

Module 6

Image Denoising and Nonlinear Filtering

- Noise Models
- Morphological and Order Statistic Filters
- Bilateral Filter
- Anisotropic Diffusion
- Non-local (NL) Means
- Wavelet Soft Thresholding
- BM3D
- Homomorphic Filtering

1

Range of Noise

- The set of values \mathfrak{R}_N the noise can take. In digital images the noise must be discrete integers. However, we will use approximate continuous models.
- For additive noise the noise range is usually symmetric: $\mathfrak{R}_N = [-Q, Q]$ where possibly $Q = \infty$.
- For multiplicative noise assume the noise range is non-negative: $\mathfrak{R}_N = [0, Q]$ where possibly $Q = \infty$.

4

IMAGE NOISE MODELS

- To proceed further in understanding and designing non-linear filters, we will need to deepen our statistical modeling of noise.
- Any channel (electrical wire, airwaves, optical fiber) is imperfect. Images sent over a channel always suffer a degradation of information:
 - (a) static (high-frequency noise or thermal noise)
 - (b) bit errors
 - (c) sensor noise
- These errors degrade
 - visual interpretation
 - computer image analysis

2

Probability Density Function of Noise

- The probability density function or PDF $f_N(q)$ is a model of the probabilities of $N(m, n)$:

$$\Pr\{a \leq N(m, n) \leq b\} = \int_a^b f_N(q) dq$$

where $[a, b] \subseteq \mathfrak{R}_N$.

- Assume that all elements $N(m, n)$ of N obey the identical probability model - the same PDF.
- So, $f_N(q)$ is not expressed as a function of (m, n) .

5

White Noise is Uncorrelated Noise

- Assume the additive white noise model:
- $$\mathbf{J} = \mathbf{I} + \mathbf{N}$$
- where \mathbf{I} is an original image and \mathbf{N} a digital white noise image.
- White noise suggests an (on average) a flat spectrum.
 - An equivalent meaning is that the elements $N(m, n)$ of \mathbf{N} are uncorrelated: any two elements $N(m, n), N(p, q)$ of the noise matrix \mathbf{N} (where $(m, n) \neq (p, q)$) affect one another little or not at all.
 - Mathematically this is equivalent to the noise being white.

3

Properties of PDF

- Two basic properties of the PDF

$$(1) f_N(q) \geq 0 \text{ for all } q \in \mathfrak{R}_N$$

$$(2) \int_{\mathfrak{R}_N} f_N(q) dq = 1$$

- Note: (1) and (2) imply that $f_N(q) \leq 1$ for all $q \in \mathfrak{R}_N$.
- For symmetric noise: $\mathfrak{R}_N = [-Q, Q]$ assume that $f_N(q)$ is a symmetric function

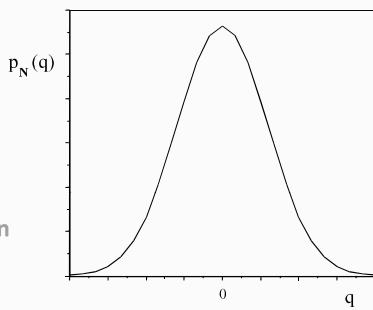
$$f_N(q) = f_N(-q)$$

6

Gaussian PDF

- Here $\Re_N = [-\infty, \infty]$ and

$$f_N(q) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(q/\sigma)^2}$$
 with scale parameter σ .
- The parameter σ^2 is the variance of the PDF
- The most common noise that occurs in applications ... and in nature.
- The usual noise found in electrical circuits, most communication channels, etc.

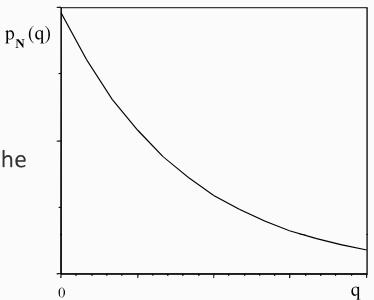


7

Exponential PDF

- Here $\Re_N = [0, \infty]$ and

$$f_N(q) = \frac{1}{\sigma} e^{-q/\sigma}$$
 with scale parameter σ .
- Parameter σ^2 is again equal to the variance and s is the mean.
- Often models multiplicative noise.
- A coherent image noise model (radar, laser, electron microscopy, etc)

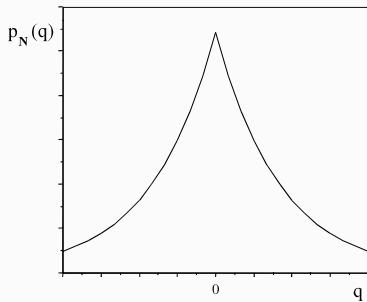


10

Laplacian PDF

- Here $\Re_N = [-\infty, \infty]$ and

$$f_N(q) = \frac{1}{\sqrt{2}\sigma} e^{-\sqrt{2}|q|/\sigma}$$
 with scale parameter σ .
- Parameter σ^2 is again the variance.
- Often models impulse noise.
- Large number of small-amplitude values, and a (relatively) large number of high-amplitude values.



8

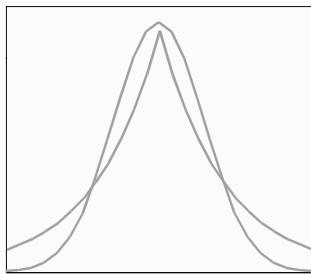
Salt-and-Pepper (S&P) Noise

- A model for significant bit errors.
- When transmitting digital image data, significant bit errors can occur:

'1' → '0'	('light' → 'dark')
'0' → '1'	('dark' → 'light')
- May be due to thermal effects or even gamma rays.
- Salt and pepper noise is not additive.

11

Gaussian vs. Laplacian PDFs



Gaussian
Laplacian

- Laplacian noise is impulsive – a heavy-tailed PDF.

9

S&P Noise Model

- Let I_{\max} and I_{\min} be the brightest and darkest allowable gray levels in image I (e.g. 0 & 255)
- Define the simple probabilities

$$p_{s-p}(q) = \begin{cases} (1-p)/2 & ; q = -1 \\ p & ; q = 0 \\ (1-p)/2 & ; q = 1 \end{cases}$$
- If J is the noisy image, then

$$J(i, j) = \begin{cases} I_{\min} & ; q = -1 \\ I(i, j) & ; q = 0 \\ I_{\max} & ; q = 1 \end{cases}$$
- This is a simple model for the worst case type of bit errors

12

Statistical Mean

- The statistical mean of N is

$$\mu_N = \int_{\mathcal{R}_N} q \cdot f_N(q) dq$$

- It measures the center about which the noise realizations cluster.

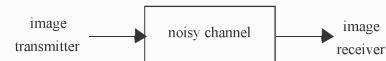
- For a symmetric PMF (Q can be ∞),

$$\mu_N = \int_Q q \cdot f_N(q) dq = 0 \quad (\text{why?})$$

13

Denoising

- Consider denoising of an image contaminated by white noise:



- Two goals of filtering:

- Smoothing - reduce noise (bit errors, channel, etc)

- Preservation - of features: edges and detail

16

Statistical Variance

- The variance of N is

$$\sigma_N^2 = \int_{\mathcal{R}_N} (q - \mu_N)^2 \cdot f_N(q) dq$$

- The standard deviation σ_N is the average distance from μ_N that the noise realizations fall.
- For symmetric (zero-mean) noise, the variance is just

$$\sigma_N^2 = \int_{\mathcal{R}_N} q^2 \cdot f_N(q) dq$$

14

Smoothing vs. Preservation

- Image denoising contains conflicting goals:

- Smoothing: high noise frequencies are attenuated
- Broadband images: both low and high frequencies are important

- Linear filtering cannot differentiate between desirable high frequencies and undesirable high frequencies

- A linear low-pass denoising filter will always:

- Reduce high frequency noise
- Blur the image

17

Mean & Variance of PDFs

- Exercises: Show that for Gaussian, Laplacian, and exponential PDFs, the variance is σ^2 .
- Also show that for exponential PDF, the mean is equal to σ .

