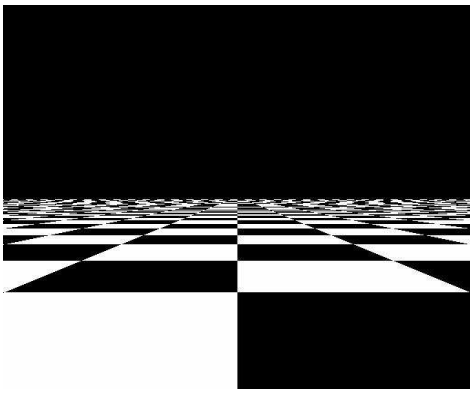
Texture and other Mappings

Texture Mapping
Bump Mapping
Displacement Mapping
Environment Mapping

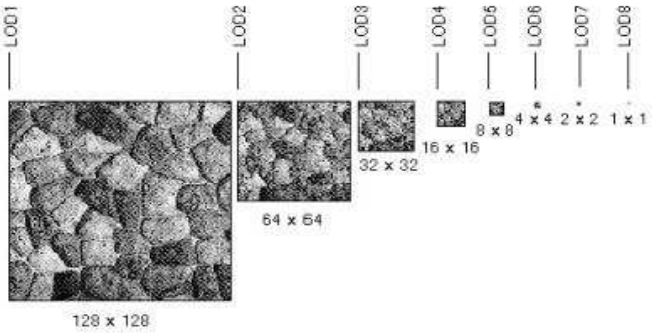
Example: Checkerboard

- Particularly severe problems in regular textures



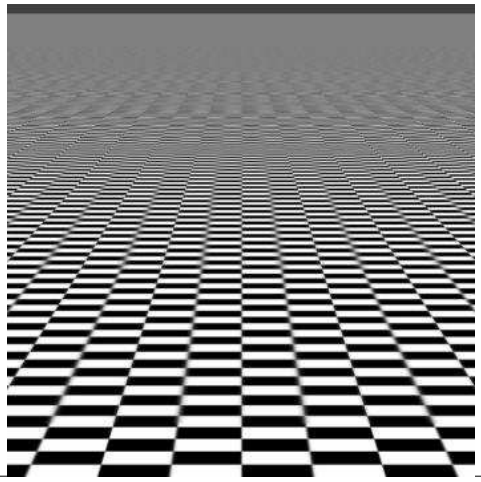
The Beginnings of a Solution: Mipmapping

- Pre-calculate how the texture should look at various distances, then use the appropriate texture at each distance. This is called *mipmapping*.
- "Mip" → "multum in parvo" or "many things in a small place"
- Each mipmap (each image below) represents a level of resolution.
- Powers of 2 make things much easier.



The Beginnings of a Solution

- Problem: Clear divisions between different depth levels
- Mipmapping alone is unsatisfactory.



Another Component: Filtering

- *Anisotropic filtering*
  - Basic filtering methods assume that a pixel on-screen maps to a square (isotropic) region of the texture
  - For surfaces tilted away from the viewer, this is not the case!

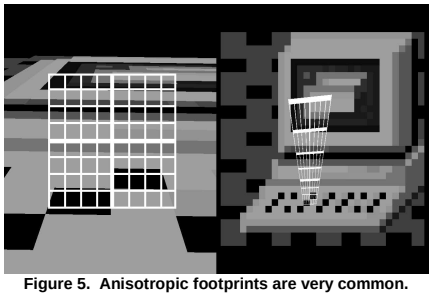
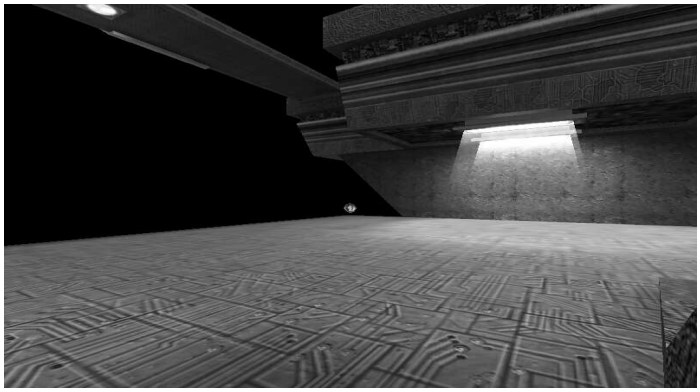


Figure 5. Anisotropic footprints are very common.

