Image Processing and Data Visualization with MATLAB

Filtering Images

Hansrudi Noser

June 28-29, 2010

UZH, Multimedia and Robotics Summer School

Contents

- Noise
- Smoothing Filters
- Sigmoid Filters
- Gradient Filters

Noise

- Digital images can contain a variety of types of noise.
- Noise often is the result of errors in the image acquisition process
  - pixel values that do not reflect the true intensities of a real scene
- Noise types
  - Gaussian noise
  - Salt and pepper noise
  - White noise

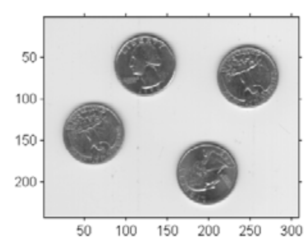
Gaussian Noise

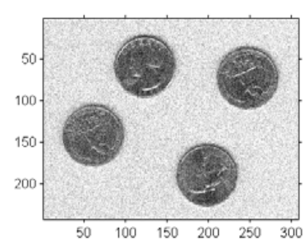
- In the additive noise model the image information is given by the true image value and an additive noise component.
- The additive deviations of the noise have a Gaussian distribution, normally with a mean of zero and a given standard deviation σ.

$$f(i,j) = s(i,j) + n(i,j)$$

- The ratio of signal and noise is given by (Signal to Noise Ratio, SNR):

$$SNR = \frac{\sigma_s}{\sigma_n} = \sqrt{\frac{\sigma_f^2}{\sigma_n^2} - 1}$$

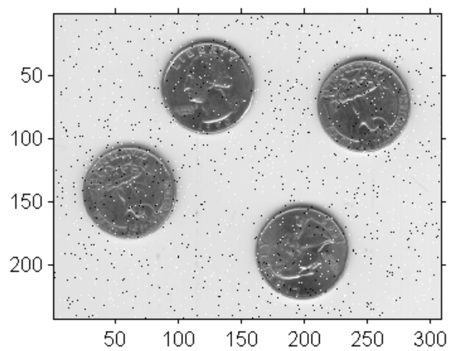




Imnoise, mean 0, variance 0.01

Salt and Pepper Noise

- *Salt and pepper* noise has extreme values and is typically produced by transmission errors.



White Noise

- White noise has a uniform frequency distribution

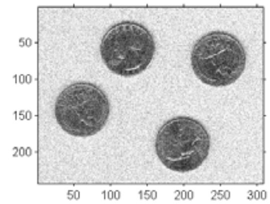
Filters

- Smoothing filters
  - Mean filtering
  - Median filter
  - Gaussian filter
- Sigmoid filter
- Gradient filter
  - Discrete deviation
  - Laplacian of Gaussian
  - Edge detection
    - Thresholds
    - Zero crossings
    - Canny method

Mean filtering

- Mean filtering, smoothing, averaging, box filtering
- The mean filter smoothes a signal and reduces noise.
- Each pixel is replaced by the mean value of its neighbors. It is a low pass filter and eliminates high frequencies.

