Physically Based Shading in Unity

Aras Pranckevičius
Rendering Dude

Outline

- New built-in shaders in Unity 5
    - What, how and why
- And all related things

Shaders in Unity 4.x

- A lot of good things are *available*
    - Marmoset Skyshop
    - Shader Forge
    - Alloy physical shaders
- A lot of good things are *possible*
    - If you know how to write shaders, that is

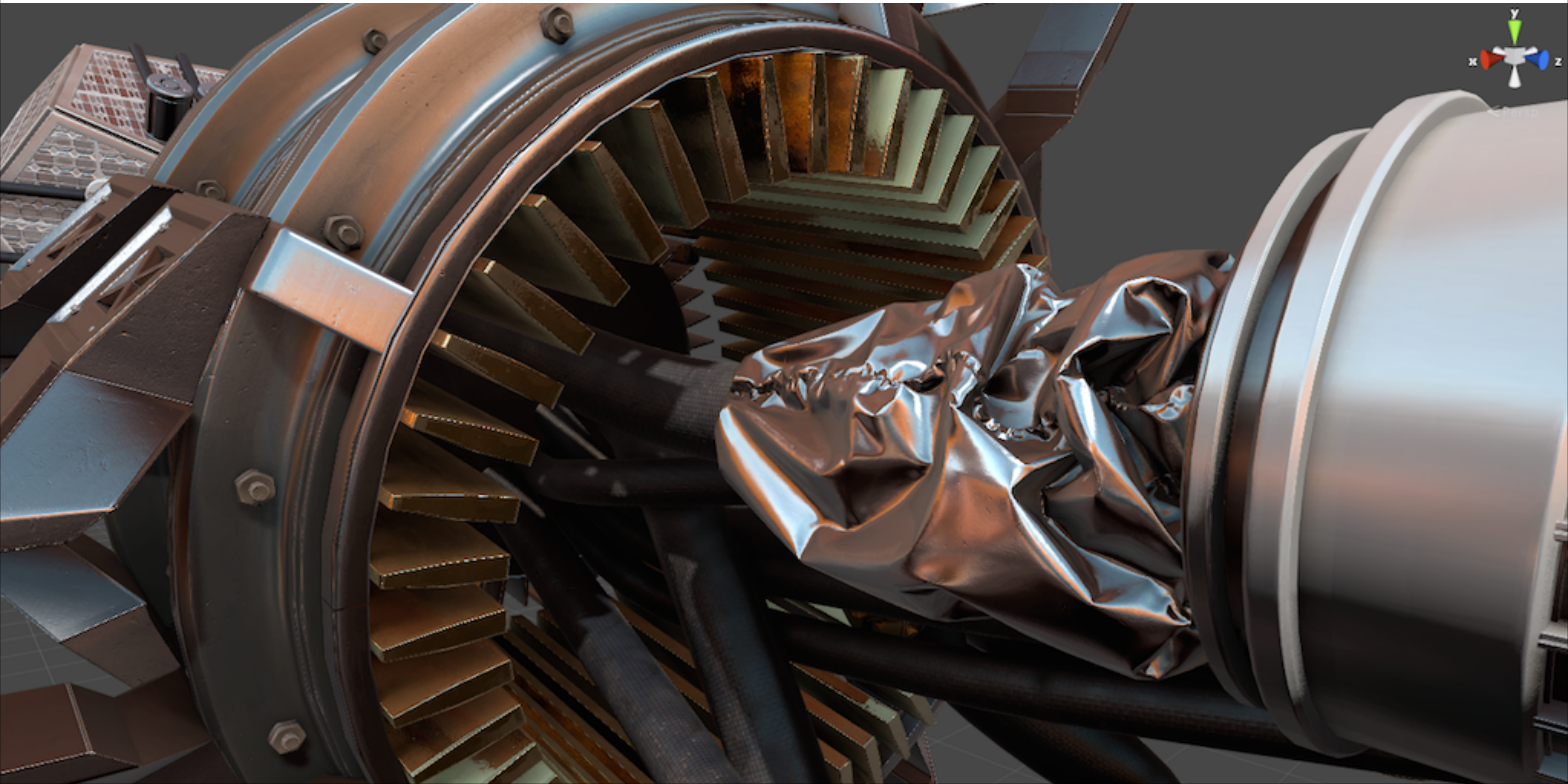
Want this

Done with one built-in shader

Shaders in Unity 5

- New built-in shaders
  - Physically based shading
  - Improved UI & workflow
- Related things
  - Deferred shading, Reflection probes, HDR, …
- Built-in could cover 80% of use cases
  - Still possible to write your own, like always

Physically Based Shading

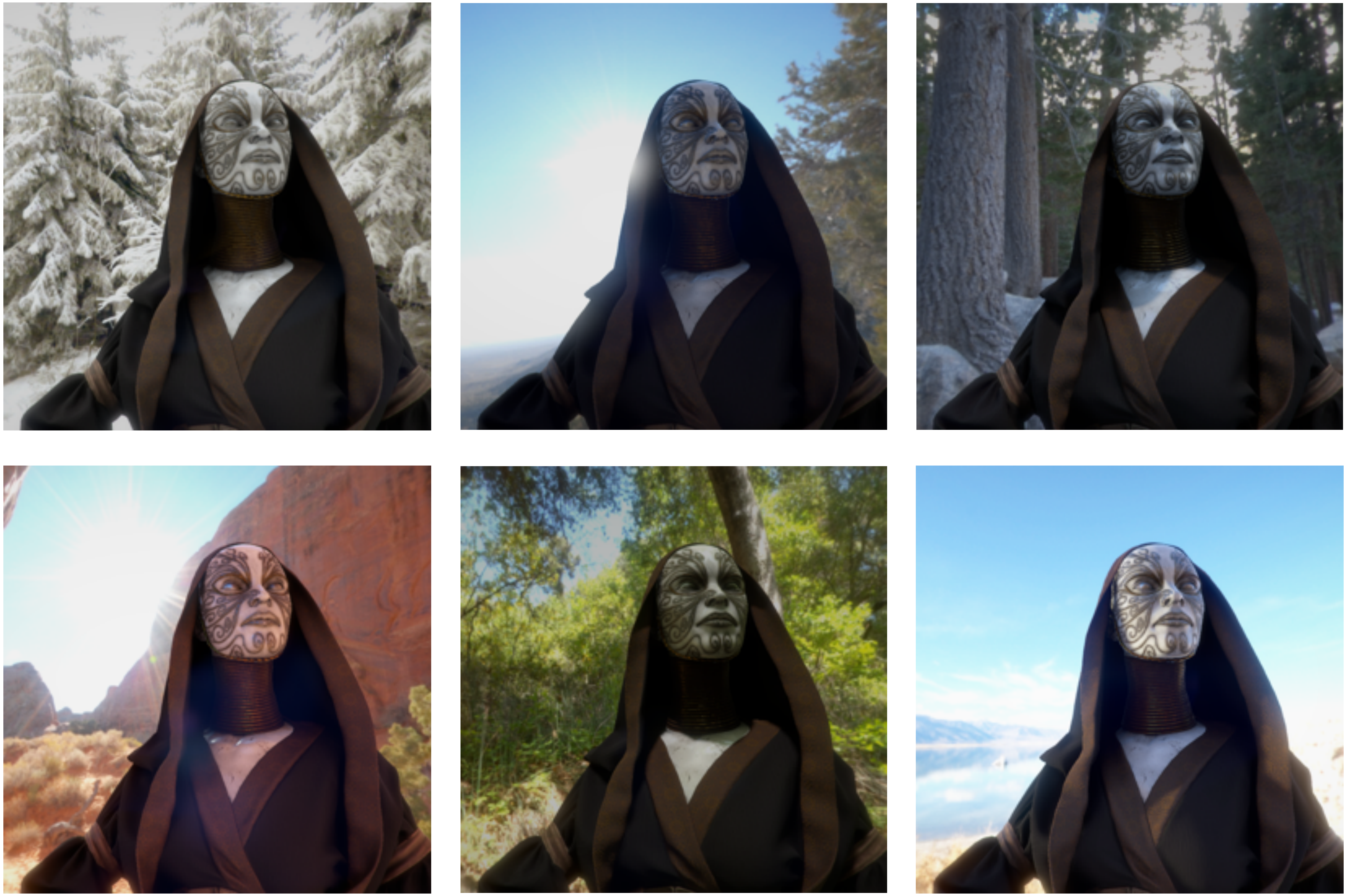


Physically Based Shading (PBS)

- Try to model how light actually behaves
  - Instead of ad-hoc models that may or might not work
- Movies have moved to it
- Games are moving to it
- Does not have to mean“photo-realism”!
  - See Pixar

PBS motivation

- Consistent, plausible look under different lighting conditions
- *Less* material parameters
- All parameter values behave “sensibly”
- More interchangeable data
  - From 3D / material scanning etc.

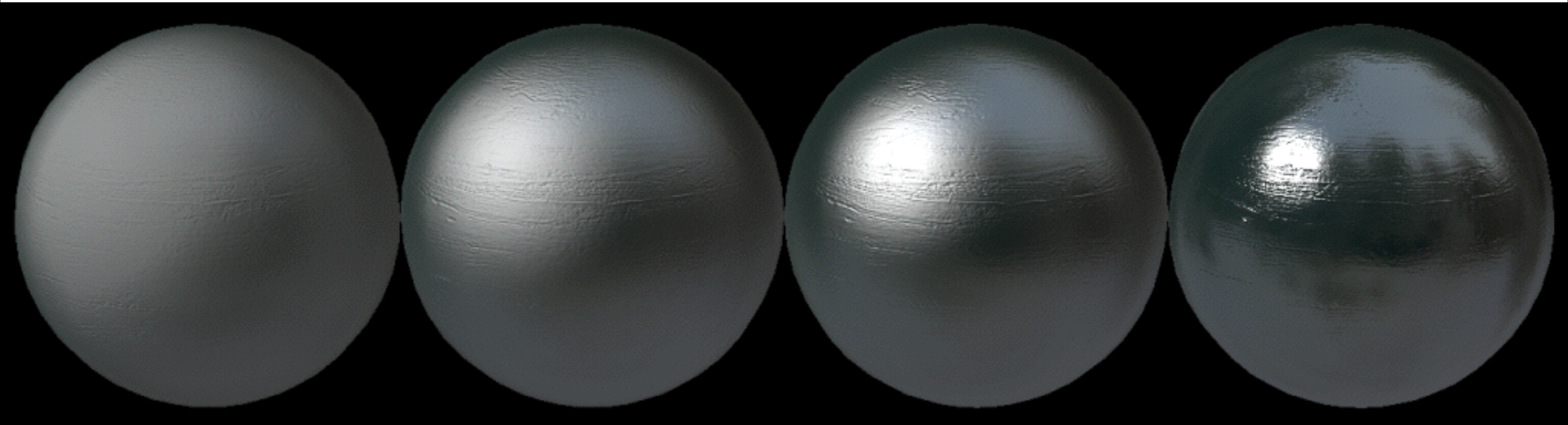


Lighting Conditions

Want same content to look good

Energy conservation

- Don’t reflect more light than you receive
- Sharper reflections = stronger; more blurry reflections = dimmer
- Specular takes away from diffuse



Specular everywhere

- Everything has specular
  - Really! Even if a tiny amount

Fresnel all the things

- Everything has Fresnel
  - Fresnel: surface becomes more reflective at grazing angles
  - *All* surfaces approach 100% reflective!
    - Kind of; Fresnel is more complicated on rough surfaces

Image Based Lighting (IBL)

- *Think Marmoset Skyshop*
- Capture environment into a cubemap
- Blur mip levels in a special way for varying roughness
- Voilà! Good looking reflections
- Place a bunch of them in the level, objects pick up best matching ones

