Course Updates

- Midterm statistics
  - **Average:** 59
  - **Standard Deviation:** 19
  - Solutions will be on the web today

- Assignment 2
  - **Theory** is due today by midnight
  - **Programming** is due a week from today (if you haven't started, you should).

Last time …

- We talked about shading
  - Flat
  - Gouraud
  - Phong
- Started talking about texture mapping

Texture Mapping
Part 2

Computer Graphics, CSCD18

Fall 2007

Instructor: Leonid Sigal

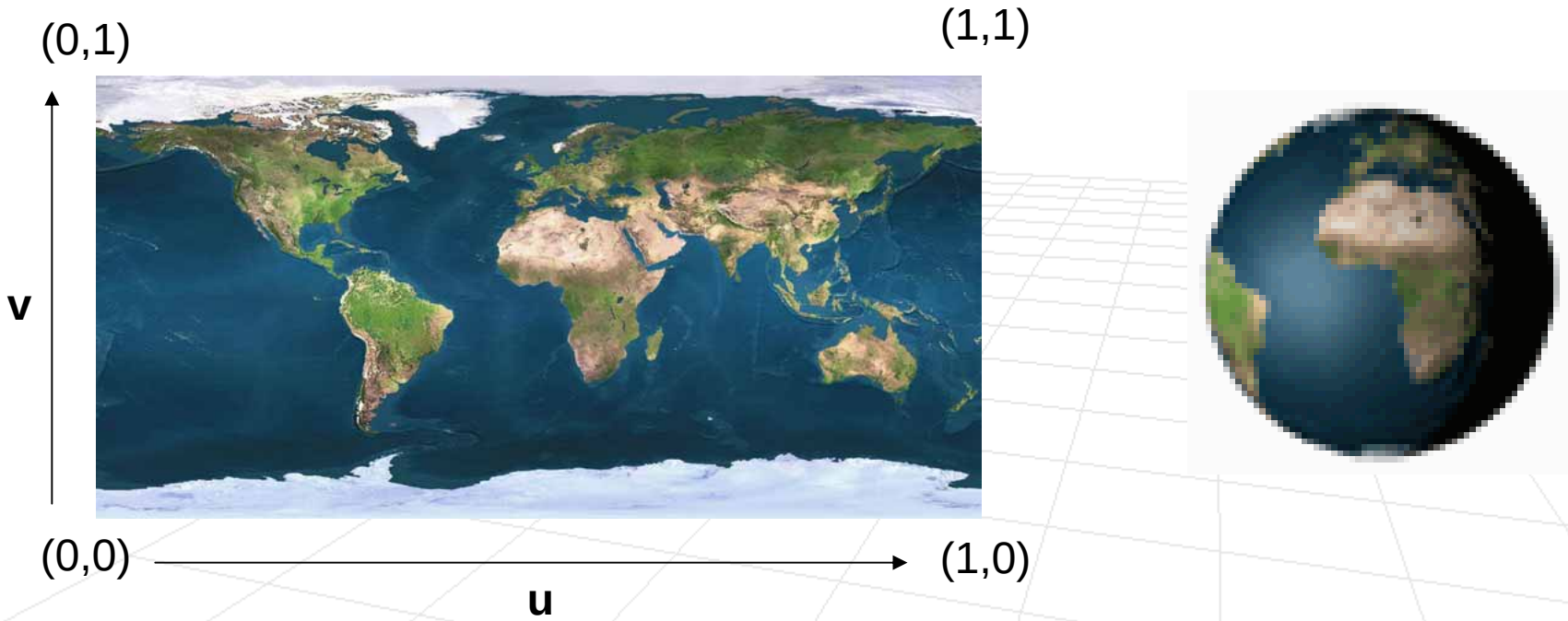
Questions we must address

- Where do textures come from?
- How do we map texture onto a surface?
- How does texture change reflectance properties and shading of the surface
- Scan conversion (how do we actually render texture mapped surface)

Mapping a texture onto a surface

- We can also use digital images as textures
  - Imagine gluing a 2D picture over a 3D surface

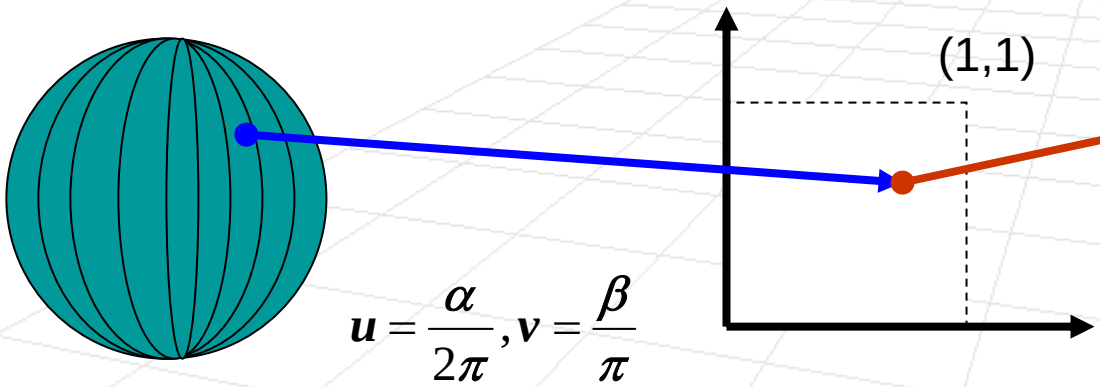
- How do we do this?
  - map a point on the arbitrary geometry to a point on an abstract unit square (we call this texture space)
  - map a point on abstract unit square to a point on the image of arbitrary dimension



