Computer Assisted Image Analysis
Lecture 3 – Local Operators

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

Chapters for this lecture

- Chapters 2.5.1 – 2.5.2, 3.4 – 3.7 (and 5.3) in Gonzales-Woods.
Subjects

- Gray level transforms (contrast, brightness).
- Image histograms (intensity distribution).
- Histogram equalization (normalized intensity distribution).
- Image arithmetic (addition, and subtraction)
Local neighborhood

Relationships between pixels

- $N_4$, 4-neighbors (also called edge neighbors).

- $N_D$, $D$-neighbors (also called diagonal, or point neighbors).

- Together, 4- and $D$-neighbors are called $N_8$, or 8-neighbors.
Local neighborhood
Adjacency and connectivity

In a binary image, two pixels $p$ and $q$ are

- 4-adjacent if they have the same value and $q$ is in the set $N_4(p)$.
- 8-adjacent if they have the same value and $q$ is in the set $N_8(p)$.
- $m$-adjacent (mixed adjacency) if they have the same value and
  - $q$ is in the set $N_4(p)$ OR
  - $q$ is in the set $N_D(p)$ AND the set $N_4(p) \cap N_4(q)$ has no pixels with the same value as $p$ and $q$.

Two pixels (or objects) are 8-, 4-, or $m$-connected if a 8-, 4-, or $m$-path can be drawn between them.
Local neighborhood

Why is this important?

How many objects are there in this image?
Spatial filtering

- The pixel value in the output image calculated from a local neighborhood of the pixel in the input image.
- The local neighborhood is described by a window, mask, kernel, template, or spatial filter (typical sizes: $3 \times 3$, $5 \times 5$, $7 \times 7$ pixels).
Spatial filtering

Examples of linear filters:

- Smoothing filters
  - Mean filters
  - Gauss filters
- Sharpening filters
  - Unsharp mask
- Edge enhancing filters
  - Sobel operator
  - Laplace operator

Examples of non-linear filters:

- Median, min, max and percentile filters
- The bilateral filter
Spatial filtering

Linear filters

- Example: A linear $3 \times 3$ filter
- Let $w_i$ be the kernel coefficients and $z_i$ the underlying pixel values at the current position
- The linear filter operator:
  $$R = \sum_{i=1...9} w_i z_i = w^T z = \langle w, z \rangle$$
- ... is a scalar product!
- The global operation is convolution of the image and the kernel, $f \ast w$
Spatial filtering

Nonlinear filters

- Essentially every other kind of function \( q(w) \).
- Example: A non-linear 3 \( \times \) 3 filter
- Let \( z_i \) be the underlying pixel values at the current position
- \( r = q(z_1, z_2, \ldots, z_9) \), where \( q \) is non-linear, e.g. \( q = \max\{z_i\} \).
- The global operation is not convolution!
1-D convolution

\[ g(x) = w(x) \ast f(x) = \sum_{s=-a}^{a} w(s)f(x - s) \]

where \( a = (L - 1)/2 \) and \( L \) is the length of \( w \). Example:

\[
\text{conv2([0 1 0 0 0 2 0 0],[1 2 3])}
\]
\[
\text{ans} = [1 2 3 0 2 4 6 0]
\]
2-D Convolution

2-D convolution

\[ g(x, y) = w(x, y) \ast f(x, y) = \sum_{s=-a}^{a} \sum_{t=-a}^{a} w(s, t) f(x - s, y - t) \]

where \( a = (L - 1)/2 \) and \( L \) is the length/width of \( w \). Example:

\[
\begin{align*}
0 & & 0 & & 0 & & 0 \\
0 & & 0 & & 0 & & 0 \\
0 & & 0 & & 1 & & 0 \\
0 & & 0 & & 0 & & 0 \\
0 & & 0 & & 0 & & 0 \\
\end{align*}
\]

>> b

b =

\[
\begin{align*}
1 & & 2 & & 3 \\
4 & & 5 & & 6 \\
7 & & 8 & & 9 \\
\end{align*}
\]

>> c = conv2(a,b,'same')

c =

\[
\begin{align*}
0 & & 0 & & 0 & & 0 \\
0 & & 1 & & 2 & & 3 \\
0 & & 4 & & 5 & & 6 \\
0 & & 7 & & 8 & & 9 \\
0 & & 0 & & 0 & & 0 \\
\end{align*}
\]
Convolution and correlation

