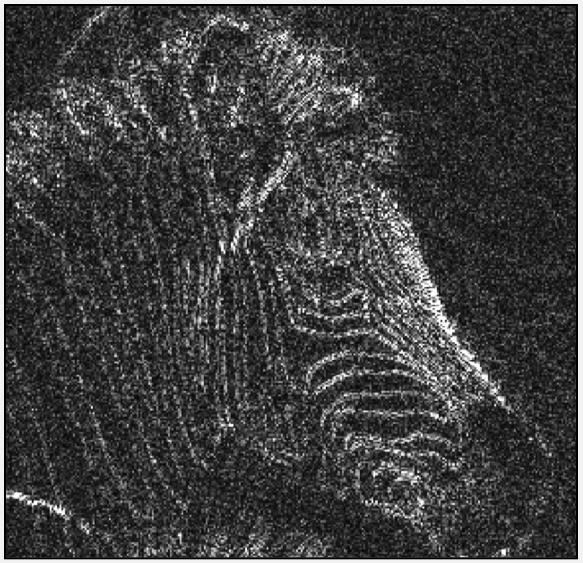
Department of Computer Science
University of Bristol

COMS30030 - Image Processing and Computer Vision



Week 02

Filtering Images

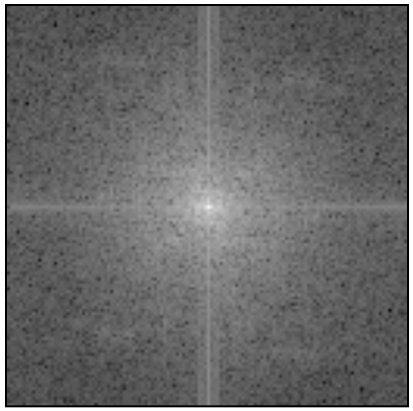
Majid Mirmehdi | majid@cs.bris.ac.uk

Image Transforms

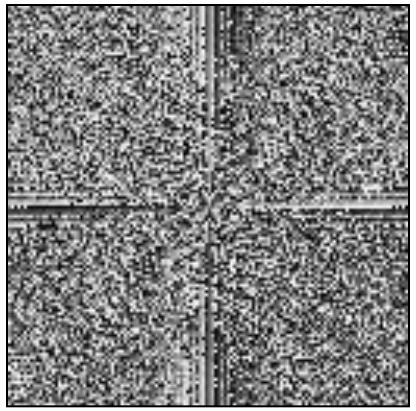
- **Image Transform** = derivation of a ***new representation*** of the input data (e.g. by re-encoding it in another [parameter] space, e.g. Fourier, DCT, Wavelet, Haar, etc.)
- Image Transforms are classic processing techniques, used in filtering, compression, feature extraction, and much more

image f (x, y)

→→→ *Fourier Transform*



power spectrum



phase spectrum

Convolution

Convolution = A mathematical operation on two functions (f and g) that produces a third function ($h=f*g$), representing how the shape of one is modified by the other.

(Wikipedia)

