Lecture 10, 11
Texture Mapping

Xin (Shane) Li

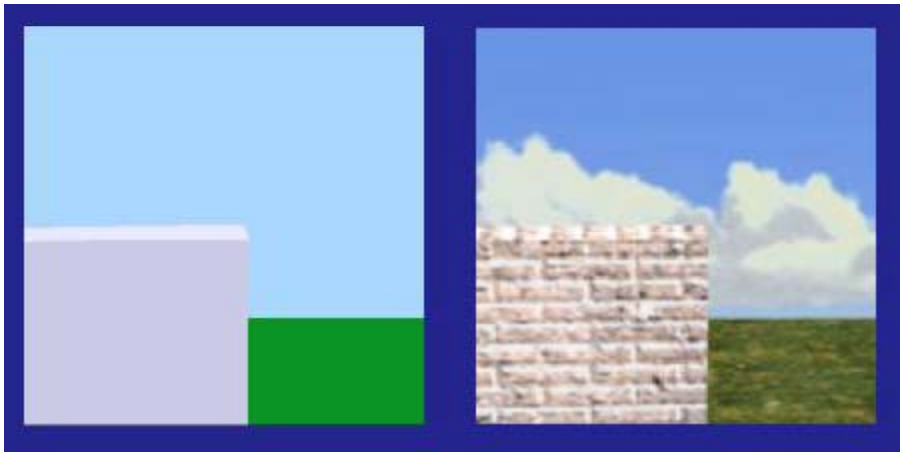
Sep. 24th and 29th, 2009

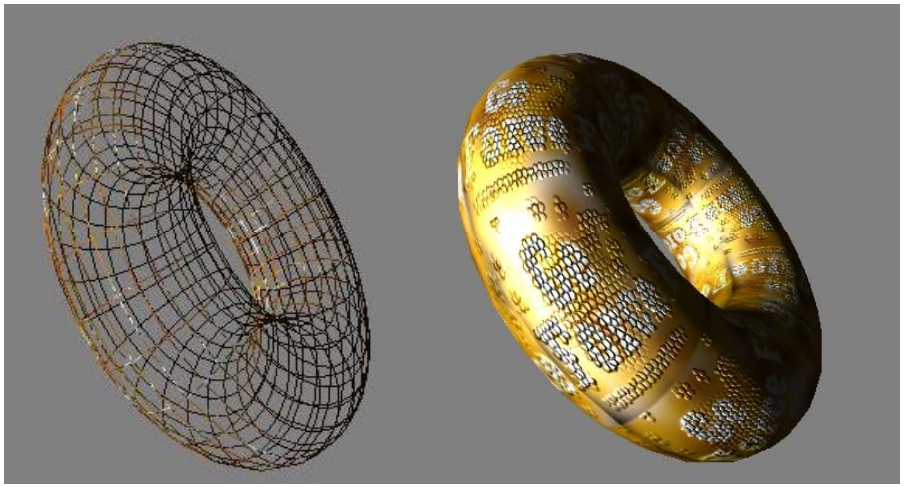
Texture Mapping

- To enhance the visual effect of an object, trade photo-realism for efficiency

  →**Texture Mapping:** texture wrapped on the object
- To add pseudo-realism to shiny animated objects by causing their surrounding environment to be reflected in them

  →**Environment mapping:** texture moved as the objects moved



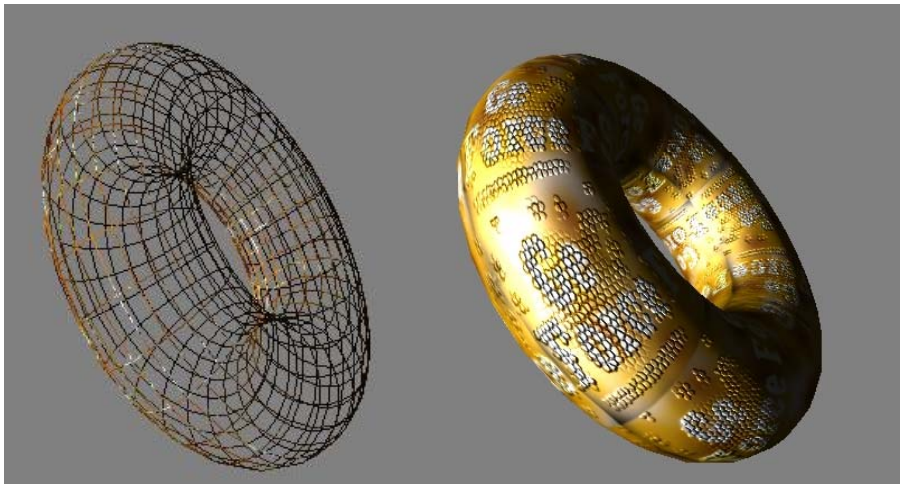


Various Mapping Techniques in CG

- 2D texture mapping
- Bump mapping
- Light maps
- Environment or reflection mapping

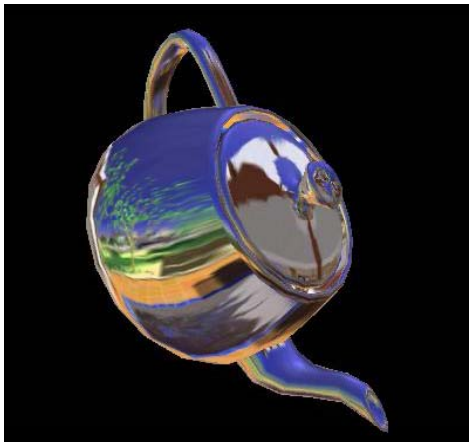
Store information in a domain → for the (later) rendering

- Texture Mapping is cheap, while global illumination computation is totally different and much more expensive



Texture Properties

What can be texture? What can be modulated with texture mapping?