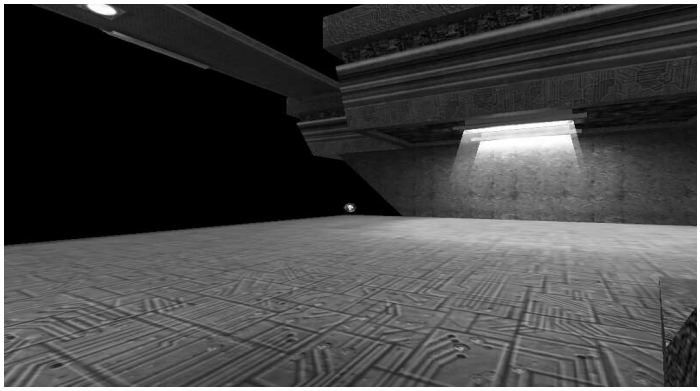
Image courtesy of nVidia

Bilinear Filtering

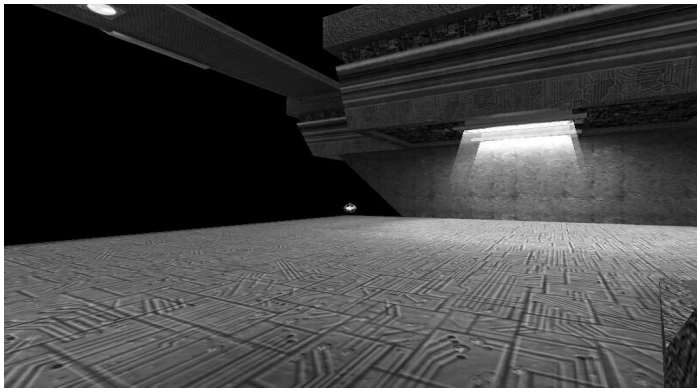


ID Software

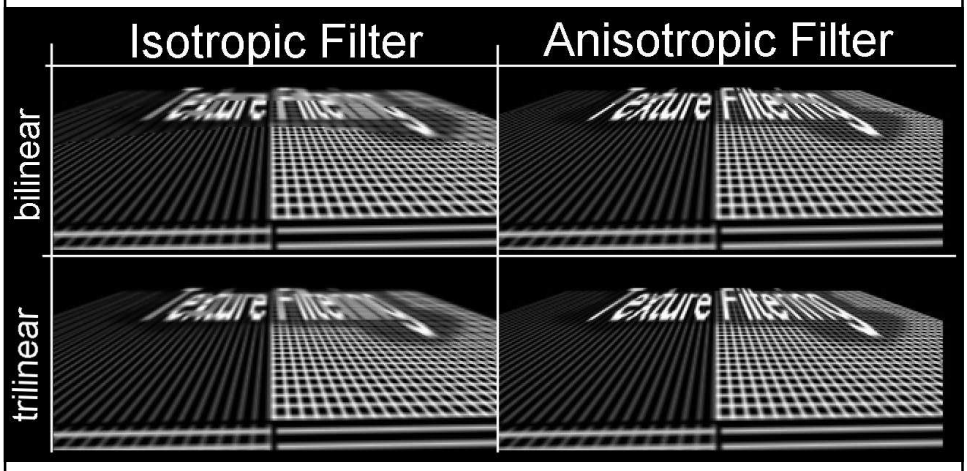
Trilinear Filtering



Anisotropic Filtering

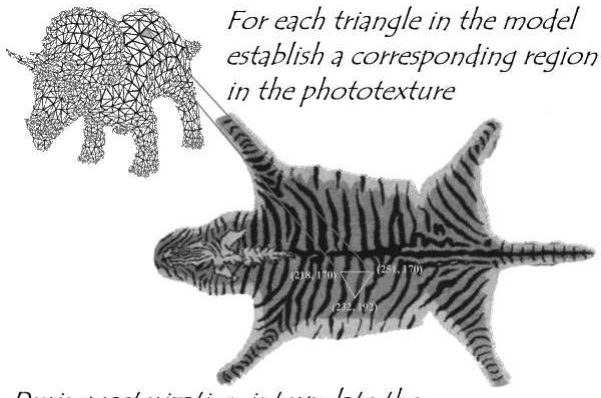


Side-by-Side Comparison



nVidia

Or design the mapping by hand

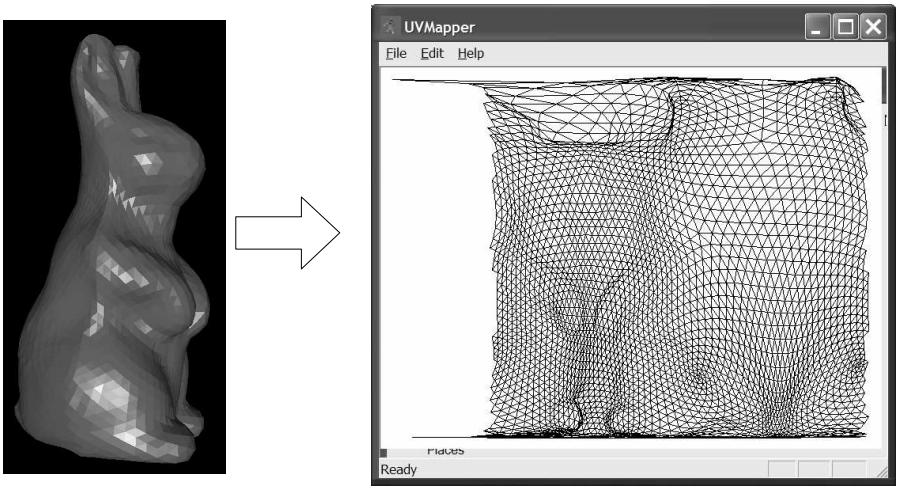


For each triangle in the model establish a corresponding region in the phototexture

During rasterization interpolate the coordinate indices into the texture map

Demo: “uvMapper”

- www.uvmapper.com

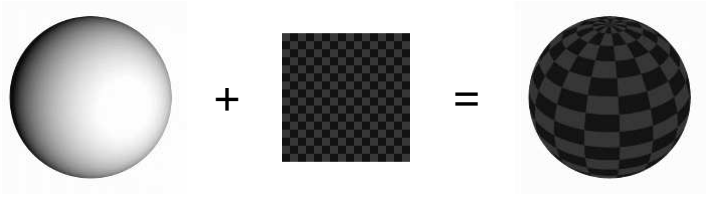


Uses for Texture Mapping

Use texture to affect a variety of parameters

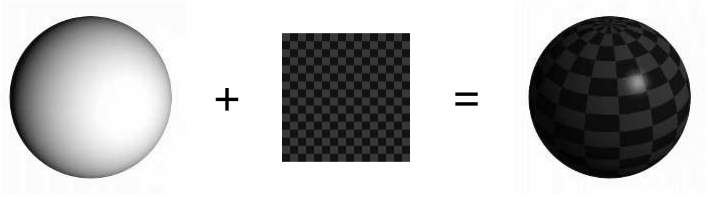
- surface color (Catmull 1974) - color (radiance) of each point on surface