15

WHY NONLINEAR FILTERING ?

- Linear filtering has important limitations.

- The basis of linear filtering is frequency or spectrum shaping:

- reduce (attenuate) unwanted frequencies
- amplify or preserve desired frequencies

18

Motivation for Nonlinear Filtering

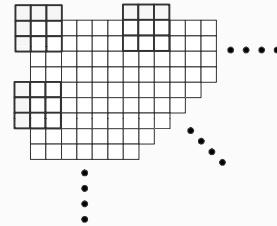
(or nonlinear processing of linear filter responses)

- Generally, nonlinear filtering **cannot** be expressed by linear convolution **nor** by frequency shaping.
- Some approaches **nonlinearly modify** linear responses
- Others **combine filter bank responses nonlinearly**
- We will study these in the important context of **image denoising**.
- The common theme: **nonlinear filters** give you capabilities that linear filters **don't have**.

19

Boundary-of-Image Processing

- As with binary morphological filters: when a window overlaps "empty space" ...



- Pixel replication is assumed: fill the "empty" window slots by the nearest image pixel.

22

MORPHOLOGICAL FILTERS

20

MORPHOLOGICAL FILTERS

- **Recall:** For image \mathbf{I} , window \mathbf{B} , the **windowed set** at (m,n) is:

$$\mathbf{B} \circ \mathbf{I}(m,n) = \{I(m-p, n-q); (p, q) \in \mathbf{B}\} .$$
- A nonlinear filter F on a **windowed set** is a nonlinear function of the pixels covered by the window.
- Denote the **nonlinear filter** F on $\mathbf{B} \circ \mathbf{I}(m,n)$ by

$$J(m,n) = F\{\mathbf{B} \circ \mathbf{I}(m,n)\} = F\{I(m-p, n-q); (p, q) \in \mathbf{B}\}$$
- Performing this at every pixel gives a **filtered image**:

$$\mathbf{J} = F[\mathbf{I}, \mathbf{B}] = [J(m,n); 0 \leq m \leq M-1, 0 \leq n \leq N-1]$$

21

MEDIAN FILTER

- The **median filter** is a nonlinear filtering device that is related to the binary majority filter (Module 2).
- Despite being "automatic" (no design), it is **very effective for image denoising**.
- Despite its simplicity, it has an **interesting theory** that justifies its use.
- The median filter is a special member of several classes of filters, including two studied here: gray-level **morphological filters**, and **order statistic filters**.

23

Order Statistics

- **Definition:** Given a set of numbers

$$X = \{X_1, X_2, \dots, X_{2P+1}\}$$

the **order statistics** (or **OS**) of the set are the same elements reordered from smallest to largest.

- The OS of X are denoted

$$X = \{X_{(1)}, X_{(2)}, \dots, X_{(2P+1)}\}$$

such that

$$X_{(1)} \leq X_{(2)} \leq \dots \leq X_{(2P+1)}$$

24

Special Order Statistics

- In particular:

$$\text{MIN}\{X_1, X_2, \dots, X_{2P+1}\} = X_{(1)}$$

$$\text{MAX}\{X_1, X_2, \dots, X_{2P+1}\} = X_{(2P+1)}$$

$$\text{MED}\{X_1, X_2, \dots, X_{2P+1}\} = X_{(P+1)}$$

- The median MED is the middle value in rank.

25

Example - 1-D Median Filter

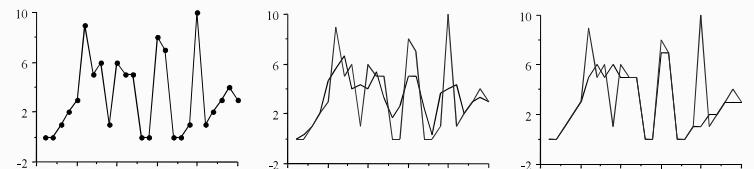


image scan line

AVE[I, B], B = ROW(3)

MED[I, B], B = ROW(3)

- Both filters smooth, but AVE blurs the structures.
- Using MED, the "noise" is smoothed effectively - large noise spikes are eradicated, rather than blurred.
- MED maintains signal structure better - the edges are sharp.

28

Median Filter

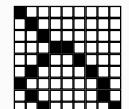
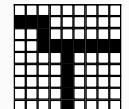
- Given image **I** and window **B**, the median filtered image is:

$$\mathbf{J} = \text{MED}[\mathbf{I}, \mathbf{B}]$$

- Each output is the median of the windowed set:

$$\begin{aligned} J(m,n) &= \text{MED}\{\mathbf{B} \circ \mathbf{I}(m,n)\} \\ &= \text{MED}\{I(m-p, n-q); (p, q) \in \mathbf{B}\} \end{aligned}$$

26



Comments on Window Shape

- B** = SQUARE smooths noise strongly, but may damage image details of interest.
- B** = CROSS reduces these effects, if known that the image contains horizontal/vertical features.
Example: Images of dot-matrix characters.
- If the image contains curvilinear details that are diagonally oriented, then **B** = CROSS can perform poorly.
- Variations:** combine the outputs of multiple CROSS, X-SHAPED, ROW, COL, etc. windows e.g. by taking their MED, AVE, etc. These "detail-preserving filters" work pretty well but the theory is heuristic.

29

Properties of the Median Filter

- The median filter smooths additive white noise.
- The median filter does not degrade edges.
- The median filter is particularly effective for removing large-amplitude noise impulses.

27

Erosion and Dilation

- Given an image **I** and a window **B**:

$$\mathbf{J} = \text{DILATE}[\mathbf{I}, \mathbf{B}]$$

if

$$J(i, j) = \text{MAX}\{\mathbf{B} \circ \mathbf{I}(i, j)\} \quad (\text{local max})$$

and

$$\mathbf{J} = \text{ERODE}[\mathbf{I}, \mathbf{B}]$$

if

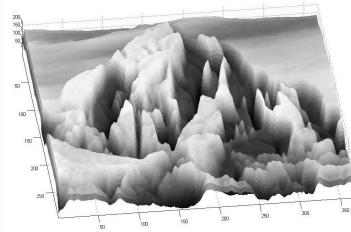
$$J(i, j) = \text{MIN}\{\mathbf{B} \circ \mathbf{I}(i, j)\} \quad (\text{local min})$$

30

Qualitative Properties of Morphological Filters



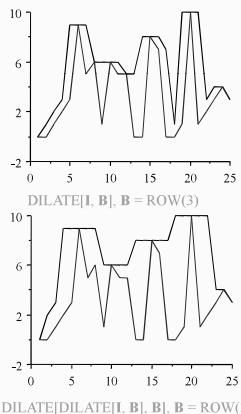
Ansel Adams' Joshua Tree



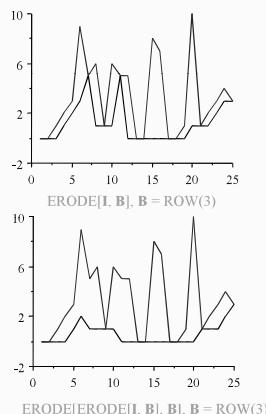
- Gray-level morphological filters effect shape – in this case, the shape of the intensity surface of the image.
- The DILATE (ERODE) filters have the following properties:
 - Increase the size of peaks (valleys)
 - Decrease the size of, or eliminate, valleys (peaks)

31

Dilation and Erosion Examples



Most valleys gone after one pass. All valleys and negative-going impulses gone after two.



Most positive impulses reduced or gone after one pass. After two, all gone.

