

Image Enhancement

DD2423 Image Analysis and Computer Vision

Mårten Björkman

Computational Vision and Active Perception
School of Computer Science and Communication

November 11, 2015

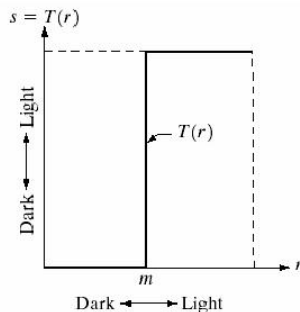
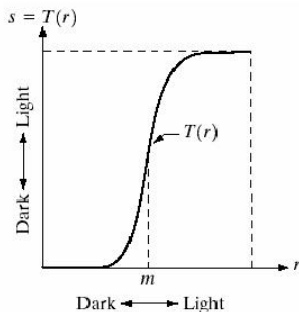
- Goal: Improve the subjective quality of the image.
- Examples:
 - Contrast enhancement
 - Noise suppression - smoothing
 - Sharpening
 - Feature enhancement
- Assumption:
 - no degradation model, otherwise its called restoration.

Image enhancement by gray-level transformations

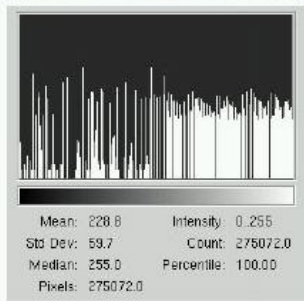
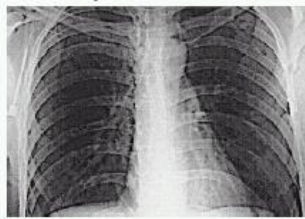
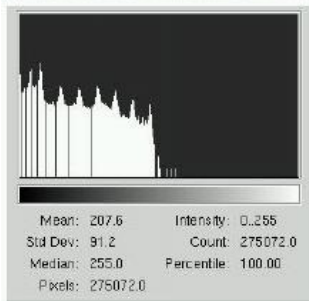
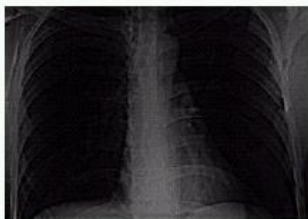
- Gray-level transformations

$$s = T(r)$$

where s and r are intensities after and before, and T may be piecewise linear, negative, logarithm transformations.

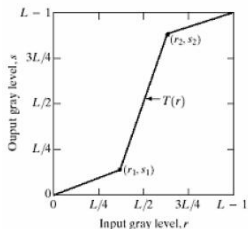
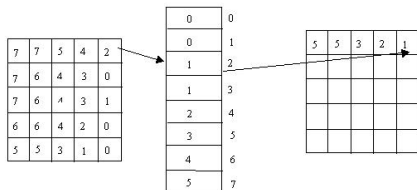


Histogram



Look-Up Tables (LUT)

Often implemented with LUTs (256 entries), at least for complex functions. Sometimes included directly in the camera.



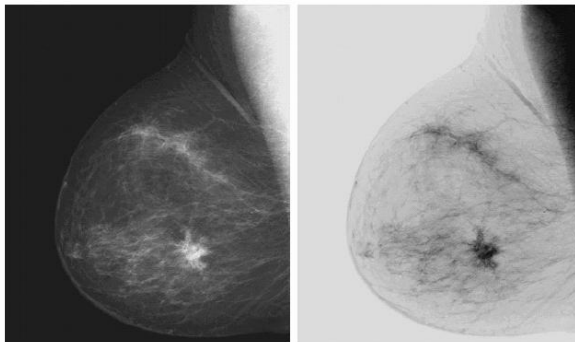


Image Negatives

(b) is simply the negative of (a), ie

$$s = L - 1 - r$$

where $L = 2^k$

of gray levels

of bits

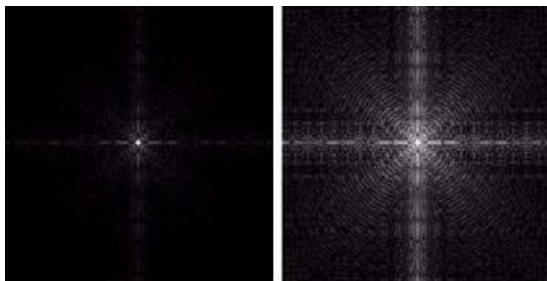
Log transformations

Useful for compressing large dynamic range and make details visible.