Mapping a texture onto a sphere

- Simplest approaches to texture mapping
  - For each face of the mesh, specify a point (u_i, v_i) for each vertex point p_i
  - Continuous mapping from parametric form of the surface onto texture, for example for sphere

$$s(\alpha,\beta) = \begin{bmatrix} x_0 + r\cos\alpha\sin\beta \\ y_0 + r\sin\alpha\sin\beta \\ z_0 + r\cos\beta \end{bmatrix}, \quad \begin{matrix} 0 \le \alpha \le 2\pi \\ 0 < \beta \le \pi \end{matrix}$$



512 x 512

What about color texture map?

- Texture albedo is really a body property, rather then a surface property (like color)

- Assuming that the texture values are $0 \leq \tau \leq 1$ (we can achieve this by normalizing intensities of the texture map image), we can simply scale the reflection coefficients of ambient and diffuse components of the Phong model accordingly

$$\tilde{\boldsymbol{r}}_d = \tau \boldsymbol{r}_d$$

$$\tilde{\boldsymbol{r}}_a = \tau \boldsymbol{r}_a$$

- We could also similarly modulate the secular reflectance coefficient as well

What about color texture map?

- Texture albedo is really a body property, rather then a surface property (like color)

- Assuming that the texture values are $0 \le \tau \le 1$ (we can achieve this by normalizing intensities of the texture map image), we can simply scale the reflection coefficients of ambient and diffuse components of the Phong model accordingly

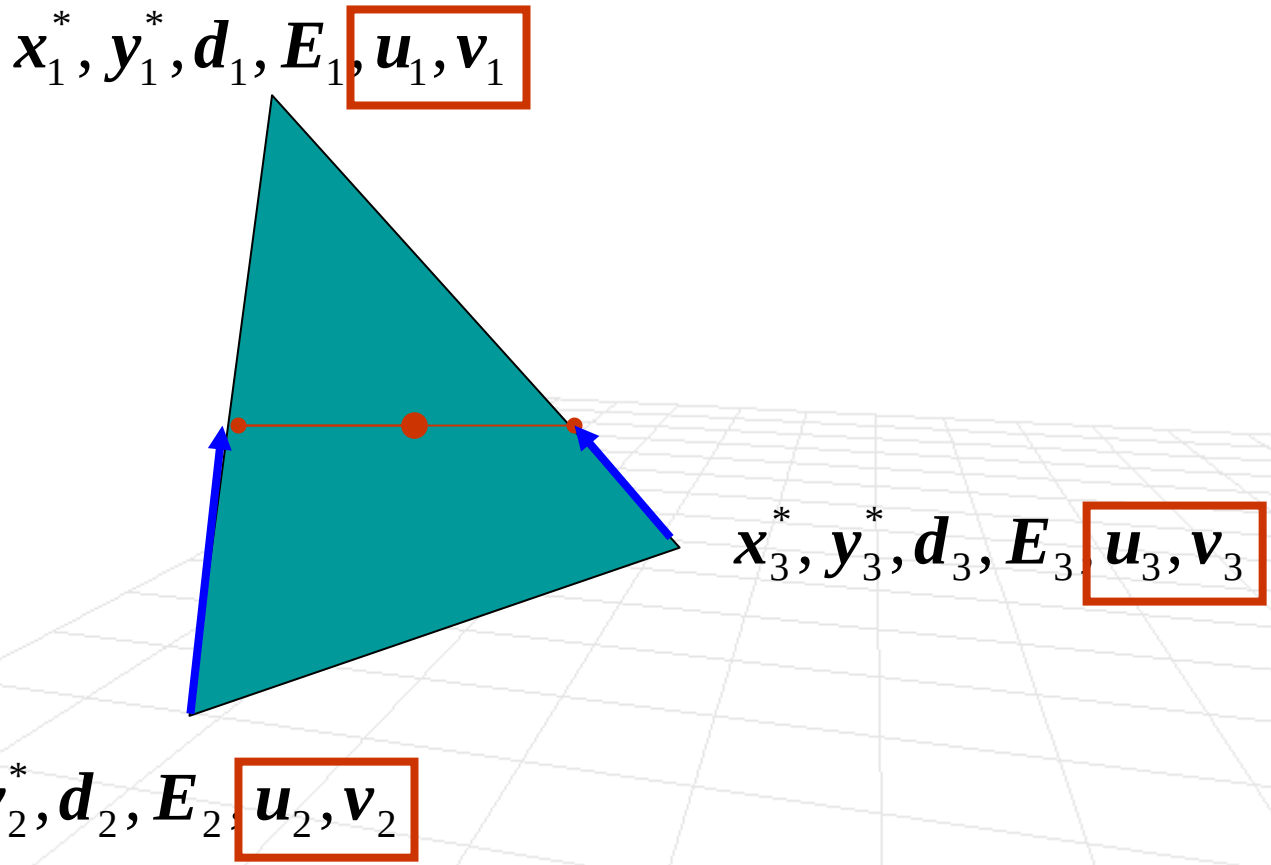
$$\tilde{r}_{d,R} = \tau_R r_{d,R} \quad \tilde{r}_{d,G} = \tau_G r_{d,G} \quad \tilde{r}_{d,B} = \tau_B r_{d,B}$$

