JACOBS UNIVERSITY

COMPUTER VISION
LECTURE 5 – QUIZ

Prof. Dr. Francesco Maurelli
2018-09-18

JACOBS UNIVERSITY

COMPUTER VISION
LECTURE 5 – PIXELS AND FILTERS

Prof. Dr. Francesco Maurelli
2018-09-18

WHAT WE WILL LEARN TODAY?

JACOBS UNIVERSITY

1. Image sampling and quantization
2. Image histograms
3. Images as functions
4. Linear systems (filters)
5. Convolution and correlation

Some background reading:
Forsyth and Ponce, Computer Vision, Chapter 7

TYPES OF IMAGES

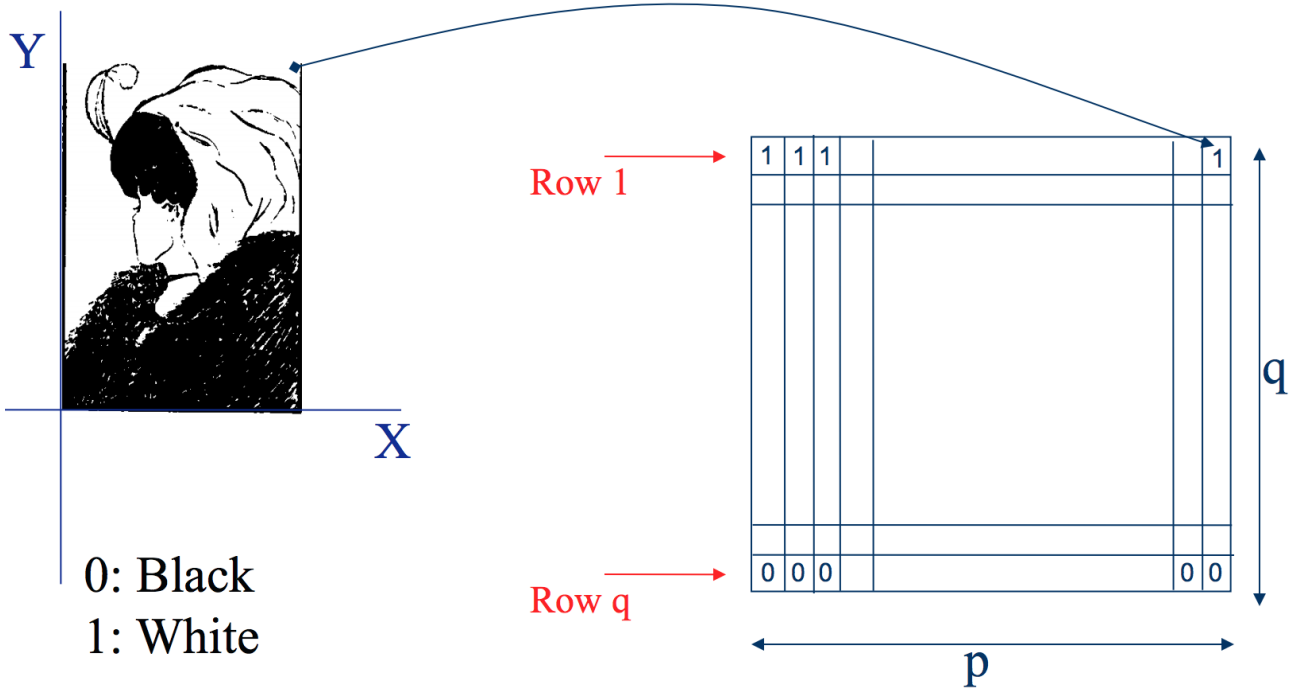
JACOBS UNIVERSITY

Binary

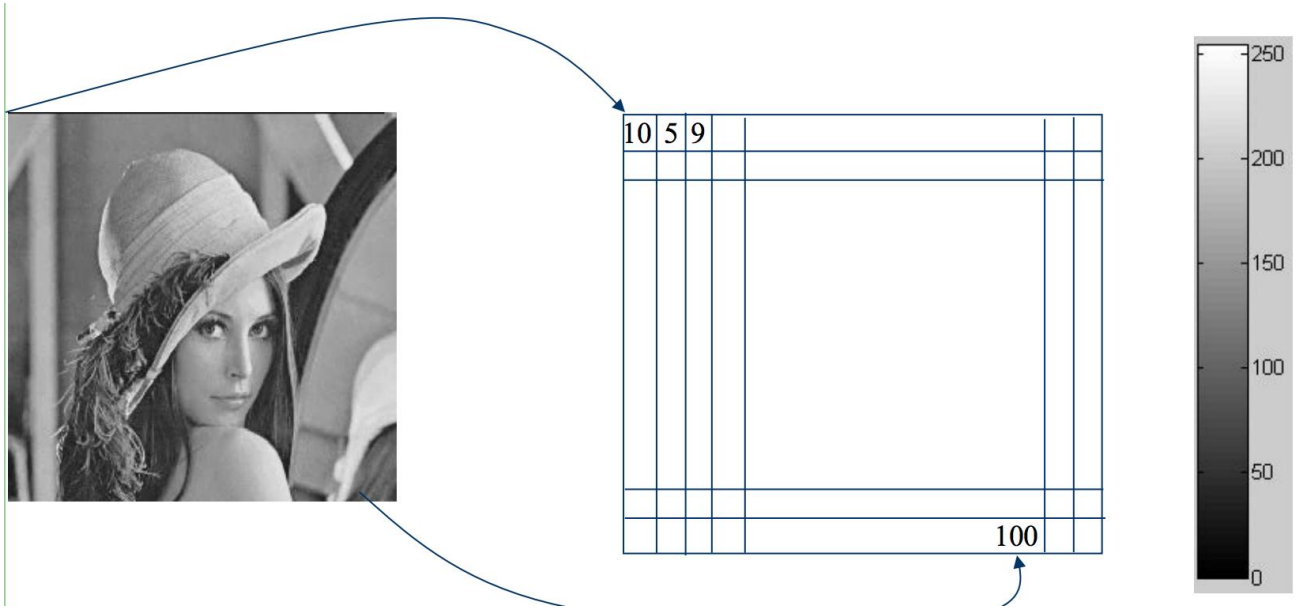
Gray Scale

Color

BINARY IMAGE REPRESENTATION

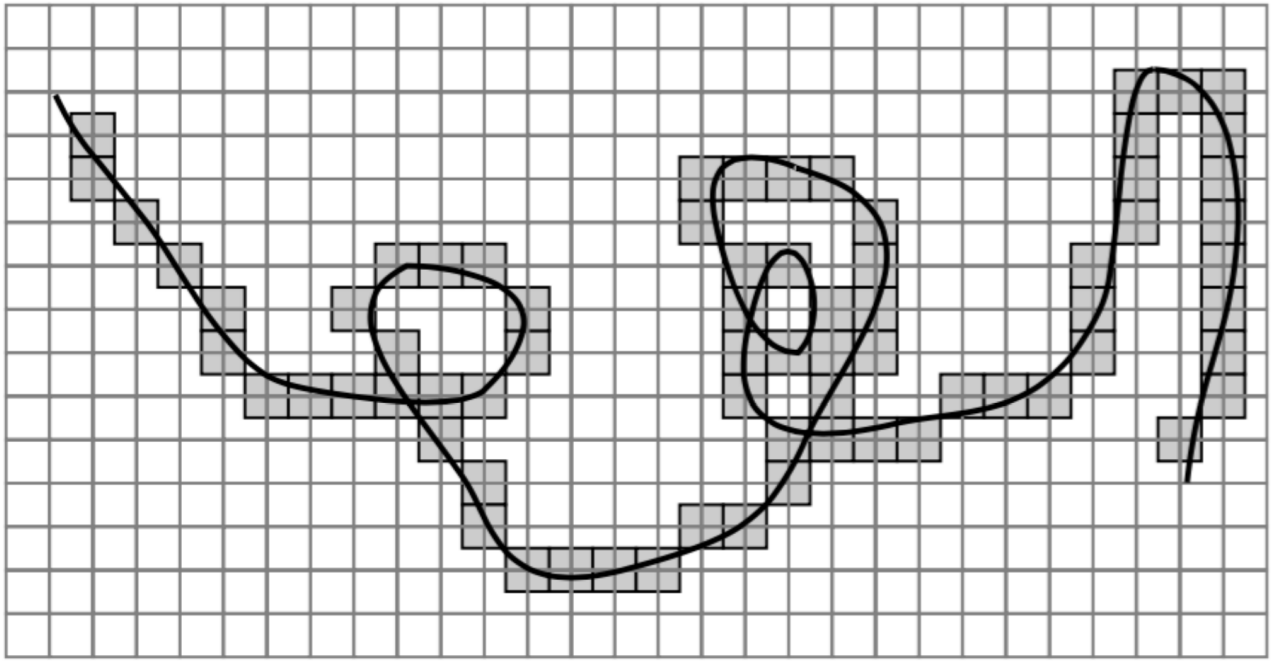


GRAYSCALE IMAGE REPRESENTATION



COLOR IMAGE - ONE CHANNEL

ERRORS DUE SAMPLING



RESOLUTION

is a **sampling** parameter, defined in dots per inch (DPI) or equivalent measures of spatial pixel density, and its standard value for recent screen technologies is 72 dpi

WHAT WE WILL LEARN TODAY?

JACOBS UNIVERSITY

1. Image sampling and quantization
2. Image histograms
3. Images as functions
4. Linear systems (filters)
5. Convolution and correlation

Some background reading:
Forsyth and Ponce, Computer Vision, Chapter 7

HISTOGRAM

1. Histogram of an image provides the frequency of the brightness (intensity) value in the image.

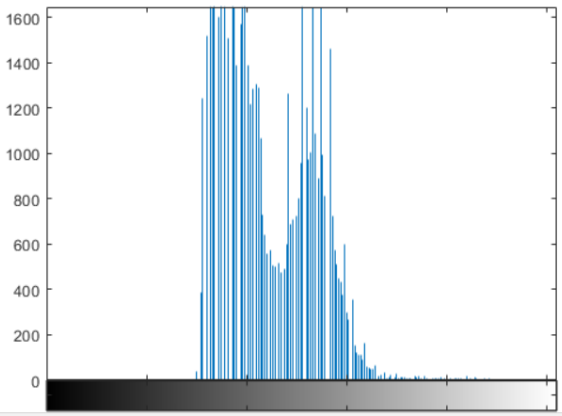
Calculate Histogram

Read a grayscale image into the workspace.

```
I = imread('pout.tif');
```

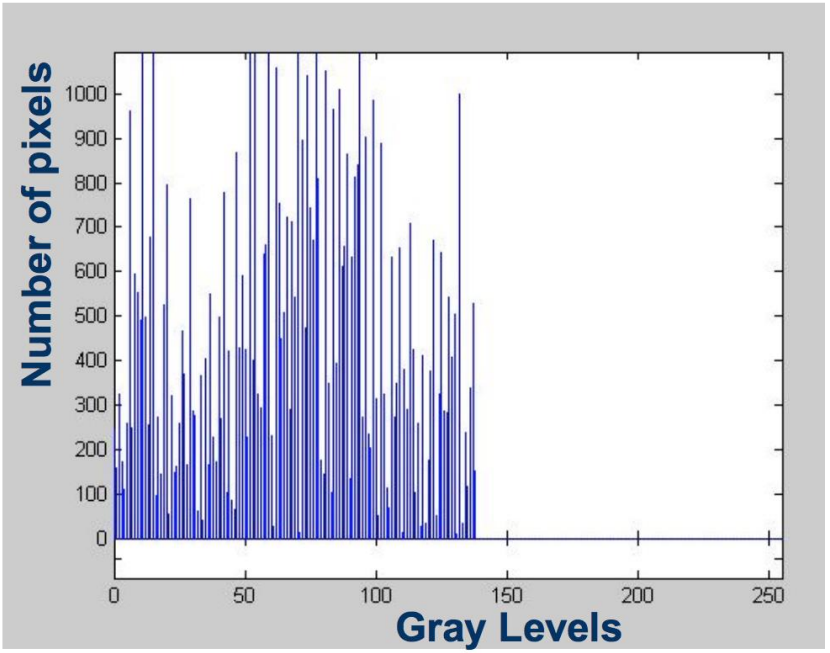
Display a histogram of the image. Since `I` is grayscale, by default the histogram will have 256 bins.

```
imhist(I)
```



HISTOGRAM

1. Histogram captures the distribution of gray levels in the image.
2. How frequently each gray level occurs in the image



WHAT WE WILL LEARN TODAY?