$$f * g = \int_{-\infty}^{\infty} f(x-t)g(x)\,\partial t$$

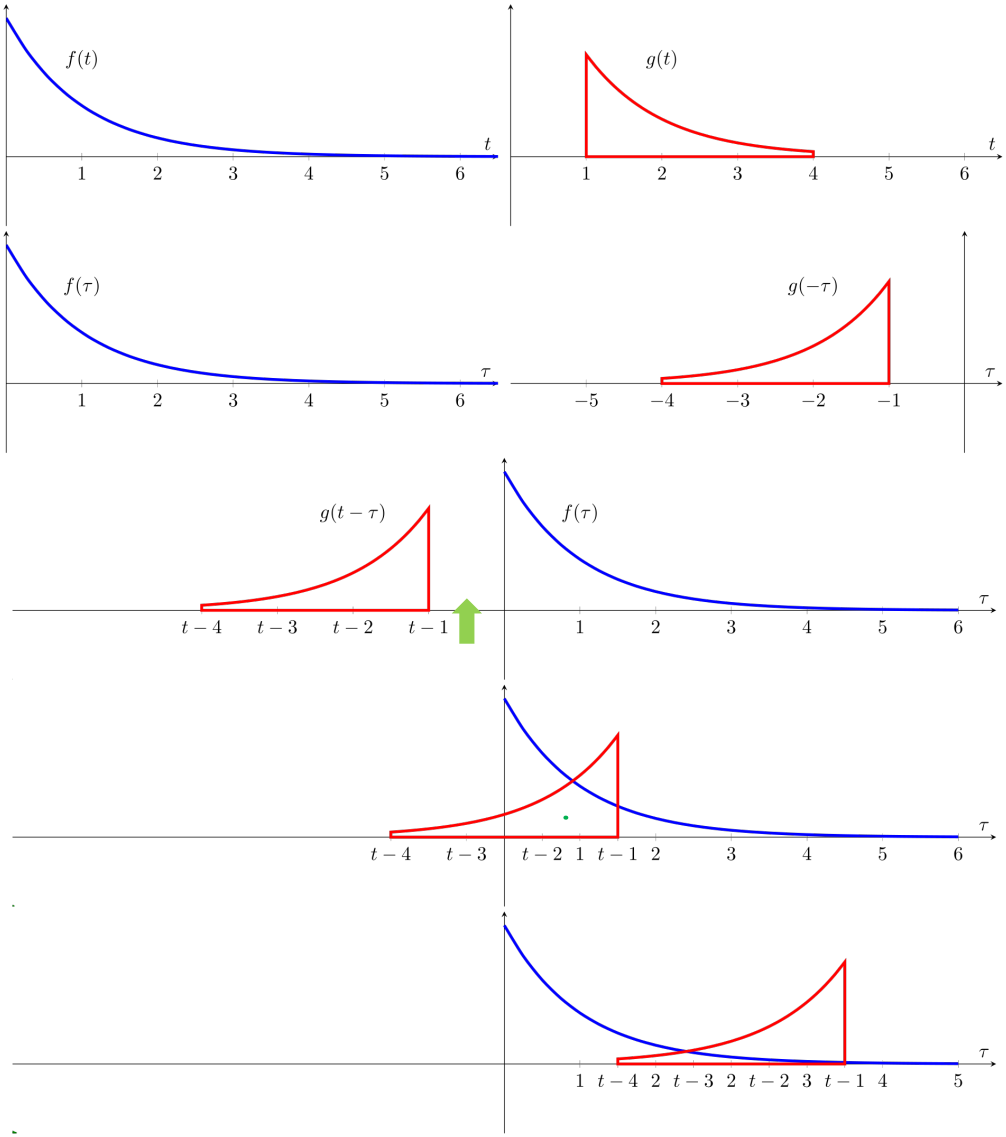
Kernel

Signal

- Used for filtering images in the spatial domain
- Application in other parameter spaces, e.g. frequency domain
- Fundamental to Convolutional Neural Nets for deep Learning
- much, much more

