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 faces in our model/world, each with separate colors. For example, however, this is difficult and slow.

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

B. 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.

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

\[
\begin{align*}
&0.0 \rightarrow s \\
&5 = 1 \rightarrow s_0 = \text{last array element. How about other s values?}
\end{align*}
\]

a. Nearest texel - let's say, s = 0.3 is between texels t_a and t_b. Choose

\[ t_a \text{ if } 0.3 < t_a \leq 0.4, \text{ otherwise } \]

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 primitives (like triangle shading) or across coordinate interpolated and looked up

iv. Values outside texture space \*Ex: s = 0.5, ask class

a. Baked wrapping procedure
b. Repeating - "tiles" e.g. \((0,1) \) is 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 pass per s or t +

D. Volume Texture Patterns - Same but 3D \([0,1]^3\) or \([0,1,0] \)

E. Texture Reduction Patterns
i. Size of object often changes so texture lookups alters based on distance from object causes distortion \*Look up in google to learn

ii. To solve - use Level of Detail (LOD) in texture, called mipmapping. Mip - another in pare similarity for much a small object.

v. Idea - use texture to generate \( 16 \times 16 \), \( 32 \times 32 \), \( 64 \times 64 \) etc. down to \( 1 \times 1 \).

Example for \( 16 \times 16 \) image generates \( 8 \times 8 \), \( 4 \times 4 \) and \( 1 \times 1 \). Then lookup mipmaps based on z depth.
II. Bump Mapping

- Motivation - texturing allows color variation but nothing else, e.g., roughness (wenges)
  - We need to modify surface lighting procedure.
  - Our map will store single values of "bumpness" used to perturb normal.
  - Values of 0.5 will keep the same, 0 values close to 0 depress normal, closer to 1 heightens normal.

- Simply multiply bump value and normal - fades detail (very weak/bed model)

- Advanced model requires "advanced calc-ics"

D. Bumps are baked up in table. Essentially, textures are an advanced form of lookup, can use this for
  - Bump map
    - Normal map - brings normal about object (advanced bump map)
    - Displacement map - disturbs geometry itself
    - Specular maps - look up specular component on vertex
      - etc.