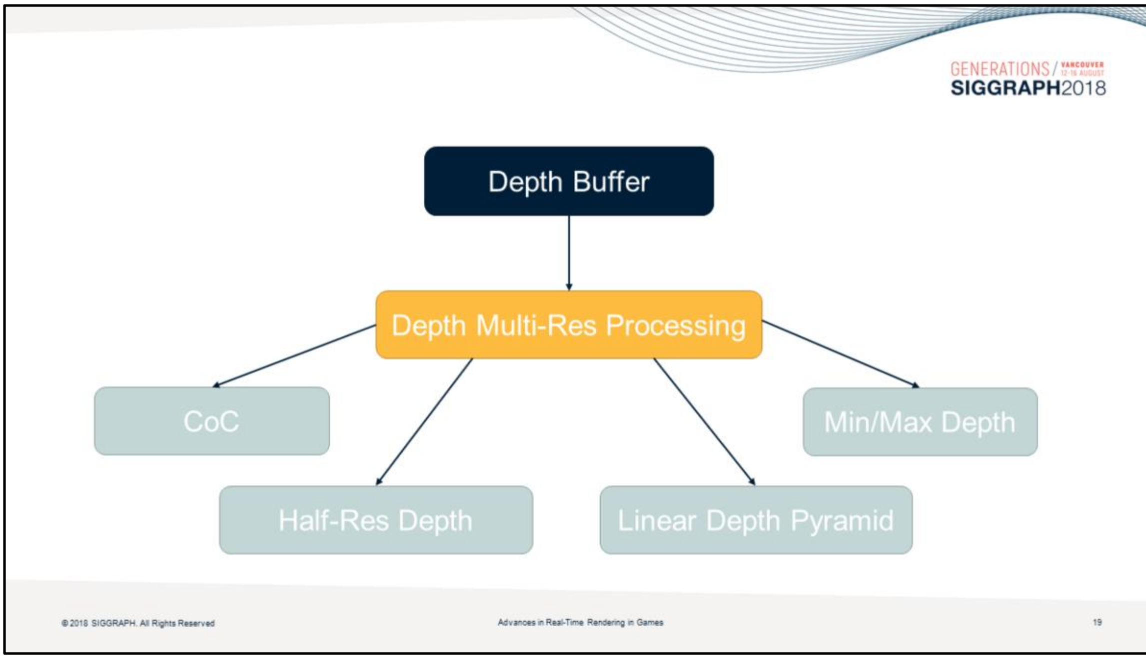
Deferred Effect	Depth Buffer
SSAO	no water
Screen Space Reflections	with water
Screen Space Shadows	no water
Shadows	no water
Lighting	no water
Fog	no water
Atmospheric Scattering	no water

We'll ignore SSAO and screen-space shadows on water, and although it'll be expensive, we can perform shadows, lighting, fog and atmospheric scattering in forward on water.

IT'S NOT THAT SIMPLE

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

- Our deferred effects like many different kinds of depth buffer:
  - Different resolutions
  - Linear vs non-linear
  - Min/max
  - Circle of confusion
- We have a specific pass to generate these.



This pass is called Depth Multi-Res Processing, the depth buffer goes in, and various downsampled and transformed depth buffers come out. This runs after the G-Buffer pass, in async compute while the shadows are running.

HALF-RES DEPTH

- Scene depth at (½, ½) resolution.
- Calculated with *min* filter.

LINEAR DEPTH PYRAMID

GENERATIONS / VANCOUVER 12-16 AUGUST SIGGRAPH2018

- Depth linearised into [0, 1] range and stored as 16-bit.
- Mip levels calculated with *min* filter.

16-bit is great as it reduces texture bandwidth needed to read depth, and the linearisation reduces ALU.

LINEAR DEPTH PYRAMID

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

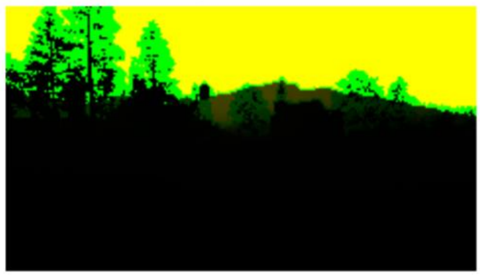
- A texture with mip maps pads to power-of-2 sizes. ☹
- Instead manually pack mips to save memory.
  - Store full-resolution in separate texture.
  - Store half-resolution and below in a packed atlas texture.
  - Helps the texture fit into ESRAM on Xbox One!

However, we had to pack this in a clever way. We wanted the mip levels all in one texture so the SSLR could hierarchically ray trace through it, but a standard packing wastes a lot of memory. This is really vital if you want to place the texture in ESRAM. We were continually juggling what we could and couldn’t fit in, and we realised this texture was wasting a lot of unnecessary space.

MIN/MAX DEPTH

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

- Depth downsampled to (1/8, 1/8) resolution.
- Both minimum and maximum per 8x8 tile stored in R16G16.



WHAT DEPTH BUFFERS DO WE NEED?

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

Deferred Effect	Depth Buffer
SSAO	linear depth pyramid
Screen Space Reflections	linear depth pyramid, half-res depth
Screen Space Shadows	linear depth pyramid
Shadows	depth, linear depth pyramid, half-res depth
Lighting	min/max depth, depth
Fog	min/max depth
Atmospheric Scattering	depth

Other effects also use depth buffers, that aren't listed here – such as half-res depth for low-resolution particles, or circle of confusion for depth of field, or motion blur.

As you can see, there's no obvious path.

POSSIBLE SOLUTION

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

- Run depth multi-res processing twice:
  - —Once with water
  - —Once without water
- Takes 0.5ms on PS4 in async compute and 8MB for the buffers
- To save performance and memory, only produce exact buffers needed?
  - —Not easy to identify and maintain

Depth multi-res processing runs in async compute with the shadows.

DECISION TIME

- Depth multi-res processing uses depth *with* water
- Pros:
  - —Artists preferred SSAO and SSS (screen space shadows) on water
  - —Optimisation as shadows, atmospheric scattering and fog are deferred on water
- Cons:
  - —Deferred shadows are on water surface
  - —Tile light culling does not properly cull lights under water
- No bugs reported by QC, so good trade-offs

MORE PROBLEMS

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

- What if we want to see transparent objects underwater?

One example of this is driving cars into the water! You don’t want their windscreens to disappear.

SPLIT TRANSPARENT PASS INTO TWO

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

- G-Buffer
- Deferred Lighting
- Transparent *Before Water*
- Water
- Transparent *After Water*
- Post-Processing

```
for each transparent object
    find water plane at XY location
    if above water plane
        render after water
    else if below water plane
        render before water
    else // if intersecting water plane
        render before and after water
    end
end
```

CULLING SOLUTIONS

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

- After water
  - Uses depth test to clip against water
- Before water
  - Culls against water clip plane in the vertex shader

CULLING SOLUTIONS

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

- After water
  - Uses depth test to clip against water
- Before water
  - Culls against water clip plane in the vertex shader

Top Tip: Don't write to SV_ClipDistance if you don't have to! We saved ourselves 0.1ms by disabling it.

EVEN MORE PROBLEMS

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

- What if we're underwater?

UNDERWATER

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

- Using depth *with* water for deferred effects no longer works.
- Many effects above water would disappear:
  - Lights (via tile light culling)
  - Sky/atmospheric scattering/fog
  - Shadows

UNDERWATER

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

- When underwater:
  - Use depth *without* water for deferred effects
  - Flip clip plane test for before/after water transparent objects
  - Disable SSLR

CONCLUSION

- It’s very hard to isolate features:
  - Water is inherently linked with SSLR and transparent objects.
  - Became linked with all deferred effects.
- Budget time for the unexpected problems, not just the features.
- No obvious (performant) answers to many of the questions.
- See [Grujic2018] for more about our water system.

FUTURE IMPROVEMENTS

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

- Replace clip plane test:
  - —Per-pixel inverse depth test against depth buffer with water.
  - —Works with sloped water and waves.
  - —Still need CPU to determine before/after water.
    - Move to a GPU rendering pipeline to assist with this.
- Look again at depth multi-res processing:
  - —Can we now determine exactly what needs depth with and without water?

A PHYSICALLY-BASED TIME OF DAY CYCLE

IN THE BEGINNING…

- First, place the sun and moon in the sky.
- Calculate sun and moon position from:
  - Longitude
  - Latitude
  - Time
- Use calculations from:
  - https://www.esrl.noaa.gov/gmd/grad/solcalc/calcdetails.html
  - [Jensen2001]

IN THE BEGINNING…

- Spectral data for both sun and moon:
  - Sun spectral radiance from [Preetham1999]
  - Moon spectral albedo from [Yapo2009]
- Moon lit in sun direction from sun data:
  - Phases are automatic from the moon's BRDF [Jensen2001]
- Sun and moon fed into Bruneton sky model:
  - Evaluate twice, for both sun and moon

IN THE BEGINNING…

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

- Local lights set in candela, lumens or EVs
- Global illumination system
  - 11 time of day key frames for sun and moon light
  - 1 key frame for local lights
  - 1 key frame for sky occlusion

PROBLEMS

1. Every day is different
2. Night looks the same as day
3. Physical lighting values cause too much contrast

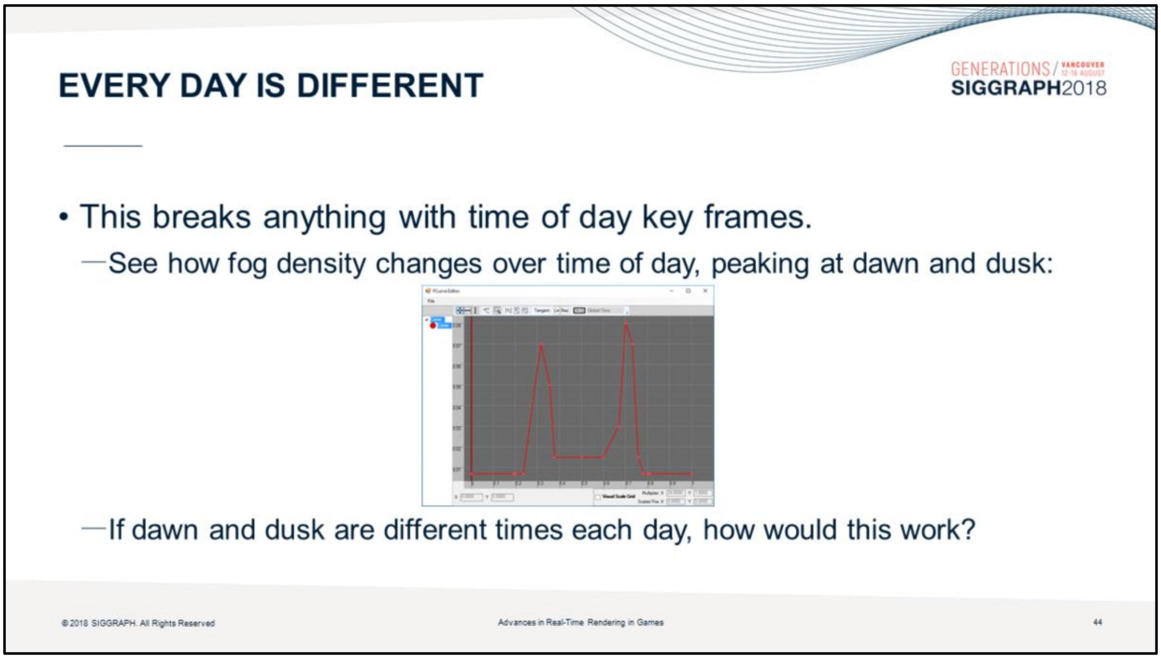
PROBLEMS

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

1. Every day is different
2. Night looks the same as day
3. Physical lighting values cause too much contrast

EVERY DAY IS DIFFERENT

- As time progresses, the sun and moon cycle changes.
- Different moon phases change the intensity of moon lighting.
- Sometimes there is no moon lighting at night at all.



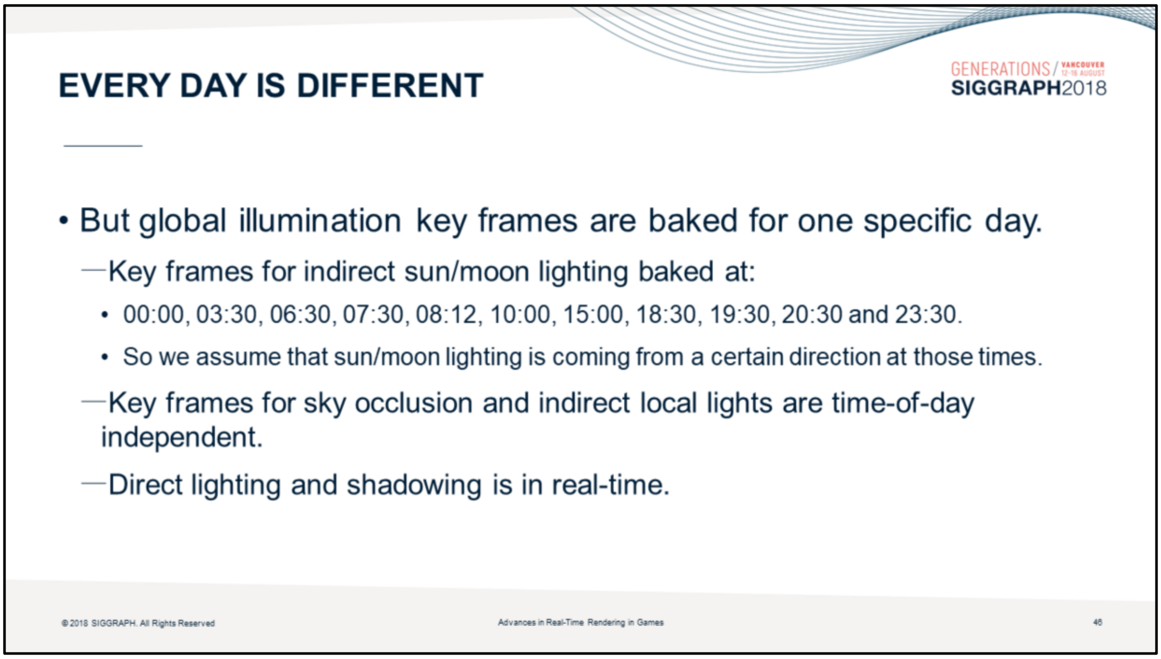
Fog inreases

EVERY DAY IS DIFFERENT

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

- Some effects use “sun elevation” instead of “time of day” to prevent this problem.

0.00	The sun’s lowest point below the horizon
0.25	The sun at the horizon (rising)
0.50	The sun’s highest point above the horizon
0.75	The sun at the horizon (setting)
1.00	The same as 0.00



We blend between the GI key frames as time of day changes to get our final sun/moon indirect lighting value. It's also worth saying that we never have both sun and moon at the same time – we take what's brightest in the scene at any given time.

So this means that GI will only work correctly for that day that we've baked.

EVERY DAY IS DIFFERENT

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

- Art issues:
  - Desire a directional (sun or moon) light at all times.
  - Lighting is hard to balance with an ever-changing scenario.
  - Night lighting is particularly hard with changing moon intensities.

SOLUTIONS

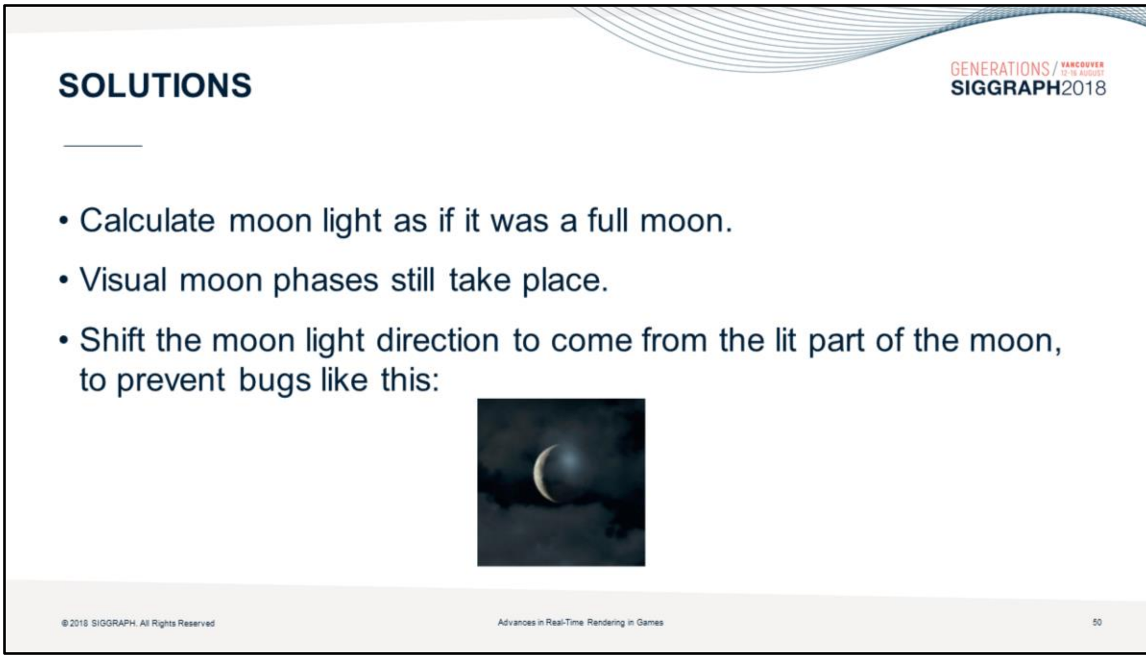
GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

- Pick a day that you like!
  - Either sun or moon is always present.
- Loop it continually.

SOLUTIONS

- Calculate sun and moon position for today and yesterday.
- Blend from today's position to yesterday's as the day progresses.
  - So tomorrow at 12:00am is the same as today at 12:00am.

```
CalculateSunPosition( timeToday, latitude, longitude, &azimuthToday, &zenithToday );
CalculateSunPosition( timeYesterday, latitude, longitude, &azimuthYesterday, &zenithYesterday );

float dayBlendFactor = secondsFromMidnight / (24.0f * 60.0f * 60.0f); // seconds in a day
float azimuth = LerpAnglesOnCircle(azimuthToday, azimuthYesterday, dayBlendFactor);
float zenith = LerpAnglesOnCircle(zenithToday, zenithYesterday, dayBlendFactor);
```



Al these things are presented as options to the artists, to lock/unlock various features, so we could easily restore any seasonal progression at any point in the future.