$$s = c \log(1 + r)$$

Example: Fourier spectrum

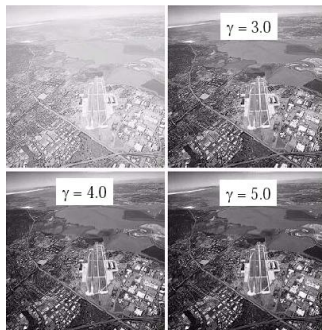
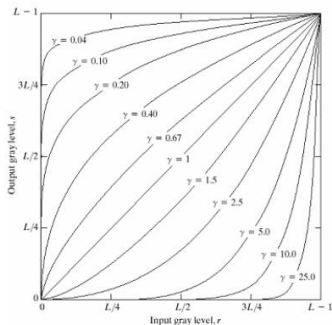
$0 \rightarrow 1.5 \times 10^6$ to $0 \rightarrow 6.2$



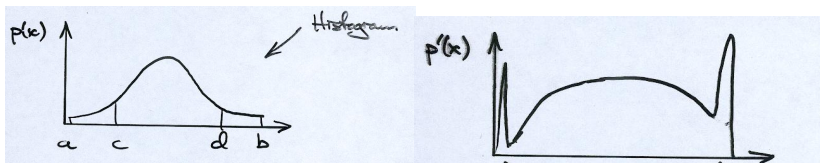
Power-law transformations

A variety of devices used for image capture, printing, and display respond according to a power law.

$$s = c r^{\gamma} \quad \text{or} \quad s = c (r + \epsilon)^{\gamma}$$



Histogram Stretching

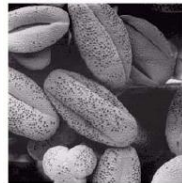
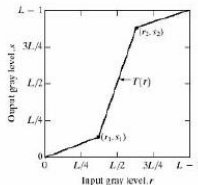


Increase contrast by letting the interval $[c, d]$ cover the entire gray-level range. Note: Information loss in $[a, c]$ and $[d, b]$.

Full-Scale
Histogram Stretch

$$(r_1, s_1) = (r_{\min}, 0)$$

$$(r_2, s_2) = (r_{\max}, L-1)$$



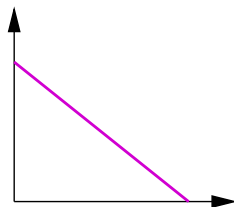
Gray-level transformations

Common requirements on transformation function $s = T(r)$:

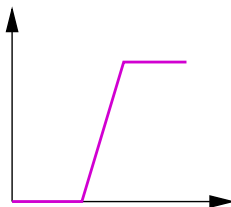
$T(r_{min}) = r_{min}$ (or opposite) - fills up entire range of gray-levels

$T(r_{max}) = r_{max}$

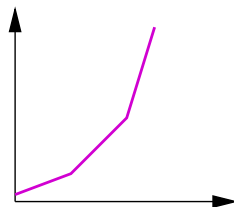
T monotonic $\Rightarrow T$ invertible (no loss of information)