JACOBS UNIVERSITY

1. Image sampling and quantization
2. Image histograms
3. Images as functions
4. Linear systems (filters)
5. Convolution and correlation

Some background reading:
Forsyth and Ponce, Computer Vision, Chapter 7

JACOBS UNIVERSITY

Cartesian coordinates

$$f[n,m] = \begin{bmatrix} \ddots & & \vdots & & \\ & f[-1,-1] & f[0,-1] & f[1,-1] & \\ \dots & f[-1,0] & \underline{f[0,0]} & f[1,0] & \dots \\ & f[-1,1] & f[0,1] & f[1,1] & \\ & & \vdots & & \ddots \end{bmatrix}$$

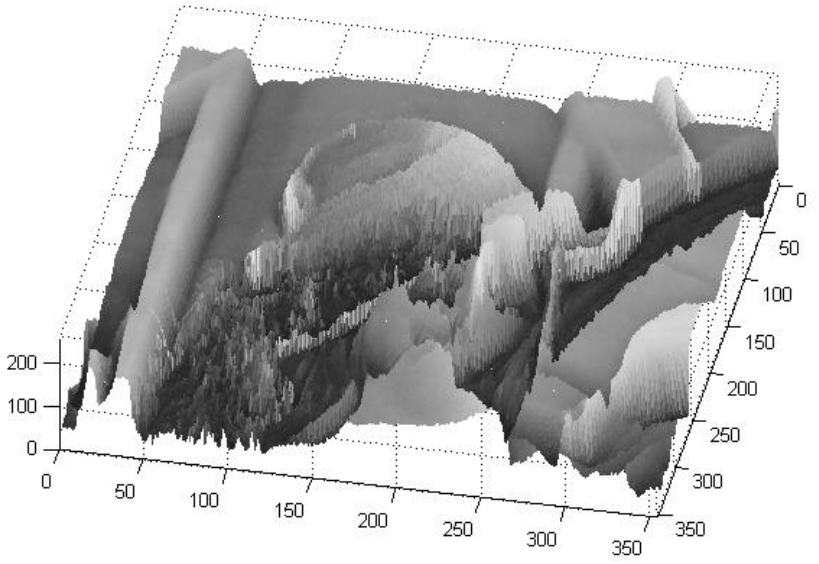
Notation for discrete functions

IMAGES AS FUNCTIONS

- **An Image** as a function f from R^2 to R^M:
  - $f(x, y)$ gives the **intensity** at position (x, y)
  - Defined over a rectangle, with a finite range:

    $f: [a,b] \times [c,d] \rightarrow [0,255]$

    Domain support — range



WHAT WE WILL LEARN TODAY?

1. Image sampling and quantization
2. Image histograms
3. Images as functions
4. Linear systems (filters)
5. Convolution and correlation

Some background reading:
Forsyth and Ponce, Computer Vision, Chapter 7

JACOBS UNIVERSITY

Filtering:

– Forming a new image whose pixel values are transformed from original pixel values

Goals:

- Goal is to extract useful information from images, or transform images into another domain where we can modify/enhance image properties
  - Features (edges, corners, blobs…)
  - super-resolution; in-painting; de-noising

JACOBS UNIVERSITY

1. we define a system as a unit that converts an input function f[n,m] into an output (or response) function g[n,m], where (n,m) are the independent variables.
   - In the case for images, (n,m) represents the **spatial position in the image.**

$$f[n,m] \rightarrow \boxed{\text{System } \mathcal{S}} \rightarrow g[n,m]$$

DE-NOISING

Salt and pepper noise

Super-resolution

In-painting

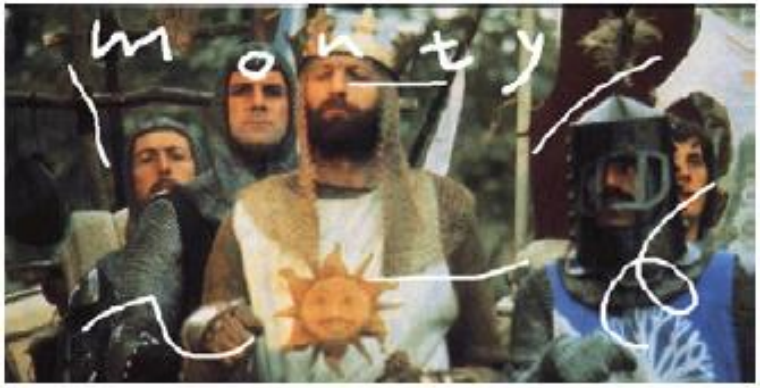


Image Inpainting, M. Bertalmío et al.
http://www.iua.upf.es/~mbertalmio//restoration.html

JACOBS UNIVERSITY

Cartesian coordinates

$$f[n,m] = \begin{bmatrix} \ddots & & \vdots & & \\ & f[-1,-1] & f[0,-1] & f[1,-1] & \\ \dots & f[-1,0] & \underline{f[0,0]} & f[1,0] & \dots \\ & f[-1,1] & f[0,1] & f[1,1] & \\ & & \vdots & & \ddots \end{bmatrix}$$

Notation for discrete functions

JACOBS UNIVERSITY

S is the **system operator**, defined as a mapping or assignment of a member of the set of possible outputs g[n,m] to each member of the set of possible inputs f[n,m].

$$f[n,m] \rightarrow \boxed{\text{System } \mathcal{S}} \rightarrow g[n,m]$$

$$g = \mathcal{S}[f], \quad g[n,m] = \mathcal{S}\{f[n,m]\}$$

$$f[n,m] \xrightarrow{\mathcal{S}} g[n,m]$$

FILTER EXAMPLE #1: MOVING AVERAGE

Moving average over a 3×3 window of neighborhood

$$g[n,m] = \frac{1}{9} \sum_{k=n-1}^{n+1} \sum_{l=m-1}^{m+1} f[k,l]$$

$$= \frac{1}{9} \sum_{k=-1}^{1} \sum_{l=-1}^{1} f[n-k, m-l]$$

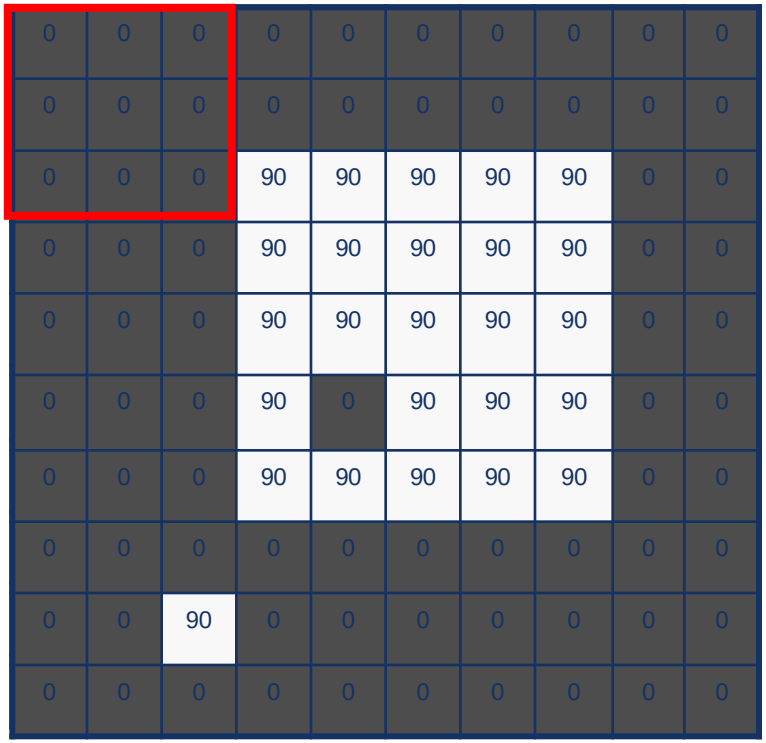
h

$\frac{1}{9}$

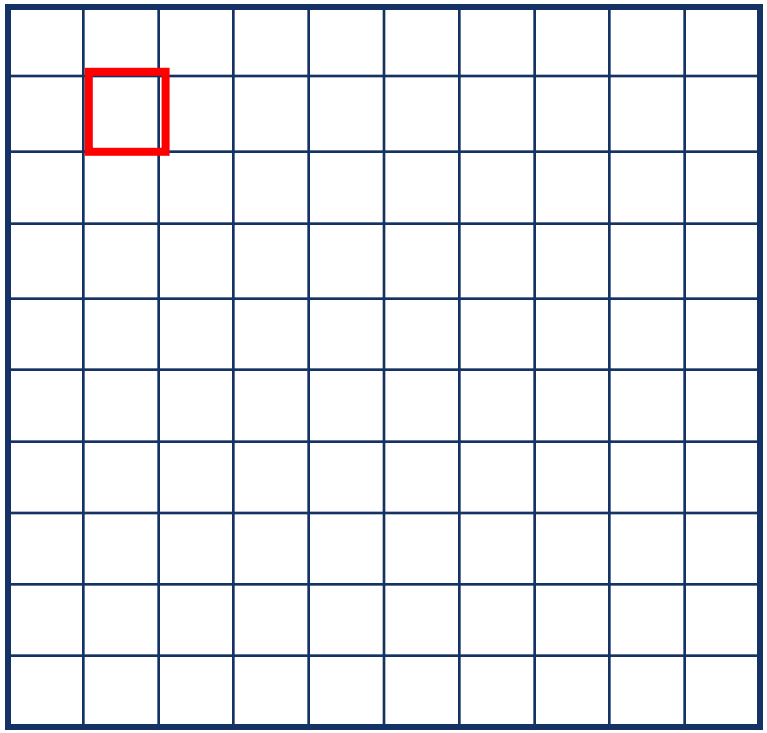
1	1	1
1	1	1
1	1	1

FILTER EXAMPLE #1: MOVING AVERAGE

$F[x, y]$



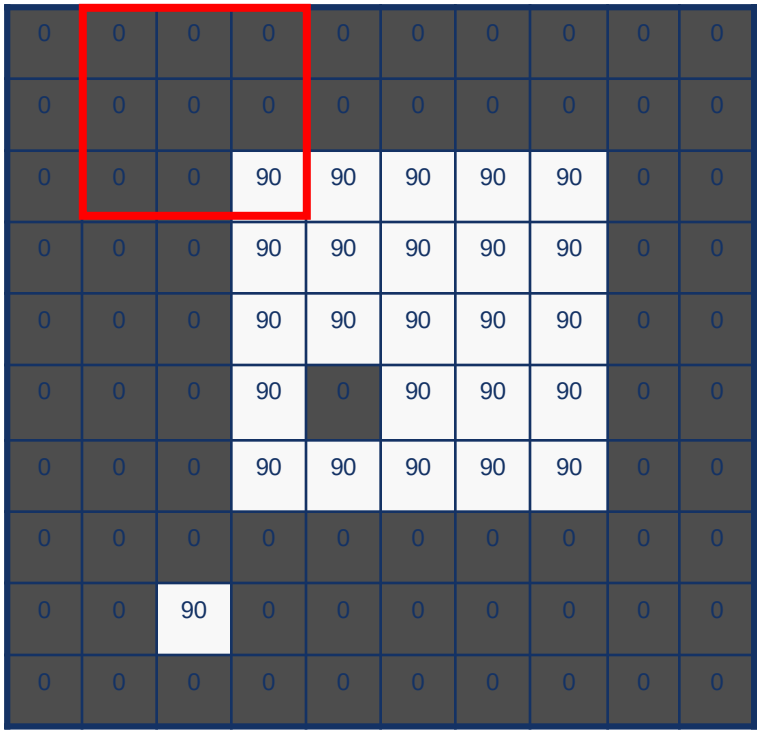
$G[x, y]$



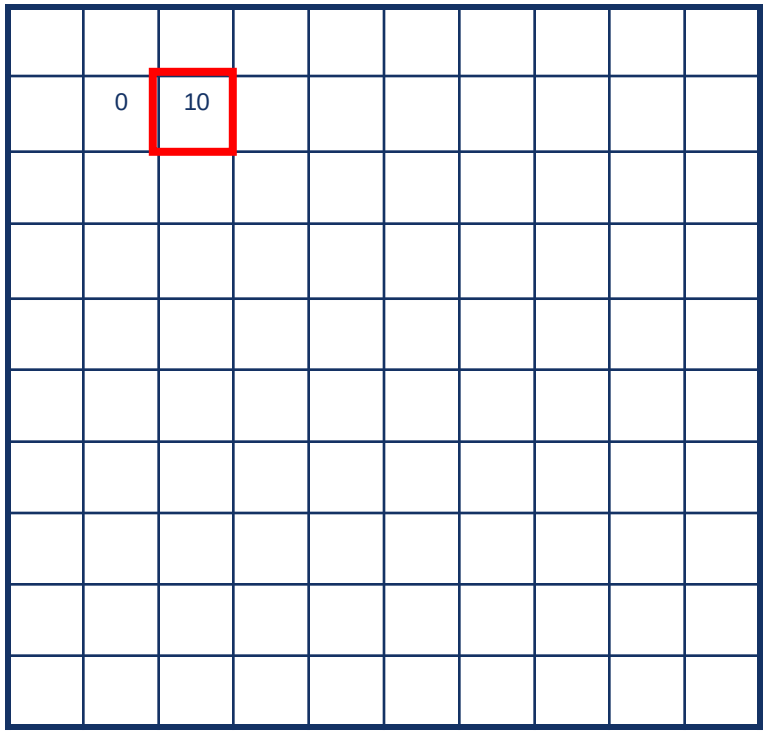
$$(f * h)[m,n] = \sum_{k,l} f[k,l]\, h[m-k, n-l]$$