- Color
- Specular Color
  - for environment reflection mapping
- Normal Vector Perturbation
  - Bump mapping
- Displacement along surface normal
  - Displacement mapping

- Transparency
  - Etched glass where a shiny surface is roughened (to cause opacity) with some decorative pattern

Mapping Among Different Spaces

Texture Space

(u,v)

Surface Parameterization

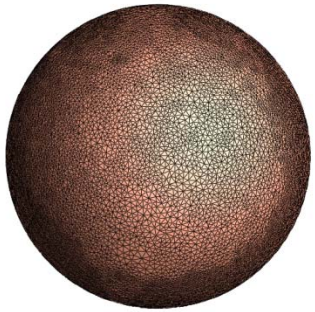
Geometry Space

(x,y,z)

Projection

Screen Space

(x,y)

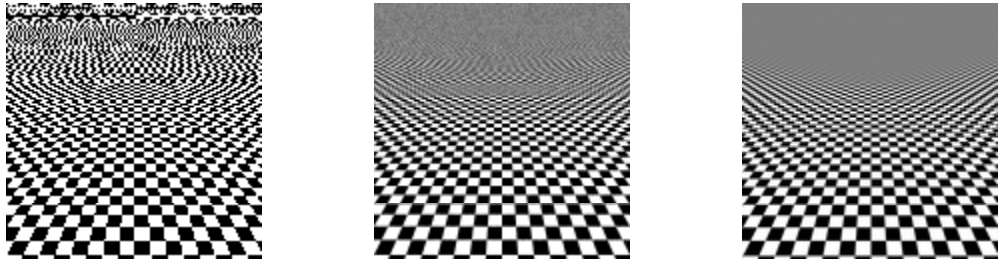


Mapping Structure

When a surface parameterization is ready…

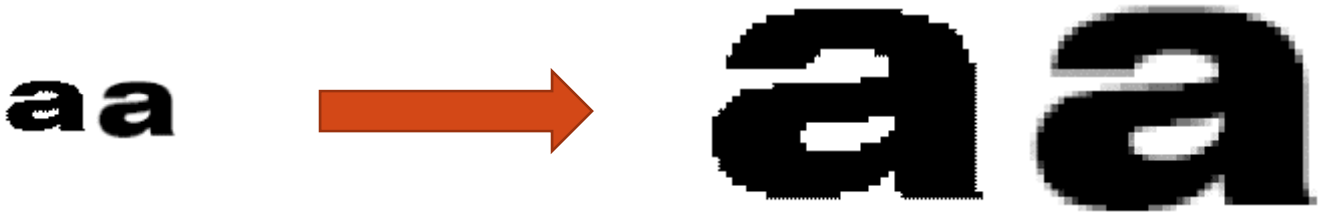
- Forward Mapping (more intuitive)
  - Start from texture space, a rubber sheet texture sticks on the surface
    e.g. : a square texel in texture space
    → curvilinear quadrilateral in screen space
- Inverse Mapping (more common)
  - Start from screen space, for each pixel, find its corresponding (u,v)
  - A square pixel produces a curvilinear quadrilateral as a pre-image
  - A filtering operation integrates the information contained in the pre-image and assigns the resulting color to the pixel (Anti-aliasing)

Anti-aliasing



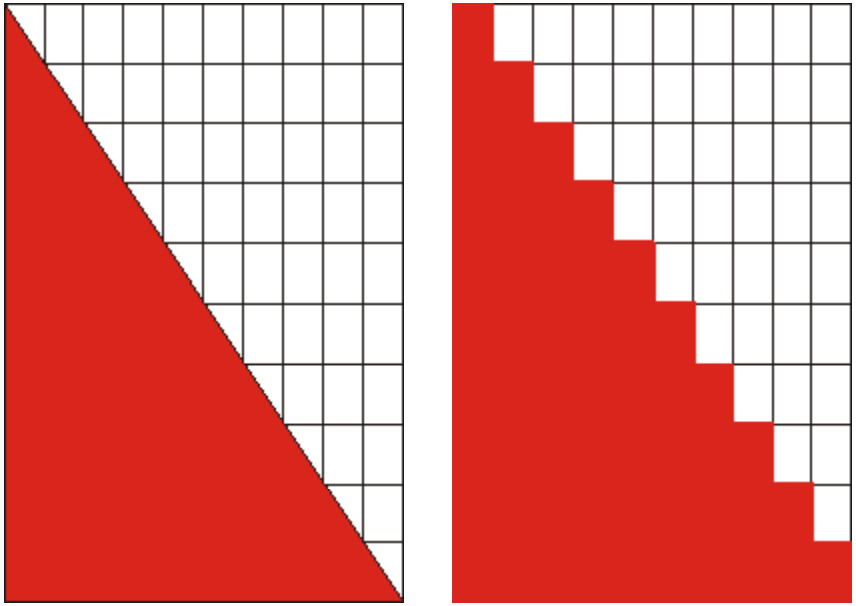
- Aliasing:
when representing a high-resolution signal at a lower resolution → distortion artifacts (usually wavy lines or bands…)

- Anti-aliasing:
for down-sampling, we filter the signal



Anti-aliasing (cont.)

- Why do we get jagged edges?

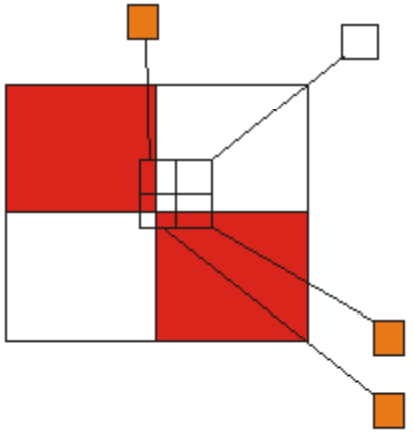


- Solution:
  → use subtle changes in color around the curved or diagonal area

Anti-aliasing (cont.)

