Linear Filtering

Chul Min Yeum
Assistant Professor
Civil and Environmental Engineering
University of Waterloo, Canada

CIVE 497 – CIVE 700: Smart Structure Technology

Last updated: 2020-12-06

UNIVERSITY OF WATERLOO
FACULTY OF ENGINEERING

Image as Functions

- We can think of an image as a function f, mapping $\mathbb{R}^2$ $(space) \rightarrow \mathbb{R}$ $(intensity)$:
  - $f(x, y)$ gives the intensity at point (x, y)
  - Realistically, images are rectangles, with a finite intensity range
  - Color depth (8bit or 16 bit). Thus:

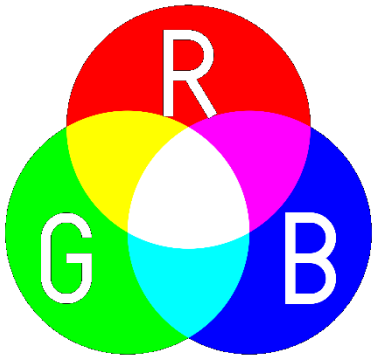
$$f: [0, w] \times [0, h] \rightarrow [0,1],$$

$$w = width, h = height$$

Q. What is the color depth?

- A color image is just three "grayscale" images pasted together.

$$f(x, y) = \begin{bmatrix} r(x, y) \\ g(x, y) \\ b(x, y) \end{bmatrix}$$, r=red, g=green, b=blue



*RGB isn't the only color space! There are others, like YCbCr, sRGB, CMY…

Example: 8bit Vs 16 bit Images

Supplement

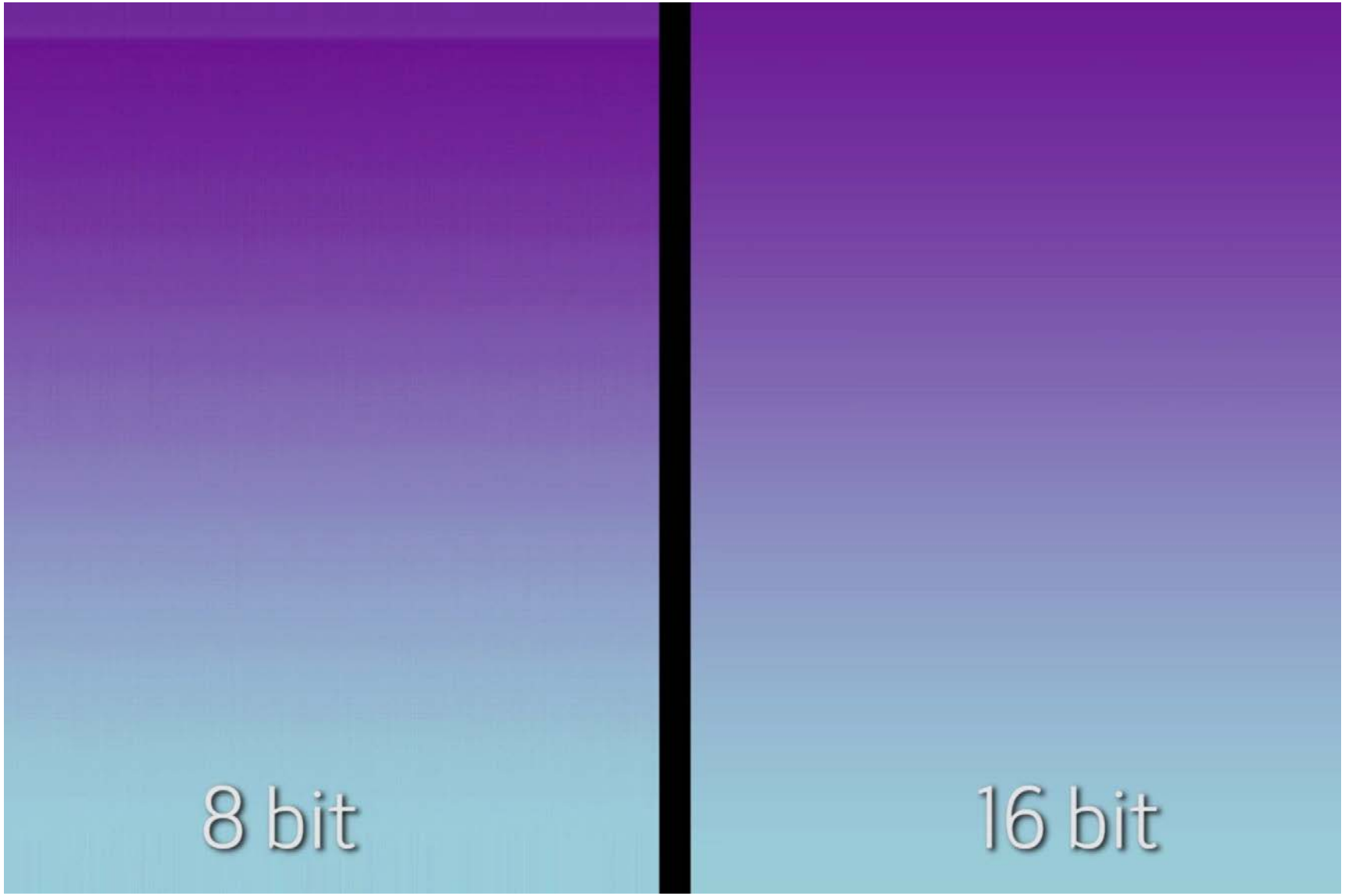
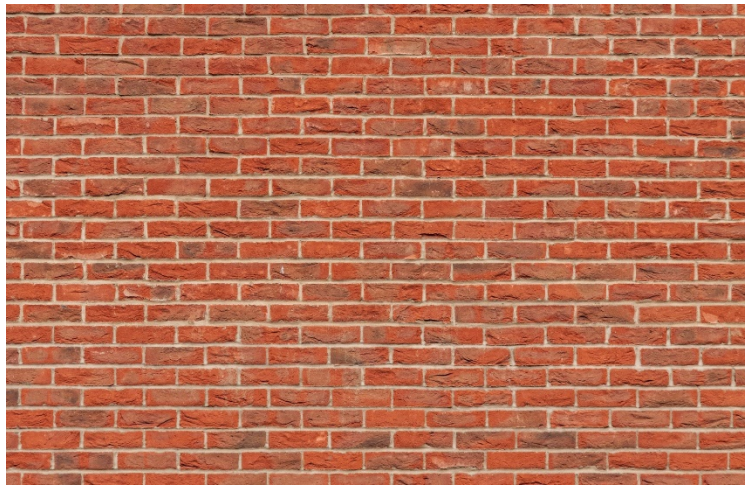
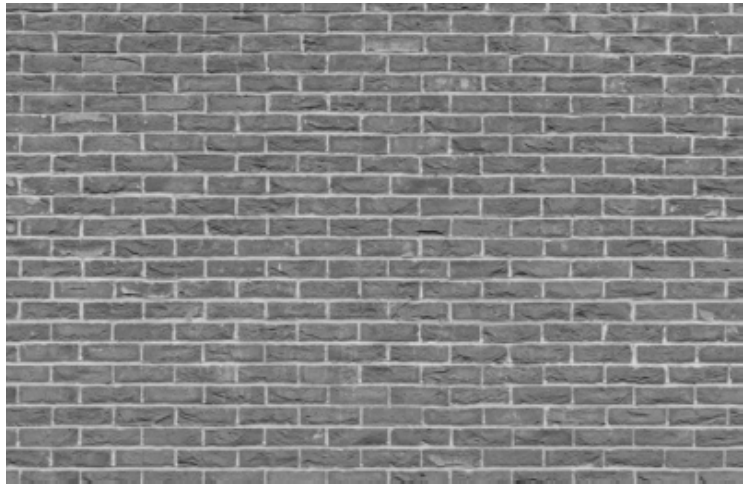


Image as Functions (Continue)

RGB



Gray



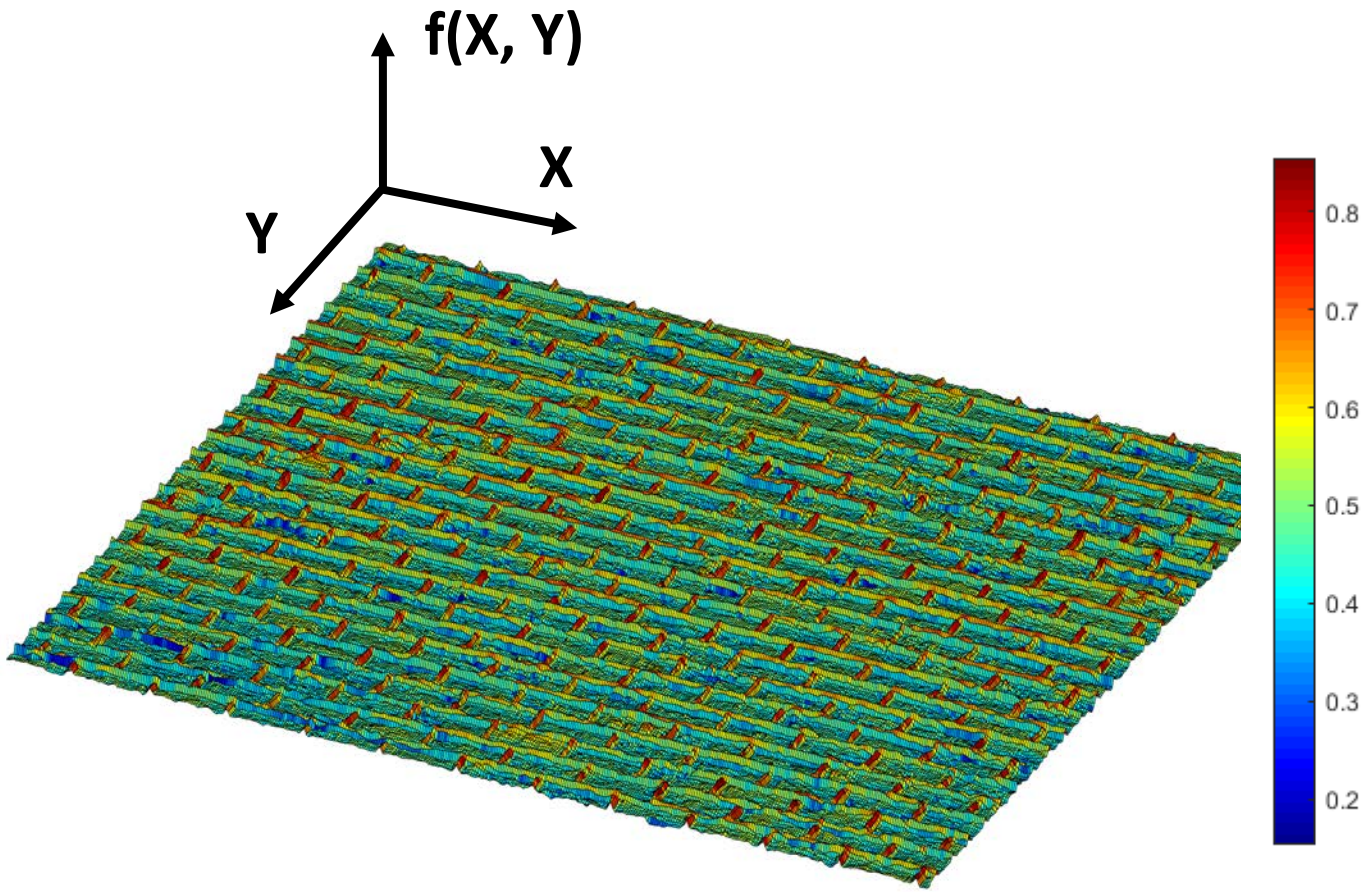
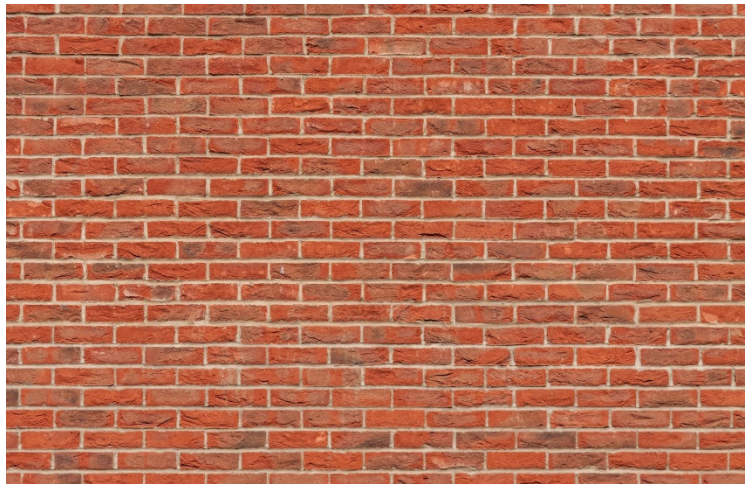
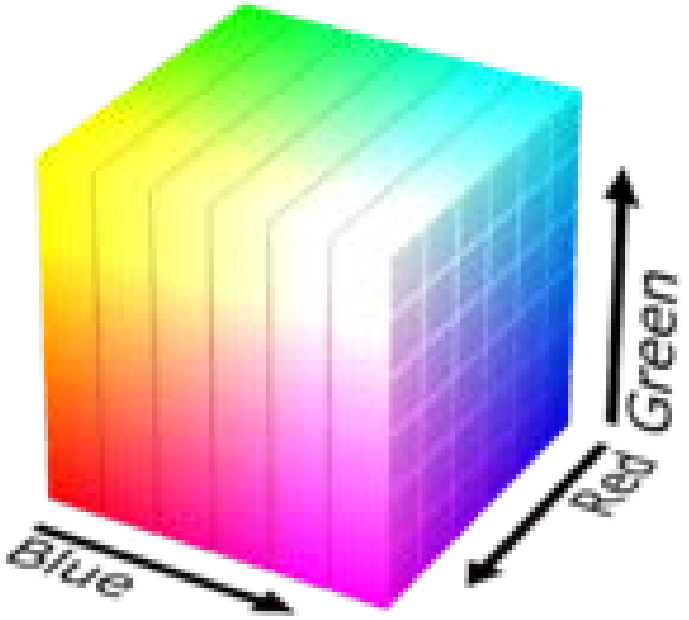
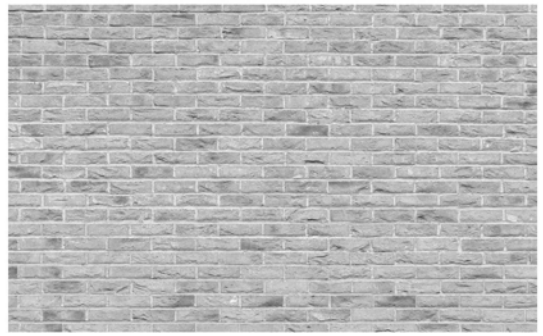


