Texture Mapping

1

Limits of Geometric Modeling

- Although graphics cards can render over 10 million polygons per second, that number is insufficient for many phenomena
  - Clouds
  - Grass
  - Terrain
  - Skin

2

Modeling an Orange

- Consider the problem of modeling an orange
- Start with an orange-colored sphere
  - Too simple
- Replace sphere with a more complex shape
  - Does not capture surface characteristics (small dimples)
  - Takes too many polygons to model all the dimples

3

Modeling an Orange

- Take a picture of a real orange, scan it, and “paste” onto simple geometric model
  - This process is texture mapping
- Still might not be sufficient because resulting surface will be smooth
  - Need to change local shape
  - Bump mapping

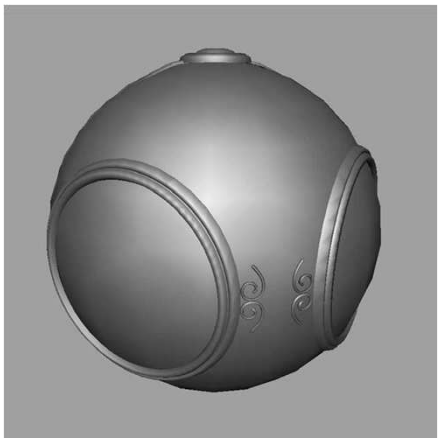
4

Three Types of Mapping

- Texture Mapping
  - Uses a pattern (or texture) to determine the color of a fragment
- Environmental Mapping
  - Allows us to create images that have the appearance of reflected materials without our having to trace reflected rays. In this technique, an image of the environment is painted onto the surface as that surface is being rendered
- Bump mapping
  - Distorts the normal vectors during the shading process to make the surface appear to have small variations in shape, such as the bumps on a real orange

5

Texture Mapping



No texture

Texture

6

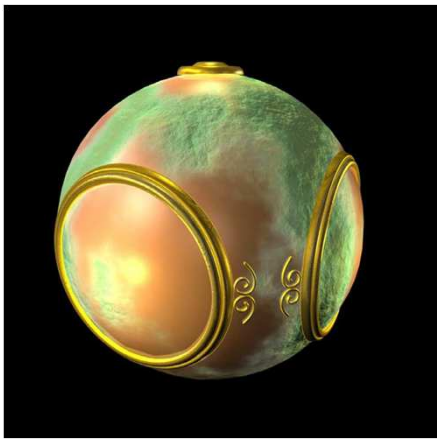
Environment Mapping

- Mapping environmental information as textures to the geometry

7

Bump Mapping

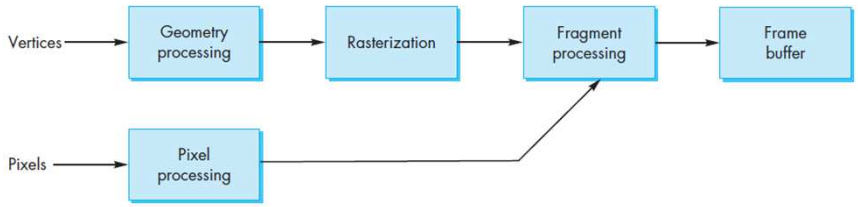
- Perturbing normals for vertices and then apply shading (Phong) model



8

Where does mapping take place?

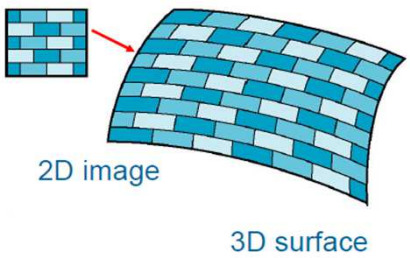
- Mapping techniques are implemented at the end of the rendering pipeline
  - Very efficient because few polygons pass down the geometric pipeline



9

Is it simple?

