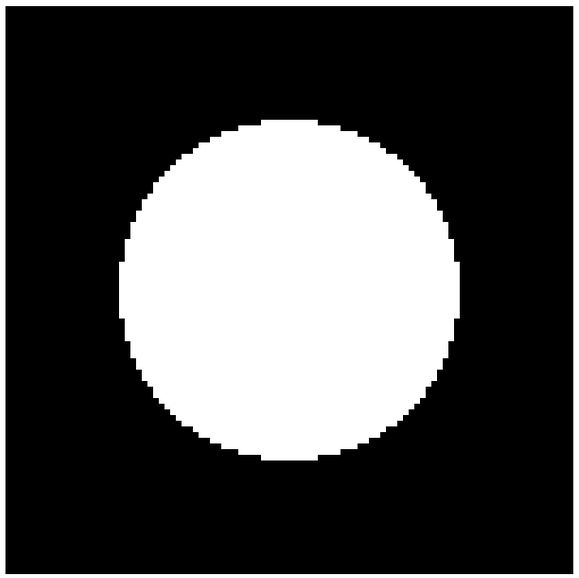
Image Enhancement

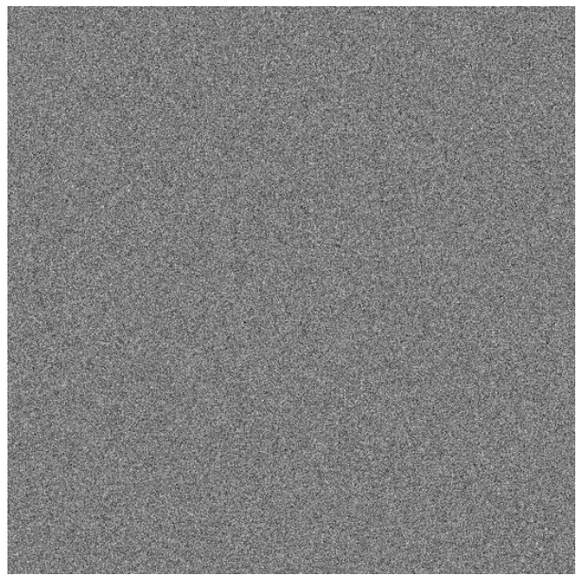
Eric Debreuve
Equipe-Projet Commune **Morpheme**
http://team.inria.fr/morpheme
Inria SAM / Lab. I3S / Lab. iBV

First, Some Examples

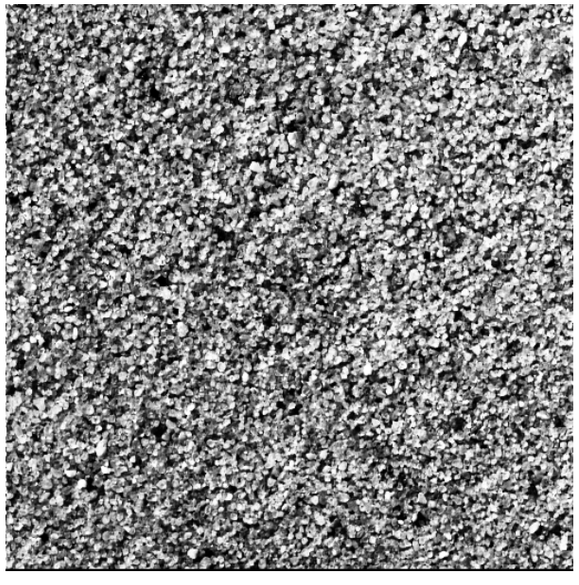


Piecewise-constant image (2 constant, or homogeneous, regions)

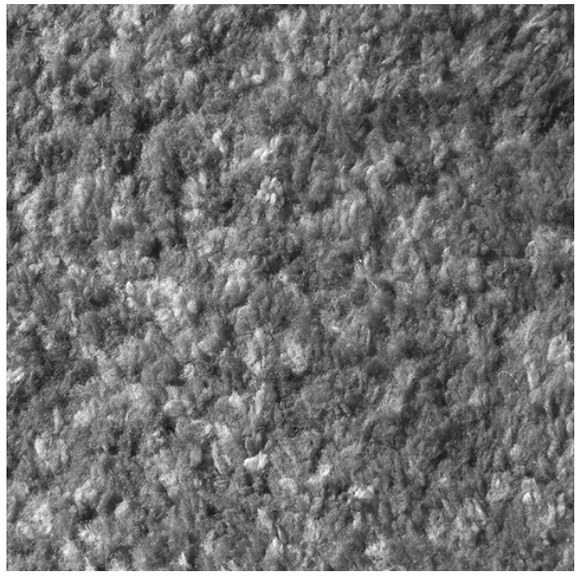
Typical grayscale image



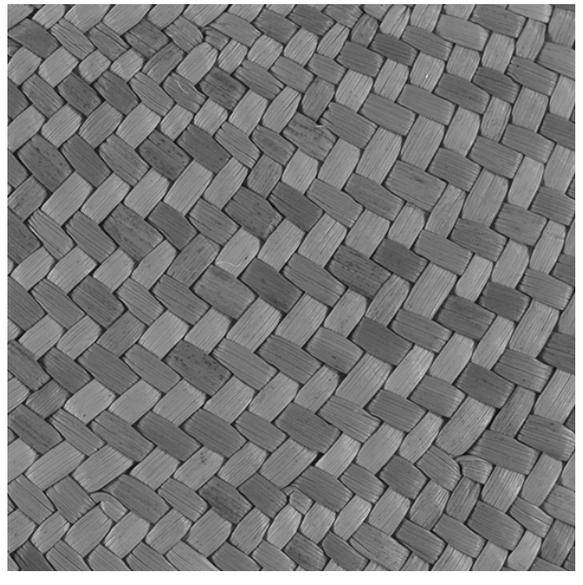
Just noise



Fine-scale texture (almost like noise)



Larger-scale texture



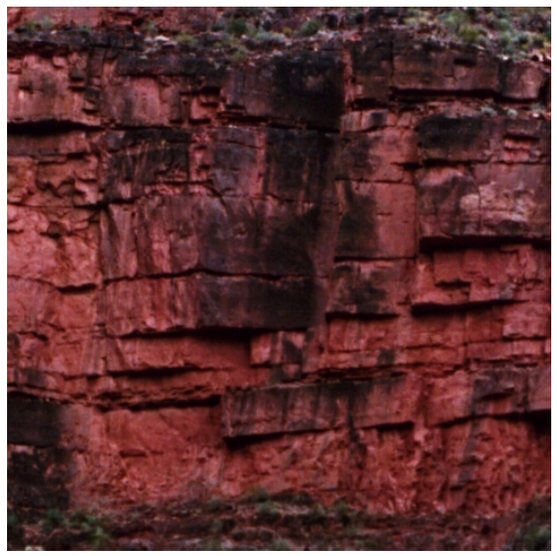
Almost regular repetition of pattern

First, Some Examples

Random with some directionality

Random repetition of pattern

Large-scale, less random texture

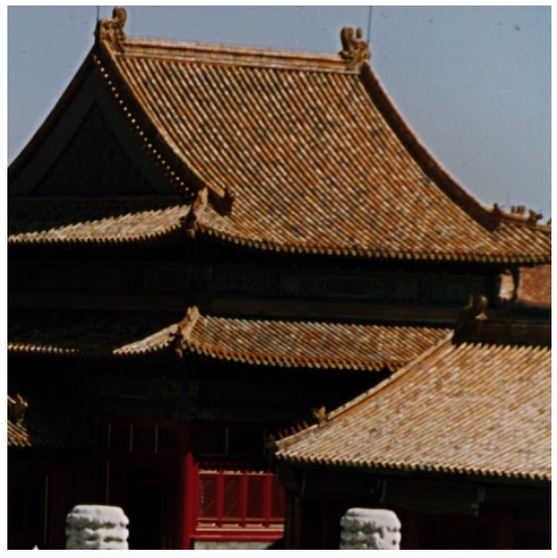


Texture on geometry

Macro-texture with randomness

Macro-texture with less randomness

Some natural textures

Some man-made textures

Some Operations on Images

Original

Noisy

Image combined with some noise (here, additive, white, Gaussian noise (AWGN))

Blurry

Image filtered by low-pass (or smoothing, or blurring) filter (here, Gaussian filter)

Contrast-enhanced

Image intensity modified using histogram equalization

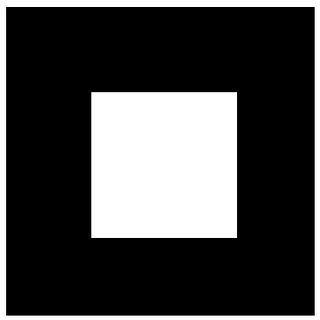
Some Operations on Images

Effect: smoothing, i.e. blurring

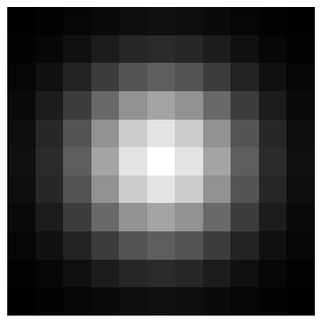
Typical uses

- Image denoising (removing noise from image)
- PSF modeling for deconvolution (to increase image resolution)

Equivalent representations of 2 classical low-pass filters



7 × 7-uniform filter

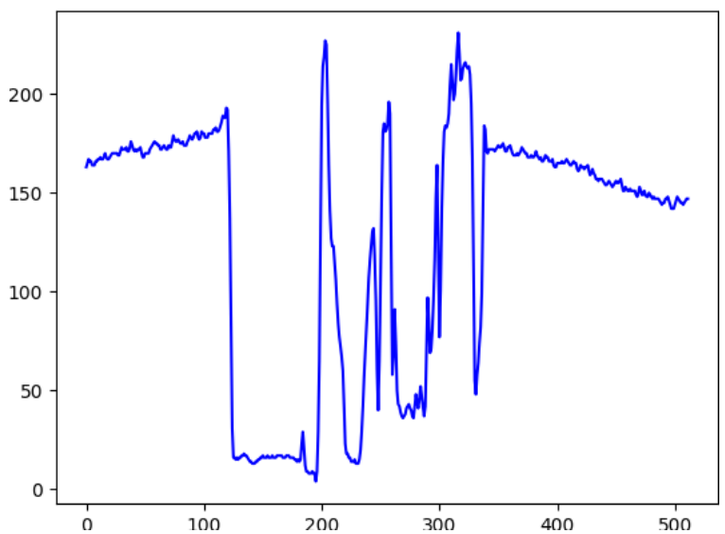


11 × 11-Gaussian filter

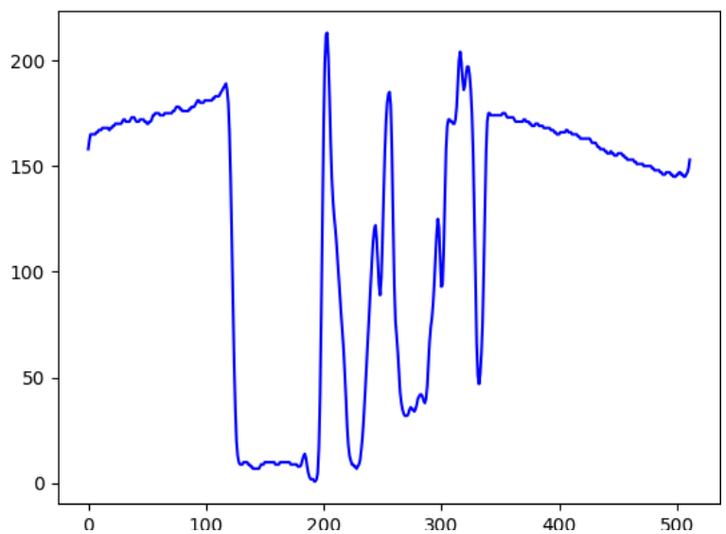
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	1	1	0	0	0	0
0	0	0	1	1	1	1	1	0	0	0
0	0	1	1	2	2	2	1	1	0	0
0	1	1	2	3	3	3	2	1	1	0
0	1	1	2	3	4	3	2	1	1	0
0	1	1	2	3	3	3	2	1	1	0
0	0	1	1	2	2	2	1	1	0	0
0	0	0	1	1	1	1	1	0	0	0
0	0	0	0	1	1	1	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0

Gaussian (times 100 and rounded to integers (hence the zero values))

Effect shown on image profiles



Original profile



Profile after smoothing

Effect: detail enhancement, where details are:

- Noise (unwanted details)
- Textures
- Edges/object contours

Typical uses

- Edge enhancement/detection
- Other sharp structure enhancement/detection
- Image sharpening

3 illustrated cases

- 1^{st} order derivative
- 2^{nd} order derivatives: Laplacian
- 2^{nd} order derivatives: Frangi filtering

High-pass Filtering

1st order derivative

Image gradient (norm of vector of horizontal and vertical (1st order) derivatives) computed using the Sobel filters

+1	0	-1
+2	0	-2
+1	0	-1

Horiz. derivative (Sobel)

-1	-2	-1
0	0	0
+1	+2	+1

Vertical derivative (Sobel)

Horizontal and vertical 2nd order derivatives: Laplacian

Image Laplacian (sum of horizontal and vertical 2nd order derivatives)

