MDL + SUBSTANCE PAINTER

Manuel Kraemer - GDC 2016
PHYSICALLY BASED LOOK DEVELOPMENT
VISUAL PERCEPTION

Don’t eat that!

- Static cues:
  - color / texture

- Dynamic cues:
  - roughness / glossyness

- Optic Flow
HUMAN VISUAL SYSTEM

“CG looks like plastic!”

- Material Perception:
  - Identification: GREAT!
  - Spatial relationships: poor
  - Inferring properties: poor
“EMPIRICAL” RENDERING
And why we moved on...

- Few things look “just right”
- Inconsistent lighting
- Very labor intensive
PHYSICALLY BASED RENDERING

The “correct” solution

- A big hump for users:
  - Physical quantities (radiance, flux, IES...)
  - Constrained parameters
  - Must be fully integrated: cameras + lights + BRDF
PHYSICALLY BASED MATERIALS

Dielectrics and conductors

- New B\(<x>\)DF:
  - Energy conservative
  - Fresnel
  - All light scatter paths
  - Fewer parameters
PHYSICALLY BASED MATERIALS

The art of constraints

• Reduced search space
  • Reduced human bias

• “Plausible” glossy materials

• Increased image fidelity

• Increased productivity

Mizuchi | ©Silicon Studio Corp

gameworks.nvidia.com
MICRO-STRUCTURES

Human Skin

- Micro-facets:
  - Off-specular peak
  - Forward scatter (skin)
  - Retro-reflective (cloth)
- Complex BRDF
- How to guess parameters?
DATA DRIVEN MATERIALS

The painter’s palette

- Quantitative measurements
- Building-blocks catalogs
- No guess-work
- Predictive rendering
  - or...
- Combine & tweak for style

gameworks.nvidia.com
PBR IN PRODUCTION
The need for standardization...

- We want:
  - Image Quality
  - Portability
  - Simplicity
- We have:
  - Proprietary GPU code
  - Different Maths / Physics

[gameworks.nvidia.com](gameworks.nvidia.com)
PROGRAMMABLE GPU SHADING
The shading “pipeline”

Texturing
- Procedurals
- uv-transforms
- Projectors
- Noise functions
- Math functions
- Render state
- ...

Material Definition
- Reflections
- Transparency
- Translucency
- ...

Material Implementation
- Light loops
- OIT /ray-continuation
- Ray-marching
- ...

gameworks.nvidia.com
DIFFERENT SKILL-SETS

The skill gap

**Programmer**
- Define how parameters & functions are interpreted by the renderer

**Material Specialist**
- Define materials construction
- Define which parameters should be exposed to users

**End-User**
- Tweak & customize the look
- Make variations
- Define the assignments

---

**Implementation (Renderer)**

<table>
<thead>
<tr>
<th>Programming</th>
<th>Materials</th>
<th>Design</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="gameworks.nvidia.com" alt="Image" /></td>
<td><img src="gameworks.nvidia.com" alt="Image" /></td>
<td><img src="gameworks.nvidia.com" alt="Image" /></td>
</tr>
</tbody>
</table>

**Definition (ISV)**

**Customization (End Users)**
MATERIAL DESCRIPTION LANGUAGE
Abstracting the API

Procedural Language → Declarative Material Model → Renderer

- Rasterizer:
  - Light Loops
  - OIT
- Ray-tracer:
  - Trace rays
- Path-tracer
  - Scatter integrators
MATERIAL DESCRIPTION LANGUAGE

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 IN YOUR GAME ENGINE?

- Powerful
- Intuitive
- Flexible
- Open
- Portable

gameworks.nvidia.com
MDL IN IRAY
Beyond the engine

Iray Photoreal Path Tracer
Iray Interactive Ray Tracer, Direct Illumination
Iray Realtime OpenGL Rasterizer

gameworks.nvidia.com
SUBSTANCE / MDL DEMO
THE END

Thanks to:

Jay Axe
Phillip Miller
Nikolaus Binder
Alexander Keller
John Spitzer