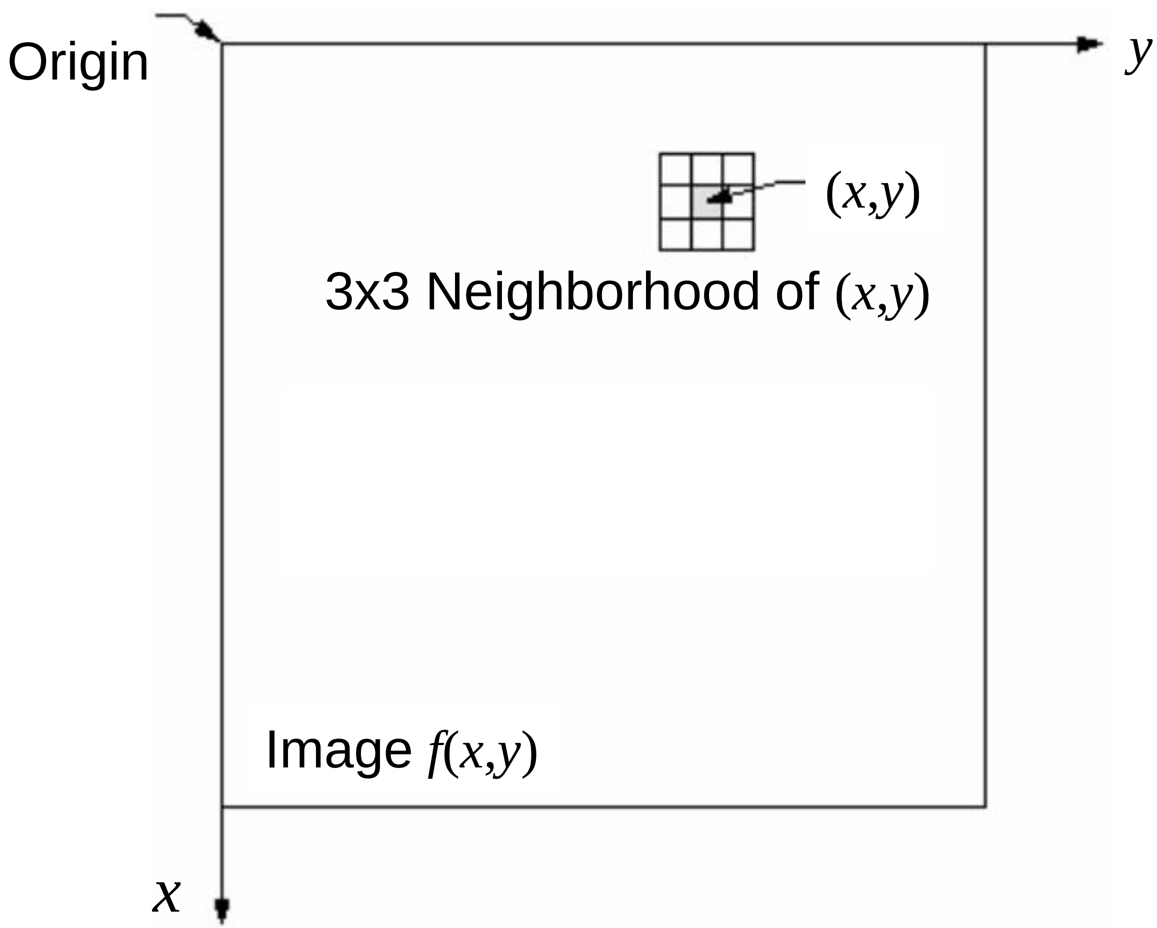
Spatial Filtering
For Image Enhancement and Restoration

Spatial-Domain Operations

The general spatial-domain processing takes the following form, where the operation T is defined over some neighborhood of (x,y):



Spatial Filtering

Goal: To determine the value of a pixel according to the values of pixels within a neighborhood.

How a linear filter (size $(2a+1)$x$(2b+1)$) works:

$$g(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) f(x+s, y+t)$$

The filter

Smoothing Filters

Goal: To create a blurred version of the input image.

A 3x3 averaging filter (box filter)

$$\frac{1}{9} \times \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

A 3x3 weighted averaging filter

$$\frac{1}{16} \times \begin{array}{|c|c|c|} \hline 1 & 2 & 1 \\ \hline 2 & 4 & 2 \\ \hline 1 & 2 & 1 \\ \hline \end{array}$$

Smoothing Filters

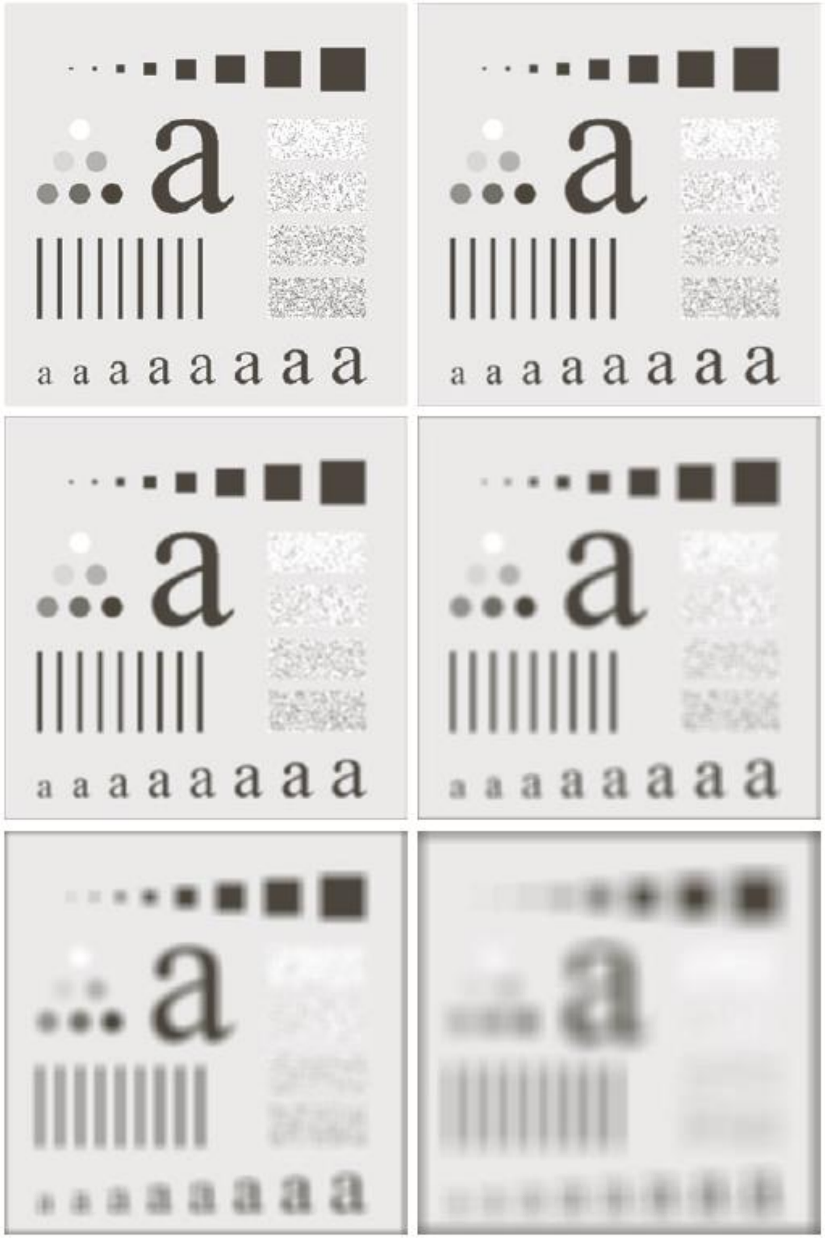
Example:

Image size: 500x500

Filter type: Box filters

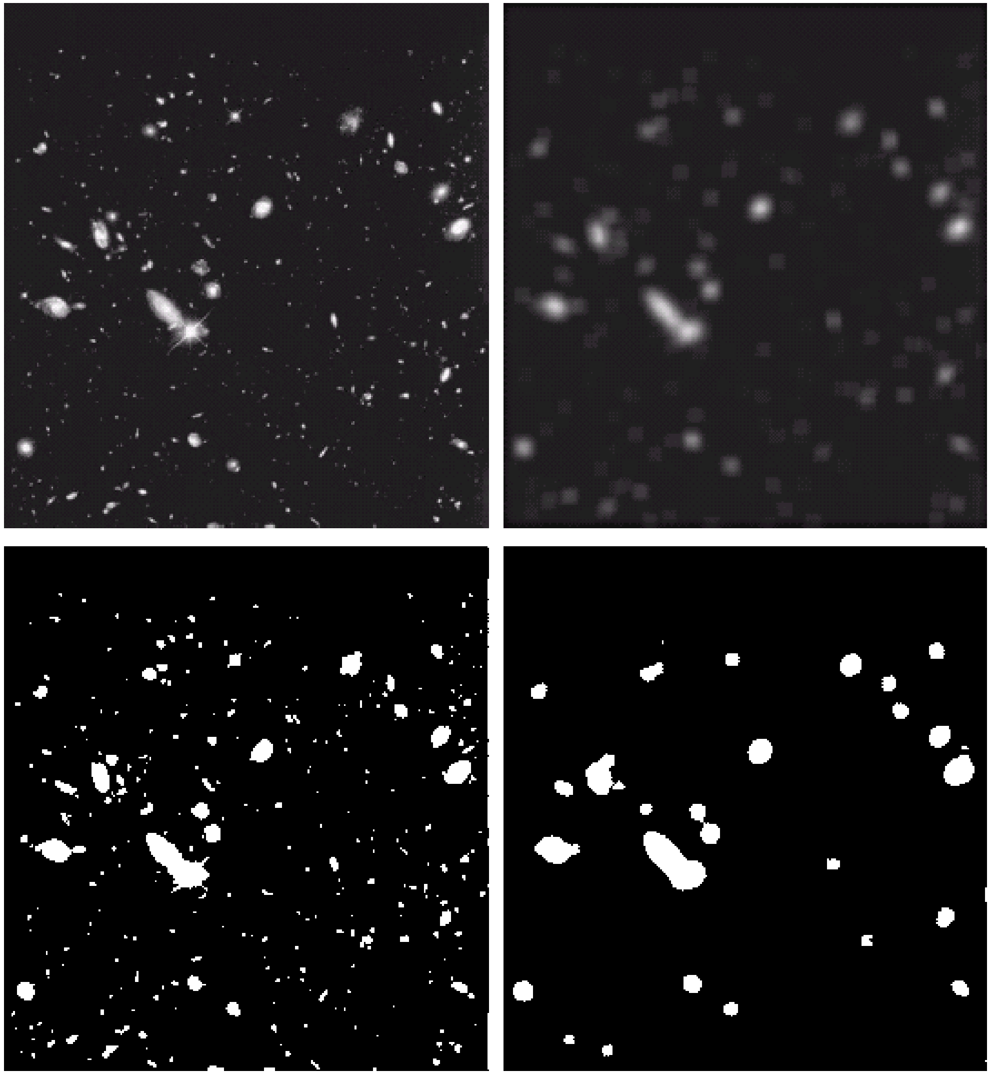
Filter sizes:

3x3, 5x5, 9x9, 15x15, 35x35



Smoothing Filters

When used with thresholding to select major objects:



Handling Image Borders

Special care has to be taken when part of the mask moves beyond the image borders while processing pixels near image borders. Some methods to handle this situation:

- Accepting a smaller output image
- Accepting the processing with a partial mask
- Padding the border with zero or other constant values
- Padding the border by row/column replication
- Padding the border by row/column reflection

Order-Statistics Filters

These filters determine the new value of the target pixel in a neighborhood based on the ***ranking*** of the pixel values in the neighborhood.

Let $v_1 \leq v_2 \leq \ldots v_n$ be the sorted pixel gray-level values within the neighborhood (n = # pixels in the neighborhood).

How a order-statistics filter works:

$$g(x, y) = \sum_{i=1}^{n} w_i v_i(x, y)$$

Some representative order-statistics filters:

- Median filters
- Weighted averaging filters with cropped extremal values
- Min and Max filters

Median Filters

This is the most commonly used order-statistics filter:

$$g(x,y)=\sum_{i=1}^{n} w_i v_i(x,y)$$

$$w_i=\begin{cases}1 & \text{if } i=(n+1)/2 \\ 0 & \text{otherwise}\end{cases} \qquad \text{assuming odd } n$$

Example (border handled with zero padding):

$$\begin{array}{|ccccc|}\hline 2&2&3&2&1\\ 3&4&3&2&3\\ 3&5&5&6&6\\ 4&5&6&7&5\\ 4&4&6&6&4\\ \hline\end{array} \xrightarrow{\text{3x3 median filter}} \begin{array}{|ccccc|}\hline 0&2&2&2&0\\ 2&3&3&3&2\\ 3&4&5&5&3\\ 4&5&6&6&5\\ 0&4&5&5&0\\ \hline\end{array}$$

Max and Min Filters

Finding local maximums / minimums:

A

2	2	3	2	1
3	4	3	2	3
3	5	5	6	6
4	5	6	7	5
4	4	6	6	4

3x3 max filter

B

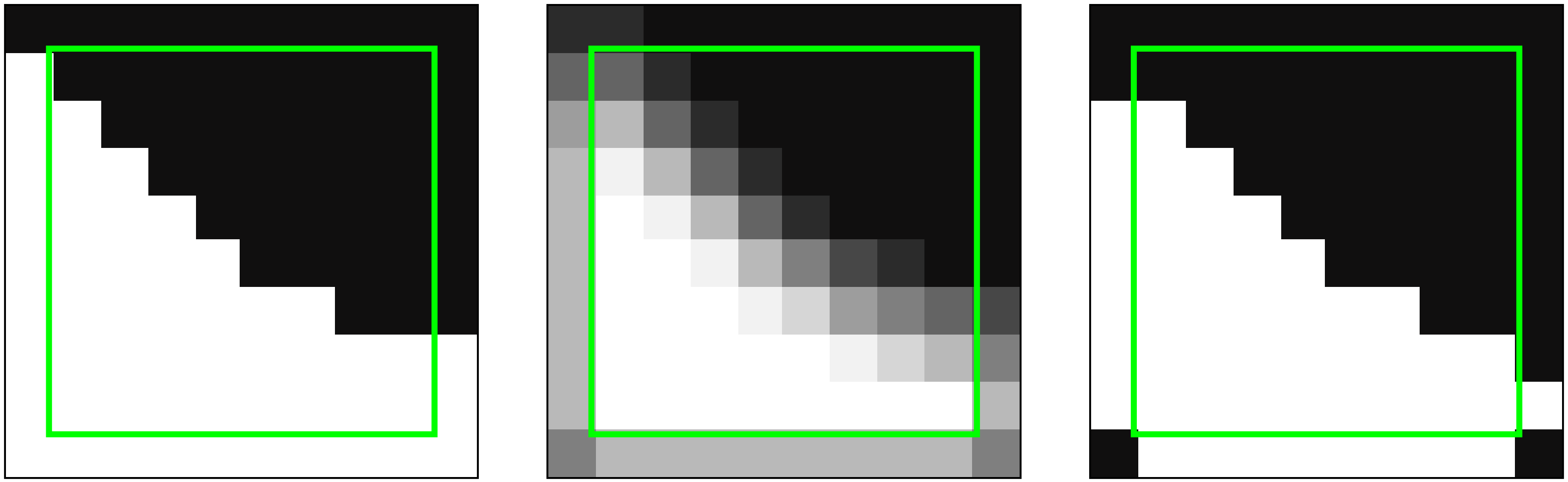
4	4	4	3	3
5	5	6	6	6
5	6	7	7	7
5	6	7	7	7
5	6	7	7	7

A=B?

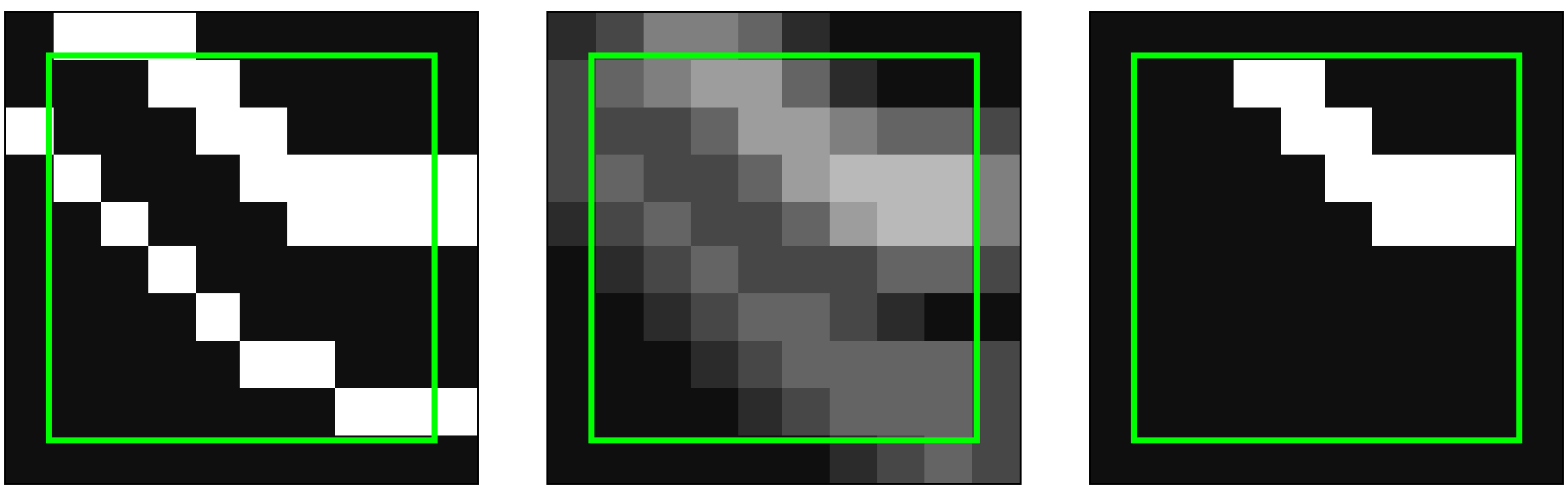
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	1	0
0	0	0	0	0

Averaging Filters vs. Median Filters

Effect on edges:



Effect on this lines:



Gray-Level Derivatives

Filters based on first or second derivatives of pixel values are used to detect and enhance changes or details.