0	1	0
1	-4	1
0	1	0

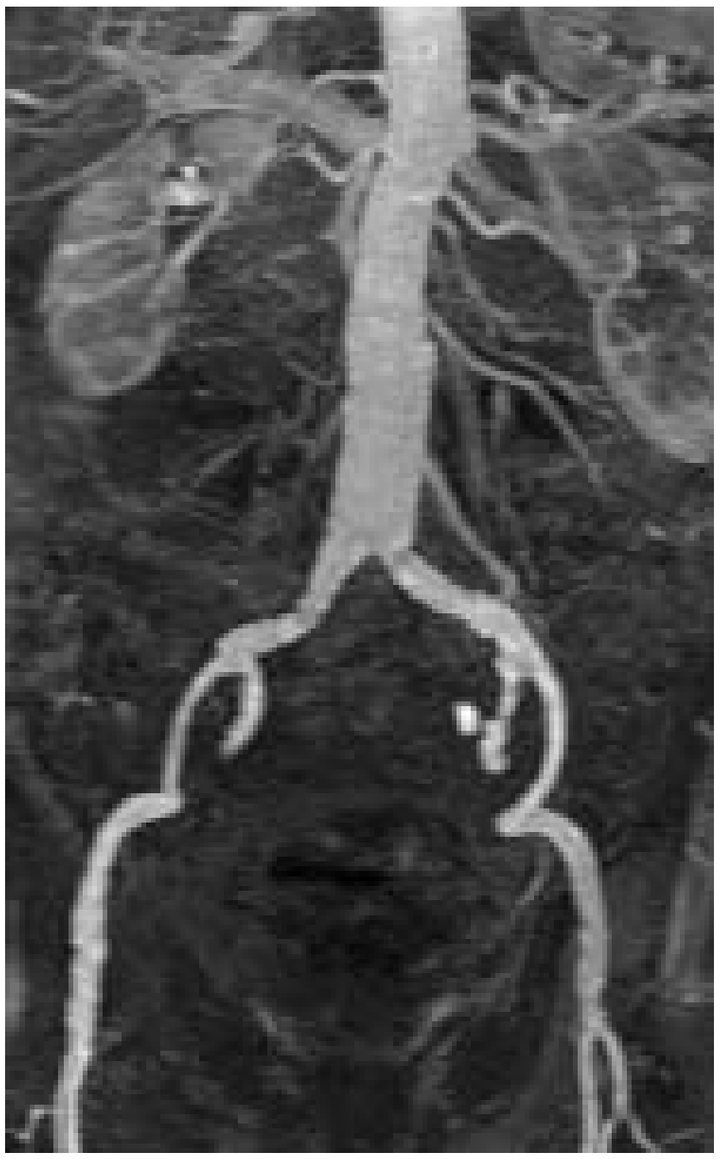
Laplacian
(4-connectivity)

1	1	1
1	-8	1
1	1	1

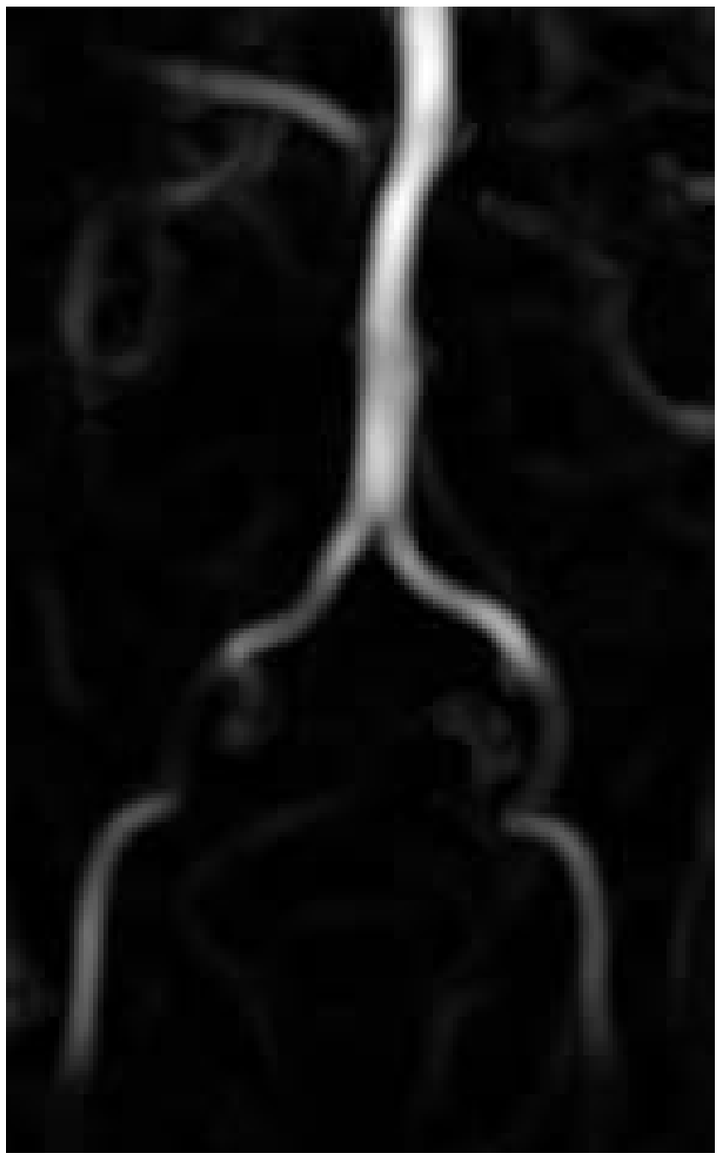
Laplacian
(8-connectivity)

High-pass Filtering

All 2nd order derivatives: e.g. Frangi filtering [Frangi *et al*, 1998]



Original



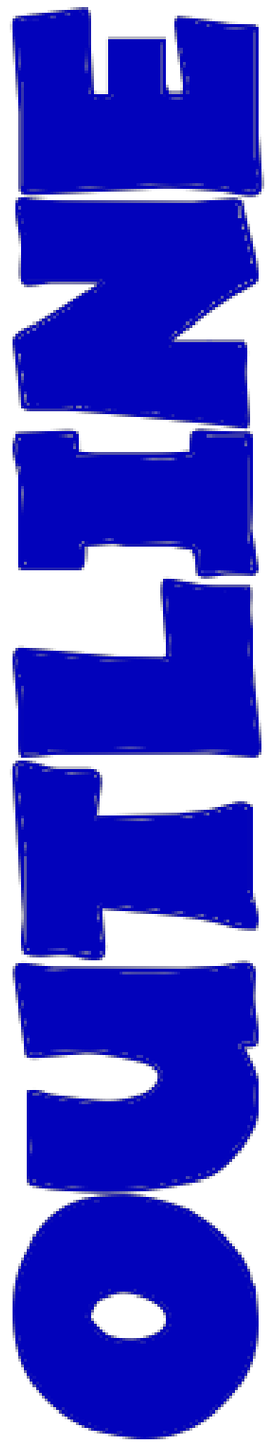
With vessel-like structures enhanced and other structures attenuated

Filtering

- Intuition for low-pass filtering
- Convolution
- Fourier transform

Filtering (and other approaches) for...

Enhancing image quality



(Low-pass) Filtering

In General

Set of related measurements

- Measurements: temperature

- Relation: chronology

Information summary

- Daily measure ⟶ weekly overview
- Operation: average
  - Loose details ⟶ gain “Big picture”

Ways to average

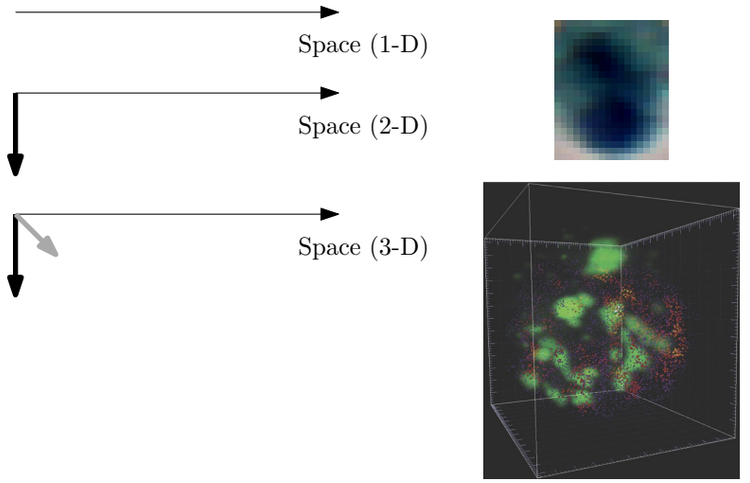
- Global: all measures ⟶ one measure
  - Cold, mild, tropical climate
- Local
  - Disjoint windows: measures of the week ⟶ average for the week: 
  - Sliding window (a.k.a. moving average): measures from yesterday, today and tomorrow ⟶ average for today: 

In Image Processing

Set of related measurements

- Measurements: light intensity or color

- Relation: spatial arrangement



Information summary

- Pixel measure ⟶ neighborhood overview

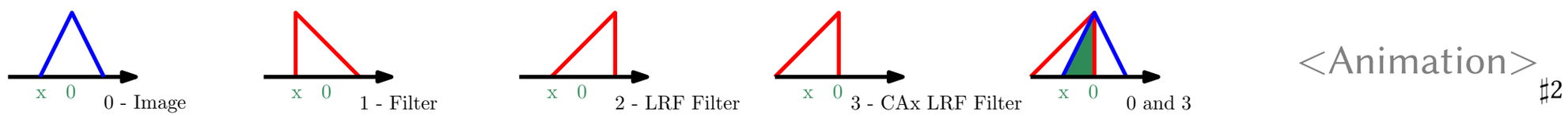
Ways to average

- Global: light or dark image
- Local
  - Disjoint windows: why not
  - Sliding window = filtering

Filtering Formalization

Convolution (1-D)

- Expression: i=image, f=filter (centered around zero), $*$=convolution symbol
  - Continuous ($x, u \in \mathbb{R}$): $(i * f)(x) = \int_{-\infty}^{+\infty} i(u) f(x-u)\,du = \int_{-\infty}^{+\infty} i(x-u) f(u)\,du$
  - Discrete ($x, u \in \mathbb{Z}$): $(i * f)(x) = \sum_{u=-\infty}^{+\infty} i(u) f(x-u) = \sum_{u=-\infty}^{+\infty} i(x-u) f(u)$
- Computation for a given x: $\underbrace{f(u)}_{\text{1 - Filter}} \rightarrow \underbrace{f(-u)}_{\text{2 - LR-flipped filter}}{}^{\sharp 1} \rightarrow \underbrace{f(x-u)}_{\text{3 - LR-flipped filter centered around x}} \rightarrow \underbrace{i(u) f(x-u)\,du}_{\text{Multiplication for all u's}} \rightarrow \underbrace{\int \cdots du}_{\text{Sum for all u's}}$



<Animation>[♯2]

- From 1-D to 2-D: $x \rightarrow (x, y)$, $u \rightarrow (u, v)$, $\int_u \rightarrow \int_u \int_v$, $\sum_u \rightarrow \sum_u \sum_v$

Computational complexity (2-D)

- Direct computation (above)
  - $\mathcal{O}(WHwh)$ if W/H=width/height of image and w/h=width/height of filter
  - $\mathcal{O}(WHw + WHh)$ if filter is separable ($f = f_1 f_2$ where f_i is 1-D)
- Using the Fourier transform (see ie_15): $\mathcal{O}(WH \log(WH))$

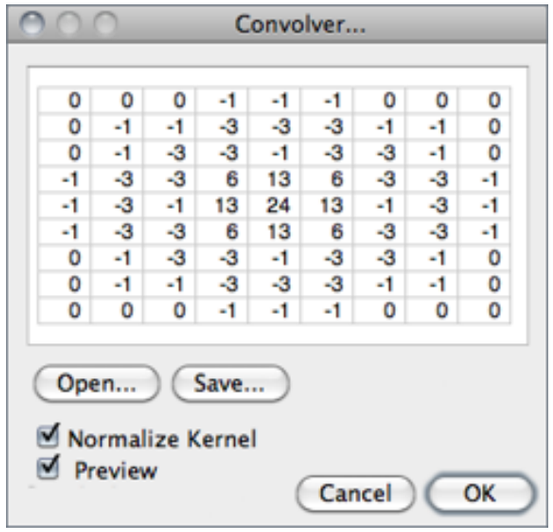
[♯1] Without effect for symmetric filters

[♯2] From Wikipedia

Filtering

Filter size

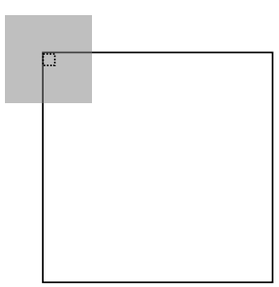
- Usually (but not necessarily) square: $w \times h = n \times n$
- Width and height: Odd numbers: 3×3 or 5×5 or 7×7...
  - Hence there is a central pixel



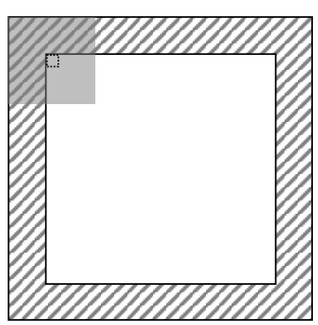
9×9-Filter from ImageJ documentation

Pixels needed to compute a convolution

- Missing image pixels when filter centered on first pixel of image



- Missing image pixels (hashed area) required by the convolution



- *Solution:* invent missing pixels

Filtering

No continuity between image and values invented outside

- Filling with zeros
- Periodicity (“silently” applied when using the Fourier transform)