FILTER EXAMPLE #1: MOVING AVERAGE

$F[x, y]$



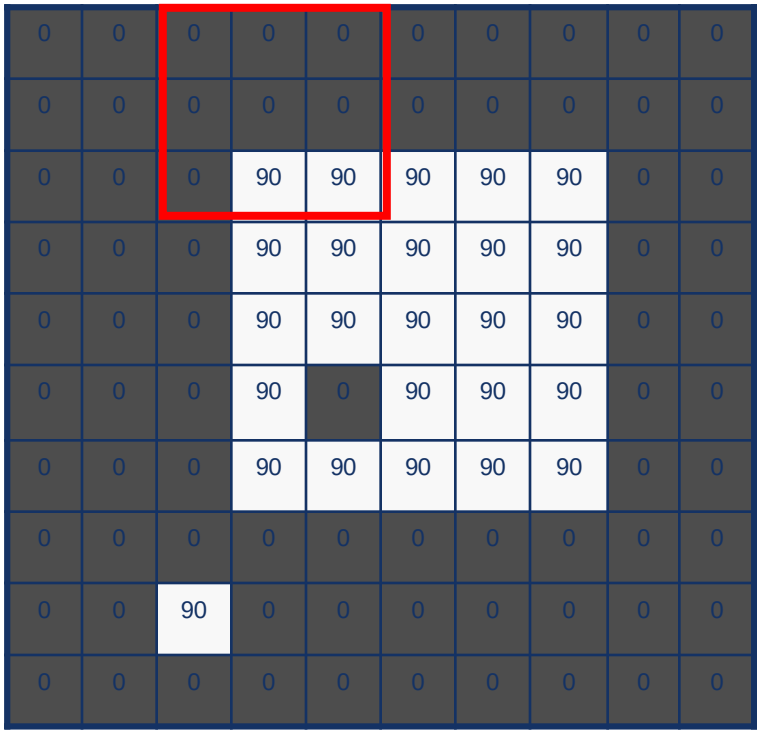
$G[x, y]$



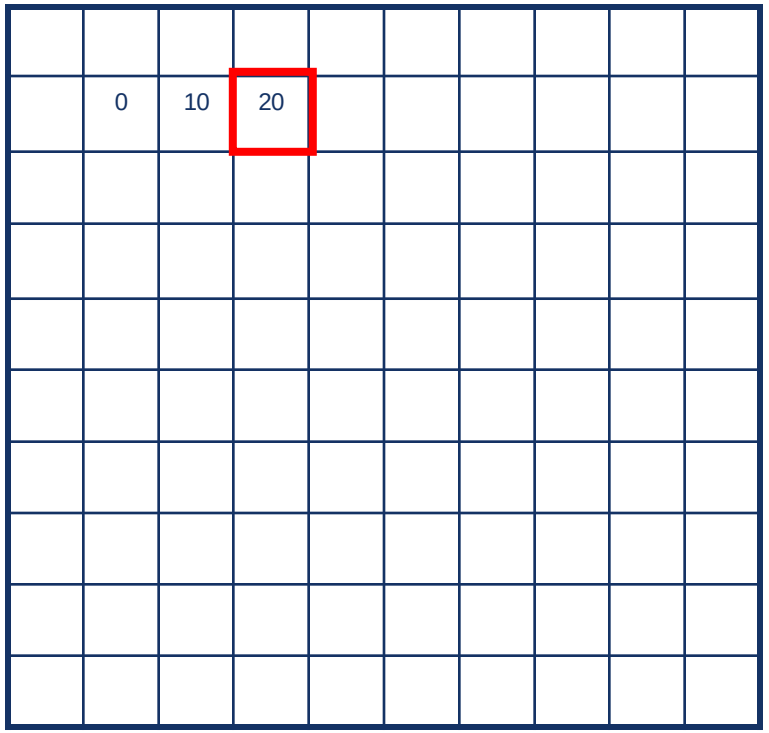
$$(f * h)[m,n] = \sum_{k,l} f[k,l]\, h[m-k, n-l]$$

FILTER EXAMPLE #1: MOVING AVERAGE

$F[x, y]$



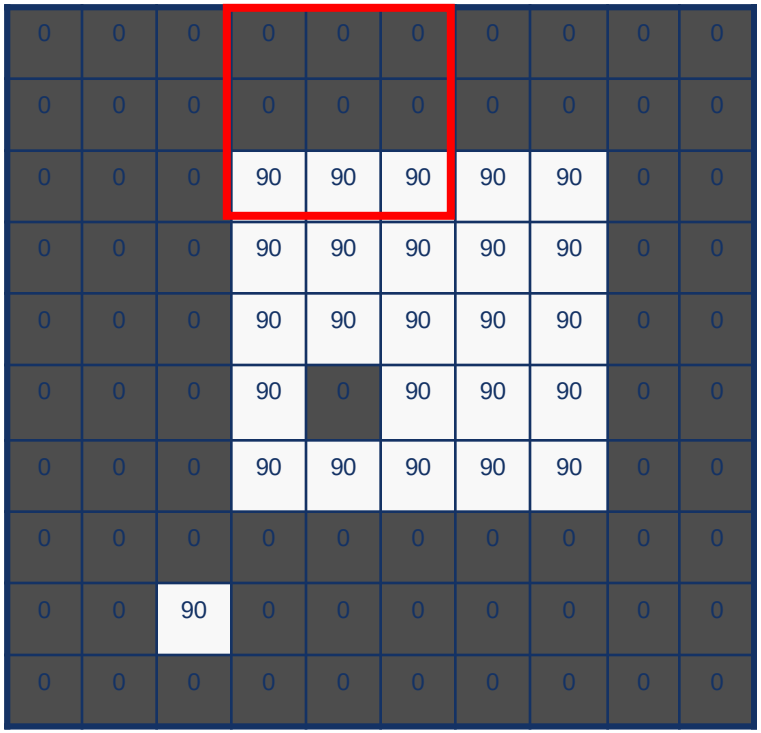
$G[x, y]$



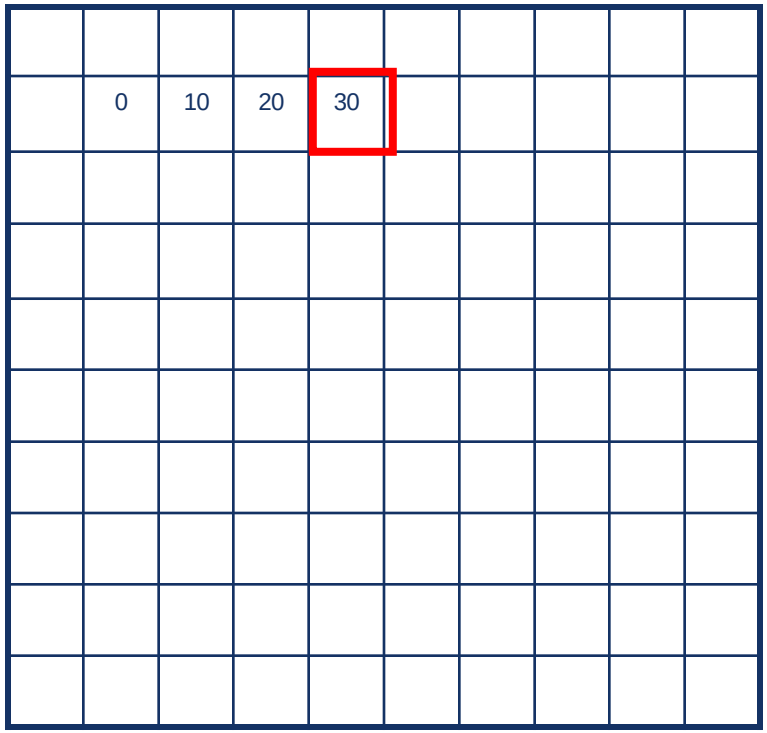
$$(f * h)[m,n] = \sum_{k,l} f[k,l]\, h[m-k, n-l]$$

FILTER EXAMPLE #1: MOVING AVERAGE

$F[x, y]$



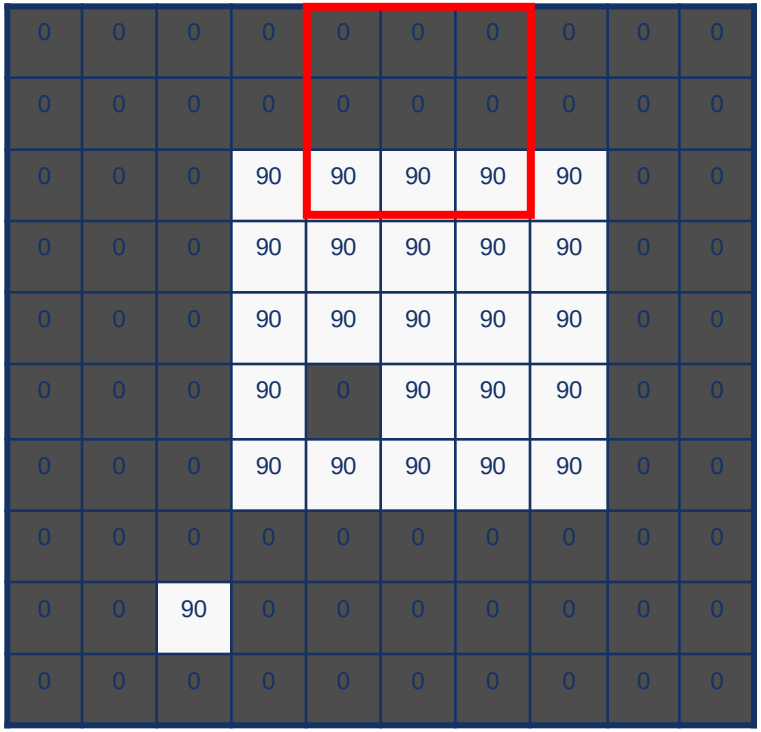
$G[x, y]$



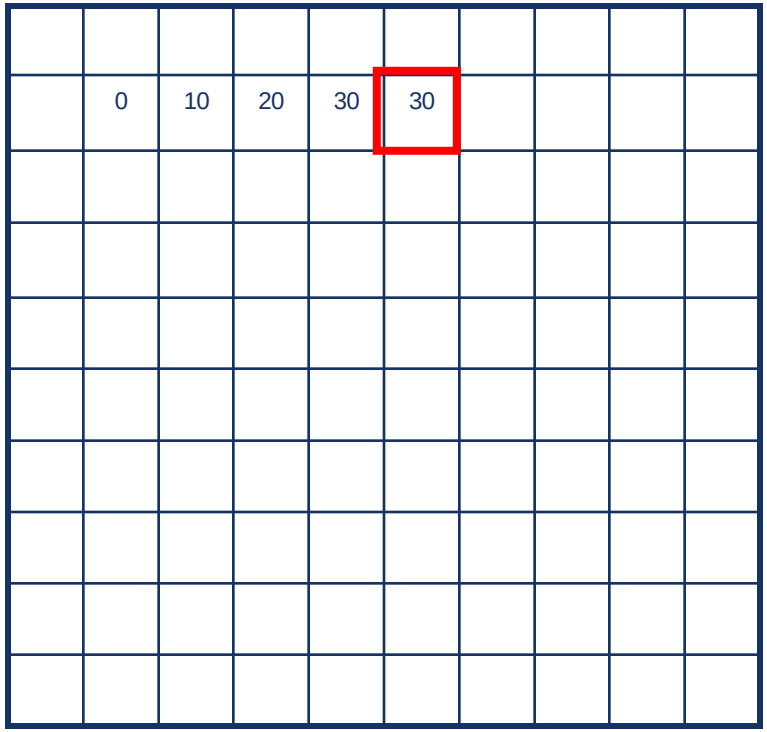
$$(f * h)[m,n] = \sum_{k,l} f[k,l]\, h[m-k, n-l]$$

FILTER EXAMPLE #1: MOVING AVERAGE

$F[x, y]$



$G[x, y]$



$$(f * h)[m,n] = \sum_{k,l} f[k,l]\, h[m-k, n-l]$$

FILTER EXAMPLE #1: MOVING AVERAGE

$F[x, y]$

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

$G[x, y]$

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

$$(f * h)[m, n] = \sum_{k,l} f[k, l]\, h[m - k, n - l]$$

In summary:

- This filter "Replaces" each pixel with an average of its neighborhood.
- Achieve smoothing effect (remove sharp features)

$$h[x, x]$$

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

FILTER EXAMPLE #1: MOVING AVERAGE

FILTER EXAMPLE #2: IMAGE SEGMENTATION

JACOBS UNIVERSITY

1. Image segmentation based on a simple threshold:

$$g[n,m] = \begin{cases} 255, & f[n,m] > 100 \\ 0, & \text{otherwise.} \end{cases}$$

JACOBS UNIVERSITY

Amplitude properties:

- Additivity $S[f_i[n,m] + f_j[n,m]] = S[f_i[n,m]] + S[f_j[n,m]]$
- Homogeneity $S[\alpha f_i[n,m]] = \alpha S[f_i[n,m]]$
- Superposition

$$S[\alpha f_i[n,m] + \beta f_j[n,m]] = \alpha S[f_i[n,m]] + \beta S[f_j[n,m]]$$

- Stability $|f[n,m]| \leq k \implies |g[n,m]| \leq ck$
- Invertibility $S^{-1}[S[f_i[n,m]]] = f[n,m]$

JACOBS UNIVERSITY

Spatial properties

- Causality $\quad \text{for } n < n_0, m < m_0, \text{ if } f[n,m] = 0 \implies g[n,m] = 0$

- Shift invariance

$$f[n - n_0, m - m_0] \xrightarrow{\mathcal{S}} g[n - n_0, m - m_0]$$

IS THE MOVING AVERAGE SYSTEM IS SHIFT INVARIANT?

$$f[n,m] \quad \xrightarrow{\mathcal{S}} \quad g[n,m] = \frac{1}{9} \sum_{k=-1}^{1} \sum_{l=-1}^{1} f[n-k, m-l]$$

$F[x,y]$

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

$G[x,y]$

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

IS THE MOVING AVERAGE SSTEM IS SHIFT INVARIANT?

$$f[n,m] \quad \xrightarrow{\mathcal{S}} \quad g[n,m] = \frac{1}{9}\sum_{k=-1}^{1}\sum_{l=-1}^{1} f[n-k, m-l]$$

$$f[n-n_0, m-m_0]$$

$$\xrightarrow{\mathcal{S}} \quad \frac{1}{9}\sum_{k=-1}^{1}\sum_{l=-1}^{1} f[(n-n_0)-k, (m-m_0)-l]$$

$$= \quad g[n-n_0, m-m_0]$$

Yes!