$$\tilde{r}_{a,R} = \tau_R r_{a,R} \quad \tilde{r}_{a,G} = \tau_G r_{a,G} \quad \tilde{r}_{a,B} = \tau_B r_{a,B}$$

- We could also similarly modulate the secular reflectance coefficient as well

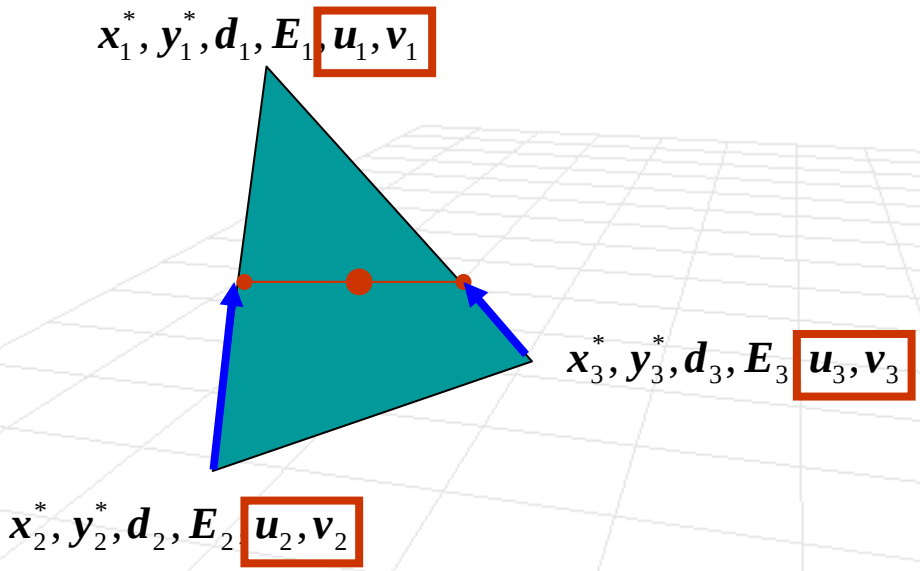
Scan Conversion with Texture Mapping

- Let's try extending the scan conversion algorithm from last class



Scan Conversion with Texture Mapping

- Let's try extending the scan conversion algorithm from last class
  - Linearly interpolate u, v along with radiance and pseudodepth
  - Scale radiance according to the texture map values



