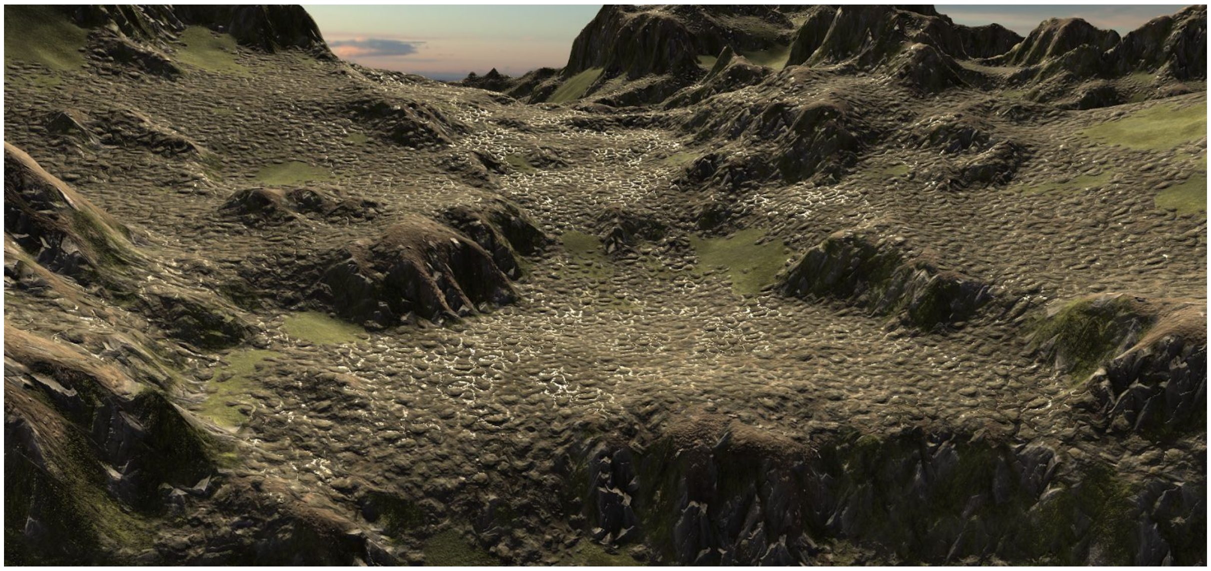
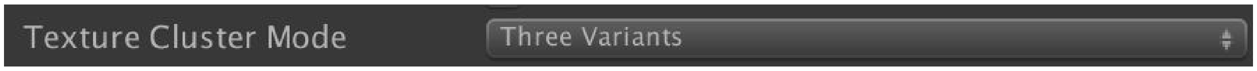
MicroSplat

