10.2.2.5 Render - Blender Render Engine - Textures - Texture Mapping

Texture Mapping

- Texture Mapping
  - Coordinates
  - Projection
  - Inheriting coordinates from the parent object
  - Coordinate Offset, Scaling and Transformation
- Environment Maps
  - Options
  - Examples
  - Limitations

Texture Mapping

Textures need mapping coordinates, to determine how they are applied to the object. The mapping specifies how the texture will ultimately wrap itself to the object.

For example, a 2D image texture could be configured to wrap itself around a cylindrical shaped object.

Coordinates

Mapping Coordinate menu

Coordinates
Mapping works by using a set of coordinates to guide the mapping process. These coordinates can come from anywhere, usually the object to which the texture is being applied to.

Global
The scene's global 3D coordinates. This is also useful for animations; if you move the object, the texture moves across it. It can be useful for letting objects appear or disappear at a certain position in space.

Object

Uses an object as source for coordinates. Often used with an *Empty*, this is an easy way to place a small image at a given point on the object. This object can also be animated, to move a texture around or through a surface.

Object
Select the name of an object.

Generated
The original undeformed coordinates of the object. This is the default option for mapping textures.

UV

UV mapping is a very precise way of mapping a 2D texture to a 3D surface. Each vertex of a mesh has its own UV co-ordinates which can be unwrapped and laid flat like a skin. You can almost think of UV coordinates as a mapping that works on a 2D plane with its own local coordinate system to the plane on which it is operating on. This mapping is especially useful when using 2D images as textures, as seen in UV Mapping. You can use multiple textures with one set of UV coordinates.

Layer
Select your UV layer to use it for mapping.

Strand/Particle
Uses normalized 1D strand texture coordinate or particle age(X) and trail position (Y). Use when texture is applied to hair strands or particles.

Window
The rendered image window coordinates. This is well suited to blending two objects.

Normal

Uses the direction of the surface's normal vector as coordinates. This is very useful when creating certain special effects that depend on viewing angle.

Reflection
Uses the direction of the reflection vector as coordinates. This is useful for adding reflection maps - you will need this input when Environment Mapping.

Stress
Uses the difference of edge length compared to original coordinates of the mesh. This is useful, for example, when a mesh is deformed by modifiers.

Tangent
Uses the optional tangent vector as texture coordinates.

Projection

Flat
Flat mapping gives the best results on single planar faces. It does produce interesting effects on the sphere, but compared to a sphere-mapped sphere the result looks flat. On faces that are not in the mapping plane the last pixel of the texture is extended, which produces stripes on the cube and cylinder.

Cube
Cube mapping often gives the most useful results when the objects are not too curvy and organic (notice the seams on the sphere).

Tube
Tube mapping 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.

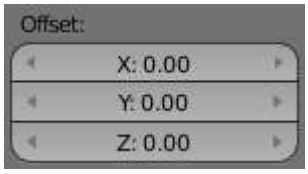
Sphere
Sphere mapping is the best type for mapping a sphere, and it is perfect for making planets and similar objects. It is often very useful for creating organic objects. It also produces interesting effects on a cylinder.

Inheriting coordinates from the parent object

From Dupli
Duplis instanced from vertices, faces, or particles, inherit texture coordinates from their parent.

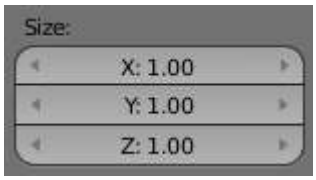
Todo: explaination

Coordinate Offset, Scaling and Transformation



Offset panel

Offset
The texture co-ordinates can be translated by an offset. Enlarging of the Ofs moves the texture towards the top left.



Size panel

Size

These buttons allow you to change the mapping of axes between the texture's own coordinate system, and the mapping system you choose (Generated, UV, etcetera.) More precisely, to each axis of the texture corresponds one of four choices, that allow you to select to which axis in the mapping system it maps! This implies several points:

- For 2D textures (such as images), only the first two rows are relevant, as they have no Z data.
- You can rotate a 2D picture a quarter turn by setting the first row (i.e. X texture axis) to Y, and the second row (Y texture axis) to X.
- When you map no texture axis (i.e. the three "void" buttons are set), you'll get a solid uniform texture, as you use zero dimension (i.e. a dot, or pixel) of it (and then Blender extends or repeats this point's color along all axes.)
- When you only map one texture axis (i.e. two "void" buttons are enabled) you'll get a "striped" texture, as you only use one dimension (i.e. a line of pixel) of it, (and then Blender stretches this line along the two other axes).
- The same goes, for 3D textures (i.e. procedural ones), when one axis is mapped to nothing, Blender extends the plan ("slice") along the relevant third axis.