- surface reflectance - reflectance coefficients k_d, k_s, or n_{shiny}
- normal vector - bump mapping (Blinn 1978)
- geometry - displacement mapping
- transparency - transparency mapping (clouds) (Gardener 1985)
- light source radiance - environment mapping (Blinn 1978)

Radiance vs. Reflectance Mapping



Sphere w/ Uniform Diffuse coefficient Radiance Map Sphere w/ Radiance Map

Texture specifies (isotropic) radiance for each point on surface

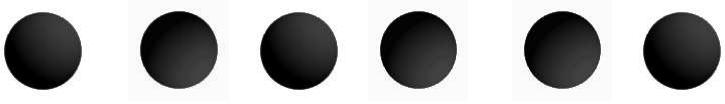


Sphere w/ Uniform Diffuse coefficient Reflectance (k_d) Map Sphere w/ Reflectance Map

Texture specifies diffuse color (k_d coefficients) for each point on surface
- three coefficients, one each for R, G, and B radiance channels

Bump Mapping: A Dirty Trick

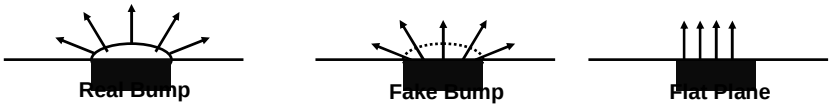
- Which spots bulge out, and which are indented?



