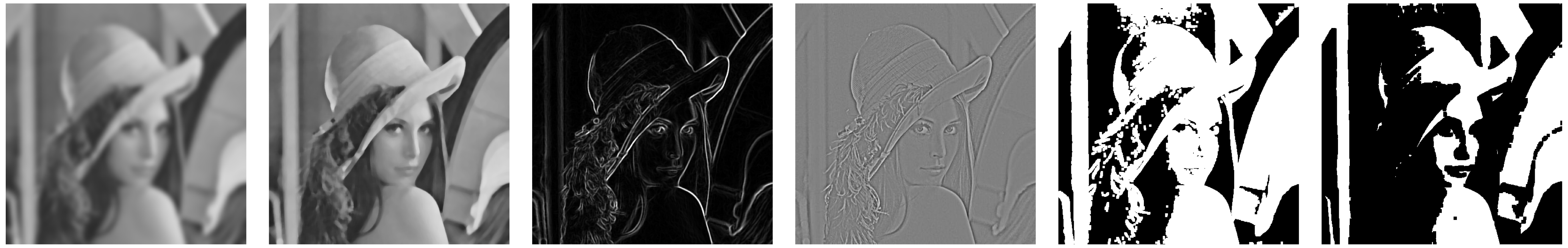
THE UNIVERSITY OF BRITISH COLUMBIA

CPSC 425: Computer Vision



Lecture 5: Image Filtering (continued)

(unless otherwise stated slides are taken or adopted from **Bob Woodham, Jim Little** and **Fred Tung**)

Menu for Today (**September 14, 2018**)

Topics:

— Gaussian and Pillbox filters

— Separability

— The Convolution Theorem

— Non-linear filters

Redings:

— **Today's** Lecture: none

— **Next** Lecture: [Optional] Forsyth & Ponce (2nd ed.) 4.4

Reminders:

— **Assignment 1:** Image Filtering and Hybrid Images due **September 24**th

Today's "**fun**" Example: Rolling Shutter

Today's "**fun**" Example: Rolling Shutter

Lecture 4: Re-cap

— The **correlation** of $F(X,Y)$ and $I(X,Y)$ is:

$$\underbrace{I'(X,Y)}_{\text{output}} = \sum_{j=-k}^{k} \sum_{i=-k}^{k} \underbrace{F(I,J)}_{\text{filter}} \underbrace{I(X+i, Y+j)}_{\text{image (signal)}}$$

— **Visual interpretation**: Superimpose the filter F on the image I at (X,Y), perform an element-wise multiply, and sum up the values

— **Convolution** is like **correlation** except filter "flipped"

if $F(X,Y) = F(-X,-Y)$ then correlation = convolution.

Lecture 4: Re-cap

Ways to handle **boundaries**

– **Ignore/discard**. Make the computation undefined for top/bottom k rows and left/right-most k columns

– **Pad with zeros**. Return zero whenever a value of I is required beyond the image bounds

– **Assume periodicity.** Top row wraps around to the bottom row; leftmost column wraps around to rightmost column.

Simple **examples** of filtering:

— copy, shift, smoothing, sharpening

Linear filter **properties**:

— superposition, scaling, shift invariance

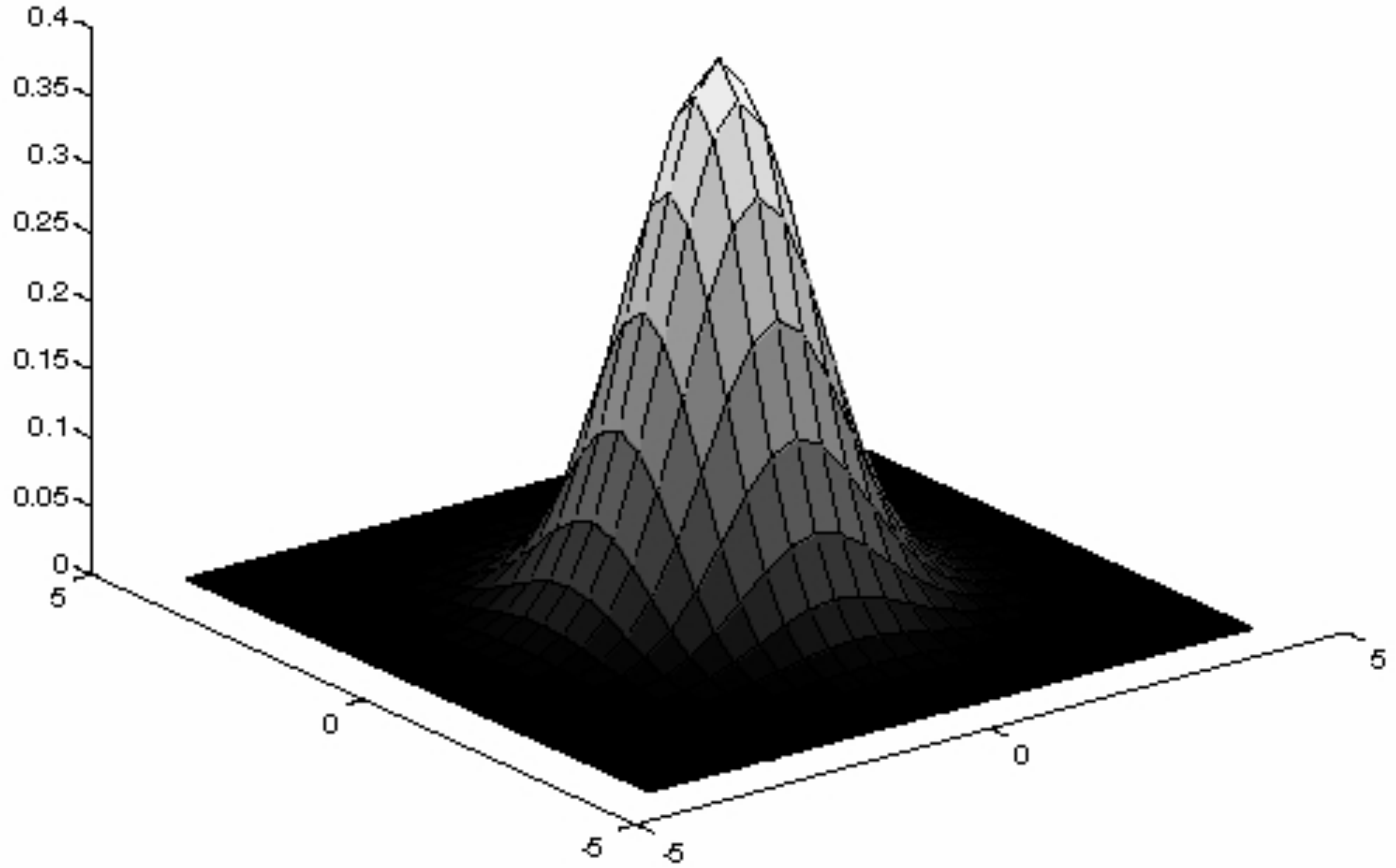
Characterization Theorem: Any linear, shift-invariant operation can be expressed as a convolution

Example 6: Smoothing with a Gaussian

Idea: Weight contributions of pixels by spatial proximity (nearness)

2D **Gaussian** (continuous case):

$$G_\sigma(x, y) = \frac{1}{2\pi\sigma^2} \exp^{-\frac{x^2+y^2}{2\sigma^2}}$$



Forsyth & Ponce (2nd ed.)
Figure 4.2

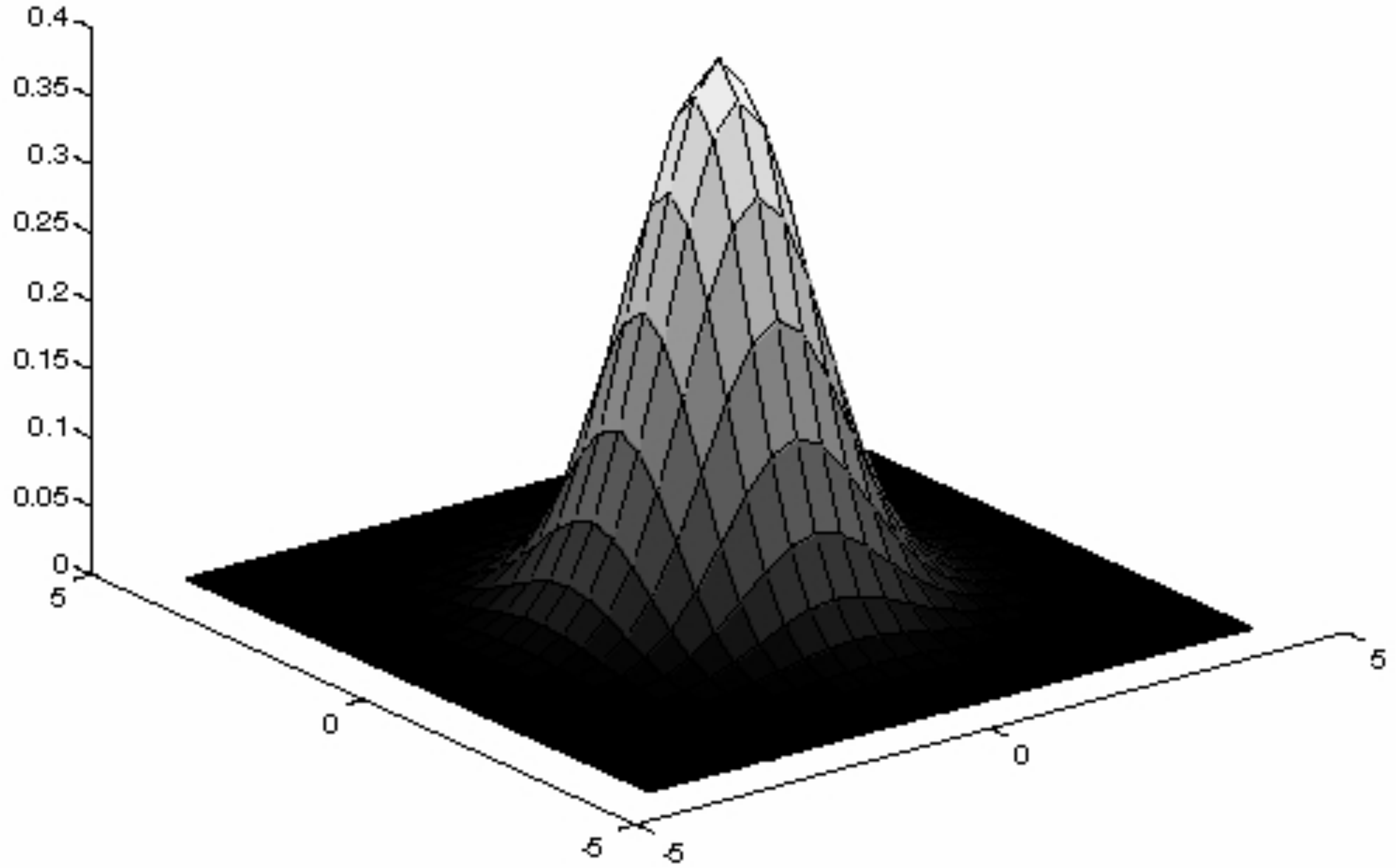
Example 6: Smoothing with a Gaussian

Idea: Weight contributions of pixels by spatial proximity (nearness)

2D **Gaussian** (continuous case):

$$G_{\sigma}(x,y) = \frac{1}{2\pi\sigma^2} \exp^{-\frac{x^2+y^2}{2\sigma^2}}$$

Standard Deviation



Forsyth & Ponce (2nd ed.)
Figure 4.2

Example 6: Smoothing with a Gaussian

Quantized an truncated **3x3 Gaussian** filter:

$G_\sigma(-1,1)$	$G_\sigma(0,1)$	$G_\sigma(1,1)$
$G_\sigma(-1,0)$	$G_\sigma(0,0)$	$G_\sigma(1,0)$
$G_\sigma(-1,-1)$	$G_\sigma(0,-1)$	$G_\sigma(1,-1)$

Example 6: Smoothing with a Gaussian

Quantized an truncated **3x3 Gaussian** filter:

$G_\sigma(-1,1) = \frac{1}{2\pi\sigma^2} \exp^{-\frac{2}{2\sigma^2}}$	$G_\sigma(0,1) = \frac{1}{2\pi\sigma^2} \exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(1,1) = \frac{1}{2\pi\sigma^2} \exp^{-\frac{2}{2\sigma^2}}$
$G_\sigma(-1,0) = \frac{1}{2\pi\sigma^2} \exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(0,0) = \frac{1}{2\pi\sigma^2}$	$G_\sigma(1,0) = \frac{1}{2\pi\sigma^2} \exp^{-\frac{1}{2\sigma^2}}$
$G_\sigma(-1,-1) = \frac{1}{2\pi\sigma^2} \exp^{-\frac{2}{2\sigma^2}}$	$G_\sigma(0,-1) = \frac{1}{2\pi\sigma^2} \exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(1,-1) = \frac{1}{2\pi\sigma^2} \exp^{-\frac{2}{2\sigma^2}}$

Example 6: Smoothing with a Gaussian

Quantized an truncated **3x3 Gaussian** filter:

$G_\sigma(-1,1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$	$G_\sigma(0,1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(1,1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$
$G_\sigma(-1,0) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(0,0) = \frac{1}{2\pi\sigma^2}$	$G_\sigma(1,0) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$
$G_\sigma(-1,-1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$	$G_\sigma(0,-1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(1,-1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$

With $\sigma = 1$:

0.059	0.097	0.059
0.097	0.159	0.097
0.059	0.097	0.059

Example 6: Smoothing with a Gaussian

Quantized an truncated **3x3 Gaussian** filter:

$G_\sigma(-1,1) = \frac{1}{2\pi\sigma^2} \exp^{-\frac{2}{2\sigma^2}}$	$G_\sigma(0,1) = \frac{1}{2\pi\sigma^2} \exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(1,1) = \frac{1}{2\pi\sigma^2} \exp^{-\frac{2}{2\sigma^2}}$
$G_\sigma(-1,0) = \frac{1}{2\pi\sigma^2} \exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(0,0) = \frac{1}{2\pi\sigma^2}$	$G_\sigma(1,0) = \frac{1}{2\pi\sigma^2} \exp^{-\frac{1}{2\sigma^2}}$
$G_\sigma(-1,-1) = \frac{1}{2\pi\sigma^2} \exp^{-\frac{2}{2\sigma^2}}$	$G_\sigma(0,-1) = \frac{1}{2\pi\sigma^2} \exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(1,-1) = \frac{1}{2\pi\sigma^2} \exp^{-\frac{2}{2\sigma^2}}$

With $\sigma = 1$:

0.059	0.097	0.059
0.097	0.159	0.097
0.059	0.097	0.059

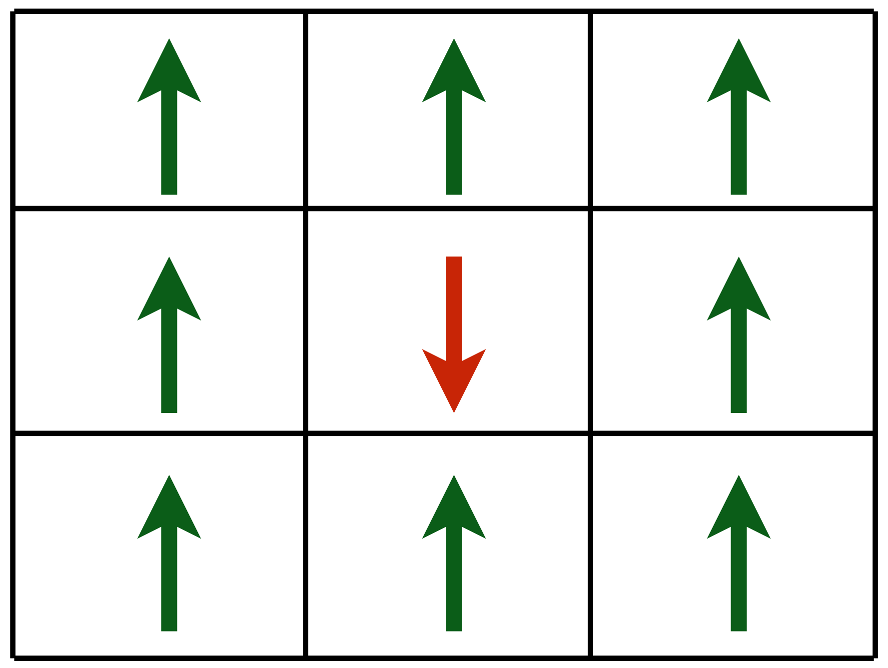
What happens if σ is larger?

Example 6: Smoothing with a Gaussian

Quantized an truncated **3x3 Gaussian** filter:

$G_\sigma(-1,1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$	$G_\sigma(0,1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(1,1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$
$G_\sigma(-1,0) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(0,0) = \frac{1}{2\pi\sigma^2}$	$G_\sigma(1,0) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$
$G_\sigma(-1,-1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$	$G_\sigma(0,-1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(1,-1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$

With $\sigma = 1$:



What happens if σ is larger?

— **More** blur

Example 6: Smoothing with a Gaussian

Quantized an truncated **3x3 Gaussian** filter:

$G_\sigma(-1,1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$	$G_\sigma(0,1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(1,1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$
$G_\sigma(-1,0) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(0,0) = \frac{1}{2\pi\sigma^2}$	$G_\sigma(1,0) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$
$G_\sigma(-1,-1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$	$G_\sigma(0,-1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(1,-1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$

With $\sigma = 1$:

0.059	0.097	0.059
0.097	0.159	0.097
0.059	0.097	0.059

What happens if σ is larger?

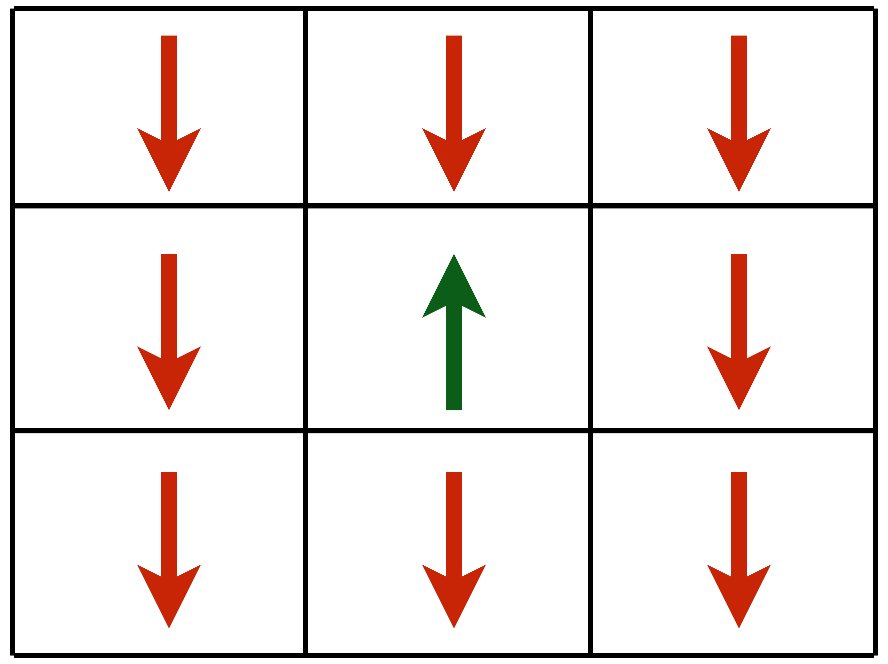
What happens if σ is smaller?

Example 6: Smoothing with a Gaussian

Quantized an truncated **3x3 Gaussian** filter:

$G_\sigma(-1,1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$	$G_\sigma(0,1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(1,1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$
$G_\sigma(-1,0) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(0,0) = \frac{1}{2\pi\sigma^2}$	$G_\sigma(1,0) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$
$G_\sigma(-1,-1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$	$G_\sigma(0,-1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(1,-1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$

With $\sigma = 1$:

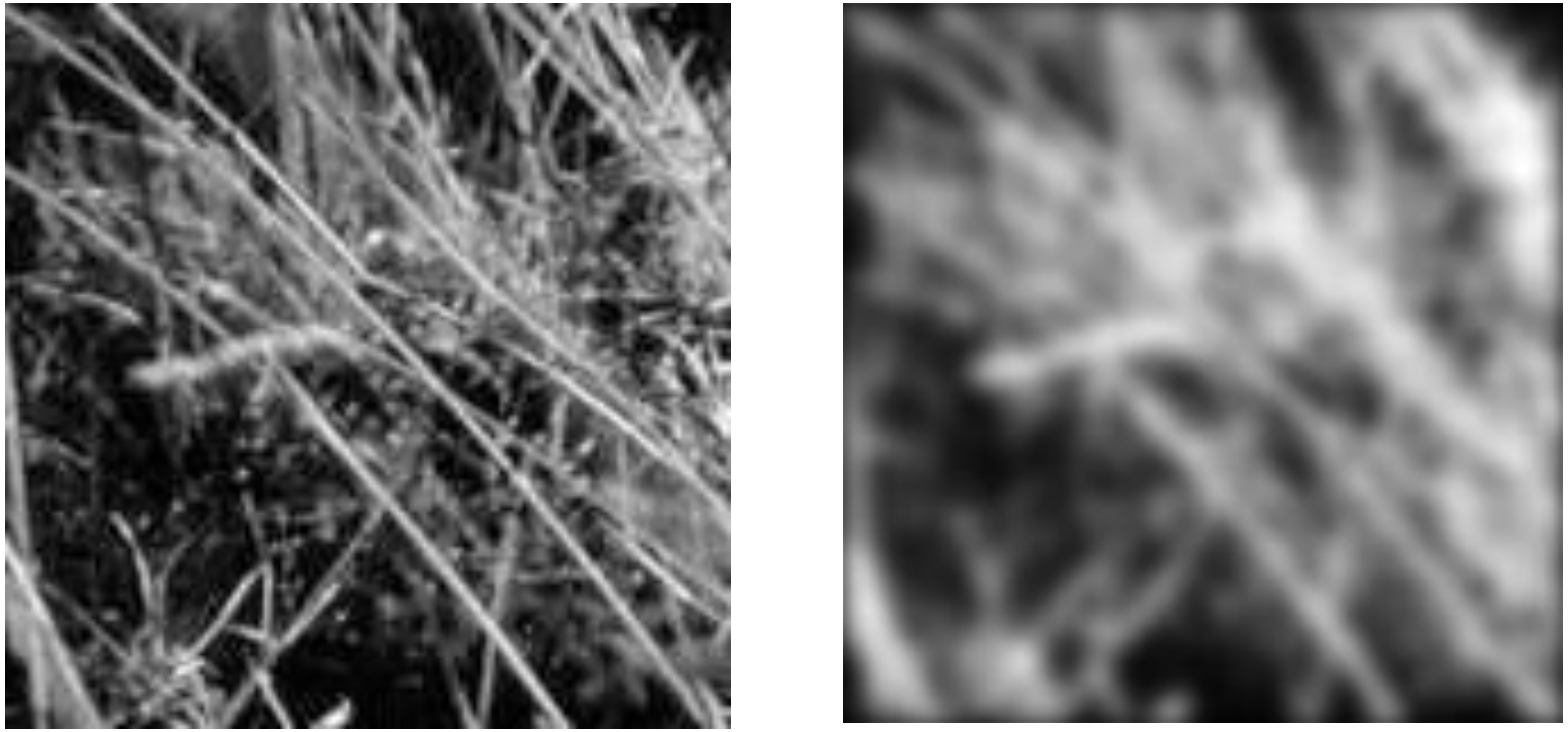


What happens if σ is larger?