So, all this is a bit hard to understand and master. Fortunately, you do not have to change these settings often, except for some special effects... Anyway, the only way to get used to them is to practice!

Environment Maps

Environment maps take a render of the 3D scene and apply it to a texture, to use for faking reflections. If you want to achieve a very realistic result, raytraced reflections are a good solution. Environment Maps are another way to create reflective surfaces, but they are not so simple to set up.

So why should one use Environment Maps?

- The main reason is probably that they can be much faster than raytracing reflections. In certain situations they need to be calculated only once, and may be reused like any ordinary texture. You may even modify the precalculated Environment Map in an image editor.
- Environment maps can also be blurred and render even faster because the resolution can then be lowered. Blurring a reflection with the raytracer always adds to the render time, sometimes quite a lot.
- *Halos* (a visualization type for particles) are not visible to raytraced reflections, so you need to setup environment maps to reflect them.
- *Keypoint strands* (another visualization type for particles) are also not visible to raytraced reflections, so you need to setup environment maps to reflect them.

Just as we render the light that reaches the viewing plane using the camera to define a viewpoint, we can render

the light that reaches the surface of an object (and hence, the light that might ultimately be reflected to the camera). Blender's environment mapping renders a cubic image map of the scene in the six cardinal directions from any point. When the six tiles of the image are mapped onto an object using the *Refl* input coordinates, they create the visual complexity that the eye expects to see from shiny reflections.

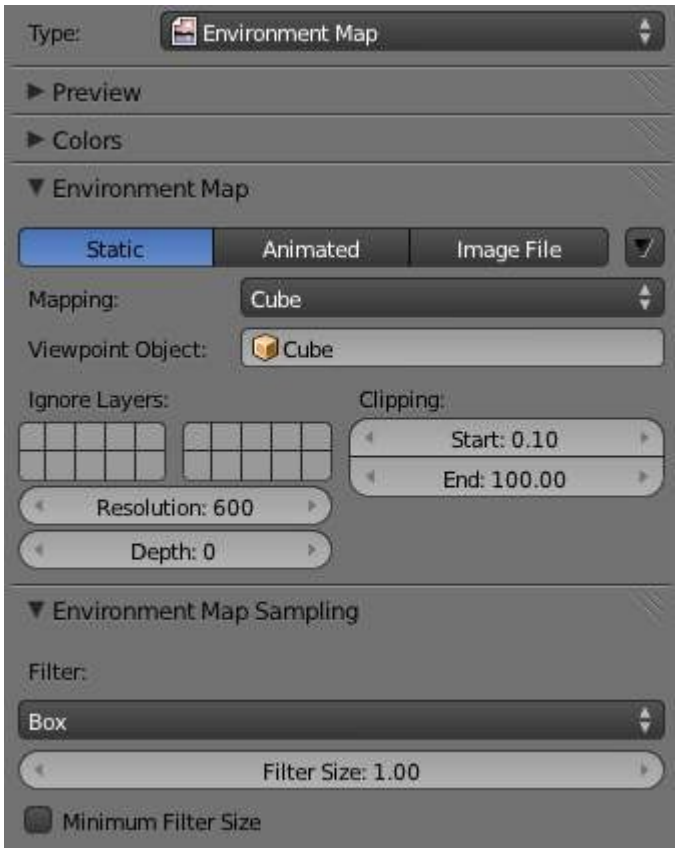
Note

It's useful to remember here that the true goal of this technique is *believability*, not *accuracy*. The eye doesn't need a physically accurate simulation of the light's travel; it just needs to be lulled into believing that the scene is real by seeing the complexity it expects. The most unbelievable thing about most rendered images is the sterility, not the inaccuracy.

Options

Important

For correct results, the mapping of an environment map texture must be set to 'Refl' (reflection co-ordinates) in the Map Input panel of the Material context.



Reflecting plane EnvMap settings.

Blender allows three types of environment maps, as you can see in *Reflecting plane EnvMap settings.* :

Static
: The map is only calculated once during an animation or after loading a file.

Animated
The map is calculated each time a rendering takes place. This means moving Objects are displayed correctly in mirroring surfaces.

