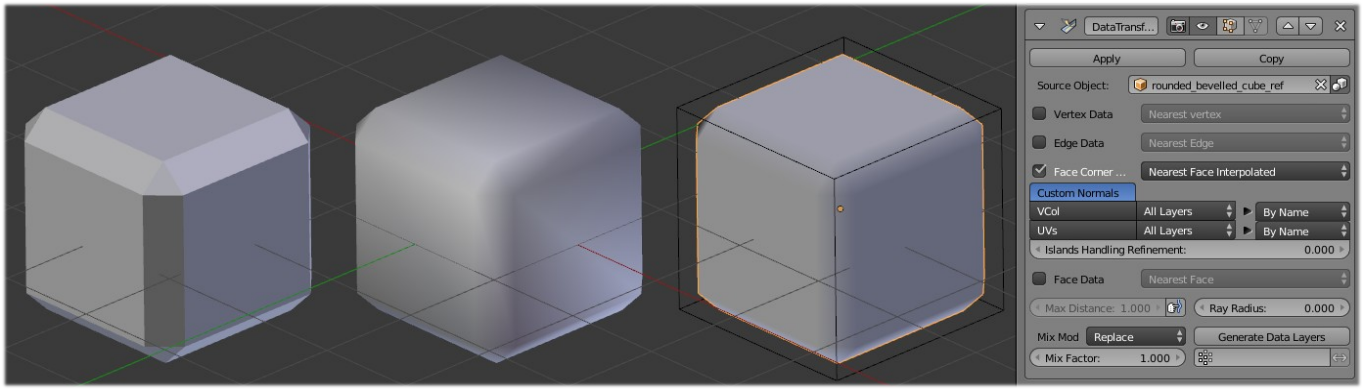
5.8.3 Modeling - Modifiers - Modify

Data Transfer Modifier

The **Data Transfer** modifier transfers several types of data from one mesh to another. Data types include vertex groups, UV layers, vertex colors, custom normals...

Transfer works by generating a mapping between source mesh’s items (vertices, edges, etc.) and destination ones, either on a one-to-one basis, or mapping several source items to a single destination one - interpolated mapping.



From left to right, a flat-shaded beveled cube, a smooth-shaded beveled cube, and an autosmooth-shaded beveled cube copying its normals from the reference, flat-shaded cube shown as wire here, to achieve the ‘fake round corners’ effect.

Options

Data Transfer modifier.

Source Object

Mesh object to copy data from.

If the button to the right of the field is unset, source and destination geometries are considered in global space when generating the mapping, otherwise they are evaluated in local space (i.e. as if both object's

centers were at the same place).

Max Distance

When the icon “finger” button to the right is enabled, this is the maximum distance between source and destination to get a successful mapping. If a destination item cannot find a source one withing that range, then it will get no transfered data.

This allows to transfer a small sub-detailed mesh onto a more complete one (e.g. from a “hand” mesh towards a “full body” one).

Ray Radius
For ray-casting-based mapping methods, the radius of the cast rays. Especially important for 1D and 2D items (i.e. vertices and edges), without some width there would be nearly no ray-casting matches...

Mix Mode

Controls how destination data are affected:

All
Replaces everything in destination (note that *Mix Factor* is still used).
Above Threshold
Only replaces destination value if it’s above given threshold (*Mix Factor*). How that threshold is interpreted depends on data type, note that for boolean values this option fakes a logical AND.
Below Threshold
Only replaces destination value if it’s below given threshold (*Mix Factor*). How that threshold is interpreted depends on data type, note that for boolean values this option fakes a logical OR.
Mix, Add, Subtract, Multiply
Apply that operation, using mix factor to control how much of source or destination value to use. Only available for a few types (vertex groups, vertex colors).

Mix Factor
How much of the transfered data gets mixed into existing one (not supported by all data types).
Vertex Group
Allows per-item fine control of the mix factor. Vertex group influence can be reverted using the small “arrow” button to the right.
Generate Data Layers
This modifier cannot generate needed data layers itself. Once the set of source data to transfer is selected, this button shall be used to generate matching destination layers.

Selection of Data to Transfer

To keep the size of the modifier reasonable, the kind of items to be affected must be selected first (vertices, edges, face corners and/or faces).

Mapping Type
How is generated the mapping between those source and destination items. Each type has its own options, see Geometry Mapping below for details.
Data Types
The left column of toggle buttons, to select which data types to transfer.
Multi-layers Data Types Options
In those cases (vertex groups, vertex colors, UVs), one can select which source layers to transfer (usually, either all of them, or a single specified one), and how to affect destination (either by matching names, matching order/position, or, if a single source is selected, by specifying manually destination layer).
Islands Handling Refinement

This setting only affects UV transfer currently. It allows to avoid a given destination face to get UV coordinates from different source UV islands. Keeping it at 0.0 means no island handling at all. Typically, small values like 0.02 are enough to get good results, but if you are mapping from a very high poly source towards a very low poly destination, you may have to raise it quite significantly.

Usage

First key thing to keep in mind when using this modifier is that **it will not create destination data layers**. *Generate Data Layers* button shall always be used for this purpose, once set of source data to transfer is selected. It should also be well understood that creating those data layers on destination mesh is **not** part of the modifier stack, which means e.g. that they will remain even once the modifier is deleted, or if source data selection is modified.

Geometry Mapping

Geometry mapping is the process by which a given destination vertex/edge/... knows **which part** of the source mesh to get its data from. It is crucial to understand this topic well to get good results with this modifier.

Topology
: The simplest option, expects both meshes to have identical number of items, and match them by order (indices). Useful e.g. between meshes that were identical copies, and got deformed differently.