PROBLEMS

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

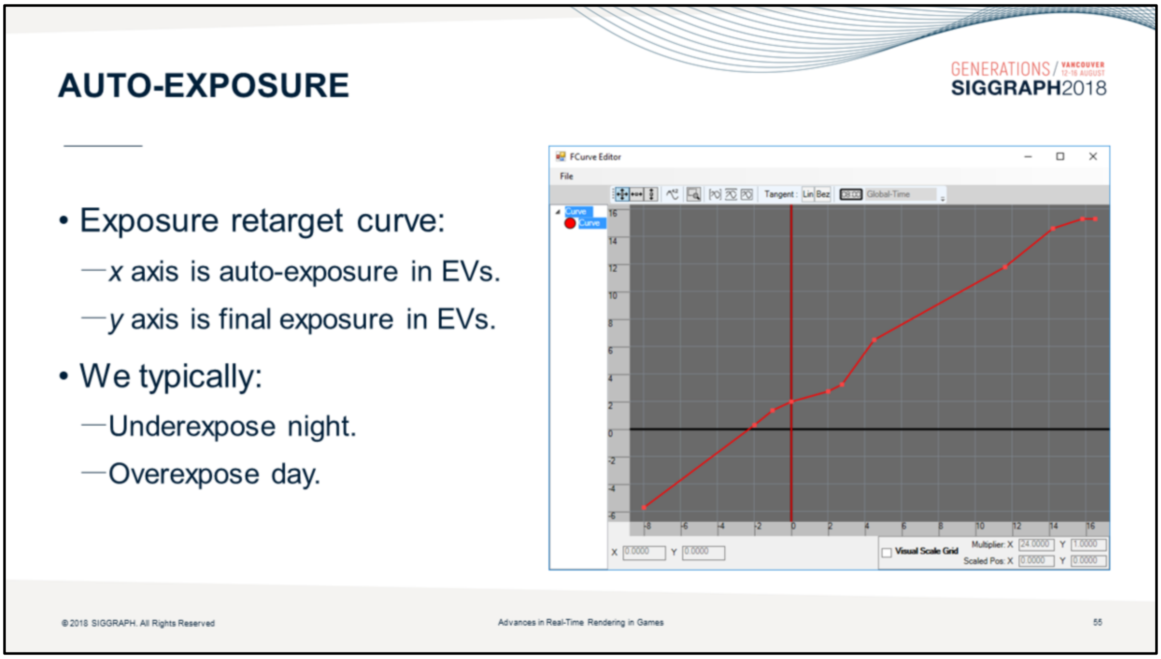
1. Every day is different
2. Night looks the same as day
3. Physical lighting values cause too much contrast


12pm


1am
SIGGRAPH 2018
"The Challenges of Rendering
an Open World in Far Cry 5"
Stephen McAuley

AUTO-EXPOSURE

- By default, auto-exposure will make both images the same brightness.
- At low EV levels, we *retarget* the exposure to underexpose to ensure the image looks dark.



So, for example, if the auto-exposure thinks the current exposure should be 2 EVs, it would look up into this curve and retarget to around 2.5 EVs. (2 EVs on the x-axis, 2.5 EVs on the y-axis).

By the way, this is one example of collaboration at Ubisoft. Projects generally do have their own engines, but even if we don't directly share code we do share a lot of ideas. This idea first came from Assassin's Creed Unity, it was adapted by Watch Dogs 2 and that's how we came to have it on Far Cry. So lots of Ubisoft games share this approach. Team work for the win!


Darkening the exposure
SIGGRAPH 2018
"The Challenges of Rendering
an Open World in Far Cry 5"
Stephen McAuley

PURKINJE EFFECT

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

- We don't simulate the Purkinje effect:
  - At low illumination levels, the human eye is more sensitive to blue light
- Human vision has two systems:
  - Photopic (cone-based)
    - Cones are more sensitive to colour and orange-yellow light, so more effective in bright light
  - Scotopic (rod-based)
    - Rods are more sensitive to light intensity and green-blue light, so more effective in low light

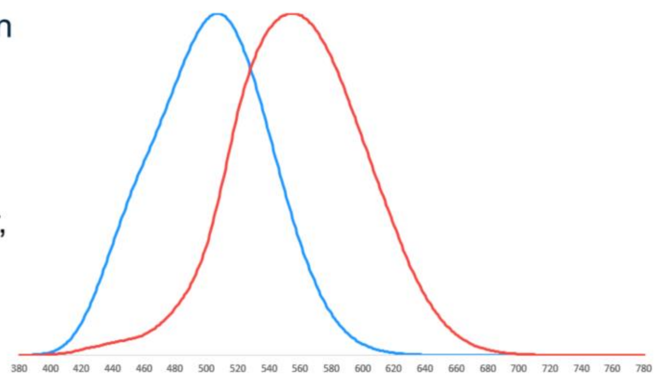
Advances in Real-Time Rendering in Games
57

On top of this there is also mesopic vision, which occurs at dawn/dusk and is a blend between the two.

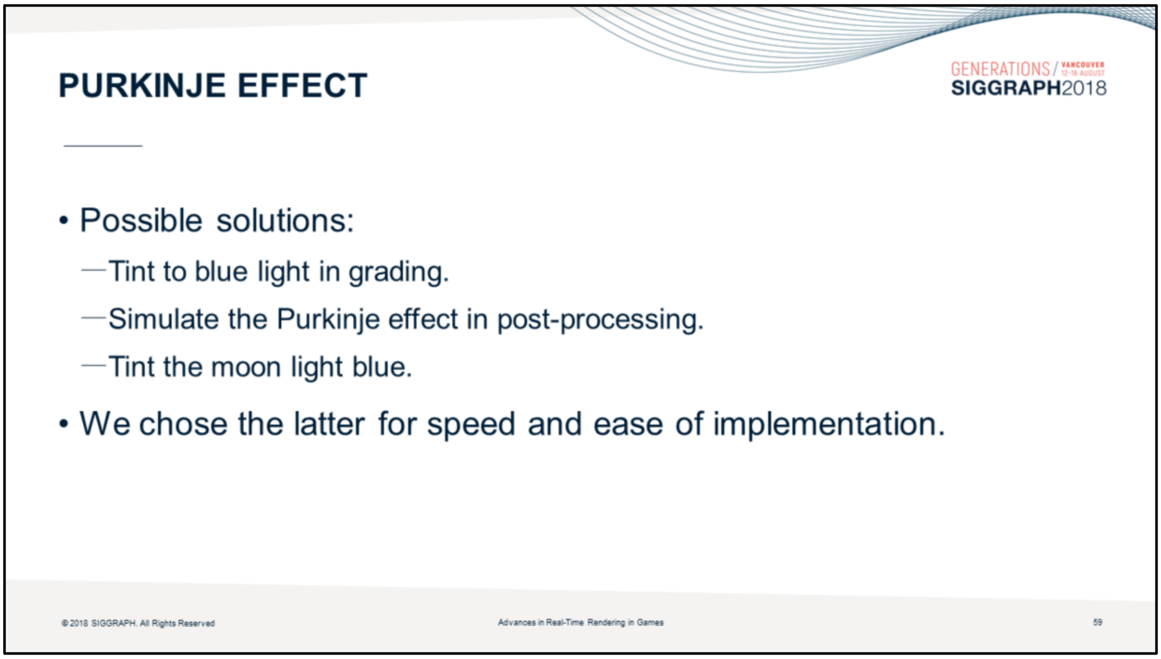
PURKINJE EFFECT

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

- Response of human vision at different wavelengths:
  - —Photopic
  - —Scotopic
- Observe the shift to lower, bluer wavelengths.



Graphs drawn with data from http://www.cvrl.org/.



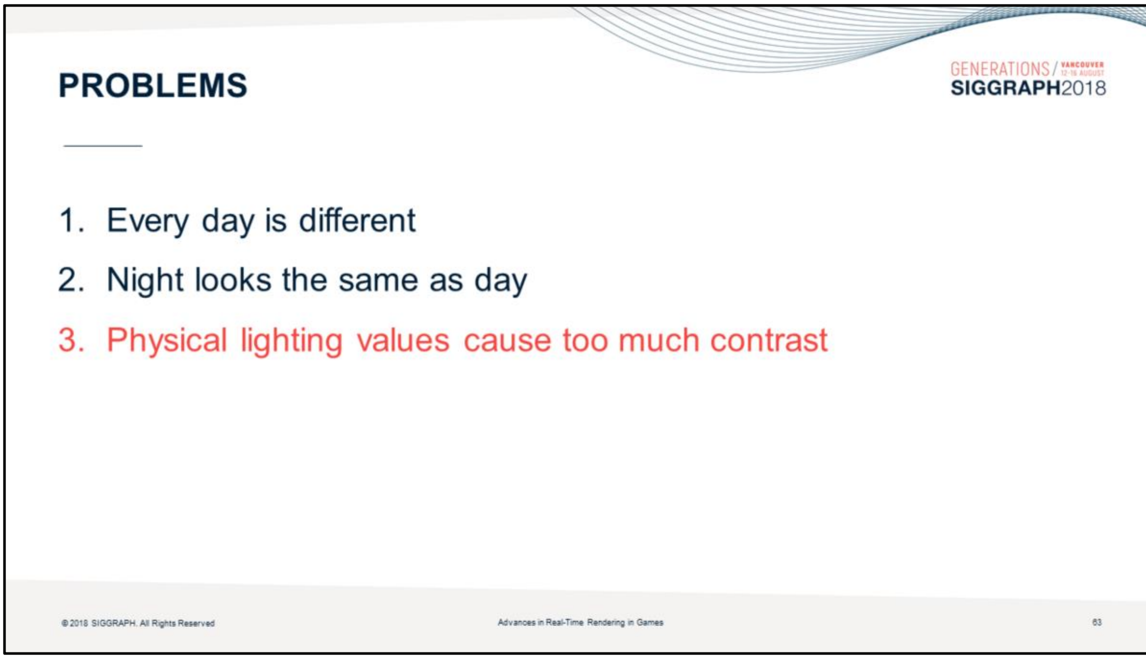
The Purkinje effect has an obvious performance cost in its implementation. We could also tint to blue light in grading, but artists found it difficult to maintain a good separation between a dark blue moon light, and the warm and red colours of local lights and fires. The easiest thing to do was tint the moon light blue – and this actually mirrors our approach for moon lighting in general, which is follow film and treat it as a separate light in the scene. In film they'd tint spot lights blue to mimic night lighting.

The downside to this is that we probably aren't going to handle mesopic vision correctly. We'll probably revisit implementing the Purkinje effect properly in the future.


Blue moon light tint
SIGGRAPH 2018
"The Challenges of Rendering
an Open World in Far Cry 5"
Stephen McAuley


Before
SIGGRAPH 2018
"The Challenges of Rendering an Open World in Far Cry 5"
Stephen McAuley

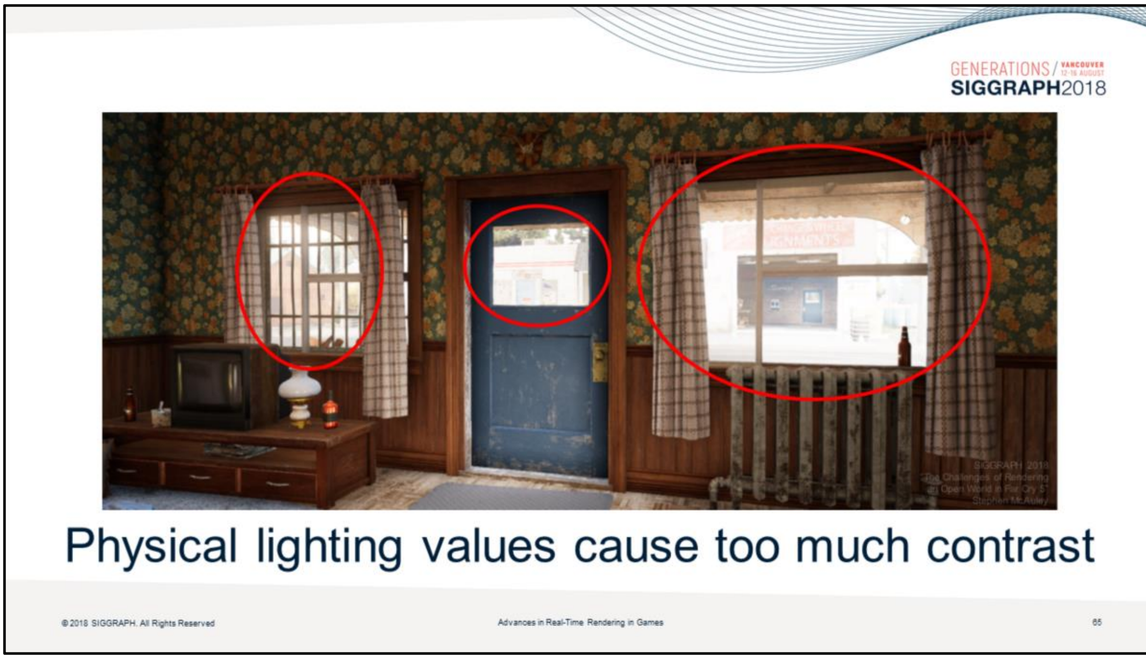

After
SIGGRAPH 2018
"The Challenges of Rendering
an Open World in Far Cry 5"
Stephen McAuley



Now that we have our single day, and night is looking somewhat like it should, we can focus on the biggest problem – which is that when using physical lighting values, it's incredibly difficult to control the range of values we get. The biggest issue being that we see too much contrast.

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

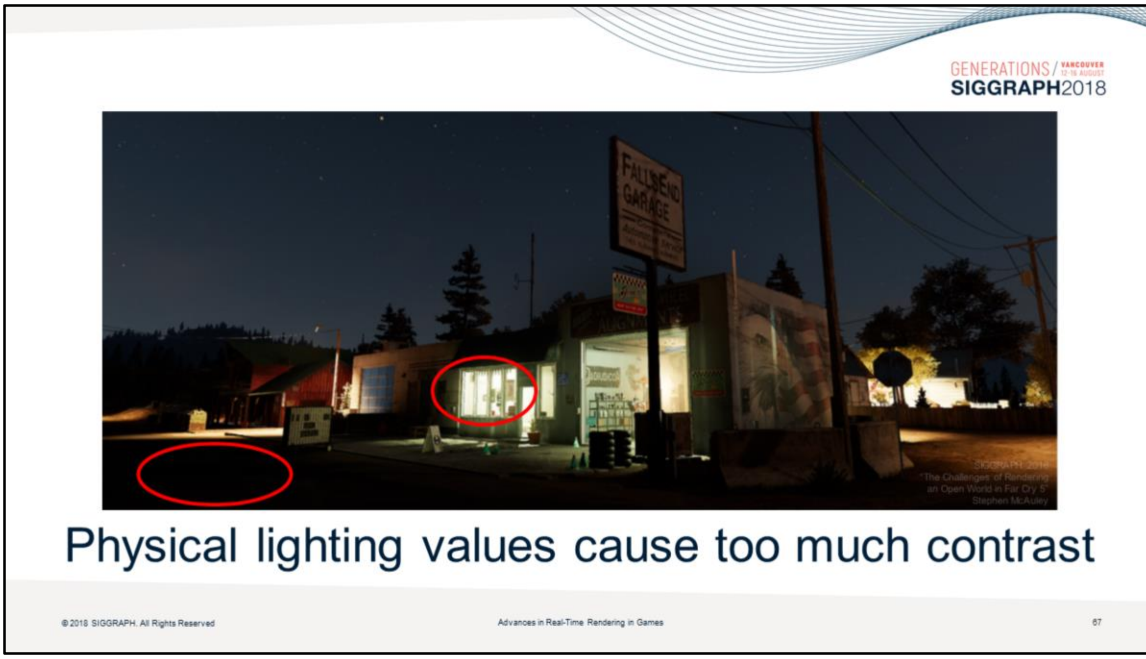
Physical lighting values cause too much contrast



In an interior looking outside, the outside is far too bright, making gameplay very hard.

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

Physical lighting values cause too much contrast



There are some incredibly dark areas, as well as some areas that are incredibly bright. Both areas are nearly unplayable.

Lighting Environment	EVs
Sun light, exterior	17
Sun shadow, exterior	13
Sun bounce, interior	6-10
Local lights	3-7
Moon light, exterior	-3
Moon shadow, exterior	-7

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

Lighting Environment	EVs
Sun light, exterior	17
Sun shadow, exterior	13
Sun bounce, interior	6-10
Local lights	3-7

Darken the sun light?

Common solution that's proposed

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

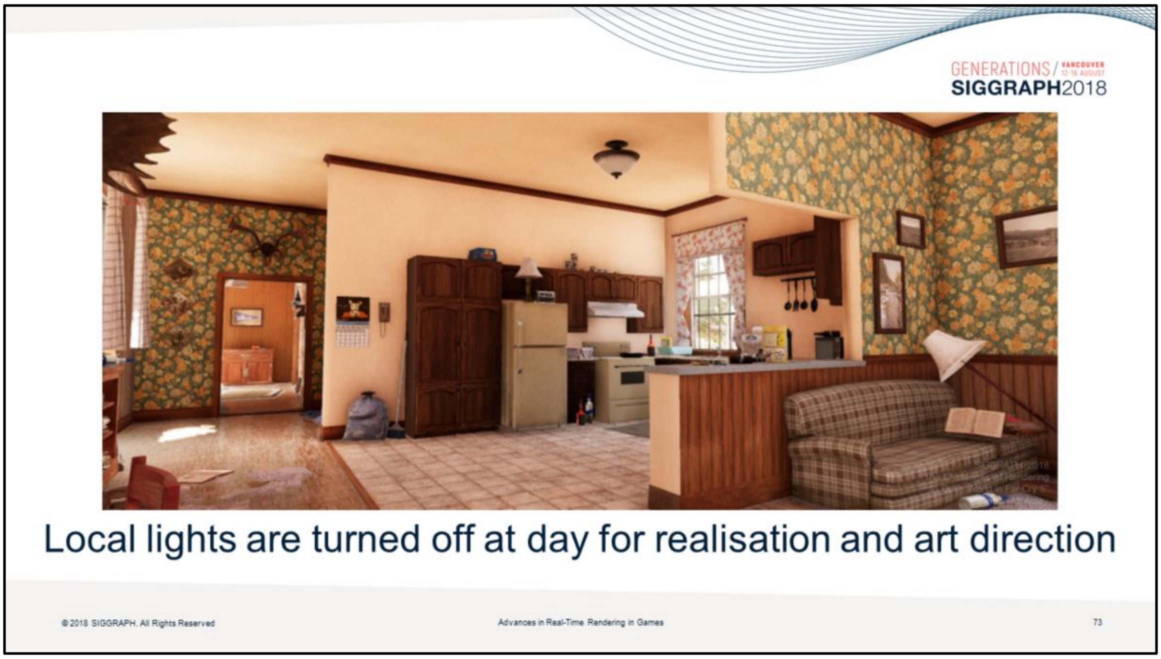
Lighting Environment	EVs
Sun light, exterior	17 → 13
Sun shadow, exterior	13 → 9
Sun bounce, interior	6-10 → 2-6
Local lights	3-7

Darken the sun light – by 4 EVs

The sun lights the sky, so if you darken the sun, you darken the sky… which darkens the sky light. And of course, darkening the sun reduces the bounce lighting coming from the sun.

