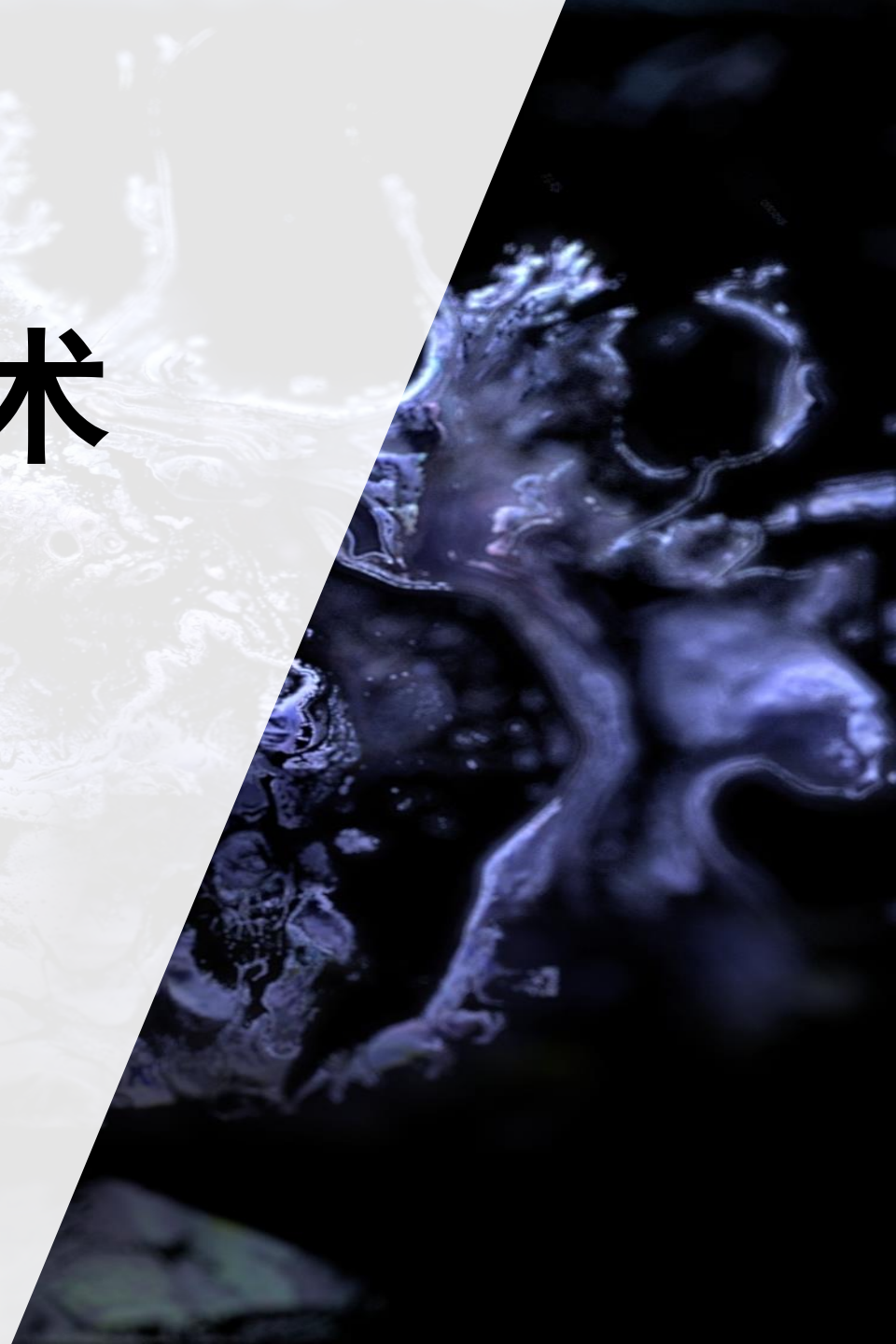
GPU光线追踪技术

助力渲染流程改进



AUTODESK

沈映

SIGGRAPH 2019

What’s new with Arnold GPU

Adrien Herubel
Arnold Core Lead - Autodesk

Steven Parker
Vice President, Professional Graphics - NVIDIA

AUTODESK

NVIDIA

Why GPU?

- Towards real-time lookdev
- More interactive lighting
- Strong demand
- GPUs are getting bigger
- Easier to scale for small & medium facilities

AUTODESK ARNOLD & NVIDIA OPTIX COLLABORATION

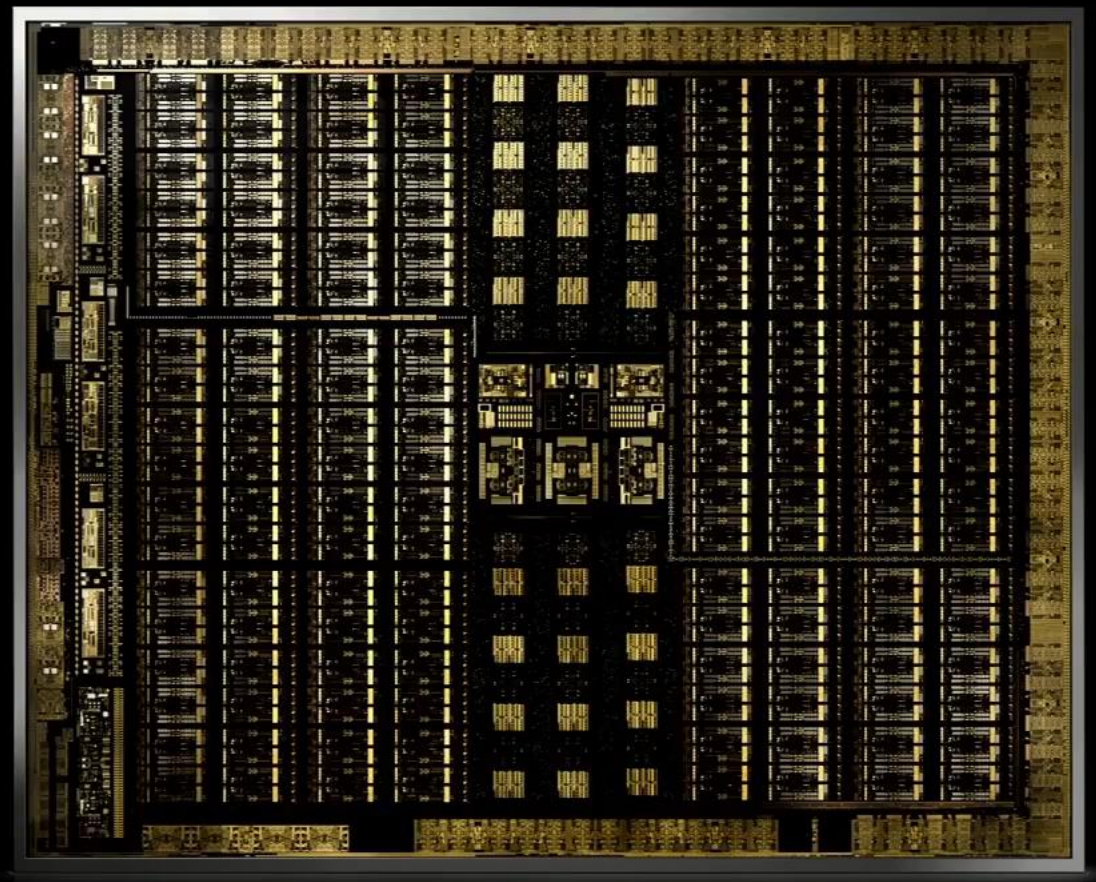
AUTODESK.

Leveraging OptiX NVIDIA Partnership

- Benefit from NVIDIA expertise
- OptiX is a compiler: good for complexity
- Great reactivity for missing features
  - Motion blur
  - Faster JIT compilation
  - OSL support
- RTX!

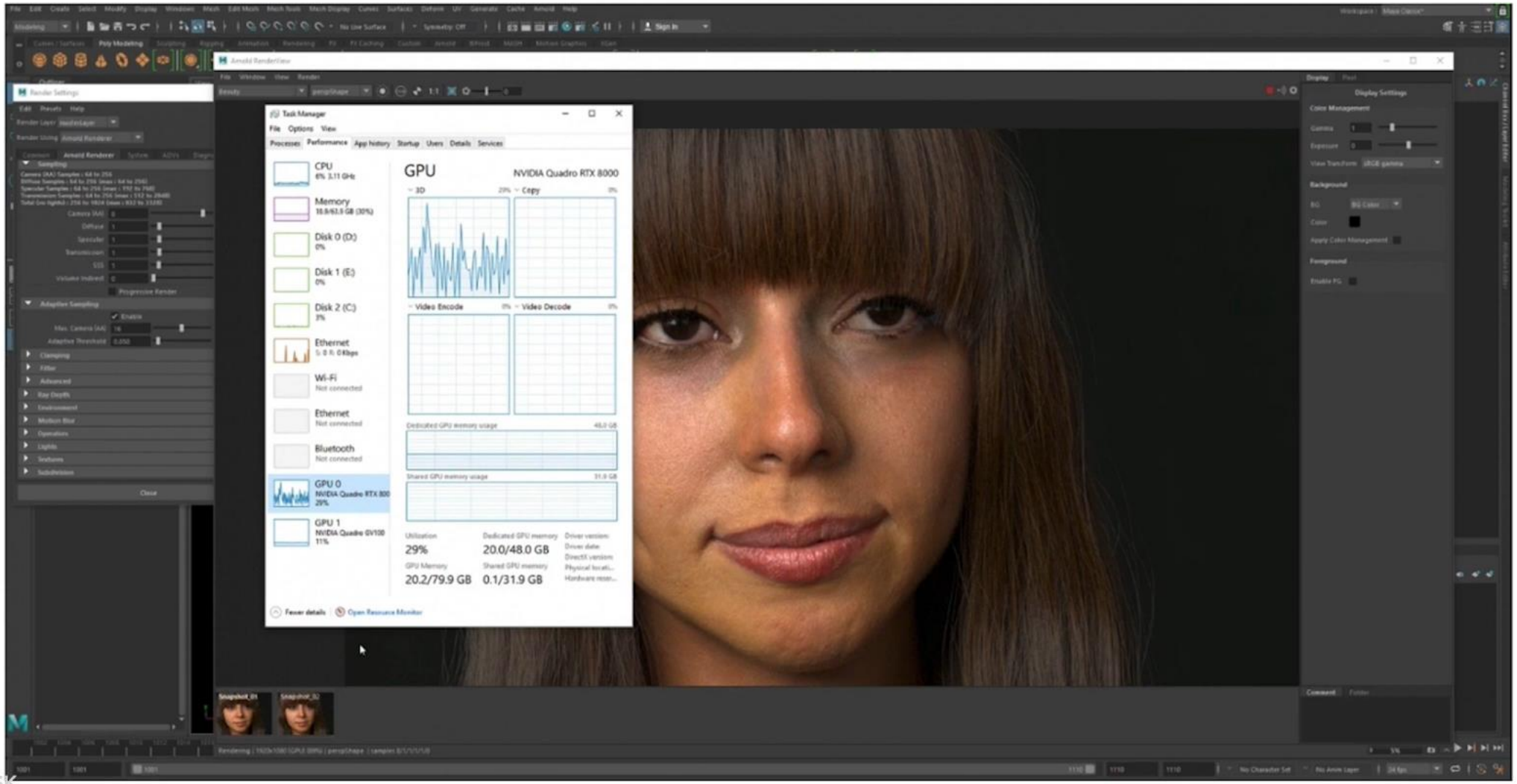
RTX Speedup

- Traditionally 30%-50% time is tracing rays
- RTX makes tracing negligible
- Up to 2x speed-up
- Render time is now mostly shading



