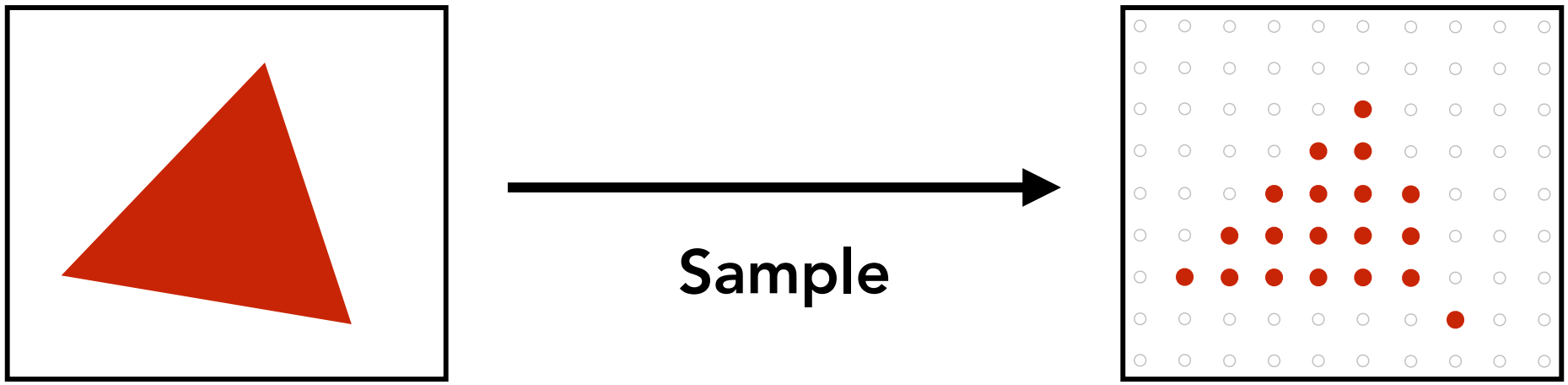
Lecture 3:

Understanding Aliasing and Antialiasing

Computer Graphics 2025

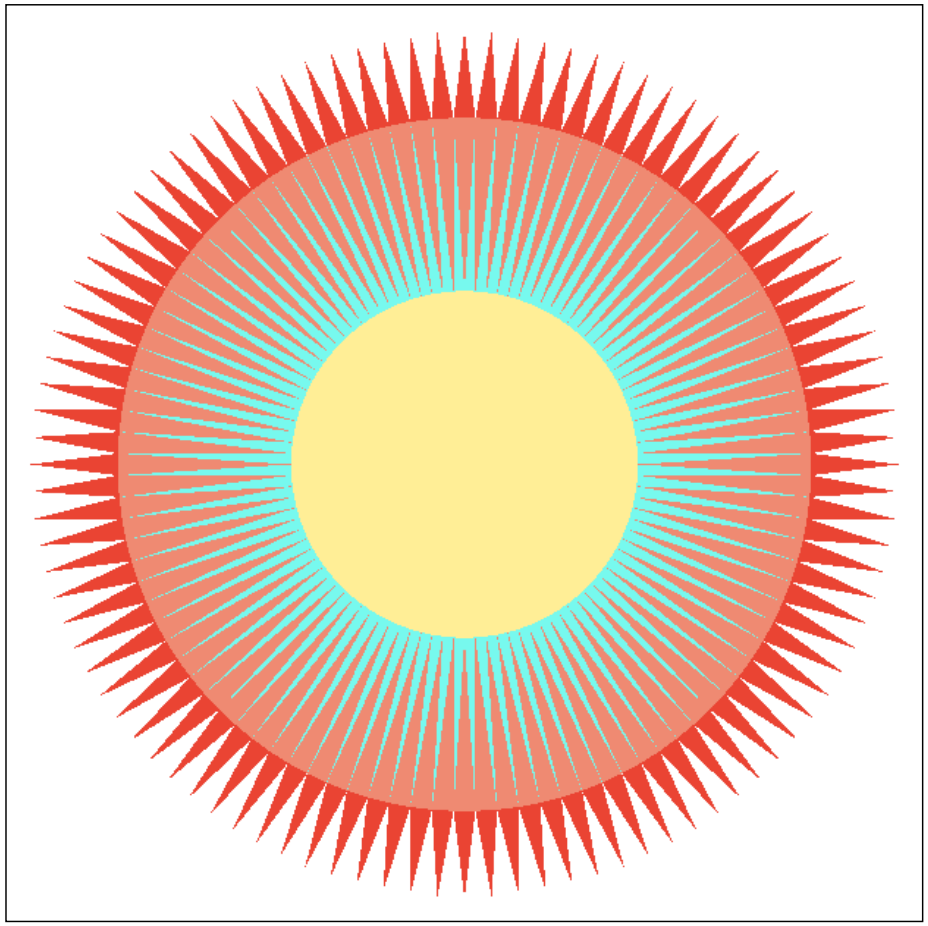
Fuzhou University - Computer Science

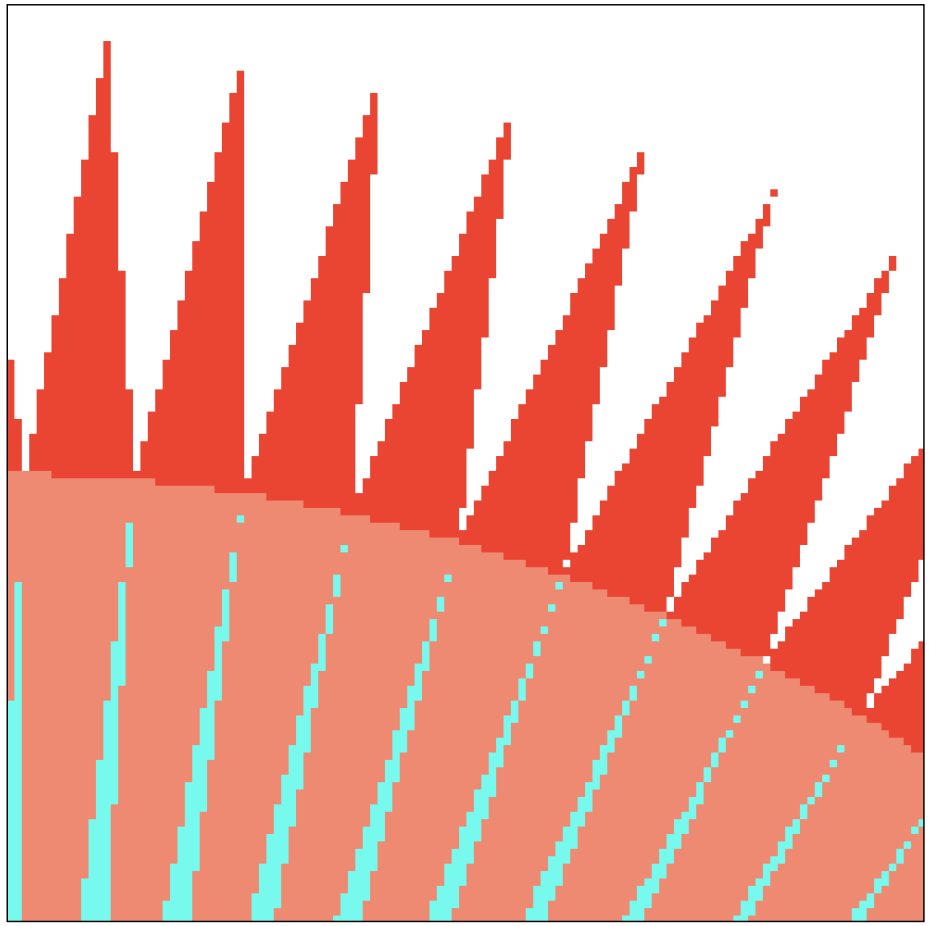
Recall Point Sampling



Note jaggies in rasterized triangle
where pixel values are pure red or white

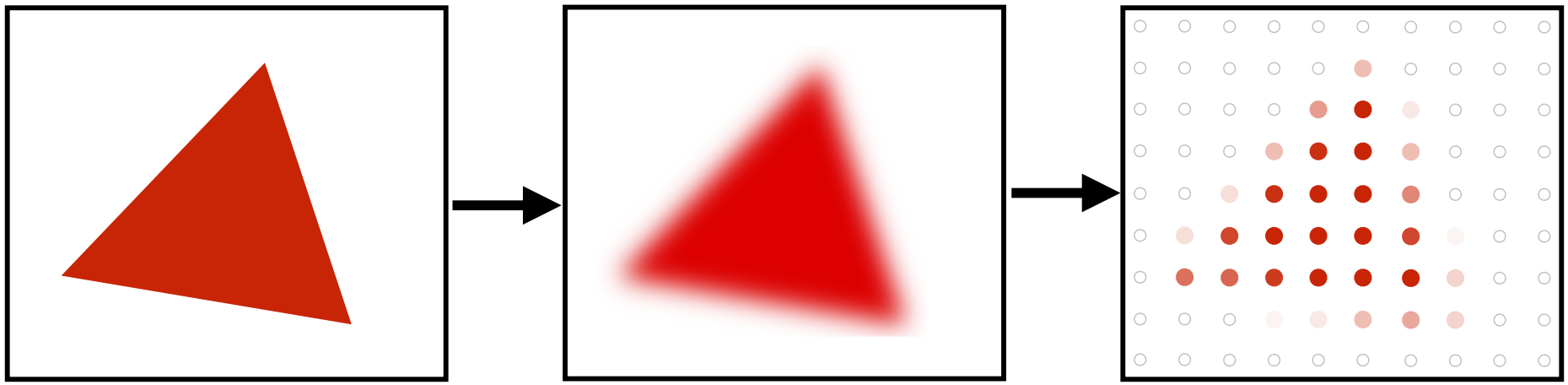
Recall Aliasing





（通俗地讲，走样/混叠的原因是信号的变化速率大于采样频率）

Recall Antialiasing Sampling



Pre-Filter

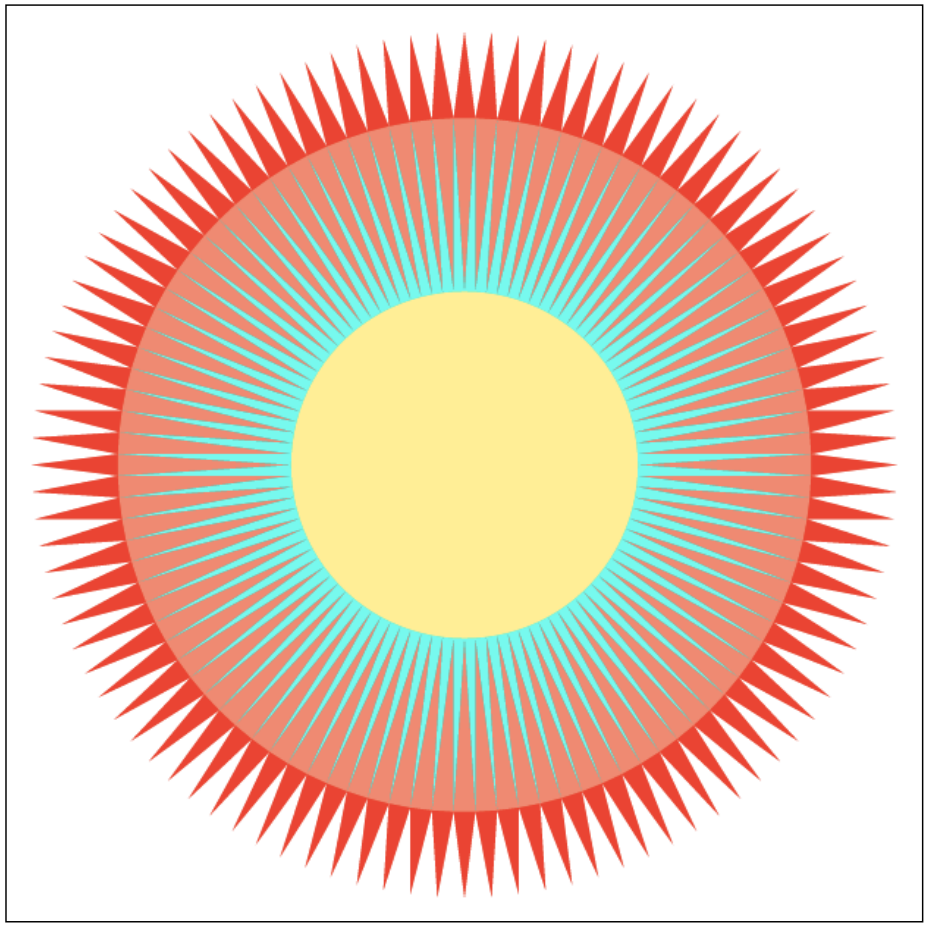
(remove high frequencies)

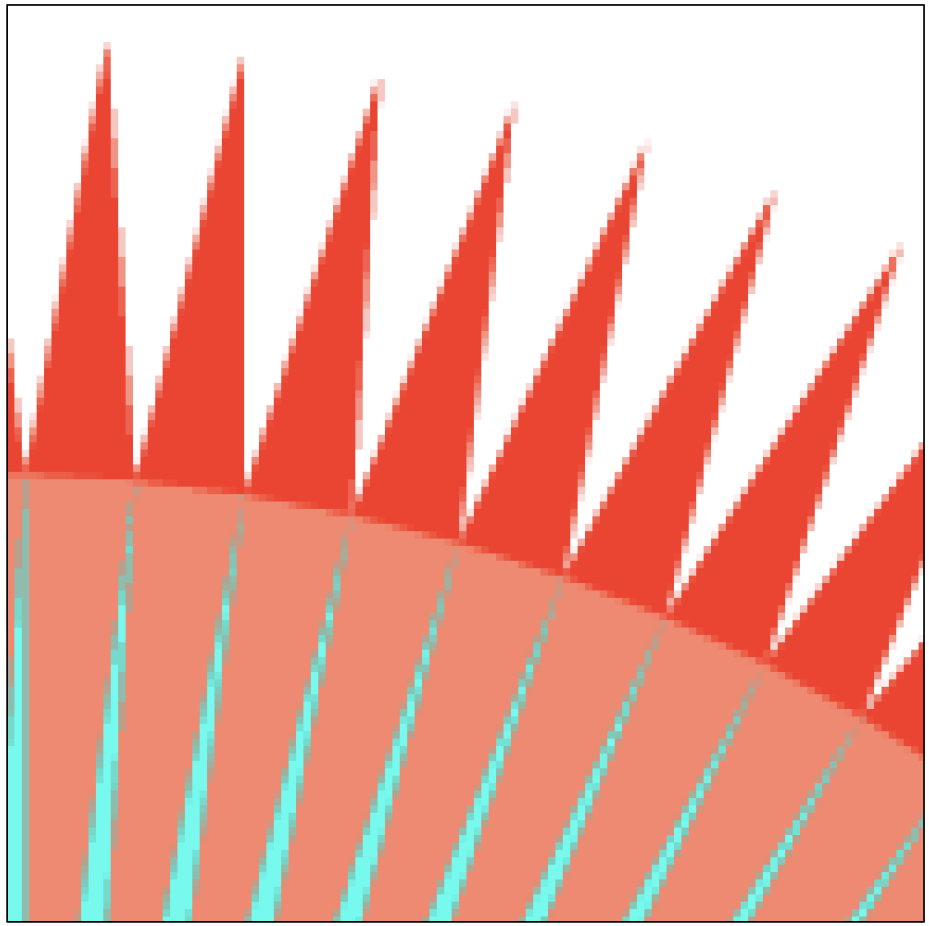
Sample

Note antialiased edges in rasterized triangle where pixel values take intermediate values

(反走样/混叠就是让信号的变化速率与采样频率尽量接近)

Recall Antialiasing





This Lecture

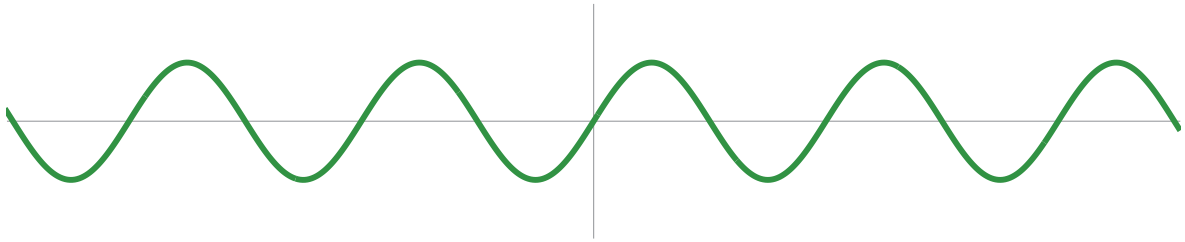
Let’s dig into the fundamental reasons why this works

