COL781: Computer Graphics

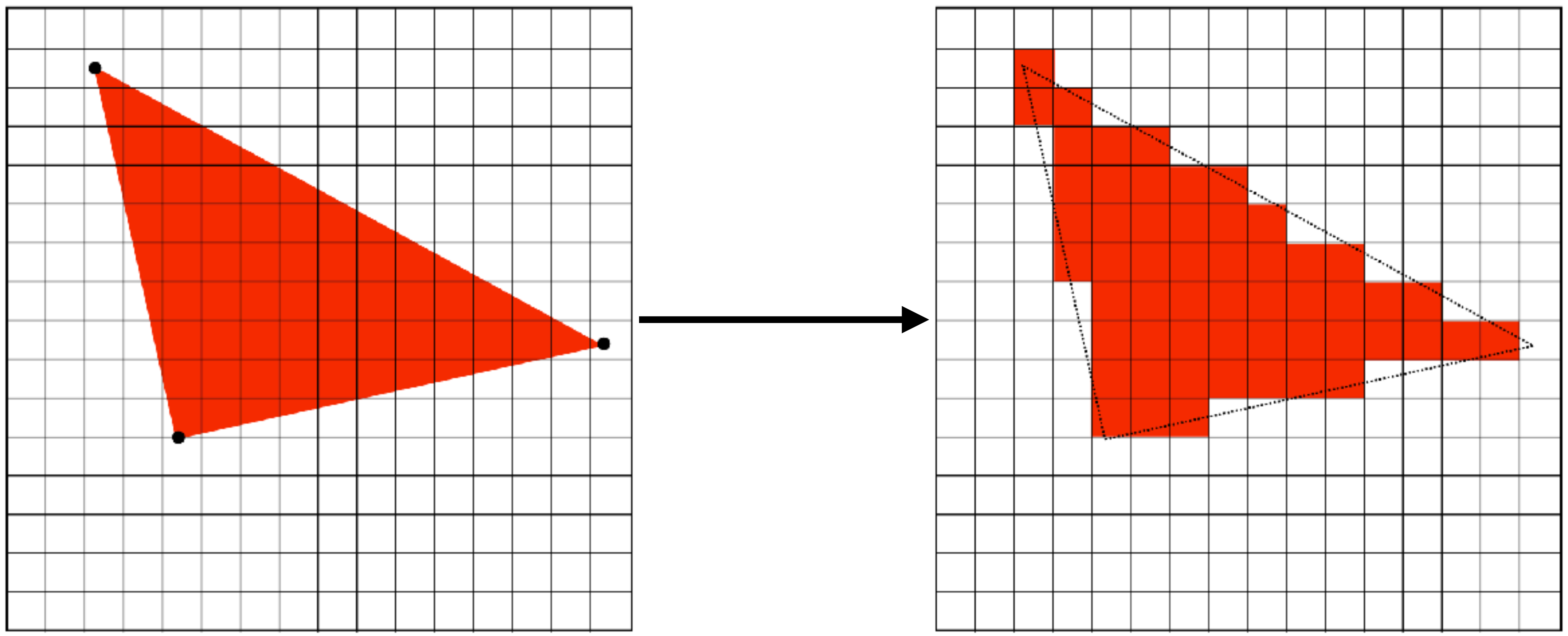
3. Sampling and Aliasing

Inigo Quilez

Last class's homework

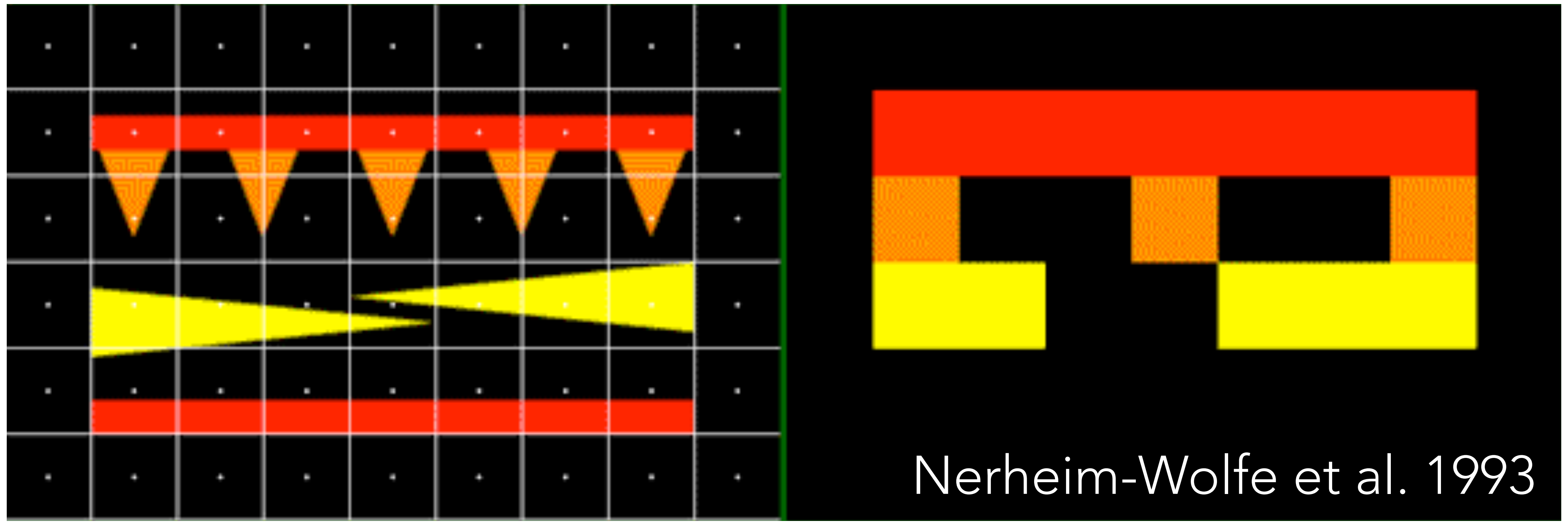
```
....................
............[]......
............[]......
..........[][][]....
........[][][][]....
........[][][][][]..
......[][][][][][]..
......[][][][][][][]
....[][][][]........
..[]................
```

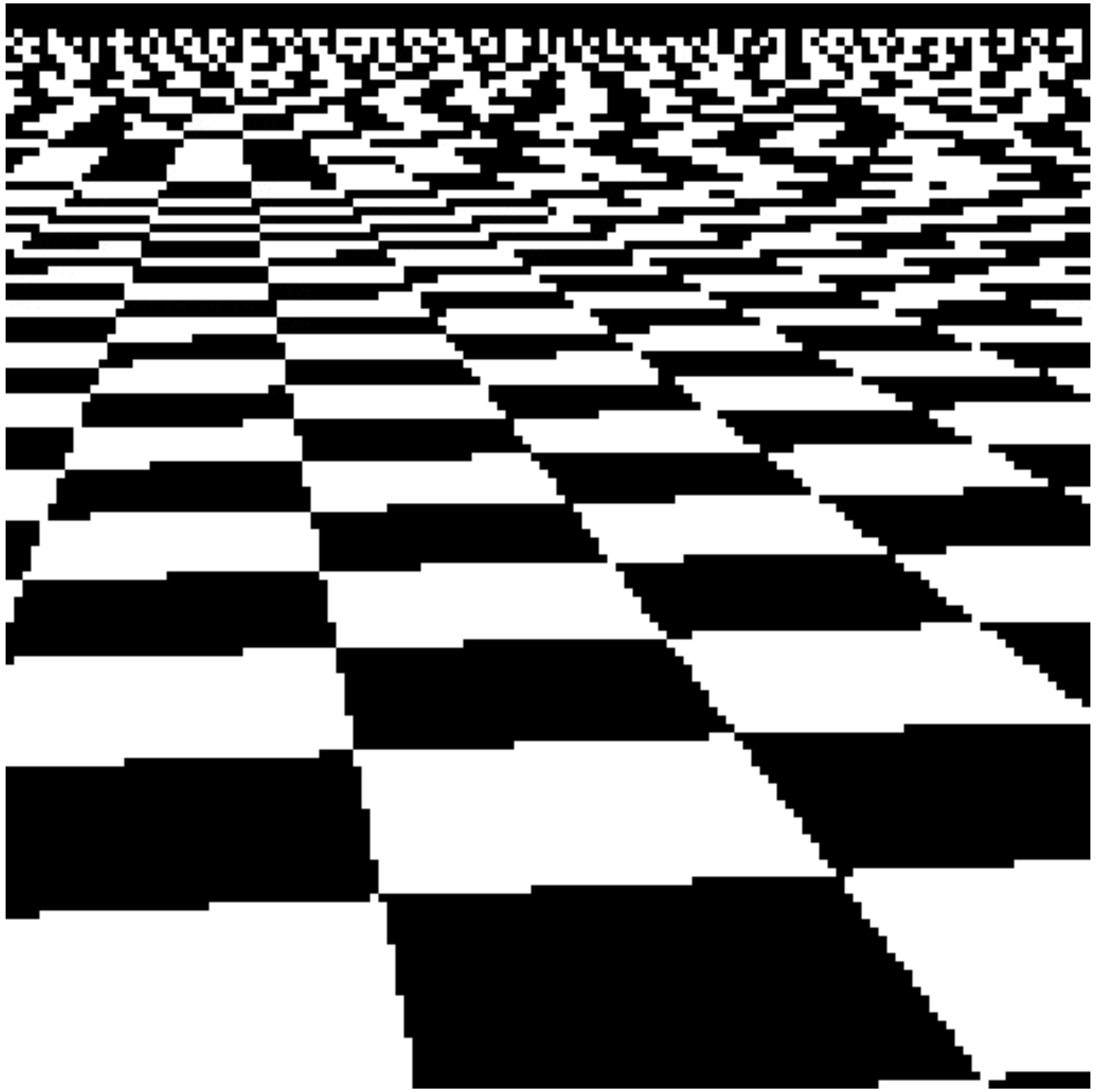
“Jaggies”



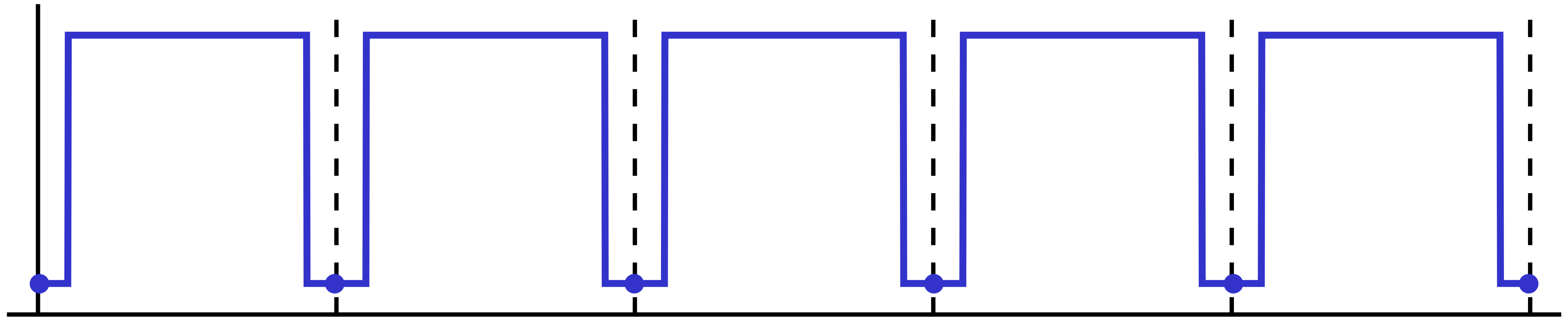
Is this a problem?

Can we do better?


Nerheim-Wolfe et al. 1993

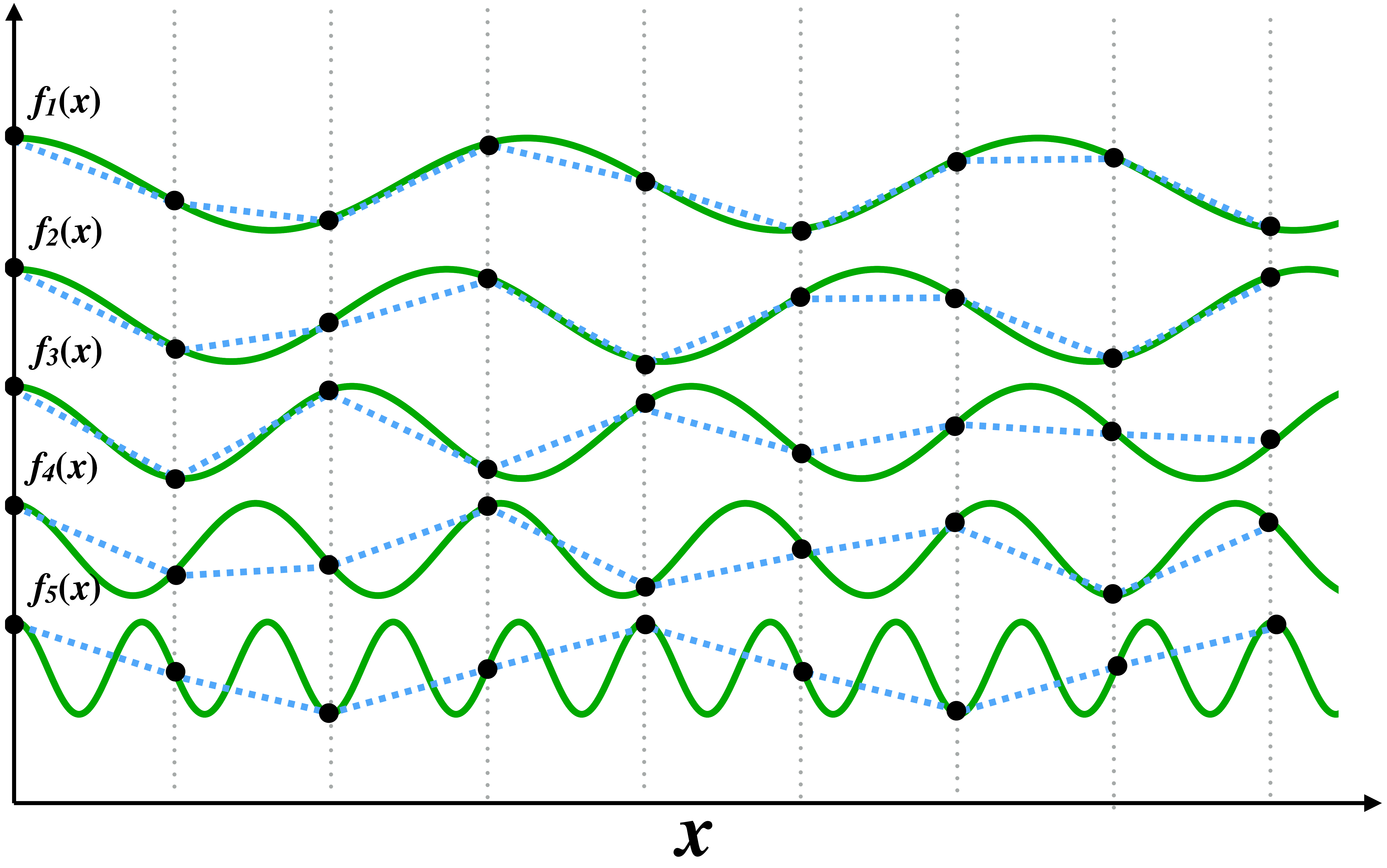


Chris Fay



Jonathan Valvano

Aliasing

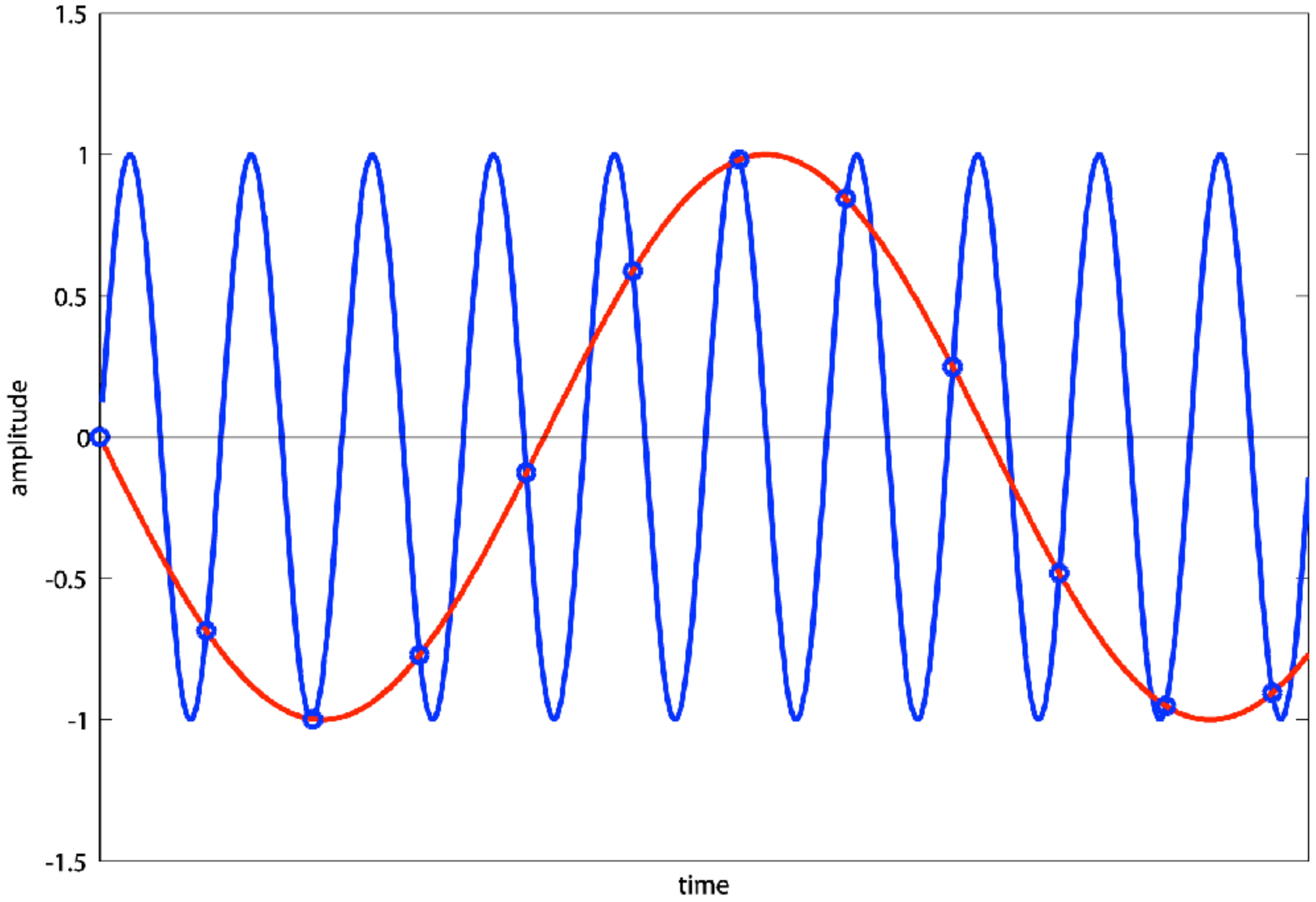


Low-frequency sinusoids can be reconstructed accurately

High-frequency sinusoids incorrectly appear as low frequencies!

