



## 13.1.9 Editors - Shader Editor - Header - Add Menu - Texture

### Table of content

Detailed Table of content.....	1
Add menu - Texture.....	7
Brick Texture.....	7
Checker Texture.....	8
Environment Texture.....	9
Gradient Texture.....	10
IES Texture.....	11
Image Texture.....	12
Magic Texture.....	14
Musgrave Texture.....	15
Noise Texture.....	17
Point Density.....	18
Sky Texture.....	20
Voronoi Texture.....	21
Wave Texture.....	23
White Noise Texture.....	25

### Detailed Table of content

#### Detailed table of content

Detailed Table of content.....	1
Add menu - Texture.....	7
Brick Texture.....	7
Inputs.....	7
Color 1, Color 2 and Mortar.....	7
Scale.....	7
Mortar Size.....	7
Mortar Smooth.....	7
Bias.....	7
Brick Width.....	7
Row Height.....	7
Properties.....	8
Offset.....	8
Frequency.....	8
Squash.....	8
Frequency.....	8
Outputs.....	8
Color.....	8
Factor.....	8
Checker Texture.....	8
Inputs.....	8
Vector.....	8
Color1, Color 2.....	8
Scale.....	8
Outputs.....	9

Color.....	9
Factor.....	9
Environment Texture.....	9
Inputs.....	9
Vector.....	9
Properties.....	9
Image.....	9
Color Space.....	9
Texture Interpolation.....	9
Linear.....	9
Closest.....	9
Cubic.....	9
Smart.....	9
Projection Method.....	10
Equirectangular.....	10
Mirror Ball.....	10
Outputs.....	10
Color.....	10
Gradient Texture.....	10
Inputs.....	10
Vector.....	10
Properties.....	10
Gradient Type.....	10
Linear.....	10
Quadratic.....	10
Easing.....	10
Diagonal.....	10
Spherical.....	10
Quadratic Sphere.....	11
Radial.....	11
Outputs.....	11
Color.....	11
Factor.....	11
IES Texture Node.....	11
Inputs.....	11
Vector.....	11
Strength.....	11
Properties.....	11
Mode.....	11
Internal.....	11
External.....	11
Outputs.....	11
Factor.....	11
Image Texture.....	12
Inputs.....	12
Vector.....	12
Properties.....	12
Image.....	12
Interpolation.....	12
Linear.....	12
Cubic.....	12
Closest.....	13
Smart.....	13

Projection.....	13
Flat.....	13
Box.....	13
Blend.....	13
Sphere.....	13
Tube.....	13
Extension.....	13
Repeat.....	13
Extend.....	13
Clip.....	13
Outputs.....	14
Color.....	14
Alpha.....	14
Magic Texture.....	14
Inputs.....	14
Vector.....	14
Scale.....	14
Distortion.....	14
Properties.....	14
Depth.....	14
Outputs.....	14
Color.....	14
Factor.....	14
Musgrave Texture.....	15
Inputs.....	15
Vector.....	15
W.....	15
Scale.....	15
Detail.....	15
Dimension.....	15
Lacunarity.....	15
Offset.....	15
Gain.....	16
Properties.....	16
Dimensions.....	16
1D.....	16
2D.....	16
3D.....	16
4D.....	16
Type.....	16
fBM (fractal Brownian Motion).....	16
Multifractal.....	16
Hybrid Multifractal.....	16
Ridged Multifractal.....	16
Hetero Terrain (Heterogeneous Terrain).....	16
Outputs.....	17
Factor.....	17
Noise Texture.....	17
Inputs.....	17
Vector.....	17
W.....	17
Scale.....	17
Detail.....	17