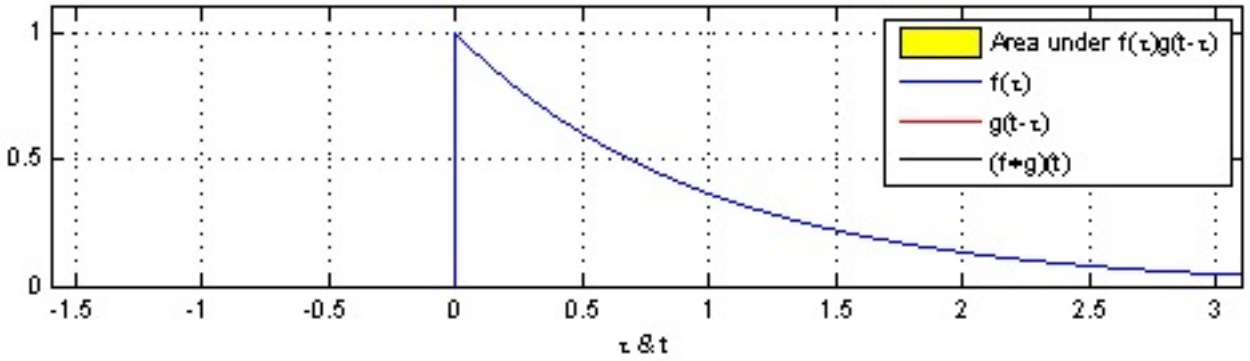
An Intuitive look at Convolution

$$f * g = \int_{-\infty}^{\infty} f(x-t)g(x)\,\partial t$$

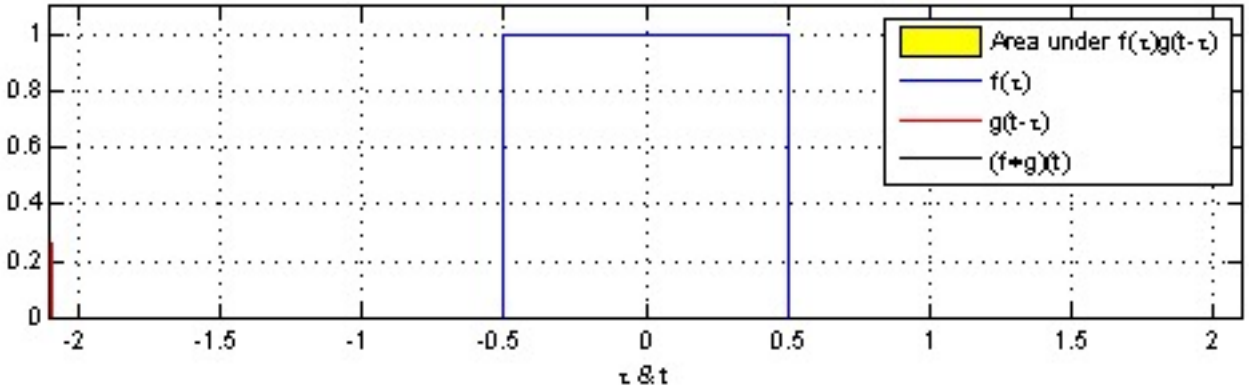


By Krishnavedala

An Intuitive look at Convolution



$$f * g = \int_{-\infty}^{\infty} f(x-t)g(x)\,\partial t$$

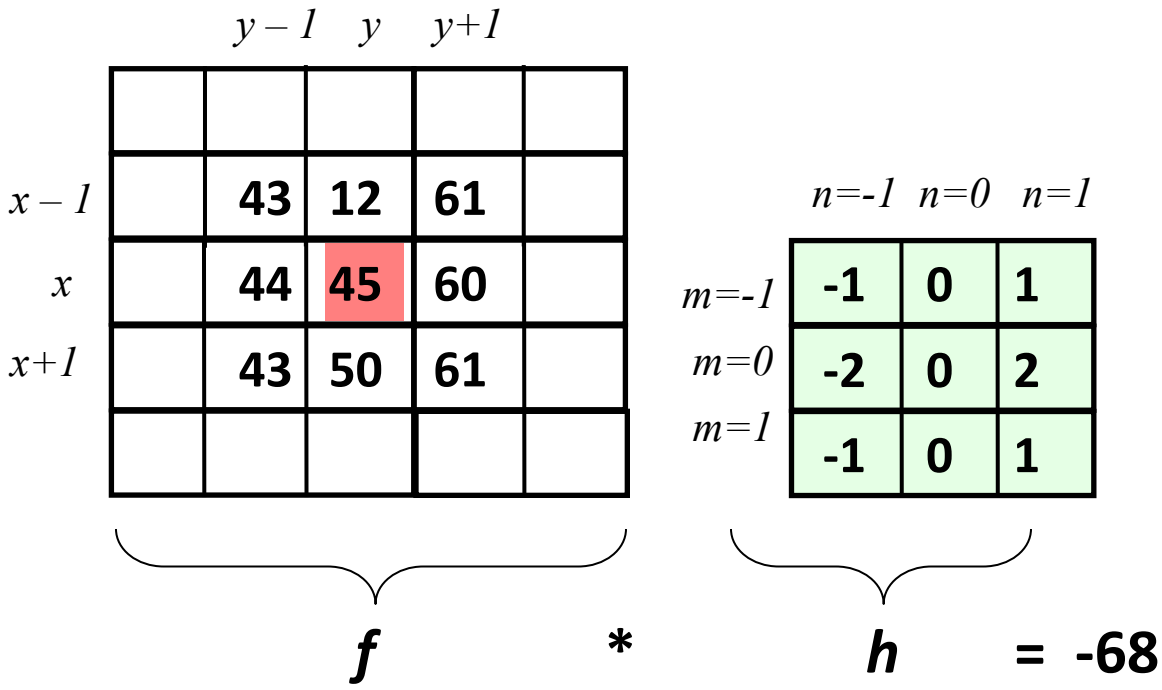


Animations Source: Brian Amberg

2D Discrete Convolution

- The discrete version of 2D convolution is defined as:

$$g(x,y) = \sum_m \sum_n f(x-m, y-n)h(m,n)$$



$$f(x+1,y+1)h(-1,-1)$$
$$+ f(x+1,y)h(-1,0)$$
$$+ f(x+1,y-1)h(-1,1)$$
$$+ f(x,y+1)h(0,-1)$$
$$+ f(x,y)h(0,0)$$
$$+ f(x,y-1)h(0,1)$$
$$+ f(x-1,y+1)h(1,-1)$$
$$+ f(x-1,y)h(1,0)$$
$$+ f(x-1,y-1)h(1,1)$$