Mean filtering

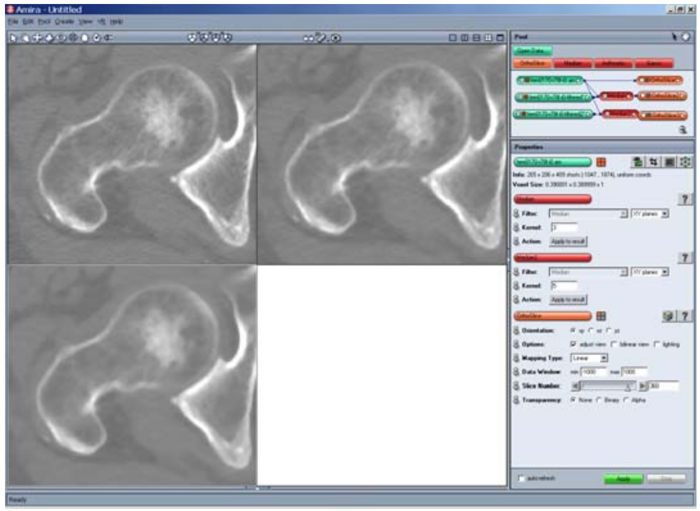


- The mean filter is a convolution filter defined by its kernel
- 1D: $\begin{bmatrix} 1/3 & 1/3 & 1/3 \end{bmatrix}$ 2D: $\begin{bmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{bmatrix}$
- If there exist high noise peaks (salt and pepper noise), a median filter produces better results.
- Multiple application of small filters (3x3) preserve better edges than the use of large filters (7x7).

Median or rank filter

- A median filter is used to reduce noise.
- It preserves edges and details better than average filters.
- It is not a linear filter but it sorts the neighbors and uses the median as new pixel value. If the number of neighbors is even, the average value of both middle pixels is taken.

Median filter example



The effect of median filters on a grayscale image (3x3 Kernel, upper right; 5x5 Kernel, lower left).

Gaussian filter

- A Gaussian filter smoothes images and reduces noise, but also the image resolution
- Its kernel represents the Gaussian curve given by:
- 1D: $G(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x^2}{2\sigma^2}}$ 2D: $G(x) = \frac{1}{\sigma^2 2\pi} e^{-\frac{x^2+y^2}{2\sigma^2}}$

Gaussian Filter

- In practice such an infinite filter is approximated by a finite convolution filter that can be computed according to a given standard deviation.
- In the one dimensional case a convenient kernel size is between the doubled or tripled standard deviation:

$$2\sigma < k < 3\sigma$$

$$k = round\left(2.5 \cdot \sigma\right)$$

- The larger the standard deviation, the stronger is also the smoothing effect of the filter, and the larger the kernel size has to be.

Gaussian Filter

- The following equations compute one dimensional Gaussian kernel in real and integer arrays.

$$Kernel_{Gauss,real}(x,\sigma) = \left[G(-k), G(-k+1), ..., G(-1), G(0), G(1), ..., G(k-1), G(k)\right] = \left[G(k), G(k-1), ..., G(1), G(0)G(1), ..., G(k-1), G(k)\right]$$

$$Kernel_{Gauss,\text{int}}(x,\sigma) = \frac{1}{s}\left[1, round\left(\frac{G(k-1)}{G(k)}\right), ..., round\left(\frac{G(1)}{G(k)}\right), round\left(\frac{G(0)}{G(k)}\right), round\left(\frac{G(1)}{G(k)}\right), ..., round\left(\frac{G(k-1)}{G(k)}\right), 1\right]$$

$$s = 2 + round\left(\frac{G(0)}{G(k)}\right) + 2 \cdot \sum_{i=1}^{k-1} round\left(\frac{G(i)}{G(k)}\right)$$

The normalization factor s reduces the sum of all kernel elements to one

Gaussian Filter

- The Gaussian filter computes a weighted average of the neighbor pixels. Therefore it preserves better edges than a simple average filter. The Gaussian filter is circular symmetric and separable.
- Therefore a 2D operator can be composed by two 1D operators in x and y direction

$$G(x, y, \sigma) = \frac{1}{\sigma^2 2\pi} e^{-\frac{x^2+y^2}{2\sigma^2}} = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x^2}{2\sigma^2}} \cdot \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{y^2}{2\sigma^2}}$$

$$G_{\sigma_2} * \left(G_{\sigma_1} * f\right) = G_{\sqrt{\sigma_1{}^2+\sigma_2{}^2}} * f$$