One-To-One Mappings

Those always select only one source item for each destination one, often based on shortest distance.

Vertices

Nearest Vertex
: Uses source's nearest vertex.

Nearest Edge Vertex
: Uses source's nearest vertex of source's nearest edge.

Nearest Face Vertex
: Uses source's nearest vertex of source's nearest face.

Edges

Nearest Vertices
: Uses source's edge which vertices are nearest from destination edge's vertices.

Nearest Edge
: Uses source's nearest edge (using edge's midpoints).

Nearest Face Edge
: Uses source's nearest edge of source's nearest face (using edge's midpoints).

Face Corners

A face corner is not a real item by itself, it's some kind of split vertex attached to a specific face. Hence both vertex (location) and face (normal, ...) aspects are used to match them together.

Nearest Corner and Best Matching Normal
: Uses source's corner having the most similar **split** normal with destination one, from those sharing the nearest source's vertex.

Nearest Corner and Best Matching Face Normal
: Uses source's corner having the most similar **face** normal with destination one, from those sharing the nearest source's vertex.

Nearest Corner of Nearest Face
: Uses source's nearest corner of source's nearest face.

Faces

Nearest Face

Uses source's nearest face.

Best Normal-Matching:

Uses source's face which normal is most similar with destination one.

Interpolated Mappings

Those use several source items for each destination one, interpolating their data during the transfer.

Vertices

Nearest Edge Interpolated

Uses nearest point on nearest source's edge, interpolates data from both source edge's vertices.

Nearest Face Interpolated

Uses nearest point on nearest source's face, interpolates data from all that source face's vertices.

Projected Face Interpolated

Uses point of face on source hit by projection of destination vertex along its own normal, interpolates data from all that source face's vertices.

Edges

Projected Edge Interpolated

This is a sampling process. Several rays are cast from along the destination's edge (interpolating both edge's vertex normals), and if enough of them hit a source's edge, all hit source edges' data are interpolated into destination one.

Face Corners

A face corner is not a real item by itself, it's some kind of split vertex attached to a specific face. Hence both vertex (location) and face (normal, ...) aspects are used to match them together.

Nearest Face Interpolated

Uses nearest point of nearest source's face, interpolates data from all that source face's corners.

Projected Face Interpolated

Uses point of face on source hit by projection of destination corner along its own normal, interpolates data from all that source face's corners.

Faces

Projected Face Interpolated

This is a sampling process. Several rays are cast from the whole destination's face (along its own normal), and if enough of them hit a source's face, all hit source faces' data are interpolated into destination one.

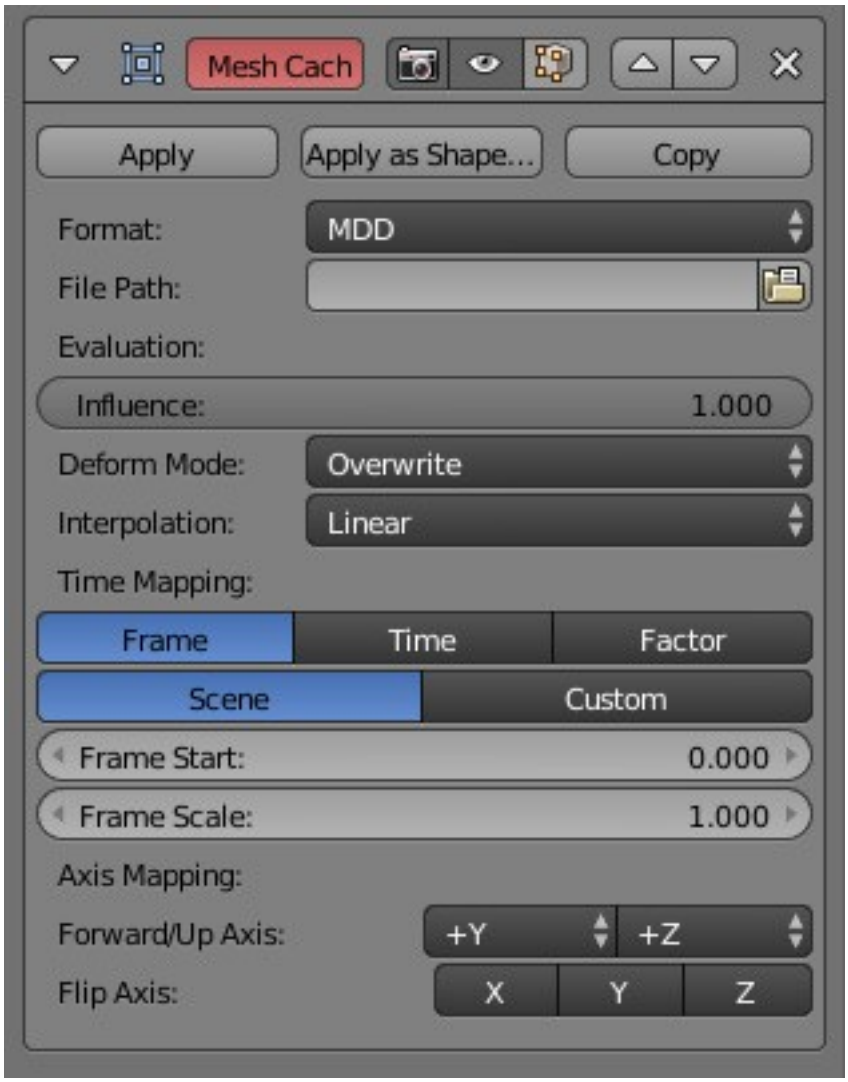
Mesh Cache Modifier

The **Mesh Cache** modifier is used so animated mesh data can be applied to a mesh and played back, deforming the mesh.