New Standard Shader

New Standard Shader

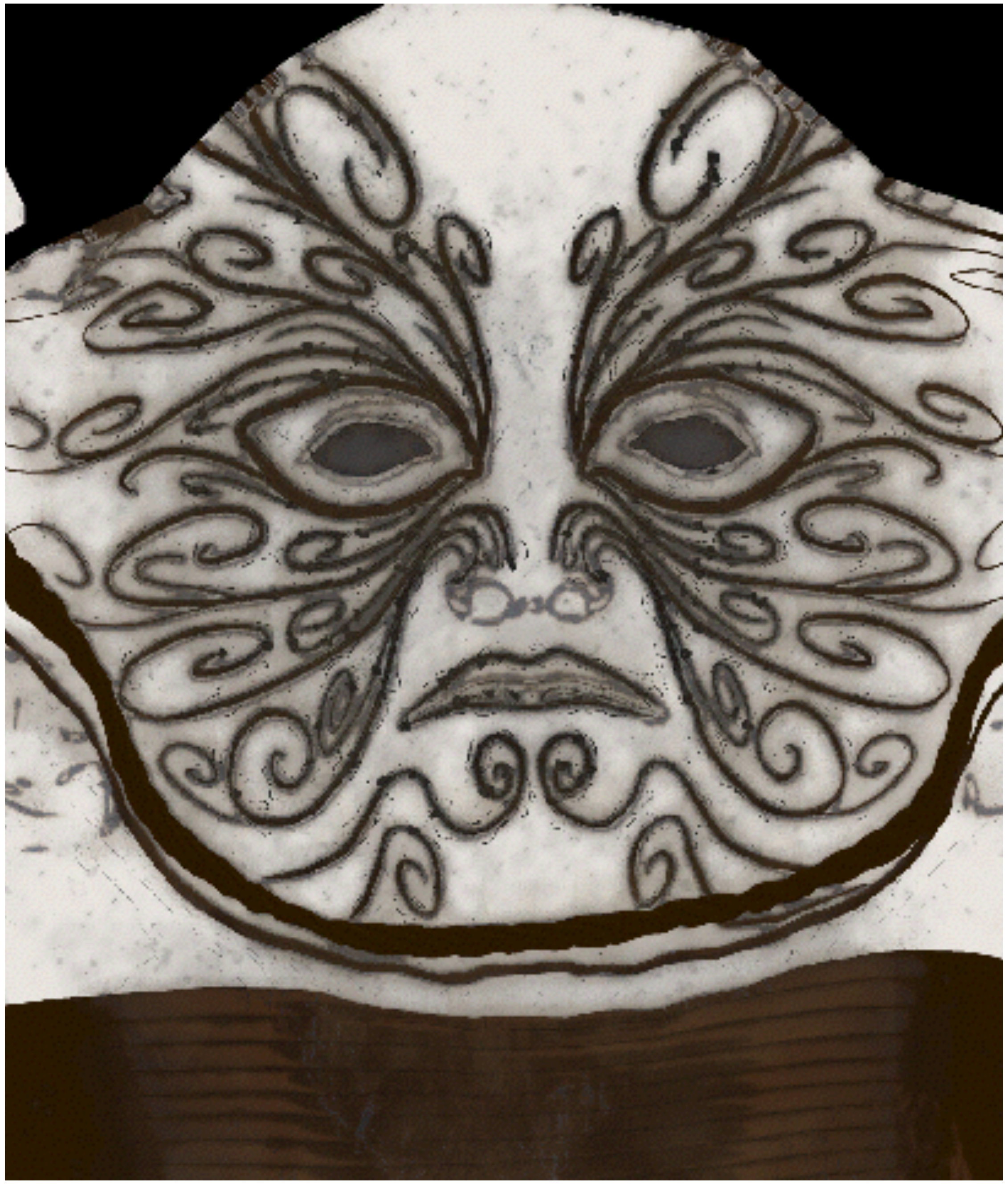
- Good for most everyday materials
  - Metals, Plastics, Wood, Concrete, Cloth, …

Primary Inputs

- Diffuse color
- Specular color
- Surface smoothness
- Normal map

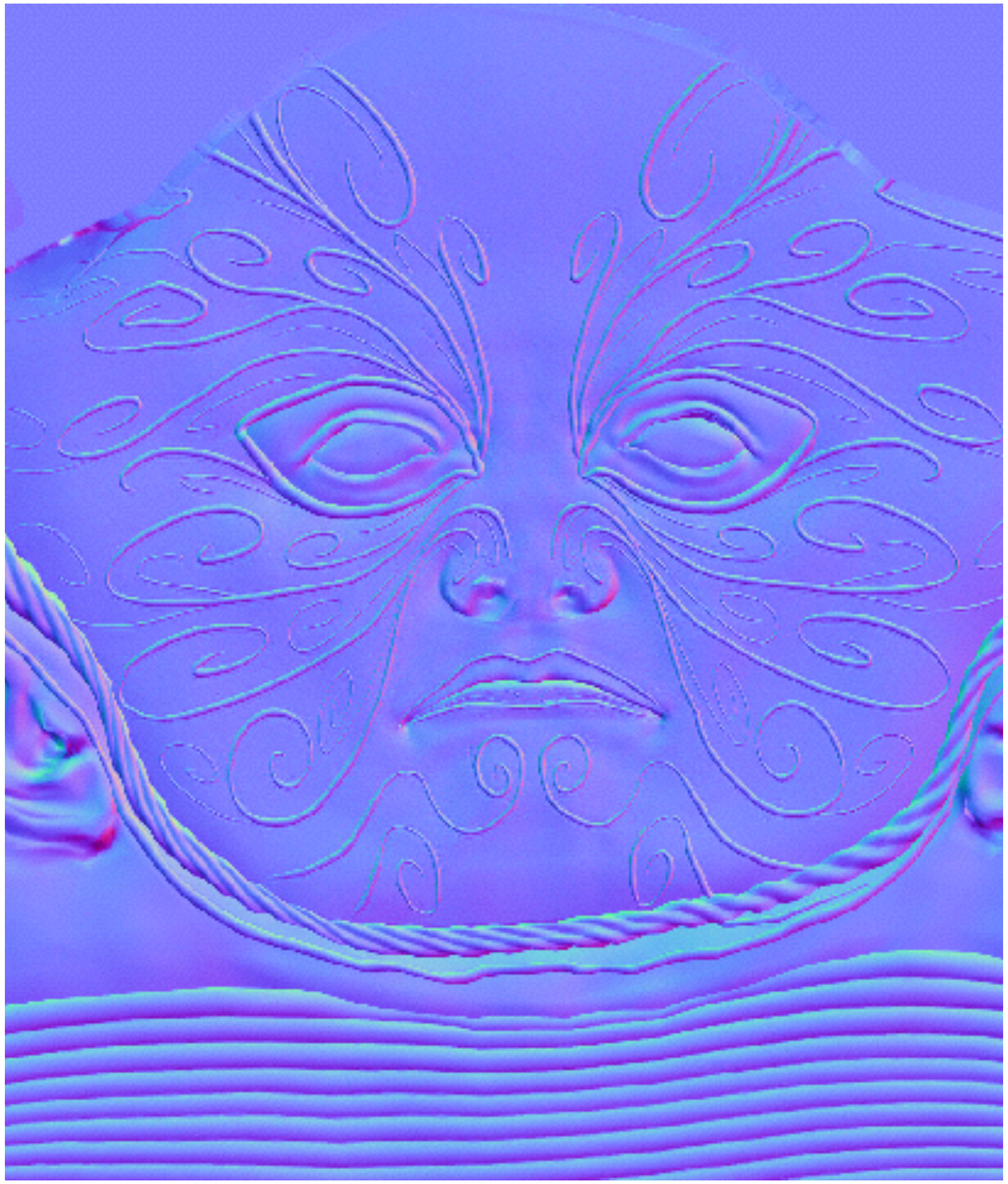
Input: Diffuse

- a.k.a. Albedo
- Should not have lighting!
- Metals have no (black) diffuse
  - Rust etc. can make it not pure black



