Texture Mapping

CS 4620 Lecture 12
Texture mapping

- Objects have properties that vary across the surface
Texture Mapping

• So we make the shading parameters vary across the surface
Texture mapping

- Adds visual complexity; makes appealing images
Texture mapping

• Color is not the same everywhere on a surface
  – one solution: multiple primitives
• Want a function that assigns a color to each point
  – the surface is a 2D domain, so that is essentially an image
  – can represent using any image representation
  – raster texture images are very popular
A definition

**Texture mapping:** a technique of defining surface properties (especially shading parameters) in such a way that they vary as a function of position on the surface.

- This is very simple!
  - but it produces complex-looking effects
Examples

• Wood gym floor with smooth finish
  – diffuse color $k_D$ varies with position
  – specular properties $k_S, n$ are constant

• Glazed pot with finger prints
  – diffuse and specular colors $k_D, k_S$ are constant
  – specular exponent $n$ varies with position

• Adding dirt to painted surfaces

• Simulating stone, fabric, …
  – to approximate effects of small-scale geometry
    • they look flat but are a lot better than nothing
Mapping textures to surfaces

- Usually the texture is an image (function of $u, v$)
  - the big question of texture mapping: where on the surface does the image go?
  - obvious only for a flat rectangle the same shape as the image
  - otherwise more interesting
- Note that 3D textures also exist
  - texture is a function of $(u, v, w)$
  - can just evaluate texture at 3D surface point
  - good for solid materials
  - often defined procedurally
Mapping textures to surfaces

• “Putting the image on the surface”
  – this means we need a function $f$ that tells where each point on the image goes
  – this looks a lot like a parametric surface function
  – for parametric surfaces you get $f$ for free
Texture coordinate functions

- Non-parametrically defined surfaces: more to do
  - can’t assign texture coordinates as we generate the surface
  - need to have the inverse of the function $f$

- Texture coordinate fn.
  $\phi : S \rightarrow \mathbb{R}^2$
  - for a vtx. at $p$
  get texture at $\phi(p)$
Texture coordinate functions

- Mapping from $S$ to $D$ can be many-to-one
  - that is, every surface point gets only one color assigned
  - but it is OK (and in fact useful) for multiple surface points to be mapped to the same texture point
- e.g. repeating tiles
Texture coordinate functions

• Define texture image as a function

\[ T : D \rightarrow C \]

– where \( C \) is the set of colors for the diffuse component

• Diffuse color (for example) at point \( p \) is then

\[ k_D(p) = T(\phi(p)) \]
Examples of coordinate functions

- A rectangle
  - image can be mapped directly, unchanged
Examples of coordinate functions

• For a sphere: latitude-longitude coordinates
  – $\phi$ maps point to its latitude and longitude
Examples of coordinate functions

- A parametric surface (e.g. spline patch)
  - surface parameterization gives mapping function directly (well, the inverse of the parameterization)
Examples of coordinate functions

• For non-parametric surfaces it is trickier
  – directly use world coordinates
• need to project one out
Examples of coordinate functions

- Non-parametric surfaces: project to parametric surface

[Möller & Haines 2002]
Examples of coordinate functions

- Triangles
  - specify \((u,v)\) for each vertex
  - define \((u,v)\) for interior by linear interpolation
Texture coordinates on meshes

• Texture coordinates become per-vertex data like vertex positions
  – can think of them as a second position: each vertex has a position in 3D space and in 2D texture space
• How to come up with vertex \((u,v)\)s?
  – use any or all of the methods just discussed
    • in practice this is how you implement those for curved surfaces approximated with triangles
  – use some kind of optimization
    • try to choose vertex \((u,v)\)s to result in a smooth, low distortion map
Reflection mapping

• Early (earliest?) non-decal use of textures
• Appearance of shiny objects
  – Phong highlights produce blurry highlights for glossy surfaces.
  – A polished (shiny) object reflects a sharp image of its environment.
• The whole key to a shiny-looking material is providing something for it to reflect.

Figure 2. (a). A shiny sphere rendered under photographically acquired real-world illumination. (b). The same sphere rendered under illumination by a point light source.
Reflection mapping

• From ray tracing we know what we’d like to compute
  – trace a recursive ray into the scene—too expensive
• If scene is infinitely far away, depends only on direction
  – a two-dimensional function
Environment map

- A function from the sphere to colors, stored as a texture.

[Blinn & Newell 1976]
Spherical environment map
Environment Maps

[Paul Debevec]
Cube environment map
Normal mapping

original mesh
4M triangles

simplified mesh
500 triangles

simplified mesh
and normal mapping
500 triangles

[Paolo Cignoni]
base subdivision surface

hand-painted displacement map (detail)

displaced surface

Paweł Filip
tolas.wordpress.com
Bump mapping

[Blinn 1978]
Displacement mapping

Geometry

Bump mapping

Displacement mapping
Another definition

Texture mapping: a general technique for storing and evaluating functions.

- They’re not just for shading parameters any more!