Texture Cluster Module

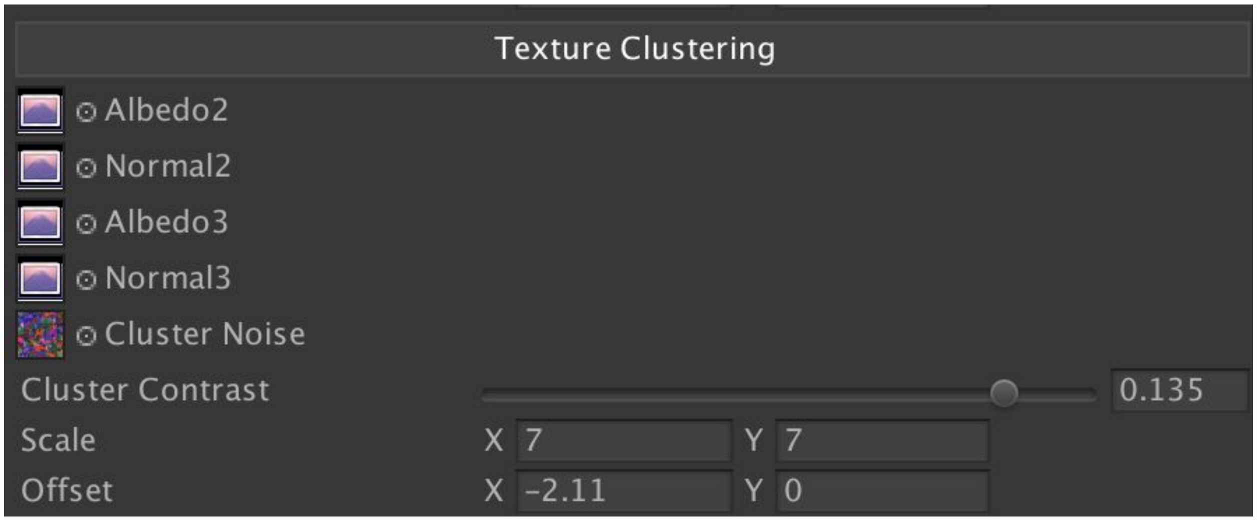


Texture clustering is a technique I invented when creating MegaSplat. The basic idea is to counteract tiling artifacts by using more textures, blending in and out of variations of a texture rather than using a single texture. Because of the way the technology in MegaSplat works, this blending was already being performed, so the technique basically came nearly for free.

MicroSplat uses a very different blending technique than MegaSplat does, and uses the traditional Unity Terrain tools, which are limited to 16 textures. However, a new technique was developed to do Texture Clustering in MicroSplat.


Texture Cluster Mode: Three Variants

Once installed, the Texture Cluster mode option will show up in the Shader Generator section. It can be set to use 2 or 3 variants per texture. What this means is that if you have a 16 texture terrain, you can provide a second set of 16 textures to blend with, or a third as well. This means while you are still limited to using 16 control textures, each of these can be texture clustered with up to 2 other textures, giving you 48 textures on a single terrain.


Texture Clustering

Albedo2

Normal2

Albedo3

Normal3

Cluster Noise

Cluster Contrast		0.135		
Scale	X	7	Y	7
Offset	X	-2.11	Y	0

Once enabled, the Texture Cluster section of the shader will let you assign a second or third set of texture arrays. We'll get to creating those in a moment. A noise texture is used to control the patterning of the texture, which can be scaled and offset across the terrain.

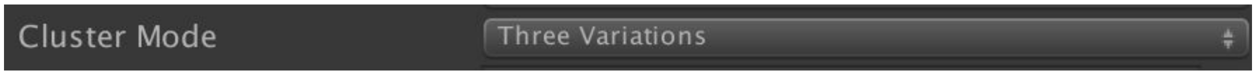
The Cluster Contrast is a separate Interpolation Contrast setting just for the height blending between clusters. This lets you have a soft blend between terrain types, but a sharp blend in the internal clustering.

The Noise Boost will sharpen the noise texture, in effect preferring to blend texture more, or switch between the textures instead.

