Texture Mapping

Reading

Required

- Angel, 8.6, 8.7, 8.9, 8.10, 9.13-9.13.2

Recommended

- Paul S. Heckbert. Survey of texture mapping. **IEEE Computer Graphics and Applications** 6(11): 56--67, November 1986.

Optional

- Woo, Neider, & Davis, Chapter 9
- James F. Blinn and Martin E. Newell. Texture and reflection in computer generated images. **Communications of the ACM** 19(10): 542--547, October 1976.

Texture mapping

Texture mapping (Woo et al., fig. 9-1)

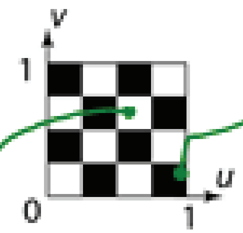
Texture mapping allows you to take a simple polygon and give it the appearance of something much more complex.

- Due to Ed Catmull, PhD thesis, 1974
- Refined by Blinn & Newell, 1976

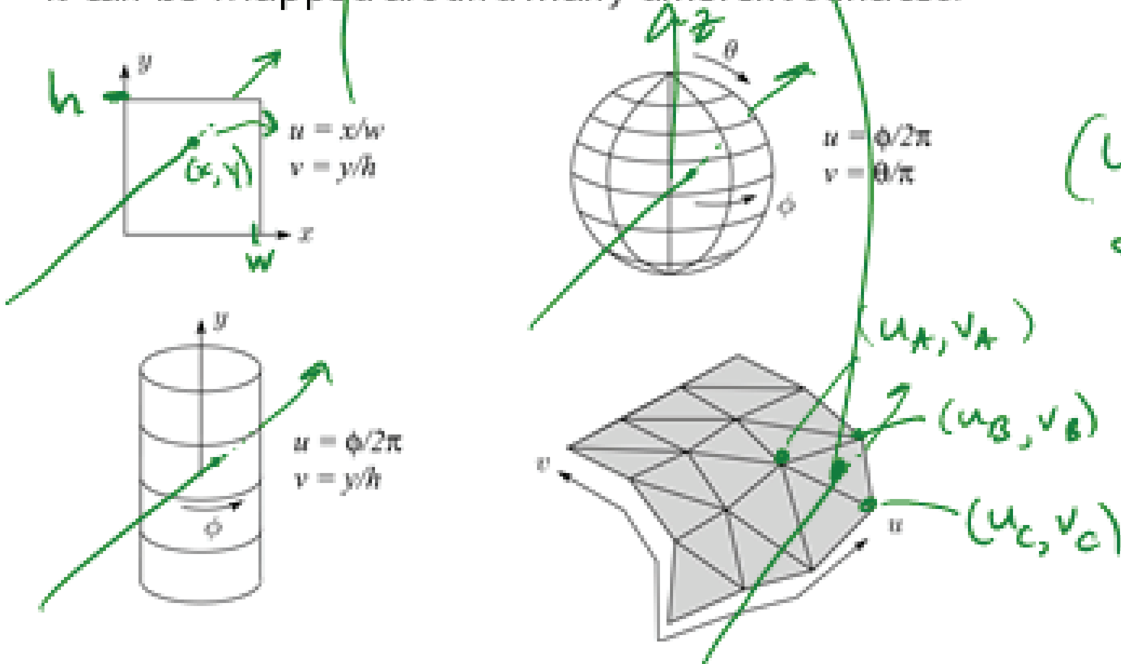
Texture mapping ensures that "all the right things" happen as a textured polygon is transformed and rendered.

Implementing texture mapping

A texture lives in it own abstract image coordinates paramaterized by (u,v) in the range ([0..1], [0..1]):



It can be wrapped around many different surfaces:



$$(u,v) = \alpha(u_A, v_A) + \beta(u_B, v_B) + \gamma(u_C, v_C)$$

In graphics hardware, texture coordinates of triangle vertices are interpolated during rasterization.