**contrast
reversal**



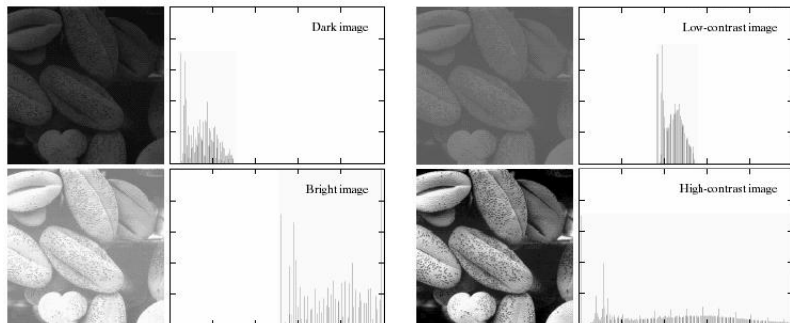
**hard
stretching**



**stepwise
linear**

Common special cases

Four images and their histograms



- Idea: Redistribute gray-levels as evenly as possible - this would correspond to a brightness distribution where all values are equally probable.
- Assume gray levels are continuous (not quantized) and have been normalized to lie between 0 and 1.
- Find transformation T that maps gray values r in the input image to gray values $s = T(r)$ in the transformed image.

Histogram equalization (continuous case)

We look for a transformation $s = T(r)$ such that the distribution $p_S(s)$ of pixel values is uniform, given a distribution from an image $p_R(r)$.

Known from probability theory:

$$p_S(s) = \left[p_R(r) \frac{dr}{ds} \right]_{r=T^{-1}(s)}$$

Let us define $T(r)$ as

$$s = T(r) = \int_0^r p_R(w) dw \Rightarrow \frac{ds}{dr} = p_R(r)$$

Then it follows that

$$p_S(s) = \left[p_R(r) \frac{dr}{ds} \right]_{r=T^{-1}(s)} = \left[p_R(r) \frac{1}{p_R(r)} \right]_{r=T^{-1}(s)} = 1$$

Histogram equalization (discrete case)

1. Compute histogram: count each distinct pixel value in the image.
2. Store cumulative sum of all the histogram values and normalize them by dividing each element by the number of pixels.

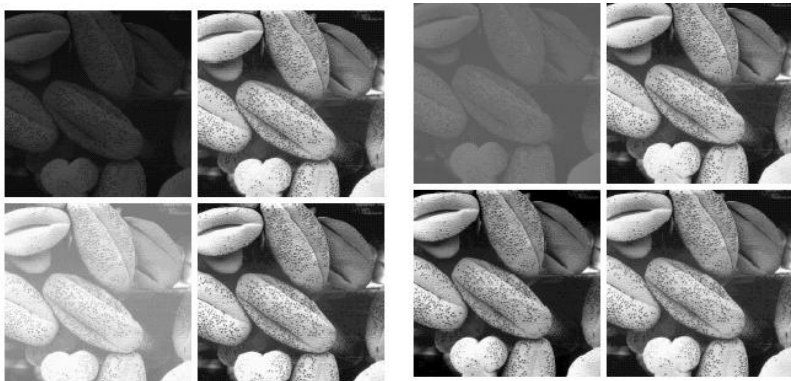
$$s_k = T(r_k) = \sum_{i=0}^k p_r(r_i) = \sum_{i=0}^k \frac{n_i}{N}, \quad 0 \leq r_k, s_k \leq 1, \quad k = 0, 1, \dots, 255$$

3. Use LUT from step 2 to transform the input image.



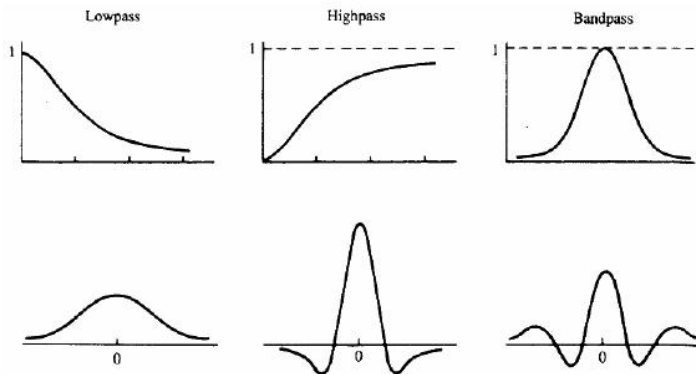
Note! Typically values of s_k are scaled up by 255 and rounded to the nearest integer so that the output values are between 0 and 255. Due to discretization the transformed image will not have a perfectly uniform histogram.