With continuity

- Closest neighbors
- Symmetrization

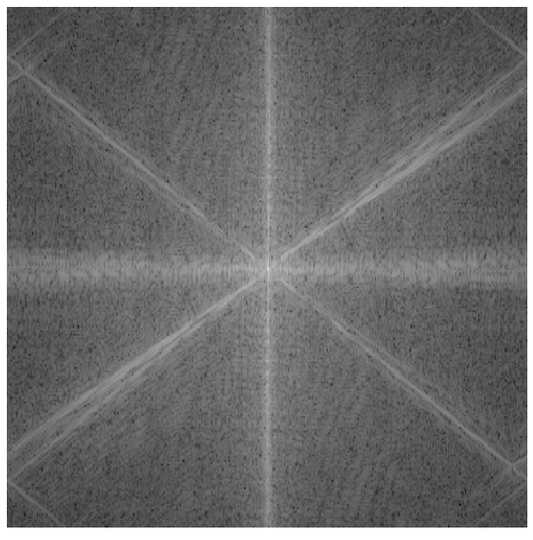
Fourier Transform

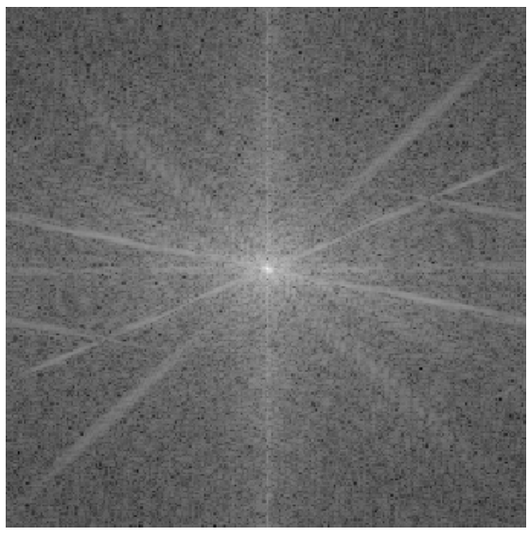
Landscape

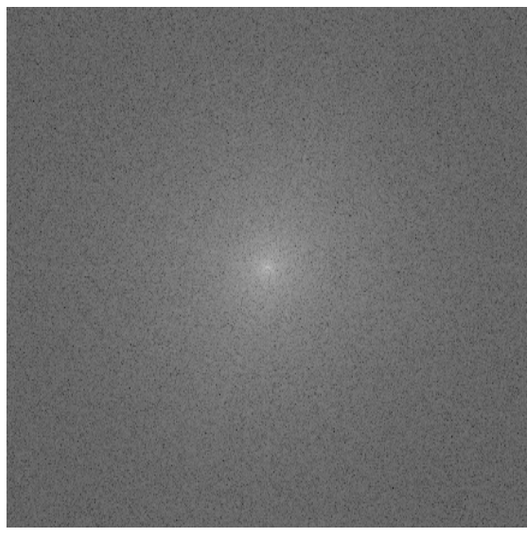
Crossroads

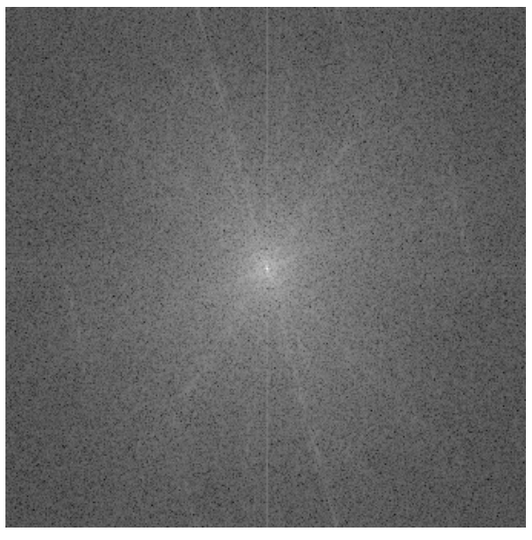
Bridge

Cameraman









Which images these (logs of) magnitudes correspond to?

Fourier Transform

Transforms a signal (here an image)...

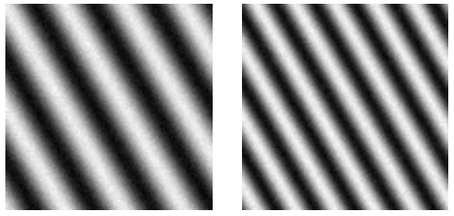
- From the spatial domain (image) into a frequency domain (Fourier representation)
- (Grayscale) Image: each pixel ⟵ one value
- Fourier representation
  - Image of the same size
  - Each pixel ⟵ a complex number (2 values)
  - These 2 values can be decomposed into magnitude and phase (⇔ angle)

Interpretation

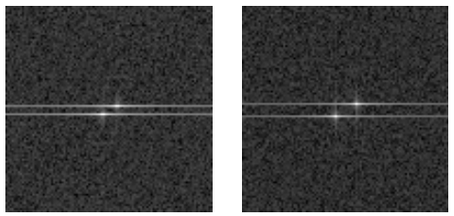
- Frequency = spatial frequency: how fast the signal changes in a given direction
- Magnitude
  - At a given rate: how often the change rate is encountered in the image
  - In a given direction: direction of the change = direction orthogonal to contours
- Phase: difficult to interpret, but crucial

Illustrations

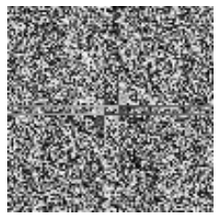
- Synthetic



Sine waves



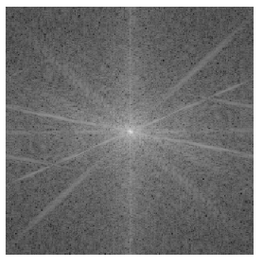
(Log of) Magnitude



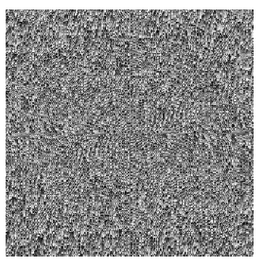
Phase

- Real example

"Cameraman"



(Log of) Magnitude



Phase

Fourier Transform...

Notations

- $\mathcal{F}(g)$ = Fourier transform of g
- $\mathcal{F}^{-1}$ = inverse Fourier transform: $g = \mathcal{F}^{-1}(\mathcal{F}(g))$

Fourier transform property

- $\mathcal{F}(i * f) = \mathcal{F}(i)\mathcal{F}(f)$

Fast Fourier Transform (FFT)

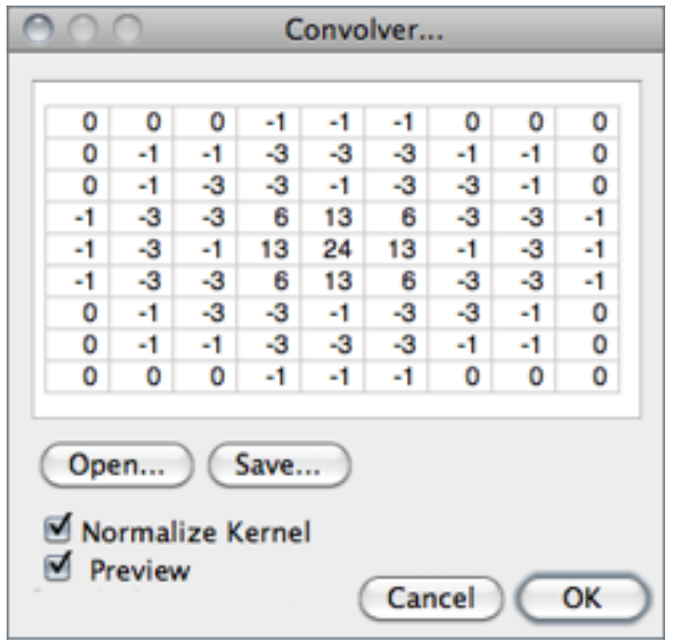
- Fast algorithm to compute the Fourier transform

Convolution computation

- $i * f = \mathcal{F}^{-1}(\mathcal{F}(i)\mathcal{F}(f))$
- Advantage: speed
  - $\mathcal{O}(WHwh) \rightarrow \mathcal{O}(WH\log(WH))$ (see ie_12)
- Consequence: (silent) boundary condition = periodicity (see ie_13)

In practice

- Prefer FFT-based implementations for filtering
- ImageJ documentation at `http://imagej.net/docs/guide/146-29.html`
  Excerpt: *"Like all ImageJ convolution operations, it assumes that out-of-image pixels have a value equal to the nearest edge pixel."* $\Rightarrow$ a priori not computed using the Fourier transform. Expect long computation times for large images using a large filter



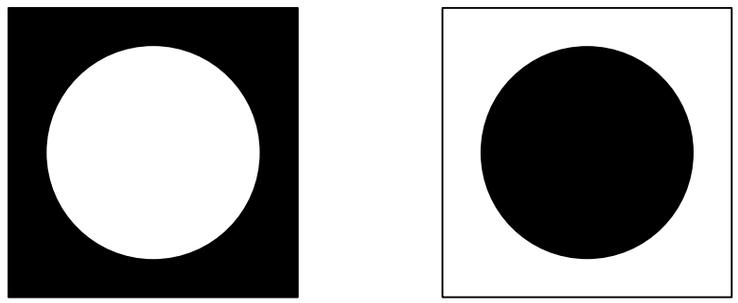
From ImageJ documentation

Low-pass and High-pass Filtering

Effect of filters

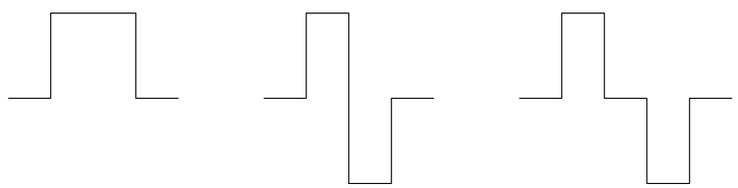
- Low-pass: typically smoothing filters
- High-pass: typically derivative filters ⇒ details/edge enhancement/detection

In Fourier domain: multiplication with...



Low-pass (left) and high-pass (right)

In Spatial domain: convolution with...



Low-pass (left) and high-pass (middle & right)

- Smoothing ⇒ filter coefficients sum to 1
- Derivation ⇒ filter coefficients sum to 0

Derivative

- Line and curve slope

- Notation
  - 1-D: I' or $\frac{dI}{dx}$
  - n-D: $\frac{\partial I}{\partial x}^{\sharp 1}, \frac{\partial I}{\partial y}, \frac{\partial I}{\partial z}, \cdots$
  - Gradient ∇I: vector of all partial derivatives
- Discrete approximation: $\frac{\partial I}{\partial x} \simeq \frac{I(x+dx) - I(x)}{dx}$
  - Usually one takes: $dx = 1$

Some classical filters (see also Gaussian filter on ie_5)

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

Uniform

+1	0	-1
+1	0	-1
+1	0	-1

Horiz. derivative (Prewitt)

+1	0	-1
+2	0	-2
+1	0	-1

Horiz. derivative (Sobel)

Laplacian (4-connectivity)

0	1	0
1	-4	1
0	1	0

1	1	1
1	-8	1
1	1	1

Laplacian (8-connectivity)

[#1] Partial derivative w.r.t. x

Filtering Purposes

Getting the "Big picture" (see ie_11)

- As a pre-processing before object detection for example

Getting rid of noise (a.k.a. denoising)

- Part of "Enhancing image quality" (see ie_23)

Getting back high resolution (a.k.a. deconvolution)

- Part of "Enhancing image quality" (see ie_30)

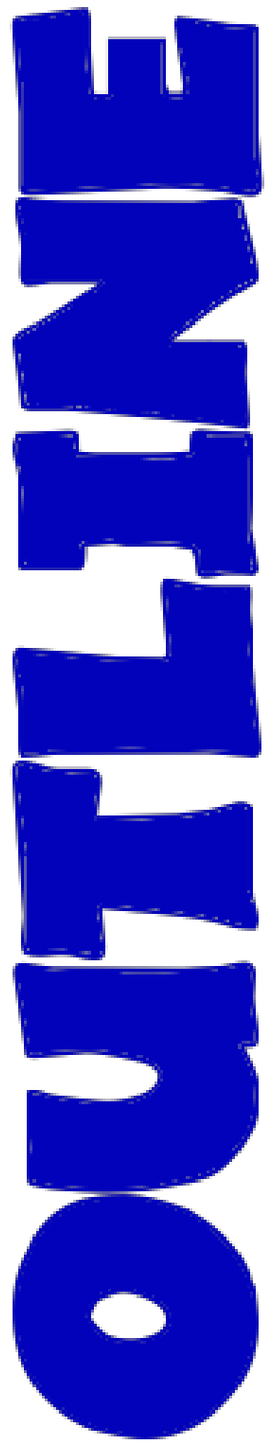
Enhancing image features

Filtering

Filtering (and other approaches) for...

- Denoising
  - Definition of noise
  - Low-pass filtering and Fourier
  - Additive noise: linear filtering
  - Multiplicative noise: "log-linear" filtering
  - Salt&Pepper noise: non-linear, median filtering
  - Non-local means
- Deconvolution
- Enhancing image features

Enhancing image quality



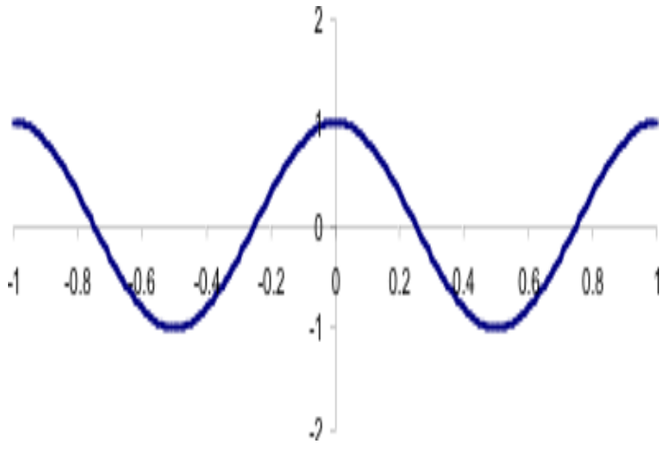
Definition of Noise

What is noise?