Note: if the surface moves/deforms, the texture goes with it.

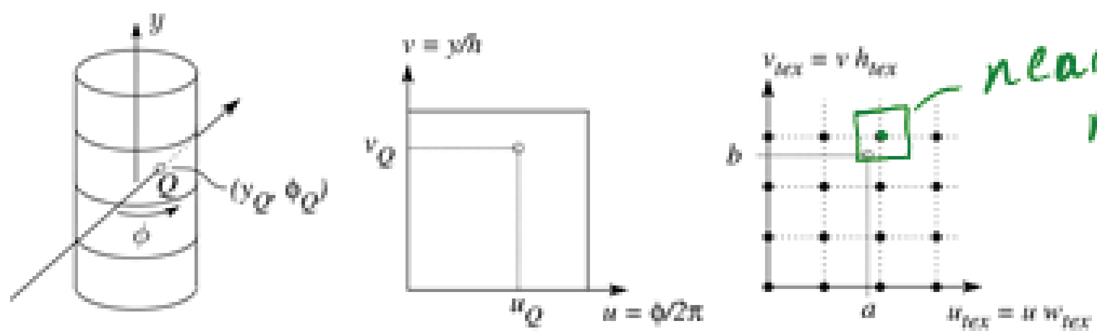
Mapping to texture image coords

The texture is usually stored as an image. Thus, we need to convert from abstract texture coordinate:

(u, v) in the range ([0..1], [0..1])

to texture image coordinates:

(u_{tex}, v_{tex}) in the range ([0.. w_{tex}], [0.. h_{tex}])



Ray intersection

Mapping to abstract texture coords

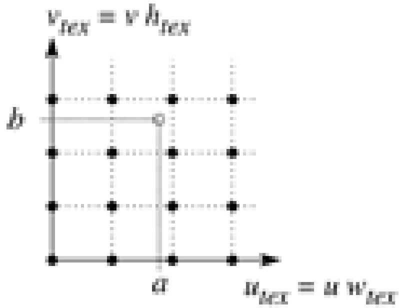
Mapping to texture pixel coords

nearest neighbor interpolation

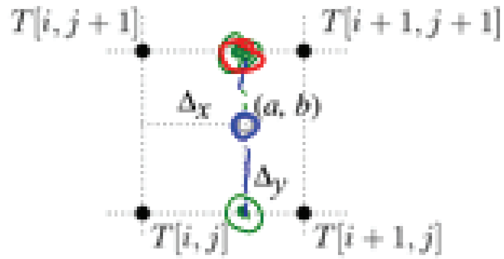
Q: What do you do when the texture sample you need lands between texture pixels?

Texture resampling

We need to resample the texture:



Mapping to texture pixel coords



Close-up

A common choice is **bilinear interpolation**:

$$T(a,b) = T(i+\Delta_x, j+\Delta_y)$$

$$= \underline{(1-\Delta_y)}\, T(i+\Delta_x, j) + \underline{\Delta_y}\, T(i+\Delta_x, j+1)$$

$$T(i+\Delta_x, j) = \underline{(1-\Delta_x)}\, T[i,j] + \underline{\Delta_x}\, T[i+1,j]$$

$$T(i+\Delta_x, j+1) = \underline{(1-\Delta_x)}\, T[i,j+1] + \underline{\Delta_x}\, T[i+1,j+1]$$

