Digital Image Processing

30 November 2006

Dr. ir. Aleksandra Pizurica  
Prof. Dr. Ir. Wilfried Philips

Aleksandra.Pizurica @telin.UGent.be  
Tel: 09/264.3415

UNIVERSITEIT GENT  
Telecommunicatie en Informatieverwerking

Image Enhancement
Objectives of Image Enhancement

• Image enhancement means enhancement for visual interpretation

• This covers a wide variety of processing methods (e.g., making details better visible, improving contrast, making images sharper,…)

• The evaluation of the results is highly subjective (hence no general theory of image enhancement)

• The main objective of image enhancement is to process an image such that the result is more suitable for a specific application.

• Image enhancement techniques are highly problem oriented (e.g., a method that enhances well an X-ray image may be unsuitable for a satellite image)

• A possible broad categorization:
  • Spatial domain enhancement techniques
  • Frequency domain enhancement techniques

Spatial Domain Enhancement

Spatial domain –
  - aggregation of image pixels

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

\( f(x,y) \) – input image

\( f(x,y) \) – output image

\( T \) – operator on \( f \) defined over some neighborhood of \((x,y)\)

Simplest case: neighborhood of size 1x1 (a single pixel)
In this case \( T \) is intensity transformation

\[ s = T(r) \]

\( r \) and \( s \) – grey values of \( f \) and \( g \), resp., at any position \((x,y)\)
### Intensity transformation

**Contrast enhancement**

The values of $r$ below $m$ are compressed into a narrow range of $s$ towards black. The opposite effect for values above $m$.

Also: a narrow intensity range around $m$ is transformed into a wider intensity range.
Basic gray-level transformations

Values of the transformation function are typically stored in a one-dimensional array.

The mapping from $r$ to $s$ is implemented as a look-up-table.

Which transformation produces this enhancement?

Image negative

Example: enhancement of digital mammograms

$r$ and $s$ – grey values of input and output image, resp., at any position

Input (and output) gray levels are in the range $[0, L-1]$
Log transformations

\[ s = T(r) \]

A narrow range of low gray-level values in the input image is mapped to a wider range of output values. The opposite is true for higher gray values.

Expands the values of dark pixels.
Compresses the values of bright pixels.

Compresses the overall dynamic range.
Allows visualisation of images with a large range of pixel values!

\[ s = c \log(1+r) \]

\( r \) and \( s \) – grey values of input and output image, resp., at any position
Assuming \( r \geq 0 \) (gray levels are in the range \([0, L-1]\))

Visualisation by Log transform

Log transform is commonly used for visualisation of data that have a big dynamic range, such as

- Fourier spectra
- Ultrasound data
- Optical Coherence Tomography (OCT) data
- Radar images
- …
Power low transformations

\[ s = c r^\gamma \]

- \( r \) and \( s \) – grey values of input and output image, resp., at any position
- \( c, \gamma \) - positive constants

\( \gamma < 1 \) \( \Rightarrow \) enhancement of details in dark regions

\( \gamma > 1 \) \( \Rightarrow \) enhancement of details in bright regions

...Power low transformations...
...Power low transformations

Histogram processing

\[ h(r_k) = n_k \]

- \( r_k \) : \( k \)-th grey level in the image
- \( n_k \) : number of pixels having gray level \( r_k \)

**Normalized histogram**

\[ p(r_k) = n_k / n \]

- \( p(r_k) \) is an estimate of probability of occurrence of gray level \( r_k \)
- \( n \): number of image pixels

- Histograms are simple to calculate in software and hardware
- Provide image statistics
- Useful in image compression, denoising, analysis
Conclusion: to flatten (equalize) the histogram, transform the image with its cumulative histogram!