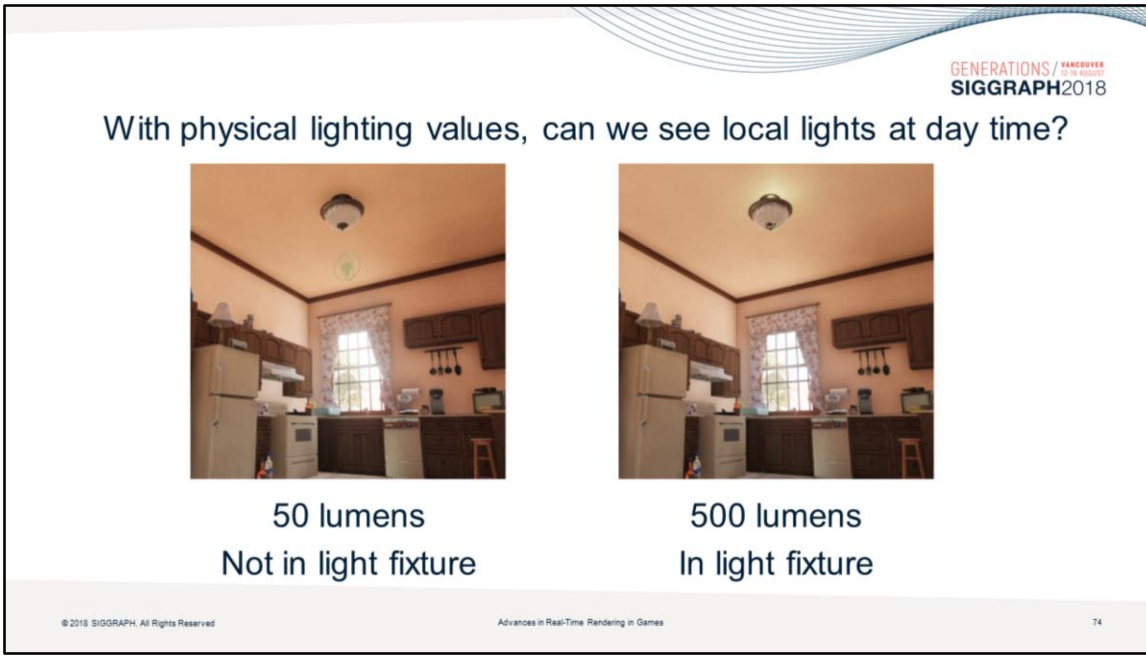
Lighting Environment	EVs
Sun light, exterior	17 → 13
Sun shadow, exterior	13 → 9
Sun bounce, interior	6-10 → 2-6
Local lights	3-7

Do local lights help reduce the contrast ratio?



Local lights don’t help either.

Art direction want something that looks like this! Having local lights would make it look like night, at day.



Lighting artists always complained that they couldn’t see local lights at day time. Notwithstanding the fact that they were often turned off during the day for realisation purposes, the major problems were not placing lights in the light fixture (so you could see their effect on the ceiling) (this one is about 75cm lower than it should be) and making them too dark. That 1/r^2 falloff is very important to understand!

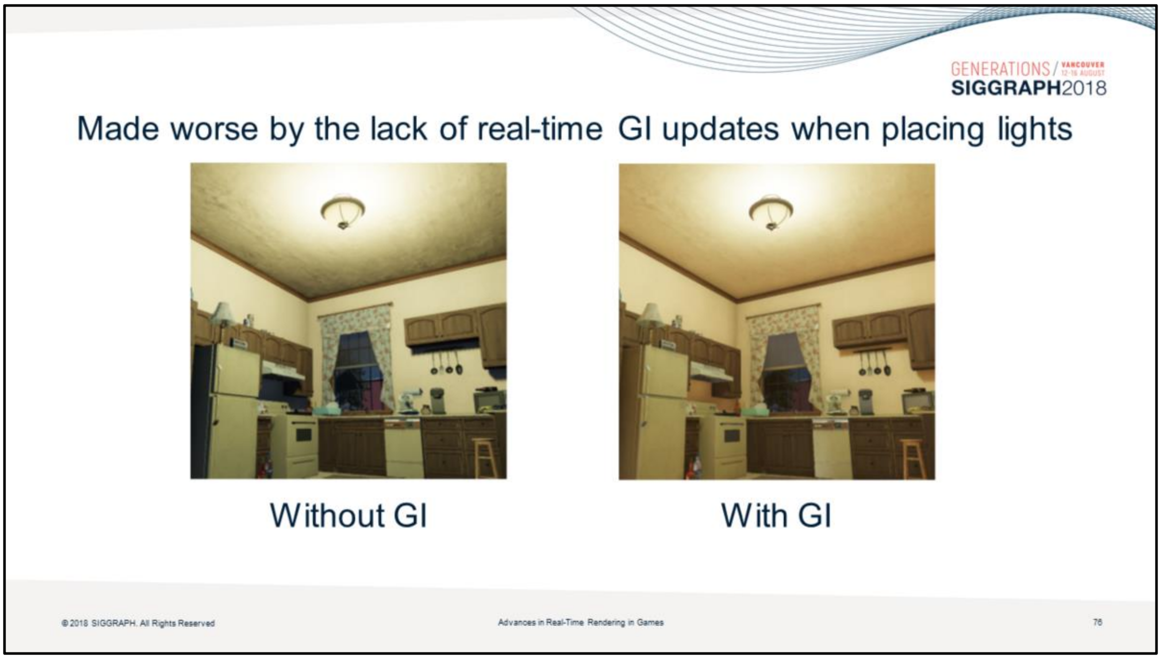
But why did they do that? Why are the lights darker than they should be, and why are they out of light fixtures?

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

At night, artists feel the lighting is too bright around the light fixture

50 lumens
Not in light fixture

500 lumens
In light fixture



Having no real-time GI feedback enhances the contrast in the scene, and makes artists move lights away from the light fixtures.

We've discovered a real problem right now. Artists aren't setting their lights up correctly, and that leads to unexpected and undesirable behaviour at day time. So let's ignore the day time contrast problem for now and let's address why lighting artist aren't setting up lights correctly.

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

…and address the problem of night.

The reason for that is the enhanced contrast they see at night. We’re going to explore this further.

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

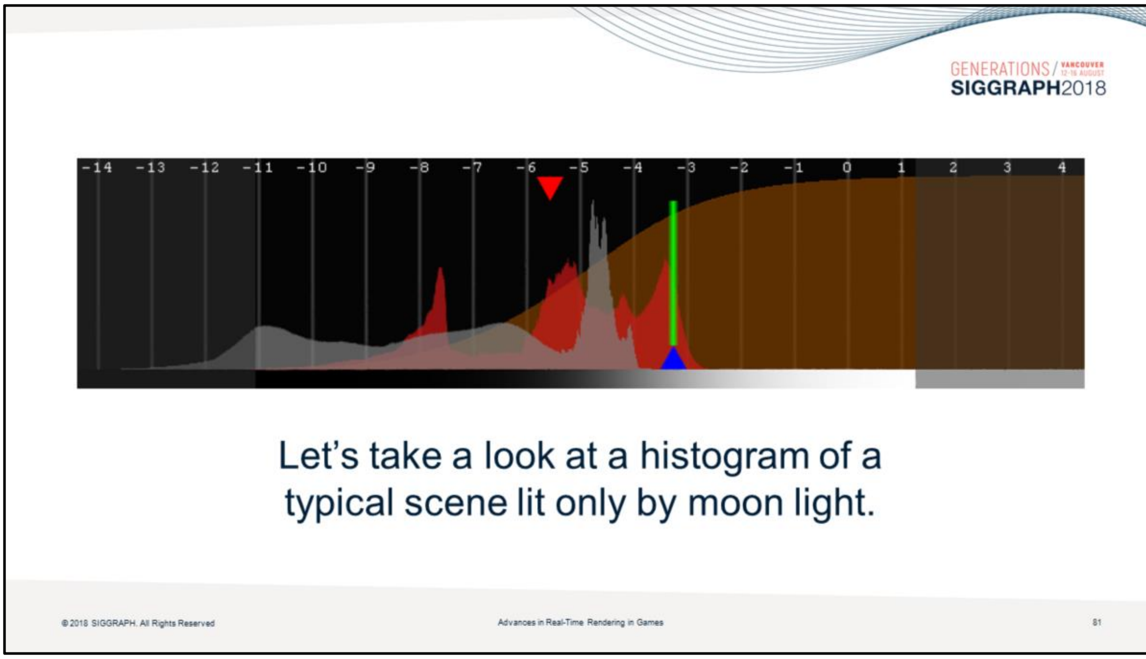
Let’s do some maths…

Light Type	Distance (m)	Illuminance (lux)
Full moon, direct light	n/a	0.07
Full moon, sky light	n/a	0.004
500 lumen bulb	0.1	50,000
500 lumen bulb	1	500
500 lumen bulb	100	0.05
5000 lumen street light	1000	0.005

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

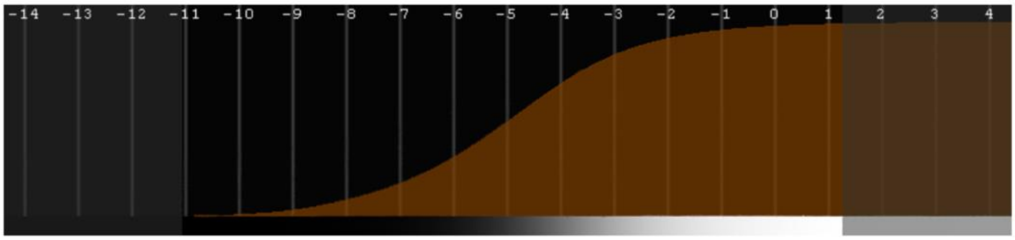
~4 stops of contrast

Light Type	Distance (m)	Illuminance (lux)
Full moon, direct light	n/a	0.07
Full moon, sky light	n/a	0.004
500 lumen bulb	0.1	50,000
500 lumen bulb	1	500
500 lumen bulb	100	0.05
5000 lumen street light	1000	0.005

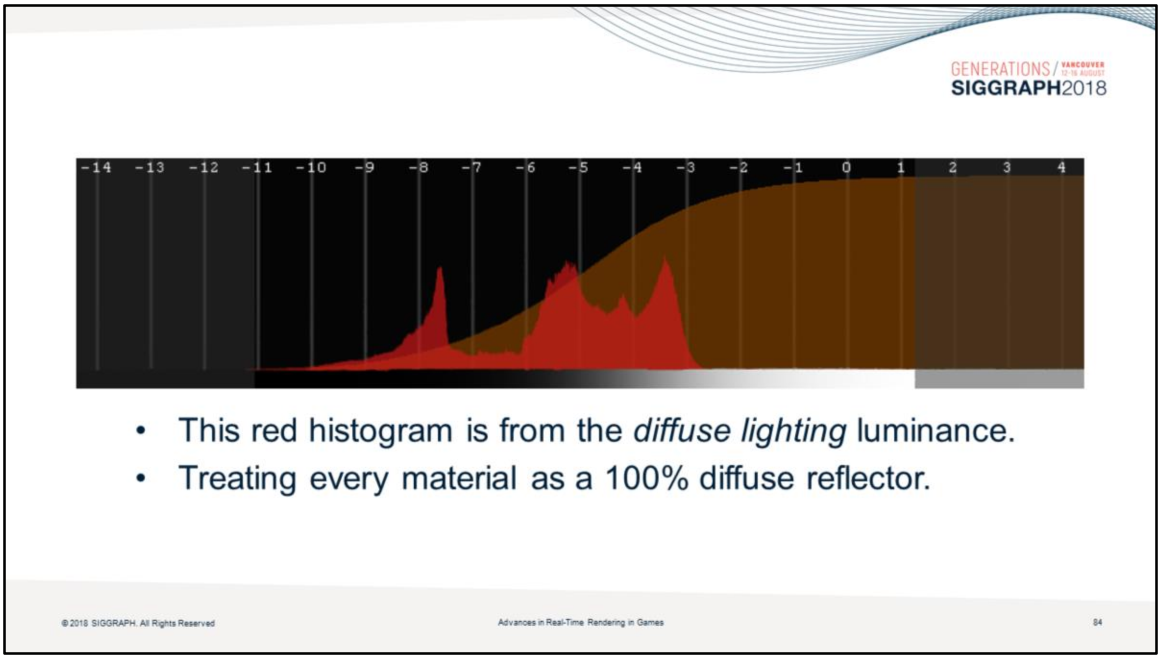


Let’s take a look at a histogram of the scene to see if 4 stops is enough contrast at night.

- EVs are displayed along the top.
- The gradient at the bottom is the luminance of that EV on screen, after tone mapping.
- The greyed out areas at either end are where we clip.

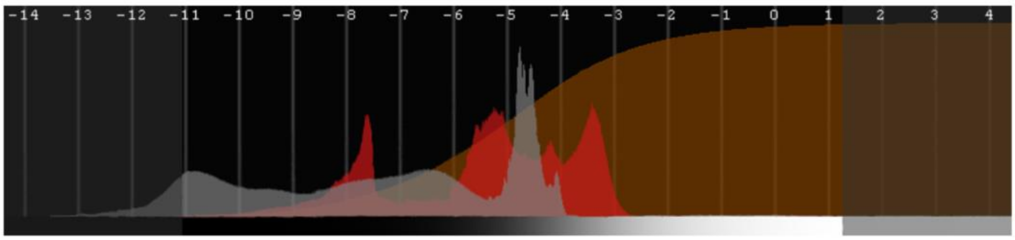


- This orange area is the shape of the tone mapping curve.

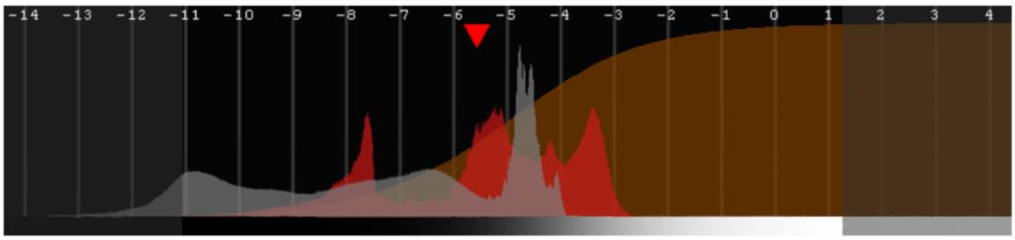


So typically we do diffuse lighting * diffuse albedo, well, this is as if we didn't do that multiply and we just take the luminance of the lighting.

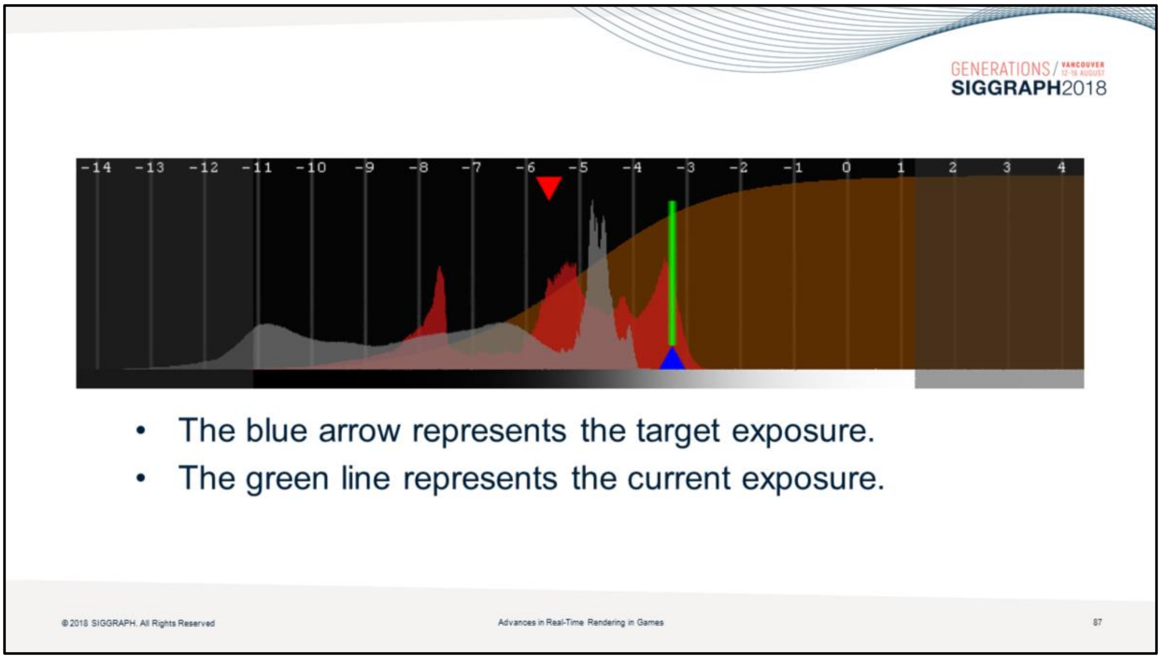
We calculate exposure from lighting luminance only, so this is why we have that information.



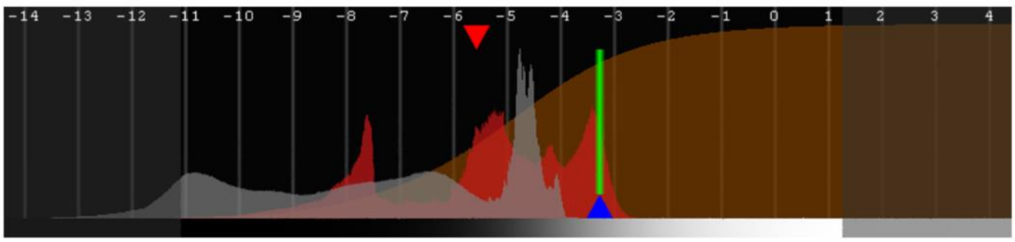
- The grey histogram is the final scene luminance.



- The red arrow represents the auto-exposure calculation.

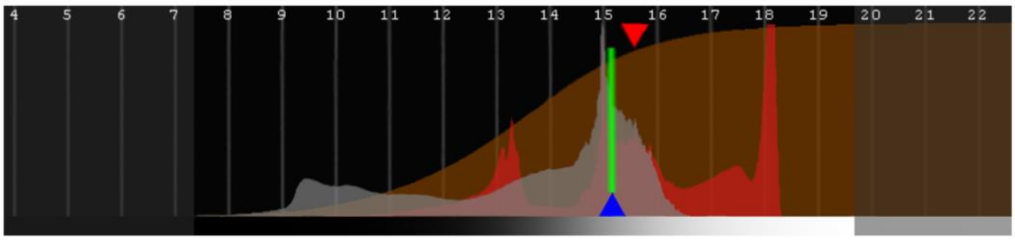


As I’m looking at a static image while capturing this histogram, the target and current exposure are the same. For dynamic images, we adapt over time to the target exposure.



