Texture Mapping

Overview

- Modeling with textures
  - motivation
  - texture coordinates
- Texture mapping implementation
- Anti-aliasing and level of detail

Texture mapping

- Model surface-detail with images
  - wrap objects with photographs
  - model and render color or “flat” detail
  - does not capture 3D detail

Texture mapping example

- Model t-shirt with logo
  - no need to model the letters and engine with triangles
  - use large base polygon
  - color it with the photograph

Texture mapping example

- Subtle wall lighting
  - no need to compute it at every frame
  - no need to model it with a lot of constant color triangles

Texture mapping example

- Subtle wall lighting
  - paste photograph on large polygon

Texture mapping example

- Non-planar surfaces work also

Texture mapping example

- Non-planar surfaces work also
  - subdivide surface into planar patches
  - assign photograph subregions to each individual patch

Texture mapping example

- Non-planar surfaces work also
  - subdivide surface into planar patches
  - assign photograph subregions to each individual patch

9

Texture mapping example

- Non-planar surfaces work also
  - subdivide surface into planar patches
  - assign photograph subregions to each individual patch

10

Texture mapping example

bark

veneer

bricks

- Generic image to represent material
  - tile pattern to cover big surface

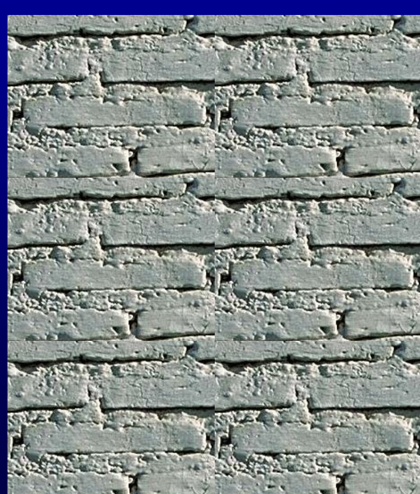
Tiling

- Repeat pattern

Tiling

- Repeat pattern

Tiling



- Repeat pattern

Tiling

- Repeat pattern
  – reduce seems by mirroring

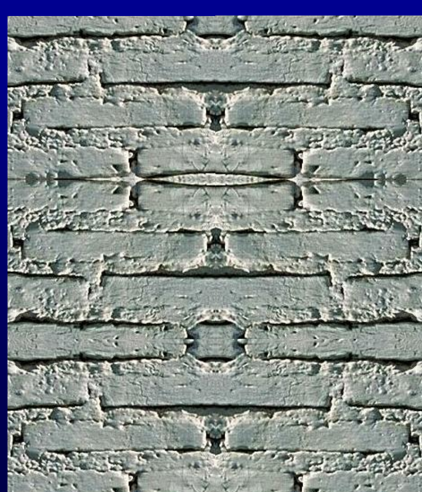
Tiling

- Repeat pattern
  – reduce seems by mirroring

Tiling

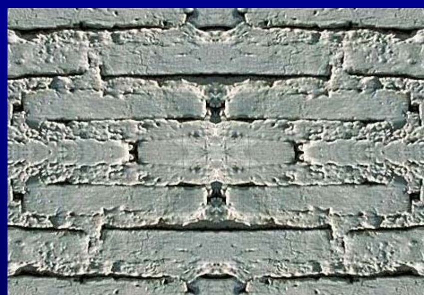
- Repeat pattern
  - reduce seems by mirroring

Tiling



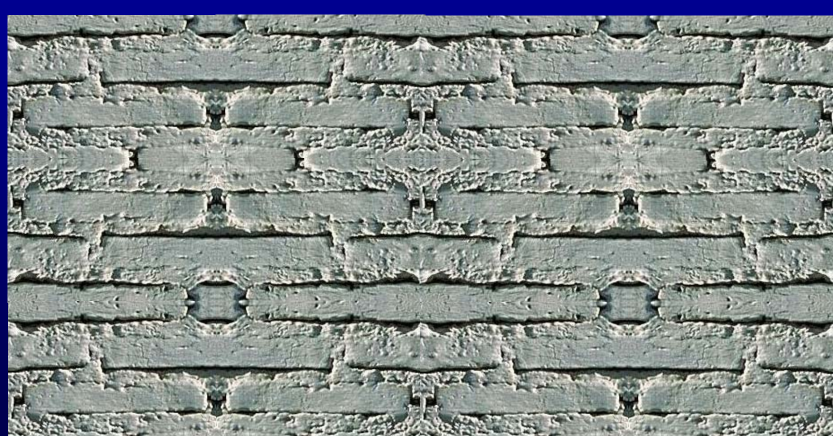
- Repeat pattern
  - reduce seems by mirroring