Since images are functions defined over a discrete space, the derivatives are defined as differences.

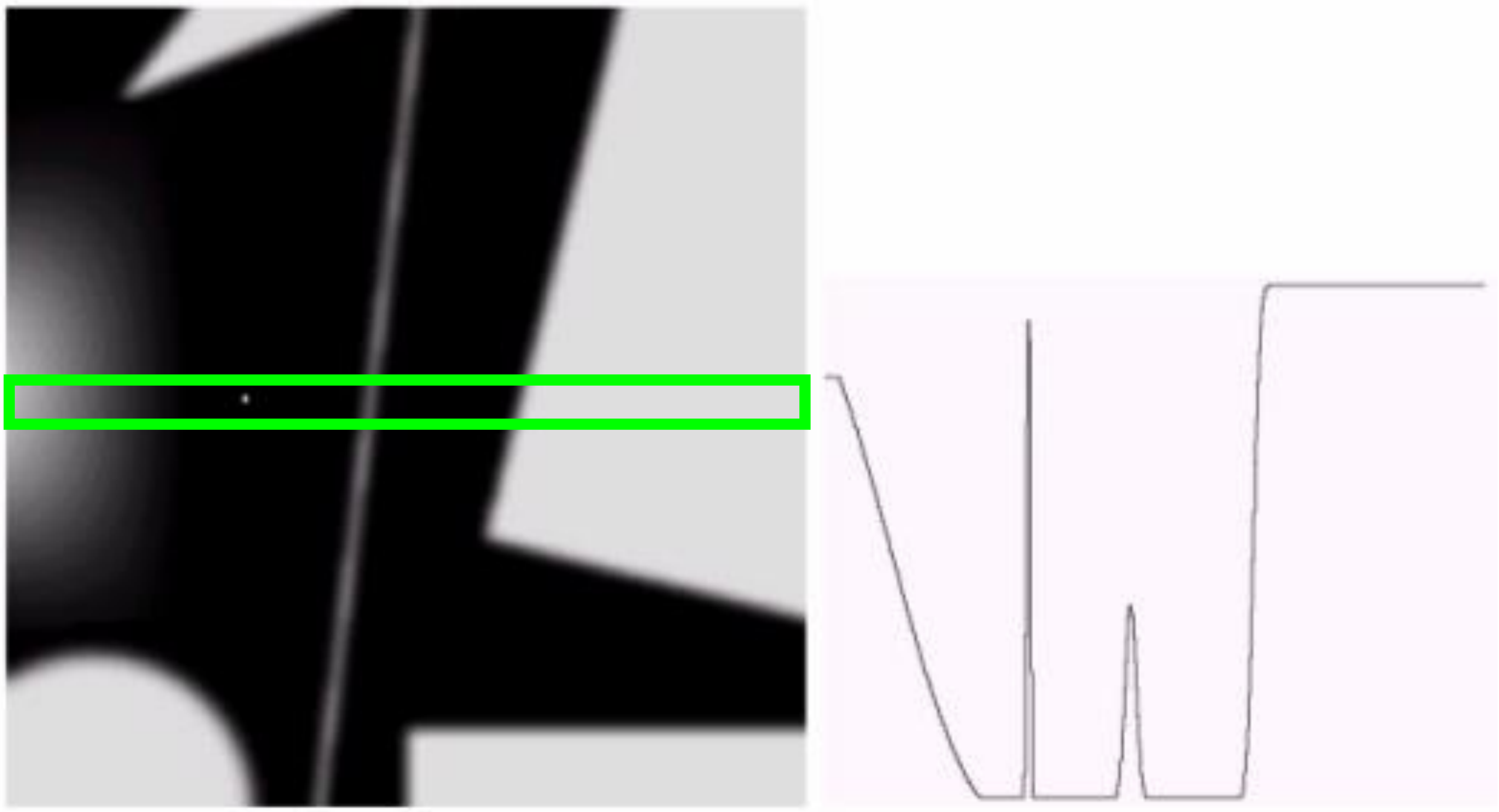
The simplest form of 1-D first derivative:

$$\frac{\partial f}{\partial x} = f(x+1, y) - f(x, y)$$

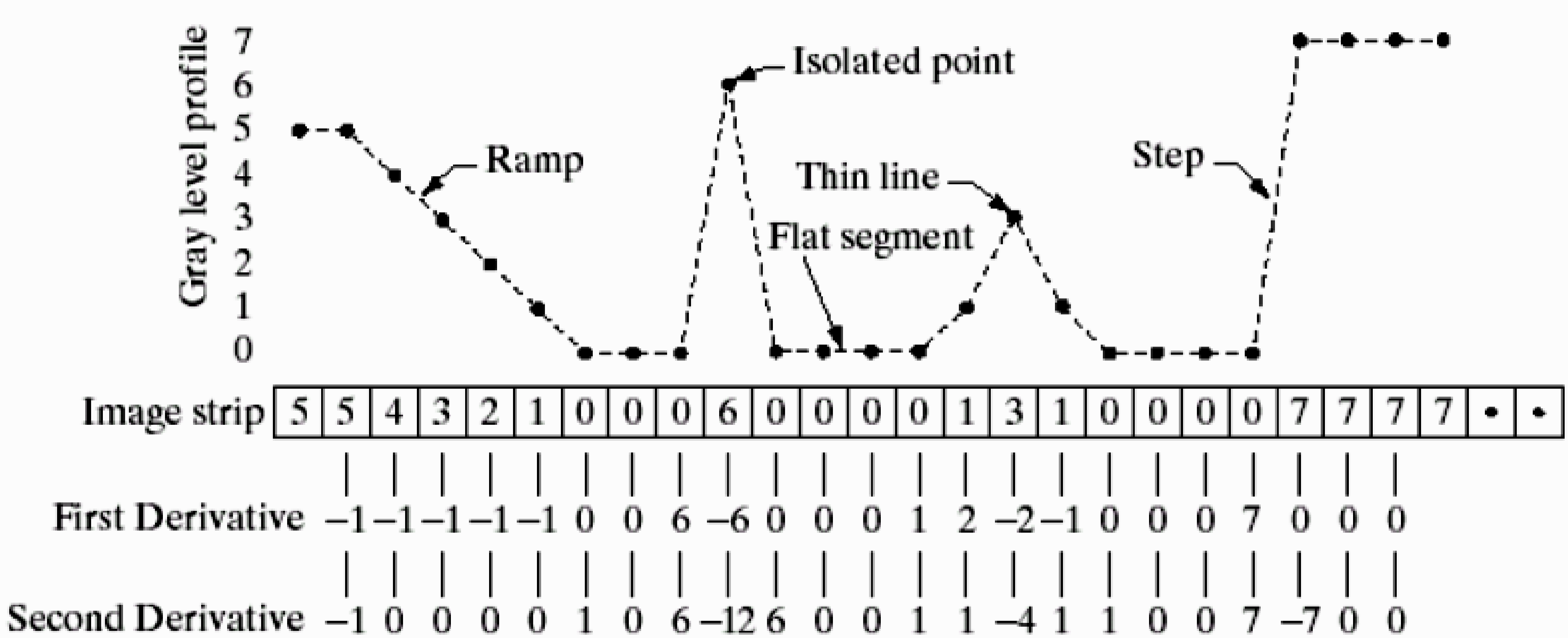
1-D second derivative:

$$\frac{\partial^2 f}{\partial x^2} = f(x+1, y) + f(x-1, y) - 2f(x, y)$$

Gray Level Derivatives



Gray Level Derivatives



Laplacian Filters

The definition of Laplacian:

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

Discrete 2-D second derivative:

$$\nabla^2 f = [f(x+1, y) + f(x-1, y) - 2f(x, y)]$$
$$+ [f(x, y+1) + f(x, y-1) - 2f(x, y)]$$

Laplacian Filters

Second derivatives:

0	1	0
1	-4	1
0	1	0

1	1	1
1	-8	1
1	1	1

Another form (used more frequently in practice):

