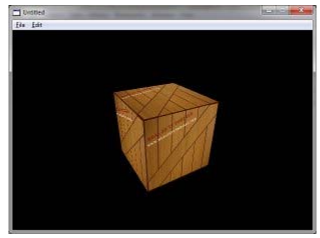
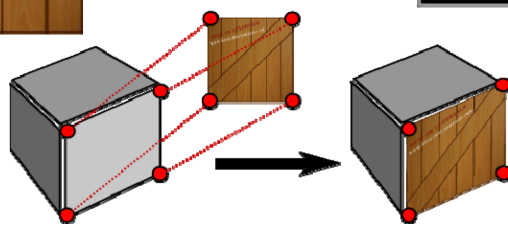
Texture Mapping

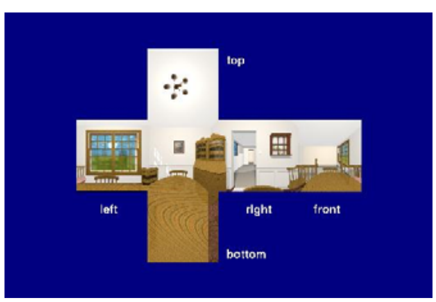
Pasting textures on surfaces





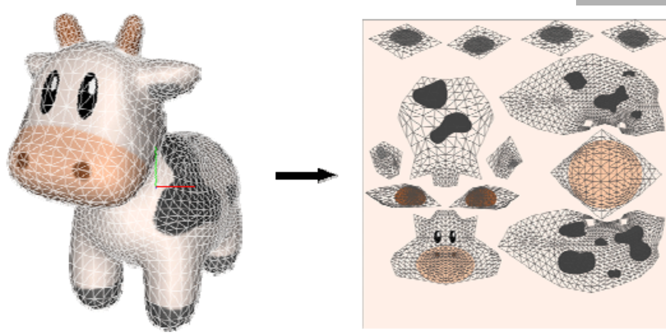
Texture Mapping

Pasting textures on surfaces



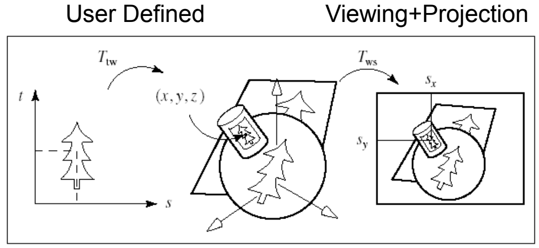
Texture Mapping

Pasting textures on surfaces



Coordinate Systems Involved

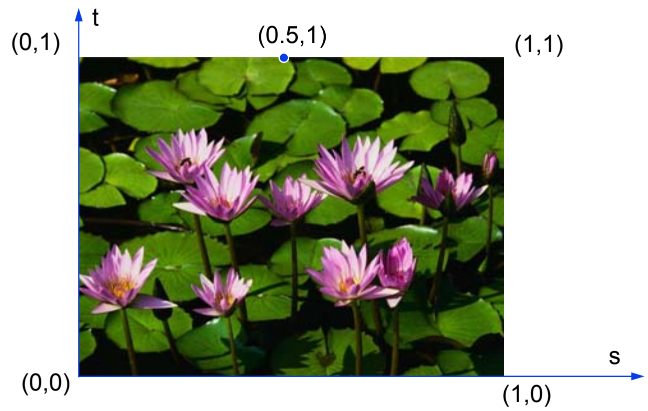
FIGURE 8.35 Drawing texture on several objects of different shape.



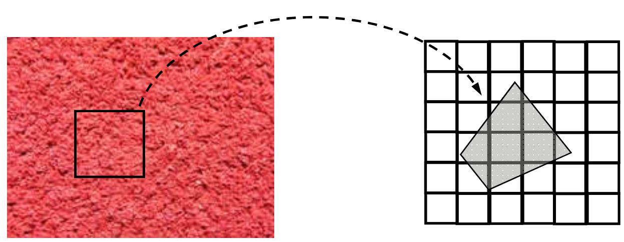
$$(s_x,s_y) = T_{ws}(T_{tw}(s,t))$$

Textures are Images

They are always assigned the shown parametric coordinates (s,t)



