glTF 2.0: PBR Materials
GTC, May 2017

Saurabh Bhatia
Program Manager, Microsoft | glTF specification contributor
saurabh@microsoft.com | @iamSBTron
glTF - Runtime 3D Asset Delivery

New market opportunities for 3D content creation and deployment!

model/gltf+json MIME type Approved by IANA
Compact to Transmit
Fast to Load
Describes Full Scenes
Runtime Neutral
Extensible
glTF Milestones

All glTF spec development on open GitHub:
https://github.com/KhronosGroup/glTF

2012 thru 2014
Design Iteration and Multiple Implementations
Original motivation: standardized way to deliver 3D into WebGL applications

Dec 2015
glTF 1.0 Spec Ratified and Released

Oct 2016
Validator Project

Spring 2017
glTF 2.0 Spec Finalization
glTF 2.0 adds Physically Based Rendering for higher-quality materials and rendering API independence

We are here!

Significant Industry Adoption
Strong glTF Momentum

Publicly Stated Support for glTF
glTF Ecosystem

Tools

Translators

Validator

Convert | Optimize

Export

Validate

Translations

Blender DIRECT export

Unity to Sketchfab Exporter

Tools

Export

Validator

Validator

glTF open source Validator

(Will support glTF 2.0)

Apps & Engines

Translator

App & Engines

Import

Export

Drag and Drop FBX -> glTF

(coming soon)

http://gltf.autodesk.io/

Other Translators

AssImp

OBJ2GLTF

glTF Pipeline

COLLADA2GLTF

Cesium converter

COLLADA

Drag and drop COLLADA -> glTF

http://cesiumjs.org/convermodel.html

Translator

Translation

Tools

Export

Validator

Validate

Apps & Engines

Import

Export
What’s new in glTF 2.0

• Physically Based Rendering (PBR) material definitions
  - Material information stored in textures

• Graphics API neutral
  - GLSL materials moved to extension
  - Proven by implementations using WebGL, Vulkan and Direct3D

• Morph Targets

• Improvements
  - Binary glTF in core
  - Enhanced Performance
glTF 2.0 Scene Description Structure

- **.gltf (JSON)**
  - Node hierarchy, PBR material textures, cameras

- **.bin**
  - Geometry: vertices and indices
  - Animation: key-frames
  - Skins: inverse-bind matrices

- **.png, .jpg, ...**
  - Textures

Geometry

Texture based PBR materials
glTF 2.0 Physically Based Rendering

• In Core: Metallic-Roughness Material model
  - baseColor – base color
  - metallic – metalness
  - roughness – roughness

• Extension: Specular-Glossiness Material model
  - diffuse – reflected diffuse color
  - specular – specular color
  - glossiness – glossiness

The two models can be combined
Metallic Roughness

PBR Materials
Intro to PBR

- Diffuse
- Glossy Specular
- Perfect Specular
- Retro-Reflective

BRDF Lighting Equation

\[ f(l, v, h) = \text{Diff}(l, n) + \frac{F(l, h) \, G(l, v, h) \, D(h)}{4(n \cdot l)(n \cdot v)} \]

- \( l \) is the light direction
- \( v \) is the view direction
- \( h \) is the half vector
- \( n \) is the normal

The metallic-roughness material model is defined by the following properties:
- \text{baseColor} - The base color of the material
- \text{metallic} - The metalness of the material
- \text{roughness} - The roughness of the material
BRDF Diffuse

\[ \text{Diff}(l, n) = (1 - F(v \ast h)) \frac{C_{\text{diff}}}{\pi} \]

Lambertian with energy conservation

\( C_{\text{diff}} \) is the diffuse reflected color. To conserve energy, the Fresnel term from specular component is subtracted from diffuse component.

\begin{align*}
\text{const dielectricSpecular} &= \text{rgb}(0.04, 0.04, 0.04) \\
\text{const black} &= \text{rgb}(0, 0, 0) \\
C_{\text{diff}} &= \text{lerp(baseColor.rgb} \ast (1 - \text{dielectricSpecular.r}), \text{black, metallic})
\end{align*}
BRDF Specular

\[ f(l, v, h) = \text{Diff}(l, n) + \frac{F(l, h) G(l, v, h) D(h)}{4(n \cdot l)(n \cdot v)} \]

BRDF Specular from Cook-Torrance
**BRDF Specular : F**

\[
F(l, h) \cdot G(l, v, h) \cdot D(h) \over 4(n \cdot l)(n \cdot v)
\]

F is the Fresnel function used to simulate the way light interacts with a surface at different viewing angles.

\[
F(l, h) = F_0 + (1 - F_0) \cdot (1 - v \cdot h)^5
\]

Schlick Fernel model

\(F_0\) is the specular reflectance at normal incidence

\[
\text{const dielectricSpecular} = \text{rgb}(0.04, 0.04, 0.04)
\]

\(F_0 = \text{lerp(dielectricSpecular, baseColor.rgb, metallic)}\)
BRDF Specular : G

\[ F(l, h) G(l, v, h) D(h) \]
\[ \frac{1}{4(n \ast l)(n \ast v)} \]

G is the geometric occlusion derived from a normal distribution function like Smith’s function

\[ G(l, v, h) = G_1(n, l)G_1(n, v) \]

\[ G_1(n, v) = \frac{2(n \ast v)}{(n \ast v) + \sqrt{\alpha^2 + (1 - \alpha^2)(n \ast v)^2}} \]

\[ \alpha = (\text{roughness})^2 \]
BRDF Specular : D

\[
\frac{F(l, h) \ G(l, v, h) \ D(h)}{4(n \cdot l)(n \cdot v)}
\]

D is the normal distribution function like GGX that defines the statistical distribution of microfacets.

\[
D(h) = \frac{\alpha^2}{\pi \ ((n \cdot h)^2 (\alpha - 1) + 1)^2}
\]

\[
\alpha = (\text{roughness})^2
\]
Resources

• Demos
  - WebGL-PBR implementation: https://github.com/moneimne/WebGL-PBR
  - glTF 2.0 Sample Models: https://github.com/KhronosGroup/glTF-Sample-Models

• Articles
  - Substance PBR-guide: https://www.allegorithmic.com/pbr-guide
  - Good example values: https://seblagarde.wordpress.com/2014/04/14/dontnod-physically-based-rendering-chart-for-unreal-engine-4/
Calls to Action

• Implement glTF 2.0
  - glTF 2.0 spec: https://github.com/KhronosGroup/glTF/tree/2.0/specification/2.0

• Open Source Projects
  - https://github.com/KhronosGroup/glTF/issues/867

• Blender exporter
  - https://github.com/KhronosGroup/glTF-Blender-Exporter

• glTF Online Resources
  - Github Page https://github.com/KhronosGroup/glTF
  - Resource Hub https://www.khronos.org/gltf/

• Join Khronos!
  - Get directly involved in the glTF Working Group
Bonus
Specular Glossiness

PBR Materials