What happens if σ is smaller?

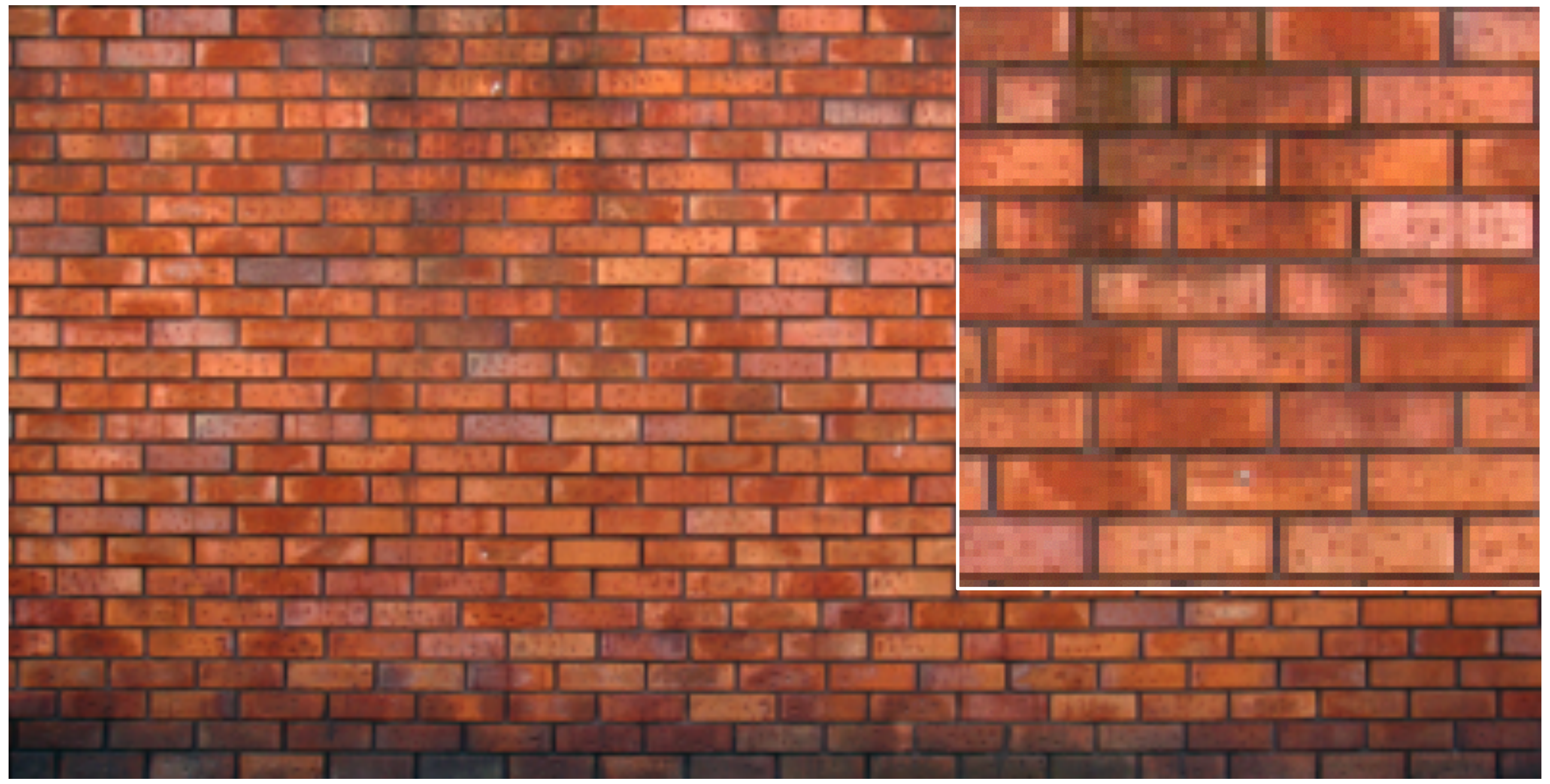
— **Less** blur

Example 6: Smoothing with a Gaussian

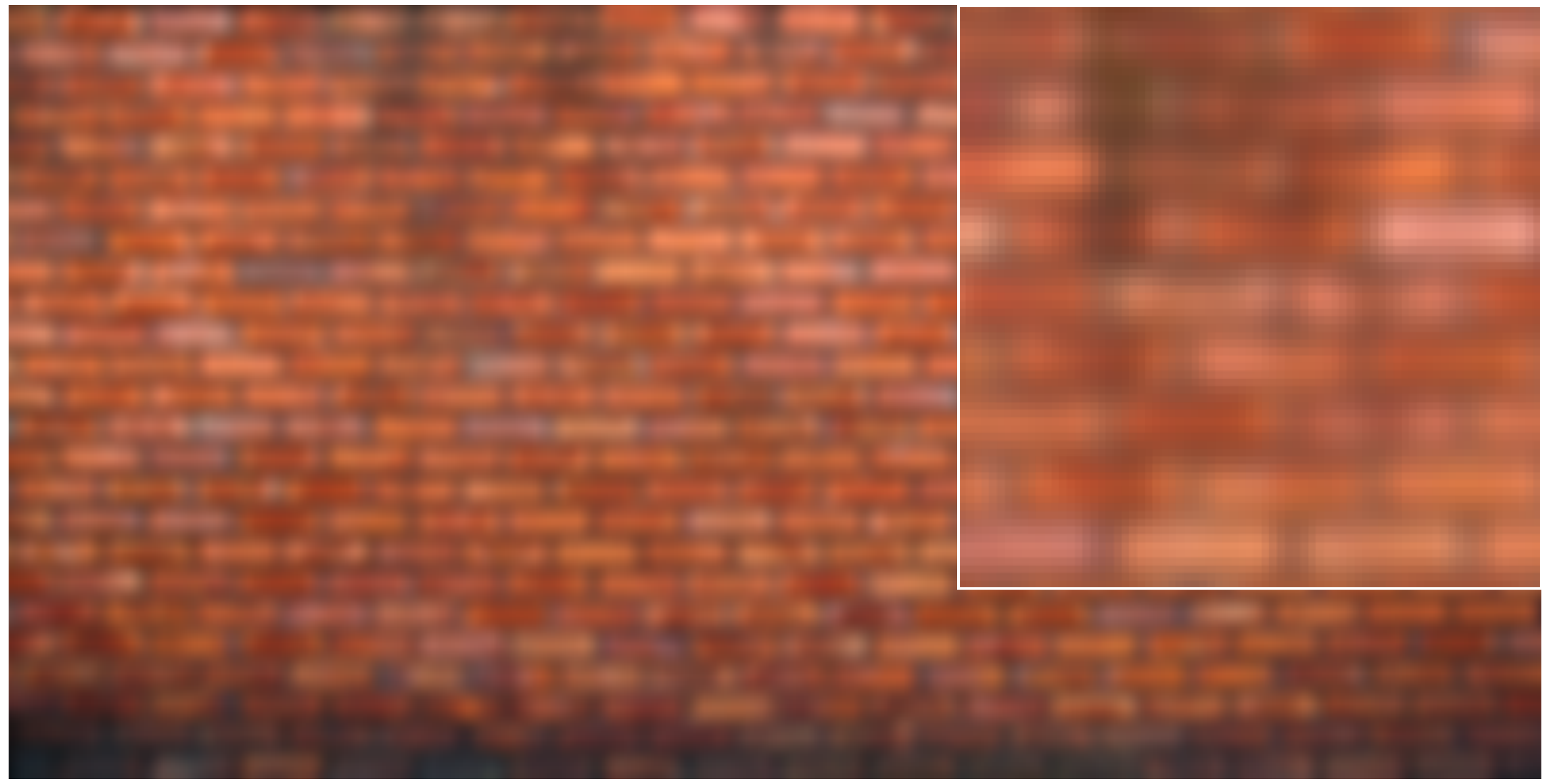


Forsyth & Ponce (2nd ed.) Figure 4.1 (left and right)

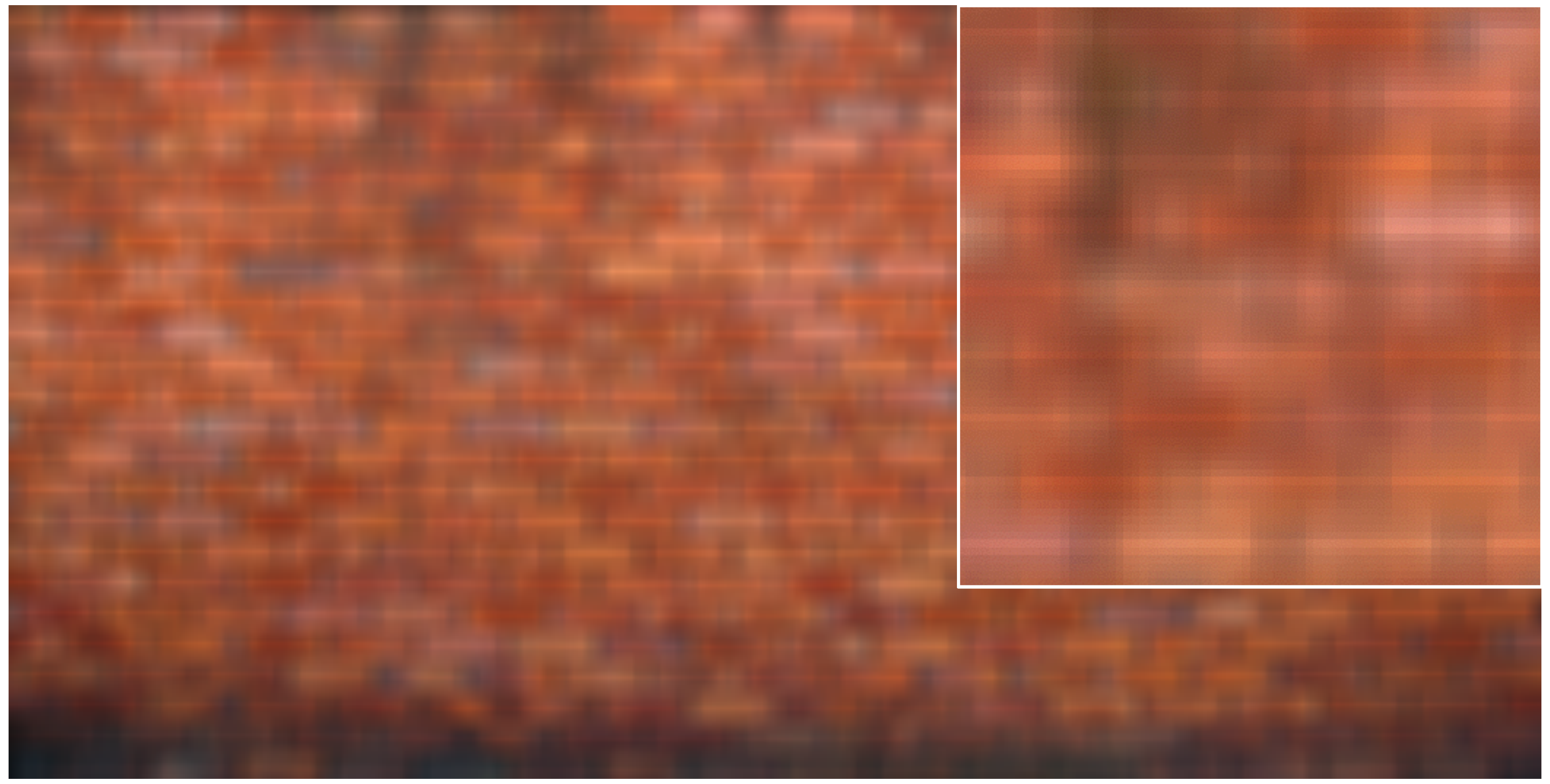
Box vs. **Gaussian** Filter



original



7x7 Gaussian



7x7 box

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Fun: How to get shadow effect?

University of British Columbia

Adopted from: Ioannis (Yannis) Gkioulekas (CMU)

Fun: How to get shadow effect?

University of British Columbia

Blur with a Gaussian kernel, then compose the blurred image with the original (with some offset)

Adopted from: Ioannis (Yannis) Gkioulekas (CMU)

Example 6: Smoothing with a Gaussian

Quantized an truncated **3x3 Gaussian** filter:

$G_\sigma(-1,1) = \frac{1}{2\pi\sigma^2} \exp^{-\frac{2}{2\sigma^2}}$	$G_\sigma(0,1) = \frac{1}{2\pi\sigma^2} \exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(1,1) = \frac{1}{2\pi\sigma^2} \exp^{-\frac{2}{2\sigma^2}}$
$G_\sigma(-1,0) = \frac{1}{2\pi\sigma^2} \exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(0,0) = \frac{1}{2\pi\sigma^2}$	$G_\sigma(1,0) = \frac{1}{2\pi\sigma^2} \exp^{-\frac{1}{2\sigma^2}}$
$G_\sigma(-1,-1) = \frac{1}{2\pi\sigma^2} \exp^{-\frac{2}{2\sigma^2}}$	$G_\sigma(0,-1) = \frac{1}{2\pi\sigma^2} \exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(1,-1) = \frac{1}{2\pi\sigma^2} \exp^{-\frac{2}{2\sigma^2}}$

With $\sigma = 1$:

0.059	0.097	0.059
0.097	0.159	0.097
0.059	0.097	0.059

What is the problem with this filter?

Example 6: Smoothing with a Gaussian

Quantized an truncated **3x3 Gaussian** filter:

$G_\sigma(-1,1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$	$G_\sigma(0,1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(1,1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$
$G_\sigma(-1,0) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(0,0) = \frac{1}{2\pi\sigma^2}$	$G_\sigma(1,0) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$
$G_\sigma(-1,-1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$	$G_\sigma(0,-1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(1,-1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$

With $\sigma = 1$:

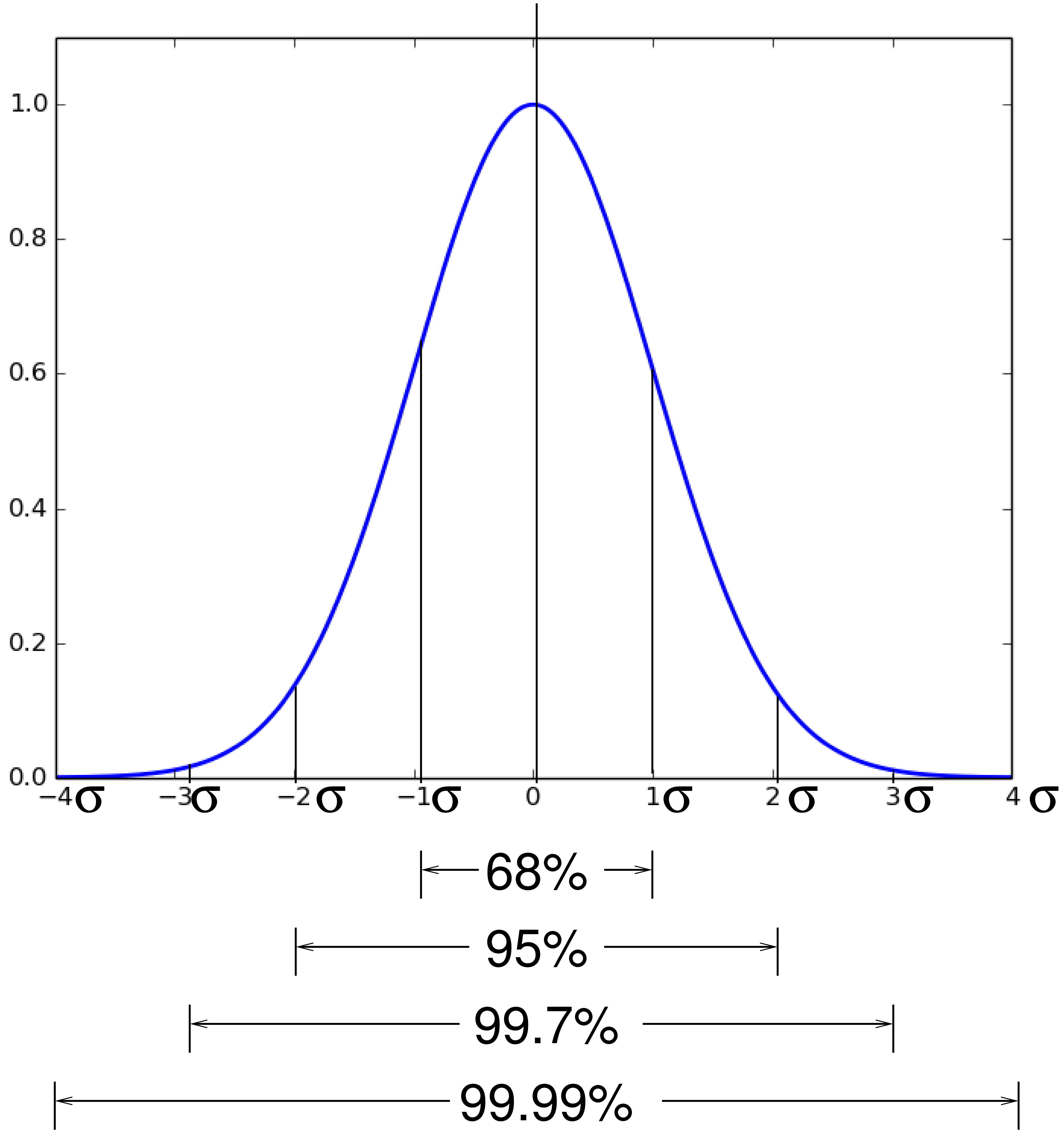
0.059	0.097	0.059
0.097	0.159	0.097
0.059	0.097	0.059

What is the problem with this filter?

does not sum to 1

truncated too much

Gaussian: Area Under the Curve



Example 6: Smoothing with a Gaussian

With $\sigma = 1$:

0.059	0.097	0.059
0.097	0.159	0.097
0.059	0.097	0.059

Better version of the Gaussian filter:

- sums to 1 (normalized)
- captures $\pm 2\sigma$

$\frac{1}{273}$

1	4	7	4	1
4	16	26	16	4
7	26	41	26	7
4	16	26	16	4
1	4	7	4	1

In general, you want the Gaussian filter to capture $\pm 3\sigma$, for $\sigma = 1$ => 7x7 filter

Efficient Implementation: **Separability**

A 2D function of x and y is **separable** if it can be written as the product of two functions, one a function only of x and the other a function only of y

Both the 2D box filter and the 2D Gaussian filter are separable

Both can be implemented as two 1D convolutions:

— First, convolve each row with a 1D filter

— Then, convolve each column with a 1D filter

— Aside: or vice versa

The **2D Gaussian** is the only (non trivial) 2D function that is both separable and rotationally invariant.

