17.09.2021

Digital Image Processing (CSE/ECE 478)

Lecture-8: Bilateral Filtering, Linearity Intro to Frequency Domain Processing

Ravi Kiran and Sudipta Banerjee

Center for Visual Information Technology (CVIT), IIIT Hyderabad

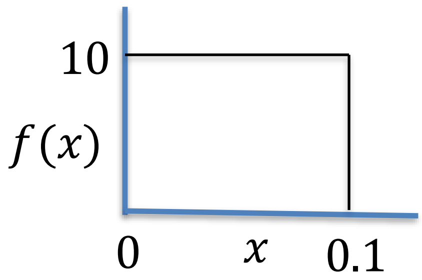
CVIT

Announcements

- MQ2: Points will be awarded to students who "attempted" qs: To represent 95% of the area within the Gaussian distribution, which of the following standard deviation values should be used?

Recap

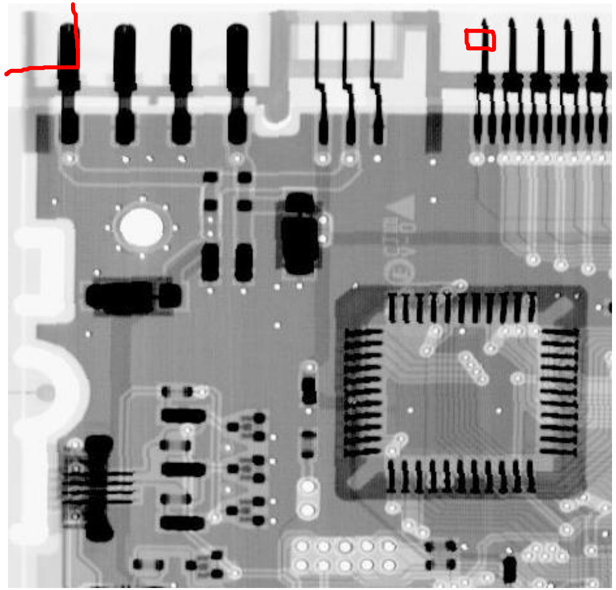
- PDF can be >1
  - E.g., Uniform distribution $U(0,0.1)$ $f(x) = K, 0 \leq x \leq 0.1$ and $f(x) = 0, elsewhere$
  - $\int_{-\infty}^{\infty} f(x)dx = \int_{0}^{0.1} K.dx = K.x|_{0}^{0.1}$ =1
  - $K = 10$



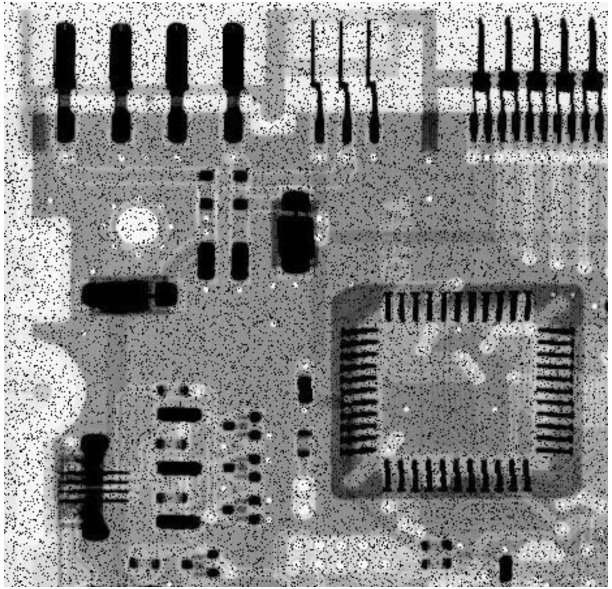
- Less consensus about Positive Laplacian (central element being positive vs. negative)

https://academic.mu.edu/phys/matthysd/web226/Lab02.htm
https://www.tutorialspoint.com/dip/laplacian_operator.htm

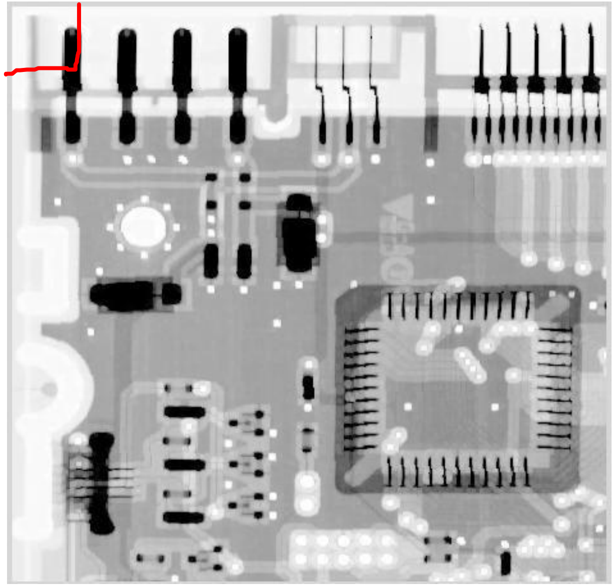
Non-linear Spatial Filters (max)



pepper noise



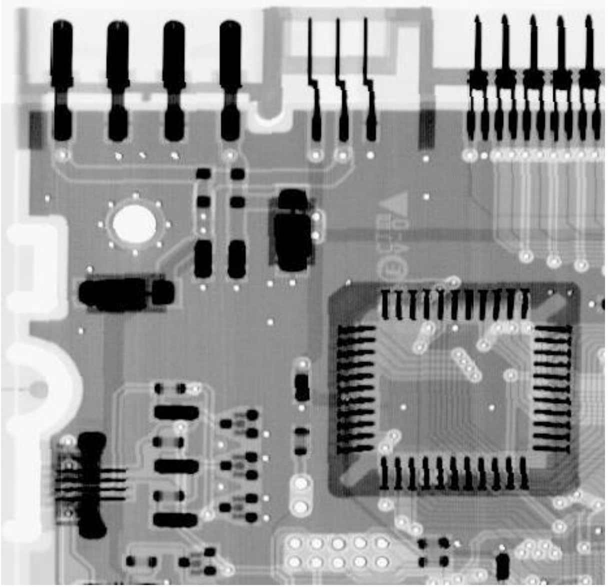
After applying max filter

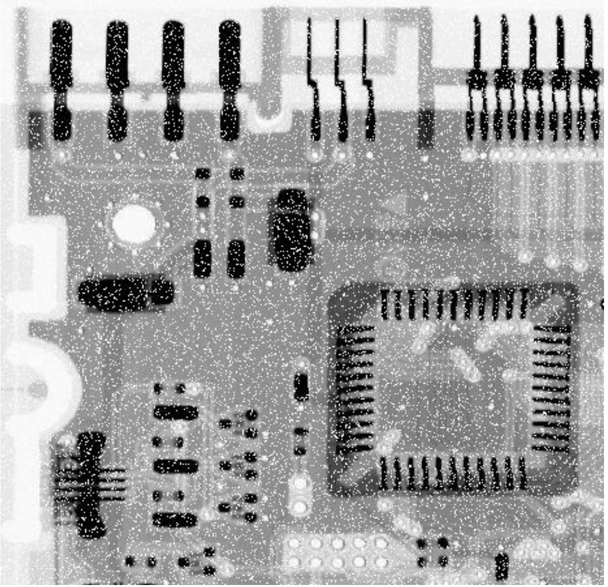


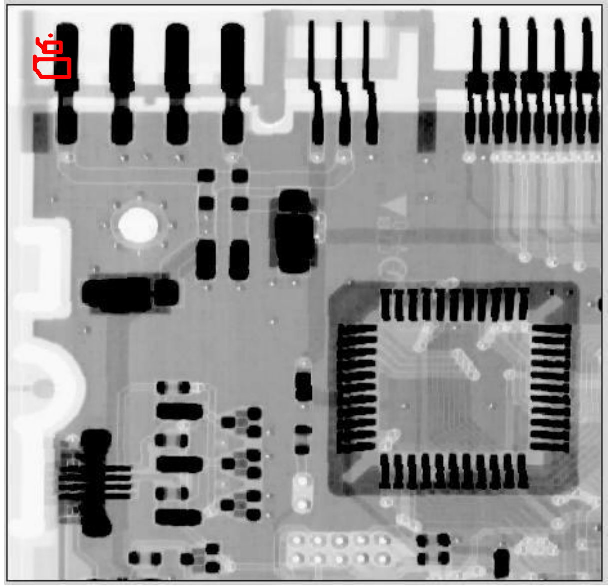
Non-linear Spatial Filters (min)

salt noise

After applying min filter



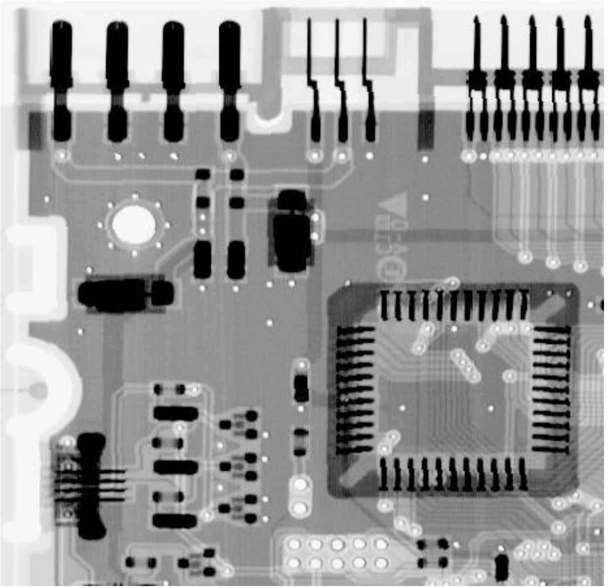


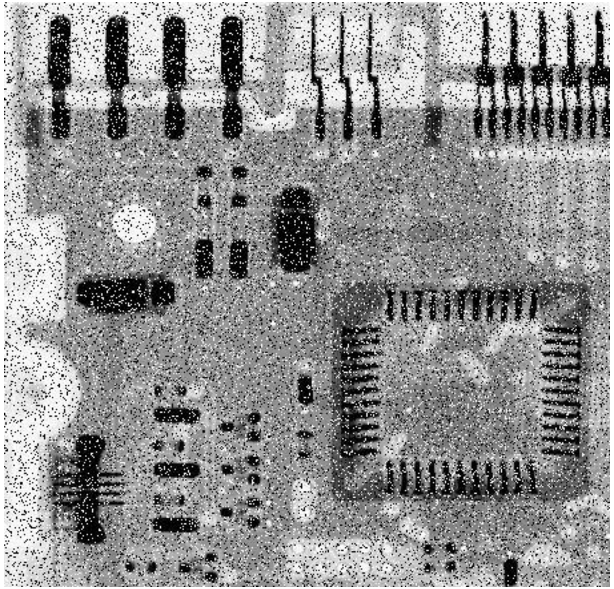


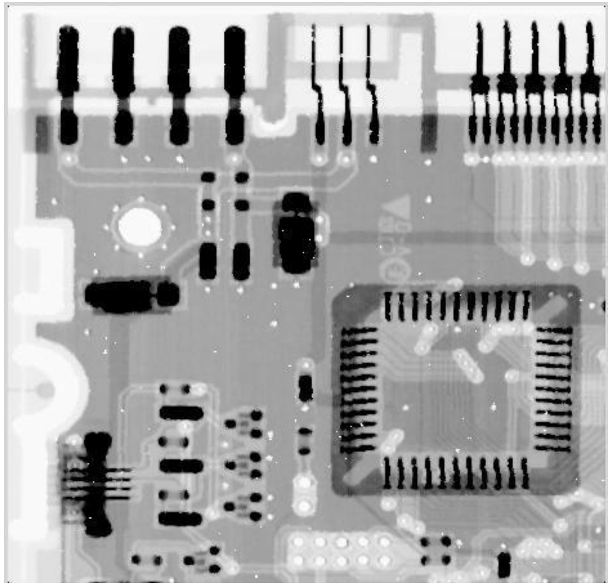
Non-linear Spatial Filters (median)

