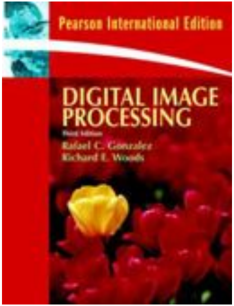
Computer Assisted Image Analysis
Lecture 2 – Point Processing

Anders Brun (anders@cb.uu.se)
Centre for Image Analysis
Swedish University of Agricultural Sciences
Uppsala University

Reading Instructions

Chapters for this lecture



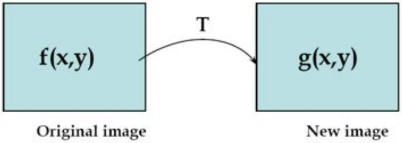
- Chapter $2.6 - 2.6.4$ and $3.1 - 3.3$ in Gonzales-Woods.

Digital images

- Images denoted by functions, e.g. $f(x, y)$ or $g(x, y)$
- Sampling in space, e.g. $(x,\, y) \in I$ and $||I|| = N$, where I is a discrete set of pixel positions.
- Quantization in amplitude (intensity), $f(x, y) \in \{0, 1, \ldots (L-1)\}$

Image Processing

In image processing, the operator T transforms the input image into an output image, $g(x, y) = T(f(x, y))$.



Typical examples of image processing

- Image restoration: reduce noise and imaging artefacts
- Image enhancement: enhance edges, lines and subtle features for easier visual inspection
- Feature extraction, as input to subsequent image analysis

Image processing does NOT increase image information!

Image Processing

- Spatial domain (lectures 2, and 3)
    - Brightness transforms, works per pixel $\rightarrow$ point processing,
    - Spatial filters, local transforms, works on small neighborhoods,
    - Geometric transforms, interpolation,
- Frequency domain (lecture 3 and 4).
    - The Fast Fourier Transform (FFT)
    - Lowpass-, bandpass- and highpass filters,
    - ...

Image Processing

In spatial domain processing, the operator T is applied to each position $(x,\, y)$ in the input image f, defined over some neighborhood of (x, y), yielding a value $s = g(x,\, y)$ as output.

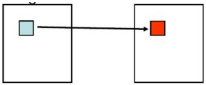
$$g(x,\, y) = T[f(x,\, y)]$$

In point processing, the operator neighborhood is the pixel itself.

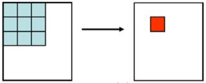
$$s = T(r), \text{where}\, r = f(x, y), s = g(x, y).$$

In spatial filtering, larger neighborhoods are used. They are referred to as masks, filters, kernel windows or templates.

Point processing



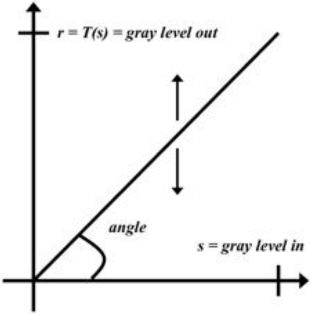
Spatial filters



Gray Level Transform

Pixel-wise transform

- Change the gray level for each individual pixel.
- Compare to television: Brightness and contrast
  - brightness: addition
  - contrast: multiplication



$> 45^\circ$	$\rightarrow$	increased contrast
$< 45^\circ$	$\rightarrow$	decreased contrast
up	$\rightarrow$	increased brightness
down	$\rightarrow$	decreased brightness

Image Histograms

A gray level histogram shows how many pixels there are at each intensity level. The bars either sum up to the total number of pixels, or to 1 (normalized) in a histogram.

