CS130 : Computer Graphics

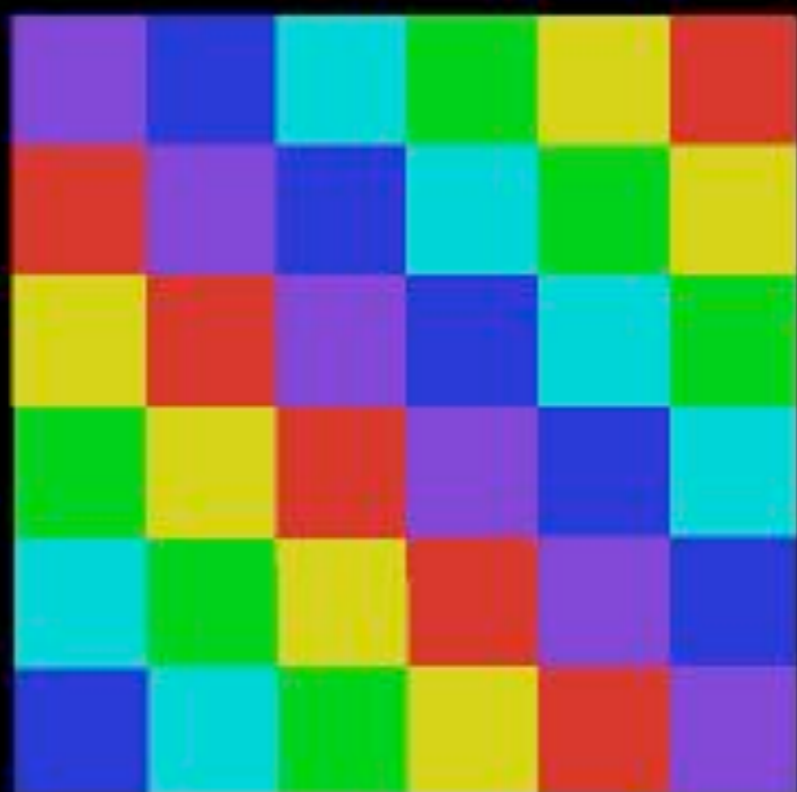## Lecture 16: Texture Mapping (cont.)

Tamar Shinar
Computer Science & Engineering
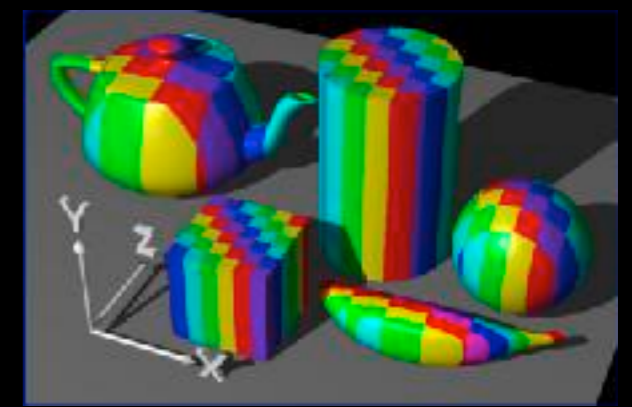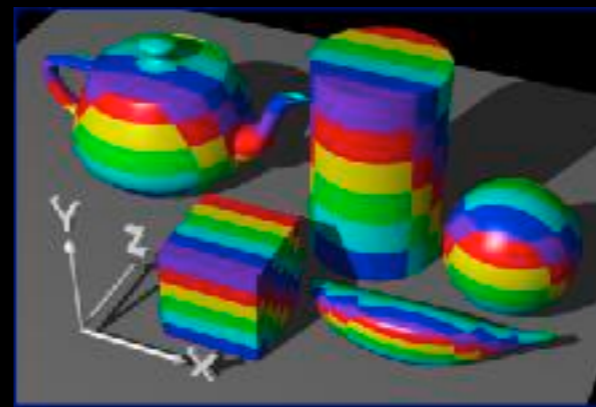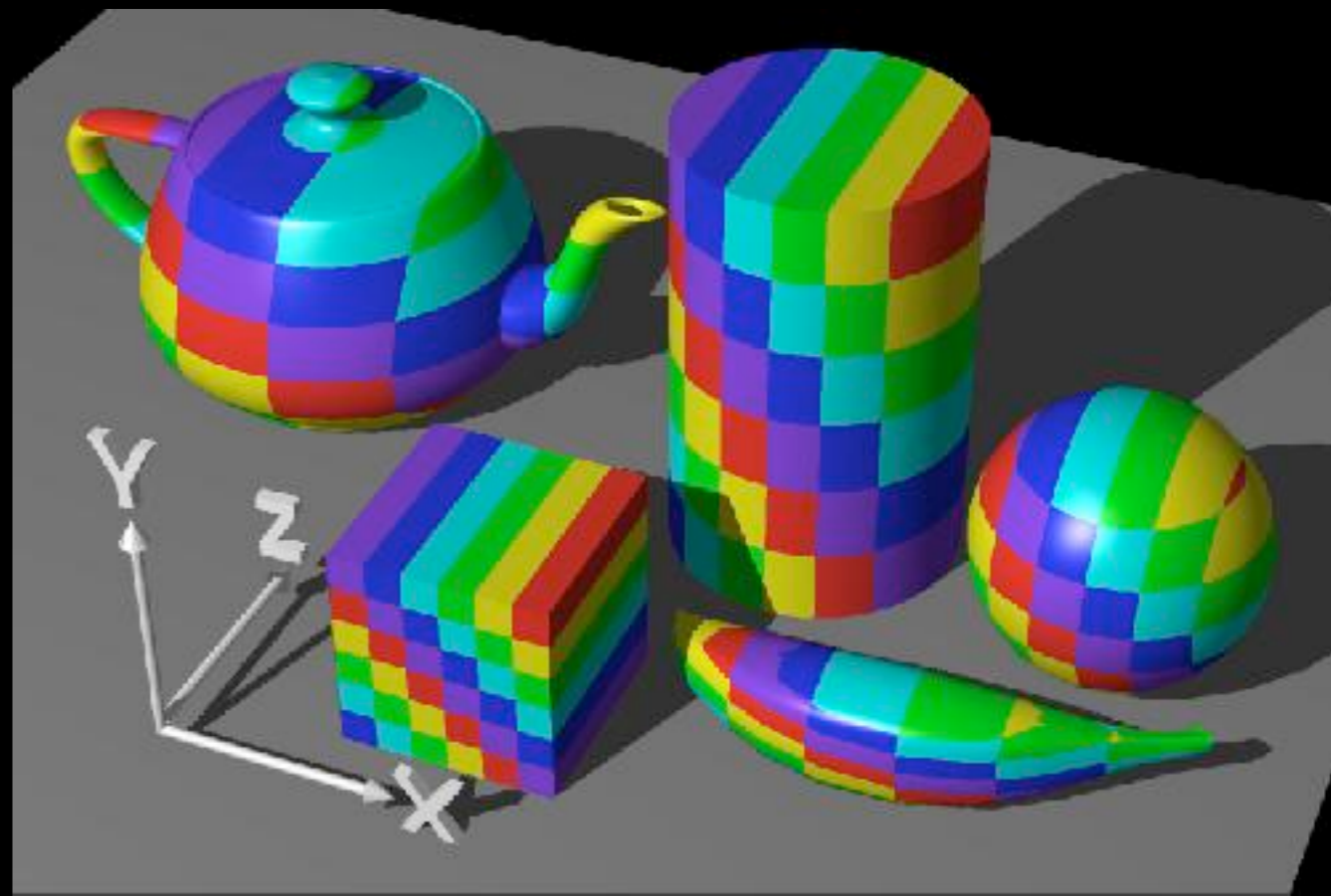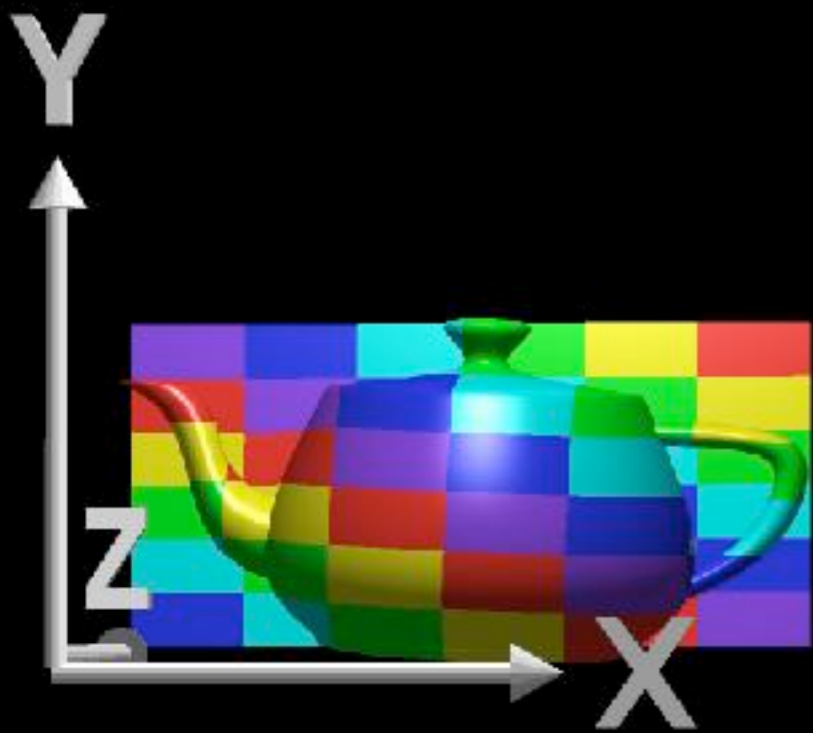UC Riverside

Given a point on the object **(x,y,z)**, what point **(u,v)** in the texture we use?
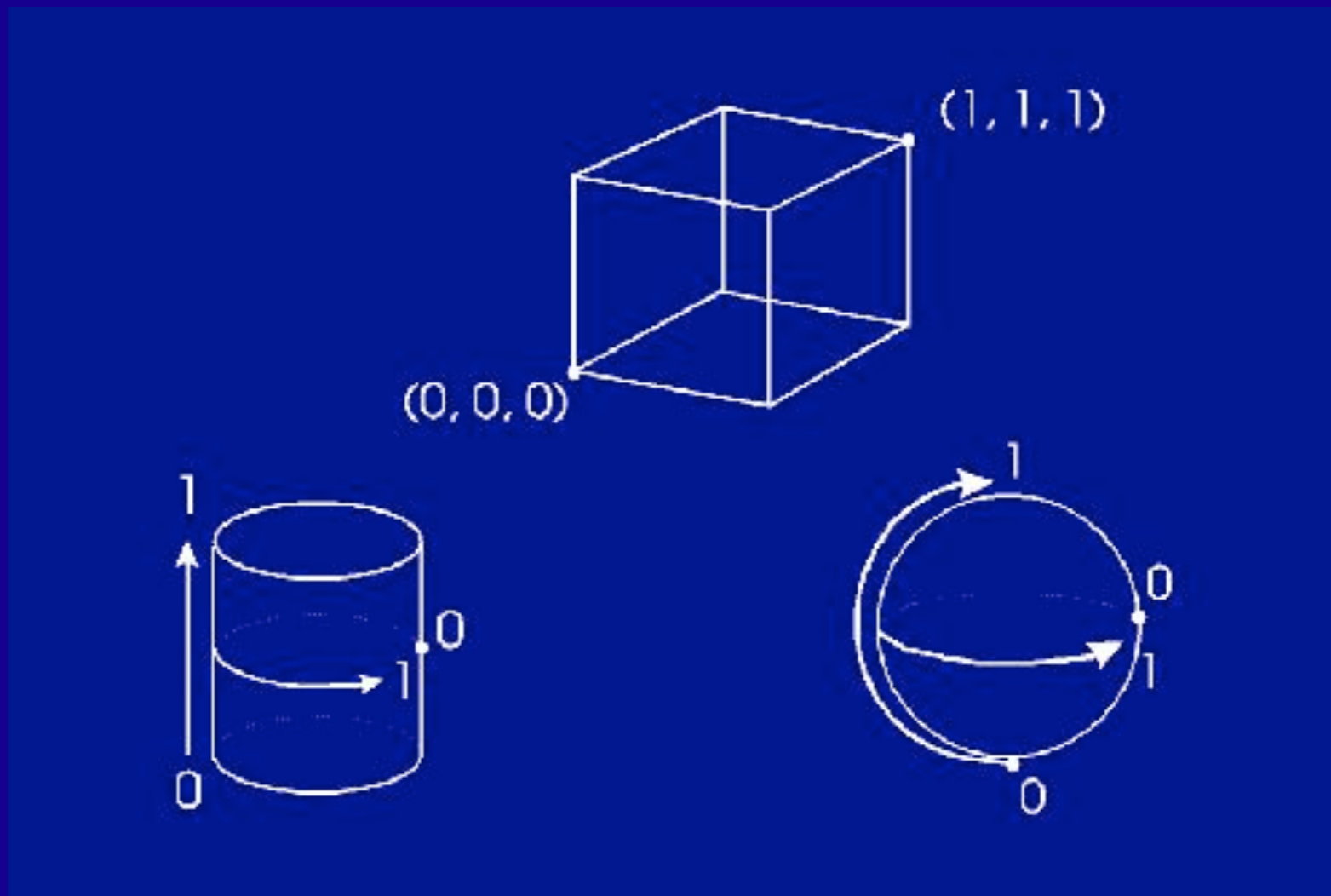
Example: planar mapping

Intermediate surfaces

First map the texture to a simpler, intermediate surface
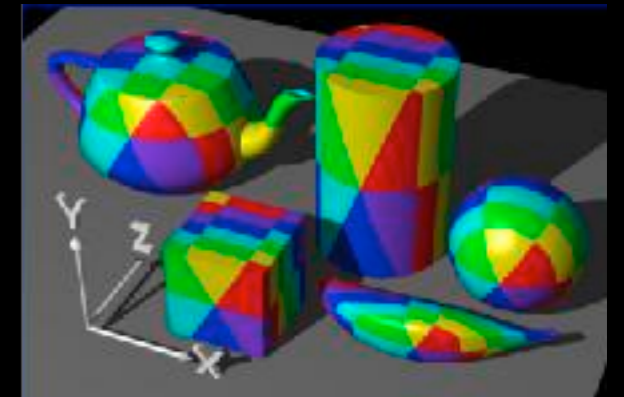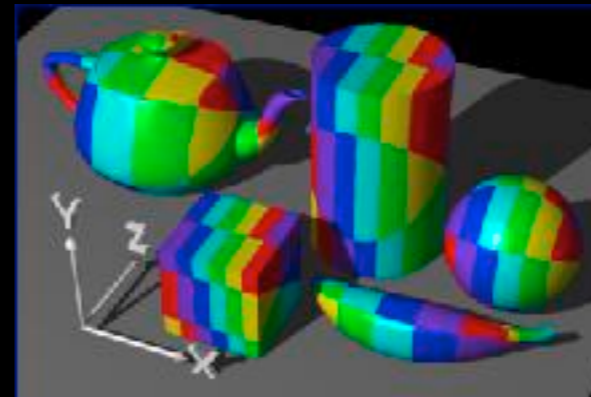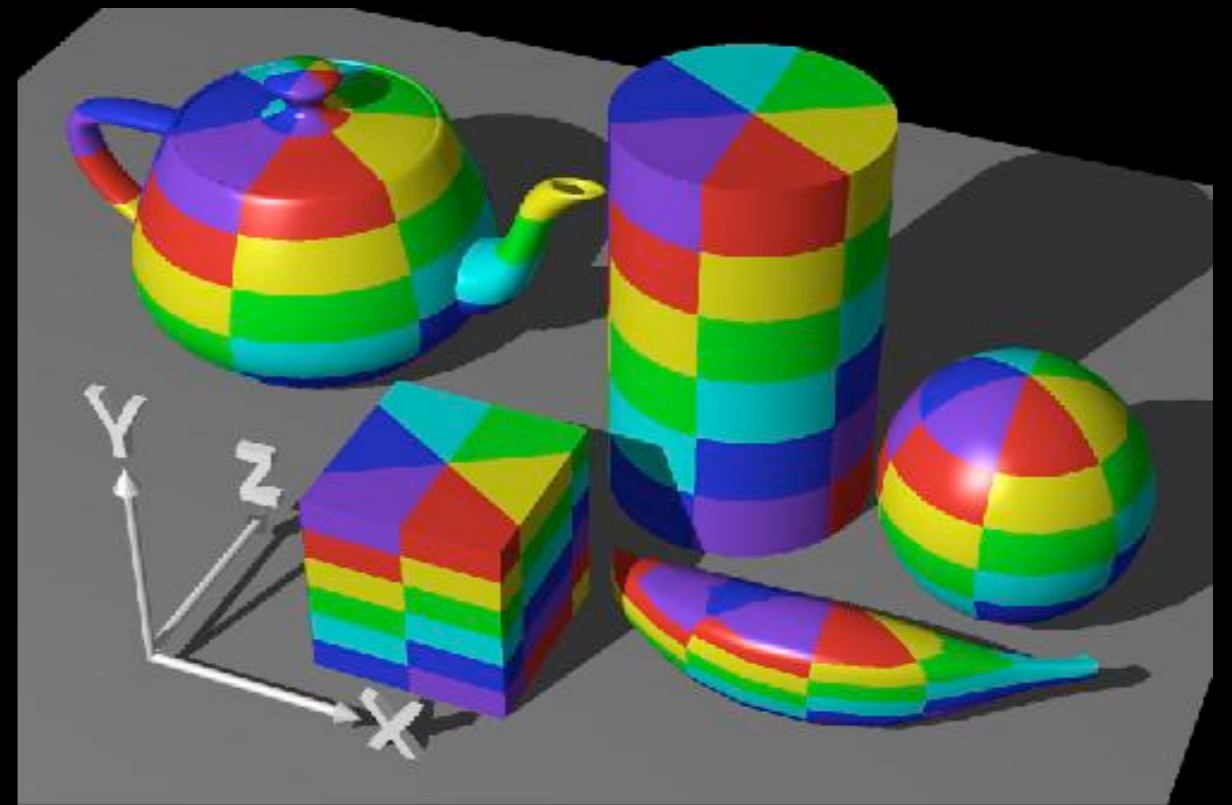
Cylindrical mapping

$(x,y,z) \rightarrow (theta, h) \rightarrow (u,v)$
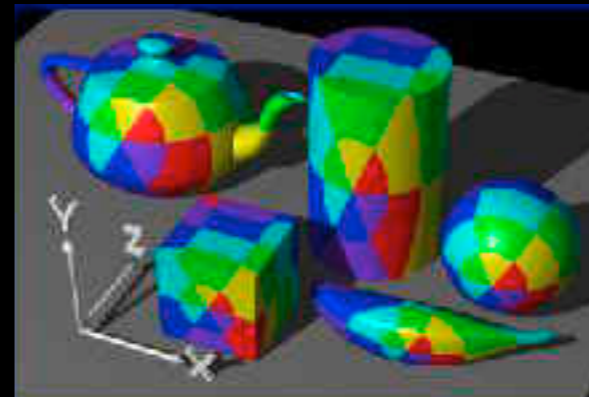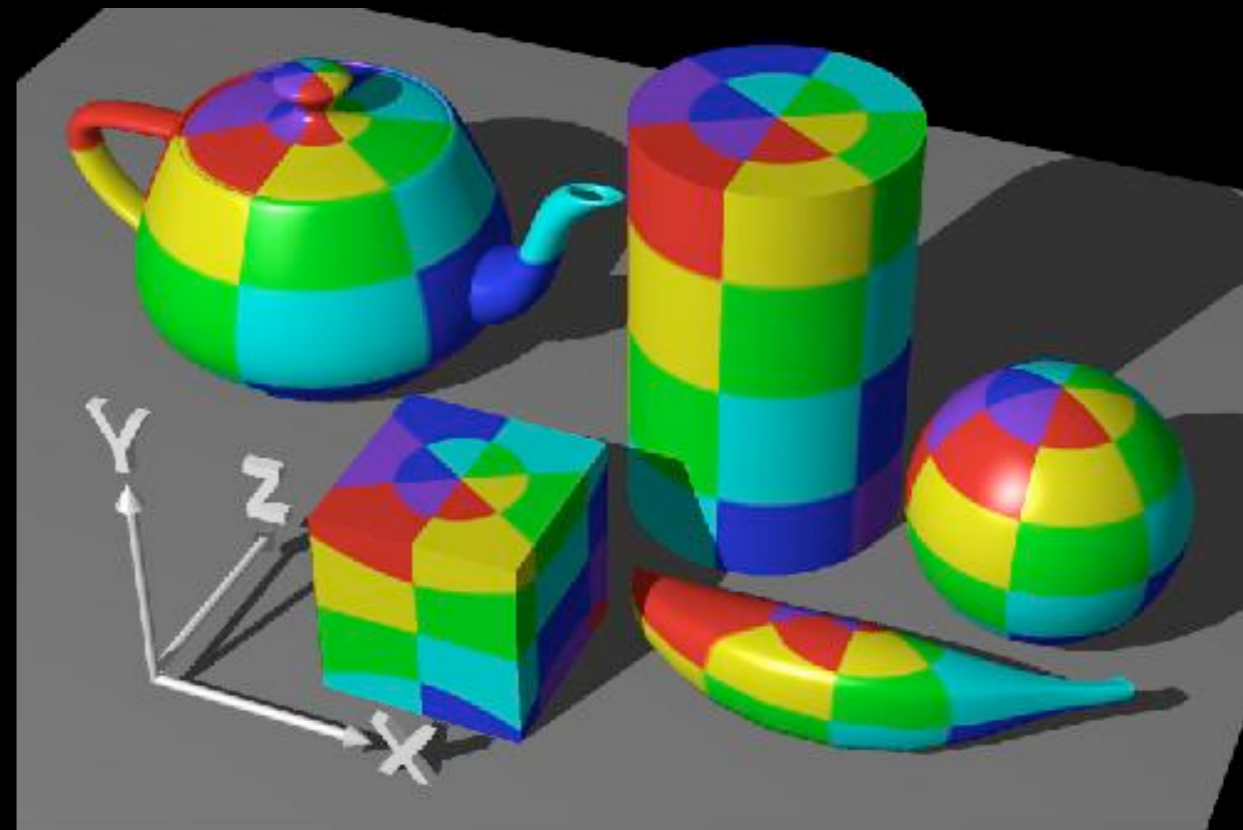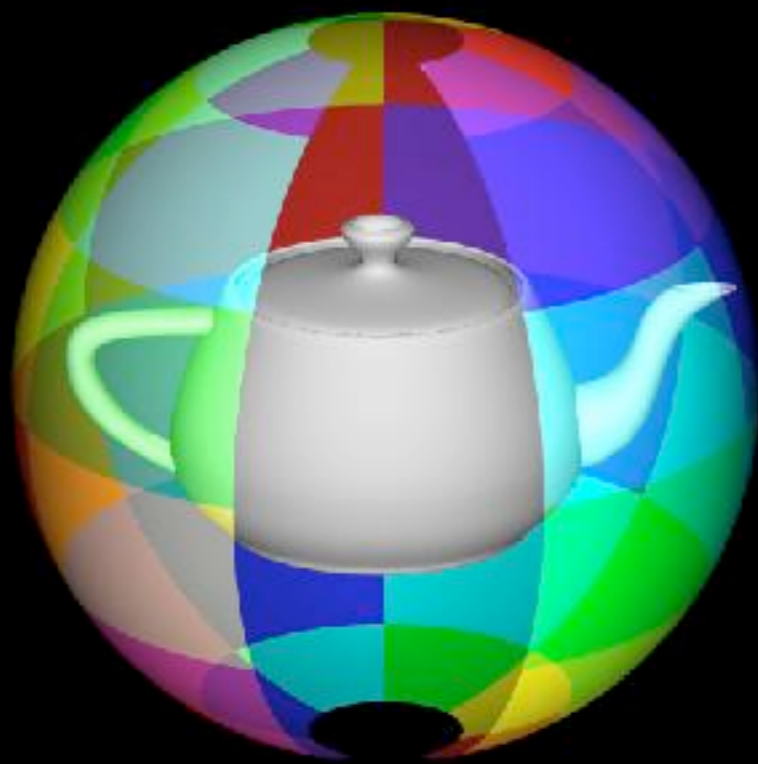

– note "pie slice" phenomena
– which coordinate axis is parallel to the cylinder axis?

Spherical Mapping

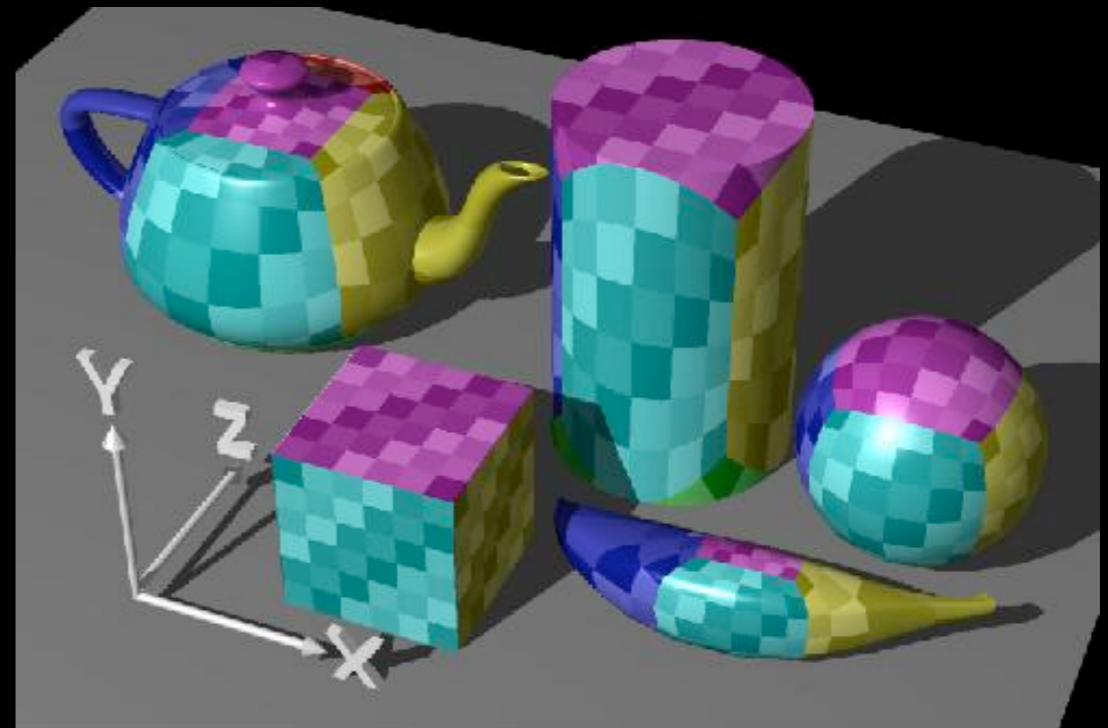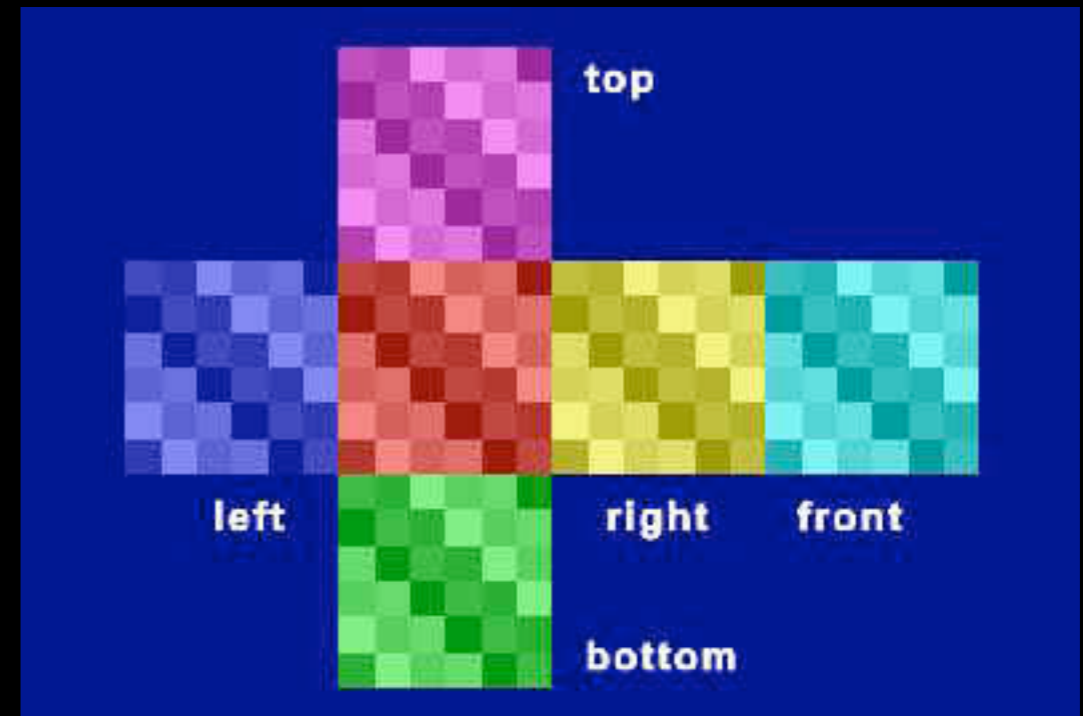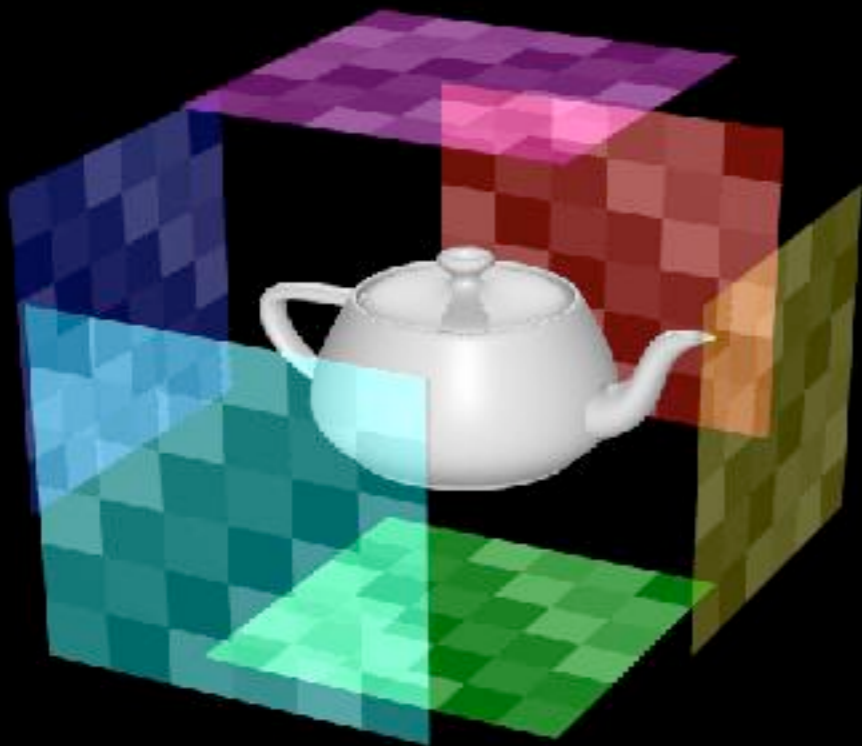$(x,y,z) \rightarrow (latitude, longitude)$
$\rightarrow (u,v)$
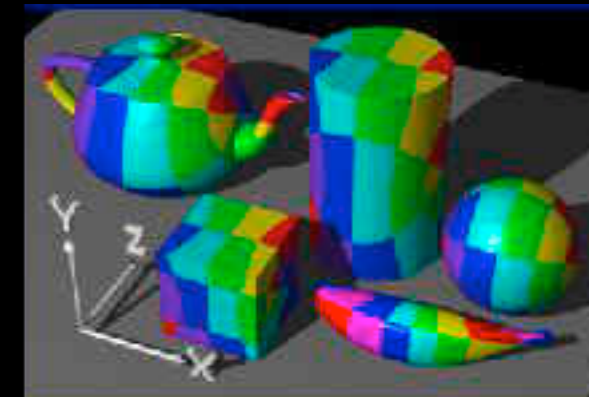
spherical map stretches squares at equator and squeezes squares at poles

Box Mapping

– similar to planar mapping
– planar projection –– choose which plane to project onto

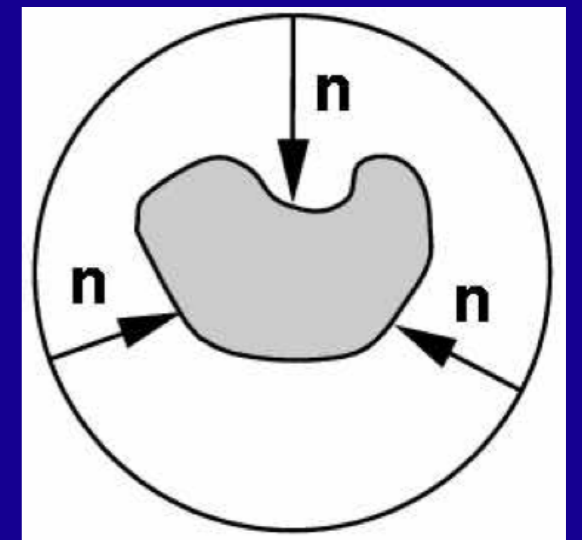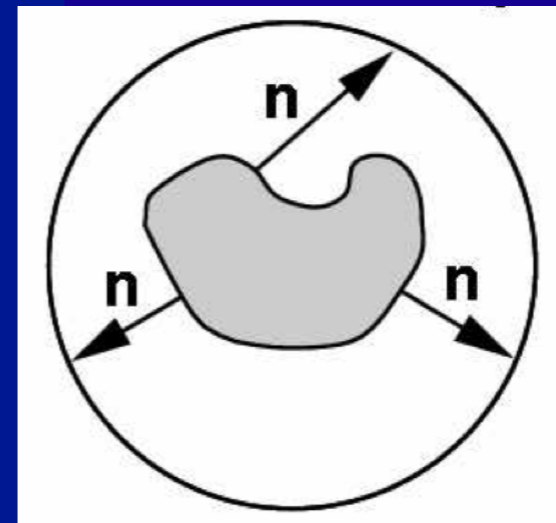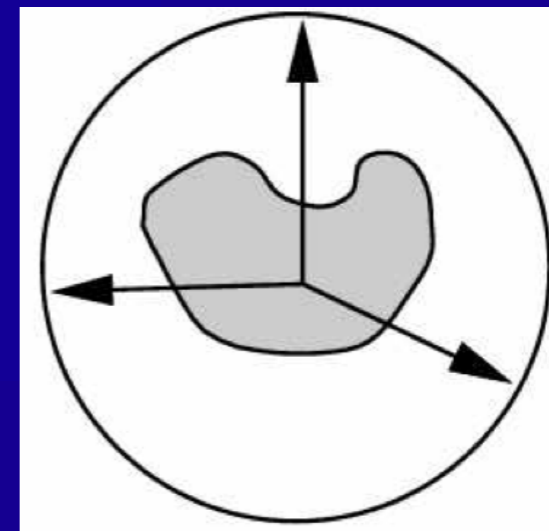# How do we map between intermediate and actual objects?

[Rosalee Wolfe]

position

surface normal

from centroid

reflection

We associated (x,y,z) on the intermediate object with the texture (u,v). But which point on the actual object is this?
We choose both the **intermediate shape** and the **mapping from the actual shape to the intermediate shape**
1. a point on the object relative to its bounding box
2. see where surface normal intersects intermediate surface
3. shoot ray from centroid through surface point to intermediate surface
4. use the reflection vector (depends on the viewer position and normal)

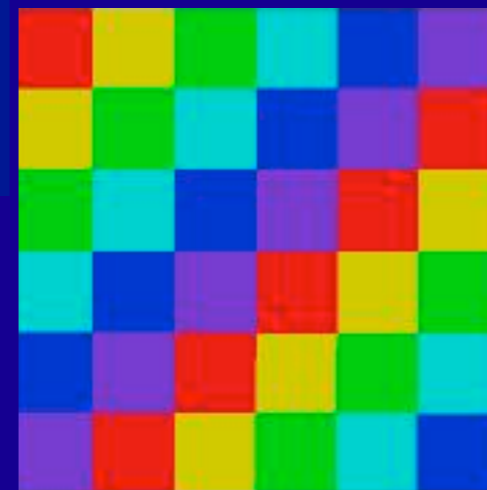# How do we map between intermediate and actual objects?
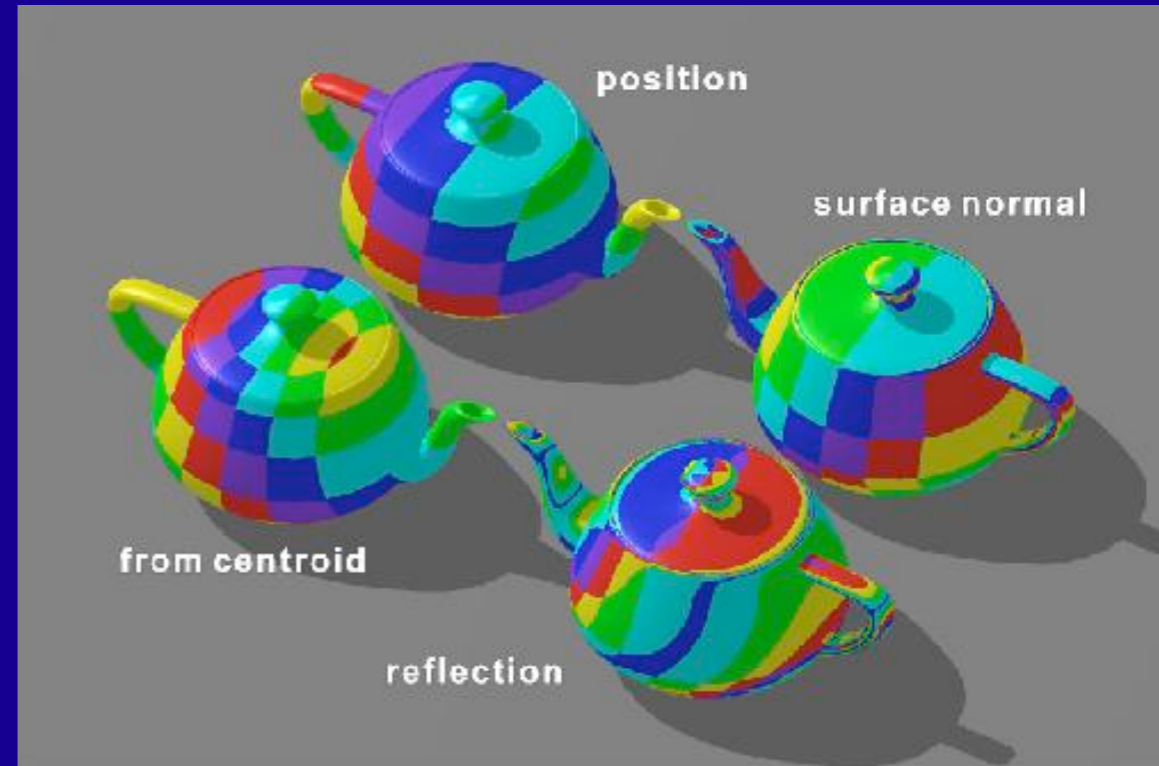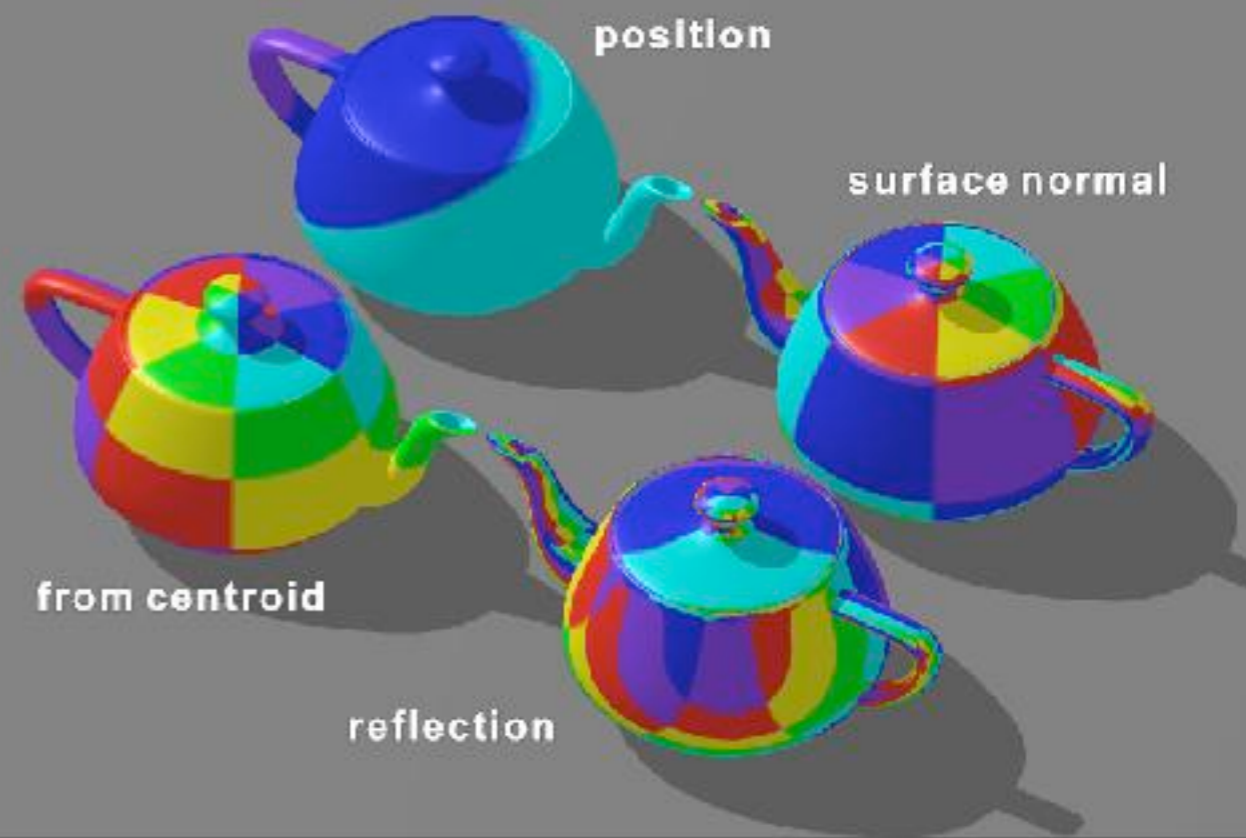

[Rosalee Wolfe]

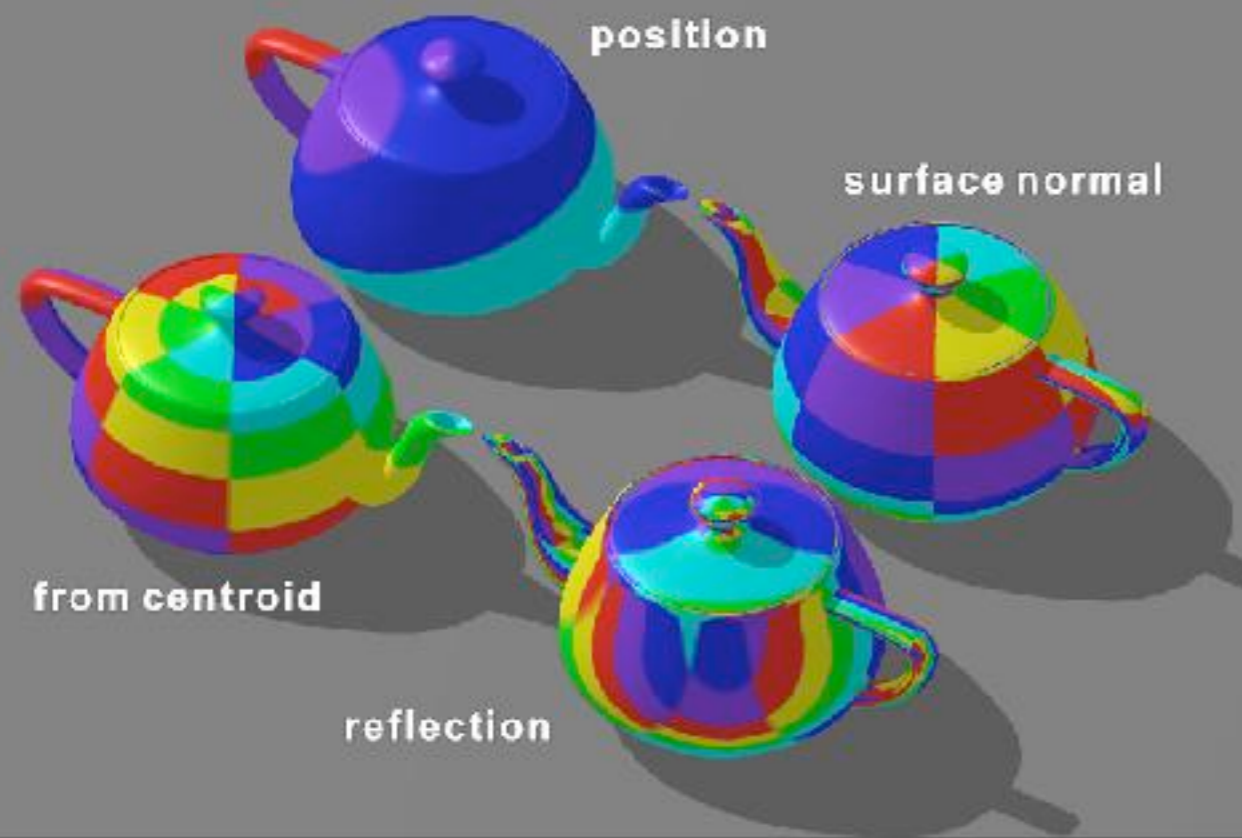position

surface normal

from centroid

reflection

What intermediate shape was used here?
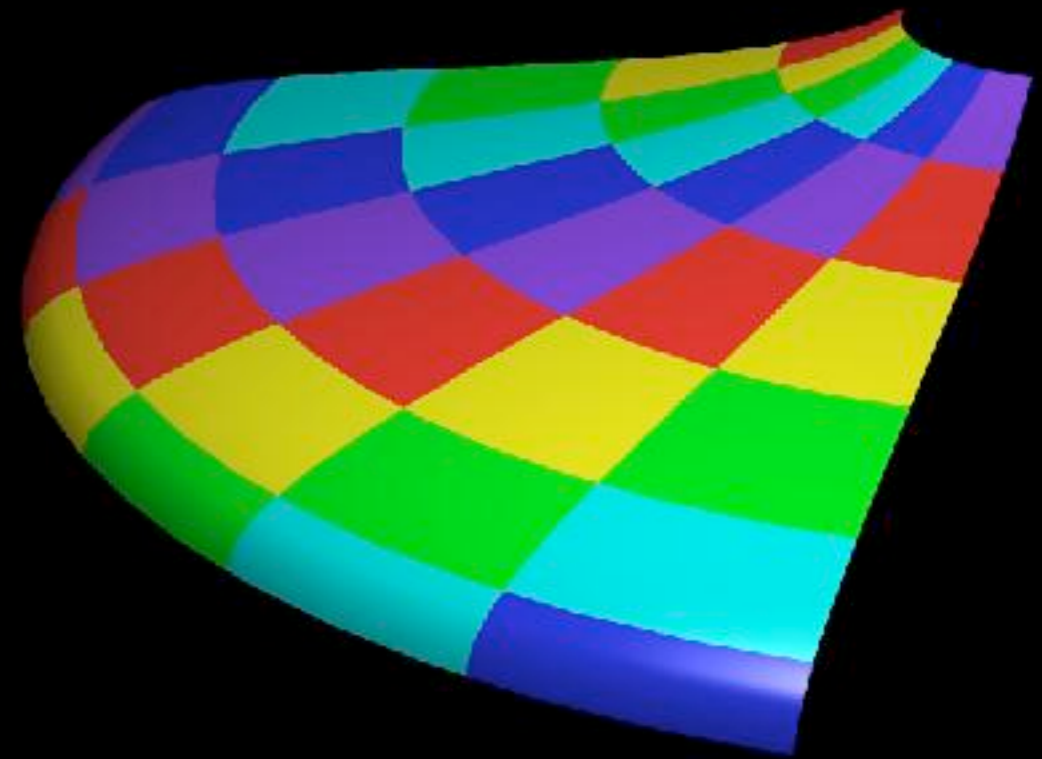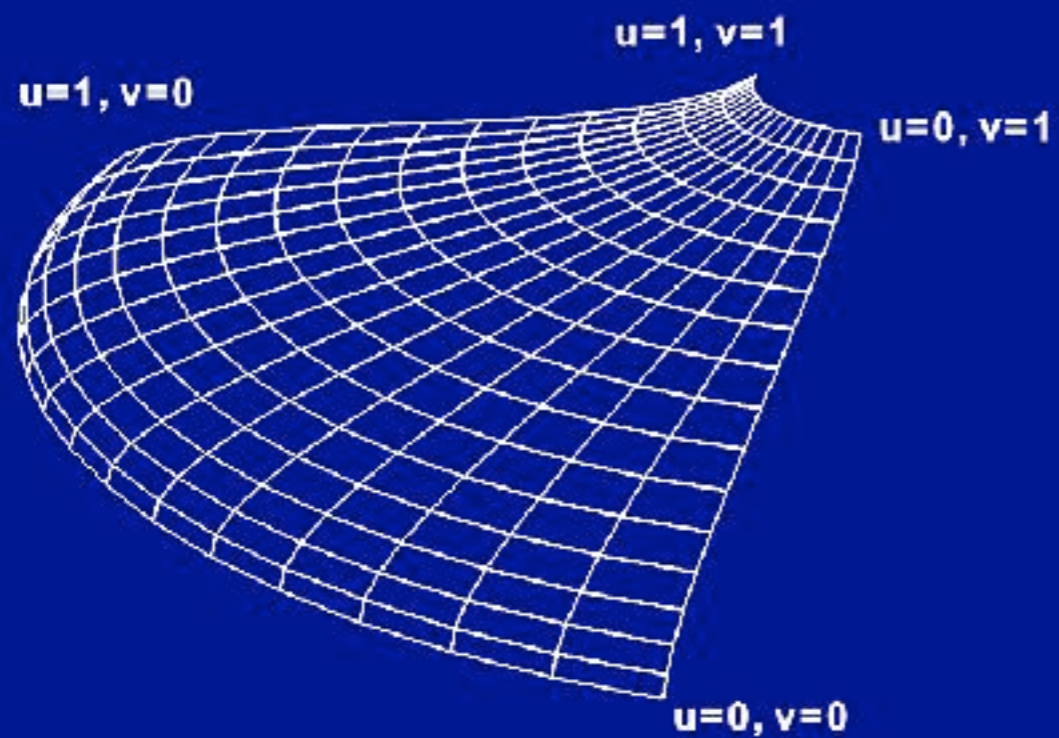
Can you tell what intermediate shape was used?
Planar map - in xy plane

Cylindrical

Spherical

Parametric Surfaces

u=1, v=0

u=1, v=1
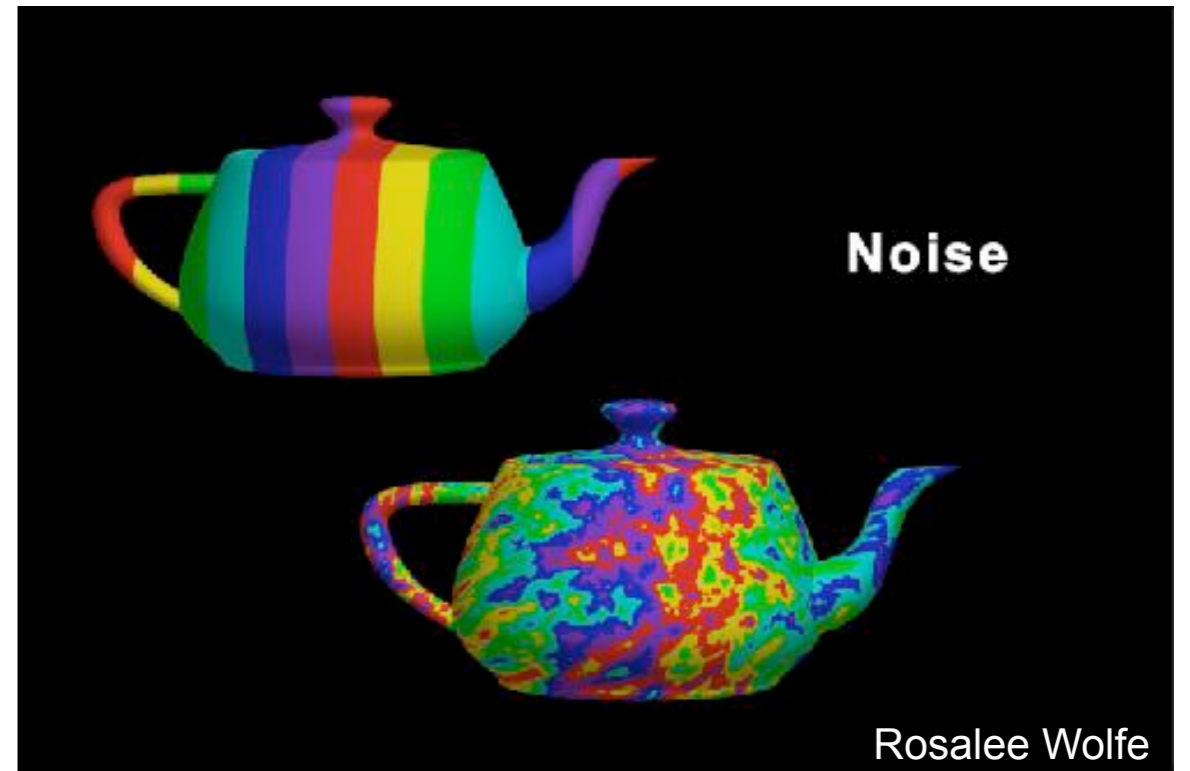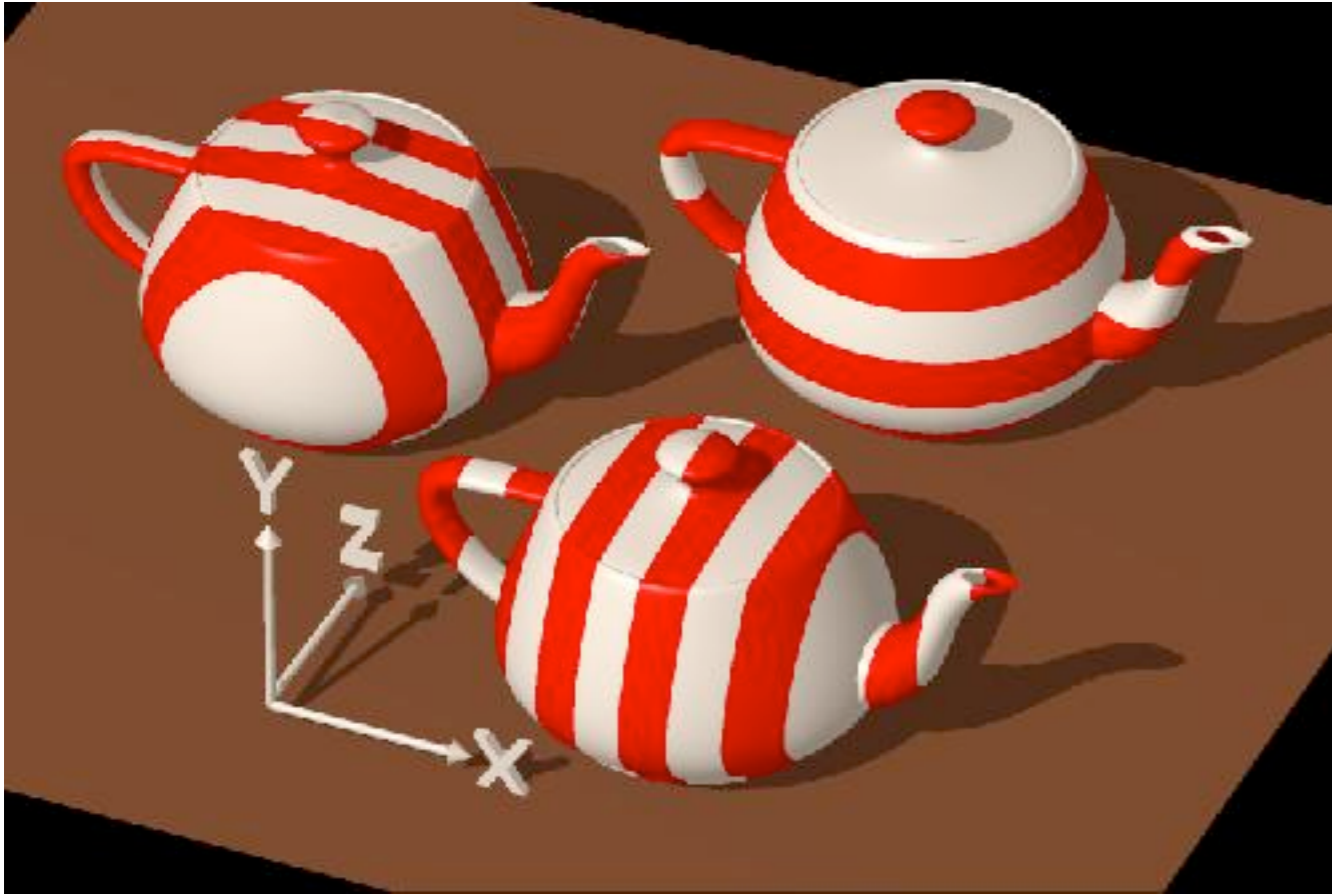
u=0, v=1

u=0, v=0

32 parametric patches

3D solid textures

can map object (x,y,z) directly to texture (u,v,w)

Procedural textures

Noise

Rosalee Wolfe
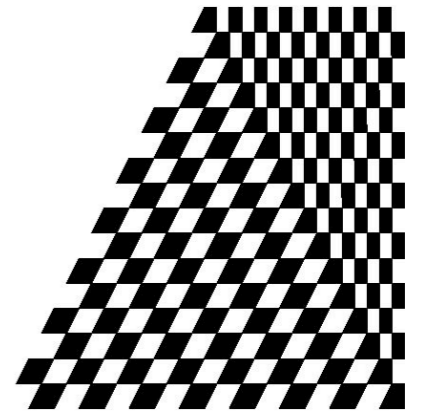
e.g., Perlin noise

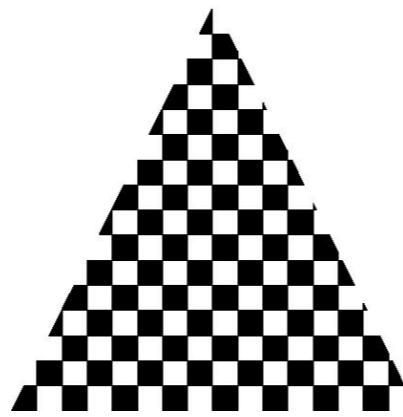# Triangles

Texturing triangles

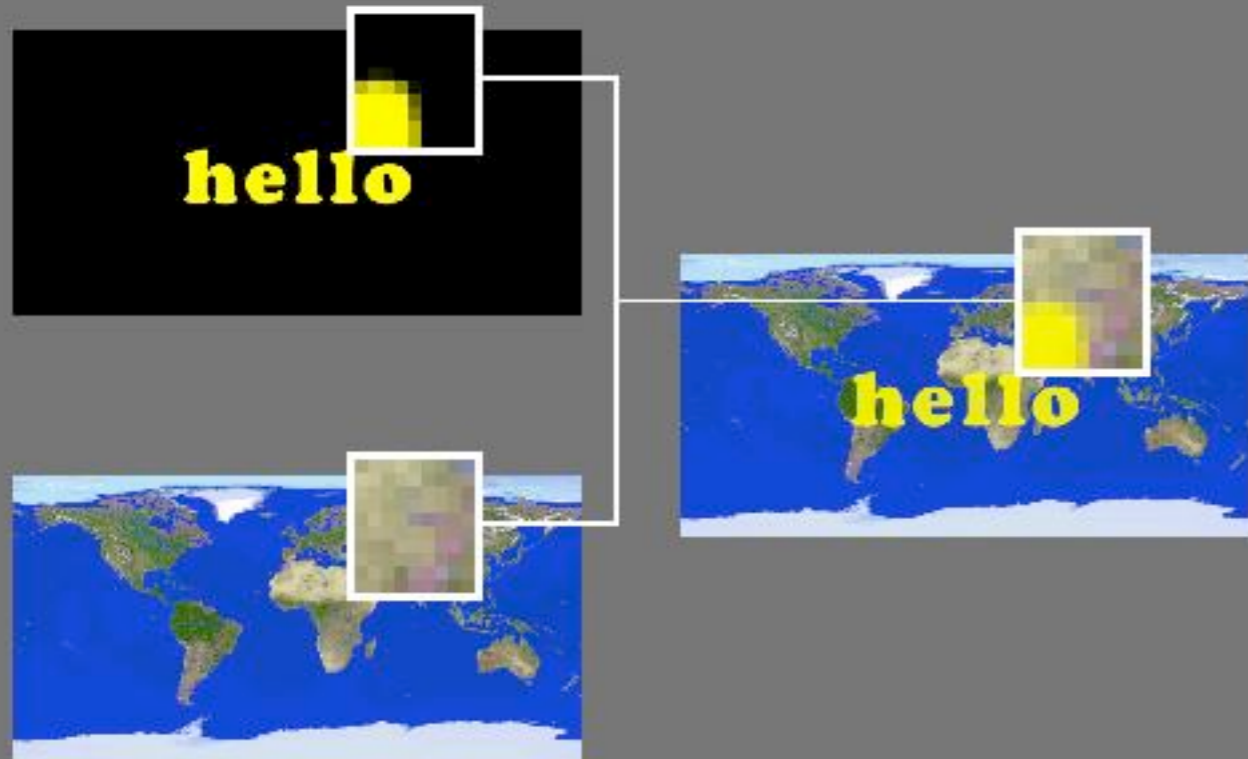Texture Space

Object Space

v

0, 1   1, 1

a

b   c

0, 0   1, 0   u

$(u,v) = (0.2, 0.8)$

A

$(0.4, 0.2)$

B   C

$(0.8, 0.4)$
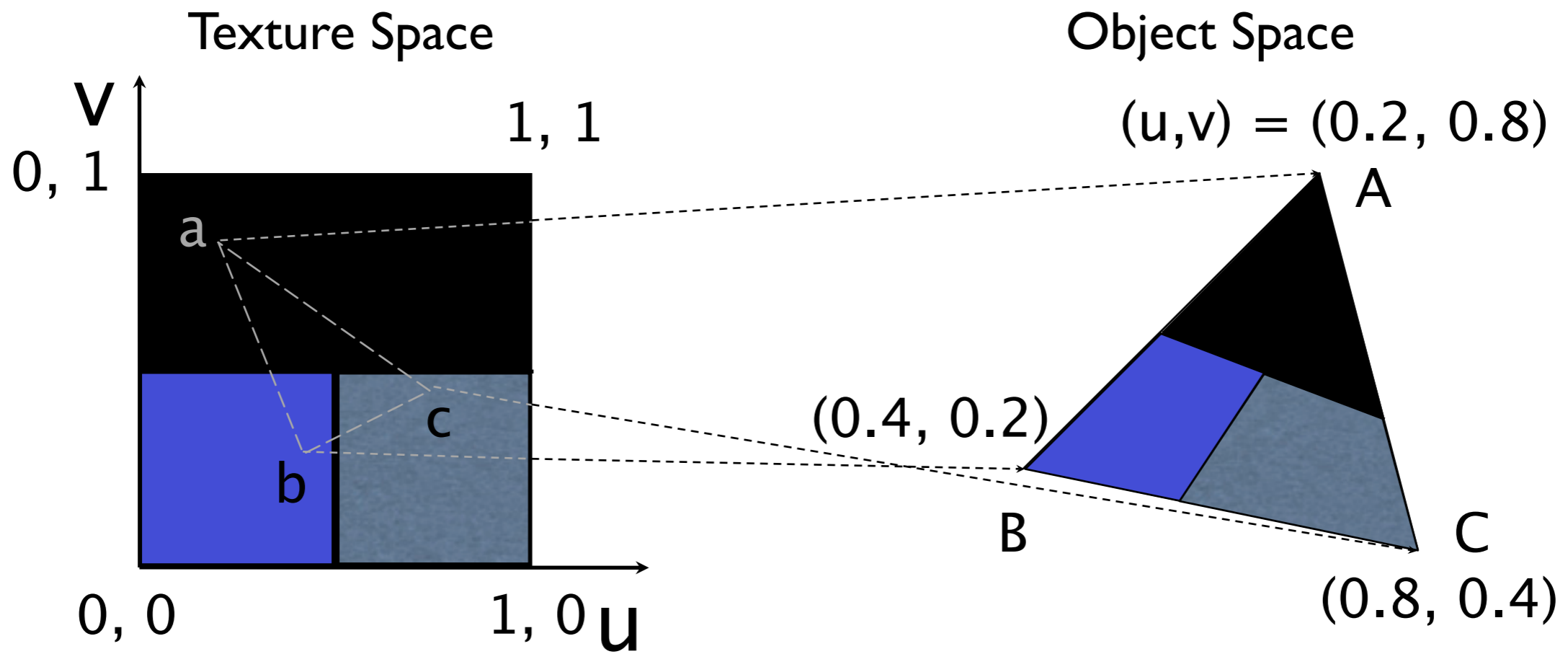
`glTexCoord*()`

Multitexturing

Fragment → Texture unit 0 → Texture unit 1 → Texture unit 2 → Frame buffer
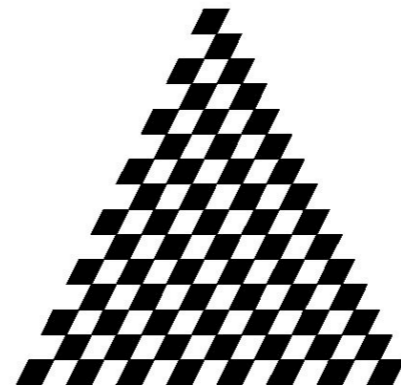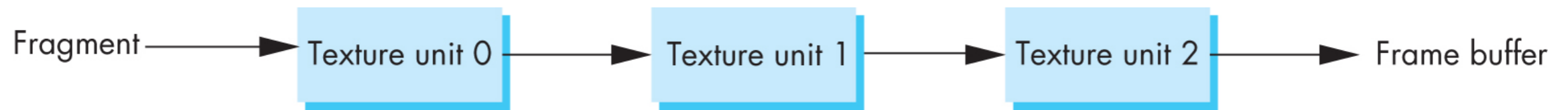
Rosalee Wolfe

Texture Sampling

Texture Mapping

Texels → Pixels
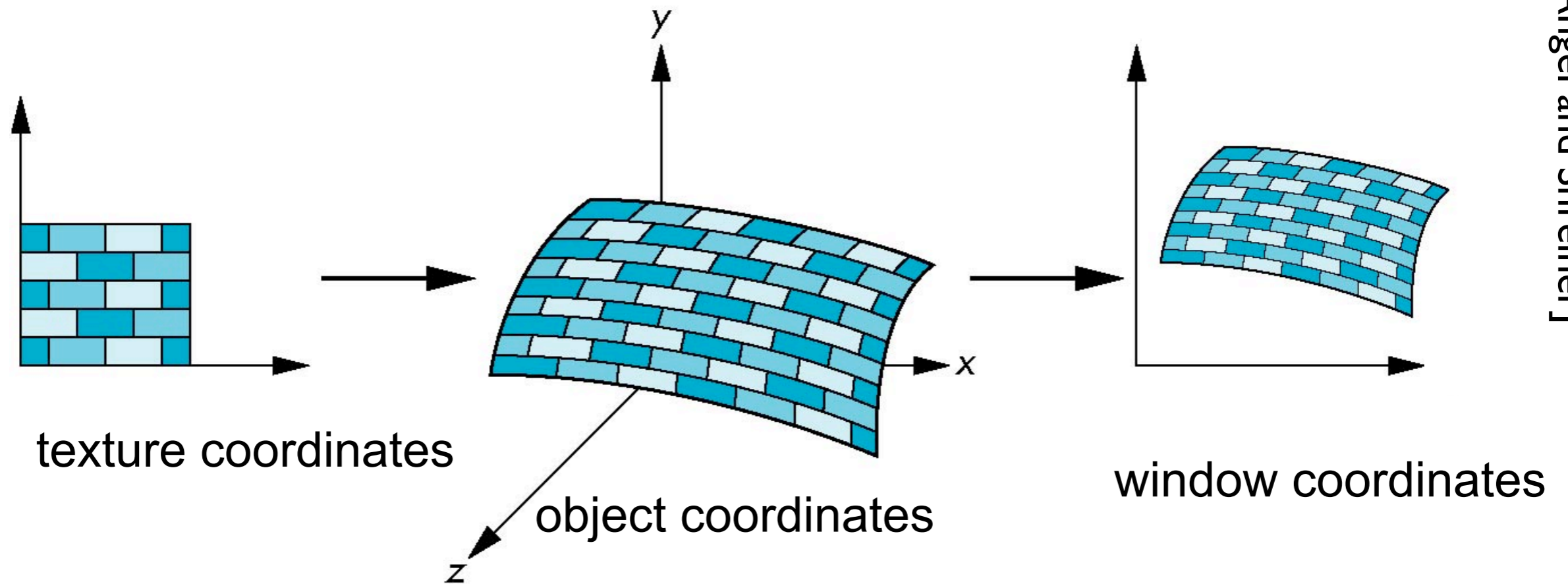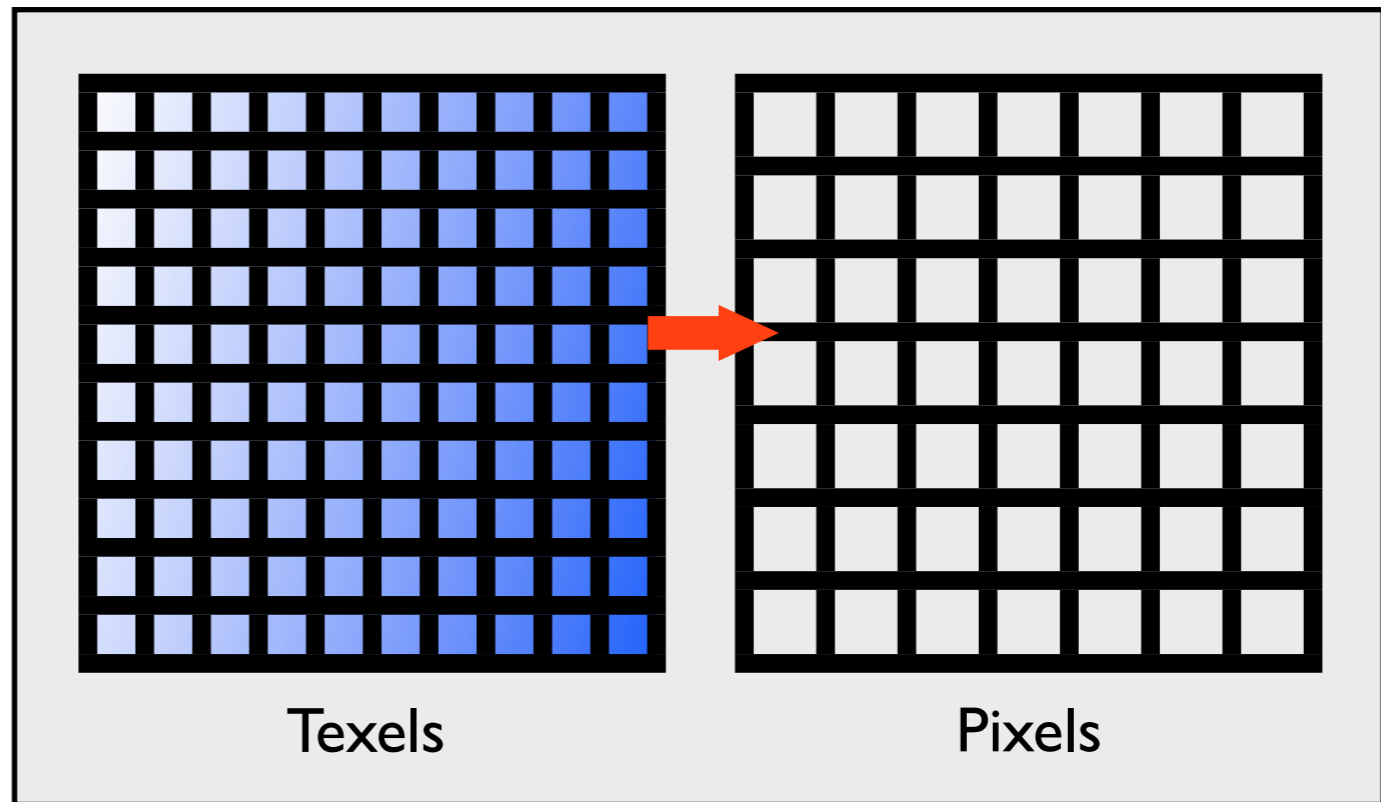
[Angel and Shreiner]

texture coordinates

object coordinates

window coordinates
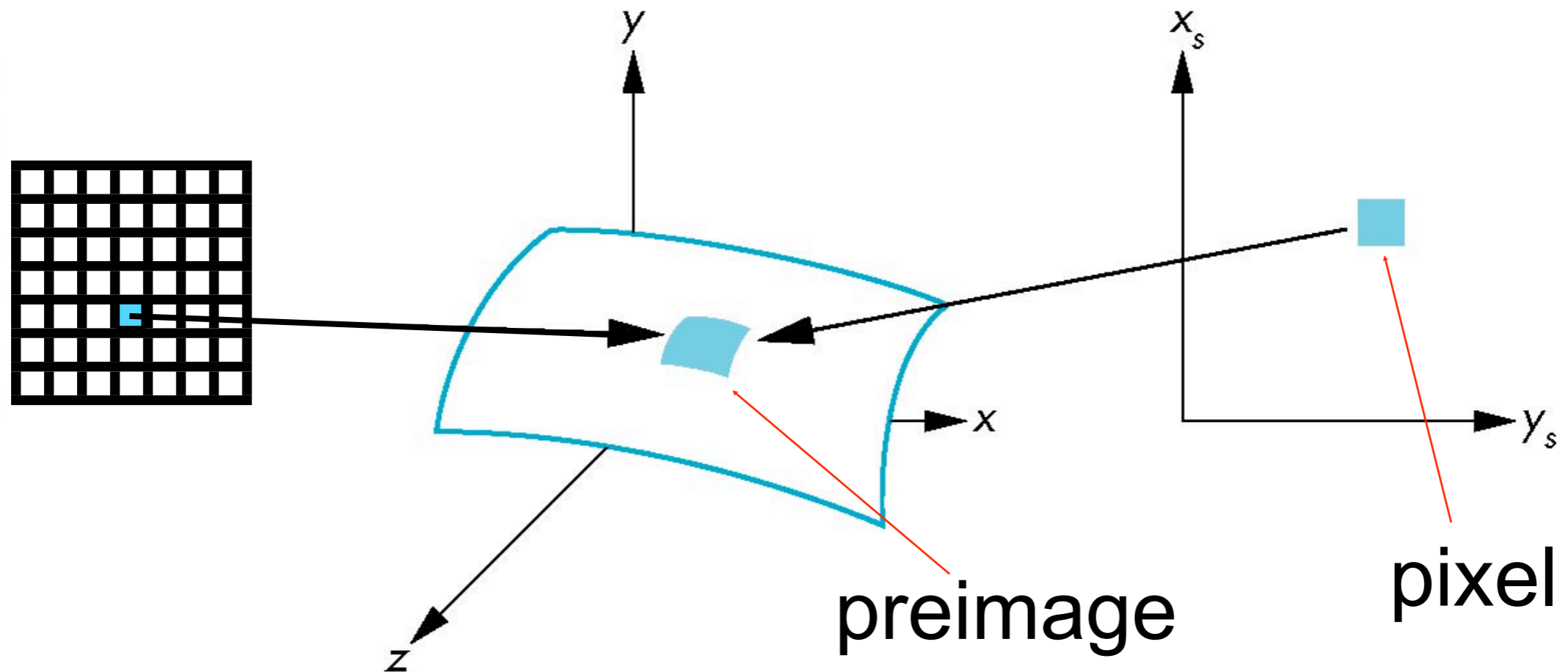
- Texture coordinates: Used to identify points in the image to be mapped
- Object Coordinates: Conceptually, where the mapping takes place
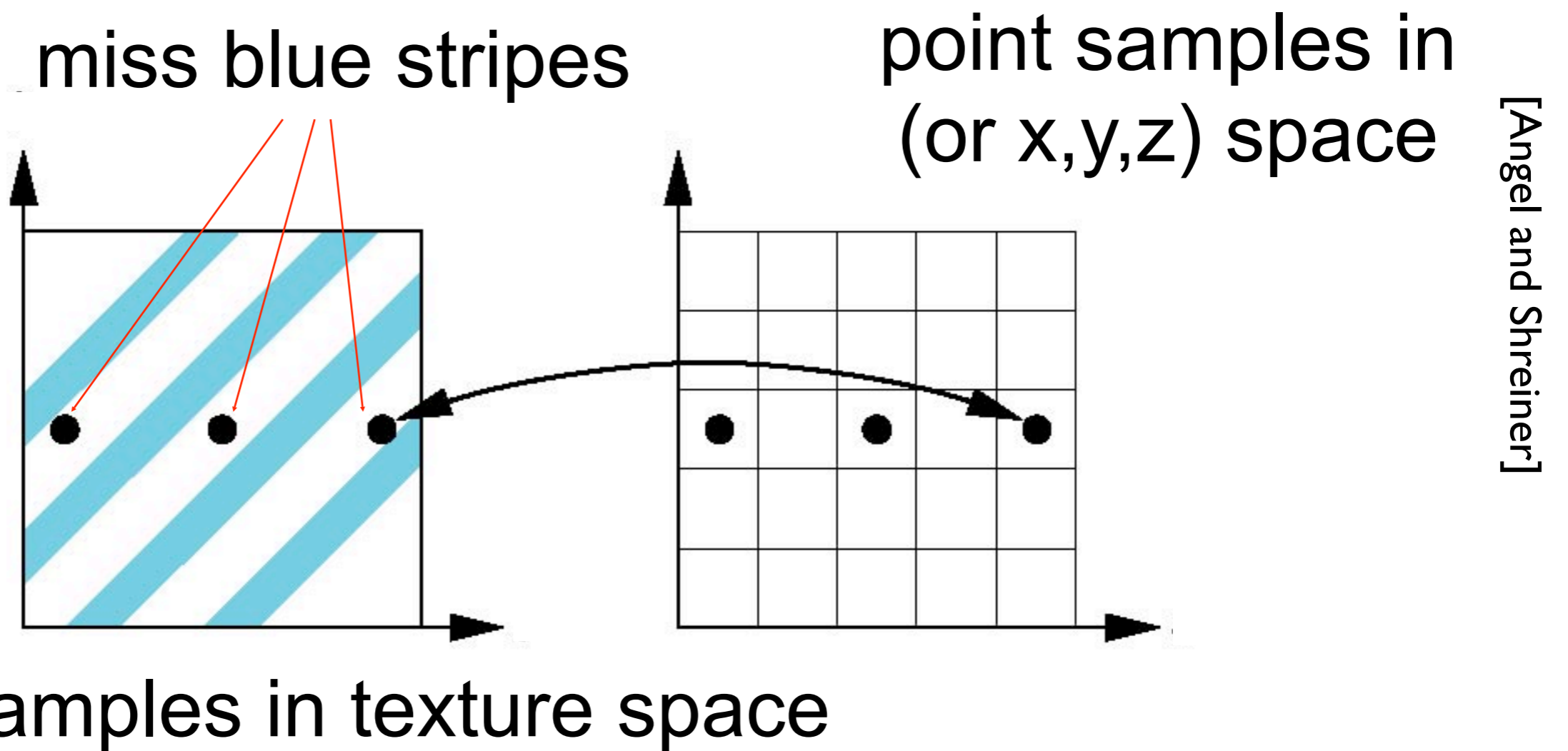- Window Coordinates: Where the final image is really produced

Point Sampling

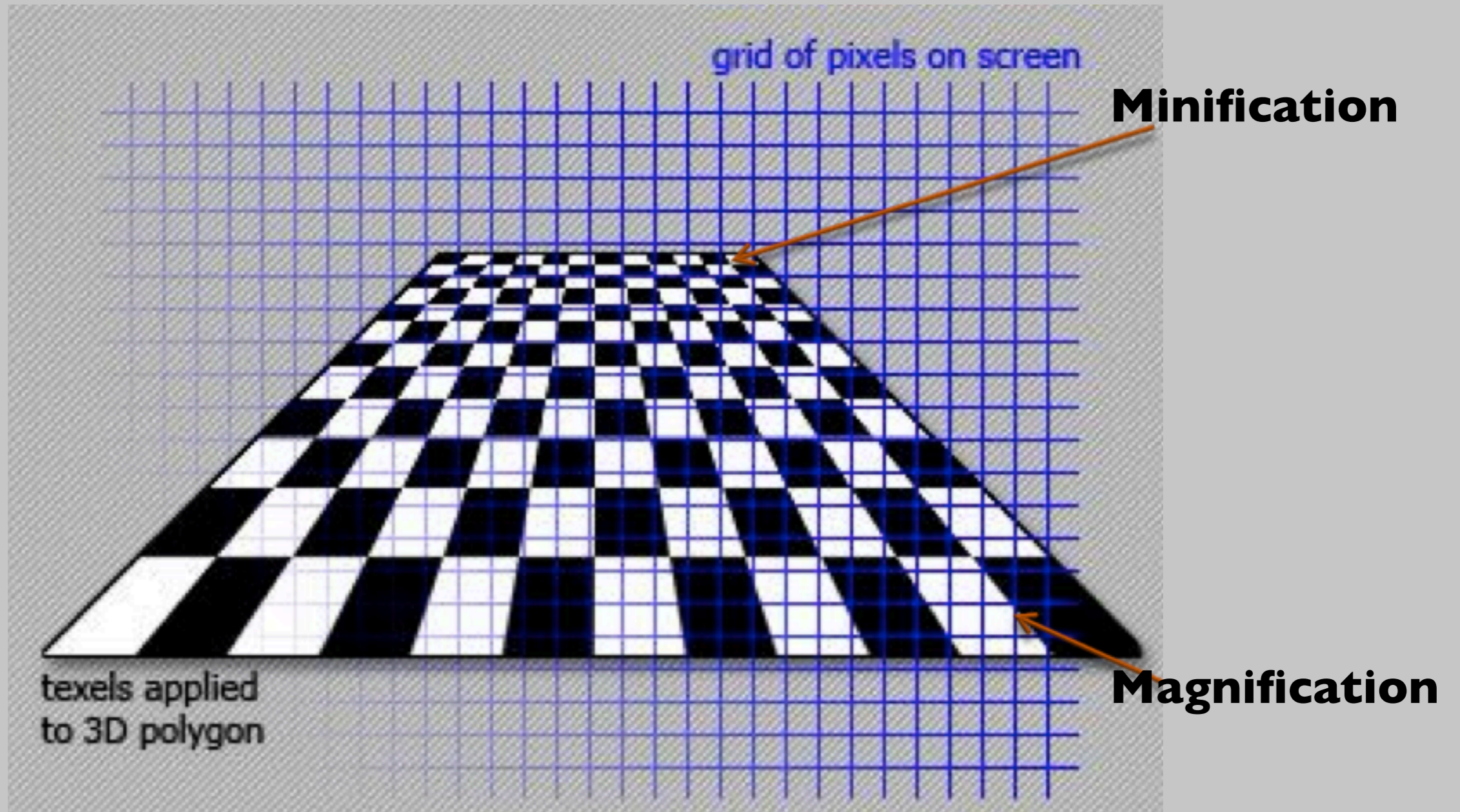Map back to texture image and use the **nearest texel**

preimage

pixel

Aliasing

Point sampling of the texture can lead to aliasing artifacts

miss blue stripes

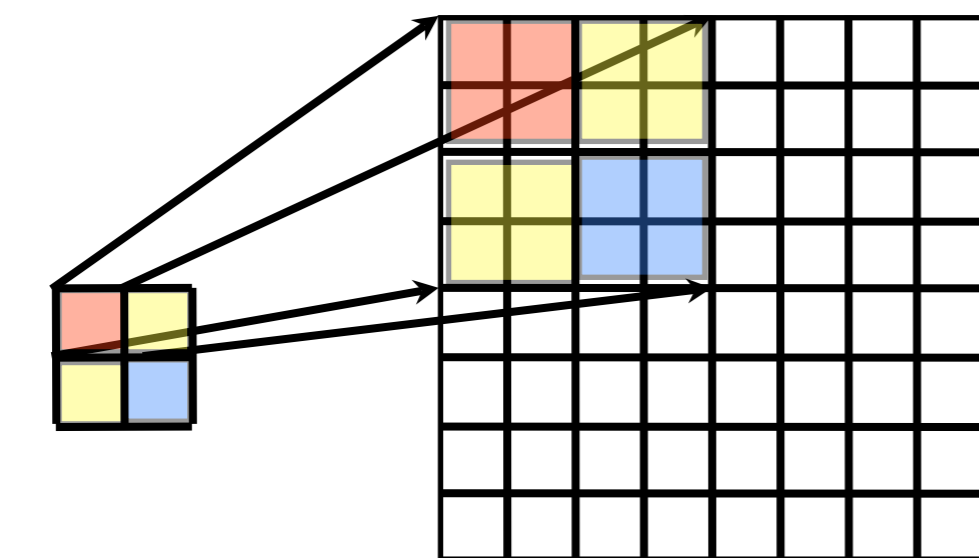point samples in (or x,y,z) space

point samples in texture space

Magnification and Minification

Magnification and Minification

More than one texel can cover a pixel (*minification*) or more than one pixel can cover a texel (*magnification*)
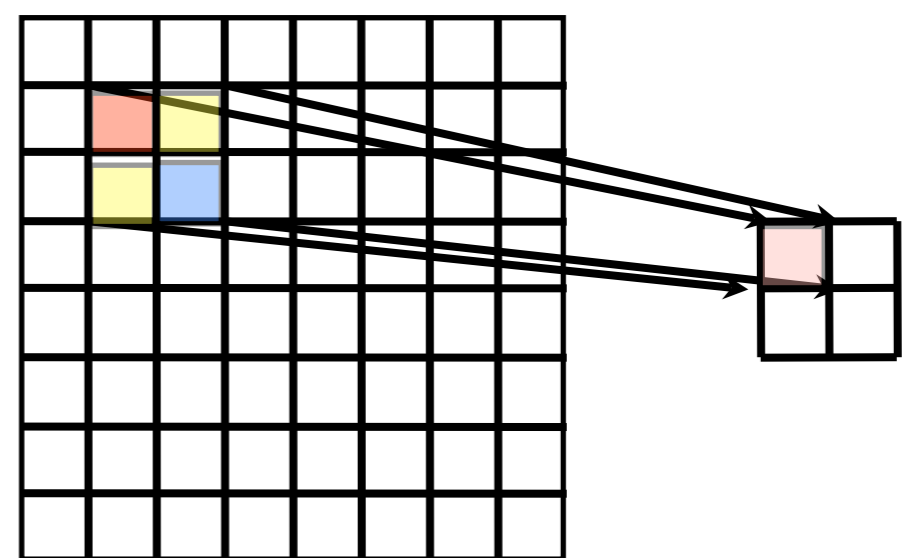
Can use point sampling (nearest texel) or linear filtering (2 x 2 filter) to obtain texture values
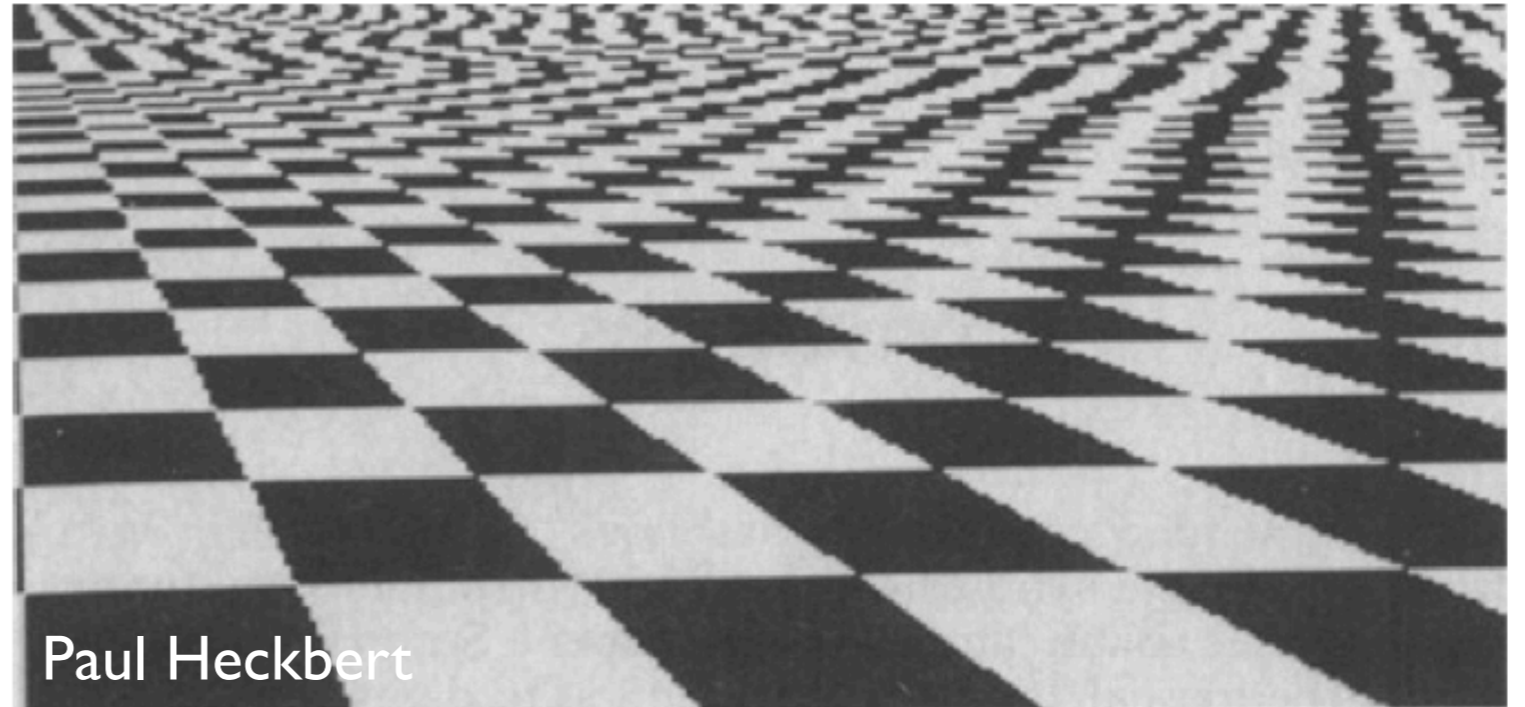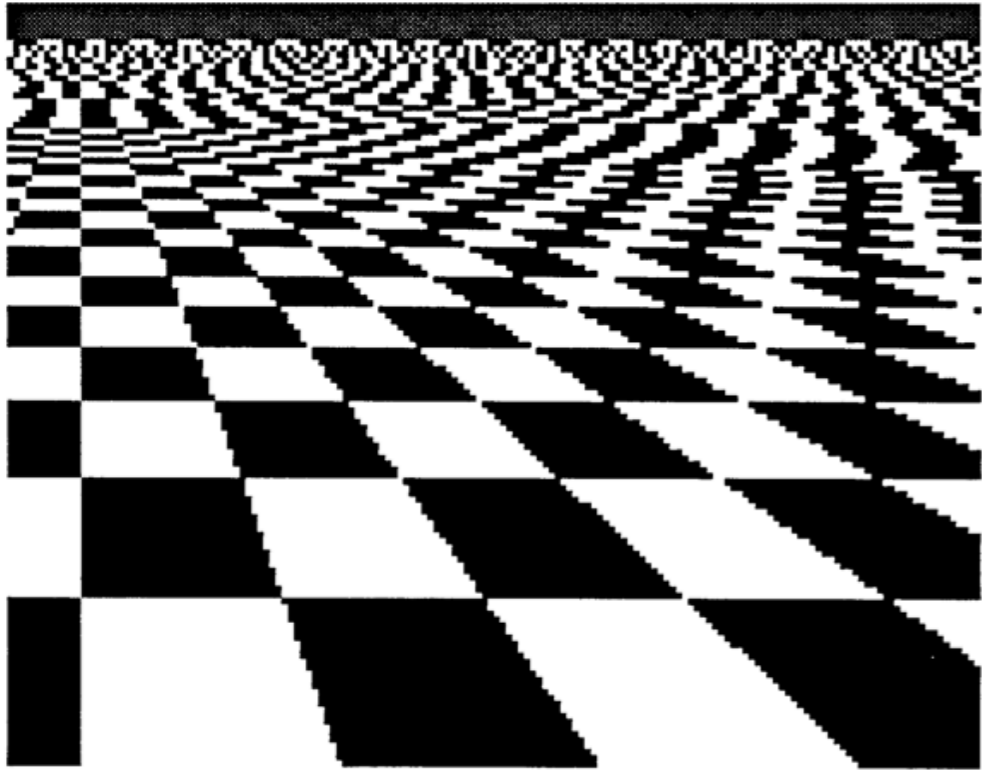

Texture   Pixels   Texture   Pixels
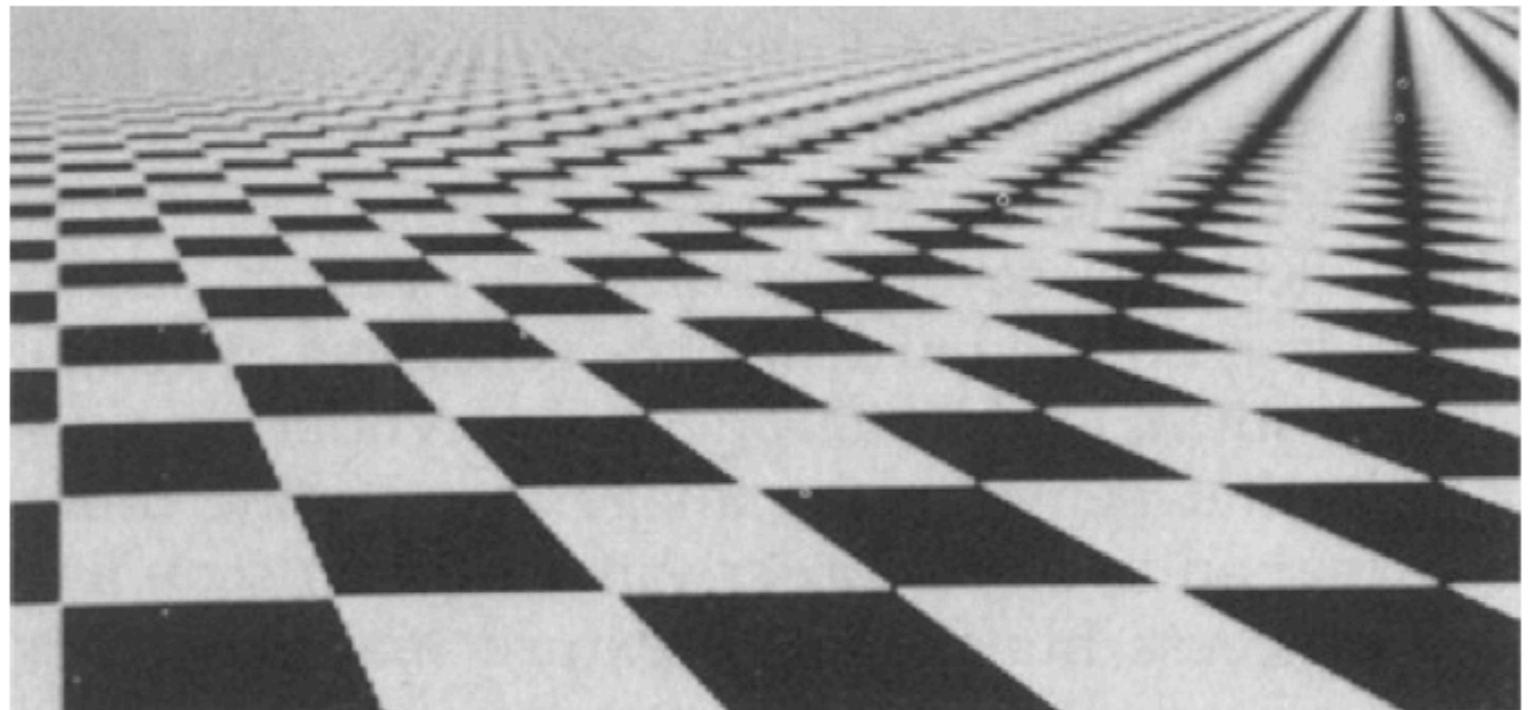
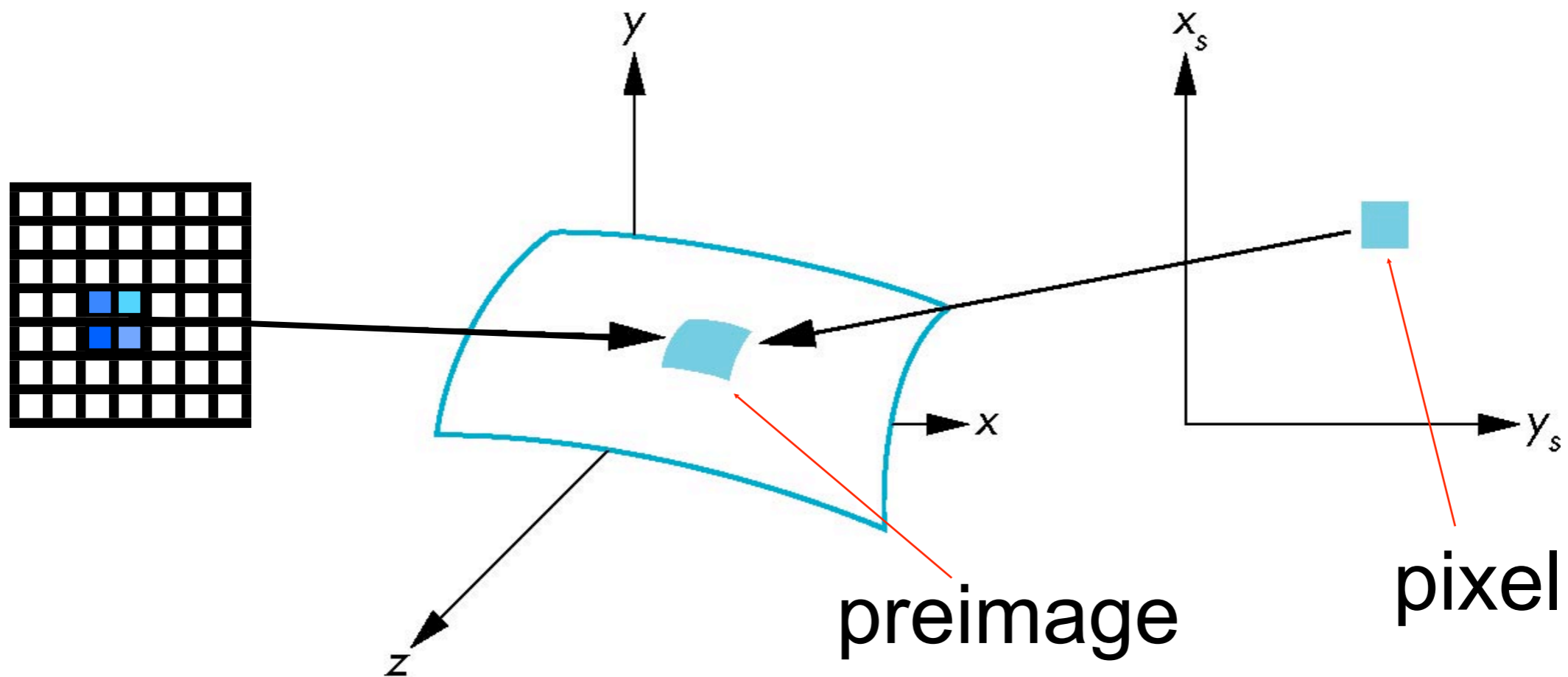**Magnification**   **Minification**

Aliasing artifacts

Paul Heckbert

We apply **filtering** to reduce aliasing artifacts

Area Averaging

A better but slower option is to use **area averaging**

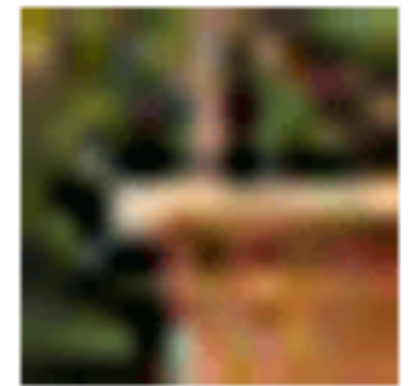preimage

pixel

[Angel and Shreiner]

Use bilinear filtering

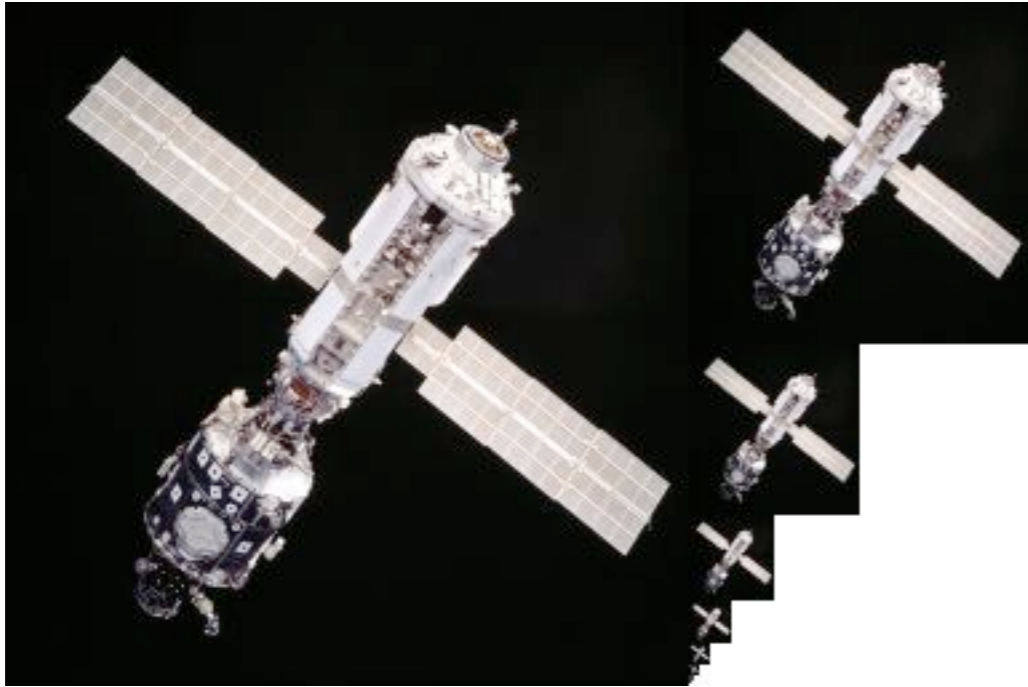p3 ● P ● p2

p0 ● ● p1
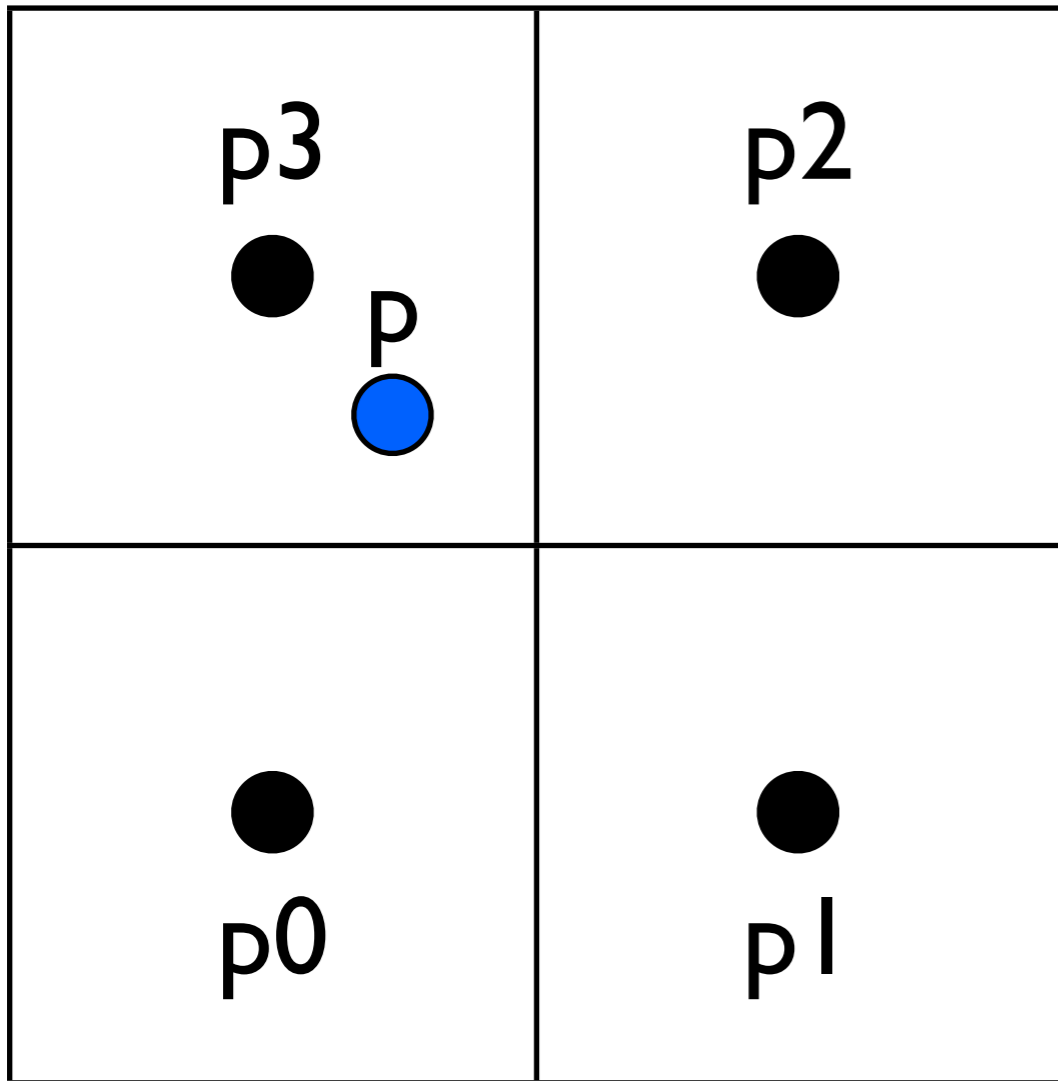
p = ?

nearest
neighbor

bilinear

bicubic

Wikipedia

mitigate magnification artifacts

smooths out the texture – no sharp boundaries
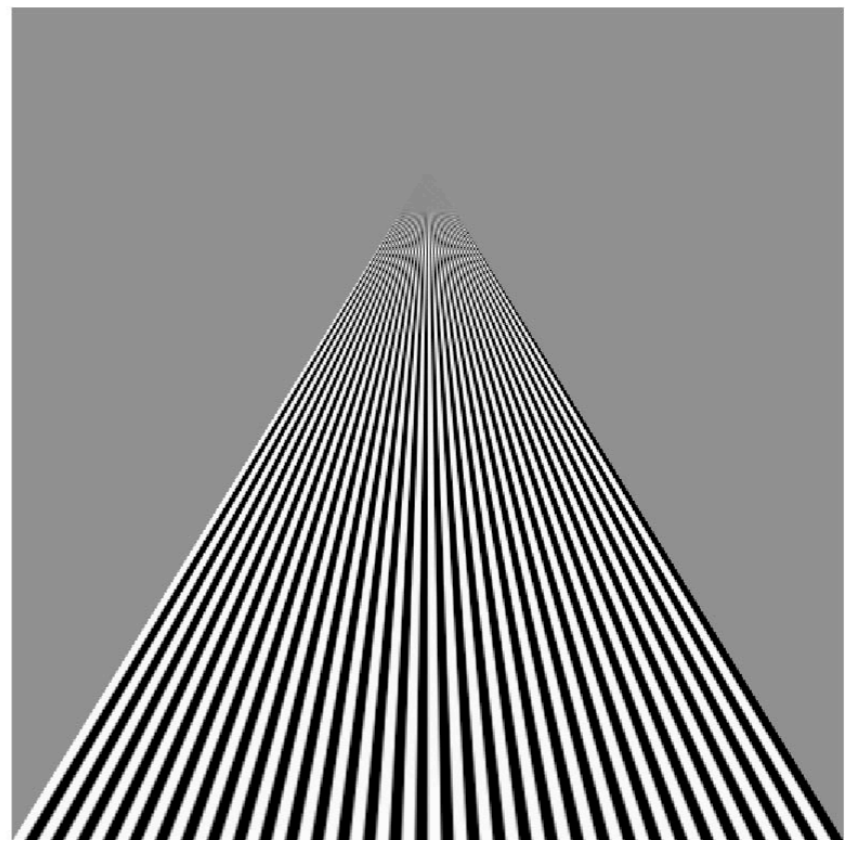
Mipmapping

Togikun, Wikimedia Commons

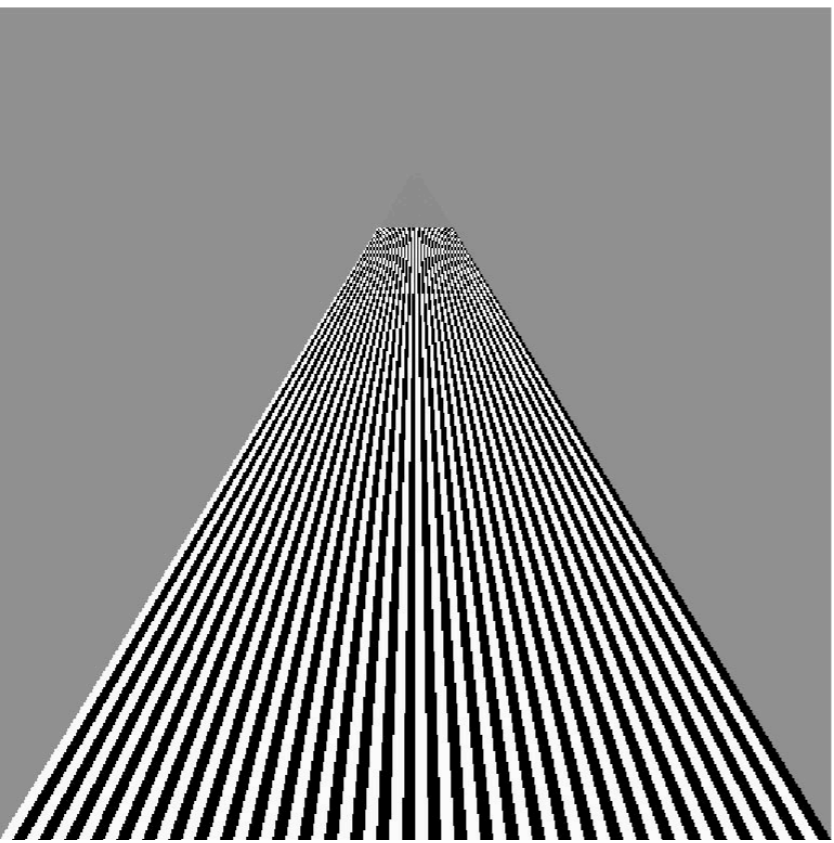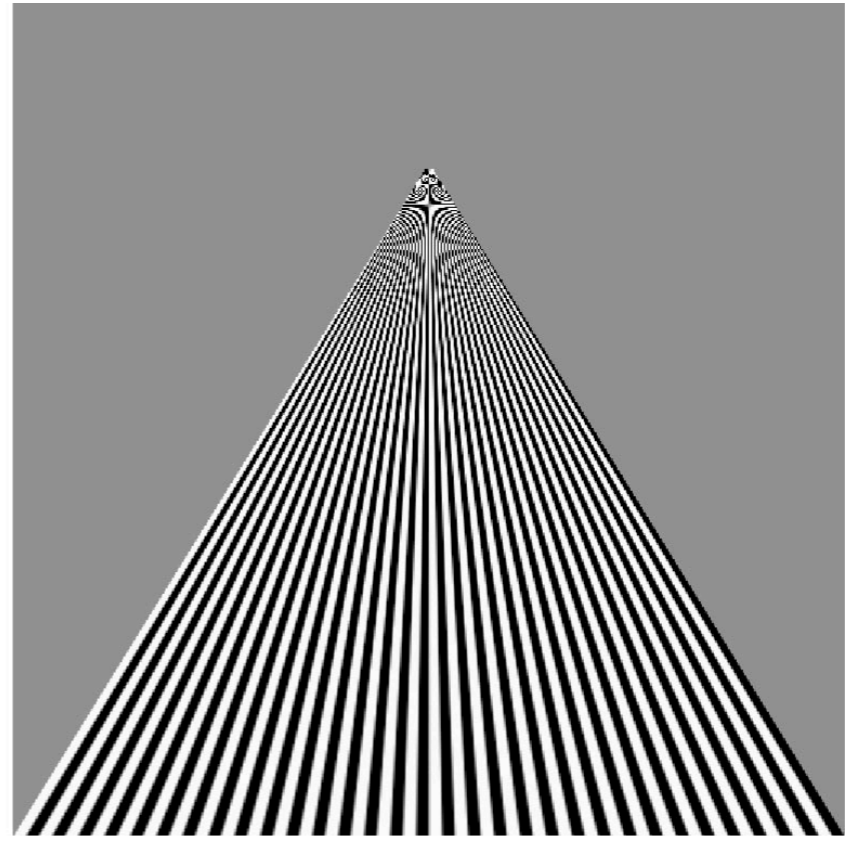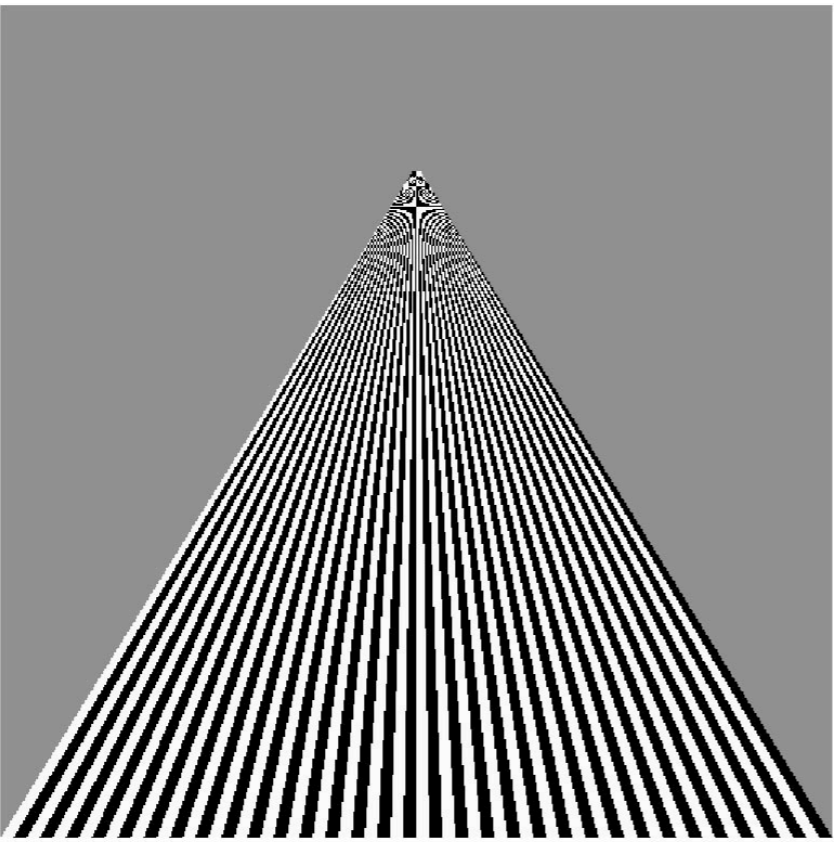128×128, 64×64, 32×32, 16×16, 8×8, 4×4, 2×2, 1×1

Reduce minification artifacts

Prefilter the texture to obtain reduced resolutions

Requires 1/3 more space
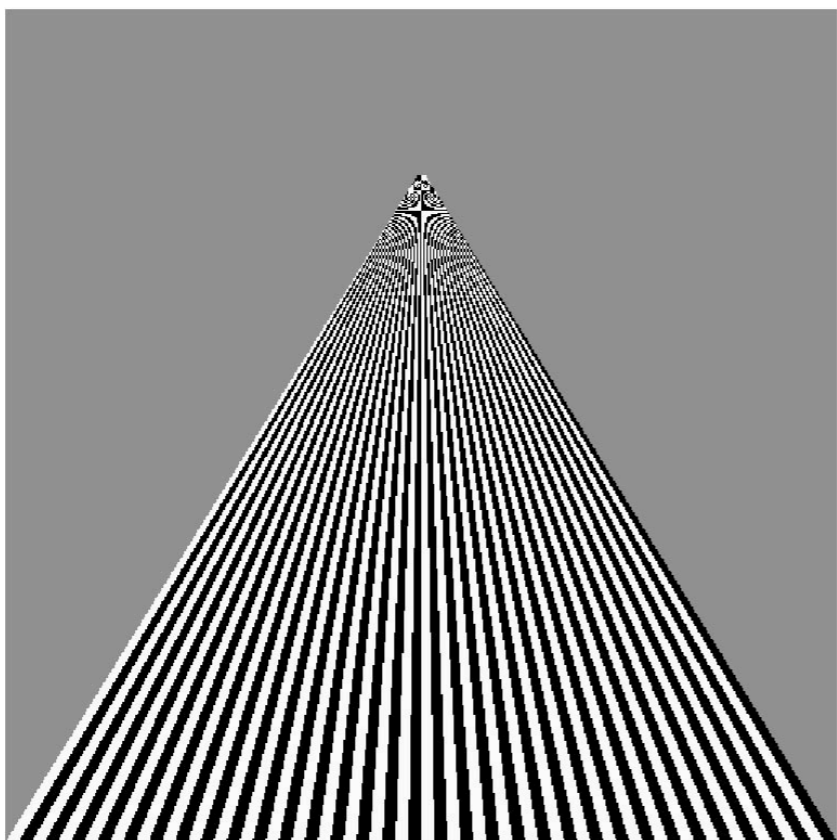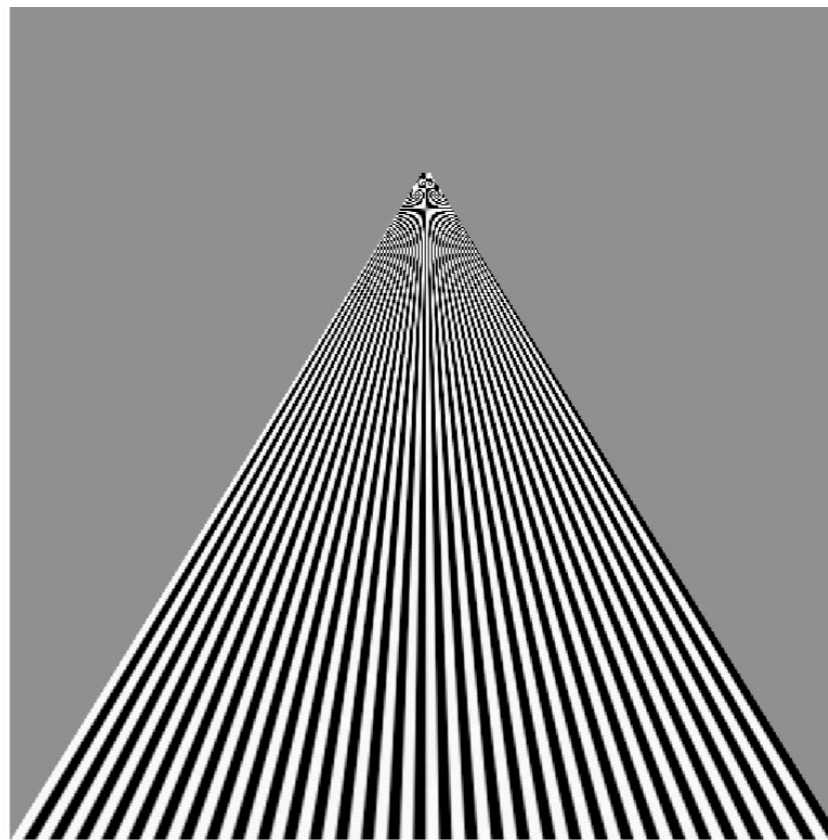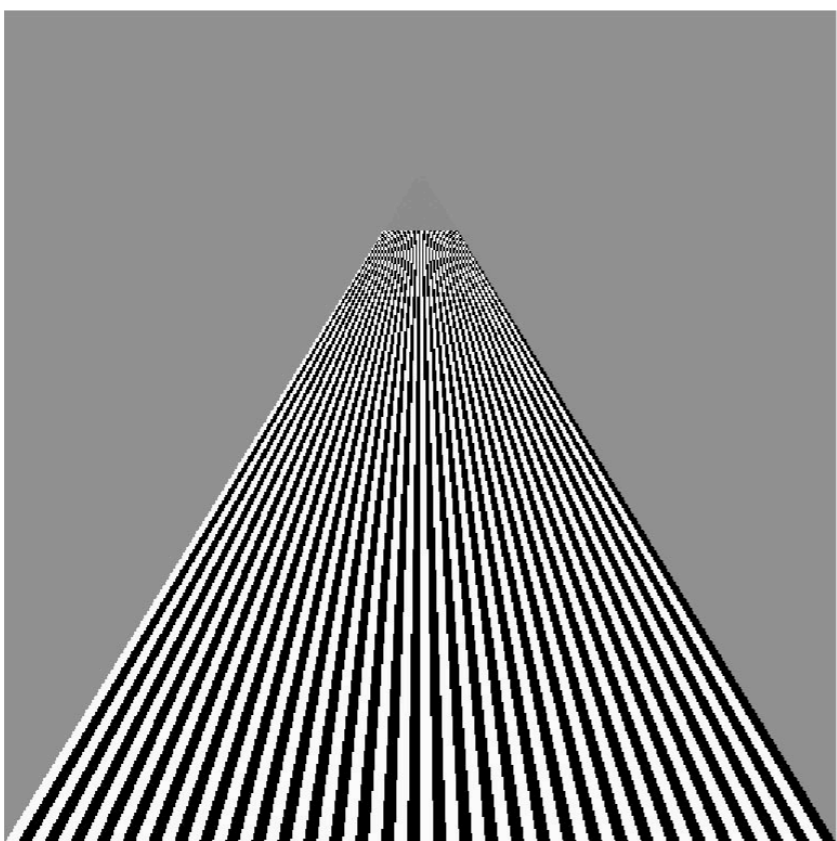
Get a texture hierarchy indexed by level

point
sampling

linear
filtering

mipmapped
point
sampling

mipmapped
linear
filtering

[Angel and Shreiner]

17