This works in a similar way to shape-keys but is aimed at playing back external files and is often used for interchange between applications.

When using this modifier, the vertex locations are overwritten.

Options



Mesh Cache modifier

Format
The input file format (currently **MDD** and **PC2** are supported).

File Path
Path to the cache file.

Evaluation:

Influence
Factor to adjust the influence of the modifiers deformation, useful for blending in/out from the cache data.

Deform Mode

This setting defaults to ‘Overwrite’ which will replace the vertex locations with those in the cache file. However you may want to use shape-keys, for example, and mix them with the mesh-cache. In this case you can select the ‘Deform’ option which integrates deformations with the mesh-cache result.

Note
This feature is limited to making smaller, isolated edits and won’t work for larger changes such as re-

posing limbs

Interpolation
None or Linear which will blend between frames; use linear when the frames in the cache file don’t match up exactly with the frames in the blend file.

Time Mapping:

Time Mode

Select how time is calculated.

Frame

Allows you to control the frames, which will ignore timing data in the file but is often useful since it gives simple control.

Time

Evaluates time in seconds, taking into account timing information from the file (offset and frame-times).

Factor

Evaluates the entire animation as a value from [0 - 1].

Play Mode

Select how playback operates.

Scene

Use the current frame from the scene to control playback.

Frame Start

Play the cache starting from this frame.

Frame Scale

Scale time by this factor (applied after the start value).

Custom

Control animation timing manually.

Evaluation Value

Property used for animation time, this gives more control of timing - typically this value will be animated.

Axis Mapping:

Forward/Up Axis

The axis for forward and up used in the source file. *Often different applications have different axis defaults for up/down front/back, so it's common to have to switch these on import.*

Flip Axis

In rare cases you may also need to flip the coordinates on an axis.

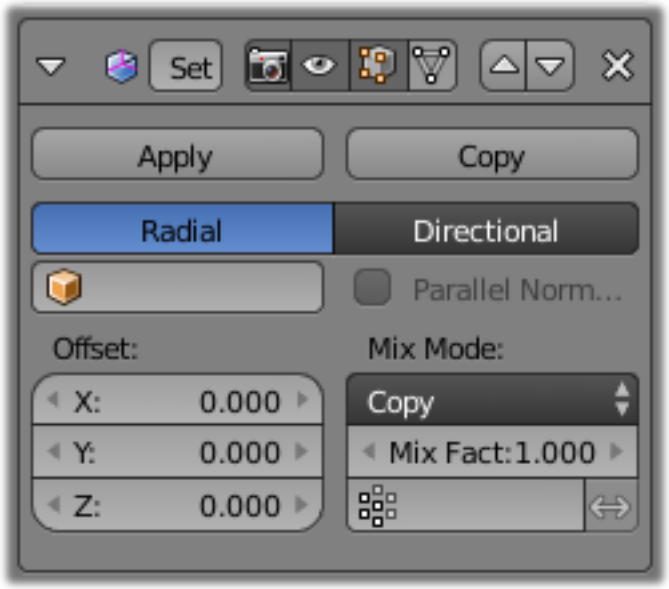
Hints

- Both MDD and PC2 depend on the vertex order on the mesh remaining unchanged; this is a limitation with the method used so take care not to add/remove vertices once this modifier is used.

Normal Edit Modifier

The **Normal Edit** modifier affects (or generates) custom normals. It uses a few simple parametric methods to compute normals (quite useful in game development and architecture areas), and mixes back those generated normals with existing ones.

Options



Normal Edit modifier.

Radial/Directional

> The two modes currently available to generate normals.
>
> Radial aligns normals with the (origin, vertex coordinates) vector, in other words all normals seems to radiate from the given center point, as if they were emitted from an ellipsoid surface.
>
> Directional makes all normals point (converge) towards a given target object.

Target Object

> Uses this object's center as reference point when generating normals.
>
> Optional in *Radial* mode, mandatory in *Directional* one.

Parallel Normals

> Makes all normals parallel to the line between both objects' centers, instead of converging towards target's center.
>
> Only relevant in *Directional* mode

Offset

> Gives modified object's center an offset before using it to generate normals.
>
> Only relevant in *Radial* mode if no *Target Object* is set, and in *Directional* mode when *Parallel Normals* is set.

Mix Mode

How to affect existing normals with newly generated ones.

Note the *Multiply* option is **not** a cross product, but a mere component-by-component multiplication.

Mix Factor
How much of the generated normals get mixed into existing ones.

Vertex Group
Allows per-item fine control of the mix factor. Vertex group influence can be reverted using the small “arrow” button to the right.