salt & pepper noise

After applying median filter







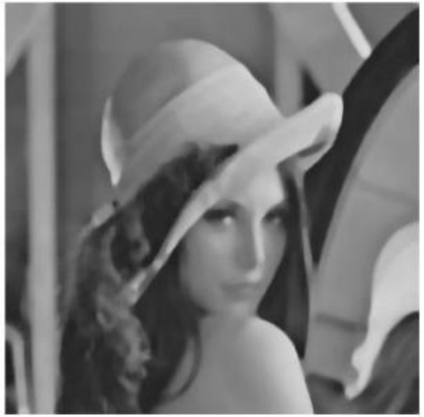
max, min, median → also known as rank / order statistic filters

- Mean: blurs image, removes simple noise, no details are preserved
- Gaussian: blurs image, preserves details only for small σ.
- Median: preserves some details, good at removing strong noise

original

3x3 mean

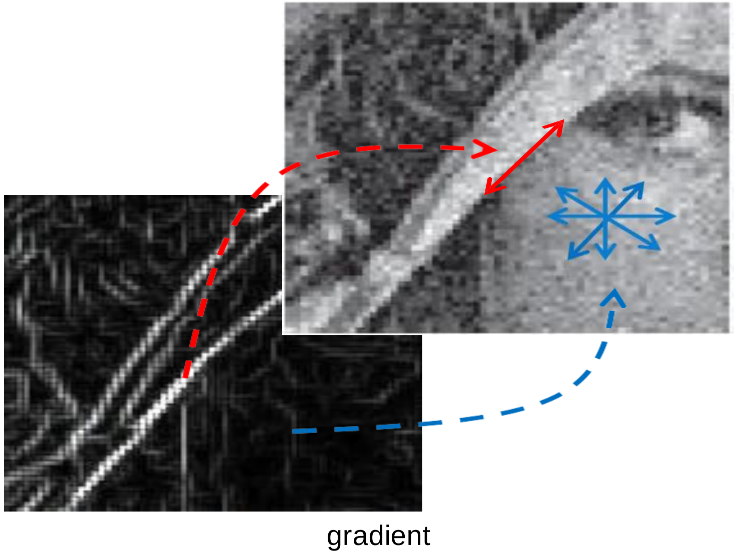
3x3 gaussian



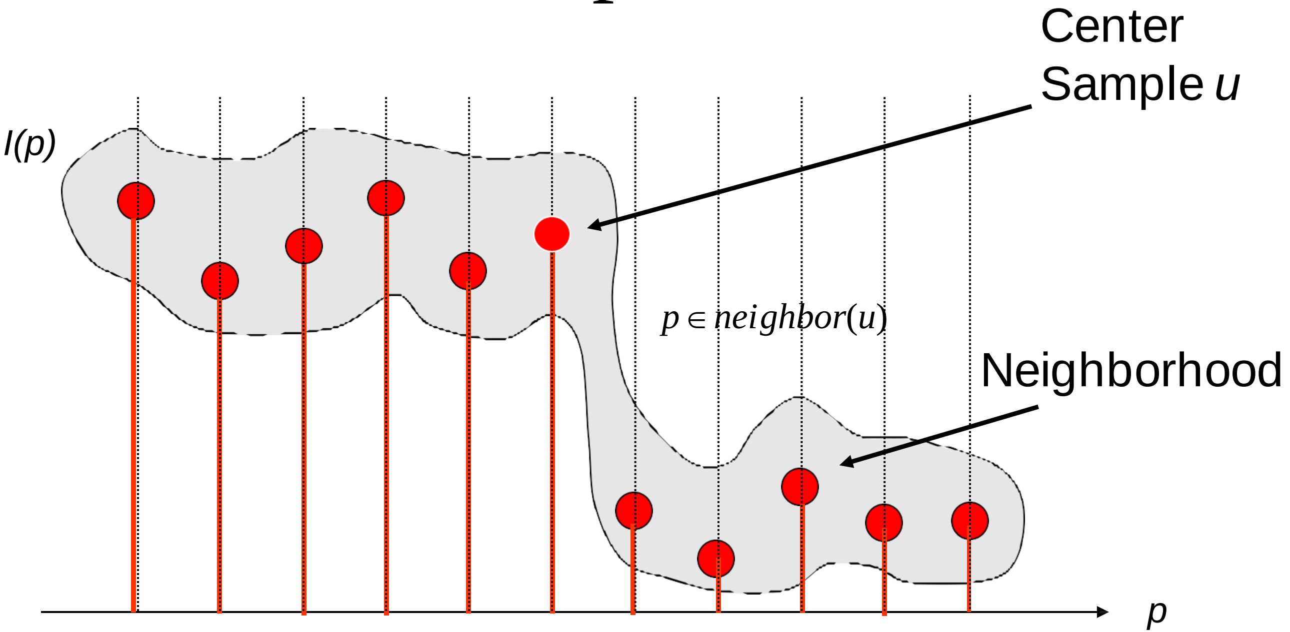
3x3 median

Edge Preserving Filtering

- Edges ⇒ smooth only along edges
- "Smooth" regions ⇒ smooth isotropically



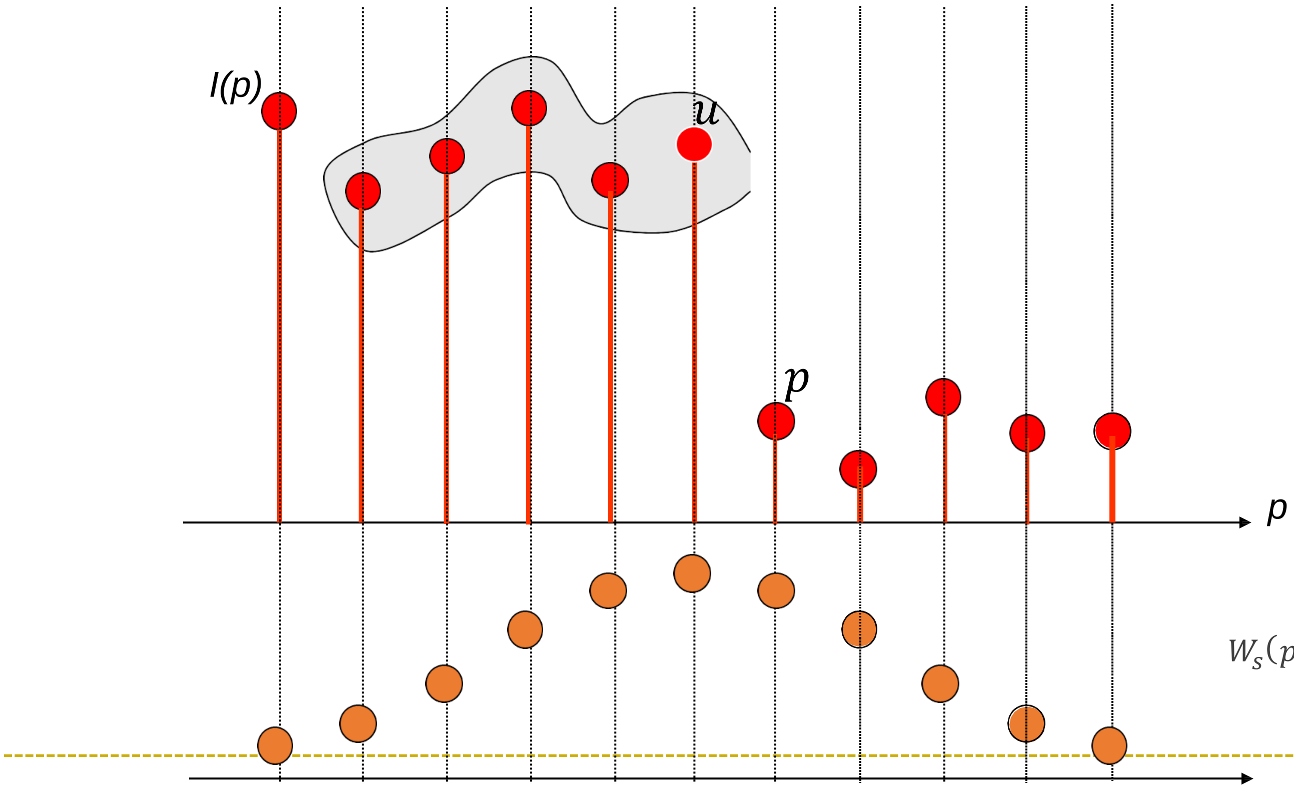
Bilateral Filters - 1D example

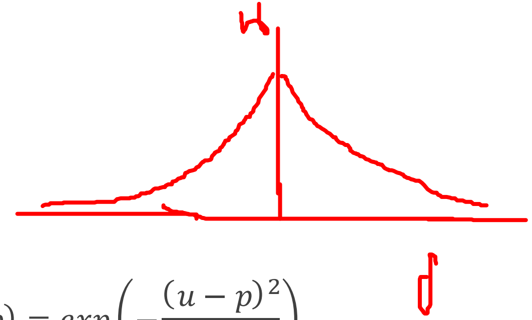


It is clear that in weighting this neighborhood, we would like to preserve the step