32

CLOSE and OPEN Filters

- The CLOSE and OPEN filters are defined by:

$$\begin{aligned}\mathbf{J} &= \text{CLOSE}[\mathbf{I}, \mathbf{B}] \\ &= \text{ERODE} [\text{DILATE} [\mathbf{I}, \mathbf{B}], \mathbf{B}]\end{aligned}$$

and

$$\begin{aligned}\mathbf{J} &= \text{OPEN}[\mathbf{I}, \mathbf{B}] \\ &= \text{DILATE} [\text{ERODE} [\mathbf{I}, \mathbf{B}], \mathbf{B}]\end{aligned}$$

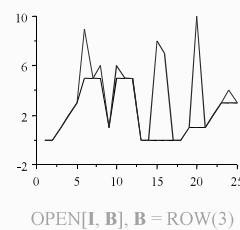
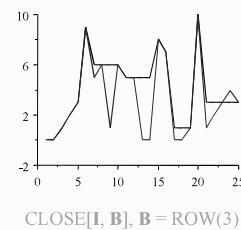
33

Properties of CLOSE and OPEN

- Effective smoothing filters similar to the median filter.
- The CLOSE (OPEN) filter has the following properties:
 - smooths noise
 - preserves edges
 - eradicates negative- (positive-) going impulses
 - leaves the signal at approximately the same level

34

Close and Open Examples



- CLOSE preserves the signal except that the negative-going impulses are gone.
- OPEN preserves the signal except the positive-going impulses are gone.

35

OPEN-CLOSE & CLOSE-OPEN Filters

- The OPEN-CLOSE and CLOSE-OPEN filters:

$$\begin{aligned}\mathbf{J} &= \text{OPEN-CLOSE} [\mathbf{I}, \mathbf{B}] \\ &= \text{OPEN} [\text{CLOSE} [\mathbf{I}, \mathbf{B}], \mathbf{B}]\end{aligned}$$

and

$$\begin{aligned}\mathbf{J} &= \text{CLOSE-OPEN} [\mathbf{I}, \mathbf{B}] \\ &= \text{CLOSE} [\text{OPEN} [\mathbf{I}, \mathbf{B}], \mathbf{B}]\end{aligned}$$

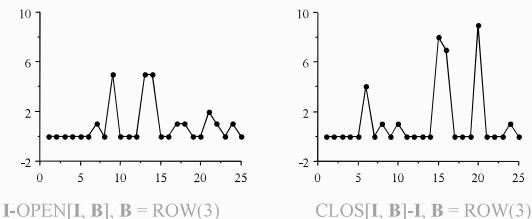
36

OPEN-CLOSE & CLOSE-OPEN

- Effective smoothing filters **very** similar to the median filter. Also very similar to **each another**.
- The CLOSE-OPEN and OPEN-CLOSE filters obey the following:
 - both **smooth noise**
 - both **preserve edges**
 - both **eradicate positive & negative impulses**

37

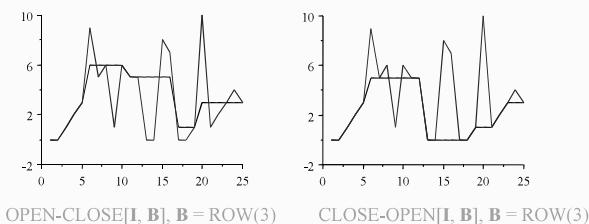
Peak/Valley Detection Examples



- Simple **thresholding** (binarization) of these could be used to determine the peak and valley locations.

40

Close-Open & Open-Close Examples



- Note the **intense smoothing of noise** (especially impulses) and the retention of the global image structure.
- Note the **different possible interpretations** of "correct image structure" under CLOSE-OPEN and OPEN-CLOSE.

38

Morphological Interpretation

- DILATE and ERODE are true morphological filters in the sense discussed in Module 2.
- As are CLOSE, OPEN, CLOSE-OPEN, and OPEN-CLOSE.
- They have the **identical interpretation** if we regard
 - **I** as a 3-D binary image with value '1' below its "plot" and '0' above its "plot"
 - **B** as a floating 2-D structuring element

41

Application: Peak/Valley Detection

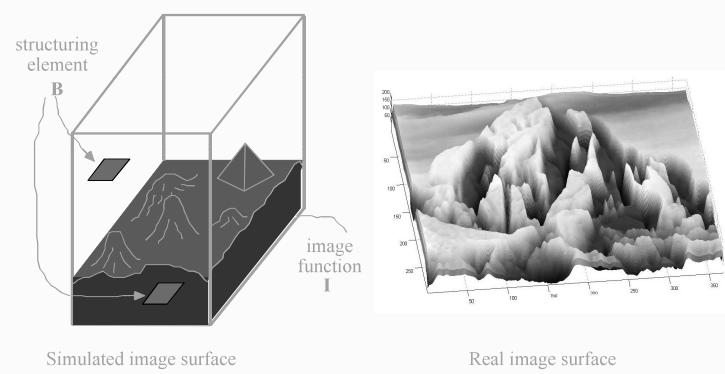
- Suppose we wish to find **bright targets or dark spots**.
- Difference** the OPENed and CLOSEd image **I**:

$$J_{\text{peak}} = I - \text{OPEN}[I, B]$$

$$J_{\text{valley}} = \text{CLOSE}[I, B] - I$$

- As we'd expect, these operations highlight the **peaks and valleys** that occur.

39



- Leads to **fast** all-Boolean algorithms & architectures.

42

Morphological Interpretation

ERODE:

- When **B** lies **above I**, the AND of all it touches is '0'.
- When **B** lies **below I**, the AND of all it touches is '1'.
- Whenever **B** crosses the boundary of **I**, the AND of all it touches is '0'.

DILATE:

- When **B** lies **above I**, the OR of all it touches is '0'.
- When **B** lies **below I**, the OR of all it touches is '1'.
- Whenever **B** crosses the boundary of **I**, the OR of all it touches is '1'.

43

Threshold Decomposition Example

$x[n] =$	5	2	7	0	3	4
$x_{\geq 7}[n] =$	0	0	1	0	0	0
$x_{\geq 6}[n] =$	0	0	1	0	0	0
$x_{\geq 5}[n] =$	1	0	1	0	0	0
$x_{\geq 4}[n] =$	1	0	1	0	0	1
$x_{\geq 3}[n] =$	1	0	1	0	1	1
$x_{\geq 2}[n] =$	1	1	1	0	1	1
$x_{\geq 1}[n] =$	1	1	1	0	1	1
$x_{\geq 0}[n] =$	1	1	1	1	1	1

- Let $B=3$, so that $m = 2^B = 8$.
- The pixels/samples take values in $[0, 7]$.
- Each threshold signal is **binary**.
- In each column, notice that there must be a **stack** of 0's on top of a **stack** of 1's.
- This is called the **stacking property**.

46

Relating Grayscale & Binary Morphological Filters

- For many morphological filters, there is a precise relationship between the grayscale and binary versions.
- As usual, we will consider the 1D case to develop an understanding of this.
- Here, the term **m-ary** is a synonym for **grayscale**.
 - Implies pixels/samples take one of m distinct values.
 - For B -bit pixels/samples, $m = 2^B$ ($= K$ in Modules 1-3).
- Several concepts involved:
 - Threshold decomposition and reconstruction.
 - The **stacking property**.
 - The notion of a **positive boolean function** or PBF.

44

Threshold Reconstruction

$x[n] =$	5	2	7	0	3	4
$x_{\geq 7}[n] =$	0	0	1	0	0	0
$x_{\geq 6}[n] =$	0	0	1	0	0	0
$x_{\geq 5}[n] =$	1	0	1	0	0	0
$x_{\geq 4}[n] =$	1	0	1	0	0	1
$x_{\geq 3}[n] =$	1	0	1	0	1	1
$x_{\geq 2}[n] =$	1	1	1	0	1	1
$x_{\geq 1}[n] =$	1	1	1	0	1	1
$x_{\geq 0}[n] =$	1	1	1	1	1	1

- Note that $x[n]$ can be reconstructed from any set of threshold signals that obey the stacking property:

$$x[n] = \max \{ \tau \mid x_{\geq \tau}[n] = 1 \}.$$
- This is shown by the red line to the left.