Four images after histogram equalization



- Spatial filtering: Filtering using spatial masks.
Spatial filters can be either linear or nonlinear.
- Linear filters can be:
 - Lowpass: eliminate high frequency components such as characterized by edges and sharp details in an image.
⇒ Net effect is image blurring.
 - Highpass: eliminate low frequency components such as slowly varying characteristics (shadings).
⇒ Net effect is sharpening of edges and other details (also noise).
 - Bandpass: eliminate outside a given frequency range.
⇒ Combination of the above. Common in practice.

Spatial filtering (examples)



Some filters in frequency domain and corresponding spatial filter masks.

- Assume you have a filter kernel $[1, 0, -1]$. How does this look like in the Fourier domain? Is it a lowpass, highpass or bandpass filter?

- Assume you have a filter kernel $[1, 0, -1]$. How does this look like in the Fourier domain? Is it a lowpass, highpass or bandpass filter?

Answer: To see this we have to express the filter in continuous domain, which we can do with Dirac functions.

$$h(x) = \delta(x) - \delta(x - 2)$$

To get the Fourier Transform we exploit the sifting property of Dirac functions.

$$\hat{h}(\omega) = \int_x h(x) e^{-i\omega x} dx = 1 - e^{-2i\omega} = e^{-i\omega} (e^{i\omega} - e^{-i\omega}) = 2ie^{-i\omega} \sin(\omega)$$

$$\|\hat{h}(\omega)\| = 2\|\sin(\omega)\|$$

Since $\|\hat{h}(0)\| = \|\hat{h}(\pi)\| = 0$ and $\|\hat{h}(\pi/2)\| = 2$, it is a bandpass filter.

Noise is the result of errors in the image acquisition that lead to pixel values that do not reflect the true intensities of the real scene (scanning devices, CCD detector, transmission).

- Signal independent additive noise (sampling noise)

$$g = f + v$$

- Signal dependent multiplicative noise (illumination variations)

$$g = f + vf = (1 + v)f$$

- Measurement noise (salt and pepper)

Local spatial averaging / Mean filtering

- Let $N(x)$ represent neighborhood of a point x and

$$G(x) = \sum_{\eta \in N(x)} C_{\eta} F(x - \eta)$$

- Often $\sum C_{\eta} = 1$, Example: $N = N_8$, $C_{\eta} = \frac{1}{9}$ gives $\begin{bmatrix} \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \end{bmatrix}$

Two main problems with mean filtering:

- A single pixel can significantly affect the mean value of all the pixels in its neighborhood (errors are spread).
- It blurs edges - a problem if we require sharp edges in the output.



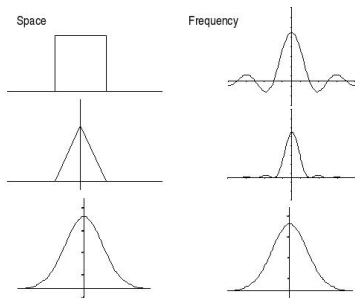
- Common requirements:
 - Coefficients should sum up to 1.
 - Symmetric up/down and left/right.
 - Center pixel has most influence on output.
 - Filter should be separable.
- These result in:

$$C_{\eta} = \begin{pmatrix} \frac{\Delta t}{2} \\ 1 - \Delta t \\ \frac{\Delta t}{2} \end{pmatrix} \begin{pmatrix} \frac{\Delta t}{2} & 1 - \Delta t & \frac{\Delta t}{2} \end{pmatrix} = \begin{pmatrix} \frac{\Delta t^2}{4} & \frac{\Delta t}{2}(1 - \Delta t) & \frac{\Delta t^2}{4} \\ \frac{\Delta t}{2}(1 - \Delta t) & (1 - \Delta t)^2 & \frac{\Delta t}{2}(1 - \Delta t) \\ \frac{\Delta t^2}{4} & \frac{\Delta t}{2}(1 - \Delta t) & \frac{\Delta t^2}{4} \end{pmatrix}$$

- Special case: $\Delta t = \frac{1}{2}$ gives $\begin{pmatrix} \frac{1}{16} & \frac{1}{8} & \frac{1}{16} \\ \frac{1}{8} & \frac{1}{4} & \frac{1}{8} \\ \frac{1}{16} & \frac{1}{8} & \frac{1}{16} \end{pmatrix}$

Low pass filters

- Most information in images is concentrated at low frequencies.
- Noise is uniformly distributed over all frequencies (white noise).
⇒ Suppress high frequency.
- Different filters have different qualities in Fourier space.



