BME I5000: Biomedical Imaging

Lecture 11
Point Spread Function, Inverse Filtering, Wiener Filtering, Sharpening, ...

Lucas C. Parra, parra@ccny.cuny.edu

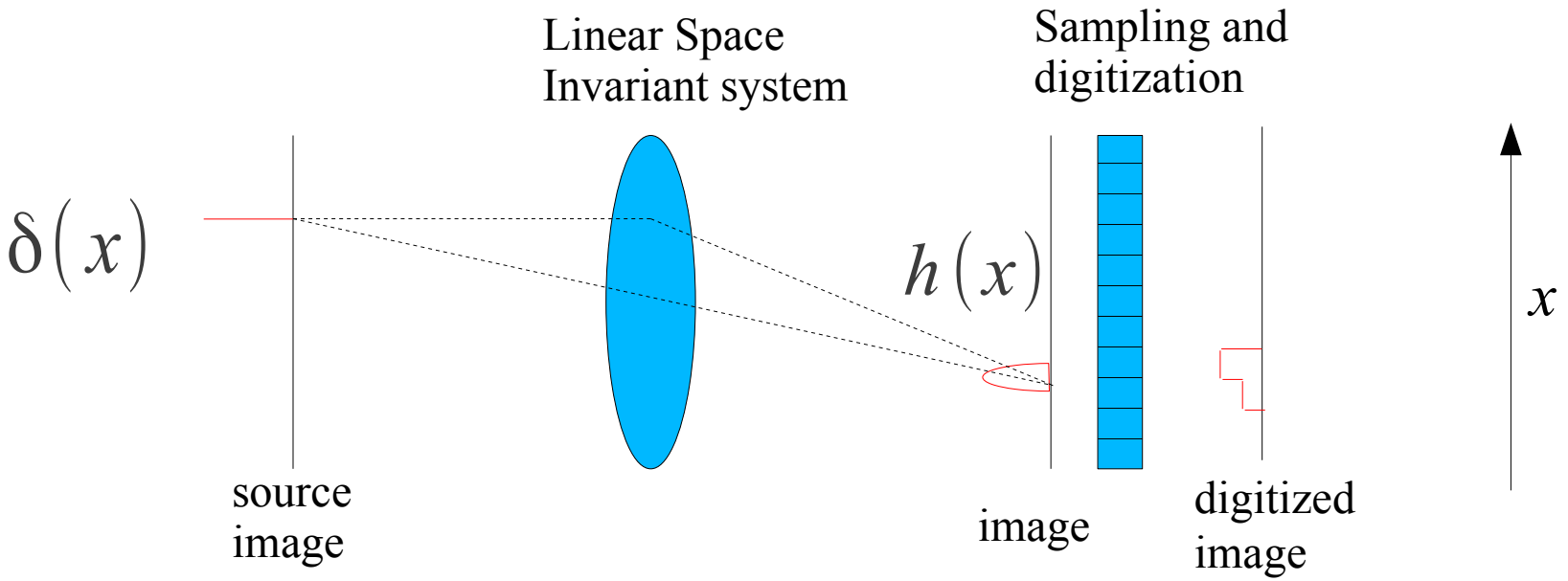
http://parralab.org/courses.html

Schedule

1. Introduction, Spatial Resolution, Intensity Resolution, Noise
2. X-Ray Imaging, Mammography, Angiography, Fluoroscopy
3. Intensity manipulations: Contrast Enhancement, Histogram Equalisation
4. Computed Tomography
5. Image Reconstruction, Radon & Fourier Transform, Filtered Back Projection
6. Nuclear Imaging, PET and SPECT
7. Maximum Likelihood Reconstruction
8. Magnetic Resonance Imaging
9. Fourier reconstruction, k-space, frequency and phase encoding
10. Optical imaging, Fluorescence, Microscopy, Confocal Imaging
11. Enhancement: Point Spread Function, Filtering, Sharpening, Wiener filter
12. Segmentation: Thresholding, Matched filter, Morphological operations
13. Pattern Recognition: Feature extraction, PCA, Wavelets
14. Pattern Recognition: Bayesian Inference, Linear classification

