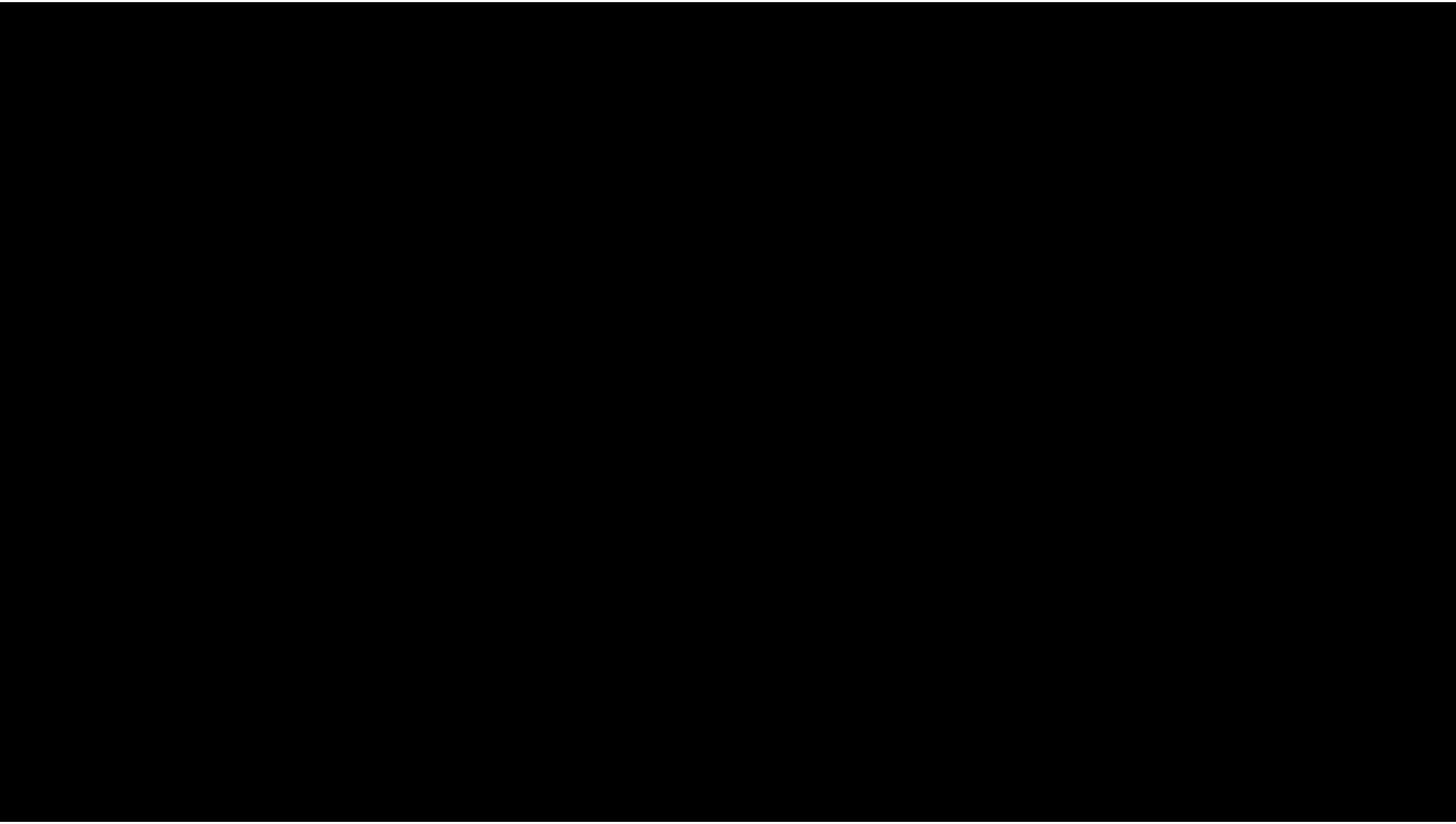
Texture Mapping

Prof. Vladlen Koltun
Computer Science Department
Stanford University

How do we simulate surface detail?

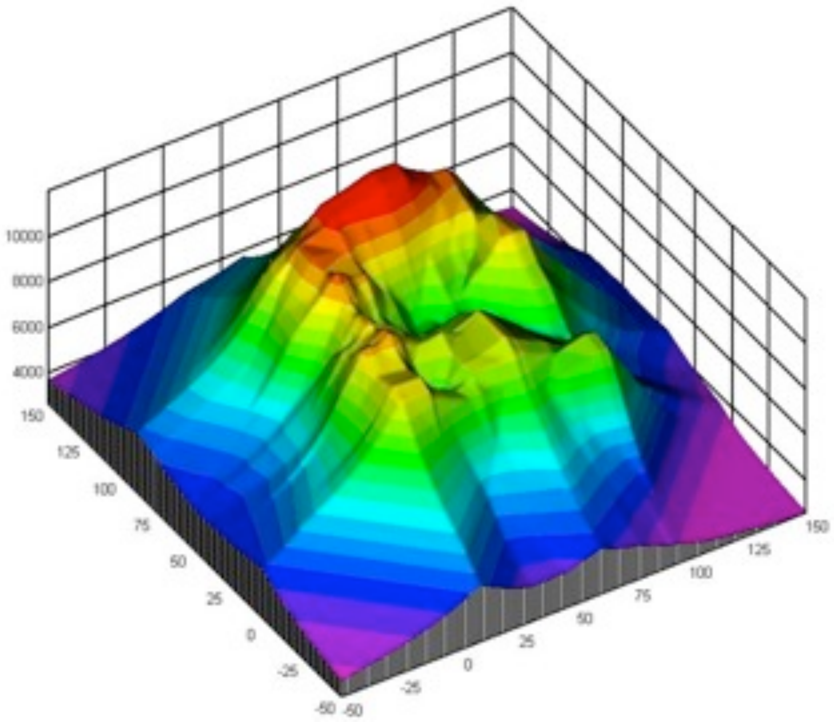
- Store functions in memory buffers and map them onto surfaces
- Functions can be generated through simulation or acquisition from the real world

Enhanced realism

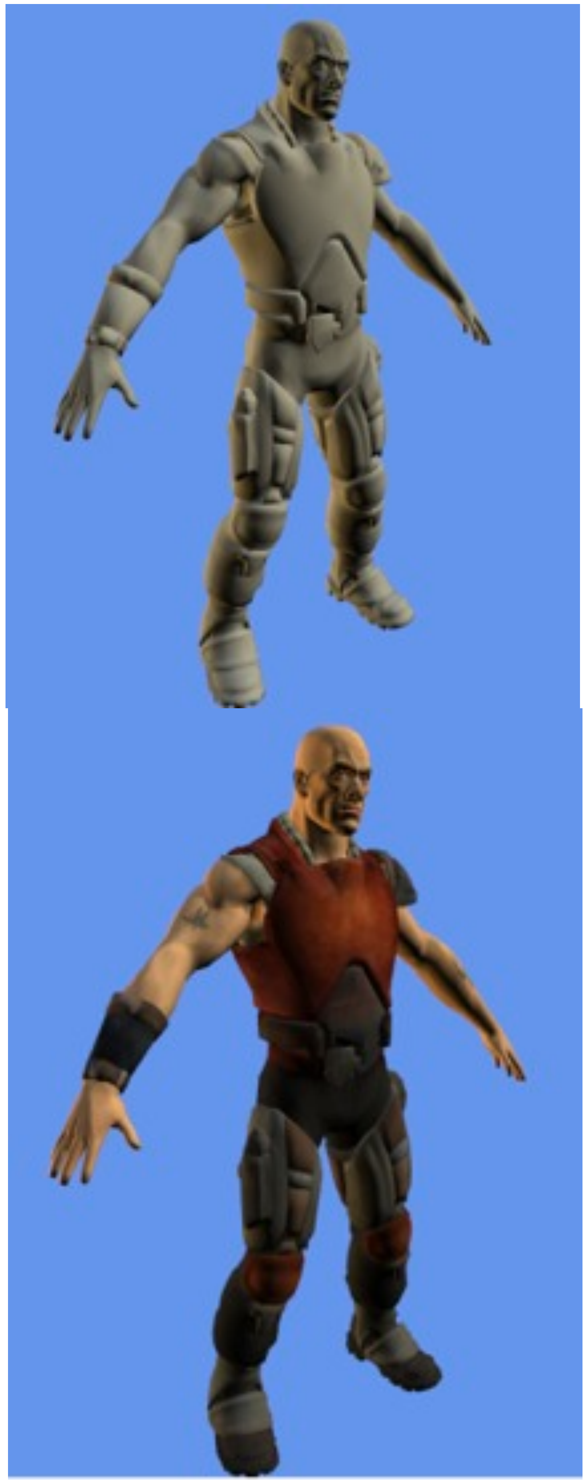


Alex Roman, “The Third and The Seventh”