- Correlation is convolution with a $180^\circ$ rotated kernel $w$ (2-D) or mirrored $w$ (1-D).
- $f \ast g = g \ast f$
- $f \ast (g \ast h) = (f \ast g) \ast h$
- $f \ast (g + h) = f \ast g + f \ast h$
- $a(f \ast g) = (af) \ast g$
Lowpass filtering, rectangular averaging
Lowpass filtering, disk averaging
Lowpass filtering, Gaussian filtering
Lowpass filtering

Carl Friedrich Gauss (1777 – 1855)

German mathematician and physical scientist. The 1-D Gauss function (normal distribution):

\[
\phi(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2} x^2}.
\]
Lowpass filtering

Notes on linear lowpass filters

- Gaussian filtering: Convolution with \( G(x, y) = \frac{1}{2\pi} e^{-\frac{1}{2} \left( \frac{x^2 + y^2}{\sigma^2} \right)} \)
- Beware of using a too small kernel, in relation to \( \sigma \), in Gaussian filtering
- Rule of thumb: For \( L \times L \) kernel, \( \sigma \approx L/5 \) is ok
- Small details removed, constant areas unchanged
- Other filters: e.g. Butterworth, Chebychev and Bessel
Edge detection, partial derivatives

Calculus:
Changes are described by partial derivatives in images.

An edge is described by its gradient *magnitude* and *direction*.

In the discrete case, we approximate the derivatives by differences. We use spatial filters, or weighted masks, looking at local neighborhoods and traverse the image by convolution.

\[
\frac{\partial f(x, y)}{\partial x} \approx f(x + 1, y) - f(x, y) = f(x, y) \ast [1, -1]
\]

\[
\frac{\partial f(x, y)}{\partial x} \approx \frac{(f(x + 1, y) - f(x - 1, y))}{2} = f(x, y) \ast [1, 0, -1]/2
\]
Edge detection, the Sobel operator

Calculating magnitude and direction

One mask is created for each possible direction. In the $3 \times 3$ case, we have eight directions, resulting in eight masks with weights. The response of each filter represents the strength, or magnitude, of the edge in that direction.

Example:

- $m_1$ gives the strength in $x$-direction.
- $m_2$ gives the strength in $y$-direction.
- edge magnitude $= (m_1^2 + m_2^2)^{1/2}$.
- direction magnitude $\tan^{-1}(m_1/m_2)$.

Alternatively, the mask giving the maximum response approximates both magnitude and direction of the gradient.
Edge detection, the Sobel operator
Spatial filtering

Irwin Sobel
Edge detection, the Sobel operator

Notes on the Sobel operators

- The Sobel operators are smoothed derivatives in the $x$- and $y$ direction, $f_x$ and $f_y$
- To obtain an edge image, these are combined: $R = \sqrt{f_x^2 + f_y^2}$
- The combined Sobel operator is a non-linear filter
Highpass filtering

Laplace operator

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

\[ \nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \]

\[ \nabla^2 f = [f(x + 1, y) + f(x - 1, y) + f(x, y + 1) + f(x, y - 1)] - 4f(x, y) \]
Highpass filtering

Pierre-Simon Laplace (1749 – 1827)
Matematician and astronomer. Laplace Equation:

$$\frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} + \frac{\partial^2 f}{\partial z^2} = 0.$$
Highpass filtering

Edge enhancing filters

- The Laplace operator: Detection of edges independent of direction.

Original image
Laplace 4n
Laplace 8n
Laplace 5x5

Compare with sum of 4 Sobel filters with different directions
Highpass filtering

Rotation independent edge detection

- Gives 0 as output in homogenous regions and output $\neq 0$ at discontinuities.
- The size of the filter decides the types of edges (discontinuities) that are found.
- Independent of edge direction and very useful when searching for curved edges (faster than $4 \times$ Sobel).
Highpass filtering, Laplacian of Gaussian
Highpass filtering

Notes on linear highpass filters

• Laplacian of Gaussian (LoG): $\nabla^2 g \ast f$, where $f$ is the image, $g$ is a Gaussian kernel and $\nabla^2$ is the Laplacian operator
• Finds features like lines and edges
• In LoG, $\sigma$ selects a scale
• Other filters: e.g. 1-Butterworth, 1-Chebychev and 1-Bessel
Enhancement, unsharp masks

Image sharpening, or “unsharp mask”. Enhancing edges and adding them to the original image.