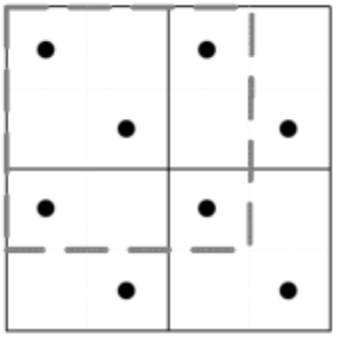
1. Supersampling

→A mathematical calculation to determine the average color for each pixel.
→best result, but slow



2. Multisampling

→More efficient, result less pretty for anti-aliasing
→A standard example: quincunx sample pattern: 4 corner pixels + 1 middle pixel



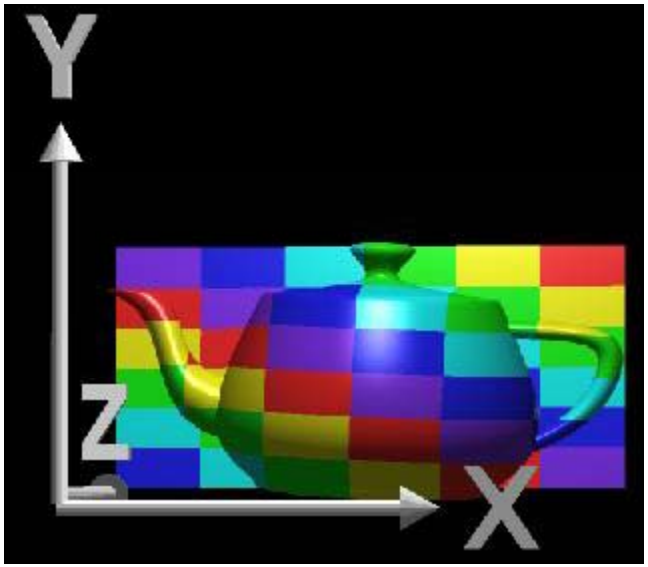
Quincunx sample pattern
each corner sample is given a weight of 1/8; sample at the center has a weight of 1/2

