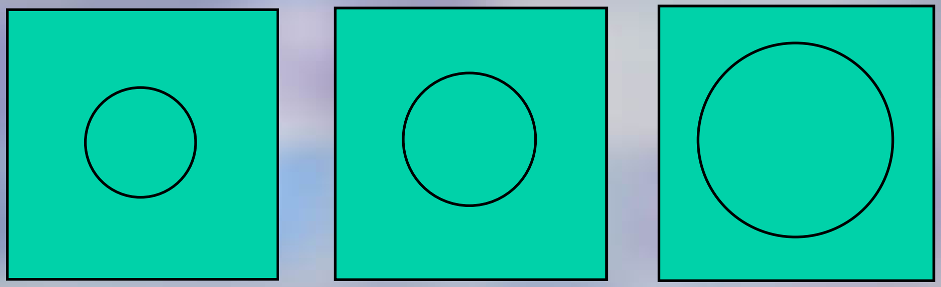
Linear Image Filtering

Monday 2 Oct 2006

- Wraparound and Linear Convolution
- Linear Image Filters
- Linear Image Denoising
- Linear Image Restoration (Deconvolution)

WRAPAROUND CONVOLUTION

- Modifying the DFT of an image changes its appearance. For example, multiplying a DFT by a zero-one mask predictably modifies image appearance:



Multiplying DFTs

- What if two arbitrary DFTs are (pointwise) multiplied together, or divided?
- $\widetilde{J}[u,v] = \widetilde{I}_1 \otimes \widetilde{I}_2$ $\qquad \widetilde{J}[u,v] = \widetilde{I}_1 \Delta \widetilde{I}_2$
- The answer has profound consequences in image processing.
- Division is a special case which need special handling if contains near-zero or zero values.

Multiplying DFTs

- Consider the pointwise product of two DFT's

$$\widetilde{J}[u,v] = \widetilde{I}_1 \otimes \widetilde{I}_2$$

- This has inverse DFT

$$J[i,j] = \frac{1}{N^2} \sum_{u,v=0}^{N-1} \widetilde{J}[u,v] W_N^{-(ui+vj)}$$

$$= \frac{1}{N^2} \sum_{u,v=0}^{N-1} \widetilde{I}_1[u,v] \cdot \widetilde{I}_2[u,v] W_N^{-(ui+vj)}$$

Inverse of product of DFTs:

$$= \frac{1}{N^2} \sum_{u,v=0}^{N-1} \{ \sum_{m,n=0}^{N-1} I_1[m,n] W_N^{um+vn} \}$$

$$\cdot \{ \sum_{p,q=0}^{N-1} I_2[p,q] W_N^{up+vq} \} W_N^{-(ui+vj)}$$

$$= \frac{1}{N^2} \sum_{m,n=0}^{N-1} I_1[m,n] \sum_{p,q=0}^{N-1} I_2[p,q] \sum_{u,v=0}^{N-1} W_N^{[u(p+m-i)+v(q+n-j)]}$$

- From the impulse function:

$$J[i,j] = \sum_{m,n=0}^{N-1} I_1[m,n] \sum_{p,q=0}^{N-1} I_2[p,q]\delta(p+m-i, q+n-k)$$

$$= \sum_{m,n=0}^{N-1} I_1[m,n] I_2[(i-m)_N, (j-n)_N)]$$

$$= \sum_{p,q=0}^{N-1} I_1[(i-p)_N, (j-q)_N)] I_2[p,q]$$

$$= I_1 \circledast I_2$$

Where we periodically extend $\mathbf{I}_1$ and $\mathbf{I}_2$

And $(x)_N$ means x mod N

Wraparound Convolution

- The summation

$$J[i,j] = \sum_{m,n=0}^{N-1} I_1[m,n] I_2[(i-m)_N, (j-n)_N]$$

- is also called cyclic convolution and circular convolution.
- Like linear convolution, it is an inner product between one sequence and a (doubly) reversed, shifted version of the other – except with indices taken modulo-M,N (M=N here for simplicity).

Wraparound Convolution

- It is a weighted sum of the elements $I_1(m, n)$ of the image $\mathbf{I}_1$, where the weights $I_2(i-m, j-n)$ are **shifted** elements of the image $\mathbf{I}_2$.

The amount of shift depends on (i, j).

(i, j) given, J(i, j), the new image, is defined by:

superimposing $\mathbf{I}_2$ directly "on top of" $\mathbf{I}_1$

reversing $\mathbf{I}_2$: $[I_2(-m, -n)]$

shifting $\mathbf{I}_2$ by an amount (i, j)

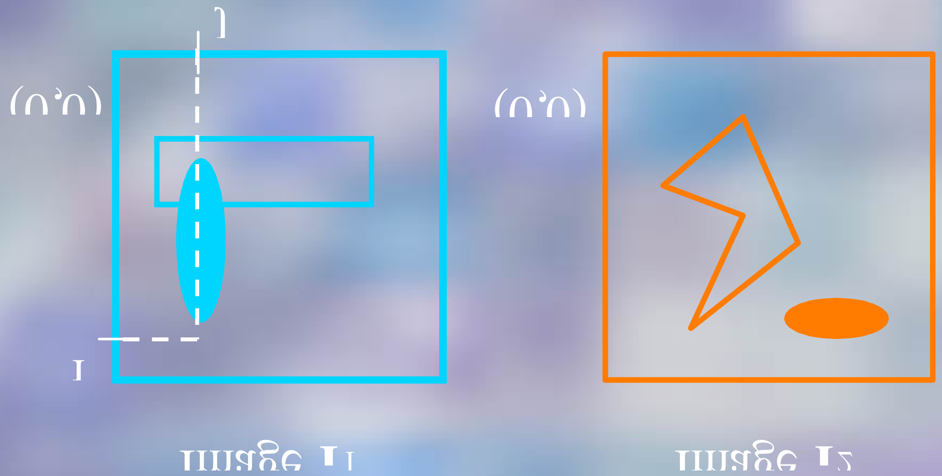
computing $I_1(m, n) \cdot I_2[(i-m)_N, (j-n)_N]$ for $0 \le m, n \le N-1$

adding the results

$$J = I_1 \circledast I_2 = IFFT_N[FFT_N[I_1] \otimes FFT_N[I_2]]$$

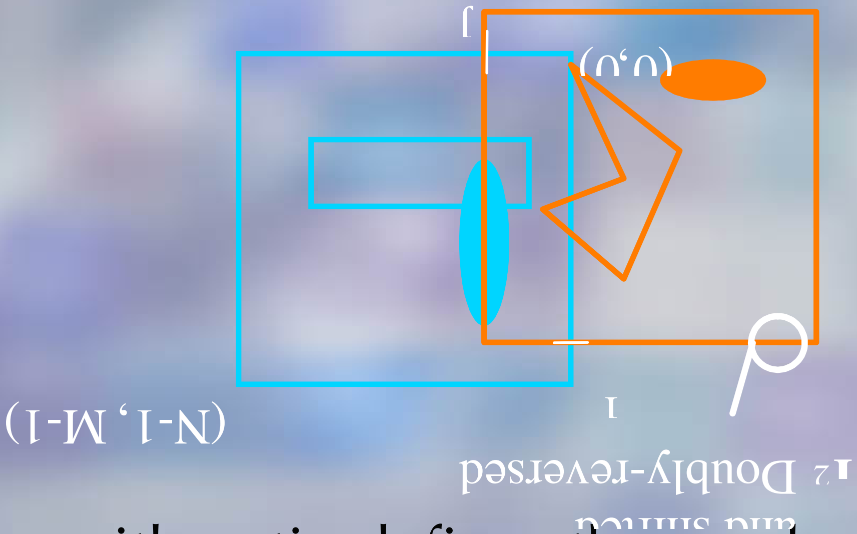
Depicting Wraparound Convolution

- Consider hypothetical images and $\mathbf{I}$ $\mathbf{I}$



- at which we wish to compute the cyclic convolution at (i, j) in the spatial domain (without DFTs).

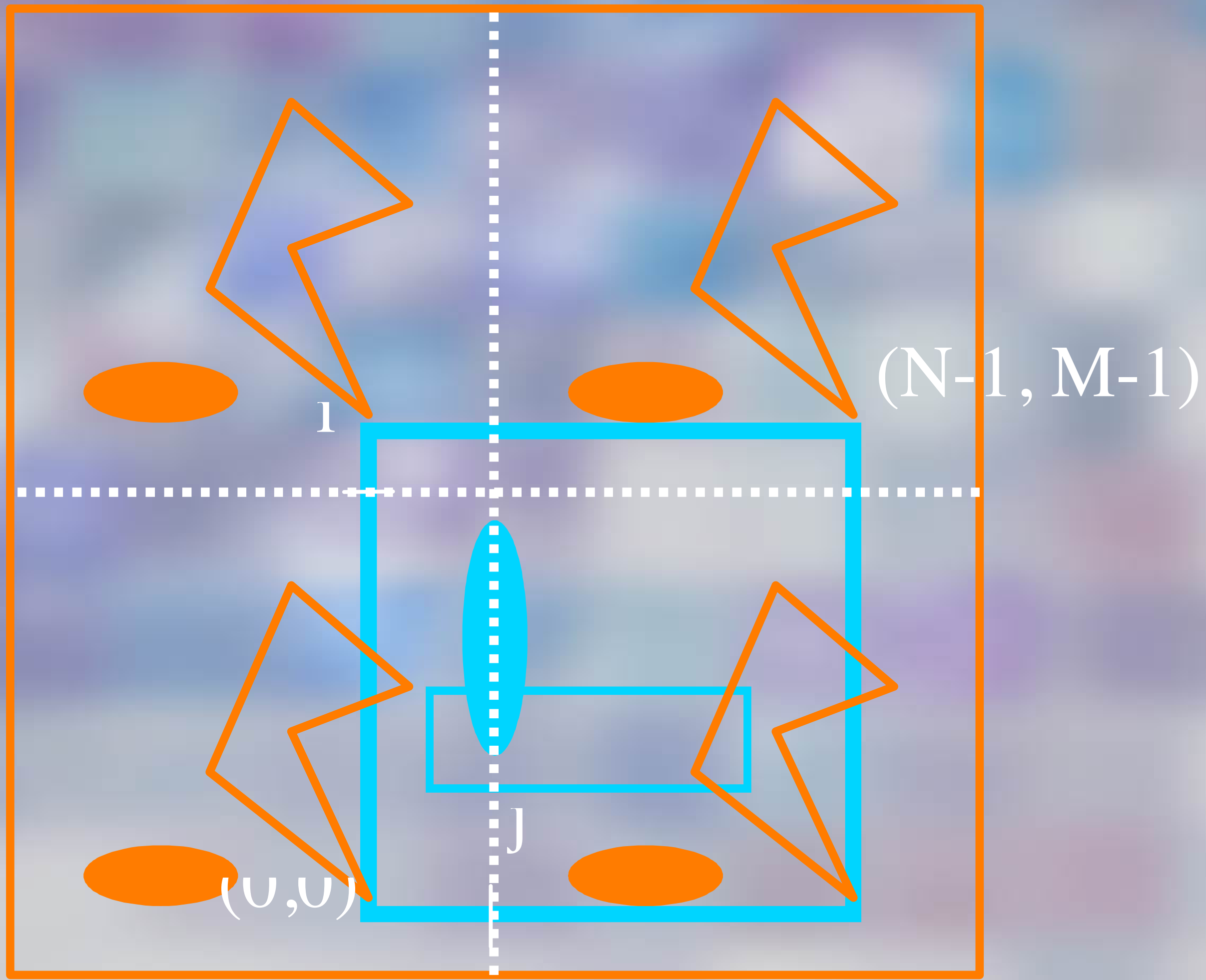
- Without wraparound:



- Modulo arithmetic defines the product for all 0 < i < N-1, 0 < j < M-1.

