Basic Image Processing Algorithms

PPKE-ITK

Lecture 5-6.

Image Recovery

Image Recovery

What is Image Recovery?

Recovery vs Enhancement

- (Recap) **Image enhancement** is the manipulation or transformation of the image to **improve the visual appearance** or to **help further** automatic **processing steps**.
  - We don’t add new information to the image, just make it more visible (e.g. increasing contrast) or highlight a part of it (e.g. dynamic range slicing).

- **Recovery:** the **modeling** and **removal** of the degradation the image is subjected to, based on some **optimality criteria**.
  - In case of recovery there is a ***degradation we want to remove***, lost ***information we want to recover*** (e.g. make blurred text readable again). It is done by modeling the degradation and making assumptions about the degradation and the original image.

Image Recovery Examples

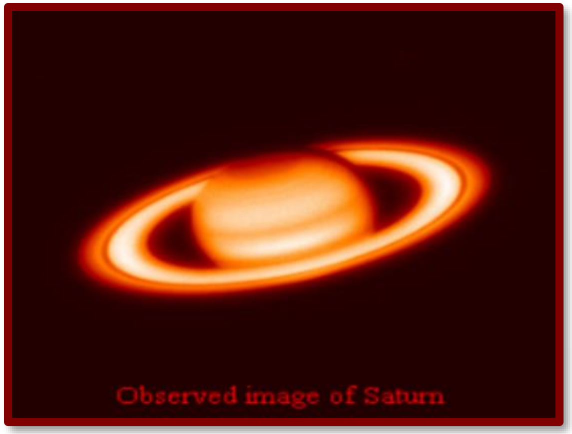
Blurred Image

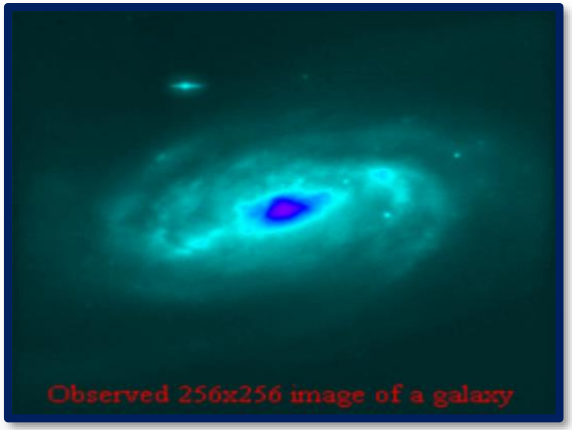
Restored Image

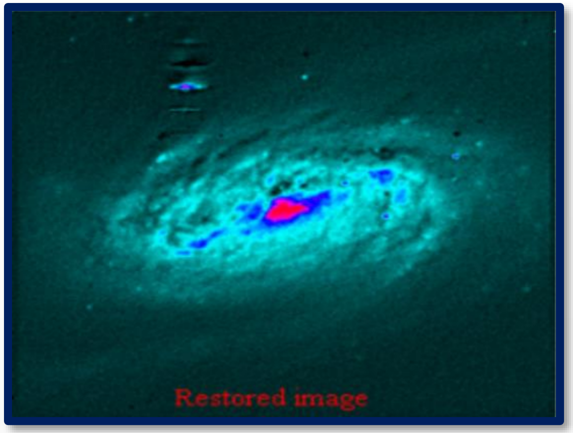
Original Image

Image Recovery Examples

- Images from the Hubble Space Telescope, taken with a defective mirror.







Source of the images: Fundamentals of Digital Image and Video Processing lectures by Aggelos K. Katsaggelos

Image Recovery Examples

- Blind restoration of image corrupted by motion blur:

Original Image with Motion Blur

Restored Image

Zhaofu Chen; Derin Babacan, S.; Molina, R.; **Katsaggelos**, AK., "Variational Bayesian Methods For Multimedia Problems," *Multimedia, IEEE Transactions on* , vol.16, no.4, pp.1000,1017, June 2014.

Image Recovery Examples

- Super-resolution:
  - The process of combining multiple low resolution images to form a high resolution image.

Single, non-enhanced frame

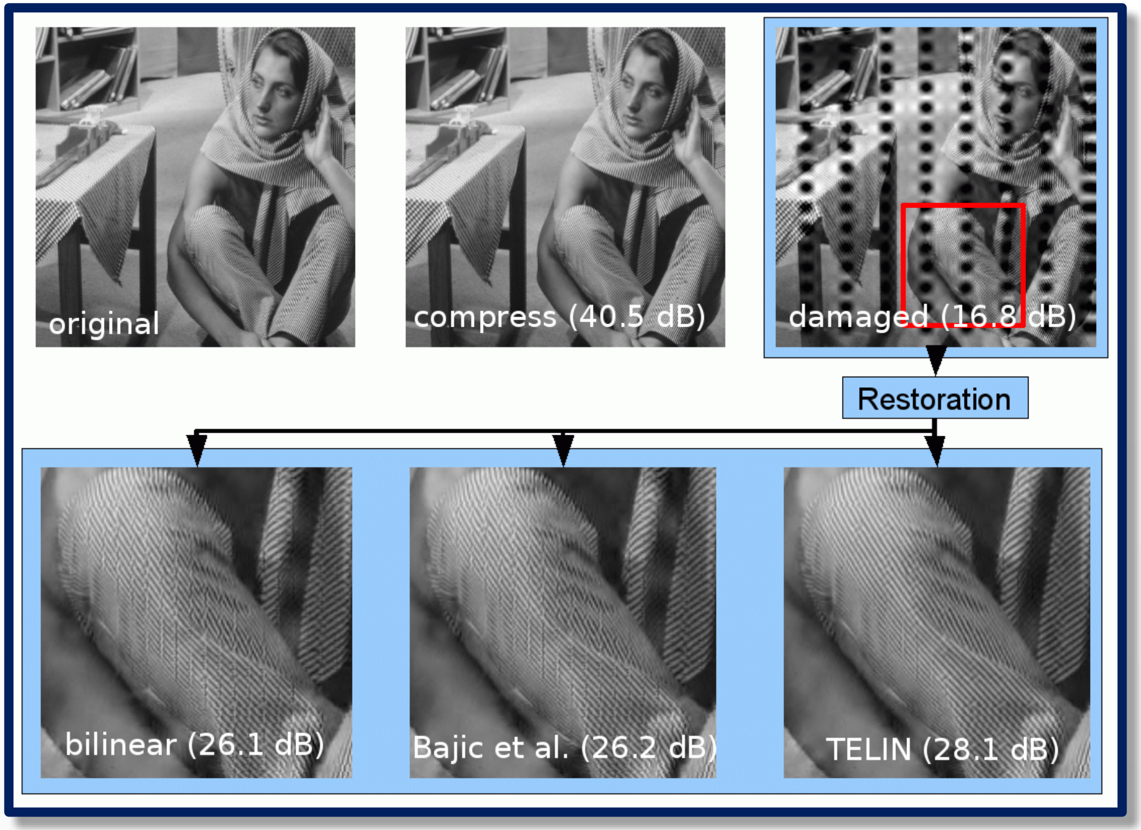
Multiple Frames of the same scene

Reconstructed Frame

Source of the Image: http://www.motiondsp.com/products/ikena/super-resolution

Image Recovery Examples

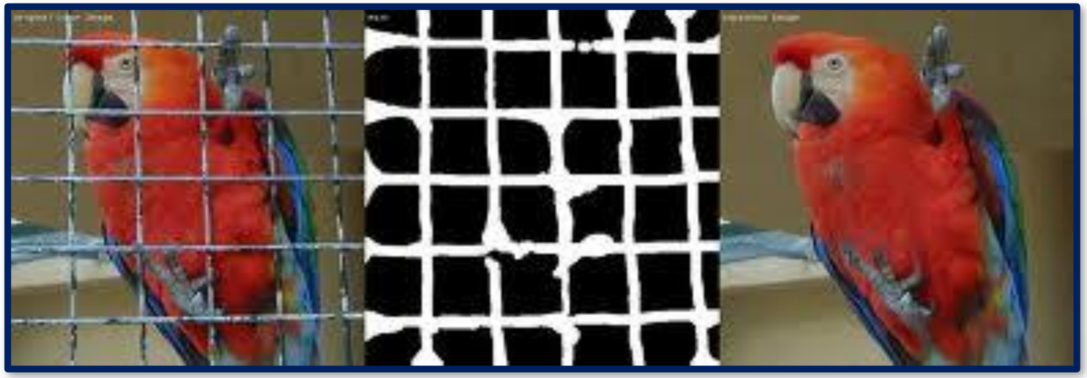
- Error Concealment:



J. Rombaut, A. Pizurica, and W. Philips, "Locally adaptive passive error concealment for wavelet coded images," IEEE Signal Processing Letters, vol. 15, pp. 178-181, 2008.

Image Recovery Examples

- Inpainting



http://www.mathworks.com/company/newsletters/articles/applying-modern-pde-techniques-to-digital-image-restoration.html

http://nbviewer.ipython.org/github/chintak/inpainting-demo/blob/master/Hello_ShopSense.ipynb

Image Recovery Examples

- Deblocking:
  - removal of blocking artifacts introduced by compression

S. Alireza Golestaneh, D. M. Chandler, "An Algorithm for JPEG Artifact Reduction via Local Edge Regeneration" *Journal of Electronic Imaging (JEI),* Jan 2014

Sources of Degradation and Forms of Recovery

Sources of Degradation

1. Motion
2. Atmospheric turbulence
3. Out-of-focus lens
4. Finite resolution of the sensors
5. Limitations of the acquisition system
6. Transmission error
7. Quantization error
8. Noise

Forms of Restoration

1. Restoration/Deconvolution
2. Removal of Compression Artifacts
3. Super-Resolution
4. Inpainting/Concealment
5. Noise smoothing

Inverse problem formulation of Recovery

- The original image *x* goes through a system (*H*), that introduces some type of degradation resulting the observed image *y*:

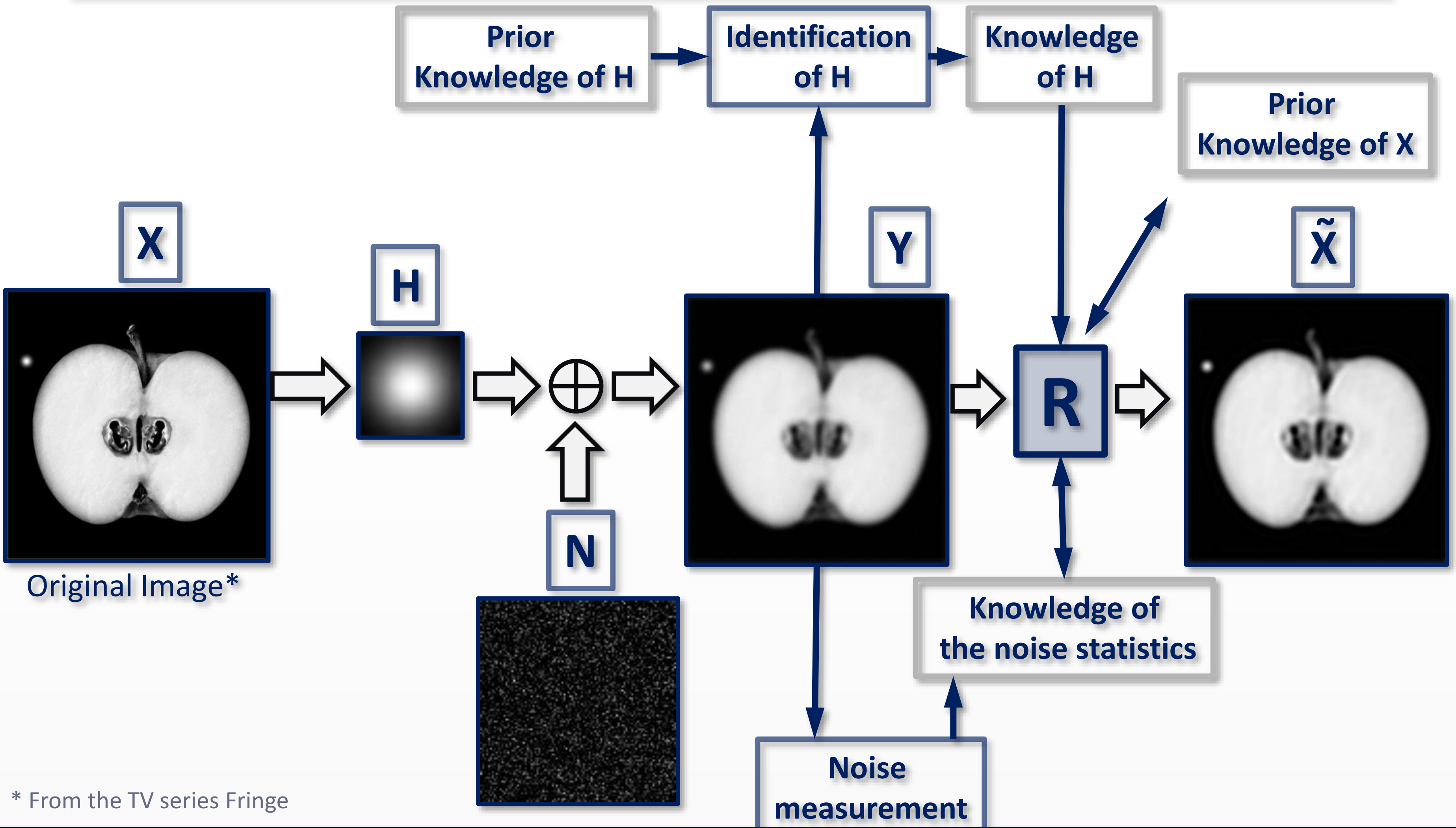
$$x(n_1, n_2) \rightarrow H \rightarrow y(n_1, n_2)$$

- The objective is to reconstruct *x* based on…
  - *y* and *H* → recovery
  - *y* → blind recovery
  - *y* and partially *H* → semi-blind recovery

Inverse problems

- If we know *x* and
  - *y* → system identification
  - *H* → system implementation

Degradation and Restoration



* From the TV series Fringe

Degradation Model

- The model of degradation for **restoration** problems:

$$y(n_1, n_2) = H\left[x(n_1, n_2)\right] + n(n_1, n_2)$$

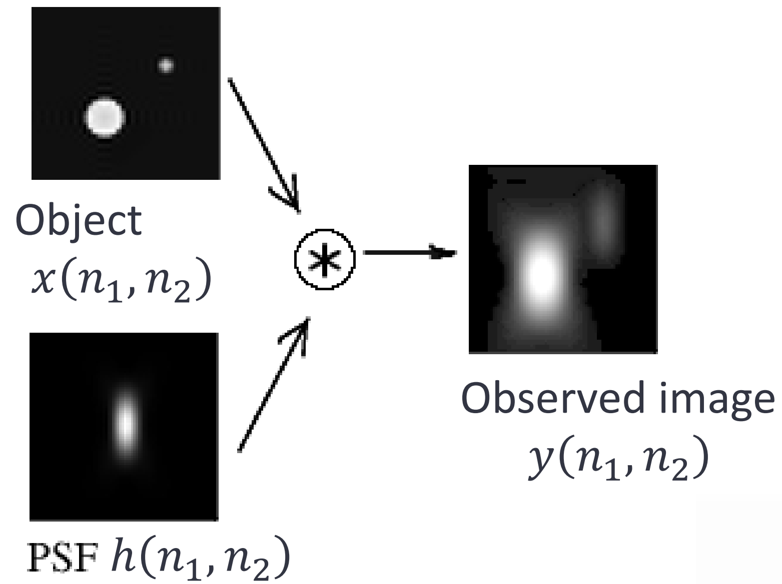
- If an LSI degradation system is assumed, with signal independent additive noise:

$$y(n_1, n_2) = x(n_1, n_2) * h(n_1, n_2) + n(n_1, n_2)$$

- The restoration problem in this case is called **deconvolution**.