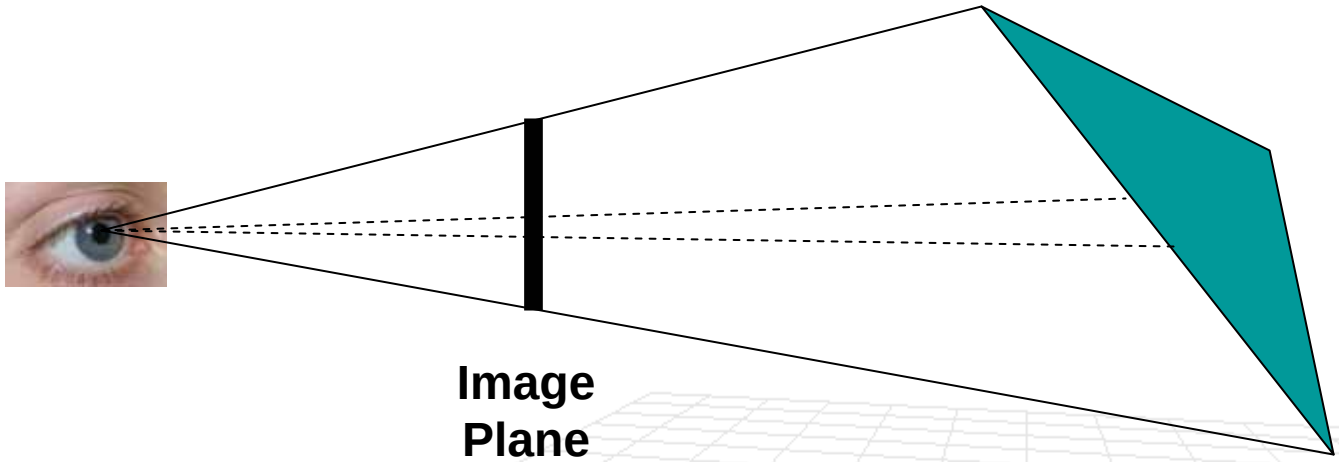
Simple Scan Conversion with Textures

- Let's try extending the scan conversion algorithm from last class
  - Linearly interpolate ***u***, ***v*** along with radiance and pseudodepth
  - Scale radiance according to the texture map values

if (d < z-buffer(x, y))
    use current u,v to index into the texture map
    to get texture value τ
    scale the radiance value $\tilde{E} = \tau E$
    putpixel(x, y,$\tilde{E}$)
    z-buffer(x, y) = d
end

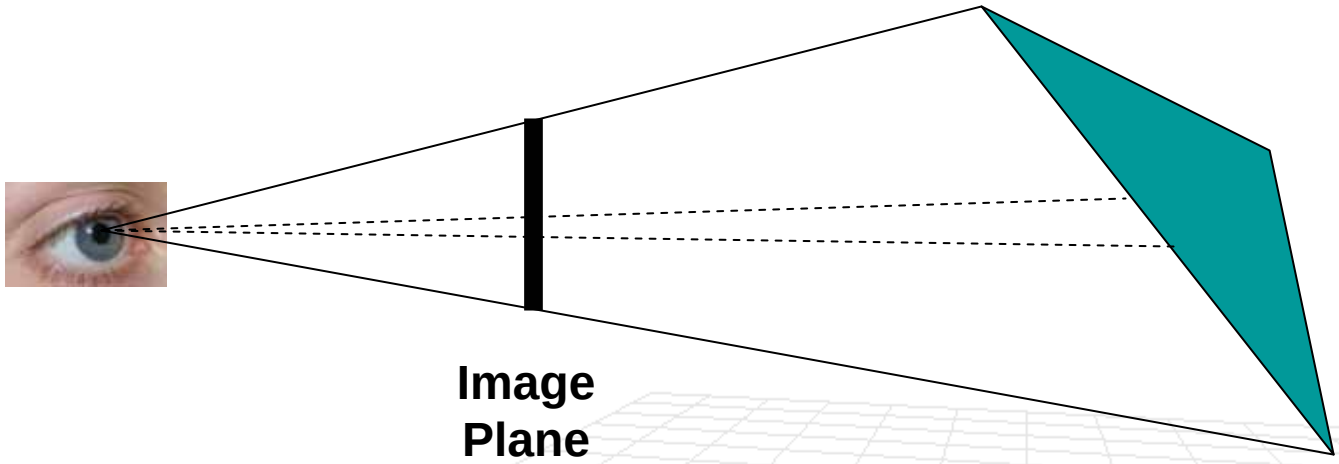
Problems with Simple Scan Conversion

- Perspective projection is non-linear
  - Lines map to lines
  - But, mid-point is not necessarily maps to mid-point



Problems with Simple Scan Conversion

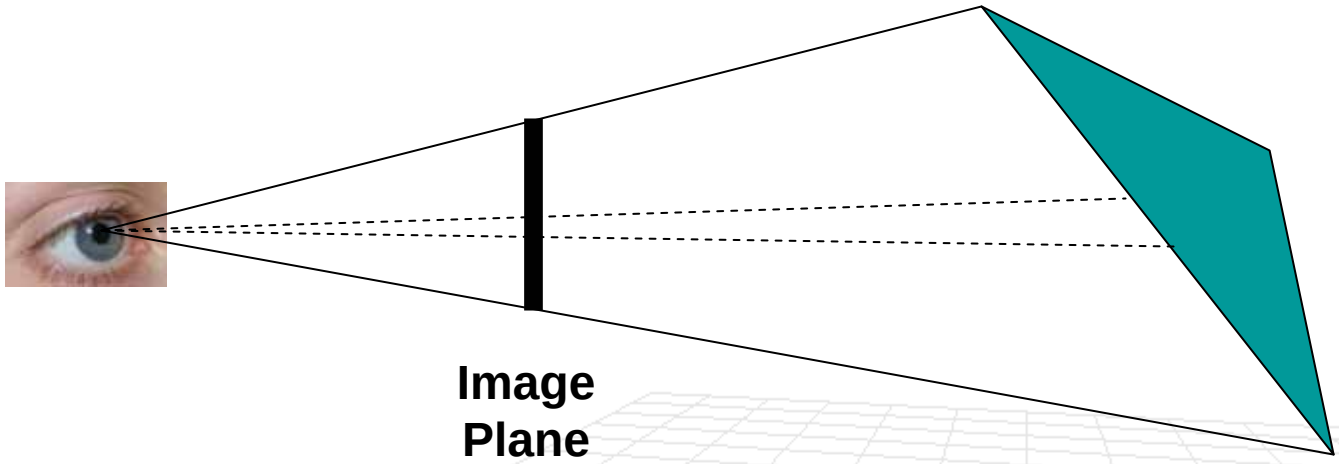
- Perspective projection is non-linear
  - Lines map to lines
  - But, mid-point is not necessarily maps to mid-point



