# Textures Tutorial

Using Cobalt™, Xenon™, Argon™

Textures and Texture Space

This document will focus on the description and use of texture space shaders within the Ashlar-Vellum line of 3D modeling products.

There are two basic types of textures used by the advanced rendering engine of Cobalt, Xenon or Argon. These are solid textures and wrapped textures.

Solid Textures

Solid textures calculate their results based upon the shading point's position in 3D space. Solid textures are able to generate arbitrarily complex and coherent patterns. Shaders that implement solid textures include the "marble" or "simple wood" color shaders. Solid textures create the appearance that an object has been carved out of a solid block of material. The visual appearance at any point on the object's exterior is determined by intersecting the object's surface with the solid material. Texture space shaders are not needed for and have no effect on solid textures.
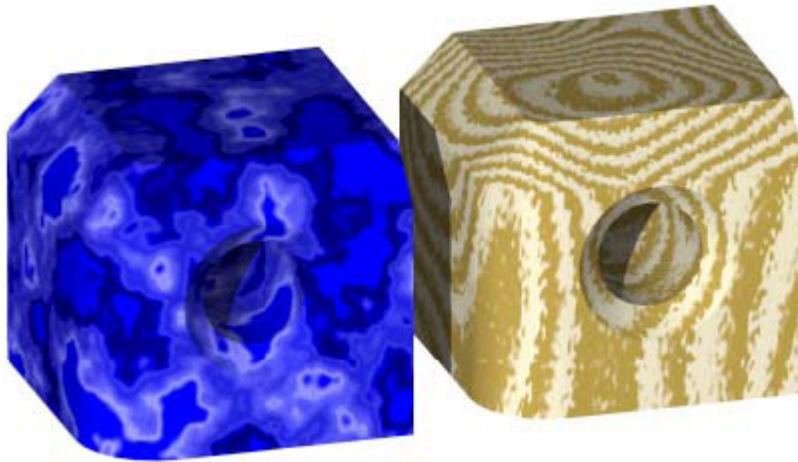

Solid textures created by the marble and simple wood color shaders.

Wrapped Textures

Wrapped textures calculate their results by mapping the shading point's position in 3D space to a 2D coordinate system. That is, wrapped textures operate within a 2D coordinate system known as texture space. Texture space shaders handle the mapping from 3D space (x, y, z) to 2-D texture space (s, t). There are many possible 3D to 2D mappings available. Hence, there are many possible texture space shaders.

Texture space is a two-dimensional coordinate system denoted by (s, t). In general, when a texture space is applied to an object and viewed from the outside, the axes formed by the increasing s-direction, t-direction and the outward surface normal vector will form a left-handed coordinate system. Texture space is configured in this way so that when a pattern (such as image data) is applied, the origin of the pattern is located at its top-left corner with s increasing to the right of the image, and t to the bottom. Also, many texture space shaders operate within the unit square such that 0 <= s,t <= 1.

The texture pattern will repeat outside of this range. This repeating pattern is often referred to as tiling. A sample texture pattern and its associated texture coordinate system are illustrated below:
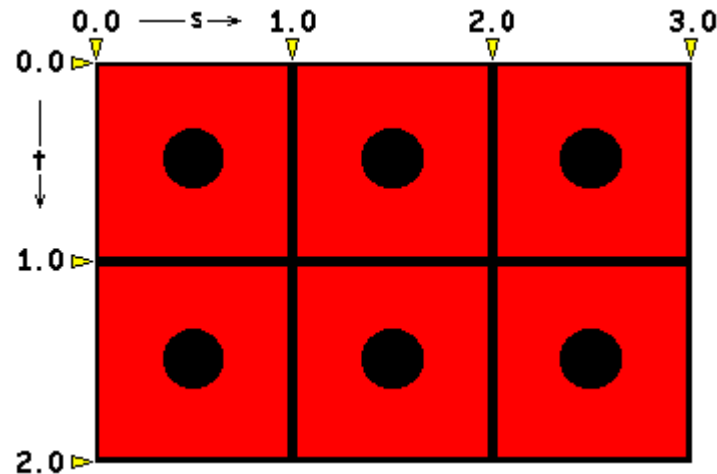
The texture space (s, t) coordinate system.
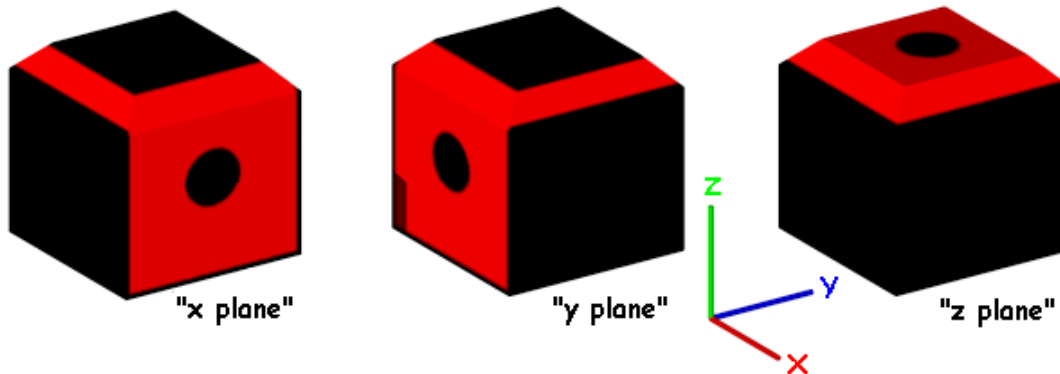
Texture Space Shaders