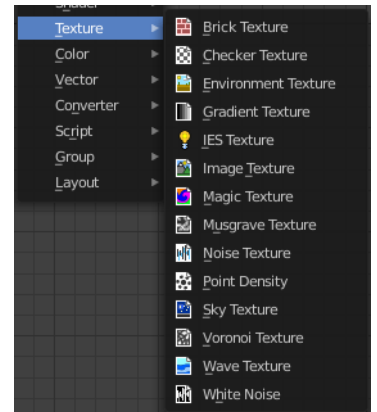
Roughness.....	17
Distortion.....	17
Properties.....	17
Dimensions.....	17
1D.....	17
2D.....	17
3D.....	18
4D.....	18
Outputs.....	18
Factor.....	18
Color.....	18
Point Density.....	18
Inputs.....	18
Vector.....	18
Properties.....	18
Point Data.....	18
Particle System.....	18
Object Vertices.....	18
Object.....	18
Particle System.....	18
Space.....	19
World Space.....	19
Object Space.....	19
Radius.....	19
Interpolation.....	19
Closest.....	19
Linear.....	19
Cubic.....	19
Resolution.....	19
Color Source.....	19
Particle Color Sources.....	19
Particle Age.....	19
Particle Speed.....	19
Particle Velocity.....	19
Vertex Color Sources.....	20
Vertex Color.....	20
Vertex Weight.....	20
Vertex Normals.....	20
Outputs.....	20
Color.....	20
Density.....	20
Sky Texture.....	20
Inputs.....	20
Vector.....	20
Properties.....	20
Sky Type.....	20
Sun Direction.....	20
Turbidity.....	20
Ground Albedo.....	21
Outputs.....	21
Color.....	21
Voronoi Texture.....	21
Inputs.....	21

Vector.....	21
W.....	21
Scale.....	21
Randomness.....	21
Properties.....	21
Dimensions.....	21
1D.....	22
2D.....	22
3D.....	22
4D.....	22
Feature.....	22
F1.....	22
Smooth F1.....	22
Distance To Edge.....	22
N-Sphere Radius.....	22
Distance Metric.....	22
Euclidean.....	22
Manhattan.....	22
Chebychev.....	22
Minkowski.....	22
Outputs.....	23
Distance.....	23
Color.....	23
Position.....	23
W.....	23
Radius.....	23
Wave Texture.....	23
Inputs.....	23
Vector.....	23
Scale.....	23
Distortion.....	23
Detail.....	23
Detail Scale.....	23
Detail Roughness.....	24
Phase Offset.....	24
Properties.....	24
Wave Type.....	24
Bands direction.....	24
Wave Profile.....	24
Saw.....	24
Sine.....	24
Triangle.....	24
Outputs.....	24
Color.....	24
Factor.....	24
White Noise Texture.....	25
Inputs.....	25
Vector.....	25
W.....	25
Properties.....	25
Dimensions.....	25
1D.....	25
2D.....	25

3D.....	25
4D.....	25
Outputs.....	25
Value.....	25

## Add menu - Texture

Here are the texture nodes. They allow you to add different texture types to the scene. The content is the same for the sub modes and the different renderers. However, a environment texture just makes sense for the world sub mode.



### Brick Texture

The Brick Texture node is used to add a procedural brick texture.

#### Inputs

##### ***Color 1, Color 2 and Mortar***

Color of the bricks and mortar.

##### ***Scale***

Overall texture scale.

##### ***Mortar Size***

The size of the filling between the bricks known as “mortar”; 0 means no mortar.

##### ***Mortar Smooth***

Blurs/softens the edge between the mortar and the bricks. This can be useful with a texture and displacement textures.

##### ***Bias***

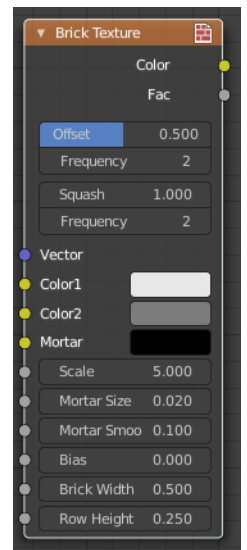
The color variation between Color 1/2. Values of -1 and 1 only use one of the two colors; values in between mix the colors.

##### ***Brick Width***

The width of the bricks.

##### ***Row Height***

The height of the brick rows.



## Properties

### **Offset**

Determines the brick offset of the various rows.

### **Frequency**

Determines the offset frequency. A value of 2 gives an even/uneven pattern of rows.

### **Squash**

Amount of brick squashing.

### **Frequency**

Brick squashing frequency.

## Outputs

### **Color**

Texture color output.

### **Factor**

Mortar mask (1 = mortar).

## Checker Texture

The Checker Texture node adds a procedural checkerboard texture.

## Inputs

### **Vector**

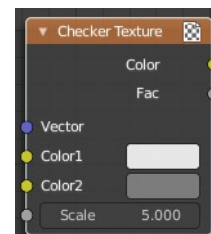
Texture coordinate to sample texture at; defaults to Generated texture coordinates if the socket is left unconnected.