Summation occurs over $0 \leq i \leq N-1$, $0 \leq j \leq M-1$

Overlay of periodic extension of shifted f^2



Computation of Wraparound Convolution

- Direct computation of

$$J[i,j] = \sum_{m,n=0}^{N-1} I_1[m,n] I_2[(i-m)_N, (j-n)_N]$$

- is simple but expensive.
- For an NxM image:
- - for each of NM coordinates: NM additions and NM multiplies
- - or (NM)(NM) multiplies
- - for N=M=512, this 2^{36} = 6.9 x 10^{10} operations

DFT Computation of Wraparound Convolution

- Because of FFT, computing # in the DFT domain is much faster, provided that N = a power of 2.
- Simply

$$J = I_1 \circledast I_2 = IFFT_N[FFT_N[I_1] \otimes FFT_N[I_2]]$$

- Computing an (NxM) FFT is O[NM· log (NM)], so computation of # is as well.
- We now will discover that # must be modified in order to make it useful.

LINEAR CONVOLUTION

- Wraparound convolution is a consequence of the periodic DFT.
- For continuous images, if two CFTs are multiplied together:

$$\text{CFT}[J](w_x, w_y) = \text{CFT}[I_{C1}](w_x, w_y) \cdot \text{CFT}[I_{C2}](w_x, w_y)$$

- then we get a useful linear convolution:

$$J(x, y) = I_{C1}(x, y) * I_{C2}(x, y)$$

- Wraparound convolution is an artifact of sampling the CFT – which causes spatial periodicity.

About Linear Convolution

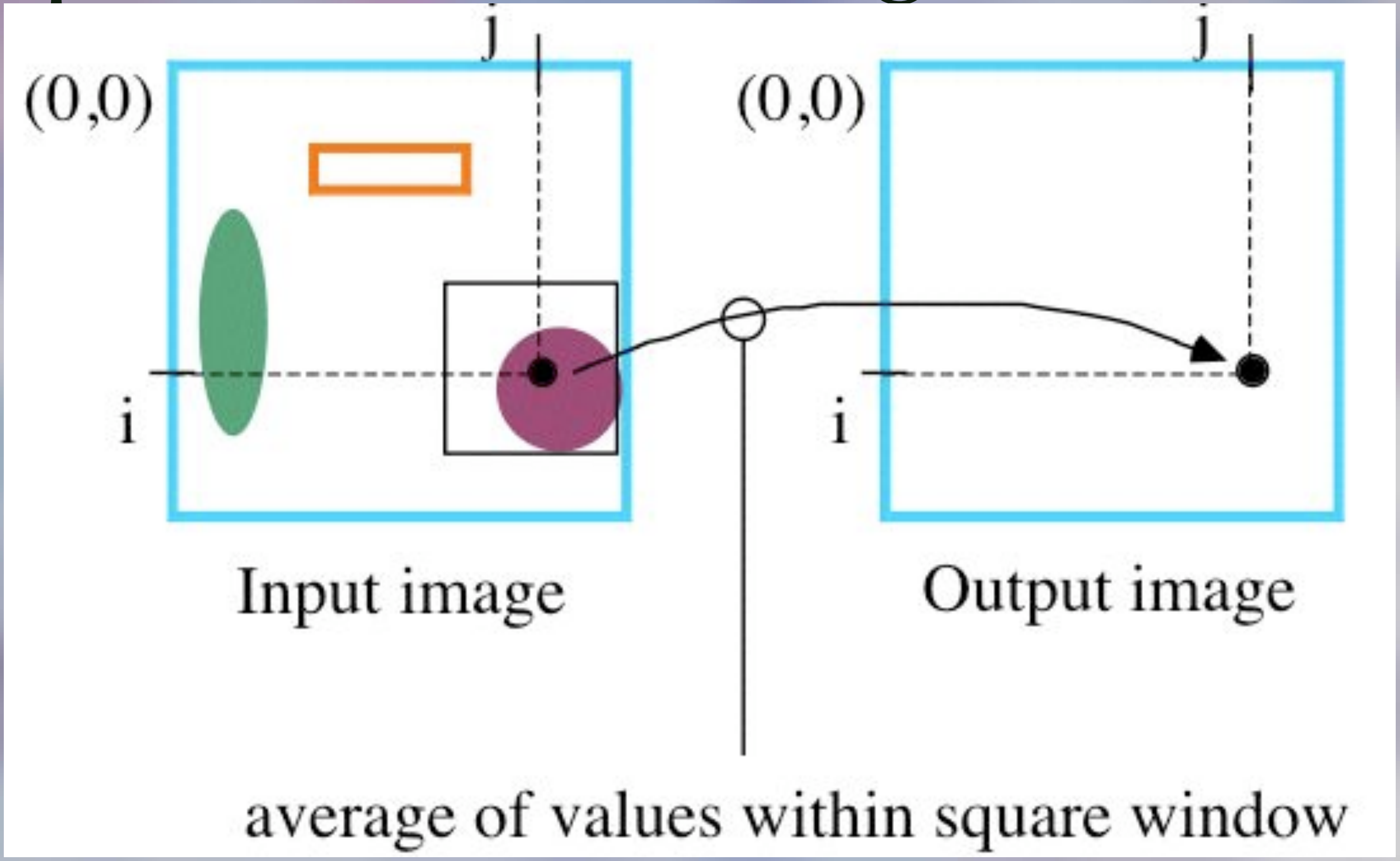
- Most of circuit theory, optics, and analog filter theory is based on linear convolution.
- And … (linear) digital filter theory also requires the concept of digital linear convolution.
- Fortunately, wraparound convolution can be used to compute linear convolution.

Undesirability of Wraparound

- A very simple type of linear convolutions is the local average operation (or averaging filter).

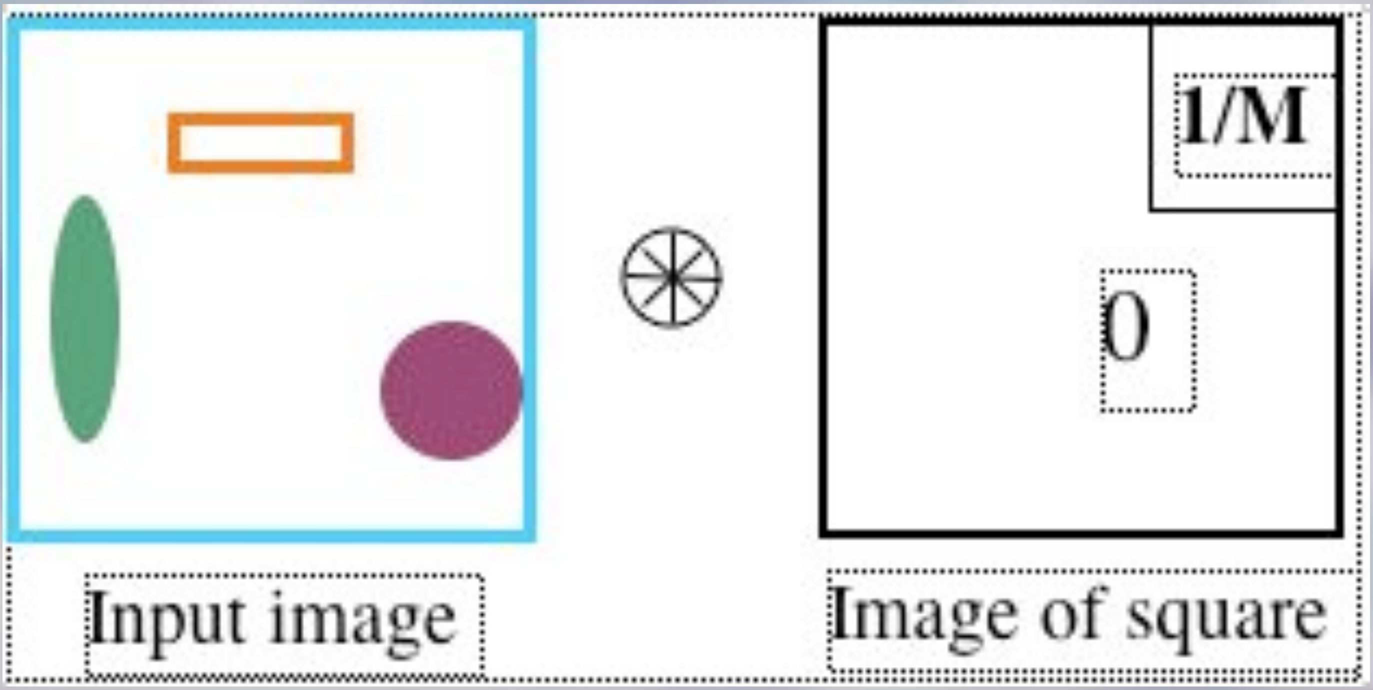
- Each image pixel is replaced by the average of its neighbors within a window:

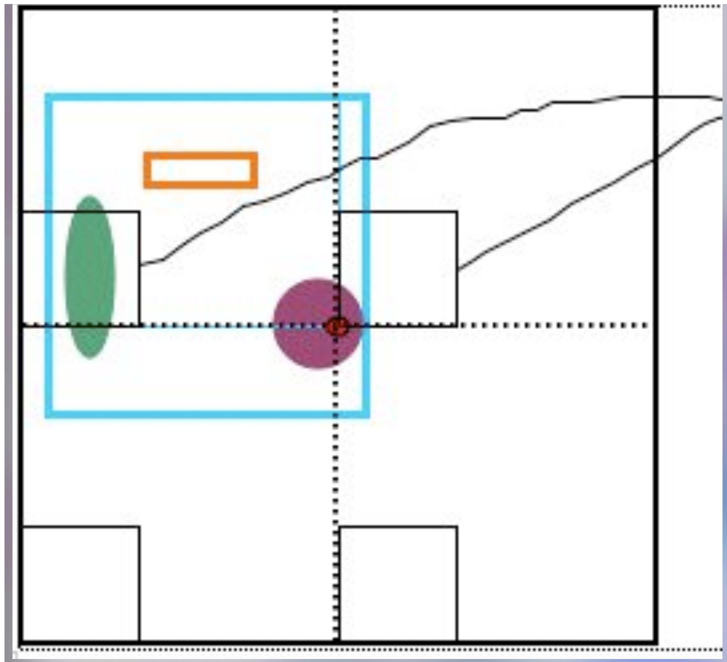
Depiction of Average Filtering



Computation of Average Filtering

- The average filter operation may be expressed (at most points) as the wraparound convolution of the image with an image of a square with intensity 1/M, where M = # pixels in the square





Opposite edges of image are being averaged together.

