

Nonlinear Filtering for Noise Smoothing

- **Frequency Domain Method for Noise Smoothing**
 - **Example:** Picture with lines on it. Can we get rid of lines?
 - ⇒ Think of the image as desired image + noise (lines).



- **Consider working in transformed domain: take DTFT or DFT.**
- **Exploit separability of noise and image.**

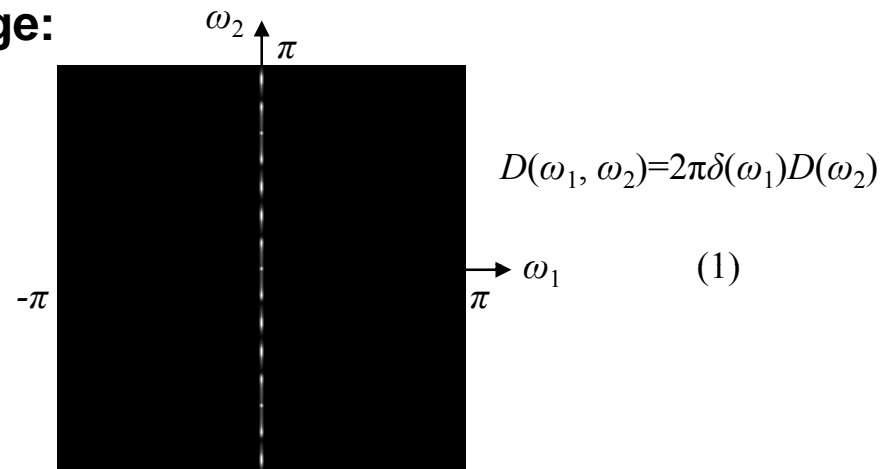
Nonlinear Filtering for Noise Smoothing

➤ Consider DTFT of noise image:

$d(n_1, n_2) = d(n_2) \cdot 1$
(constant along n_1)



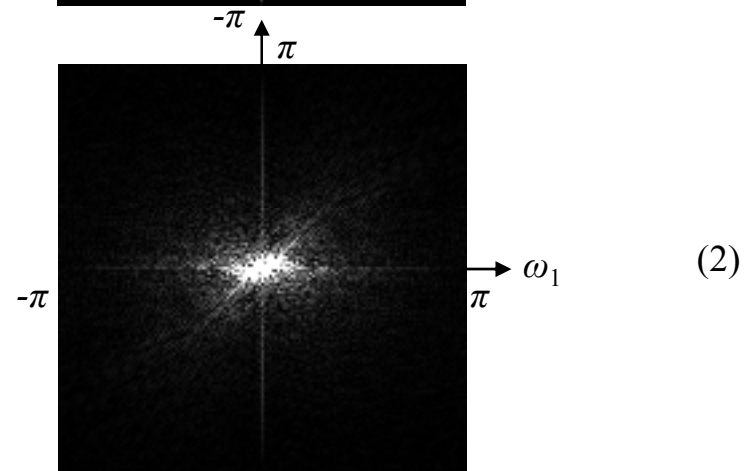
⇒



➤ DTFT of the desired image:

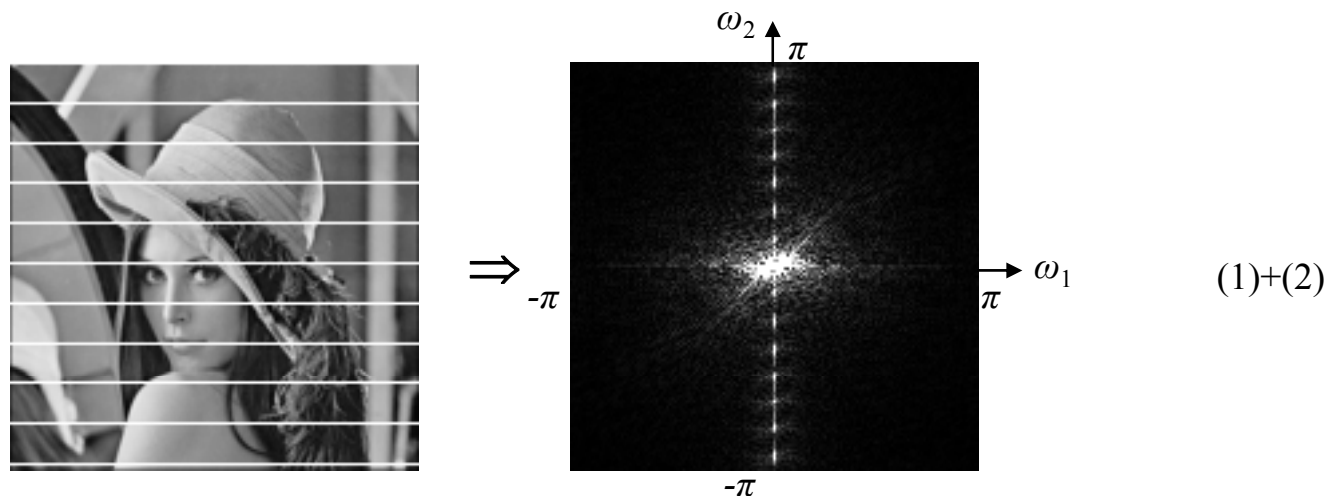


⇒



Nonlinear Filtering for Noise Smoothing

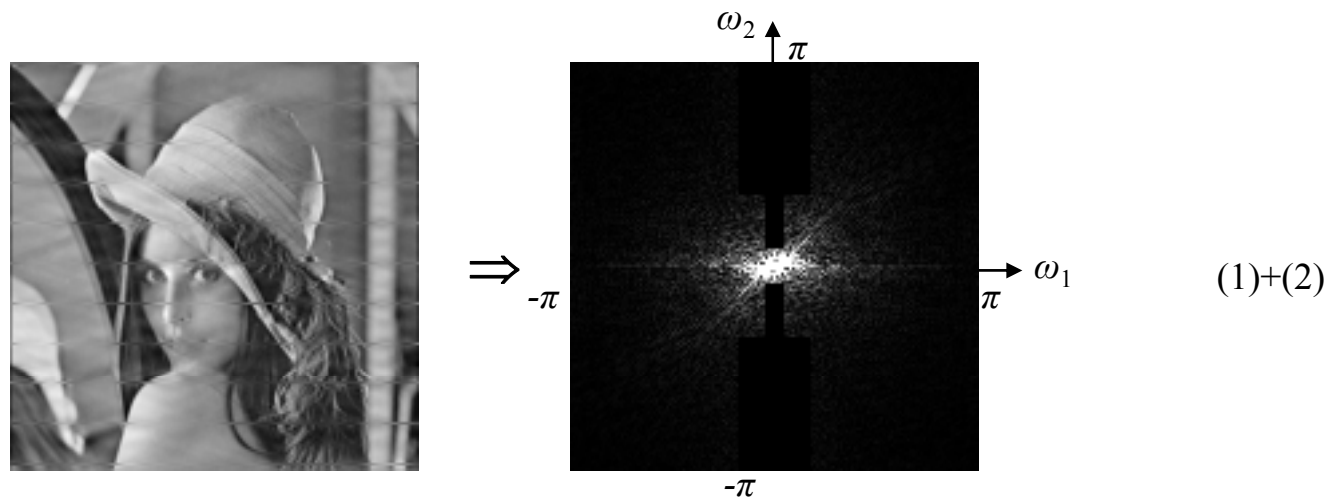
- DTFT of original (noisy) image:



- In the frequency domain, the signal and noise are separable (approximately) \Rightarrow null out noise component by removing dots in frequency domain, then take inverse transform \Rightarrow the noise is gone or significantly reduced.

Nonlinear Filtering for Noise Smoothing

➤ DTFT of filtered image:



Edge Enhancement

- **Motivation: Edges are very important for intelligibility, segmentation, analysis, and identification.**
- **Transform Compression**