There are several common texture space shaders provided with the advanced rendering engine of the Ashlar-Vellum line of 3D modeling products. These shaders are named *x plane*, *y plane*, *z plane*, *auto axis*, *arbitrary plane*, *cylindrical* and *spherical*. The applied material initially determines an object's texture space shader. This can be changed and edited at any time using the Render Material Settings dialog box. This dialog is displayed by pressing the Advanced… button on the Material page of the Edit Object dialog box.
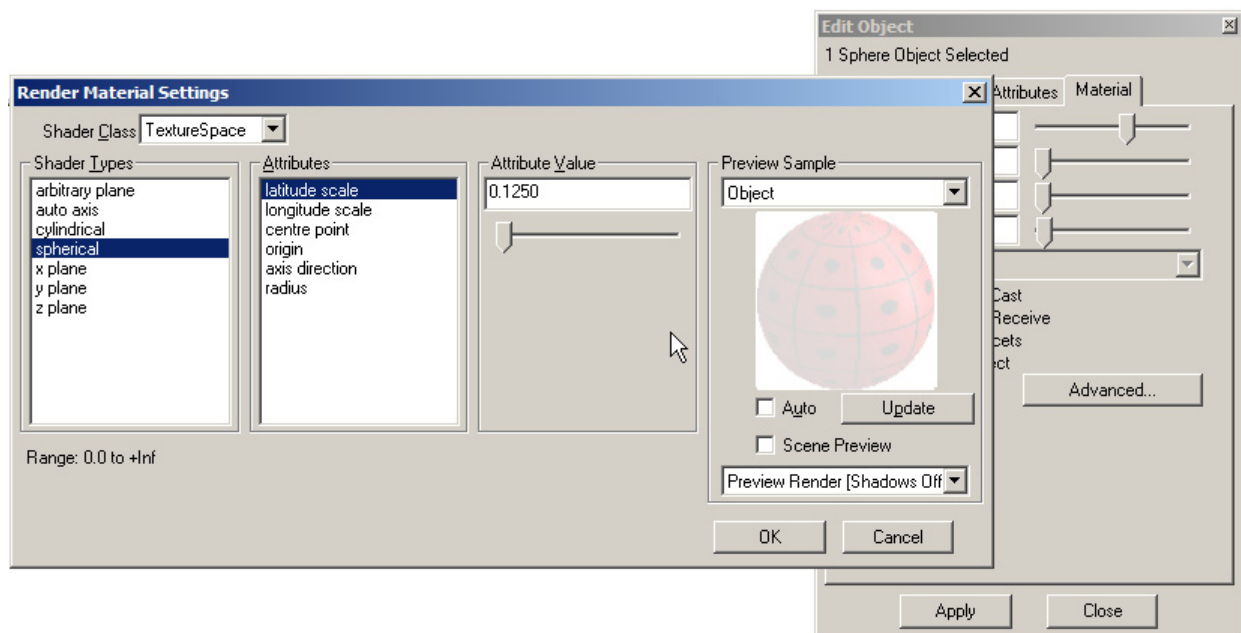

The Render Material Settings dialog box.

The X Plane, Y Plane, Z Plane Texture Space Shaders.

The *x plane*, *y plane* and *z plane* texture space shaders are the thee simplest shaders. They take a texture and project it along a vector parallel to the x, y, or z-axis respectively. The linear projection used by these shaders results in the pattern appearing on both "sides" of the object. The figure below demonstrates the effect of these texture space shaders when used with the *wrapped image* color shader.