And look at how to implement antialiased rasterization

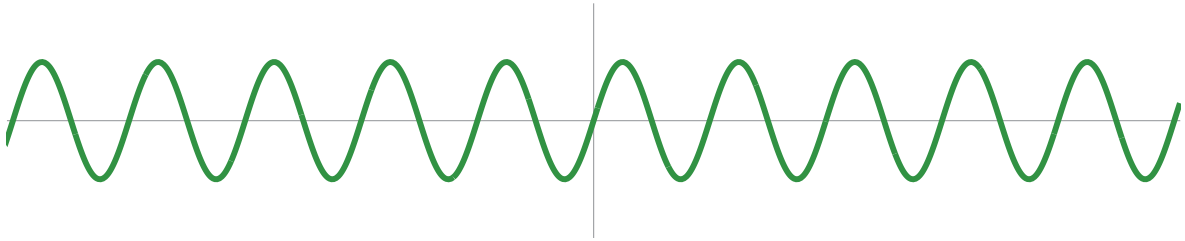
Signal Processing: Frequency Space

Example: sound can be expressed as a superposition of frequencies

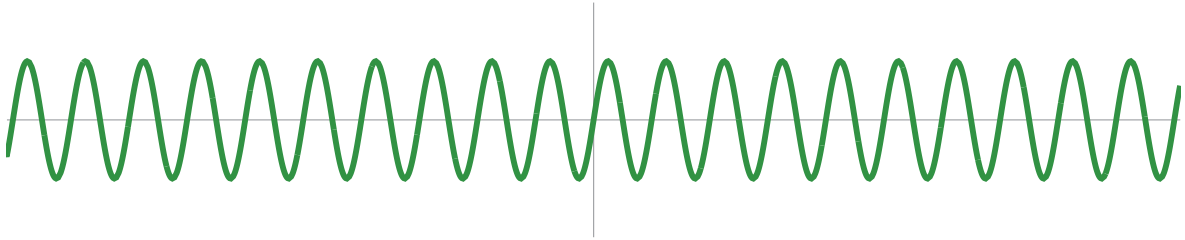
$f_1(x) = sin(\pi x)$



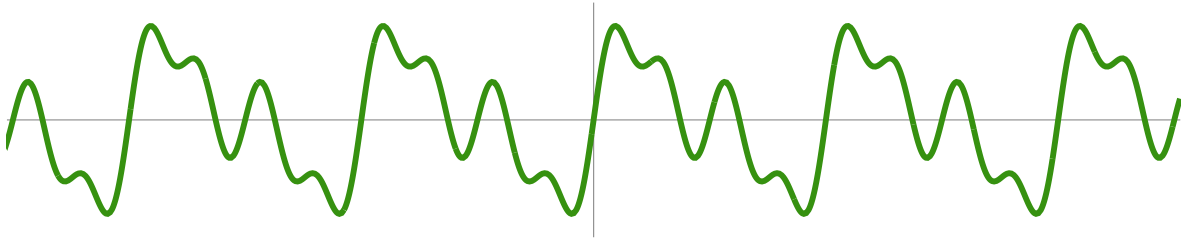
$f_2(x) = sin(2\pi x)$



$f_4(x) = sin(4\pi x)$



$f(x) = f_1(x) + 0.75\, f_2(x) + 0.5\, f_4(x)$

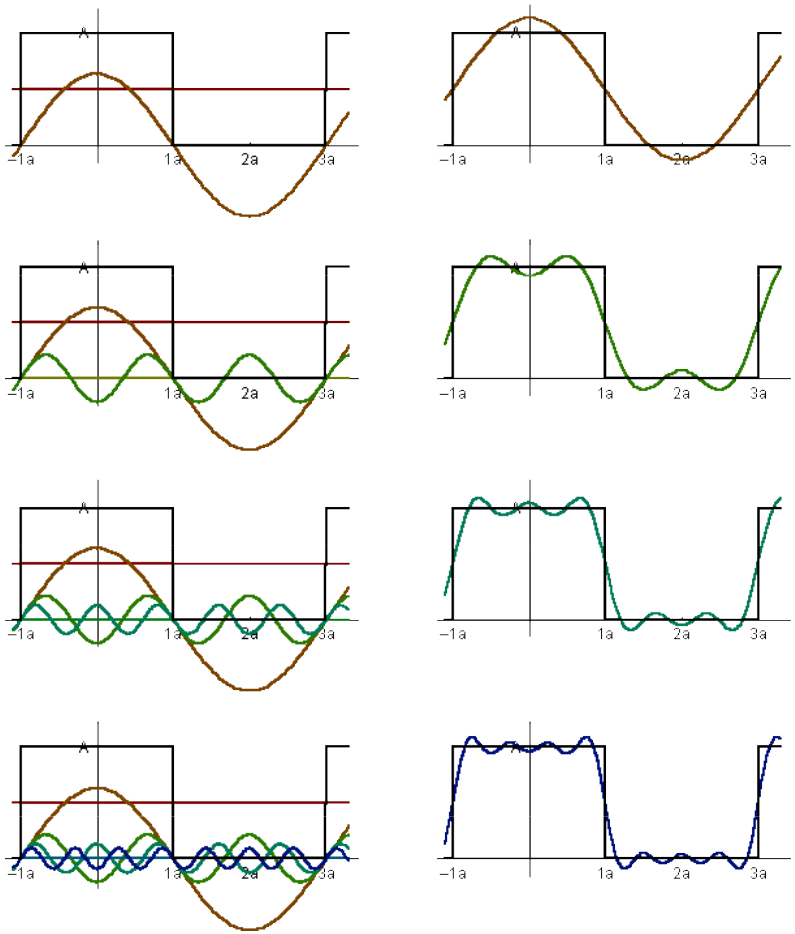


频率叠加

Fourier Transform

Represent a function as a weighted sum of sines and cosines

Joseph Fourier 1768 - 1830

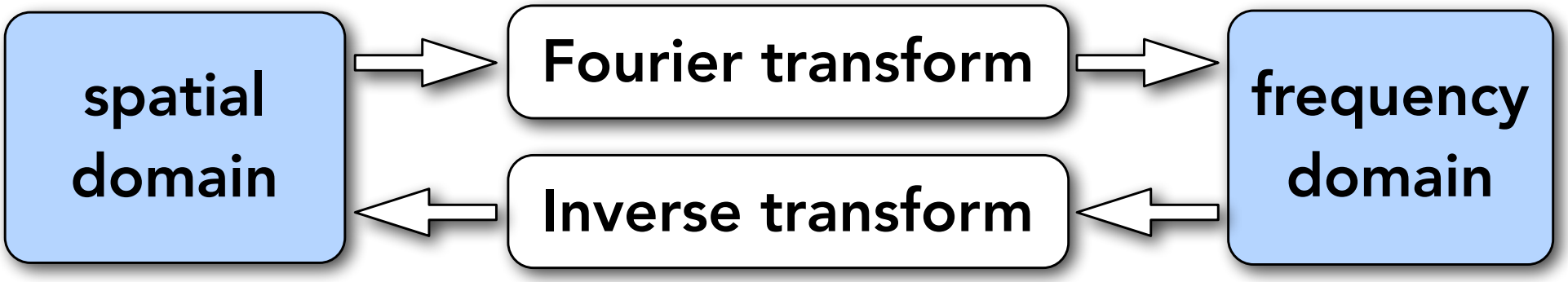


$$f(x) = \frac{A}{2} + \frac{2A\cos(t\omega)}{\pi} - \frac{2A\cos(3t\omega)}{3\pi} + \frac{2A\cos(5t\omega)}{5\pi} - \frac{2A\cos(7t\omega)}{7\pi} + \cdots$$

傅立叶级数：周期信号可以通过傅立叶展开，表示成为一系列简单周期函数的线性组合。

Fourier Transform Decomposes A Signal Into Frequencies

$$f(x) \qquad F(\omega) = \int_{-\infty}^{\infty} f(x) e^{-2\pi i \omega x} dx \qquad F(\omega)$$



$$f(x) = \int_{-\infty}^{\infty} F(\omega) e^{2\pi i \omega x} d\omega$$

Recall $e^{ix} = \cos x + i \sin x$

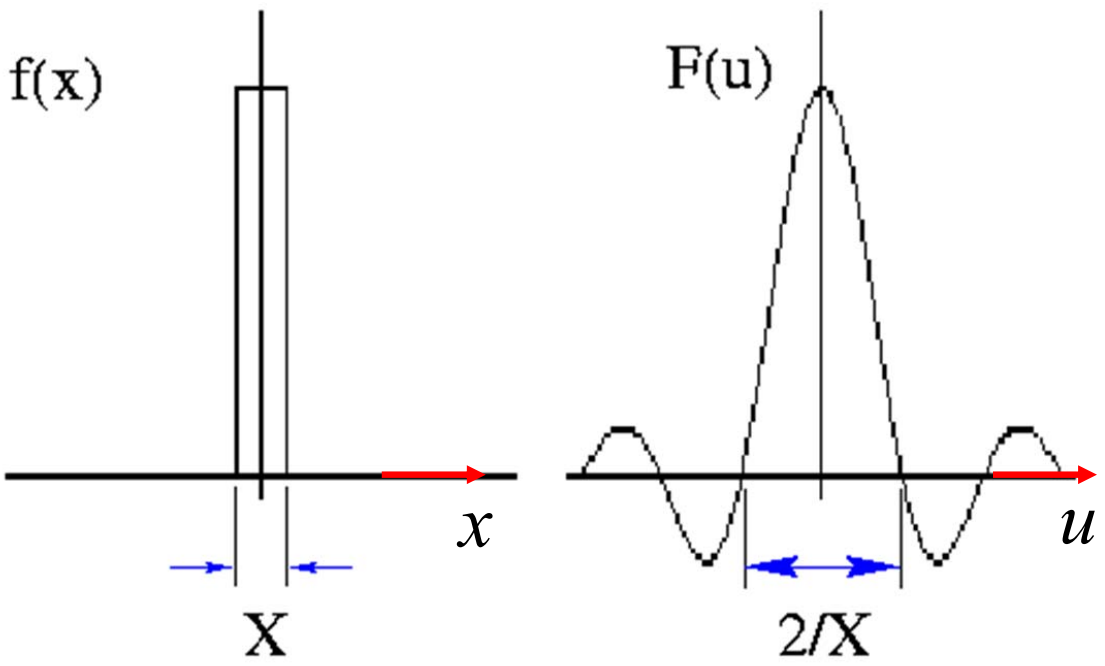
傅立叶变换：任意信号可以通过傅立叶变换分解成一系列不同频率的正余弦函数。

$F(\omega)$是$f(x)$和某个基函数（与频率ω相关）的内积，可看成是在该基函数上的投影大小。

Example: 1D Fourier Transform

$$f(x) = \begin{cases} 1, & |x| < \frac{X}{2}, \\ 0, & |x| \geq \frac{X}{2}. \end{cases}$$

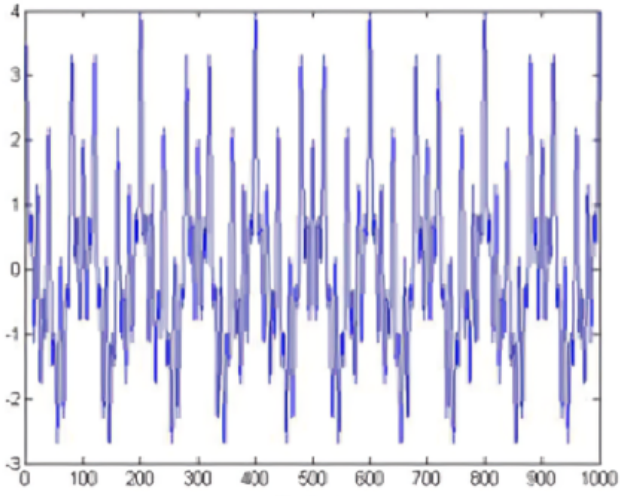
$$\begin{aligned} F(u) &= \int_{-\infty}^{\infty} f(x) e^{-j2\pi ux} dx \\ &= \int_{-X/2}^{X/2} e^{-j2\pi ux} dx \\ &= \frac{1}{-j2\pi u} [e^{-j2\pi uX/2} - e^{j2\pi uX/2}] \\ &= X \frac{\sin(\pi Xu)}{(\pi Xu)} = X \mathrm{sinc}(\pi \mathrm{Xu}). \end{aligned}$$



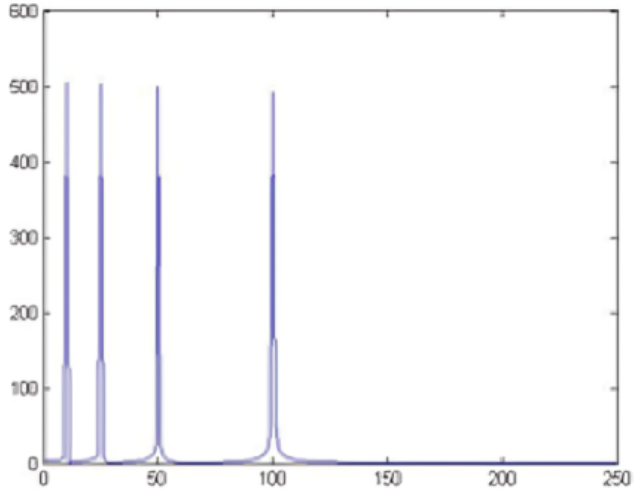
Discrete Fourier Transform

$$F(\omega) = \int_{-\infty}^{\infty} f(x) e^{-2\pi i \omega x} dx$$

离散信号的傅立叶变换 $X_k = \sum_{m=0}^{n-1} x_m e^{-i 2\pi k m / n} \qquad k = 0, \ldots, n-1$



FFT



时域信号

信号频谱

$x(t) = \cos(2\pi * 10t) + \cos(2\pi * 25t) + \cos(2\pi * 50t) + \cos(2\pi * 100t)$

10, 25, 50, 100Hz

2D Fourier Transform

Definition

$$F(u,v) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)e^{-j2\pi(ux+vy)}\,dx\,dy,$$
$$f(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} F(u,v)e^{j2\pi(ux+vy)}\,du\,dv$$

In 1D the Fourier transform is based on a decompostion into functions $e^{j2\pi ux} = \cos 2\pi ux + j \sin 2\pi ux$ which form an orthogonal basis. Similarly in 2D

$$e^{j2\pi(ux+vy)} = \cos 2\pi(ux+vy) + j \sin 2\pi(ux+vy)$$

$F(u,v)$ is complex in general,

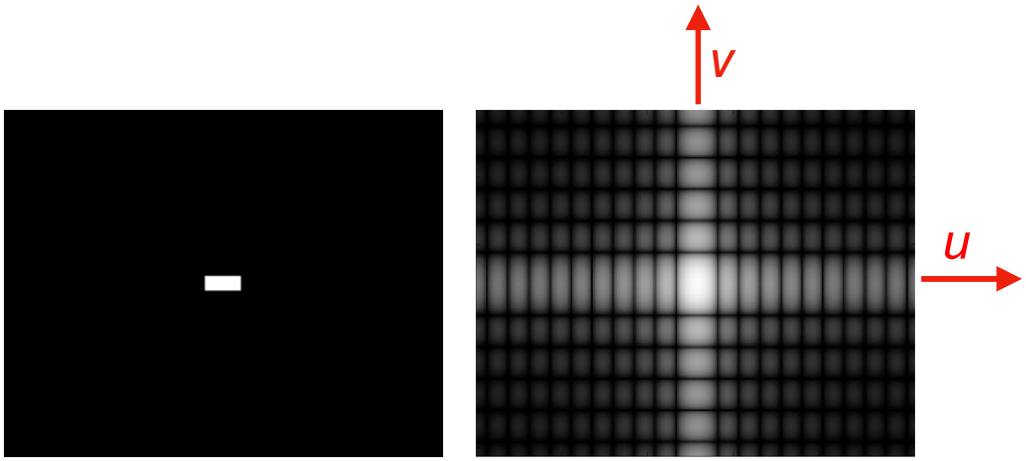
$$F(u,v) = F_{\mathrm{R}}(u,v) + jF_{\mathrm{I}}(u,v)$$

$|F(u,v)|$ is the magnitude spectrum

$\arctan(F_{\mathrm{I}}(u,v)/F_{\mathrm{R}}(u,v))$ is the phase angle spectrum.

Example: 2D Fourier Transform

rectangle centred at origin
with sides of length X and Y



$$F(u,v) = \int\int f(x,y)e^{-j2\pi(ux+vy)}dxdy,$$

$$= \int_{-X/2}^{X/2} e^{-j2\pi ux}dx \int_{-Y/2}^{Y/2} e^{-j2\pi vy}dy, \quad \text{separability}$$

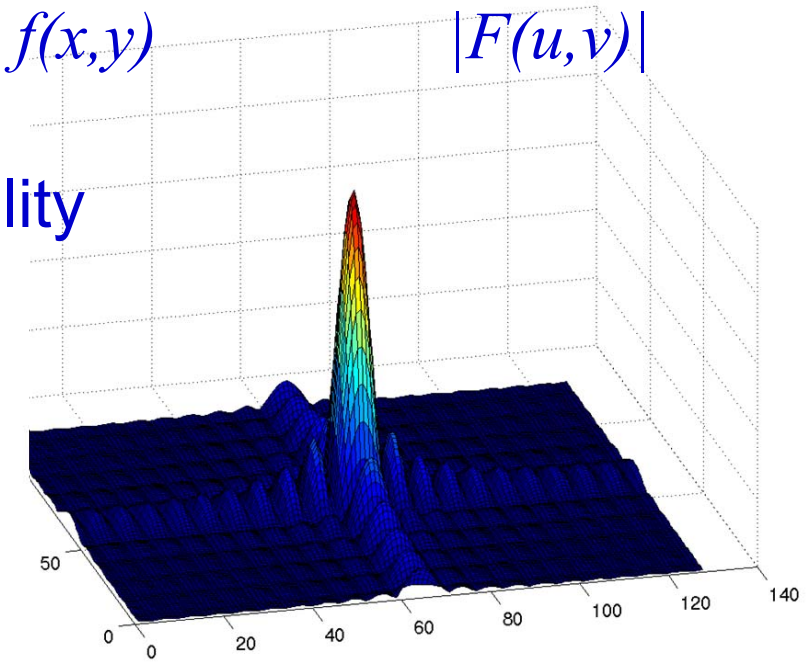
$$= \left[\frac{e^{-j2\pi ux}}{-j2\pi u}\right]_{-X/2}^{X/2} \left[\frac{e^{-j2\pi vy}}{-j2\pi v}\right]_{-Y/2}^{Y/2},$$

$$= \frac{1}{-j2\pi u}\left[e^{-juX} - e^{juX}\right] \frac{1}{-j2\pi v}\left[e^{-jvY} - e^{jvY}\right],$$

$$= XY\left[\frac{sin(\pi Xu)}{\pi Xu}\right]\left[\frac{sin(2\pi Yv)}{\pi Yv}\right]$$

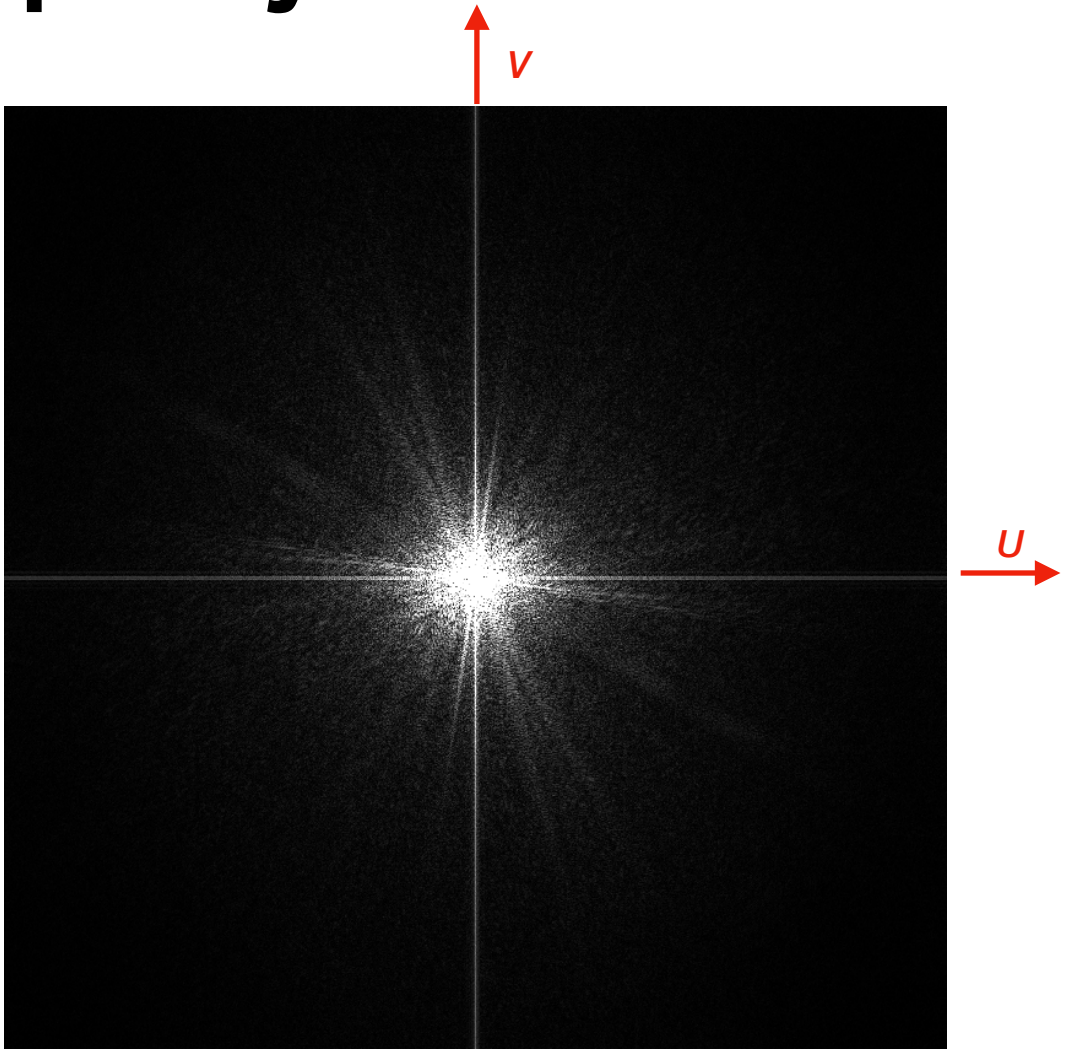
$$= XY\mathrm{sinc}(\pi \mathrm{Xu})\mathrm{sinc}(\pi \mathrm{Yv}).$$

$f(x,y)$ $|F(u,v)|$



$|F(u,v)|$

Example: Image Frequency Content



Spatial Domain

$f(x, y)$

2D DFT

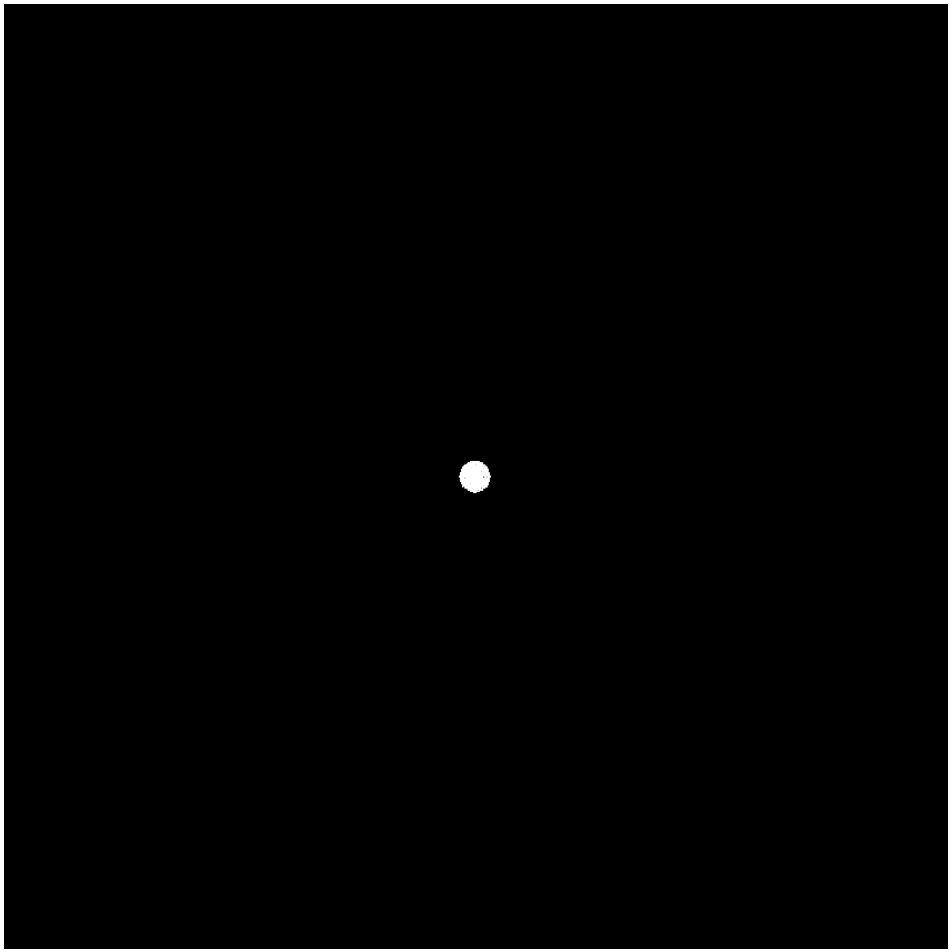
Frequency Domain

$|F(u, v)|$

Filtering = Getting rid of certain frequency contents

Filter Out High Frequencies (Blur)