- Answer: None! This is a flat image.
- The human visual system is hard-coded to expect light from above
- In CG, we can perturb the normal vector without having to make any actual change to the shape.

Bump Mapping

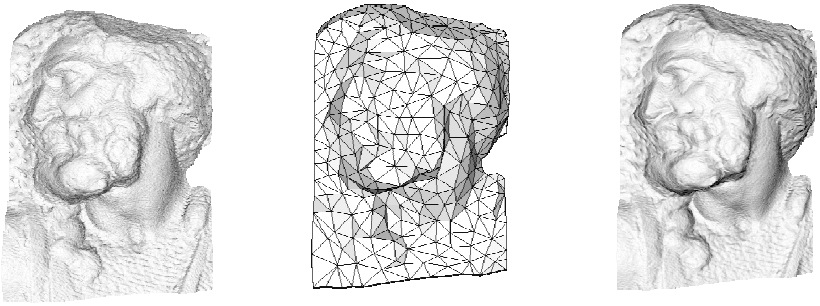
- Basic texture mapping paints on to a smooth surface
- How do you make a surface look *rough*?
  - Option 1: model the surface with many small polygons
  - Option 2: perturb the normal vectors before the shading calculation



Sphere w/Diffuse Texture Map | Bump Map | Sphere w/Diffuse Texture + Bump Map

Bump Mapping

- We can perturb the normal vector without having to make any actual change to the shape.
- This illusion can be seen through—how?

