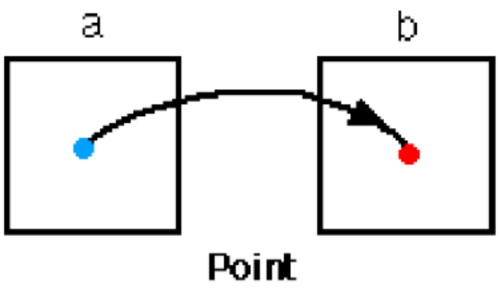
Digital Image Processing (CSE/ECE 478)

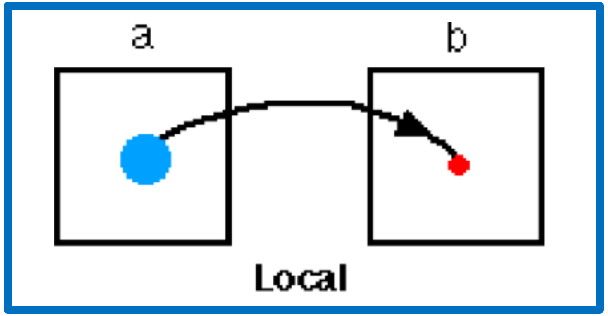
Lecture 6 : Spatial Filters (Part 2)

Ravi Kiran

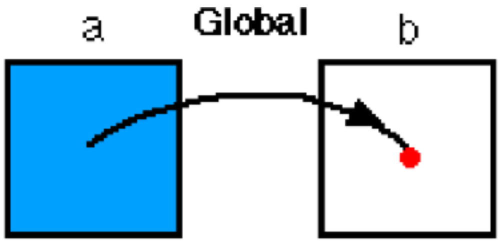
Spatial Domain Processing

- Manipulating Pixels Directly in Spatial Domain
- Point to Point



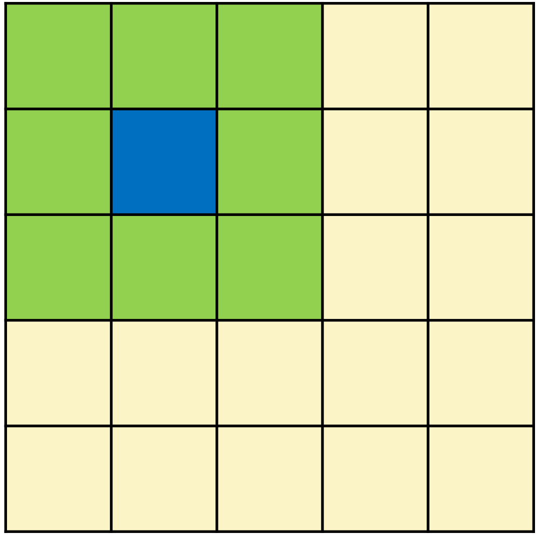


- **Neighborhood to Point**
- Global Attribute to Point



Smoothing as Averaging

I



H → Mask

Kernel

Filter

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

$$I'(u,v) \leftarrow \frac{1}{9} \cdot \sum_{j=-1}^{1} \sum_{i=-1}^{1} I(u+i, v+j)$$

$$I'(u,v) \leftarrow \sum_{j=-1}^{1} \sum_{i=-1}^{1} I(u+i, v+j)$$

- $H(i,j)$

Effect of Mask Size

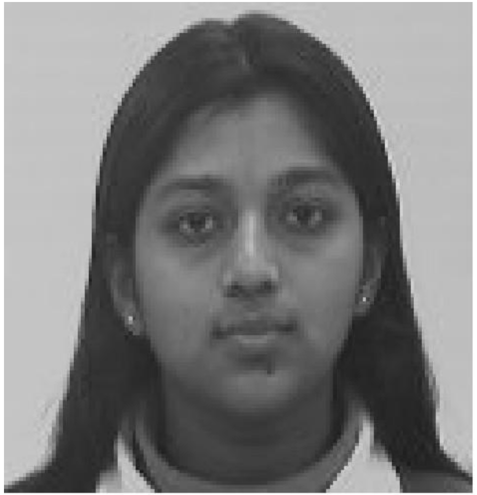
Original Image

[3x3]

[5x5]

[7x7]

Effect of Repeated Smoothing



Before

After

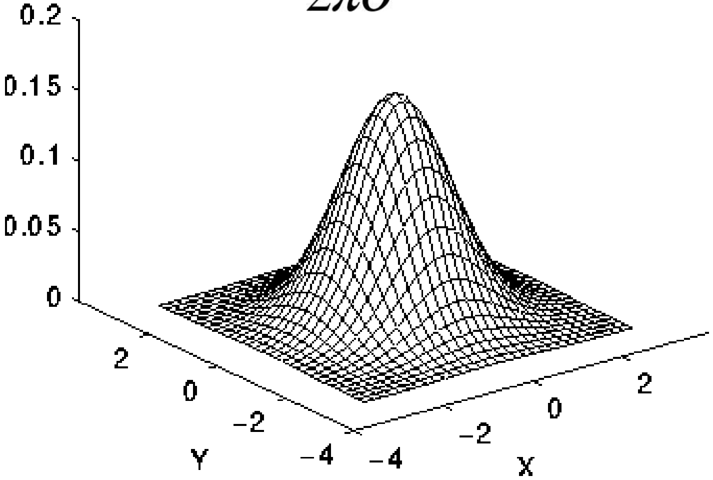
After repeated averaging

NOTE: Can get the effect of larger filters by smoothing repeatedly with smaller filters

Gaussian Smoothing

- Mask weights are samples of a Gaussian Function

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



$\frac{1}{265}$

1	4	6	4	1
4	16	26	16	4
6	26	43	26	6
4	16	26	16	4
1	4	6	4	1

5×5 Gaussian filter, $\sigma=1$

Sharpening Filter

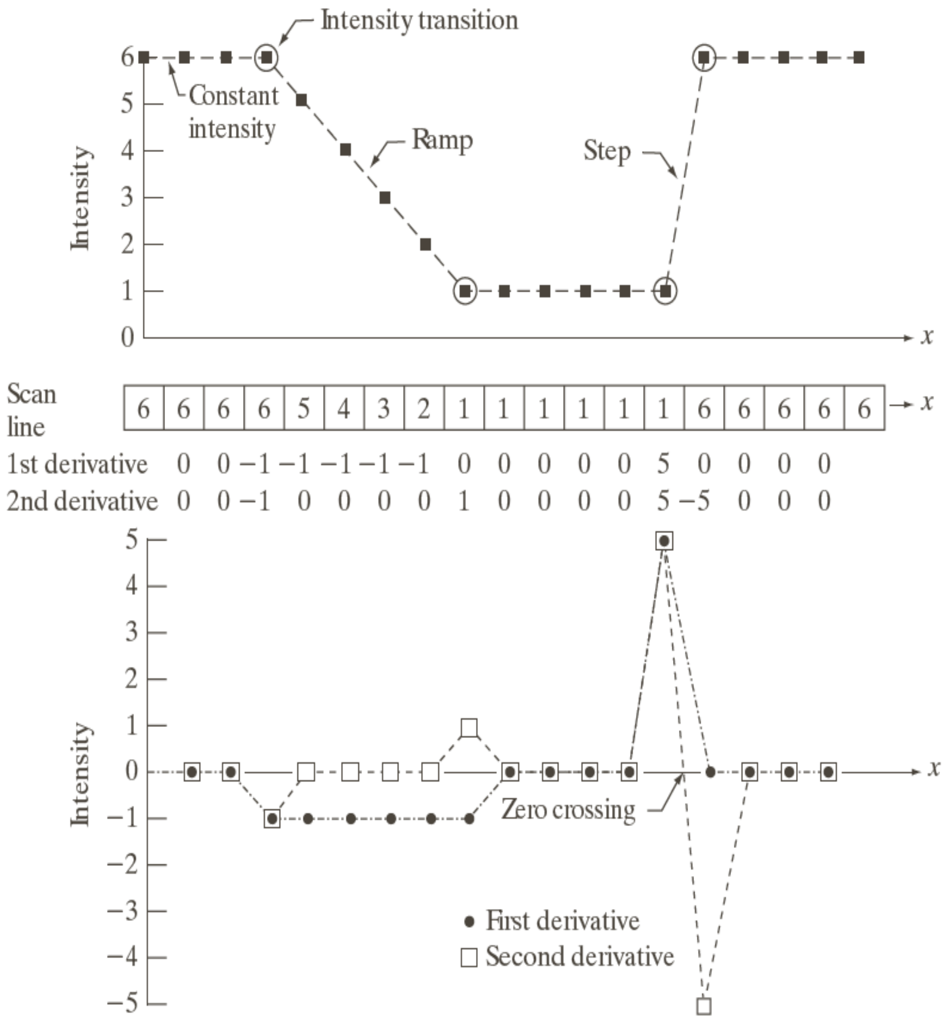
- Objective of sharpening is to highlight fine detail in an image or to enhance detail that has been blurred.
- Smoothing → Averaging → Summation → Integration
- Sharpening → Difference

- First Derivative

$$\frac{\partial f}{\partial x} = f(x+1) - f(x)$$

- Second Derivative

$$\frac{\partial^2 f}{\partial x^2} = f(x+1) + f(x-1) - 2f(x)$$



Laplacian Filter

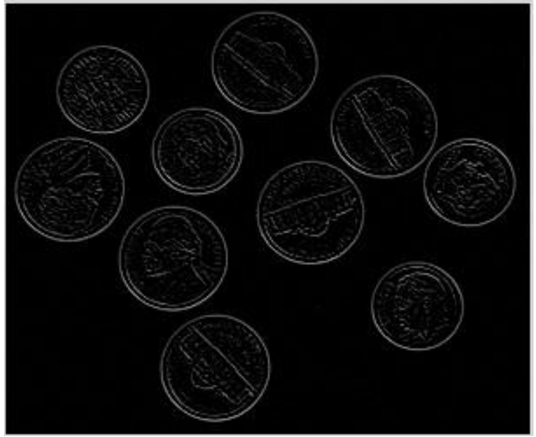
$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial \cancel{x^2}\, y^2}$$

$$\nabla^2 f = f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1) - 4f(x, y)$$

0	1	0
1	-4	1
0	1	0

Laplacian Filters

$\nabla^2 I(u,v)$

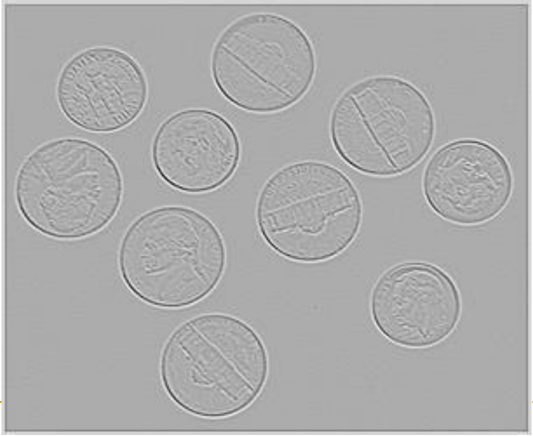


$I(u,v)$

$I'(u,v)$

$\nabla^2 I(u,v) + 128$

(ForVisualization)



Unsharp Masking (and Highboost Filtering)

- **High boost filter**: amplify input image, then subtract a lowpass image

$$Highboost = A\ Original - Lowpass$$

$$= (A - 1)\ Original + Original - Lowpass$$

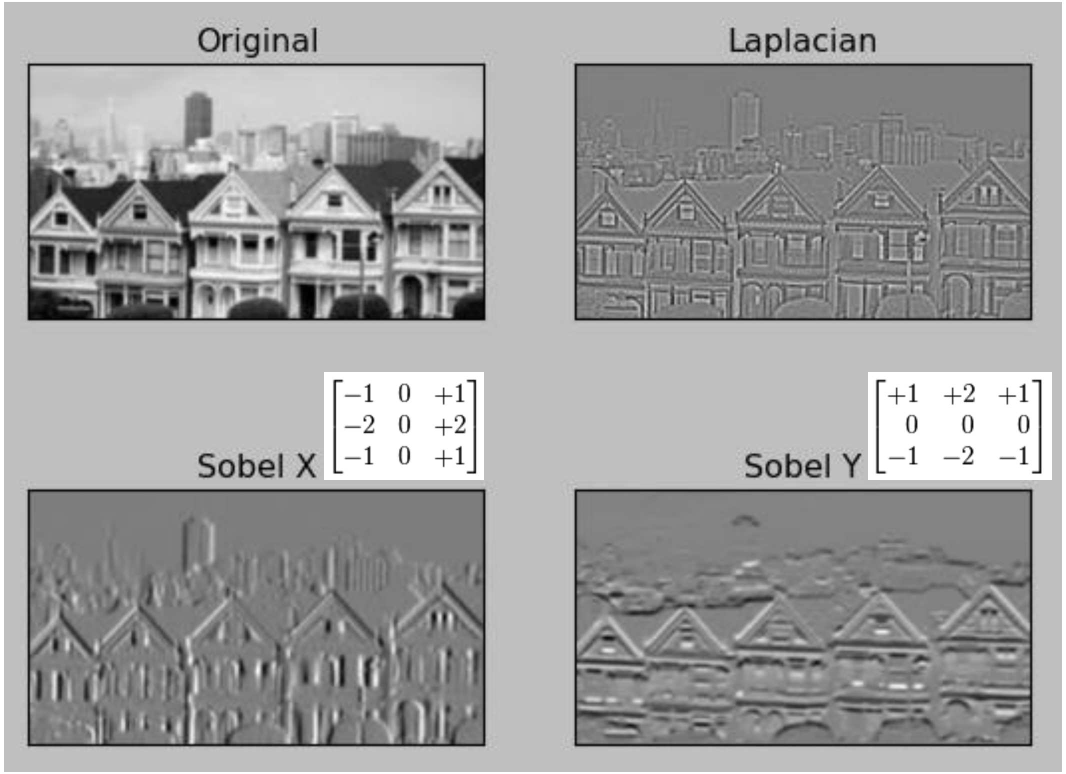
$$= (A - 1)\ Original + Highpass$$

(A-I) 