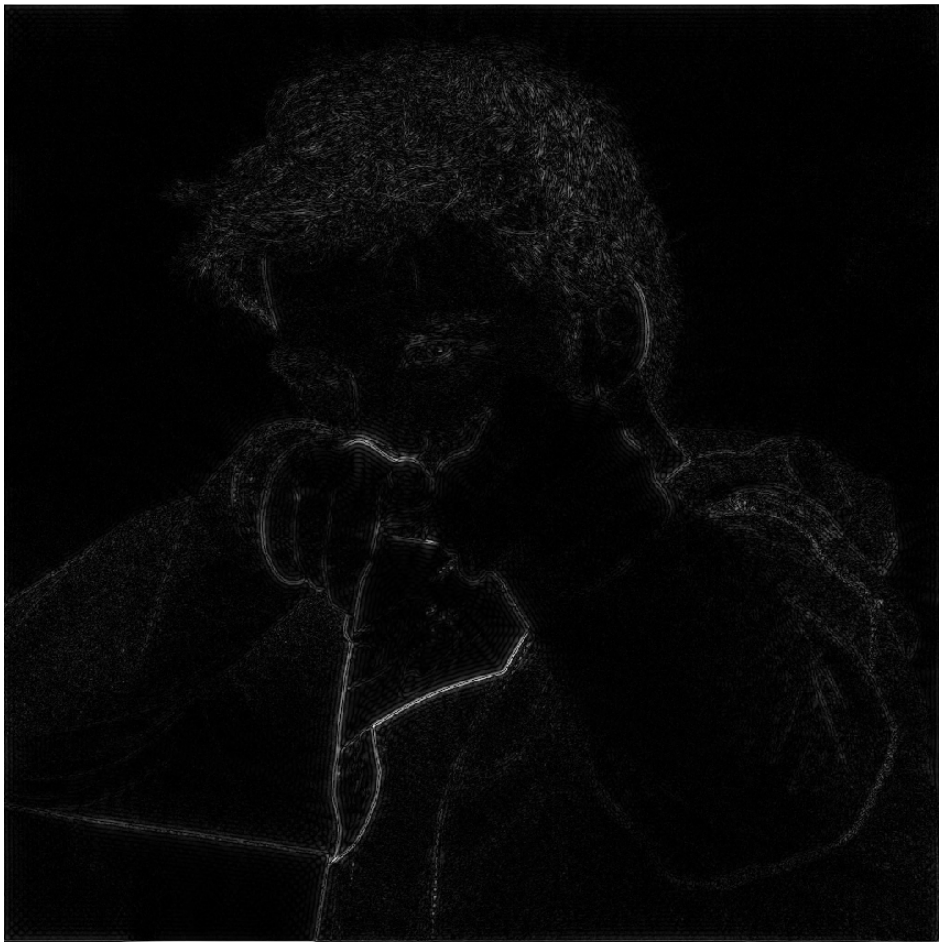
Spatial Domain



Frequency Domain

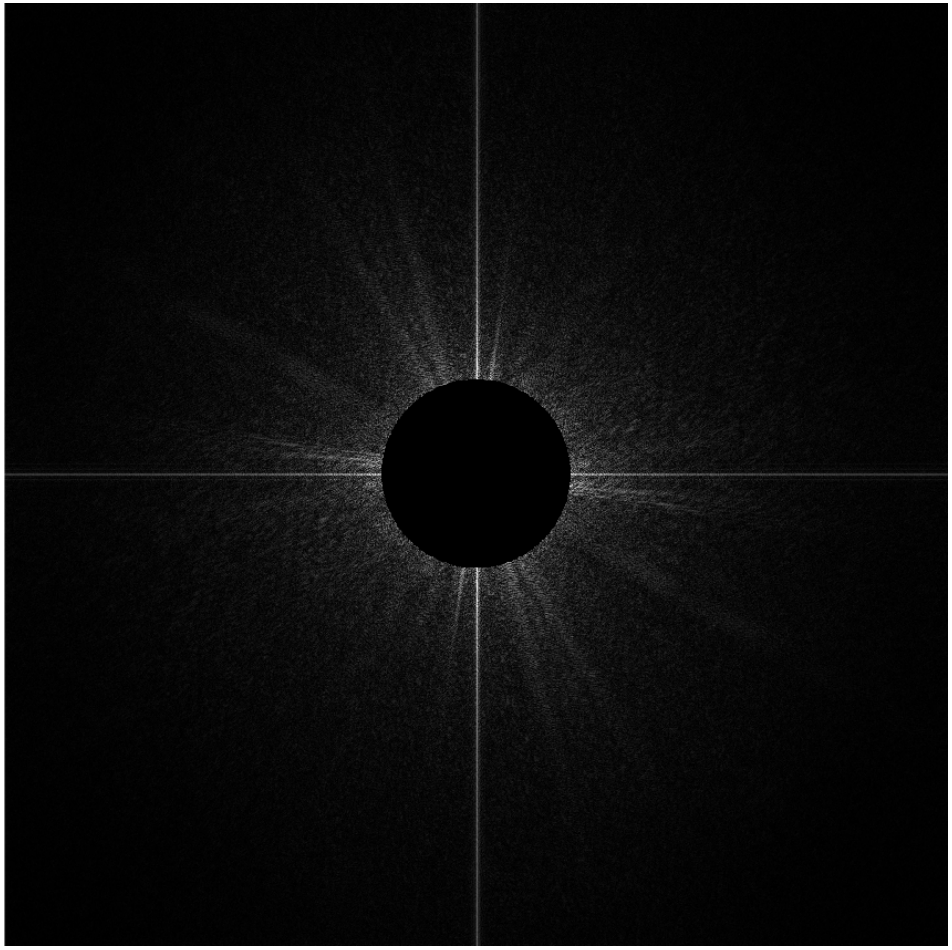
(Low frequency = Smooth gradients)

Filter Out Low Frequencies Only (Edges)



Spatial Domain

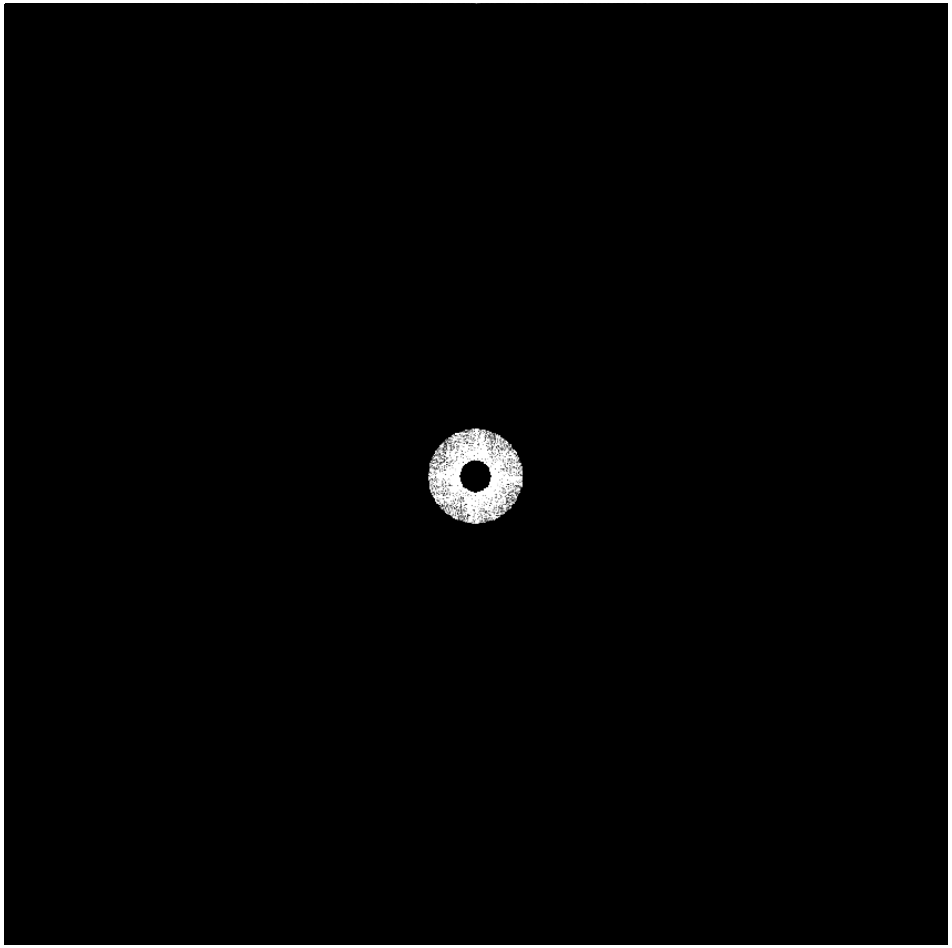
(High frequency = Sharp edges)



Frequency Domain

Filter Out Low and High Frequencies

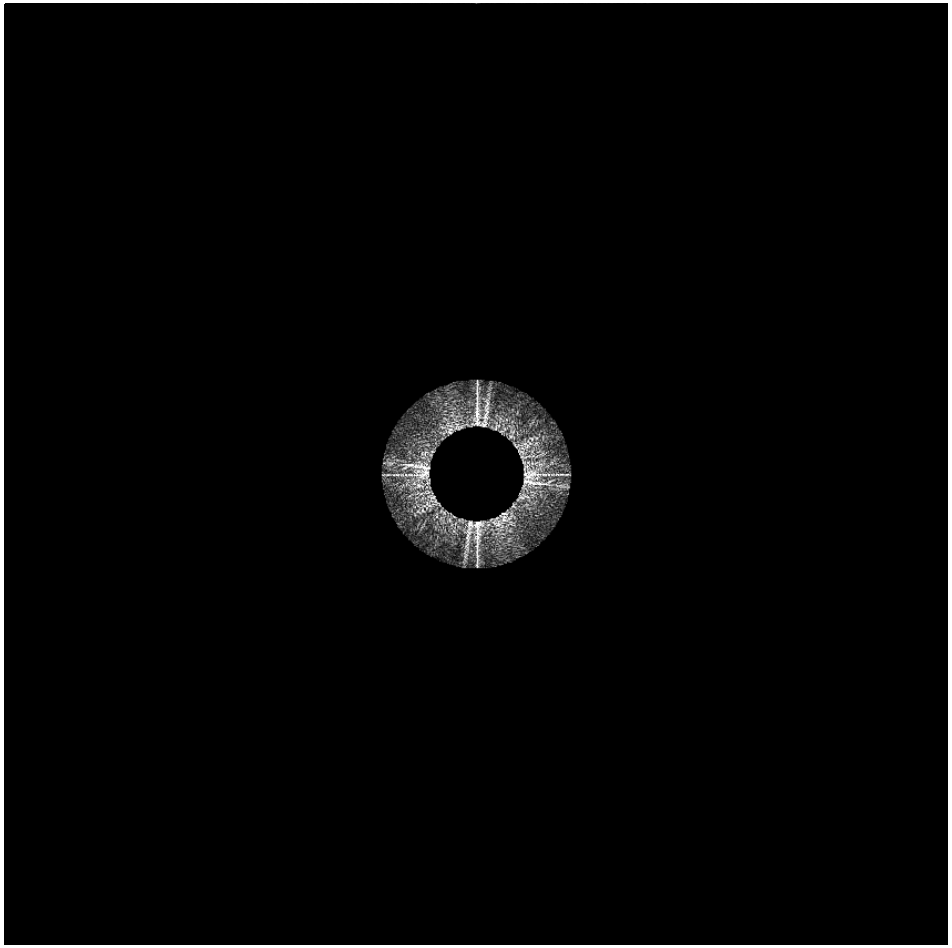
Spatial Domain



Frequency Domain

Filter Out Low and High Frequencies

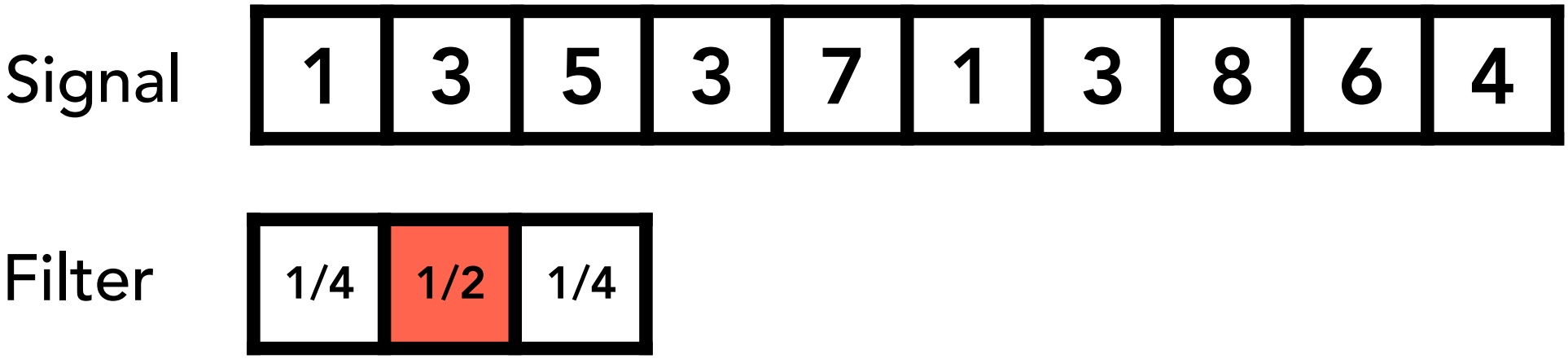
Spatial Domain



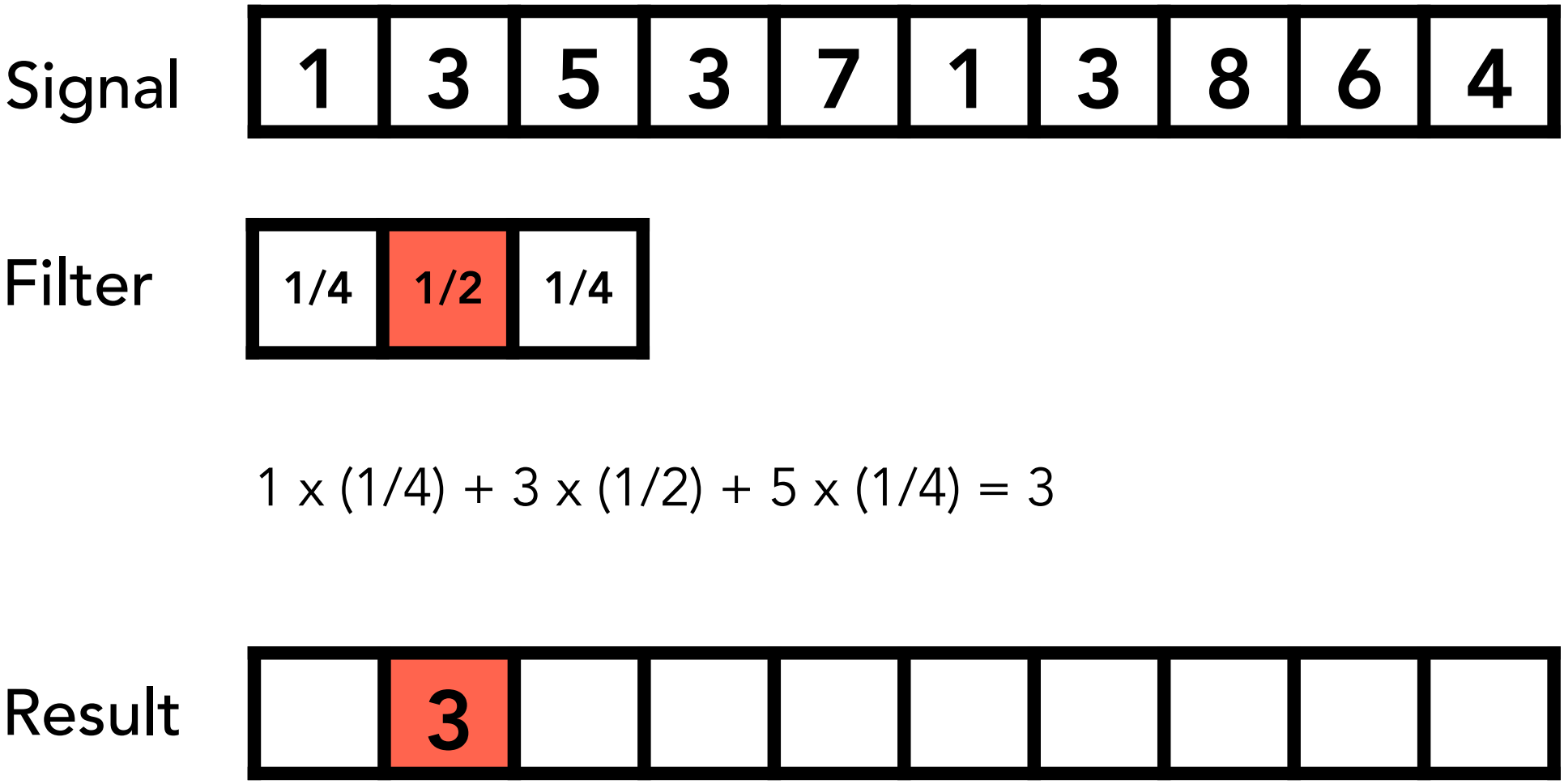
Frequency Domain

Filtering = Convolution

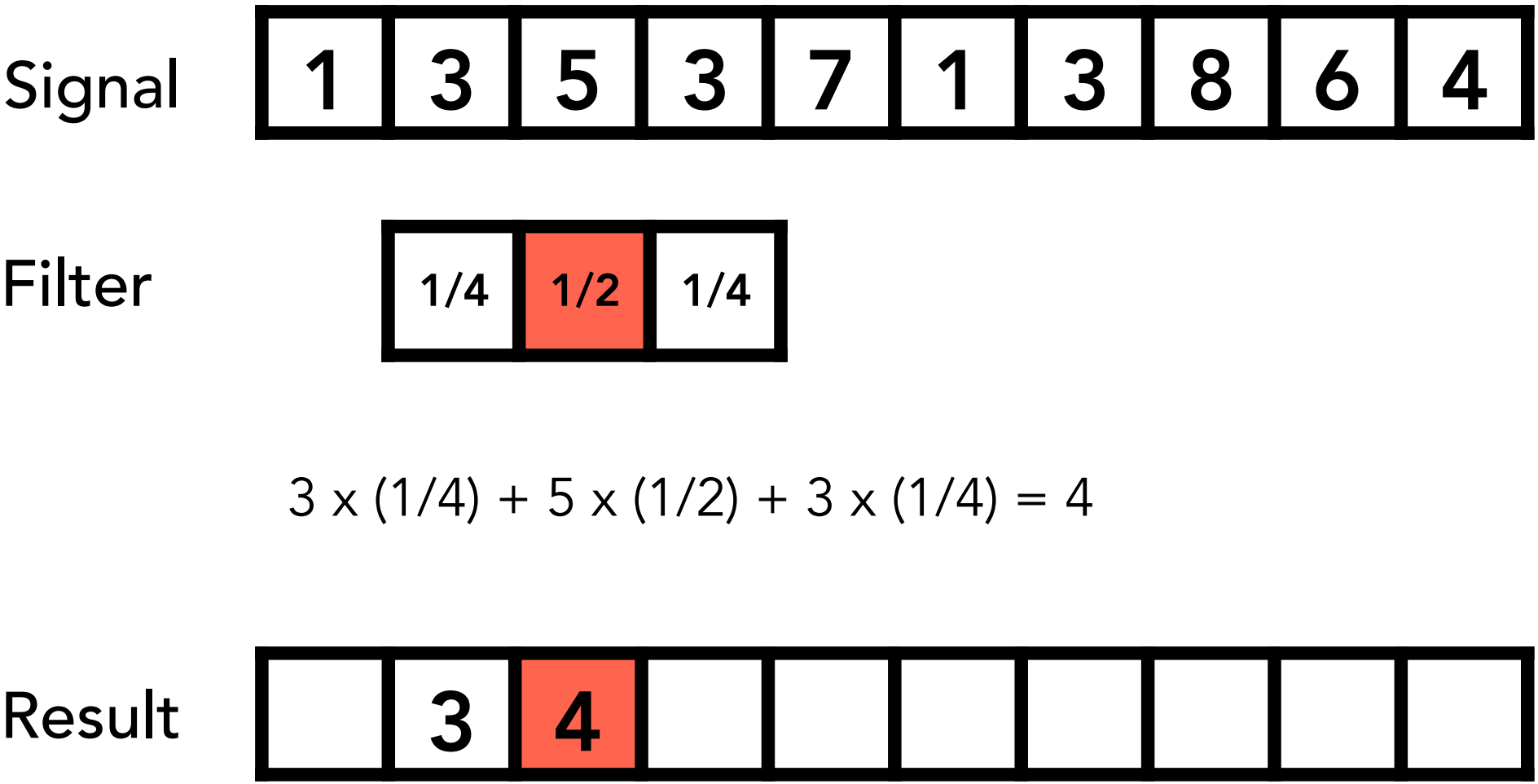
Convolution



Convolution



Convolution



Convolution Function

卷积函数定义 $(f * g)(t) = f(t) * g(t) = \int_{-\infty}^{+\infty} f(\tau)g(t-\tau)d\tau$