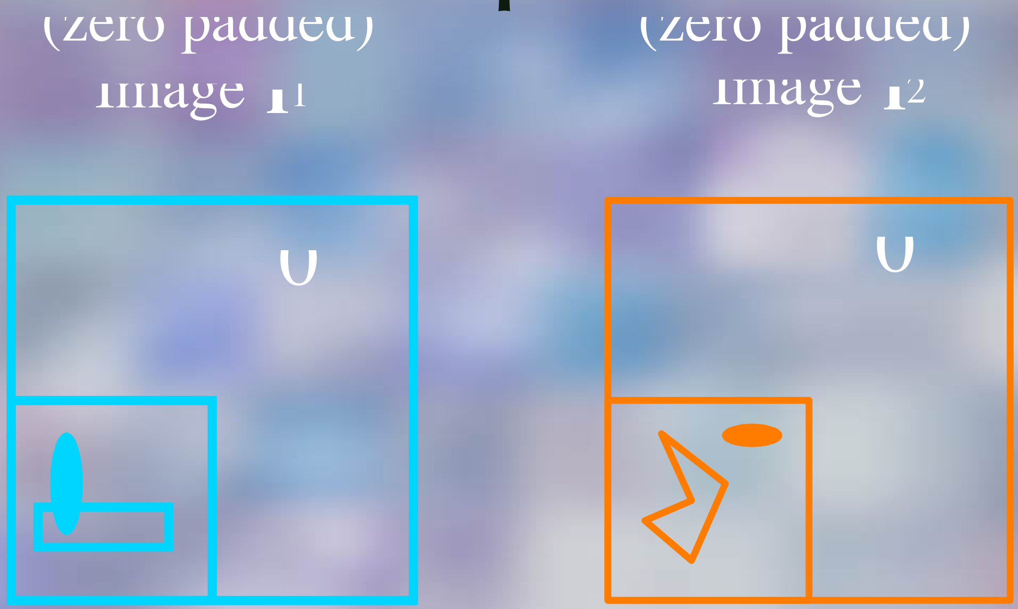
Wraparound Effect

- Near the image borders, however, wraparound effects occur.
- Usually, it is desirable to average only neighboring elements …
- … and convolution should superimpose and weight images according to their spatial ordering rather than DFT-induced periodic ordering.
- The effect is much worse if the filter is large.
- If the filter is small, then the effect can (perhaps) be trimmed from along borders

Linear Convolution by Zero Padding

- Adapting wraparound convolution to do linear convolution is conceptually simple.
- Accomplished by padding the two image arrays with zero values.
- Typically, both image arrays are doubled in size:

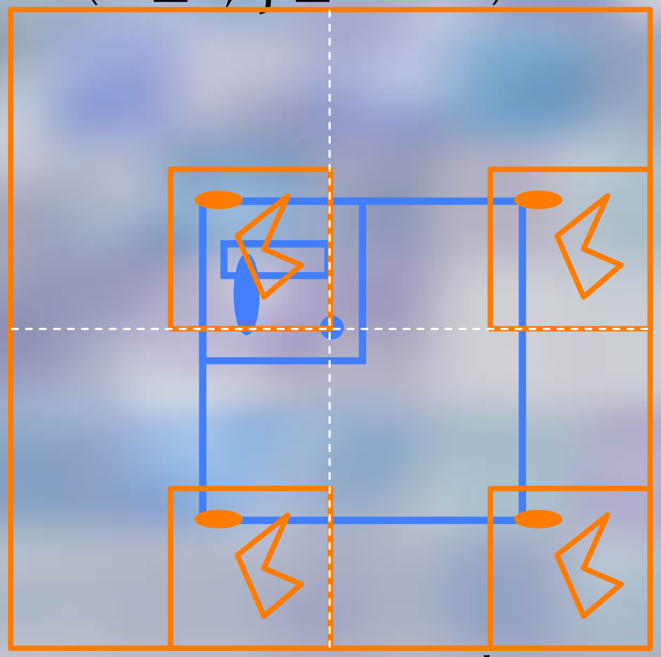
2N x 2M zero padded images



- Wraparound eliminated, since the "moving" image is weighted by zero values outside the image domain.
- Can be seen by looking at the overlaps when computing the convolution at a point (i, j):

Wraparound Cancelling Visualized

- Remember, the summations take place only within the blue shaded square ($0 \leq i, j \leq 2N-1$).



Linear convolution by zero padding.

Instead of summing over the periodic extension of the "moving image," zero values are summed with the weighted interior values.

DFT Computation of Linear Convolution

- Let $I_1´$, $I_2´$, and J´ be the 2N x 2N zero-padded versions of the images, and apply the FFT

$$J' = I'_1 \circledast I'_2 = IFFT_{2N}[FFT_{2N}[I'_1] \otimes FFT_{2N}[I'_2]]$$

- then the NxN sub-image with elements

$$J(i,j) = J'(i,j); (N/2)+1 \leq i,j \leq 3N/2$$

- contains the linear convolution result.

Recap DFT-Based Linear Convolution

- By multiplying zero-padded DFTs, then taking the IFFT, one obtains
- The linear convolution is larger than NxM (in fact 2Nx2M) but the interesting part is contained in NxM J.
- To convolve an NxM image with a small filter (say PxQ), where P,Q < N,M: pad the filter with zeros to size NxM.
- If P,Q << N,M, it may be faster to perform the linear convolution in the space domain.

Direct Linear Convolution

- Assume images I1, I2 are not periodically extended (not using the DFT!), and assume that

$$I_1[i,j] = I_2[i,j] = 0$$

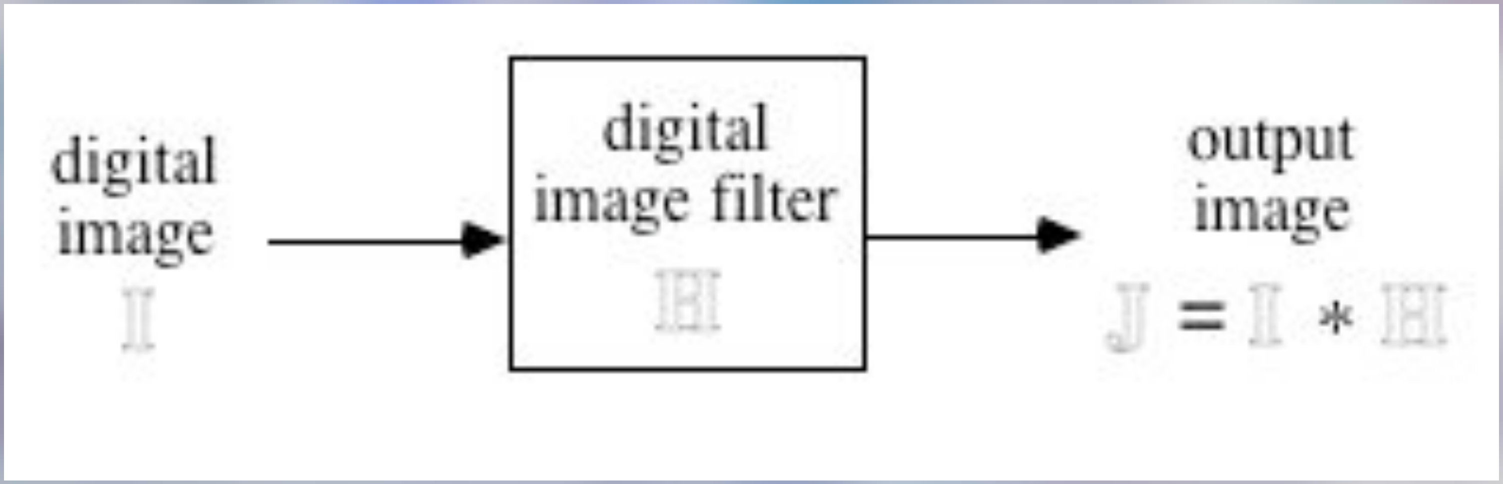
- whenever i < 0 or j < 0 or i > N-1 or j > M-1.

- In this case

$$J[i,j] = \sum_{m,n=0}^{N-1} I_1[m,n] I_2[i-m, j-n)]$$

LINEAR IMAGE FILTERING

- A process that transforms a signal or image I by linear convolution is a type of linear system.



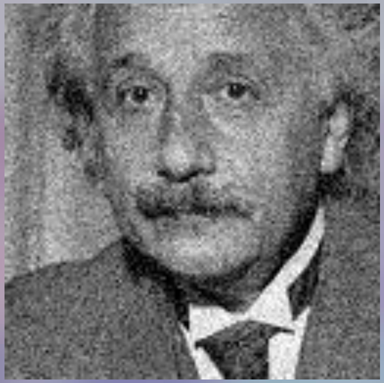
Goals of Linear Image Filtering

- Process sampled, quantized images to transform them into
- - images of better quality (by some criteria)
- - images with certain features enhanced
- - images with certain features de-emphasized or eradicated

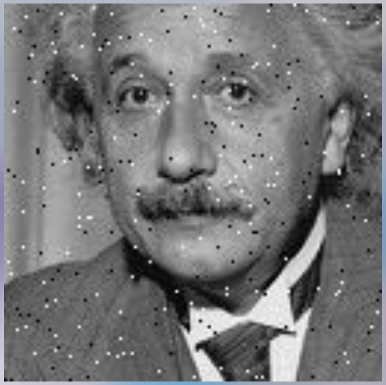
Some Specific Goals

- smoothing - remove noise from bit errors, transmission, etc
- deblurring - increase sharpness of blurred images
- sharpening - emphasize significant features, such as edges
- combinations of these

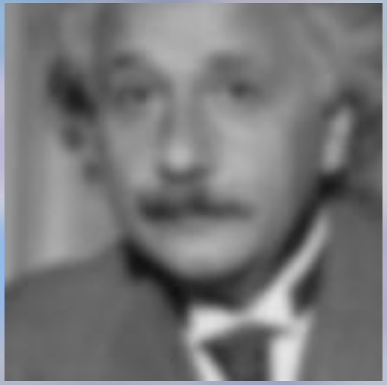
Variety of Image Distortions



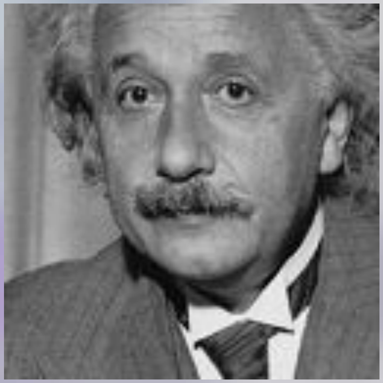
gaussian white noise



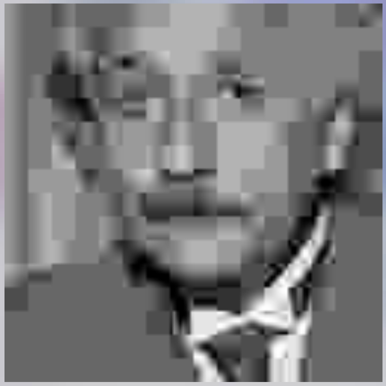
impulse noise



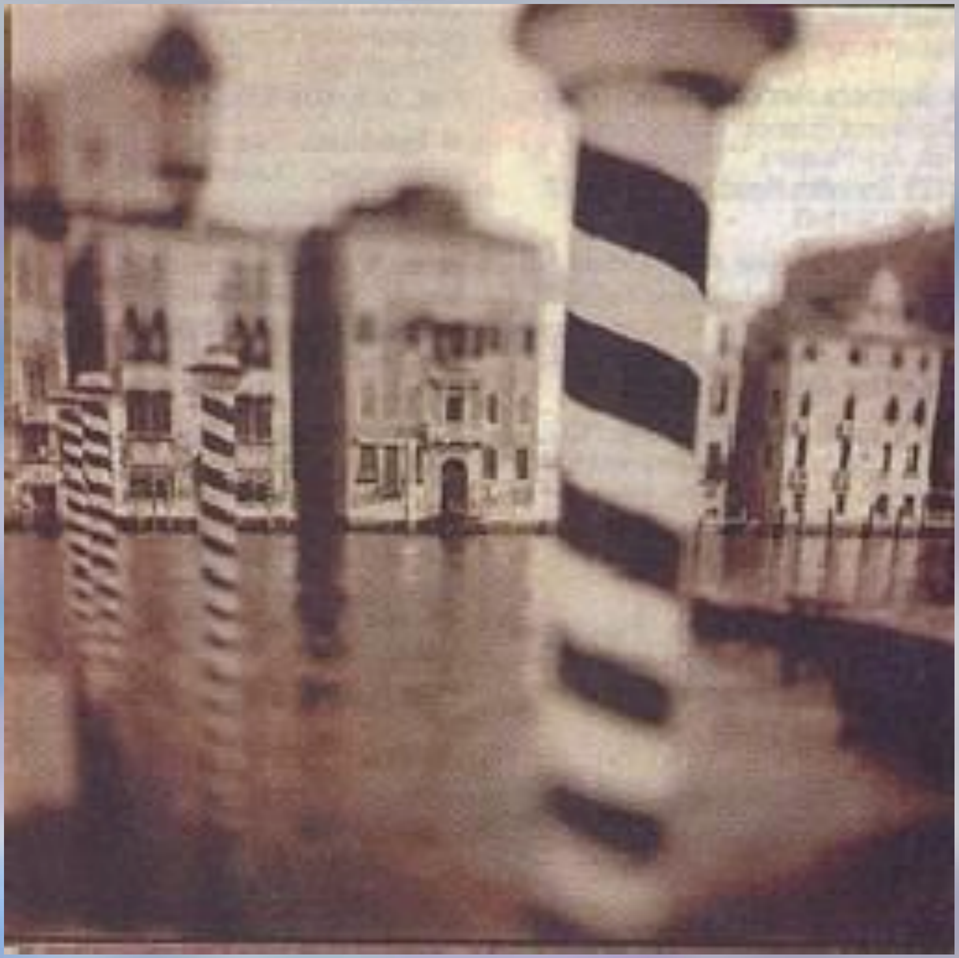
blur



Albert



JPEG compression



- A Tough One!
- Try to *undo* ("engineering problem") or, more interestingly, ***create*** this effect (creative application).

Low-Pass, Band-Pass, and High-Pass Filters

- The terms low-pass, band-pass, and high-pass are qualitative descriptions of a system's frequency response.
- "Low-pass" - attenuates all but the "lower" frequencies.
- "Band-pass" - attenuates all but an intermediate range of "middle" frequencies.
- "High-pass" - attenuates all but the "higher" frequencies.
- We have seen examples of these: the zero-one frequency masking results.

Generic Uses of Filter Types

- Low-pass filters are typically used to
- - smooth noise
- - blur image details to emphasize gross features

- High-pass filters are typically used to
- - enhance image details and contrast
- - remove image blur

- Bandpass filters are usually special-purpose

Example Low-Pass Filter

- The Gaussian filter with frequency response

$$\widetilde{H}_{continuous}(\omega_1, \omega_2) = e^{-2\pi^2\sigma^2(\omega_1^2+\omega_2^2)}$$

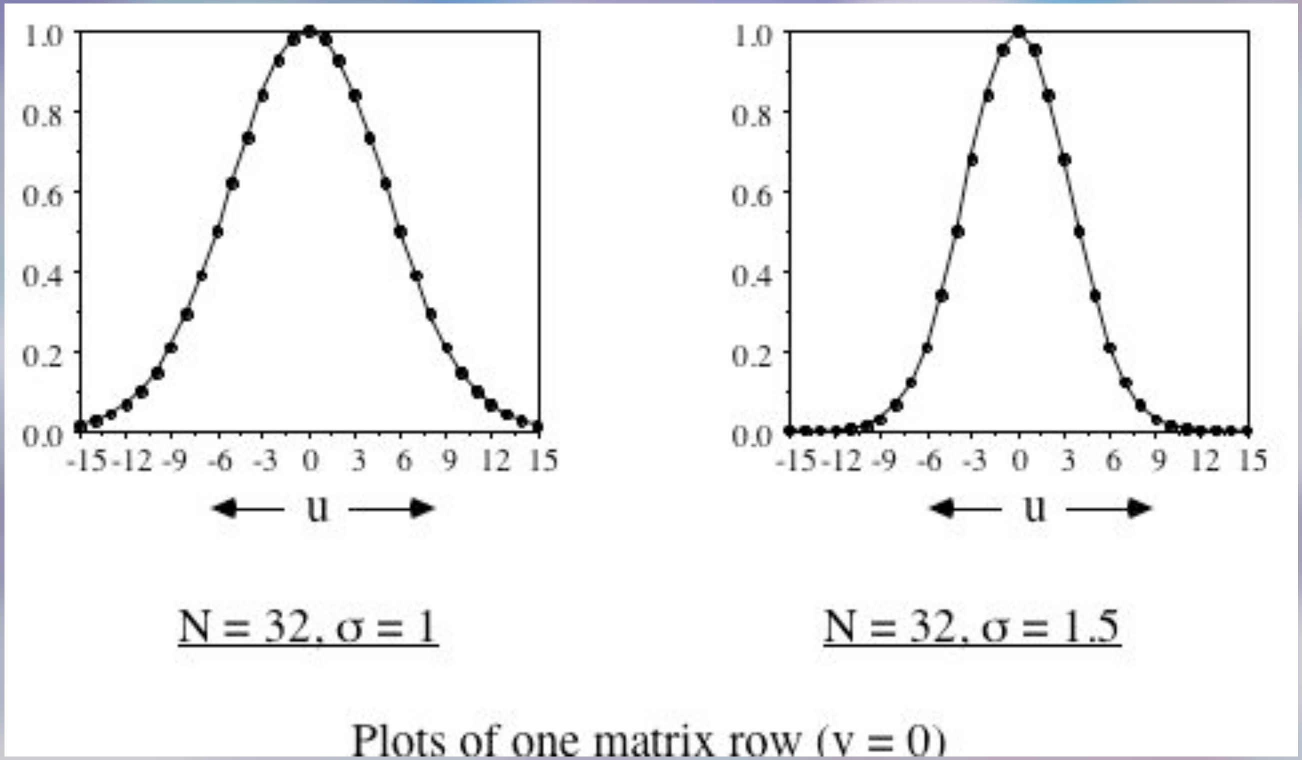
hence, sampling at $\omega_1 = \frac{u}{N}, \omega_2 = \frac{v}{N} \quad 0 \leq |u|, |v| \leq \frac{N}{2} - 1$

$$\widetilde{H}(u, v) = e^{-2\pi^2\sigma^2(u^2+v^2)/N^2}$$

- which quickly falls at larger frequencies.

- The Gaussian is an important low-pass filter.

Gaussian Filter Profile



N = 32, σ = 1

N = 32, σ = 1.5

Plots of one matrix row (v = 0)

Example Band-Pass Filter

- Can define a BP filter as the difference of two LPFs identical except for a scaling factor.
- A common choice in image processing is the difference-of-gaussians (DOG) filter, with frequency scaling factor K:

$$\widetilde{H}_C(\omega_1, \omega_2) = e^{-2(\sigma\pi)^2(\omega_1^2+\omega_2^2)} - e^{-2(K\sigma\pi)^2(\omega_1^2+\omega_2^2)}$$

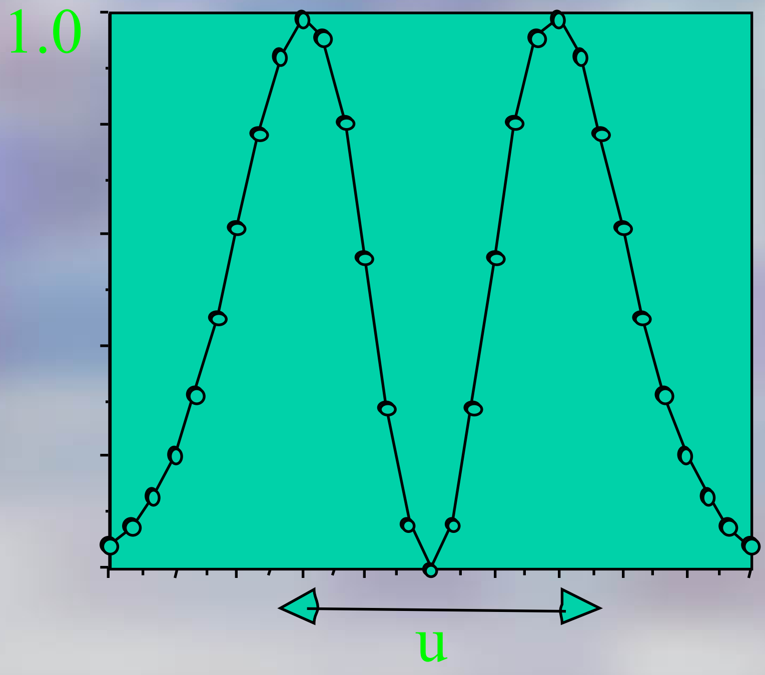
- hence

$$\widetilde{H}(u, v) = e^{-2(\sigma\pi)^2(u^2+v^2)/N^2} - e^{-2(K\sigma\pi)^2(u^2+v^2)/N^2}$$

- Typically, K ≈ 1.5.

DOG Filter Profile

- DOG filters are very useful for image analysis – and in human visual modelling.
- Take K=1.5, $\sigma < 5$



N=32

Example High-Pass Filter

- The Laplacian filter is also important

$$\widetilde{H}_C(\omega_1, \omega_2) = A(\omega_1^2 + \omega_2^2)$$

- hence

$$\widetilde{H}(u, v) = A(u^2 + v^2)/N^2$$

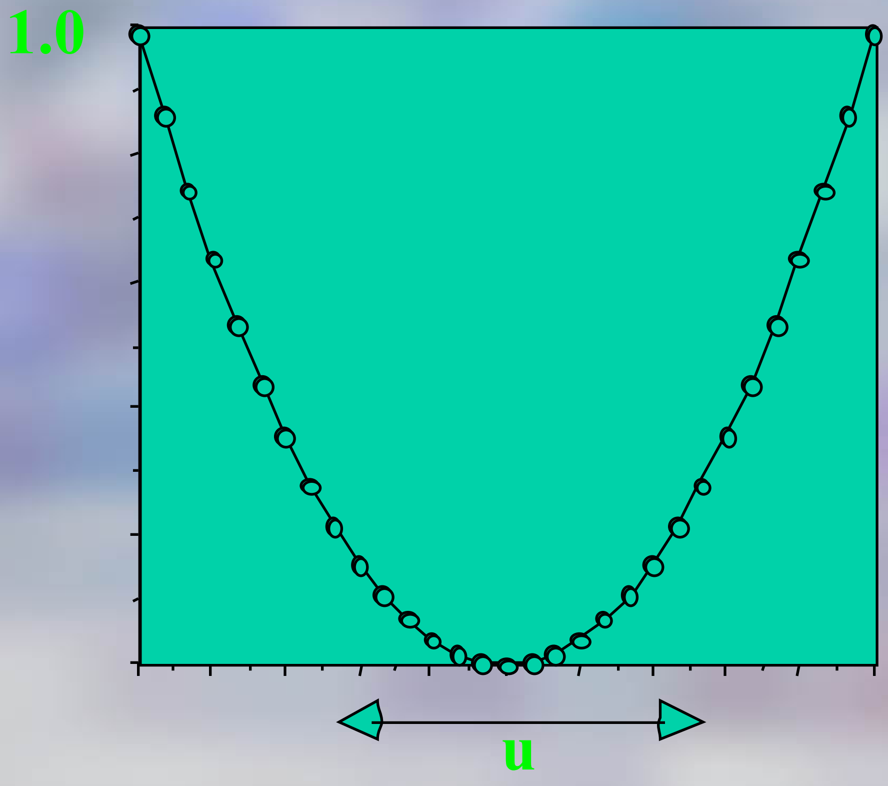
- An approximation to the Fourier transform of the continuous Laplacian:

$$\Delta = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$$

(Heat equation ++!)

Laplacian Profile

- A = 4.5, N = 32 $\widetilde{H}(u,v) = A(u^2 + v^2)/N^2$



LINEAR IMAGE DENOISING

- Linear image denoising means a process that smooths noise without destroying the image information.
- The noise is usually modeled as additive or multiplicative.
- We consider additive noise now.
- Multiplicative noise is better handled by a homomorphic filtering that uses nonlinearity.

Additive White Noise Model

- Model additive white noise as an image N with highly chaotic, unpredictable elements.
- Can be thermal circuit noise, channel noise, sensor noise, etc.
- Noise may effect the continuous image before sampling:

$$J_C(x, y) = I_C(x, y) + N_C(x, y)$$

where N is the white noise

Zero-Mean White Noise

- The white noise is zero-mean if the limit of the average of P arbitrary noise image $N_C(x_i, y_i)$; $i = 1$,...., P: vanishes as $P \to \infty$:

$$mean_P[N_C] = \frac{1}{P}\sum_{i=1}^{P} N_C(x_i, y_i)$$

$$mean_P[N_C] \to 0 \text{ as } P \to \infty$$

- On average, the noise falls around the value zero.*

- *Strictly speaking, the noise is also "mean-ergodic."

Spectrum of White Noise

- The noise energy spectrum is the Fourier transform of N

$$\widetilde{N}(\omega_1, \omega_2)$$

- If the noise is white, then, on average, the energy spectrum will be flat (flat spectrum = 'white'):

$$\underset{P \to \infty}{mean_P}[|\widetilde{N}(\omega_1, \omega_2)|^2] \to \eta \qquad \forall(\omega_1, \omega_2)$$

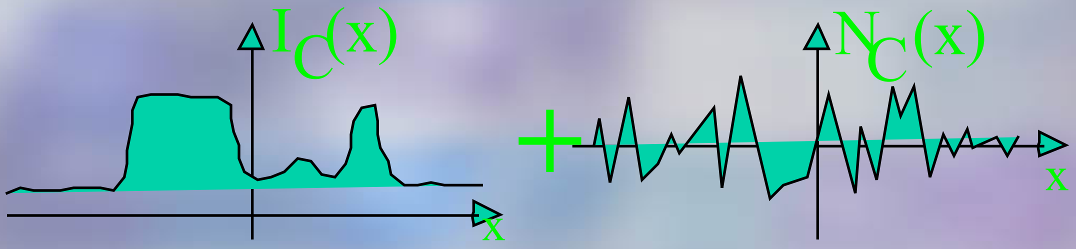
- Note: η is called noise power.

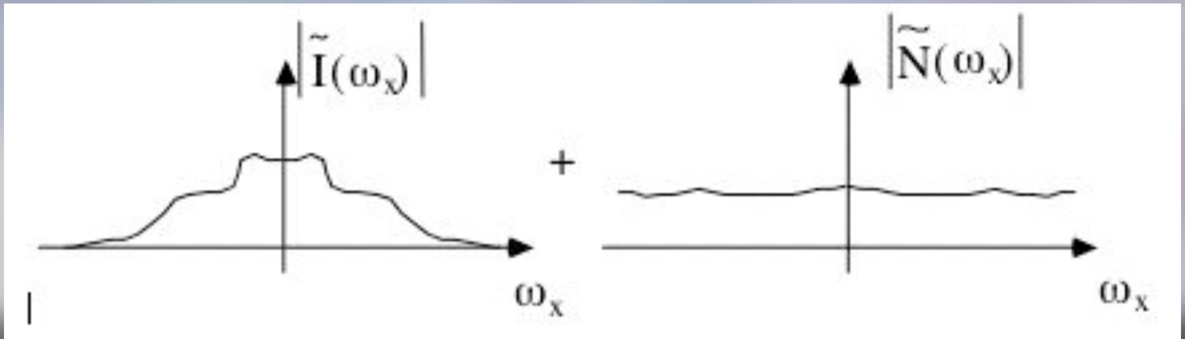
White Noise Model

- White noise is an approximate model of additive broadband noise:

$$J'_C(\omega_x, \omega_y) = I'_C(\omega_x, \omega_y) + N'_C(\omega_x, \omega_y)$$

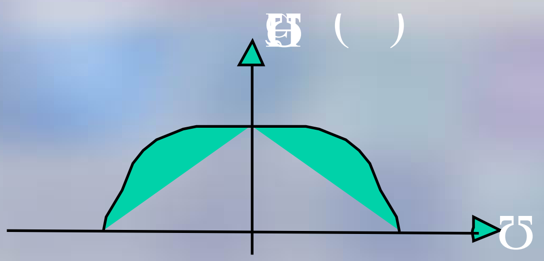
' denotes transform





Linear Denoising

- Objective: Remove as much of the high-frequency noise as possible while preserving as much of the image spectrum as possible.
- Generally accomplished by a Low Pass Filter of fairly wide bandwidth (images are fairly wideband):



Denoising - Gaussian Filter

- The isotropic Gaussian filter is an effective :

$$\widetilde{H}(u,v) = \widetilde{H}(u^2+v^2) = e^{-2(\sigma\pi)^2(u^2+v^2)/N^2}$$

- It gives more weight to "closer" neighbors.

- DFT design: Set the half-peak bandwidth $\sqrt{u^2+v^2} = U_{cutoff}$
Solve for σ:

$$e^{-2\pi^2\sigma^2 U_{cutoff}^2/N^2} = 1/2$$

$$\sigma = \frac{NU_{cutoff}}{\pi}\sqrt{log\sqrt{2}}$$

LINEAR IMAGE DEBLURRING

- Often an image that is obtained digitally has already been corrupted by a linear process.
- This may be due to motion blur, blurring due to defocusing, etc.
- We can model such an observed image as the result of a linear convolution:
- $J_C(x, y) = G_C(x, y) * I_C(x, y)$

so the FFT

$$\widetilde{J}_C(\omega_1, \omega_2) = \widetilde{G}_C(\omega_1, \omega_2) \cdot \widetilde{I}_C(\omega_1, \omega_2)$$

Digital Blur Function

- The sampled image will then be of the form (assuming sufficient sampling rate

$$J = G * I$$

- with DFT

$$\widetilde{J} = \widetilde{G} \otimes \widetilde{I}$$

- The distortion G is almost always low-pass (blurring).
- Our goal is to use digital filtering to reduce blur – a VERY hard problem!

Deblur - Inverse Filter

- Often it is possible to make an estimate of the distortion G.
- This may be possible by examining the physics of the situation.
- For example, motion blur (relative camera movement) is usually along one direction. If this can be determined, then a filter can be designed.
- The effect of a camera can often be determined – and hence, a digital deblur filter designed.

Deconvolution

- Reversing the linear blur G is deconvolution. It is done using the inverse filter of the distortion:

$$\widetilde{G}^{inverse}(u,v) = 1/\widetilde{G}(u,v) \qquad 0 \leq |u|,|v| \leq \frac{N}{2} - 1$$

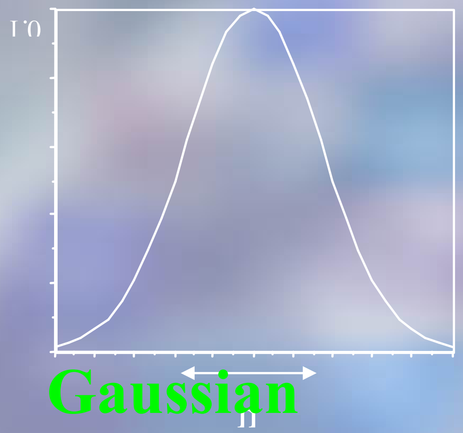
- Then the DFT of the restored image is:

$$\widetilde{K} = \widetilde{G}^{inverse} \otimes \widetilde{G} \otimes \widetilde{I}$$

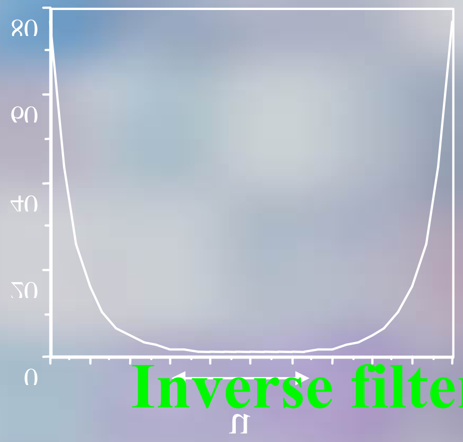
The challenge, of course, is to model G

Blur Estimation

- An estimate of blur G might be obtainable.
- The inverse of low-pass blur is high-pass:



Gaussian distortion

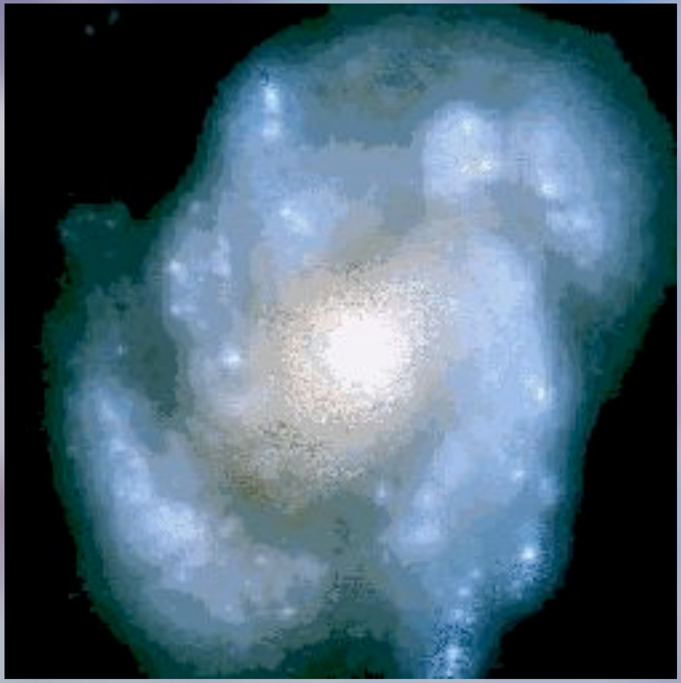


Inverse filter

Other Results

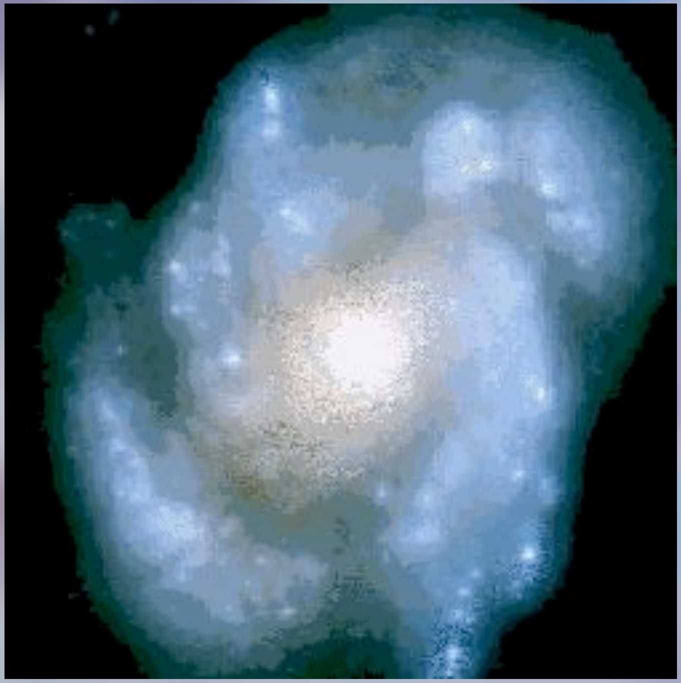
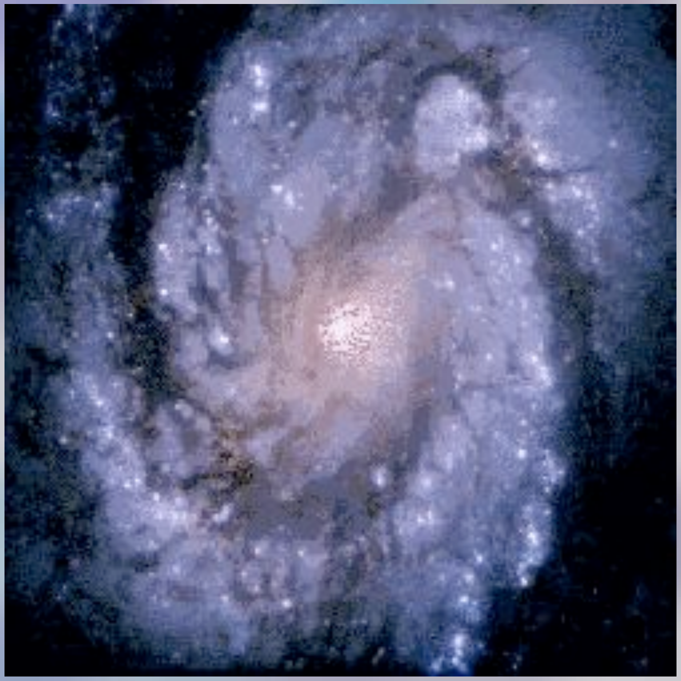
Hubble Telescope

- Wide Field Planetary Camera
- Galaxy M100



Hubble Telescope

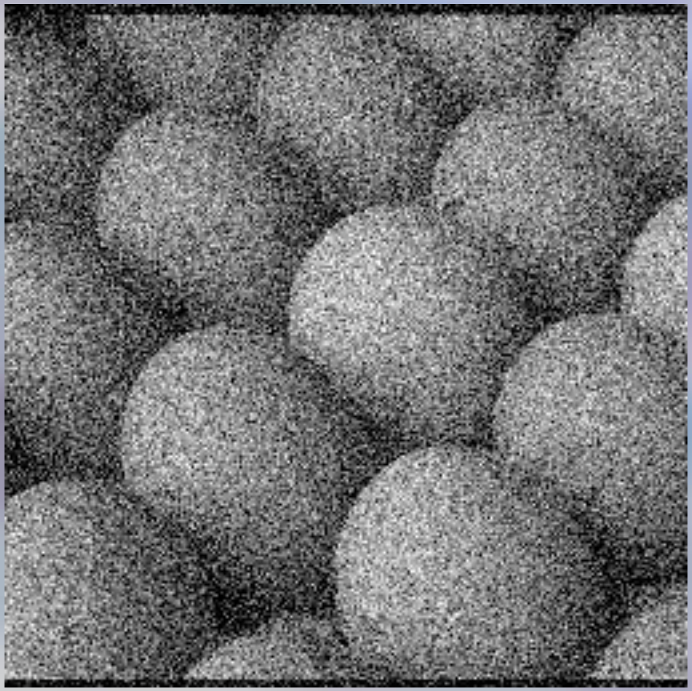
- Wide Field Planetary Camera
- Galaxy M100

- after repairing spherical aberration

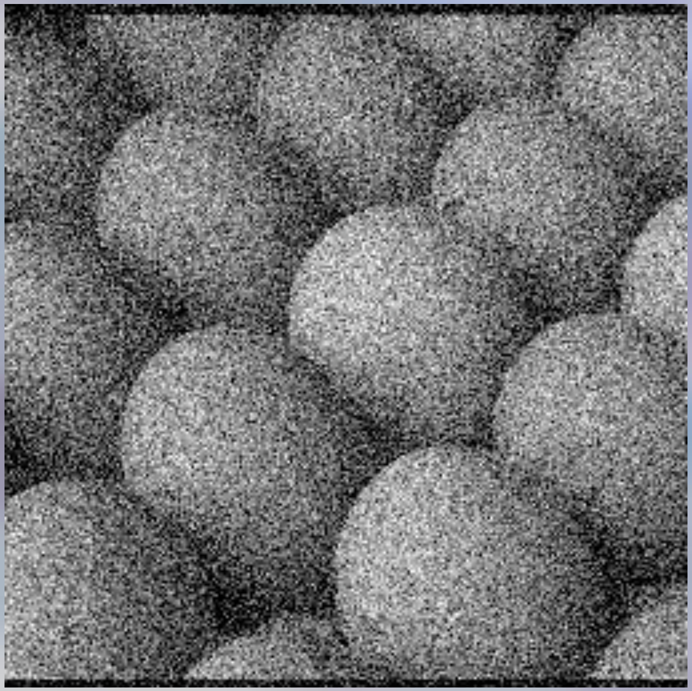
Average Filtering

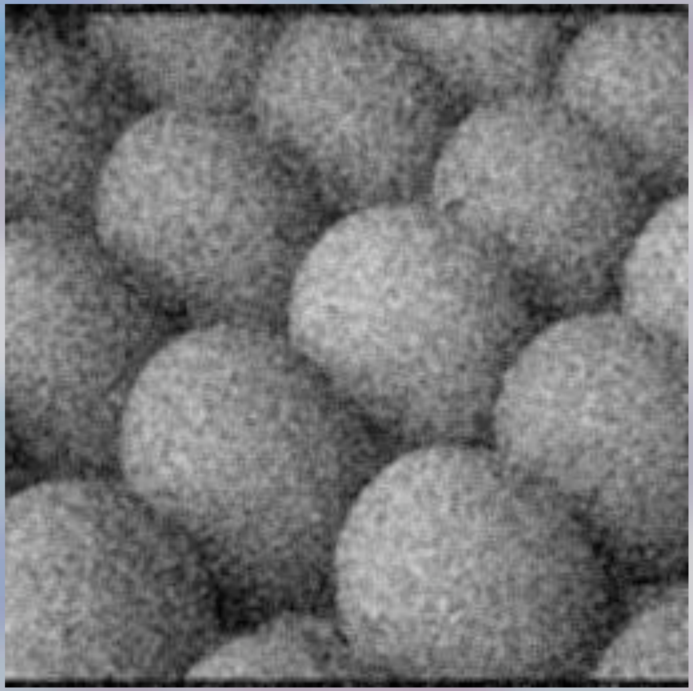
- Eggs + Gaussian noise:



Average Filtering

- Eggs + Gaussian noise:

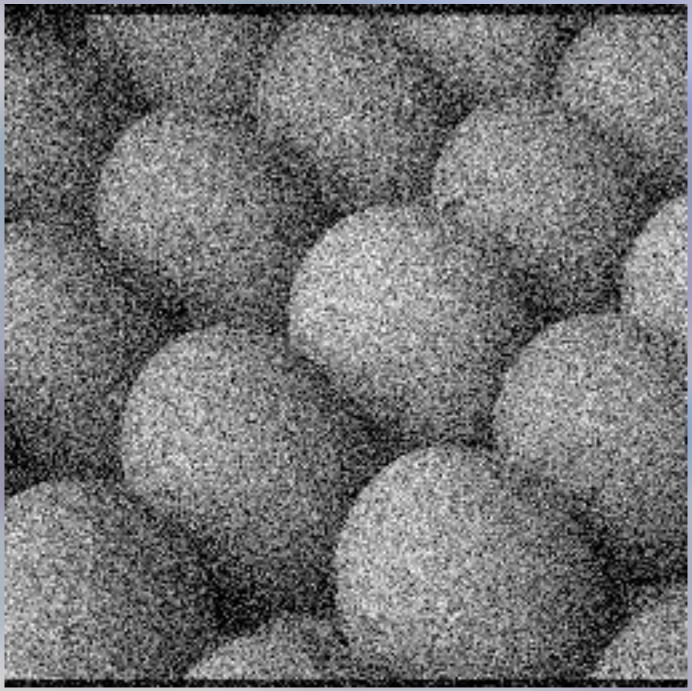


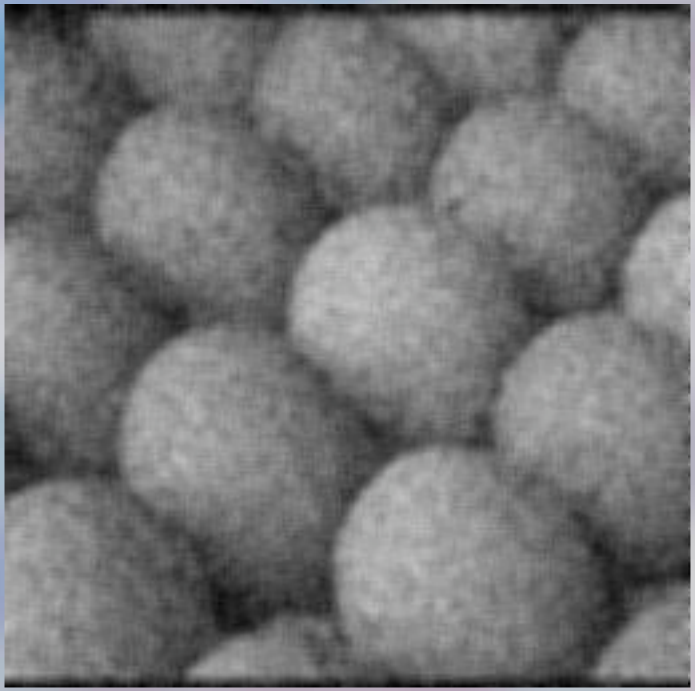


AVE[eggs, SQUARE (9)]

Average Filtering

- Eggs + Gaussian noise:

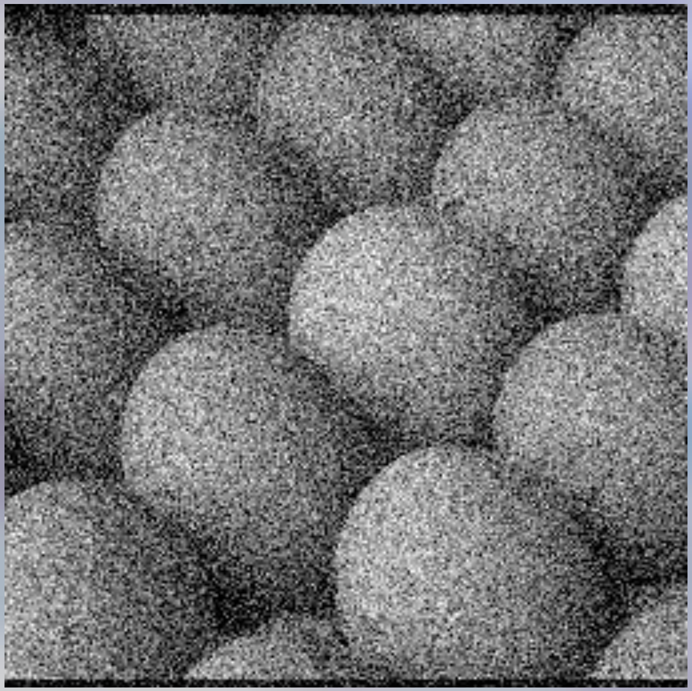


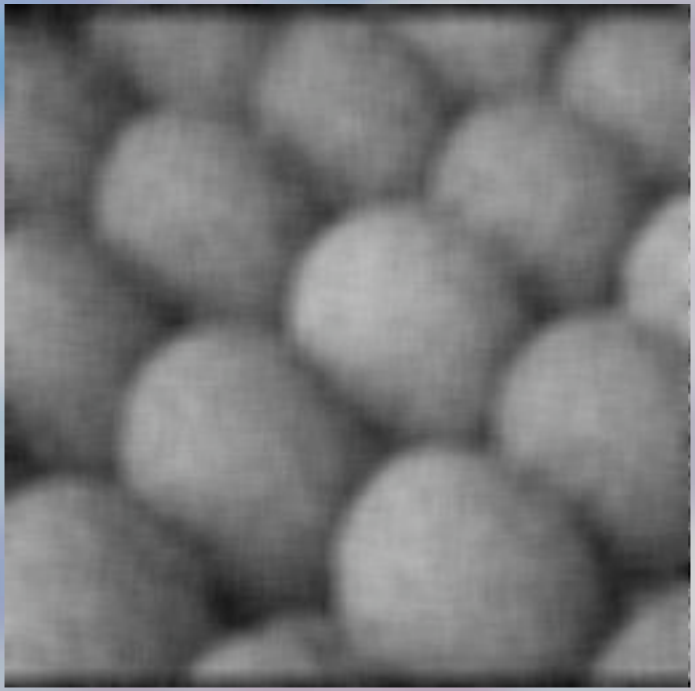


AVE[eggs, SQUARE (25)]

Average Filtering

- Eggs + Gaussian noise:

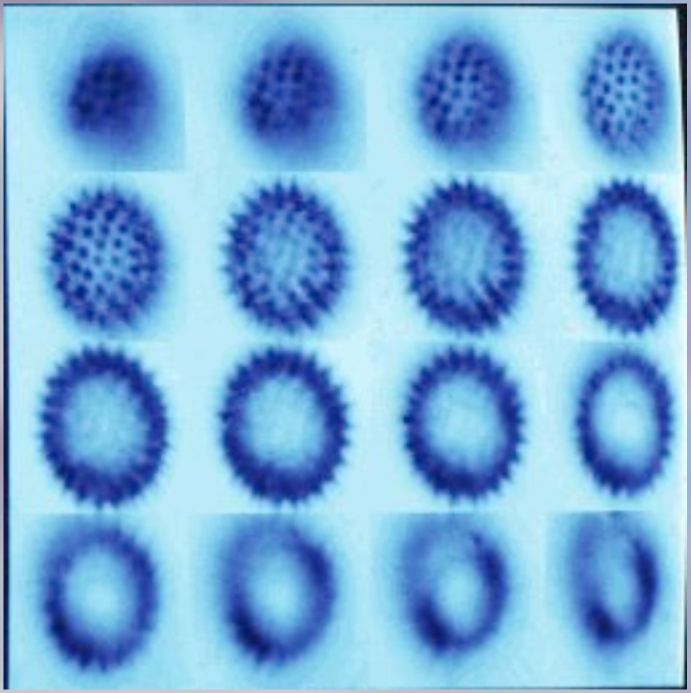


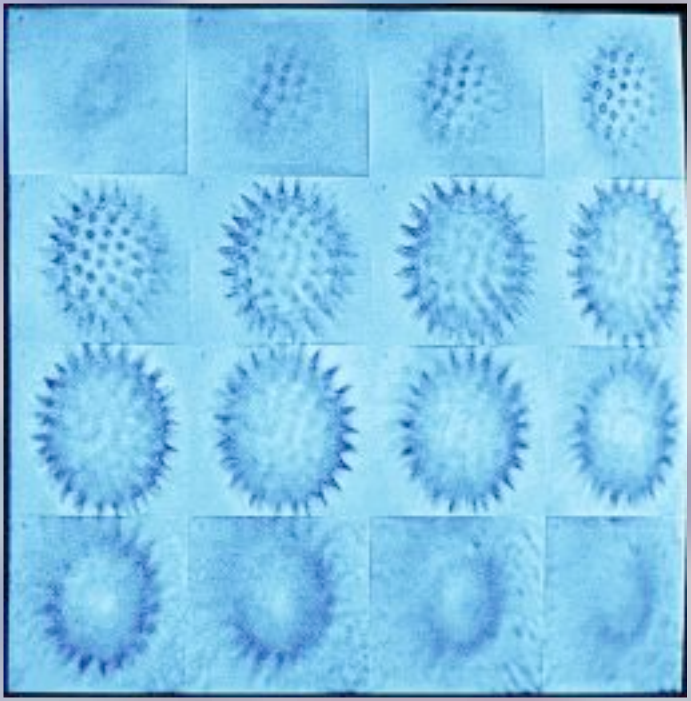


AVE[eggs, SQUARE (81)]

Optical Serial Sectioning Microscopy

- Sequence of sections of pollen grains

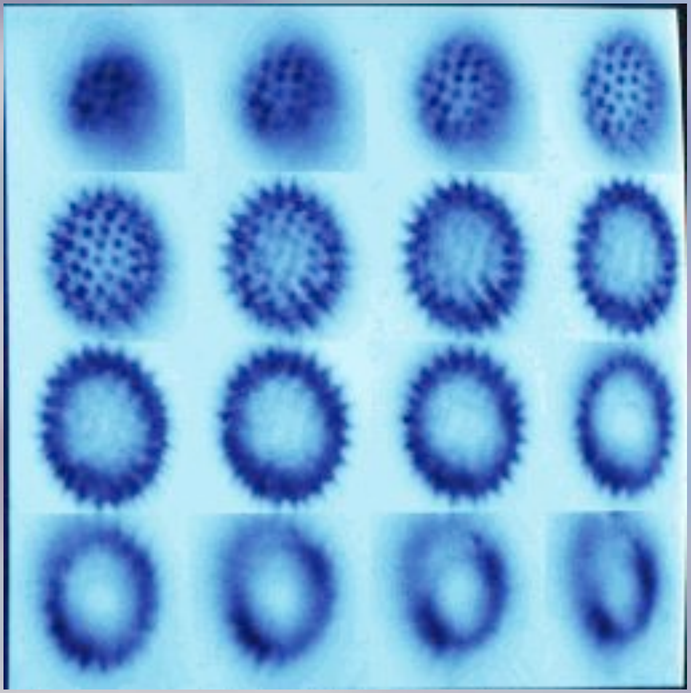


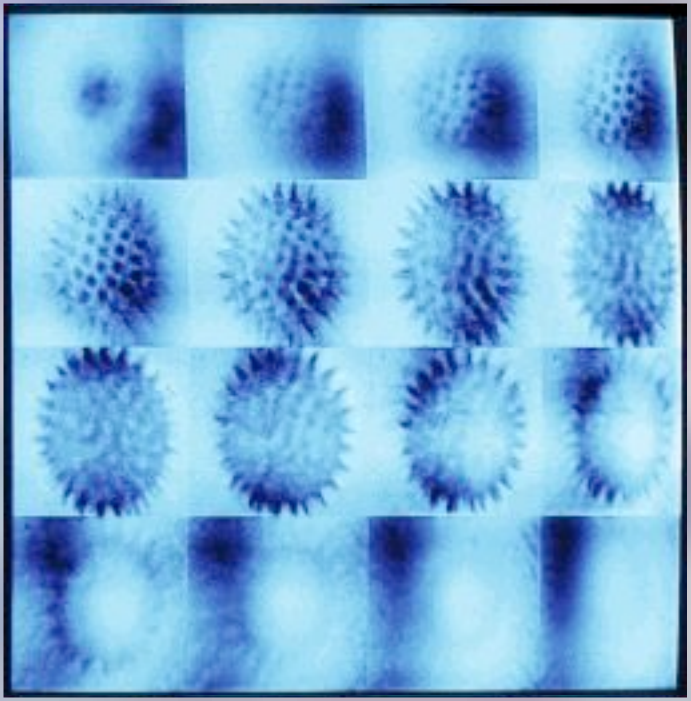


Inverse filtering:
High frequencies suppressed

Optical Serial Sectioning Microscopy

- Sequence of sections of pollen grains





Wiener filtering:
good for blur + noise

more blur

- Deblurring
- Pseudo-inverse
- Wiener filter

Deblur - Missing Frequencies

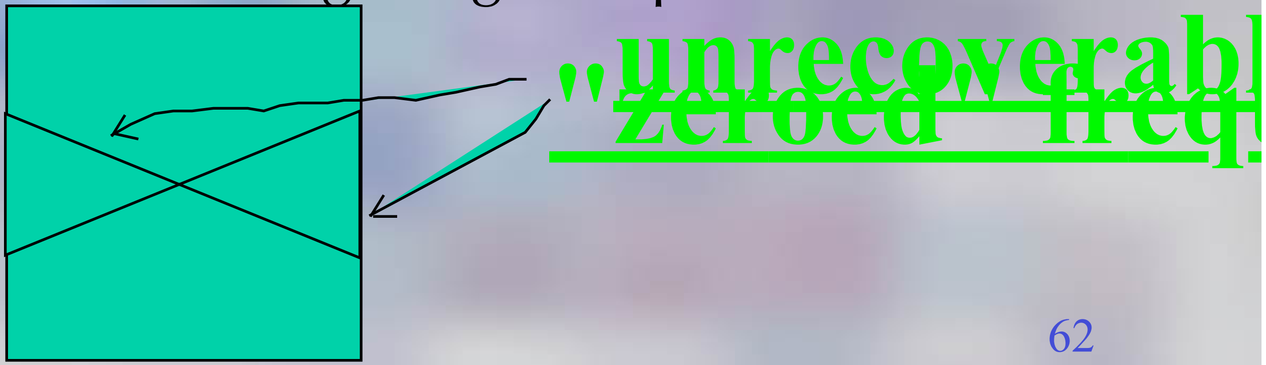
- Unfortunately, things are not always so "ideal" in the real world.
- Sometimes the blur frequency response takes zero value (s).
- If

$H(u_0, v_0) = 0$ for some (u_0, v_0), then $H_{inverse}(u_0, v_0) =$

- which is meaningless.

Zeroed Frequencies

- The reality: any frequencies that are zeroed by a linear distortion are unrecoverable in practice (at least by linear means) - lost forever!
- The best that can be done is to reverse the distortion at the non-zero values.
- Sometimes much of the frequency plane is lost. Some optical systems remove a large angular spread of frequencies:



Pseudo-Inverse Filter

- The pseudo-inverse filter is defined

  [illegible]

- Thus no attempt is made to recover lost frequencies.
- The pseudo-inverse is set to zero in the known region of missing frequencies – a conservative approach.
- In this way spurious (noise) frequencies will be eradicated.

Deblur in the Presence of Noise

- A worse case is when the image I is distorted both by linear blur G and additive noise N:

$$J = G I + N$$

- This may occur, e.g., if an image is linearly distorted then sent over a noisy channel.
- The DFT:

$$\hat{J} = \hat{G} \hat{I} + \hat{N}$$

Filtering a Blurred, Noisy Image

- Filtering with a linear filter H will produce the result

$$K \quad H * J \quad * H \quad C * I \quad H \quad N$$

- or

$$K \quad H \quad J \quad H \quad C \quad I \quad H \quad N$$

- The problem is that neither a low-pass filter (to smooth noise, but won't correct the blur) nor a high-pass filter (the inverse filter, which will amplify the noise) will work.

Failure of Inverse Filter

- If the inverse filter were used, then

$$K^{inverse}C \quad = \quad f \quad H^{inverse}C \quad N$$

- or

$$K^{inverse}C \quad \otimes \quad r \quad H^{inverse}C \quad \otimes \quad N$$

- In this case the blur is corrected, but the restored image has horribly amplified high-frequency noise added to it.

Wiener Filter

- The Wiener filter (after Norbert Wiener) or minimum-mean-square-error (MMSE) filter is a “best” linear approach.
- The Wiener filter for blur G and white noise N is

[illegible] $=$

- Often the noise factor η is unknown or unobtainable. The designer will usually experiment with heuristic values for η.
- In fact, better visual results may often be obtained by

Wiener Filter Rationale

- We won't derive the Wiener filter here. But:
- If η = 0 (no noise), the Wiener filter reduces to the inverse filter: [illegible]
- which is highly desirable.

Wiener Filter Rationale

- If $G(u, v) = 1$ for all (u, v) (no blur) the Wiener filter reduces to:

$$G_{\text{Wiener}}(u, v) =$$

- which does nothing except scale the variance so that the MSE is minimized.
- So, the Wiener filter is not useful unless there is blur.

Pseudo-Wiener Filter

- Obviously, if there are frequencies zeroed by the linear distortion G then it is best to define a pseudo-Wiener filter:

$$[illegible] = 0 \text{ if } G(u, v) = 0$$

- Noise in the "missing region" of frequencies will be eradicated.
- DEMO (σblur < 4, η << 1, σnoise < 10)

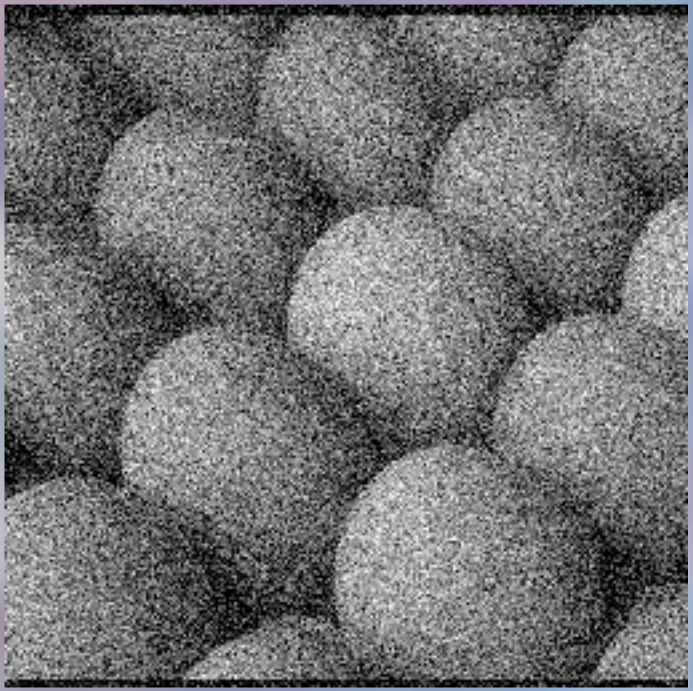
Shroud of Turin Image

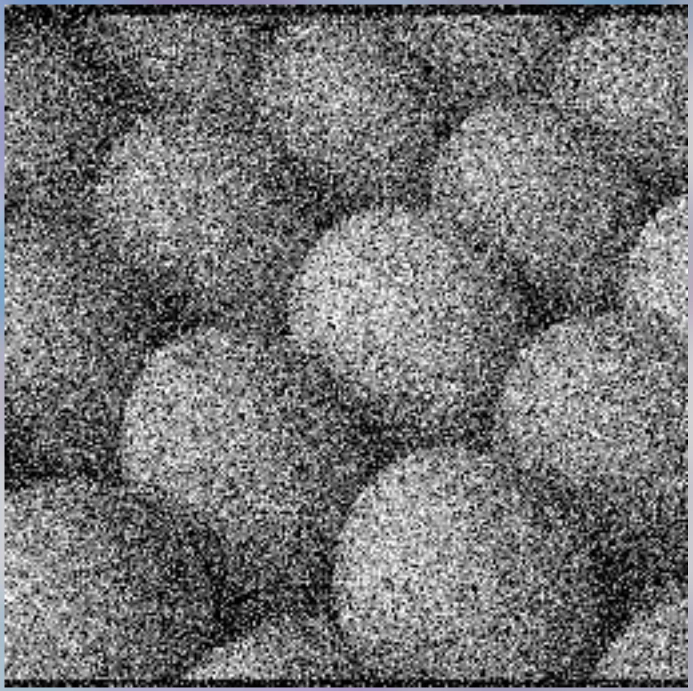
- An intensely enhanced, denoised, deblurred, etc etc etc and debated image

Making Noise

- Gaussian Additive Noise
- Laplacian Additive White Noise
- Exponential Multiplicative White Noise
- Salt and Pepper Noise
- What Is Noise?

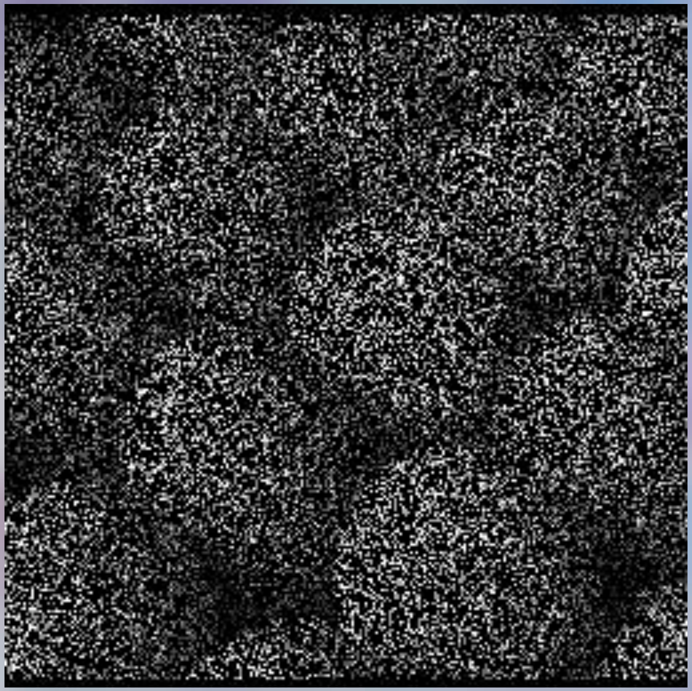
Additive White Noise

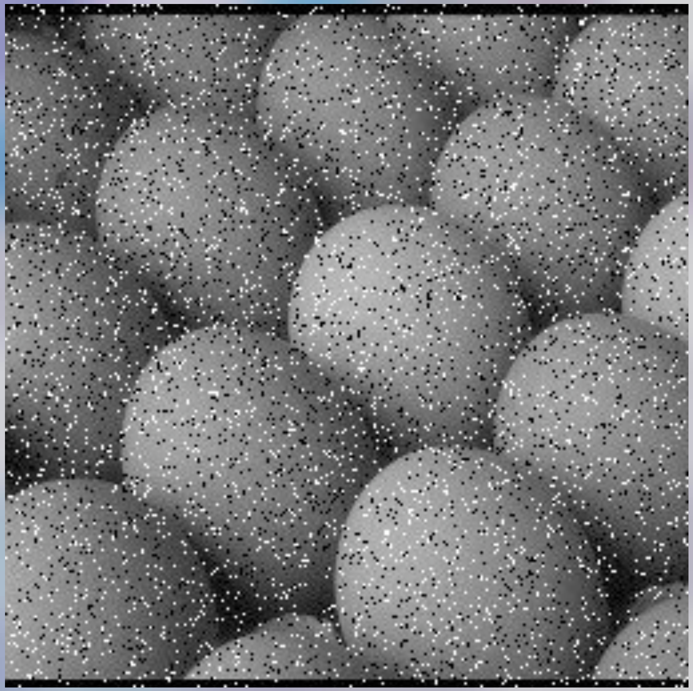




- Gaussian Laplacian

More Noise





- Exponential Multiplicative White Noise

Salt and Pepper

Comments

- ***Non***-linear filtering methods include
  - weighted median filters,
  - image zooming,
  - sharpening,
  - edge detection

What is noise?
Source of synthesis texture

- Signal vs. Noise
- Attention, John Cage, music as organized sound