Texture to Screen

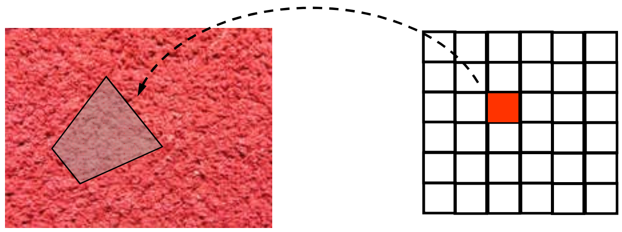


$(s_x,s_y) = T_{ws}(T_{tw}(s,t))$

We would have to calculate pixel coverages

Screen to Texture

Better approach



$(s,t) = T_{wt}(T_{sw}(s_x,s_y))$

Requires inverting the projection matrix

From Texture to World (Object)

To apply a texture to an object, we must find a correspondance between (s,t) and some object coordinate system

- Mapping via a parametric representation of the object space
- Manually

Mapping the Texture to an Object Parametric Representation

Linear transformation

From texture space (s,t) to object space (u,v)

$u = u(s,t) = a_u\, s + b_u\, t + c_u$

$v = v(s,t) = a_v\, s + b_v\, t + c_v$

s in [0,1]

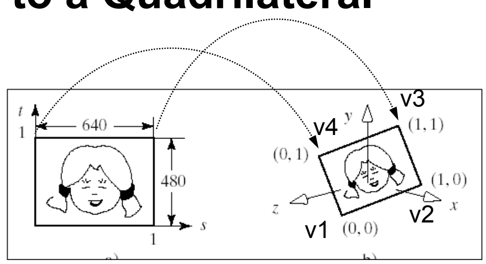
t in [0,1]

Example 1: Image to a Quadrilateral

Simply

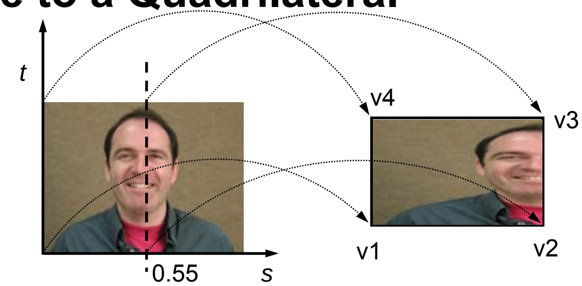
$u = u(s,t) = s$

$v = v(s,t) = t$




```
glTexCoord2f(0,0) ;  glVertex3dv(v1) ;
glTexCoord2f(1,0) ;  glVertex3dv(v2) ;
glTexCoord2f(1,1) ;  glVertex3dv(v3) ;
glTexCoord2f(0,1) ;  glVertex3dv(v4) ;
```

Example 2:
Piece of Image to a Quadrilateral



Use only left part

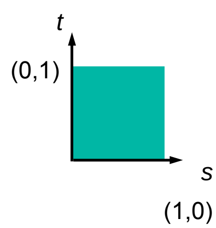
$u = u(s,t) = 0.55s$

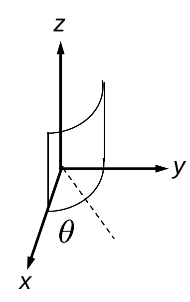
$v = v(s,t) = t$

```
glTexCoord2f(0,0) ;  glVertex3dv(v1) ;
glTexCoord2f(0.55,0) ;  glVertex3dv(v2) ;
glTexCoord2f(0.55,1) ;  glVertex3dv(v3) ;
glTexCoord2f(0,1) ;  glVertex3dv(v4) ;
```

Packing textures for efficiency

Example 3: Square Texture to Cylinder





Parametric form of cylinder:

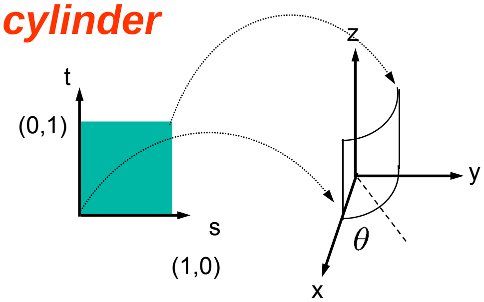
$x = r\cos\theta, \quad y = r\sin\theta, \quad z$

