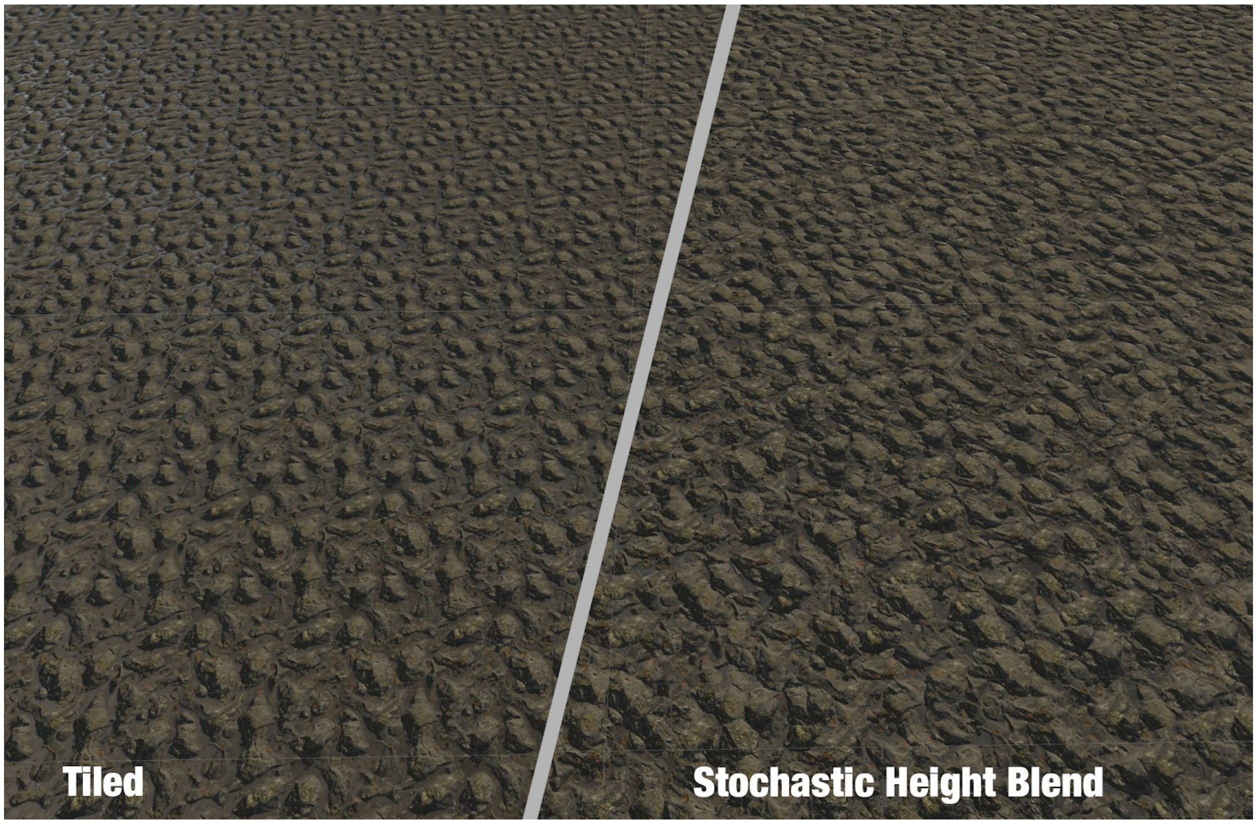
Stochastic Height Blending Node

(For Amplify Shader Editor)

Documentation

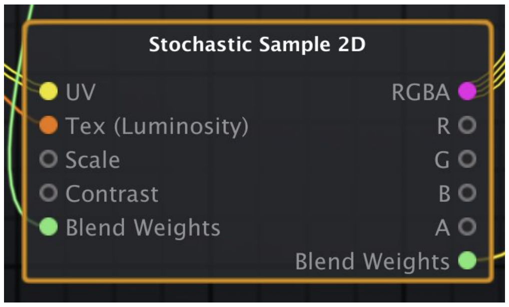


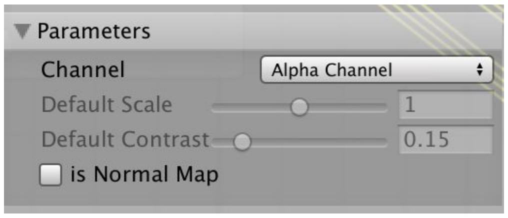
Once installed, this package adds a new node to the Amplify Shader Editor.

