Discussion 03

Sampling & Texture Mapping

Computer Graphics and Imaging
UC Berkeley CS 184

Discussion 3 Announcements

- **Homework 1** has been released! Checkpoint is due on **Friday**. There are **no** slip days for checkpoints
- Homeworks can be completed with a partner. Search for a partner on Ed :)

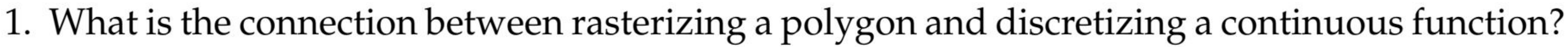
Sampling

1. What is the connection between rasterizing a polygon and discretizing a continuous function?

2. Why does aliasing occur? Describe or draw an example of aliasing of any kind.

3. What is the connection between applying a box blur to an image and supersampling each pixel within the image?

1. What is the connection between rasterizing a polygon and discretizing a continuous function?



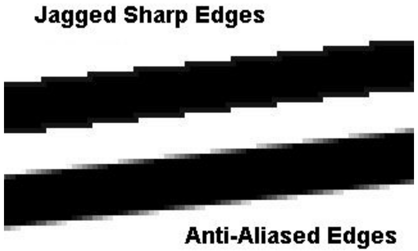
1. What is the connection between rasterizing a polygon and discretizing a continuous function?

- For each triangle **T**, define a function inside(**T**, **x**, **y**) that is 1 if (**x**, **y**) is inside **T** and 0 otherwise.
- Rasterizing a polygon is the same as discretizing inside!

2. Why does aliasing occur? Describe or draw an example of aliasing of any kind.

2. Why does aliasing occur? Describe or draw an example of aliasing of any kind.

Aliasing occurs when high-frequency signals are under-sampled.

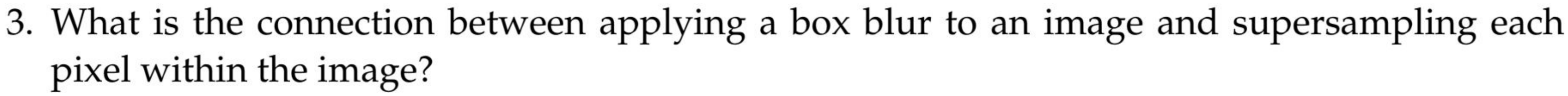


Jaggies

Wagon wheel effect

Moire effect

3. What is the connection between applying a box blur to an image and supersampling each pixel within the image?



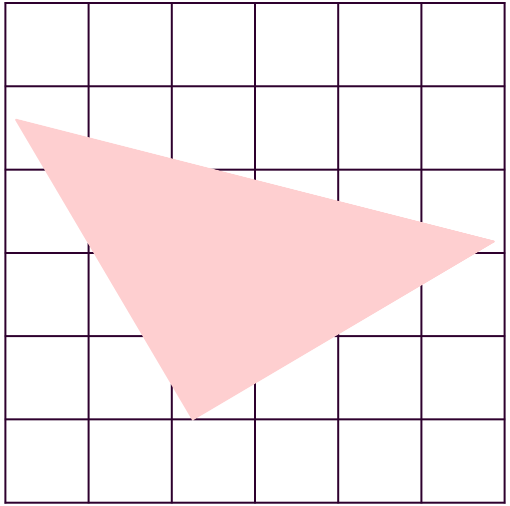
3. What is the connection between applying a box blur to an image and supersampling each pixel within the image?

- In each case, we take subsamples from each pixel and average the subsamples.
- 1-pixel box blur is a continuous version of (discrete) supersampling!

Rasterization

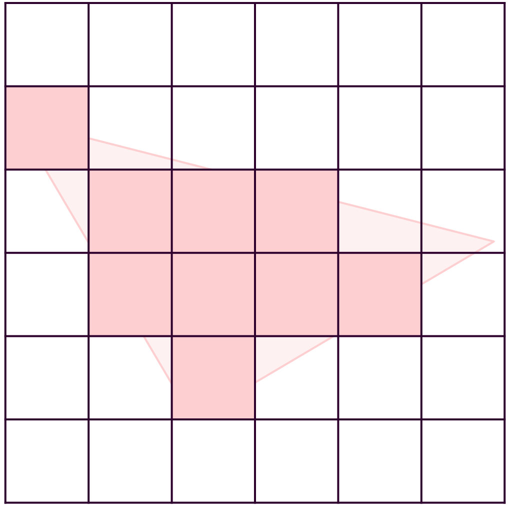
Rasterization

- Screens are *raster* displays.
- We can specify our graphics with triangles → but how do these triangles actually render on a screen?



Rasterization

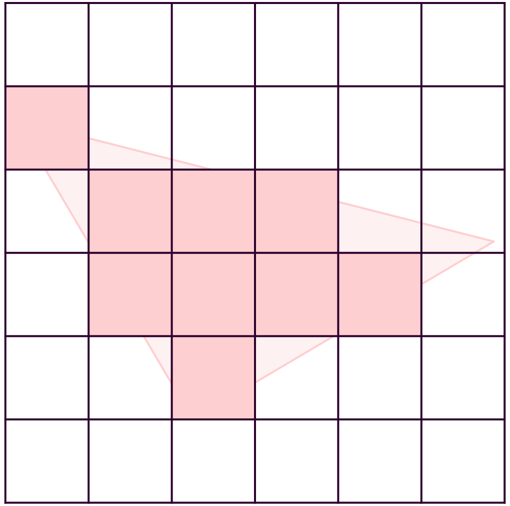
- Screens are *raster* displays.
- We can specify our graphics with triangles → but how do these triangles actually render on a screen?



Rasterization

- Screens are *raster* displays.
- We can specify our graphics with triangles → but how do these triangles actually render on a screen?

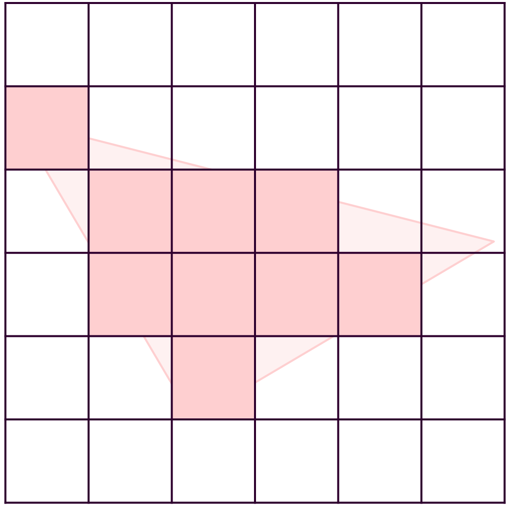
Sample the triangle at <u>discrete points</u> to approximate the continuous triangle signal!



Rasterization

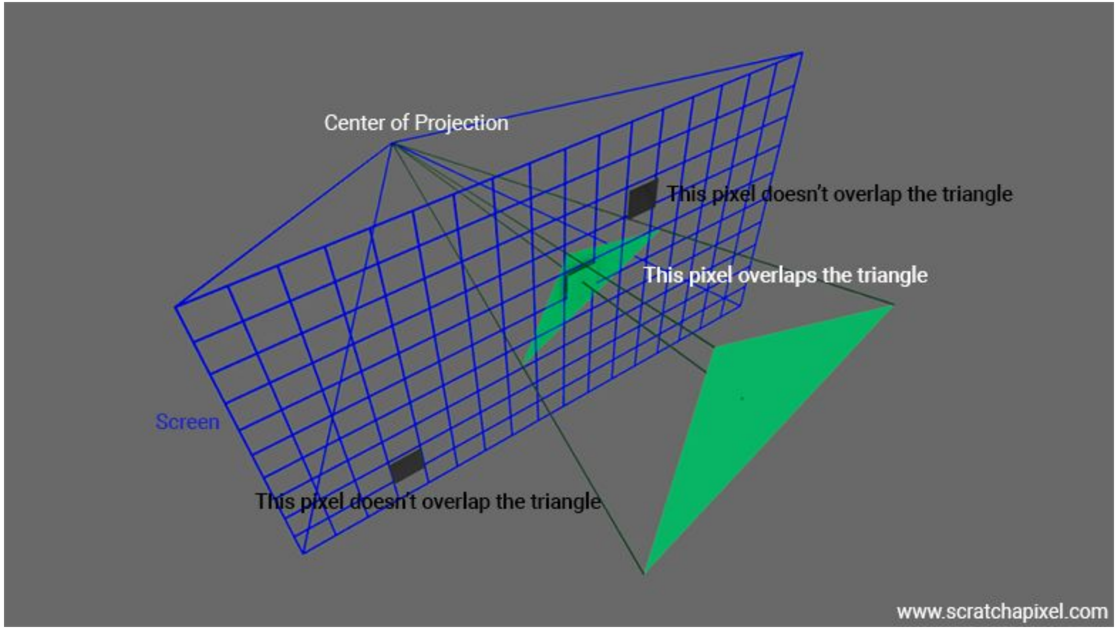
- Screens are *raster* displays.
- We can specify our graphics with triangles → but how do these triangles actually render on a screen?

Sample the triangle at <u>discrete points</u> to approximate the continuous triangle signal!



Sidenote. If you're viewing this presentation, you're viewing a raster display!

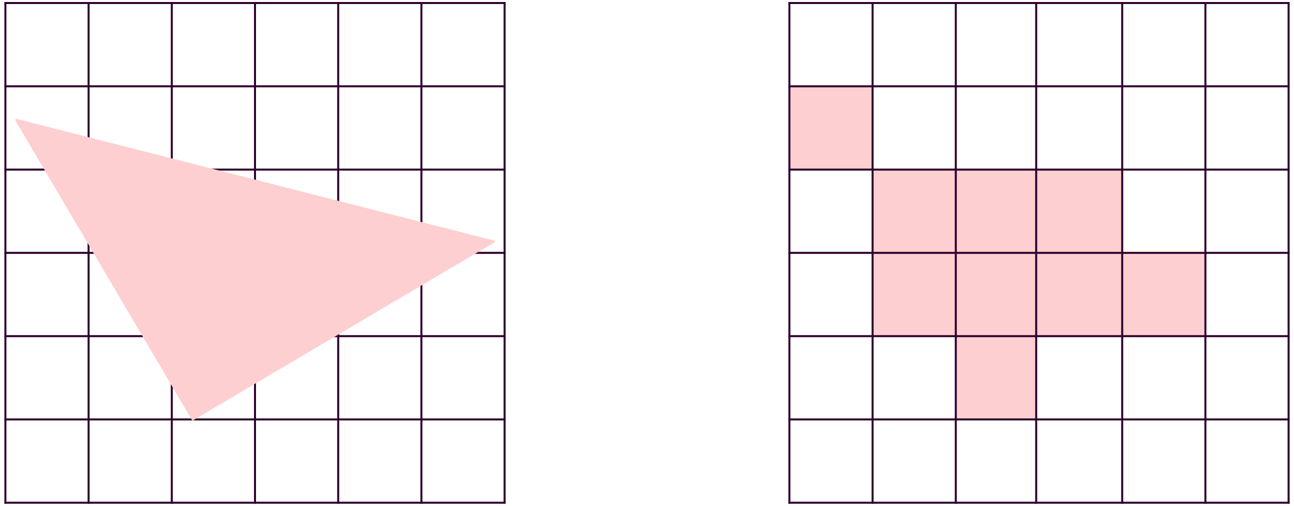
Even the original pink triangle is the rasterization of a triangle defined in continuous 2D space 😁


Center of Projection
This pixel doesn't overlap the triangle
This pixel overlaps the triangle
Screen
This pixel doesn't overlap the triangle
www.scratchapixel.com

Aliasing

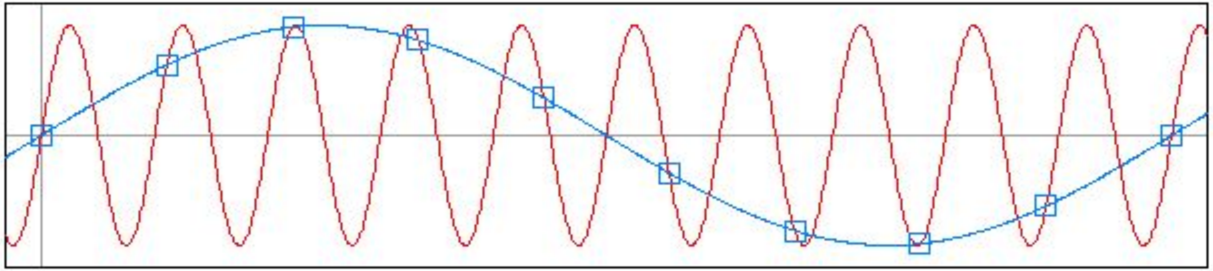
Aliasing

- Aliasing occurs when a high-frequency signal is insufficiently sampled.



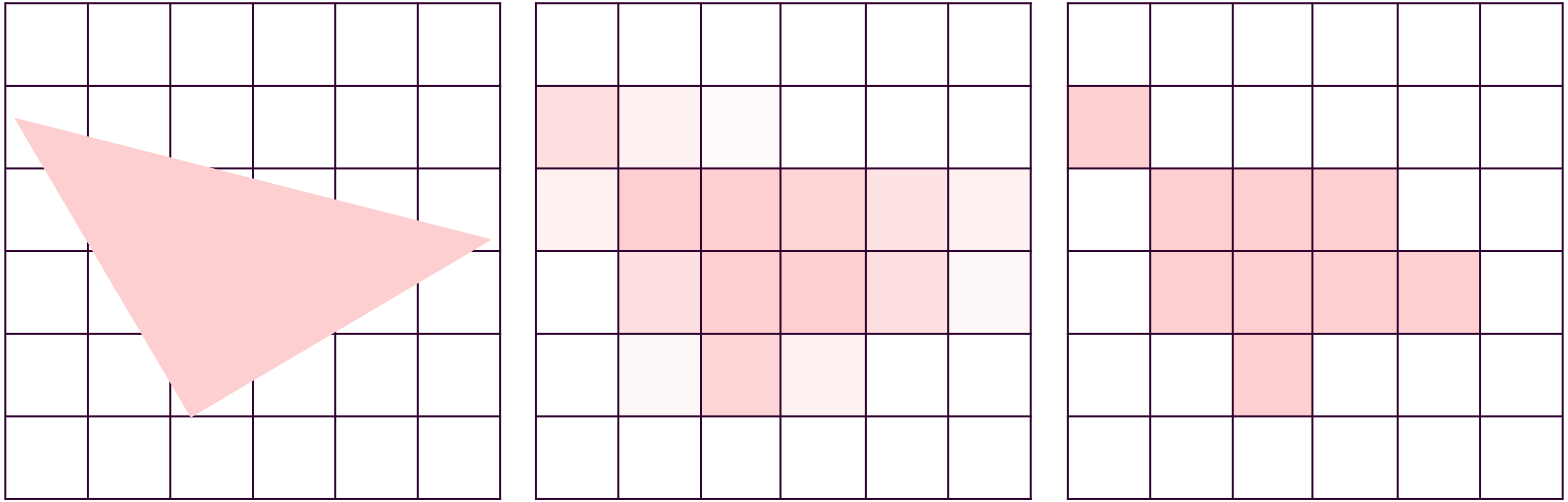
Aliasing

- Aliasing occurs when a high-frequency signal is insufficiently sampled.
- An insufficiently sampled signal could erroneously resemble a continuous signal of lower frequency.