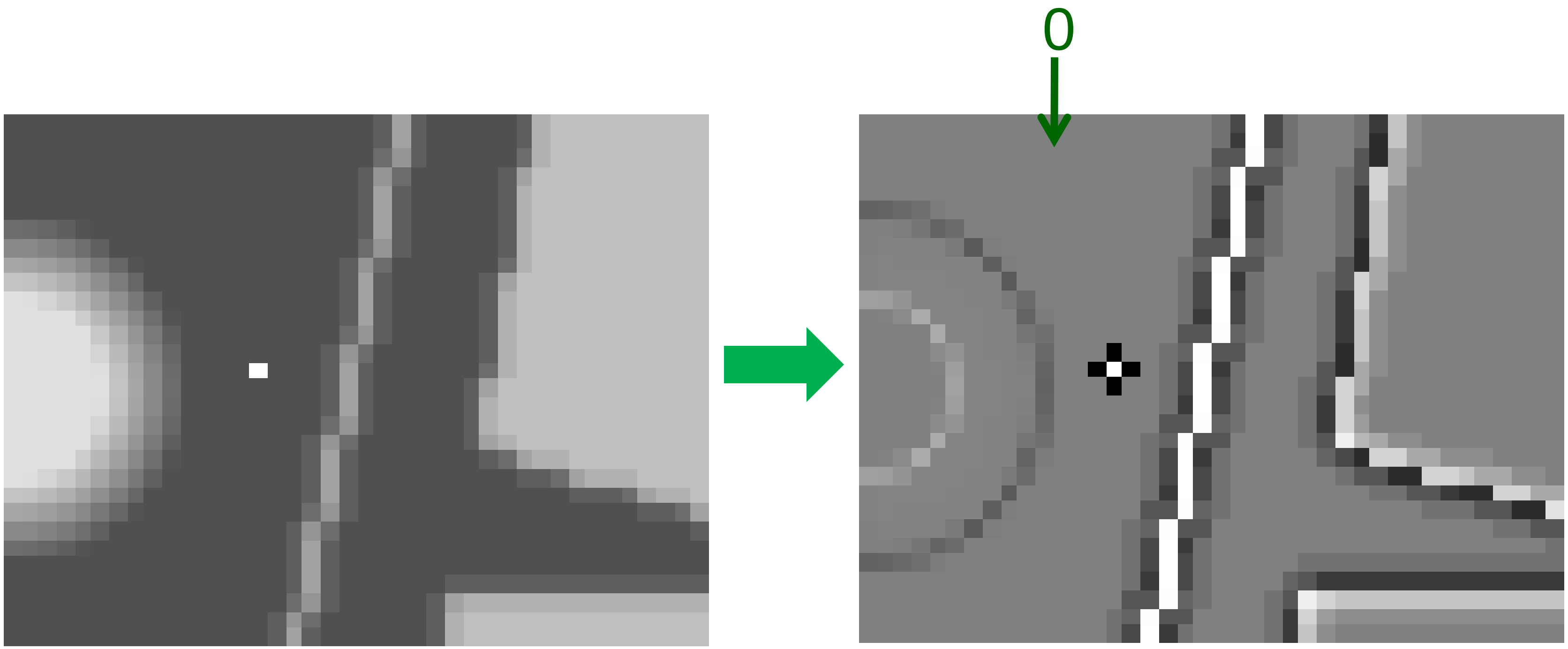
0	-1	0
-1	4	-1
0	-1	0

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

Always be specific about which form you're using.

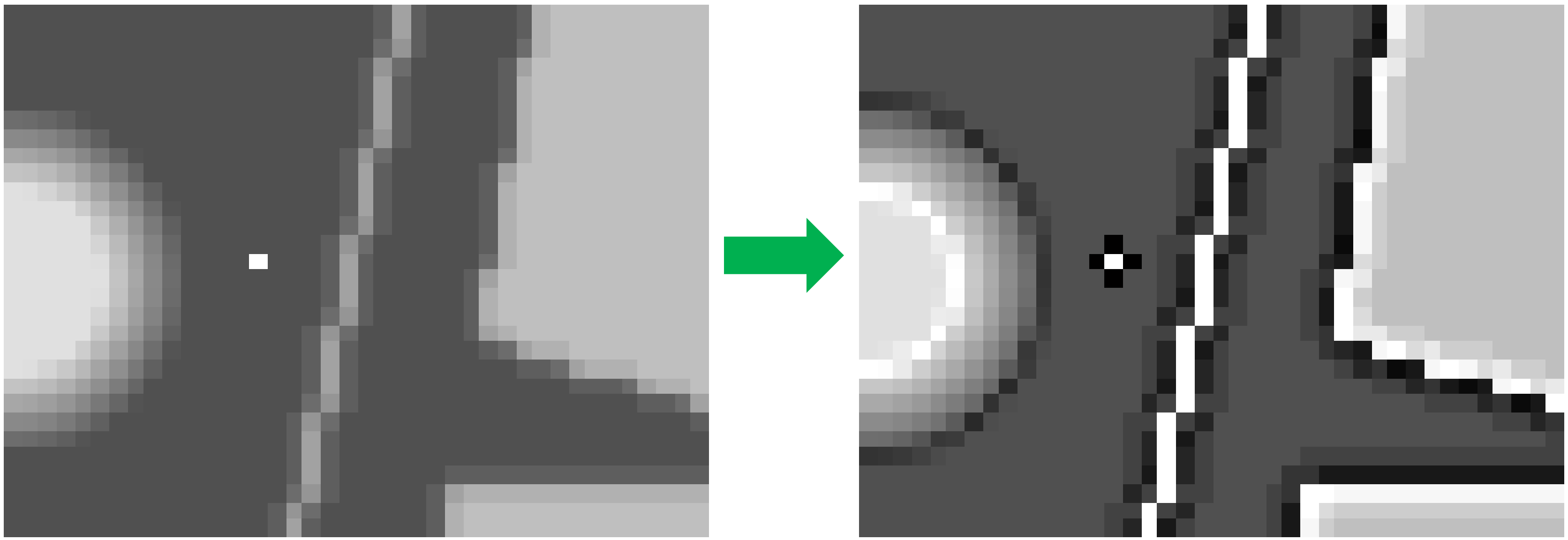
Laplacian Filters

Second derivatives:



0	-1	0
-1	4	-1
0	-1	0

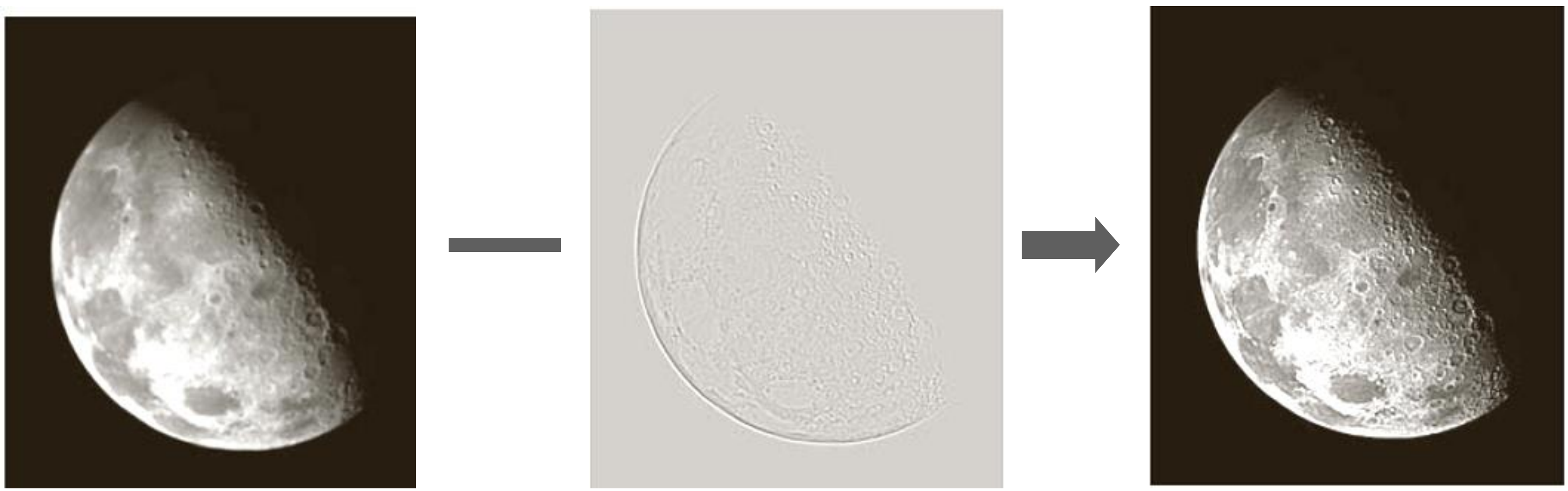
Sharpening Filters



$$\begin{array}{|c|c|c|}\hline 0 & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & 0 & 0 \\ \hline \end{array} + \begin{array}{|c|c|c|}\hline 0 & -1 & 0 \\ \hline -1 & 4 & -1 \\ \hline 0 & -1 & 0 \\ \hline \end{array} = \begin{array}{|c|c|c|}\hline 0 & -1 & 0 \\ \hline -1 & 5 & -1 \\ \hline 0 & -1 & 0 \\ \hline \end{array}$$