+ 

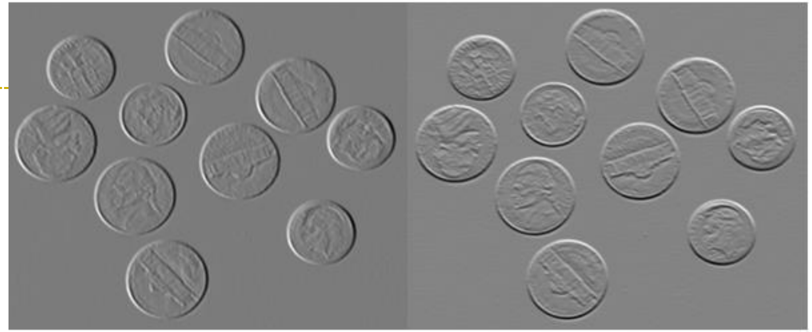
= 

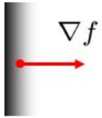
Sobel Edge Masks



0	−1	0
−1	4	−1
0	−1	0

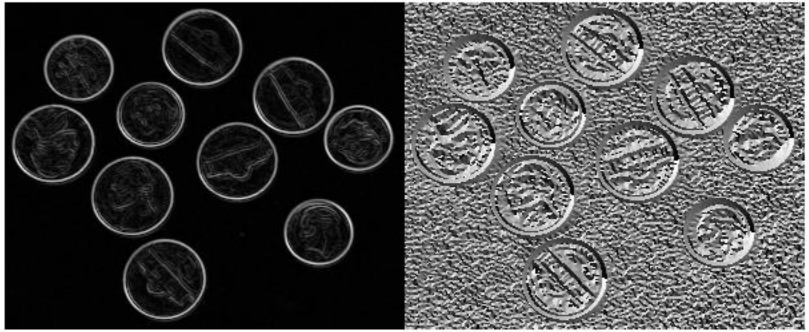
Edge Magnitude and Gradient

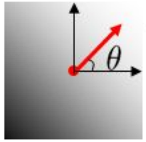




$$\nabla f = \left[\frac{\partial f}{\partial x}, 0\right]$$

$$\nabla f = \left[0, \frac{\partial f}{\partial y}\right]$$



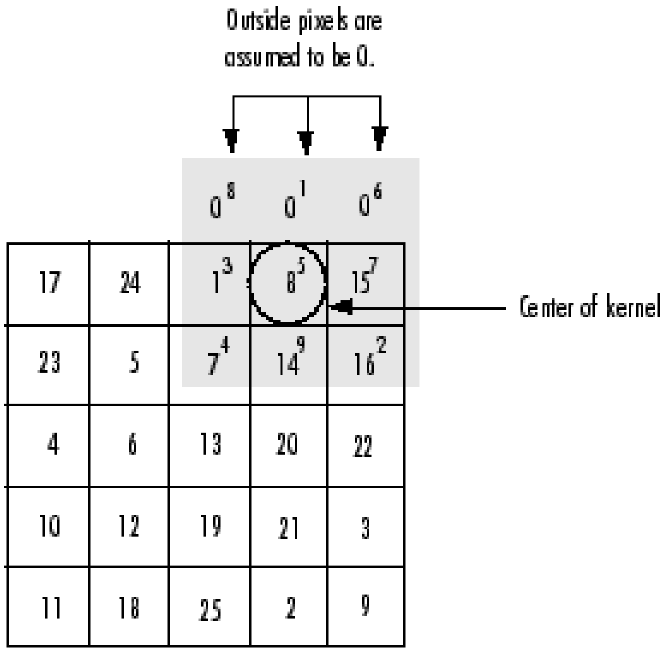


$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

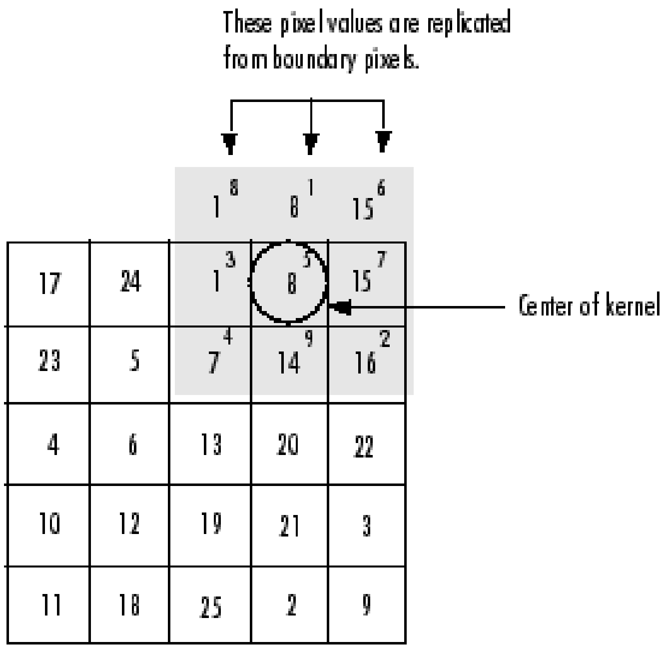
$$\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

$$\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} \Big/ \frac{\partial f}{\partial x}\right)$$

Image Padding



zero



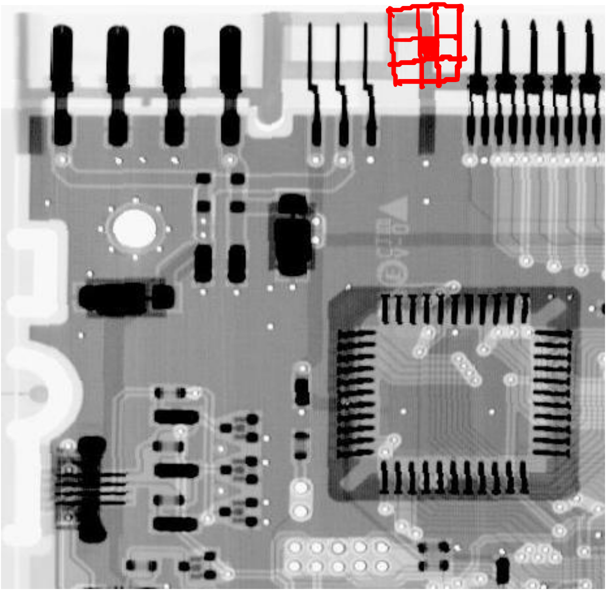
replicate

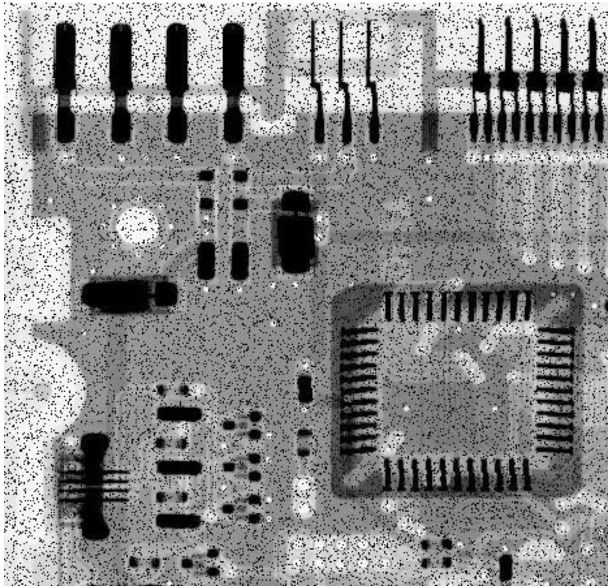
Spatial Domain Filtering - Approaches

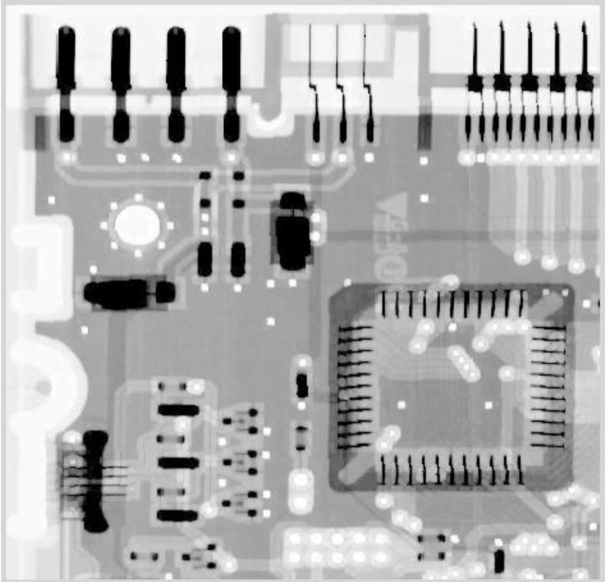
- Linear
- Non-linear

Other Spatial Filters (non linear)

pepper noise

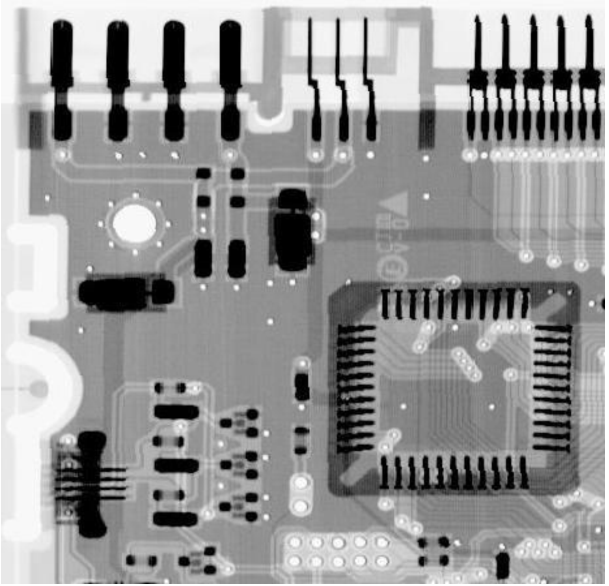




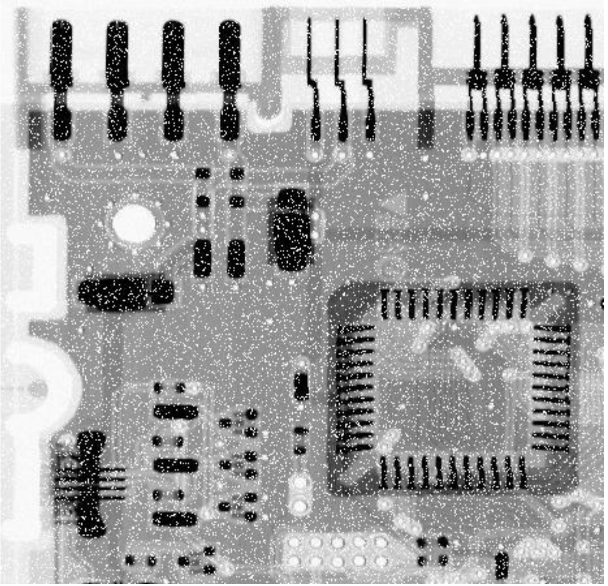


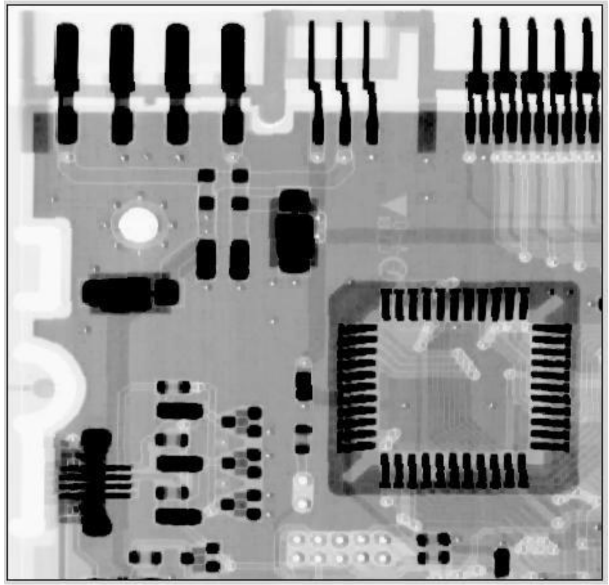
max filter

Other Spatial Filters (non linear)



salt noise

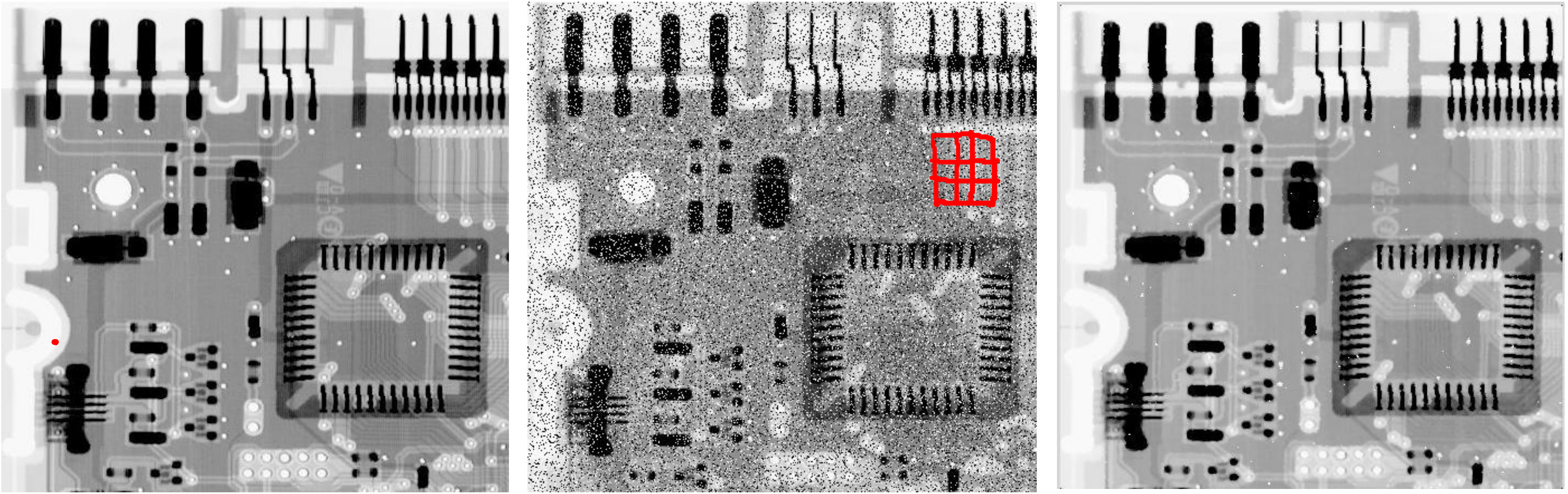




min filter

Other Spatial Filters (median filter – non linear)

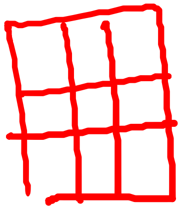
salt & pepper noise



max, min, median → also known as order statistic filters

Other Spatial Filters

- Geometric mean
- Harmonic mean
- Contra harmonic mean
- Mid Point filter
- Alpha trimmed mean filter
- …..