2D Discrete Correlation

- The discrete version of 2D correlation is defined as:

$$g(x,y) = \sum_m \sum_n f(x+m, y+n)h(m,n)$$

	n=-1	n=0	n=1
m=-1	-1	0	1
m=0	-2	0	2
m=1	-1	0	1

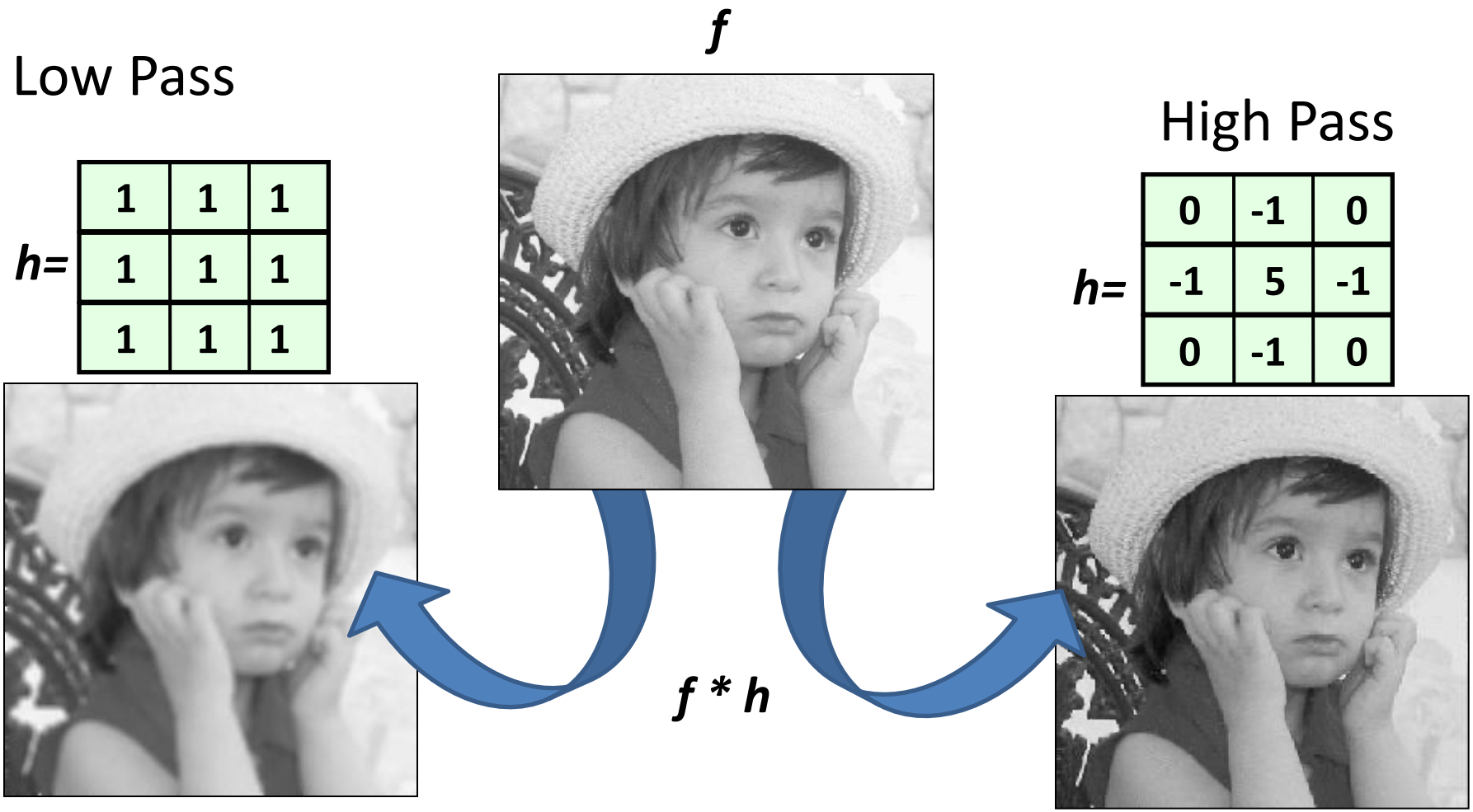
	y − 1	y	y+1
x − 1	43	12	61
x	44	45	60
x+1	43	50	61

68

Correlation ≡ Convolution when kernel is symmetric under 180° rotation, e.g.

1	2	1

Spatial Low/High Pass Filtering



Normalising the Convolution Result!

The weights will affect pixel values and the result could fall outside the 0-255 range.

- If all weights are positive:
  - Normalise them such that they sum to 1.
- If there are both positive and negative weights:
  - They should sum to 0 (but not always!).

