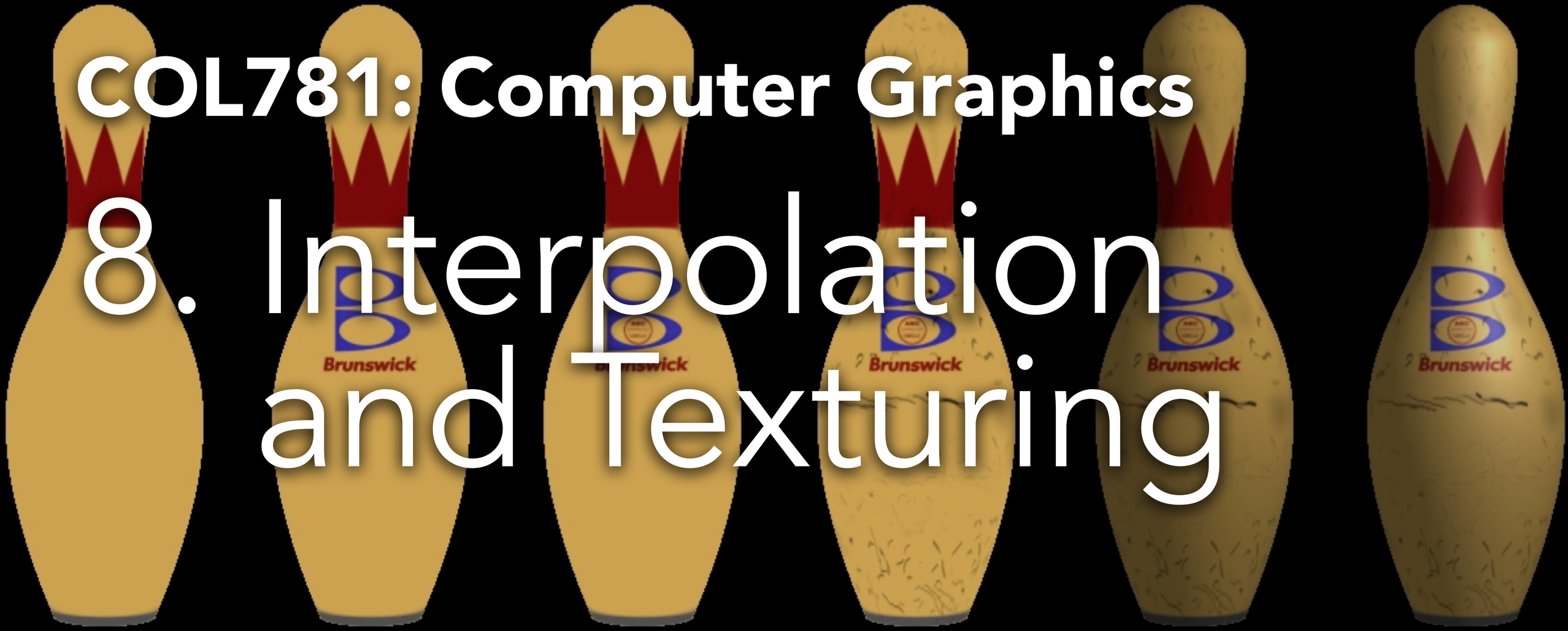
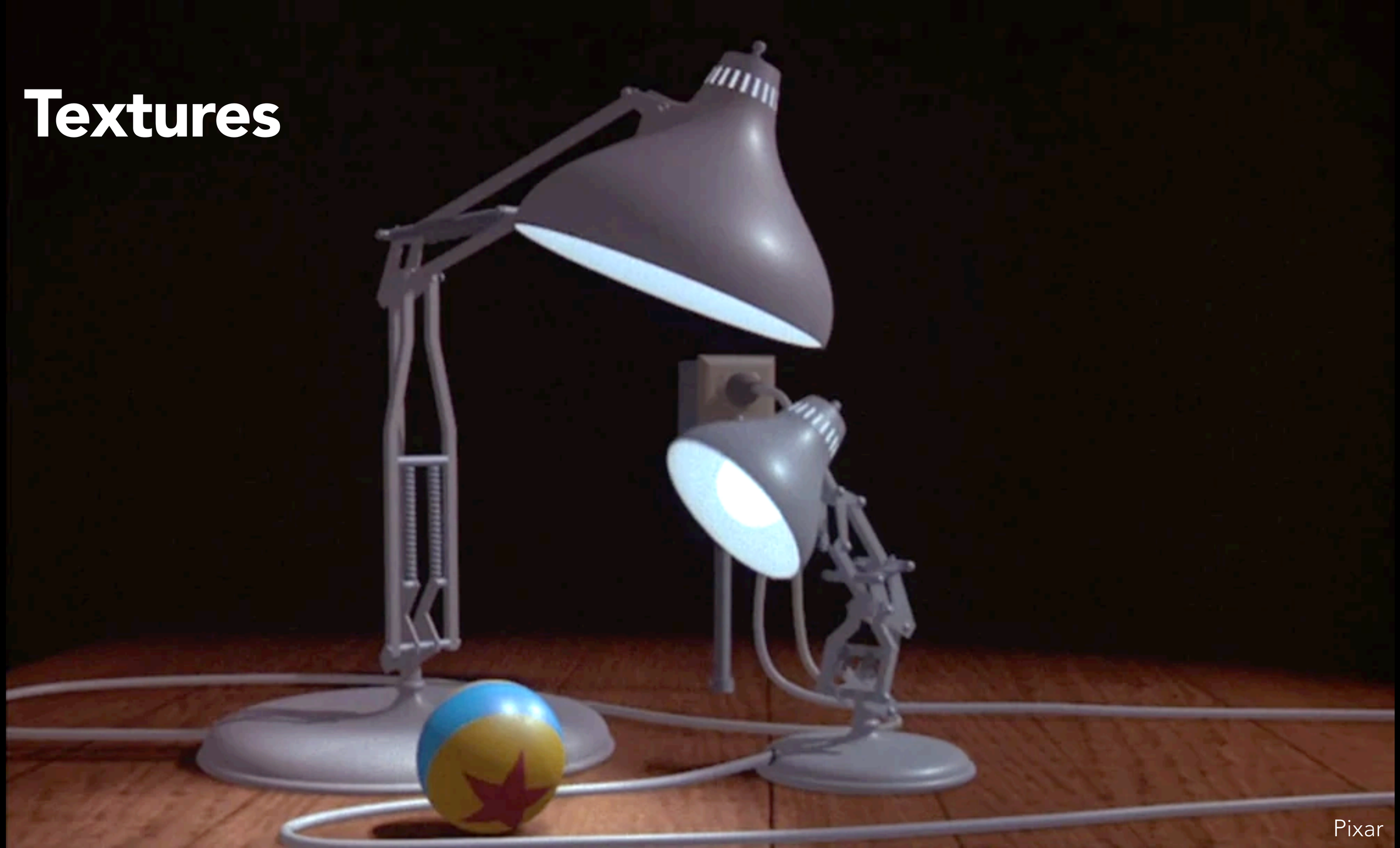
COL781: Computer Graphics

8. Interpolation and Texturing



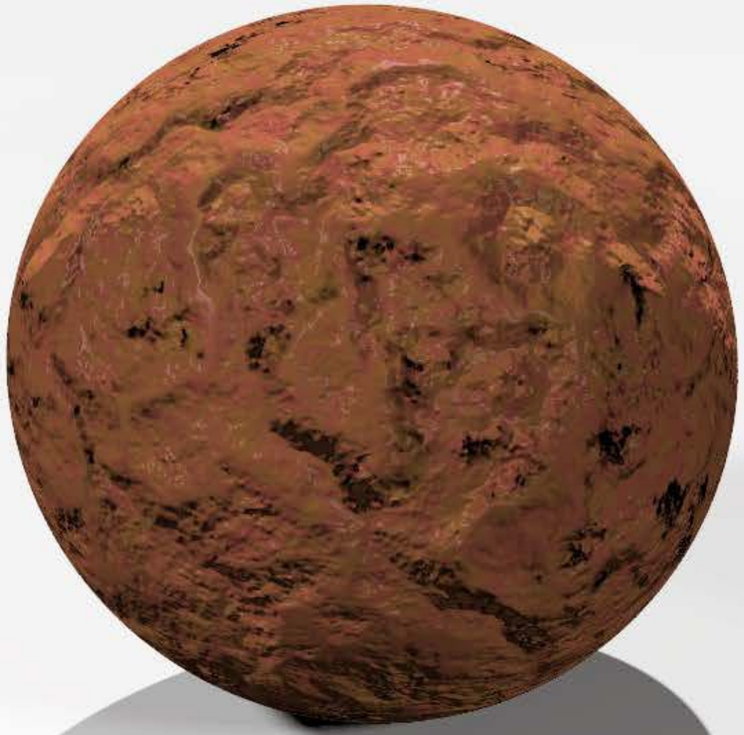
Chan et al.

