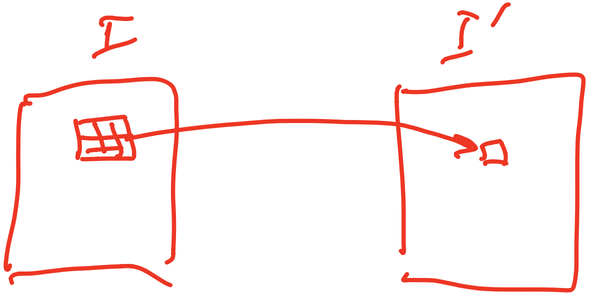
Mean / box filter

A kernel with all values h(x) or h(x,y) equal gives a box filter.



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

$f[x,y]$

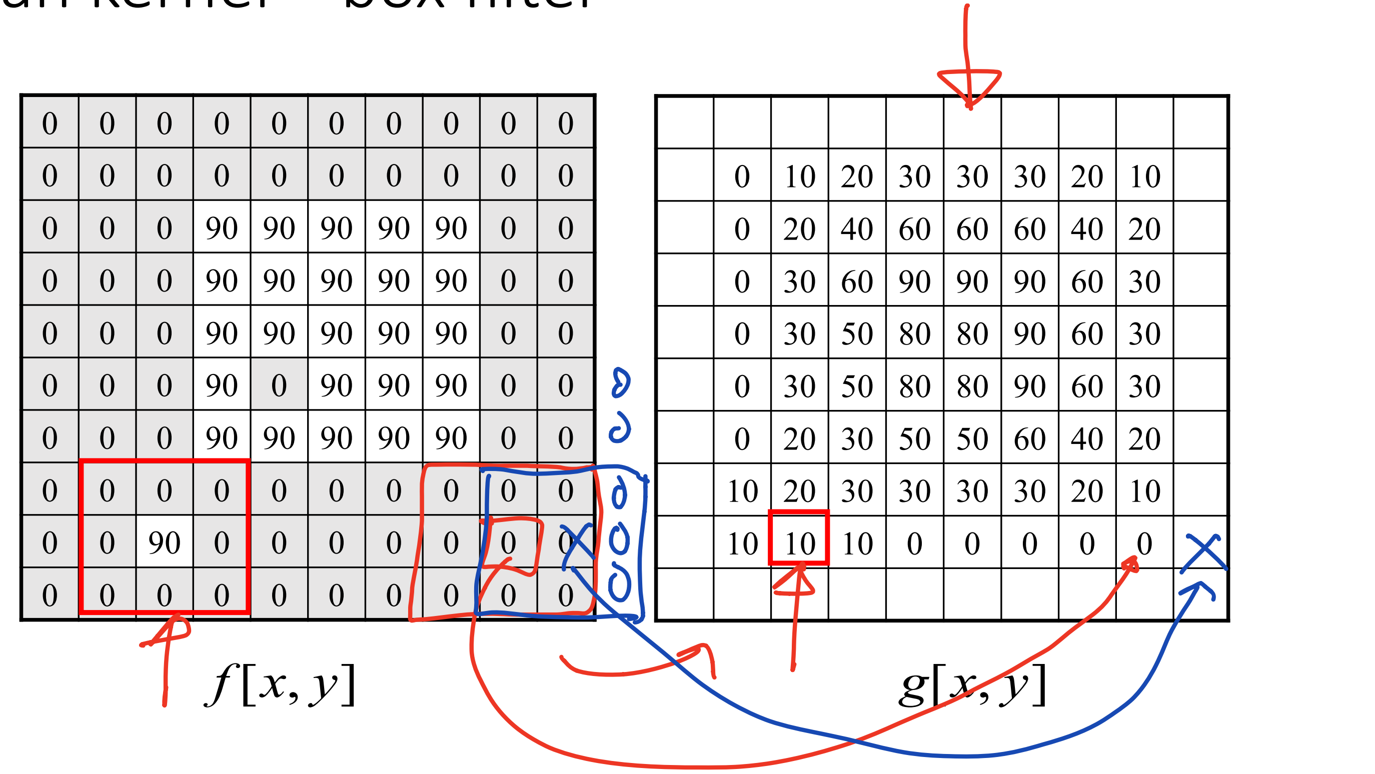
1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

$h[u,v]$

If we want the output to have approximately the same amount of energy, the kernel values should sum to one.