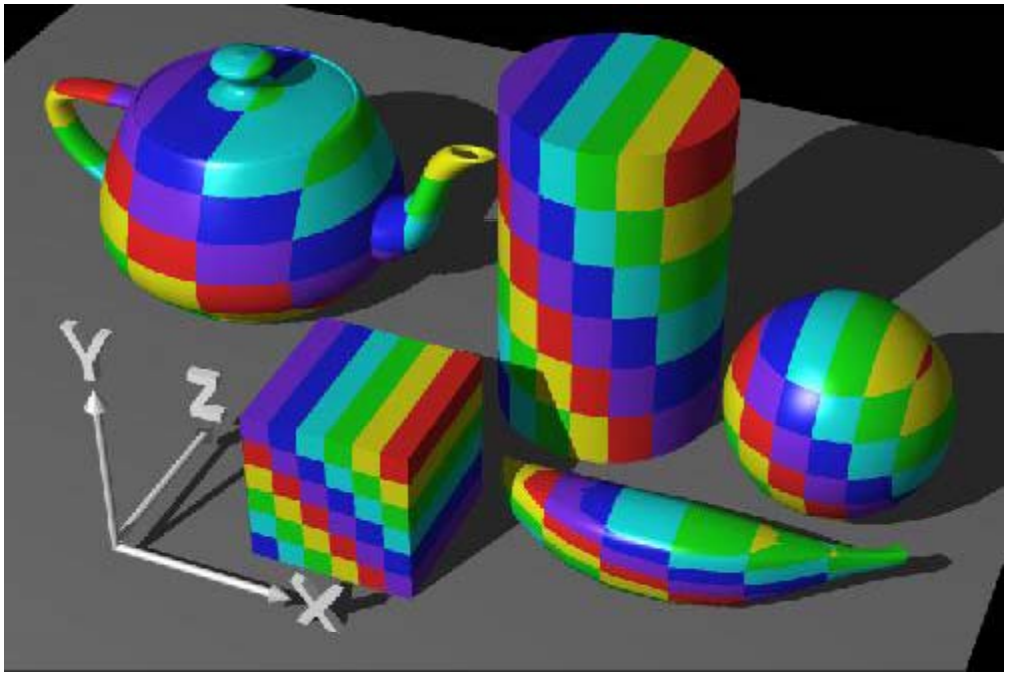
Intermediate Mapping Methods

- Based on projection
- Use an intermediate surface

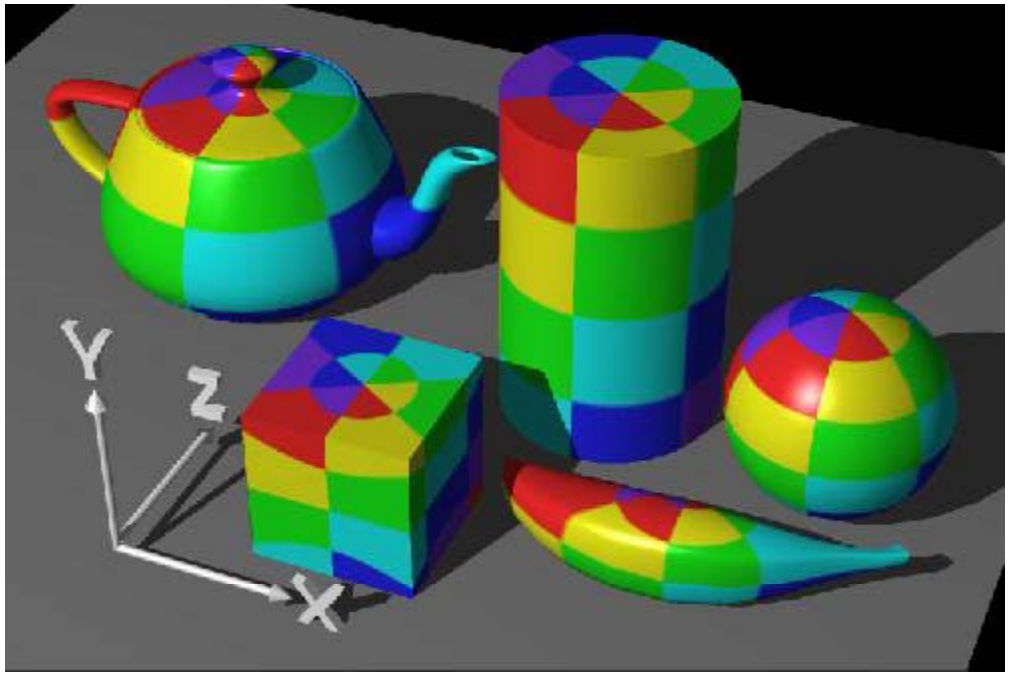
Examples of two-part texture mapping. The intermediate surfaces are (left) a plane; (middle) a cylinder; and (right) a sphere.

