Image Analysis (FMAN20) Lecture 4, 2017

KALLE ÅSTRÖM
Image Analysis - Motivation
Mr Fifield, analysera konstruktionen.
Hela det inre.
Today: Image Filters

Smooth/Sharpen Images...  Find edges...  Find waldo...
Overview – Convolutions

1. Convolution
   1. Uses information in several pixels
   2. Can be seen as scalar product for all submatrices
   3. Can be seen as linear classifier for submatrices
   4. Detects things that looks like itself

2. Convolution theorem
   3. Connecting linear algebra, Fourier transform and convolutions
Convolution Operator

\[ g = f \ast h \]

\[ g(i, j) = \sum_u \sum_v f(i - u, j - v)h(u, v) \]
Cross-Correlation
Sliding scalar product

g(i, j) = \sum \sum f(i + y, j + x)\tilde{h}(y, x)

Compare with convolution

\[ g(i, j) = \sum \sum f(i - u, j - v)h(u, v) \]
\[ \tilde{h}(u, v) = h(-u, -v) \]
Why use convolution?
Cross-correlation seems much simpler.

Convolutions have several easy to use properties

\[
\begin{align*}
f \ast h &= h \ast f, \\
f \ast (g \ast h) &= (f \ast g) \ast h, \\
f \ast (g + h) &= f \ast g + f \ast h, \\
a(f \ast g) &= (af) \ast g, \\
\delta \ast f &= f, \\
\partial(f \ast g) &= (\partial f) \ast g,
\end{align*}
\]
Convolutions and linear systems

Convolution as image transform

$S$ is called a system or filter.

$I$ $S$ is linear if $S(f_1 + f_2) = S(f_1) + S(f_2)$.

Implies that $g(x) = \int h(x, y) f(y) \, dy$,

where $h$ called the impulse response.

$I$ $S$ is translation invariant if $S(f(x)) = g(x)$ and $S(f(x+a)) = g(x+a)$.

Any linear and translation invariant system can be represented as a convolution.
Convolutions and linear systems, point spread function

\[ g = f \ast h \]

\[ h = \delta \ast h \]
Motivation: noise reduction

- We can measure **noise** in multiple images of the same static scene.
- How could we reduce the noise, i.e., give an estimate of the true intensities?
Motivation: noise reduction

- How could we reduce the noise, i.e., give an estimate of the true intensities?
- What if there’s only one image?
First attempt at a solution

- Let's replace each pixel with an average of all the values in its neighborhood
- Assumptions:
  - Expect pixels to be like their neighbors
  - Expect noise processes to be independent from pixel to pixel
First attempt at a solution

- Let’s replace each pixel with an average of all the values in its neighborhood
- Moving average in 1D:

Source: S. Marschner
Weighted Moving Average

- Can add weights to our moving average
- *Weights* $[1, 1, 1, 1, 1] \div 5$

Source: S. Marschner
Weighted Moving Average

- Non-uniform weights \([1, 4, 6, 4, 1]/16\)

Source: S. Marschner
Moving Average In 2D

\[ F[x, y] \quad G[x, y] \]
Moving Average In 2D

\[ F[x, y] \quad \text{and} \quad G[x, y] \]

Source: S. Seitz
Moving Average In 2D

\[ F[x, y] \]  

\[ G[x, y] \]

Source: S. Seitz
Moving Average In 2D

\[ F[x, y] \]

\[ G[x, y] \]
Moving Average In 2D

\[ F[x, y] \]

\[ G[x, y] \]

Source: S. Seitz
Moving Average In 2D

\[ F[x, y] \]

\[ G[x, y] \]
Convolution

- Convolution:
  - Flip the filter in both dimensions (bottom to top, right to left)
  - Then apply cross-correlation
Median filter

- No new pixel values introduced
- Removes spikes: good for impulse, salt & pepper noise
Median filter