Gaussian Filter Examples

$\sigma = 1 \Rightarrow k = round(2.5 \cdot 1) = 3$

- 0.0044 0.054 0.242 0.3989 0.242 0.054 0.0044

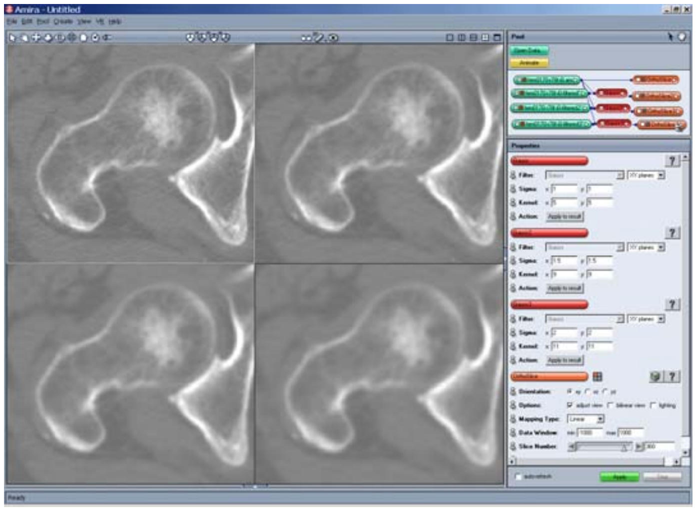
$\sigma = 1.5 \Rightarrow k = round(2.5 \cdot 1.5) = 4$

- 0.0076 0.036 0.1093 0.213 0.266 0.213 0.1093 0.036 0.0076

$\sigma = 2 \Rightarrow k = round(2.5 \cdot 2) = 5$

- 0.0088 0.027 0.0648 0.121 0.176 0.1995
0.176 0.121 0.0648 0.027 0.0088

Gaussian Filter Example



The effect of Gaussian filters on a grayscale image (upper left). (Upper right: sigma 1, 5x5 kernel; lower left: sigma 1.5, 9x9 kernel; lower right: sigma 2, 11x11 kernel)

Sigmoid Filter

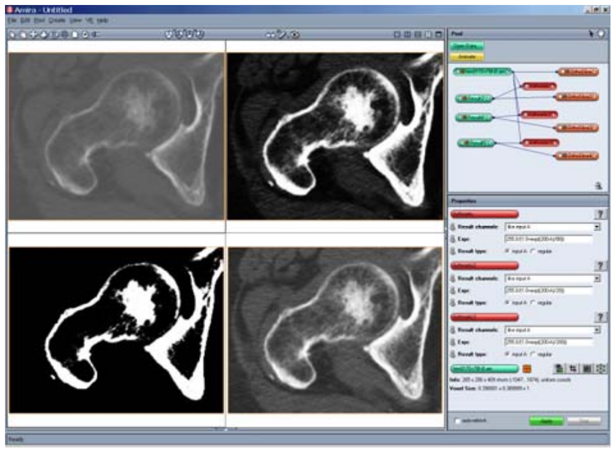
- With a Sigmoid filter a gray value range *a* around a given value *b* can be enhanced.
- It is a one pixel operator.
- The new pixel value is determined by the following equation:

$$f(x) = \min + \frac{\max - \min}{1 + e^{\frac{b-x}{a}}} \in [\min, \max]$$

- The values a and be define the value range to be enhanced and the interval [min,max] determines the target range of the new values. We get:

$$f(b) = 0.5(\max - \min)$$
$$f(-\infty) = \min$$
$$f(\infty) = \max$$

Sigmoid Filter



Sigmoid-Filter applied on a grayscale image (upper left): (upper right: a=80; lower left: a=20; lower right: a=200). The gray value window of the original window is [0, 255]. The window of the target images is the same. The centre of the value range to enhance is 200 (min=0, max=255, b=200).