- So, distortion depends on the slope of the surface with respect to line of sight
- **Why did we not care about this before?**

Problems with Simple Scan Conversion

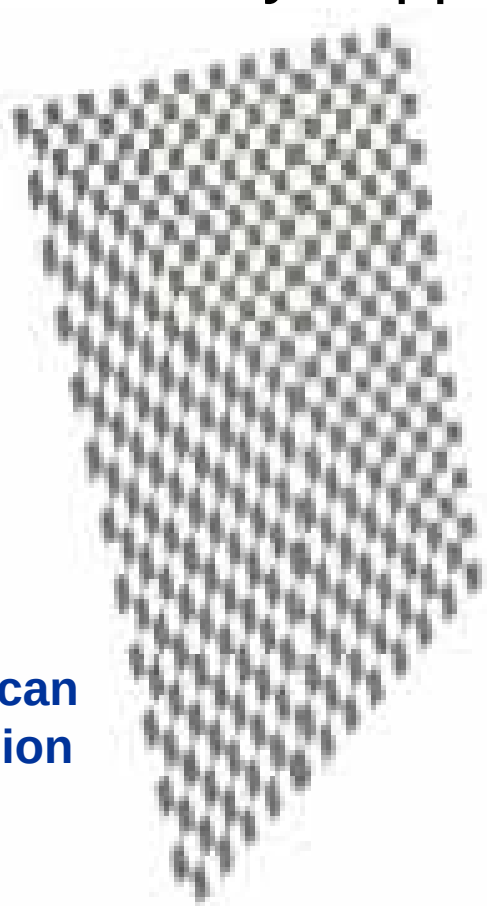
- Perspective projection is non-linear
  - Lines map to lines
  - But, mid-point is not necessarily maps to mid-point



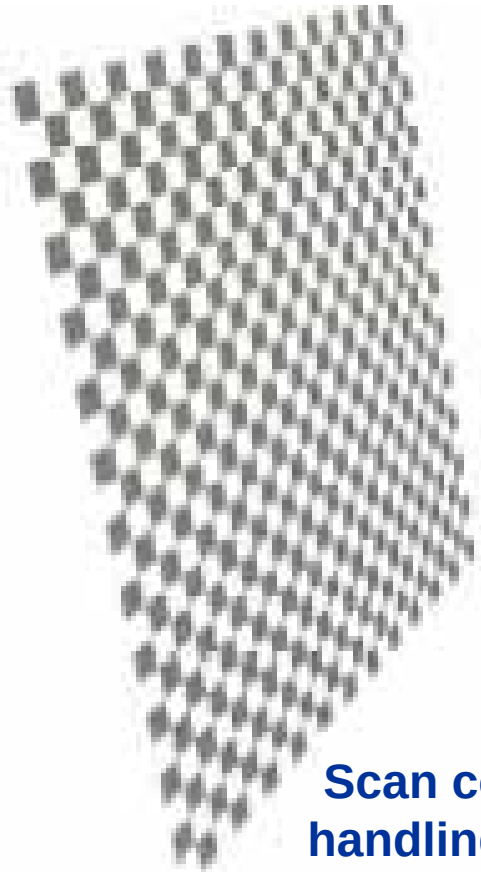
- So, distortion depends on the slope of the surface with respect to line of sight
  - It is evident only during animation and for textures with straight lines

Problems with Simple Scan Conversion

- Does this really happen in practice?