Textures



Pixar

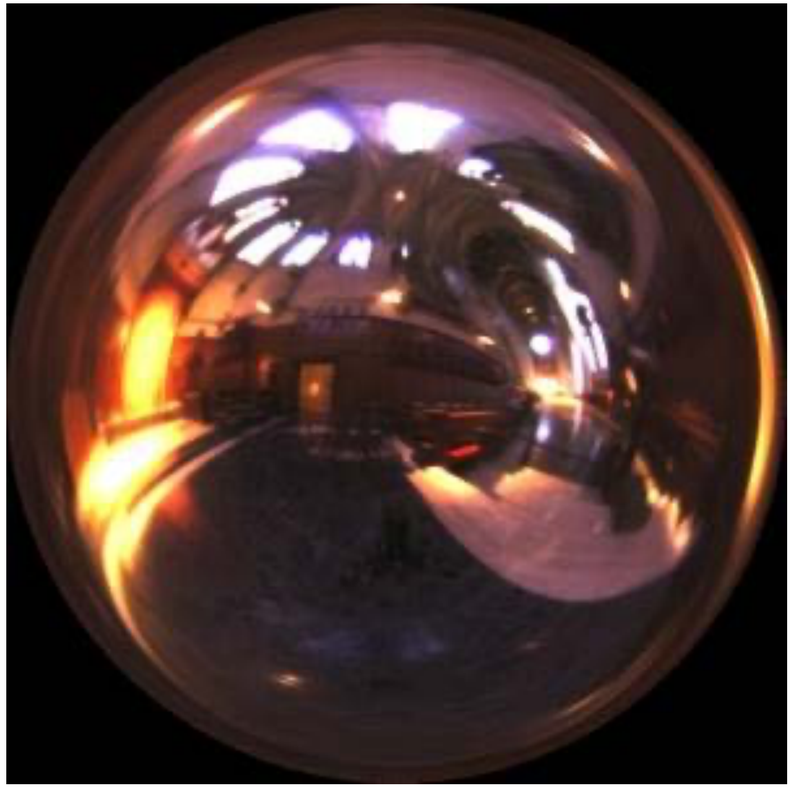

Crytek
(C)2013 CRYTEK GMBH. ALL RIGHTS RESERVED. RYSE IS A REGISTERED TRADEMARK OF CRYTEK GMBH
RYSE
SON OF ROME



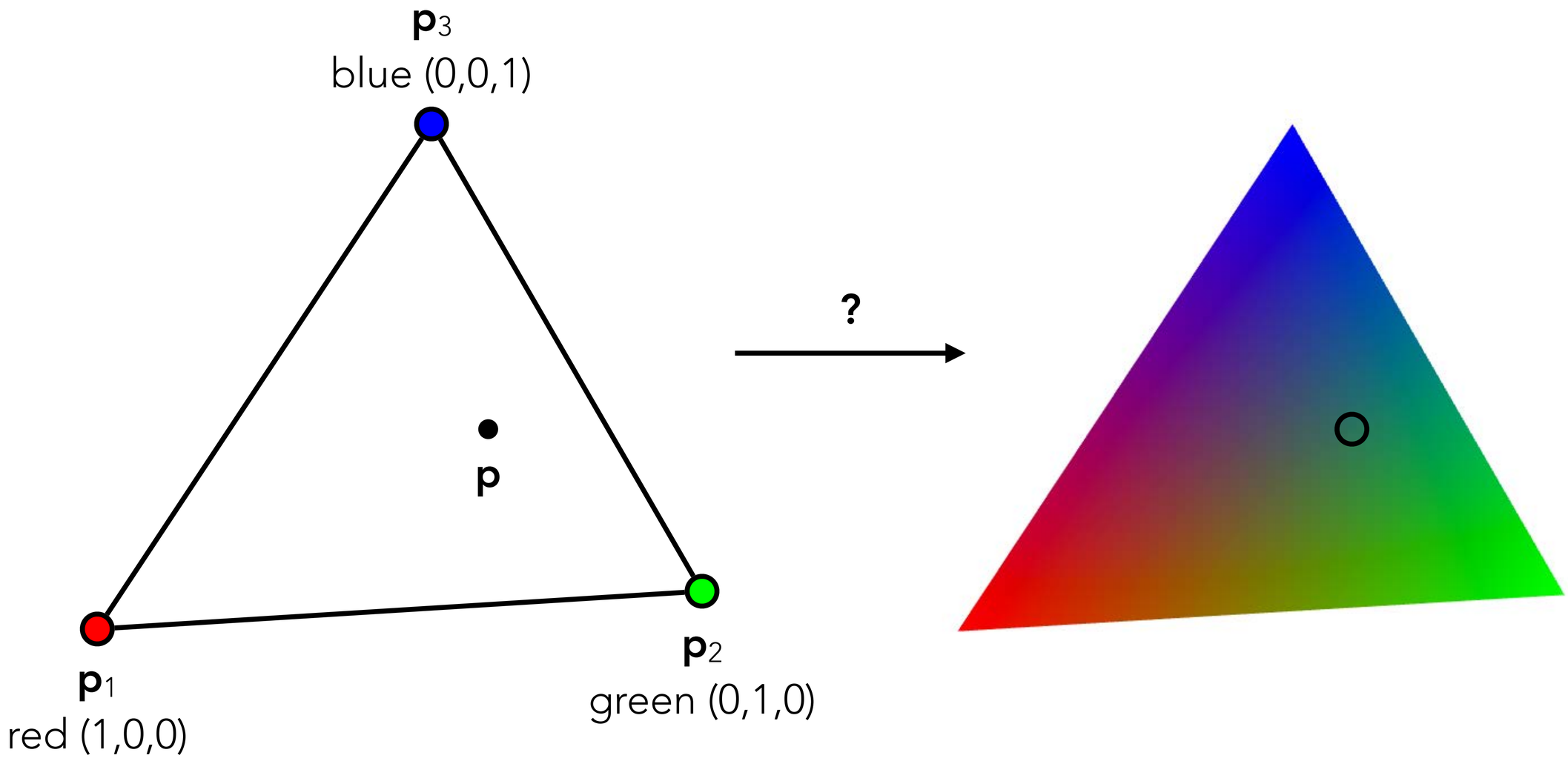
Normal mapping

Displacement mapping

Environment maps



Warm-up: Vertex colours

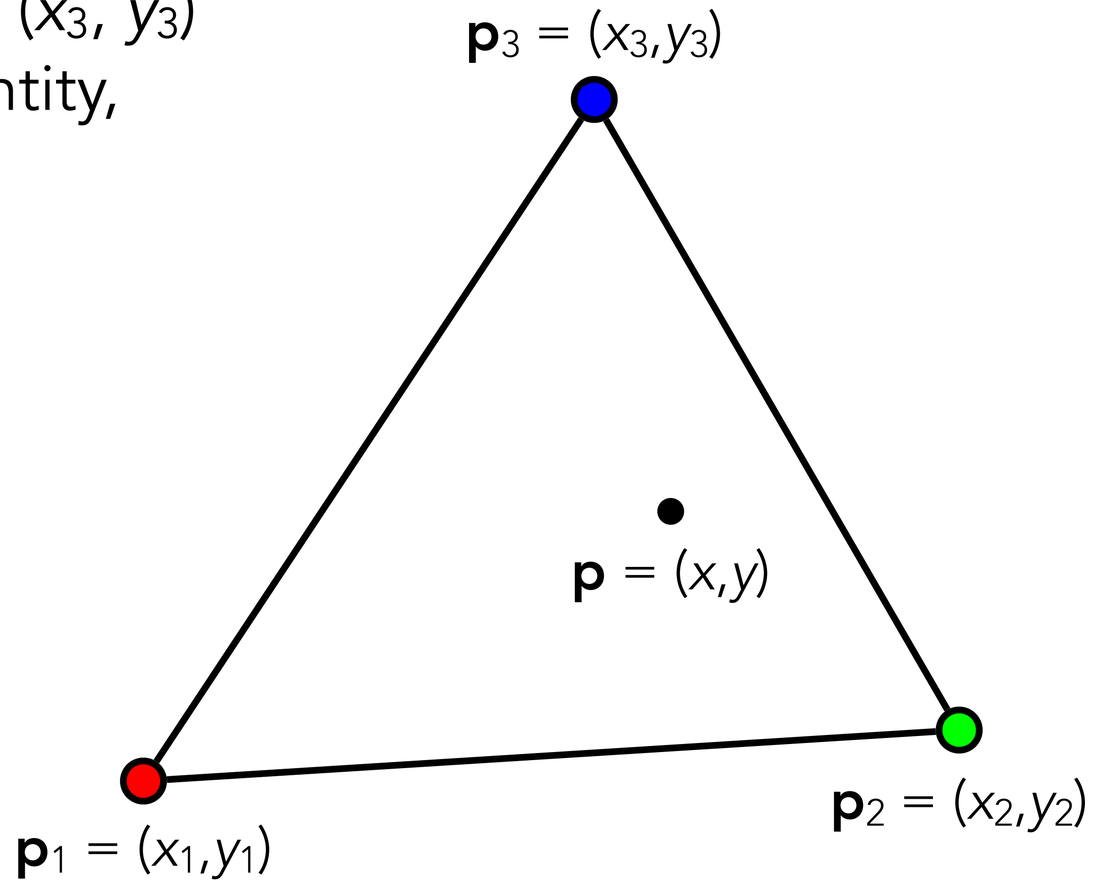


Given vertices $\mathbf{p}_1 = (x_1, y_1)$, $\mathbf{p}_2 = (x_2, y_2)$, $\mathbf{p}_3 = (x_3, y_3)$ with specified values q_1, q_2, q_3 of some quantity, what should be q on some other point $\mathbf{p}$?

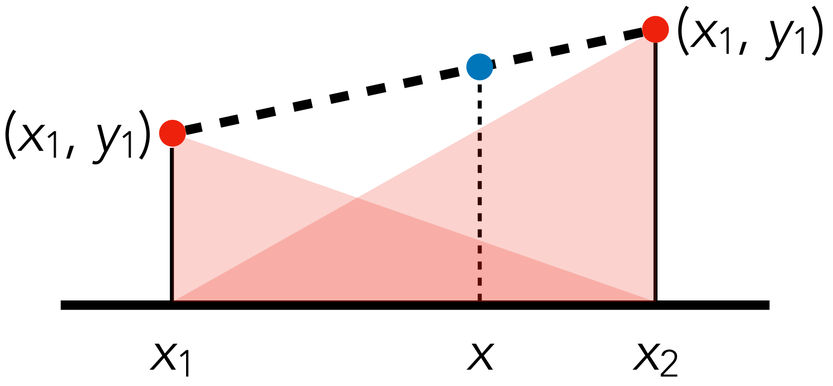
One way: Assume q is an affine function $q(x, y) = ax + by + c$. Find a, b, c by solving

$$q_1 = ax_1 + by_1 + c$$
$$q_2 = ax_2 + by_2 + c$$
$$q_3 = ax_3 + by_3 + c$$

Works, but tedious!



Consider linear interpolation in 1D first.



Strategy 1: Let $y(x) = ax + b$, solve for a, b…

Strategy 2: Let $t = \dfrac{x - x_1}{x_2 - x_1}$, then $y(x) = (1-t)\, y_1 + t\, y_2$.

Actually a linear combination of two basis functions:

$$\phi_1 = 1 - t, \quad \phi_2 = t$$
$$y(x) = \phi_1 y_1 + \phi_2 y_2$$

Notice that $\phi_1(x_1) = 1$, $\phi_1(x_2) = 0$ and similarly for ϕ_2. Also $\phi_1(x) + \phi_2(x) = 1$

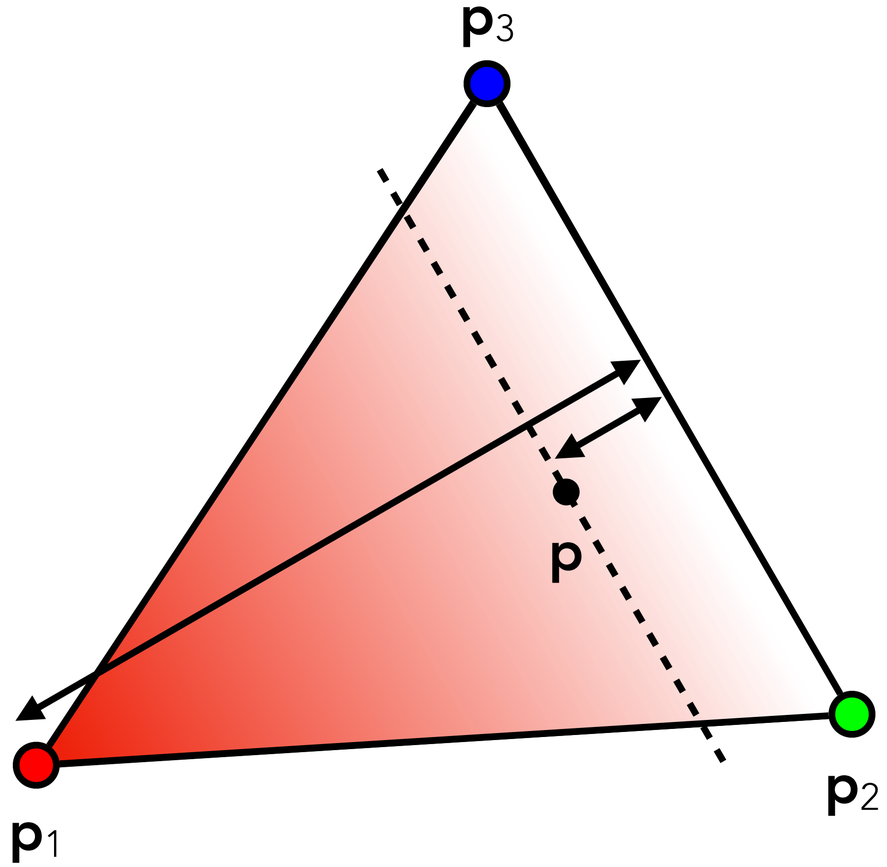
Can we do the same for a triangle?

Choose basis functions ϕ_1, ϕ_2, ϕ_3 such that

- $\phi_1 = 1$ at $\mathbf{p}_1$, 0 at $\mathbf{p}_2$ and $\mathbf{p}_3$
- Similarly for ϕ_2 and ϕ_3…
- $\phi_1 + \phi_2 + \phi_3 = 1$ everywhere.

$$\phi_1(\mathbf{p}) = \frac{\text{dist}(\mathbf{p}, \mathbf{p}_2\mathbf{p}_3)}{\text{dist}(\mathbf{p}_1, \mathbf{p}_2\mathbf{p}_3)}$$

Do they add up to 1?