- At night, we underexposure to make the image look dark.
- The contrast between moon and sky light is compressed into the bottom end of the histogram.

Also notice that we're clipping quite significantly in the bottom end. ☹



- At day, we have the same contrast between sun and sky lighting.
- But we slightly overexpose, so the range is better used.

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

~13 stops of contrast

Light Type	Distance (m)	Illuminance (lux)
Full moon, direct light	n/a	0.07
Full moon, sky light	n/a	0.004
500 lumen bulb	0.1	50,000
500 lumen bulb	1	500
500 lumen bulb	100	0.05
5000 lumen street light	1000	0.005

But it gets worse…

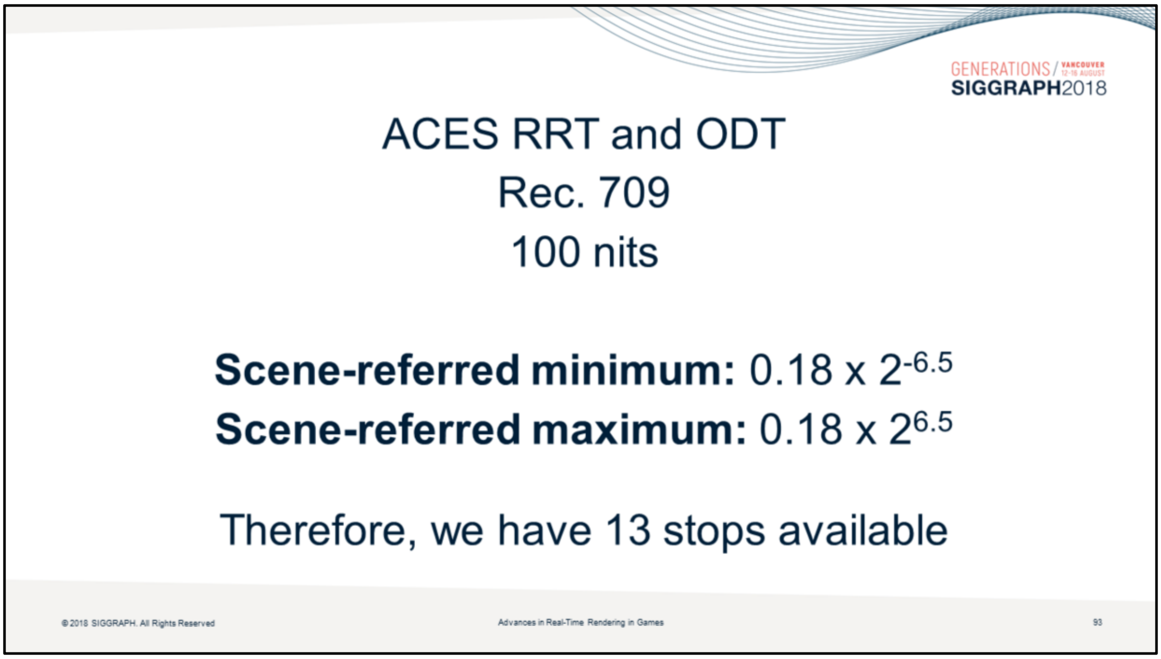
GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

~17 stops of contrast

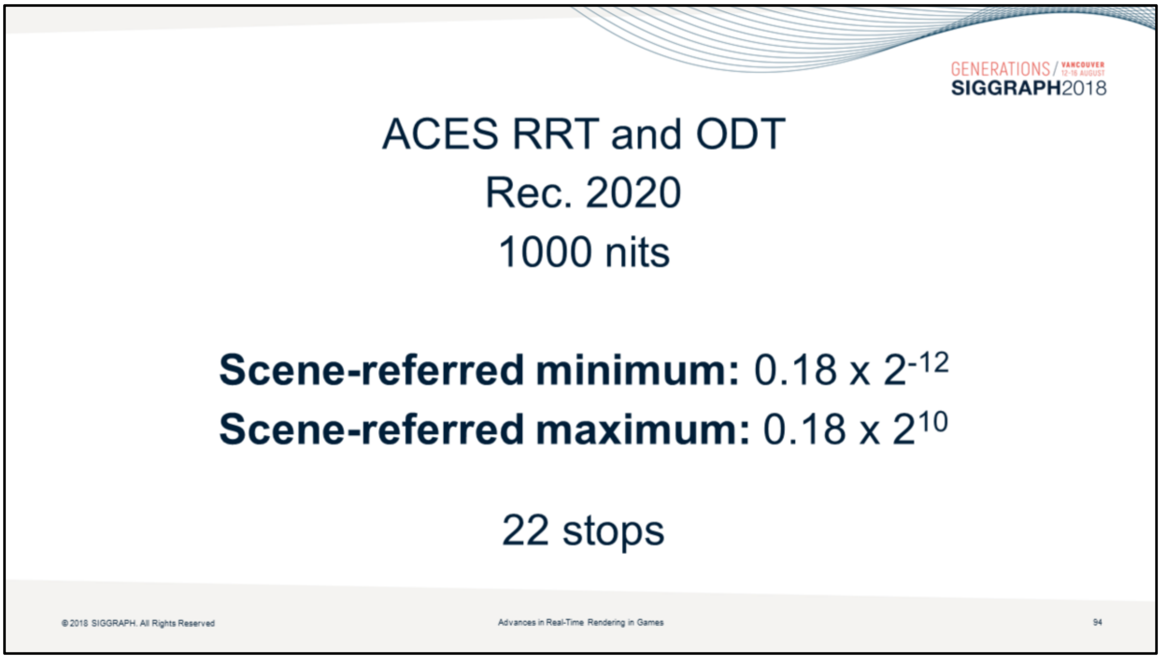
Light Type	Distance (m)	Illuminance (lux)
Full moon, direct light	n/a	0.07
Full moon, sky light	n/a	0.004
500 lumen bulb	0.1	50,000
500 lumen bulb	1	500
500 lumen bulb	100	0.05
5000 lumen street light	1000	0.005

~24 stops of contrast

Light Type	Distance (m)	Illuminance (lux)
Full moon, direct light	n/a	0.07
Full moon, sky light	n/a	0.004
500 lumen bulb	0.1	50,000
500 lumen bulb	1	500
500 lumen bulb	100	0.05
5000 lumen street light	1000	0.005



Only 13 stops! This isn't going to be enough to hold the full range of contrast we need at night time. And some of those stops will be pretty compressed, so it's no guarantee we'll have usable detail there.



But it’s worth noting that if we move to HDR, suddenly we have more stops available to us. Again, it’s unclear how many of those are “useful” but on Far Cry 5 we definitely noticed the increase in detail in bright and dark areas compared with SDR that made the game much more playable.

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

ACES RRT and ODT
Rec. 2020
2000 nits

Scene-referred minimum: 0.18×2^{-12}

Scene-referred maximum: 0.18×2^{11}

23 stops

ACES RRT and ODT
Rec. 2020
4000 nits

Scene-referred minimum: 0.18×2^{-12}
Scene-referred maximum: 0.18×2^{12}

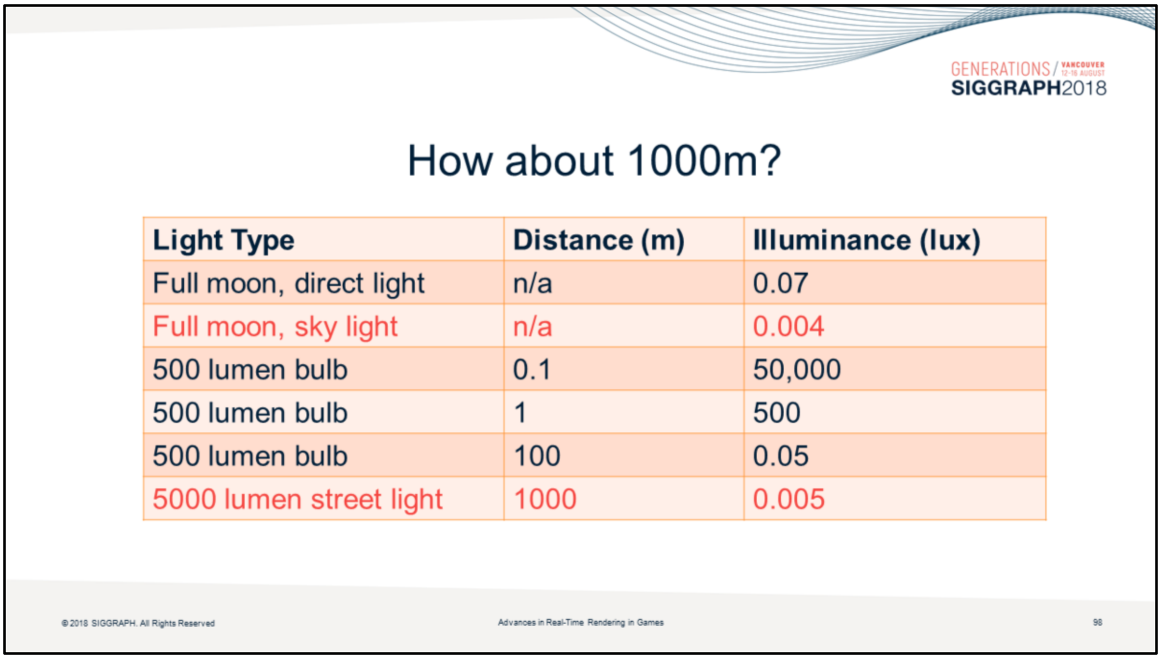
24 stops

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

100m light radius anyone?

Light Type	Distance (m)	Illuminance (lux)
Full moon, direct light	n/a	0.07
Full moon, sky light	n/a	0.004
500 lumen bulb	0.1	50,000
500 lumen bulb	1	500
500 lumen bulb	100	0.05
5000 lumen street light	1000	0.005

Another huge problem at night is how we clamp our light radiuses. Let’s compare some lighting values – full moon light is barely brighter than the light from a 500 lumen bulb 100m away.



And the light from a 5000 lumen street light 1000m away is brighter than the sky light!

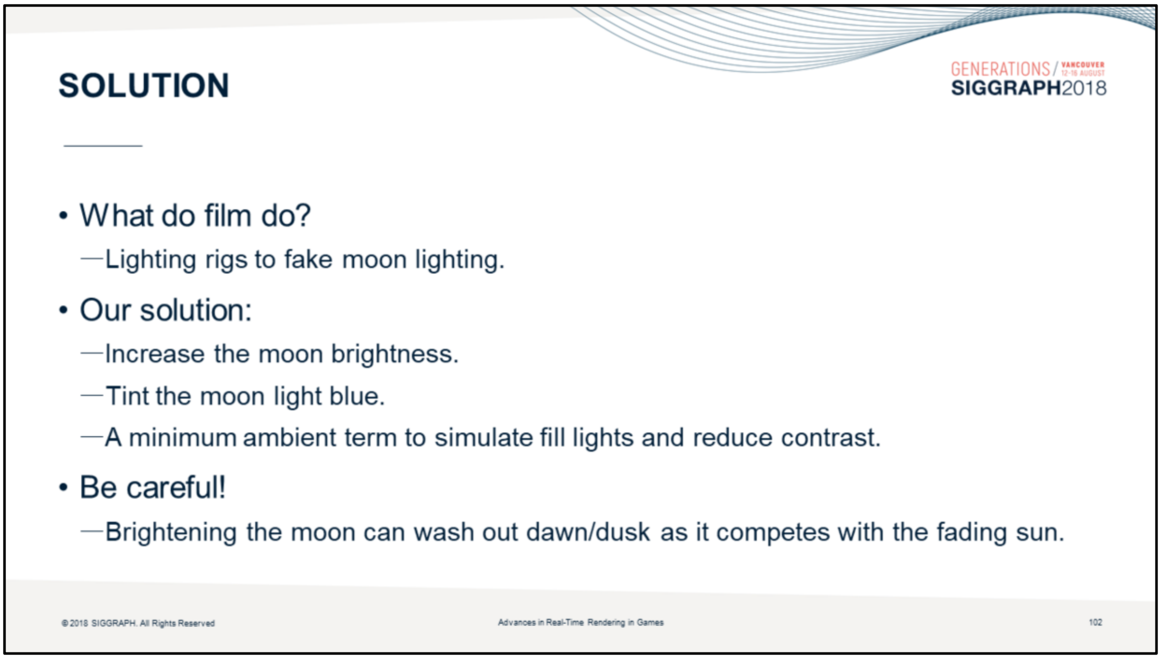

SIGGRAPH 2018
"The Challenges of Rendering
an Open World in Far Cry 5"
Stephen McAuley


5000 lumen light, ~100m behind the camera
SIGGRAPH 2018
Stephen McAuley

CONCLUSION

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

- We cannot support the contrast range of physical lighting values at night.
- Clamping light radiuses only increases the contrast range.
- Lighting artists darken lights to support night.
  - Causes incorrect behaviour at other times of day.



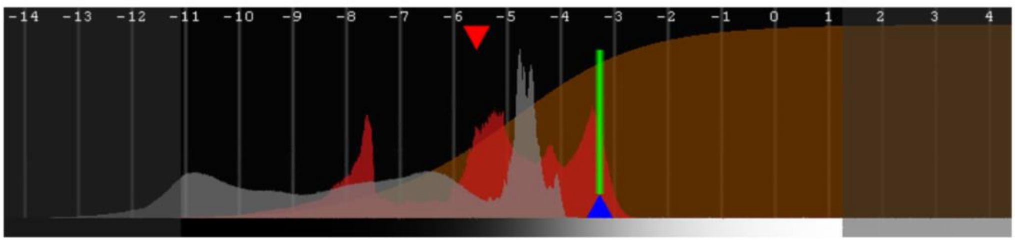
This increase of the moon's brightness is also why we using the Purkinje effect to simulate night's blue look would have been risky for us – we couldn't necessarily have used its correct physical values and it might have influenced too much of the scene.


FALL'S END
GARAGE
Complete
Automotive Service
WHEEL
ALIGNMENTS
Before
SIGGRAPH 2018
"The Challenges of Rendering
an Open World in Far Cry 5"
Stephen McAuley

Here you can see the moon and local lighting come into the same range, as well as contrast in the shadows being reduced. You can also see the blue moon tint as well, another difference between the images.

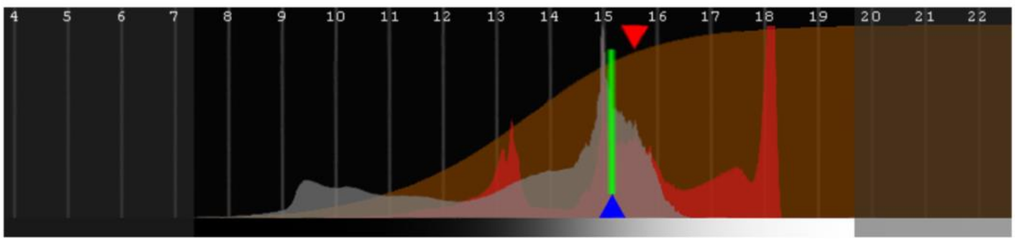
ONE MORE THING…

- Let's return to those histograms…
- At night, our range is roughly -11 to +1 EVs.



ONE MORE THING…

- At day, our range is roughly 7 to 20 EVs.



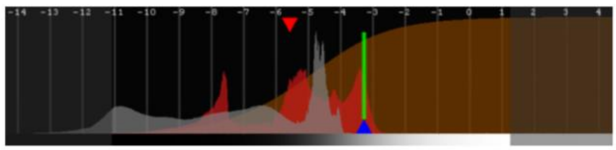
EMISSIVE OBJECTS

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

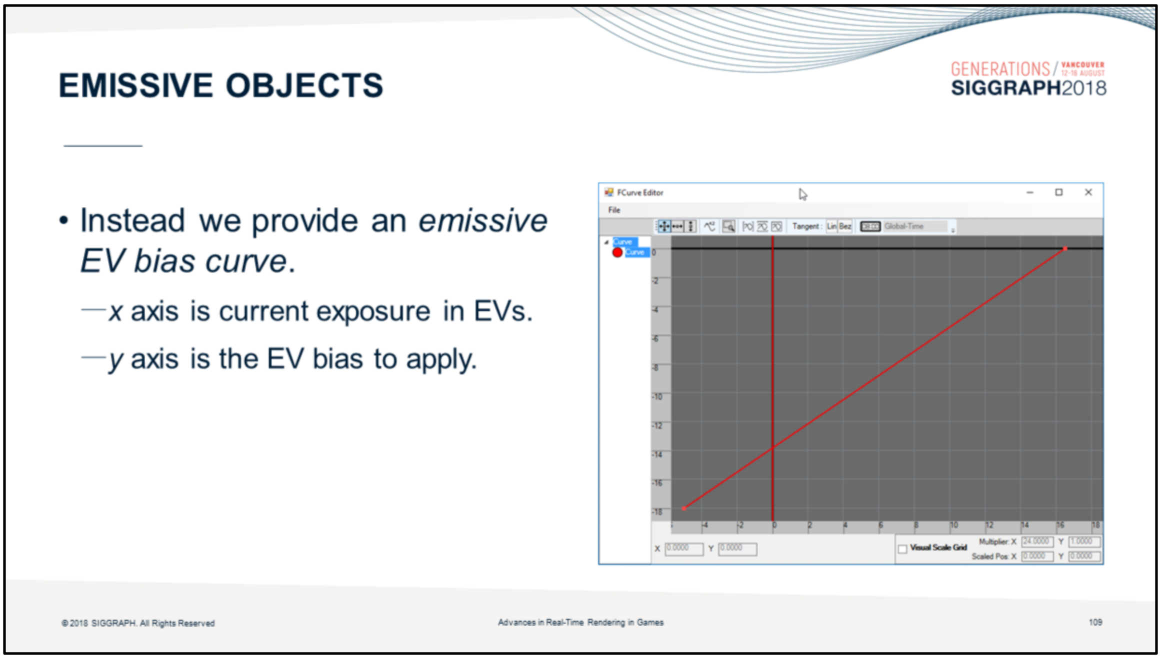
- What happens if you place an emissive object in the scene with a brightness of 2 to 6 EVs?
  - Pure black/invisible at day.
  - Pure white at night.
- This makes it incredibly hard to balance particle effects.

EMISSIVE OBJECTS

- The problem is that emissive objects don’t automatically emit light.
  - A light of 4 EVs in this scene would bring up the entire histogram:



- Not always easy or possible for all emissive objects to be lights.