$(i_1 \; i_2 \; \ldots \; i_9)^{1/9}$

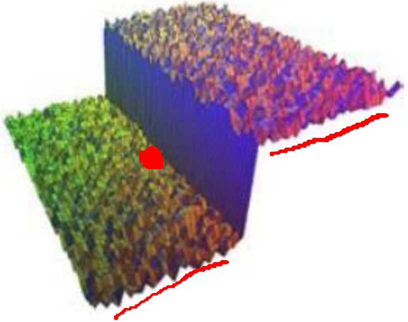
Bilateral Filtering

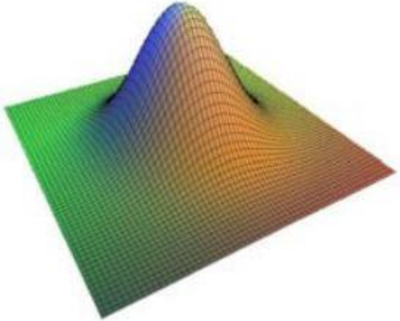
Original image taken from cs.cityu.edu.hk

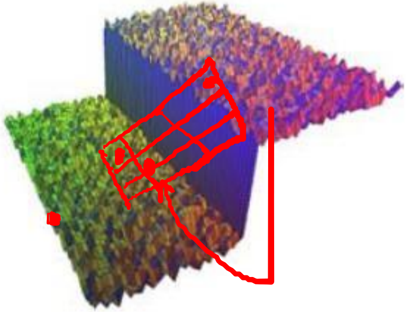
Bilateral Filtering

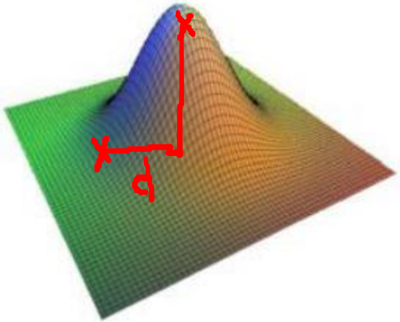
Original image from mfullywoodco.hol.es

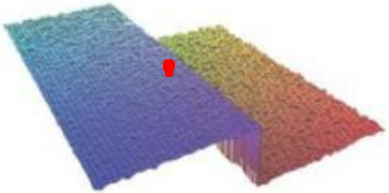
L · G

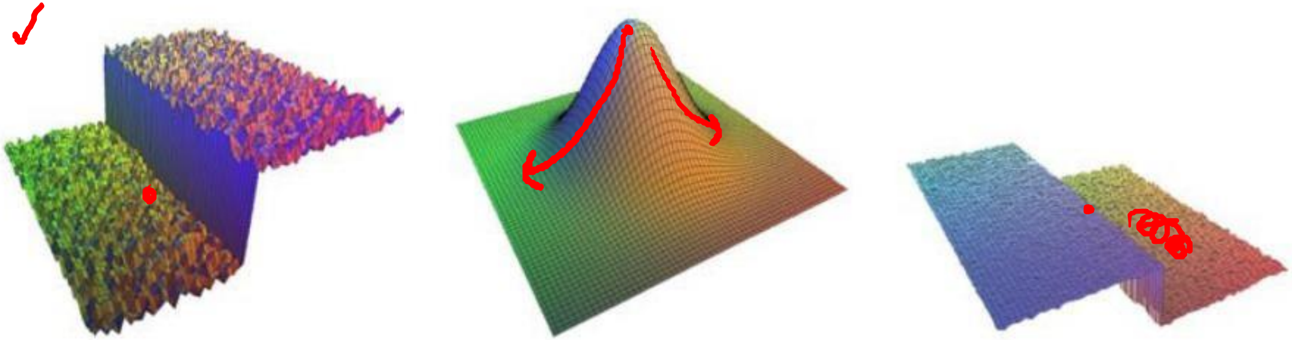




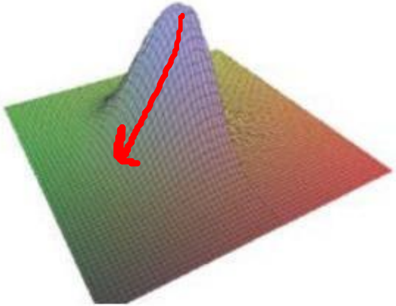


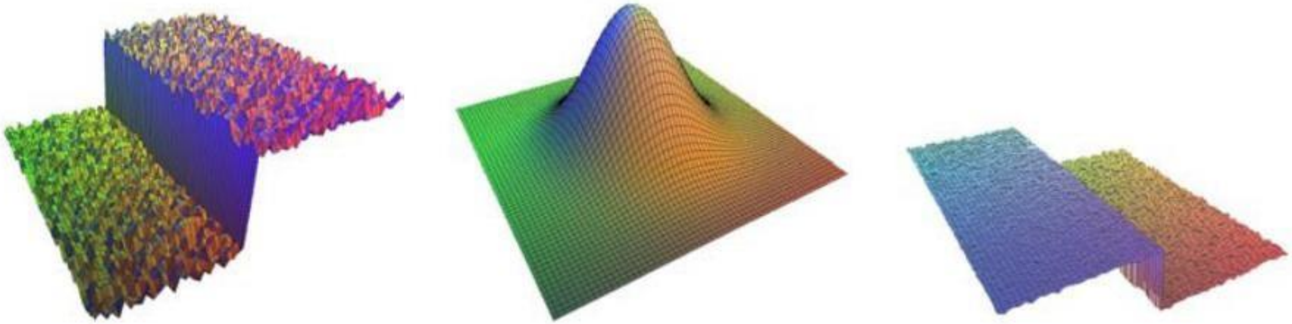

d



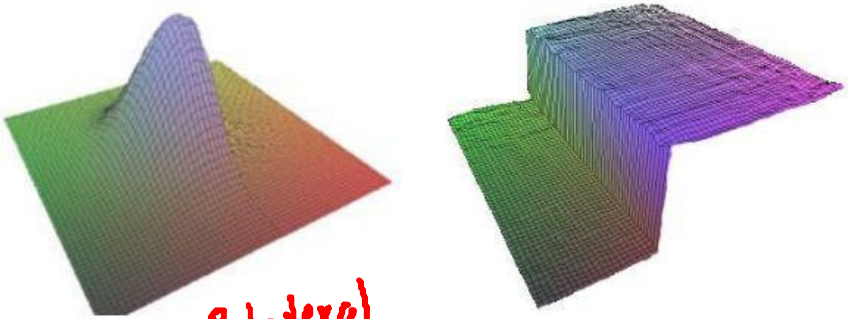


(a) (b) (c)

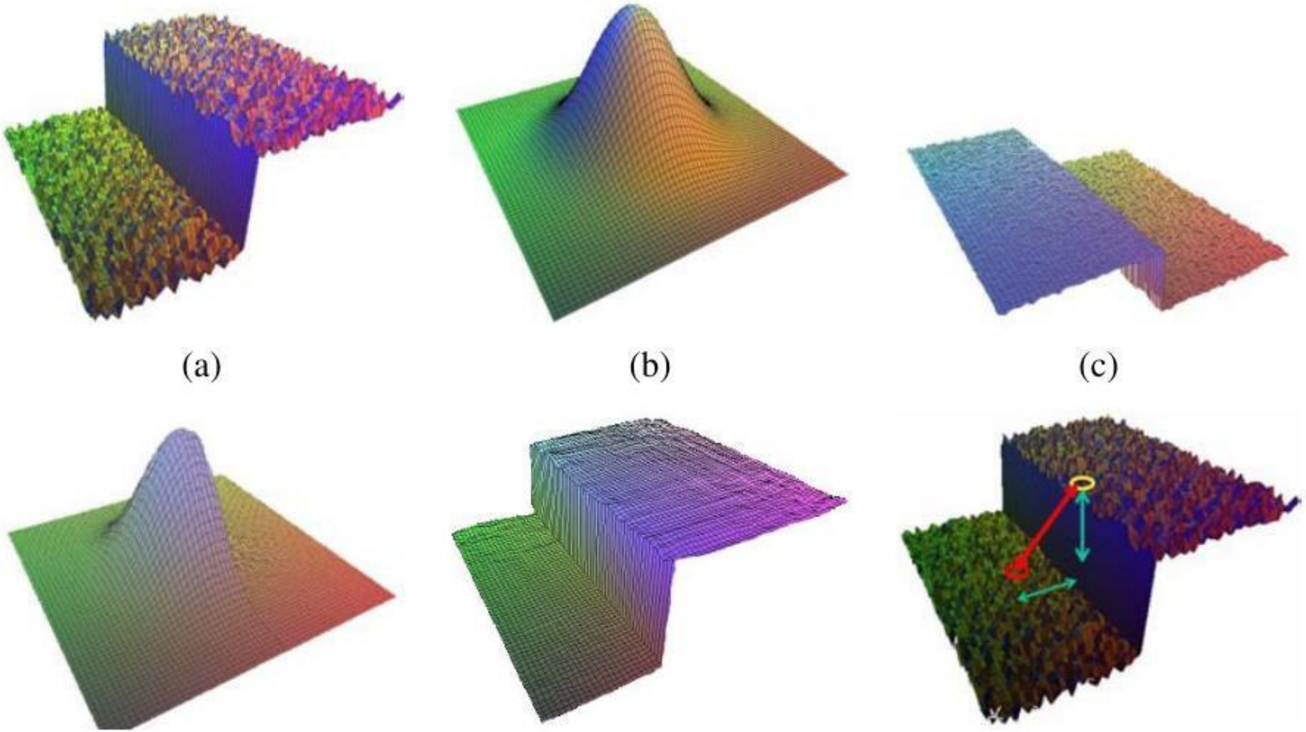




(a) (b) (c)



Bilateral Filter

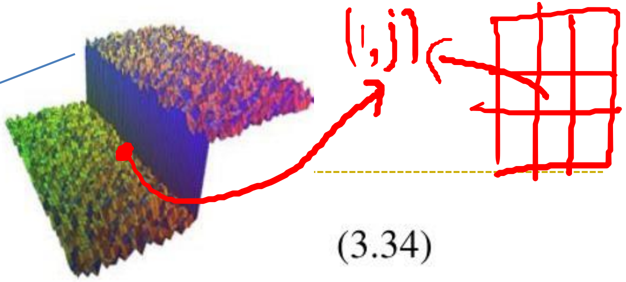


(a) (b) (c)

(d) (e) (f)

Figure 3.20 Bilateral filtering (Durand and Dorsey 2002) © 2002 ACM: (a) noisy step edge input; (b) domain filter (Gaussian); (c) range filter (similarity to center pixel value); (d) bilateral filter; (e) filtered step edge output; (f) 3D distance between pixels.

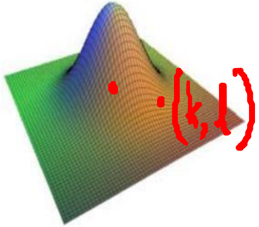
Bilateral Filter



$$g(i,j) = \frac{\sum_{k,l} f(k,l)w(i,j,k,l)}{\sum_{k,l} w(i,j,k,l)}. \tag{3.34}$$

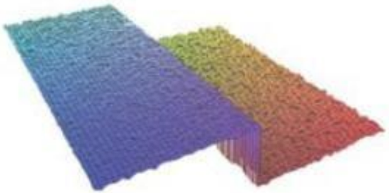
The weighting coefficient $w(i,j,k,l)$ depends on the product of a *domain kernel* (Figure 3.19c),

$$d(i,j,k,l) = \exp\left(-\frac{(i-k)^2 + (j-l)^2}{2\sigma_d^2}\right),$$

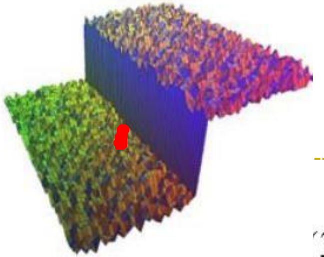


and a data-dependent *range kernel* (Figure 3.19d),

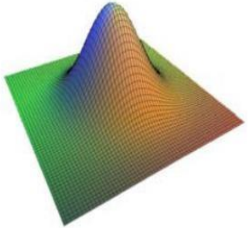
$$r(i,j,k,l) = \exp\left(-\frac{\|f(i,j) - f(k,l)\|^2}{2\sigma_r^2}\right).$$



Bilateral Filter



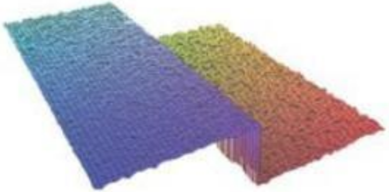
$$g(i,j) = \frac{\sum_{k,l} f(k,l) w(i,j,k,l)}{\sum_{k,l} w(i,j,k,l)}. \tag{3.34}$$