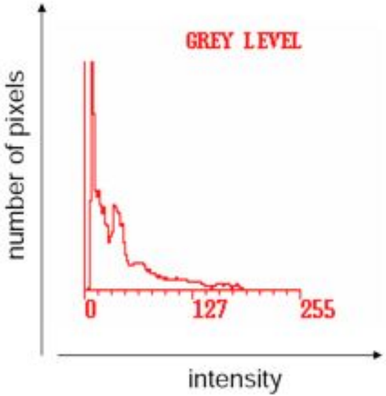
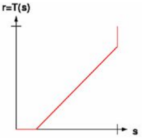
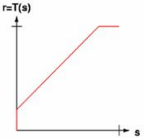


Image Processing

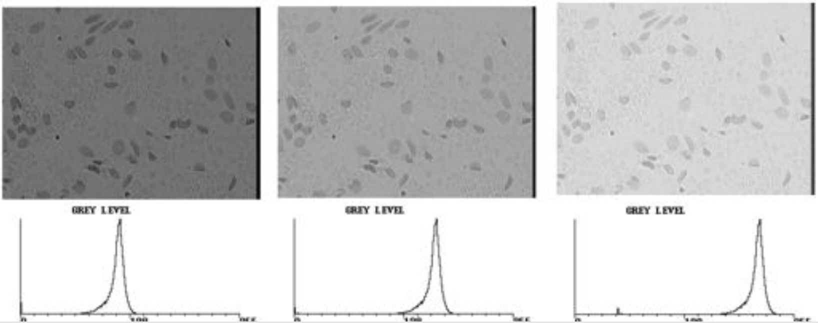
Brightness

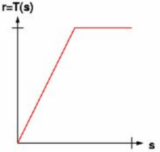


Subtract.

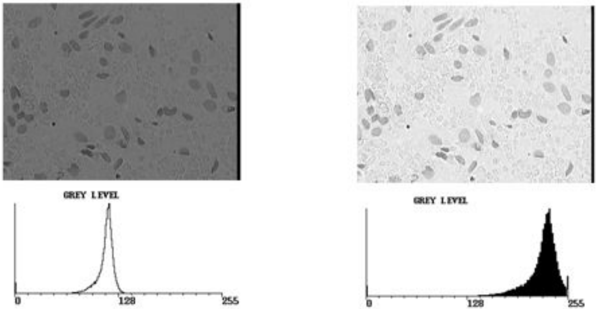


Add.



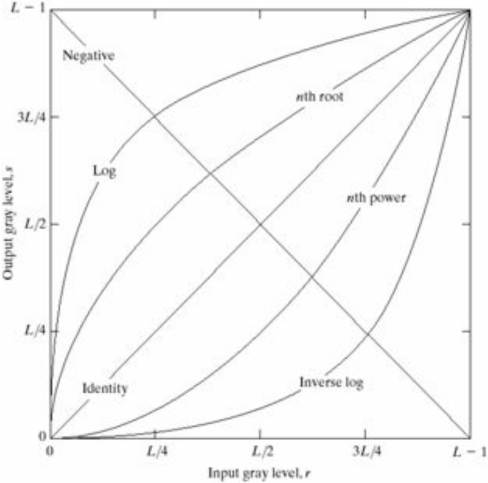


Multiply



Gray Level Transformations

Some basic gray level transformation functions used for image enhancement.



Gray Level Transformations

Original image

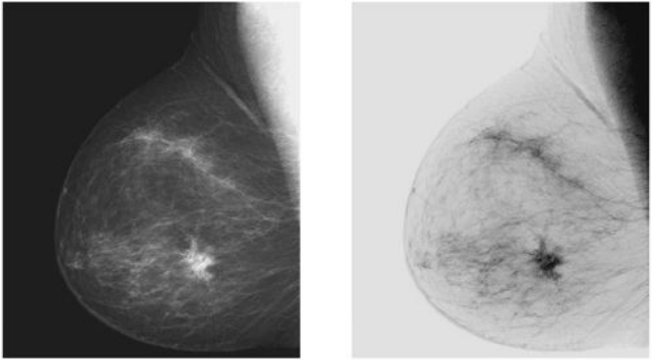
(Neutral transform)

Inverse transform (Negative)

Logarithmic transform

Gray Level Transformations

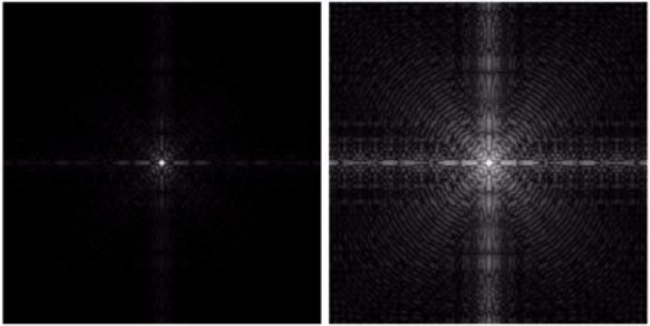
Negative or positive



- Original digital mammogram (left).
- Image negative to enhance white or gray details embedded in dark regions (right).