Surface parameters: $\quad u = \theta, \quad v = z$

with $0 \leq u \leq \pi/2$, and $0 \leq v \leq 1$

Example 3: Square Texture to Cylinder

Square texture to cylinder

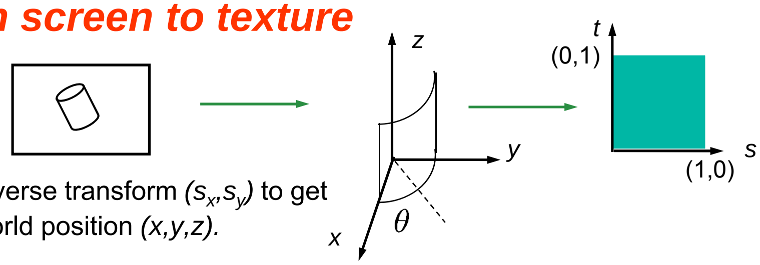


We pick the following linear transformation that maps $(s,t) = (0,0)$ to $(u,v) = (0,0)$ and $(s,t) = (1,1)$ to $(u,v) = (\frac{\pi}{2}, 1)$:

$$u = s\frac{\pi}{2}, \quad v = t$$

Example 3: Square Texture to Cylinder

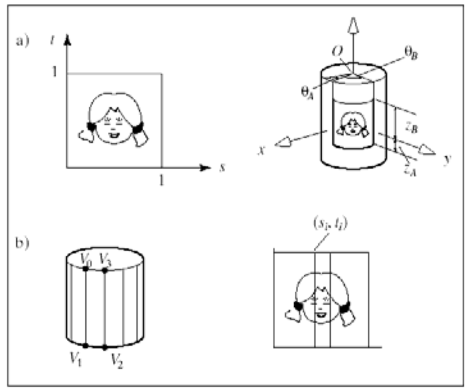
From screen to texture



1. Inverse transform (s_x, s_y) to get world position *(x,y,z).*
2. Then having *(x,y,z)*,

$$u = \tan^{-1}(y/x), \quad v = z$$
$$s = 2u/\pi, \quad t = v$$

Wrapping Textures on Curved Surfaces



$$s = \frac{\theta - \theta_a}{\theta_b - \theta_a}, \quad t = \frac{z - z_a}{z_b - z_a}$$

Cylinder with N faces

Left edge at azimuth $\theta_i = 2\pi i / N$

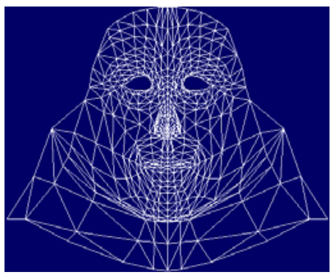
Upper left vertex texture coordinates $s_i = \dfrac{\theta_i - \theta_a}{\theta_b - \theta_a}, \quad t_i = 1$

Guerrilla CG Tutorial 09: The Basics of UV Mapping

Example 4: Facial Shape and Texture Capture

Range Image

Texture Image

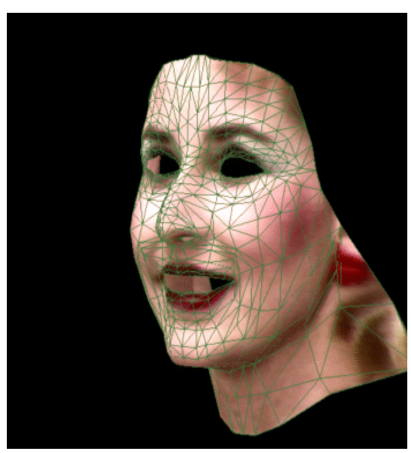


Generic Facial Mesh

Textured 3D Geometric Model

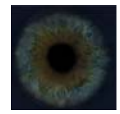
Texture map coordinates

- Positions of fitted mesh nodes in RGB texture image



Example 5: Multiple Texture Maps
Many Vertices and Texture Coordinates

Geometry* + *Texture maps



+ lighting =

How Does this Work With the Graphics Pipeline?

Rendering polygons

Only vertices go down the graphics pipeline

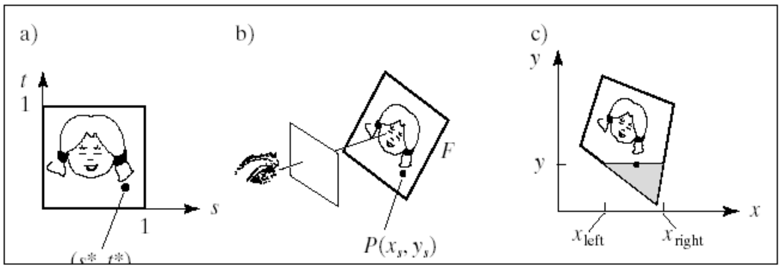
Interior points?