对两个函数重叠长度的乘积的积分

$g(n)$为滤波器（卷积核）

离散形式 $f(n) * g(n) = \sum_{m=-\infty}^{+\infty} f(n-m)g(m)$

例子： $g(n) = 1/3, -1 \leq n \leq 1$

卷积后$f(n)$变为

$$(f * g)(n) = f(n-1)g(-1) + f(n-0)g(0) + f(n+1)g(1)$$

如果将参加卷积的一个函数看作区间的指示函数（只在一个区间有定义，其余地方为*0*），卷积还可以被看作是"滑动平均"（定义域上滑动，重叠区域平均）

Convolution Theorem

时域卷积定理： $F[f(t) * g(t)] = F_f(\omega) \cdot F_g(\omega)$

两信号在时域的卷积积分对应于在频域中该两信号的傅立叶变换的乘积。

频域卷积定理： $F[f(t) \cdot g(t)] = \frac{1}{2\pi} F_f(\omega) * F_g(\omega)$

两信号在时域的乘积对应于这两个信号傅立叶变换的卷积除以2π。

时域定理证明

首先，卷积定义为

$$f_1(t) * f_2(t) = \int_{-\infty}^{+\infty} f_1(\tau) f_2(t-\tau) d\tau$$

然后，代入傅立叶变换公式

$$F[f(t)] = F(\omega) = \int_{-\infty}^{+\infty} f(t) e^{-j\omega t} dt$$

由此可得

$$F[f_1(t) * f_2(t)] = \int_{-\infty}^{+\infty} \left[\int_{-\infty}^{+\infty} f_1(\tau) f_2(t-\tau) d\tau \right] e^{-j\omega t} dt$$

$$= \int_{-\infty}^{+\infty} f_1(\tau) \left[\int_{-\infty}^{+\infty} f_2(t-\tau) e^{-j\omega t} dt \right] d\tau$$

$$= \int_{-\infty}^{+\infty} f_1(\tau) F_2(\omega) e^{-j\omega\tau} d\tau$$

$$= F_2(\omega) \int_{-\infty}^{+\infty} f_1(\tau) e^{-j\omega\tau} d\tau$$

$$= F_2(\omega) F_1(\omega)$$

频域定理证明

设 $F_1(\omega) = F[f_1(t)]$，$F_2(\omega) = F[f_2(t)]$，IF 表示傅立叶逆变换，则

$$IF[F_1(\omega) * F_2(\omega)] = IF\left[\int_{-\infty}^{+\infty} F_1(\mu) F_2(\omega-\mu) d\mu \right]$$

$$= \frac{1}{2\pi} \int_{-\infty}^{+\infty} \left[\int_{-\infty}^{+\infty} F_1(\mu) F_2(\omega-\mu) d\mu \right] e^{j\omega t} d\omega$$

$$= \frac{1}{2\pi} \int_{-\infty}^{+\infty} \left[\int_{-\infty}^{+\infty} F_1(\mu) F_2(\omega-\mu) e^{j\mu t} e^{j(\omega-\mu)t} d\omega \right] d\mu$$

$$= \frac{1}{2\pi} \int_{-\infty}^{+\infty} F_1(\mu) e^{j\mu t} \left[\int_{-\infty}^{+\infty} F_2(\omega-\mu) e^{j(\omega-\mu)t} d(\omega-\mu) \right] d\mu$$

$$= f_2(t) \int_{-\infty}^{+\infty} F_1(\mu) e^{j\mu t} d\mu$$

$$= 2\pi f_1(t) f_2(t)$$

因此有

$$f_1(t) f_2(t) = \frac{1}{2\pi} IF[F_1(\omega) * F_2(\omega)] = IF\left[\frac{1}{2\pi} F_1(\omega) * F_2(\omega) \right]$$

Convolution Theorem

Convolution in the spatial domain is equal to multiplication in the frequency domain, and vice versa

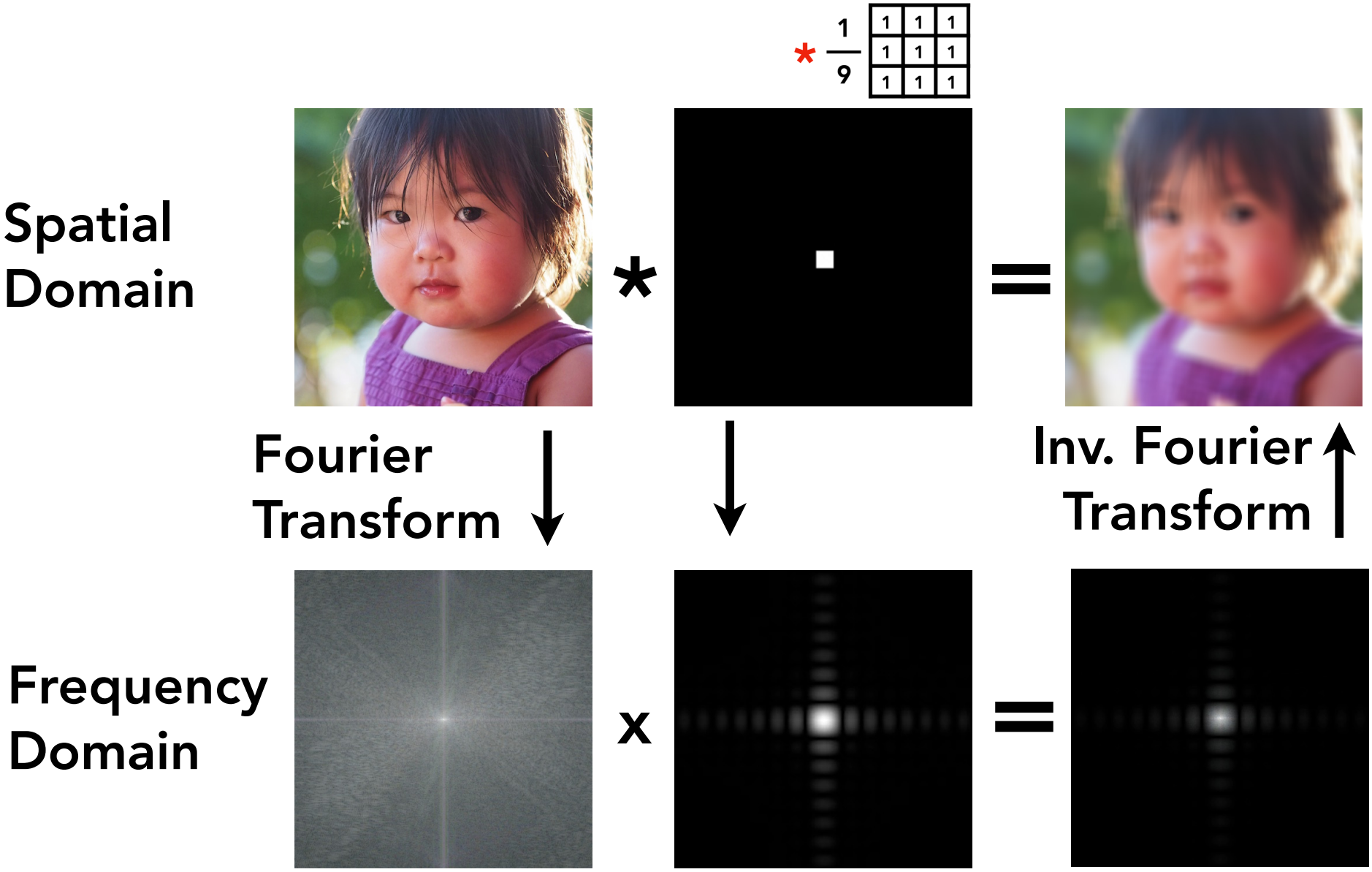
Option 1:

- **Filter by convolution in the spatial domain**

Option 2:

- **Transform to frequency domain (Fourier transform)**
- **Multiply by Fourier transform of convolution kernel**
- **Transform back to spatial domain (inverse Fourier)**