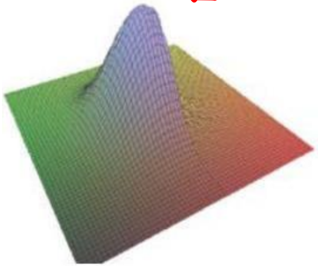
The weighting coefficient $w(i,j,k,l)$ depends on the product of a *domain kernel* (Figure 3.19c),

$$d(i,j,k,l) = \exp\left(-\frac{(i-k)^2 + (j-l)^2}{2\sigma_d^2}\right), \tag{3.35}$$

and a data-dependent *range kernel* (Figure 3.19d),



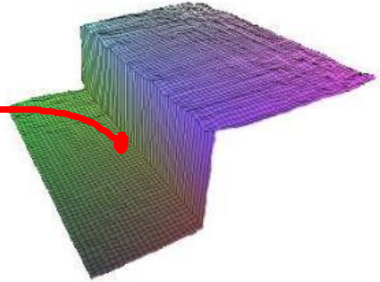
$$r(i,j,k,l) = \exp\left(-\frac{\|f(i,j) - f(k,l)\|^2}{2\sigma_r^2}\right).$$

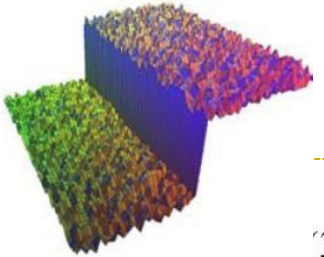


When multiplied together, these yield the data-dependent *bilateral weight function*

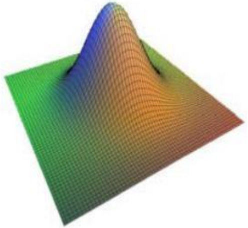
$$w(i,j,k,l) = \exp\left(-\frac{(i-k)^2 + (j-l)^2}{2\sigma_d^2} - \frac{\|f(i,j) - f(k,l)\|^2}{2\sigma_r^2}\right). \tag{3.37}$$

g(i,j)





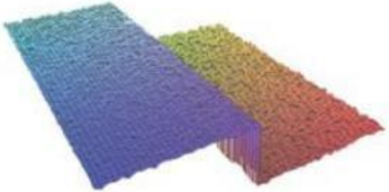
$$g(i,j) = \frac{\sum_{k,l} f(k,l)w(i,j,k,l)}{\sum_{k,l} w(i,j,k,l)}. \tag{3.34}$$



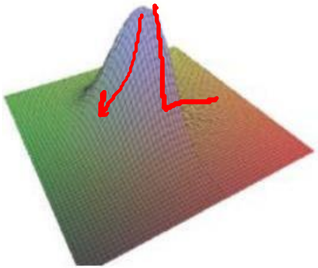
The weighting coefficient $w(i,j,k,l)$ depends on the product of a *domain kernel* (Figure 3.19c),

$$d(i,j,k,l) = \exp\left(-\frac{(i-k)^2+(j-l)^2}{2\sigma_d^2}\right), \tag{3.35}$$

and a data-dependent *range kernel* (Figure 3.19d),



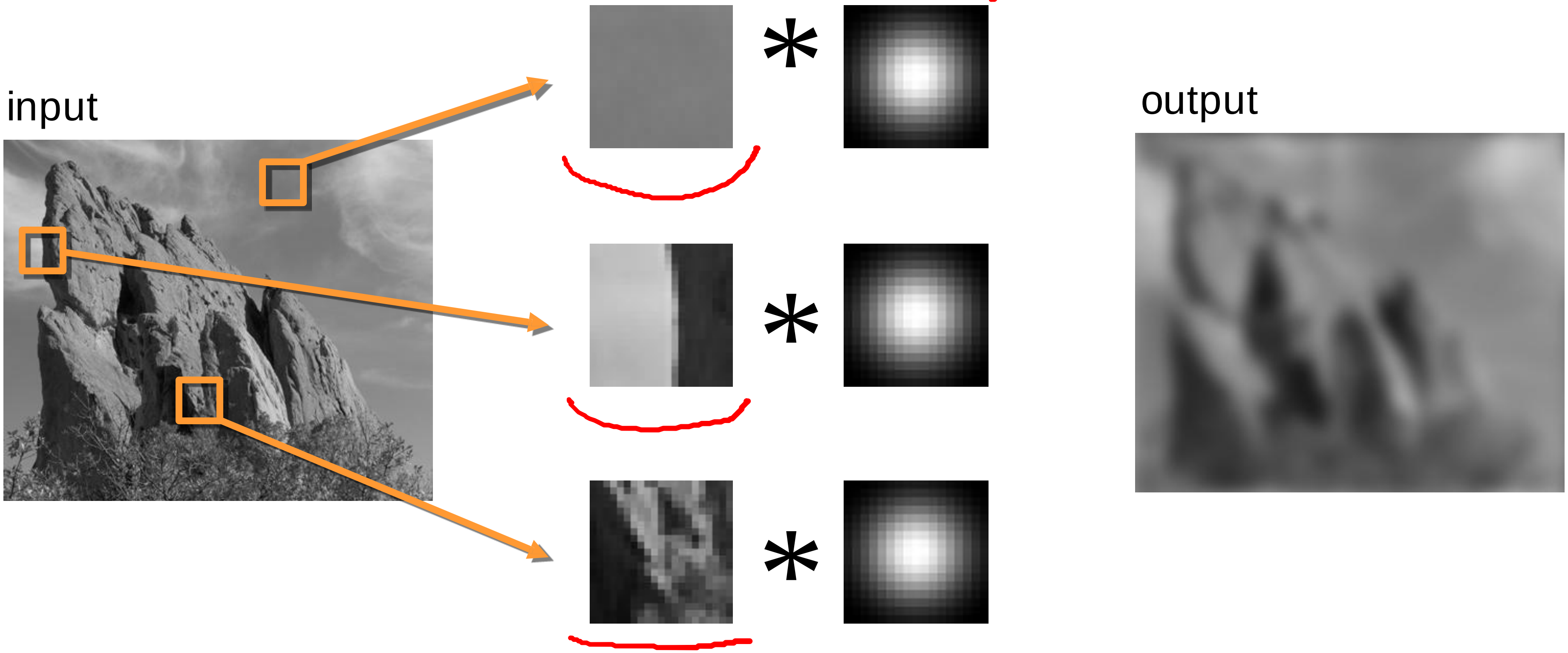
$$r(i,j,k,l) = \exp\left(-\frac{\|f(i,j)-f(k,l)\|^2}{2\sigma_r^2}\right).$$



When multiplied together, these yield the data-dependent *bilateral weight function*

$$w(i,j,k,l) = \exp\left(-\frac{(i-k)^2+(j-l)^2}{2\sigma_d^2} - \frac{\|f(i,j)-f(k,l)\|^2}{2\sigma_r^2}\right). \tag{3.37}$$

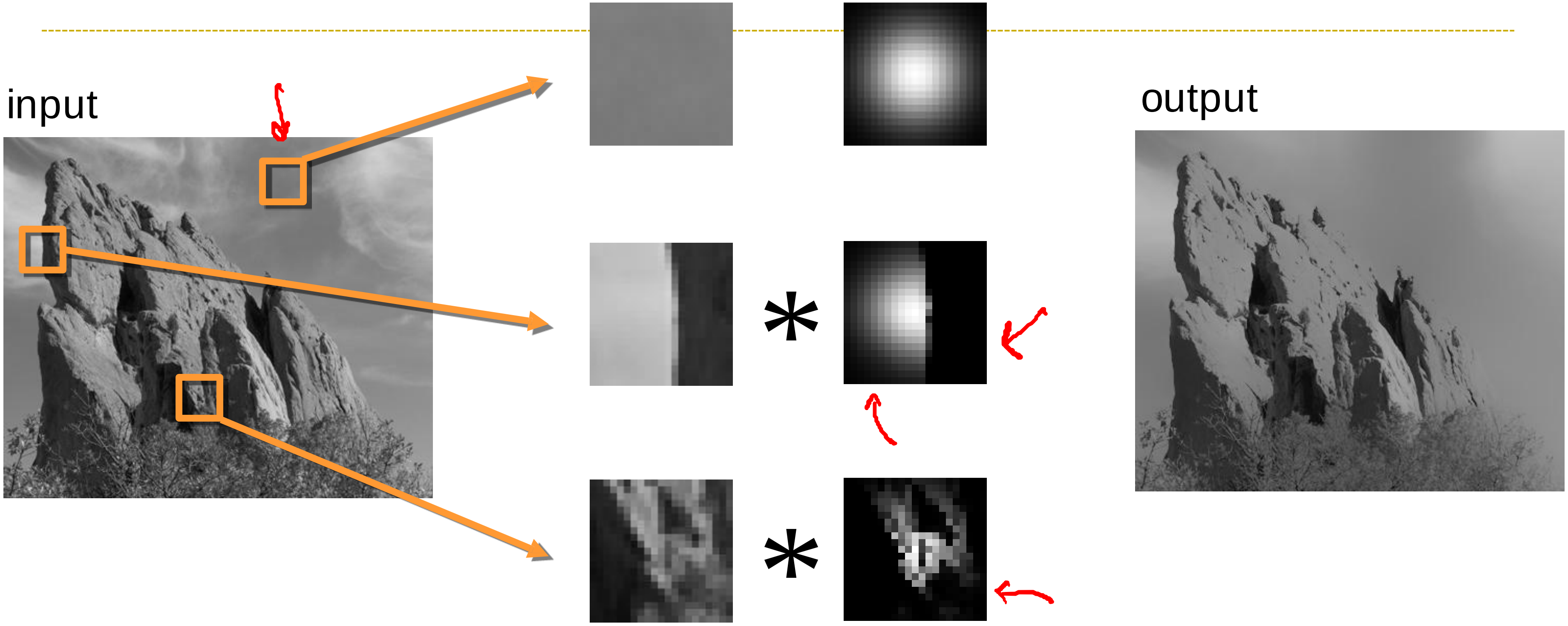
Usual Gaussian Filtering



Same Gaussian kernel everywhere.

Source: Sylvain Paris

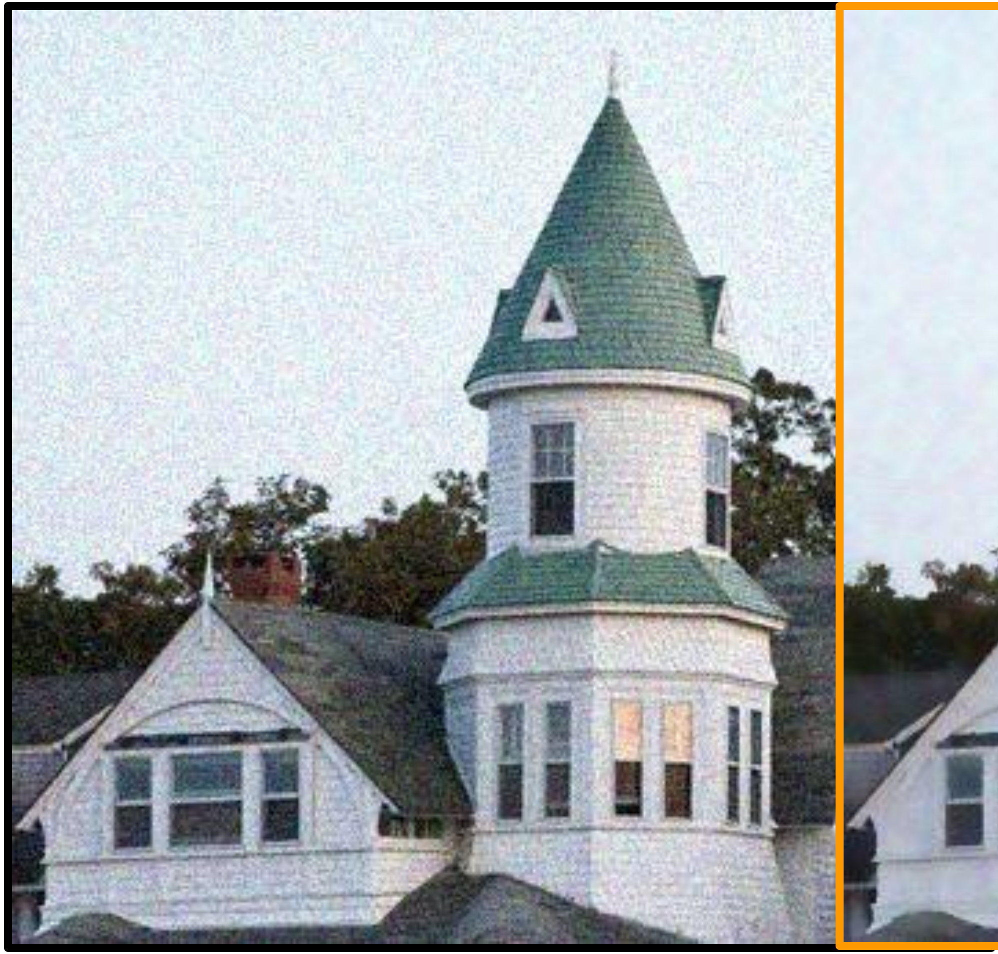
Bilateral Filtering



The kernel shape depends on the image content.