Image as Functions (Continue)

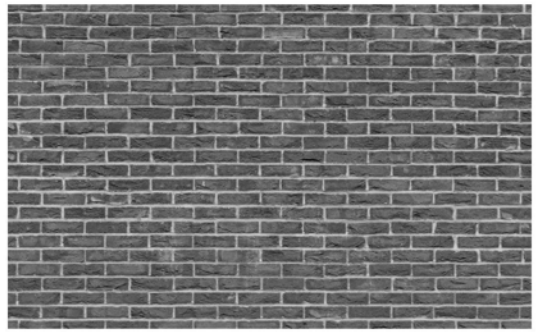
RGB



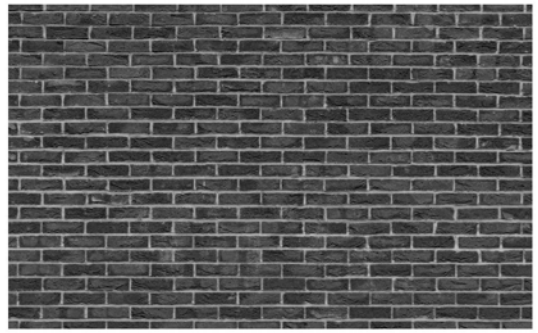




R



G



B

Grayscale in each channel

Linear Filtering?

- **Filtering**: Modify or enhance data using a function
- **Linear**: The function follows linear properties of scaling and superposition.
  - Superposition: $\mathrm{h}(A + B) = h(A) + h(B)$
  - Scaling: $\mathrm{h}(\alpha A) = \alpha h(A)$
- Simply, it's linear combination: $\mathrm{h}(X, Y, Z) = aX + bY + cZ$

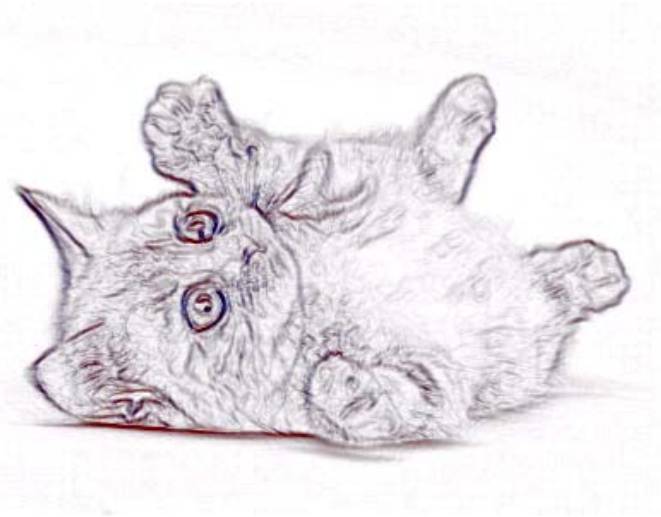
* h is a linear filter.

Linear Filtering?

- Linear filtering is used to:
  - Reduce noise in data
  - Extract features from data

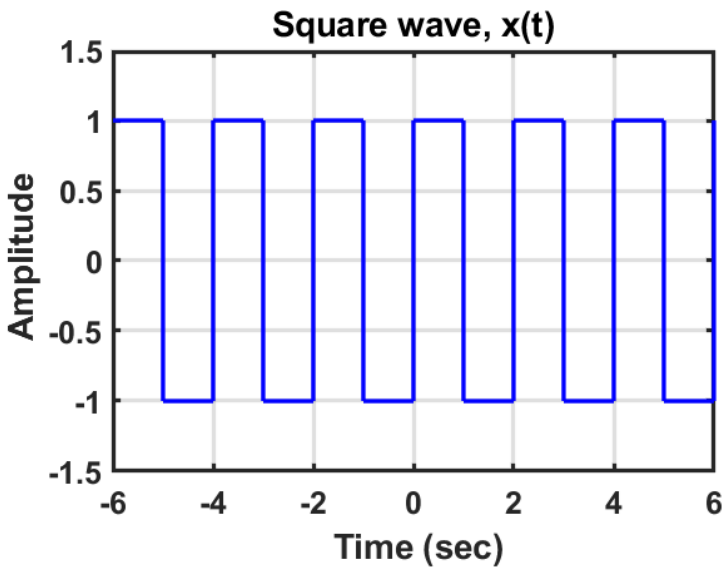
Noisy image

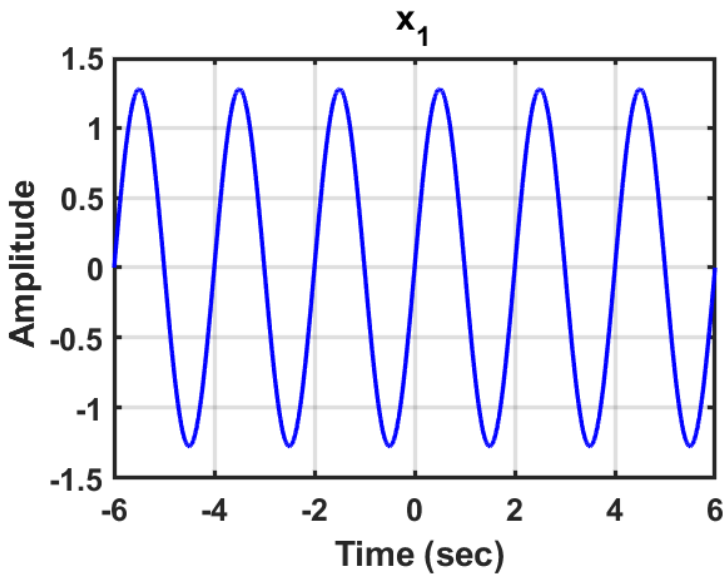
Denoised image

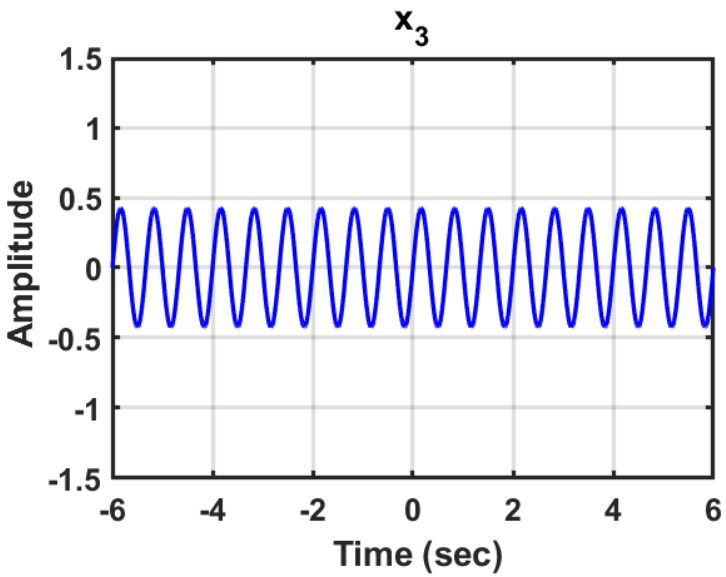


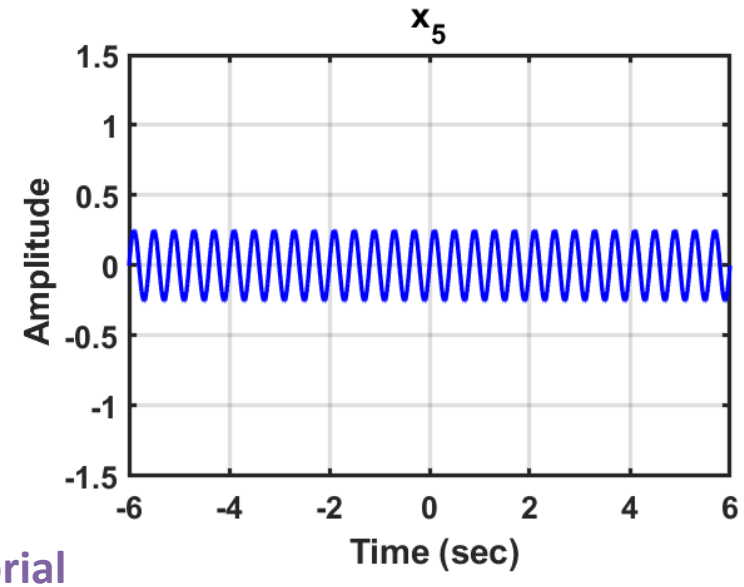
Recall: Fourier Series of a Square Wave

Supplement







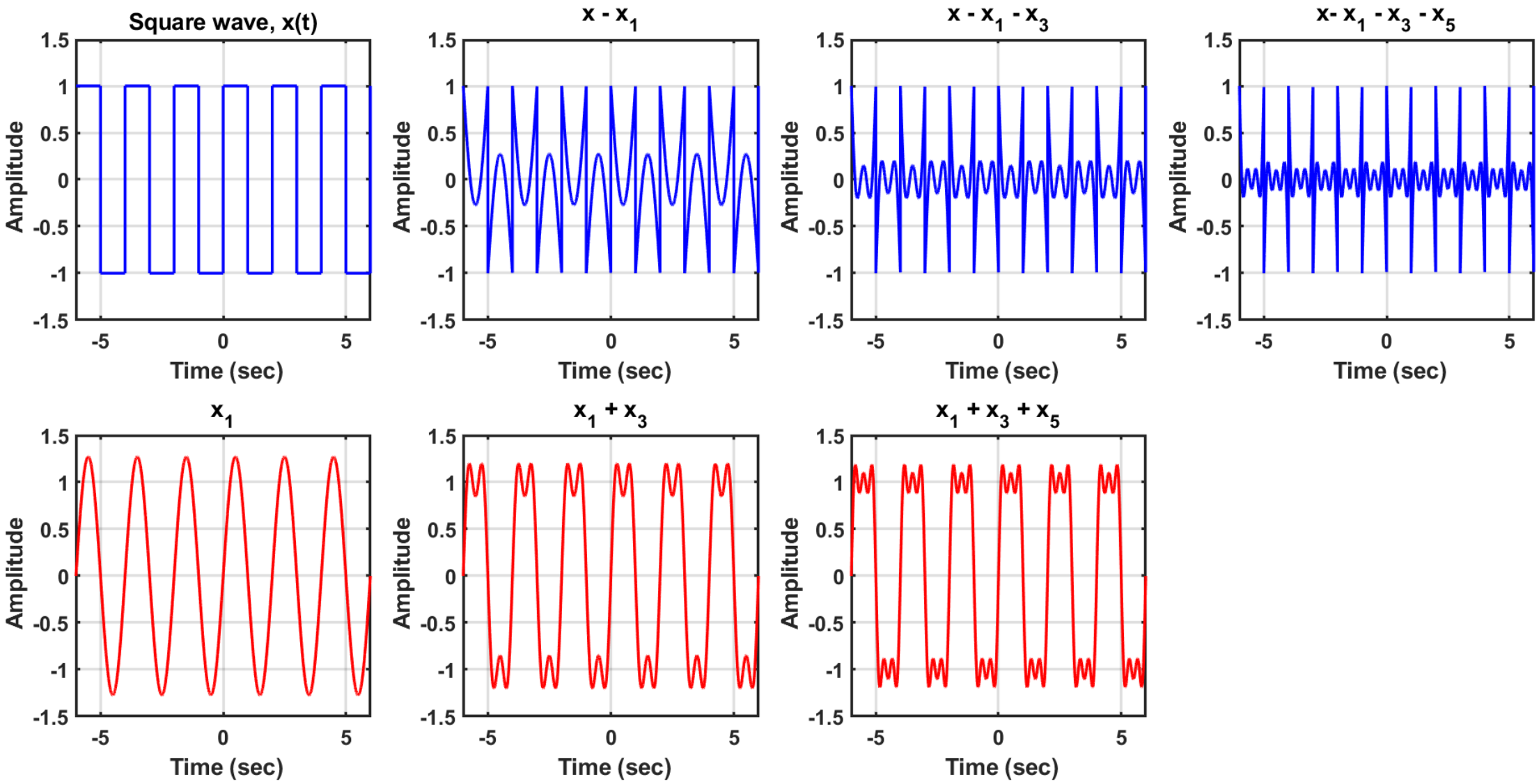


$$x(t) = \sum_{n=1}^{\infty} \frac{2}{n\pi}(1 - cosn\pi) * \sin\left(\frac{2\pi nt}{T_p}\right) = \sum_{n=1}^{\infty} x_n(t)$$

