Sharing Physically Based Materials Between Renderers with MDL

Jan Jordan Software Product Manager MDL
Lutz Kettner Director Advanced Rendering and Materials

August 13, SIGGRAPH 2018

Agenda

Introduction to NVIDIA Material Definition Language MDL

Matching the appearance of a single material within different rendering techniques

MDL ecosystem

Become part of the ecosystem

Introduction

NVIDIA® MDL

The **NVIDIA Material Definition Language (MDL)**

is technology developed by NVIDIA

to define **physically-based** materials

for physically-based rendering solutions.


B TZ 2012
D
Iray Photoreal

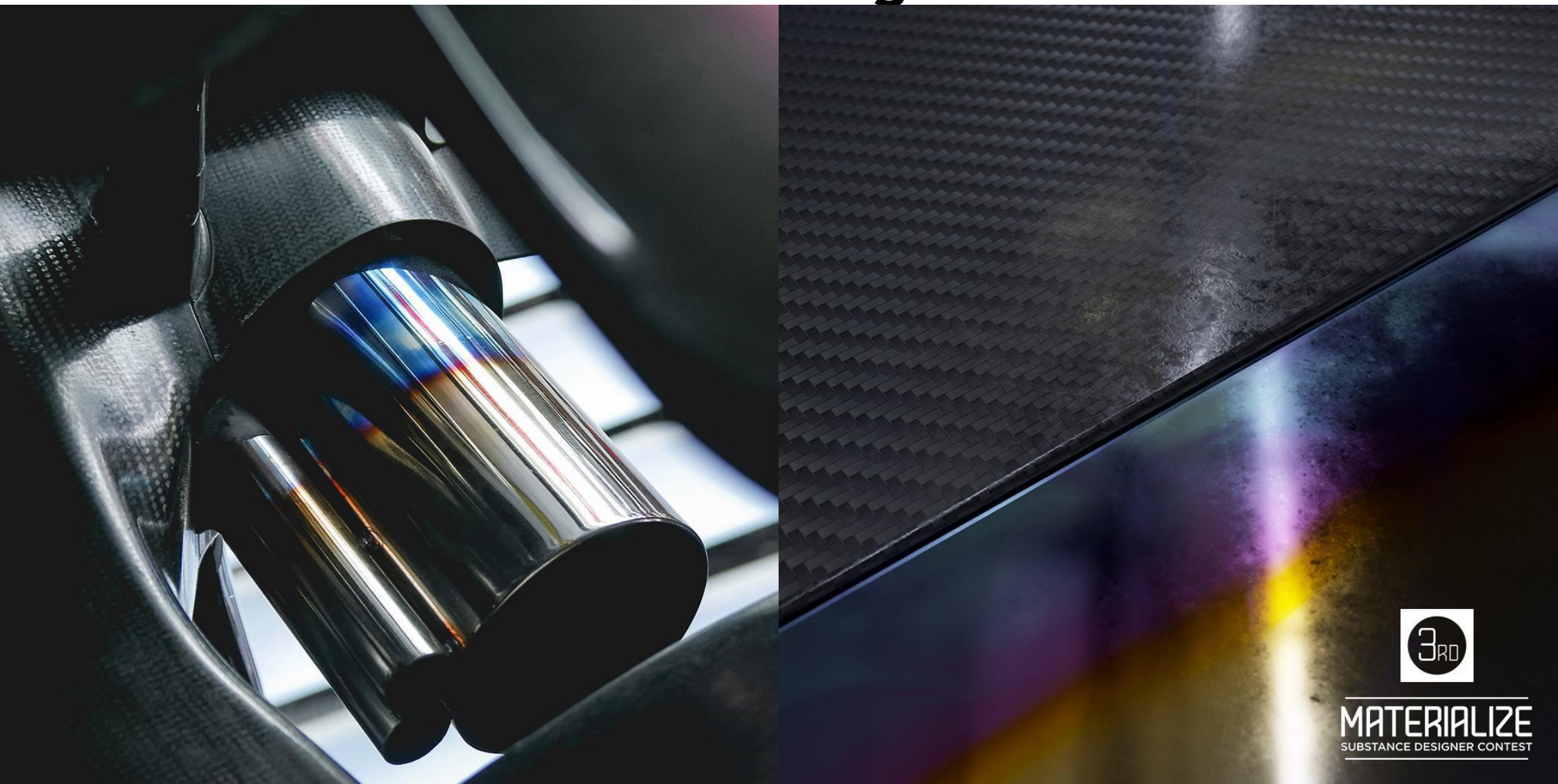
courtesy Harley Davidson

Photographer: Simon Puschmann/Rendering and Retouching: [zerone]


Iray Photoreal

MDL in Substance Designer

MDL in Substance Designer



Matching the appearance of a single material within different rendering techniques

One Scene for Different Renderers

Realtime Rasterizer

Interactive Raytracer

Pathtracer

Share scene and MDL materials for a **consistent look**

Switching renderers with no scene modifications

Traditional Shading Language Parts

Texturing

- Texture lookups
- Procedurals
- Uv-transforms
- Projectors
- Noise functions
- Math functions

Material Definition

- Glossy reflection
- Transparency
- Translucency

Material Implementation

- Light loops / trace N rays
- OIT / ray-continuation
- Ray marching

NVIDIA MDL

Renderer

Procedural Programming Language

- Texture lookups
- Procedurals
- Uv-transforms
- Projectors
- Noise functions
- Math functions