Point spread function (PSF) of an imaging system

- **PSF:** system's impulse response:
  - „An image" $h(n_1, n_2)$, which describes the response of an imaging system to a point source or point object
  - the degree of spreading (blurring) of the point object is a measure for the quality of an imaging system.
  - observed image can be taken as the convolution of the object and the PSF



- **Optical transfer function:**
  - the Fourier transform of the PSF.
  - convolution in the spatial (PSF) domain becomes simple element wise multiplication in the Fourier domain

$$y(n_1, n_2) = x(n_1, n_2) * h(n_1, n_2)$$

$$Y(\omega_1, \omega_2) = X(\omega_1, \omega_2) \cdot H(\omega_1, \omega_2)$$

Convolution in matrix-vector form

- 1D convolution can be represented in a matrix-vector form:

$$y(n) = x(n) * h(n) = \sum_k x(k)h(n-k)\,, \text{ where } \quad \begin{array}{l} x : 1 \times N \\ h : 1 \times L \\ y : 1 \times (N + L - 1) \end{array}$$

$$\begin{bmatrix} y(0) \\ y(1) \\ \\ \vdots \\ \\ y(N+L-2) \end{bmatrix} = \begin{bmatrix} h(0) & 0 & \cdots & \cdots & 0 \\ h(1) & h(0) & 0 & \cdots & 0 \\ h(2) & h(1) & h(0) & \cdots & 0 \\ \vdots & \vdots & & \ddots & \vdots \\ h(L-1) & \vdots & & & h(0) \\ 0 & h(L-1) & & & \vdots \\ \vdots & \vdots & & & h(L-1) \end{bmatrix} \cdot \begin{bmatrix} x(0) \\ x(1) \\ \\ \vdots \\ \\ x(N-1) \end{bmatrix}$$

$$\boldsymbol{y} = \boldsymbol{H} \cdot \boldsymbol{x}$$

H: ***block circulant*** **matrix.**

LSI Image filtering

- The $N_1 \times N_2$ images involved must be lexicographically ordered. That means that an image is converted to a column vector by pasting the rows one by one after converting them to columns.
  - An image of size 256×256 is converted to a column vector of size 65536×1.
- An LSI degradation model can be written in a matrix form, where the images are vectors and the degradation process is a **huge but sparse block circulant** matrix:

$$\mathrm{y} = \mathrm{H}\mathrm{x}$$

- The above relationship is ideal. What really happens is

$$\mathrm{y} = \mathrm{H}\mathrm{x} + \mathrm{n}$$

where: n is a noise component

Convolution in matrix-vector form

- Matrix-vector form of degradation:

$$\mathrm{y} = \mathrm{Hx} + \mathrm{n}$$

  - x, n and y are column vectors of size $N_1 N_2 \times 1$
  - if the system is LSI, then ***H*** is a ***block circulant* matrix.**
- Representation in the frequency domain:
  - for block circulant matrices it can be shown*, that we can take it to the Fourier domain elementwise, so the following equation holds:

$$\mathrm{Y}(\omega_1,\omega_2) = \mathrm{H}(\omega_1,\omega_2)\cdot \mathrm{X}(\omega_1,\omega_2) + \mathrm{N}(\omega_1,\omega_2),\ \text{ where } \begin{cases} \omega_1 = 0,\ldots,\mathrm{N}_1 - 1 \\ \omega_2 = 0,\ldots,\mathrm{N}_2 - 1 \end{cases}$$

Here all matrices have a size of $N_1 \times N_2$:

- $X(\omega_1, \omega_2)$: DFT of the $x(n_1, n_2)$ 2D input image (matrix!)
- H(ω_1, ω_2): DFT of the point spread function,
- N(ω_1, ω_2): DFT of the noise
- Y(ω_1, ω_2): DFT of the output

*proof available as supplementary material optional to read, not part of exam

Image Recovery

Deconvolution Algorithms

Inverse Filter

- Simplest deconvolution filter, developed for LSI systems.
- Can be easily implemented in the frequency domain as the inverse of the degradation filter.
- Main limitations and drawbacks:
  - Strong noise amplification
  - The degradation system has to be known a priori.
- The degradation equation: $\boxed{\boldsymbol{y} = \boldsymbol{H}\mathbf{x} + \boldsymbol{n}}$
  - In this problem we know $\boldsymbol{H}$ and $\boldsymbol{y}$ and we are looking for a descent **x**
  - The objective is to find $\boldsymbol{x}$ that minimizes the Euclidian norm of the error:

$$\underset{x}{\operatorname{argmin}}\left(J(x)\right) = \underset{x}{\operatorname{argmin}}\left(\left\|y - Hx\right\|^2\right)$$

Inverse Filter

- The problem is formulated as follows: we are looking to minimize the Euclidian norm of the error:

$$\underset{x}{\operatorname{argmin}}\left(J(x)\right) = \underset{x}{\operatorname{argmin}}\left(\|y - Hx\|^2\right)$$

- The first derivative of the minimization function must be set to zero.

$$\frac{\partial J(x)}{\partial x} = 0 \Rightarrow \frac{\partial}{\partial x}\left(y^T y - 2x^T H^T y + x^T H^T Hx\right)$$

$$= -2H^T y + 2H^T Hx = 0$$

$$H^T Hx = H^T y$$

Generalized Inverse

$$x = \left(H^T H\right)^+ H^T y$$

Inverse Filter

- We have that in Matrix-vector form (Mvf):

$$HH^T x = H^T y$$

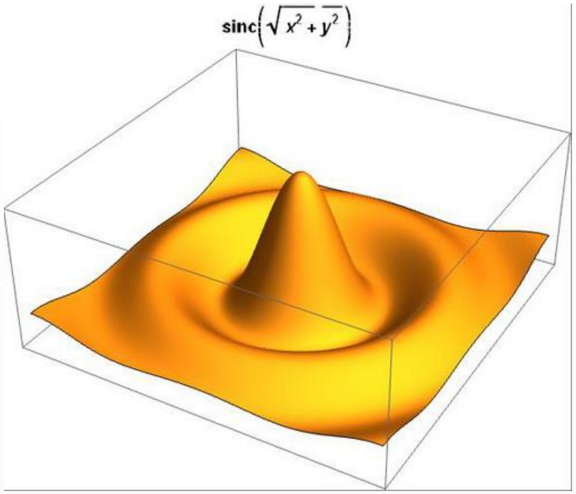
- Frequency domain representation: if we take the DFT of the above relationship in both sides we have:

$$|H(\omega_1,\omega_2)|^2 \cdot X(\omega_1,\omega_2) = H^*(\omega_1,\omega_2) \cdot Y(\omega_1,\omega_2)$$

  - *Recap:* connection between the Mvf and the DFT representation of LSI systems (slide 20)
  - *We do not prove here*: if the DFT of an LSI transform H is $H(\omega_1,\omega_2)$, then the DFT of H^T is $H^*(\omega_1,\omega_2)$ (complex conjugate)
  - Easy to prove: for any complex number $c \cdot c^* = |c|^2$:
    - $c \cdot c^* = (x + jy) \cdot (x - jy) = x^2 - jxy + jxy - jjy^2 = x^2 + y^2 = |c|^2$

Inverse Filter

- We have that: $$X(\omega_1,\omega_2)=\frac{H^*(\omega_1,\omega_2)Y(\omega_1,\omega_2)}{\left|H(\omega_1,\omega_2)\right|^2}$$
- **Problem:** It is very likely that $H(\omega_1,\omega_2)$ is 0 or very small at certain frequency pairs.
- For example, $H(\omega_1,\omega_2)$ could be a $sinc$ function.
- In general, since $H(\omega_1,\omega_2)$ is a low pass filter, it is very likely that its values drop off rapidly as the distance of (ω_1,ω_2) from the origin $(0,0)$ increases.