Usage

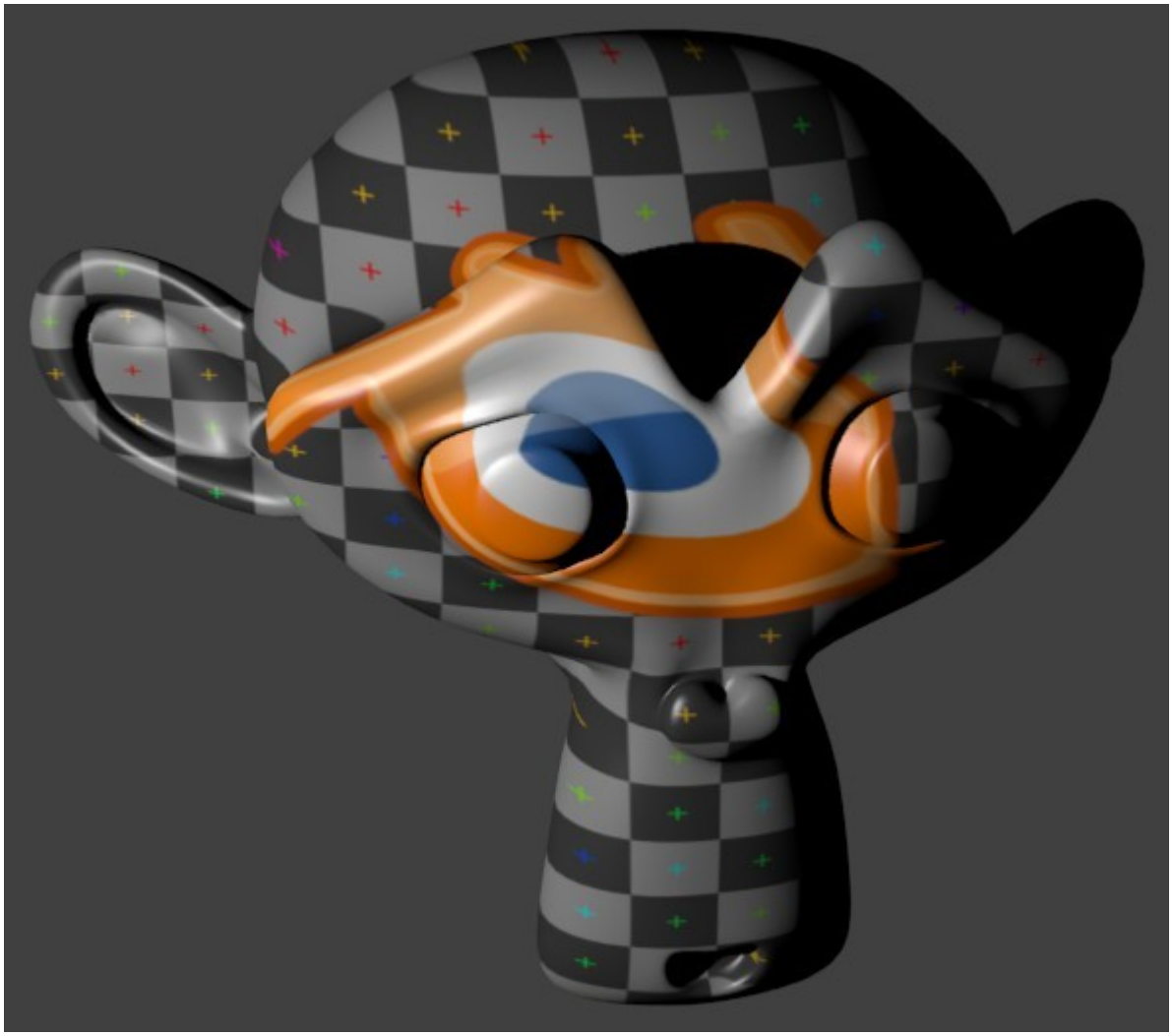
This modifier can be used to quickly generate radial normals for low-poly tree foliage, or “fix” shading of toon-like rendering by partially bending default normals...

The only mandatory prerequisite to use it is to enable *Auto Smooth* option in Mesh properties, *Normals* panel.

Tip

More complex normal manipulations can be achieved by copying normals from one mesh to another, see the *Data Transfer modifier.*

UV Project Modifier

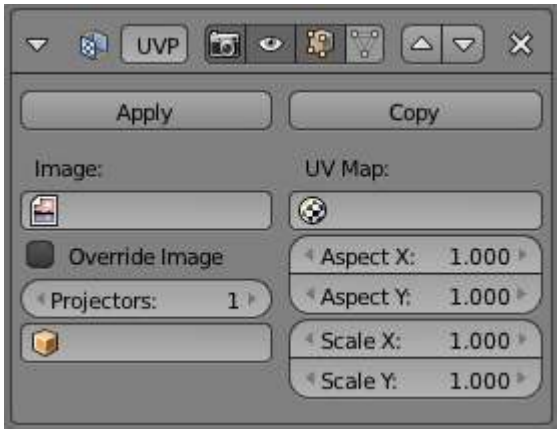


Projecting the Blender logo onto Suzanne.

The **UV Project** Modifier acts like a slide projector. It emits a UV map from the negative Z-axis of a controller object (such as an *empty*), and applies it to the object as the "light" hits it. It can optionally override the objects face texture.

Download an example

Options



UV layer
: Which UV layer to modify. Defaults to the active rendering layer.

Image
: The image associated with this modifier. Not required; you can just project a UV for use elsewhere. *Override Image*, below, defines how the image is used.

Override Image

- When true, the Face Texture of all vertices on the mesh is replaced with the Image. This will cause the image to repeat, which is usually undesirable.
- When false, the modifier is limited to faces with the Image as their Face Texture.

Projectors

Up to ten projector objects are supported. Each face will choose the closest and aligned projector with it's surface normal. Projections emit from the negative Z-axis (i.e. straight down a camera or lamp). If the projector is a camera, the projection will adhere to its perspective/orthographic setting.

Objects

Specify the projector Object

Aspect X/Y and Scale X/Y

These allow simple manipulation of the image. Only apply when a camera is used as projector Object.

Usage

General

UV Project is great for making spotlights more diverse, and also for creating decals to break up repetition.

The modifier's Image property is not generally used - instead, a texture mapped to the UV layer that the modifier targets is added to the object's Material. This allows you to prevent the image from repeating by setting *Texture –> Image Mapping –> Extension* to *Clip*.