1. Root Filtering

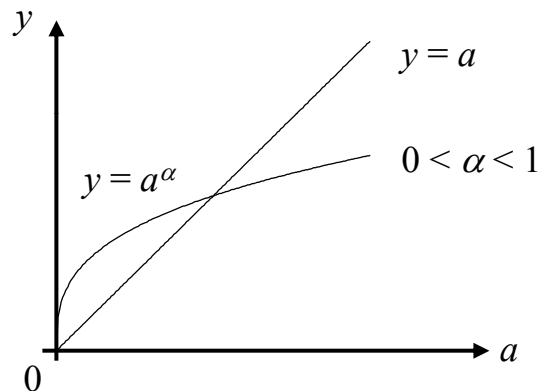
$$x(n_1 n_2) \leftrightarrow X(K_1, K_2) = \text{Transform (such as DFT)}$$

$$\begin{aligned} X_o(K_1, K_2) &= |X(K_1, K_2)|^{\alpha-1} X(K_1, K_2); \quad 0 \leq \alpha \leq 1 \\ &= |X(K_1, K_2)|^\alpha e^{j\theta(K_1, K_2)}; \quad 0 \leq \alpha \leq 1 \end{aligned}$$

- **Note: Since $0 \leq \alpha \leq 1 \Rightarrow |X(K_1, K_2)|^\alpha = \alpha$ -root of magnitude of $X(K_1, K_2)$.**

Edge Enhancement

- **Observation:** (about behavior)



- **Effect of α -rooting:** boosts low-amplitude components or coefficients and de-emphasizes high-amplitude components or coefficients.

Edge Enhancement

- For natural or common images, high-frequency components, which correspond to edges, tend to have low amplitudes, and low-frequency components tend to have high amplitudes.
- ⇒ α – rooting boosts high–frequency components which correspond to edges.
- α – rooting can be done in any transform domain: DFT, DCT, or Hadamard domain.
- Note: Blur is associated with loss in high-frequency region: boosting the high-frequency components can help in improving the blurred image.

Edge Enhancement

2. Generalized Cepstrum and Homomorphic Filtering

- Same as α – rooting but logarithm used to compress the dynamic range in transform domain.

$$x(n_1, n_2) \leftrightarrow X(K_1, K_2) = |X(K_1, K_2)| e^{j\theta(K_1, K_2)}$$

$$X_o(K_1, K_2) = \log |X(K_1, K_2)| e^{j\theta(K_1, K_2)}$$

$$x_o(n_1, n_2) = \text{inverse transform} [X_o(K_1, K_2)]$$

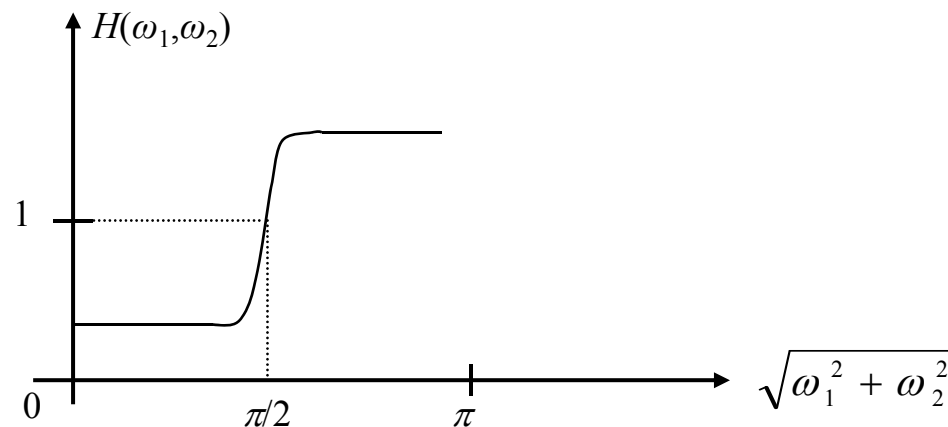
↑
Generalized Cepstrum (also called Generalized Homomorphic Transform)

- Note: In practice, positive constant is added to magnitude of X to avoid negative infinity.

Edge Enhancement

- **High-Pass Filtering (HPF)**

- Edges contribute to high-frequency components.
- High-pass filtering can be used to boost the high-frequency components and de-emphasize low-frequency components.

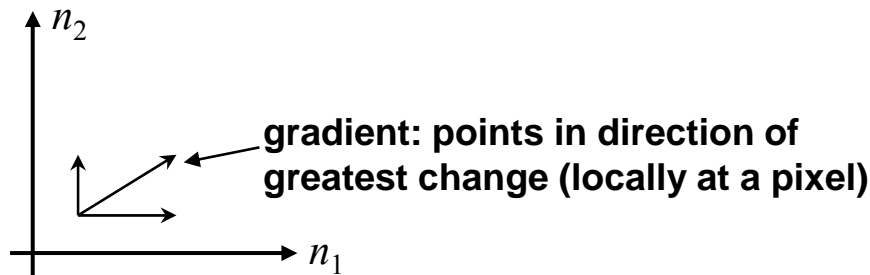


- High-pass filtering can help in increasing local contrast and in sharpening the image, but usually results in an increased background noise.

Edge Detection and Sharpening

- One basic tool: Gradient operator

$$g_x(n_1, n_2) = \nabla x(n_1, n_2) = \begin{bmatrix} \frac{\partial x(n_1, n_2)}{\partial n_1} \\ \frac{\partial x(n_1, n_2)}{\partial n_2} \end{bmatrix} = \begin{bmatrix} g_1(n_1, n_2) \\ g_2(n_1, n_2) \end{bmatrix}; \quad x(n_1, n_2) - \text{input image}$$



- The Gradient magnitude is commonly used for edge detection

$$|g(n_1, n_2)| = \left(\left(\frac{\partial x(n_1, n_2)}{\partial n_1} \right)^2 + \left(\frac{\partial x(n_1, n_2)}{\partial n_2} \right)^2 \right)^{\frac{1}{2}}$$

Edge Detection and Sharpening

- Detect edges based on first-order or second-order gradient information.
- Edge detection (Recall):
 - The Gradient magnitude commonly used to detect edges:

$$g_x(n_1, n_2) = \nabla x(n_1, n_2) = \begin{bmatrix} \frac{\partial x(n_1, n_2)}{\partial n_1} \\ \frac{\partial x(n_1, n_2)}{\partial n_2} \end{bmatrix} = \begin{bmatrix} g_1(n_1, n_2) \\ g_2(n_1, n_2) \end{bmatrix}; \quad x(n_1, n_2) - \text{input image}$$

$$|g(n_1, n_2)| = \left(\left(\frac{\partial x(n_1, n_2)}{\partial n_1} \right)^2 + \left(\frac{\partial x(n_1, n_2)}{\partial n_2} \right)^2 \right)^{\frac{1}{2}}$$

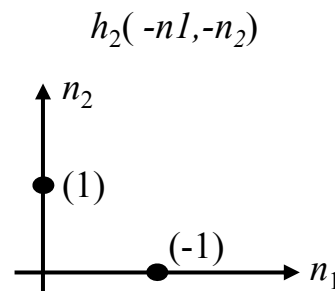
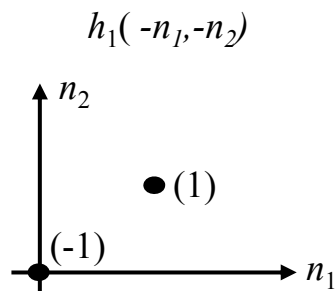
Edge Detection and Sharpening

- Recall: How do we realize (compute) the gradient in discrete time?
- Example: Robert's Gradient

$$g_1(n_1, n_2) = \frac{\partial x(n_1, n_2)}{\partial n_1} = x(n_1 + 1, n_2 + 1) - x(n_1, n_2)$$

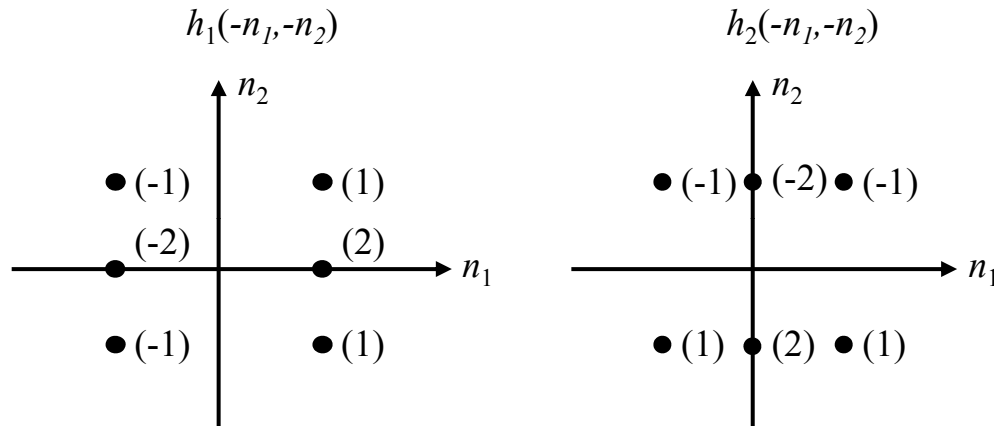
$$g_2(n_1, n_2) = \frac{\partial x(n_1, n_2)}{\partial n_2} = x(n_1, n_2 + 1) - x(n_1 + 1, n_2)$$

$$|g(n_1, n_2)| = \sqrt{|x(n_1 + 1, n_2 + 1) - x(n_1, n_2)|^2 + |x(n_1, n_2 + 1) - x(n_1 + 1, n_2)|^2}$$

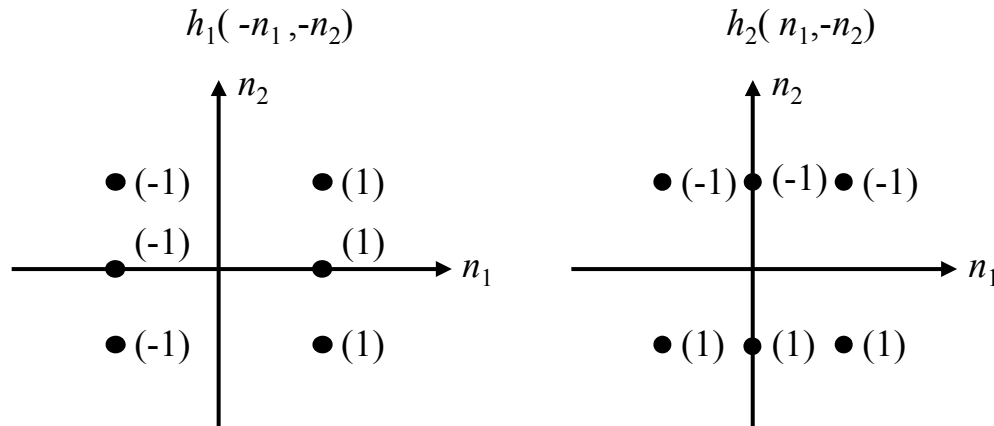


Edge Detection and Sharpening

➤ Sobel's Gradient



➤ Prewitt's Gradient



Edge Detection and Sharpening

- **Question:** How do we apply gradient to perform edge enhancement
- **Enhancement rule**

$$x_o(n_1, n_2) = \begin{cases} f(n_1, n_2); & \text{when } |g(n_1, n_2)| > \text{Threshold} \\ x(n_1, n_2); & \text{Otherwise} \end{cases}$$

where $f(n_1, n_2) = \alpha x(n_1, n_2) =$ **scaled version of original**

or $f(n_1, n_2) = \alpha g(n_1, n_2) =$ **scaled version of gradient**

- **Note:** Edge detection methods based on computing some form of gradient or based on computing a difference are sensitive to noise.
 - ✓ **Example:** Background noise appears as isolated edge points \Rightarrow noise smoothing can be applied before edge detection (recommended).

Edge Detection and Sharpening

- **Laplacian operator**

- **Edge is detected not only when $|g(n_1, n_2)|$ is large, but also zero crossings are considered.**

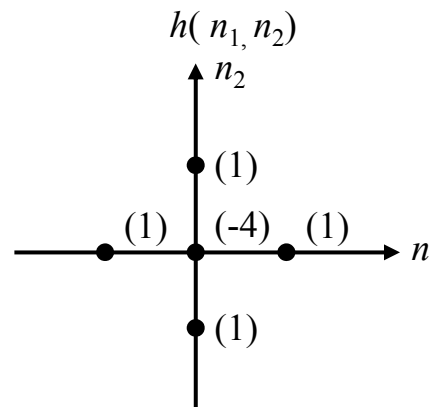
- **Laplacian: 2nd derivative in 1D case.**

- **Gradient: 1st derivative in 1D case.**

- **Laplacian:** $L(n_1, n_2) = \nabla^2 x(n_1, n_2) = \frac{\partial^2 x(n_1, n_2)}{\partial n_1^2} + \frac{\partial^2 x(n_1, n_2)}{\partial n_2^2}$

- ✓ **One possible implementation:**

$$L(n_1, n_2) = \nabla^2 x(n_1, n_2) = x(n_1 + 1, n_2) + x(n_1 - 1, n_2) + x(n_1, n_2 + 1) + x(n_1, n_2 - 1) - 4x(n_1, n_2)$$
$$= x(n_1, n_2) ** h(n_1, n_2)$$



Edge Detection and Sharpening

➤ **Solution:**

At zero-crossing, check local variance: $\left\{ \begin{array}{l} \text{If large variance} \Rightarrow \text{edge.} \\ \text{No edge otherwise.} \end{array} \right.$

➤ **Enhancement rule:**

$$x_o(n_1, n_2) = \begin{cases} f(n_1, n_2); & \text{when } |L_x(n_1, n_2)| \leq \varepsilon \text{ and large variance} \\ x(n_1, n_2); & \text{Otherwise} \end{cases}$$

- **Note:** Edge detection methods are used for detecting object/region boundaries for segmentation and image analysis.

Morphological Filters

- **Mathematical morphology operations have been used extensively for image enhancement.**
- **Basic mathematical morphology operations:**
 - **Erosion**
 - **Dilation**
 - **Opening**
 - **Closing**
- **Mathematical morphological operations were initially defined for binary images and later expanded to gray-scale images.**

Morphological Filters

- **Erosion**

- **Erosion reduces (erodes) the selected area (object, block) of the binary image by an amount specified by the structuring element.**
- **Mathematical notation:**

Let Y be the erosion of the image X with a structuring element B :

$$Y = X \ominus B = \left\{ 1 \text{ for all } (n_1, n_2) \text{ such that } B_{n_1, n_2} \subseteq X; 0, \text{ else.} \right\}$$

where B_{n_1, n_2} is the shifted version of B centered at (n_1, n_2) .

Morphological Filters

- **Dilation**

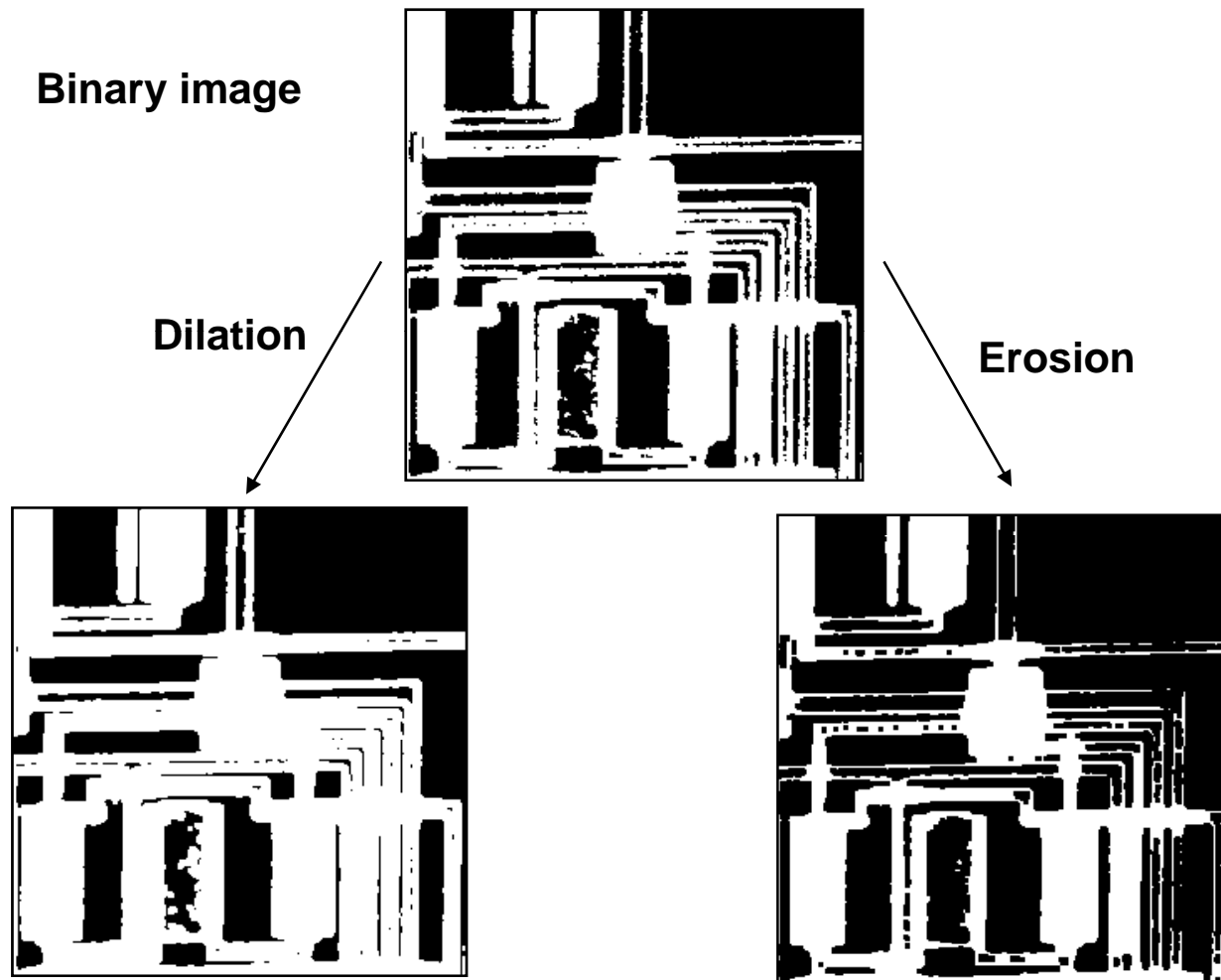
- **Dilation expands the boundary of selected area (object, block) in binary image.**
- **Mathematical notation:**

Let Y be the dilation of the image X w with a structuring element B :

$$Y = X \oplus B = \{ 1 \text{ for all } (n_1, n_2) \text{ such that } B_{n_1, n_2} \cap X \neq 0; 0, \text{ else.} \}$$

where B_{n_1, n_2} is the shifted version of B centered at (n_1, n_2) .

Morphological Filters



Morphological Filters

- **Closing: Dilation followed by erosion.**
 - Has effect of filling in gulfs and small holes, connecting objects that are in close proximity and smoothing edge boundaries.
 - Mathematical notation: Closing of X with structuring element B

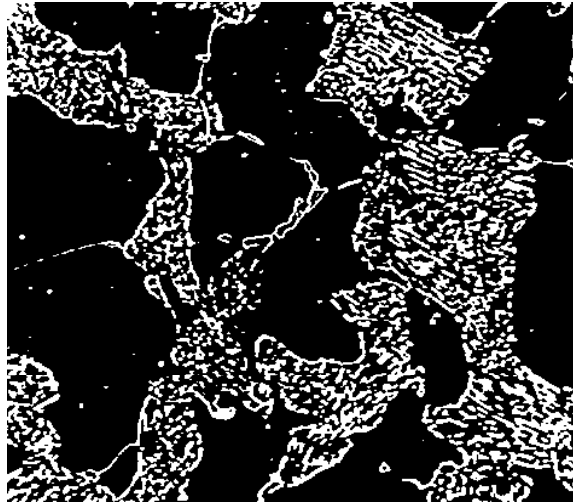
$$X \bullet B = (X \oplus B) \ominus B$$

- **Opening: Erosion followed by dilation.**
 - Objects smaller than the structuring element are deleted.
 - Large objects are retained.
 - Mathematical notation: Opening of X with a structuring element B

$$X \circ B = (X \ominus B) \oplus B$$

Morphological Filters

Binary image



Closing



Opening



Morphological Filters

- **Gray Scale Operations for a 3x3 structuring element centered at processed pixel $x(n_1, n_2)$:**

- **Gray-scale erosion:**

$$y(n_1, n_2) = \min[x(n_1, n_2), x(n_1, n_2 + 1), x(n_1 - 1, n_2 + 1), \dots, x(n_1 + 1, n_2 + 1)]$$

- **Gray-scale dilation:**

$$y(n_1, n_2) = \max[x(n_1, n_2), x(n_1, n_2 + 1), x(n_1 - 1, n_2 + 1), \dots, x(n_1 + 1, n_2 + 1)]$$

- **Gray-scale Openings and Closings:**

- ✓ **Gray-scale opening = gray-scale erosion followed by gray-scale dilation.**
- ✓ **Gray-scale closing = gray-scale dilation followed by gray-scale erosion.**