Source: Sylvain Paris

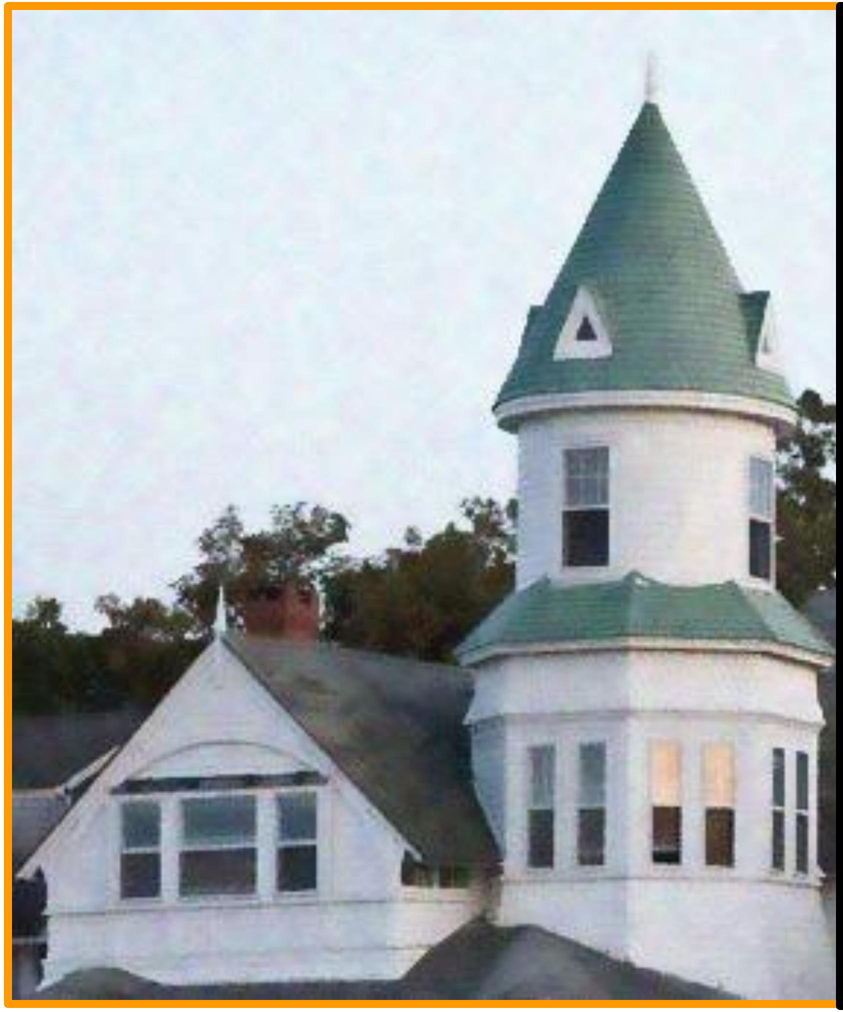
Noisy input



Bilateral filter 7x7 window

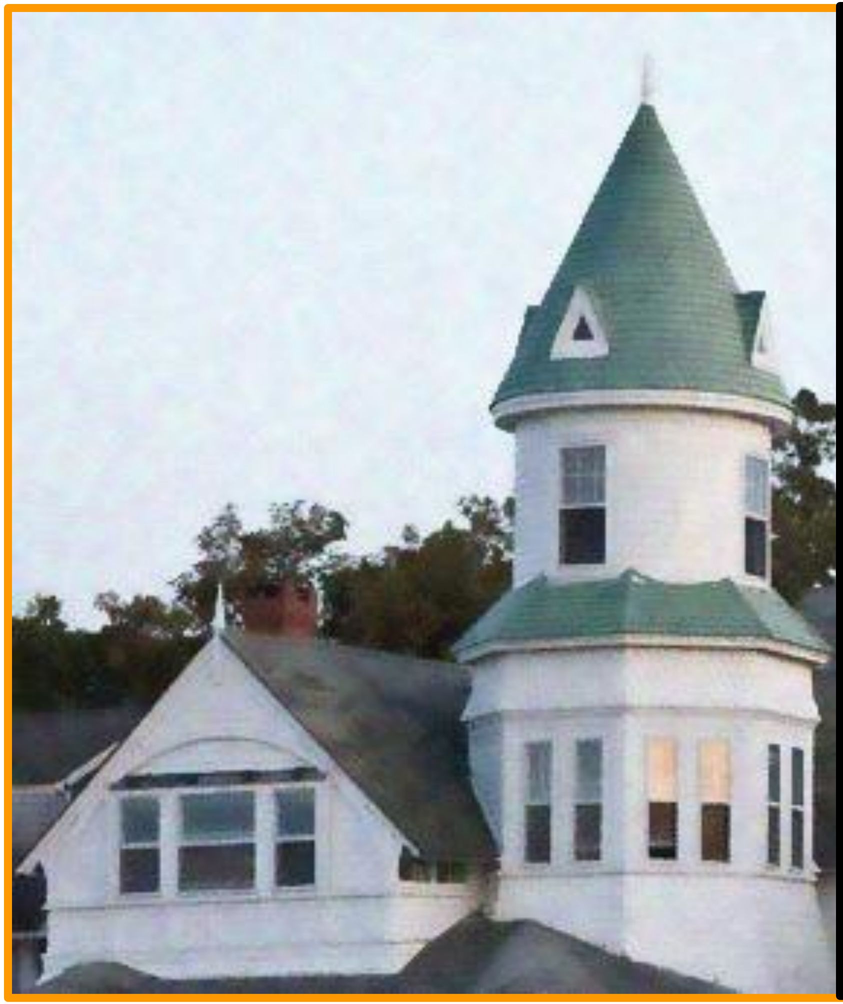
Source: Sylvain Paris

Bilateral filter



Median 3x3

Bilateral filter



Median

Other Important Filters

- Laplacian of Gaussian
  - Noise Suppression

Robert Collins
CSE486

LoG Filter

- First smooth (Gaussian filter),
- Then, find zero-crossings (Laplacian filter):
  - $O(x,y) = \nabla^2(I(x,y) * G(x,y))$

Just another linear filter.

$$\underbrace{\nabla^2(f(x,y) \otimes G(x,y))}_{\text{Laplacian of Gaussian-filtered image}} = \underbrace{\nabla^2 G(x,y) \otimes f(x,y)}_{\text{Laplacian of Gaussian (LoG) -filtered image}}$$

Laplacian of
Gaussian-filtered image

Laplacian of Gaussian (LoG)
-filtered image

Do you see the distinction?

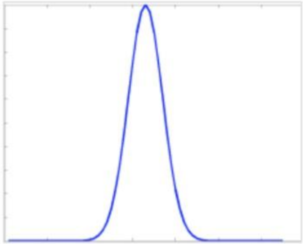
Other Important Filters

- Laplacian of Gaussian
  - Noise Suppression

Robert Collins
CSE486

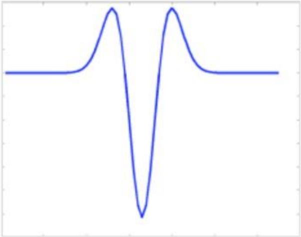
1D Gaussian and Derivatives

$$g(x) = e^{-\frac{x^2}{2\sigma^2}}$$



$$g'(x) = -\frac{1}{2\sigma^2}2xe^{-\frac{x^2}{2\sigma^2}} = -\frac{x}{\sigma^2}e^{-\frac{x^2}{2\sigma^2}}$$

$$g''(x) = (\frac{x^2}{\sigma^4} - \frac{1}{\sigma^2})e^{-\frac{x^2}{2\sigma^2}}$$



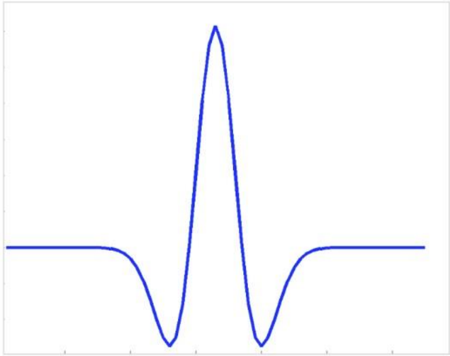
Other Important Filters

- Laplacian of Gaussian
  - Noise Suppression
-

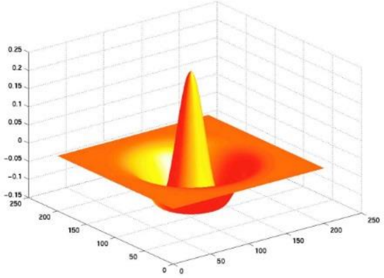
Robert Collins
CSE486

Second Derivative of a Gaussian

$$g''(x) = \left(\frac{x^2}{\sigma^4} - \frac{1}{\sigma^2}\right)e^{-\frac{x^2}{2\sigma^2}}$$



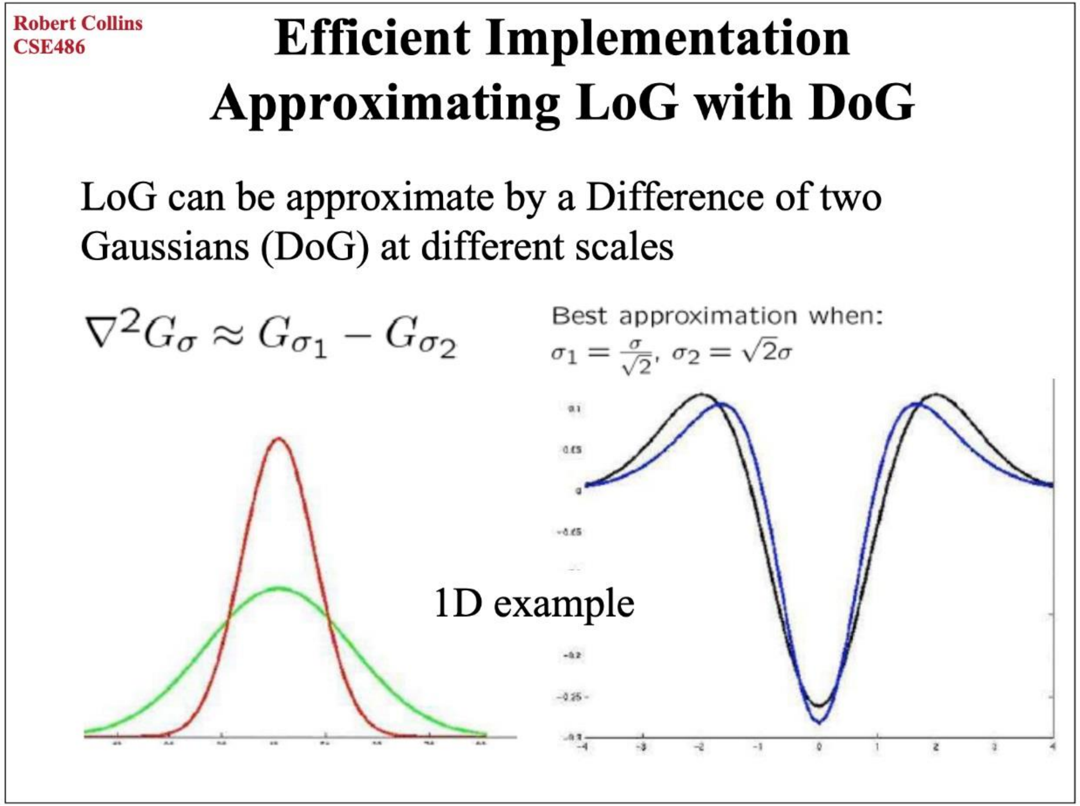
2D analog



LoG "Mexican Hat"

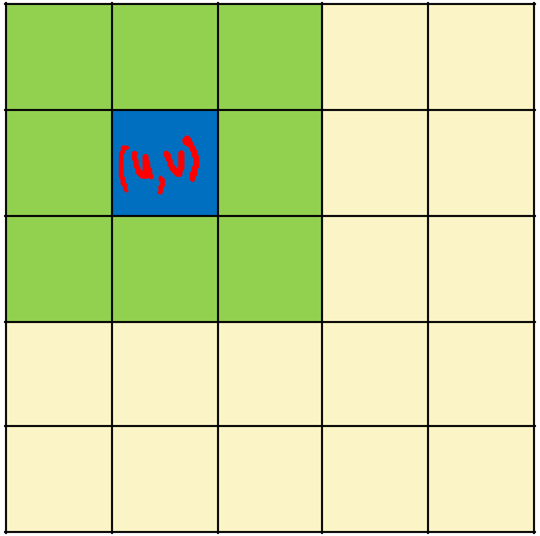
Other Important Filters

- Laplacian of Gaussian
  - Noise Suppression
- Difference of Gaussian
  - Band-pass



Remember this?

I



H → Mask

Kernel

Filter

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

$$I'(u,v) \leftarrow \frac{1}{9} \cdot \sum_{j=-1}^{1} \sum_{i=-1}^{1} I(u+i, v+j)$$

$$I'(u,v) \leftarrow \sum_{j=-1}^{1} \sum_{i=-1}^{1} I(u+i, v+j) \boxed{\cdot H(i,j)}$$

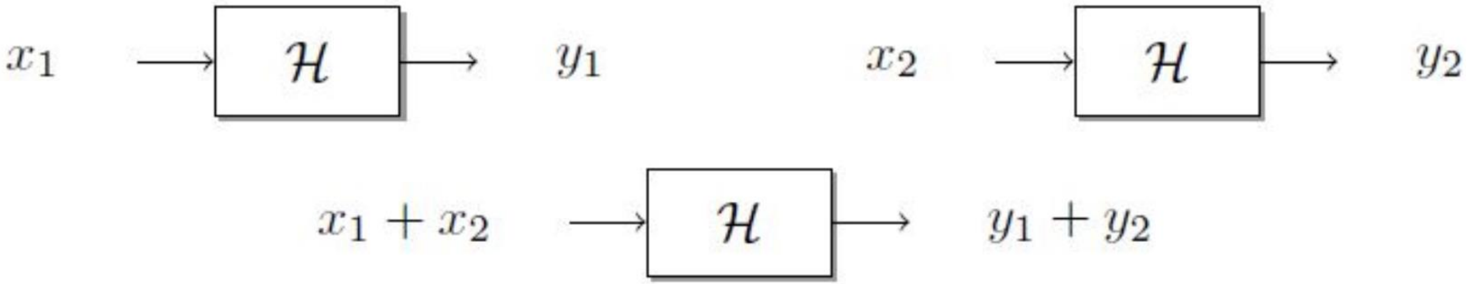
A system $\mathcal{H}$ is **linear** if it satisfies the following two properties:

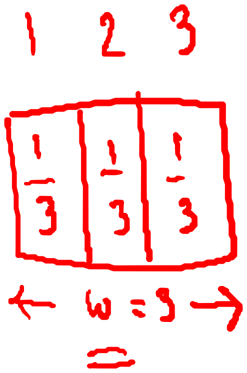
1) Scaling

$$\mathcal{H}\{\alpha x\} = \alpha \mathcal{H} \quad \forall\, \alpha \in \mathbb{C}$$

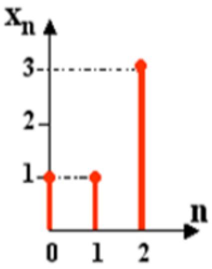
2) Additivity

$$\text{If } y_1 = \mathcal{H}\{x_1\} \text{ and } y_2 = \mathcal{H}\{x_2\} \text{ then } \mathcal{H}\{x_1 + x_2\} = y_1 + y_2$$





$\frac{1}{w}$



A system $\mathcal{H}$ is **linear** if it satisfies the following two properties:

1) Scaling

$$\mathcal{H}\{\alpha x\} = \alpha \mathcal{H} \quad \forall \alpha \in \mathbb{C}$$

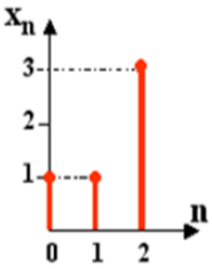
$$x \longrightarrow \boxed{\mathcal{H}} \longrightarrow y \qquad \alpha x \longrightarrow \boxed{\mathcal{H}} \longrightarrow \alpha y$$

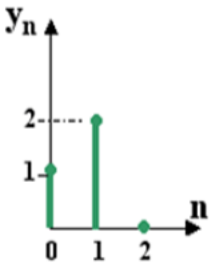
2) Additivity

$$\text{If } y_1 = \mathcal{H}\{x_1\} \text{ and } y_2 = \mathcal{H}\{x_2\} \text{ then } \mathcal{H}\{x_1 + x_2\} = y_1 + y_2$$

$$x_1 \longrightarrow \boxed{\mathcal{H}} \longrightarrow y_1 \qquad x_2 \longrightarrow \boxed{\mathcal{H}} \longrightarrow y_2$$

$$x_1 + x_2 \longrightarrow \boxed{\mathcal{H}} \longrightarrow y_1 + y_2$$





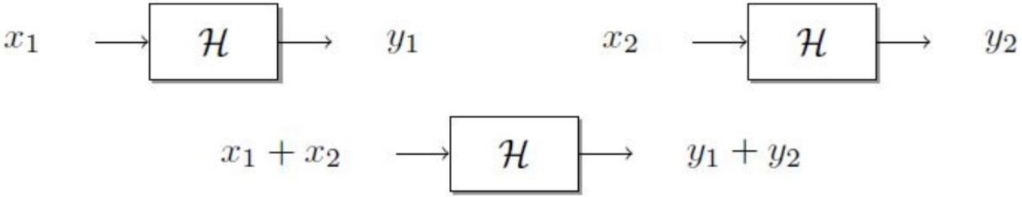
A system $\mathcal{H}$ is **linear** if it satisfies the following two properties:

1) Scaling

$$\mathcal{H}\{\alpha x\} = \alpha \mathcal{H} \quad \forall\, \alpha \in \mathbb{C}$$

2) Additivity

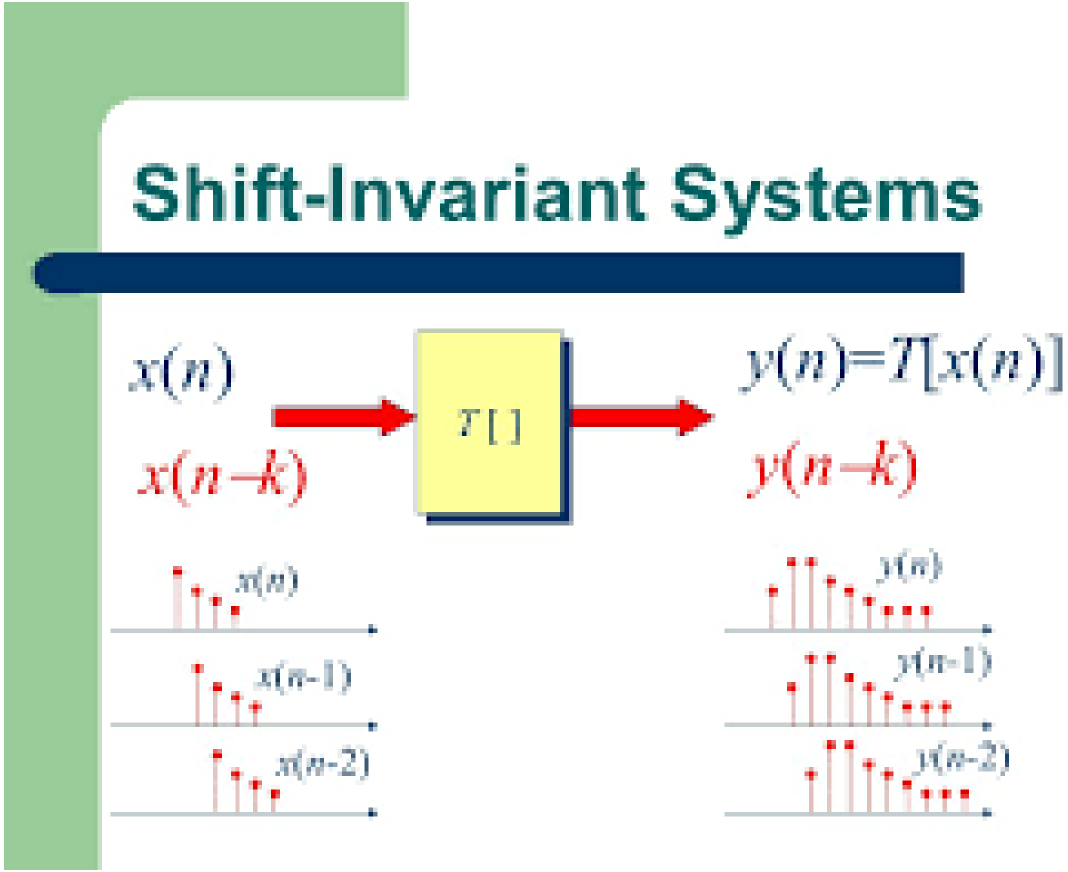
$$\text{If } y_1 = \mathcal{H}\{x_1\} \text{ and } y_2 = \mathcal{H}\{x_2\} \text{ then } \mathcal{H}\{x_1 + x_2\} = y_1 + y_2$$



Linear Systems

$$x(n) \longrightarrow \boxed{T[\;]} \longrightarrow y(n)=T[x(n)]$$

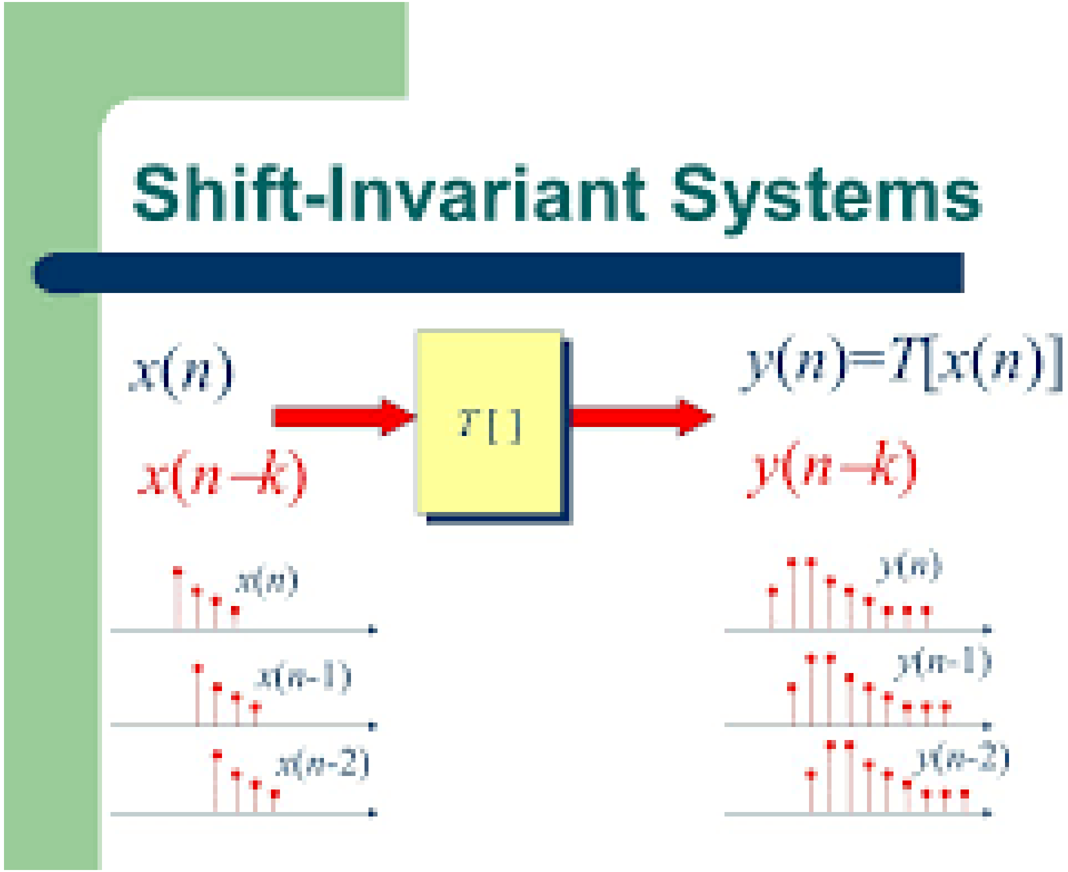
$$\boxed{T[ax_1(n)+bx_2(n)] = aT[x_1(n)]+bT[x_2(n)]}$$



0

E.g. $x1[n]$=[**4**,2,3,1,5,9]

- Mean filter (w=3)
- Zero padding
- *y1[n] = ?*



E.g. $x1[n]$=[**4**,2,3,1,5,9]

- Mean filter (w=3)
- Zero padding
- *y1[n] = ?*

$x2[n]=x1[n-3]$=[**0**,0,0,4,2,3,1,5,9]

y2[n] = ?

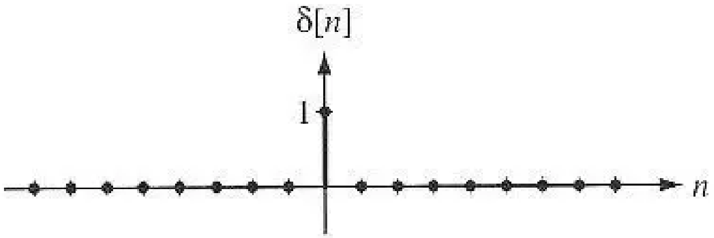
- Linearity and Shift-invariance are independent properties
- Shift-invariance does not imply linearity (and vice-versa)

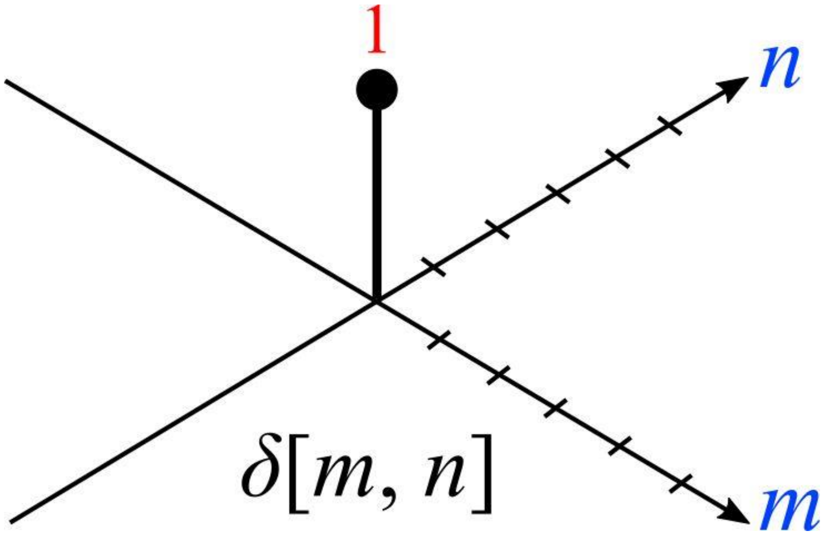
- Linear, Shift-Invariant Systems ?