This mapping is cumulative histogram

\[ s_k = T(r_k) = \sum_{j=0}^{k} p(r_j) = \sum_{j=0}^{k} \frac{n_j}{n} \]

Local processing

It is often useful to apply intensity transformations locally (within local windows)

In some cases such processing creates artifacts! (e.g., can create non-existing structures in an almost flat background)
**Basics of spatial filtering**

\[ g(x, y) = m(-1,-1)f(x-1,y-1) + \ldots + m(1,1)f(x+1,y+1) \]

\[ g(x, y) = \sum_{(s,t)\in\Omega} m(s,t)f(x+s,y+t) \]

\[ \Omega = \{(s,t) : s,t = -1, 0, 1\} \]

*support of the mask*

**Smoothing spatial filters**

Input image | Output image
---|---

\[ b(x, y) = \sum_{(s,t)\in\Omega} g(s + x, t + y)m(s,t) \]

*Correlation!*

\[ \Omega = \{(x, y) : x, y = -1, 0, 1\} \]

\[ \Omega \text{ includes usually (0,0) but not always!} \]
Example: mean filter

This filter partly suppresses noise but blurs the edges.
Note: this filter is separable.

\[
M = \frac{1}{25} \begin{bmatrix}
1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 \\
\end{bmatrix}
\]

Order statistics filters

Order statistics filters are nonlinear spatial filters.

Operation: (1) perform ordering (ranking) the pixels within a filter mask.
(2) replace the value of the center pixel by the ranking result

Best known example in this class: median filter
Order statistics filters: Median filter

Basic idea: remove “outliers”
The median is a more robust statistical measure than mean

Properties of the median filter...

+removes isolated pulses
+preserves uniform areas
-removes thin vertical, horizontal, diagonal lines
+preserves sharp horizontal and vertical edges
+preserves “soft” horizontal and vertical edges
... Properties of the median filter...

Some of these problems can be avoided by choosing not-rectangular windows.

Removal of impuls noise

Median filter removes isolated noise peaks, without blurring the image.
... Reduction of impulse noise ...

Noise-free original

median over 3x3

Median filter removes isolated noise peaks,
without blurring the image

... Reduction of impulse noise

mean over 3x3

median over 3x3

Linear filters do not take into account impulsive (isolated) character of the noise and blur the image

The median filter performs in this case much better than the linear filters
Median filter and reduction of white noise

For not-isolated noise peaks (e.g., white Gaussian noise) median filter is not very efficient.

Repeated application of the median filter

Iterating median filter can remove noise better, but flat blobs appear in the image
Sharpening spatial filters

- Principal objective: highlight fine detail or enhance image detail
- Often based on the first-order and second-order derivatives
- The derivatives of digital functions are defined in terms of differences

\[
\frac{\partial f}{\partial x} \approx f(x+1) - f(x) \\
\frac{\partial^2 f}{\partial^2 x} \approx f(x+1) + f(x-1) - 2f(x)
\]

- In most applications second-order derivative is better suited for image enhancement
- First-order derivatives are used for edge extraction but also have important use in image enhancement

Second order derivatives: Laplacian...

\[
\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \\
\approx f(x-1,y) - 4f(x,y) + f(x+1,y) + f(x,y-1) + f(x,y+1)
\]

digital implementation

lineair filter with filter mask

\[
\begin{bmatrix}
0 & 1 & 0 \\
1 & -4 & 1 \\
0 & 1 & 0
\end{bmatrix}
\]

Enhancement:

\[
g(x, y) = f(x, y) - \nabla^2 f(x, y)
\]
**Second order derivatives: Laplacian**

- **Original image (a)**
- **Laplacian filtered (b)**
- **Scaled for visualisation (c)**
- **Difference (a)-(b)**

Enhancement:
\[ g(x, y) = f(x, y) - \nabla^2 f(x, y) \]

---

**Unsharp masking**

Subtract a blurred version of an image from itself:
\[ f_s(x, y) = f(x, y) - \tilde{f}(x, y) \]

A generalization: **high-boost filtering**
\[ f_{hb}(x, y) = Af(x, y) - \tilde{f}(x, y) \]

This can be re-written as
\[ f_{hb}(x, y) = (A - 1) f(x, y) + f(x, y) - \tilde{f}(x, y) = (A - 1) f(x, y) + f_s(x, y) \]

If we choose that the sharp image as: \( f_s(x, y) = -\nabla^2 f(x, y) \)
\[ f_{hb}(x, y) = Af(x, y) - \nabla^2 f(x, y) \]
“Unsharp masking” – a usual formulation

Unsharp masking: approximate inverse filter for Gaussian PSF with parameter $\sigma$:

$$g(x, y) = f(x, y) - \frac{\sigma^2}{2} \nabla^2 f(x, y)$$

Filter mask:

$$\begin{bmatrix}
0 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 0 \\
\end{bmatrix} - z_0 \begin{bmatrix}
0 & 1 & 0 \\
1 & -4 & 1 \\
0 & 1 & 0 \\
\end{bmatrix}$$

with $z_0 = \frac{\sigma^2}{2}$

This filter amplifies noise. The noise amplification increases when $z_0$ increases.