Tiling



- Repeat pattern
  - reduce seems by mirroring
  - reduce seems by choosing tile that covers one period of repeated texture

Tiling



Texture mapping limitations

21

Bricks are similar not identical

22

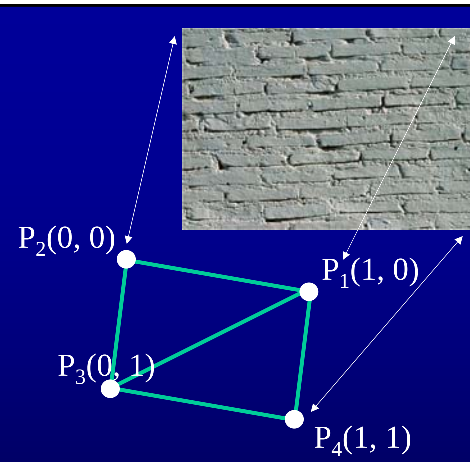
Texture mapping limitations

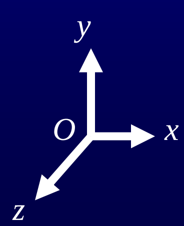
- Shiny floor
  - reflection is view dependent

Texture coordinates

- Mechanism for attaching the texture map to the surface modeled
  - a pair of floats (s, t) for each triangle vertex
  - corners of the image are (0, 0), (0, 1), (1, 1), and (1, 0)
  - tiling indicated with tex. coords. > 1
  - *texels* – color samples in texture maps

Texture coordinates





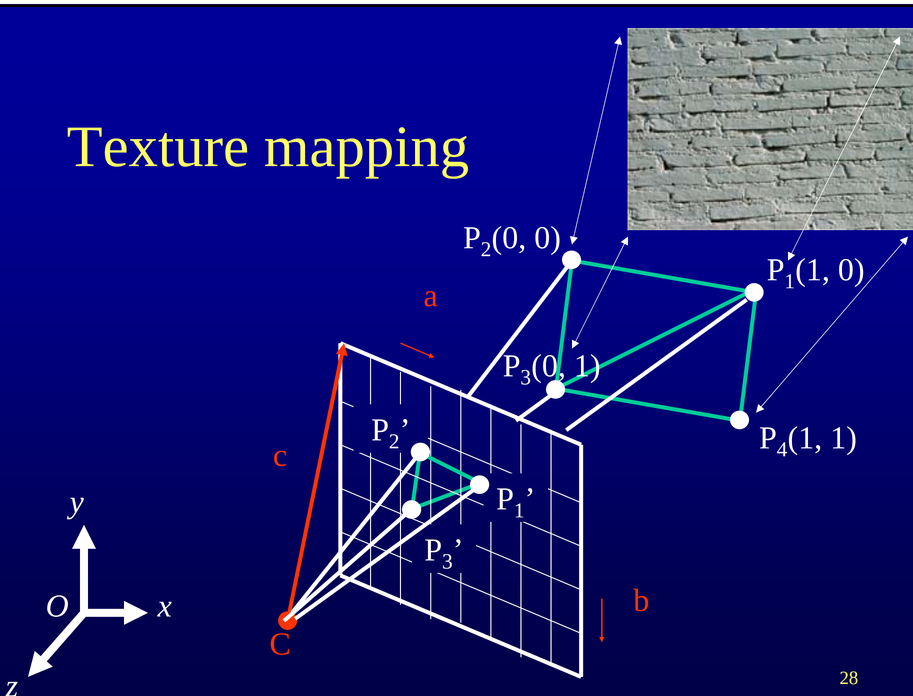
Overview

- Modeling with textures
  - motivation
  - texture coordinates
- Texture mapping implementation
- Anti-aliasing and level of detail

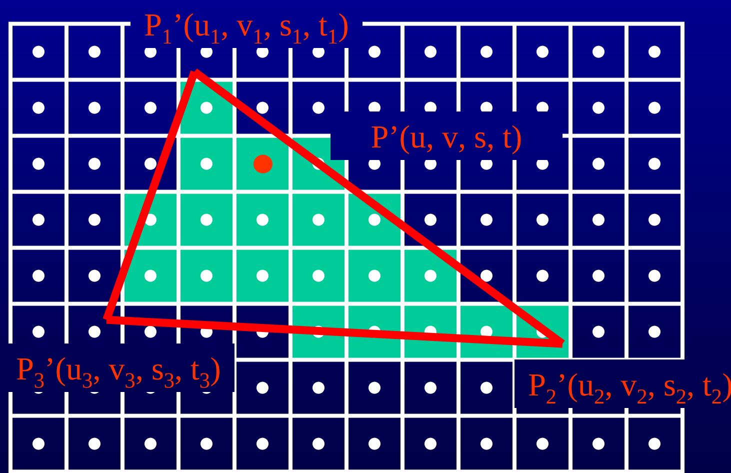
Overview

- Modeling with textures
  - motivation
  - texture coordinates
- Texture mapping implementation
- Anti-aliasing and level of detail