You can see that our VFX artists chose to calibrate all their effects at day time, where exposure is around 16 EVs. Then nothing will happen to their particle effects. As exposure decreases, the EV bias linearly decreases. I was hoping they could tolerate more of a difference in effect brightness between day and night, but I think that was a hard mindset to change.

This is also evidence that night is the real problem, not day. First, I suggested they calibrated all effects at night time, then set the bias for day, but instead they went the other way around… they sensed that day time felt more "correct" by default.

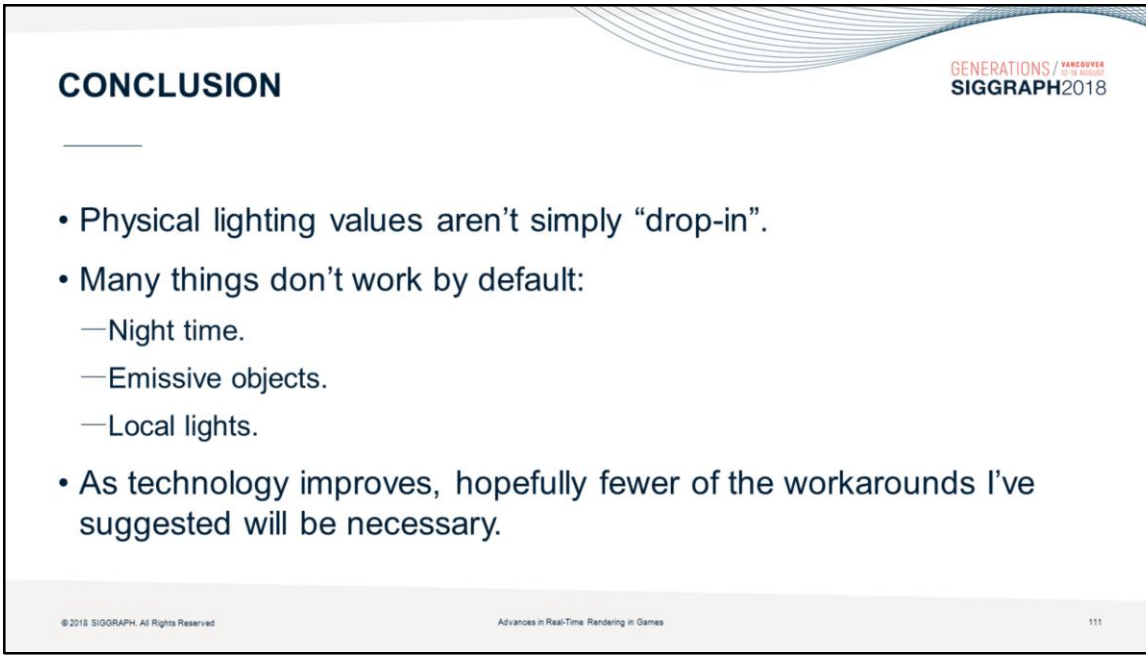
EMISSIVE OBJECTS

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

- Algorithm:
  - Given current exposure, look up emissive EV bias in curve.
    - Exposure is read back on CPU.
    - EmissiveEVBias is passed up to shaders as a constant.
  - Bias emissive lighting by that value.

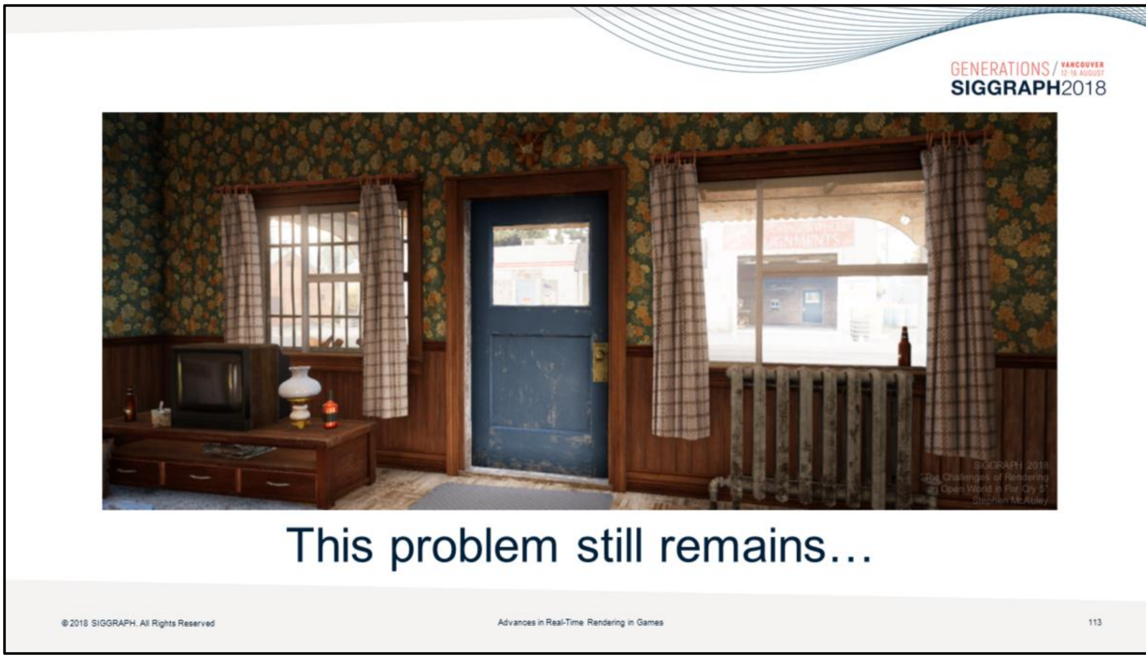
```
float emissiveIntensity = EV100ToLuminance(EmissiveEV + EffectsEmissiveEVBias);
outputColor.color0      = EmissiveColor * emissiveIntensity;
```


The artist sets EmissiveEV and EmissiveColor, and we bias the EmissiveEV by our EffectsEmissiveEVBias. We also have to convert EVs to luminance.



Also, please don't take anything I've suggested here as gospel. Maybe you have a better idea of how to fix the problems, or a different experience of the issues. Our workarounds have enabled us to ship a game and we're very happy about that, but we're definitely going to look at changing what we do and improving for future projects.

LOCAL TONE MAPPING



The contrast between interiors and exteriors is really hard for gameplay.

LOCAL TONE MAPPING

- Don’t want to break physical correctness by hacking lighting ratios.
- Local tone mapping is the answer:
  - Tone map different regions of the image differently.
  - Reduce bright areas to bring them into a smaller dynamic range.
- Bart Wronski wrote two excellent blog posts on the subject:
  - https://bartwronski.com/2016/08/29/localized-tonemapping/
  - https://bartwronski.com/2016/09/01/dynamic-range-and-evs/

POST-PROCESSING LOCAL TONEMAPPING

- First attempt was a post-processing approach.
- Based on Bart Wronski's shadows/highlights method:
  - https://bartwronski.com/2016/08/29/localized-tonemapping/

POST-PROCESSING LOCAL TONEMAPPING

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

- Global exposure:
  - Compute the average luminance of the entire screen
- Local exposure:
  - Compute the average luminance of different regions of the screen

POST-PROCESSING LOCAL TONEMAPPING

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

- Algorithm:
  - —Calculate scene log luminance in a full-resolution buffer
  - —Very large bilateral blur
    - Bilateral based on colour, not depth
  - —Apply local exposure
    - Sample buffer to obtain local exposure value

Now, this is the ideal algorithm…

POST-PROCESSING LOCAL TONEMAPPING

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

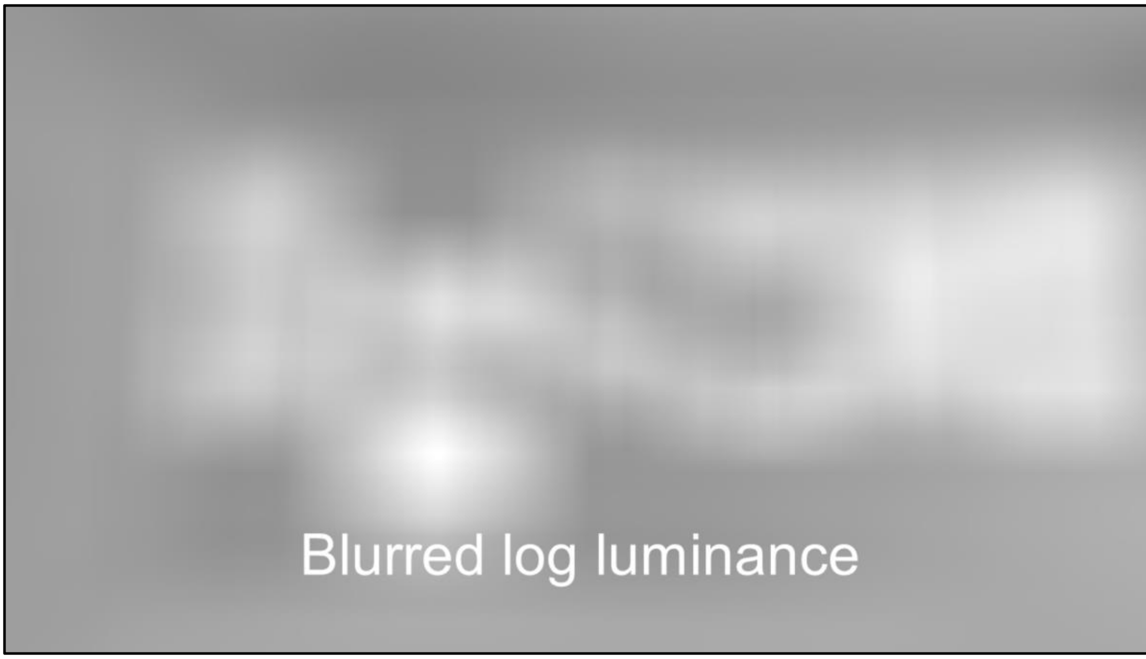
- Algorithm:
  - —Calculate scene log luminance in a low-resolution buffer
  - —Large blur
  - —Apply local exposure
    - Sample buffer to obtain local exposure value

We can’t afford the good version, sadly. ☹ At low resolution, the extra-cost of the bilateral blur is a bit pointless, in fact, it’s going to sharpen haloes that you see rather than soften them.


CLOSED
SIGGRAPH 2018
"The Challenges of Rendering
an Open World in Far Cry 5"


Scene luminance


Low-resolution log luminance


Blurred log luminance


Reducing the highlights
CLOSED
SIGGRAPH 2018
"The Challenges of Rendering
an Open World in Far Cry 5"

Sadly, you get lots of artefacts from haloes. Some things are darkened when they shouldn't be, some things aren't darkened when they should be! The door and window frames in general also get darker.


Before
CLOSED
SIGGRAPH 2018
"The Challenges of Rendering
an Open World in Far Cry 5"

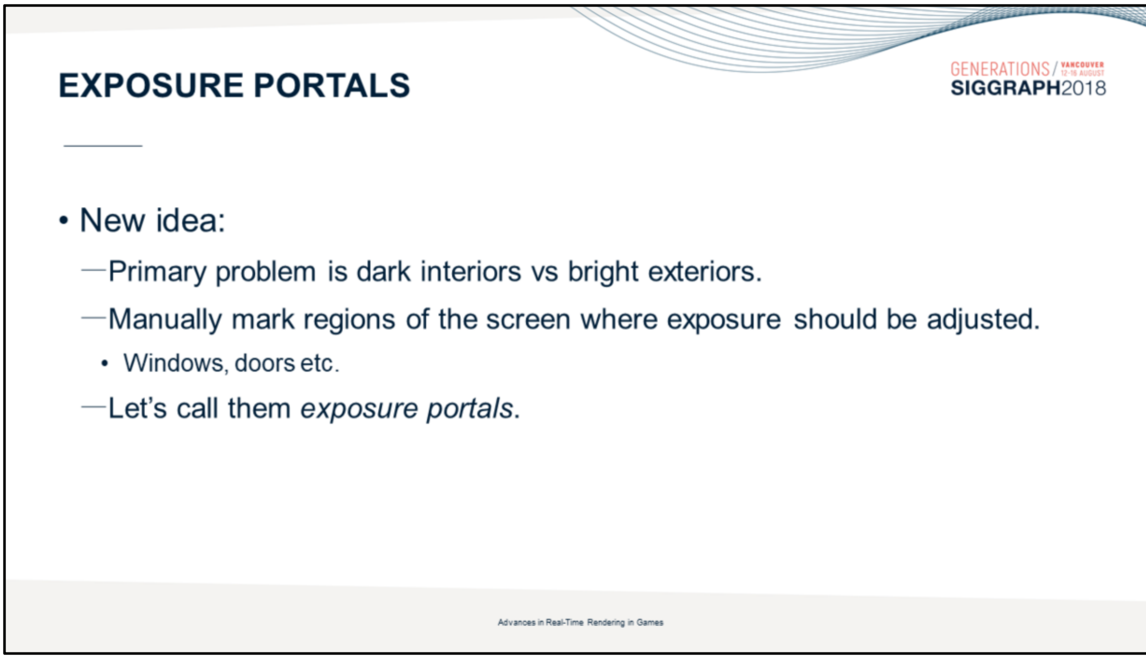
Maybe the haloes are subtle, but we deemed them unacceptable.

POST-PROCESSING LOCAL TONEMAPPING

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

- Pros:
  - Fast
  - Non-intrusive
- Cons:
  - Haloes

It’s really fast at a low resolution! But sadly, the haloes make this technique unusable, and we can’t afford the performance to make the bilateral version work.



If you think about it, the geometry is sort of taking the place of a bilateral blur, guiding where you'd want the blur to be.

Or another way, this is like editing a photo in Photoshop – you might manually mark the areas you want to adjust. We can do that, but in 3D space, by placing geometry.

EXPOSURE PORTALS

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

- Two-sided geometry.
- Multiplicative blending to darken or lighten what is behind.
- Fade out when close to the camera.


No Exposure Portals


Exposure Portals, EV Bias -1.0


Exposure Portal Locations


The Problem


Depth-Based Fade

EXPOSURE PORTALS

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

- Problems where objects enter and exit.
  - Even with a depth-based fade.
- Costly for artists to place.
- Sorting problems with other blended objects.
- Expensive to render each portal individually.
  - Needed for sorting reasons.

GI-BASED LOCAL TONE MAPPING

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

- Our friends from Assassin’s Creed Origins came to the rescue:

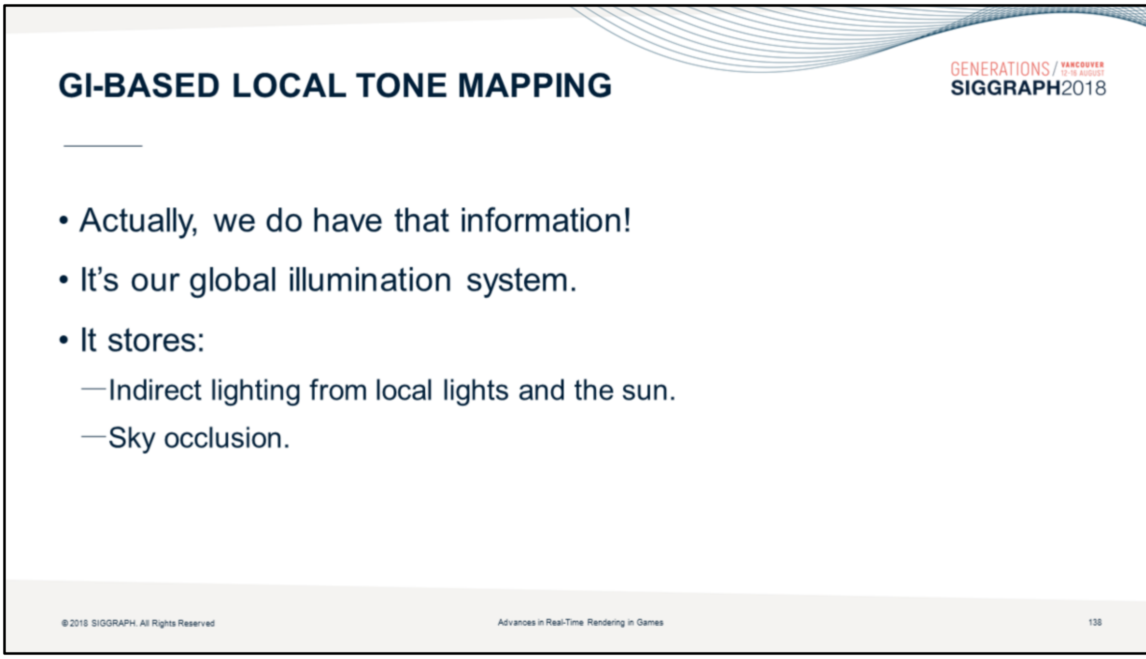
 Advances in Real-Time Rendering in Games 136

Thanks to Ulrich Haar for coming up with this technique!

GI-BASED LOCAL TONE MAPPING

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

- A bilateral blur is used in local exposure to:
  - Differentiate between areas of the screen that are:
    - Close in 2D space.
    - Far in 3D space.
  - Provide a local average of lighting values.
- What if we already had a local average of lighting values in 3D space?



Another way of looking at it is this: crudely, we want to differentiate between indoor and outdoor scenes, and sky occlusion is something that gives you that information.

GI-BASED LOCAL TONE MAPPING

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

- Algorithm:
  - —Create a reference middle grey from current scene exposure.
  - —Calculate average lighting luminance at current pixel:
    - Sky lighting plus indirect lighting.
    - Ignore all direct lights (including sun).
  - —Compare values and adjust pixel lighting accordingly.
    - Happens in all lighting shaders.

Part of the reason we ignore direct lights is that we want an AVERAGE of the lighting, and we'd have to calculate something special if we took direct lighting into account.