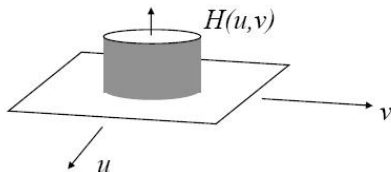
Ideal low pass filter

A transfer function for a 2-D ideal lowpass filter (ILPF) is given as

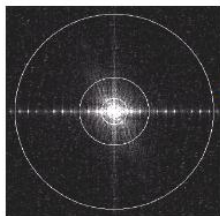
$$H(u,v) = \begin{cases} 1 & \text{if } D(u,v) \leq D_0 \\ 0 & \text{if } D(u,v) > D_0 \end{cases}$$

where D_0 is a stated nonnegative quantity (the cutoff frequency) and $D(u,v)$ is the distance from the point (u,v) to the center of the frequency plane

$$D(u,v) = \sqrt{u^2 + v^2}$$



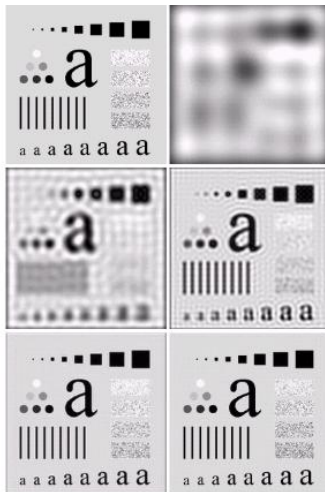
Ideal low pass filter



FT



**Ideal in frequency
domain means
non-ideal in
spatial domain,
vice versa.**



**ringing
and
blurring**

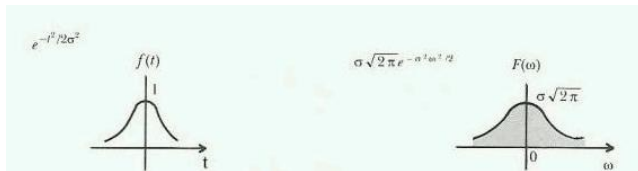
Gaussian low-pass filter

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

$$\hat{g}(u, v; \sigma^2) = e^{-\sigma^2(u^2+v^2)/2}$$

where $(u^2 + v^2)$ = squared distance from the origin.

- The parameter measures spread of Gaussian curve. Smaller the value, the larger the cutoff frequency and milder the filtering.
When $(x^2 + y^2) = \sigma^2$, the filter is at 0.607 of its maximum value.



- Note: Gaussian in spatial domain and Gaussian in frequency.

- The filter $(\frac{\Delta t}{2}, 1 - \Delta t, \frac{\Delta t}{2})$ can for $\Delta t = \frac{1}{2}$ be written

$$(\frac{1}{4}, \frac{1}{2}, \frac{1}{4}) = \frac{1}{4}(1, 2, 1) = \frac{1}{2}(1, 1) * \frac{1}{2}(1, 1)$$

Repeated use of $(1, 1)$ kernels gives rise to Pascal's triangle.

	1																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																					
--	---	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

- Central limit theorem \Rightarrow kernels approach Gaussian kernels.

Image averaging: Average vs. Gaussian



Original

Average

Gaussian

- Nonlinear spatial filters also operate on neighborhoods.
- Operations are based directly on pixel values in neighborhood. They do not explicitly use coefficient values as in filter masks.
- Purpose: Incorporate prior knowledge to avoid destructive behavior, typically at edges and corners.
- Basic methods:
 - median filtering
 - selective averaging
 - weighted averaging

- Anisotropic smoothing: smooth differently in different directions, usually in order to preserve edges.
- Idea: smooth pixels based on similarity $s(p, q)$ between pixels.

$$p' = \frac{\sum_{q \in N(p)} q \cdot s(p, q)}{\sum s(p, q)}$$

- Similarity $s(p, q)$ can be measured in colour, position, etc.
- Examples:

$$s(p, q) = e^{-\left(\frac{p-q}{K}\right)^2}, \quad s(p, q) = \frac{1}{1 + \left(\frac{p-q}{K}\right)^2}$$

- Problems: Different kernels at different positions \Rightarrow Impossible to analyse in frequency space.