Separability: Box Filter Example

Standard (3x3)

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$$F(X,Y) = F(X)F(Y)$$

filter

$$\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

	0	10	20	30	30	30	20	10	
	0	20	40	60	60	60	40	20	
	0	30	50	80	80	90	60	30	
	0	30	50	80	80	90	60	30	
	0	20	30	50	50	60	40	20	
	0	10	20	30	30	30	20	10	
	10	10	10	10	0	0	0	0	
	10	30	10	10	0	0	0	0	

Separable

image $I(X,Y)$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$F(X)$

filter

$$\frac{1}{3}\begin{bmatrix}1 & 1 & 1\end{bmatrix}$$

	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	
	0	30	60	90	90	90	60	30	
	0	30	60	90	90	90	60	30	
	0	30	30	60	60	90	60	30	
	0	30	60	90	90	90	60	30	
	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	
	30	30	30	30	0	0	0	0	
	0	0	0	0	0	0	0	0	

$F(Y)$

filter

$$\frac{1}{3}\begin{bmatrix}1\\ 1\\ 1\end{bmatrix}$$

output $I'(X,Y)$

	0	10	20	30	30	30	20	10	
	0	20	40	60	60	60	40	20	
	0	30	50	80	80	90	60	30	
	0	30	50	80	80	90	60	30	
	0	20	30	50	50	60	40	20	
	0	10	20	30	30	30	20	10	
	10	10	10	10	0	0	0	0	
	10	30	10	10	0	0	0	0	

Efficient Implementation: **Separability**

For example, recall the 2D **Gaussian**:

$$G_\sigma(x, y) = \frac{1}{2\pi\sigma^2} \exp^{-\frac{x^2+y^2}{2\sigma^2}}$$

The 2D Gaussian can be expressed as a product of two functions, one a function of x and another a function of y

Efficient Implementation: **Separability**

For example, recall the 2D **Gaussian**:

$$G_\sigma(x, y) = \frac{1}{2\pi\sigma^2} \exp^{-\frac{x^2+y^2}{2\sigma^2}}$$

$$= \underbrace{\boxed{\left(\frac{1}{\sqrt{2\pi}\sigma} \exp^{-\frac{x^2}{2\sigma^2}}\right)}}_{\text{function of x}} \underbrace{\boxed{\left(\frac{1}{\sqrt{2\pi}\sigma} \exp^{-\frac{y^2}{2\sigma^2}}\right)}}_{\text{function of y}}$$

The 2D Gaussian can be expressed as a product of two functions, one a function of x and another a function of y

Efficient Implementation: **Separability**

For example, recall the 2D **Gaussian**:

$$G_\sigma(x,y) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{x^2+y^2}{2\sigma^2}}$$

$$= \underbrace{\boxed{\left(\frac{1}{\sqrt{2\pi}\sigma}\exp^{-\frac{x^2}{2\sigma^2}}\right)}}_{\text{function of x}}\underbrace{\boxed{\left(\frac{1}{\sqrt{2\pi}\sigma}\exp^{-\frac{y^2}{2\sigma^2}}\right)}}_{\text{function of y}}$$

The 2D Gaussian can be expressed as a product of two functions, one a function of x and another a function of y

In this case the two functions are (identical) 1D Gaussians

Efficient Implementation: **Separability**

Naive implementation of 2D **Gaussian**:

At each pixel, (X, Y), there are	$m \times m$	multiplications
There are	$n \times n$	pixels in (X, Y)
Total:	$m^2 \times n^2$	multiplications

Efficient Implementation: **Separability**

Naive implementation of 2D **Gaussian**:

At each pixel, (X, Y), there are $m \times m$ multiplications

There are $n \times n$ pixels in (X, Y)

Total: $m^2 \times n^2$ multiplications

Separable 2D **Gaussian**:

Efficient Implementation: **Separability**

Naive implementation of 2D **Gaussian**:

At each pixel, (X,Y), there are $m \times m$ multiplications

There are $n \times n$ pixels in (X,Y)

Total: $m^2 \times n^2$ multiplications

Separable 2D **Gaussian**:

At each pixel, (X,Y), there are $2m$ multiplications

There are $n \times n$ pixels in (X,Y)

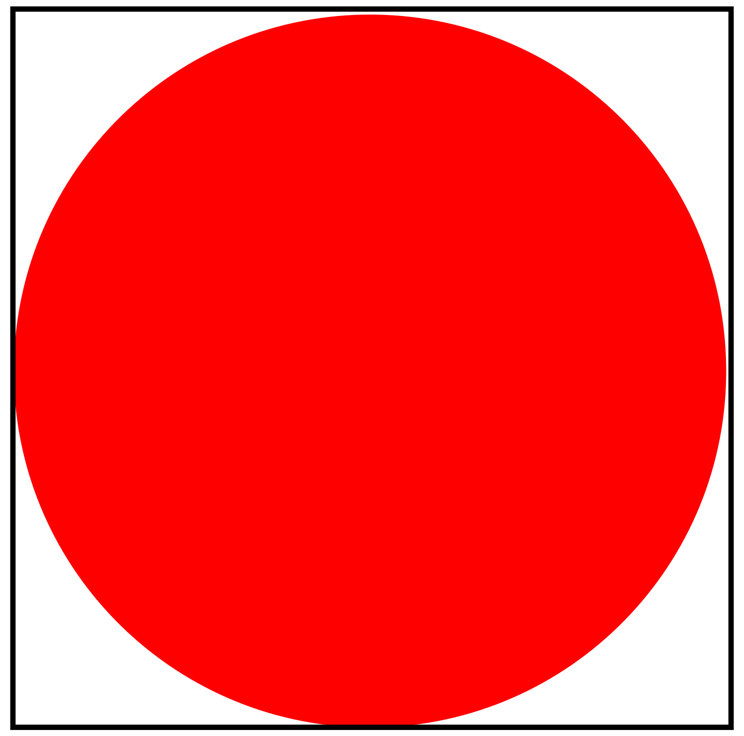
Total: $2m \times n^2$ multiplications

Example 7: Smoothing with a Pillbox

Let the radius (i.e., half diameter) of the filter be r

In a contentious domain, a 2D (circular) pillbox filter, $f(x, y)$, is defined as:

$$f(x,y) = \frac{1}{\pi r^2} \begin{cases} 1 & \text{if } \; x^2 + y^2 \leq r^2 \\ 0 & \text{otherwise} \end{cases}$$



The scaling constant, $\frac{1}{\pi r^2}$, ensures that the area of the filter is one

Example 7: Smoothing with a Pillbox

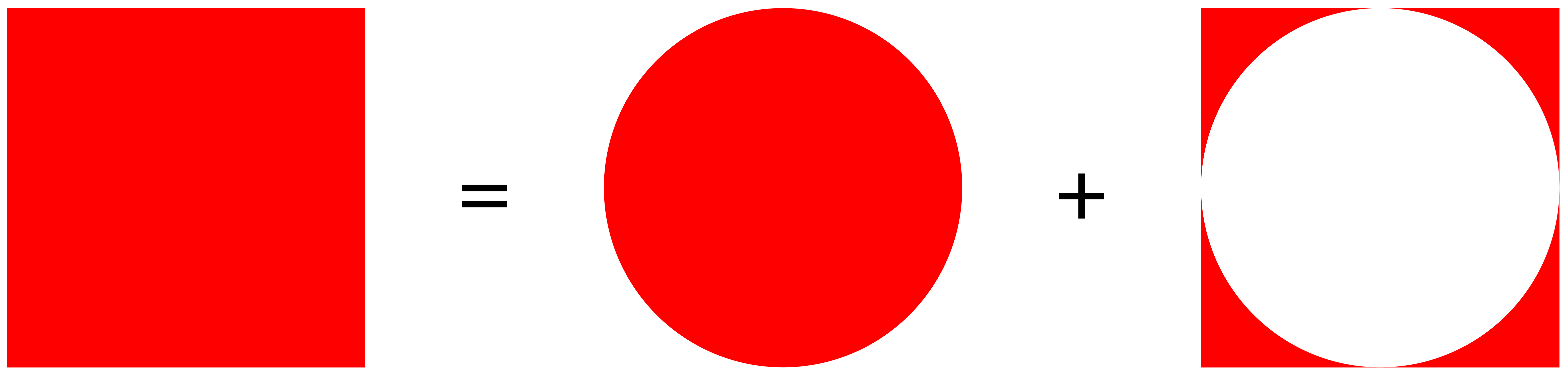
Recall that the 2D Gaussian is the only (non trivial) 2D function that is both **separable** and **rotationally invariant**.

A **2D pillbox** is rotationally invariant but not separable.

There are occasions when we want to convolve an image with a 2D pillbox. Thus, it worth exploring possibilities for **efficient implementation**.

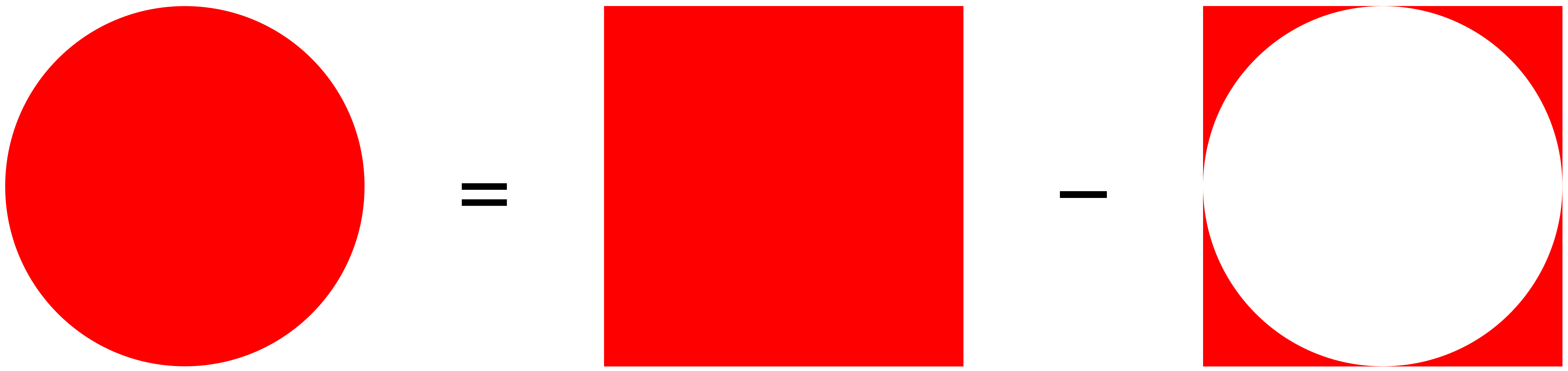
Example 7: Smoothing with a Pillbox

A 2D box filter can be expressed as the sum of a 2D pillbox and some “extra corner bits”

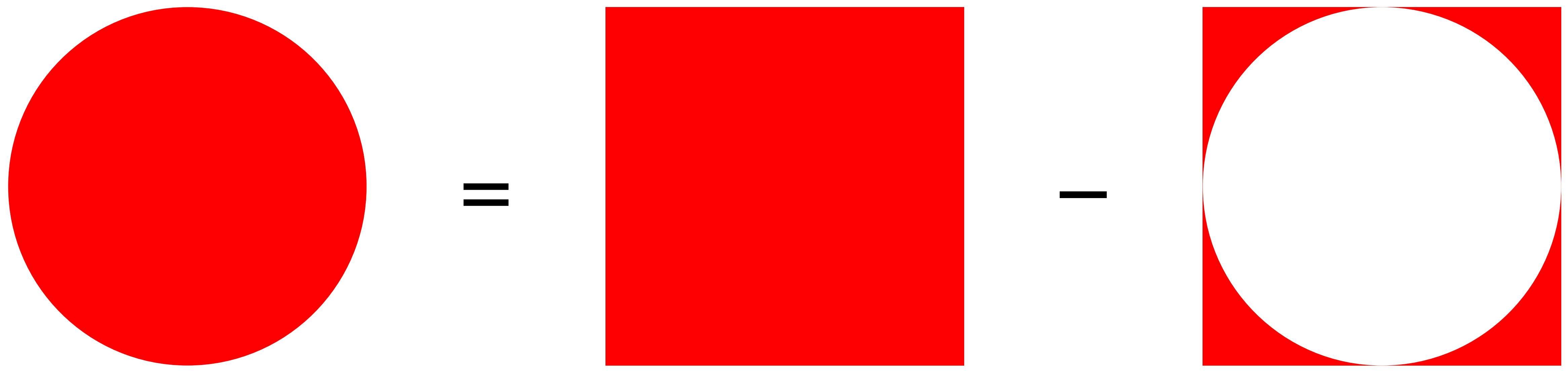


Example 7: Smoothing with a Pillbox

Therefore, a 2D pillbox filter can be expressed as the difference of a 2D box filter and those same “extra corner bits”



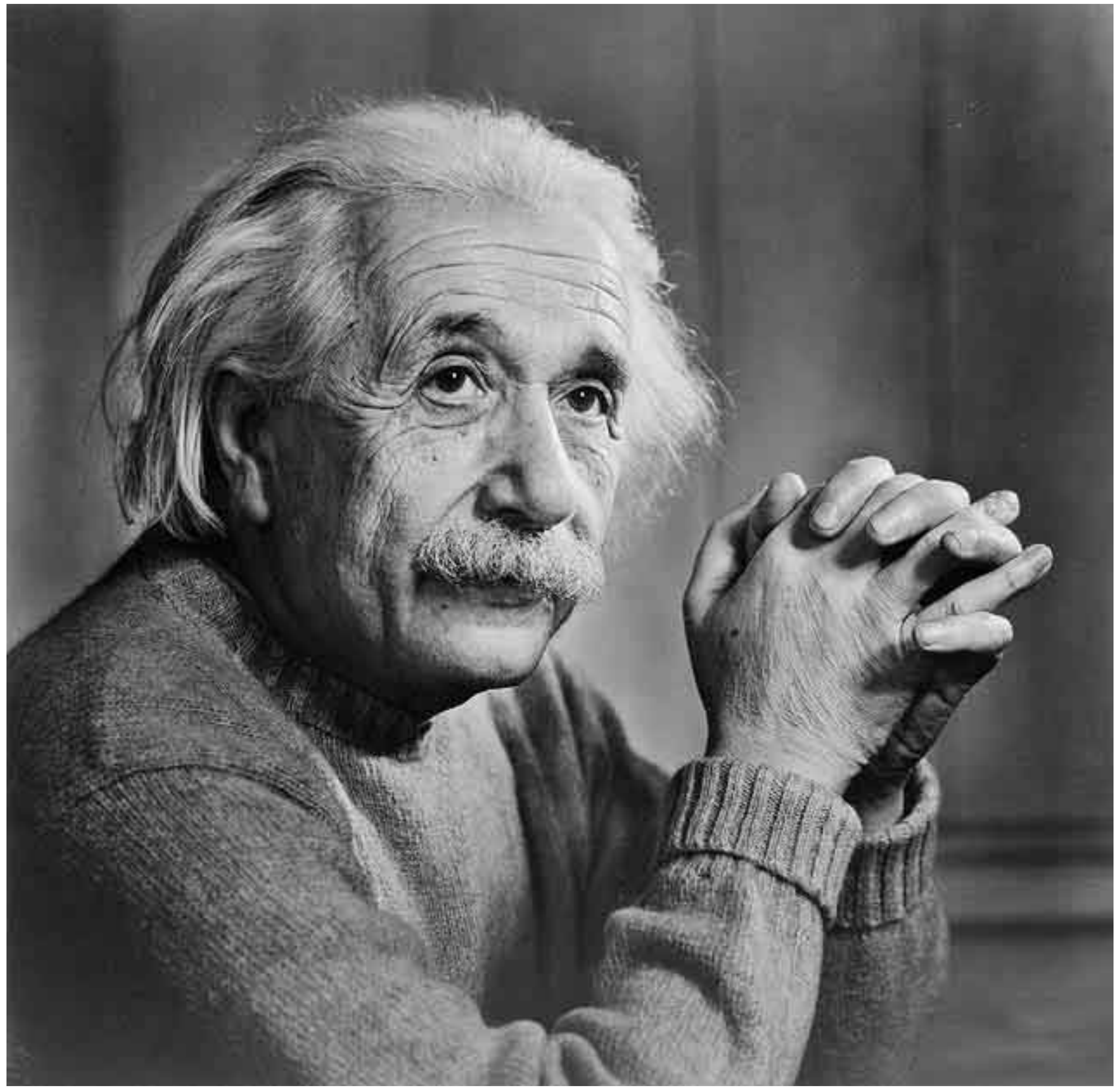
Example 7: Smoothing with a Pillbox



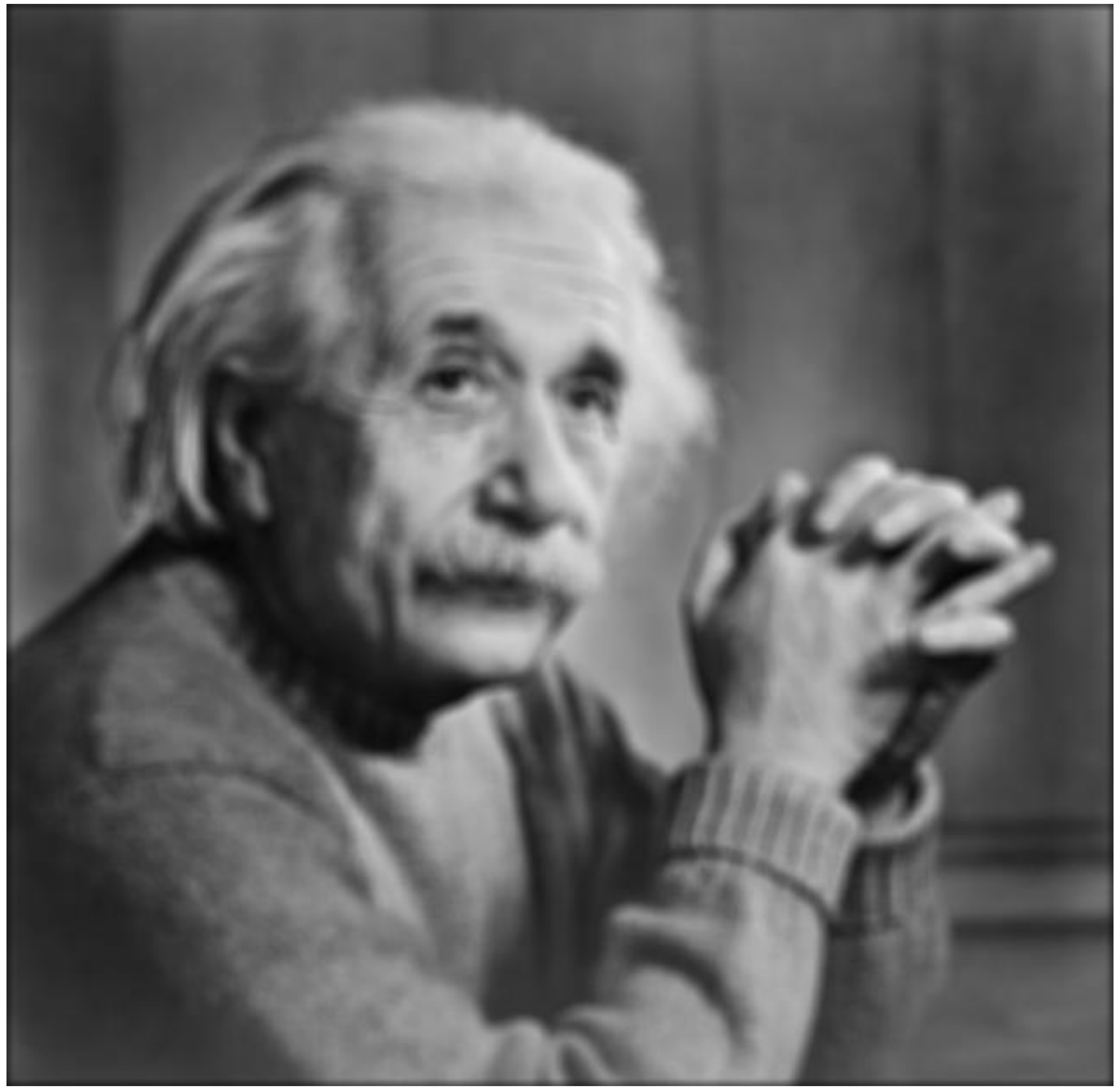
Implementing convolution with a 2D pillbox filter as the difference between convolution with a box filter and convolution with the “extra corner bits” filter allows us to take advantage of the separability of a box filter

Further, we can postpone scaling the output to a single, final step so that convolution involves filters containing all 0’s and 1’s
— This means the required convolutions can be implemented without any multiplication at all

Example 7: Smoothing with a Pillbox



Original



11 x 11 Pillbox

Speeding Up **Convolution** (The Convolution Theorem)

Let z be the product of two numbers, x and y, that is,

$$z = xy$$

Speeding Up **Convolution** (The Convolution Theorem)

Let z be the product of two numbers, x and y, that is,

$$z = xy$$

Taking logarithms of both sides, one obtains

$$\ln z = \ln x + \ln y$$

Speeding Up **Convolution** (The Convolution Theorem)

Let z be the product of two numbers, x and y, that is,

$$z = xy$$

Taking logarithms of both sides, one obtains

$$\ln z = \ln x + \ln y$$

Therefore.

$$z = \exp^{\ln z} = \exp^{(\ln x + \ln y)}$$

Speeding Up **Convolution** (The Convolution Theorem)

Let z be the product of two numbers, x and y, that is,

$$z = xy$$

Taking logarithms of both sides, one obtains

$$\ln z = \ln x + \ln y$$

Therefore.

$$z = \exp^{\ln z} = \exp^{(\ln x + \ln y)}$$

Interpretation: At the expense of two $\ln()$ and one $\exp()$ computations, multiplication is reduced to admission

Speeding Up **Rotation**

Another analogy: **2D rotation of a point by an angle** α about the origin

The standard approach, in Euclidean coordinates, involves a matrix multiplication

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

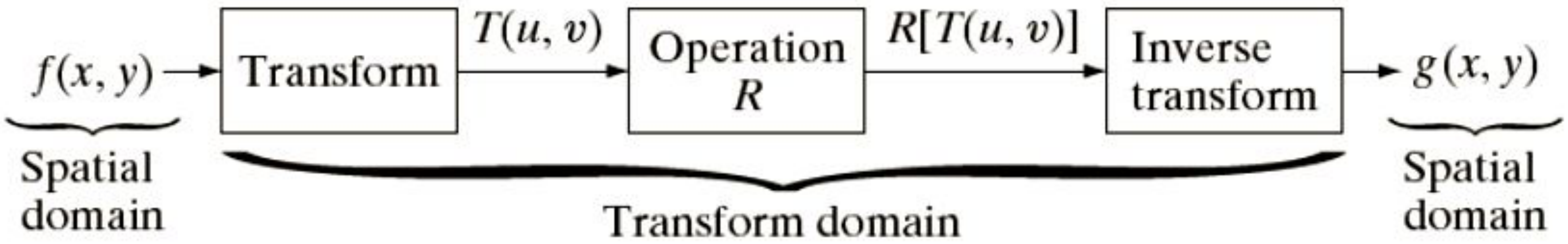
Suppose we transform to polar coordinates

$$(x, y) \rightarrow (\rho, \theta) \rightarrow (\rho, \theta + \alpha) \rightarrow (x', y')$$

Rotation becomes addition, at expense of one polar coordinate transform and one inverse polar coordinate transform

Speeding Up **Convolution** (The Convolution Theorem)

Similarly, some image processing operations become cheaper in a transform domain



Gonzales & Woods (3rd ed.) Figure 2.39

Speeding Up **Convolution** (The Convolution Theorem)

Convolution **Theorem**:

Let $i'(x, y) = f(x, y) \otimes i(x, y)$

then $\mathcal{I}'(w_x, w_y) = \mathcal{F}(w_x, w_y)\ \mathcal{I}(w_x, w_y)$

where $\mathcal{I}'(w_x, w_y)$, $\mathcal{F}(w_x, w_y)$, and $\mathcal{I}(w_x, w_y)$ are Fourier transforms of $i'(x, y)$, $f(x, y)$ and $i(x, y)$

At the expense of two **Fourier** transforms and one inverse Fourier transform, convolution can be reduced to (complex) multiplication

Existential **Choice**

Fourier Transform (you will **NOT** be tested on this)

Basic building block:

$$A \sin(\omega x + \phi)$$

Fourier's claim: Add enough of these to get <u>any</u> periodic signal you want!