47

Threshold Decomposition

- Let $x[n]$ be an m -ary signal.
- Define the binary threshold signal

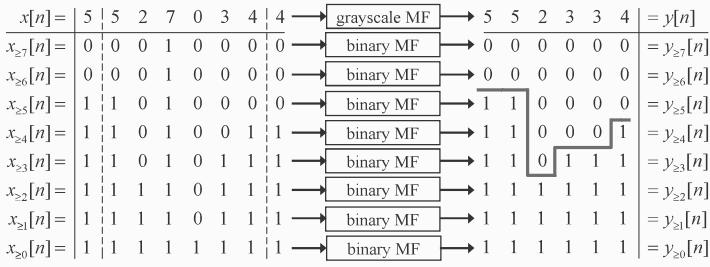
$$x_{\geq \tau}[n] = \begin{cases} 1, & \text{for all } n \text{ s.t. } x[n] \geq \tau, \\ 0, & \text{for all } n \text{ s.t. } x[n] < \tau. \end{cases}$$
- The **threshold decomposition** of $x[n]$ is defined by the set of threshold signals $x_{\geq \tau}[n]$ for $\tau = [0, 1, 2, \dots, m-1]$.

45

Median Filtering by Threshold Decomposition

- On the next page, we will apply a **grayscale** 3-point median filter to $x[n]$.
- In parallel, we will apply **binary** 3-point median filters to each of the threshold signals $x_{\geq \tau}[n]$.
- As is typical for morphological and order statistic filters, we will handle the edge effects by replication.

48



- The result of applying the grayscale median filter to the m-ary signal is the **same** as the result of applying the binary median filters to the threshold signals!

49

When Does this Work?

- Amazing!
- Does this work for other types of filters?
- The answer is YES.
 - It works for any filter that preserves the stacking property at the outputs of the binary filters.
 - Filters that preserve the stacking property are called **stack filters**. They are a **very large** class of nonlinear filters.
- Fact: the stacking property is preserved if the binary version of the filter is a positive boolean function (PBF).
- Def: a PBF is a binary function where the minimum sum of products form contains no complemented variables.
- Ex: 3-pt binary MF: $y = x_1x_2 + x_1x_3 + x_2x_3$.

50

Stack Filters

- Stack filters are important for two main reasons:
 - There is a theoretical framework for designing stack filters to minimize the mean absolute error (MAE) in the output signal.
 - There are techniques for elegant fast hardware implementations.
- Examples of 3-point PBF's that define stack filters:
 - MED: $y = x_1x_2 + x_1x_3 + x_2x_3$
 - DILATE/MAX: $y = x_1 + x_2 + x_3$
 - ERODE/MIN: $y = x_1x_2x_3$

51

Morphological Filters as Stack Filters

- Because ERODE and DILATE are stack filters, so are OPEN, CLOSE, Close-Open, and Open-Close.
- This is the “detailed version” of what is meant by the figure on page 6.42.
- Ex: OPEN($x[n], \mathbf{B}$) with $\mathbf{B}=\text{ROW}(3)$:
 - Because $\text{OPEN}(x[n], \mathbf{B}) = \text{DILATE}[\text{ERODE}(x[n], \mathbf{B})]$, this is actually a 5-point filtering operation.
 - The PBF for the binary filters is given by
$$y = x_1x_2x_3 + x_2x_3x_4 + x_3x_4x_5$$
- It takes a lot more work to derive the PBFs for CLOSE, Close-Open, and Open-Close.

52

Fast Implementations

- Fast implementations of grayscale morphological and order statistic filters are challenging because they require **sorting** the pixels in the windowed set.
- But the binary versions of these filters can be implemented with boolean logic: AND, OR, XNOR, which do not require sorting.
- For stack filters, fast integrated circuits can be designed to:
 - perform the threshold decomposition,
 - filter the binary threshold signals,
 - reconstruct the output from the filtered threshold signals.

53

Fast Sorting

- Sorting is an expensive operation, especially for large window sizes.
- Let M be the number of pixels in the windowed set.
- Time complexity of common sorting algorithms:
 - $\mathcal{O}(M^2)$: quicksort, insertion sort, selection sort...
 - $\mathcal{O}(M \log M)$: merge sort, heapsort, introsort...
- In hardware, space complexity can often be traded one-for-one for time complexity.
- If the binary version of the filter is a PBF, then there are **very fast** bit-serial realizations that require only B clock cycles.

54

Bit-Serial Median Filtering

- This algorithm is due to Delman (Proc. SPIE, vol. 298, 1981, pp. 184-188).
- For an image \mathbf{I} , suppose we want to implement the filter $\mathbf{J} = \text{MED}(\mathbf{I}, \mathbf{B})$ with $\mathbf{B} = \text{CROSS}(5)$.
- Suppose $\mathbf{B}=3$ -bit pixels and let the windowed set be

$$\mathbf{B} \odot \mathbf{I}(i, j) = \{2, 0, 1, 5, 6\}.$$

- The median is 2.
- With good sorting hardware,
 - insertion sort requires **at least 25 clock cycles**.
 - merge sort requires **at least 15 clock cycles**.
- The bit-serial algorithm finds the median in only three clock cycles!

55

Third Clock Cycle

Second Change

0	1	0
0	0	0
0	0	0
1	1	1
1	1	1

Majority → 010

- Take the majority of the lo-order bits. This is the lo-order bit of the median (a zero in this case).
- We are done! The median is found in only three clock cycles!
- This approach was generalized by Chen to develop fast bit-serial realizations for **any** stack filter (IEEE Trans. Circuits, Syst., vol. 36, no. 6, June 1989, pp. 785-794).

- For B -bit input pixels, each output pixel is computed in only B clock cycles.

58

First Clock Cycle

Original Data

0	1	0
0	0	0
0	0	1
1	0	1
1	1	0

Majority → 0--

First Change

0	1	0
0	0	0
0	0	1
1	1	1
1	1	1

56

Second Clock Cycle

First Change

0	1	0
0	0	0
0	0	1
1	1	1
1	1	1

Majority → 01-

Second Change

0	1	0
0	0	0
0	0	0
1	1	1
1	1	1

57

ORDER STATIC FILTERS

- Take the majority of the hi-order bits. This is the hi-order bit of the median (a zero in this case).
- Any pixel with a hi-order bit of one is **bigger** than the median.
 - Changing it to **all ones** will **not** change the median.
- This is the **First Change**, which is clocked into the registers by the rising edge of the second clock cycle.

56

- Take the majority of the second bits. This is the second bit of the median (a one in this case).
- Any pixel with a second bit of zero is **smaller** than the median.
 - Changing it to **all zeros** will **not** change the median.
- This is the **Second Change**, which is clocked into the registers by the rising edge of the third clock cycle.

ORDER STATIC FILTERS

- Recall the OS of a set $\mathbf{X} = \{X_1, \dots, X_{2p+1}\}$:

$$\mathbf{X}_{\text{OS}} = \{X_{(1)}, \dots, X_{(2p+1)}\}$$

such that

$$X_{(1)} \leq \dots \leq X_{(2p+1)}$$

- Define a linear combination of order statistics

$$\sum_{p=1}^{2p+1} A_p X_{(p)} = \mathbf{A}^T \mathbf{X}_{\text{OS}}$$

where $\mathbf{A} = [A_1, \dots, A_{2p+1}]^T$. This defines a class of nonlinear filters called **order statistic (OS) filters**.

60

OS Filter Coefficients

- The filter weights are assumed to sum to 1:

$$\sum_{p=1}^{2P+1} A_p = A^T e = 1$$

where $e = [1, 1, \dots, 1]^T$. Then if $\mathbf{X} = c \cdot e$ is constant then $A^T \mathbf{X}_{OS} = c$, and the filter is then level-preserving – like a low-pass linear filter.

