Medical image analysis TBMI02
Mini project

Anders Eklund (anders.eklund@liu.se), your teacher for:

- Mini project
- MRI lab
- Image registration lab
- Image segmentation lab
2D Enhancement of MR-image

Result from last years students

Original image.

Enhanced image,
Applications of image denoising

- Improve image quality (help medical doctors / radiologists to look at the images)
- Counteract an increased noise level caused by lowering the radiation dose for CT
- Preprocessing before image registration and image segmentation
- Preprocessing before visualization (e.g. volume rendering)
Image denoising algorithms

There are many algorithms for image denoising

- Lowpass filtering, removes noise but also important structures
- Anisotropic diffusion, Perona and Malik (1990)
- Bilateral filtering, Tomasi and Manduchi (1998)
- Non-local means, Buades et al. (2005)
Quadrature filtering applications

Quadrature filters are not only used for adaptive filtering (tensors). Can also be used for

- Feature detection / extraction, e.g. for machine learning
- Phase based image registration (image registration laboration)
- Phase based image segmentation

Filter optimization is important for all these filtering applications, but the goals may differ.

For phase based image registration, a smooth phase is most important.
Mini project: Adaptive filtering

- Filter optimization (2D), class 1
- Tensor representation of local structure, class 2
- Adaptive filtering image enhancement, class 2

2 classes

3 computer laborations

Lab-PM
Mini project: Examination

1. Demonstrate working code during lab.

2. Written report.
   - Answers to the questions in the lab-PM.
   - Lots of images of filters and intermediate results.
   - Motivate you settings and parameters.

Last date to hand in the report Dec 3

Test a lot at the laborations - plenty of time.

Work in groups of one or two people.
Mini project: Literature

Granlund, Knutsson: *Signal Processing for Computer Vision*

- **ch 4** the Fourier transform.
- **ch 5** Filter optimization + Advanced filter opt paper, sec 1-5.
- **ch 6** Orientation and tensor parts.
- **ch 10** Adaptive filtering.
Adaptive filtering in 2D, 3D, 4D

- In this course we will only work with 2D adaptive filtering
- The methods can (easily) be extended to 3D and 4D data
- 3D data are volumes or 2D + time (e.g. ultrasound)
- 4D data are volumes over time (e.g. fMRI, CT, ultrasound)
- Video of 4D adaptive filtering of CT data
Mini project: Filter optimization

Filter optimization wish list:

- Minimal deviation from an ideal frequency function (e.g. lognormal quadrature).
- Small spatial support (a large support can mix up several events / features).
- Adapted to the image spectrum.
- Adapted to the noise spectrum.

Conflicting demands that relates back to the uncertainty principle.

Some demands are easy to formulate in the Fourier domain (FD) (ideal frequency function).
Some demands are best formulated in the spatial domain (SD) (small filter).
## Mini project: Filter optimization

Filter optimization as a least squares problem using ideal and weight functions.

- $F_i(u)$ – frequency ideal function.
- $F_w(u)$ – frequency weight function.
- $f_i(x)$ – spatial ideal function.
- $f_w(x)$ – spatial weight function.

Note lower case letters for SD and upper case letters for FD.

The $B$ matrix transforms the filter from the spatial domain to the frequency domain.

\[
\epsilon^2 = \|F_w(F_i - B f)\|^2 + \alpha \|f_w(f_i - f)\|^2 \quad \frac{\partial \epsilon^2}{\partial f} = 0
\]

Use Matlab script `krnopt`
Mini project: Filter optimization

Why (weighted) least squares?

Based on the $L_2$ norm, i.e. total error $= \sum_i error_i^2$

The error function is convex, convex optimization problem

A global optimum can be found by setting the derivative (with respect to the filter coefficients) of the error function to 0, and then solving the resulting equation system

See the filter optimization paper for details

Note that it is harder to find the best solution for an $L_1$ norm, where total error $= \sum_i |error_i|$
Mini project: Filter optimization

\[ \epsilon^2 = \|F_w(F_i - Bf)\|^2 + \alpha\|f_w(f_i - f)\|^2 \]

\(\alpha (> 0)\) controls how important the error is in the spatial domain, compared to the frequency domain