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Fourier Transform (you will **NOT** be tested on this)

Basic building block:

$$A \sin(\omega x + \phi)$$

amplitude

sinusoid

angular frequency

variable

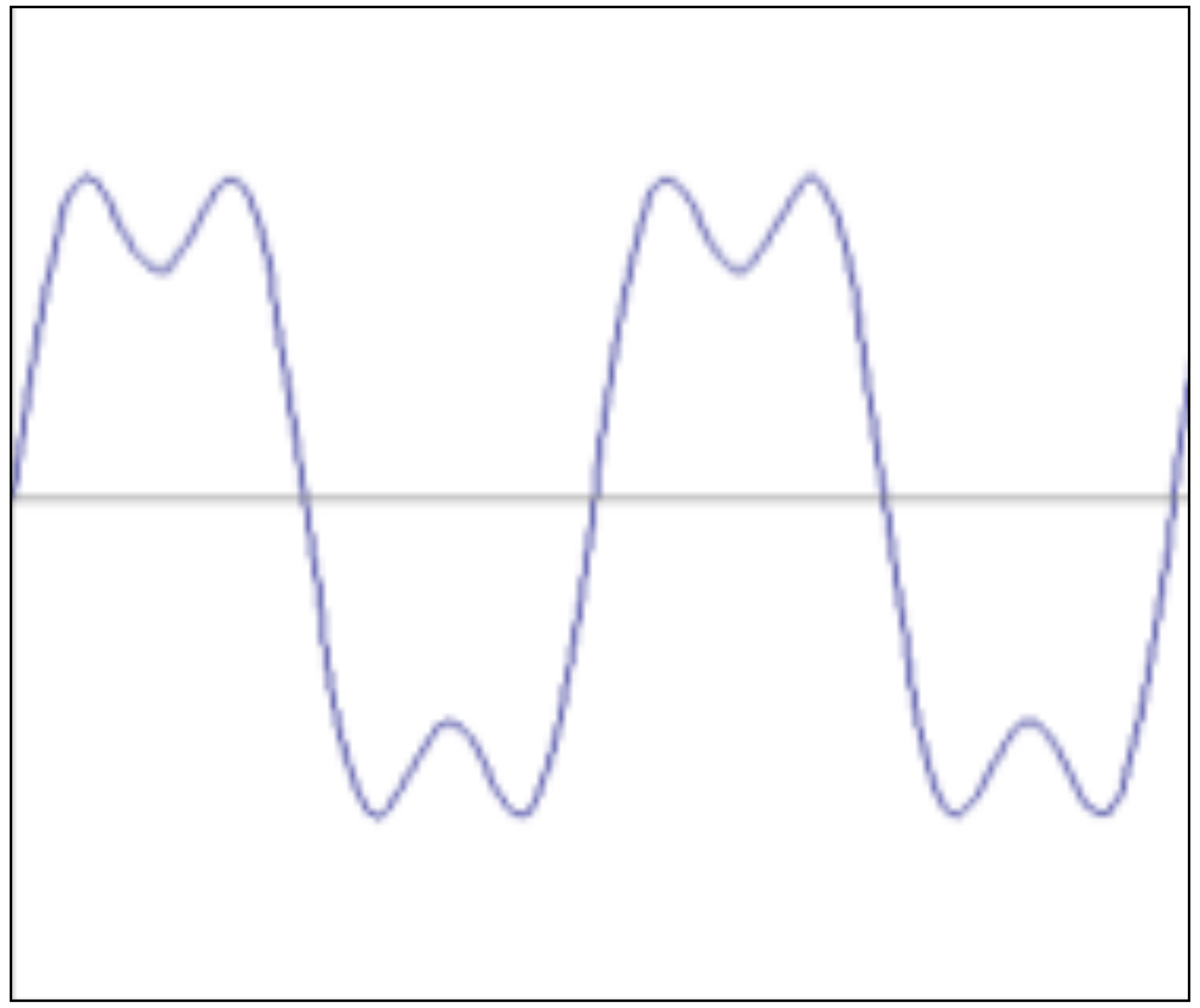
phase

Fourier's claim: Add enough of these to get <u>any</u> periodic signal you want!

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Fourier Transform (you will **NOT** be tested on this)

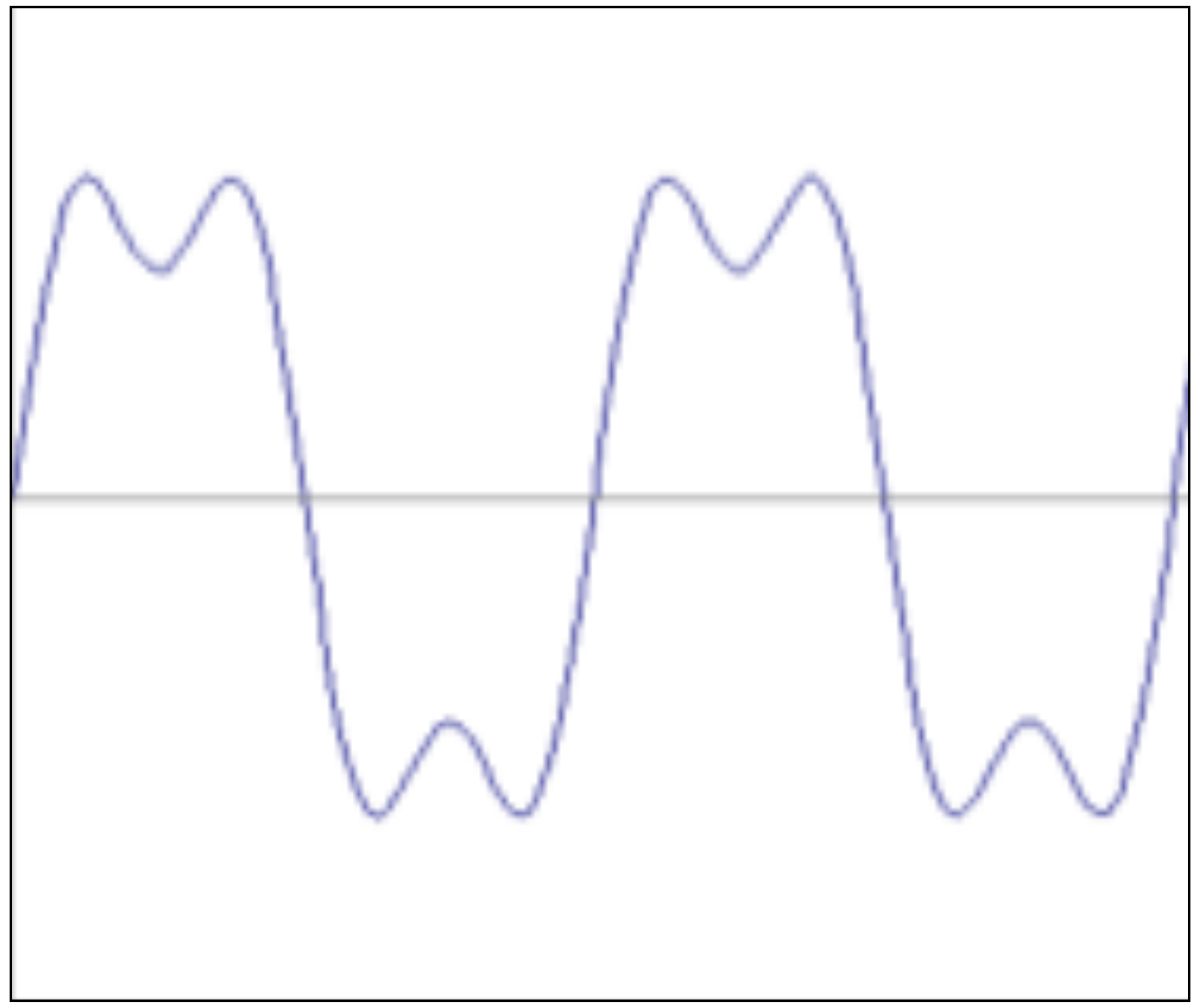
How would you generate this function?

 = ? + ?

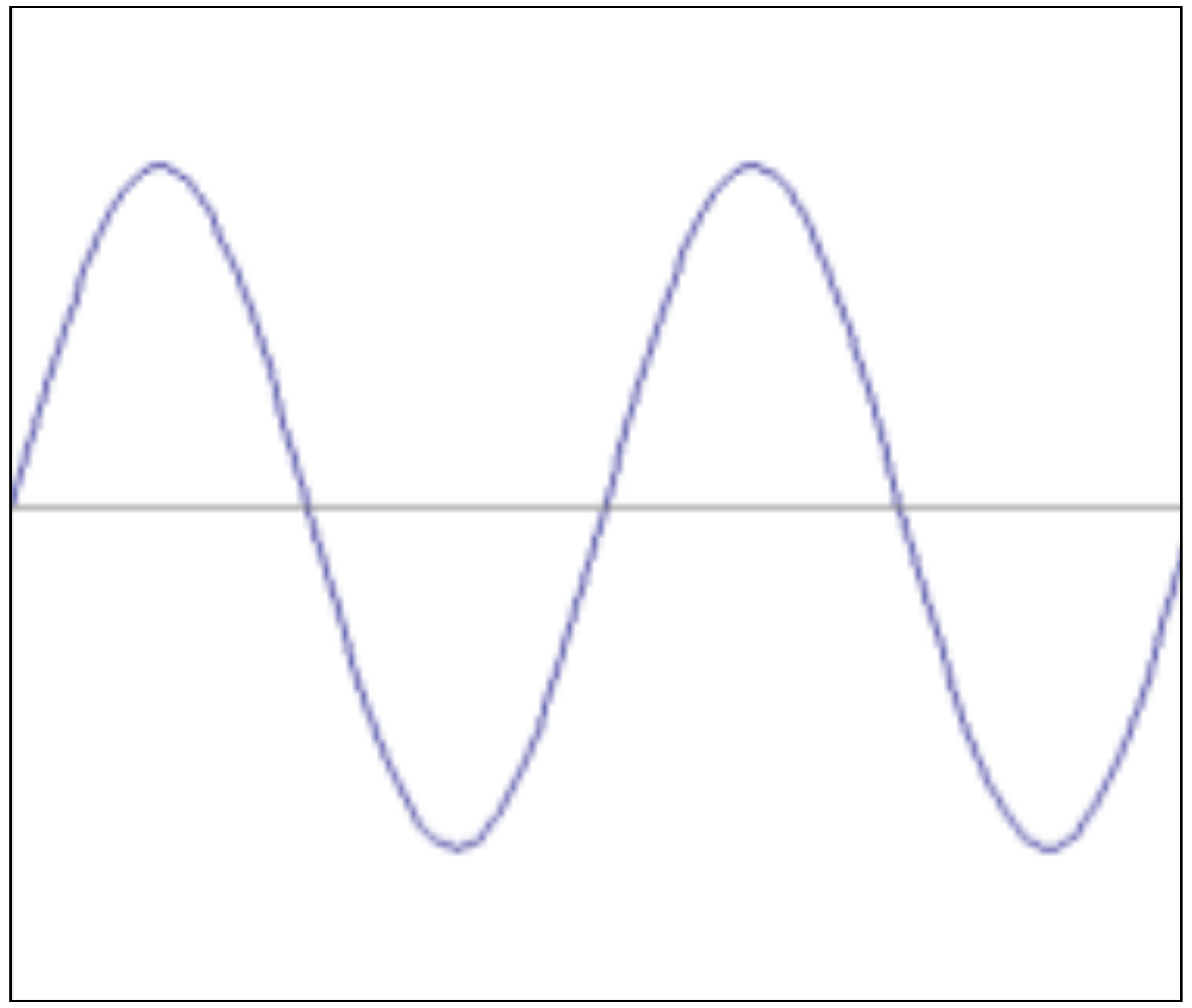
Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Fourier Transform (you will **NOT** be tested on this)

How would you generate this function?



=



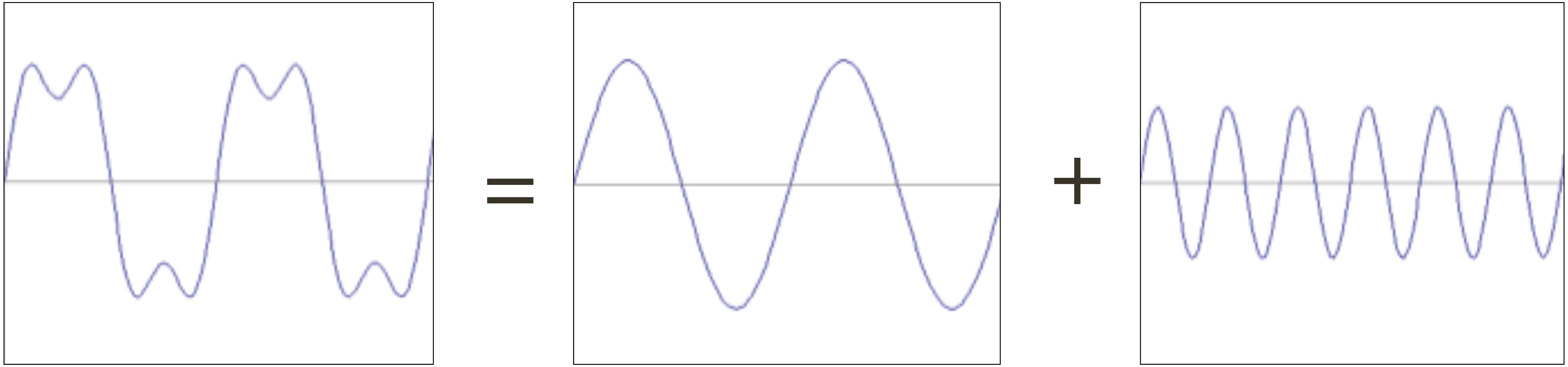
$\sin(2\pi x)$

+ ?

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Fourier Transform (you will **NOT** be tested on this)

How would you generate this function?



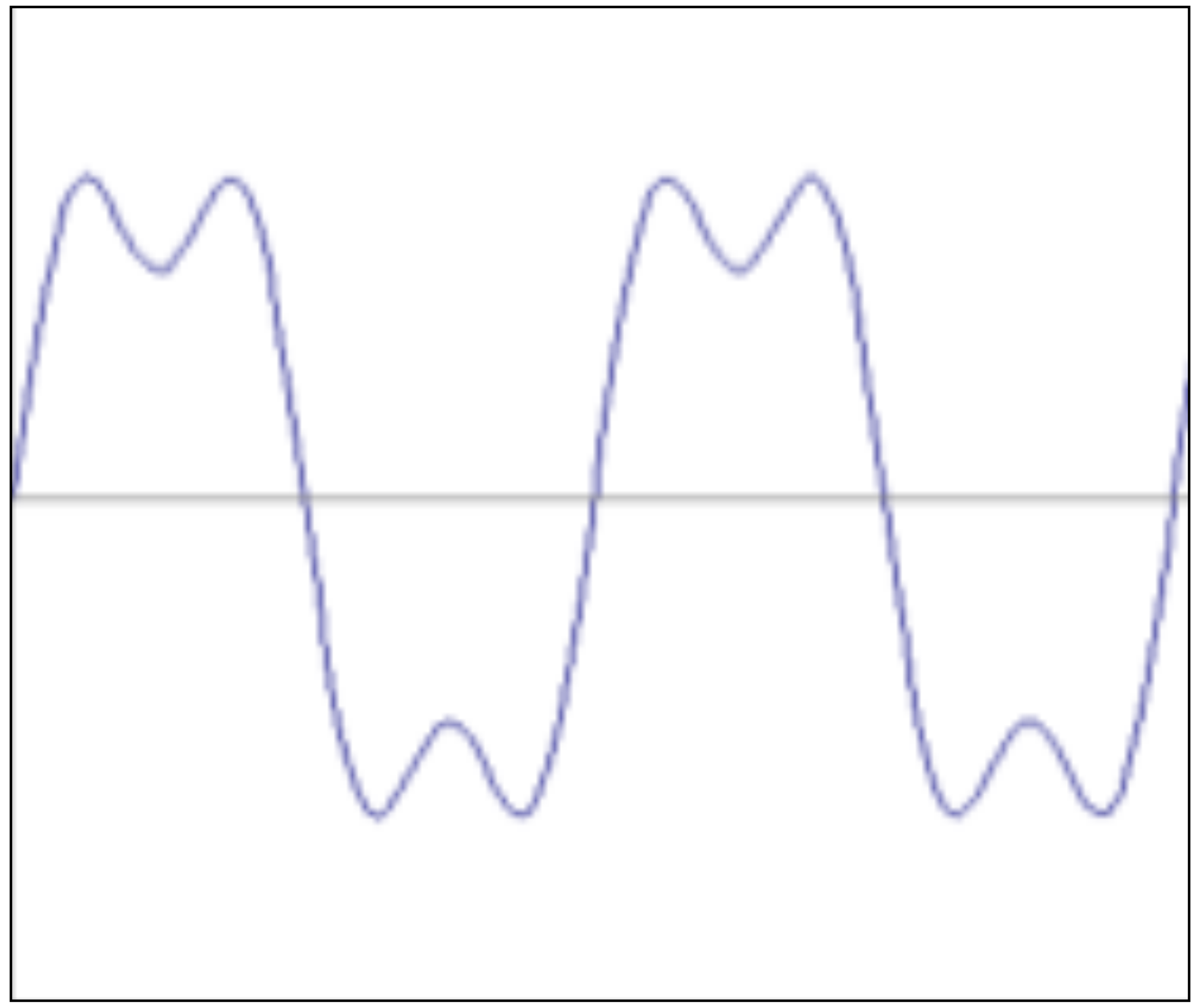
$\sin(2\pi x)$

$\frac{1}{3}\sin(2\pi 3x)$

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

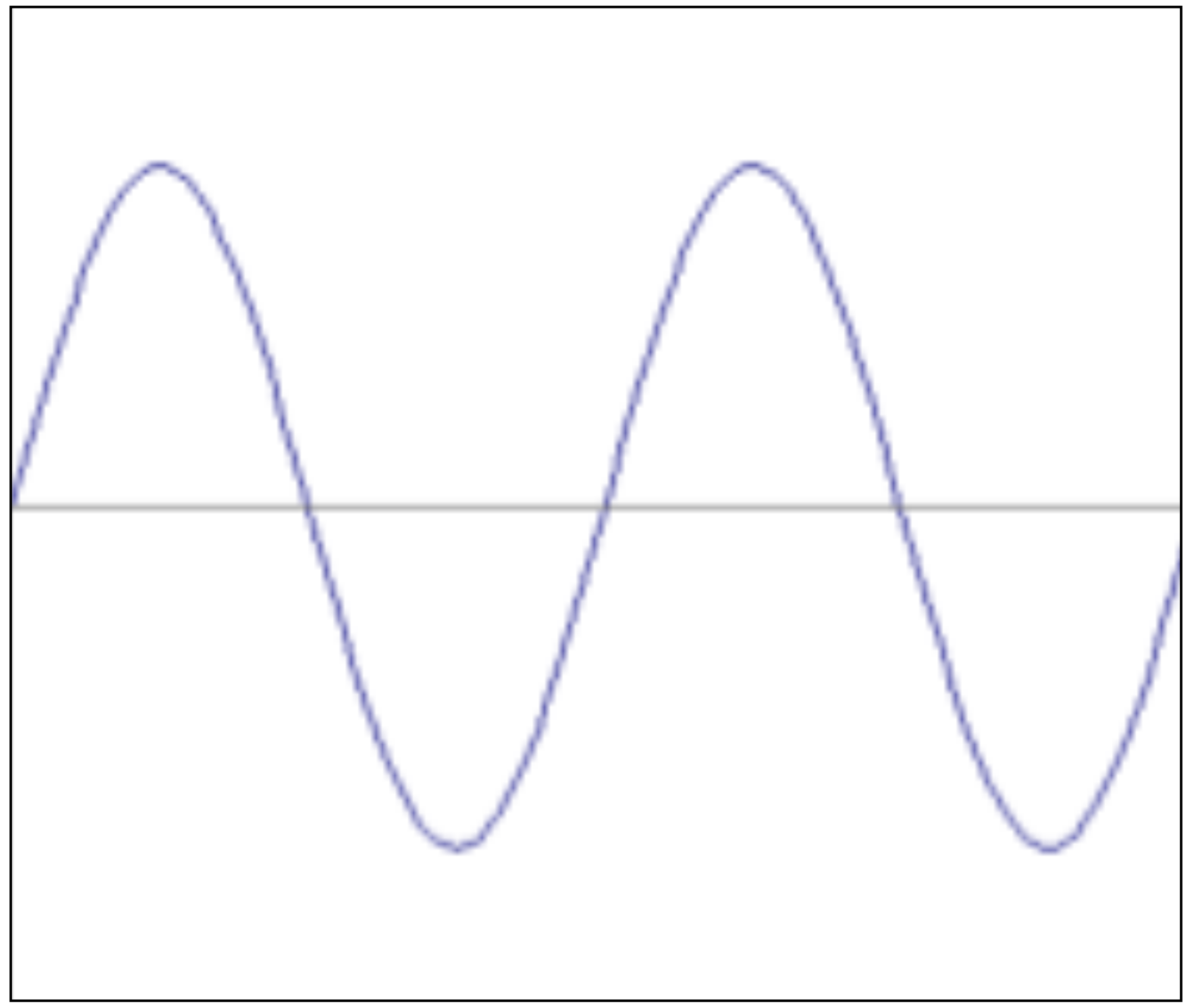
Fourier Transform (you will **NOT** be tested on this)

How would you generate this function?