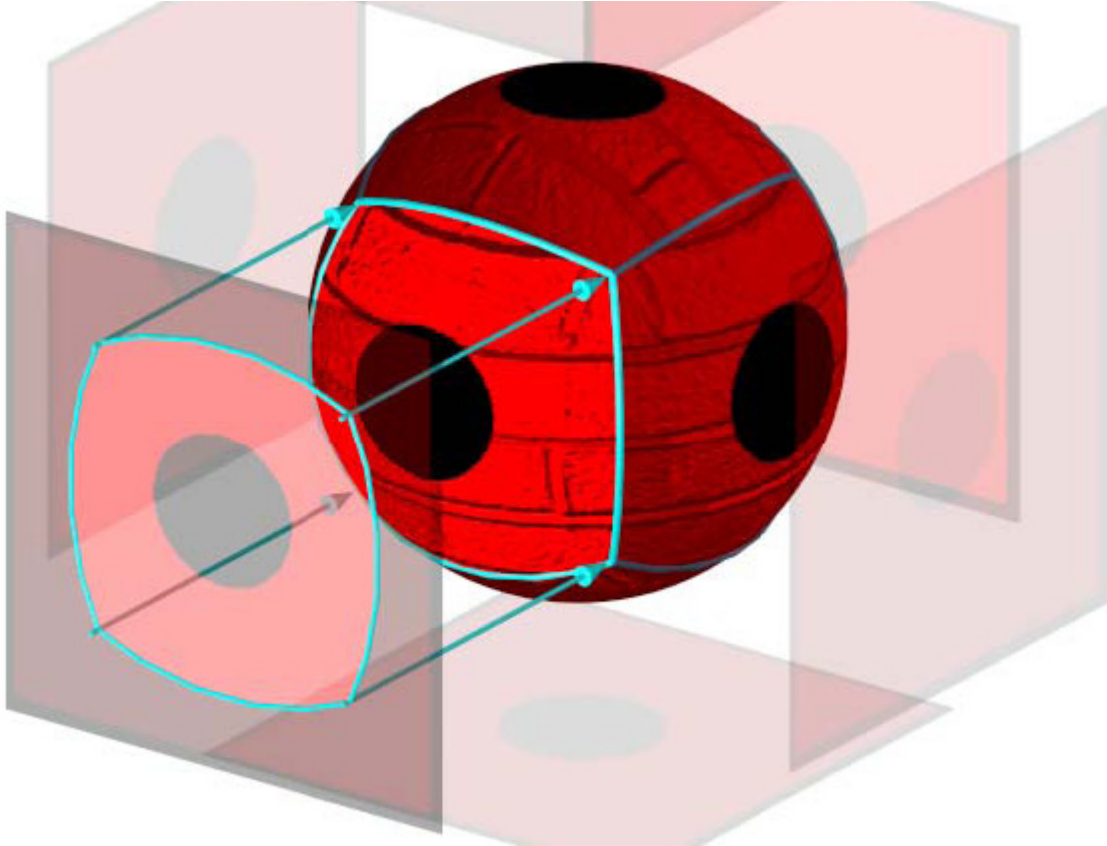**The *x plane*, *y plane* and *z plane* texture space shaders used with the "wrapped image" color shader.**

These shaders only have one attribute: scale. The scale controls the number of times the texture will repeat per model inch (TilesPerInch = 1.0/scale). The figure below demonstrates the effect scale has on the "wrapped image" color shader when used with the *x plane*, *y plane* and *z plane* texture space.