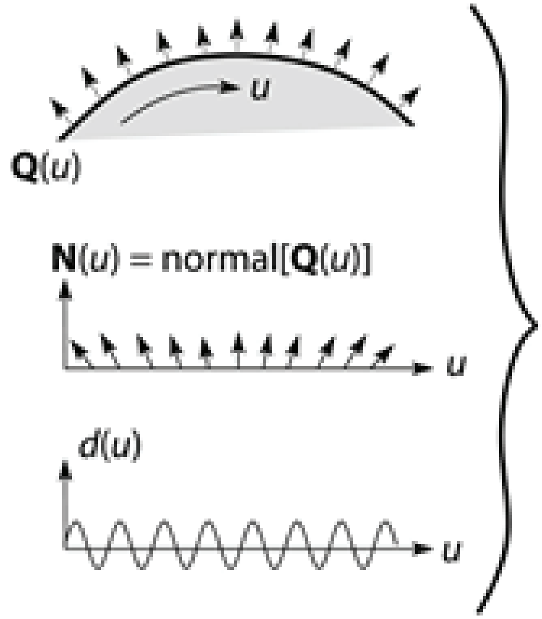
$$T(a,b) = \underline{(1-\Delta_y)(1-\Delta_x)}\, T[i,j] + \underline{(1-\Delta_y)\Delta_x}\, T[i+1,j] +$$

$$\underline{\Delta_y(1-\Delta_x)}\, T[i,j+1] + \underline{\Delta_y \Delta_x}\, T[i+1,j+1]$$

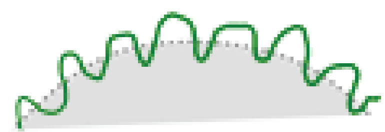
Displacement mapping

Textures can be used for more than just color.

In **displacement mapping**, a texture is used to perturb the surface geometry itself. Here's the idea in 2D:



$$\tilde{\mathbf{Q}}(u) = \mathbf{Q}(u) + d(u)\mathbf{N}(u)$$



- These displacements "animate" with the surface
- In 3D, you would of course have (u,v) parameters instead of just u.

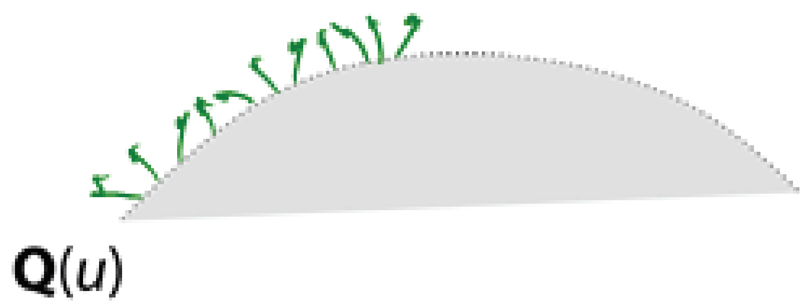
Q: Do you have to do hidden surface calculations on $\tilde{\mathbf{Q}}$? Y

Bump mapping

In **bump mapping**, a texture is used to perturb the normal:

- Use the original, simpler geometry, $\mathbf{Q}(u)$, for hidden surfaces
- Use the normal from the displacement map for shading:

$$\tilde{\mathbf{N}} = \text{normal}[\tilde{\mathbf{Q}}(u)]$$



Q: What artifacts in the images would reveal that bump mapping is a fake?