Plane Projection

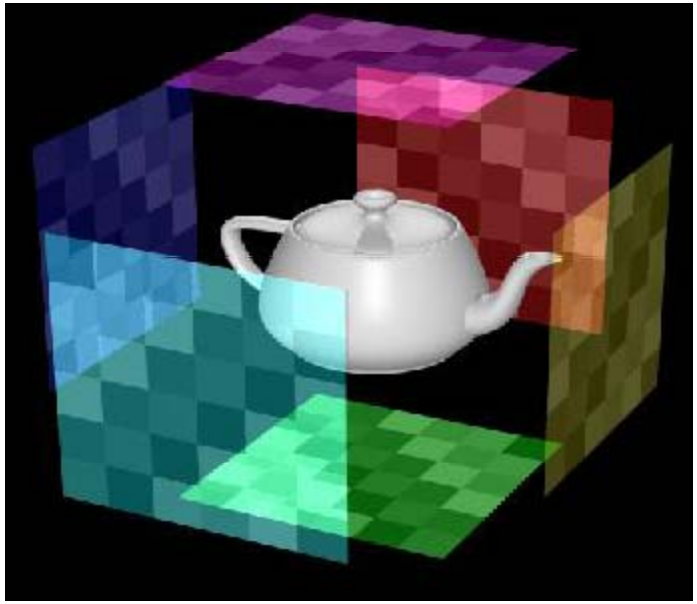


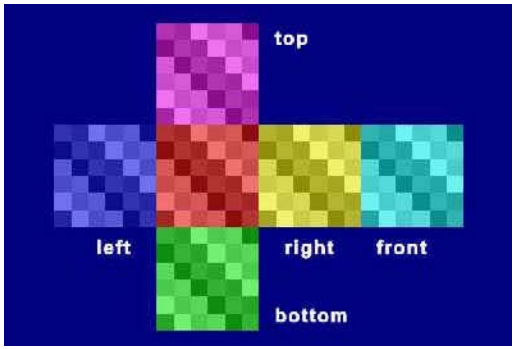


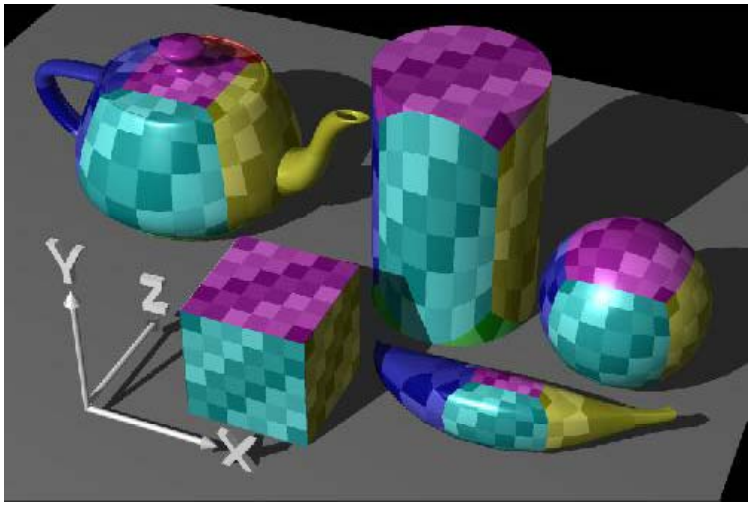
Cylinder Projection



Cube Projection







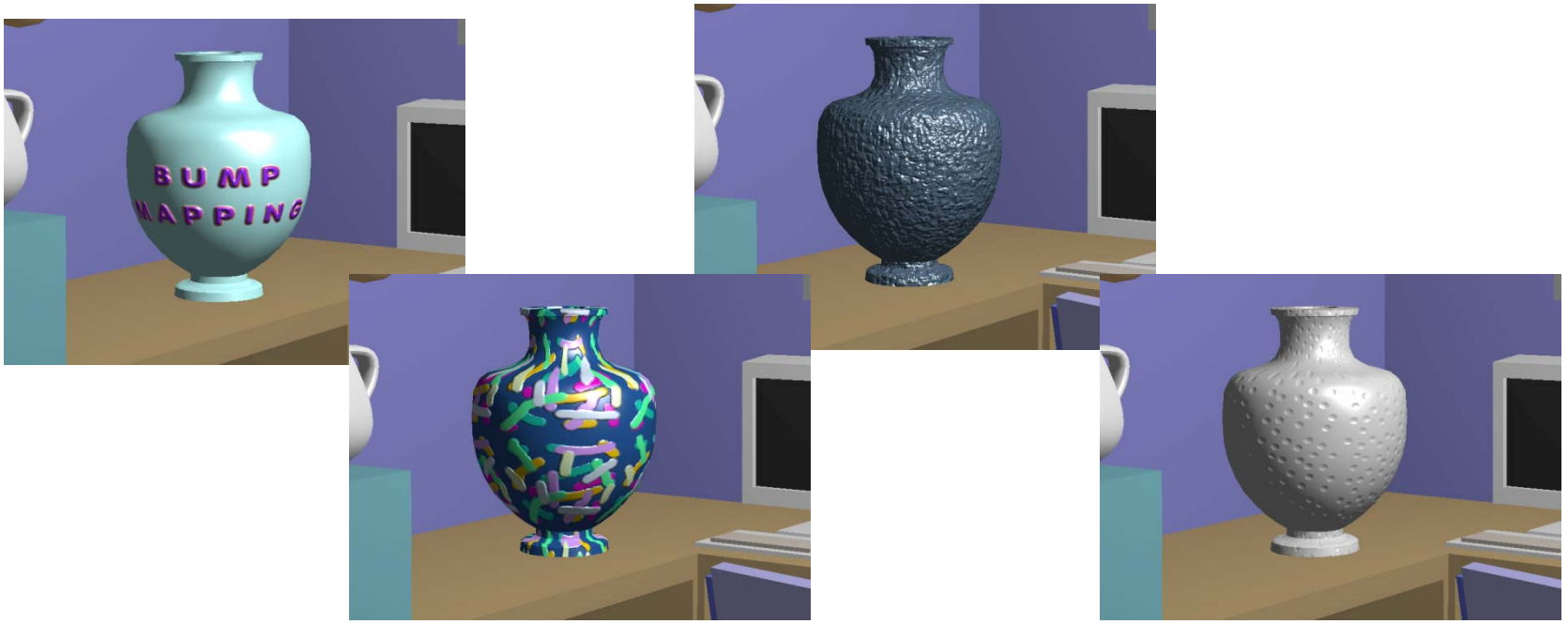
When will the projection method fail?

Solution: Intrinsic Parameterization Methods (later)

Bump Mapping and Normal Mapping

- Bump Mapping : to enable a (low resolution) surface to
  - Appear as if it were wrinkled or dimpled
  - Without the need to model these depressions geometrically

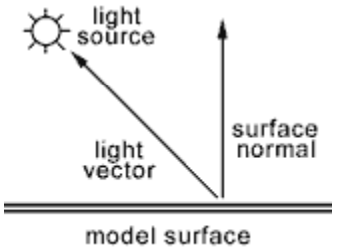
  →modify normal according to info in 2D bump map
  - Problem: geometry doesn't change, silhouette follows original geometry
- Normal mapping : use "normal" to enrich details



How does these normal work?

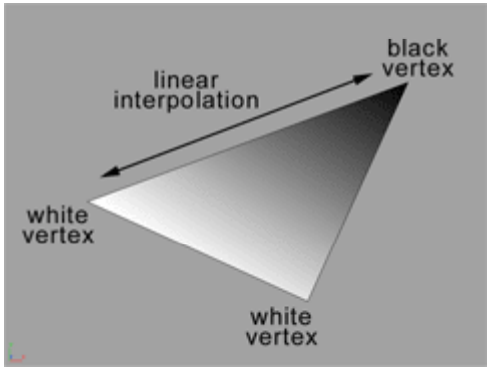
- About the lighting:
  - B = N • L

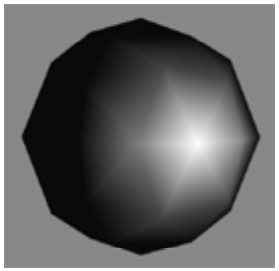
    (brightness ← normal dot product light vector)

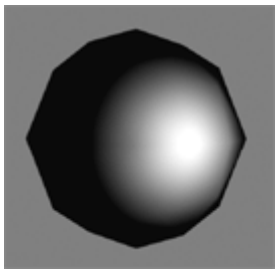


❑Gouraud shading: (in most real-time video game models)
- ➢only compute the normal and lighting on vertices
- ➢linear interpolate the lighting on interior pixels

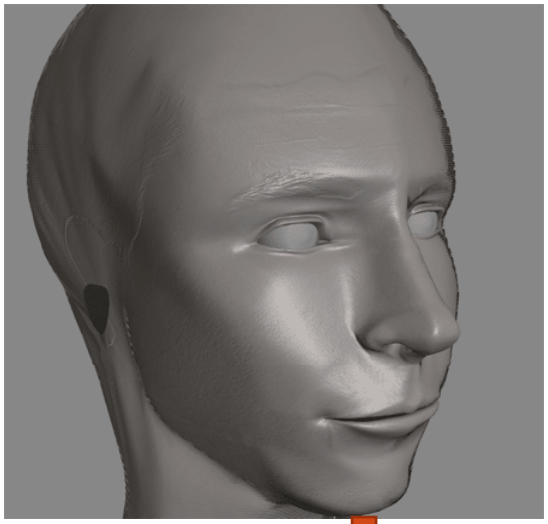
❑With normal texture:
- ➢per-pixel lighting (on each pixel, we have normal now)



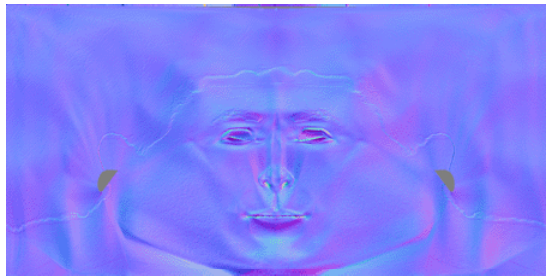




How does these normal work?