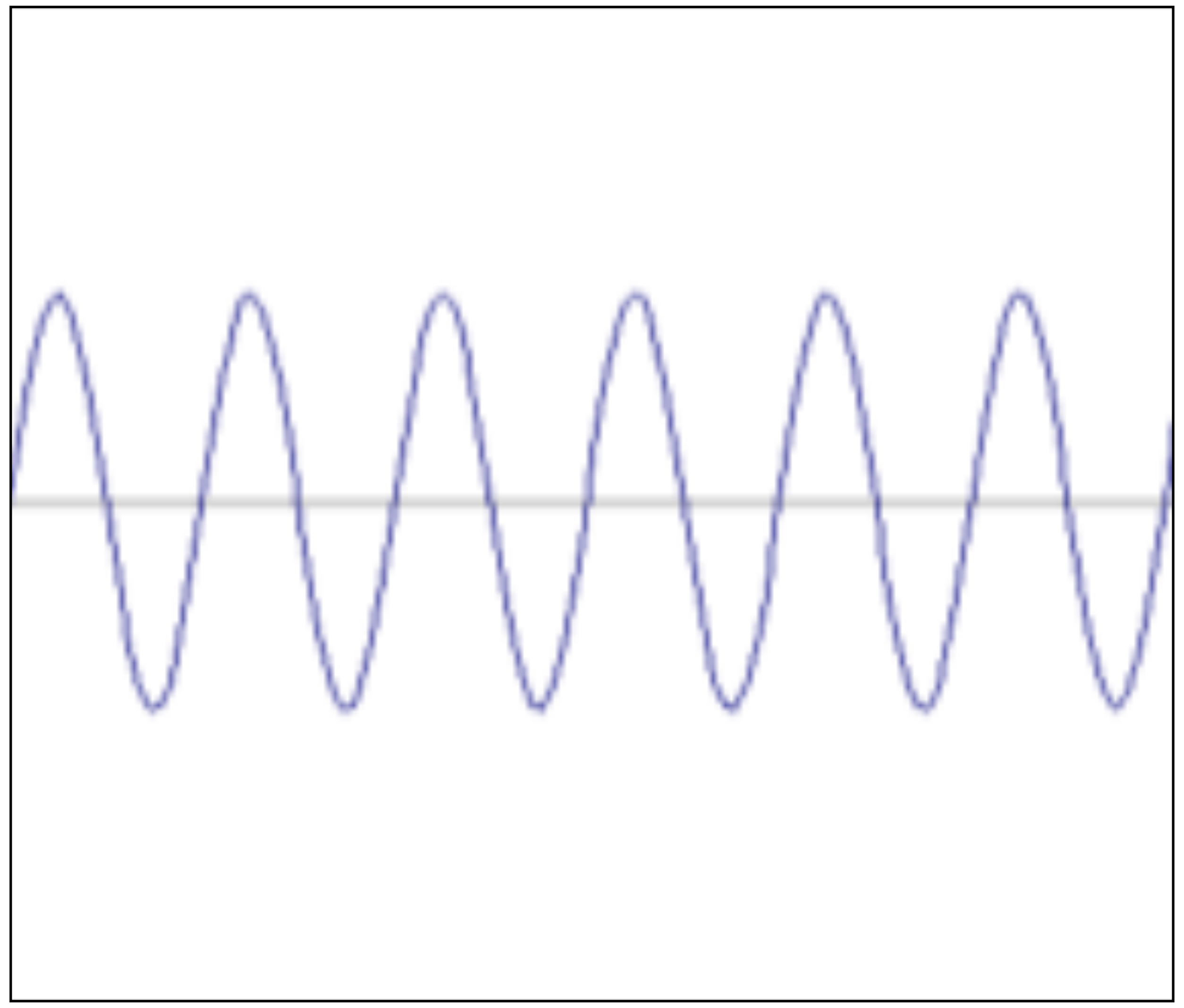
$$f(x) = \sin(2\pi x) + \frac{1}{3}\sin(2\pi 3x)$$

=



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

+

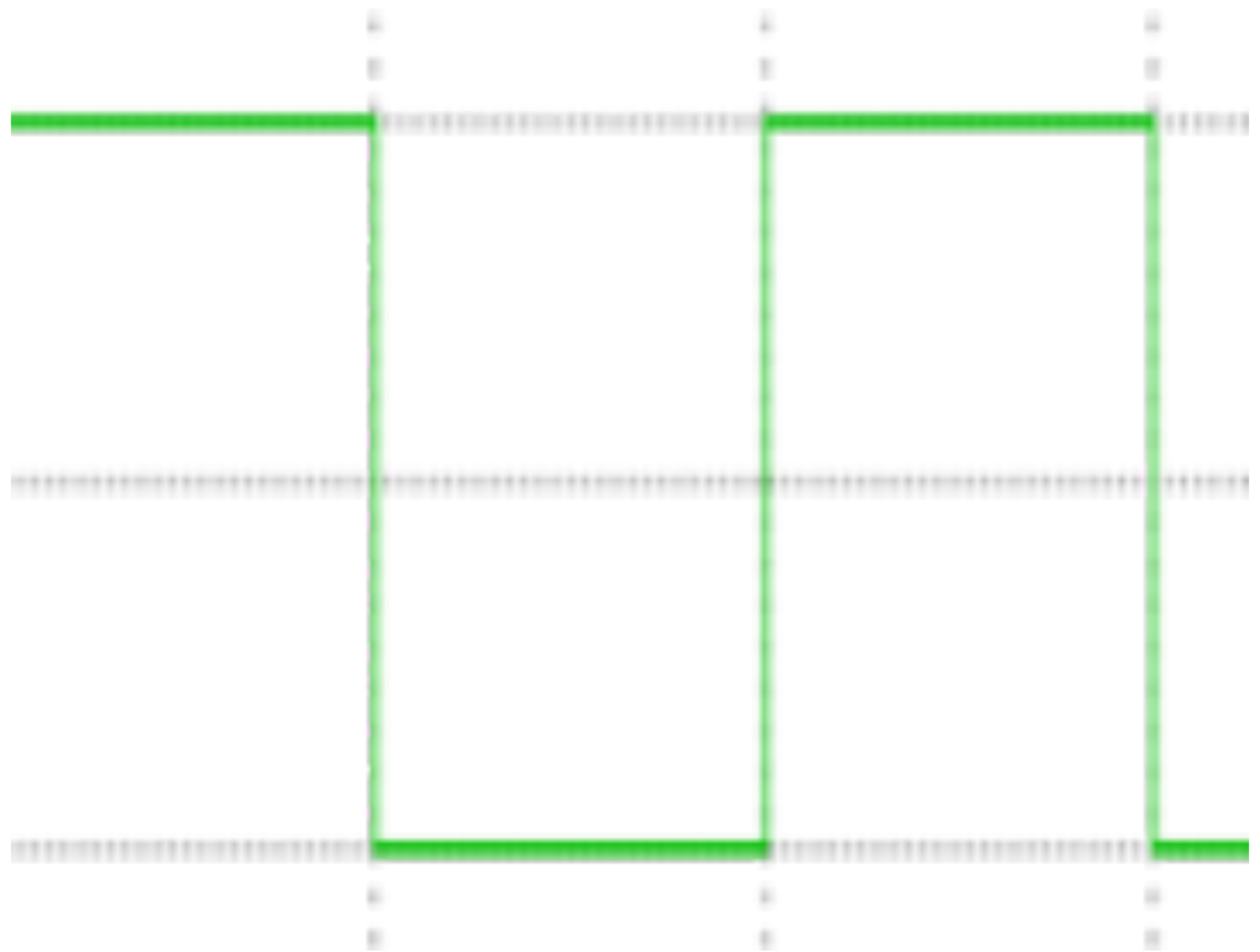


$$\frac{1}{3}\sin(2\pi 3x)$$

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Fourier Transform (you will **NOT** be tested on this)

How would you generate this function?



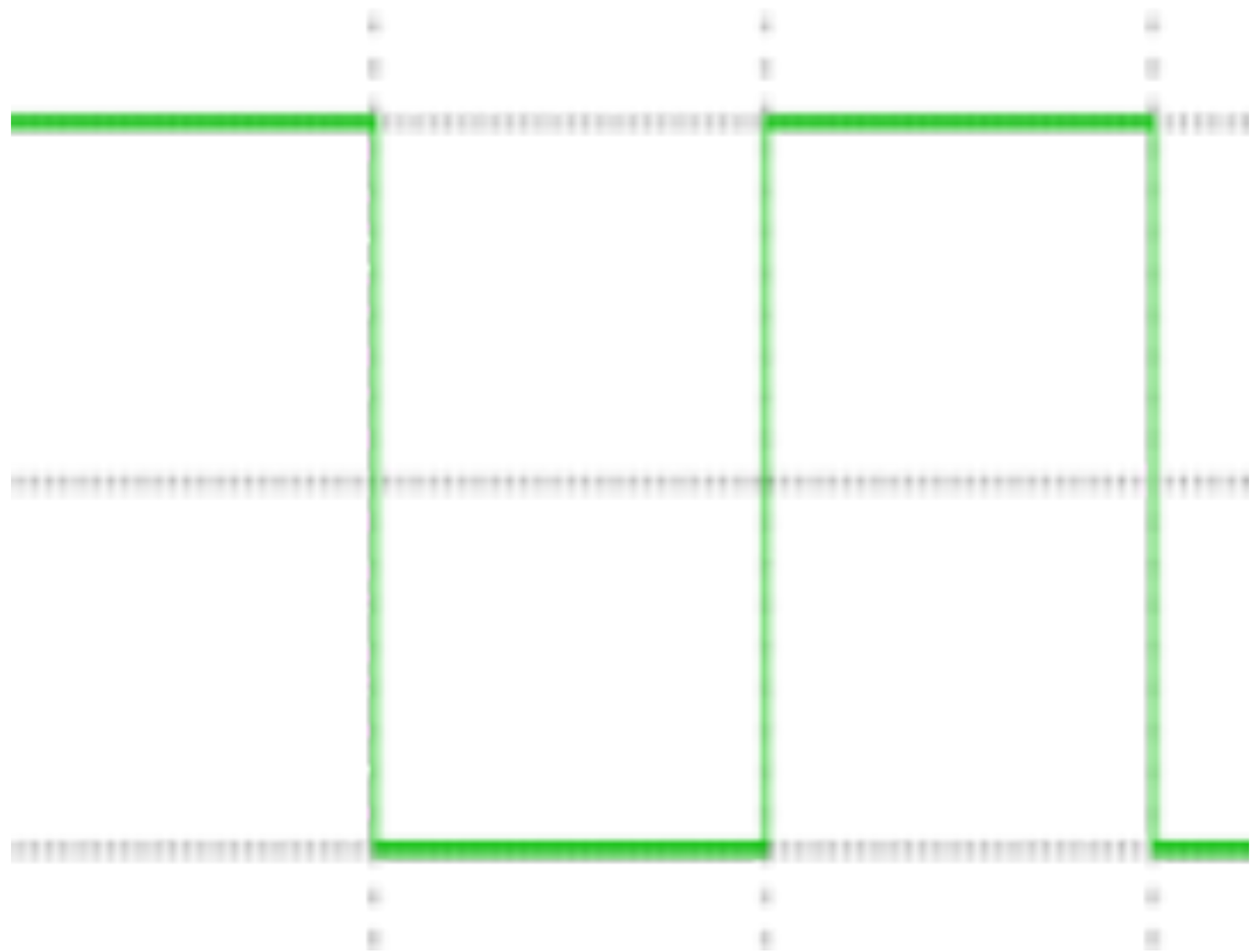
$\approx$? + ?

square wave

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

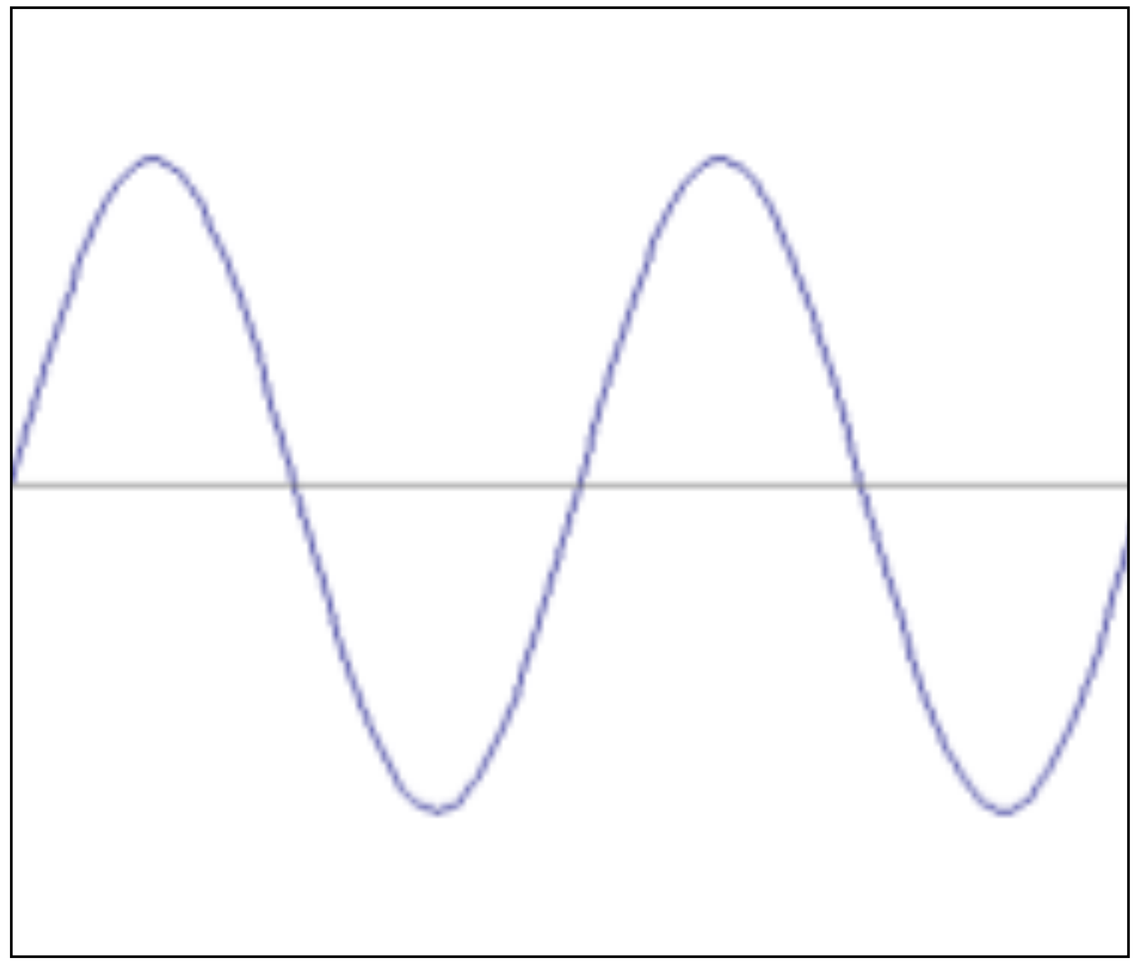
Fourier Transform (you will **NOT** be tested on this)

How would you generate this function?

