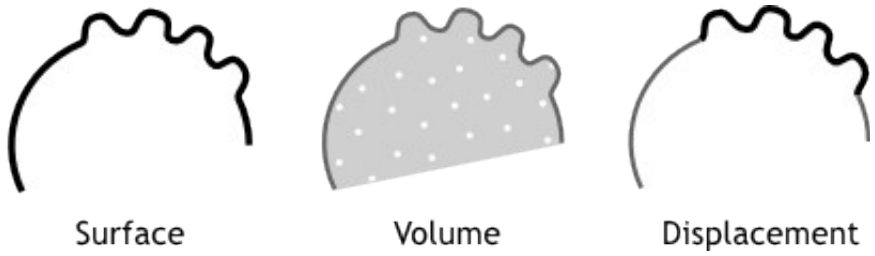
10.3.3 Render - Cycles Render Engine - Materials

Materials

- Introduction
  - `Surface Shader`
  - `Volume Shader`
  - `Displacement`
  - Energy Conservation
- Displacement
  - Type
  - Subdivision
- Surface
  - Terminology
  - BSDF Parameters
- Volume
  - Volume Shaders
  - Density
  - Volume Material

Introduction

Materials define the appearance of meshes, curves and other objects. They consist of three shaders, defining the appearance of the surface of the mesh, the volume inside the mesh, and displacement of the surface of the mesh.



Surface Shader

The surface shader defines the light interaction at the surface of the mesh. One or more BSDF s specify if incoming light is reflected back, refracted into the mesh, or absorbed.

Emission defines how light is emitted from the surface, allowing any surface to become a light source.

Volume Shader

When the surface shader does not reflect or absorb light, it enters into the volume. If no volume shader is specified, it will pass straight through to the other side of the mesh.

If it is defined, a volume shader describes the light interaction as it passes through the volume of the mesh. Light may be scattered, absorbed, or emitted at any point in the volume.

A material may have both a surface and a volume shader, or only one of either. Using both may be useful for materials such as glass, water or ice, where you want some of the light to be absorbed as it passes through the surface, combined with e.g. a glass or glossy shader at the surface.

Displacement

The shape of the surface and the volume inside it may be altered by displacement shaders. This way, textures can then be used to make the mesh surface more detailed.

Depending on the settings, the displacement may be virtual, only modifying the surface normals to give the impression of displacement, which is known as bump mapping, or a combination of real and virtual

displacement.

Energy Conservation

The material system is built with physics-based rendering in mind, cleanly separating how a material looks and which rendering algorithm is used to render it. This makes it easier to achieve realistic results and balanced lighting, though there are a few things to keep in mind.

In order for materials to work well with global illumination, they should be, speaking in terms of physics, energy conserving. That means they cannot reflect more light than comes in. This property is not strictly enforced, but if colors are in the range 0.0 to 1.0, and BSDF s are only mixed together with the Mix Shader node, this will automatically be true.

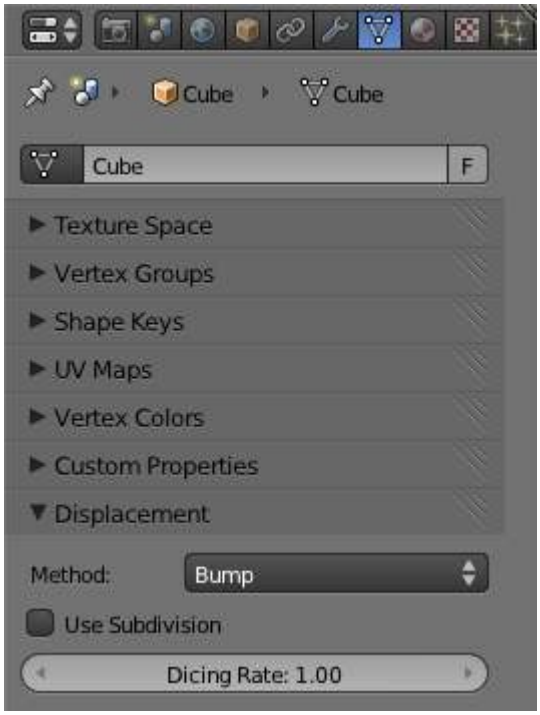
It is however possible to break this, with color values higher than 1.0 or using the Add Shader node, but one must be careful when doing this to keep materials behaving predictably under various lighting conditions. It can result in a reflection adding light into the system at each bounce, turning a BSDF into a kind of emitter.