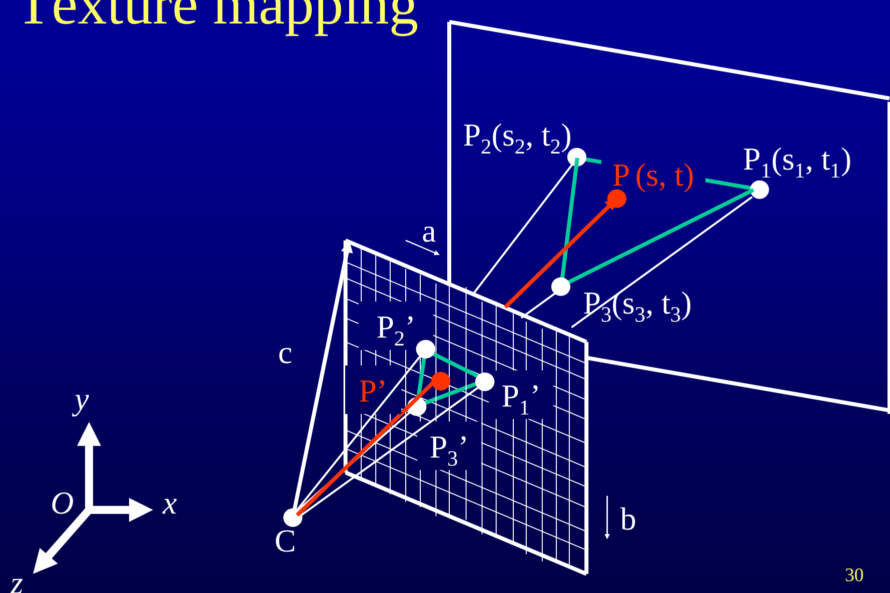
Texture mapping



Find texel for current pixel

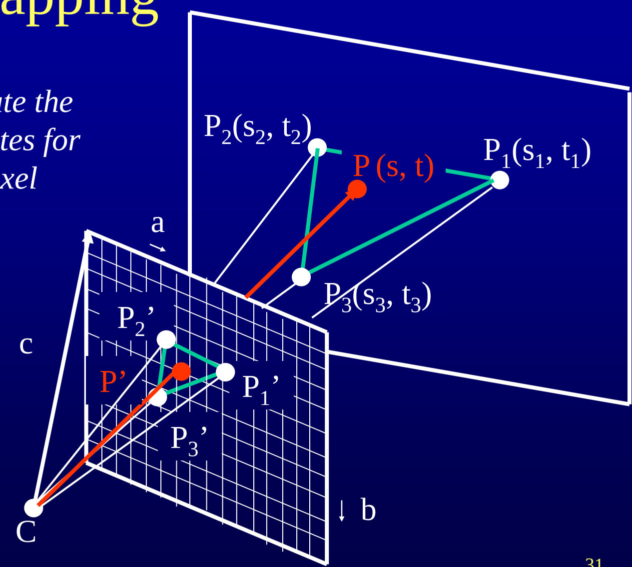


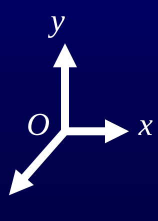
Texture mapping



Texture mapping

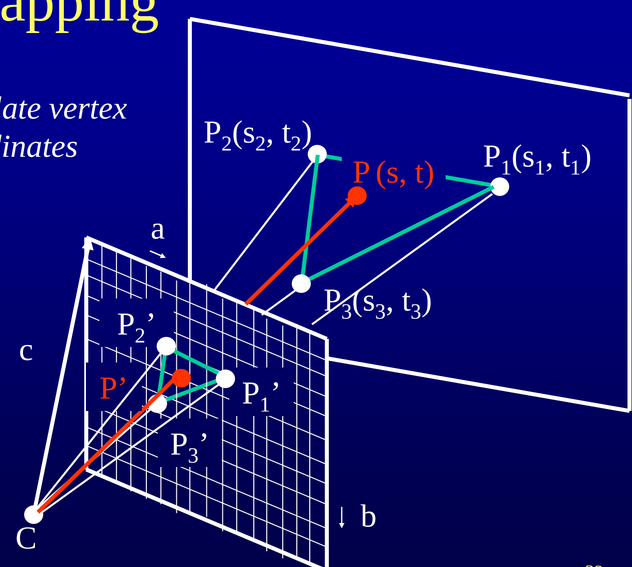
problem: *compute the texture coordinates for an interior pixel*

