

## Texture Mapping

### Introduced to increase realism

- Lighting/shading models not enough
- **Hide geometric simplicity**
- Images convey illusion of geometry
- Map a brick wall texture on a flat polygon
- Create bumpy effect on surface
- **Associate 2D information with 3D surface**
- Point on surface corresponds to a point in texture
- "Paint" image onto polygon

© W. Habel et al., 2006

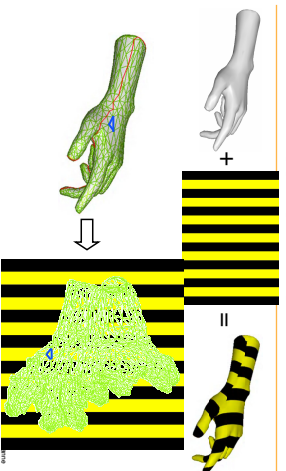
## Texture Mapping

- Real life objects have nonuniform colors, normals
- To generate realistic objects, reproduce coloring & normal variations = **texture**
- Can often replace complex geometric details



© W. Habel et al., 2006

## Texture Mapping Example

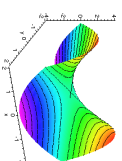


© W. Habel et al., 2006

## Texture Mapping

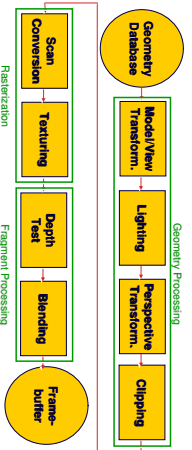
### Textures of other dimensions

- 1D: represent isovalues
- e.g.: contour lines, temp. ...
- `glTexCoor1d f (s)`



© W. Habel et al., 2006

## The Rendering Pipeline



© W. Habel et al., 2006

## Texture Mapping and Sampling

**CPSC 314**



© W. Habel et al., 2006

## Texture Coordinates

### Texture image: 2D array of color values (texels)

- Assigning texture coordinates  $(s, t)$  at vertex with object coordinates  $(x, y, z, w)$
- Use interpolated  $(s, t)$  for texel lookup at each pixel
- Use value to modify a polygon's color
- Or other surface property
- Specified by programmer or artist



© W. Habel et al., 2006

## Color Texture Mapping

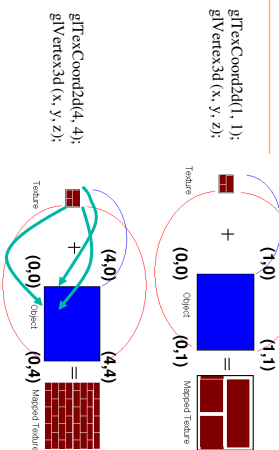
### Define color (RGB) for each pt on surface

- Two approaches
- Surface texture map
- Volumetric texture



© W. Habel et al., 2006

## Tiled Texture Map



© W. Habel et al., 2006

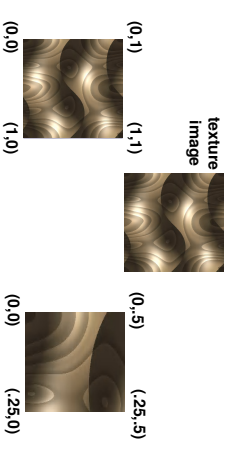
## Texture Lookup: Tiling and Clamping

### What if s or t is outside the interval [0...1]?

- **Multiple choices**
- Use fractional part of texture coordinates
- Cyclic repetition
- `glTexParameterf ... GL_TEXTURE_WRAP_S`
- `glTexParameteri ... GL_TEXTURE_WRAP_T`
- `GL_REPEAT`
- `GL_CLAMP`
- `GL_CLAMP_TO_EDGE`
- `GL_MIRROR_REPEAT`
- Clamp every component to range [0...1]
- Re-use color values from texture image border
- `glTexParameterf ... GL_TEXTURE_WRAP_S`
- `glTexParameteri ... GL_TEXTURE_WRAP_T`
- `GL_REPEAT`
- `GL_CLAMP`
- `GL_CLAMP_TO_EDGE`
- `GL_MIRROR_REPEAT`