IS THE MOVING AVERAGE SYSTEM IS CASUAL?

$$f[n,m] \quad \xrightarrow{\mathcal{S}} \quad g[n,m] = \frac{1}{9} \sum_{k=-1}^{1} \sum_{l=-1}^{1} f[n-k, m-l]$$

$F[x,y]$

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

$G[x,y]$

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

$$\text{for } n < n_0, m < m_0, \text{ if } f[n,m] = 0 \implies g[n,m] = 0$$

$$f[n,m] \rightarrow \boxed{\text{System } \mathcal{S}} \rightarrow g[n,m]$$

- Linear filtering:
  - Form a new image whose pixels are a weighted sum of original pixel values
  - Use the same set of weights at each point

- **S** is a linear system (function) iff it *S satisfies*

$$S[\alpha f_i[n,m] + \beta f_j[h,m]] = \alpha S[f_i[n,m]] + \beta S[f_j[h,m]]$$

superposition property

$$f[n,m] \rightarrow \boxed{\text{System } \mathcal{S}} \rightarrow g[n,m]$$

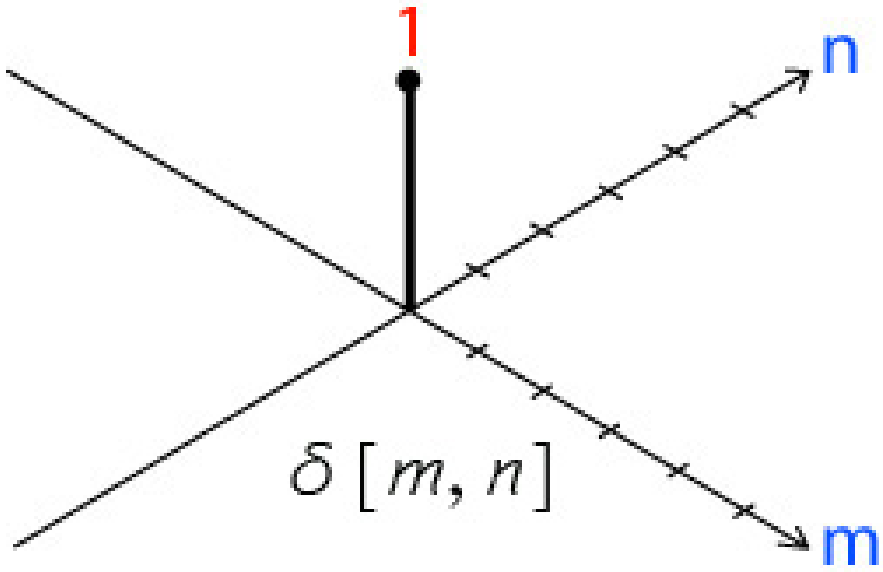
- Is the moving average a linear system?

- Is thresholding a linear system?
  - f1[n,m] + f2[n,m] > T
  - f1[n,m] < T
  - f2[n,m]<T

No!

2D IMPULSE FUNCTION

1. 1 at [0,0].
2. 0 everywhere else



JACOBS UNIVERSITY

Impulse response

$$\delta_2[n,m] \rightarrow \boxed{\mathcal{S}} \rightarrow h[n,m]$$

$$\delta_2[n-k,m-l] \rightarrow \boxed{\mathcal{S}\,(\mathrm{SI})} \rightarrow h[n-k,m-l]$$

JACOBS UNIVERSITY

Example: impulse response of the 3 by 3 moving average filter:

$$h[n,m] = \frac{1}{9}\sum_{k=-1}^{1}\sum_{l=-1}^{1}\delta_2[n-k,m-l]$$

$$= \begin{bmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{bmatrix}$$

h

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

WHAT WE WILL LEARN TODAY?

JACOBS UNIVERSITY

1. Images as functions
2. Linear systems (filters)
3. Convolution and correlation

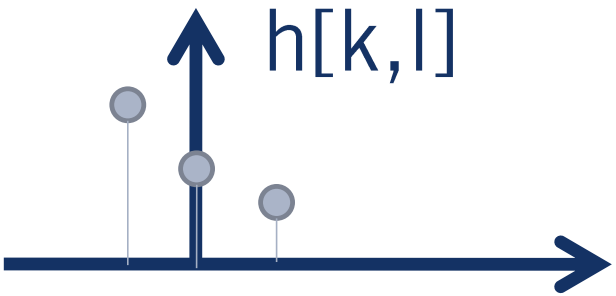
Some background reading:
Forsyth and Ponce, Computer Vision, Chapter 7

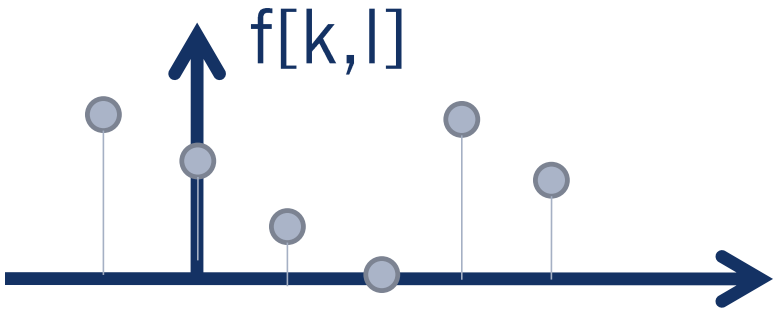
1D DISCRETE CONVOLUTION (SYMBOL: *)

JACOBS UNIVERSITY

We are going to convolve a function **f** with a filter **h.**

$$g[n] = \sum_k f[k]h[n-k]$$



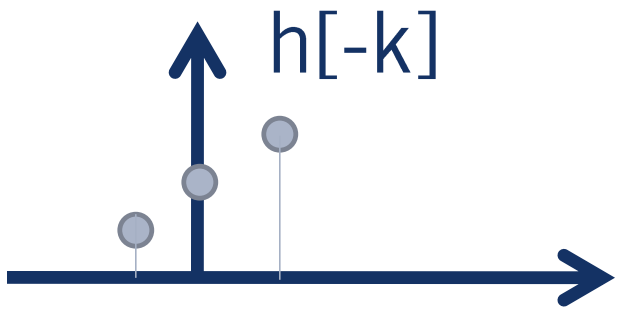


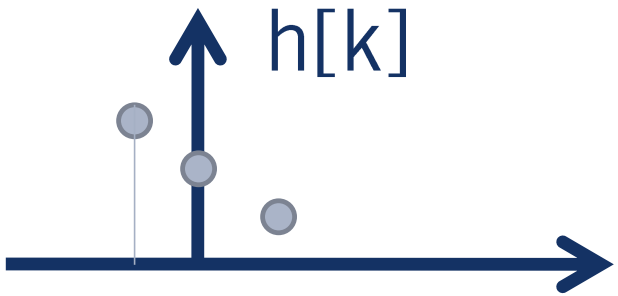
1D DISCRETE CONVOLUTION (SYMBOL: *)

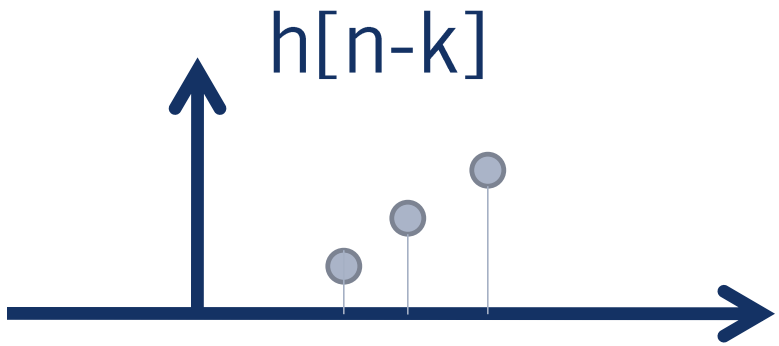
We are going to convolve a function **f** with a filter **h.**

$$g[n] = \sum_k f[k]h[n-k]$$

We first need to calculate h[n-k, m-l]

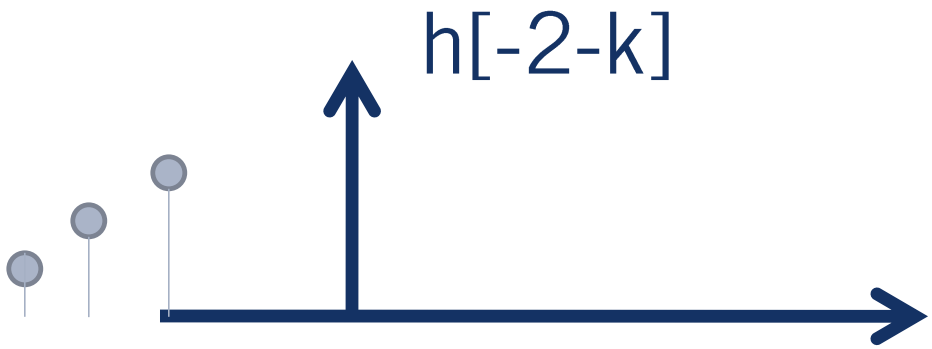


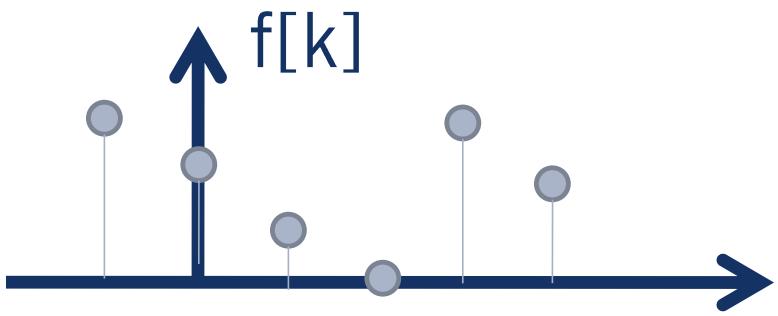


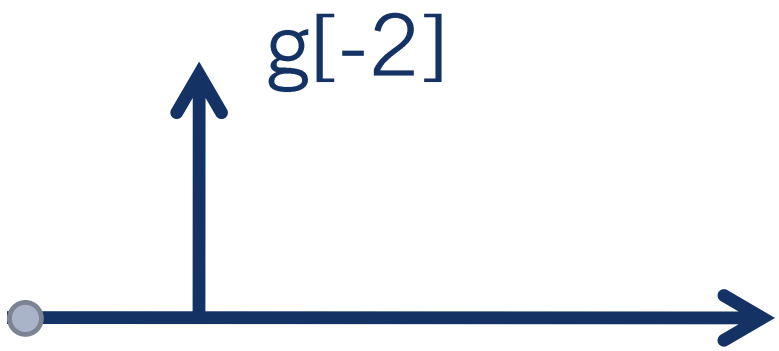


1D DISCRETE CONVOLUTION (SYMBOL: *)

We are going to convolve a function **f** with a filter **h.**

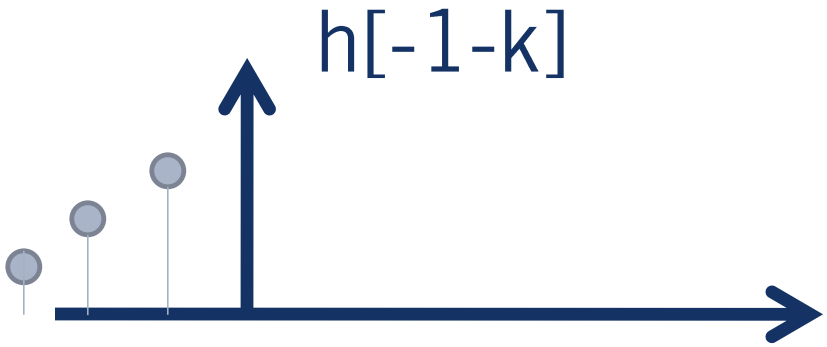


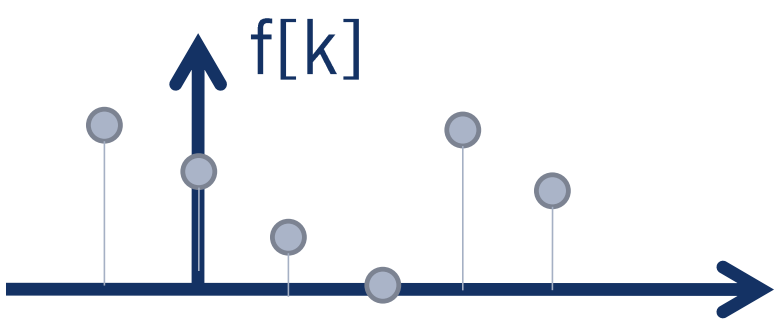




1D DISCRETE CONVOLUTION (SYMBOL: *)

