Texture and bump mapping

COM3404

Richard Everson

School of Engineering, Computer Science and Mathematics
University of Exeter

R.M.Everson@exeter.ac.uk
http://www.secamlocal.ex.ac.uk/studyres/COM304

Outline

1. Texture mapping
   - Anti-aliasing and MIP maps
   - Reflection mapping
2. Bump mapping

References

- Fundamentals of 3D Computer Graphics. Watt. Chapters 4, 5 & 6.
- Computer Graphics: Principles and Practice. Foley et al (1995). Chapters 15 & 16.
- Teaching Texture Mapping Visually. Rosalee Wolfe (1997) http://www.siggraph.org/education/materials/HyperGraph/mapping/r_wolfe/r_wolfe_mapping_1.htm

Texture mapping

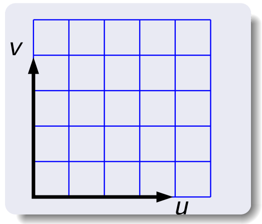
Map a two-dimensional texture (image) onto the surface of a three-dimensional object.

- Texture is two-dimensional. Coordinates (u, v).
- Mapped surface is two-dimensional. Coordinates (x_w, y_w, z_w)
- Screen is two-dimensional. Coordinates (x_s, y_s).

Example: Texture mapping a sphere

Equation of a sphere

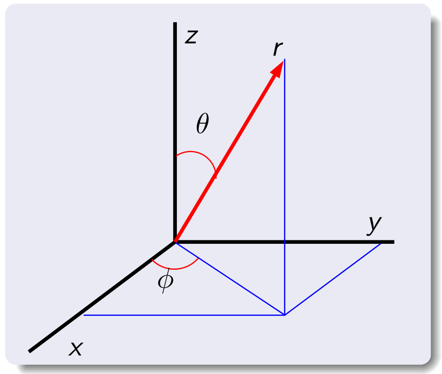
$$x_w = R\sin(\theta)\cos(\phi)$$
$$y_w = R\sin(\theta)\cos(\phi)$$
$$z_w = R\cos(\theta)$$



θ is polar angle from z axis
ϕ is azimuthal angle from x axis

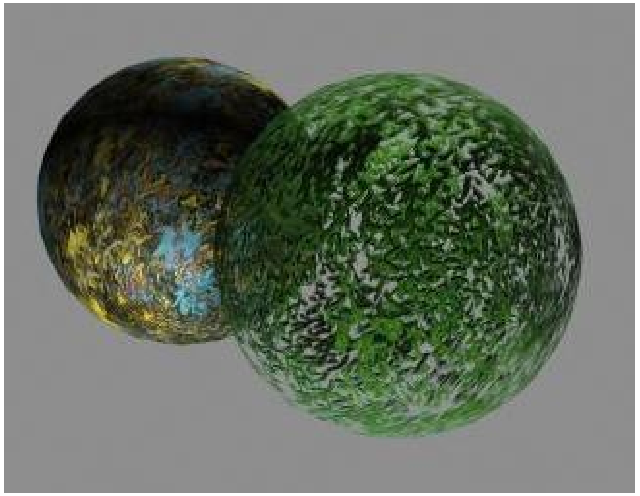
Mapping Put $\theta = \pi v$ and $\phi = 2\pi u$

$$v = \theta/\pi = \arccos(z_w/R)/\pi$$
$$u = \arccos(x_w/R\sin(\pi v))/(2\pi)$$



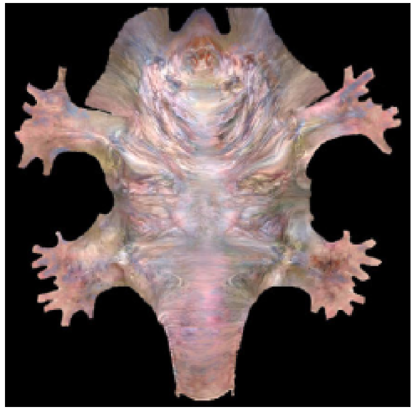
Can compute (u, v) and therefore shading for any point given world coordinates.