Arnold GPU Beta on OptiX

Goal: feature parity



OPTIX / ARNOLD COLLABORATIVE FEATURES

A fruitful engagement

Demand-loaded buffers and textures for large scenes

Separable compilation for large / many shaders

Memory reduction for efficient resource utilization

Stack memory optimizations

Call-graph optimizations

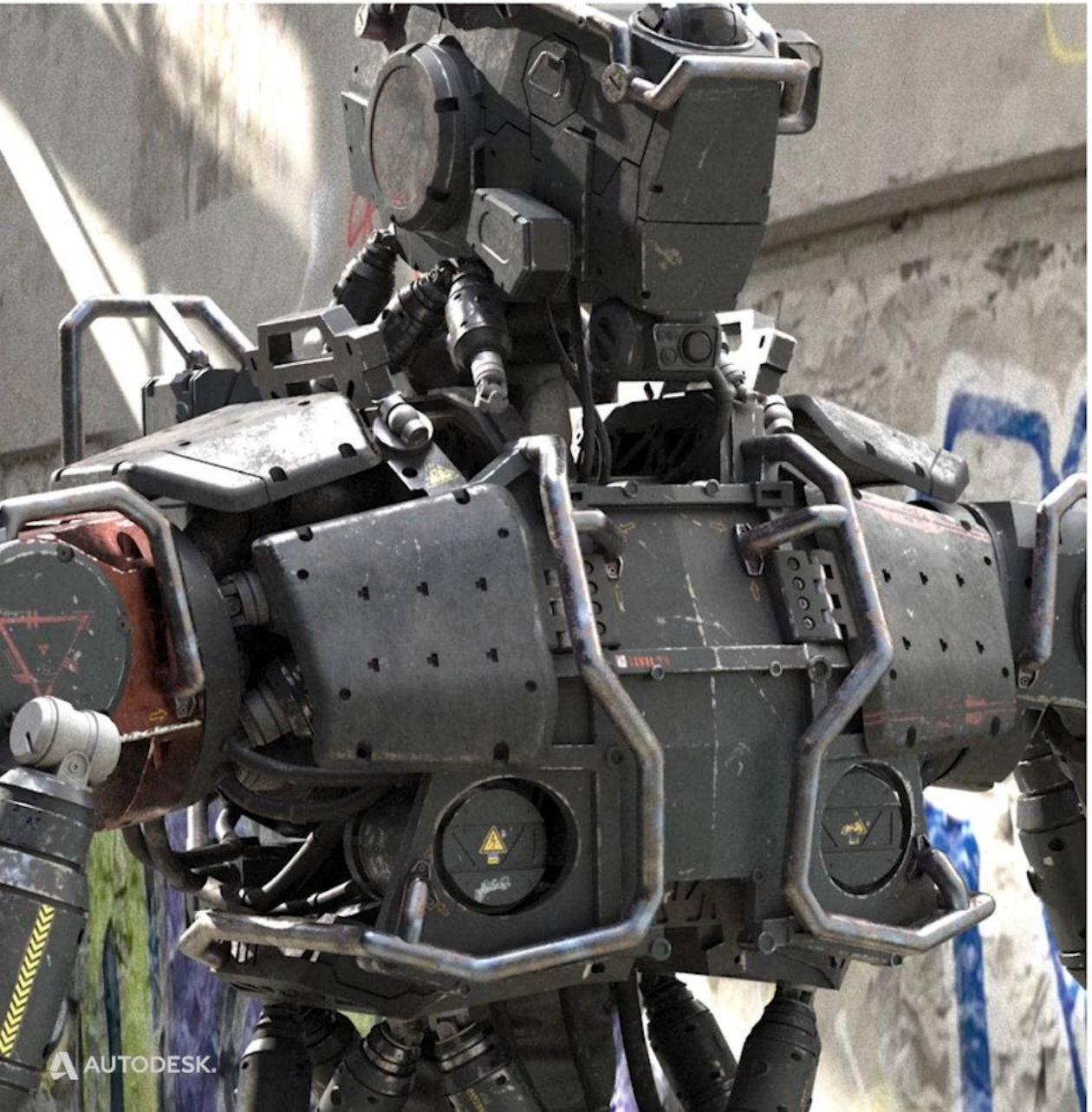
Memory reduction

Performance optimization

AI denoiser optimization

Testing, testing, testing

Initial results courtesy AutoDesk



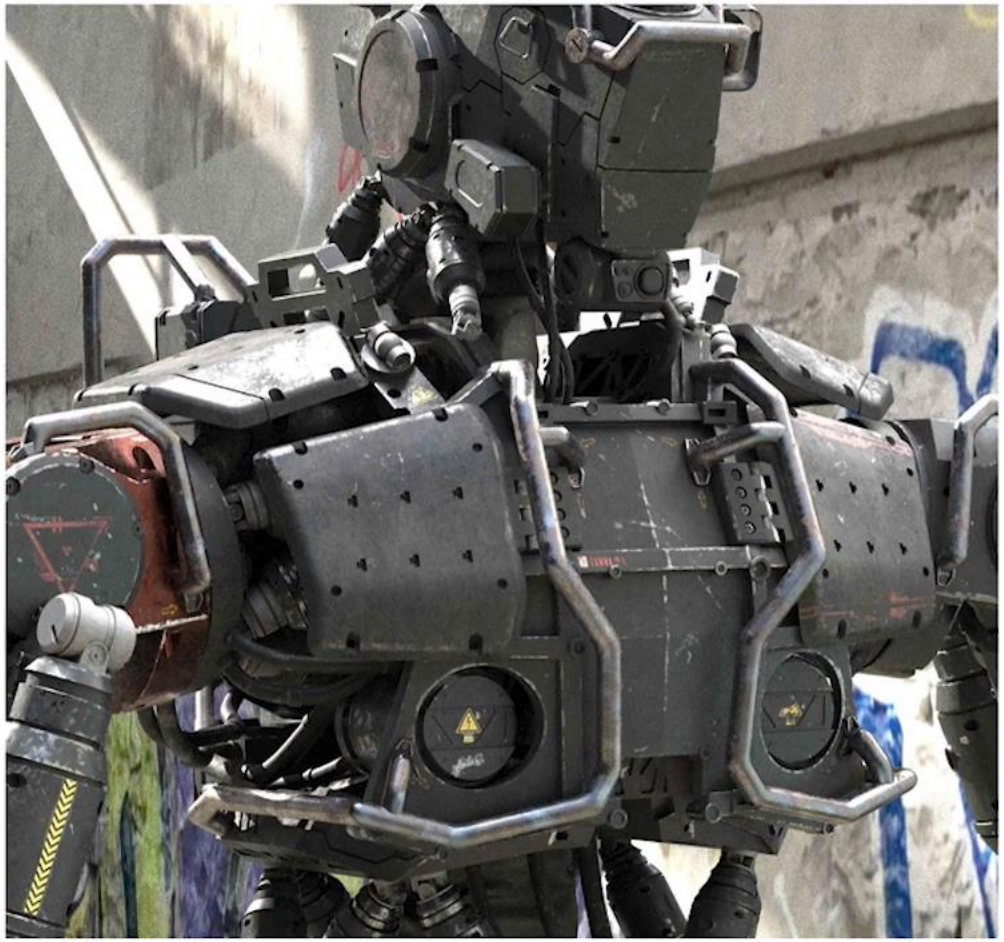
Initial results courtesy AutoDesk

Near cam		Pre-load textures	On-demand textures
	Time to first pixel (mm:ss)	1:57	0:14
	Time to completion (mm:ss)	2:29	0:59
	GPU memory usage (MiB)	17040	12878
Far Cam			
	Time to first pixel (mm:ss)	1:57	0:12
	Time to completion (mm:ss)	2:03	0:33
	GPU memory usage (MiB)	17040	6386

Texture on-demand loading

WIP

	Load all textures	**On-demand loading**
Time to first pixel	01:57	**00:14**
Render time	02:29	**00:59**
GPU Memory (MiB)	17040	**12878**



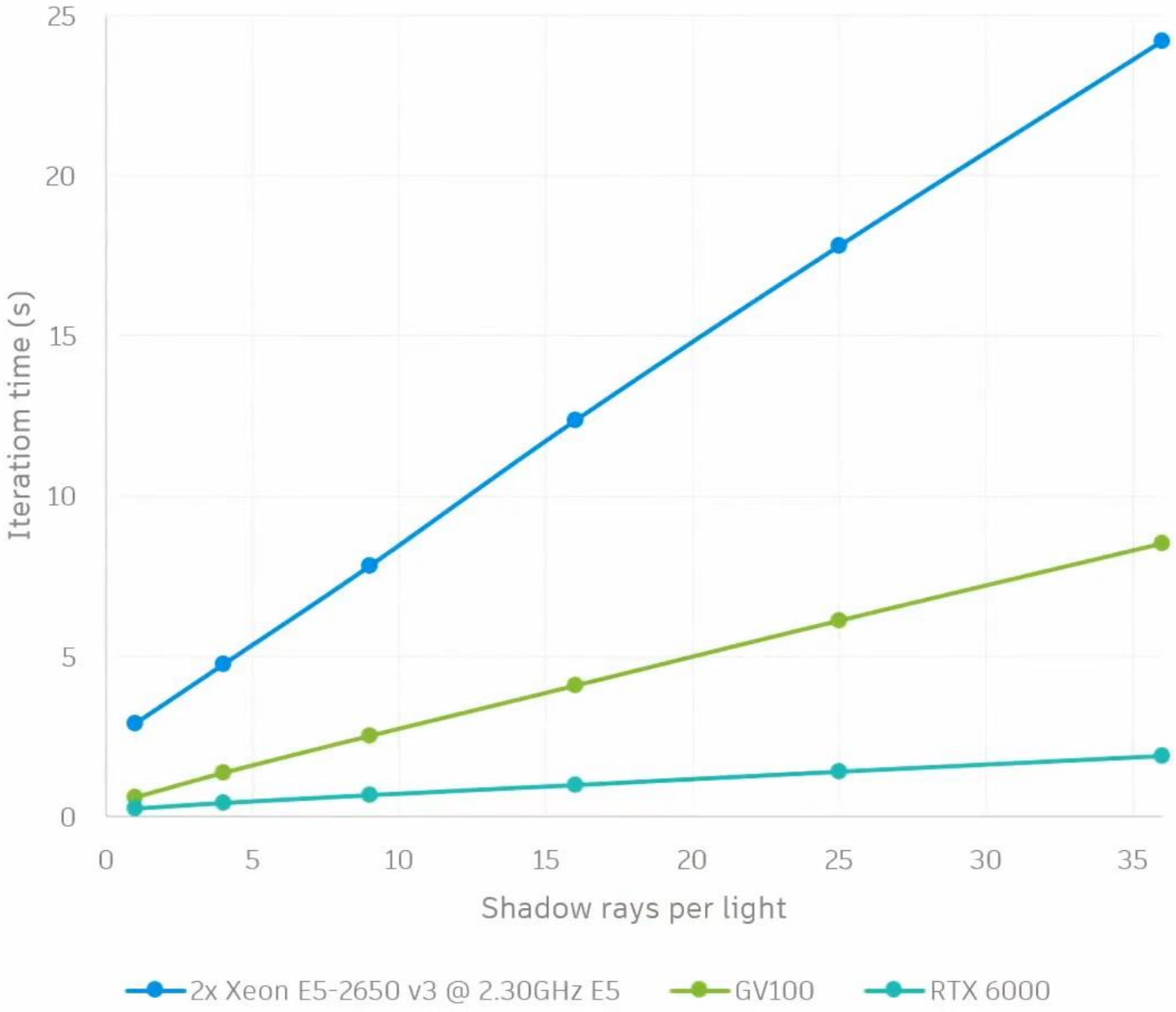
Texture on-demand loading

WIP

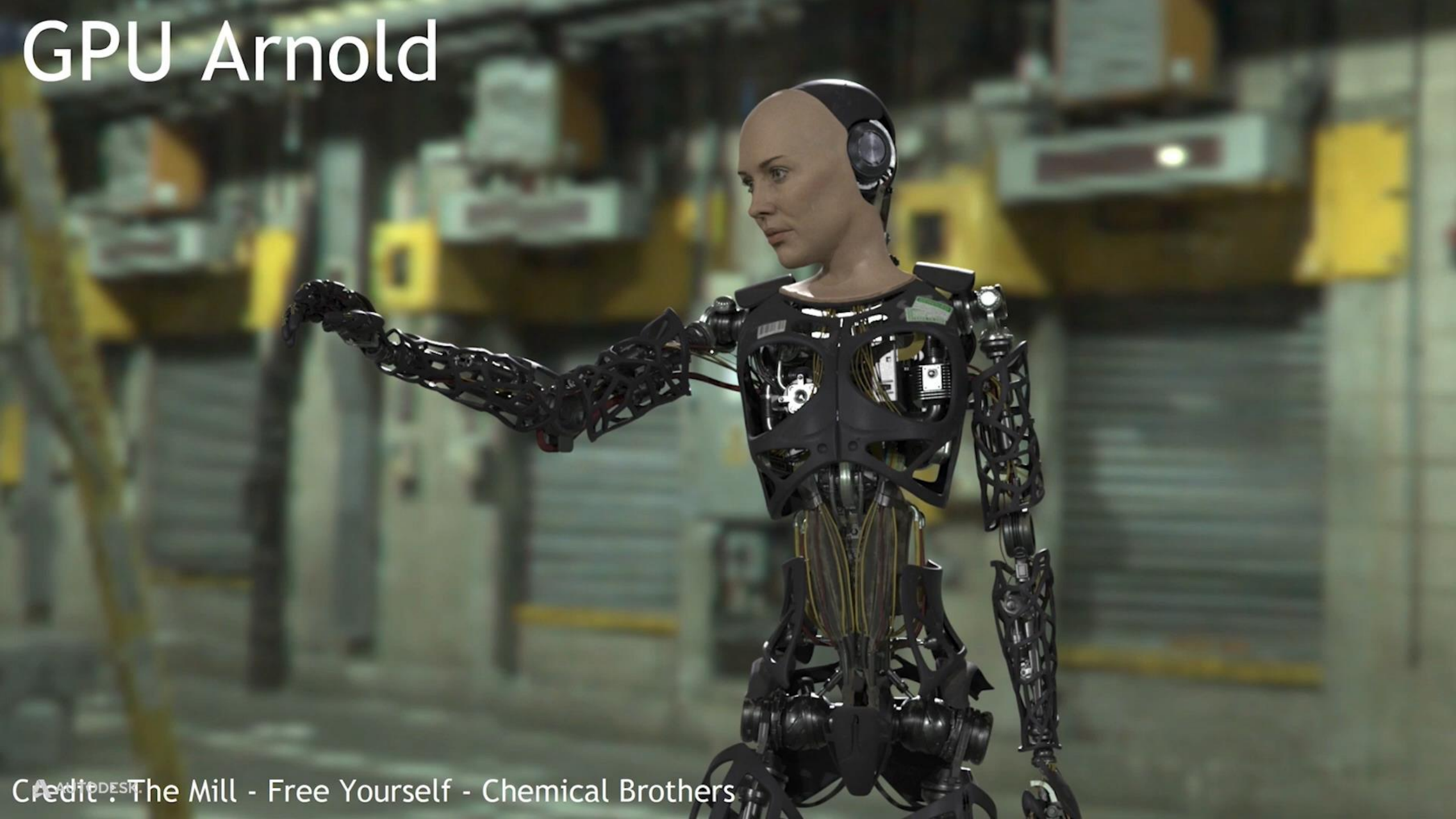
	load all textures	**On-demand loading**
Time to first pixel	01:57	**00:12**
Render time	02:03	**00:33**
GPU Memory (MiB)	17040	**6386**

