Sharing Physically Based Materials Between Renderers with MDL

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

July 29, SIGGRAPH 2019
Introduction to NVIDIA Material Definition Language MDL

Matching the appearance of a single material within different rendering techniques

Defining physically-based materials

MDL ecosystem

Become part of the ecosystem
Introduction
The NVIDIA Material Definition Language (MDL) is technology developed by NVIDIA to define physically-based materials for physically-based rendering solutions.
Matching the Appearance of a Single Material Within Different Rendering Techniques
One Scene for Different Renderers

Share scene and MDL materials for a consistent look

Switching renderers with no scene modifications
## Traditional Shading Language Parts

<table>
<thead>
<tr>
<th>Texturing</th>
<th>Material Definition</th>
<th>Material Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Texture lookups</td>
<td>Glossy reflection</td>
<td>Light loops / trace N rays</td>
</tr>
<tr>
<td>Procedurals</td>
<td>Transparency</td>
<td>OIT / ray-continuation</td>
</tr>
<tr>
<td>Uv-transforms</td>
<td>Translucency</td>
<td>Ray marching</td>
</tr>
<tr>
<td>Projectors</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Noise functions</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Math functions</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Procedural Programming Language
- Texture lookups
- Procedurals
- Uv-transforms
- Projectors
- Noise functions
- Math functions

Declarative Material Definition
- Glossy reflection
- Transparency
- Translucency

Renderer
- Rasterizer
  - Light loops / OIT
- Raytracer
  - Trace N rays
- Pathtracer
  - Ray-marching
Procedural Programming Language → Declarative Material Definition

- Rasterizer
  - Light loops / OIT
- Raytracer
  - Trace N rays
- Pathtracer
  - Ray-marching

Renderer
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

**Material**

<table>
<thead>
<tr>
<th>Surface</th>
<th>Volume</th>
<th>Geometry</th>
</tr>
</thead>
</table>

**Backface**

...
MDL Material Model

- **material**
  - surface
    - bsdf
      - scattering
  - backface
    - ...
- **volume**
- **geometry**
MDL Material Model

- **material**
  - **surface**
    - bsdf: scattering
    - emission
      - edf: emission
      - intensity
  - backface...
  - volume
  - geometry
MDL Material Model

**material**

**surface**
- bsdf
- scattering
- emission
- edf
- intensity

**volume**
- vdf
- scattering
- scattering_coefficient
- absorption_coefficient

**geometry**
...
MDL Material Model

### material

#### surface
- **bsdf** scattering
- **edf** emission
  - intensity

#### volume
- **vdf** scattering
  - scattering_coefficient
  - absorption_coefficient

#### geometry
- displacement
- cutout_opacity
- normal

---

...
# MDL Material Model

## Material

### Surface
- **bsdf** scattering
- **emission**
  - **edf** emission
  - intensity

### Volume
- **vdf** scattering
  - scattering_coefficient
  - absorption_coefficient

### Geometry
- displacement
- cutout_opacity
- normal

## Backface
- ior
- thin_walled
MDL Elemental Distribution Functions

Bidirectional Scattering Distribution Functions

Diffuse Reflection

Diffuse Transmission

Glossy (various)

Backscatter Glossy

Specular Reflection


Measured BSDF
MDL  Elemental Distribution Functions

Emissive Distribution Functions

Volume Distribution Functions

Henyey-Greenstein
MDL Distribution Function Modifiers

- Tint
- Thin Film
- Directional Factor
- Measured Curve Factor
MDL Distribution Functions Combiners

Normalized Mix
Clamped Mix
Weighted Layer

Fresnel Layer
Custom Curve Layer
Measured Curve Layer
MDL Layered Material Example
Defining Physically-based Materials With Source Code
Defining a Material Using MDL
MDL is a ‘C’ like language. The material viewed as a struct

```c
struct material {
    bool thin_walled;
    material_surface surface;
    material_surface backface;
    color ior;
    material_volume volume;
    material_geometry geometry;
};
```
Defining a Material Using MDL

MDL is a ‘C’ like language. The material and its components viewed as a struct

```c
struct material {
    bool thin_walled;
    material_surface surface;
    material_surface backface;
    color ior;
    material_volume volume;
    material_geometry geometry;
};

struct material_surface {
    bsdf scattering;
    material_emission emission;
};
```
Defining a Material Using MDL

MDL is a ‘C’ like language. The material and its components viewed as a struct

```c
struct material {
    bool thin_walled = false;
    material_surface surface = material_surface();
    material_surface backface = material_surface();
    color ior = color(1.0);
    material_volume volume = material_volume();
    material_geometry geometry = material_geometry();
};

struct material_surface {
    bsdf scattering = bsdf();
    material_emission emission = material_emission();
};
```
Defining a Material Using MDL

Material struct is already fully defined

```cpp
material();
```
Defining a Material Using MDL

Material struct is already fully defined

material();
Defining a Material Using MDL

Creating new materials

```
material name  ( material-parameters )
  = material  ( material-arguments );
```
Defining a Material Using MDL

```plaintext
material plaster() = material(
    surface: material_surface(
        scattering: df::diffuse_reflection_bsdf()
    )
);
```
Defining a Material Using MDL