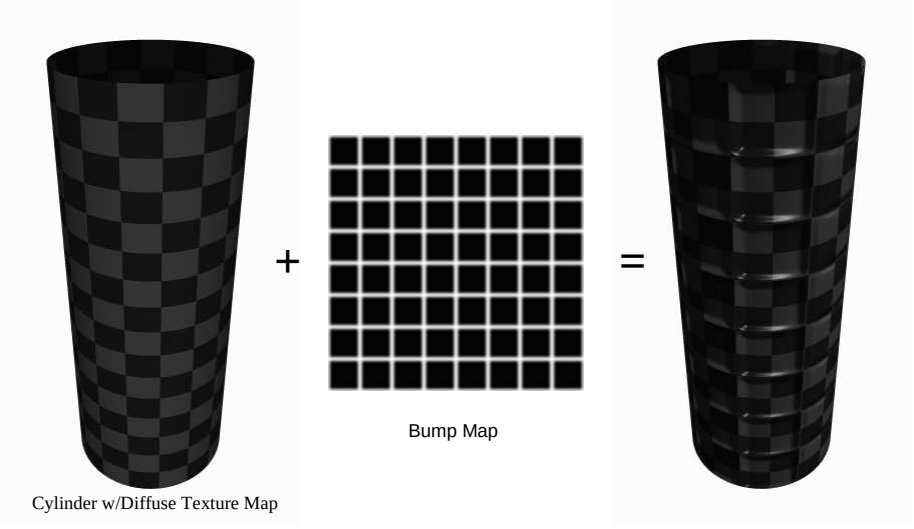


Original model (5M) | Simplified (500) | Simple model with bump map

Bump Mapping

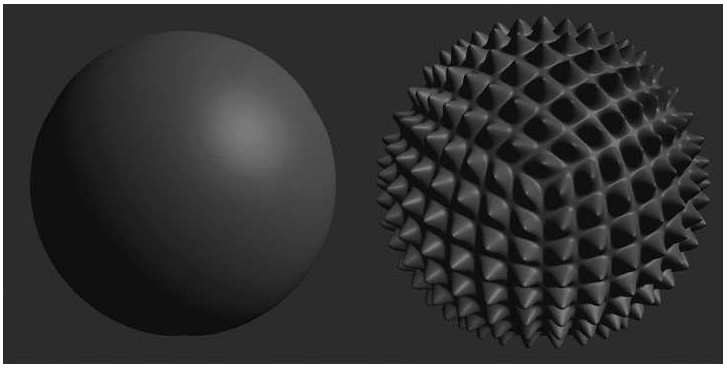
Greg Turk

Another Bump Mapping Example



Displacement Mapping

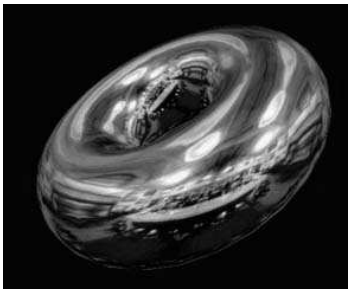
- Use texture map to displace each point on the surface
  - Texture value gives amount to move in direction normal to surface



- How is this different from bump mapping?

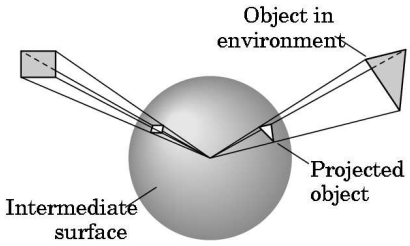
Environment Mapping

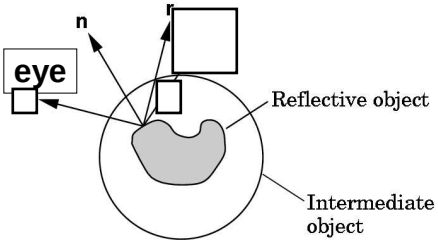
Specular reflections that mirror the environment



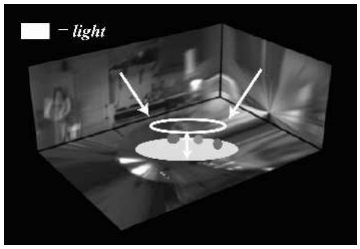
Environment Mapping

Specular reflections that mirror the environment

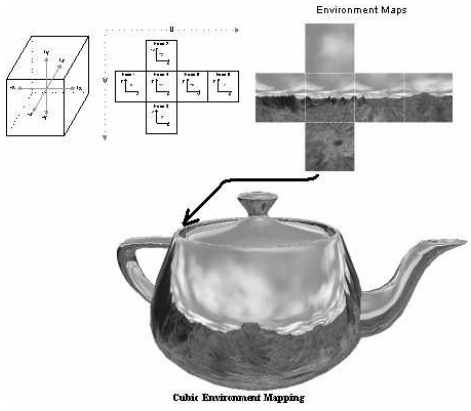




Cube is a natural intermediate object for a room



Environment Mapping: Cube Maps

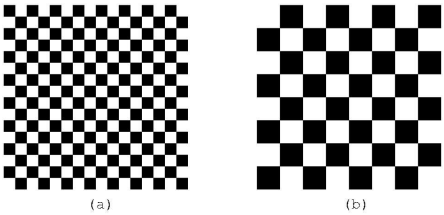


Basics of Texture Mapping in OpenGL

```
Glubyte my_texels[512][512][3];
Gluint texID;

glGenTextures(1, &texID);
glBindTexture(GL_TEXTURE_2D, texID);
glTexImage2D(GL_TEXTURE_2D, 0, GL_RGB, 512, 512, 0,
             GL_RGB,GL_UNSIGNED_BYTE, my_texels);
/* level, components, w, h, border, format, type, tarray */

/* assign texture coordinates */
glEnable(GL_TEXTURE_2D);
glBegin(GL_QUAD);
       glTexCoord2f(0.0, 0.0);
       glVertex3f(x1,y1,z1);
       glTexCoord2f(1.0, 0.0);
       glVertex3f(x2,y2,z2);
       glTexCoord2f(1.0,1.0);
       glVertex3f(x3,y3,z3);
       glTexCoord2f(0.0,1.0);
       glVertex3f(x4,y4,z4);
glEnd();
glDisable(GL_TEXTURE_2D);
```