Image File

When saved as an image file, environment maps can be loaded from disk. This option allows the fastest rendering with environment maps, and also gives the ability to modify or use the environment map in an external application.

When using planar reflections, if the camera is the only moving object and you have a reflecting plane, the Empty must move too and you must use *Anim* environment map. If the reflecting object is small and the Empty is in its center, the environment map can be *Static*, even if the object itself rotates since the Empty does not move. If, on the other hand, the Object translates the Empty should follow it and the environment map be of *Anim* type.

Options in dropdown menu:

Clear Environment Map
Clears the currently rendered environment map from memory. This is useful to refresh a *Static* environment maps and you have changed things in your scene since the last time the environment map was rendered. *Anim* environment maps do this automatically on every render.

Save Environment Map
Saves the currently stored static environment map to disk as an image file. This can be loaded again with *Load*.

Clear All Environment Maps
Does the same as *Free Data*, but with all environment maps in the scene. This is a useful shortcut when using recursive environment maps (when the *Depth* is greater than 0).

Note
EnvMap calculation can be disabled at a global level by the EnvMap Tog Button in the Render Panel of the Rendering Buttons.

Viewpoint Object

Environment maps are created from the perspective of a specified object. The location of this object will determine how ‘correct’ the reflection looks, though different locations are needed for different reflecting surfaces. Usually, an Empty is used as this object.

- For planar reflections, the object should be in a location mirrored from the camera, on the other side of the plane of reflection (see Examples). This is the most accurate usage of Environment maps.
- For spherical reflections, the object should be in the center of the sphere. Generally, if the reflecting sphere’s object center point is in the center of its vertices, you can just use the name of the actual sphere object as the *Ob:*
- For irregular reflections, there’s no hard and fast rule, you will probably need to experiment and hope that the inaccuracy doesn’t matter.

Ignore Layers

The layers to exclude from the environment map creation. Since environment maps work by rendering the scene from the location of the *Ob:* object, you will need to exclude the actual reflecting surface from the

environment map, otherwise it will occlude other objects that should be reflected on the surface itself.

Eg. If you are rendering an environment map from the center of a sphere, all the environment map will show by default is the inside of the sphere. You will need to move the sphere to a separate layer, then exclude that layer from the environment map render, so that the environment map will show (and hence reflect) all the objects outside the sphere.

Resolution
The resolution of the cubic environment map render. Higher resolutions will give a sharper texture (reflection), but will be slower to render.

Depth
The number of recursive environment map renders. If there are multiple reflecting objects using environment maps in the scene, some may appear solid, as they won't render each other's reflections. In order to show reflections within reflections, the environment maps need to be made multiple times, recursively, so that the effects of one environment map can be seen in another environment map. See Examples.

Clipping Start/End
The clipping boundaries of the virtual camera when rendering the environment map. Sets the minimum and maximum distance from the camera that will be visible in the map.

Environment Map Sampling

Filter

Box
Box Filter

EWA

Elliptical Weighted Average - one of the most efficient direct convolution algorithms developed by Paul Heckbert and Ned Greene in the 1980s. For each texel, EWA samples, weights, and accumulates texels within an elliptical footprint and then divides the result by the sum of the weights.

Eccentricity
Maximum eccentricity (higher gives less blur at distant/oblique angles, but is also slower)

FELINE

FELINE (Fast Elliptical Lines), uses several isotropic probes at several points along a line in texture space to produce an anisotropic filter to reduce aliasing artifacts without considerably increasing rendering time.

Probes
Maximum number of samples (higher gives less blur at distant/oblique angles, but is also slower)

Area

Eccentricity

Maximum eccentricity (higher gives less blur at distant/oblique angles, but is also slower)

Filter Size

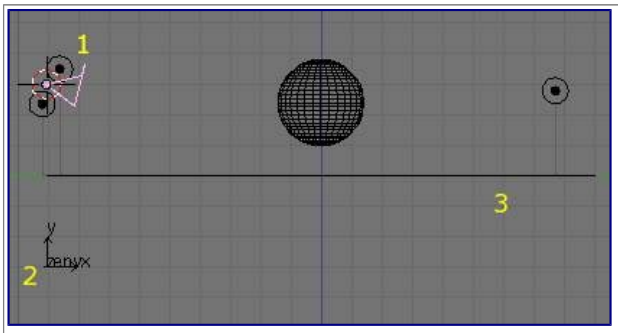
The amount of blurring applied to the texture. Higher values will blur the environment map to fake blurry reflections.