Gradient Filter: Discrete Derivation of order one

- The first partial derivation in an image is given by

$$\frac{\partial f(x,y)}{\partial x} \approx g_x = \frac{f(x,y) - f(x-\Delta x, y)}{\Delta x} \qquad \frac{\partial f(x,y)}{\partial y} \approx g_y = \frac{f(x,y) - f(x, y-\Delta y)}{\Delta y}$$

- In the discrete case we get

$$\Delta x = 1, \Delta y = 1$$
$$g_x = f(x,y) - f(x-1, y)$$
$$g_y = f(x,y) - f(x, y-1)$$

- The corresponding kernels are given by: $\begin{bmatrix} -1 & 1 \end{bmatrix}$ $\begin{bmatrix} -1 \\ 1 \end{bmatrix}$

Gradient Image

- The gradient image g(x,y) shows the regions of the largest variation in the image.
- It consists of the lengths of the gradient vectors $\begin{bmatrix} g_x \\ g_y \end{bmatrix}$

$$g(x,y) = \sqrt{g_x^{\ 2} + g_y^{\ 2}}$$

- The directions of the 2D vectors are given by the angle

$$\Phi(x,y) = \arctan \frac{g_x}{g_y}$$

Prewitt and Sobel Gradient Filter

- Use of larger neighborhood to reduce noise

$$\frac{\partial f(x,y)}{\partial x} = \lim_{h \to 0} \frac{f(x+h,y) - f(x-h,y)}{2h} \overset{h=1}{\approx} \frac{f(x+1,y) - f(x-1,y)}{2}$$

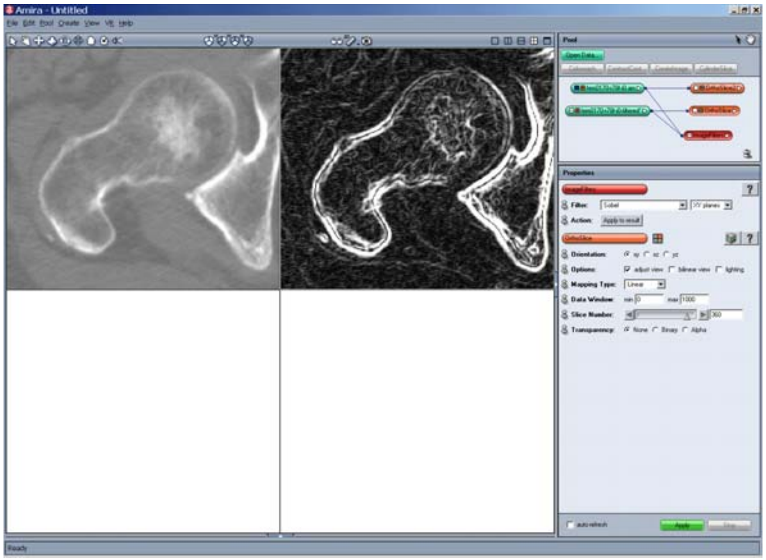
- Prewitt filter:
  - 3 point neighborhood
  - Strong effect on horizontal and vertical edges
  - Direction dependent

$$\begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}, \quad \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$

- Sobel filter:
  - Weighted smoothing
  - Slightly directional (horizontal and vertical edges)
  - Most popular gradient filter

$$\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}, \quad \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

Sobel Filter Example



Laplace Filter (2nd Derivation)

- The second derivation in x-direction:

$$g_{xx} = g_{x+1} - g_x =$$
$$\left(f(x+1,y) - f(x,y)\right) - \left(f(x,y) - f(x-1,y)\right) =$$
$$f(x+1,y) + f(x-1,y) - 2f(x,y)$$

- The second derivation y-direction

$$g_{yy} = f(x,y+1) + f(x,y-1) - 2f(x,y)$$

- The Laplace operator is given by the sum of the second partial derivations

$$\nabla^2 f(x,y) \approx \frac{\partial^2 f(x,y)}{\partial x^2} + \frac{\partial^2 f(x,y)}{\partial y^2}$$

- The corresponding 1D kernel matrices $\begin{bmatrix} 1 & -2 & 1 \end{bmatrix}$ $\begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix}$