### **Color1, Color 2**

Color of the checkers.

### **Scale**

Overall texture scale. The scale is a factor of the bounding box of the face divided by the scale. For example, a scale of 15 will result in 15 alternate patterns over the overall UV bounding box. Different patterns could be achieved using other nodes to give different input patterns to this socket. For example, using the Math Node.





## Outputs

### **Color**

Texture color output.

### **Factor**

Checker 1 mask (1 = Checker 1).

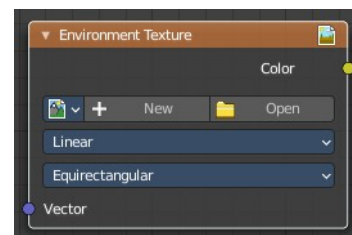
## Environment Texture

Load an environment texture to light your scene.

## Inputs

### **Vector**

Texture coordinate for texture look-up. If this socket is left unconnected, the image is mapped as environment with the Z axis as up.



## Properties

### **Image**

Image data-block used as the image source. Additional settings can be found in Sidebar > Item > Properties: These include options to control the alpha channel along with addition options for the color space. These addition options are documented with the rest of Common Image Settings.

### **Color Space**

Type of data that the image contains, either Color or Non-Color Data. For most color textures the default of Color should be used, but in case of e.g. a bump or alpha map, the pixel values should be interpreted as Non-Color Data, to avoid doing any unwanted color space conversions.

### **Texture Interpolation**

Interpolation method used for the environment texture.

#### **Linear**

Regular quality interpolation.

#### **Closest**

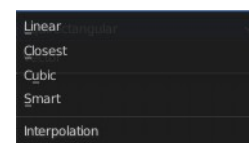
No interpolation, use closest pixel.

#### **Cubic**

Smoother, better quality interpolation.

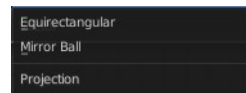
#### **Smart**

Bicubic when magnifying, otherwise Bilinear is used. This is only available for OSL.



## ***Projection Method***

Allows you to use different types of environmental maps.



### **Equirectangular**

Projection from an Equirectangular photo.

### **Mirror Ball**

Projection from an orthographic photo or mirror ball.

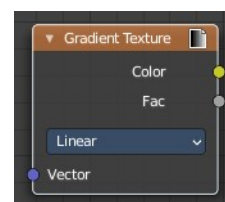
## **Outputs**

### ***Color***

RGB color from the image.

## **Gradient Texture**

The Gradient Texture node generates interpolated color and intensity values based on the input vector.



## **Inputs**

### ***Vector***

Texture coordinate to sample texture at; defaults to Generated texture coordinates if the socket is left unconnected.

## **Properties**

### ***Gradient Type***

Controls the type of gradient generated.

#### **Linear**

Directly returns the input X coordinate.

#### **Quadratic**

Interpolates the input X coordinate quadratically.

#### **Easing**

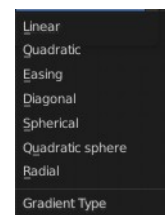
Uses a combination of quadratic and linear interpolation to return a smooth gradient from the input X coordinate.

#### **Diagonal**

Averages the input X and Y coordinates.

#### **Spherical**

Creates an inverse gradient using the length of the input vector; the maximum value is at (0, 0, 0).



## Quadratic Sphere

The same as Spherical, except interpolated quadratically.

## Radial

Returns a value based on the angle of the input around the Z axis.

## Outputs

### *Color*

Texture color output.

### *Factor*

Texture intensity output.

---

## IES Texture

The IES Texture is used to match real world lights based on IES files. IES files store the directional intensity distribution of light sources.

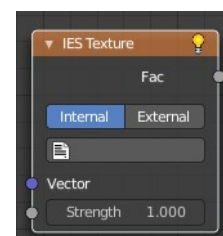
## Inputs

### *Vector*

Texture coordinate for lookup in the light distribution. Defaults to the normal.

### *Strength*

Light strength multiplier.



## Properties

### *Mode*

#### **Internal**

Use IES profile from a file embedded in a text data-block in the blend-file, for easy distribution.

#### **External**

Load IES profile from a file on the drive.