GI-BASED LOCAL TONE MAPPING

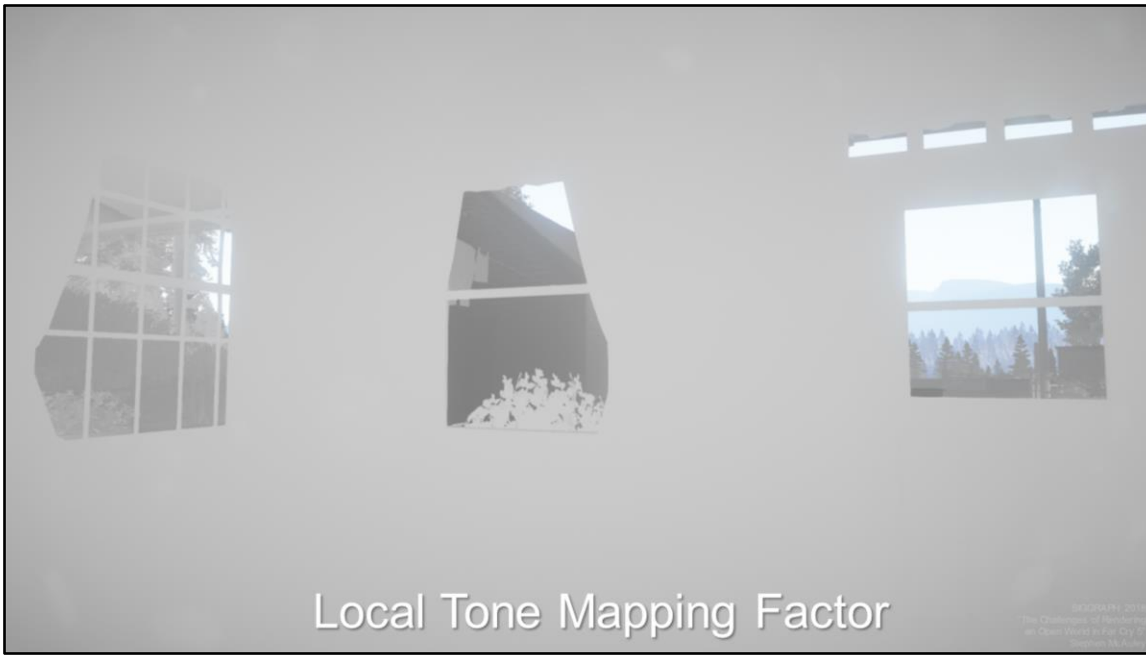
GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

```
float3 ambient;
float skyVisibilityDC;
float indirectDC;
SampleEvaluateGIAndSky(lightingInput.positionCS, lightingInput.normalWS,
                       ambient, skyVisibilityDC, indirectDC);

lightingOutput.localToneMappingScale =
    GetLocalToneMappingScale(skyVisibilityDC, indirectDC);

...

lightingOutput.diffuse  *= lightingOutput.localToneMappingScale;
lightingOutput.specular *= lightingOutput.localToneMappingScale;
```


Before
SIGGRAPH 2018
"The Challenges of Rendering
an Open World in Far Cry 5"
Stephen McAuley


GI
SIGGRAPH 2018
"The Challenges of Rendering
an Open World in Far Cry 5"
Stephen McAuley

Average means that we remove the directional component. We don’t want that much individual detail for local tone mapping.



We take that average luminance value, and create a local tone mapping factor from it. Note how outside is darker than inside.


Before
SIGGRAPH 2018
"The Challenges of Rendering
an Open World in Far Cry 5"
Stephen McAuley

Yes, it’s not a huge difference, but it doesn’t need to be. It solves our gameplay problems and our artistic problems, but still keeps a good interior/exterior contrast.

GI-BASED LOCAL TONE MAPPING

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

- Pros:
  - Robust with no artefacts
- Cons:
  - Intrusive
    - Needs to be added to all lighting shaders
  - Small extra cost
    - Extra ALU and VGPRs to all lighting shaders

FUTURE PLANS

- Also use local tone mapping to boost dark areas at night.
  - Hopefully removes the need to have a minimum ambient value.
- Revisit post-processing based local tone mapping.
  - Less intrusive.
  - Can we manage the haloes better?

Problem solved!

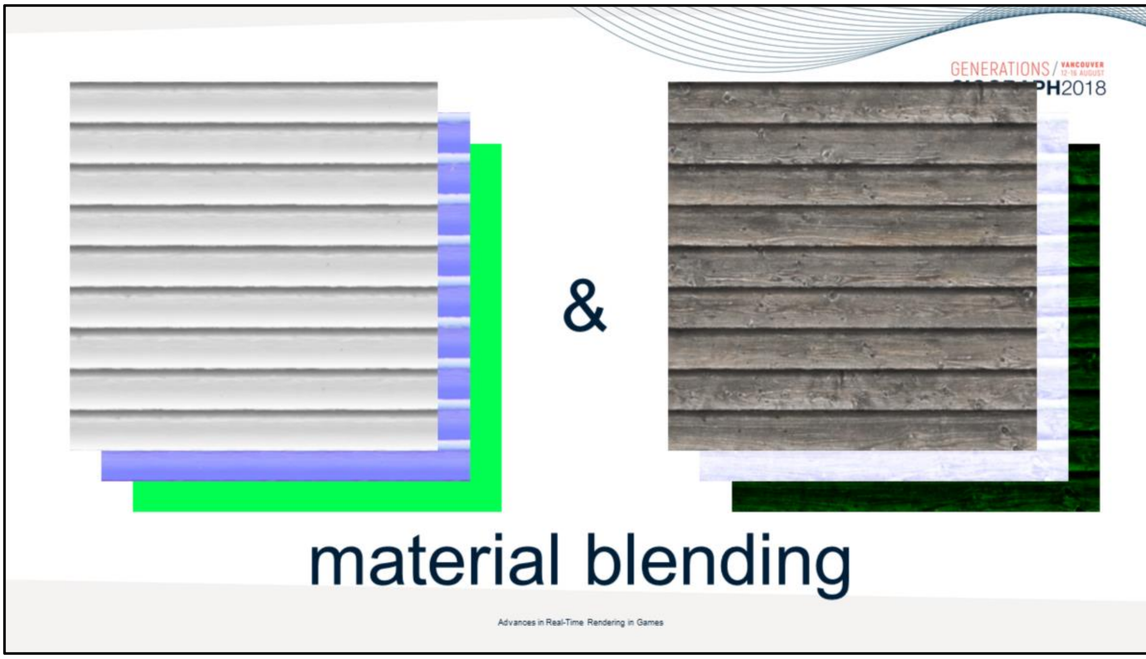
GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

ART PRODUCTION

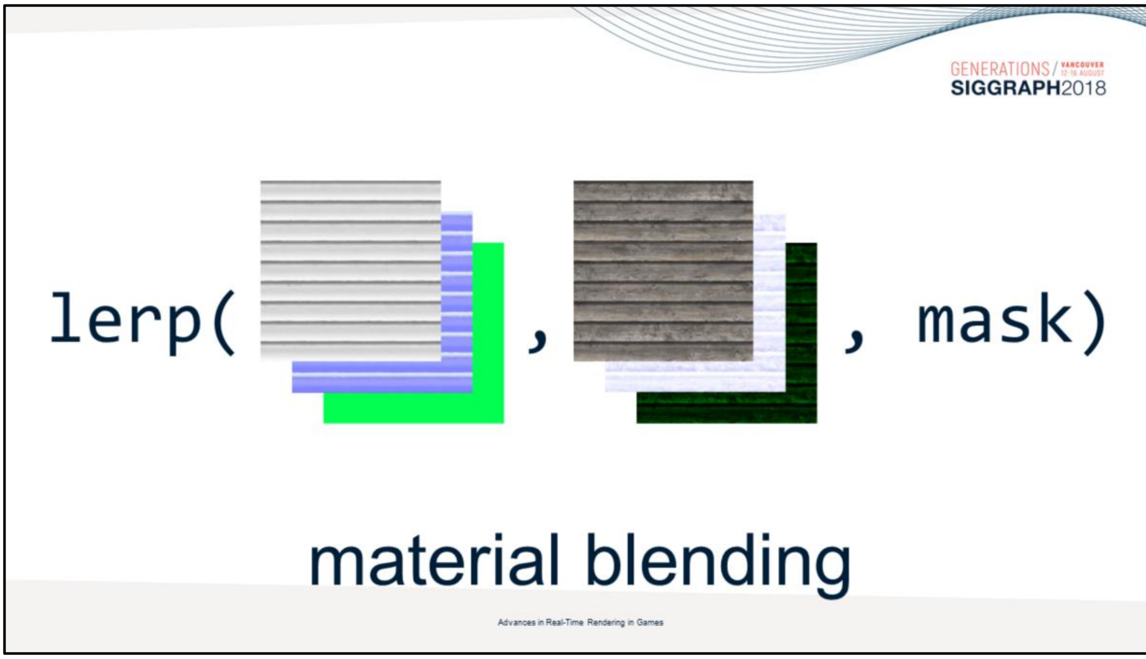
POPULATING AN OPEN WORLD

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

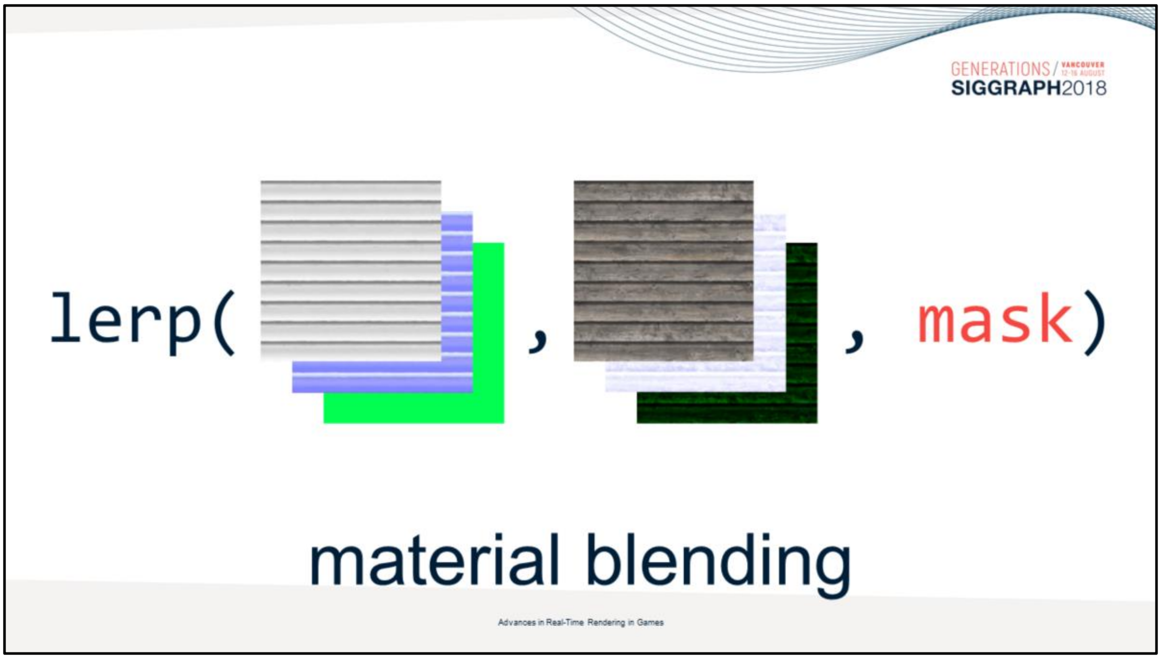
- A huge amount of content to produce, stream and display.
- Desire from art direction for many unique assets.
- First-person view means high texel density is required:
  - Aim for 6 texels/cm.
- Artists need to re-use textures.
- Material blending in shaders is mandatory.



Let’s use an example of someone wanting to blend between white painted wood and bare wood.

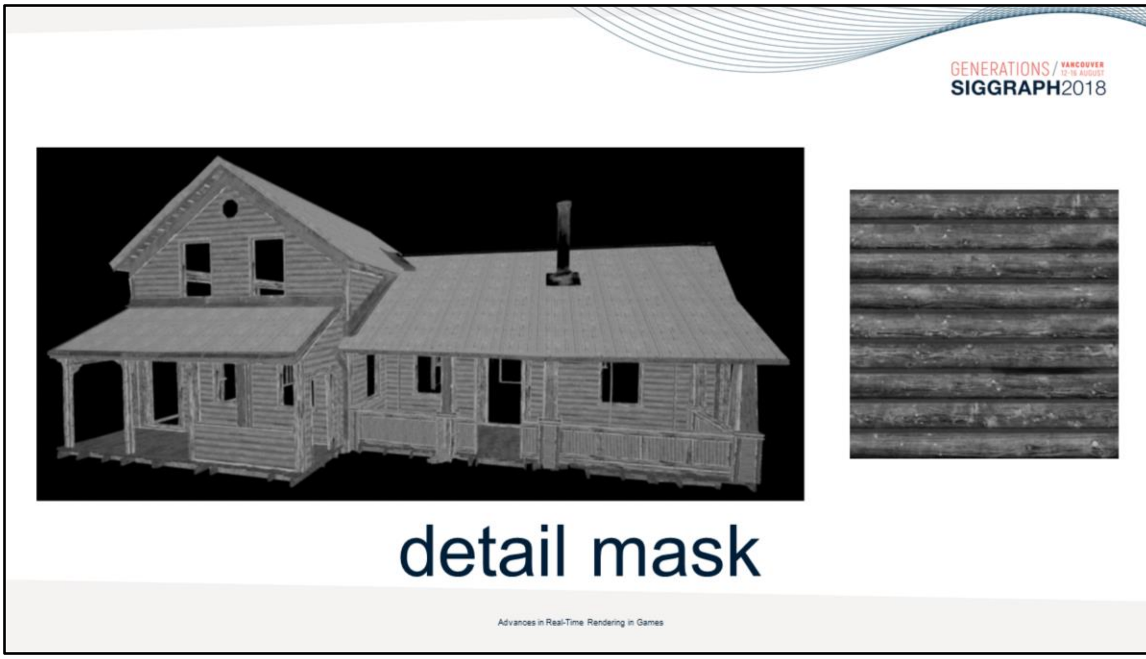


Typically we’d blend albedo, normal, material properties like this…

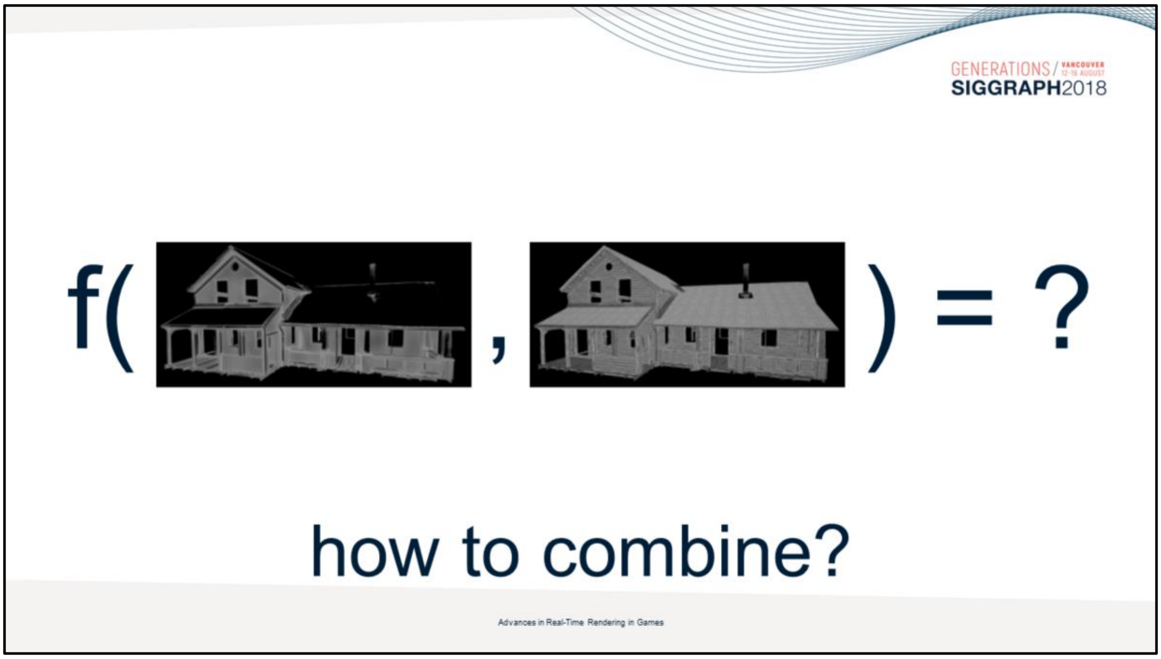


…but what does the mask that we use to blend actually consist of…

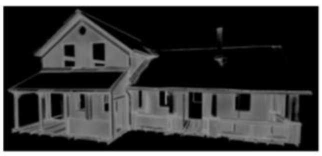
We’d like to use a unique mask for a building like this, giving the artists a lot of control and making the building look realistic, adding weathering features where you really think they’d be. However, it’s pretty low resolution.



So to supplement it, we provide a detail mask layered on top. Here it’s very much related to the structure of the wood, so the paint would say, flake off in the cracks between the wooden planks first.



