

Bump mapping, Textures

To improve models, we use curved surfaces, procedural detail, and *textures*.

Textures and bump maps are processed in the pixel shader or fragment shader.

Bump mapping

perturb normal \mathbf{n} at point \mathbf{p} as if \mathbf{p} had been moved by d in the direction of the normal \mathbf{n} :

$$\mathbf{p}^{\text{new}} = \mathbf{p} + d\mathbf{n}.$$

Compute

$$\begin{aligned} \frac{\partial \mathbf{p}^{\text{new}}}{\partial u} \times \frac{\partial \mathbf{p}^{\text{new}}}{\partial v} &= \left(\frac{\partial \mathbf{p}}{\partial u} + \frac{\partial d}{\partial u} \mathbf{n} + d \frac{\partial \mathbf{n}}{\partial u} \right) \times \left(\frac{\partial \mathbf{p}}{\partial v} + \frac{\partial d}{\partial v} \mathbf{n} + d \frac{\partial \mathbf{n}}{\partial v} \right) \\ &= \mathbf{n} + \frac{\partial d}{\partial u} \mathbf{n} \times \frac{\partial \mathbf{p}}{\partial v} + \frac{\partial d}{\partial v} \mathbf{n} \times \frac{\partial \mathbf{p}}{\partial u} + O\left(\frac{\partial \mathbf{n}}{\partial u}, \frac{\partial \mathbf{n}}{\partial v}\right). \end{aligned}$$

NOTE NORMAL: $(N_u = n_u - N(n_u \cdot N)) / \|n\|$ where $n_u := p_{uu} \times p_v + p_u \times p_{uv}$. The leading terms of the perturbation of the normal are therefore

$$\frac{\partial d}{\partial u} \mathbf{n} \times \frac{\partial \mathbf{p}}{\partial v} + \frac{\partial d}{\partial v} \mathbf{n} \times \frac{\partial \mathbf{p}}{\partial u}.$$

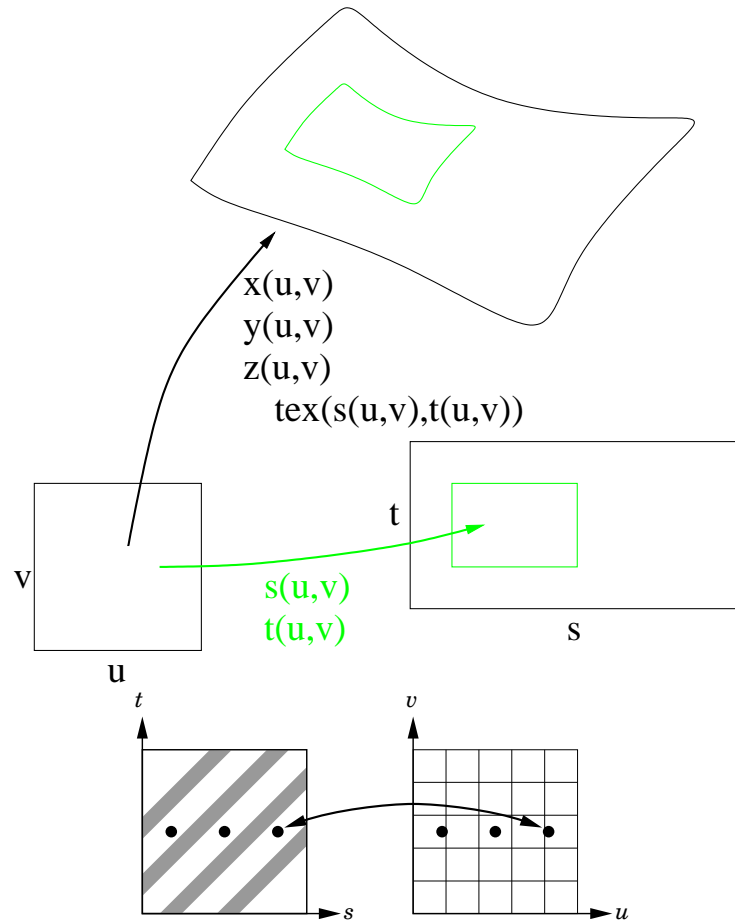
Texture types

- 2D texture : pasting an image onto a surface (challenges: distortion and aliasing)
- Intermediate cylinder, sphere texture. Transfer texture from an intermediate object (sphere or cylinder) for better *parametrization*; use the object normal.
- *environment map*, cube map:
place viewer at object center. Transfer resulting image as texture (possibly via intermediate object)
- video texture
- 3D texture — generated random, x,y,z direct, discrete grid

`glTexCoord 9texgen.c` `9checker.c`

Challenges of texturing

- distortion (flat to sphere)
- want pixel – so need map from screen coordinates to texture coordinates;
- areas, not points should be mapped **bilinear interpolation**
- aliasing (Moiré pattern) – pointwise: might miss, average: smears out



9projtext.pdf

Texture details

- many bit patterns (formats) (gimp exports C-arrays!)
- texture wrapping

- GL_LINEAR, GL_NEAREST
- mipmapping, `9mipmap.c`
- texture shading: (RGB) pp403
 - replace* (color=tex(texture), alpha=frag(ment))
 - modulate* (color=frag*tex, alpha=frag)
 - decal* (color=tex, alpha=frag)
 - blend* (color=frag*(1-tex)+tex*environment color, alpha=frag)

Attach texture to a patch: `10texturesurf.c`

- create texture: `glTexImage2D*(GL_TEXTURE_2D, LOD=0, internalformat=GL_R3_G3_B2, width=2m, height, border, input_format, type=ubyte, texels)` `gluScaleImage` (to 2^m), `glCopyTexImage` (from framebuffer), texture proxy (to determine whether resources suffice), *mipmaps* and *LOD*, `glGenTextures`, `glBindTexture`
- how to apply texture: `glTexEnv*({decal,replace,modulate,blend})` see texture functions
- `glEnable`
- `glTexCoord*`, `glVertex*`,