Convolution Theorem



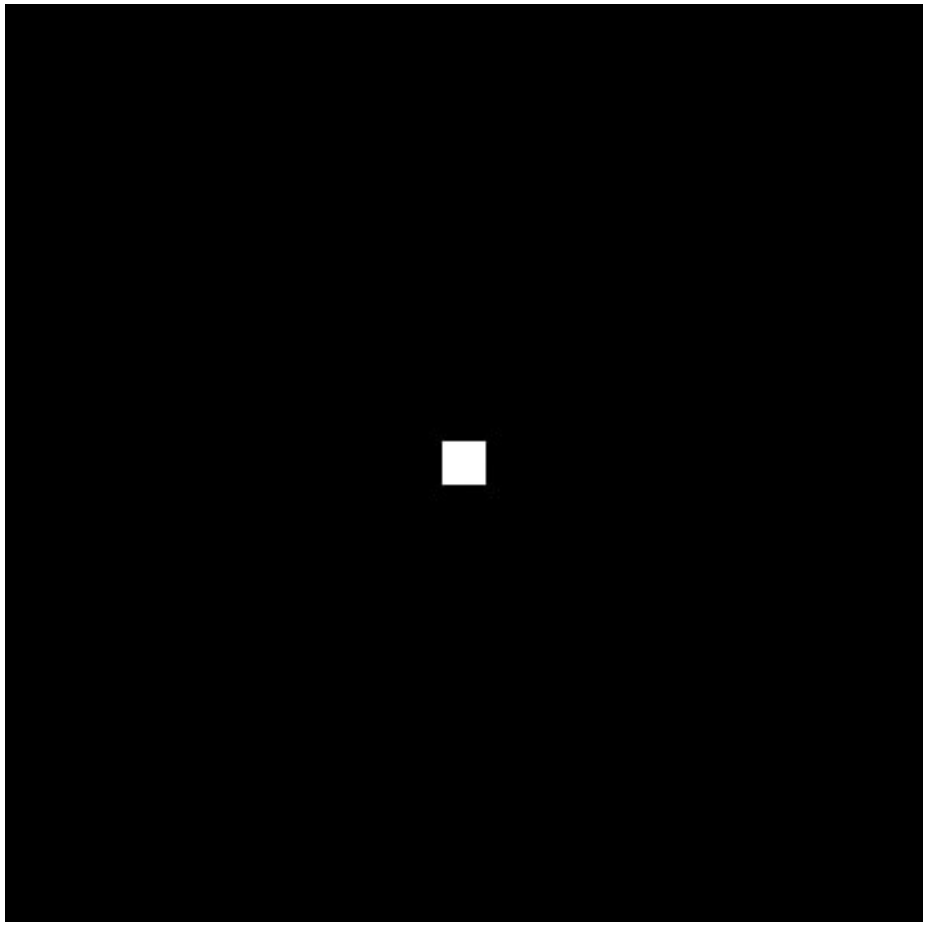
Box Filter

$$\frac{1}{9}$$

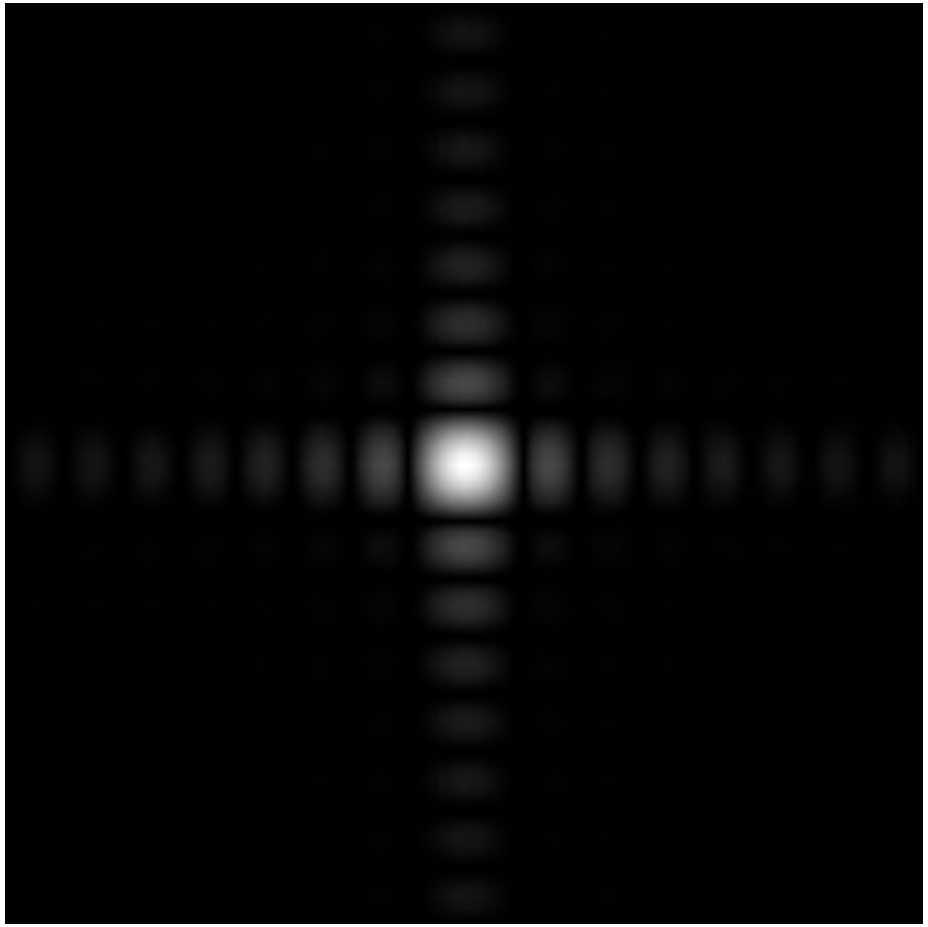
1	1	1
1	1	1
1	1	1

Example: 3x3 box filter

Box Function = “Low Pass” Filter

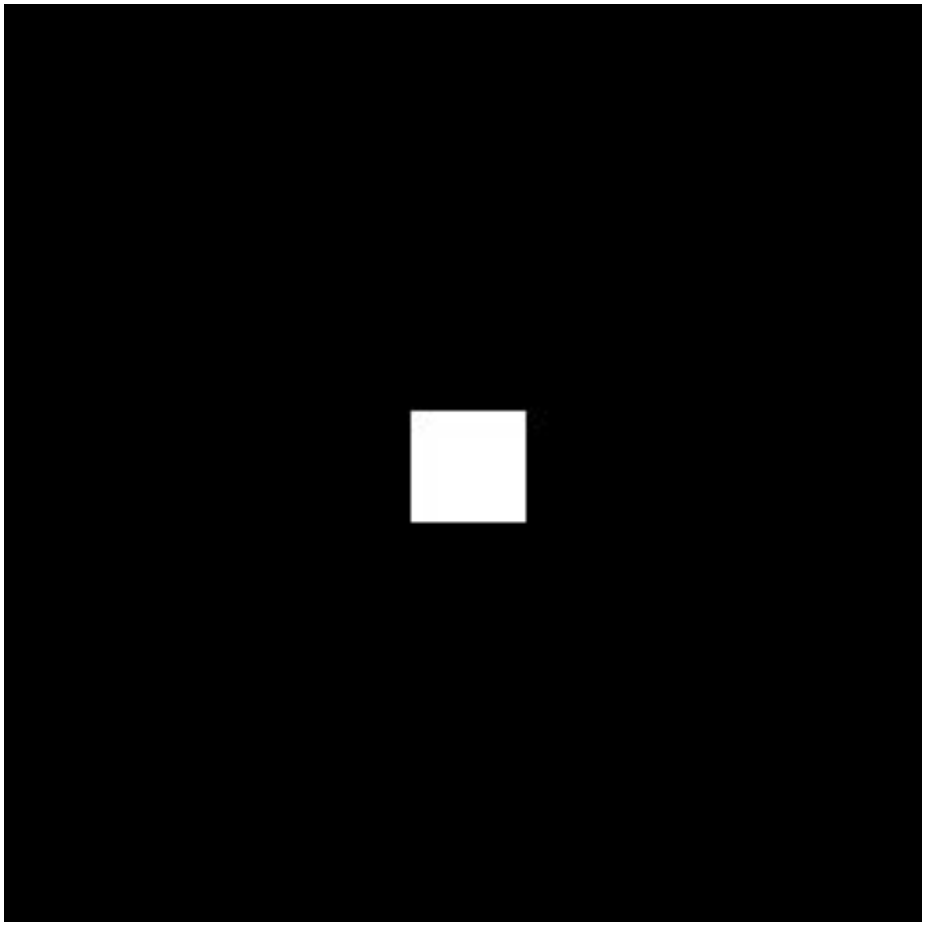


Spatial Domain

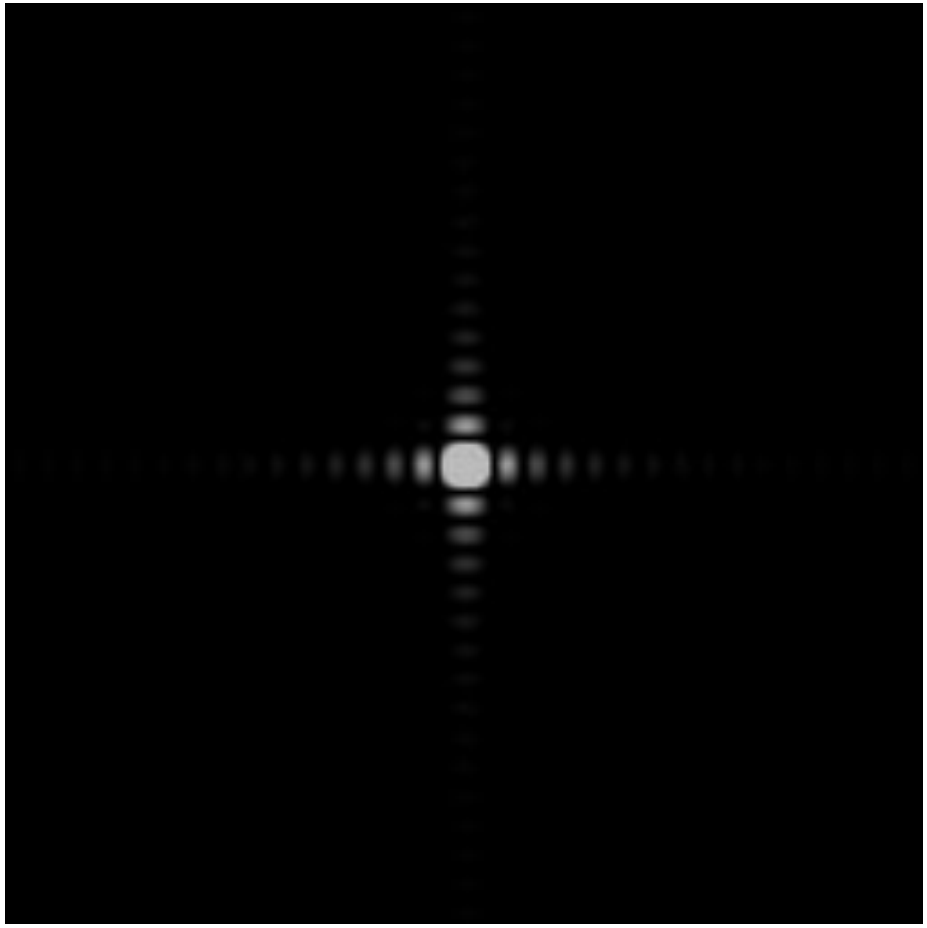


Frequency Domain

Wider Filter Kernel = Lower Frequencies



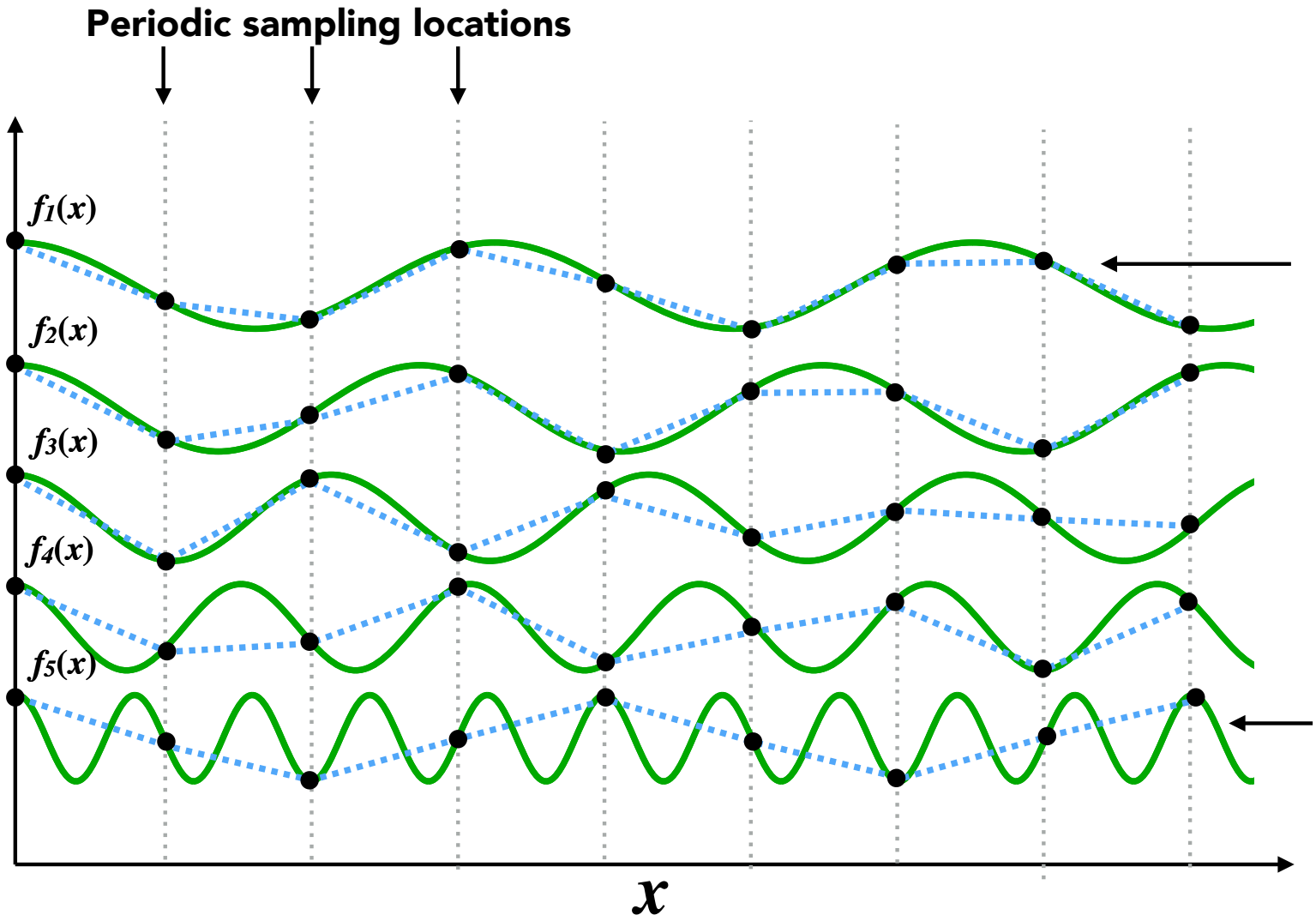
Spatial Domain



Frequency Domain

Having introduced frequency space and filtering, we now explain aliasing and anti-aliasing in signal sampling.

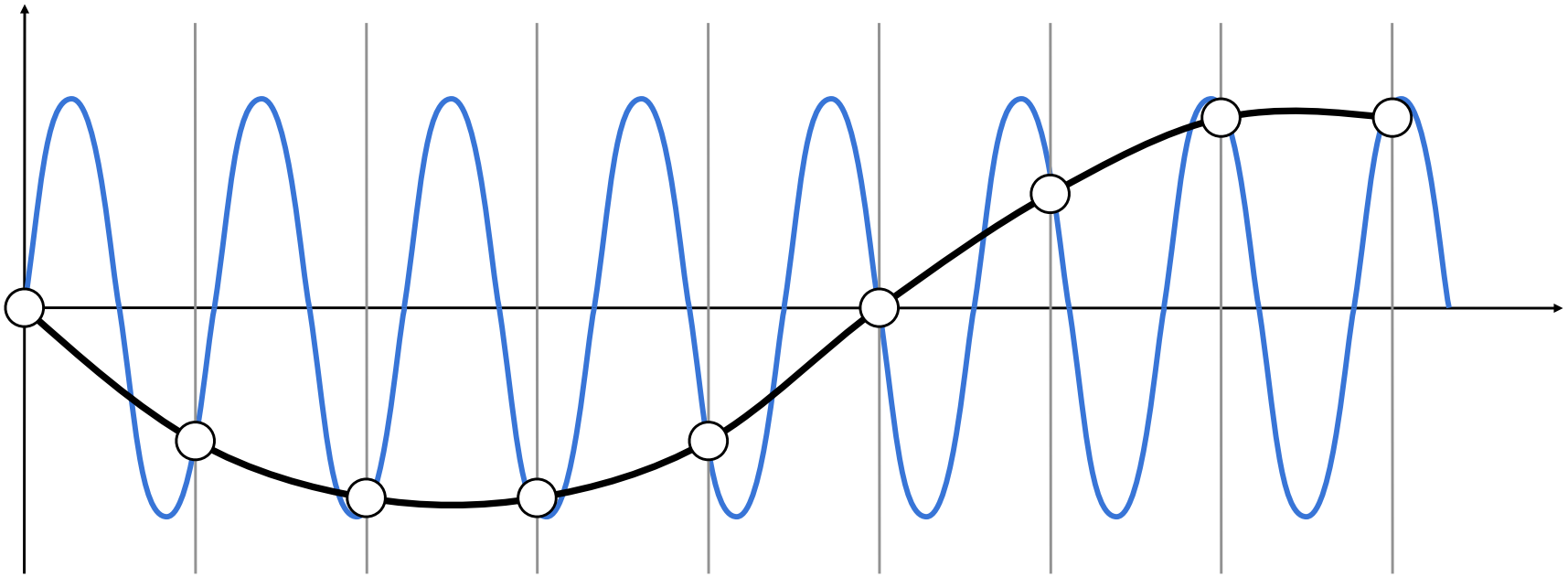
Higher Frequencies Need Faster Sampling



Low-frequency signal: sampled adequately for reasonable reconstruction

High-frequency signal is insufficiently sampled: reconstruction incorrectly appears to be from a low frequency signal

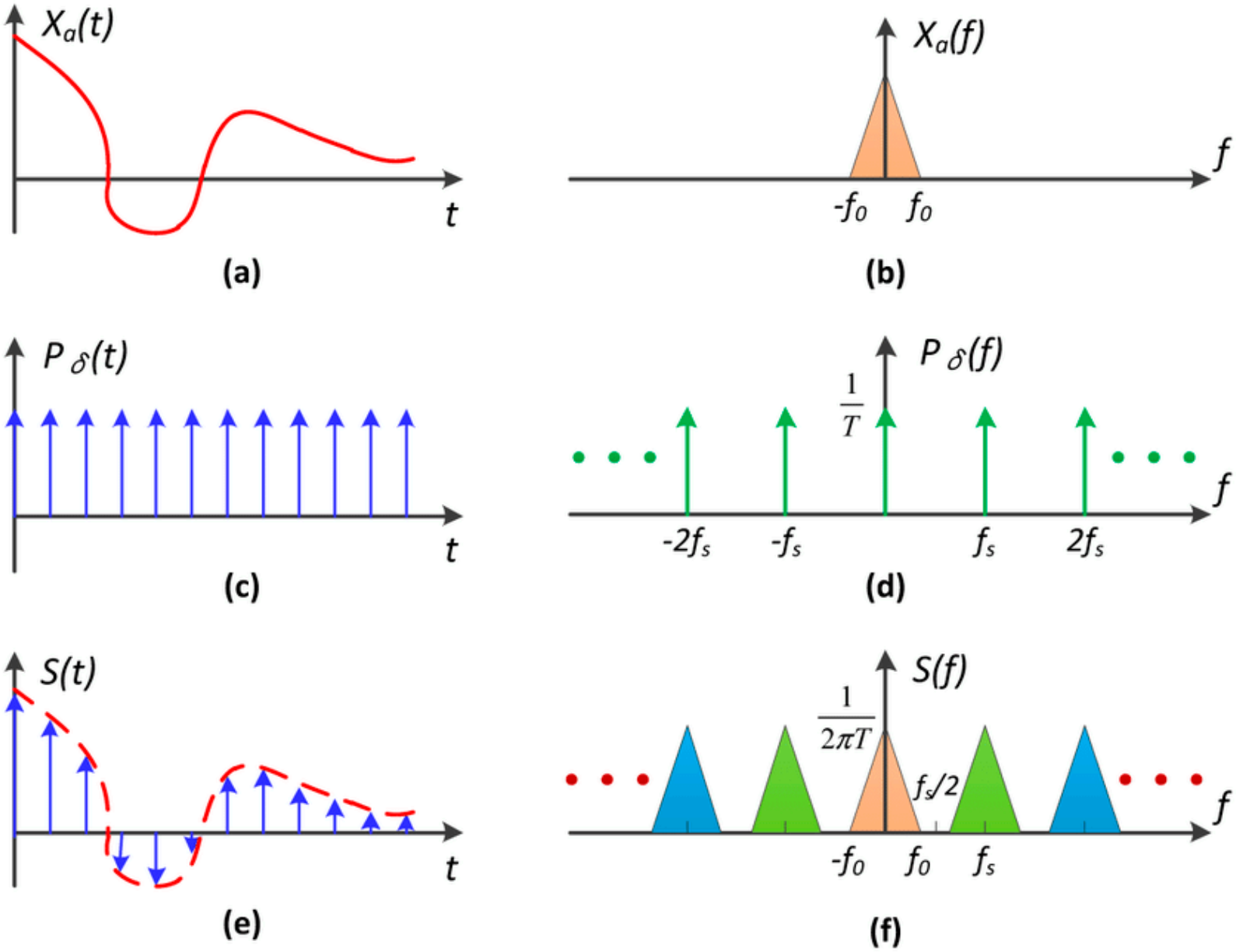
Undersampling Creates Frequency Aliases



High-frequency signal is insufficiently sampled: samples erroneously appear to be from a low-frequency signal

Two frequencies that are indistinguishable at a given sampling rate are called “aliases”

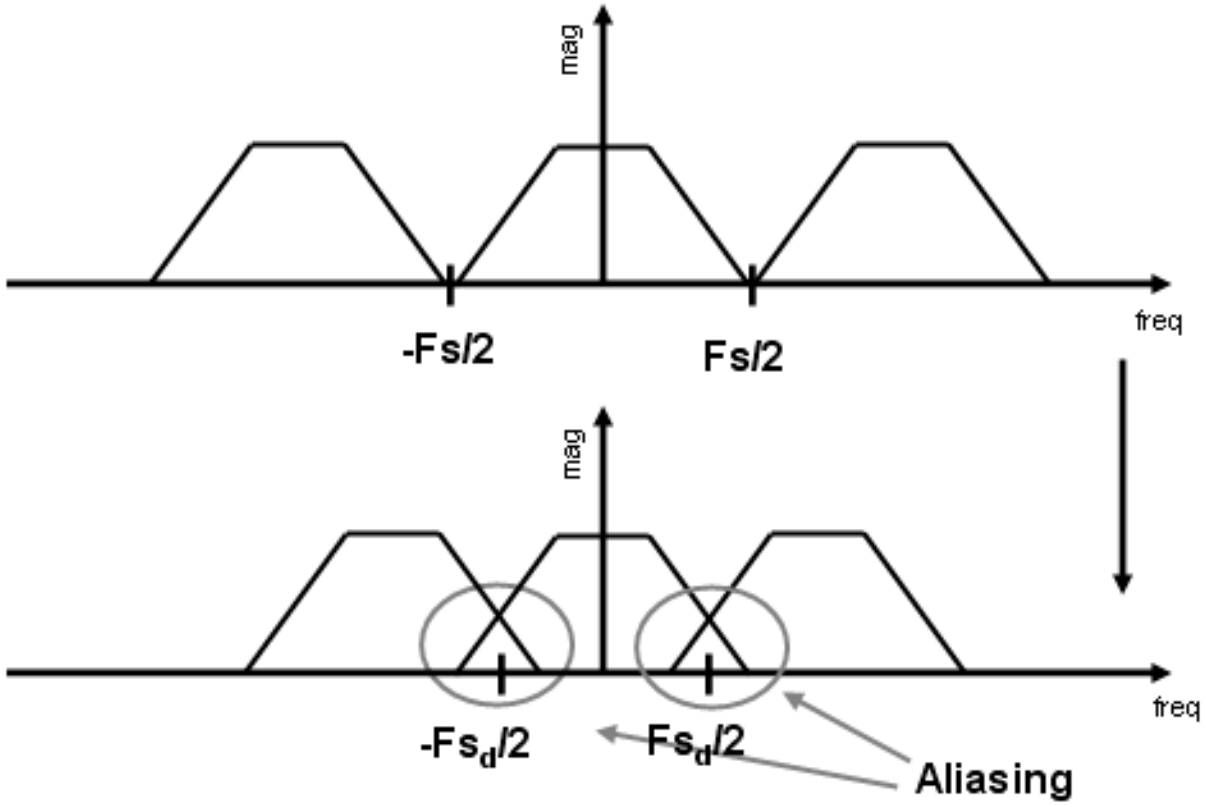
Sampling = Repeating Frequency Contents

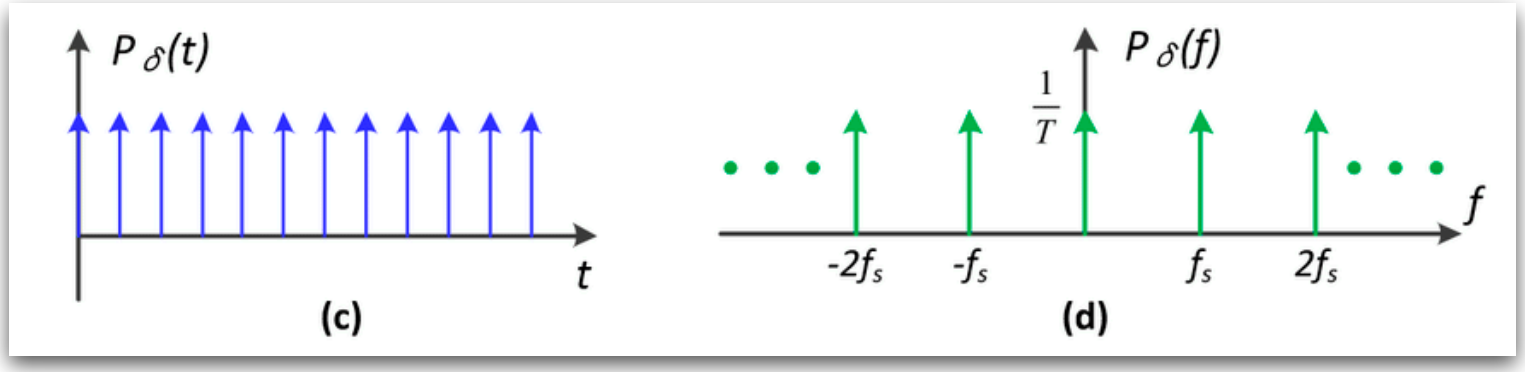


Aliasing = Mixed Frequency Contents

Dense sampling:

Sparse sampling:



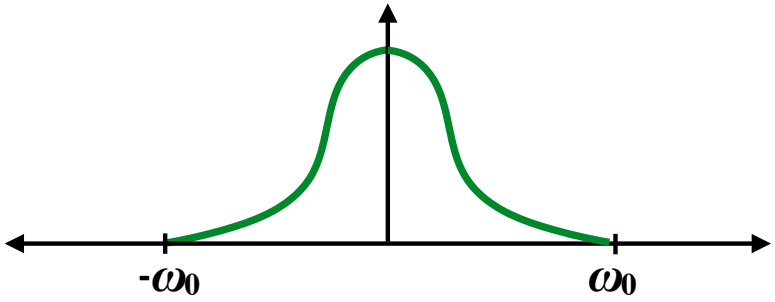


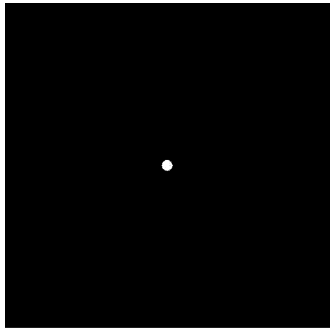
Nyquist-Shannon Theorem （香农采样定理）

- Consider a band-limited signal: has no frequencies above ω_0

  1D: consider low-pass filtered audio signal

  2D: recall the blurred image example from a few slides ago





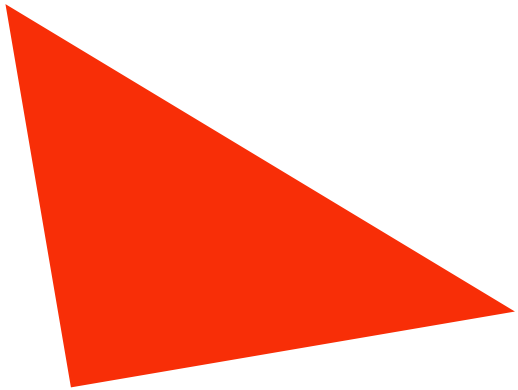
- The signal can be perfectly reconstructed if sampled with period T = 1 / 2ω_0

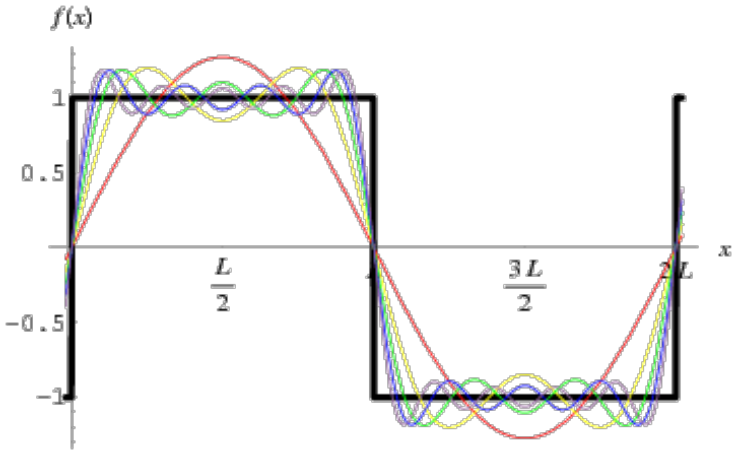
Challenges of Sampling-based Approaches

- Our signals are not always *band-limited* in computer graphics.

  Why?

Hint:





Antialiasing

How Can We Reduce Aliasing Error?

Increase sampling rate (increase Nyquist frequency)

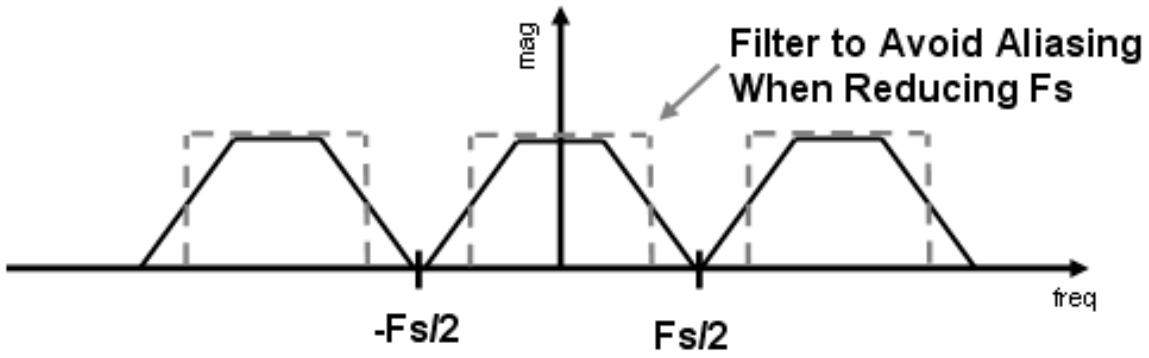
- Higher resolution displays, sensors, framebuffers…
- But: costly & may need very high resolution

Antialiasing

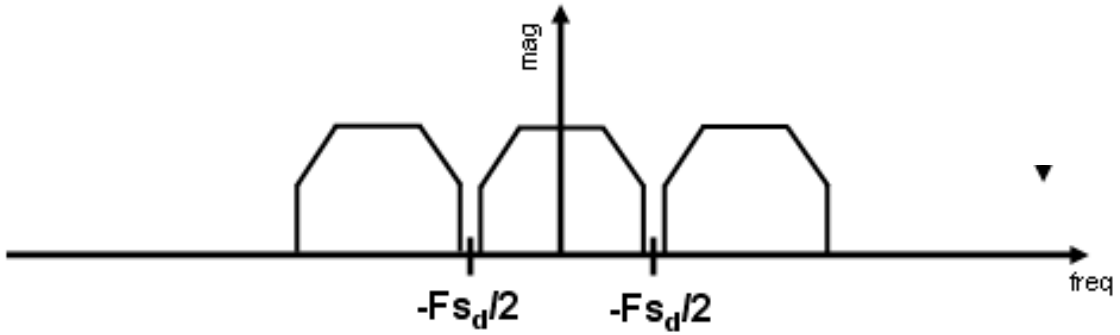
- Simple idea: remove (or reduce) signal frequencies above the Nyquist frequency before sampling
- How? Filter out high frequencies before sampling.

Antialiasing = Limiting, then repeating

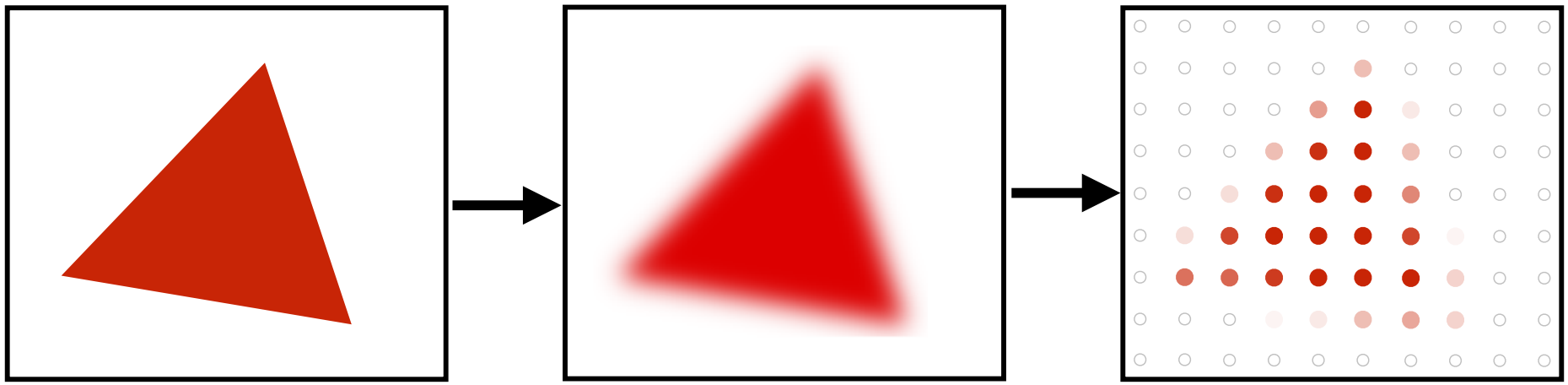
Filtering



Then sparse sampling



Antialiased Sampling



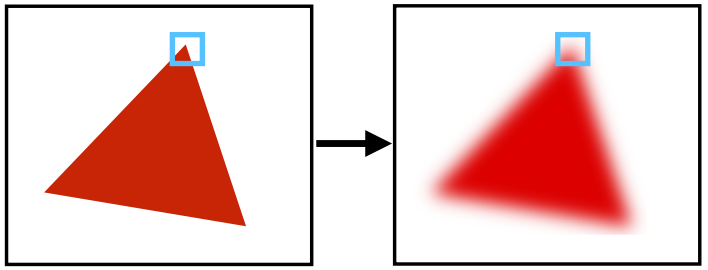
Pre-Filter

(remove frequencies above Nyquist)

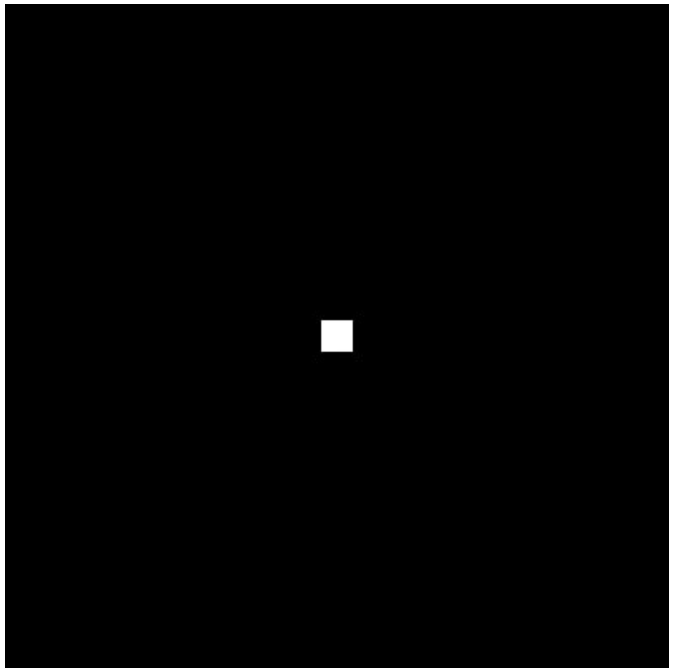
Sample

Note antialiased edges in rasterized triangle where pixel values take intermediate values

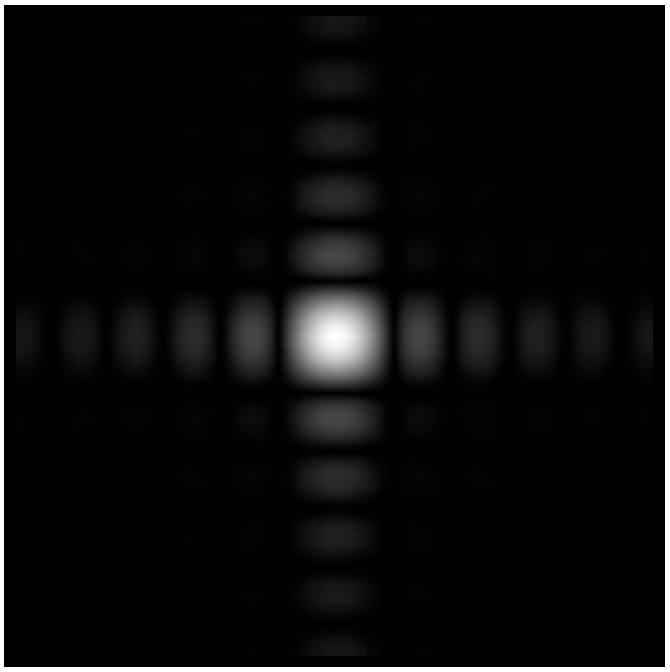
A Practical Pre-Filter



A 1 pixel-width box filter will attenuate frequencies whose period is less than or equal to 1 pixel-width



Spatial Domain



Frequency Domain

This is practical to implement — why?

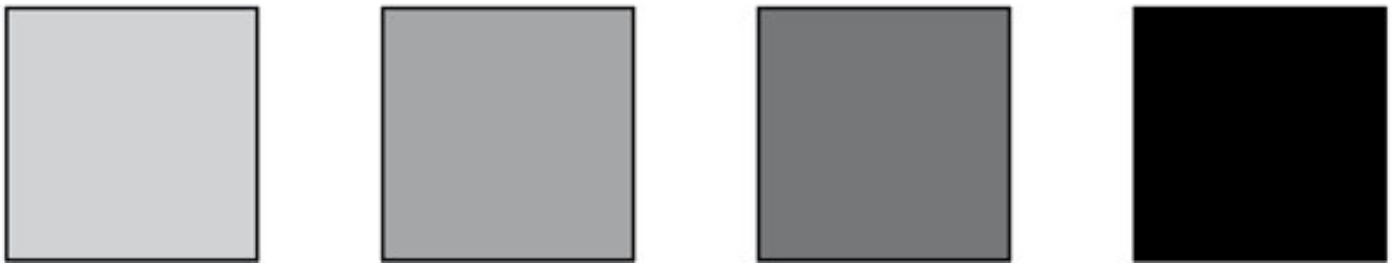
Antialiasing by Computing Average Pixel Value

等价于均值滤波

In rasterizing one triangle, the average value inside a pixel area of f(x,y) = inside(triangle,x,y) is equal to the area of the pixel covered by the triangle.

Original

Filtered

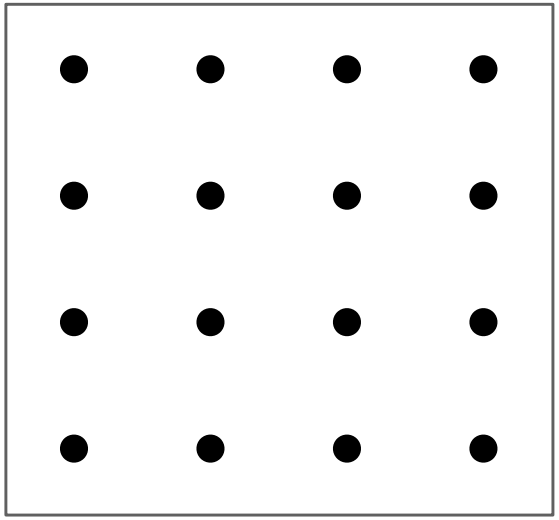


1 pixel width

Antialiasing By Supersampling

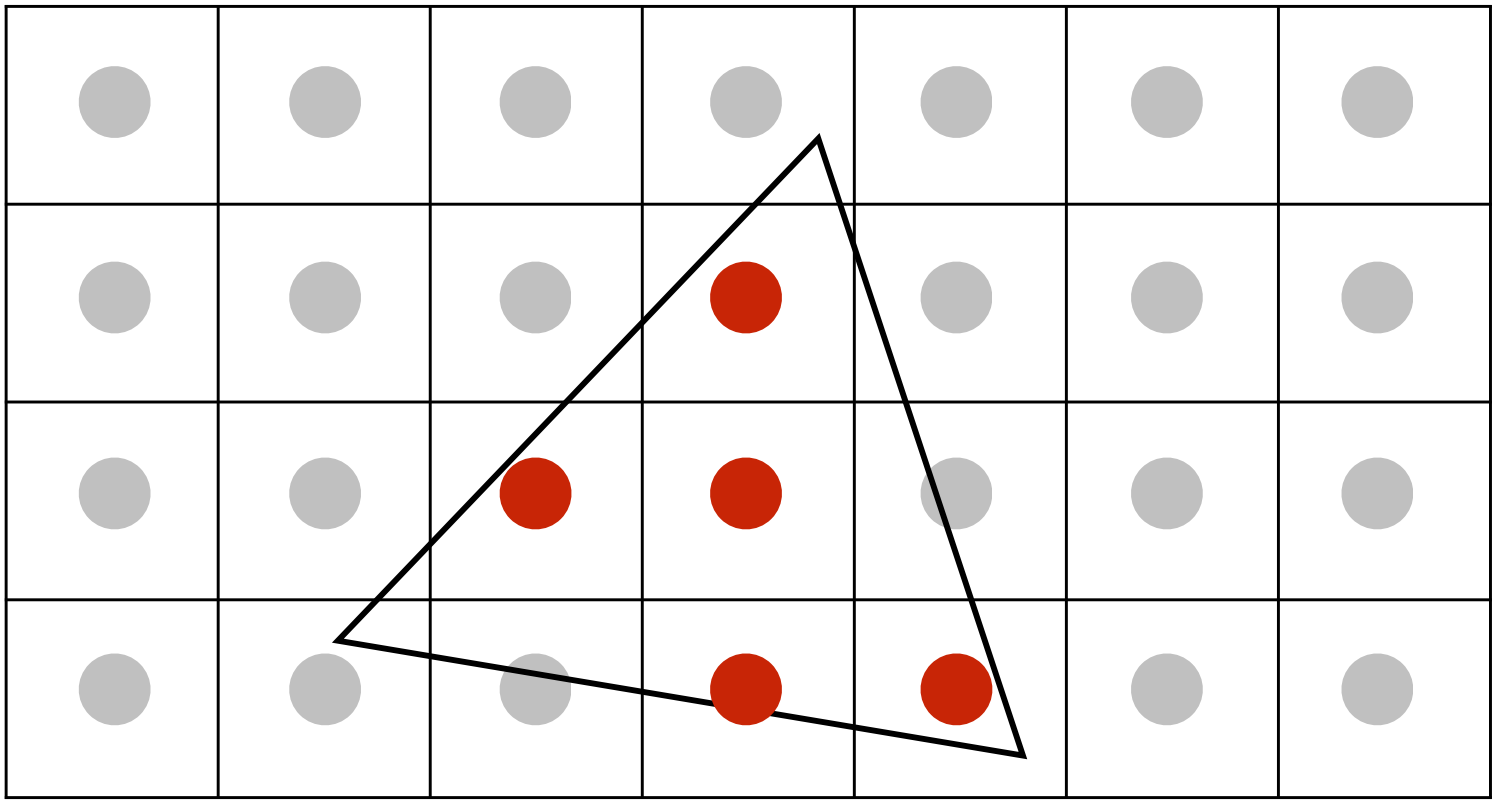
Supersampling

We can approximate the effect of the 1-pixel box filter by sampling multiple locations within a pixel and averaging their values:



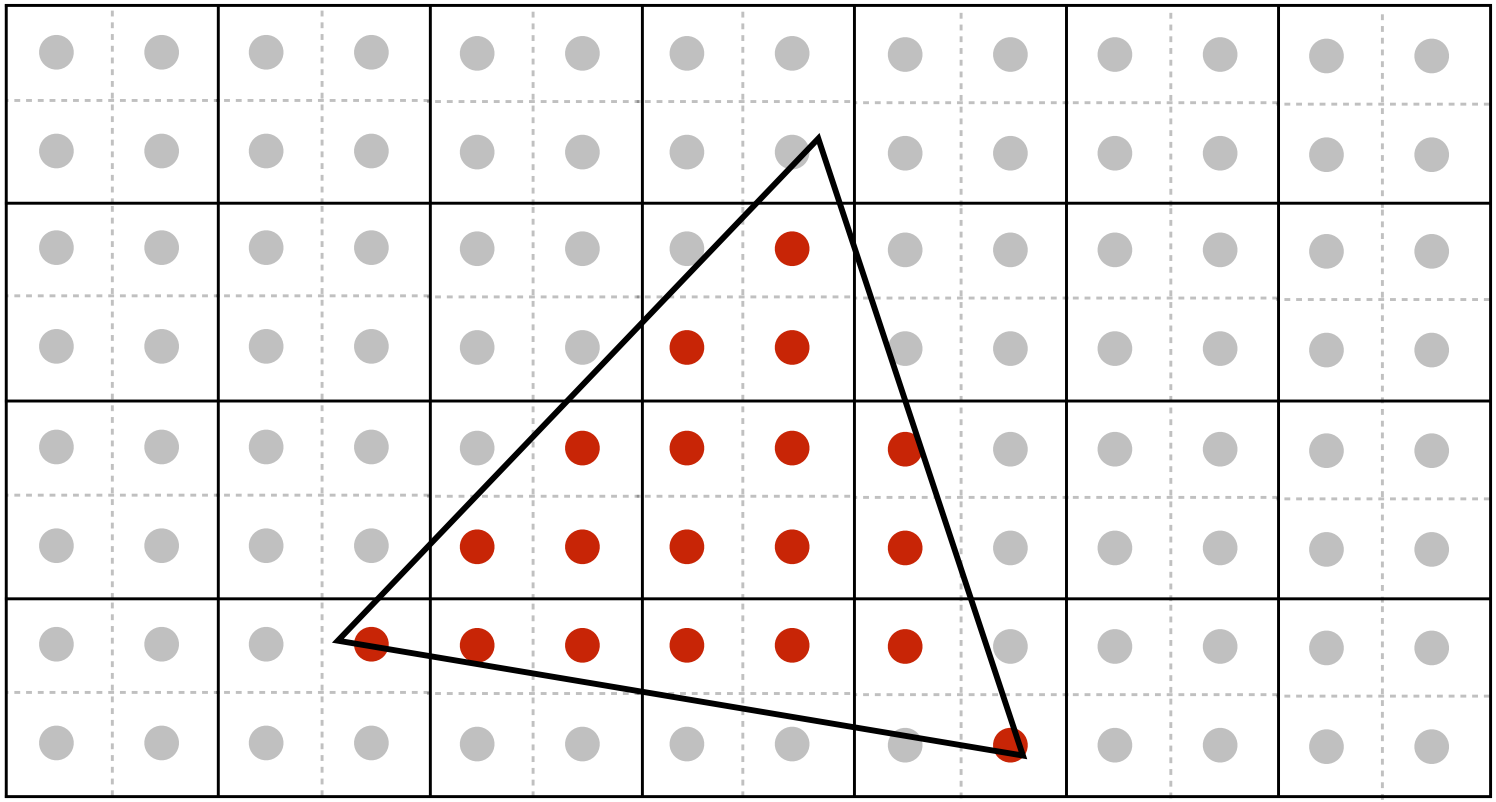
4x4 supersampling

Point Sampling: One Sample Per Pixel



Supersampling: Step 1

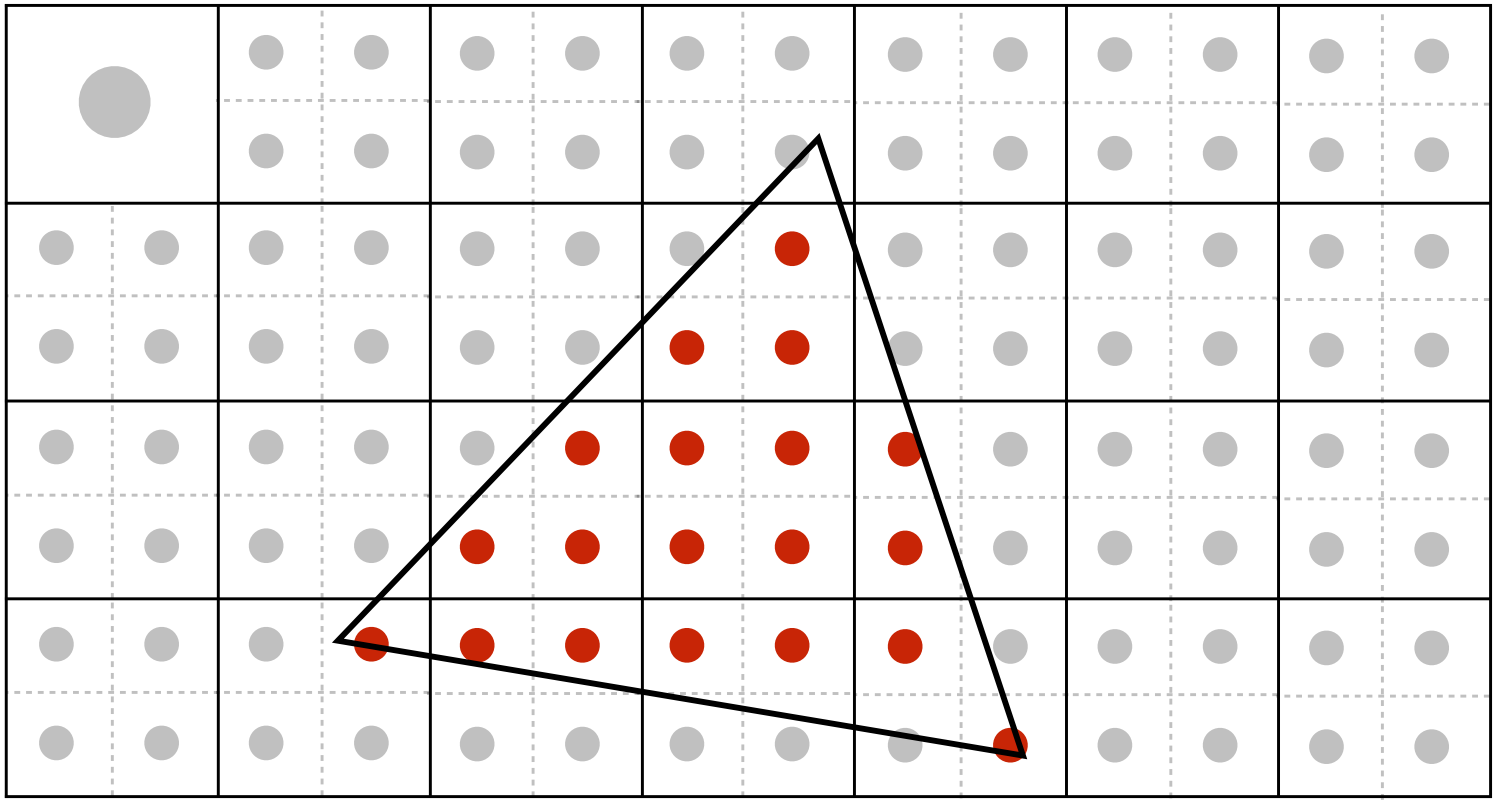
Take NxN samples in each pixel.