scale=1.00   scale=0.50   scale=0.25
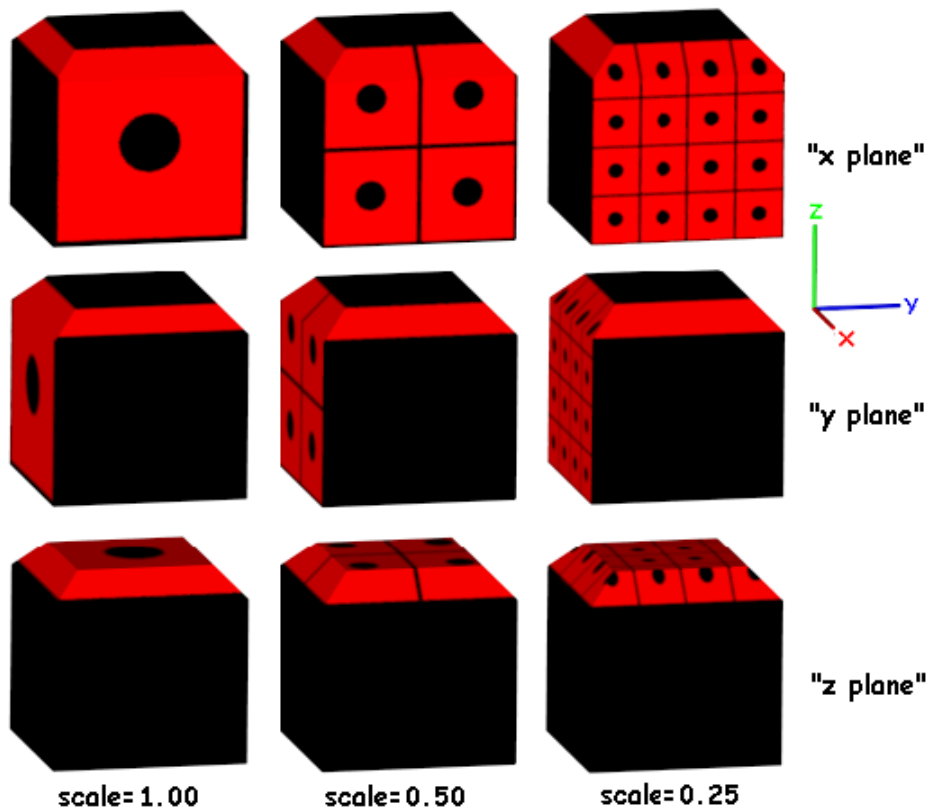
The *x plane, y plane* and *z plane* texture space with different scale values.
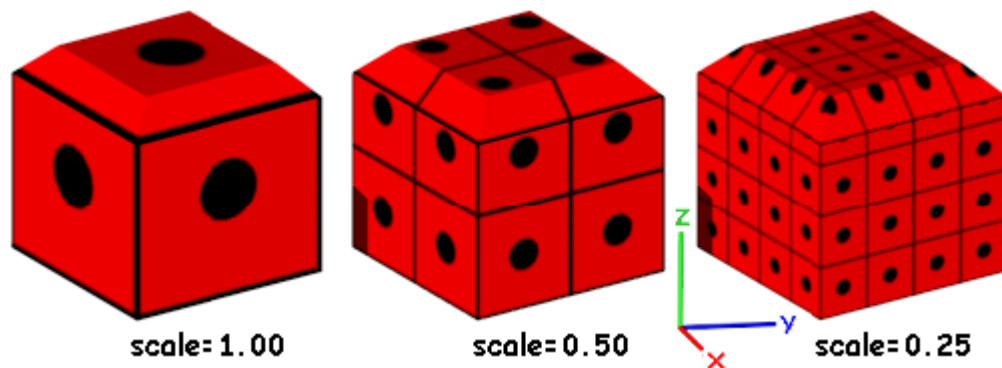
The Auto Axis Texture Space Shader

The *auto axis* texture space shader is a hybrid of the *x plane*, *y plane*, and *z plane* texture space shaders. This shader selects the x, y, or z plane texture space shader whose projection is most "closely aligned" to the surface normal at each evaluation point. When used on a sphere, the *auto axis* texture space shader produces a pattern that resembles the patches on a soccer ball.

**The *auto axis* texture space shader mapped onto a sphere.**

Like the xyz plane shaders, this shader only has one attribute: scale. The scale controls the number of times the texture will repeat per model inch (TilesPerInch = 1.0/scale). The figure below demonstrates the effect scale has on the "wrapped image" color shader when used with the *auto axis* texture space.