RTX Experiments
Increasing shadow rays

- Promising results for direct specular
- Cheap rays unlocking new research



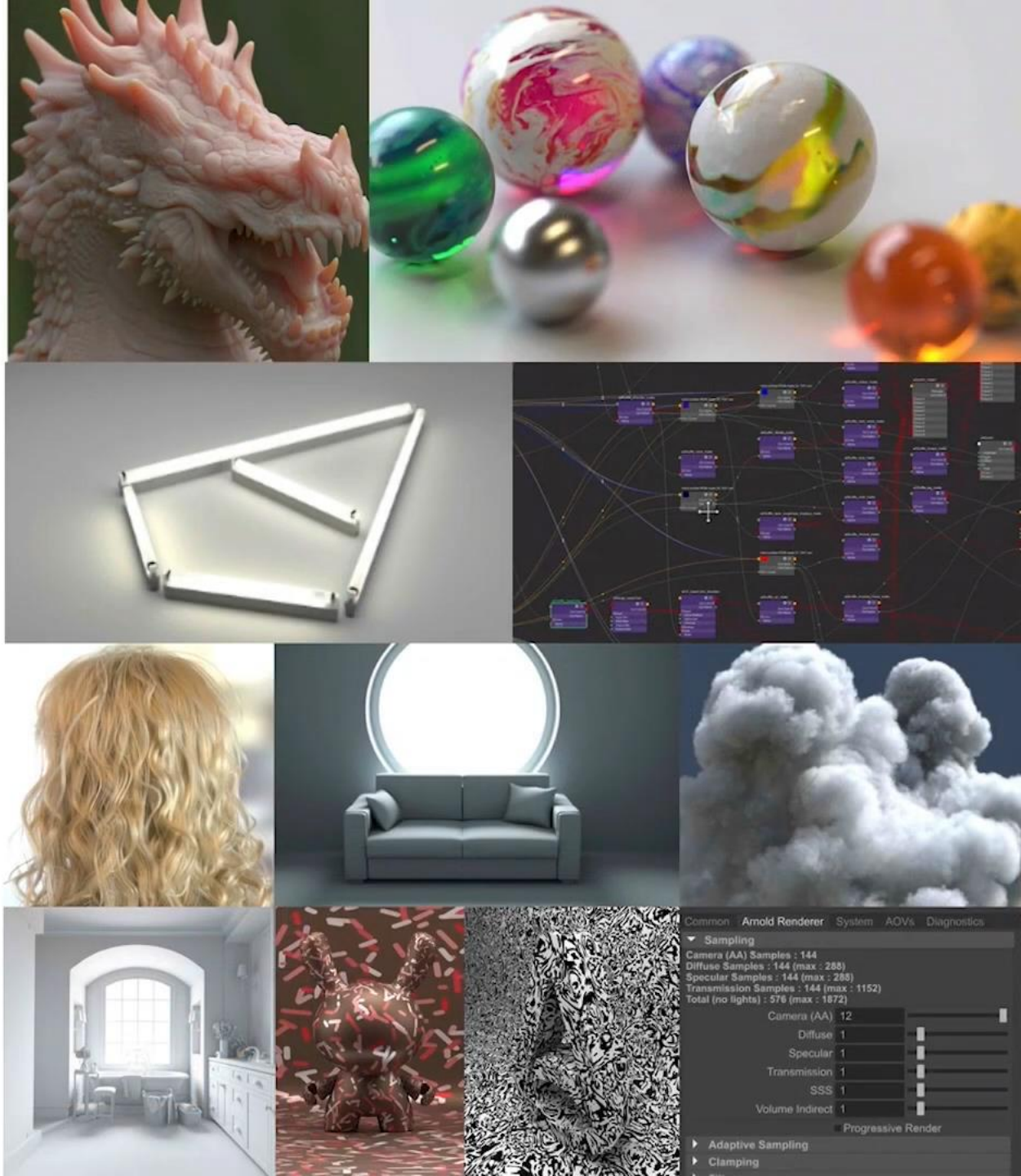
GPU Arnold



Credit: The Mill - Free Yourself - Chemical Brothers

Feature Compatibility

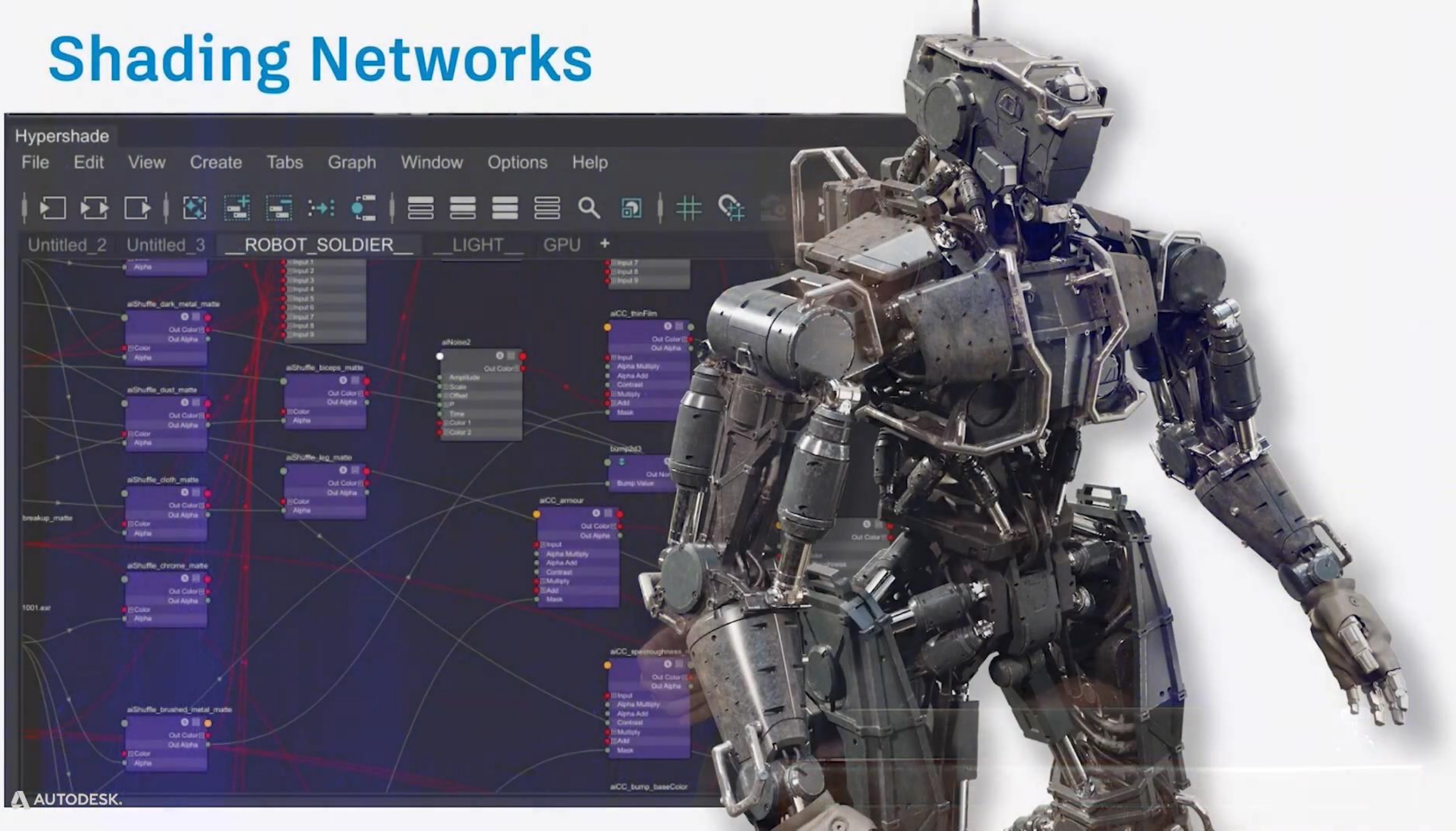
- Final frame renderer
- Support all features
- Some features are hard
- Arnold is ~~very~~ too flexible/customizable
- Focus on character lookdev first
- Eventually support everything

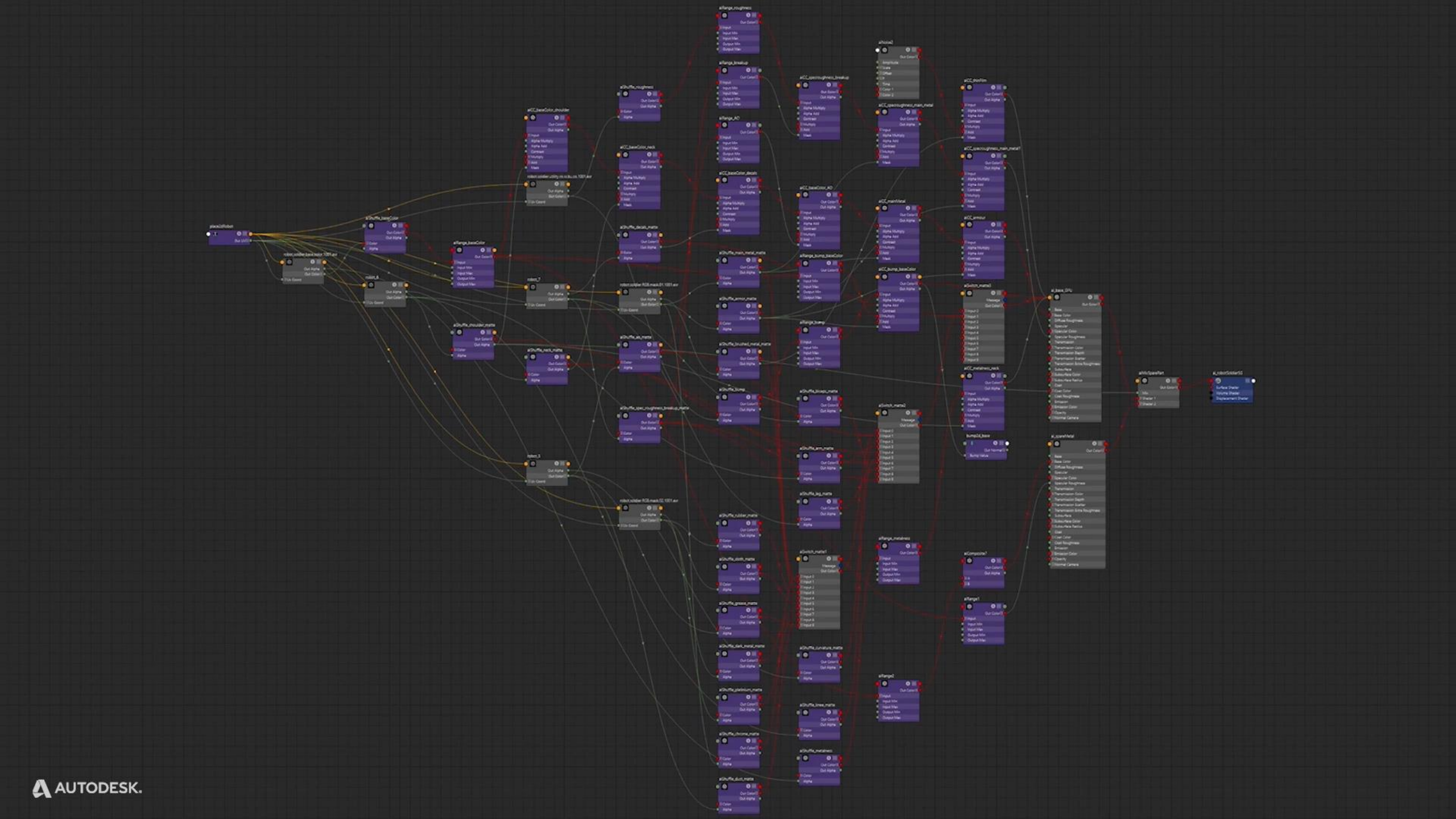


Pixel Compatibility

- Levels of compatibility
  - Good enough for lookdev ← Beta
  - A device per shot ← First release
  - A device per frame
  - Hybrid ← Eventually
- Performance tradeoffs
  - Hardware texturing
  - Filtering
  - Splitting

