Hand-Drawn Shader Pack

Description

This package contains shaders that create a number of stylised “hand-drawn” effects. Separate shaders render the outline and fill of objects, and these may be combined in different ways to create a wide variety of effects. The package contains example materials to mimic the style of fine pencil, ballpoint pen, felt-tip pen, chalk, and crayon.

This document contains a brief description of each shader. The complete source code of each shader is comprehensively commented, and you are encouraged to examine them and modify them as desired!

Revision History

Version 1.2 (Feb 2015)

- Shaders upgraded to be compatible with Unity 5.
- New demo scene added

Version 1.1 (Mar 2014)

- Rewrite of Multitextured shader to accept fill textures as RGBA channels of a single image, rather than requiring four separate RGB images.
- Various code improvements to enhance performance on mobile devices

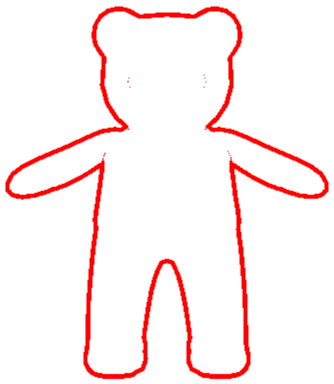
Version 1.0 (Nov 2013)

- Initial Release

Outline Shaders

Simple

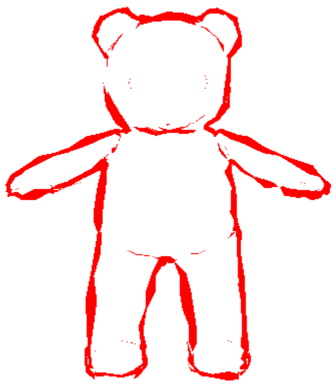
Shader: Hand-Drawn/Outline/Simple
File: Outline_Simple.shader
Used in Following Example Materials: Felt-tip Pen, Pencil



The Simple Outline shader creates a single-colour outline of set width around the outside of an object. The RGB components of the *_OutlineColour* parameter determine the colour of the outline, while the A value sets transparency. The *_OutlineWidth* parameter controls the width of the outline. If the *CONSTANT_THICKNESS* directive is defined then the outline width remains constant however far the object lies from the camera. If *CONSTANT_THICKNESS* is not defined (e.g. by commenting out line 93 of Outline_Simple.shader) then the outline width scales with distance from camera.

Animated

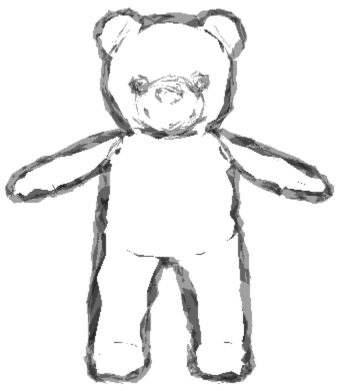
Shader: Hand-Drawn/Outline/Animated
File: Outline_Animated.shader



The Animated Outline shader extends the functionality of the Simple Outline shader by randomly adjusting the position of the outline vertices, leading to an imprecise “scribbled” outline. The maximum offset by which vertices are shifted is controlled by the *_Scribbliness* parameter. The rate at which the outline is redrawn is set by the *_RedrawRate* parameter (specified in frames per second). Setting *_RedrawRate = 0* will lead to the outline distortion not being animated at all.

Overdrawn

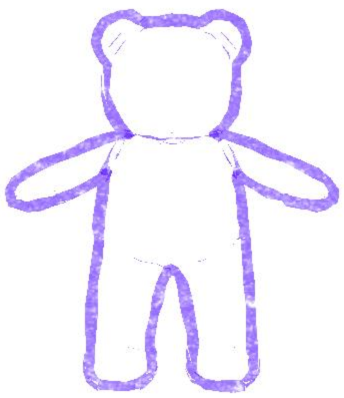
Shader: Hand-Drawn/Outline/Overdrawn
File: Outline_4Pass.shader
Used in Following Example Materials: Ink



The Overdrawn Outline extends the functionality of the Animated Outline shader by using several passes (default 4) to overlay the outline several times on top of itself. Combined with a random offset for each pass controlled by the _*Scribbliness* factor, this leads to a sketched effect where the same line segment is retraced several times.

Textured

Shader: Hand-Drawn/Outline/Animated+Textured
File: Outline_AnimatedTextured.shader
Used in Following Example Materials: Crayon



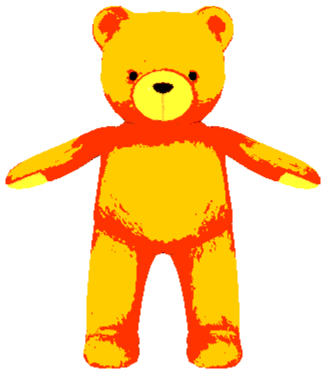
The Textured Outline exposes the same functionality as the animated shader, with the additional ability to specify a texture, _*OutlineTex*, that is blended with the _*OutlineColour* to determine the outline around the model. Note that rather than being calculated relative to the model, the outline texture is overlaid in screen space, giving the effect that the entire screen area is a consistent, rough surface.