Anisotropic smoothing



Note: the image is smoother, but individual hairs are not blurred out.

Median filtering

$$G(m) = \text{median}_{k \in N(m)} F(k)$$

Properties:

- + Preserves the value in 1D monotonic structures (shading).
- + Preserves the position of 1D step edges.
- + Eliminates local extreme values (e.g. salt-and-pepper).
- Creates painting-like images.

	123	125	126	130	140
	122	124	126	127	135
	118	120	150	125	134
	119	115	119	123	133
	111	116	110	120	130

Neighbourhood values:

115, 119, 120, 123, 124,
125, 126, 127, 150

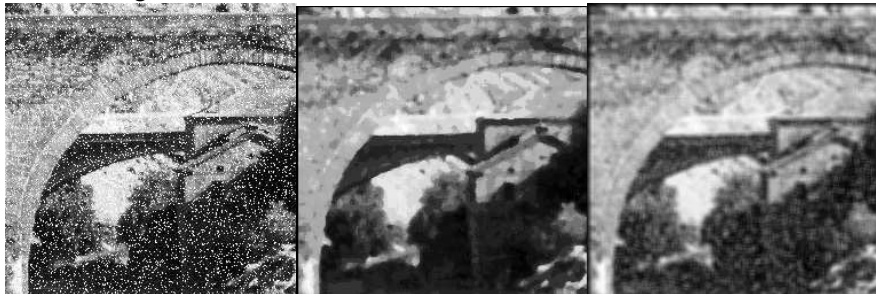
Median value: 124

Median filtering - example

Original

med5x5

mean5x5



- Purpose: Enhance local contrast, highlight fine details.
- Methods:
 - Unsharp masking
 - High-pass filtering (spectral)
 - Differentiation (first and second order derivatives)
- Common desirable property:
 - Isotropy (rotational invariance)
- Common problems:
 - Differentiation and high-pass filtering enhance noise
- Difference compared to grey-level transformations:
 - Spatial variations are taken into account

Easiest way: unsharp masking

- Idea: “subtract out the blur”
- Blur image \rightarrow subtract from original \rightarrow weight \rightarrow add to original

$$g(x,y) = f(x,y) + \alpha(f(x,y) - \bar{f}(x,y))$$

	Original ($\times A$)		Original		Blur																											
$\frac{1}{A}$	<table border="1"><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>A</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	A	0	0	0	0	+	<table border="1"><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>5</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	5	0	0	0	0	-	<table border="1"><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr></table>	0	1	0	1	1	1	0	1	0
0	0	0																														
0	A	0																														
0	0	0																														
0	0	0																														
0	5	0																														
0	0	0																														
0	1	0																														
1	1	1																														
0	1	0																														
	$\left. \right]$																															
		= $\frac{1}{A}$	<table border="1"><tr><td>0</td><td>-1</td><td>0</td></tr><tr><td>-1</td><td>A+4</td><td>-1</td></tr><tr><td>0</td><td>-1</td><td>0</td></tr></table>	0	-1	0	-1	A+4	-1	0	-1	0																				
0	-1	0																														
-1	A+4	-1																														
0	-1	0																														

- Sharpening with a high-pass filter:

$$G(u, v) = F(u, v) + \alpha(H_{hp}(u, v)F(u, v))$$

- Quite similar to unsharp masking, but in Fourier domain.

$$H_{hp}(u, v) = 1 - H_{lp}(u, v)$$

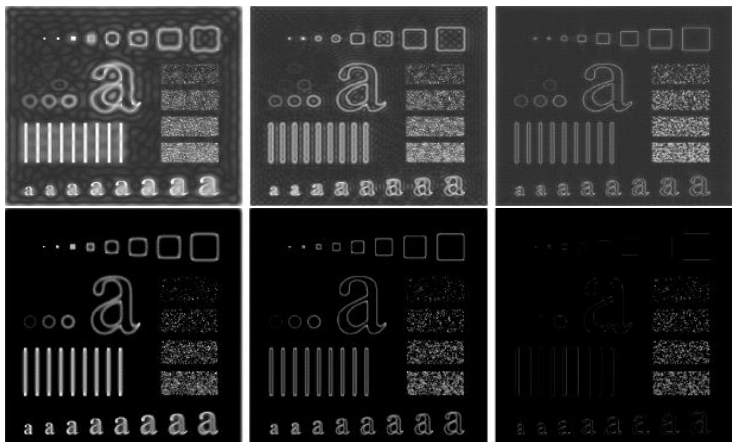
Ideal:

$$H_{hp}(u, v) = \begin{cases} 1 & \text{if } (u^2 + v^2) > D_0^2 \\ 0 & \text{if } (u^2 + v^2) \leq D_0^2 \end{cases}$$

Gaussian:

$$H_{hp}(u, v) = 1 - e^{-\sigma^2(u^2+v^2)/2}$$

High-pass filters



Results with ideal (top) and Gaussian (bottom) filters.

Changes in functions are normally computed with derivatives, but images are discrete and we have to approximate.

- Requirements for a first order derivative operator:
 1. zero in flat areas
 2. non-zero along ramp signals of constant slope
 3. non-zero in the onset and end of a gray-level step or ramp
- Requirements for a second order derivative operator:
 1. zero in flat areas
 2. zero along ramp signals of constant slope
 3. non-zero at the onset and end of a gray-level step or ramp

- Basic definition of a first order x-wise derivative operator:

$$f_x = f(x+1, y) - f(x, y)$$

Similarly, a first order y-wise derivative f_y can be defined.

- More common in practice (derivative at x , not $x+0.5$):

$$f_x = \frac{1}{2}(f(x+1, y) - f(x-1, y))$$

- Second order x-wise derivative operator:

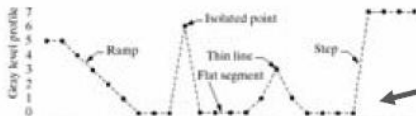
$$f_{xx} = f(x+1, y) + f(x-1, y) - 2f(x, y)$$

Derivatives

"Simple"
original image



1D horizontal gray
level profiles
along center of
image including
isolated noise
point



"Simplified"
profile

f	5	5	4	3	2	1	0	0	0	6	0	...	0	0	7	7	7	7
f_x		-1	-1	-1	-1	-1	0	0	6	-6	0	...	0	7	0	0	0	
f_{xx}		-1	0	0	0	0	1	0	6	-12	6	...	0	7	-7	0	0	

Examples of differentiation operators

- We are interested in filters whose response is independent of the direction of discontinuities in the image.

Isotropic filters are rotationally invariant: rotating the image and then applying the filter is the same as applying the filter first and then rotating the image.

- **Gradient:** $\nabla f = (f_x, f_y)$

First order, linear, non-isotropic

- **Gradient magnitude:** $|\nabla f| = \sqrt{f_x^2 + f_y^2}$

First order, non-linear, isotropic

- **Laplacian:** $\nabla^2 f = f_{xx} + f_{yy}$

2nd order, linear, isotropic

Fourier transform of a derivative

$$\mathcal{F}[f(x)] = \hat{f}(\omega)$$

$$\begin{aligned}\mathcal{F}[f_x(x)] &= \lim_{h \rightarrow 0} \frac{\mathcal{F}[f(x+h)] - \mathcal{F}[f(x-h)]}{2h} = \\ &= \lim_{h \rightarrow 0} \frac{\hat{f}(\omega)e^{i\omega h} - \hat{f}(\omega)e^{-i\omega h}}{2h} = \lim_{h \rightarrow 0} \frac{\hat{f}(\omega) \cdot 2i \sin(\omega h)}{2h} = \\ &= i\omega \hat{f}(\omega) \lim_{h \rightarrow 0} \frac{\sin(\omega h)}{\omega h} = i\omega \hat{f}(\omega)\end{aligned}$$

Laplacian operator

$$\mathfrak{S}\left[\frac{\partial^2 f(x, y)}{\partial x^2} + \frac{\partial^2 f(x, y)}{\partial y^2}\right] = \boxed{-(u^2 + v^2)} F(u, v)$$

\downarrow

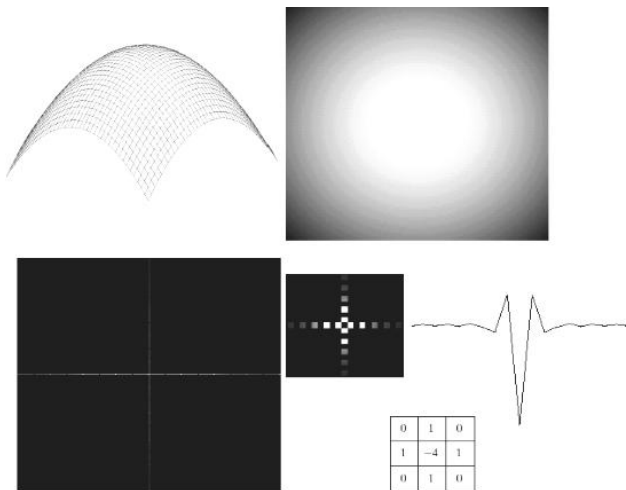
$$H_1(u, v) = -(u^2 + v^2)$$

\downarrow **Frequency domain**




$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \xrightarrow{\text{Spatial domain}} \text{Laplacian operator}$$

Isotropic: depends only on the distance from origin, not on the angle.

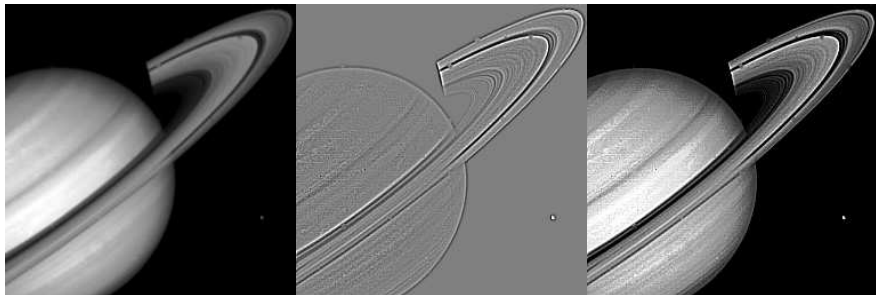
Laplacian operator



Laplacian in frequency (upper) and spatial (lower) domain.

	enhanced image	Original image	Laplacian output
Spatial domain			
	$g(x, y) = f(x, y) - \nabla^2 f(x, y)$		
Frequency domain	$G(u, v) = F(u, v) + (u^2 + v^2)F(u, v)$		
			Laplacian
new operator	$H_2(u, v) = 1 + (u^2 + v^2) = 1 - H_1(u, v)$		

Application of the Laplacian operator



Original image (left), application of Laplacian operator (middle), and subtraction of the Laplacian from the original image (right).

Summary of good questions

- Why would you like to do image enhancement?
- Mention a typical grey-level transformation. When to use it?
- What do histogram stretching and compression mean?
- What are the principles of histogram equalization?
- What are the differences between lowpass, bandpass and highpass filters?
- What kind of noise can you have?
- Why does image averaging work?
- Why are ideal lowpass filter rarely used in practice?
- What characteristics does a Gaussian filter have?
- What is the difference between mean and median filters?
- How can you do sharpening?
- How can you approximate a first order derivative?
- What is a Laplacian?

- Gonzalez & Woods: Chapters 3.2 - 3.6, 4.7 - 4.10
- Szeliski Chapters 3.1 - 3.2 and 3.3.1