Aliasing

- Aliasing occurs when a high-frequency signal is insufficiently sampled.



Supersampling

Anti-Aliasing

Anti-Aliasing

- What if we removed high-frequency signals *before sampling*?

Original image

@ Photo by Jan van der Wolf

Anti-Aliasing

- What if we removed high-frequency signals *before sampling*?
- High-frequency spatial signals → abrupt spatial changes, like <u>edges</u>.

Original image

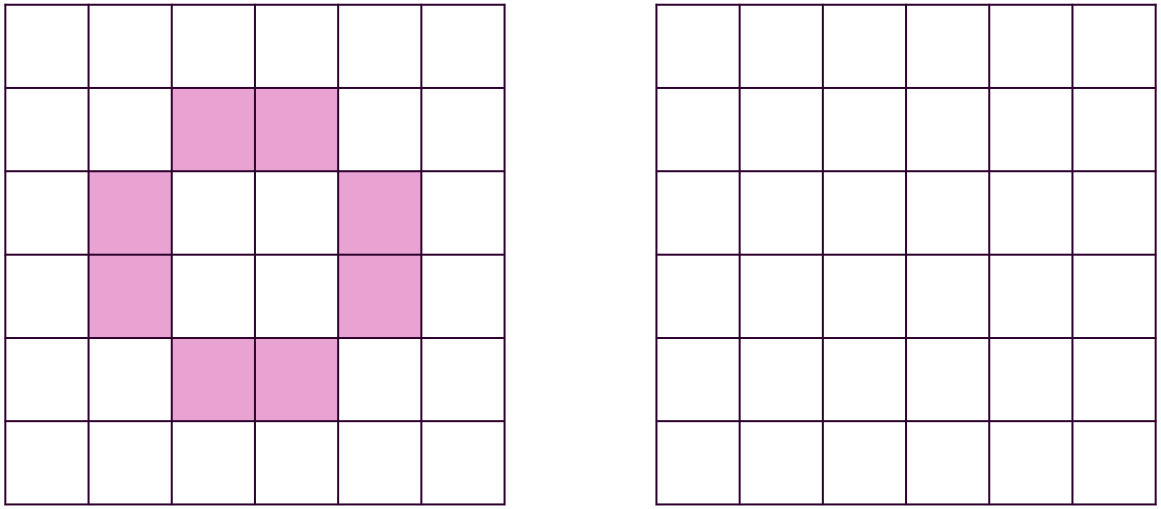
@ Photo by Jan van der Wolf

Anti-Aliasing

- What if we removed high-frequency signals *before sampling*?
- High-frequency spatial signals → abrupt spatial changes, like edges.
- As a result, filtering out high-frequencies → blurring image!

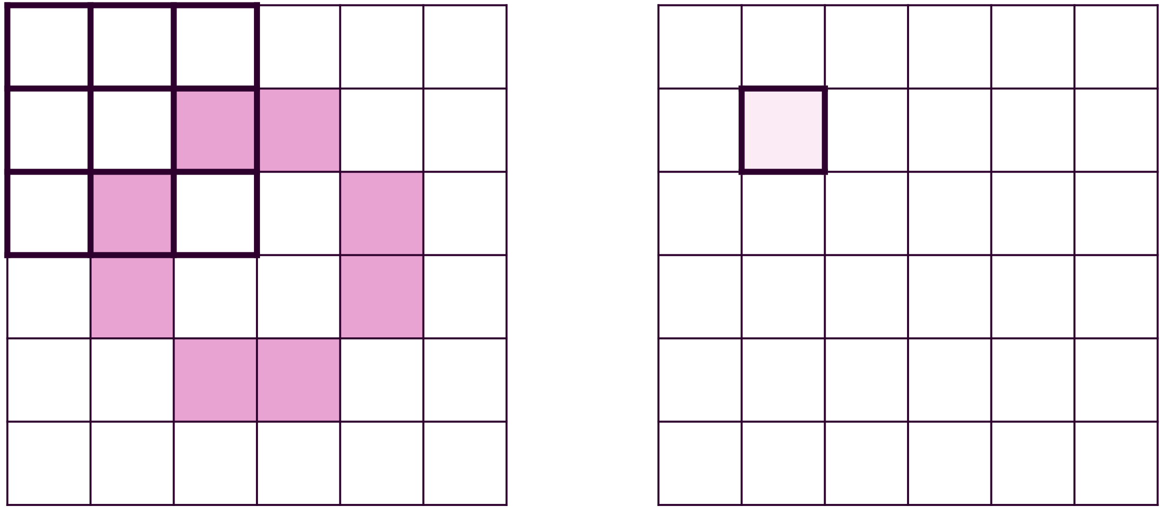
Box Blur

- Each pixel value is the average of the neighboring pixel values.



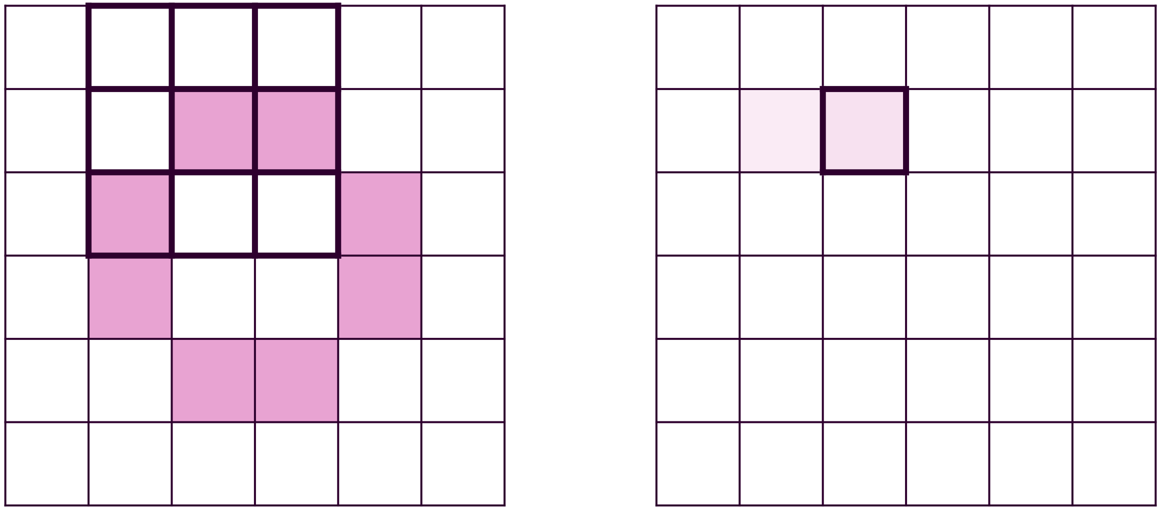
Box Blur

- Each pixel value is the average of the neighboring pixel values.



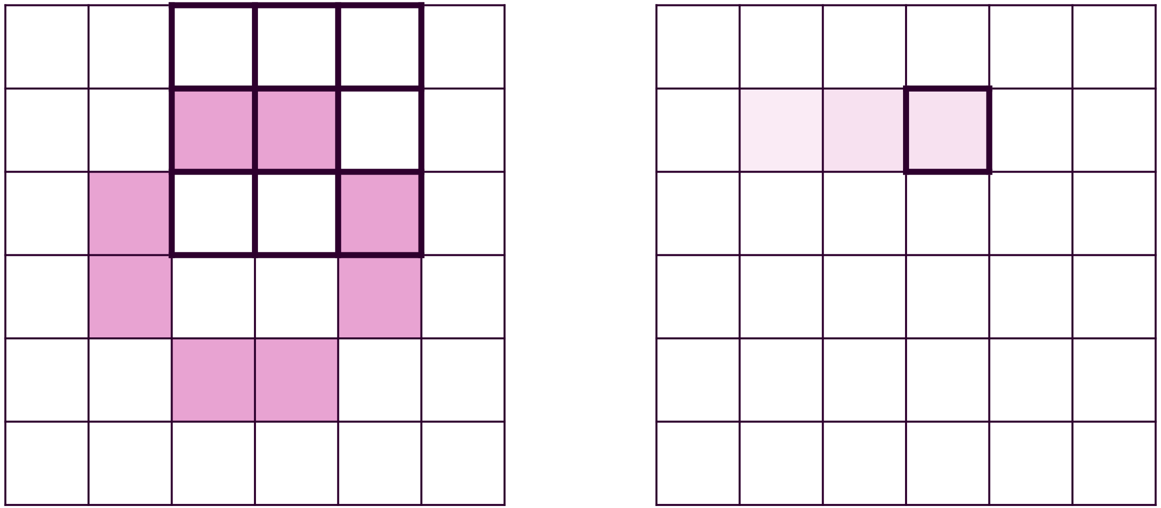
Box Blur

- Each pixel value is the average of the neighboring pixel values.



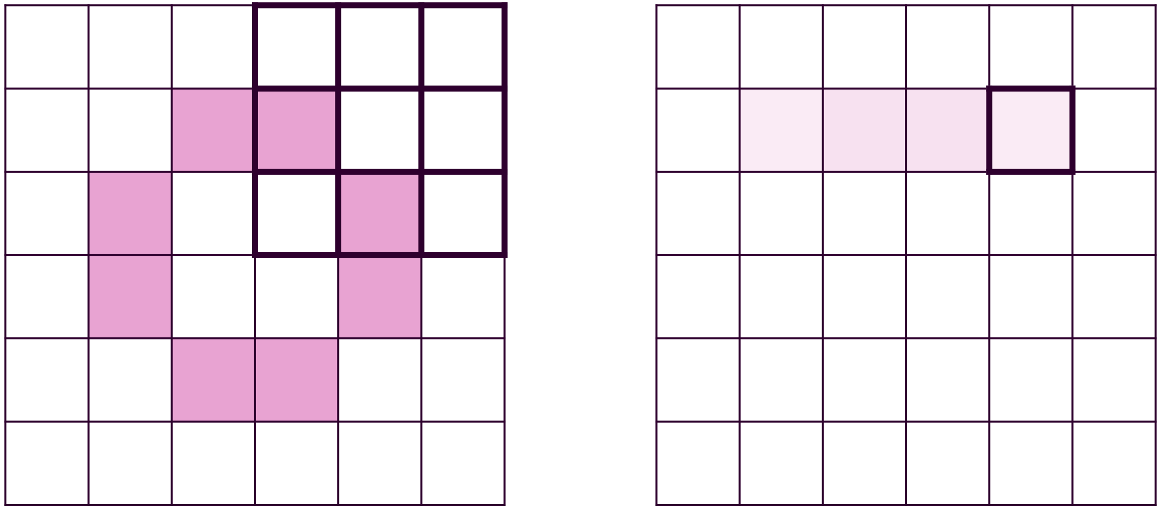
Box Blur

- Each pixel value is the average of the neighboring pixel values.



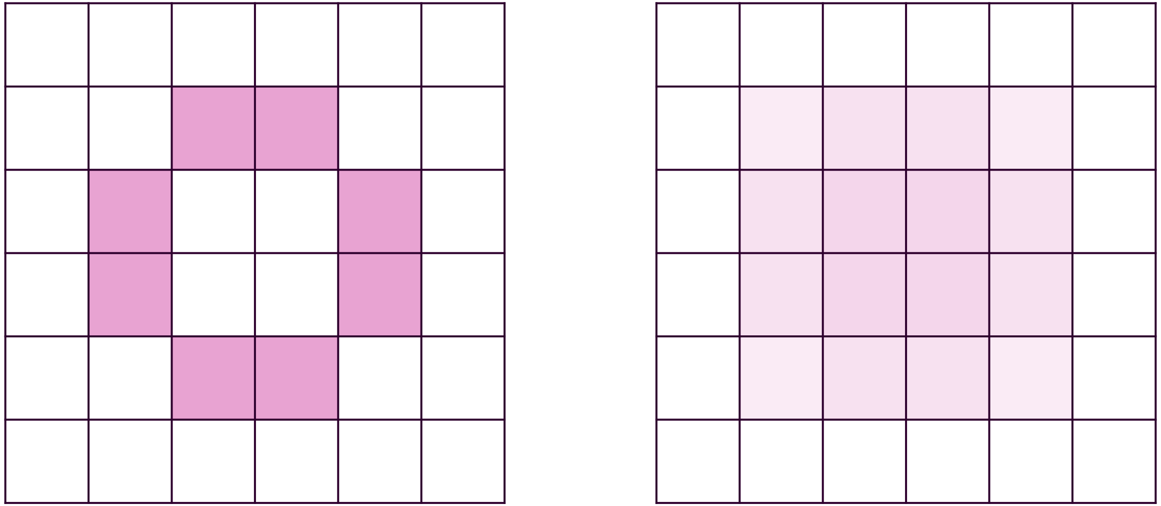
Box Blur

- Each pixel value is the average of the neighboring pixel values.



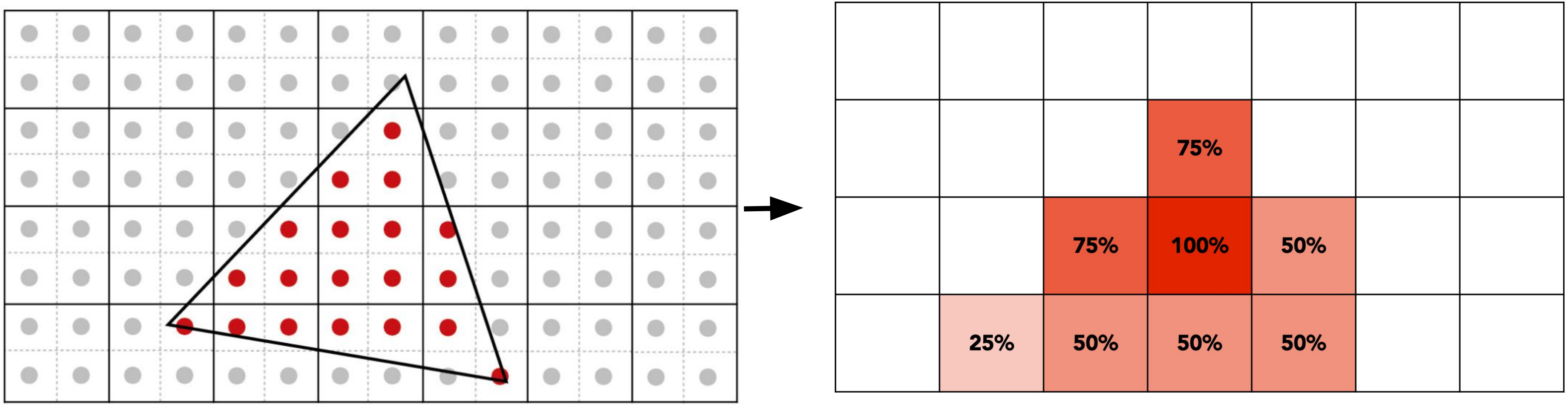
Box Blur

- Each pixel value is the average of the neighboring pixel values.
- Approximates convolution of continuous signal (low-pass filter).



How to reduce aliasing?

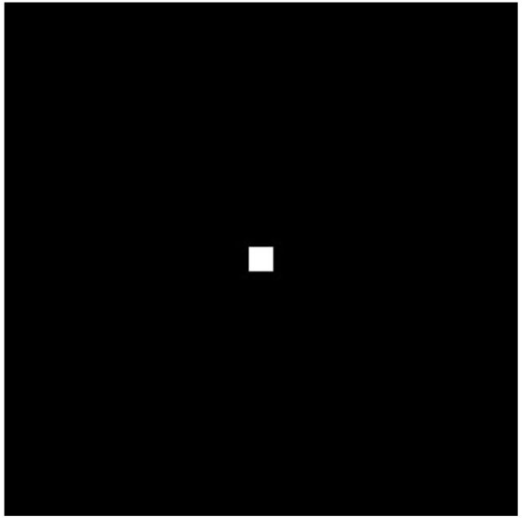
Supersampling: artificially increase the sampling rate above sampling frequency



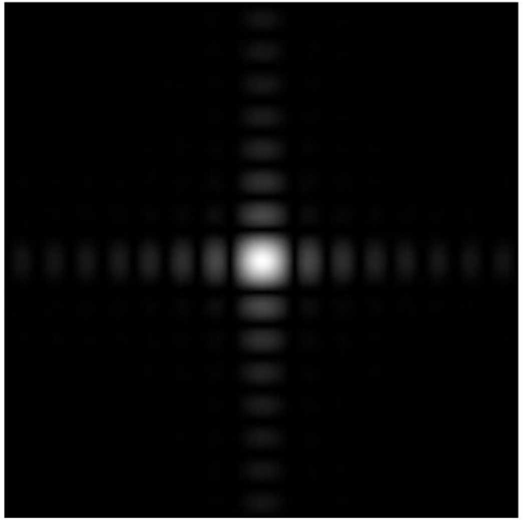
			75%			
		75%	100%	50%		
	25%	50%	50%	50%		

How to reduce aliasing?

1-pixel box filter: average over a 1-pixel area – low pass filter!

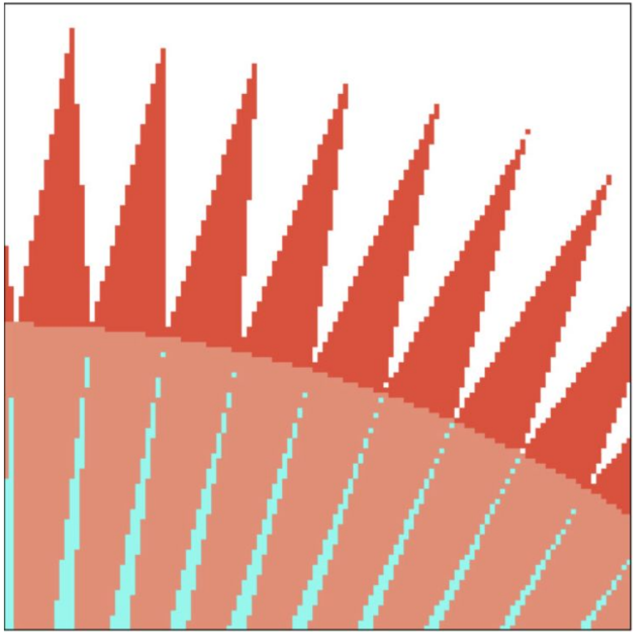


Spatial Domain

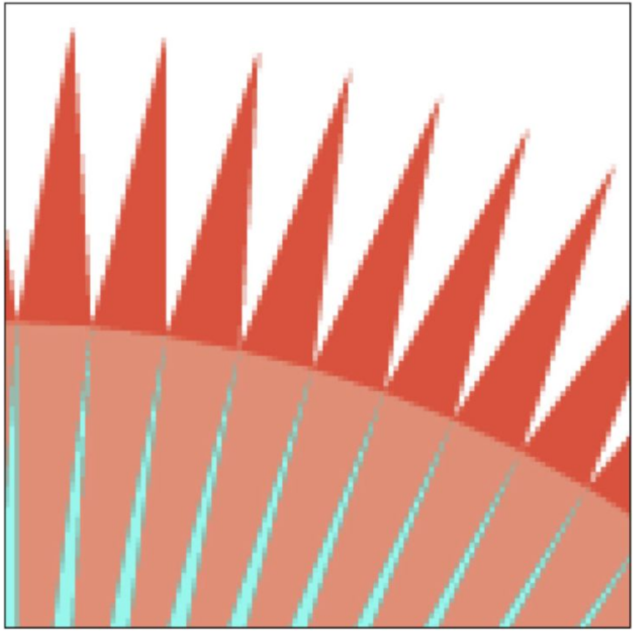


Frequency Domain

You’ll implement supersampling in HW1 !



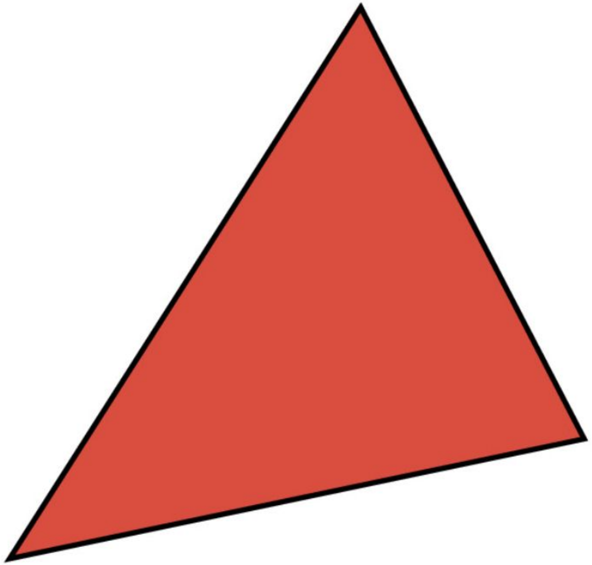
aliased



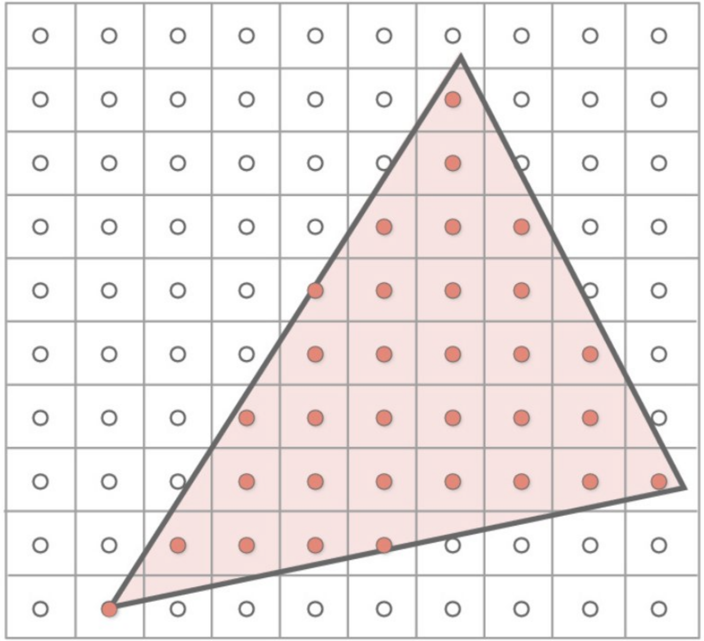
anti-aliased

Sampling & Aliasing

Sampling a Triangle

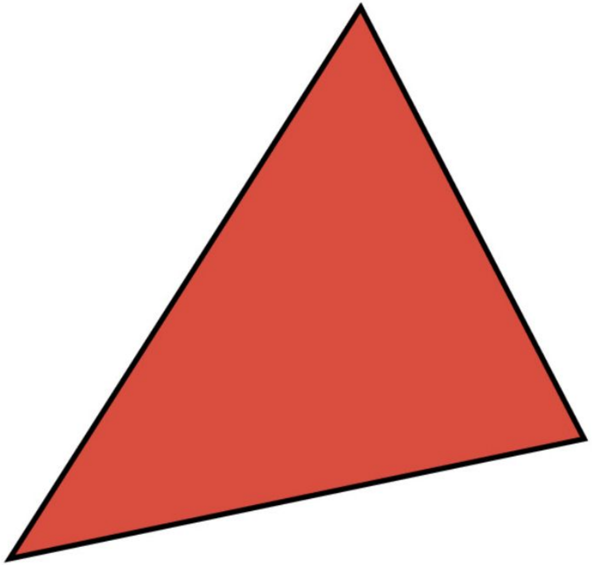


continuous signal

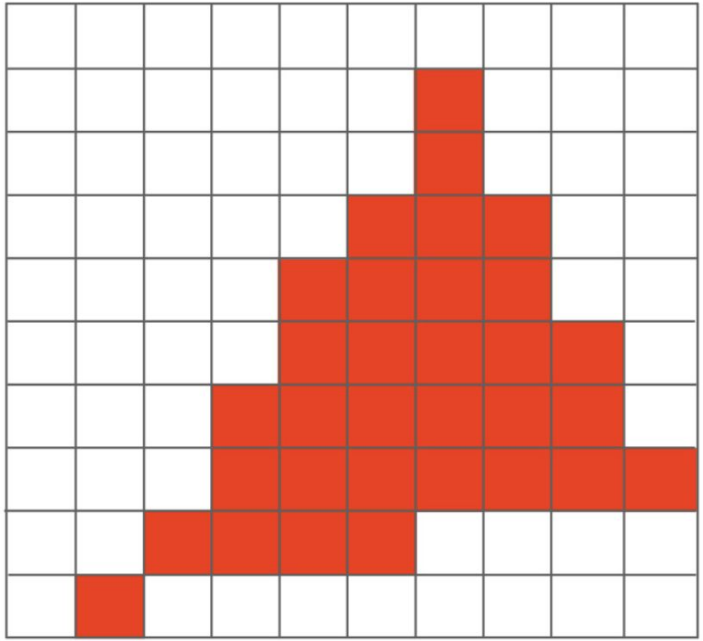


overlay on pixel grid

Sampling a Triangle

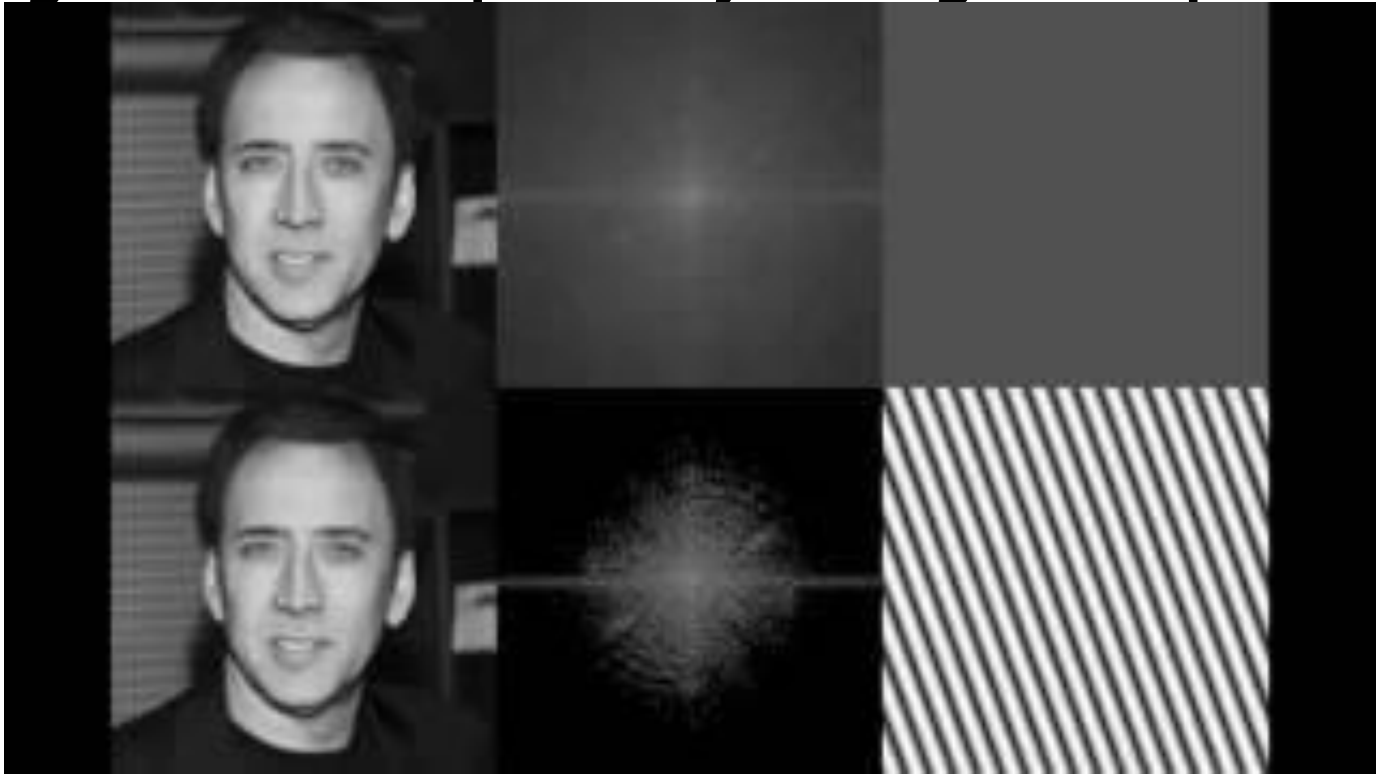


continuous signal



aliased (jaggies)

An image can be composed by adding all frequencies



Nyquist Theorem

Nyquist frequency

The Nyquist frequency is half of the sampling frequency of your device.

$$f_{nyquist} = \frac{1}{2} f_{sampling}$$

Ex: If your camera has a frame rate of 24 (runs at 24 Hz), the Nyquist frequency is 12 Hz.

Nyquist theorem

There is no aliasing from frequencies in the signal that are **less than** the Nyquist frequency.

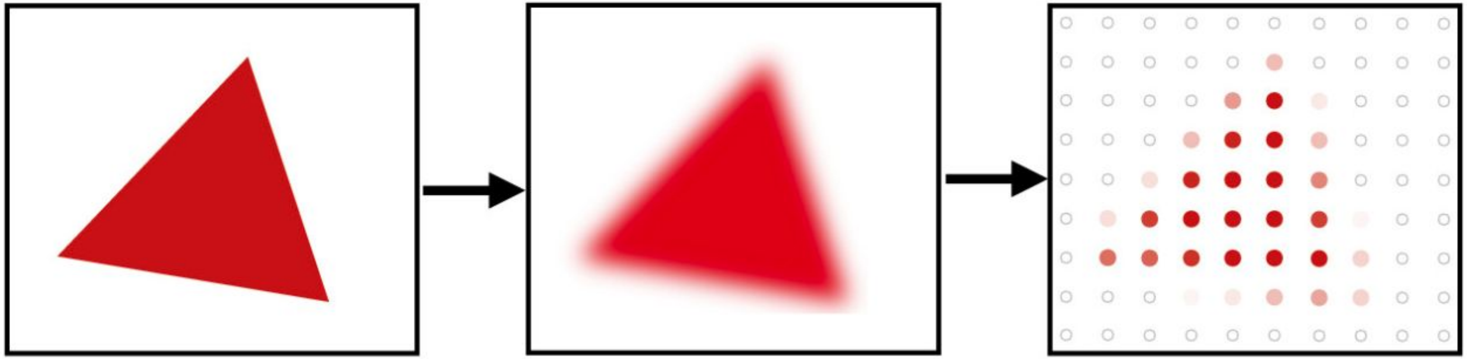
Ex (previous slide): Your camera can capture signals with frequency **less than 12 Hz** without aliasing.

Nyquist theorem

There is no aliasing from frequencies in the signal that are **less than** the Nyquist frequency.

Ex: Your 24 Hz camera can capture signals with frequency **less than 12 Hz** without aliasing.

How do we avoid aliasing?

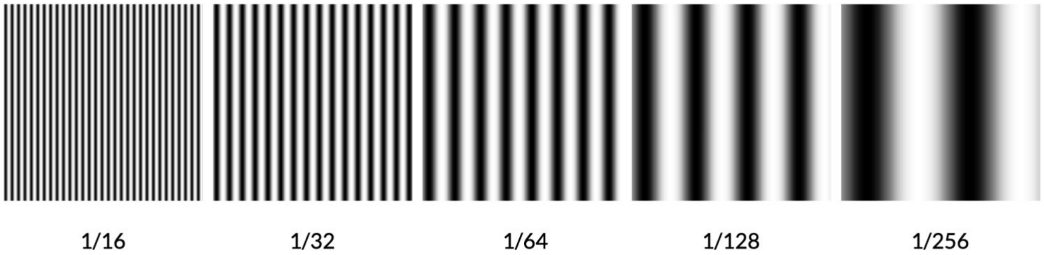


Pre-Filter

(remove frequencies above Nyquist)

Sample

1. We have five examples of spatial frequencies in cycles per pixel (given by the number below the image). What frequency should each image be sampled at to avoid aliasing?



2. We have two wheels that rotate at different speeds with different numbers of spokes that we want to record with our camera.

   - **Wheel A** has 4 spokes and rotates at a rate of 6 rotations per second.
   - **Wheel B** has 6 spokes and rotates at a rate of 5 rotations per second.

   What frame rate (in frames per second) would our camera need to avoid aliasing effects?

1. Ashley's video camera records at 128 frames per second. The helicopter she is filming has eight blades that rotate. At what rotation rate(s) of the helicopter rotor will Ashley's video suffer from aliasing effects? Give your answer in rotations per second.

Nyquist Theorem

Nyquist Frequency

$$f_{nyquist} = \frac{1}{2} f_{sampling}$$

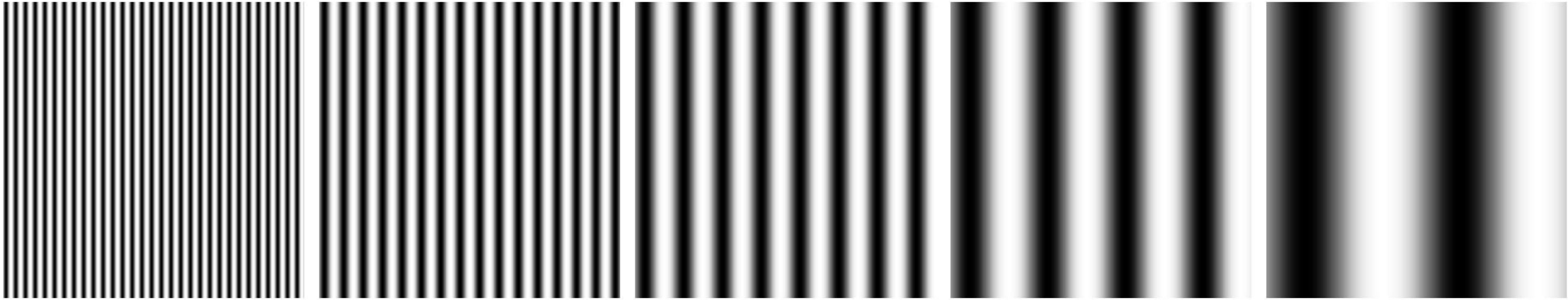
- Given a **sampling rate**, what **input frequencies** can be recovered?
- The highest frequency that can be sampled before losing information (aliasing)
- Ex: If your device can sample at 100Hz, then Nyquist frequency is 50Hz.

Sampling Frequency

$$f_{sampling} > 2 f_{signal}$$

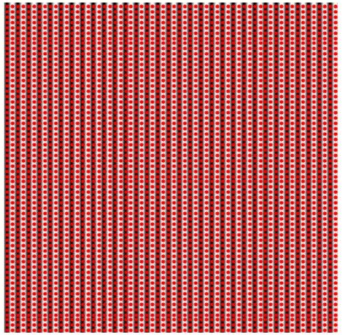
- Given an **input frequency**, what's the minimum **sampling rate** needed to recover original frequency?
- The lowest frequency that you can sample with before aliasing
- Ex: If the signal is 20Hz, then your sampling rate should be above 40Hz.

Theoretical Example of Sampling Spatial Frequencies



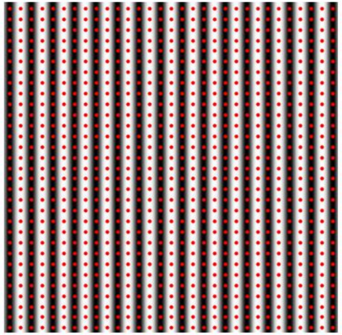
- Each number represents the frequency or the pixels per cycle
- What is the sampling rate needed for each of these?

Nyquist Sampling of Max Spatial Frequencies



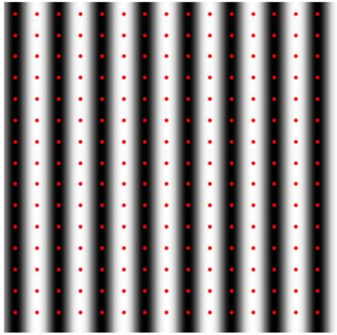
1/16

> 2 * 1/16 = ⅛



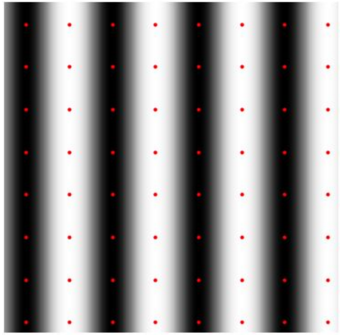
1/32

> 1/16



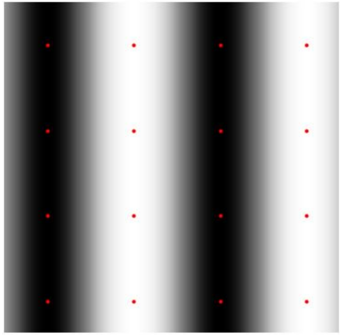
1/64

> 1/32



1/128

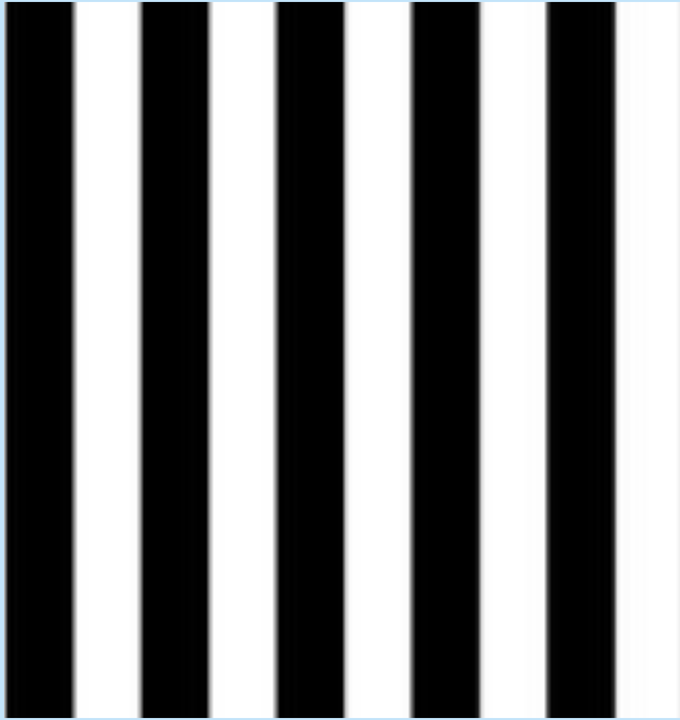
> 1/64



1/256

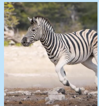
> 1/128

Sampling – Image Frequencies

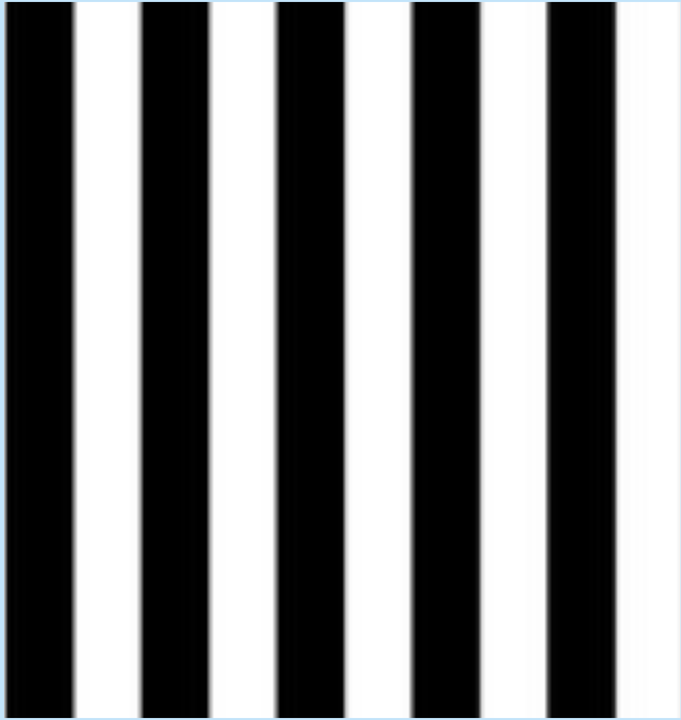


frequency = 1/32;

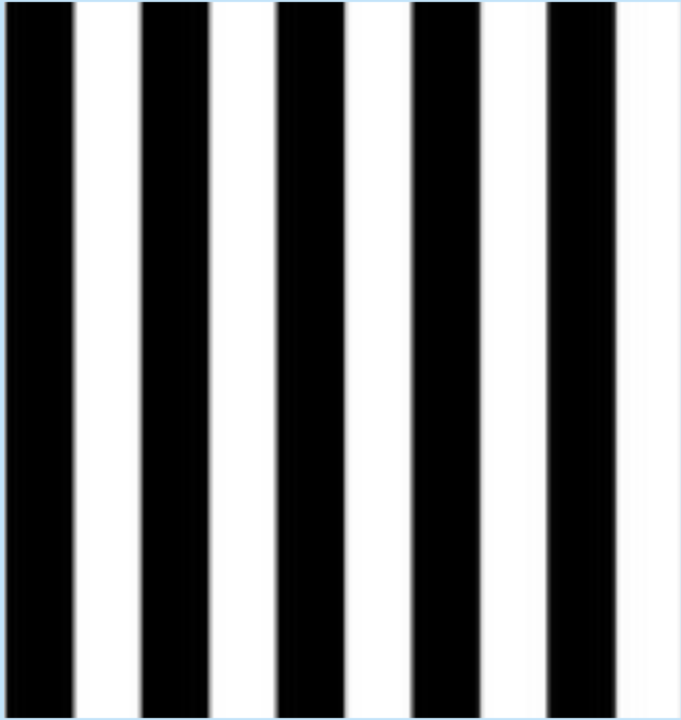
32 pixels per cycle (1 black and 1 white stripe is a cycle)

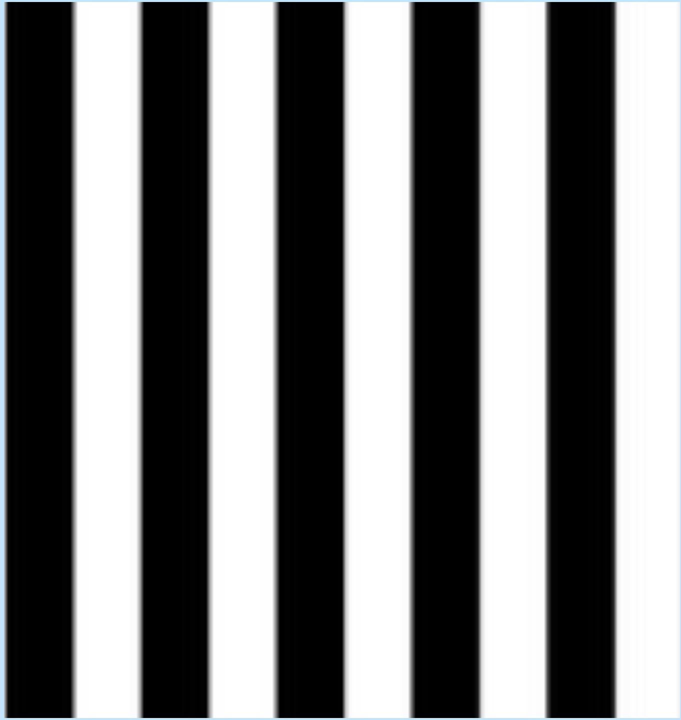


Sampling – Image Frequencies

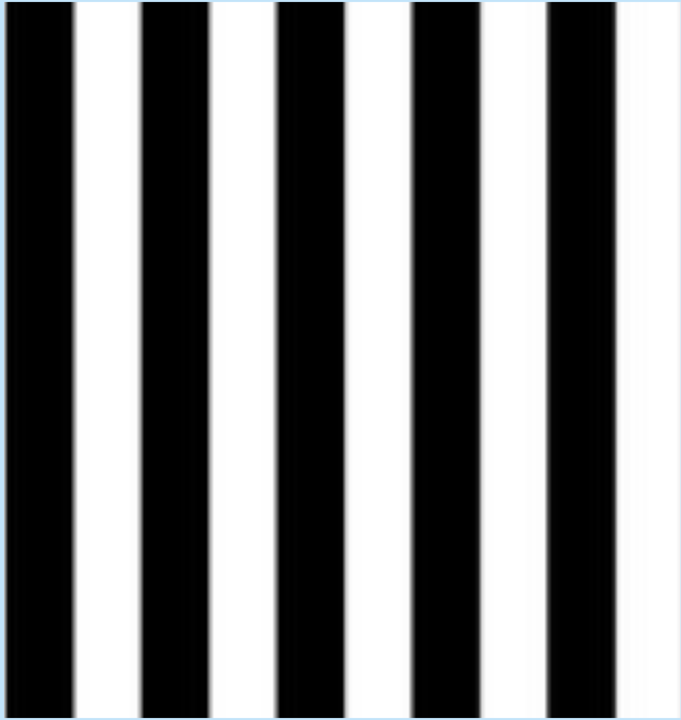


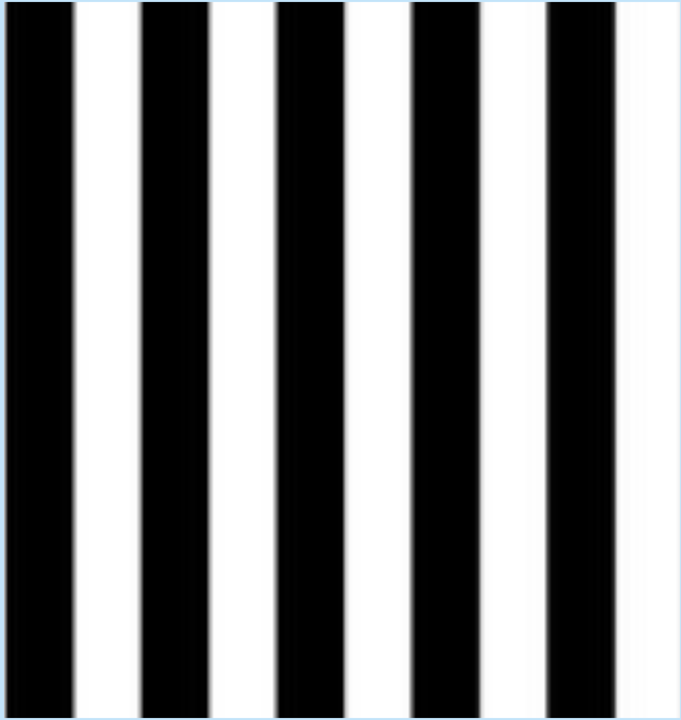
Sampling – Image Frequencies





Sampling – Image Frequencies



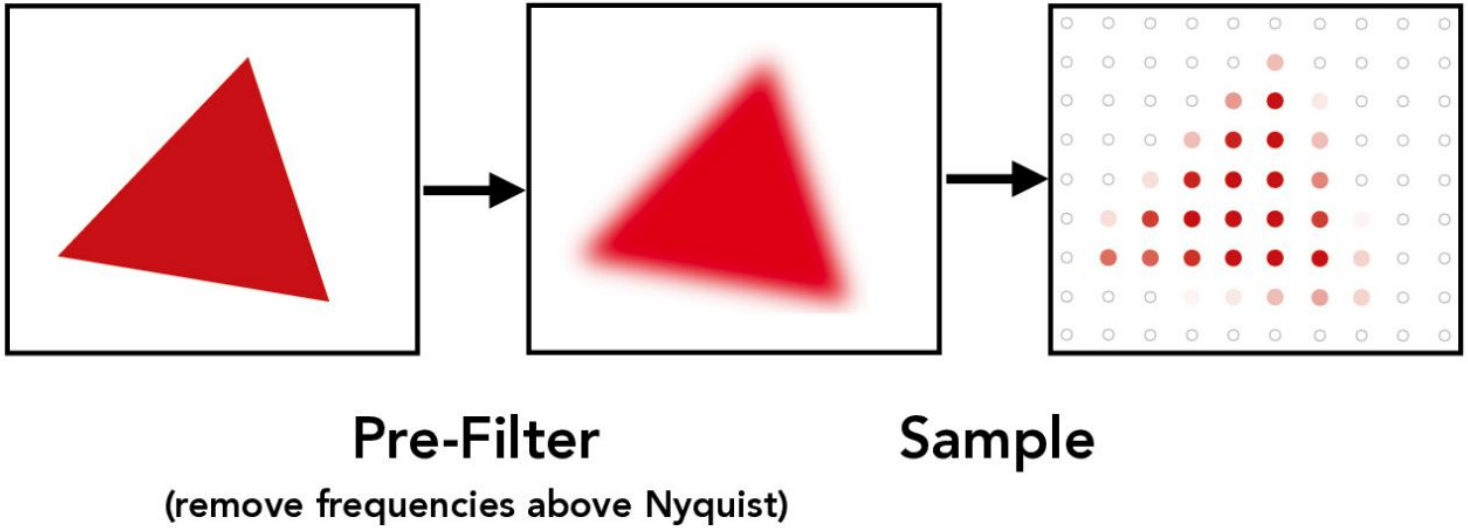


How to reduce aliasing?

$$f_{nyquist} = \frac{1}{2} f_{sampling}$$

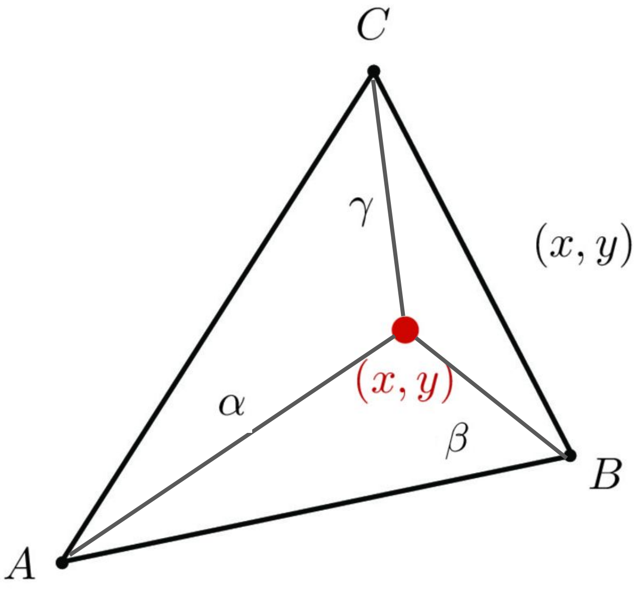
Filter the input

Remove all frequencies above (and equal to) the Nyquist frequency.



Barycentric Coordinates

Barycentric Coordinates



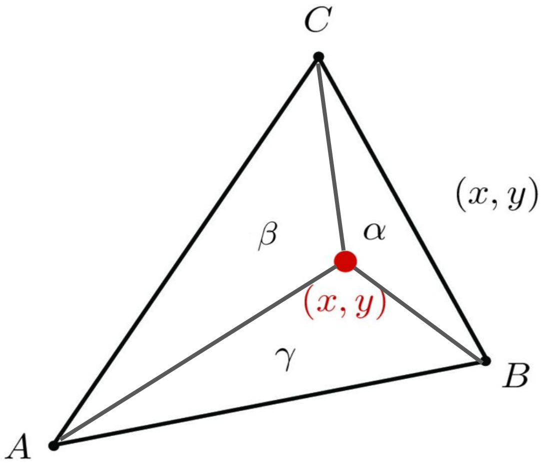
$$(x, y) = \alpha\, A + \beta\, B + \gamma\, C$$
$$\alpha + \beta + \gamma = 1$$

https://www.geogebra.org/m/ZuvmPjmy

- Represents the **distance** from a given point to the triangle’s vertices
- Alpha, beta, and gamma act as weights

Barycentric Coordinates

- (x, y) as weighted sum of values at triangle vertices.
- If *A*, *B*, *C* are colors, *αA + βB + γC* is an in-between color.



$$(x, y) = \alpha\, A + \beta\, B + \gamma\, C$$

$$\alpha + \beta + \gamma = 1$$

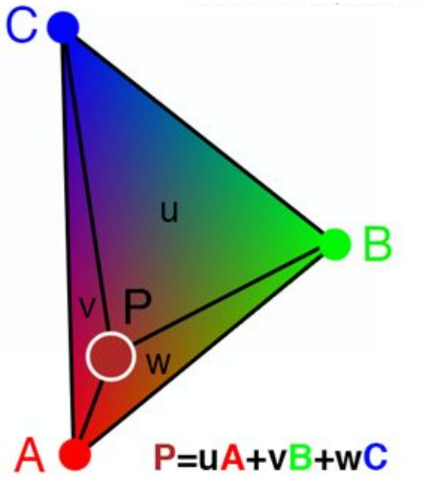
$$\alpha = \frac{-(x - x_B)(y_C - y_B) + (y - y_B)(x_C - x_B)}{-(x_A - x_B)(y_C - y_B) + (y_A - y_B)(x_C - x_B)}$$

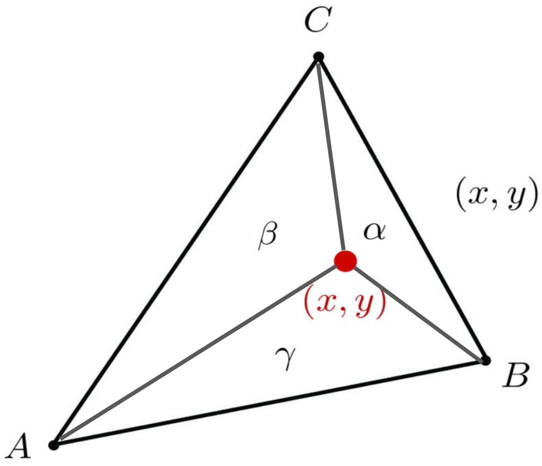
$$\beta = \frac{-(x - x_C)(y_A - y_C) + (y - y_C)(x_A - x_C)}{-(x_B - x_C)(y_A - y_C) + (y_B - y_C)(x_A - x_C)}$$

$$\gamma = 1 - \alpha - \beta$$

Barycentric Coordinates

- (x, y) as weighted sum of values at triangle vertices.
- If *A*, *B*, *C* are colors, *αA + βB + γC* is an in-between color.

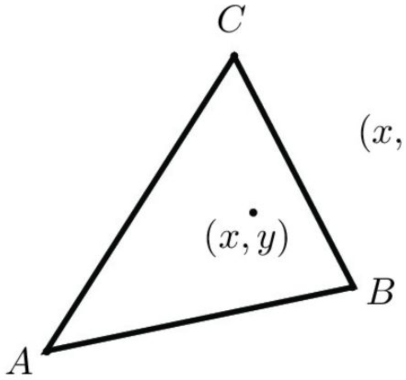




$$(x, y) = \alpha\, A + \beta\, B + \gamma\, C$$
$$\alpha + \beta + \gamma = 1$$

$$\alpha = \frac{-(x - x_B)(y_C - y_B) + (y - y_B)(x_C - x_B)}{-(x_A - x_B)(y_C - y_B) + (y_A - y_B)(x_C - x_B)}$$
$$\beta = \frac{-(x - x_C)(y_A - y_C) + (y - y_C)(x_A - x_C)}{-(x_B - x_C)(y_A - y_C) + (y_B - y_C)(x_A - x_C)}$$
$$\gamma = 1 - \alpha - \beta$$

Barycentric Coordinate Formulas



$$(x, y) = \alpha\, A + \beta\, B + \gamma\, C$$

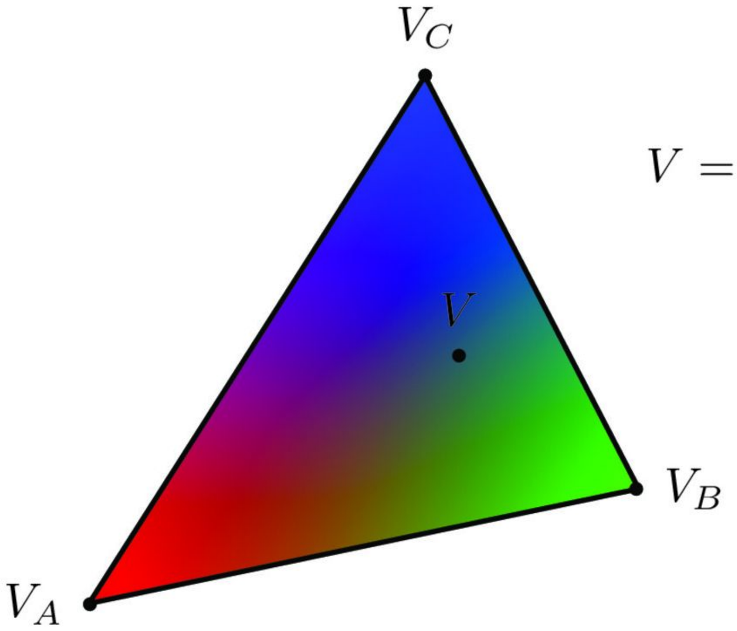
$$\alpha + \beta + \gamma = 1$$

$$\alpha = \frac{-(x - x_B)(y_C - y_B) + (y - y_B)(x_C - x_B)}{-(x_A - x_B)(y_C - y_B) + (y_A - y_B)(x_C - x_B)}$$

$$\beta = \frac{-(x - x_C)(y_A - y_C) + (y - y_C)(x_A - x_C)}{-(x_B - x_C)(y_A - y_C) + (y_B - y_C)(x_A - x_C)}$$

$$\gamma = 1 - \alpha - \beta$$

Barycentric Interpolation



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

V_A, V_B, V_C can be positions, texture coordinates, color, normal vectors, material attributes…

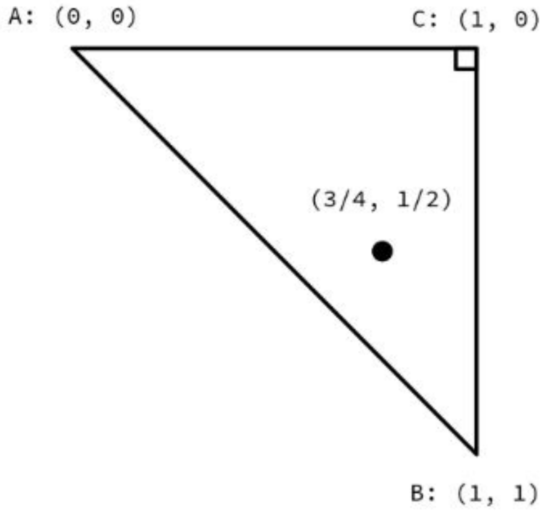
$$(x, y) = \alpha A + \beta B + \gamma C$$

1. What happens if α, β, or γ is less than zero?

2. What is the range of (x, y) if $\alpha + \beta + \gamma = 1$ and $\alpha = 0$?

3. What is the range of (x, y) if $\alpha + \beta + \gamma = 1$ and $\alpha < 0$?

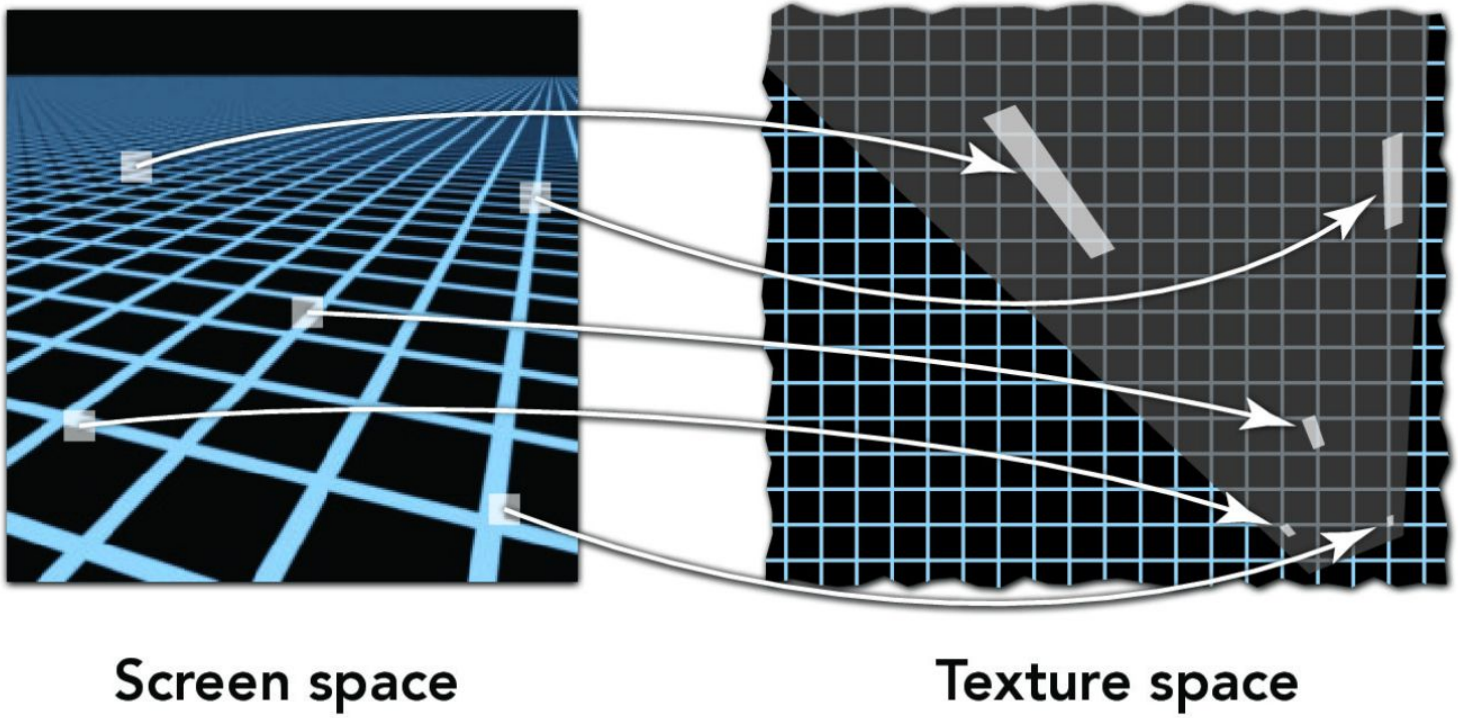
4. What are the barycentric coordinates of the point corresponding to the screen-space coordinates $(x, y) = (\frac{3}{4}, \frac{1}{2})$ in the following diagram? (Hint: Use the definition of barycentric coordinates, not the closed-form solution.)



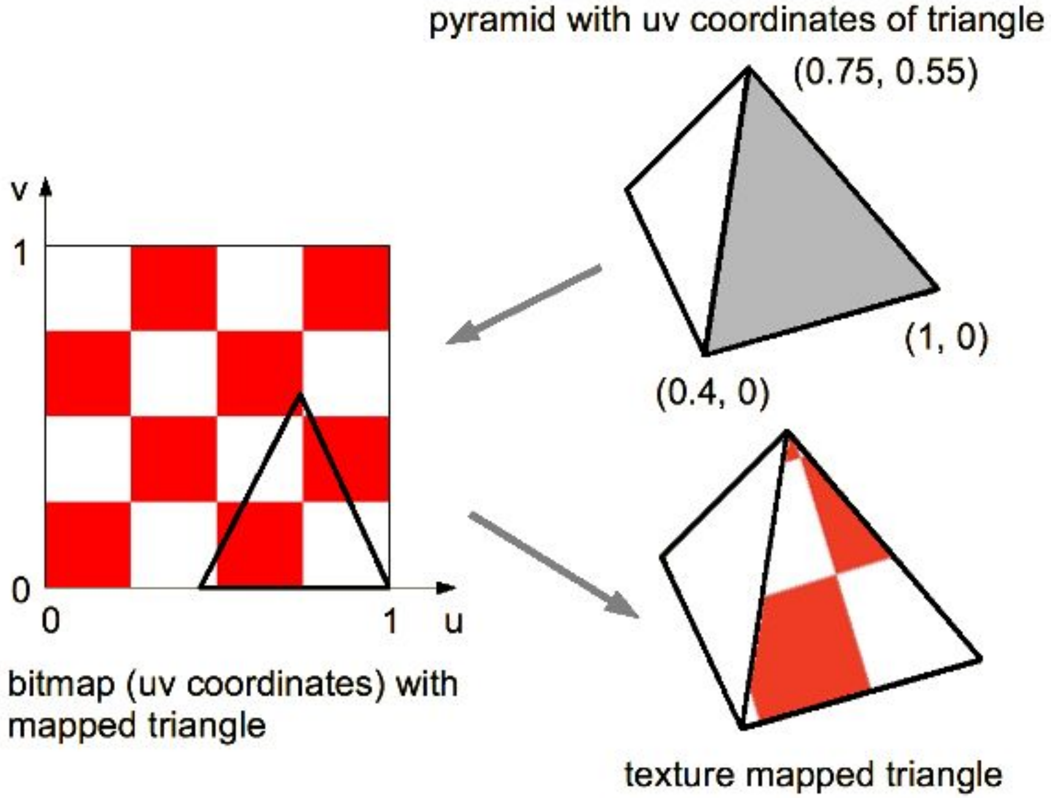
5. If the RGB values of A, B, C are `(1, 0, 0)`, `(0, 1, 0)`, `(0, 0, 1)`, respectively, then what is the interpolated color at the selected point above?

Texture Mapping

Texture Coordinates



Texture Coordinates

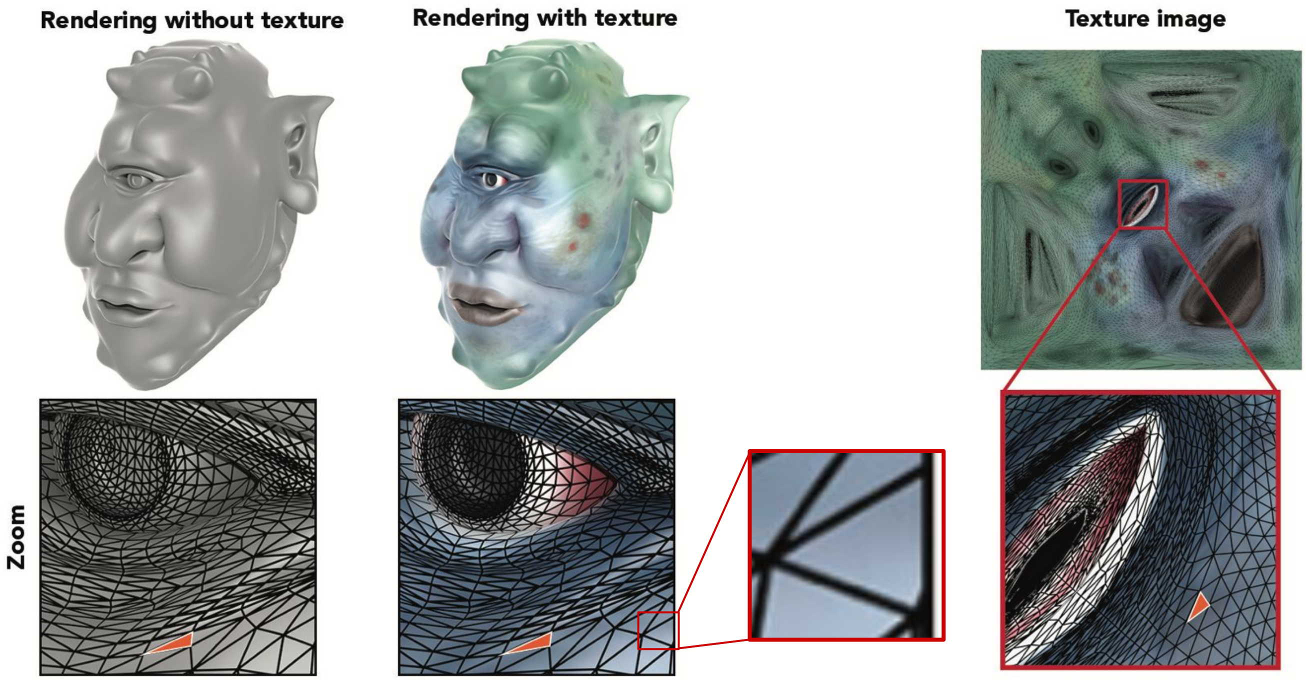


pyramid with uv coordinates of triangle

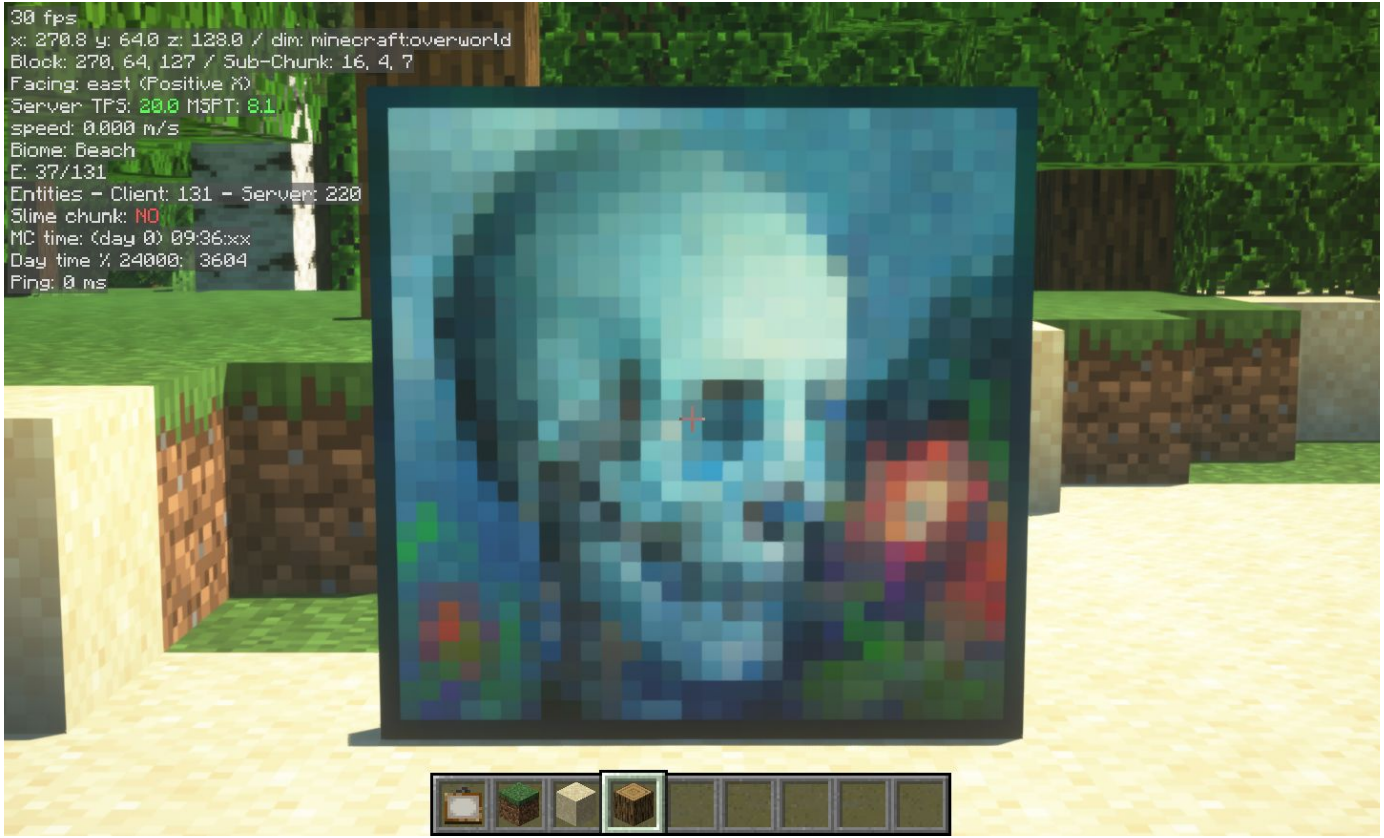
bitmap (uv coordinates) with mapped triangle

texture mapped triangle

Texture Coordinates

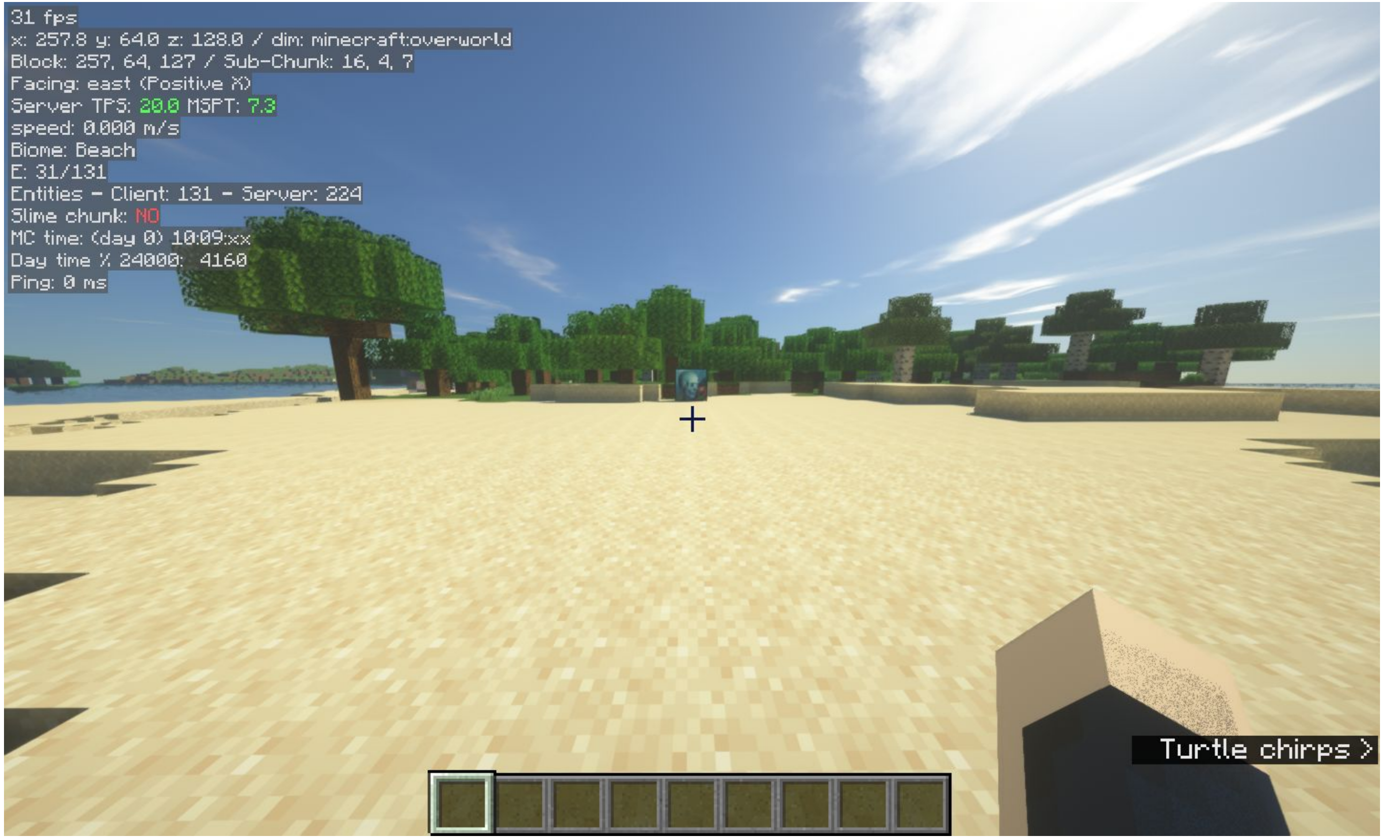


Dealing with aliasing – Minecraft



```
30 fps
x: 270.8 y: 64.0 z: 128.0 / dim: minecraft:overworld
Block: 270, 64, 127 / Sub-Chunk: 16, 4, 7
Facing: east (Positive X)
Server TPS: 20.0 MSPT: 8.1
speed: 0.000 m/s
Biome: Beach
E: 37/131
Entities - Client: 131 - Server: 220
Slime chunk: NO
MC time: (day 0) 09:36:xx
Day time % 24000: 3604
Ping: 0 ms
```

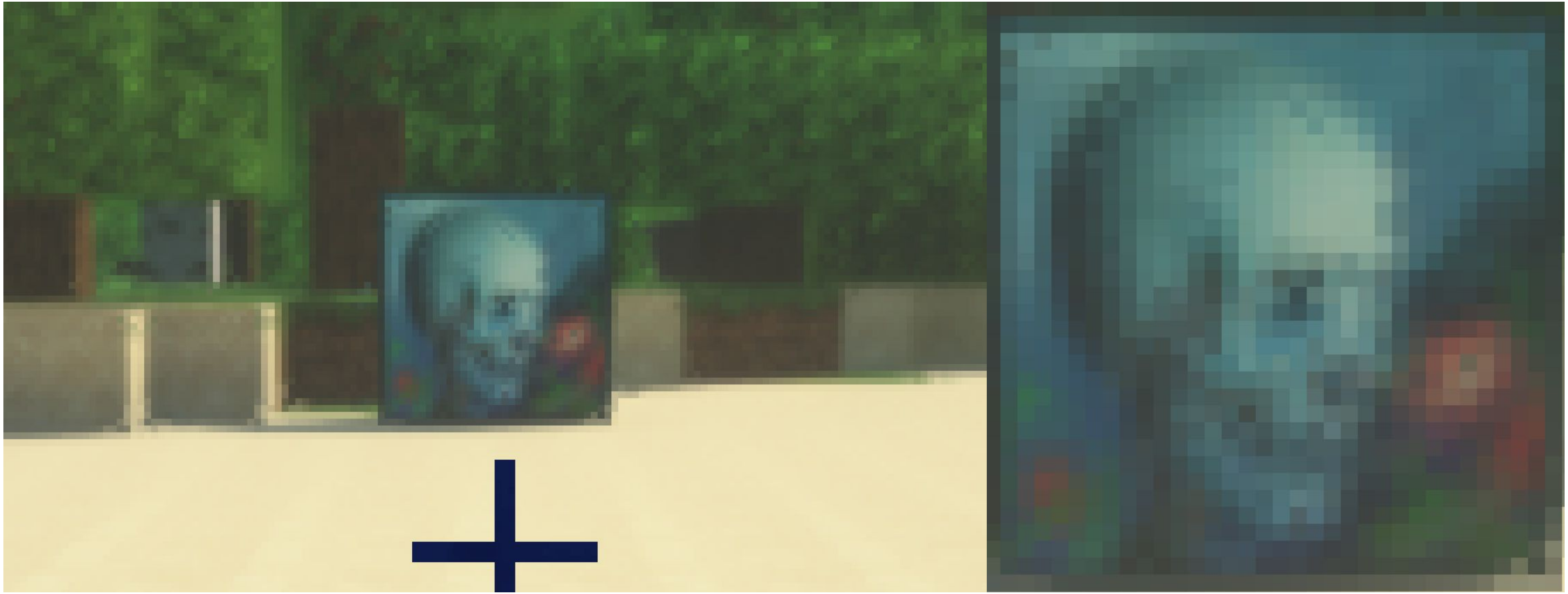
Dealing with aliasing – Minecraft



```
31 fps
x: 257.8 y: 64.0 z: 128.0 / dim: minecraft:overworld
Block: 257, 64, 127 / Sub-Chunk: 16, 4, 7
Facing: east (Positive X)
Server TPS: 20.0 MSPT: 7.3
speed: 0.000 m/s
Biome: Beach
E: 31/131
Entities - Client: 131 - Server: 224
Slime chunk: NO
MC time: (day 0) 10:09:xx
Day time % 24000:  4160
Ping: 0 ms
```

```
Turtle chirps >
```

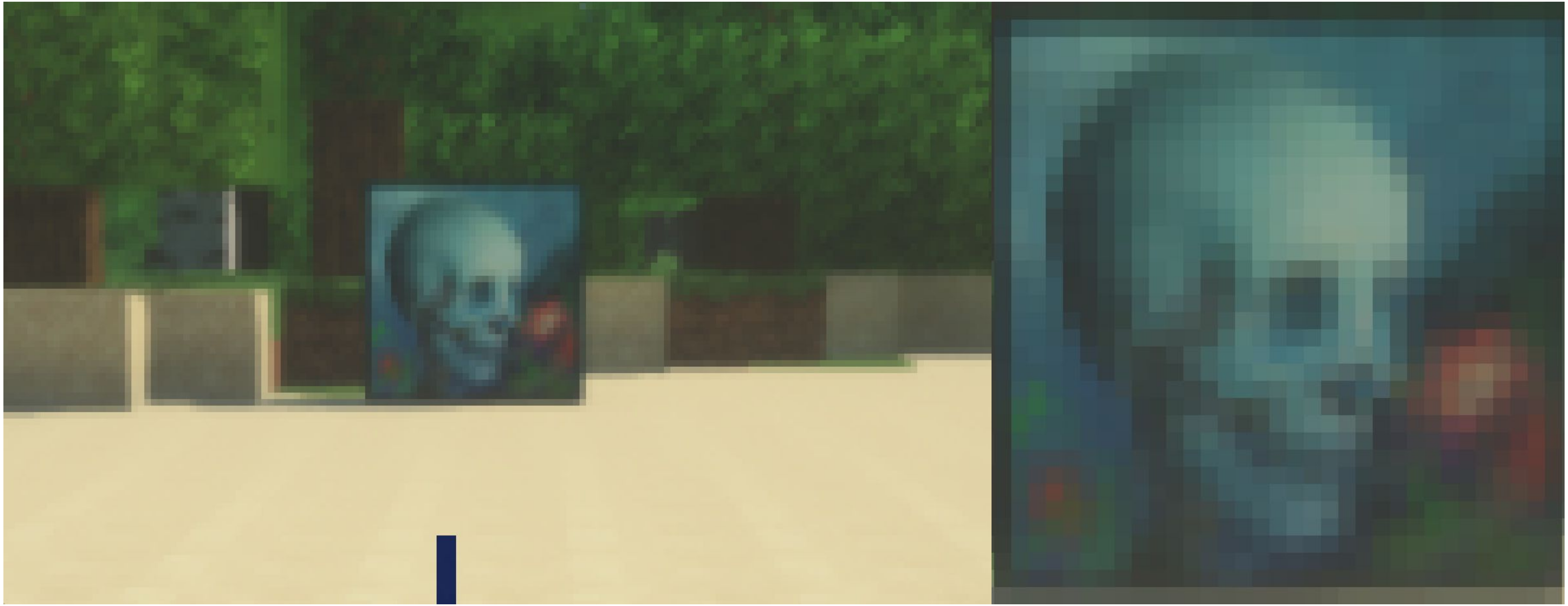
Dealing with aliasing – Minecraft



Antialiasing disabled (not using mipmaps):

visible artifacts

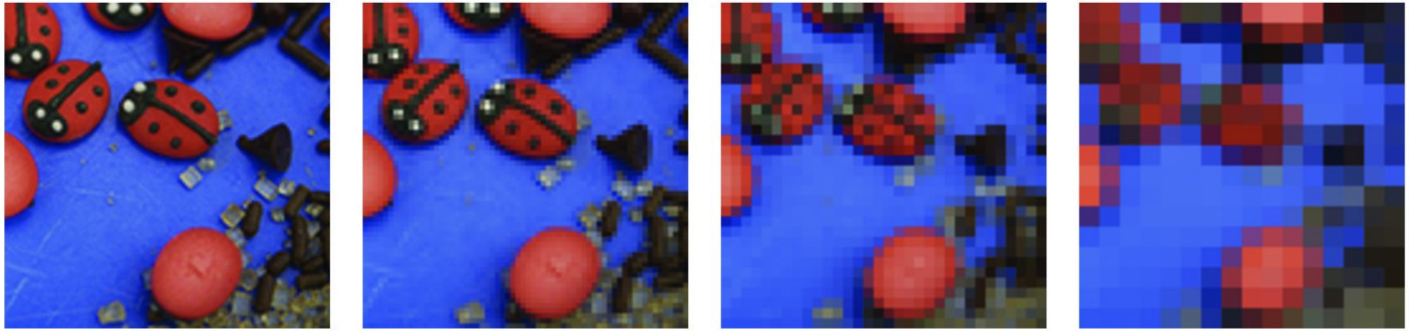
Dealing with aliasing – Minecraft



Antialiasing via mipmaps enabled

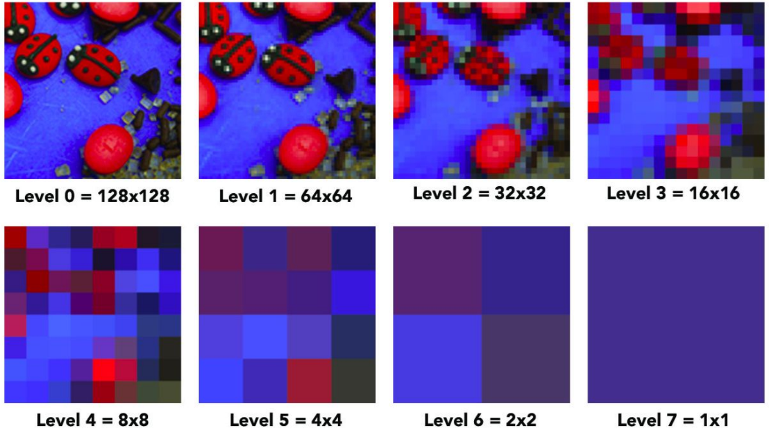
Antialiasing Textures

- Super-sampling, then down-sampling, is equivalent to low-pass filtering.
- Sample lower-resolution textures to avoid aliasing.
- Avoids expensive computations.



Antialiasing Textures

1. Pre-compute lower resolution versions of texture
2. Store textures in mipmap, using texture cache (if on GPUs).
3. Adaptively choose mipmap level, *D*, according to scene.



Level 0 = 128x128

Level 1 = 64x64

Level 2 = 32x32

Level 3 = 16x16

Level 4 = 8x8

Level 5 = 4x4

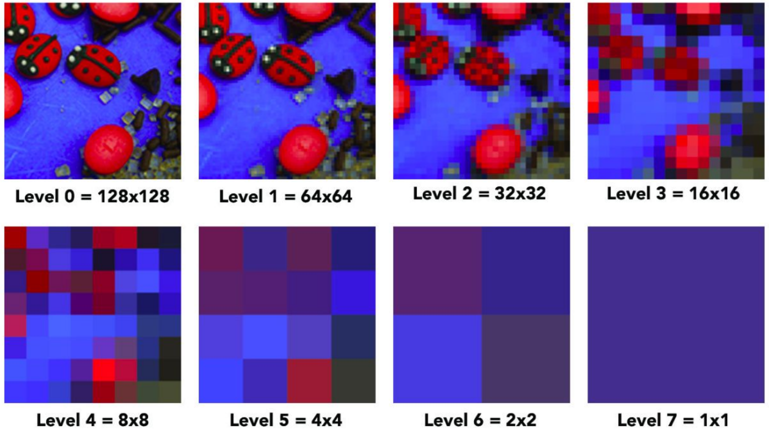
Level 6 = 2x2

Level 7 = 1x1

Antialiasing Textures

1. Pre-compute lower resolution versions of texture
2. Store textures in mipmap, using texture cache (if on GPUs).
3. Adaptively choose mipmap level according to scene.

Halve the dimensions each time →



Level 0 = 128x128

Level 1 = 64x64

Level 2 = 32x32

Level 3 = 16x16

Level 4 = 8x8

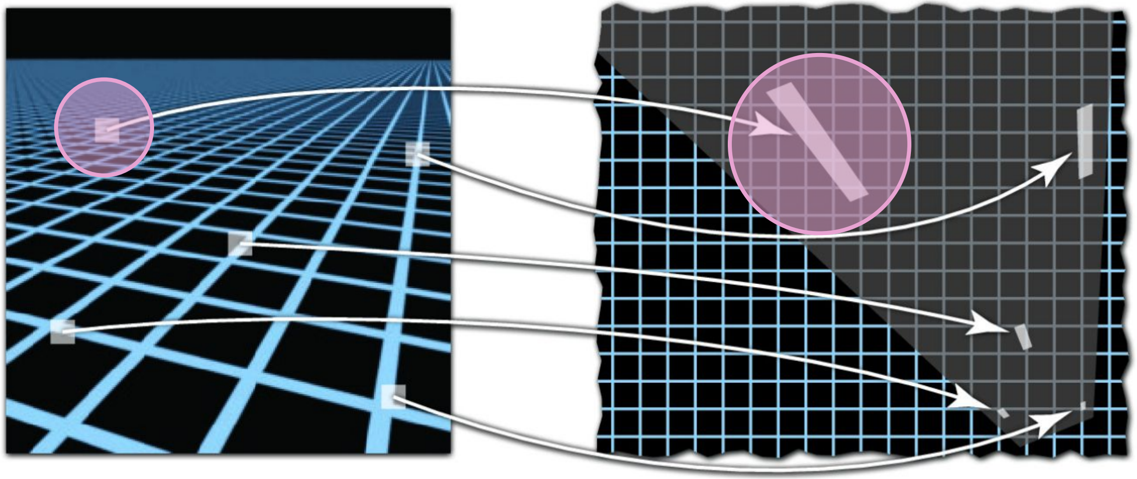
Level 5 = 4x4

Level 6 = 2x2

Level 7 = 1x1

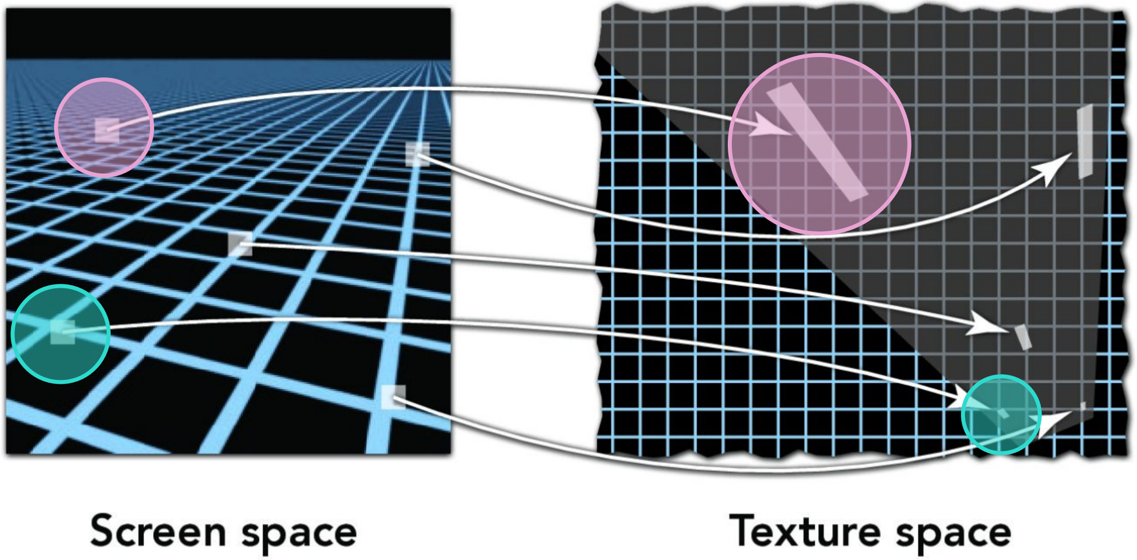
Choosing Mipmap Levels

- Big jump in texture space → use a blurred texture → high *D*.
- Small jump in texture space → use a high resolution texture → low *D*.



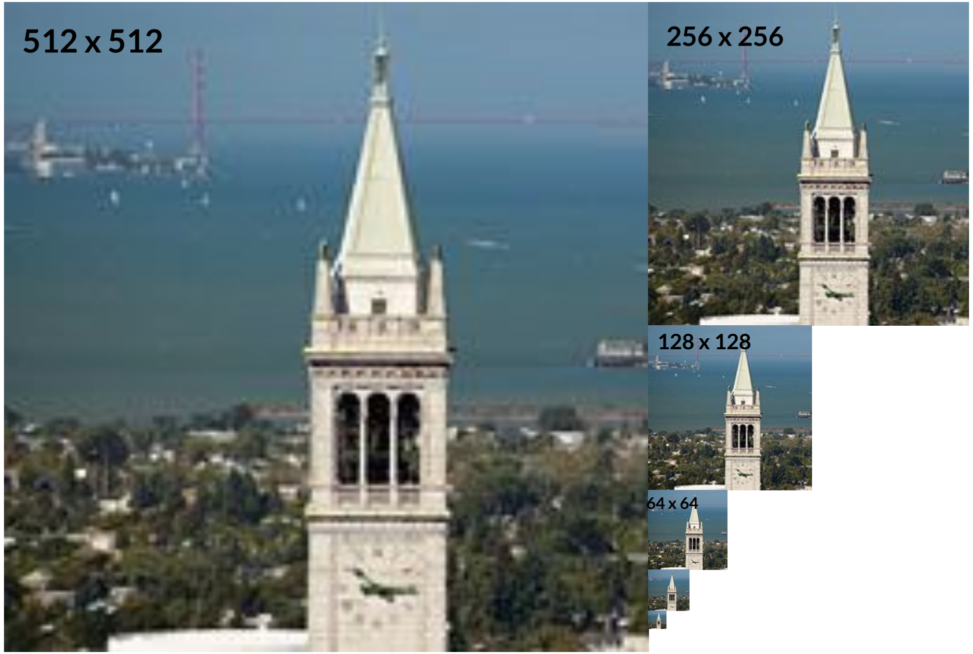
Choosing Mipmap Levels

- Big jump in texture space → use a <u>blurred</u> texture → high *D*.
- Small jump in texture space → use a <u>high resolution</u> texture → low *D*.

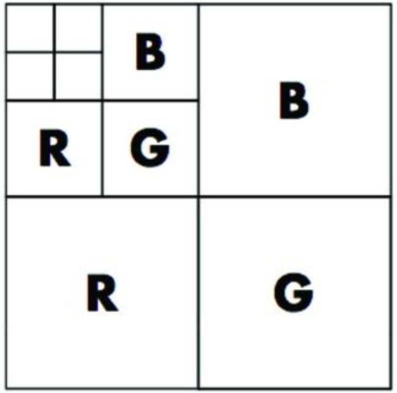


1. The resolution of a mipmap at level 3 is what fraction of the size of the original texture?

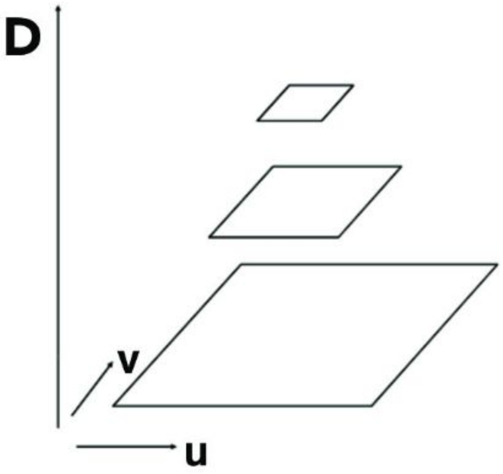
Mipmap Example



Mipmaps



Williams' original proposed mipmap layout



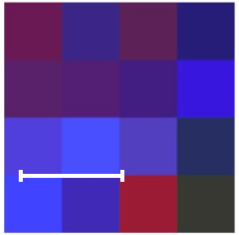
"Mip hierarchy" level = D

2. A mipmap takes up what fraction of the original′s size in memory?

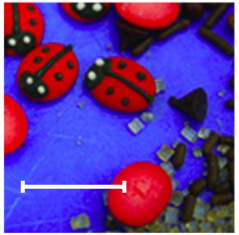
How do we determine which Mipmap level to use?

Idea: Look at a pixel's neighbors:

- If there's a big jump in texture space, we should use a blurred texture
- If there's a small jump in texture space, we should use a high res texture



Level 5

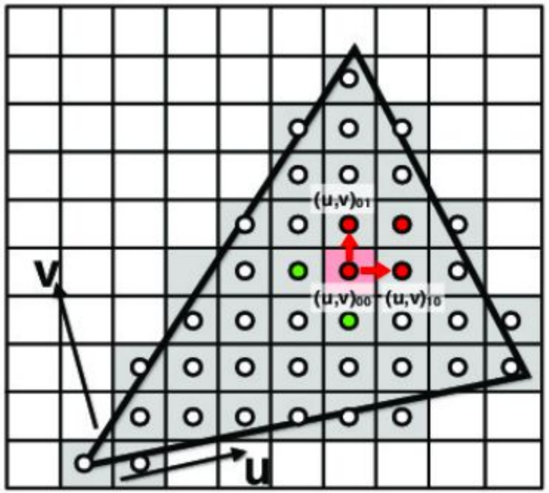


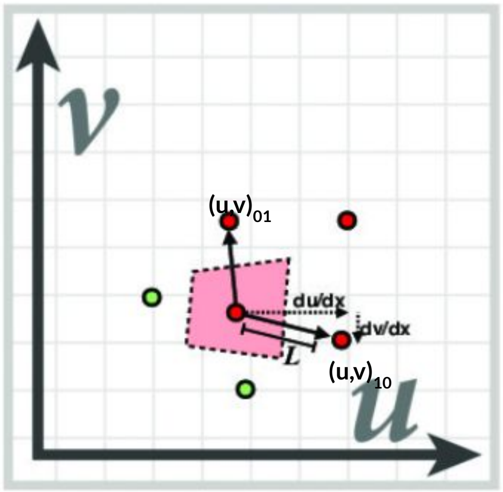
Level 0

$$D = \log_2 L$$

D = level
L = factor that you downsample by

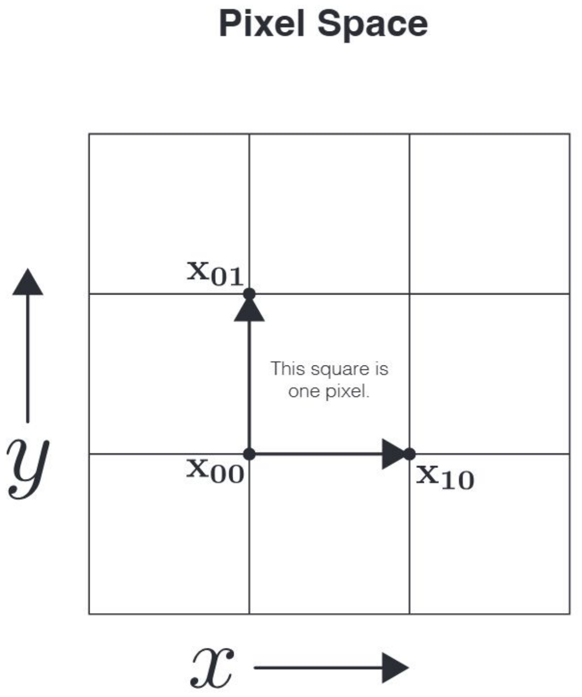
Mathematical Deep Dive - Mipmap Equation

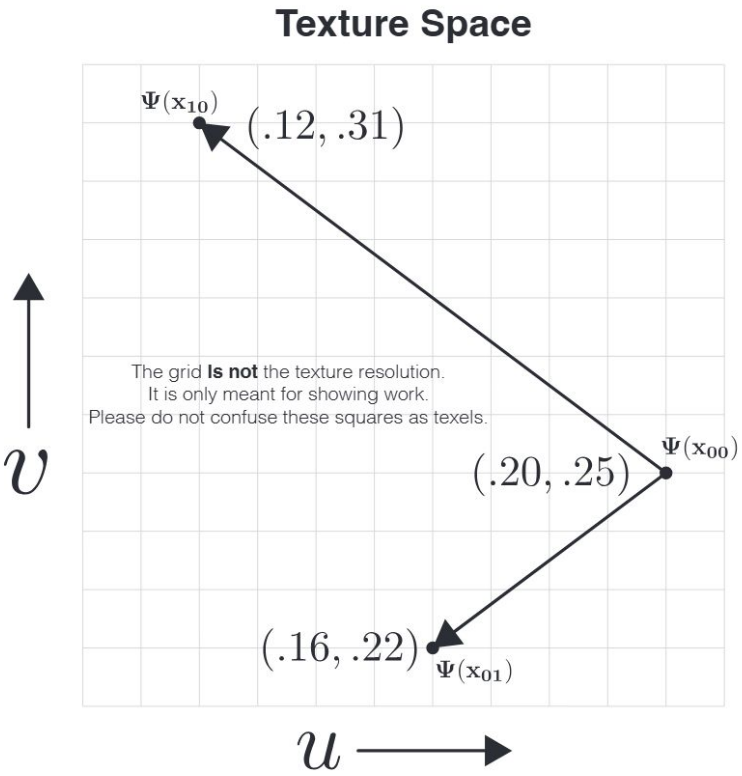




$$L = \max\left(\sqrt{\left(\frac{du}{dx}\right)^2 + \left(\frac{dv}{dx}\right)^2}, \sqrt{\left(\frac{du}{dy}\right)^2 + \left(\frac{dv}{dy}\right)^2}\right)$$

3. We are given pixel point $\mathbf{x_{00}} = (x, y)$, as well as $\mathbf{x_{01}}$ and $\mathbf{x_{10}}$, each 1-pixel away from $\mathbf{x_{00}}$. These three points are mapped to their corresponding locations in texture space by the mapping $\mathbf{\Psi}(\mathbf{x})$. *Note that the mapping uses barycentric coordinates!* The texel space is in the range $[0, 1]$. At what mipmap level, L, should we sample to retrieve the texture for point $\mathbf{x_{00}}$?





Texture Space
$\mathbf{\Psi}(\mathbf{x_{10}})$
$(.12, .31)$
$\mathbf{\Psi}(\mathbf{x_{00}})$
$(.20, .25)$
$(.16, .22)$
$\mathbf{\Psi}(\mathbf{x_{01}})$
v
u

The grid **Is not** the texture resolution.
It is only meant for showing work.
Please do not confuse these squares as texels.

4. For mipmap levels 0 through 5, the texture value at $\mathbf{\Psi}(x_{00})$ is given by:

$$T_0 = 0.38, \quad T_1 = 0.42, \quad T_2 = 0.36, \quad T_3 = 0.40, \quad T_4 = 0.39, \quad T_5 = 0.37.$$

Given our answer to the previous question, what texture value should we use?

5. How can we combine values from two neighboring mipmap levels to get an even smoother result?

3. How do we determine which level of a mipmap to use for a particular screen-space pixel (x, y)?

Mipmap Equation

$$D = \log_2 L$$

Bilinear Filtering

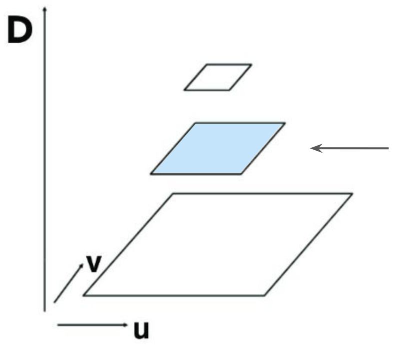
Only use values from

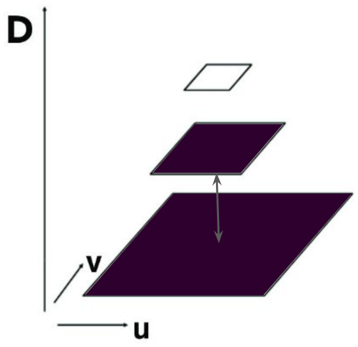
one mipmap level

Trilinear Filtering

Linearly interpolate between

two mipmap levels





4. How can we combine values from two neighboring mipmap levels and why would we do that?

Mathematical Deep Dive - Mipmap Equation

Why max?

Consider this case:

$$\frac{du}{dx} = 1, \frac{dv}{dx} = 1, \frac{du}{dy} = 8, \frac{dv}{dy} = 10$$

Using max(): $L = \max\left(\sqrt{2}, \sqrt{164}\right) = \sqrt{164} = \ 12.8 \rightarrow \ D = \log_2 12.8 = 3.678$

Using min(): $L = \min\left(\sqrt{2}, \sqrt{164}\right) = \sqrt{2} = \ 1.41 \rightarrow \ D = \log_2 1.41 = 0.4957$

Using avg(): $L = avg(\sqrt{2}, \sqrt{164}) = \sqrt{2} = \ 7.105 \rightarrow \ D = \log_2 7.105 = 2.829$

Using min would mean we lose 8-10 texture pixels worth of info per pixel in the y direction!

Intuition: We always want to use the "worst-case" mipmap to avoid aliasing

- Would rather we lose detail in one direction/have a blurry image than try to squeeze too much detail in the other direction

Mathematical Deep Dive - Mipmap Equation

Why $\log_2$?

In Practice, we use one of two techniques:

- Bilinear Filtering - we only use values from one mipmap level
- Trilinear Filtering - we interpolate between two mipmap level

Bilinear: If we only want one level we use: $D = \lfloor \log_2 L \rfloor$

Consider $\frac{du}{dx}, \frac{dv}{dx}, \frac{du}{dy}, \frac{dv}{dy}$ are all 1: $L = \sqrt{2}$ → D = 0

$\frac{du}{dx}, \frac{dv}{dx}, \frac{du}{dy}, \frac{dv}{dy}$ are all 2: $L = \sqrt{8}$ → D = 1

Exactly what we want!

$\frac{du}{dx}, \frac{dv}{dx}, \frac{du}{dy}, \frac{dv}{dy}$ are all 4: $L = \sqrt{32}$ → D = 2

Trilinear: $\log_2$ keeps a consistent scaling as the size increases

Let’s Take Attendance.

- Be sure to select Discussion 3 and input your TA’s secret word 😉
- Any feedback? Let us know!

Additional References

- The Rasterization Stage
- Shoelace Theorem
- https://jtsorlinis.github.io/rendering-tutorial/