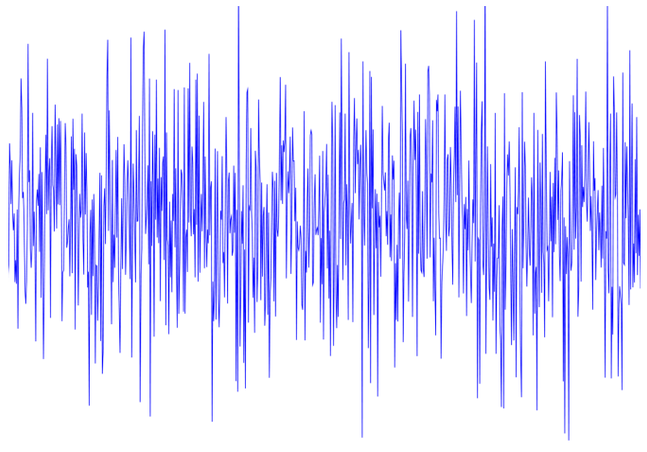
- There is a main audio *signal* (music playing)
- There is an audio disturbance (car passing by): *noise*
- Subjective notion
  - In a night club: if *signal* is conversation, *noise* is music

What is noise in an image?

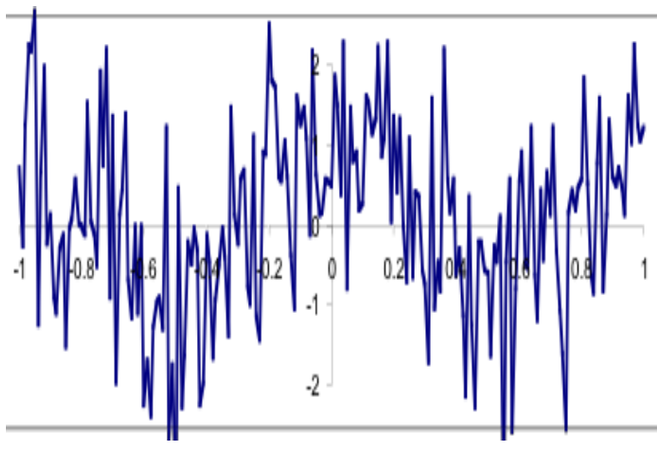
- Low-amplitude, high-frequency, random signal on top of image
  - Low-amplitude: if same amplitude as image, possible to distinguish between noise and image?
  - High-frequency: fast variation from pixel to pixel (see ie_16)
  - Random: properties can be approximated (mean, variance…), but actual values are unknown



Signal



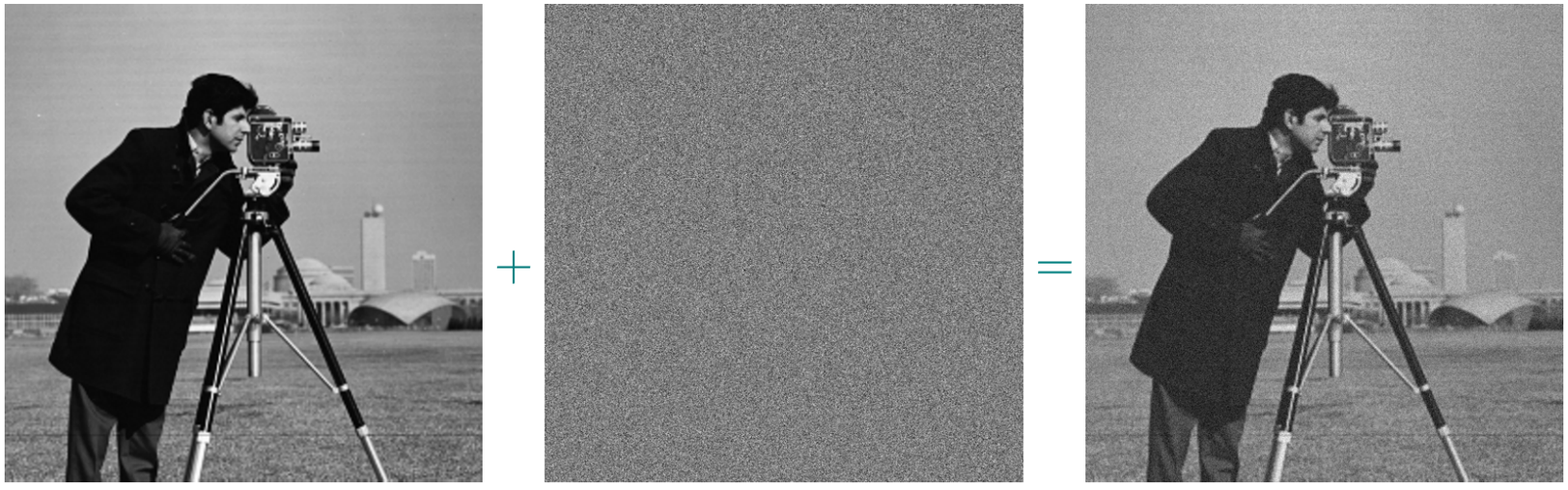
Noise



Noisy signal

Types of Noise

Additive: adds up to the image



Multiplicative: multiplies image intensities with random values

Poisson: replace image intensities with random values (in a specific way)

Salt&Pepper (a.k.a. impulse): replace image intensities with 0 or 255[♯1] randomly

[♯1] For 8-bit images

Why low-pass filtering?

Signal frequencies

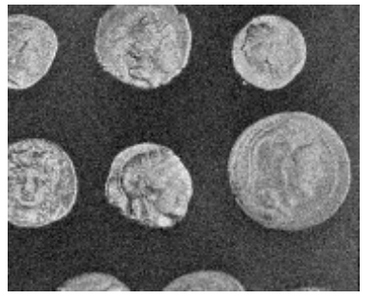
- Low $\Leftrightarrow$ near constant areas
- High $\Leftrightarrow$ textures, edges, object contours
- Very high $\Leftrightarrow$ noise

Noisy image: additive noise model

- Image = signal with some low-, medium-, and high-frequency contents
- Noise = signal composed only of very high frequencies
- Noisy image = Image + Noise

Low-pass filtering (linear operation)

- Remove the highest components of Noisy image to get Image back
  - Low-pass = let low frequencies pass through filter
    while blocking (filtering out) high frequencies



- Problem: find frontier between image high frequencies and noise very high ones

Trade-off noise/details

- Image frequency range: $[\text{low}_\text{I}, \text{high}_\text{I}]$
- Noise frequency range: $[\text{low}_\text{N}, \text{high}_\text{N}]$
  - Usually: $[\text{low}_\text{I}, \text{high}_\text{I}] \cap [\text{low}_\text{N}, \text{high}_\text{N}] \neq \phi$

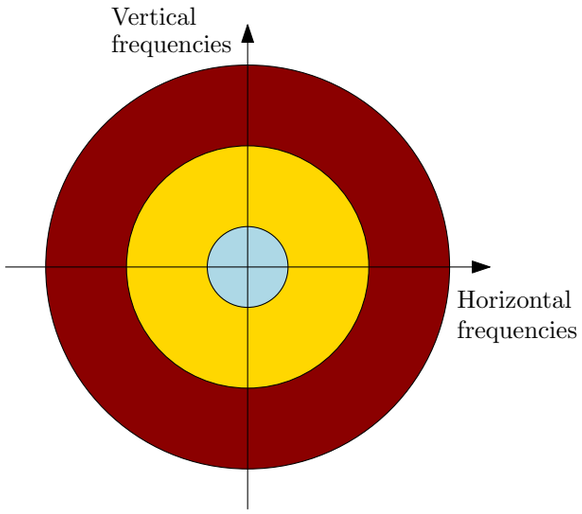
$\Rightarrow$ While removing noise, some image details will be removed too

$\Rightarrow$ After denoising: slightly smooth/blurry version of original image

Low-pass Filtering and Fourier

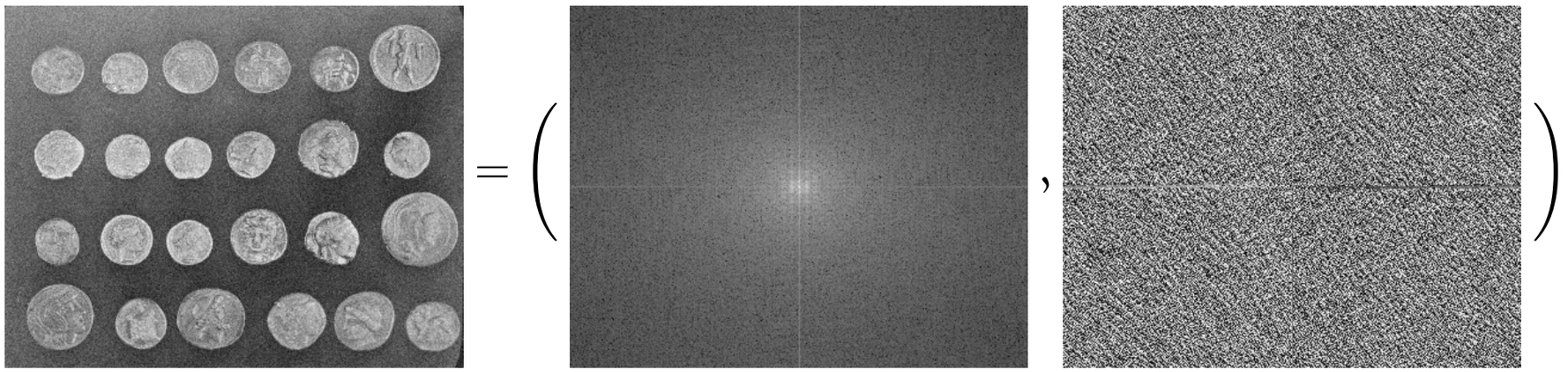
Link with Fourier domain

- Energy of directional frequencies in signal
  - Close to origin: low frequencies
  - Away from origin: high frequencies
  - Far away from origin: very high frequencies

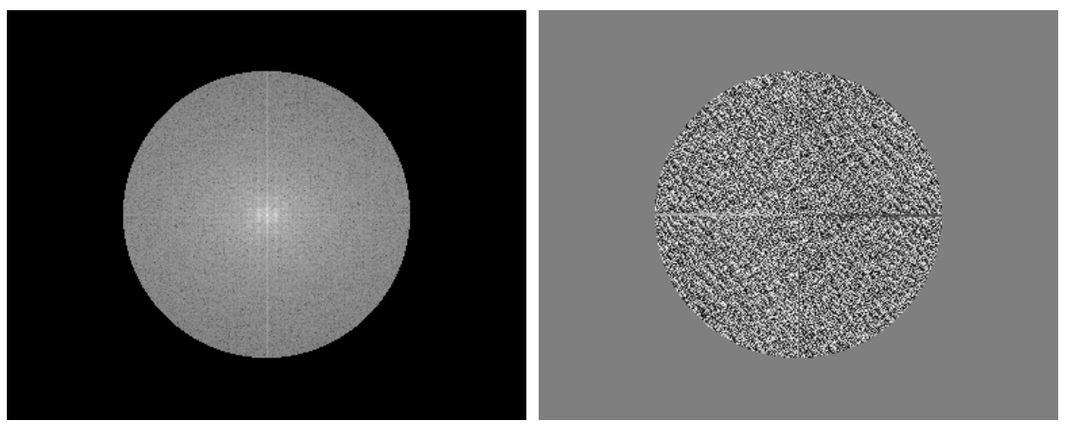


Crude procedure

- Compute Fourier transform of noisy image



- Set highest frequencies to zero



- Compute inverse Fourier transform of result

Multiplicative and Salt&Pepper Noises

Denoising, but not true filtering

Multiplicative noise

- Model: $\text{Acq} = \text{Perfect} \times \text{Noise}$
- Then: $\log(\text{Acq}) = \log(\text{Perfect}) + \log(\text{Noise})$
  - $\Rightarrow$ Apply method for additive noise
    - However $\log(\text{Noise})$ might not satisfy properties required by method

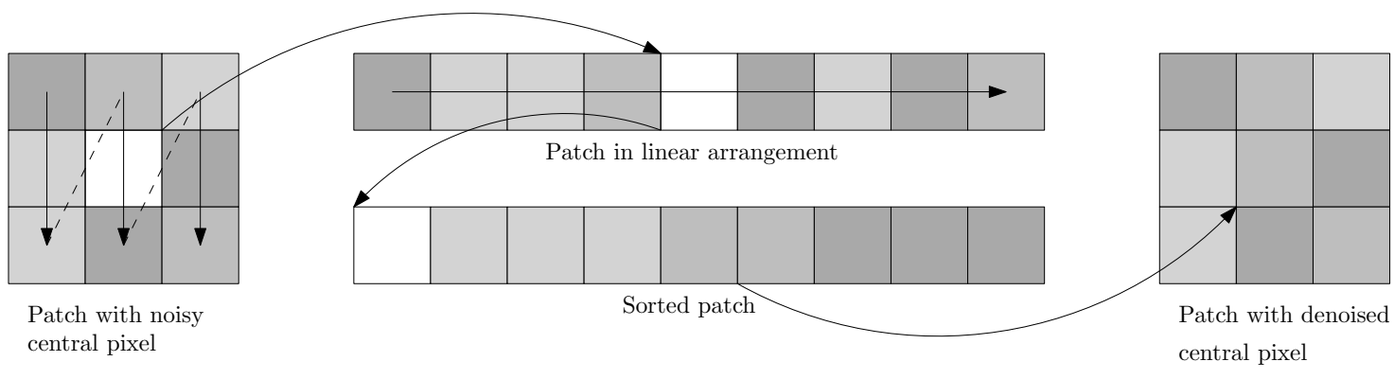
Salt&Pepper noise

- For a given pixel
  - Take a small patch[♯1] around it (say an $n \times n$-square, n odd)
  - This patch is probably *rather homogeneous*
  - Sort its n^2 intensities:

    $$\underbrace{i_1 \leq i_2 \leq \cdots}_{(n^2-1)/2} \leq i_m \leq \underbrace{\cdots \leq i_{n^2}}_{(n^2-1)/2}$$

    where $m = \dfrac{(n-1)(n+1)}{2} + 1$
  - Replace pixel intensity with i_m

- Why does it works?
  - If pixel (with intensity i) is not noise, then $i_m \simeq i$
    - Median filter has almost no effect
  - If pixel is noise, then
    - Either i smaller than intensities in patch and sorted first
    - Or i higher than intensities in patch and sorted last
    - In both cases, replacing i with i_m removes noise