Salt and pepper noise

Plots of a row of the image

Source: M.
Smoothing with a Gaussian

Parameter $\sigma$ is the “scale” / “width” / “spread” of the Gaussian kernel, and controls the amount of smoothing.

for $\text{sigma}=1:3:10$
    $h = \text{fspecial('gaussian', fsize, sigma)}$;
    $\text{out} = \text{imfilter(im, h)}$;
    $\text{imsho}\text{w(out)}$;
    $\text{pause}$;
end
Partial derivatives of an image

\[ \frac{\partial f(x, y)}{\partial x} \]

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

\[ \frac{\partial f(x, y)}{\partial y} \]

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

Which shows changes with respect to \( x \)?

(showing flipped filters)
Effects of noise

Consider a single row or column of the image

- Plotting intensity as a function of position gives a signal

\[ f(x) \]

\[ \frac{d}{dx} f(x) \]

Where is the edge?
Solution: smooth first

Where is the edge? Look for peaks in $\frac{\partial}{\partial x}(h \ast f)$
Derivative property of convolution

\[
\frac{\partial}{\partial x} (h \ast f) = \left(\frac{\partial}{\partial x} h\right) \ast f
\]

Differentiation property of convolution.
Derivative of Gaussian filters

Image showing 3D plots for x-direction and y-direction.
Effect of $\sigma$ on derivatives

The apparent structures differ depending on Gaussian’s scale parameter.

Larger values: larger scale edges detected
Smaller values: finer features detected

$\sigma = 1$ pixel $\quad \sigma = 3$ pixels
Template matching

- Filters as **templates**:  
  Note that filters look like the effects they are intended to find --- “matched filters”

- Use normalized cross-correlation score to find a given pattern (template) in the image.
  - Szeliski Eq. 8.11

- Normalization needed to control for relative brightnesses.
Template matching

Scene

Template (mask)

A toy example
Template matching

Detected template

Template
Template matching

Detected template

Correlation map
Where’s Waldo?

Scene

Template
Where’s Waldo?

Scene

Template
Where’s Waldo?

Detected template

Correlation map
Template matching

Scene

What if the template is not identical to some subimage in the scene?
So, what scale to choose?

It depends what we’re looking for.

Too fine of a scale…can’t see the forest for the trees.
Too coarse of a scale…can’t tell the maple grain from the cherry
Predict the filtered outputs

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

= ?

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

= ?

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>2</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

= ?

\[ \frac{1}{9} \]
Practice with linear filters

Original

\[
\begin{array}{ccc}
0 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 0 \\
\end{array}
\]

Source: D. Lowe
Practice with linear filters

Original

Filtered (no change)

Source: D. Lowe
Practice with linear filters

Original

\[
\begin{array}{ccc}
0 & 0 & 0 \\
0 & 0 & 1 \\
0 & 0 & 0 \\
\end{array}
\]

Source: D. Lowe
Practice with linear filters

Original

Shifted left by 1 pixel with correlation

Source: D. Lowe
Practice with linear filters

Original

Source: D. Lowe
Practice with linear filters

Original

Blur (with a box filter)

Source: D. Lowe
Practice with linear filters

Original

\[ \begin{array}{ccc}
0 & 0 & 0 \\
0 & 2 & 0 \\
0 & 0 & 0 \\
\end{array} \begin{array}{c}
- \\
\frac{1}{9} \\
\end{array} \begin{array}{ccc}
1 & 1 & 1 \\
1 & 1 & 1 \\
1 & 1 & 1 \\
\end{array} \]
Practice with linear filters

Sharpening filter
- Accentuates differences with local average

Source: D. Lowe
Filtering examples: sharpening

before

after
Convolutions and the Fourier Transform

• What to do near the edge of the image?
• Understanding the Fourier Transform
• The convolution Theorem
• Understanding Convolutions using the Fourier Transform
What to do near the edge of the image?

It is important to know where position zero in sequences are. Compare with decimal-comma in reals.

\[ 12 \cdot 12 = 144, \ 1.2 \cdot 12 = 14.4 \]

What notations should we use?