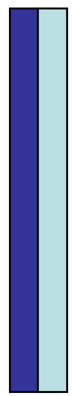
Sharpening Filters

Example in application:



Unsharp Masking

This idea comes from a technique for sharpening photographic pictures before there is digital image processing.



original negative + blurred positive

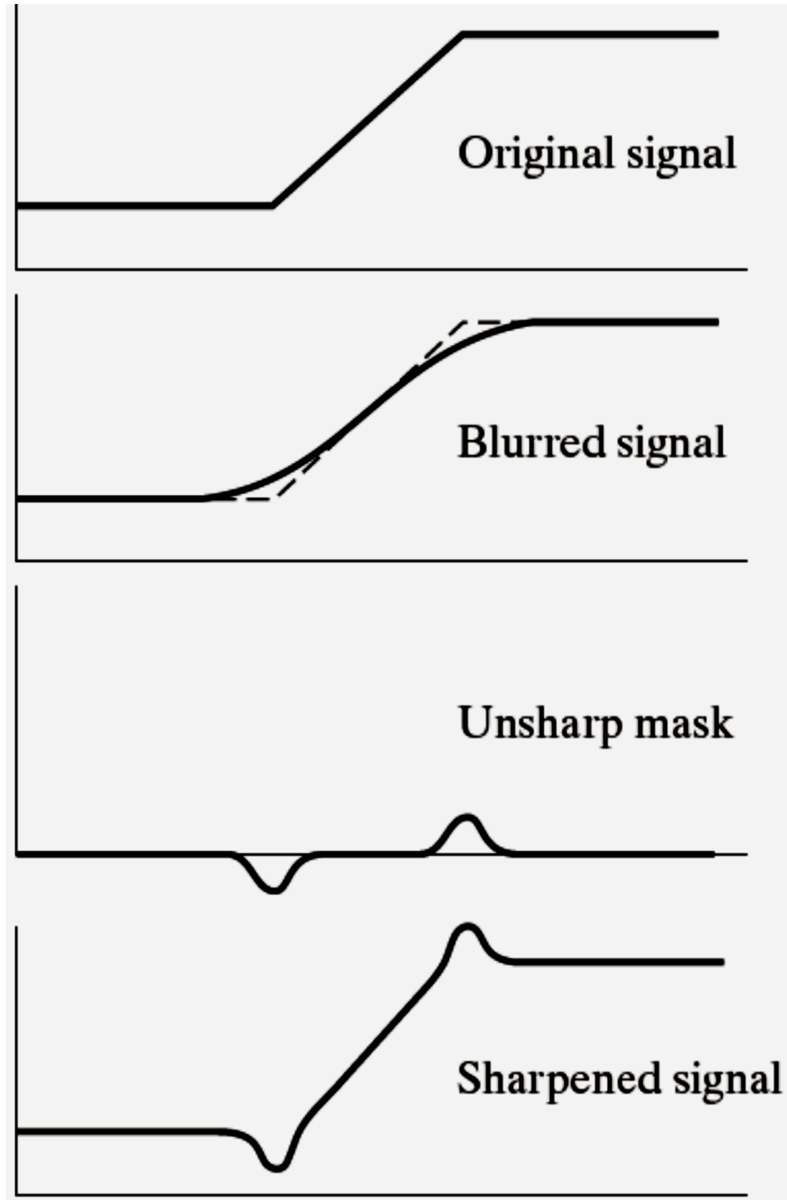
= combined negative

The digital form:

$$g_{mask}(x, y) = f(x, y) - \bar{f}(x, y)$$

$$g(x, y) = f(x, y) + k * g_{mask}(x, y)$$

Unsharp Masking



Restoration vs. Enhancement

- Enhancement: To improve the appearance of an image from its current form (e.g., contrast enhancement): more subjective.
- Restoration: To undo the effect of "degradation" so that an image is as close to its original form as possible: more objective.
- Enhancement may include tasks in restoration.
- Here we focus on the restoration of noisy images.

Image Degradation and Restoration

Model of degradation:

$$g(x,y) = h(x,y) * f(x,y) + \eta(x,y)$$

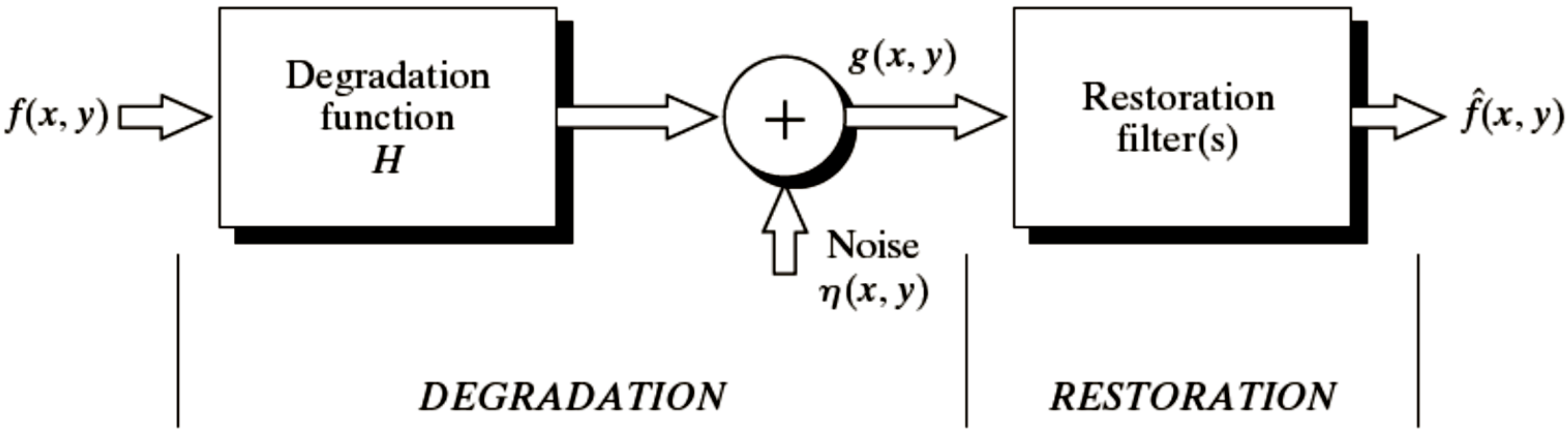


Image Noise Models

- Noise usually comes into an image during acquisition or transmission.
- Noise models: Functional forms of the probability density function (pdf) of noise, $\eta(x, y)$.
- Six different models are given in the textbook; some models are most appropriate for certain imaging modalities.
- At this stage we only consider location-independent noise (i.e., the noise pdf is the same for every pixel).