Gray Level Transformations

Log transformations



Visualize patterns in the dark region of an image

- Fourier spectrum (left).
- Result of applying the log transform (right).

Histogram Equalization

Idea: Create an image with evenly distributed gray levels, for visual contrast enhancement

- The normalized gray level histogram gives the probability for a pixel to have a certain gray level, $p_k = n_k/N$
- Transform the image using the cumulative density function, $\mathrm{cdf}(k) = \sum_{i=0}^{k} p_i$ (or $= \int_{i=0}^{k} p(i)di$ in the continuous case)

Histogram Equalization

- Continuous formula, where $p(i)$ is the probability measure of the i grayvalue in the image if $s, r \in [0, L]$

$$s = T(r) = L \int_{i=0}^{r} p(i) = L \,\mathrm{cdf}(r)$$

- Discrete formula, where n_i is the number of pixels with intensity i and N is the total number of pixels and s_k and $r_k \in \{0, 1, \ldots, (L-1)\}$:

$$s_k = T(r_k) = (L-1)\frac{\sum_{j=0}^{k} n_j}{N}$$

- Both formulas try to stretch r_{min} to 0 and r_{max} to either L or $(L-1)$ (But do they succeed?)
- The histogram for the output image is uniform (*theoretically* in the continuous case), why not in our digital images?

Histogram Equalization

Why does this work?

- Let p_r be the normalized histogram (probability function) for the input image $f(x, y)$
- Transform $f(x, y)$ using $s = T(r) = L \int_0^r p_r(w)dw$
- Leibniz´s Rule $\frac{ds}{dr} = \frac{dT(r)}{dr} = L\frac{d}{dr}\left[\int_0^r p_r(w)dw\right] = Lp_r(r)$.
- Then from probability theory we have a formula for the probability density function (histogram) of the transformed variable (image), p_s

$$
\begin{aligned}
p_s &= p_r(r)|\frac{dr}{ds}| \\
&= p_r(r)\left|\frac{1}{Lp_r(r)}\right| \\
&= 1/L
\end{aligned}
$$

Histogram Equalization

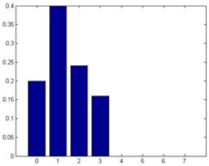
Original image.

Result of histogram equalization.

Histogram Equalization Example

Intensity	0	1	2	3	4	5	6	7
Number of pixels	10	20	12	8	0	0	0	0

$$p(0) = 10/50 = 0.2$$
$$p(1) = 20/50 = 0.4$$
$$p(2) = 12/50 = 0.24$$
$$p(3) = 8/50 = 0.16$$
$$p(r) = 0/50 = 0, r = 4, 5, 6, 7$$

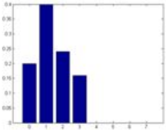


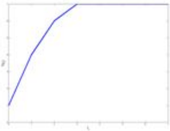
Histogram Equalization Example (cont.)

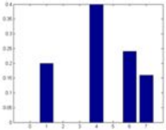
$$s_k = T(r_k) = (L-1)\frac{\sum_{j=0}^{k} n_j}{N} = (L-1)\sum_{j=0}^{k} p(j)$$

$$T(0) = 7*(p(0)) \approx 1$$
$$T(1) = 7*(p(0)+p(1)) \approx 4$$
$$T(2) = 7*(p(0)+p(1)+p(2)) \approx 6$$
$$T(3) = 7*(p(0)+p(1)+p(2)+p(3)) = 7$$
$$T(r) = 7, r = 4,5,6,7$$

