Lecture: Pixels and Filters

Juan Carlos Niebles and Ranjay Krishna

Stanford Vision Lab

What we will learn today?

- Color
- Image sampling and quantization
- Image histograms
- Images as functions
- Linear systems (filters)
- Convolution and correlation

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

What we will learn today?

- Color
- Image sampling and quantization
- Image histograms
- Images as functions
- Linear systems (filters)
- Convolution and correlation

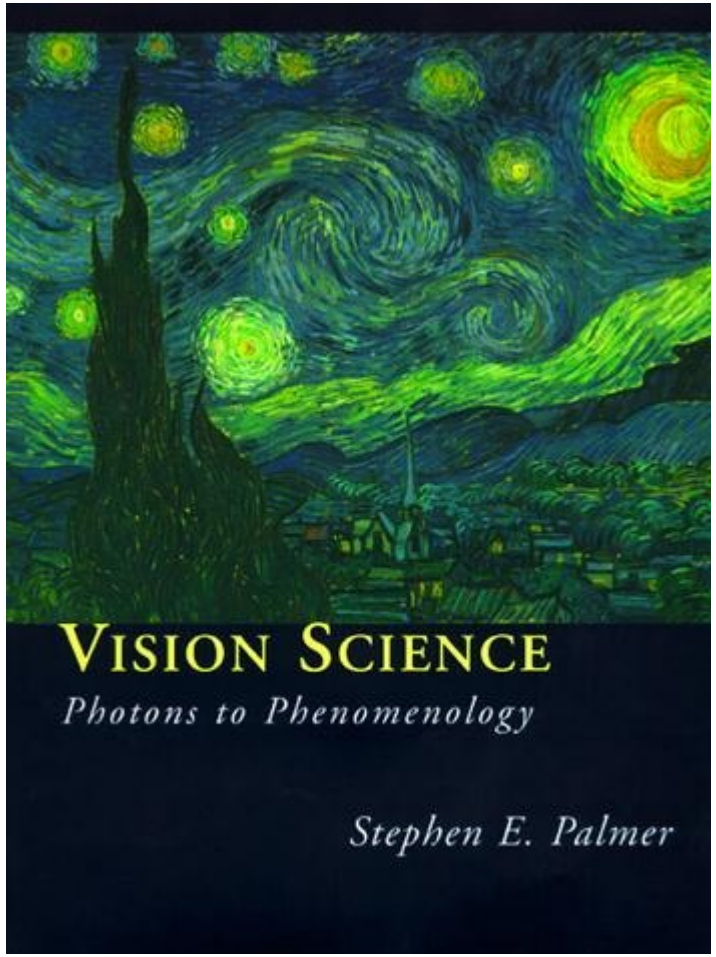
Overview of Color

- What is color?
- Color spaces
- White balancing

What is color?

- The result of interaction between physical light in the environment and our visual system.

- A psychological property of our visual experiences when we look at objects and lights, not a physical property of those objects or lights.



Slide credit: Lana Lazebnik

Interaction of light and surfaces

What is the observed color of any surface under monochromatic light?

Olafur Eliasson, Room for one color

James Turrell, Breathing Light

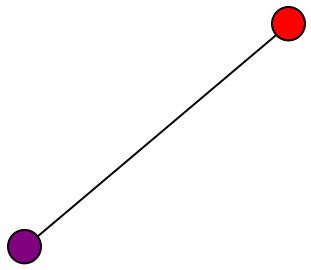
Inspired Drake in “Hotline Bling”

Overview of Color

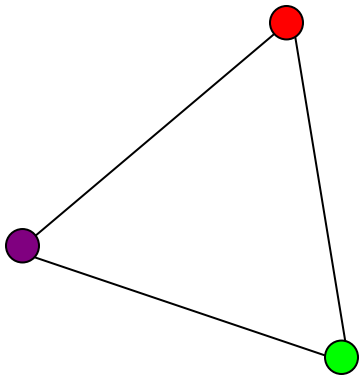
- What is color?
- Color spaces
- Use cases

Linear color spaces

- Defined by a choice of three *primaries*
- The coordinates of a color are given by the weights of the primaries used to match it



mixing two lights produces colors that lie along a straight line in color space

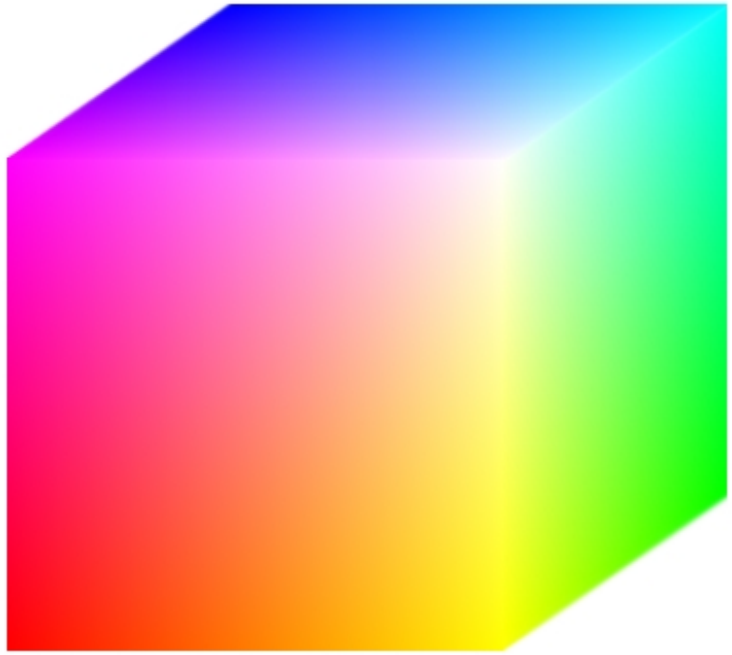


mixing three lights produces colors that lie within the triangle they define in color space

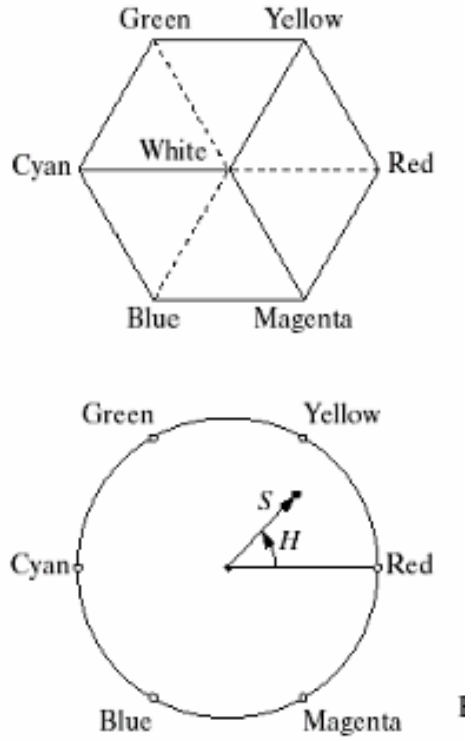
RGB space

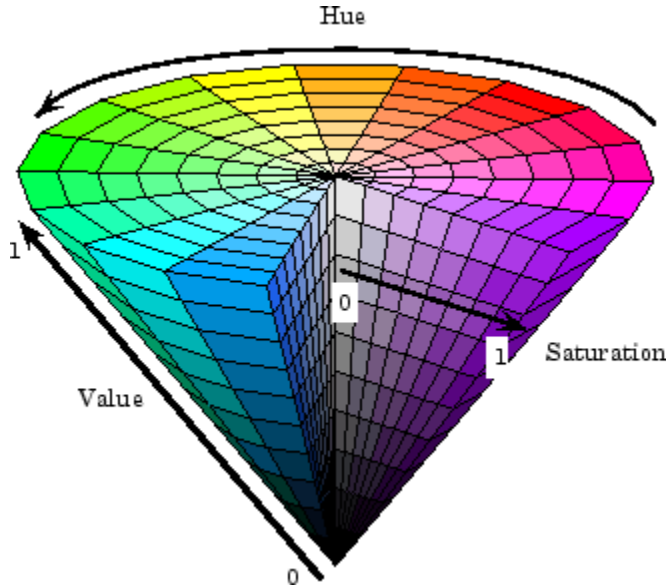
- Primaries are monochromatic lights (for monitors, they correspond to the three types of phosphors)

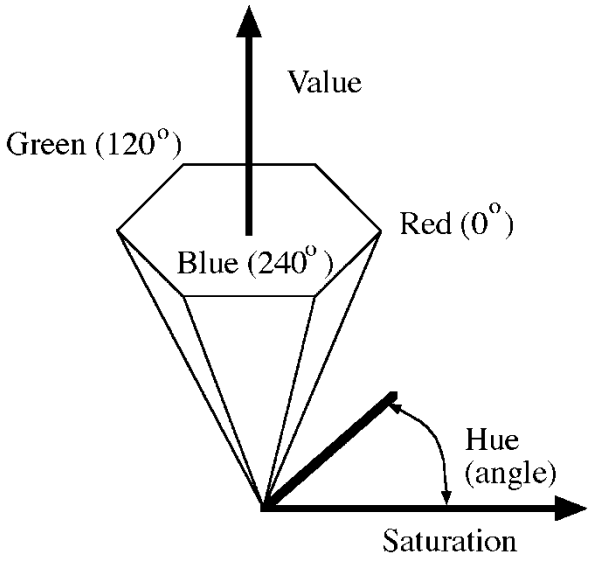
RGB primaries



Nonlinear color spaces: HSV







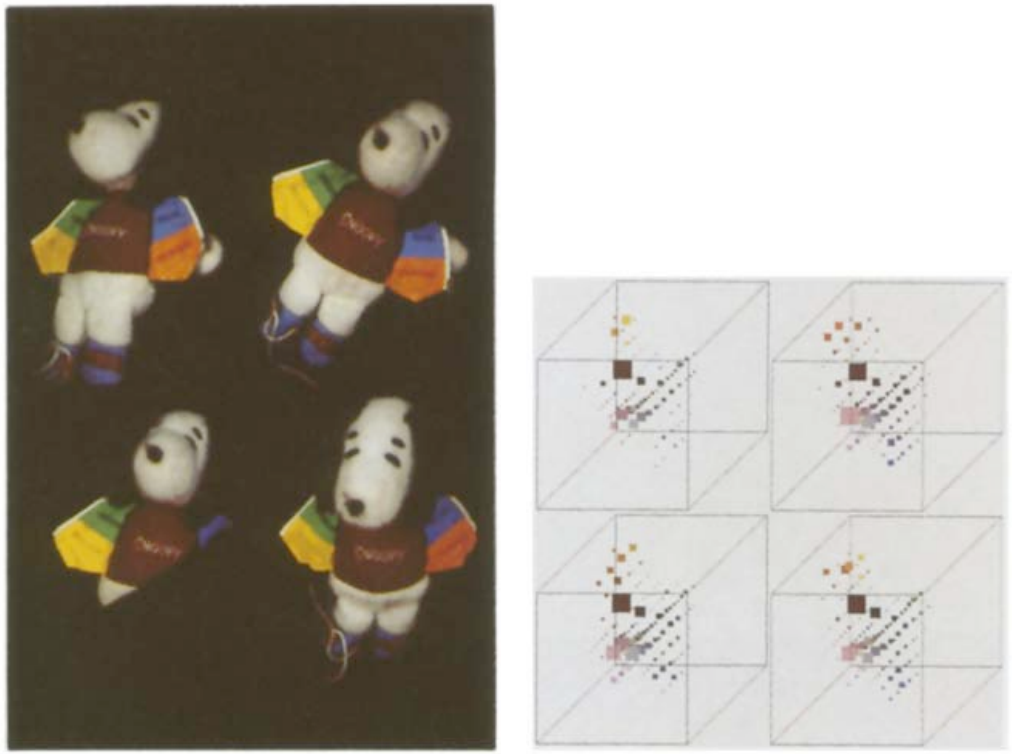
- Perceptually meaningful dimensions: Hue, Saturation, Value (Intensity)
- RGB cube on its vertex

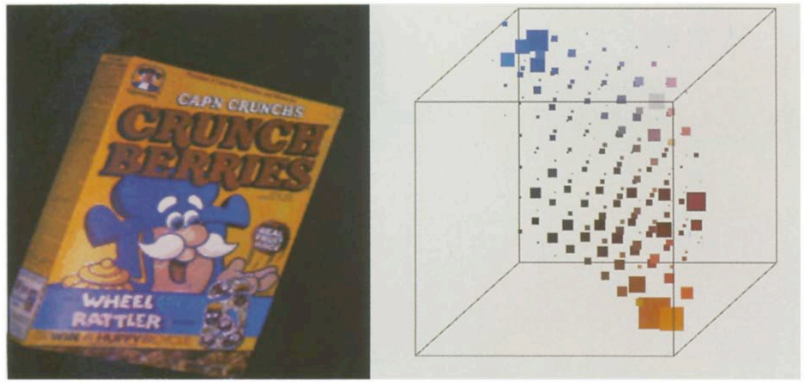
Overview of Color

- What is color?
- Color spaces
- Use cases

Uses of color in computer vision

Color histograms for indexing and retrieval

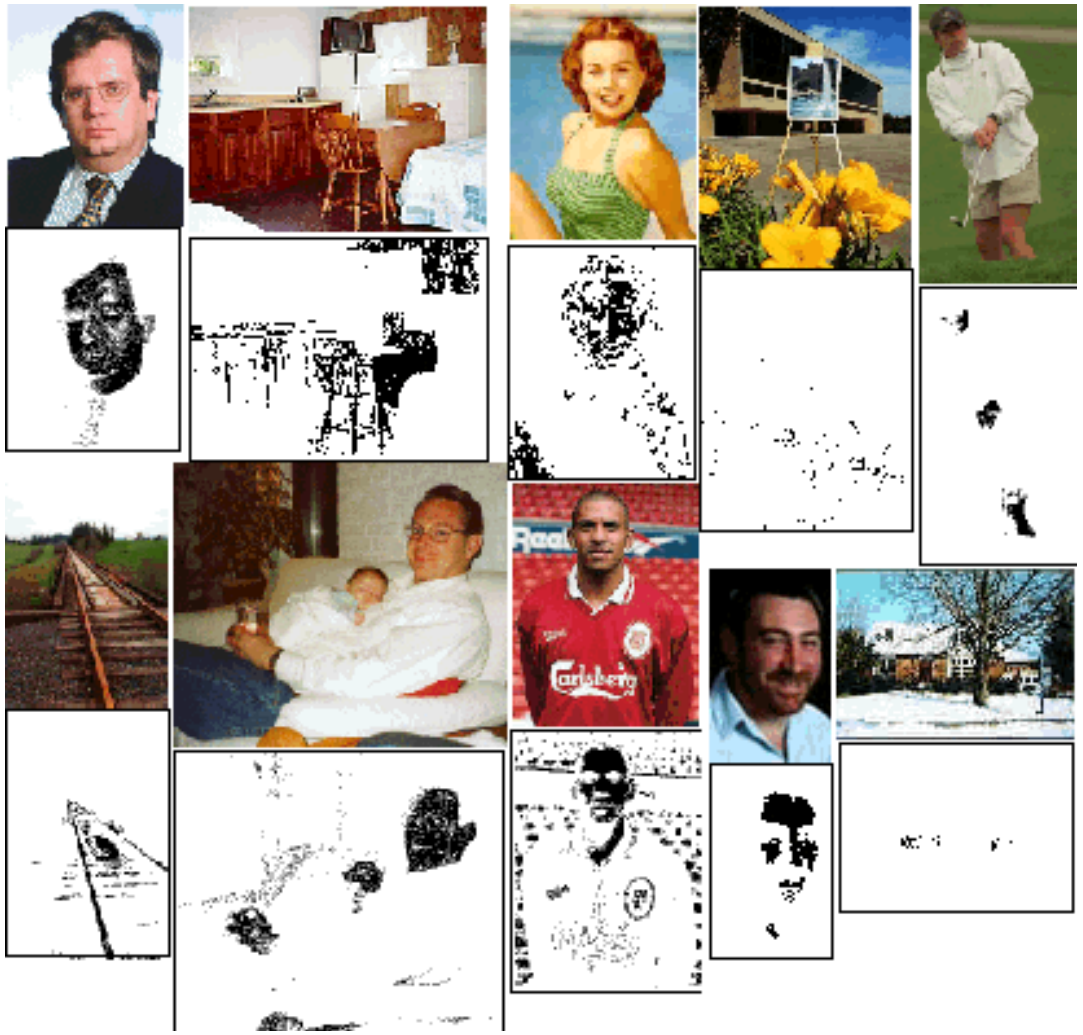




Swain and Ballard, Color Indexing, IJCV 1991.

Uses of color in computer vision

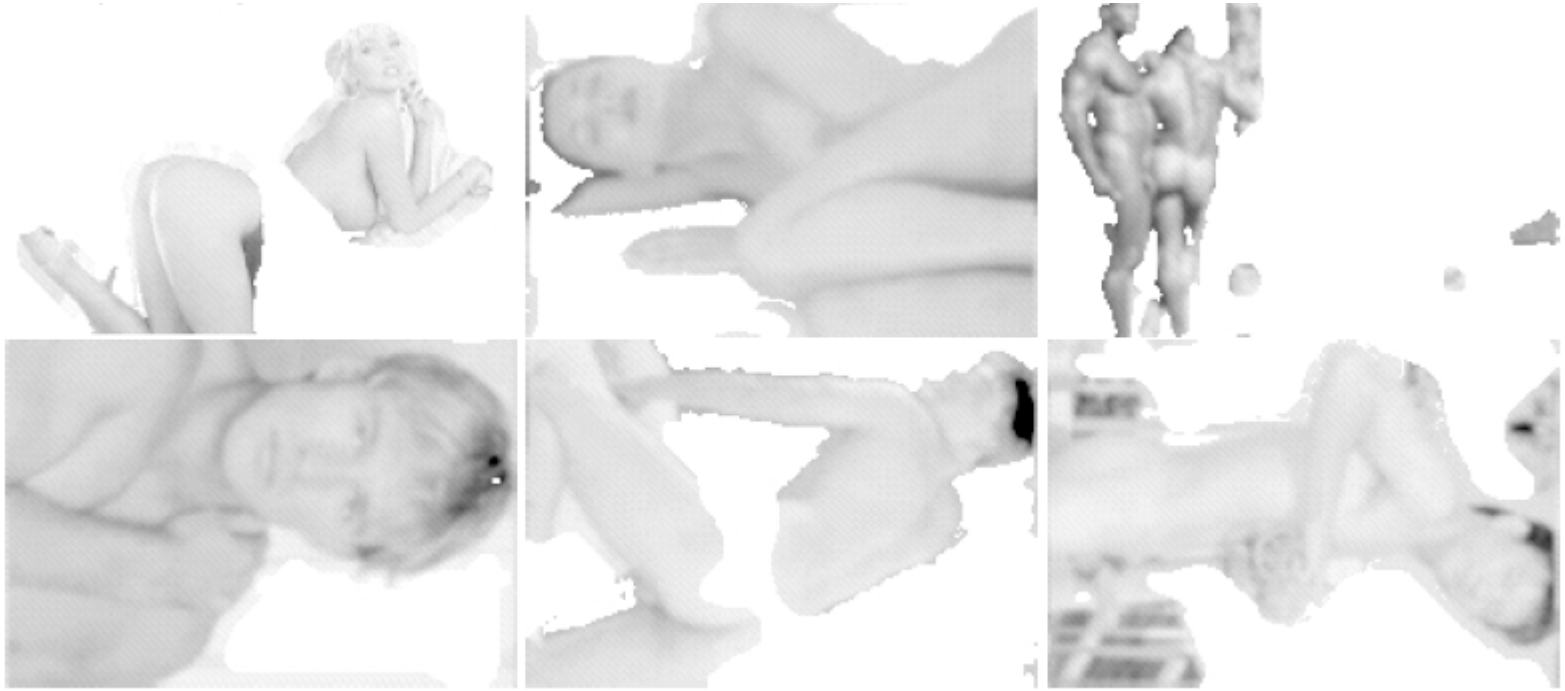
Skin detection



M. Jones and J. Rehg, Statistical Color Models with Application to Skin Detection, IJCV 2002.

Uses of color in computer vision

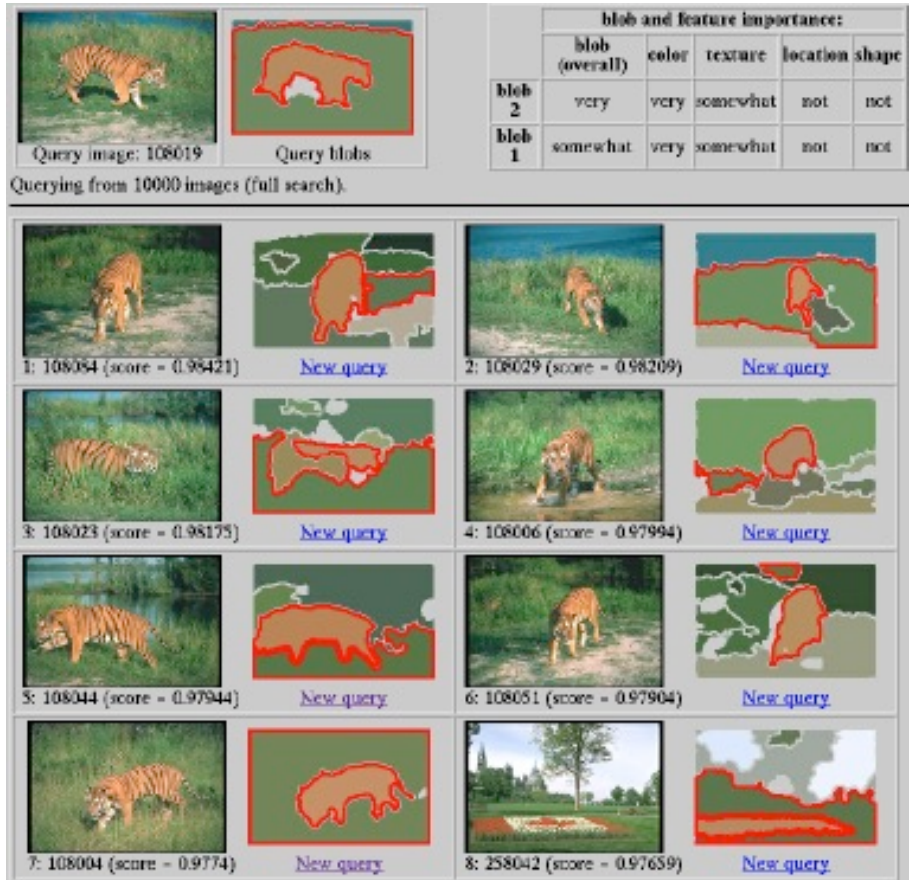
Nude people detection



Forsyth, D.A. and Fleck, M. M., ``Automatic Detection of Human Nudes," International Journal of Computer Vision , 32 , 1, 63–77, August, 1999

Uses of color in computer vision

Image segmentation and retrieval

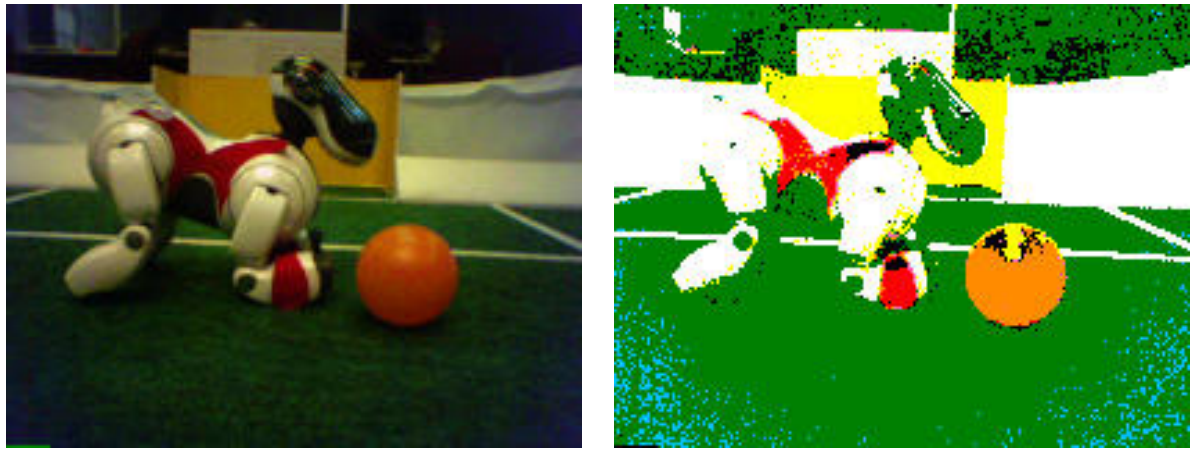


C. Carson, S. Belongie, H. Greenspan, and Ji. Malik, Blobworld: Image segmentation using Expectation−Maximization and its application to image querying, ICVIS 1999.

Source: S. Lazebnik

Uses of color in computer vision

Robot soccer

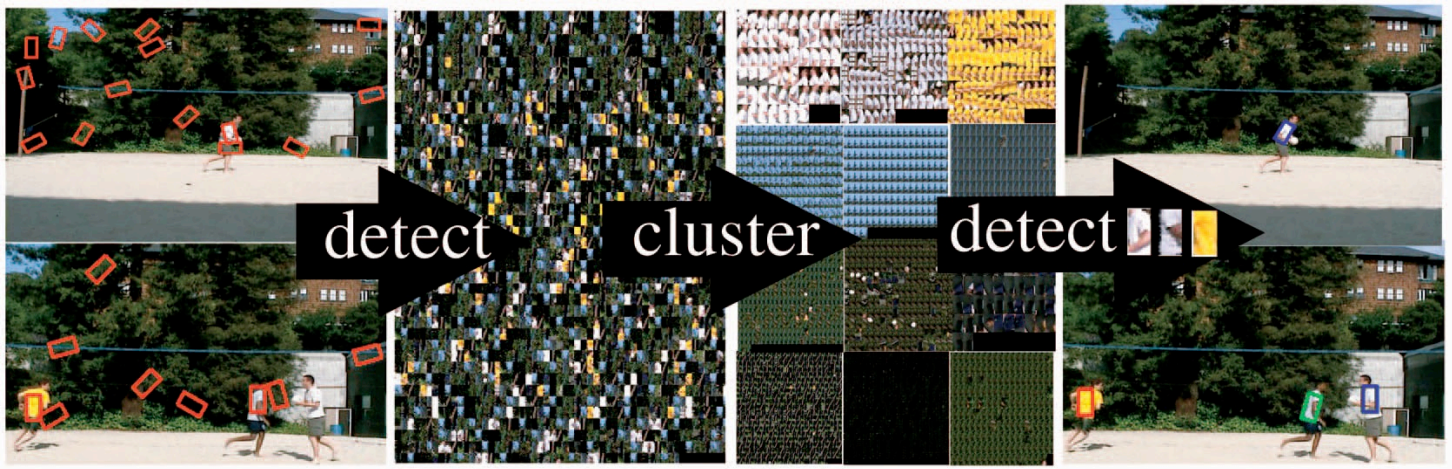


M. Sridharan and P. Stone, Towards Eliminating Manual Color Calibration at RoboCup. RoboCup-2005: Robot Soccer World Cup IX, Springer Verlag, 2006

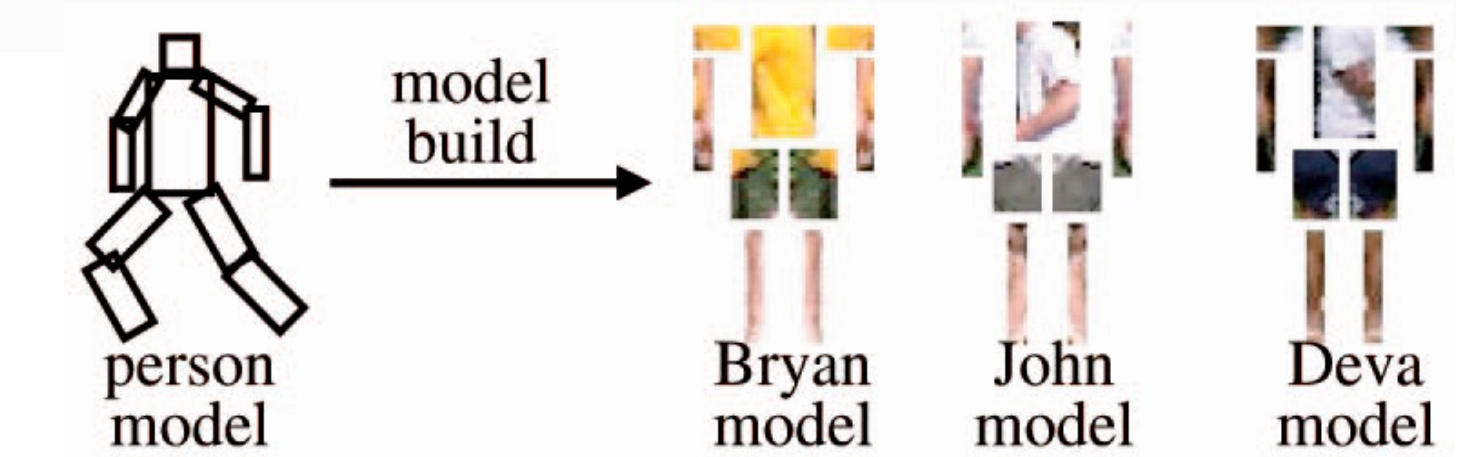
Source: K. Grauman

Uses of color in computer vision

Building appearance models for tracking



D. Ramanan, D. Forsyth, and A. Zisserman. Tracking People by Learning their Appearance. PAMI 2007.



Source: S. Lazebnik

What we will learn today?

- Color
- Image sampling and quantization
- Image histograms
- Images as functions
- Linear systems (filters)
- Convolution and correlation

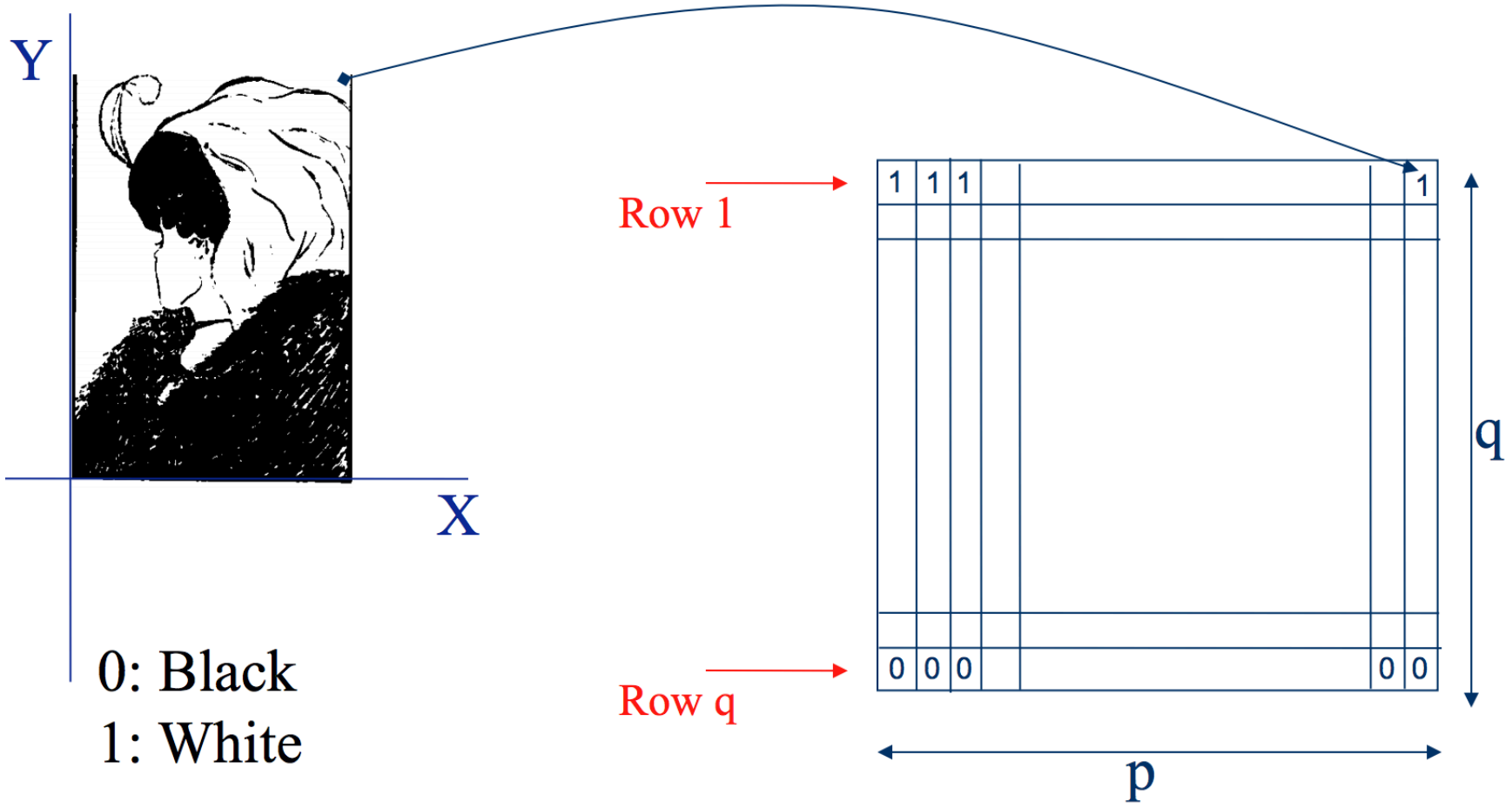
Types of Images

Binary

Gray Scale

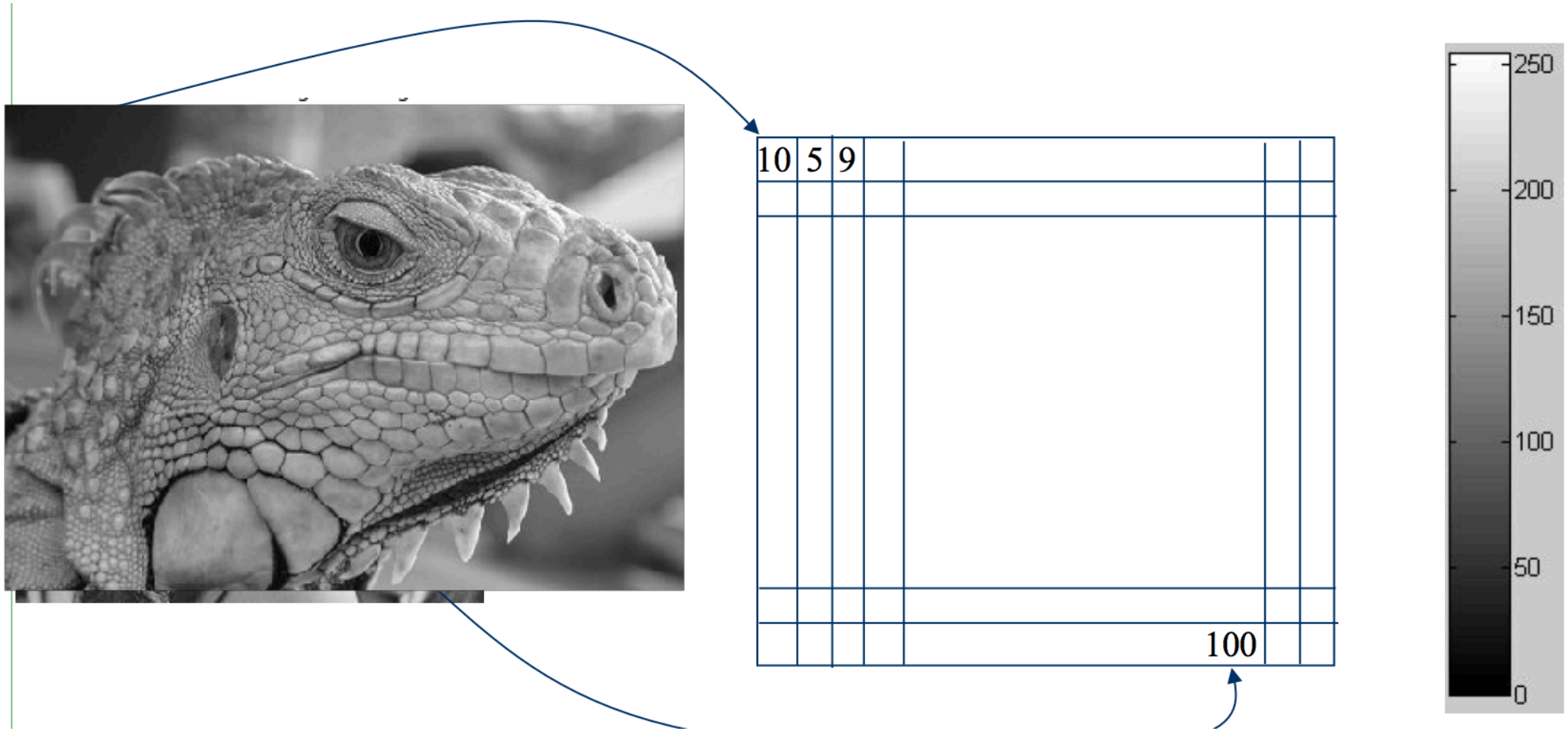
Color

Binary image representation



Slide credit: Ulas Bagci

Grayscale image representation



Slide credit: Ulas Bagci

Color Image - one channel

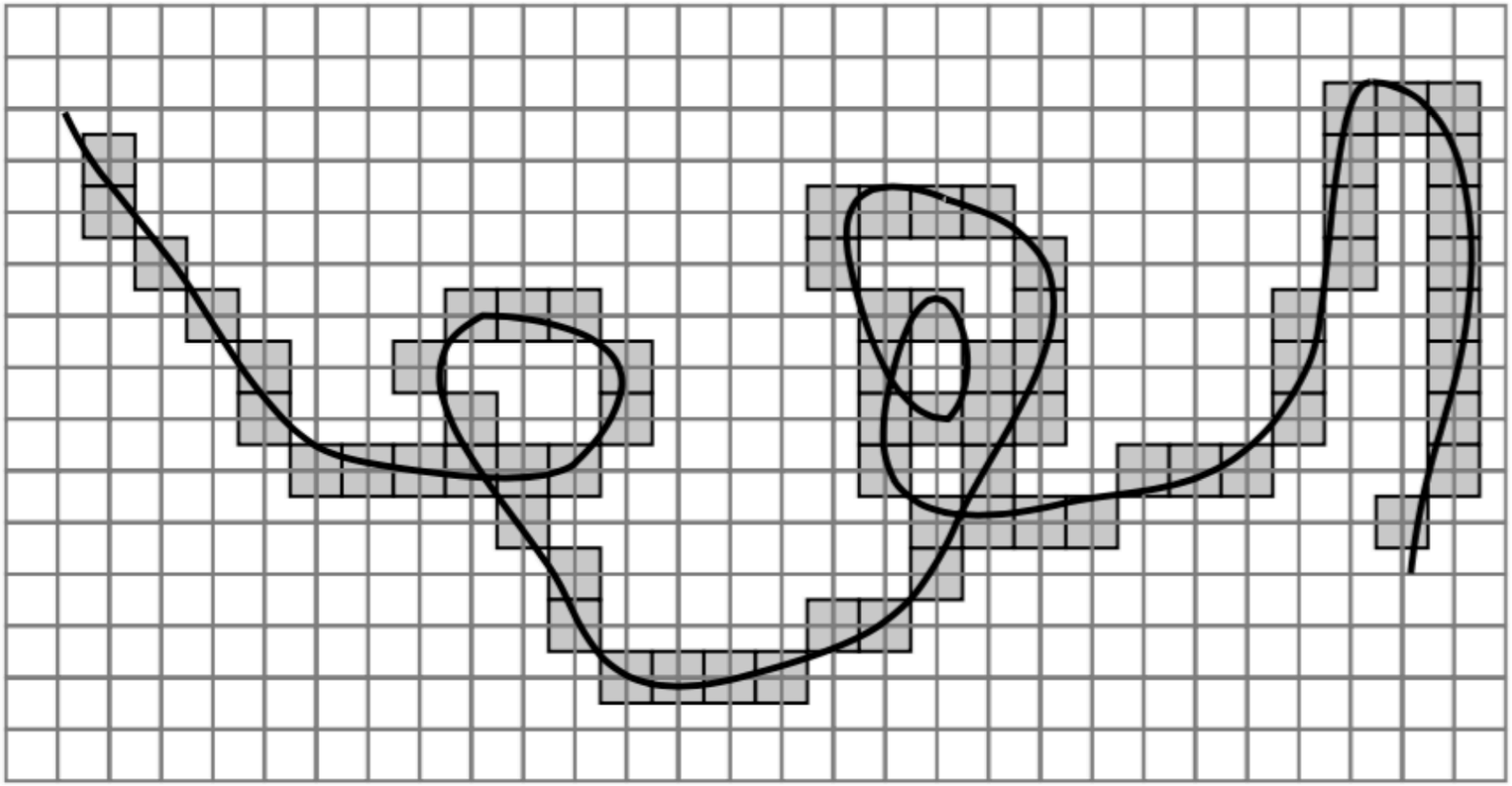
Color image representation

Slide credit: Ulas Bagci

Images are sampled

What happens when we zoom into the images we capture?

Errors due Sampling



Slide credit: Ulas Bagci

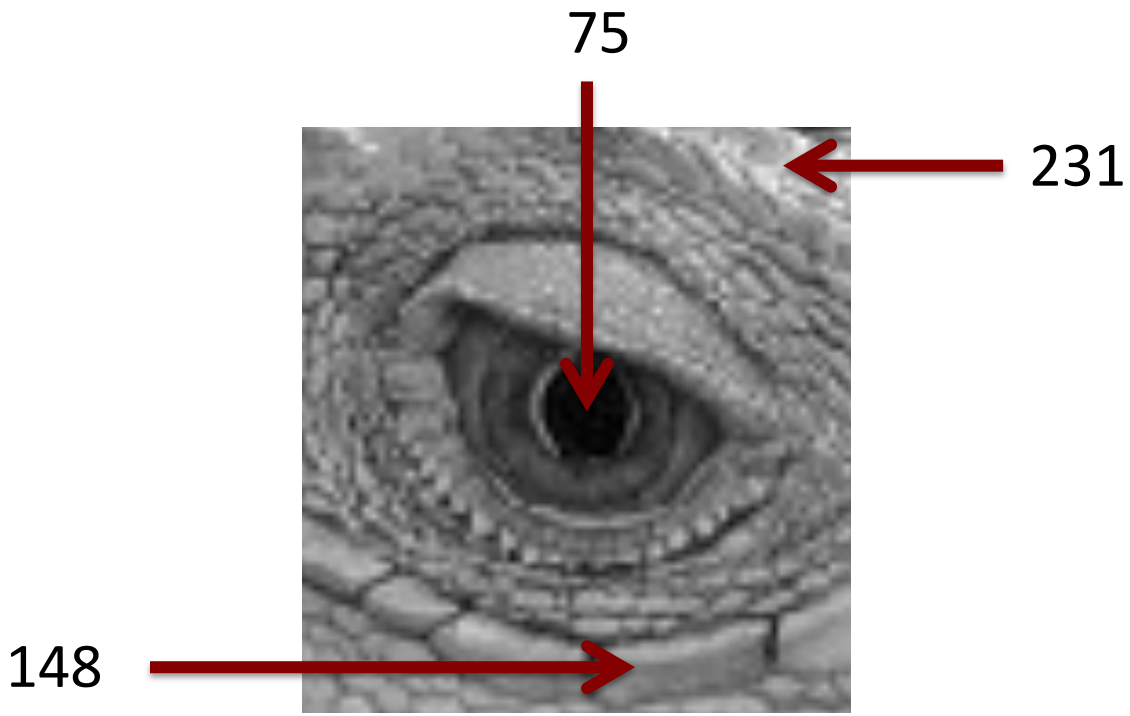
Resolution

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

Slide credit: Ulas Bagci

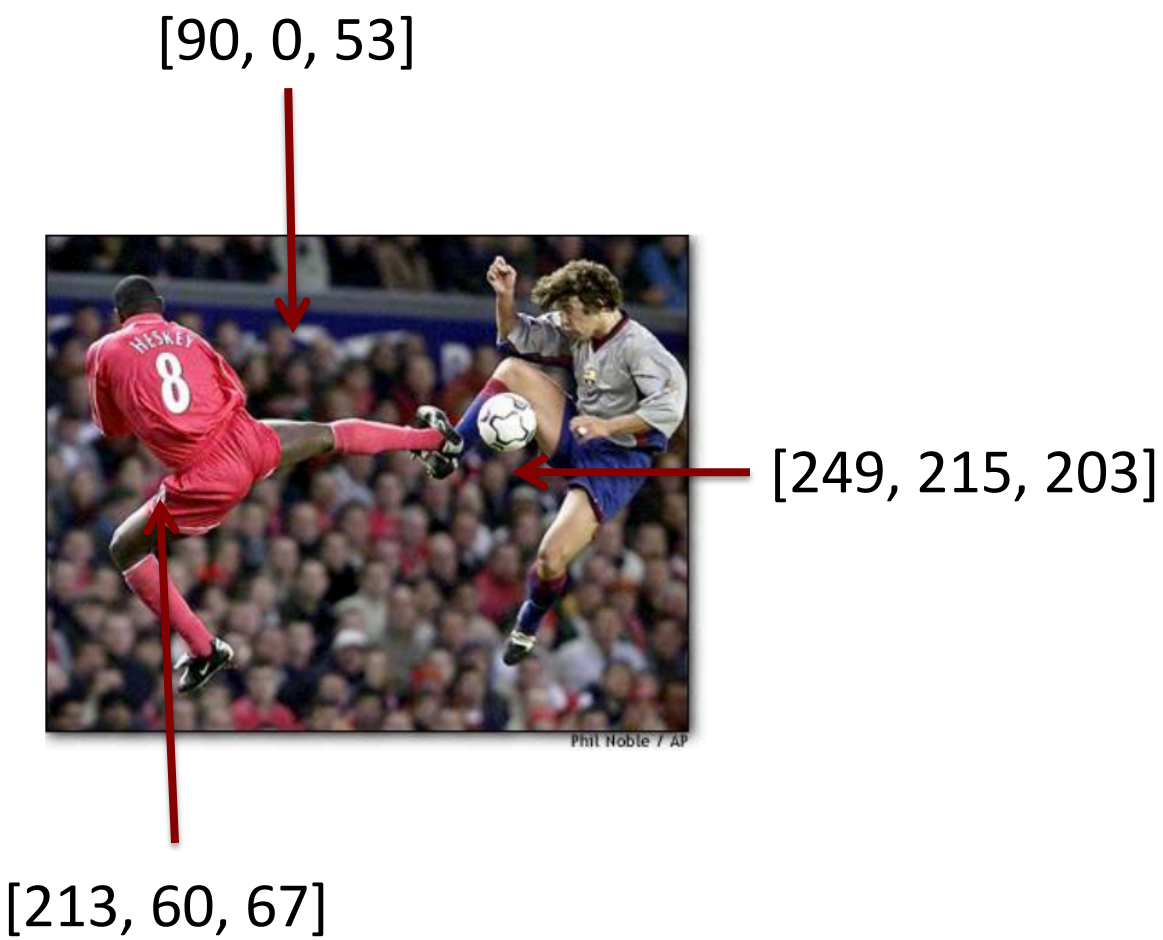
Images are Sampled and Quantized

- An image contains discrete number of pixels
  - A simple example
  - Pixel value:
    - "grayscale"
      (or "intensity"): [0,255]



Images are Sampled and Quantized

- An image contains discrete number of pixels
  - A simple example
  - Pixel value:
    - "grayscale"
      (or "intensity"): [0,255]
    - "color"
      - RGB: [R, G, B]
      - Lab: [L, a, b]
      - HSV: [H, S, V]



With this loss of information (from sampling and quantization),

Can we still use images for useful tasks?

What we will learn today?

- Color
- Image sampling and quantization
- Image histograms
- Images as functions
- Linear systems (filters)
- Convolution and correlation

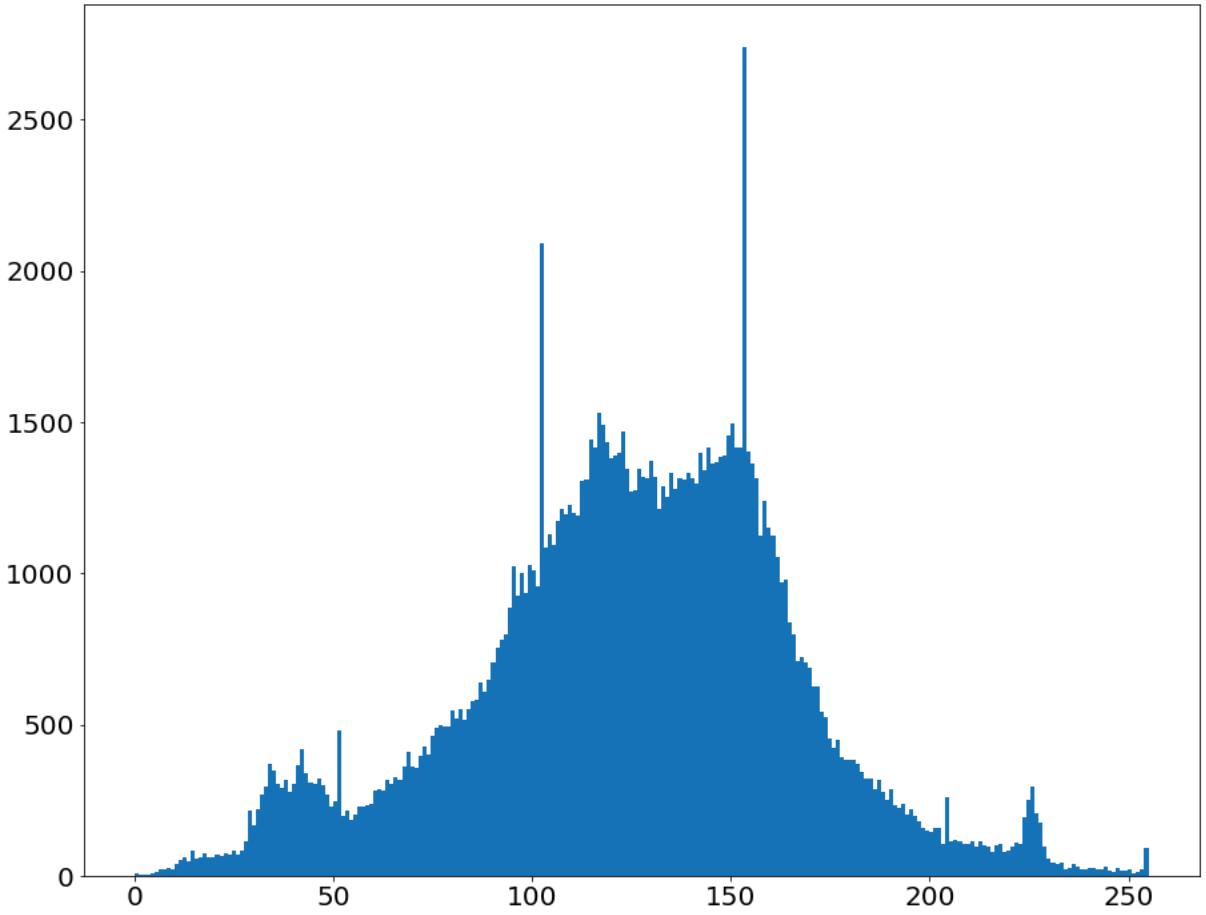
Histogram

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

```
def histogram(im):
    h = np.zeros(255)
    for row in im.shape[0]:
        for col in im.shape[1]:
            val = im[row, col]
            h[val] += 1
```

Histogram

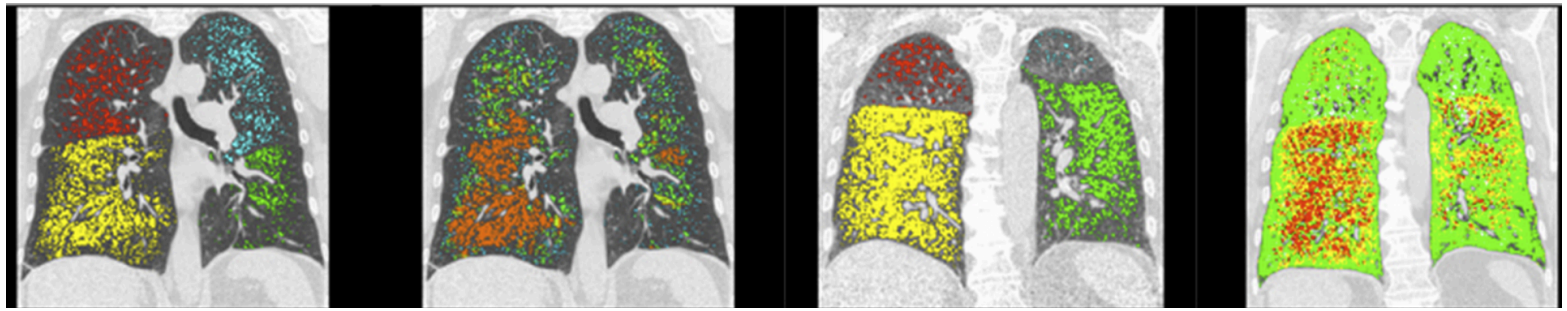
- Histogram captures the distribution of gray levels in the image.
- How frequently each gray level occurs in the image

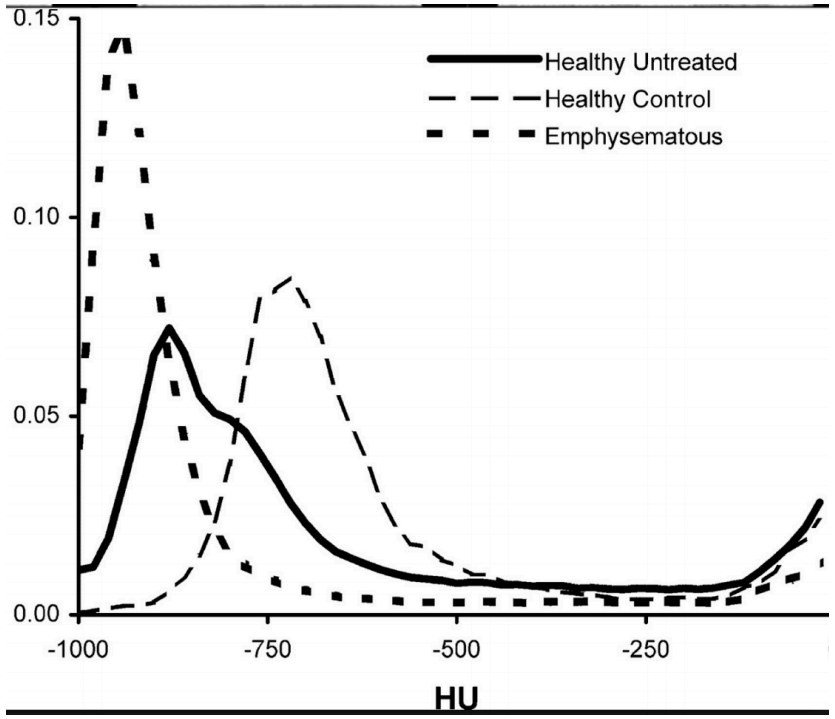


Histogram

Slide credit: Dr. Mubarak Shah

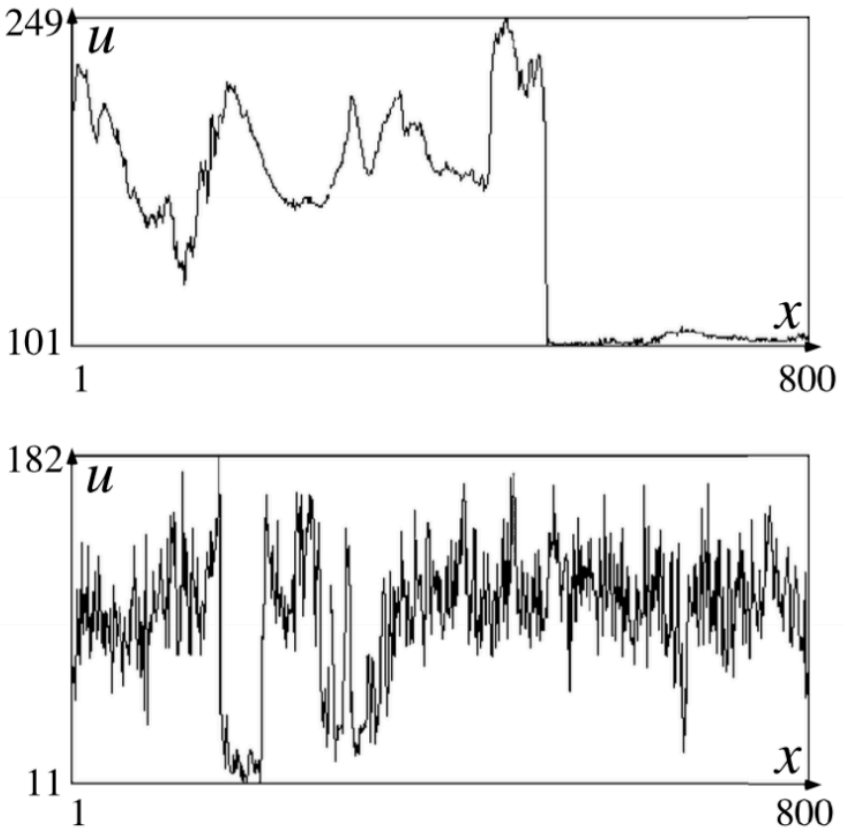
Histogram – use case





Slide credit: Dr. Mubarak Shah

Histogram – another use case



Slide credit: Dr. Mubarak Shah

What we will learn today?

- Color
- Image sampling and quantization
- Image histograms
- Images as functions
- Linear systems (filters)
- Convolution and correlation

Images as discrete functions

- Images are usually **digital** (**discrete**):
  - **Sample** the 2D space on a regular grid

- Represented as a matrix of integer values

pixel

m

n

62	79	23	119	120	105	4	0
10	10	9	62	12	78	34	0
10	58	197	46	46	0	0	48
176	135	5	188	191	68	0	49
2	1	1	29	26	37	0	77
0	89	144	147	187	102	62	208
255	252	0	166	123	62	0	31
166	63	127	17	1	0	99	30

Images as coordinates

Cartesian coordinates

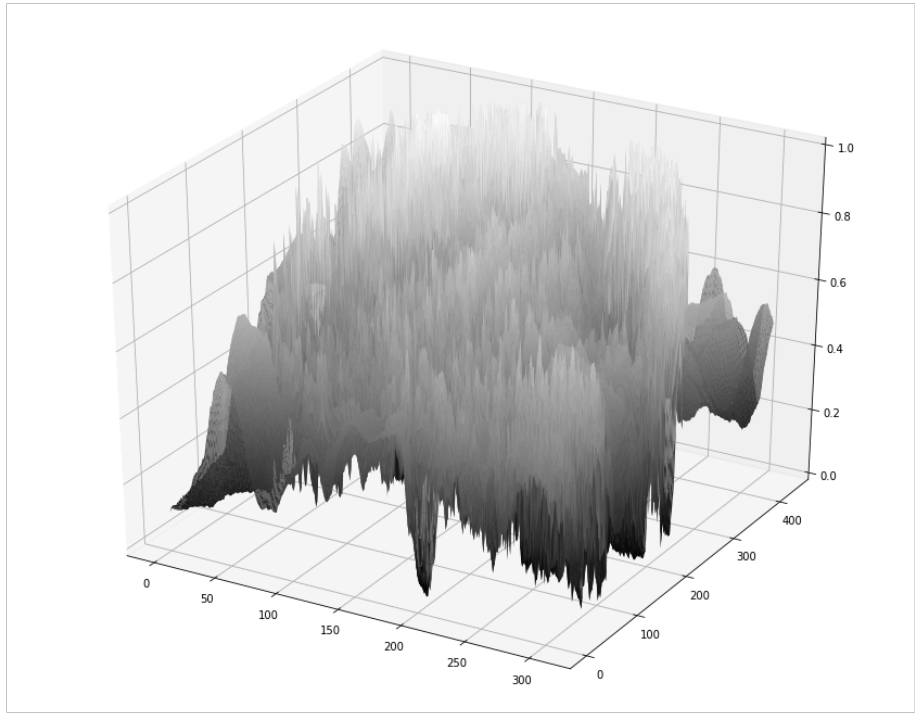
$$f[n,m] = \begin{bmatrix} \ddots & & \vdots & & \\ & f[-1,-1] & f[-1,0] & f[-1,1] & \\ \dots & f[0,-1] & \underline{f[0,0]} & f[0,1] & \dots \\ & f[1,-1] & f[1,0] & f[1,1] & \\ & & \vdots & & \ddots \end{bmatrix}$$

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:

$$f: \underbrace{[a,b] \times [c,d]}_{\text{Domain support}} \rightarrow \underbrace{[0,255]}_{\text{range}}$$



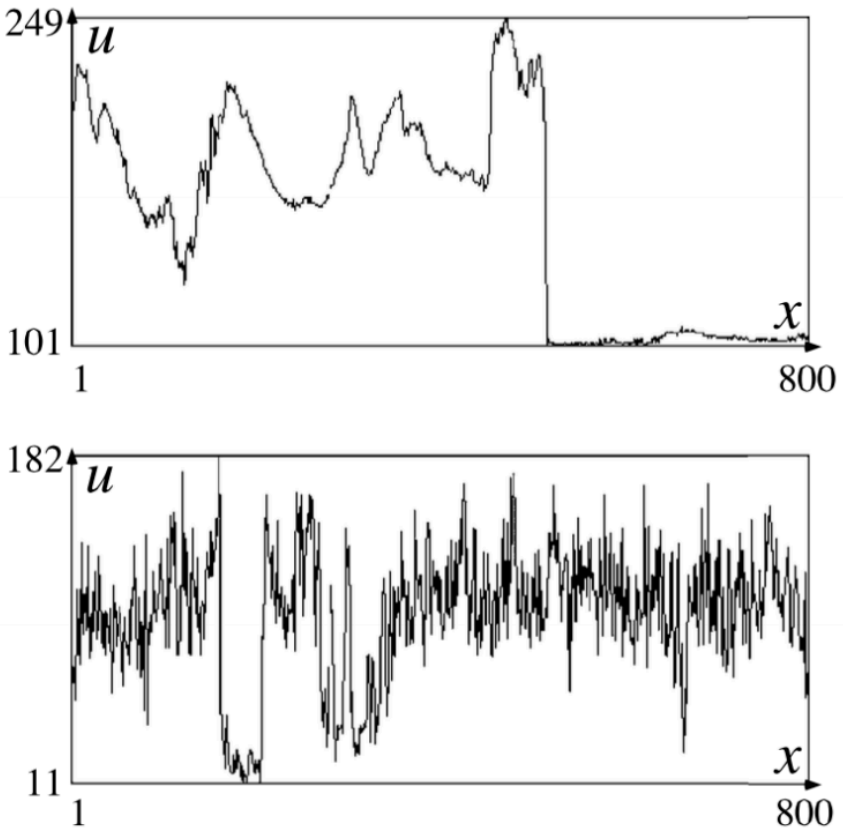
Images as functions

- **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:

$$f: \underbrace{[a,b] \times [c,d]}_{\text{Domain support}} \rightarrow \underbrace{[0,255]}_{\text{range}}$$

- A color image: $$f(x,y) = \begin{bmatrix} r(x,y) \\ g(x,y) \\ b(x,y) \end{bmatrix}$$

Histograms are a type of image function



What we will learn today?

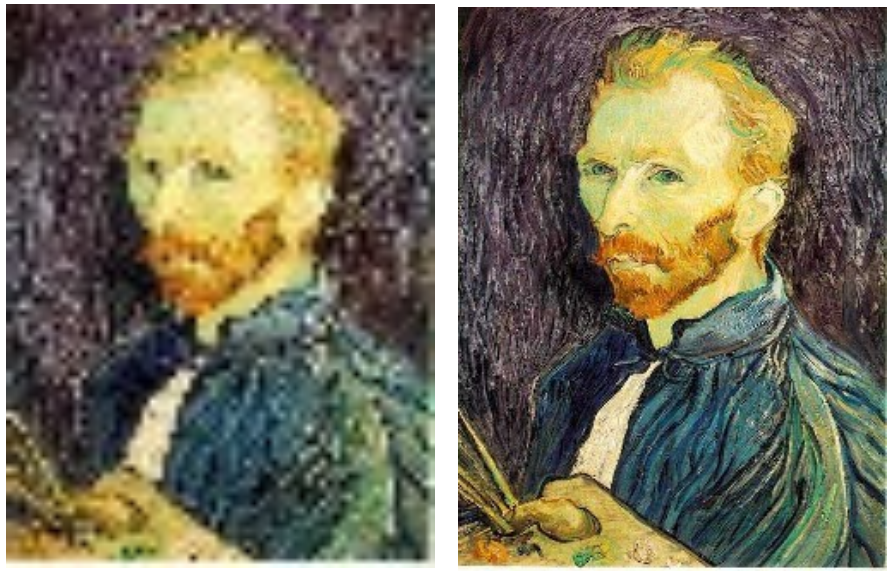
- Color
- Image sampling and quantization
- Image histograms
- Images as functions
- Linear systems (filters)
- Convolution and correlation

Applications of linear systems or filters

De-noising

Salt and pepper noise

Super-resolution



In-painting

Bertamio et al

Systems and Filters

Filtering:

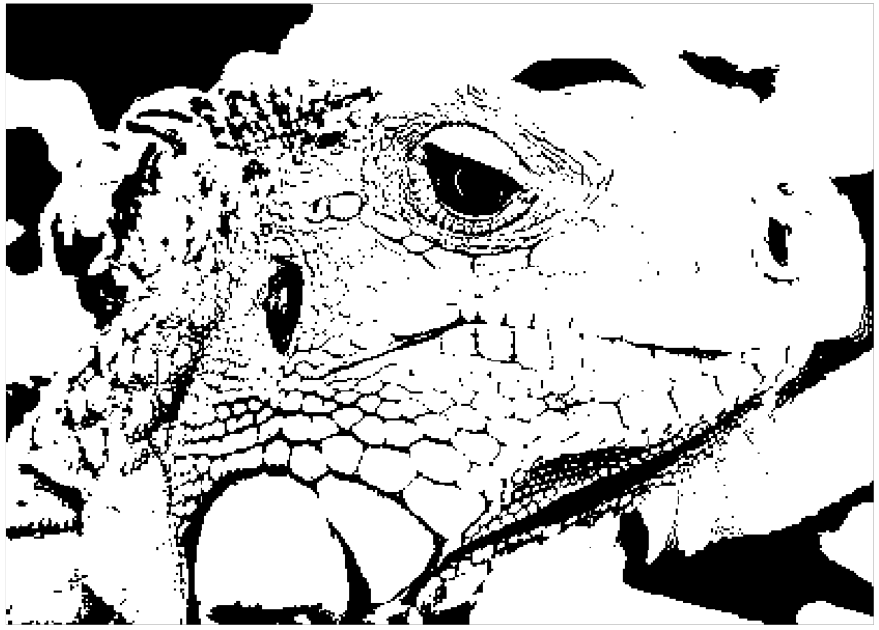
– Forming a new image whose pixel values are transformed from original pixel values

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

Intuition behind linear systems

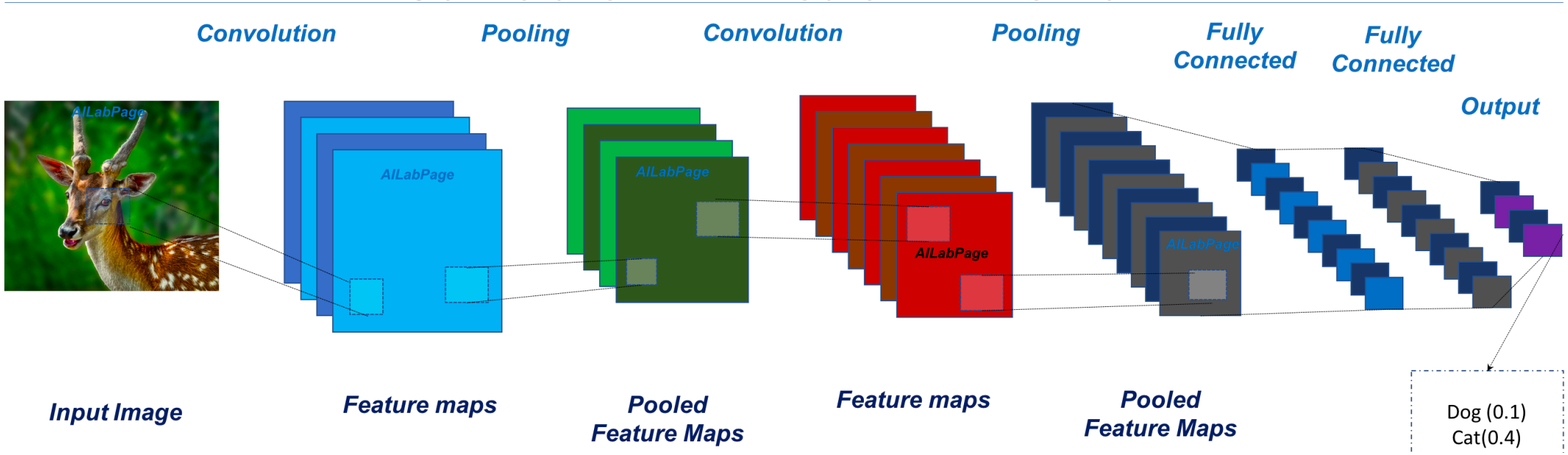
- We will view linear systems are a type of function that operates over images
- Translating an image or multiplying by a constant leaves the semantic content intact
  - but can reveal interesting patterns



Aside

- Neural networks and specifically convolutional neural networks are a type of system or non-linear system that contains multiple individual linear sub-systems.

- (we will learn more about this later in class)



System and Filters

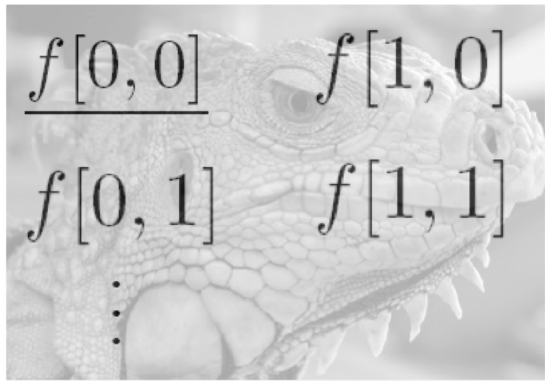
- we define a system as a unit that converts an input function f[n,m] into an output (or response) function g[n,m], where (n,m) are the independent variables.
  - In the case for images, (n,m) represents the **spatial position in the image.**

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

Images as coordinates

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

2D discrete systems (filters)

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

Original image

Smoothed image

Filter example #1: Moving Average

2D DS 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

$f[n,m]$

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

$g[n,m]$

Courtesy of S. Seitz

Filter example #1: Moving Average

$f[n,m]$

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

$g[n,m]$

	0	10							

Filter example #1: Moving Average

$f[n,m]$

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

$g[n,m]$

	0	10	20						

Filter example #1: Moving Average

$f[n,m]$

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

$g[n,m]$

	0	10	20	30					

Filter example #1: Moving Average

$f[n,m]$

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

$g[n,m]$

	0	10	20	30	30				

Filter example #1: Moving Average

$f[n,m]$

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

$g[n,m]$

	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	

Filter example #1: Moving Average

In summary:

- This filter "Replaces" each pixel with an average of its neighborhood.

- Achieve smoothing effect (remove sharp features)

$$h[\cdot\,,\cdot\,]$$

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

Filter example #1: Moving Average

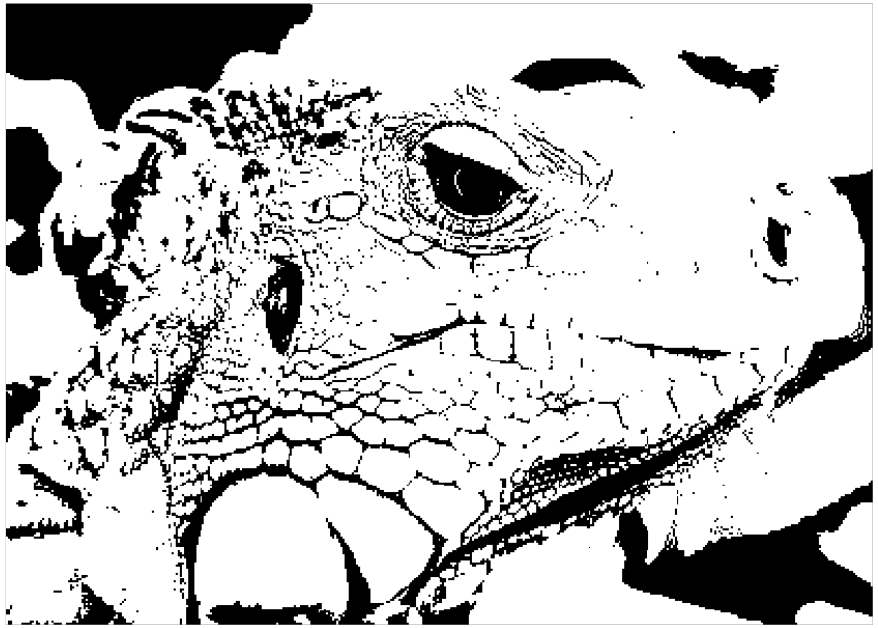
Original image

Smoothed image

Filter example #2: Image Segmentation

- Image segmentation based on a simple threshold:

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



Properties of systems

- 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

$$S^{-1}[S[f_i[n,m]]] = f[n,m]$$

Properties of systems

- Spatial properties
  - Causality

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

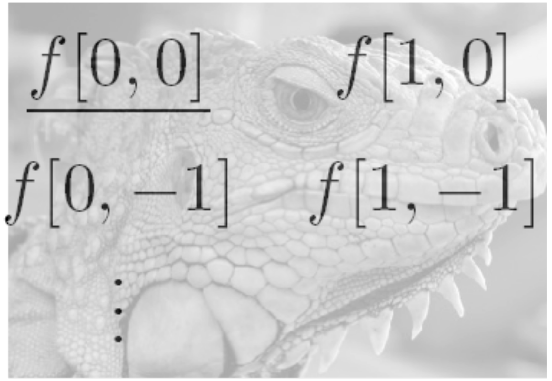
  - Shift invariance:

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

What does shifting an image look like?

Cartesian coordinates

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



Is the moving average system is shift invariant?

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

$\mathrm{f}[n,m]$

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

$\mathrm{g}[n,m]$

	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 system is shift invariant?

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

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

$$\mathrm{g}[n-n_0,m-m_0] = \frac{1}{9}\sum_{k=-1}^{1}\sum_{l=-1}^{1} f[n-n_0-k,m-m_0-l]$$

Let $f[n-n_0,m-m_0]$ be a shifted input of $f[n,m]$

Now let's pass $f[n-n_0,m-m_0]$ through the system:

$$f[n-n_0,m-m_0] \xrightarrow{s} \frac{1}{9}\sum_{k=-1}^{1}\sum_{l=-1}^{1} f[n-n_0-k,m-m_0-l]$$

Yes!

Is the moving average system is casual?

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

$\mathrm{f}[n,m]$

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

$\mathrm{g}[n,m]$

	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	

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

Linear Systems (filters)

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

- 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*

$$S[\,\alpha f_i[n,m] + \beta f_j[k,l]\,] \quad = \quad \alpha\, S[\,f_i[n,m]] \;+\; \beta\, S[\,f_j[k,l]\,]$$

superposition property

Linear Systems (filters)

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

- 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!

Linear shift invariant systems

- Satisfies two properties:

- Superposition property

$$S[\,\alpha f_i[n,m] + \beta f_j[k,l]\,] \quad = \quad \alpha\, S[\,f_i[n,m]] \;+\; \beta\, S[\,f_j[k,l]\,]$$

- Shift invariance:

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

moving average system is shift invariant and linear

- We are going to use this as an example to dive into interesting properties about linear shift-invariant systems.

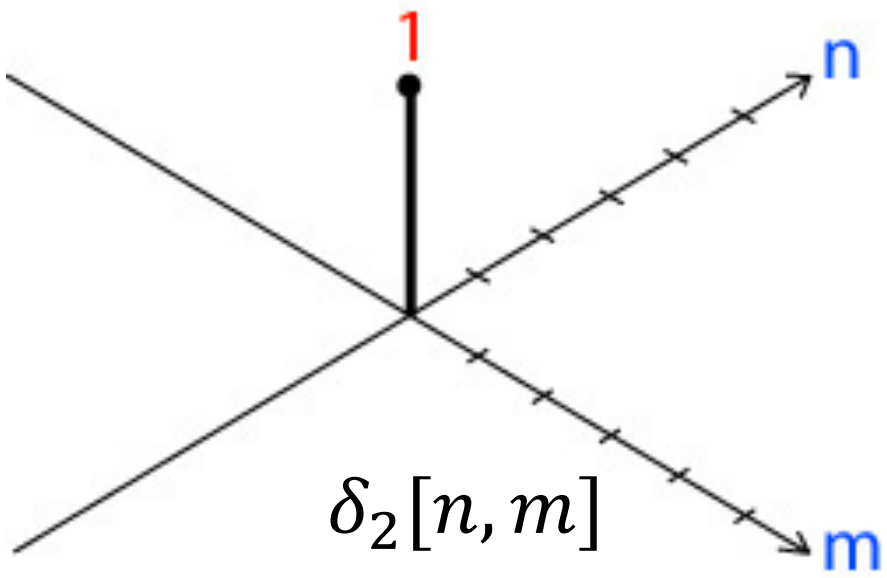
- Why are linear shift invariant systems important?

Our visual system is a

linear shift invariant system

2D impulse function

- Let's look at a special function
- 1 at [0,0].
- 0 everywhere else

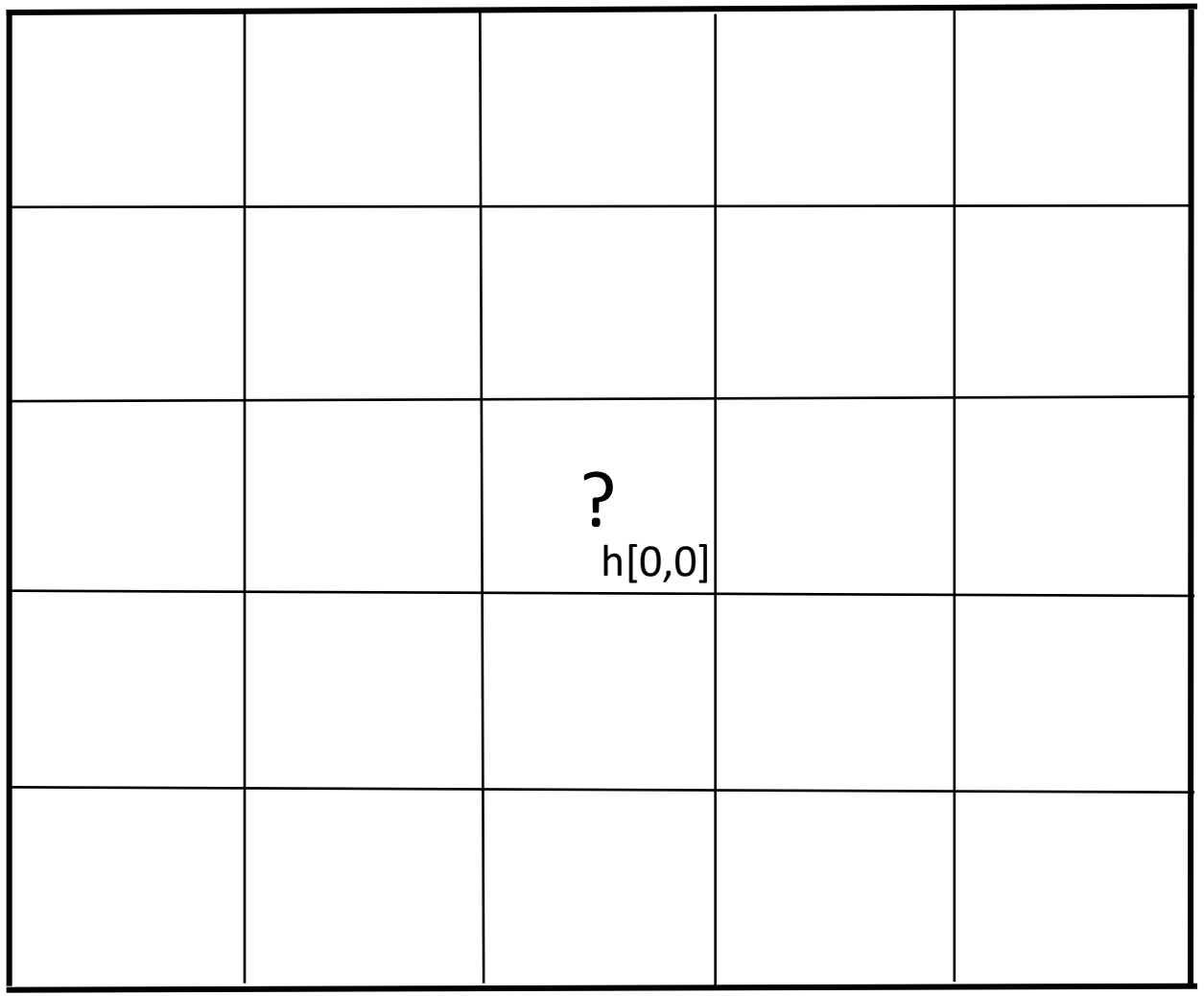


Impulse response to the moving average filter

$$\delta_2 \xrightarrow{S} h[n,m]$$

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

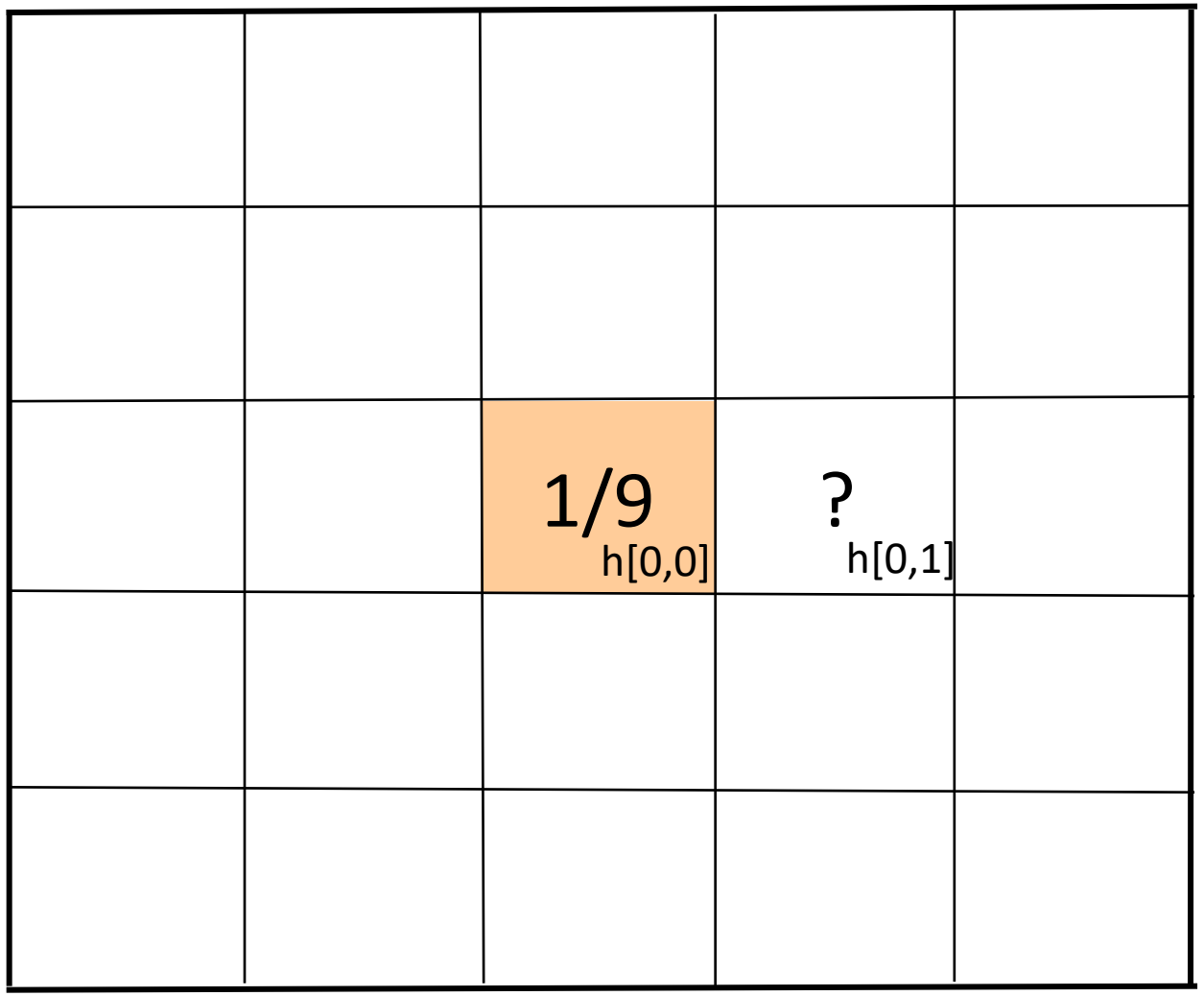
Impulse response to the moving average filter



$$\delta_2 \xrightarrow{S} h[n,m]$$

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

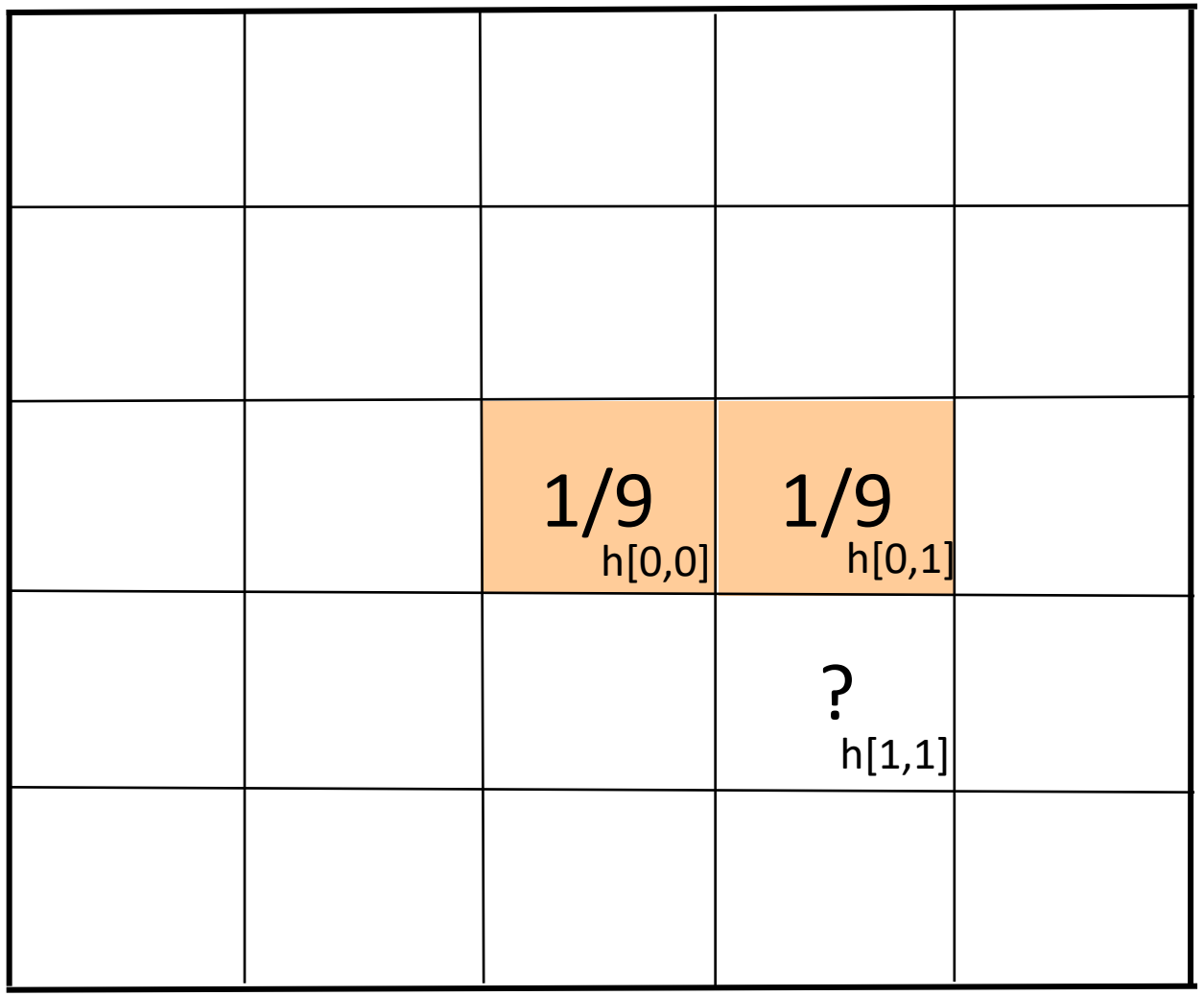
Impulse response to the moving average filter



$$\delta_2 \xrightarrow{S} h[n,m]$$

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

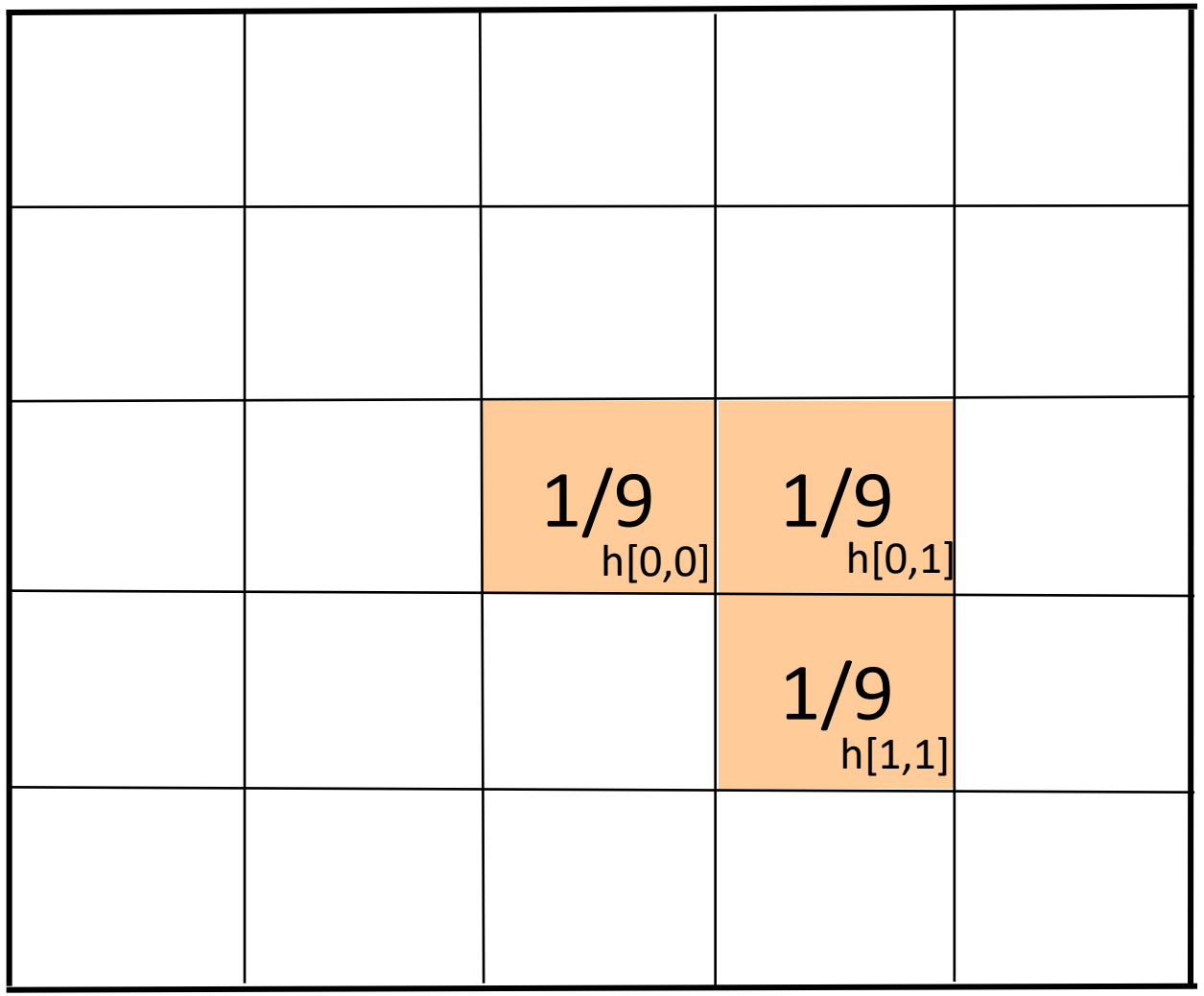
Impulse response to the moving average filter



$$\delta_2 \xrightarrow{S} h[n,m]$$

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

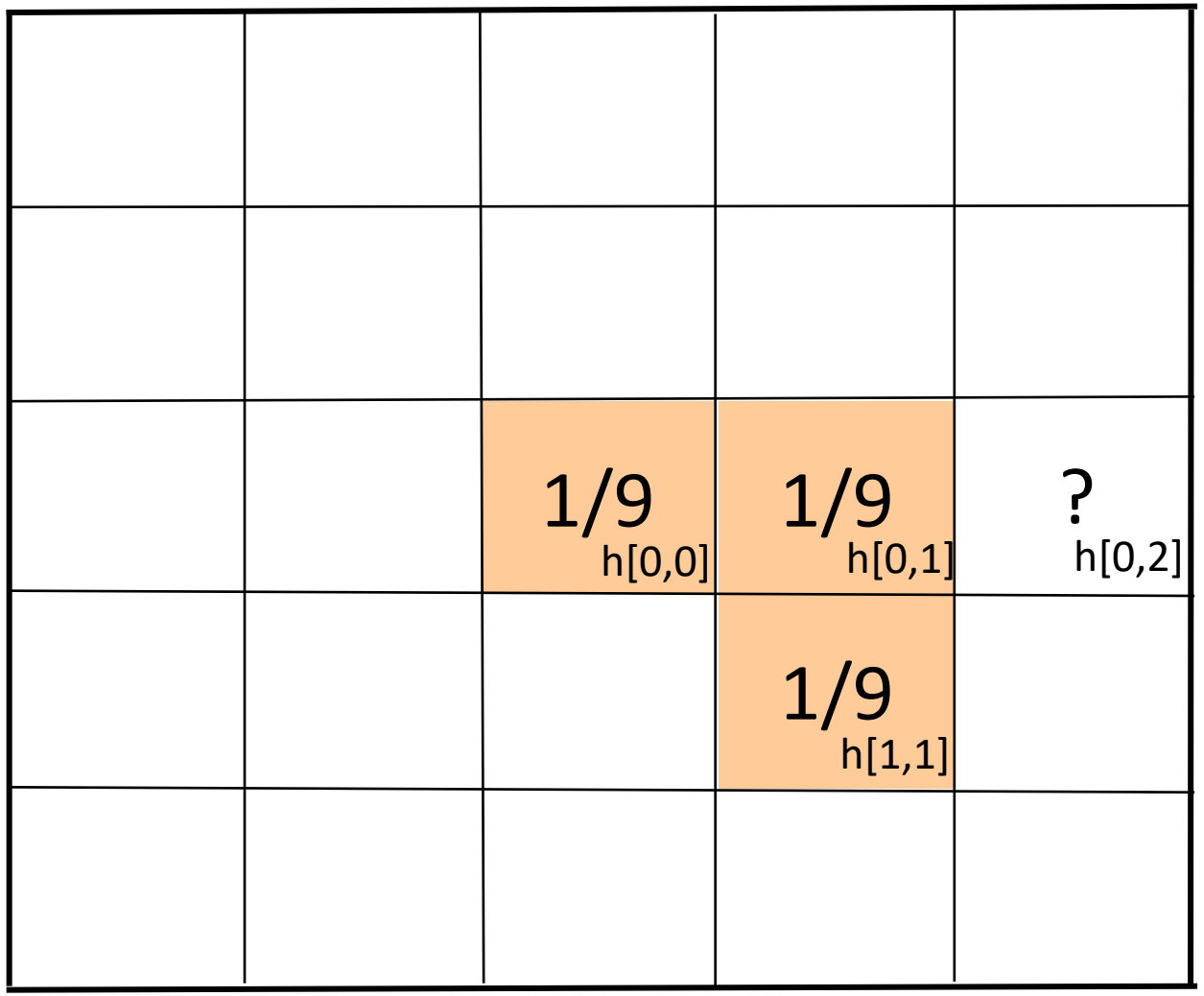
Impulse response to the moving average filter



$$\delta_2 \xrightarrow{S} h[n,m]$$

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

Impulse response to the moving average filter



$$\delta_2 \xrightarrow{S} h[n,m]$$

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

Impulse response to the moving average filter

		1/9 h[0,0]	1/9 h[0,1]	0 h[0,2]
			1/9 h[1,1]	

$$\delta_2 \xrightarrow{S} h[n,m]$$

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

Impulse response to the moving average filter

0	0	0	0	0
0	1/9 h[-1,-1]	1/9	1/9	0
0	1/9	1/9 h[0,0]	1/9 h[0,1]	0 h[0,2]
0	1/9	1/9	1/9 h[1,1]	0
0	0	0	0	0

$$\delta_2 \xrightarrow{S} g[n,m]$$

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

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

Any linear shift invariant system

- By passing an impulse response into a linear system, we get it's impulse response.
- So, if we don't know what the linear system is doing, we can pass an impulse into it to get a filter $h[n,m]$ that tells us what the system is actually doing.

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

- But how do we use $h[n,m]$ to calculate $\mathrm{g}[n,m]$ from $\mathrm{f}[n,m]$

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

Remember the Moving Average filter and how we already used it's impulse response

$f[n,m]$

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

$g[n,m]$

Courtesy of S. Seitz

Remember the Moving Average filter and how we already used it's impulse response

$f[n,m]$

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

$g[n,m]$

	0	10							

Remember the Moving Average filter and how we already used it's impulse response

$f[n,m]$

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

$g[n,m]$

	0	10	20						

Remember the Moving Average filter and how we already used it's impulse response

$f[n,m]$

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

$g[n,m]$

	0	10	20	30					

Remember the Moving Average filter and how we already used it's impulse response

$f[n,m]$

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

$g[n,m]$

	0	10	20	30	30				

Remember the Moving Average filter and how we already used it's impulse response

$f[n,m]$

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

$g[n,m]$

	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	

General linear shift invariant system

- Let's say our input f is a 3x3 image:

f[0,0]	f[0,1]	f[1,1]
f[1,0]	f[1,1]	f[1,2]
f[2,0]	f[2,1]	f[2,2]

We can rewrite $f[n,m]$ as a sum of delta functions:

$$\begin{aligned} f[n,m] = {} & f[0,0] \times \delta_2[n-0, m-0] \\ & + f[0,1] \times \delta_2[n-0, m-1] \\ & + \ \ldots \\ & + f[n,m] \times \delta_2[0,0] \\ & + \ \ldots \end{aligned}$$

Or you can write it as: $$f[n,m] = \sum_{k=-\infty}^{\infty} \sum_{l=\infty}^{\infty} f[k,l] \times \delta_2[n-k, m-l]$$

3 properties we need:

Now, we know what happens when we send a delta function through a LSI system:

$$\delta_2[n,m] \longrightarrow \boxed{\text{System } \mathcal{S}} \longrightarrow h[n,m]$$

We also know that LSI systems shift the output if the input is shifted:

$$\delta_2[n-k,m-l] \longrightarrow \boxed{\text{System } \mathcal{S}} \longrightarrow h[n-k,m-l]$$

Finally, the superposition principle:

$$S[\,\alpha f_i[n,m] + \beta f_j[k,l]\,] \quad = \quad \alpha\, S[\,f_i[n,m]] \;+\; \beta\, S[\,f_j[k,l]\,]$$

We can generalize this superposition principle…

$$S[\,\alpha f_i[n,m] + \beta f_j[k,l]\,] \quad = \quad \alpha\, S[\,f_i[n,m]] \;+\; \beta\, S[\,f_j[k,l]\,]$$

… with our summation of deltas…

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

… as follows:

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

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

Linear shift invariant system

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

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

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

Linear Shift Invariant systems

An LSI system is completely specified by its impulse response.

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

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

Discrete convolution

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

What we will learn today?

- Color
- Image sampling and quantization
- Image histograms
- Images as functions
- Linear systems (filters)
- Convolution and correlation

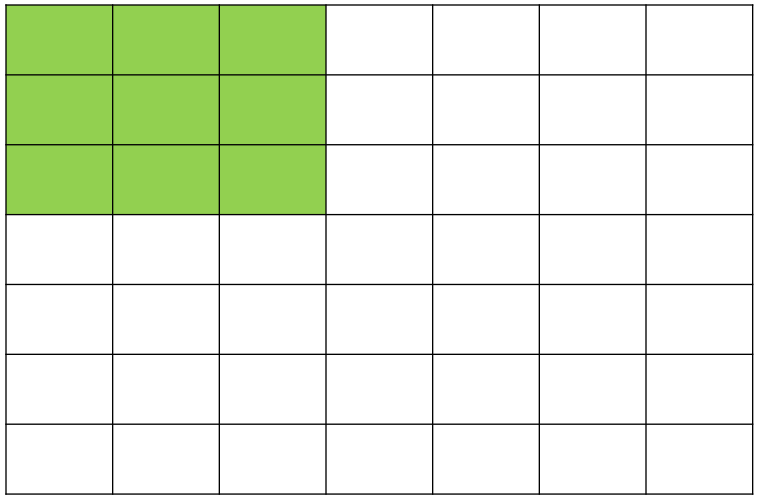
Some background reading:
Forsyth and Ponce, Computer Vision, Chapter 7

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.

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



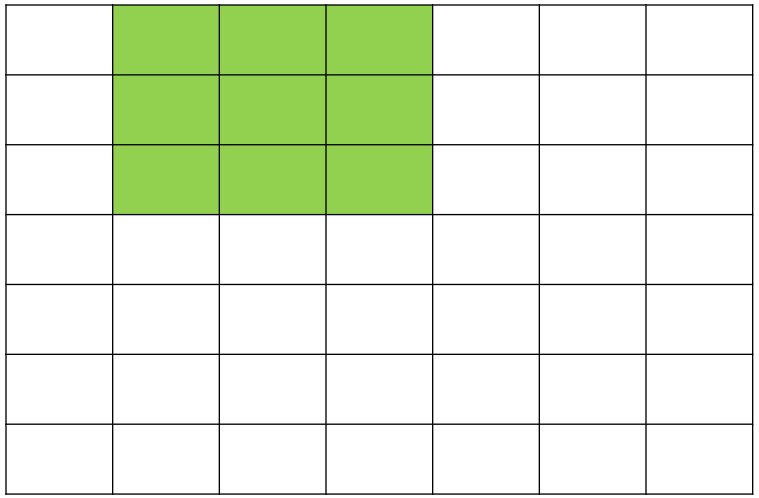
Assume we have a filter(h[,]) that is 3x3. and an image (f[,]) that is 7x7.

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.

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



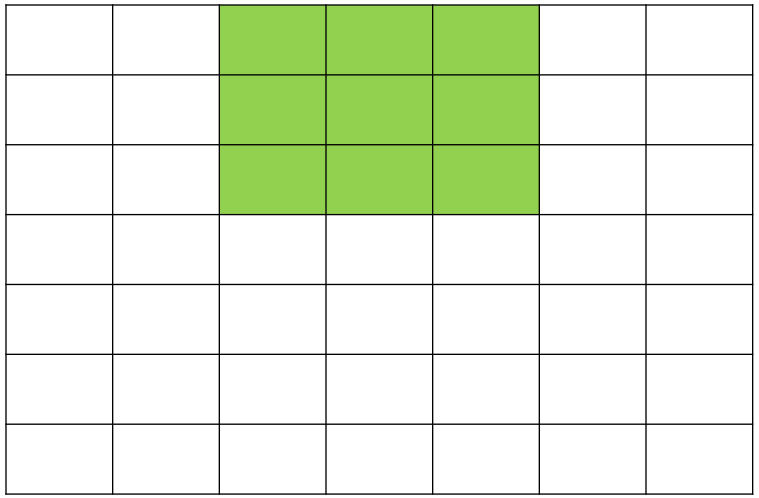
Assume we have a filter(h[,]) that is 3x3. and an image (f[,]) that is 7x7.

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.

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



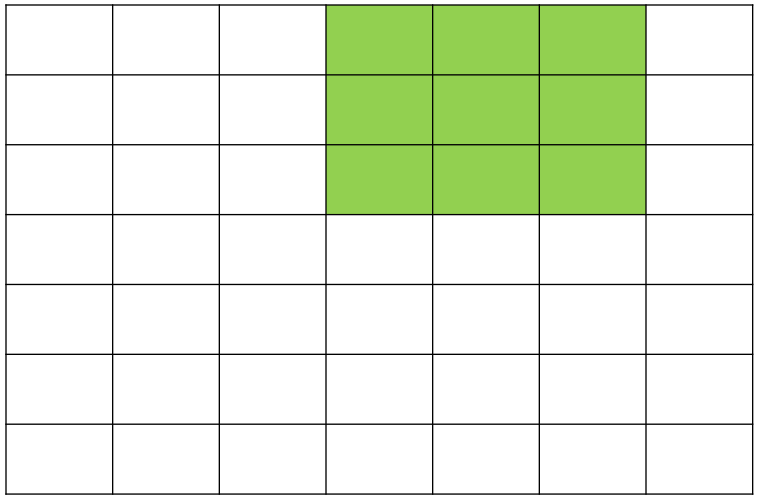
Assume we have a filter(h[,]) that is 3x3. and an image (f[,]) that is 7x7.

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.

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



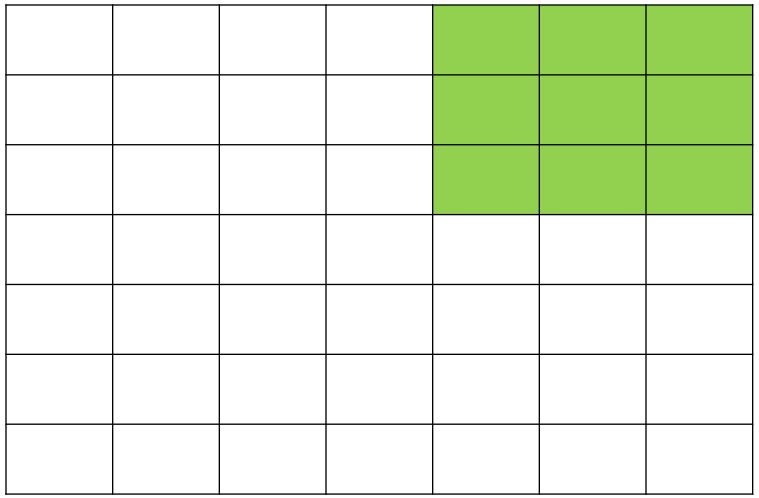
Assume we have a filter(h[,]) that is 3x3. and an image (f[,]) that is 7x7.

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.

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



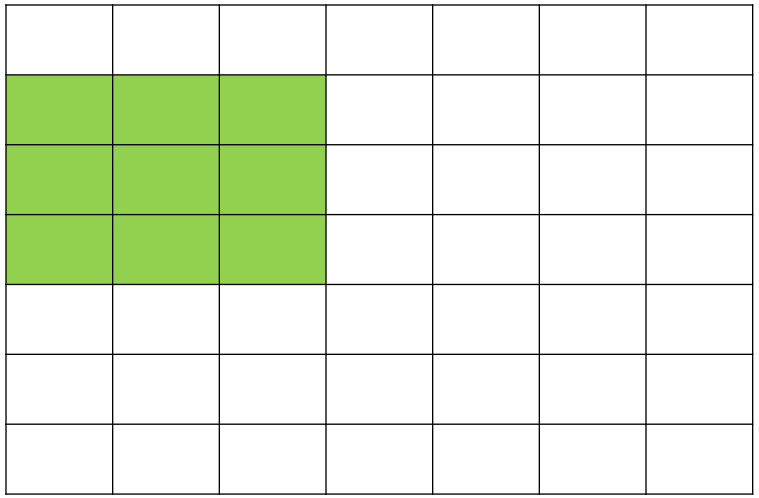
Assume we have a filter(h[,]) that is 3x3. and an image (f[,]) that is 7x7.

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.

Convolution

$$f * h = \sum_k \sum_l f(k,l)\,h(-k,-l)$$

f = Image

h = Kernel

h

h_1	h_2	h_3
h_4	h_5	h_6
h_7	h_8	h_9

$X - flip$

h_7	h_8	h_9
h_4	h_5	h_6
h_1	h_2	h_3

$Y - flip$

f

f_1	f_2	f_3
f_4	f_5	f_6
f_7	f_8	f_9

$*$

h_9	h_8	h_7
h_6	h_5	h_4
h_3	h_2	h_1

$$\begin{aligned} f * h = {} & f_1 h_9 + f_2 h_8 + f_3 h_7 \\ & + f_4 h_6 + f_5 h_5 + f_6 h_4 \\ & + f_7 h_3 + f_8 h_2 + f_9 h_1 \end{aligned}$$

Slide credit: Ulas Bagci

2D convolution example

Input

1	2	3
4	5	6
7	8	9

Kernel

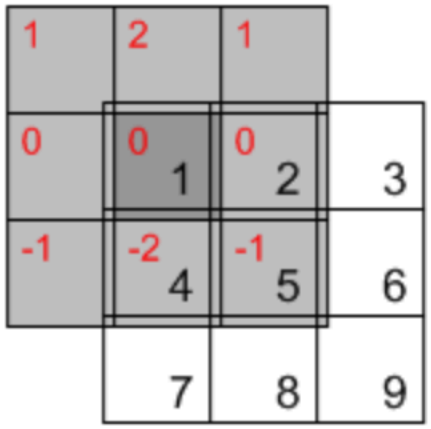
n \ m	-1	0	1
-1	-1	-2	-1
0	0	0	0
1	1	2	1

Output

-13	-20	-17
-18	-24	-18
13	20	17

Slide credit: Song Ho Ahn

2D convolution example



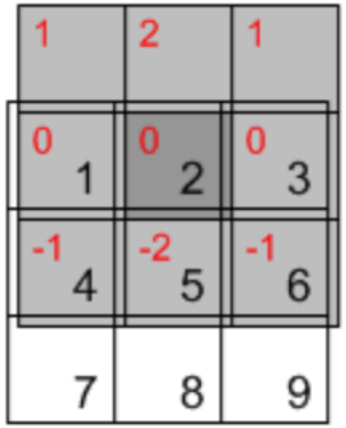
$$
\begin{aligned}
&= x[-1,-1]\cdot h[1,1] + x[0,-1]\cdot h[0,1] + x[1,-1]\cdot h[-1,1] \\
&\quad + x[-1,0]\cdot h[1,0] + x[0,0]\cdot h[0,0] + x[1,0]\cdot h[-1,0] \\
&\quad + x[-1,1]\cdot h[1,-1] + x[0,1]\cdot h[0,-1] + x[1,1]\cdot h[-1,-1] \\
&= 0\cdot 1 + 0\cdot 2 + 0\cdot 1 + 0\cdot 0 + 1\cdot 0 + 2\cdot 0 + 0\cdot(-1) + 4\cdot(-2) + 5\cdot(-1) = -13
\end{aligned}
$$

-13	-20	-17
-18	-24	-18
13	20	17

Output

Slide credit: Song Ho Ahn

2D convolution example



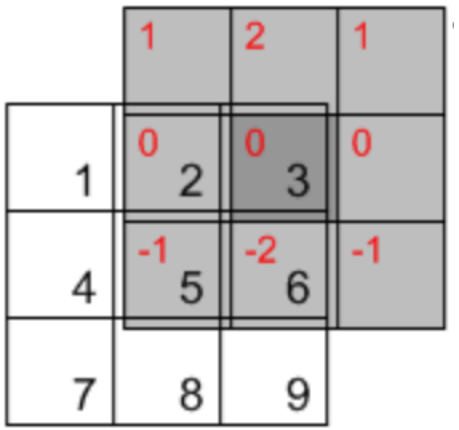
$$
\begin{aligned}
&= x[0,-1]\cdot h[1,1] + x[1,-1]\cdot h[0,1] + x[2,-1]\cdot h[-1,1] \\
&\quad + x[0,0]\cdot h[1,0] + x[1,0]\cdot h[0,0] + x[2,0]\cdot h[-1,0] \\
&\quad + x[0,1]\cdot h[1,-1] + x[1,1]\cdot h[0,-1] + x[2,1]\cdot h[-1,-1] \\
&= 0\cdot 1 + 0\cdot 2 + 0\cdot 1 + 1\cdot 0 + 2\cdot 0 + 3\cdot 0 + 4\cdot(-1) + 5\cdot(-2) + 6\cdot(-1) = -20
\end{aligned}
$$

-13	-20	-17
-18	-24	-18
13	20	17

Output

Slide credit: Song Ho Ahn

2D convolution example



$$
\begin{aligned}
&= x[1,-1]\cdot h[1,1] + x[2,-1]\cdot h[0,1] + x[3,-1]\cdot h[-1,1] \\
&\quad + x[1,0]\cdot h[1,0] + x[2,0]\cdot h[0,0] + x[3,0]\cdot h[-1,0] \\
&\quad + x[1,1]\cdot h[1,-1] + x[2,1]\cdot h[0,-1] + x[3,1]\cdot h[-1,-1] \\
&= 0\cdot 1 + 0\cdot 2 + 0\cdot 1 + 2\cdot 0 + 3\cdot 0 + 0\cdot 0 + 5\cdot(-1) + 6\cdot(-2) + 0\cdot(-1) = -17
\end{aligned}
$$

-13	-20	-17
-18	-24	-18
13	20	17

Output

Slide credit: Song Ho Ahn

2D convolution example

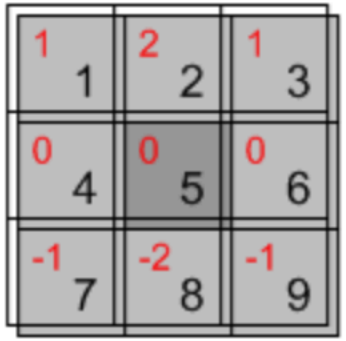
$$
\begin{aligned}
&= x[-1,0]\cdot h[1,1] + x[0,0]\cdot h[0,1] + x[1,0]\cdot h[-1,1] \\
&\quad + x[-1,1]\cdot h[1,0] + x[0,1]\cdot h[0,0] + x[1,1]\cdot h[-1,0] \\
&\quad + x[-1,2]\cdot h[1,-1] + x[0,2]\cdot h[0,-1] + x[1,2]\cdot h[-1,-1] \\
&= 0\cdot 1 + 1\cdot 2 + 2\cdot 1 + 0\cdot 0 + 4\cdot 0 + 5\cdot 0 + 0\cdot(-1) + 7\cdot(-2) + 8\cdot(-1) = -18
\end{aligned}
$$

-13	-20	-17
-18	-24	-18
13	20	17

Output

Slide credit: Song Ho Ahn

2D convolution example



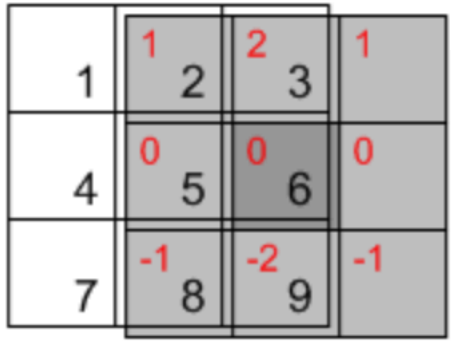
$$
\begin{aligned}
&= x[0,0]\cdot h[1,1] + x[1,0]\cdot h[0,1] + x[2,0]\cdot h[-1,1] \\
&\quad + x[0,1]\cdot h[1,0] + x[1,1]\cdot h[0,0] + x[2,1]\cdot h[-1,0] \\
&\quad + x[0,2]\cdot h[1,-1] + x[1,2]\cdot h[0,-1] + x[2,2]\cdot h[-1,-1] \\
&= 1\cdot 1 + 2\cdot 2 + 3\cdot 1 + 4\cdot 0 + 5\cdot 0 + 6\cdot 0 + 7\cdot(-1) + 8\cdot(-2) + 9\cdot(-1) = -24
\end{aligned}
$$

-13	-20	-17
-18	-24	-18
13	20	17

Output

Slide credit: Song Ho Ahn

2D convolution example



$$
\begin{aligned}
&= x[1,0]\cdot h[1,1] + x[2,0]\cdot h[0,1] + x[3,0]\cdot h[-1,1] \\
&\quad + x[1,1]\cdot h[1,0] + x[2,1]\cdot h[0,0] + x[3,1]\cdot h[-1,0] \\
&\quad + x[1,2]\cdot h[1,-1] + x[2,2]\cdot h[0,-1] + x[3,2]\cdot h[-1,-1] \\
&= 2\cdot 1 + 3\cdot 2 + 0\cdot 1 + 5\cdot 0 + 6\cdot 0 + 0\cdot 0 + 8\cdot(-1) + 9\cdot(-2) + 0\cdot(-1) = -18
\end{aligned}
$$

-13	-20	-17
-18	-24	-18
13	20	17

Output

Slide credit: Song Ho Ahn

Convolution in 2D - examples

Original

$*$

•0	•0	•0
•0	•1	•0
•0	•0	•0

$=$?

Convolution in 2D - examples

Original

$*$

•0	•0	•0
•0	•1	•0
•0	•0	•0

$=$

Filtered
(no change)

Convolution in 2D - examples

Original

$*$

•0	•0	•0
•0	•0	•1
•0	•0	•0

$=$?

Convolution in 2D - examples

Original

$*$

•0	•0	•0
•0	•0	•1
•0	•0	•0

$=$

Shifted right
By 1 pixel

Convolution in 2D - examples

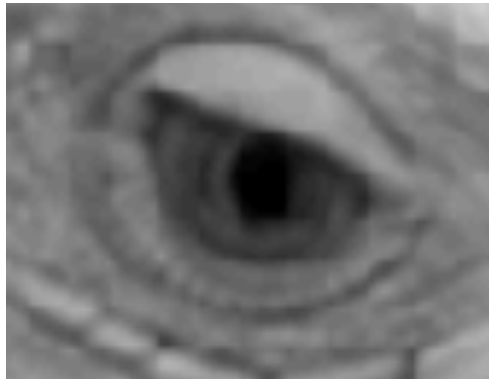
Original

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

Convolution in 2D - examples

Original

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



Blur (with a box filter)

Convolution in 2D - examples

Original

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

(Note that filter sums to 1)

"details of the image"

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

What does blurring take away?

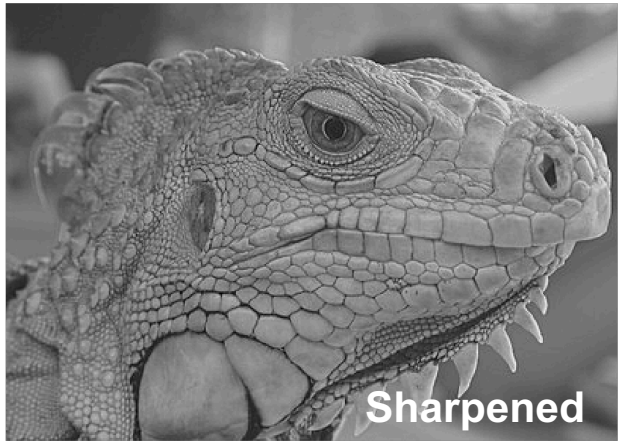
–

=

- Let’s add it back:

+

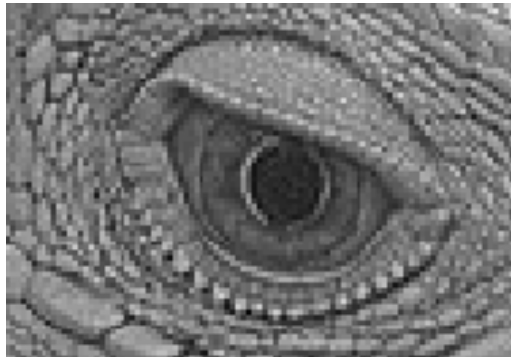
=



Convolution in 2D – Sharpening filter

Original

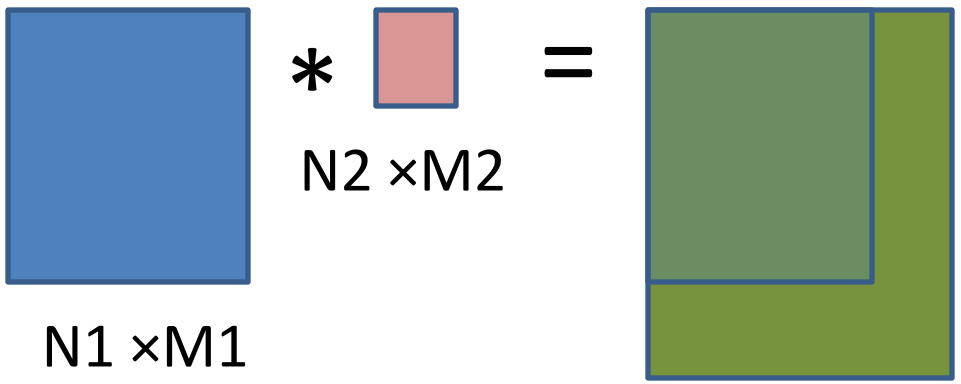
$$\begin{bmatrix} \bullet 0 & \bullet 0 & \bullet 0 \\ \bullet 0 & \bullet 2 & \bullet 0 \\ \bullet 0 & \bullet 0 & \bullet 0 \end{bmatrix} - \frac{1}{9}\begin{bmatrix} \bullet 1 & \bullet 1 & \bullet 1 \\ \bullet 1 & \bullet 1 & \bullet 1 \\ \bullet 1 & \bullet 1 & \bullet 1 \end{bmatrix} =$$



Sharpening filter: Accentuates differences with local average

Implementation detail: Image support and edge effect

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

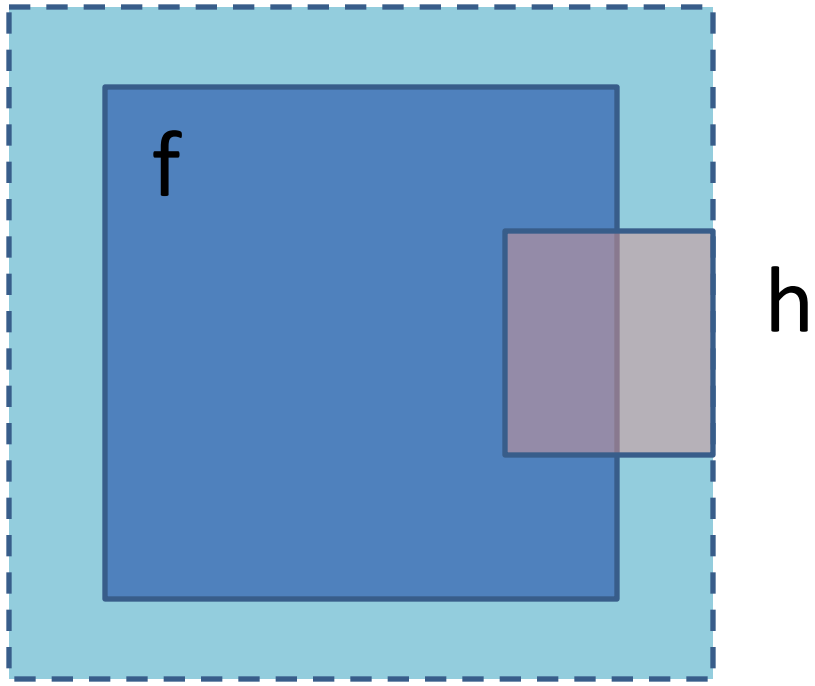


$(N1 + N2 - 1) \times (M1 + M2 - 1)$

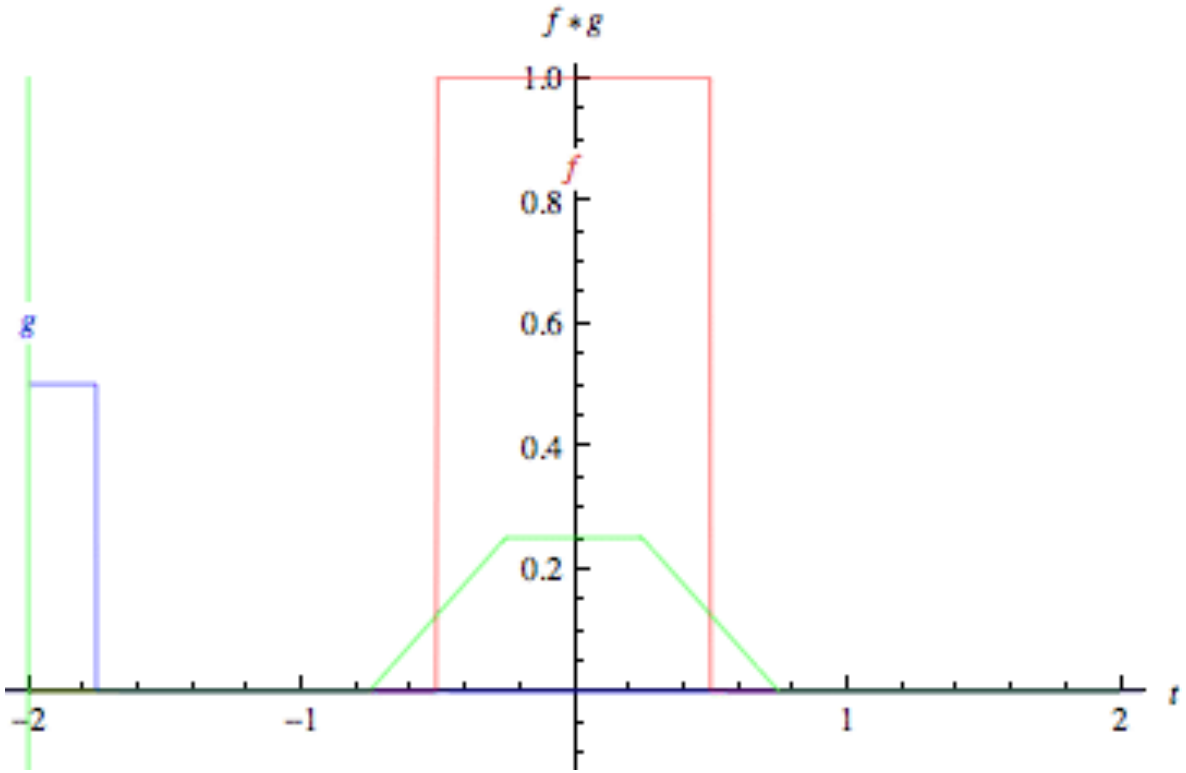
Image support and edge effect

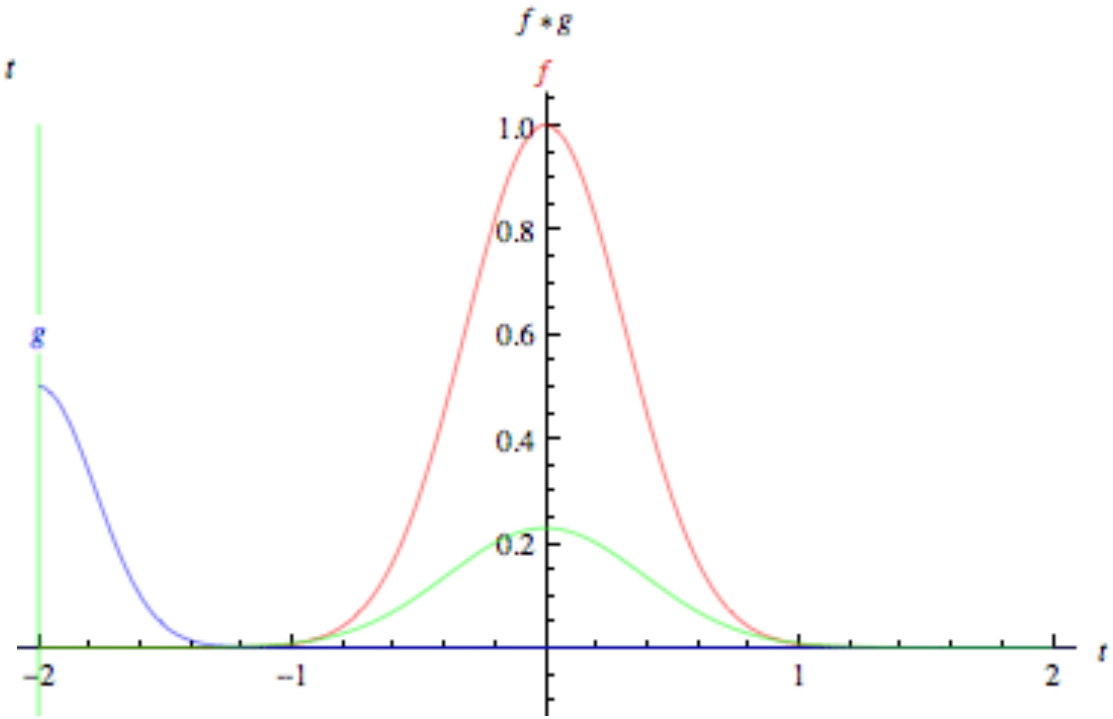
•A computer will only convolve **finite support signals.**

• What happens at the edge?



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





Slide credit: Wolfram Alpha

What we will learn today?

- Color
- Image sampling and quantization
- Image histograms
- Images as functions
- Linear systems (filters)
- Convolution and correlation

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

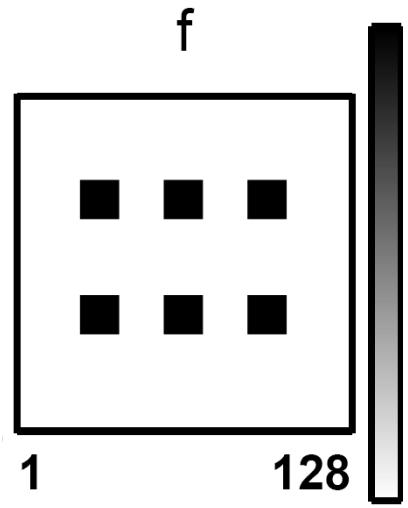
(Cross) correlation – symbol: $**$

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

$$f[n,m] ** h[n,m] = \sum_k \sum_l f[k,l]h[n-k,m-l]$$

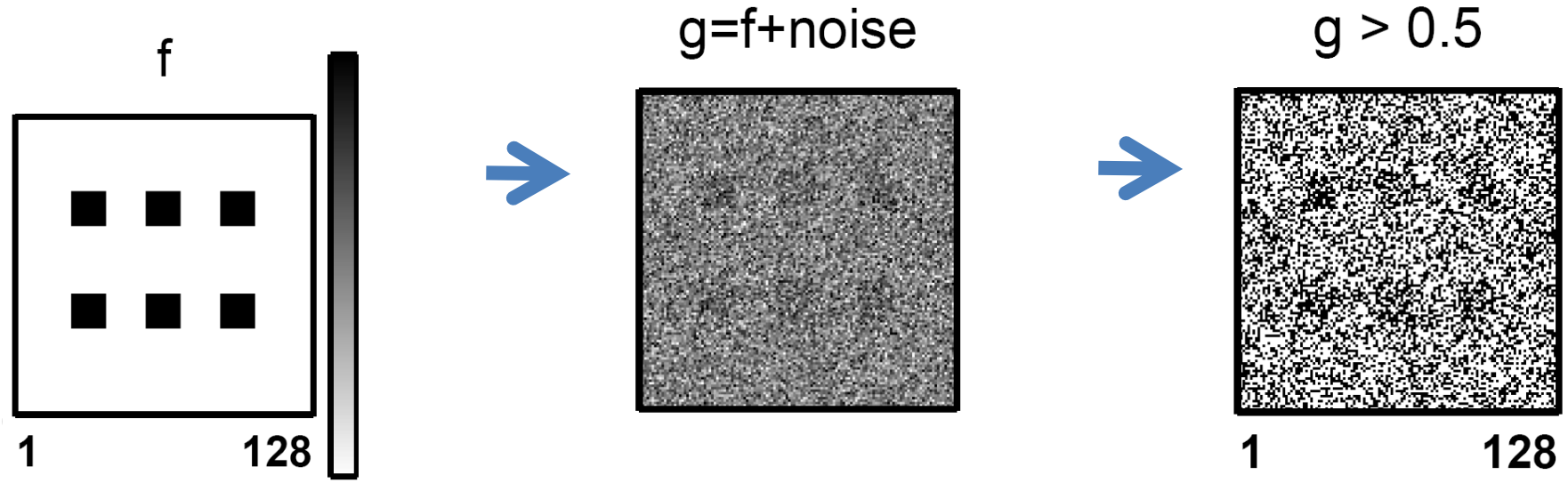
- Equivalent to a convolution without the flip
- Use it to measure 'similarity' between f and h.

(Cross) correlation – example



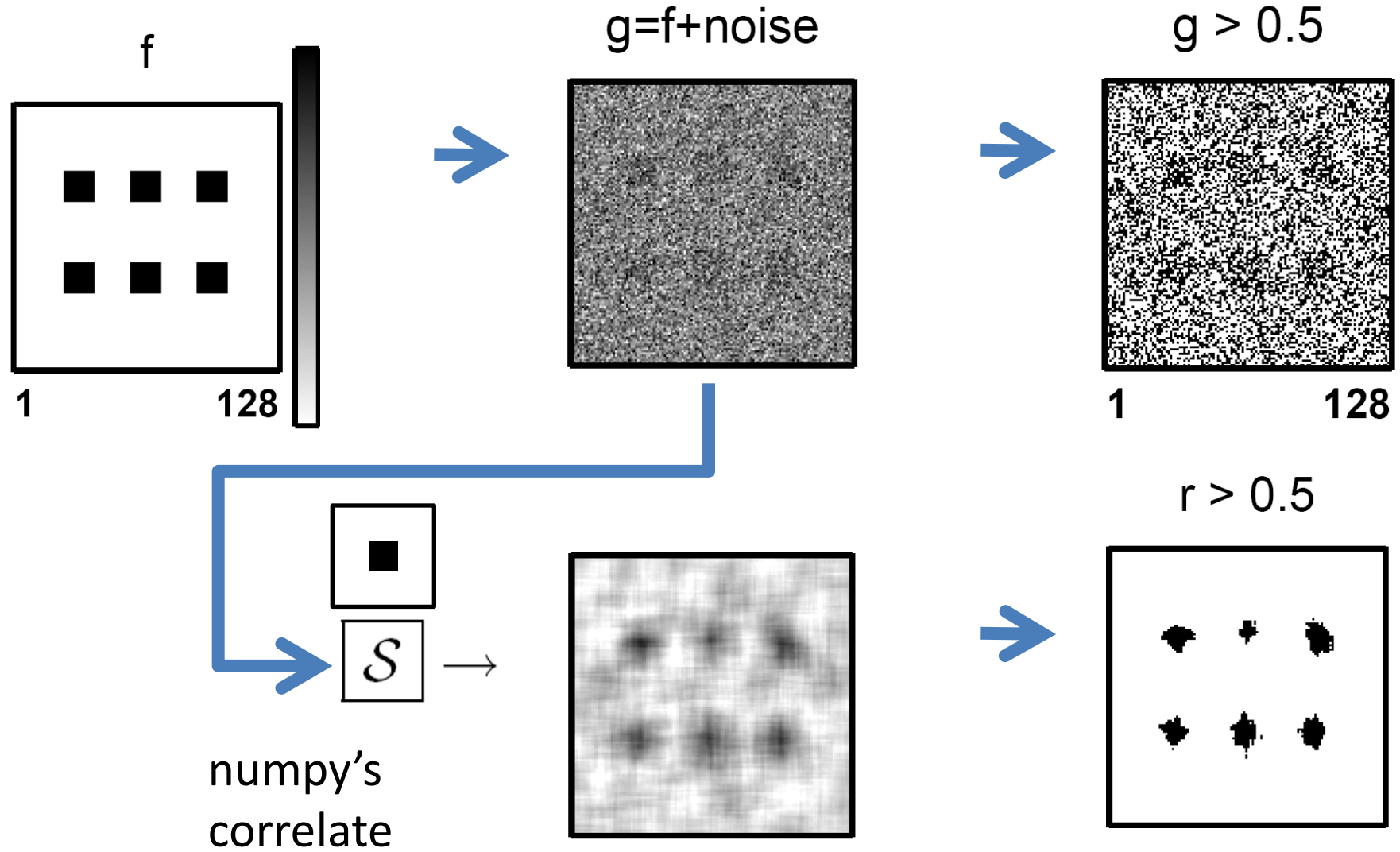
Courtesy of J. Fessler

(Cross) correlation – example



Courtesy of J. Fessler

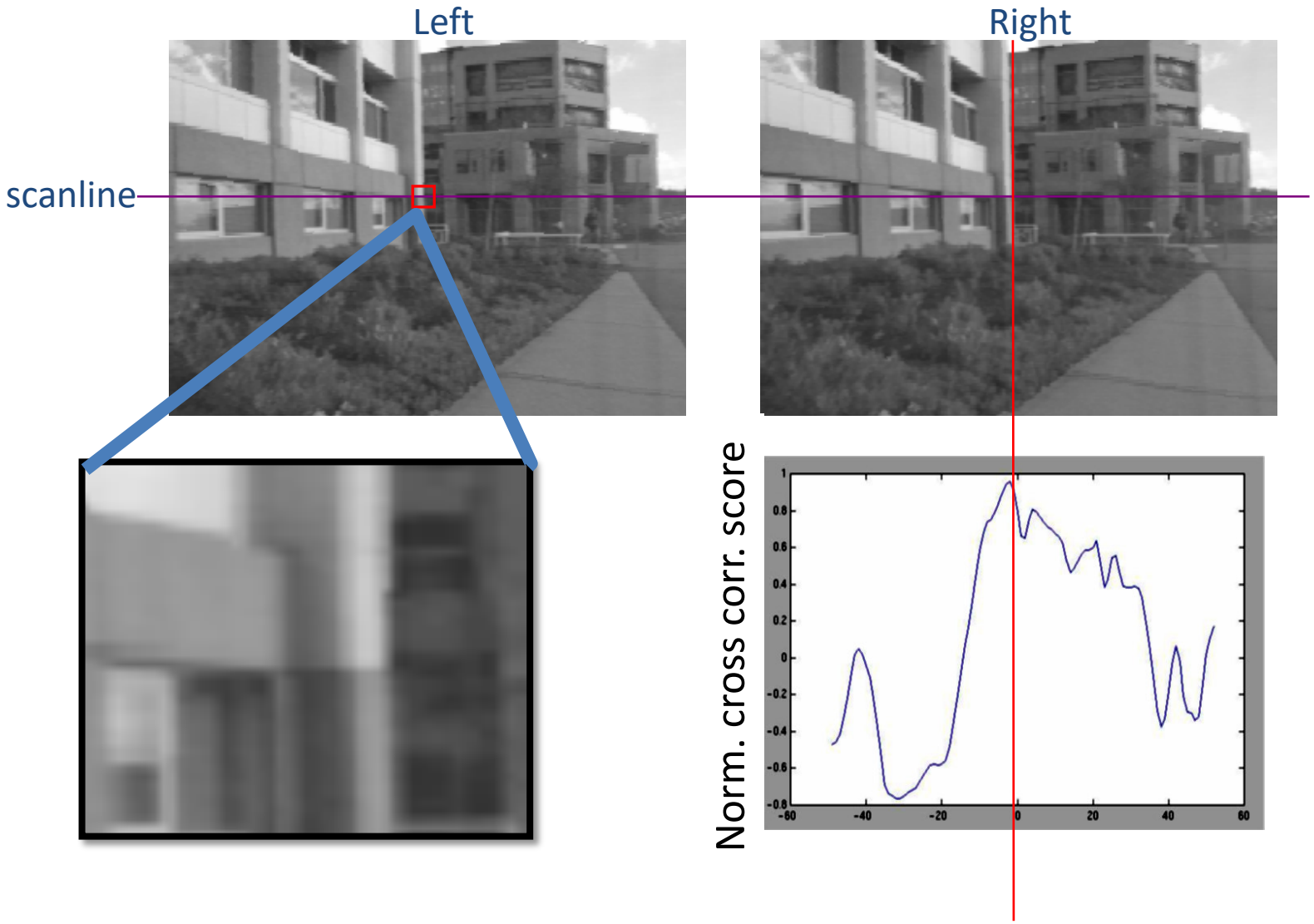
(Cross) correlation – example

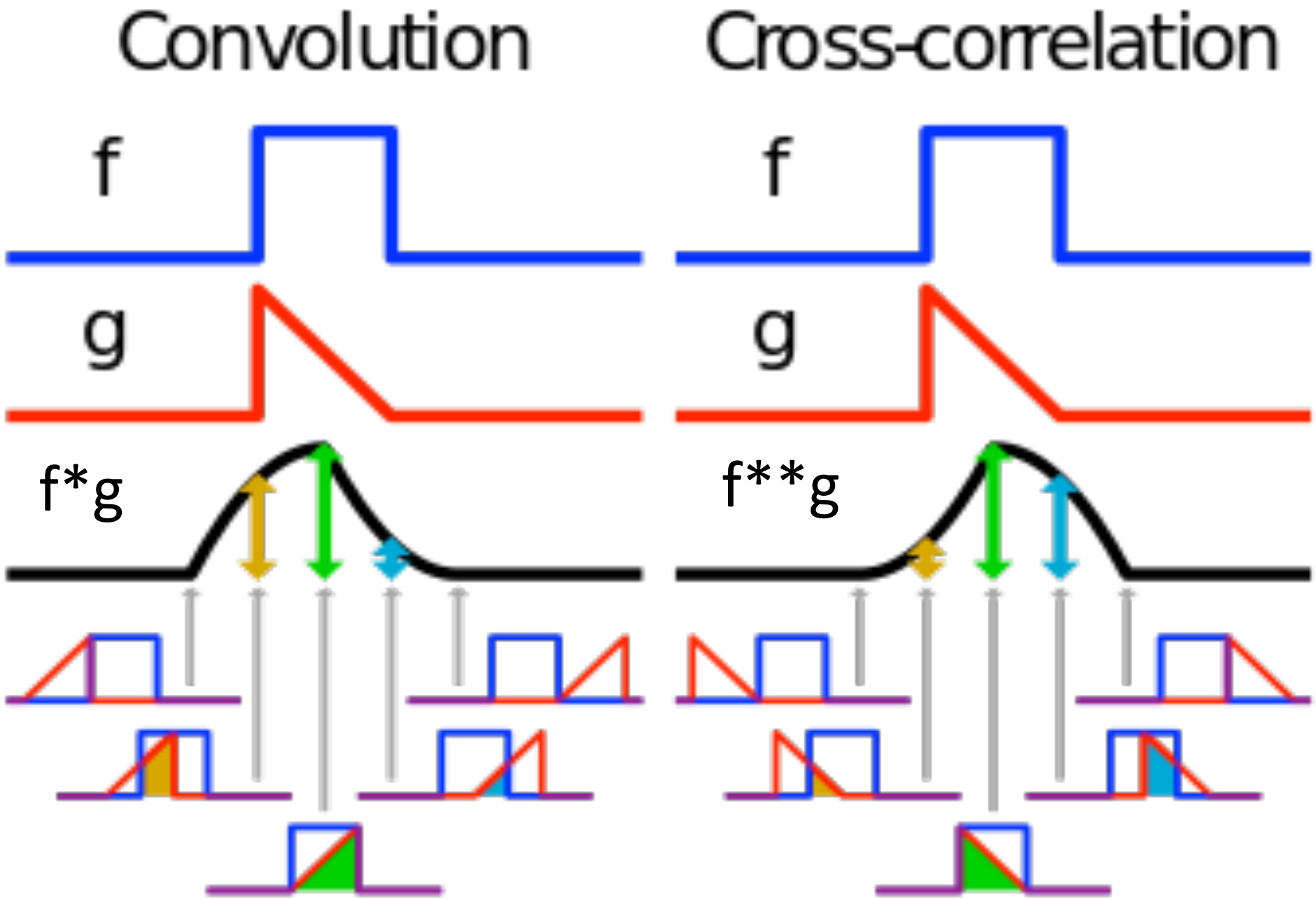


Courtesy of J. Fessler

Courtesy of J. Fessler

(Cross) correlation – example




Convolution
Cross-correlation
f
g
f*g
f
g
f**g

Cross Correlation Application: Vision system for TV remote control

- uses template matching

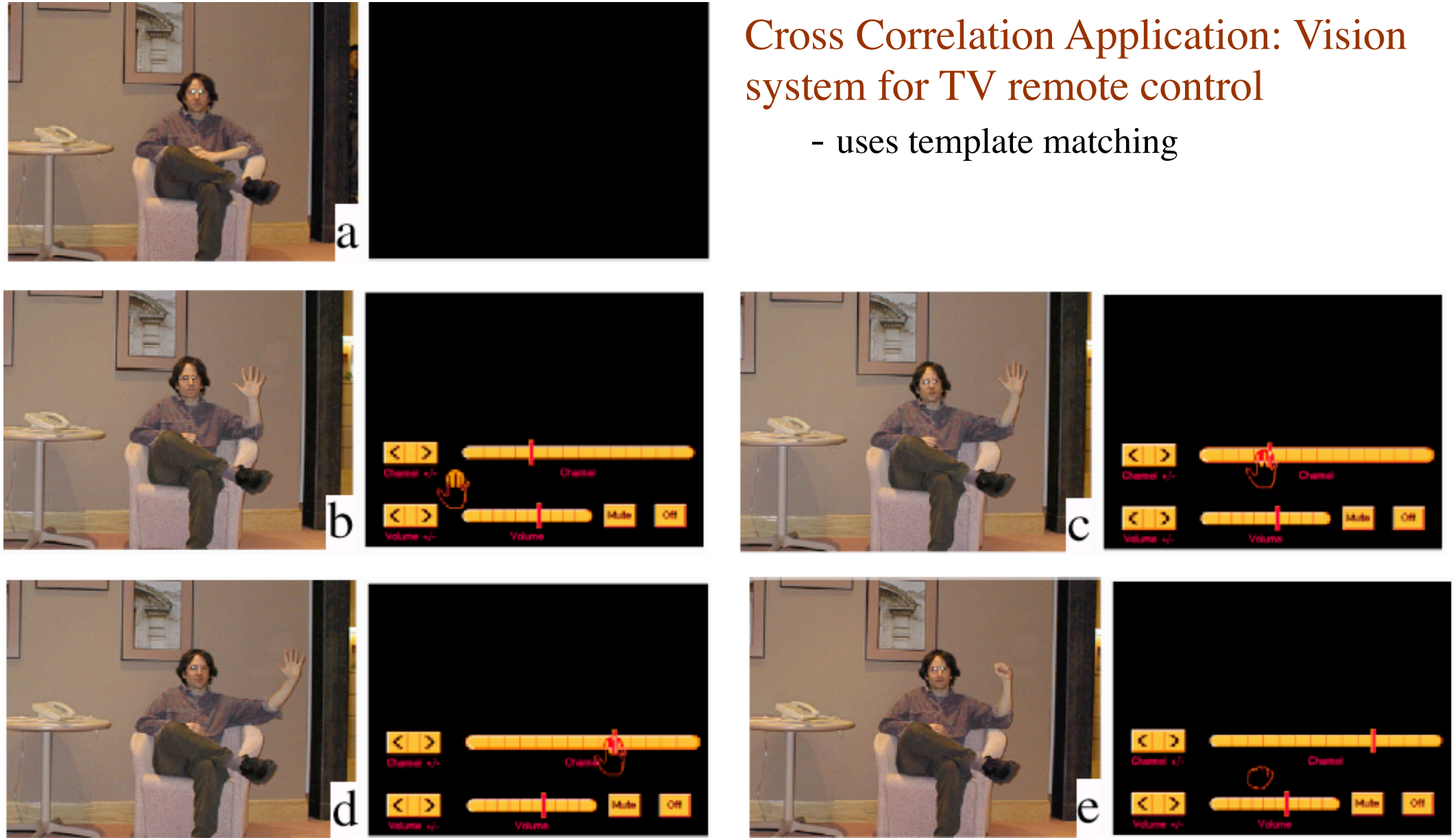


Figure from "Computer Vision for Interactive Computer Graphics," W.Freeman et al, IEEE Computer Graphics and Applications, 1998 copyright 1998, IEEE

Properties

- 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! $h_1 ** h_2 = h_2 ** h_1$

Properties

- 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

- When is correlation equivalent to convolution?
- In other words, when is f**g = f*g?

Convolution vs. (Cross) Correlation

- A **<u>convolution</u>** is an integral that expresses the amount of overlap of one function as it is shifted over another function.
  - convolution is a filtering operation

- **<u>Correlation</u>** 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?

- Image sampling and quantization
- Image histograms
- Images as functions
- Linear systems (filters)
- Convolution and correlation