Suggestion: underline the element at position zero, e.g.

\[ < 1, 1 > \ast < 1, 2, 1 > = < 1, 3, 3, 1 > \]

but

\[ < 1, 1 > \ast < 1, 2, 1 > = < 1, 3, 3, 1 > \]

and

\[ < 1, 1 > \ast < 1, 2, 1 > = < 1, 3, 3, 1 > \]
What to do near the edge of the image?

In practice we do not have infinite images. How should we treat the edges of the image? What values should one assume 'outside' the image. Some common choices are

1. Only calculate the result where we can be certain. The result is a smaller image.
2. Assume that there are zeros outside the image. This often means that we introduce artificial sharp edges at the border.
3. Make a periodic expansion of the image, i.e. assume that the image is periodic. This fits well with the theory for discrete Fourier transform.
What to do near the edge of the image?

Assume that one would like to convolute the image

\[
f = \begin{bmatrix}
1 & 2 & 3 & 5 \\
1 & 3 & 2 & 1 \\
2 & 2 & 2 & 2
\end{bmatrix}
\]

with the smoothing filter

\[
h = \begin{bmatrix}
1 & 1 \\
1 & 1
\end{bmatrix}
\]
What to do near the edge of the image?

(1) Don’t let $h$ extend outside $f$

$$\begin{bmatrix}
7 & 10 & 11 \\
8 & 9 & 7 \\
\end{bmatrix}$$

(2) Extend with zeros $\Rightarrow$ equal or larger resulting $h \ast f$-image

$$\begin{bmatrix}
1 & 3 & 5 & 8 & 5 \\
2 & 7 & 10 & 11 & 6 \\
3 & 8 & 9 & 7 & 3 \\
2 & 4 & 4 & 4 & 2 \\
\end{bmatrix}$$
What to do near the edge of the image?

(3) Extend \( f \) and \( h \) to periodic functions with the same period: 
\( f_p, h_p \Rightarrow \) periodic \( h_p * f_p \) result with same period

\[
\begin{bmatrix}
10 & 7 & 9 & 12 \\
8 & 7 & 10 & 11 \\
6 & 8 & 9 & 7
\end{bmatrix}
\]

Here we have also made a periodic function of \( h \):

\[
h = \begin{bmatrix}
1 & 1 & 0 & 0 \\
1 & 1 & 0 & 0 \\
0 & 0 & 0 & 0
\end{bmatrix}.
\]
Understanding the Fourier Transform

- After fftshift low frequencies are located in the centre of the FFT image.
- High frequencies are located at the borders of the FFT image.
- The fact that images are considered as periodic functions, the borders of the images can influence the FFT.
- High frequencies in one direction (e.g. edges) can be seen as high intensities in the FFT image in the same direction.
Understanding the Fourier Transform

Observe the boundary effects!
Understanding the Fourier Transform

Small boundary effects due to more uniform background.
Understanding the Fourier Transform

Sharp edges in the image gives high frequencies in the FFT image.
Understanding the Fourier Transform

Observe the directions of the edges and the high frequency components.
Understanding the Fourier Transform

Several edges can give rise to several high frequency components.
Understanding the Fourier Transform

Image b and log(abs(fftshift(fft2(b)))

Identify the different edge directions and the corresponding high frequency components.

Kalle Åström
Image Analysis - Lecture 3
Understanding the Fourier Transform

The pattern in the fan gives rise to several high frequency components.

Kalle Åström
Image Analysis - Lecture 3
The Convolution Theorem (First in 1D)

1D vector with N components

Think of f as periodic

The period = N \Rightarrow f(x + N) = f(x) for every x
Understanding the map \( f \rightarrow g \)

\[
g(x) = f \ast h(x) = \sum_{m=1}^{N} h(x - m + 1)f(m)
\]

\( f \mapsto g \) is a linear mapping \( \Rightarrow g = C_h f \).