Original       Lowpass       Laplace       Combined
Enhancement, unsharp masks
Enhancement, unsharp masks

Notes on the unsharp masks

• A weighted sum of the image and a highpass,
  \[ g(x, y) = f(x, y) + \alpha \nabla^2 f(x, y) \]

• A weighted difference of the image and a lowpass,
  \[ g(x, y) = (1 + \alpha)f(x, y) - \alpha G_\sigma \ast f(x, y) \]

• The term “unsharp mask” comes from astronomy

• Typical operation in programs like Photoshop and GIMP
Linear filters and border effects

• Close to the image border, a the kernel neighborhood contains unknown values
• Commonly these are assumed to be 0
• The result is a dark border around the filtered image
• A trick to remove this is: Filter an image filled with ones (1:s) and divide the result with this.
Other ways to minimize border effects include:

- Choose another value than 0 for pixels outside the image, 
  \[ f(\text{end} + i) = 43 \]
- Copy the closest known pixel value at the image border, 
  \[ f(\text{end} + i) = f(\text{end}) \]
- Repeat the image periodically, e.g. 
  \[ f(\text{end} + i) = f(i) \]
- Repeat the image by mirroring, e.g. 
  \[ f(\text{end} + i) = f(\text{end} - i) \]
Spatial filtering
Non-linear, or order statistics filter

Median and percentile filters

- Look at the local histogram in the neighborhood
- Do something smarter than just the average...
- Below, the middle pixel has the value 63
- The median of the neighborhood is 95 and the median is 80
- In general, the median is a more robust estimator
Spatial filtering
Non-linear, or order statistics filter

Median and percentile filters

- Median: Sort the values in the neighborhood, pick the middle one
- Percentile: Pick another than the middle

35 63 67 67 74 76 78 79 79 80 80 80 83 86 88 98 104 105 126
137 141 150 152 159
Spatial filtering
Optimality of the mean and median

Mean
For a set of pixels $z_i$ in an image neighborhood, the mean value $m$ minimizes the sum of squared differences.

$$
\varphi(m) = \sum_i ||z_i - m||^2
$$

Median
For a set of pixels $z_i$ in an image neighborhood, the median value $c$ minimizes the sum of absolute differences. The 2-norm, $|| \cdot ||$, allows this functional to be used to compute the median value when $z_i$ is a vector-valued set (e.g. RGB images).

$$
\theta(c) = \sum_i ||z_i - c||
$$
Local Histogram Equalization

For each pixel, perform histogram equalization in a neighborhood and store the result in middle pixel.

1. Original image.
2. Result of global histogram equalization.
3. Result of local histogram equalization using a $7 \times 7$ neighborhood about each pixel.
Other Filters

- The Bilateral Filter (Carlo Tomasi & Roberto Manduchi)
- Steerable Filters (Freeman, Knutsson)
- Non-local means (Buades, Coll & Morel)
Other Filters
The Bilateral Filter

- The Bilateral Filter (Carlo Tomasi & Roberto Manduchi)
- Similar to “selective Gaussian blur” in GIMP.
- When smoothing, only average with pixels that have colors similar to the middle pixel in the current neighborhood
Other Filters
The Bilateral Filter

- Add additional spatial weight depending on intensity difference
- \( w_i(k,m) = Gauss_1(f(L/2, L/2) - f(k,m), \sigma_i) \)
- \( w_s(k,m) = Gauss_2(L/2 - k, L/2 - m, \sigma_s) \)
- \( r = \frac{1}{C} \sum_{k,m=1}^{L} w_i(k,m)w_s(k,m)f(k,m) \)
- \( C \) is a normalizing factor
- \( Gauss_1 \) is a 1-D function, \( Gauss_2 \) is a 2-D function
- It's like convolution, except the extra \( w_i \) factor
- Drawback, cannot deal with shot noise
Other Filters

Homomorphic Filtering

- Transform image $f(x, y)$ using the log function
- Apply a linear filter $H$ (low- or highpass)
- Transformed back using the exp function
- Highpass = remove uneven illumination, lowpass = remove multiplicative noise

\[ f(x, y) \xrightarrow{\text{ln}} H \xrightarrow{*h(x,y)} \xrightarrow{\text{exp}} g(x, y) \]
Broken contours can be re-connected by smoothing.
Spatial filtering

Smoothing as post-processing

Pixelized images look better when annoying high frequencies are removed
Competition (non-mandatory)

- Download competition.zip
- Design method to restore noisy image
- Compare with ground truth using MSE\(^1\) excluding 4 px border
- Email team name and file restore.m\(^2\) to anders@cb.uu.se
- Open until Sun 2013-03-03, highscore list, 100 SEK price

\(^1\)MSE = "mean squared error"
\(^2\)called as newim = restore(im) or restore
Review Questions

- Problems 2.11, 3.18, 3.19, 3.25 and 3.28 in Gonzales-Woods.
Until Next Lecture

• Read, read, read
• The image processing challenge!
• Do the review questions