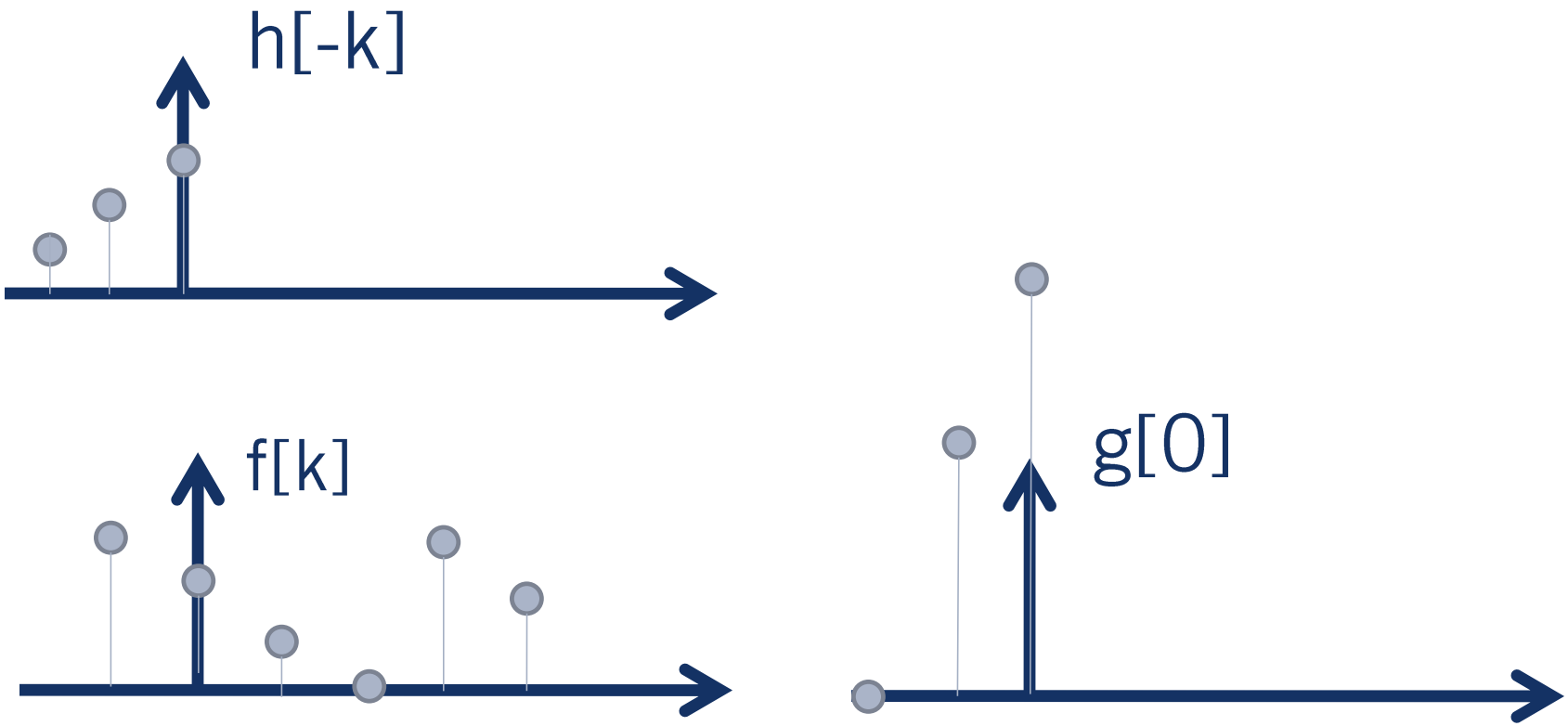
We are going to convolve a function **f** with a filter **h.**





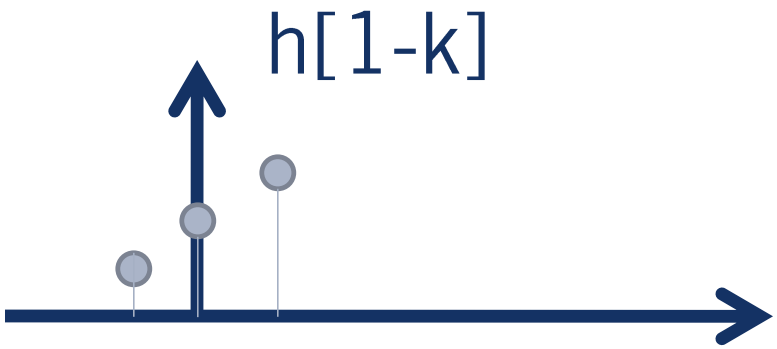
1D DISCRETE CONVOLUTION (SYMBOL: *)

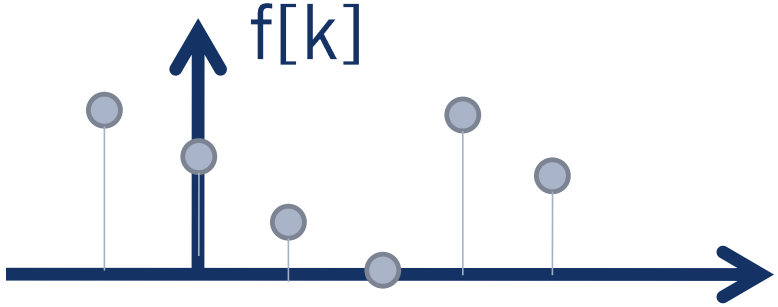
We are going to convolve a function **f** with a filter **h.**

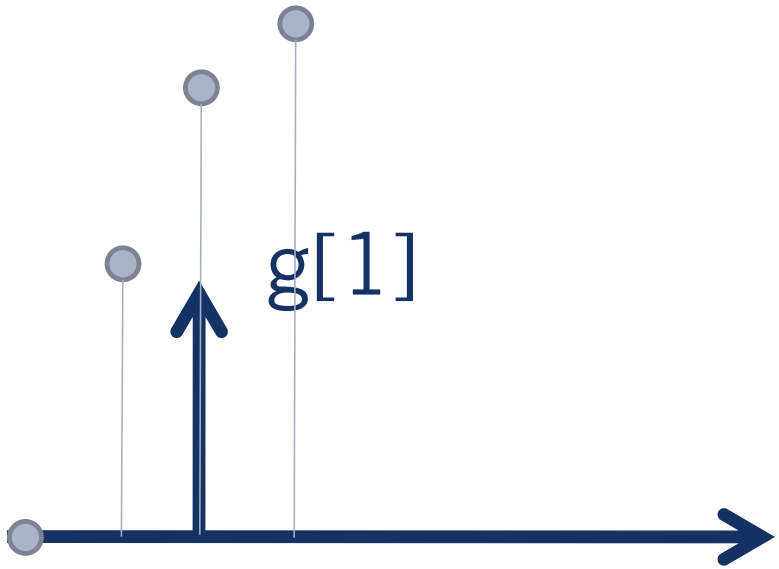


1D DISCRETE CONVOLUTION (SYMBOL: *)

We are going to convolve a function **f** with a filter **h.**



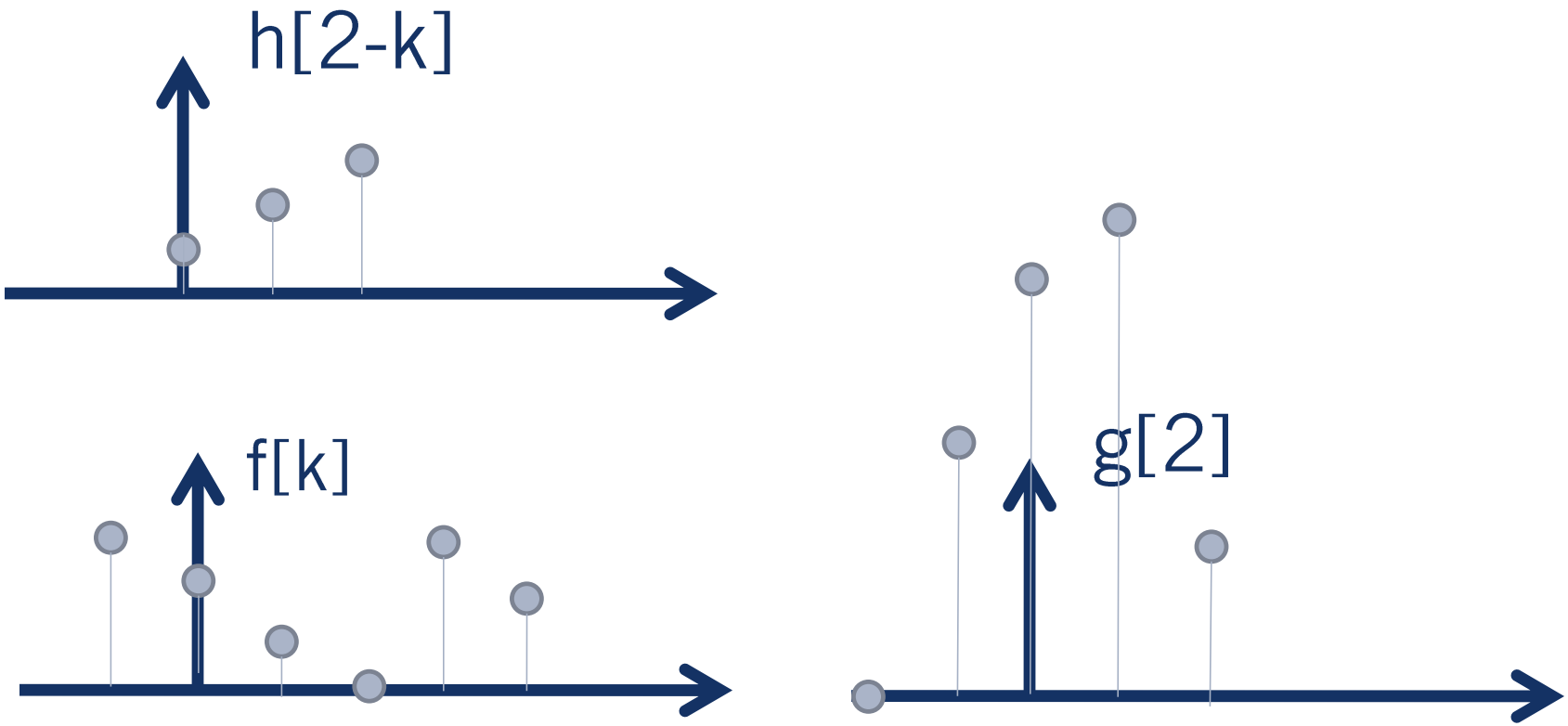




1D DISCRETE CONVOLUTION (SYMBOL: *)

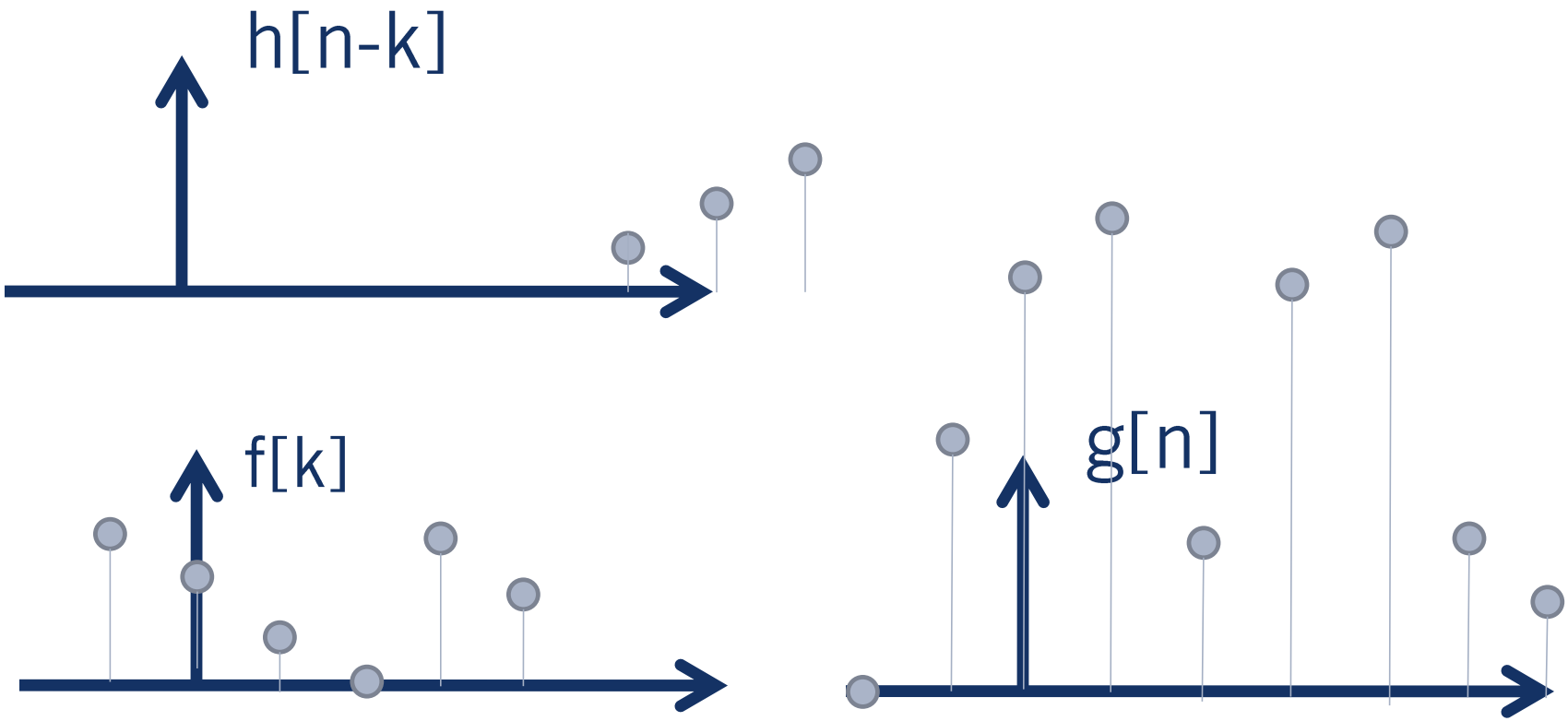
JACOBS UNIVERSITY

We are going to convolve a function **f** with a filter **h.**



1D DISCRETE CONVOLUTION (SYMBOL: *)