What does \( C_h \) look like?

\[
\begin{bmatrix}
g(1) \\
g(2) \\
\vdots \\
g(N)
\end{bmatrix}
= 
\begin{bmatrix}
h(1) & h(N) & \cdots & h(3) & h(2) \\
h(2) & h(1) & \ddots & \vdots & \vdots \\
\vdots & \ddots & \ddots & \ddots & \vdots \\
h(N) & h(N-1) & \cdots & h(2) & h(1)
\end{bmatrix}
\begin{bmatrix}
f(1) \\
f(2) \\
\vdots \\
f(N)
\end{bmatrix}
\]

\textit{circulant matrix of} \( h \).
The Convolution Theorem

Eigenvalues of circulant matrix

We have \( g = h \ast f \Rightarrow g = C_h f \). Introduce

\[
\mathcal{F}_N = \begin{bmatrix}
\phi_0 & \phi_1 & \ldots & \phi_{N-1}
\end{bmatrix}, \quad \mathcal{F}_N = \text{the Fourier matrix}
\]

Lemma

\[
C_h \phi_i = H(i) \phi_i,
\]

where \( H = \mathcal{F}_N h \).

In other words: All circulant matrices have the same eigenvectors, i.e. the columns \( \phi_i \) of the conjugated Fourier Matrix. The eigenvalue corresponding to the eigenvector \( \phi_i \) is the \( i \):th Fourier coefficient of \( h \).
The Convolution Theorem
Eigenvectors of circulant matrix

Proof: Start with $C_h \phi_i = h \ast \phi_i = \phi_i \ast h$. This implies that

$$(C_h \phi_i)(x) = \sum_k \omega^{-(x-k)i} h(k) =$$

$$= \omega^{-xi} \sum_k \omega^{ki} h(k) = H(i) \omega^{-xi},$$

where $H_i$ denotes the Fourier transform of $h$ at frequency $i$. 
The Convolution Theorem
Eigenvectors of circulant matrix

The lemma gives

\[ C_h \mathcal{F}_N = \mathcal{F}_N \Lambda_H \]

Multiply with \( \mathcal{F}_N \) from the left:

\[ \mathcal{F}_N C_h \mathcal{F}_N = \mathcal{F}_N \mathcal{F}_N \Lambda_H = N I \Lambda_H = N \Lambda_H \]

Multiply then with \( \mathcal{F}_N \) from the right:

\[ \mathcal{F}_N C_h \mathcal{F}_N \mathcal{F}_N = N \Lambda_H \mathcal{F}_N \]

\[ \mathcal{F}_N C_h = \Lambda_H \mathcal{F}_N \]
The Convolution Theorem

Eigenvalues of circulant matrix

\[ C_h \overline{F_N} = \overline{F_N} \Lambda_H \]

\[ F_N C_h = \Lambda_H F_N \]
The Convolution Theorem

\[ \mathcal{F}_N C_h = \Lambda_H \mathcal{F}_N \]

\[ \mathcal{F}_N h \ast f = \begin{bmatrix} H(1)F(1) \\ \vdots \\ H(N)F(N) \end{bmatrix} \]

\[ \mathcal{F} (h \ast f) = \mathcal{F}_N C_h f = \Lambda_H \mathcal{F}_N f = \Lambda_H F = [H(1)F(1) \cdots H(N)F(N)]^T. \]
'Periodic' Convolution and Fourier Transform in 2D

\[ h = f \ast g \]

\[ H(u, v) = F(u, v)G(u, v) \]

\[ \text{FFT2}(f) \ast \text{FFT2}(g) = \text{FFT2}(\text{conv2}(f, g, 'periodic')) \]
Using FFT for convolutions

1. $f \rightarrow FFT \rightarrow F$
2. $h \rightarrow FFT \rightarrow H$
3. $H, F \rightarrow \times \rightarrow H \cdot F$
4. $H \cdot F \rightarrow IFFT \rightarrow h \ast f$