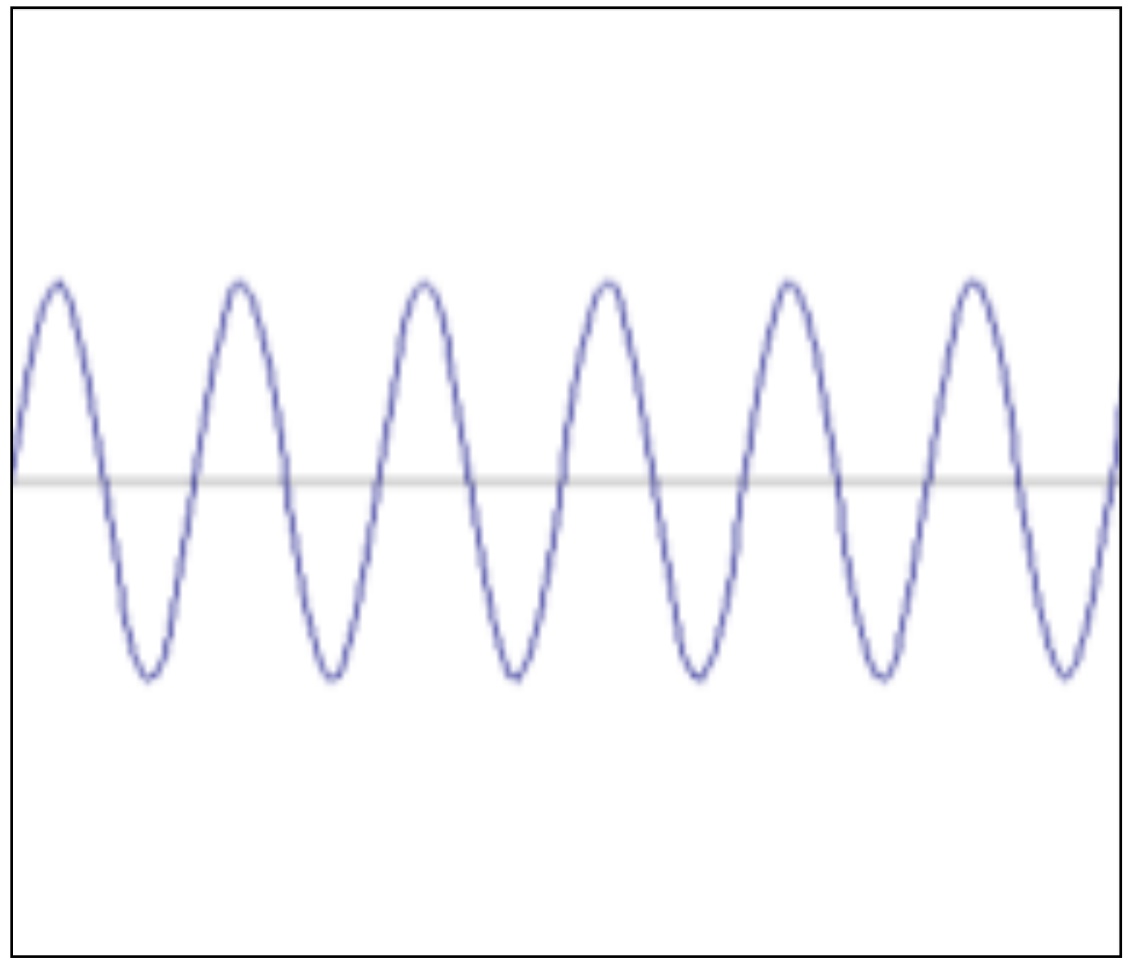


square wave

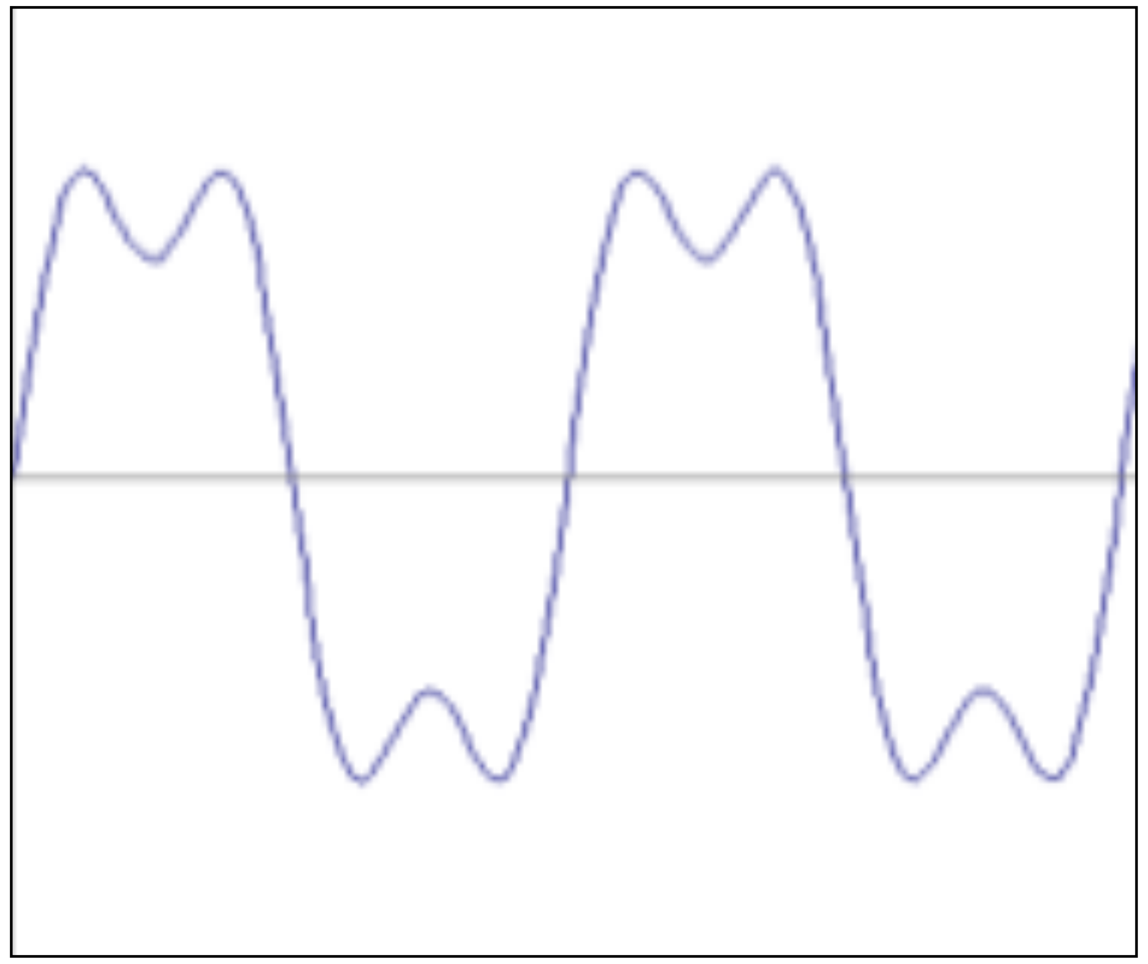
$\approx$



$+$



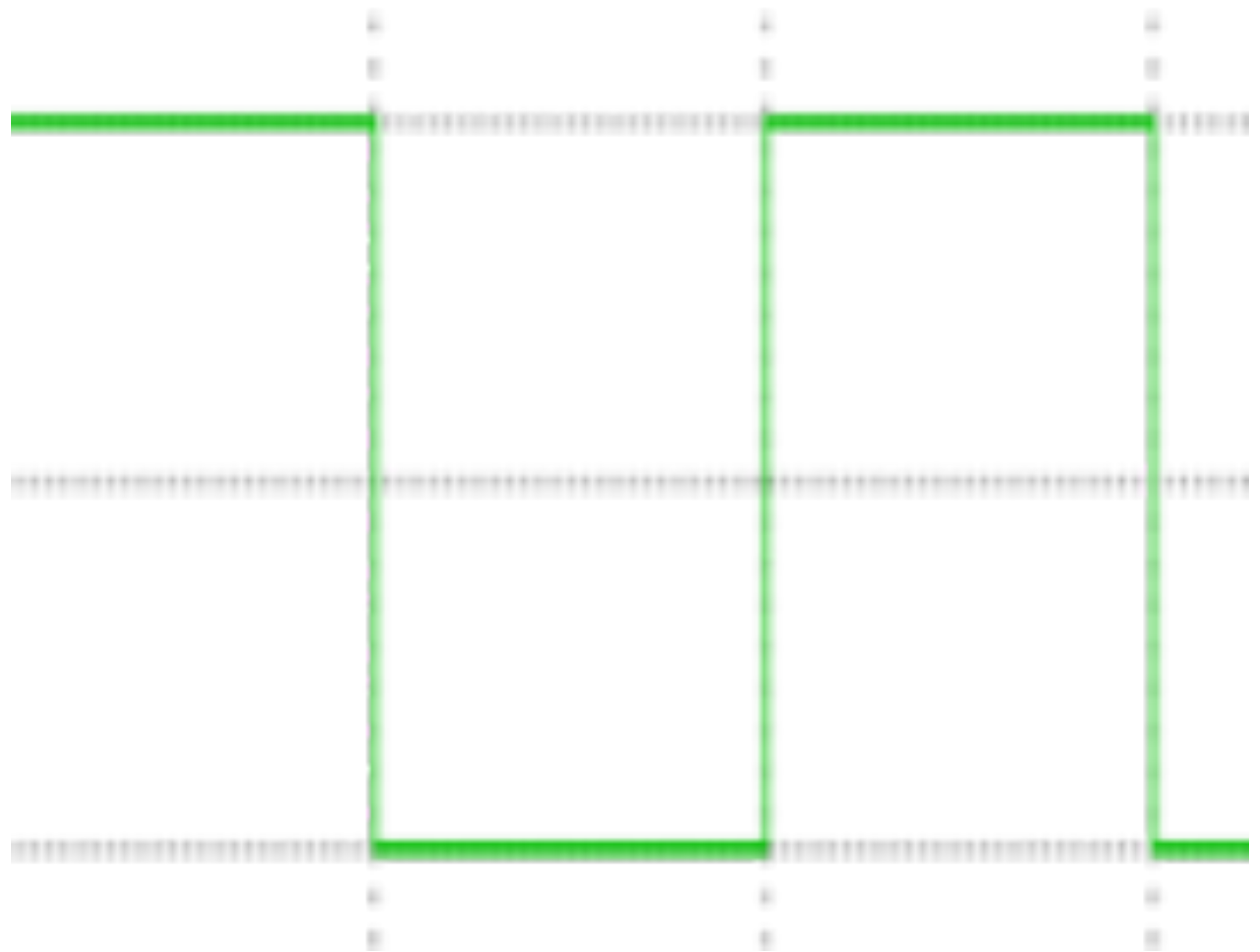
$=$



Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

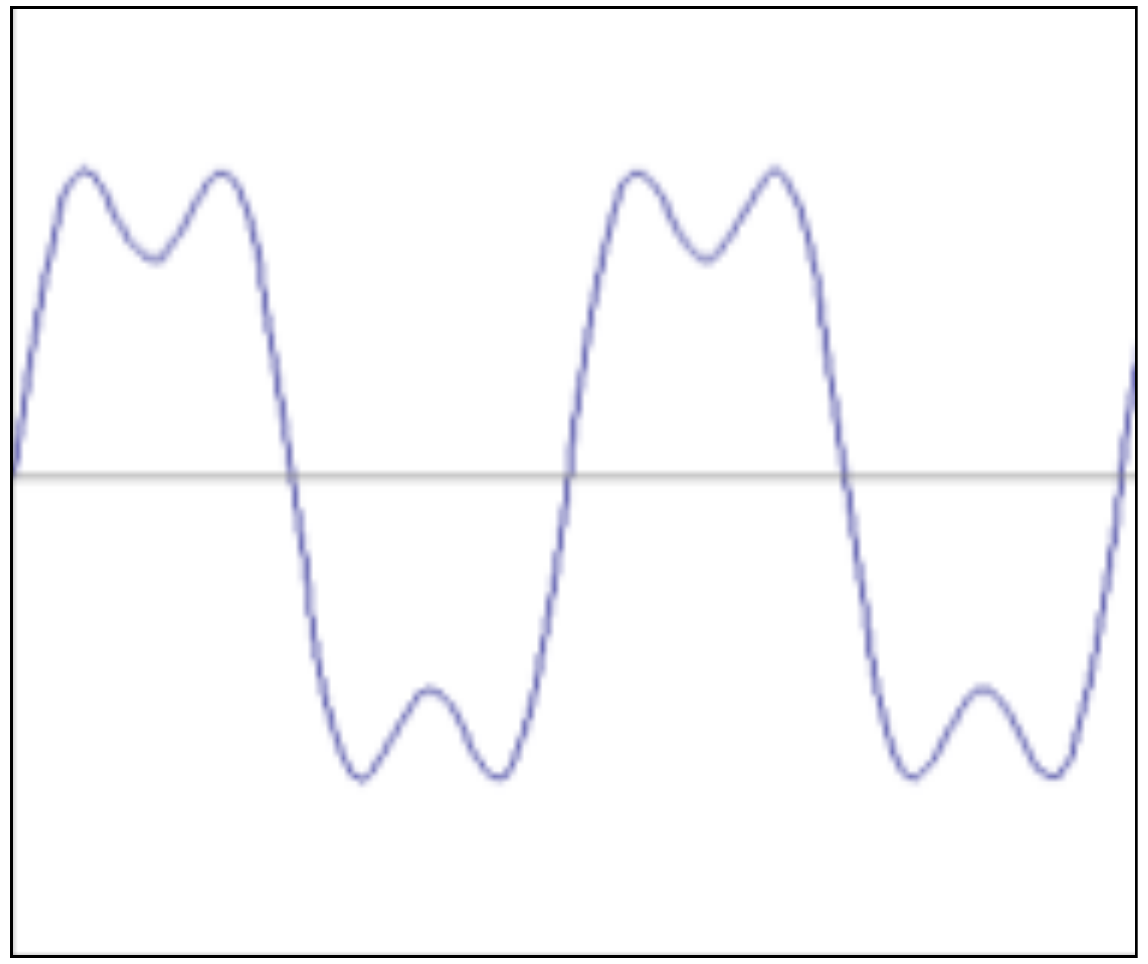
Fourier Transform (you will **NOT** be tested on this)

How would you generate this function?

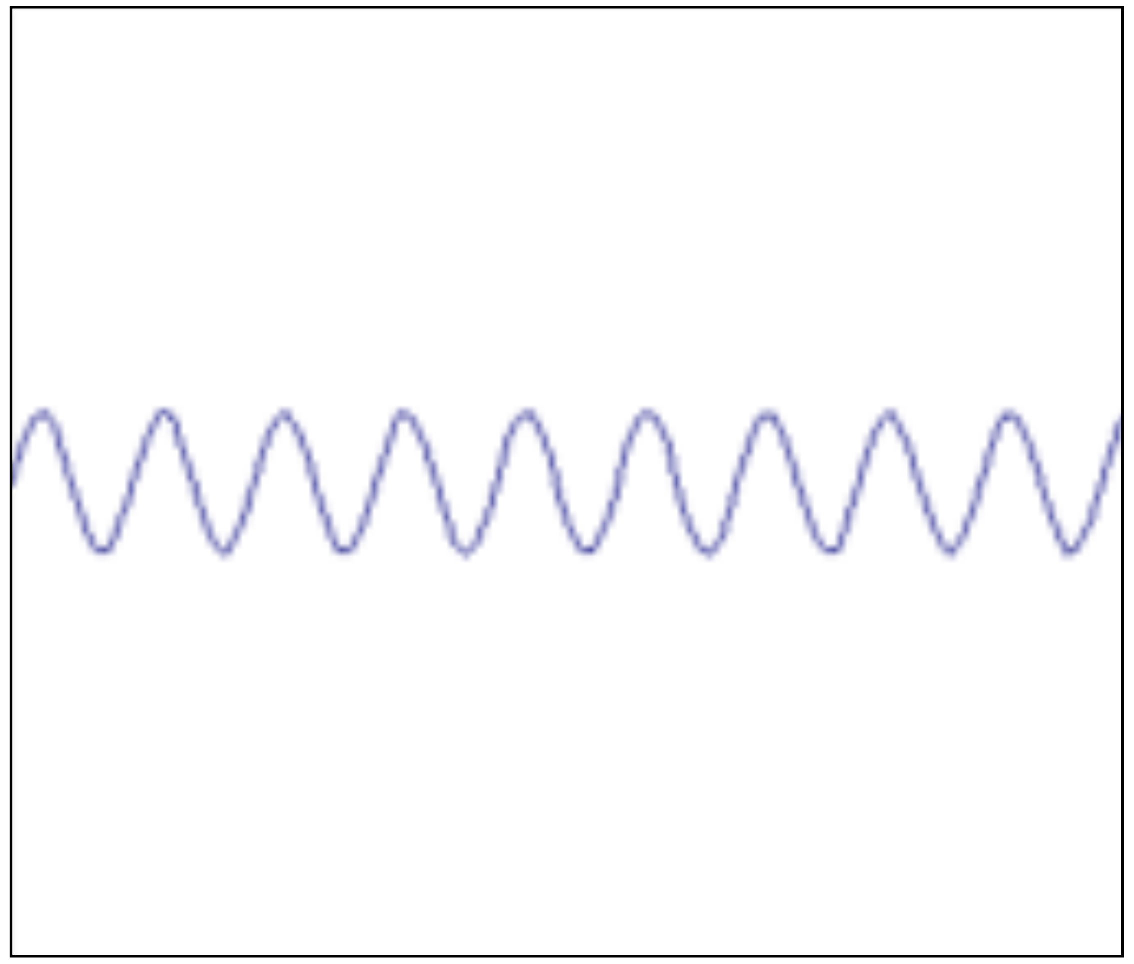


square wave

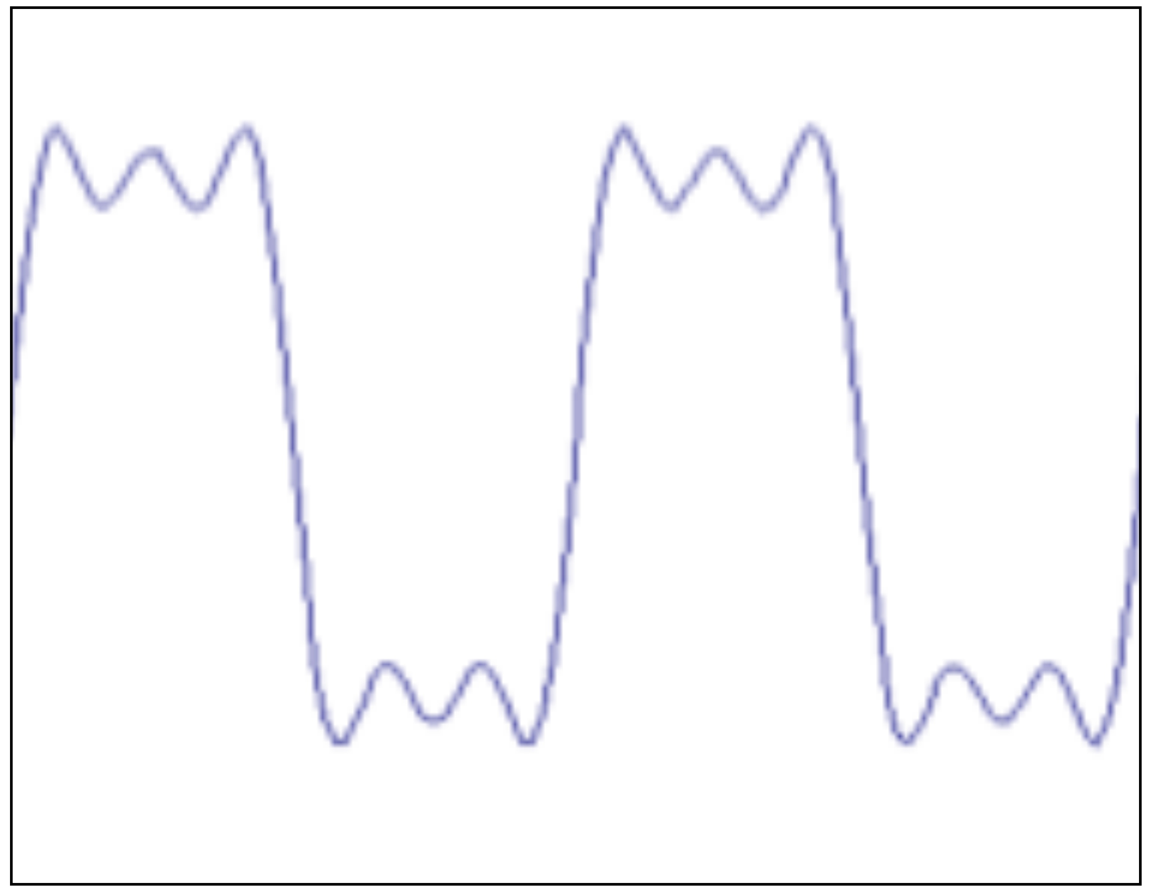
$\approx$



$+$



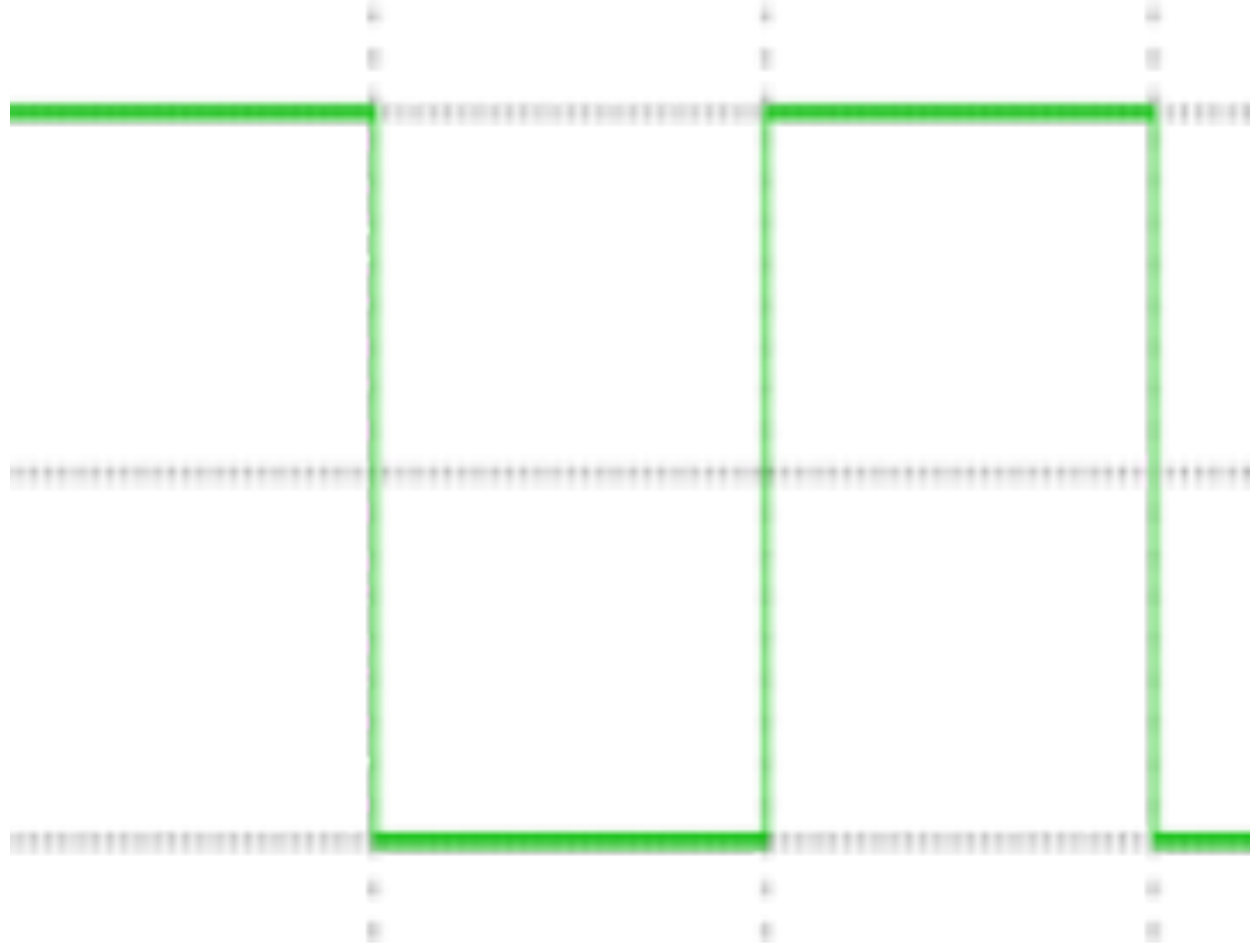
$=$



Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

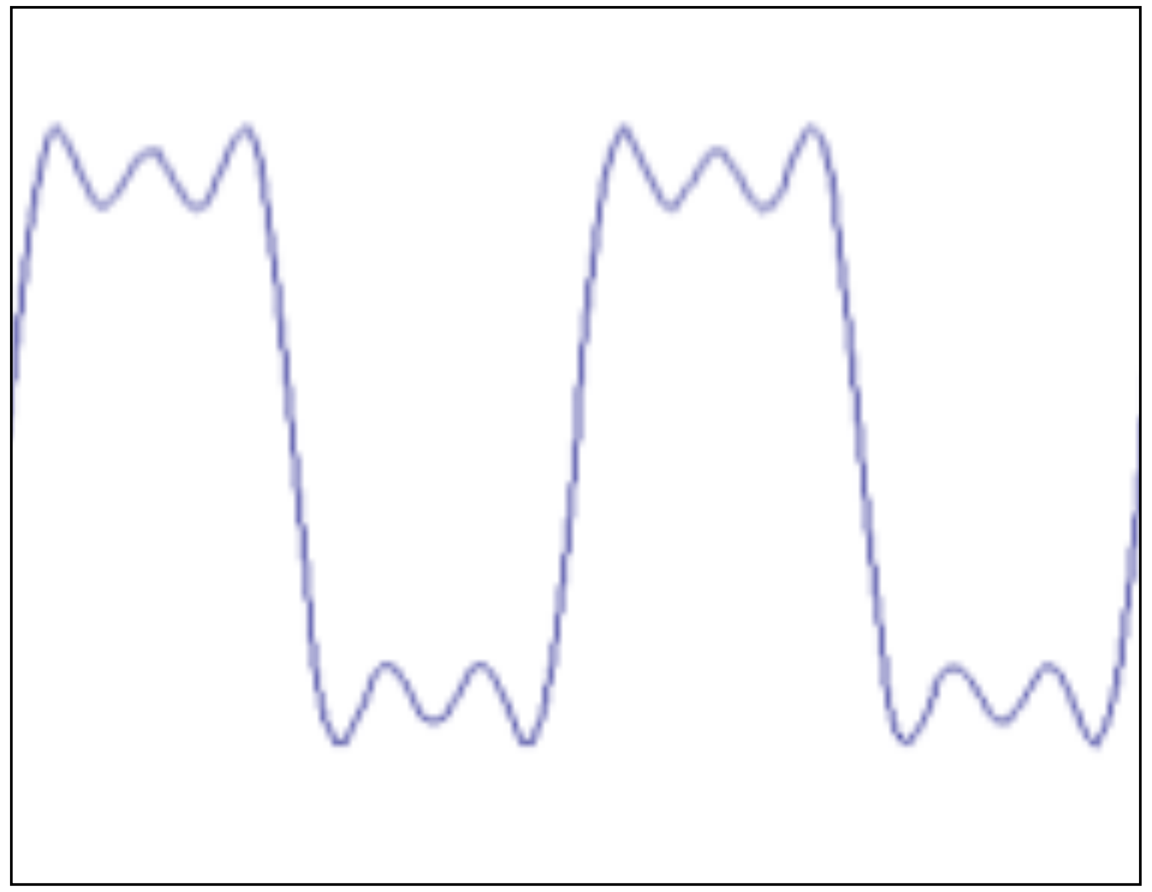
Fourier Transform (you will **NOT** be tested on this)

How would you generate this function?