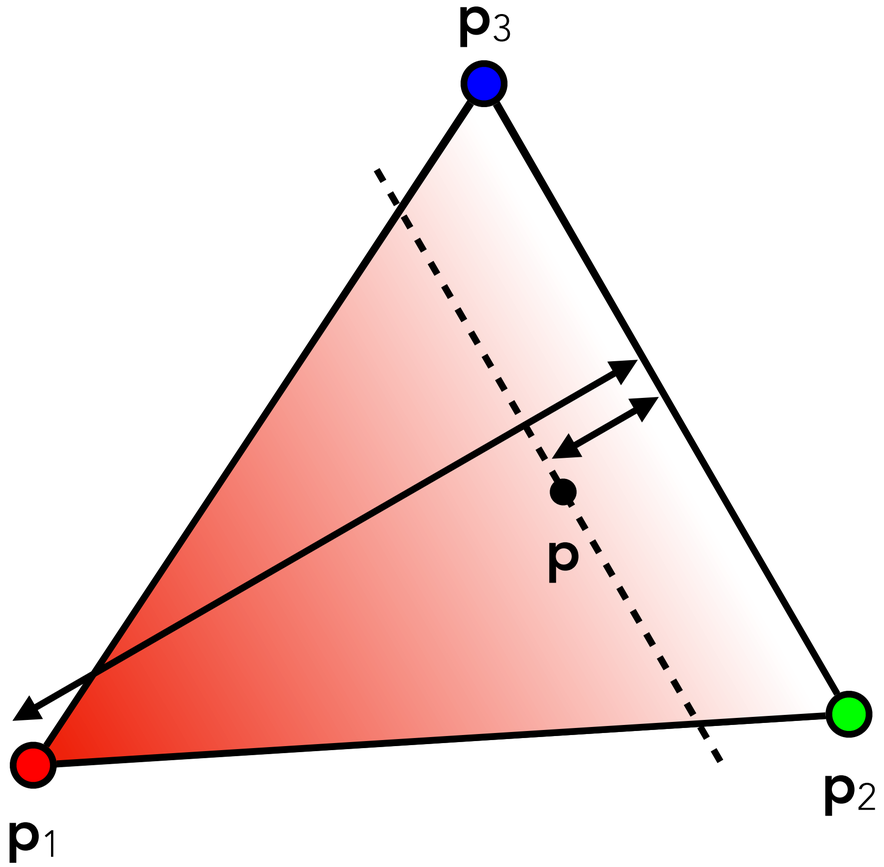
$$\phi_1(\mathbf{p}) = \frac{\text{dist}(\mathbf{p}, \mathbf{p}_2\mathbf{p}_3)}{\text{dist}(\mathbf{p}_1, \mathbf{p}_2\mathbf{p}_3)}$$

Similarly for ϕ_2 and ϕ_3. Do they add up to 1?

"No clever tricks" solution:

ϕ_1, ϕ_2, ϕ_3 are affine functions of $\mathbf{p}$.
So, $\phi_1 + \phi_2 + \phi_3$ is also affine in $\mathbf{p}$.

But $\phi_1 + \phi_2 + \phi_3 = 1$ at $\mathbf{p}_1$, $\mathbf{p}_2$, $\mathbf{p}_3$.

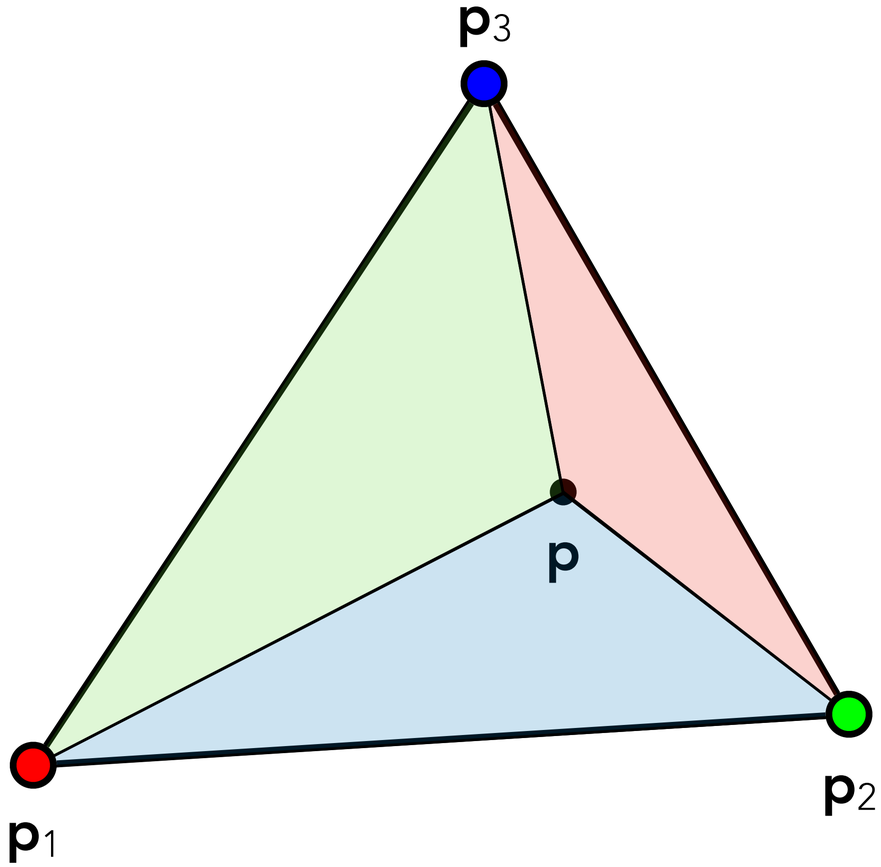


An affine function in 2D is uniquely determined by its values at 3 points, so it must be 1 everywhere.

$$\phi_1(\mathbf{p}) = \frac{\text{area}(\mathbf{p}\mathbf{p}_2\mathbf{p}_3)}{\text{area}(\mathbf{p}_1\mathbf{p}_2\mathbf{p}_3)}$$

$$\phi_2(\mathbf{p}) = \frac{\text{area}(\mathbf{p}_1\mathbf{p}\mathbf{p}_3)}{\text{area}(\mathbf{p}_1\mathbf{p}_2\mathbf{p}_3)}$$

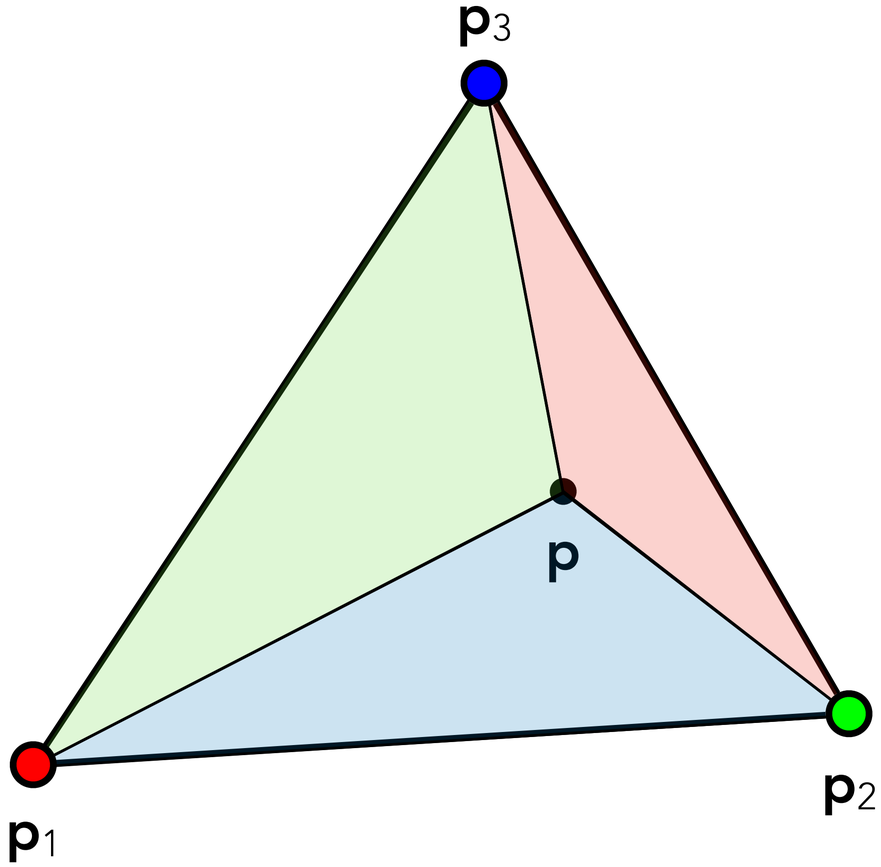
$$\phi_3(\mathbf{p}) = \frac{\text{area}(\mathbf{p}_1\mathbf{p}_2\mathbf{p})}{\text{area}(\mathbf{p}_1\mathbf{p}_2\mathbf{p}_3)}$$



Barycentric coordinates

The values of ϕ_1, ϕ_2, ϕ_3 act as a (redundant) coordinate system for the triangle!

- $(\phi_1,\phi_2,\phi_3) = (1, 0, 0)$: $\mathbf{p} = \mathbf{p}_1$
- $(\phi_1,\phi_2,\phi_3) = (0, ½, ½)$: $\mathbf{p}$ = midpoint of $\mathbf{p}_2$ & $\mathbf{p}_3$
- $(\phi_1,\phi_2,\phi_3) = (⅓, ⅓, ⅓)$: $\mathbf{p}$ = centroid

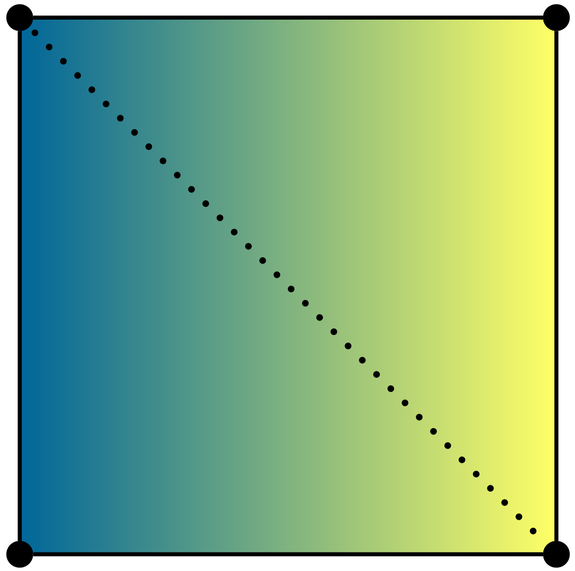


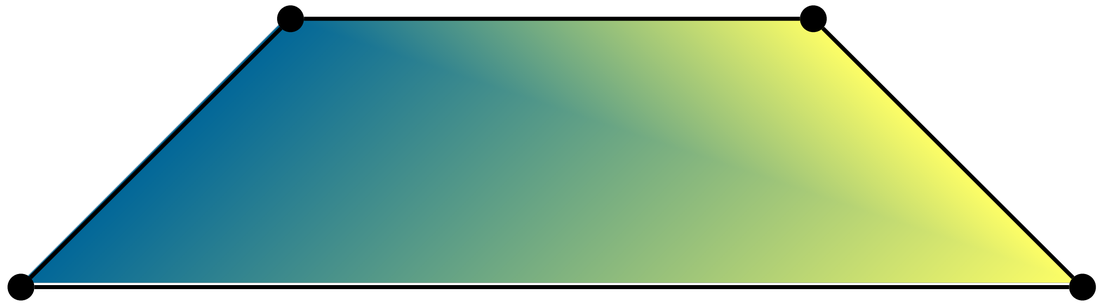
In general, $\mathbf{p} = \phi_1\mathbf{p}_1 + \phi_2\mathbf{p}_2 + \phi_3\mathbf{p}_3$

Interpolate anything else via $q(\mathbf{p}) = \phi_1 q_1 + \phi_2 q_2 + \phi_3 q_3$

Puzzle:

I take a square made of two triangles, map the vertices to the screen using perspective projection, then interpolate colours using barycentric coordinates.

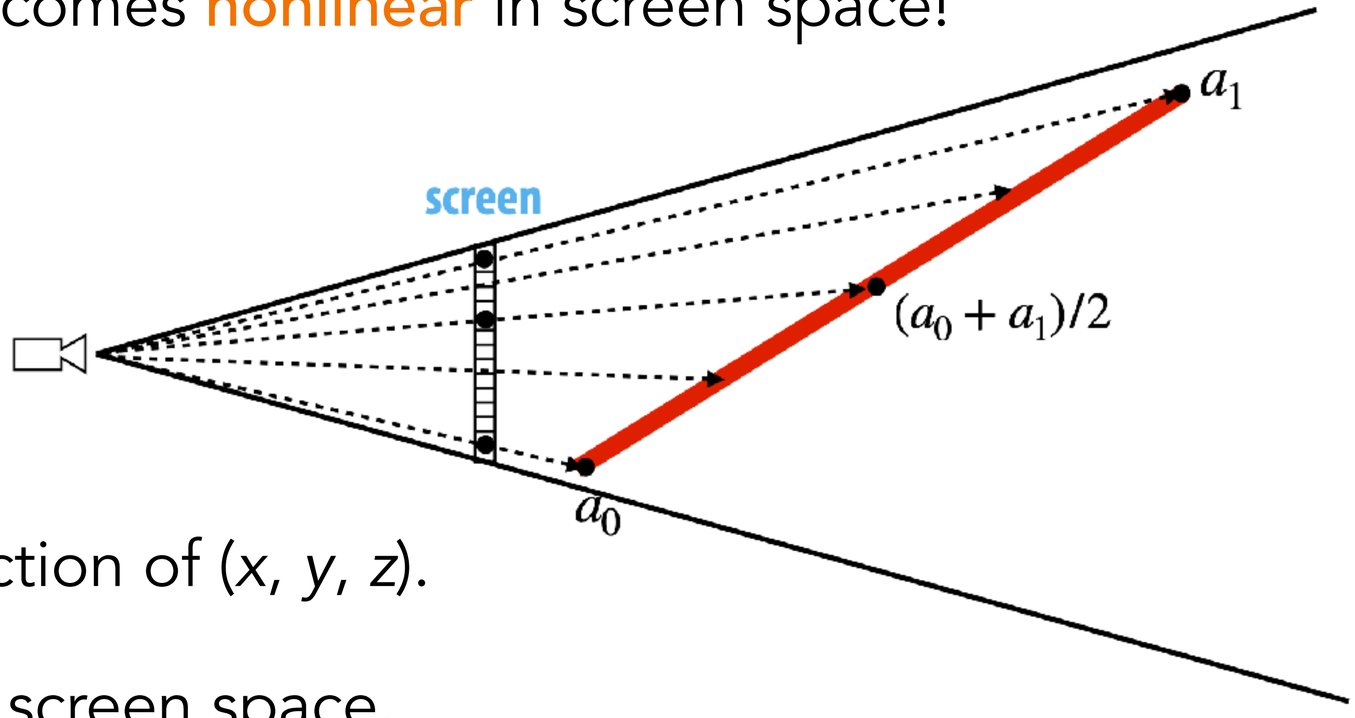




Why does the colour gradient look distorted?

We ought to do linear interpolation in world space.

After perspective division, this becomes nonlinear in screen space!



Mathematically: q is an affine function of (x, y, z).

(x, y, z) is projected to $(x/z, y/z)$ in screen space.

But then q is not an affine function of $(x/z, y/z)$.

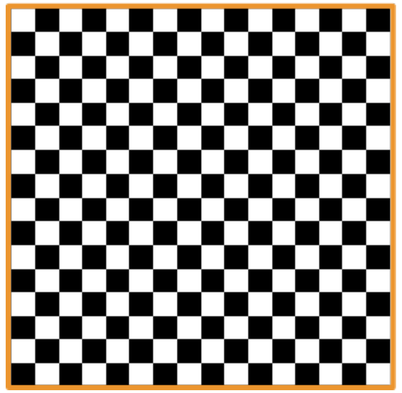
Perspective-correct interpolation

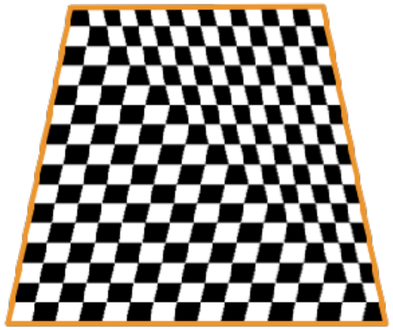
If q is an affine function of (x, y, z), then q/z is an affine function of $(x/z, y/z)$.

Algorithm:

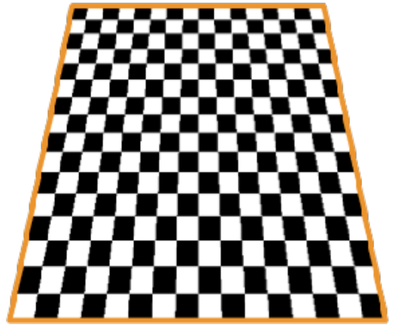
For each vertex:
- We know q_i and z_i
- Compute $1/z_i$ and q_i/z_i

For each sample:
- Interpolate $1/z$ and q/z from vertices
- Desired q is $(q/z)/(1/z)$



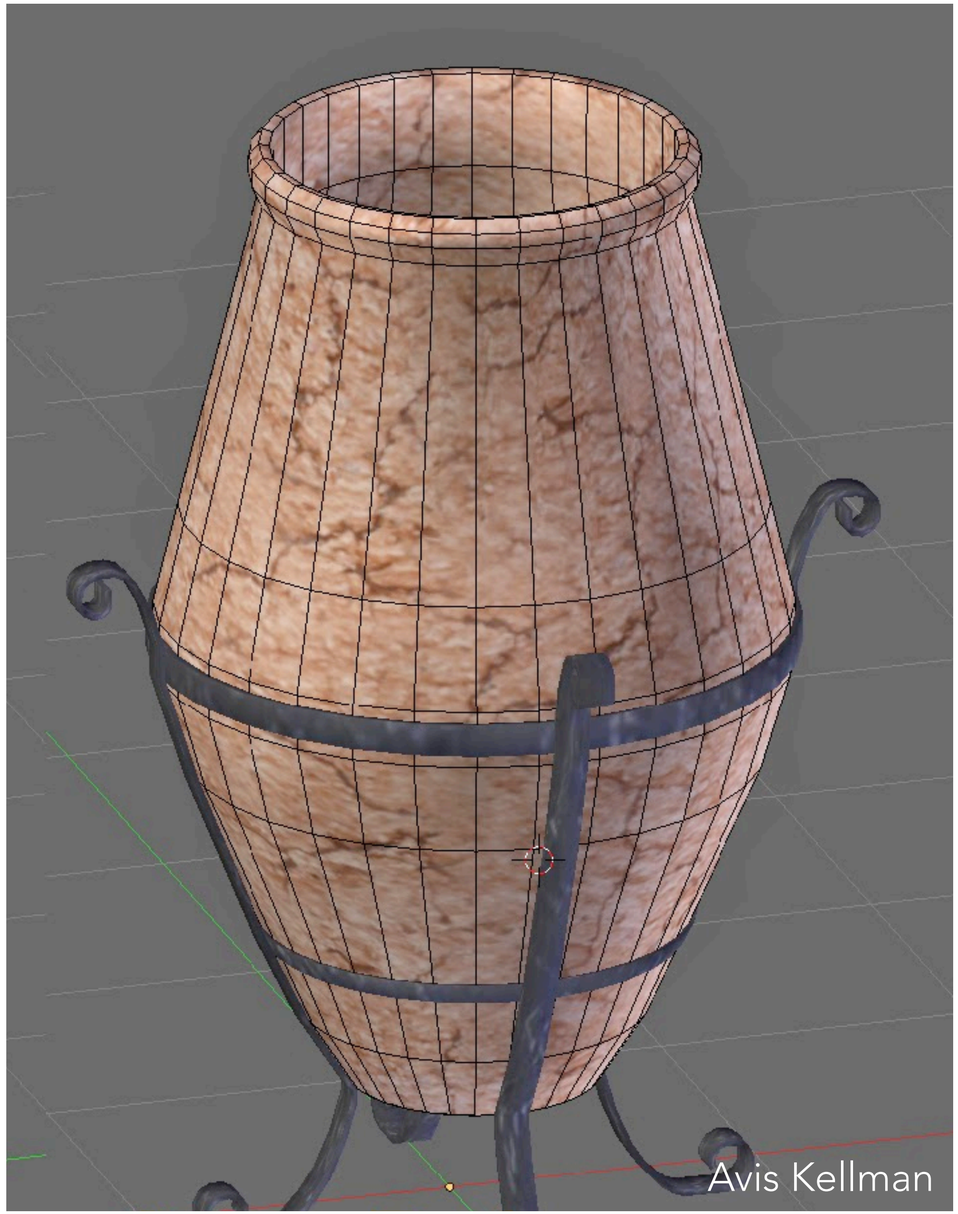


Screen space interpolation



Perspective-correct interpolation

For an elegant geometrical interpretation, see Marschner & Shirley Sec. 11.2.4



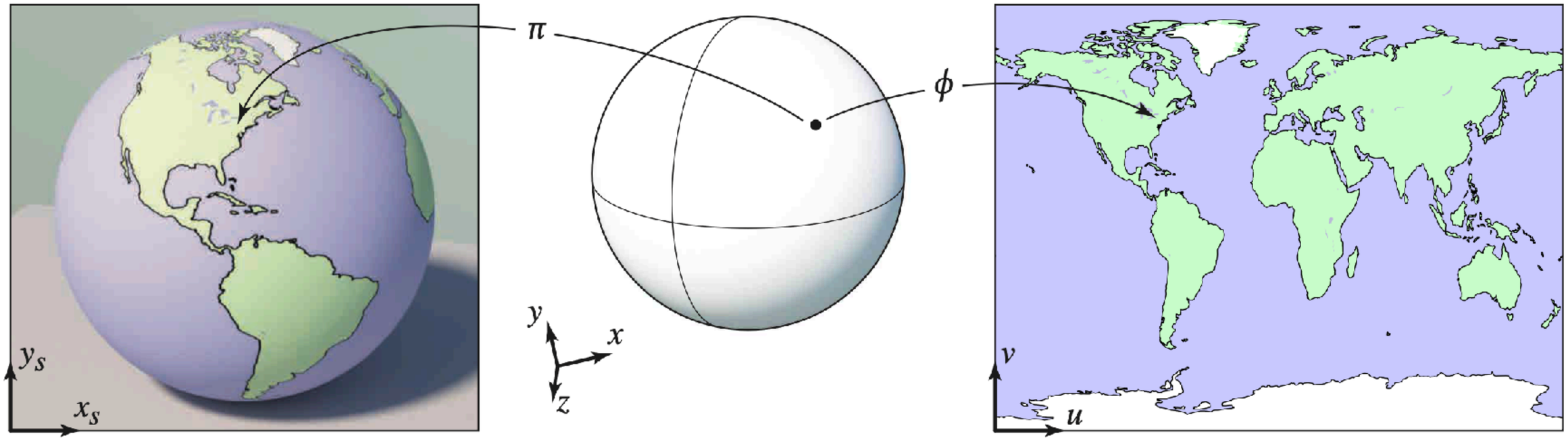
Detail (e.g. colour, normal, etc.) is some function from surface points to e.g. RGB

How to store such a function?

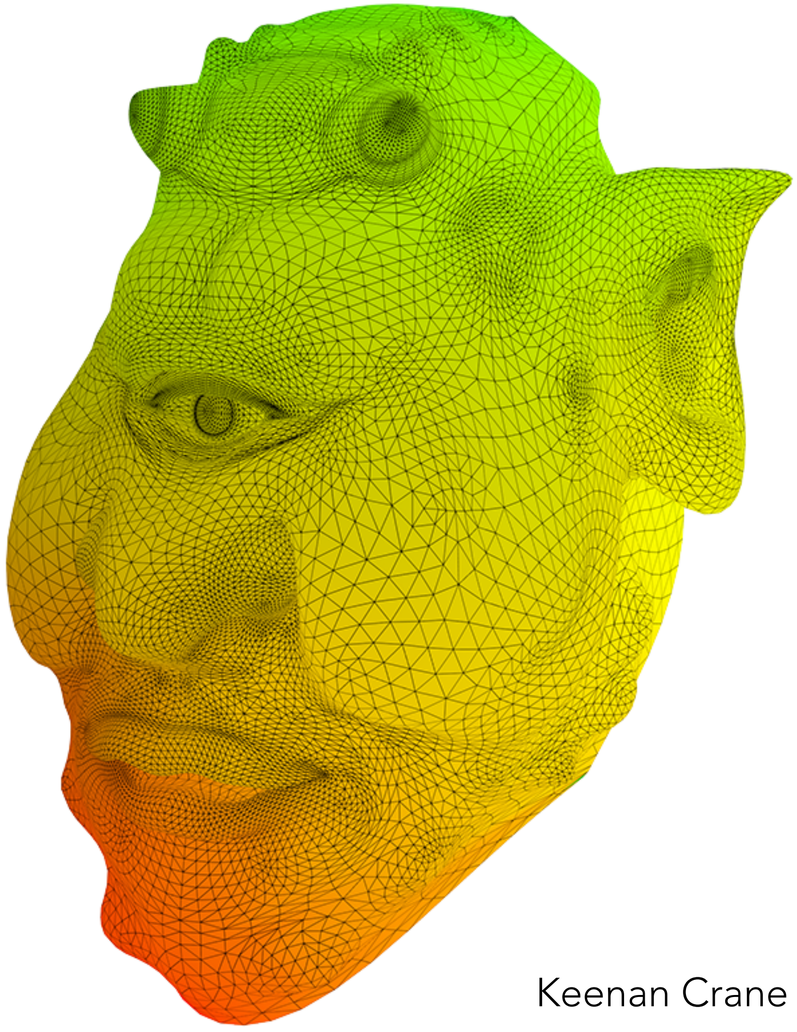
- 3D lookup table $(x,y,z) \rightarrow (r,g,b)$?
- 2D lookup table $(u,v) \rightarrow (r,g,b)$?

Then we also need to map surface points (x,y,z) to **texture coordinates** (u,v)

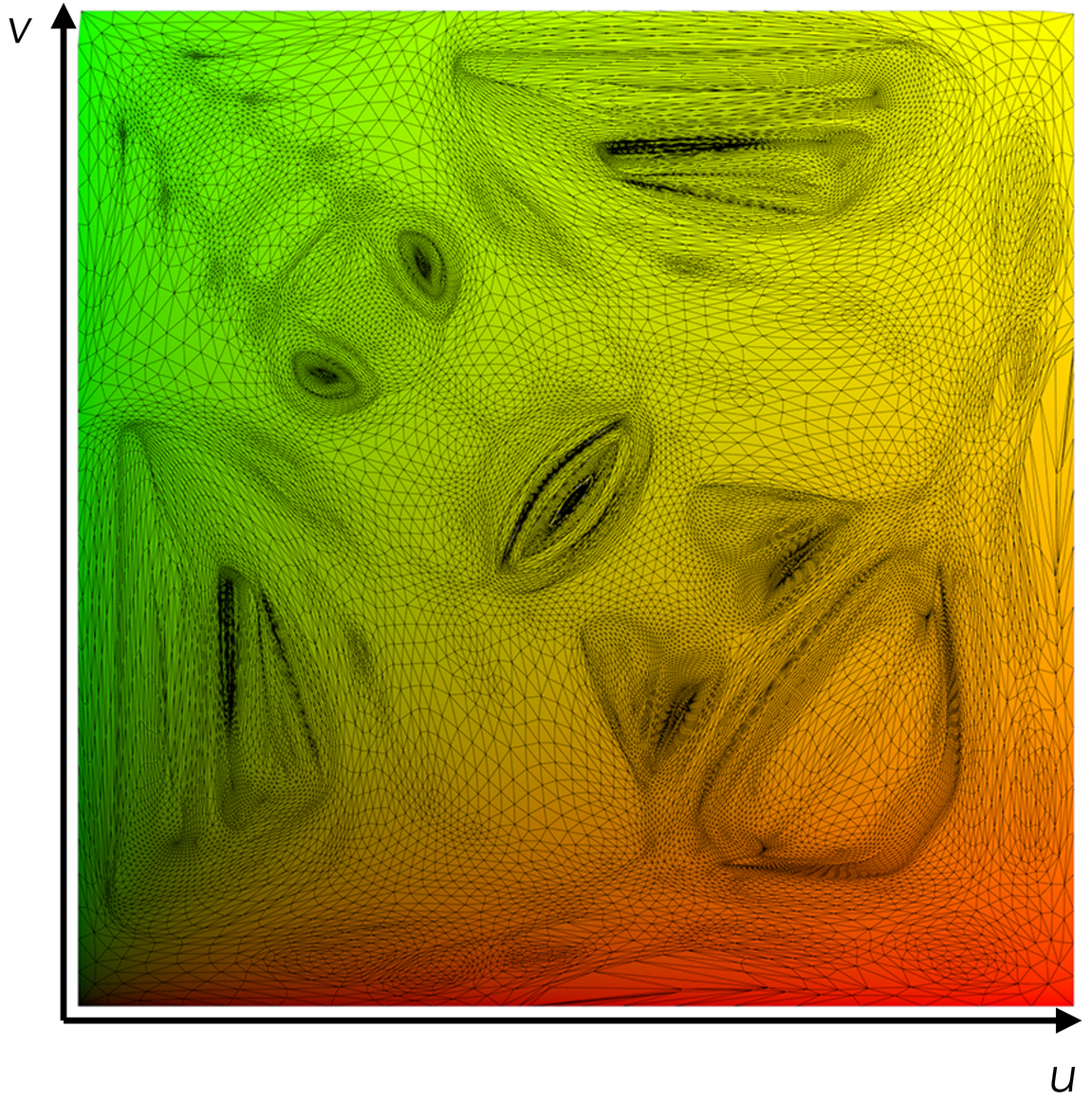
Texture mapping

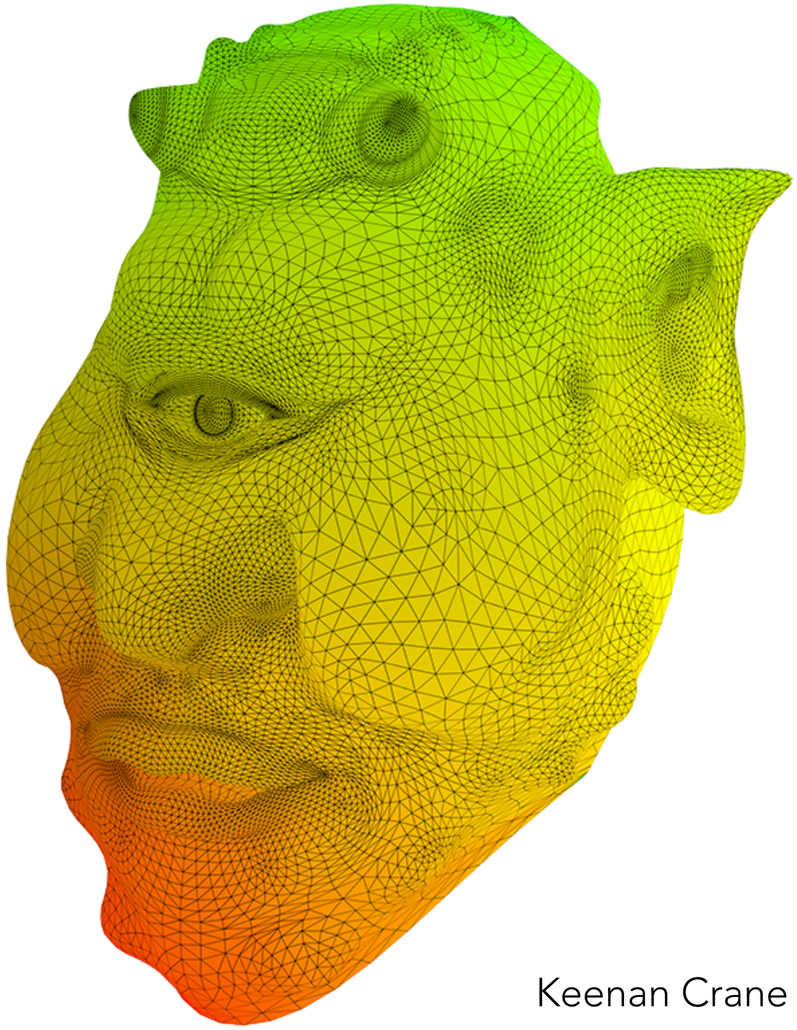


Screen space World space Texture space

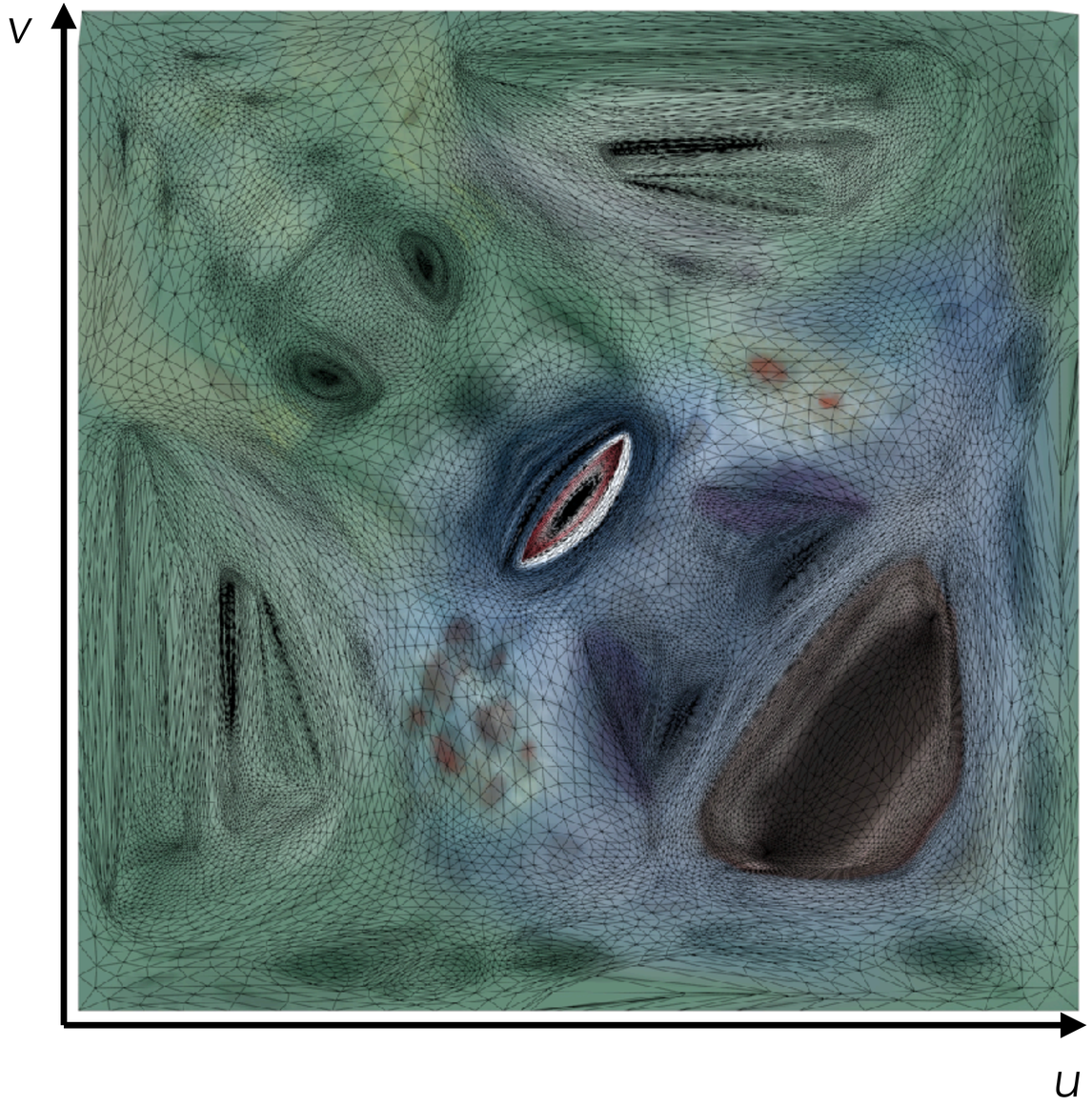


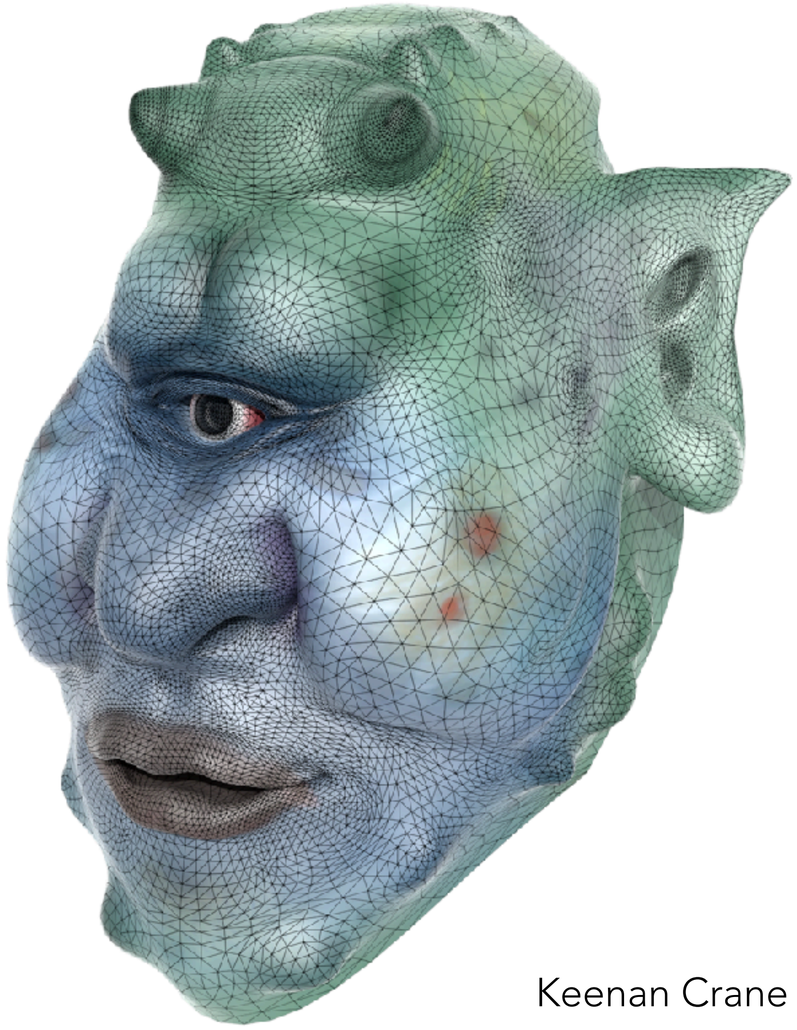
Keenan Crane



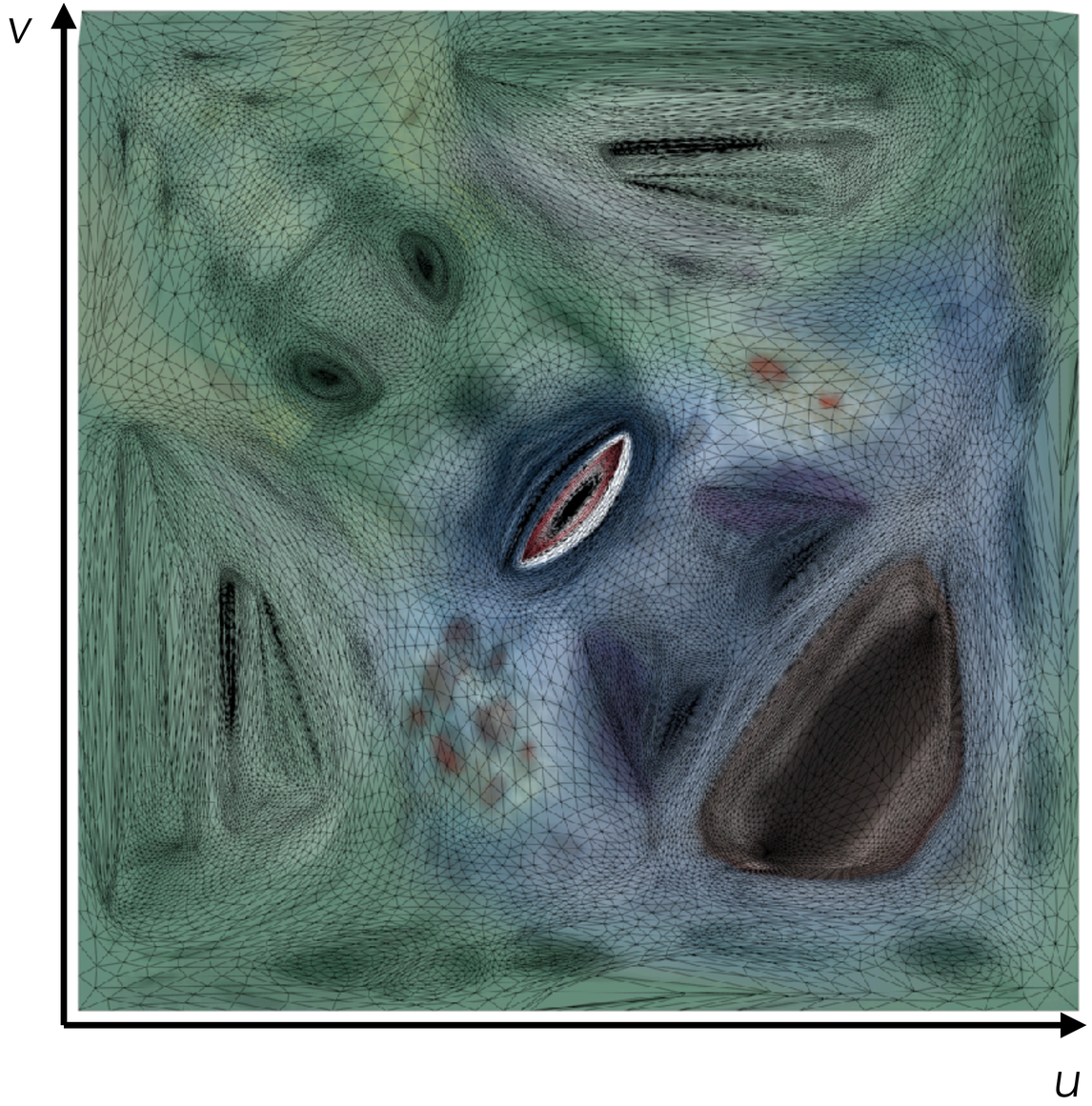


Keenan Crane

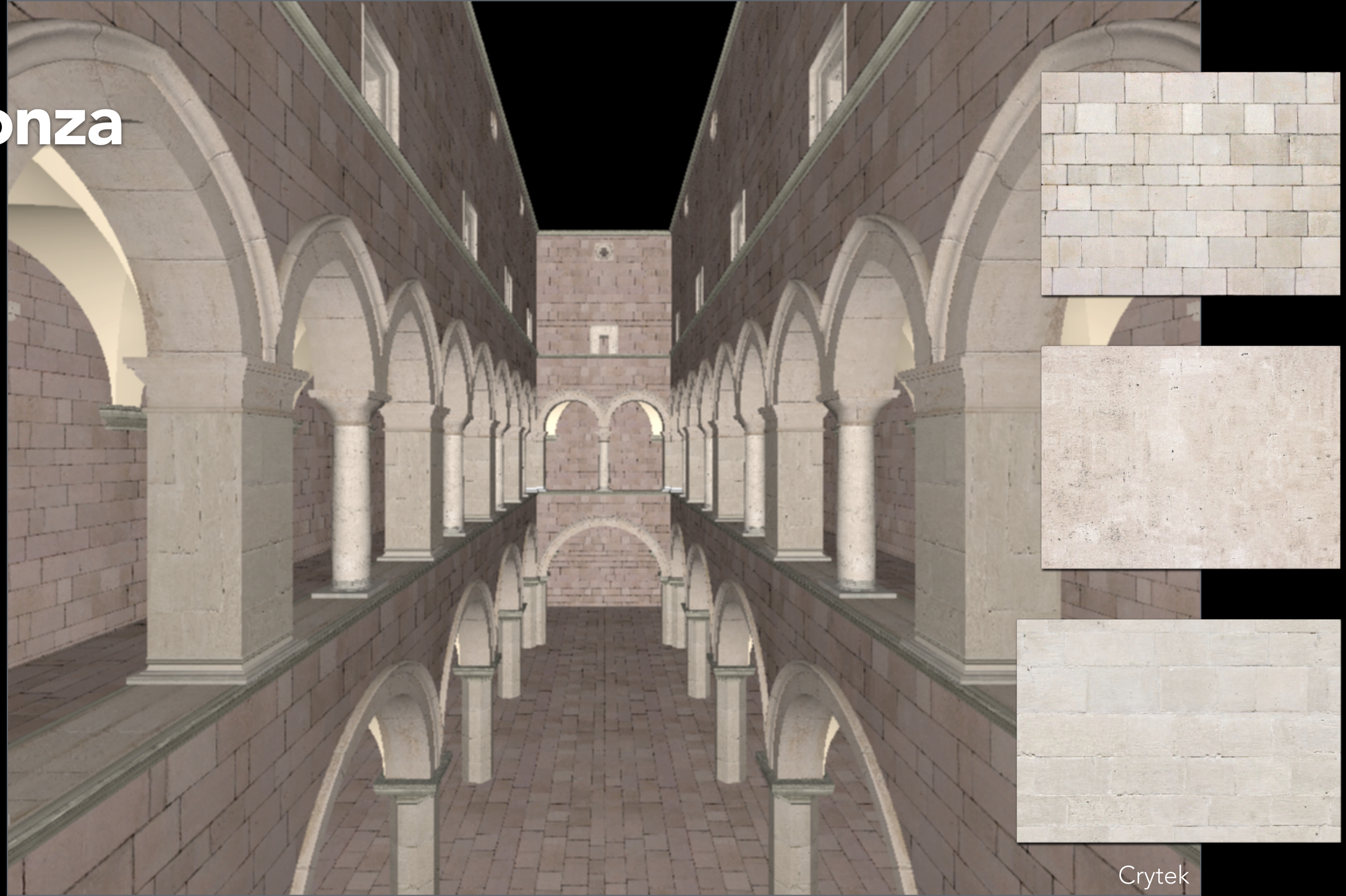




Keenan Crane

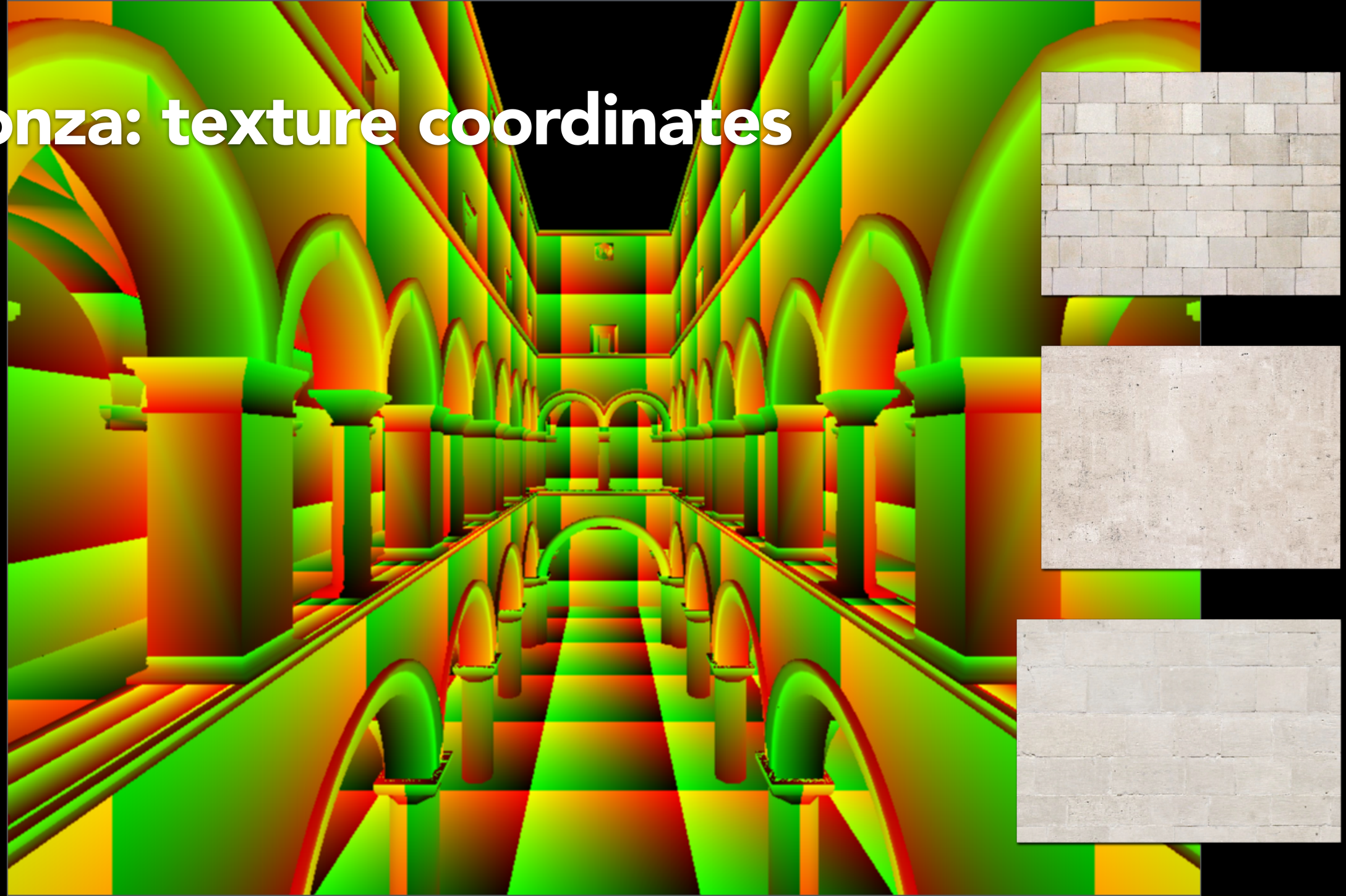


Sponza



Crytek

Sponza: texture coordinates

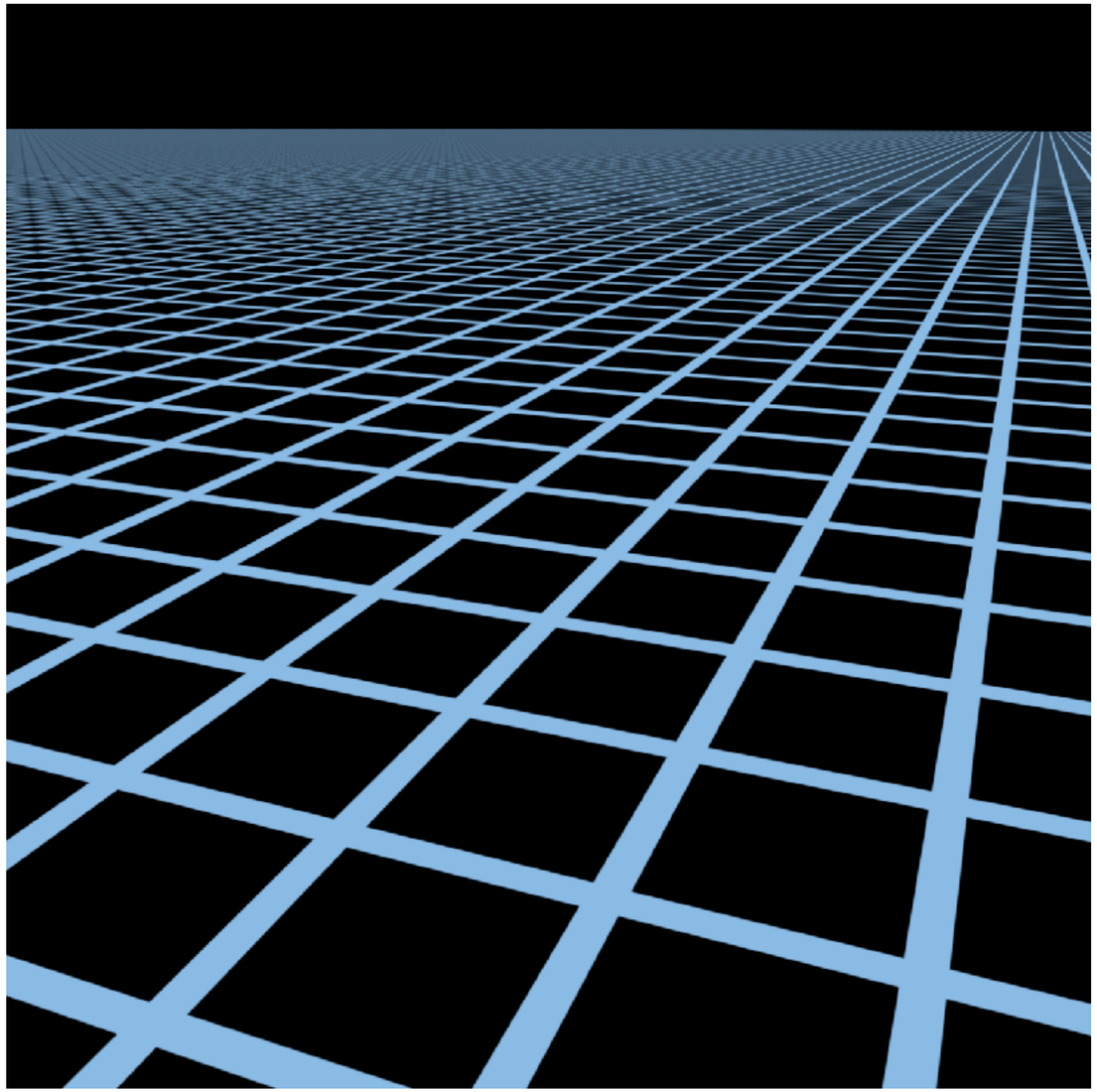


Drawing textured triangles

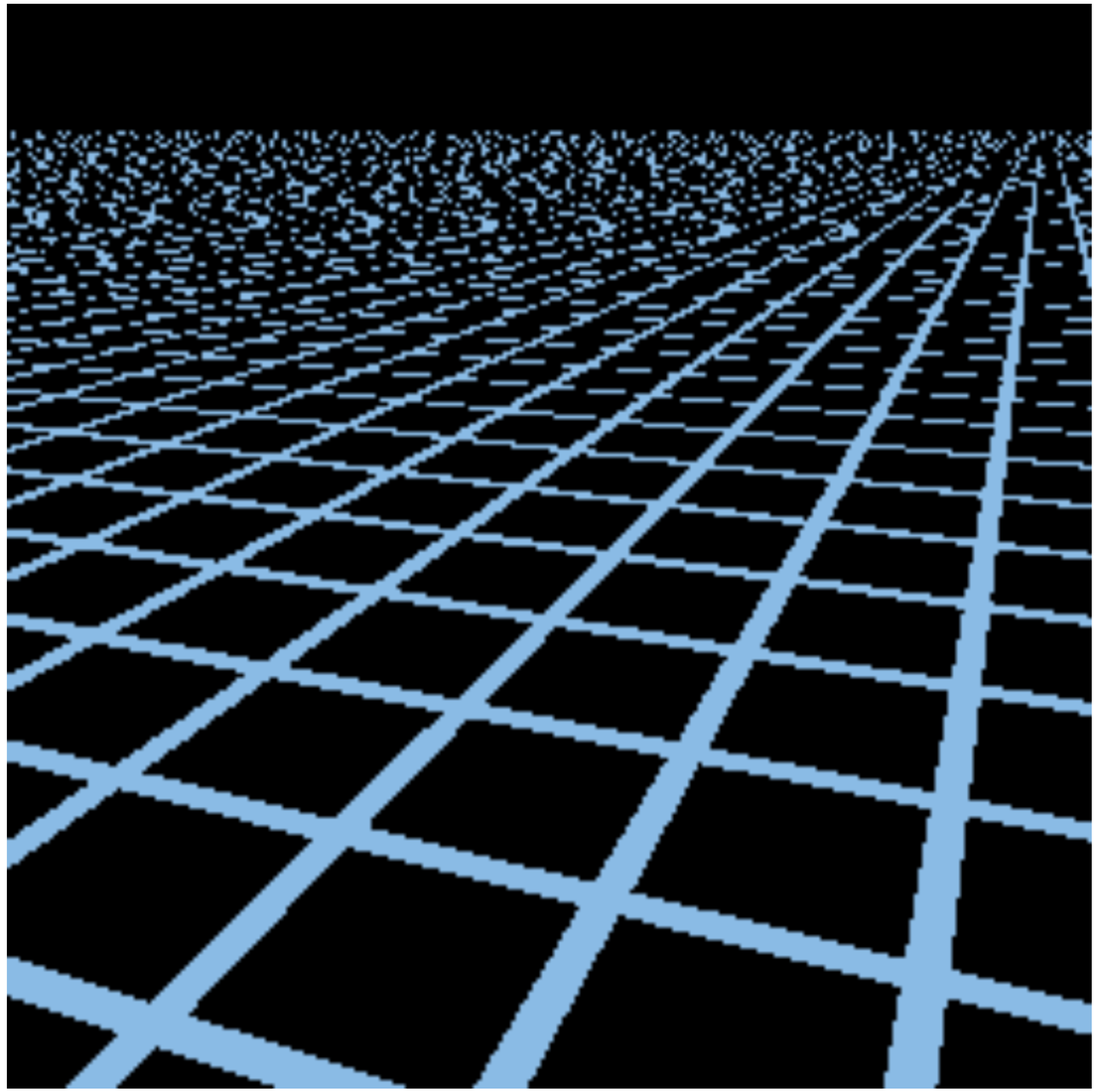
Inputs: (i) mesh with vertex positions (*x*,*y*,*z*) and texture coordinates (*u*,*v*), (ii) texture image

Naïve algorithm:

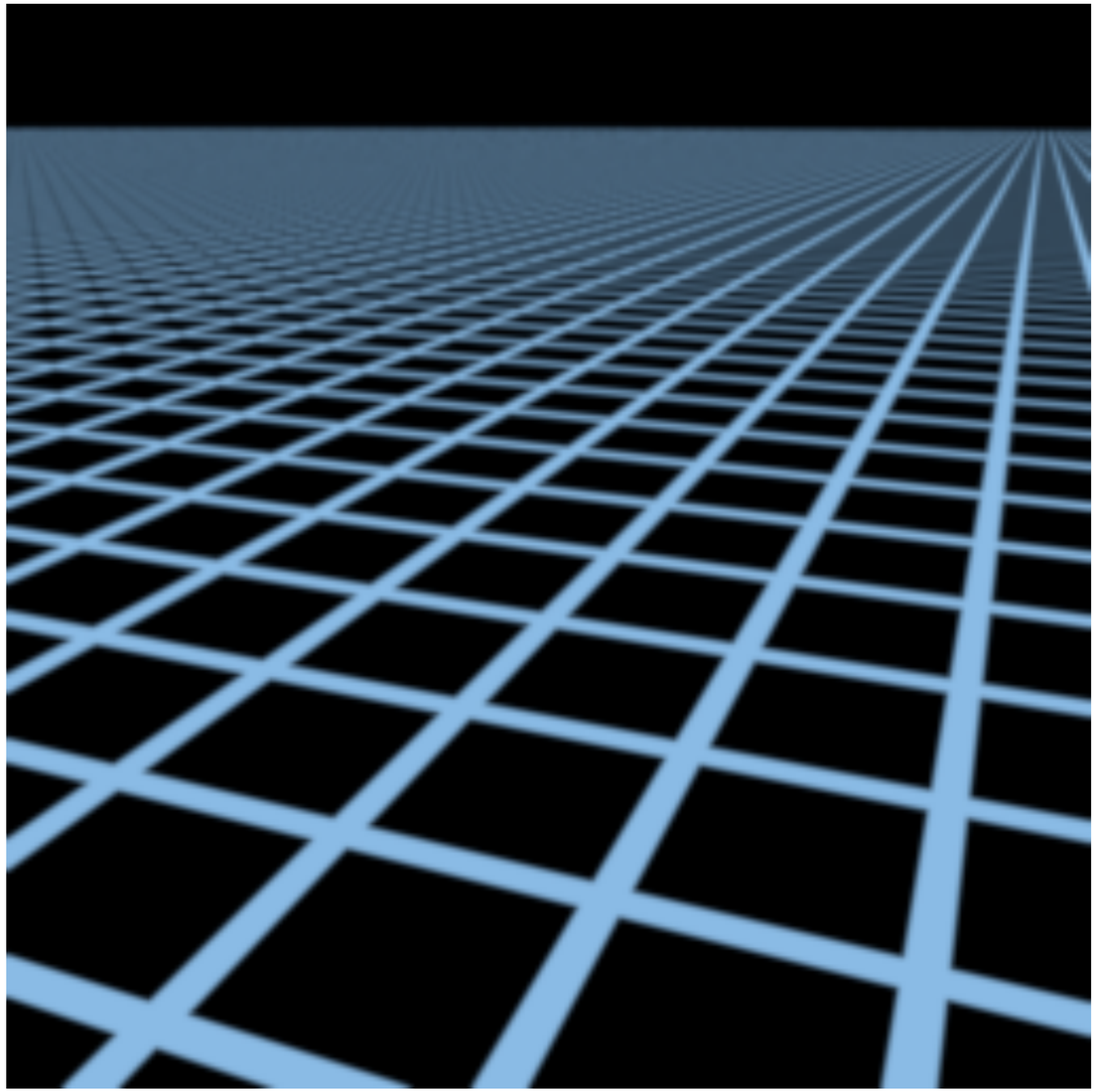
```
for each triangle (i,j,k):
    for each rasterized sample:
        (u,v) = interpolate (ui,vi), (uj,vj), (uk,vk)
        texColor = sample texture at (u,v)
        sample.color = texcolor
```



High-res reference (1280×1280)



Point sampling (256×256)



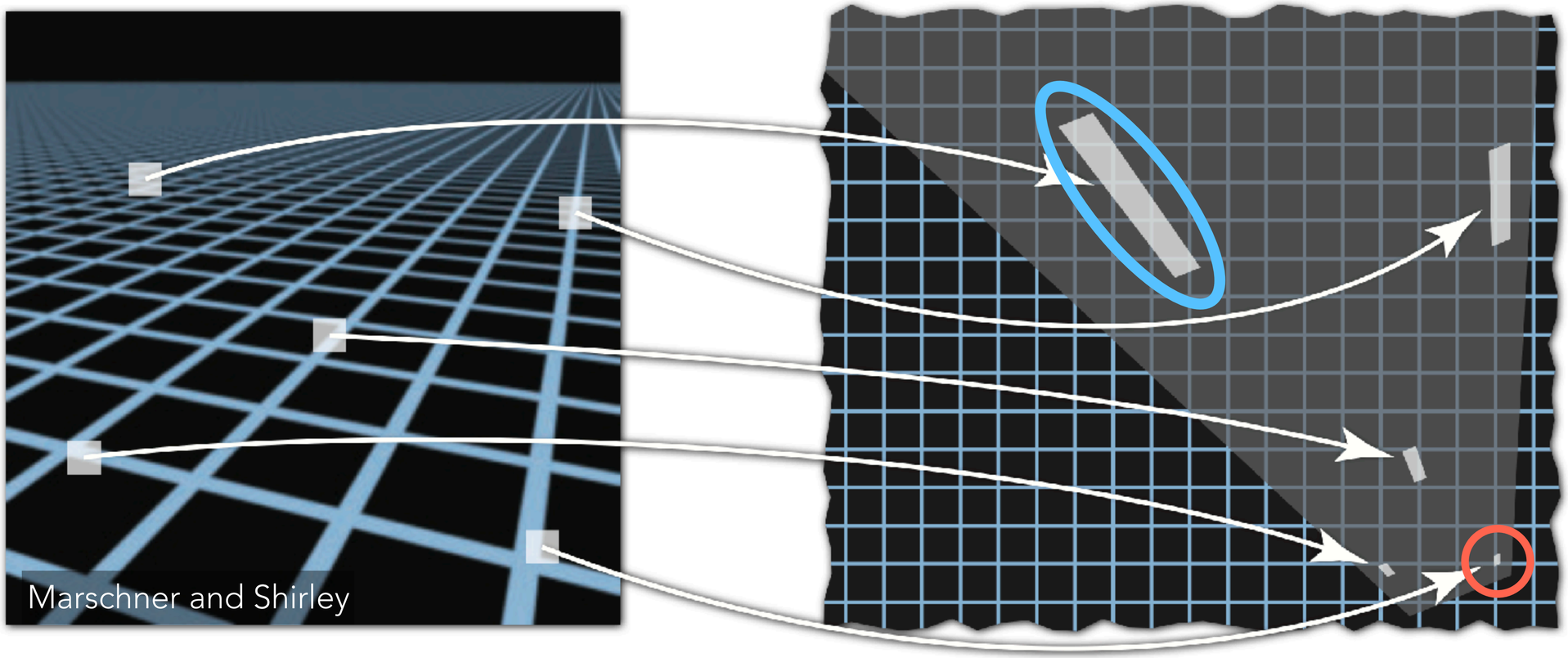
Supersampling (256×256, 512 spp)

"Easy, just do supersampling"

Sure, but:

- Higher frequencies, finer detail ⇒ need more samples to avoid aliasing
- Perspective projection creates arbitrarily high frequencies!
- Texture sampling can be expensive (memory latency)

Can we antialias textures more efficiently?



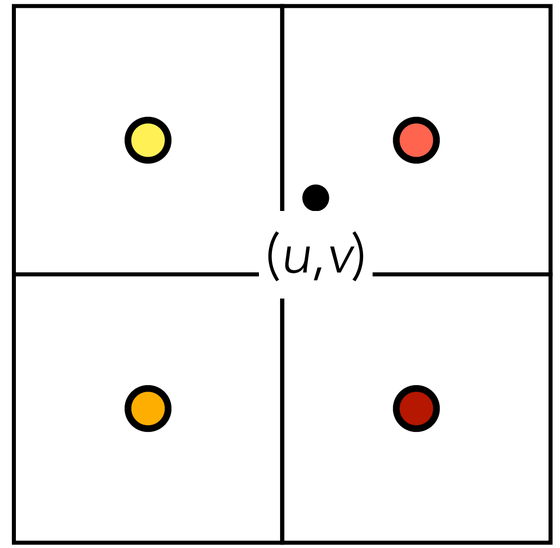
Texture mapping creates a very irregular sampling pattern!

- Some regions are **magnified**: multiple screen samples per texture pixel (texel)
- Some regions are "**minified**": multiple texels per sample

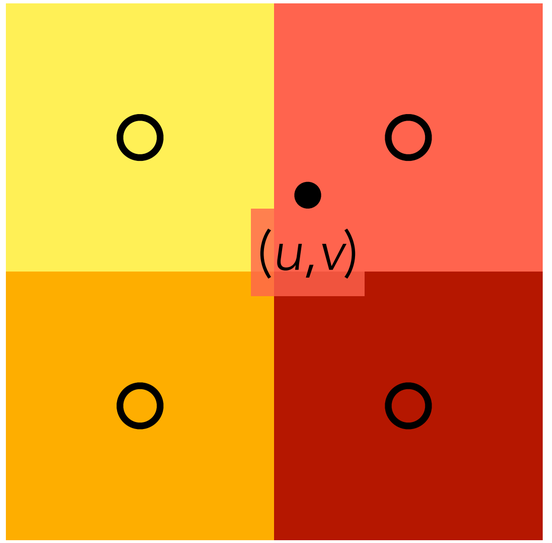
Magnification

Easy case, no aliasing. Just need to “look up” texture value at non-integer location (u,v)

Signal reconstruction ≈ interpolation



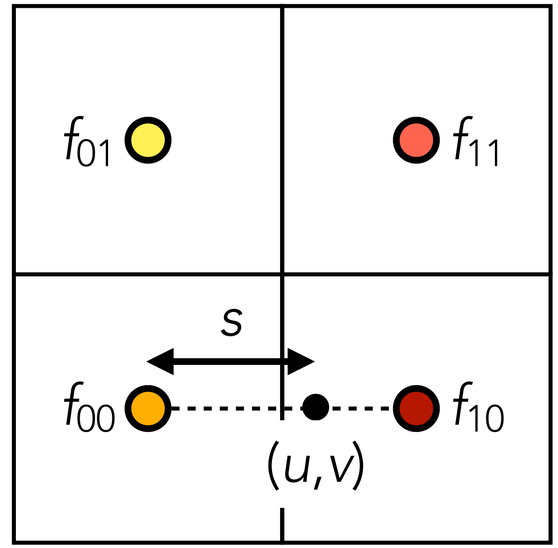
Simple and crude: nearest neighbour



Bilinear interpolation

If sample point lay exactly on a row, we could do linear interpolation:

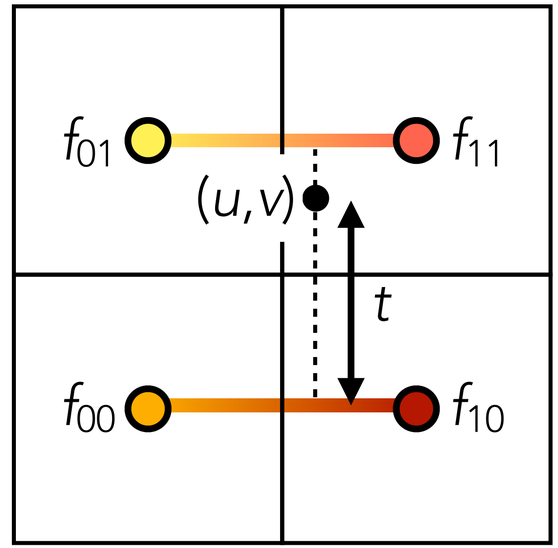
$$f(u,v) = \text{lerp}(s, f_{00}, f_{10})$$
$$= (1-s)\, f_{00} + s\, f_{10}$$



In general position:

$$f(u,v) = \text{lerp}(t, \text{lerp}(s, f_{00}, f_{10}), \text{lerp}(s, f_{01}, f_{11}))$$

$$= (1-s)(1-t)\, f_{00} + s\,(1-t)\, f_{10} + (1-s)\, t\, f_{01} + s\, t\, f_{01}$$



Homework

Check out Assignment 1!