Transforms and Texture Mapping

Computer Graphics and Imaging
UC Berkeley CS184
Summer 2020

Announcements

- Lecture Guides for this week are up
  - Gradescope checkpoints later today
- Project 1 - Released!
  - Please try to accept the assignment, build ASAP — very short timeline
  - Due Friday at 11:59 PM
  - Max 2 slip days
- Project 2 will be released on Friday
  - Keeping this pace for the rest of the class!

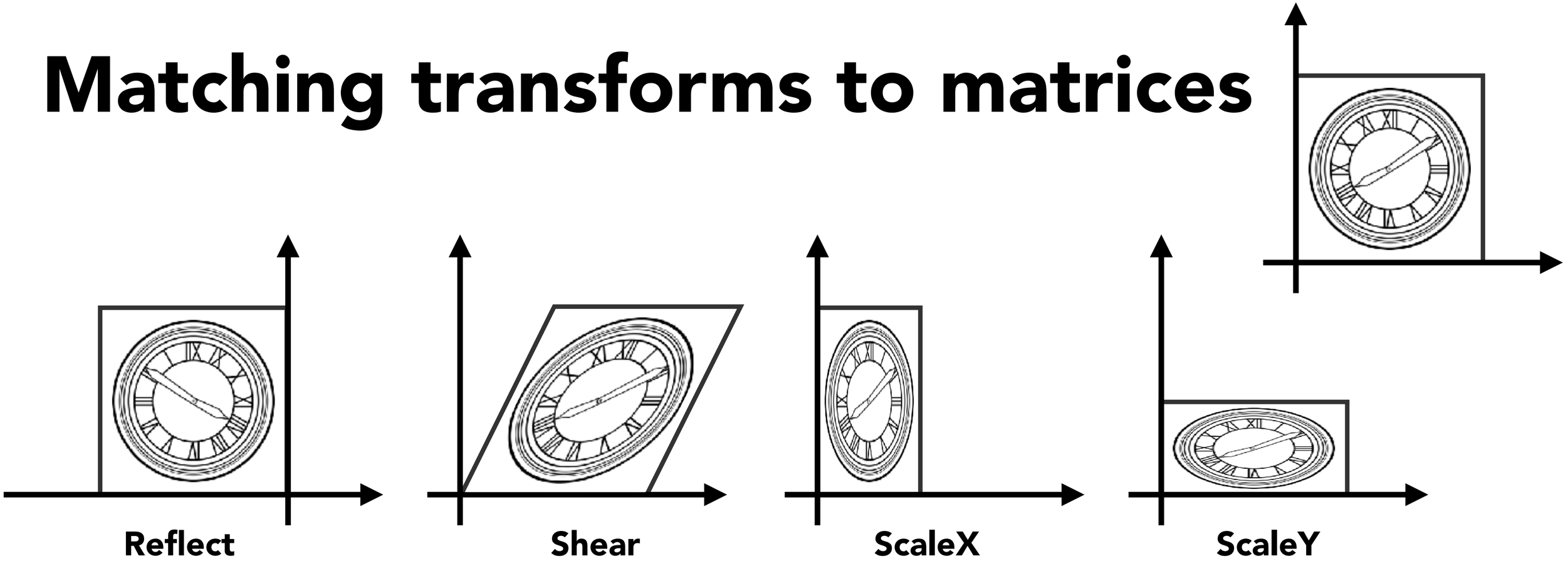
Transforms

(Today, 2D — we'll return to 3D transforms later!)

Core Ideas

- Transforms = matrices
  - Not commutative — order matters!
    - Matrices multiplied right to left
  - Compose basic transforms into more complex ones
- Homogenous coordinates — augment coordinates so we can represent affine transforms (linear map + translation) as a single matrix multiple
- Can think of transforming objects as transforming coordinate frames

Matching transforms to matrices



A

$$\begin{bmatrix} .5 & 0 \\ 0 & 1 \end{bmatrix}$$

B

$$\begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$$

C

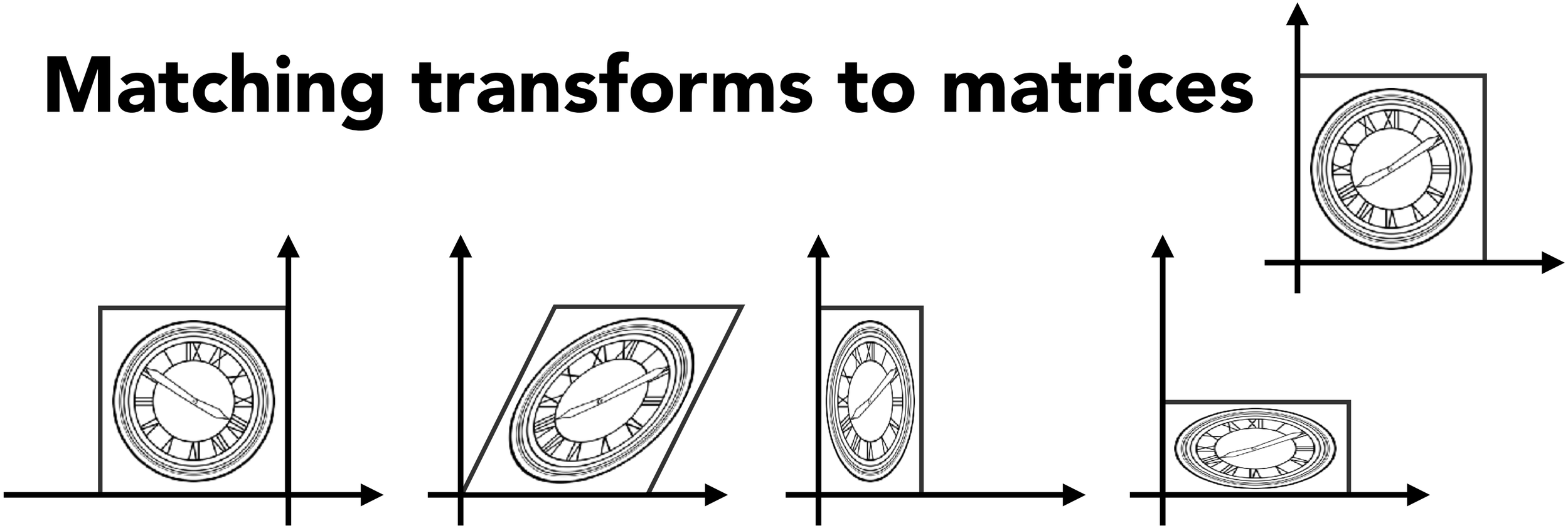
$$\begin{bmatrix} 1 & .5 \\ 0 & 1 \end{bmatrix}$$

D

$$\begin{bmatrix} 1 & 0 \\ 0 & .5 \end{bmatrix}$$

These are all examples of _____ transforms.

Matching transforms to matrices



A

$$\begin{bmatrix} .5 & 0 \\ 0 & 1 \end{bmatrix}$$

B

$$\begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$$

C

$$\begin{bmatrix} 1 & .5 \\ 0 & 1 \end{bmatrix}$$

D

$$\begin{bmatrix} 1 & 0 \\ 0 & .5 \end{bmatrix}$$

These are all examples of <u>linear</u> transforms.

Homogenous coordinates

What about translations?

Add a third coordinate (*w-coordinate*)

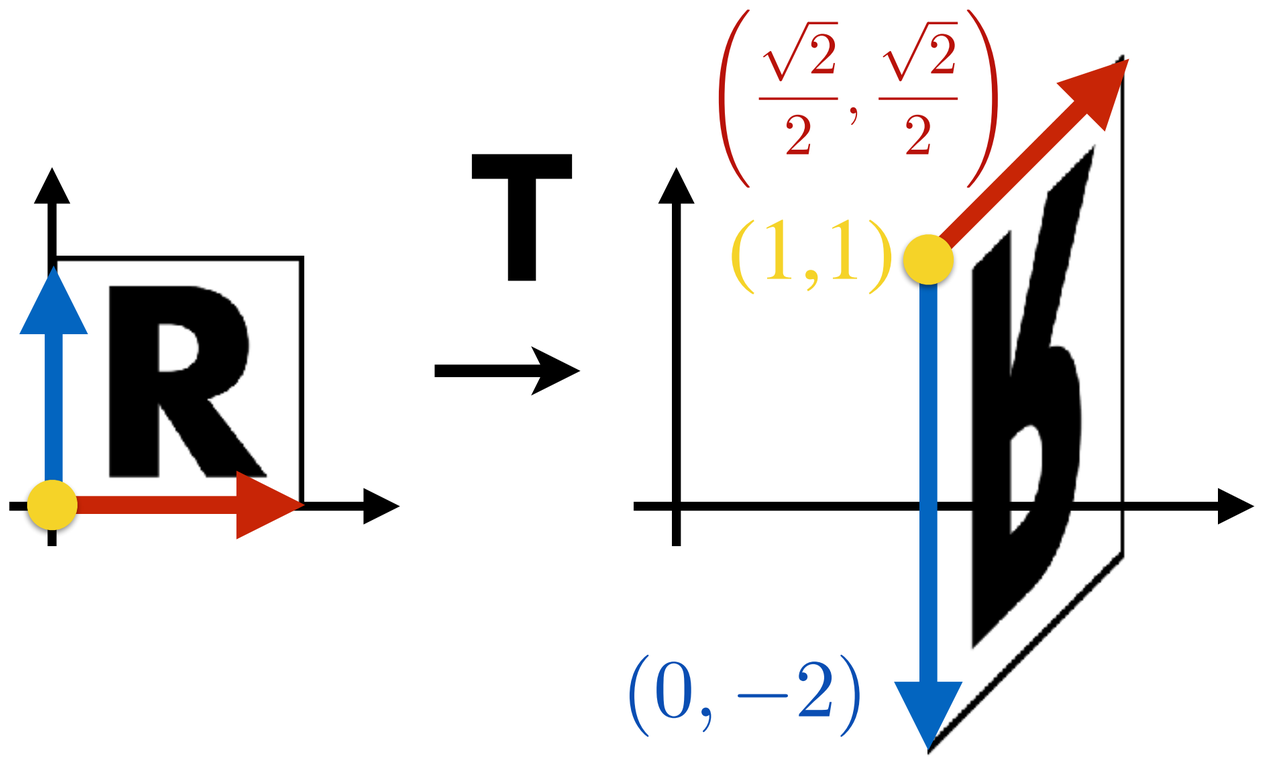
- 2D point = $(x, y, 1)^T$
- 2D vector = $(x, y, 0)^T$

Matrix representation of translations

$$\begin{pmatrix} x' \\ y' \\ w' \end{pmatrix} = \begin{pmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} x + t_x \\ y + t_y \\ 1 \end{pmatrix}$$

Coordinate systems

Write down a matrix T representing this transform:

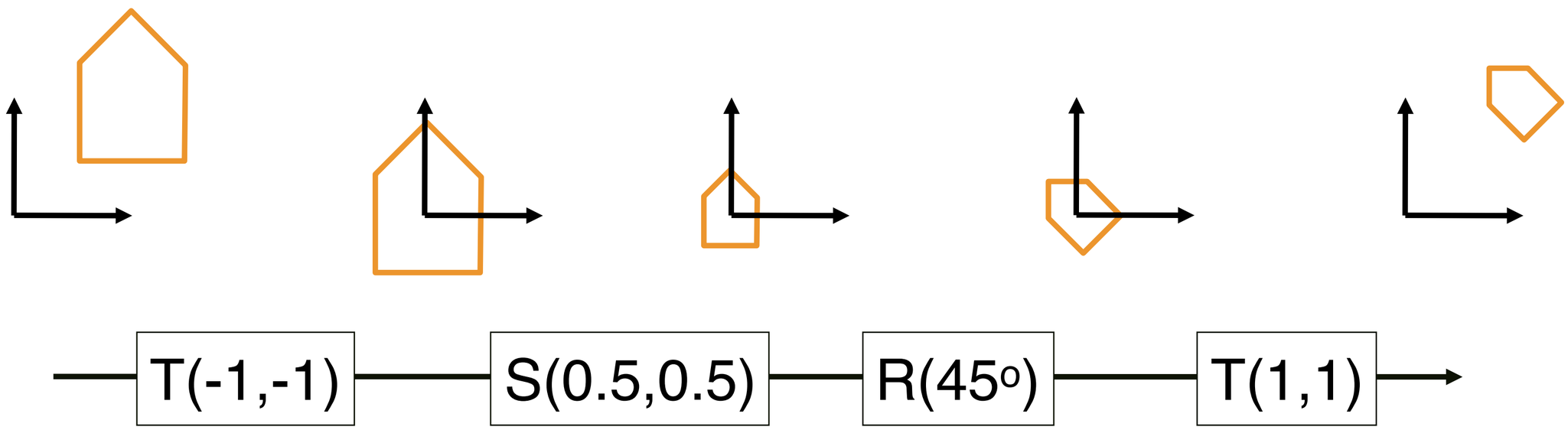


$$\begin{bmatrix} \mathbf{u} & \mathbf{v} & \mathbf{o} \\ 0 & 0 & 1 \end{bmatrix}$$

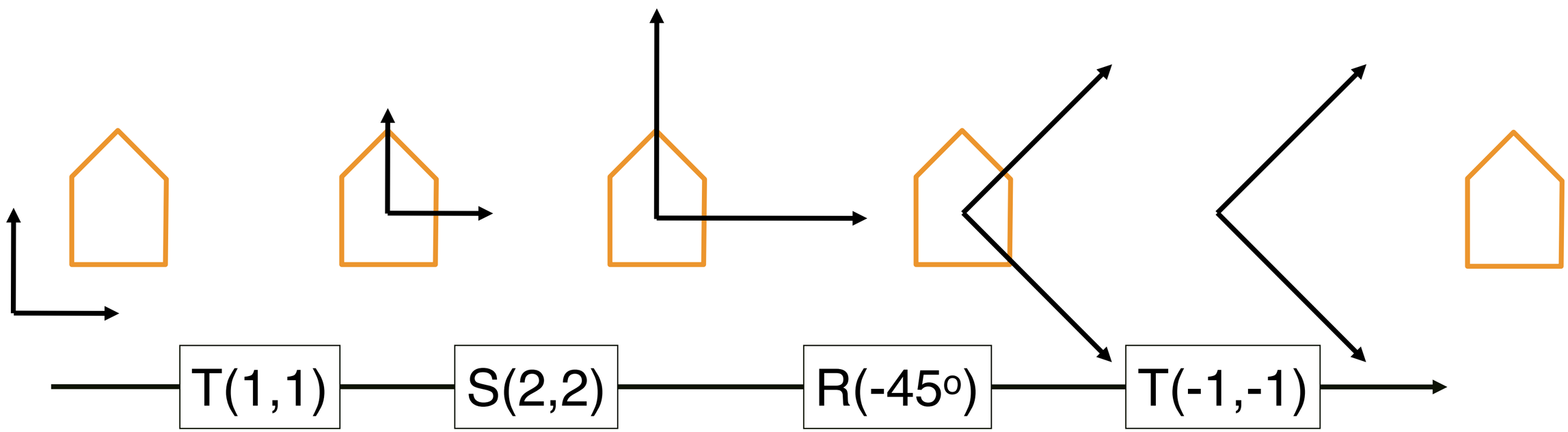
$$\begin{bmatrix} \frac{\sqrt{2}}{2} & 0 & 1 \\ \frac{\sqrt{2}}{2} & -2 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$

Coordinate systems

Interpretation 1: Transforms object points



Interpretation 2: Transforms coordinate system



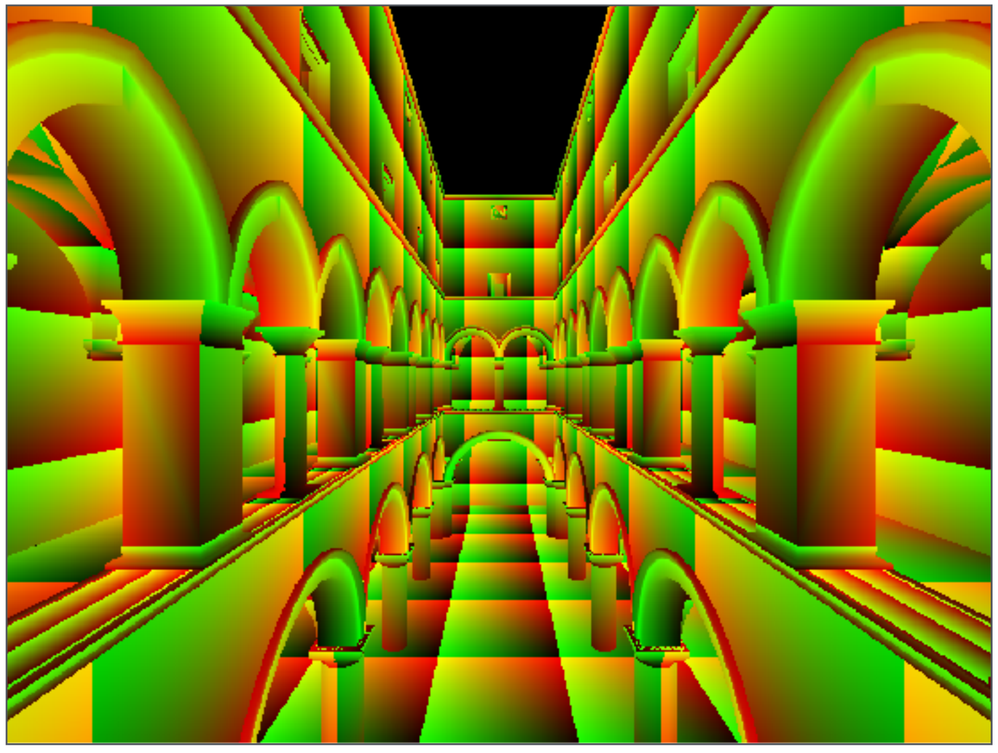
Texture Mapping

Core Ideas

- Translating between pixel space and texture space
  - Texture coordinates
  - Barycentric coordinates
- Applying textures is sampling
  - Sampling rate: Magnification vs. Minification
  - Screen pixel area vs. telex area (footprint)
- Antialiasing textures
  - Bilinear filtering
  - Mipmapping

Texture Coordinates (u-v coordinates)

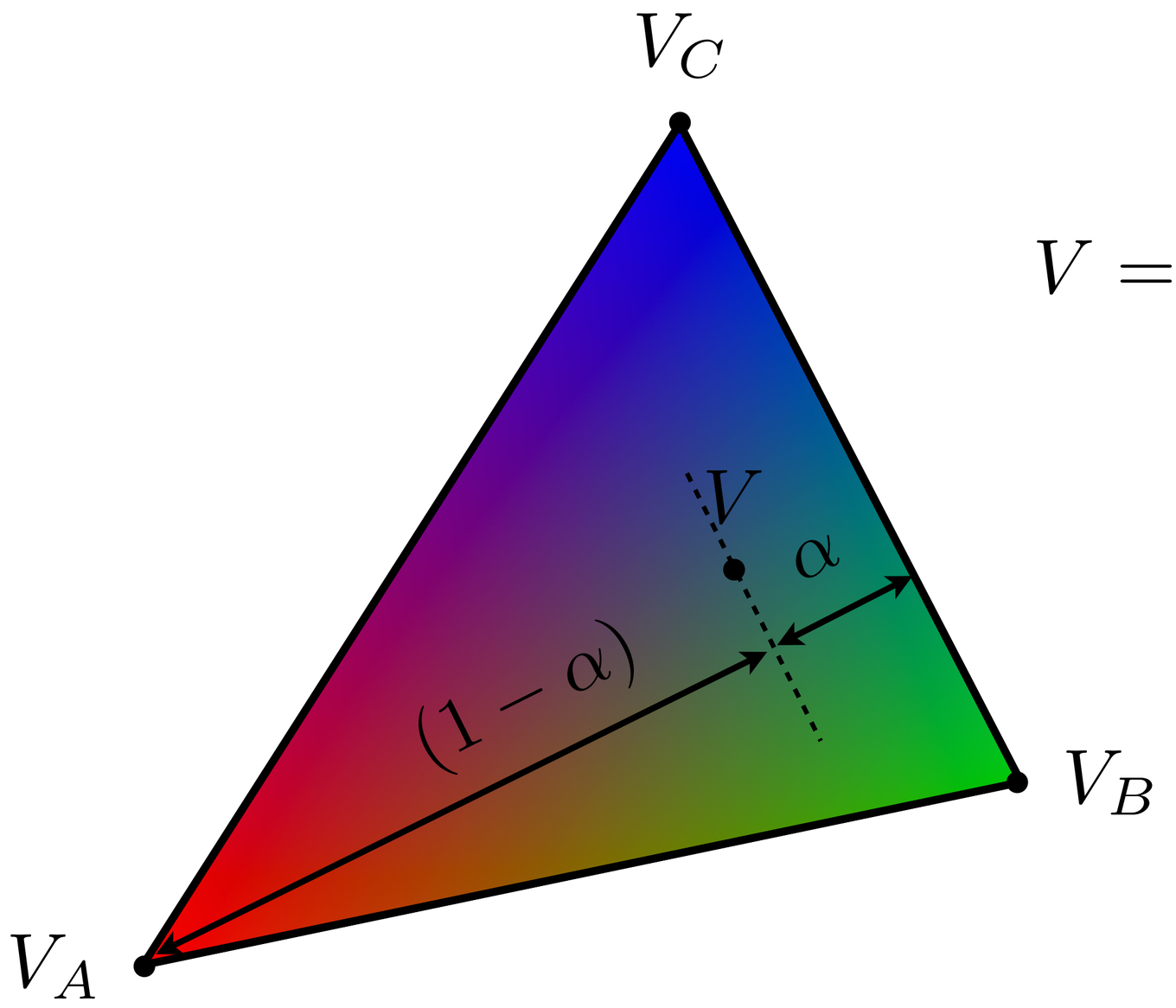
Sponza Palace Model



Visualization of texture coordinates

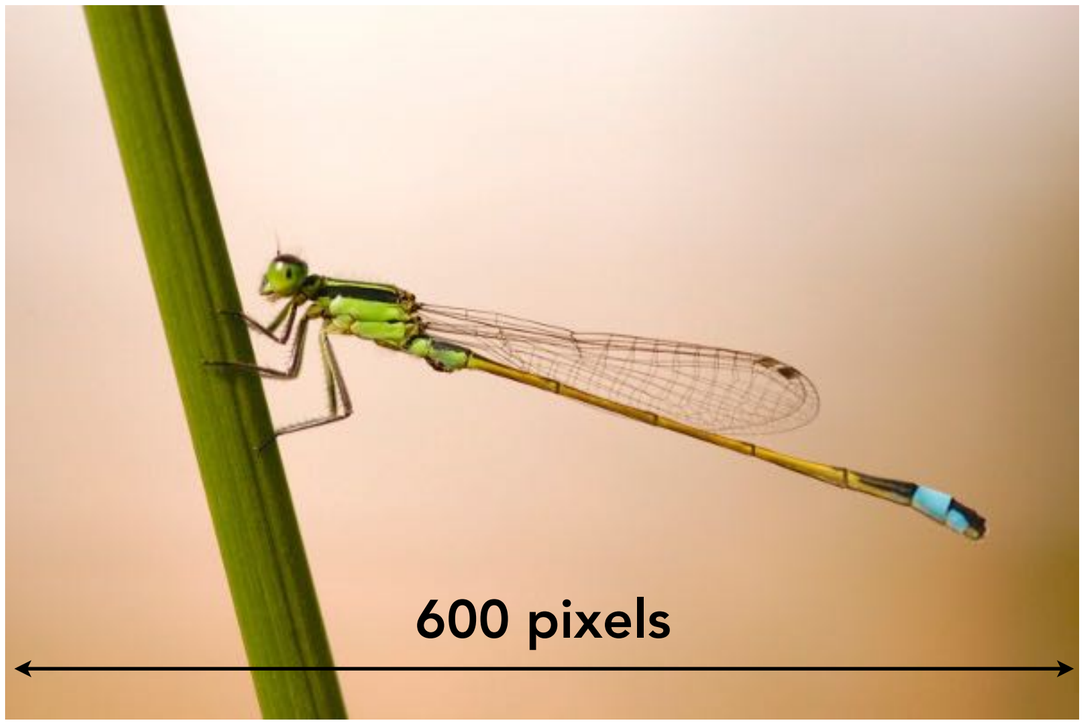
Example textures used

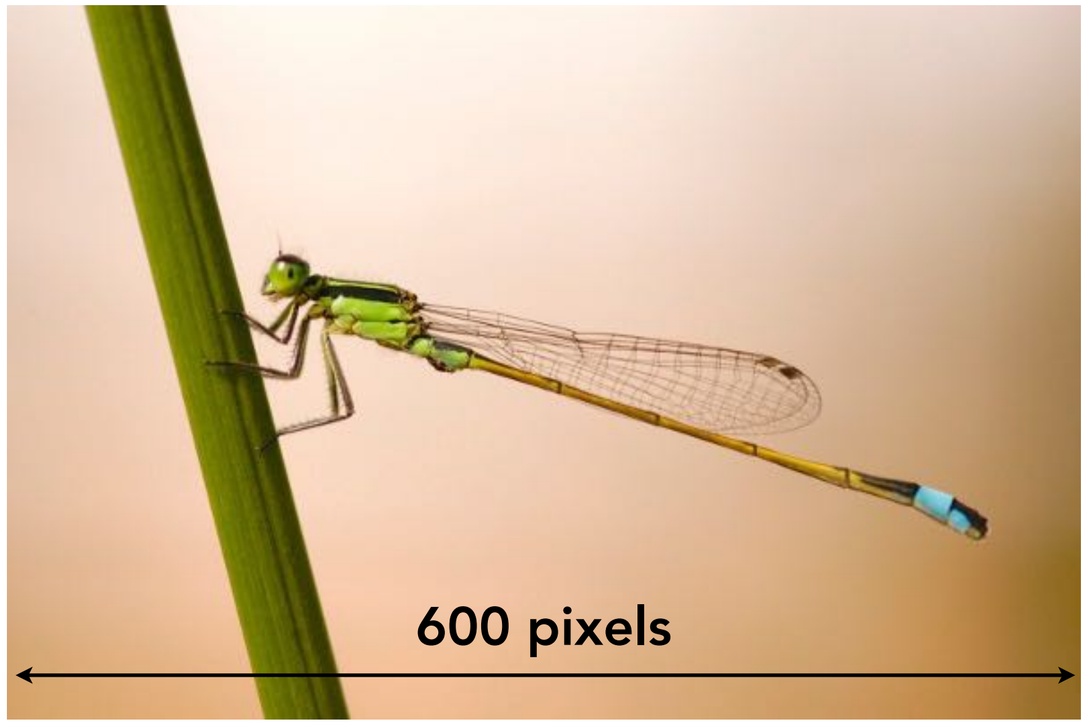
Barycentric Coordinates

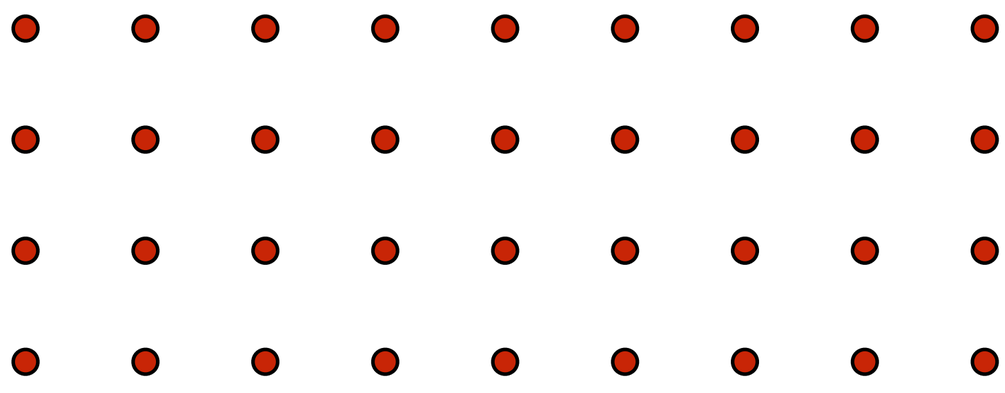


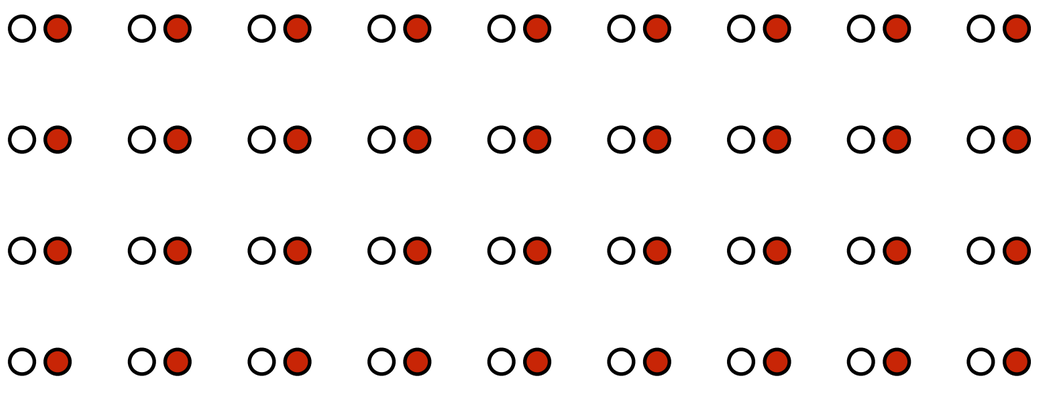
$$V = \alpha\, V_A + \beta\, V_B + \gamma\, V_C$$

Apply Textures = Sampling!









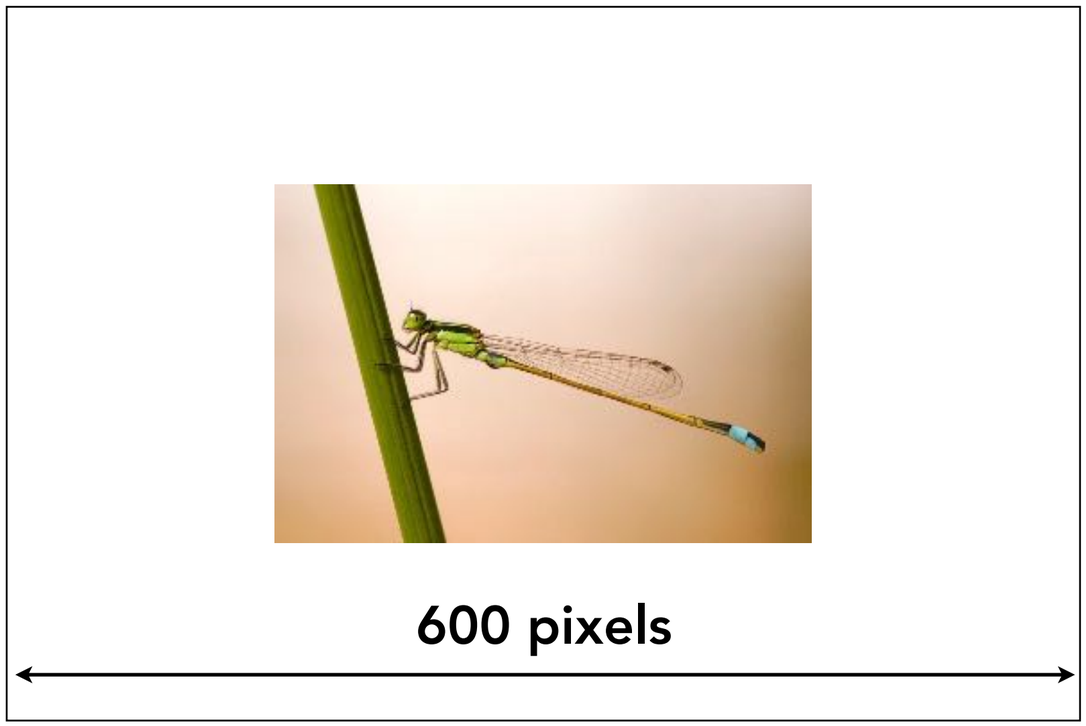
Screen space (x,y)

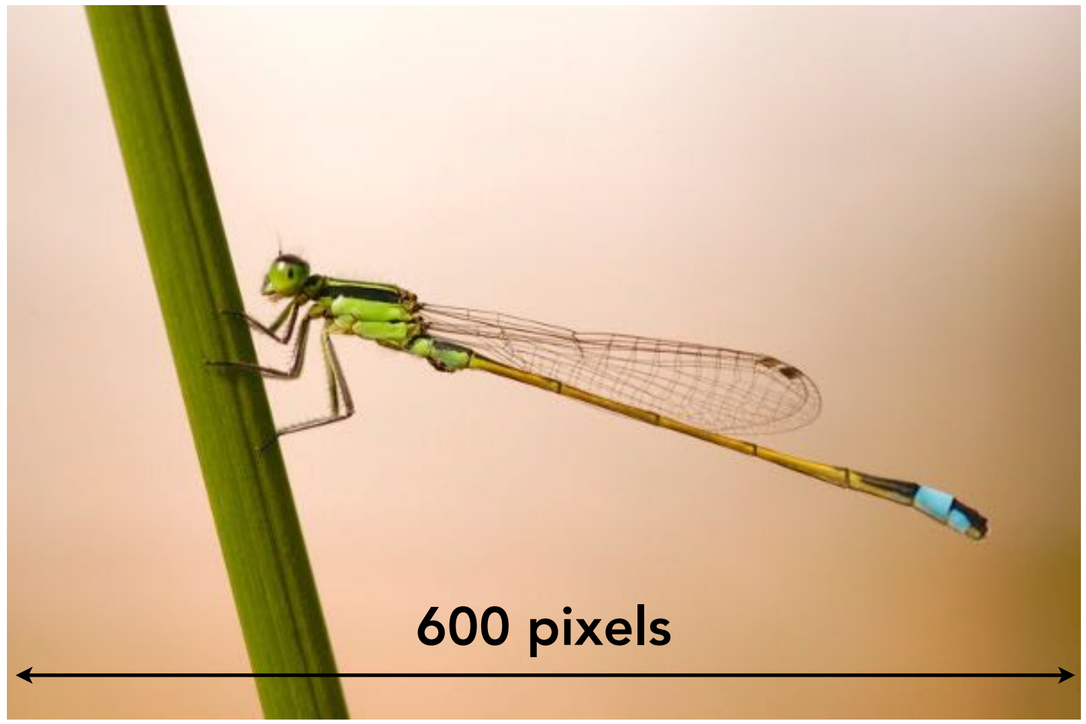
Texture space (u,v)

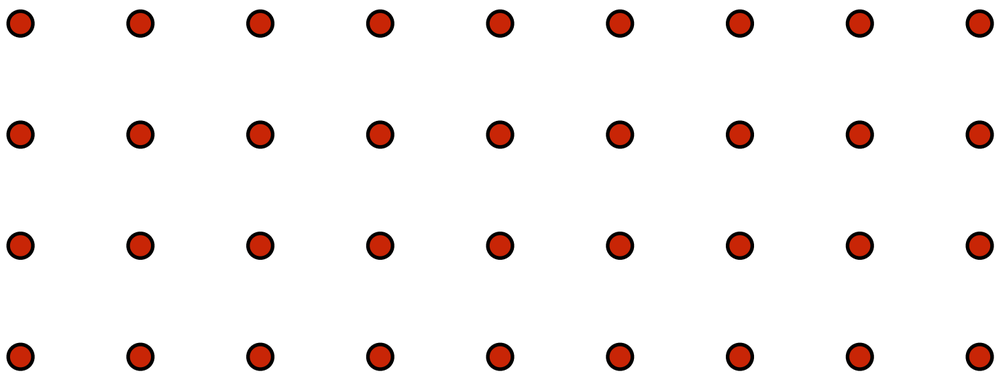
1:1 mapping

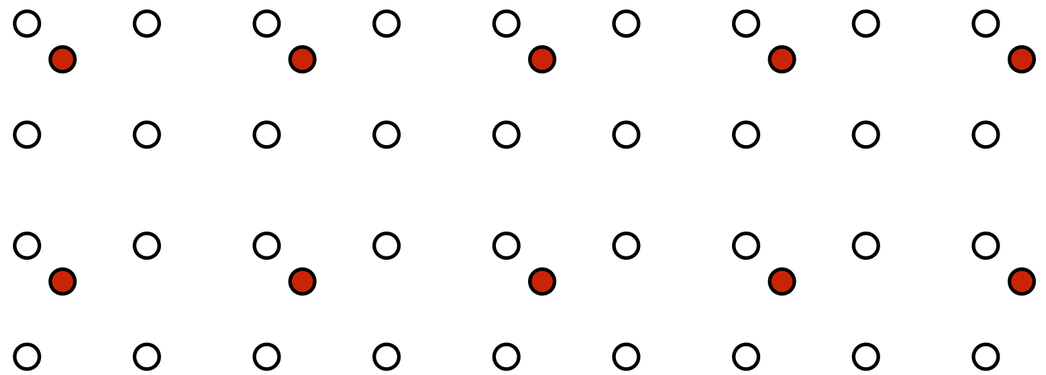
???

Apply Textures = Sampling!









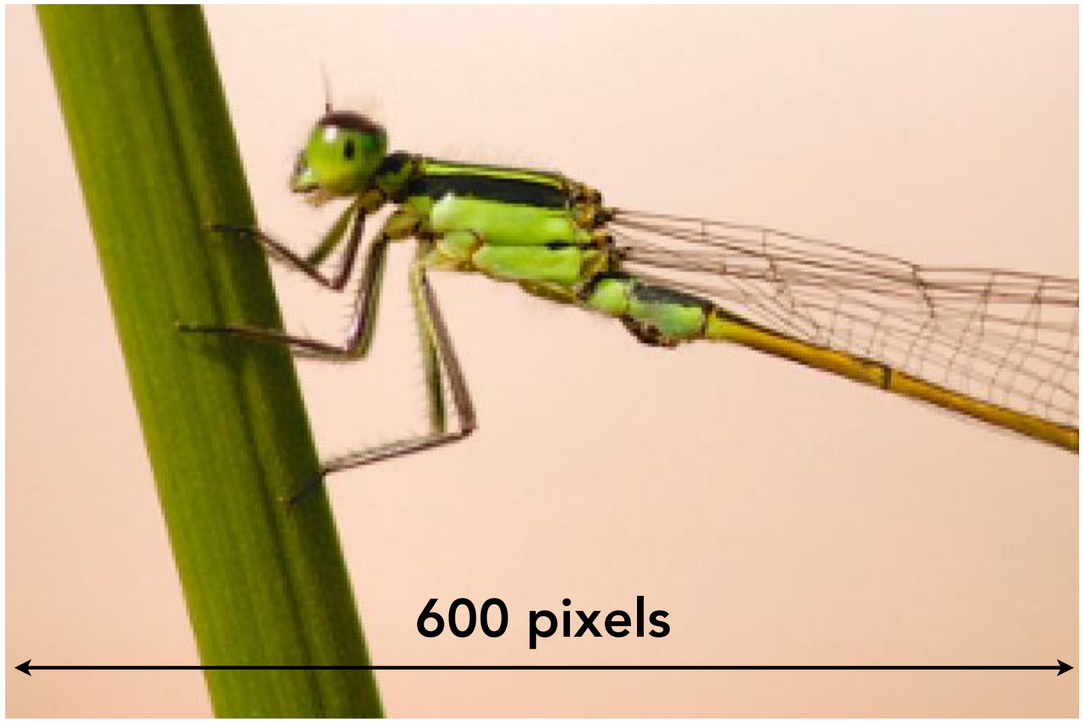
Screen space (x,y)

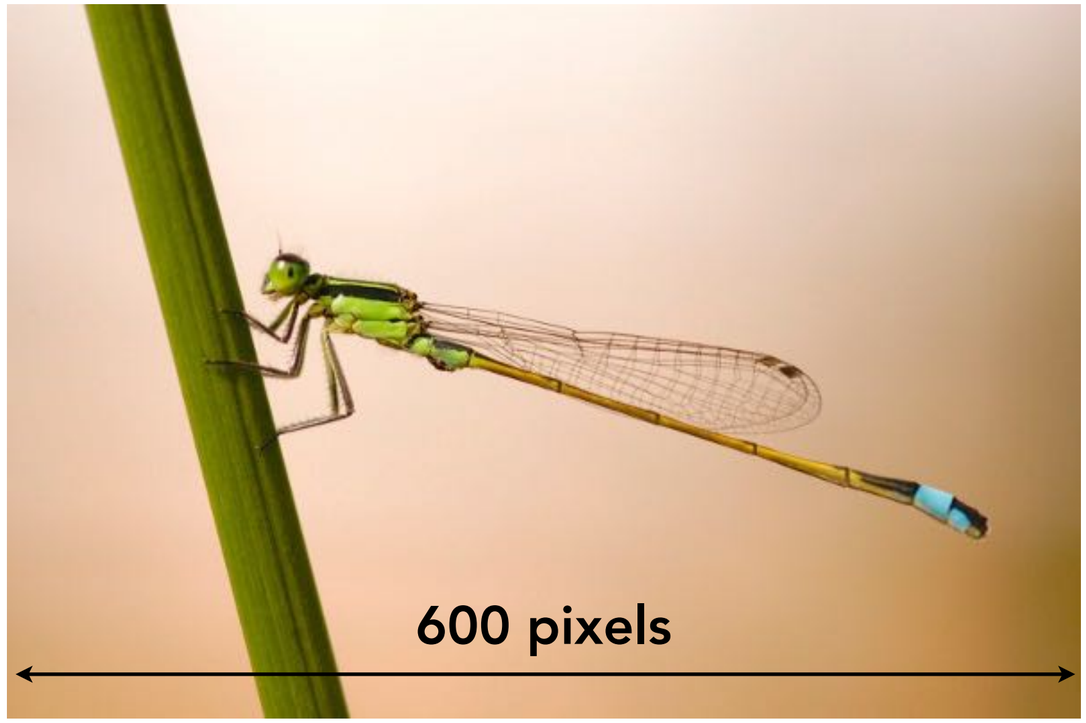
Texture space (u,v)

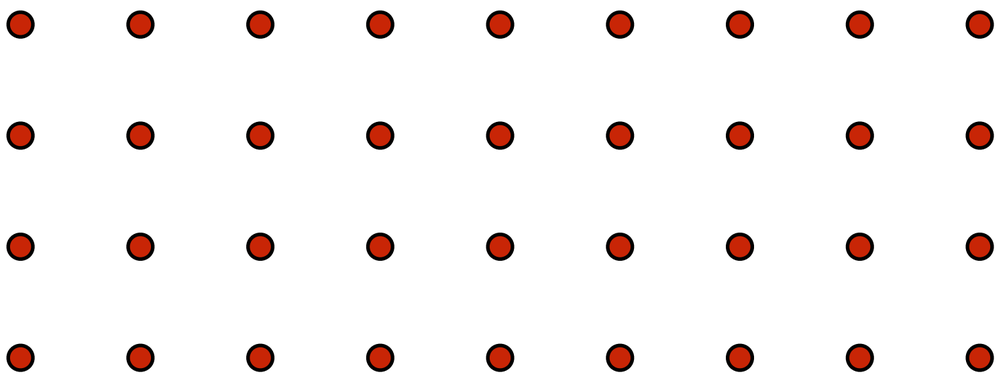
"Minified"

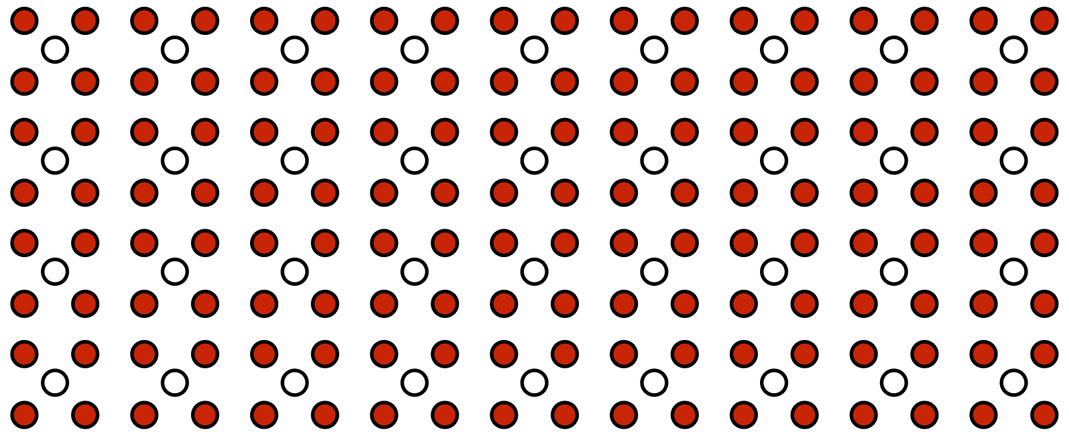
???

Apply Textures = Sampling!









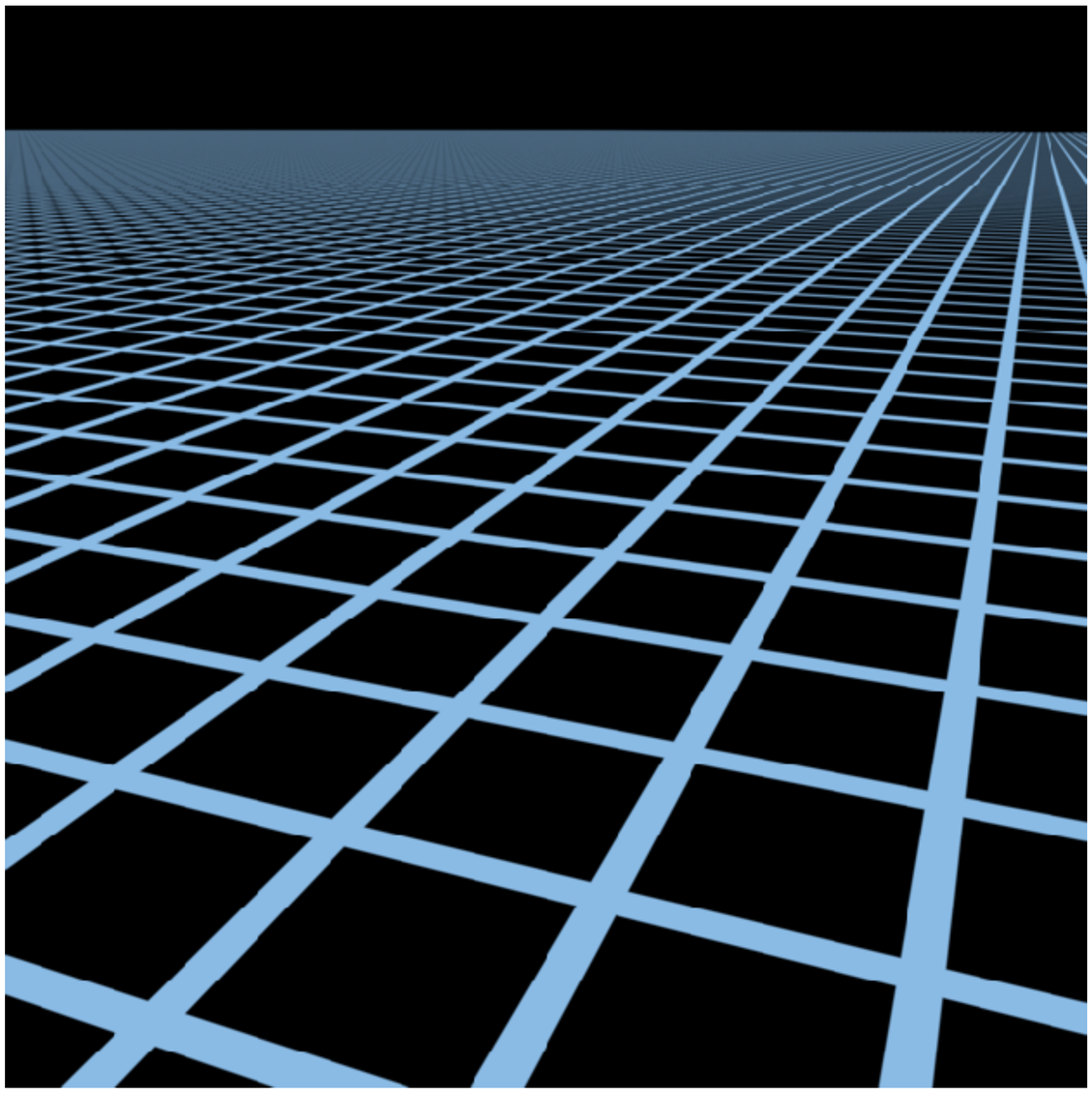
Screen space (x,y)

Texture space (u,v)

Magnified

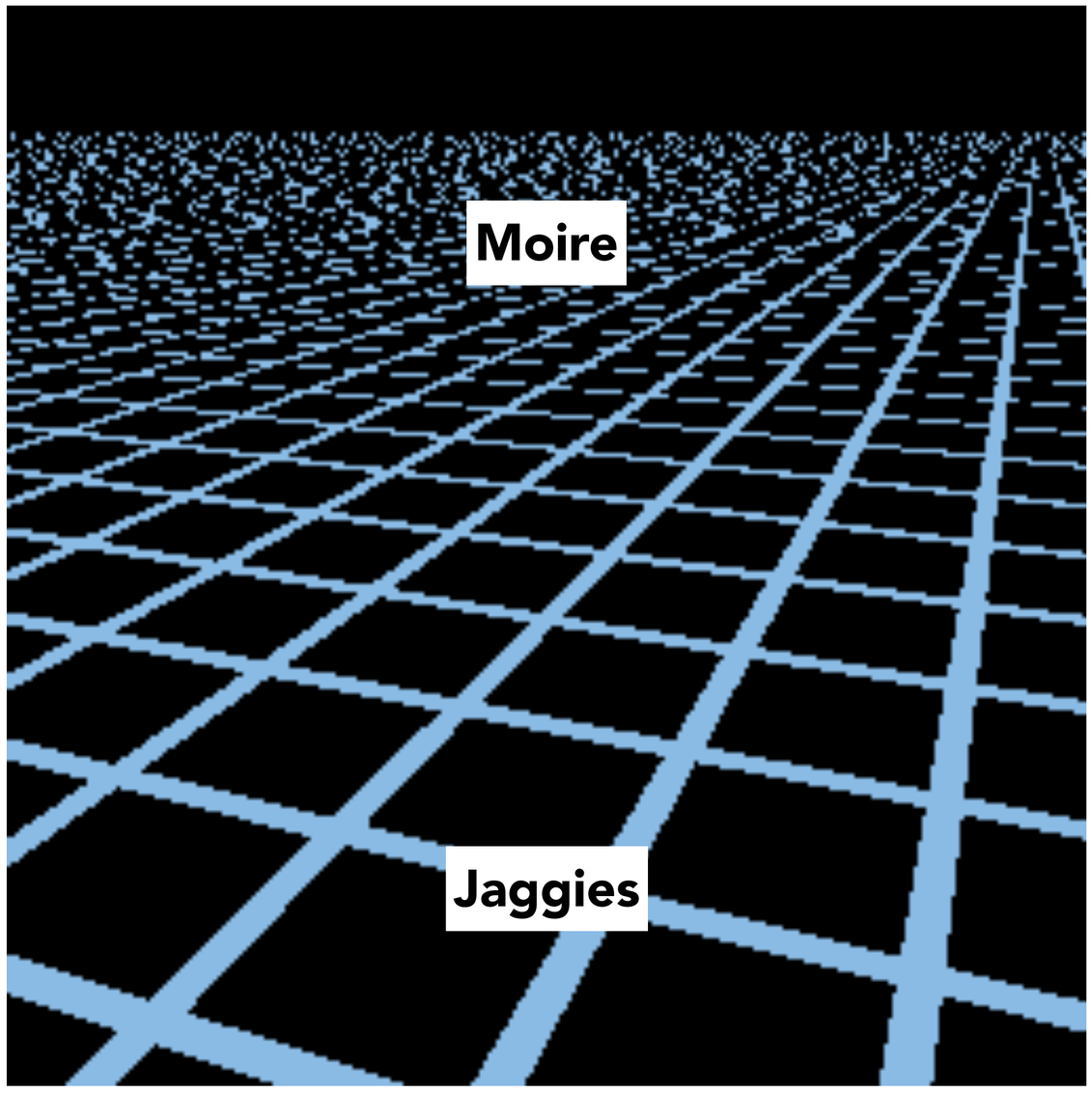
???

Apply Textures = Sampling!



High-res reference

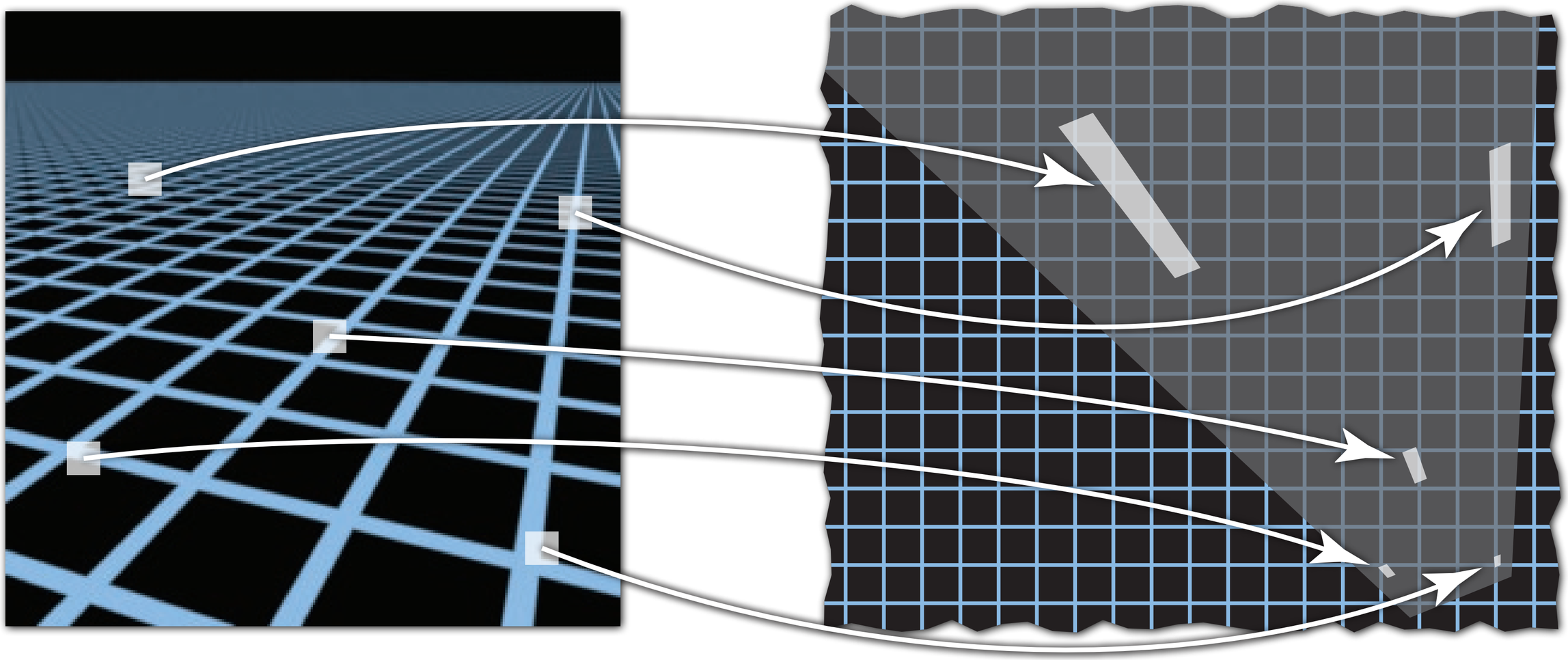
Source image: 1280x1280 pixels



Point sampling

256x256 pixels

Screen Pixel Footprint in Texture



Screen space Texture space

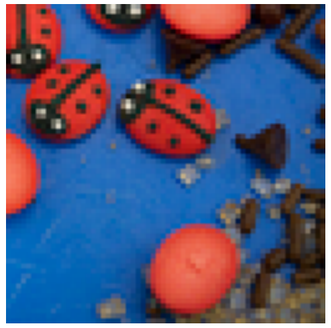
NB: texture sampling pattern not rectilinear or isotropic

Mipmaps

Store increasingly downsampled versions of the texture

Sample from the “appropriate” mipmap level

Level 0 = 128x128

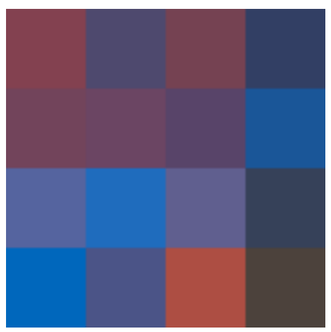


Level 1 = 64x64

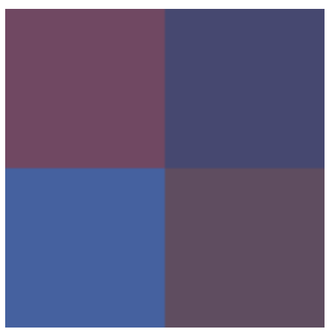
Level 2 = 32x32

Level 3 = 16x16

Level 4 = 8x8



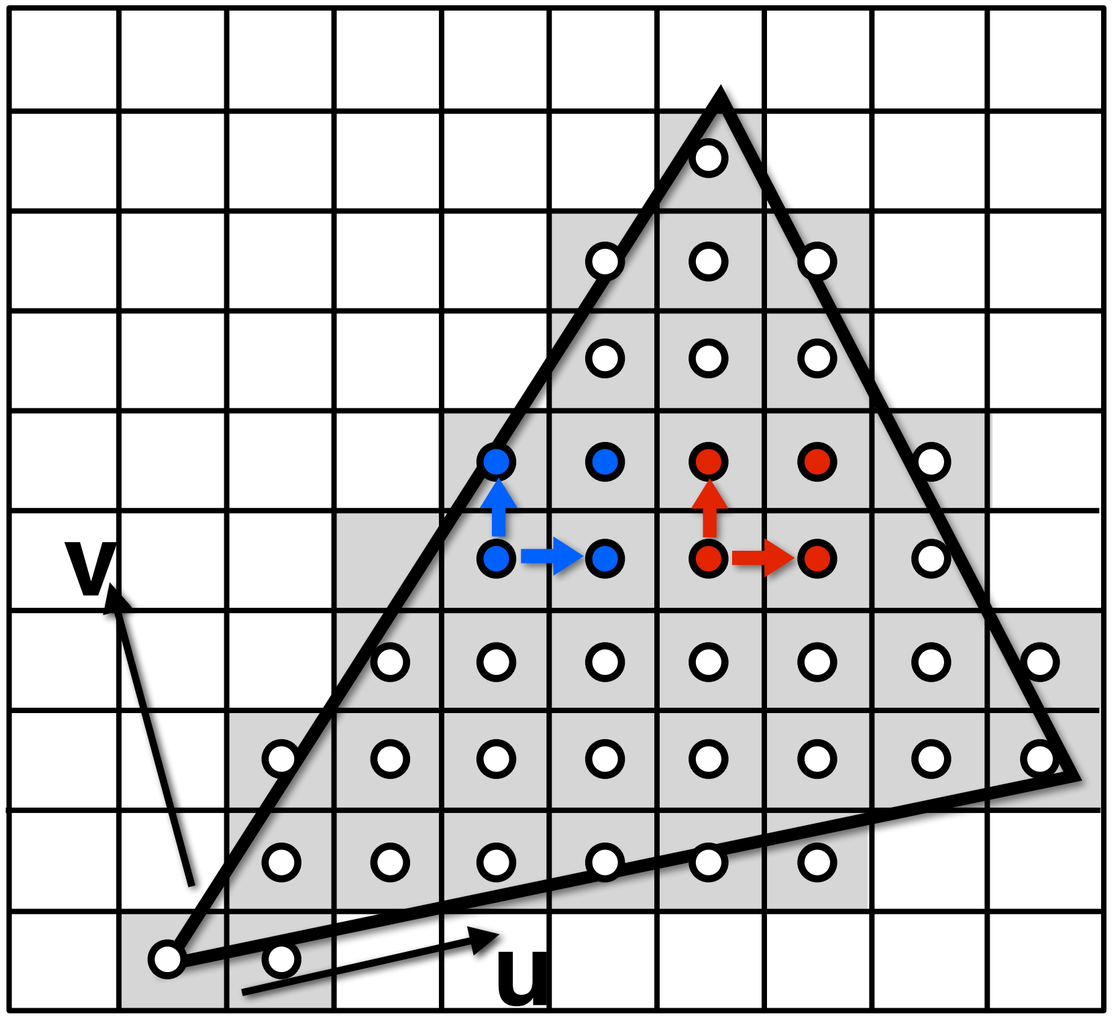
Level 5 = 4x4



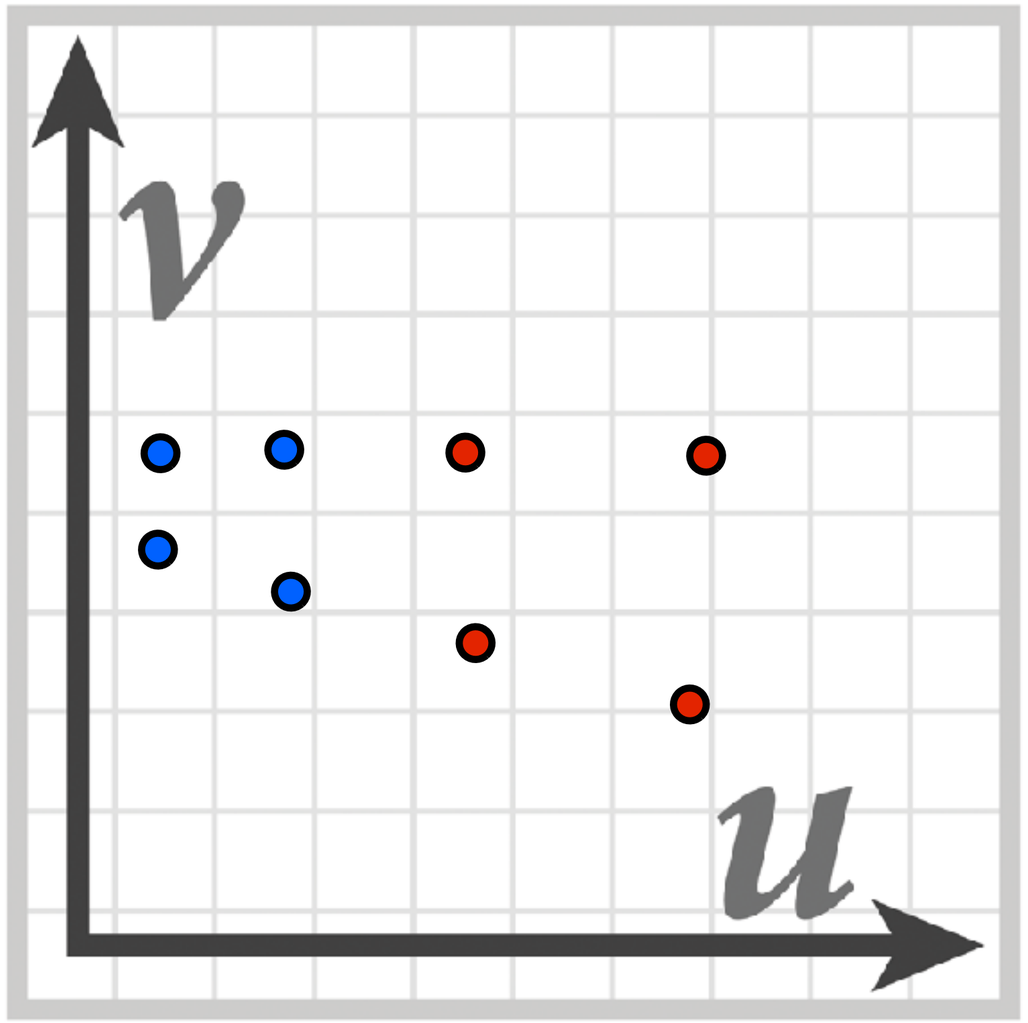
Level 6 = 2x2

Level 7 = 1x1

Computing Mipmap Level D



Screen space (x,y)



Texture space (u,v)

Estimate texture footprint using texture coordinates of neighboring screen samples

Demo!

Acknowledgments

Thanks to Kayvon Fatahalian, Steve Marschner and Mark Pauly for presentation resources.

Many thanks to Ren Ng for original lecture slides.