Texture mapping a sphere



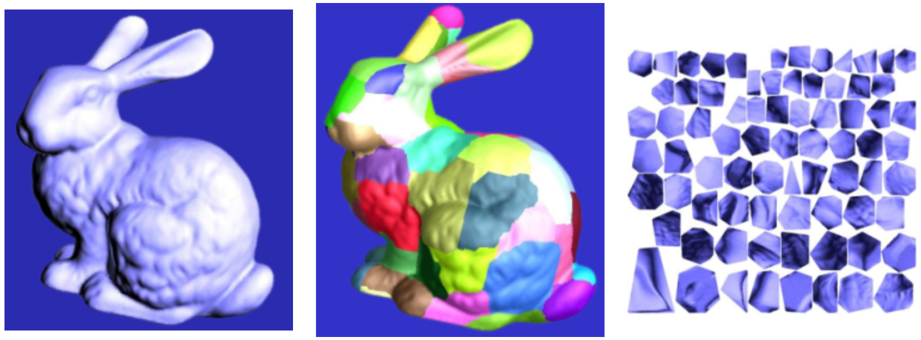
Texture and opacity maps

Defining the texture-object mapping: Pelting



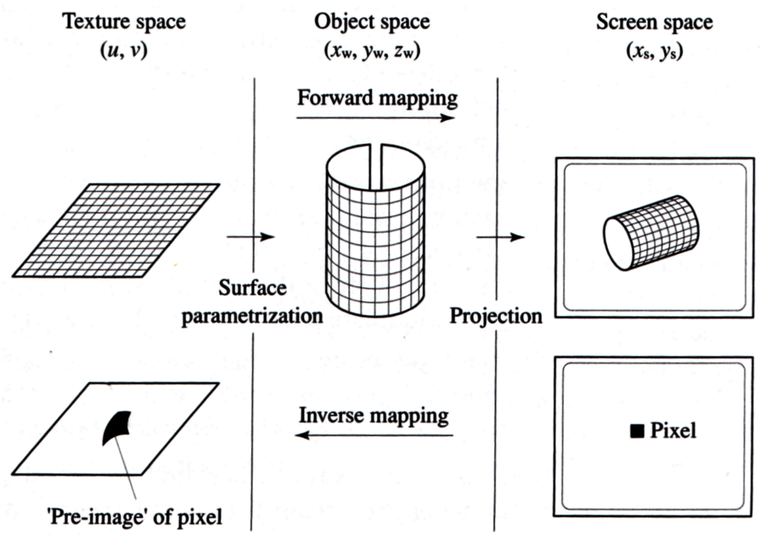
Piponi & Borshukov, *Seamless texture mapping of subdivision surfaces by model pelting and texture blending*, SIGGRAPH, 2000

Defining the texture-object mapping: Atlas



Sander, Snyder, Gortler, Hoppe. *Texture Mapping Progressive Meshes,* SIGGRAPH, 2001

Forward and inverse mapping



Mapping $(u, v) \mapsto (x_s, y_s)$ is two-dimensional to two-dimensional

Forward and inverse mapping

Mapping from texture to screen is non-linear

- A single pixel's pre-image may be several texels
- A single texel may lie entirely within a screen pixel.

Forward

- Simple to comprehend

Inverse mapping

- Suited to scan-line algorithms
- Efficient: only required textures are computed
- If pre-image of pixel covers several texels then texture should be super-sampled.
- If pre-image of pixel lies within a single texel then low-pass filtering required to prevent anti-aliasing.

Bi-linear interpolation

Model (approximate) the mapping $(u, v) \mapsto (x_s, y_s)$ as a rational linear projective transform

$$x = \frac{au + bv + c}{gu + hv + i} \qquad y = \frac{du + ev + f}{gu + hv + i}$$

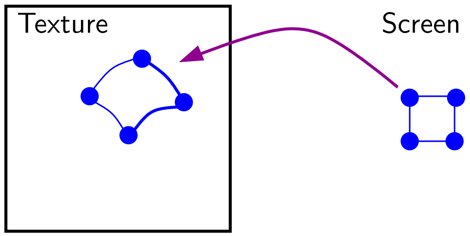
Inverse mapping in homogeneous coordinates:

$$\begin{bmatrix} u' \\ v' \\ q \end{bmatrix} = \begin{bmatrix} ei - fh & ch - bi & bf - ce \\ fg - di & ai - cg & cd - af \\ dh - eg & bg - ah & ae - bd \end{bmatrix} \begin{bmatrix} x'_s \\ y'_s \\ w \end{bmatrix}$$

with $(x_s, y_s) = (x'_s/w, y'_s/w)$ and $(u, v) = (u'/q, v'/q)$

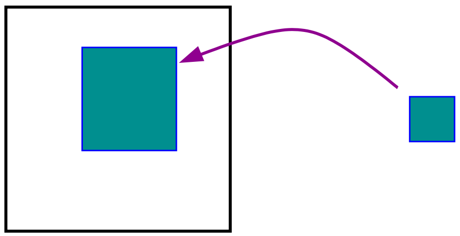
- Coefficients a to i found by solving equations for the mapping of four corners of pixel in screen coordinates to quadrilateral in texture coordinates.