Stochastic Sample 2D

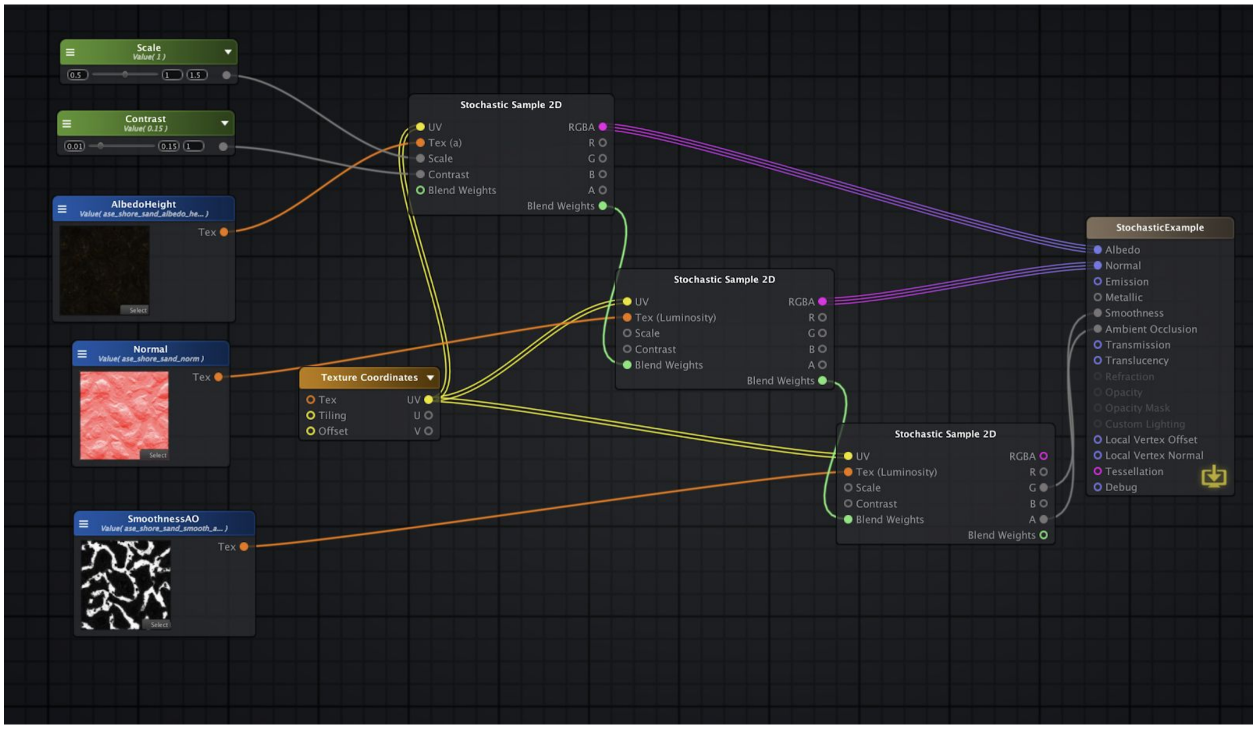
You can find the Stochastic Sample 2D node under the Stochastic Menu. It functions much like a texture sample node, but with a few extra inputs and outputs.

Internally, the node constructs 3 randomly placed virtual triangles on your texture and samples them, then blends the texture together based on a height blend operator. This creates the effect of a surface which is similar to your original texture, but never repeats or tiles.

 Stochastic Sample 2D UV Tex (Luminosity) Scale Contrast Blend Weights RGBA R G B A Blend Weights	Inputs: - **Texture** and **UV** are the same as a standard texture node - **Contrast** controls how wide the blend area is between the samples. I generally prefer a low value here, which produces a very sharp blend. Values should stay in the 0.001 to 0.999 range. Contrast is ignored if the Blend input is used. - **Scale** controls how large the virtual triangles are. When a large scale value is used, a virtual triangle may cover an area larger than the texture, causing some tiling to appear. When low values are used, the texture may start to synthesize from choosing very small sections of the original texture. Best values are between 0.5 and 1.5. Scale is ignored if the Blend input is used. - The Blend Weights input is used to chain the blending result from previous samples into the current one. For instance, you might sample the diffuse and base the blend off of a height map in it's alpha channel, then use those results to blend the normal and specular textures in the same manner. - Outputs:

 Parameters Channel: Alpha Channel Default Scale: 1 Default Contrast: 0.15 is Normal Map	- **RGBA**, **R**, **G**, **B**, **A** are the same as the regular sampler node, and output the results of the sample - **Blend Weights** outputs the blend weights from the height map operator. Properties - Channel selects which channel of the texture is used for the blend operator. Luminosity can also be selected, which uses the Luminosity of the RGB channels of the texture. This is ignored if the blend is provided from a previous node. - Default values for **Scale** and **Contrast** are available. These are disabled when the Blend Weight input is used. - **Is Normal Map** should be set when sampling normal maps. -

Examples



An example is included in the Examples folder. In this example, a height map is stored in the albedo's alpha channel. The result of that blend is chained to the normal and smoothness/ao samplers via the Blend input and output. Textures, Scale and Contrast is exposed for the user. Note that the third texture has smoothness in the G channel, and AO in the alpha channel.

Performance

A stochastic sampler requires 3 samples of the original texture, and a some computation to compute the noise required to generate the offset 'virtual' triangles. Overall, it is roughly comparable to a technique like triplanar texturing in terms of performance cost.