Model Imaging System

Model for a simple imaging system ...



often assumes linear shift invariance (LSI). The image of a point like source, $s(x)=\delta(x)$, on the detector combining all blurring effects of the imaging process is called the ***Points Spread Function*** $h(x)$:

$$h(x)=LSI[\delta(x)]$$

Model Imaging System

A common simplified mathematical model for an imaging process is that of a linear system with additive noise:

The source image $s(x,y)$ passes through a Linear Sift Invariant transformation $h(x,y)$ and sensing generates additive noise $n(i,j)$. The linear transformation is given by a convolution:

$$g(x,y)=h(x,y)*s(x,y)+n(x,y)$$

Point Spread Function

The image of an arbitrary source $s(x)$ is then given by a convolution

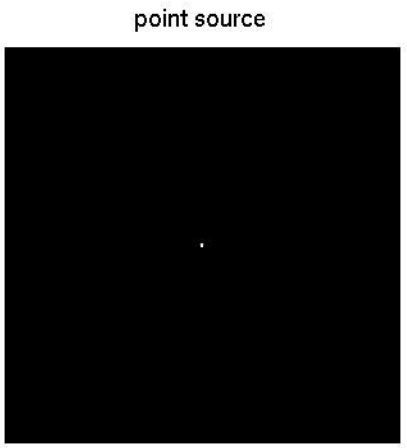
$$\begin{aligned} g(x) &= LSI[s(x)] = LSI\left[\int dx'\,\delta(x-x')s(x')\right] \\ &= \int dx'\,s(x')\,LSI[\delta(x-x')] \\ &= \int dx'\,s(x')h(x-x') \\ &= \int dx'\,h(x')s(x-x') \\ &= h(x)*s(x) \end{aligned}$$

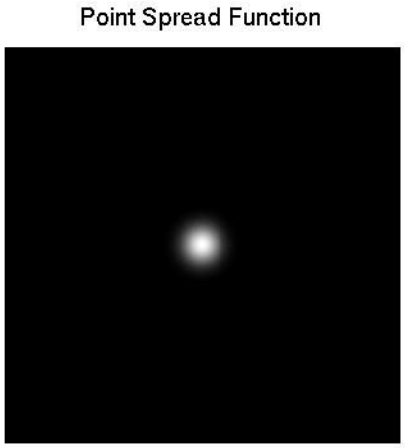
For a 2D discrete array (image) we write the convolutions as

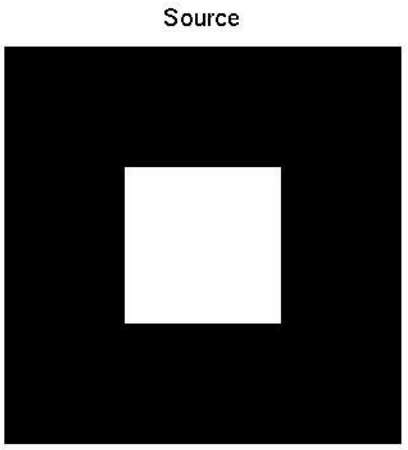
$$\begin{aligned} g(x,y) &= \sum_{x'=1}^{N}\sum_{y'=1}^{M} h(x-x',y-y')s(x',y') \\ &= h(x,y)*s(x,y) \end{aligned}$$

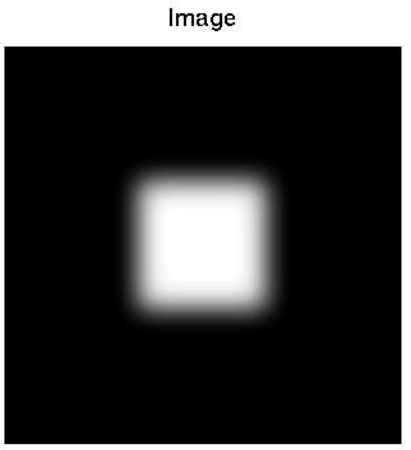
Point Spread Function

```
>> g = conv2(h,s);
```