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

Smooth

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

3x3 Gaussian Blur

$$\frac{-1}{256}\begin{bmatrix}1 & 4 & 6 & 4 & 1\\ 4 & 16 & 24 & 16 & 4\\ 6 & 24 & -476 & 24 & 6\\ 4 & 16 & 24 & 16 & 4\\ 1 & 4 & 6 & 4 & 1\end{bmatrix}$$

Unsharp Masking

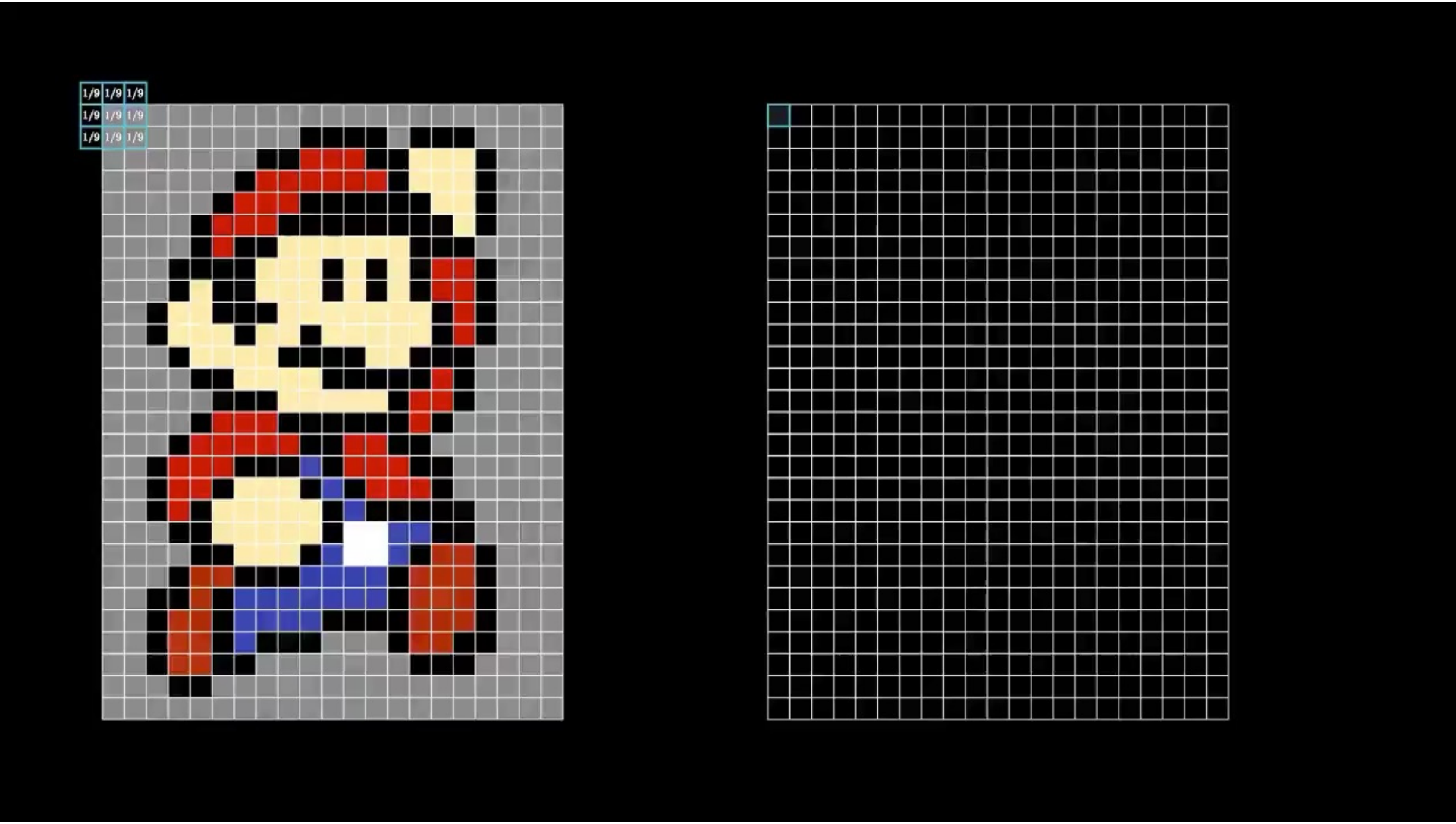
$$\begin{bmatrix}0 & -1 & 0\\ -1 & 5 & -1\\ 0 & -1 & 0\end{bmatrix}$$