Anti-aliasing and MIP maps



Approximate pre-image of pixels in texture space by a square of size 2^n

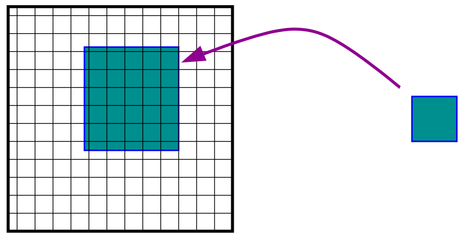
- Permits precalculation for anti-aliasing



Anti-aliasing and MIP maps

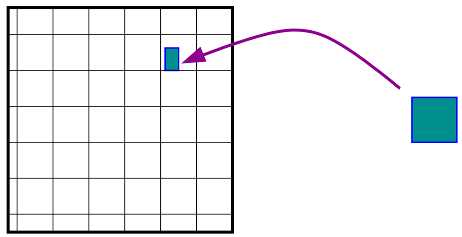
Compression from texture space to screen space

- Several texels map to a single pixel
- To avoid aliasing errors, texture image should be sampled many times to calculate pixels, but very expensive.



Magnification from texture space to screen space

- Single pixel lies within texel
- Higher resolution texture map required



MIP mapping for compression

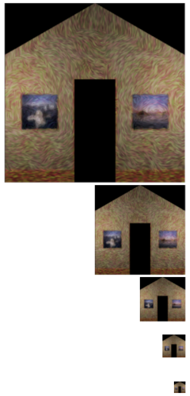
Precompute subsampled copies of the texture map for sizes $2^n, 2^{n-1}, \dots 1$

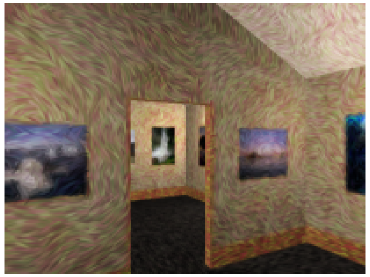
Select a texel from the appropriately sampled map

- surface is close to camera, use high resolution map
- surface is distant from camera, use low resolution map

Linear interpolation between the levels improves quality

Cost is constant per pixel





Reflection mapping

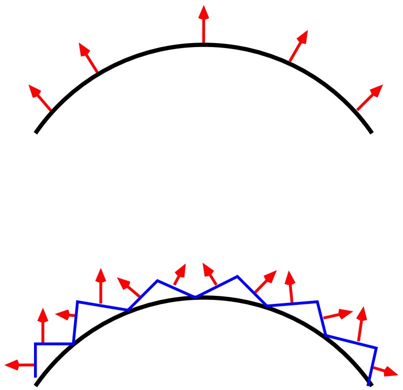
1. Render the scene from viewpoint of the reflecting surface.
2. Texture map scene onto reflecting surface.
3. Render scene from usual viewpoint.

Models only a *single* reflection, rather than reflections from multiple surfaces.

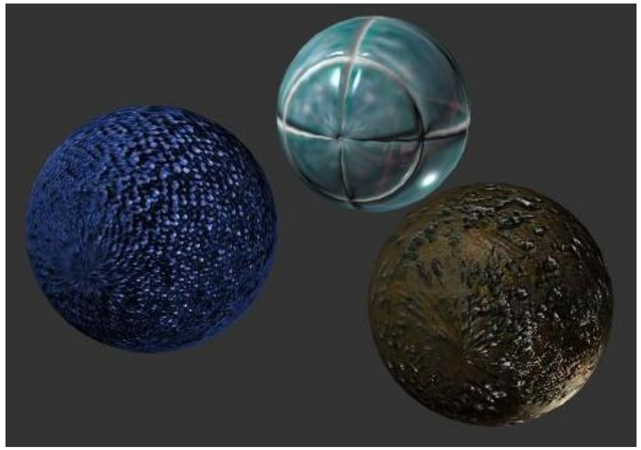
Bump mapping

Simulate height variations by adjusting directions of surface normals

- Perception of surface height depends on lighting
- Phong and Gouraud lighting models depend on surface normal to set intensity
- Adjust surface normal directions according to a *bump map* describing bumps to be simulated



Bump mapping



- Silhouette of spheres is smooth!