Perspective Cameras

When using perspective cameras or spot lamps, you will likely want to enable the **UV Project** Material Option (available in the materials panel), This uses a different UV interpolation to prevent distortion.

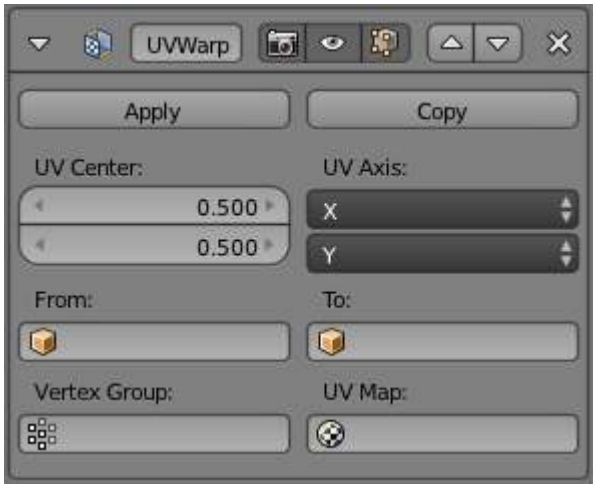
Note
This option is not yet available for Cycles

UV Warp Modifier

The **UV Warp** modifier uses two objects to define a transformation which is applied to the chosen UV coordinates.

Its purpose is to give you direct control over the object's UVs in the 3D View, allowing you to directly translate, rotate and scale existing UV coordinates using controller objects or bones.

Options



UV Center
: The center point of the UV map to use when applying scale or rotation. With (0, 0) at the bottom left and (1, 1) at the top right. Defaults to (0.5, 0.5).

UV Axis
: The axes to use when mapping the 3D coordinates into 2D.

From/To
: The two objects used to define the transformation. See *Usage* below.

Vertex Group
: The vertex group can be used to scale the influence of the transformation per-vertex.

UV Map
: Which UV map to modify. Defaults to the active rendering layer.

Usage

How the UVs are warped is determined by the difference between the transforms (location, rotation and scale) of the *from* and *to* objects.

If the *to* object has the same transforms as the *from* object, the UVs will not be changed.

Assuming the *UV Axis* of the modifier is X/Y and the scale of the objects are (1, 1, 1), if the *to* object is one unit away from the *from* object on the X-axis, the UVs will be transformed on the U-axis (horizontally) by one full UV space (the entire width of the image)

Vertex Weight Modifiers

The Vertex Weight modifiers work on a vertex group of the affected object, by modifying its weights and/or which vertices belong to the vertex group.

Warning

These modifiers do implicit clamping of weight values in the standard `[0.0, 1.0]` range. All values below `0.0` will be set to `0.0`, and all values above `1.0` will be set to `1.0`.

There are currently three Vertex Weight modifiers:

- Vertex Weight Edit Modifier

- Vertex Weight Mix Modifier
- Vertex Weight Proximity Modifier

Common Settings

The influence/masking part of Vertex Weight modifiers.

The three Vertex Weight modifiers share a few settings, controlling their influence on the affected vertex group.

Global Influence

The overall influence of the modifier (`0.0` will leave the vertex group's weights untouched, `1.0` is standard influence).

Warning

Influence only affects weights, adding/removing of vertices to/from vertex group is not prevented by setting this value to `0.0`.

Vertex Group Mask

An additional vertex group, the weights of which will be multiplied with the global influence value for each vertex. If a vertex is not in the masking vertex group, its weight will be not be affected.

Texture

An additional texture, the values of which will be multiplied with the global influence value for each vertex.

This is a standard texture *data-block* control. When set, it reveals other settings:

Texture Coordinates

How the texture is mapped to the mesh.

Local

Use local vertex coordinates.

Global
Use vertex coordinates in global space.

Object
Use vertex coordinates in another object's space.

UV
Use a UV layer's coordinates.

Use Channel

Which channel to use as weight factor source/

Red/Green/Blue/Alpha
One of the color channels' values.

Intensity
The average of the RGB channels (If RGB = `1.0, 0.0, 0.0`, value is `0.33`)

Value
The highest value of the RGB channels (If RGB = `1.0, 0.0, 0.0`, value is `1.0`)

Hue
Uses the hue value from the standard color wheel (e.g. blue has a higher hue value than yellow)

Saturation
Uses the saturation value (e.g. pure red's value is `1.0`, gray is `0.0`)

Note

All of the channels above are gamma corrected, except for *Intensity*.

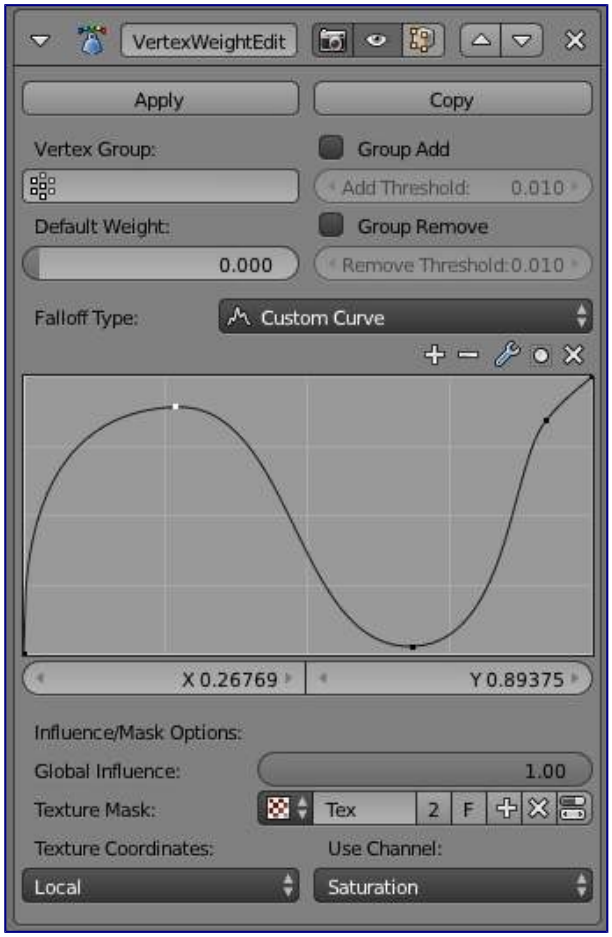
Object
The object to be used as reference for *Object* mapping.