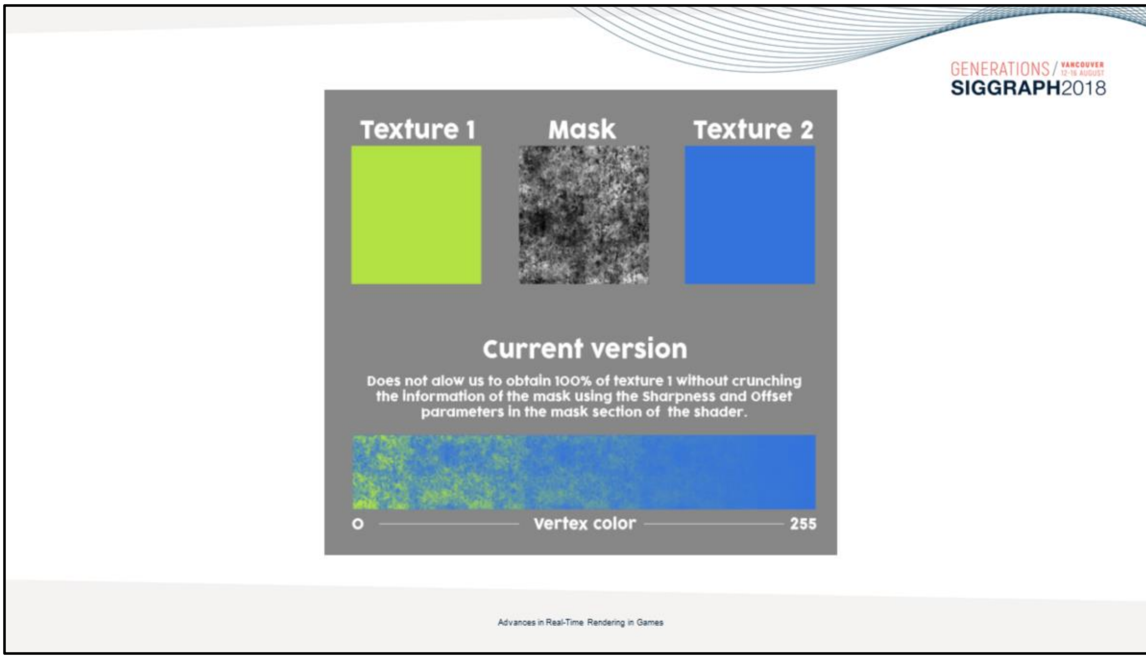
But of course, the real question is how to combine these masks in an easy way for artists.

 x  = 

multiply

unique mask	result
0	`material1`
1	`lerp(material1, material2, detailMask)`
?	`material2`

multiply



In fact, our artists pointed out to use how bad this is! I got this lovely little diagram in an e-mail explaining the problem.

unique mask	result
0	`material1`
0.5	`lerp(material1, material2, detailMask)`
1	`material2`

desired behaviour

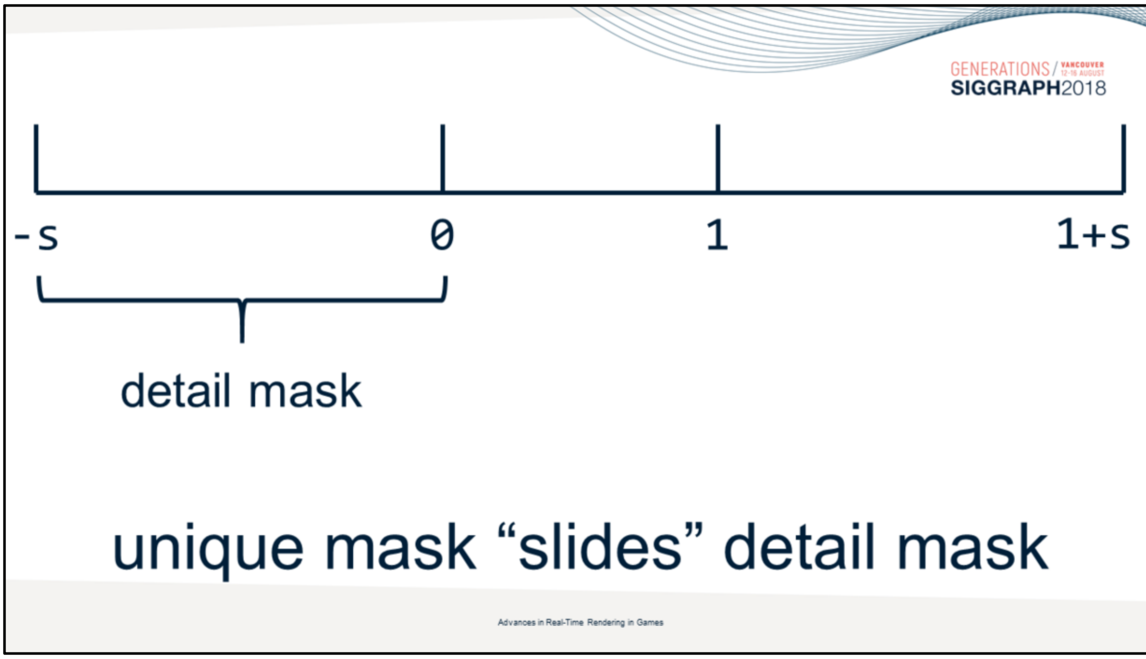
GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

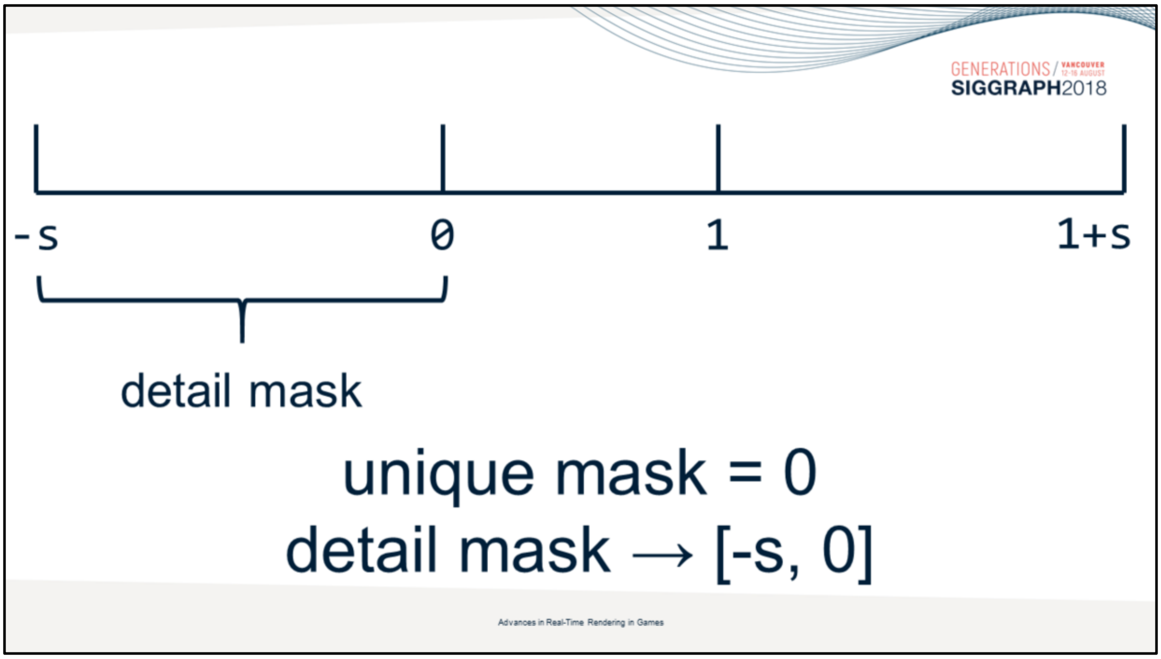
0.5 1.0 2.0

oh, let’s throw in “sharpness” s too

$$\underbrace{0 \qquad\qquad\qquad s}_{\text{detail mask}}$$

sharpness scales the detail mask

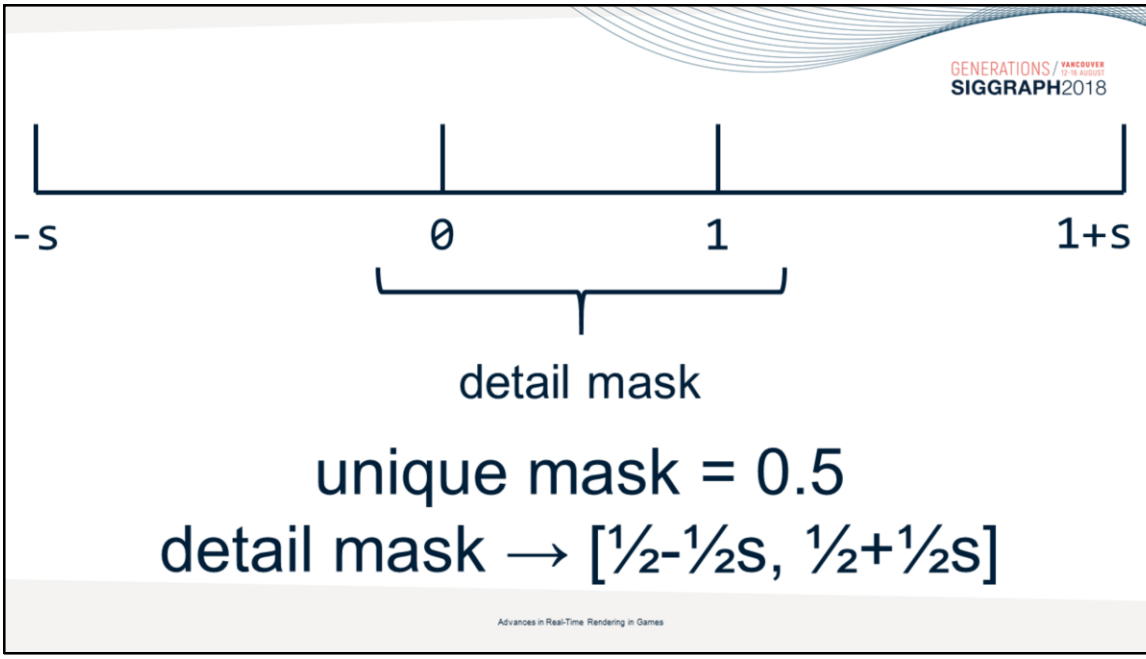

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018
-s
0
1
1+s
detail mask
unique mask “slides” detail mask
Advances in Real-Time Rendering in Games


GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018
-s
0
1
1+s
detail mask
unique mask = 0
detail mask → [-s, 0]
Advances in Real-Time Rendering in Games

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

unique mask = 1

detail mask → [1, 1+s]


GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018
-s
0
1
1+s
detail mask
unique mask = 0.5
detail mask → [½-½s, ½+½s]
Advances in Real-Time Rendering in Games

look for equation of the form

```
s x detail mask + m x unique mask + c
```


GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

unique mask	detail mask
0	[-s, 0]
1	[1, 1+s]

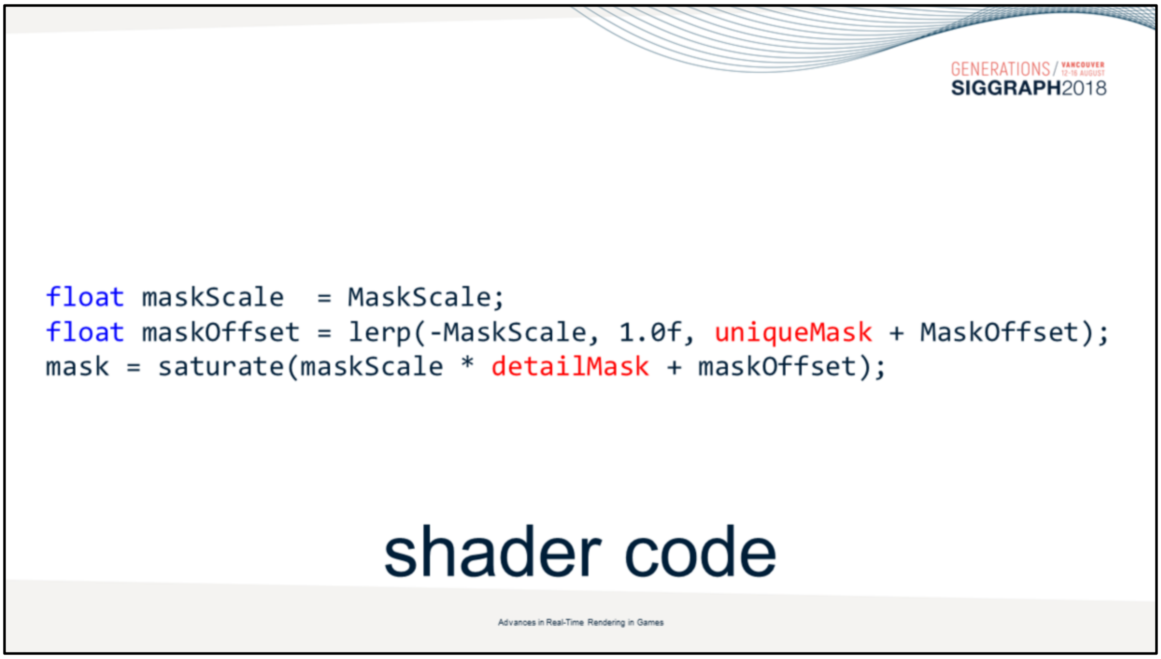
look for gradient and offset

unique mask	detail mask
0	[-s, 0]
1	[1, 1+s]

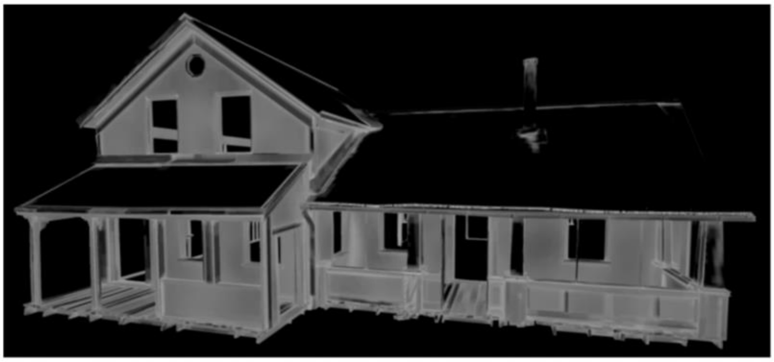
`m = 1 + s, c = -s`

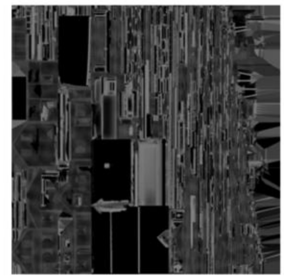
final equation

`s x detail mask + (1 + s) x unique mask - s`

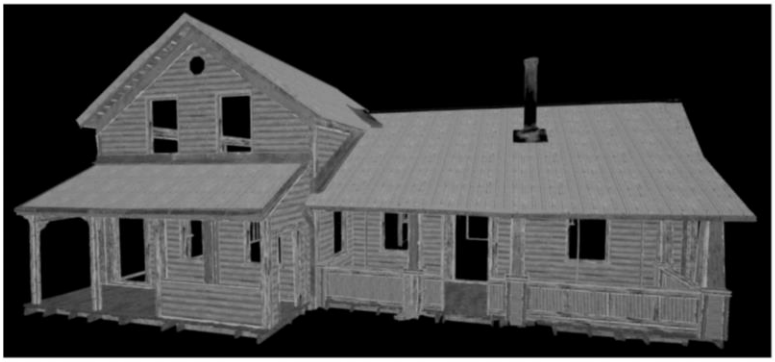


Notice the additional MaskOffset parameter. This can be used to bias the unique mask, in case we want to turn one material always on or off. That can be animated too – for example, to simulate natural-looking weathering over time.



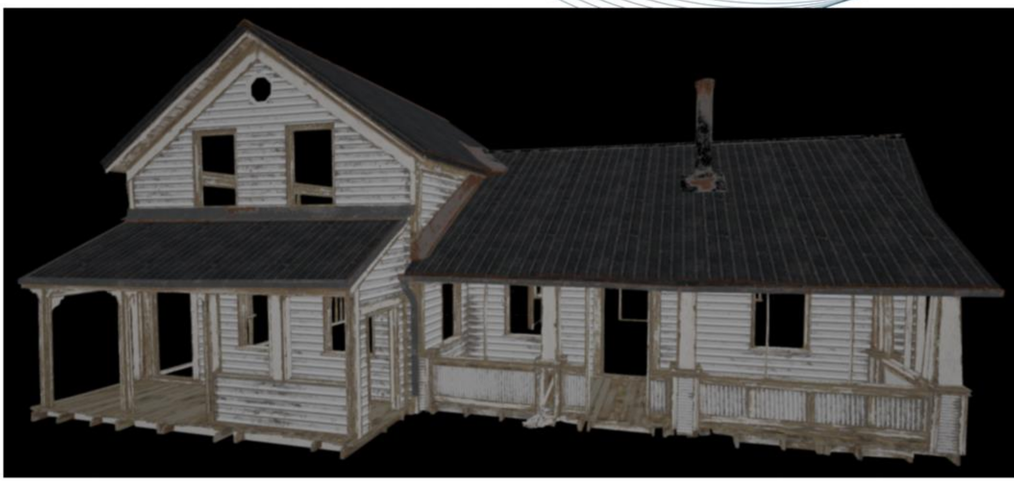


unique mask



detail mask

final mask



final albedo

final asset

MATERIAL BLENDING

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

- Unique look for many assets.
- High resolution material blending.
- Low(er) memory requirements for the masks.

COMPLEX PARAMETER SETTING

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

- Material blending and texture re-use can increase ALU.
- Let's take an example:
  - An artist would like to rotate UVs in the shader:

```
float sinTheta, cosTheta;
sincos(2.0f * Pi * Rotation / 180.0f, sinTheta, cosTheta);
float u_ =  cosTheta * u + sinTheta * v;
float v_ = -sinTheta * u + cosTheta * v;
```

COMPLEX PARAMETER SETTING

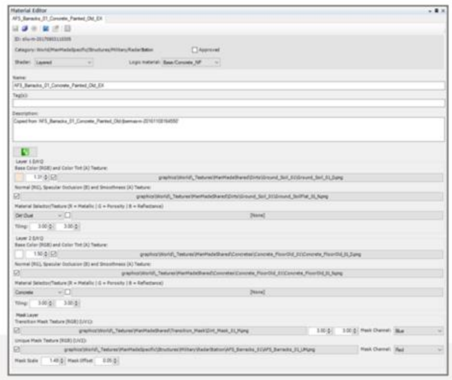
GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

- That sincos is expensive!
  - Plus we have to convert Rotation from degrees to radians.
- Can we precalculate the result?

```
float sinTheta, cosTheta;
sincos(2.0f * Pi * Rotation / 180.0f, sinTheta, cosTheta);
float u_ =  cosTheta * u + sinTheta * v;
float v_ = -sinTheta * u + cosTheta * v;
```

MATERIAL DESCRIPTORS

- Each material type has a material descriptor .xml file, describing:
  - —The shader to use
  - —The parameters to set
  - —The user interface



