PxrMultiTexture

PxrMultiTexture was built to reference up to 10 textures, blend them in various ways and allow for various map selection methods.

Like all texture style nodes, this node takes a manifold which describes a 2D domain to apply a the texture to. The default behavior, if no manifold is attached, is to apply over the s,t domain defined on the geometry.

PxrMultiTexture is only compatible with PxrRoundCube, PxrTileManifold and PxrRandomTextureManifold. Other manifolds are NOT usable with this pattern.

PxrMultiTexture doesn't support Texture atlas format files: it makes no sense for its usage. This plugin does support RTX procedural texture plugins, though.

Input Parameters

Multi Manifold

Provides the domain over which to apply the textures, through either PxrRoundCube or PxrTileManifold. If not connected, it will defaults to s,t.

This input is only compatible with nodes providing a "resultMulti" output, like PxrRoundCube, PxrTileManifold, PxrRandomTextureManifold.

Textures

Filename (0 to 9)

The filename of the texture. You should have at least one texture in slot 0.

Texture Parameters

These parameters are valid for all textures (0 to 9).

First Channel Offset

Offsets the first channel to be looked up.

Usually, you want to start with channel 0 (RGB of RGBA texture), but if you are using a multi-channel openexr texture, you may need to point to the correct channel.

Let's say we have a texture containing 9 channels organised in 3 RGB groups (diff.r, diff.g, diff.b, spec.r, spec.g, spec.b, ior.r, ior.g, ior.b). If you want to read the "spec" RGB channels, you will have to set firstChannel to 3.

Invert T

Inverts the t parameter for the texture lookup.

Filter

Selects different reconstruction filters that can be used during texture lookups. The filters available for PxrTexture are:
Blur
Specifies how much to blur the image retrieved from the texture file.

Mip Interpolate
Selects whether to interpolate between adjacent resolutions in the multi-resolution texture, resulting in smoother transitions between levels.

Missing Color
If there is an error opening the texture, use this color.

Missing Alpha
If there is an error opening the texture, use this alpha.

Linearize
Apply the reverse sRGB transform your texture. If you are painting textures in sRGB space (default for most paint packages) but viewing your data in data linear space, your textures will look washed out. This will apply the sRGB transform to your texture, which should make it appear visually linear again.

Randomize
Manifold patterns outputting a multi manifold encode which texture should be used using a 2D coordinate system. PxrMultiTexture can use those coordinates to compute a random seed to drive local variations.

Random Source
For each object to get a different variation, you need to select something unique about them to create a unique seed. You have a choice between the object's id and the object's name. These attributes are created by the software outputting the RIB and depending on your host, one may work better than the other.

<table>
<thead>
<tr>
<th>Value</th>
<th>Source</th>
<th>Depends on</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Object id</td>
<td>Attribute &quot;identifier&quot; &quot;float id&quot;</td>
</tr>
</tbody>
</table>
Seed

When set to zero, all PxrMultiTexture patterns using the same manifold will compute the same random seed. If you want one of these patterns to compute a different variation, set the randomSeed to a non-zero value:

Hue

Limit random hue shift. Ranges from 0 (no variation) to 1 (full variation).

Saturation

Limit random saturation scale. Ranges from 0 (no variation) to 1 (full variation).

Luminance

Limit random luminance scale. Ranges from 0 (no variation) to 1 (full variation).

Gamma

Limit random gamma variation. Ranges from 0 (no variation) to 1 (full variation).

Hue Mode

How the color hue will be randomized:

<table>
<thead>
<tr>
<th>Value</th>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Centered</td>
<td>The variation will be centered around the current value</td>
</tr>
<tr>
<td>1</td>
<td>Additive</td>
<td>the variation will be added to the current value</td>
</tr>
<tr>
<td>2</td>
<td>Subtractive</td>
<td>the variation will be subtracted from the current value</td>
</tr>
</tbody>
</table>

Saturation Mode

How the color saturation will be randomized:

<table>
<thead>
<tr>
<th>Value</th>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Centered</td>
<td>The variation will be centered around the current value</td>
</tr>
<tr>
<td>1</td>
<td>Additive</td>
<td>the variation will be added to the current value</td>
</tr>
<tr>
<td>2</td>
<td>Subtractive</td>
<td>the variation will be subtracted from the current value</td>
</tr>
</tbody>
</table>

Luminance Mode

How the color luminance will be randomized:

<table>
<thead>
<tr>
<th>Value</th>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Centered</td>
<td>The variation will be centered around the current value</td>
</tr>
</tbody>
</table>
### Gamma Mode

How the color gamma will be varied:

<table>
<thead>
<tr>
<th>Value</th>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Centered</td>
<td>The variation will be centered around the current value</td>
</tr>
<tr>
<td>1</td>
<td>Additive</td>
<td>the variation will be added to the current value</td>
</tr>
<tr>
<td>2</td>
<td>Subtractive</td>
<td>the variation will be substracted from the current value</td>
</tr>
</tbody>
</table>

### Output Parameters

- **resultRGB**
  The filtered color result. Note that all results are looked up starting at the "startChannel" offset.

- **resultR**
  The R channel result

- **resultG**
  The G channel result

- **resultB**
  The B channel result

- **resultA**
  If alpha is present and resultRGB is connected it returns the channel after RGB. If RGB is not connected it returns the first channel.