## Outputs

### *Factor*

Light intensity, typically plugged into the Strength input of an Emission node.

---

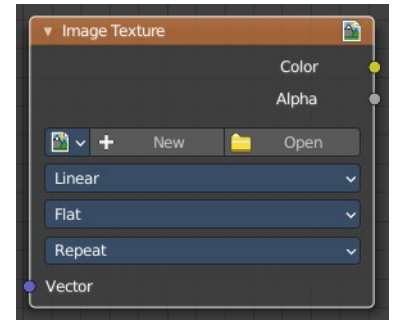
## Image Texture

The Image Texture is used to add an image file as a texture.

### Inputs

#### Vector

Texture coordinate for texture look-up. If this socket is left unconnected, UV coordinates from the active UV render layer are used.

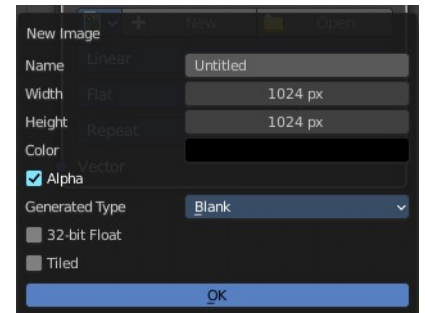


### Properties

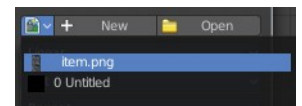
#### Image

Open an image, choose an existing image, or generate a new image.

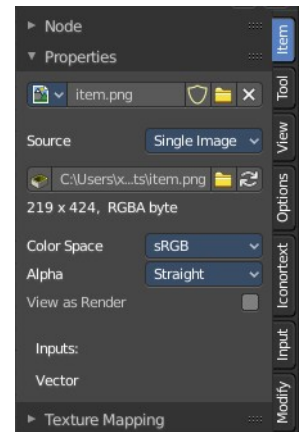
When you click at Open then a file browser opens up. When you click at New then a popup dialog opens. up where you can create a new image.



The image browser at the left allows you to pick an already existing texture from the scene.



More image settings can be found in the Sidebar in the Items tab. Usually you find in the Item tab the very same settings like in the selected node. But the Image texture node is an exception. It shows here the usual image related settings too. This will be explained in the sidebar chapter.



### Interpolation

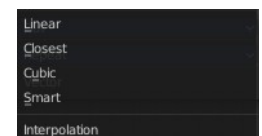
Method to scale images up or down for rendering.

#### Linear

Regular quality interpolation.

#### Cubic

Smoother, better quality interpolation. For bump maps this should be used to get best results.



## **Closest**

No interpolation, use only closest pixel for rendering pixel art.

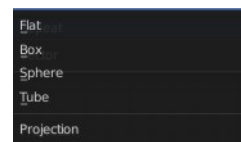
## **Smart**

### **Cycles Only**

Only for Open Shading Language. Use cubic interpolation when scaling up and linear when scaling down, for a better performance and sharpness.

## ***Projection***

Projection to use for mapping the textures.



## **Flat**

Uses the XY coordinates for mapping.

## **Box**

Maps the image to the six sides of a virtual box, based on the normal, using XY, YZ and XYZ coordinates depending on the side.

## **Blend**

For Box mapping, the amount to blend between sides of the box, to get rid of sharp transitions between the different sides. Blending is useful to map a procedural-like image texture pattern seamlessly on a model. 0.0 gives no blending; higher values give a smoother transition.

## **Sphere**

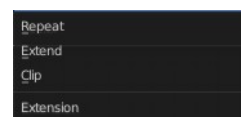
Sphere mapping is the best type for mapping a sphere, and it is perfect for making planets and similar objects. It is often very useful for creating organic objects.

## **Tube**

Maps the texture around an object like a label on a bottle. The texture is therefore more stretched on the cylinder. This mapping is of course very good for making the label on a bottle, or assigning stickers to rounded objects. However, this is not a cylindrical mapping so the ends of the cylinder are undefined.

## ***Extension***

Extension defines how the image is extrapolated past the original bounds:



### **Repeat**

Will repeat the image horizontally and vertically giving tiled-looking result.

### **Extend**

Will extend the image by repeating pixels on its edges.

### **Clip**

Clip to the original image size and set all the exterior pixels values to transparent black.

## Outputs

### **Color**

RGB color from image. If the image has alpha, the color is premultiplied with alpha if the Alpha output is used, and unpremultiplied or straight if the Alpha output is not used.

### **Alpha**

Alpha channel from image.

---

## Magic Texture

The Magic Texture node is used to add a procedural psychedelic color texture.

### Inputs

#### **Vector**

Texture coordinate to sample texture at; defaults to Generated texture coordinates if the socket is left unconnected.

#### **Scale**

Scale of the texture.

#### **Distortion**

Amount of distortion.

### Properties

#### **Depth**

Number of iterations.

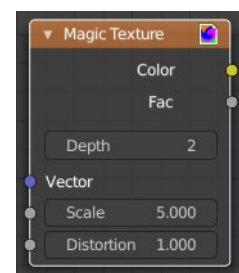
### Outputs

#### **Color**

Texture color output.

#### **Factor**

Texture intensity output.

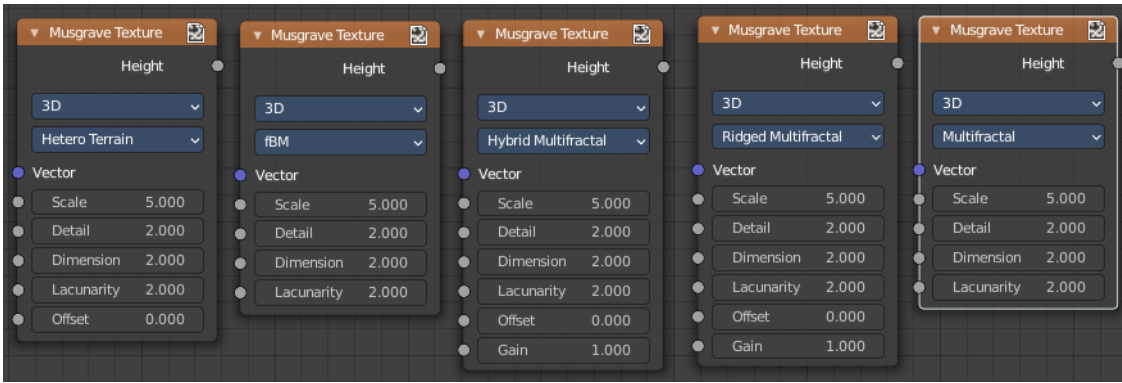


## Musgrave Texture

The Musgrave Texture node evaluates a fractal Perlin noise at the input texture coordinates. Compared to the noise texture, the Musgrave Texture allows greater control over how octaves are combined.

### Inputs

The available input types change, dependent of the chosen Type and dimensions.



### Vector

Texture coordinate to evaluate the noise at. Defaults to Generated texture coordinates if the socket is left unconnected.

### W

Texture coordinate to evaluate the noise at. Appears with 4 dimensions.

### Scale

Scale of the base noise octave.

### Detail

Number of noise octaves. The fractional part of the input is multiplied by the magnitude of the highest octave. Higher number of octaves corresponds to a higher render time.

### Dimension

The difference between the magnitude of each two consecutive octaves. Larger values corresponds to smaller magnitudes for higher octaves.

### Lacunarity

The difference between the scale of each two consecutive octaves. Larger values corresponds to larger scale for higher octaves.

### Offset

An added offset to each octave, determines the level where the highest octave will appear.

## ***Gain***

An extra multiplier to tune the magnitude of octaves.

## **Properties**

### ***Dimensions***

The dimensions of the space to evaluate the noise in.



#### **1D**

Evaluate the noise in 1D space at the input *W*.

#### **2D**

Evaluate the noise in 2D space at the input Vector. The Z component is ignored.

#### **3D**

Evaluate the noise in 3D space at the input Vector.

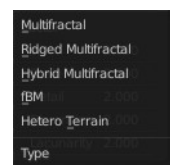
#### **4D**

Evaluate the noise in 4D space at the input Vector and the input *W* as the fourth dimension.

Higher dimensions corresponds to higher render time, so lower dimensions should be used unless higher dimensions are necessary.

## ***Type***

Type of the Musgrave texture.



### **fBM (fractal Brownian Motion)**

Produces an unnatural homogeneous and isotropic result. Uses an additive cascade, the values are simply added together.

### **Multifractal**

The result is more uneven (varies with location), more similar to a real terrain. Uses a multiplicative cascade.

### **Hybrid Multifractal**

Creates peaks and valleys with different roughness values, like real mountains rise out of flat plains. Combines the additive cascade with a multiplicative cascade.

### **Ridged Multifractal**

Creates sharp peaks. Calculates the absolute value of the noise, creating “canyons”, and then flips the surface upside down.

### **Hetero Terrain (Heterogeneous Terrain)**

Similar to Hybrid Multifractal creates a heterogeneous terrain, but with the likeness of river channels.



## Outputs

### **Factor**

Texture value.

## Noise Texture

The Noise Texture node evaluates a fractal Perlin noise at the input texture coordinates.

### Inputs

The inputs are dynamic, they become available if needed depending on the node properties.

### **Vector**

Texture coordinate to evaluate the noise at; defaults to Generated texture coordinates if the socket is left unconnected.

### **W**

Texture coordinate to evaluate the noise at.

### **Scale**

Scale of the base noise octave.

### **Detail**

Number of noise octaves. The fractional part of the input is multiplied by the magnitude of the highest octave. Higher number of octaves corresponds to a higher render time.

### **Roughness**

Adds a roughness noise.

### **Distortion**

Amount of distortion.

## Properties

### **Dimensions**

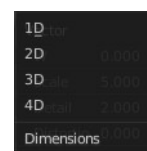
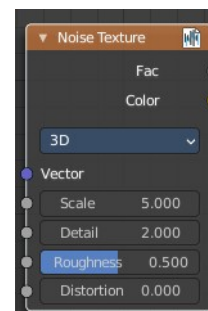
The dimensions of the space to evaluate the noise in.

#### **1D**

Evaluate the noise in 1D space at the input W.

#### **2D**

Evaluate the noise in 2D space at the input Vector. The Z component is ignored.



### 3D

Evaluate the noise in 3D space at the input Vector.

### 4D

Evaluate the noise in 4D space at the input Vector and the input W as the fourth dimension.

## Outputs

### **Factor**

Value of fractal noise.

### **Color**

Color with different fractal noise in each component.

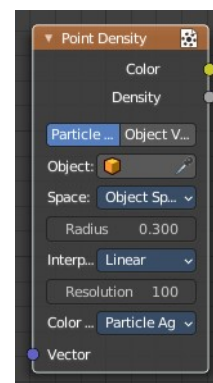
## Point Density

The Point Density node is used to add volumetric points for each particle or vertex of another object.

## Inputs

### **Vector**

Texture coordinate to sample texture at; defaults to global position (Position output of Geometry node) if the socket is left unconnected.



## Properties

### **Point Data**

Where to get points from.

### **Particle System**

Use each particle position from the specified particle system.

### **Object Vertices**

Use each vertex position from the specified object.

### **Object**

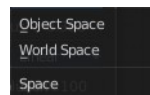
Which object's vertices or particle system will be used.

### **Particle System**

Particle positions from this system will be used.

## **Space**

The coordinate system for mapping points.



## **World Space**

Map each point exactly where the source particle/vertex is.

## **Object Space**

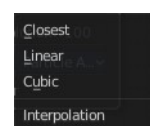
Fit the points from the source particles/vertices inside the bounding box of the object with the point density texture.

## **Radius**

Size of the points.

## **Interpolation**

Texel filtering type.



## **Closest**

No interpolation, use nearest texel. Produces blocky looking points.

## **Linear**

Interpolate linearly between texels, producing soft, round points.

## **Cubic**

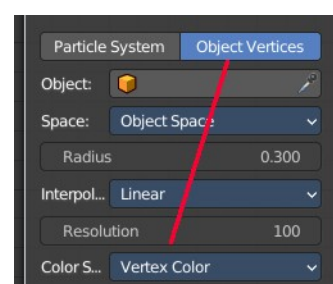
Use cubic falloff, producing very soft points. Useful when points are very densely packed.

## **Resolution**

The dimensions of the texture holding the point data.

## **Color Source**

Which attribute of the particle system or mesh is used to color the output. Switch to Object vertices to show the Vertex color sources.



## **Particle Color Sources**

### **Particle Age**

Lifetime mapped as (0.0 - 1.0) intensity.

### **Particle Speed**

Particle speed (absolute magnitude of velocity) mapped as (0.0 - 1.0) intensity.