Tutorial

Recall: Fourier Series of a Square Wave (Continue)

Supplement



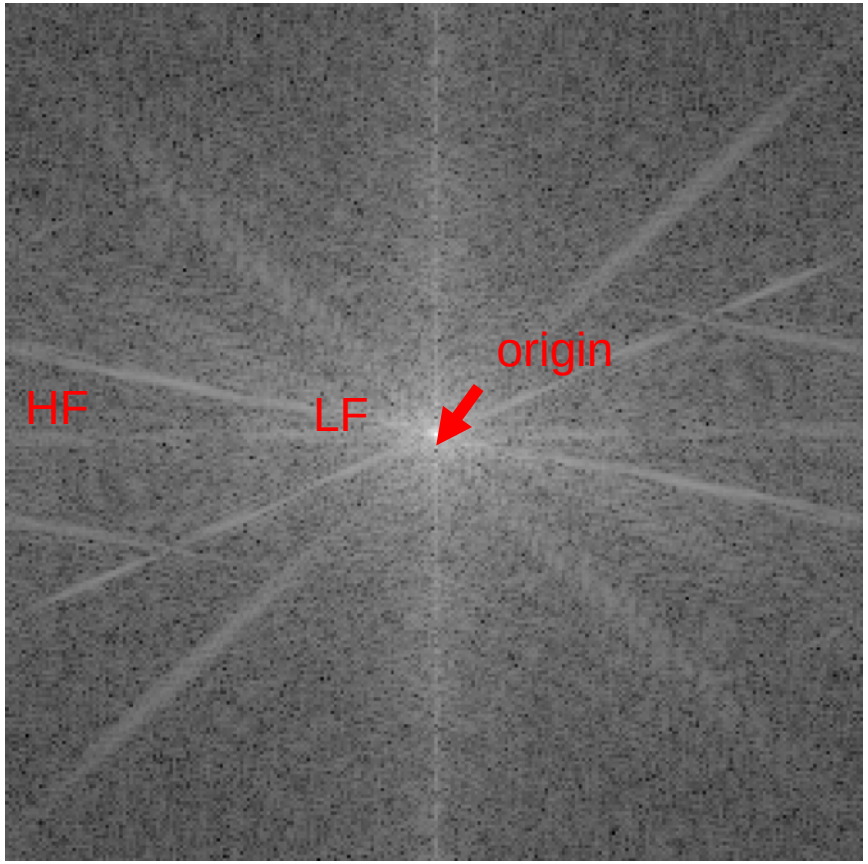
Tutorial

Image in a Frequency Domain

- Here is an image and its Fourier transform

Low frequency (LF) ≈ solid colors
High frequency (HF) ≈ noise/edges

2D image

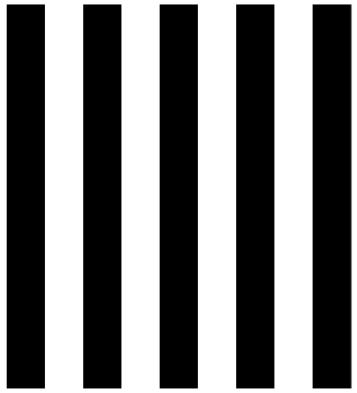


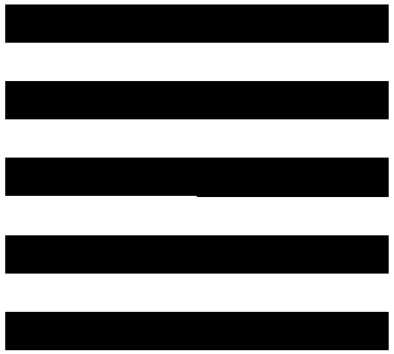
2D FFT

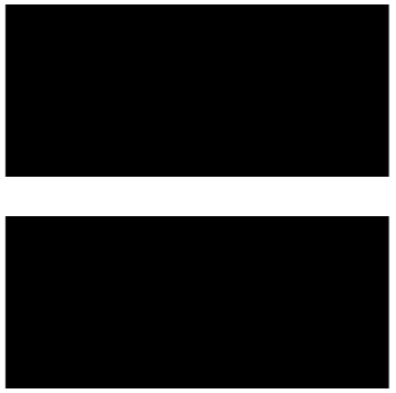
Image in a Frequency Domain (Patterns)

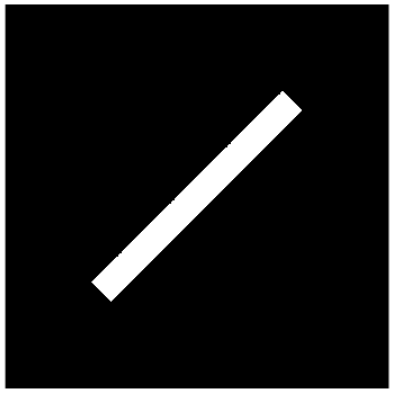
Supplement

Image

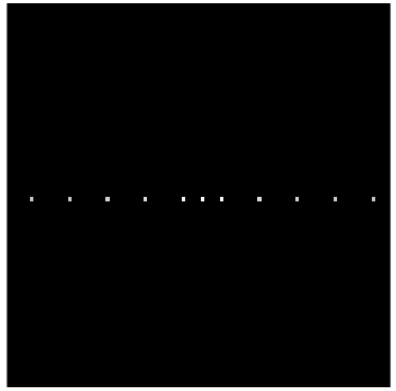


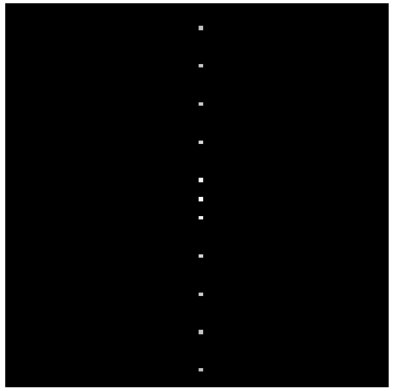


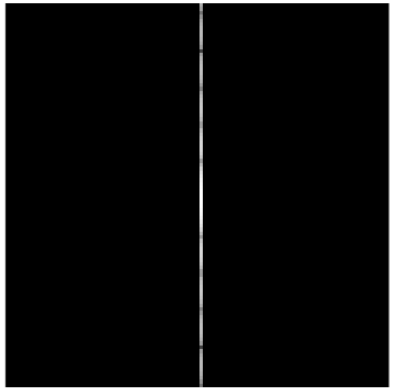


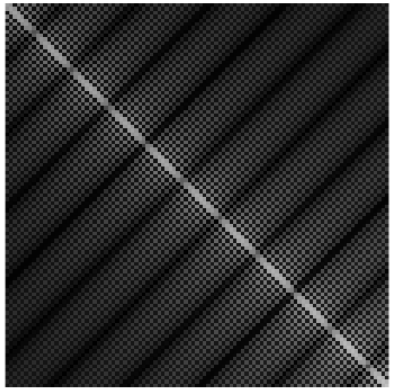


Frequency



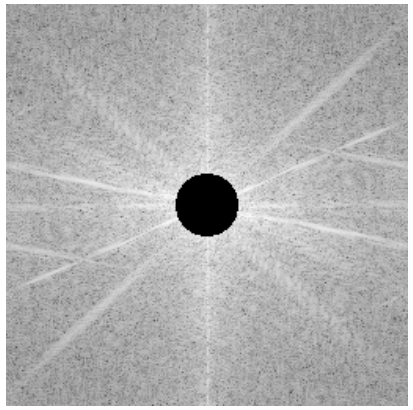




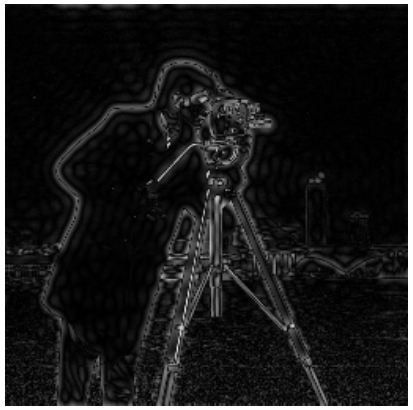


Tutorial

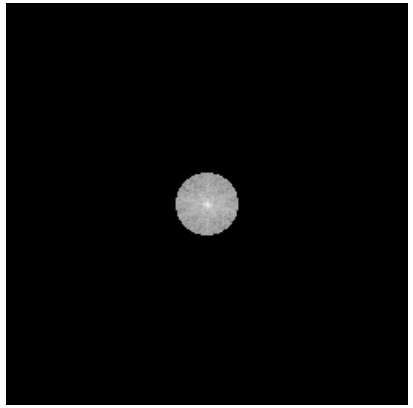
How Filtering Works in Frequency



Remove low frequency (only pass high frequency)



Highlighting edge



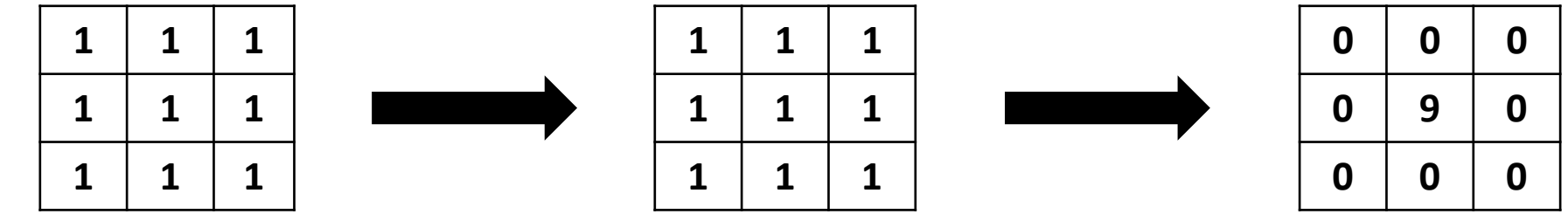
Remove high frequency (only pass low frequency)

Blurring

Linear Filtering: Simple Example

- One simple version : linear filtering (cross-correlation or convolution)
  - Replace each pixel by a linear combination of its neighbors

- The prescription for the linear combination (how to compute the value using its own and nearby pixels) is called the “kernel” (or “mask” or “filtering”)



Cross-Correlation Definition

Cross-correlation is a measure of similarity between the signal and the kernel.

