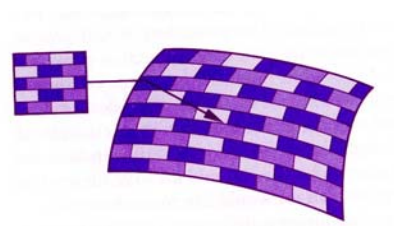
CS 112 - Texture Mapping

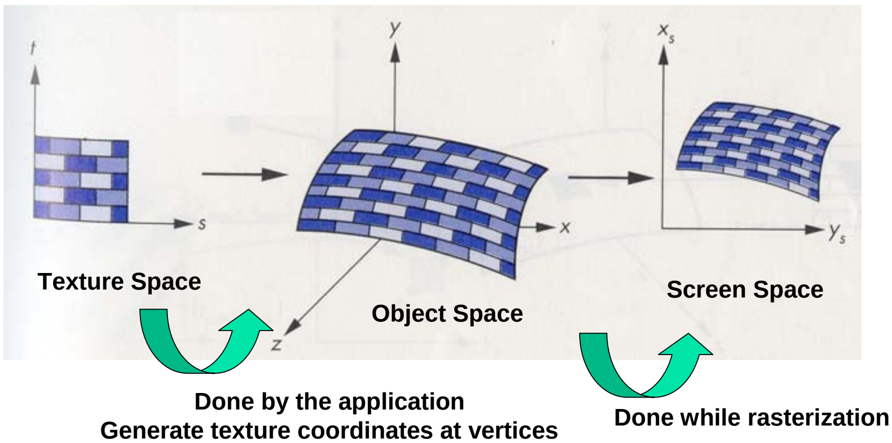
What is Texture Mapping?

- Color is not sufficient for realistic appearances
- Wrap (Map) a image on a surface
  - Like a wall-paper
  - Like gift wrapping



2D Texture Mapping

- Three spaces



Texture Space to Object Space

- Rectangular image mapped to arbitrary surfaces
  - The texture will get stretched differently at different places on the surface based on the curvature
  - Imagine wrapping a rectangular image on a sphere
  - Two Ways to do it

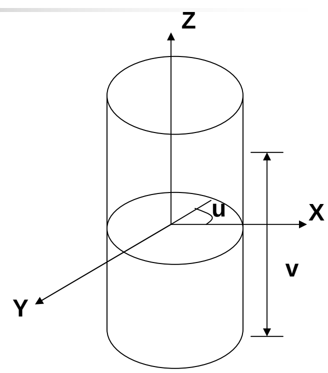
Method 1

- Find the parametric representation of the surface defined by parameters (u,v)
  - Since 2D object embedded in real world
- Map (u,v) to (s,t) – (s,t) varies from 0 to 1
- Find the (u,v) for each vertex in the tessalated object and find the corresponding (s,t)

Aditi Majumder, CS 112 Slide 5

Example: Open Cylinder

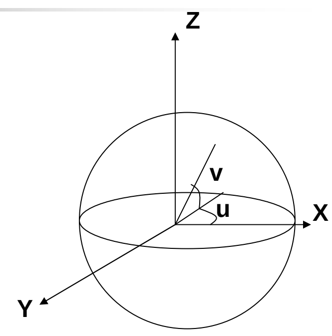
- u – angle, $-180 \leq u \leq 180$
- v – height, $0 \leq v \leq 1$
- $x = R\cos(u)$
- $y = R\sin(u)$
- $z = v$
- Map (s,t) to (u,v)
  - $s = (u+180/360)$
  - $t = v$



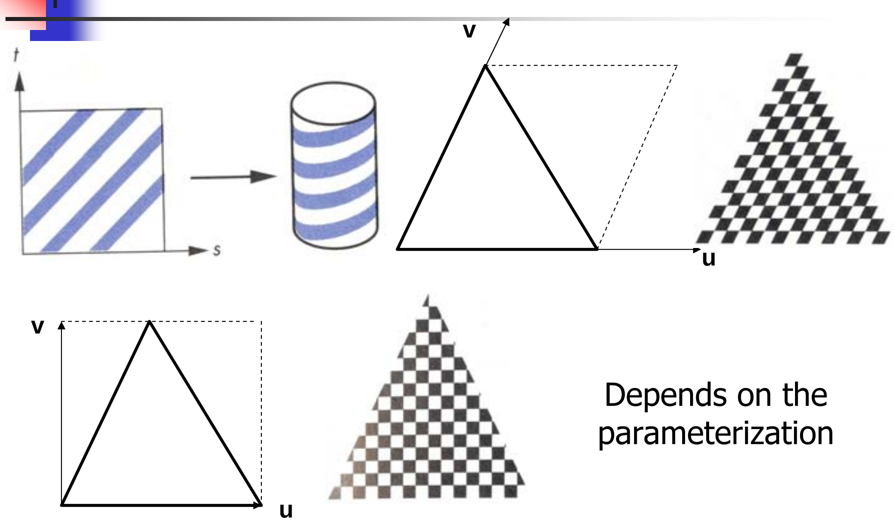
Aditi Majumder, CS 112 Slide 6

Example: Sphere

- u – horizontal angle
  - $-180 \le u \le 180$
- v – vertical angle
  - $-90 \le v \le 90$
- $x = R \cos(v) \cos(u)$
- $y = R \cos(v) \sin(u)$
- $z = R \sin(v)$
- Map (s,t) to (u,v)
  - $s = (u+180)/360$
  - $t = (v+90)/180$



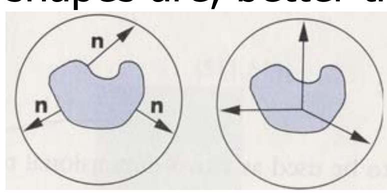
Results



Depends on the parameterization

Method 2: Intermediate Geometry

- Difficult to parameterize arbitrary geometry
- Define intermediate simple surface and parameterize it: a plane, sphere or cylinder
- Enclose arbitrary geometry within simple geometry
- More close these shapes are, better the mapping



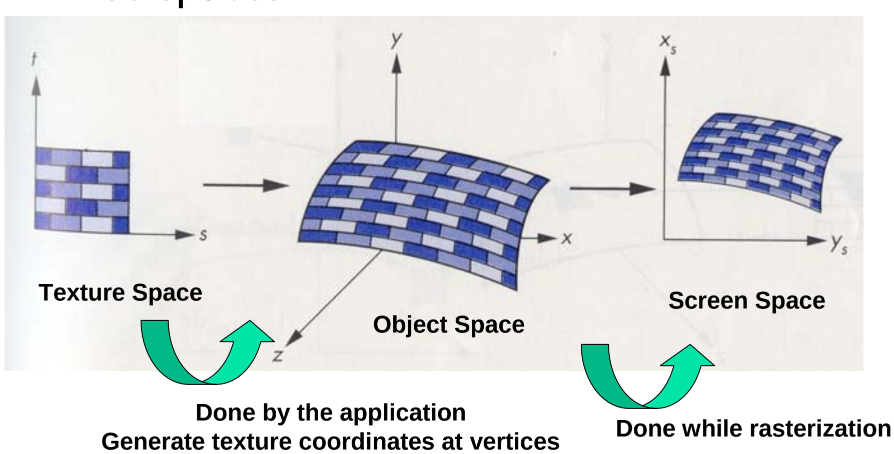
Result (Planar Mapping)

Result (Cylindrical Mapping)

Aditi Majumder, CS 112 Slide 11

2D Texture Mapping

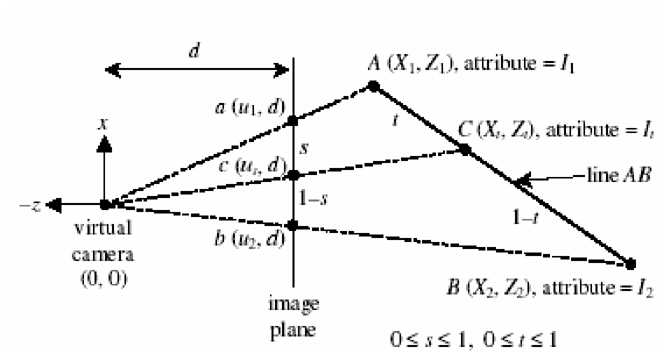
- Three spaces



Aditi Majumder, CS 112 Slide 12

Object Space to Screen Space

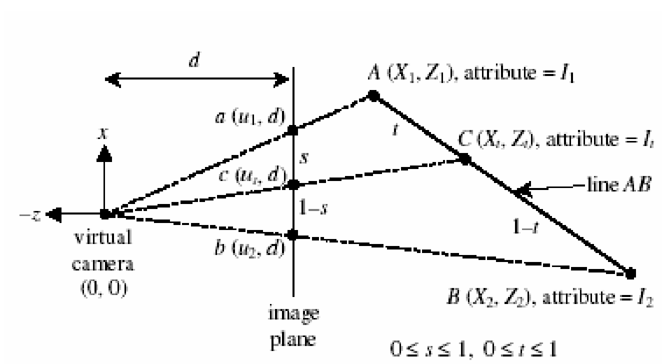
- The texture coordinates are known in the object space
- Needs to be interpolated in the screen space



Aditi Majumder, CS 112 Slide 13

Interpolation of Attributes

$$I_t = I_1 + t(I_2 - I_1) \qquad t = \frac{sZ_1}{sZ_1 + (1-s)Z_2} \qquad I_t = \left(\frac{I_1}{Z_1} + s\left(\frac{I_2}{Z_2} - \frac{I_1}{Z_1}\right)\right) \Big/ \frac{1}{Z_t}$$



Aditi Majumder, CS 112 Slide 14

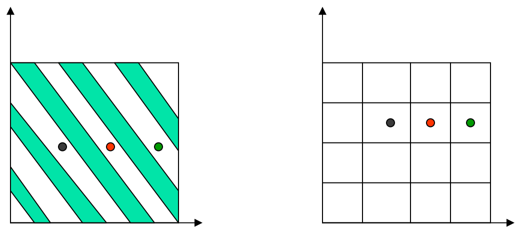
Sampling the Texture

- You have FP numbers between 0 and 1 for each pixel
- How do you get the colors from the texture image?

Point Sampling

- Multiply by the texture size to generate another FP value
- Round off the FP values to integers (GL_NEAREST)
- Pick the color of the integer texel

Aliasing Problems



- Miss the stripes completely
- Texture is not adequately sampled by the pixels

Linear Interpolation

- Multiply by the texture size to generate another FP value
- Interpolate the color from the four nearest texels using bilinear interpolation (GL_LINEAR)
- Does not remove aliasing completely since sampling is still inadequate

Mipmapping

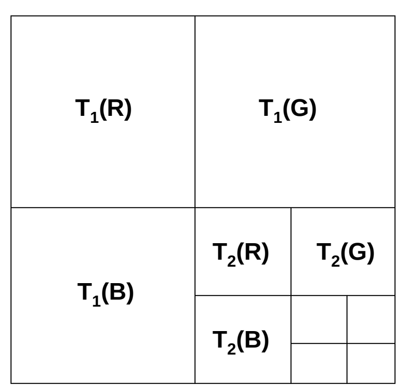
- Ideally, we should filter the image and subsample it to reduce the frequency content
- Then we should pick the color from this subsampled image to avoid aliasing
- The lower frequency content will make the sampling adequate

Aditi Majumder, CS 112 Slide 19

Mipmapping

- Special way of storing images of different resolutions
- T_1:128x128 (RGB)
- T_2:64x64 (RGB)
- T_3:32x32 (RGB)
- And so on…
- Choose appropriate resolution based on screen space projection

Size: 4 x original texture



Aditi Majumder, CS 112 Slide 20

Point sampling in Mipmap

- Can be done in two ways
  - Round off and sample one color from the texture
  - Interpolate from the nearest four texels of the texture