CG – T10 – Textures

L:CC, MI:ERSI

Miguel Tavares Coimbra

(course and slides designed by Verónica Costa Orvalho)

introduction

textures are a way to
add detail to a surface

how?

1. model the surface with more polygons
    . it is hard to model subtle details
    . more surface details, more rendering speed

2. map a texture to the surface
    . allows including more detail on the surface
      without affecting the rendering speed

Images from http://www.cgtextures.com

how?

1. model the surface with more polygons
     . it is hard to model subtle details
     . more surface details, more rendering speed

2. map a texture to the surface

     . allows including more detail on the surface
   without affecting the rendering speed

what is a texture?

. a bitmap image <u>NOT</u> vector
- screen: 100 dpi aprox; print: 150-300dpi
- pixels in a grid
- resolution dependent
- resizing reduces quality (resample)
- easily converted
- restricted to rectangle
- minimal support for transparency

. store in a 2D array: texture[height][width]
. defined by <u>texels</u>: pixels of the texture
. texels coordinates: $(s,t) \in [0,1]$

bitmap formats

- BMP
- GIF: support transparency
- JPEG, JPG
- PNG: support transparency
- PICT (Macintosh)
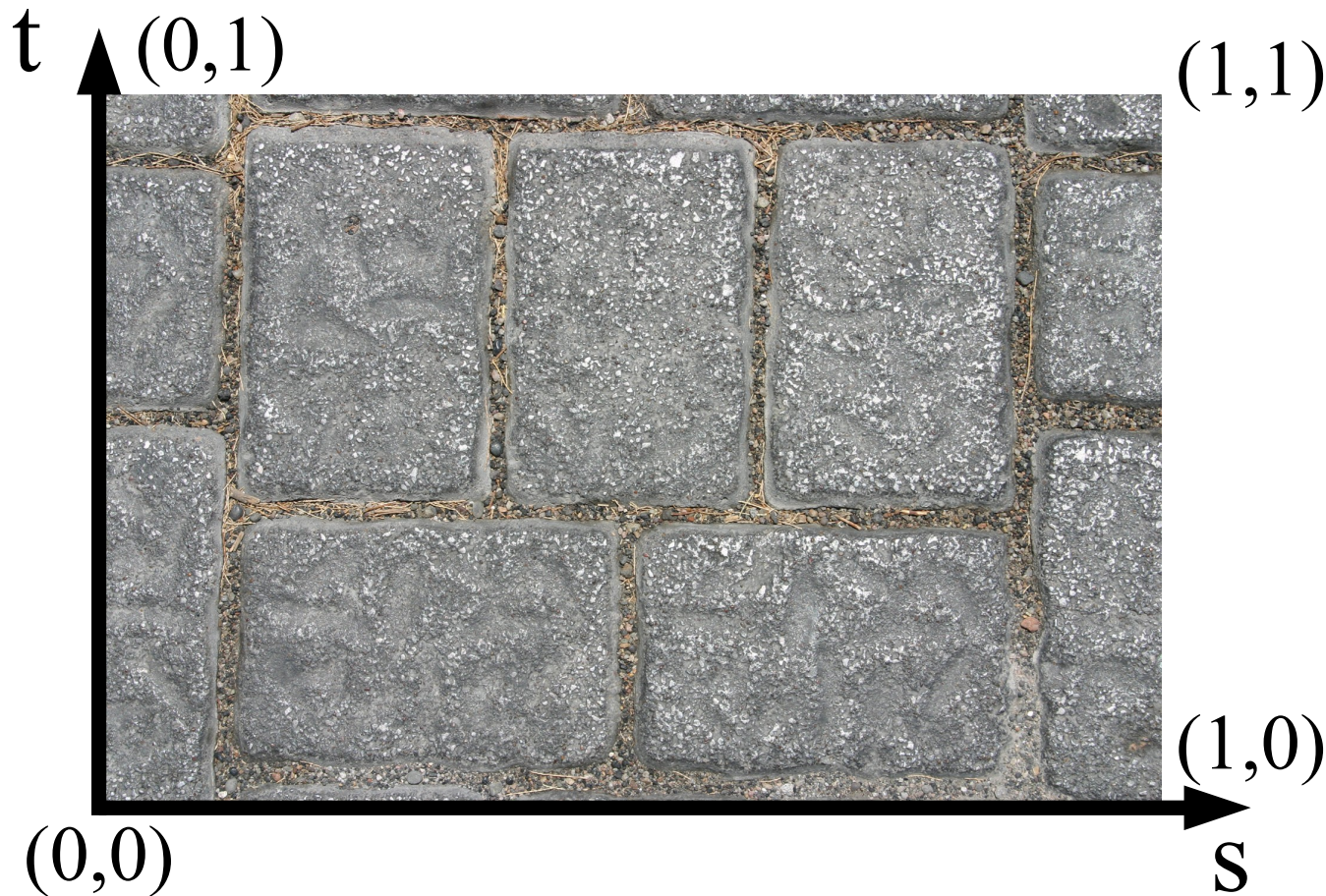- PCX
- TIFF
- PSD (Adobe Photoshop)

vector

. scalable

. resolution independent

. no background

. cartoon-like

. inappropriate for photo-realistic images
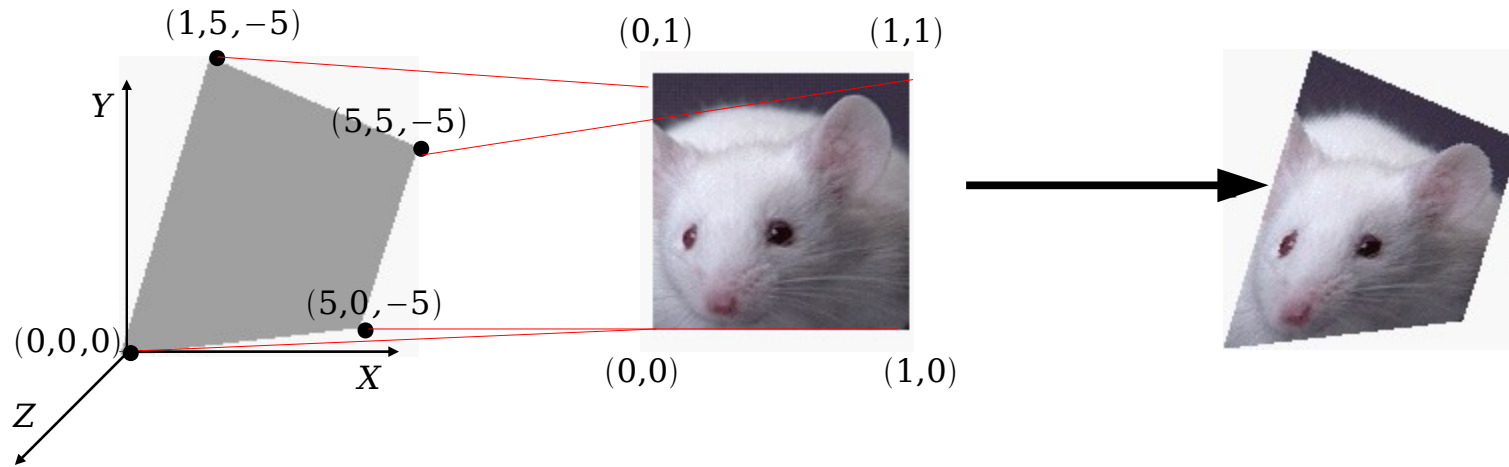
. metafiles contain both raster and vector data
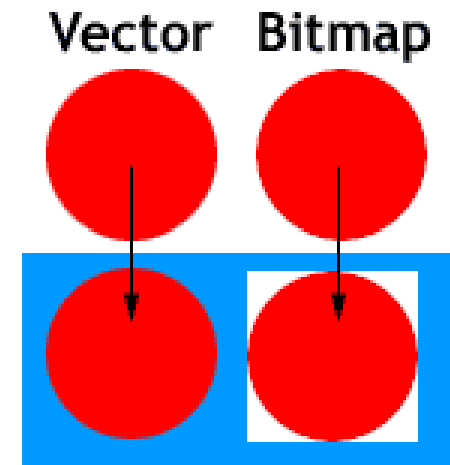
textures

how to map a texture to a surface?

t (0,1) (1,1)

(0,0) (1,0)

s

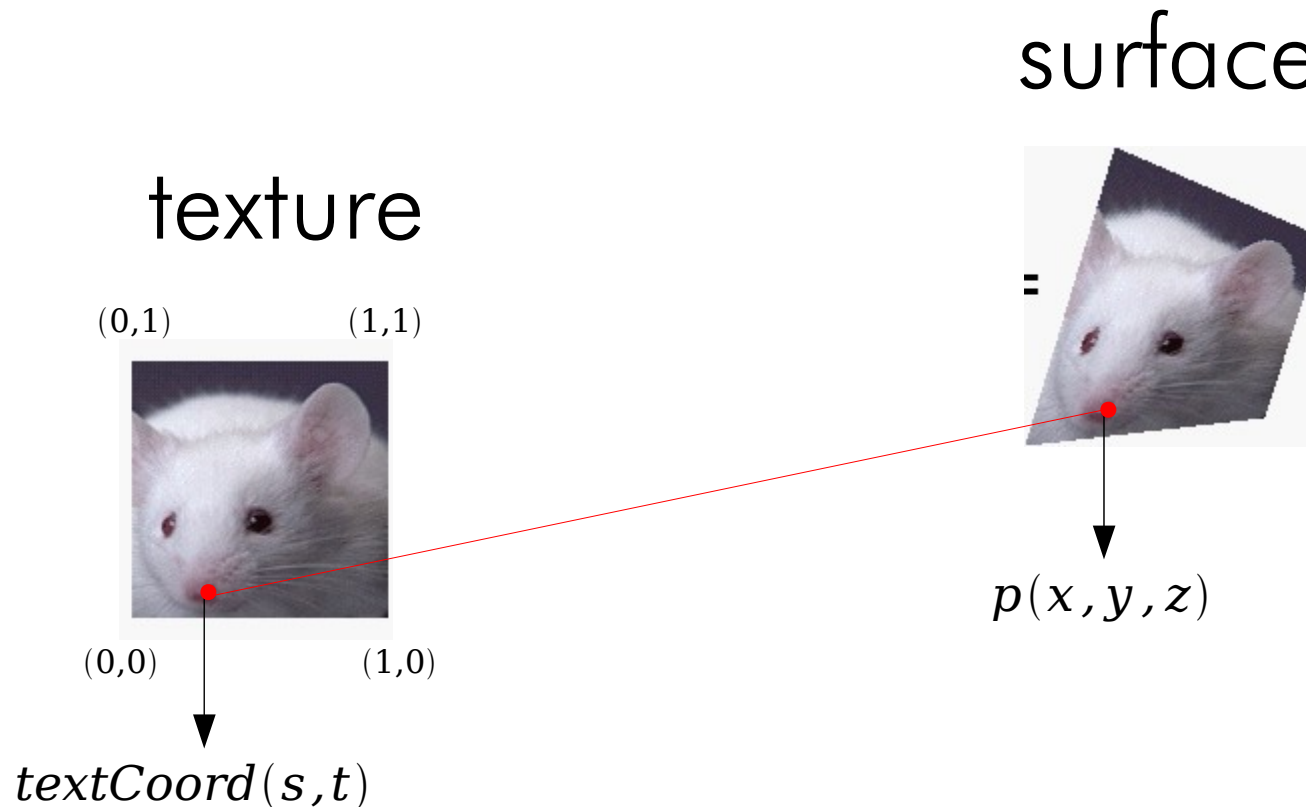apply once or repeat

textures

polygons can have:
. arbitrary size
. arbitrary shape

textures

texture value lookup:

there is a unique texture coordinate (s,t) that corresponds to a location on the surface

surface

texture

(0,1)   (1,1)

$p(x,y,z)$

(0,0)   (1,0)

$textCoord(s,t)$

textures
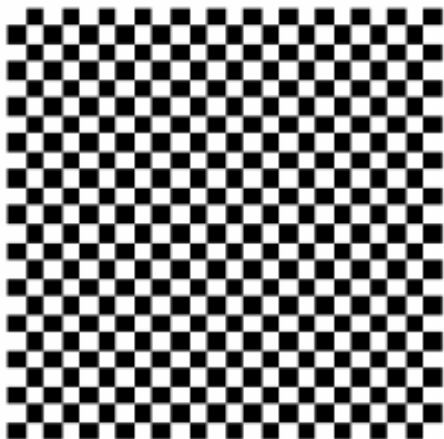
what happens if the texture
coordinates are outside [0,1]?

textures

what happens if the texture coordinates are outside [0,1]?
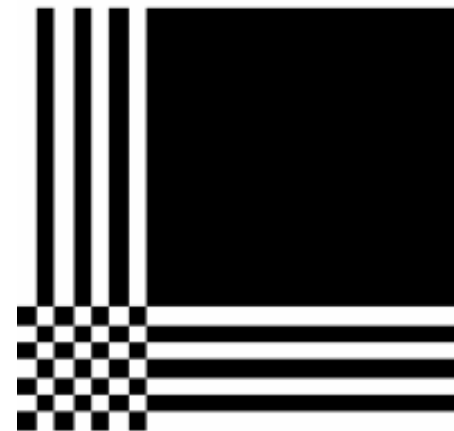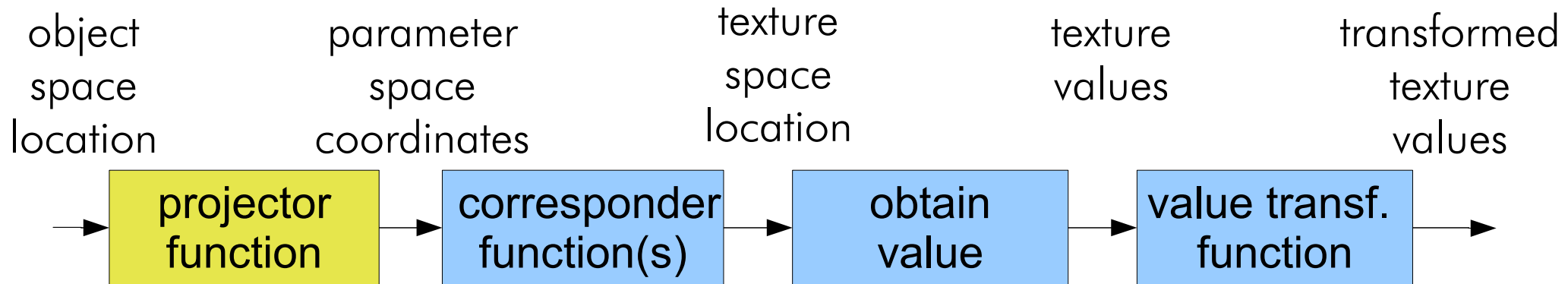
↓

repeat or clamp

textures

normalize [0,1]

it is possible to choose
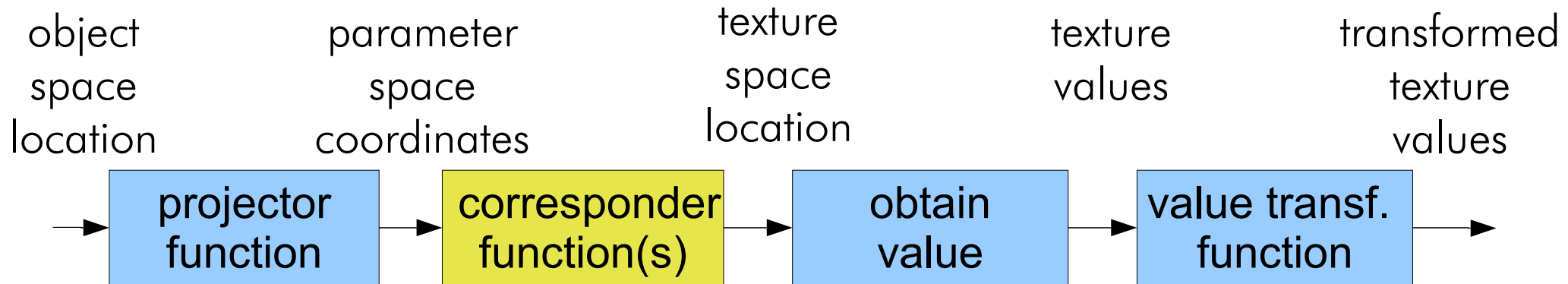part of the texture

texture pipeline

obtaining the surface's location and projecting into parameter space, usually 2D (u,v) space

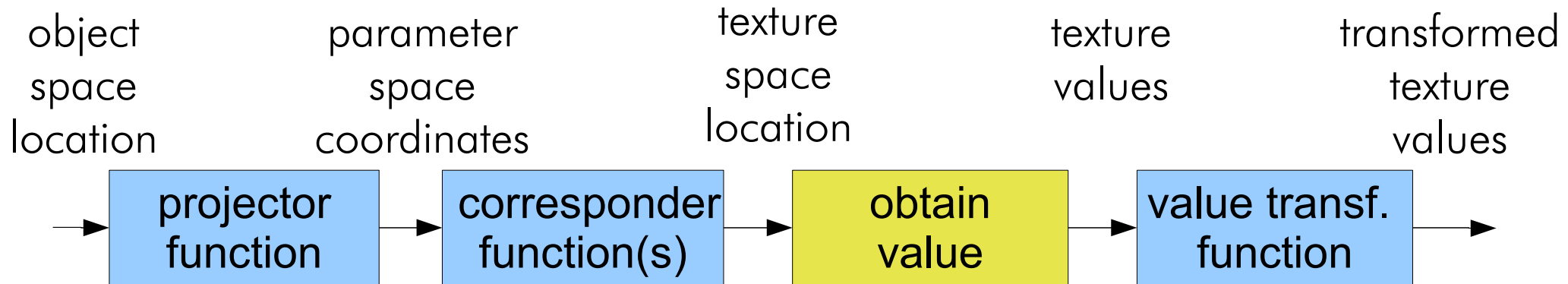projection: spherical, cylindrical, planar

texture pipeline

object space location → [projector function] → parameter space coordinates → [corresponder function(s)] → texture space location → [obtain value] → texture values → [value transf. function] → transformed texture values
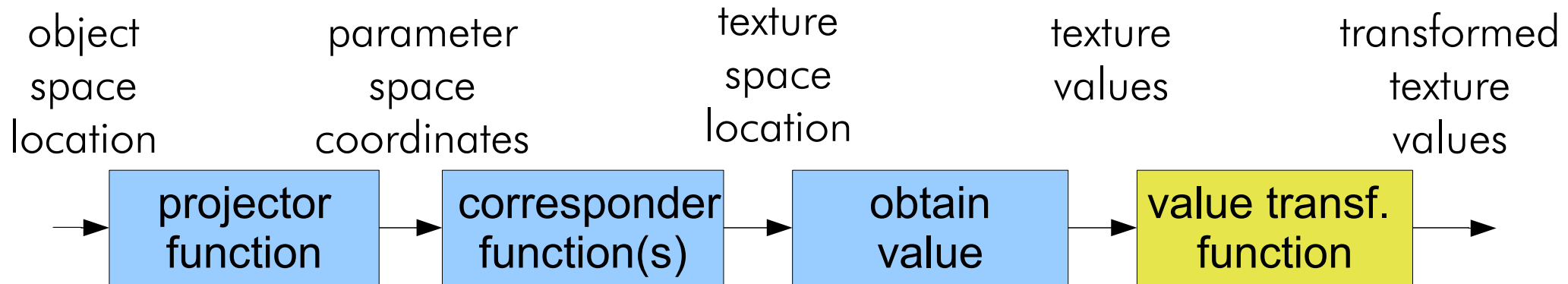
convert parameter-space coord to texture space coord.

. enables textures transofrmations: scale, rotate, translate, shear....
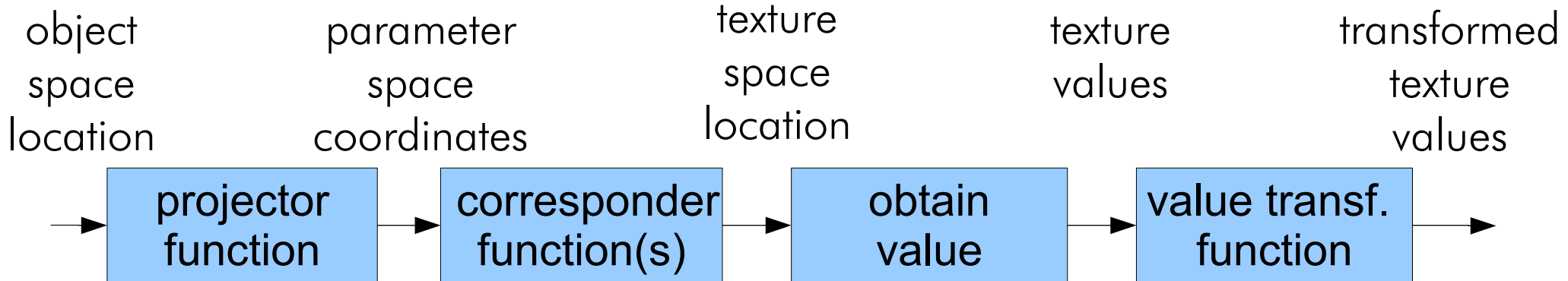. control the way an image is applied: repeat, mirror, clamp....

texture pipeline

object space location → **projector function** → parameter space coordinates → **corresponder function(s)** → texture space location → **obtain value** → texture values → **value transf. function** → transformed texture values
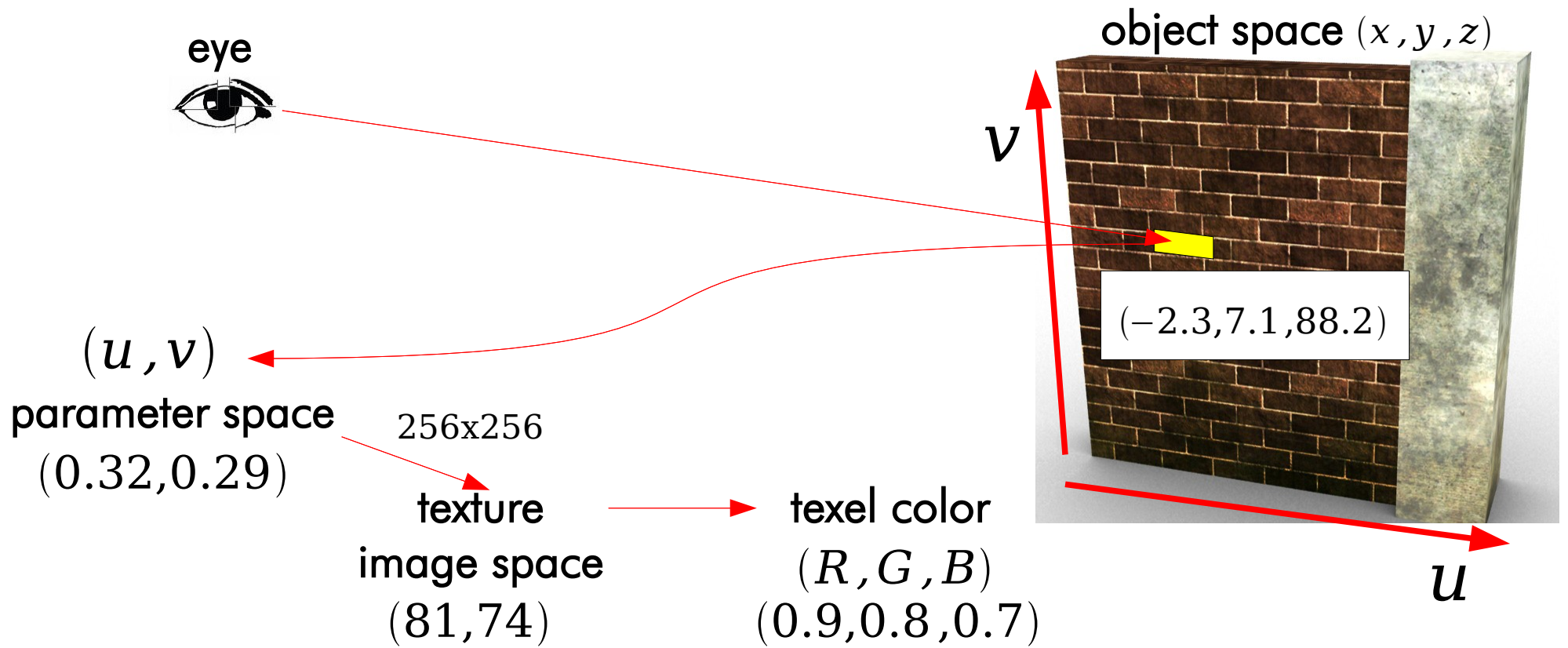
produce texture space coordinate to obtain texture values. We need to retrieve texel information from the image.

texture pipeline

object space location → **projector function** → parameter space coordinates → **corresponder function(s)** → texture space location → **obtain value** → texture values → **value transf. function** → transformed texture values

[optional] multiplies the texture values by the color if the image is too dark.

texture pipeline

eye

object space (x,y,z)

v

(u,v)
parameter space
$(0.32,0.29)$

$(-2.3,7.1,88.2)$

256x256

texture
image space
$(81,74)$

texel color
(R,G,B)
$(0.9,0.8,0.7)$

u

object space location	parameter space coordinates	texture space location	texture values	transformed texture values
→ projector function →	corresponder function(s) →	obtain value →	value transf. function →

filtering

what happens if a texture coordinate
doesn't fit directly a pixel in the texture but
in between?
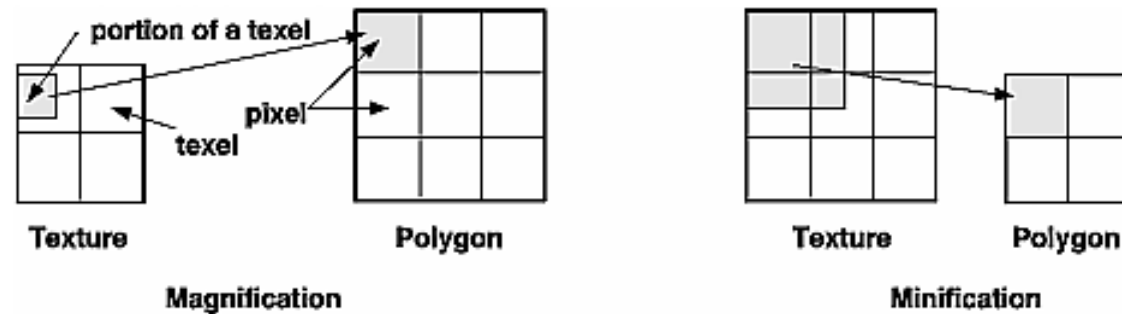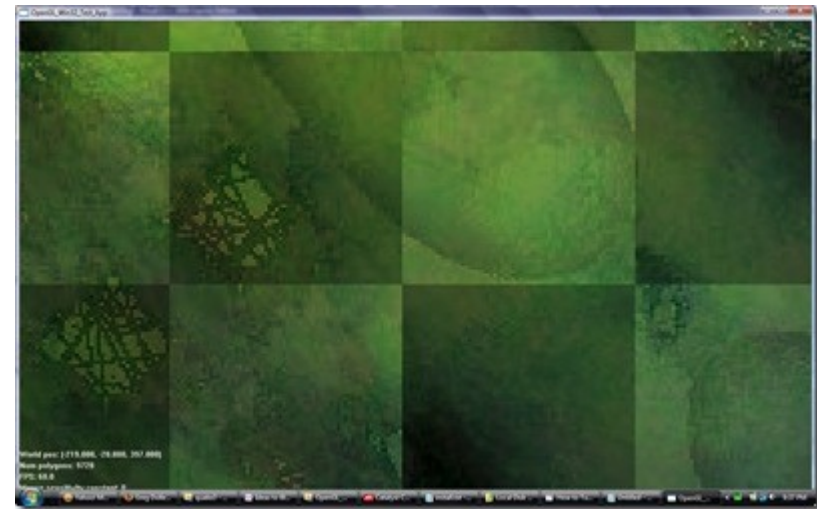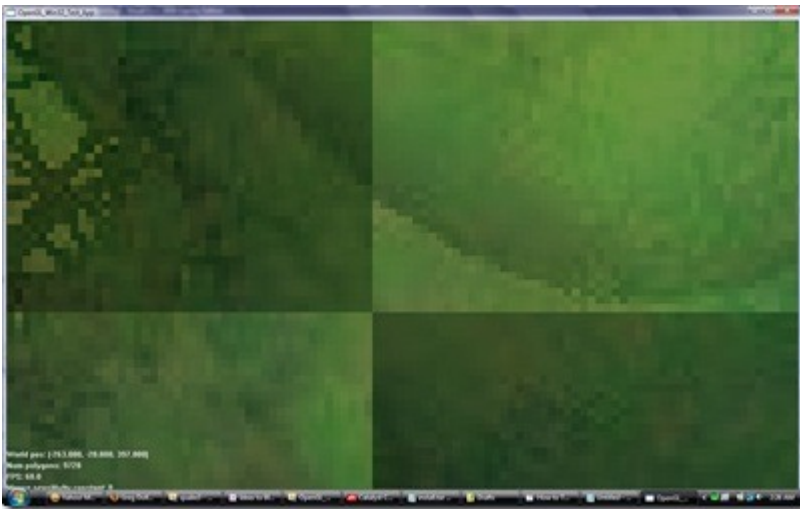

Some methods:
- . nearest-neighbor interpolation
- . nearest neighbor with mipmapping
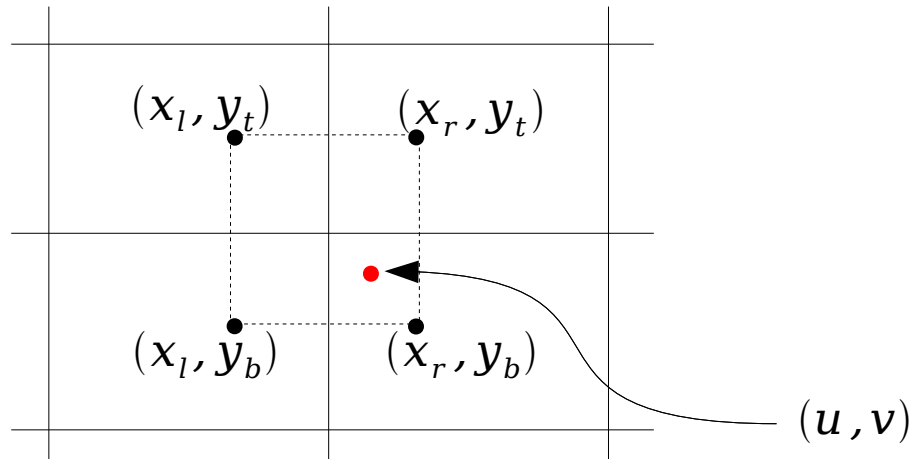- . bilinear filtering
- . trilinear filtering

filtering

texture magnification

bilinear filtering using weighted average
of 4 nearest texels

filtering

texture magnification: bilinear interpolation

$$b(p_u,p_v)=(1-u')(1-v')t(x_l,y_b)+u'(1-v')t(x_r,y_b)$$

$$+(1-u')v't(x_l,y_t)+u'v't(x_r,y_t)$$

images from http://gregs-blog.com