Not just bitmaps



one-dimensional

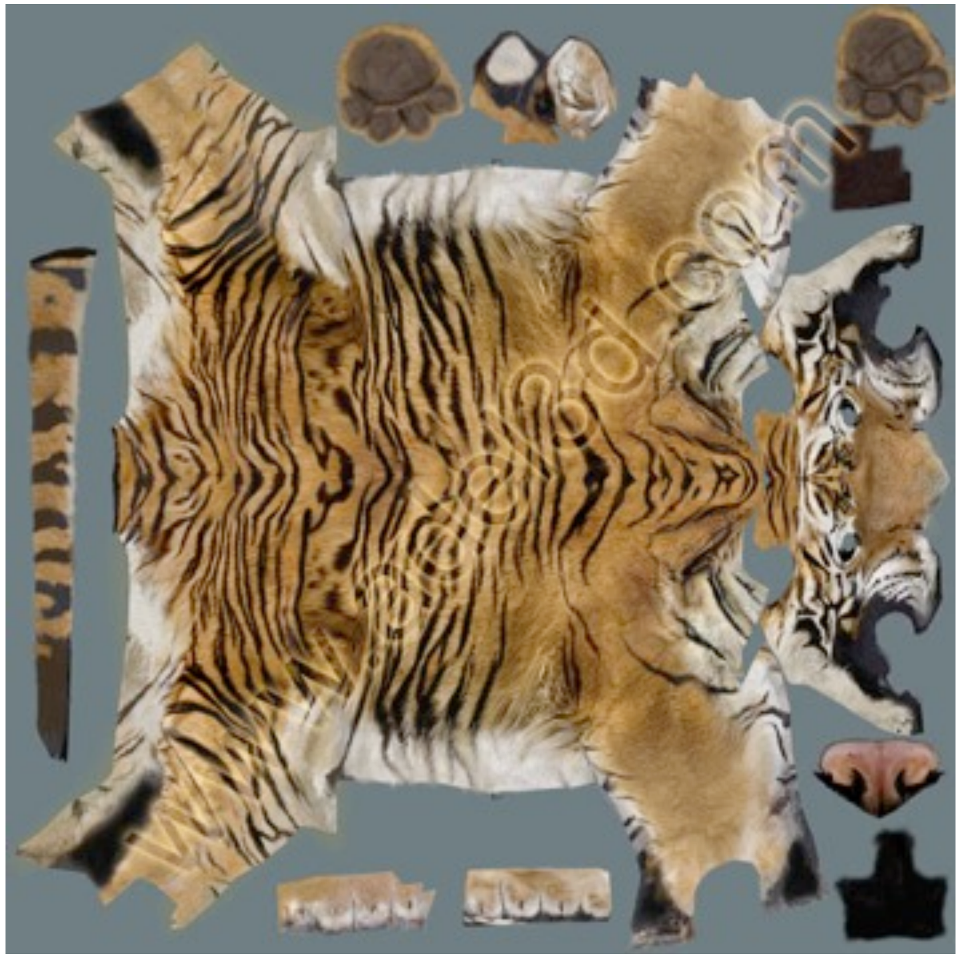


two-dimensional

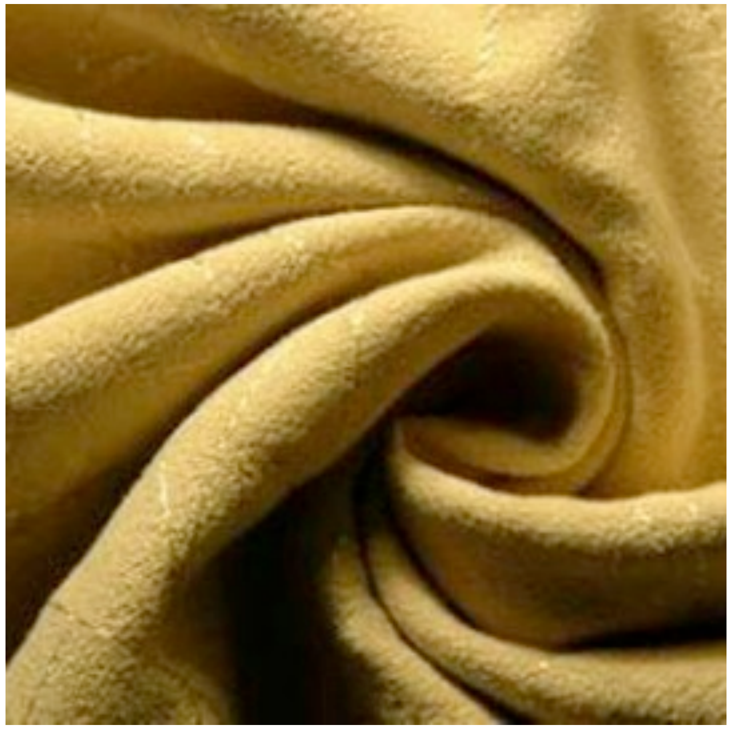
three-dimensional

Shannonbowling, Shawn Hargreaves, Dong et al. 2008

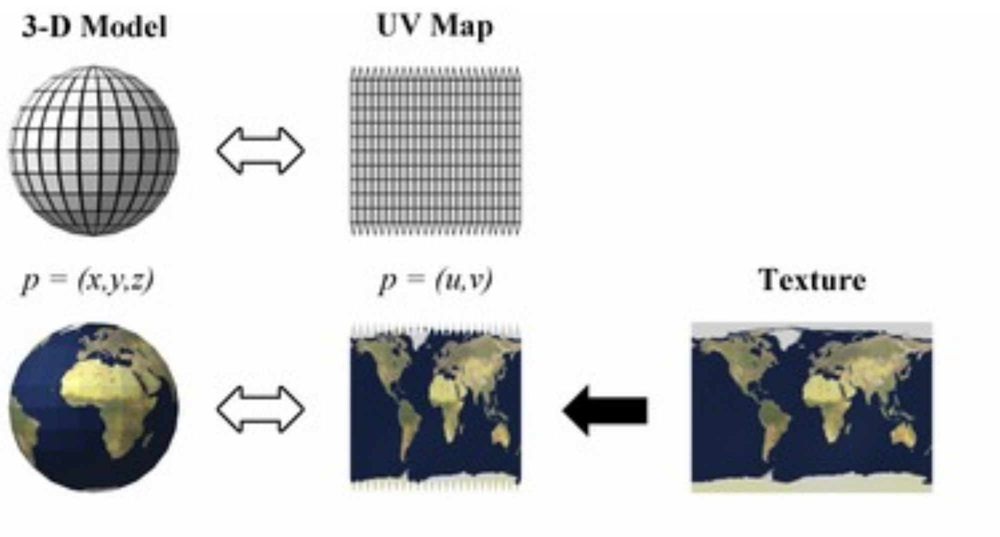
Not really texture



≠