New materials can have parameters

```mdl
material plaster (color plaster_color = color(.7))
  = material(
      surface: material_surface (
          scattering: df::diffuse_reflection_bsdf (  
            tint: plaster_color
          )
      )
  );
```
Defining a Material Using MDL

Create complex materials by layering

```plaintext
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  
            )  
        )  
    )
);
```
MDL Procedural Programming Language

C-like language for function definitions

Function results feed into material and function parameters

“Shader graphs” are equivalent to function call graphs
Defining a Function Using MDL

Functions allow control flow like loops, switches, conditionals

```mdl
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;
}
```
Defining a Function Using MDL

Call graph of functions substitute shader graphs

```
material perlin_noise_material()
  = plaster(
    plaster_color: color(
      summed_perlin_noise(
        point: state::texture_coordinate(0)
      )
    )
  )
```

```
Call graph of functions substitute shader graphs
```
MDL 1.5
MDL Encapsulated File Format (MDLE)

One material - fully self contained in one file
Includes textures, previews, etc. in the file
Renaming and copying works

... work just like textures
MDL 1.5
Internationalization (i18n)

Localization of all MDL string annotations

Based on OASIS standard XLIFF 1.2: XML Localisation Interchange File Format
http://docs.oasis-open.org/xliff/xliff-core/xliff-core.html

Package and module XLIFF files in MDL file hierarchy

Example

C:Users%USERNAME%Documentsmdl
    nvidia\vMaterials\fr.xlf  MDL search path
    nvidia\vMaterials\AEC\Glass\Mirror_fr.xlf  French vMaterial package XLIFF file
    nvidia\vMaterials\AEC\Glass\Mirror_fr.xlf  French Mirror module XLIFF file
MDL 1.5

Hair shading

```markdown
struct material {
    ...
    hair_bsdf hair;
};

hair_bsdf chang_hair_bsdf {
    float diffuse_reflection_weight = 0.0;
    color diffuse_reflection_tint = color(1.0);
    float2 roughness_R = float2(0.0);
    float2 roughness_TT = roughness_R;
    float2 roughness_TRT = roughness_TT;
    float cuticle_angle = 0.0;
    color absorption_coefficient = color();
    float ior = 1.55;
};
```
**MDL 1.5**

Microfacet coloring to support flip-flop car paints and more

1D measured curve (MDL >=1.4)

2D measured curve (new in MDL 1.5)
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 0.x**: Iray 2013
- **2011**
  - MDL 0.x
  - Iray 2013

- **MDL 1.0**: Public Specification
- **2012**
  - MDL 1.0
  - Public Specification

- **MDL 1.1**: JIT Compile
- **2013**
  - MDL 1.1
  - JIT Compile
  - Bunkspeed
  - Catia V6

- **MDL 1.2**: vMaterials
- **2014**
  - MDL 1.2
  - vMaterials
  - Schmidt ray (3ds May, Maya)
  - Substance Designer

- **MDL 1.3**: Advisory Council
- **2015**
  - MDL 1.3
  - Advisory Council
  - Holodeck
  - Daz 3d

- **MDL 1.4**: Public SDK
- **2016**
  - MDL 1.4
  - Public SDK
  - Unreal Studio 4.20
  - Adobe Dimension

- **MDL 1.5**: Open Source SDK
- **2017**
  - MDL 1.5
  - Open Source SDK
  - Nvidia Iray Plugins
  - ESI IC.IDO

- **2018**
  - MDL 1.5
  - Open Source SDK
  - Unreal Studio 4.20
  - Nvidia Iray Plugins
  - ESI IC.IDO

- **2019**
  - MDL 1.5
  - Open Source SDK
  - Nvidia Iray Plugins
  - ESI IC.IDO
  - Megascans
  - Vizoo
  - Solidworks Visualize
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
SOLIDWORKS Visualize

MDL import since 10/2018, tweaking + viewport preview coming
Epic Unreal Studio
“Real-time workflows for enterprise”  www.unrealengine.com/studio
MDL support through DATASMITH
Vizoo xTex
MDL export in the next release

“Vizoo is the number one supplier of Soft-and Hardware solutions for the physically accurate digitization of material swatches in the fashion industry.“

www.vizoo3d.com
MEDULR
Online MDL editor and material library

Discover, create and share materials. We're building a global community to create the worlds largest material library.

www.medulr.com
Quixel Megascans
“Incredible scans and tools for creatives.”
quixel.com
POC MDL in Arnold