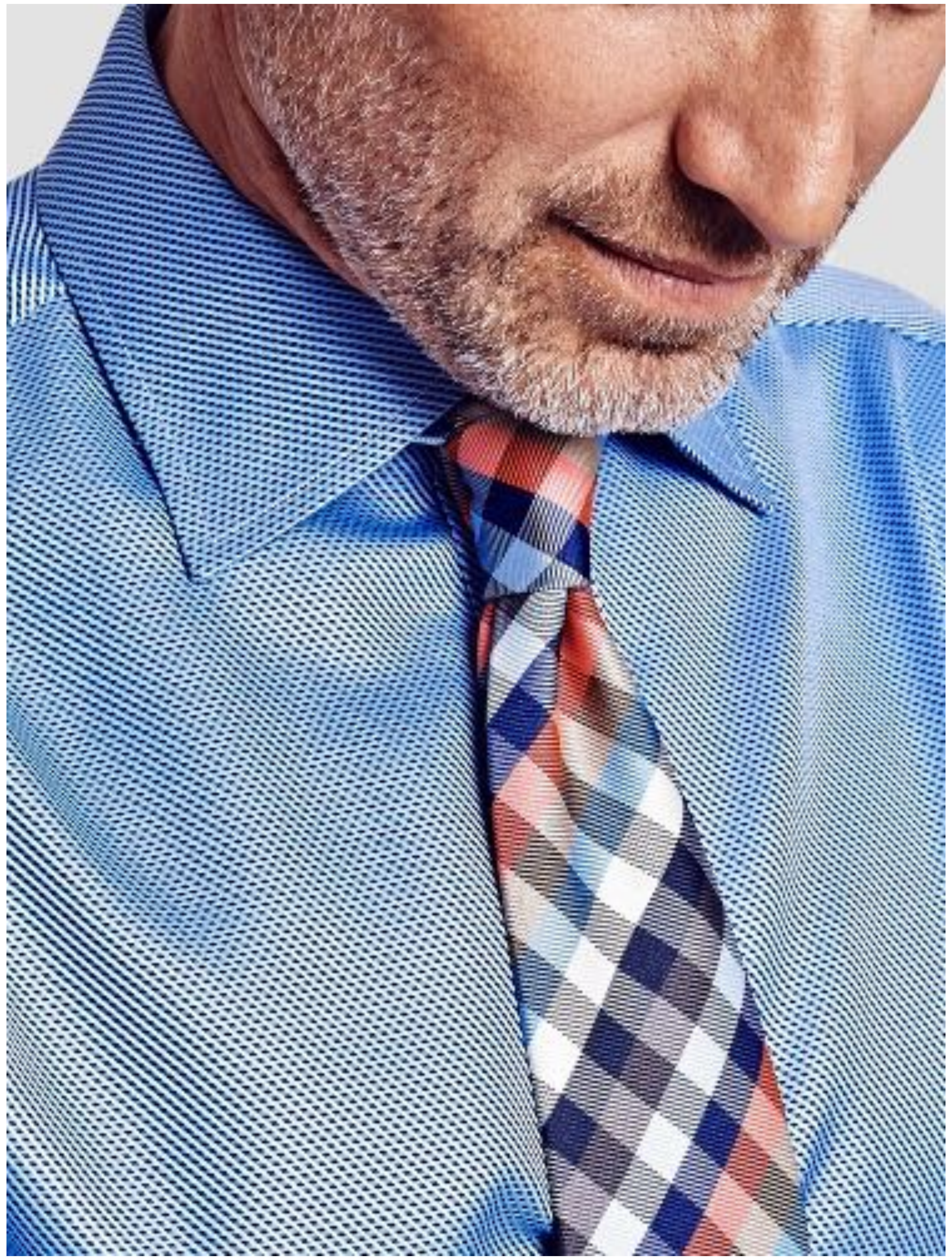
Alias = false name

Aliasing



A sinusoid with frequency higher than the sampling rate is indistinguishable from a low-frequency sinusoid!