Shading Networks



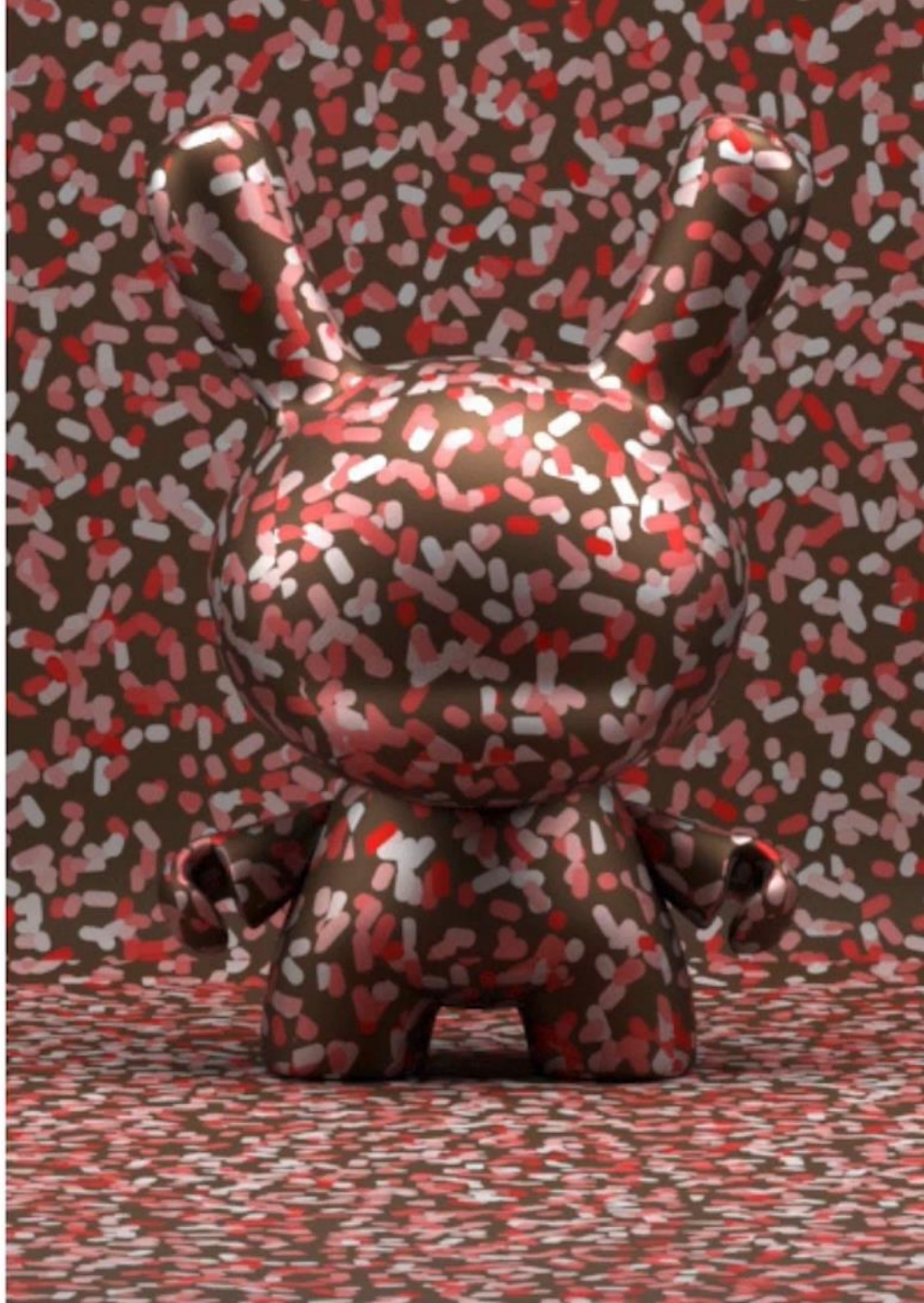


OSL on GPU

- Introducing OSL support on GPU in 5.4

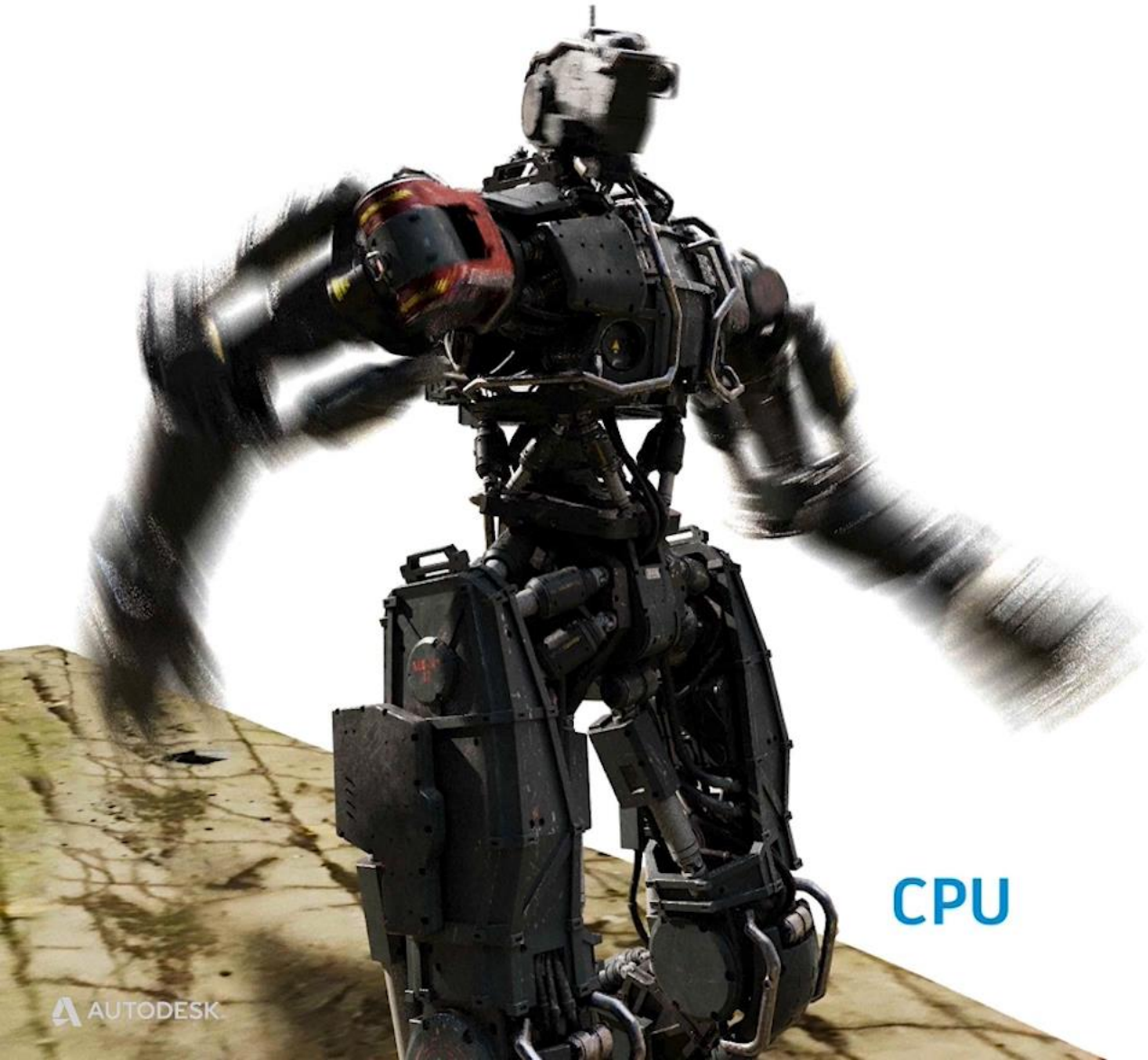
Coming later

- OSL textures
- More transforms
- OSL



Motion Blur

CPU vs GPU

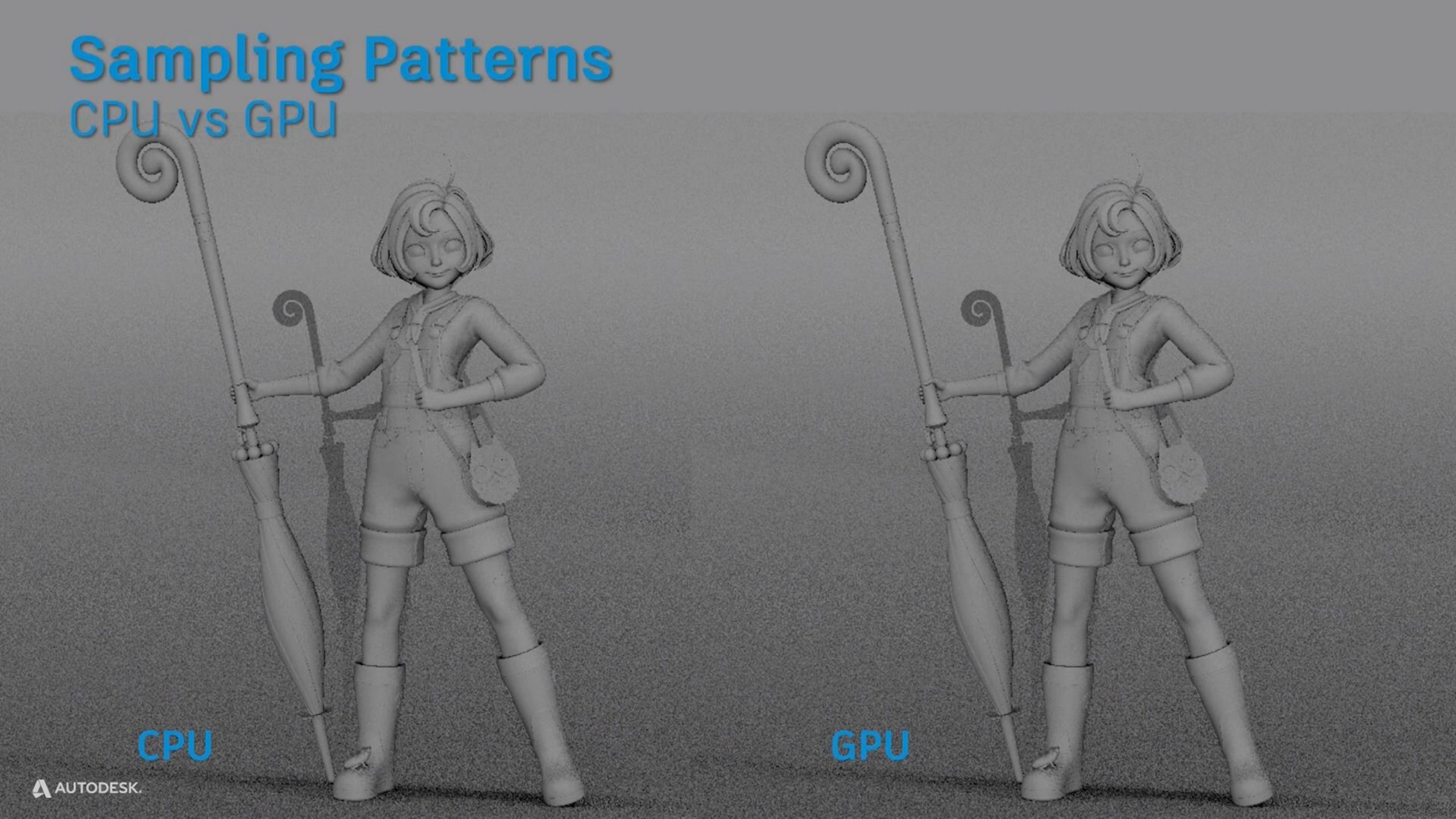


Motion Blur

CPU vs GPU

Sampling Patterns

CPU vs GPU



Lights Sampling

CPU vs GPU

AUTODESK.

CPU vs GPU

GTC Robot

CPU

GPU

AUTODESK.

CPU vs GPU

Nova Girl

CPU

GPU

AUTODESK.

CPU vs GPU
Cloth

AUTODESK.

CPU vs GPU

Cloth

Improved Adaptive Sampling

CPU vs GPU Benchmarks

CPU 6-3-3-3-3-0 15 min 14 sec 1k render

GPU 20-1-1-1-1-0 2 min 56 sec 1k render

CPU vs GPU Benchmarks

RTX Benchmarks

CPU vs GPU

Cloth

CPU

GPU

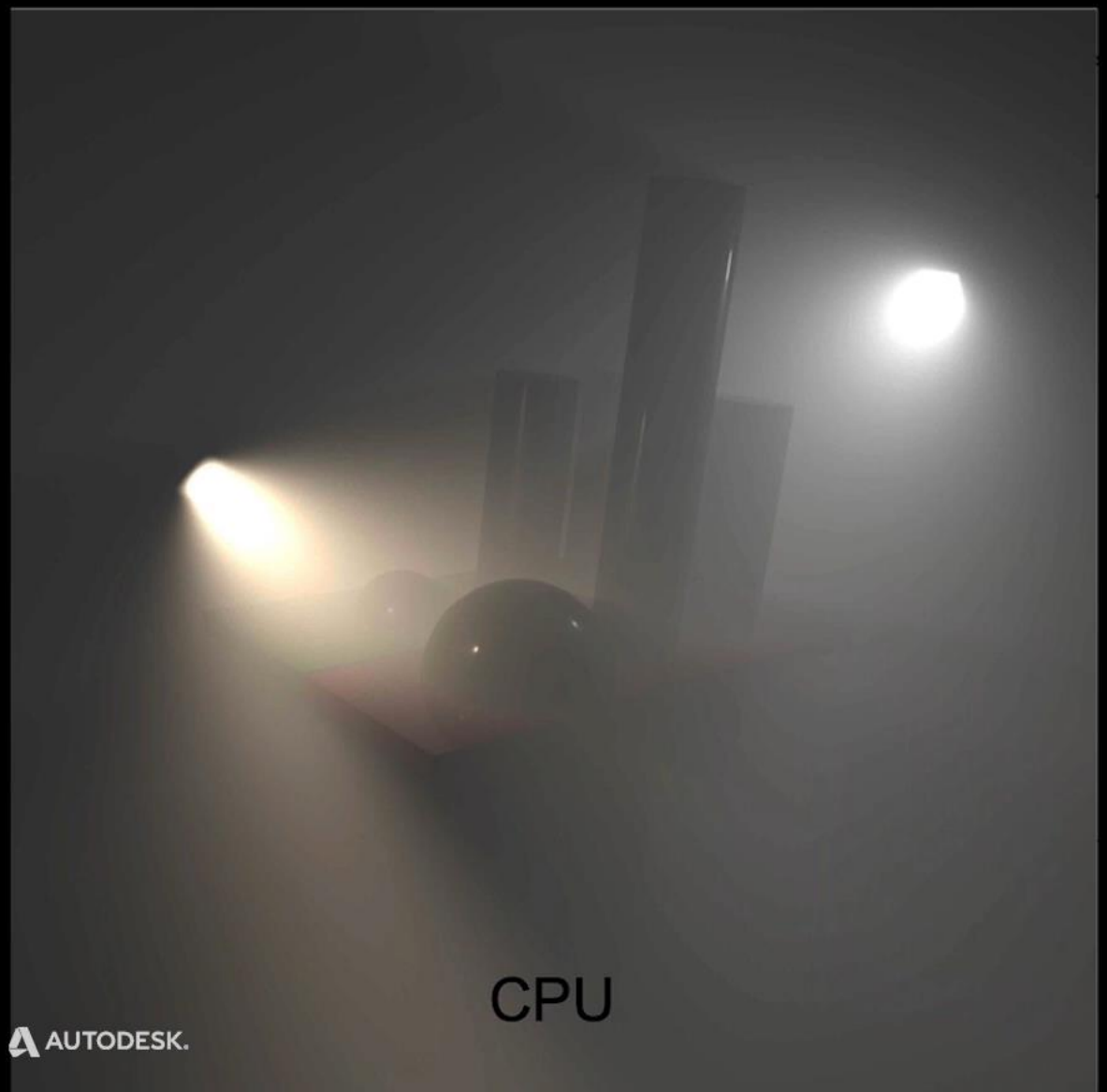
AUTODESK.