Declarative Material Definition

- Glossy reflection
- Transparency
- Translucency

Rasterizer

Light loops / OIT

Raytracer

Trace N rays

Pathtracer

Ray-marching

NVIDIA MDL

Procedural Programming Language → **Declarative Material Definition**

Renderer

Rasterizer

Light loops / OIT

Raytracer

Trace N rays

Pathtracer

Ray-marching

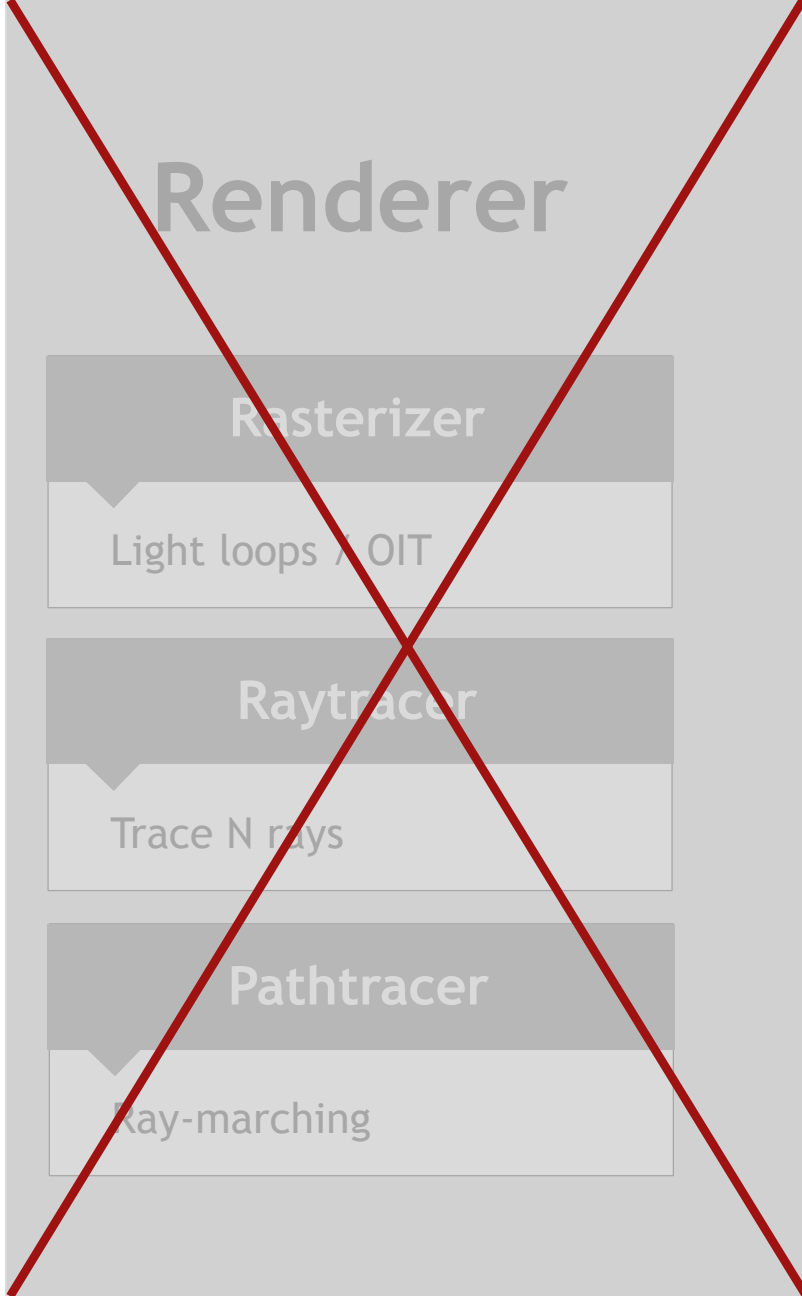
NVIDIA MDL

Procedural Programming Language → **Declarative Material Definition**

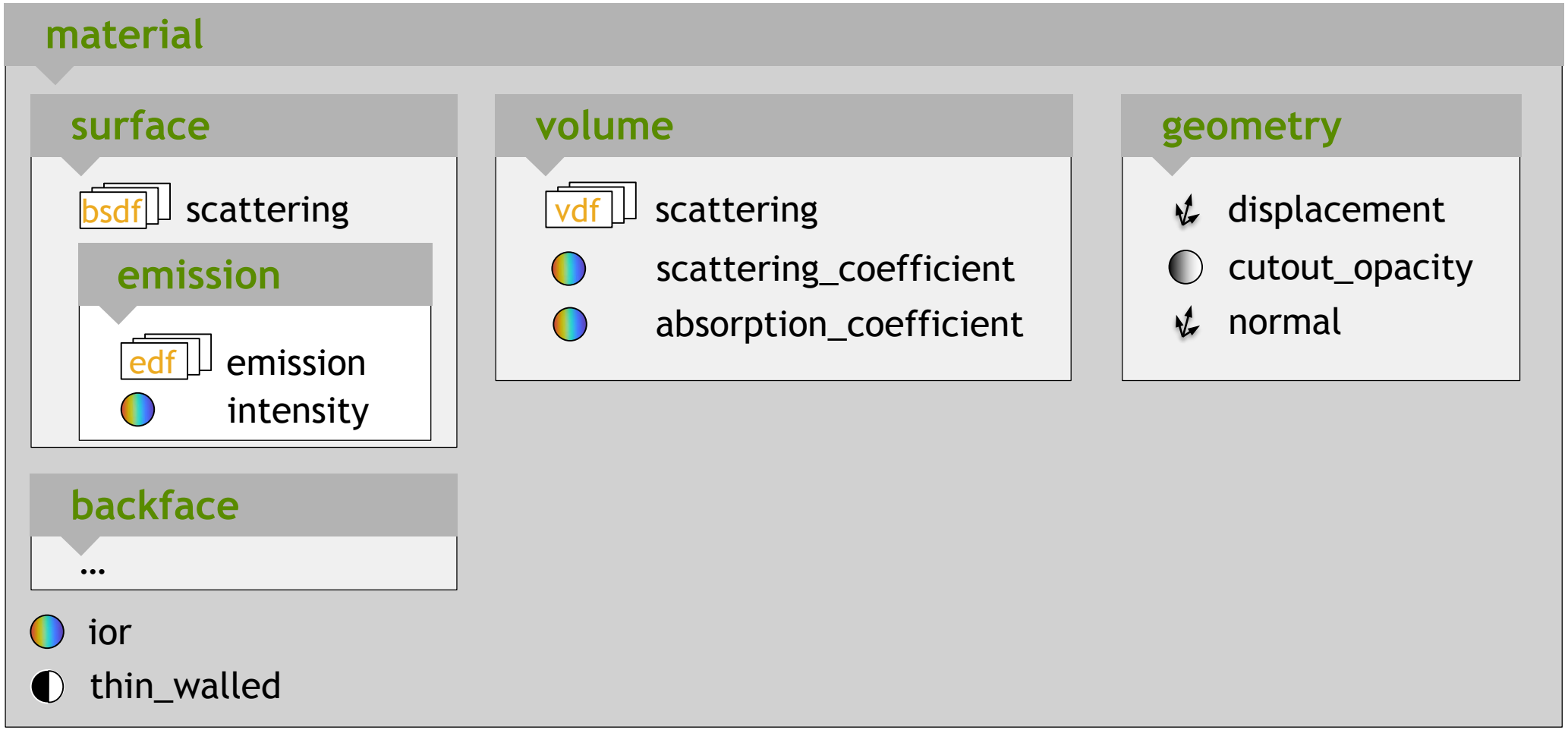
MDL is not a Shading Language

MDL defines what to compute, **not** how to compute it

- no programmable shading
- no light loops or access to illumination
- no trace call
- no sampling
- no camera dependence