Intensity	0	1	2	3	4	5	6	7
Number of pixels	0	10	0	0	20	0	12	8

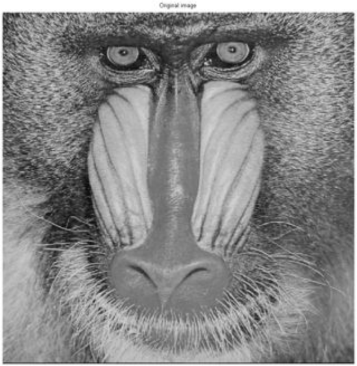






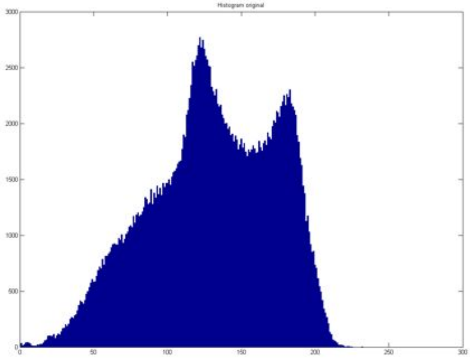
Histogram Equalization

Example: Original image $f(x, y)$



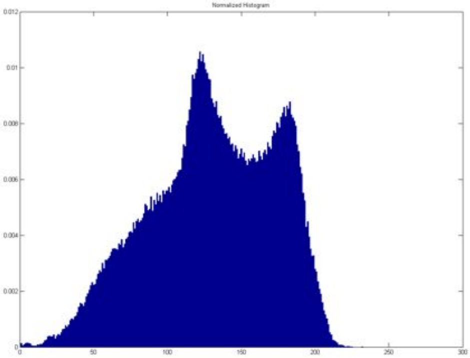
Histogram Equalization

Example: Histogram



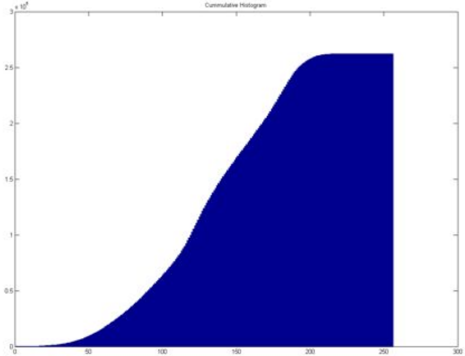
Histogram Equalization

Example: Normalized histogram



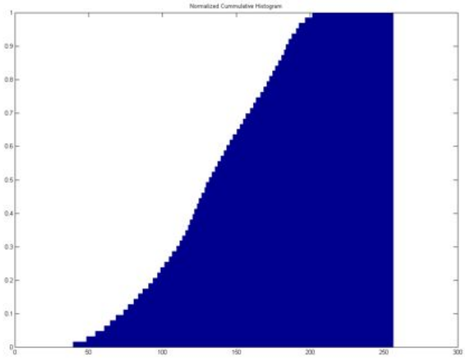
Histogram Equalization

Example: Cumulative histogram



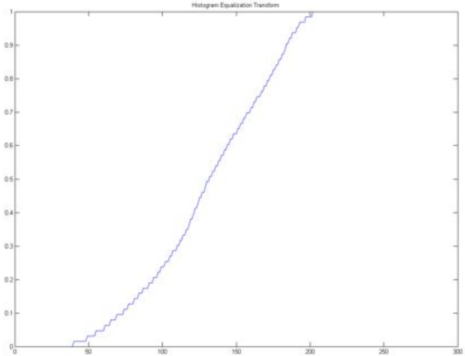
Histogram Equalization

Example: Normalized cumulative histogram



Histogram Equalization

Example: Histogram equalization transform

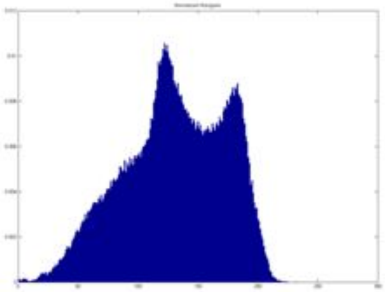


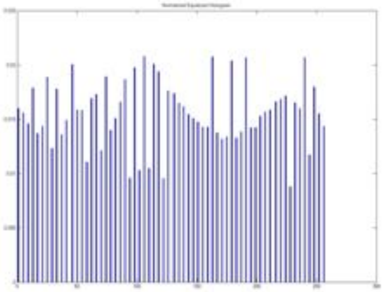
Histogram Equalization

Example: Histogram equalization

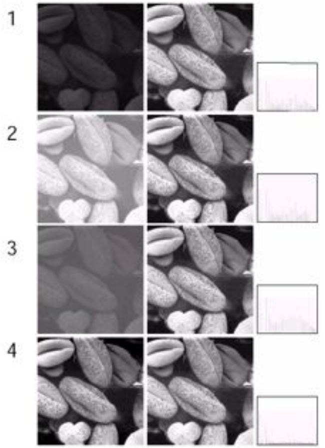
Histogram Equalization

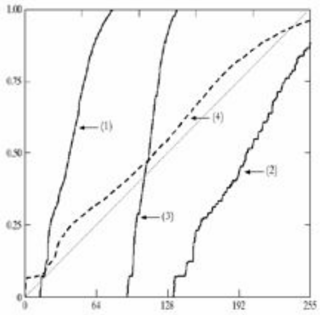
Example: Equalized histograms





Histogram Equalization