Displacement

Implementation not finished yet, marked as an experimental feature.

The shape of the surface and the volume inside its mesh may be altered by the displacement shaders. This way, textures can then be used to make the mesh surface more detailed.

Type



Depending on the settings, the displacement may be virtual, only modifying the surface normals to give the impression of displacement, known as bump mapping, or a combination of real and virtual displacement. The

displacement type options are:

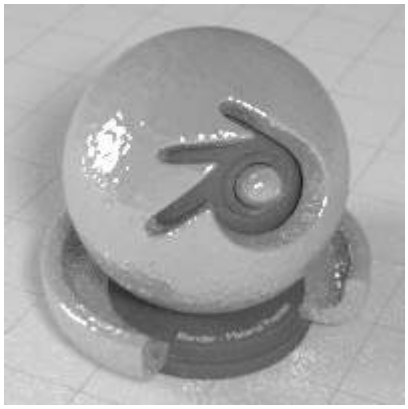
True Displacement
: Mesh vertices will be displaced before rendering, modifying the actual mesh. This gives the best quality results, if the mesh is finely subdivided. As a result this method is also the most memory intensive.

Bump Mapping
: When executing the surface shader, a modified surface normal is used instead of the true normal. This is a quick alternative to true displacement, but only an approximation. Surface silhouettes will not be accurate and there will be no self-shadowing of the displacement.

Displacement + Bump
: Both methods can be combined, to do displacement on a coarser mesh, and use bump mapping for the final details.

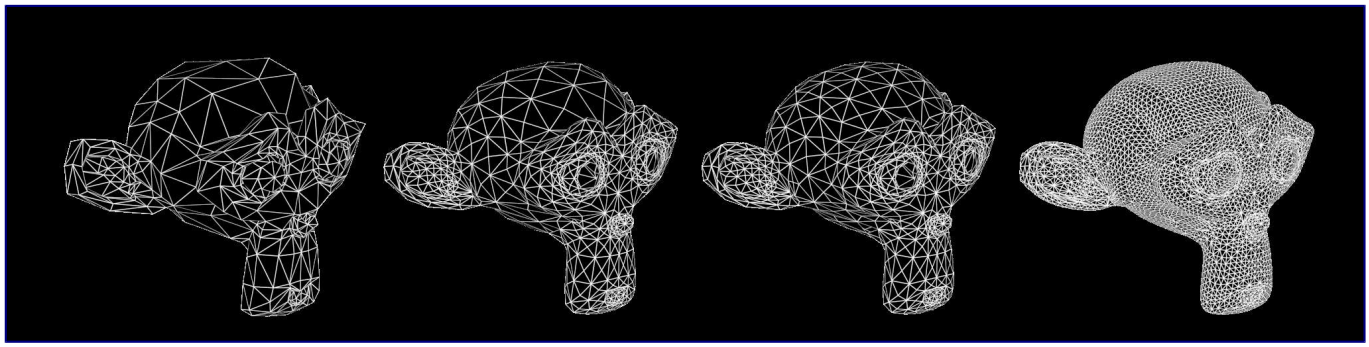
Subdivision

Implementation not finished yet, marked as an experimental feature.



Bump Mapped Displacement

When using *True Displacement* or *Displacement + Bump* and enabling *Use Subdivision* you can reduce the **Dicing Rate** to subdivide the mesh. This only affects the render and does not show in the viewport (but does show in *Rendered Shading Mode*). Displacement can also be done manually by use of the Displacement Modifier.