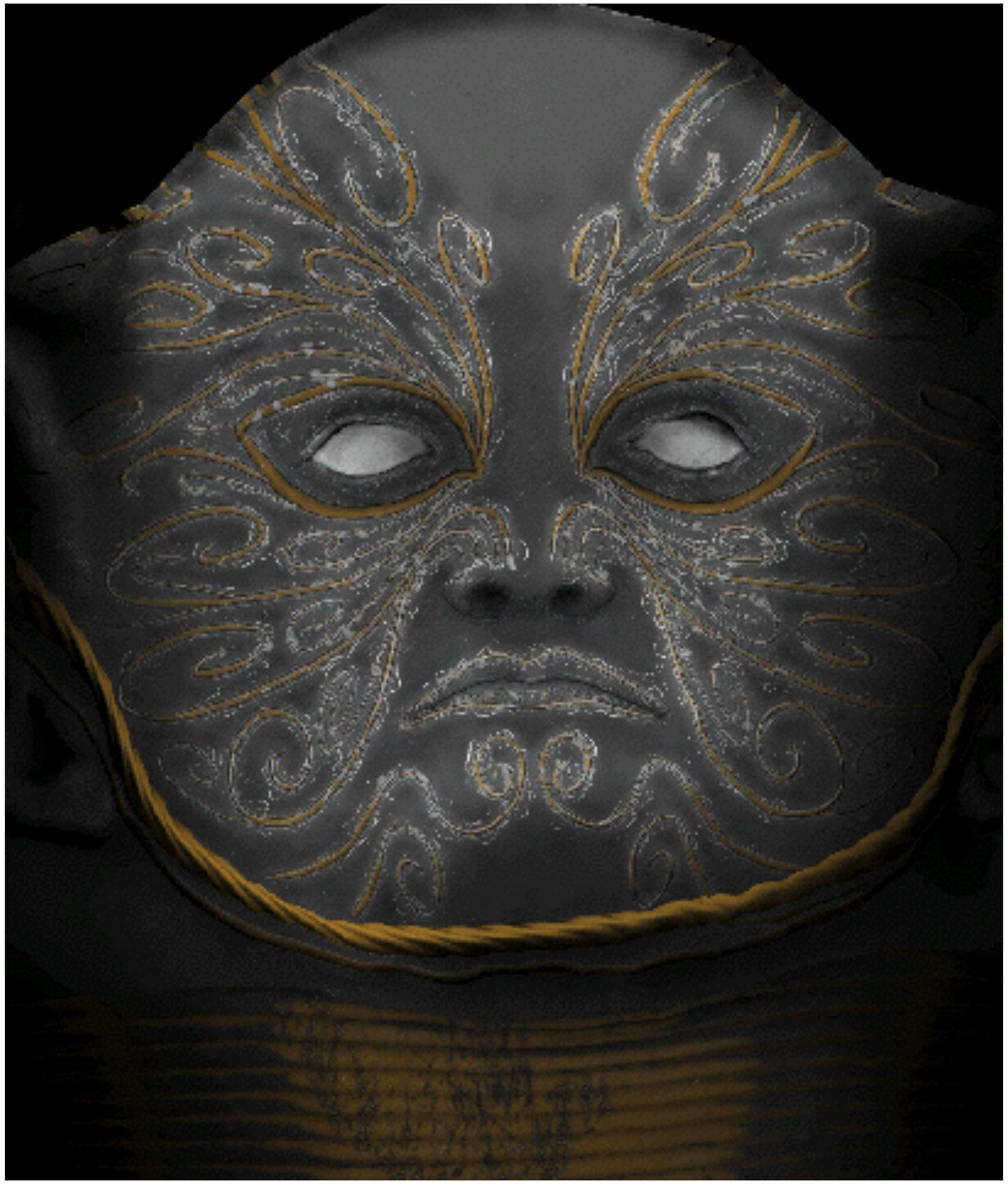
Input: Normals

- The usual tangent space normal map



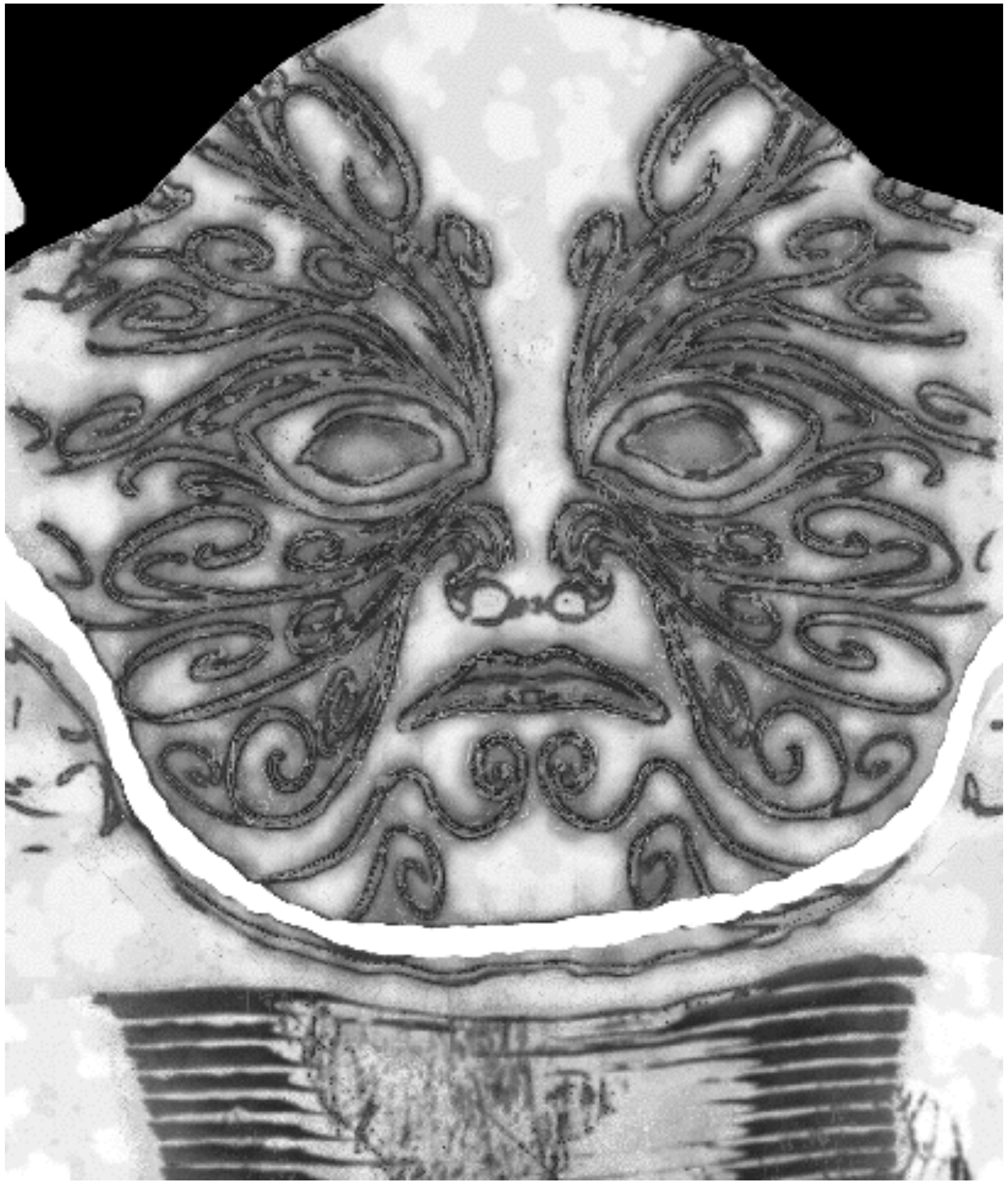
Input: Specular

- Only metals have colored specular
- Non-metals: gray and quite dark
- Very few details in a single material



Input: Smoothness

- a.k.a. glossiness
- or opposite of "roughness"
- 0=rough, 1=smooth
- Interesting detail in this map



Optional inputs

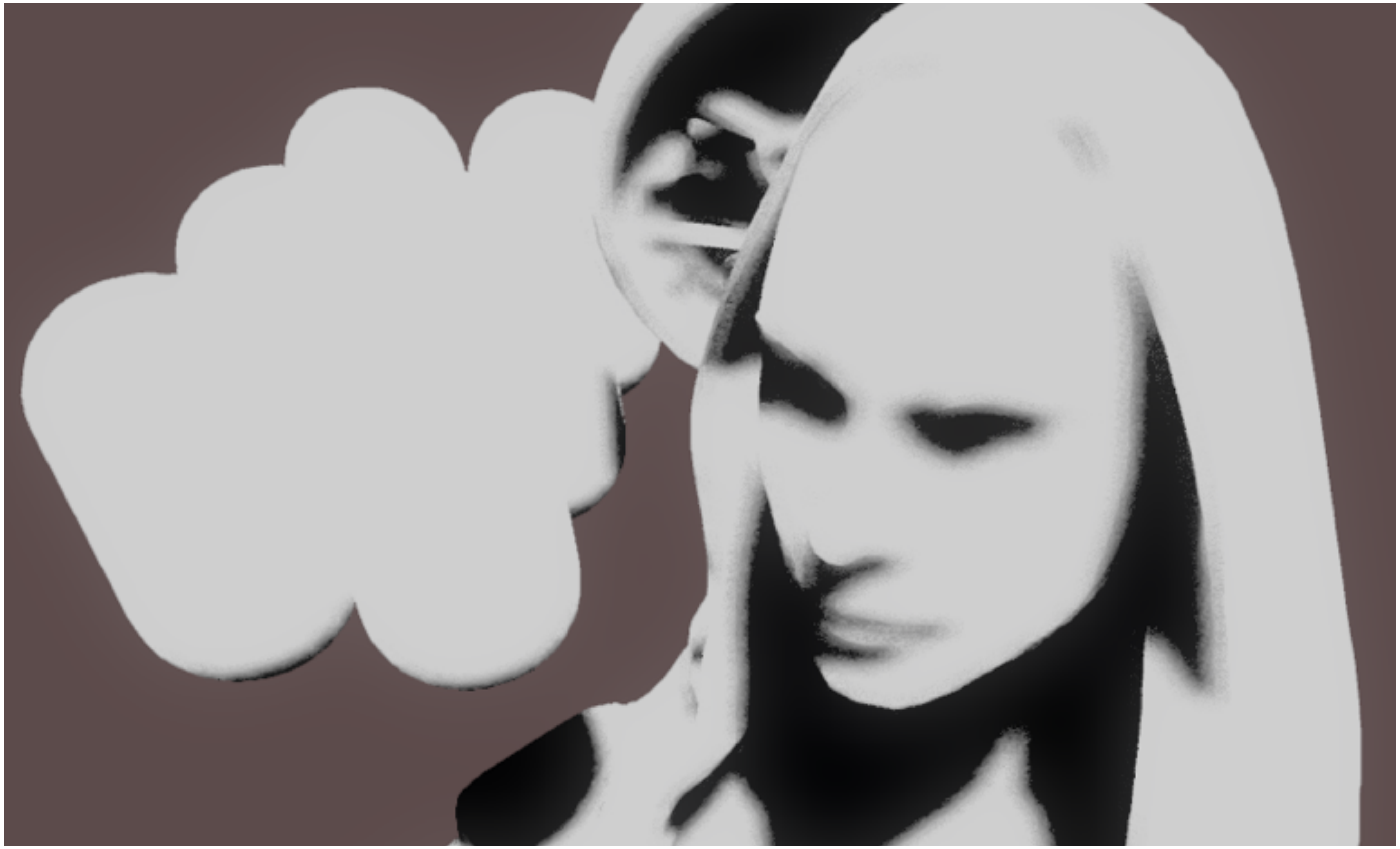
- Emission
- Ambient Occlusion
- Detail albedo/normal maps
- Heightmap (for parallax)

Automatic inputs

- Specular reflection cubemap
  - From reflection probes or global in scene
- Light probes, lightmaps, …

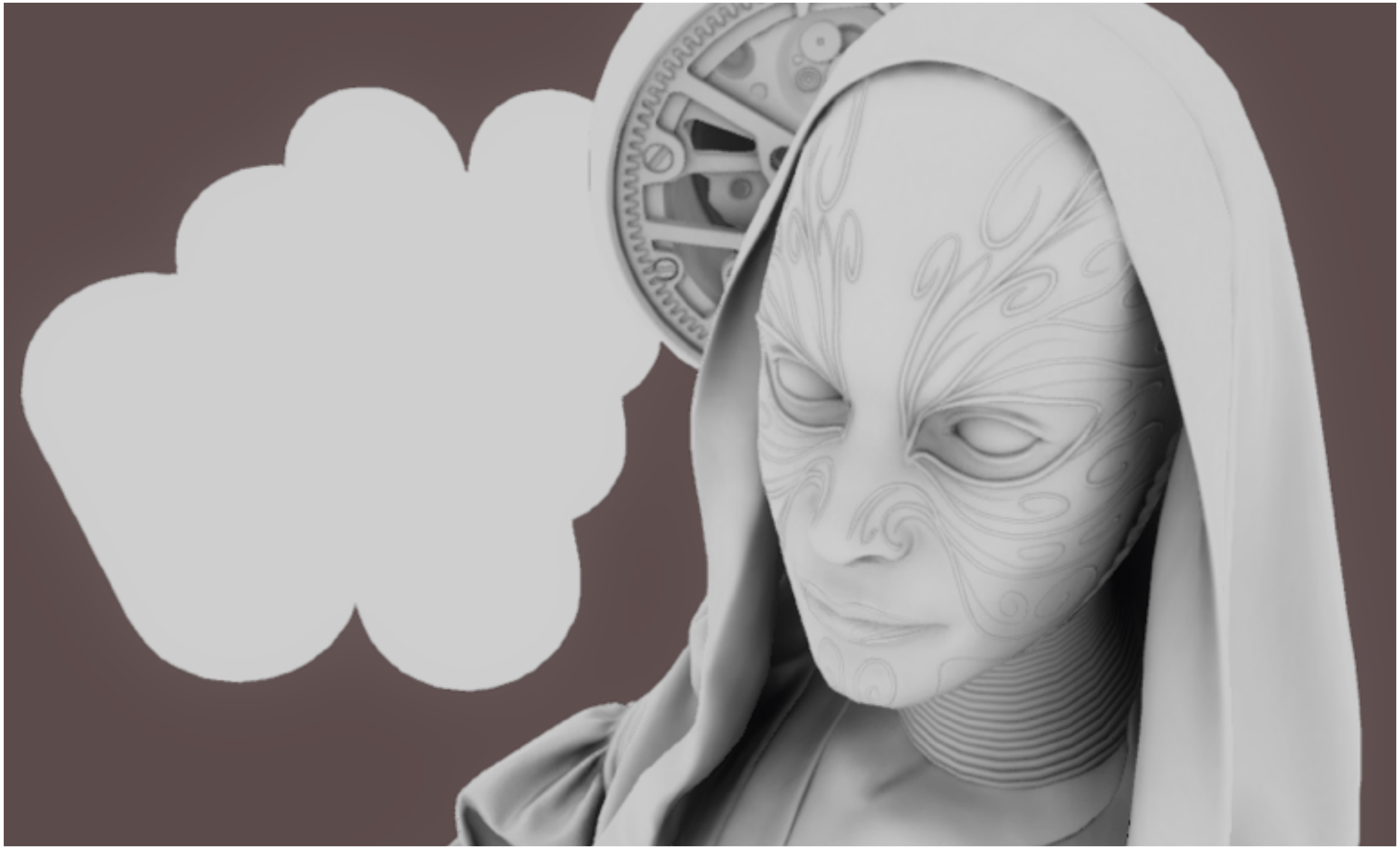
Shading maths

- Microfaced based BRDF
- Inspired by Disney's 2012 research
  - Single smoothness for diffuse & specular
  - Specular takes away from diffuse
- Cook-Torrance, Schlick Fresnel, Blinn-Phong NDF



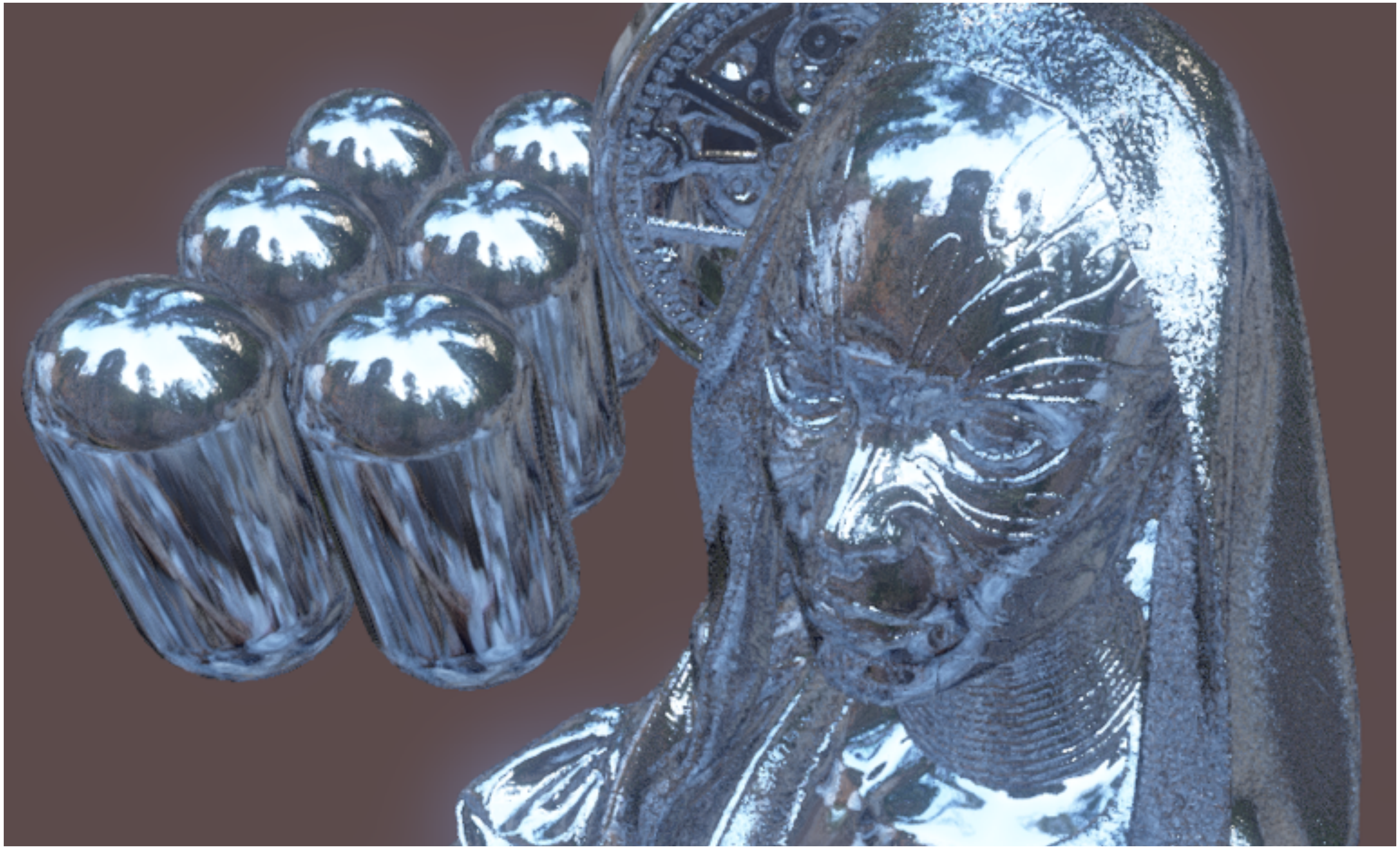
Light attenuation

Here we’re going over shading step by step



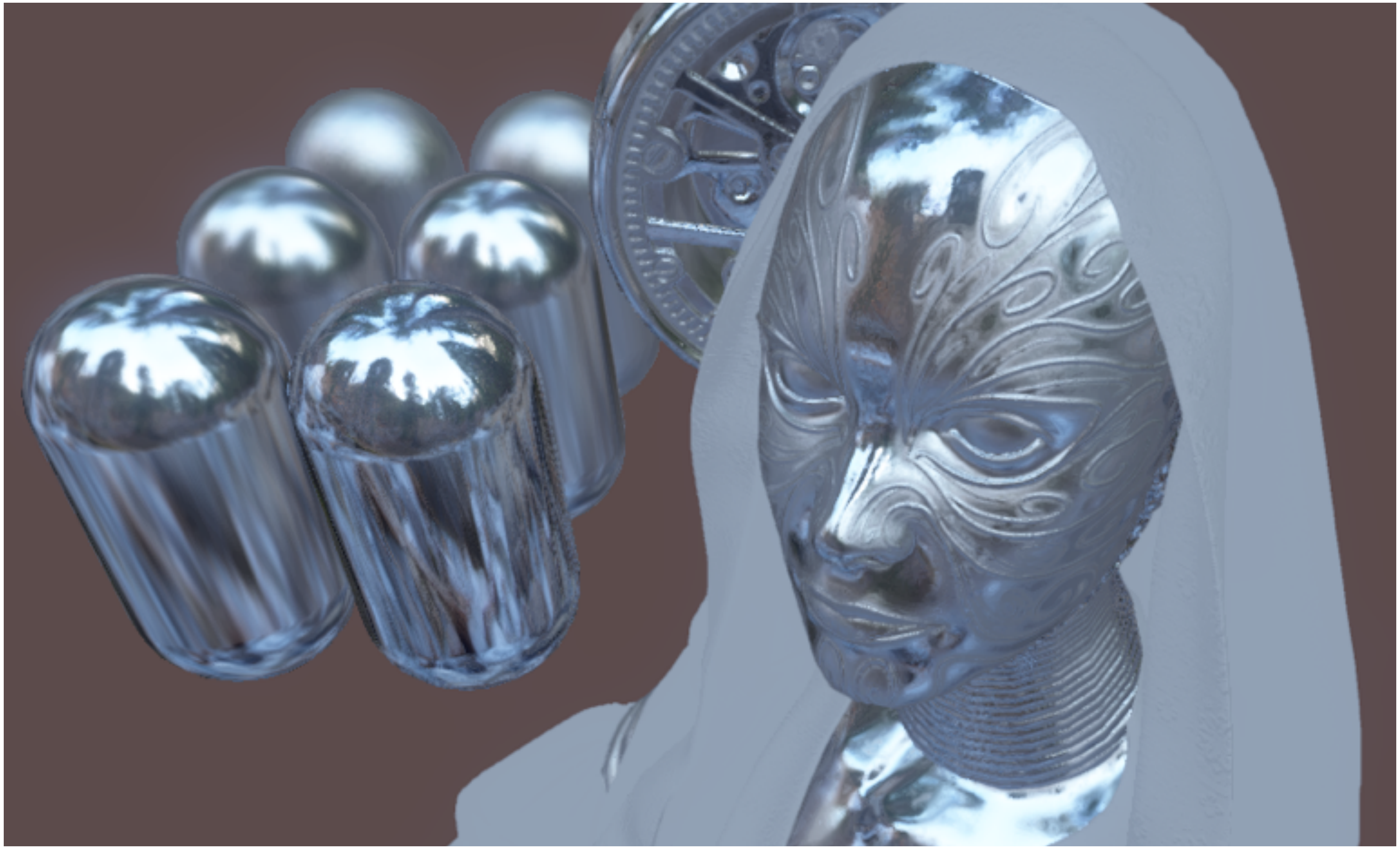
Ambient Occlusion

AO from baked texture



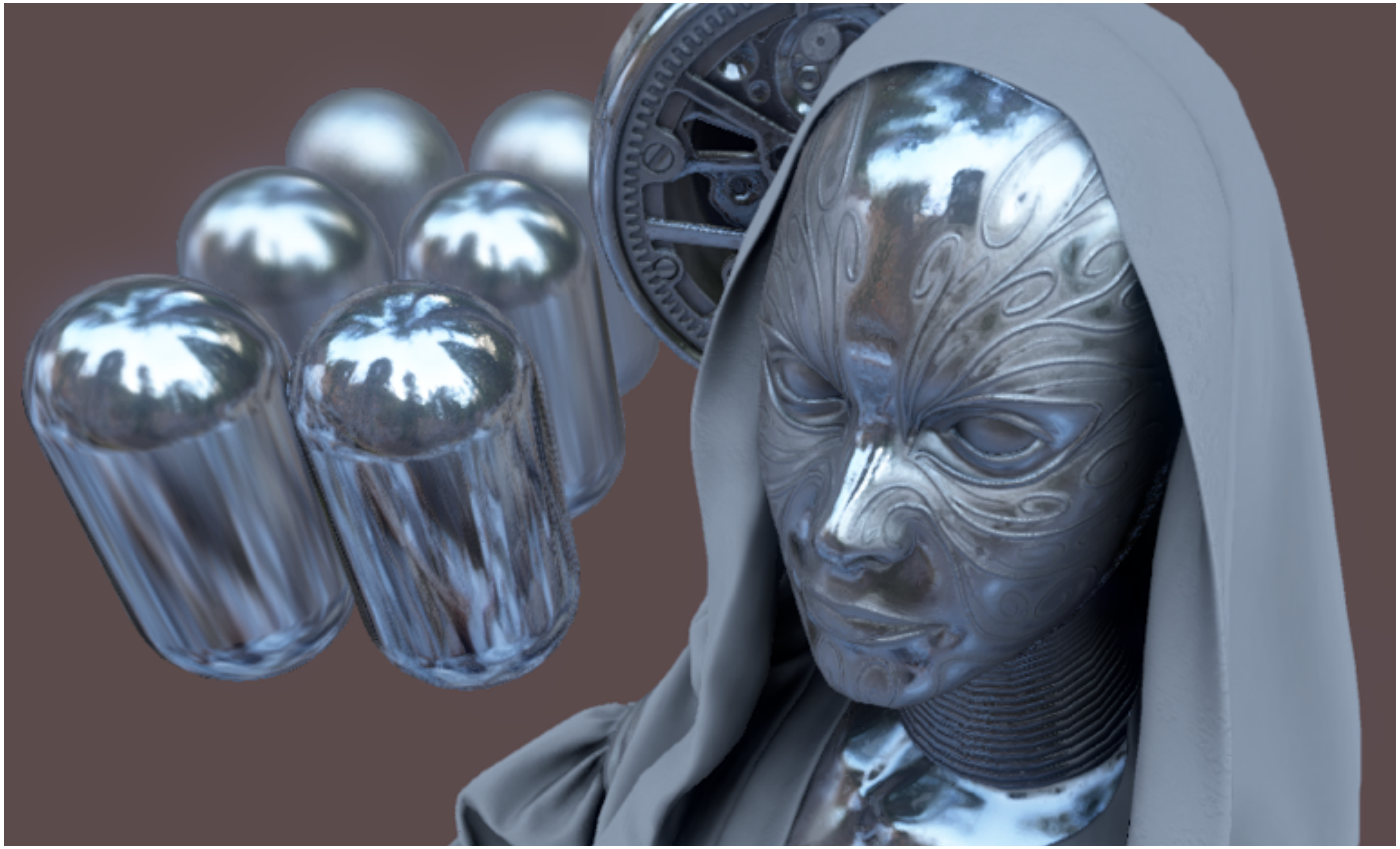
Environment

Just raw cubemap



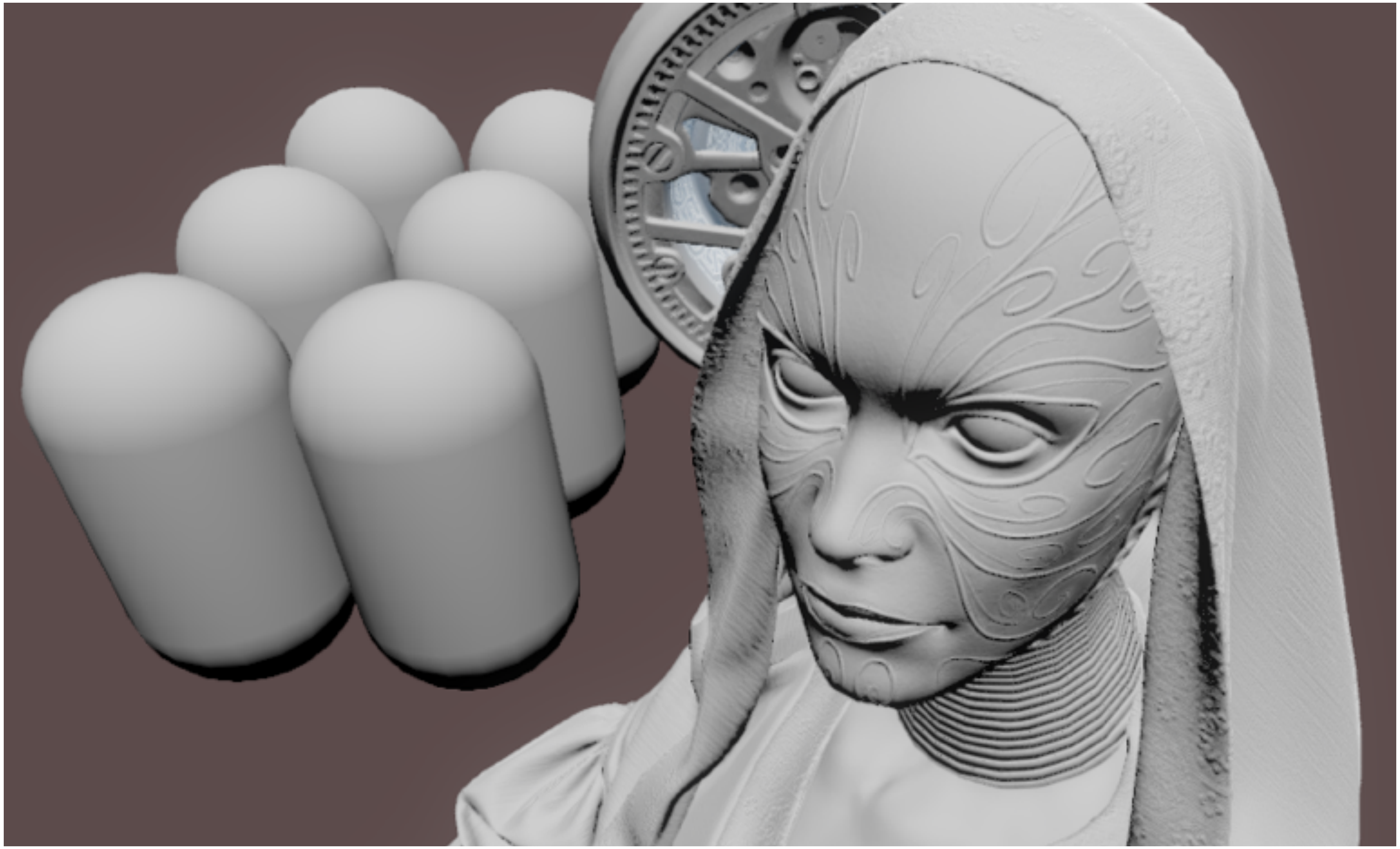
Environment

With proper blur based on smoothness

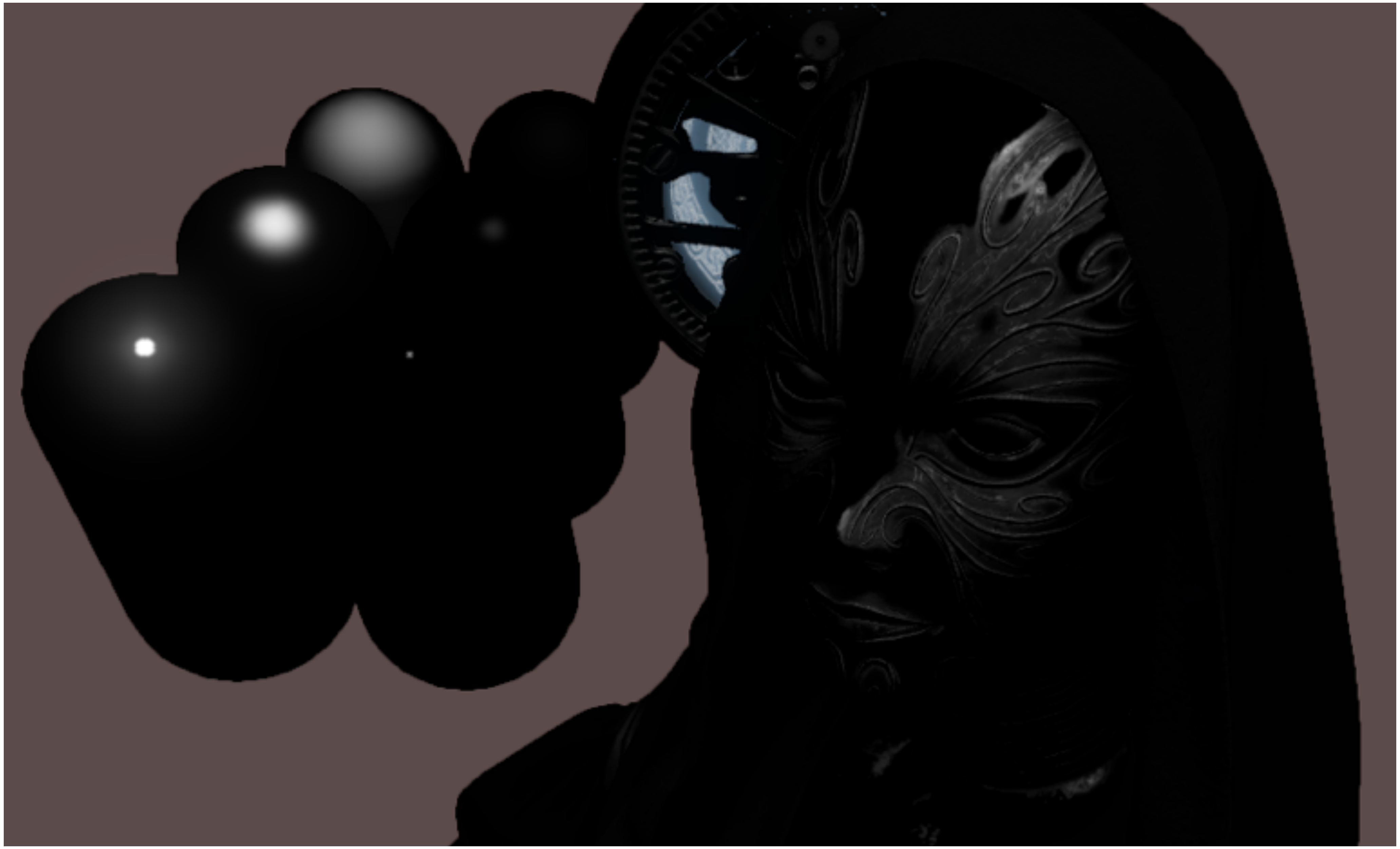


Environment

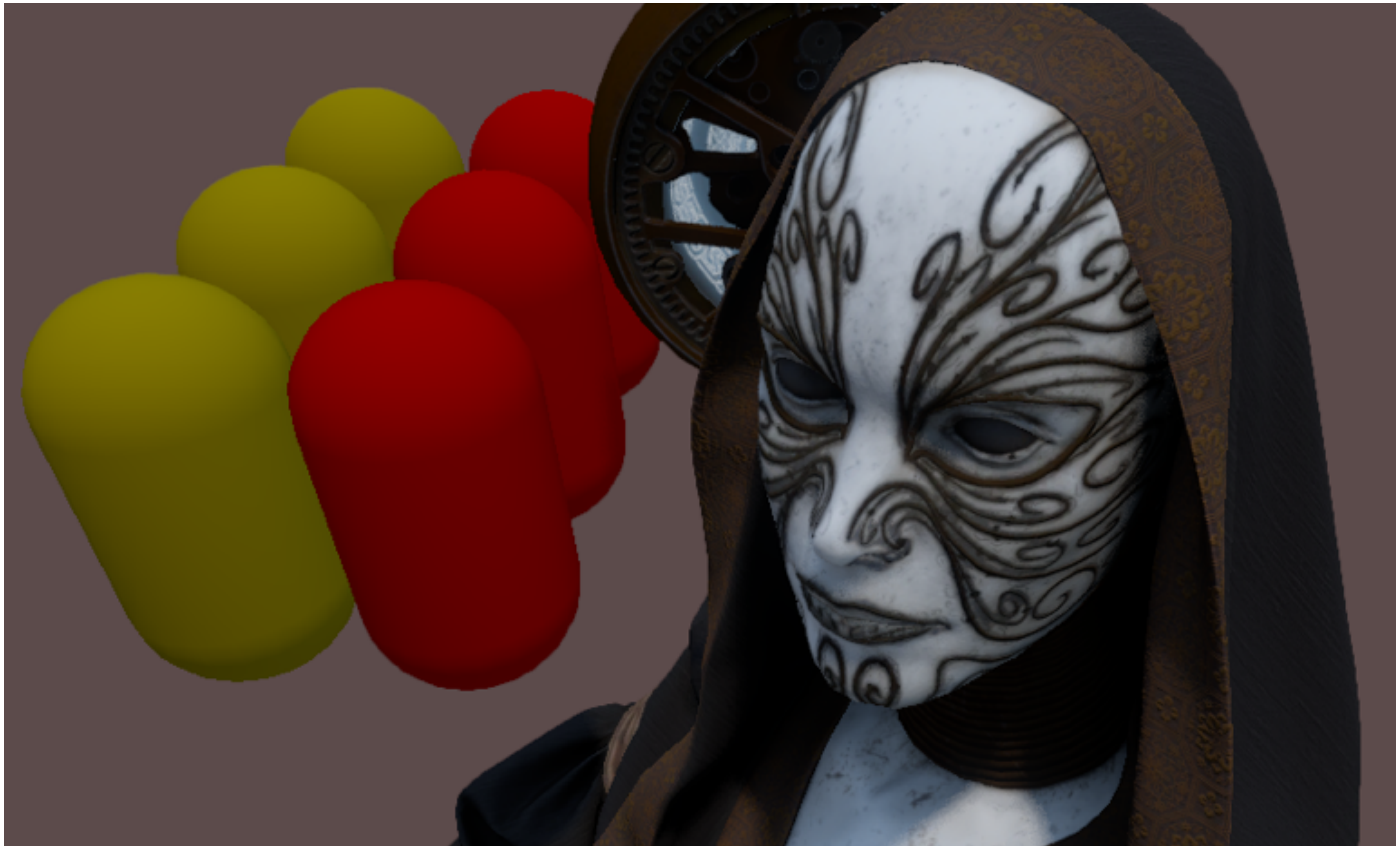
And with ambient occlusion