Transformations for image $1 - 4$. Note that the transform for figure 4 (dashed line) is close to the neutral transform (dotted line).

Histogram Equalization

Not always "optimal" for visual quality

Original.

Equalized.

Manual choice.

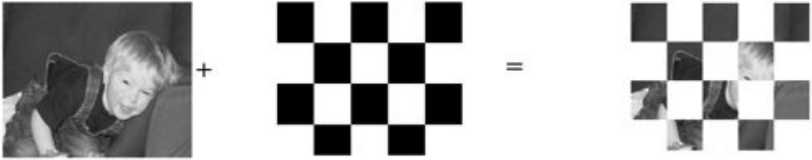
Histogram Matching

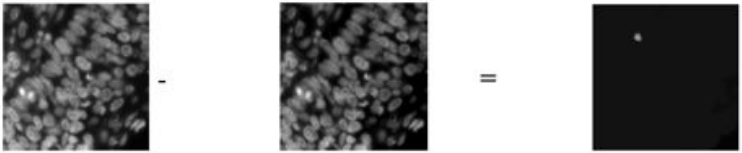
Transform image $f(x, y)$ to match the histogram of image $g(x, y)$

- If $s = T(r)$ maps $f(x, y)$ to a uniform histogram
- and $u = G(t)$ maps $g(x, y)$ to a uniform histogram
- Then $s = G^{-1}(T(r))$ maps $f(x, y)$ to have a histogram similar to $g(x, y)$

Arithmetic/Logical Operations

- Information from two different images with the same size can be combined by adding, subtracting, multiplying or comparing the pixel values, pixel by pixel. Rounding to fit $[0, L-1]$.
- For enhancement, segmentation, change detection.





Arithmetic/Logical Operations

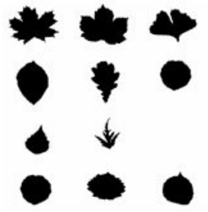


Image 1.

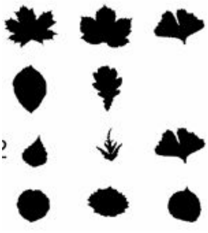
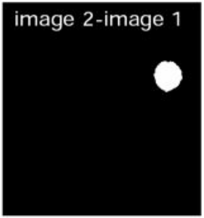
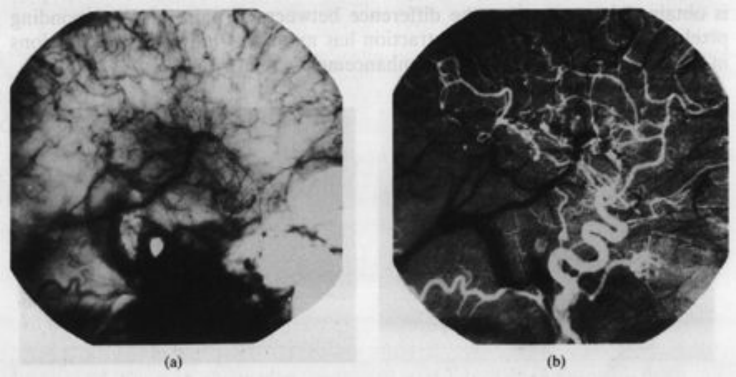


Image 2.



Arithmetic/Logical Operations

Enhancement by image subtraction



(a) Mask image.

(b) Image (after injection of dye into the bloodstream) with mask subtracted out.

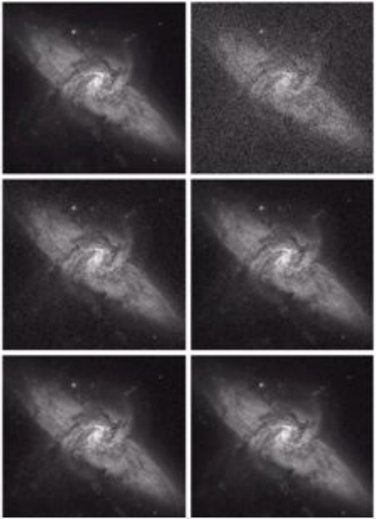
Arithmetic/Logical Operations

Images as Vectors

- We may regard images as vectors, i.e. a ordered set of scalars
- All pointwise arithmetic works for both images and vectors
- In fact, sometimes even the geometrical interpretation of vectors is natural for images, e.g. orthogonality
- However, by subtracting two images we may end up with *negative* pixel values. What is that?! Negative coefficients are natural for vectors, but *not* for e.g. light intensities or densities.
- Solution: Let´s not care too much about that ... Deal with negative, very large and floating-point values by rounding to the closest integer in $[0, L-1]$ before saving the resulting image.

Arithmetic/Logical Operations

Reduction of noise by averaging



Noise can be reduced by observing the same scene over a long period of time, and averaging the images. Top: original and a noisy image. Then noisy images averaged 8, 16, 64 and 128 times.

- Averaging yields a normally distributed resulting image (Central Limit Theorem)
- Averaging approaches the expected value of the noisy images (Law of large numbers)
- The standard deviation, after averaging M noisy uncorrelated images with standard deviation σ, is $\frac{1}{\sqrt{M}}\sigma$.
- (However, this only works for noise or image artefacts with expectation value zero, i.e. it is fine for Gaussian distributed noise but not for Poisson distributed noise.)

Linear vs Non-Linear Operations

- An operator H is linear if
  - $H[a_i f i(x,y) + a_j f_j(x,y)] = a_i H[f_i(x,y)] + a_j H[f_j(x,y)]]$
- Linear operators have properties that make them useful in image analysis, in particular for image filtering
- The class of non-linear operators is huge
- Example: $\sin$ is non-linear ("The Freshman´s dream")
  - $\sin(f_i(x,y)) + \sin(f_j(x,y)) \neq \sin(f_i(x,y) + f_j(x,y))$

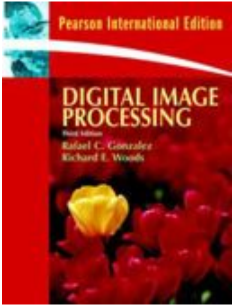
Vectorization vs Looping

- In Matlab, it is often useful to vectorize code: Operate on all pixels in an image at once. (`.*`, `./`, `+`, `-`)
- For-loops are slower in Matlab.
- However, in languages such as C, for-loops are fast!
- Good to know how to vectorize code in Matlab, as well as how to construct for-loops that are more useful in C.

Lab 1

- The first lab contains a mix of things to get you started
- Unfortunately, it is scheduled early, so some concepts such as local operators have not been introduced
- Read ahead and ask for help!

Review Questions



- Problems 2.22, 2.18, 2.9, 3.1, 3.5 and 3.6 in Gonzales-Woods.
- Download answers from `http://www.imageprocessingplace.com`.

Until Next Lecture

- Read, read, read
- Experiment in Matlab
- Do the review questions