scale= 1.00   scale= 0.50   scale= 0.25

The "auto axis" texture space with different scale values.
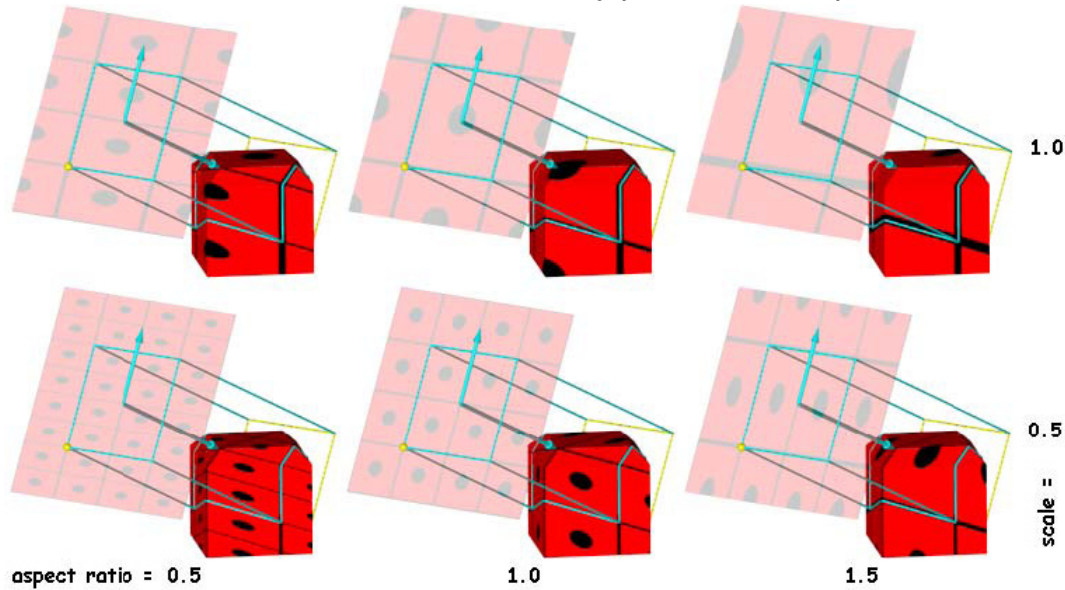
The Arbitrary Plane Texture Space Shader

The *arbitrary plane* texture space shader projects a texture along a user defined vector. The linear projection used by this shader results in the pattern appearing on both "sides" of the object. With the proper settings, the "arbitrary plane" texture space shader can be made to emulate each of the *x plane, y plane* and *z plane* texture space shaders. The figure below demonstrates the effect of the *arbitrary plane* texture space shader when used with the "wrapped image" color shader.
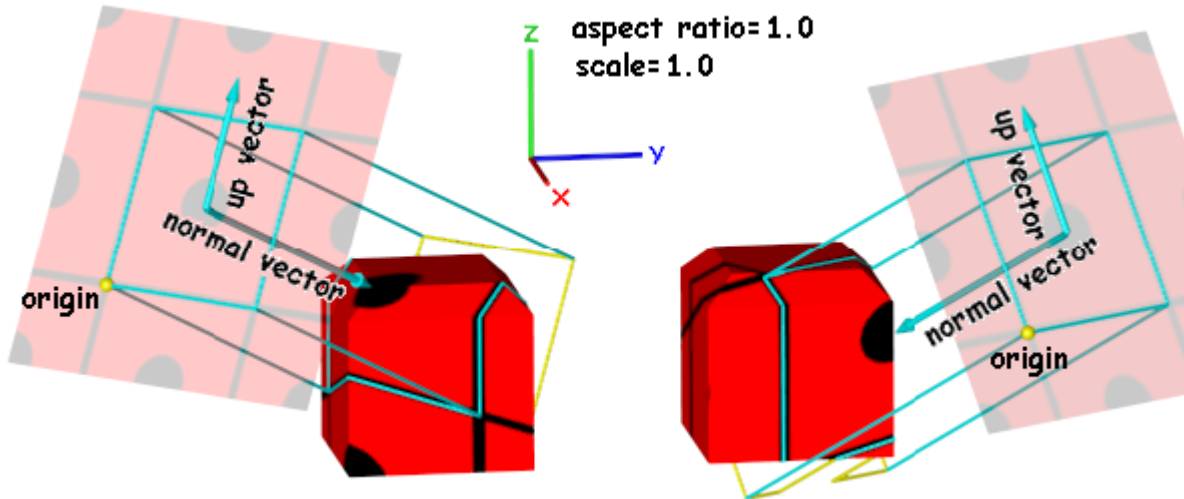
**The "arbitrary plane" texture space shader used with
the "wrapped image" color shader.**

The *arbitrary plane* texture space shader has five attributes: scale, aspect ratio, origin, normal vector and up vector. The scale controls the number of times the texture will repeat in the plane per model inch (TilesPerInch = 1.0/scale). The aspect ratio is computed as one unit of height divided by one unit of its width. The scale and aspect ratio work together such that the horizontal scale is equal to scale, and the vertical scale is the product of scale and aspect ratio. The origin defines the corner of the applied texture image. The normal vector defines the plane's perpendicular direction. The up vector defines the textures vertical orientation within the plane.