Calculate texture coordinates by interpolation along scanlines

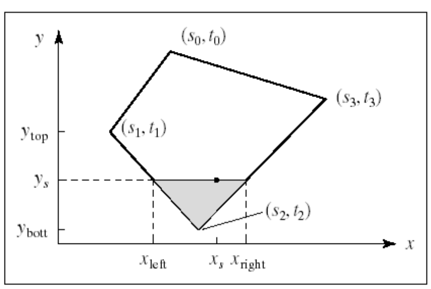
Rendering the Texture

Scanline in screen space

- Generating *s*,*t* coordinates for each pixel



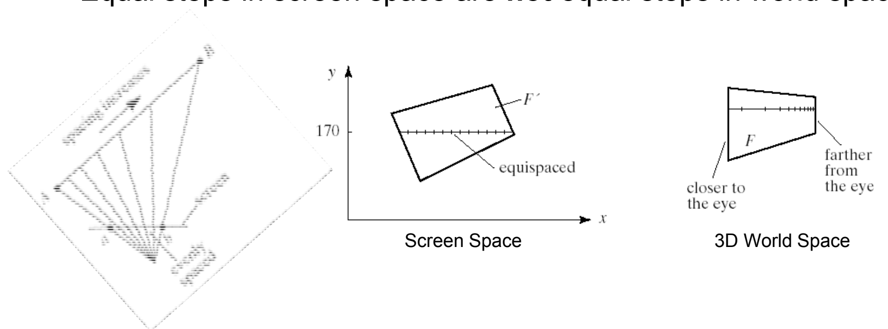
Interpolation of Texture Coordinates



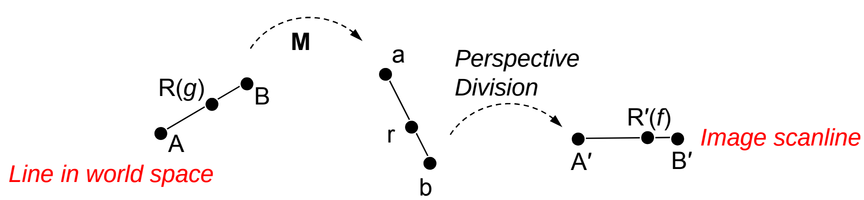
Problem

Perspective foreshortening

- Scan conversion takes equal steps along scanline in screen space
- Equal steps in screen space are **not** equal steps in world space



Reminder: In-Between Points



$$\left.\begin{aligned} R'_1(f) &= \frac{lerp(a_1, b_1, g)}{lerp(a_4, b_4, g)} \\ R'_1(f) &= lerp\left(\frac{a_1}{a_4}, \frac{b_1}{b_4}, f\right) \end{aligned}\right\} \Rightarrow g = \frac{f}{lerp(\frac{b_4}{a_4}, 1, f)}$$

substituting this in $R(g) = (1-g)A + gB$ yields

$$R_1 = \frac{lerp(\frac{A_1}{a_4}, \frac{B_1}{b_4}, f)}{lerp(\frac{1}{a_4}, \frac{1}{b_4}, f)}$$

and similarly for R_2 and R_3

Rendering Images Incrementally

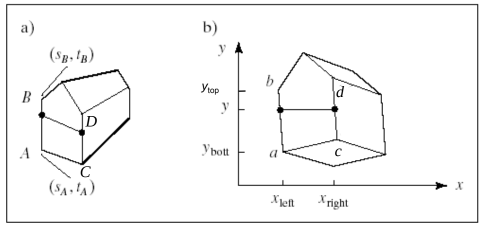
A maps to a (homogeneous)

B maps to b

C maps to c

D maps to d

For scanline y and two edges:

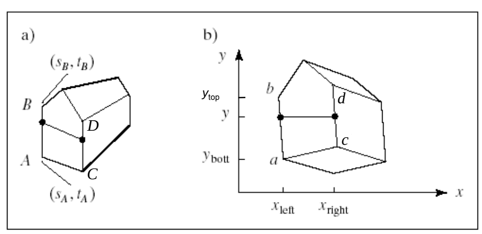


$$s_{left}(y) = \frac{lerp(\frac{s_A}{a_4}, \frac{s_B}{b_4}, f_l)}{lerp(\frac{1}{a_4}, \frac{1}{b_4}, f_l)}, \qquad s_{right}(y) = \frac{lerp(\frac{s_C}{c_4}, \frac{s_D}{d_4}, f_r)}{lerp(\frac{1}{c_4}, \frac{1}{d_4}, f_r)}$$