PSF – Smoothing, Sharpening

There are a few simple choices of PSF we can apply to an image to improve image quality:

Smoothing: Simple low pass filter to remover high frequency noise

$$h_{LP}(x,y) = 1/16 \times \begin{array}{|c|c|c|}\hline 1 & 2 & 1 \\ \hline 2 & 4 & 2 \\ \hline 1 & 2 & 1 \\ \hline \end{array}$$

Sharpening: Simple high pass filter that enhances edges, equivalent to a second derivative (Laplacian filter)

$h_{HP}(x,y)$ =

0	1	0
1	-4	1
0	1	0

or

$h_{HP}(x,y)$ =

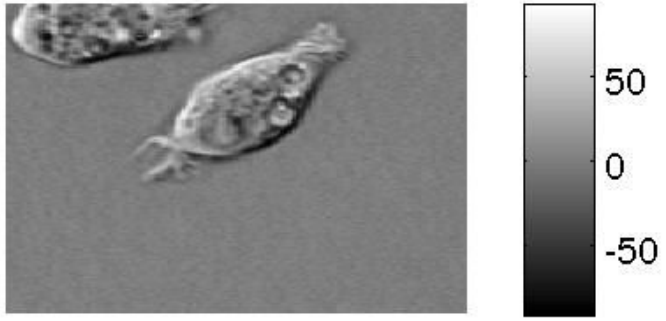
-1	-1	-1
-1	8	-1
-1	-1	-1

Often h_{LP} is implemented with a 2D Gaussian PSF, and h_{HP} with its second derivative. This way it is easier to control the scale.

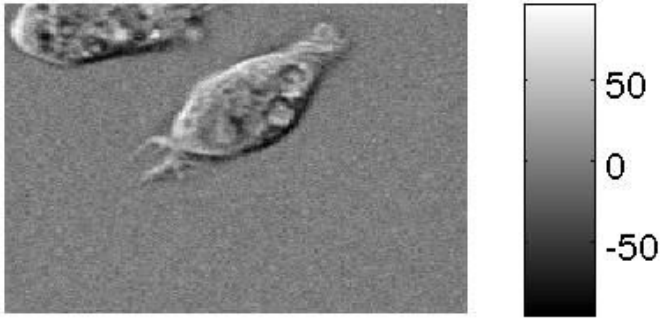
Un-sharp masking: $h_{hp}(x,y) = \delta(x,y) - h_{lp}(x,y)$

PSF – Smoothing, Sharpening

original image



0dB noise added



Low-pass filtered

HP filtered original

thresholded at 50

connected regions

Assignment – simple edge detection

- Use an image from the previous slide and determine the circumference.
- Filter the image to remove background noise and highlight edges.
- Use automatic theresholding to determine the edges, e.g. Otsu's method.
- You can filter the relevant areas using morphological image operations on the thresholded image, e.g. "open" and "close".
- You can also use region properties to select the area that belongs to the edge.
- Hopefully it is singly-connected and you can determine the length of the edge, if not, again you can use morphological operations to "close" the edge.
- The goal is not to get this perfectly, but learn the effects of the different parameters. (low-pass filter size, threshold value, region size, size of "structure element"etc.).
- Compare this to Canny and Sobel edge detection, see matlab function edge().