Fill Shaders

Quantized

Shader: Hand-Drawn/Fill/Quantized
File: Fill_Quantized.shader

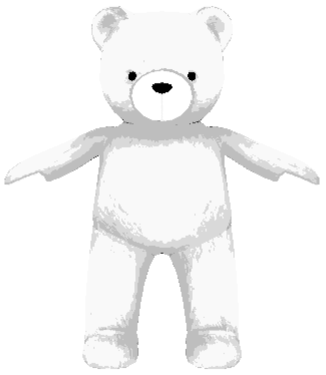
Used in Following Example Materials: Felt-tip Pen



The Quantized Fill shader takes a base diffuse texture and applies a simple quantization to reduce the number of discrete colours used in the output (a technique also known as posterization). This is useful to simulate materials in which the colour palette is limited – e.g. a set of felt-tip pens. The degree of quantization is controlled by the _*QuantizationLevels* parameter – smaller values lead to a more reduced palette.

Smoothed Greyscale

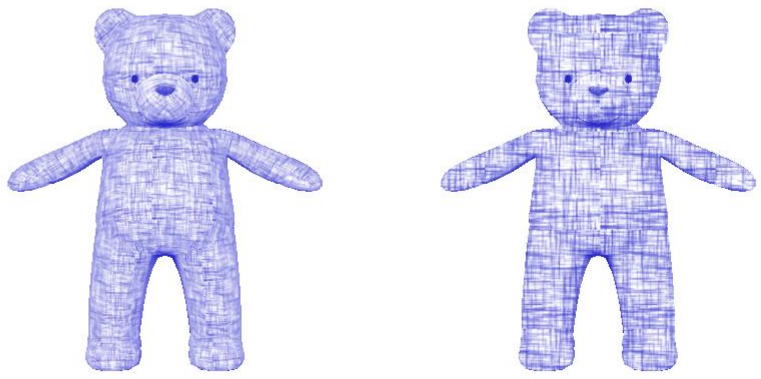
Shader: Hand-Drawn/Fill/SmoothedGreyscale
File: Fill_SmoothedGreyscale.shader
Example Material: Ink



The Smoothed Greyscale Fill shader takes a base diffuse texture, and smooths it using a simple averaging of neighbouring pixels. The resulting smoothed image is then quantized and contrast boosted before being converted to greyscale. The result is an effective shading using a limited monochrome palette, such as might be used in an ink painting, for example.

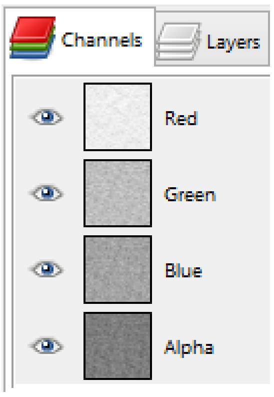
Fill MultiTextured

Shader: Hand-Drawn/Fill/MultiTextured(Alpha) / Hand-Drawn/Fill/MultiTextured(Multiply)
File: Fill_Multitextured(Alpha).shader / Fill_Multitextured(Multiply).shader
Example Material: Chalk, Ballpoint Pen



Hatching texture applied in modelspace (l) and screenspace (r)

The Multitextured Fill shader takes a base diffuse texture and an additional RGBA image, in which the RGBA channels contain four monochrome fill textures, increasing in density from R (least dense) -> G -> B -> A (most dense).



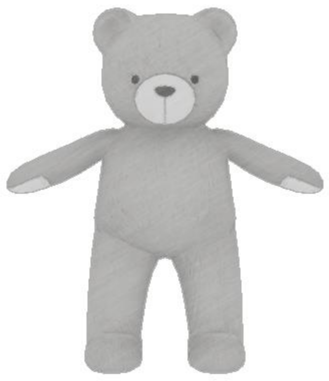
Example RGBA fill texture, demonstrating increasing density of channels.

The first pass of the shader calculates the appearance of the object with the diffuse texture applied, accounting for lighting in the scene (lightmaps ambient, spherical harmonic, and main directional light). The brightness of each pixel in the resulting image is then sampled, and this is used to select the appropriate replacement fill texture. Fill textures may be applied in model space or in screen space, based on *#define SCREEN_ALIGNED* (line 59). Finally, the composite fill texture is tinted based on the supplied *_FillColour* value. This technique can be used to create, e.g. hatched shading effects, in which the fill textures represent increasing levels of hatch intensity. The offset applied to the fill textures may be periodically changed by a random factor based on the chosen *_RedrawRate*. This can create the effect of a series of hand-shaded frames of animation.

There are two versions of the shader – one uses alpha blending (as used by Chalk material) and the other multiplicative blending (as used by Ballpoint Pen) – the choice depends on the type of material you are trying to simulate. You may also need to modify the shader depending on whether the RGBA textures supplied are white on black or black on white.

Textured

Shader: Hand-Drawn/Fill/Textured
File: Fill_Textured.shader
Example Material: Pencil



The Textured Fill shader takes a base diffuse texture, a fill texture, and a fill colour. It returns the RGB value from the base colour, with the alpha channel based on the inverse brightness of the underlying fill texture. So, areas in which the base texture was very dark have high alpha (mostly opaque), whereas areas in which the base texture was light become increasingly less alpha (mostly transparent). This simulates the effect of varying pressure with which a pencil is applied when shading light and dark areas of an image, for example.