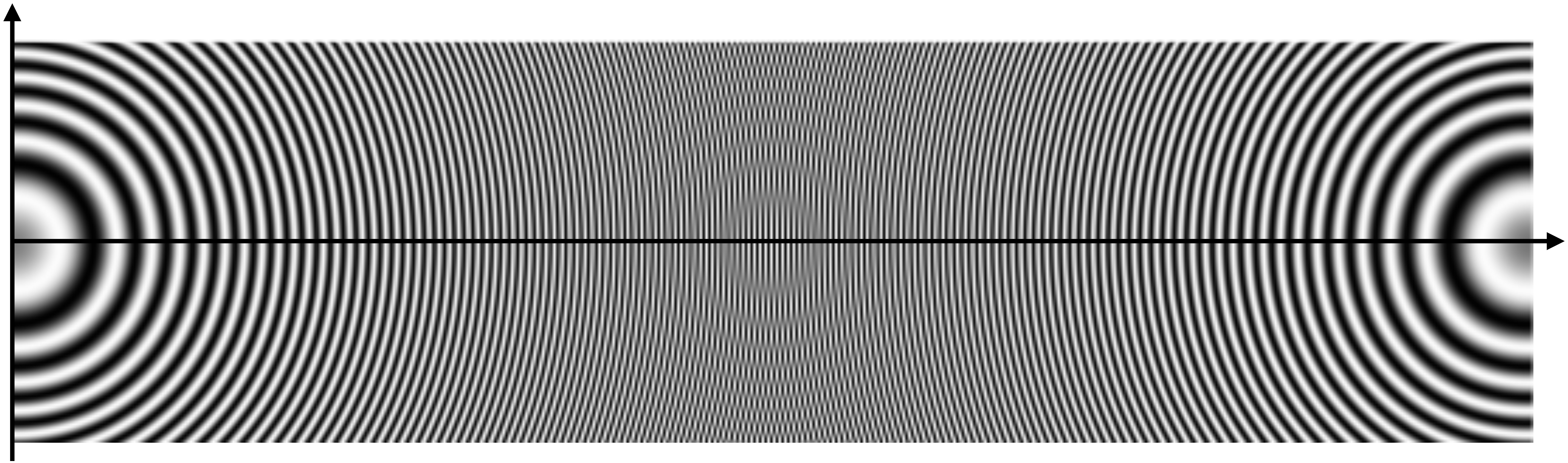
Aliasing in images

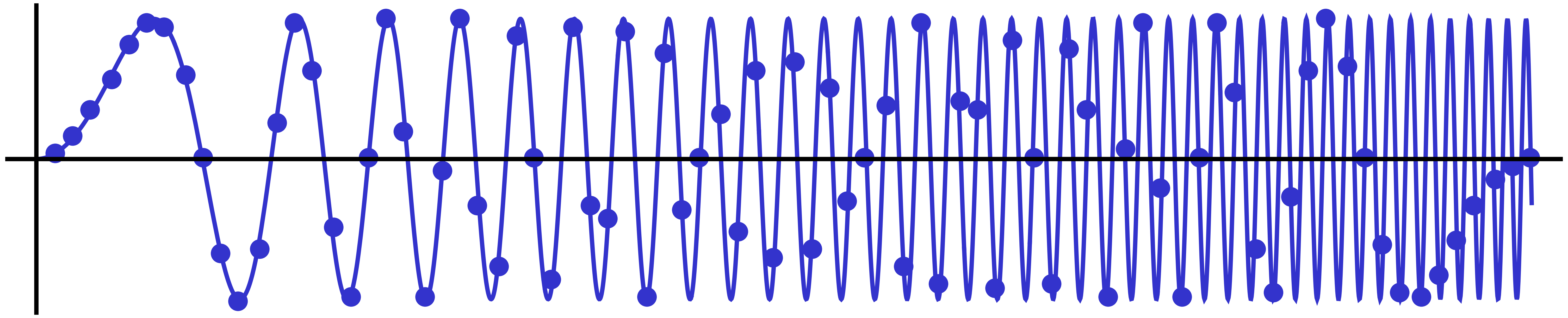


Original image

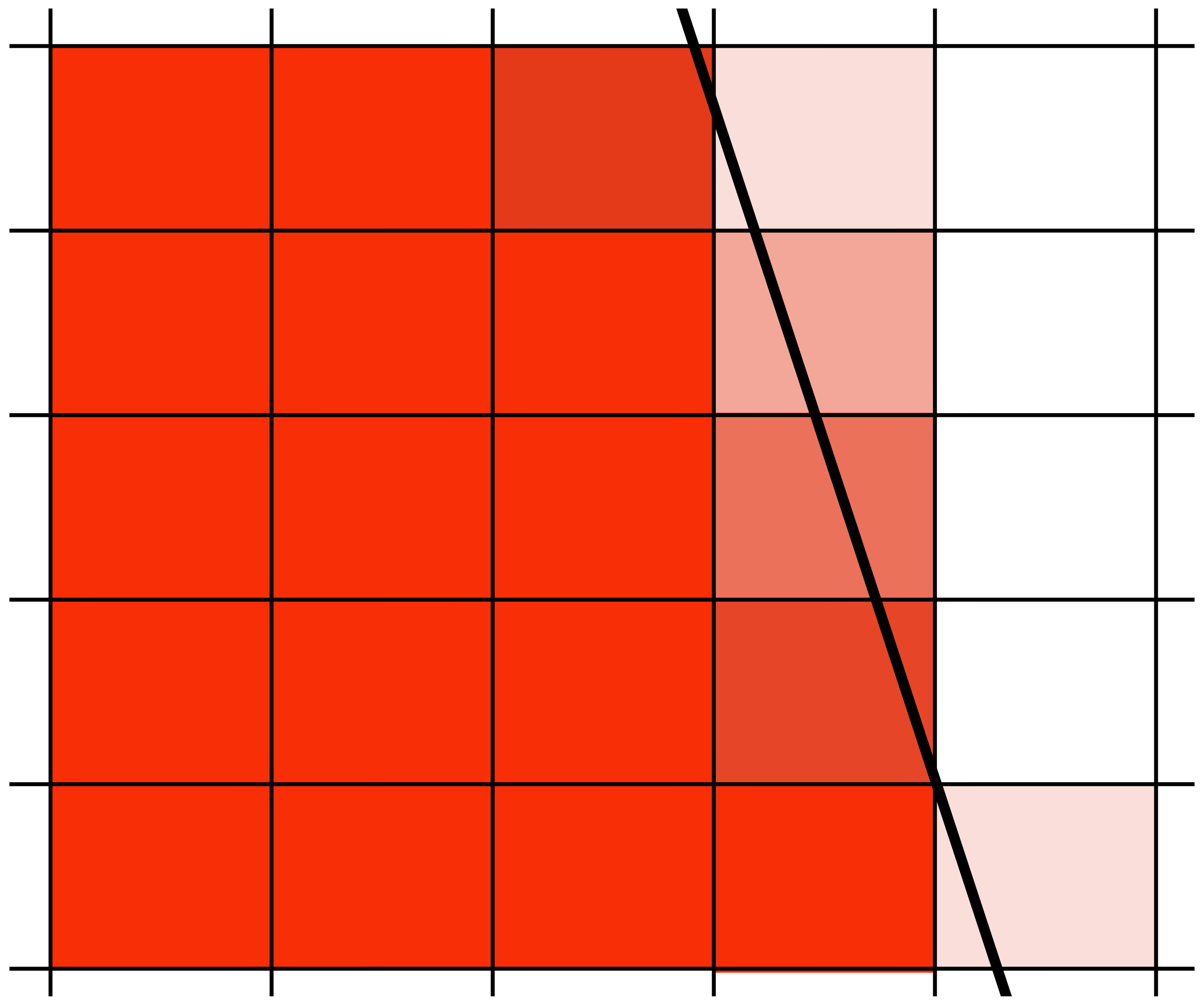
Dropping half the rows and columns

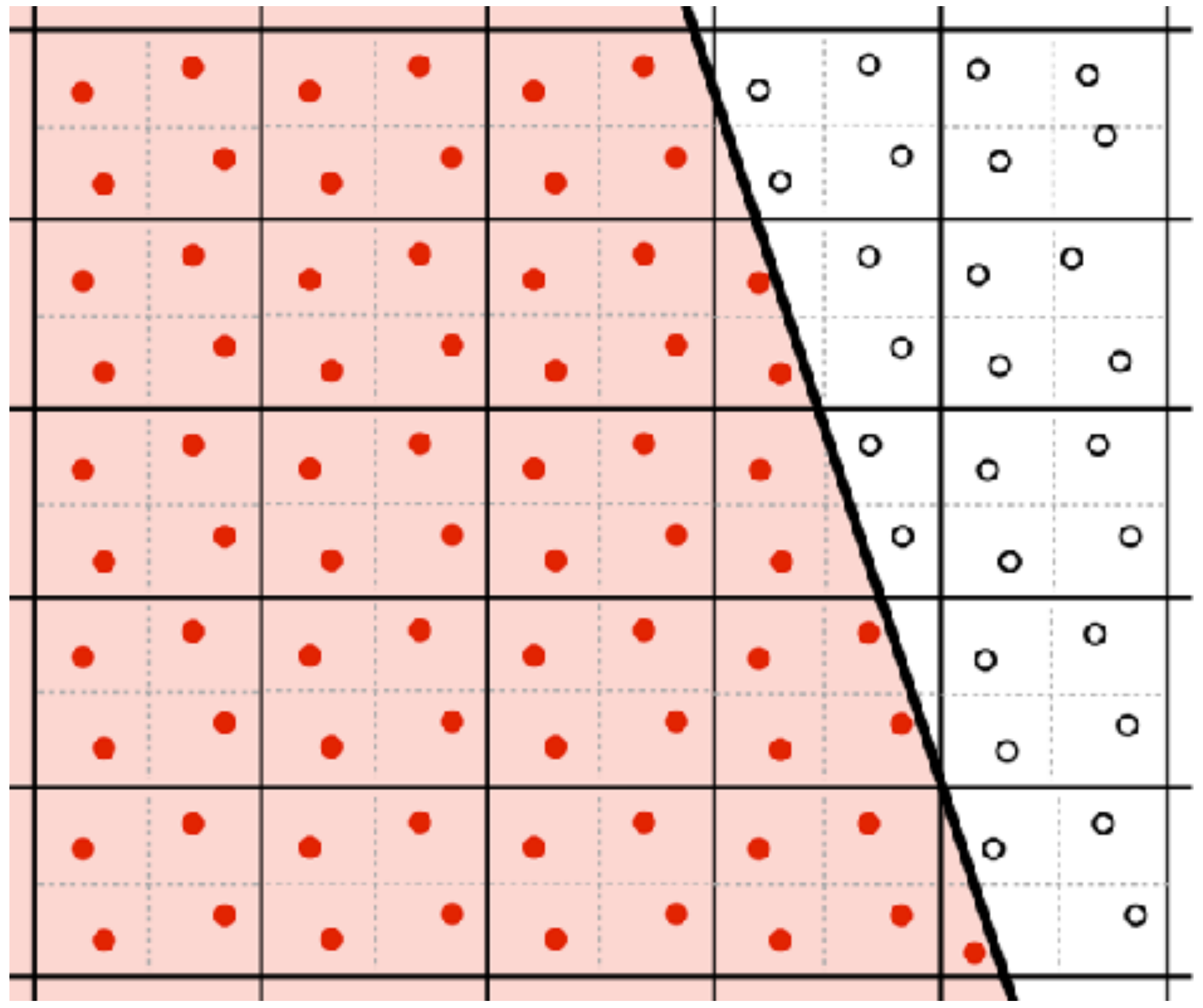
Zone plate: $f(x,y) = \sin(x^2 + y^2)$





Anti-aliasing by averaging

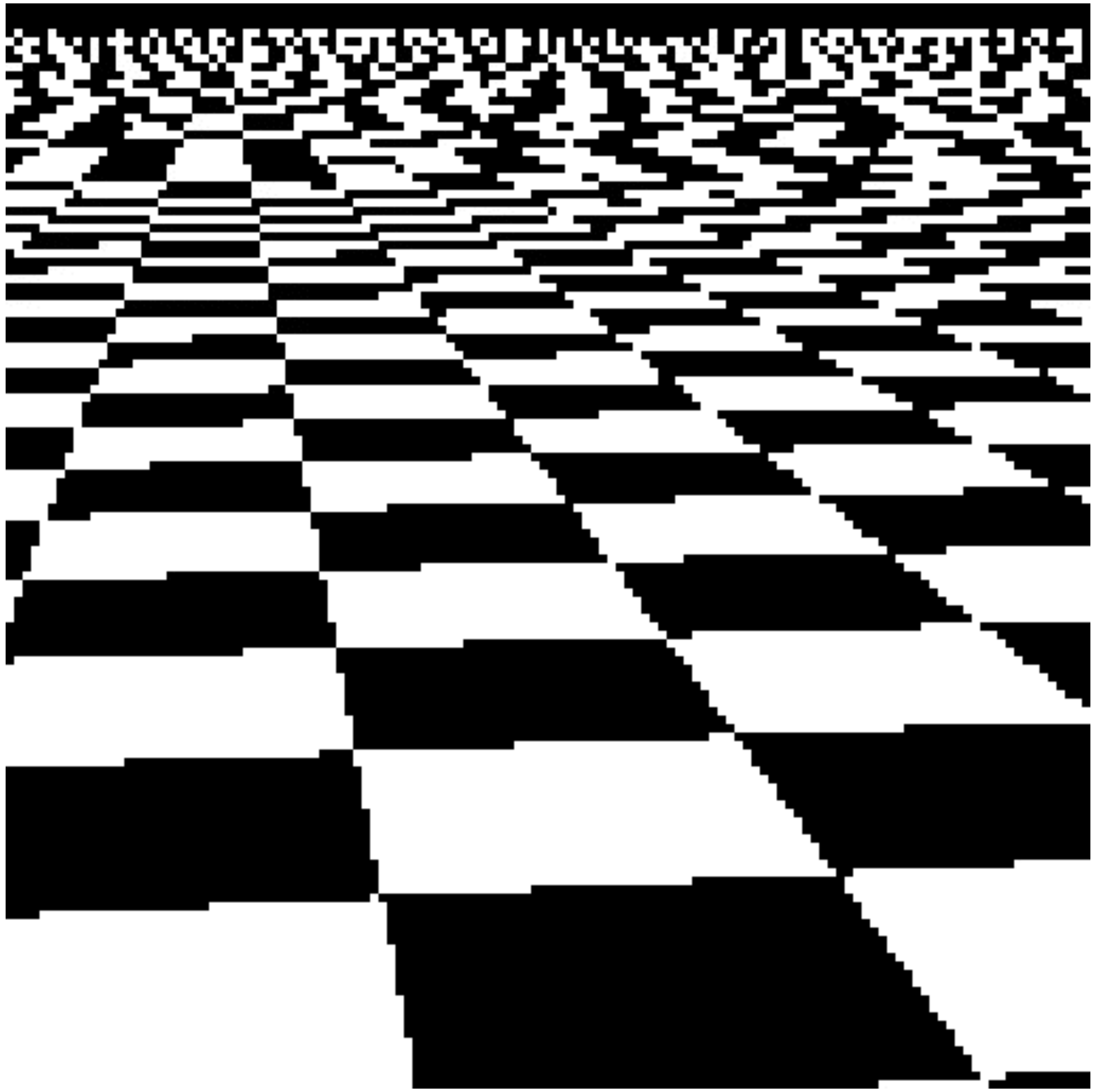




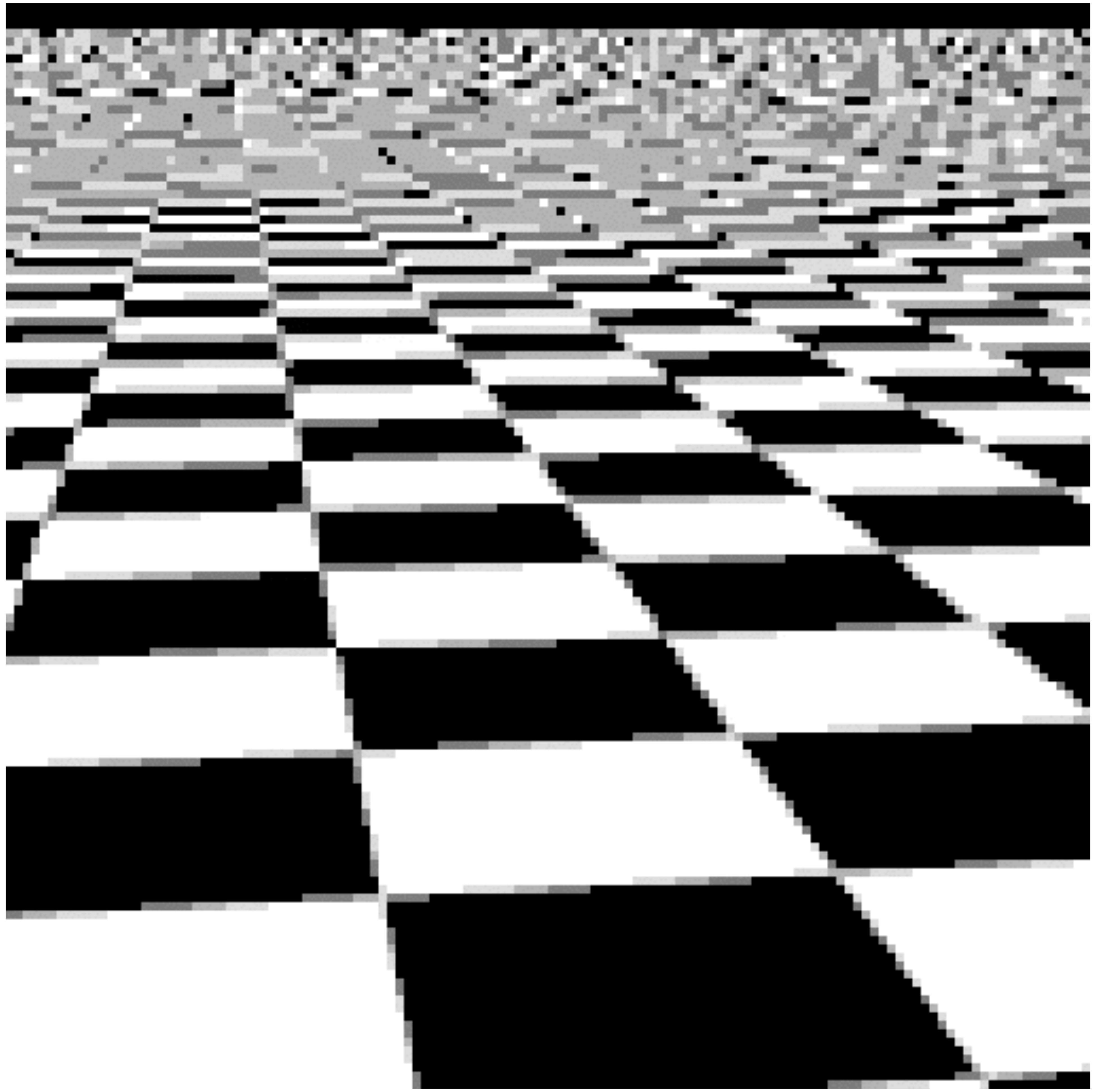
Intuitive idea: the colour of a pixel should be the average colour in the square.