Assignment – simple segmentation

- The objective of this project is to measure the size of the mouse lungs in the CT volume “mouse_ct.mat” posted on the website.
- Filter the image to remove background noise.
- Use theresholding to capture the dark area of the lung volume.
- You can filter the relevant areas using morphological image operations on the thresholded image, e.g. “open” and “close”.
- You can also use region properties to rule out area that do not belongs to lung or are inside the lung.
- The goal is not to get this perfectly, but learn the effects of the different parameters. (low-pass filter size, threshold value, region size, size of “structure element”etc.).
- There are 3 repeated volumes in the same image. Measure and report the size of each of the 3 volumes

Point Spread Function - K-space

In ***k-space*** (Fourier Domain) this can be written as

$$G(k_x, k_y) = H(k_x, k_y)\, S(k_x, k_y) + N(k_x, k_y)$$

where G, H, S, and N are the 2D Fourier Transforms (FT) of g, h, s, and n respectively.

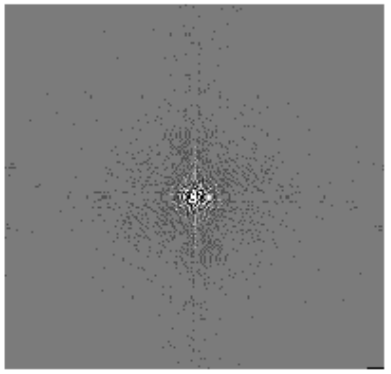
Often one considers the ideal noise free case, N=0:

$$G(k_x, k_y) = H(k_x, k_y)\, S(k_x, k_y)$$

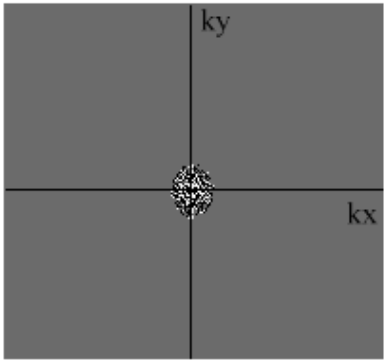
Demonstrate: Low-pass filter, high-pass filter, band-pass filter, Laplacian, un-sharp masking, etc. in MATLAB.

k-space

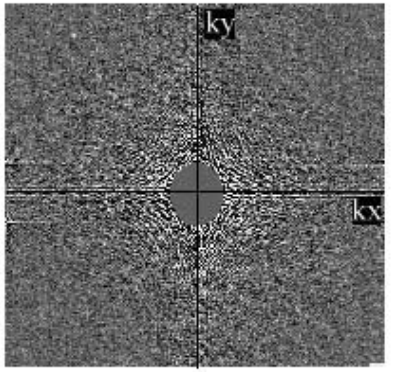
Effect of *H* on *S* in k-space:



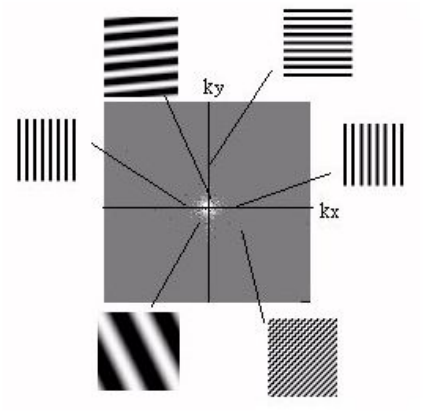
The Mona Lisa in k-Space



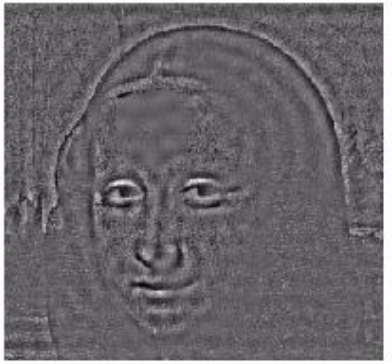
k-Space



k-Space



Low Frequency Mona



High Frequency Mona

Inverse Filtering

In the case of zero noise, N=0, and a known PSF we can undo the effect of H and recover the original image with an inverse filter:

$$S(k_x, k_y) = \frac{1}{H(k_x, k_y)} G(k_x, k_y)$$

That is by convolving with the inverse FT of H^{-1}:

$$s(x, y) = FT^{-1}\left[\frac{1}{H(k_x, k_y)}\right] * g(x, y)$$

Inverse Filtering

Problem: $H(k_x,k_y)$ may be zero or very small for some frequencies. At those frequencies even small noise will be stronger than the signal, and $1/H(k_x,k_y)$ will primarily enhance the noise!

Solution: Wiener Filtering

Assignment 10:
1. Filter (convolve) an image with an PSF of your choice.
2. Compute the inverse filter using the DFT.
3. Recover the original image by filtering with this inverse filter.
4. Show all three images, your filter, and its inverse filter, and the difference image between original and recovered image.
5. Corrupt the filtered image with additive noise (10dB SNR) and repeat step 3 and 4.
6. Now repeat the same (convolution+noise) and recover the original image using a Wiener filter.

Wiener Filtering

Wiener filter considers the effect of noise N and the magnitude of H . It give the optimal estimate $\hat{S}(k_x, k_y)$ as:

$$\hat{S}(k_x, k_y) = \frac{\sigma_S^2 H^*}{|H|^2 \sigma_S^2 + \sigma_N^2} G(k_x, k_y)$$

$\sigma^2_S = E[|S|^2]$, $\sigma^2_N = E[|N|^2]$ are the **power-spectra** of s and n: Expected value of the absolute square of the FT.

$|H|^2$ is the **magnitude response** of h : Absolute square of the FT.

(Dependency on x, y and k_x,k_y is omitted here to simplify notation)

This estimate is 'optimal' in that it represents the maximum *aposteriory* (MAP) estimate assuming zero mean Gaussian distributed spectra.

Prob. Estimation - Maximum *A Posteriori*

review

Sometimes *prior information* on the model parameters is available in the form of a **prior probability density**

$$p_S(S)$$

This prior information can be combined with the likelihood of the data with Bayes' theorem

$$p(S|data) = \frac{p(data|S)\, p_S(S)}{p(data)}$$

This is the conditional distribution of the parameter S after having observed *data*. It is therefor called **posterior probability density**.

Prob. Estimation - Maximum *A Posteriori*

review

The ML estimate can be biased towards the prior assumption using the posterior.

This gives the **maximum *a posteriori* (MAP)** estimate

$$\hat{S} = \arg\max_S \; p(S|data)$$

The MAP estimate gives the

most probable model given the observations.

Notice how this biases the ML estimate:

$$\hat{S} = \arg\max_S \; \ln p(S|data)$$
$$= \arg\max_S \; \left[\ln p(data|S) + \ln p(S)\right]$$

Wiener Filtering

In this case we are trying to estimate parameters S from *data* G, and assume that S and N are Gaussian:

$$p_S(S) \propto \exp\left(-\frac{|S|^2}{2\sigma_S^2}\right) \qquad p_N(N) \propto \exp\left(-\frac{|N|^2}{2\sigma_N^2}\right)$$

The log aposteriory probability is then

$$\begin{aligned}\ln p(S|G) &= \ln p_N(G|S) + \ln p_S(S) + const. \\ &= -|HS-G|^2/(2\sigma_N^2) - |S|^2/(2\sigma_S^2) + const.\end{aligned}$$

Setting derivative $\mathrm{d}\ln p(S|G)/\mathrm{d}S^* = 0$, and solving for S gives the Wiener Filter solution:

$$\hat{S} = \frac{\sigma_s^2 H^*}{|H|^2\sigma_s^2 + \sigma_N^2} G$$

Wiener Filtering

Back in the space domain this is

$$\hat{s}(x,y) = FT^{-1}\left[\frac{\sigma_s^2 H^*}{|H|^2\sigma_s^2 + \sigma_N^2}\right] * g(x,y)$$

Three important limiting cases:

For $N=0$ the filter [ ] becomes a inverse filter: $\left[H^{-1}\right]$

For H=1, filter reduces to
i.e. 0/1 gain for low/high SNR respectively. $\left[\frac{\sigma_s^2}{\sigma_s^2 + \sigma_N^2}\right]$

For, $H \approx 0$, or ,SNR=$\sigma_S/\sigma_N \approx 0$, the filter [ ] reduces to: $\left[\frac{\sigma_s^2}{\sigma_N^2} H^*\right]$

Wiener Filtering

Back in the space domain this is

$$\hat{s}(x,y) = FT^{-1}\left[\frac{\sigma_s^2 H^*}{|H|^2\sigma_s^2 + \sigma_N^2}\right] * g(x,y)$$

Multiplying with H^* corresponds to a correlation with H:

$$\sum_{x'=-\infty}^{\infty} g[x+x']h^*[x'] \quad \Leftrightarrow \quad G(k)H^*(k)$$

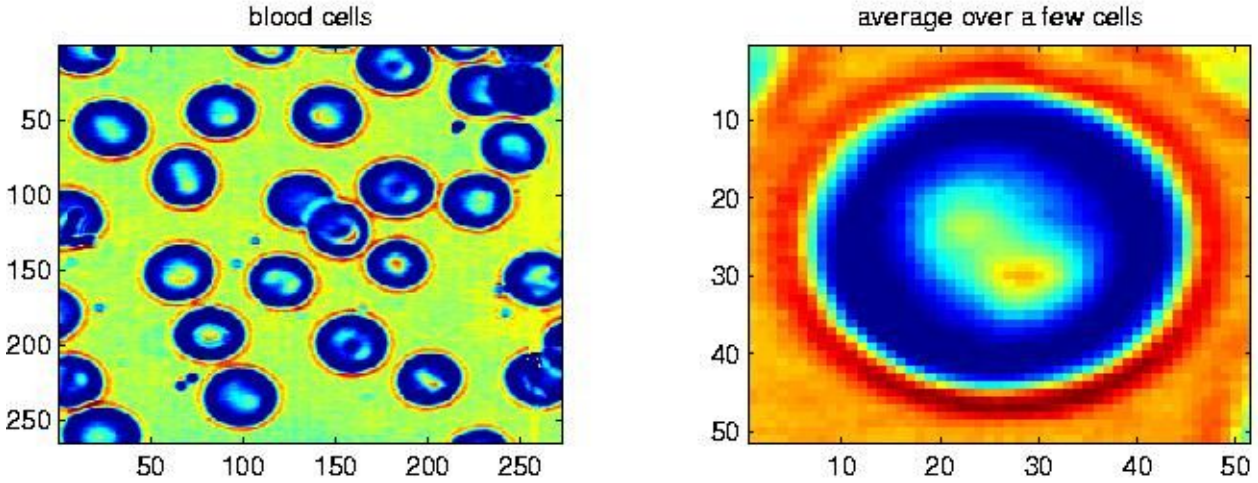
To understand its effect a filter that does not affect the magnitude of any frequency, but only its phase, i.e. $|H|^2=1$:

$$\hat{S} = H^* G = H^* H S = S$$

Therefore H^* corrects the phase effects of H.

Template matching

Sometimes one would like to find all the occurrences of a pattern h(x,y) in an image:



We want to find the locations x_o,y_o where the image $g(x,y)$ *matches* the pattern $h(x,y)$:

$$x_o, y_o = \underset{x',y'}{argmin} \sum_{x\,y} |g(x+x', y+y') - h(x,y)|^2$$

Template matching

The minimum is achieved at the maximum of of the correlation between g and h.

$$c_{gh}(x',y')=\sum_{xy} g(x+x',y+y')h(x,y)$$

$$x_o,y_o=\underset{x',y'}{argmax}\, c_{gh}(x',y')$$

Note that the correlation in the frequency domain can be computed at a product with H^*

$$\sum_{x'=-\infty}^{\infty} g[x+x']h^*[x'] \quad \Leftrightarrow \quad G(k)H^*(k)$$

(The Wiener filter does a form of matching)

Template matching

In summary: compute the correlation and find its maxima.