MDL Material Model



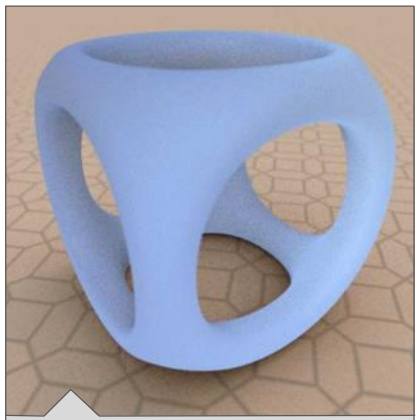
MDL Elemental Distribution Functions

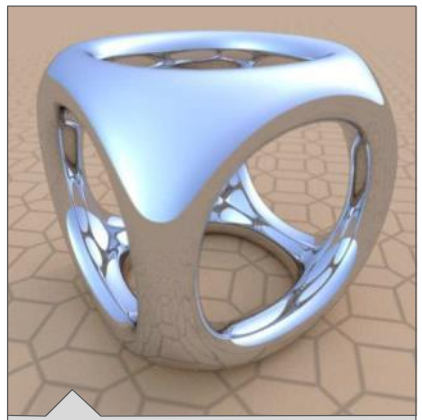
Bidirectional
Scattering
Distribution
Functions

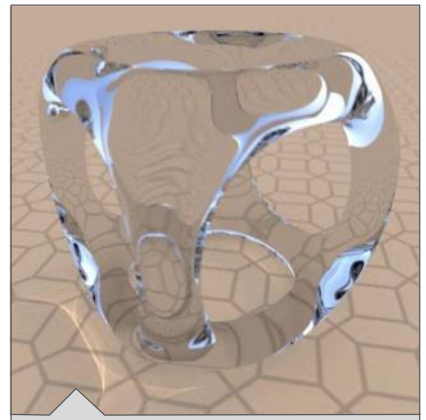

Diffuse Reflection


Diffuse Transmission


Glossy (various)


Backscatter Glossy


Specular Reflection

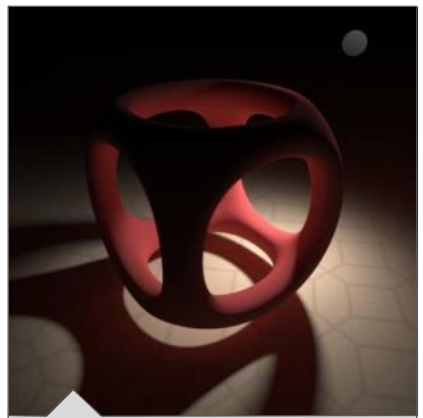

Spec. Refl.+Transm.