Let x be the signal, h be the kernel (basically convolution but inverted):

$$h(t) \otimes x(t) = \int_{-\infty}^{\infty} h(\tau) x(t + \tau) d\tau$$

This is called a cross-correlation operation:

$$g = h \otimes f$$

$$h(t) * x(t) = \int_{-\infty}^{\infty} h(\tau) x(t - \tau) d\tau$$

Convolution

Comparison between Cross-Correlation and Convolution

Supplement

Cross-correlation

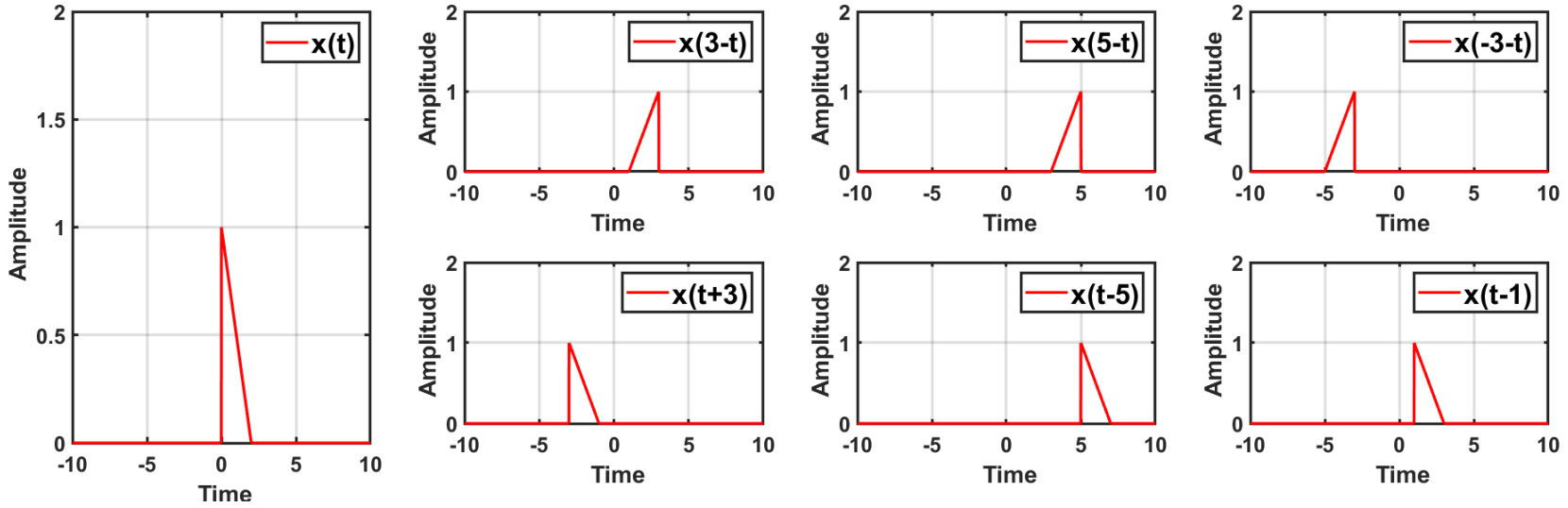
$$h(t)\otimes x(t) = \int_{-\infty}^{\infty} h(\tau)x(t+\tau)d\tau$$

Convolution (commutative)

$$h(t)\otimes x(t) = \int_{-\infty}^{\infty} h(\tau)x(t-\tau)d\tau$$

Replace $t-\tau$ to τ'

$$h(t)\otimes x(t) = \int_{-\infty}^{\infty} x(\tau')h(t-\tau')d\tau'$$



Kernel in the cross-correlation

Kernel in the convolution

Tutorial

2D Cross-Correlation Using a Kernel

Cross-correlation is a measure of similarity between the image and the kernel.

Let F be the image, H be the kernel (of size $2k+1$ by $2k+1$),

G be the output image. **For G at point i, j:**

Q. Why is this a measure of similarity?

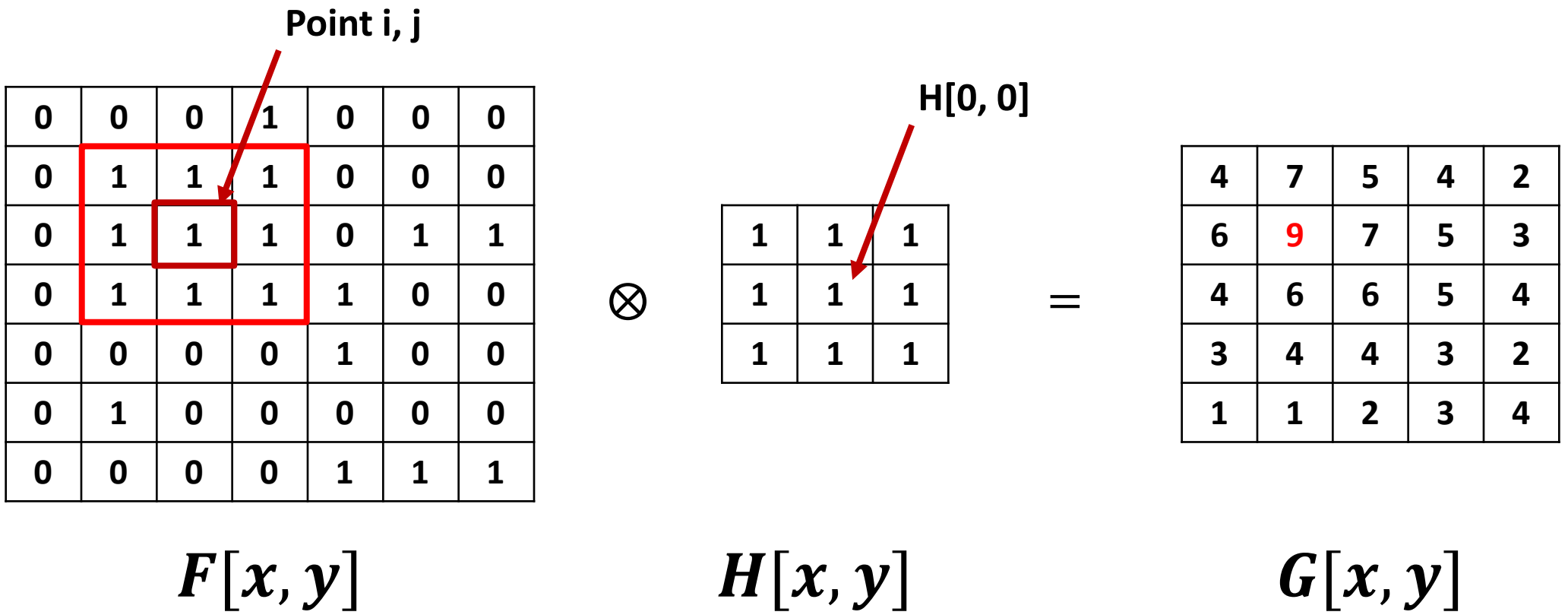
$$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u,v]F[i+u, j+v]$$

Again, this is called a cross-correlation operation:

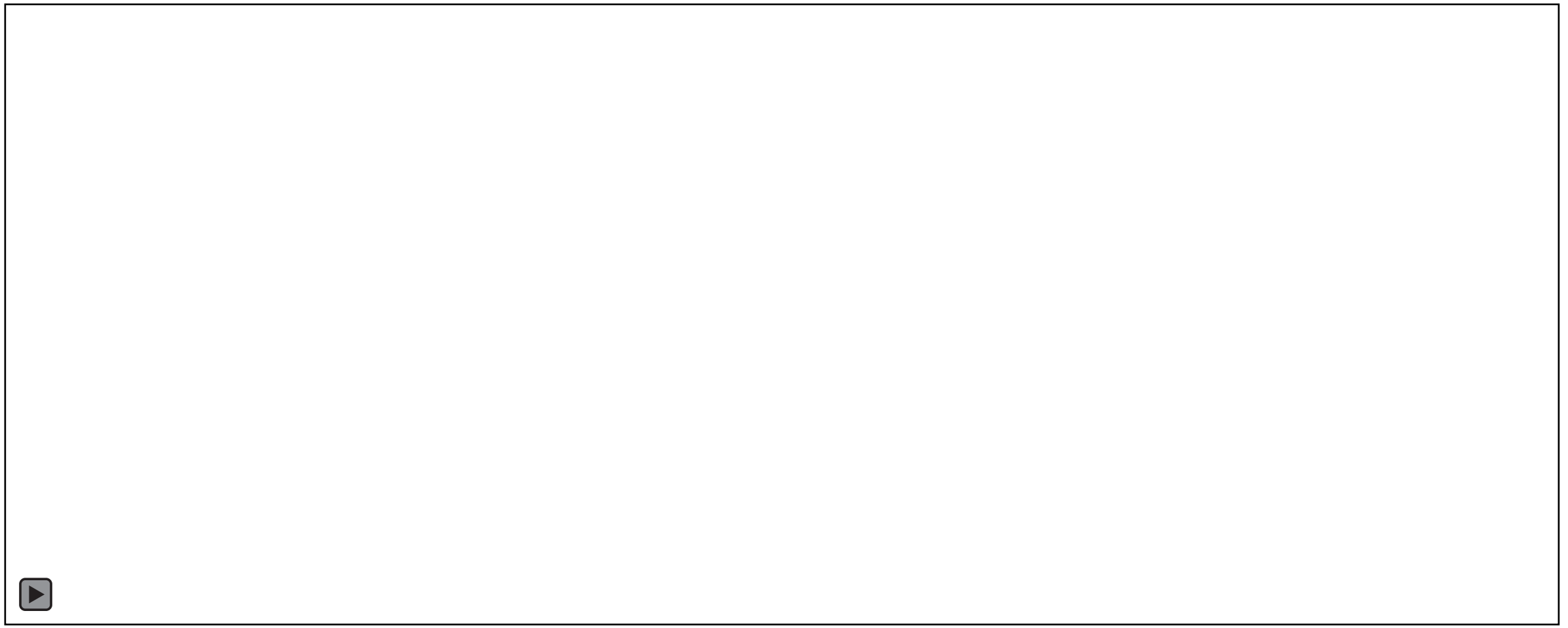
$$G = H \otimes F$$

2D Cross-Correlation Example

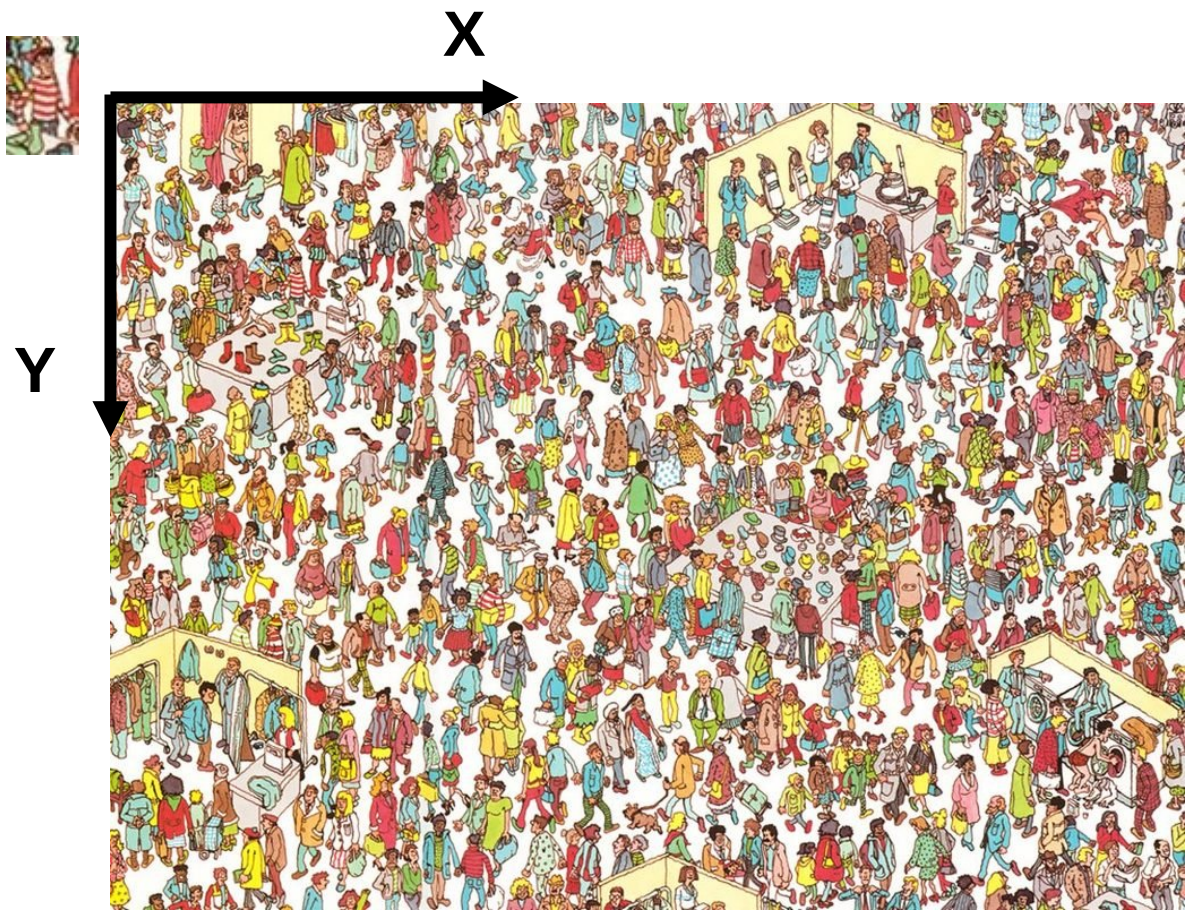
$$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u,v]F[i+u, j+v]$$