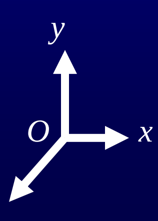




Texture mapping

solution: *interpolate vertex texture coordinates*





Overview

- Modeling with textures
  - motivation
  - texture coordinates
- Texture mapping implementation
- Anti-aliasing and level of detail

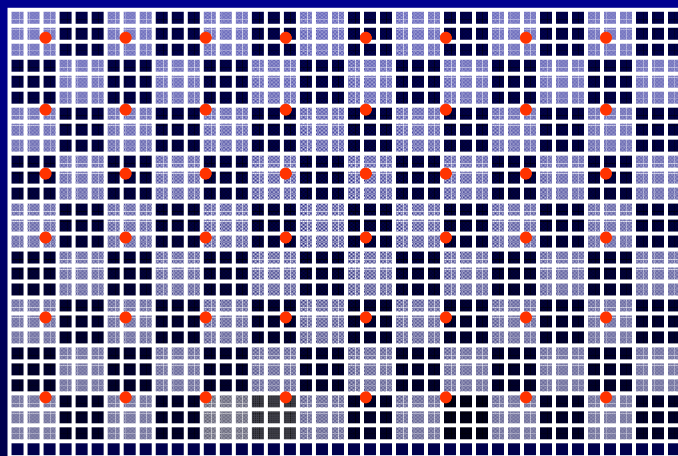
Overview

- Modeling with textures
  - motivation
  - texture coordinates
- Texture mapping implementation
- Anti-aliasing and level of detail

Aliasing

- sampling locations (desired image pixel centers)

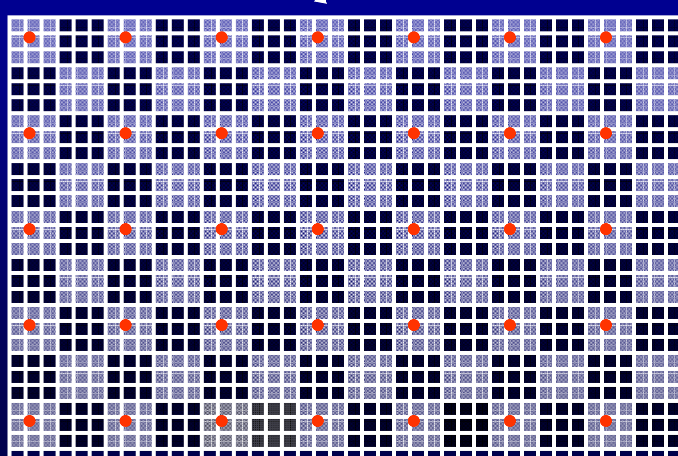
texture map



Aliasing

- sampling locations (desired image pixel centers)

texture map

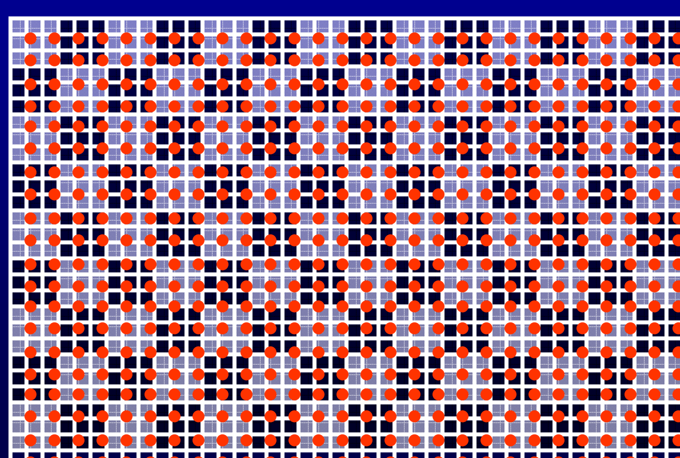


Aliasing

- High frequencies pose as lower frequencies
  - display resolution gives maximum displayable frequency
  - if not sufficient high frequencies are called (aliased as) low frequencies
- Nyquist law
  - max frequency displayable is half the sampling frequency