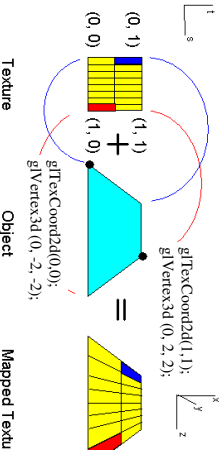
© W. Habel et al., 2006

## Fractional Texture Coordinates



© W. Habel et al., 2006

## Example Texture Map



© W. Habel et al., 2006

## Texture Objects and Binding

### Texture object

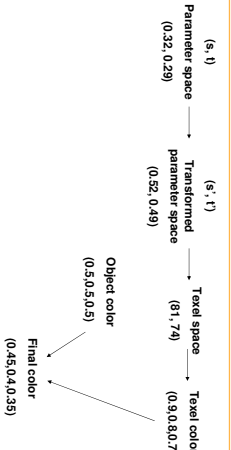
- An OpenGL data type that keeps textures resident in memory and provides identifiers to easily access them
- Provides efficiency gains over having to repeatedly load and reload a texture
- You can prioritize textures to keep in memory
- OpenGL uses least recently used (LRU) if no priority is assigned

### Texture binding

- Which texture to use right now
- Switch between preloaded textures

© W. Habel et al., 2006

## Texture Pipeline



© W. Habel et al., 2006

## Texture Functions

### Ways of applying texture colour:

- `GL_REPLACE`
- Directly use as surface color: no lighting effects
- `GL_MODULATE`
- Modulate surface color: multiply old color by new value
- Texturing happens after lighting, not first
- `GL_DECAL`
- Like replace, but modulate alpha to support transparency
- `GL_BLEND`
- Blend surface color with existing on-screen color
- `glTexEnvf (GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE, <mode>)`

© W. Habel et al., 2006

## Texture Coordinate Transformation

### Motivation

- Change scale, orientation of texture on an object

### Approach

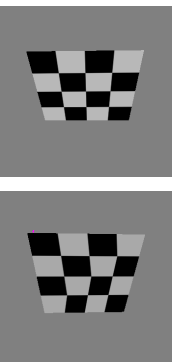
- Texture matrix stack
- Transforms specified (or generated) tex coords
- `glMatrixMode (GL_TEXTURE ) ;`
- `glLoadIdentity ( ) ;`
- `glRotate ( ) ;`
- ...
- More flexible than changing (s,t) coordinates

© W. Habel et al., 2006

## Texture Mapping

### Texture coordinate interpolation

- Perspective foreshortening problem



© W. Habel and M. Beetz/Primer

## Low-Level Details

### Large range of functions for controlling layout of texture data

- State how the data in your image is arranged  
e.g.: `glTexStorage(GL_LINEAR, ALIGNMENT, 1)` tells OpenGL not to skip bytes at the end of a row
- You must state how you want the texture to be put in memory, how many bits per "pixel", which channels...
- Textures must have a size of power of 2**
- Common sizes are 32x32, 64x64, 256x256
- But don't need to be square, i.e. 32x64 is fine
- Smaller uses less memory, and there is a finite amount of texture memory on graphics cards

© W. Habel and M. Beetz/Primer

## Basic OpenGL Texturing (cont.)

### Enable texturing:

- `glEnable(GL_TEXTURE_2D)`

### State how the texture will be used:

- `glTexEnvf(...)`

### Specify texture coordinates for the polygon:

- Use `glTexCoord2f(s, t)` before each vertex:
  - `glTexCoord2f(0, 0) ;`
  - `glVertex3f(x, y, z) ;`

© W. Habel and M. Beetz/Primer

## Basic OpenGL Texturing

### Create a texture object and fill it with texture data:

- `glGenTextures(num, &indices)` to get identifiers for the objects
- `glBindTexture(GL_TEXTURE_2D, identifier)` to bind
- Following texture commands refer to the bound texture
  - `glTexParameteri(GL_TEXTURE_2D, ..., ...)` to specify parameters for use when applying the texture
  - `glTexImage2D(GL_TEXTURE_2D, ..., ...)` to specify the texture data (the image itself)