Subdivision Off - On, Dicing Rate 1.0 - 0.3 - 0.05 (Monkeys look identical in viewport, no modifiers)

Surface

The surface shader defines the light interaction at the surface of the mesh. One or more BSDF s specify if

incoming light is reflected back, refracted into the mesh, or absorbed.

Emission defines how light is emitted from the surface, allowing any surface to become a light source.

Terminology

BSDF stands for bidirectional scattering distribution function.
: It defines how light is reflected and refracted at a surface.

Reflection BSDF s
: Reflect an incoming ray on the same side of the surface.

Transmission BSDF s
: Transmit an incoming ray through the surface, leaving on the other side.

Refraction BSDF s are a type of Transmission,
: Transmitting an incoming ray and changing its direction as it exits on the other side of the surface.

BSDF Parameters

A major difference from non-physically based renderers is that direct light reflection from lamps and indirect light reflection of other surfaces are not decoupled, but rather handled using a single BSDF. This limits the possibilities a bit, but we believe overall it is helpful in creating consistent-looking renders with fewer parameters to tune.

For glossy BSDF s, **roughness** parameters control the sharpness of the reflection, from 0.0 (perfectly sharp) to 1.0 (very soft). Compared to **hardness** or **exponent** parameters, it has the advantage of being in the range 0.0..1.0, and as a result gives more linear control and is more easily textureable. The relation is roughly: *roughness = 1 - 1/hardness*

Volume

Volume rendering can be used to render effects like fire, smoke, mist, absorption in glass, and many other effects that can’t be represented by surface meshes alone.

To set up a volume, you create a mesh that defines the bounds within which the volume exists. In the material you typically remove the surface nodes and instead connect volume nodes to define the shading inside the volume. For effects such as absorption in glass you can use both a surface and volume shader. The world can also use a volume shader to create effects such as mist.

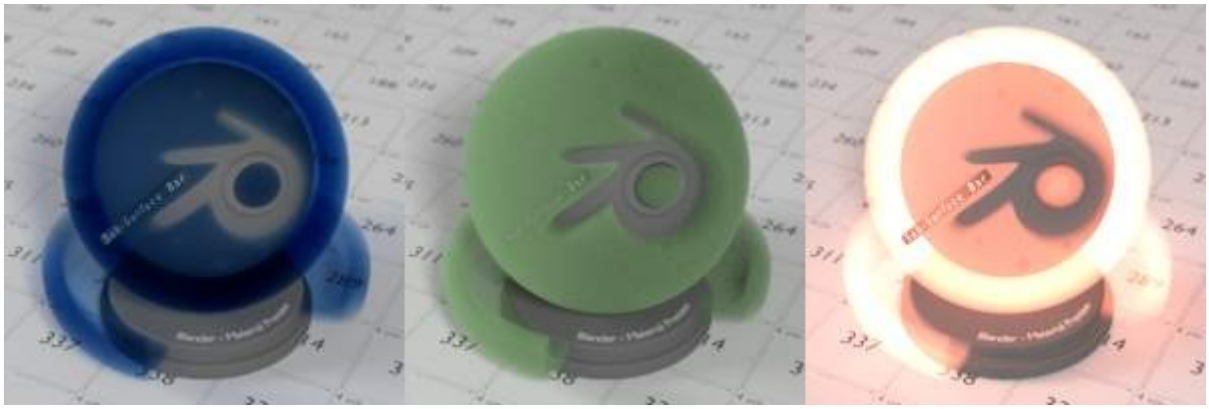
Volume Shaders

We support three volume shader nodes, that model particular effects as light passes through the volume and interacts with it.

- Volume Absorption will absorb part of the light as it passes through the volume. This can be used to shade for example black smoke or colored glass objects, or mixed with the volume scatter node. This node is somewhat similar to the transparent BSDF node, it blocks part of the light and lets other light pass straight through.
- Volume Scatter lets light scatter in other directions as it hits particles in the volume. The anisotropy

defines in which direction the light is more likely to scatter. A value of 0 will let light scatter evenly in all directions (somewhat similar to the diffuse BSDF node), negative values let light scatter mostly backwards, and positive values let light scatter mostly forward. This can be used to shade white smoke or clouds for example.