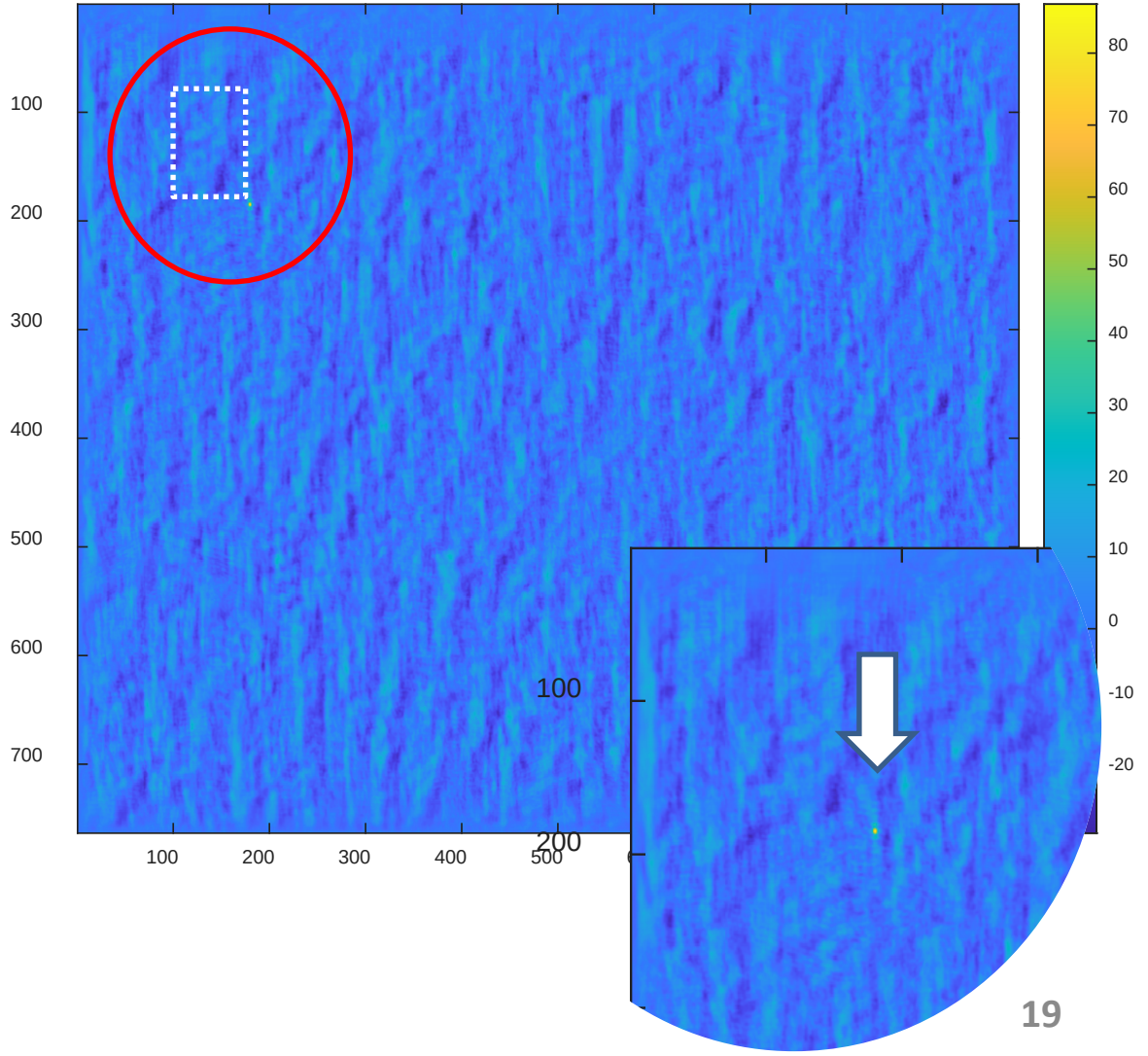


Computation of the Cross-Correlation Between Two Matrices



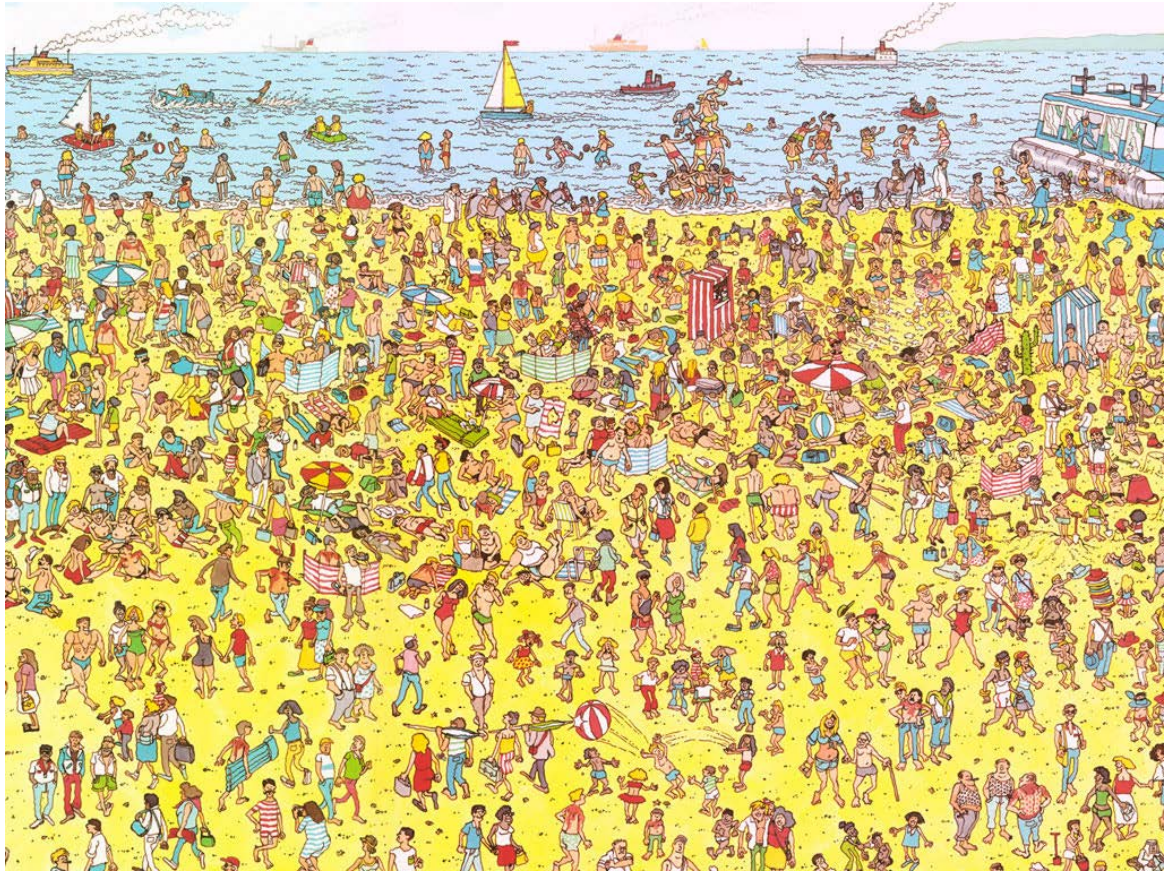
Example: Finding Waldo from the Image (Template Matching)

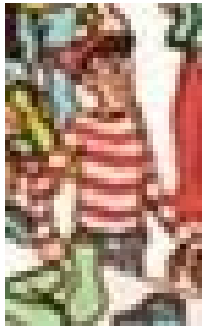




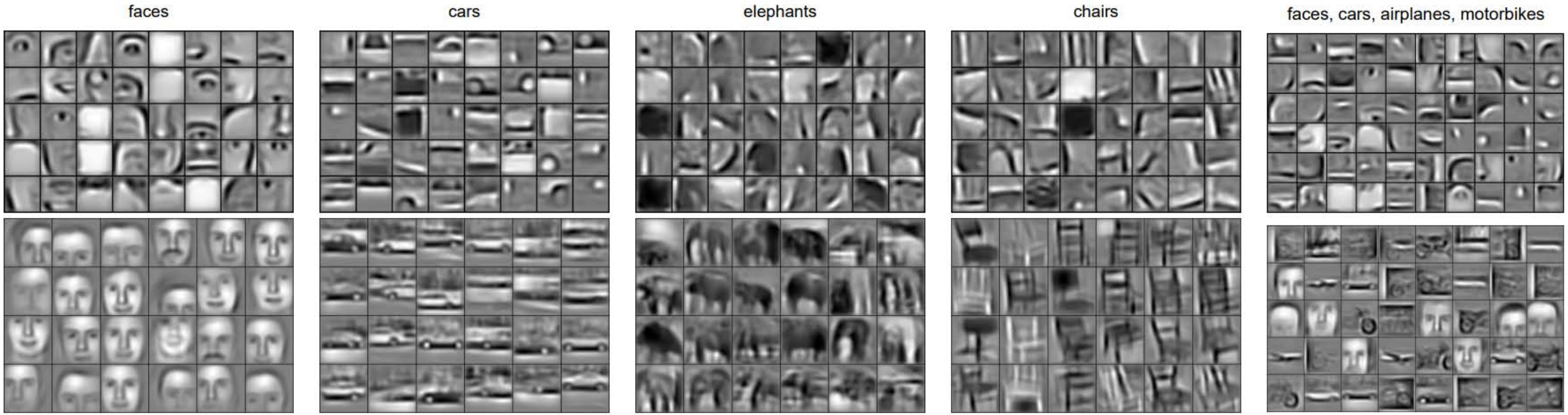
Cross-correlation: Scanning the (pre-processed) image with a waldo image.

Challenges of the Template Matching





Applications: Convolution Neural Network



2D Convolution

Let F be the image, H be the kernel (of size 2k+1 x 2k+1),

G be the output image

$$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u,v]F[i-u, j-v]$$

Only difference is that the kernel is "flipped" horizontally and vertically.

This is called a convolution operation:

$$G = H * F$$

Convolution Properties: Commutative, Associative, and Linearity

- Commutative: $F * H = H * F$
- Associative: $F * (H * L) = (H * F) * L$
- Linearity: $F * (H_1 + H_2) = F * H_1 + F * H_2$
- Relationship with differentiation: $(F * H)' = F' * H = F * H'$

Linear Filter: Generating an Identical Image

0	0	0
0	1	0
0	0	0

H_i

0	0	0
0	0	1
0	0	0

H_s

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

H_r

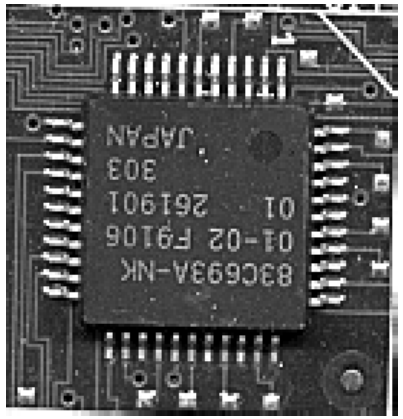
-0.11	-0.11	-0.11
-0.11	1.89	-0.11
-0.11	-0.11	-0.11