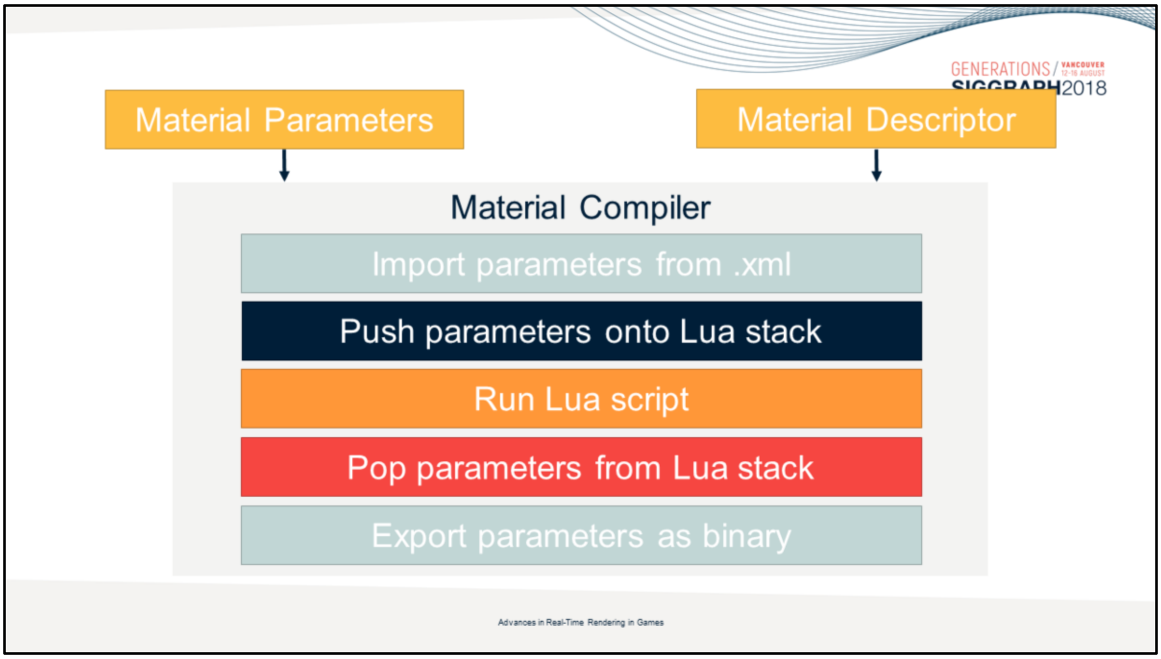
MATERIAL DESCRIPTORS

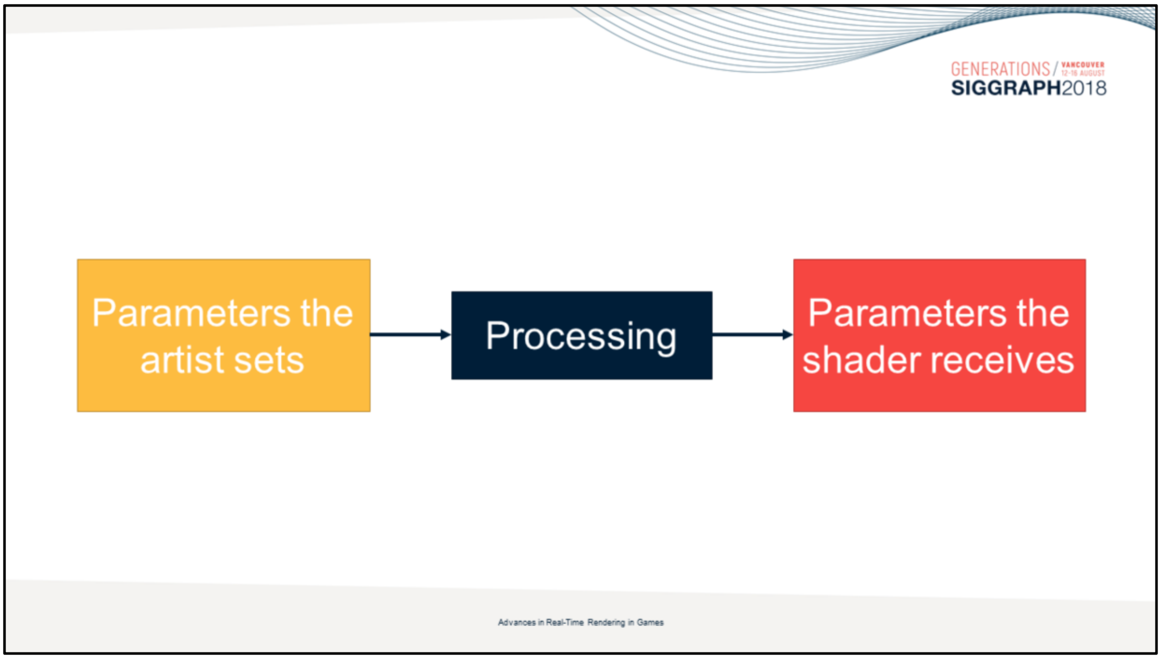
GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

- Add Lua script to our material descriptor:

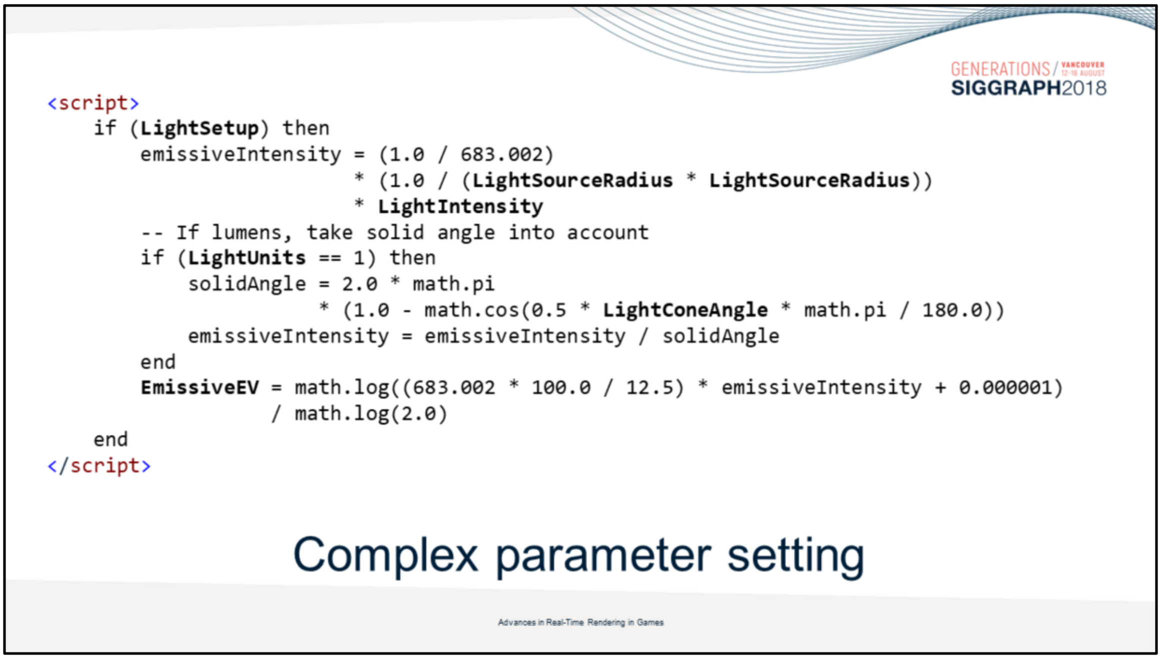
```
<script>
    SinTheta = math.sin(math.rad(Rotation))
    CosTheta = math.cos(math.rad(Rotation))
</script>
```

- The artist sets **`Rotation`**.
- The shader receives **`SinTheta`** and **`CosTheta`**.

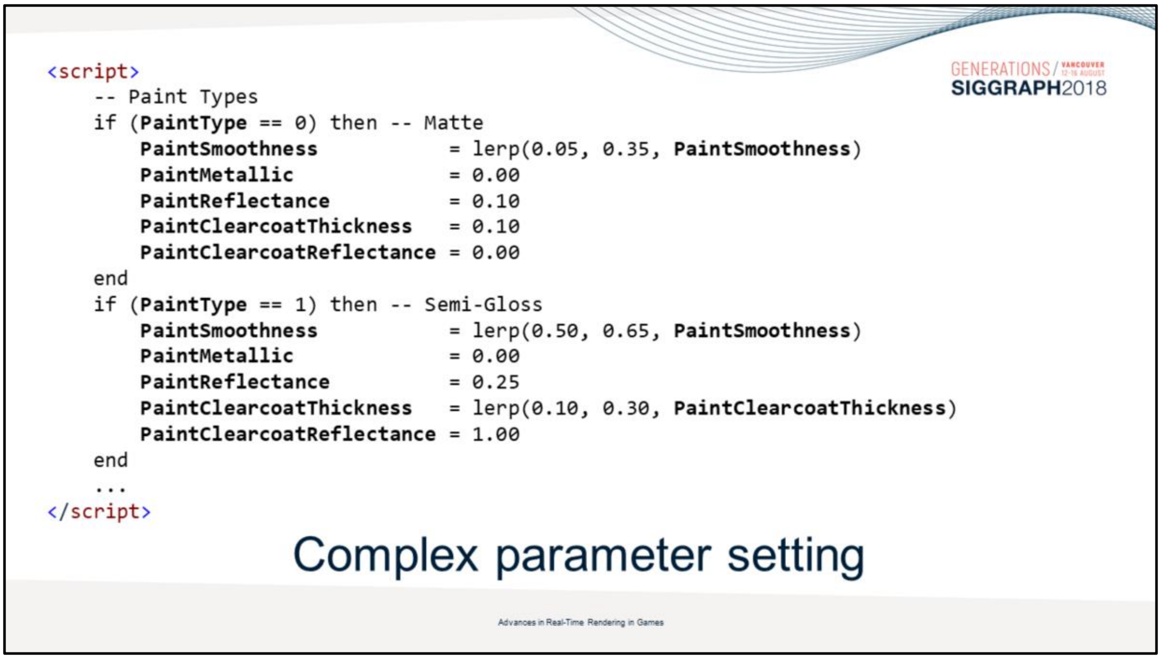

GENERATIONS / VANCOUVER 12-16 AUGUST
2018
Material Parameters
Material Descriptor
Material Compiler
Import parameters from .xml
Push parameters onto Lua stack
Run Lua script
Pop parameters from Lua stack
Export parameters as binary
Advances in Real-Time Rendering in Games



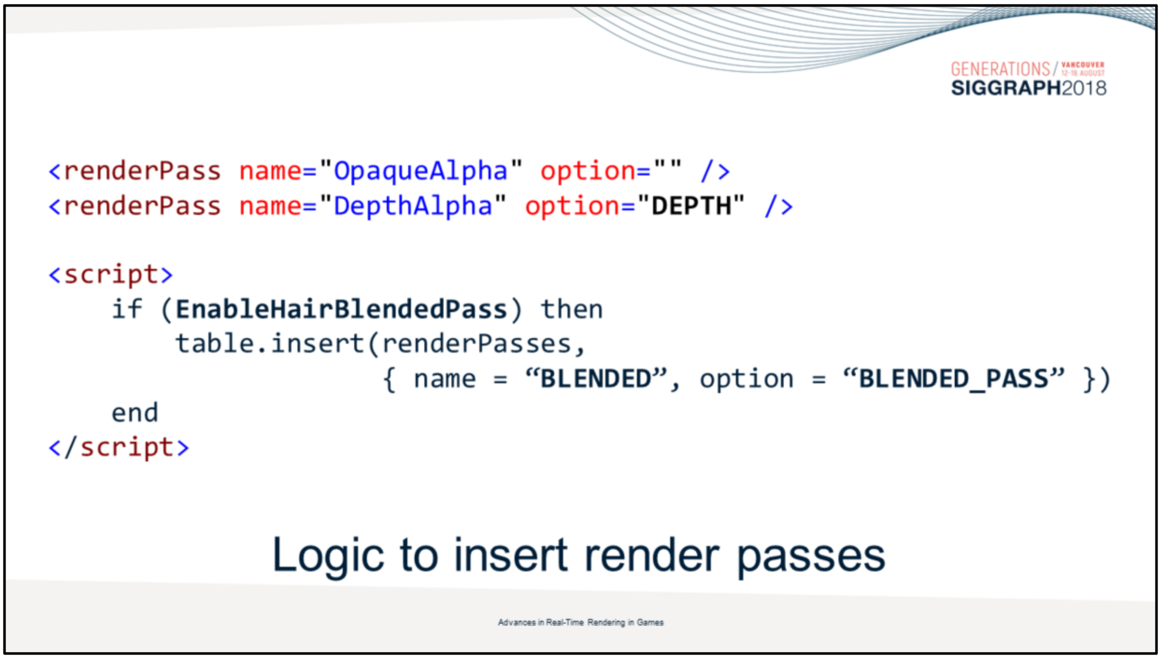
Or perhaps if we present this more abstractly, this is what is happening. If we think about it, there is no real need for the two to be the same.



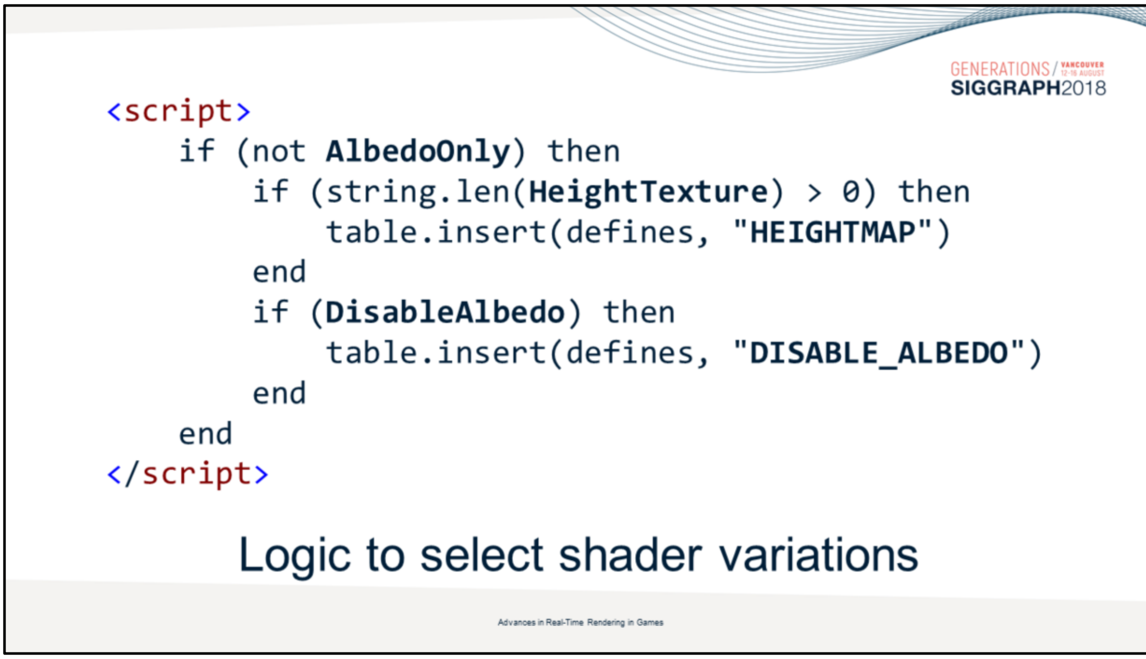
This is taken from an emissive shader, where we'd like the artists to optionally set it up with the same parameters they'd use for a light – for example, if this was an emissive material for a light bulb. They can choose the lighting units, the light source radius, the light intensity and cone angle, and we'll calculate an EmissiveEV value from it. The thing is, the shader only ever receives the emissive EV value, it doesn't need to know about any of the UI or setup which is pretty awesome.



Another example is from our car paint shader. Rather than set material properties directly, artists can use a drop down box to select a matte material, semi-gloss material or many more. The Lua script then fills in the output parameters with the correct material properties. Yes, there are probably more fancy and elegant ways of doing this, but this works and enabled our technical artist to improve the UI for the artists with no code or tools support.



We can also do more than just set parameters. Our material descriptors also describe what render passes to render for a material, like the old DX9 effect system. This is a hair shader, where we always render a depth pass, an opaque alpha-test pass and then optionally an alpha-blended pass for the edges of the hair. This optional render pass can be controlled by the artist in the material, and added via the Lua script. The code doesn’t need to know anything about it and this just happens in the Lua script.



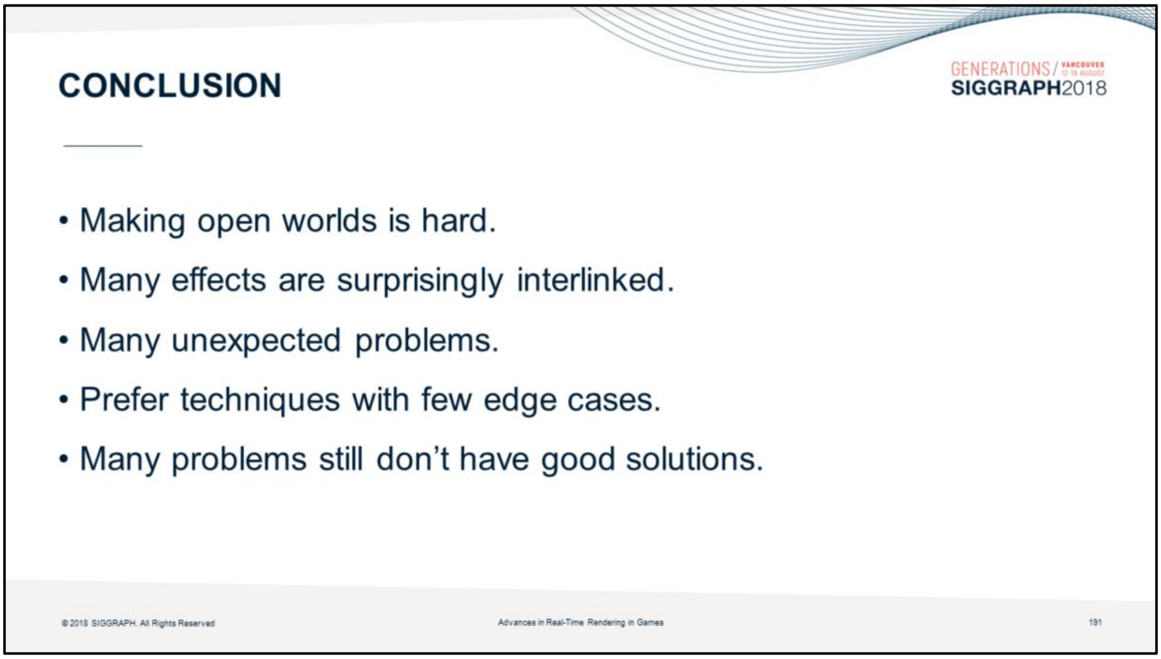
Or finally, we can have complex logic to select shader variations, like shown here for a decal shader. Depending on what tick boxes the artist selects, and what textures they provide, we can choose the right shader to use.

CONCLUSION

GENERATIONS / VANCOUVER 12-16 AUGUST
SIGGRAPH2018

- First-person open worlds also pose art production challenges.
- Think about usability for the artists:
  - Parameters that are easy to understand.
  - Parameters that respond in expected ways.
- Consider developing tools that can be used for multiple purposes:
  - The Lua script solves more than one problem.

CONCLUSION



It's also important to ensure that we chase down the real problems. Artists might come to us with solutions, like "darken the sun", but we have to discover the root of the issue that they're having.

THANKS

Nathalie Dubois
Jeremy Moore
Mickael Gilabert
Jean-Sebastien Guay
Olivier Painnot
Jean-François Tremblay
Jendrik Illner
Alexandre Ribard
Aurora Huang
Doug Clayton

Branislav Grujic
Mao Zhen Yu
Colin Weick
Pavlo Turchyn
Dmytro Rozovik
Anton Remezenko
Louis-Philippe Cantin
Rowan Clark
Ulrich Haar
Jean-Sebastien Guay

REFERENCES

[Grujic2018] Water Rendering in Far Cry 5, Branislav Grujic & Cristian Cutocheras, GDC 2018

[Jensen2001] A Physically Based Night Sky Model, Henrik Wann Jensen et. al., SIGGRAPH 2001

[Moore2018] Terrain Rendering in Far Cry 5, Jeremy Moore, GDC 2018

[Preetham1999] A Practical Analytic Model for Daylight, A. J. Preetham et. al., SIGGRAPH 1999

[Yapo2009] Rendering Lunar Eclipses, Theodore C. Yapo & Barbara Cutler, Graphics Interface 2009

Questions?