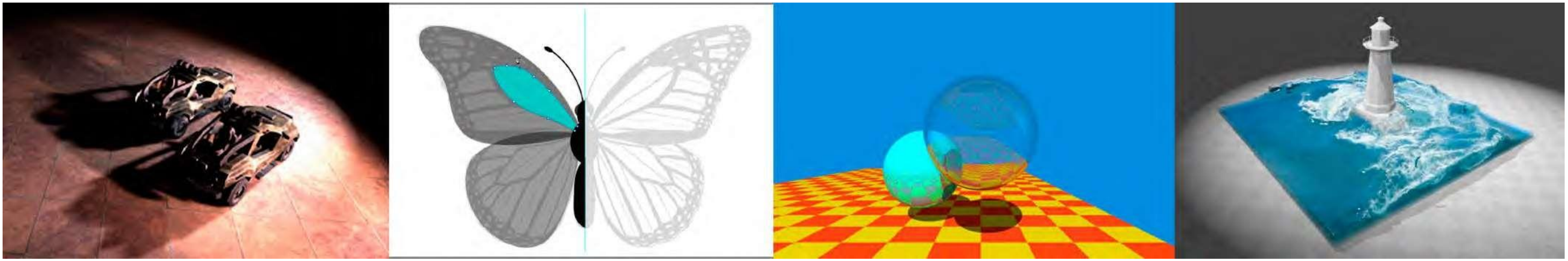
Computer Graphics

Shading (Texture Mapping cont.)



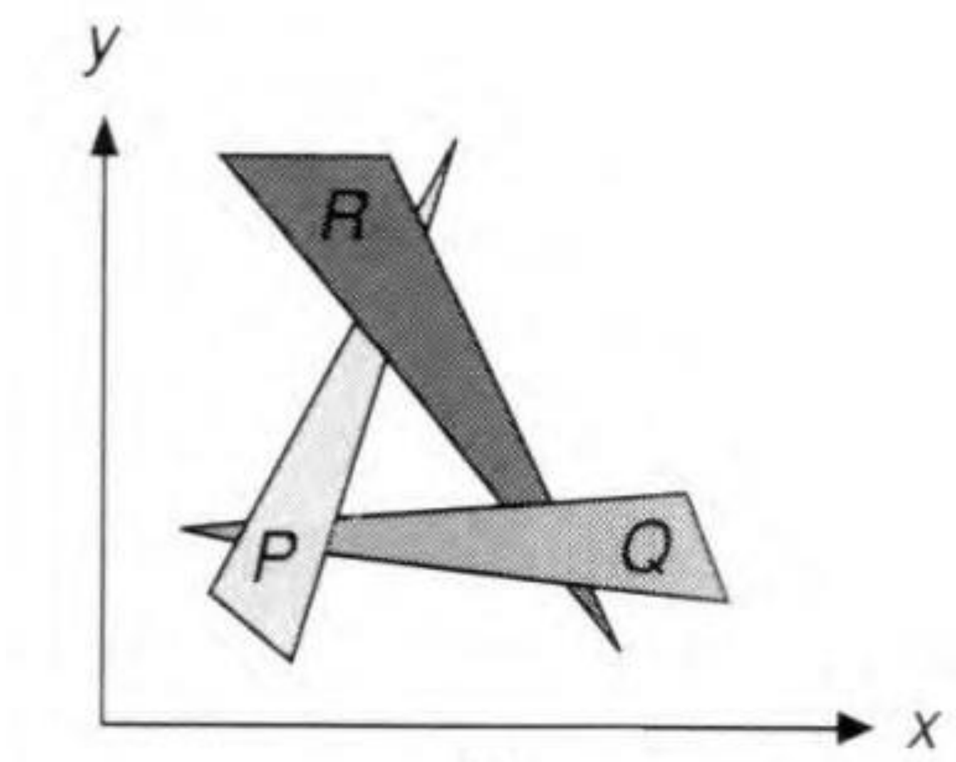
Last Lectures

- Visibility / occlusion
  - Z-buffering
- Shading
  - Blinn-Phong reflectance model
  - At a specific shading point
  - Shading frequencies
  - Graphics pipeline
  - Texture mapping
  - Barycentric coordinates

Painter's Algorithm

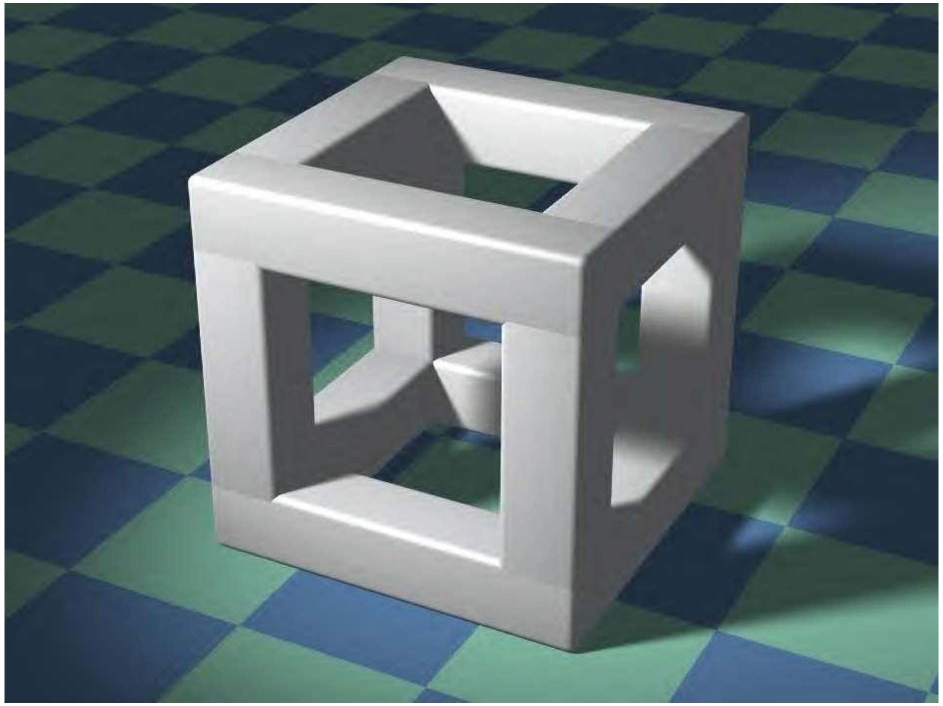
Requires sorting in depth (O(n log n) for n triangles)

Can have unresolvable depth order

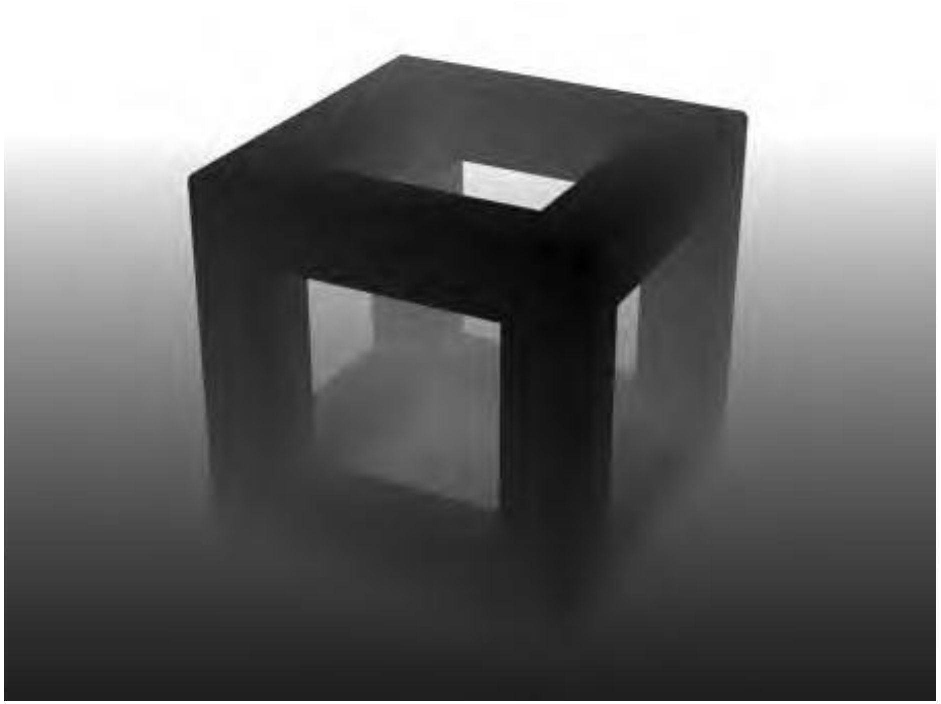


[Foley et al.]

Z-Buffer Example



Rendering



Depth / Zbuffer

Image source: Dominic Alves, flickr.

Z-Buffer Complexity

Complexity

- O(n) for n triangles (assuming constant coverage)
- How is it possible to sort n triangles in linear time?

Drawing triangles in different orders?

Most important visibility algorithm

- Implemented in hardware for all GPUs

Shading: Definition

- In Merriam-Webster Dictionary

  shad·ing, [ˈʃeɪdɪŋ], noun
  The darkening or coloring of an illustration or diagram with parallel lines or a block of color.

  使用平行线或色块使得插图或图案变暗或着色

- In this course

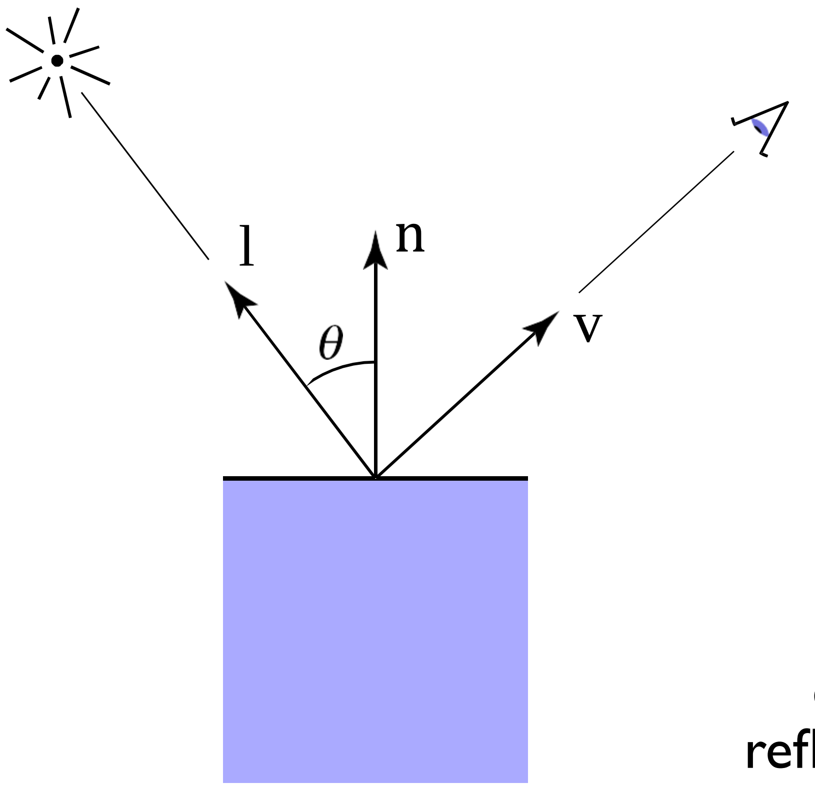
  The process of applying a material to an object.

Perceptual Observations

Photo credit: Jessica Andrews, flickr

Lambertian (Diffuse) Shading

Shading independent of view direction



energy arrived
at the shading point

$$L_d = k_d\,(I/r^2)\,\max(0, \mathbf{n} \cdot \mathbf{l})$$

diffusely
reflected light

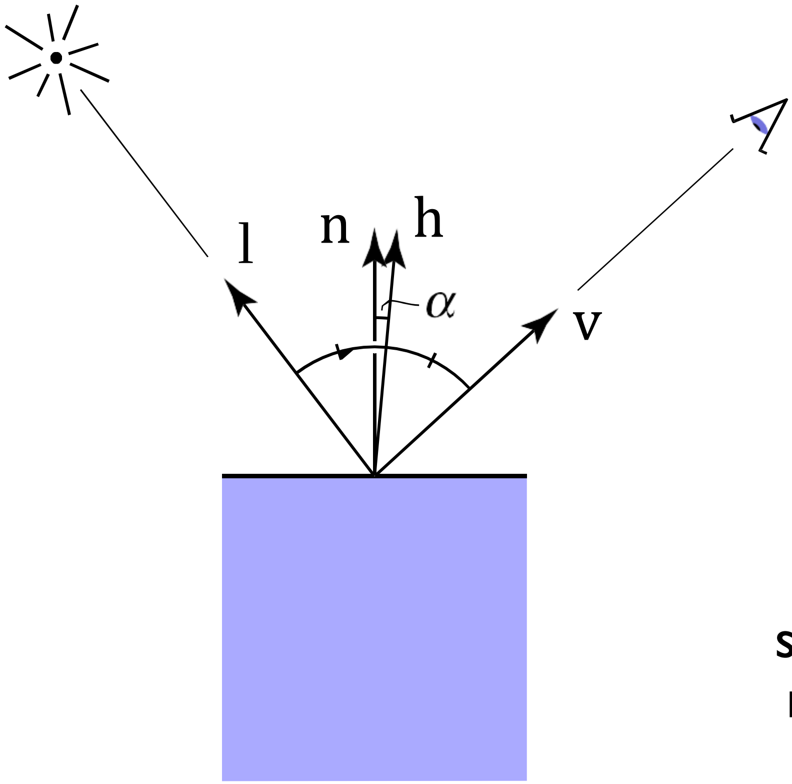
diffuse
coefficient
(color)

energy received
by the shading point

Specular Term (Blinn-Phong)

V close to mirror direction ⇔ half vector near normal

- Measure “near” by dot product of unit vectors



$$\mathbf{h} = \text{bisector}(\mathbf{v}, \mathbf{l})$$

(半程向量)

$$= \frac{\mathbf{v} + \mathbf{l}}{\|\mathbf{v} + \mathbf{l}\|}$$

$$L_s = k_s\,(I/r^2)\max(0, \cos\alpha)^p$$

$$= k_s\,(I/r^2)\max(0, \mathbf{n} \cdot \mathbf{h})^p$$

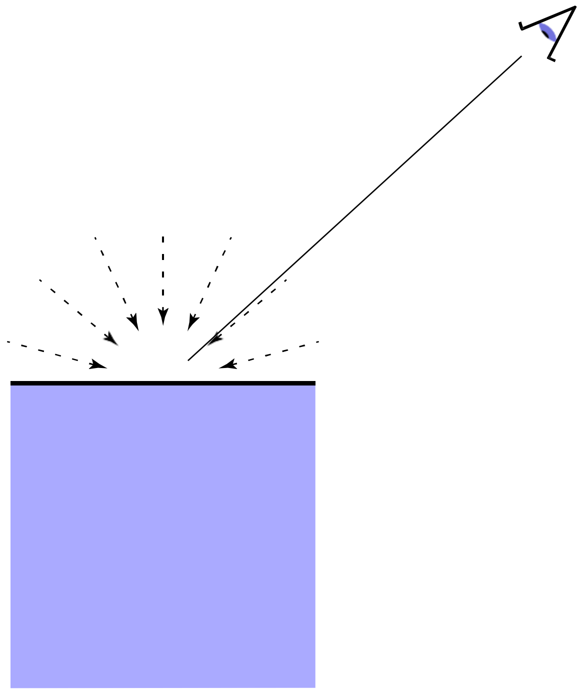
specularly reflected light

specular coefficient

Ambient Term

Shading that does not depend on anything

- Add constant color to account for disregarded illumination and fill in black shadows
- This is approximate / fake!

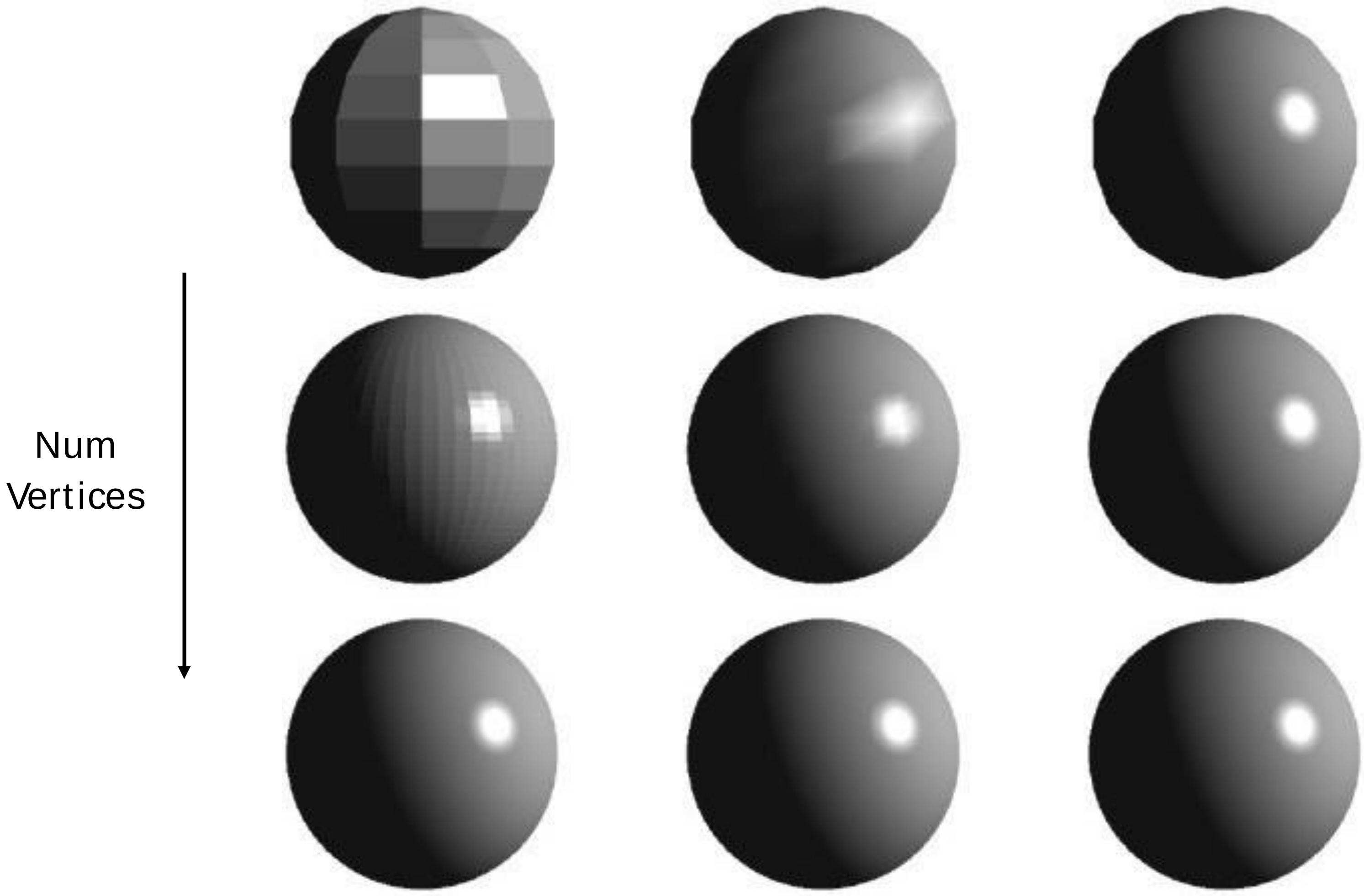


$$L_a = k_a \, I_a$$

ambient coefficient

reflected ambient light

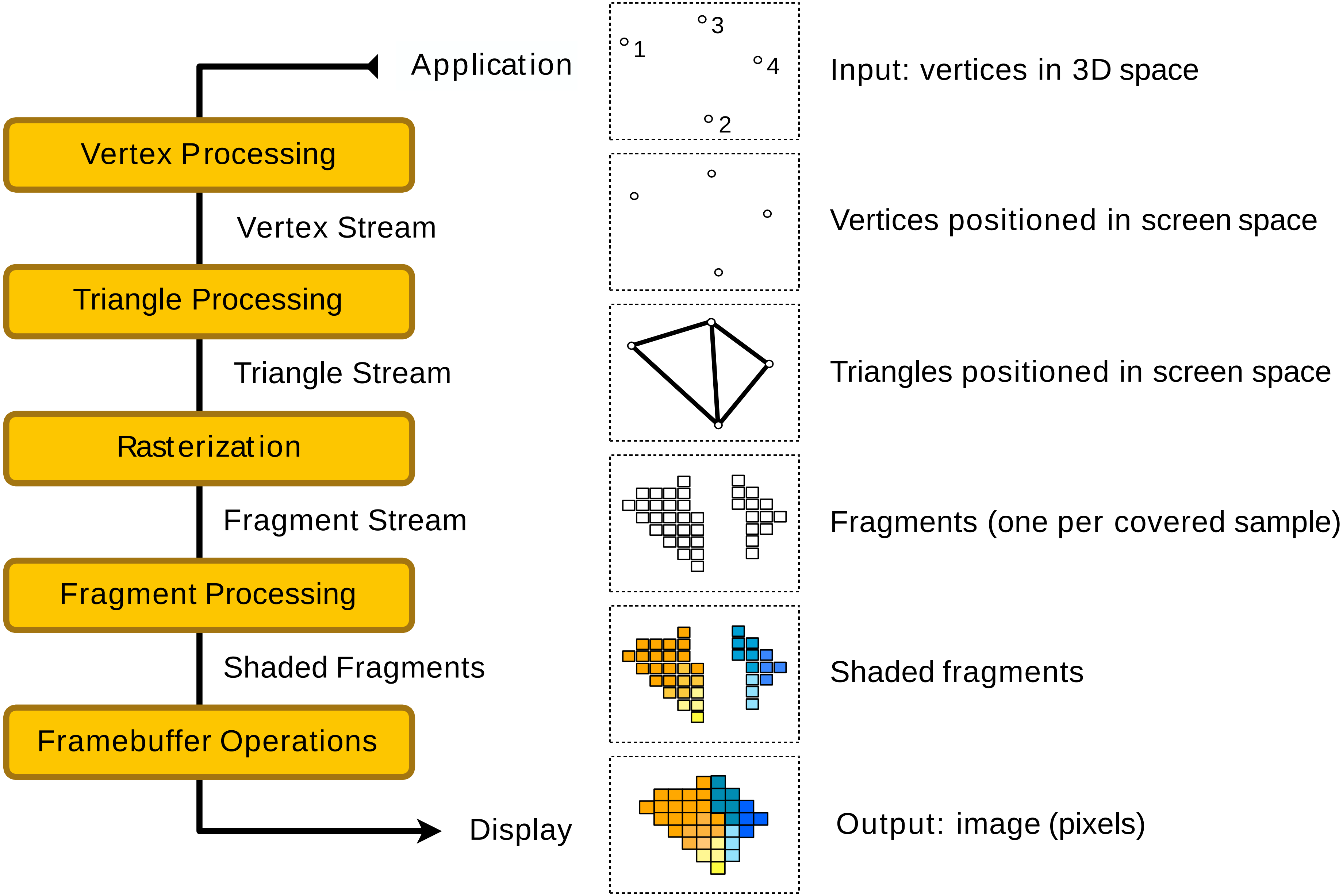
Shading Frequency: Face, Vertex or Pixel



Shading freq. :	Face	Vertex	Pixel
Shading type :	Flat	Gouraud	Phong

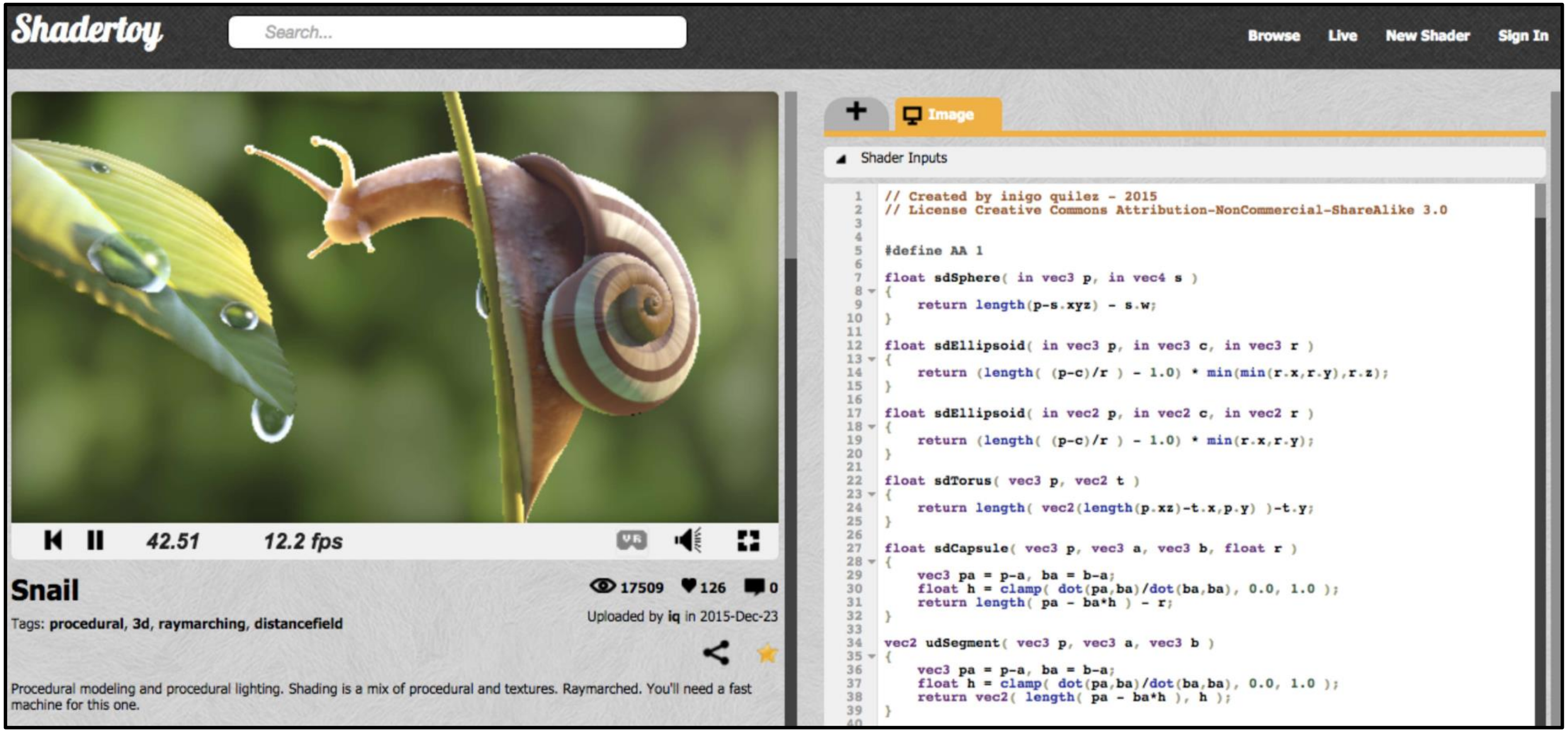
Image credit: Happyman, http://cg2010studio.com/

Graphics Pipeline



演示

Snail Shader Program



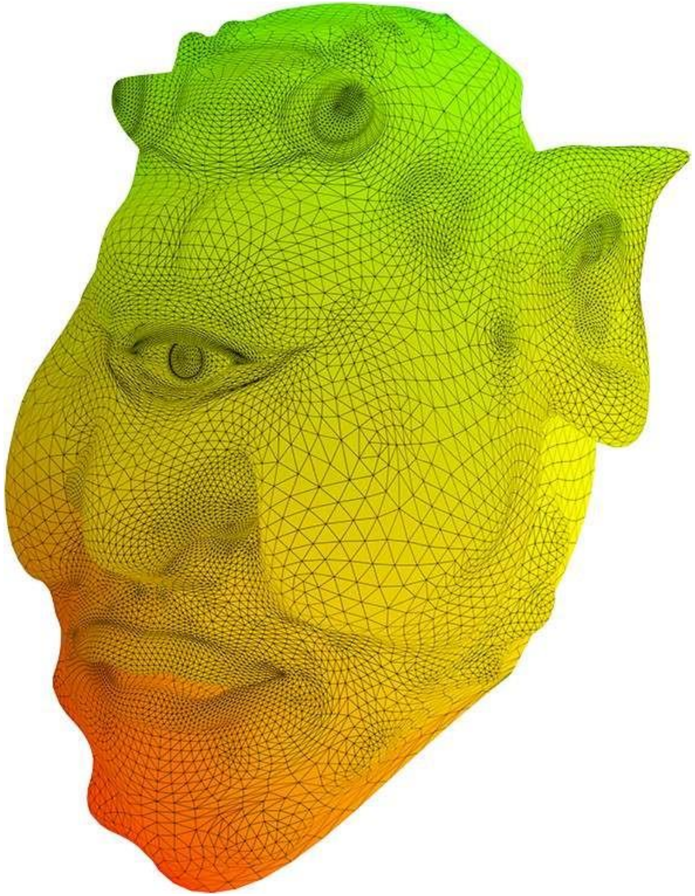
Inigo Quilez

Procedurally modeled, 800 line shader.
http://shadertoy.com/view/ld3Gz2

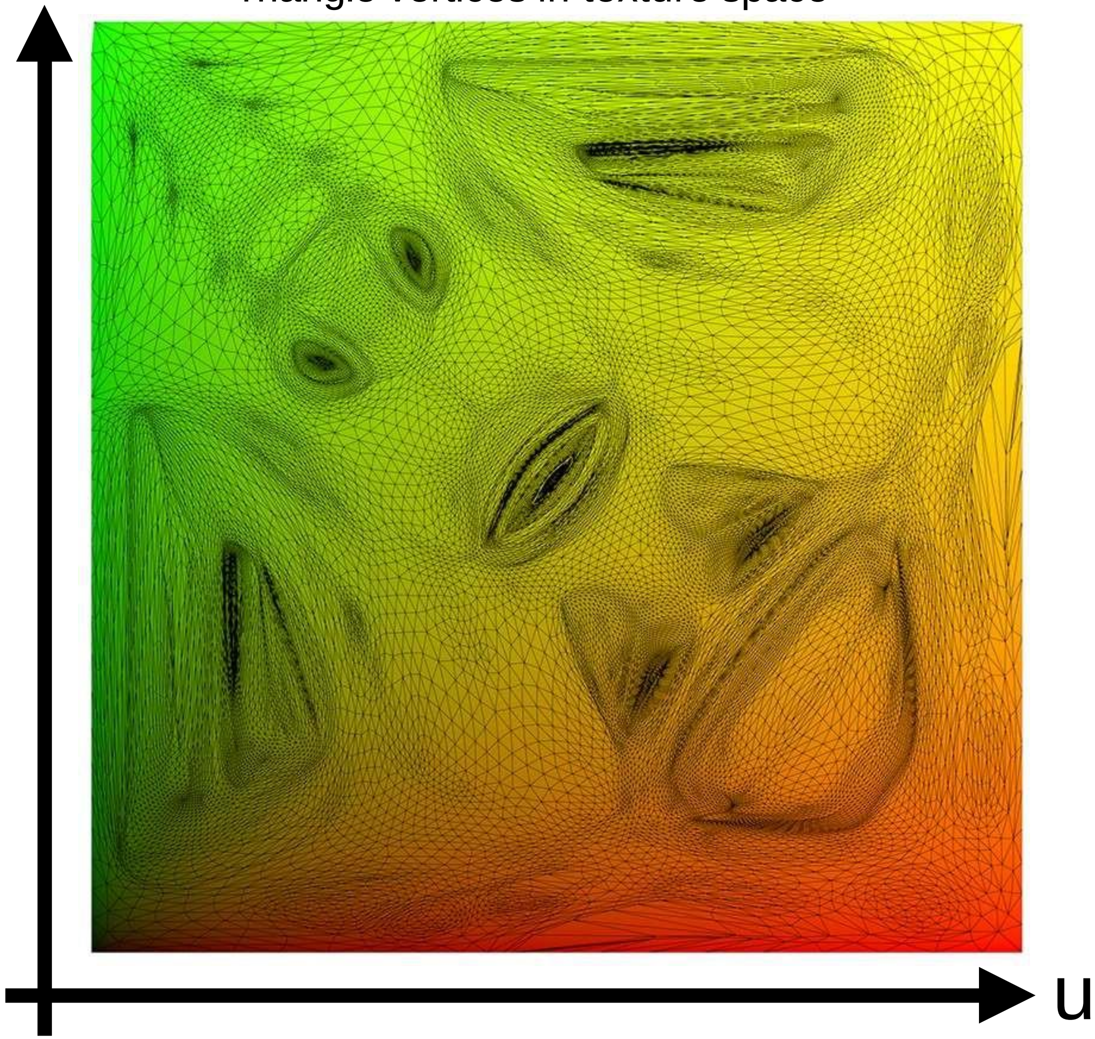
Visualization of Texture Coordinates

Each triangle vertex is assigned a texture coordinate (u,v)

Visualization of texture coordinates



Triangle vertices in texture space



单选题 1分

设置

Bisector vector (半程向量)出现在Blinn-Phong模型的?

A Ambient term

B Diffusion term

C Specular term

D Phong term

提交

单选题 1分

设置

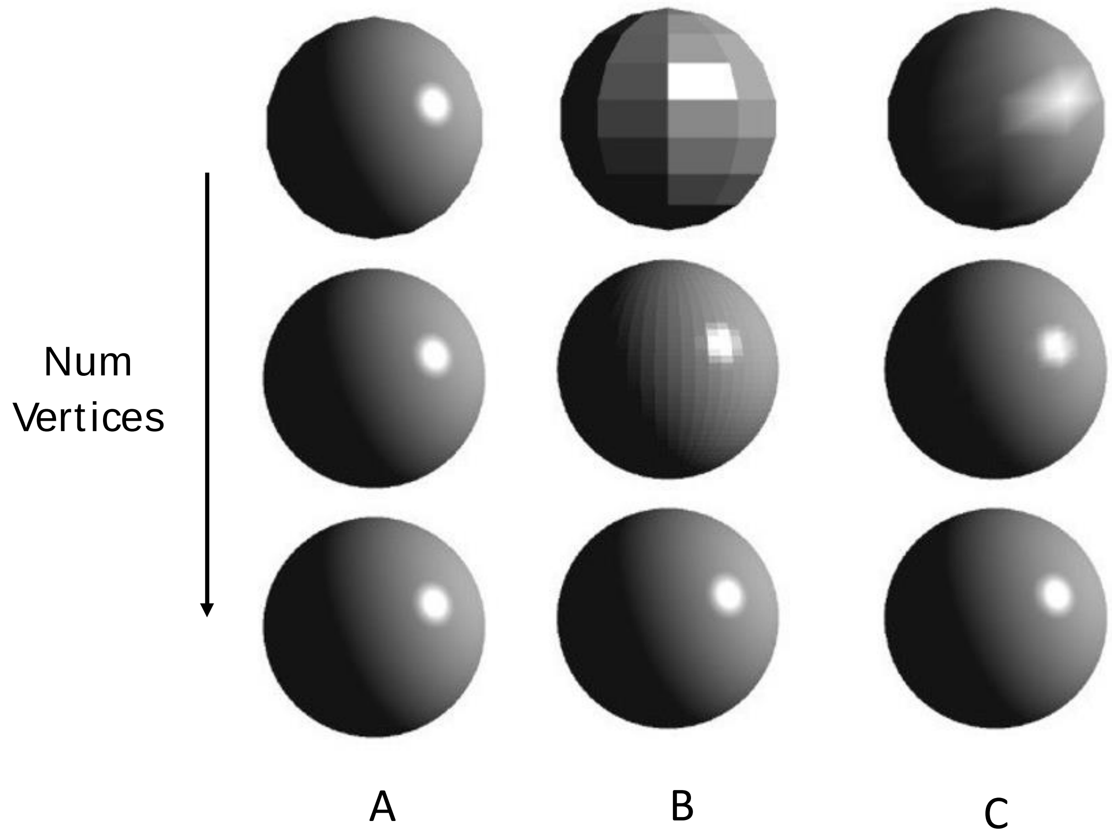
描述着色频率的示意图中，空格ABC处应该依次填?

A Face， Vertex, Pixel

B Vertex, Pixel, Face

C Face, Pixel, Vertex

D Pixel, Face, Vertex

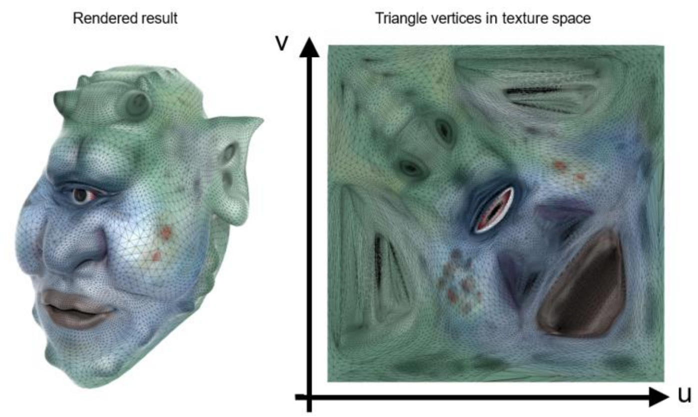


提交

Interpolation Across Triangles: Barycentric Coordinates

(重心坐标)

Interpolation Across Triangles



Why do we want to interpolate?

- Specify values at vertices
- Obtain smoothly varying values across triangles

What do we want to interpolate?

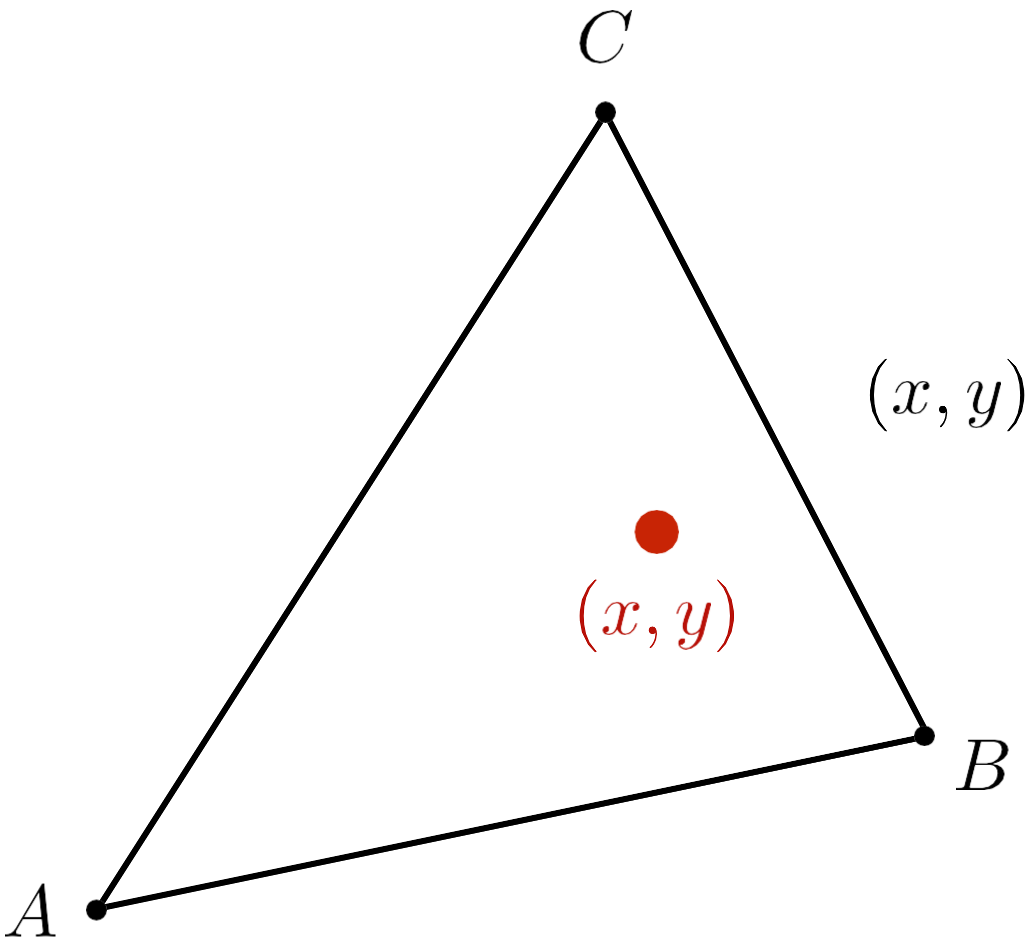
- Texture coordinates, colors, normal vectors, …

How do we interpolate?

- Barycentric coordinates

Barycentric Coordinates

A coordinate system for triangles (α, β, γ)



$$(x, y) = \alpha\, A + \beta\, B + \gamma\, C$$

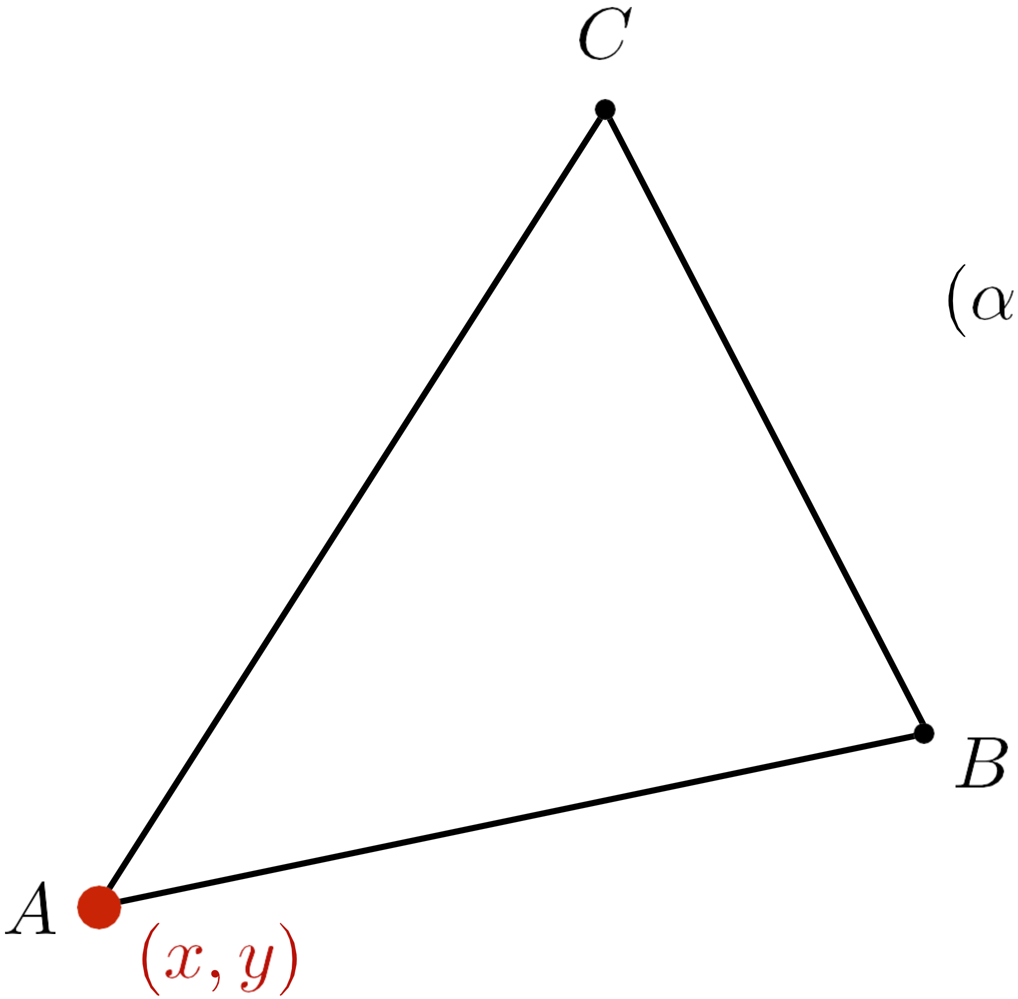
$$\alpha + \beta + \gamma = 1$$

Inside the triangle if all three coordinates are non-negative

https://www.inf.usi.ch/hormann/barycentric/index.html

Barycentric Coordinates

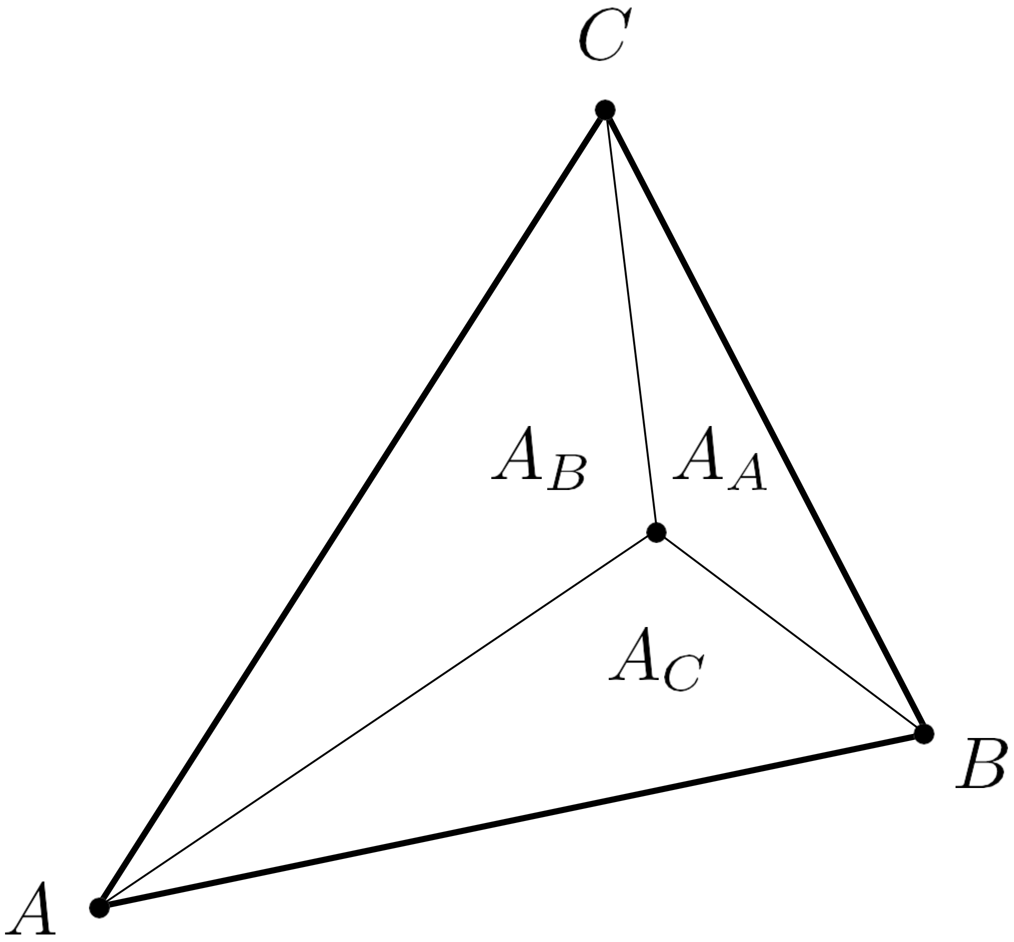
What's the barycentric coordinate of A?



$$\begin{aligned}(\alpha, \beta, \gamma) &= (1, 0, 0) \\ (x, y) &= \alpha\, A + \beta\, B + \gamma\, C \\ &= A\end{aligned}$$

Barycentric Coordinates

Geometric viewpoint — proportional areas



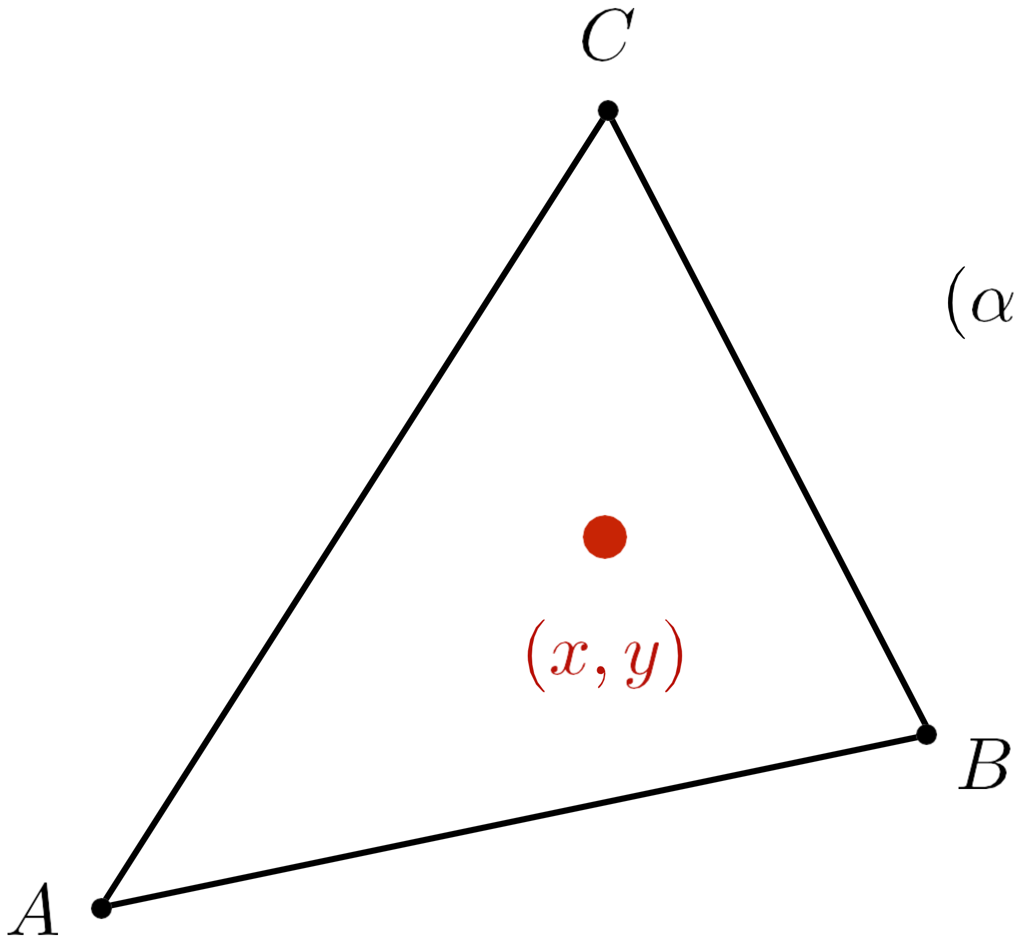
$$\alpha = \frac{A_A}{A_A + A_B + A_C}$$

$$\beta = \frac{A_B}{A_A + A_B + A_C}$$

$$\gamma = \frac{A_C}{A_A + A_B + A_C}$$

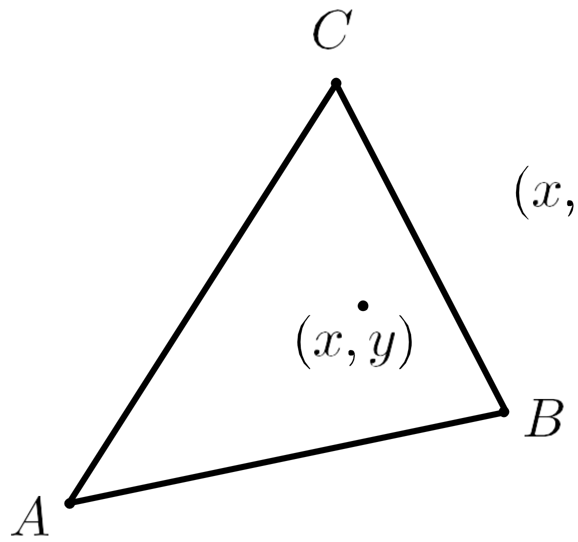
Barycentric Coordinates

What's the barycentric coordinate of the centroid?



$$(\alpha, \beta, \gamma) = \left(\tfrac{1}{3}, \tfrac{1}{3}, \tfrac{1}{3}\right)$$
$$(x, y) = \tfrac{1}{3}\,A + \tfrac{1}{3}\,B + \tfrac{1}{3}\,C$$

Barycentric Coordinates: Formulas



$$(x,y) = \alpha\, A + \beta\, B + \gamma\, C$$
$$\alpha + \beta + \gamma = 1$$

$$\alpha = \frac{-(x - x_B)(y_C - y_B) + (y - y_B)(x_C - x_B)}{-(x_A - x_B)(y_C - y_B) + (y_A - y_B)(x_C - x_B)}$$

$$\beta = \frac{-(x - x_C)(y_A - y_C) + (y - y_C)(x_A - x_C)}{-(x_B - x_C)(y_A - y_C) + (y_B - y_C)(x_A - x_C)}$$

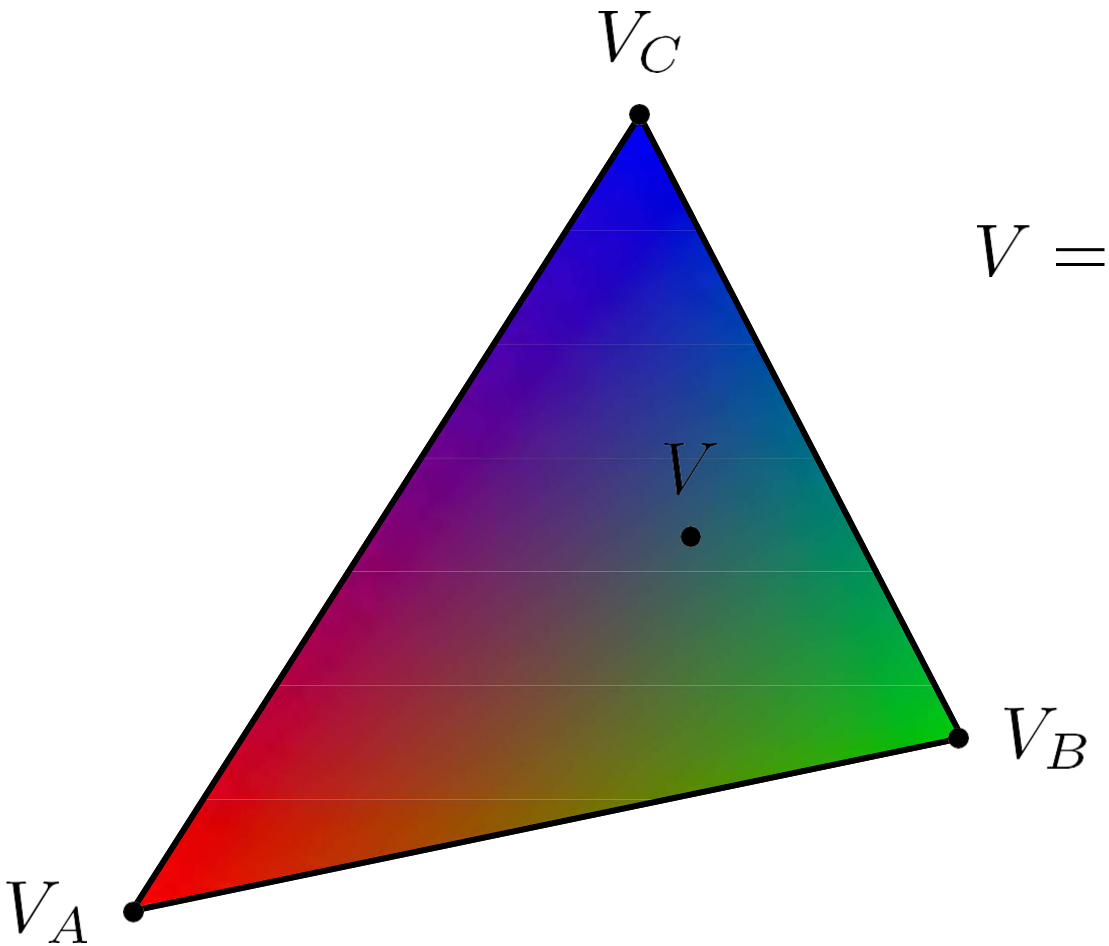
$$\gamma = 1 - \alpha - \beta$$

$A(x_1,y_1), B(x_2,y_2), C(x_3,y_3)$，平移后有
$A(0,0), B(x_2-x_1, y_2-y_1), C(x_3-x_1, y_3-y_1)$，则有:

$$S_{\Delta ABC} = \frac{1}{2}|(x_2-x_1)(y_3-y_1)-(y_2-y_1)(x_3-x_1)| = \frac{1}{2}|x_1y_2+x_2y_3+x_3y_1 - x_2y_1 - x_3y_2 - x_1y_3|$$

Using Barycentric Coordinates

Linearly interpolate values at vertices



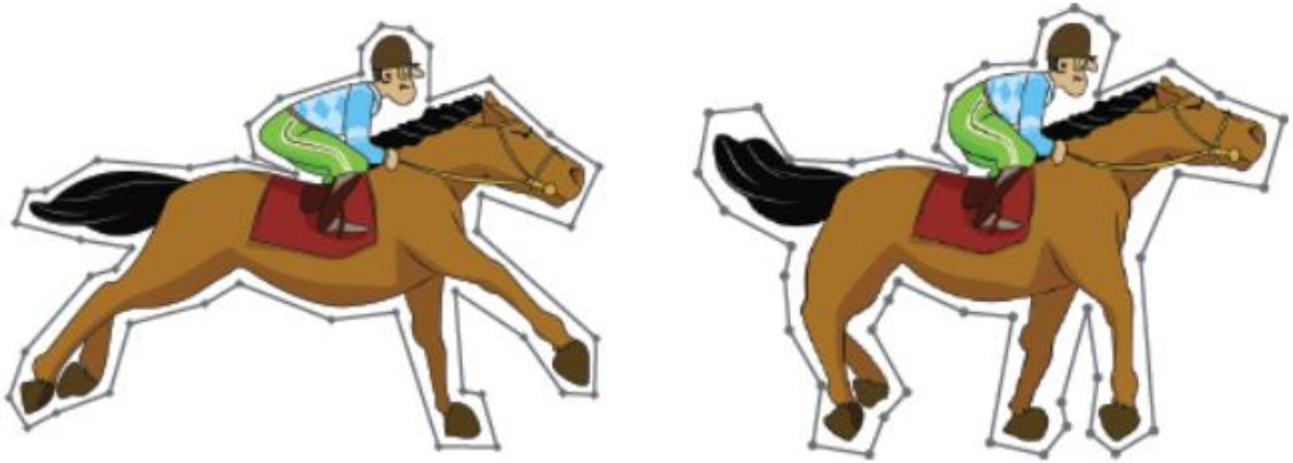
$$V = \alpha V_A + \beta V_B + \gamma V_C$$

V_A, V_B, V_C can be positions, texture coordinates, color, normal, depth, material attributes…

However, barycentric coordinates are not invariant under projection!

Generalized Barycentric Coordinates

These coordinates have been later generalized to support simple polygons in 2D and polyhedra in 3D.



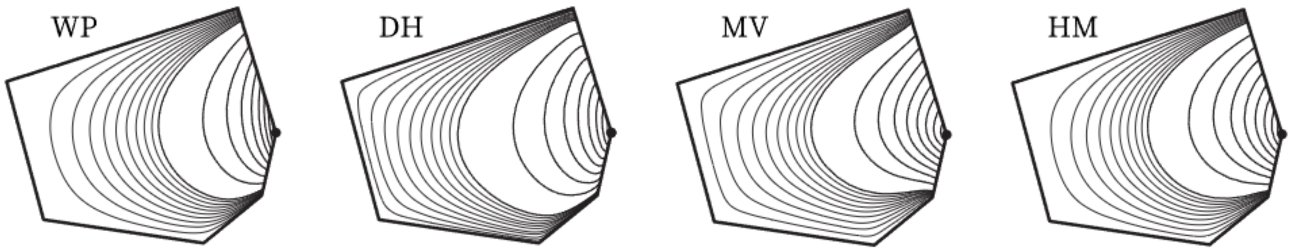


Figure 101.1 Wachspress (WP), discrete harmonic (DH), mean value (MV), and harmonic (HM) coordinate functions for a convex polygon plotted with respect to the marked vertex.

https://doc.cgal.org/latest/Barycentric_coordinates_2/index.html#Chapter_2D_Generalized_Barycentric_Coordinates

Applying Textures

Simple Texture Mapping: Diffuse Color

Usually a pixel's center

for each rasterized screen sample (x,y):

(u,v) = evaluate texture coordinate at (x,y)

texcolor = texture.sample(u,v);

set sample's color to texcolor;

Using barycentric coordinates!

Usually the diffuse albedo Kd
(recall the Blinn-Phong reflectance model)

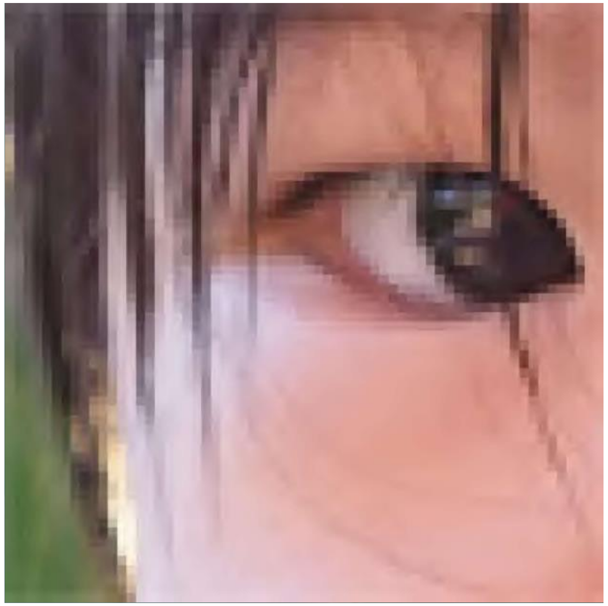
Texture Magnification

(What if the texture is too small?)

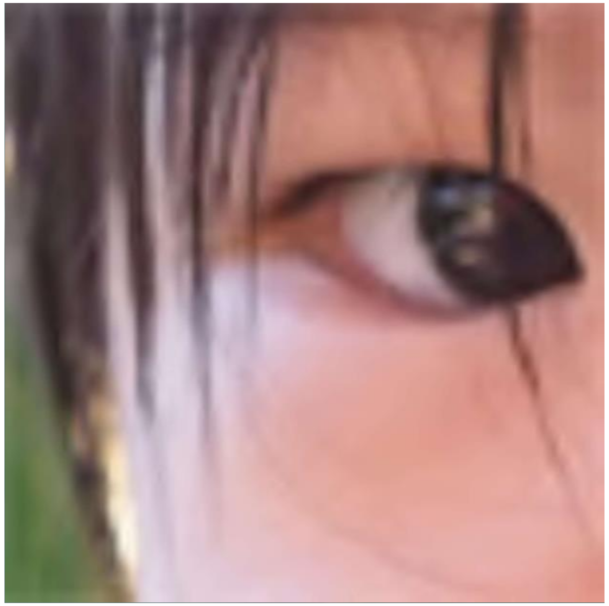
Texture Magnification - Easy Case

Generally don't want this — insufficient texture resolution

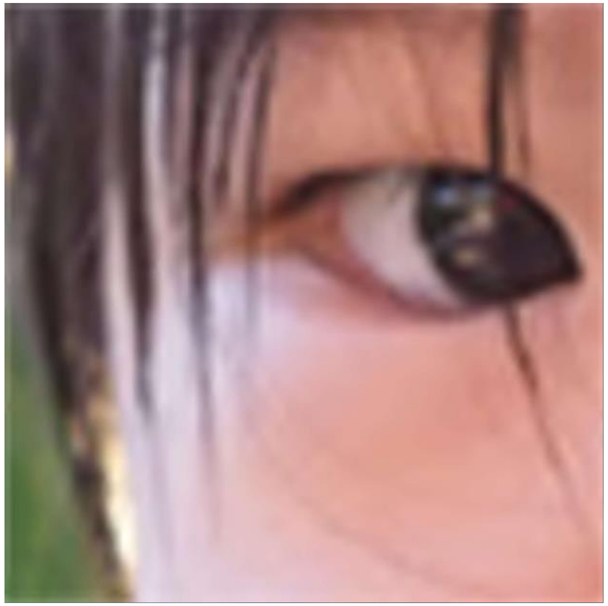
A pixel on a texture — a texel (纹理元素、纹素)



Nearest

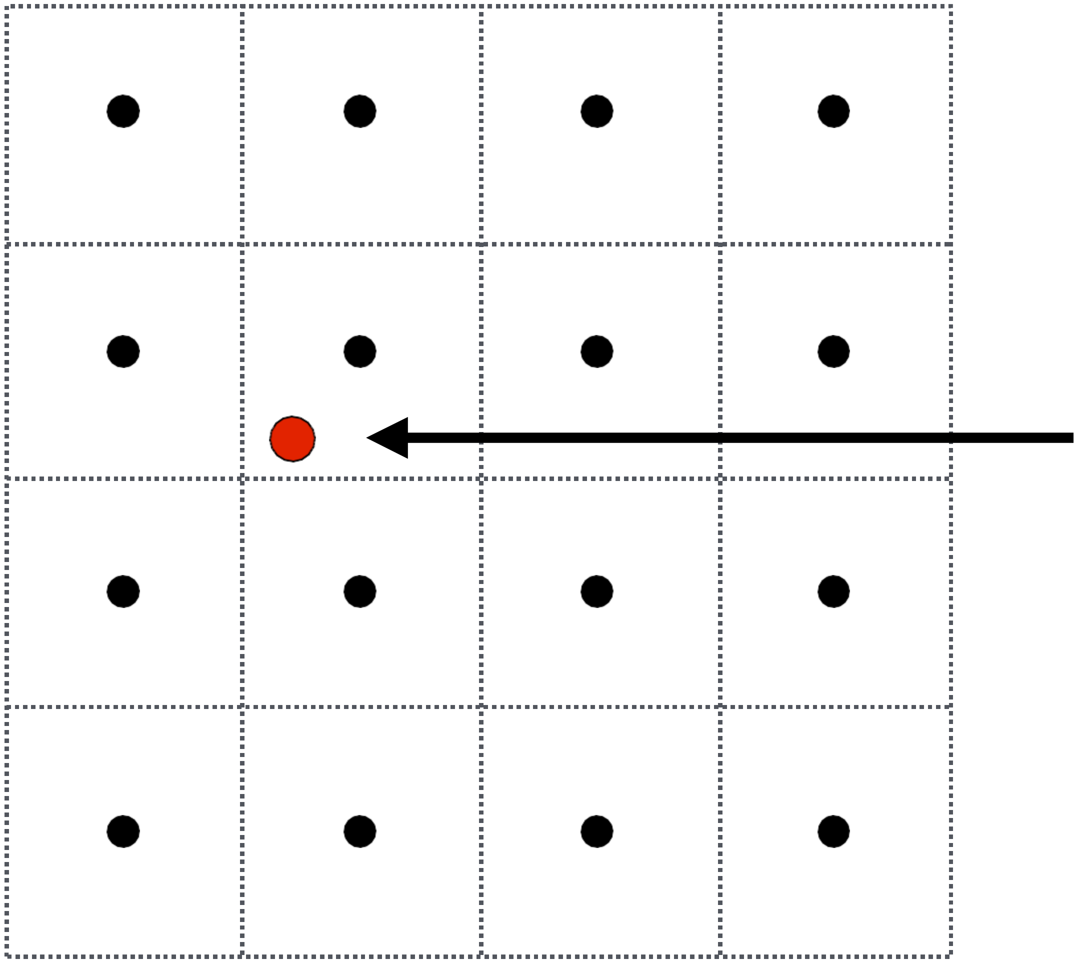


Bilinear



Bicubic

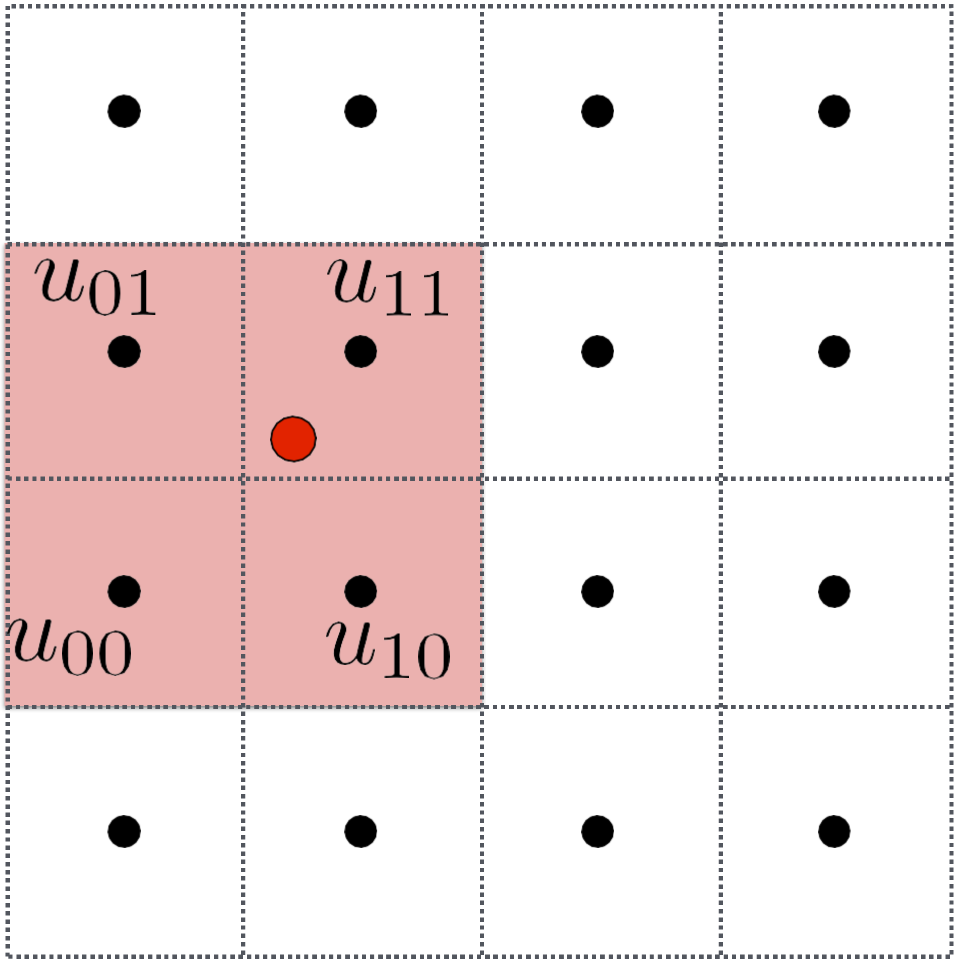
Bilinear Interpolation



Want to sample texture value f(x,y) at red point

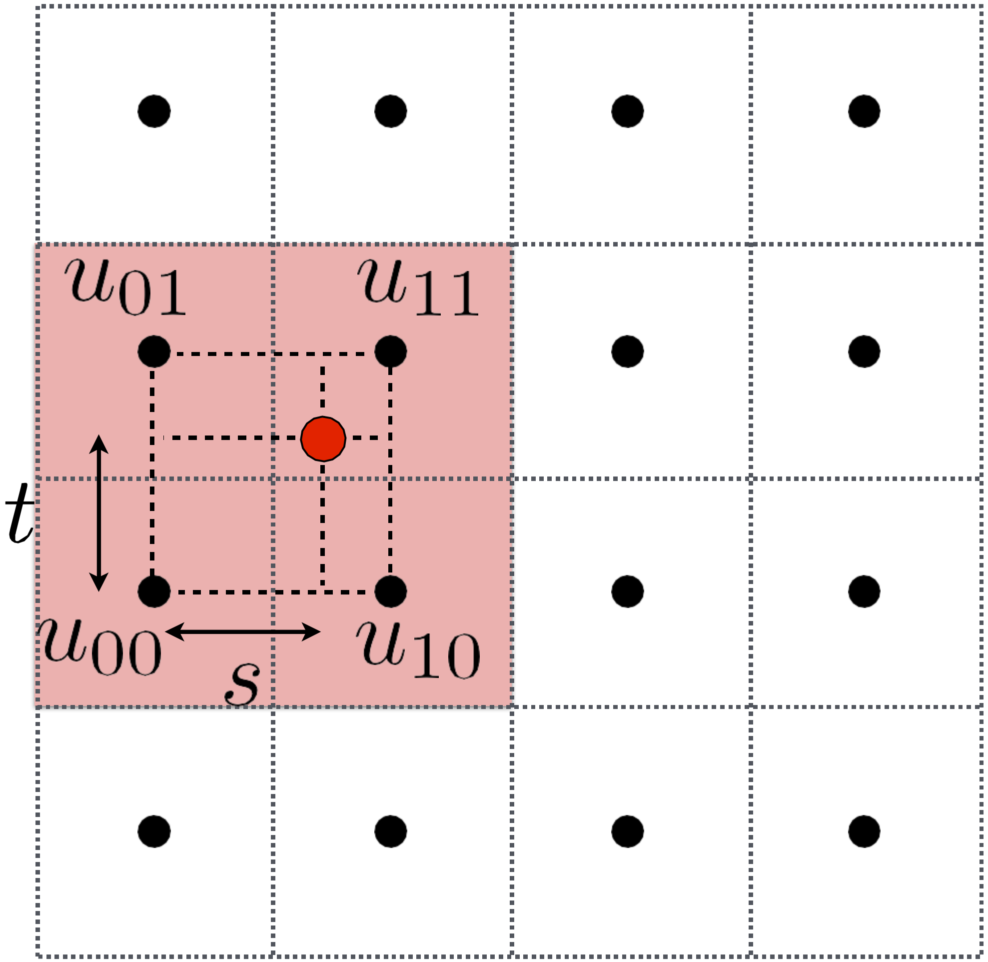
Black points indicate texture sample locations

Bilinear Interpolation



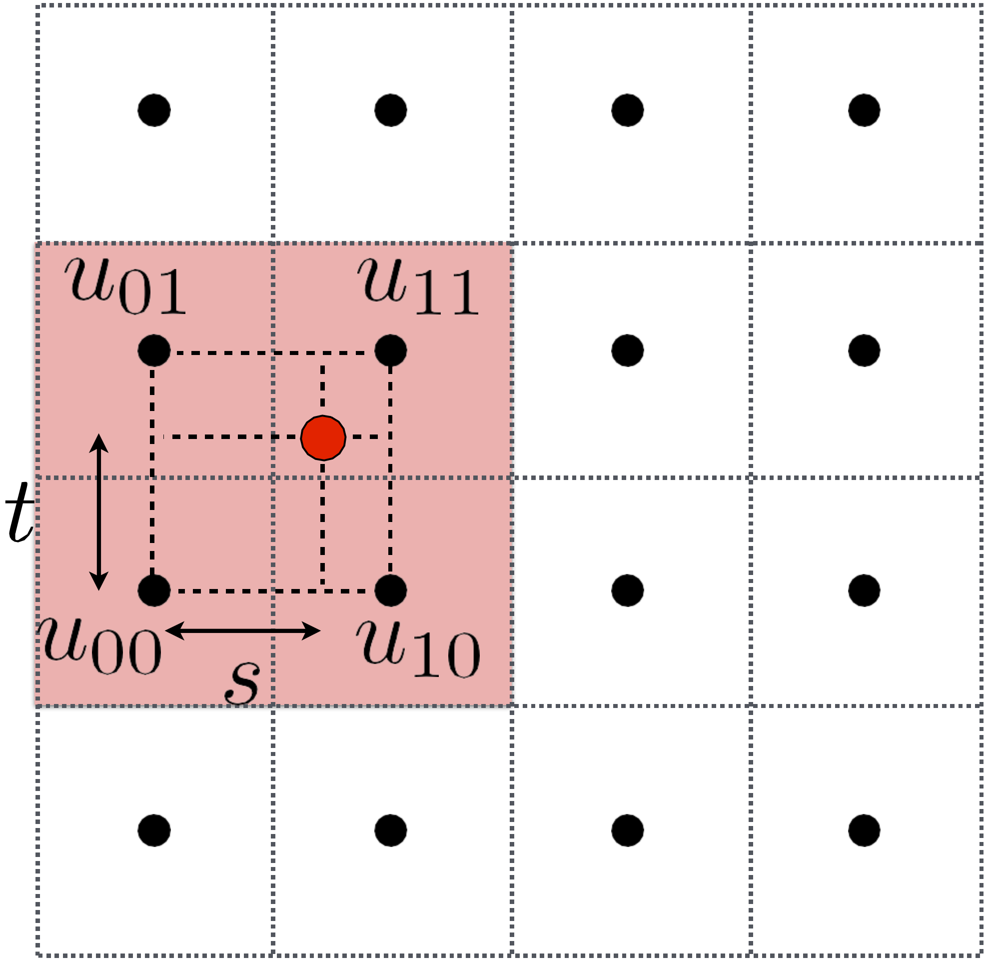
Take 4 nearest sample locations, with texture values as labeled.

Bilinear Interpolation



And fractional offsets, (s,t) as shown

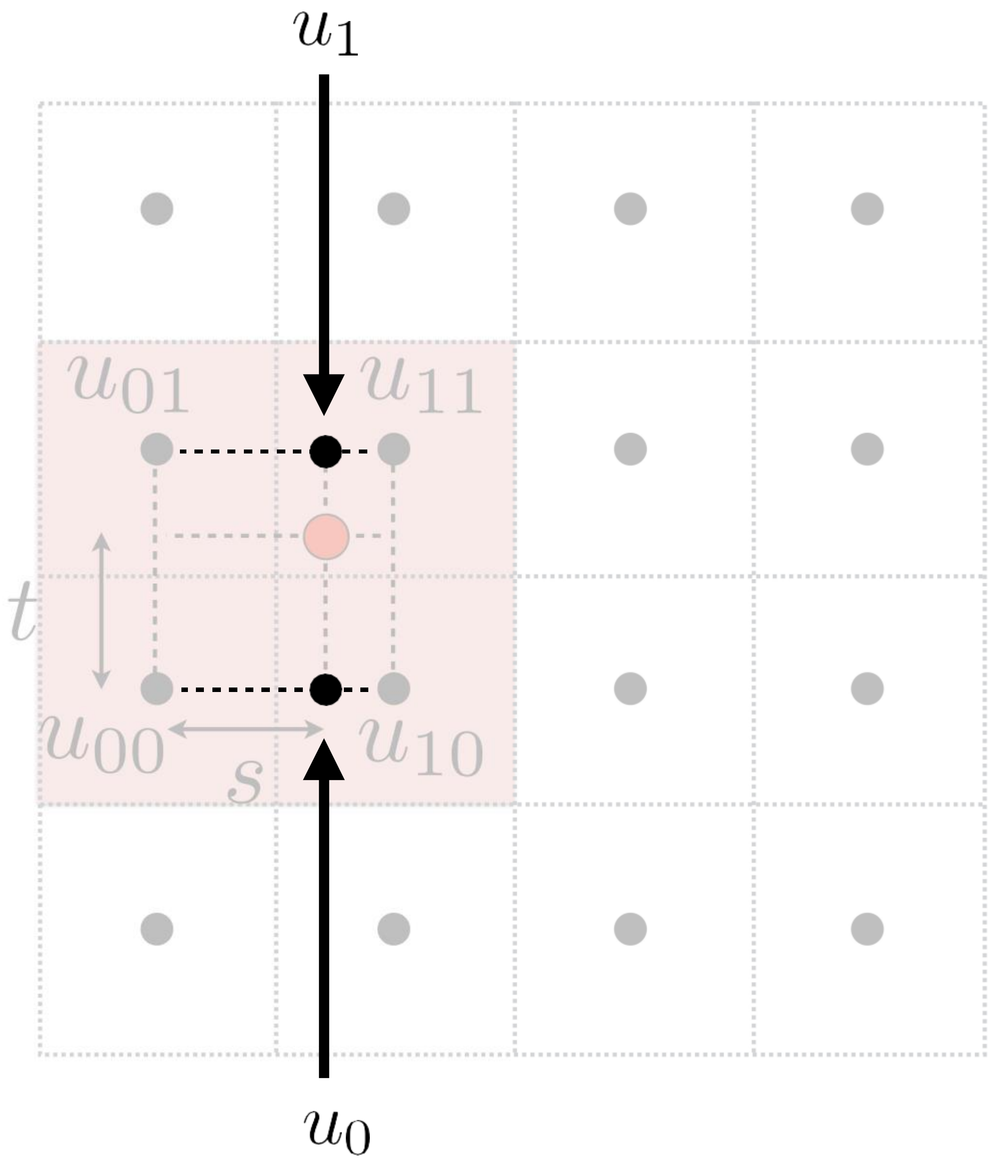
Bilinear Interpolation



Linear interpolation (1D)

$$\mathrm{lerp}(x, v_0, v_1) = v_0 + x(v_1 - v_0)$$

Bilinear Interpolation



Linear interpolation (1D)

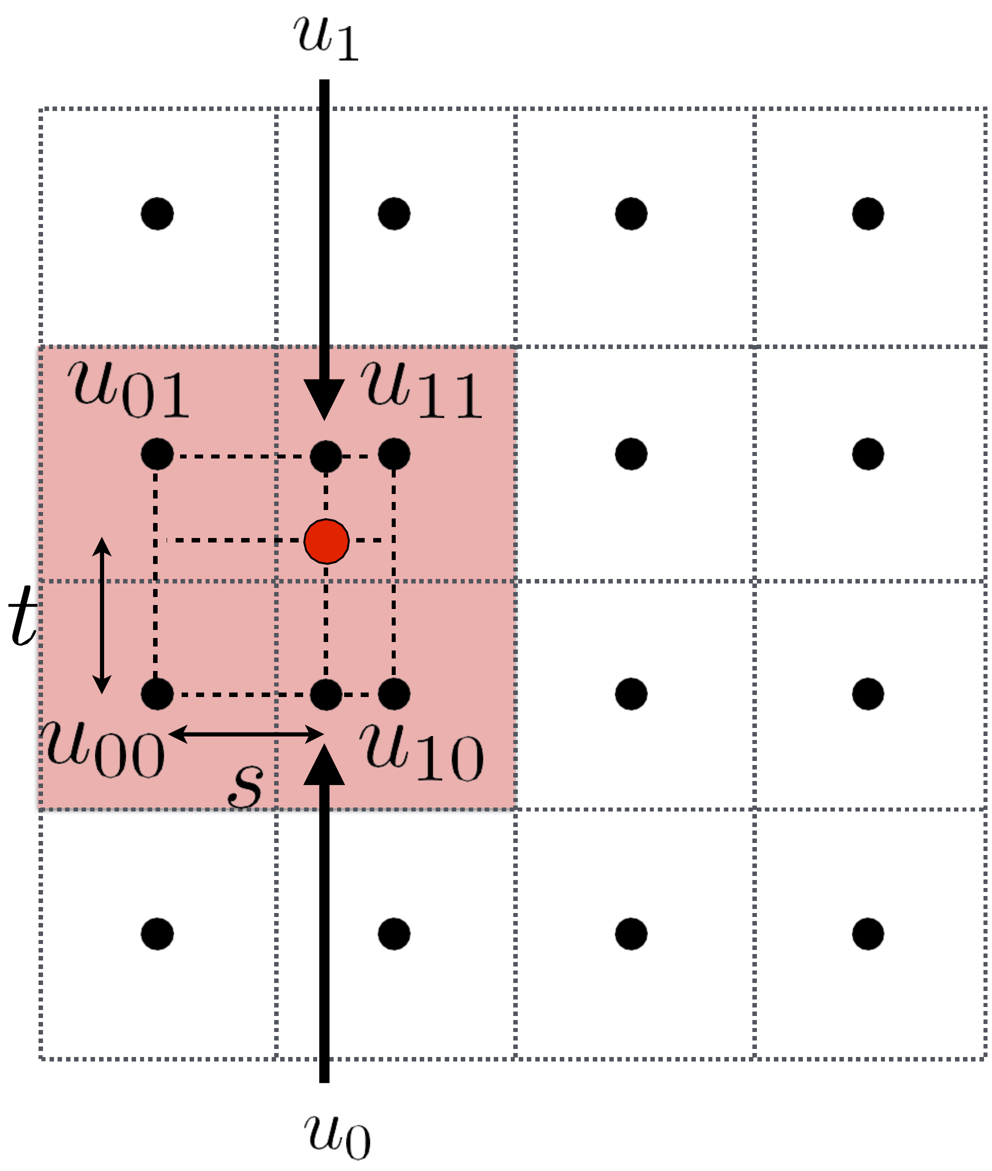
$$\mathrm{lerp}(x, v_0, v_1) = v_0 + x(v_1 - v_0)$$

Two helper lerps (horizontal)

$$u_0 = \mathrm{lerp}(s, u_{00}, u_{10})$$

$$u_1 = \mathrm{lerp}(s, u_{01}, u_{11})$$

Bilinear Interpolation



Linear interpolation (1D)

$$\mathrm{lerp}(x, v_0, v_1) = v_0 + x(v_1 - v_0)$$

Two helper lerps

$$u_0 = \mathrm{lerp}(s, u_{00}, u_{10})$$

$$u_1 = \mathrm{lerp}(s, u_{01}, u_{11})$$

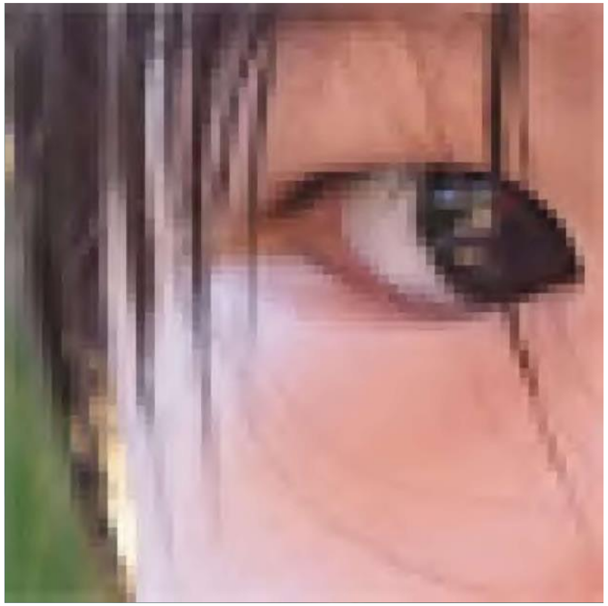
Final vertical lerp, to get result:

$$f(x, y) = \mathrm{lerp}(t, u_0, u_1)$$

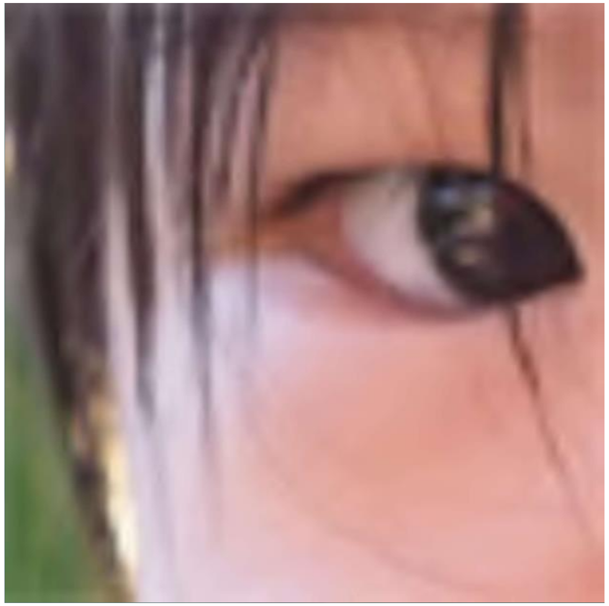
演示权重的推导?

Texture Magnification - Easy Case

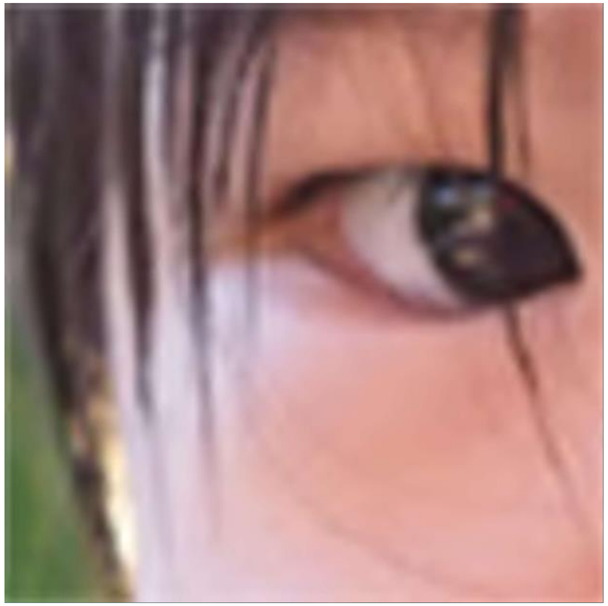
Bilinear interpolation usually gives pretty good results at reasonable costs



Nearest



Bilinear

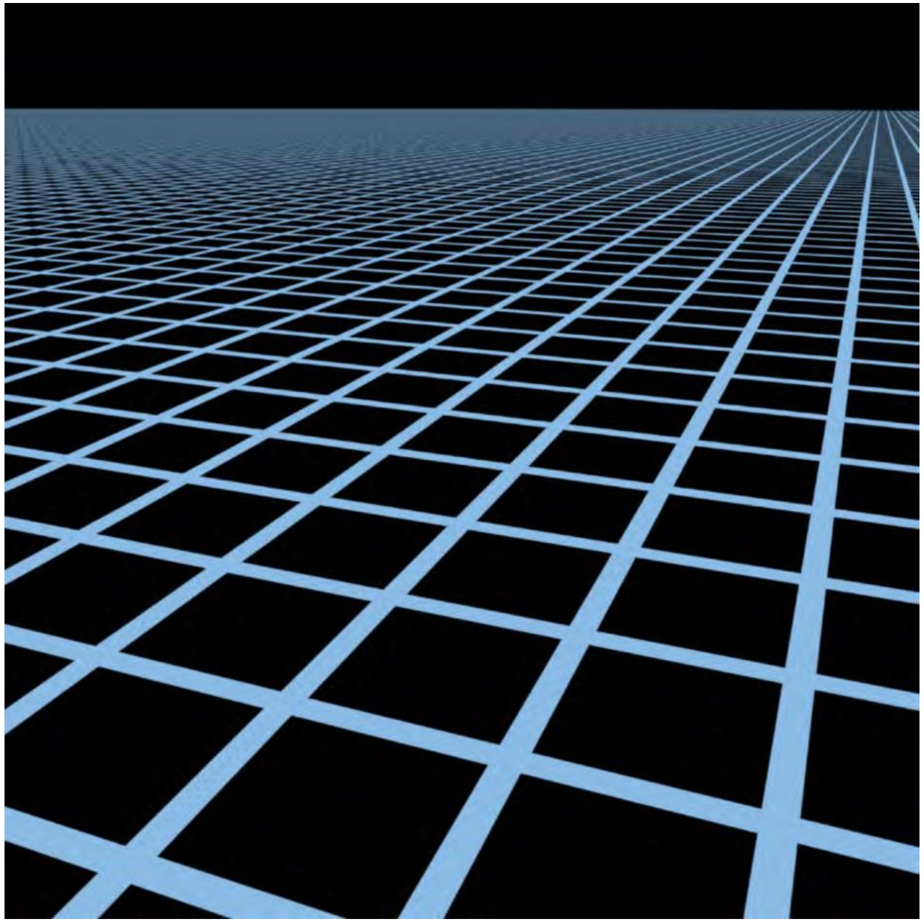


Bicubic

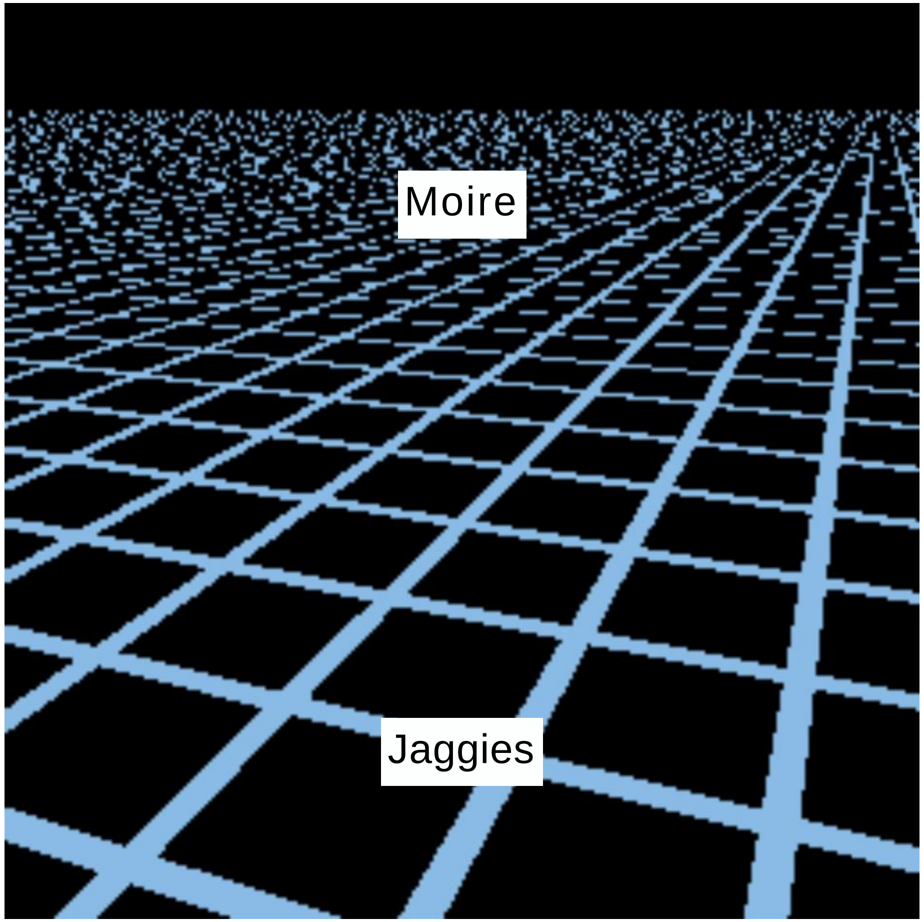
Texture Magnification (hard case)

(What if the texture is too large?)

Point Sampling Textures — Problem

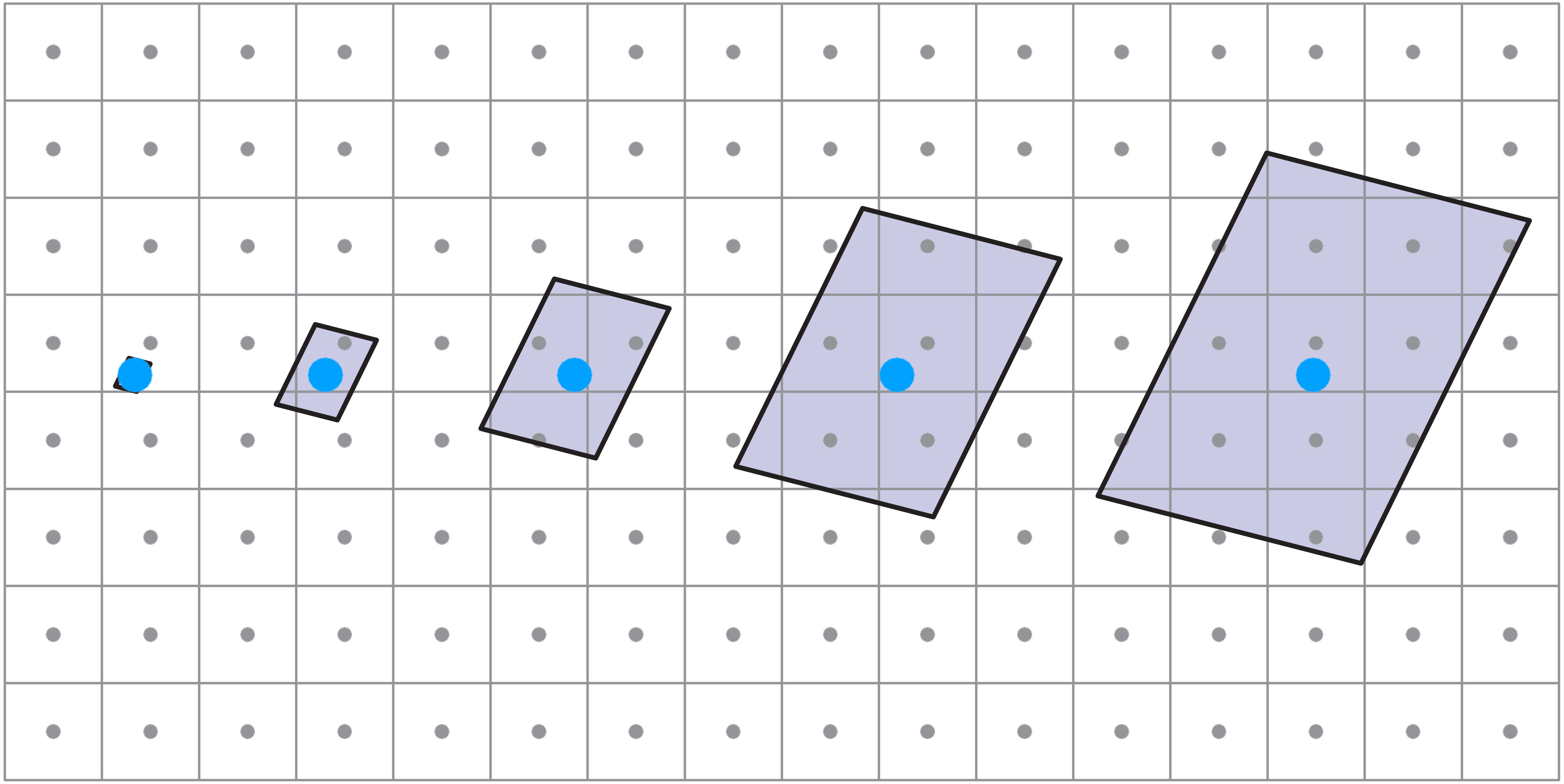


Reference



Point sampled

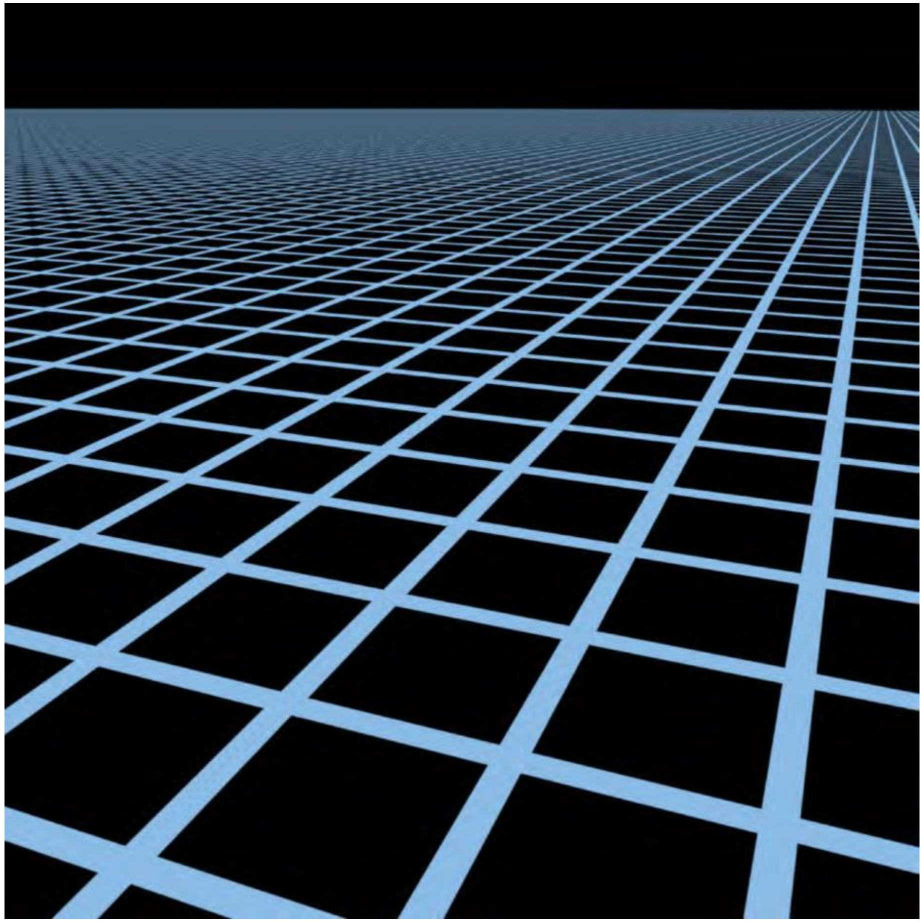
Screen Pixel "Footprint" in Texture



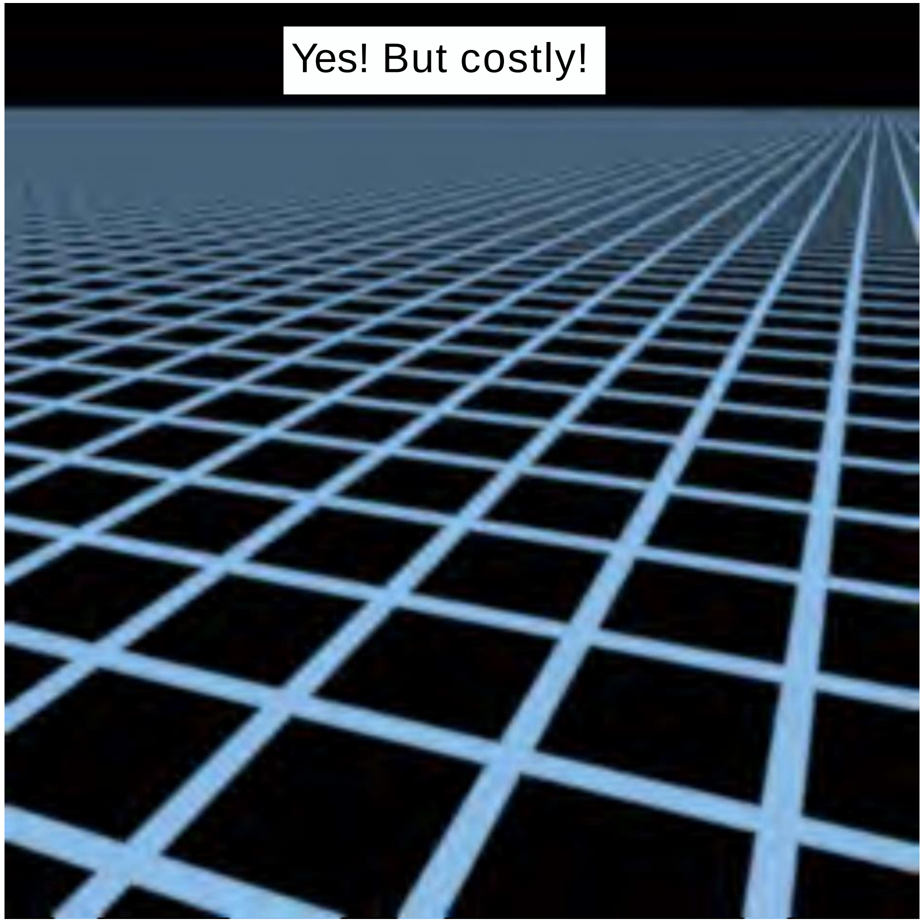
Upsampling
(Magnification)

Downsampling
(Minification)

Will Supersampling Do Antialiasing?



Multisample Anti-Aliasing
(MSAA)
多重采样抗锯齿



512x supersampling

Antialiasing — Supersampling?

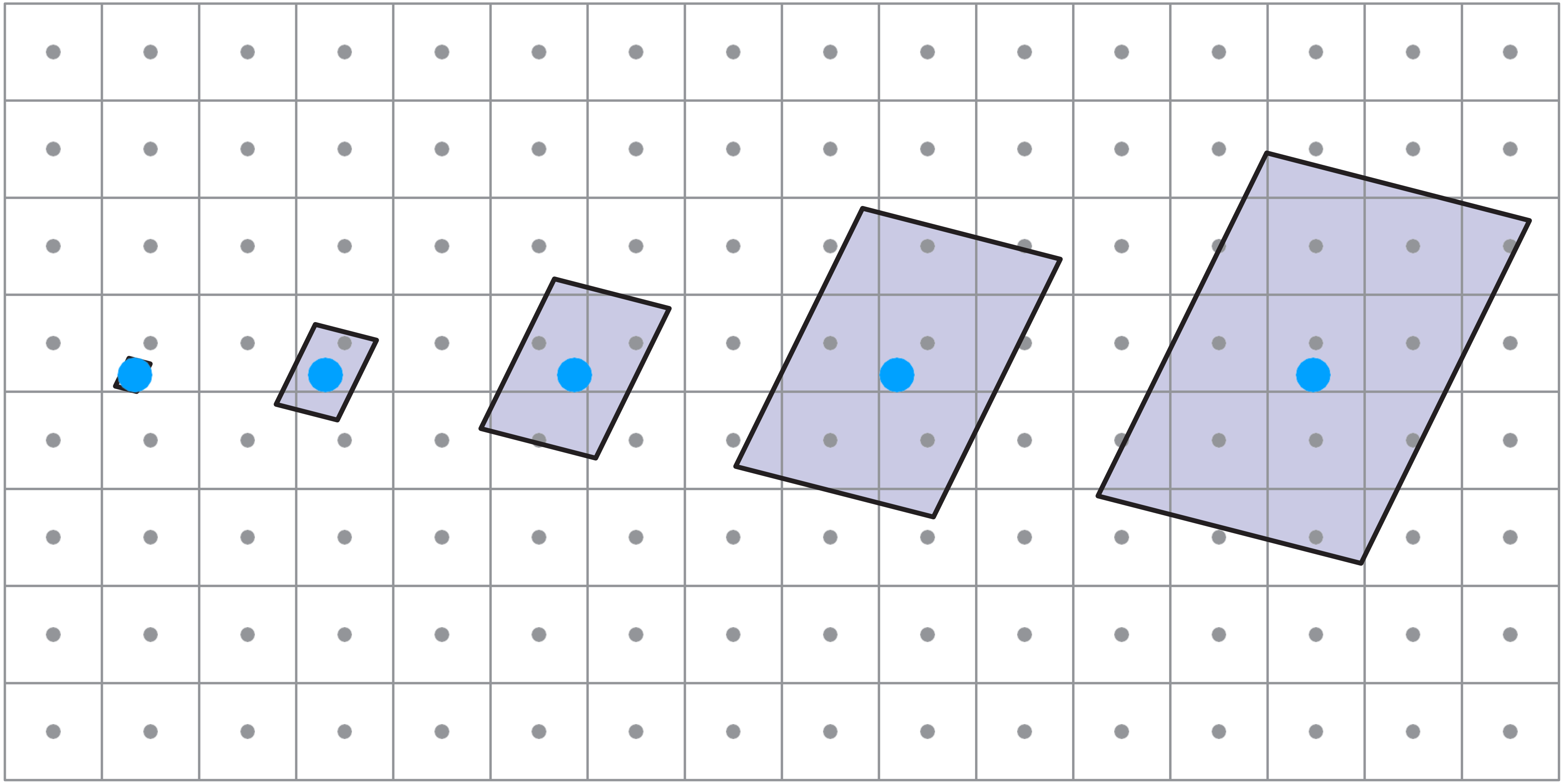
Will supersampling work?

- Yes, high quality, but costly
- When highly minified, many texels in pixel footprint
- Signal frequency too large in a pixel
- Need even higher sampling frequency

Let’s understand this problem in another way

- What if we don’t sample?
- Just need to get the average value within a range!

Point Query vs. (Avg.) Range Query



Different Pixels -> Different-Sized Footprints

Mipmap

Allowing (fast, approx., square) range queries

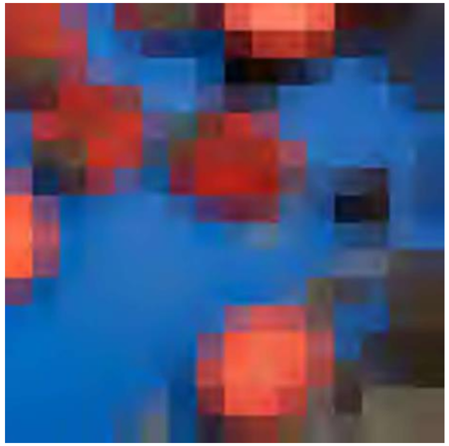
Mipmap (L. Williams 83)

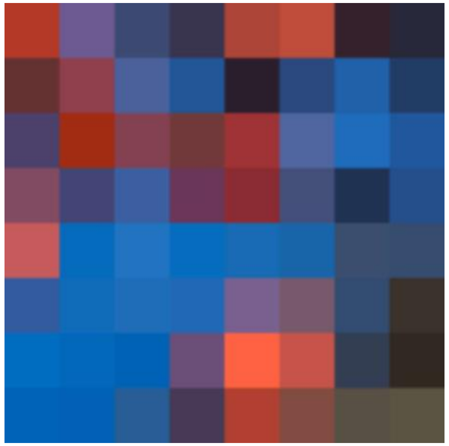
“Mip” comes from the Latin “multum in parvo", meaning a multitude in a small space

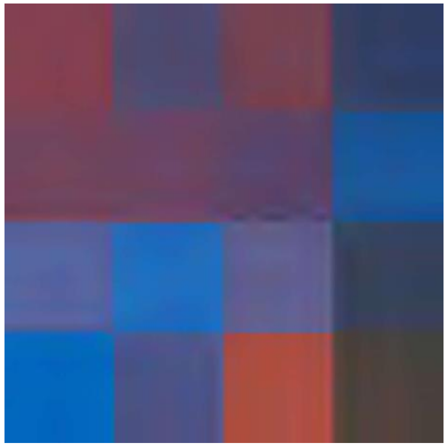

Level 0 = 128x128

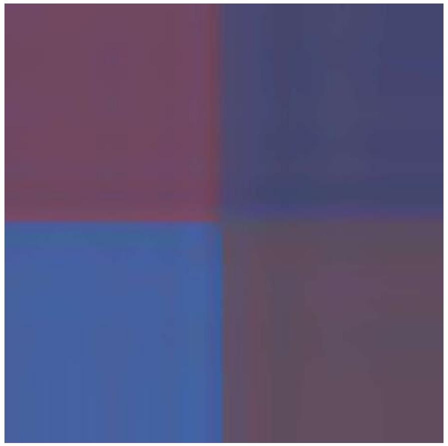

Level 1 = 64x64


Level 2 = 32x32


Level 3 = 16x16

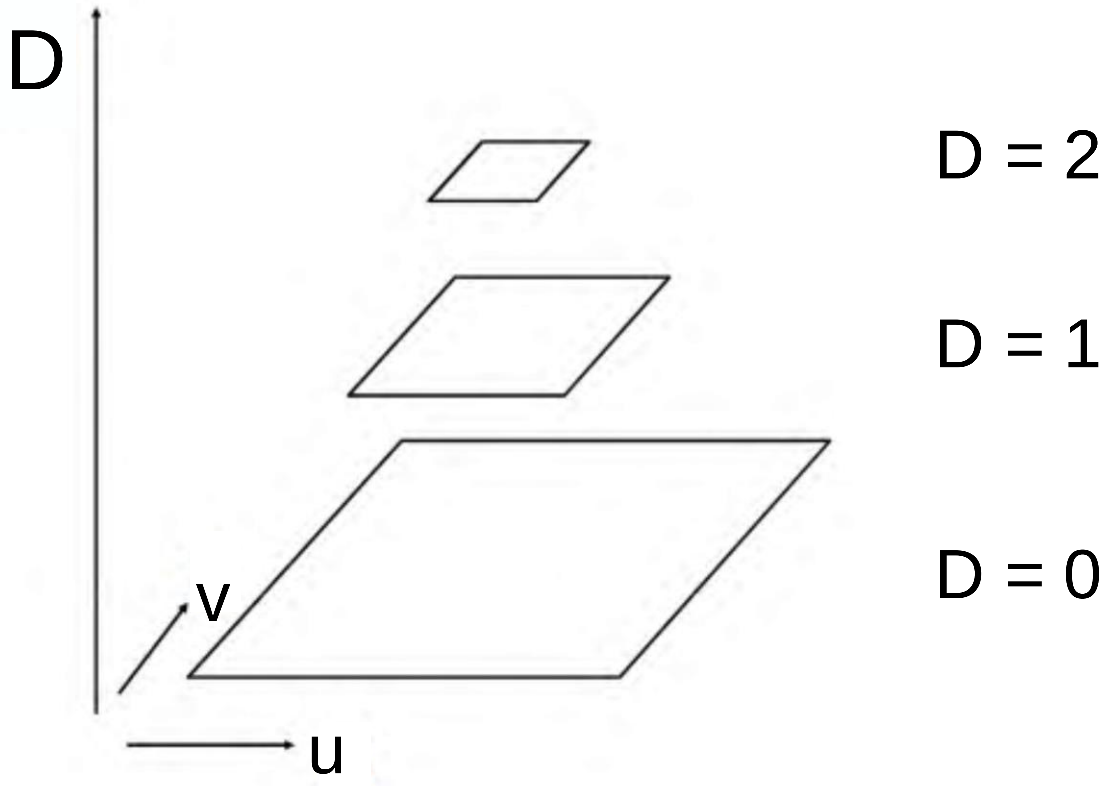

Level 4 = 8x8


Level 5 = 4x4


Level 6 = 2x2


Level 7 = 1x1

Mipmap (L. Williams 83)

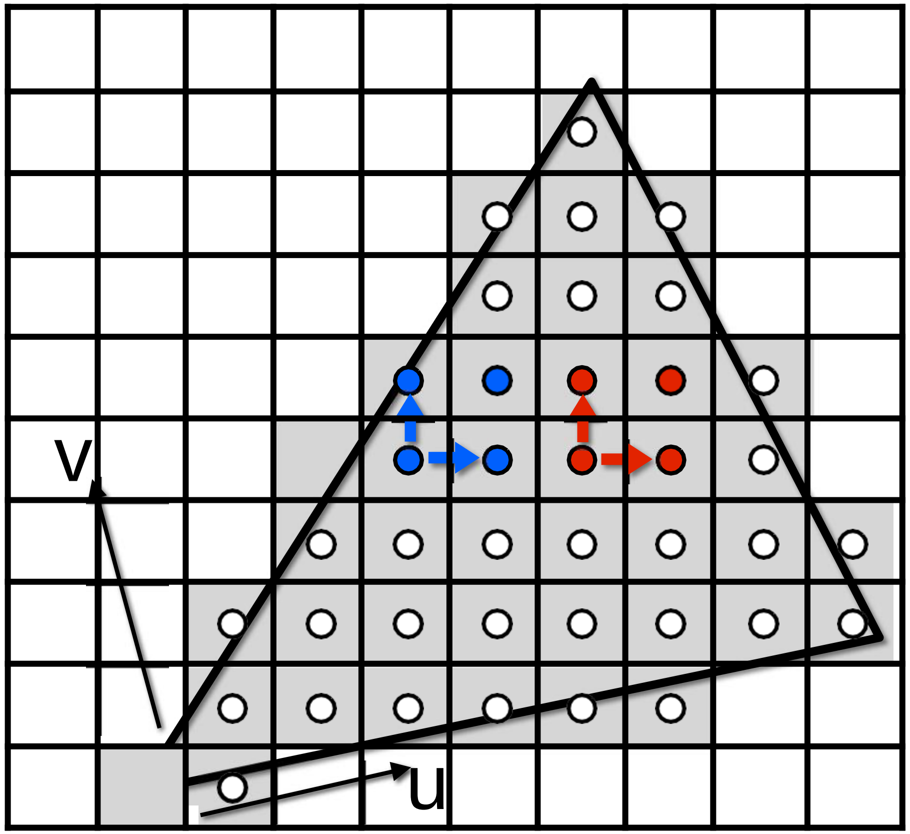


"Mip hierarchy"

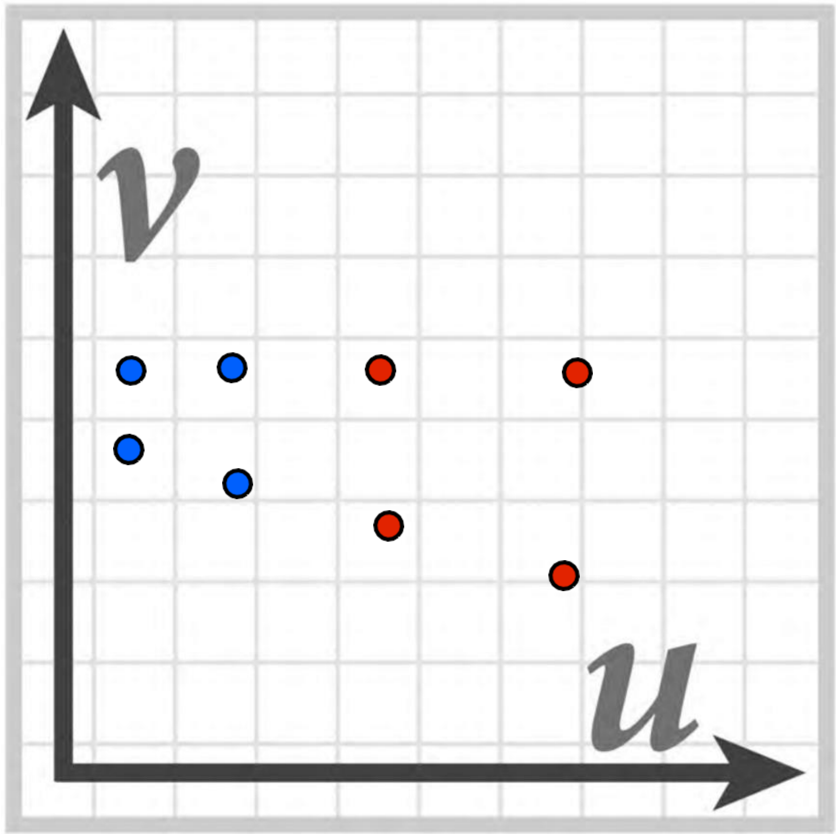
level = D

What is the storage overhead of a mipmap?

Computing Mipmap Level D



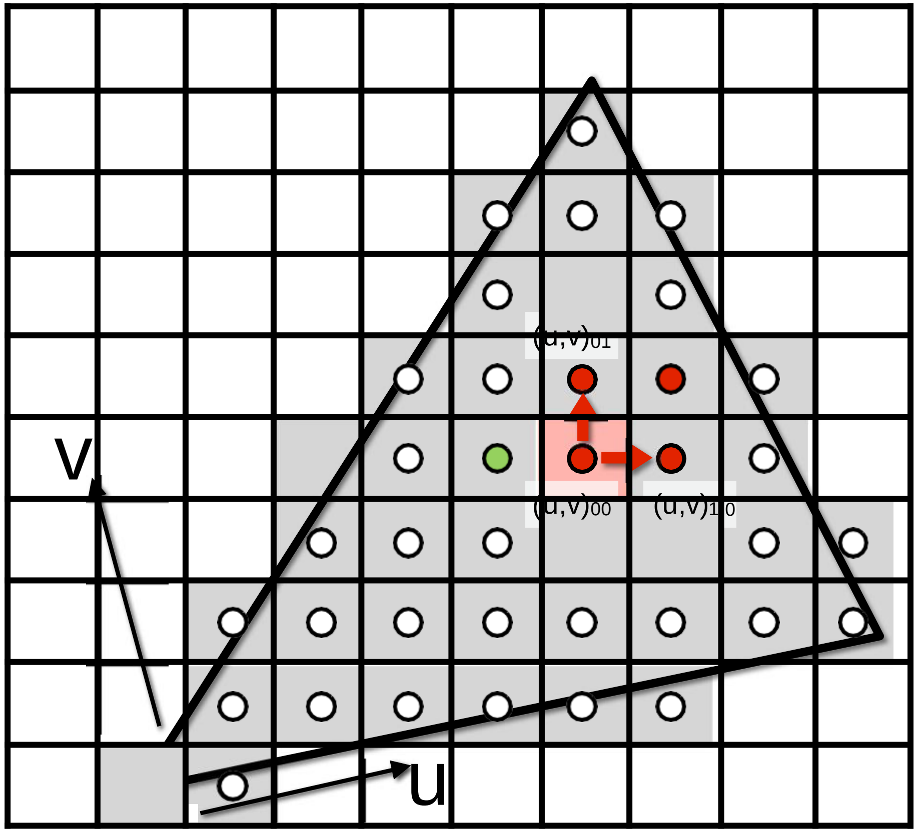
Screen space (x,y)

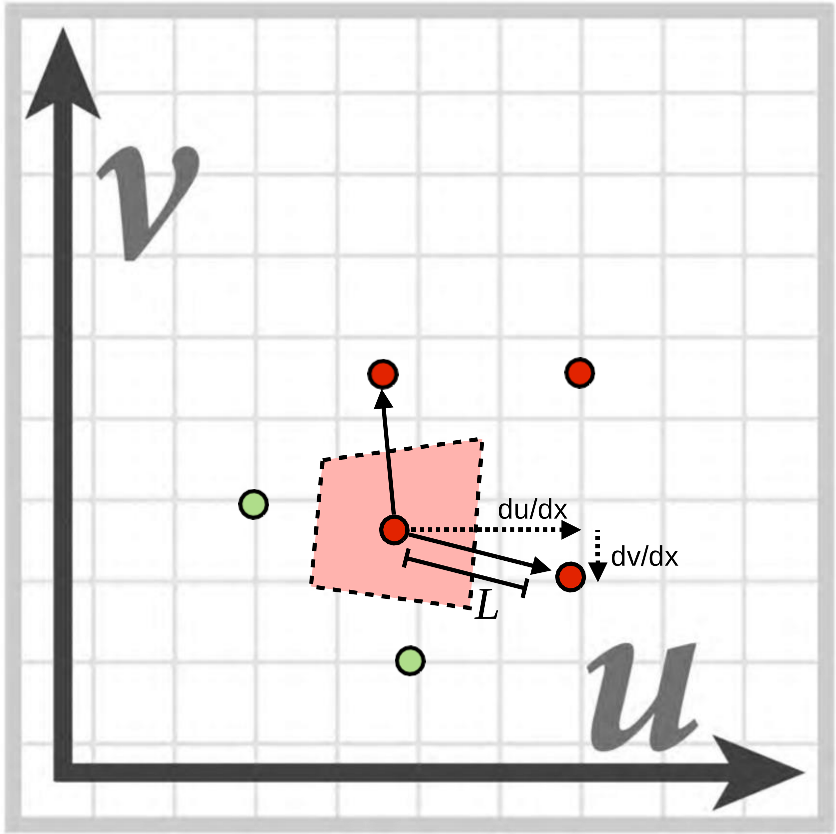


Texture space (u,v)

Estimate texture footprint using texture coordinates of neighboring screen samples

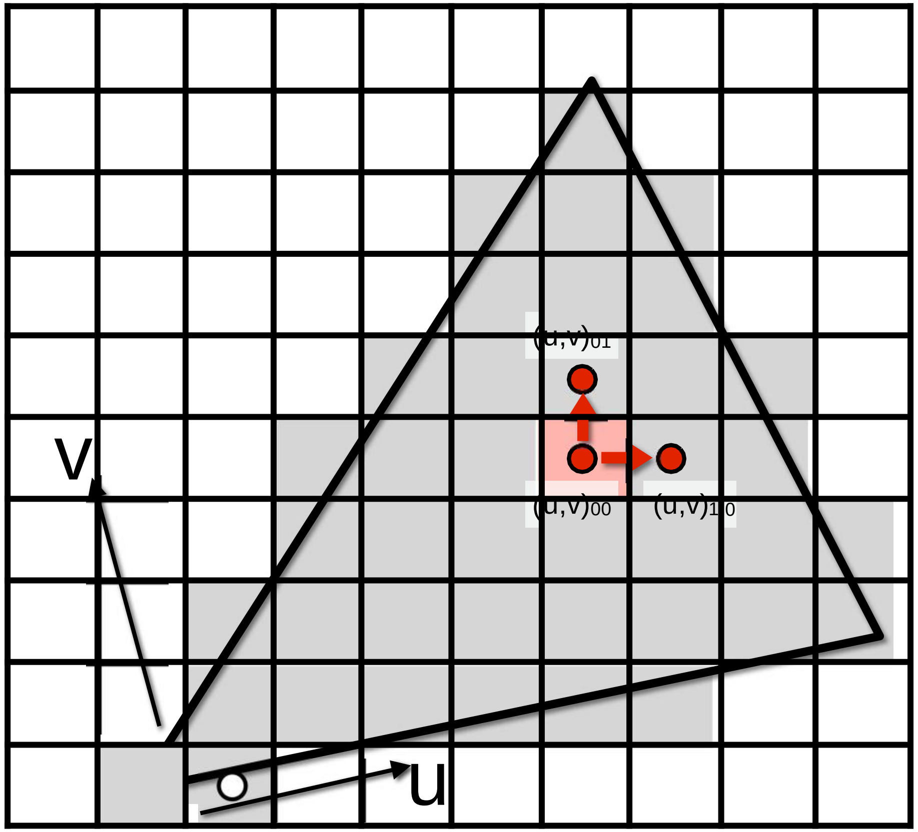
Computing Mipmap Level D

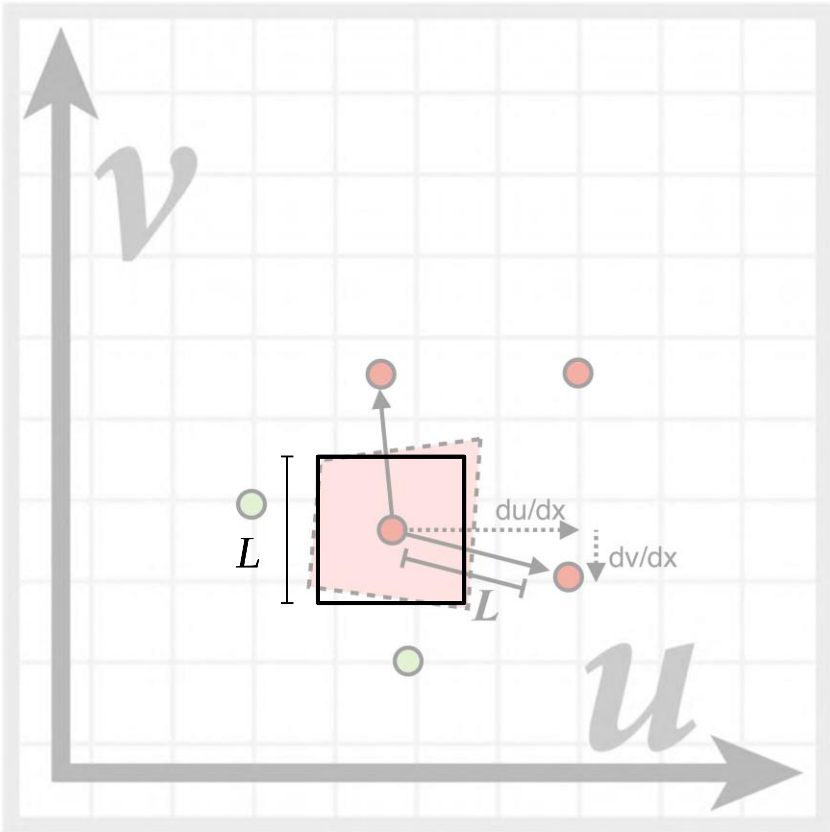




$$D = \log_2 L \qquad L = \max\left(\sqrt{\left(\frac{du}{dx}\right)^2 + \left(\frac{dv}{dx}\right)^2}, \sqrt{\left(\frac{du}{dy}\right)^2 + \left(\frac{dv}{dy}\right)^2}\right)$$

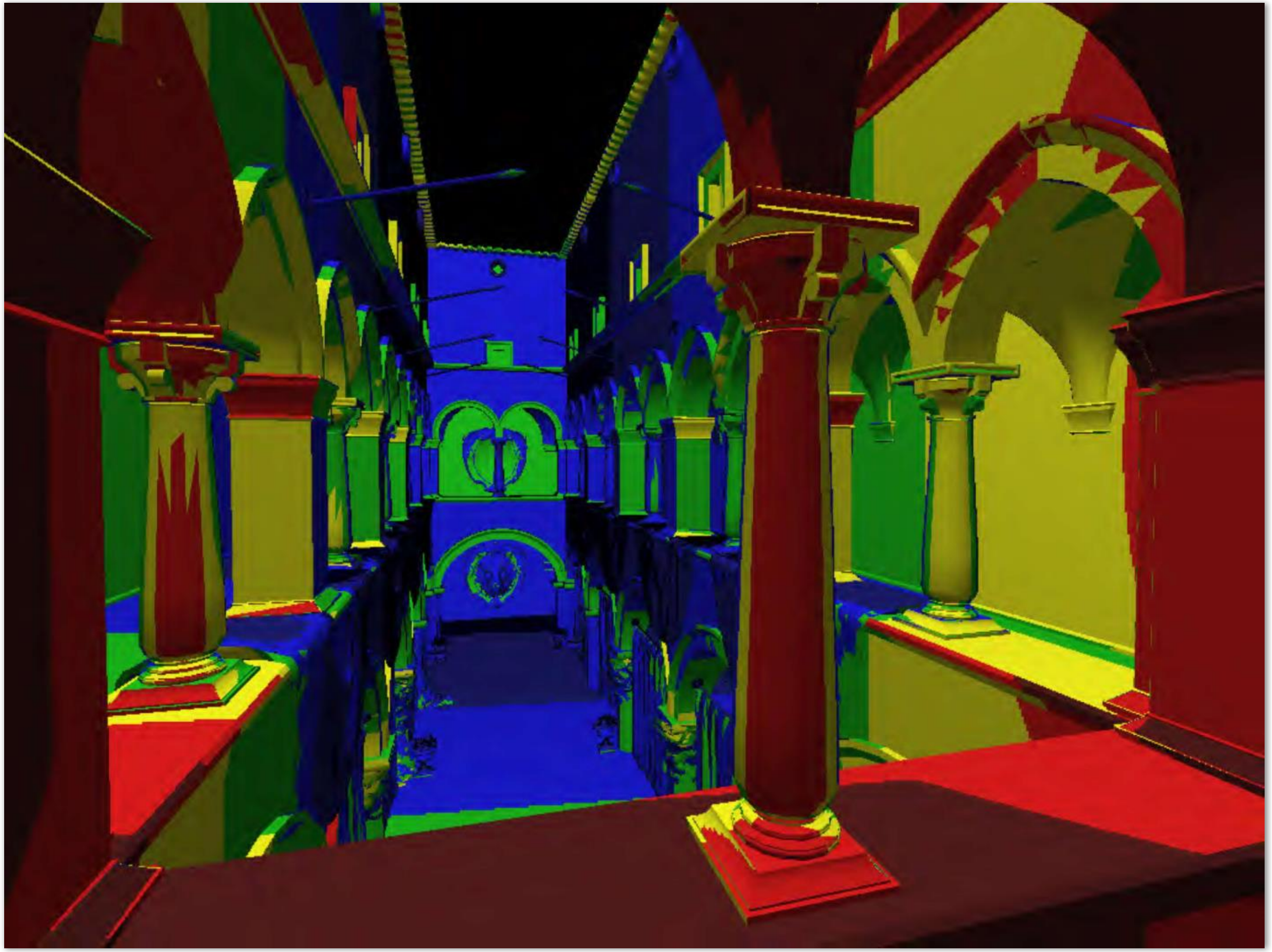
Computing Mipmap Level D





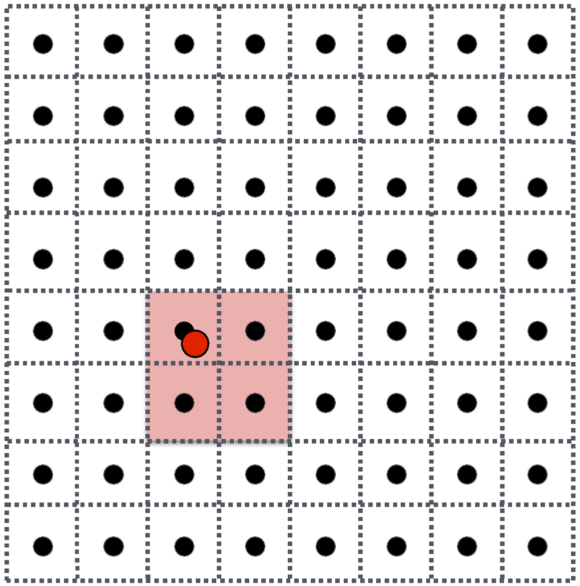
$$D = \log_2 L \qquad L = \max\left(\sqrt{\left(\frac{du}{dx}\right)^2 + \left(\frac{dv}{dx}\right)^2}, \sqrt{\left(\frac{du}{dy}\right)^2 + \left(\frac{dv}{dy}\right)^2}\right)$$

Visualization of Mipmap Level

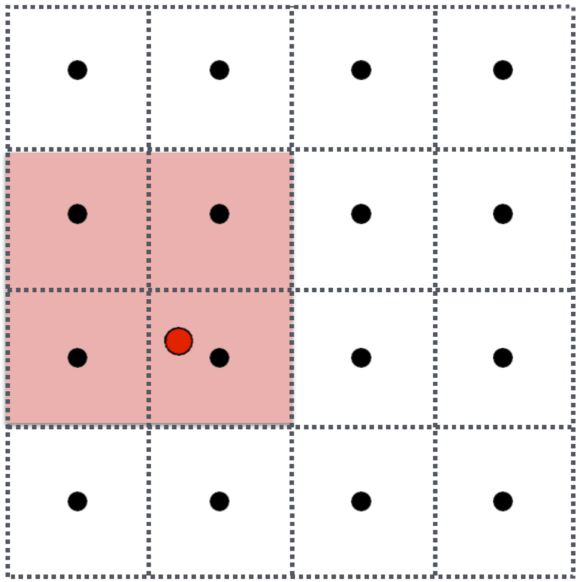


D rounded to nearest integer level

Trilinear Interpolation



Mipmap Level D



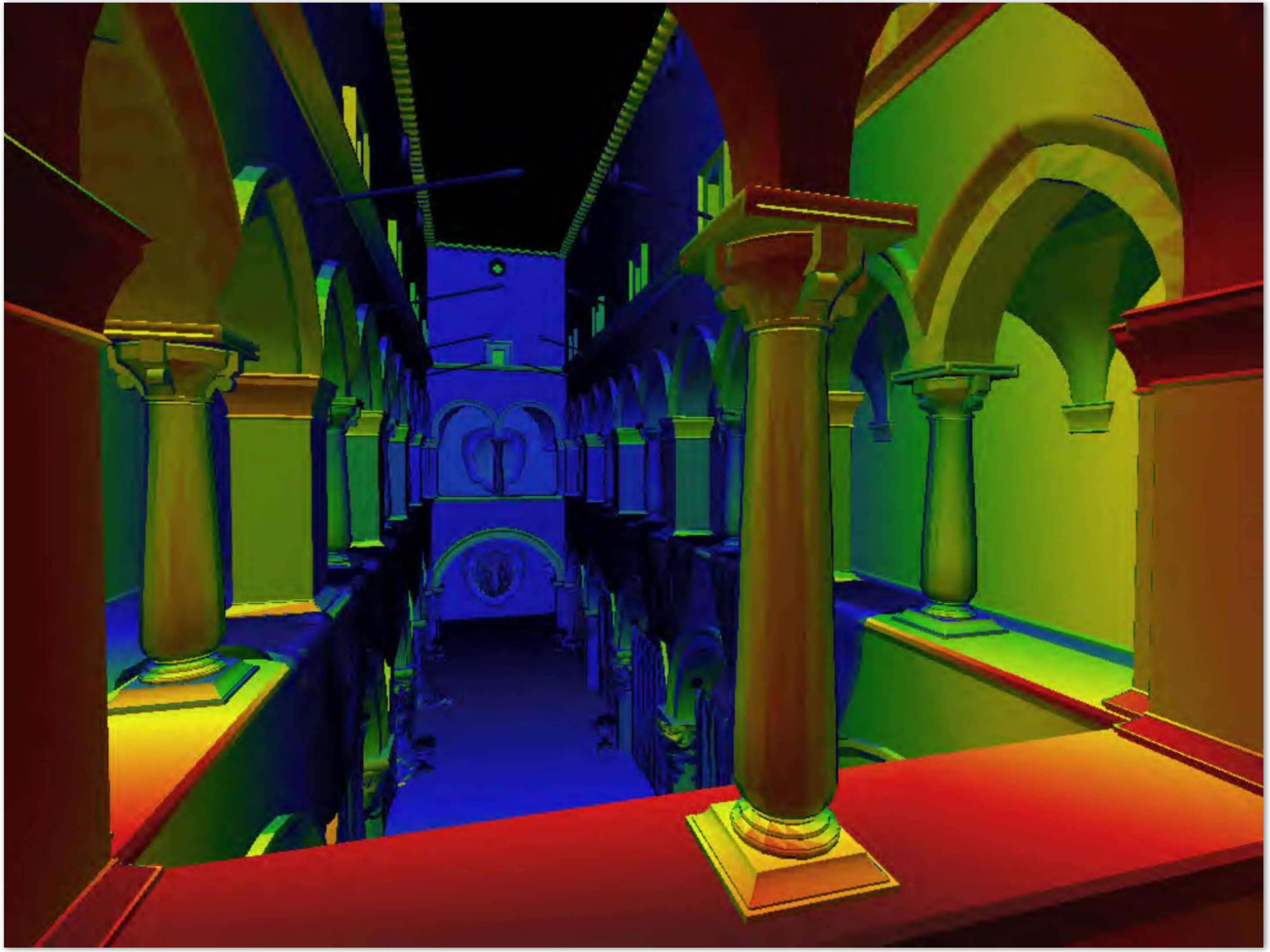
Mipmap Level D+1

Bilinear result

Bilinear result

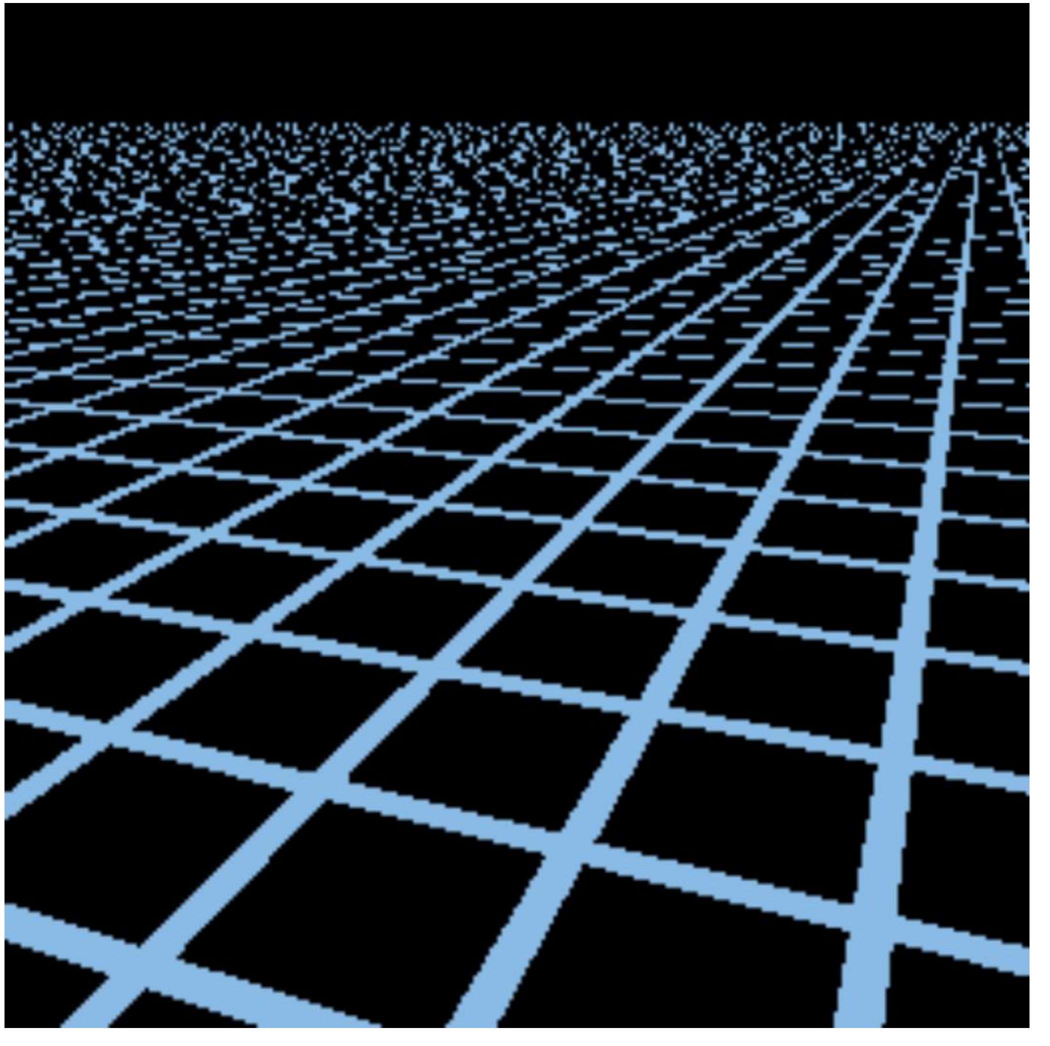
Linear interpolation based on continuous D value

Visualization of Mipmap Level



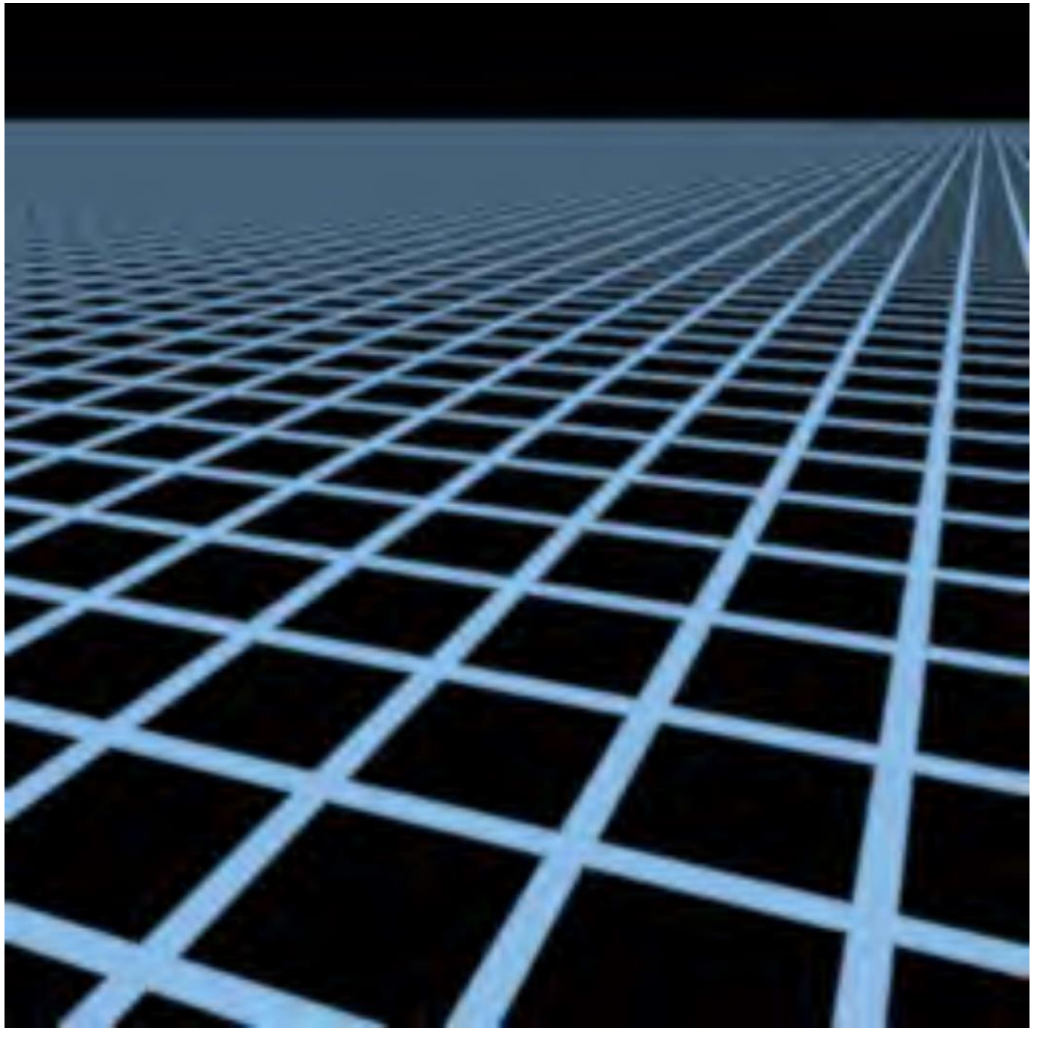
Trilinear filtering: visualization of continuous D

Mipmap Limitations



Point sampling

Mipmap Limitations

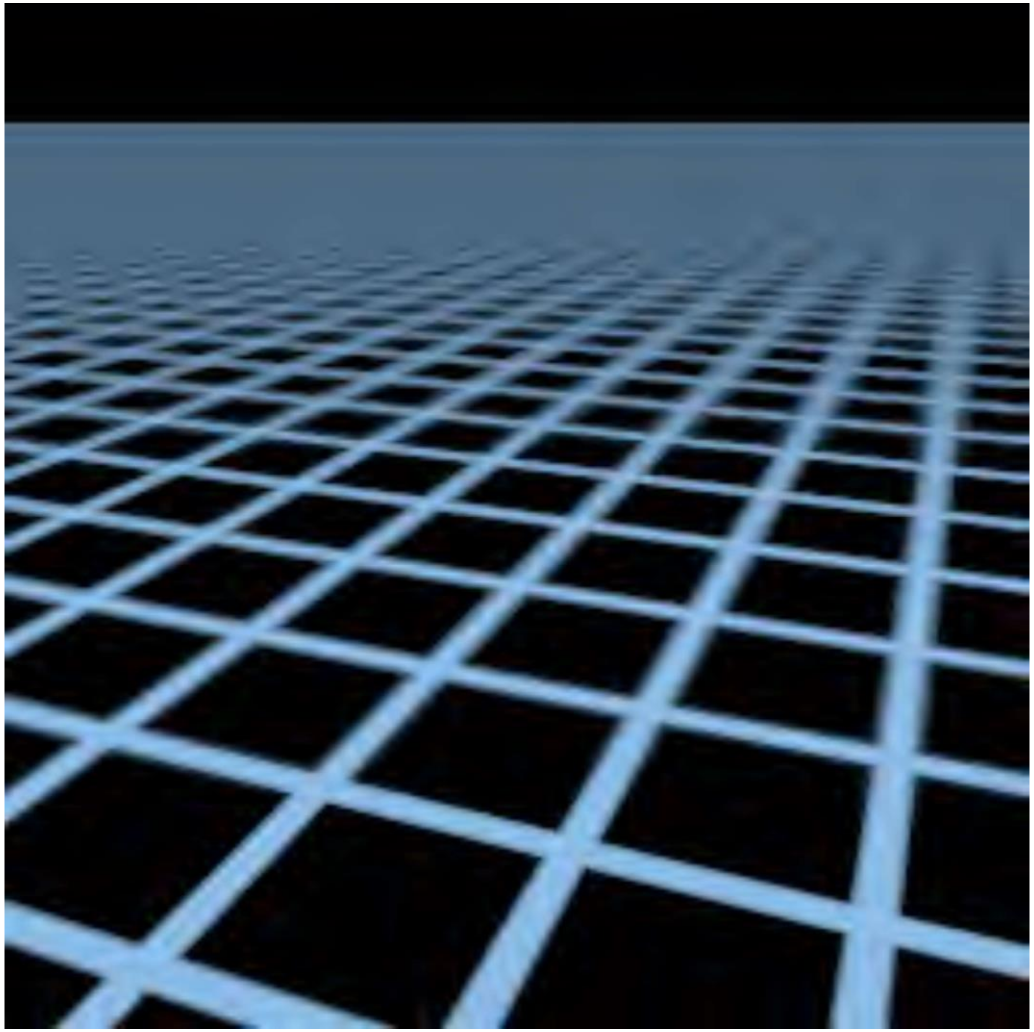


Supersampling 512x (assume this is correct)

Mipmap Limitations

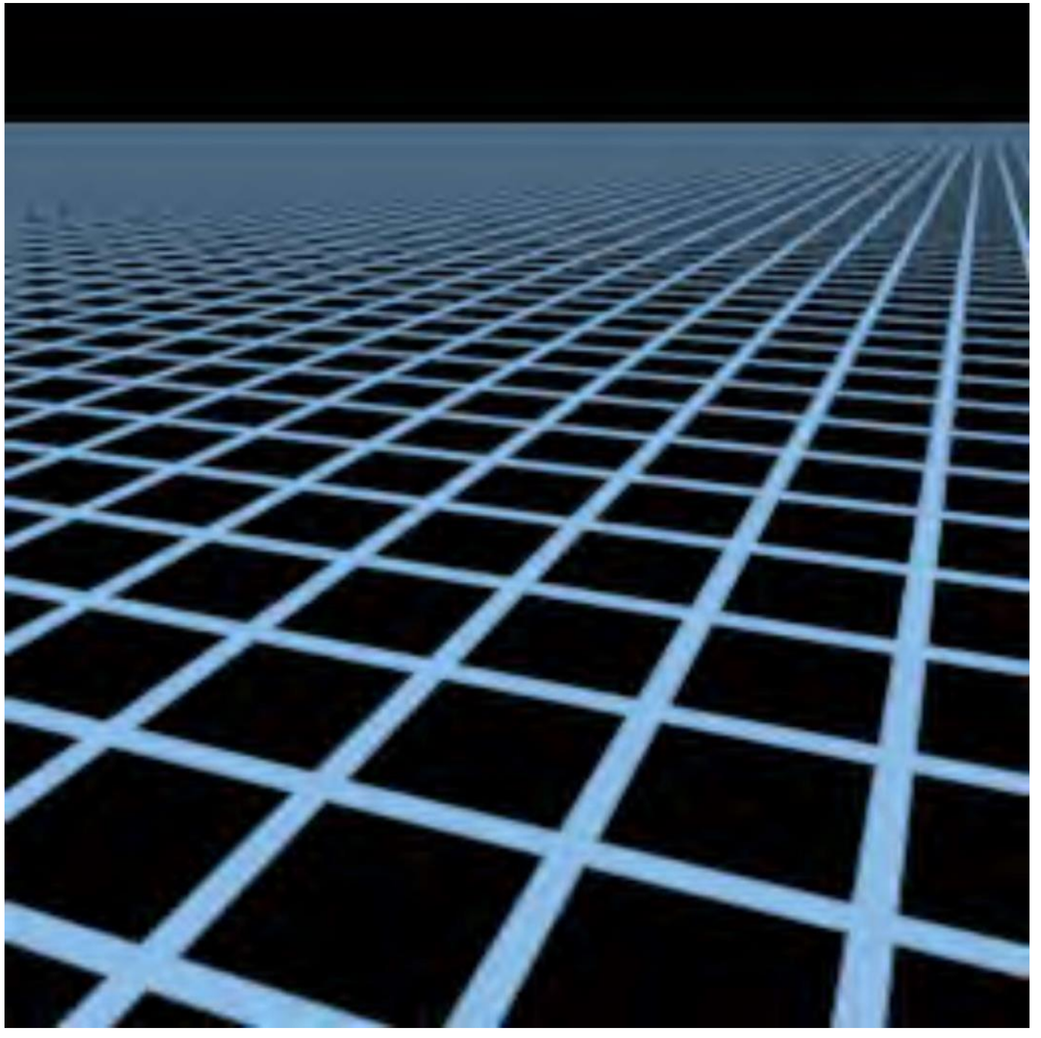
Overblur

Why?



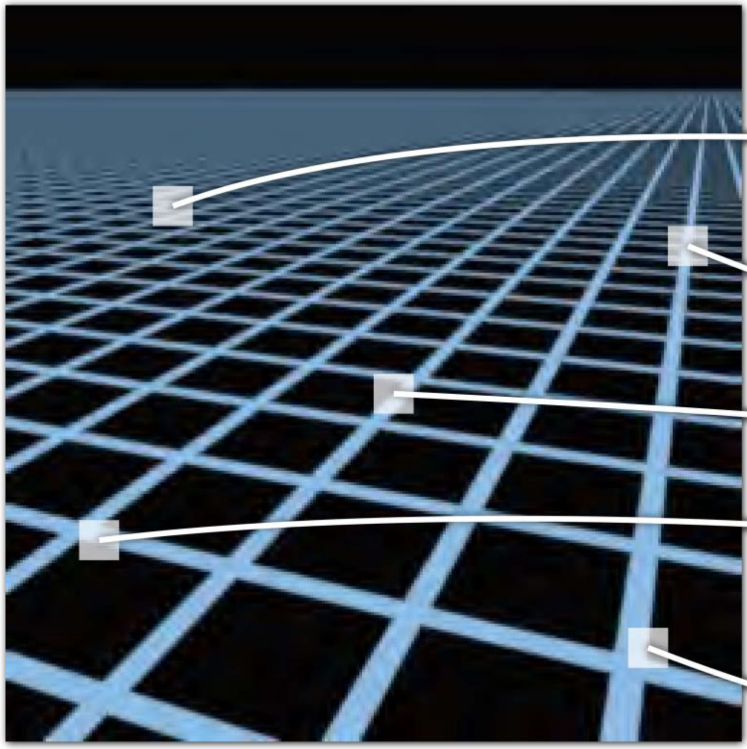
Mipmap trilinear sampling

Anisotropic Filtering

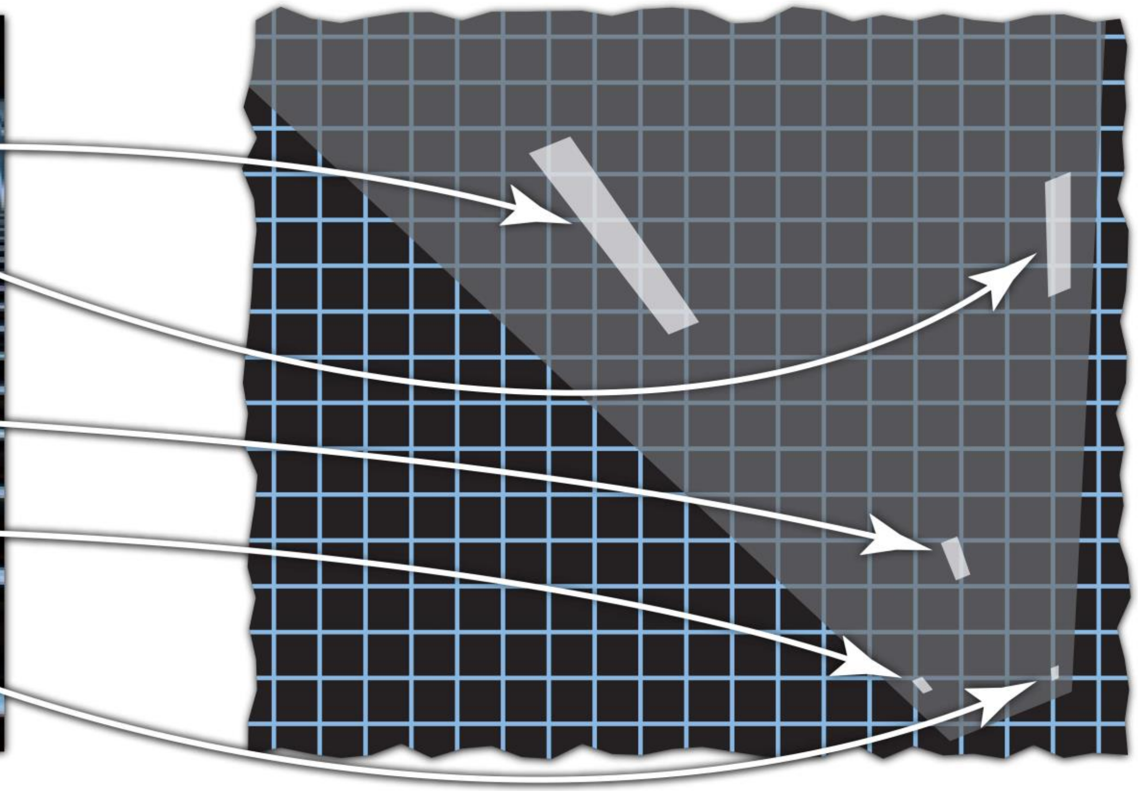


Better than Mipmap!

Irregular Pixel Footprint in Texture



Screen space



Texture space

Anisotropic Filtering

Ripmaps and summed area tables

- Can look up axis-aligned rectangular zones
- Diagonal footprints still a problem

Wikipedia

Anisotropic Filtering

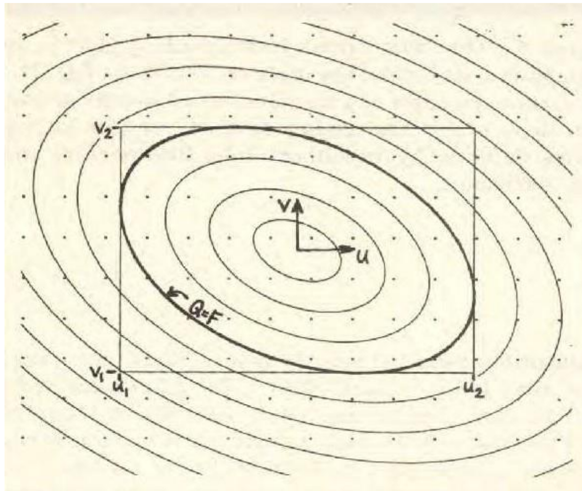
Ripmaps and summed area tables

- Can look up axis-aligned rectangular zones
- Diagonal footprints still a problem

EWA filtering

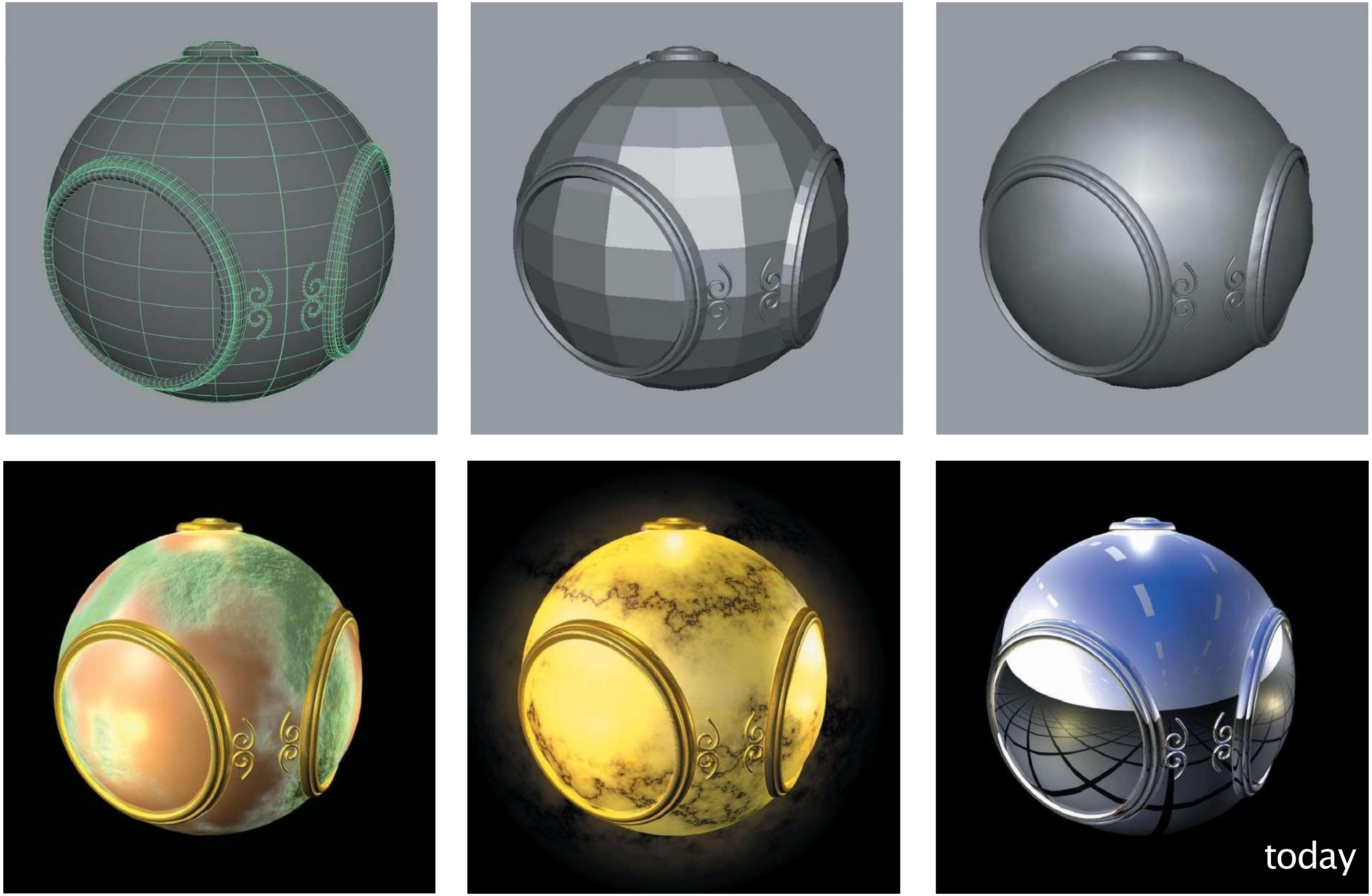
- Use multiple lookups
- Weighted average
- Mipmap hierarchy still helps
- Can handle irregular footprints

Wikipedia



Greene & Heckbert ‘86

Last Lectures



Applications of Textures

Many, Many Uses for Texturing

In modern GPUs, texture = memory + range query (filtering)

- General method to bring data to fragment calculations

Many applications

- Environment lighting
- Store microgeometry
- Procedural textures
- Solid modeling
- Volume rendering
- …

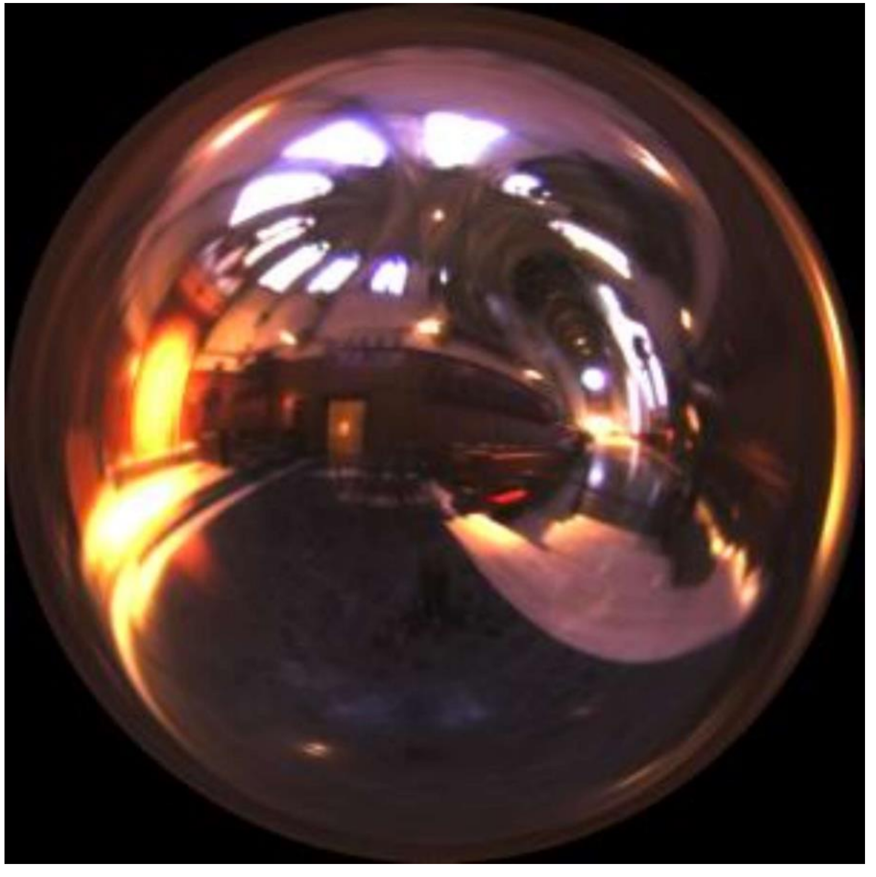
Environment Map

Light from the environment

Rendering with the environment

[Blinn & Newell 1976]

Environmental Lighting



Environment map (left) used to render realistic lighting

Spherical Environment Map

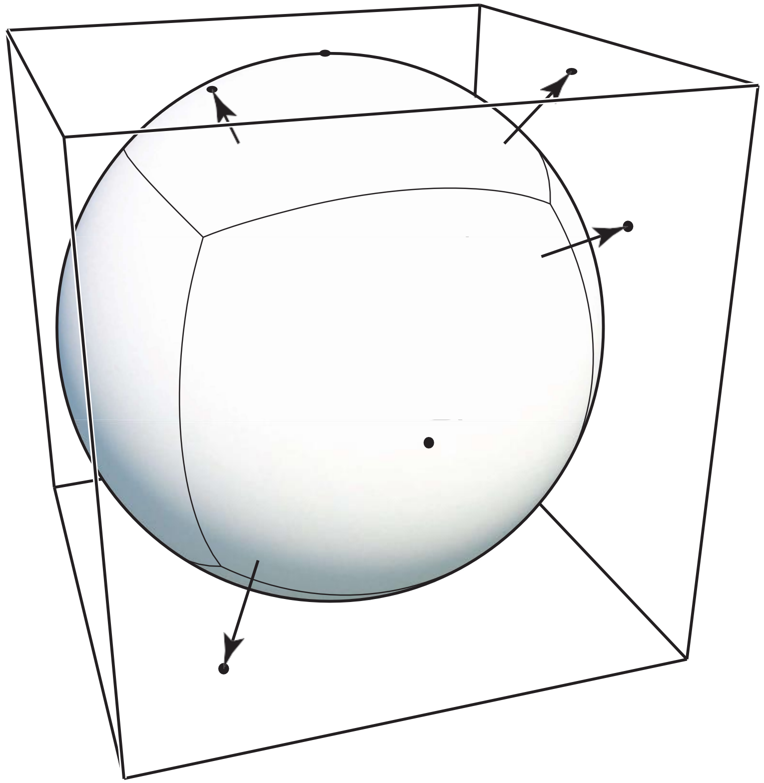
Hand with Reflecting Sphere. M. C. Escher, 1935. lithograph

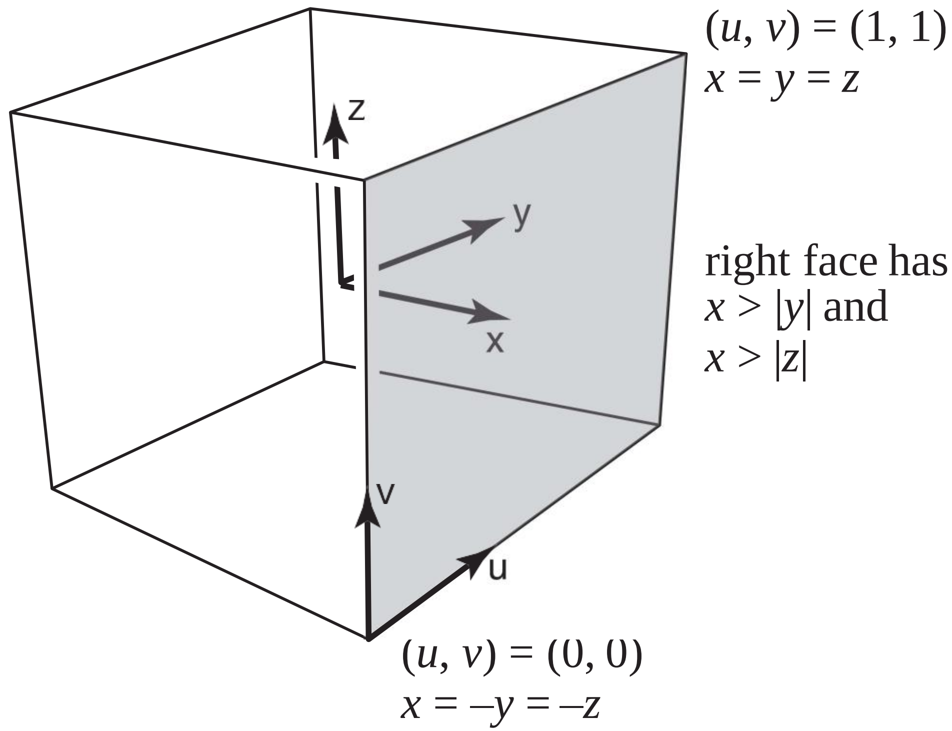
Light Probes, Paul Debevec

Spherical Map — Problem

Prone to distortion (top and bottom parts)!

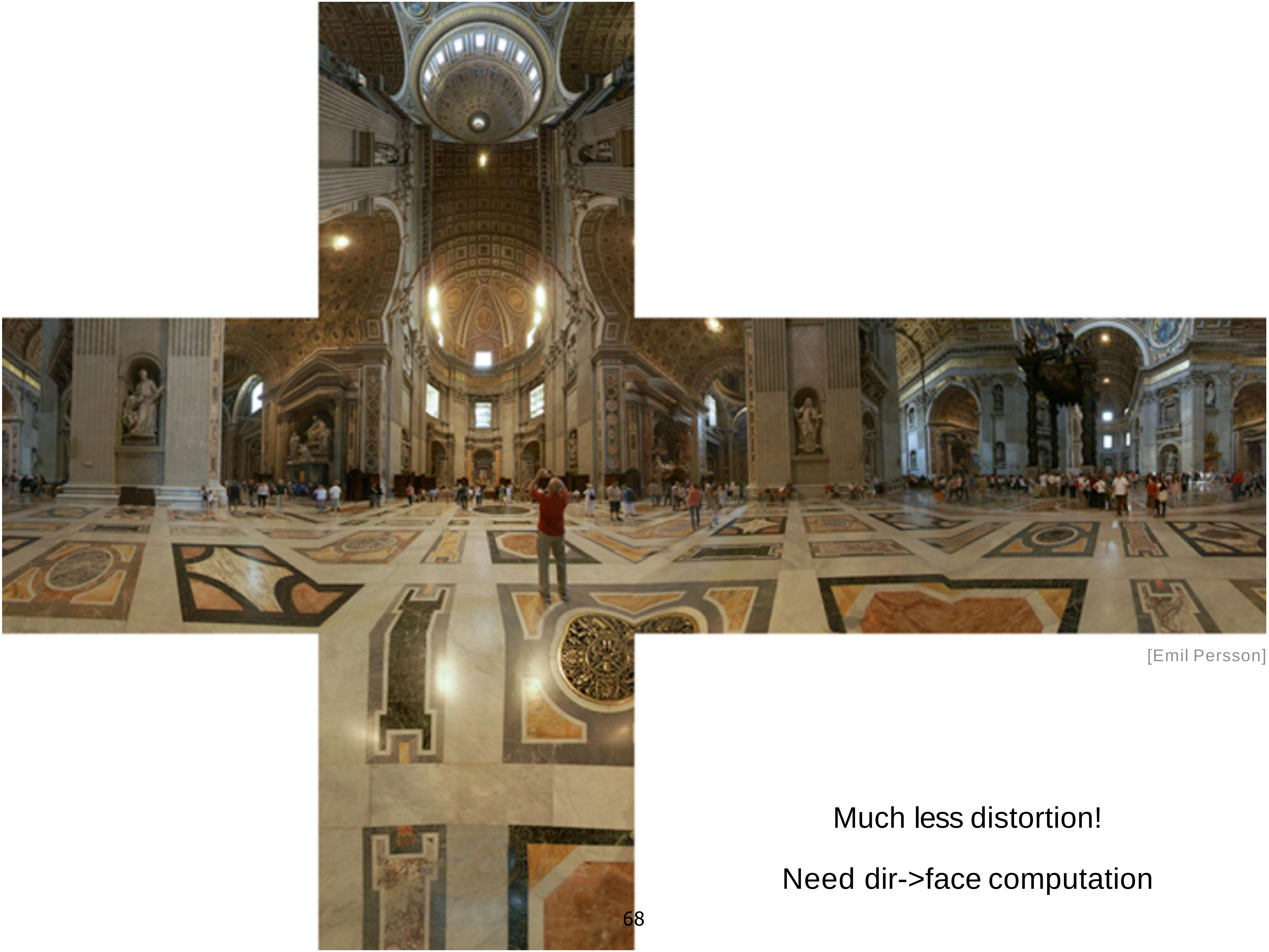
Cube Map





A vector maps to cube point along that direction.

The cube is textured with 6 square texture maps.



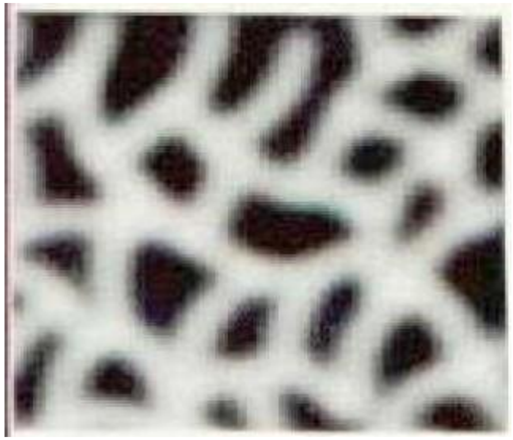
[Emil Persson]

Much less distortion!

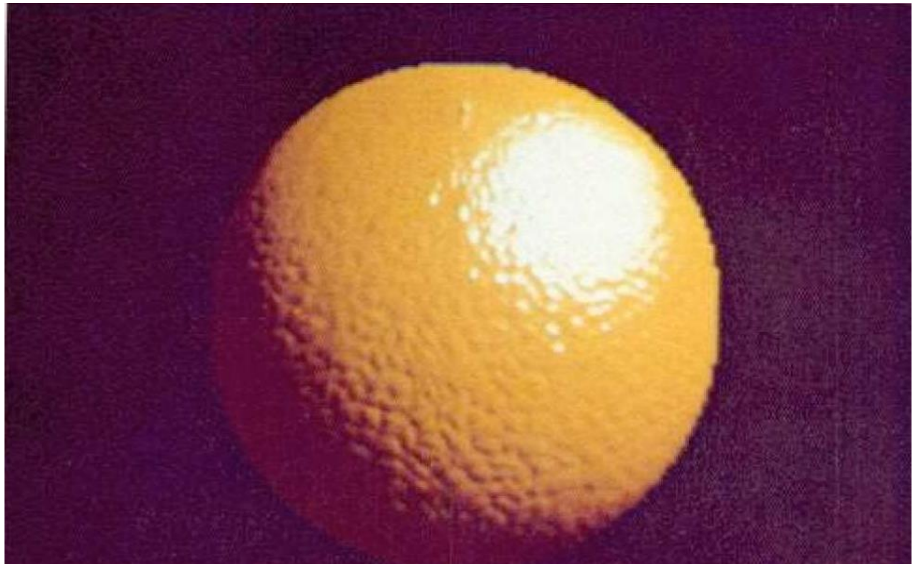
Need dir->face computation

Textures can affect shading!

- Textures doesn't have to only represent colors
  - What if it stores the height / normal?
  - Bump / normal mapping
  - Fake the detailed geometry



Relative height to the underlying surface

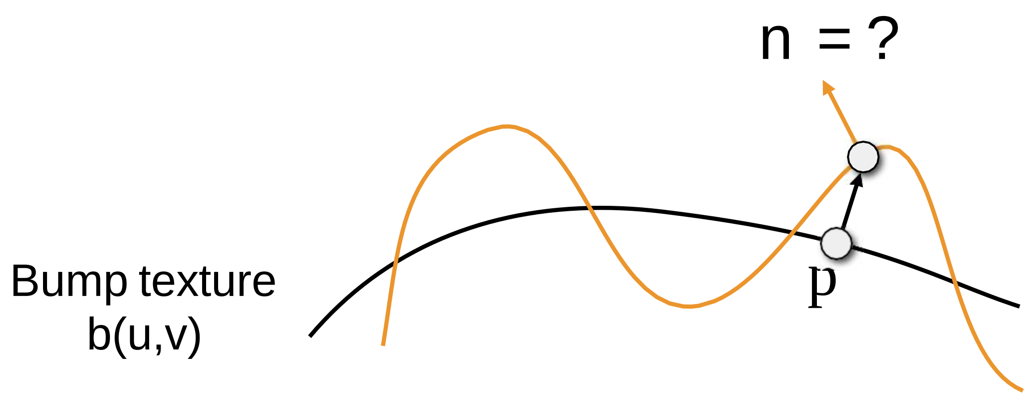


height -> normal -> shading

Bump Mapping

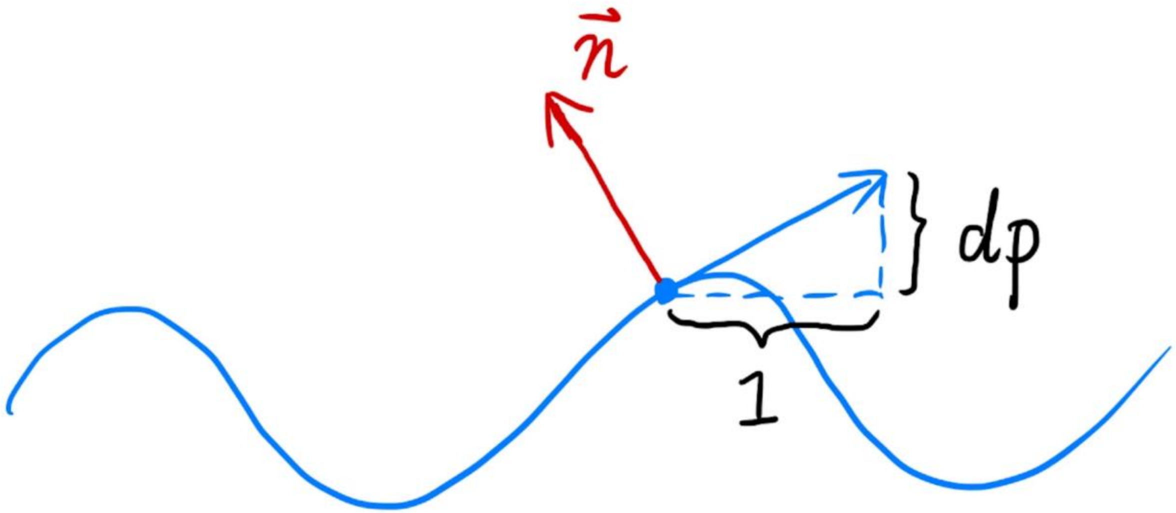
Adding surface detail without adding more triangles

- Perturb surface normal per pixel
  (for shading computations only)
- "Height shift" per texel defined by a texture
- How to modify normal vector?



How to perturb the normal (in flatland)

- Original surface normal n(p) = (0, 1)
- Derivative at p is dp = c * [h(p+1) - h(p)]
- Perturbed normal is then n(p) = (-dp, 1).normalized()

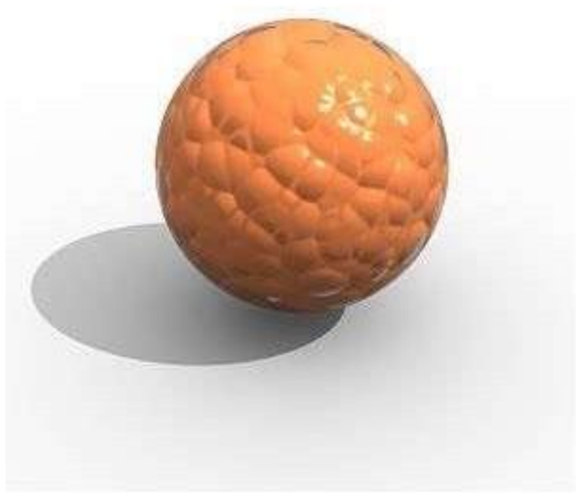


How to perturb the normal (in 3D)

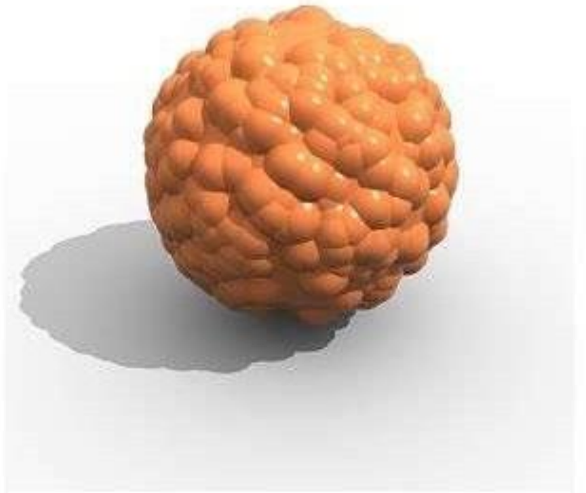
- Original surface normal n(p) = (0, 0, 1)
- Derivatives at p are
  - dp/du = c1 * [h(u+1) - h(u)]
  - dp/dv = c2 * [h(v+1) - h(v)]
- Perturbed normal is n = (-dp/du, -dp/dv, 1).normalized()
- Note that this is in local coordinate!
  More will be elaborated in FAQ of HW3

Textures can affect shading!

- Displacement mapping — a more advanced approach
  - Uses the same texture as in bumping mapping
  - Actually moves the vertices

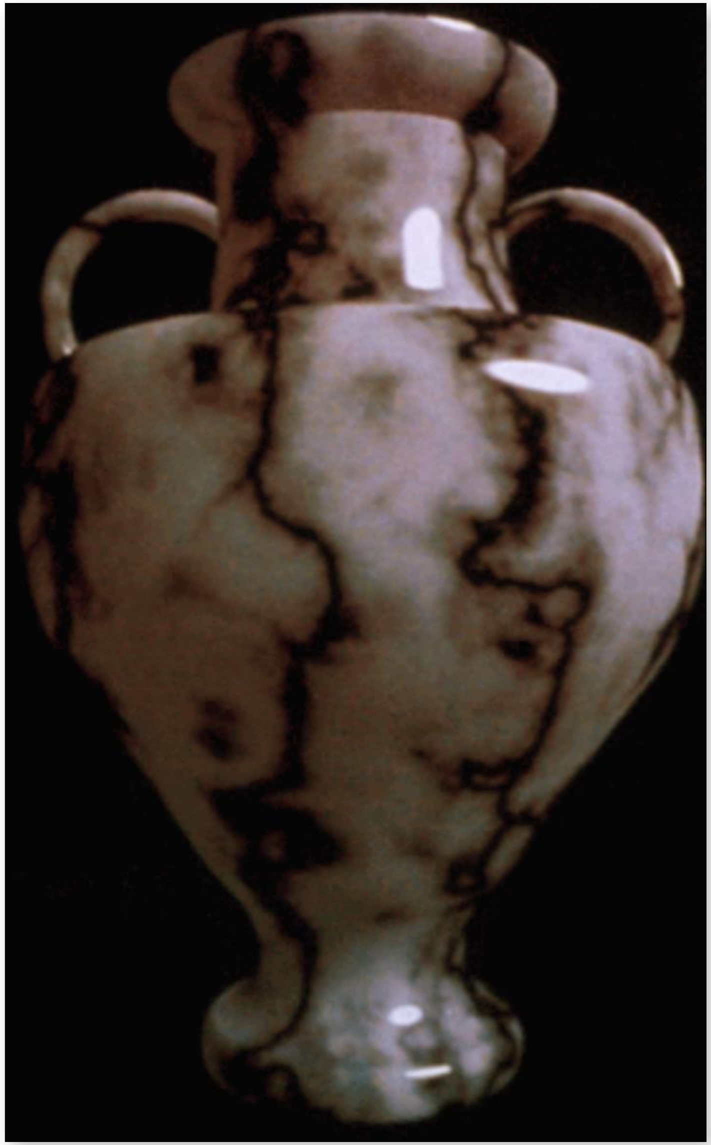


Bump / Normal mapping



Displacement mapping

3D Procedural Noise + Solid Modeling



Perlin noise, Ken Perlin

Provide Precomputed Shading

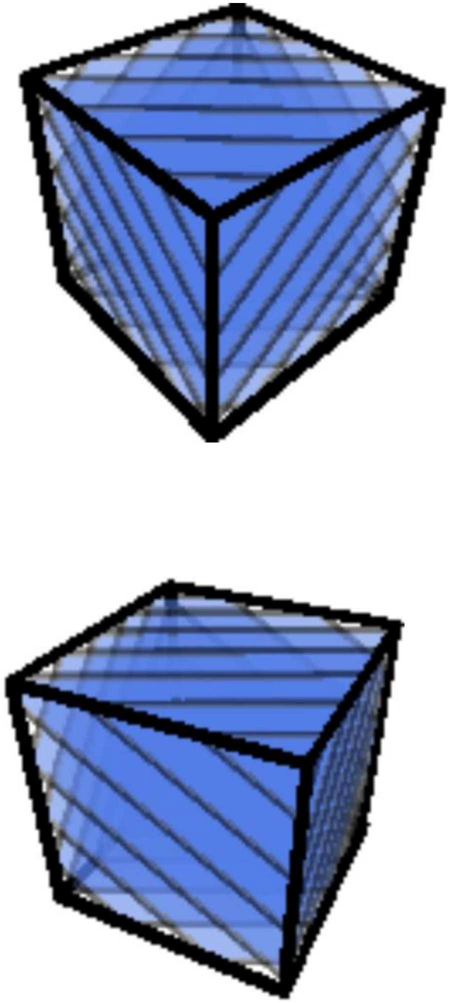
Simple shading

Ambient occlusion texture map

With ambient occlusion

Autodesk

3D Textures and Volume Rendering



Thank you!