Slide credit: Tania Stathaki Imperial College London

Inverse filtering for noisy scenarios

- Simplification: consider a system where $H(\omega_1, \omega_2)$ is real*. In this case, the inverse filter output is calculated as:

$$\hat{X}_{inv}(\omega_1, \omega_2) = \frac{Y(\omega_1, \omega_2)}{H(\omega_1, \omega_2)}$$

- Assume, that in fact the degradation system output is affected by noise $N(\omega_1, \omega_2)$:

$$Y(\omega_1, \omega_2) = H(\omega_1, \omega_2) \cdot X(\omega_1, \omega_2) + N(\omega_1, \omega_2)$$

- In this case:

$$\underbrace{\mathrm{X}(\omega_1, \omega_2)}_{\text{Real signal}} = \frac{\mathrm{Y}(\omega_1, \omega_2) - \mathrm{N}(\omega_1, \omega_2)}{\mathrm{H}(\omega_1, \omega_2)} = \underbrace{\frac{\mathrm{Y}(\omega_1, \omega_2)}{\mathrm{H}(\omega_1, \omega_2)}}_{\hat{X}_{inv}(\omega_1, \omega_2)} - \underbrace{\frac{\mathrm{N}(\omega_1, \omega_2)}{\mathrm{H}(\omega_1, \omega_2)}}_{\text{Reconstruction error}}$$

*holds for central symmetric PSF

Inverse filtering for noisy scenarios

- Filter output is affected by noise $N(\omega_1, \omega_2)$:

$$X(\omega_1,\omega_2) = \frac{Y(\omega_1,\omega_2) - N(\omega_1,\omega_2)}{H(\omega_1,\omega_2)} = \frac{Y(\omega_1,\omega_2)}{H(\omega_1,\omega_2)} - \boxed{\frac{N(\omega_1,\omega_2)}{H(\omega_1,\omega_2)}}$$

- **Problem:** It is definite that while $H(\omega_1, \omega_2)$ is 0 or very small at certain frequency pairs, $N(\omega_1, \omega_2)$ is large.
- Note that $H(\omega_1, \omega_2)$ is a low pass filter, whereas $N(\omega_1, \omega_2)$ is an all pass function. Therefore, the term $\frac{N(\omega_1,\omega_2)}{H(\omega_1,\omega_2)}$ (error of estimation for $X(\omega_1, \omega_2)$) can be huge!
- The drawback of this method is the **strong amplification of** noise - Inverse filtering fails in that case ☹

Pseudo-inverse filtering

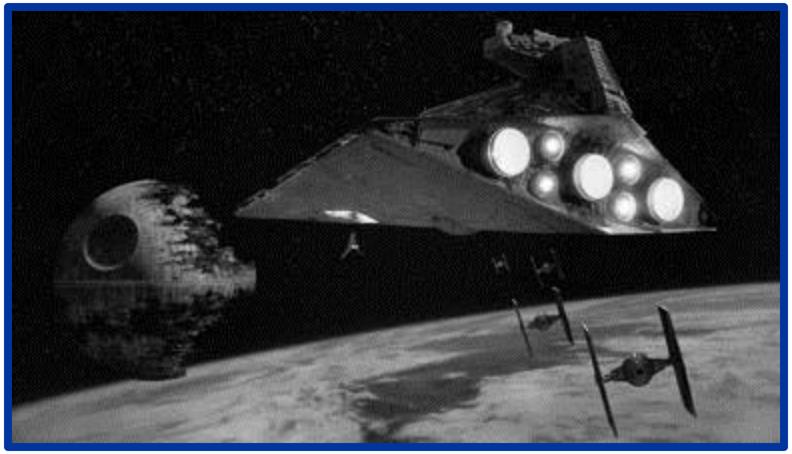
- Instead of the conventional inverse filter, we implement the following:

$$X(\omega_1,\omega_2)=\begin{cases}\dfrac{H^*(\omega_1,\omega_2)Y(\omega_1,\omega_2)}{\left|H(\omega_1,\omega_2)\right|^2} & \text{if } \left|H(\omega_1,\omega_2)\right|\geq T\\ T & \text{if } \left|H(\omega_1,\omega_2)\right|< T\end{cases}$$

- The parameter T (called **threshold** in the figures in the next slides) is a small number chosen by the user.
- This filter is called **pseudo-inverse** or **generalized** inverse filter.

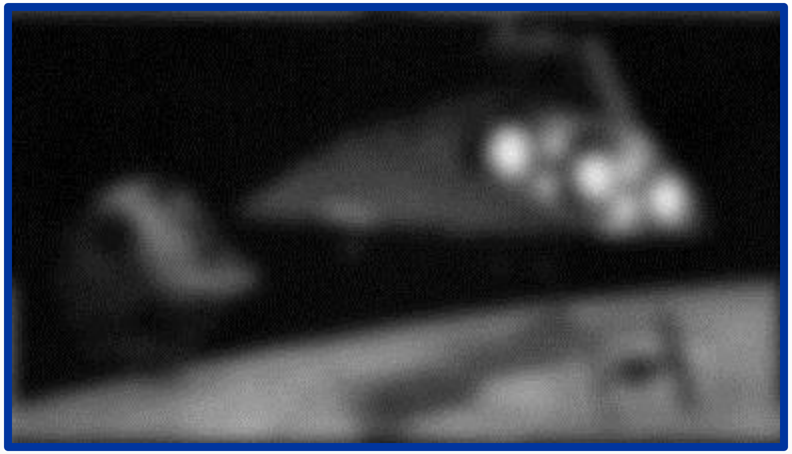
Pseudo-inverse filtering with different thresholds