Image Noise Models

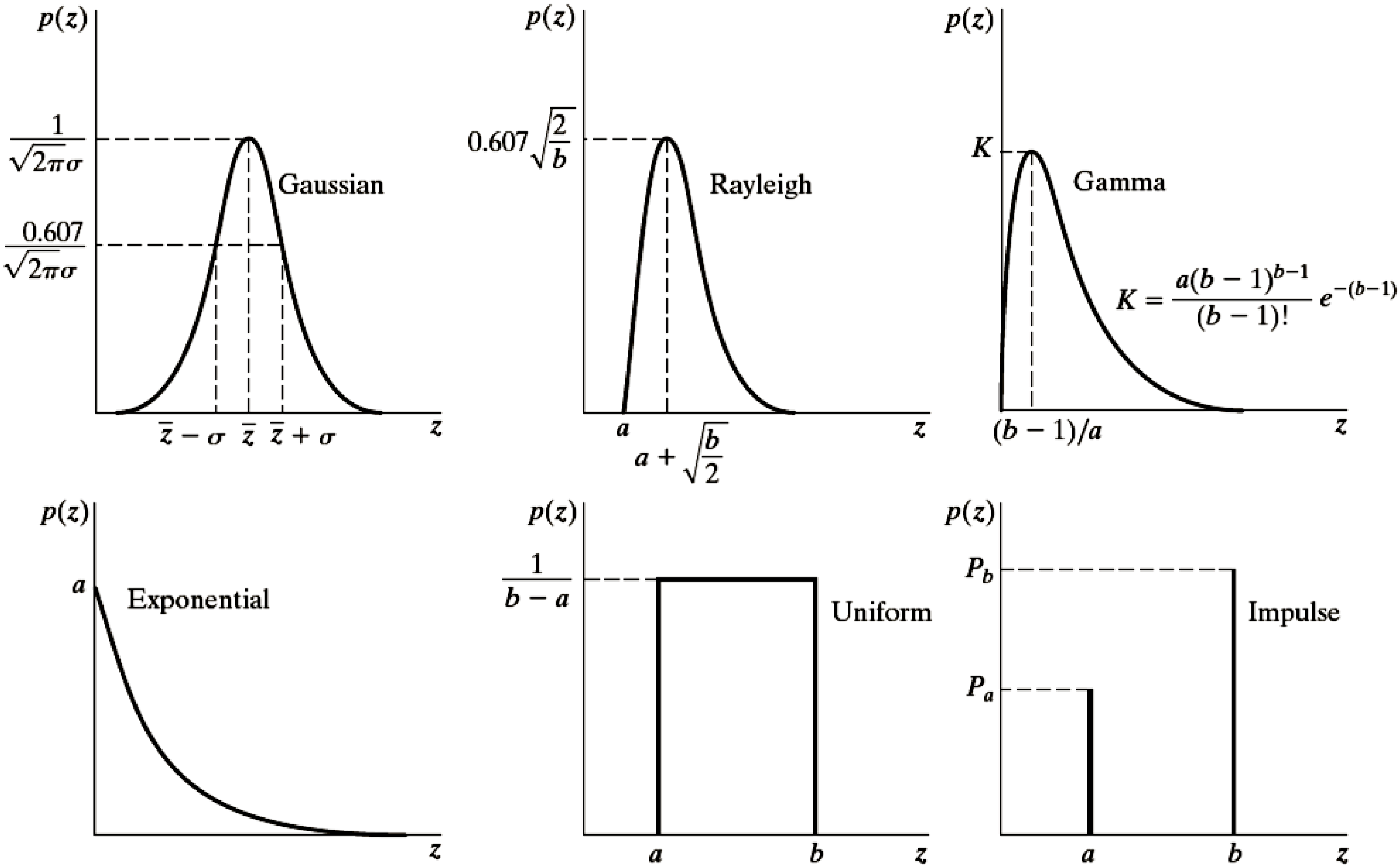
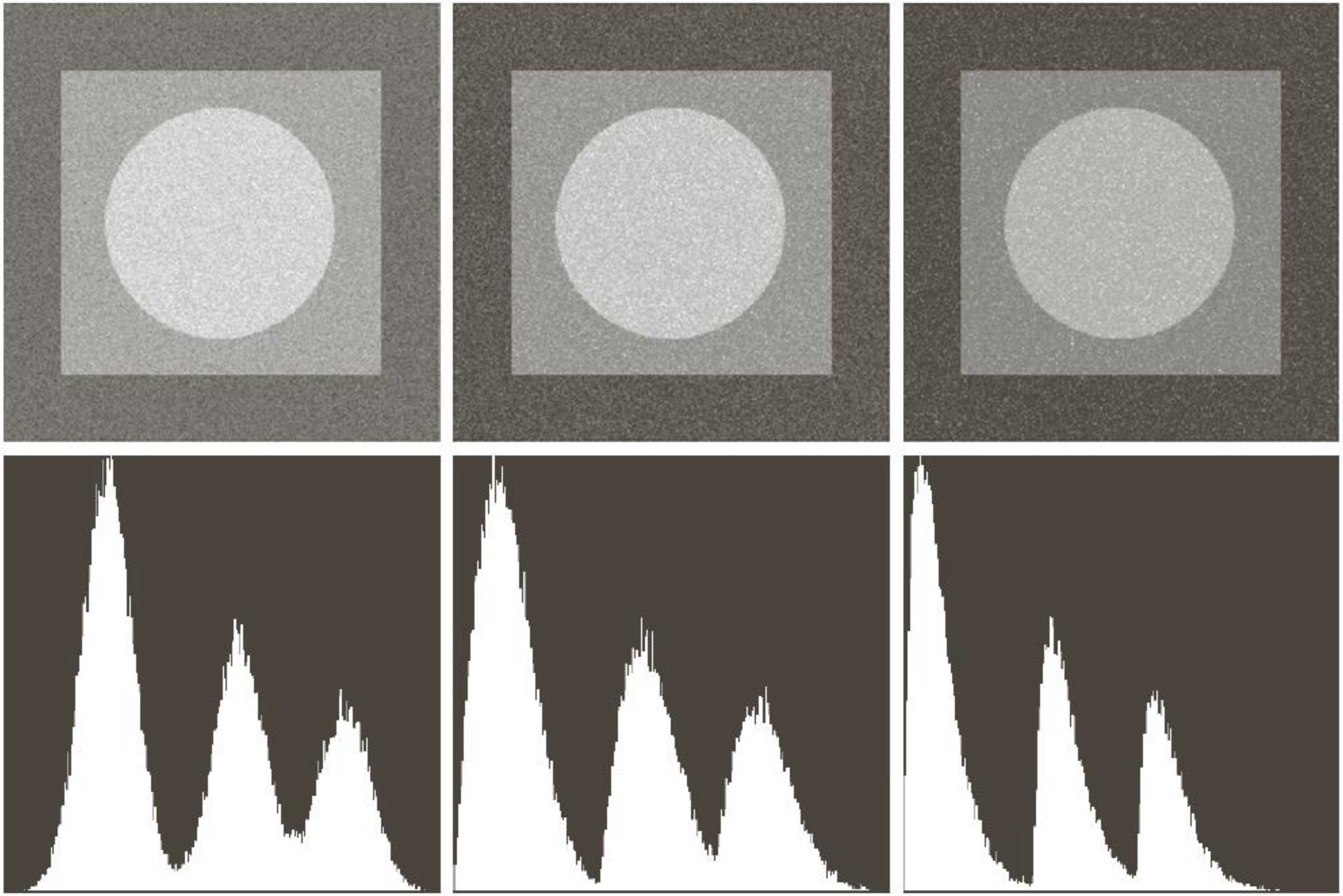
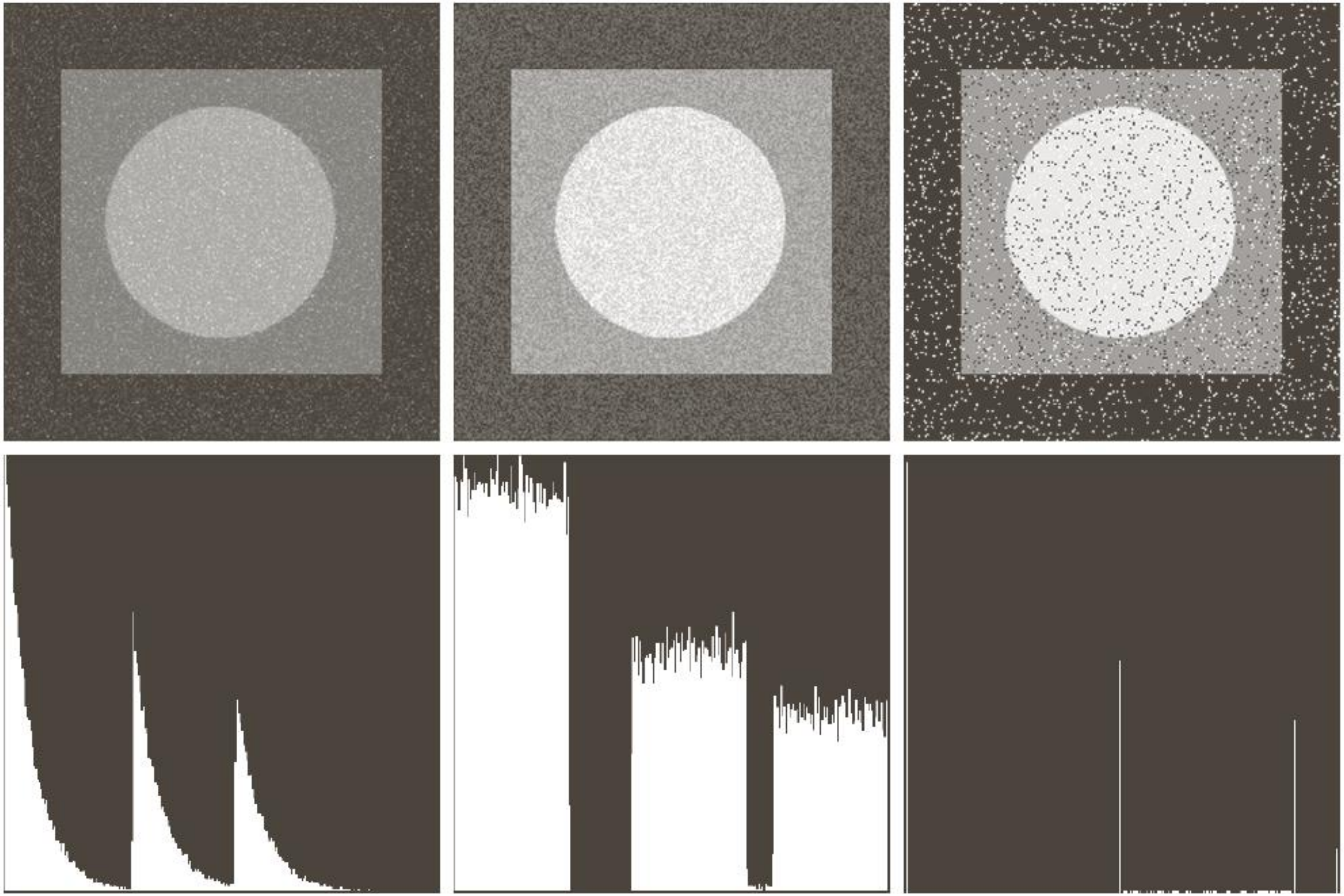