Gaussian Weights

- Gaussian Weights

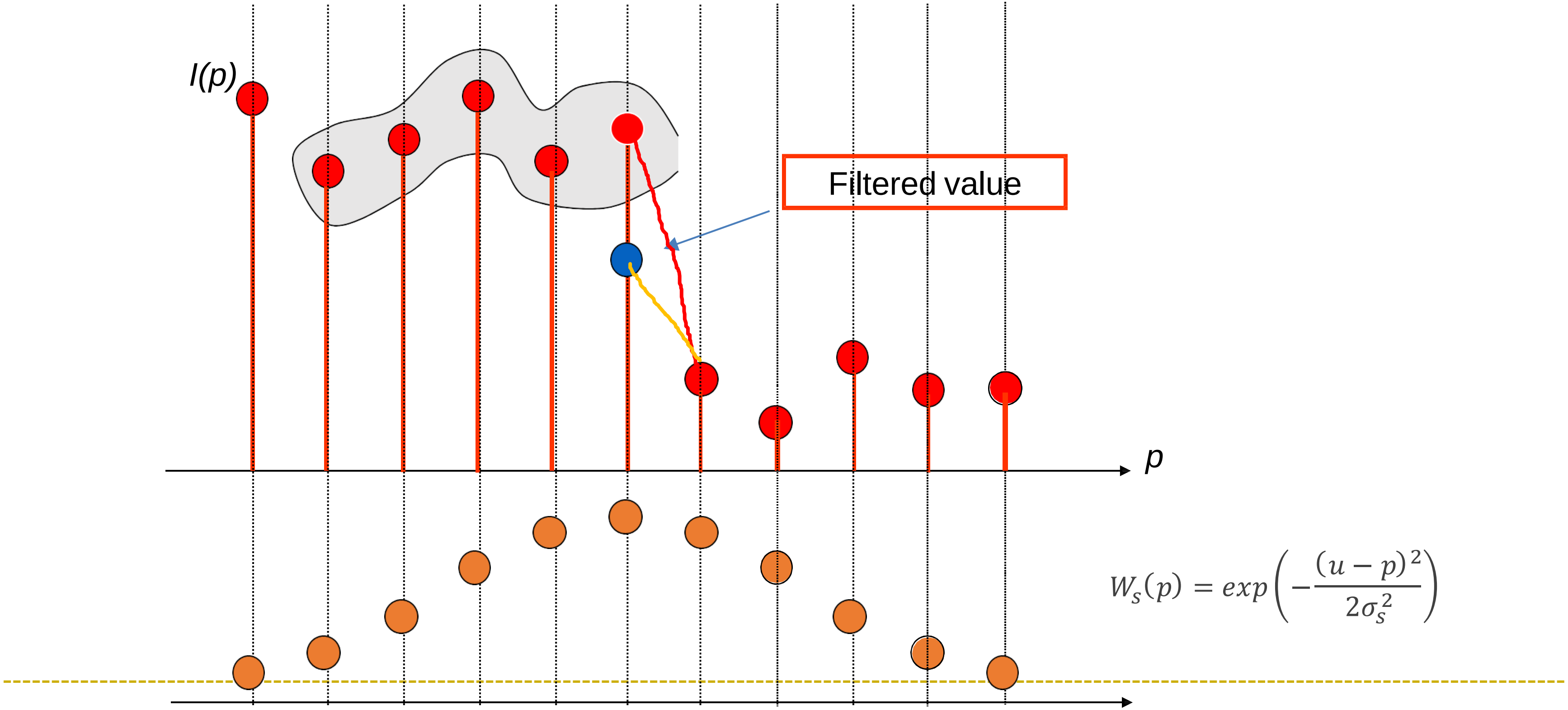




$$W_s(p) = exp\left(-\frac{(u-p)^2}{2\sigma_s^2}\right)$$

Edge loss

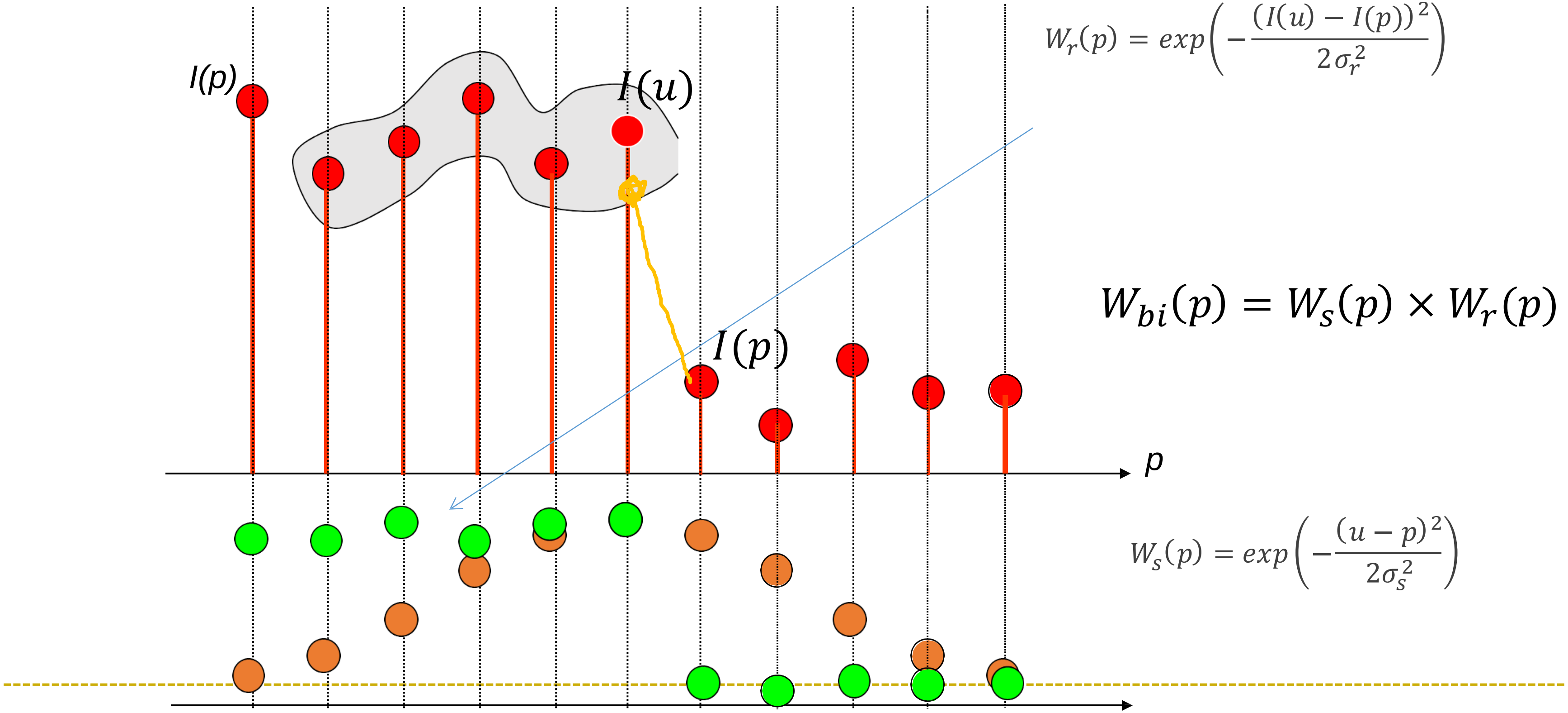
- Edge is smoothed/lost



$$W_s(p) = exp\left(-\frac{(u-p)^2}{2\sigma_s^2}\right)$$

Photometric Weights

- Introducing Photometric weights



$$W_r(p) = exp\left(-\frac{(I(u)-I(p))^2}{2\sigma_r^2}\right)$$

$$W_{bi}(p) = W_s(p) \times W_r(p)$$

$$W_s(p) = exp\left(-\frac{(u-p)^2}{2\sigma_s^2}\right)$$

Bilateral Filtering

- Filter Weights derived from both geometric and photometric distances

Denoise

Edge preserving

$$I'(u) = \frac{\sum_{p \in N(u)} e^{-\frac{\|u-p\|^2}{2\sigma_s^2}} e^{-\frac{|I(u)-I(p)|}{2\sigma_r^2}} I(p)}{\sum_{p \in N(u)} e^{-\frac{\|u-p\|^2}{2\sigma_s^2}} e^{-\frac{|I(u)-I(p)|}{2\sigma_r^2}}}$$

Normalization

Bilateral Filtering

- Filter Weights derived from both geometric and photometric distances

$$BF[I]_{\mathbf{p}} = \frac{1}{W_{\mathbf{p}}} \sum_{\mathbf{q} \in S} G_{\sigma_s}(\| \mathbf{p} - \mathbf{q} \|) \, G_{\sigma_r}(| I_{\mathbf{p}} - I_{\mathbf{q}} |) \, I_{\mathbf{q}}$$

space

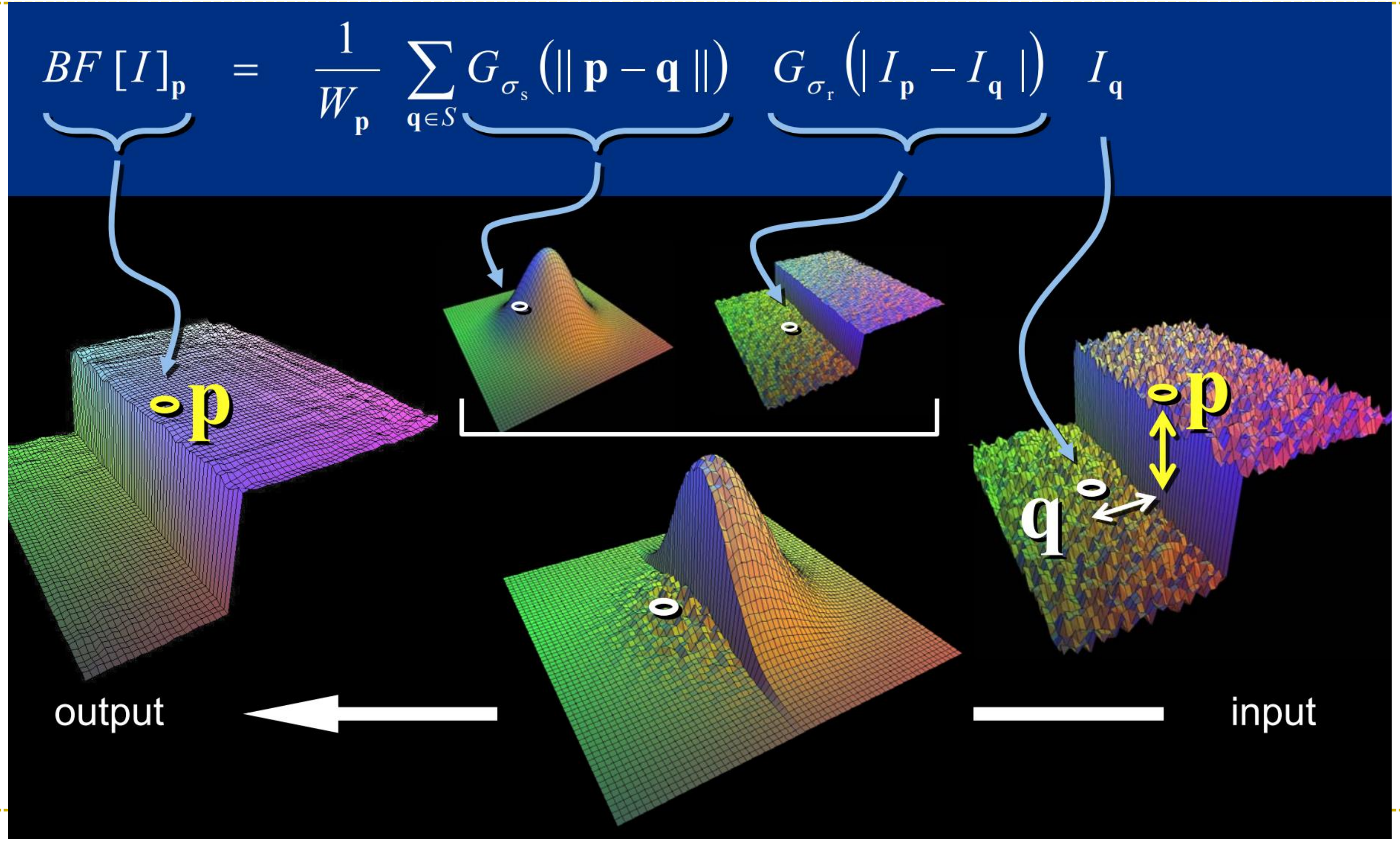
range

normalization

Source: Sylvain Paris, "A Gentle Introduction to Bilateral Filtering and Its Applications", SIGGRAPH 2007