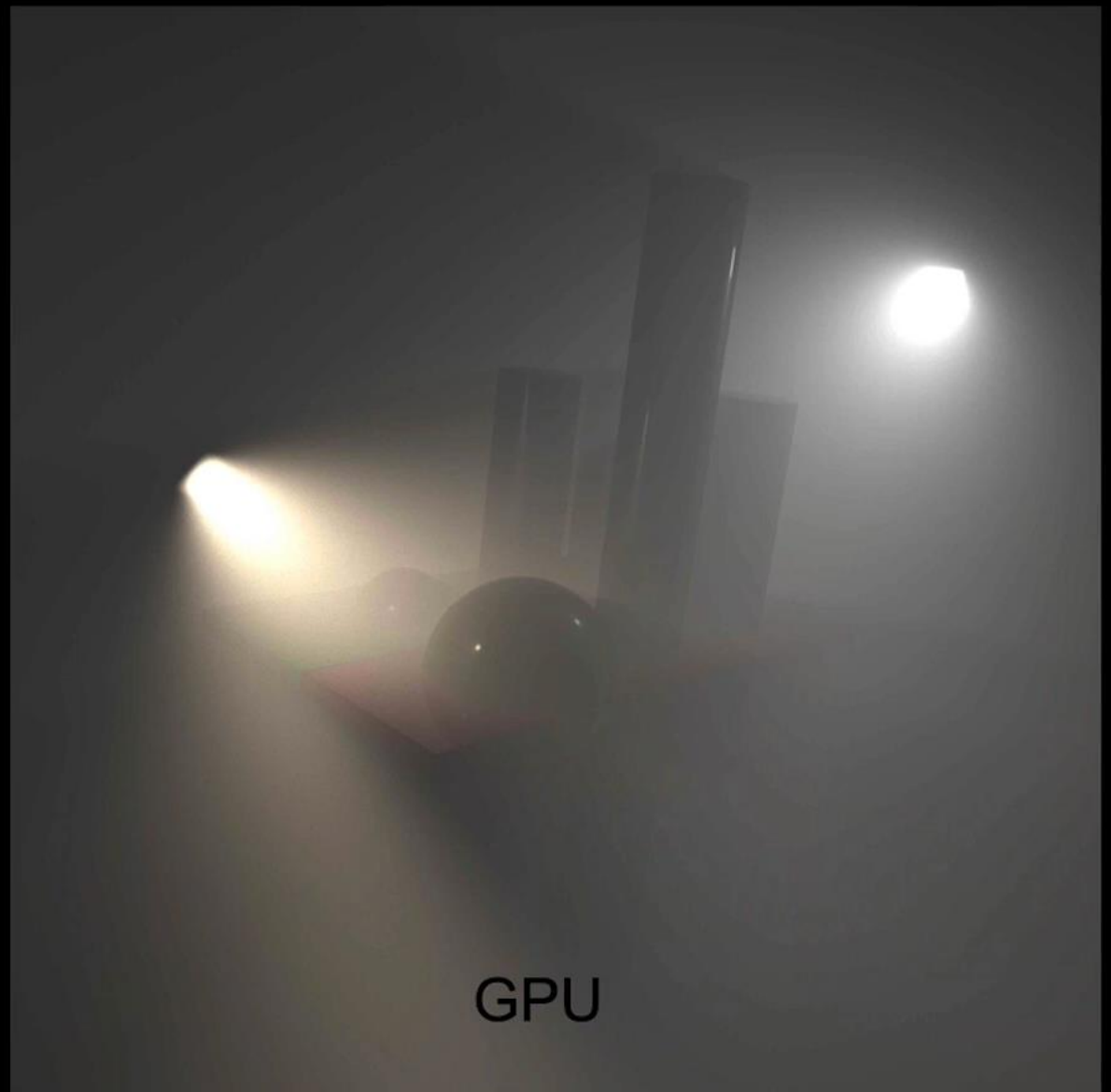
CPU vs GPU

Chad Ashley

AUTODESK.

GPU Atmospherics





GPU Atmospherics

RandomWalk v2

Improved Adaptive Sampling

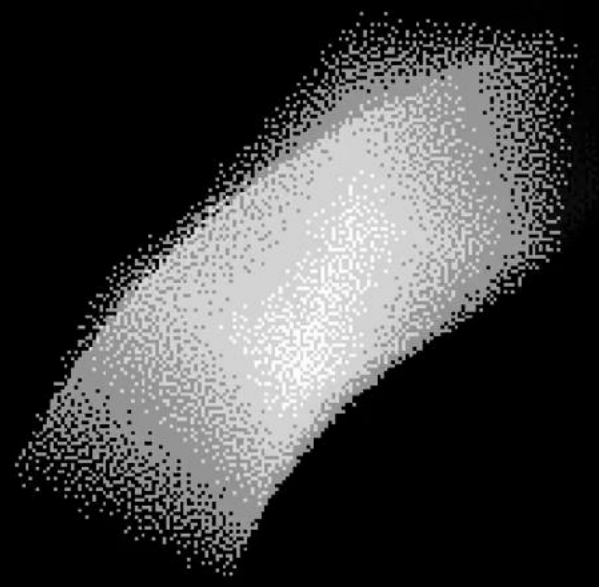
RTX-6000 GPU DOF camAA - 20 1 min 53 sec

Improved Adaptive Sampling

RTX-6000 GPU camAA - 20 1080p 2min 15 sec

GPU standard_hair

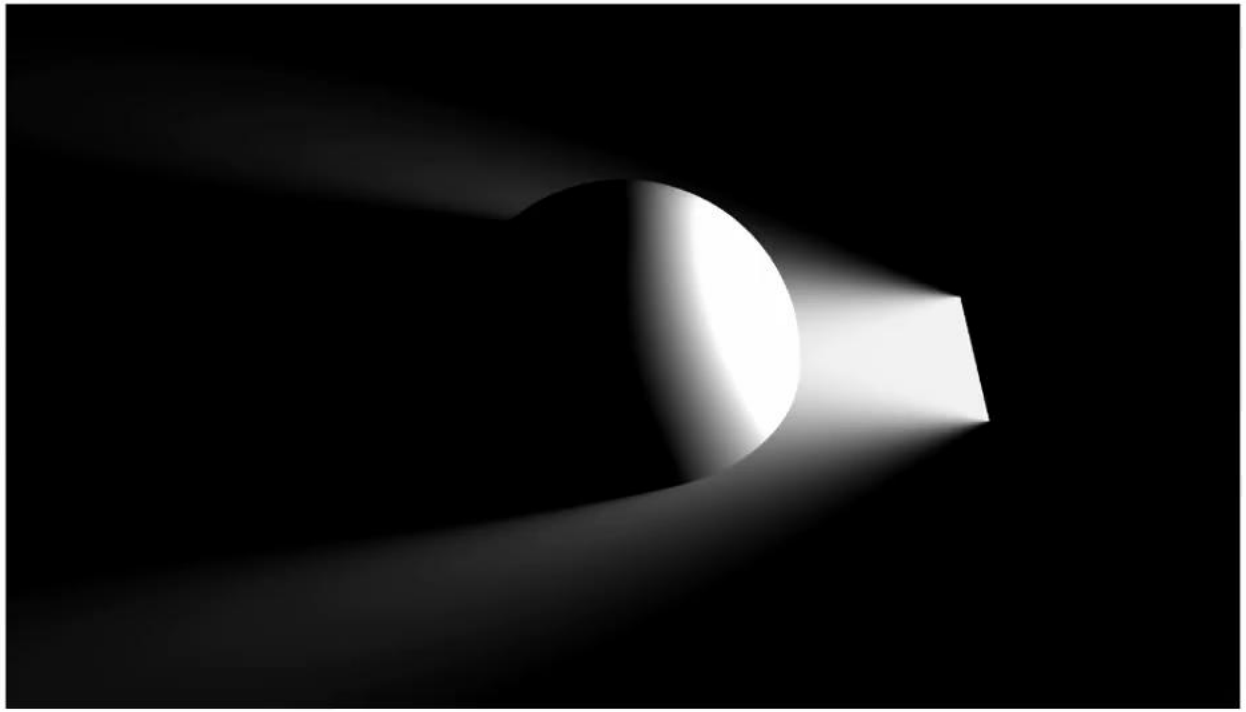
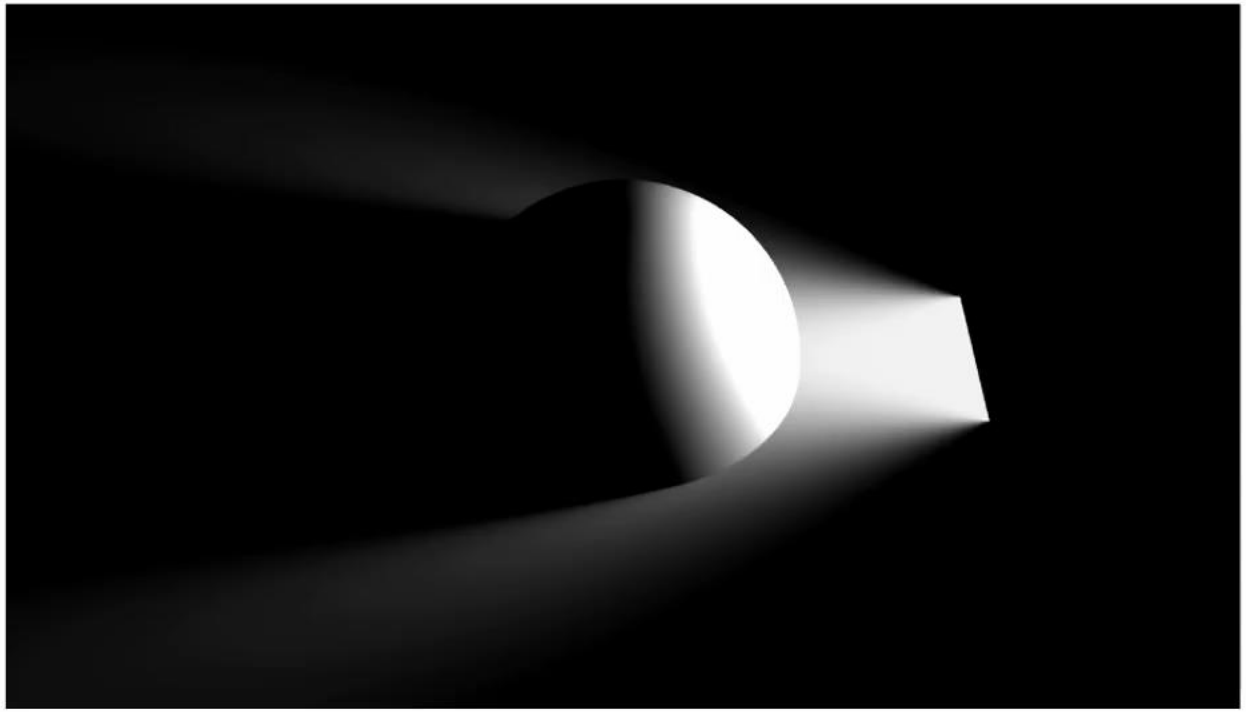
Transformation Motion Blur
CPU vs GPU



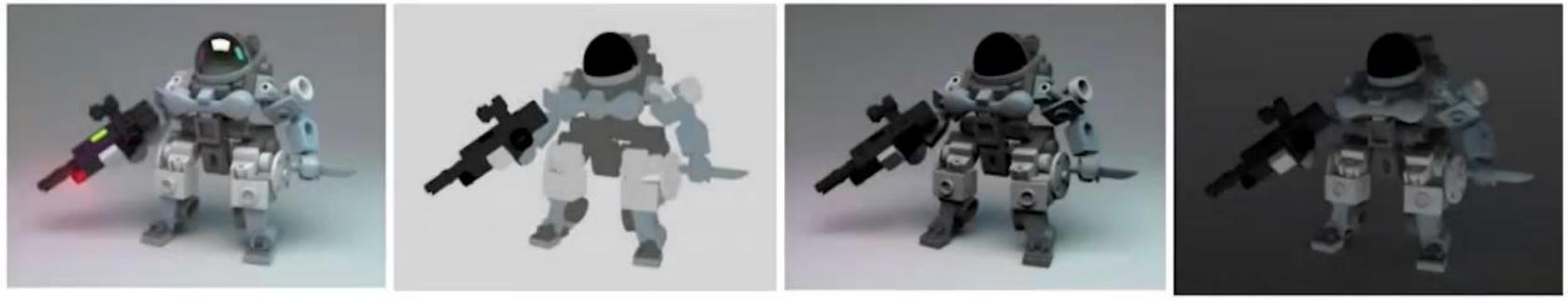
CPU

GPU

Light Sampling

CPU	GPU
	

Light Path Expressions



Beauty: C.*

Diffuse Albedo: C<RD>A

Diffuse Direct: C<RD>L

Diffuse Indirect: C<RD>[DSVOB].*

Diffuse: C<RD>.*

Direct: C[DSV]L

Emission: C[LO]