The computational complexity of using FFT for a convolution is:

$$2 \frac{N \log N}{2} + N + \frac{N \log N}{2} \sim \frac{3}{2} N \log N$$

Calculation based on the definition gives complexity $N^2$
**Frequency function**

\[ g(x) = h \ast f = \int h(x - y)f(y)dy \]

\[ \mathcal{F}g = G, \mathcal{F}h = H, \mathcal{F}f = F \]

\[ G(u, v) = H(u, v)F(u, v) \]

**Definition**

\( H = \mathcal{F}(h) \) is called the **frequency function** of \( h \).
Filter for image enhancement

<table>
<thead>
<tr>
<th>signal plane</th>
<th>frequency plane</th>
</tr>
</thead>
<tbody>
<tr>
<td>smoothing</td>
<td>low pass</td>
</tr>
<tr>
<td>sharpening</td>
<td>high pass</td>
</tr>
</tbody>
</table>

For discrete functions: \( DFT(h \ast f)(u, v) = H(u, v)F(u, v) \).
Let the output, \( g \) be given by the convolution

\[
g(x) = S(f)(x) = \int h(x - y)f(y)dy,
\]

where \( f \) represents the input and \( h \) the impulse response.

If \( g(x) \) only depends on \( f \)'s values in a surrounding (=a small window) of \( x \) then \( S \) is called a window operator.

The window is given by \( \{ x \mid h(x) \neq 0 \} \).
Assume that \( f(x, y) \) represents a continuous image. Let
\[
h(x, y) = \text{rect}(x) \text{rect}(y) .
\]
Then
\[
S(f) = h \ast f = \int_{K(x,y)} f(s, t) \, ds \, dt ,
\]
where the region of integration \( K(x, y) \) is a unit square with centre at \( (x, y) \).
S is called a **mean value operator**. The fourier transform gives

\[ H(u) = 4 \text{sinc}(2\pi u) \text{sinc}(2\pi v) . \]

The scaling rule (page 148 in Forsythe-Ponce)

\[ f(\lambda x) \rightarrow \frac{1}{\lambda} F\left(\frac{u}{\lambda}\right) . \]
signal space: image, filter, result

frequency space: image, filter result
signal space: image, filter, result

frequency space: image, filter result
signal space: image, filter, result

frequency space: image, filter result
Example

Notice that

\[
\phi(x) = \frac{1}{\sqrt{2\sigma^2\pi}} e^{-x^2/(2\sigma^2)} \quad \rightarrow \quad \Phi(u) = e^{-2(\sigma\pi u)^2}.
\]
Larger $\sigma$ gives
Example

Differentiation

\[ \frac{\partial f}{\partial x} \rightarrow 2\pi iuF(u) \]

\[ H(u) = 2\pi iu \]
Sensitive to noise.
Combine with smoothing:

\[ f \to \phi \ast f \to \frac{\partial}{\partial x} \phi \ast f \]

\[ \frac{\partial}{\partial x} \phi = -\frac{x}{\sqrt{2\sigma^6\pi}} e^{-x^2/(2\sigma^2)} \]
Filter example: differentiation in the $y$-direction

Kalle Åström
Image Analysis - Lecture 3
Filter example: differentiation in the x-direction

Kalle Åström
Image Analysis - Lecture 3
Filter example: Differentiation and Gaussian in $x$-direction

Kalle Åström

Image Analysis - Lecture 3
Filter example: Differentiation and Gaussian in $y$-direction

Kalle Åström
Image Analysis - Lecture 3
Filter example: Differentiation and Gaussian in a general direction

Kalle Åström

Image Analysis - Lecture 3
Filter example: Increasing $|\mathbf{H}(\mathbf{f})|$
Review

• Convolution (with flip) and cross-correlation (without flip)
  • Properties
  • Examples
• Convolution theorem
• Interpreting convolutions through the Fourier transform

• Read lecture notes
• Experiment with matlab demo scripts
• Finish assignment 1
Master’s Thesis Suggestion of the day