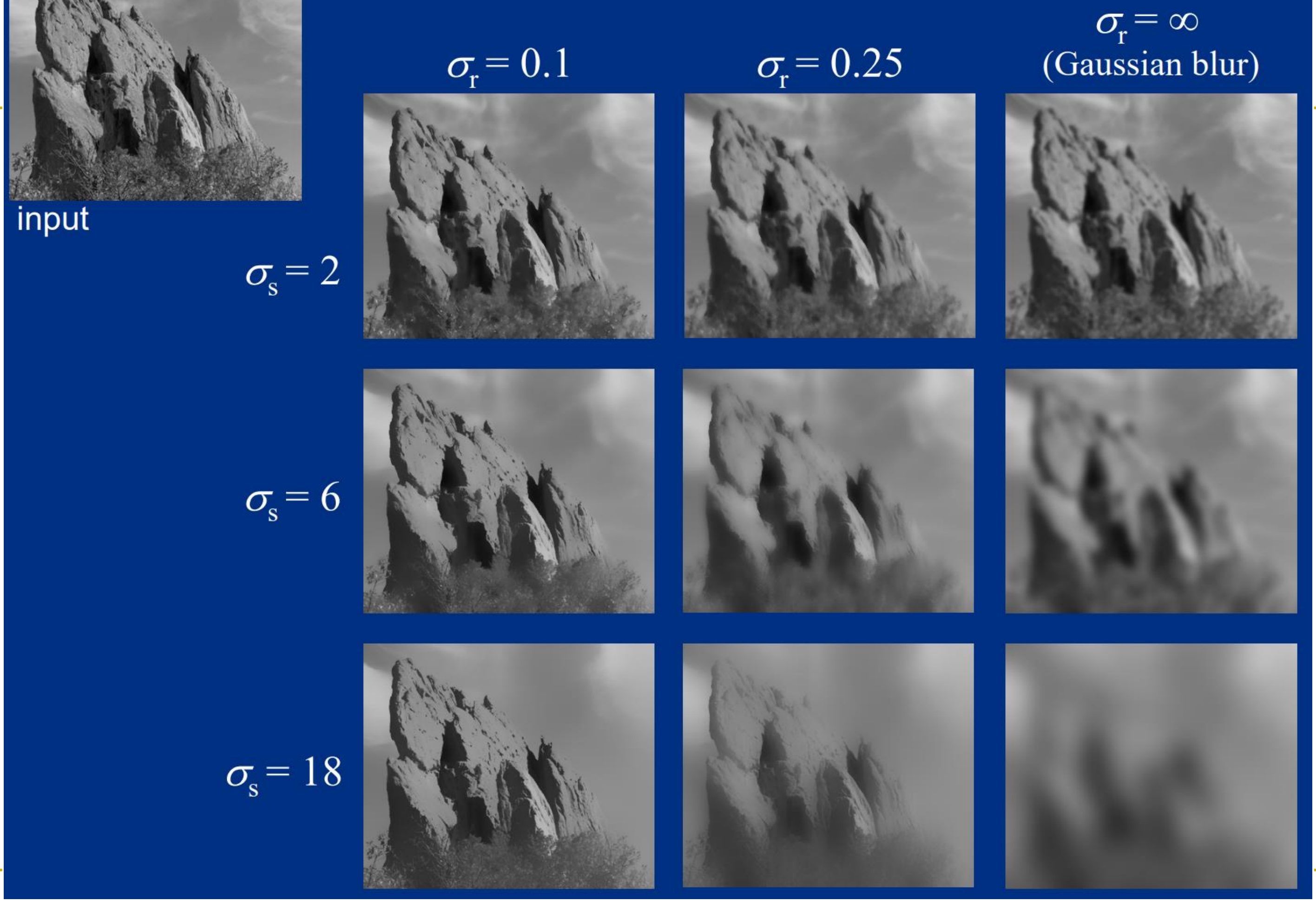
Bilateral Filtering

- Illustration of bilateral filter changes



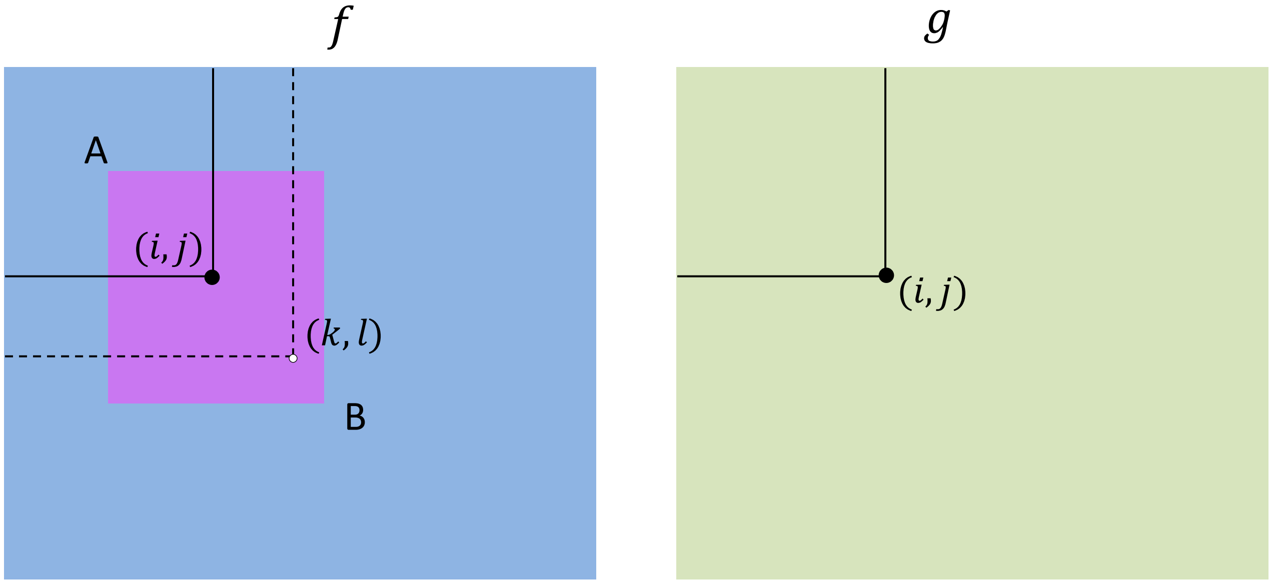
Source: Sylvain Paris, "A Gentle Introduction to Bilateral Filtering and Its Applications", SIGGRAPH 2007

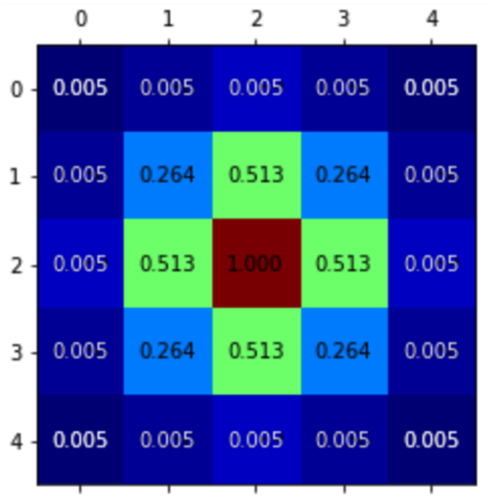
Exploring the Parameter Space



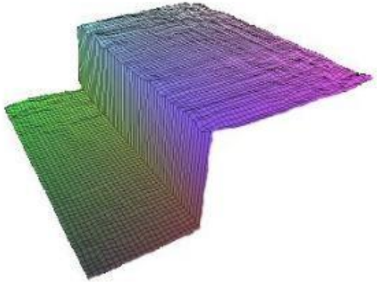
Source: Sylvain Paris, “A Gentle Introduction to Bilateral Filtering and Its Applications”, SIGGRAPH 2007

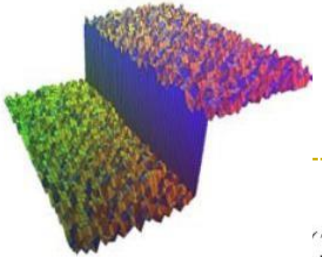
Linear Spatial Filter



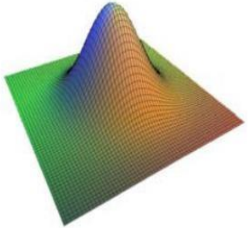


$$g(i, j) = \frac{\sum_{k,l} f(k, l)w(i, j, k, l)}{\sum_{k,l} w(i, j, k, l)}$$





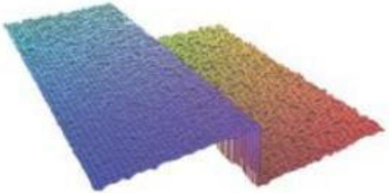
$$g(i,j) = \frac{\sum_{k,l} f(k,l)w(i,j,k,l)}{\sum_{k,l} w(i,j,k,l)}. \tag{3.34}$$



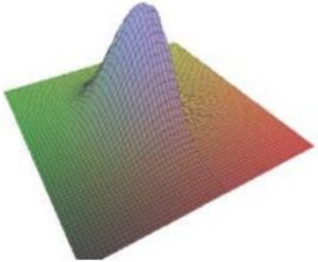
The weighting coefficient $w(i,j,k,l)$ depends on the product of a *domain kernel* (Figure 3.19c),