Input	LTI system	Output
Pre-synaptic action potentials	Synapse	Post-synaptic conductance
Visual stimulus	Eye	Retinal image
Stimulus contrast	Retinal ganglion cell	Firing rate
Injected Current	Passive neural membrane	Membrane potential

- Impulse Function

$$\delta[n] = \begin{cases} 1, & n = 0 \\ 0, & n \neq 0 \end{cases}$$



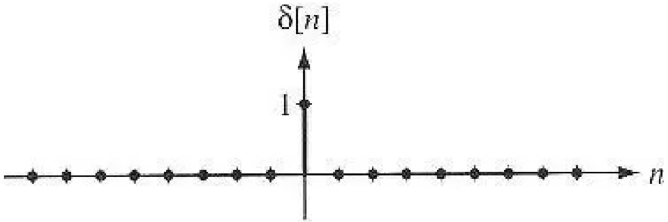


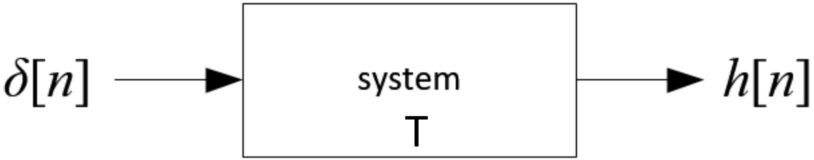
0	0	0	0	0
0	0	0	0	0
0	0	1	0	0
0	0	0	0	0
0	0	0	0	0

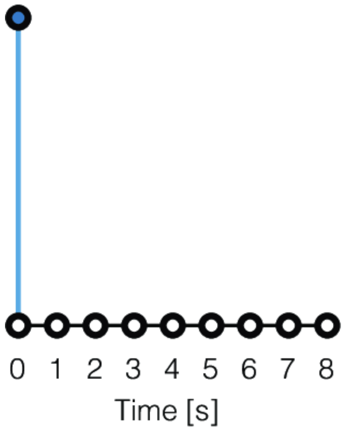
Impulse Function

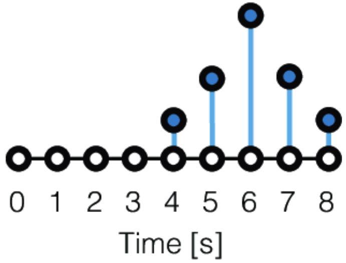
Impulse Response of a system T

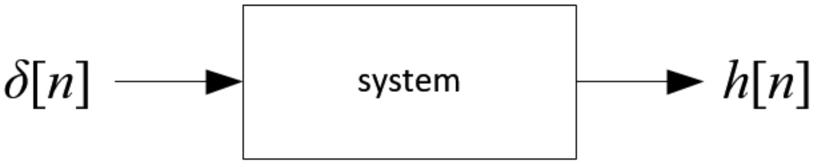
$$\boxed{\delta[n] = \begin{cases} 1, & n = 0 \\ 0, & n \neq 0 \end{cases}}$$



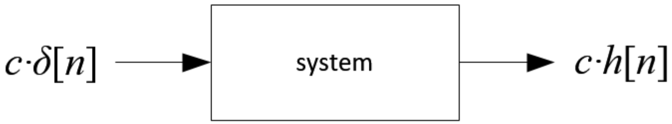




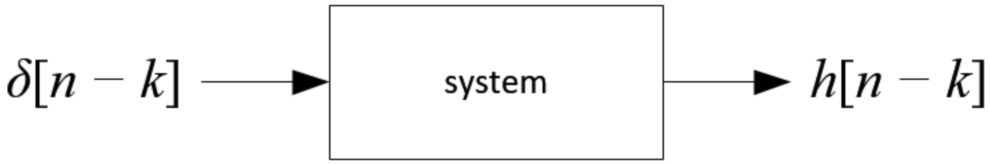


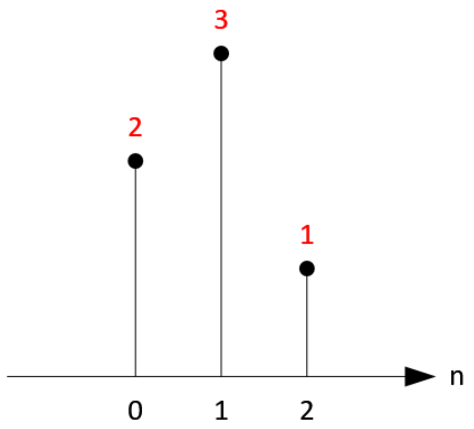


If system is linear

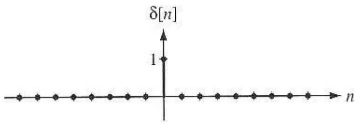


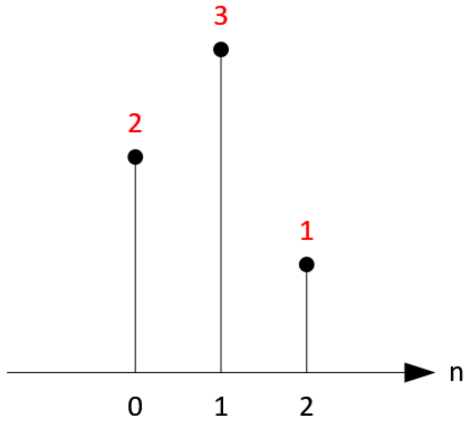
If system is shift-invariant




3
2
1
n
0
1
2

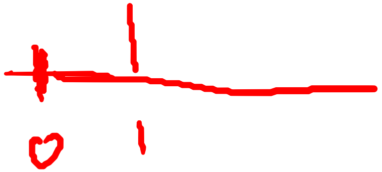
$$\delta[n]=\begin{cases}1\,, & n=0\\ 0\,, & n\neq 0\end{cases}$$

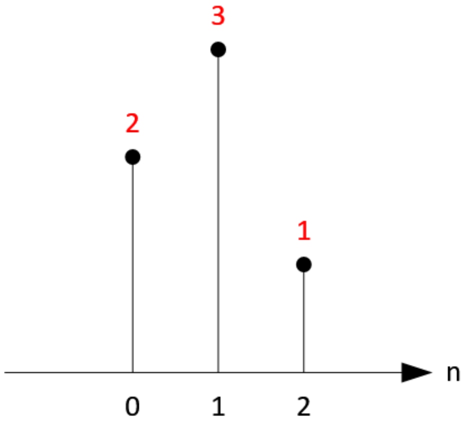




$$x[0] = x[0]\cdot\delta[n] \qquad = 2\cdot\delta[n-0]$$
$$x[1] = x[1]\cdot\delta[n-1] = 3\cdot\delta[n-1]$$
$$x[2] = x[2]\cdot\delta[n-2] = 1\cdot\delta[n-2]$$

$$x[n] = x[0]\cdot\delta[n-0] + \underbrace{x[1]}\cdot\underbrace{\delta[n-\overset{k}{1}]} + x[2]\cdot\delta[n-2]$$





$$x[0] = x[0] \cdot \delta[n] \qquad = 2 \cdot \delta[n-0]$$

$$x[1] = x[1] \cdot \delta[n-1] = 3 \cdot \delta[n-1]$$

$$x[2] = x[2] \cdot \delta[n-2] = 1 \cdot \delta[n-2]$$

$$x[n] = x[0] \cdot \delta[n-0] + x[1] \cdot \delta[n-1] + x[2] \cdot \delta[n-2]$$

$$x[n] = \sum_{k=-\infty}^{\infty} x[k] \cdot \delta[n-k]$$

A signal can be written as sum of scaled and shifted delta functions

$$x[n] = \sum_{k=-\infty}^{\infty} \underbrace{x[k]}_{c_k} \cdot \delta[n-k]$$

$$\sum_k c_k \cdot \delta[n-k] \longrightarrow \boxed{\text{system}} \longrightarrow \sum_k c_k \cdot h[n-k]$$

Because system is LSI

$$T\left(\sum_k c_k \, \delta[n-k]\right)$$

$$= \sum_k T\left(c_k \, \delta[n-k]\right)$$

$$= \sum_k c_k \, T\left(\delta[n-k]\right)$$

$$= \sum_k c_k \, h[n-k]$$

$$y[n] = \sum x[k] \cdot h[n-k]$$

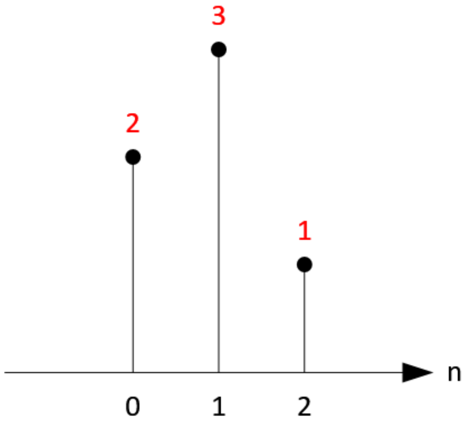
$$x[n] = \sum_{k=-\infty}^{\infty} x[k] \cdot \delta[n-k]$$

$$\sum_k c_k \cdot \delta[n-k] \longrightarrow \boxed{\text{system}} \longrightarrow \sum_k c_k \cdot h[n-k]$$

Because system is LSI

$$y[n] = \sum_{k} \overbrace{x[k]}^{c_k} \cdot h[n-k]$$

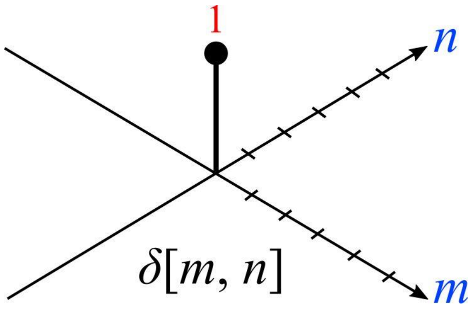
Convolution of x and h : y = x * h

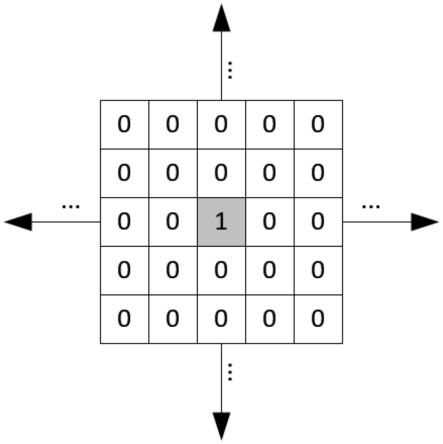


$$x[n] = \sum_{k=-\infty}^{\infty} x[k] \cdot \delta[n-k]$$

A signal can be written as sum of scaled and shifted delta functions

2-D



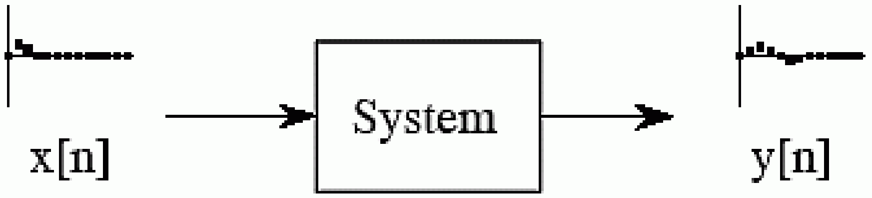


$$x[m,n]=\sum_{j=-\infty}^{\infty}\sum_{i=-\infty}^{\infty}x[i,j]\cdot\delta[m-i,n-j]$$

A signal can be written as sum of scaled and shifted delta functions

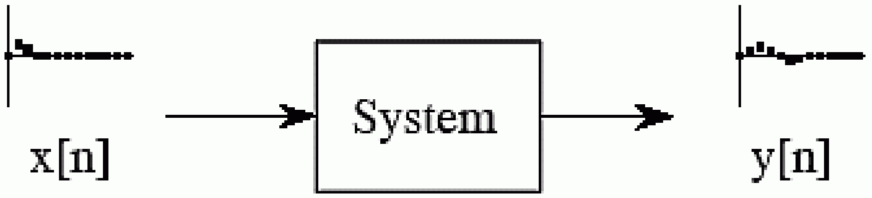
$$x[m,n] = \sum_{j=-\infty}^{\infty} \sum_{i=-\infty}^{\infty} x[i,j] \cdot \delta[m-i, n-j]$$

$$y[m,n] = x[m,n] * h[m,n] = \sum_{j=-\infty}^{\infty} \sum_{i=-\infty}^{\infty} x[i,j] \cdot h[m-i, n-j]$$



$$x[m,n] = \sum_{j=-\infty}^{\infty} \sum_{i=-\infty}^{\infty} x[i,j] \cdot \delta[m-i, n-j]$$

$$y[m,n] = x[m,n] * h[m,n] = \sum_{j=-\infty}^{\infty} \sum_{i=-\infty}^{\infty} x[i,j] \cdot h[m-i, n-j]$$



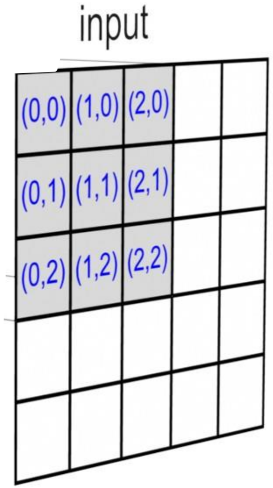
$$x[m,n] = \sum_{j=-\infty}^{\infty} \sum_{i=-\infty}^{\infty} x[i,j] \cdot \delta[m-i, n-j]$$

$$y[m,n] = x[m,n] * h[m,n] = \sum_{j=-\infty}^{\infty} \sum_{i=-\infty}^{\infty} x[i,j] \cdot h[m-i, n-j]$$

n \ m	-1	0	1
-1	a	b	c
0	d	e	f
1	g	h	i

$$y[1,1] = \sum_{j=-\infty}^{\infty} \sum_{i=-\infty}^{\infty} x[i,j] \cdot h[1-i, 1-j]$$