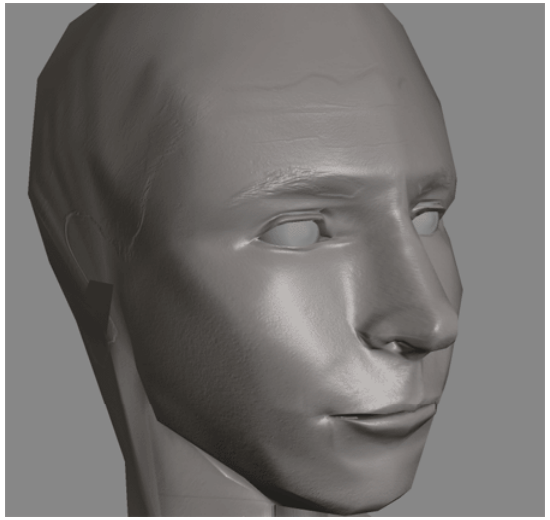


original mesh



normal as the texture

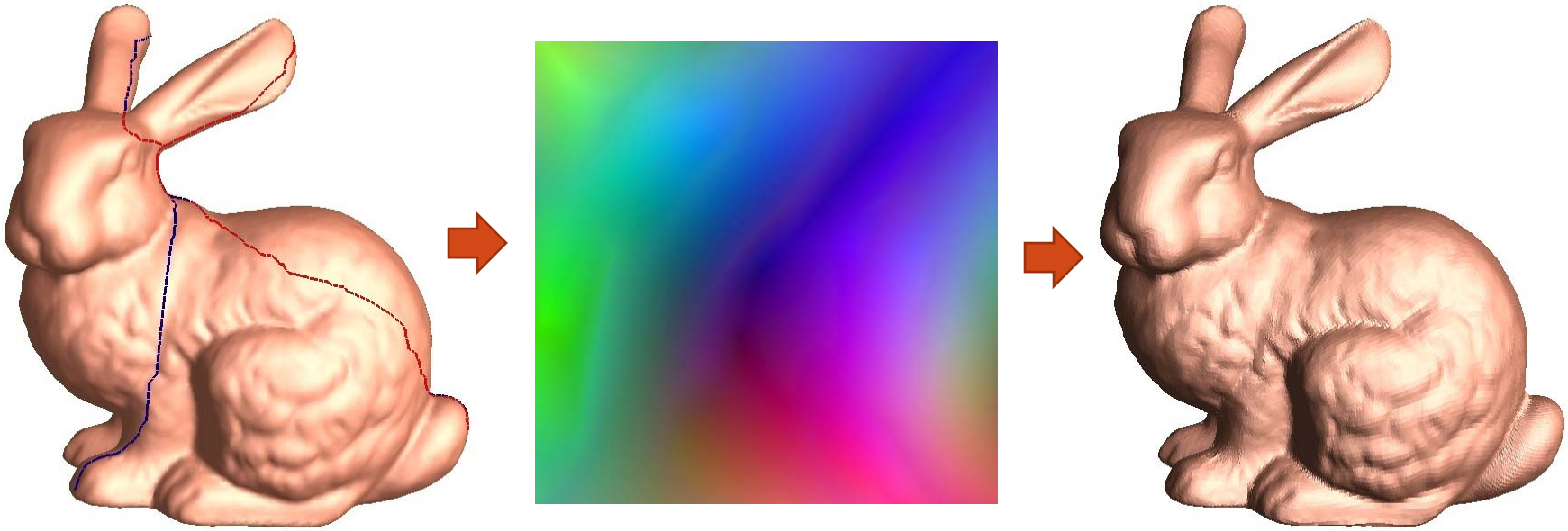
simplified mesh



texture mapped simplified mesh

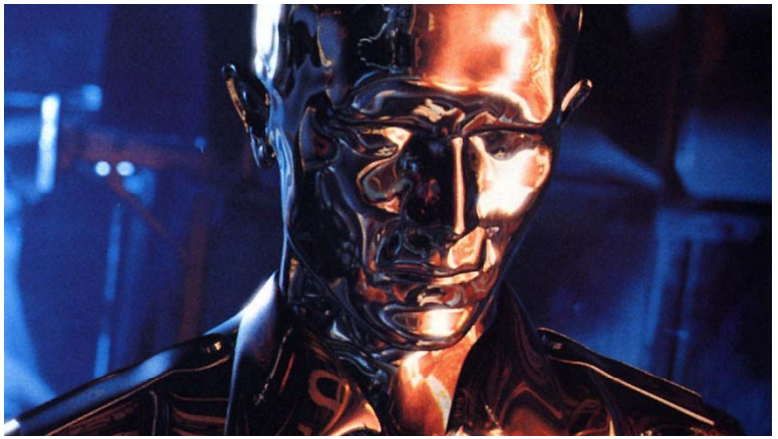
Geometry as texture

- Geometry Image [Gu, Gortler, and Hoppe, SIGGraph02]
  - Store the geometry (x,y,z) of each vertex → (r,g,b) in the texture space
  - Work for general surfaces, but need a cutting preprocess
    - Good cutting → less distortion
    - An intuitive topological method to generate the cut



Environment Mapping

--Some surfaces texture should reflect the surrounding
(example: Movie "Terminator")



- also called "reflection mapping"