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

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



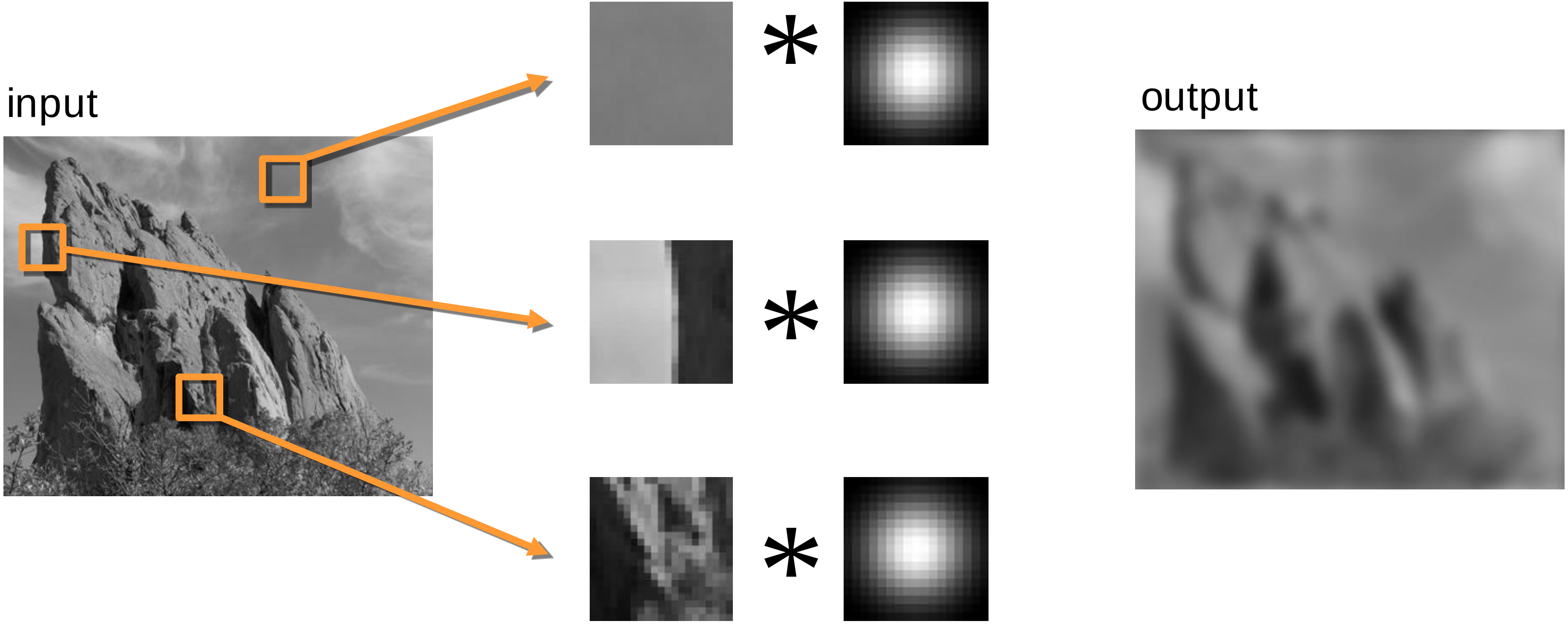
$$r(i,j,k,l) = \exp\left(-\frac{\|f(i,j)-f(k,l)\|^2}{2\sigma_r^2}\right).$$



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

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

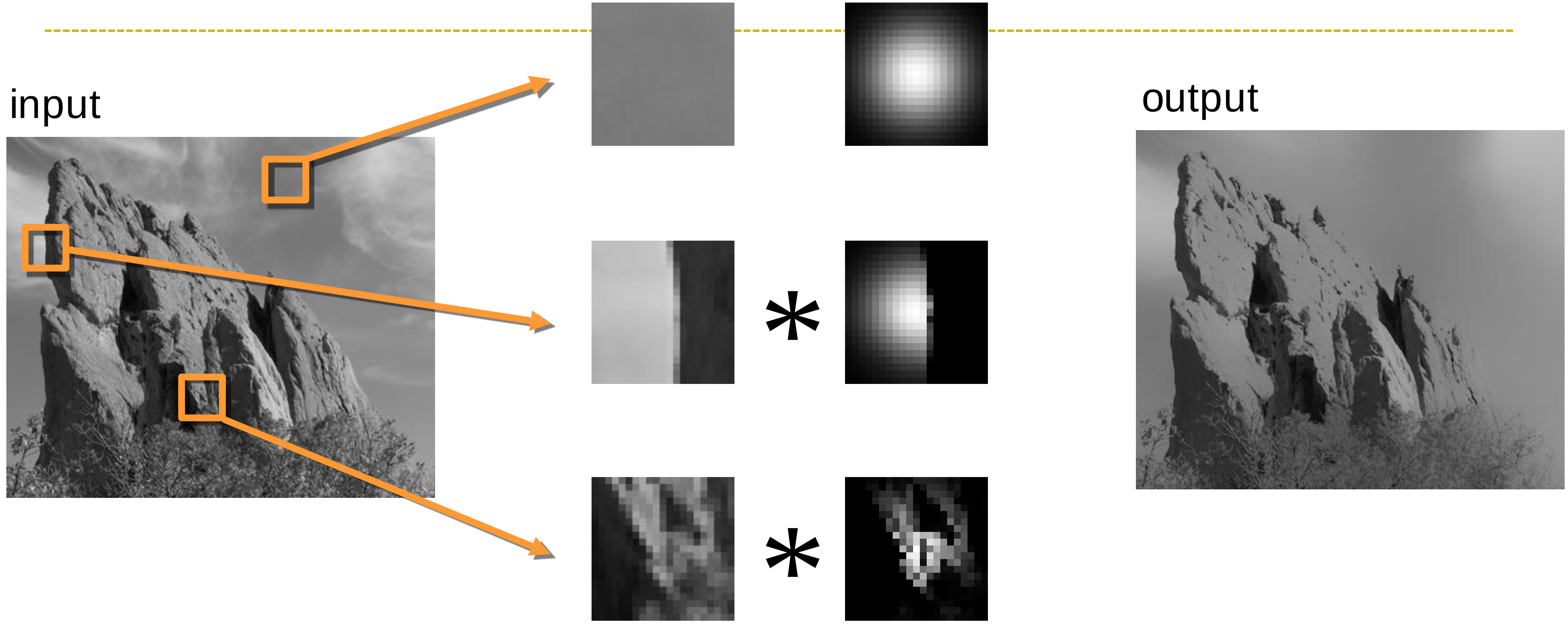
Usual Gaussian Filtering



Same Gaussian kernel everywhere.

Source: Sylvain Paris

Bilateral Filtering



The kernel shape depends on the image content.

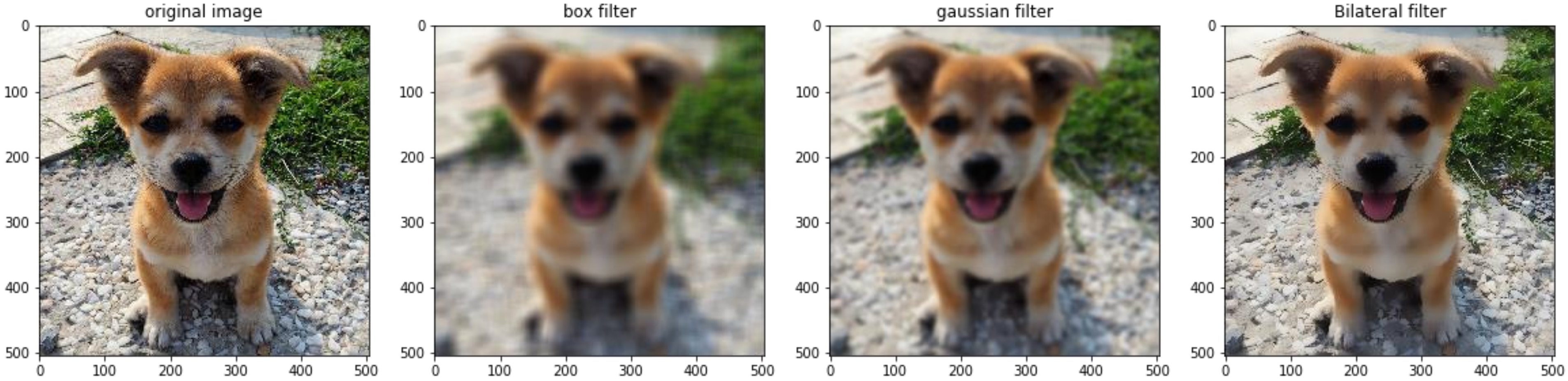
Iterating Bilateral Filter

$$I_{(n+1)} = BF\left[I_{(n)} \right]$$

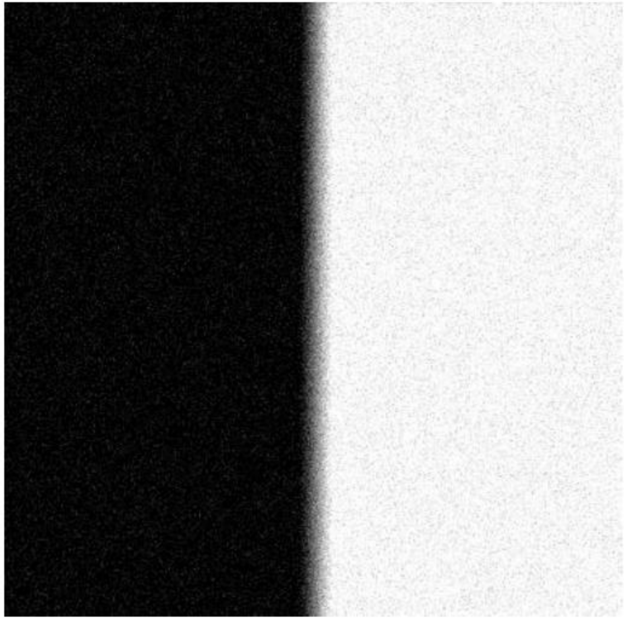
- Generate more piecewise-flat images
- Often not needed in computational photo.