- Computed exactly if you can
- Or approximated by averaging multiple samples

Supersampling

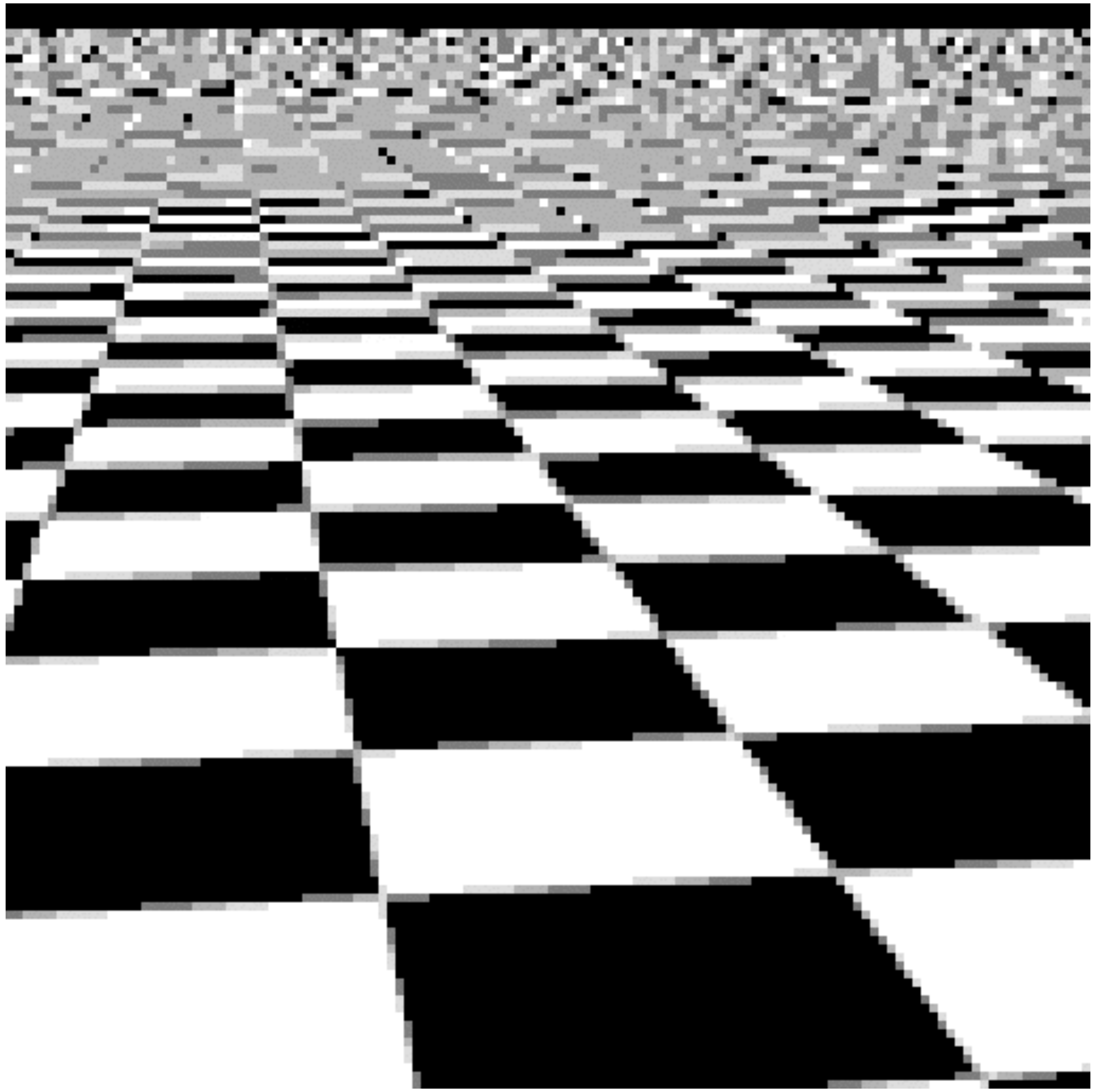


1 sample per pixel

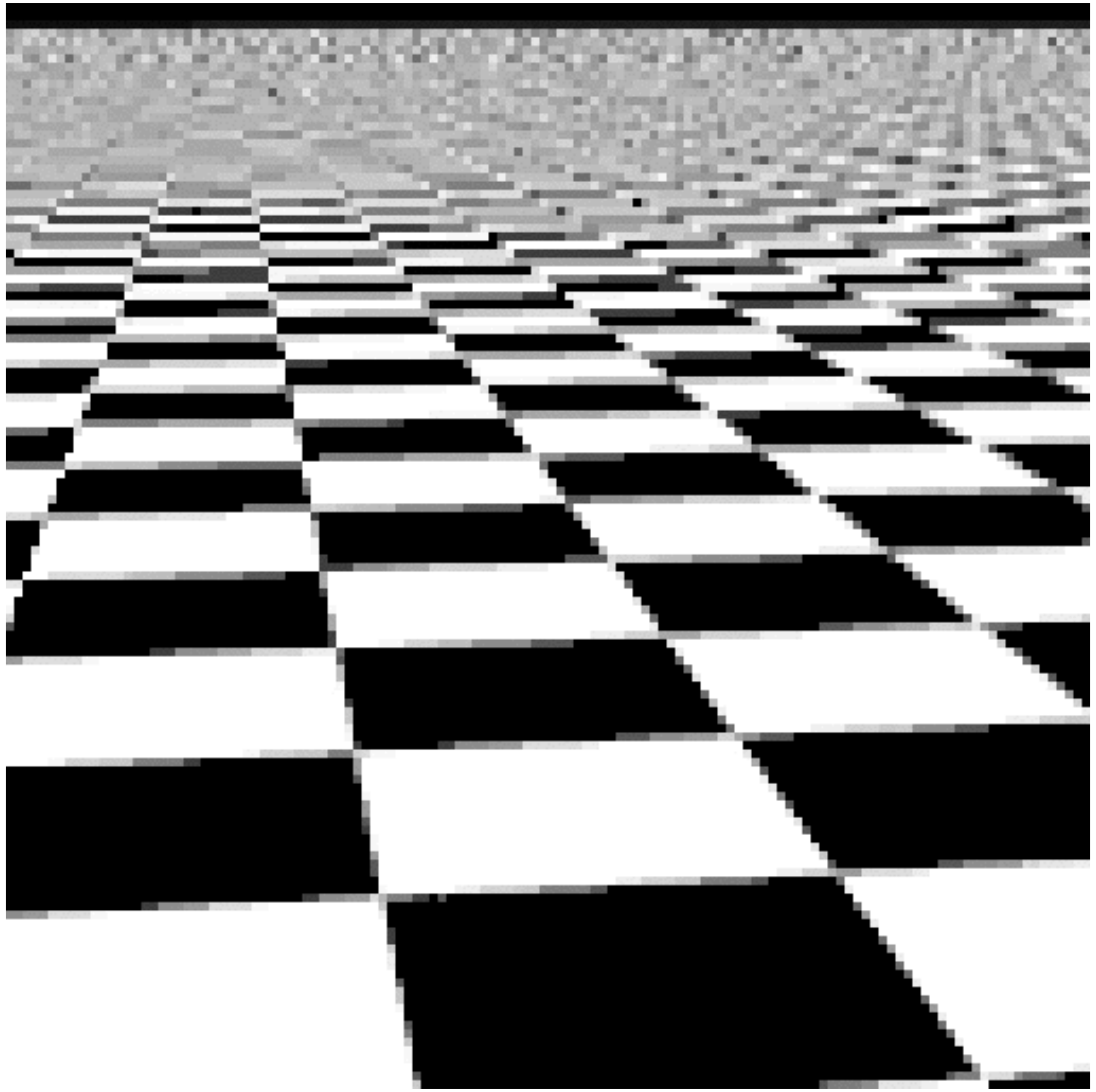


2×2 supersampling

Supersampling

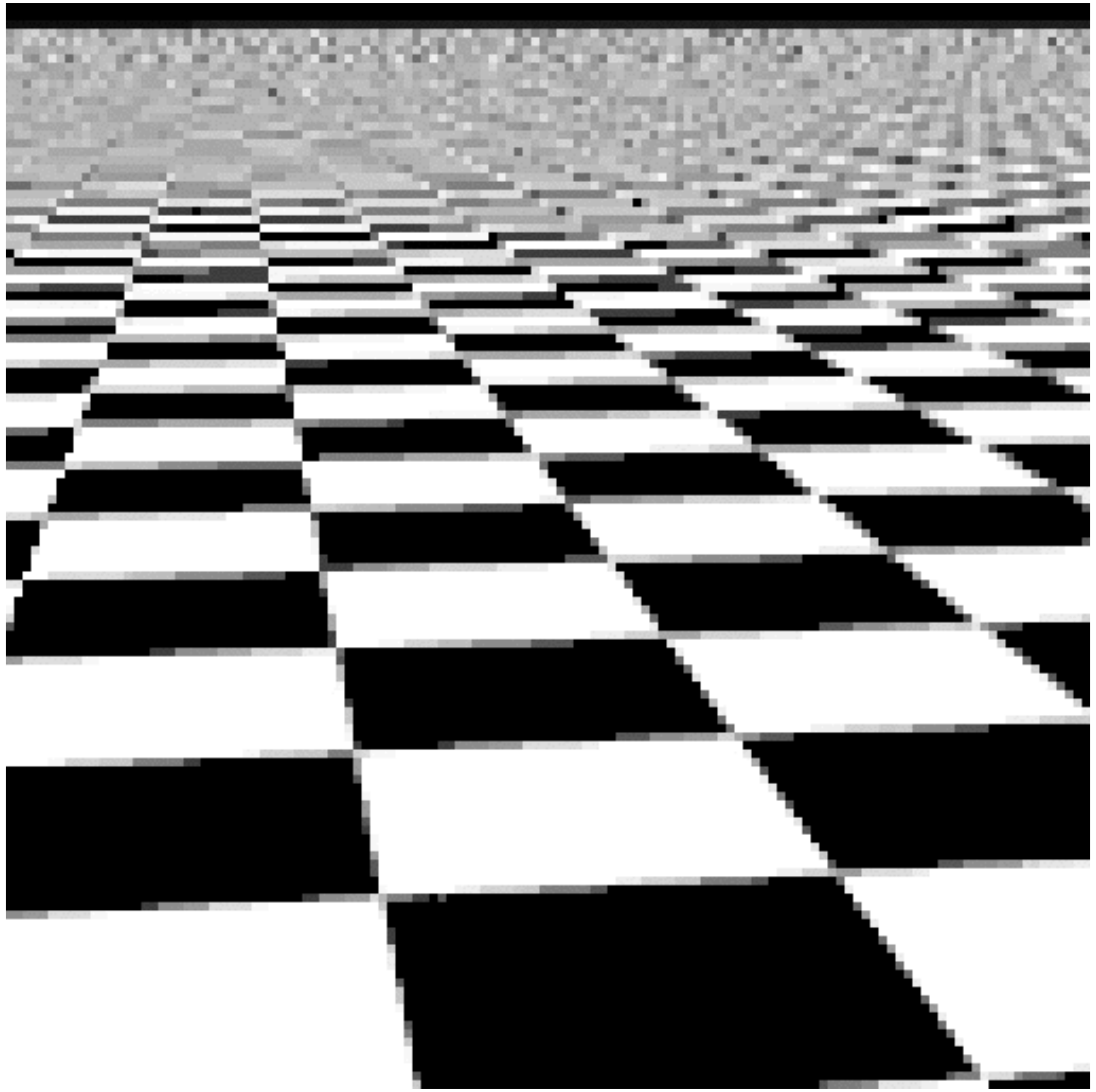


2×2 supersampling

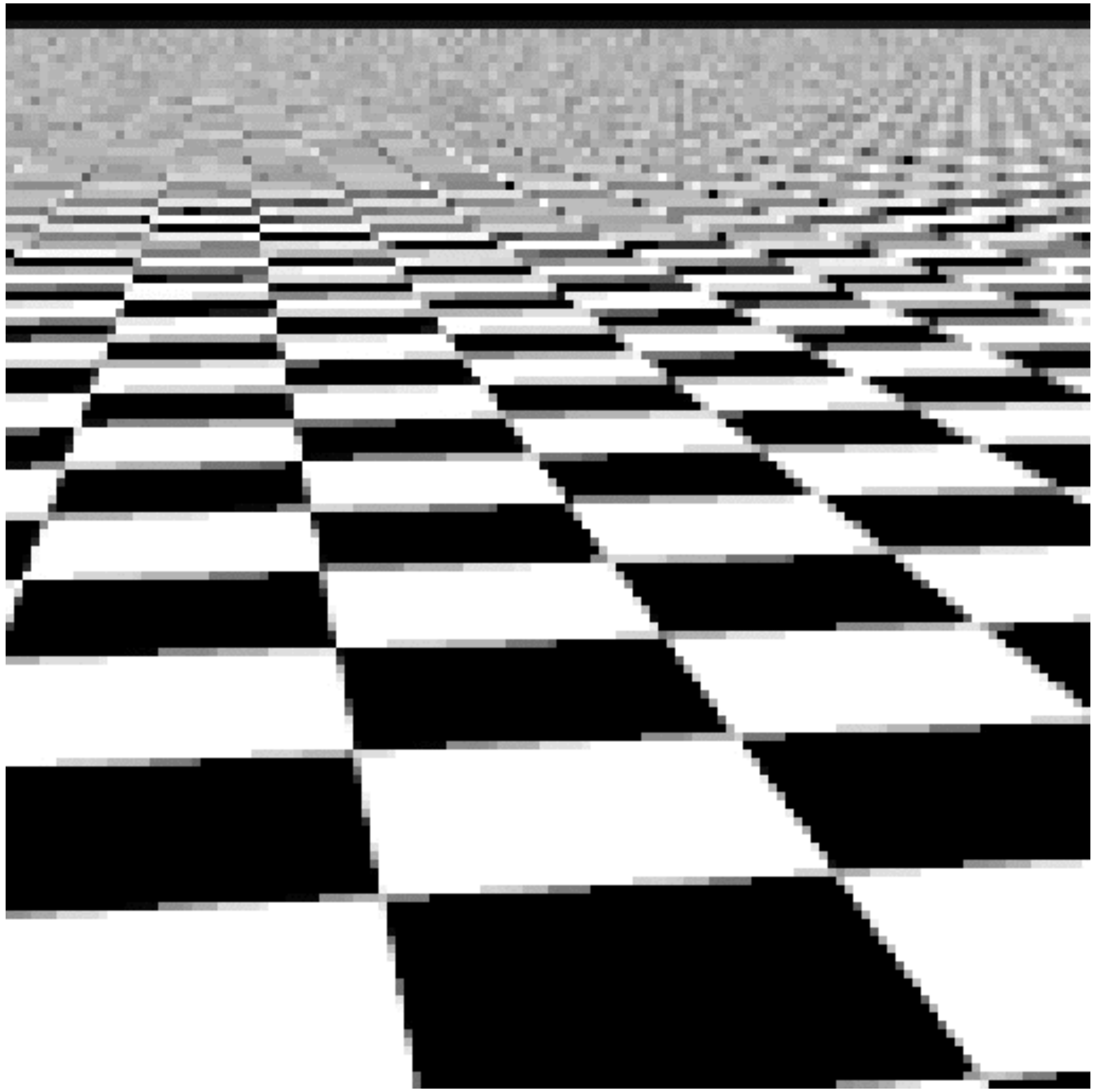


4×4 supersampling

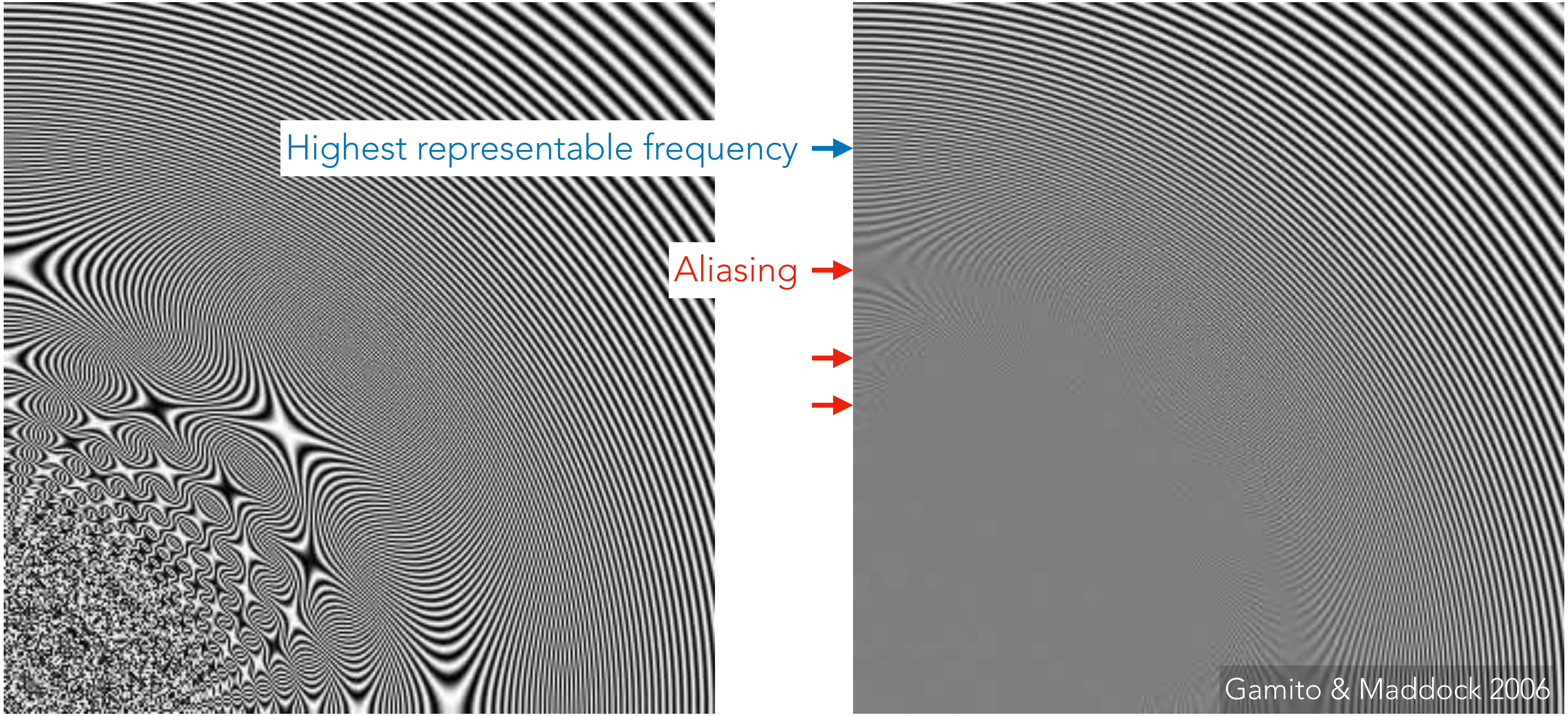
Supersampling



4×4 supersampling



32×32 supersampling



1 sample per pixel

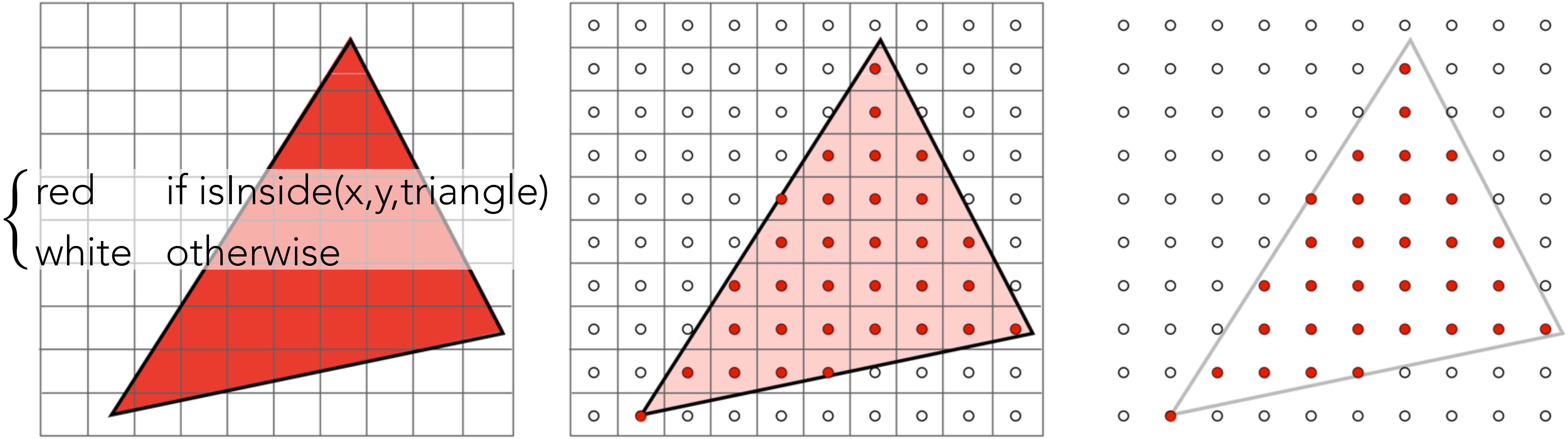
Averaging over pixel area

Pretty good! But why do we still have aliasing?

Signal processing

Sampling and reconstruction

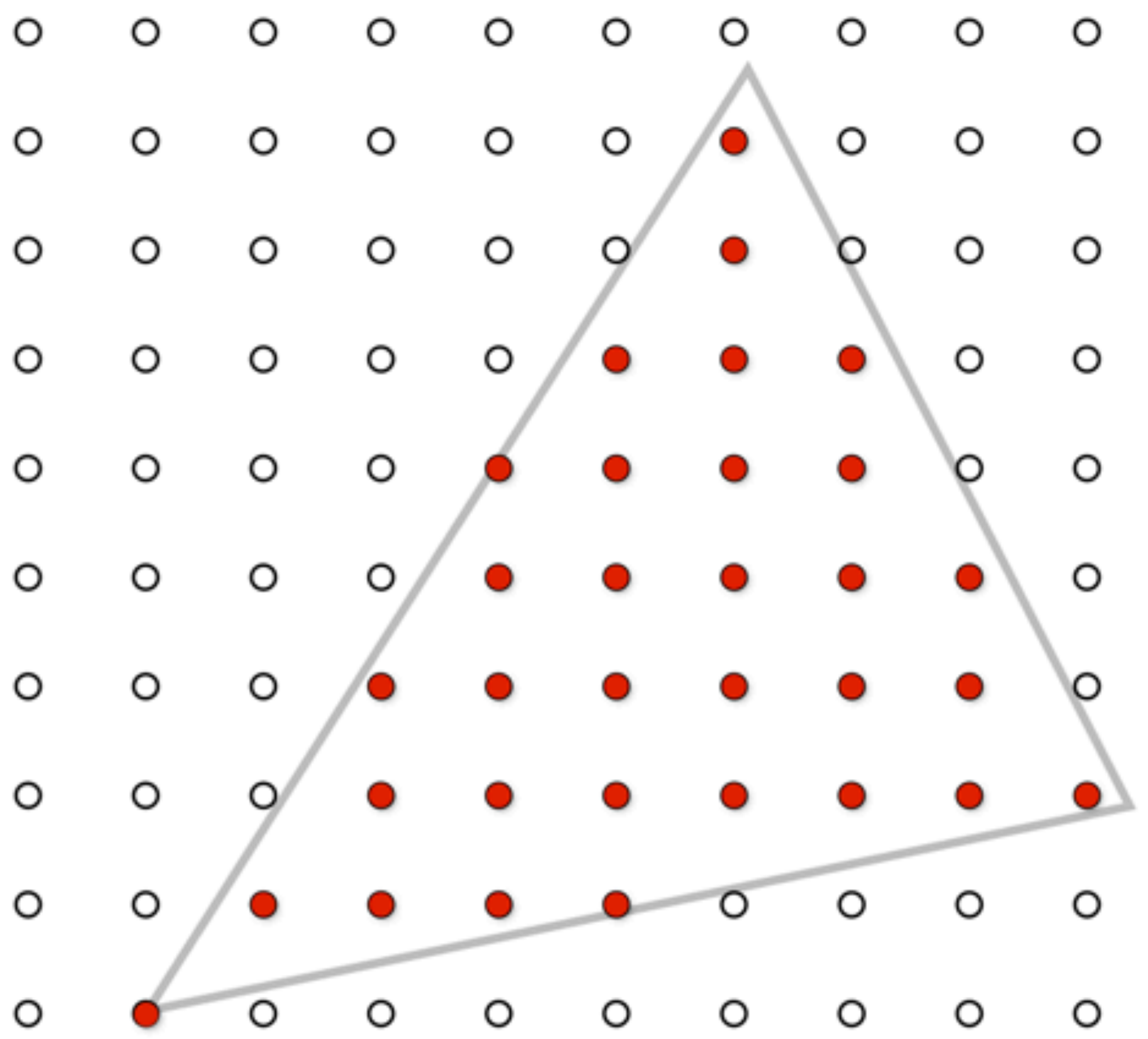
What we actually want to display is a spatial signal $\mathbb{R}^2 \rightarrow$ Colour

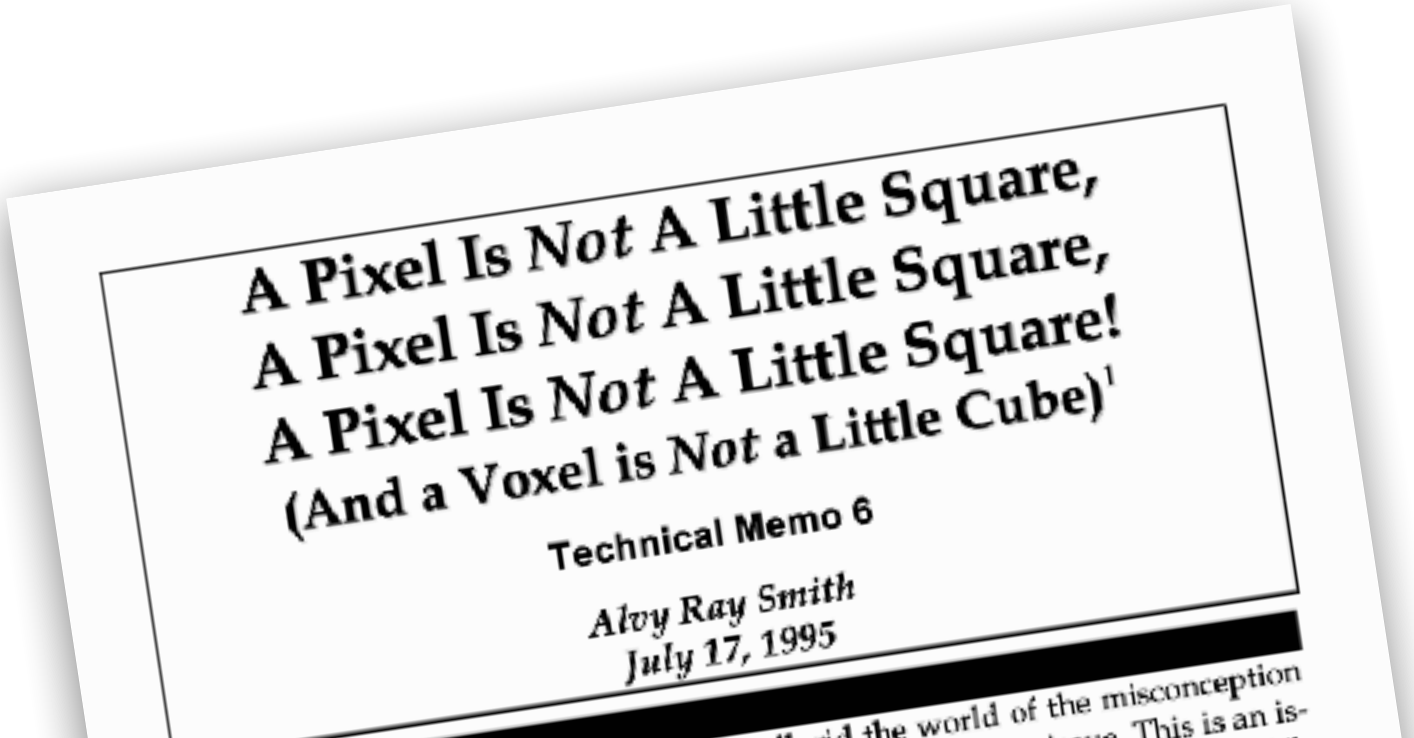


In rasterization, we are sampling the signal at finitely many points.
After sampling, all we know are the values at the sample points.