Grungy details we've ignored

- Specify s or t out of range? Use **`GL_TEXTURE_WRAP`** in **`glTexParameter`** because many textures are carefully designed to repeat

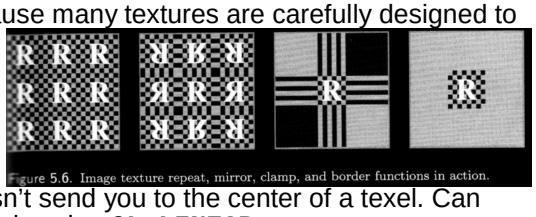
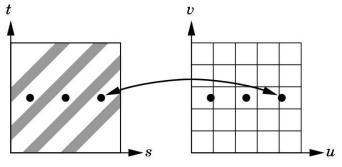


Figure 5.6. Image texture repeat, mirror, clamp, and border functions in action.

- Aliasing? Mapping doesn't send you to the center of a texel. Can average nearest 2x2 texels using **`GL_LINEAR`**

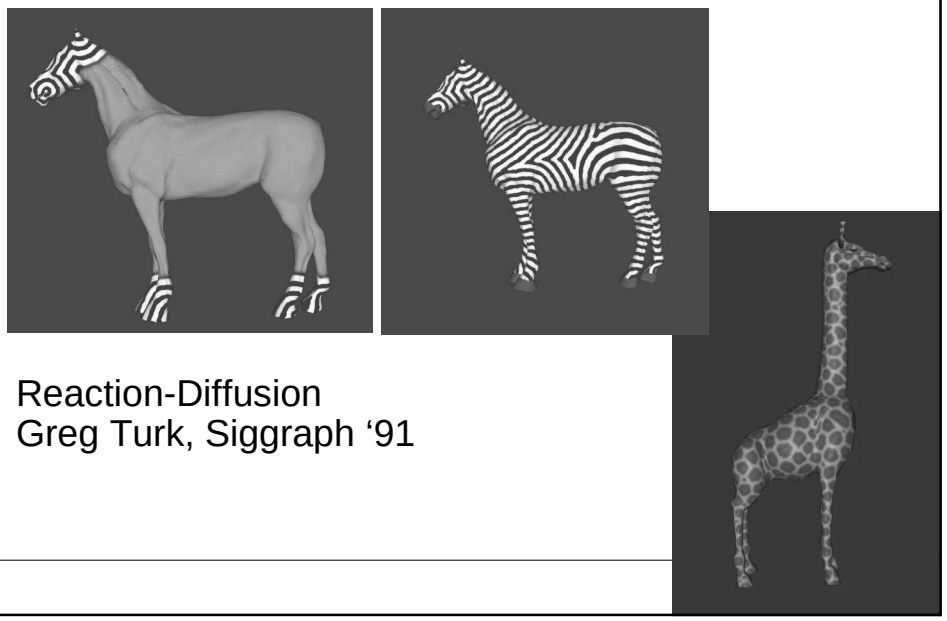


- Mipmapping: use textures of varying resolutions. 64x64 becomes 32x32,16x16,8x8,4x4,2x2 and 1x1 arrays with **`gluBuild2Dmipmaps`**

Texture Generation

Photographs
Drawings
Procedural methods (2D or 3D)
- Associate each x,y,z value directly with an s,t,r value in the texture block (sculpting in marble and granite)

Procedural Methods



Reaction-Diffusion
Greg Turk, Siggraph '91

Solid Textures

- Have a 3-D array of texture values (e.g., a block of marble)
  - Use a function [xyz] -> [RGB] to map colors to points in space
- Such a 3D map is called a *solid texture map*
- In practice the map is often defined procedurally
  - No need to store an entire 3D array of colors
  - Just define a function to generate a color for each 3D point
- The most interesting solid textures are random ones
  - a great marble algorithm has now become cliché
- Evaluate the texture coordinates in object coordinates - otherwise moving the object changes its texture!

From: *An Image Synthesizer* by Ken Perlin, SIGGRAPH '85