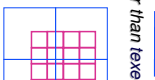


© W. Habel and M. Beetz/Primer

## Reconstruction

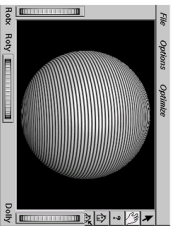
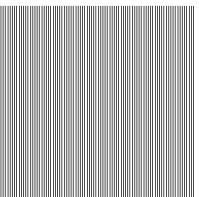
How to deal with:

- Pixels that are much larger than texels?**
- Apply filtering, "averaging"
- Pixels that are much smaller than texels?**
- Interpolate



© W. Habel and M. Beetz/Primer

## Reconstruction



(Image courtesy of Kiriakos Kutulakos, U Rochester)

© W. Habel and M. Beetz/Primer

## Texture Coordinate Interpolation

### Perspective correct interpolation

- $a, b, \gamma$ :
- Barycentric coordinates of a point  $P$  in a triangle
- $s_0, s_1, s_2$ :
- Texture coordinates of vertices
- $w_0, w_1, w_2$ :
- Homogeneous coordinates of vertices

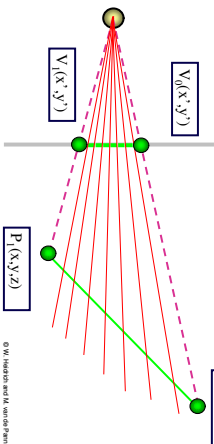
$$s = \frac{\alpha \cdot s_0 / w_0 + \beta \cdot s_1 / w_1 + \gamma \cdot s_2 / w_2}{\alpha / w_0 + \beta / w_1 + \gamma / w_2}$$

© W. Habel and M. Beetz/Primer

## Interpolation: Screen vs. World Space

### Screen space interpolation incorrect

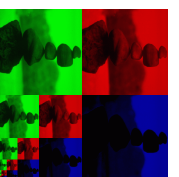
- Problem ignored with shading, but artifacts more visible with texturing



© W. Habel and M. Beetz/Primer

## MIPmap storage

### only 1/3 more space required

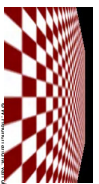
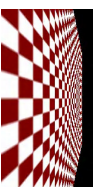


© W. Habel and M. Beetz/Primer

## MIPmaps

### Multum in parvo -- many things in a small place

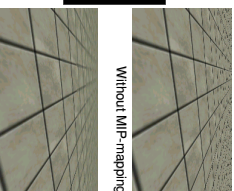
- Prespecify a series of prefiltered texture maps of decreasing resolutions
- Requires more texture storage
- Avoid shimmering and flashing as objects move
- glSubBuffer1d2DnKPrimage**
- Automatically constructs a family of textures from original texture size down to 1x1 without
- with



© W. Habel and M. Beetz/Primer

## MIPmapping

use "image pyramid" to precompute averaged versions of the texture

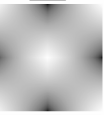
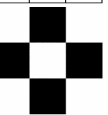


© W. Habel and M. Beetz/Primer

## Interpolating Textures

- Nearest neighbor
- Bilinear
- Hermite

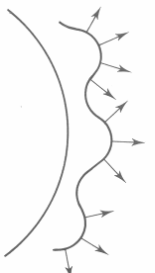
1	0	1
0	1	0
1	0	1



© W. Habel and M. Beetz/Primer

## Bump Mapping

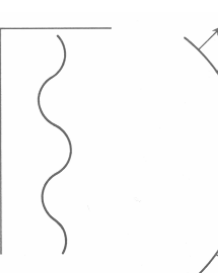
$O(u)$   
Lengthening or shortening  $O(u)$  using  $B(u)$



© W. Habel and M. Beetz/Primer

## Bump Mapping

$O(u)$   
Original surface

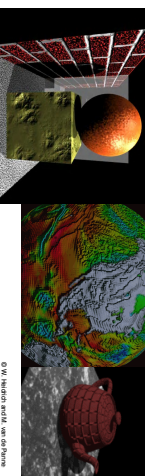


© W. Habel and M. Beetz/Primer

## Bump Mapping: Normals As Texture

Object surface often not smooth -- to recreate correctly need complex geometry model  
Can control shape "effect" by locally perturbing surface normal