Image Noise Models



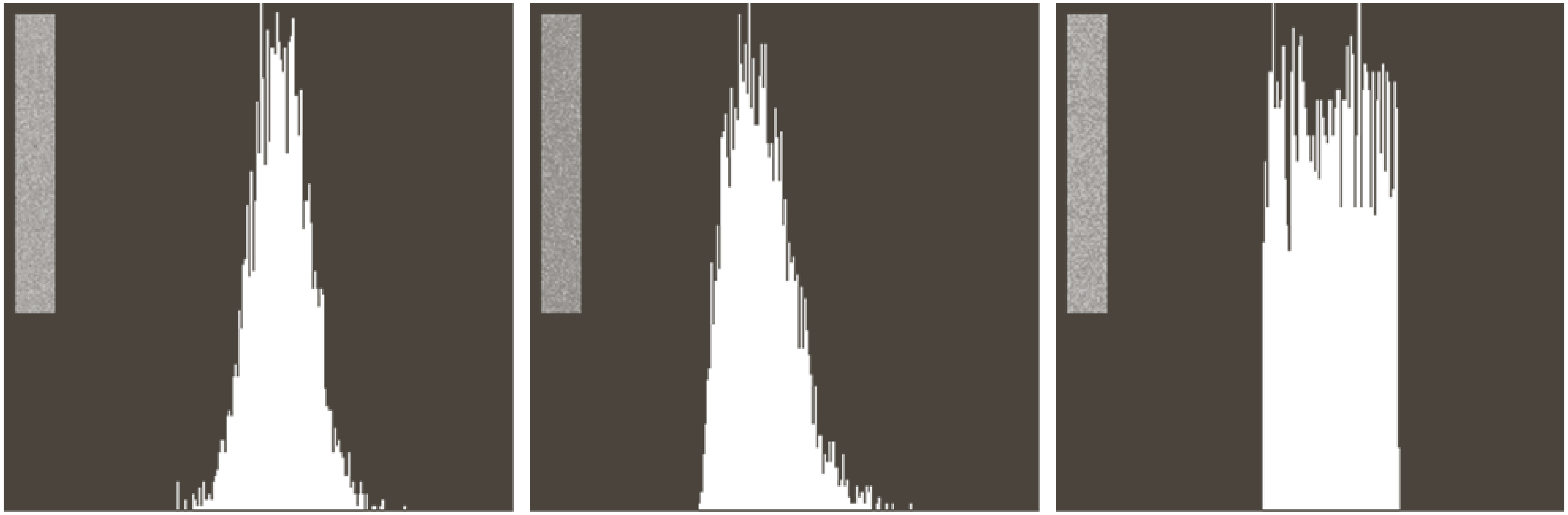
Gaussian Raleigh Gamma

Image Noise Models



Exponential Uniform Salf-and-Pepper

Estimating Noise Parameters



Usually a uniform region with mid-gray average intensity is needed.

Gaussian Smoothing Filter

This is a smoothing filter that resembles a 2-D Gaussian function. (A Gaussian has the advantage of being smooth in both the spatial and frequency domains.)

A normalized isotropic 2-D Gaussian:

$$h(s,t) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{s^2 + t^2}{2\sigma^2} \right)$$

Ideally, this filter should have a size that covers the whole image because $h(s,t)$ is nonzero everywhere. In practice, filter size of a few times the standard deviation σ is sufficient.

Gaussian Smoothing Filter

Example Gaussian filter: 7x7 with $\sigma = 1$:

0.0000	0.0002	0.0011	0.0018	0.0011	0.0002	0.0000
0.0002	0.0029	0.0131	0.0216	0.0131	0.0029	0.0002
0.0011	0.0131	0.0586	0.0966	0.0586	0.0131	0.0011
0.0018	0.0216	0.0966	**0.1592**	0.0966	0.0216	0.0018
0.0011	0.0131	0.0586	0.0966	0.0586	0.0131	0.0011
0.0002	0.0029	0.0131	0.0216	0.0131	0.0029	0.0002
0.0000	0.0002	0.0011	0.0018	0.0011	0.0002	0.0000

Gaussian filters are widely used in, for example, preprocessing of the image before other tasks (such as edge detection) to reduce spurious contents.

Order-Statitic Filters for Denoising

- Median filter: $$\hat{f}(x,y) = \underset{(s,t)\in S_{xy}}{\text{median}}\, g(s,t)$$

- Midpoint filter: $$\hat{f}(x,y) = \frac{1}{2}\left[\max_{(s,t)\in S_{xy}} g(s,t) + \min_{(s,t)\in S_{xy}} g(s,t)\right]$$

- Alpha-trimmed filter: A compromise between median and averaging filters

$$g(x,y) = \sum_{i=1}^{n} w_i v_i(x,y)$$

$$w_i = \begin{cases} 1/(n-2k) & \text{if } i > k \text{ and } i \le n-k \\ 0 & \text{otherwise} \end{cases}$$

assuming odd n

Averaging vs. Median Filters

5x5 averaging filter

5x5 median filter

Adaptive Filters

- For noise pdf that is Gaussian, uniform, etc.:
  - Main problem: Blurring of edges or image details
  - Approach: Take different smoothing actions according to whether a pixel appears to be located at an edge or image detail.
- For impulse noise:
  - Main problem: Removal of image details that are not impulse noise
  - Approach: Double-check whether a pixel appears to be an impulse noise, and do no smoothing if it is not.

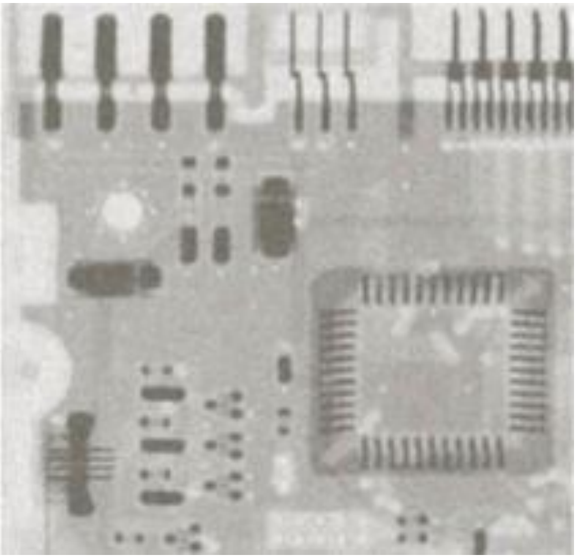
Adaptive Local Noise Reduction Filter

- Based on the ratio between σ_η^2 (estimated noise variance) and σ_L^2 (local variance)
- $\sigma_\eta^2 = \sigma_L^2$: Local variance is mainly due to noise ➔ Apply averaging filter
- $\sigma_\eta^2 << \sigma_L^2$: Local variance is not due to noise ➔ Apply very little smoothing to preserve the local information

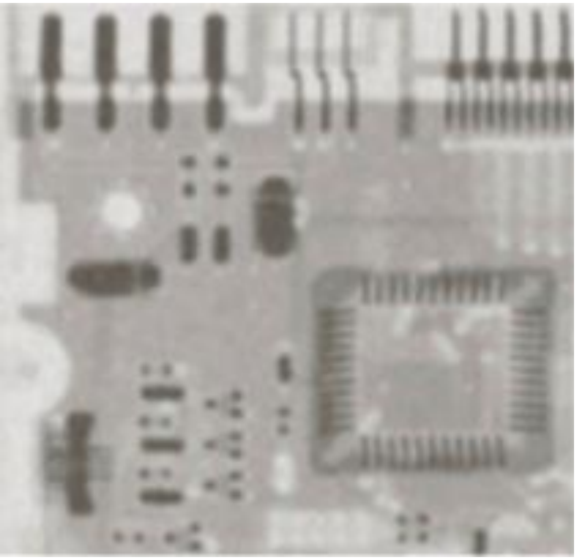
$$\hat{f}(x,y) = g(x,y) - \frac{\sigma_\eta^2}{\sigma_L^2}\left[g(x,y) - \overset{\substack{\text{Local mean}\\ \downarrow}}{m_L}\right]$$

