TEXTURE MAPPING

SAUMITRA BAGCHI
Texture:
The feel, appearance, or consistency of a surface or a substance.
Texture mapping is a method for adding detail, surface texture (a bitmap or raster image), or color to a computer-generated graphic or 3D model.
WHY DO WE NEED TEXTURE MAPPING

Because objects look fake and boring without it.
ADDING TEXTURE IMPROVES REALISM
THE 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
MODELING AN ORANGE

• Consider the problem of modeling an orange (the fruit)
• 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
MODELING AN ORANGE

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

• adding per-pixel surface details without raising the geometric complexity of a scene
• keep the number of vertices and primitives low
• add detail per fragment
• modelling and rendering time is saved by keeping the geometrical complexity low
TYPES OF MAPPING

**Texture Mapping**
- Uses images to fill inside of polygons

**Environment (reflection mapping)**
- Uses a picture of the environment for texture maps
- Allows simulation of highly specular surfaces

**Bump mapping**
- Emulates altering normal vectors during the rendering process
TEXTURE MAPPING

Instead of calculating color, shade, light, etc. for each pixel we just paste images to our objects in order to create the illusion of realism
TEXTURE MAPPING

Mapping a 2D Texture onto a 3D Object Using a UV Map

- stretching a 2D texture over the surface of a 3D shape
- The answer to that is a technique called **UV Mapping**
TEXTURE MAPPING

• 3D space defined by \((x, y, z)\) coordinates
• We want to take a 2D square texture and map it onto our 3D geometry
• 2D coordinate system on top of the texture, with \((0,0)\) in the bottom left and \((1,1)\) in the top right.
• we’ll call these 2D textures coordinates by the letters \((u, v)\), which is where the name UV mapping comes from.
TEXTURE MAPPING

• Let’s create a texture now to help us visualize this. We’ll use a simple black and white checker pattern and label a few of the UV coordinates. Once we are done, we’ll have something that looks like this:

• **UV mapping** is the process by which we map 2D \((u,v)\) coordinates onto 3D \((x, y, z)\) coordinates:

\[
(u,v) \rightarrow (x,y,z)
\]
TEXTURE MAPPING

In the figure below, we’re showing how we want the texture to map onto the front face of the cube.

We’ve also drawn in the (x, y, z) coordinates of the vertices of this face, so we can see that we want this mapping from UV coordinates to 3D coordinates:

- (0,0) → (-1, -1, 1)
- (0,1) → (-1, 1, 1)
- (1,1) → (1, 1, 1)
- (1,0) → (1, -1, 1)
TEXTURE MAPPING
ENVIRONMENT (REFLECTION MAPPING)

• Is an efficient image based lighting technique
• Approximating the appearance of a reflective surface by means of a precomputed texture image.
**BUMP MAPPING**

- Is a technique in computer graphics for simulating bumps and wrinkles on the surface of an object.
- This is achieved by perturbing the surface normals of the object and using the perturbed normal during lighting calculations.
WHERE DOES MAPPING TAKE PLACE?

Mapping techniques are implemented at the end of the rendering pipeline.
IS IT SIMPLE?

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

• Parametric coordinates
  • May be used to model curves and surfaces
• Texture coordinates
  • Used to identify points in the image to be mapped
• Object or World Coordinates
  • Conceptually, where the mapping takes place
• Window Coordinates
  • Where the final image is really produced
TEXTURE MAPPING
MAPPING FUNCTIONS

• Basic problem is how to find the maps
• Consider mapping from texture coordinates to a point 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
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)
TWO-PART MAPPING

• One solution to the mapping problem is to first map the texture to a simple intermediate surface

• Example: map to cylinder
Cylindrical Mapping

parametric cylinder

\[
\begin{align*}
x &= r \cos 2\pi u \\
y &= r \sin 2\pi u \\
z &= v/h
\end{align*}
\]

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

\[
\begin{align*}
s &= u \\
t &= v
\end{align*}
\]

maps from texture space
NON-PARAMETRIC TEXTURE MAPPING

- With “non-parametric texture mapping”
  - Texture size and orientation are fixed
  - They are unrelated to size and orientation of polygon
PARAMETRIC TEXTURE MAPPING

With parametric texture mapping, texture size and orientation are tied to the polygon:

- Separate texture space and screen space
- Texture the polygon as before but in texture space
- Deform (render) the textured polygon into screen space
TEXTURE MAPPING IN OPENGL

• Three steps to applying a texture

1. specify the texture
   – read or generate image
   – assign to texture
   – enable texturing

2. assign texture coordinates to vertices
   – Proper mapping function is left to application

3. specify texture parameters
   – wrapping, filtering
TEXTURE MAPPING
A texel, texture element, or texture pixel is the fundamental unit of texture space, used in computer graphics. Textures are represented by arrays of texels, just as pictures are represented by arrays of pixels.
IMPLEMENTING TEXTURE MAPPING

• A texture lives in its own abstract image coordinates parameterized by \((u,v)\) in the range \(([0..1], [0..1])\):

• It can be wrapped around many different surfaces:

• Computing \((u,v)\) texture coordinates in a ray tracer is fairly

• Note: if the surface moves/deforms, the texture goes with it.
ANTI-ALIASING

• Aliasing is the distortion produced by representing a high resolution signal at a lower resolution
• Anti-aliasing aims to remove this distortion
REFERENCES

https://en.wikipedia.org/wiki/Texture_mapping
https://discoverthreejs.com/book/1-first-steps/4-textures-intro/
http://courses.cs.vt.edu/~cs4204/lectures/texture_mapping.pdf
THANK YOU