- Called median filter
  - But not a filter in convolutional sense
  - It is a rank filter

[♯1] Patch: small piece of image

Non-Local Means (NL-Means) and UINTA

Denoising, but not filtering

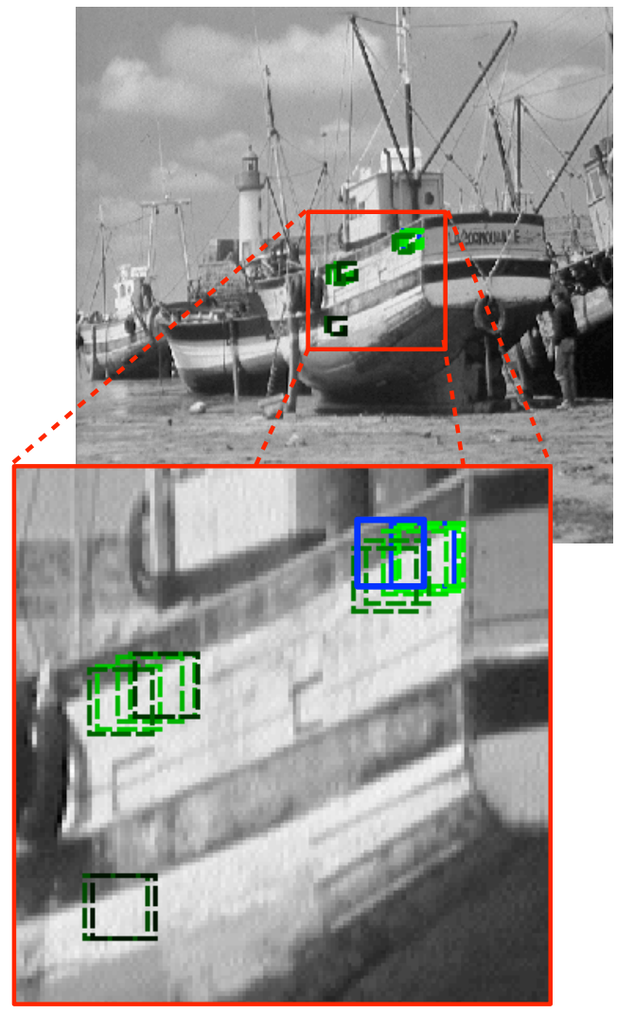
General idea

- Patch: small piece of image (typically, an $n \times n$-square)
- Assumption: independent, identically distributed noise
- For a given patch
  - There certainly exists similar patches in image

  - Each patch P_i = *same* noiseless image patch P + noise patch N_i
    - If mean of noise is 0, then

$$\frac{1}{n}\sum_{i=1}^{n} P_i = P + \underbrace{\frac{1}{n}\sum_{i=1}^{n} N_i}_{\simeq \text{Patch of 0's}}$$

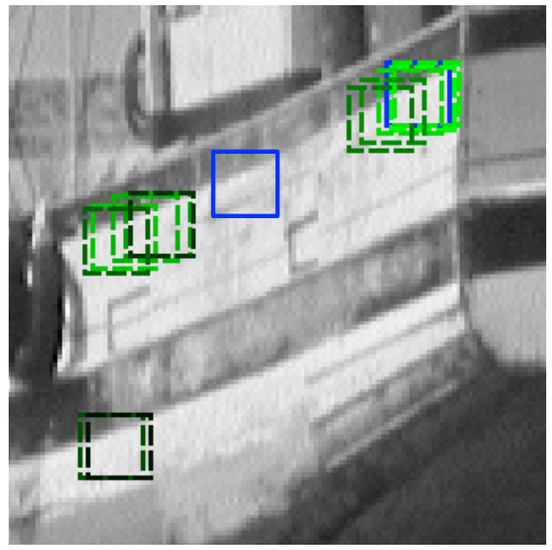
- Do that for all the patches of image



Advantage

- No spatial smoothing (smoothing is done across patches)

$\Rightarrow$ *No loss of details*

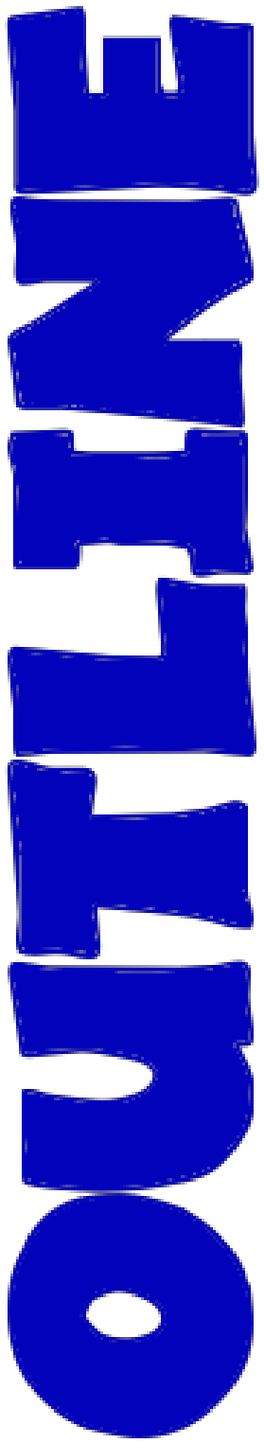


Filtering

Filtering (and other approaches) for...

- Denoising
- Deconvolution
  - Loss of resolution and PSF
  - Always possible?
  - General principle
- Enhancing image features

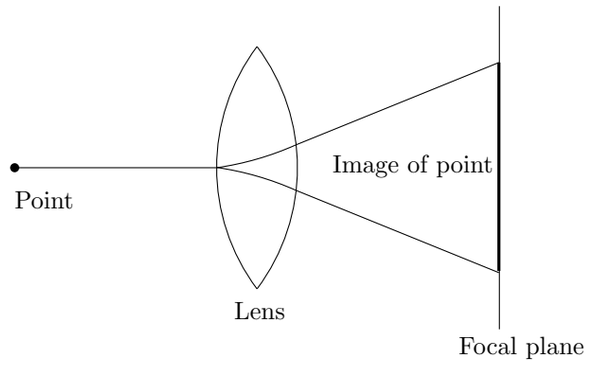
Enhancing image quality



Loss of Resolution

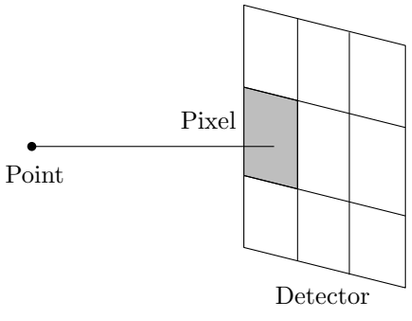
Problem

- Optical system not perfect
- Image sensor composed of "not infinitely thin" receptors
- Overall: point in reality $\longrightarrow$ blob on acquired image
  - Called loss of resolution
  - This blob = Point Spread Function (PSF)



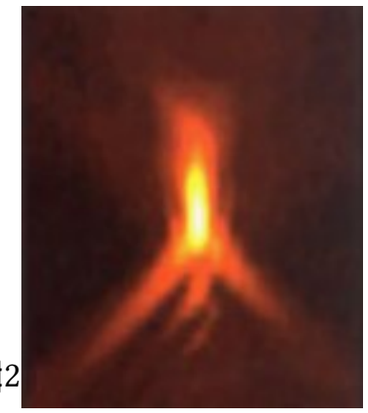
Resolution definition

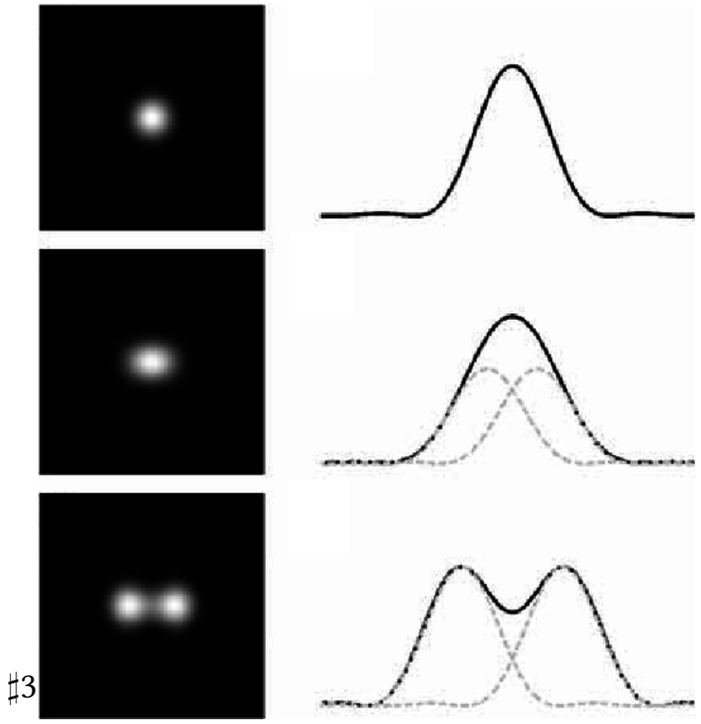
- Smallest distance between 2 points having distinguishable blobs



PSF can...

- depend on distance to lens
- depend on position in image (local), or not (global)
- be radially symmetric (isotropic), or not (anisotropic)
- Letting alone depth dependence[♯1], typical cases are:
  - Global, isotropic
  - Global, anisotropic
  - Local, anisotropic


♯2


♯3

[♯1] Assuming there is a fixed depth of interest
[♯3] From Molecular Expressions
[♯3] From Bertocchi *et al*, 2013

Deconvolution

An Inverse Problem

Purpose

- From acquired image with degraded resolution
  - Ideally: find a way to go back to the high-resolution image that was "at the entrance of the lens"
  - In practice: find a way to compute an image with higher resolution
- Forward way: from image "at the entrance of the lens" to acquired image
- Backward way: the opposite

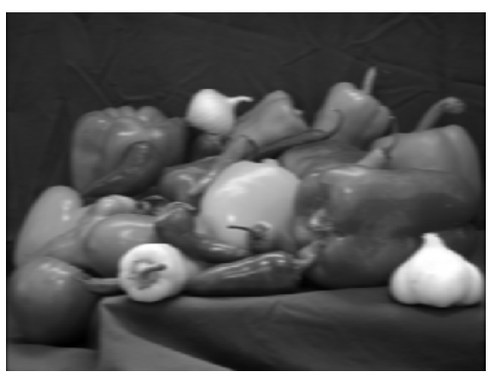
Phenomenon modeling

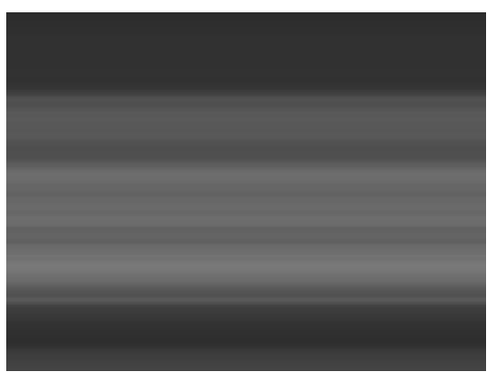
- Phenomenon = "point $\longrightarrow$ blob" (just like smoothing)

  $\Rightarrow$ Model = convolution with a low pass filter, and
  - Low pass filter = PSF (property of convolution)

Deconvolution always possible?

- Intuition
  - Forward way: $\underbrace{\{1,2,1,4,2\}}_{\text{Data}} \xrightarrow[\substack{\text{Convolution:}\\ 1+2+1+4+2}]{} \underbrace{10}_{\text{Acquisition}}$
  - Backward way (the inverse problem): acquisition is 10. What were the 5 numbers of the operation?
- Example with smoother and smoother image



Deconvolution

Phenomenon = convolution with PSF

- Assumption: PSF is *known*
  (theoretical approximation or experimental measure)

Modeling

- $\text{Acq} = \text{Perfect} * \text{PSF}$
- In Fourier domain: $\mathcal{F}(\text{Acq}) = \mathcal{F}(\text{Perfect})\mathcal{F}(\text{PSF})$

Then:

$$\mathcal{F}(\text{Perfect}) = \frac{\mathcal{F}(\text{Acq})}{\mathcal{F}(\text{PSF})} \qquad \text{if } \mathcal{F}(\text{PSF}) \neq 0 \text{ everywhere}$$

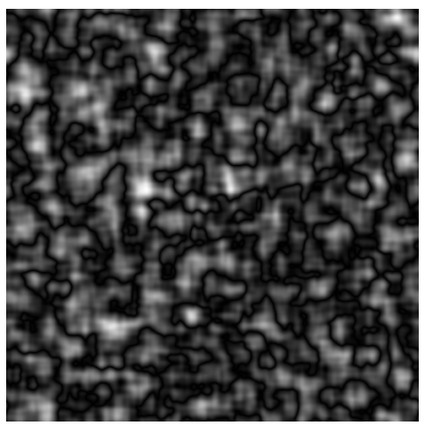
$$\Leftrightarrow \text{Perfect} = \mathcal{F}^{-1}\left(\frac{\mathcal{F}(\text{Acq})}{\mathcal{F}(\text{PSF})}\right) \qquad \text{(naive solution)}$$

In practice though

- Acq also degraded by noise: $\text{Acq} = \text{Perfect} * \text{PSF} + \text{Noise}$
- Therefore: $\mathcal{F}(\text{Perfect}) = \dfrac{\mathcal{F}(\text{Acq}) - \mathcal{F}(\text{Noise})}{\mathcal{F}(\text{PSF})}$
- Problem
  - $\mathcal{F}(\text{Noise})$ high for high frequencies
  - $\mathcal{F}(\text{PSF})$ low for high frequencies
  - High frequencies: $\dfrac{\text{high}}{\text{low}} \rightarrow$ very high
    - Noise amplification
    - Naive solution dominated by noise

Cameraman

Acq = blur + (unnoticeable) noise



Naive deconvolution

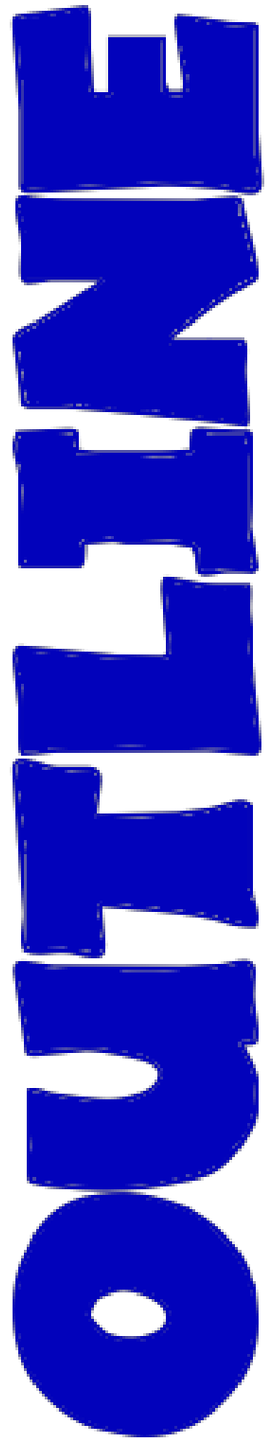
Naive deconvolution if setting noise to zero

Filtering

Filtering (and other approaches) for…

- Denoising
- Deconvolution
- Enhancing image features
  - High-pass filtering
  - Edge enhancement (or sharpening)
  - Linear structure enhancement: Frangi

Enhancing image quality



Enhancing Image Features

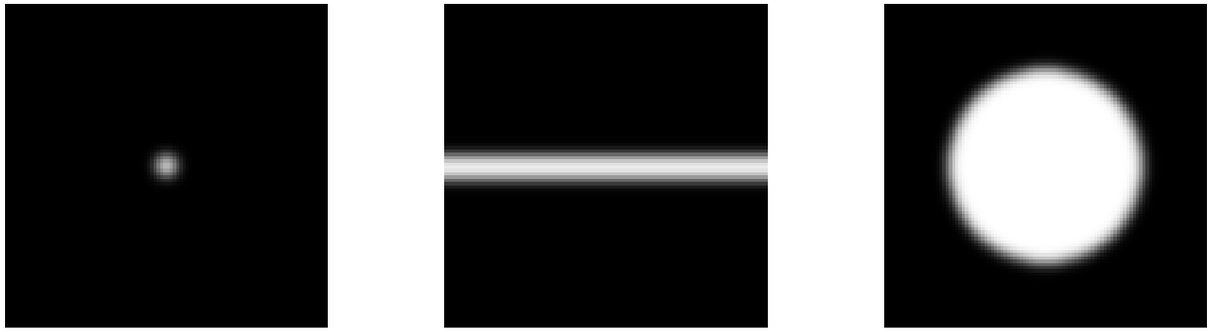
Image feature?

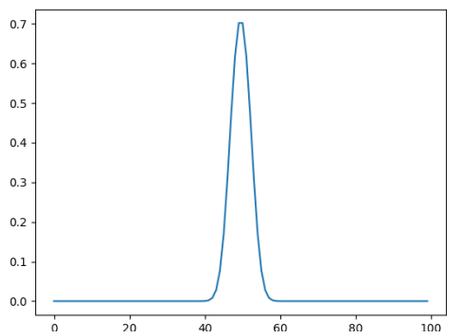
- Noticeable element in image
  - Something that pops up from its neighborhood
    - Small area contrasting with its neighborhood: small peak or well
    - Long area " " " : crest line or narrow valley
    - Frontier between 2 rather homogeneous areas: edge, object contour
  - Fast variation of intensity across feature

    ⇒ High frequency

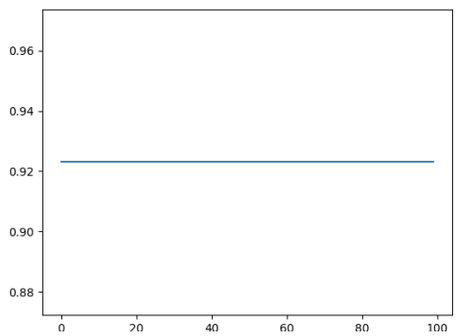
    ⇒ Detection using high-pass filtering

Illustration with some profiles

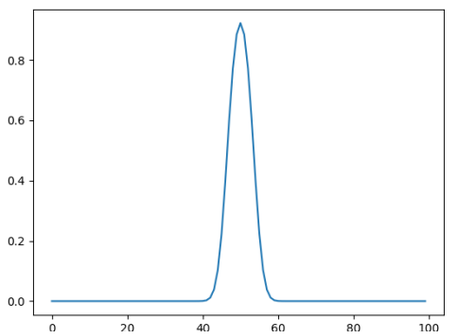




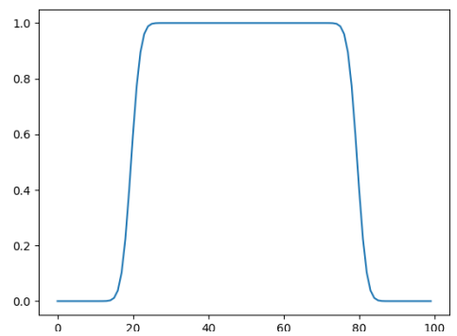
Peak profile



Line profile along



Line profile across



Disk profile

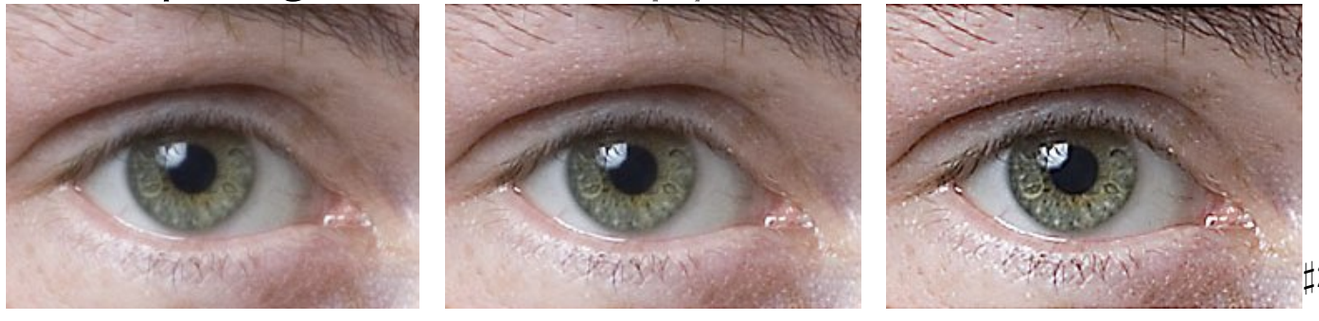
Edge Enhancement (a.k.a. Sharpening)

Principle

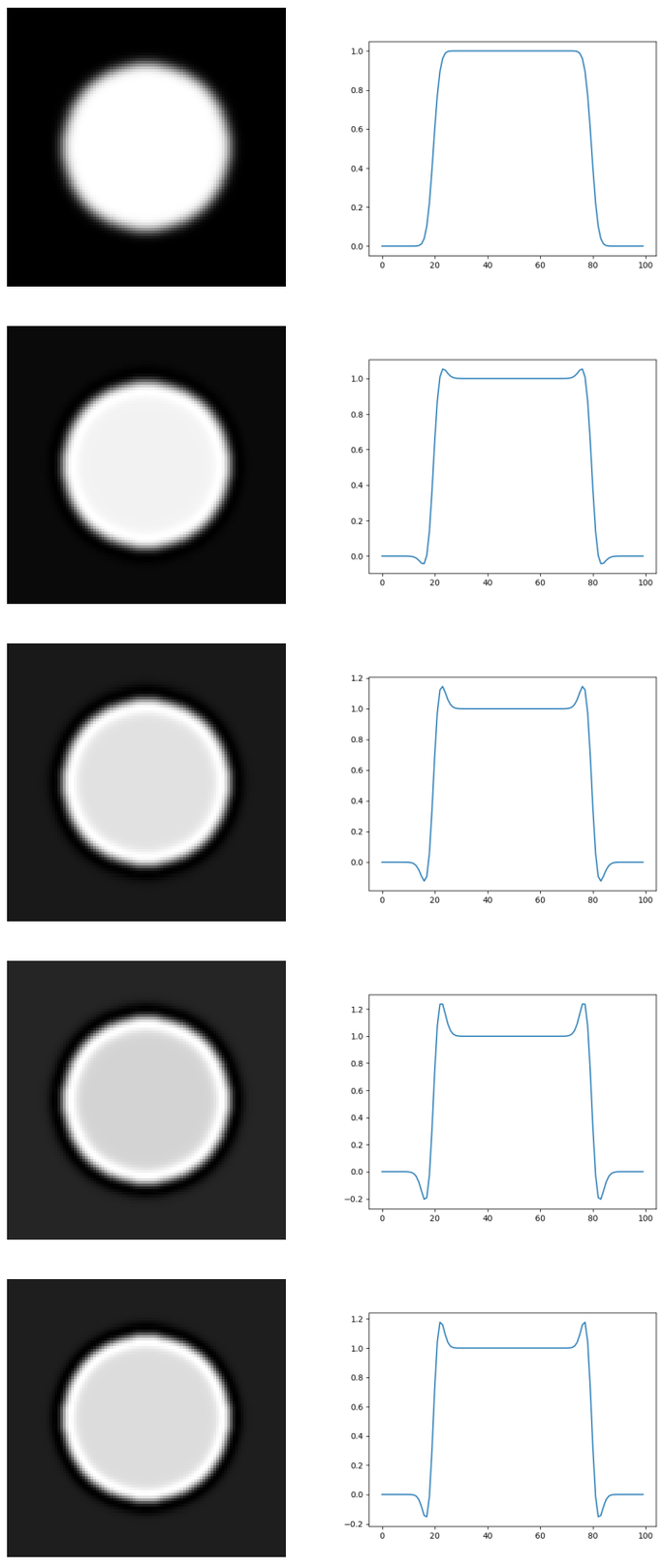
- Edge in a wide sense: also textures, details
- Amplification of high-frequencies in image
  - Locate high-frequency areas
  - Increase contrast locally in these areas
    - E.g., by increasing and decreasing intensity on each side of edge appropriately

Classical method: unsharp masking (USM)

- Intuition
  - Image I contains low and high frequencies
  - Blurred image $B^{\sharp 1}$ contains *only* low frequencies
  - "I − B" contains *only* high frequencies
  - Adding I − B to I emphasizes high frequencies
- Formulation
  - Sharpened image $S = I + \alpha(I - B)$
  - α: sharpening amount (or simply "amount")



$^{\sharp 2}$



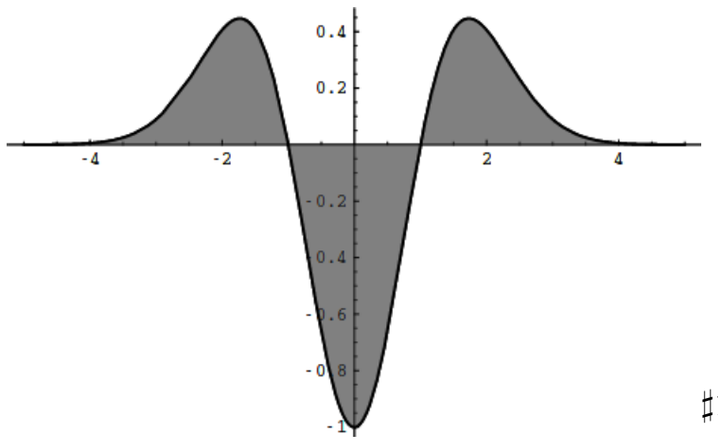
$^{\sharp 1}$Obtained by low-pass filtering: $B = I * F_\sigma$, where F_σ is low-pass filter with *blurring power* σ

$^{\sharp 2}$From Wikipedia

Linear Structure Enhancement [Frangi *et al*, 1998]

Based-on study of Hessian matrix at each pixel

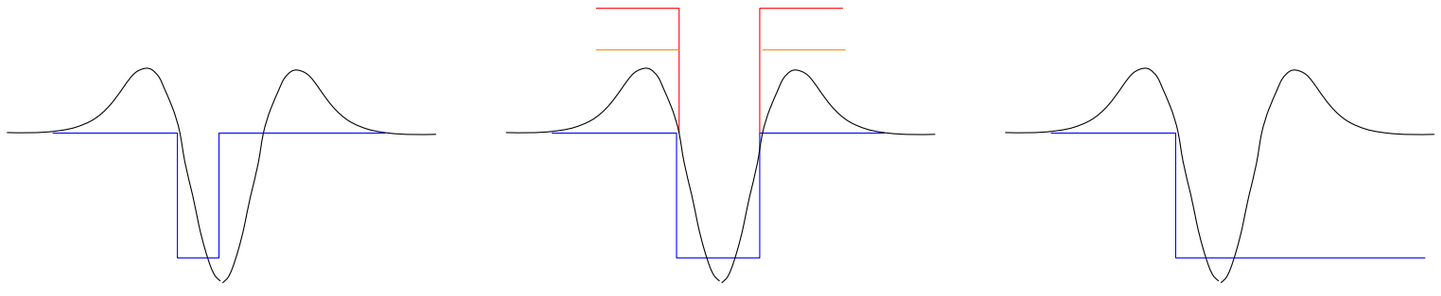
- Hessian matrix: $\begin{bmatrix} \frac{\partial^2 I}{\partial x^2} & \frac{\partial^2 I}{\partial x \partial y} \\ \frac{\partial^2 I}{\partial y \partial x} & \frac{\partial^2 I}{\partial y^2} \end{bmatrix}$ in 2-D and $\begin{bmatrix} \frac{\partial^2 I}{\partial x^2} & \frac{\partial^2 I}{\partial x \partial y} & \frac{\partial^2 I}{\partial x \partial z} \\ \frac{\partial^2 I}{\partial y \partial x} & \frac{\partial^2 I}{\partial y^2} & \frac{\partial^2 I}{\partial y \partial z} \\ \frac{\partial^2 I}{\partial z \partial x} & \frac{\partial^2 I}{\partial z \partial y} & \frac{\partial^2 I}{\partial z^2} \end{bmatrix}$ in 3-D
- Corresponding filter of given width (1-D)



♯1

Intuition

- When does the filter respond the most?



♯1 From Frangi *et al*, 1998

Linear Structure Enhancement [Frangi *et al*, 1998]

Based-on eigen decomposition of Hessian matrix

- Eigen decomposition (a.k.a. spectral decomposition) of matrix A
  - Find all real values λ_i and unit vectors v_i[♯1] s.t. $Av_i = \lambda_i v_i$
    - When they exist
    - Always exist for real symmetric matrices like Hessian matrix
  - Interpretation: directions along which vectors are stretched by A but keep their direction

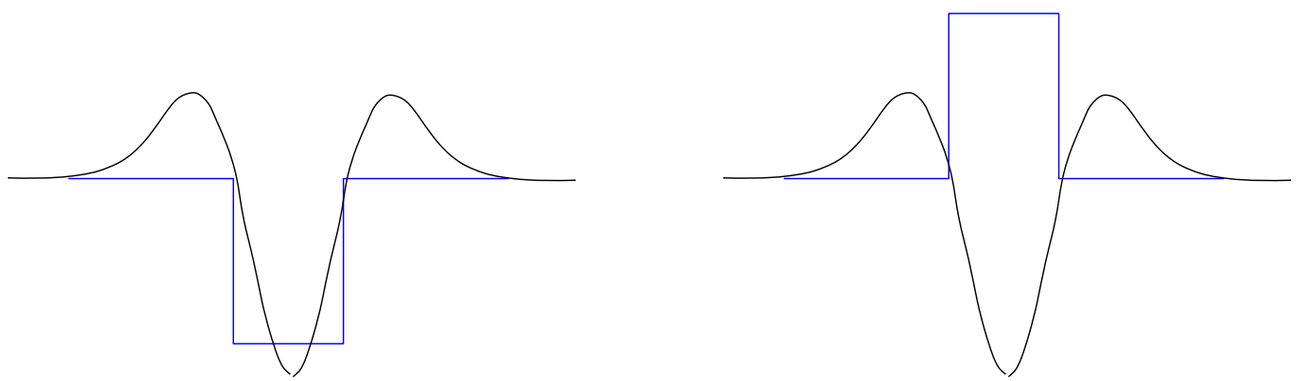
Hessian matrix and image structures

- λ_i: eigen values of Hessian matrix with $|\lambda_1| < |\lambda_2| < |\lambda_3|$
- Then:

	2-D		3-D		
Structure	λ_1	λ_2	λ_1	λ_2	λ_3
Line/Tube	L	H	L	H	H
Plate	*N/A*		L	L	H
Blob	H	H	H	H	H

where L/H means "Low value"/"High value in absolute value"

  - Sign of H: positive/negative ⇒ dark/bright structure



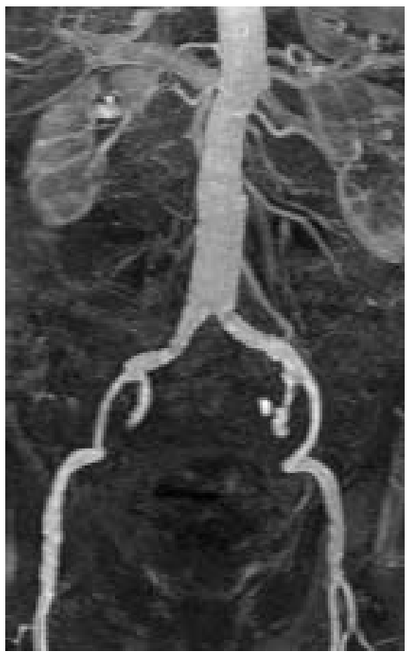
[♯1] λ_i: eigen value of A; v_i: eigen vector of A

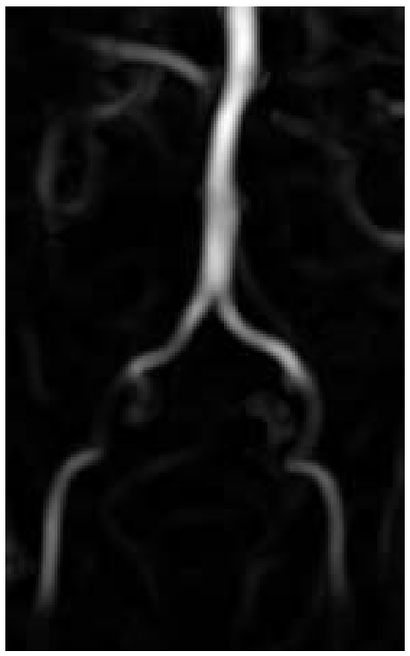
Linear Structure Enhancement [Frangi *et al*, 1998]

Multiscale procedure

- Compute bright tubeness measure T for a range of widths
  - $R_a = \frac{|\lambda_2|}{|\lambda_3|}, R_b = \frac{|\lambda_1|}{\sqrt{|\lambda_2 \lambda_3|}}, S = \sqrt{\sum \lambda_i^2}$
  - α, β, γ: sensitivity parameters of measure (i.e., parameters)
  - 2-D: $T = \exp\left(-\frac{R_b^2}{2\beta^2}\right)\left[1 - \exp\left(-\frac{S^2}{2\gamma^2}\right)\right]$ or 0 if $\lambda_2 > 0$
  - 3-D: $T = \left[1 - \exp\left(-\frac{R_a^2}{2\alpha^2}\right)\right]\exp\left(-\frac{R_b^2}{2\beta^2}\right)\left[1 - \exp\left(-\frac{S^2}{2\gamma^2}\right)\right]$ or 0 if λ_2 or $\lambda_3 > 0$
- Retain maximum response (among all widths) at each pixel

Example from Frangi *et al*, 1998



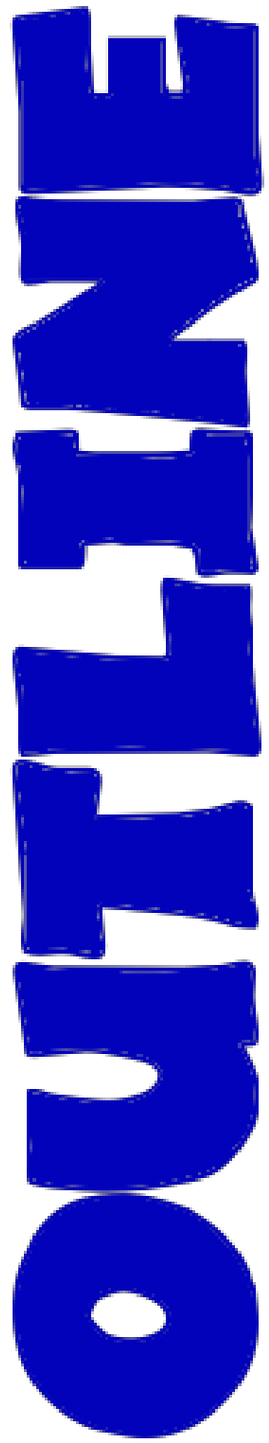


Filtering

Filtering (and other approaches) for...

Enhancing image quality

- Denoising (see ie_21)
- Deconvolution (see ie_28)
- Intensity scaling
- Contrast enhancement by histogram equalization
- Mathematical morphology



Intensity Scaling

Opening an image...

- ...It is all black

Image representations

- 4 planes: 3 color planes, 1 transparency plane (e.g., RGBA)
- 3 planes: 3 color planes (e.g., RGB)
- 2 planes: 1 grayscale plane, 1 transparency plane
- 1 plane: 1 grayscale plane

Plane depth: number of bits to represent a value

- Microscope: Typical acquisition depths = 8, 10, 12, 14, 16
- Image file: Typical storage depths = 1, 2, 4, 8, 16
- Screen: Display depth = 8

Intensity Scaling

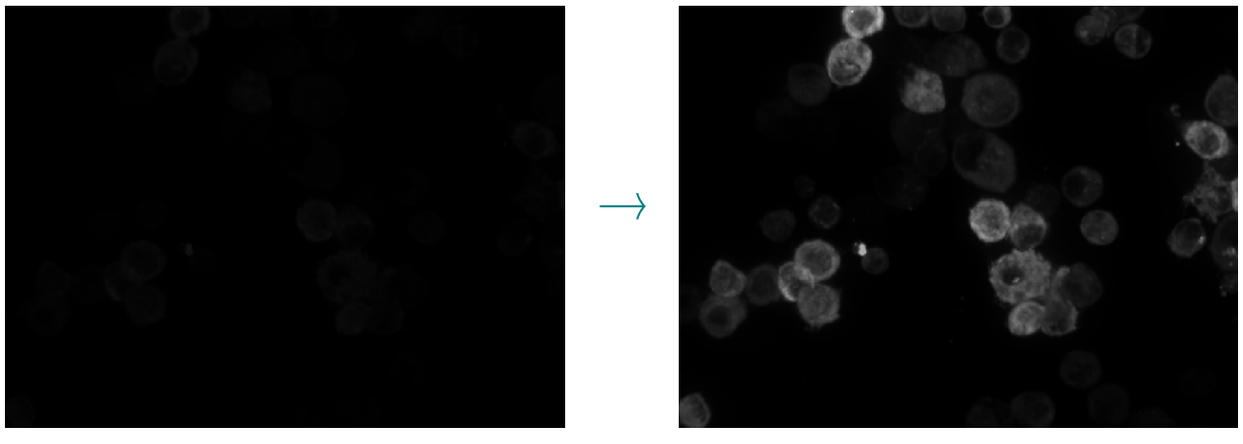
Depth correspondences

Acquisition	Storage	Operation	Display
8	8	Nothing	8
10, 12, 14, 16	16	*See below*	8

Acq depth	8	10	12	14	16
Max rep value[♯1]	255	1023	4095	16383	65535
Acq/Storage_16		0.015	0.06	0.25	1

- Operations
  - Nothing
    - But image viewer **will** do something since display only accepts 8-bit planes
    - It might do "$255 \times \underbrace{\underbrace{\text{plane}/(2^{\text{storage depth}}-1)}_{\in[0,1]}}_{\in[0,255]}$" $\Rightarrow$ dark, or very dark, or almost all black image
  - Acq-related scaling: $255 \times \text{plane}/(2^{\text{acquisition depth}}-1)$
  - Max-based scaling: $255 \times \text{plane}/\max(\text{among all planes})$

Opening an image, taking care of the acquisition depth



Op = nothing (left to img viewer) *Op = 255 × plane/ max*

♯1 $2^n - 1$ = max representable value on n bits (0,1), just like $10^n - 1$ = max representable value on n digits (0,1,$\cdots$,9) (e.g., 2 digits $\Rightarrow 10^2 - 1 = 99$)

Histogram Equalization

Purpose

- Increase image contrast globally

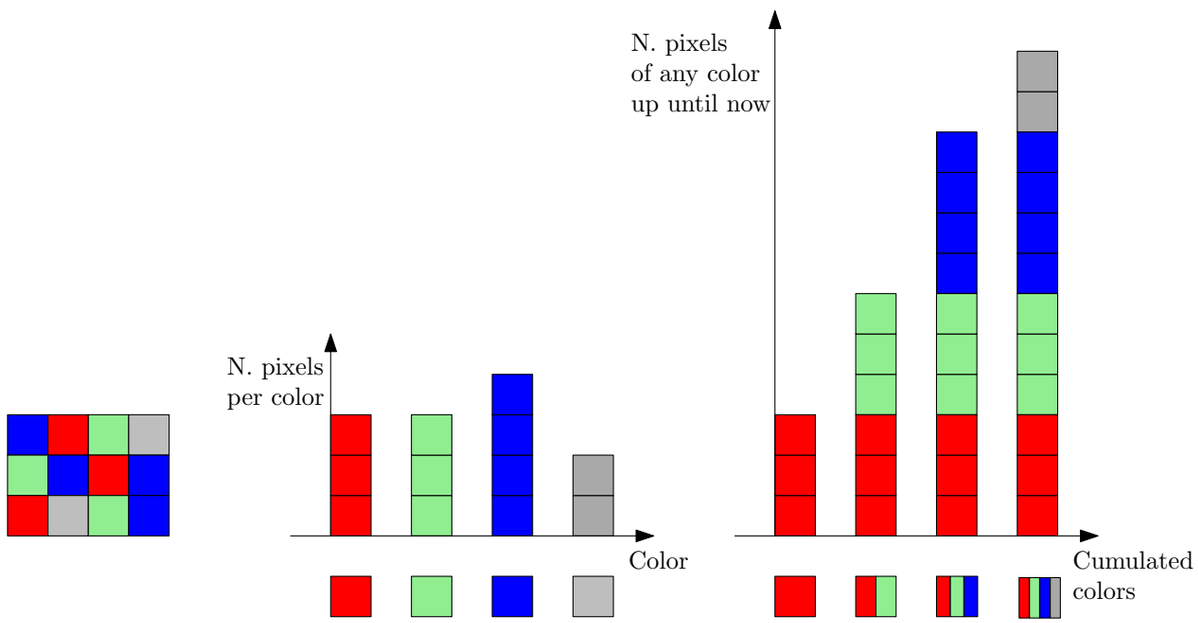
Principle

- Redistribute intensities of the histogram (1) $\rightarrow$ constant histogram (2)
  - Most frequent intensities (1) replaced with least frequent ones

  $\Rightarrow$ all intensities equally frequent (2)

Properties

- Invertible transformation
- Simple and fast
- Useful if relative intensities more important than absolute values of intensities



Histogram Equalization

Notations

- Image I
  - Size $W \times H$ ($N = W \times H$) and
  - Intensity range $[0, L]$ (8-bit plane $\Rightarrow L = 255$)
- Histogram
  - $h_I(i)$ = number of pixels of intensity i in I
  - Property: $\sum_i h_I(i) = N$
- Normalized histogram
  - $p_I(i)$ = (n.o. pixels of intensity i) / N
  - Property: $\sum_i p_I(i) = 1$
- Cumulative norm. histogram
  - $c_I(i) = \sum_{j=0}^{i} p_I(j)$
  - Property: $c_I(i >= \text{max intensity in } I) = 1$

Problem

- Find intensity transformation $i \rightarrow T(i)$ such that new image J has
  - norm. histogram $p_J(i) \simeq 1/(L+1)$

  $\Leftrightarrow$ cumulative norm. histogram $c_J(i) \simeq i/L$

Details about $p_J(i) \simeq 1/(L+1)$ and $c_J(i) \simeq i/L$

- $p_J(i)N \stackrel{\triangle}{=} h_J(i) = N/(L+1)$
  - Must be integer...
  - ...but N not a multiple of $L+1$ in general!
- $c_J(0) = \sum_{j=0}^{0} p_J(j) = p_J(0)$
  - Then $c_J(0)$ must be equal to $1/(L+1)$

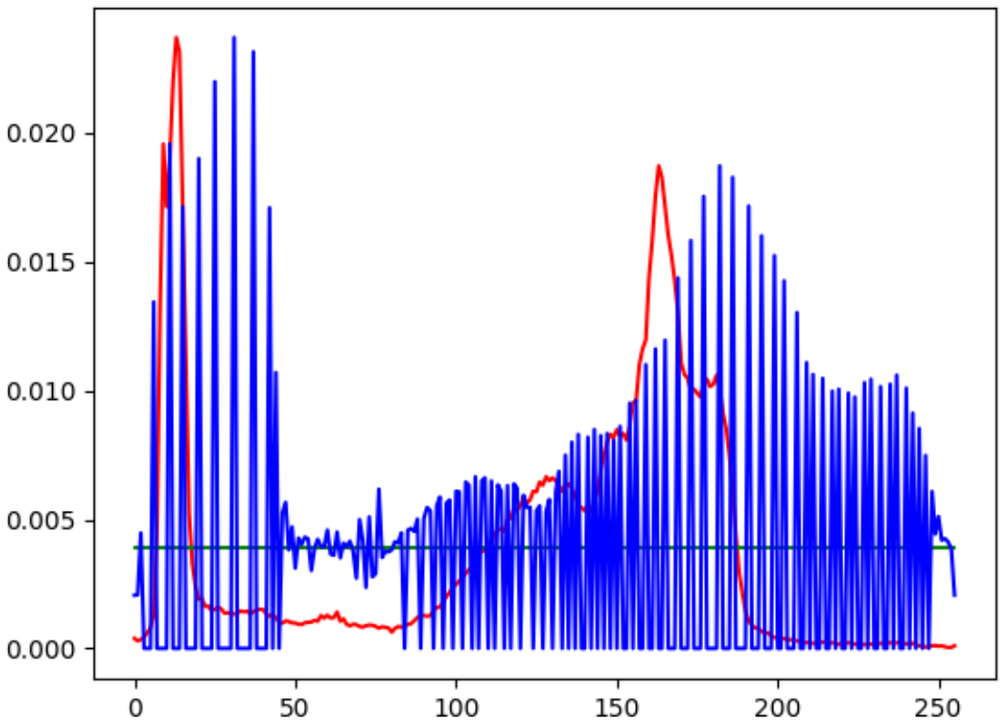
  $\Rightarrow c_J(i) = (i+1)/(L+1)$
- Only approximated solution

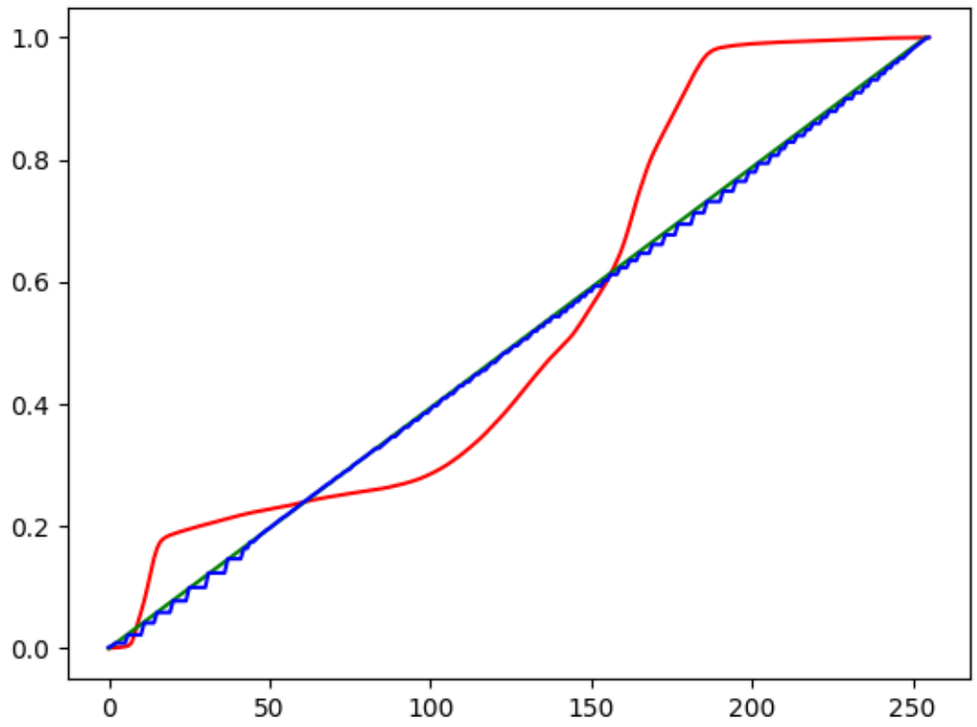
Solution

- $I_{0L} = L \dfrac{I - \min(I)}{\max(I) - \min(I)}$
  - I_{0L} = version of I with intensities fully covering $[0, L]$
- For any $i \in [0, L]$, $T(i) = L \dfrac{c_{I_{0L}}(i) - c_{I_{0L}}(0)}{1 - c_{I_{0L}}(0)}$
- Or directly for any $i \in [\min(I), \max(I)]$
  - $T(i) = L \dfrac{c_I(i) - c_I(\min(I))}{1 - c_I(\min(I))}$

Histogram Equalization

Illustration



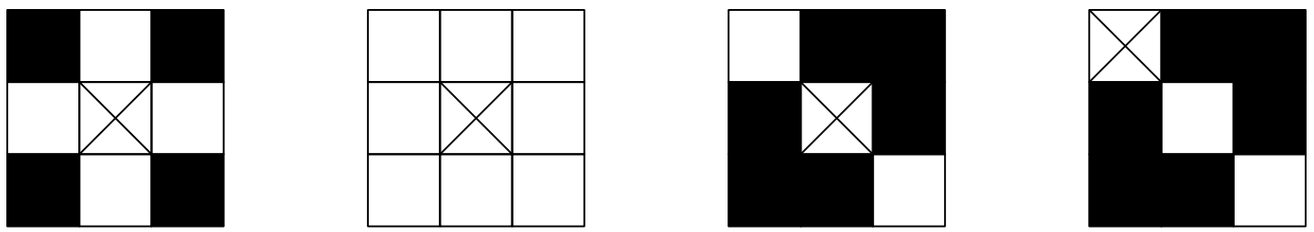


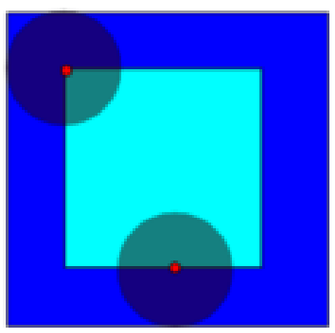
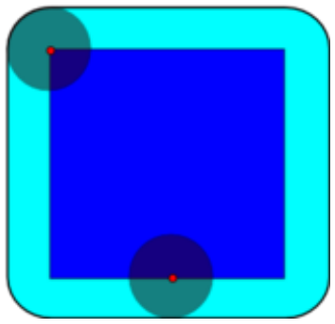
Warning

- *Filtering*: not in convolutional sense
- *and More*: detection (see future classes)

Analysis and processing of geometrical structures (in binary images)

- One basic object: structuring element S = a small shape w/ a ref. point
  - Like a filter, but binary ⇒ better described as a mask



- Two basic operations
  - Erosion E:  and dilation D:  (Images from Wikipedia)
- Two basic compositions
  - Opening = erosion followed by dilation, or dilation of the erosion: $D(E(I))$
  - Closing = dilation followed by erosion, or erosion of the dilation: $E(D(I))$
- Then other combinations
  - Morphological gradient, top-hat filter…

Opening

- First erosion
  - Objects *smaller* than S disappear
  - Objects *larger* than S shrink
- Then dilation
  - Shrunk objects grow back to their *original size and shape*[♯1]
- Overall: opening can be used to clean up an image by removing small objects (noise)

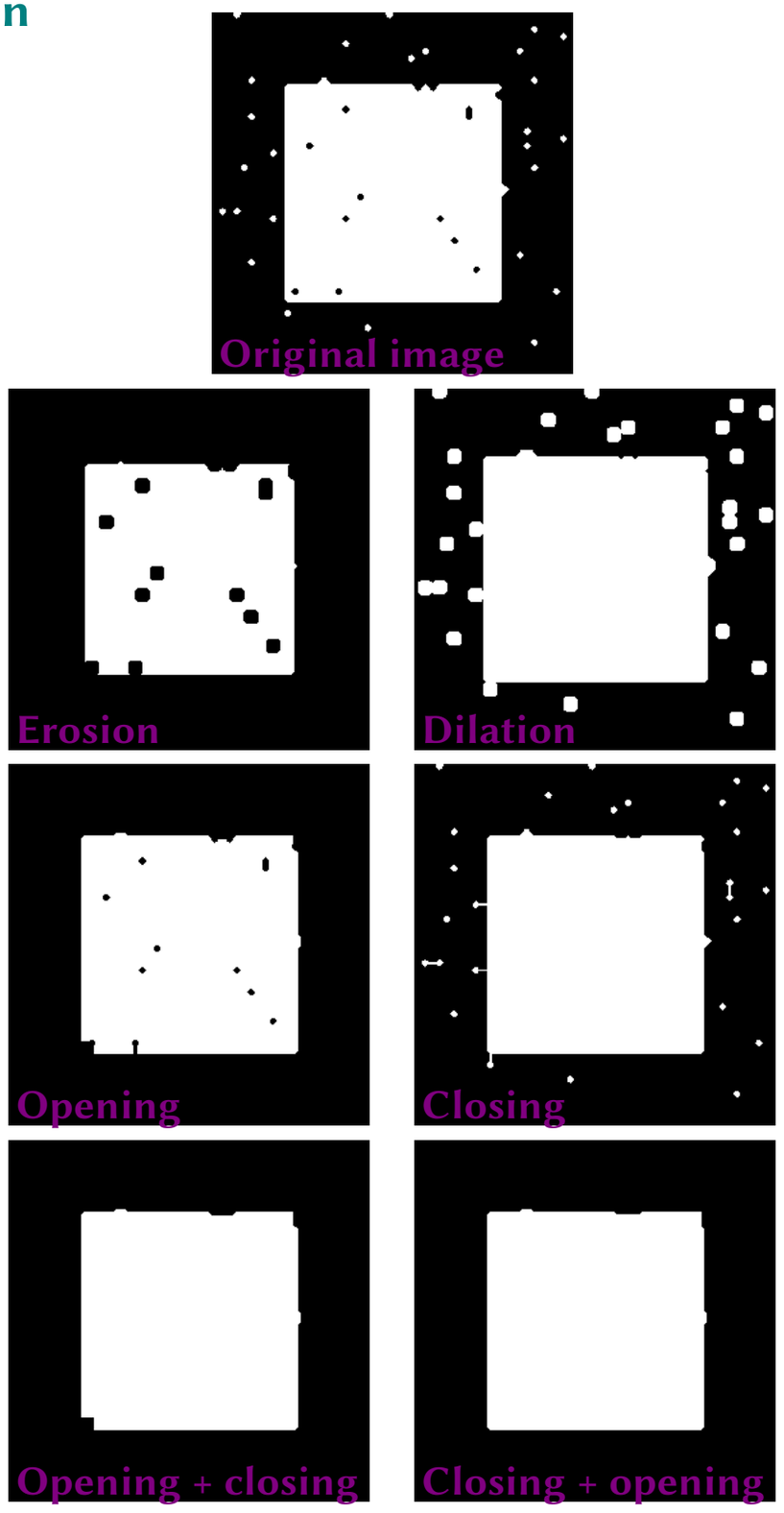
Closing

- First dilation
  - Holes *smaller* than S disappear (they get filled)
  - Objects grow
- Then erosion
  - Enlarged objects shrink back to their *original size and shape*[♯2]
- Overall: closing can be used to clean up objects by filling their holes

[♯1] Not exactly, but close

[♯2] Not exactly, but close

Illustration



Mathematical Morphology

Caution

- If reasoning in terms of inclusion (erosion) and non-empty intersection (dilation)
- Then the symmetric structural element S^{sym} is used for the dilation (instead of S)

Binary images $\rightarrow$ grayscale images

- Structural element $\rightarrow$ structural function
- Usually flat (i.e. piecewise constant) functions are used
  - Then replace...
    - "ref. point $\leftarrow$ 1 if S totally inside structure" with "ref. point $\leftarrow$ min intensity over S"
    - "ref. point $\leftarrow$ 1 if S^{sym} partially inside structure"with "ref. point $\leftarrow$ max intensity over S^{sym}"

Illustration