Simple scan conversion

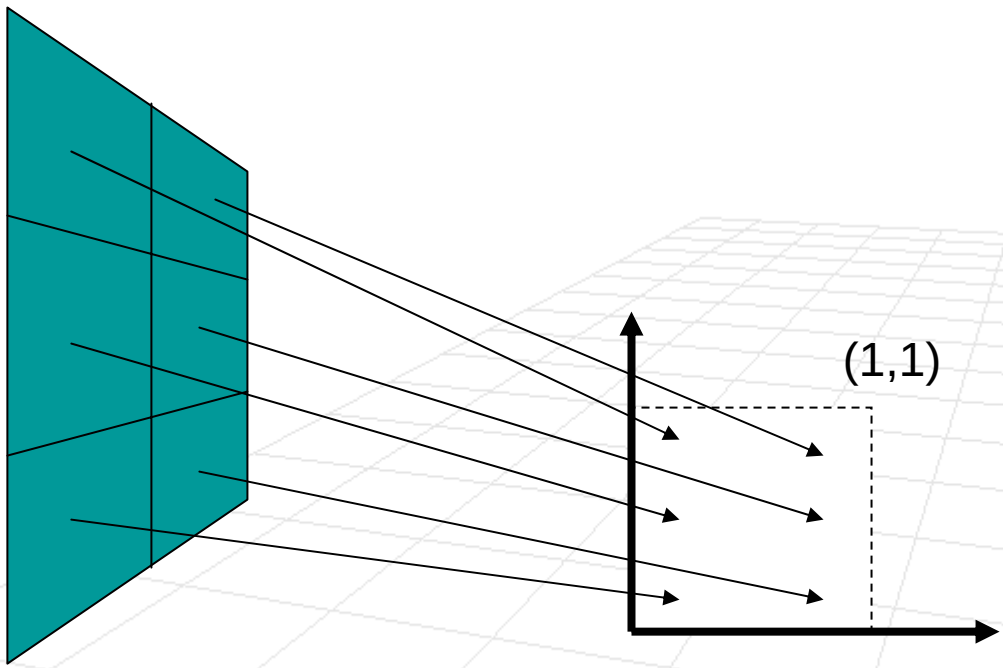


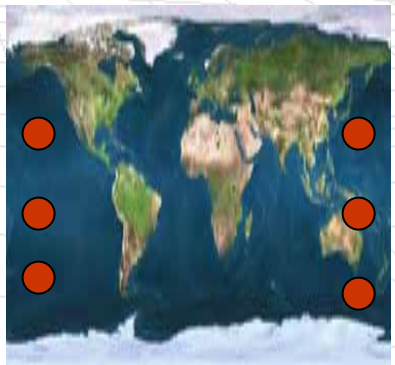
Scan conversion with handling of perspective

- We need to handle perspective effects during scan conversion (more complicated)

Aliasing

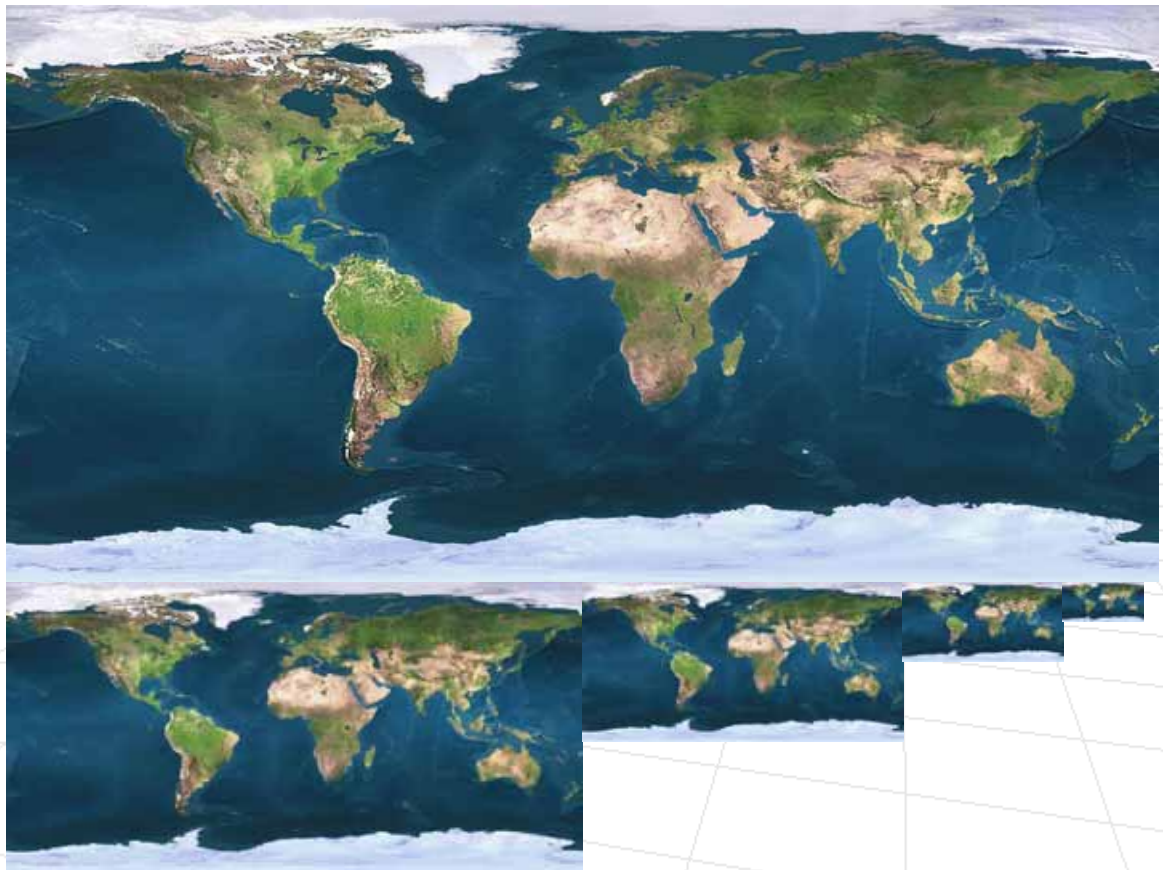
- **Another Problem:** When adjacent pixels in the image plane are rendered, the corresponding coordinates in the texture can be far apart (if the object is far away and we have high resolution texture) and sampling artifacts can be seen.