Measured BSDF

MDL Elemental Distribution Functions

Emissive
Distribution
Functions

Diffuse



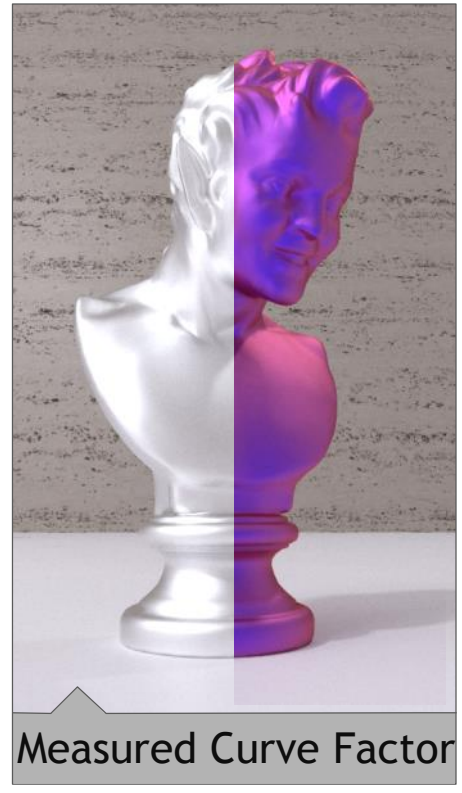
Spot

IES Profile

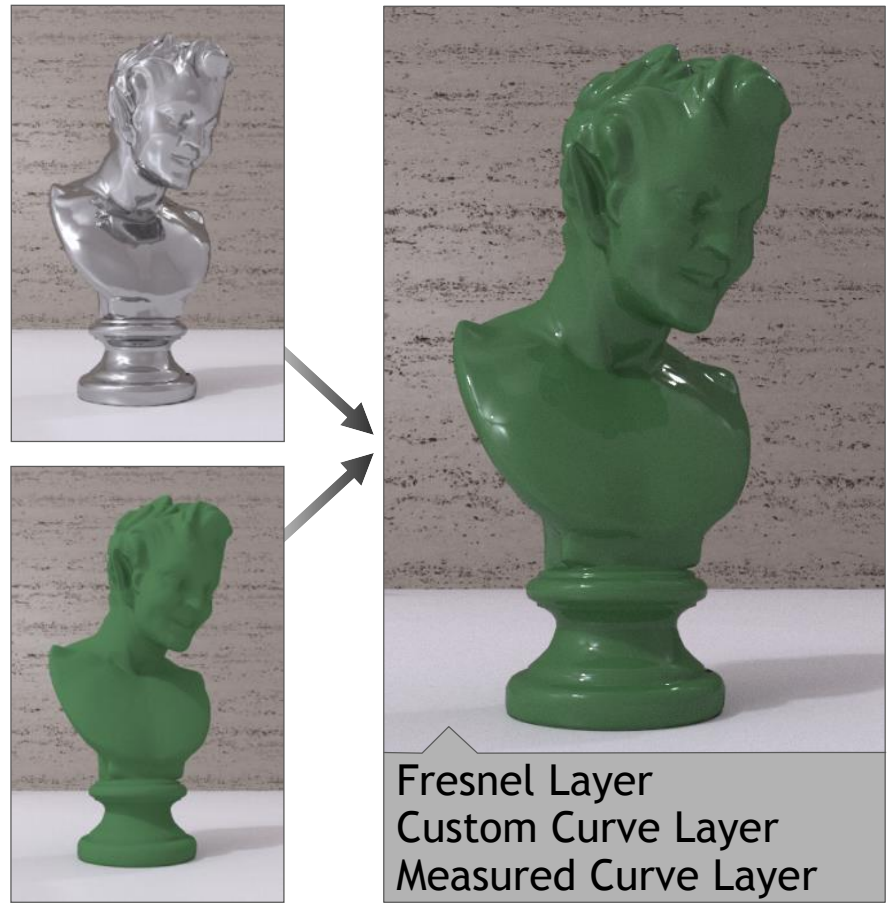
Volume
Distribution
Functions

Henyey-Greenstein

MDL Distribution Function Modifiers



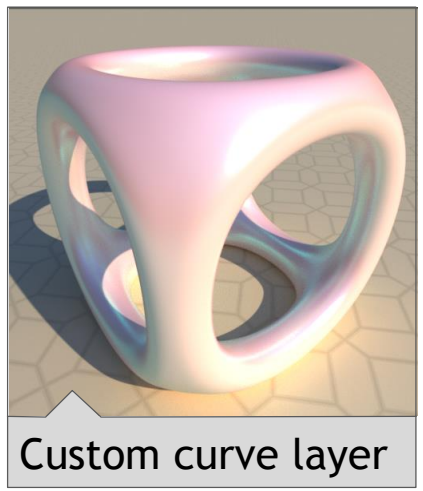
MDL Distribution Functions Combiners



MDL 1.4: New BSDF

Modifier:
Complex ior factor

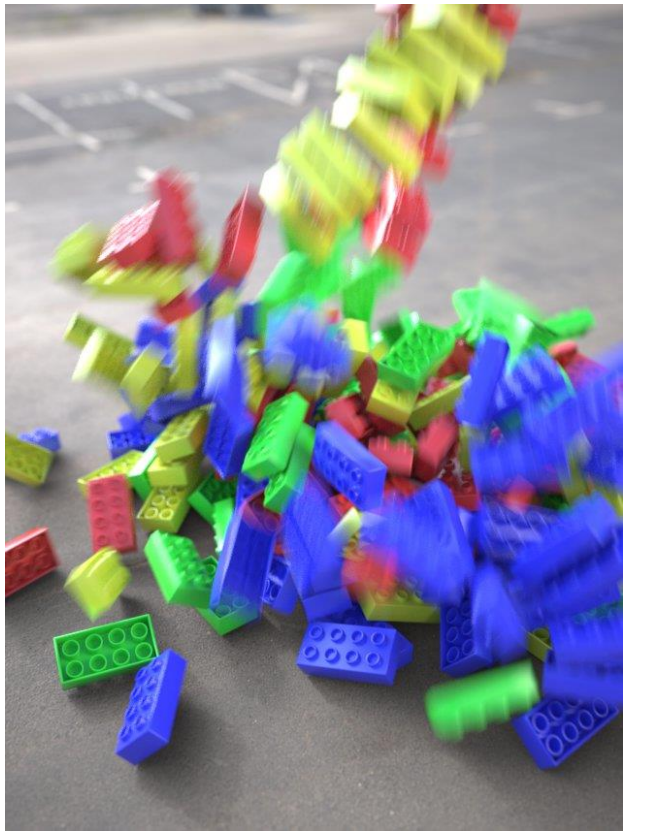
Combiners:
All weights can be color now



Defining a Material Using MDL

Create complex materials by layering

```
material plastic(
        color diffuse_color = color(.15,0.4,0.0),
        float roughness = 0.05
) = material(
        surface: material_surface(
                scattering: df::fresnel_layer (
                        ior: color(1.5),
                        layer: df::simple_glossy_bsdf (
                                roughness_u: glossy_roughness
                        ),
                        base: df::diffuse_reflection_bsdf (
                                tint: diffuse_color )
                )
        )
);
```



Defining a Function Using MDL

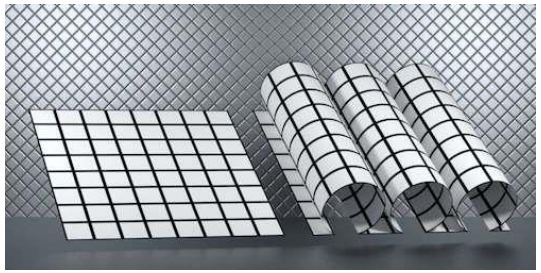
Functions allow control flow like loops, switches, conditionals

```
float summed_perlin_noise (
        float3 point,
        int level_count=4,
        float level_scale=0.5,
        float point_scale=2.0,
        bool turbulence=false)
{
        float scale = 0.5, noise_sum = 0.0;
        float3 level_point = point;
        for (int i = 0; i < level_count; i++)
        {
                float noise_value = perlin_noise(level_point);
                if (turbulence)
                        noise_value = math::abs(noise_value);
                else noise_value = 0.5 + 0.5 * noise_value;
                noise_sum += noise_value * scale;
                scale *= level_scale;
                level_point *= point_scale;
        }
        return noise_sum;
}
```