The figure below demonstrates the effect scale and aspect ratio have on the "wrapped image" color shader when used with the *arbitrary plane* texture space.

The *arbitrary plane* texture space with different scale and aspect ratio values.
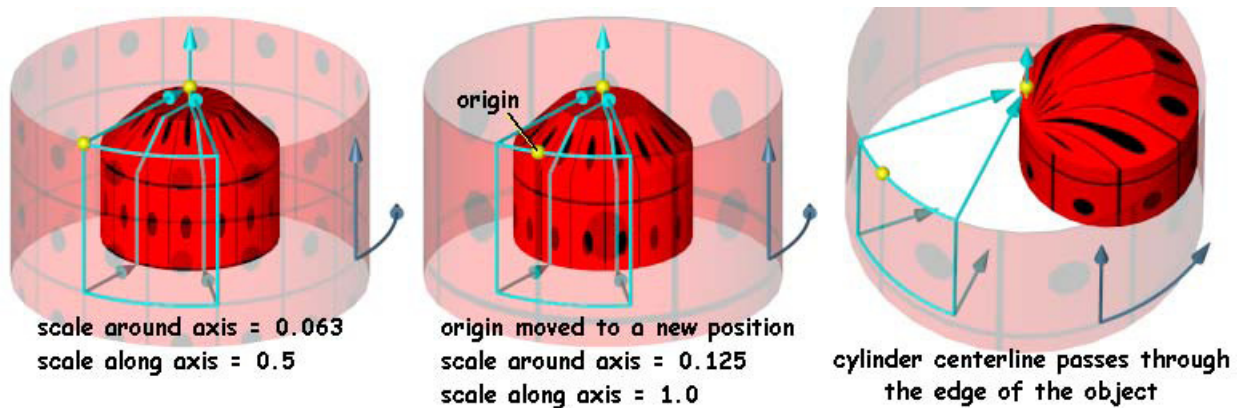
The Cylindrical Texture Space Shader

The *cylindrical* texture space shader projects a texture along the planar radials of an imaginary, user defined cylinder. The radial projection used by this shader causes the texture pattern to degenerate to a line as it approaches the cylinder's centerline. This distortion is most noticeable at the top and bottom areas of the object. The figure below demonstrates the effect of the *cylindrical* texture space shader when used with the "wrapped image" color shader.
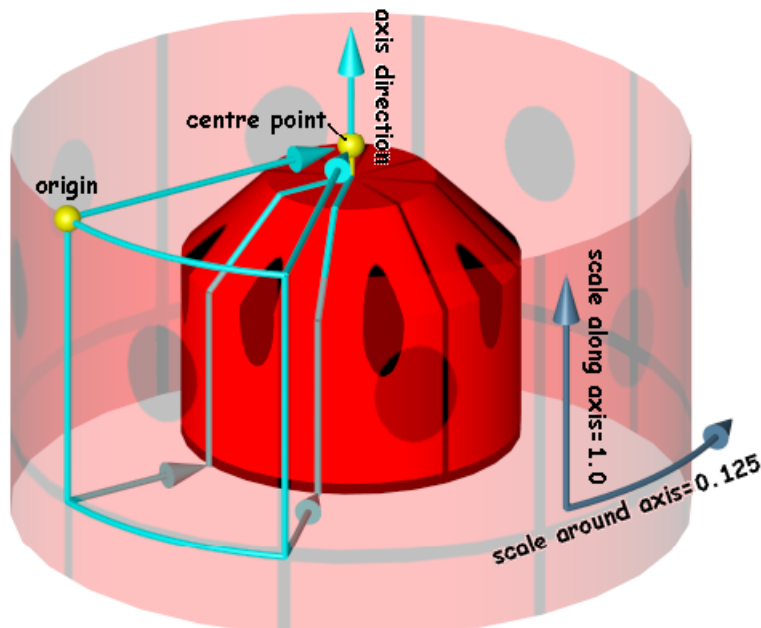
**The *cylindrical* texture space shader used with
the "wrapped image" color shader.**

The *cylindrical* texture space shader has five attributes: scale around axis, scale along axis, center point, axis direction and origin. The scale around axis controls the number of times the texture will repeat around the circumference of the cylinder per model inch. The scale along axis controls the number of times the texture will repeat along the height of the cylinder per model inch (TilesPerInch = 1.0/scale). The center point lies on the cylinder's centerline. The axis direction defines the orientation of the cylinder's centerline and the "up" direction for the texture. The origin lies on the surface of the cylinder and defines the corner of the applied texture image.