Sharpen

$$\begin{bmatrix}-1 & -1 & -1\\ -1 & 8 & -1\\ -1 & -1 & -1\end{bmatrix}$$

Basic Edge Detector

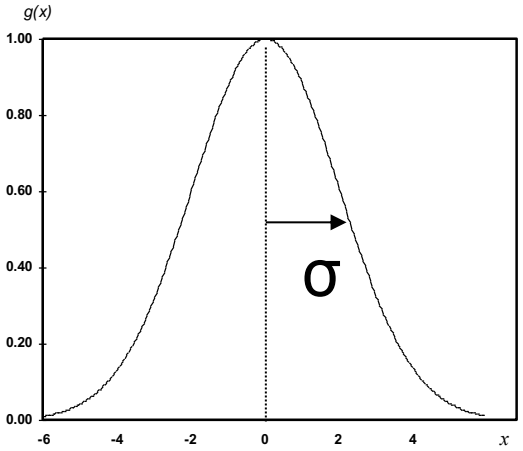
Convolution illustrated



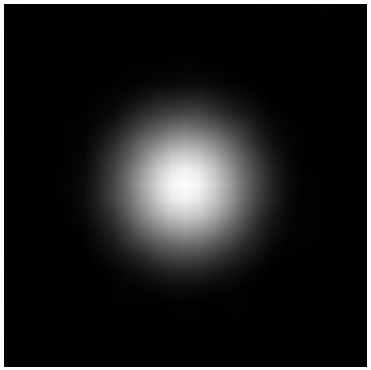
Animations Source: Grant Sanderson

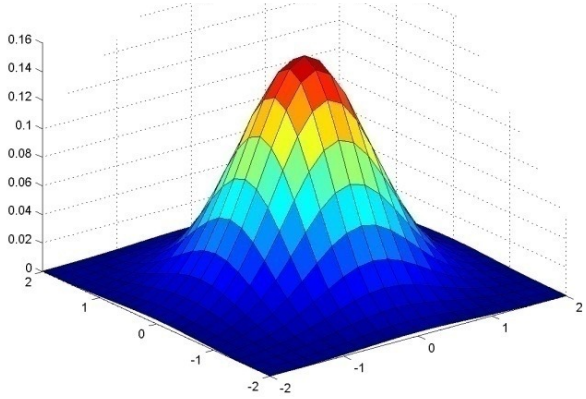
Gaussian Filter Kernel

The *Gaussian* filter is very commonly used in image processing. The parameter σ determines the width of the filter and hence the amount of smoothing.



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



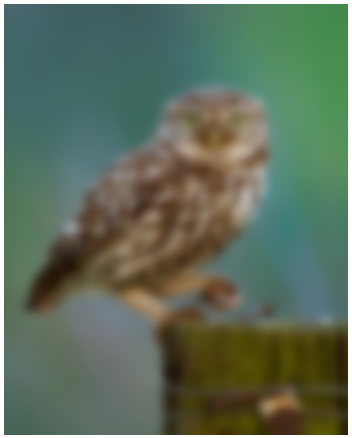


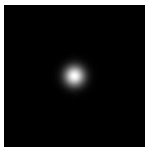
$$g(x,y) = \frac{1}{2\pi\sigma} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Applying the Gaussian filter