MDL Handbook

MDL Handbook

www.mdlhandbook.com



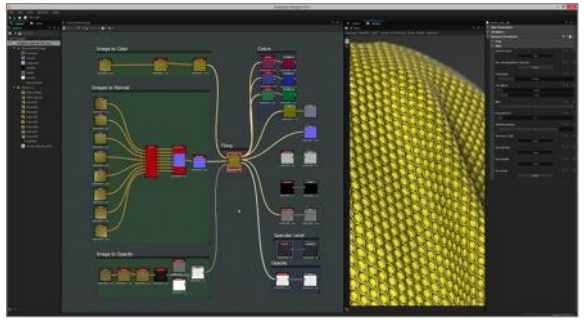
Added displacement since 2017

Cloth example

4 anisotropic glossy highlights + translucency

Additional MDL Benefits

Measured Materials

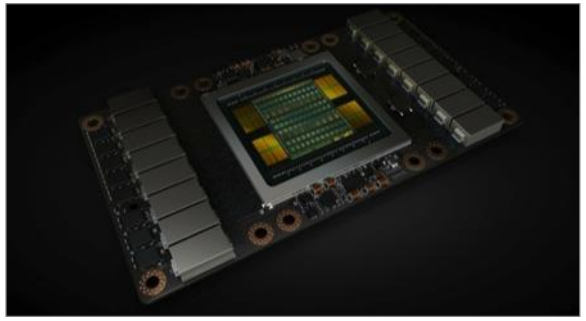


Spatially Varying BRDF

AxF from X-Rite

Measure Isotropic BSDF

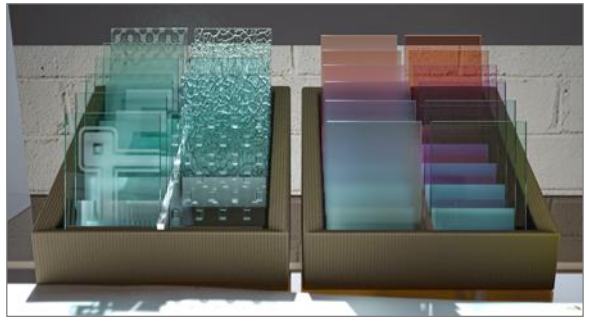
Designed for Parallelism



Little data dependencies

Side-effect free functions

Material Catalogs

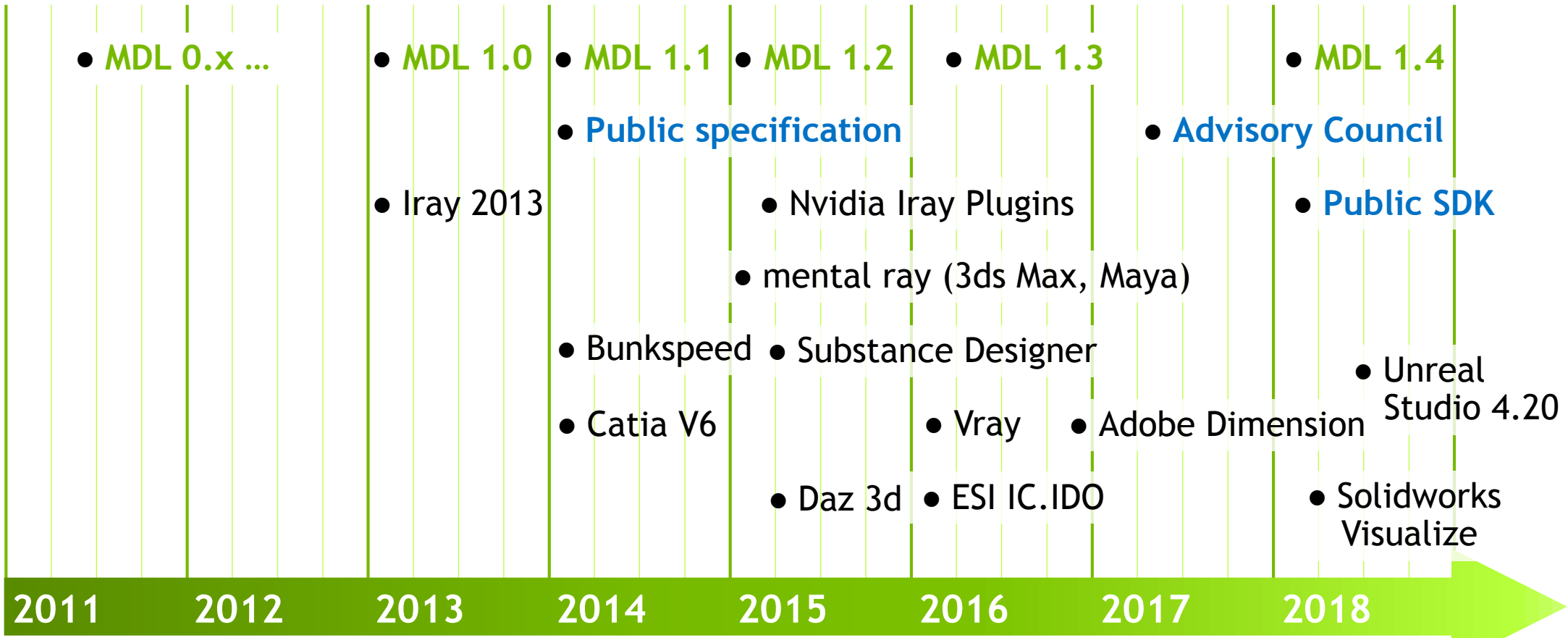


Modules and packages

Archives

MDL ecosystem

MDL – Past, Present and Future



MDL Advisory Council

Companies sharing our vision of MDL

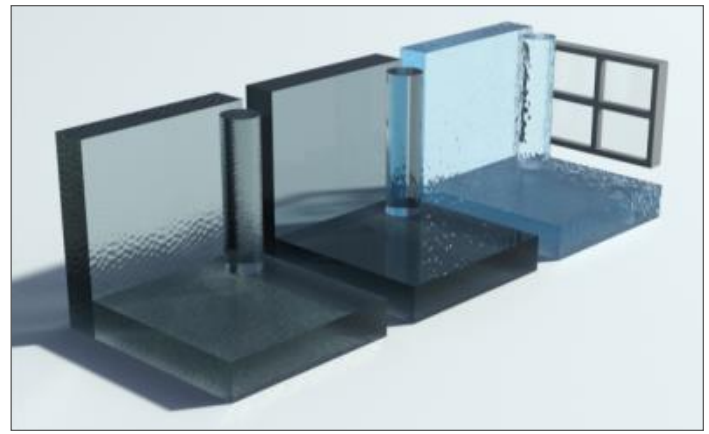
Joint direction of MDL and the MDL eco system

Include expertise other companies have gained in the field and with MDL

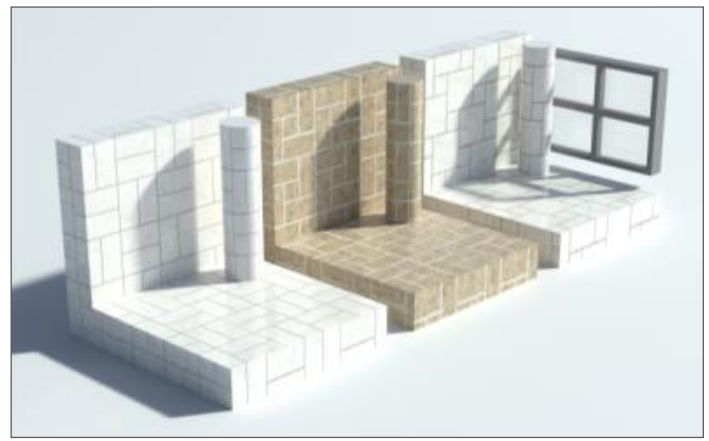
NVIDIA vMaterials 1.5

~1600 MDL materials verified for accuracy - FREE TO USE







Become part of the ecosystem

Become Part of the Ecosystem

Integrate MDL enabled renderer

MDL is included

Write your own compiler

Based on the freely available MDL Specification

Use the MDL SDK

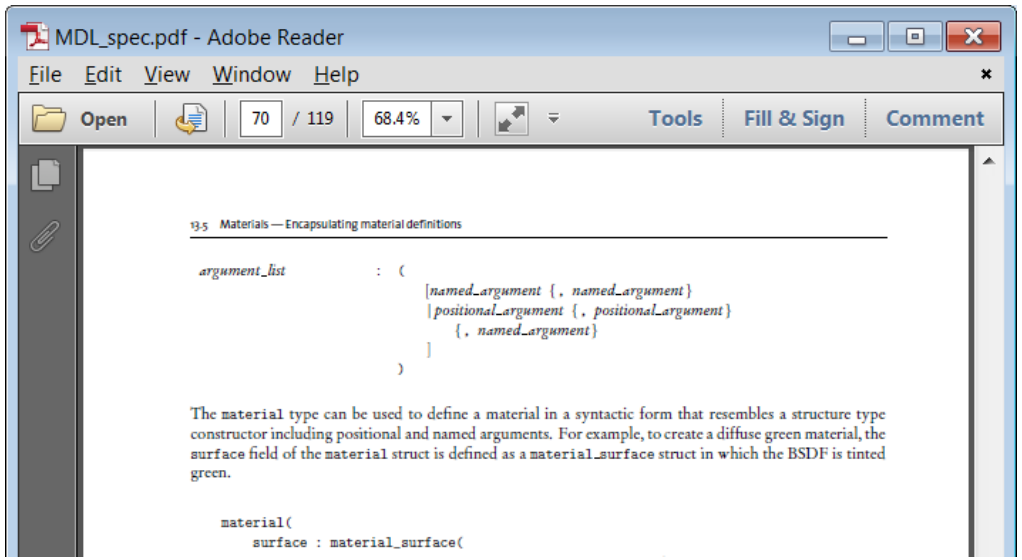
Published under the NVIDIA Designworks License and …

Write Your Own Compiler

MDL Specification

Language specification document
Free to use

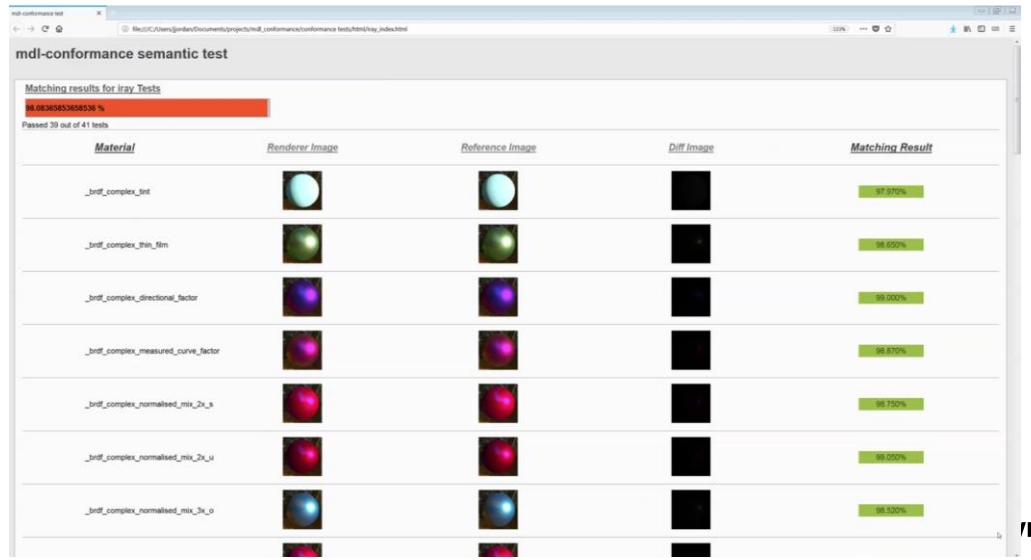
http://www.nvidia.com/mdl/



MDL conformance test suite

Syntactic conformance tests
Semantic conformance tests

Available on request



MDL SDK 2018.1

Features

MDL 1.4

DB for MDL definitions

DAG view on materials
several compilation modes

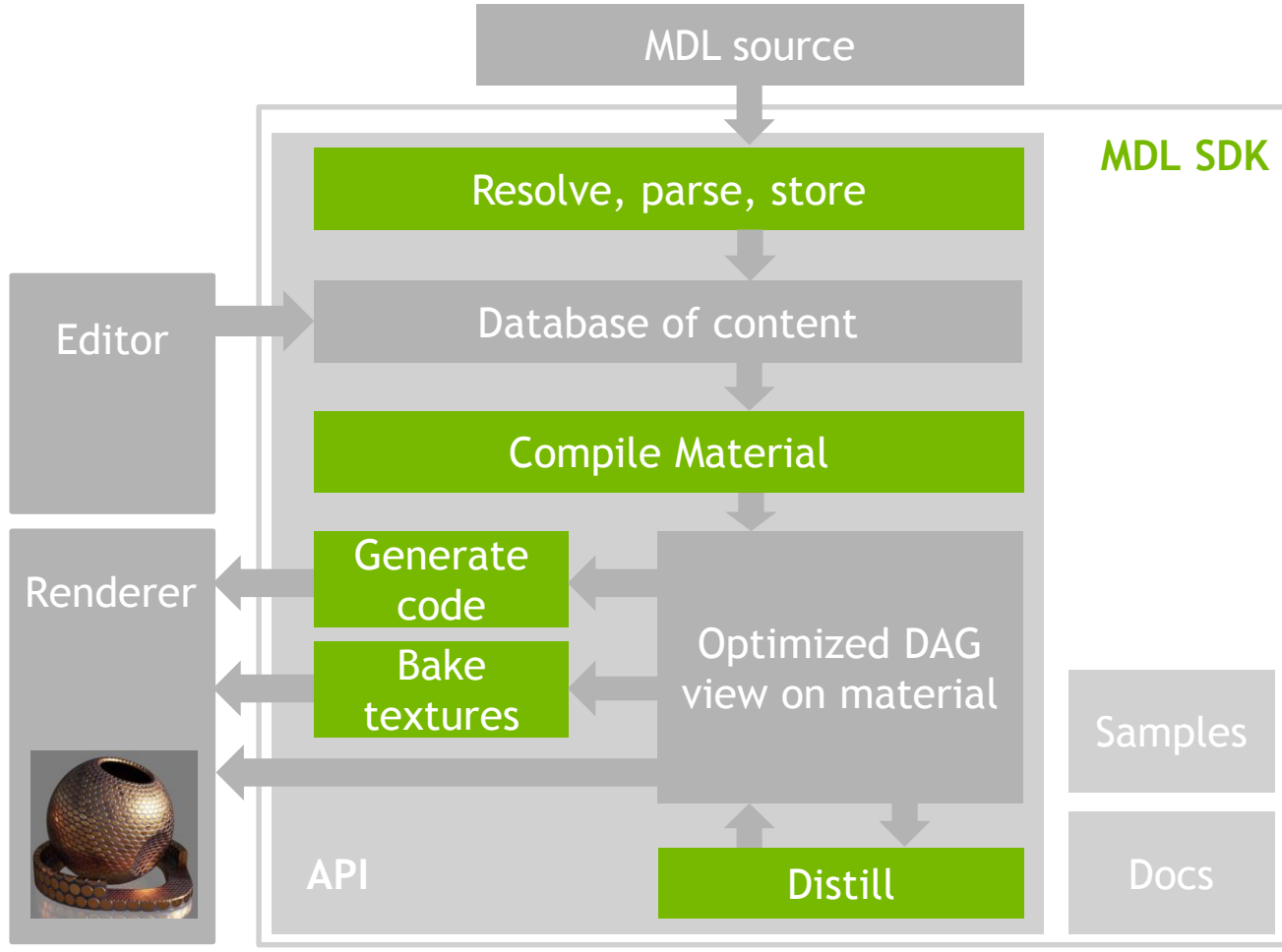
MDL editing

Code generators
PTX, LLVM IR, x86, GLSL (fcts. only)

Distiller and texture baker

Samples

Documentation and tutorials



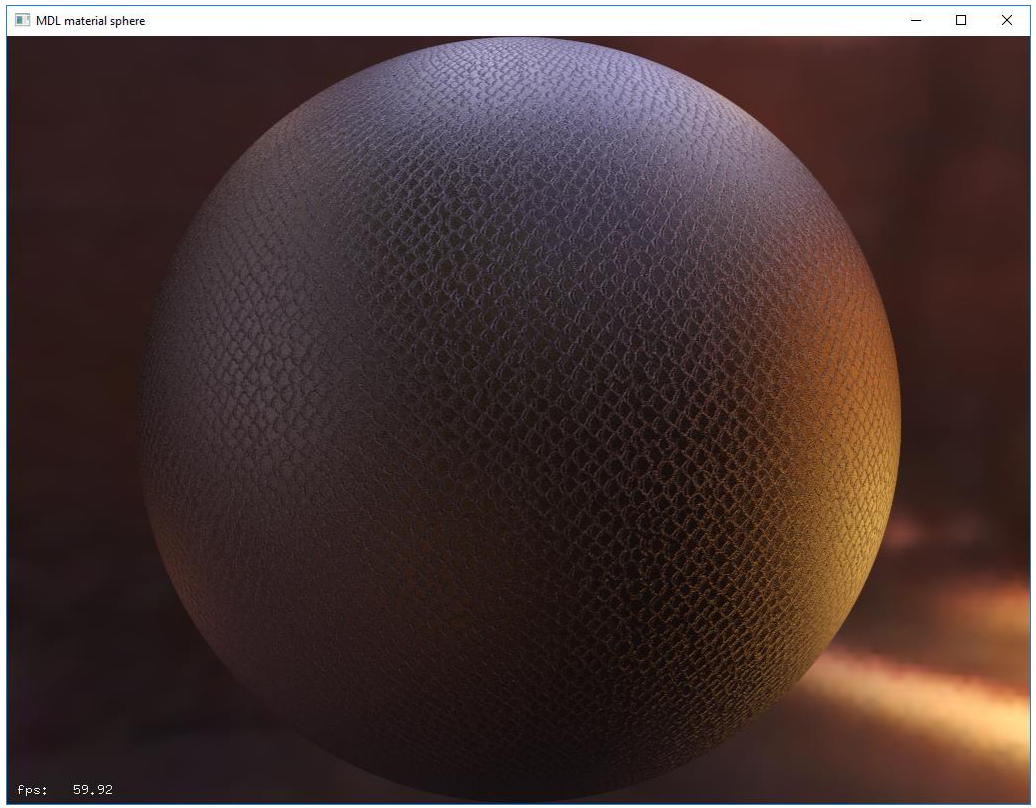
MDL SDK and OptiX

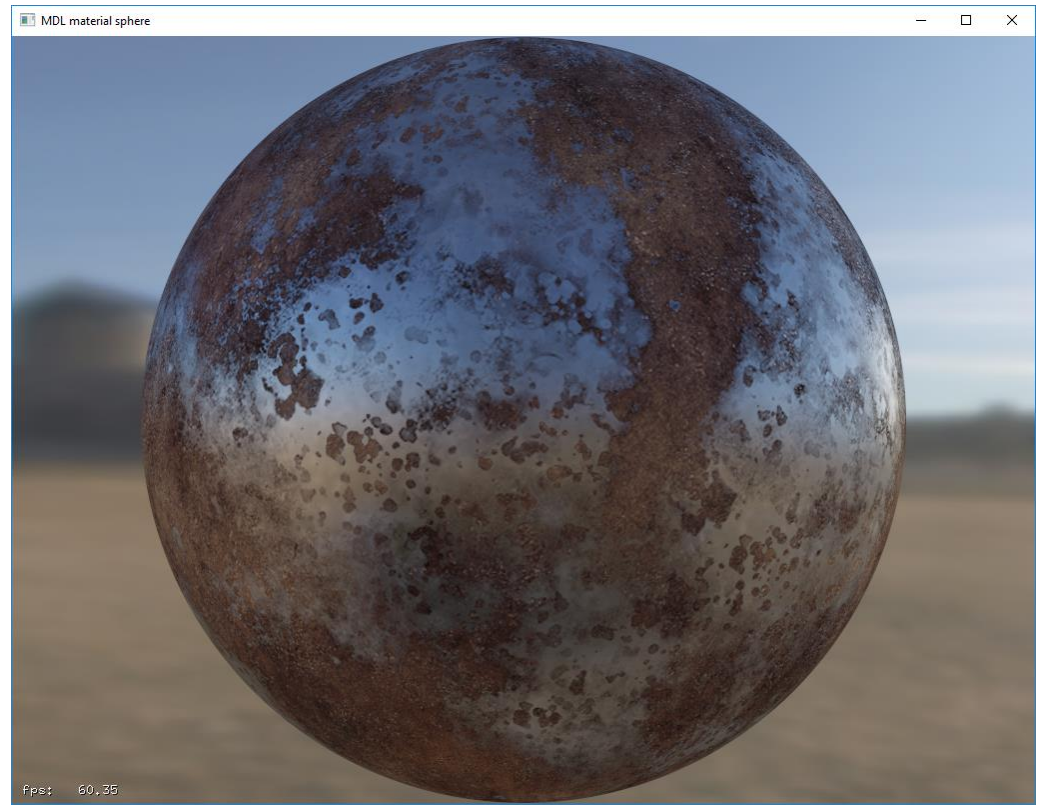
We will give you all you need…

- MDL SDK PTX backend produce PTX code suitable to be called by OptiX available since MDL SDK 2017.1