Indirect: C[DSV][DSVOB].*

Specular Direct: C<RS>L

Specular Indirect: C<RS>[DSVOB].*

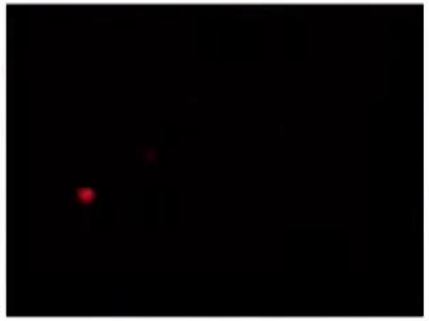
Specular: C<RS>.*

SSS Albedo: C<TD>A

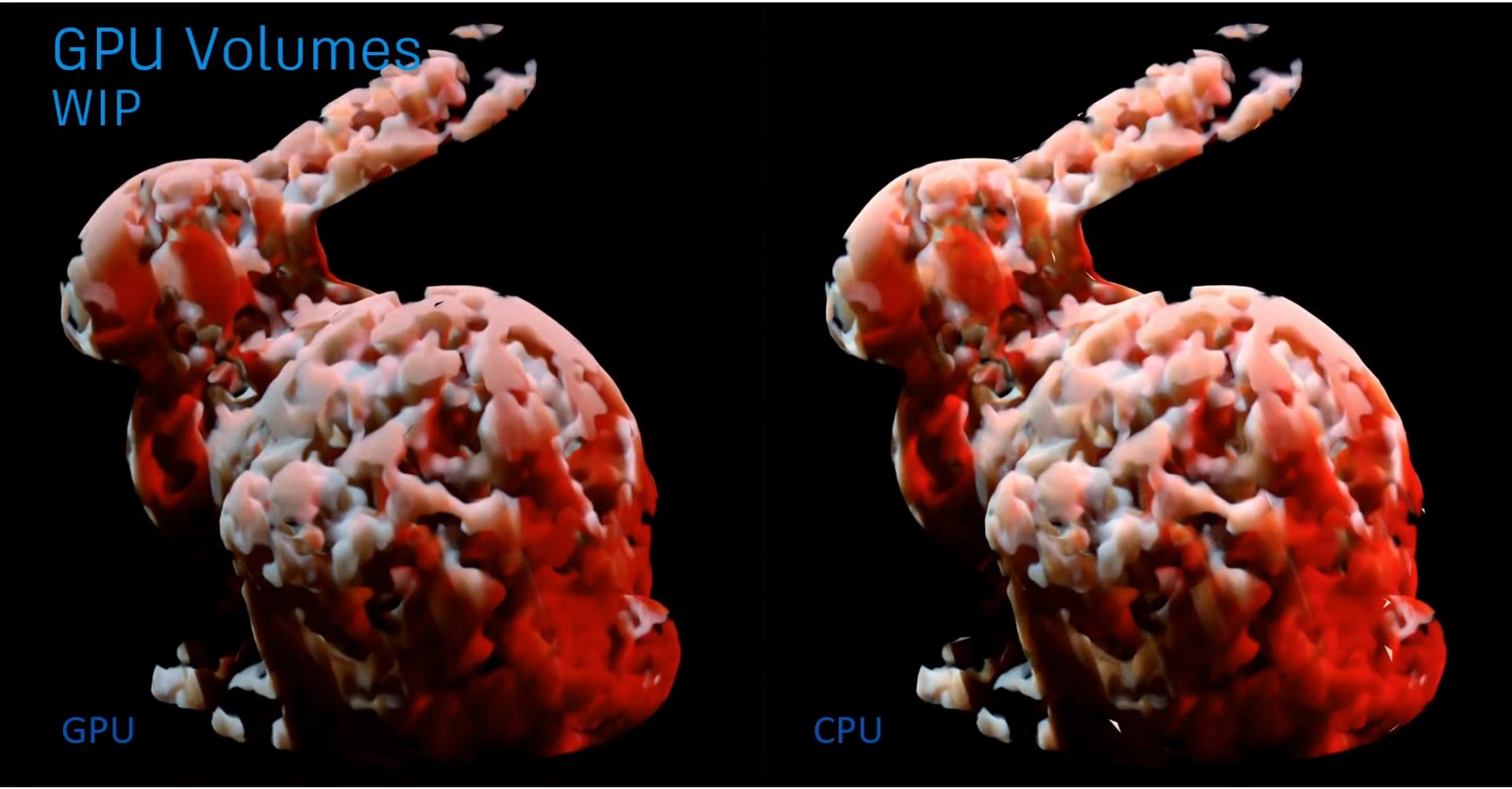
Direct: C[DSV]L

Indirect: C[DSV][DSVOB].*

SSS: C<TD>.*

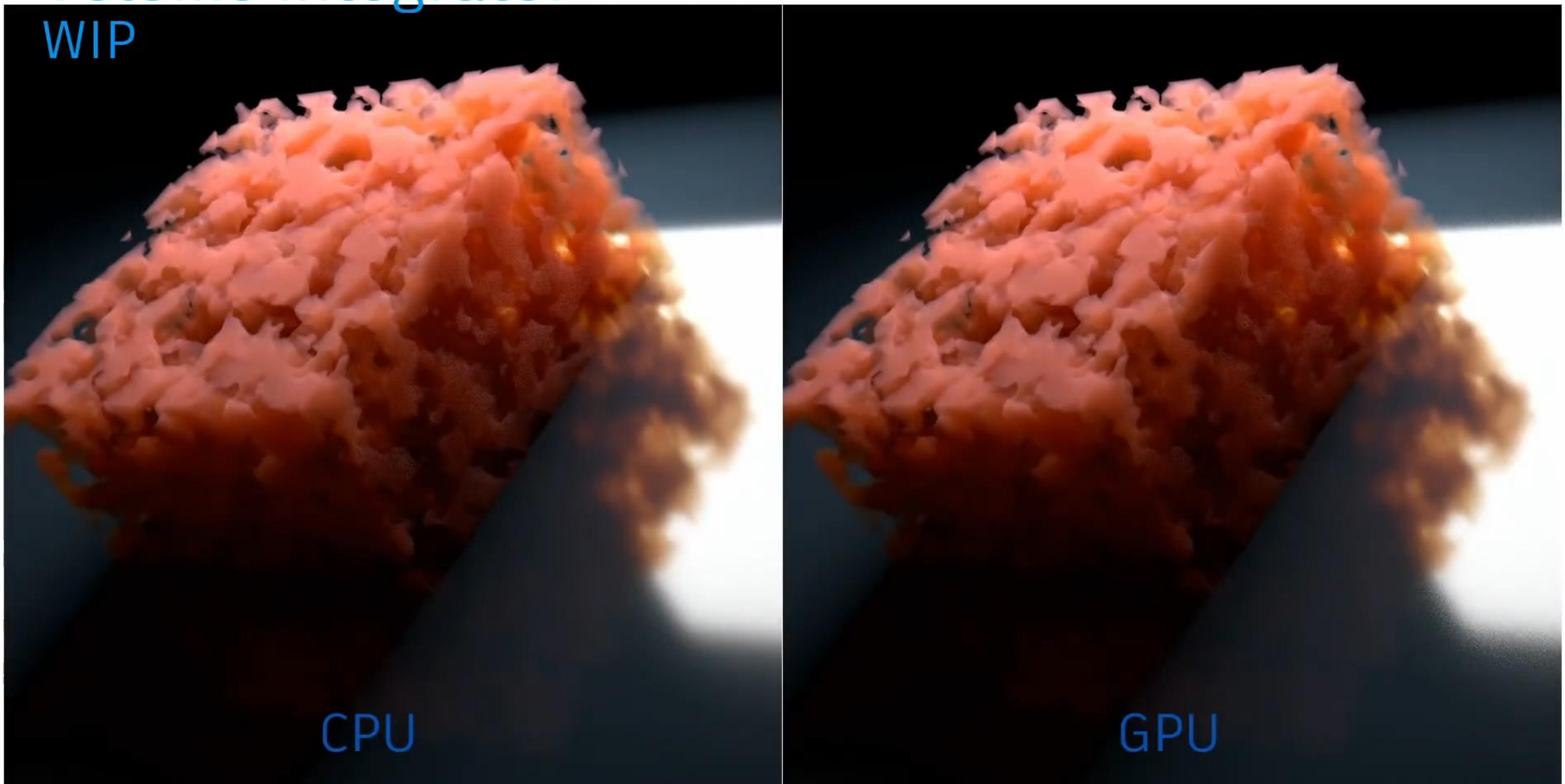


Transmission: C<TS>.*

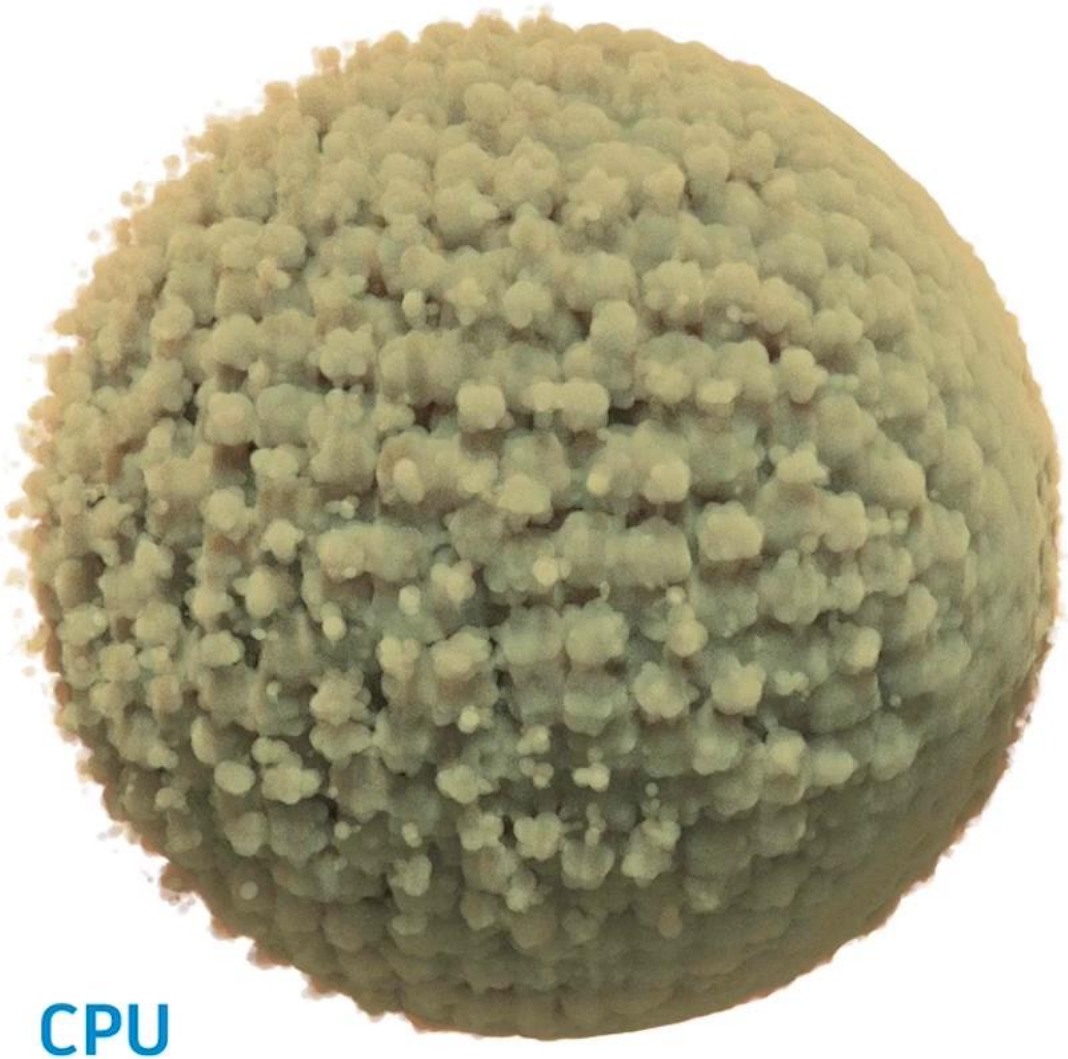

GPU Volumes
WIP
GPU
CPU

Volume Integrator

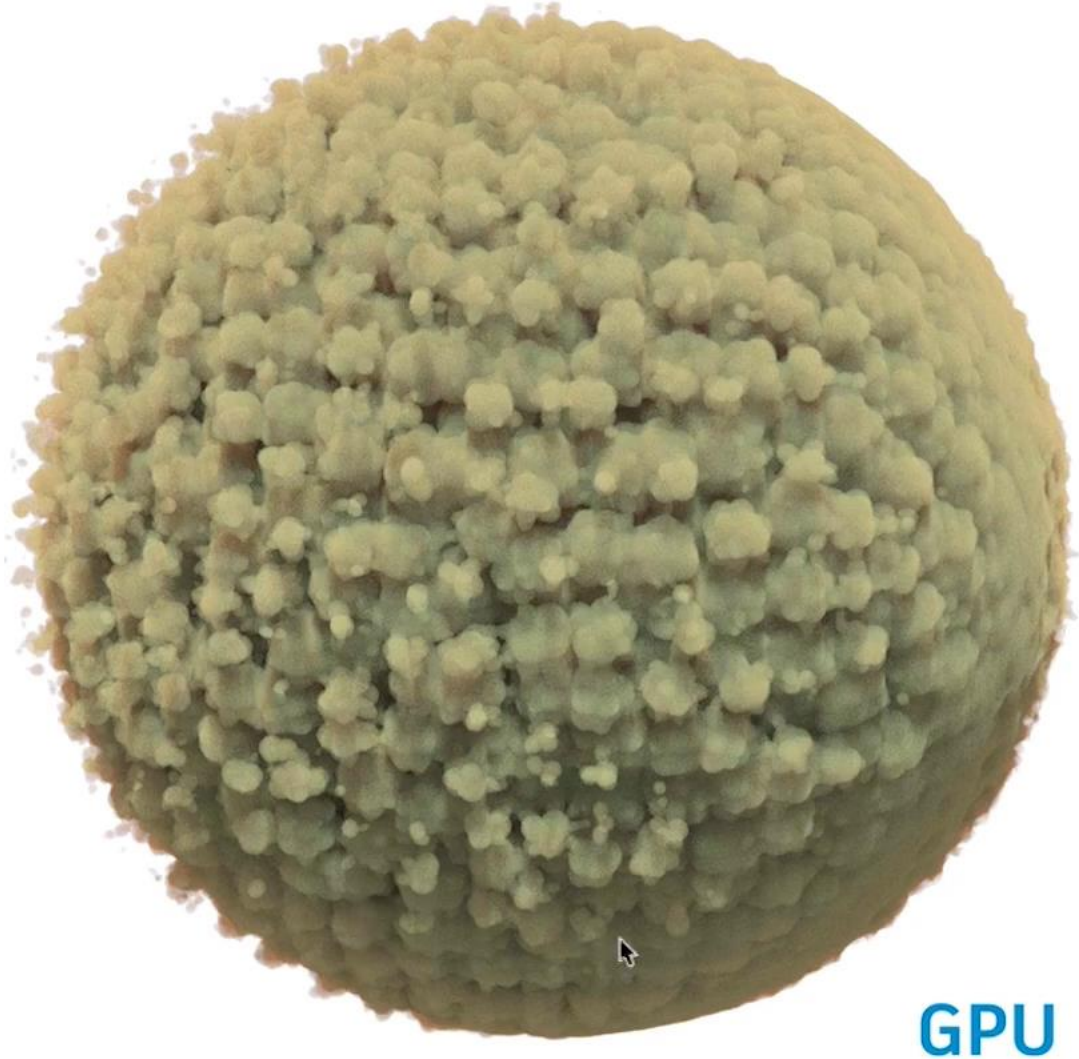
WIP



Volume Displacement



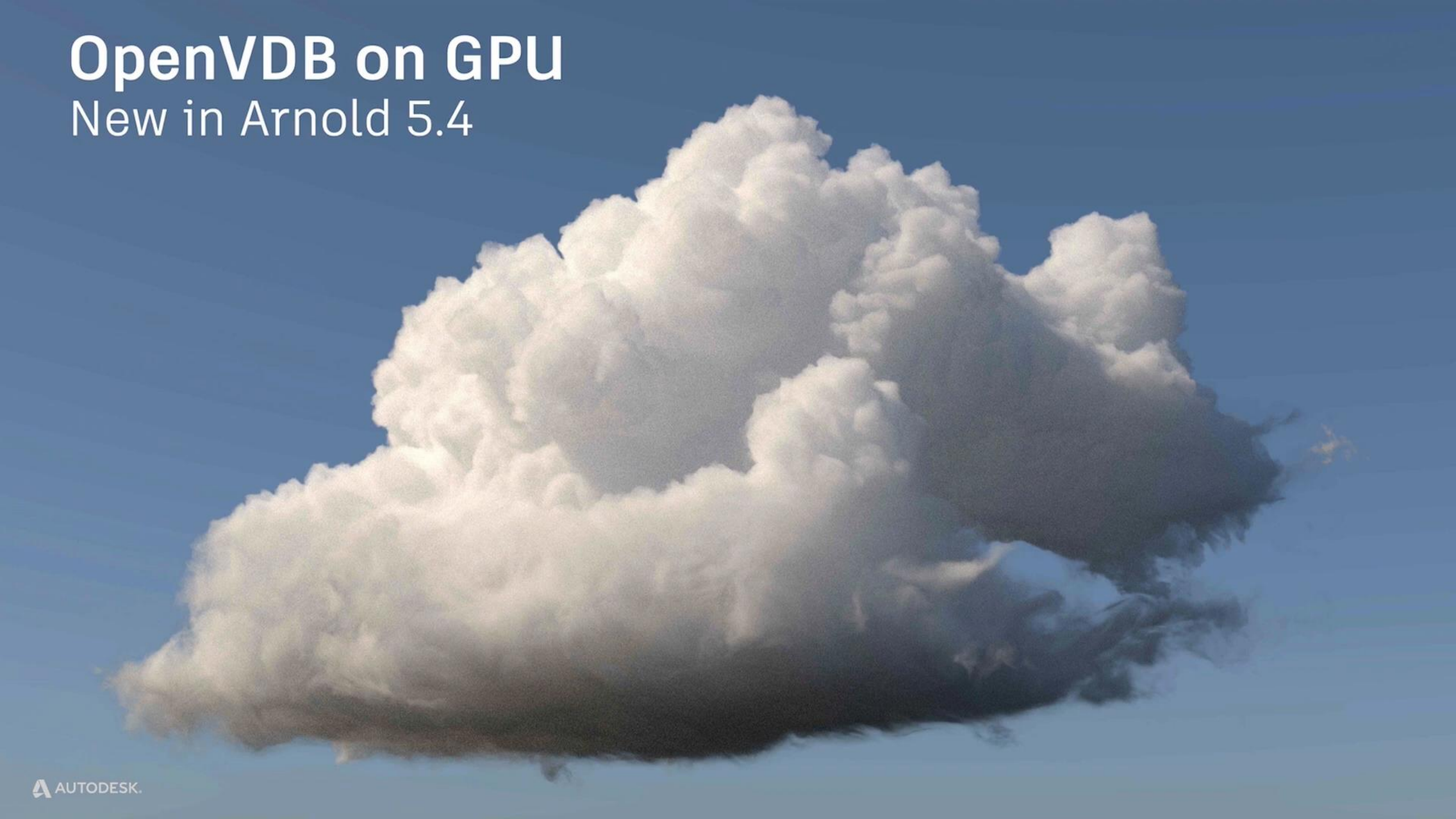
CPU



GPU

OpenVDB on GPU

New in Arnold 5.4

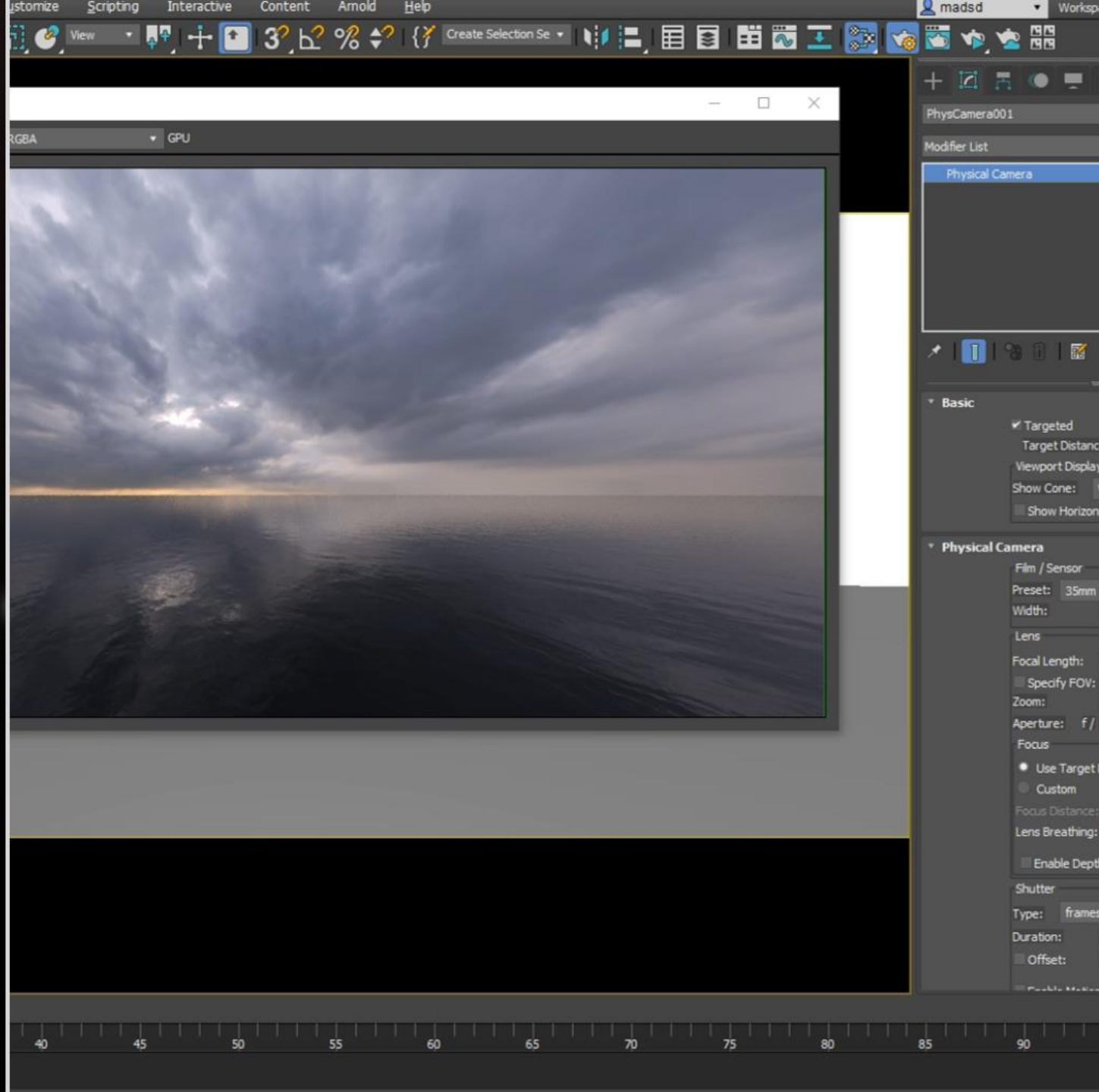


AUTODESK.

Arnold GPU

NEW FEATURES & SUPPORT

- Improved Open Shading Language (OSL) and OpenVDB support
- On-demand texture loading
- New Shadow Matte shader
- Support for most LPEs, lights, and shaders
- Support for all cameras
- Reduced noise



API Compatibility

- Arnold is a library
- Binary compatible
- Device selection API

```
#include <ai_api.h>

int main(int argc, char* argv[])
{
   int error;

   // start arnold API
   AiBegin();

   // load scene
   error = AiASSLoad(argv[1]);

   // switch to GPU rendering
   AiNodeSetUInt(AiUniverseGetOptions(), "render_device", 1);

   // select the second GPU
   AtArray* ids = AiArray(1, 0, AI_TYPE_UINT, 1);
   AiDeviceSelect(AI_DEVICE_TYPE_GPU, ids);

