CommonSurfaceShader Revisited: Improvements and Experiences
Karsten Schwenk, Yvonne Jung, Gerrit Voß, Timo Sturm, Johannes Behr
Outline

- Introduction
  - Why CommonSurfaceShader?
  - The original CommonSurfaceShader

- Improvements
  - Energy conservation
  - Anisotropy
  - Layered materials

- Open Problems

- Conclusion
Introduction – Why CommonSurfaceShader

- Material, Texture, etc.
  - Portable, easy to specify
  - Not powerful enough to model modern features
  - … at least not intuitively

- Shaders
  - Very powerful
  - Not very portable

- CommonSurfaceShader
  - Solid default material
  - Portable and expressive
Introduction – Original CommonSurfaceShader

“Declarative Surface Shader“

- Shader in X3D‘s node hierarchy
- But no explicit code, just data fields with fixed semantics

```xml
<Appearance>
  <CommonSurfaceShader
diffuseFactor='0.2 0.2 0.2' specularFactor='1 0 1' shininessFactor='0.8'>
  <SurfaceShaderTexture containerField='diffuseTexture'>
    <TextureTransform2D scale='10 10'/>
    <ImageTexture url='bongo_karl.jpg'/>
  </SurfaceShaderTexture>
  </CommonSurfaceShader>
</Appearance>
```
Introduction – Original CommonSurfaceShader

- Three components
  - Core (diffuse, glossy, emission, ambient, …)
  - Bump (bump/normal mapping)
  - Specular (perfect specular reflection/refraction)
Introduction – Original CommonSurfaceShader

- Only Core
Introduction – Original CommonSurfaceShader

- Core + Bump
Introduction – Original CommonSurfaceShader

- Core + Bump + Specular
Outline

- Introduction
  - Why CommonSurfaceShader?
  - The original CommonSurfaceShader

- Improvements
  - Energy conservation
  - Anisotropy
  - Layered materials

- Open Problems

- Conclusion
Improvements – Energy Conservation

Problem:
- BRDF of original core component not energy conserving
- Closer to Material node and OpenGL‘s FFP material
- But a HUGE problem for physically-based rendering
  - Implausible results in HDR environments
  - Multiple light bounces not correct
Improvements – Energy Conservation

- Improvement:
  - Require normalized, energy-conserving BRDF

- But which BRDF?
  - Torrance-Sparrow microfacet model with a Blinn microfacet distribution
  - Or close approximation
  - What is “close approximation“?
    - Not really defined yet

- Why so fuzzy?
  - More freedom for implementations
  - Problematic if exact port of appearance is required
Improvements – Anisotropy

- Problem:
  - BRDF of original core component not anisotropic
  - Can be limiting for some materials (e.g. brushed metal)
Improvements – Anisotropy

Solution:
- Allow anisotropic BRDF
- Shininess now SFVec2f
- But support optional (for now…)
- Fallback: use average exponent
Improvements – Layered Materials

Problem:
- Core and specular components independent in original design
- Fast, easy to implement
- Diffuse (or glossy) layer underneath specular layer not correct
Improvements – Layered Materials

Problem:

- Core and specular components independent in original design
- Fast, easy to implement
- Diffuse (or glossy) layer underneath specular layer not correct
Improvements – Layered Materials

Problem:

- Core and specular components independent in original design
- Fast, easy to implement
- Diffuse (or glossy) layer underneath specular layer not correct
Improvements – Layered Materials

Problem:
- Core and specular components independent in original design
- Fast, easy to implement
- Diffuse (or glossy) layer underneath specular layer not correct
Improvements – Layered Materials

Solution:

- Extent meaning of fresnelBlend parameter
- Now blends between
  - Physically correct
    - Fresnel reflection/refraction
    - Core beneath specular
  - Physically implausible
    - Constant reflection/refraction
    - Core and specular added
Outline

- Introduction
  - Why CommonSurfaceShader?
  - The original CommonSurfaceShader

- Improvements
  - Energy conservation
  - Anisotropy
  - Layered materials

- Open Problems

- Conclusion
Open Problems – Layered Materials

- A lot of conflicting goals
  - Generality vs. compactness
  - Portability vs. expressiveness
  - Real-time vs. off-line
  - Backwards compatibility vs. new shiny stuff

- Concrete questions
  - Core-BRDF specification?
  - Alpha vs. transparency in diffuse under specular?
  - Which features are mandatory?
  - What will go into spec?
  - More declarative SurfaceShaders?
Introduction

Why CommonSurfaceShader?

The original CommonSurfaceShader

Improvements

Energy conservation

Anisotropy

Layered materials

Open Problems

Conclusion
Conclusion

- Presented improved version of CommonSurfaceShader
  - More physically plausible
  - Did not bloat interface
  - More freedom for implementations

- Limitations
  - Some materials require compromises
  - Some cannot be modeled at all

- All in all:
  - Pretty decent and up-to-date default Material
  - Supplement (maybe even replacement) for X3D-Material
CommonSurfaceShader Revisited: Improvements and Experiences
Karsten Schwenk, Yvonne Jung, Gerrit Voß, Timo Sturm, Johannes Behr

Thank you.
karsten.schwenk@igd.fraunhofer.de