Full Material Compilation

OptiX and CUDA rendering samples with MDL SDK 2018.0 and OptiX 5.1





MDL Distilling

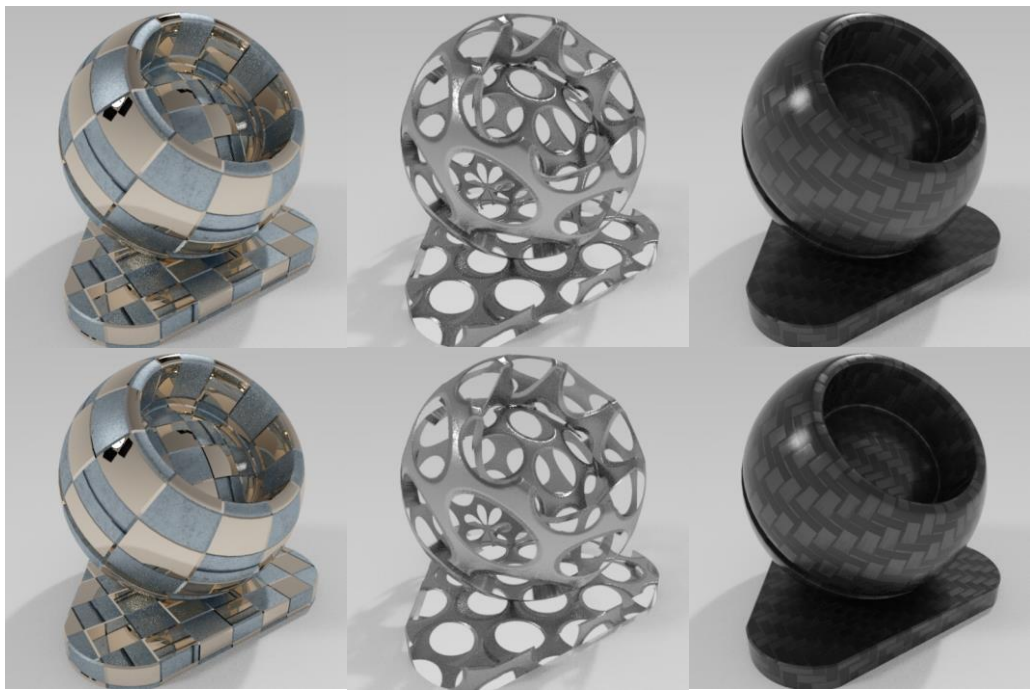
Released as part of Iray/MDL SDK

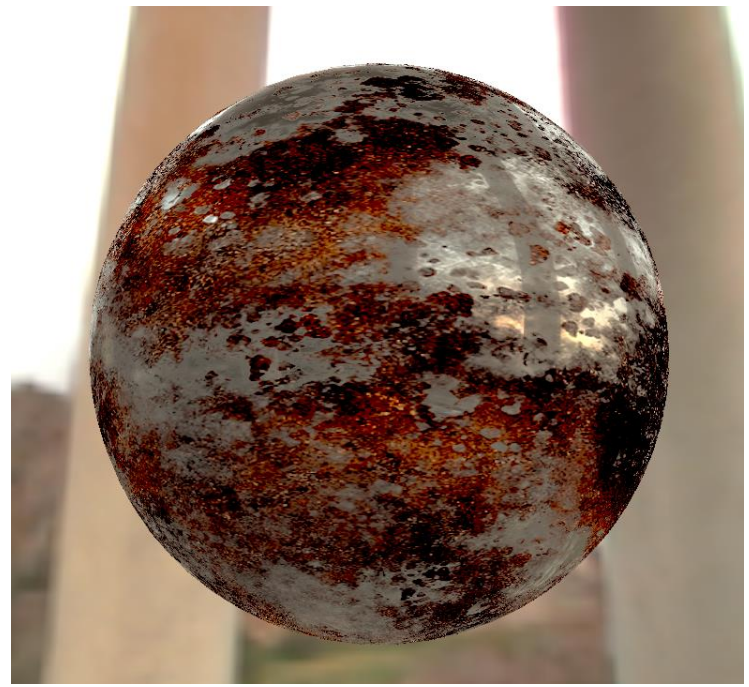
Example: UE4 target with clearcoat and transparency through alpha

GLSL rendering sample using Distilling and baking comes with MDL SDK 2018.1

MDL

UE4





May the Source Be with You

NVIDIA Open Sources the MDL SDK

Today https://github.com/NVIDIA/MDL-SDK

BSD 3-clause license

Full MDL SDK

- 48 modules, 570 files, 310 KLOC
- Excluding
  - MDL Distilling and texture baking
  - GLSL compiler back-end
- Added MDL Core API
- Includes MDL Core Definitions and more

Feature image courtesy of Adobe, created by art director Vladimir Petkovic.

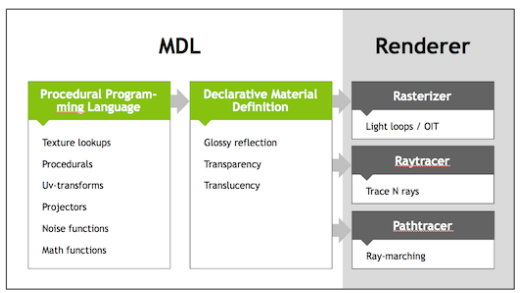
MDL Core API

A Lower-level Compiler API in the MDL SDK

MDL SDK API	MDL Core API
Higher-level API for easy integration	API close to the compiler
Reference counted interfaces	Objects managed in arenas
Mutable objects	Immutable objects
In-memory store	Stateless compiler
Texture and resource importer	Callbacks

MDL Takeaways

What is MDL



Declarative Material Definition

Procedural Programming Language

MDL Ecosystem

NVIDIA vMaterials

MDL Advisory Council

Starting Material

Open Source release

MDL Specification

MDL Handbook

MDL SDK

MDL Backend Examples

Conformance Test Suite

Further Information on MDL

www.nvidia.com/mdl
raytracing-docs.nvidia.com/mdl/index.html

MDL Birds of a Feather
Wednesday 12:30 – 2:00 PM
East Building, Room 13

Documents

NVIDIA Material Definition Language ▫ Technical Introduction
Material Definition Language ▫ Handbook
NVIDIA Material Definition Language ▫ Language Specification