We are going to convolve a function **f** with a filter **h.**



1D DISCRETE CONVOLUTION (SYMBOL: *)

JACOBS UNIVERSITY

In summary, the steps for discrete convolution are:

- Fold h[k,l] about origin to form h[–k]
- Shift the folded results by n to form h[n – k]
- Multiply h[n – k] by f[k]
- Sum over all k
- Repeat for every n

$$g[n] = \sum_k f[k][h - k]$$

n

JACOBS UNIVERSITY

In a different notation (g is the kernel now)

$$(f * g)(i) = \sum_{j=1}^{m} g(j) \cdot f(i - j + m/2)$$

$f =$

10	50	60	10	20	40	30

$g =$

1/3	1/3	1/3

10	50	60	10	20	40	30
	1/3	1/3	1/3			

10	50	60	10	20	30	40
0	1/3	1/3	1/3	0	0	0

$$\frac{1}{3}50 + \frac{1}{3}60 + \frac{1}{3}10 = \frac{50}{3} + \frac{60}{3} + \frac{10}{3} = \frac{120}{3} = 40$$

n

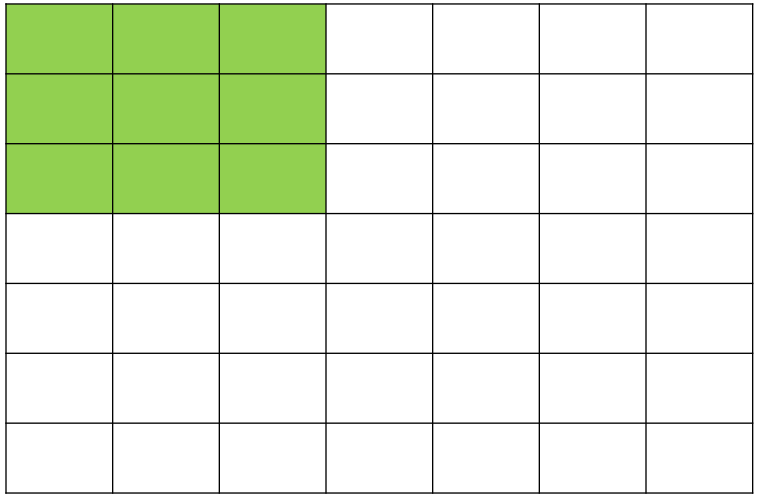
2D CONVOLUTION

JACOBS UNIVERSITY

2D convolution is very similar to 1D.

- The main difference is that we now have to iterate over 2 axis instead of 1.

$$f[n,m] * h[n,m] = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} f[k,l]\, h[n-k, m-l]$$



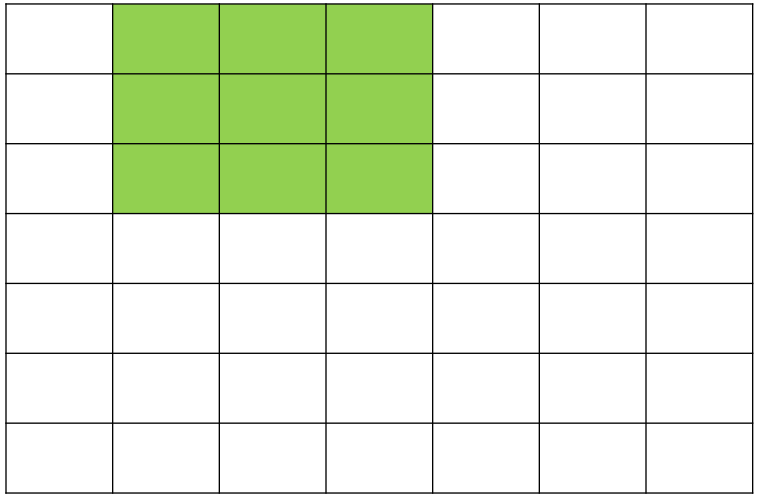
Assume we have a filter(h[,]) that is 3x3. and an image (f[,]) that is 7x7.

JACOBS UNIVERSITY

2D convolution is very similar to 1D.

- The main difference is that we now have to iterate over 2 axis instead of 1.

$$f[n,m] * h[n,m] = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} f[k,l]\, h[n-k, m-l]$$



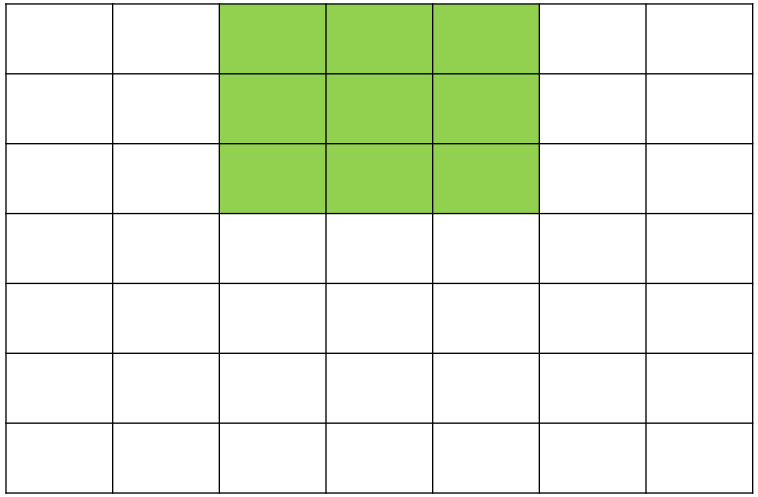
Assume we have a filter(h[,]) that is 3x3. and an image (f[,]) that is 7x7.

JACOBS UNIVERSITY

2D convolution is very similar to 1D.

- The main difference is that we now have to iterate over 2 axis instead of 1.

$$f[n,m] * h[n,m] = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} f[k,l]\, h[n-k, m-l]$$



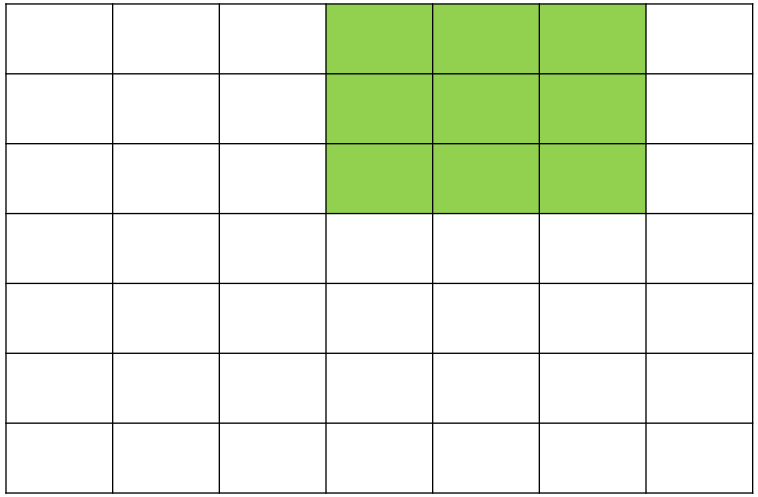
Assume we have a filter(h[,]) that is 3x3. and an image (f[,]) that is 7x7.

JACOBS UNIVERSITY

2D convolution is very similar to 1D.

- The main difference is that we now have to iterate over 2 axis instead of 1.

$$f[n,m] * h[n,m] = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} f[k,l]\, h[n-k, m-l]$$



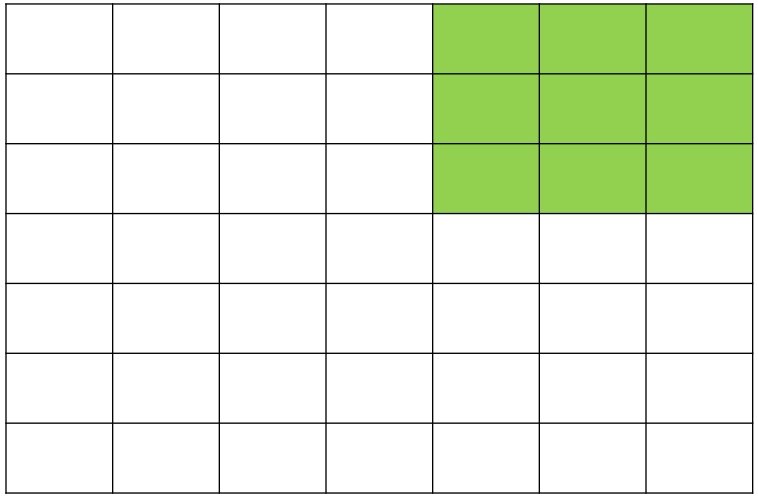
Assume we have a filter(h[,]) that is 3x3. and an image (f[,]) that is 7x7.

JACOBS UNIVERSITY

2D convolution is very similar to 1D.

- The main difference is that we now have to iterate over 2 axis instead of 1.

$$f[n,m] * h[n,m] = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} f[k,l]\, h[n-k, m-l]$$



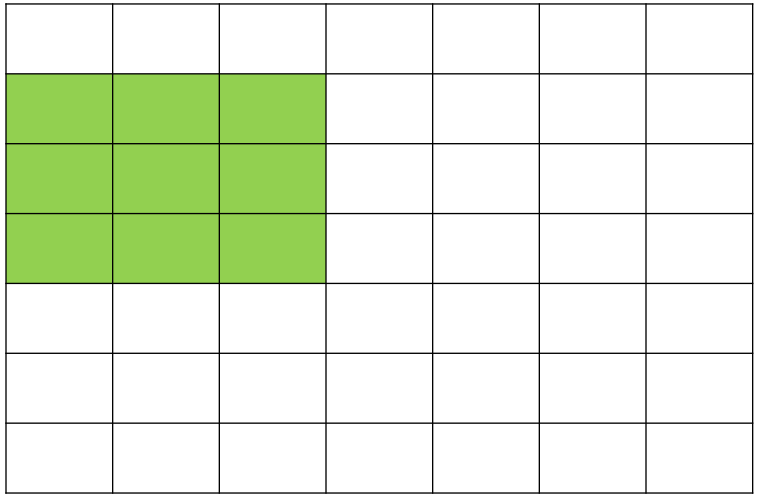
Assume we have a filter(h[,]) that is 3x3. and an image (f[,]) that is 7x7.

2D CONVOLUTION

2D convolution is very similar to 1D.

- The main difference is that we now have to iterate over 2 axis instead of 1.

$$f[n,m] * h[n,m] = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} f[k,l]\, h[n-k, m-l]$$



Assume we have a filter(h[,]) that is 3x3. and an image (f[,]) that is 7x7.

JACOBS UNIVERSITY

An LSI system is completely specified by its impulse response.

shifting property of the delta function

$$f[n,m] = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} f[k,l]\, \delta_2[n-k, m-l]$$

superposition

$$\rightarrow \boxed{\mathcal{S}\ \text{LSI}} \rightarrow \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} f[k,l]\, h[n-k, m-l]$$

$$\delta_2[n,m] \rightarrow \boxed{\mathcal{S}} \rightarrow h[n,m]$$

Discrete convolution

$$f[n,m] * h[n,m]$$

CONVOLUTION IN 2D - EXAMPLES

Original

$$*\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} = ?$$

CONVOLUTION IN 2D - EXAMPLES

Original

$$*$$

•0	•0	•0
•0	•1	•0
•0	•0	•0

$$=$$

Filtered
(no change)

CONVOLUTION IN 2D - EXAMPLES

Original

$$*\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} = ?$$

CONVOLUTION IN 2D - EXAMPLES

Original

$$*\quad \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \quad =$$

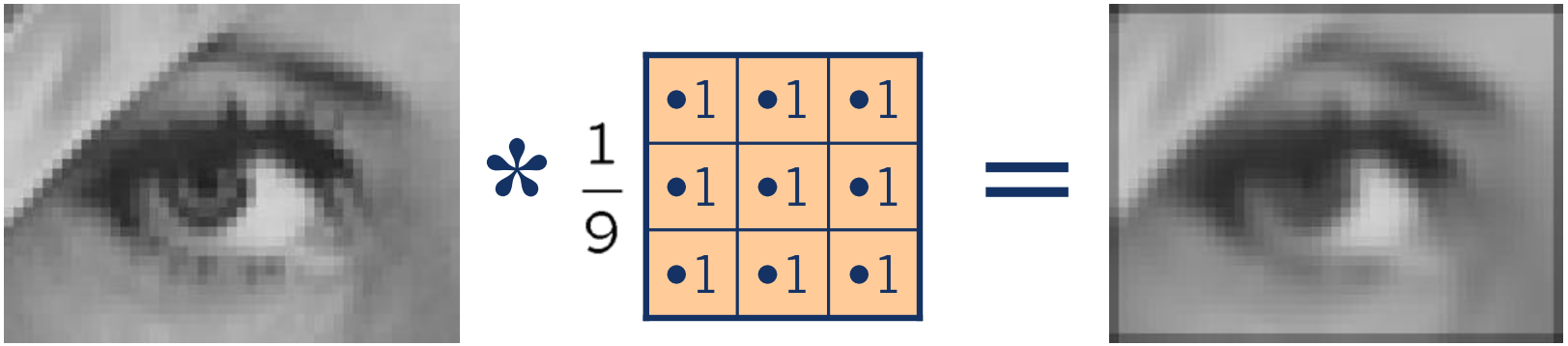
Shifted right
By 1 pixel

CONVOLUTION IN 2D - EXAMPLES

Original

$$* \frac{1}{9} \begin{bmatrix} \bullet 1 & \bullet 1 & \bullet 1 \\ \bullet 1 & \bullet 1 & \bullet 1 \\ \bullet 1 & \bullet 1 & \bullet 1 \end{bmatrix} = ?$$

CONVOLUTION IN 2D - EXAMPLES



Original

Blur (with a box filter)

CONVOLUTION IN 2D - EXAMPLES

Original

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} = ?$$

(Note that filter sums to 1)

"details of the image"

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

WHAT DOES BLURRING TAKE AWAY?