2D Laplace Filter

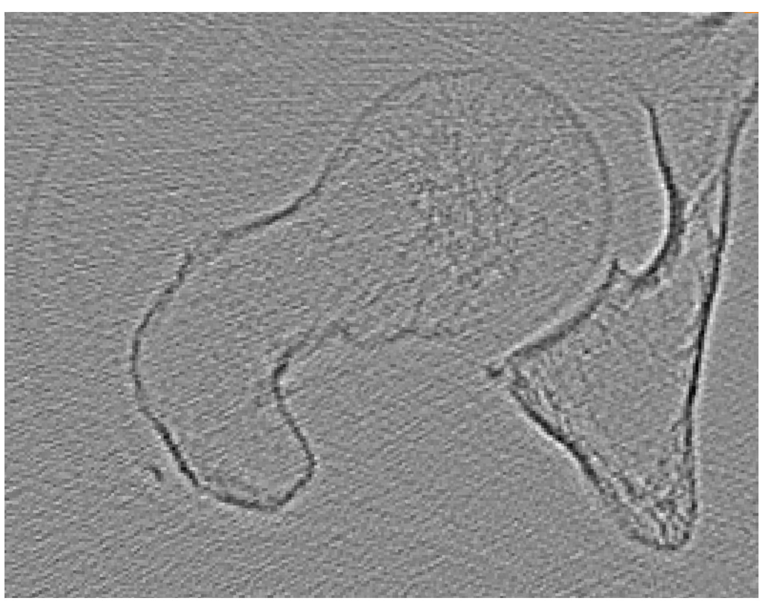
- In the 2D case the Laplace operator is:

$$\nabla^2 f(x,y) = g_{xx} + g_{yy} = f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y)$$

- 4 neighborhood kernel: $\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$

- 8 neighborhood kernel: $\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$
- Properties:
  - Rotation invariant
  - Sensitive to noise
  - Strong detection of single points, thin lines, and end of lines (only 1 pixel thick)

Laplace Operator Example



Laplacian of Gaussian

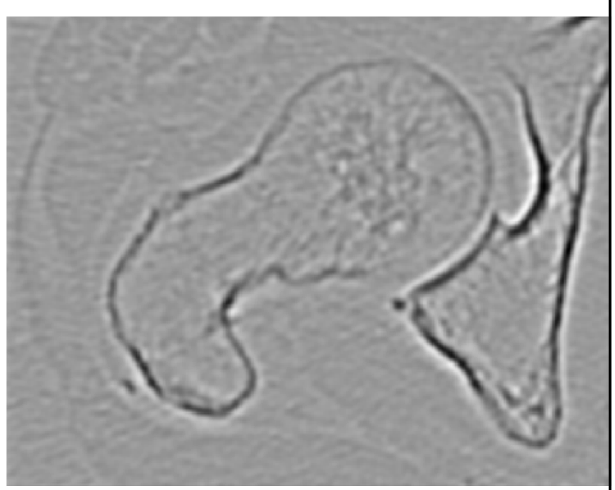
- As the Laplace filter is **sensitive to noise** the image should be smoothed before using it.
- The Laplacian of Gaussian (**LOG**) combines a **low pass** filter with a **high pass filter**, i.e. it smoothes first the image and enhances afterwards the contrast.
- According to neuro physiological studies this principle is **similar to human vision**.
- First a Gaussian filter is applied, followed by a Laplace filter:

$$\nabla^2 \left(G(x,y) * f(x,y) \right) = \left(\nabla^2 G(x,y) \right) * f(x,y)$$

$$\nabla^2 G(x,y) = \frac{1}{\pi\sigma^4} \left(\frac{x^2+y^2}{2\sigma^2} - 1 \right) e^{-\frac{x^2+y^2}{2\sigma^2}}$$

DOG or Mexican-Hat Operator

- The LOG can be approximated by the difference of two Gaussian filters with different variances, called also Difference of Gaussian (DOG) or Mexican-Hat Operator because of its shape



LOG Example

- The filter size should be at least 6 . In the slide before a 13x13 Filter with Sigma = 1.5 was used. It is given by:

0.0000	0.0000	0.0000	0.0000	0.0001	0.0001	0.0001	0.0001	0.0001	0.0000	0.0000	0.0000	0.0000
0.0000	0.0000	0.0001	0.0002	0.0005	0.0009	0.0011	0.0009	0.0005	0.0002	0.0001	0.0000	0.0000
0.0000	0.0001	0.0003	0.0011	0.0025	0.0040	0.0046	0.0040	0.0025	0.0011	0.0003	0.0001	0.0000
0.0000	0.0002	0.0011	0.0035	0.0066	0.0083	0.0085	0.0083	0.0066	0.0035	0.0011	0.0002	0.0000
0.0001	0.0005	0.0025	0.0066	0.0083	0.0023	-0.0029	0.0023	0.0083	0.0066	0.0025	0.0005	0.0001
0.0001	0.0009	0.0040	0.0083	0.0023	-0.0224	-0.0392	-0.0224	0.0023	0.0083	0.0040	0.0009	0.0001
0.0001	0.0011	0.0046	0.0085	-0.0029	-0.0392	-0.0629	-0.0392	-0.0029	0.0085	0.0046	0.0011	0.0001
0.0001	0.0009	0.0040	0.0083	0.0023	-0.0224	-0.0392	-0.0224	0.0023	0.0083	0.0040	0.0009	0.0001
0.0001	0.0005	0.0025	0.0066	0.0083	0.0023	-0.0029	0.0023	0.0083	0.0066	0.0025	0.0005	0.0001
0.0000	0.0002	0.0011	0.0035	0.0066	0.0083	0.0085	0.0083	0.0066	0.0035	0.0011	0.0002	0.0000
0.0000	0.0001	0.0003	0.0011	0.0025	0.0040	0.0046	0.0040	0.0025	0.0011	0.0003	0.0001	0.0000
0.0000	0.0000	0.0001	0.0002	0.0005	0.0009	0.0011	0.0009	0.0005	0.0002	0.0001	0.0000	0.0000
0.0000	0.0000	0.0000	0.0000	0.0001	0.0001	0.0001	0.0001	0.0001	0.0000	0.0000	0.0000	0.0000

Edge detection

- Edge detection is an important topic in image processing and feature extraction.
- With gradient filters edges can be enhanced.
- However these filters are very sensitive for noise
- It is necessary to smooth images beforehand.

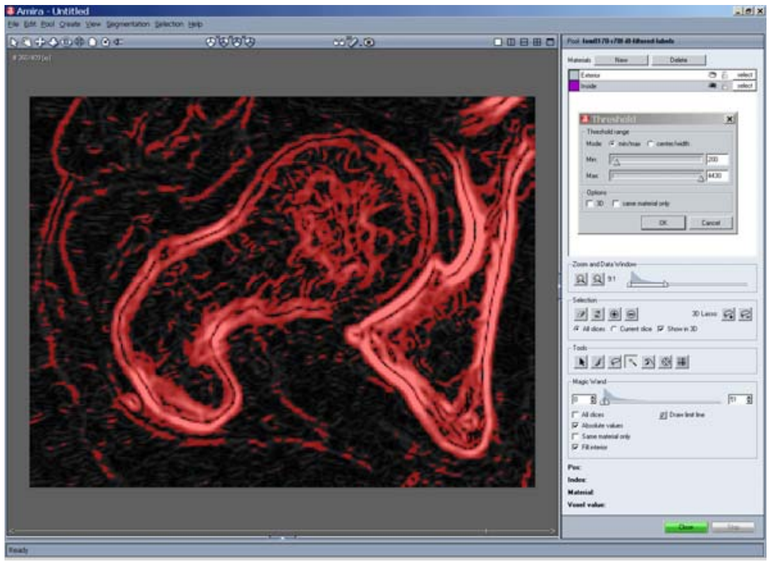
Edge detection: Thresholds

- A simple method for detecting edges consists in
  - enhancing edges with a gradient filter
  - then to apply thresholding to label the edges by choosing a threshold and
  - defining all pixels that are larger than the threshold T as edge pixels.
- However in practice this method is sensitive to noise and doesn't work well.
- The choice of the threshold value is critical.