UV Layer
The UV layer to be used for *UV* mapping.

Viewing Modified Weights

You can view the modified weights in *Weight Paint* mode. This also implies that you'll have to disable the Vertex Weight modifiers if you want to see the original weights of the vertex group you are editing.

Vertex Weight Edit Modifier



The Vertex Weight Edit modifier panel.

This modifier is intended to edit the weights of one vertex group.

The general process is the following, for each vertex:

- [Optional] It does the mapping, either through one of the predefined functions, or a custom mapping curve.
- It applies the influence factor, and optionally the vertex group or texture mask (`0.0` means original weight, `1.0` means fully mapped weight).
- It applies back the weight to the vertex, and/or it might optionally remove the vertex from the group if its weight is below a given threshold, or add it if it's above a given threshold.

Options

Vertex Group
: The vertex group to affect.

Default Weight
: The default weight to assign to all vertices not in the given vertex group.

Falloff Type

: Type of mapping:

  Linear
  : No mapping.

  Custom Curve
  : Allows the user to manually define the mapping using a curve.

  Sharp, Smooth, Root and Sphere
  : These are classical mapping functions, from spikiest to roundest.

Random
Uses a random value for each vertex.
Median Step
Creates binary weights (`0.0` or `1.0`), with `0.5` as cutting value.

Group Add
Adds vertices with a final weight over *Add Threshold* to the vertex group.

Group Remove
Removes vertices with a final weight below *Remove Threshold* from the vertex group.

Vertex Weight Mix Modifier

The Vertex Weight Mix modifier panel.

This modifier mixes a second vertex group (or a simple value) into the affected vertex group, using different operations.

Options

Vertex Group A
The vertex group to affect.

Default Weight A
The default weight to assign to all vertices not in the given vertex group.

Vertex Group B
The second vertex group to mix into the affected one. Leave it empty if you only want to mix in a simple value.

Default Weight B
The default weight to assign to all vertices not in the given second vertex group.

Mix Mode

How the vertex group weights are affected by the other vertex group's weights.

Replace weights
Replaces affected weights with the second group's weights.

Add to weights
Adds the values of *Group B* to *Group A*.

Subtract from weights
Subtracts the values of *Group B* from *Group A*.

Multiply weights
Multiplies the values of *Group B* with *Group A*.

Divide weights

Divides the values of *Group A* by *Group B*.

Difference
: Subtracts the smaller of the two values from the larger.

Average
: Adds the values together, then divides by 2.

Mix Set

Choose which vertices will be affected.

All vertices
: Affects all vertices, disregarding the vertex groups content.

Vertices from group A
: Affects only vertices belonging to the affected vertex group.

Vertices from group B
: Affects only vertices belonging to the second vertex group.

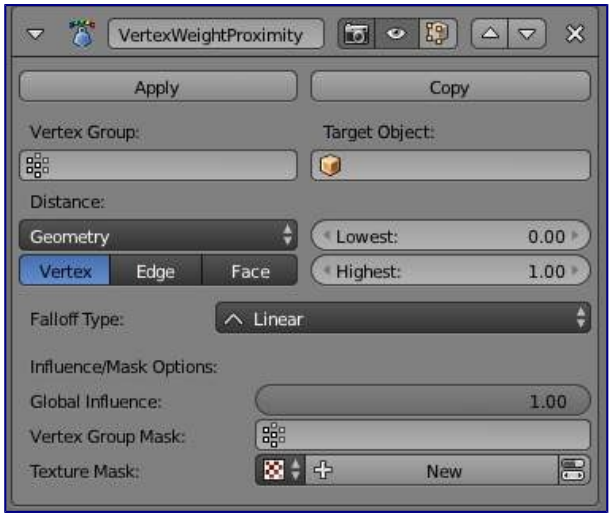
Vertices from one group
: Affects only vertices belonging to at least one of the vertex groups.

Vertices from both groups
: Affects only vertices belonging to both vertex groups.

Warning

When using *All vertices*, *Vertices from group B* or *Vertices from one group*, vertices might be added to the affected vertex group.

Vertex Weight Proximity Modifier



The Vertex Weight Proximity modifier panel.

This modifier sets the weights of the given vertex group, based on the distance between the object (or its vertices), and another target object (or its geometry).

Options

Vertex Group
: The vertex group to affect.

Target Object
: The object from which to compute distances.

Proximity mode

Object Distance
: Use the distance between the modified mesh object and the target object as weight for all vertices in the affected vertex group.

Geometry Distance
: Use the distance between each vertex and the target object, or its geometry.

The *Geometry Distance* mode has three additional options, *Vertex*, *Edge* and *Face*. If you enable more than one of them, the shortest distance will be used. If the target object has no geometry (e.g. an empty or camera), it will use the location of the object itself.

Vertex
: This will set each vertex's weight from its distance to the nearest vertex of the target object.

Edge
: This will set each vertex's weight from its distance to the nearest edge of the target object.