H_p

0.08	0.12	0.08
0.12	0.20	0.12
0.08	0.12	0.08

H_g

Generating an Identical Image

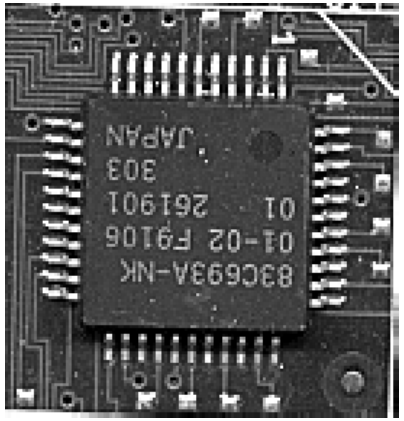


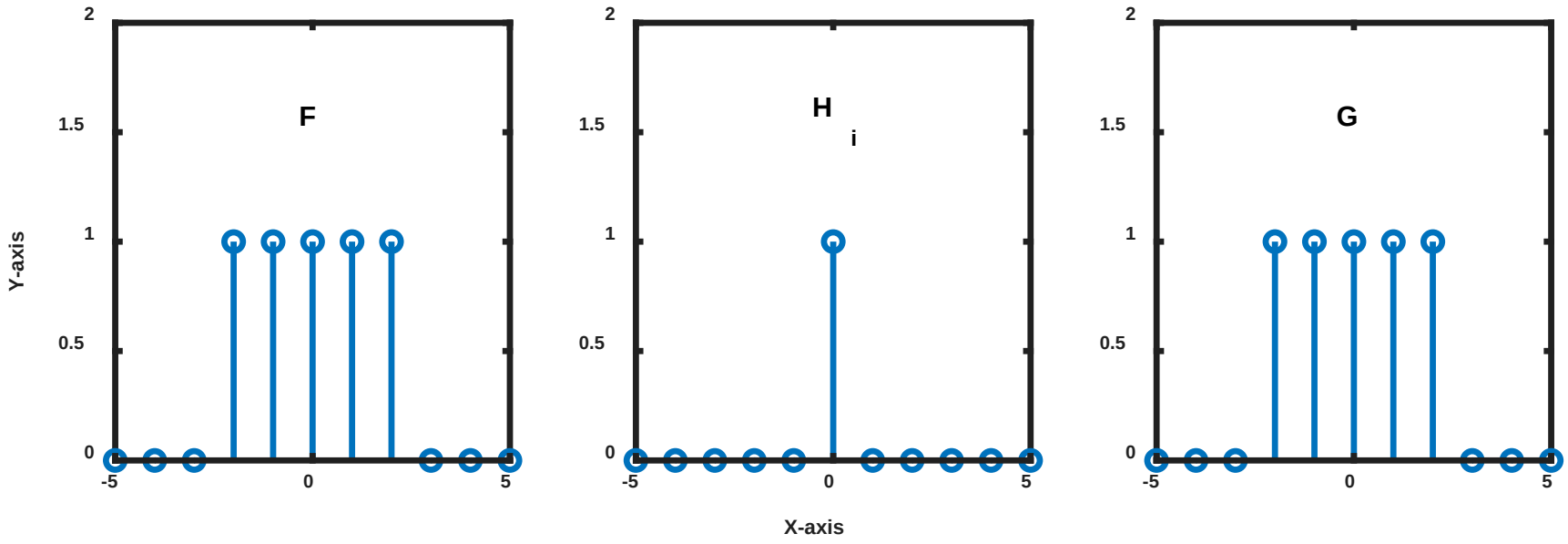
$*$

0	0	0
0	1	0
0	0	0

H_i

$=$

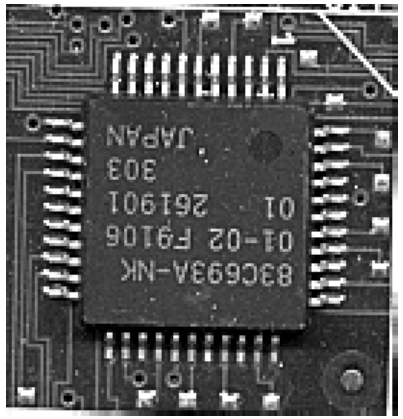




$$F * H_i = G$$

Tutorial

Shifting Pixels

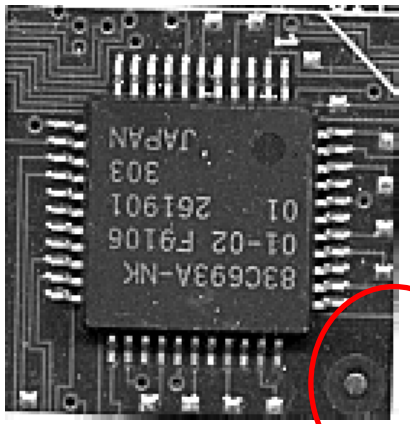


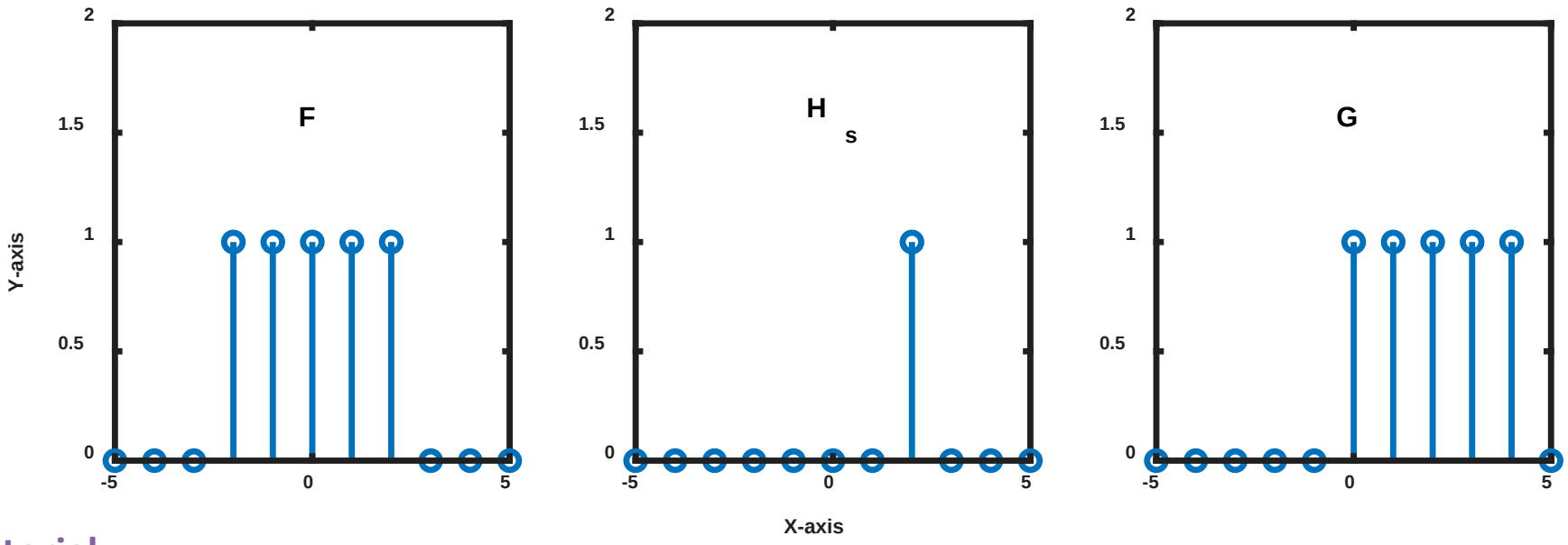
$*$

0	0	0
0	0	1
0	0	0

H_s

$=$

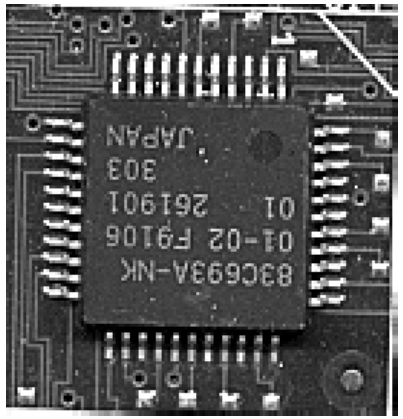




$$F * H_s = G$$

Tutorial

Blurring

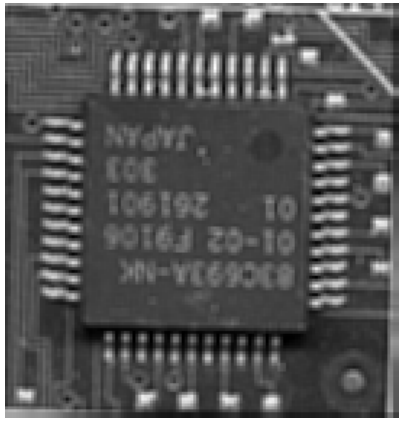


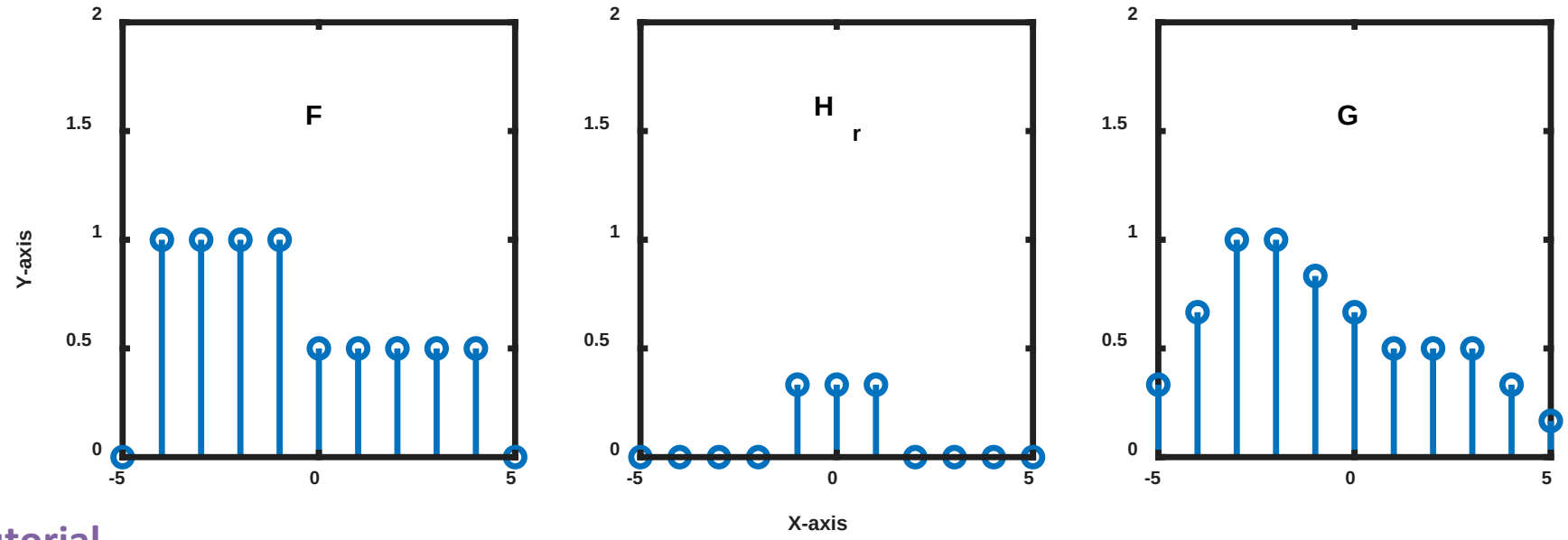
$*$

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

H_r

$=$

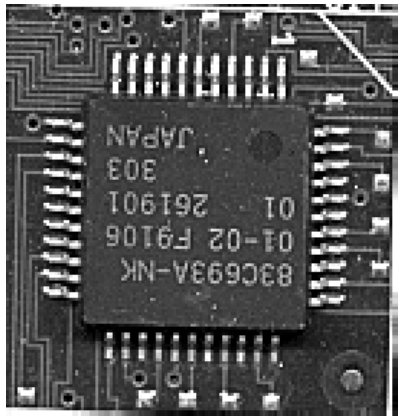




$$F * H_r = G$$

Tutorial

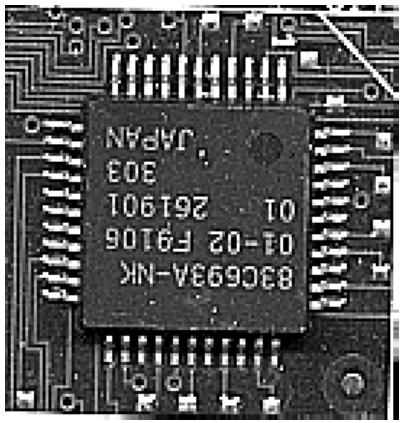
Sharpening

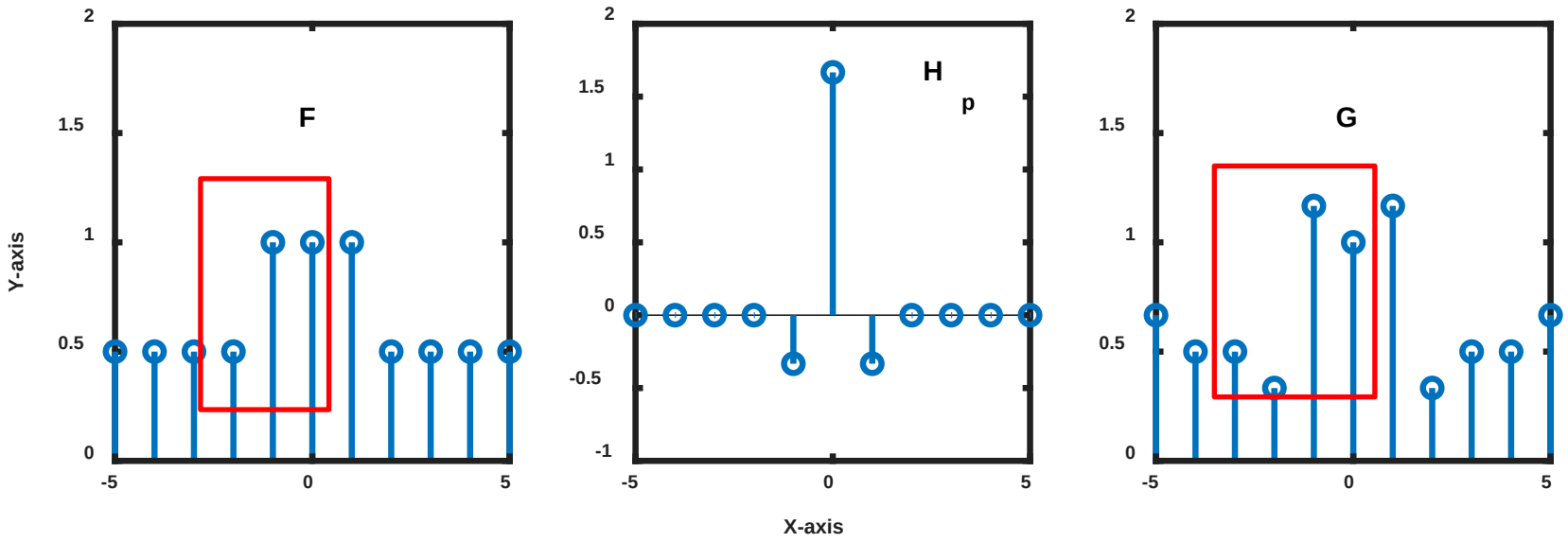


$*$

-0.11	-0.11	-0.11
-0.11	1.89	-0.11
-0.11	-0.11	-0.11

$=$





Making the difference larger

Q. What's difference?

$$F * H_p = G$$

Tutorial

How the Sharpening Filter Works

0	0	0
0	2	0
0	0	0

$2H_i$

$-$

0.11	0.11	0.11
0.11	0.11	0.11
0.11	0.11	0.11

H_r

$=$

-0.11	-0.11	-0.11
-0.11	1.89	-0.11
-0.11	-0.11	-0.11

H_p

$$F * H_p = F * (H_i + H_i - H_r) = F + \boxed{(F - F * H_r)}$$

0	0	0
0	α+1	0
0	0	0

$-\ \alpha$

0.11	0.11	0.11
0.11	0.11	0.11
0.11	0.11	0.11

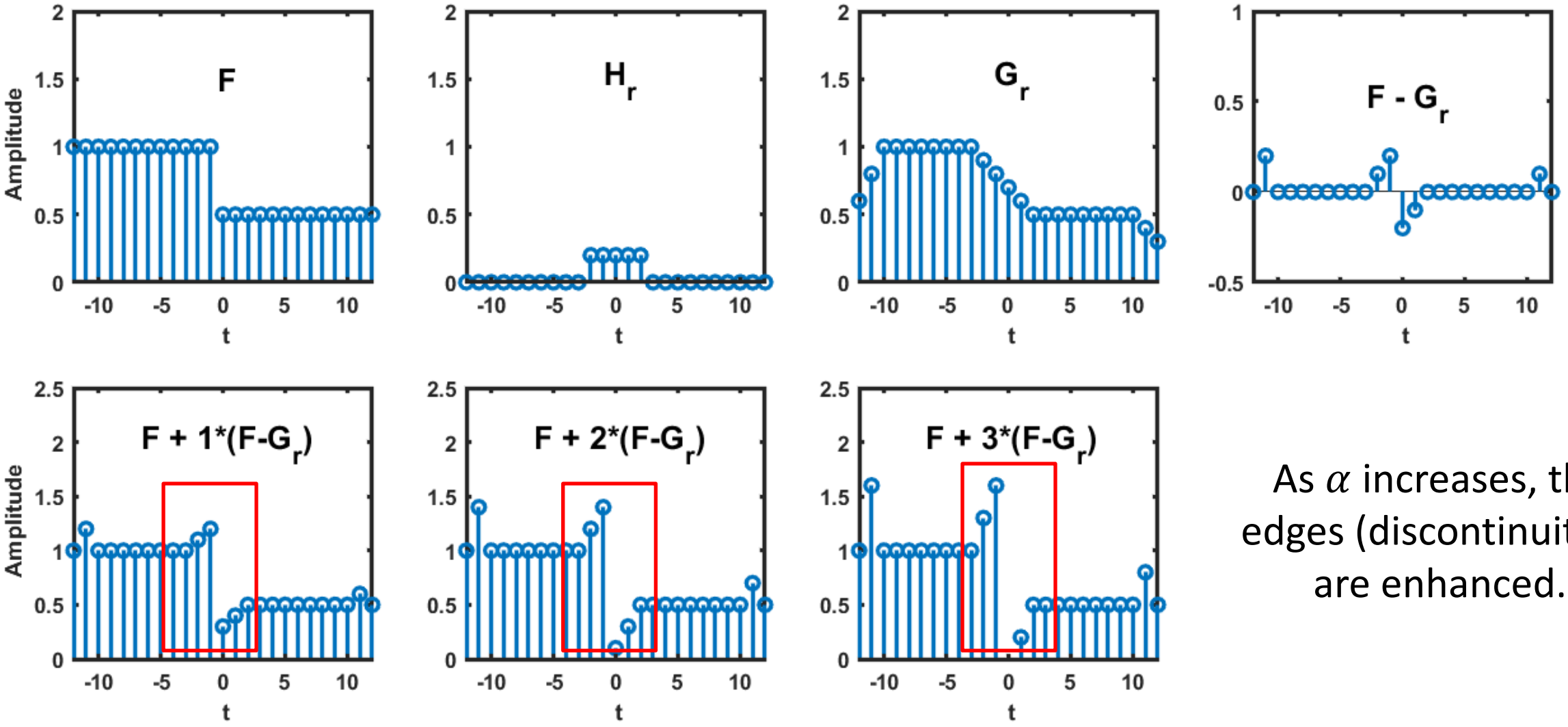
$=$

-0.11α	-0.11α	-0.11α
-0.11α	1+0.89α	-0.11α
-0.11α	-0.11α	-0.11α

Adding this component in the original image

$$F + \alpha(F - F * H_r)$$

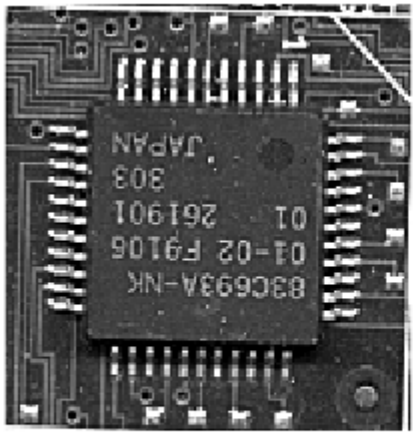