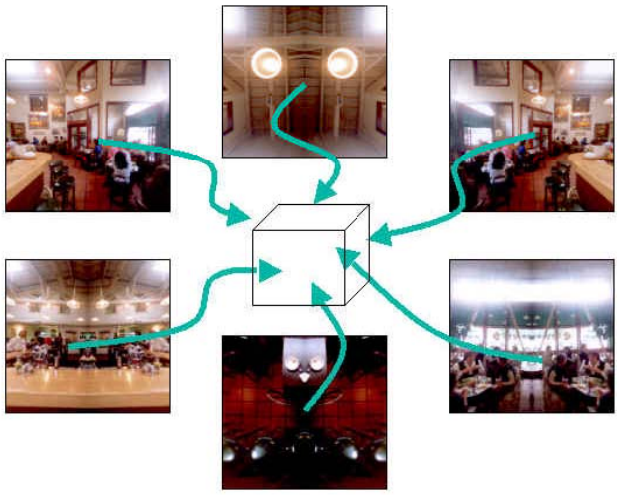
  → a shortcut to rendering shiny objects that reflect its surrounding environment
  - Ray tracing process → map construction (offline) + indexing (online)
  - Nearly every 3D computer game today uses this form of texture mapping

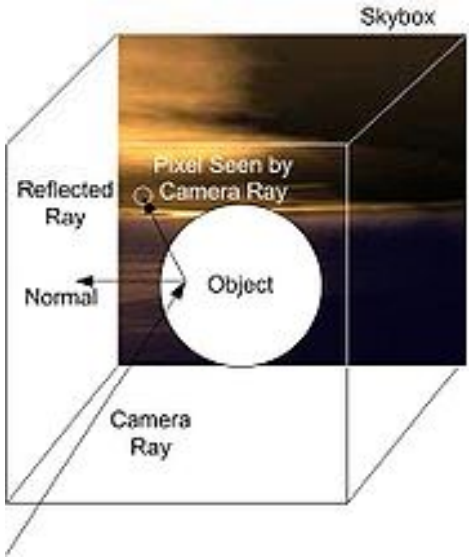
- Not a single image wrapped onto the surface:
  - when the viewer position changes, or the object moves → the reflection changes
  - ➢ should map surface points to an appropriate reflected direction in the 360 degree environment surrounding the object

Codes Demo

Environment Mapping (cont.)

- Common Mapping Techniques for Environment Mapping:
  - Sphere Mapping
  - Cubic Mapping
- Environment texture
  - Pre-computed and stored by projections
  - indexed by a 3D direction vector
- Problems:
  - Geometrically correct when objects is small w.r.t. the environment
  - Only reflect the environment – not itself → wrong for concave objects
  - Separate maps are required for different objects in one scene





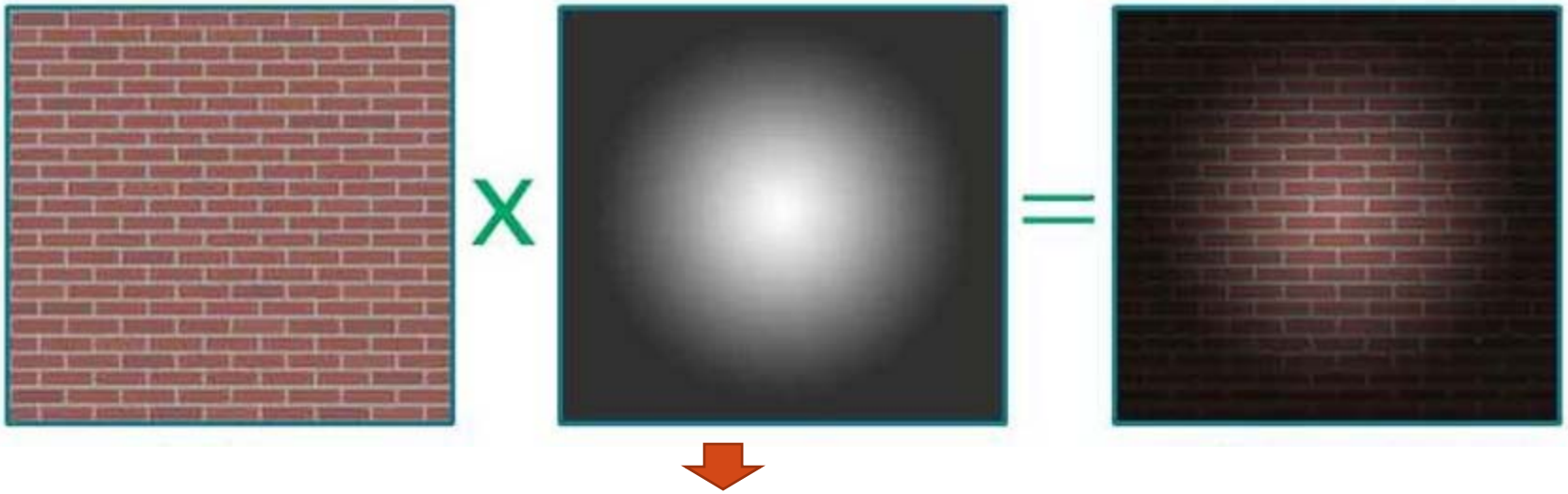
Light Mapping

- To enable lighting to be pre-calculated and stored as a two-dimensional texture map
  - Pre-compute:
    - Vertex brightness using distance from each vertex to a light
    - Pixel brightness using multitexture when texture map is also used
  - Shading → Indexing
  - Can stored separately from other texture maps with lower resolution

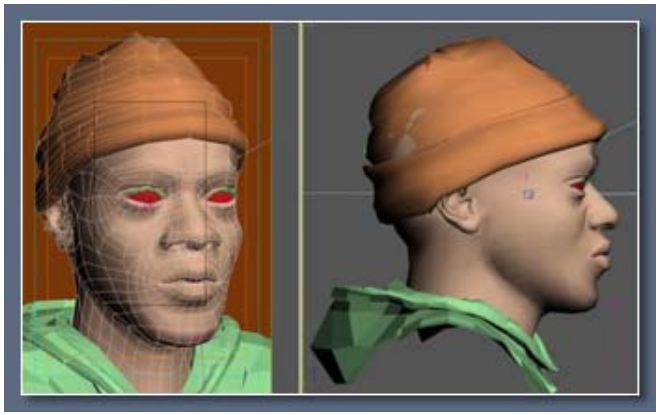
Example: Quake
(a famous first-person shooter video game)

❑ Moving objects? → multiple maps + interpolation

Light Mapping (cont.)