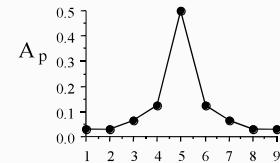
- The OS coefficients are assumed symmetric:

$$A_i = A_{2P+2-i} \text{ for } 1 \leq i \leq 2P+1$$

61

Median-Like OS Filters

- Generally, if the coefficients A are concentrated near the middle:



then $OS_A[\mathbf{I}, \mathbf{B}]$ is much like a median filter:

- preserve edges well
- smooth noise
- reduce impulses

BUT

- will not "streak" or "blotch" as much
- will suppress non-impulse noise better (more later)

64

Defining OS filters

- Given an image \mathbf{I} , a window \mathbf{B} , and a coefficient set \mathbf{A} , the OS filter with coefficients \mathbf{A} is defined by:

$$\mathbf{J} = OS_A[\mathbf{I}, \mathbf{B}]$$

if

$$J(m,n) = A^T \{B \circ I(m,n)\}_{OS}$$

where

$$\{B \circ I(m,n)\}_{OS}$$

are the OS of

$$\{B \circ I(m,n)\}.$$

- These are just the ordered values of the pixels covered by the window \mathbf{B} when it's centered at (m,n) .

62

General Properties of OS Filters

- The behavior of a filter $OS_A[\mathbf{I}, \mathbf{B}]$ is largely determined by its coefficients \mathbf{A} .
- Some important OS filters $OS_A[\mathbf{J}, \mathbf{B}]$ are:

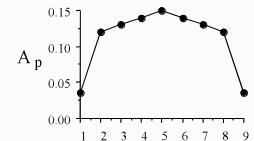
- $\mathbf{A} = [0, \dots, 0, 1, 0, \dots, 0]^T$ - median filter

- $\mathbf{A} = \left[\frac{1}{2P+1}, \dots, \frac{1}{2P+1} \right]^T$ - average filter

63

Average-Like OS Filters

- Generally, if the coefficients are widely distributed:



then $OS_A[\mathbf{I}, \mathbf{B}]$ will act much like an average filter:

- preserve edges poorly (better than AVE)
- smooth noise
- blur impulses (not as badly as AVE)

- We will now explore the utility of these filters relative to some of the noise models we have looked at.

65

OS Filter Design Criterion

- The goal of a smoothing filter can be stated as reduce the noise variance.
- OS filters are best characterized by their ability to reduce noise variance.
- For certain well-known noise PMFs, specific choices of the coefficients \mathbf{A} of $OS_A[\mathbf{J}, \mathbf{B}]$ result using the variance reduction criterion.

66

Gaussian Noise – Average Filter

- The average filter is **best** among all OS filters at reducing Gaussian noise variance. It is also the best linear filter. In fact, it is the best filter of any kind!
- If \mathbf{B} contains $2P+1$ elements, then $\text{AVE}[\mathbf{J}, \mathbf{B}]$ reduces (divides) the variance of Gaussian noise by a factor $\frac{1}{2P+1}$
- In fact, $\text{AVE}[\mathbf{J}, \mathbf{B}]$ reduces the variance of **any white noise** by a factor $\frac{1}{2P+1}$
- This is pretty good, but for non-Gaussian PMFs other OS filters can do much better.
- Major caveat:** The above does not account for the effect on the signal. Average filtering blurs the signal as it reduces noise, so there exists a **tradeoff** between noise reduction and signal blurring.

67

Robust OS Filter Coefficients

- A fairly sophisticated **statistical analysis** yields the **best OS_A** filter for **specific noise and image models**.
- Fortunately, a great property of **certain OS filters** is the fact that they are **robust**.
- A filter is **robust** if it gives **close to optimal performance** for a **wide variety** of noise types.
- OS filters OS_A have been shown to be **remarkably robust** for coefficients \mathbf{A} that resemble a cross between MED and AVE.

70

Laplacian Noise – Median Filter

- The median filter MED $[\mathbf{J}, \mathbf{B}]$ is the asymptotically **best** OS filter for reducing the variance of Laplacian noise (as window size increases).
- If \mathbf{B} contains $2P+1$ elements, then MED $[\mathbf{J}, \mathbf{B}]$ reduces Laplacian noise variance by a factor of (approximately) $2 \cdot (2P+1)$ - quite an **improvement!**
- MED $[\mathbf{J}, \mathbf{B}]$ is **fantastic** for removing salt-and-pepper noise. No better filter!
- If the noise is Gaussian, then MED $[\mathbf{J}, \mathbf{B}]$ reduces the noise variance by a factor of (approximately) only $(0.75) \cdot (2P+1)$ - not an improvement!
- Extended Caveat:** The median filter is noted to **not blur images**, even as the window size increases.

68

Trimmed Mean Filters

- A simple robust OS filter class is the **trimmed mean OS filters** TM_Q $[\mathbf{J}, \mathbf{B}]$ with coefficients
$$\mathbf{A} = \left[\underbrace{0, \dots, 0}_{Q}, \underbrace{\frac{1}{2(P-Q)+1}, \dots, \frac{1}{2(P-Q)+1}}_{2(P-Q)+1}, \underbrace{0, \dots, 0}_{Q} \right]^T$$
where $Q < P$ (usually $Q \approx P/2$).
- Similar to AVE since the innermost $2(Q-P)+1$ OS are averaged. If $Q = 0$, then TM_Q $[\mathbf{J}, \mathbf{B}] = \text{AVE}[\mathbf{J}, \mathbf{B}]$.
- Similar to MED since **only** inner OS are averaged, the rest of the OS being "trimmed." Indeed, if $Q = P$, then TM_Q $[\mathbf{J}, \mathbf{B}] = \text{MED}[\mathbf{J}, \mathbf{B}]$.

71

Median Filter vs. Average Filter

- Owing to its very important **edge-preserving property**, MED is generally regarded as **superior** to AVE (or any other linear filter!) for **reducing white noise in images**.
- However, as with any application, care should be taken in analyzing the situation rather than blindly applying a filter.
- Both MED and AVE are used **far too often** in situations where a design process might be **more advantageous**.

69

Trimmed Mean Filter Performance

- Trimmed mean filters **smooth noise well** (including impulse and Gaussian noise) and **preserve edges well**.
- For **both Gaussian and Laplacian noise**, TM_{P/2} reduces the variance by a factor $> (0.9) \cdot \text{OPTIMAL}$

where **OPTIMAL** is $(2P+1)$ for Gaussian noise and $2 \cdot (2P+1)$ for Laplacian noise.

72

Generalized Gaussian PDF

- The generalized gaussian PDF is used to model wide behaviors of noise:

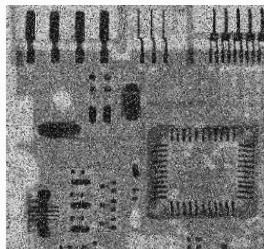
$$f_\gamma(x) = K e^{-|x/\sigma|^\gamma} \quad -\infty < x < \infty$$

- For $\gamma < 1$, $f_\gamma(x)$ is **very heavy-tailed**.
- For $\gamma = 1$, $f_1(x)$ is **laplacian**.
- For $\gamma = 2$, $f_2(x)$ is **gaussian**.
- As $\gamma \rightarrow \infty$, $f_\infty(x)$ is **uniform**.
- For a wide range of values of γ

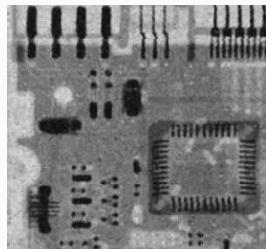
$TM_{P/2} > (0.9) \cdot \text{OPTIMAL!}$

73

Example – AWGN / S&P Noise



AWGN and S&P Noise



50% Trimmed Mean

Time-consuming!
Good examples:
 $N=3$, SQUARE, $P=2$
 $N = 5$, CROSS, $P=3$
AWGN with S&P

74

BILATERAL FILTER

- Another principled approach to **image denoising**.
- Like OS filters, it **modifies linear filtering**.
- Like OS filters, it seeks to smooth while **retaining (edge) structures**.

76

BILATERAL FILTER

Bilateral Filter Concept

- Observation:** 2-D linear filtering is a method of **weighting by spatial distances** [$i = (i, j)$, $m = (m, n)$]:

$$J_G(i) = \sum_m I(m) G(\|m - i\|)$$

- If linear filter G is **isotropic**, e.g., a 2-D gaussian LPF*

$$G(i, j) = K_G \exp[-(i^2 + j^2) / \sigma_G^2]$$

then $\|m - i\| = \sqrt{(m_i - i)^2 + (m_j - j)^2}$ is **Euclidean distance**.

*The value of K_G is such that G has unit volume.

77

Bilateral Filter Concept

- New idea:** Instead of weighting by spatial distance, weight by **luminance similarity**

$$J_H(i) = \sum_m I(m) H[I(m) - I(i)]$$

- NOT** a linear filter operation. H could be a 1-D gaussian weighting function*

$$H(i) = K_H \exp[-i^2 / \sigma_H^2]$$

- Not very useful by itself.**

*The value of K_H is such that H has unit area.

78

75

Bilateral Filter Definition

- Combine spatial distance weighting with luminance similarity weighting:

$$J(\mathbf{i}) = \sum_{\mathbf{m}} I(\mathbf{m}) G(\|\mathbf{m} - \mathbf{i}\|) H[I(\mathbf{m}) - I(\mathbf{i})]$$

- Computation is straightforward and about twice as expensive as just spatial linear filtering.
 - However, this can be sped up by pre-computing all values of H for every pair of luminances, and storing them in a look-up table (LUT).

79

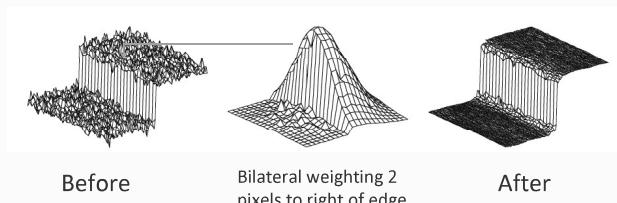
Comments on Bilateral Filter

- Similar to median filter, emphasizes luminances close in value to current pixel.
 - Does not have a statistical (or other) design basis, so best used with care, e.g., retouching or low-level noise smoothing.
 - The idea of doing linear filtering that is decoupled near edges is very powerful.
 - This is also the theme of the next method.

82

Example

- Noisy edge filtered by bilateral filter (Tomasi, ICCV 1998).
 - Uses gaussian LPF and gaussian similarity



80

NON-LOCAL (NL) MEANS

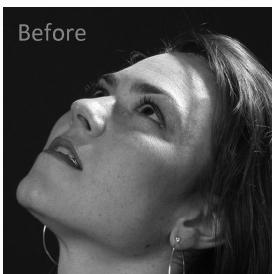
83

Color Example

- Can extend to color using 3-D similarity weighting.
 - If $\mathbf{I} = [\mathbf{R}, \mathbf{G}, \mathbf{B}]$, then use

$$H[R(m)-R(i), G(m)-G(i), B(m)-B(i)]$$

where $H(i, j, k) = K_H \exp\left[-(i^2 + j^2 + k^2)/\sigma_H^2\right]$



81

NL Means Concept

- Idea: Estimate the current pixel luminance as a weighted average of all pixel luminances
 - The weights are decided by neighborhood luminance similarity.
 - The weight changes from pixel to pixel.
 - In all the following, $i = (i, j)$, $m = (m, n)$, $p = (p, q)$.
 - The method is expensive but effective.

84

NL Means Concept

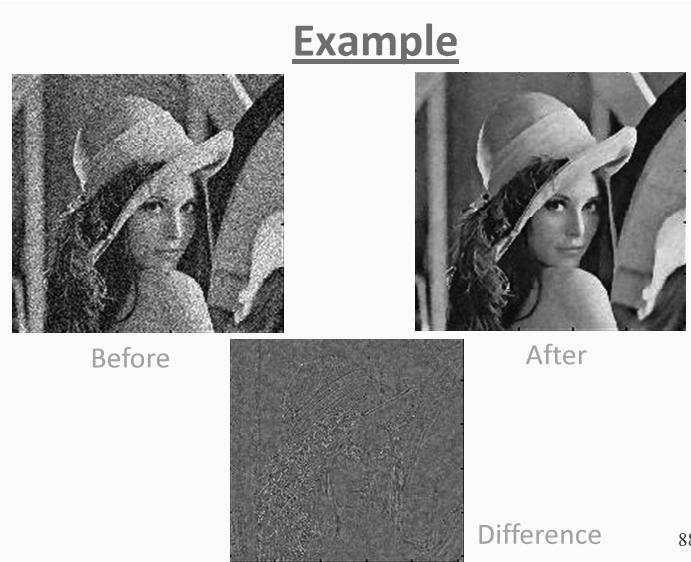
- Given a window \mathbf{B} , compute the luminance similarity of every windowed set $\mathbf{B} \circ \mathbf{I}(i)$ with every other windowed set $\mathbf{B} \circ \mathbf{I}(m)$:

$$W(i, m) = K_w \exp \left[-\|\mathbf{B} \circ \mathbf{I}(i) - \mathbf{B} \circ \mathbf{I}(m)\| / \sigma_w^2 \right]$$

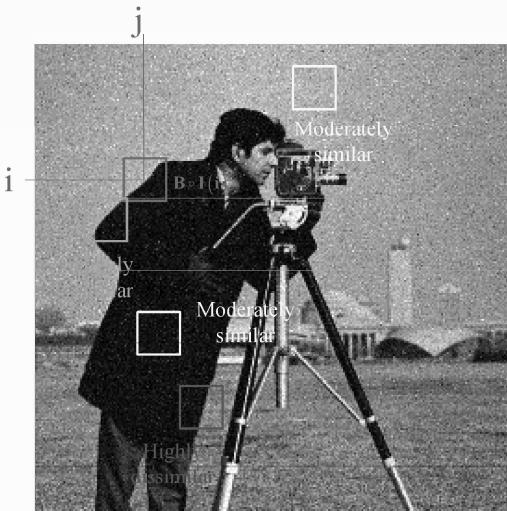
where

$$\begin{aligned} \|\mathbf{B} \circ \mathbf{I}(i) - \mathbf{B} \circ \mathbf{I}(m)\| &= \sum_{p \in \mathbf{B}} [I(i-p) - I(m-p)]^2 \\ &= \sum_{(p,q) \in \mathbf{B}} [I(i-p, j-q) - I(m-p, n-q)]^2 \end{aligned}$$

85

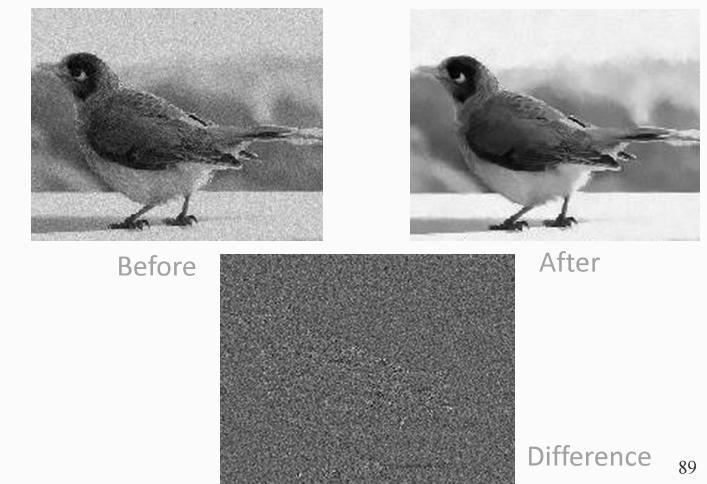


88



86

Example



89

NL Means Definition

- Given a window \mathbf{B} , and the weighting function W just defined, the **NL-means** filtered image \mathbf{I} is

$$\mathbf{J}(i) = \sum_m W(m, i) I(m)$$

- Thus each pixel value is replaced by a weighted sum of all luminances, where the weight increases with neighborhood similarity.

87

Comments

- NL-Means** is very similar in spirit to **frame averaging**.
- The main drawback is the large computation / search.
- It can be modified to compare/search less of the image.
- It works best when the image contains a lot of redundancy (periodic, large smooth regions, textures).
- It can fail at unique image patches.

90

ANISOTROPIC DIFFUSION

91

The Diffusion Equation

- **Continuous Formulation** - a system of partial differential equations:

$$\frac{\partial \mathbf{I}}{\partial t} = \operatorname{div}[\mathbf{c} \nabla \mathbf{I}] = c_{EW} \frac{\partial^2 I(x,y)}{\partial x^2} + c_{NS} \frac{\partial^2 I(x,y)}{\partial y^2} \quad \mathbf{c} = \begin{bmatrix} c_{EW} & 0 \\ 0 & c_{NS} \end{bmatrix}$$

where \mathbf{I} is the image, \mathbf{c} is a 2×2 matrix of diffusion coefficients, and div is the divergence operator

- The discrete formulation is much easier to understand...

Note: Recall

$$\operatorname{div}(\mathbf{F}) = \frac{\partial f_1}{\partial x} + \frac{\partial f_2}{\partial y} \text{ where } \mathbf{F} = \begin{bmatrix} f_1 \\ f_2 \end{bmatrix}$$

94

ANISOTROPIC DIFFUSION

- A powerful approach to both image denoising and edge detection.
- Fact: Gaussian blur or smoothing is analogous to physical diffusion (e.g., of heat).
- In fact the gaussian function is a solution to a PDE called the **diffusion equation**.

92

Concept of Anisotropic Diffusion

- Diffuse (smooth image pixels) where the gradient magnitude is small.
- Inhibit diffusion where gradient magnitude is large.
- **Anisotropic** since smoothing depends on direction.
- Convolution with a Gaussian is **isotropic** diffusion.
- **Rationale:**
 - Smooth image without destroying image detail
 - Inhibit smoothing across edges (inter-region smoothing)
 - Encourage smoothing between edges (intra-region smoothing)

93

Approximating Derivatives

$$\frac{\partial \mathbf{I}}{\partial t} \rightarrow I_{t+1}(m,n) - I_t(m,n)$$

where time t will be iteration number of the diffusion process.

$$\frac{\partial^2 I(x,y)}{\partial x^2} \rightarrow I(m, n+1) - 2I(m, n) + I(m, n-1)$$

$$\frac{\partial^2 I(x,y)}{\partial y^2} \rightarrow I(m+1, n) - 2I(m, n) + I(m-1, n)$$

which approximates twice derivatives by twice differences.

95

Diffusion Equation: Discrete Form

- Iterative approximation using differences:

$$I_{t+1}(m,n) = I_t(m,n) + \frac{1}{4} \left[c_{NS}(m,n) \nabla_{NS}^{(2)} I_t(m,n) + c_{EW}(m,n) \nabla_{EW}^{(2)} I_t(m,n) \right]$$

where ...

$$\nabla_{NS}^{(2)} I_t(m,n) = I(m,n+1) - 2I(m,n) + I(m,n-1)$$

$$\nabla_{EW}^{(2)} I_t(m,n) = I(m+1,n) - 2I(m,n) + I(m-1,n)$$

Note: If $c_{NS} = c_{EW} = c$, then the diffusion equation becomes the *heat equation* $\frac{\partial \mathbf{I}}{\partial t} = c \nabla^2 \mathbf{I}$

96

Diffusion Coefficients

- Selection of the diffusion coefficients c_{NS} , c_{EW} is important. Usually $c_{NS} = c_{EW}$.
- The idea is to inhibit smoothing over edges, encourage smoothing on non-edge image regions. So c_{NS} , c_{EW} should be edge-sensitive.
- At each pixel $I(i, j)$, diffusion proceeds in a given direction according to the diffusion coefficient corresponding to that direction and that location.

97

Diffusion Coefficients

- One popular form

$$c_d(m, n) = \exp\left\{-\left[\frac{\nabla_d I(m, n)}{k}\right]^2\right\} \quad d = NS, EW$$

where k is an edge threshold – as the gradient increases, c_d rapidly falls.

- Properties of $c_d(m, n)$:
 - approaches 0 near edges – **disenabling diffusion**
 - approaches 1 over smooth areas – **enabling diffusion**

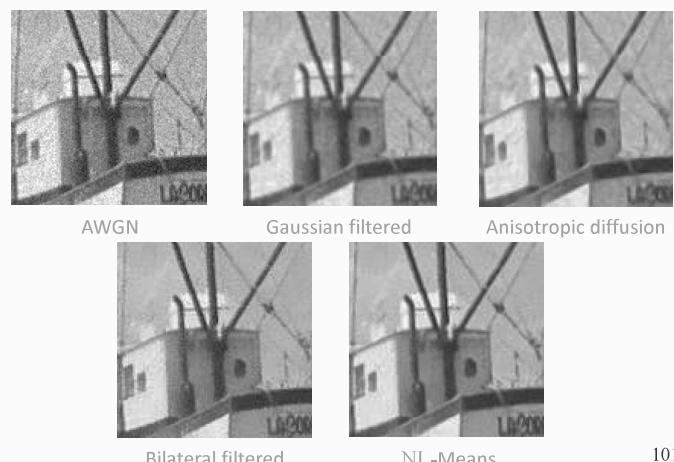
98

Comments on Anisotropic Diffusion

- Really a method of alternating edge detection with **edge-decoupled smoothing**.
- Number of iterations controls degree of smoothing (scale)
- Edge detection applied to the smoothed image can be quite effective.

100

Comparison



101

Anisotropic Diffusion Example



Laplacian Noise



8 iterations of
anisotropic diffusion

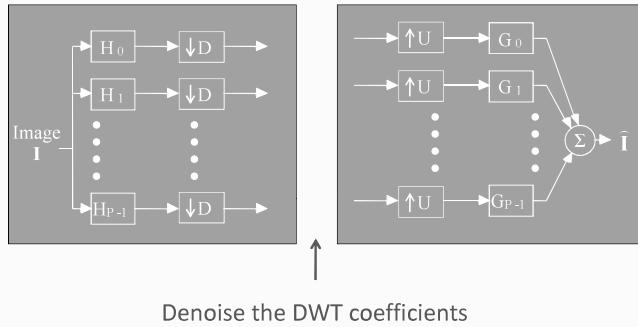
99

WAVELET SHRINKAGE

102

Wavelet Shrinkage

Perfect reconstruction analysis/synthesis filterbank or DWT



103

Wavelet Soft Thresholding

- Two methods of thresholding:

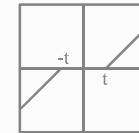
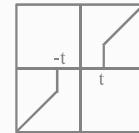
- Hard threshold:** signal zeroed if below threshold
- Soft threshold:** where in addition to hard threshold, the threshold is subtracted if above threshold.

Hard threshold

$$y_p^t(i) = \begin{cases} y_p(i) & \text{if } |y_p(i)| > t \\ 0 & \text{if } |y_p(i)| \leq t \end{cases}$$

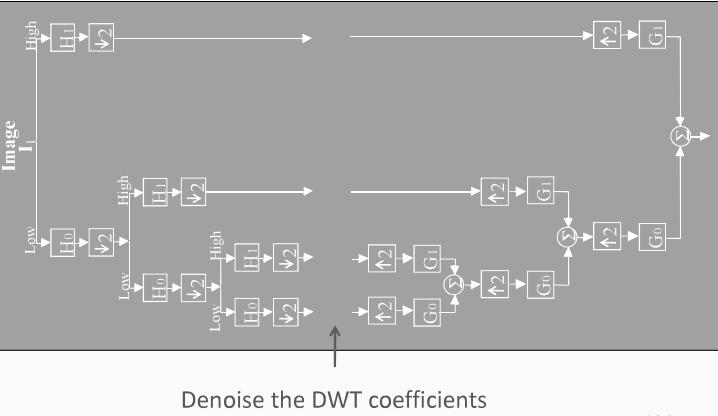
Soft threshold

$$y_p^t(i) = \text{sgn}[y_p(i)] \max\{0, |y_p(i)| - t\}$$



106

Wavelet Shrinkage



104

Wavelet Shrinkage

- Soft thresholding** is much more effective. It “shrinks” the wavelet coefficients.
- Works since additive **noise increases coefficient energy**. Soft thresholding reduces the energy.
- Threshold can be chosen to **optimize** many criteria such as MSE, Bayesian risk, etc.

107

Wavelet Shrinkage

- The process of denoising is accomplished simply by thresholding the DWT coefficients (filter responses).
- A threshold is applied ($p \neq 0$ since baseband signal is smooth already):

$$H_p \rightarrow \downarrow D \rightarrow y_p(i) \rightarrow t \rightarrow y_p^t(i)$$

105

Wavelet Shrinkage

- MMSE threshold (“SureShrink”)**

$$t = \sigma \sqrt{2 \log NM}$$

where σ^2 = estimated noise variance, and NM = # pixels in the image.

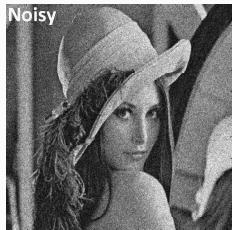
- Bayesian threshold (an SNR) (“BayesShrink”)**

$$t = \sigma_{\text{signal}} / \sigma_{\text{noise}}$$

is estimated for each filter response (band), hence **adaptive**. When SNR is large, little thresholding. When large, heavy thresholding.

108

Wavelet Shrinkage Examples



BM3D

109

112

DWT Used

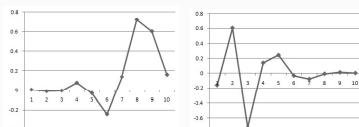
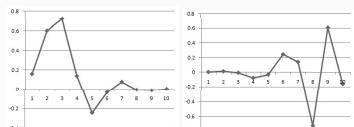
- There is no clear choice (yet) on wavelet filters to use.
- Here we used a 4-level DWT with the “Daubechies 10” orthogonal wavelet (D10)

Analysis LP

```
H_d(0) = 0.1601023980
H_d(1) = 0.6038292698
H_d(2) = 0.7243085284
H_d(3) = 0.1384281459
H_d(4) = -0.2422948871
H_d(5) = -0.0322448696
H_d(6) = 0.0775714938
H_d(7) = -0.0062414902
H_d(8) = -0.0125807520
H_d(9) = 0.0033357253
```

Analysis HP

```
H_1(0) = 0.0033357253
H_1(1) = 0.0125807520
H_1(2) = -0.0062414902
H_1(3) = -0.0775714938
H_1(4) = -0.0322448696
H_1(5) = 0.2422948871
H_1(6) = 0.1384281459
H_1(7) = -0.7243085284
H_1(8) = 0.6038292698
H_1(9) = -0.1601023980
```

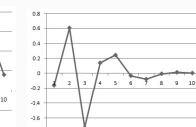


Synthesis LP

```
G_d(0) = 0.0033357253
G_d(1) = -0.0125807520
G_d(2) = -0.0062414902
G_d(3) = 0.0775714938
G_d(4) = -0.0322448696
G_d(5) = -0.2422948871
G_d(6) = 0.1384281459
G_d(7) = 0.7243085284
G_d(8) = -0.6038292698
G_d(9) = -0.1601023980
```

Synthesis HP

```
G_1(0) = -0.1601023980
G_1(1) = 0.6038292698
G_1(2) = -0.7243085284
G_1(3) = 0.1384281459
G_1(4) = 0.2422948871
G_1(5) = -0.0322448696
G_1(6) = -0.0775714938
G_1(7) = -0.0062414902
G_1(8) = 0.0125807520
G_1(9) = 0.0033357253
```



Overview

- Quite a **complicated algorithm** – too much! Uses **many** of the ideas we have covered
- Image broken into **blocks**, blocks then matched by **similarity** into groups.
- Estimates are **collaboratively formed** for each **block** within each group.
- This is done by a transformation of the group and a **shrinkage process**.
- Perhaps the best denoising algorithm extant.

Dabov, Foi, Katkovnik, Egiazarian, IEEE Trans on Image Processing, Aug 2007.

113

Comments

- Since DWTs can be computed **very fast**, so can soft threshold based denoising.
- The results are **optimal in wavelet space** and the appearance is **reasonably good**.

Examples



111

114

HOMOMORPHIC FILTERING

- Assume a **multiplicative white noise model**:

$$\mathbf{J} = \mathbf{I} \otimes \mathbf{N}$$
 where \mathbf{I} is the original image and \mathbf{N} is a **white noise image**.
- Thus

$$J(m,n) = I(m,n)N(m,n)$$
 for $0 \leq m \leq M-1, 0 \leq n \leq N-1$.
- Direct linear filtering ineffective, since the noise / image spectra not added.

115

Logarithmic Point Operation

- Idea is similar to log point operations used for histogram improvement: de-emphasize dominant bright image pixels.
- More than that! Since $\log(a \cdot b) = \log(a) + \log(b)$, then

$$\log [J(m,n)] = \log [I(m,n)] + \log [N(m,n)].$$
- So: the point operation converts the problem into that of smoothing

$$\mathbf{J}' = \mathbf{I}' + \mathbf{N}'$$
 where
 $I'(m,n) = \log [I(m,n)], N'(m,n) = \log [N(m,n)], J'(m,n) = \log [J(m,n)]$
- Importantly, \mathbf{N}' is still a white noise image.
- This is just the additive white noise problem studied earlier!

118

Multiplicative Noise

- Multiplicative noise is always **positive** since the sensed image intensities must be.
- Multiplicative noise often has an **exponential PMF** in practical applications, such as **radar, laser-based imaging, electron microscopy**, etc.
- Synthetic aperture radar** images are an excellent example.
- Multiplicative noise appears **much worse** in bright image regions (since it multiplies the gray levels) and can be **hardly noticeable** in dark regions.

116

Filtering

- The filtering problem here is very similar to the linear and nonlinear ones studied earlier in this Module and in Module 5.
- Depending on the noise statistics, AVE, MED, OS_A, OPEN-CLOSE, CLOSE-OPEN or other linear or nonlinear filter may be used.
- The **theoretical performance** of these various filters in homomorphic systems still remains an open question.
- Most will work reasonably well; in particular MED has been observed to be effective if the noise is exponential and the image contains edges.
- The objective is of course to produce a filtered result

$$F[\mathbf{J}', \mathbf{B}] \approx \mathbf{I}'$$

119

Homomorphic Approach

- Homomorphic filtering is succinctly expressed by the **following diagram**:



- We'll study each step:

117

Exponential Point Operation

- If the filtering operation succeeds:

$$F[\mathbf{J}', \mathbf{B}] \approx \mathbf{I}'$$

then the output image \mathbf{K} will approximate

$$\begin{aligned}
 K(i,j) &\approx \exp[I'(i,j)] \\
 &= I(i,j)
 \end{aligned}$$

120

Comments

- We shall now study **some BIG topics – image quality and compression... onward to Module 7.**