2x2 supersampling

Supersampling: Step 2

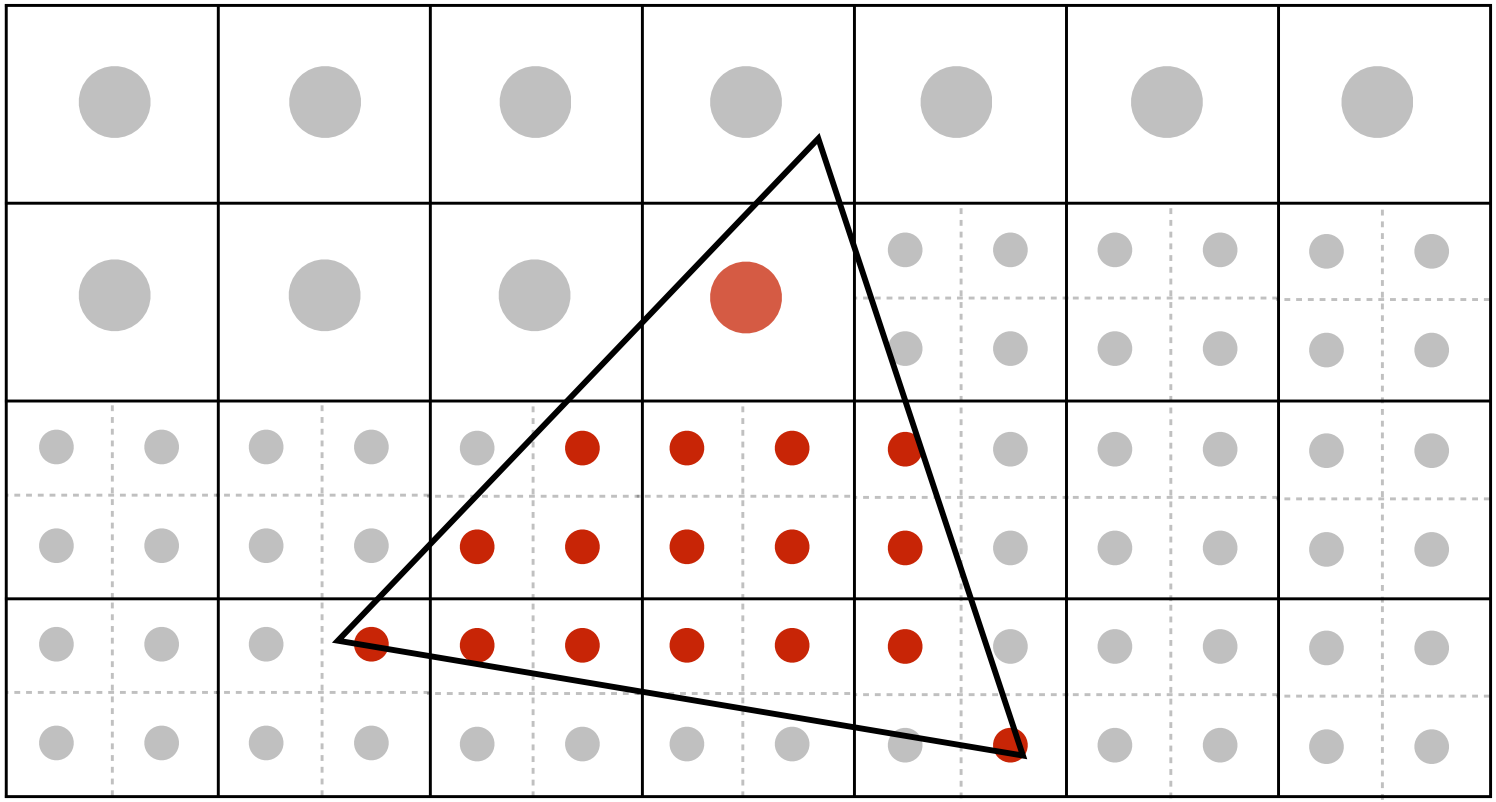
Average the NxN samples “inside” each pixel.



Averaging down

Supersampling: Step 2

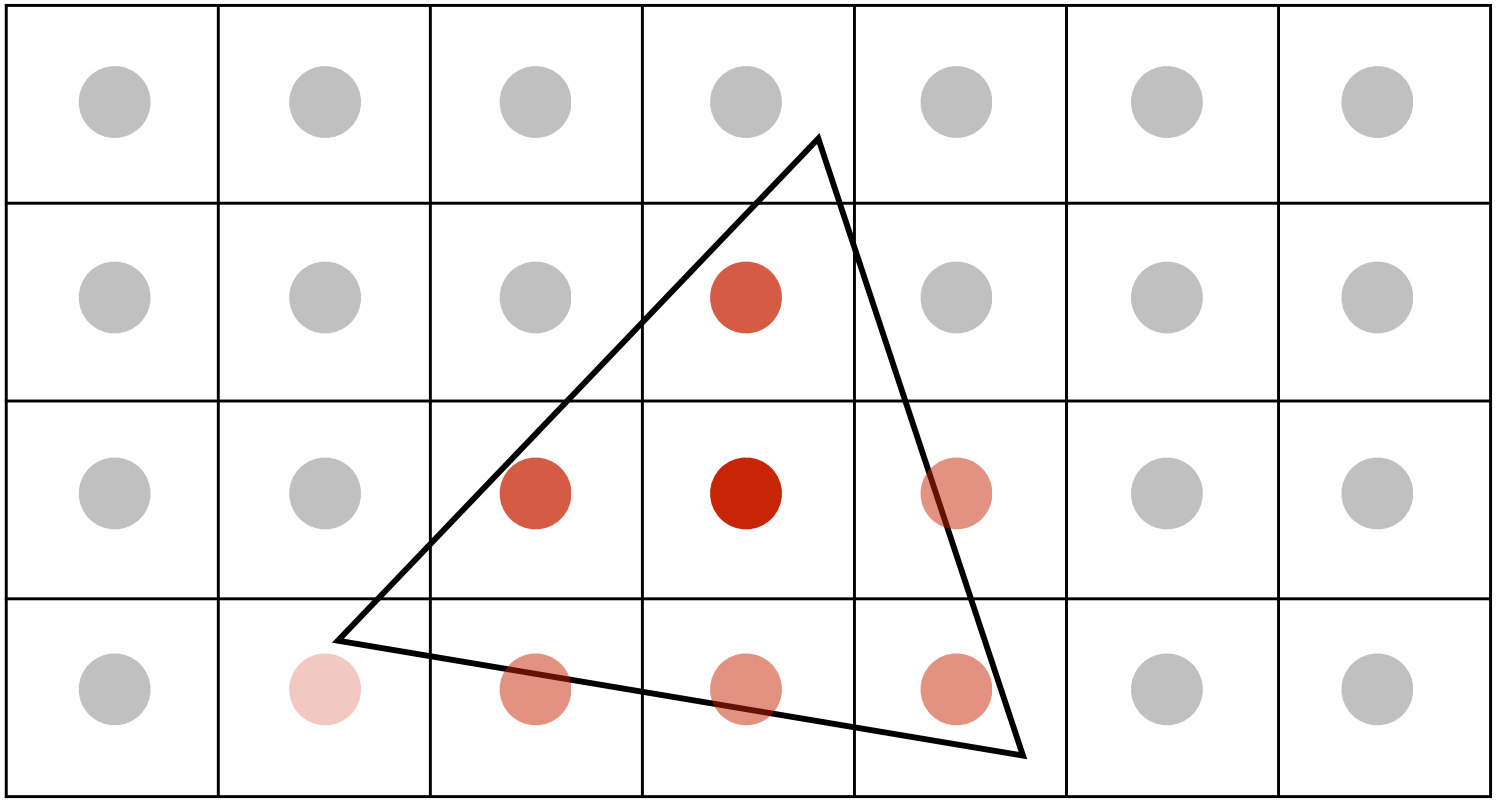
Average the NxN samples “inside” each pixel.



Averaging down

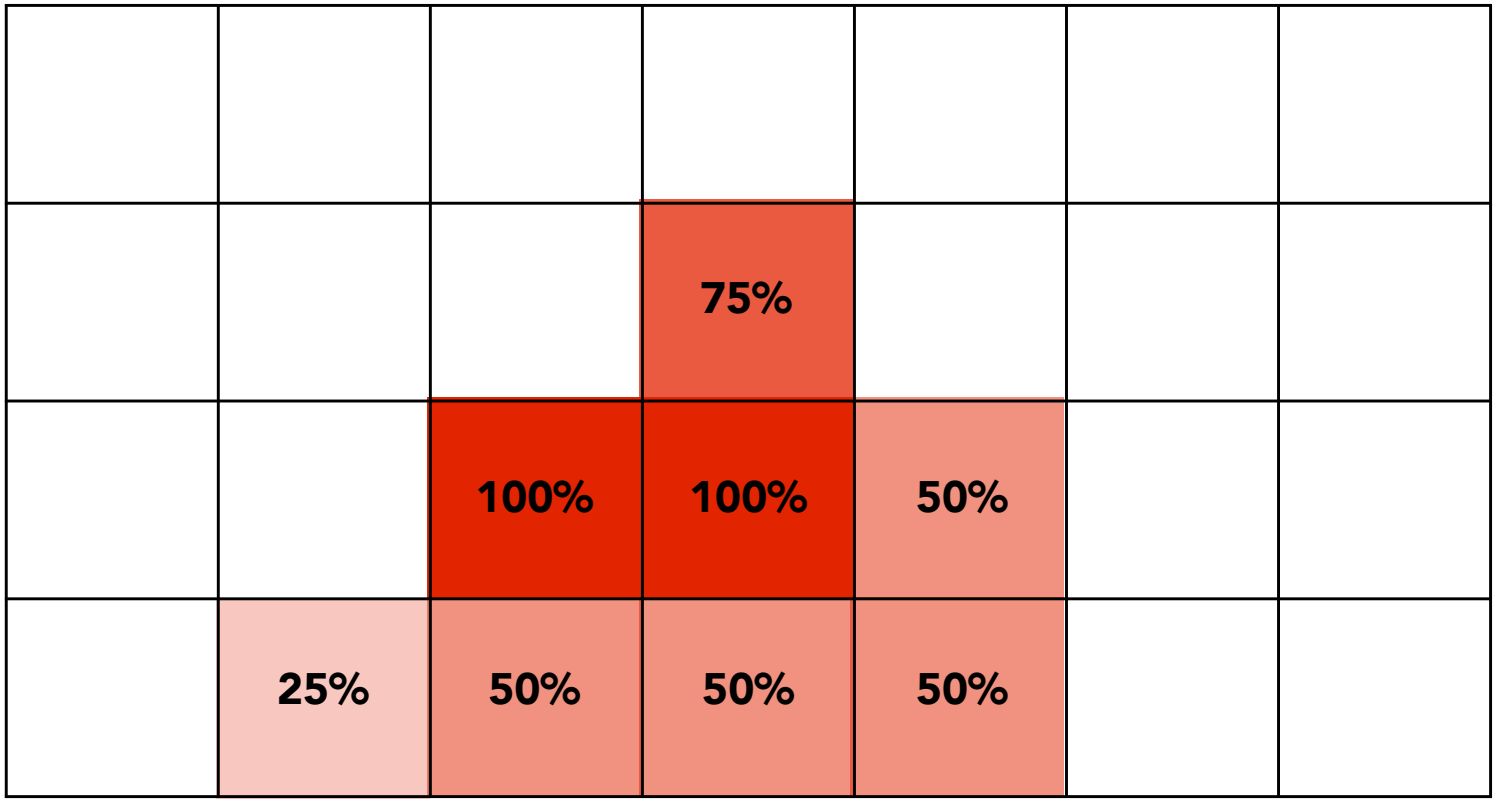
Supersampling: Step 2

Average the NxN samples “inside” each pixel.