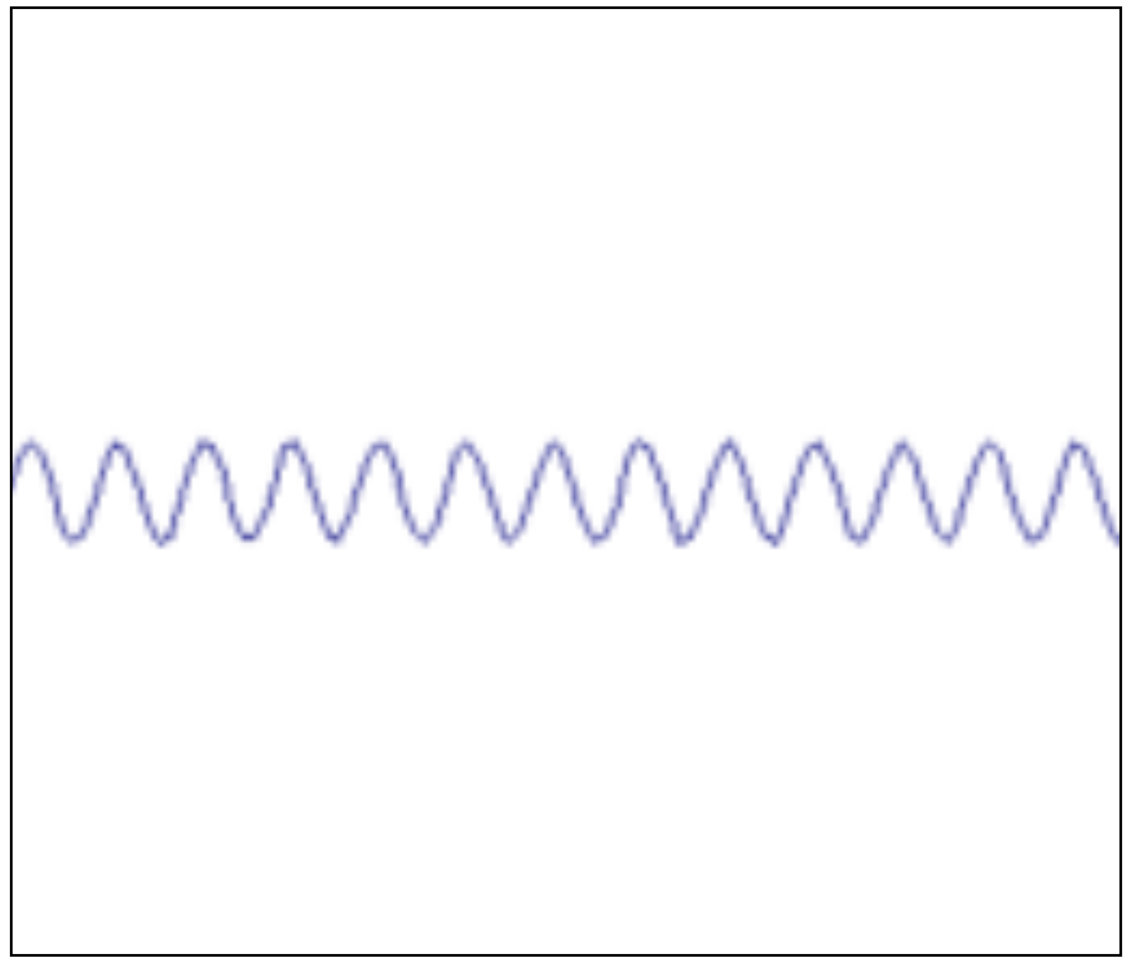


square wave

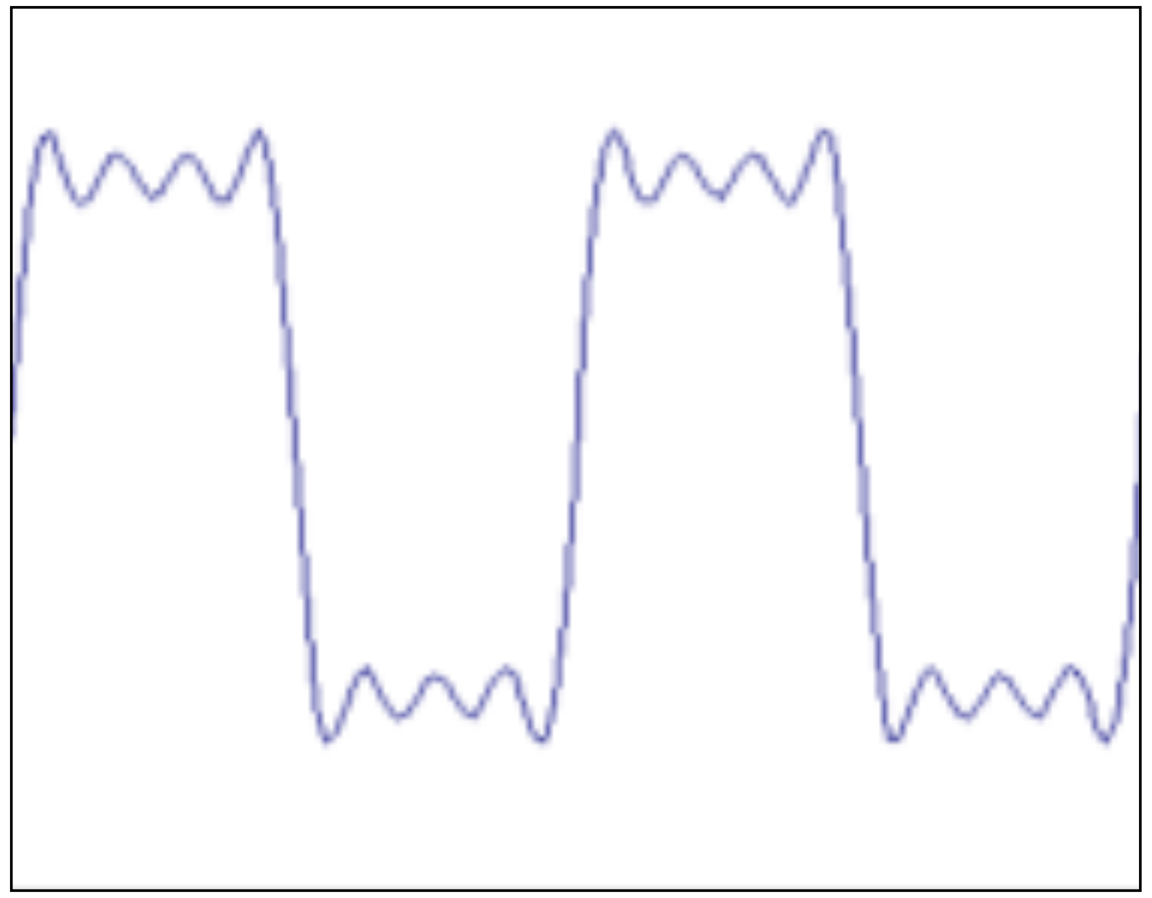
≈



+



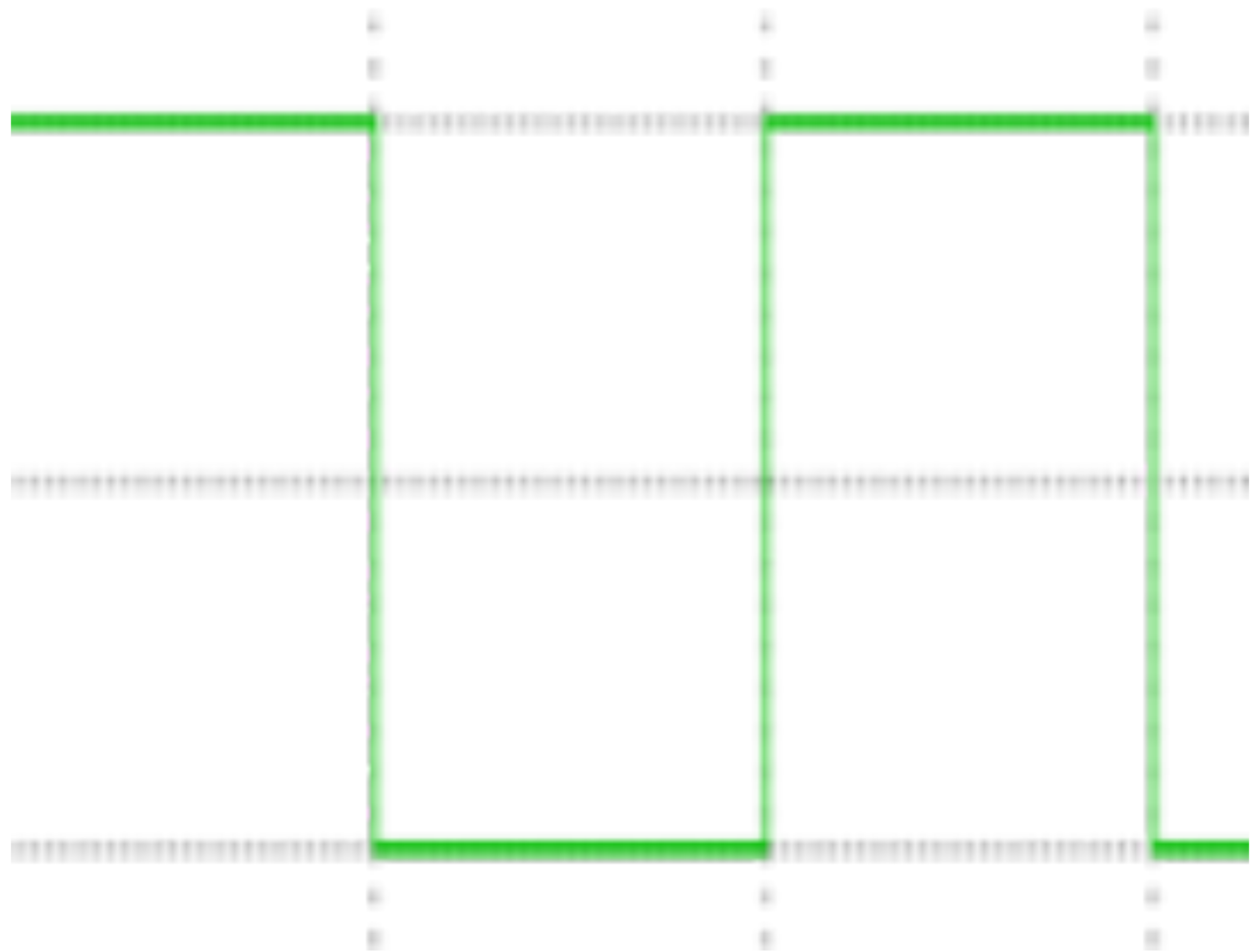
=



Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

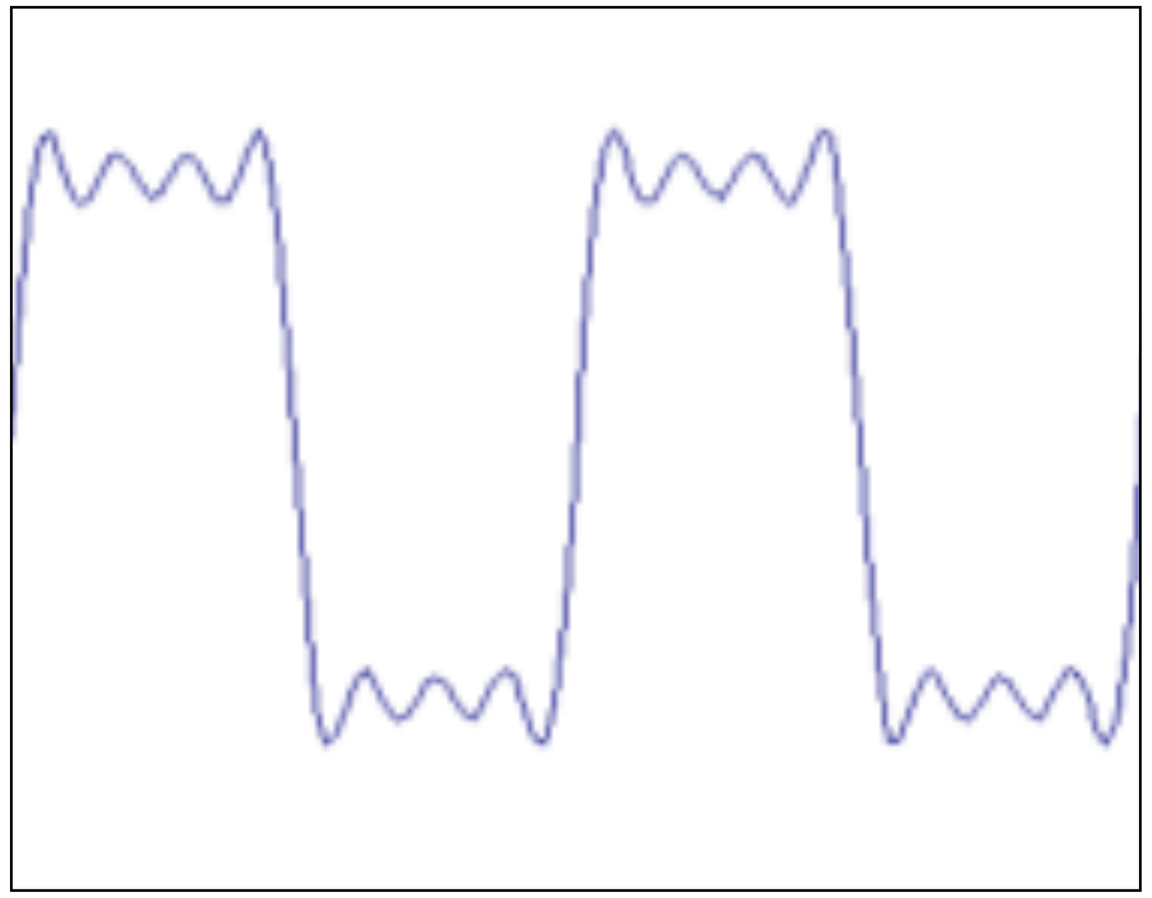
Fourier Transform (you will **NOT** be tested on this)

How would you generate this function?