Once we have s_{left} and s_{right} another hyperbolic interpolation fills in the scanline

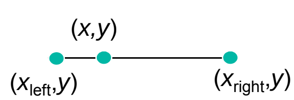
Interpolation Along the Scanline

$$s_{left}(y) = \frac{lerp(\frac{s_A}{a_4}, \frac{s_B}{b_4}, f_l)}{lerp(\frac{1}{a_4}, \frac{1}{b_4}, f_l)},$$



$$s_{right}(y) = \frac{lerp(\frac{s_C}{c_4}, \frac{s_D}{d_4}, f_r)}{lerp(\frac{1}{c_4}, \frac{1}{d_4}, f_r)}$$

$$s(x, y) = \frac{lerp(\frac{s_{left}}{h_{left}}, \frac{s_{right}}{h_{right}}, f)}{lerp(\frac{1}{h_{left}}, \frac{1}{h_{right}}, f)}$$



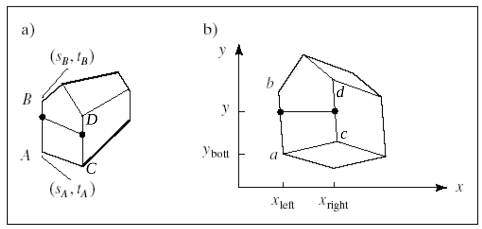
What are *f* and the *h*'s?

Interpolation Along the Scanline

$$s_{left}(y) = \frac{lerp(\frac{s_A}{a_4}, \frac{s_B}{b_4}, f_l)}{lerp(\frac{1}{a_4}, \frac{1}{b_4}, f_l\ )}, \quad s_{right}(y) = \frac{lerp(\frac{s_C}{c_4}, \frac{s_D}{d_4}, f_r)}{lerp(\frac{1}{c_4}, \frac{1}{d_4}, f_r)}$$

$$s(x,y) = \frac{lerp(\frac{s_{left}}{h_{left}}, \frac{s_{right}}{h_{right}}, f\ )}{lerp(\frac{1}{h_{left}}, \frac{1}{h_{right}}, f}$$

$$h_{left} = lerp(a_4, b_4, f_l)$$
$$h_{right} = lerp(c_4, d_4, f_r)$$
$$f = (x - x_{left})/(x_{right} - x_{left})$$



Interpolating Information (Incrementally)

Texture coordinates, Color, Normal, etc.

Right edge (1,2):

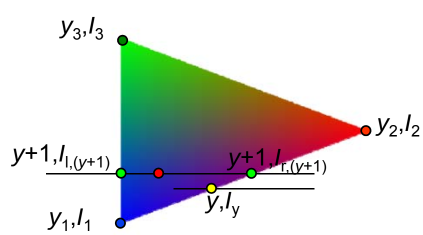
$$\frac{I_{r,(y+1)} - I_{r,y}}{(y+1) - y} = \frac{I_1 - I_2}{y_1 - y_2} \Rightarrow I_{r,(y+1)} = I_{r,y} + \frac{I_1 - I_2}{y_1 - y_2}$$

Left Edge (1,3):

$$\frac{I_{l,(y+1)} - I_{l,y}}{(y+1) - y} = \frac{I_1 - I_3}{y_1 - y_3} \Rightarrow I_{l,(y+1)} = I_{l,y} + \frac{I_1 - I_3}{y_1 - y_3}$$

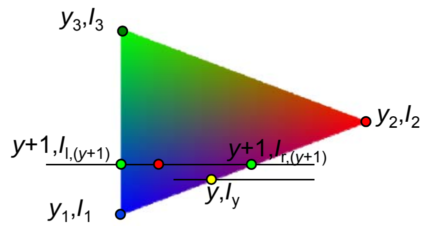
Along scanline:

$$\frac{I_{(x+1)} - I_x}{(x+1) - x} = \frac{I_r - I_l}{x_r - x_l} \Rightarrow I_{r,(y+1)} = I_{r,y} + \frac{I_r - I_l}{x_r - x_l}$$



Interpolating Information (Incrementally)

Color, Normal,
Texture coordinates



Right edge (1,2):

$$\frac{I_{r,(y+1)} - I_{r,y}}{(y+1)-y} = \frac{I_1 - I_2}{y_1 - y_2} \Rightarrow I_{r,(y+1)} = I_{r,y} + \frac{I_1 - I_2}{y_1 - y_2}$$

Left Edge (1,3):