### **Particle Velocity**

XYZ velocity mapped to RGB colors.

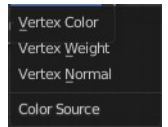


## Vertex Color Sources

### Vertex Color

Use a vertex color layer for coloring the point density texture.

Note. Vertex colors are defined per face corner. A single vertex can have as many different colors as faces it is part of. The actual color of the point density texture is averaged from all vertex corners.



### Vertex Weight

Use weights from a vertex group as intensity values.

### Vertex Normals

Use object-space vertex normals as RGB values.

## Outputs

### Color

Texture color output.

### Density

Density of volume.

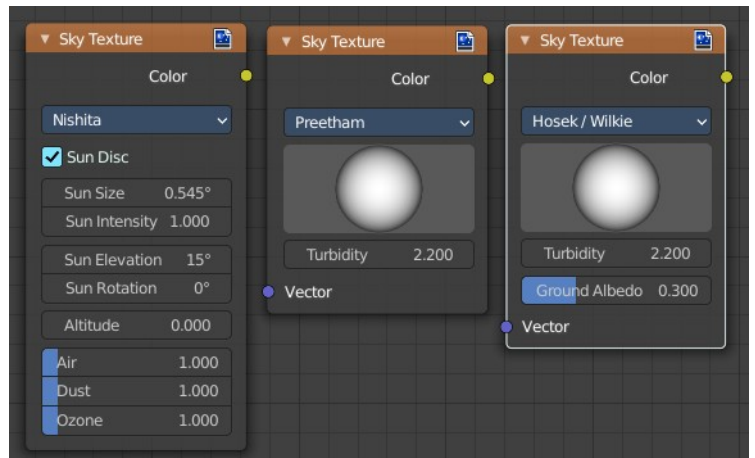
## Sky Texture

The Sky Texture node adds a procedural Sky texture.

### Inputs

#### Vector

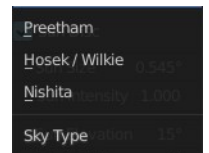
Texture coordinate to sample texture at; defaults to Generated texture coordinates if the socket is left unconnected.



## Properties

### Sky Type

Sky model to use. You have the choice between three methods. Nishita, Preetham and Hosek/Wilkie



### Sun Direction

Sun direction vector. Click at the image and drag to change the sun direction.



### Turbidity

Atmospheric turbidity. Some reference values:

2: Arctic like

3: clear sky

6: warm/moist day

10: hazy day

## **Ground Albedo**

Amount of light reflected from the planet surface back into the atmosphere. (RGB(0, 0, 0) is black, RGB(1, 1, 1) is white.)

## **Outputs**

### **Color**

Texture color output.

---

## **Voronoi Texture**

The Voronoi Texture node evaluates a Worley Noise at the input texture coordinates.

### **Inputs**

The inputs are dynamic, they become available if needed depending on the node properties.

### **Vector**

Texture coordinate to evaluate the noise at; defaults to Generated texture coordinates if the socket is left unconnected.

### **W**

Texture coordinate to evaluate the noise at.

### **Scale**

Scale of the noise.

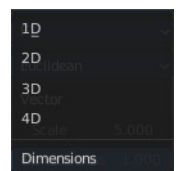
### **Randomness**

The randomness of the noise.

## **Properties**

### **Dimensions**

The dimensions of the space to evaluate the noise in.



## 1D

Evaluate the noise in 1D space at the input W.

## 2D

Evaluate the noise in 2D space at the input Vector. The Z component is ignored.

## 3D

Evaluate the noise in 3D space at the input Vector.

## 4D

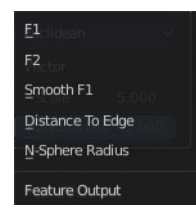
Evaluate the noise in 4D space at the input Vector and the input W as the fourth dimension.

## Feature

The Voronoi feature that the node will compute and return.

### F1

Compute and return the distance to the closest feature point as well as its position and color.



### Smooth F1

Compute and return a smooth version of F1.

### Distance To Edge

Compute and return the distance to the edges of the Voronoi cells.

### N-Sphere Radius

Compute and return the radius of the n-sphere inscribed in the Voronoi cells. In other words, it is half the distance between the closest feature point and the feature point closest to it.

## Distance Metric

The distance metric used to compute the texture.

### Euclidean

Use the Euclidean distance metric.

### Manhattan

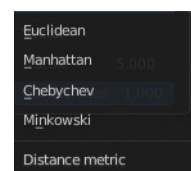
Use the Manhattan distance metric.

### Chebychev

Use the Chebychev distance metric.

### Minkowski

Use the Minkowski distance metric. The Minkowski distance is a generalization of the aforementioned metrics with an Exponent as a parameter. Minkowski with an exponent of one is equivalent to the Manhattan distance metric. Minkowski with an exponent of two is equivalent to the Euclidean distance metric. Minkowski with an infinite exponent is equivalent to the Chebychev distance metric.



## Outputs

### ***Distance***

The Distance.

### ***Color***

Cell color. The color is arbitrary.

### ***Position***

Position of feature point.

### ***W***

Position of feature point.

### ***Radius***

N-Sphere radius.

Note. In some configurations of the node, especially for low values of Randomness, rendering artifacts may occur. This happens due to the same reasons described in the Notes section in the White Noise Texture page and can be fixed in a similar manner as described there.

---

## Wave Texture

The Wave Texture node adds procedural bands or rings with noise distortion.

## Inputs

### ***Vector***

Texture coordinate to sample texture at; defaults to Generated texture coordinates if the socket is left unconnected.

### ***Scale***

Overall texture scale.

### ***Distortion***

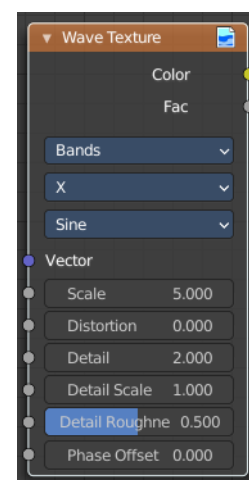
Amount of distortion of the wave (similar to the Marble texture in Blender Internal).

### ***Detail***

Amount of distortion noise detail.

### ***Detail Scale***

Scale of distortion noise.



## ***Detail Roughness***

Adds a roughness noise.

## ***Phase Offset***

Set an offset for the phase.

## **Properties**

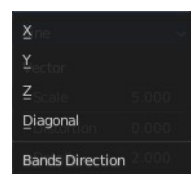
### ***Wave Type***

Bands or Rings shaped waves.



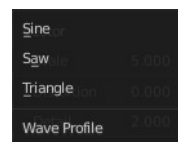
### ***Bands direction***

In which direction the bands should point.



### ***Wave Profile***

Controls the shape and look of the wave type.



### ***Saw***

Uses a saw tooth profile.

### ***Sine***

Uses the standard sine profile.

### ***Triangle***

Uses a triangle shape.

## **Outputs**

### ***Color***

Texture color output.

### ***Factor***

Texture intensity output.



## White Noise Texture

This node adds noise.

### Inputs

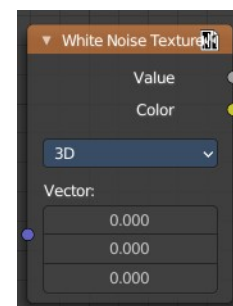
The inputs are dynamic, they become available if needed depending on the node properties.

#### **Vector**

Vector used as seed in 2D, 3D, and 4D dimensions.

#### **W**

Value used as seed in 1D and 4D dimensions.



## Properties

### **Dimensions**

The dimensions of the space to evaluate the noise in.

#### **1D**

The W input is used as seed.

#### **2D**

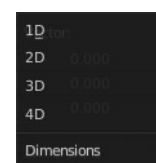
The X and Y components of the Vector input are used as seed.

#### **3D**

The Vector input is used as seed.

#### **4D**

Both the Vector input and the W input are used as seed.



## Outputs

### **Value**

Output random value.

Note! The slightest difference in seed values would result in completely different outputs. Consequently, bad precision may have significant impact on the output. Usually, we can mitigate this issue by:

Eliminating the problematic seed value. If the problematic seed value is constant, it should be eliminated by choosing a lower dimension or multiplying it by zero.

Adding an arbitrary value to the seed. The issue might only happen at certain boundaries, like unit boundaries, so simply adding an arbitrary value might solve the issue.

Taking the absolute value of the seed. In computing, zero may be positive or negative, so taking the absolute values unifies the zero into a single value.