Minimum Filter Size

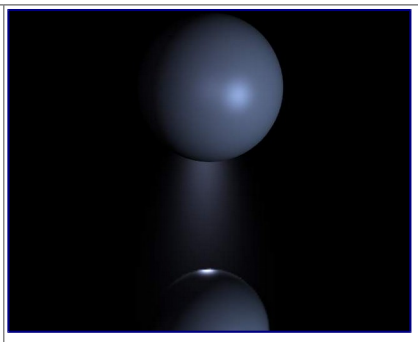
Use Filter Size as a minimal filter value in pixels

Examples

In this example, an empty is used as the *Ob:* of the reflecting plane's environment map. It is located in the specular position of the camera with respect to the reflecting surface. (This is possible, strictly speaking, only for planar reflecting surfaces.) Ideally, the location of the empty would mirror the location of the camera across the plane of the polygon onto which it is being mapped.

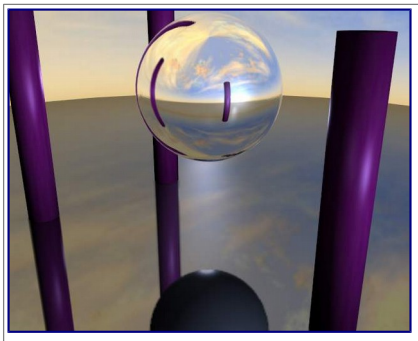


Planar reflection example. 1: Camera, 2: Empty, 3: Reflecting Plane.

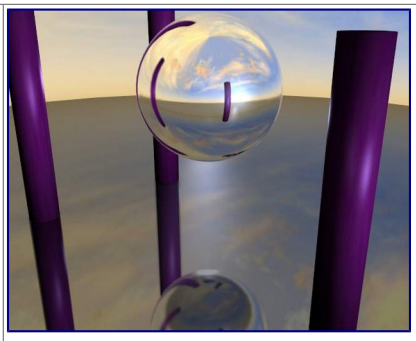


Sphere on a reflecting surface.

The following images show the effect of the *Depth*. The first render has depth set to 0. This means the environment map on the plane has rendered before the environment map of the sphere, so the sphere's reflection isn't shown. By raising the *Depth*, the environment map is rendered recursively, in order to get reflections of reflections.



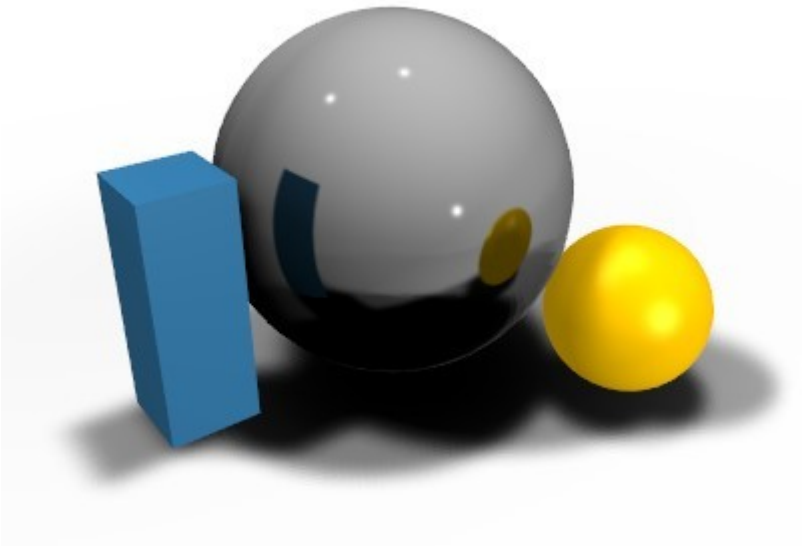
Reflecting sphere on a reflecting surface.



Reflecting sphere on a reflecting surface with multiple reflections.

Limitations

Because environment maps are calculated from the exact location of the *Viewpoint Object's* object center, and not from actual reflecting surface, they can often be inaccurate, especially with spheres. In the following image, the rectangular prism and the smaller spheres are touching the sides of the large reflecting sphere, but because the environment map is calculated from the center of the sphere, the surrounding objects look artificially far away.



Inaccurate spherical reflection, the colored objects are artificially offset