Equal weight to all pixels within the neighborhood

Mean kernel – box filter



$f[x, y]$

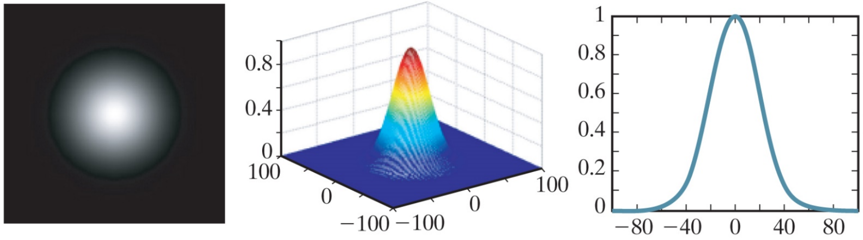
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$g[x, y]$

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

(5.2) Gaussian filters

Figure 5.3 A Gaussian is a bell curve. From left to right: The 2D isotropic Gaussian viewed as an image where the gray level of each pixel is proportional to the value of the Gaussian function at that point, the 2D isotropic Gaussian viewed as a surface in 3D, and the 1D Gaussian function (or, equivalently, a slice through the 2D Gaussian function, obtained by intersecting it with a vertical plane).

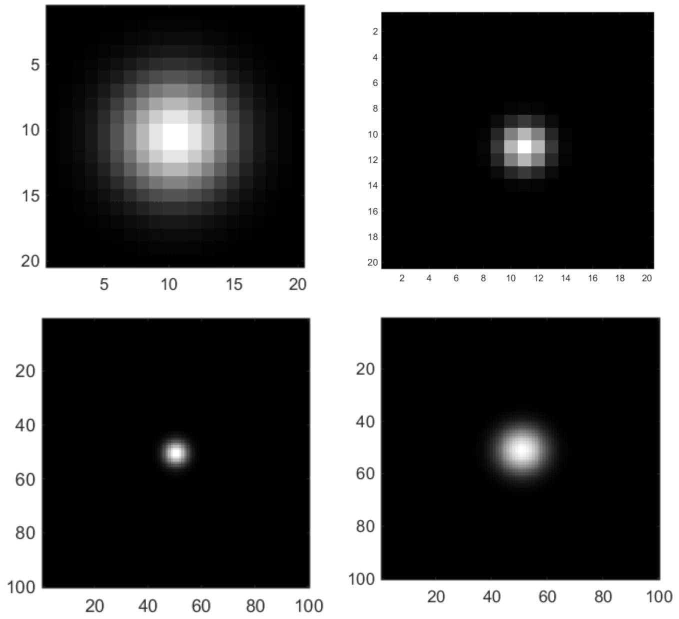


Stan Birchfield

$$\boxed{gauss_{\sigma^2}(x) \equiv \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-x^2}{2\sigma^2}\right)}$$

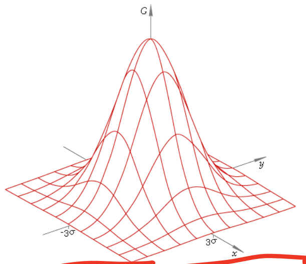
Convolving a box filter with itself yields an approximation to a Gaussian kernel.

Gaussian filters



Space domain

Frequency domain (magnitude of DFT)



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

Gaussian function in space domain becomes gaussian in frequency domain.

Gaussian filtering

A **Gaussian kernel** gives less weight to pixels further from the center of the window

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

$f[x, y]$

$$\frac{1}{16} \cdot \begin{bmatrix} 1 & 2 & 1 \\ 2 & \boxed{4} & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

$h[u, v]$

11.25	22.5

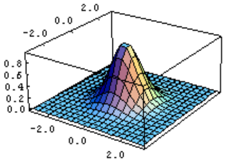
→

10	10

from box

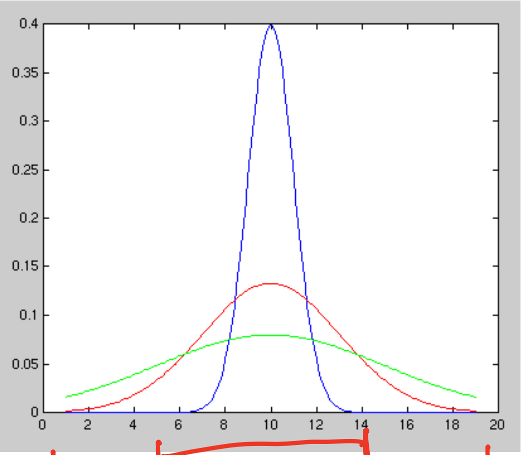
is a discrete approximation of a Gaussian function:

$$h(u, v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{\sigma^2}}$$



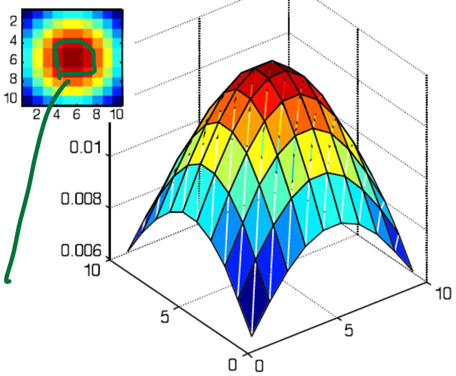
Note about Finite Kernel Support

Gaussian function has infinite support. In discrete filtering, we want a finite kernel