- Although the idea is simple - map an image to a surface - there are 3 or 4 coordinate systems involved 2D



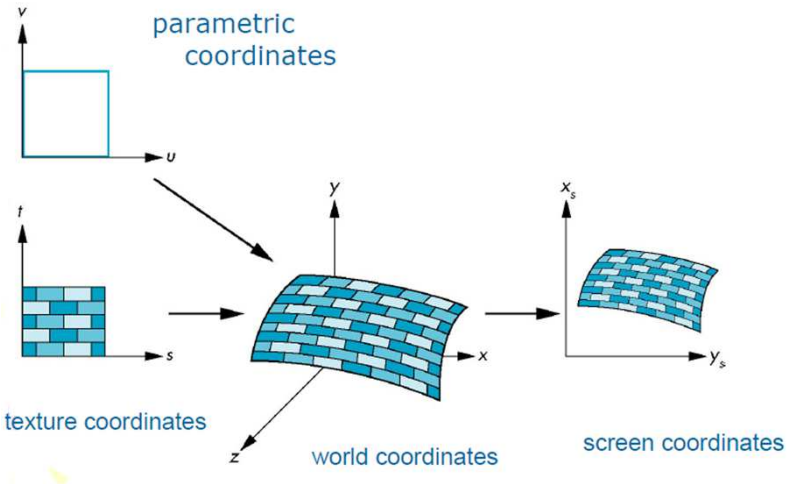
10

Coordinate Systems

- Parametric coordinates
  - May be used to model curved surfaces
- Texture coordinates
  - Used to identify points in the image to be mapped
- World Coordinates
  - Conceptually, where the mapping takes place
- Screen Coordinates
  - Where the final image is really produced

11

Texture Mapping

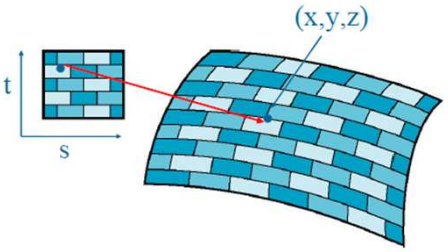


12

Mapping Functions

- Basic problem is how to find the maps
- Consider mapping from texture coordinates to a point on a surface
- Appear to need three functions

$$x = x(s,t)$$
$$y = y(s,t)$$
$$z = z(s,t)$$



- But we really want to go the other way

13

Backward Mapping

- We really want to go backwards
  - Given a pixel, we want to know to which point on an object it corresponds
  - Given a point on an object, we want to know to which point in the texture it corresponds
- Need a map of the form

$$s = s(x,y,z)$$
$$t = t(x,y,z)$$

- Such functions are difficult to find in general: "parameterization" problem

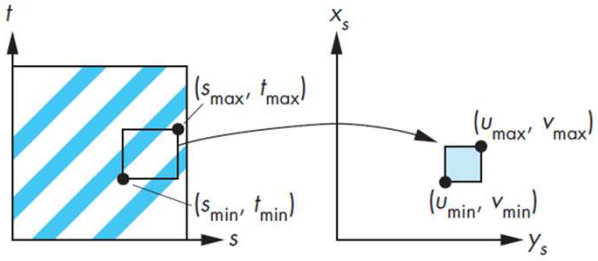
14

Linear Mapping

- Map a texture to a group of parametric surface patches

$$u = u_{min} + \frac{s - s_{min}}{s_{max} - s_{min}}(u_{max} - u_{min})$$

$$v = v_{min} + \frac{t - t_{min}}{t_{max} - t_{min}}(v_{max} - v_{min})$$

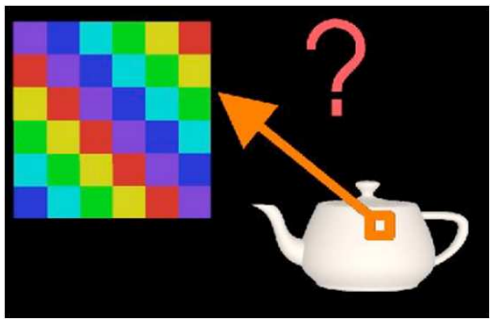


- This mapping is easy to apply, but it does not take into account the curvature of the surface.

15

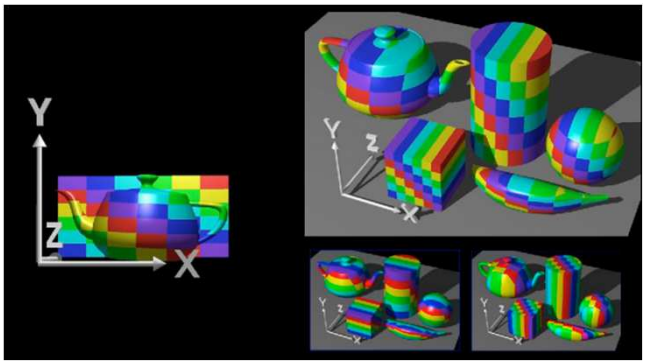
Linear Mapping

- Given a point on the object (x, y, z), what point (u, v) in the texture we use?



16

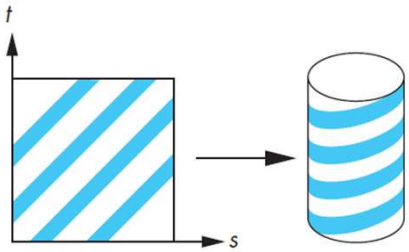
Linear Mapping - Exampel



17

Two part Mapping

- One solution to the mapping problem is to first map the texture to a simple intermediate surface
- Example: map to cylinder



18

Cylindrical Mapping

- Parametric cylinder

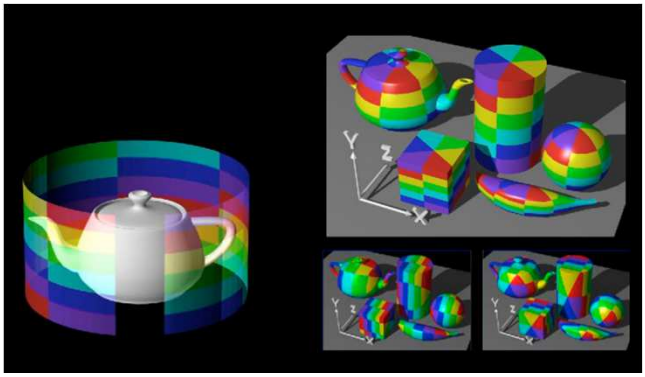
$$x = rcos(2\pi u)$$
$$y = rsin(2\pi u)$$
$$z = v/h$$

- Maps rectangle in u, v space to cylinder of radius r and height h in world coordinates

$$s = u$$
$$t = v$$

19

Cylindrical Mapping - Example



20

Spherical Map

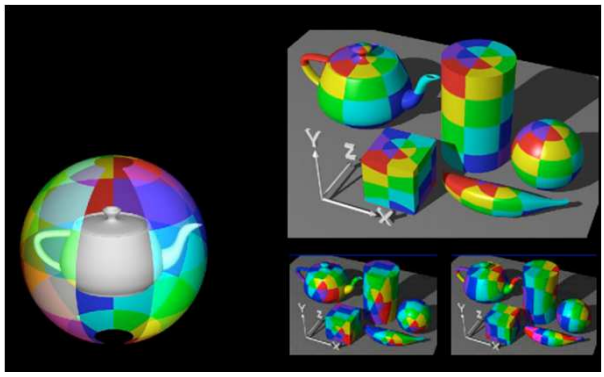
- We can use a parametric sphere

$$x = rcos(2\pi u)$$
$$y = rsin(2\pi u)\cos(2\pi v)$$
$$z = rsin(2\pi u)\sin(2\pi v)$$

- in a similar manner to the cylinder but have to decide where to put the distortion

21

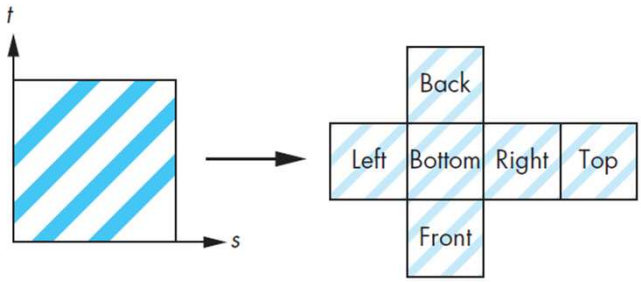
Spherical Map - Example



22

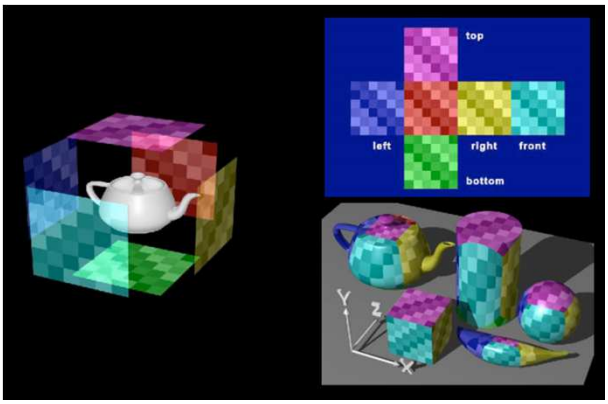
Box Mapping

- Easy to use with simple orthographic projection
- Also used in environmental maps



23

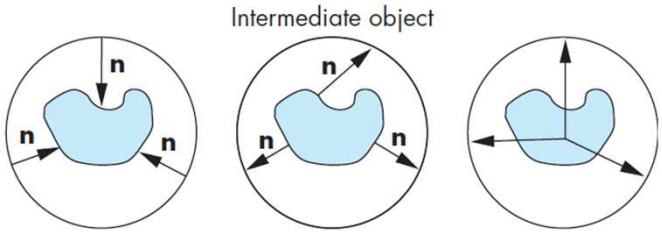
Box Mapping - Example



24

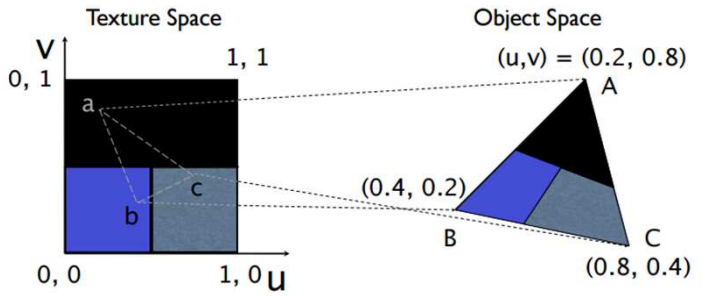
Second Mapping

- Map from intermediate object to actual object
- Three possible strategies
  - Normals from intermediate to actual
  - Normals from actual to intermediate
  - Vectors from center of intermediate



25

Texturing triangles



Good selection

Poor selection

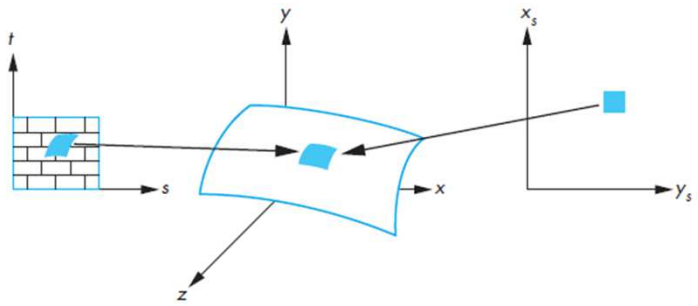
Stretched over trapezoid

`glTexCoord*()`

26

Point Sampling

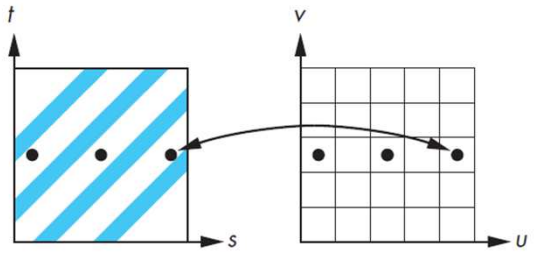
- Interested in mapping from screen to texture coordinates
- Map back to texture image and use the nearest texel



27

What is aliasing?

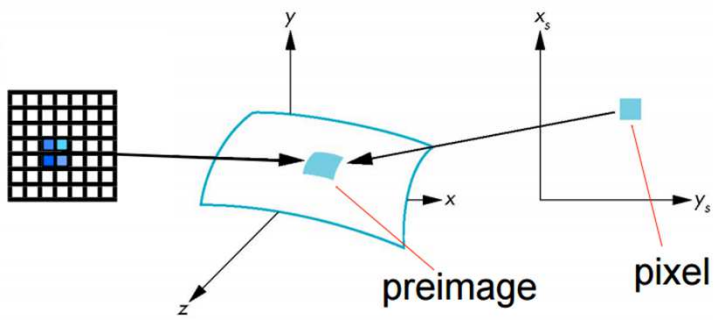
- An on-screen pixel does not always map neatly to a texel. Particularly severe problems in regular textures



28

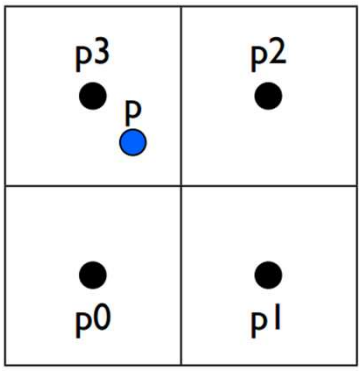
Area Averaging

- A better but slower option is to use area averaging



29

Use bilinear filtering



p = ?

nearest neighbor | bilinear | bicubic

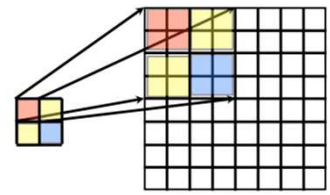
Wikipedia

mitigate magnification artifacts

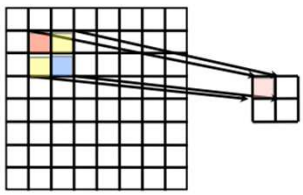
30

Magnification and Minification

- More than one texel can cover a pixel (minification) or more than one pixel can cover a texel (magnification)
- Can use point sampling (nearest texel) or linear filtering (2 x 2 filter) to obtain texture values



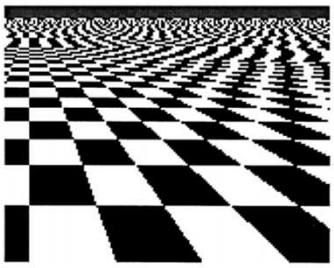
Magnification

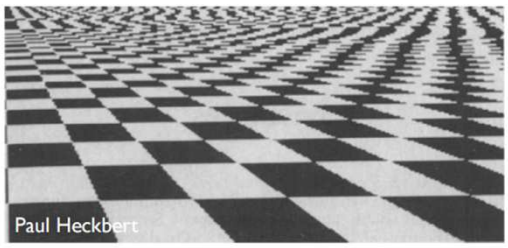


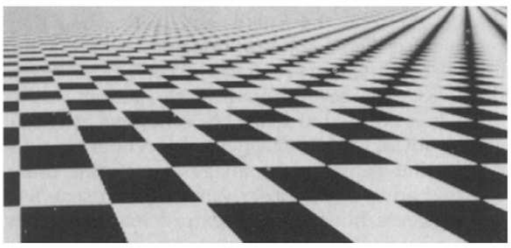
Minification

31

Aliasing artifacts





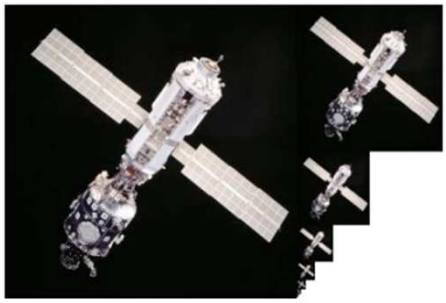


We apply **filtering** to reduce aliasing artifacts

32

Mipmapped Textures

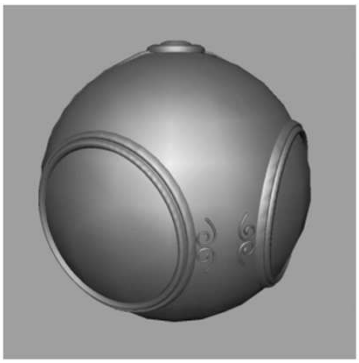
- Reduce minification artifacts
- *Mipmapping* allows for prefiltered texture maps of decreasing resolutions
- Lessens interpolation errors for smaller textured objects



33

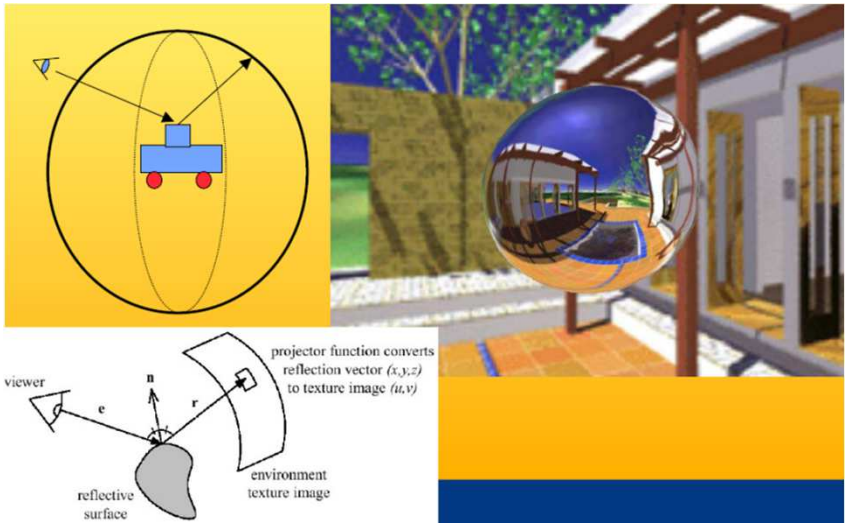
Environment Mapping

- Used to create appearance of reflective and refractive surfaces without ray tracing which requires global calculations



34

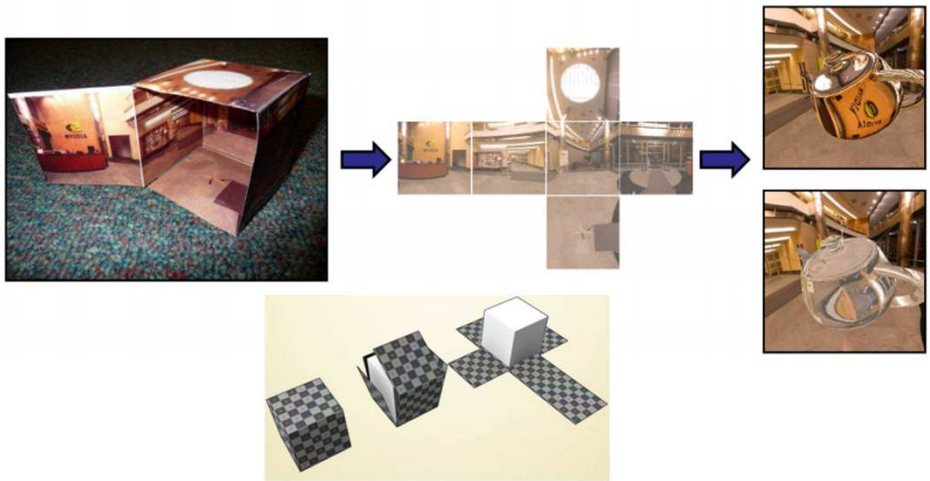
Environment Mapping



35

Cube Map

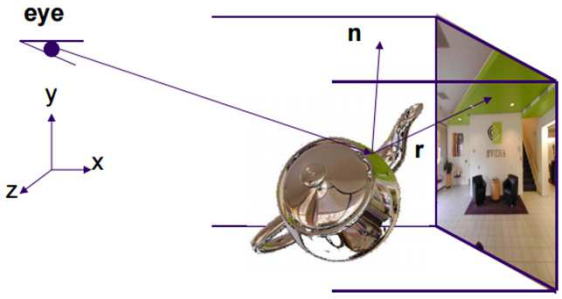
- Stores “environment” around objects as 6 sides of a cube



36

Reflection Mapping

- Need to compute reflection vector, R
- Use R by for lookup
- OpenGL hardware supports cube maps, makes lookup easier



37