Comparisons

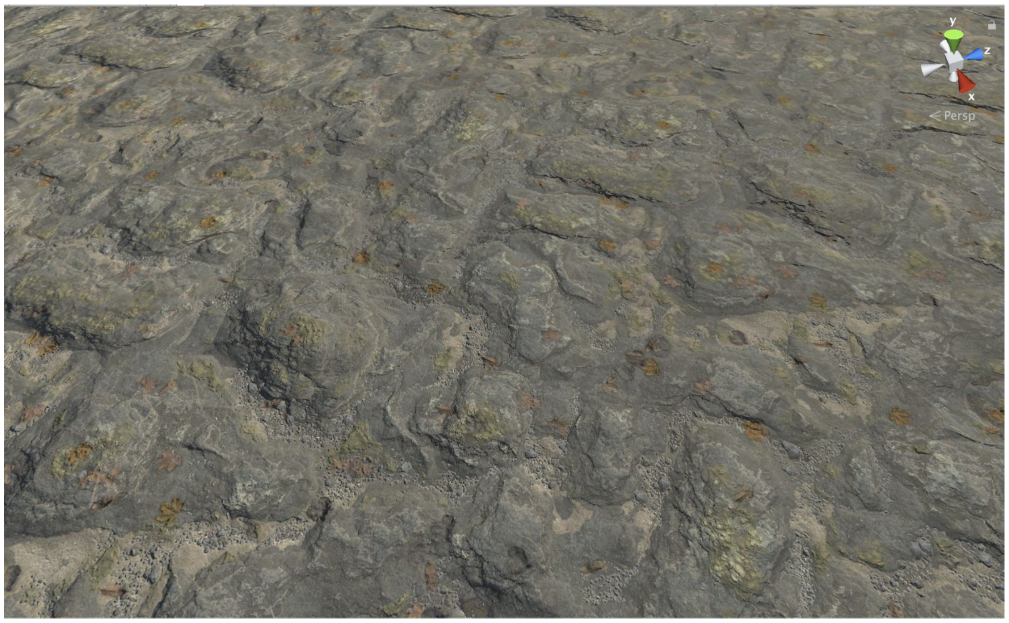
The idea for this technique is based off my work with Texture Clustering (in MegaSplat and MicroSplat) and Erik Heitz's paper "High Performance By Example Noise using a Histogram Preserving Blending Operator".

Texture Clustering uses multiple variations of a texture, chosen by a noise function, blended together by a height blending operator. It can produce a wider variation of surface detail that any of the other techniques, but requires significant authoring of variation textures, and more textures accessed per pixel.

Stochastic Sampling with a Histogram Preserving Blending Operator does not require variation textures, but requires a heavy preprocessing of texture data into a custom format, extra lookup textures which need to be generated, special handling for texture compression, as well as more complex runtime calculations.

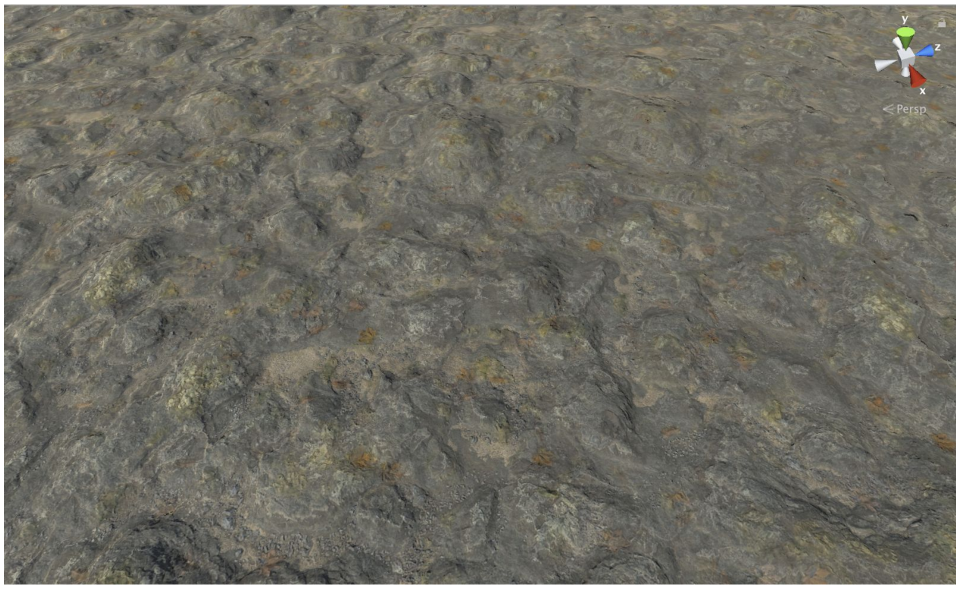
Stochastic Height Blending provides the best of both worlds. No additional textures are needed to produce a non-tiling surface, textures can remain in their native formats with no special code for handling compression formats, and the runtime calculations are relatively simple. Further, I believe this technique better preserves the original look of the texture, as the area of the blends is much smaller, allowing your original texture detail to shine through more often. Here are some comparison shots:

Tiled Texture



/

Stochastic Sampling with Histogram preservation:



Stochastic Height Blending: