Lecture 3:
Pixels and Filters

Harbin Institute of Technology（Shenzhen）

Jingyong Su

sujingyong@hit.edu.cn

What we will learn today

- Images as functions
- Linear systems (filters)
- Convolution and correlation

Images as functions

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

- Represented as a matrix of integer values

$j \rightarrow$ (columns), $i \downarrow$ (rows)

Pixel

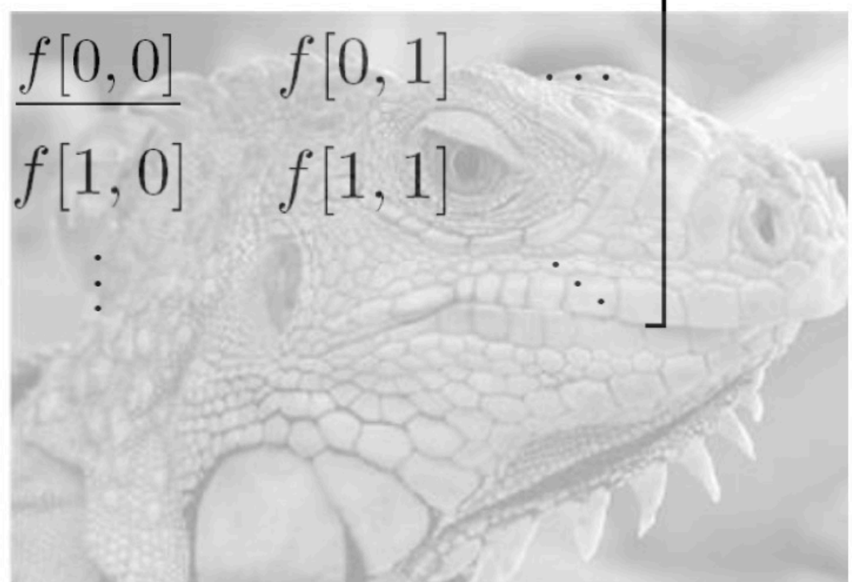
62	79	23	119	120	105	4	0
10	10	9	62	12	78	34	0
10	58	197	46	46	0	0	48
176	135	5	188	191	68	0	49
2	1	1	29	26	37	0	77
0	89	144	147	187	102	62	208
255	252	0	166	123	62	0	31
166	63	127	17	1	0	99	30

Images as functions

Cartesian coordinates

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

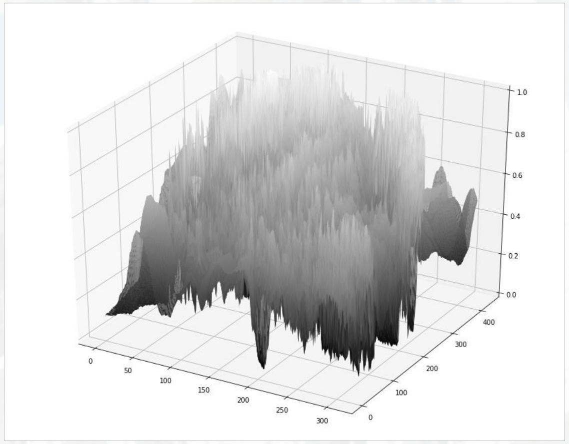
Notation for discrete functions



Images as functions

- An image as a function f from R^2 to R^N
  - $f(x, y)$ 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^N
  - $f(x,y)$ 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: $f(x,y) = \begin{bmatrix} r(x,y) \\ g(x,y) \\ b(x,y) \end{bmatrix}$

What we will learn today

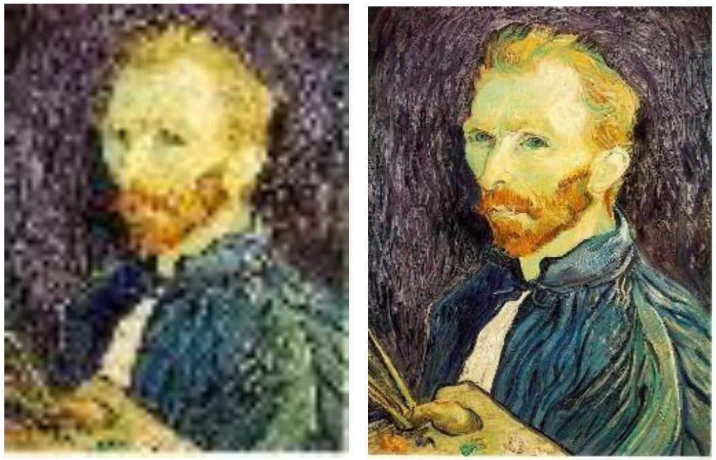
- Images as functions
- Linear systems (filters)
- Convolution and correlation

Applications of Linear Systems and Filters

De-noising

Salt and pepper noise

Super-resolution



In-painting

Systems and Filters

Filtering:
Form a new image whose pixel values are transformed from original pixel values

Goals:
Extract useful information from images
- Features (edges, corners, blobs…)

Modify or enhance image properties
- Super-resolution; in-painting; de-noising

Systems and Filters

S is the system operator, defined as a mapping/assignment that transforms the input f into the output g.

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

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

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

Filter example #1: Moving Average

2D moving average over a 3×3 neighborhood window

Filter example #1: Moving Average

2D moving average over a 3×3 neighborhood window

Sum over rows

Sum over columns

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

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

Filter "kernel", "mask"

h

$$\frac{1}{9} \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

Filter example #1: Moving Average

In summary:

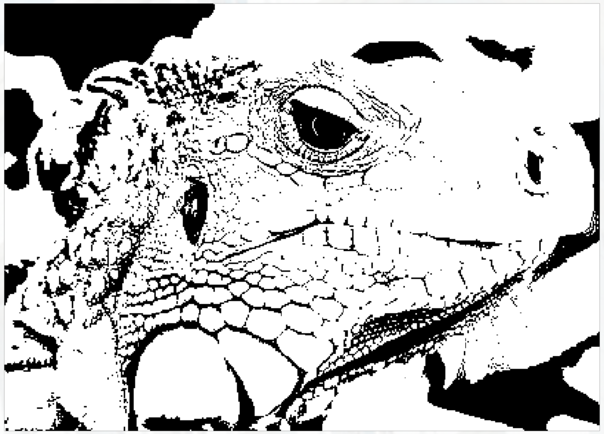
- This filter “transforms” each pixel value into the average value of its neighborhood

- Achieve smoothing effect (remove sharp)

Filter example #2: Image Segmentation

Image segmentation based on a simple threshold:

$$g[n,m] = \begin{cases} 1, & f[n,m] > 100 \\ 0, & \text{otherwise.} \end{cases}$$



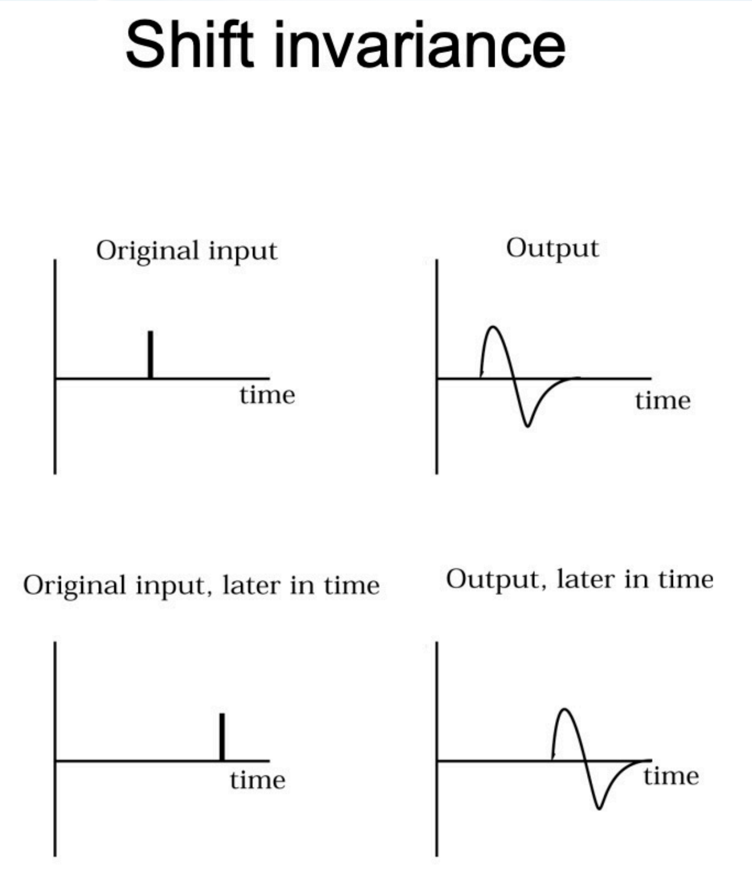
Systems can be classified based on their properties

Amplitude properties

- Linearity
- Stability
- Invertibility

Spatial properties

- Causality
- Separability
- Memory
- Shift invariance
- Rotation invariance



Linear Shift Invariant (LSI) systems

Linearity

$$S\{\alpha f_1[n,m] + \beta f_2[n,m]\} = \alpha S\{f_1[n,m]\} + \beta S\{f_2[n,m]\}$$

Shift invariance

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

For a system to be shift-invariant (or time-invariant) means that a time-shifted version of the input yields a time-shifted version of the output:

$$y(t) = \mathrm{T}[x(t)]$$

$$y(t - s) = \mathrm{T}[x(t - s)]$$

The response $y(t - s)$ is identical to the response $y(t)$, except that it is shifted in time.

Linear Shift Invariant (LSI) systems

- An LSI system is completely specified by its impulse response *h*.

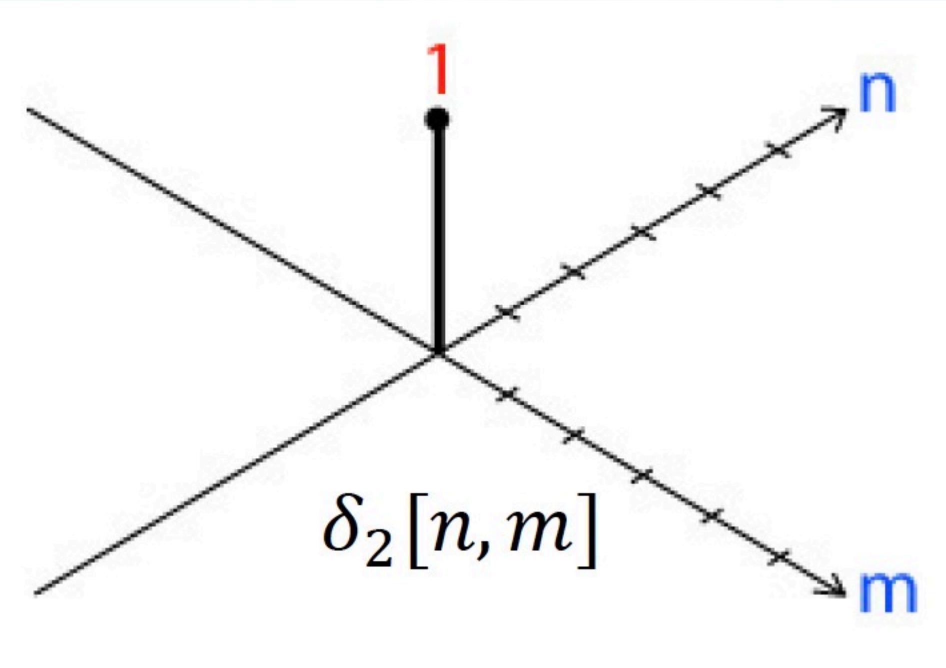
$$\delta_2[n,m] \rightarrow \boxed{\mathcal{S}} \rightarrow h[n,m]$$

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

- By passing an impulse function into an LSI system, we get its impulse response.

2D Impulse Function

- 1 at [0,0]
- 0 everywhere else



Impulse response to the moving average filter

- Recall the expression for our 3x3 moving average filter:

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

- We can use it to obtain an expression for the impulse response

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

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

Impulse response of the 3 x 3 moving average filter

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

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

h

$$\frac{1}{9}\begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline\end{array}$$

Linear Shift Invariant (LSI) systems

- An LSI system is completely specified by its impulse response *h*.
  - For any input *f* , we can compute the output *g* in terms of the impulse response *h*.

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

  - In the following, we'll derive an expression for *g* in terms of *h.*
  - *We know the LSI system satisfies the linearity property and the shift invariance property. We also know h:*

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

Key idea: write down *f* as a sum of impulses

Let’s say our input is a 3 x 3 image:

$$\begin{bmatrix} f[0,0] & f[0,1] & f[1,1] \\ f[1,0] & f[1,1] & f[1,2] \\ f[2,0] & f[2,1] & f[2,2] \end{bmatrix} = \begin{bmatrix} f[0,0] & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & f[0,1] & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} + \ldots + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & f[2,2] \end{bmatrix}$$

$$= f[0,0]* \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} + f[0,1]* \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} + \ldots + f[2,2]* \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= f[0,0] \cdot \delta_2[n,m] + f[0,1] \cdot \delta_2[n,m-1] + \ldots + f[2,2] \cdot \delta[n-2, n-2]$$

Key idea: write down *f* as a sum of impulses

- More generally:

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

- We can now use linearity to see what the output is:

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

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

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

Key idea: write down *f* as a sum of impulses

- We have

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

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

- Recall, by the shift invariance property that:

$$S\{\delta_2[n-k, m-l]\} = h[n-k, m-l]$$

Key idea: write down *f* as a sum of impulses

- We have

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

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

- Which means,

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

Linear Shift Invariant (LSI) systems

- An LSI system is completely specified by its impulse response *h*.
  - For any input *f* , we can compute the output *g* in terms of the impulse response *h*.

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

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

Discrete convolution

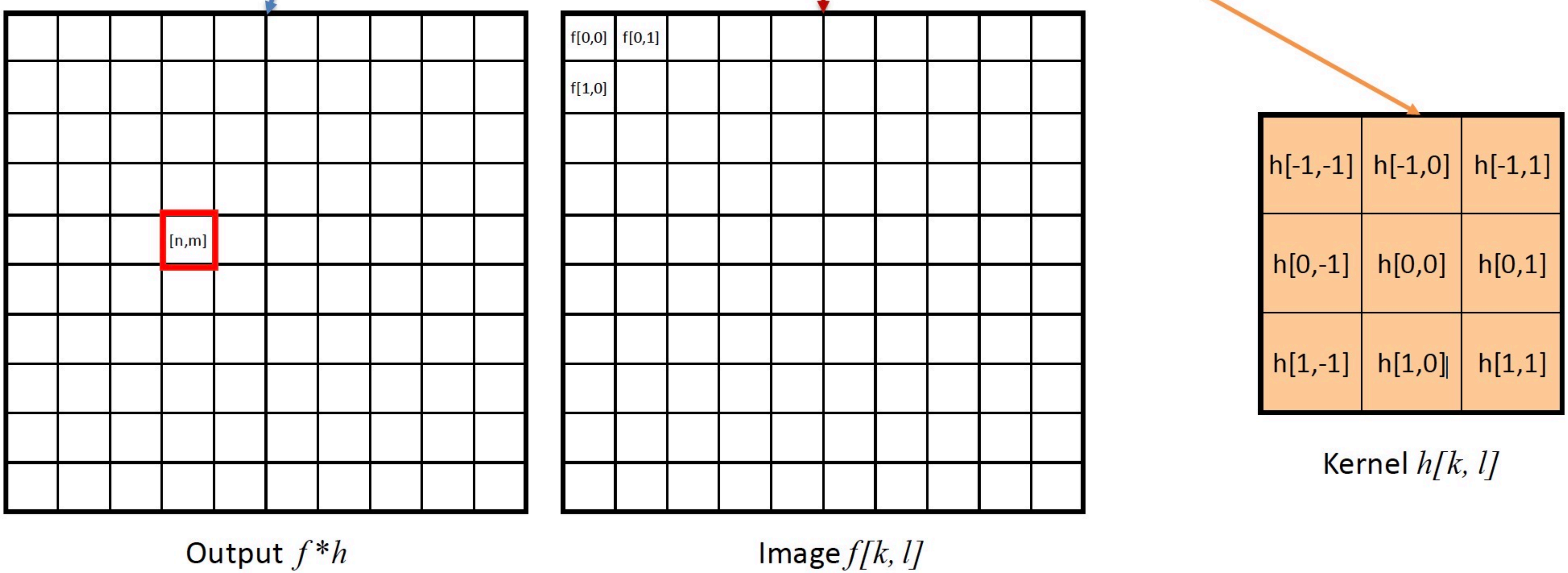
$$\boxed{f[n,m] * h[n,m] = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} f[k,l] \cdot h[n-k,m-l]}$$

What we will learn today

- Images as functions
- Linear systems (filters)
- Convolution and correlation

2D Discrete Convolution

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



2D Discrete Convolution

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

h[-1,-1]	h[-1,0]	h[-1,1]
h[0,-1]	h[0,0]	h[0,1]
h[1,-1]	h[1,0]	h[1,1]

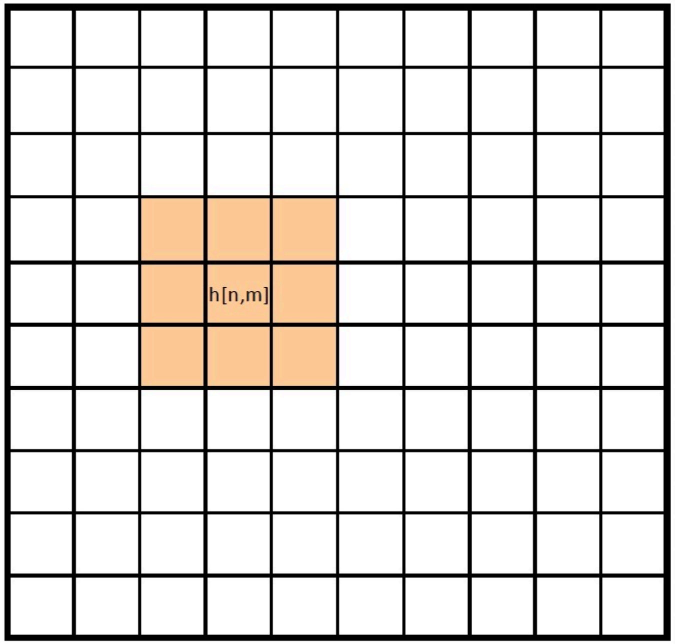
Kernel *h[k, l]*

Fold

h[1,1]	h[1,0]	h[1,-1]
h[0,1]	h[0,0]	h[0,-1]
h[-1, 1]	h[-1,0]	h[-1,-1]

Kernel *h[-k, -l]*

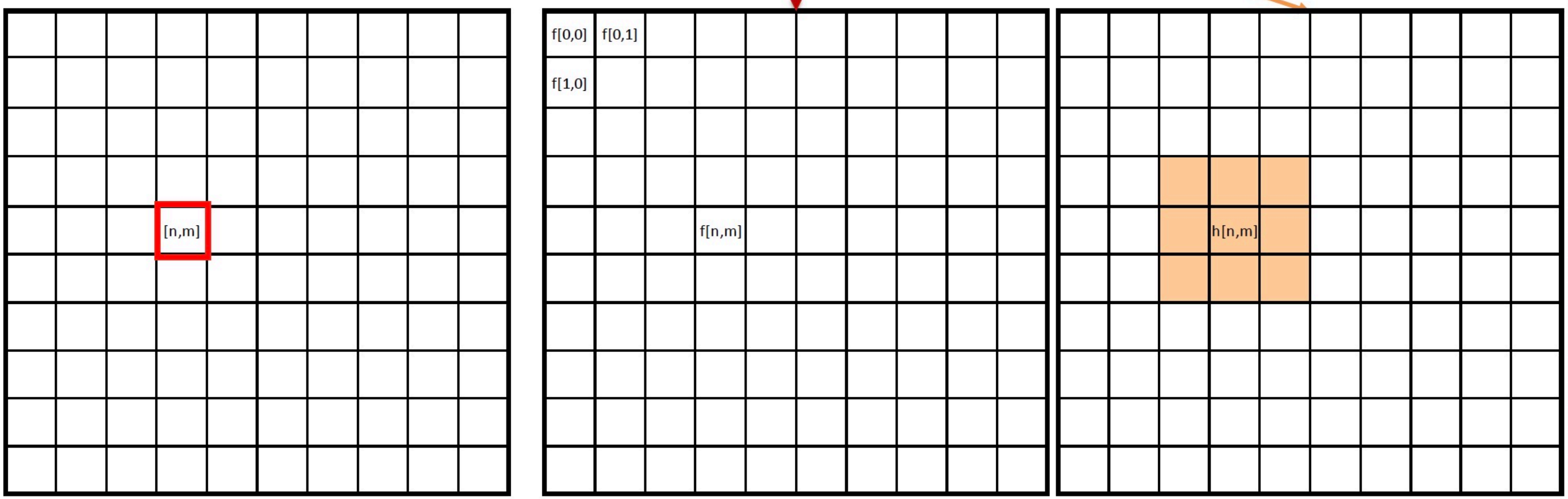
Shift



Kernel *h[n-k, m-l]*

2D Discrete Convolution

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



Output $f*h$ Image $f[k, l]$ Kernel $h[n\text{-}k, m\text{-}l]$

2D Discrete Convolution

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

Algorithm:

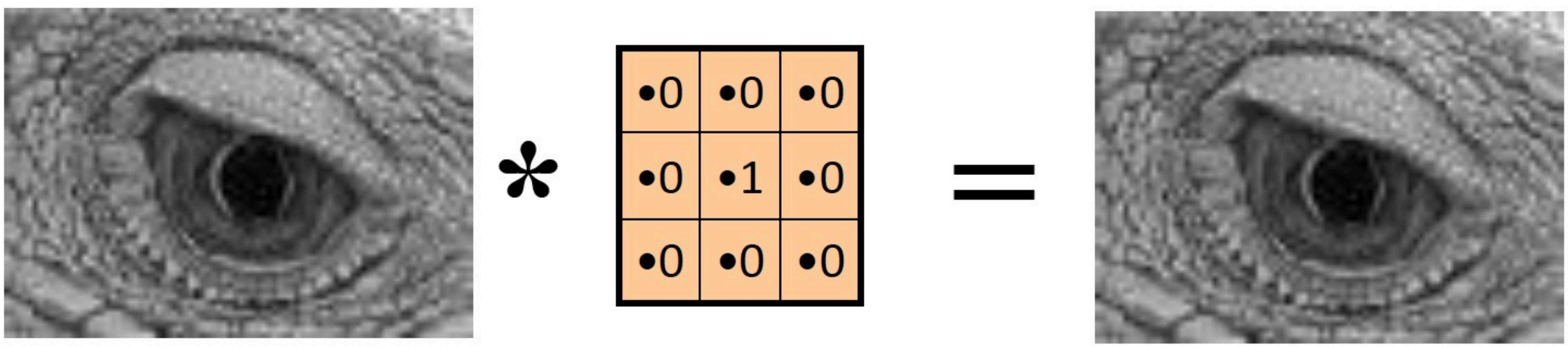
- Fold $h[k, l]$ about origin to form $h[-k, -l]$
- Shift the folded results by n, m to form $h[n - k, m - l]$
- Multiply $h[n - k, m - l]$ by $f[k, l]$
- Sum over all k, l
- Repeat for every n, m

Convolution in 2D -- examples

Original

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

Convolution in 2D -- examples



Original

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

$$* \left(\begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \right) = ?$$

(Note that filter sums to 1)

Convolution in 2D -- examples

Original

$$* \left(\begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \right) = ?$$

(Note that filter sums to 1)

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} + \overbrace{\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}}^{\text{“details of the image”}}$$

What does blurring take away?

−

=

- Let's add it back:

+

=

What we will learn today

- Images as functions
- Linear systems (filters)
- Convolution and correlation

(Cross) Correlation – symbol: **

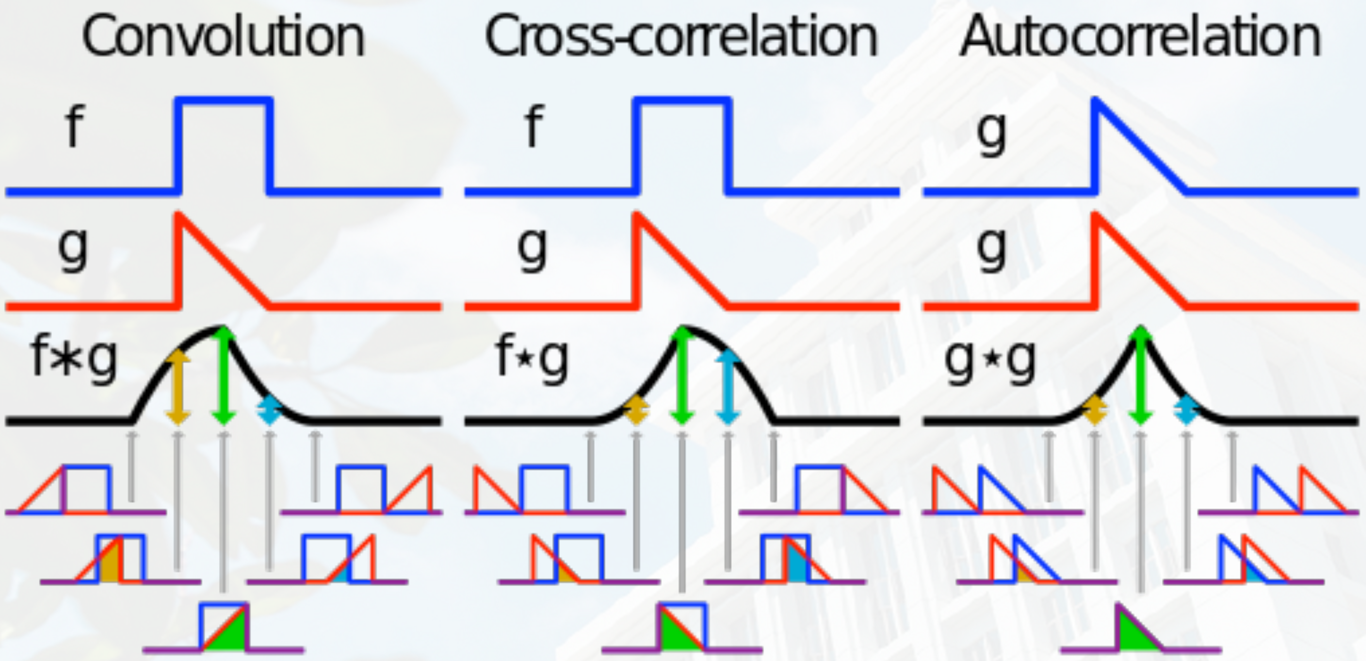
Cross correlation of two 2D signals *f*[*n,m*] and *h*[*n,m*]

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

- Equivalent to a convolution without the flip

- Use it to measure ‘similarity’ between *f* and *h*

Convolution vs (Cross) Correlation



Convolution vs (Cross) Correlation

- **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

- **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

- Images as functions
- Linear systems (filters)
- Convolution and correlation