How the Sharpening Filter Works (Signal Example) (Continue)



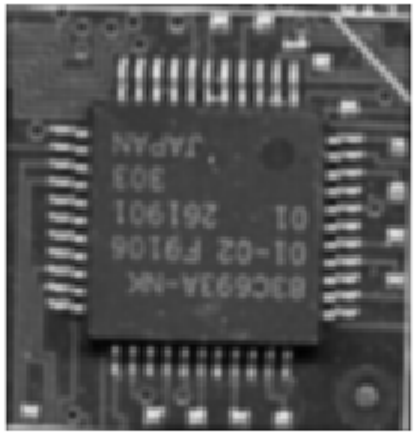
As α increases, the edges (discontinuities) are enhanced.

Tutorial

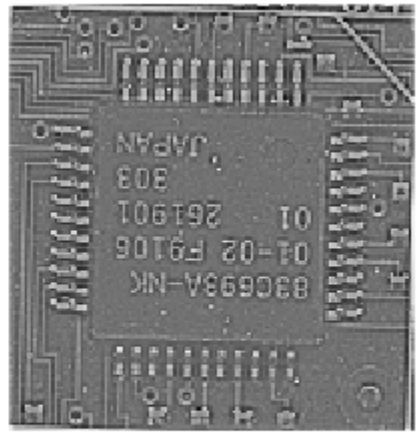
How the Sharpening Filter Works (Image Example)



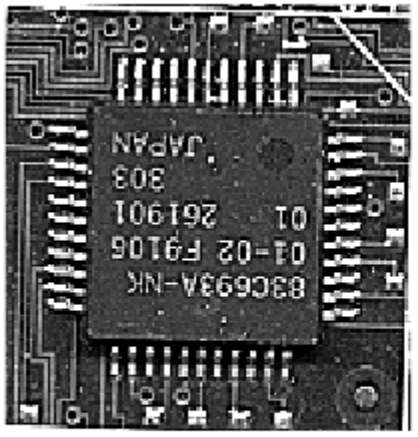
F



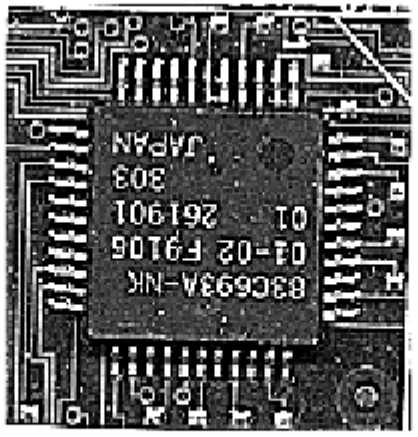
G_r



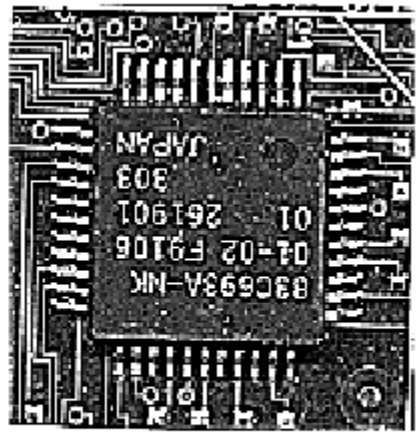
$F - G_r$



$F + (F - G)$



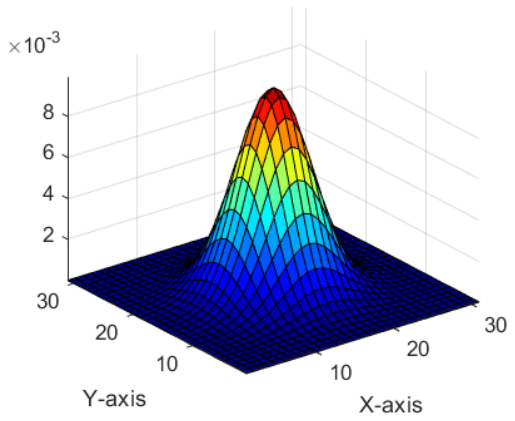
$F + 3(F - G)$

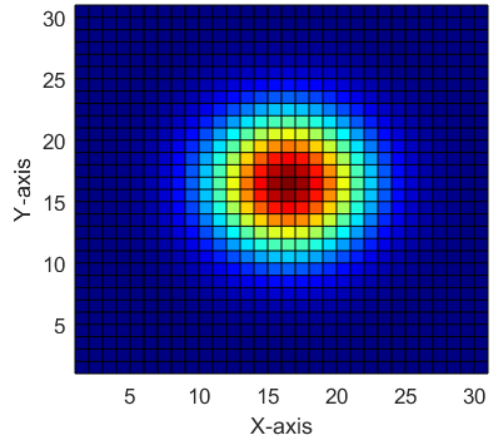


$F + 5(F - G)$

Gaussian Filter

```
clear; close all; clc;

h = fspecial('gaussian',3,1);

numSize = 3;
[x, y] = meshgrid(1:numSize);
x = x-(round(numSize/2));
y = y-(round(numSize/2));
sigma = 1;

G_sigma = 1/(2*pi*sigma^2)*exp(-(x.^2 + y.^2)/(2*sigma^2));
G_sigma = G_sigma/sum(G_sigma,'all');

figure(1);
subplot(121); PlotMat(h,gca,'float'); % from MATLAB built-in function
subplot(122); PlotMat(G_sigma,gca,'float'); % from equation

fig2 = figure(2);
subplot(121); h = fspecial('gaussian', 31,4);
surf(h); axis tight; colormap(jet)
set(fig2,'Position', [100 100 800 300]);
xlabel('X-axis'); ylabel('Y-axis');

subplot(122); surf(h); view(0,90);
xlabel('X-axis'); ylabel('Y-axis'); axis tight
```





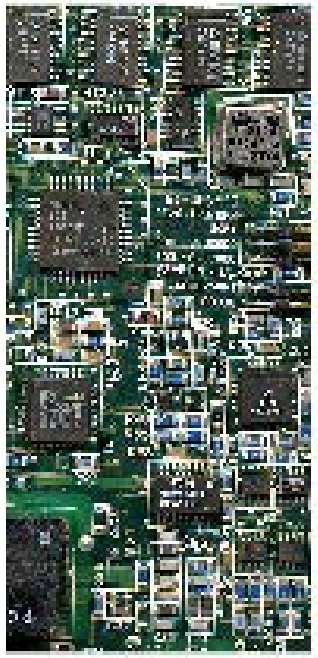
0.08	0.12	0.08
0.12	0.20	0.12
0.08	0.12	0.08

0.08	0.12	0.08
0.12	0.20	0.12
0.08	0.12	0.08

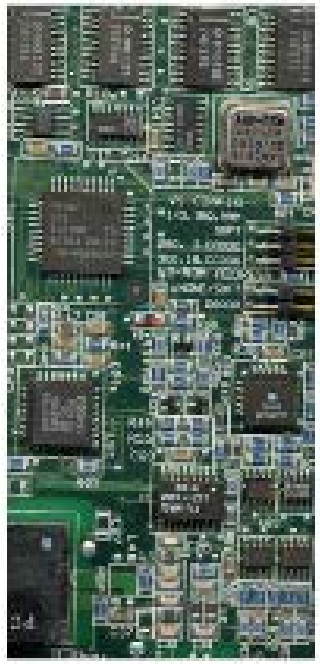
$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Tutorial