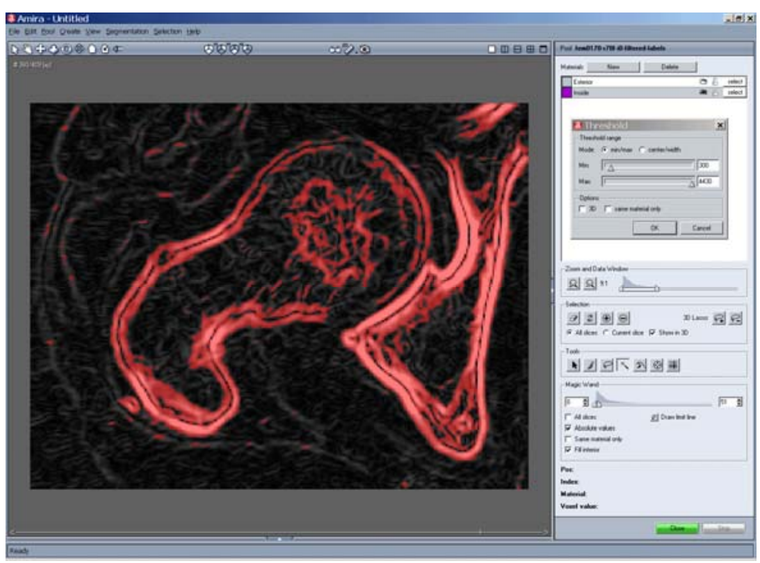
Examples of Thresholding

- The following images have been processed
  - with a Gaussian filter (3x3, 1),
  - a Sobel filter,
  - and labeled by thresholding with different threshold values (200, 300, 400, 500).
- Low threshold values segment also noisy parts
- High thresholds miss weak edges.