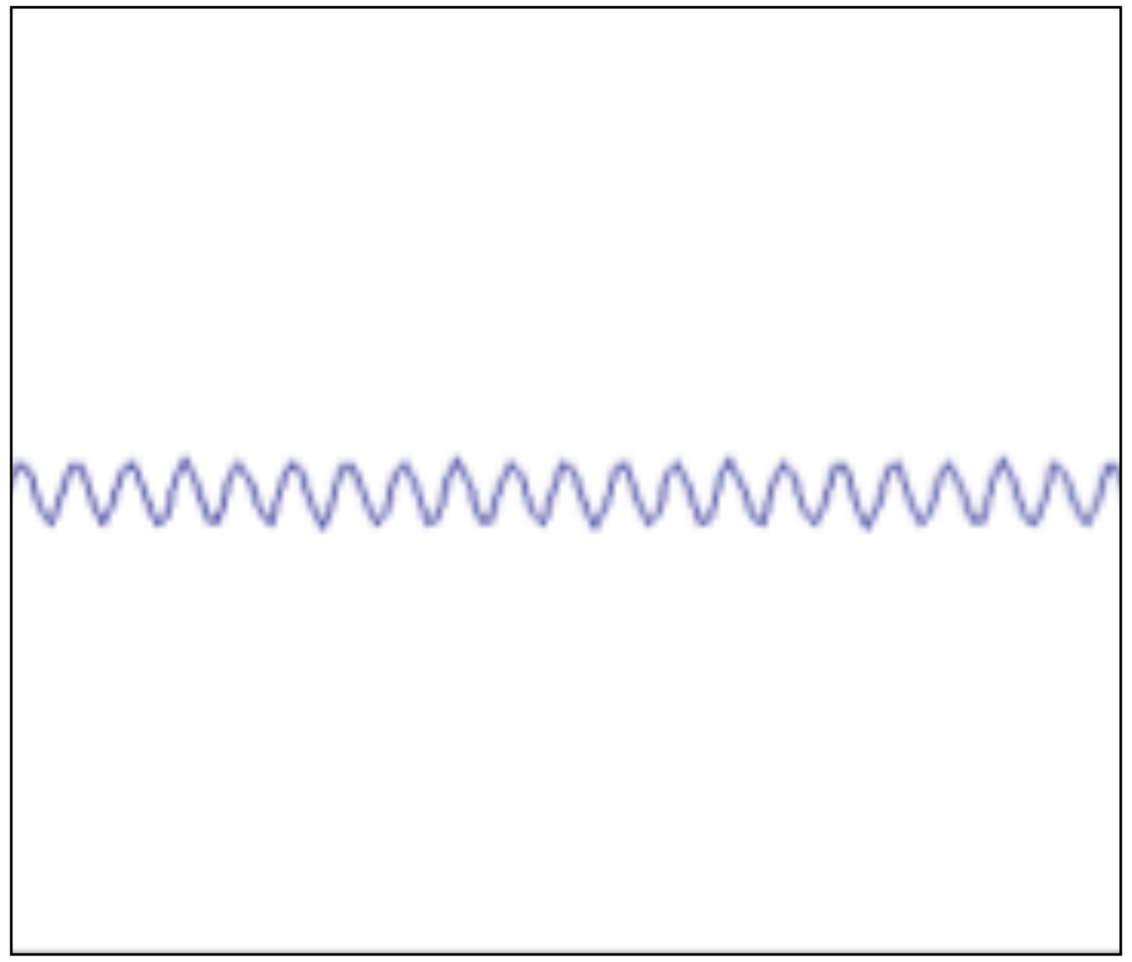


square wave

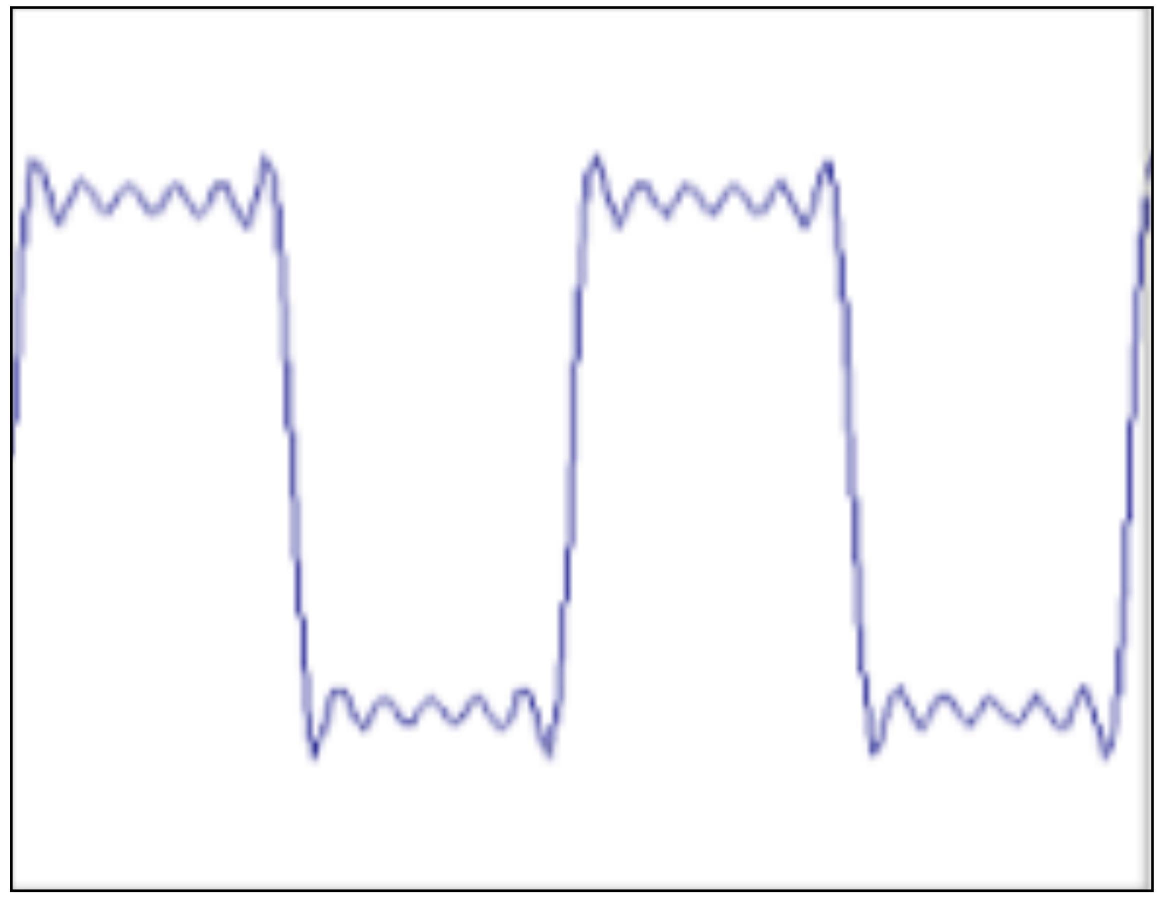
$\approx$



$+$



$=$

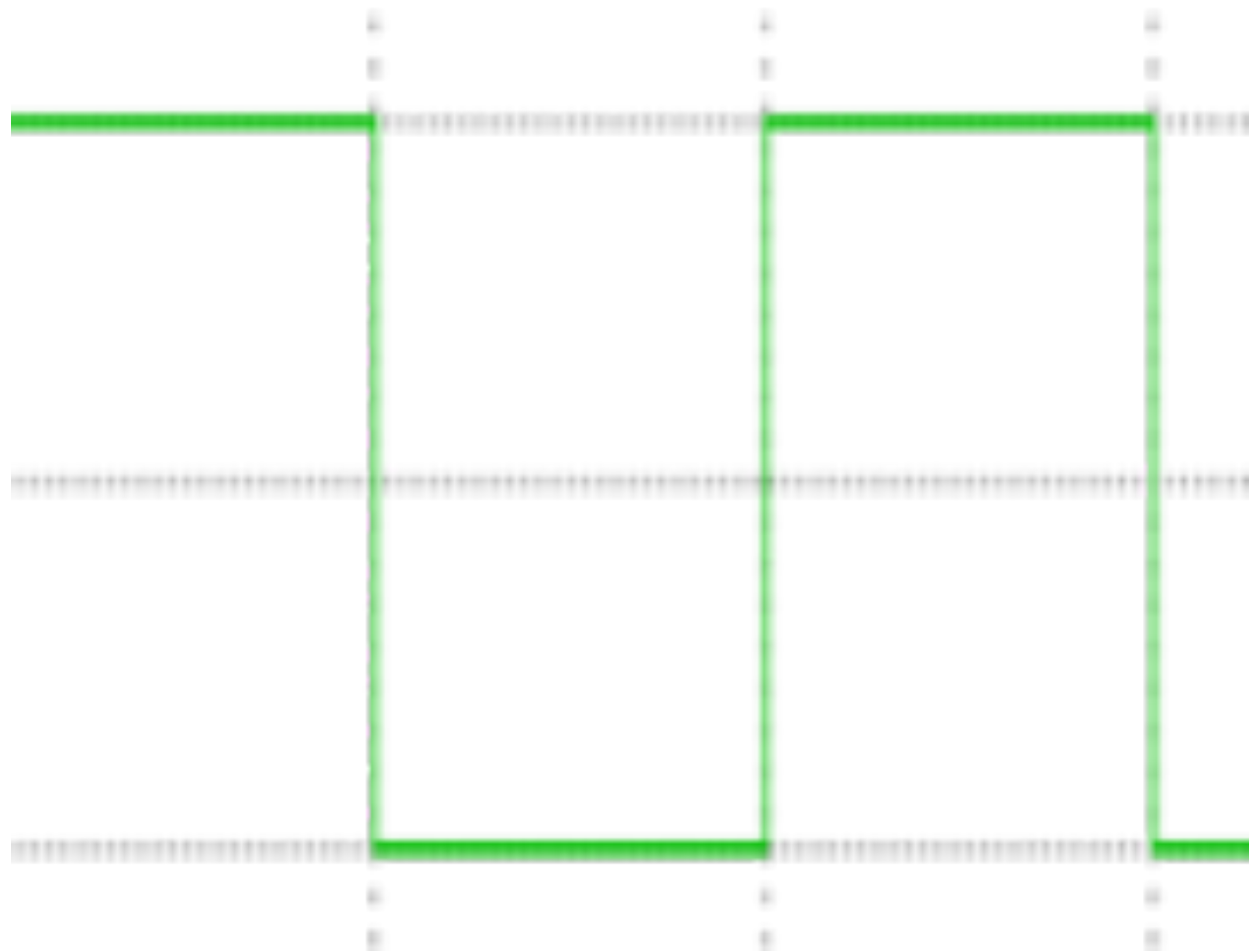


How would you express this mathematically?

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Fourier Transform (you will **NOT** be tested on this)

How would you generate this function?



$$= \; A \sum_{k=1}^{\infty} \frac{1}{k} \sin(2\pi k x)$$

square wave

infinite sum of sine waves

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Fourier Transform (you will **NOT** be tested on this)

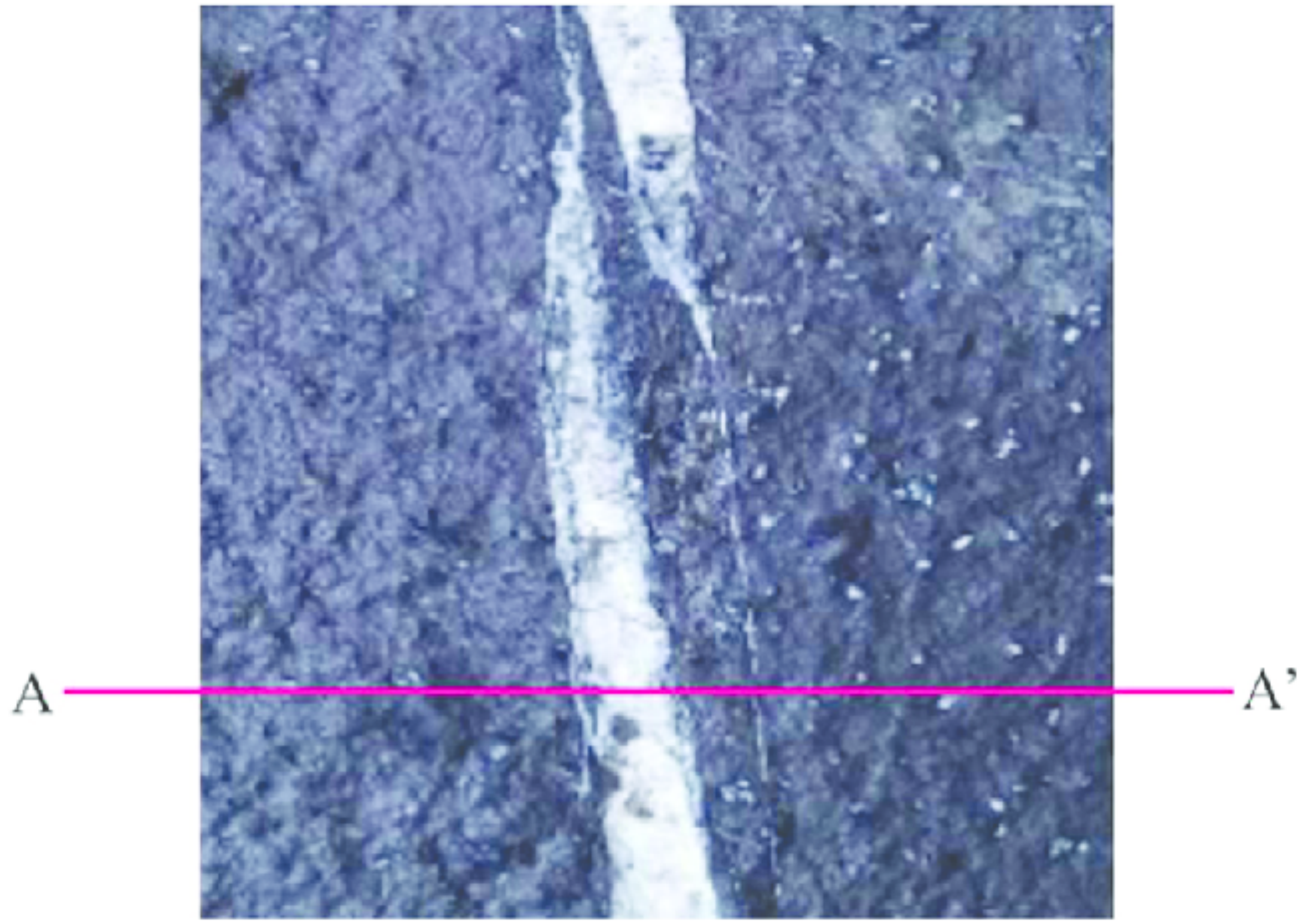
Basic building block:

$$A \sin(\omega x + \phi)$$

Fourier's claim: Add enough of these to get <u>any</u> periodic signal you want!

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Fourier Transform (you will **NOT** be tested on this)



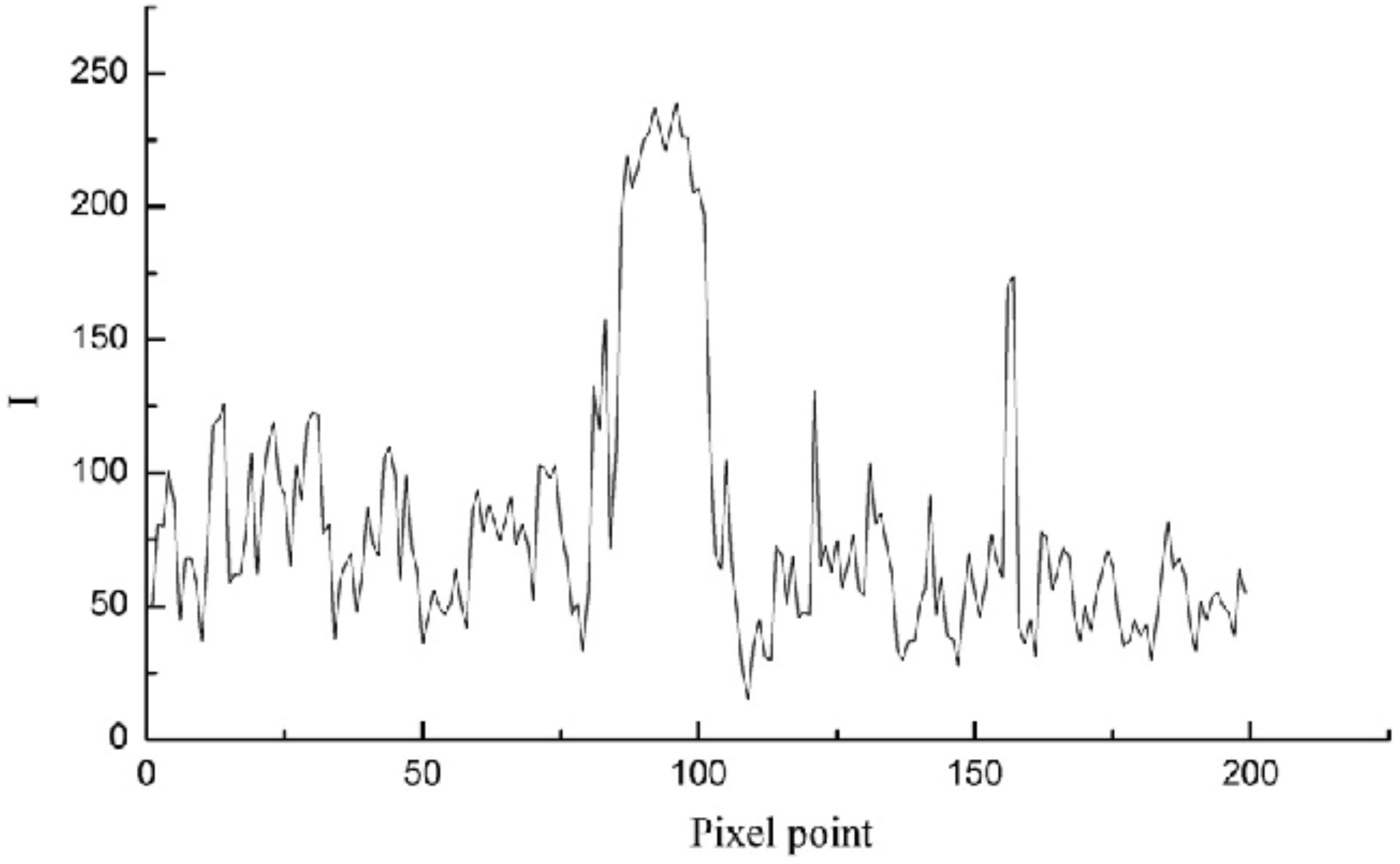
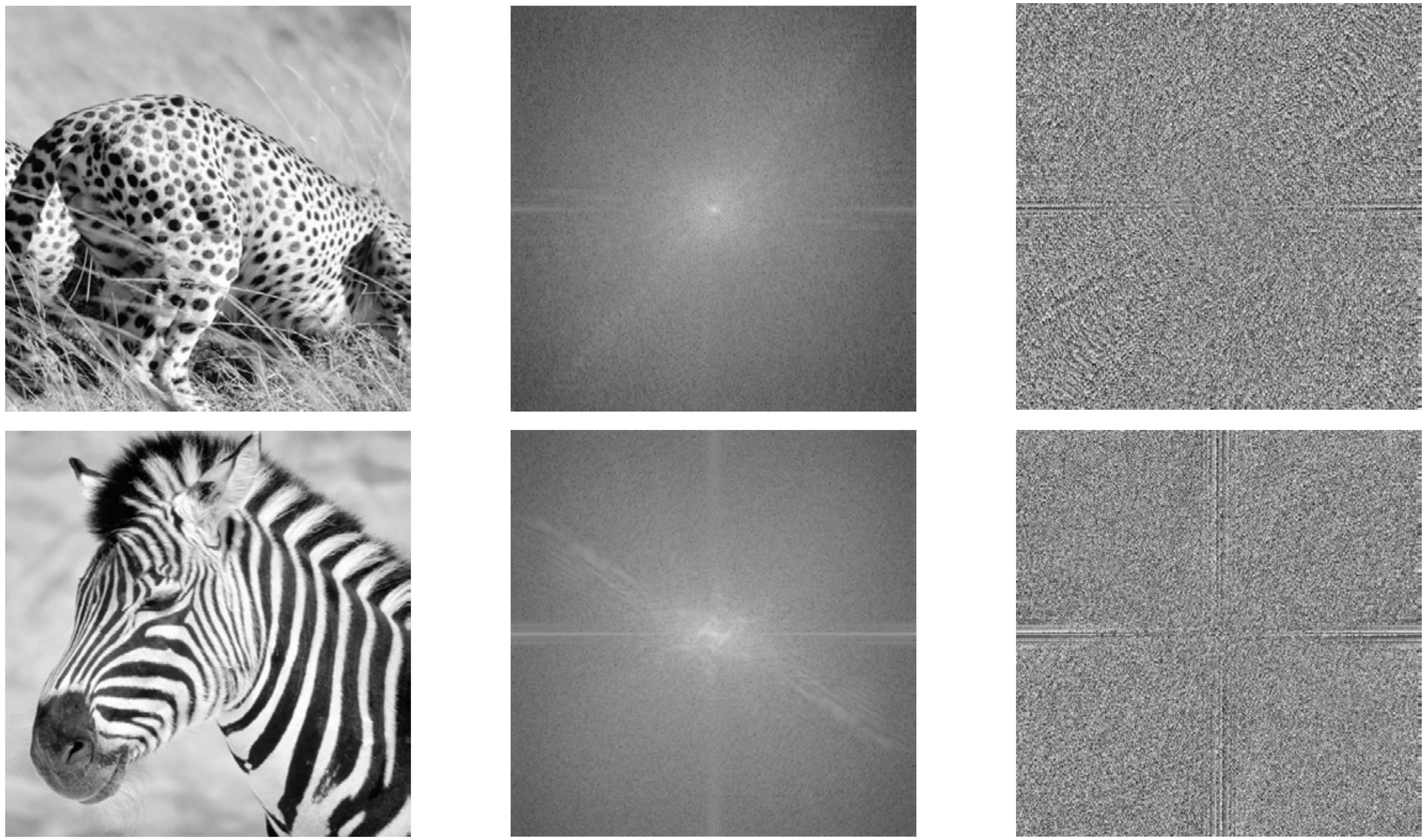


Image from: Numerical Simulation and Fractal Analysis of Mesoscopic Scale Failure in Shale Using Digital Images

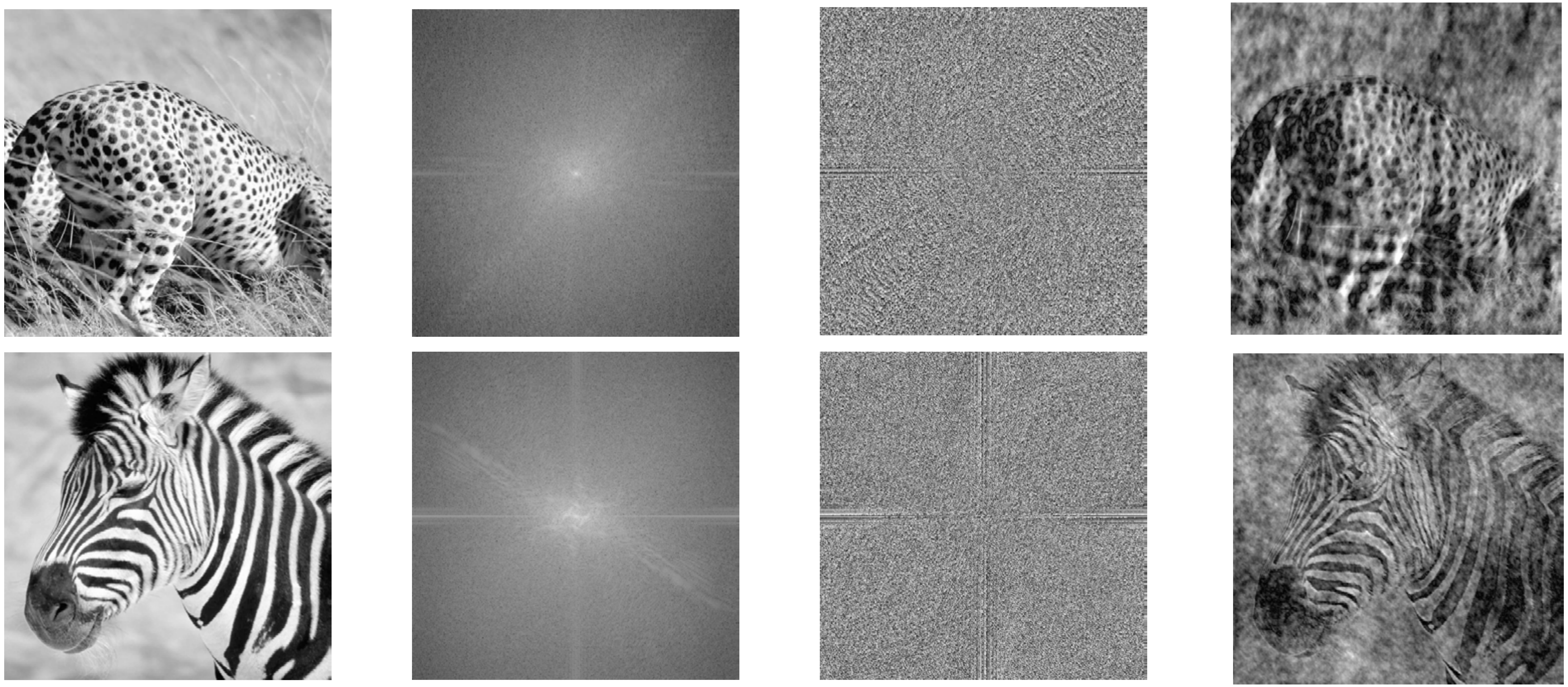
Fourier Transform (you will **NOT** be tested on this)



amplitude phase

Forsyth & Ponce (2nd ed.) Figure 4.6

Fourier Transform (you will **NOT** be tested on this)



cheetah phase with zebra amplitude

zebra phase with cheetah amplitude

amplitude

phase

Forsyth & Ponce (2nd ed.) Figure 4.6

Speeding Up **Convolution** (The Convolution Theorem)

Convolution **Theorem**:

Let $$i'(x,y) = f(x,y) \otimes i(x,y)$$

then $$\mathcal{I}'(w_x, w_y) = \mathcal{F}(w_x, w_y)\, \mathcal{I}(w_x, w_y)$$

where $\mathcal{I}'(w_x, w_y)$, $\mathcal{F}(w_x, w_y)$, and $\mathcal{I}(w_x, w_y)$ are Fourier transforms of $i'(x,y)$, $f(x,y)$ and $i(x,y)$

At the expense of two **Fourier** transforms and one inverse Fourier transform, convolution can be reduced to (complex) multiplication

Speeding Up **Convolution** (The Convolution Theorem)

General implementation of **convolution**:

At each pixel, (X, Y), there are $m \times m$ multiplications

There are $n \times n$ pixels in (X, Y)

Total: $m^2 \times n^2$ multiplications

Convolution if FFT space:

Cost of FFT/IFFT for image: $\mathcal{O}(n^2 \log n)$

Cost of FFT/IFFT for filter: $\mathcal{O}(m^2 \log m)$

Cost of convolution: $\mathcal{O}(n^2)$

Summary

We covered two additional linear filters: **Gaussian**, **pillbox**

Separability (of a 2D filter) allows for more efficient implementation (as two 1D filters)