Inverse Filter

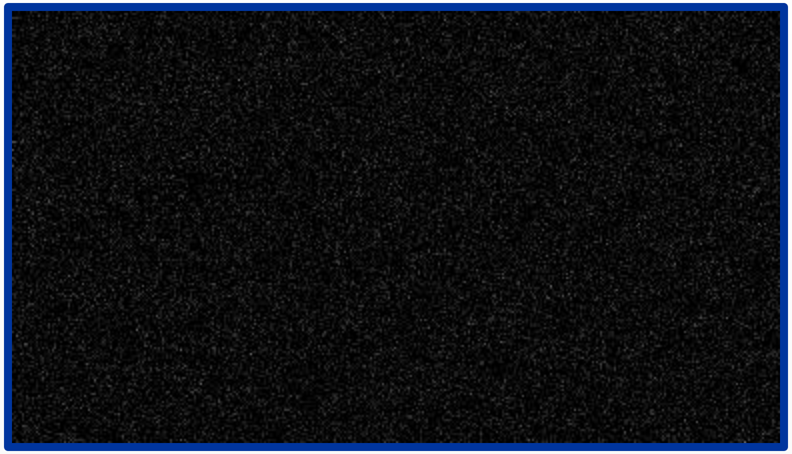


Original Image

Blurred Image

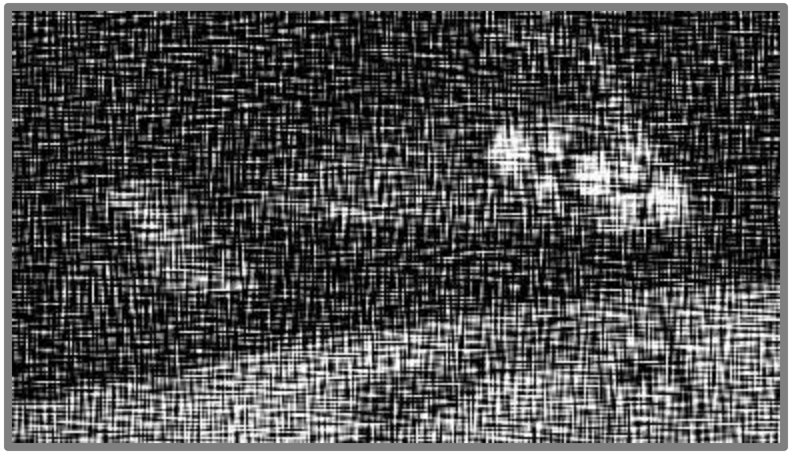


Blurred Image with Additional Noise

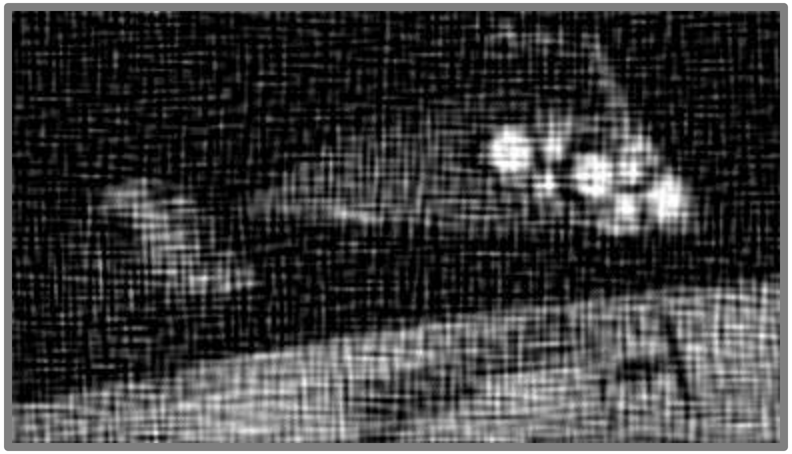


The Noise Component (amplified 10 times)

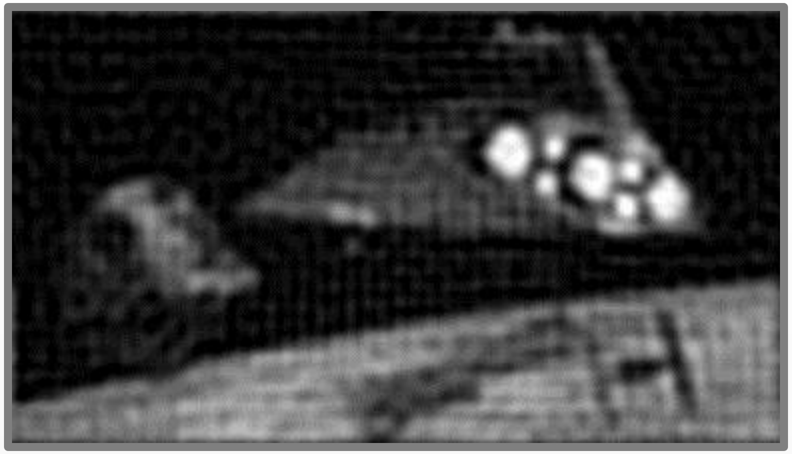
Inverse Filter



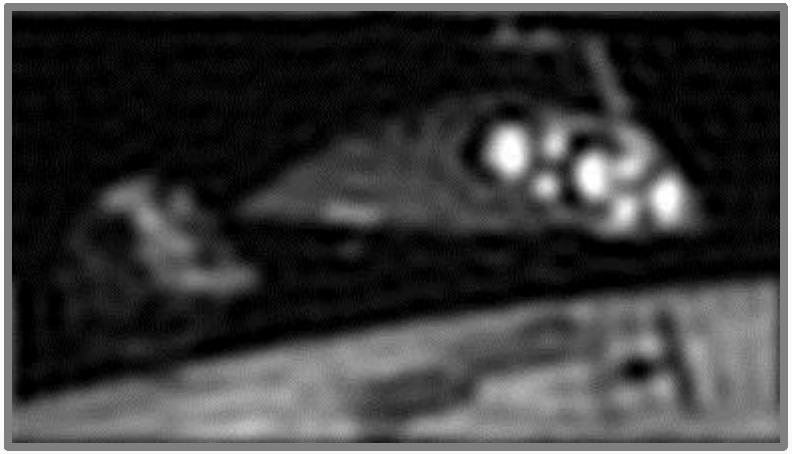
Reconstructed Image (with *T=0.005*)



Reconstructed Image (with *T=0.01*)

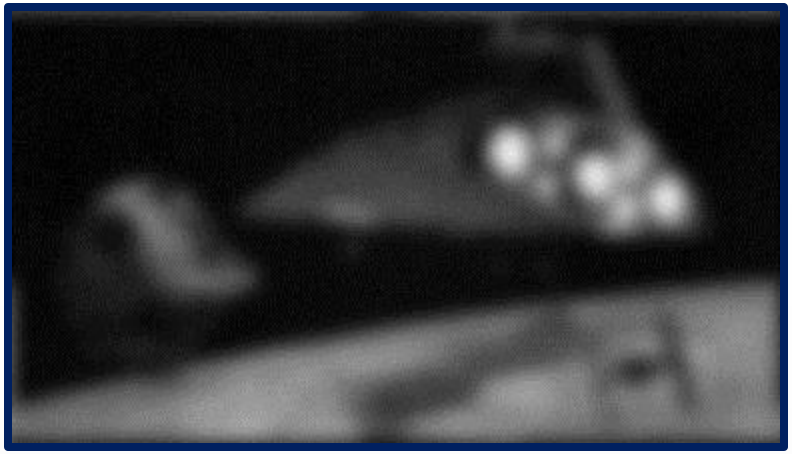


Reconstructed Image (with *T=0.02*)

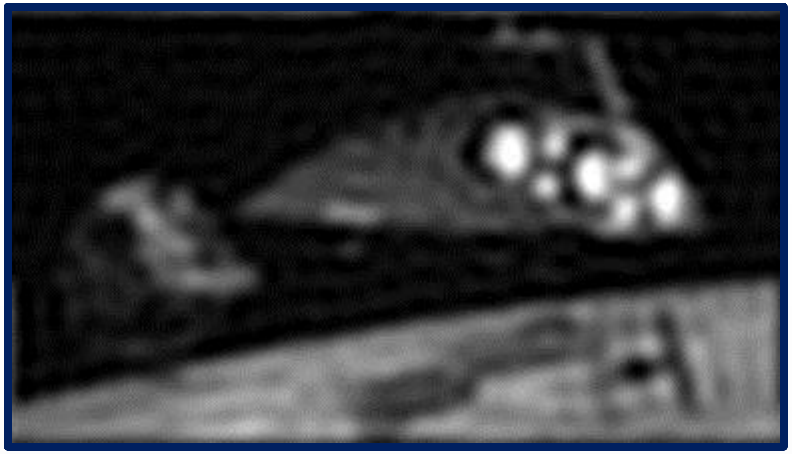


Reconstructed Image (with *T=0.1*)

