CS130 : Computer Graphics

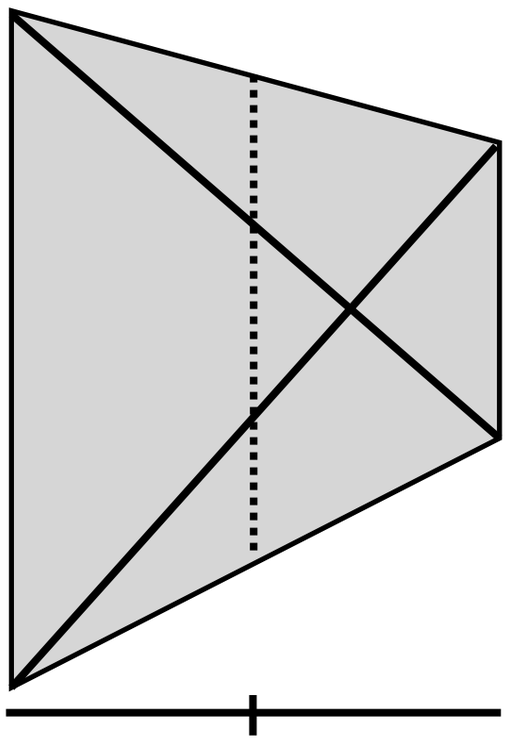
Texture Mapping (cont.)

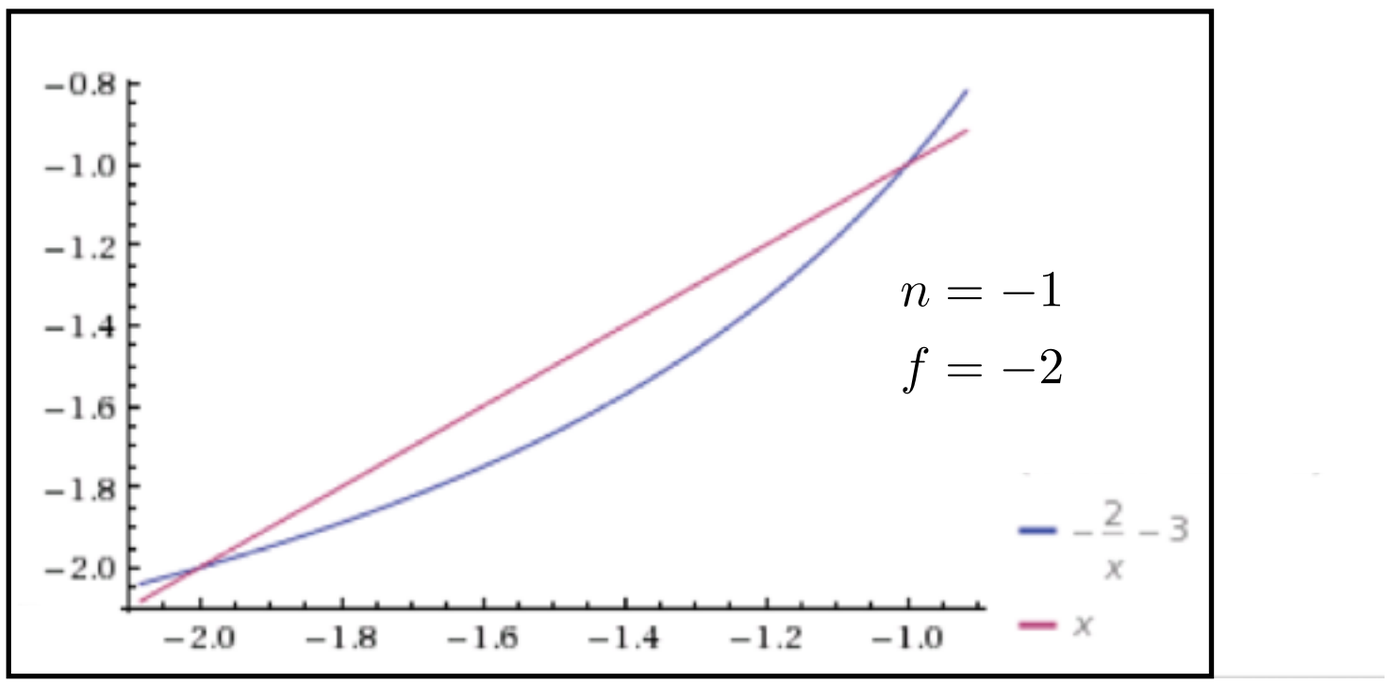
Tamar Shinar
Computer Science & Engineering
UC Riverside

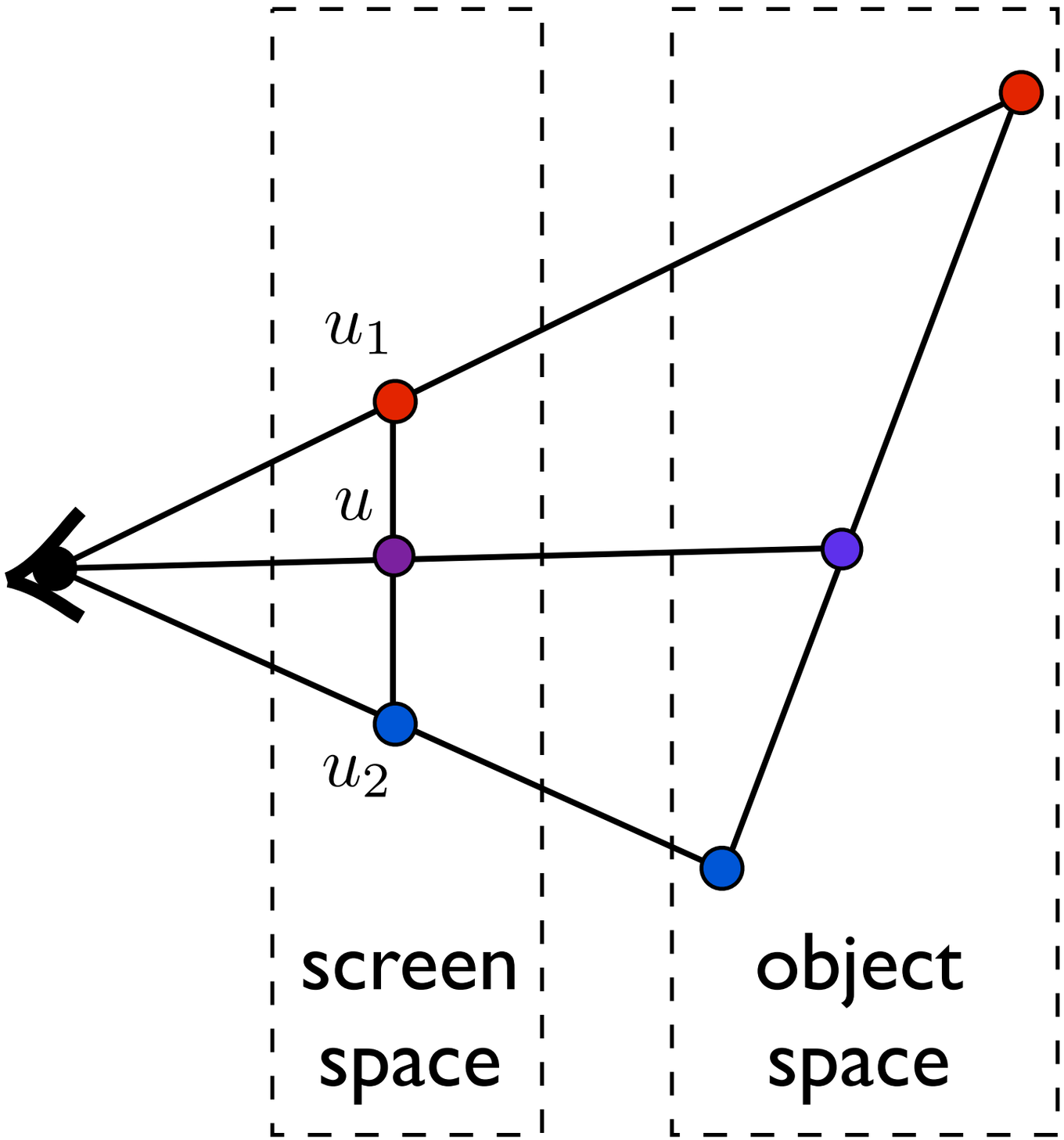
Perspective correct interpolation

Perspective correct interpolation

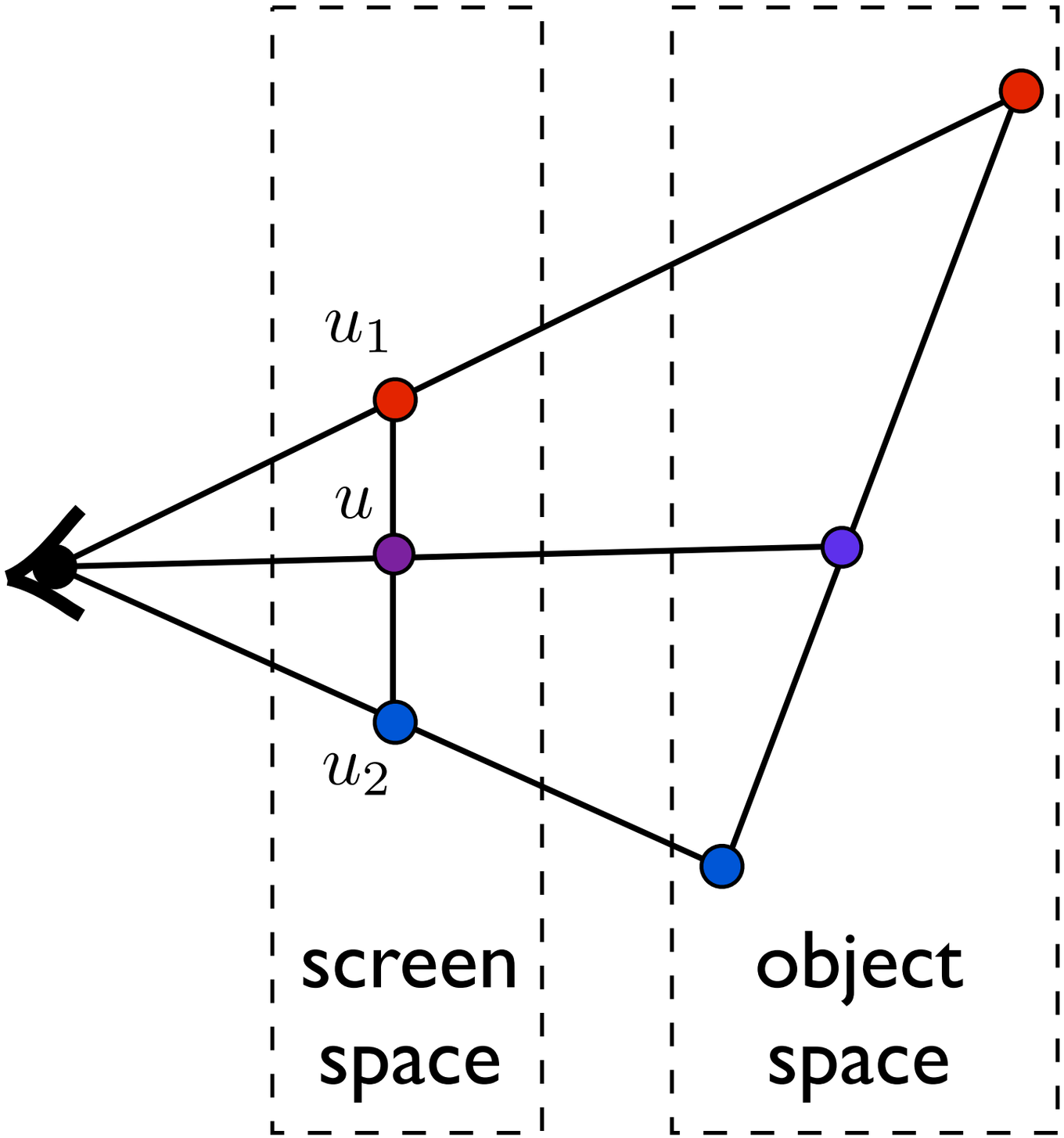
- In assignment 1, we found barycentric coordinates in 2D screen space
  - but not the correct object space barycentric coords
  - these coordinates were okay for z-buffer test





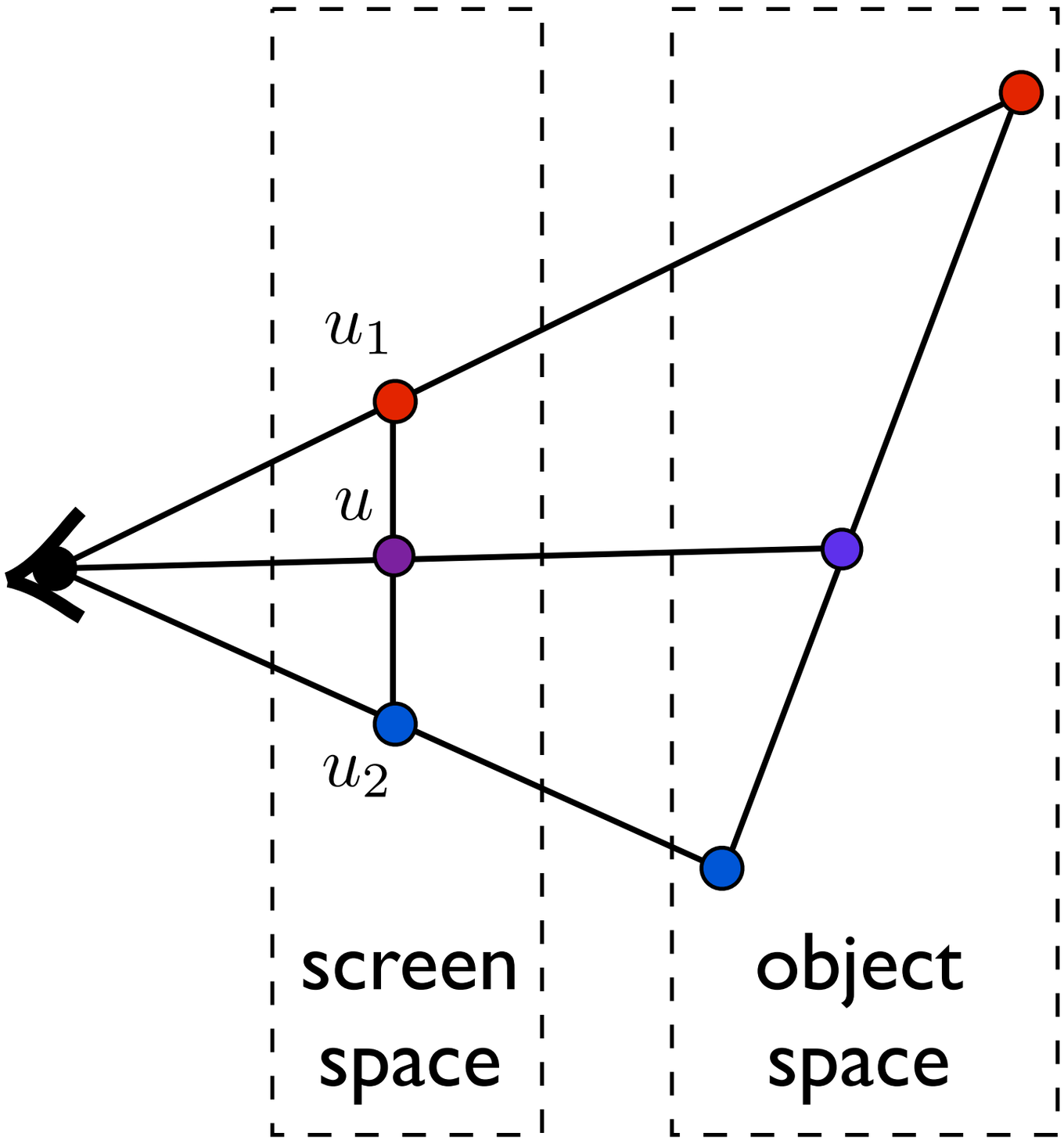


$$u = \frac{1}{2}u_1 + \frac{1}{2}u_2$$

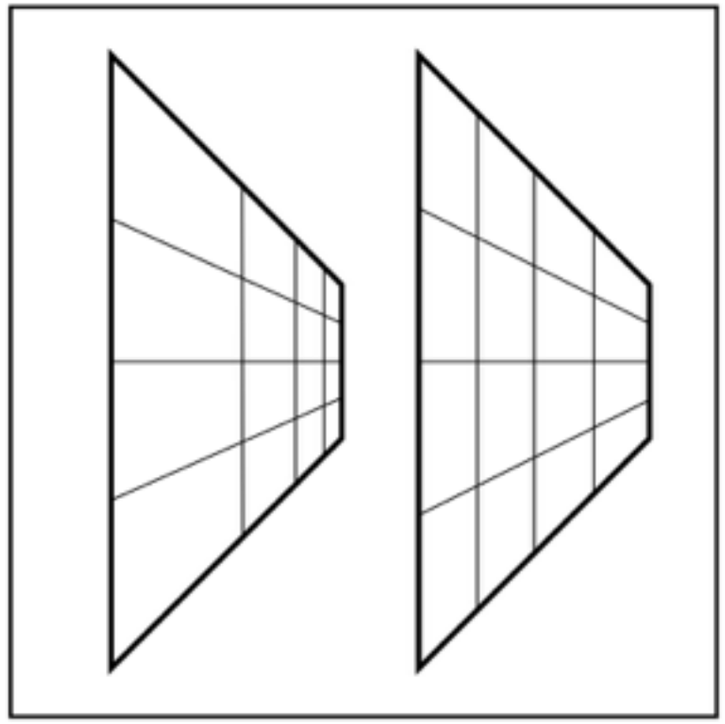


$$u = \frac{1}{2}u_1 + \frac{1}{2}u_2$$

Interpolation with screen space weights is incorrect



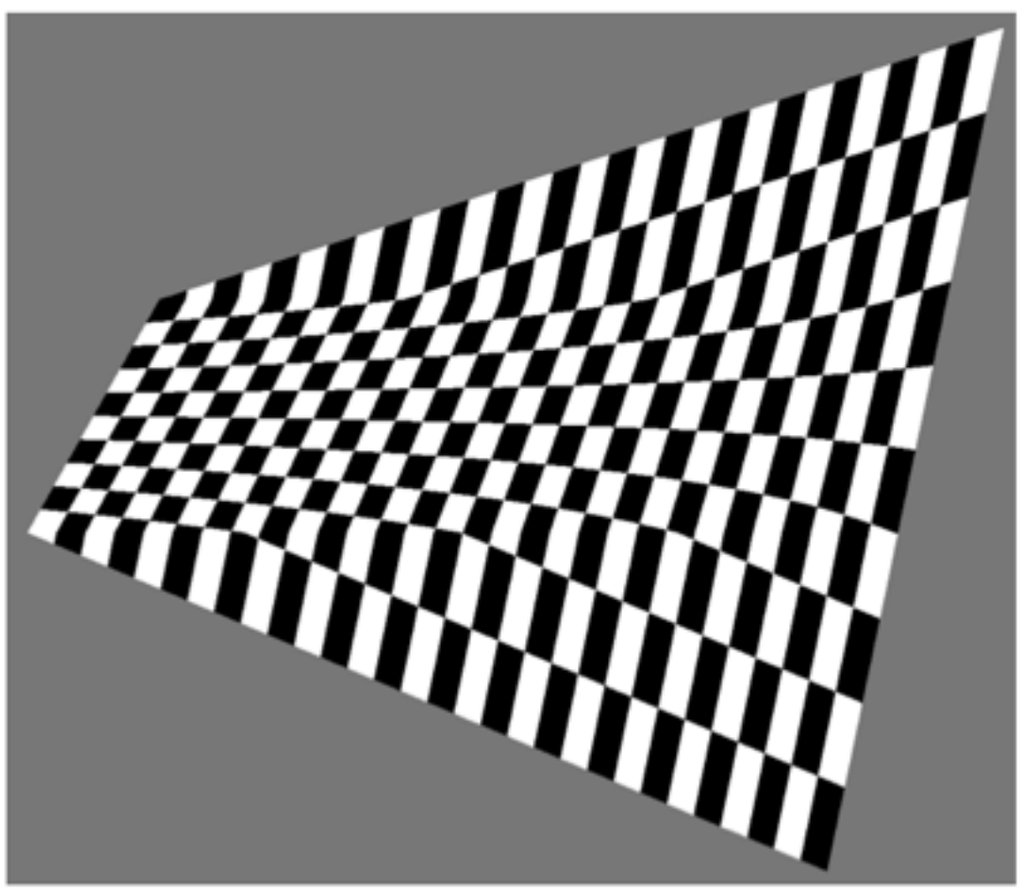
$$u = \frac{1}{2}u_1 + \frac{1}{2}u_2$$

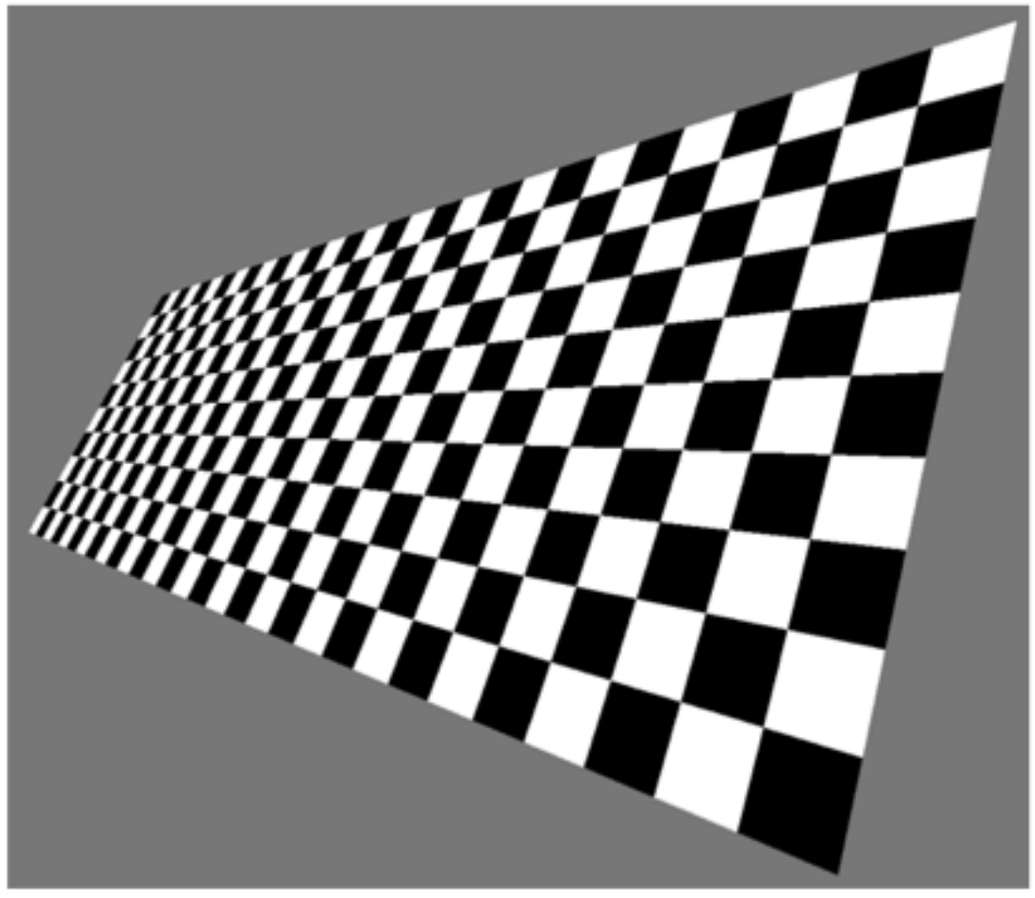


correct distorted

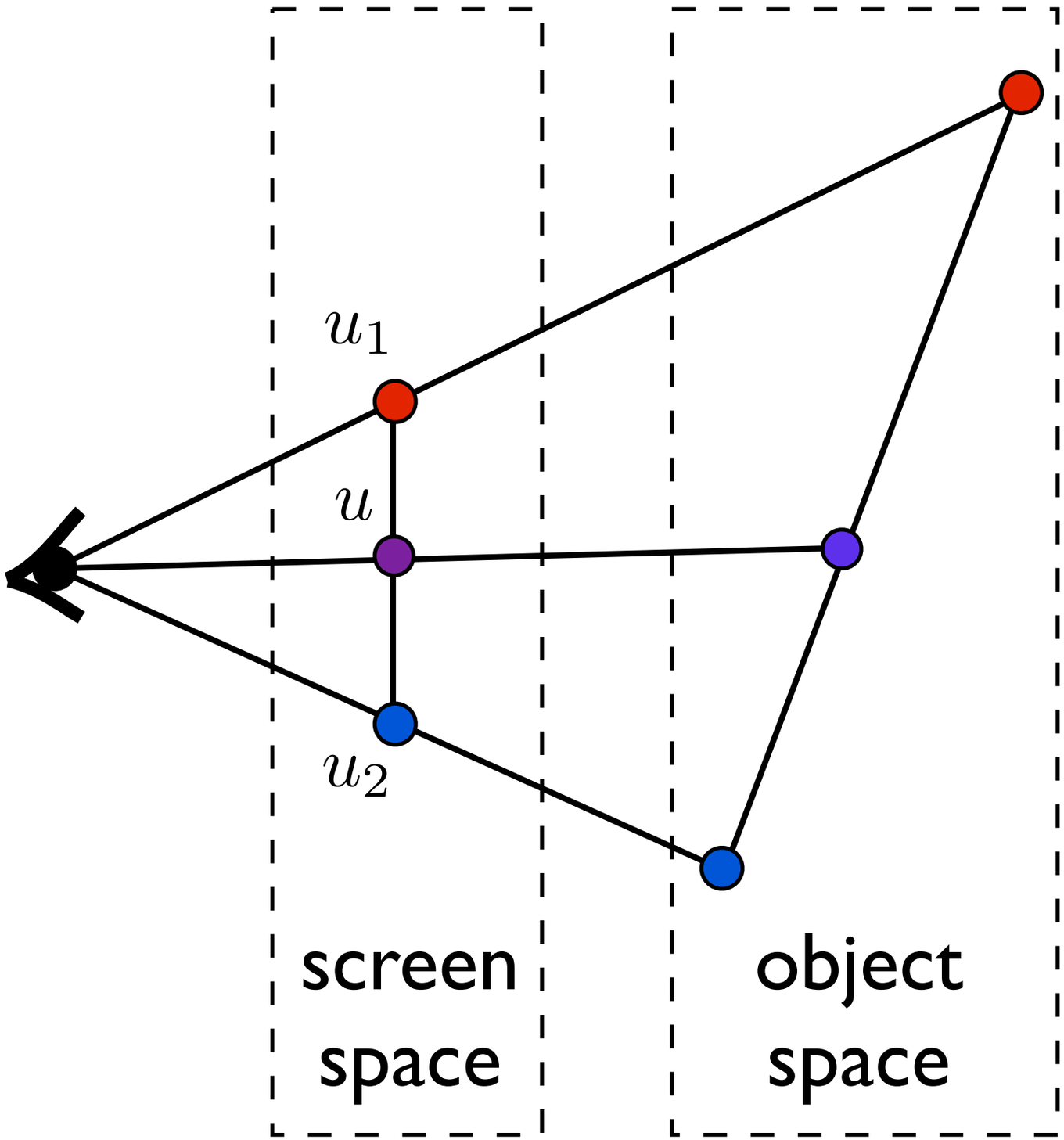
Perspective correct interpolation

Using screen space weights looks wrong for textures





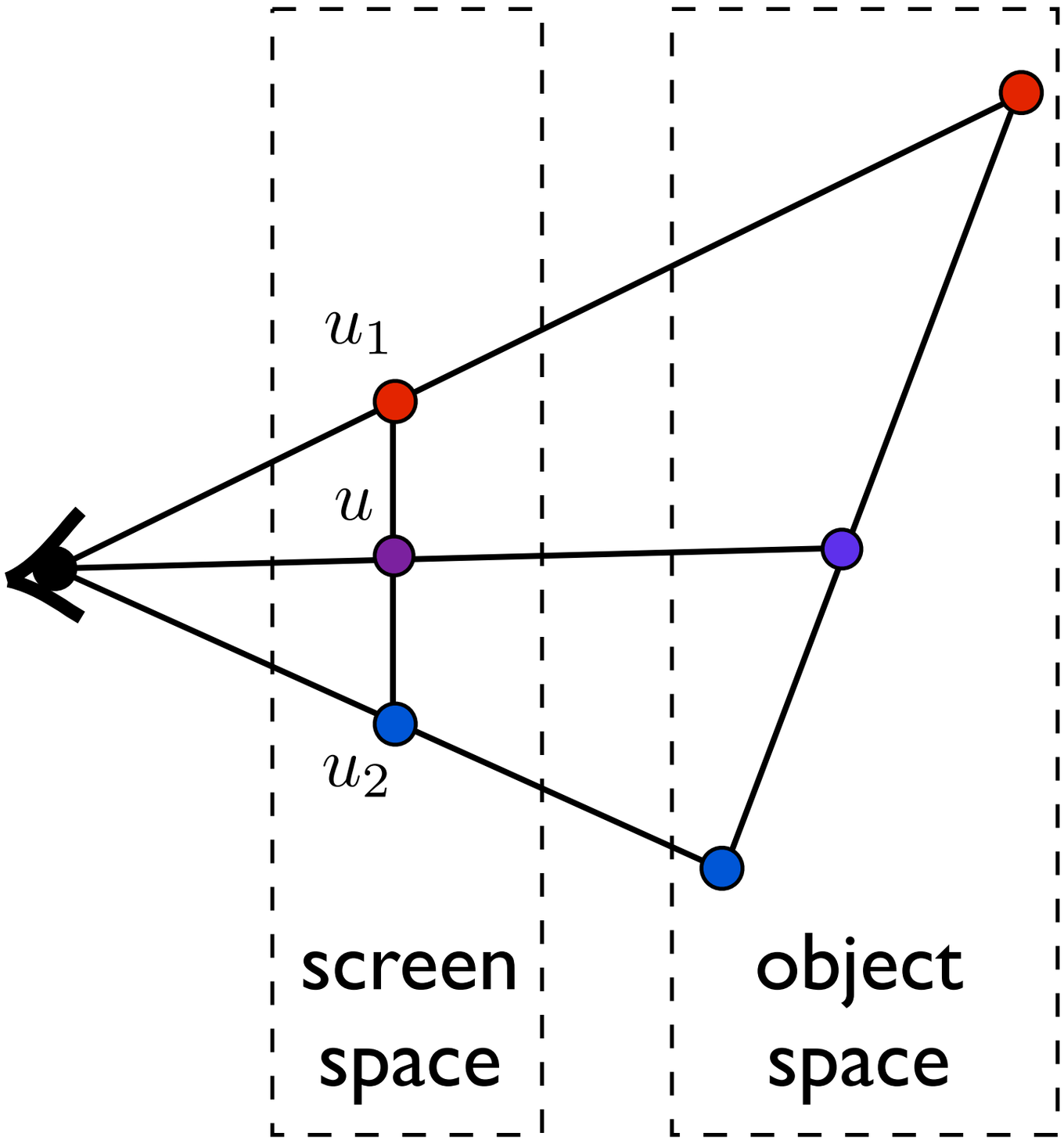
[Heckbert and Morton, 1990]



$$u = \frac{1}{2}u_1 + \frac{1}{2}u_2$$

Do we need to transform back to object space?

$$\mathbf{v}_{\mathrm{sc}} = M_{\mathrm{vp}} M_{\mathrm{pers}} M_{\mathrm{cam}} \mathbf{v}$$



$$u = \frac{1}{2}u_1 + \frac{1}{2}u_2$$

Do we need to transform back to object space?

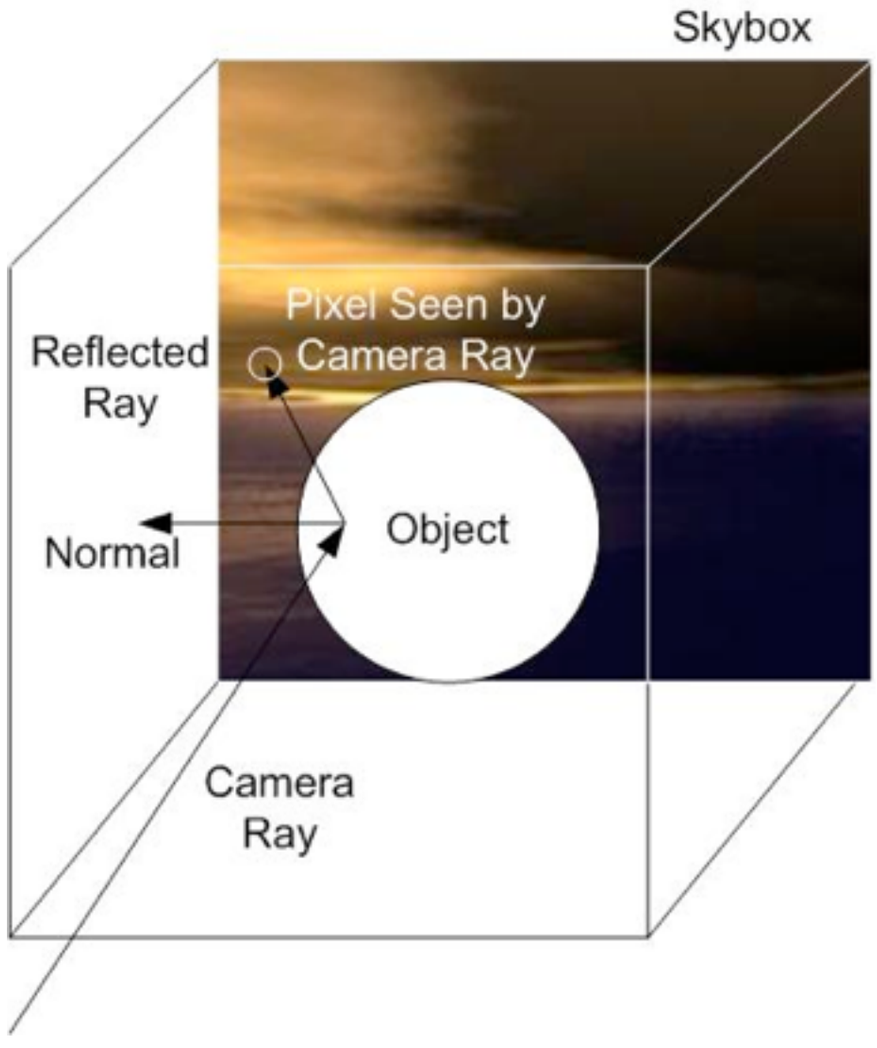
NO!

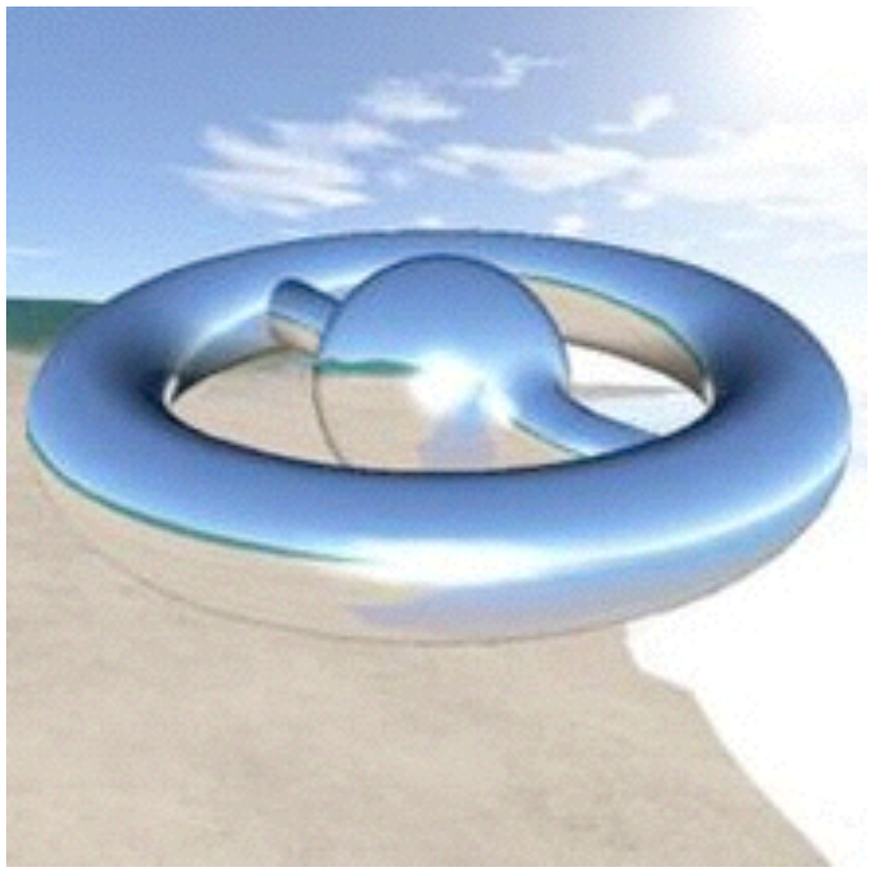
<whiteboard>

Environment mapping

Environment Mapping

Use a texture for the distant environment simulate the effect of ray tracing more cheaply

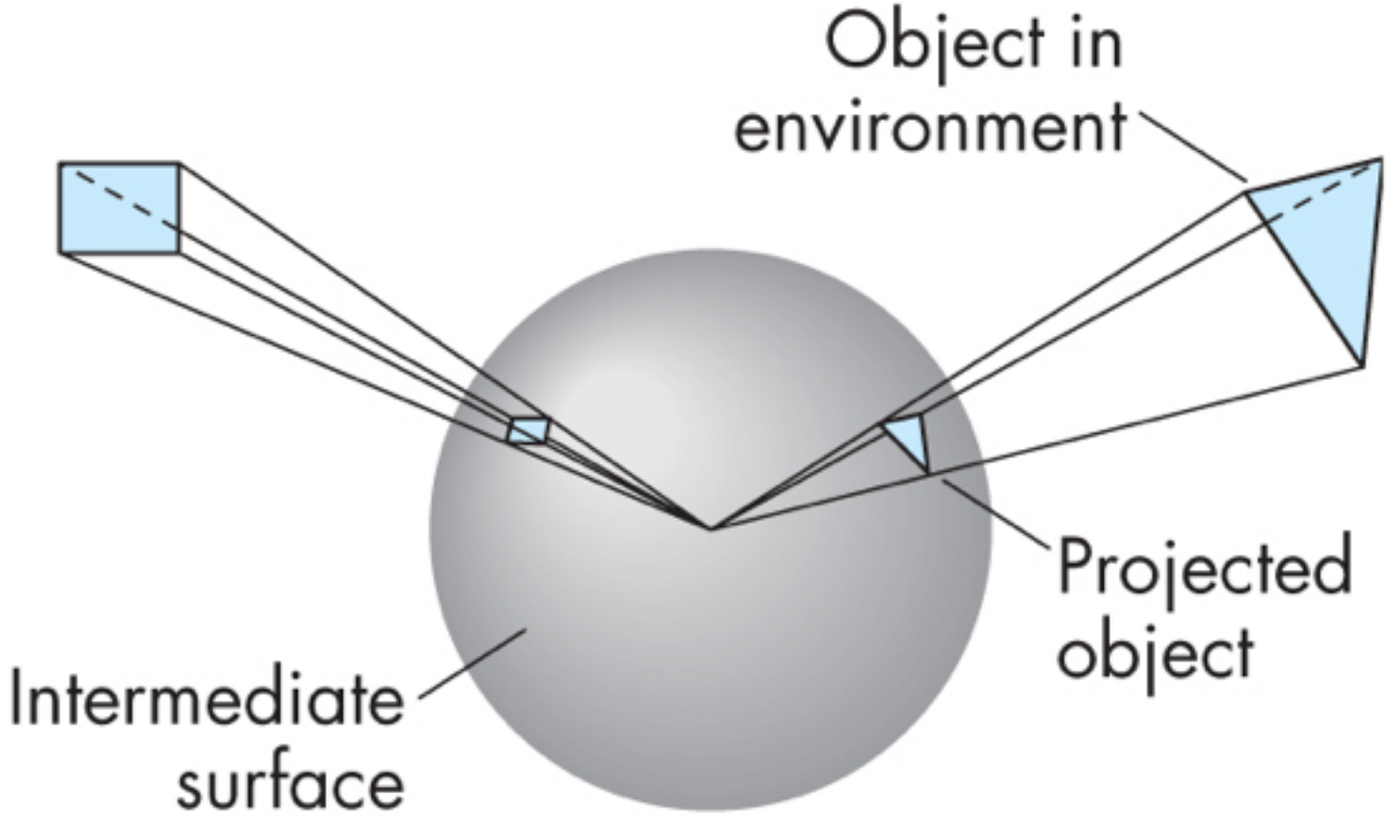




Wikimedia Commons

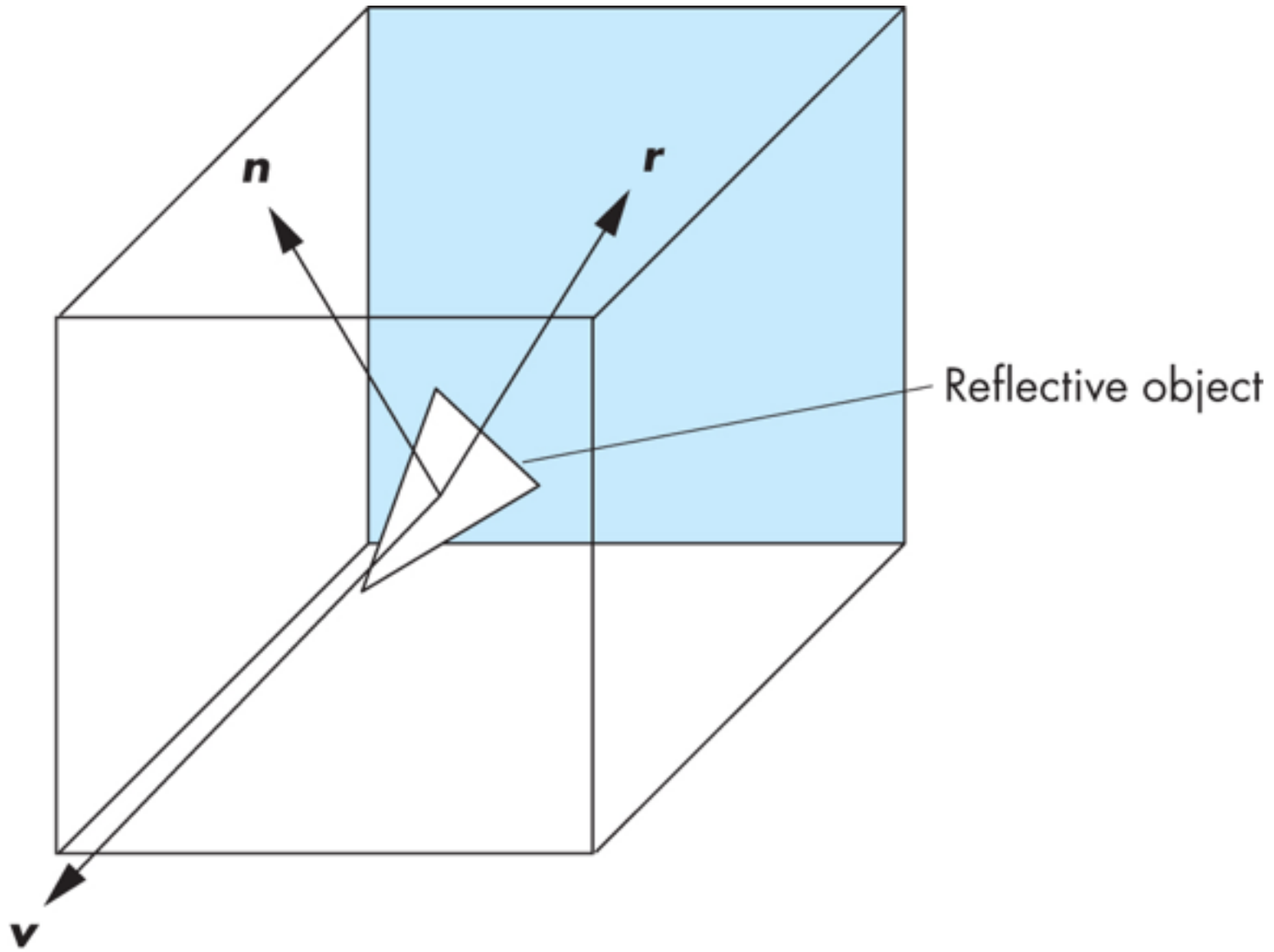
Sphere Mapping

- Project objects in the environment onto sphere centered at eye
- unwrap and store as texture
- use reflection direction to lookup texture value

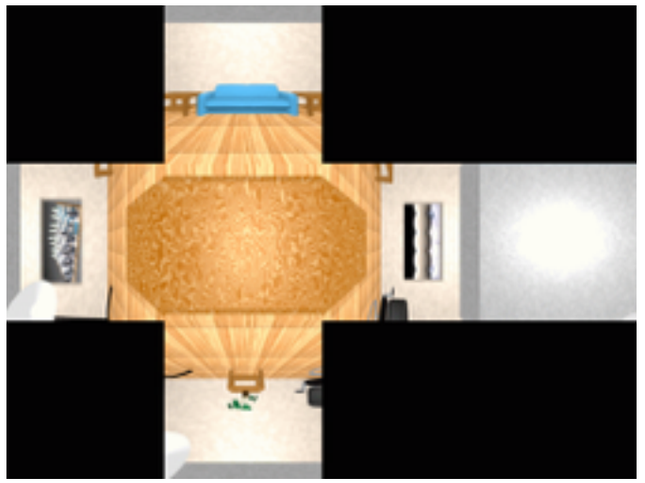


Cube Mapping

- Compute six projections, one for each wall
- store as texture
- use reflection direction to lookup texture value



Different environment maps



www.reindelsoftware.com

Blinn/Newell latitude mapping

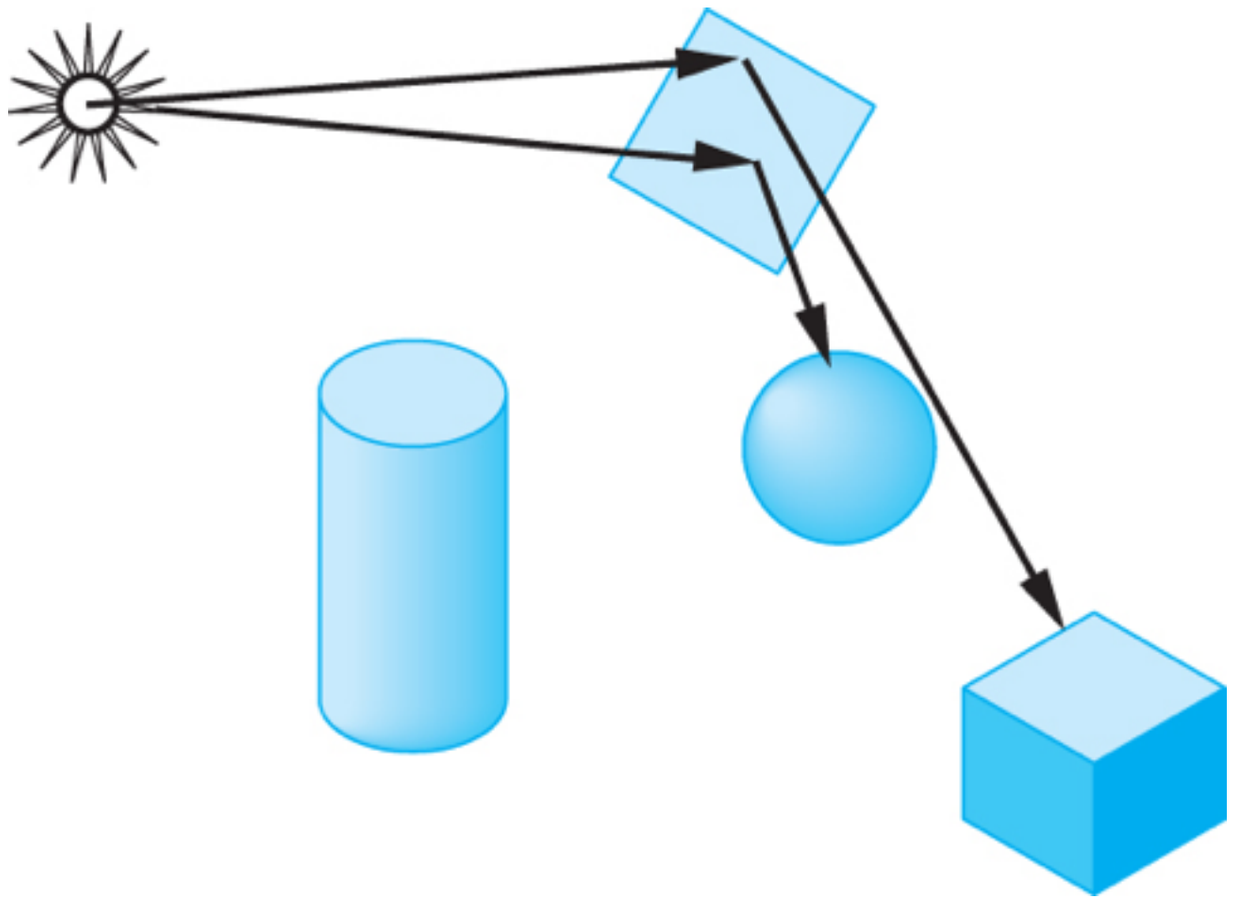
OpenGL spherical mapping

Cube mapping

Environment Mapping

Create the effect of a mirror with two-pass rendering

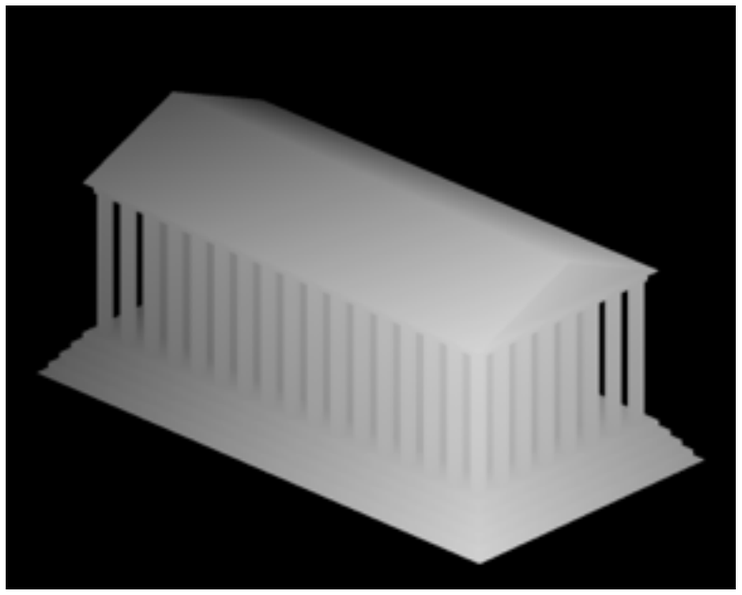
1. First pass: render the scene from the perspective of the mirror
2. Second pass: render from original pov; use the first image as a texture for the mirror

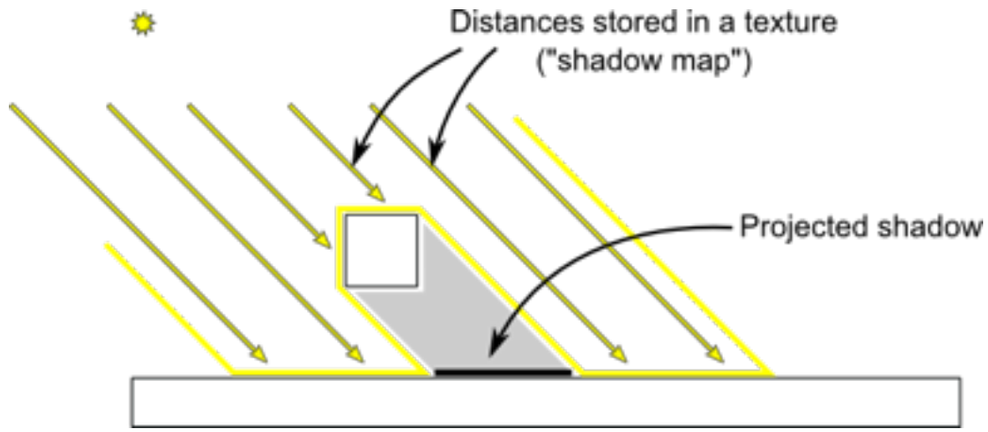


Shadow Mapping

2 passes:

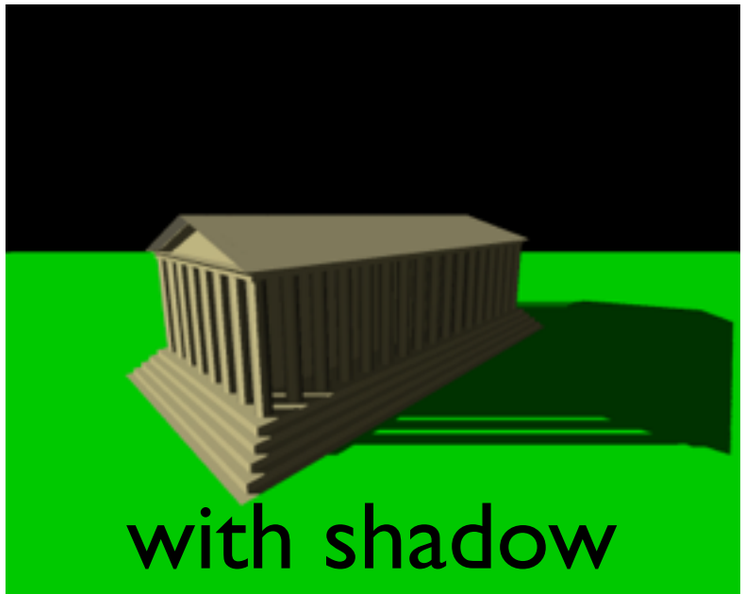
1. render scene from pov of light and store z-buffer in a texture

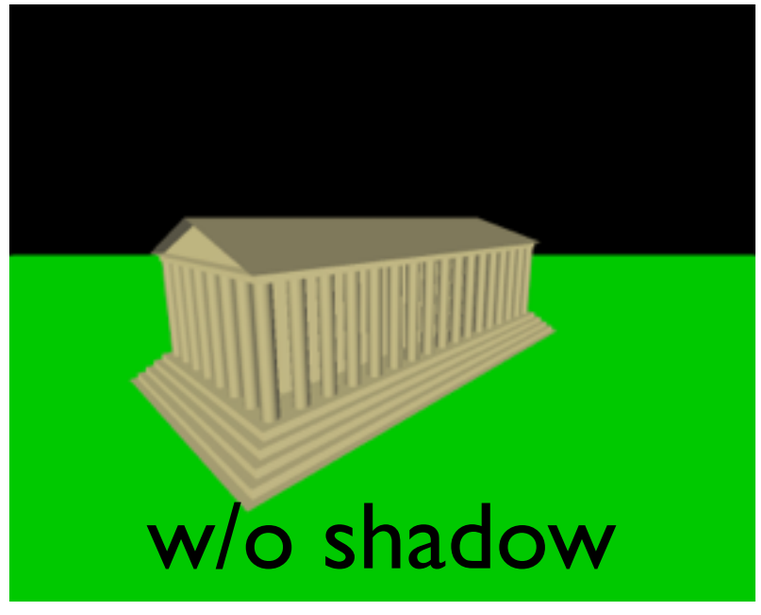




http://www.opengl-tutorial.org/intermediate-tutorials/tutorial-16-shadow-mapping/

2. when rendering scene from desired pov, also render from light pov and test pixel against stored texture





Wikimedia Commons

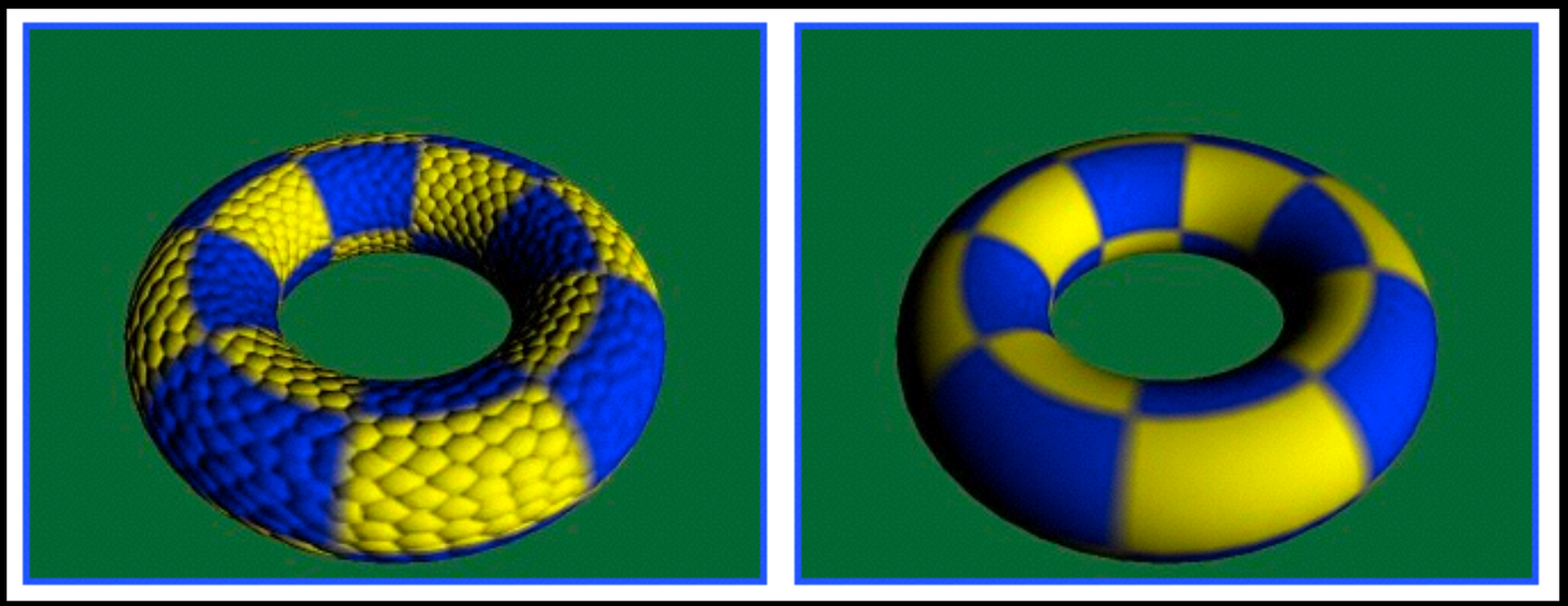
Bump Mapping

perturb
normal
vectors

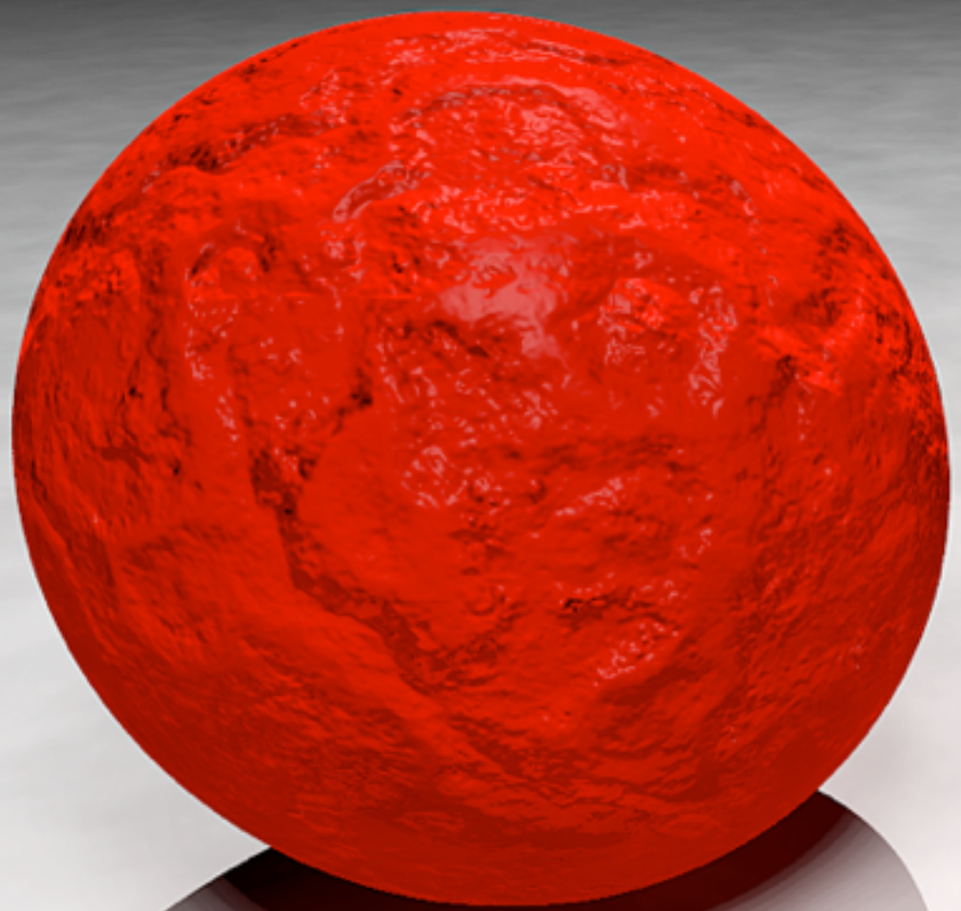
doesn't
affect
silhouette

[DirectXTutorial.com]

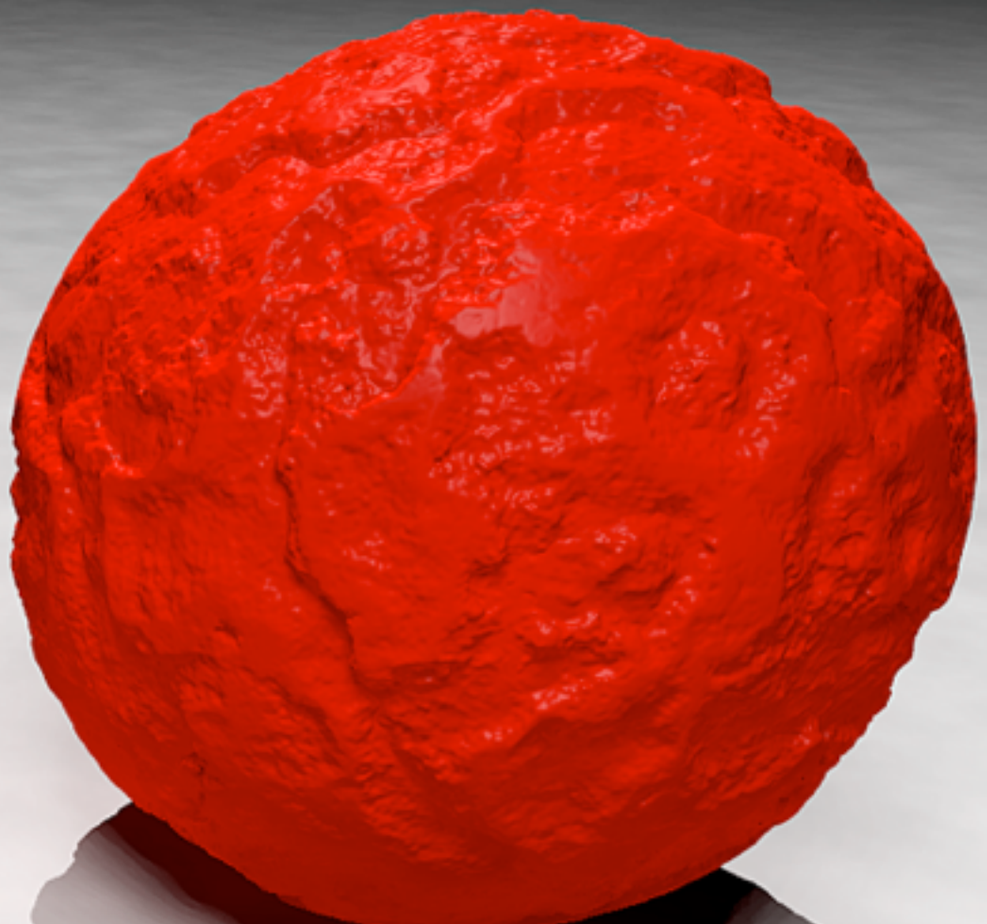
http://www.lg.clanhost.cz



http://www.paulsprojects.net/tutorials/simplebump/simplebump.html



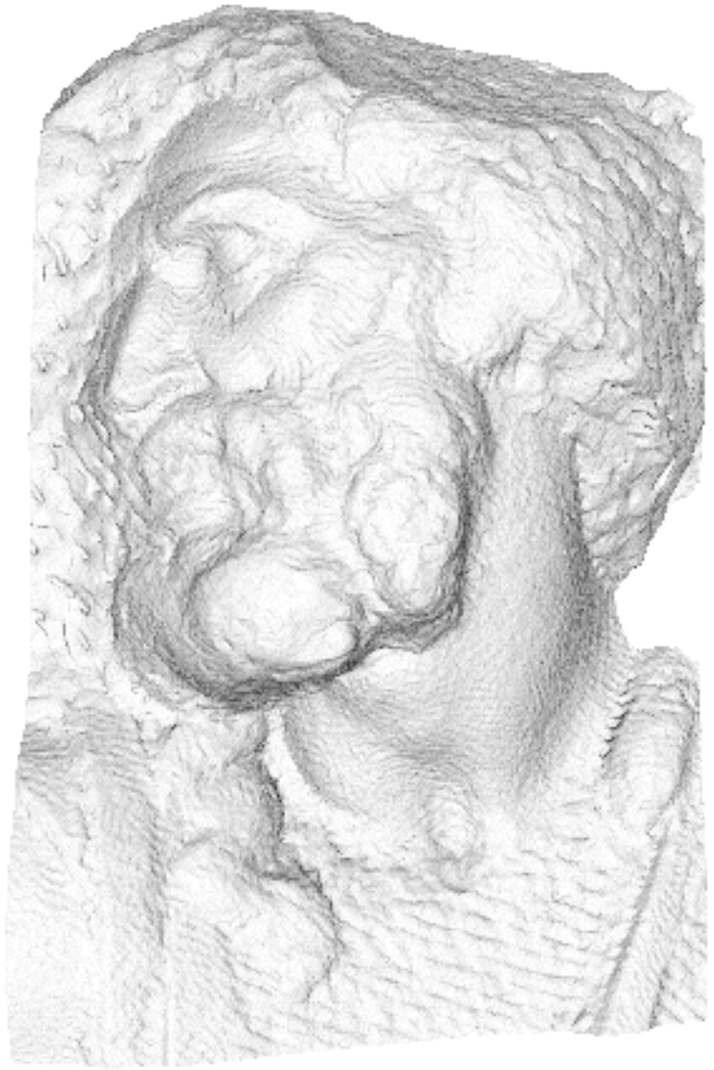
bump mapping



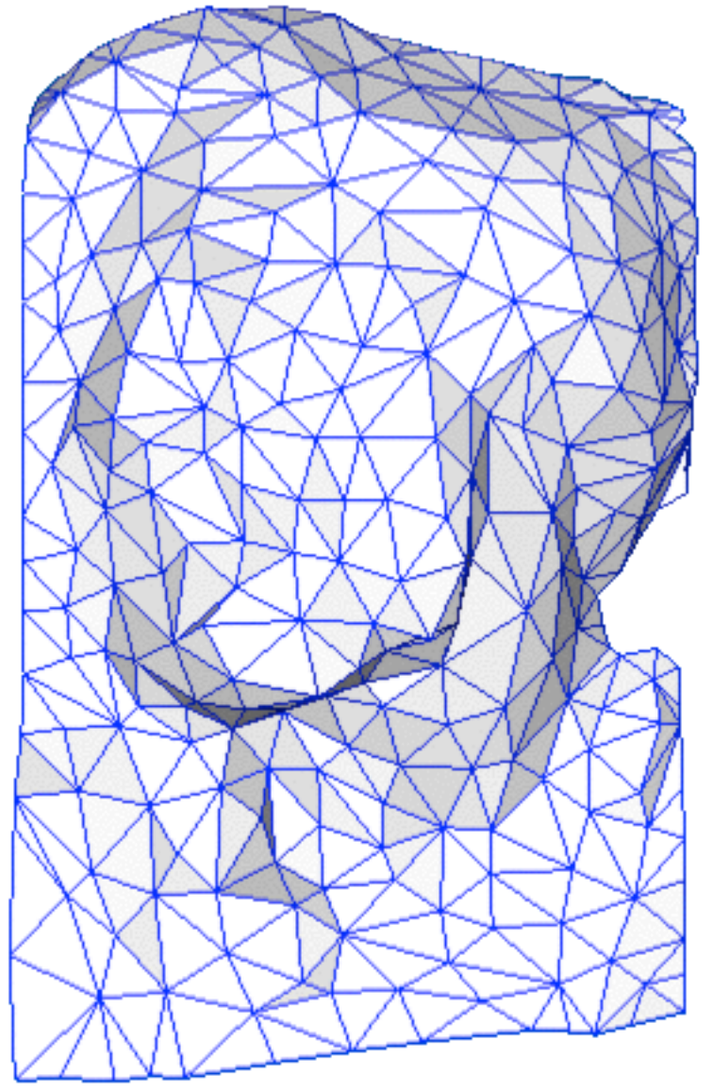
geometric detail

Wikimedia Commons

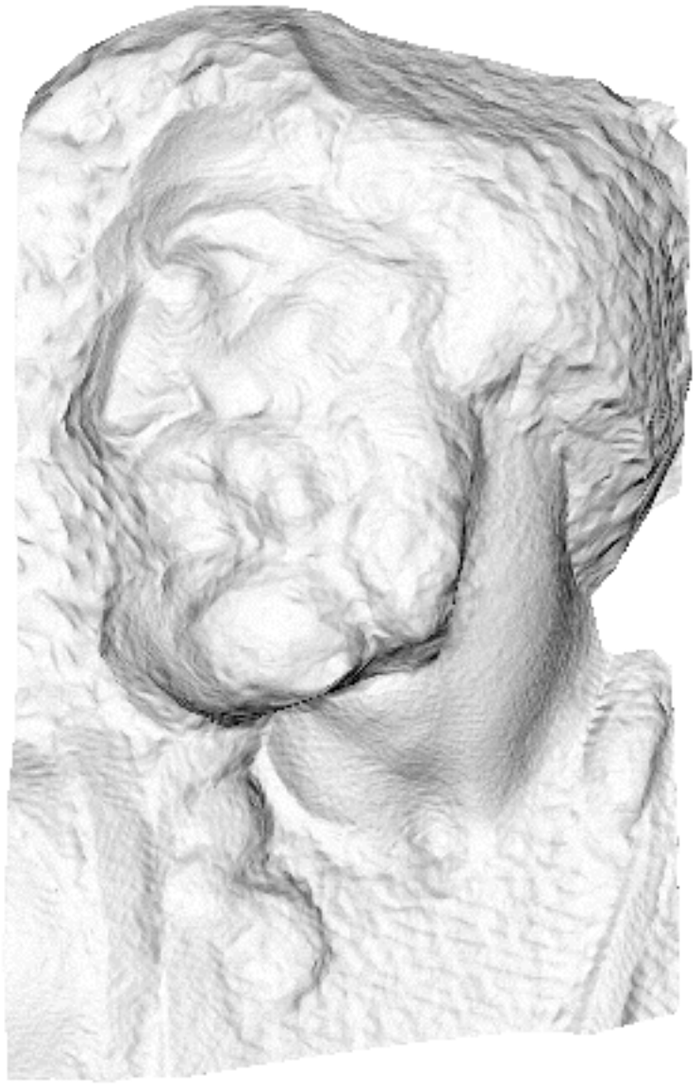
Normal Mapping



original mesh
4M triangles



simplified mesh
500 triangles



simplified mesh
and normal mapping
500 triangles

Wikimedia Commons

Normal Mapping

Example of a normal map (center) with the scene it was calculated from (left) and the result when applied to a flat surface (right).