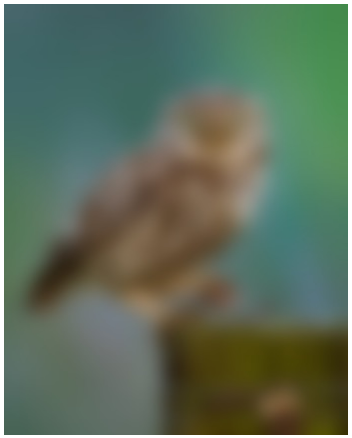
Original

$\sigma = 1$

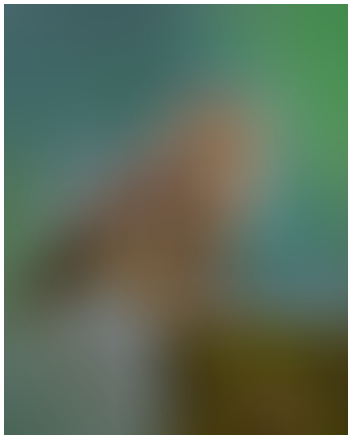


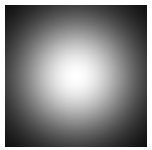


$\sigma = 5$



$\sigma = 10$



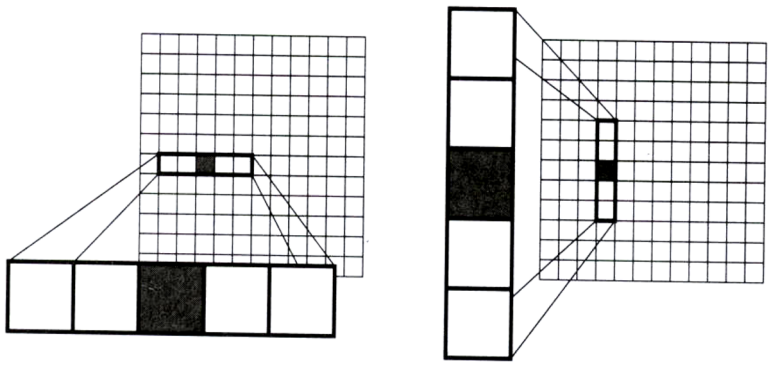


$\sigma = 30$

Source: Noah Snavely

Is the Gaussian Filter Computationally Expensive?

Filtering with a 2D Gaussian can be achieved faster using two 1D Gaussian filters! This is because the linear Gaussian kernel is *separable*.



$$K = \frac{1}{256}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix} = \frac{1}{16}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \end{bmatrix} * \frac{1}{16}\begin{bmatrix} 1 \\ 4 \\ 6 \\ 4 \\ 1 \end{bmatrix}$$

Filtering twice with a $\boldsymbol{m}\ \boldsymbol{x}\ \boldsymbol{n}$ Gaussian is equivalent to filtering with a $(\boldsymbol{m}+\boldsymbol{n}-\boldsymbol{1})\ \boldsymbol{x}\ (\boldsymbol{m}+\boldsymbol{n}-\boldsymbol{1})$ filter:

→ a significant reduction in computations!

Sources from fiveko.com and E.G.M. Petrakis

Separability of the Gaussian Filter

Separability means that a 2D convolution can be reduced to the product of two 1D convolutions - one on the rows and one on the columns:

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

$$= \left(\frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{x^2}{2\sigma^2}}\right) \left(\frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{y^2}{2\sigma^2}}\right)$$

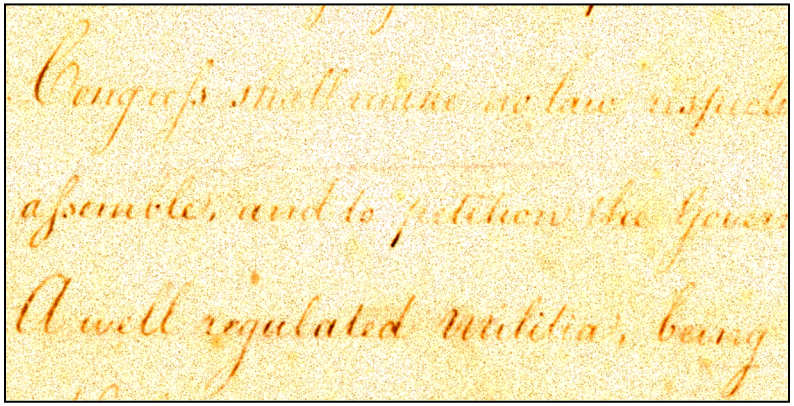
$$= g(x)\, g(y)$$

Sources from fiveko.com and E.G.M. Petrakis

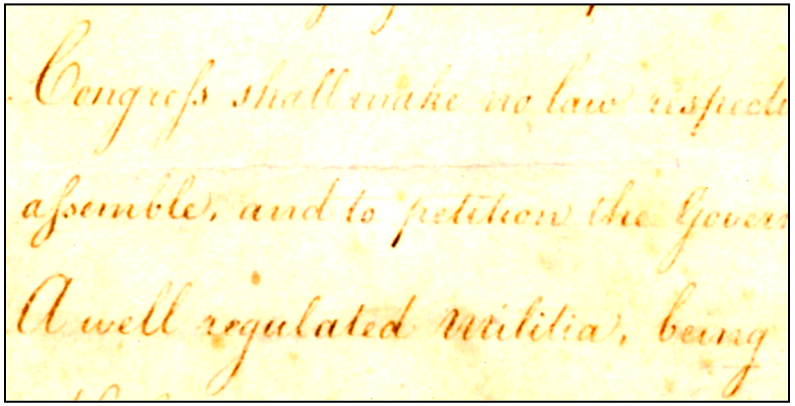
Noise Removal: The Median Filter

- Returns the median value of the pixels in a neighborhood
- Is non-linear
- Is similar to a uniform blurring filter which returns the mean value of the pixels in a neighborhood of a pixel