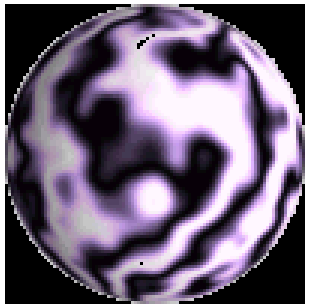
Difficulty in 2D Texture Mapping



Need low-distorted mapping, sometimes not easy to compute!

3D Texture

- Challenges for wrapping texture image onto surfaces (for 2D texture map):
  - Distortion control could be non-trivial
  - Topological discontinuity could be awkward
- Procedural texture
  - Define a continuous texture function over the whole R^3 space
  - Spatial Efficiency : functions instead of 3D texture images



...

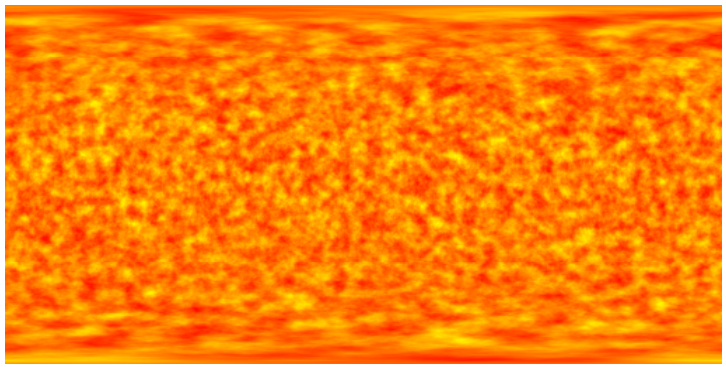
Check the paper: [Perlin Noise 1985]
– using noise function to simulate turbulence

3D Texture

- Challenges for wrapping texture image onto surfaces (for 2D texture map):
  - Distortion control could be non-trivial
  - Topological discontinuity could be awkward
- Procedural texture
  - Define a continuous texture function over the whole R^3 space
  - Spatial Efficiency : functions instead of 3D texture images
- Texture synthesis
  - Not a mapping problem any more
  - Less texture patterns (less resources)

3D Texture

- Challenges for wrapping texture image onto surfaces (for 2D texture map):
  - Distortion control could be non-trivial
  - Topological discontinuity could be awkward
- Procedural texture
  - Define a continuous texture function over the whole R^3 space
  - Spatial Efficiency
- Texture synthesis
  - Not a mapping problem any more
  - Less texture patterns (less resources) compared to 2D texture



3D Texture (cont.)

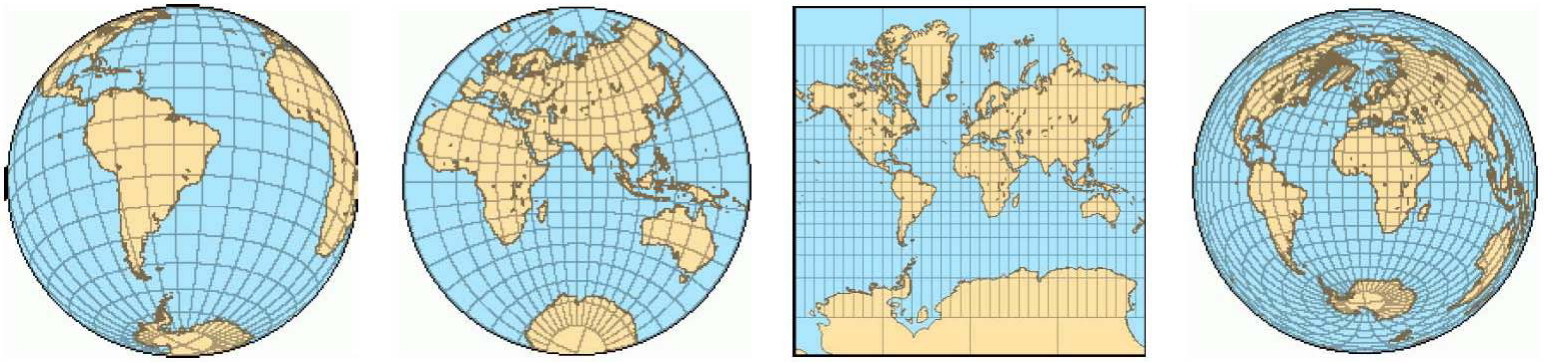
Another big visual difference:

-- Texture moving with the object

Later:

The surface mapping (parameterization) problem:

How to control the distortion? (What distortion?)
How about other issues – boundary continuity, poles?



Announcement

- No class this Thursday (Fall Break):
- No class next week:
  - Each project team:
    1) please meet and discuss your course project
    2) let me know your project topic
    3) prepare your slides
- Before Monday October 12th:
    1) send me your report (algorithm description, plan/schedule, and task division)
    2) send me your presentation slides
- Tuesday October 13th : mid-term presentations
- Orders of presentations