Note: if the noise amplification is a problem, choose $z_0 < \frac{\sigma^2}{2}$

$\Rightarrow$ blurring increases but not necessarily excessively

$\Rightarrow$ less noise amplification

Unsharp masking: example

Test image 1

Gaussian filtered ($z_0=0.5$)

$$\begin{bmatrix}
0 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 0 \\
\end{bmatrix} - z_0 \begin{bmatrix}
0 & 1 & 0 \\
1 & -4 & 1 \\
0 & 1 & 0 \\
\end{bmatrix}$$

with $z_0=0.5$

If $z_0$ is bigger (e.g. $z_0=1$) the resulting image is extra sharp

Test image 2

Gaussian filtered ($z_0=0.5$) + ruis
Unsharp masking and noise

Test image 1  
Test image 2

originals

filtered; \( z_0 = 1 \)

sharper  \( \rightarrow \) noisier

Homomorphic filtering
Homomorphic filtering

Goal: local contrast enhancement: compensation of spatially varying illumination

Illumination model: \( b'(x,y) = b(x,y) \cdot I(x,y) \) met \( b(x,y) \) the ideal image and \( I(x,y) \) the not-uniform lighting

\( \Rightarrow \log(b'(x,y)) = \log(b(x,y)) + \log(I(x,y)) \)

Assumption: \( \log(I(x,y)) \) varies slowly w.r.t \( \log(b(x,y)) \)

\( \Rightarrow \) remove \( \log(I(x,y)) \) by high pass filtering \( \log(b'(x,y)) \)

Example: homomorphic filter

Remark:
- Homomorphic filter is here applied to Y-component; U and V were not changed
- Truncation of RGB-values \(<0\) en \(>255\)
Example: homomorphic filter

Homomorphic filter followed by an extra intensity transform to improve global contrast

Filtering in the Fourier domain

2D-DFT of an image

\( B_{k,j} \)

2D-DFT of the mask

\( \tilde{H}_{k,j} \)

\( \times \)

\( \circ \)

2D-DFT of the filtered image

inverse DFT

filtered image
Vergelijking van het aantal berekeningen

Filteren van een $N \times N$ beeld met een $n \times n$ masker:
- Via convolutie/correlatie:
  - scheidbaar filter: $4n^2-2$ bewerkingen/pixel
  - niet-scheidbaar filter: $2n^2-1$ bewerkingen/pixel

- Via de DFT (als we de DFT van het masker al kennen):
  
  $$\frac{4N^2 \log_2 N + N^2}{N^2} = 4 \log_2 N + 1$$ bewerkingen/pixel

Vermenigvuldiging in frequentiedomein

Voorwaartse en inverse DFT (via FFT)

De DFT-implementatie is voordeliger voor grote maskers
- Break-even punt (niet-scheidbaar): $n \approx \sqrt{2 \log_2 N}$ (b.v. $N=256 \Rightarrow n=4$)

Voorbeelden: laagdoorlaatfilters

Uitmiddeling over 3x3 omgeving

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

zorgt ervoor dat $\tilde{H}(0,0) = 1$ ⇒ behoudt gemiddelde grijswaarde

Binomiaal-filter (3x3)

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

geen zijlobes! gewenst als men hoge frequenties wil onderdrukken
Filter in het fourierdomein: voorbeeld

Filter: \( H_{k,l} = 1 \) binnen het blauw vierkant en 0 erbuiten

Dit komt erop neer de fouriercoëfficiënten bij hoge spatiale frequenties nul te maken ⇒ laagdoorlaatfilter

Invloed op het beeld: • "Ringing" (rimpels)
• "Blurring" (wazig maken)