Sampling and reconstruction

From the samples, we can try to reconstruct the original signal in various ways





A Pixel Is *Not* A Little Square,
A Pixel Is *Not* A Little Square,
A Pixel Is *Not* A Little Square!
(And a Voxel is *Not* a Little Cube)[1]

Technical Memo 6

Alvy Ray Smith
July 17, 1995

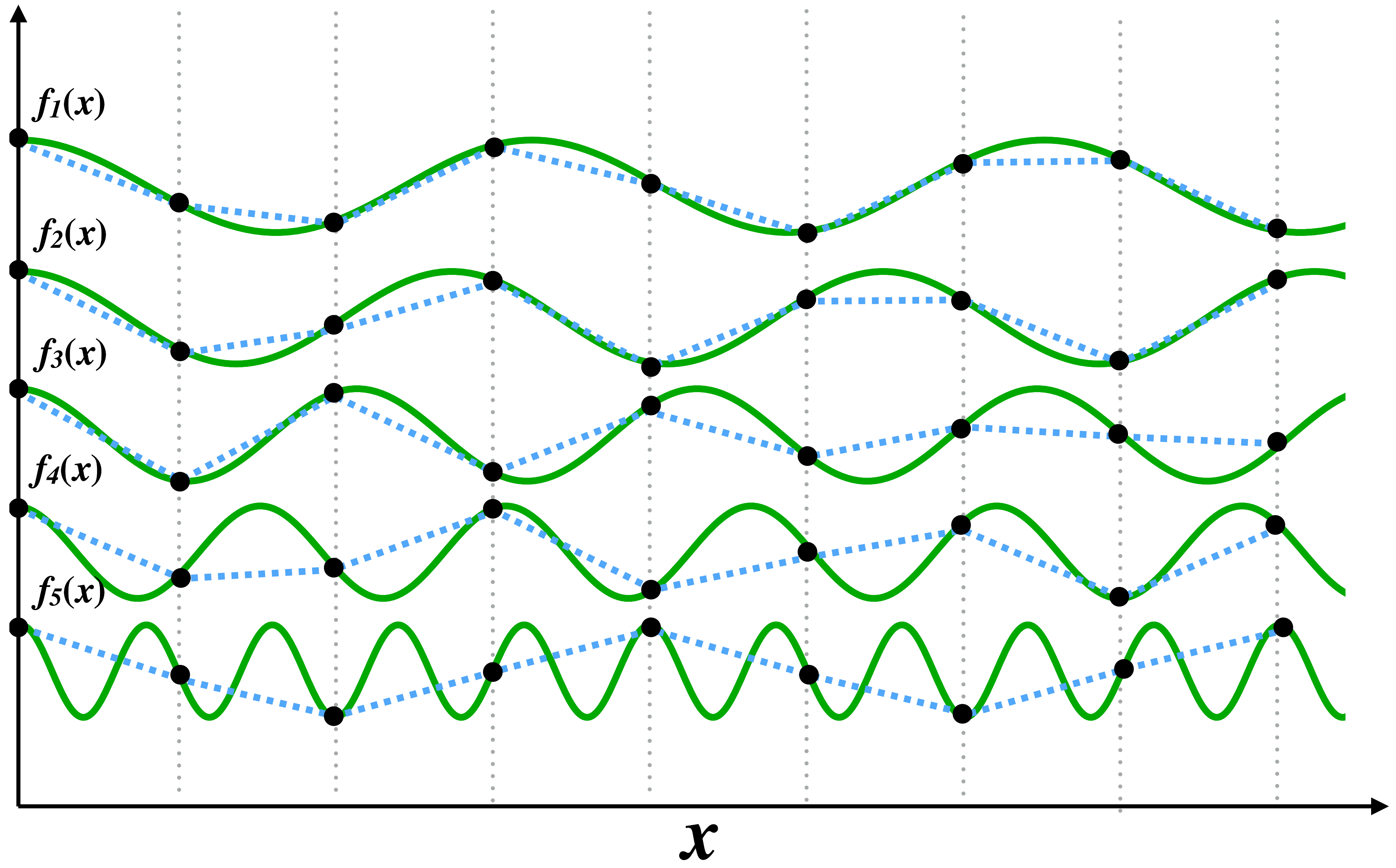
Nearest neighbour

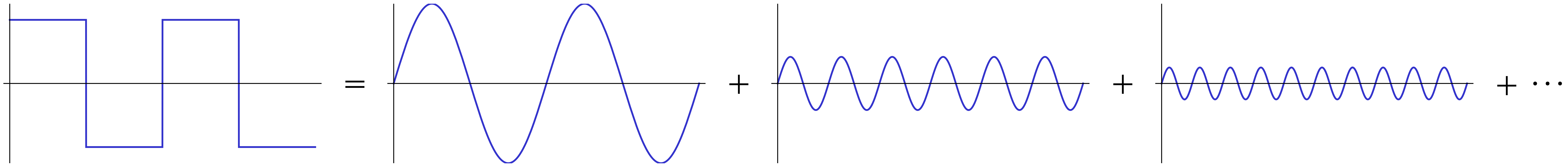
Bilinear interpolation

We already know that insufficient sampling rate causes aliasing in sinusoidal signals.

What about other signals?

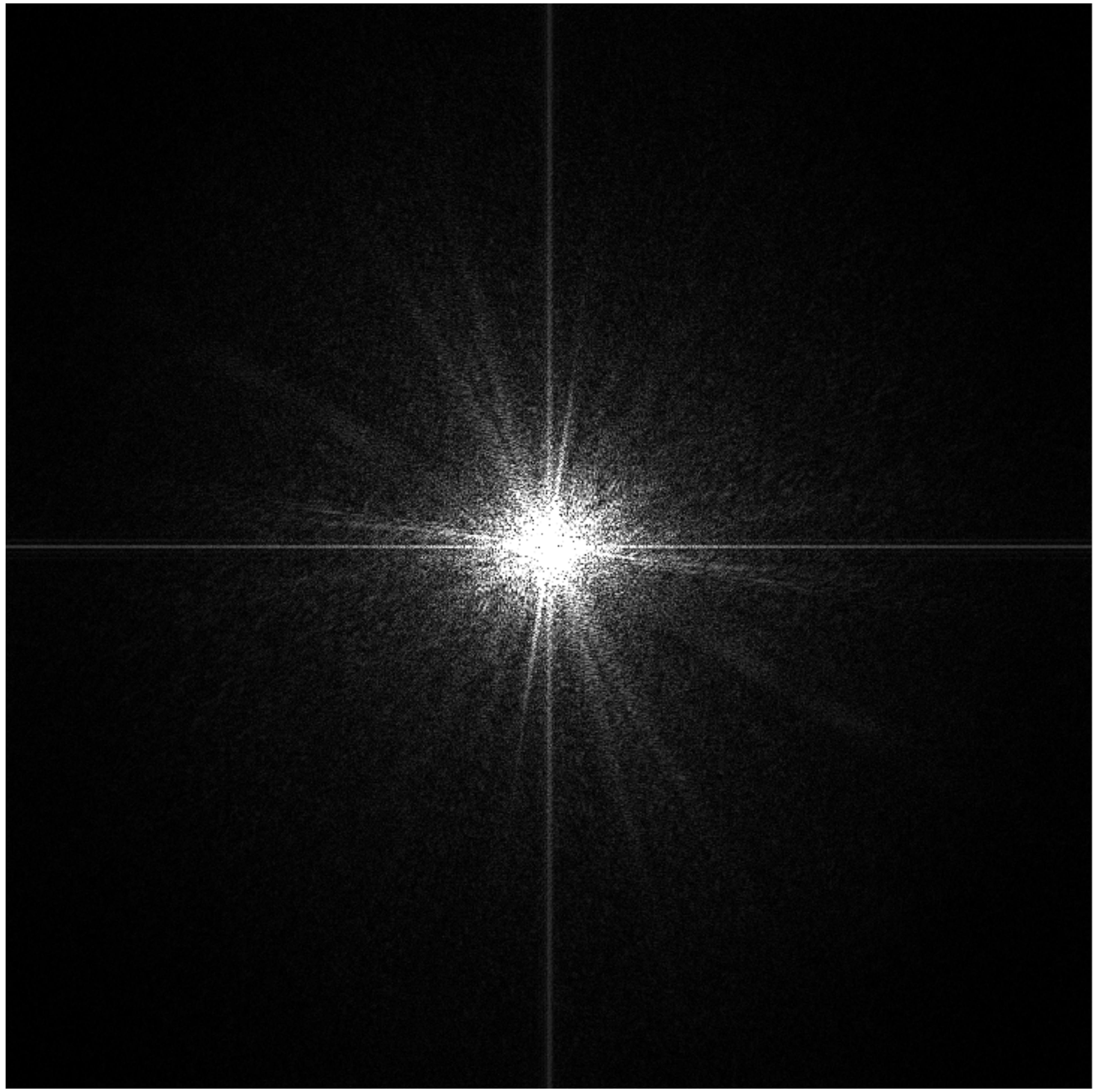
Fourier transform: Any signal can be decomposed into a sum of sinusoids!