Effect of Gaussian Window Sizes

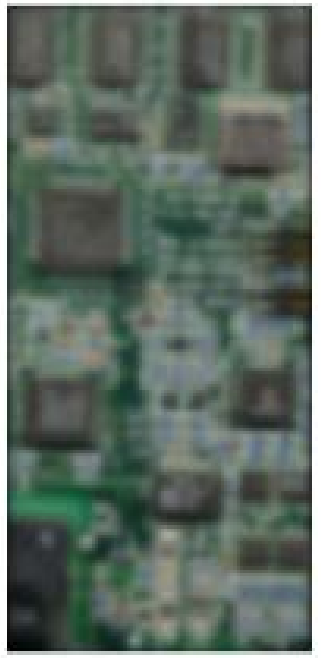


Original



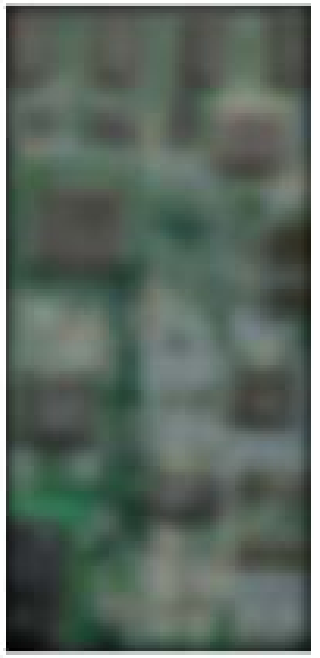
$f1$

σ =1



$f2$

σ =5



$f3$

σ =10

$f4$

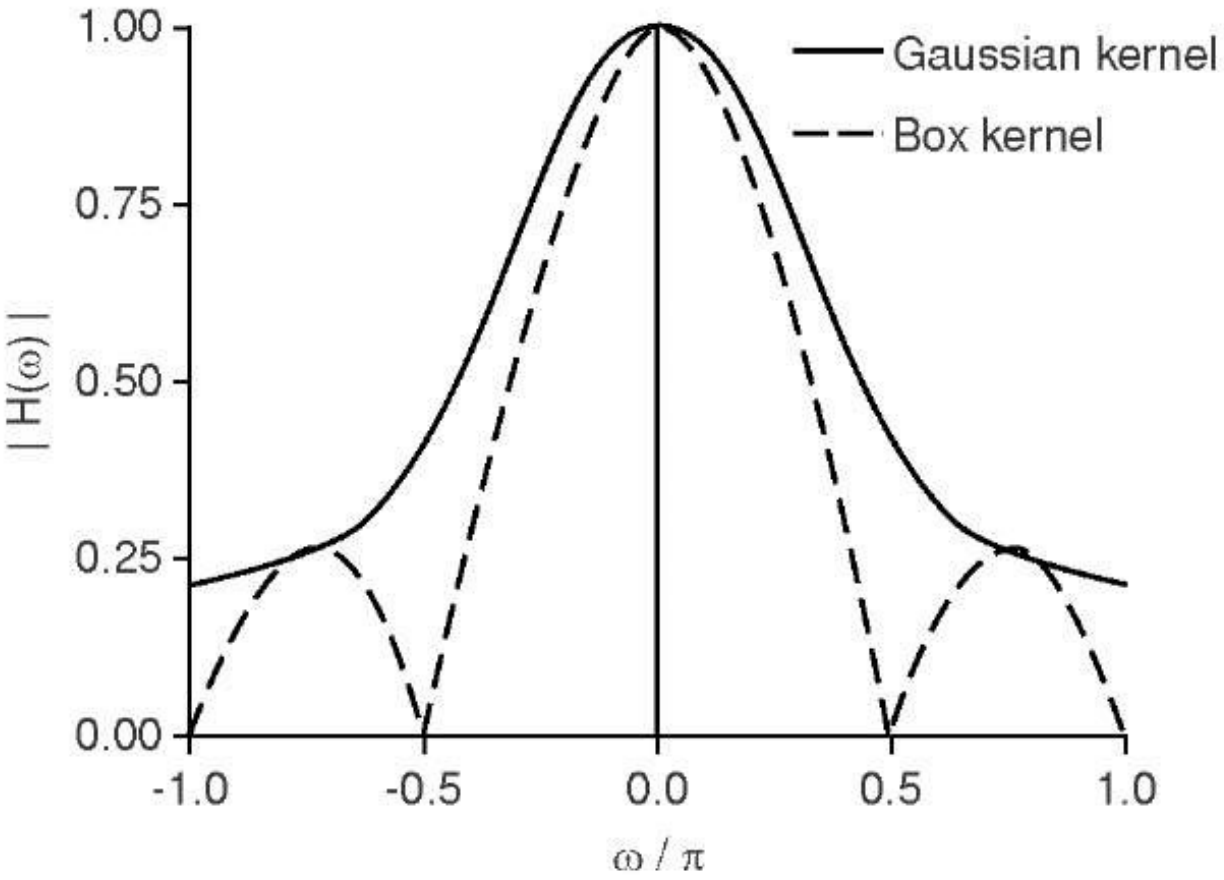
σ =30

```
f1 =  fspecial('gaussian', 101,1);
f2 =  fspecial('gaussian', 101,5);
f3 =  fspecial('gaussian', 101,10);
f4 =  fspecial('gaussian', 101,30);
```

Tutorial

Difference of the Between a Gaussian Filter and Box Filter

Supplement



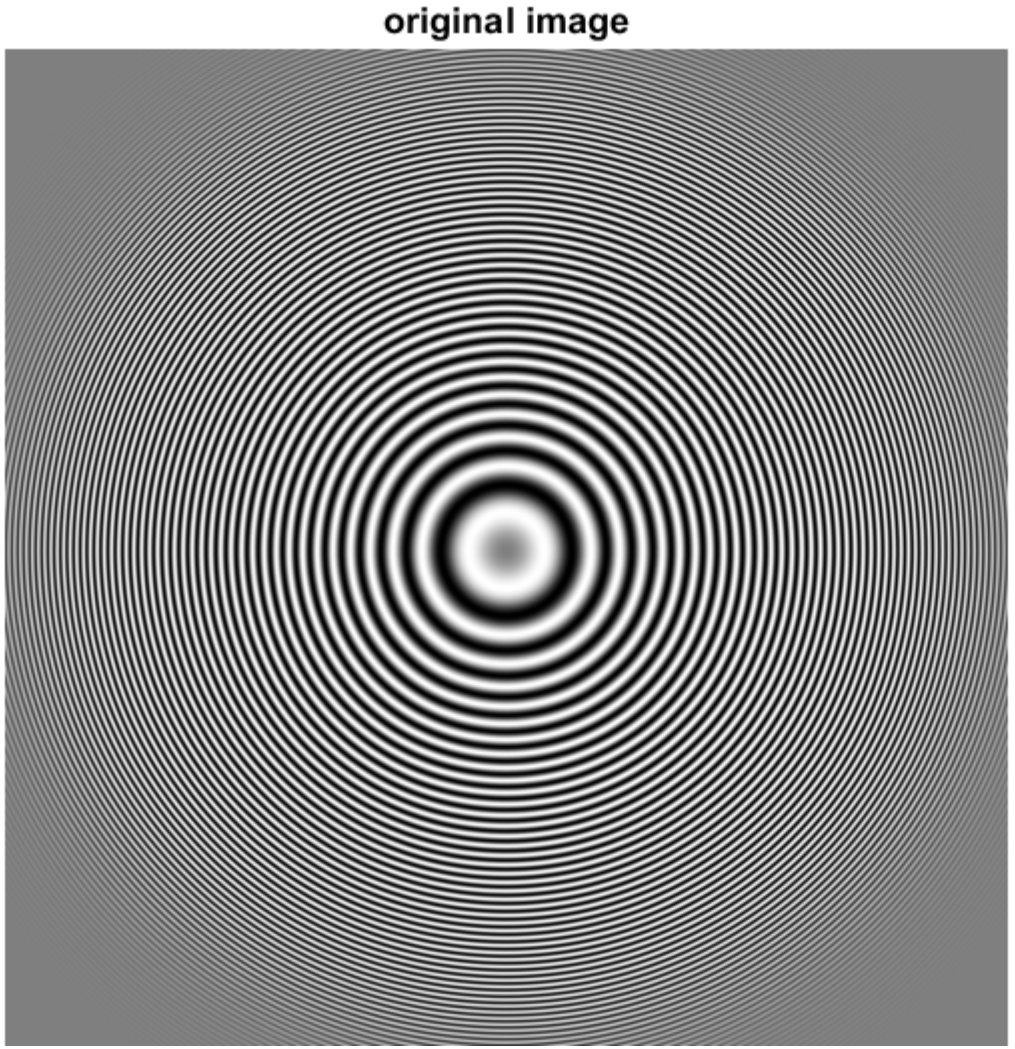
Gaussian

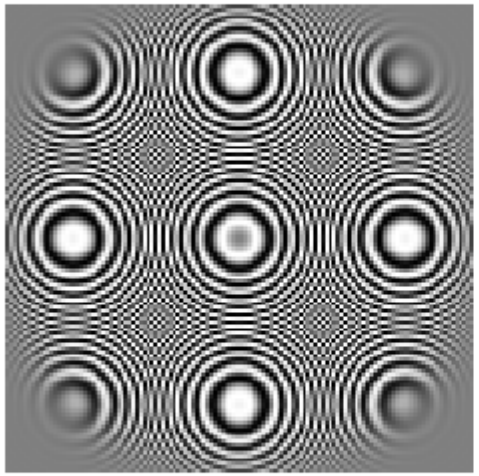
Box

Tutorial

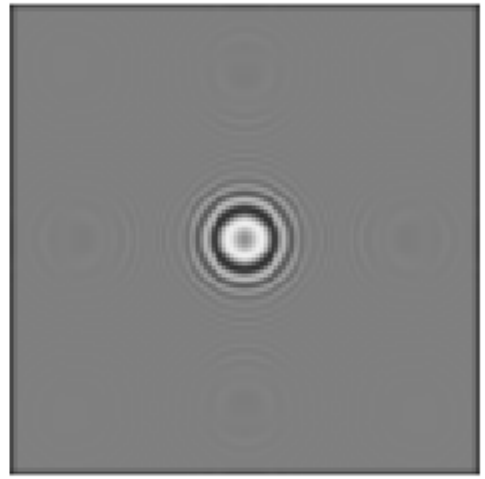
Image Resizing

Supplement





Resizing without applying gaussian filter



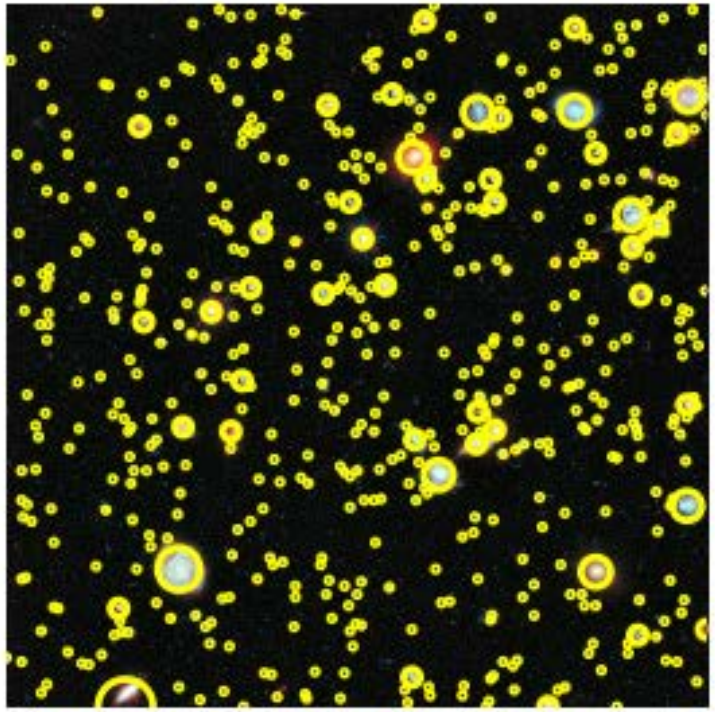
Resizing after applying gaussian filter

Tutorial

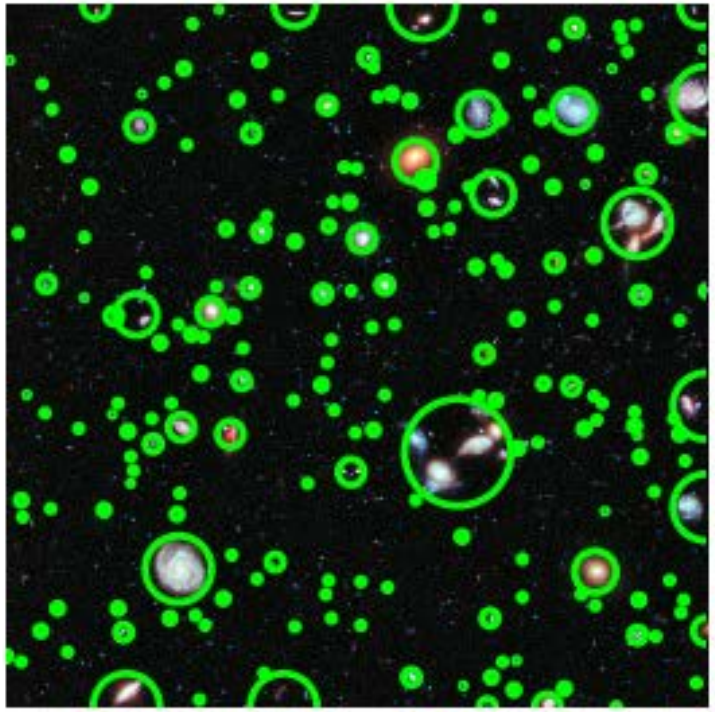
https://blogs.mathworks.com/steve/2017/01/16/aliasing-and-image-resizing-part-3/

Applications: Blob Detection (for Feature Detection)

Laplacian of Gaussian



Difference of Gaussian



Determinant of Hessian