Lecture 08- Texture and Surface Detailing (Chapter 18)

I. Texture mapping

A. Motivation - currently to add detail to our scene we would increase the number of facets in our model/world each w/ separate colors for example. However, this is difficult and slow.

i. Texture mapping is the process of "painting" an image onto a surface, its easier for an artist and faster to render.

ii. Textures are 1D, 2D, 3D arrays of colors and texture space is bounded in $[0,1]^d$ a component is called a texel.

B. Linear texture Pattern

i. 1D color array. Could be used for stripes on a flag [G][R]

ii. Texture space is [0,1] referenced by a coordinate s.

s=0 first array element, s=1 is last array element. *How about other s values?*

a. Nearest texel - Let's say s=v is between texels t_a and t_b. Choose a if $s-a < b-s$, b otherwise

b. Linear combination - Color $c = (s-a)t_a + (b-s)t_b$

iii. Interpolation - Colors can be looked up per vertex and then interpolated across primitive (like Gouraud shading) or texel coordinate interpolated and looked up per pixel (like Phong shading), for example. A line: a ... b

iv. Coordinate values outside texture space *Ex. $s \notin [0,1]$. Ask class*

a. Called "wrapping" procedure

b. Repeating - "modulus" ie. [1,2] is a repeat of [0,1], etc. Can do a mirrored version.

c. Clamping - Any value outside of [0,1] is clamped to 0 or 1

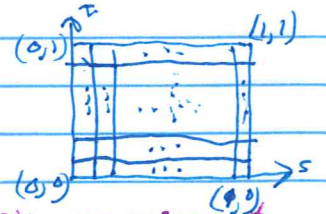
d. Border - Any value outside of [0,1], is assigned a border color

C. Surface Texture Patterns

i. 2D texture - 2D array of colors

ii. Texture space is $[0,1]^2$ referenced by coordinate (s,t)

This time coordinate between 4 array elements. Choices? Wrapping? *Ask class to walk through examples* *Note, can do separate procs per s vs t*



D. Volume Texture Patterns - Same but 3D, $[0,1]^3$ or (s,t,r)

E. Texture Reduction Patterns

i. Size of object often changes so texture lookup alters based on distance from object* causes distortion *Look up in google to show*

ii. To solve - use Level of Detail (LoD) in texture. Called mipmapping. MIP - multum in parvo, Latin for much in small object.

iii. Idea - use texture to generate 1/2 size, 1/4 size, 1/8 size etc down to 1x1.

example for 16x16 image generate 8x8, 4x4, 2x2, 1x1. Then look up mipmap based on z depth.

II. Bump Mapping (~~outdated~~ older method but idea is vital to understand possibilities)

A. Motivation - texturing allows color variation but nothing else. eg. roughness (oranges)
We need to modify surface lighting procedure.
i. Our map will store single values of "bumpiness" used to perturb normal.
ii. Values of 0.5 will keep the same, values close to 0 depress normal, close to 1 heightens normal.

B. Simply multiply bump value and normal - fakes detail. (very weak/bad model)

C. Advanced model required "advanced calculus"

D. Bumps are looked up in table. Essentially textures are an advanced form of look-up. Can use this for
i Bump map
ii Normal map - changing normal about object (advanced bump map)
iii. Displacement map - disturbs geometry itself
iv. Specular maps - look up specular component of vertex
v. Etc.