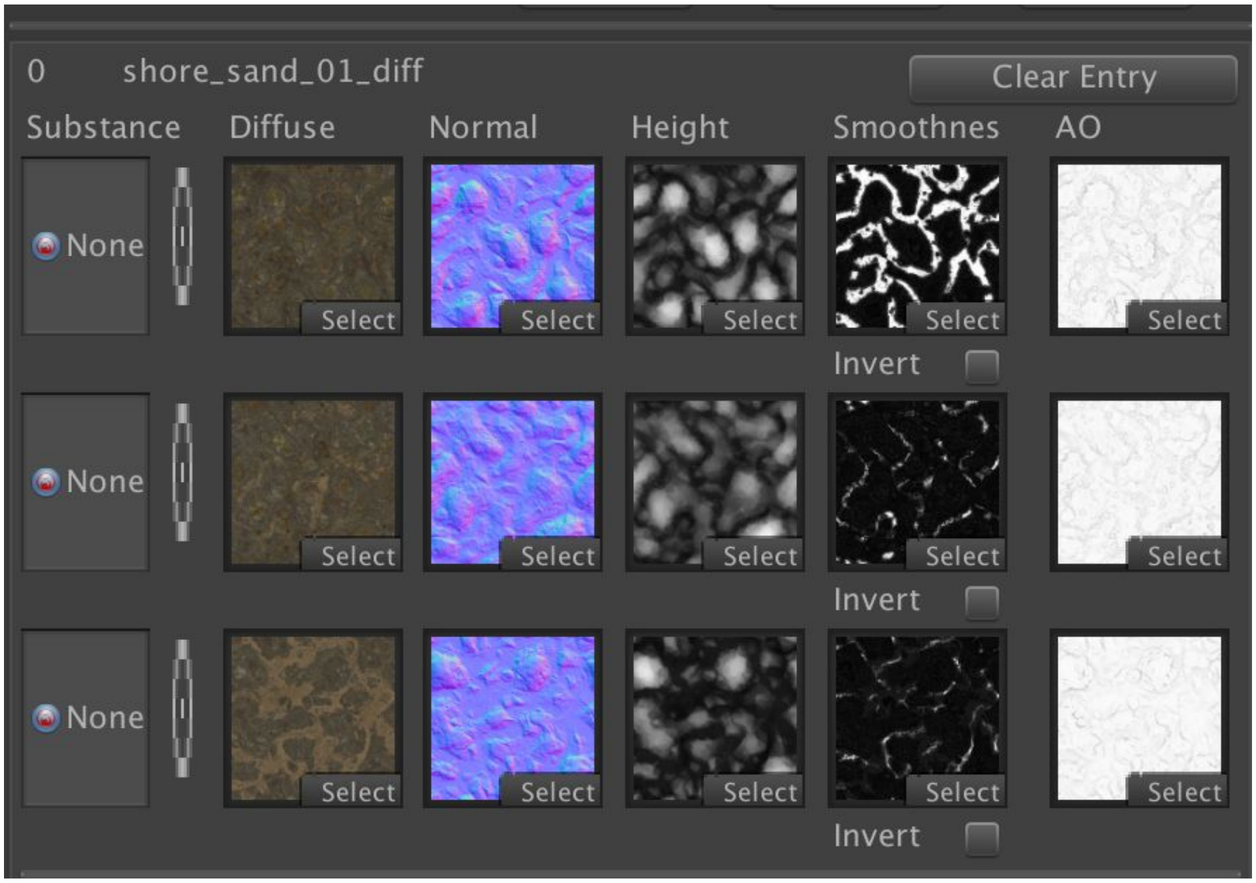
Both properties are available on a per texture basis. Note that all per-texture properties apply to the entire cluster.

Packing Textures

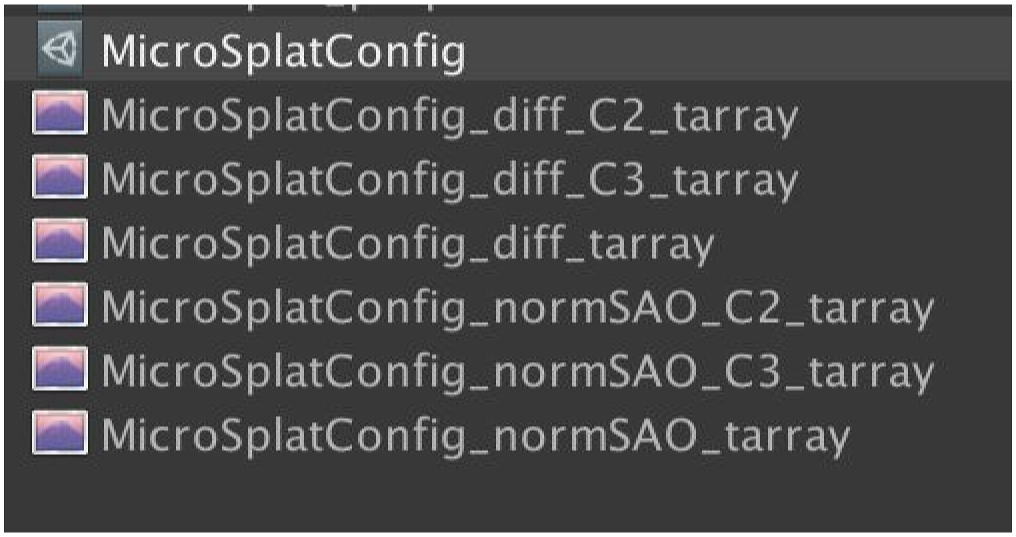
If you have turned this option on, your terrain likely became very white. This is because you need to pack extra textures in for the clustering to use.


Cluster Mode: Three Variations

If you go back to your Texture Array Config (usually located in the MegaSplatData directory by your terrain), a new option will be available in the config, allowing you to specify how many texture variations you plan to use for each texture.



Once set, each texture selection area will be replaced with either 2 or 3 rows for your texture variations. Once you have finished setting up your variants, pressing update will export out 2 or 3 sets of texture arrays, which you can then assign to your shader.


MicroSplatConfig
MicroSplatConfig_diff_C2_tarray
MicroSplatConfig_diff_C3_tarray
MicroSplatConfig_diff_tarray
MicroSplatConfig_normSAO_C2_tarray
MicroSplatConfig_normSAO_C3_tarray
MicroSplatConfig_normSAO_tarray

The config, and resulting arrays. Assign them to your material, and every texture will be varied based on the noise settings.

Performance

A 3 variant texture cluster is going to sample 3 times as much as a non-clustered version of the same shader, and sample from 3 times as many textures. This can be quite expensive when combined with things like Triplanar Texturing (also a 3x) and distance resampling (2x). However, there are many tools to help manage this in MicroSplat. You can use a 2 variant cluster and still get high quality clustering in many cases. Alternatively, changing the blend quality from Best to Balanced or Fastest can drastically cut down the number of samples.

Blend Quality	Base	Texture Cluster 3	Distance Resample	Triplanar	Total
Best	8 samples	*3	*2	*3	144 samples
Balanced	6 samples	*3	*2	*3	108 samples
Fastest	4 samples	*3	*2	*3	72 samples
Best	8 samples	*3	*<disabled>*	*3	72 samples
Balanced	6 samples	*3	*<disabled>*	*3	54 samples
Fastest	4 samples	*3	*<disabled>*	*3	36 samples
Best	8 samples	*3	*<disabled>*	*<disabled>*	24 samples
Balanced	6 samples	*3	*<disabled>*	*<disabled>*	18 samples
Fastest	4 samples	*3	*<disabled>*	*<disabled>*	12 samples

At high sample counts, most shaders will become bottlenecked by memory bandwidth, especially if those samples are all going to different textures or different areas of a texture. However, if bandwidth is the bottleneck, the Clustering can actually help that.

Imagine we design our terrain with large textures- Like 2k textures. Since they are so high resolution, we can stretch them over a larger area, which helps reduce tiling artifacts. But at that size, the GPU has to access a lot of memory to draw the whole texture.

With Texture Clustering, we can create a surface from multiple textures blended together which doesn’t tile, because it is constantly changing between several textures. So we could create this from 3 1k textures instead of 1 2k texture, and cover a larger area without tiling artifacts! This actually reduces the total memory we have to sample, and thus reduces the total bandwidth.

Tips & Tricks

Texture Clusters are not just limited to creating variations of the same texture. You could put any three textures together, and they will get height blended based on the noise pattern.

The noise pattern is easily customized. While it is global for all of the clusters, it is just a texture with a weight for each of the textures in the R, G, and B channels. Note that you should never have pure black in this image, as some texture should always have some weight. Changing the noise texture can have huge effects on how the clusters look.

For the best anti-tiling, you want the textures to change often enough that you don't see texture repeat, but no so often that you don't see the entirety of a texture. This means the noise texture used can be very small, which helps performance, because it only needs to provide a new value every so often.

Finally, I would recommend tools like Substance Designer or Filter Forge for creating textures in this manner. These tools are procedural, and make it very easy to generate multiple variants of a texture. I would also recommend playing with texture clusters of completely different textures to get a feel for how the system works, this can make it very obvious how textures are chosen and allow you to get the correct noise pattern and scale for high quality results before applying it to subtle variations.