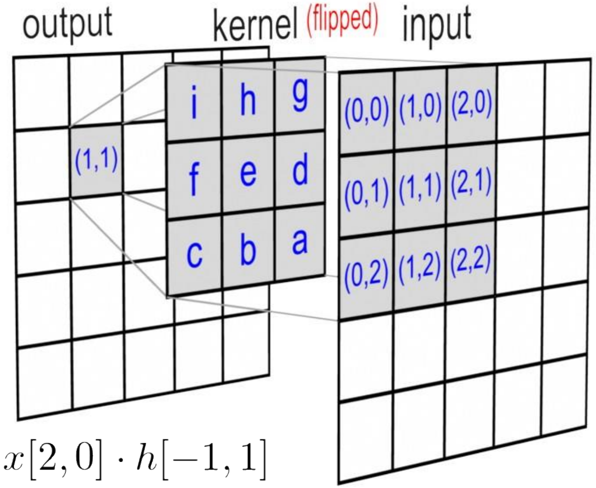
$$\begin{aligned} = \quad & x[0,0] \cdot h[1,1] + x[1,0] \cdot h[0,1] + x[2,0] \cdot h[-1,1] \\ & + x[0,1] \cdot h[1,0] + x[1,1] \cdot h[0,0] + x[2,1] \cdot h[-1,0] \\ & + x[0,2] \cdot h[1,-1] + x[1,2] \cdot h[0,-1] + x[2,2] \cdot h[-1,-1] \end{aligned}$$



$$x[m,n] = \sum_{j=-\infty}^{\infty} \sum_{i=-\infty}^{\infty} x[i,j] \cdot \delta[m-i, n-j]$$

$$y[m,n] = x[m,n] * h[m,n] = \sum_{j=-\infty}^{\infty} \sum_{i=-\infty}^{\infty} x[i,j] \cdot h[m-i, n-j]$$

n \ m	-1	0	1
-1	a	b	c
0	d	e	f
1	g	h	i



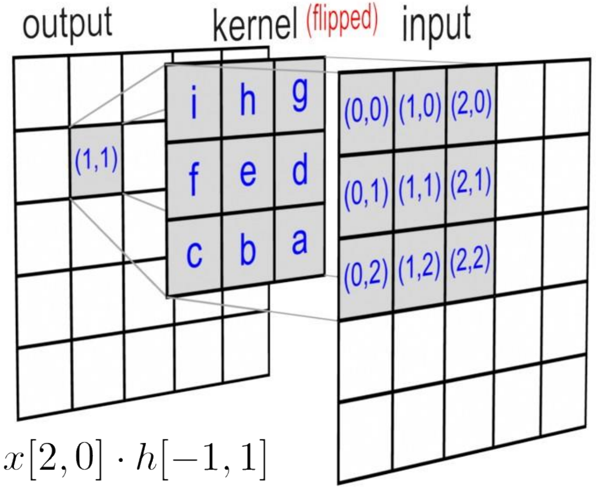
$$\begin{aligned} y[1,1] &= \sum_{j=-\infty}^{\infty} \sum_{i=-\infty}^{\infty} x[i,j] \cdot h[1-i, 1-j] \\ &= x[0,0] \cdot h[1,1] + x[1,0] \cdot h[0,1] + x[2,0] \cdot h[-1,1] \\ &\quad + x[0,1] \cdot h[1,0] + x[1,1] \cdot h[0,0] + x[2,1] \cdot h[-1,0] \\ &\quad + x[0,2] \cdot h[1,-1] + x[1,2] \cdot h[0,-1] + x[2,2] \cdot h[-1,-1] \end{aligned}$$

$$x[m,n] = \sum_{j=-\infty}^{\infty} \sum_{i=-\infty}^{\infty} x[i,j] \cdot \delta[m-i, n-j]$$

If the filter is symmetric, flipping kernel is not necessary

$$y[m,n] = x[m,n] * h[m,n] = \sum_{j=-\infty}^{\infty} \sum_{i=-\infty}^{\infty} x[i,j] \cdot h[m-i, n-j]$$

n \ m	-1	0	1
-1	a	b	c
0	d	e	f
1	g	h	i

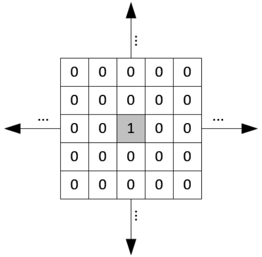


$$y[1,1] = \sum_{j=-\infty}^{\infty} \sum_{i=-\infty}^{\infty} x[i,j] \cdot h[1-i, 1-j]$$

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

- Above convolution works on linear and shift invariant system

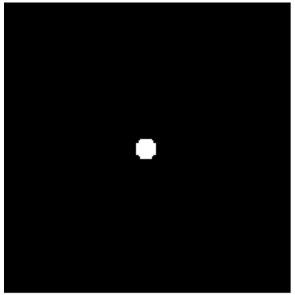
Blurring and Impulse Function



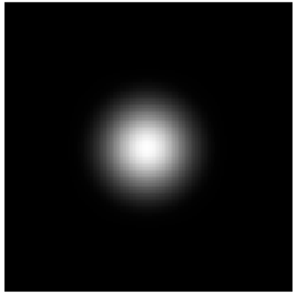
$\frac{1}{273}$

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

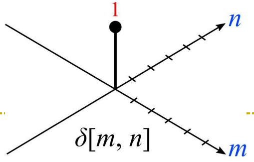
Blurred



Convolution



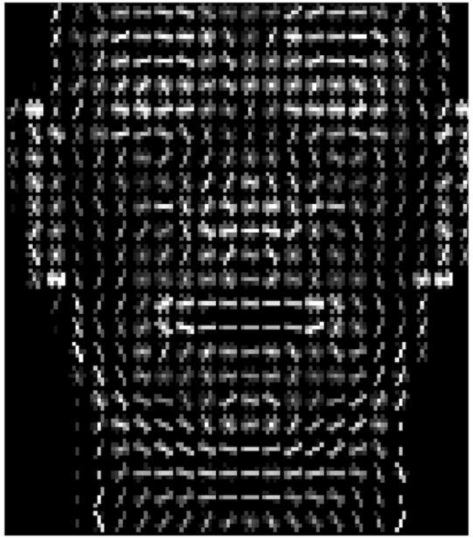
Point



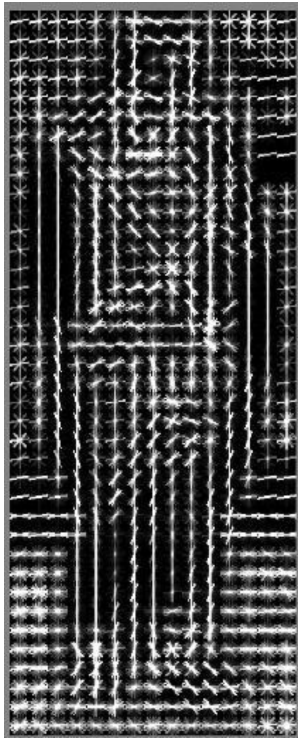
Filters for Image Analysis

- We learned filters for image enhancement
- But also very important for image analysis tasks
- Fundamental tool for Feature based Image Representation
  - Necessary Computer Vision and Machine Learning

Visualizing Gradients

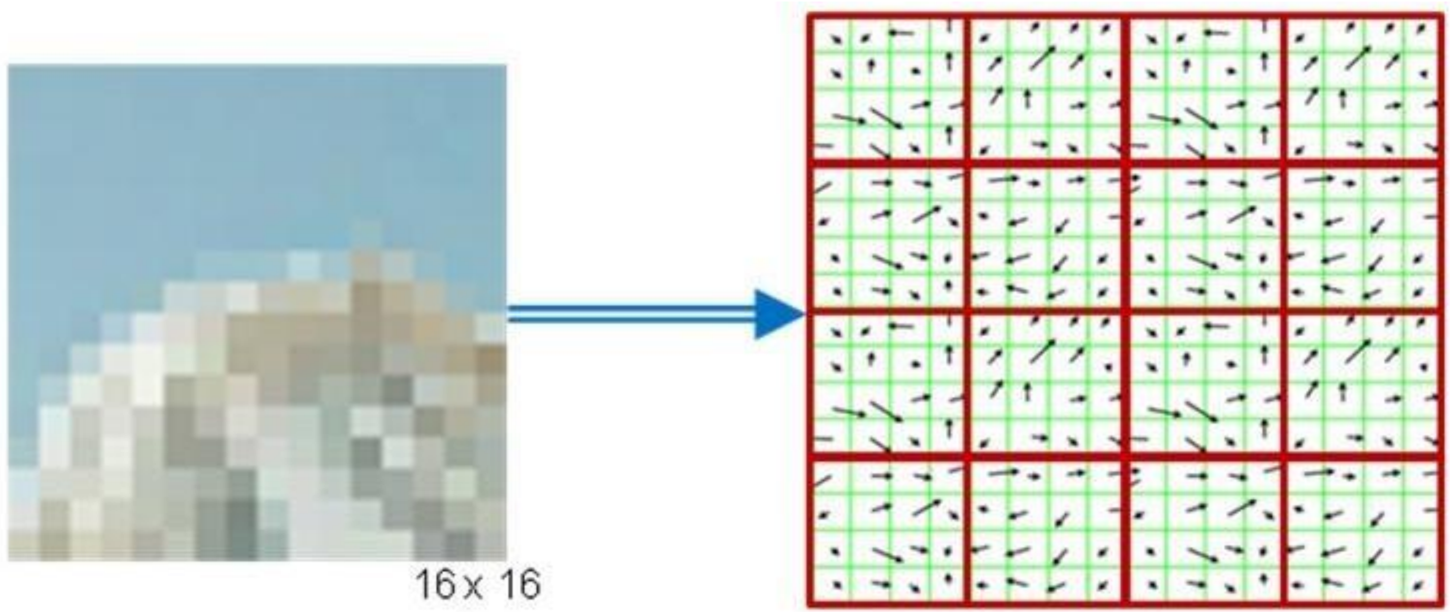


Face

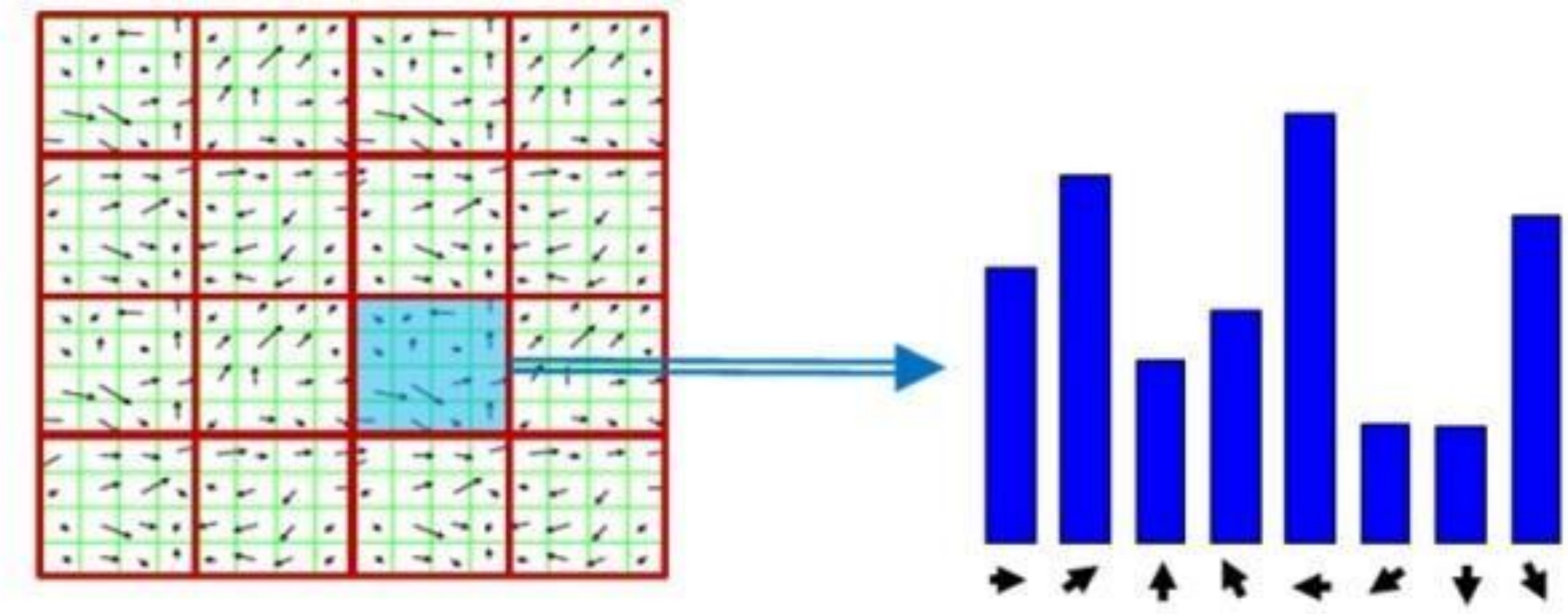


Person

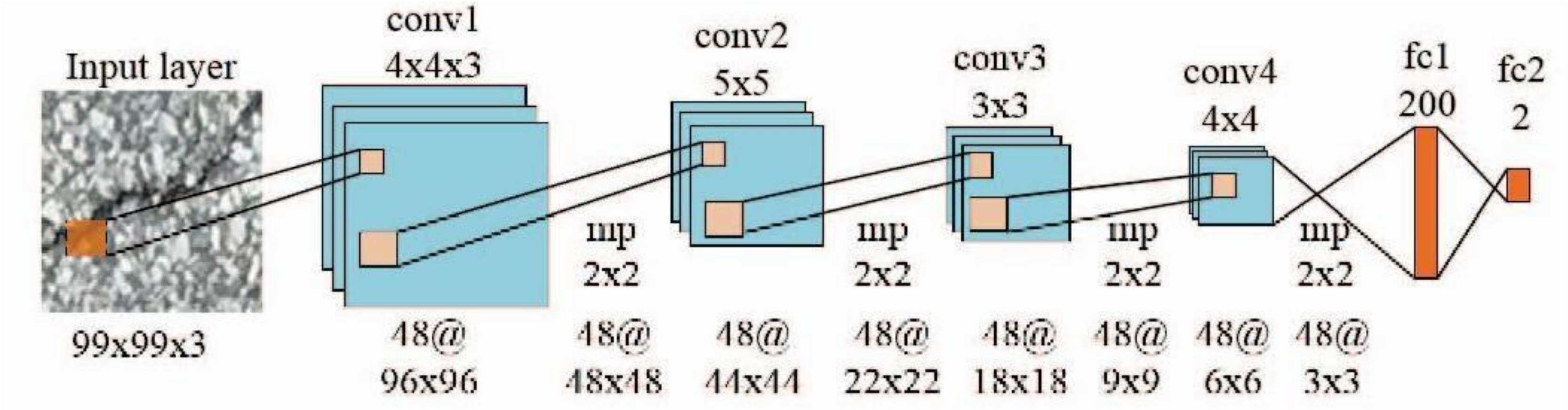
Gradient Orientations



Histogram of Gradient Orientations



Learning Filters



Central to Convolutional Neural Network

Summary

- Linear Filtering – moving a weight mask over the input image, multiplying weights with intensity values, and summing them up to produce output image
- Linear Filtering as Convolution
  - Part of larger LSI systems
- Nonlinear Filtering – min, max, median, bilateral (mask is data-dependent)

References

- GW Chapter – 3.4, 3.5.2
- Szeliski Book : Computer Vision and Applications
  - Bilateral Filtering
- Convolution and LSI:

- http://www.songho.ca/dsp/convolution/convolution.html
- http://www.ceri.memphis.edu/people/smalley/ESCI7355/Ch6_Linear_Systems_Conv.pdf