   // render
   AiRender(AI_RENDER_MODE_CAMERA);

   // close down API
   AiEnd();

   return 0;
}
```

Shader API

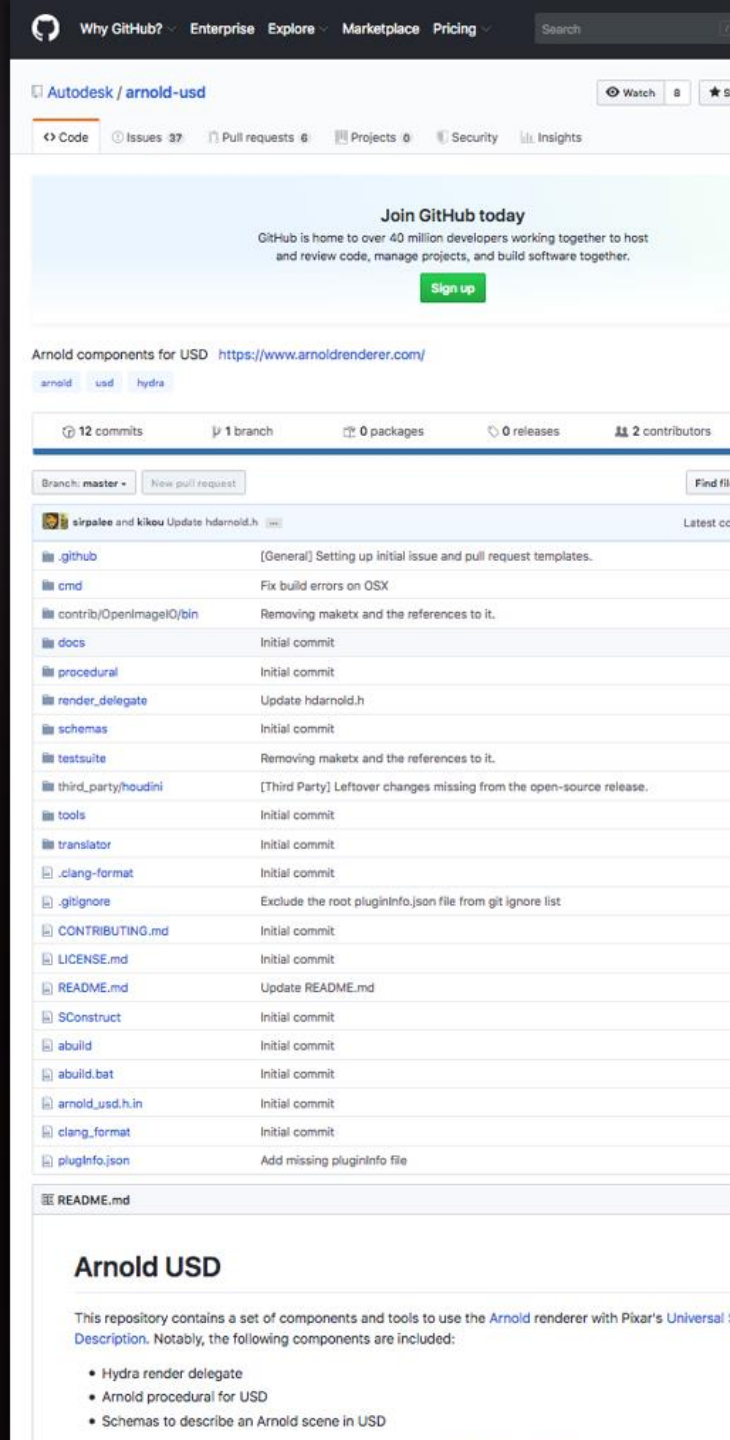
- Arnold C API compatibility
- Shared shader codebase
- Arnold API generates PTX
- Third-party shader support is in sight

```
shader_evaluate
{
    int mode = AiShaderEvalParamEnum(p_mode);

    if (mode == AI_SHADER_MIX_BLEND)
    {
        float mix = AiClamp(AiShaderEvalParamFlt(p_mix), 0.0f, 1.0f);
        float negmix = 1.0f - mix;

        if (mix < AI_EPSILON)
        {
            sg->out.CLOSURE() = AiShaderEvalParamClosure(p_shader1);
        }
        else if (negmix < AI_EPSILON)
        {
            sg->out.CLOSURE() = AiShaderEvalParamClosure(p_shader2);
        }
        else
        {
            AtClosureList shader1 = AiShaderEvalParamClosure(p_shader1);
            AtClosureList shader2 = AiShaderEvalParamClosure(p_shader2);

            shader1 *= negmix;
            shader2 *= mix;

            shader1.add(shader2);
            sg->out.CLOSURE() = shader1;
        }
    }
    else
    {
        AtClosureList shader1 = AiShaderEvalParamClosure(p_shader1);
        AtClosureList shader2 = AiShaderEvalParamClosure(p_shader2);

        shader1.add(shader2);
        sg->out.CLOSURE() = shader1;
    }
}
```