When using the *cylindrical* texture space shader, the distance between the *origin* and the centerline will *not* affect the appearance of the resulting image. Only the angular and vertical position of the *origin* relative to the centerline will affect the resulting image. The figure below demonstrates different applications the "wrapped image" color shader used with the *cylindrical* texture space.

scale around axis = 0.063
scale along axis = 0.5

origin moved to a new position
scale around axis = 0.125
scale along axis = 1.0

cylinder centerline passes through
the edge of the object

The *cylindrical* texture space with different configurations.

The Spherical Texture Space Shader

The *spherical* texture space shader projects a texture along the radials of an imaginary, user defined sphere. The radial projection used by this shader causes the texture pattern to degenerate to a point as it approaches either the sphere's center or poles. This distortion is most noticeable at the top and bottom (polar) areas of the object. The figure below demonstrates the effect of the *spherical* texture space shader when used with the "wrapped image" color shader.
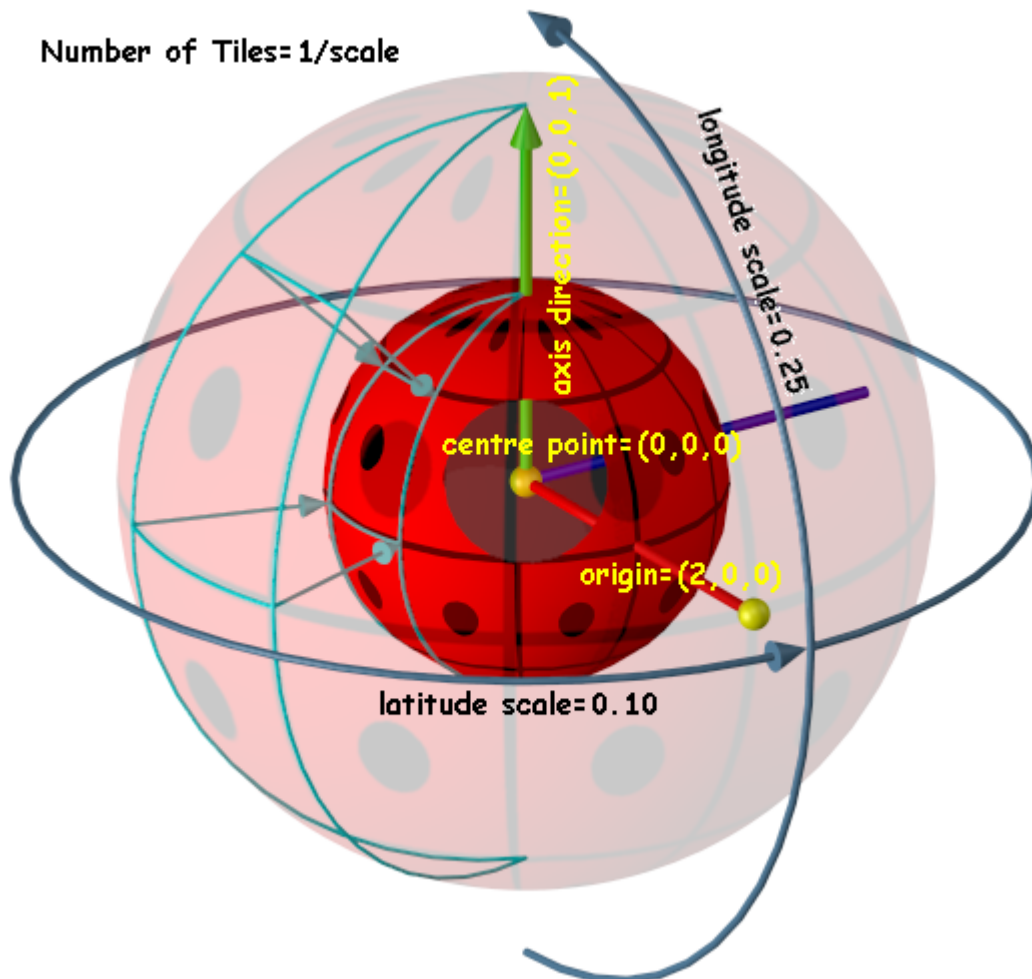
**The *spherical* texture space shader used with
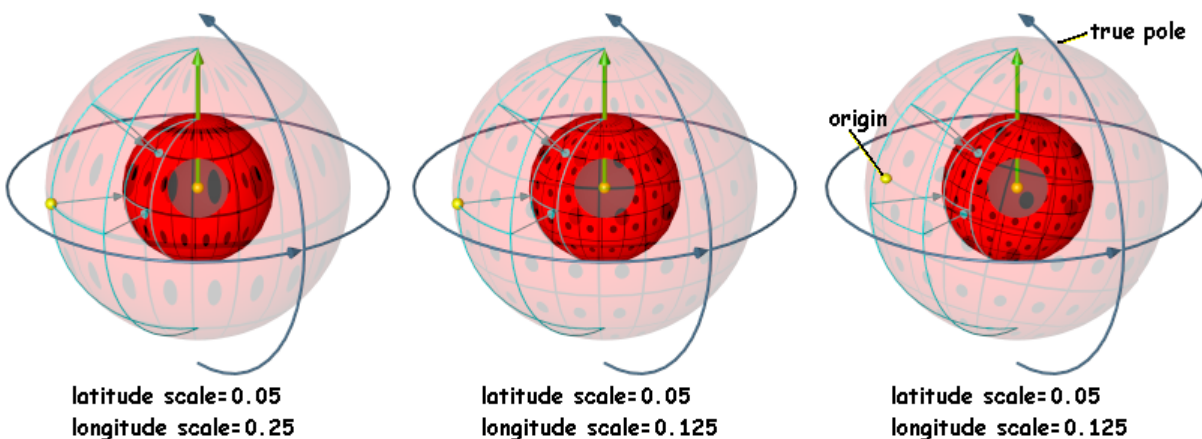the "wrapped image" color shader.**