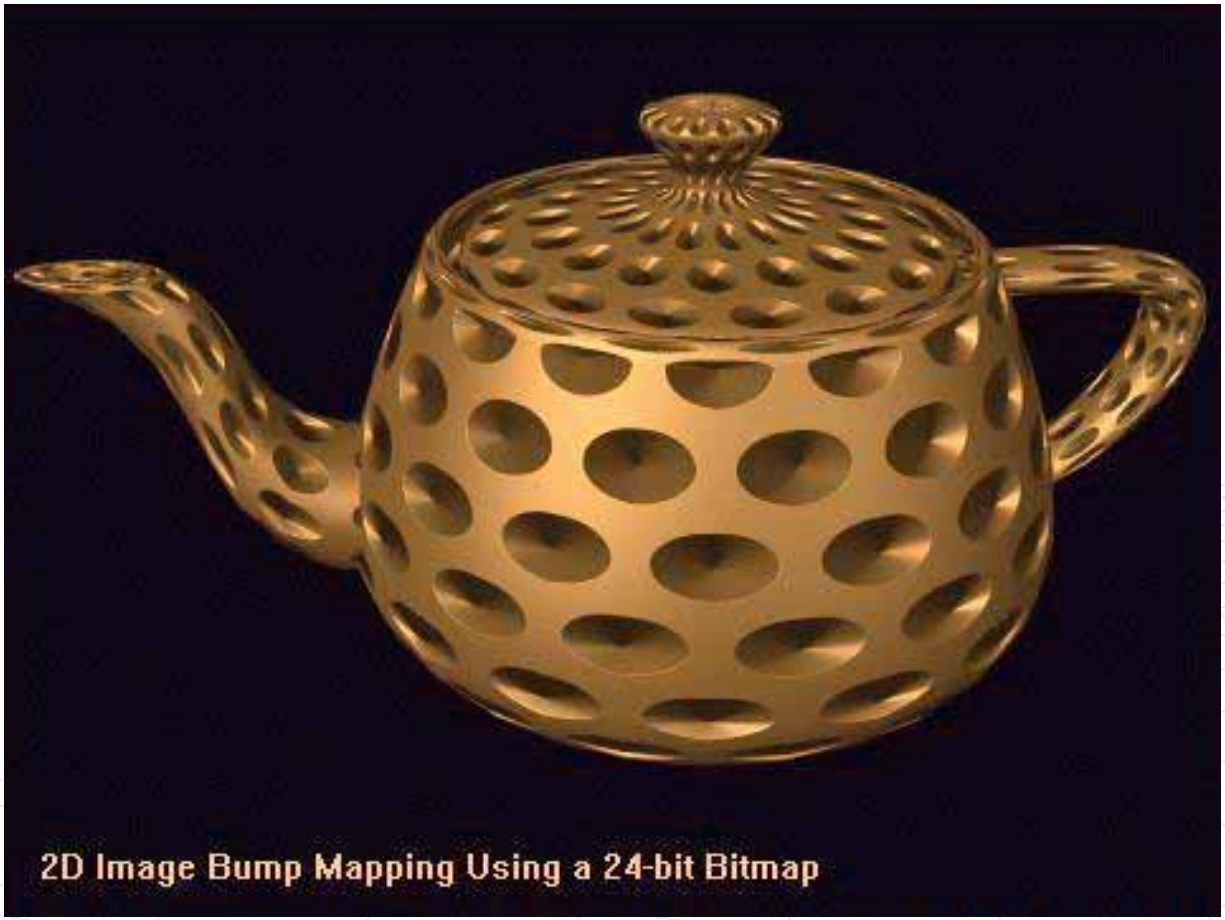
Mipmapping

- **Solution:** use high resolution texture for rendering objects that are close, and low-resolution texture when the object is far away



Bump Mapping

- **Idea:** Instead of perturbing reflectance properties, why don't we perturb the normals? **What's the difference?**