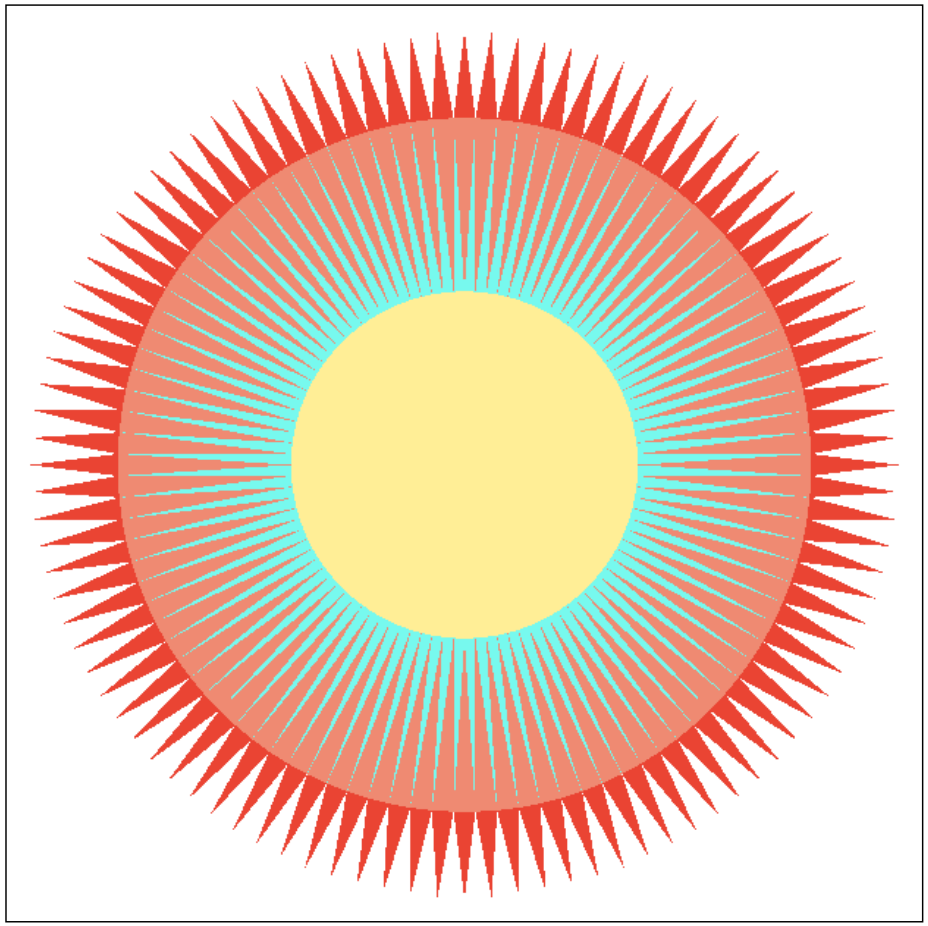


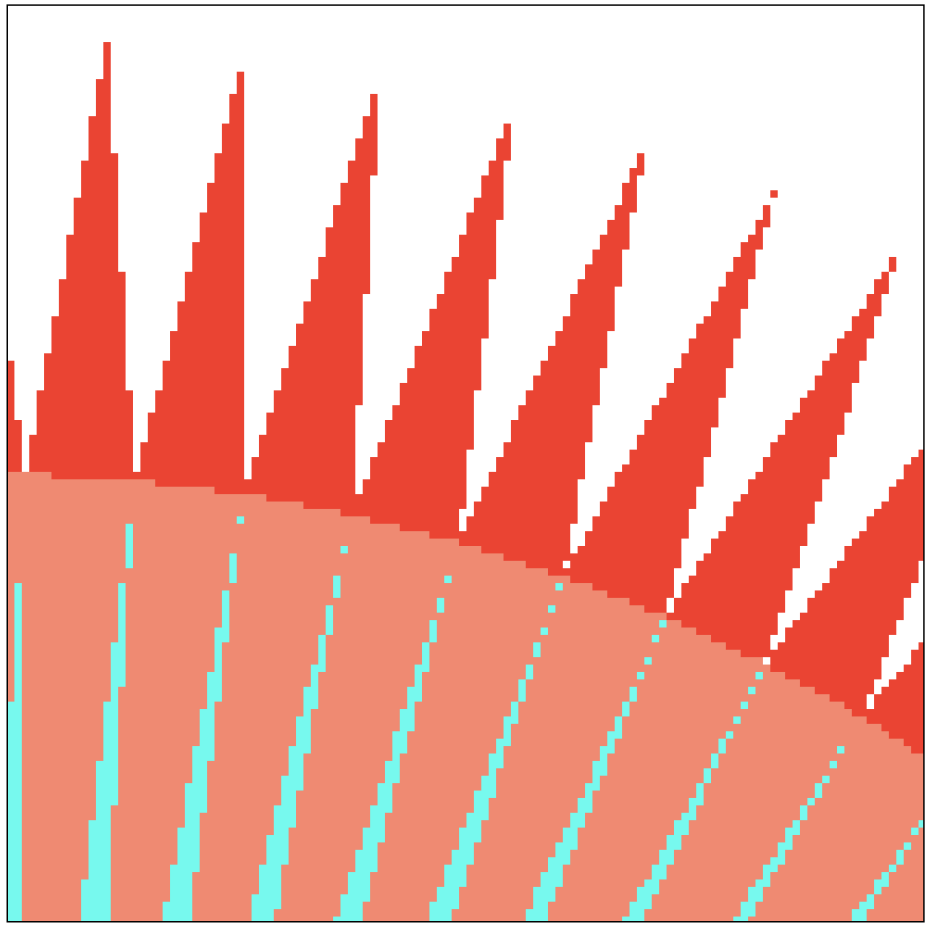
Supersampling: Result

This is the corresponding signal emitted by the display



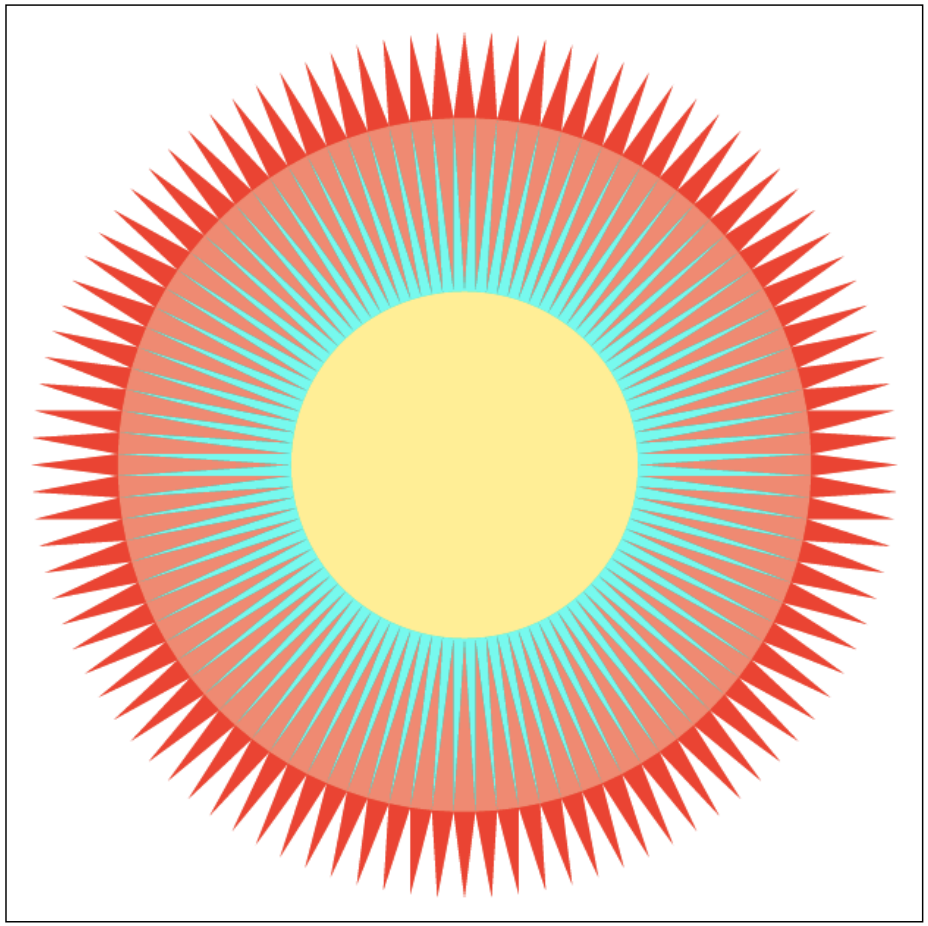
Point Sampling

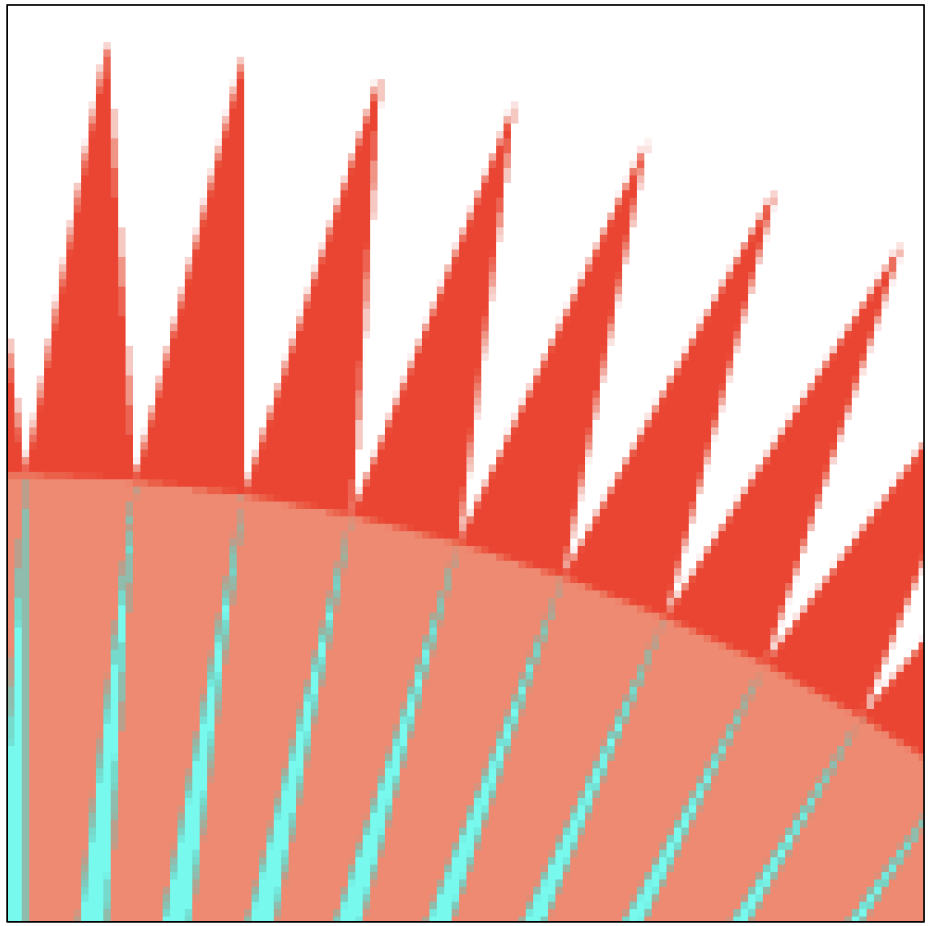




One sample per pixel

4x4 Supersampling + Box filter





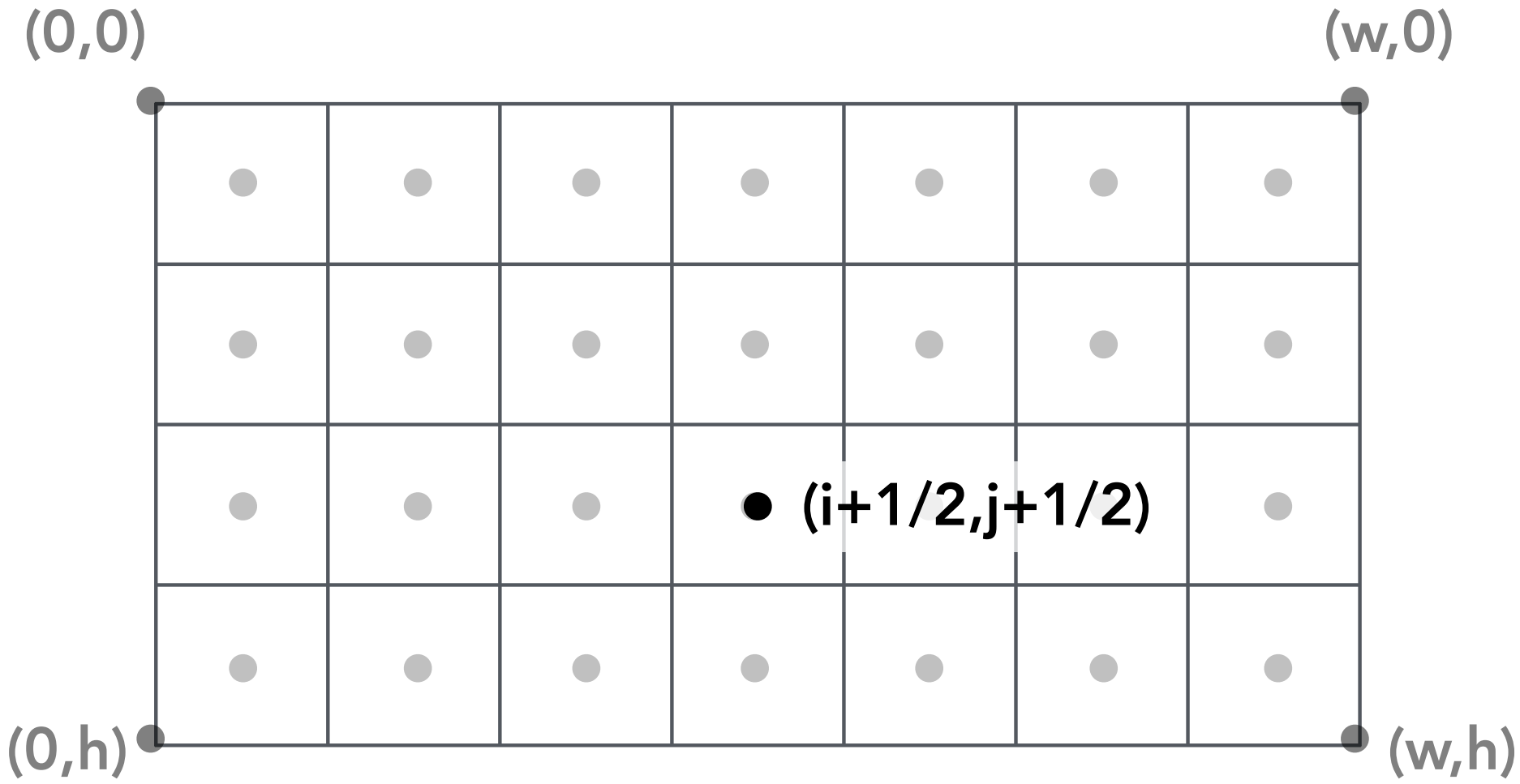
Pixel value is average of 4x4 samples per pixel

Antialiasing By Supersampling - Summary

- Antialiasing = remove frequencies above Nyquist before sampling
- We can attenuate these frequencies quite well with a 1-pixel box filter (convolution)
- We approximated the 1-pixel box sampling by supersampling and averaging
- Simple, good idea - high image quality, but costly
- May feel “right”, but can get even higher quality!

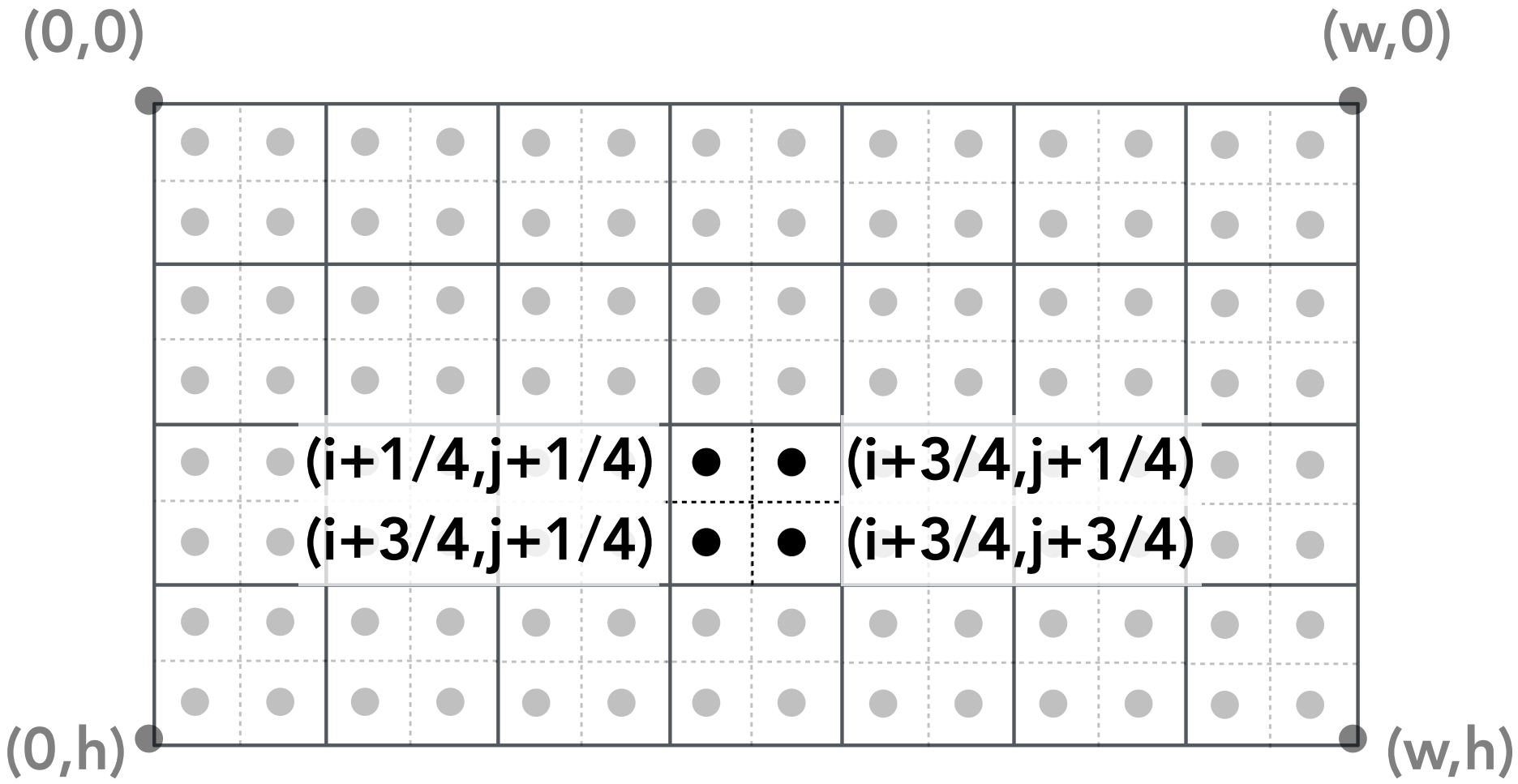
Supersampling Implementation Tips

Tip 1: Sample Locations



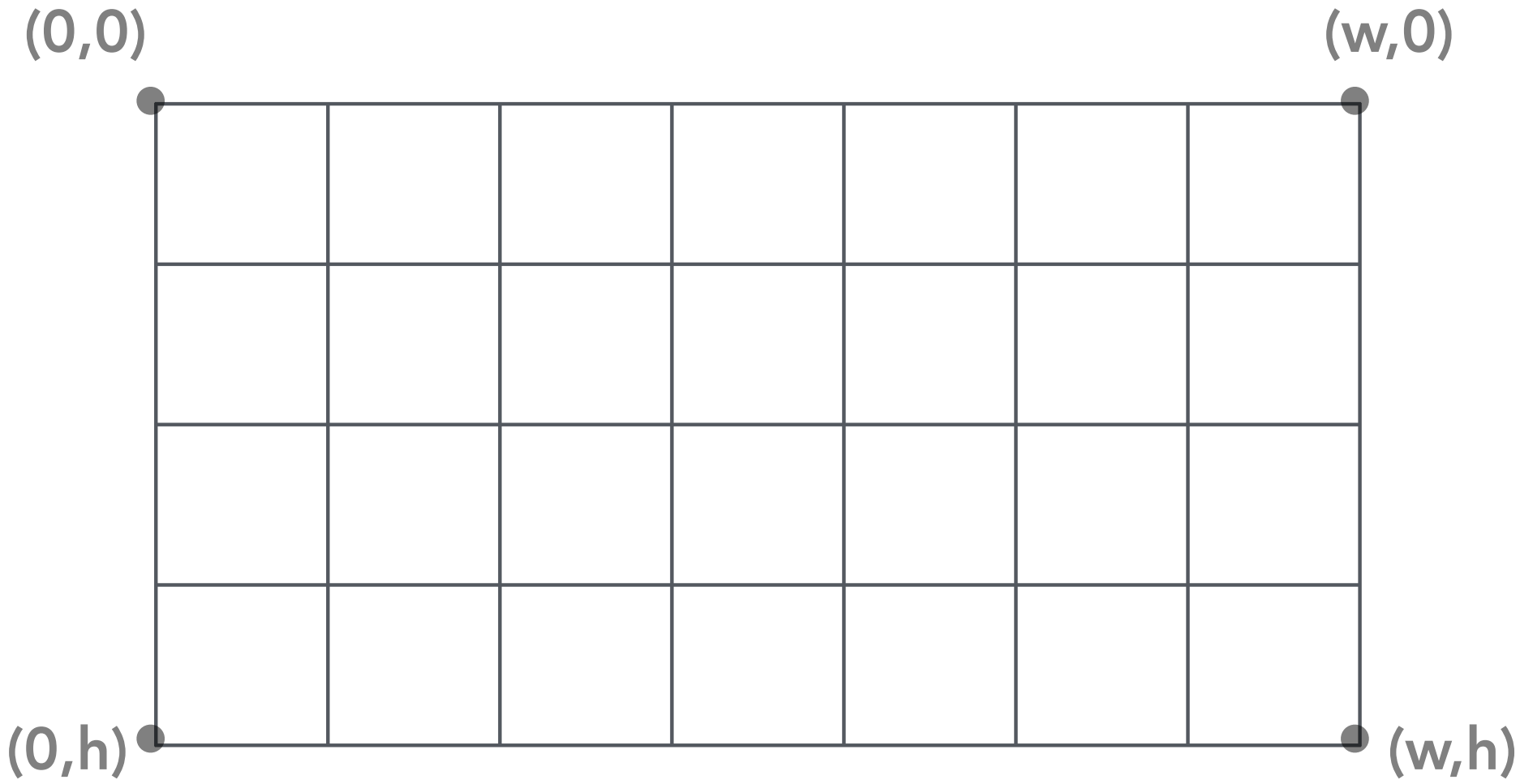
Regular sampling: sample location for pixel (i,j)

Tip 1: Sample Locations



2x2 supersampling: locations for pixel (i,j)

Tip 1: Sample Locations



Sample locations for NxN supersampling?

Tip 2: Supersampling Multiple Triangles

So far, we rasterized only a single triangle:

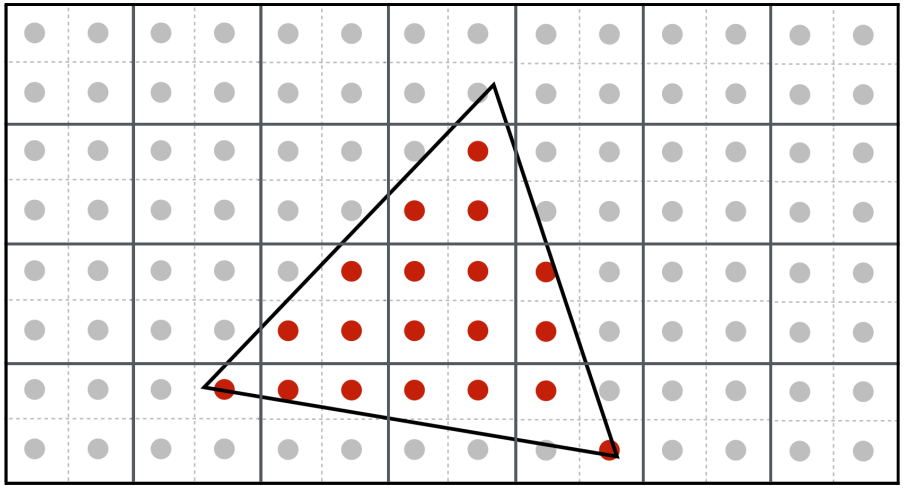
- Supersample
- Then average down

How should this change when we rasterize N triangles in the same image?

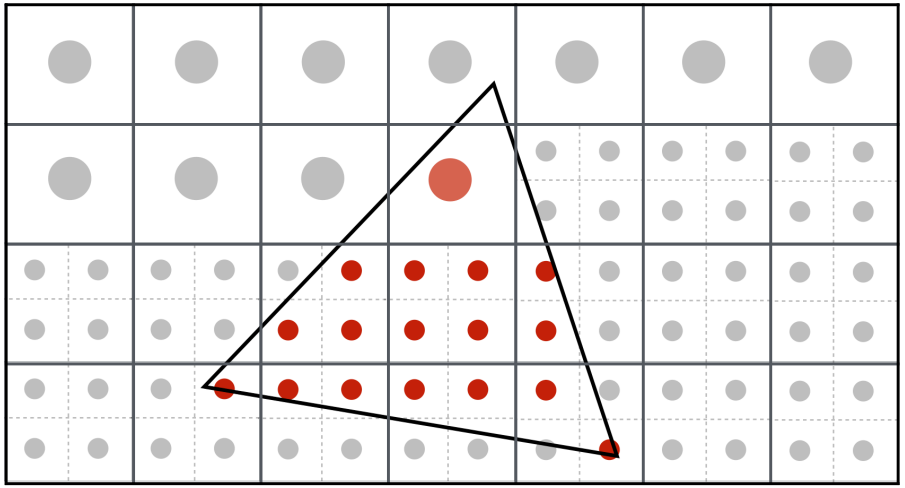
- Supersample and average down each triangle, one by one?
- Or supersample all N triangles onto a high-res grid, then average down?

What are the algorithmic implications?

- E.g. what is the minimum memory needed?



Supersample



Average Down

Antialiasing (AA) Today

- SSAA (Supersampling AA)
  - Exact implementation: MSAA (Multi-Sampling AA)
  - Approximate implementation: FXAA (Fast Approximate AA)
  - TAA (Temporal AA) (当前帧和历史帧进行加权融合)
  - DLSS (Deep Learning Super Sampling)
- Essentially still “not enough samples” problem

Things to Remember

Signal processing key concepts:

- Frequency domain vs spatial domain
- Filters in the frequency domain scale frequencies
- Filters in the sampling domain = convolution

Sampling and aliasing

- Image generation involves sampling
- Nyquist frequency is half the sampling rate
- Frequencies above Nyquist appear as aliasing artifacts
- Antialiasing = filter out high frequencies before sampling
- Interpret supersampling as (approx) box pre-filter antialiasing