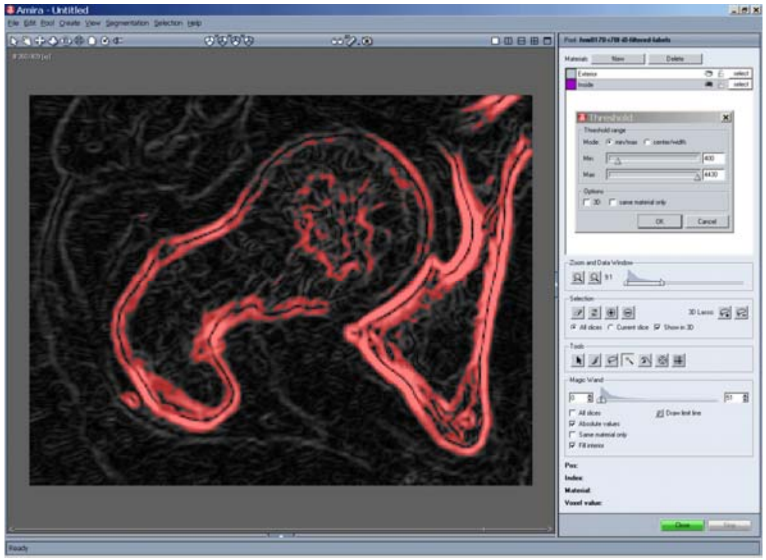
Threshold: 200



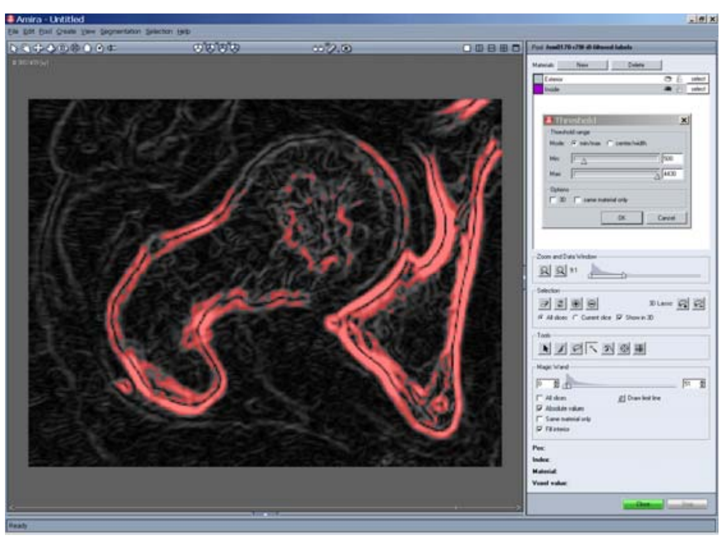
Threshold: 300



Threshold: 400



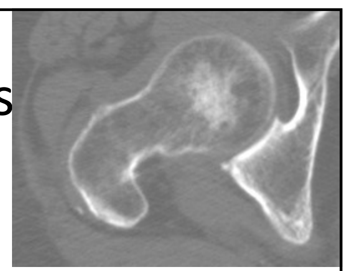
Threshold: 500



Zero Crossings with Laplacians

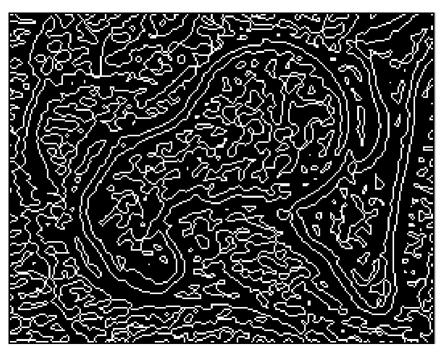
- Another possibility for detecting edges in images consists in using Laplacian filters, and in marking the zero crossings as edges.
- This procedure produces sharp edges but is strongly disturbed by the presence of noise.

Example of Zero Crossings



Two LOG (default sigma 2),

automatic threshold 0.027



threshold 0

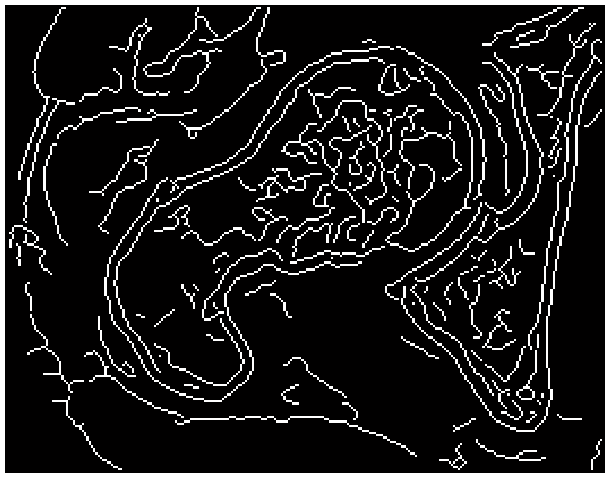
Canny method

- The Canny method applies two thresholds to the gradient:
  - a high threshold for low edge sensitivity and
  - a low threshold for high edge sensitivity.
- The method
  - starts with the low sensitivity result and
  - then grows it to include connected edge pixels from the high sensitivity result.
- This helps fill in gaps in the detected edges.
- The Canny method is robust and is frequently used in practice

Canny Method: Principle

- First the image is smoothed with a Gaussian kernel.
- Then a gradient image is produced.
- Next the regions around local maxima are suppressed in order to produce only one pixel large edges.
- Then the edges are refined by a two threshold method.
  - If a pixel has a value larger than the higher threshold then it is immediately recognized as edge pixel.
  - If a value is below the lower threshold then the pixel is immediately rejected.
  - If the pixel value is between both thresholds then it is only accepted if there exists a path to an edge pixel whose value is larger than the higher threshold.

Canny Method: Example



Canny method for edge detection applied on 8bit JPG image, Threshold Min Max: 0.0375 0.0938, Sigma Gauss: 1