Inverse Filter



Blurred Image with Additional Noise



Reconstructed Image (with *T=0.1*)

Matlab code of an inverse filter

```
T=.2;
x=double(imread('lena.bmp')); %read original image
N1=size(x,1); N2=size(x,2);
figure(1); imagesc(x); colormap(gray); %display original image
w=5; h=ones(w,w)/w^2; % PSF of bluring
X=fft2(x); % DFT of original image
H=fft2(h,N1,N2); % DFT of PSF
Y=X.*H; % DFT of blurred image
y=ifft2(Y)+10*randn(N1,N2); %observed image: blurred + additive
noise
Y=fft2(y); % DFT of the observed image
figure(2); imagesc(abs(ifft2(Y))); colormap(gray); %display
observed image
BF=find(abs(H)<T);
H(BF)=T;
invH=ones(N1,N2)./H;
X1=Y.*invH;
im=abs(ifft2(X1)); % reconstructed image
figure(3); imagesc(im); colormap(gray) %display result
```

Constrained Least Square Methods

- The objective is to reduce the noise amplification effect of the inverse filter by adding extra constraints about the restored image:
  - On one hand we still have the term describing the solution's fidelity to the data:

$$\underset{x}{\operatorname{argmin}}\left(J(x)\right) = \underset{x}{\operatorname{argmin}}\left(\left\|y - Hx\right\|^{2}\right)$$

  - But we also have a second term, incorporating some ***prior knowledge*** about the smoothness of the original image:

$$\left\|Cx\right\|_{2}^{2} < \varepsilon$$

  - Putting the two together with the introduction of α:

$$\underset{x}{\operatorname{argmin}}\left(\left\|y - Hx\right\|_{2}^{2} + \alpha\left\|Cx\right\|_{2}^{2}\right)$$

Constrained Least Square Methods

$$\underset{x}{\operatorname{argmin}}\left(\|y - Hx\|_2^2 + \alpha\|Cx\|_2^2\right) \Rightarrow \quad x = \left(H^T H + \alpha C^T C\right)^+ H^T y$$

- *C* is a high pass filter
  - Intuitively this means that we want to keep under control the amount of energy contained in the high frequencies on the restored image.
- α is the regularization parameter
- In the frequency domain (for H and C block circulant) we have the following formula:

$$X(\omega_1, \omega_2) = \frac{H^*(\omega_1, \omega_2)}{|H(\omega_1, \omega_2)|^2 + \alpha|C(\omega_1, \omega_2)|^2} Y(\omega_1, \omega_2)$$

  - if α is 0, we get back the simple Least Square method (Inverse Filter)

Constrained Least Square Methods

- (Optional) Different types of regularization:
  - CLS:

$$\tilde{x}(\alpha)_{CLS} = \underset{x}{\operatorname{argmin}} \left(\|y - Hx\|_2^2 + \alpha \|Cx\|_2^2 \right)$$

  - Maximum Entropy Regularization:

$$\tilde{x}(\alpha)_{ME} = \underset{x}{\operatorname{argmin}} \left(\|y - Hx\|_2^2 + \alpha \sum_{i=1}^{N} x_i \log(x_i) \right)$$

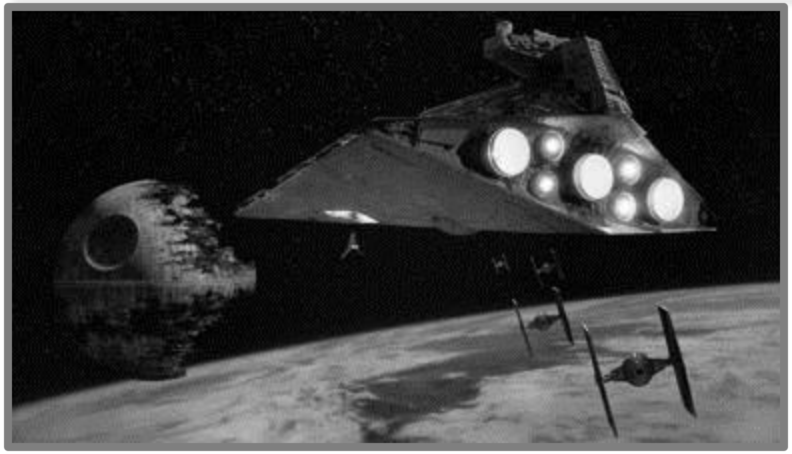
  - Total Variation Regularization:

$$\tilde{x}(\alpha)_{TV} = \underset{x}{\operatorname{argmin}} \left(\|y - Hx\|_2^2 + \alpha \sum_{i=1}^{N} \left| [\Delta x]_i \right| \right)$$

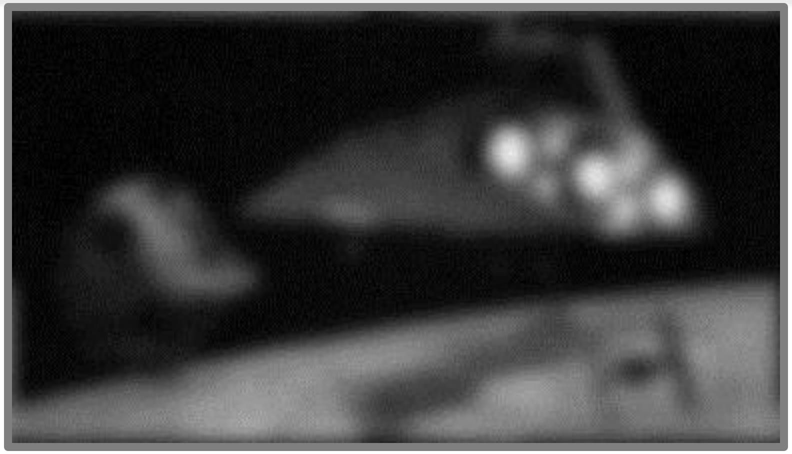
  - l_p – norms:

$$J(z) = \|z\|_p^p = \sum_{i=1}^{N} |z_i|^p, \qquad 1 \le p \le 2$$

Constrained Least Square Methods

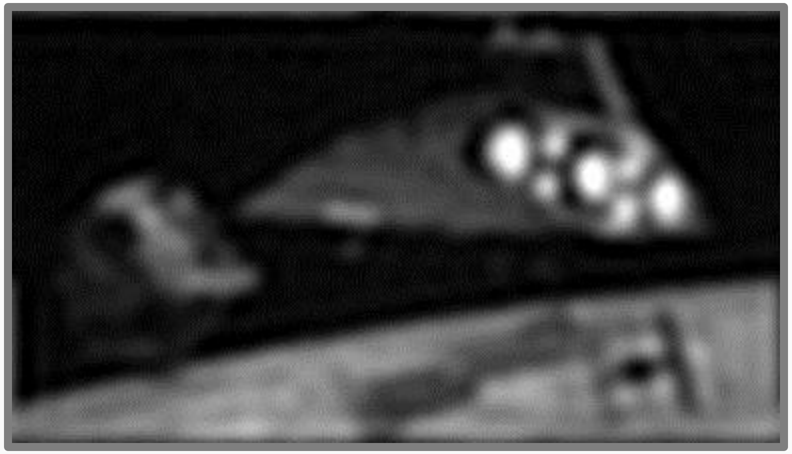


Original Image



Blurred Image with Additional Noise

Reconstructed Image (CLS with α=0.1)

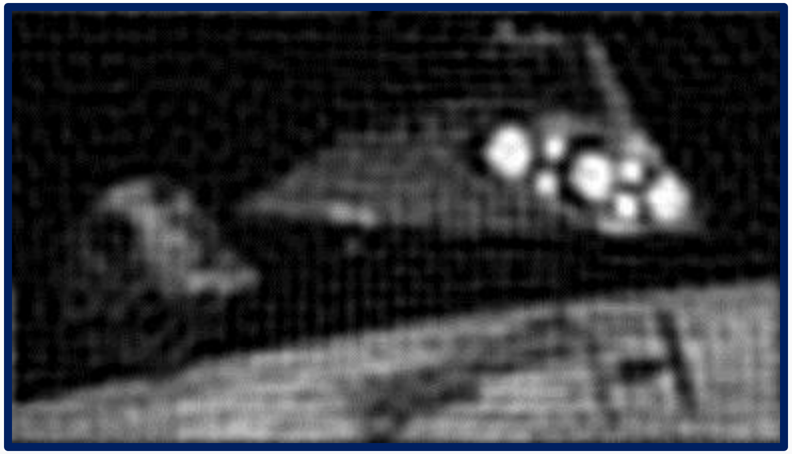


Reconstructed Image (CLS with α=0.5)

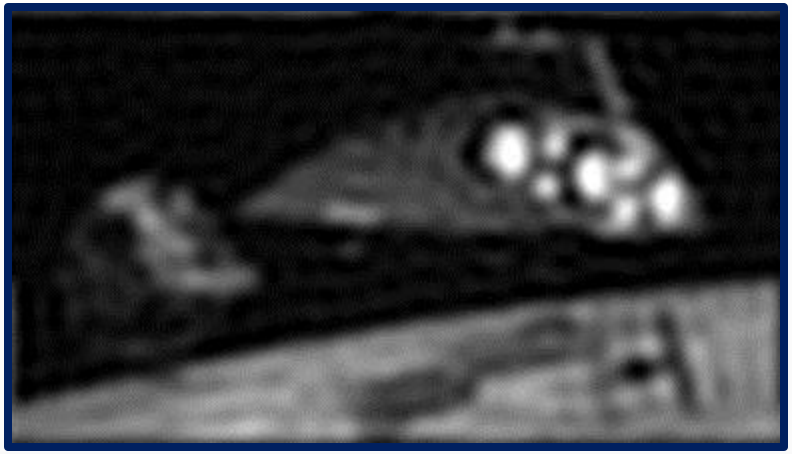
CLS vs. LS

Reconstructed Image (CLS with α=0.1)

Reconstructed Image (CLS with α=0.5)



Reconstructed Image (LS with *T=0.02*)



Reconstructed Image (LS with *T=0.1*)

Wiener Filter

- Stochastic restoration approach:
  - Treat the image as a sample from a 2D random field.
  - The image is part of a class of samples (an ensemble), realizations of the same random field.

$$\boxed{R(\omega_1,\omega_2) = \frac{H^*(\omega_1,\omega_2)\cdot P_{xx}(\omega_1,\omega_2)}{\left|H(\omega_1,\omega_2)\right|^2\cdot P_{xx}(\omega_1,\omega_2)+P_{NN}(\omega_1,\omega_2)}}$$

Expected value over many realizations of the 2D random field

  - Autocorrelation:

$$R_{ff}(n_1,n_2,n_3,n_4) = E\left[f(n_1,n_2)f^*(n_3,n_4)\right]$$

  - Power-Spectrum (**Wide Sense Stationarity (WSS)** input)

$$P_{ff}(\omega_1,\omega_2) = \mathrm{DFT}\left\{R_{ff}(d_1,d_2)\right\}\Big|_{d_1=n_1-n_3,\,d_2=n_2-n_4}$$

Fourier transformation

Autocorrelation calculation - some details

Expected value over many realizations of the 2D random field

- Definition of autocorrelation:

$$R_{ff}(n_1, n_2, n_3, n_4) = E\left[f(n_1, n_2)\, f^*(n_3, n_4)\right]$$

- Wide Sense Stationarity property (WSS):

$$R_{ff}(n_1, n_2, n_3, n_4) = R_{ff}(n_1 - n_3, n_2 - n_4) = R_{ff}(d_1, d_2)$$

- Ergodicity:
    - ensemble average is equal to spatial average

$$R_{ff}(d_1, d_2) = \lim_{N \to \infty} \frac{1}{(2N+1)^2} \sum_{k_1=-N}^{N} \sum_{k_2=-N}^{N} f(k_1, k_2)\, f^*(k_1 - d_1, k_2 - d_2)$$

Autocorrelation calculation the Fourier domain

- Recap (from textures lecture) Wiener-Khinchin Theorem
  - Input image: $x(n_1, n_2)$
  - Fourier transform: $\{X(\omega_1, \omega_2)\} = \mathrm{DFT}\{x(n_1, n_2)\}$
  - Power spectrum: $P_{xx}(\omega_1, \omega_2) = X(\omega_1, \omega_2) \cdot X^*(\omega_1, \omega_2)$
  - Autocorrelation: inverse Fourier transform of the power spectrum: $\{R_{xx}(d_1, d_2)\} = \mathrm{IDFT}\{P_{xx}(\omega_1, \omega_2)\}$

Wiener Filter

- The degradation model: $y(n_1,n_2) = x(n_1,n_2) * h(n_1,n_2) + n(n_1,n_2)$
- The objective:

$$\tilde{x}(n_1,n_2) = \underset{\tilde{x}(n_1,n_2)}{\operatorname{argmin}} \; E\left[\left|x(n_1,n_2) - \tilde{x}(n_1,n_2)\right|^2\right]$$

- We look for the solution in the following format, assuming an LSI restoration model:

$$\tilde{x}(n_1,n_2) = r(n_1,n_2) * y(n_1,n_2)$$

  - The input image is assumed to be WSS with autocorrelation $R_{xx}(n_1,n_2)$.
- The recovered image is obtained in the Fourier domain:

$$\boxed{X(\omega_1,\omega_2) = R(\omega_1,\omega_2) \cdot Y(\omega_1,\omega_2)}$$

Recovered image — Wiener filter — Observed degraded image

Wiener Filter

- Assumptions:
  - that both the input image and the noise are WSS.
  - The restoration error and the signal (the observed image) is orthogonal:

$$E\left[e(n_1,n_2)y^*(n_3,n_4)\right] = E\left[(x(n_1,n_2) - \tilde{x}(n_1,n_2))y^*(n_3,n_4)\right] = 0, \quad \forall (n_1,n_2),(n_3,n_4)$$

  - It can be shown* that the following transfer function implements the above constraint:

$$R(\omega_1,\omega_2) = \frac{H^*(\omega_1,\omega_2)\cdot P_{xx}(\omega_1,\omega_2)}{\left|H(\omega_1,\omega_2)\right|^2\cdot P_{xx}(\omega_1,\omega_2) + P_{NN}(\omega_1,\omega_2)}$$

*proof available as supplementary material
optional to read, not part of exam

Wiener Filter and CLS Filter

- Wiener filter:

$$R(\omega_1,\omega_2) = \frac{H^*(\omega_1,\omega_2)\cdot P_{xx}(\omega_1,\omega_2)}{|H(\omega_1,\omega_2)|^2\cdot P_{xx}(\omega_1,\omega_2) + P_{NN}(\omega_1,\omega_2)} =$$

$$= \frac{H^*(\omega_1,\omega_2)}{|H(\omega_1,\omega_2)|^2 + \frac{P_{NN}(\omega_1,\omega_2)}{P_{xx}(\omega_1,\omega_2)}} = \frac{H^*(\omega_1,\omega_2)}{|H(\omega_1,\omega_2)|^2 + \frac{\sigma_N^2}{P_{xx}(\omega_1,\omega_2)}}$$

Assuming white noise

Noise to signal ratio

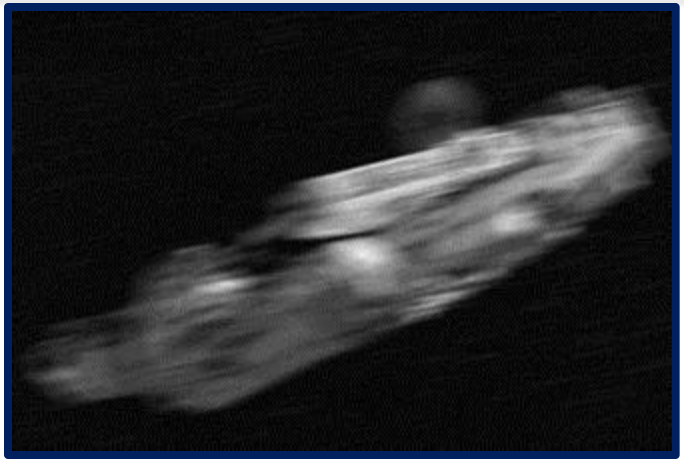
- CLS filter:

$$R(\omega_1,\omega_2) = \frac{H^*(\omega_1,\omega_2)}{|H(\omega_1,\omega_2)|^2 + \alpha|C(\omega_1,\omega_2)|^2}$$

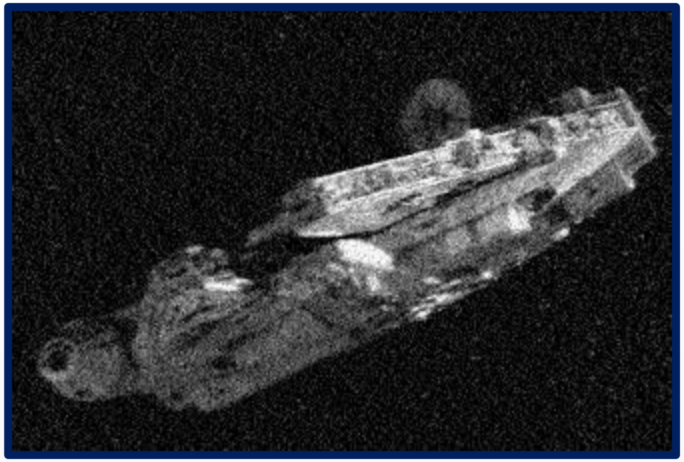
- With the right choice of *C* and *α*, CLS filter is the same as the Wiener filter.

Wiener Filter

Original Image



Motion Blurred Noisy Image

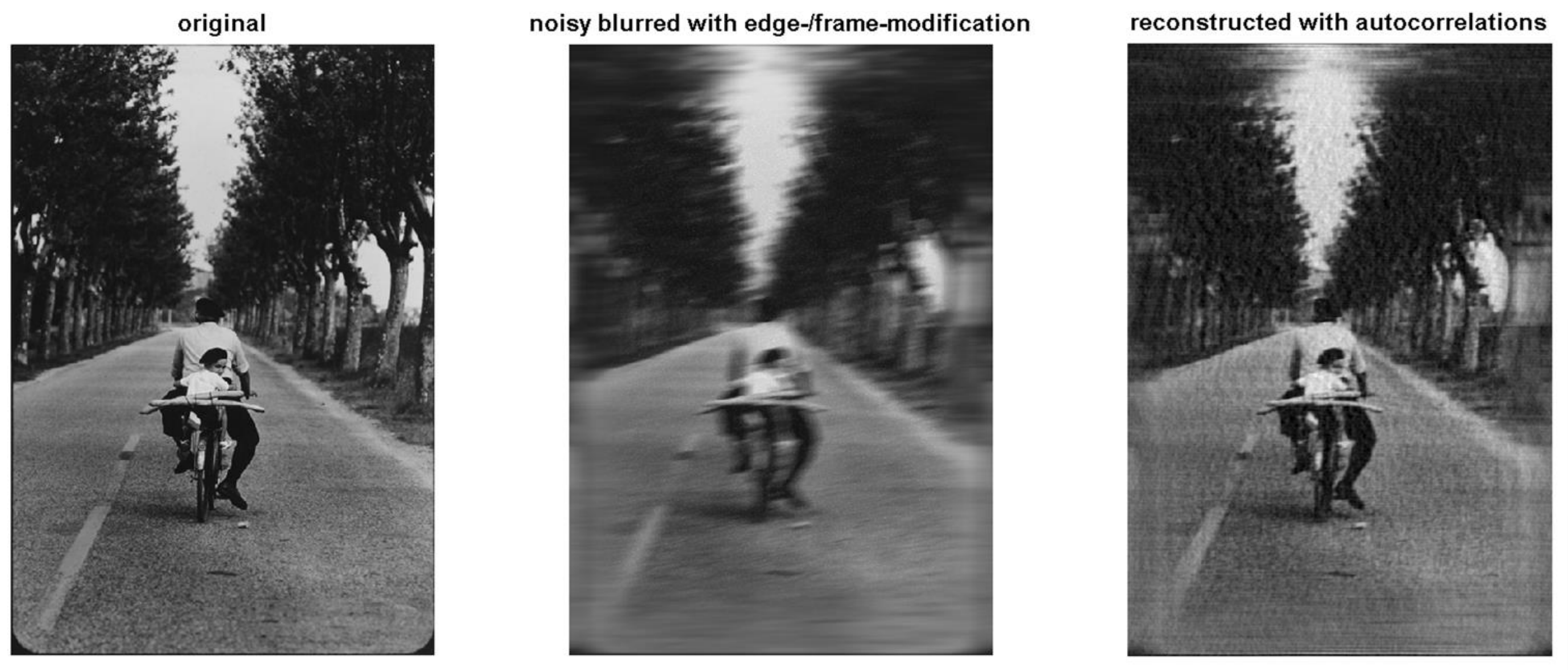


Reconstructed with Wiener filter

Reconstructed with CLS filter

Wiener Lab task on week 7

- See more practical details on the corresponding laboratory excercise!



Main Sources and Further Readings

- Fundamentals of Digital Image and Video Processing lectures by Aggelos K. Katsaggelos
- Babacan, D. S., R. Molina, and A. K. Katsaggelos, "Total Variation Super Resolution Using A Variational Approach", IEEE International Conf. on Image Processing 2008, San Diego, USA, 10/10/2008.
- J. Rombaut, A. Pizurica, and W. Philips, "Locally adaptive passive error concealment for wavelet coded images," IEEE Signal Processing Letters, vol. 15, pp. 178-181, 2008.
- Zhaofu Chen; Derin Babacan, S.; Molina, R.; Katsaggelos, AK., "Variational Bayesian Methods For Multimedia Problems," *Multimedia, IEEE Transactions on* , vol.16, no.4, pp.1000,1017, June 2014
- http://www.mathworks.com/company/newsletters/articles/applying-modern-pde-techniques-to-digital-image-restoration.html
- S. Alireza Golestaneh, D. M. Chandler, "An Algorithm for JPEG Artifact Reduction via Local Edge Regeneration" *Journal of Electronic Imaging (JEI),* Jan 2014