Fourier transform of an image

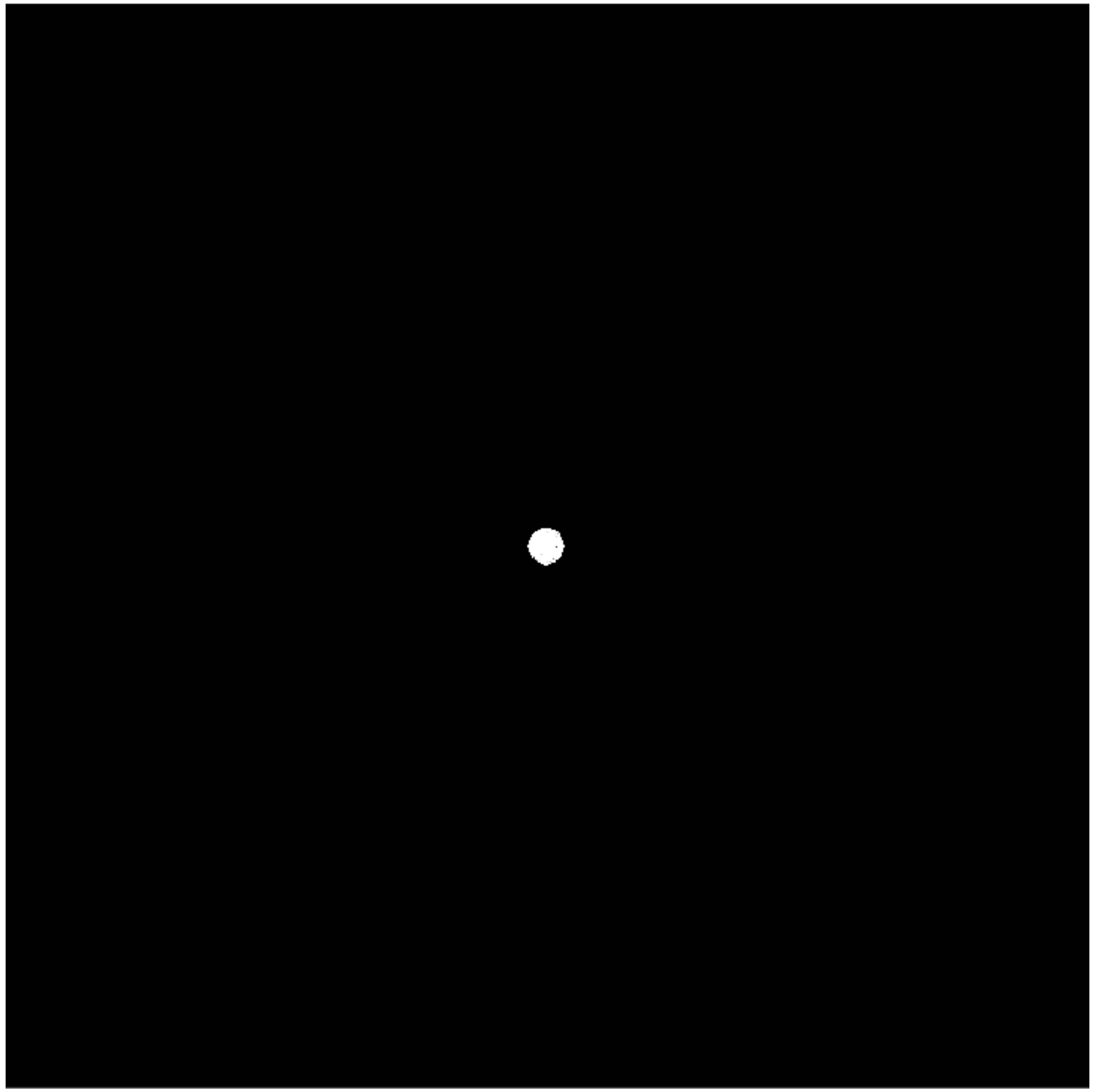
Spatial domain



Frequency domain

Low frequencies

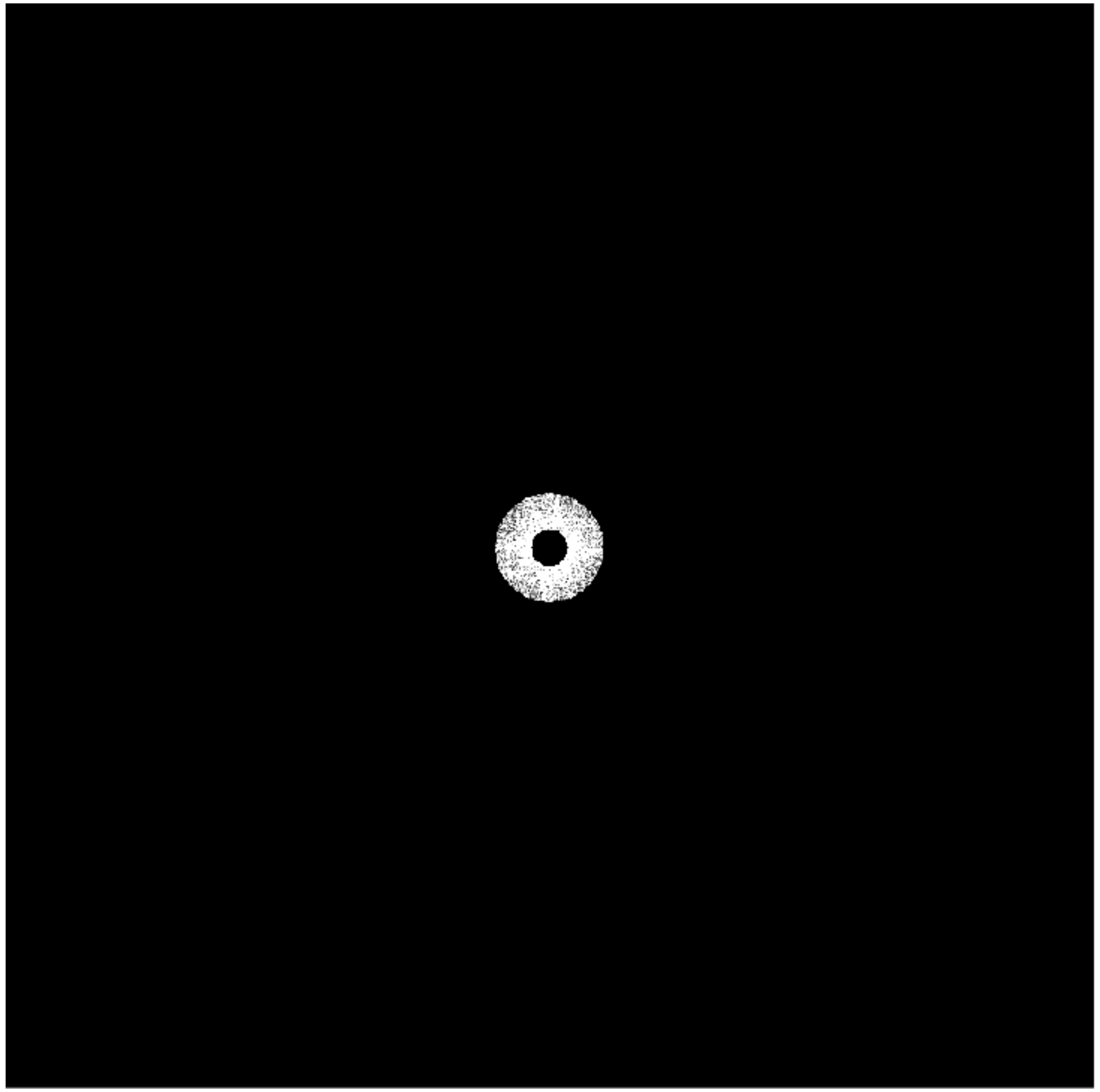
Spatial domain



Frequency domain

Medium frequencies

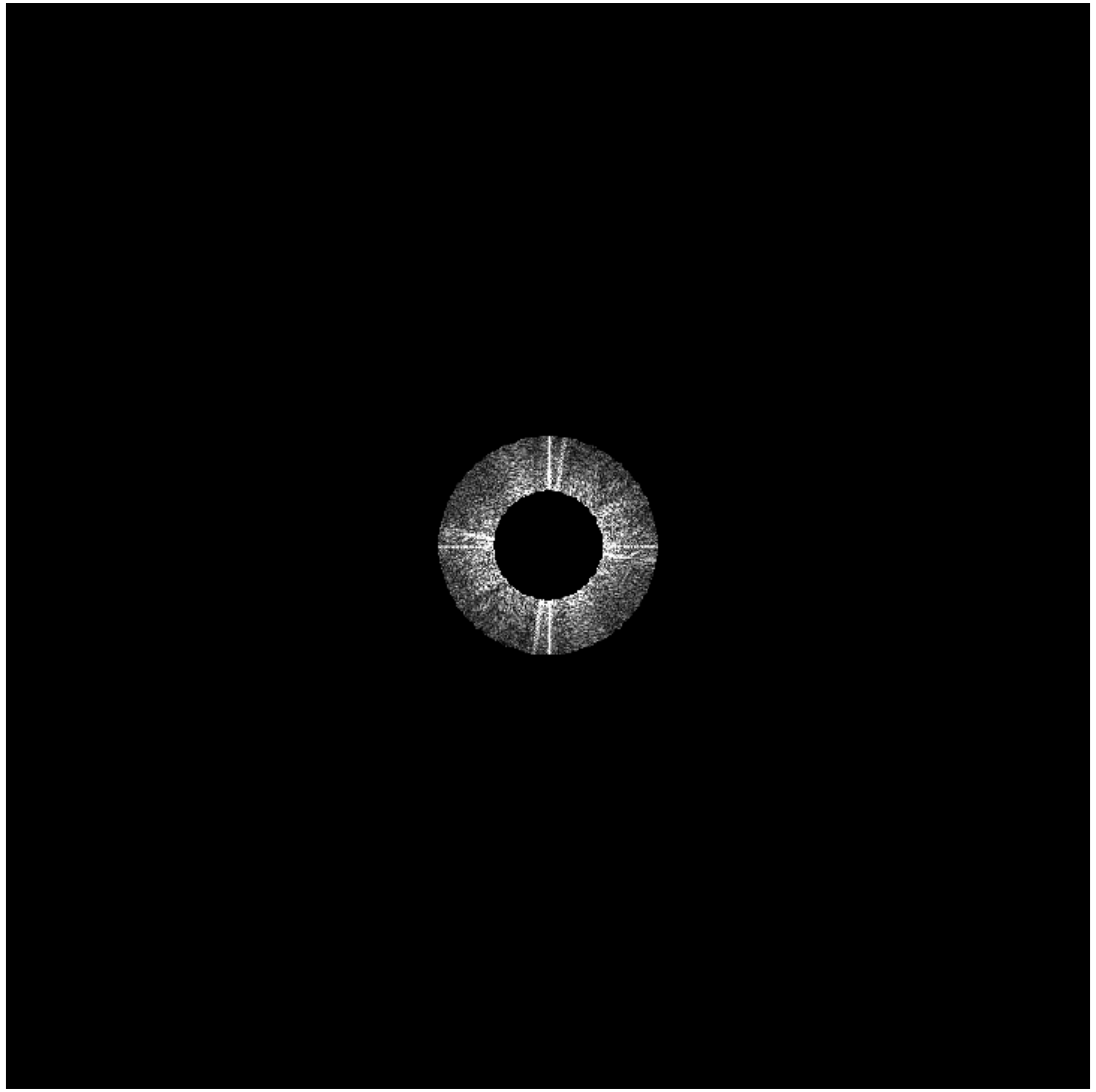
Spatial domain



Frequency domain

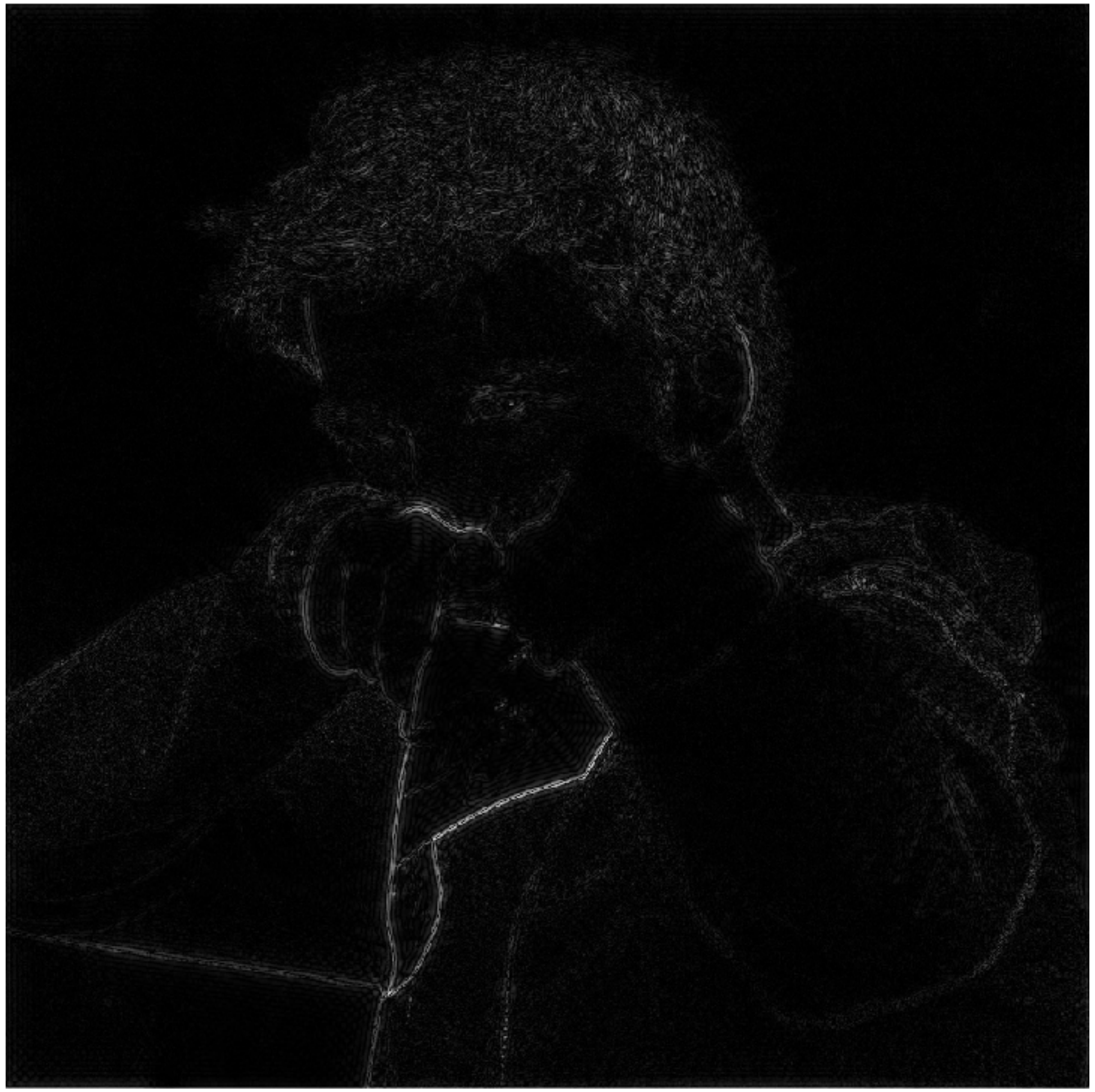
Medium frequencies

Spatial domain

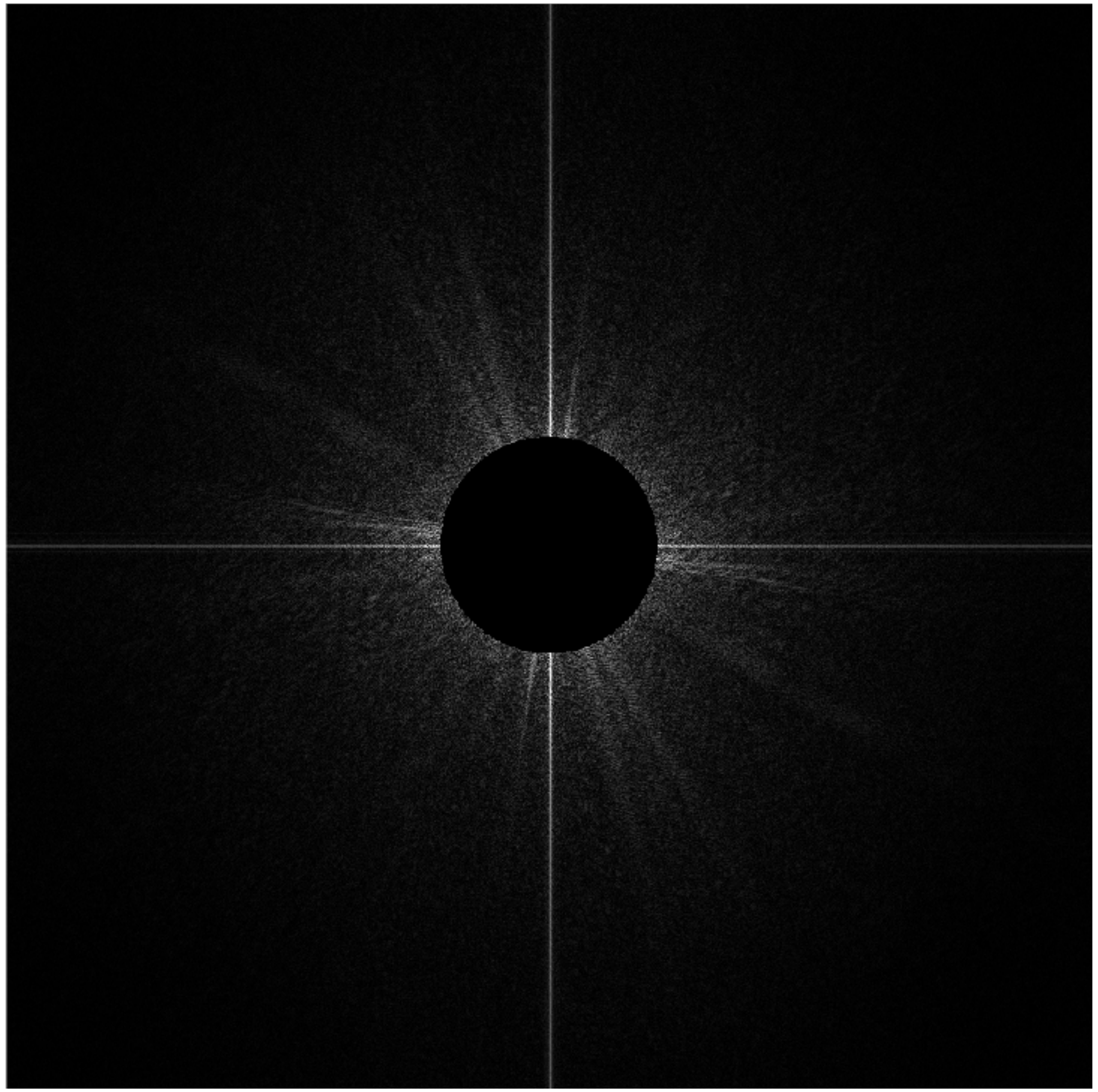


Frequency domain

High frequencies



Spatial domain

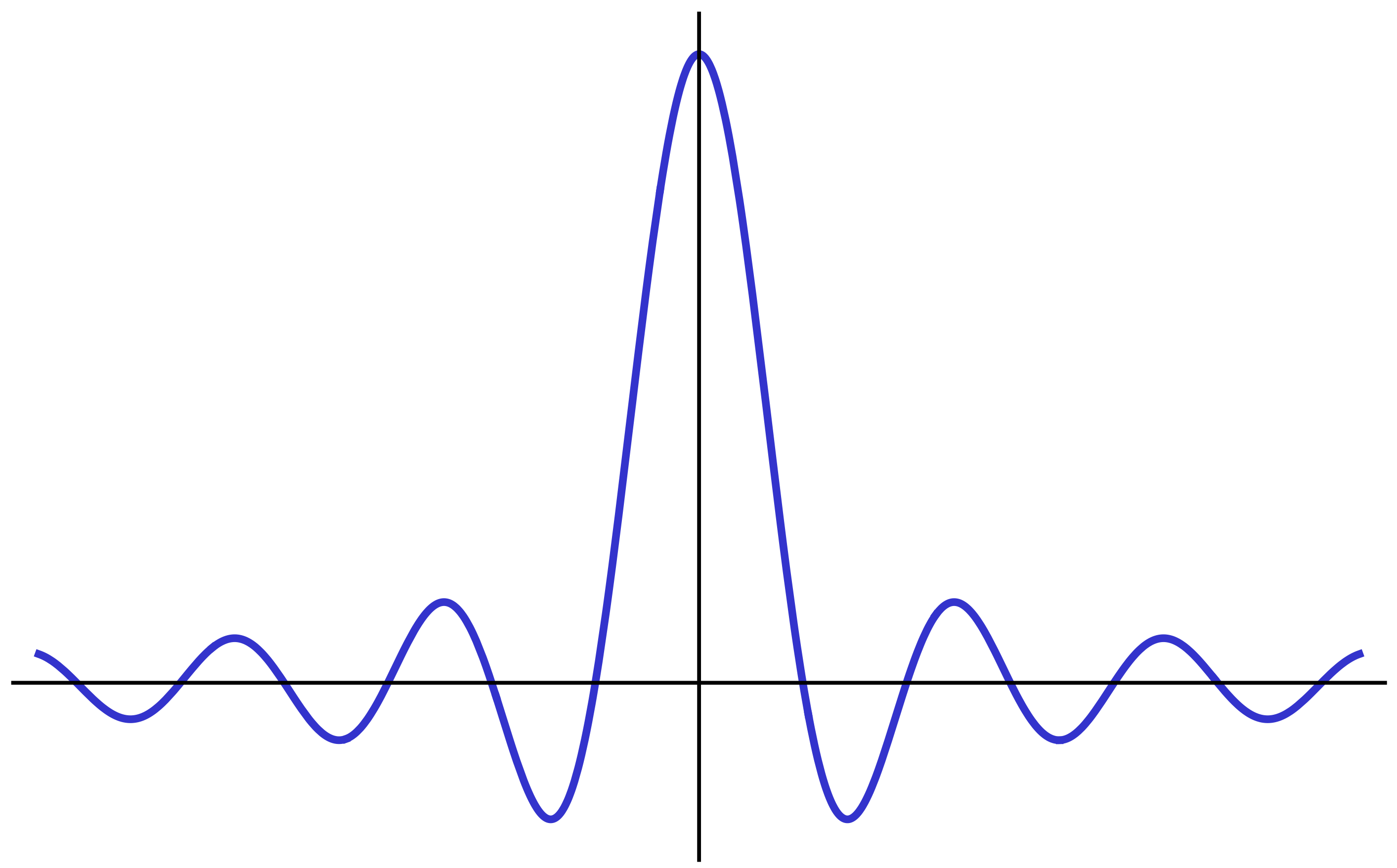


Frequency domain

The Nyquist-Shannon sampling theorem

If a signal has no frequencies higher than some cutoff B,
then it can be perfectly reconstructed after sampling with spacing $1/(2B)$.