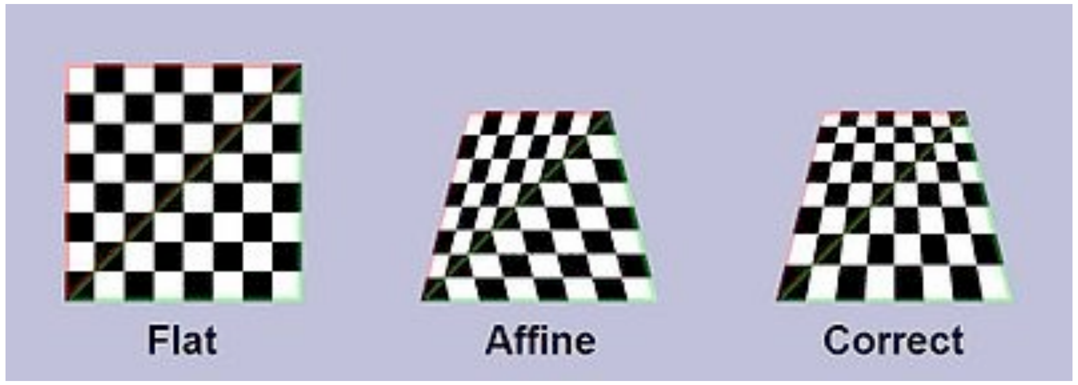
UV mapping

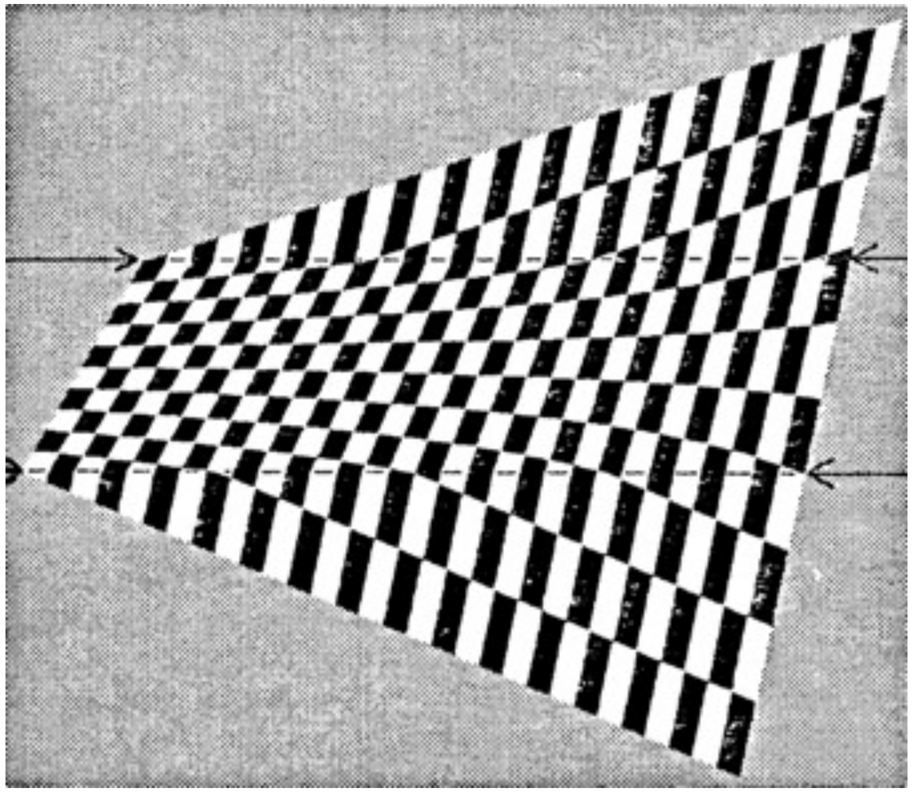


- Associate texture coordinates (u,v) with vertices
- Interpolate within polygons
- Use texture values to determine reflection coefficients

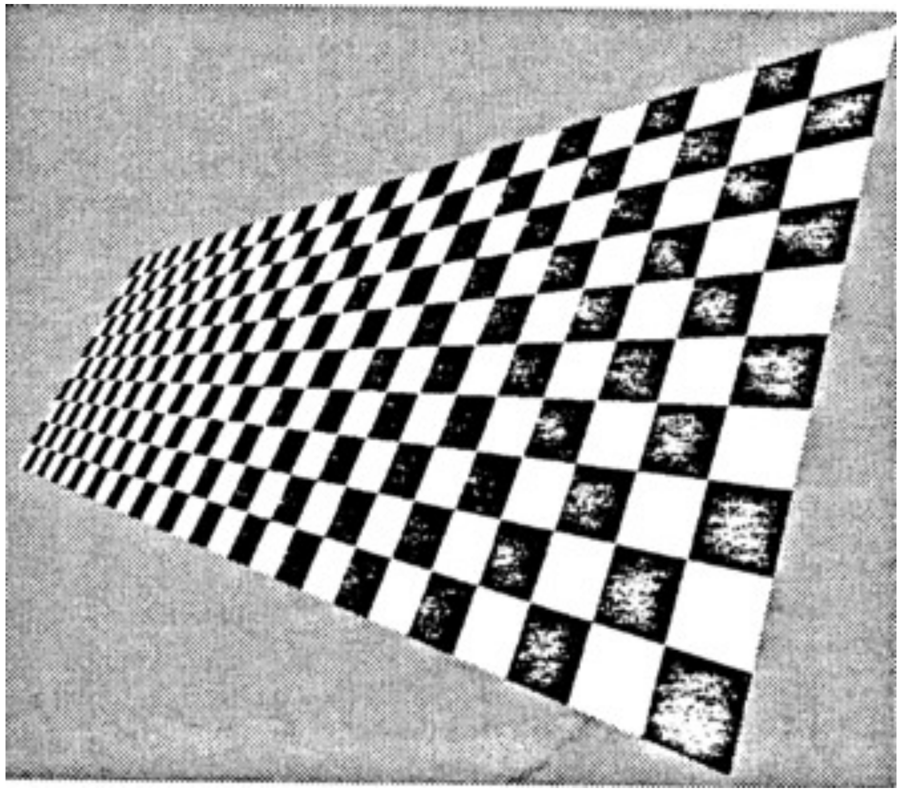
Tschmits (Wikimedia Commons)

Perspective-correct interpolation essential





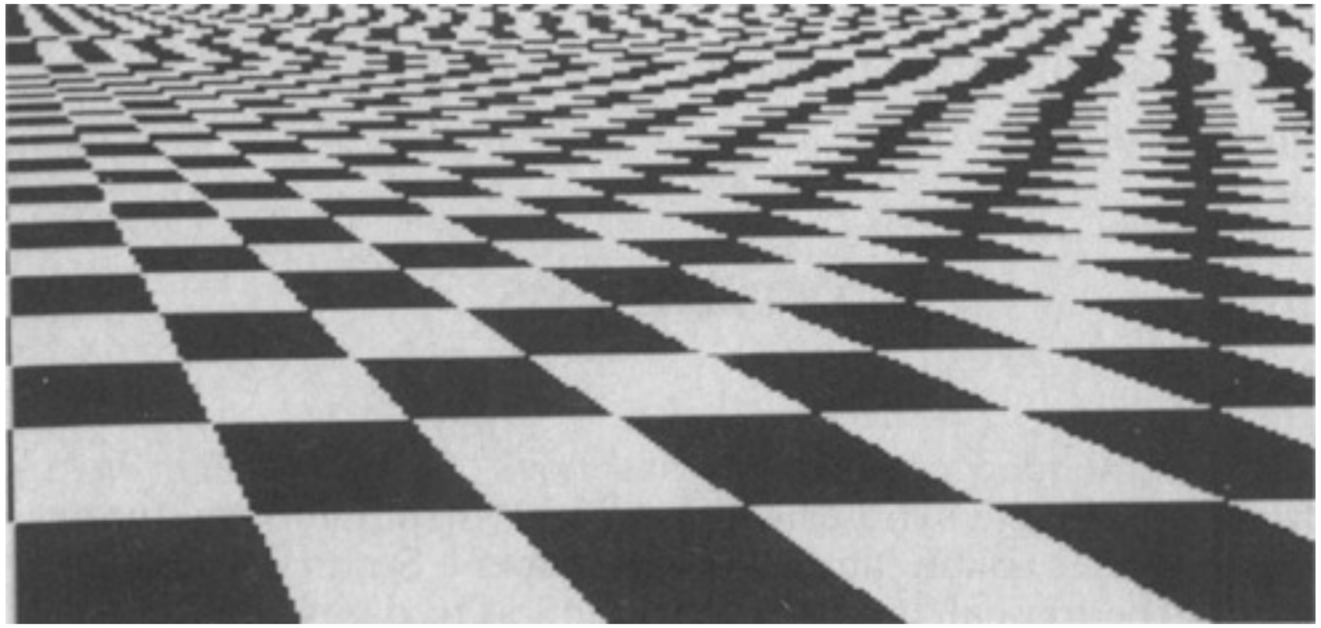
linear



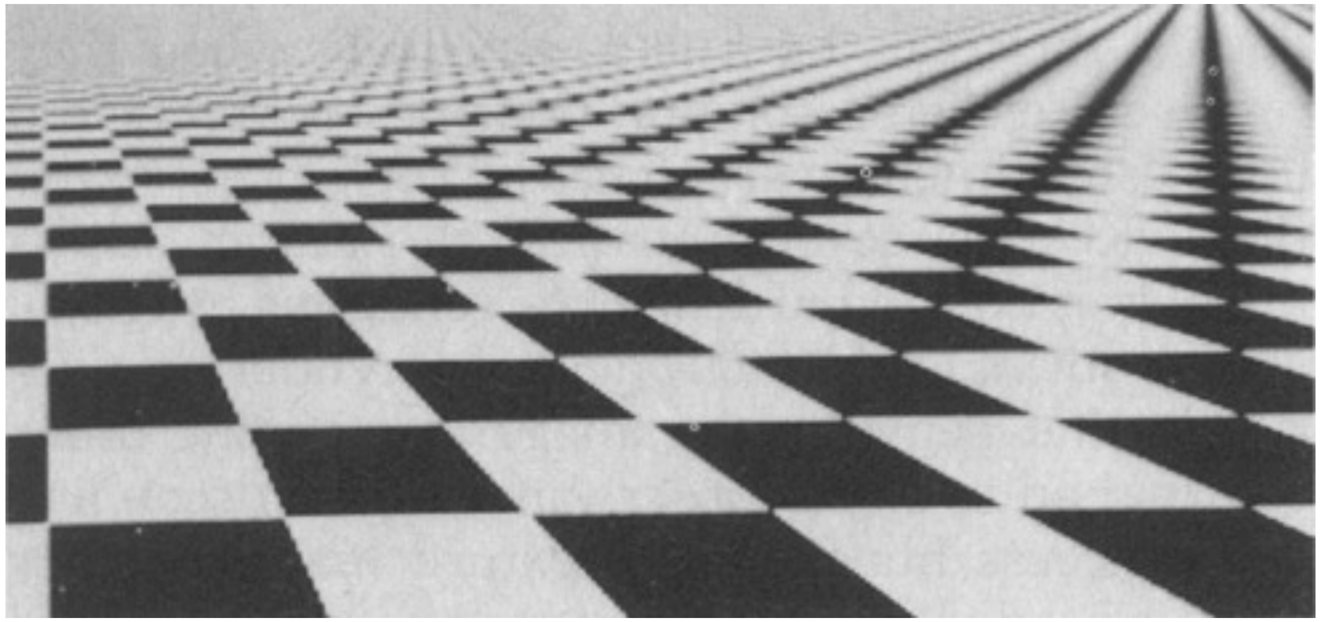
perspective-correct

Rainwarrior (Wikimedia Commons), Paul Heckbert

Aliasing



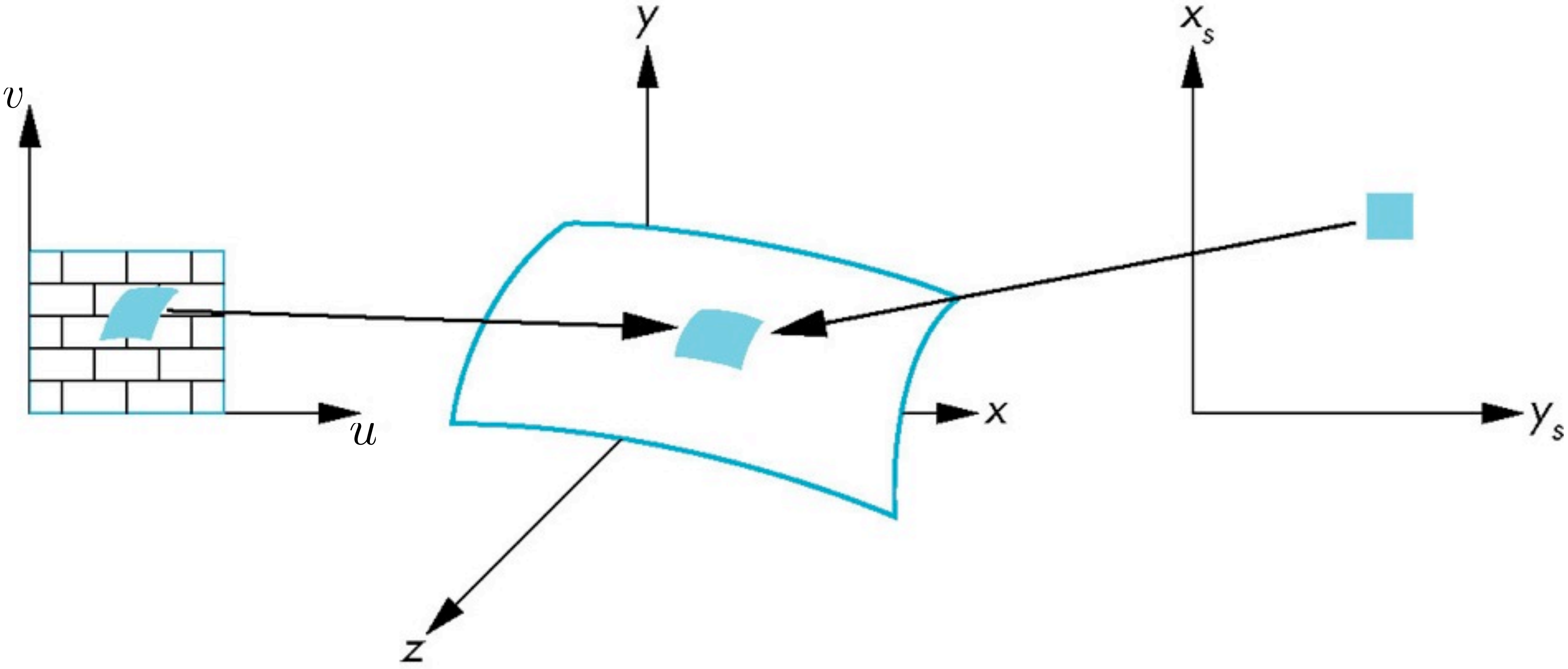
unfiltered



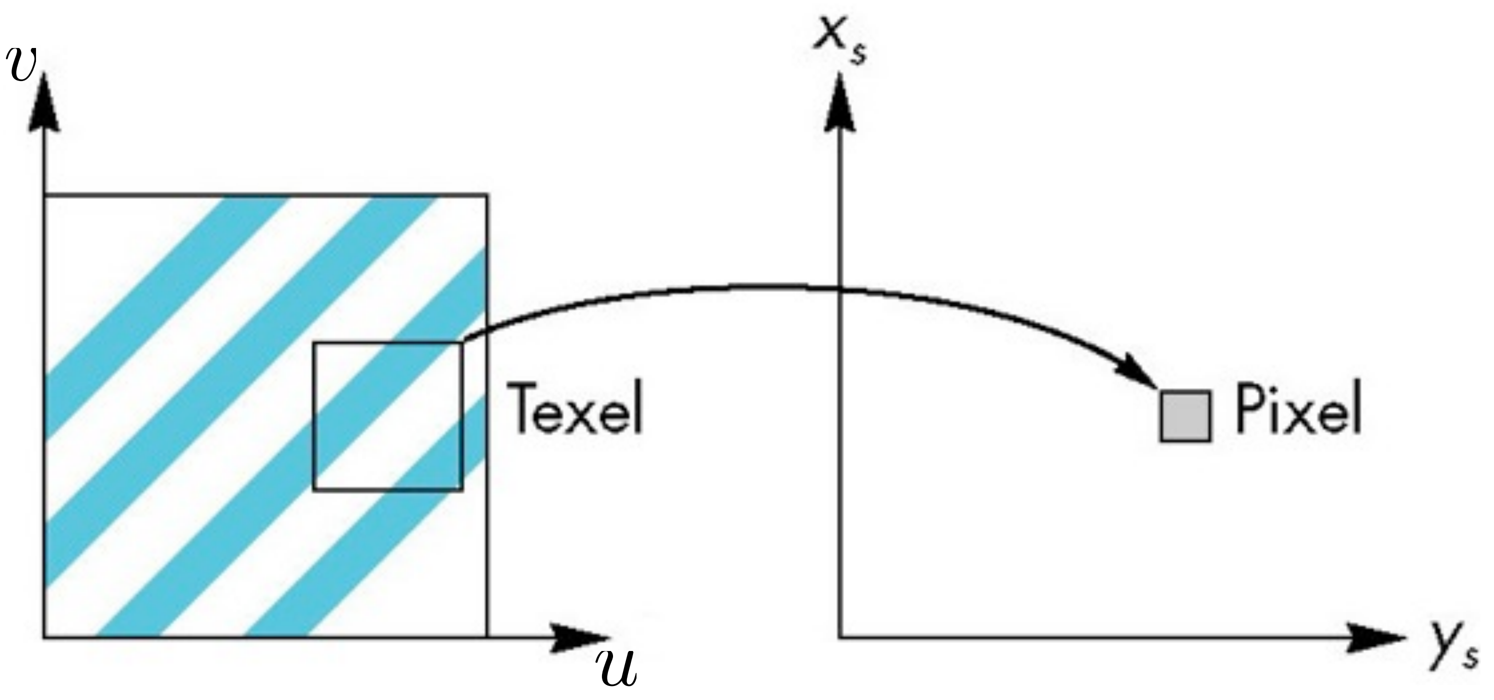
filtered

Paul Heckbert

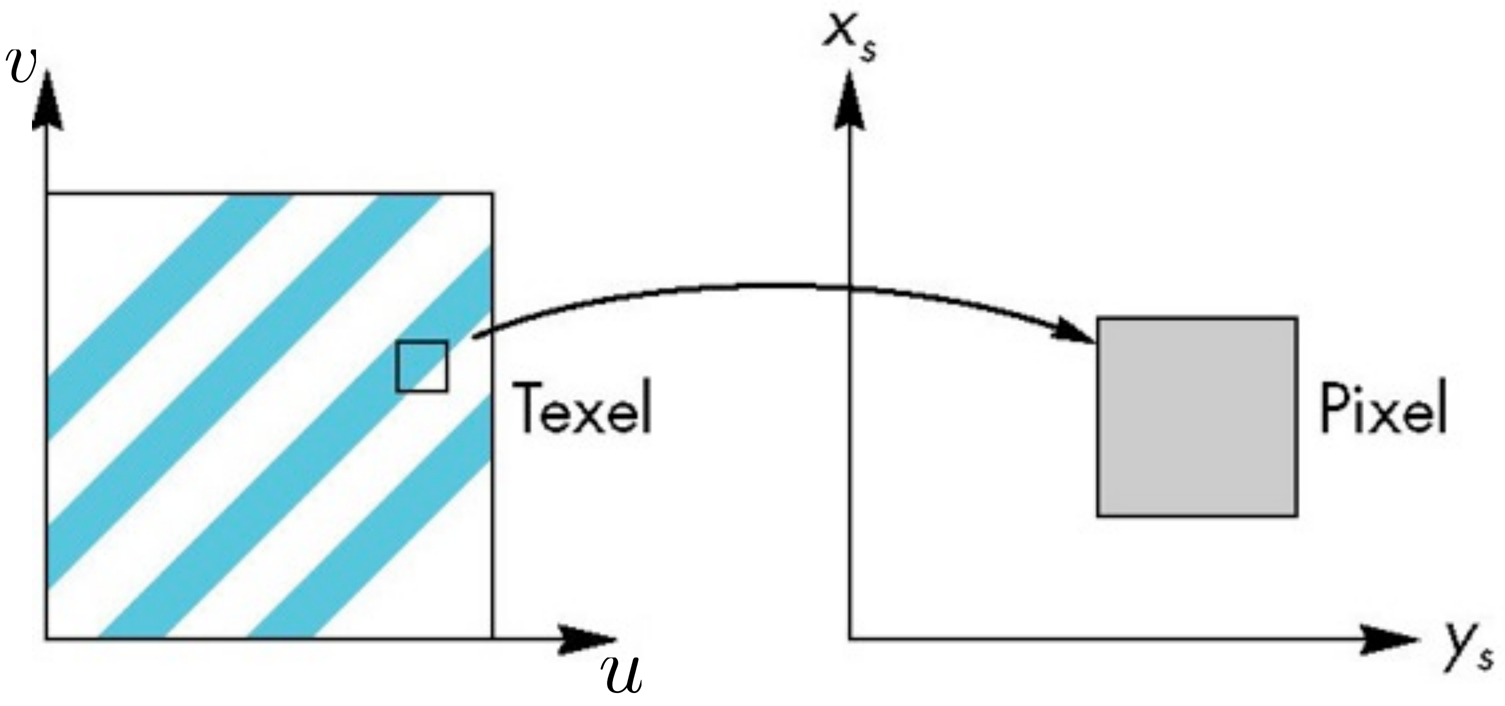
Texture filtering



Minification and magnification

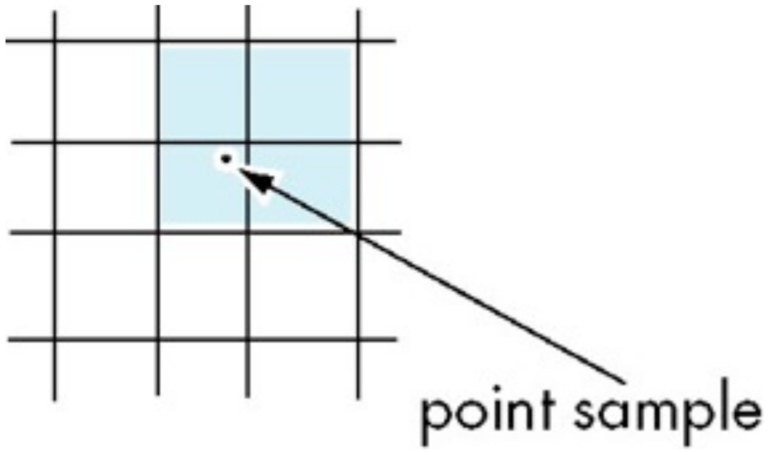


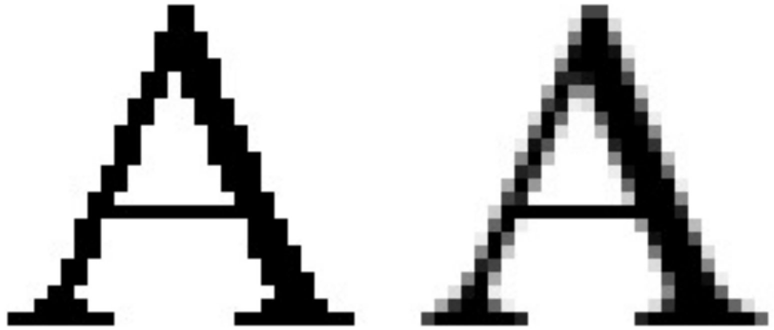
minification



magnification

Bilinear filtering

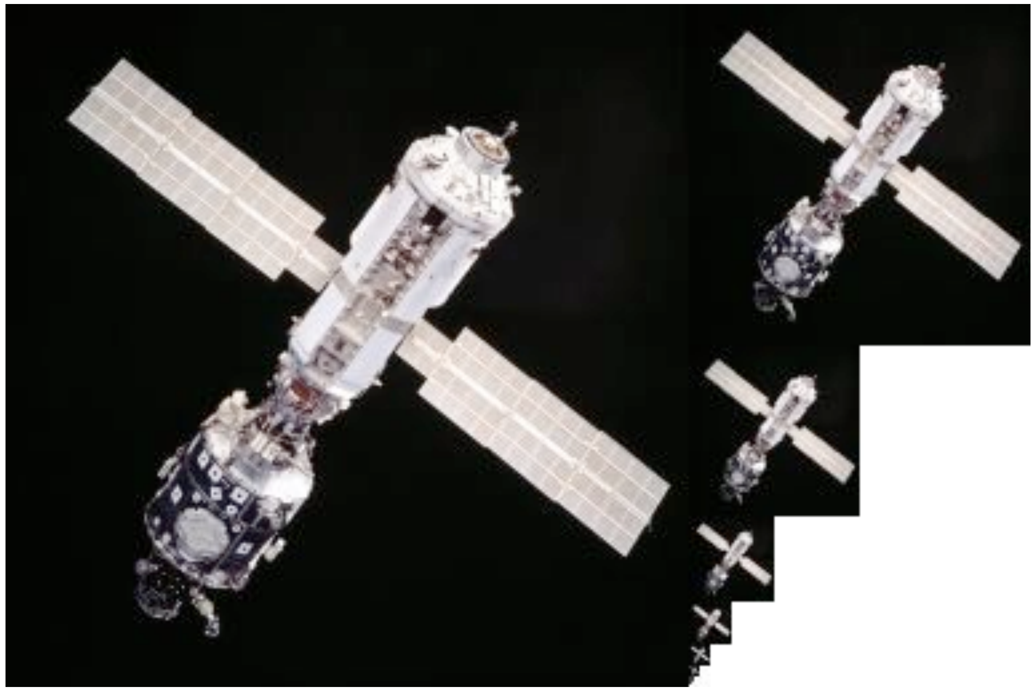




- Instead of rounding interpolated (u,v) coordinates to integer values, blend the four adjacent texels. Weigh contributions based on distance.
- Aimed to address aliasing artifacts due to magnification

Mwyann (Wikimedia Commons)

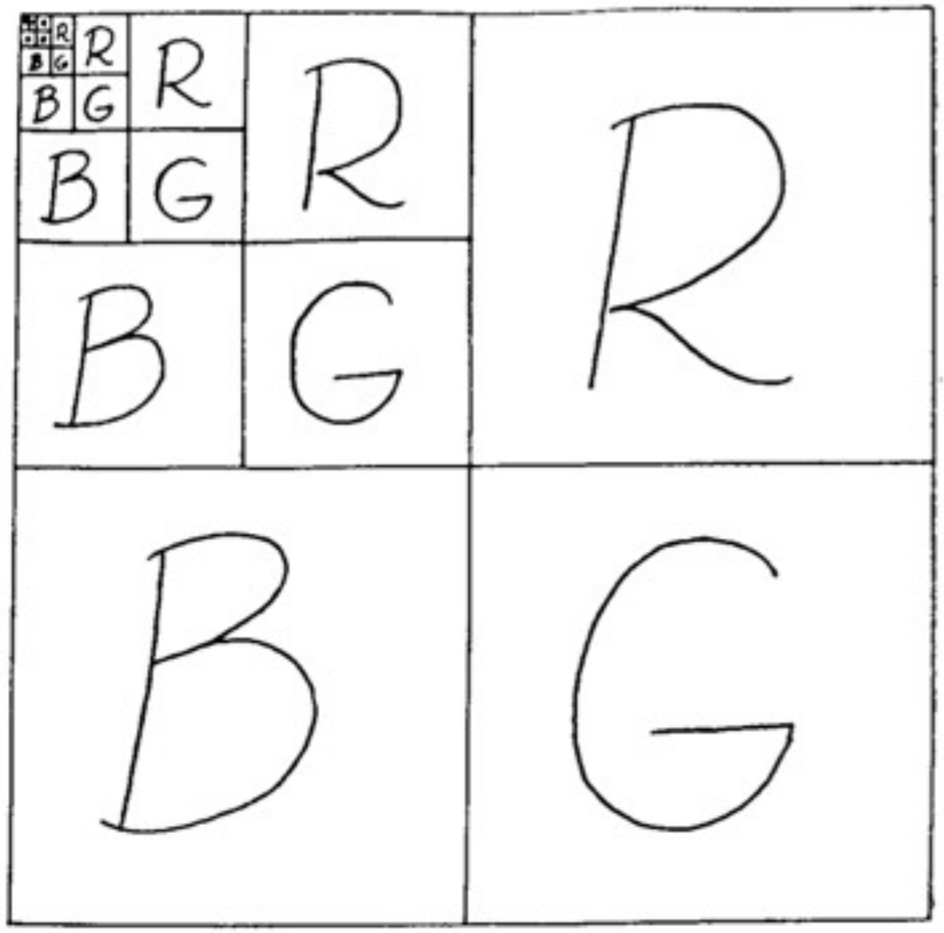
Mipmapping

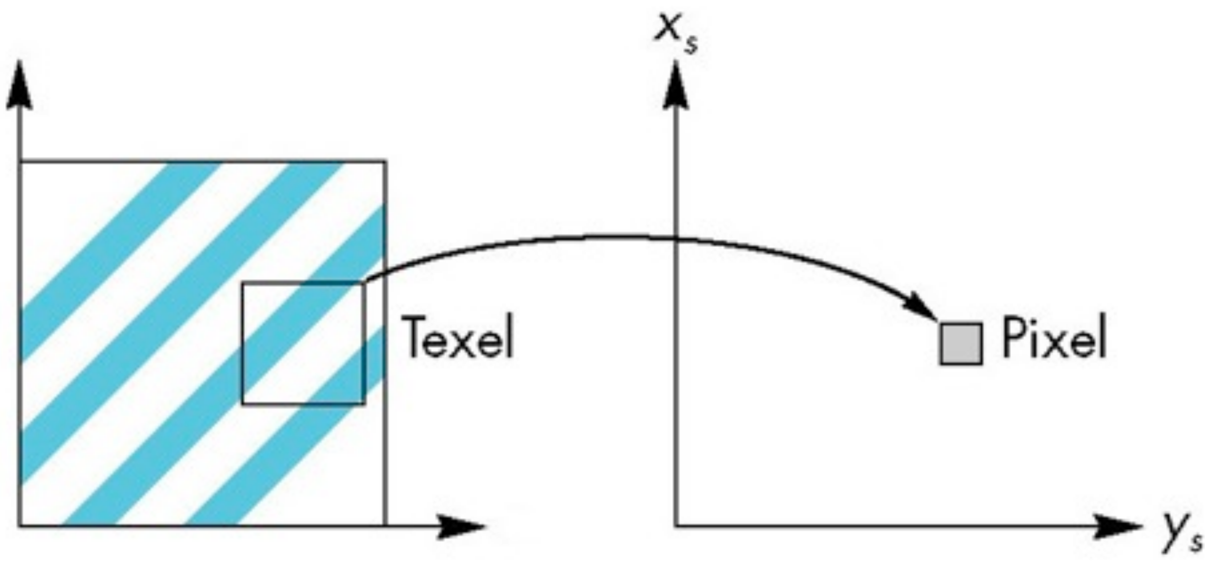


- Pre-filter the texture map at multiple resolutions, store the hierarchy
- “mip” stands for “multum in parvo”, meaning “much in little”
- Aimed to address aliasing artifacts due to minification

Tokigun (Wikimedia Commons)

Mipmapping

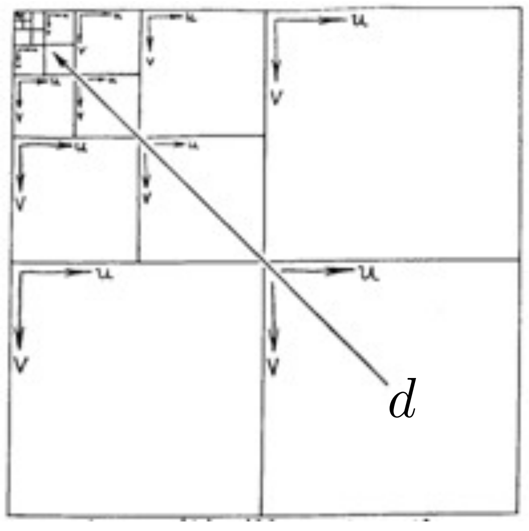




- 1/3 increase in memory requirements
- Key simplification: approximate the pre-image of a pixel by a square

Williams, SIGGRAPH 1983

Mipmapping

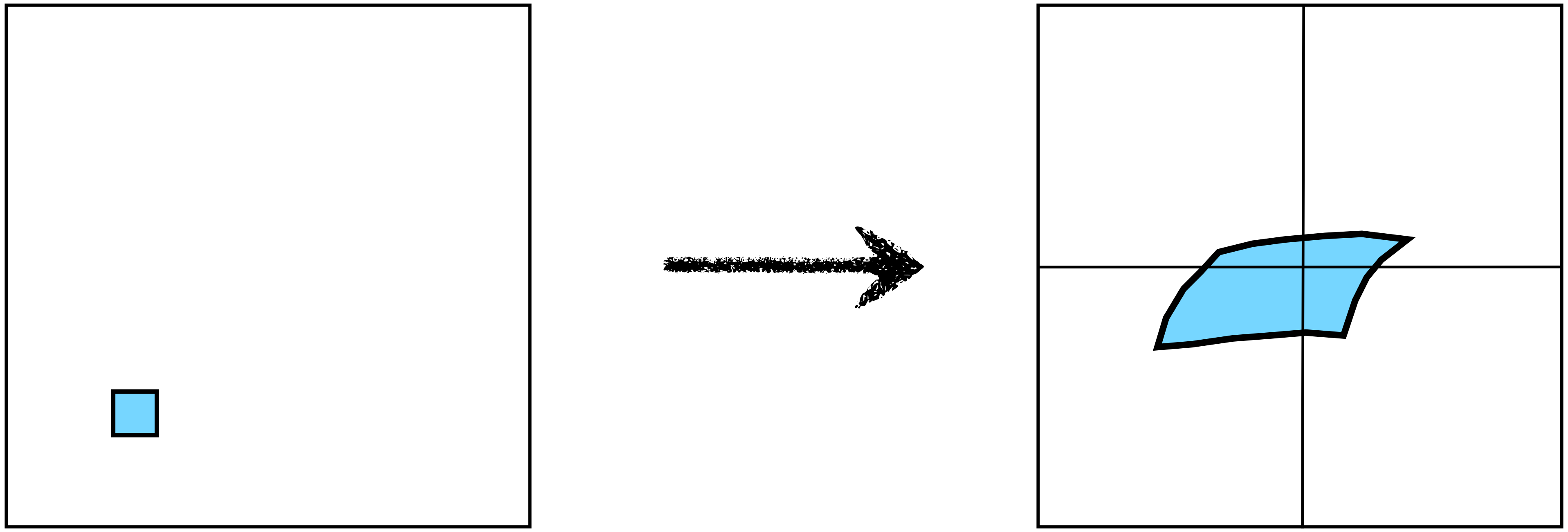


$$D = \max\left(\sqrt{\left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial v}{\partial x}\right)^2}, \sqrt{\left(\frac{\partial u}{\partial y}\right)^2 + \left(\frac{\partial v}{\partial y}\right)^2}\right) \qquad d = \log D$$

- Hierarchy indexed by mipmap level d, which ranges from 0 to log(w)
- Texture value produced by bilinear interpolation within two mipmap levels adjacent to d, then linear interpolation between the two values

Williams, SIGGRAPH 1983

Mipmapping



Anisotropic Filtering

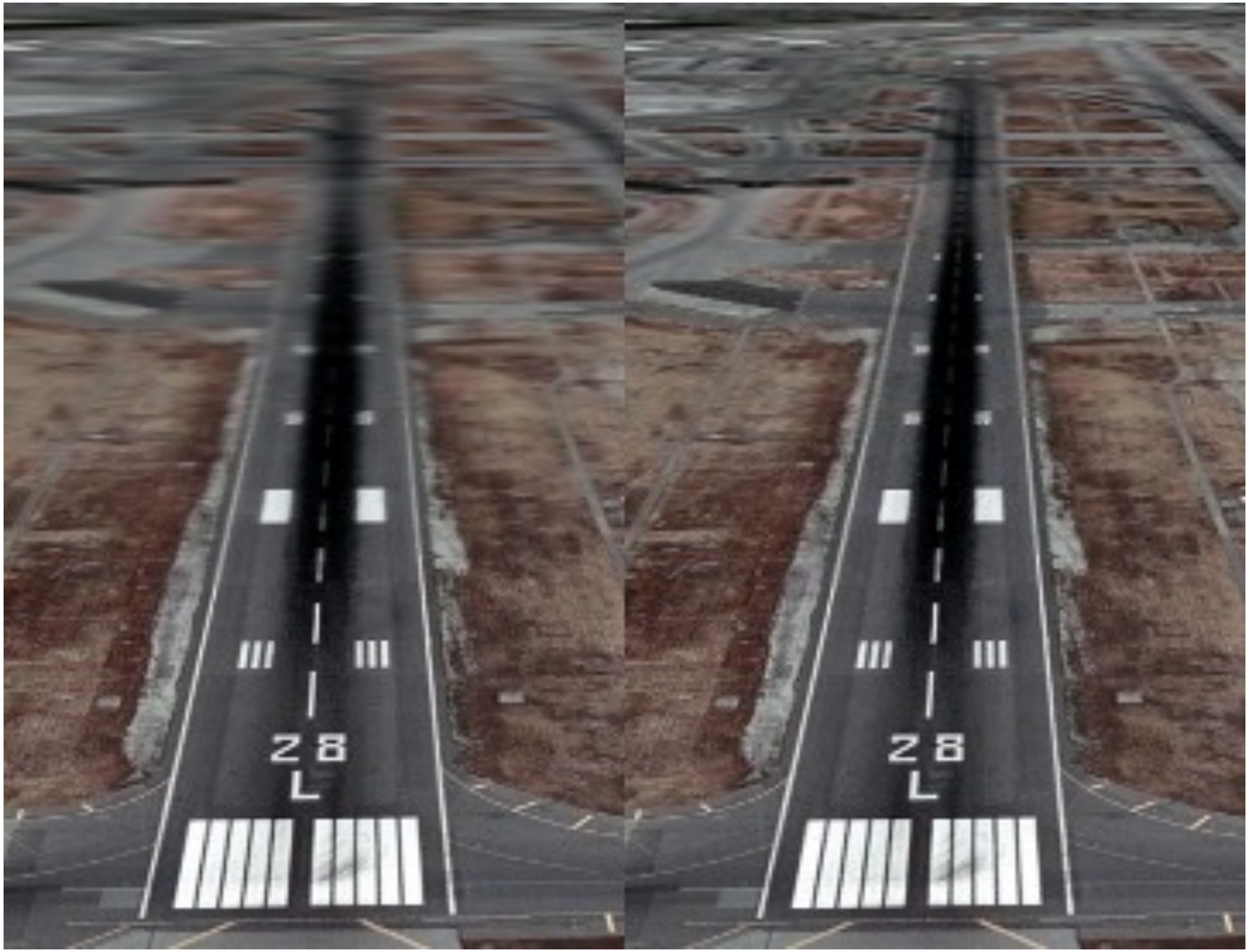
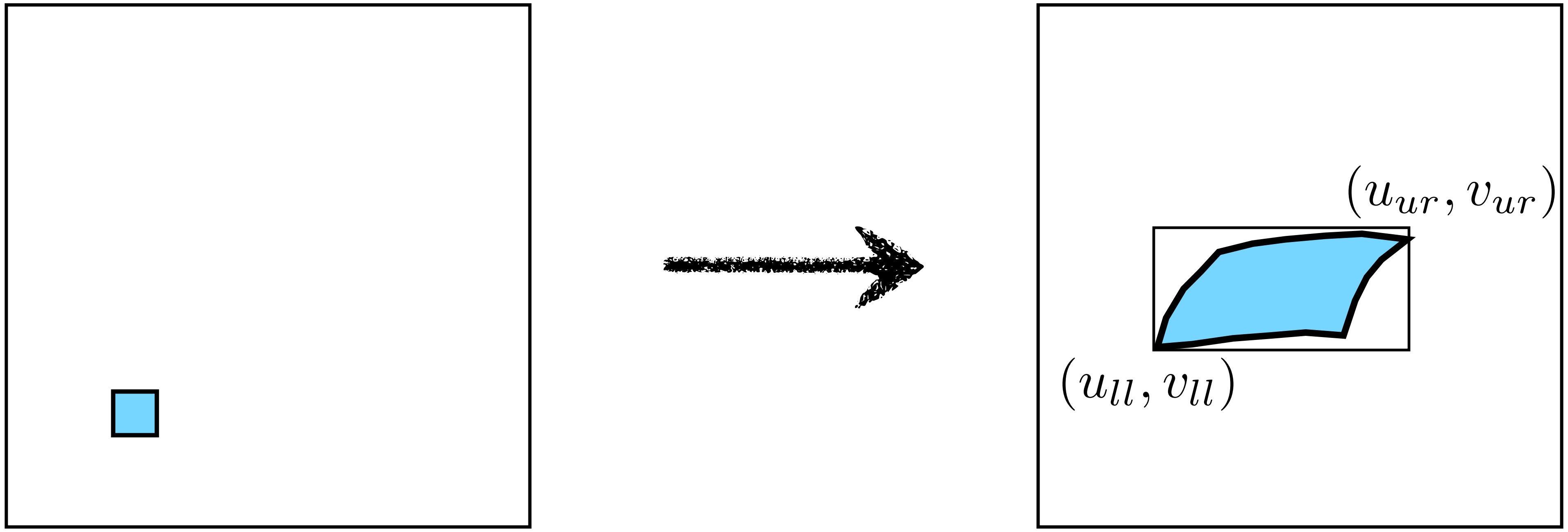


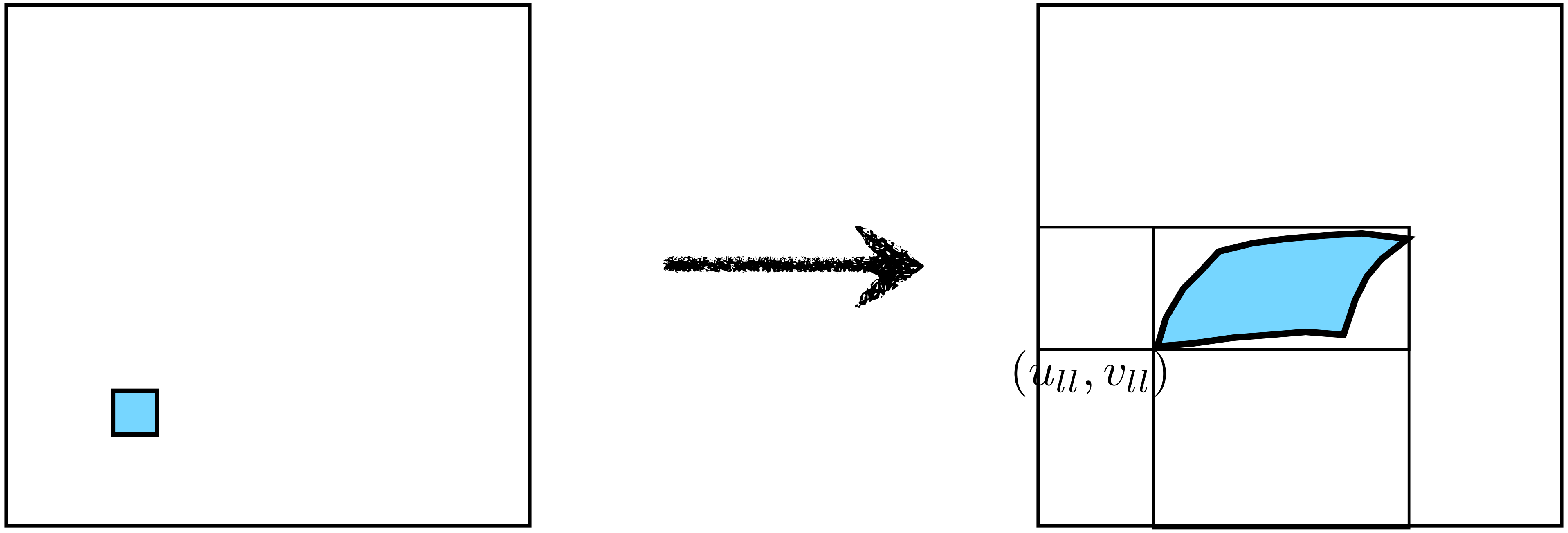
Image from Wikipedia

Summed-Area Tables



- Precompute an “image” S in which every pixel (x,y) stores the sum of values of all pixels (i,j) in the texture such that i≤x and j≤y.
- For each fragment, compute the texture coordinates for the lower left corner and the upper right corner of its pixel: $(u_{ll}, v_{ll}), (u_{ur}, v_{ur})$

Summed-Area Tables

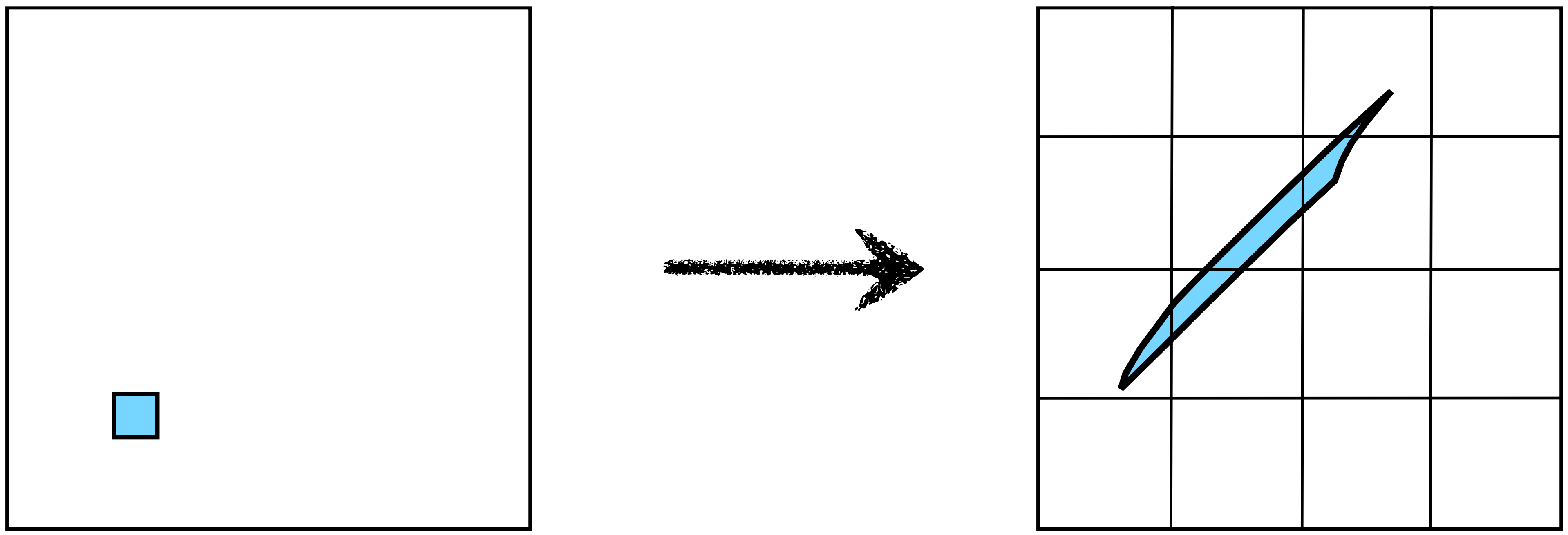


- Precompute an "image" S in which every pixel (x,y) stores the sum of values of all pixels (i,j) in the texture such that i≤x and j≤y.

- For each fragment, compute the texture coordinates for the bottom left corner and the upper right corner of its pixel: $(u_{ll}, v_{ll}), (u_{ur}, v_{ur})$

- Take $$\frac{S(u_{ur}, v_{ur}) - S(u_{ll}, v_{ur}) - S(u_{ur}, v_{ll}) + S(u_{ll}, v_{ll})}{(u_{ur} - u_{ll})(v_{ur} - v_{ll})}$$

Even More Accurate Filtering



$$D = \min\left(\sqrt{\left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial v}{\partial x}\right)^2}, \sqrt{\left(\frac{\partial u}{\partial y}\right)^2 + \left(\frac{\partial v}{\partial y}\right)^2}\right) \qquad d = \log D$$

- Take the mipmap level determined by the shorter edge of the pre-image
- Rasterize the pre-image as line segment
- Slower, but advantageous for narrow elongated pre-images