- Random perturbation
- Directional change over region

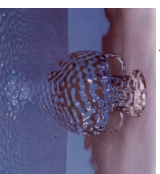


© W. Habel and M. Beetz/Primer

## Texture Parameters

### In addition to color can control other material/object properties

- Surface normal (bump mapping)
- Reflected color (environment mapping)



© W. Habel and M. Beetz/Primer

## Cube Mapping

### 6 planar textures, sides of cube

- Point camera in 6 different directions, facing out from origin

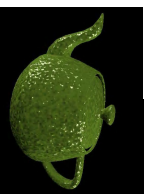


## Volumetric Bump Mapping

Marble



Bump

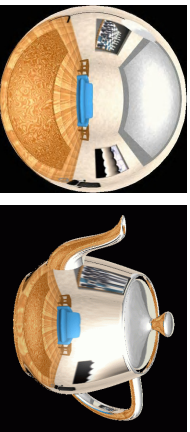


© W. Haddon and D. Shuttleworth

## Sphere Mapping

### Texture is distorted fish-eye view

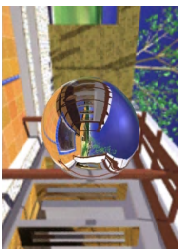
- Point camera at mirrored sphere
- Spherical texture mapping creates texture coordinates that correctly index into this texture map



## Environment Mapping

### Cheap way to achieve reflective effect

- Generate image of surrounding
- Map to object as texture



## Displacement Mapping

### Bump mapping gets silhouettes wrong

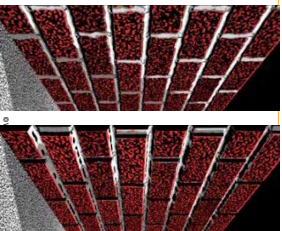
- Shadows wrong too

### Change surface geometry instead

- Need to subdivide surface

### GPU support

- Bump and displacement mapping not directly supported; require per-pixel lighting
- However, modern GPUs allow for programming both yourself

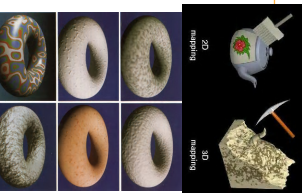


## Volumetric Texture

### Define texture pattern over 3D domain - 3D space containing the object

- Texture function can be digitized or procedural
- For each point on object compute texture from point location in space

### Common for natural material/irregular textures (stone, wood, etc...)



## Cube Mapping

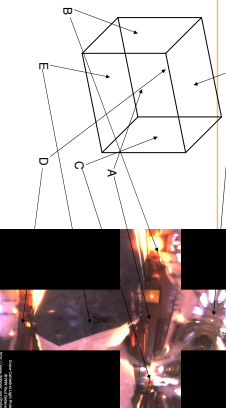
### Direction of reflection vector $r$ selects the face of the cube to be indexed

- Co-ordinate with largest magnitude
  - e.g., the vector  $(0.2, 0.5, -0.84)$  selects the  $-Z$  face
- Remaining two coordinates (normalized by the 3<sup>rd</sup> coordinate) selects the pixel from the face.
  - E.g.,  $(-0.2, 0.5)$  gets mapped to  $(0.38, 0.80)$ .

### Difficulty in interpolating across faces

© W. Haddon and D. Shuttleworth

## Cube Mapping



## Procedural Textures

### Several good explanations

- Text book Section 10.1 <http://www.noisemachine.com/talk1>
- [http://respaces.virgin.net/hugo\\_eblais/modelling\\_perturb.htm](http://respaces.virgin.net/hugo_eblais/modelling_perturb.htm)
- <http://www.cdo.nyu.edu/netcode/perturb-noise-math-faq.html>



## Procedural Textures

### Generate "image" on the fly, instead of loading from disk

- Often saves space
- Allows arbitrary level of detail

© W. Haddon and D. Shuttleworth