Source: Sylvain Paris, “A Gentle Introduction to Bilateral Filtering and Its Applications”, SIGGRAPH 2007

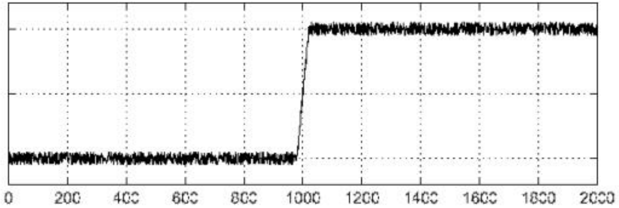
Fig. 2.4 Iterations: the bilateral filter can be applied iteratively, and the result progressively approximates a piecewise constant signal. This effect can help achieve a limited-palette, cartoon-like rendition of images [72]. Here, $\sigma_s = 8$ and $\sigma_r = 0.1$.


original image
box filter
gaussian filter
Bilateral filter
0
100
200
300
400
500

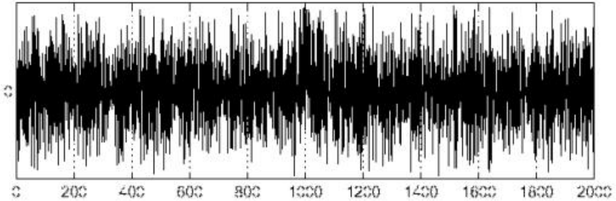
Effect of noise on derivatives



Noisy image

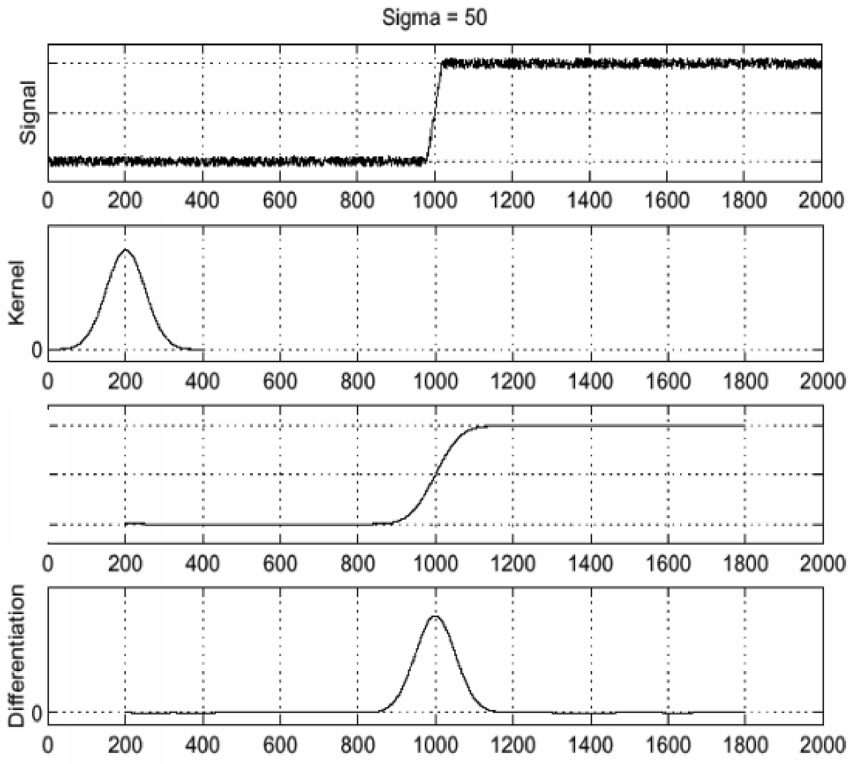


$I[x, y = j]$



$\frac{d}{dx} I[x, y = j]$

Solution: smooth first



f

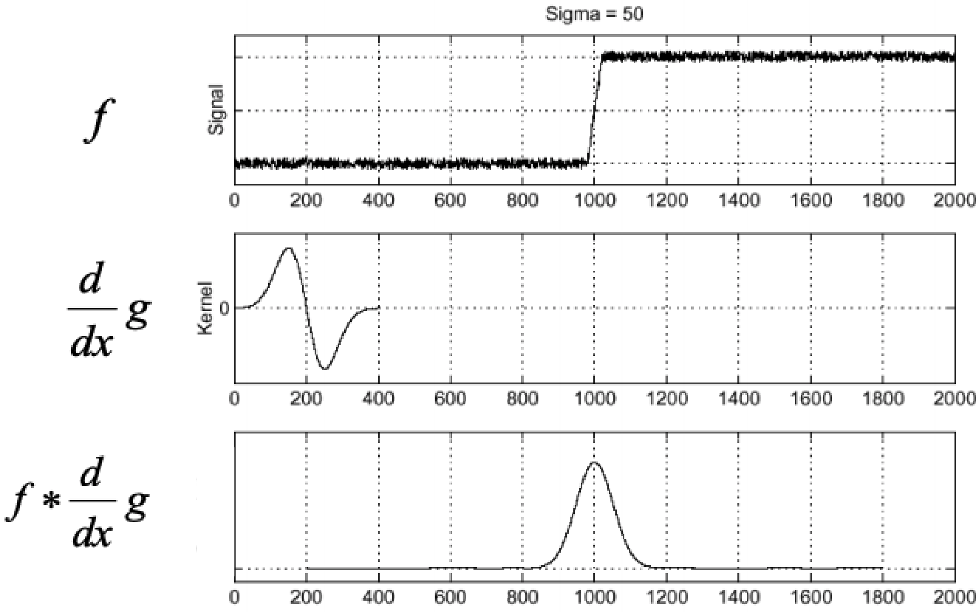
g

$f * g$

$\frac{d}{dx}(f * g)$

$$\frac{d}{dx}(f * g) = f * \frac{d}{dx} g$$

- This saves us one operation:



Source: S. Seitz

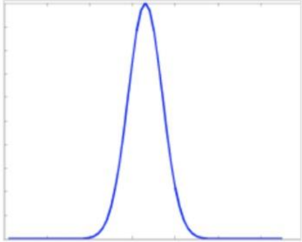
Other Important Filters

- Laplacian of Gaussian
  - Noise Suppression

Robert Collins
CSE486

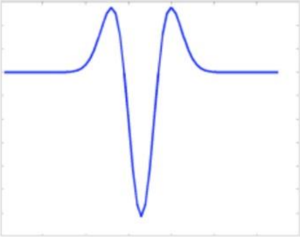
1D Gaussian and Derivatives

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



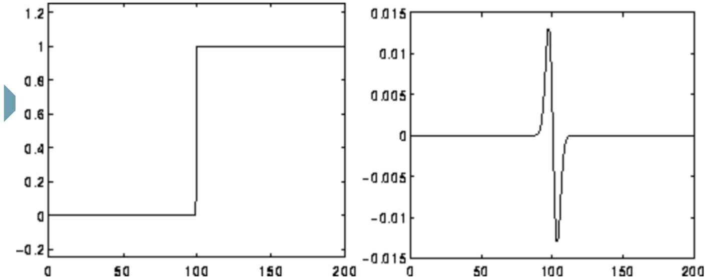
$$g'(x) = -\frac{1}{2\sigma^2} 2xe^{-\frac{x^2}{2\sigma^2}} = -\frac{x}{\sigma^2} e^{-\frac{x^2}{2\sigma^2}}$$

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



Other Important Filters

- Laplacian of Gaussian
  - Noise Suppression



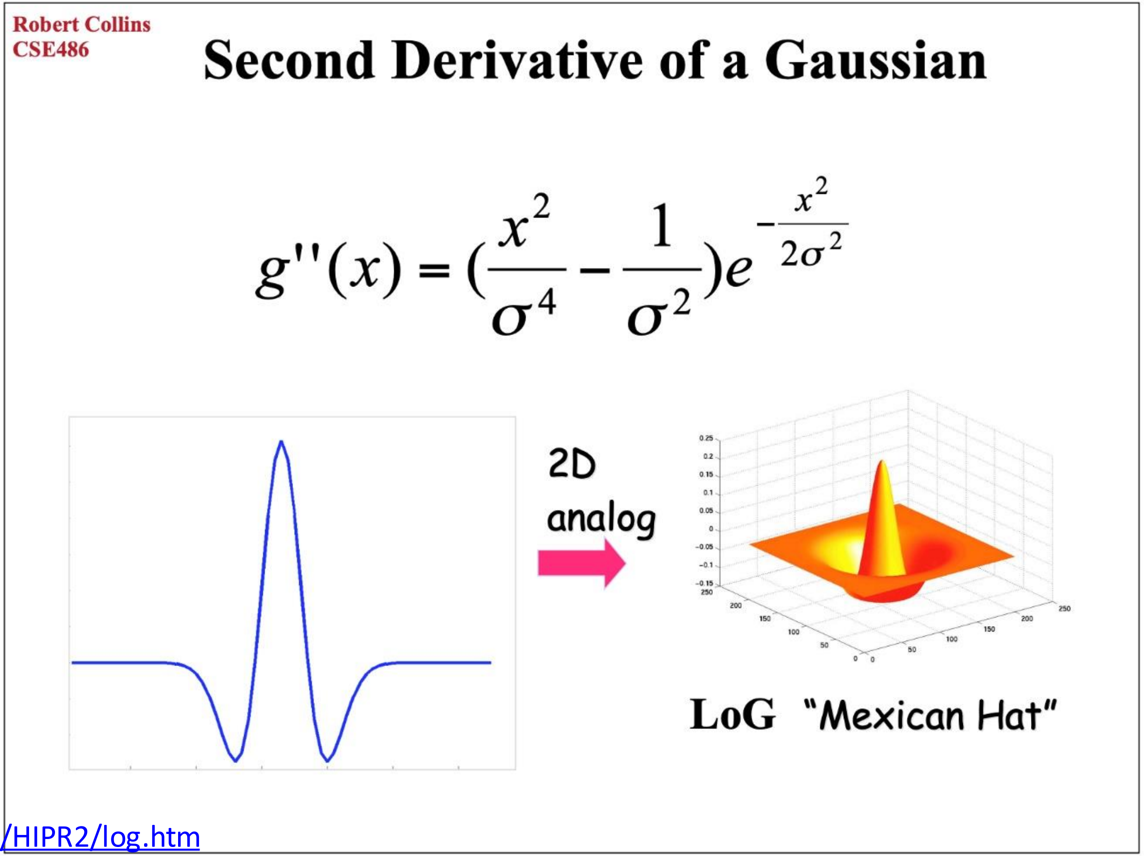


Image Source: https://homepages.inf.ed.ac.uk/rbf/HIPR2/log.htm

Other Important Filters

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

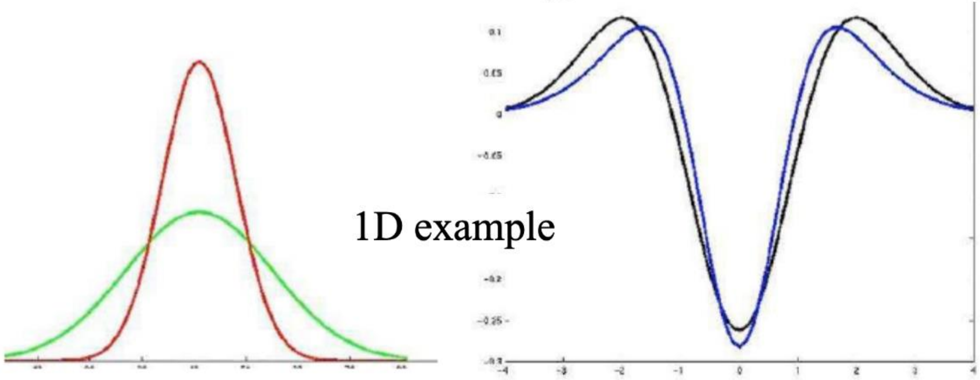
Robert Collins
CSE486

Efficient Implementation
Approximating LoG with DoG

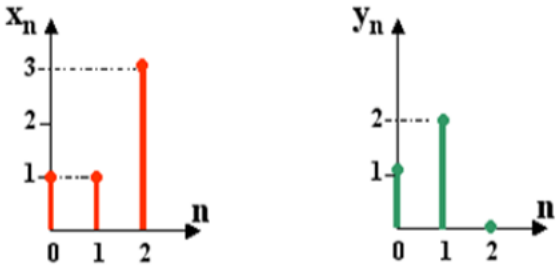
LoG can be approximate by a Difference of two Gaussians (DoG) at different scales

$$\nabla^2 G_\sigma \approx G_{\sigma_1} - G_{\sigma_2}$$

Best approximation when:
$\sigma_1 = \frac{\sigma}{\sqrt{2}},\ \sigma_2 = \sqrt{2}\sigma$