Example for a filter with a spatial size of 11 x 11 pixels

\(f\) is 11 x 11, reshaped to 121 x 1 vector

\(f_i\) is 11 x 11, reshaped to 121 x 1 vector

\(f_w\) is 121 x 121 diagonal matrix (only values in the diagonal)

Norm of matrix A can be defined as \(\|A\|^2 = \sum_{i,j} A_{i,j}^2\)
Mini project: Filter optimization

$$\epsilon^2 = \|F_w(F_i - B f)\|^2 + \alpha\|f_w(f_i - f)\|^2$$

Use 23 x 23 samples in frequency domain
(as the Fourier transform of the filter is continuous, to get a better estimate of the error)

$F_i$ is 23 x 23, reshaped to 529 x 1 vector

$B$ is 529 x 121, performs base change from spatial domain to frequency domain

$F_w$ is 529 x 529 diagonal matrix (only values in the diagonal)

Norm of matrix $A$ can be defined as $\|A\|^2 = \sum_{i,j} A_{i,j}^2$
Mini project: Filter optimization

$F_i(u)$ – ideal function in the FD. Use e.g. the Matlab scripts `quadrature`, `enhlp`, `enhbp`, … to generate different $F_i(u)$.

$F_w(u)$ – weight function in FD. The spectrum of the image defines the importance of a close fit to $F_i(u)$. Use `F_weightgensnr` to generate $F_w(u)$. 
Mini project: Filter optimization

Example of frequency ideal function, quadrature filter (zero in half the frequency domain)

Center frequency $u_0 = \pi/3$, bandwidth $\beta = 2$ octaves

Lognormal radial function  \hspace{1cm} \cos^2$ angular function  \hspace{1cm} Quadrature filter in FD

Corresponding functions in kernel optimizer, quadrature = bplognorm .* cosangle

bplognorm  \hspace{2cm} \text{cosangle}  \hspace{2cm} \text{quadrature}
Mini project: Filter optimization

\[ f_i(x) - \text{ ideal function SD. Not } \mathcal{F}^{-1}(F_i(u))! \]

\[ f_i(x) = \begin{cases} 1 & x = 0 \\ 0 & x \neq 0 \end{cases} \]

The most compact filter there is.

\[ f_w(x) - \text{ weight function in SD. Preserve spatial locality.} \]

\[ f_w(x) \approx x^2. \]

Use \texttt{goodsw} to generate \( f_w(x) \).
Mini project: Filter optimization

Decide spatial support size $n \times n$ pixels

Always use an odd number $n = 7, 9, 11, \ldots$

Ex. lognormal quadrature filter $u_o, \beta$

- Center frequency $u_o = \sqrt{u_h \cdot u_l}$
- Bandwidth $\beta = \frac{\ln(u_h / u_l)}{\ln 2}$ [Octaves]
- Where $u_h$ and $u_l$ are 6dB cut off frequencies.

$u_o$ corresponds to the wave length $\lambda_0 = \frac{2\pi}{u_0}$

<table>
<thead>
<tr>
<th>$u_0$</th>
<th>$\lambda_0$ (pixels)</th>
<th>$n$ (pixels)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\pi/2$</td>
<td>4</td>
<td>7, 9</td>
</tr>
<tr>
<td>$\pi/4$</td>
<td>8</td>
<td>13, 15</td>
</tr>
<tr>
<td>$\pi/6$</td>
<td>12</td>
<td>23, 25</td>
</tr>
</tbody>
</table>

Let $n \geq 1.5 - 2\lambda_0$ pixels.
Mini project: Filter optimization

How can the center frequency $u_o$ and the bandwidth be selected?

What are the smallest structures that should be detected? Gives $u_h$

What are the largest structures that should be detected? Gives $u_l$

Calculate center frequency as $u_o = \sqrt{u_h \cdot u_l}$

Calculate bandwidth as $\beta = \frac{\ln(u_h / u_l)}{\ln 2}$

How do we figure out the wavelength of the structures we are interested in?

Use a local Fourier transform, centered at the structures of interest

LIFT: Local Interactive FT Demo
Mini project: Filter optimization

Coordinates and sampling in FD

\[ F(u) = \sum_{x=-\frac{n-1}{2}}^{\frac{n-1}{2}} f(x) e^{-iux} \]

\( f(x) \) is a discrete function.