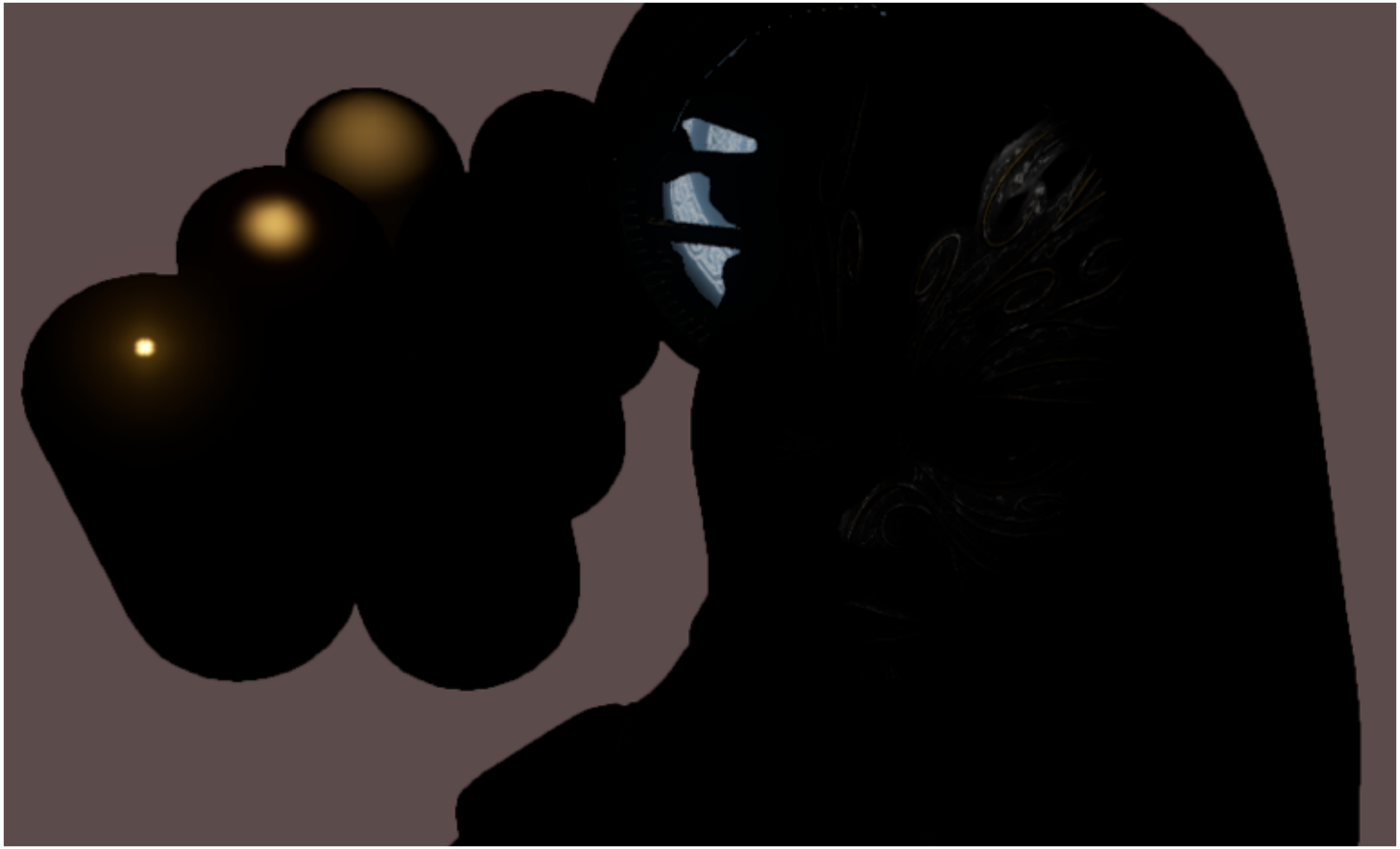
Diffuse factor



Specular factor



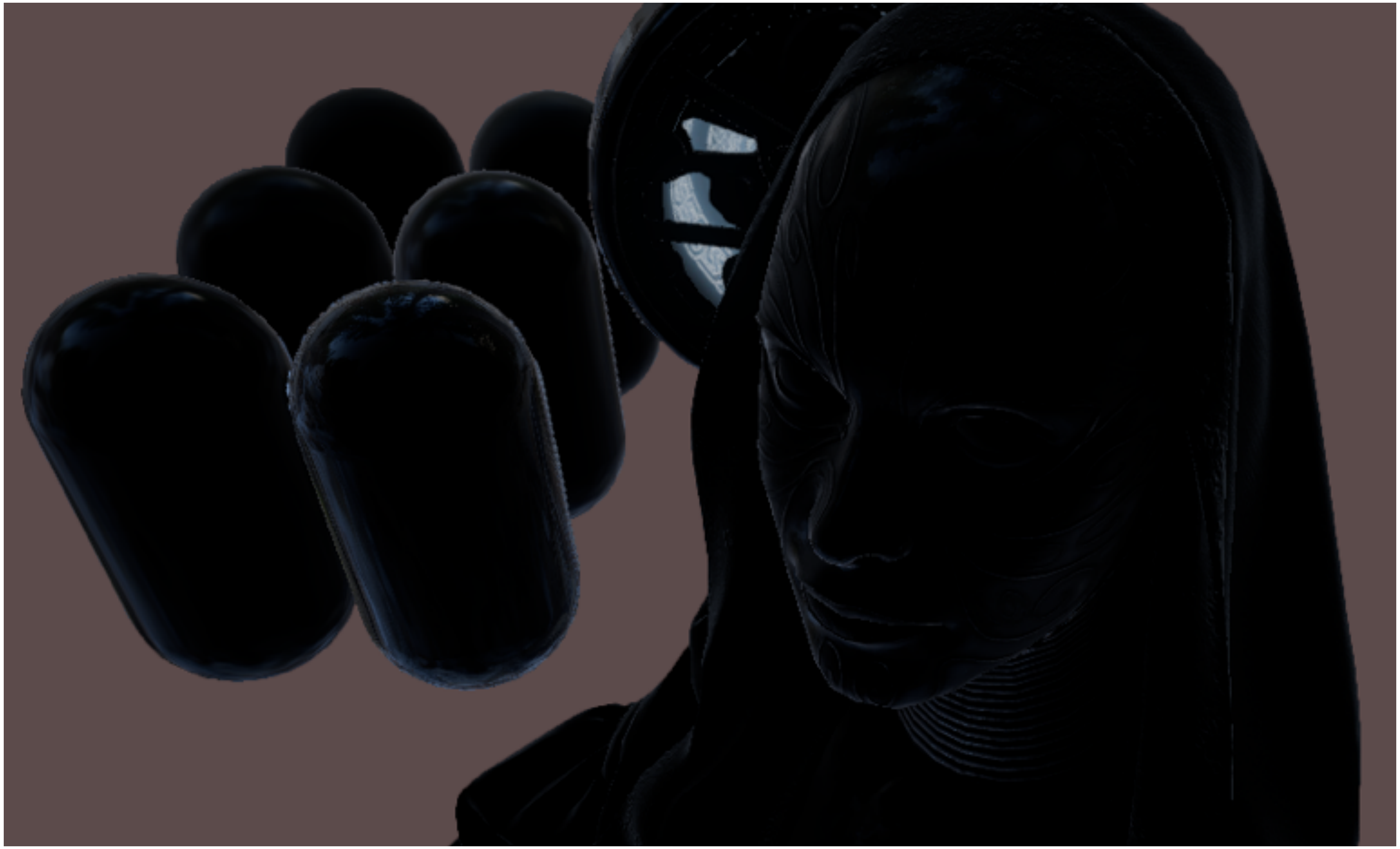
Diffuse part



Specular from light

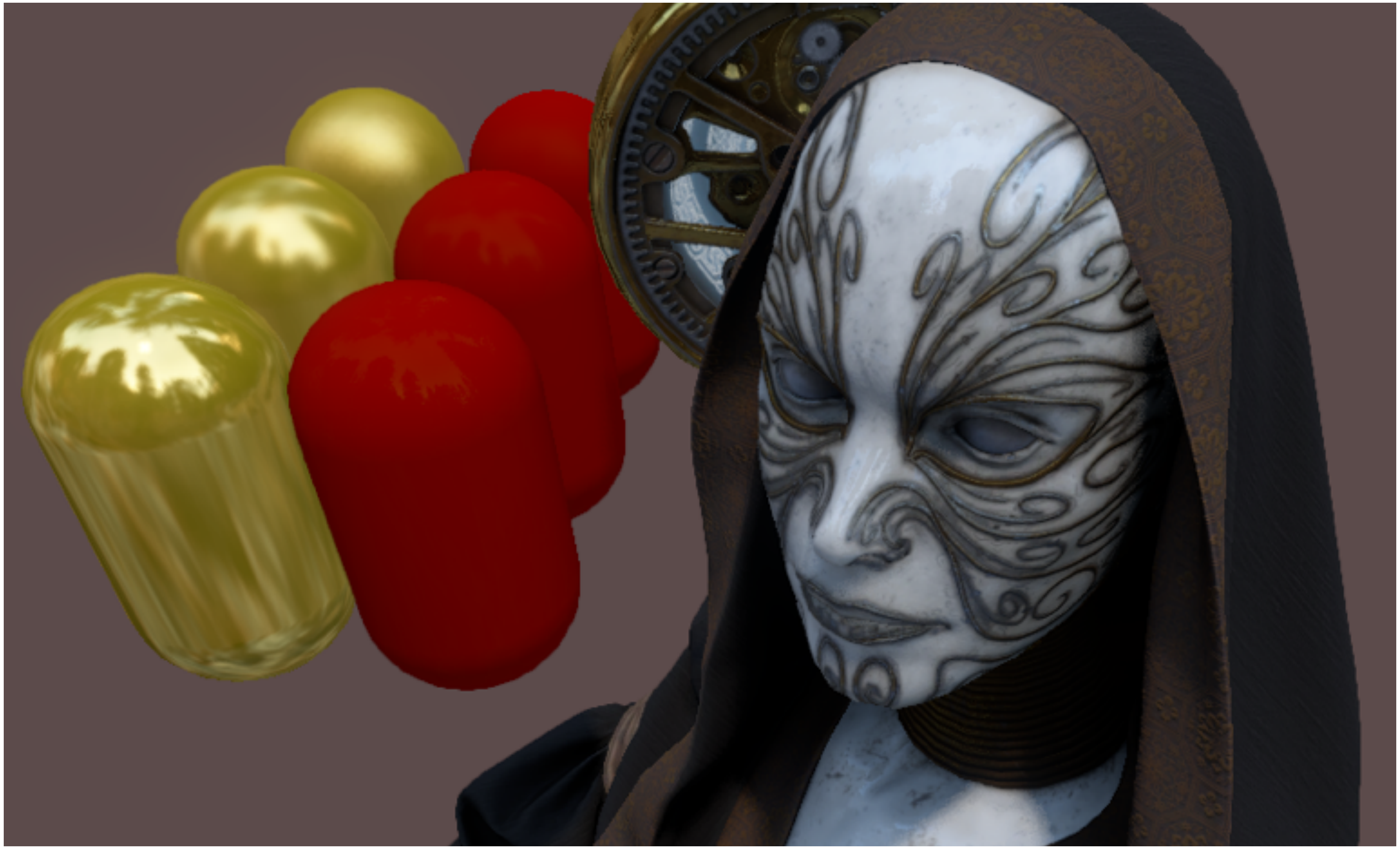
Reflection for metals

Coming from the environment



Fresnel reflection for non-metals

Coming from the environment

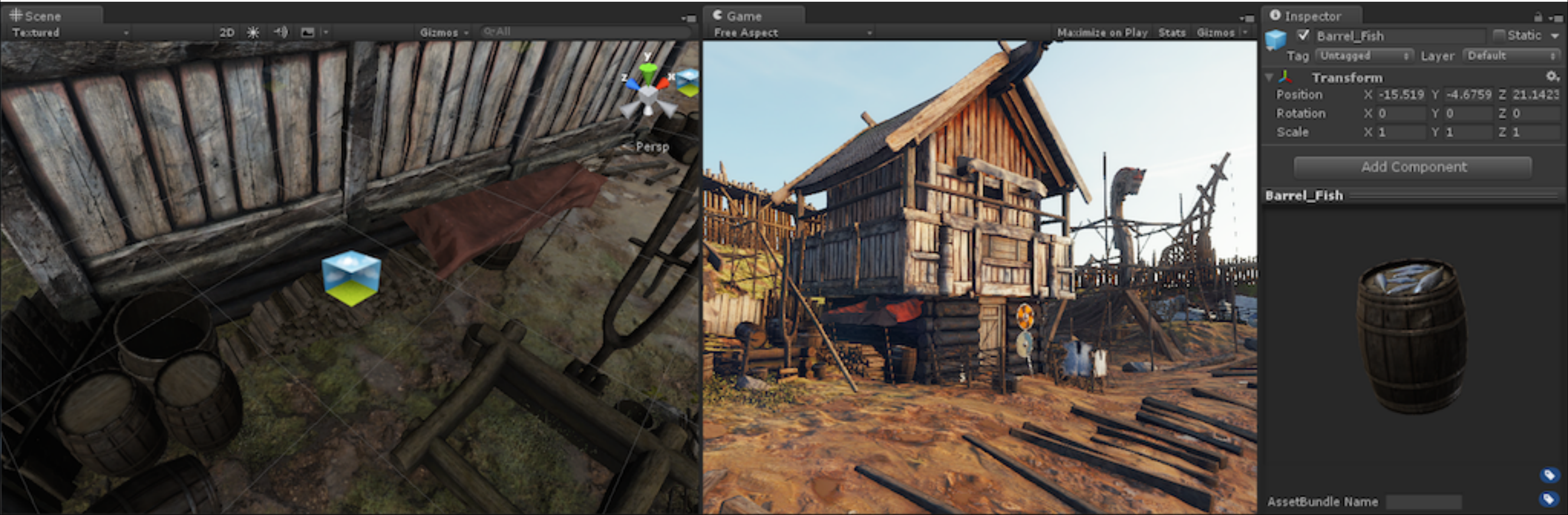


Diffuse+Specular

Everything

Diffuse + Specular + that Fresnel I could not explain in simple words

Workflow



☑ Barrel_Fish □ Static

Tag: Untagged Layer: Default