- Emission will emit light from the volume. This can be used to shade fire for example.



Volume Shader: Absorption / Absorption + Scatter / Emission

Density

All volume shaders have a density input. The density defines how much of the light will interact with the volume, getting absorbed or scattered, and how much will pass straight through. For effects such as smoke you would specify a density field to indicate where in the volume there is smoke and how much (density bigger than 0), and where there is no smoke (density equals 0).

Volumes in real life consist of particles, a higher density means there are more particles per unit volume. More particles means there is a higher chance for light to collide with a particle and get absorbed or scattered, rather than passing straight through.

Volume Material

Interaction with the Surface Shader

A material may have both a surface and a volume shader, or only one of either. Using both may be useful for materials such as glass, water or ice, where you want some of the light to be absorbed as it passes through the surface, combined with e.g. a glass or glossy shader at the surface.

When the surface shader does not reflect or absorb light, it enters into the volume. If no volume shader is specified, it will pass straight through to the other side of the mesh. If it is defined, a volume shader describes the light interaction as it passes through the volume of the mesh. Light may be scattered, absorbed, or emitted at any point in the volume.

Mesh Topology

Meshes used for volume render should be closed and manifold. That means that there should be no holes in the mesh. Each edge must be connected to exactly 2 faces such that there are no holes or T-shaped faces where 3 or more faces are connected to an edge.

Normals must point outside for correct results. The normals are used to determine if a ray enters or exits a volume, and if they point in a wrong direction, or there is a hole in the mesh, then the renderer is unable to decide what is the inside or outside of the volume.

These rules are the same as for rendering glass refraction correctly.

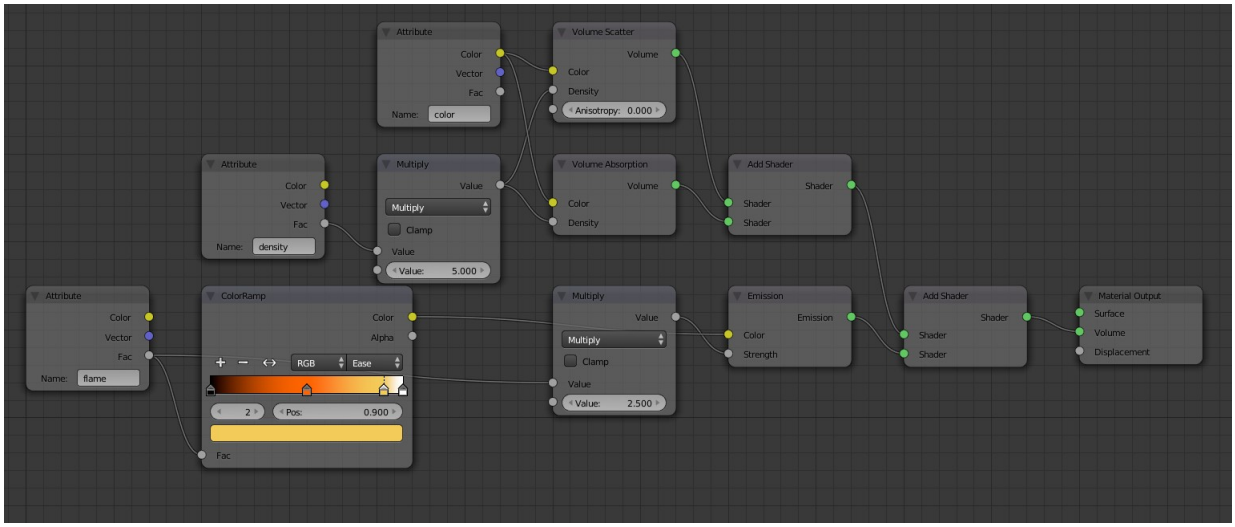
Volume World

A volume shader can also be applied to the entirely world, filling the entire space.