–

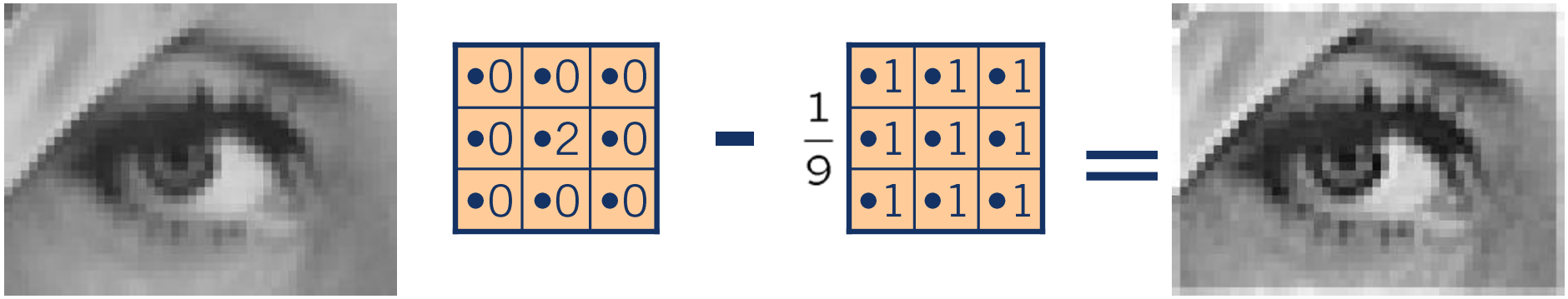
=

- Let's add it back:

+ a

=

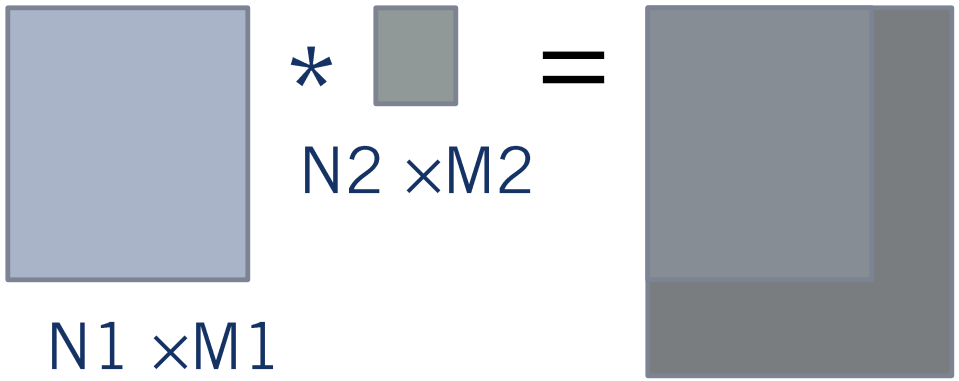
CONVOLUTION IN 2D – SHARPENING FILTER



Original

Sharpening filter: Accentuates differences with local average

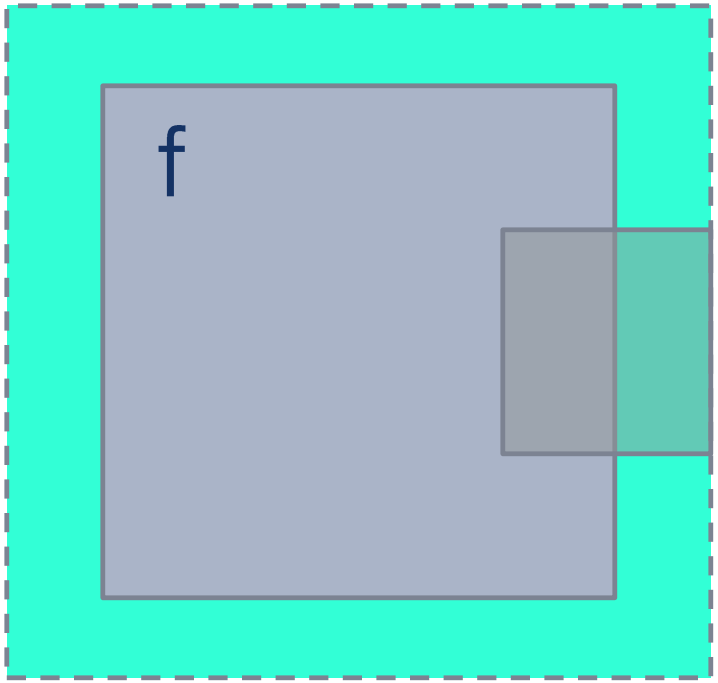
- A computer will only convolve
  finite support signals.
  - That is: images that are zero for n,m outside some rectangular region
  - MATLAB's conv2 performs 2D DS convolution of finite-support signals.



JACOBS UNIVERSITY

•A computer will only convolve **finite support signals.**

• What happens at the edge?



h

- zero “padding”
- edge value replication
- mirror extension
- more (beyond the scope of this class)

-> Matlab conv2 uses zero-padding

WHAT WE WILL LEARN TODAY?

JACOBS UNIVERSITY

1. Image sampling and quantization
2. Image histograms
3. Images as functions
4. Linear systems (filters)
5. Convolution and correlation

Some background reading:
Forsyth and Ponce, Computer Vision, Chapter 7

(CROSS) CORRELATION (SYMBOL: **)

Cross correlation of two 2D signals f[n,m] and g[n,m]

$$r_{fg}[k,l] \triangleq \sum_{n=-\infty}^{\infty} \sum_{m=-\infty}^{\infty} f[n,m]\, g^*[n-k, m-l]$$

$$= \sum_{n=-\infty}^{\infty} \sum_{m=-\infty}^{\infty} f[n+k, m+l]\, g^*[n,m], \quad k,l \in \mathbb{Z}.$$

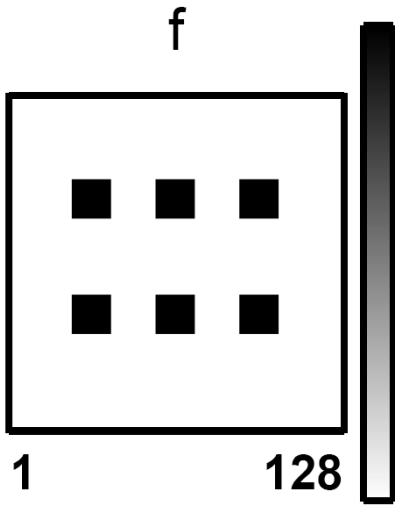
(k, l) is called the **lag**

- Equivalent to a convolution without the flip

$$r_{fg}[n,m] = f[n,m] * g^*[-n,-m]$$

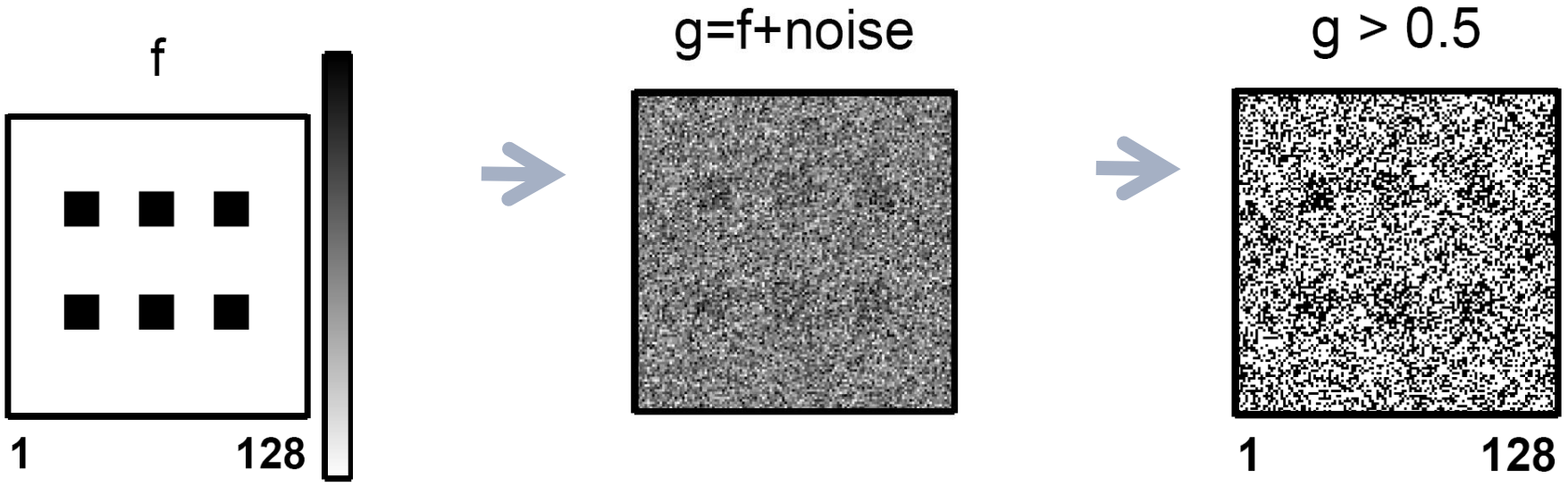
(g* is defined as the *complex conjugate* of g. In this class, g(n,m) are real numbers, hence g*=g.)

(CROSS) CORRELATION – EXAMPLE



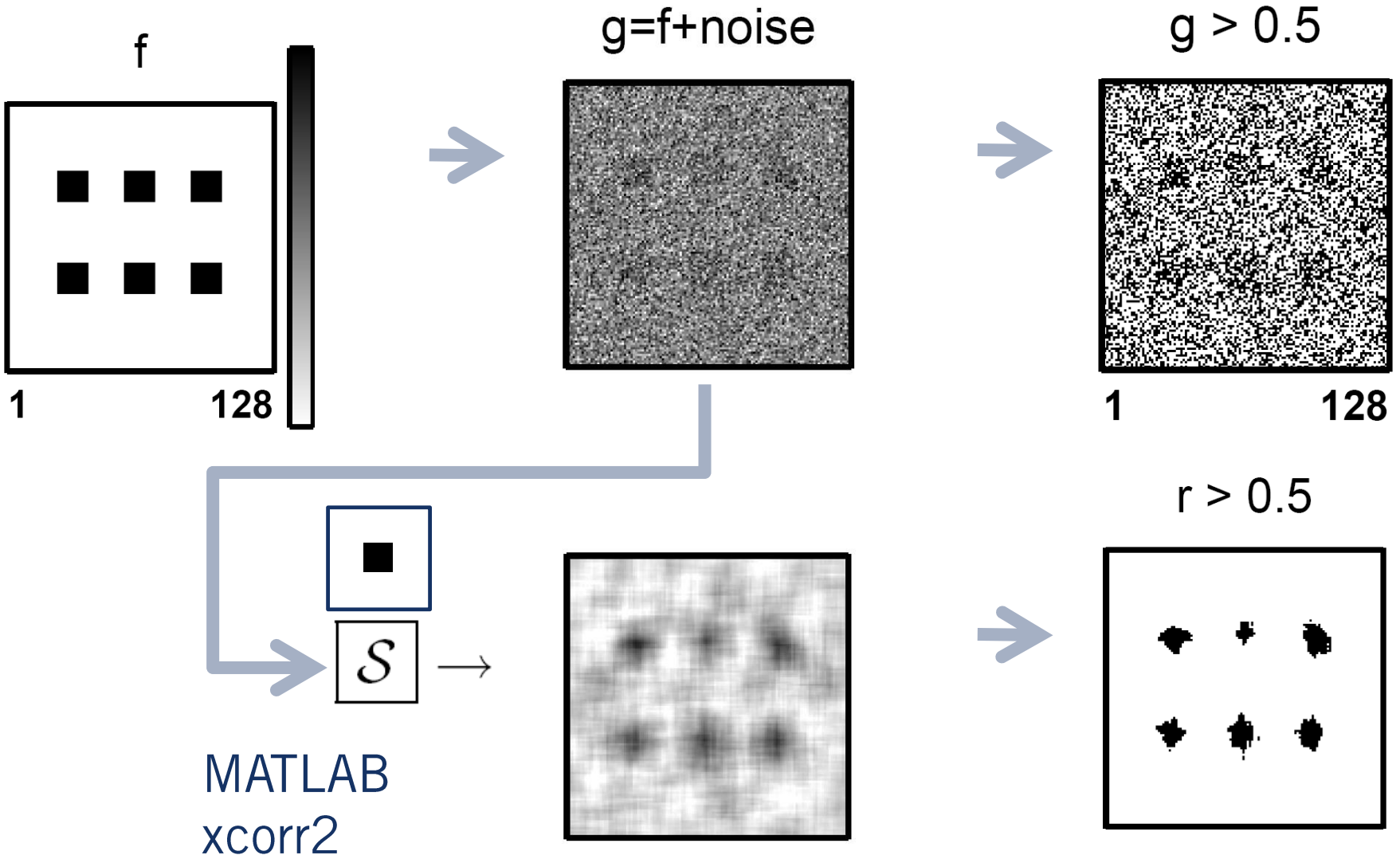
Courtesy of J. Fessler

(CROSS) CORRELATION – EXAMPLE



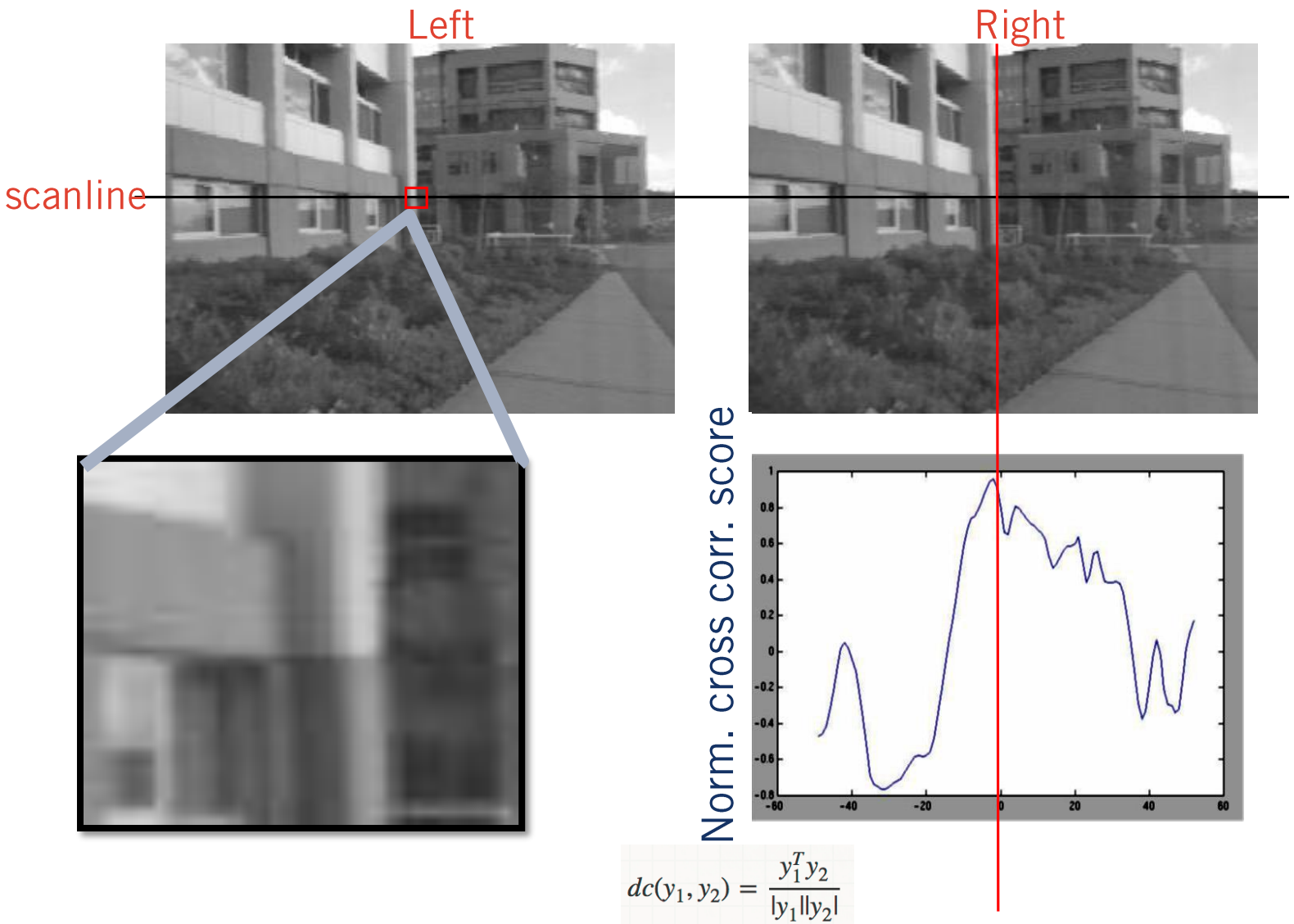
Courtesy of J. Fessler

(CROSS) CORRELATION – EXAMPLE



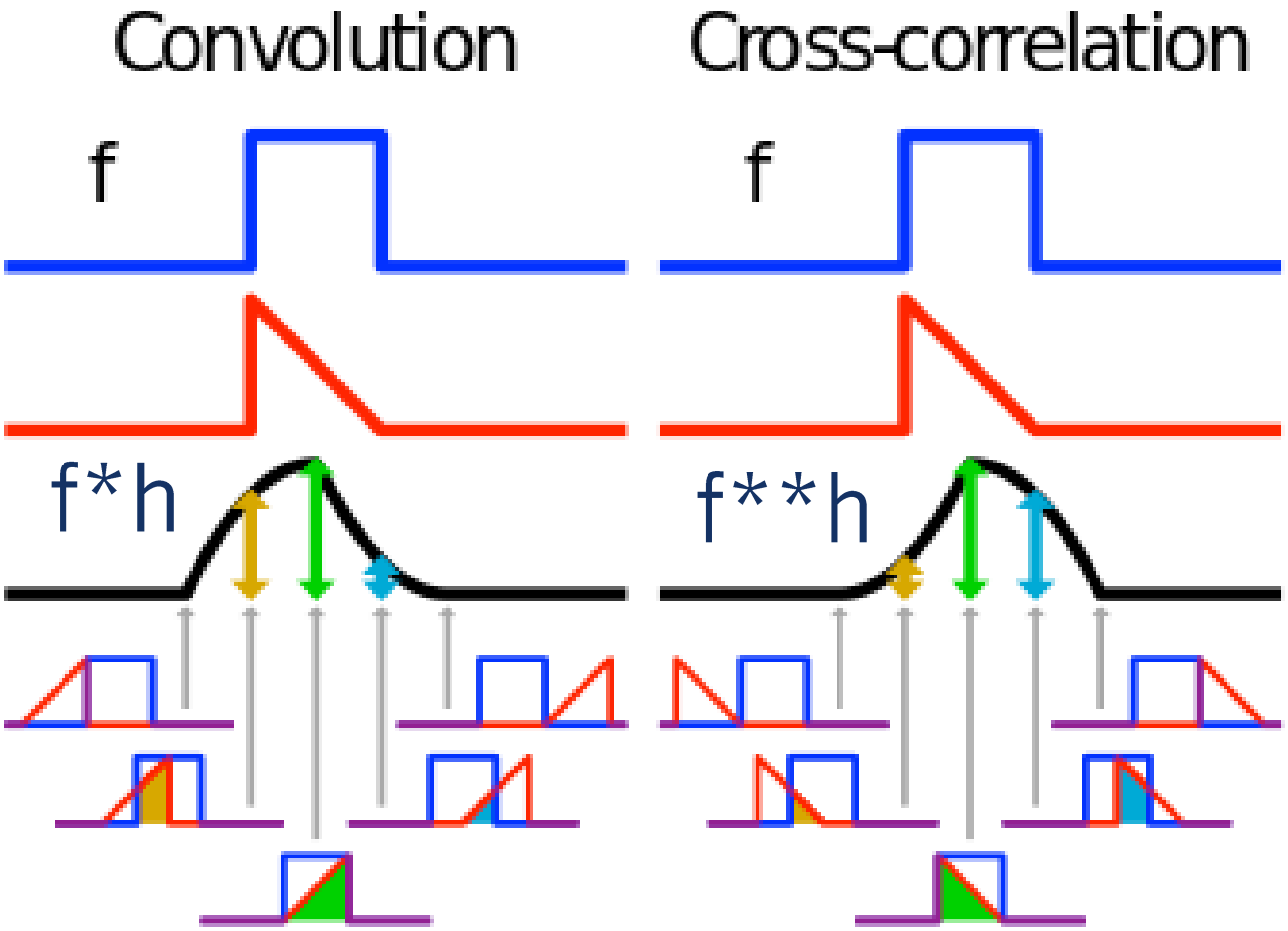
Courtesy of J. Fessler

(CROSS) CORRELATION – EXAMPLE



$$dc(y_1, y_2) = \frac{y_1^T y_2}{|y_1||y_2|}$$

CONVOLUTION VS. (CROSS) CORRELATION



Cross Correlation Application: Vision system for TV remote control

- uses template matching

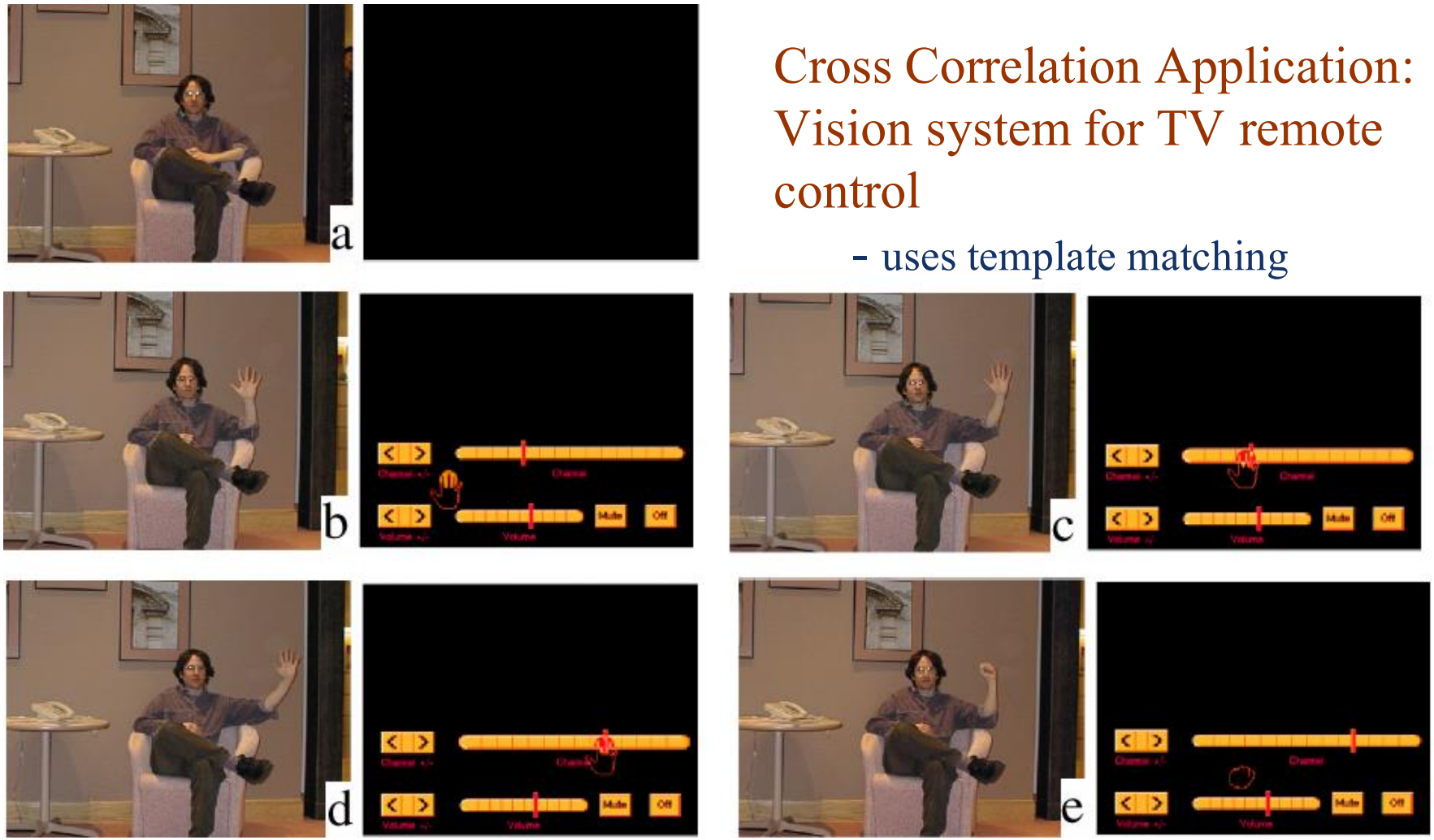


Figure from "Computer Vision for Interactive Computer Graphics," W.Freeman et al, IEEE Computer Graphics and Applications, 1998 copyright 1998, IEEE

JACOBS UNIVERSITY

- **Associative property:**

$$(f ** h_1) ** h_2 = f ** (h_1 ** h_2)$$

- **Distributive property:**

$$f ** (h_1 + h_2) = (f ** h_1) + (f ** h_2)$$

The order doesn't matter! $h_1 ** h_2 = h_2 ** h_1$

JACOBS UNIVERSITY

- **Shift property:**

$$f[n,m] ** \delta_2[n-n_0, m-m_0] = f[n-n_0, m-m_0]$$

- **Shift-invariance:**

$$\begin{aligned} g[n,m] &= f[n,m] ** h[n,m] \\ &\Longrightarrow f[n-l_1, m-l_1] ** h[n-l_2, m-l_2] \\ &= g[n-l_1-l_2, m-l_1-l_2] \end{aligned}$$

CONVOLUTION VS. (CROSS) CORRELATION

JACOBS UNIVERSITY

1. A **convolution** is an integral that expresses the amount of overlap of one function as it is shifted over another function.

   → convolution is a **filtering** operation

1. **Correlation** compares the ***similarity*** *of* ***two*** *sets of* ***data.*** Correlation computes a measure of similarity of two input signals as they are shifted by one another. The correlation result reaches a maximum at the time when the two signals match best.

   → correlation is a measure of **relatedness** of two signals

WHAT WE HAVE LEARNED TODAY?

JACOBS UNIVERSITY

1. Image sampling and quantization
2. Image histograms
3. Images as functions
4. Linear systems (filters)
5. Convolution and correlation

ANY QUESTIONS?