Linear System



A system T is **linear** if it satisfies the following two properties:

1) Scaling

$$x[n] \longrightarrow \boxed{T} \longrightarrow y[n] \quad \Rightarrow \quad \alpha\, x[n] \longrightarrow \boxed{T} \longrightarrow \alpha\, y[n]$$

2) Additivity

$$x_1[n] \longrightarrow \boxed{T} \longrightarrow y_1[n] \qquad x_2[n] \longrightarrow \boxed{T} \longrightarrow y_2[n]$$

$$\Rightarrow \quad x_1[n] + x_2[n] \longrightarrow \boxed{T} \longrightarrow y_1[n] + y_2[n]$$

‘Linear’ Spatial Filtering

Linear System

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

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

Convolution / Linear Filters

- Smoothing (Average, Gaussian)
- Edge Filters (Prewitt, Sobel, Laplacian)

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

$$\begin{array}{c|ccc} i \backslash j & -1 & 0 & 1 \\ \hline -1 & a & b & c \\ 0 & d & e & f \\ 1 & g & h & i \end{array}$$

120	190	140	150	200
17	21	30	8	27
89	123	150	73	56
10	178	140	150	18
190	14	76	69	87

x

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

=

	98			

Convolution vs Cross-Correlation

- Cross-correlation: operation of sliding the kernel/filter across the image and computing SOP

$$H^{\circ}I(x,y) = \sum_{i=-N}^{N}\sum_{j=-N}^{N} H(i,j)\cdot I(x+i,y+j) \qquad \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

- Convolution involves rotating the kernel/filter by 180° (flip rows and then flip columns), slide the kernel and compute SOP

$$H * I(x,y) = \sum_{i=-N}^{N}\sum_{j=-N}^{N} H(i,j)\cdot I(x-i,y-j) \qquad \begin{bmatrix} 9 & 8 & 7 \\ 6 & 5 & 4 \\ 3 & 2 & 1 \end{bmatrix}$$

Convolution establishes link between operations in the frequency domain and the effect of spatial filters

References

- GW Chapter – 3.4
- Convolution:

- http://www.songho.ca/dsp/convolution/convolution.html
- http://www.ceri.memphis.edu/people/smalley/ESCI7355/Ch6_Linear_Systems_Conv.pdf
- https://towardsdatascience.com/convolution-vs-correlation-af868b6b4fb5

Image Processing – Two Paradigms

- Directly manipulating pixels in spatial domain

- Manipulating in transform domain

Spatial vs. Transform Domain Processing

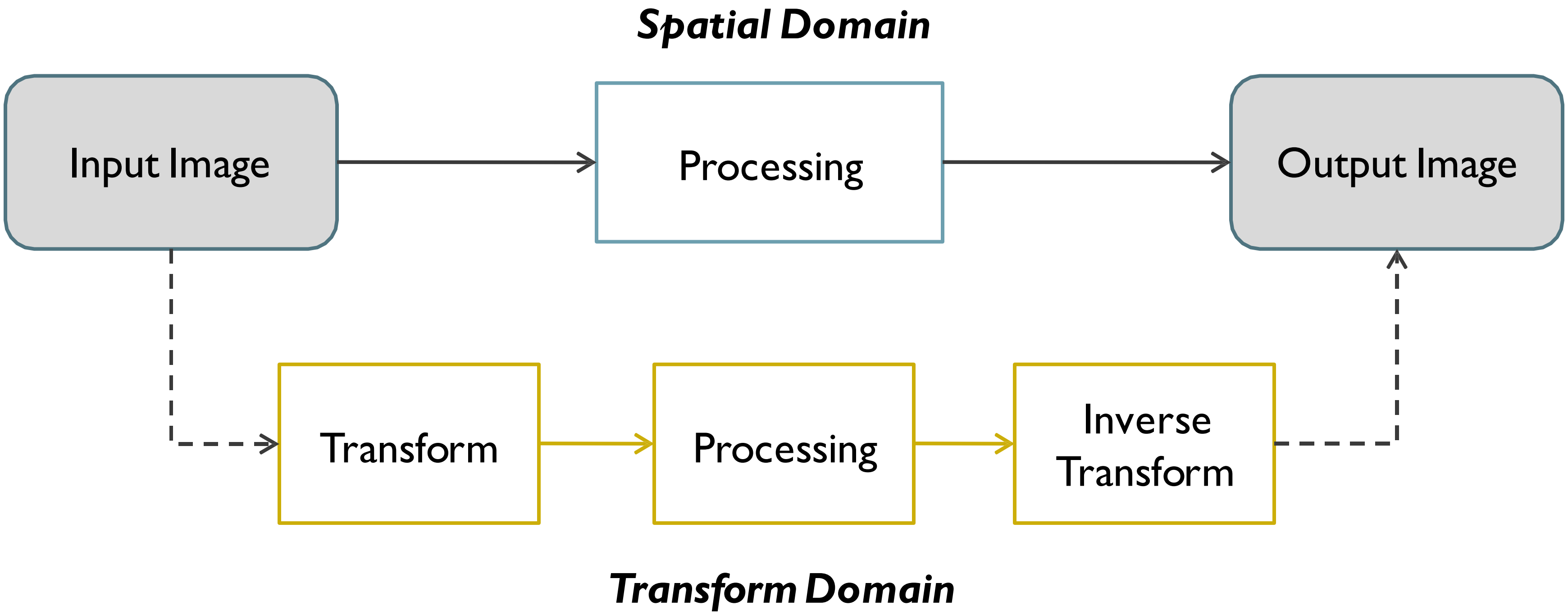
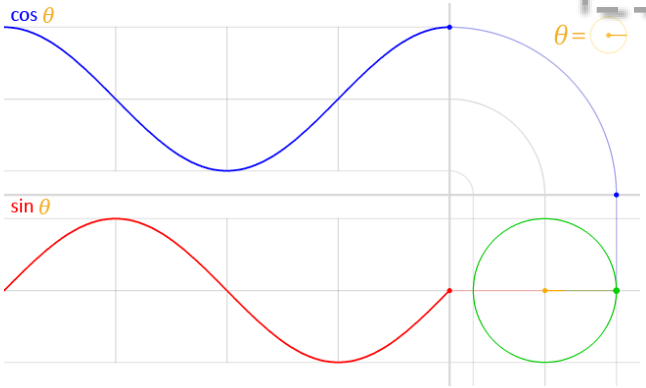
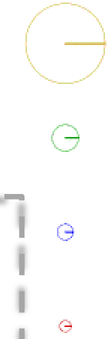


Image Enhancement in Frequency Domain – Preliminary Concepts

Periodic Signals

- Periodic → **Frequency** of occurrence
  - Repetitions/<Unit> (cycles/sec = Hz)





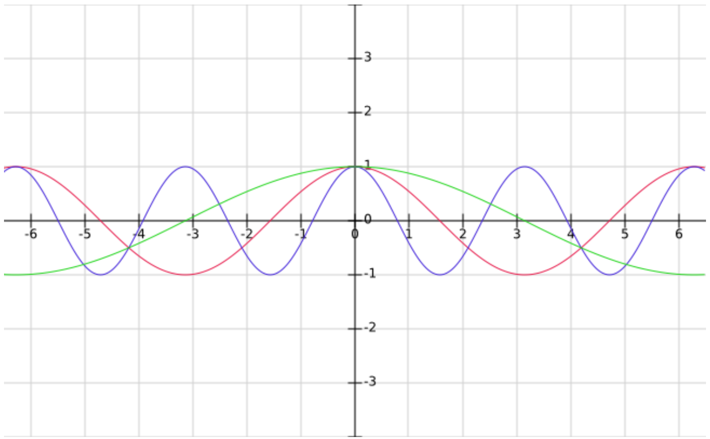
$$x(t) = A\cos(\omega t) = A\cos(\boxed{2\pi} f t) = A\cos(\frac{2\pi}{T} t)$$

Angular frequency

Fundamental Period

Simple periodic signals

- $x(t) = A\cos(t)$
- $x(t) = A\cos(2t)$
- $x(t) = A\cos(t/2)$

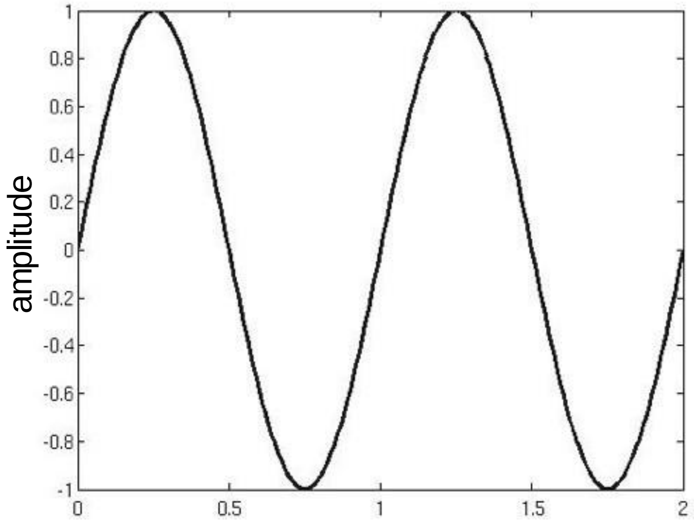


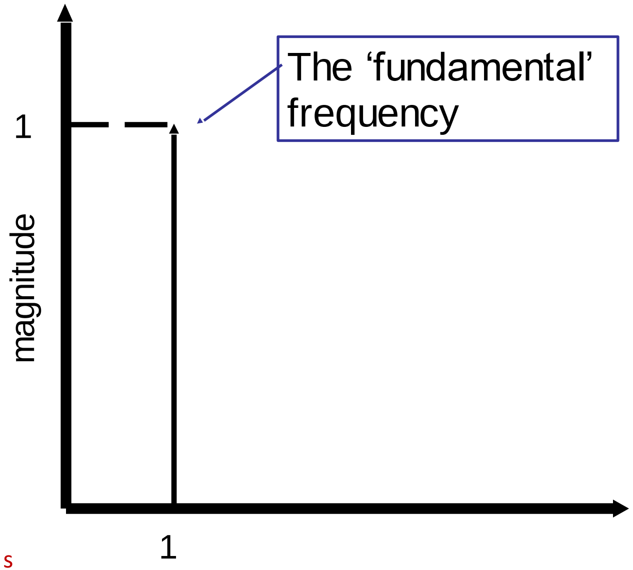
- $x(t) = A\cos(\omega t) = A\, cos(2\pi f t) = A\, cos(\frac{2\pi}{T} t)$