Rendering Omniverse content in Maya and Omniverse Kit
# MDL Ecosystem

## Libraries/Scan
- NVIDIA VMaterials
- Substance Source
- Adobe Stock
- Vizoo xTex
- Quixel Megascans

## Creation
- Substance Designer
- medulr

## Rendering
- NVIDIA Iray
- Chaosgroup Vray
- UE4
- Omniverse RTX

## Applications inc Renderers
- Siemens NX11
- Solidworks Vis.
- Dassault Catia V6
- Iray DCC Plugins
- Other Iray Prod’s
- Daz 3D Studio
- Adobe Dimensions
- Vray Max/Maya
- Unreal Studio
- NVIDIA Omniverse
Focus on Material Exchange
Freely choose where to author material content

create

Substance Designer

modify

Iray for Rhino

consume

Chaosgroup V-RAY
NVIDIA vMaterials 1.6 - Just released
~1700 MDL materials verified for accuracy - FREE TO USE
Simple, contextual material parameters, example: age
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
RENDERING

Iray SDK
OptiX SDK
MDL SDK
NV Pro Pipeline
vMaterials

VOXELS

GVDB Voxels
VXGI

VIDEO

GPUDirect for Video
Video Codec SDK

MANAGEMENT

GRID SW MGMT SDK
NVAPI/NVWMI

DISPLAY

Multi-Display
Capture SDK
Warp and Blend

https://developer.nvidia.com/designworks
MDL SDK 2019

Features

MDL 1.5
DB for MDL definitions
DAG view on materials
several compilation modes
MDL editing

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

Distiller and texture baker
Code examples
Documentation and tutorials
MDL and RTX

Materials tricky for today's game engines become feasible with RTX

- Anisotropic glossy reflections
- True refractive and volumetric materials
- Measured BRDF
- Proper translucency
- Complex glossy lobe shape and color

MDL materials make RTX shine!
MDL SDK and RTX
The MDL SDK directly generates material code for use in RTX enabled renderer

Microsoft DXR
- HLSL back-end since MDL SDK 2019.0.1
- example path tracer in the SDK with glTF support

NVIDIA OptiX
- PTX back-end since MDL SDK 2018.1
  example program available as part of Optix 5.1 & 6

Integrating MDL with an RTX based renderer is simple!
Automatic Derivatives for Texture Filtering

OptiX sample renderer integration: Derivatives off
Automatic Derivatives for Texture Filtering

OptiX sample renderer integration: Derivatives on
MDL in Realtime Rendering
Three approaches

1. Ubershader
2. Compilation: on-demand shader generation
3. Distillation to fixed material model

All based on MDL SDK
Distillation to Fixed Material Model

**MDL Material**
- Complex BSDF layering
- Complex procedurals

**Fixed Material Model**
- Simple BSDF structure
- One texture per parameter
Distillation to Fixed Material Model

MDL Material

- Complex BSDF layering
- Complex procedurals

Distillation

Fixed Material Model

- Simple BSDF structure
- One texture per parameter
Distillation to Fixed Material Model

**MDL Material**
- Complex BSDF layering
- Complex procedural

**Fixed Material Model**
- Simple BSDF structure
- One texture per parameter
Distillation to Fixed Material Model

**MDL Material**
- Complex BSDF layering
- Complex procedurals

**Fixed Material Model**
- Simple BSDF structure
- One texture per parameter

Approximate render result: Some materials will look quite different.
Distillation to Fixed Material Model

**MDL Material**
- Complex BSDF layering
- Complex procedurals

**Fixed Material Model**
- Simple BSDF structure
- One texture per parameter

Fast projection of material instances: Realtime editing

Approximate render result: Some materials will look quite different
Distillation to Fixed Material Model

**MDL Material**
- Complex BSDF layering
- Complex procedurals

**Fixed Material Model**
- Simple BSDF structure
- One texture per parameter

**Fast projection of material instances: Realtime editing**

**Flexible framework to target different fixed models not a fixed MDL subset (no “MDL lite”)**

**Approximate render result:**
Some materials will look quite different
Distillation to Fixed Material Model

Results on vMaterials

diffuse-only

Fresnel( glossy, diffuse)

original
MDL Distilling

Released as part of Iray/MDL SDK

Multiple distilling targets (diffuse only, diffuse_glossy, UE4, new: transmissive PBR)

Original:
Iray MDL

Projection:
Dassault Stellar with Enterprise PBR
First Anniversary of the MDL SDK open source release

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

6 releases shipped since SIGGRAPH 2018

HLSL backend and example just added

BSD 3-clause license

Full MDL SDK
- Added MDL Core API
- Excluding MDL Distilling and texture baking
- GLSL compiler back-end
- Includes MDL Core Definitions and more
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 rendering Examples
- Conformance Test Suite
Further Information on MDL


Documents

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

GTC On-Demand

on-demand-gtc.gputechconf.com

MDL@SIGGRAPH

Talk: Sharing Physically Based Materials Between Renderers with MDL
Mon 10 AM  Room 501AB

BoF: Material Definition Language (MDL): Application Independent PBR Materials
Wed 12:30PM  Room 503

Talk: NVIDIA Omniverse: An Open, USD Based Collaboration Platform for Constructing and Simulating Virtual Worlds
Tue 11:00AM  Room 503

Omniverse Demos at the NVIDIA booth
Tue - Thu  Exhibition

GTC On-Demand  Room 503

Wed 12:30PM

Exhibition

Tue - Thu