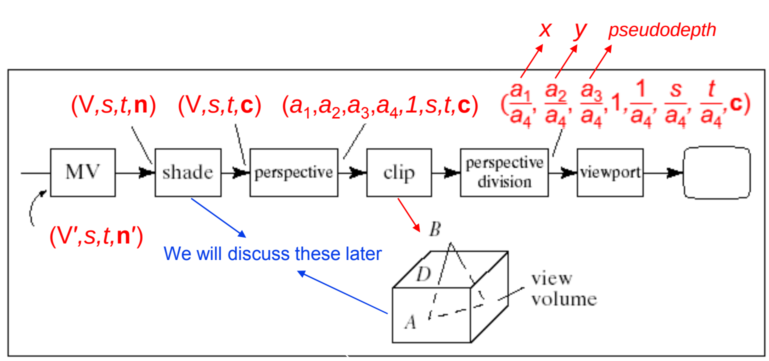
$$\frac{I_{l,(y+1)} - I_{l,y}}{(y+1)-y} = \frac{I_1 - I_3}{y_1 - y_3} \Rightarrow I_{l,(y+1)} = I_{l,y} + \frac{I_1 - I_3}{y_1 - y_3}$$

Along scanline:

$$\frac{I_{(x+1)} - I_x}{(x+1)-x} = \frac{I_r - I_l}{x_r - x_l} \Rightarrow I_{r,(y+1)} = I_{r,y} + \frac{I_r - I_l}{x_r - x_l}$$

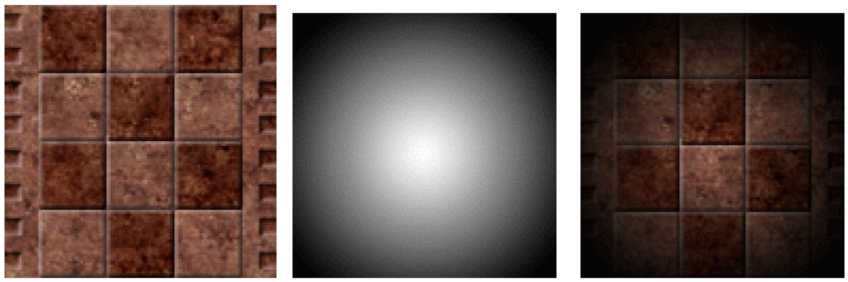
Constant along the line

Pipeline with Hyperbolic Interpolation

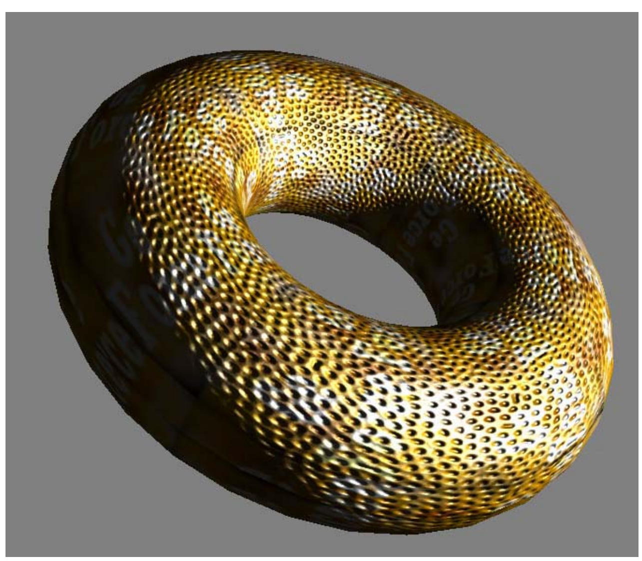


Light Maps

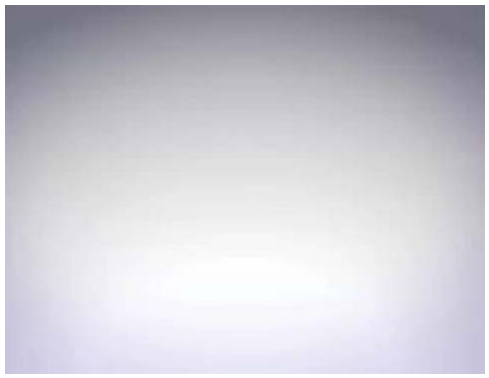
For static objects



Bump Mapping



Guerrilla CG Tutorial 10: Displacement and Bump Mapping



Procedural Texture

Volumetric textures

C = B(x,y,z)