Transform

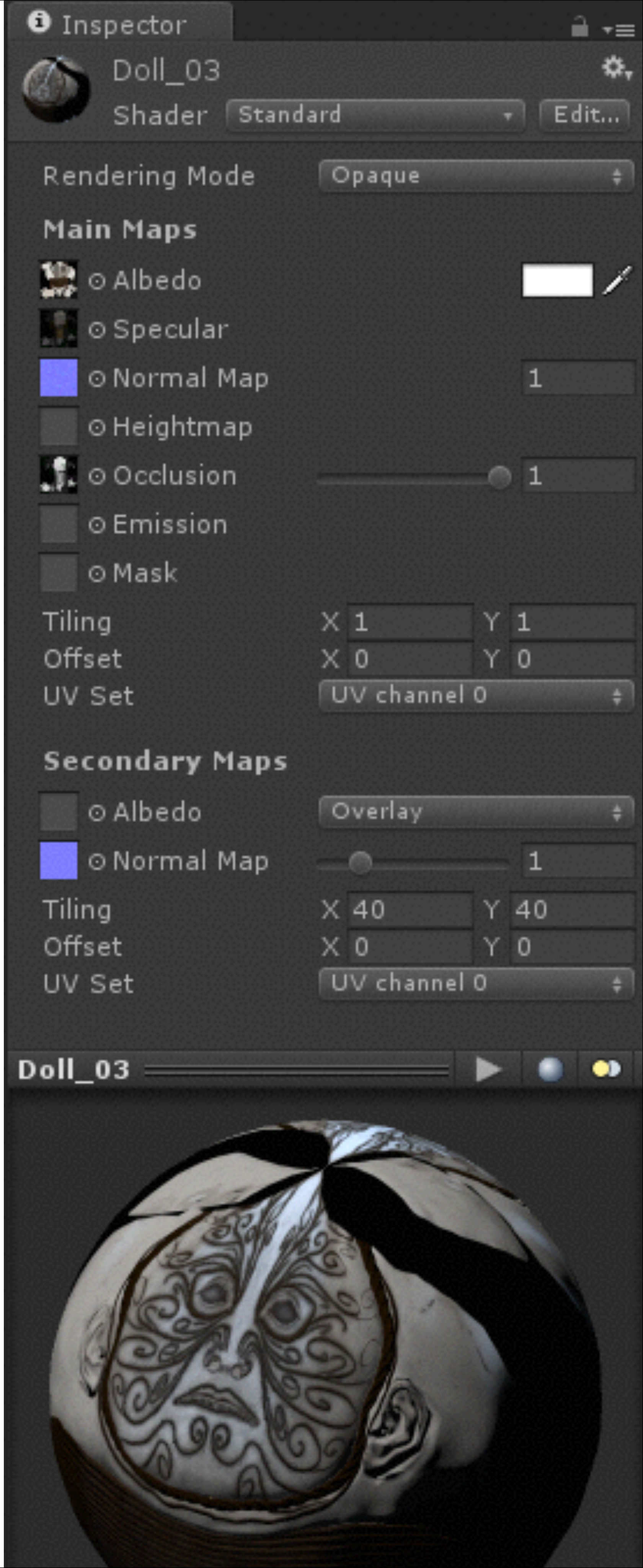
Position	X	-15.519	Y	-4.6759	Z	21.1423
Rotation	X	0	Y	0	Z	0
Scale	X	1	Y	1	Z	1

One shader

- Less guessing at which shader to use
    - “So what is alpha bumped unlit reflective again?”
- No “holes” in variants
    - “I need emission, parallax & alpha, but there is no such shader”

Configurable

- Enable features by assigning textures
- Compact UI
- Additional controls next to relevant textures



Inspector

Doll_03

Shader: Standard | Edit...