Currently this is most useful for night time or other dark scenes, as the world surface shader or sun lamps will have no effect if a volume shader is used. This is because the world background is assumed to be infinitely far away, which is accurate enough for the sun for example. However for modeling effects such as fog or atmospheric scattering, it is not a good assumption that the volume fills the entire space, as most of the distance between the sun and the earth is empty space. For such effects it is be better to create a volume object surrounding the scene. The size of this object will determine how much light is scattered or absorbed.

Smoke

Creating a smoke material for cycles can be difficult however the image below shows a good setup on how to do this.



Smoke and Fire Material

Scattering Bounces

Real world effects such as scattering in clouds or subsurface scattering require many scattering bounces. However unbiased rendering of such effects is slow and noisy. In typical movie production scenes only 0 or 1 bounces might be used to keep render times under control. The effect you get when rendering with 0 volume bounces is what is known as “single scattering”, the effect from more bounces is “multiple scattering”.

For rendering materials like skin or milk, the subsurface scattering shader is an approximation of such multiple scattering effects that is significantly more efficient but not as accurate.

For materials such as clouds or smoke that do not have a well defined surface, volume rendering is required. These look best with many scattering bounces, but in practice one might have to limit the number of bounces to

keep render times acceptable.

Limitations

Currently we do not support:

- Correct ray visibility for volume meshes

Not available on GPU:

- Smoke/Fire rendering
- Equi Angular / MIS Volume Sampling
- Volume Multi Light sampling

Texture Editing

3D viewport draw types, UV mapping, and texture painting work somewhat differently when Cycles is enabled. UV Maps no longer get image textures assigned themselves; rather they must always be assigned by adding an image texture node to a material.

3D Viewport Draw Types

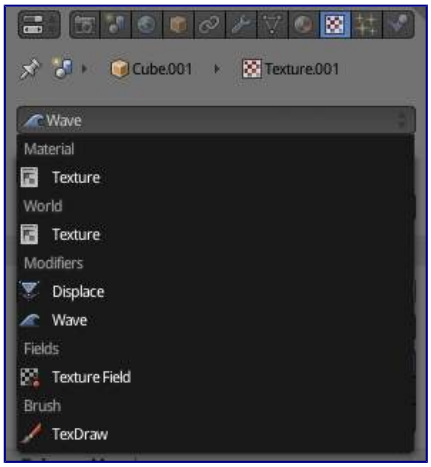
The Texture draw types used for Blender Internal have been replaced by three others in Cycles:

Texture
: This draw mode is used for editing, painting and mapping individual textures. Lighting is the same as in solid mode, so this is similar to the existing textured solid for Blender Internal. The texture drawn is the active image texture node for the material.

Material
: A simplified version of the entire material is drawn using GLSL shaders. This uses solid lighting, and also is mostly useful for editing, painting and mapping textures, but while seeing how they integrate with the material.

Rendered
: In this draw mode the render engine does the drawing, interactively refining the full rendered image by taking more samples. Unlike offline rendering, objects still use the viewport rather than render resolution and visibility.

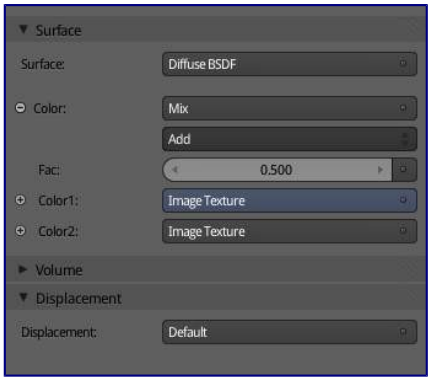
Texture Properties



In the texture properties, the texture can now be selected from a list that contains all texture nodes from the world, lamps and materials, but also from e.g. modifiers, brushes and physics fields.

For shading nodes, the available textures are Cycles textures. For others, Blender textures are still used, but this will change in the future.

Painting & UV Editing



For texture paint mode, the image that is painted on is taken from the active image texture node. This can be selected in the node editor or the texture properties, and it is indicated as blue in the material properties.

For UV mapping, the active UV map as specified in the mesh properties is used. Assigning images in the image editor also affects the active image texture node.