Face
: This will set each vertex's weight from its distance to the nearest face of the target object.

Lowest
: Distance mapping to `0.0` weight.

Highest
: Distance mapping to `1.0` weight.

Falloff Type
: Some predefined mapping functions, see Vertex Weight Edit Modifier.

Tip
Lowest can be set above *Highest* to reverse the mapping.

Examples

Using Distance from a Target Object

As a first example, let's dynamically control a *Wave* modifier with a modified vertex group.

Add a *Grid* mesh, with many vertices (e.g. a **100×100** vertices), and `10` BU side-length. Switch to *Edit* mode (`Tab`), and in the *Object Data* properties, *Vertex Groups* panel, add a vertex group. Assign to it all your mesh's vertices (with e.g. a `1.0` weight). Go back to *Object* mode.

Then, go to the *Modifiers* properties, and add a *Vertex Weight Proximity* modifier. Set the mode to *Object Distance*. Select your vertex group, and the target object you want (here I used the lamp).

You will likely have to adjust the linear mapping of the weights produced by the *Vertex Weight Proximity* modifier. To do so, edit *Lowest Dist* and *Highest Dist* so that the first corresponds to the distance between your target object and the vertices you want to have lowest weight, and similarly with the second and highest weight...

Now add a *Wave* modifier, set it to your liking, and use the same vertex group to control it.

Animate your target object, making it move over the grid. As you can see, the waves are only visible around the reference object! Note that you can insert a *Vertex Weight Edit* modifier before the *Wave* one, and use its *Custom Curve* mapping to get larger/narrower "wave influence's slopes".

https://vimeo.com/30187079

The Blender file, `TEST_1` scene.

Using Distance from a Target Object's Geometry

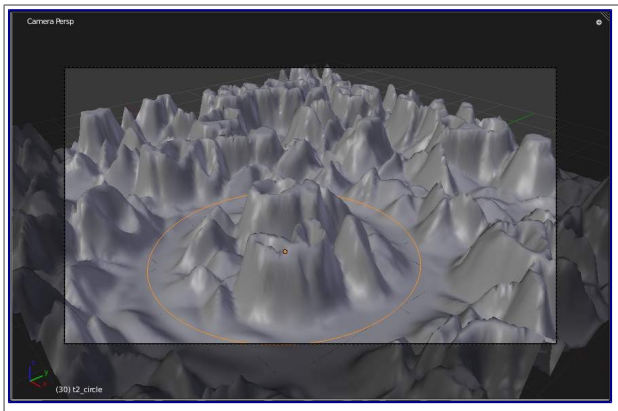
We're going to illustrate this with a *Displace* modifier.

Add a **10×10** BU **100×100** vertices grid, and in *Edit* mode, add to it a vertex group containing all of its vertices, as above. You can even further sub-divide it with a first *Subsurf* modifier.

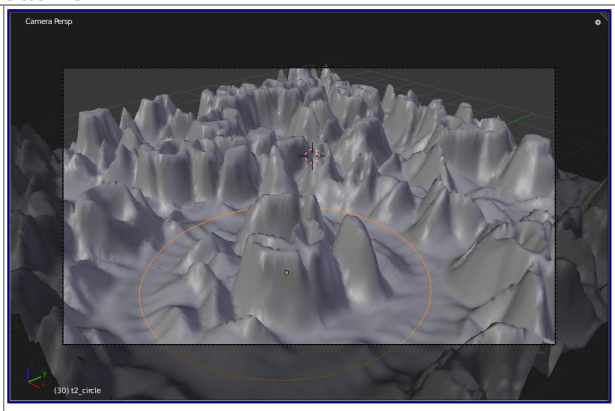
Now add a curve circle, and place it `0.25` BU above the grid. Scale it up a bit (e.g. `4.0`).

Back to the grid object, add to it a *Vertex Weight Proximity* modifier, in *Geometry Distance* mode. Enable *Edge* (if you use *Vertex* only, and your curve has a low U definition, you would get wavy patterns, see (*Wavy patterns*)).

Wavy patterns.



Distance from edges.



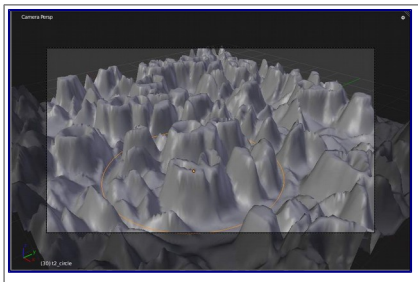
Distance from vertices.

Set the *Lowest Dist* to `0.2`, and the *Highest Dist* to `2.0`, to map back the computed distances into the regular weight range.

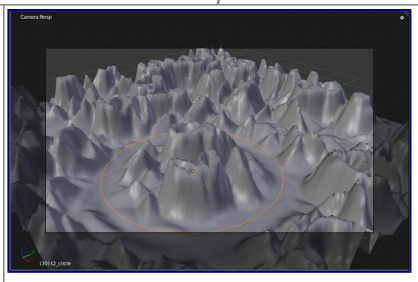
Add a third *Displace* modifier and affect it the texture you like. Now, we want the vertices of the grid nearest to the curve circle to remain undisplaced. As they will get weights near zero, this means that you have to set the *Midlevel* of the displace to `0.0`. Make it use our affected vertex group, and that's it! Your nice mountains just shrink to a flat plane near the curve circle.

As in the previous example, you can insert a *Vertex Weight Edit* modifier before the *Displace* one, and play with the *Custom Curve* mapping to get a larger/narrower "valley"...