Rendering Mode: Opaque

Main Maps

Albedo

Specular

Normal Map: 1

Heightmap

Occlusion: 1

Emission

Mask

Tiling: X 1, Y 1

Offset: X 0, Y 0

UV Set: UV channel 0

Secondary Maps

Albedo: Overlay

Normal Map: 1

Tiling: X 40, Y 40

Offset: X 0, Y 0

UV Set: UV channel 0

Doll_03

Configurable

- Internally, lots of shader variants
  - No emission -> uses faster variant
  - No detail texture -> uses faster variant
- Unused variants not included into game data
  - Otherwise shader would take 200MB…

Related Things

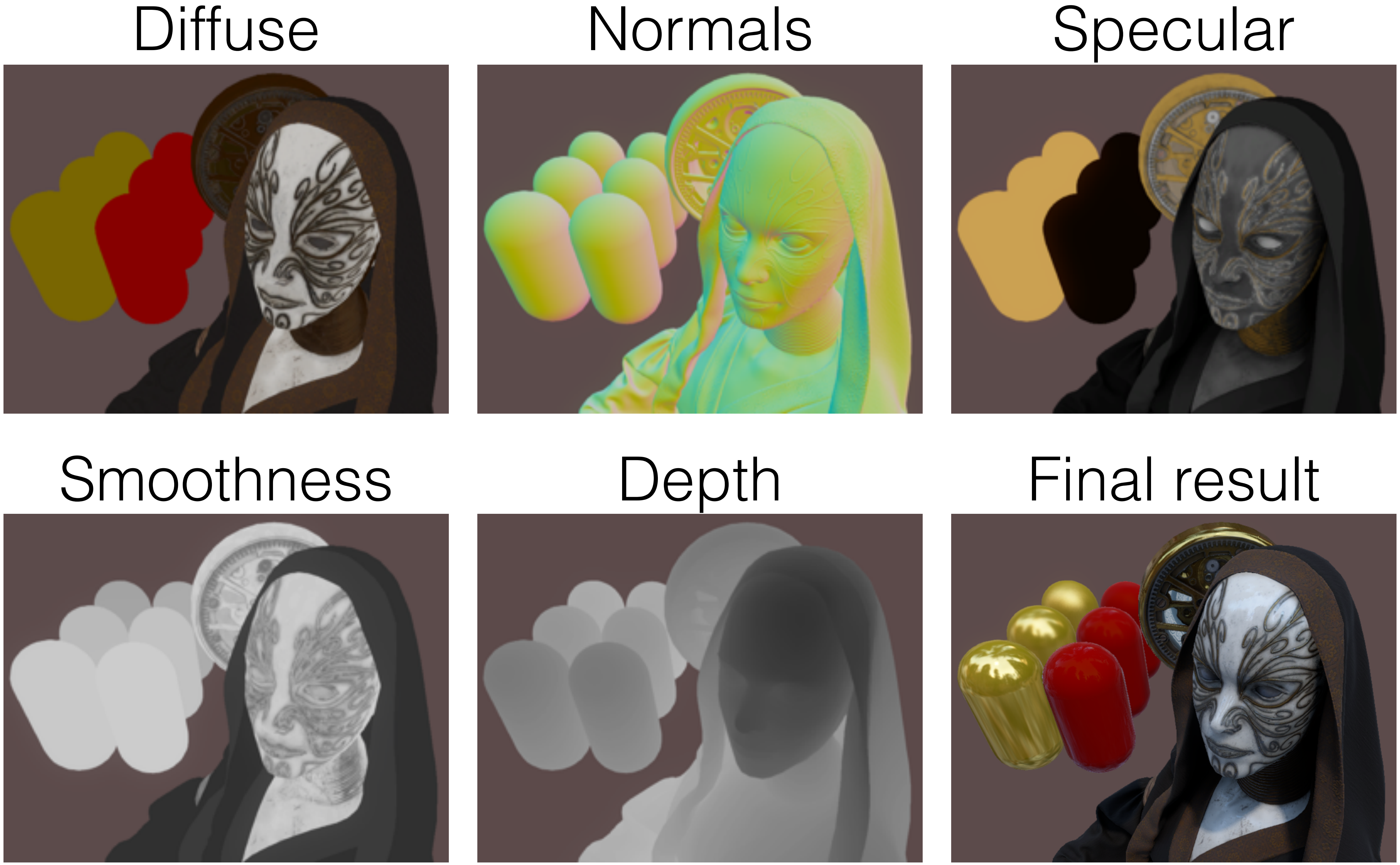
Deferred Shading

- Old deferred lighting (a.k.a. light pre-pass) had minimal g-buffer, and two geometry passes
    - Not enough space in g-buffer for new shader
    - Two geometry passes not nice either
- So we’re making full deferred shading
    - Old one stays as “legacy”

Deferred Shading

- G-buffer layout
  - RT0: diffuse color (rgb)
  - RT1: specular color (rgb), smoothness (a)
  - RT2: world normal (rgb; 10.2)
  - RT3: emission/light buffer; FP16 when HDR
  - Z-buffer: depth
- 160bpp (ldr), 192bpp (hdr)
  - Quite “fat”
  - Likely not final yet

Deferred Shading



Diffuse

Normals

Specular

Smoothness

Depth

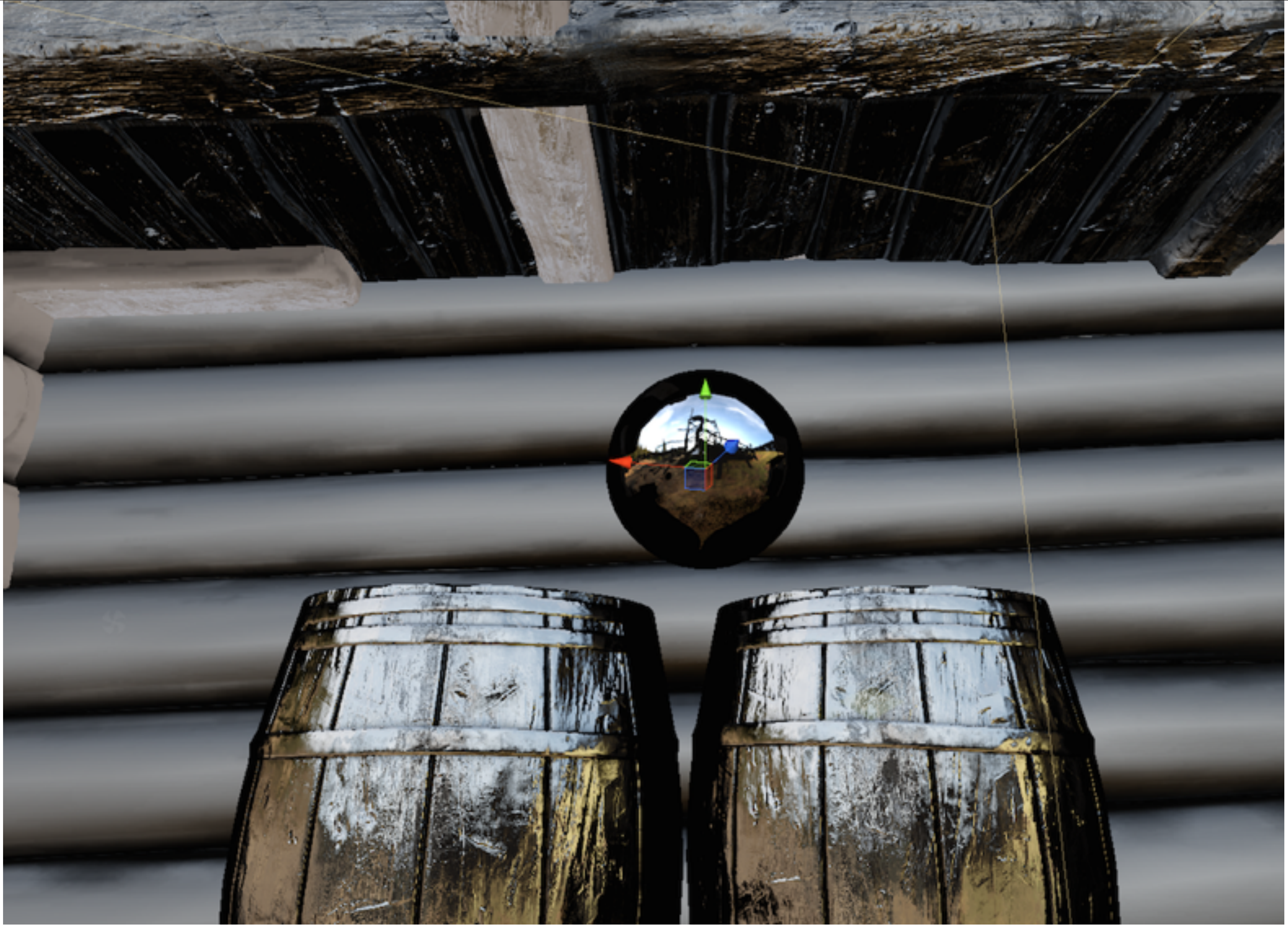
Final result

Deferred Shading

- Thinking about extensibility
  - Custom g-buffer layouts
  - Custom stuff rendered around (g-buffer, lighting) passes
- Very much WIP area right now

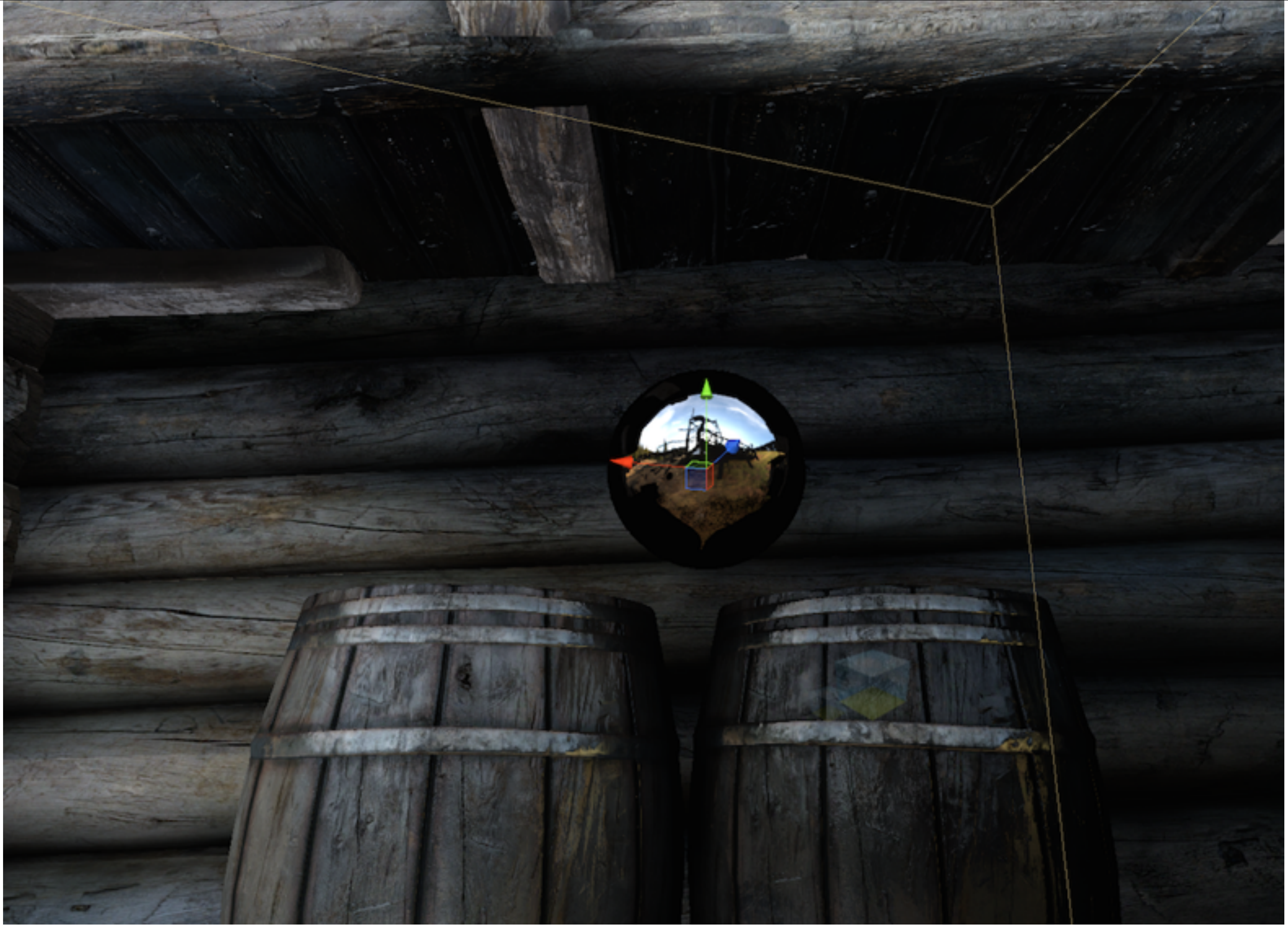
Reflection Probes

- Similar to LightProbes, just for reflection
- A box with size placed in the level
    - Baked cubemap with specular mip levels
    - Can also be realtime rendered (expensive)
- Objects pick up best probe for them
    - And data is fed into shaders