Arnold 6

USD Components

Now Available on GitHub



Shared Scene

- Scene is shared between CPU/GPU
  - Procedural
  - Instancing
  - Operators
- No overhead if GPU is not used
- Live device switching

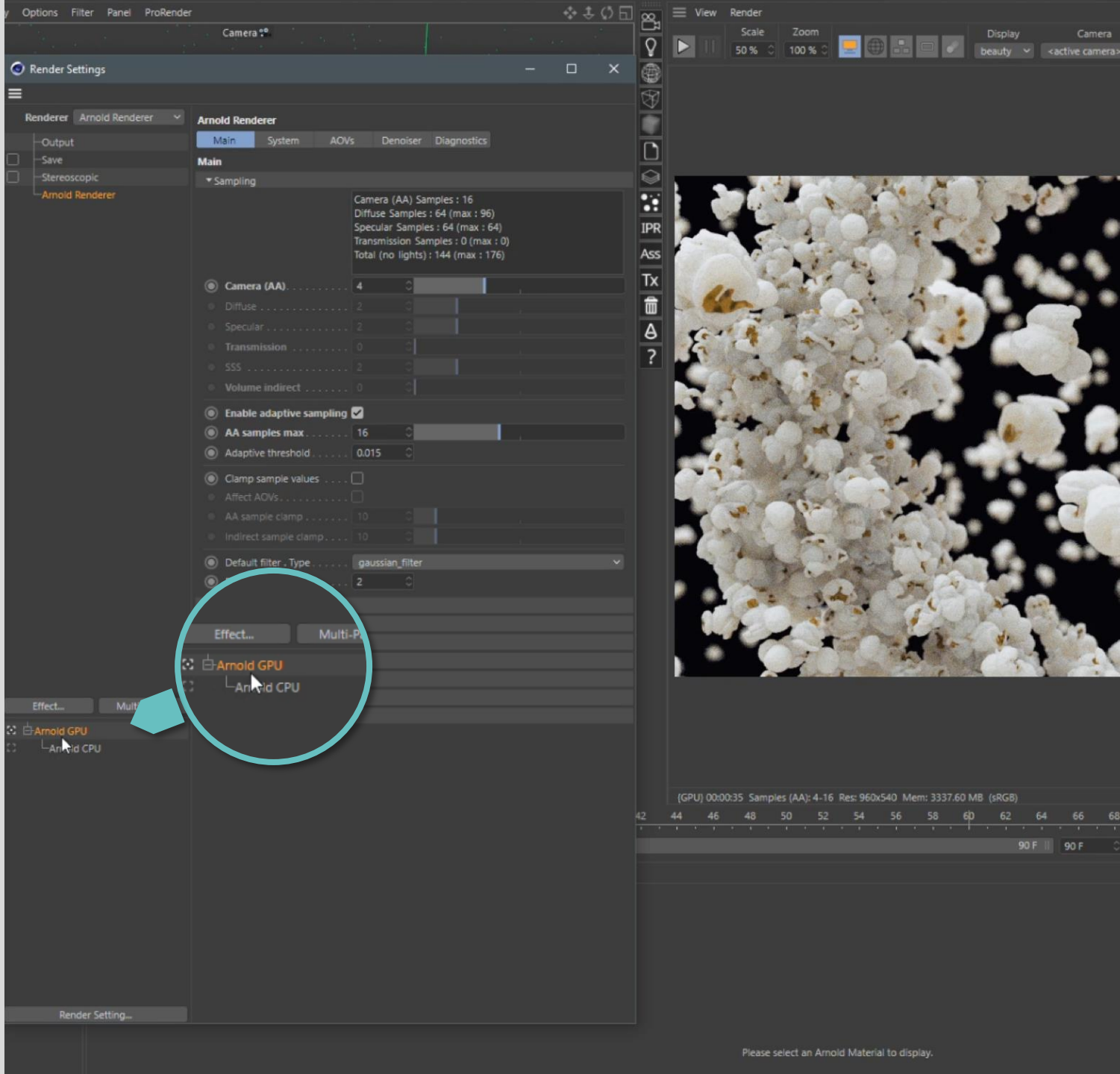
Shared Scene

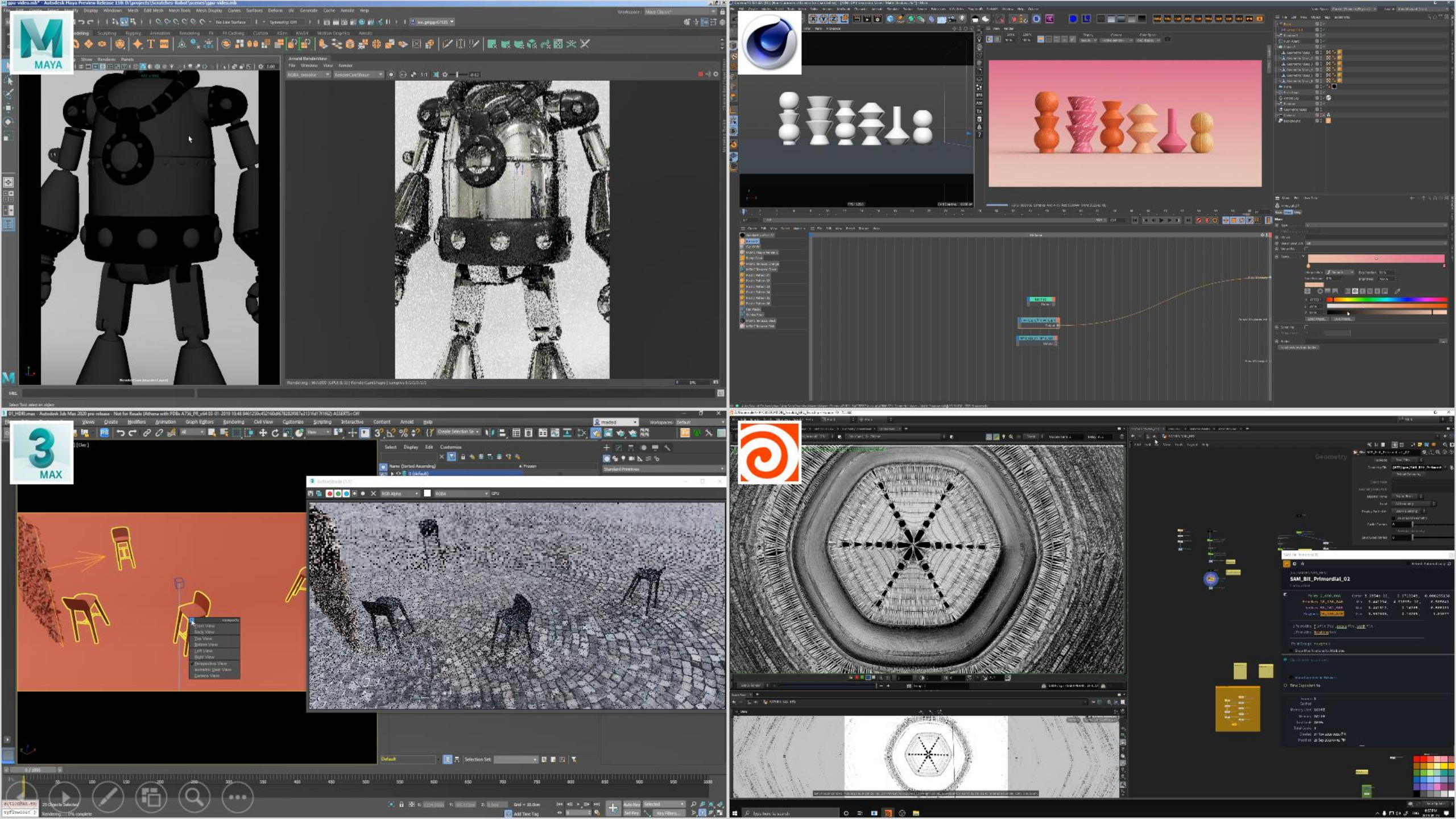
- Scene is shared between CPU/GPU
  - Procedural
  - Instancing
  - Operators
- No overhead if GPU is not
- Live device switching

Arnold GPU

FLEXIBILITY

- Switch seamlessly between CPU and GPU rendering
- Choose the type of rendering best suited to your needs

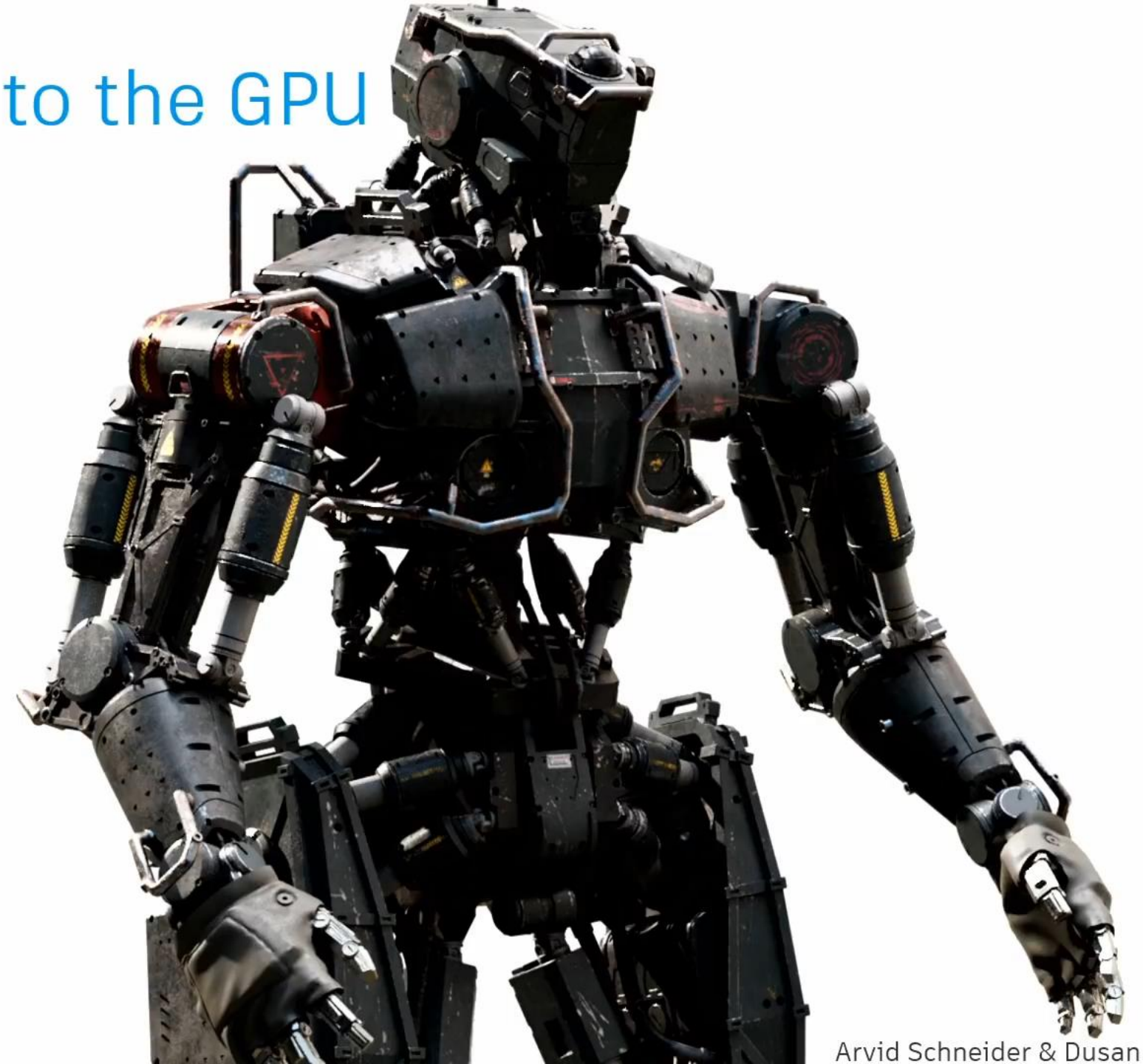



MAYA
3
MAX

Bringing Arnold to the GPU

Goals

- A single renderer
- Feature compatibility
- Pixel compatibility
- API compatibility



Arvid Schneider & Dusan Kovic

谢 谢

AUTODESK ying.shen@autodesk.com