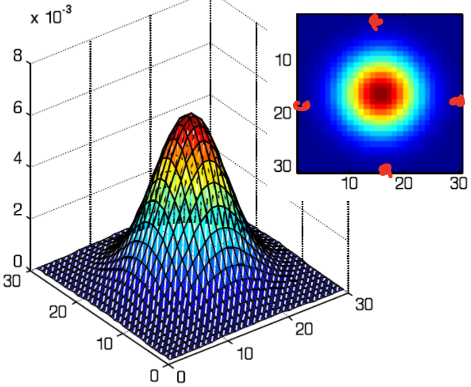
Shape of Gaussian with different σ

10x10



σ = 5 with 10x10 kernel

30x30

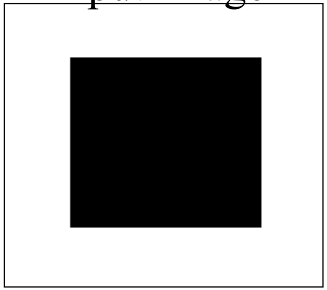


σ = 5 with 30x30 kernel

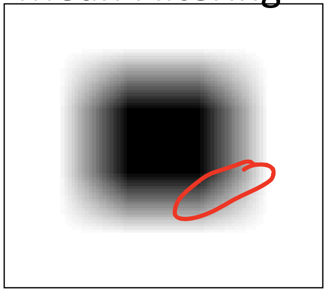
if σ large and support too small
⇒ almost box filter !

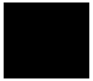
Box filter vs. Gauss filter

Input image



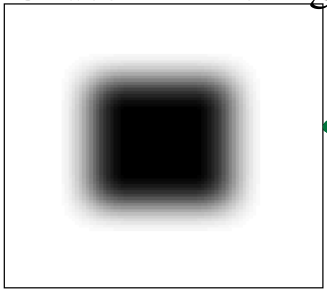
Mean Filtering

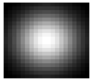




Mean filter, 20x20

Gaussian filtering





Gaussian filter, 20x20, σ = 5

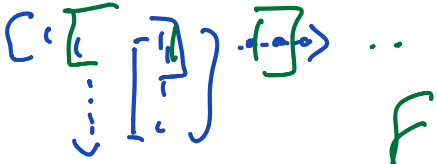
Separable filters

matrix

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

- Gaussian filters are separable.
- If the 2D filter kernel can be written as convolution of 2 1D filter kernels, then the 2D filter is separable. More computational effective to implement!
- If kernel viewed as matrix, separability occurs if all rows and columns are linearly dependent. In other words, 2D kernel has rank = 1. In this case the 2D kernel can be viewed as outer product of the two 1D kernels when viewed as vectors.

$$\frac{1}{3}\begin{bmatrix}1\\1\\1\end{bmatrix} \circledast \frac{1}{3}\begin{bmatrix}1&1&1\end{bmatrix}=\frac{1}{9}\cdot\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}$$



$$\begin{bmatrix}\begin{bmatrix}1\\1\\1\end{bmatrix}&1&1\\ &\vdots& \end{bmatrix}$$

convolution

2D Convolution with separable kernel

- When the kernel is separable there is a compact, elegant notation for the 2D convolution:

$$\boxed{\mathbf{I}' = \mathbf{G}_h \mathbf{I} \mathbf{G}_v^{\top}}$$

Where $\mathbf{G}_h$ is the convolution matrix constructed from the 1D kernel $\mathbf{g}_h$.

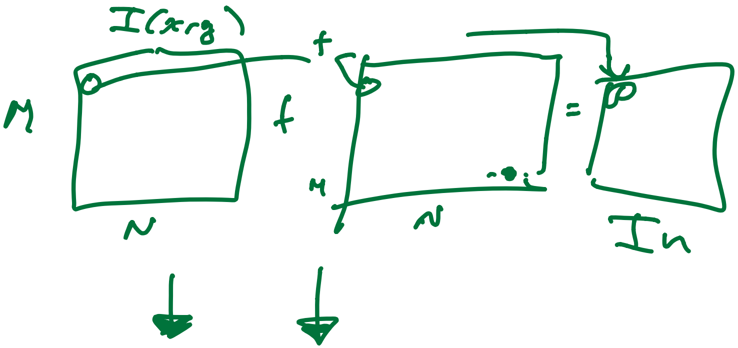
Instead of doing a 2D convolution we do 2 X 1D convolution:

→ First convolve over the columns. (or rows)

→ Then convolve results of first convolution ower the rows. (or columns)

$$\underset{\sim}{g} = [1 \quad 2 \quad 1]$$

$$\underset{\sim}{G} \begin{bmatrix} 1 & 2 & 1 & 0 & 0 & \cdots & & \\ 0 & 1 & 2 & 1 & 0 & 0 & \cdots & \\ 0 & 0 & 1 & 2 & 1 & 0 & \cdots & \\ 0 & 0 & 0 & 1 & 2 & 1 & 0 & \cdots \\ \vdots & & & & & & & \end{bmatrix} \left(\underset{\sim}{I} \right)$$



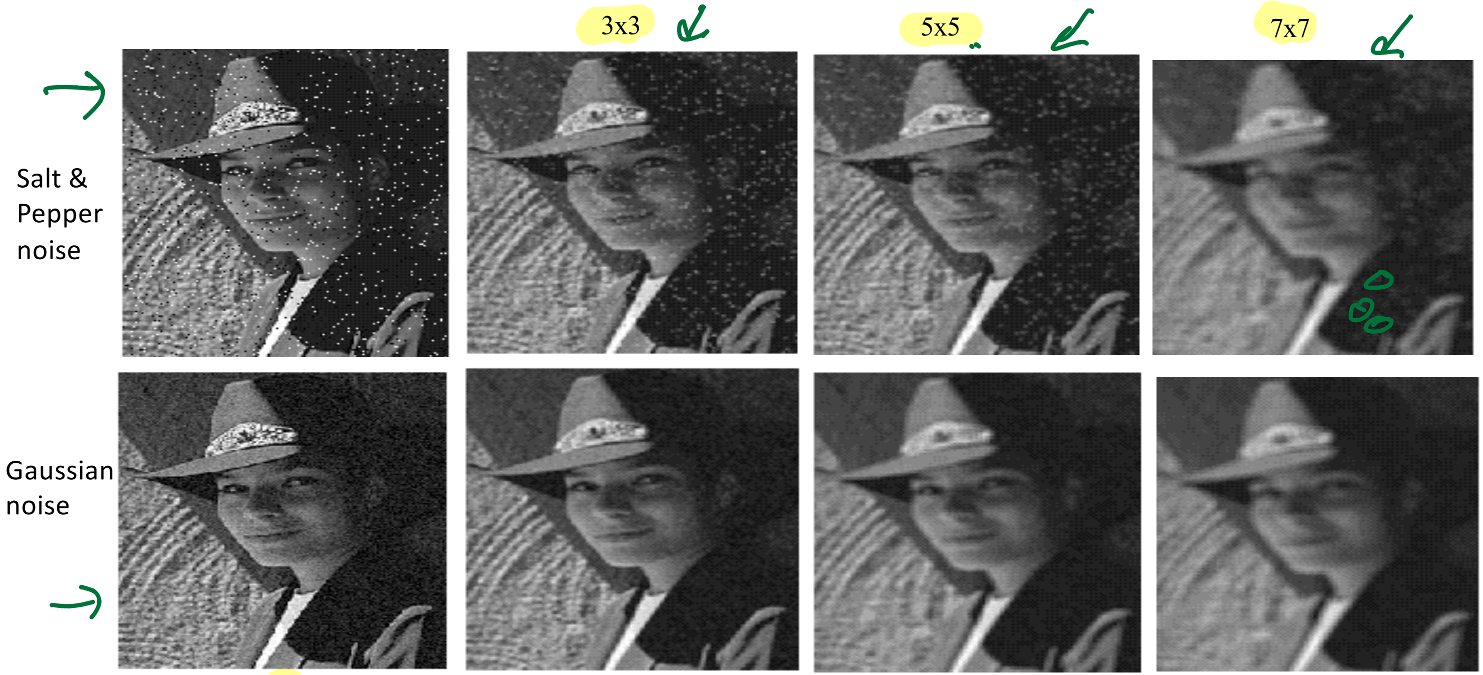
Noise

- Additive noise: In(x,y) = I(x,y) +n(x,y)
- Additive gaussian noise: the noise follows a random gaussian distribution with some sigma and typically mean=0.
- **Salt-and-pepper noise**: each pixel is set to either the minimum (“pepper”) or maximum (“salt”) possible gray level, or it remains unchanged:

$$I'(x,y) = \begin{cases} 0 & \text{if } 0 \le \xi < p \\ 255 & \text{if } p \le \xi < p+q \\ I(x,y) & \text{otherwise} \end{cases} \qquad \xi \sim U(0,1)$$

Effect of mean /box filtering

$\frac{1}{9}$.



Side effect - blur

Additiv Gaussian noise
Linear Gaussian filtering