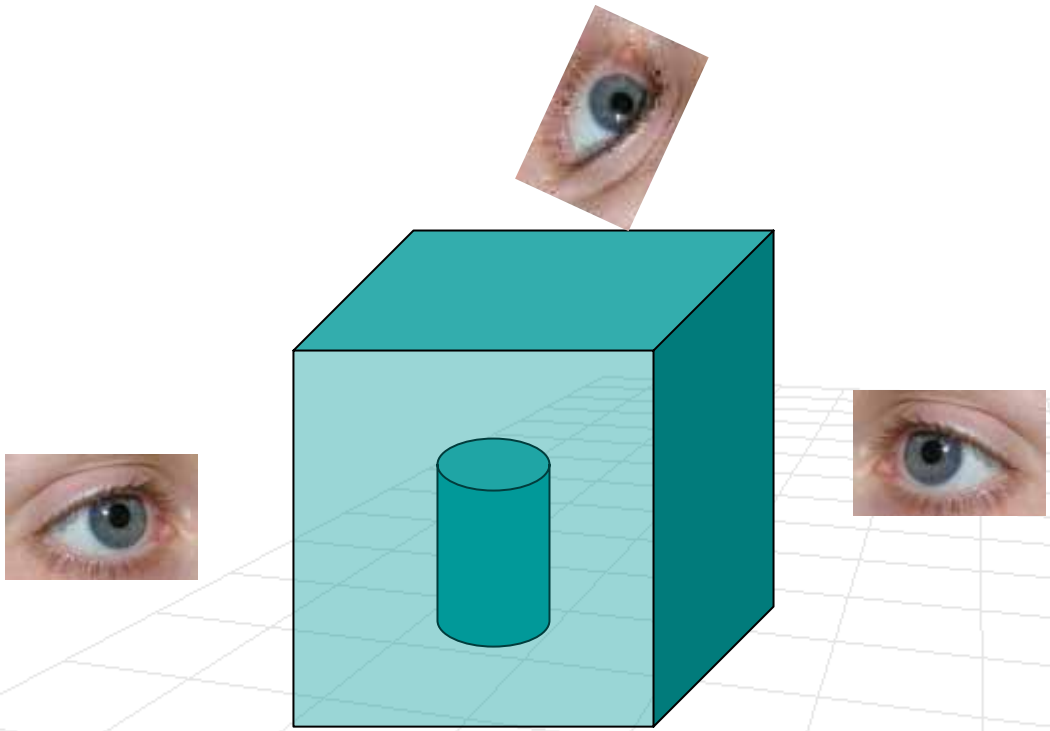


Bump Mapping

Texture Mapping **Bump Mapping**

Environment Mapping

- **Idea:** Use a texture map that corresponds to the view of the environment to achieve the effect of reflectance in a given object.



Cube environment mapping

Environment Mapping

- **Idea:** Use a texture map that corresponds to the view of the environment to achieve the effect of reflectance in a given object.

Image from slides by Aditi Majumder

Environment Mapping

- **Idea:** Use a texture map that corresponds to the view of the environment to achieve the effect of reflectance in a given object.

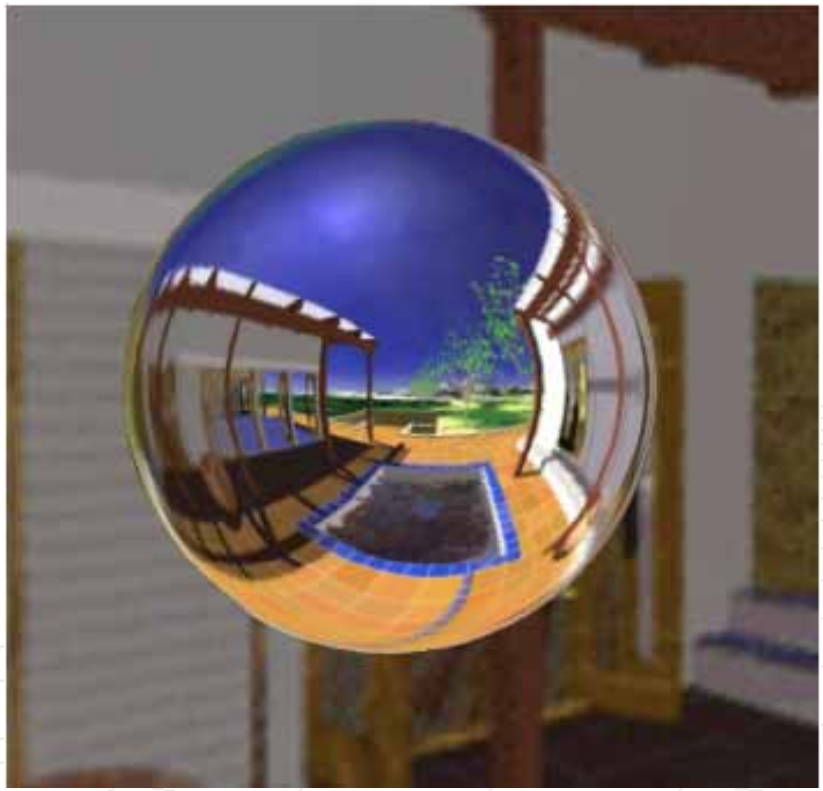


Image from slides by Aditi Majumder