- i.e. two samples per period of the highest-frequency component
- Perfect reconstruction requires a sinc kernel: $(\sin x)/x$

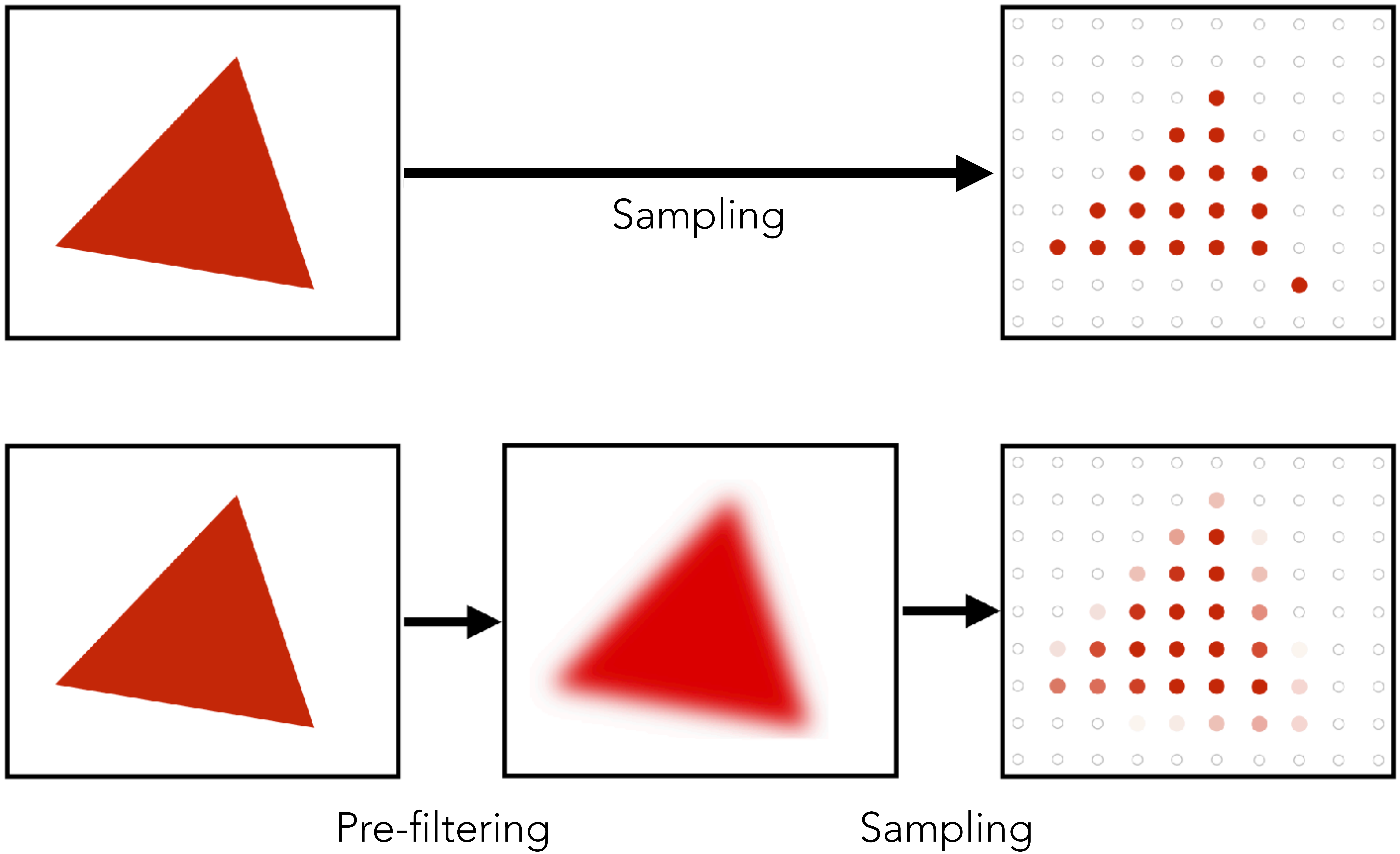


The Nyquist-Shannon sampling theorem

If a signal has no frequencies higher than some cutoff B,
then it can be perfectly reconstructed after sampling with spacing $1/(2B)$.

Practical consequences: To get a faithful reconstruction, we have to eliminate frequencies above the sampling limit!

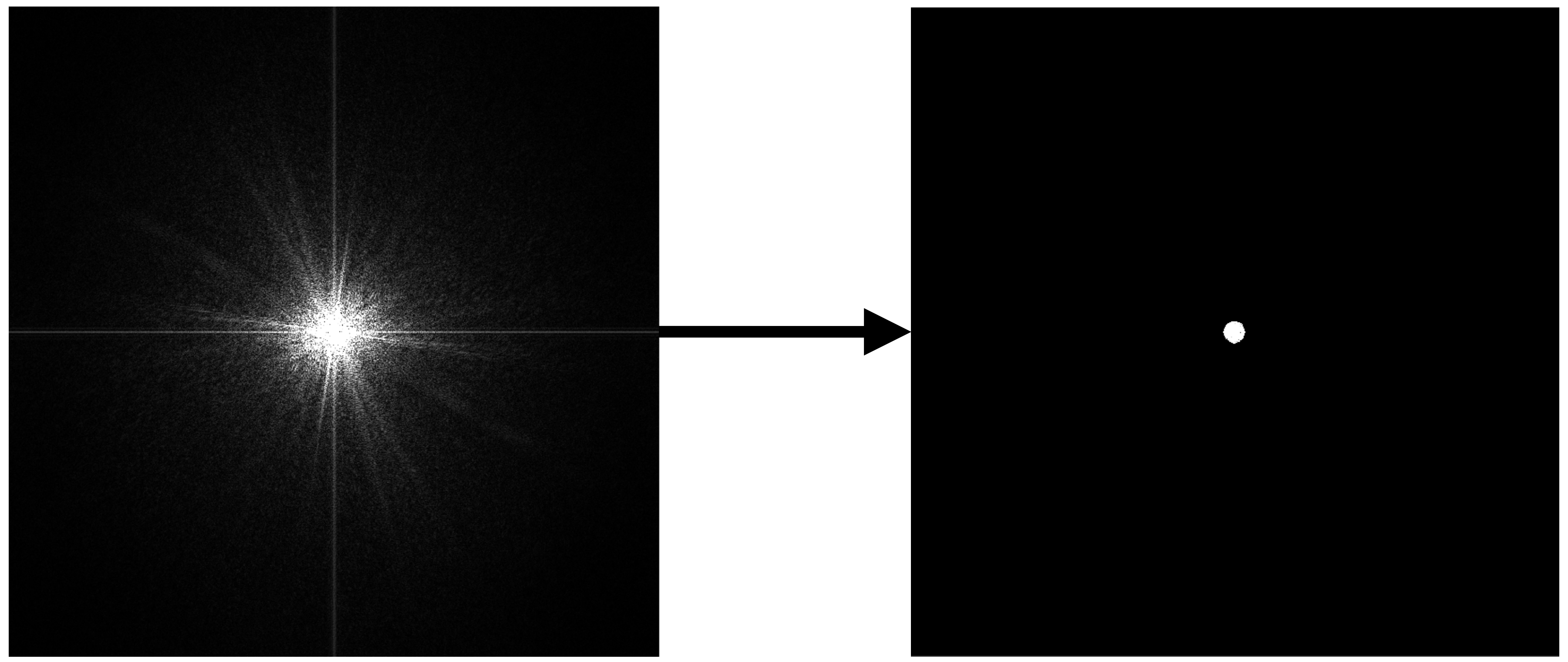
Removing high frequencies = blurring


Sampling
Pre-filtering
Sampling

The theoretical basis of anti-aliasing

Ideally, we'd like to:

1. Take the Fourier transform
2. Multiply high-frequency components by 0
3. Take the inverse Fourier transform
4. Sample the filtered signal



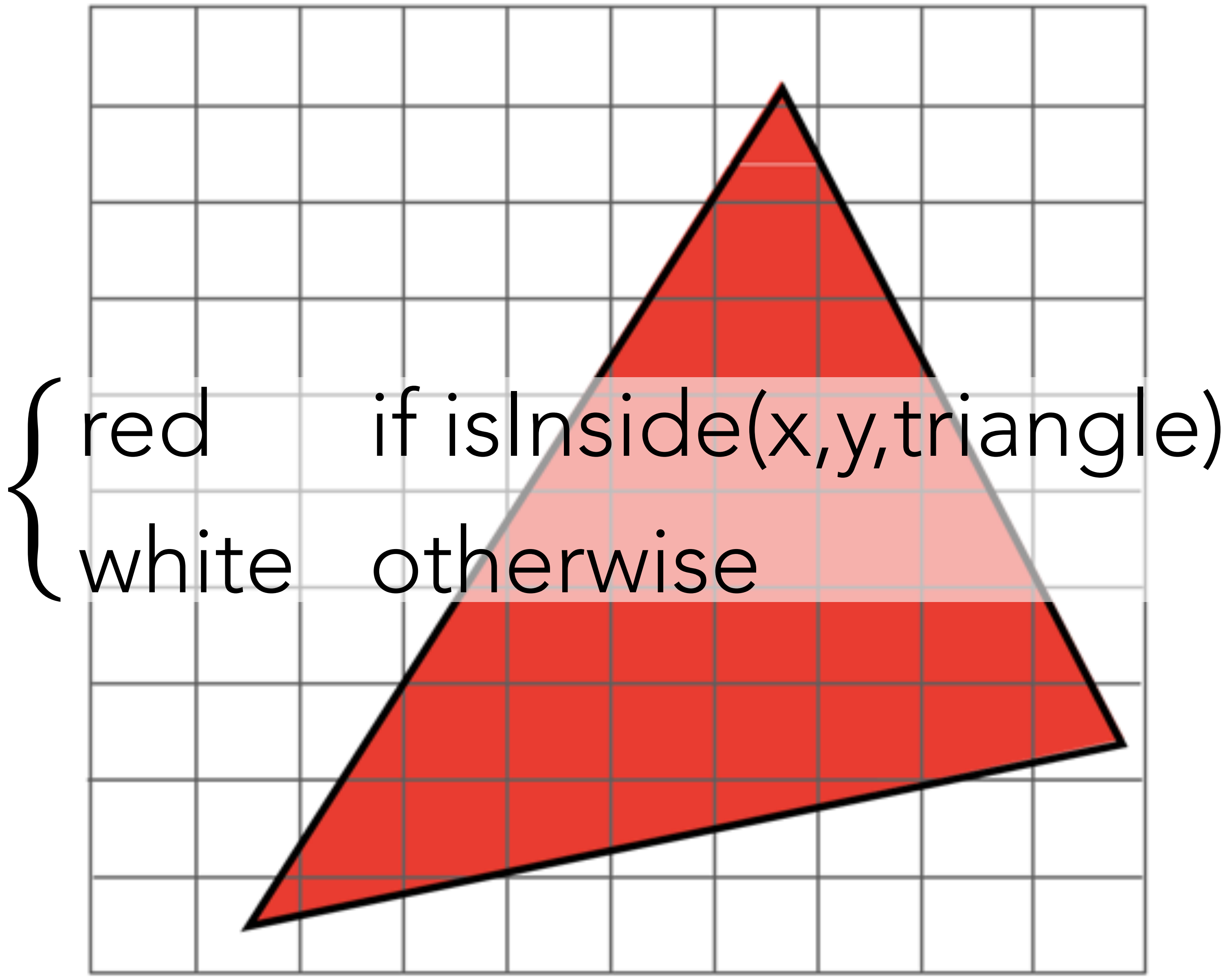
Convolution theorem: Multiplication in the frequency domain = convolution in the spatial domain

Ideally, we'd like to:

1. Convolve the signal with a sinc

2. Sample the filtered signal

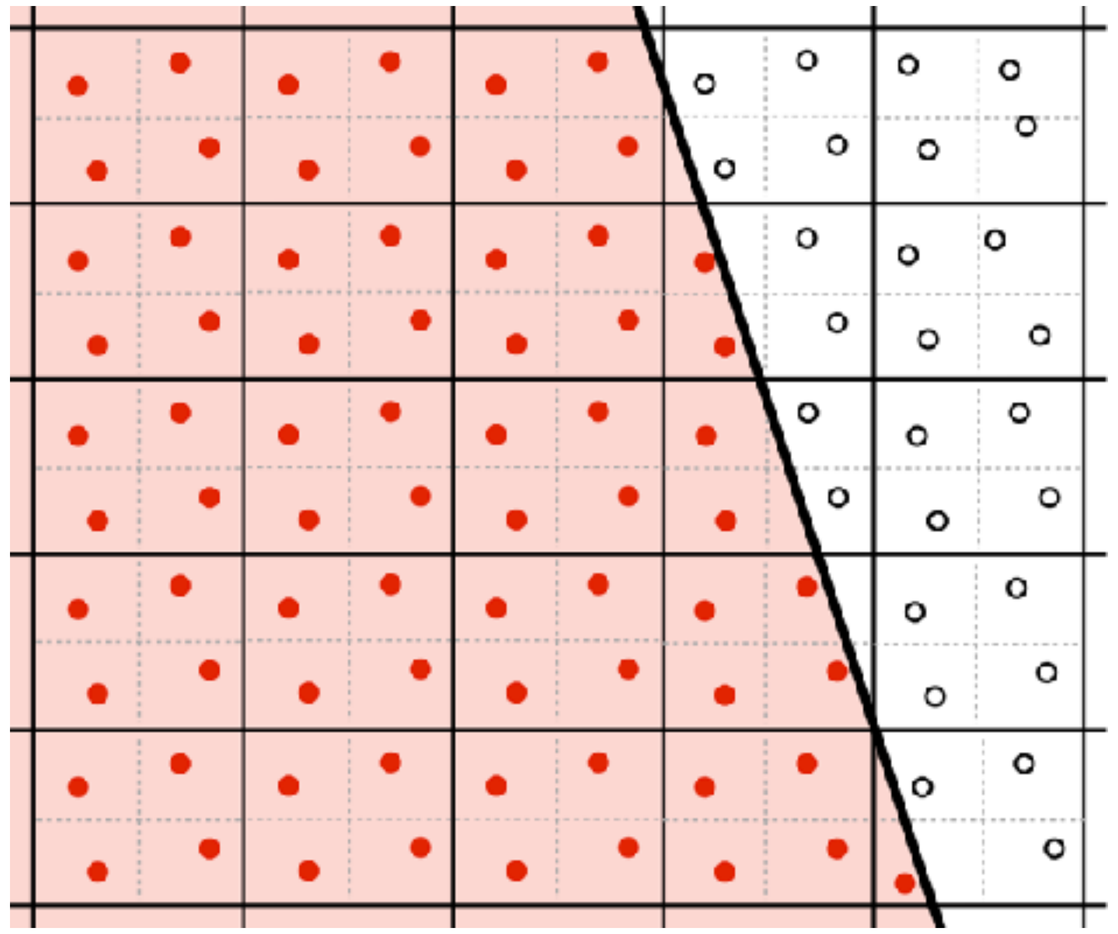
But, of course, we don't know how to convolve a shape with a sinc.