The *spherical* texture space shader has five attributes: latitude scale, longitude scale, center point, origin and axis direction. The latitude scale controls the number of times the texture will repeat around the equator of the sphere per model inch. The longitude scale controls the number of times the texture will repeat from pole to pole on the sphere per model inch (TilesPerInch = 1.0/scale). The center point defines the location of the sphere's center. The origin lies on the sphere's equator and defines the corner of the applied texture. The axis direction defines the orientation of the sphere's "true polar axis" and the "up" direction for the texture.

When using the *spherical* texture space shader, the distance between the origin and the center point will *not* affect the appearance of the resulting image. Only the angular position of the origin relative to the center point will affect the resulting image. Also, in the general case, the axis direction will not be the "true polar axis." The figure below diagrams the vector equations used internally to compute the true polar axis.
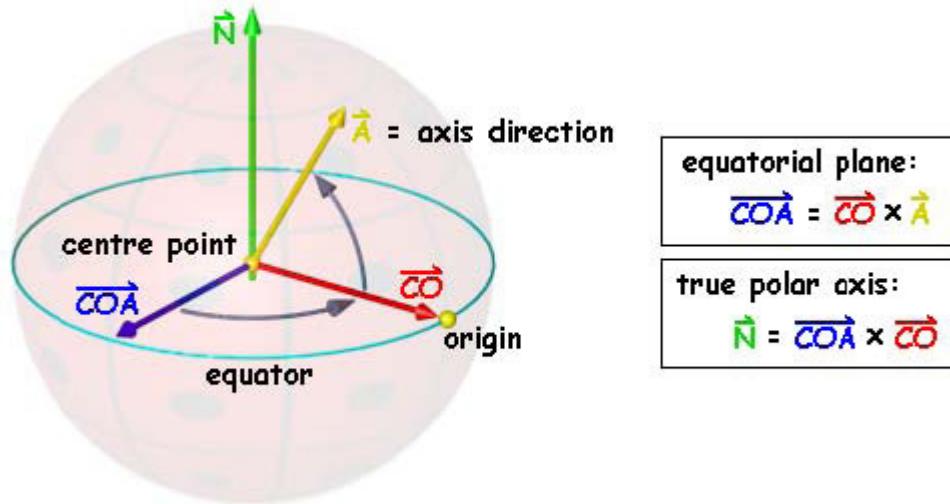
Calculating the true polar axis.

The equator will be perpendicular to the plane formed by the origin, center point vector (CO) and the axis direction (A). The axis direction is *not* required to be perpendicular to the vector CO. However, the axis direction must never be parallel to the vector CO. The figure below demonstrates different applications the "wrapped image" color shader used with the *spherical* texture space.

latitude scale=0.05
longitude scale=0.25

latitude scale=0.05
longitude scale=0.125

latitude scale=0.05
longitude scale=0.125

The *spherical* texture space with different configurations.