Reflection Probe

Providing environment for nearby objects

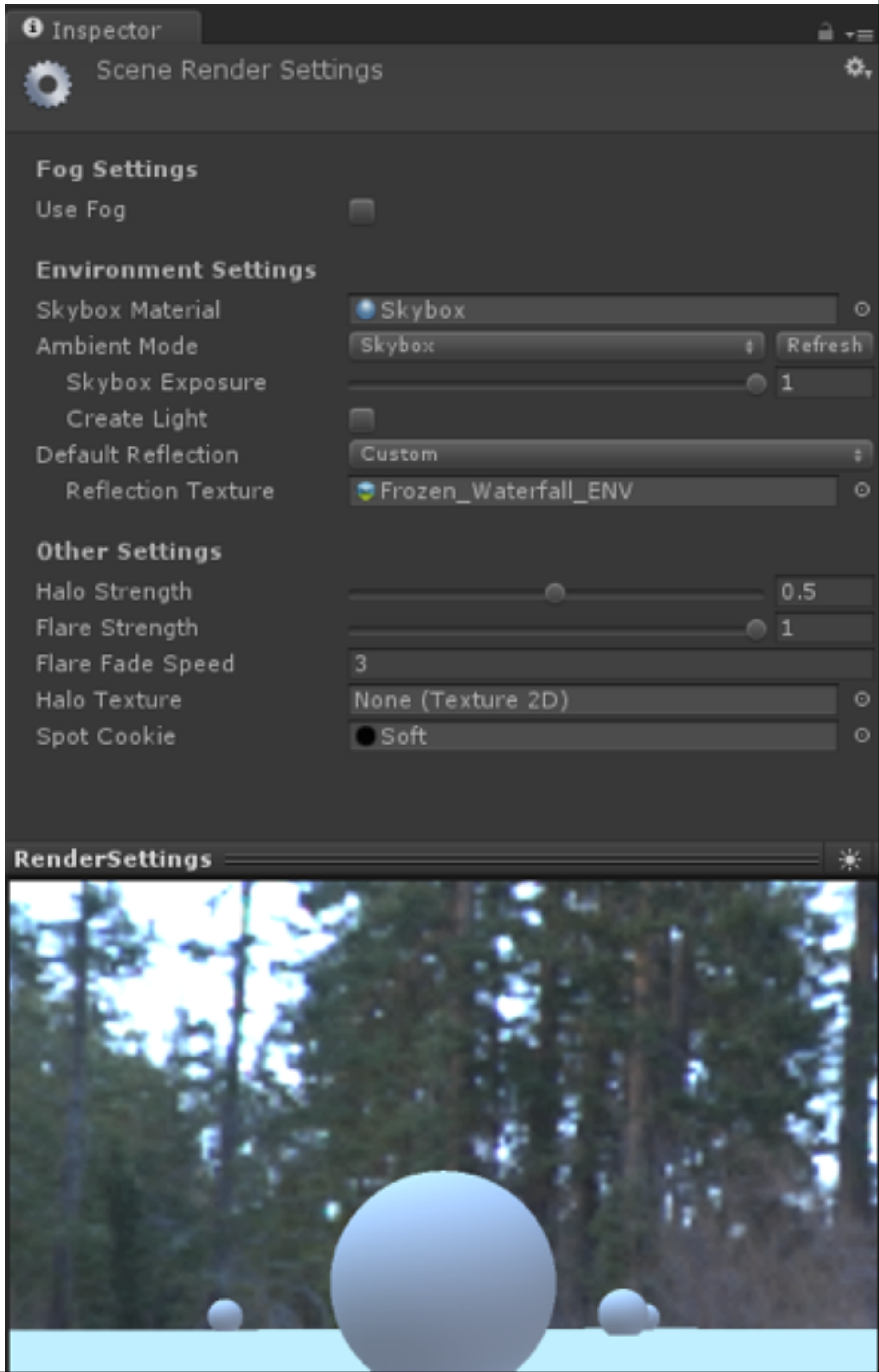


Reflection Probe

Final shaded result

Cubemaps

- Improved cubemap generation from textures
- Cubemap texture compression
- Specular & diffuse convolution for IBL
- Large cubemaps? 64 bit editor ;)
- RenderSettings get a default reflection cube
  - Also directional ambient, better inspector, …



Inspector

Scene Render Settings

Fog Settings

Use Fog □

Environment Settings

Skybox Material: Skybox

Ambient Mode: Skybox — Refresh

Skybox Exposure: 1

Create Light □

Default Reflection: Custom

Reflection Texture: Frozen_Waterfall_ENV

Other Settings

Halo Strength: 0.5

Flare Strength: 1

Flare Fade Speed: 3

Halo Texture: None (Texture 2D)

Spot Cookie: Soft

RenderSettings

HDR

- Scene view can be HDR now
  - When main camera is HDR
  - Uses same tonemapper as main camera
- HDR texture import
  - Encode to RGBM, including cubemaps
  - .hdr/.exr formats

Performance of new shader?

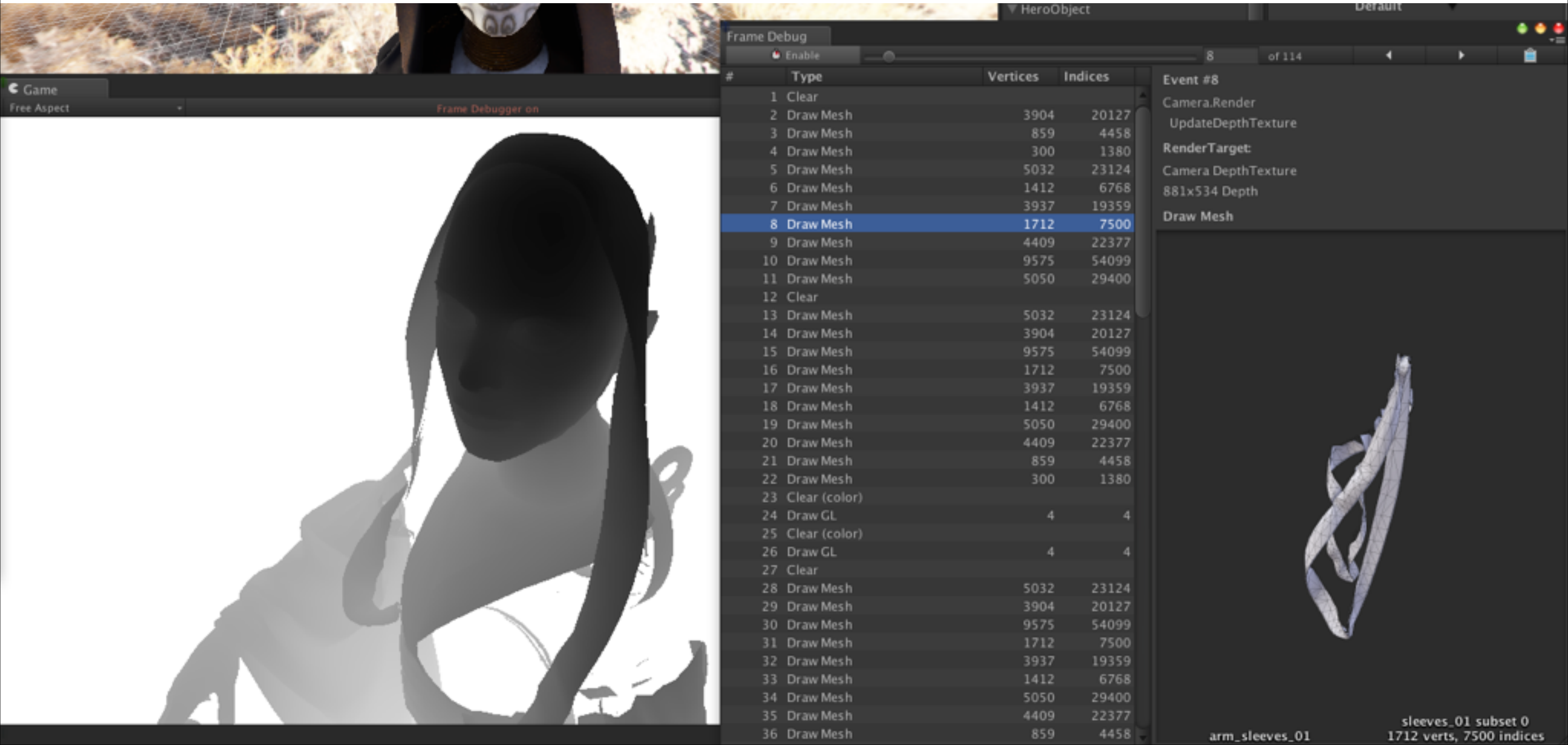
- Right now ok for decent PC & consoles
- Working on mobile/low-end fallbacks
  - Minimalist Cook-Torrance
  - Math baked into lookup texture; smoothness in mip chain
  - Fresnel approximation with 4.0 power
  - tex*Lod -> tex*Bias for DX9 SM2 / GLES2.0
  - No detail texture
  - …

Preemptive answers

- Integration with new GI stuff?
  - Yes.
- Area lights?
  - Probably not in 5.0, but yes, will need them.
- What if I don’t need all this?
  - You can write your own shaders like you always could.
- More shaders?
  - Not in 5.0, but having “clear coat” (car paint etc.) & skin built-in would make sense.

Questions?

Random Bonus Feature



#	Type	Vertices	Indices
1	Clear		
2	Draw Mesh	3904	20127
3	Draw Mesh	859	4458
4	Draw Mesh	300	1380
5	Draw Mesh	5032	23124
6	Draw Mesh	1412	6768
7	Draw Mesh	3937	19359
8	Draw Mesh	1712	7500
9	Draw Mesh	4409	22377
10	Draw Mesh	9575	54099
11	Draw Mesh	5050	29400
12	Clear		
13	Draw Mesh	5032	23124
14	Draw Mesh	3904	20127
15	Draw Mesh	9575	54099
16	Draw Mesh	1712	7500
17	Draw Mesh	3937	19359
18	Draw Mesh	1412	6768
19	Draw Mesh	5050	29400
20	Draw Mesh	4409	22377
21	Draw Mesh	859	4458
22	Draw Mesh	300	1380
23	Clear (color)		
24	Draw GL	4	4
25	Clear (color)		
26	Draw GL	4	4
27	Clear		
28	Draw Mesh	5032	23124
29	Draw Mesh	3904	20127
30	Draw Mesh	9575	54099
31	Draw Mesh	1712	7500
32	Draw Mesh	3937	19359
33	Draw Mesh	1412	6768
34	Draw Mesh	5050	29400
35	Draw Mesh	4409	22377
36	Draw Mesh	859	4458

Event #8

Camera.Render
UpdateDepthTexture

RenderTarget:

Camera DepthTexture
881x534 Depth

Draw Mesh

arm_sleeves_01

sleeves_01 subset 0
1712 verts, 7500 indices

Frame Debugger

- Step through draw calls and see exactly how the frame is rendered
- Current render target, mesh being rendered etc.