original



median filtered

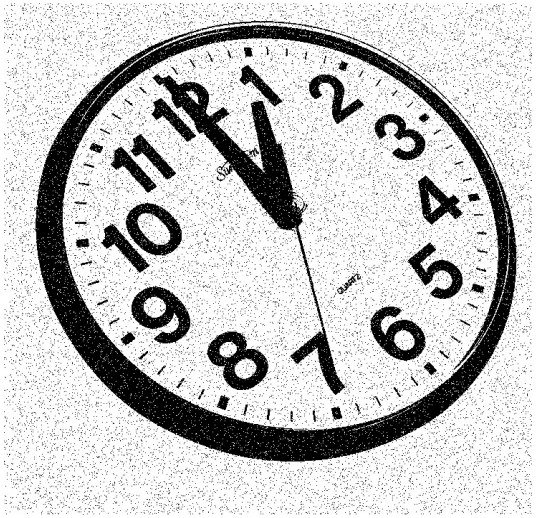


Slide by Richard Alan Peters II

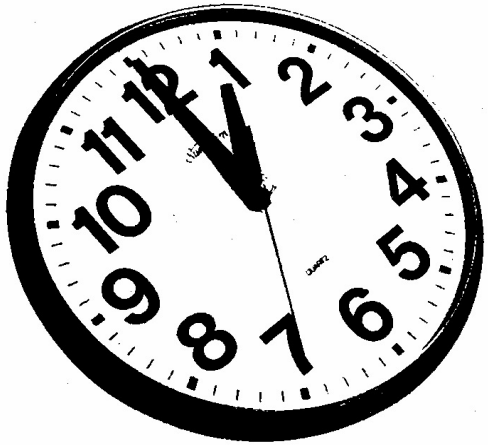
Median Filter: the algorithm

1. Let I be a monochrome image.
2. Let Z define a neighborhood of arbitrary shape.
3. At each pixel location, $\mathbf{p}=(x,y)$, in I …
4. … select the n pixels in the Z-neighborhood of $\mathbf{p}$,
5. … sort the n pixels in the neighborhood of $\mathbf{p}$ by value into a list $L(j)$ for $j = 1,\ldots,n$.
6. The output value at $\mathbf{p}$ is $L(m)$, where $m = \lfloor n/2 \rfloor + 1$.

Original



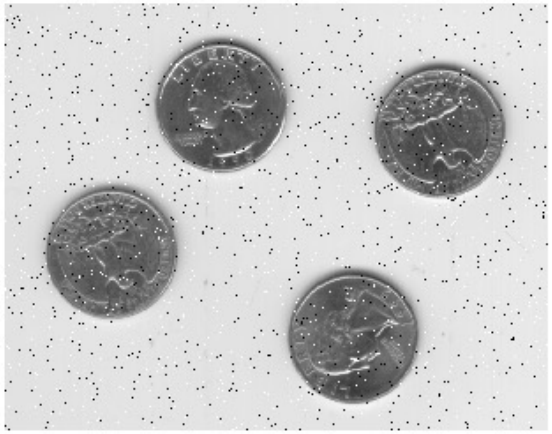
Median filtered



Slide by Richard Alan Peters II

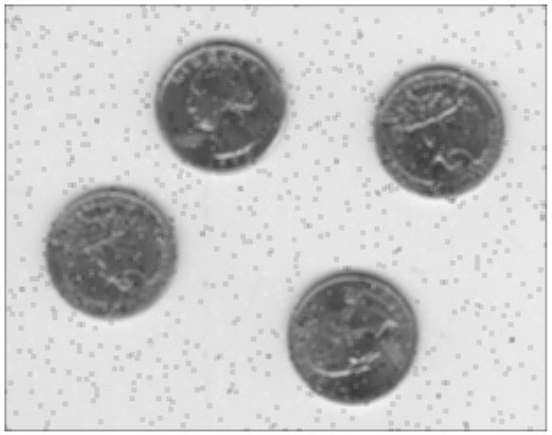
Median filtering of grayscale images

Original



salt & pepper noise added

3x3 averaging filter



3x3 median filter

Sharpening Revisited

Source: S. Lazebnik

What does blurring take away?

Let's add it back:

Summary: Image Filtering

- Images are often corrupted by random variations in intensity, illumination, or have poor contrast and cannot be used directly

- *Filtering* allows us to achieve:
  - *Enhancement:* improves contrast
  - *Smoothing:* removes noise
  - *Template matching*: detects known patterns
  - Feature Extraction: provides clues about objects etc., for further analysis
  - Many other uses…

Adapted from E.G.M. Petrakis