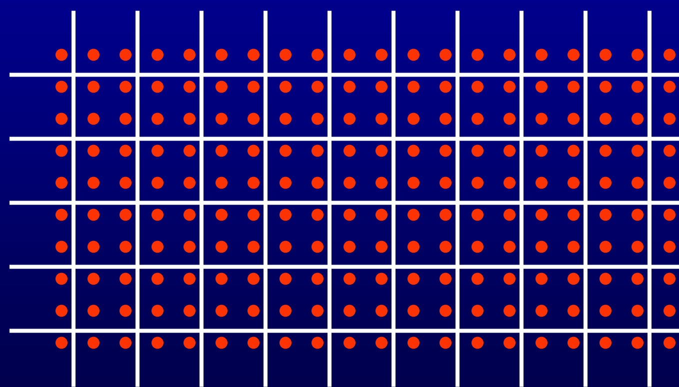
Nyquist law

for given texture map, one should sample at least this frequently to avoid aliasing



Nyquist law

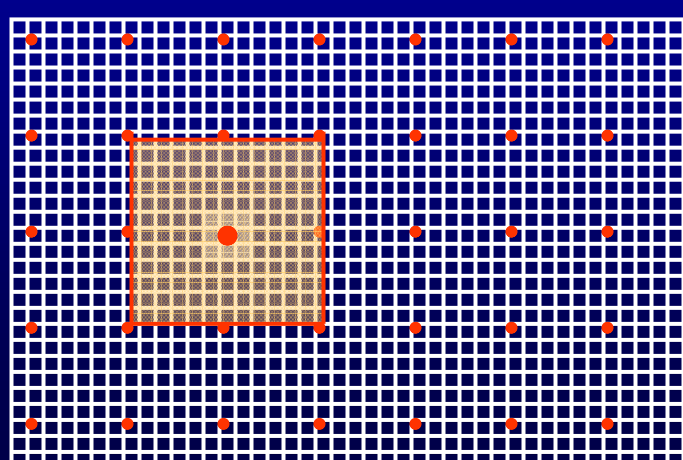
for given texture map *resolution*, one should sample at least this frequently to avoid aliasing



Possible Antialiasing Solution

convolution with 2 *output*-pixel-wide kernel

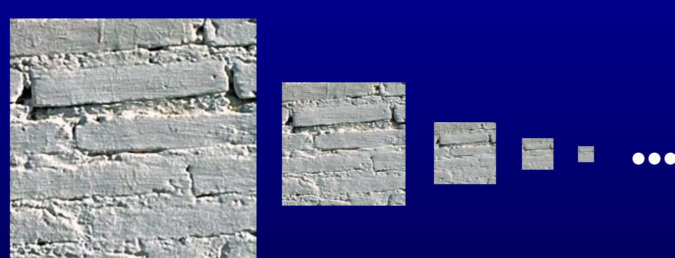
EXPENSIVE



Level of detail

- Adapt texture resolution to desired image resolution
- Mip-mapping
  - – texture is filtered as preprocess to several resolutions
  - – at runtime
    - find out required resolution
    - use corresponding version of texture map

Mip-mapping: example

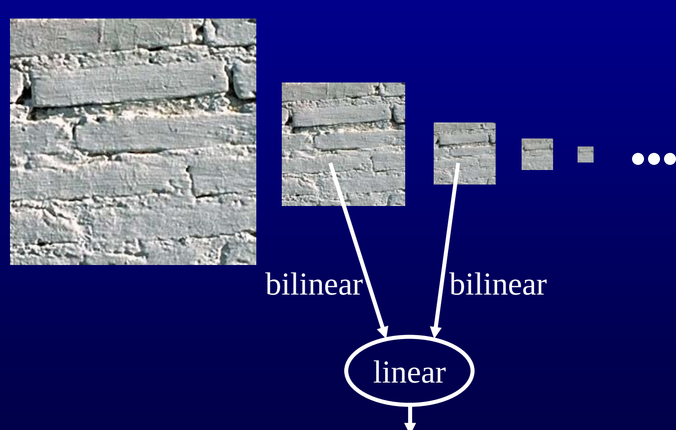


256x256 128x128 64x64 …

Trilinear filtering

- Use two most appropriate resolutions of texture map
  - lookup color values with bilinear interpolation in each texture version
  - linearly interpolate in between the two color values

Trilinear filtering: example



Anisotropic filtering

- Different levels of detail are needed along the two directions in the texture map
  - filter differently along s and t

desired image view of polygon

63

312

• trilinear interpolation between levels 32 and 64 blurs too much along the s direction

• use levels 256 – 512 for s and 32 – 64 for t