Angular frequency

Signal and Frequency Domains

$$y = \sin(2\pi t)$$





NOTE: t=0 is time from where signal is considered

time/length

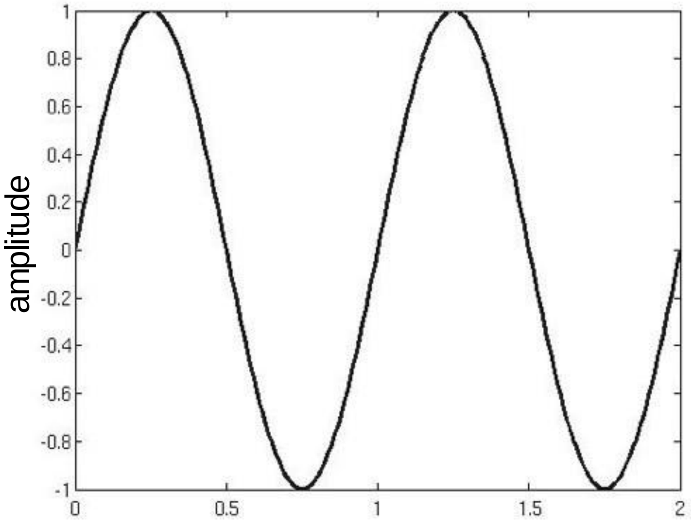
NOTE: t is bounded only for physical signals

Frequency (# of 'cycles' per unit time)

Signal domain ⟷ **Frequency domain**

Signal and Frequency Domains

$$y = \sin(2\pi s)$$

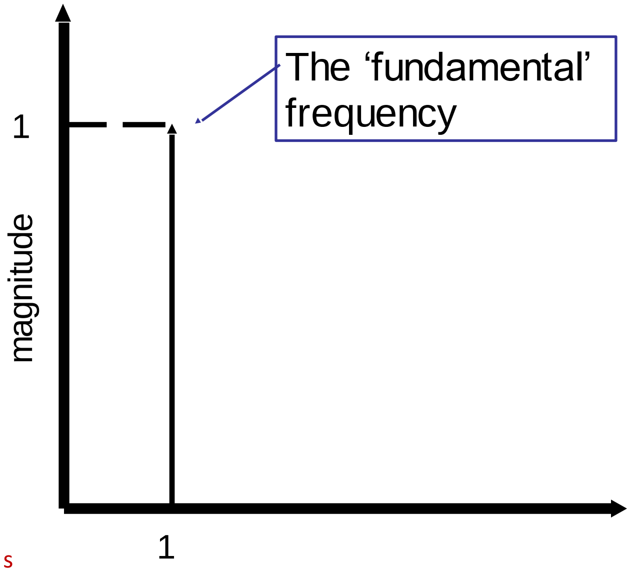


NOTE: s=0 is location from where signal is considered

spatial distance (s)

Signal domain

NOTE: s is bounded only for physical signals



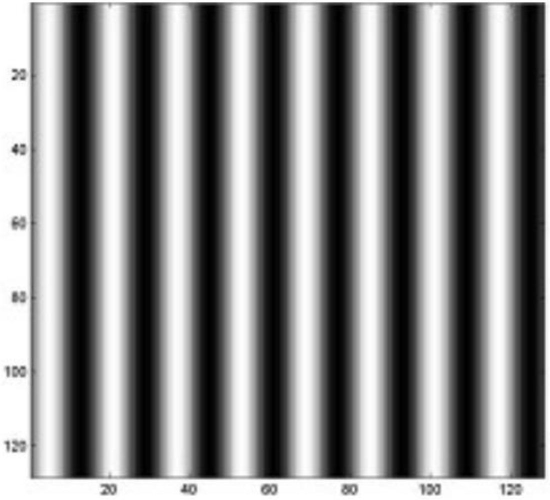
Frequency (# of cycles per unit distance)

Frequency domain

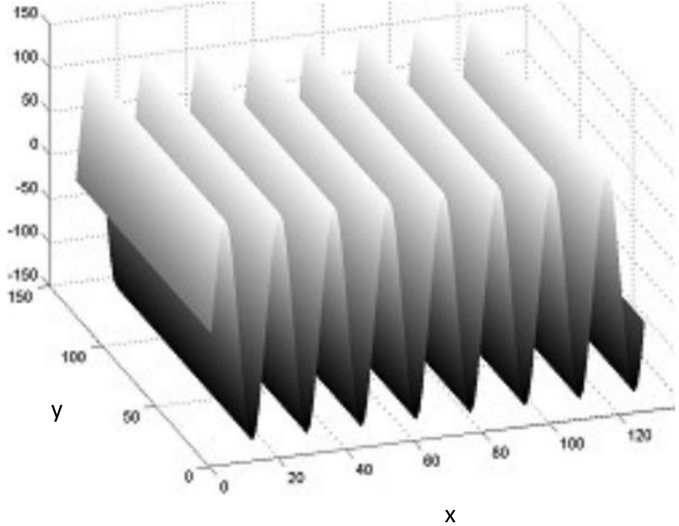
Periodic Images

$I(x, y) = 128\ sin(2\pi x/16)$

128 x 128 spatial signal

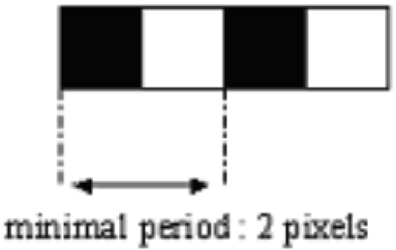


Sinusoid pattern repeats every 16 pixels
f = 1/16 cycles/pixel



Periodic Images

Spatial period = Minimal # of pixels between two identical patterns in a “periodic” image



$$\Rightarrow \qquad \nu_{max} = \frac{1}{\text{minimal period}} = \frac{1}{2}$$

Spatial frequency representation

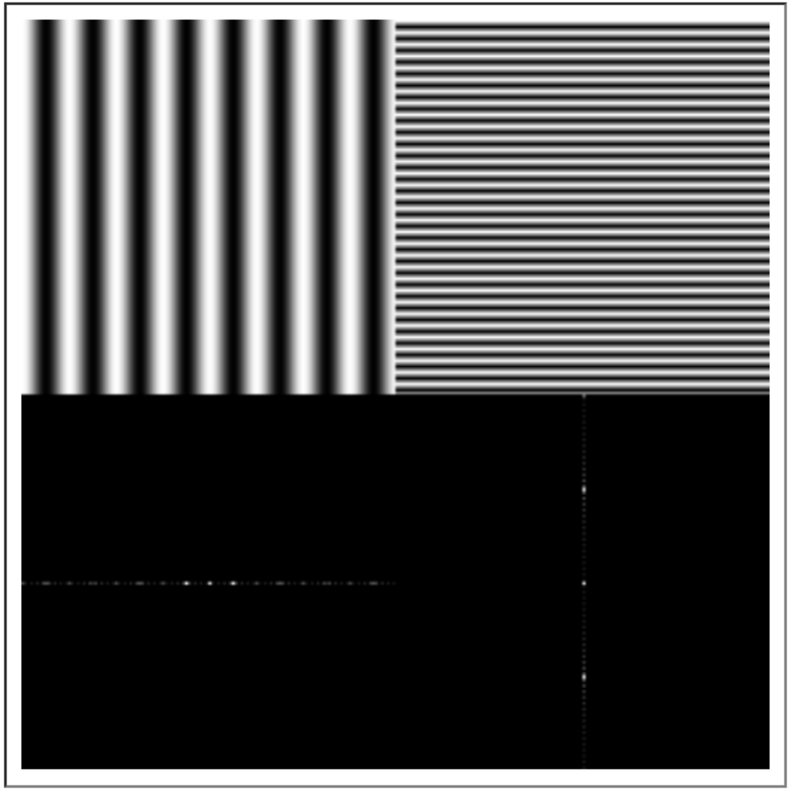
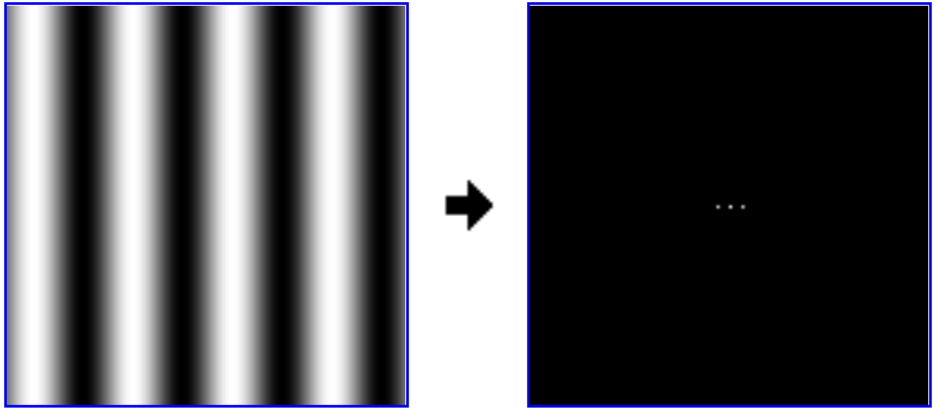
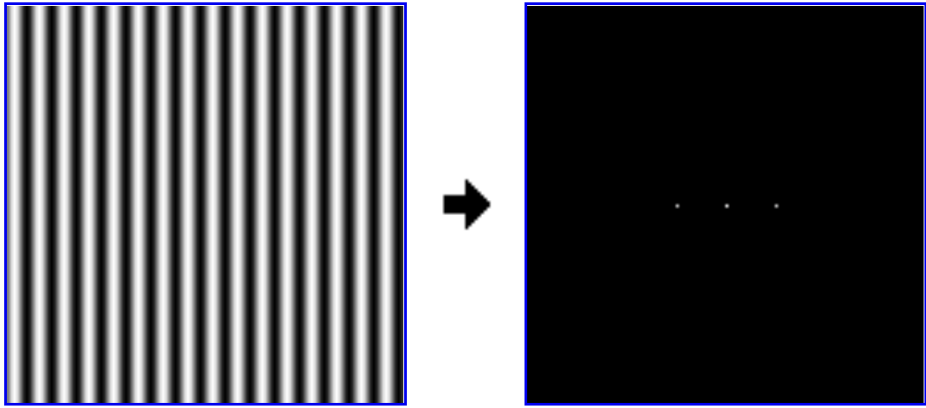


Image Courtesy: https://www.cs.unm.edu/~brayer/vision/fourier.html

Spatial frequency representation



Sine wave with 4 cycles



Sine wave with 16 cycles

Image Courtesy: https://legacy.imagemagick.org/Usage/fourier/