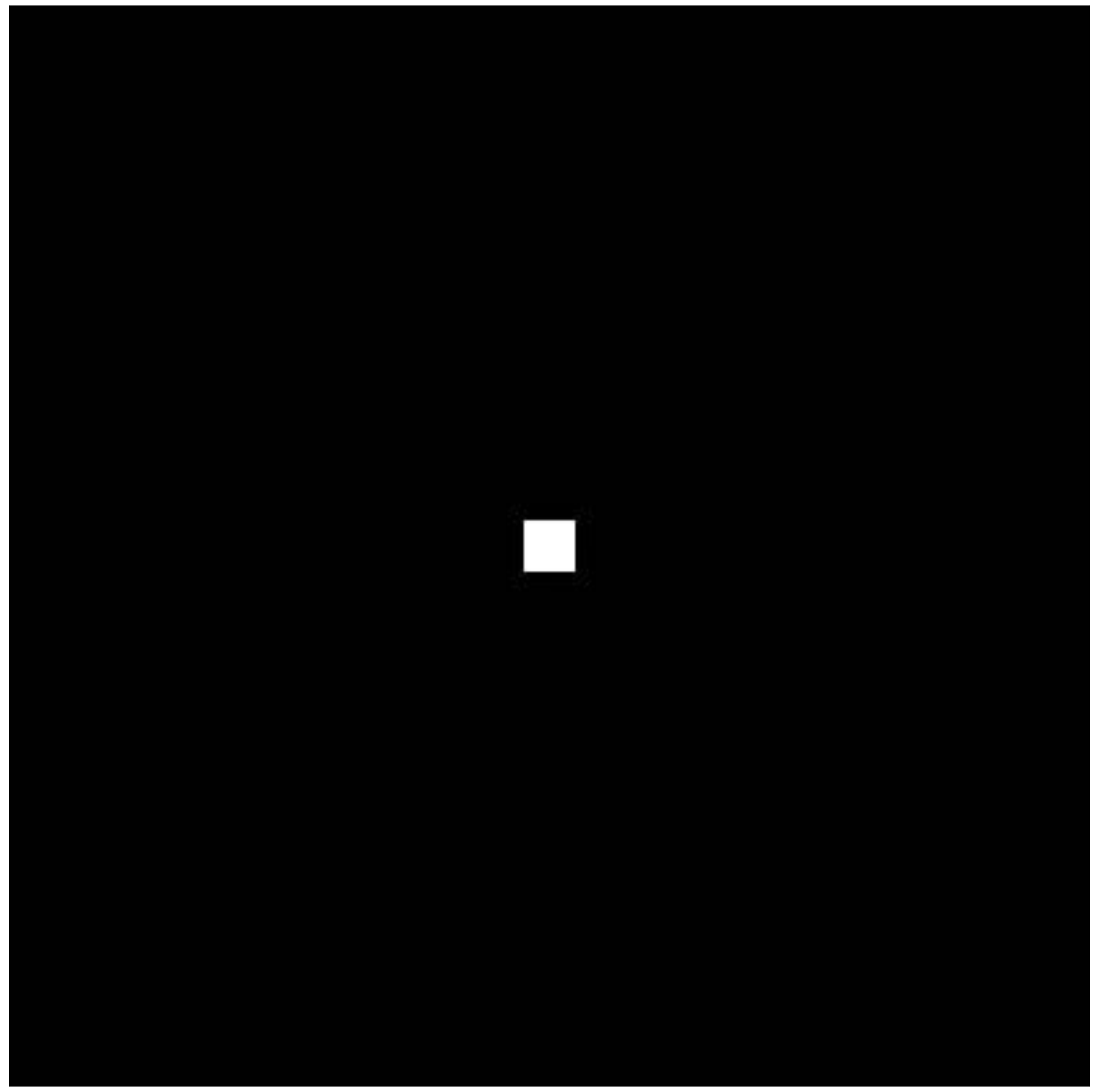


Convolution with sinc + sampling
= weighted integral of all nearby points
≈ weighted average of some nearby points

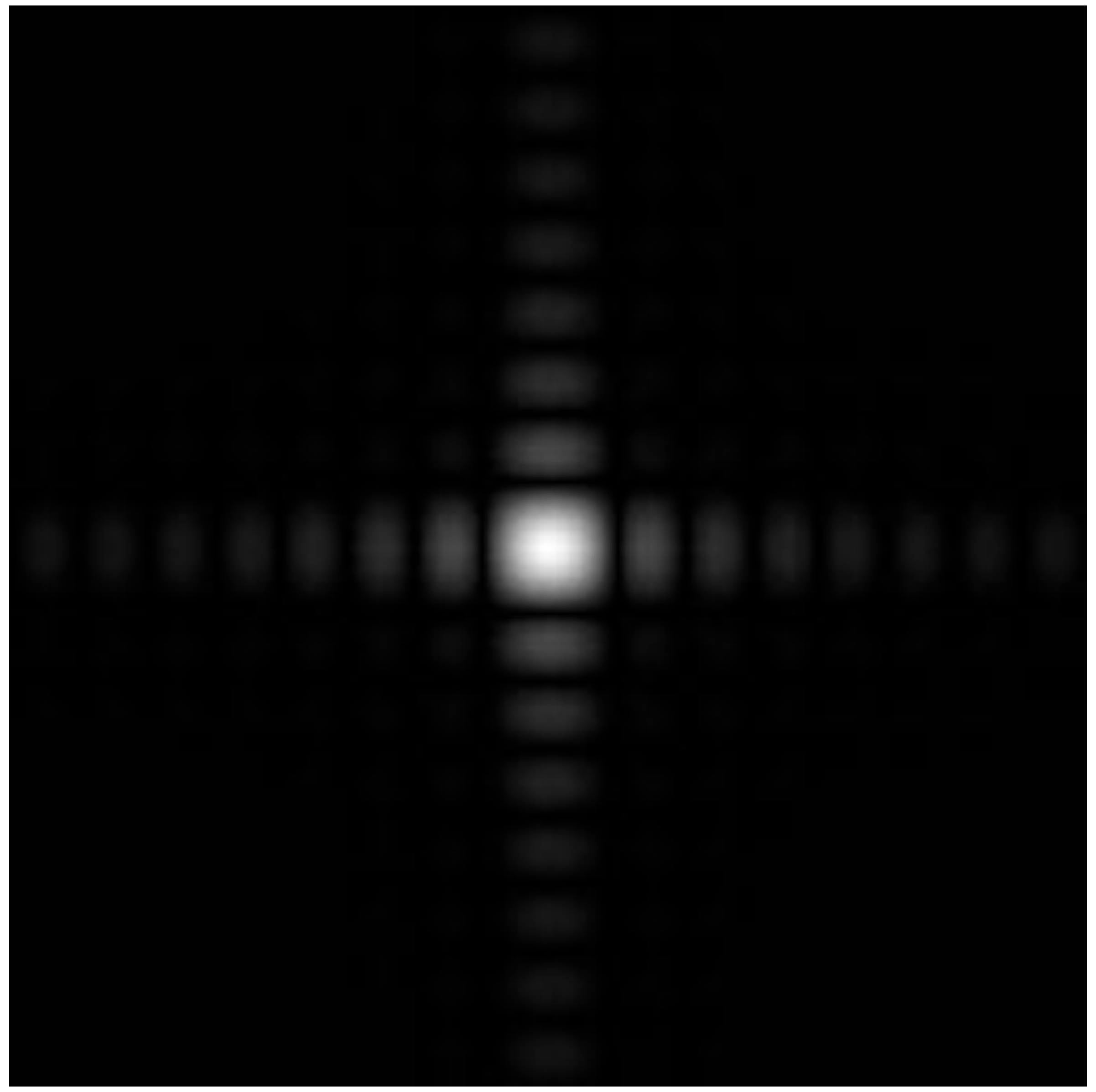
Convolution with sinc + sampling
= weighted integral of all nearby points
≈ weighted average of some nearby points

Naïve averaging = unweighted average
≈ convolution with box filter

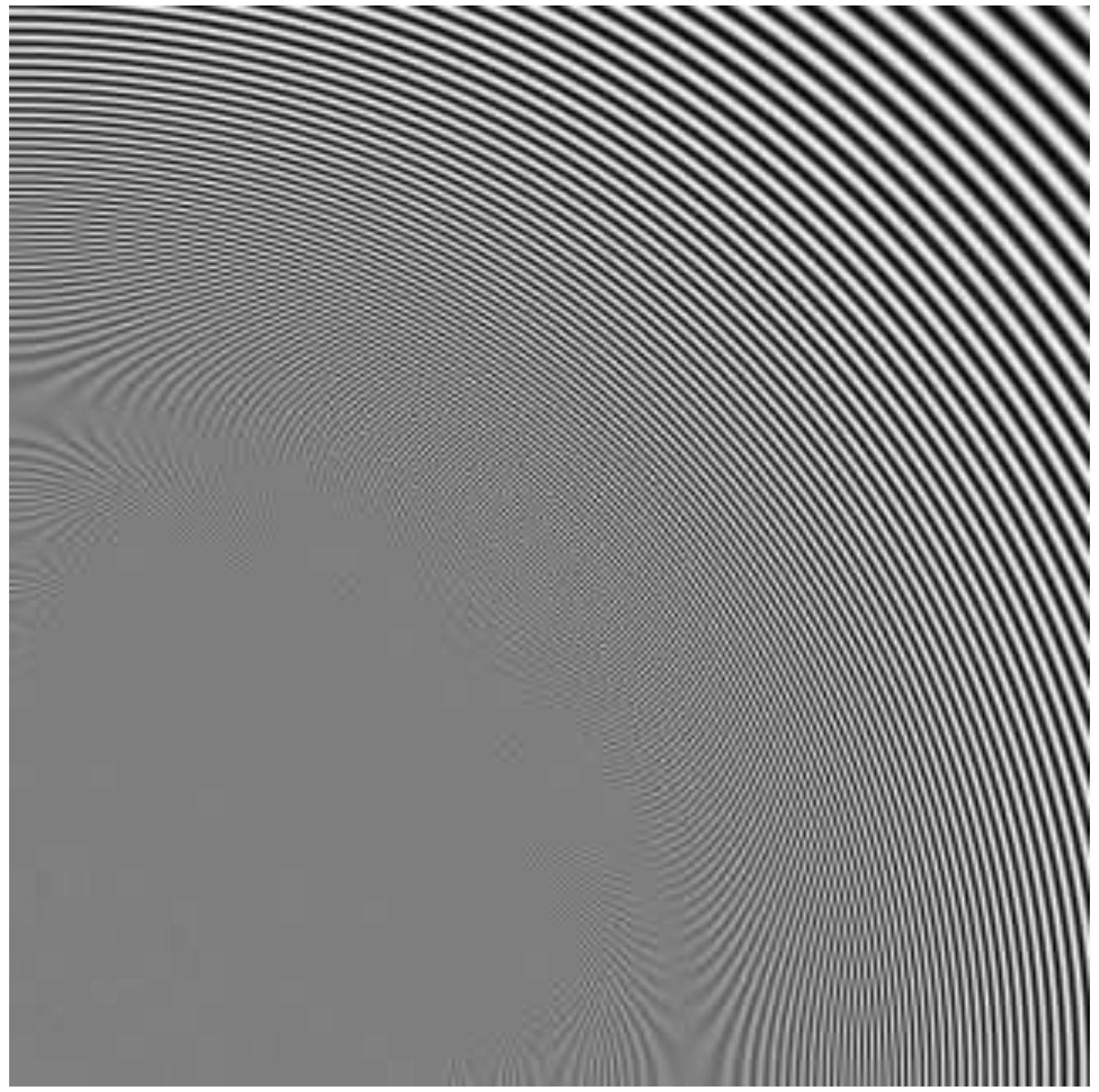




Spatial domain



Frequency domain



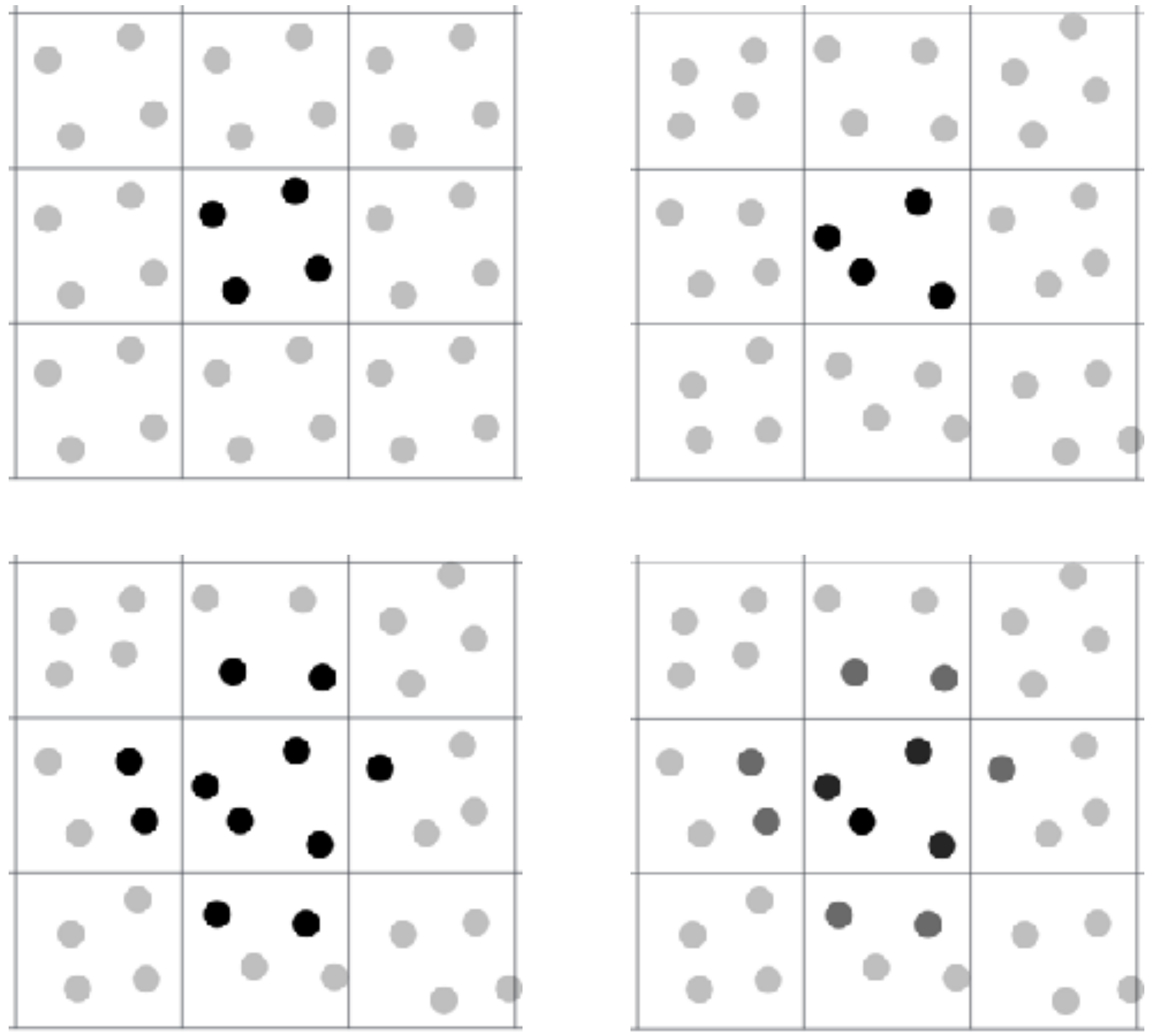
Now we understand:

- why aliasing really happens
- the theoretically ideal (but impractical) solution
- why box filtering is good but not perfect

Much more can be said!

- What are good choices for sampling locations?
- Does the ideal filter also "look good" visually?

But we'll stop here.

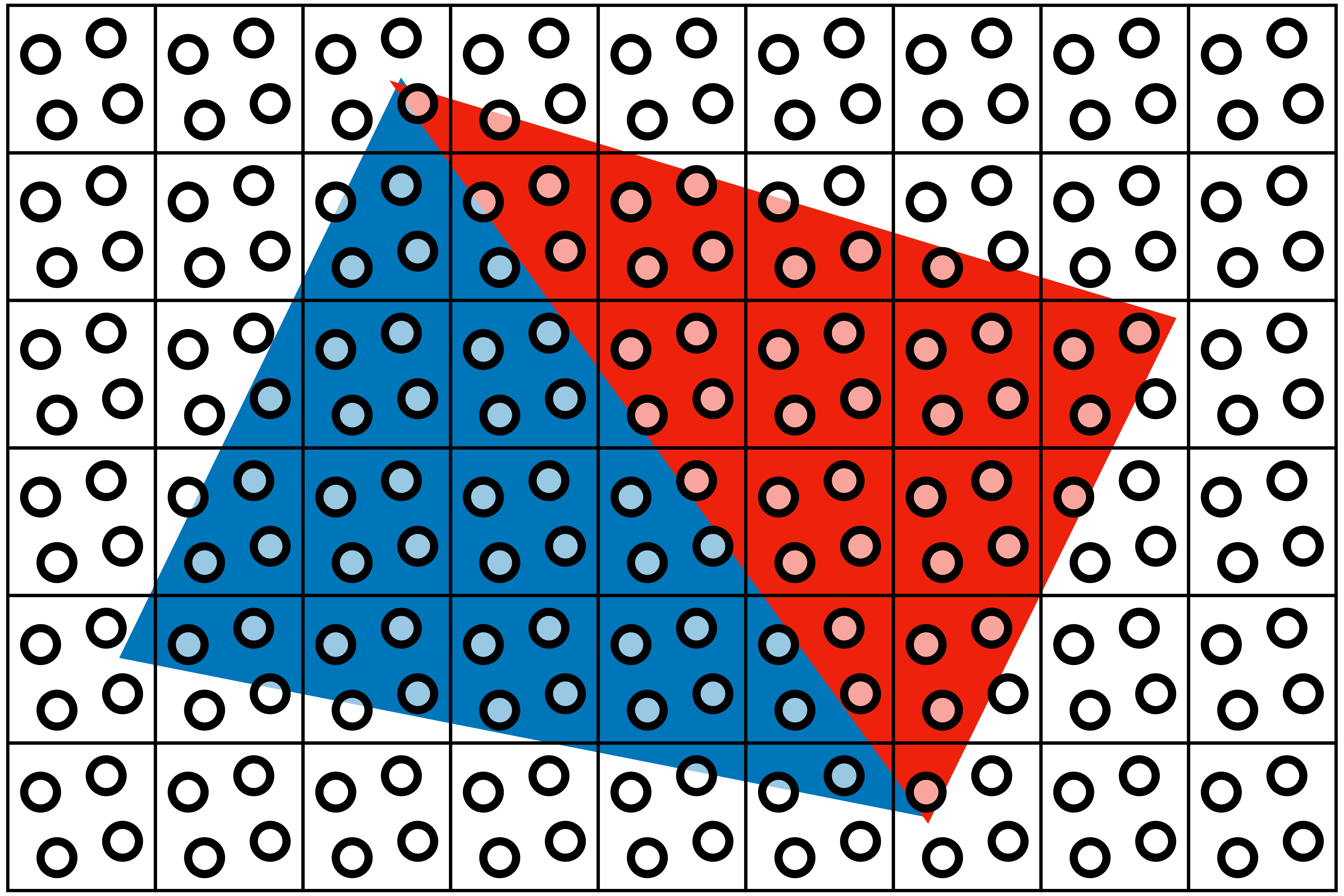


Something to think about

Suppose you want to draw multiple triangles. When should you average a pixel's sample values down to a single colour?

- After drawing each triangle?
- Only in the end?

How do these choices affect the image quality and the memory usage?



Acknowledgements

This lecture's slides are heavily based on those of Ren Ng and Keenan Crane.