Adaptive Local Noise Reduction Filter

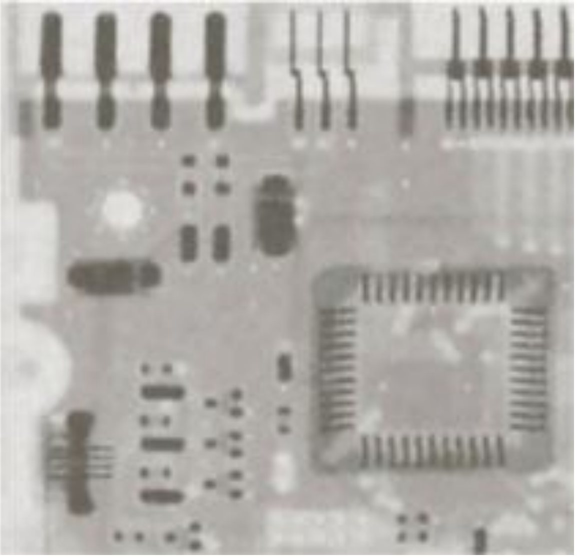
Noisy Original



Averaging Filter



Adaptive Filter

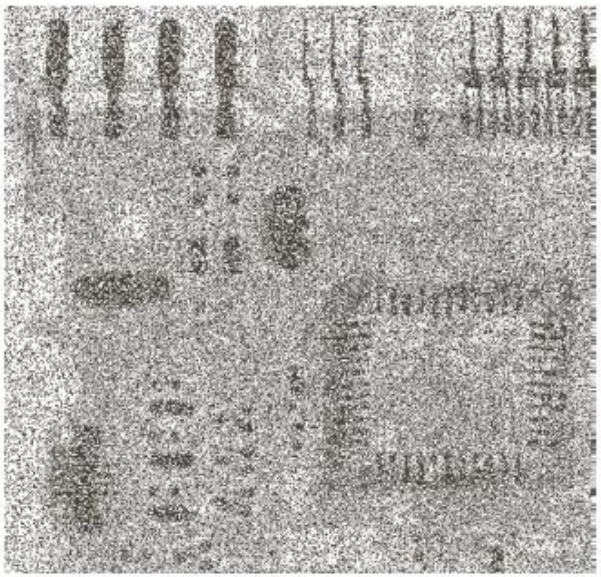


Adaptive Median Filter

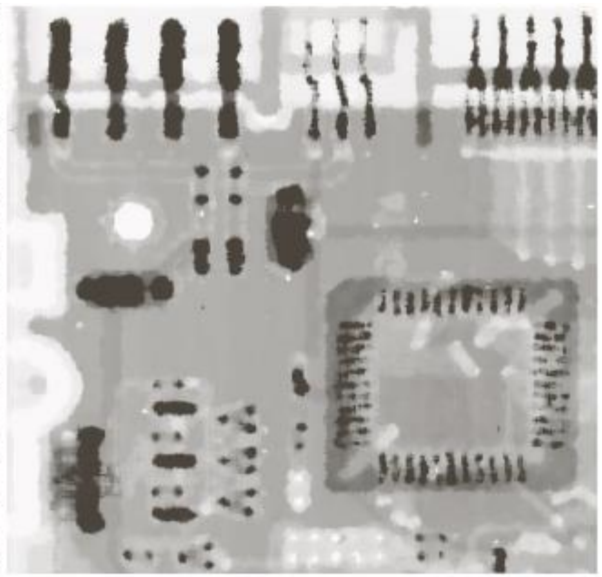
- Specifically designed to handle images with high density of impulse noises.
- The decision is based on whether z_{xy} (original pixel value) and z_{med} (local median) appear to be impulse values.
- z_{med} is a local extreme value: It's possible that z_{med} is corrupted by impulse noise ➔ Increase filter size (if allowed) or output z_{xy}.
- z_{med} is not a local extreme value (not corrupted):
  - If z_{xy} is a local extreme value ➔ z_{xy} is likely corrupted by impulse noise ➔ output z_{med}.
  - If z_{xy} is not a local extreme value ➔ z_{xy} is not corrupted by impulse noise ➔ output z_{xy}.

Adaptive Median Filter

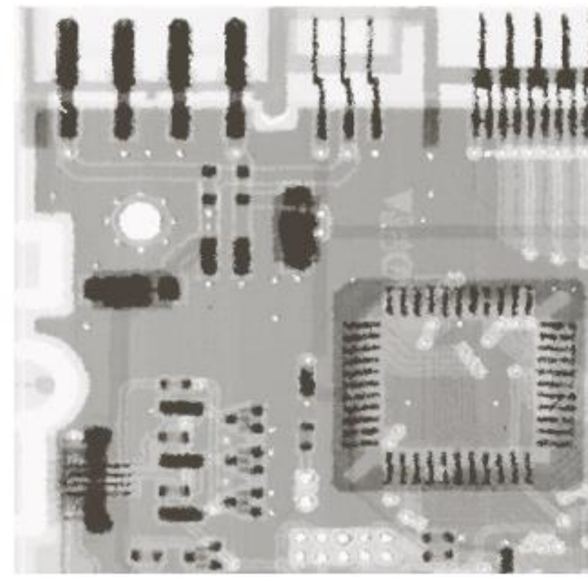
Noisy Original



Median Filter



Adaptive Median Filter



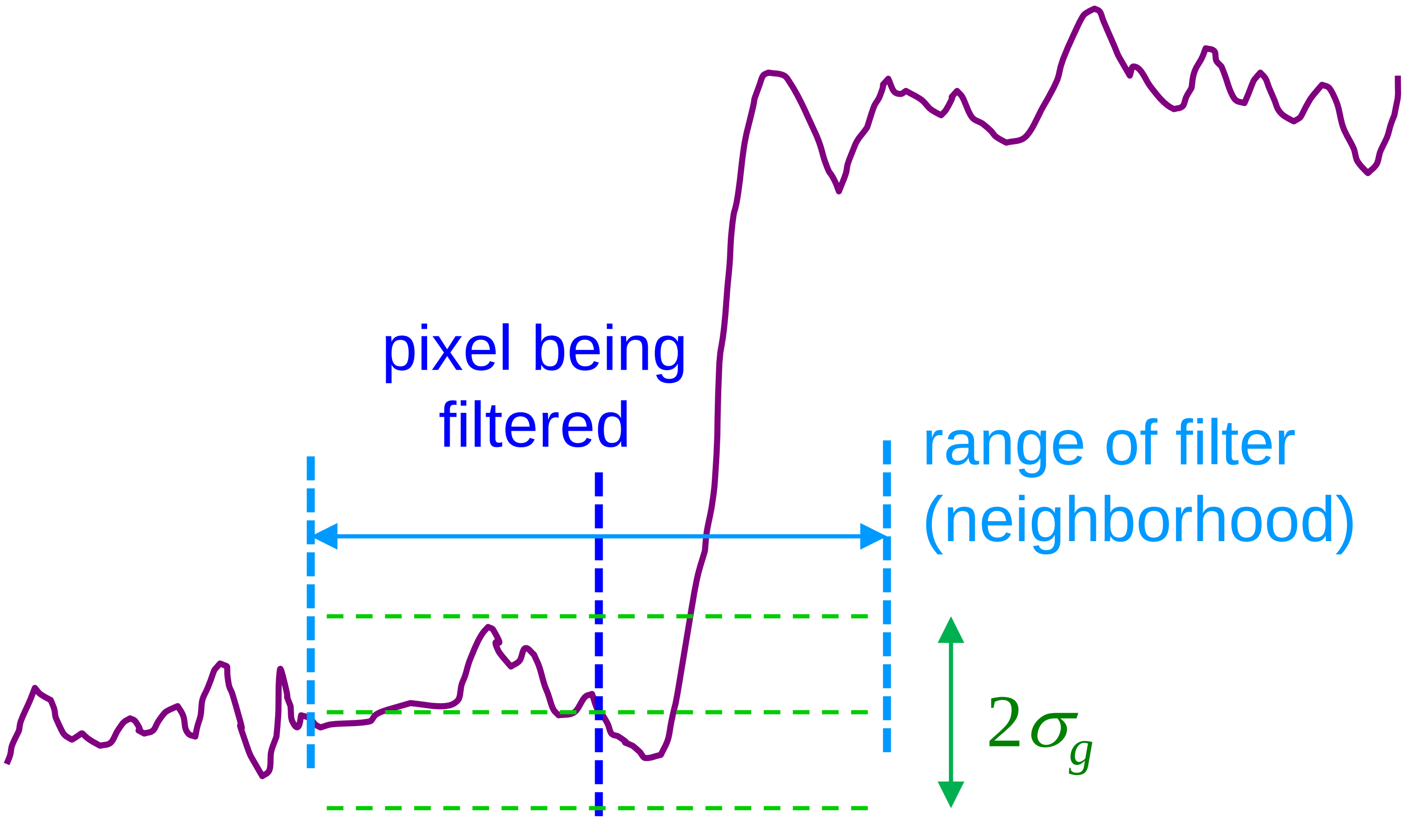
Bilateral Filter

- A widely used smoothing filter with better edge preservation than basic averaging filters.
- Can be considered a weighted Gaussian smoothing filter, where the additional weight is based on intensity difference.

$$\hat{f}(x,y) = \frac{\sum_{(s,t)\in S_{xy}} w(s,t)\,g(s,t)}{\sum_{(s,t)\in S_{xy}} w(s,t)}$$

$$w(s,t) = \exp\left[-\frac{\|(x,y)-(s,t)\|^2}{2\sigma_d^2}\right]\exp\left[-\frac{|g(x,y)-g(s,t)|^2}{2\sigma_g^2}\right]$$

Bilateral Filter



The value of σ_g is usually estimated from the source image.

Bilateral Filter

An example: