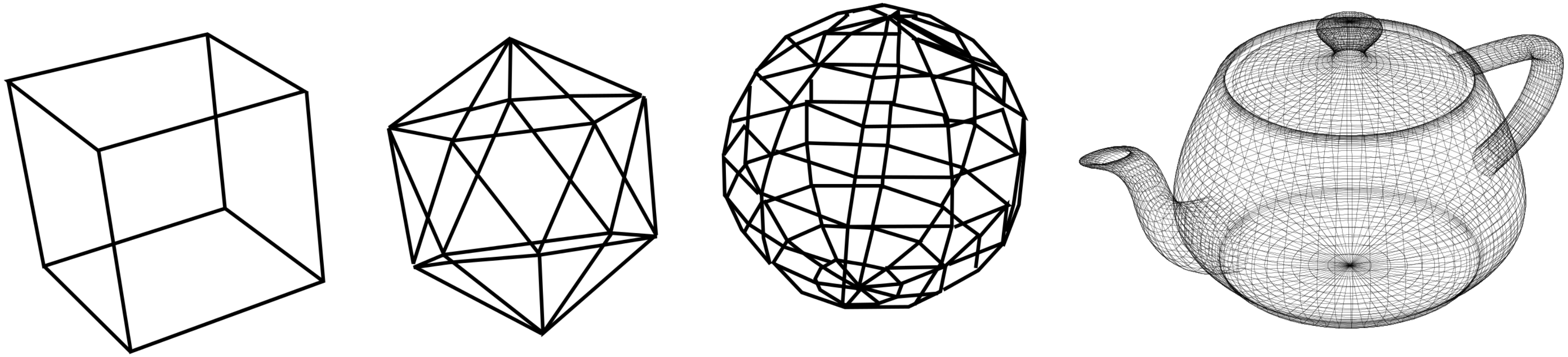
VIETNAM ACADEMY OF SCIENCE AND TECHNOLOGY
UNIVERSITY OF SCIENCE AND TECHNOLOGY OF HANOI

COMPUTER GRAPHICS

Lecture 6: Texturing

Lecturer: Dr. NGUYEN Hoang Ha



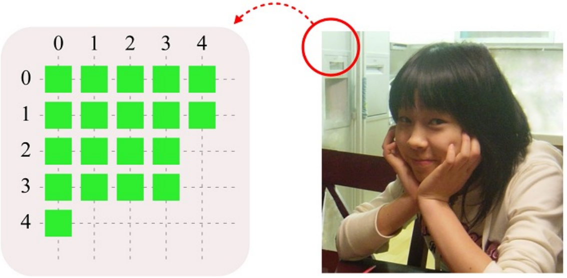
Reference: JungHyun Han. 2011. 3D Graphics for Game Programming (1st ed.), chapter 8

A□□□□□□□□□

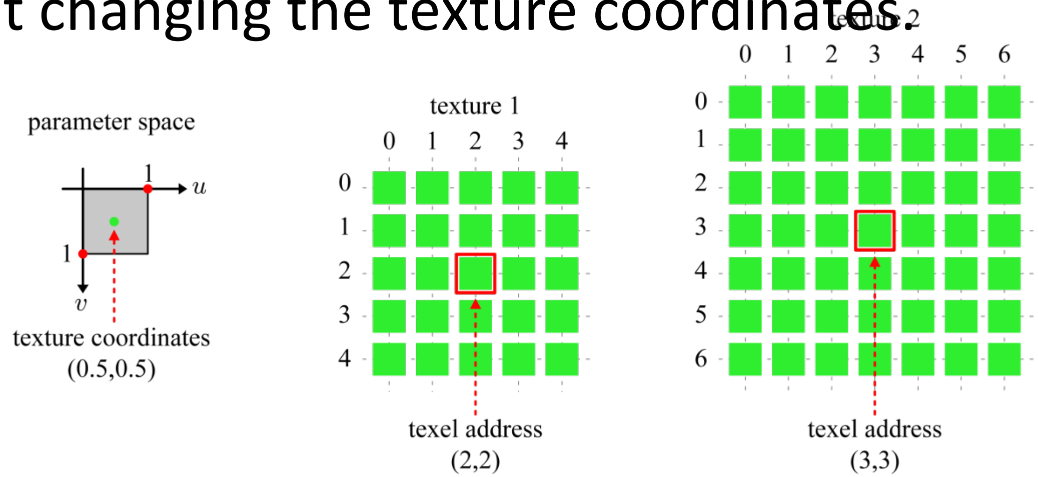
- Revisiting texture concept
- Texture Addressing Mode
- Texture Filtering
- Mipmapping
- Anisotropic Filtering

[illegible]C[illegible]

- An image texture is a 2D array of *texels* (texture elements). Each texel has a unique address, i.e., 2D array index.

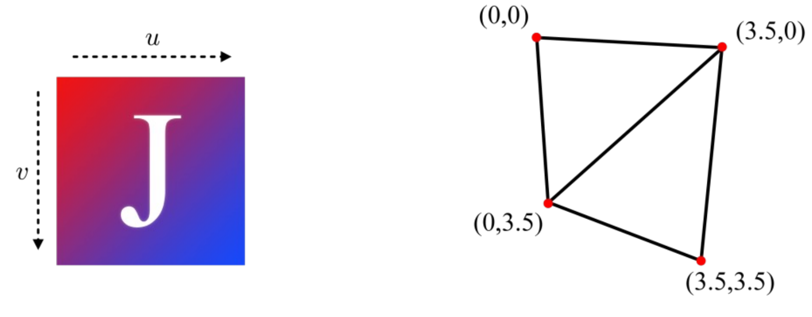


- Use normalized *texture coordinates* (*u*,*v*) in [0,1] → multiple images of different resolutions can be glued to a surface without changing the texture coordinates.



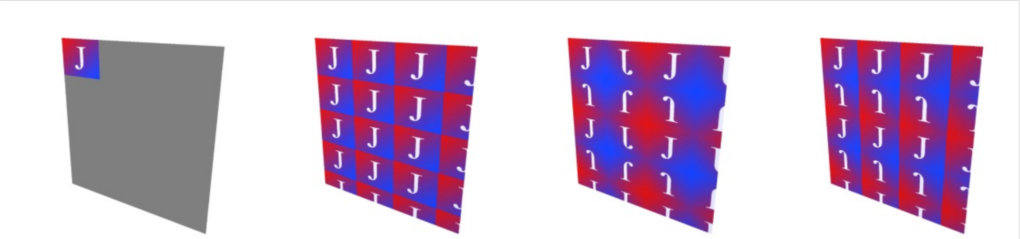
[illegible]A[illegible]

- The texture coordinates (u,v) are not necessarily in the range of [0,1]. The *texture addressing mode* handles (u,v)s outside the range.
  - *B*[illegible]
  - [illegible]
  - [illegible]



(a)

(b)



(c)

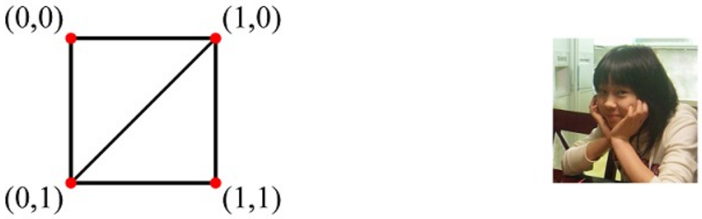
(d)

(e)

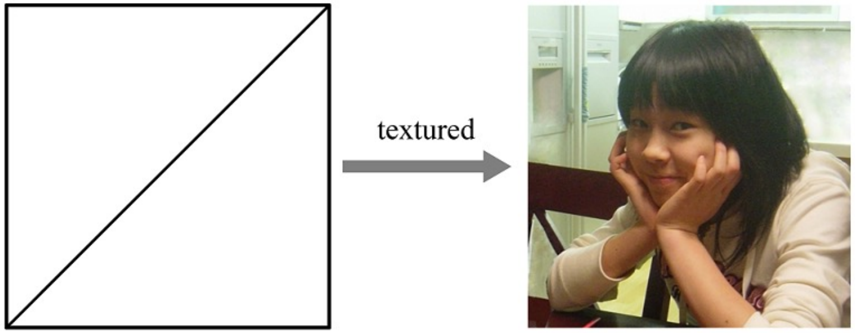
(f)

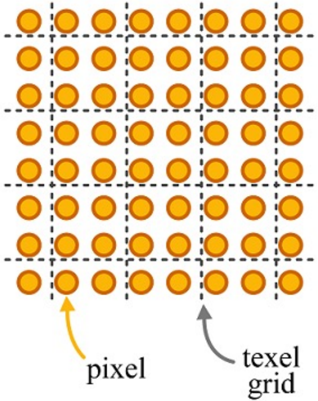
[illegible] F[illegible] & [illegible]

- Consider a quad. For each fragment located at (*x*,*y*) in the screen, its texture coordinates (*u*,*v*) are mapped to the texel address (t_x,t_y). We say that the fragment at (*x*,*y*) is *projected* onto (t_x,t_y).



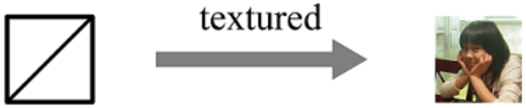
- (t_x,t_y) are floating-point values → texels around (t_x,t_y) are sampled. This sampling process is called *texture filtering*.
- The screen-space quad may appear larger than the image texture, and therefore the texture is *magnified* so as to fit to the quad. There are more pixels than texels.

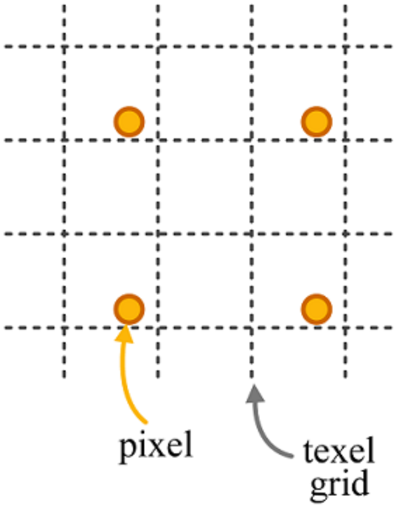




[illegible] F [illegible]
[illegible] & [illegible] ([illegible])

- In contrast, the screen-space quad may appear smaller than the image texture, and the texture is minified. The pixels are *sparsely projected* onto the texture space.

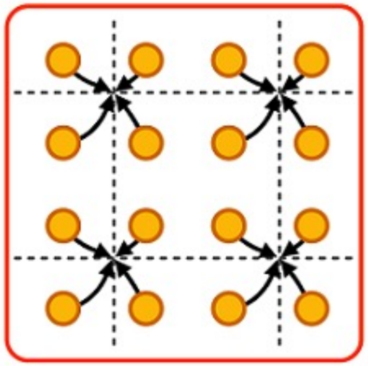




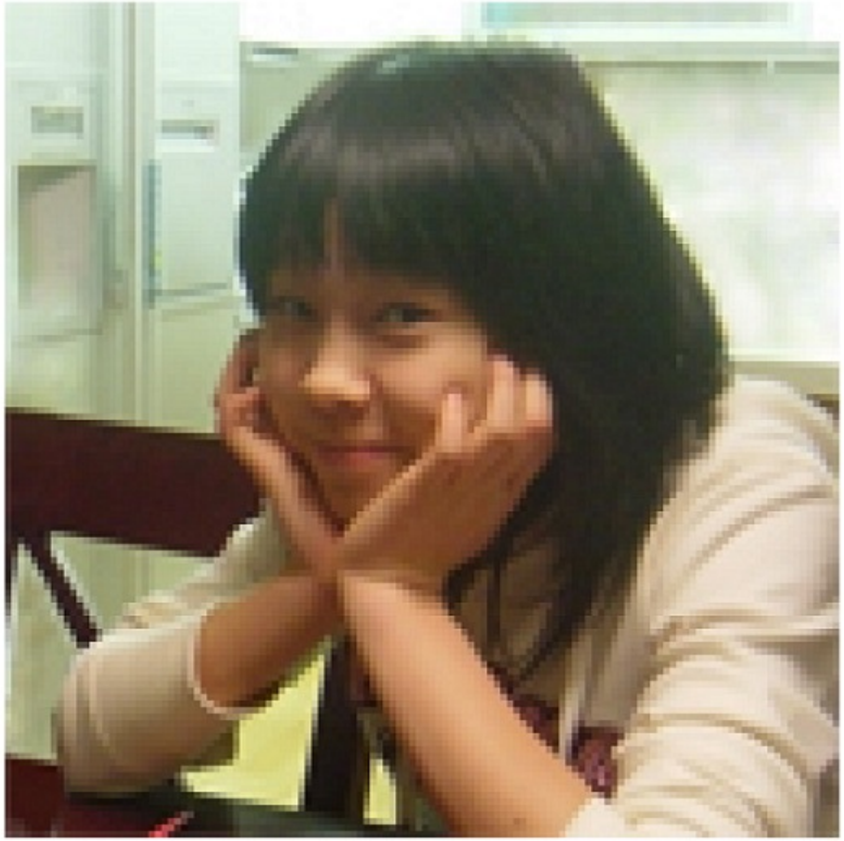
- Summary
  - [illegible]: [illegible]
  - [illegible]: [illegible]

F [illegible]

- Option 1: Nearest point sampling
  - A [illegible]
  - C [illegible], [illegible], [illegible].

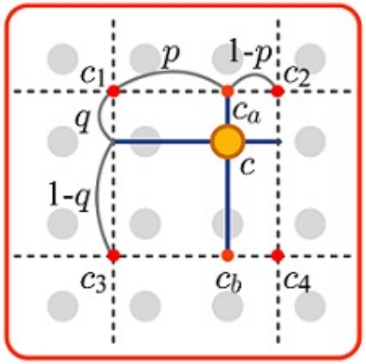


$(\lfloor t_x+0.5 \rfloor, \lfloor t_y+0.5 \rfloor)$



F[illegible] ([illegible])

- Option 2: Bilinear interpolation
  - [illegible]

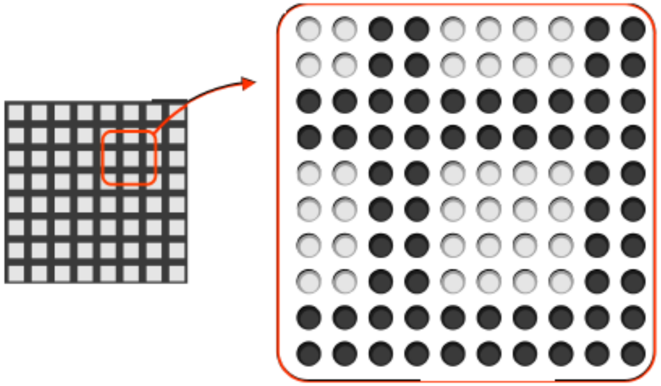


$$p = (t_x - \lfloor t_x \rfloor)$$
$$q = (t_y - \lfloor t_y \rfloor)$$

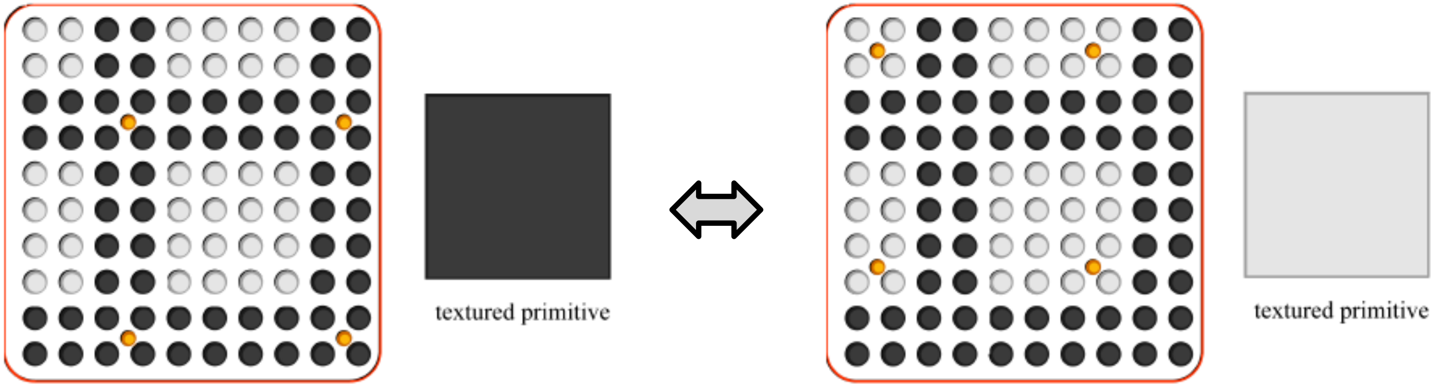
$$c_a = (1-p)c_1 + pc_2$$
$$c_b = (1-p)c_3 + pc_4$$
$$c = (1-q)c_a + qc_b$$

F

- Consider the following checker-board image texture.

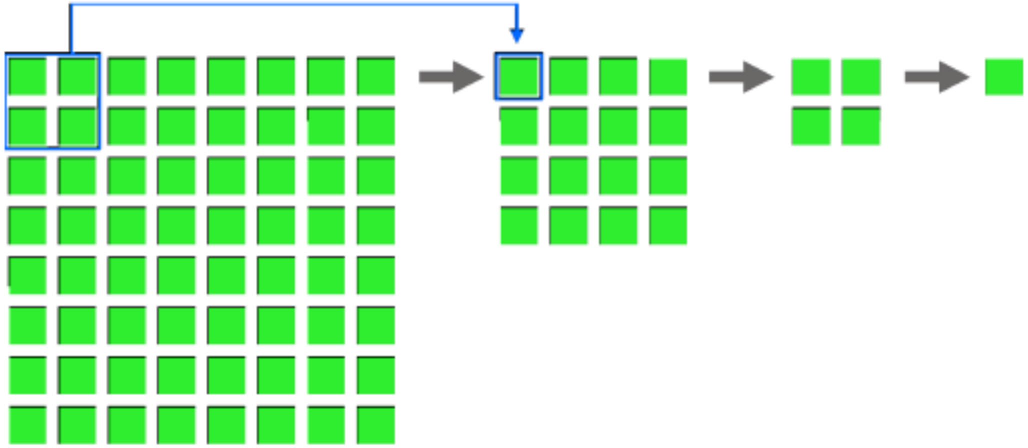


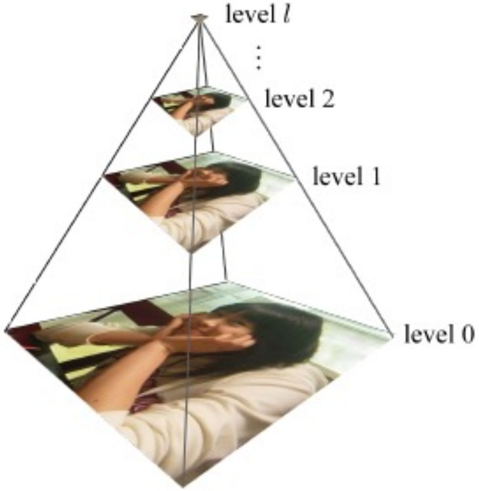
- If all pixels are surrounded by dark-gray texels, the textured primitive appears dark gray. If all pixels are surrounded by light-gray texels, the textured primitive appears light gray.



□□□□□

- Reason of minification problem: the texture is larger than the screen-space primitive.
- A solution - Decrease the texture size such that there is roughly one-to-one correspondence between pixels and texels.
- For decreasing the texture size, *down-sampling* is adopted.

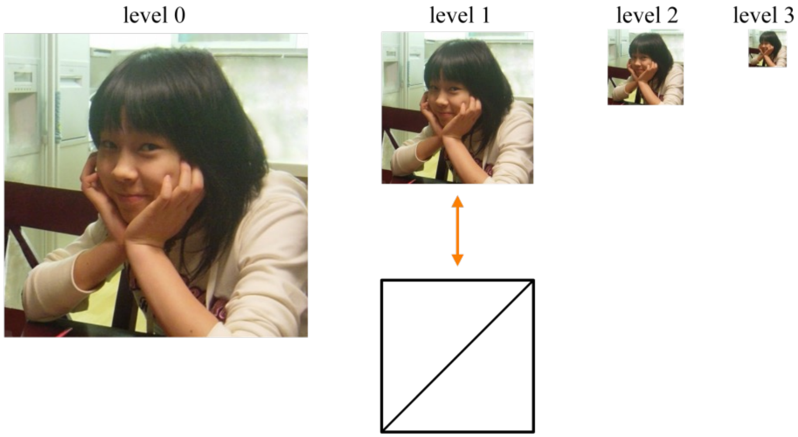




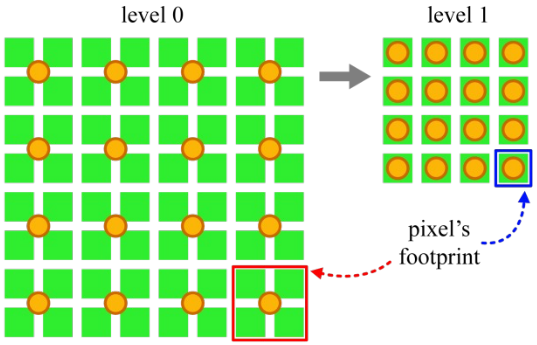
- Given an original texture of 2^lx2^l resolution, a texture pyramid of (l + 1) levels is constructed, where the original texture is located at level 0.
- The pyramid is called a mipmap.
- The level to filter is named *level of detail*, and is denoted by λ.

[illegible] ([illegible])

- In the example, the quad and level-1 texture have the same size. A pixel's *footprint* covers 2x2 texels in level 0, but covers a single texel in level 1.
- When a pixel footprint covers *mxm* texels of level-0 texture, λ is set to $\log_2 m$.



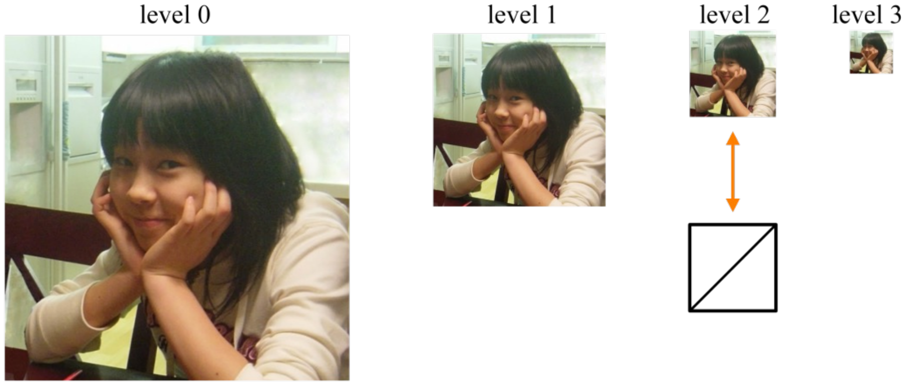
(a)



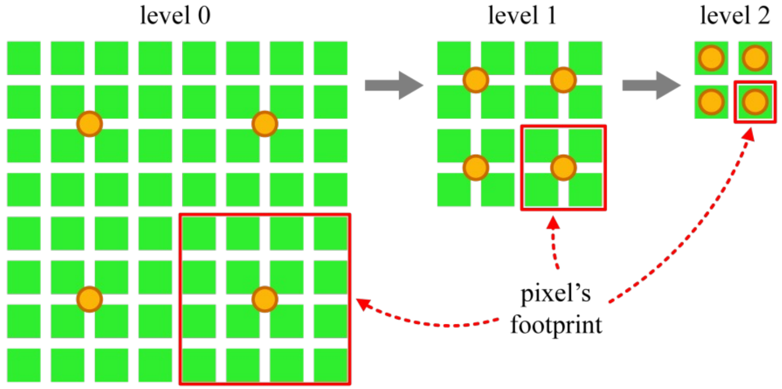
(b)

[illegible]

- In the example, the screen-space quad and level-2 texture have the same size. A pixel's footprint covers exactly a single texel in level-2 texture, which is then filtered.



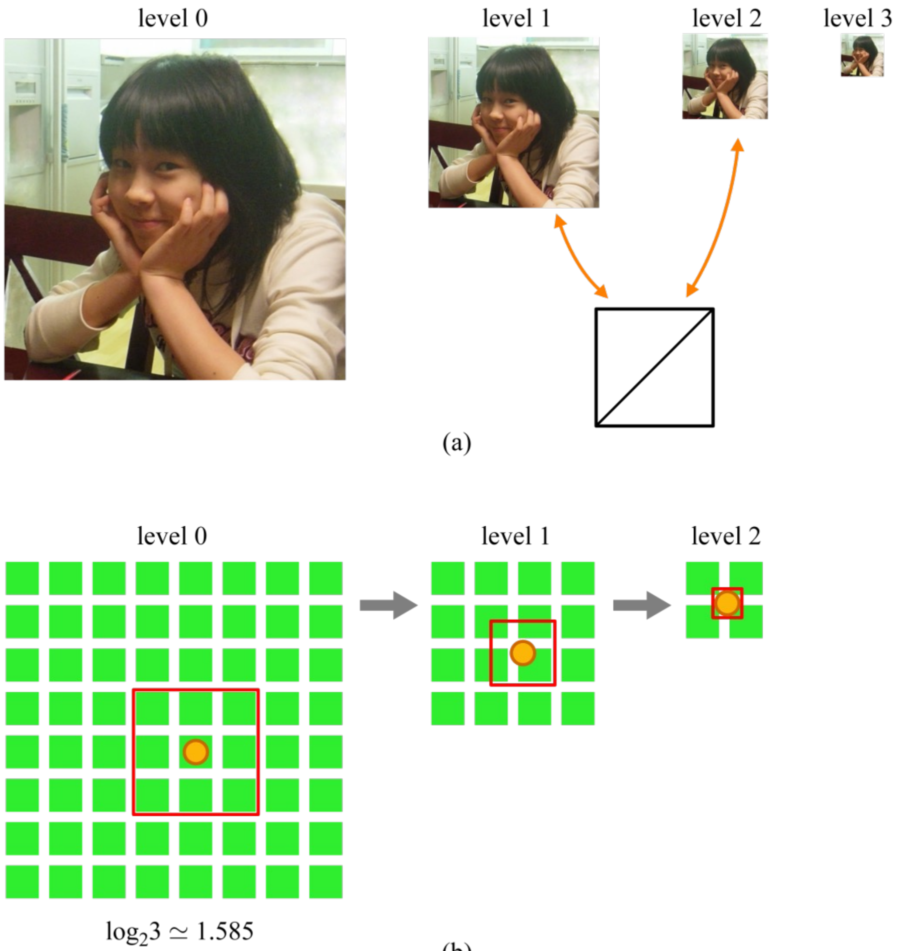
(a)



(b)

[illegible]([illegible])[illegible]

- E.g: $\lambda = \log_2 3 = 1.585$.., and so we see levels 1 and 2, more formally $\lfloor\lambda\rfloor$ and $\lceil\lambda\rceil$.



(a)

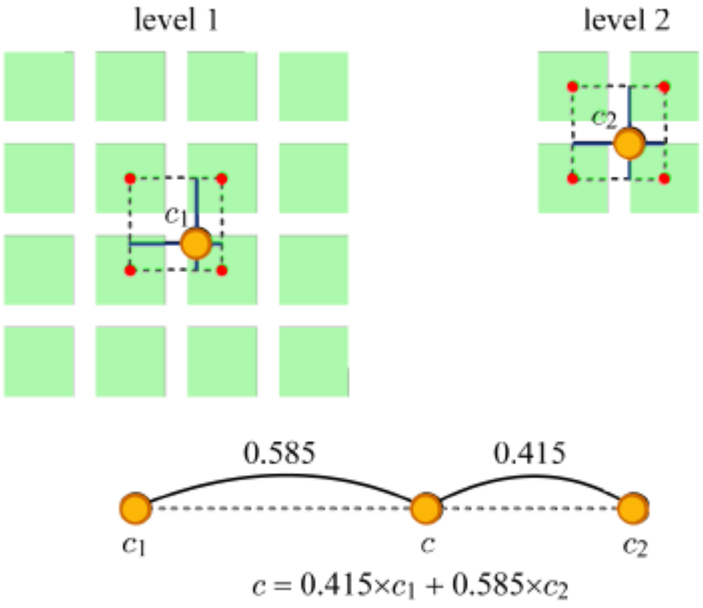
(b)

[illegible] ([illegible])[illegible]

- Between ⌊λ⌋ and ⌈λ⌉, which one to choose?
  - [illegible]

    $$i_\lambda = \lfloor \lambda + 0.5 \rfloor$$

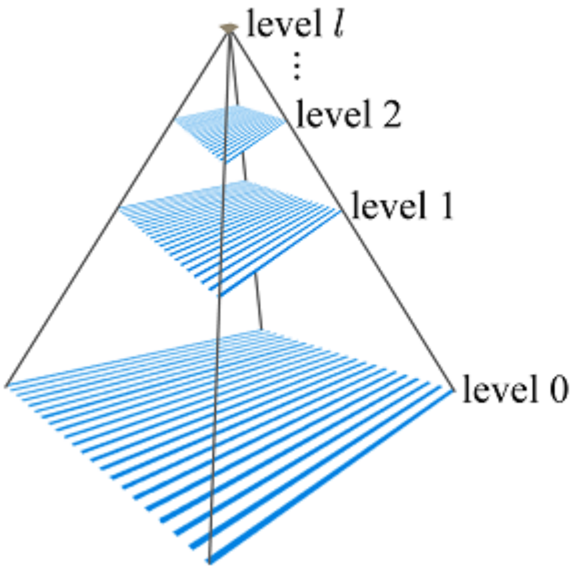
  - [illegible]

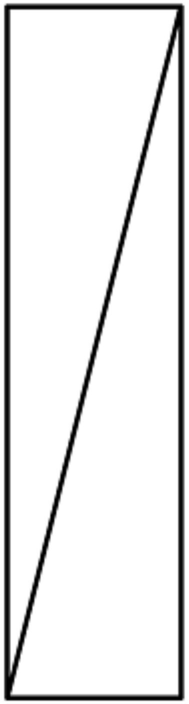


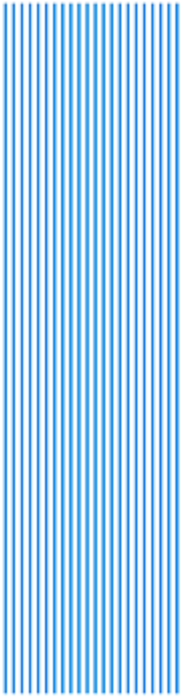
(c)

□□□□□□□□□□□□□□- E□□□□□□□□

- Consider a mipmap of stripe texture, and a long thin quad.
- When the quad is orthogonal to the view direction, the texturing result will be as follows.

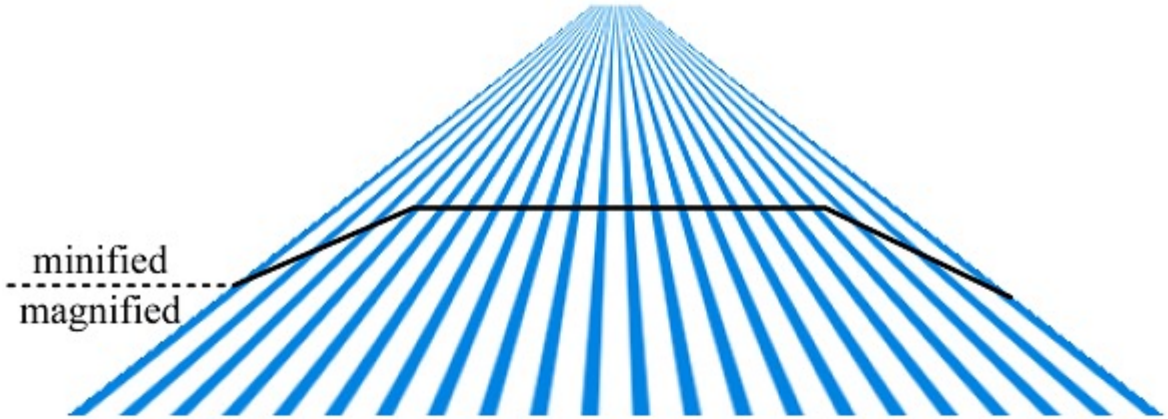






[illegible]

- What if the quad is angled obliquely away from the viewer?
- In general, for each pixel of the screen-space primitive, it is determined whether the texture is magnified or minified.
- The black lines shown below partition the quad into the magnified and minified parts.
  - [illegible]
  - [illegible]

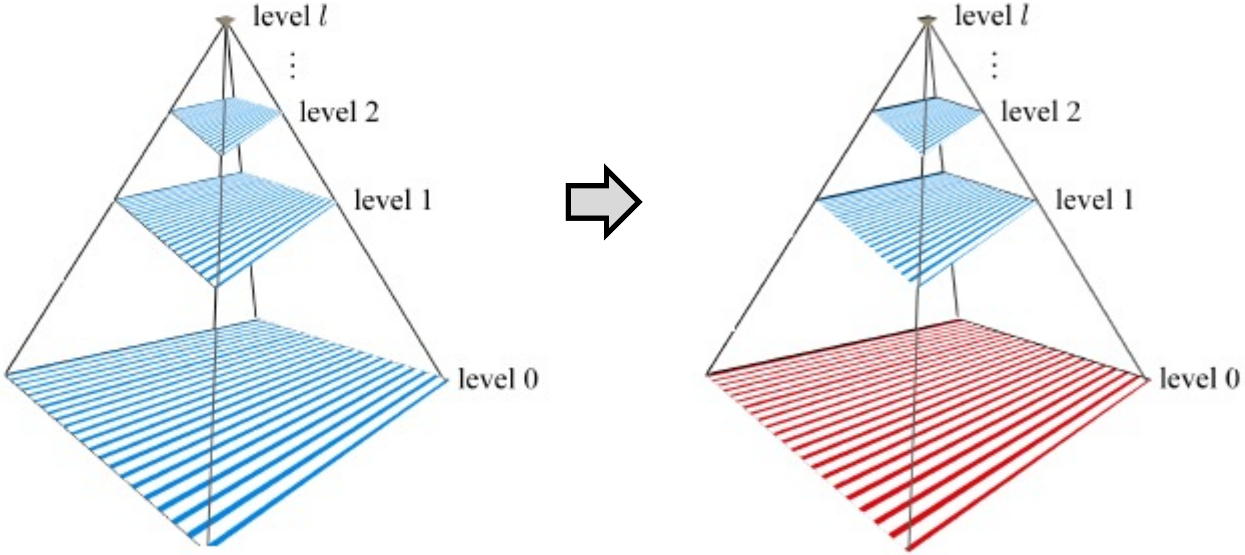


□□□□□□□□□□□□□E□□□□□□□□(□□□□□□)□

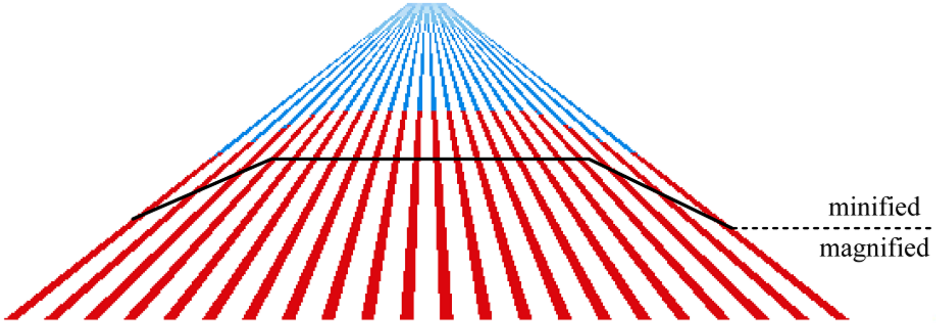
- Mipmapping is implemented in hardware. However, the graphics APIs allow us to control the mipmap filtering mechanism.
- The major control is over magnification filter (MAGFILTER), minification filter (MINFILTER), and mipmap level selection (LEVEL).
  - □AGFI□□E□
    - □□□□□□□□□□□□□□□□□□□□□□(□EA□E□□)
    - B□□□□□□□□□□□□□□□□□□□□□(BI□I□EA□)
  - □I□FI□□E□
    - □□□□□□□□□□□□□□□□□□□□□□(□EA□E□□)
    - B□□□□□□□□□□□□□□□□□□□□□(BI□I□EA□)
  - □E□E□
    - □□□□□□□□□□□□□(□EA□E□□)
    - □□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□(□I□EA□)

[illegible] E [illegible] ([illegible])

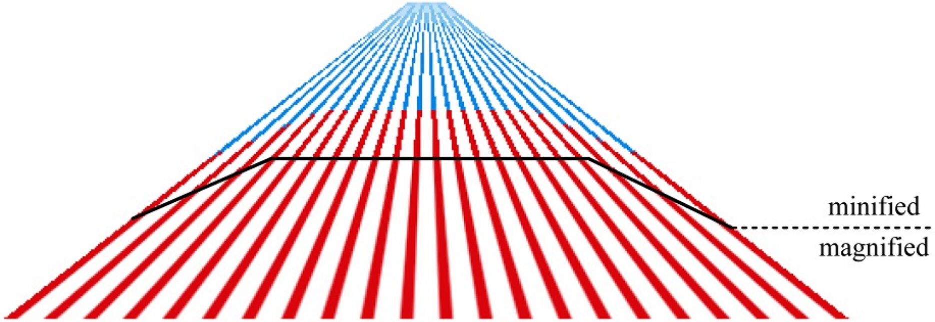
- Let us replace level-0 texture for visualization purpose solely



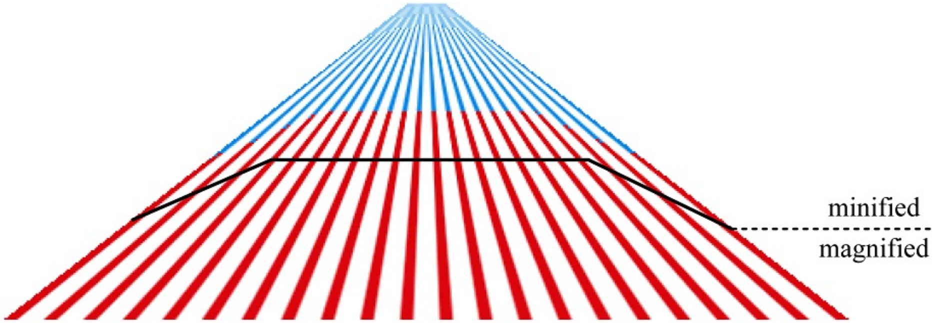
[illegible] E [illegible] ([illegible])



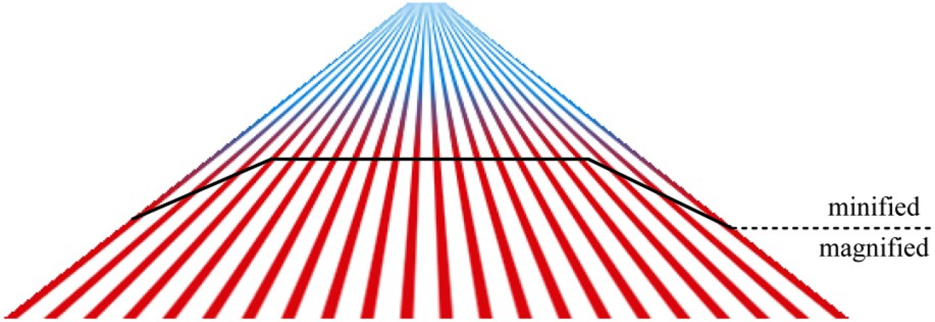
(MAGFILTER, MINFILTER, LEVEL) = (NEAREST, NEAREST, NEAREST)



(MAGFILTER, MINFILTER, LEVEL) = (BILINEAR, NEAREST, NEAREST)



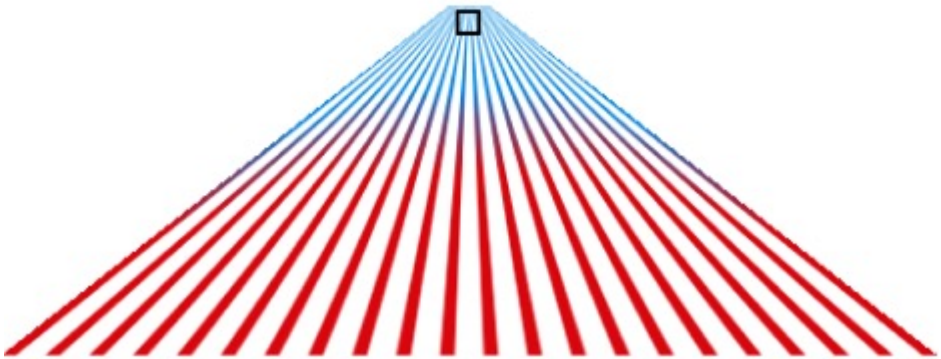
(MAGFILTER, MINFILTER, LEVEL) = (BILINEAR, BILINEAR, NEAREST)

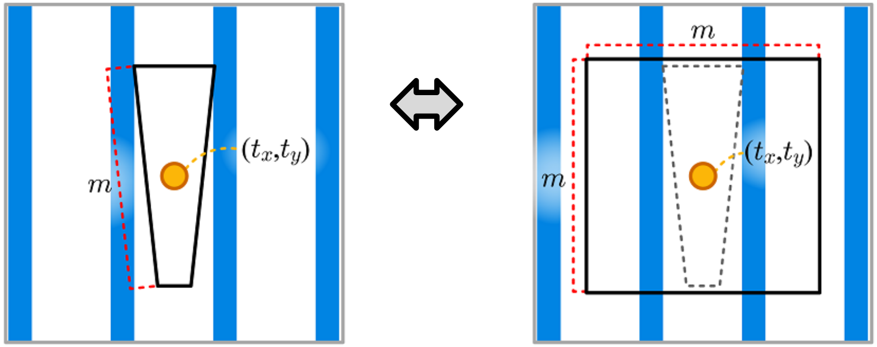


(MAGFILTER, MINFILTER, LEVEL) = (BILINEAR, BILINEAR, LINEAR)

A[illegible] F[illegible]

- Consider a square pixel and its *footprint* in the texture space.
  - A [illegible]
  - [illegible] λ, [illegible] λ [illegible] 2 [illegible] F[illegible] λ [illegible] x [illegible]
  - [illegible] x [illegible] C[illegible]
  - [illegible]

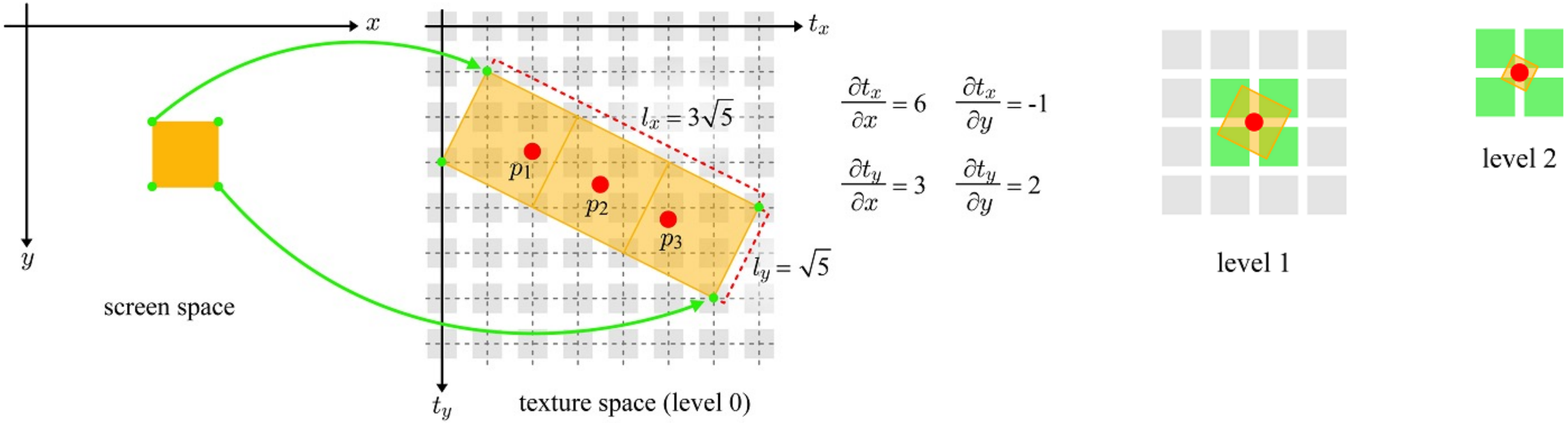




- What is desired is *anisotropic filtering,* which filters only the area covered by the anisotropic footprint.

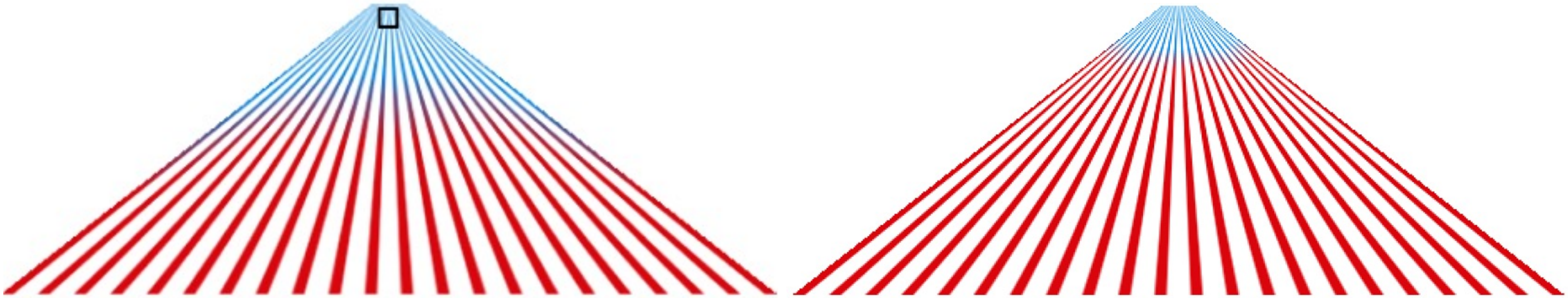
A[illegible] F[illegible] ([illegible])

- Anisotropic filtering algorithm – an example with a parallelogram footprint
  - [illegible]-[illegible], [illegible].
  - E[illegible], [illegible].
  - [illegible] λ.
  - F[illegible] ([illegible]), [illegible].
  - A[illegible].

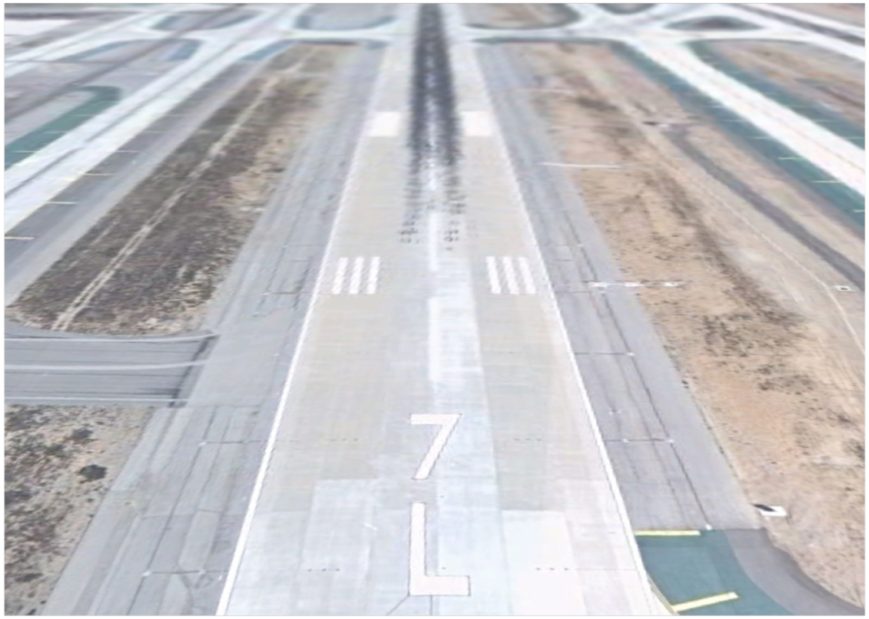

x
y
screen space
t_x
t_y
$l_x = 3\sqrt{5}$
p_1
p_2
p_3
$l_y = \sqrt{5}$
texture space (level 0)
$\frac{\partial t_x}{\partial x} = 6$
$\frac{\partial t_x}{\partial y} = -1$
$\frac{\partial t_y}{\partial x} = 3$
$\frac{\partial t_y}{\partial y} = 2$
level 1
level 2

A [illegible] F [illegible] ([illegible])

- Compare the results.
  - [illegible]
  - [illegible]



[illegible].A [illegible] F[illegible]



(a)

(b)

□□□□□□□□□□□□□□□G□

```
data = new unsigned char [imageSize]; // Create a buffer
// then Read the actual data fread(data,1,imageSize,file);
GLuint textureID; // Create one OpenGL texture
glGenTextures(1, &textureID);
// "Bind" the newly created texture : all future texture functions will modify this texture
glBindTexture(GL_TEXTURE_2D, textureID);

// Give the image to OpenGL
glTexImage2D(GL_TEXTURE_2D, 0,GL_RGB, width, height, 0, GL_BGR, GL_UNSIGNED_BYTE, data);

// ... nice trilinear filtering ...
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_REPEAT);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_REPEAT);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR_MIPMAP_LINEAR);
// ... which requires mipmaps. Generate them automatically.
glGenerateMipmap(GL_TEXTURE_2D);
……………………
// Bind our texture in Texture Unit 0
glBindTexture(GL_TEXTURE_2D, Texture);
```