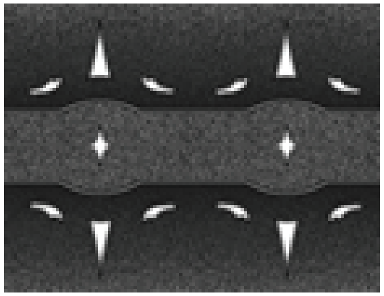
No self ~~occlus~~ occlusions

Silhouettes wrong

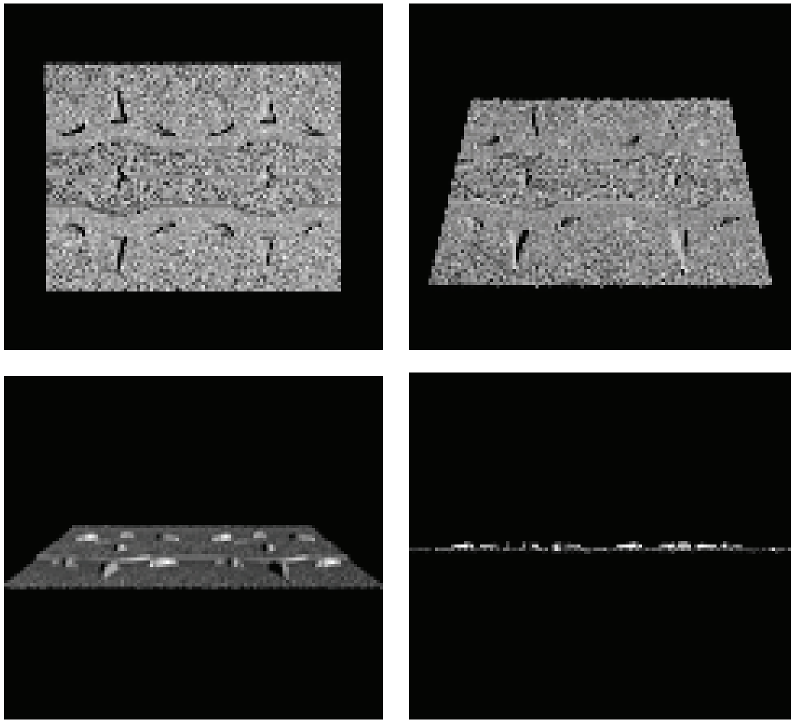
cast shadows

Displacement vs. bump mapping

Input texture



Rendered as displacement map over a rectangular surface



Displacement vs. bump mapping (cont'd)

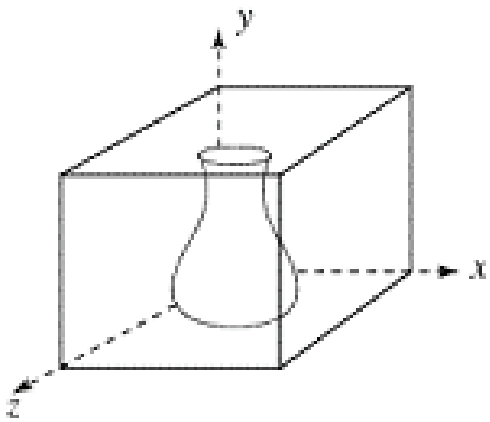
Original rendering

Rendering with bump map wrapped around a cylinder

Bump map and rendering by Wyvern Aldinger

Solid textures

Q: What kinds of artifacts might you see from using a marble veneer instead of real marble?



One solution is to use **solid textures**:

- Use model-space coordinates to index into a 3D texture
- Like "carving" the object from the material

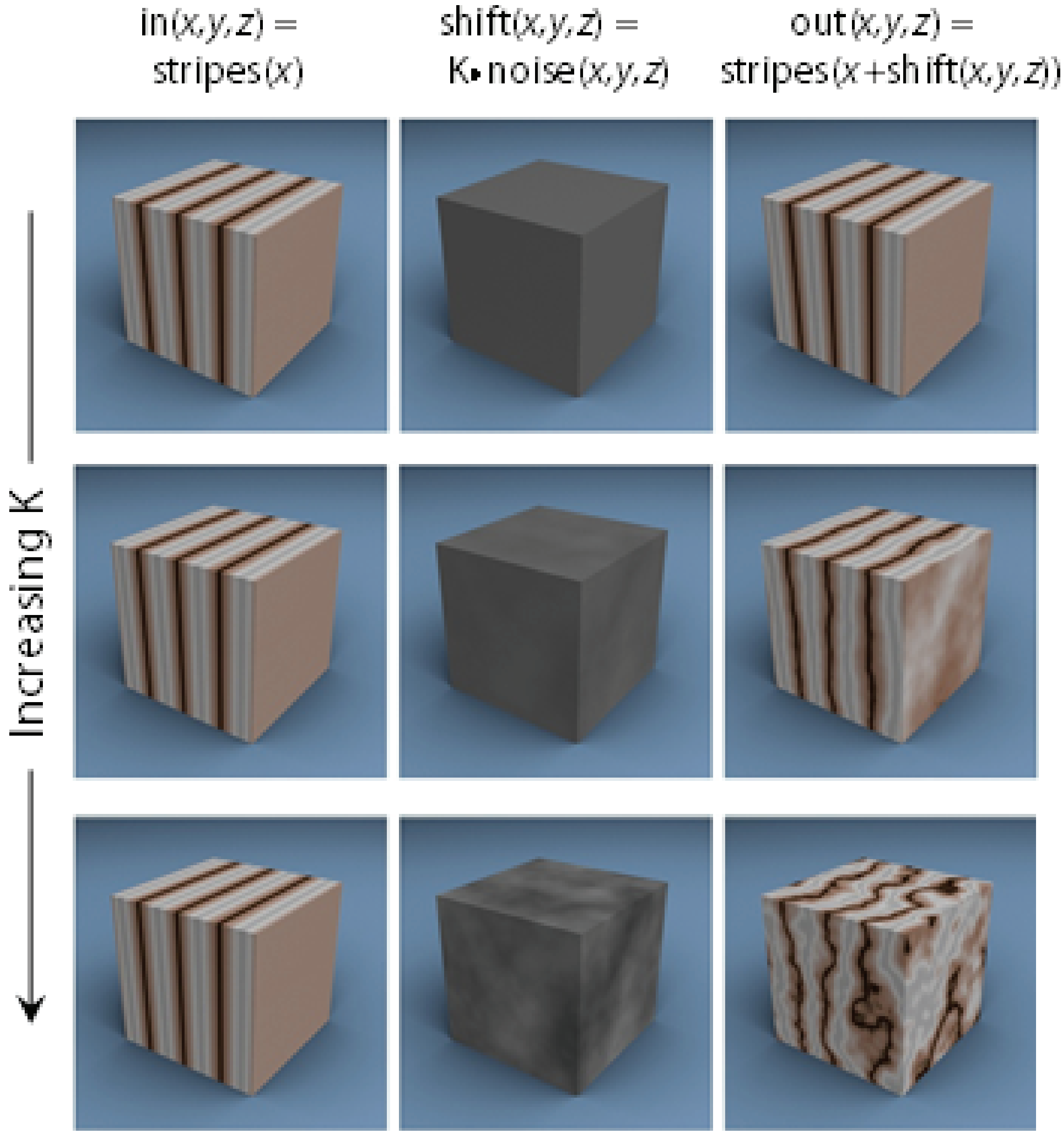
One difficulty of solid texturing is coming up with the textures.

Solid textures (cont'd)

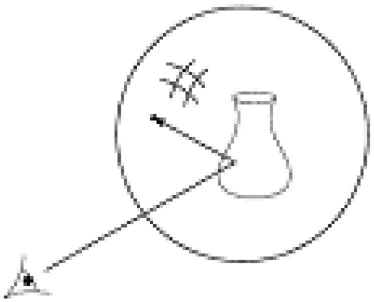
Here's an example for a vase cut from a solid marble texture:

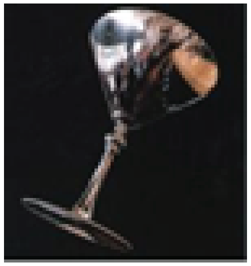
Solid marble texture by Ken Perlin, (Foley, IV-21)

Solid textures (cont'd)



Environment mapping





In **environment mapping** (also known as **reflection mapping**), a texture is used to model an object's environment:

- Rays are bounced off objects into environment
- Color of the environment used to determine color of the illumination
- Really, a simplified form of ray tracing
- Environment mapping works well when there is just a single object – or in conjunction with ray tracing

Under simplifying assumptions, environment mapping can be implemented in hardware.

With a ray tracer, the concept is easily extended to handle refraction as well as reflection.

Summary

What to take home from this lecture:

1. The meaning of the boldfaced terms.

2. Familiarity with the various kinds of texture mapping, including their strengths and limitations.