\( F(u) \) is continuous (since the function \( e \) is continuous) and periodic (2 \( \pi \)).

A continuous function can be sampled arbitrarily dense

How to sample \( F(u) \) and how many samples, \( N \), should be used?
Sample $F(u)$ using FFT coordinates ($N$ is even)

$u \in \left[-\frac{N}{2} : \frac{N}{2} - 1\right] \frac{2\pi}{N}$  

- Samples in $-\pi, -\pi/2, 0, \pi/2$
- One sample in the origin is good!
- Impossible to have different values close to $\pi$ and $-\pi$

Sample $F(u)$ using KRNOPT coordinates ($N$ is odd)

$u \in \left[-\frac{N-1}{2} : \frac{N-1}{2}\right] \frac{2\pi}{N}$  

- Samples in $-4\pi/5, -2\pi/5, 0, 2\pi/5, 4\pi/5$
- One sample in the origin is good!
- symmetric for values close to $\pi$ and $-\pi$
Mini project: Filter optimization

How many samples $N$ to use in the FD?

We sample $F(u)$ and measure the error $F(u) - F_i(u)$ using $N$ sample points. For small $N$ ($N = n$) the error will be small at the sample points but may be very large in between!

Remember that $F(u)$ can be sampled arbitrarily dense.

Use as large $N$ as possible (larger $N$ will of course lead to more calculations, especially in 3D and 4D)

$N \geq 2n + 1$ (in each dimension)

SD: $15 \times 15$ pixels $\rightarrow$ FD: $31 \times 31$ samples in $]-\pi, \pi[$
Mini project: Filter optimization

Example of what happens when N is too small, in this case N = 3, error = 0
Mini project: Filter optimization

krnopt matlab script

\[
[f, F] = \text{krnopt}(\underbrace{F_i, F_w, f_m, f_i, f_w}_{FD, SD})
\]

All in and outputs are \(wa\) (weight arrays)

\[
wa = \begin{cases} 
    \text{Data: multidim array} \\
    \text{FD/SD-flag} \\
    \text{coordinate system (SD: pixels \ FD: \(]-\pi, \pi[\))}
\end{cases}
\]

putdata - to put a Matlab matrix into a wa
getdata - to get a Matlab matrix from a wa
Most element wise operations work for \(wa\) use \(.*\) and \(./\)
Many functions for generation of weight and ideal functions exist in the kerngen library e.g. quadrature, \(F\_weightgensnr\), goodsw, see labPM.
Mini project: Filter optimization

Filter optimization hints

- Use no spatial weight first time
- Compute error using `filterr`

Most important FD-error
\[
\delta_2 = \frac{\|F_w(F_i - F)\|}{\|F_w F_i\|}
\]
and DC-error (weighted and unweighted)
(a DC-error (0 Hz) is serious for a bandpass filter)

- plot filters in both domains, use `wamesh`
  - No ripple in the borders of SD kernel
  - Note shape and magnitude in the FD

- Increase spatial weight gradually and observe the error and the filter shapes.
Mini project: Filter optimization

Spatial weight = 0, filter is not flat at the edges (spatial domain), ripples in frequency domain for high frequencies (remember that the frequency weight is the highest for the low frequencies)
Mini project: Filter optimization

Spatial weight = 10, filter is flat at the corners (highest spatial penalty) but not at the edges

Spatial domain (real part)  Frequency domain
Mini project: Filter optimization

Spatial weight = 50, the filter is now rather flat at the corners as well as the edges. The filter also looks better in the frequency domain.

Spatial domain (real part)  Frequency domain
Mini project: Filter optimization

Spatial weight = 100, pretty similar to a spatial weight of 50

Spatial domain (real part)  Frequency domain
Mini project: Filter optimization

Spatial weight = 1000, virtually no ripples in the frequency domain, but the filter is further from the frequency ideal function (lognormal quadrature)
Mini project: Filter optimization

Spatial weight = 10000, the filter is extremely compact in the spatial domain (Dirac impulse), but the filter is very far from the frequency ideal function (lognormal quadrature)

Spatial domain (real part)  Frequency domain