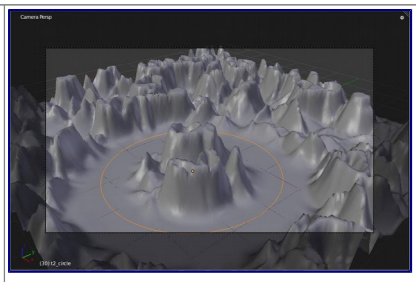
Curve Map variations.



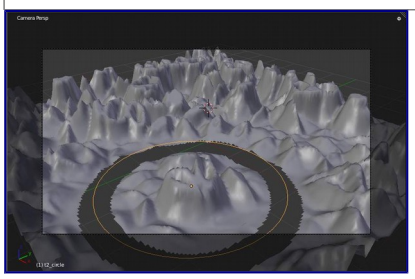
Concave-type mapping curve.



No mapping curve (linear).



Convex-type mapping curve.



Vertices with a computed weight below 0.1 removed from the vertex group.

You can also add a fifth *Mask* modifier, and enable *Vertex Weight Edit* ‘s *Group Remove* option, with a *Remove Threshold* of `0.1`, to see the bottom of your valley disappear.

https://vimeo.com/30188564

The Blender file, `TEST_2` scene.

Using a Texture and the Mapping Curve

Here we are going to create a sort of strange alien wave (yes, another example with the *Wave* modifier... but it’s a highly visual one; it’s easy to see the vertex group effects on it...).

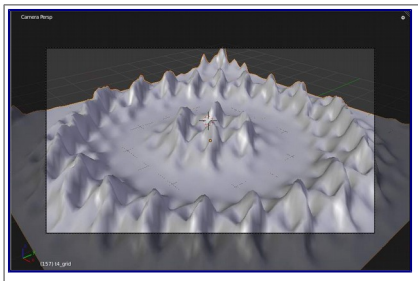
So as above, add a **100×100** grid. This time, add a vertex group, but without assigning any vertex to it - we’ll do this dynamically.

Add a first *Vertex Weight Mix* modifier, set the *Vertex Group A* field with a *Default Weight A* of `0.0`, and set *Default Weight B* to `1.0`. Leave the *Mix Mode* to *Replace weights*, and select *All vertices* as *Mix Set*. This way, all vertices are affected. As none are in the affected vertex group, they all have a default weight of `0.0`, which is replaced by the second default weight (`1.0`). And all those vertices are also added to the affected vertex group.

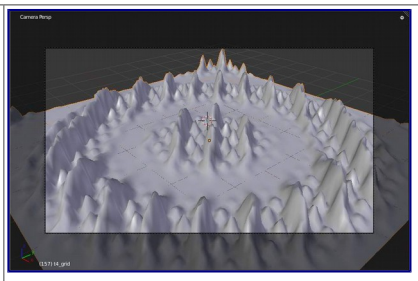
Now, select or create a masking texture - here I chose a default *Magic* one. The values of this texture will control how much of the “second weight” (`1.0`) replaces the “first weight” (`0.0`)... In other words, they are taken as weight values!

You can then select which texture coordinates and channel to use. Leave the mapping to the default *Local* option, and play with the various channels...

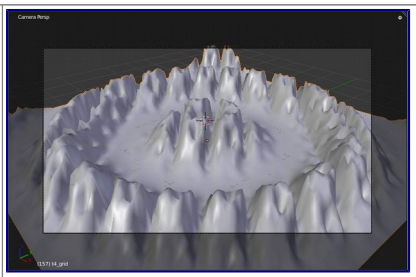
Texture channel variations.



Using intensity.



Using Red.



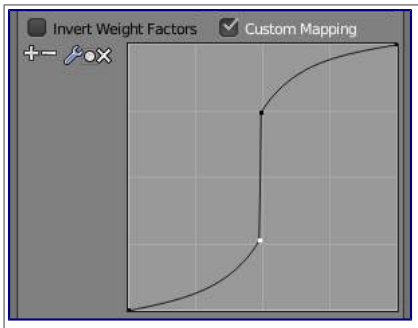
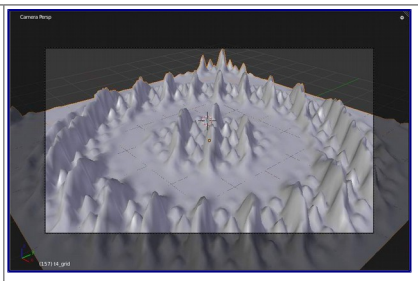
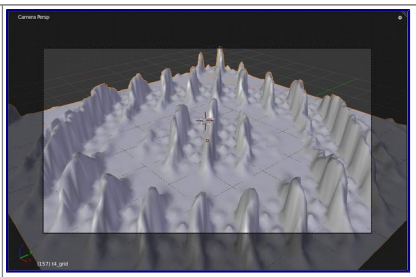
Using Saturation.

Don’t forget to add a *Wave* modifier, and select your vertex group in it!

You can use the weights created this way directly, but if you want to play with the curve mapping, you must add the famous *Vertex Weight Edit* modifier, and enable its *Custom Curve* mapping.

By default, it’s a one-to-one linear mapping - in other words, it does nothing! Change it to something like in (*A customized mapping curve*), which maps `[0.0, 0.5]` to `[0.0, 0.25]` and `[0.5, 1.0]` to `[0.75, 1.0]`, thus producing nearly only weights below `0.25`, and above `0.75` : this creates great “walls” in the waves...

Custom mapping curve.

 A customized mapping curve.	 Custom Mapping disabled.	 Custom Mapping enabled.

https://vimeo.com/30188814

The Blender file, TEST_4 scene.

See Also

- The Development page.