

Spatial Filtering

- BACKGROUND
- SMOOTHING FILTERS
- SHARPENING FILTERS

Image Processing Methods in Spatial Domain

Spatial domain refers to the image plane itself.

Image processing methods in spatial domain may be divided into 2 main categories

1. Point operations/Intensity transformation

- operate on *single pixels* of an image
- principally for the purpose of contrast manipulation and image thresholding

2. Spatial filtering

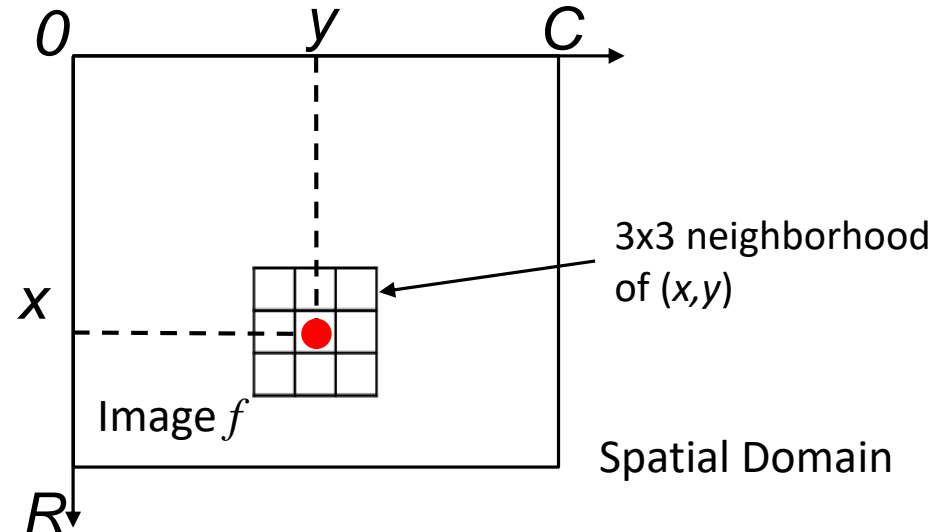
- process the pixel in a small *neighborhood of pixels* around the given pixel
- deals with performing operations, such as image sharpening

Image Processing Methods in Spatial Domain

The spatial domain processes can be denoted by the expression,

$$g(x, y) = T[f(x, y)]$$

Where $f(x, y)$ is the input image, $g(x, y)$ is the output image and T is the operator on f defined over the (x, y) or a neighbourhood of point (x, y) .

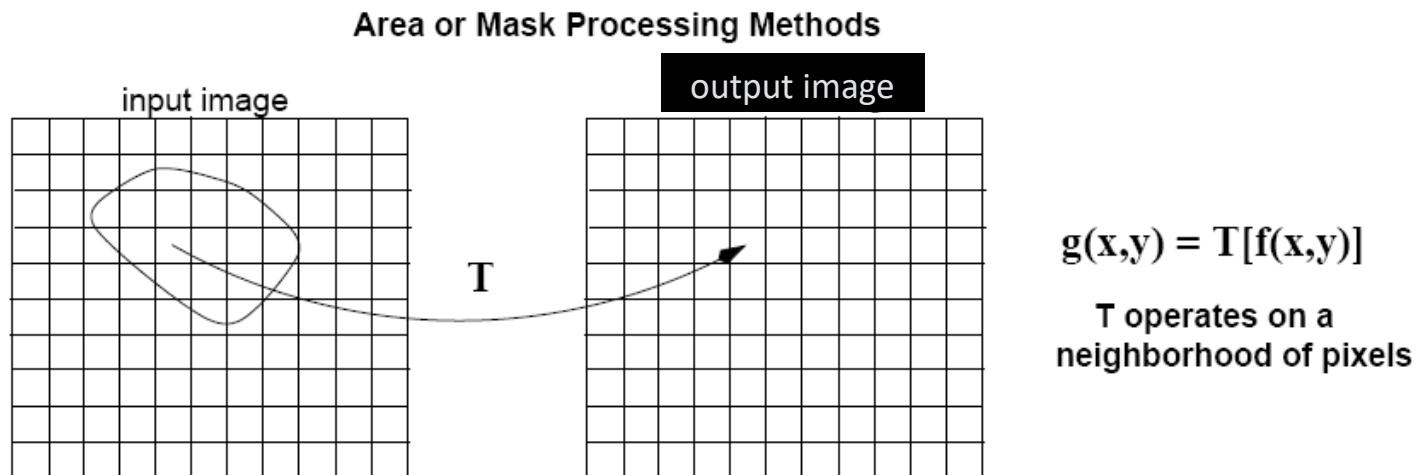


Spatial Filtering

Spatial filtering consists of:

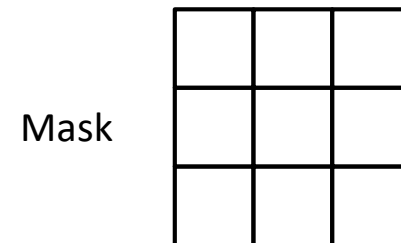
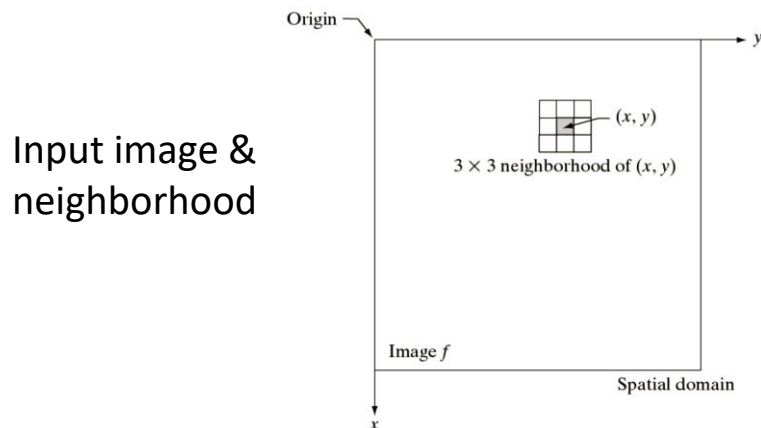
- (1) A local neighborhood in the input image (eg. a small square)
- (2) A predefined operation (T) that is performed on the pixels inside the neighborhood.

The predefined operations (T) is embodied in a *sub image* with the same dimensions as the neighborhood.



Spatial Filtering

- Typically, the neighborhood and sub-image are square and their size are much smaller than the image.
 - e.g., 3×3 or 5×5 pixels
 - The sub image is called a *mask*, *filter* or *kernel*.
 - The values in the mask are referred to as *coefficients* or *weights*, rather than pixels.
- The filtering is performed by shifting the mask over the whole image so that the center of the mask visits each pixel in the input image.

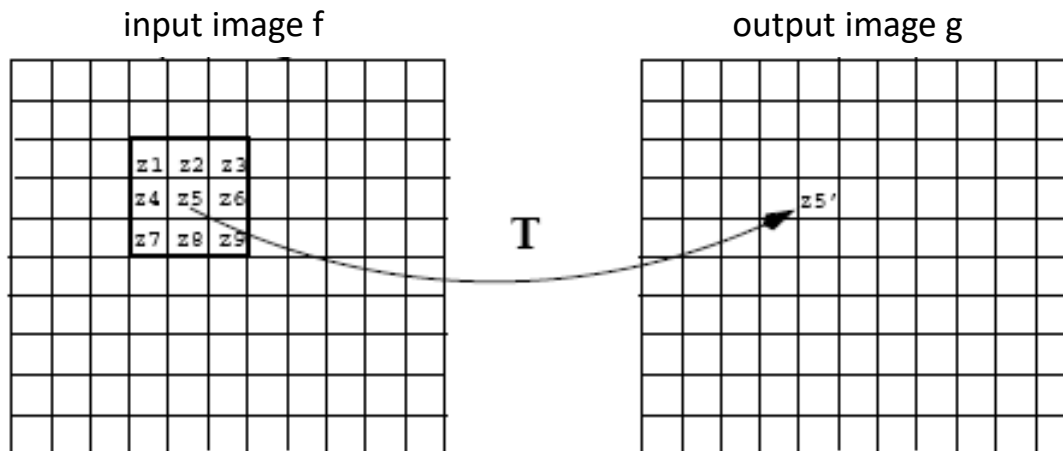


Spatial Filtering

Example: apply a mask of weights

w1	w2	w3
w4	w5	w6
w7	w8	w9

Area or Mask Processing Methods

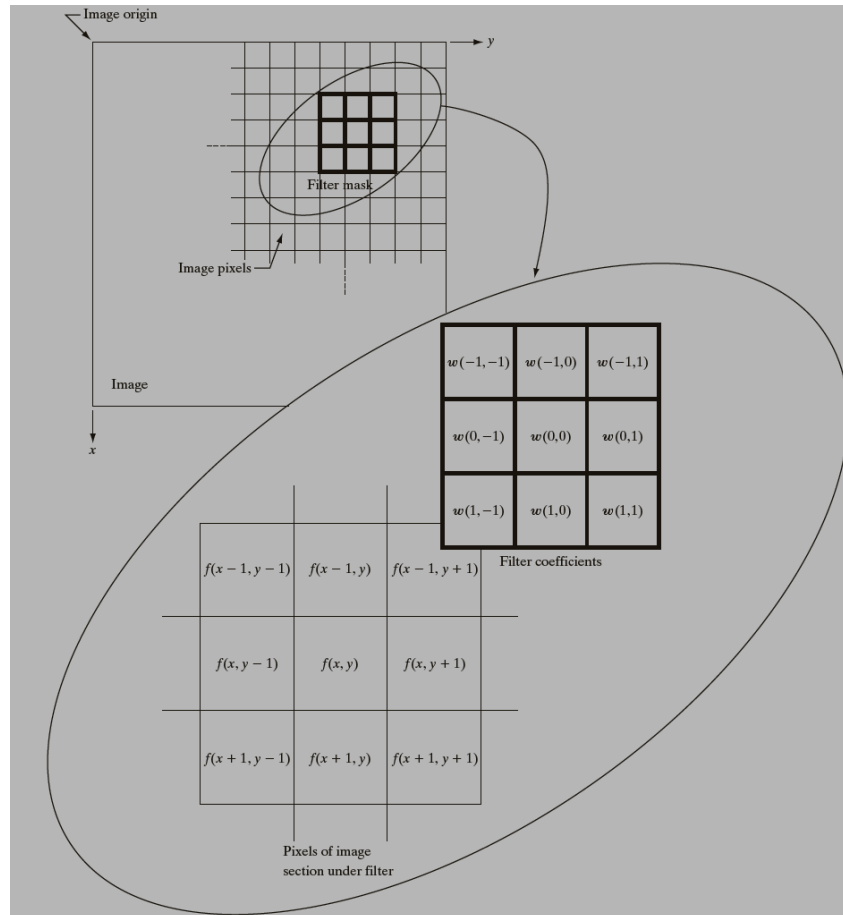


$$g(x,y) = T[f(x,y)]$$

T operates on a neighborhood of pixels

$$z5' = R = w1 * z1 + w2 * z2 + \dots + w9 * z9$$

Spatial Filtering



Assume the origin of the mask is the center of the mask.

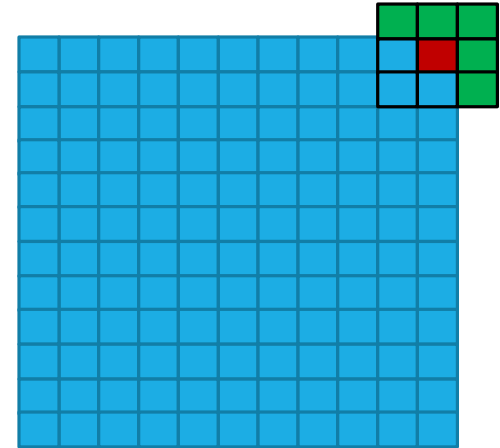
A filtered image is generated as the centre of the mask moves to every pixel in the input image.

Border problem

The values of pixels outside the image but involved in the filtering process need to be estimated.

Solutions

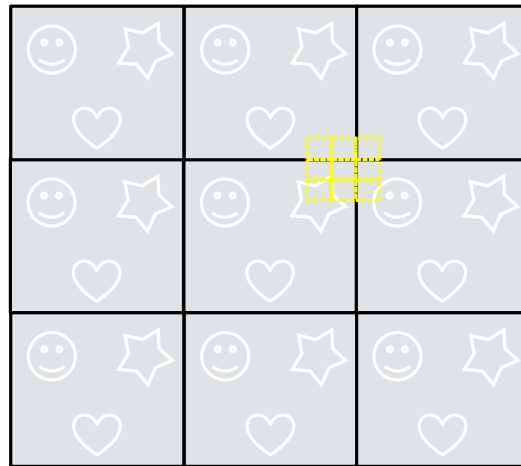
1. change filter size along the border
2. enlarge the image
 - Fill with zeros
 - Periodic extension of the image
 - Mirroring the borders.



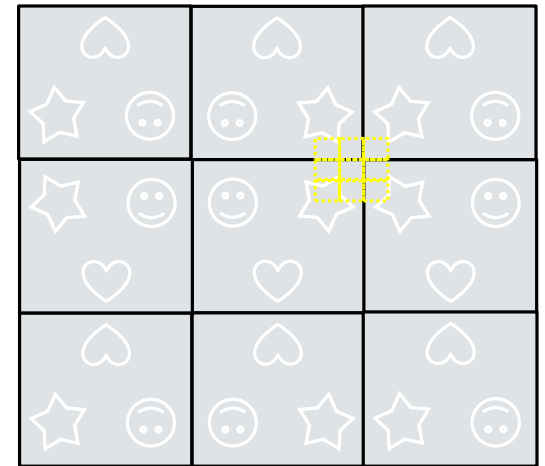
Border problem



Fill with zeros



Periodic extension



Mirroring

Linear vs Non-Linear Spatial Filtering

A filtering method is **linear** when the output is a weighted sum of the input pixels.

w1	w2	w3
w4	w5	w6
w7	w8	w9

$$z5' = R = w1 * z1 + w2 * z2 + \dots + w9 * z9$$

Methods that do not satisfy the above property are called **non-linear**.

- e.g.

$$z5' = \max(z_k, k=1, 2, \dots, 9)$$

Linear Spatial Filtering Methods

Two main linear spatial filtering methods:

- **Correlation**
 - The process of moving a filter mask over the image and computing the ***sum of products*** at each location.
 - Correlation is a function of displacement of the filter.
- **Convolution**
 - The same process except that the filter is firstly rotated by 180 degree.

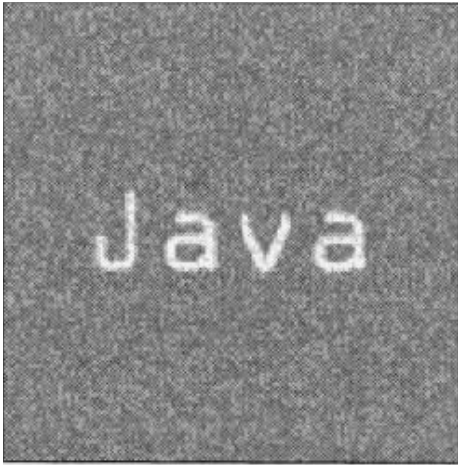
Correlation

The correlation of a filter $w(x, y)$ of size $m \times n$ with an image $f(x, y)$, denoted by $w(x, y) \bullet f(x, y)$

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

Where $a = (m - 1)/2$, $b = (n - 1)/2$, and m and n are odd integers.

Correlation



Often used in applications where we need to measure the similarity between images or parts of images (e.g., pattern matching).



Convolution

The convolution of a filter $w(x, y)$ of size $m \times n$ with an image $f(x, y)$, denoted by $w(x, y) \circ f(x, y)$

Rotate f by
180 degree

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

Note:

1. For notational simplicity, we **flip** f instead of w both horizontally and vertically.
2. If $w(x, y)$ is symmetric, that is $w(x, y) = w(-x, -y)$, then convolution is equivalent to correlation!

Convolution-2D illustration

f

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

w

1	2	3
4	5	6
7	8	9

*Mask already flipped and the origin of the mask is the center of the mask by default.

Convolution-2D illustration

Initial position of w

1	2	3				
4	5	6	0	0	0	
7	8	9	0	0	0	
	0	0	1	0	0	
	0	0	0	1	2	3
	0	0	0	4	5	6
				7	8	9

finish position of w

Convolution-2D illustration

Filling the zeros

1	2	3	0	0	0	0
4	5	6	0	0	0	0
7	8	9	0	0	0	0
0	0	0	1	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

The convolution

	0					

$$0*1 + 0*2 + 0*3 + 0*4 + 0*5 + 0*6 + 0*7 + 0*8 + 0*9 = 0$$

Convolution-2D illustration

0	1	2	3	0	0	0
0	4	5	6	0	0	0
0	7	8	9	0	0	0
0	0	0	1	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

The convolution

	0	0				

$$0*1 + 0*2 + 0*3 + 0*4 + 0*5 + 0*6 + 0*7 + 0*8 + 0*9 = 0$$

Convolution-2D illustration

0	0	1	2	3	0	0
0	0	4	5	6	0	0
0	0	7	8	9	0	0
0	0	0	1	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

The convolution

	0	0	0			

$$0*1 + 0*2 + 0*3 + 0*4 + 0*5 + 0*6 + 0*7 + 0*8 + 0*9 = 0$$

Convolution-2D illustration

0	0	1	2	3	0	0
0	0	4	5	6	0	0
0	0	7	8	9	0	0
0	0	0	1	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

The convolution

	0	0	0			

$$0*1 + 0*2 + 0*3 + 0*4 + 0*5 + 0*6 + 0*7 + 0*8 + 0*9 = 0$$

Convolution-2D illustration

0	0	0	0	0	0	0
1	2	3	0	0	0	0
4	5	6	0	0	0	0
7	8	9	1	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

The convolution

	0	0	0	0	0	
	0					

$$0*1 + 0*2 + 0*3 + 0*4 + 0*5 + 0*6 + 0*7 + 0*8 + 0*9 = 0$$

Convolution-2D illustration

0	0	0	0	0	0	0
0	1	2	3	0	0	0
0	4	5	6	0	0	0
0	7	8	9	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

The convolution

	0	0	0	0	0	
	0	9				

$$0*1 + 0*2 + 0*3 + 0*4 + 0*5 + 0*6 + 0*7 + 0*8 + 1*9 = 9$$

Convolution-2D illustration

0	0	0	0	0	0	0
0	0	1	2	3	0	0
0	0	4	5	6	0	0
0	0	7	8	9	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

The convolution

	0	0	0	0	0	
	0	9	8			

$$0*1 + 0*2 + 0*3 + 0*4 + 0*5 + 0*6 + 0*7 + 1*8 + 0*9 = 8$$

Convolution-2D illustration

0	0	0	0	0	0	0
0	0	0	1	2	3	0
0	0	0	4	5	6	0
0	0	0	7	8	9	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

The convolution

	0	0	0	0	0	
	0	9	8	7		

$$0*1 + 0*2 + 0*3 + 0*4 + 0*5 + 0*6 + 1*7 + 0*8 + 0*9 = 7$$

Convolution-2D illustration

0	0	0	0	0	0	0
0	0	0	0	1	2	3
0	0	0	0	4	5	6
0	0	0	1	7	8	9
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

The convolution

	0	0	0	0	0	
	0	9	8	7	0	

$$0*1 + 0*2 + 0*3 + 0*4 + 0*5 + 0*6 + 0*7 + 0*8 + 1*9 = 9$$

Convolution-2D illustration

0	0	0	0	0	0	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	0	1	2	3
0	0	0	0	4	5	6
0	0	0	0	7	8	9

The convolution

	0	0	0	0	0	
	0	9	8	7	0	
	0	6	5	4	0	
	0	3	2	1	0	
	0	0	0	0	0	

$$0*1 + 0*2 + 0*3 + 0*4 + 0*5 + 0*6 + 0*7 + 0*8 + 0*9 = 0$$

Convolution-2D illustration

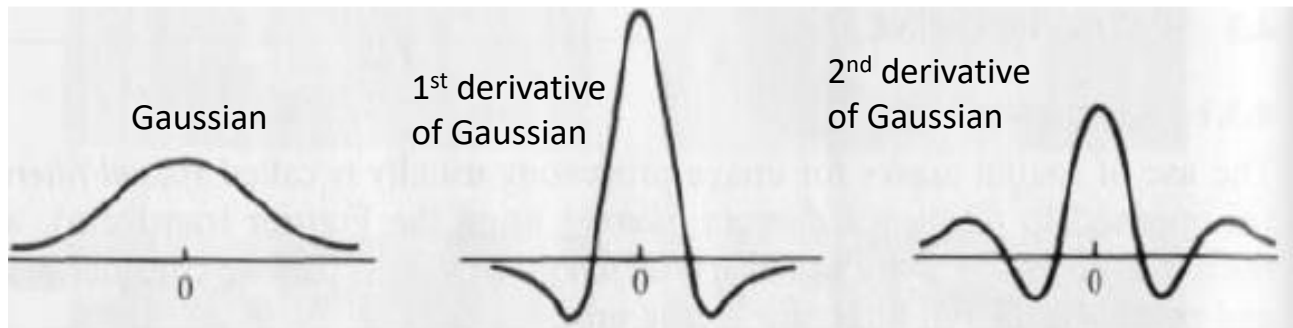
The cropped convolution

0	0	0	0	0
0	9	8	7	0
0	6	5	4	0
0	3	2	1	0
0	0	0	0	0

How to choose the elements of a mask /filter?

Typically, by sampling certain functions:

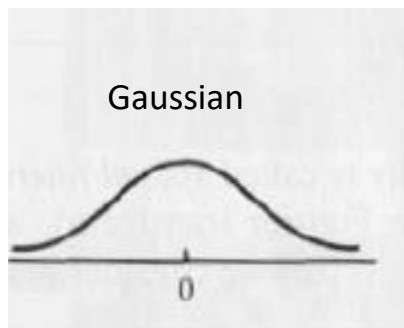
w1	w2	w3
w4	w5	w6
w7	w8	w9



Smoothing Filters (Low-pass)

Smoothing (i.e., low-pass filters)

- Filter-out the high frequencies.
- For blurring and noise reduction. e.g. removing small details prior to object extraction and bridging of small gaps in lines or curves.
- The elements of the mask must be positive.
- Sum of mask elements is 1.



Smoothing Filters: Averaging

Box filter (size $m \times n$)

$$W = \frac{1}{mn} \sum_{i=1}^{mn} w_i$$

Normalization is needed to conserve the total energy of the image.
(the sum of all intensity levels)

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

$$\frac{1}{25} \times \begin{array}{|c|c|c|c|c|} \hline 1 & 1 & 1 & 1 & 1 \\ \hline 1 & 1 & 1 & 1 & 1 \\ \hline 1 & 1 & 1 & 1 & 1 \\ \hline 1 & 1 & 1 & 1 & 1 \\ \hline 1 & 1 & 1 & 1 & 1 \\ \hline \end{array}$$

[illegible]

Smoothing Filters: Averaging

Mask size determines the degree of smoothing and loss of details.

original



3x3



5x5



7x7

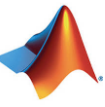


15x15



25x25

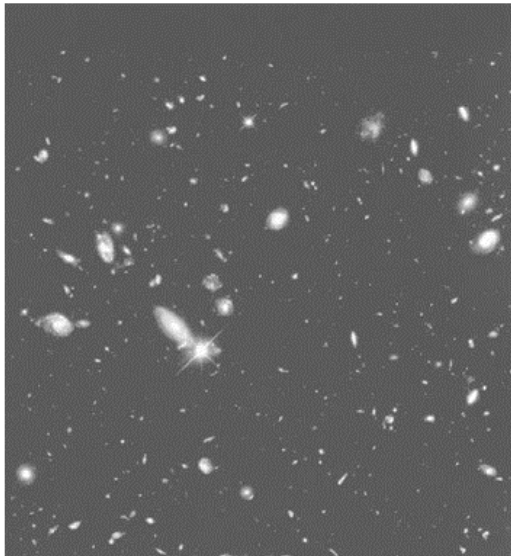




Smoothing Filters: Averaging

Example: extract largest and brightest objects.

original



15 x 15 averaging

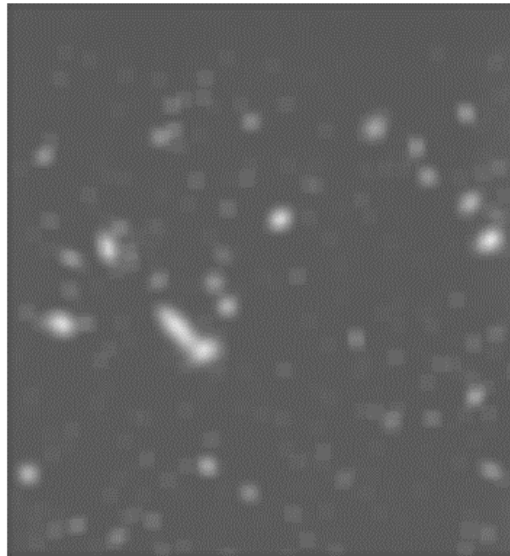
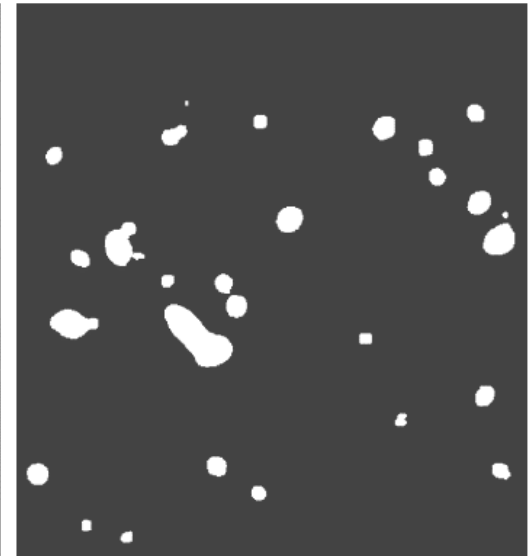


image thresholding



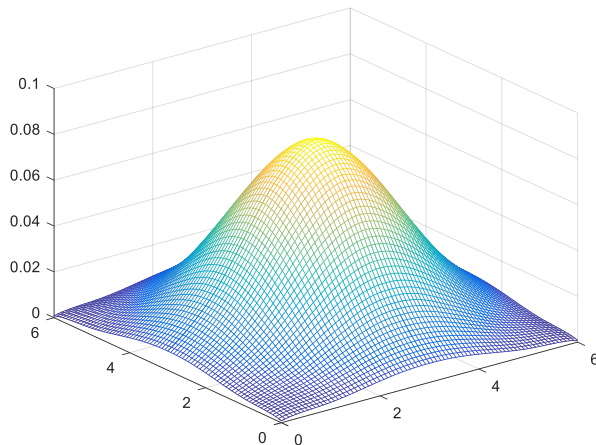
Smoothing filters: Gaussian

The weights are samples of the Gaussian function

$$G_{\sigma} = \frac{1}{2\pi\sigma^2} \exp^{-\frac{x^2+y^2}{2\sigma^2}}$$

mask size (height or width) $\geq 5\sigma$ (subtends 98.76% of the area.)

e.g. $\sigma = 1.4$, mask size=7×7



7×7 Gaussian mask

1	1	2	2	2	1	1
2	1	2	2	4	2	2
3	2	2	4	8	4	2
4	2	4	8	16	8	4
5	1	2	4	8	4	2
6	1	2	2	4	2	1
7	1	1	2	2	2	1
1	2	3	4	5	6	7

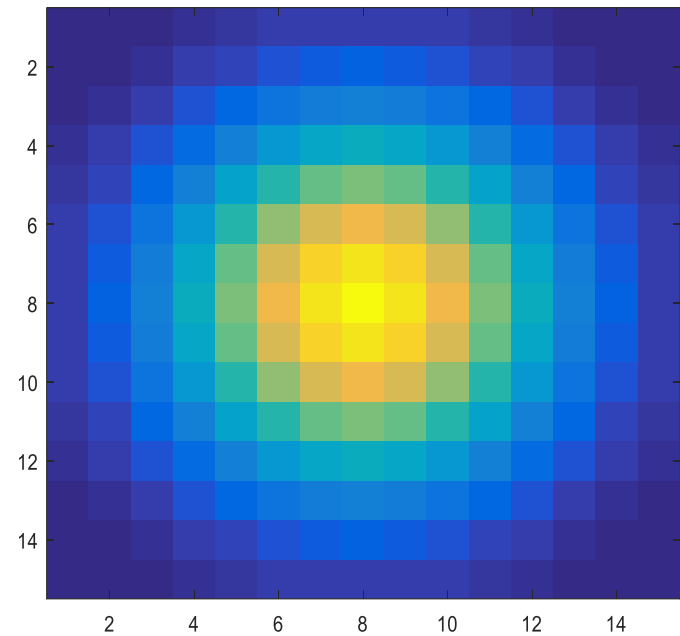
Smoothing filters: Gaussian

- As σ increases, more samples must be obtained to represent the Gaussian function accurately.
- σ controls the amount of smoothing.

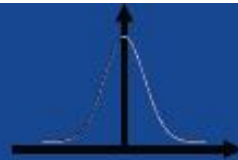
$\sigma = 3$

Mask size=15×15

2	2	3	4	5	5	6	6	6	5	5	4	3	2	2
2	3	4	5	7	7	8	8	8	7	7	5	4	3	2
3	4	6	7	9	10	10	11	10	10	9	7	6	4	3
4	5	7	9	10	12	13	13	13	12	10	9	7	5	4
5	7	9	11	13	14	15	16	15	14	13	11	9	7	5
5	7	10	12	14	16	17	18	17	16	14	12	10	7	5
6	8	10	13	15	17	19	19	19	17	15	13	10	8	6
6	8	11	13	16	18	19	20	19	18	16	13	11	8	6
6	8	10	13	15	17	19	19	19	17	15	13	10	8	6
5	7	10	12	14	16	17	18	17	16	14	12	10	7	5
5	7	9	11	13	14	15	16	15	14	13	11	9	7	5
4	5	7	9	10	12	13	13	13	12	10	9	7	5	4
3	4	6	7	9	10	10	11	10	10	9	7	6	4	3
2	3	4	5	7	7	8	8	8	7	7	5	4	3	2
2	2	3	4	5	5	6	6	6	5	5	4	3	2	2



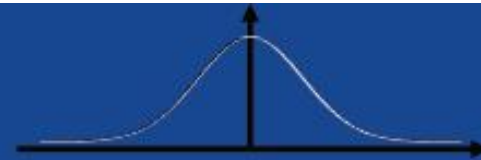
Smoothing filters: Gaussian



small σ



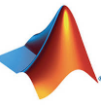
limited smoothing



large σ



strong smoothing



Averaging vs Gaussian



Averaging



Gaussian

Smoothing Filters: Median

- Non-linear filter
- The response is based on ordering (ranking) the pixels contained in the image area encompassed by the filter.
- Very effective for removing “salt and pepper” noise (i.e., random occurrences of black and white pixels).

WHY?

Image

Image with noise

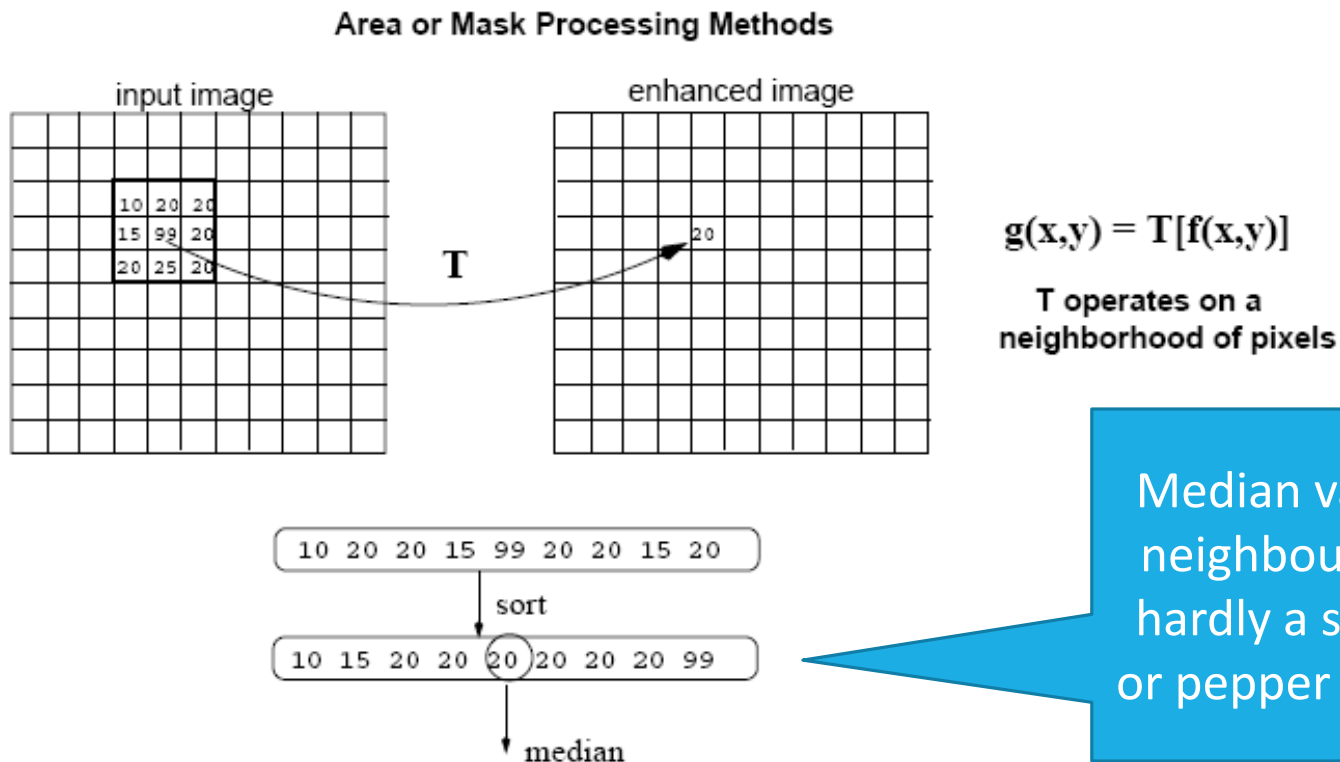
Averaging

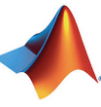
Median filter



Smoothing Filters: Median

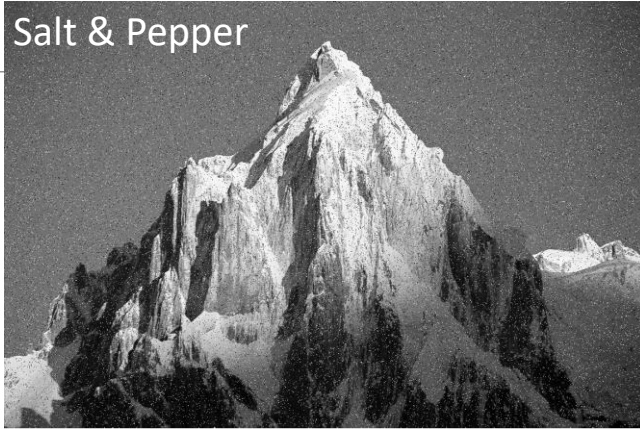
Replace each pixel by the median in a neighborhood around the pixel.



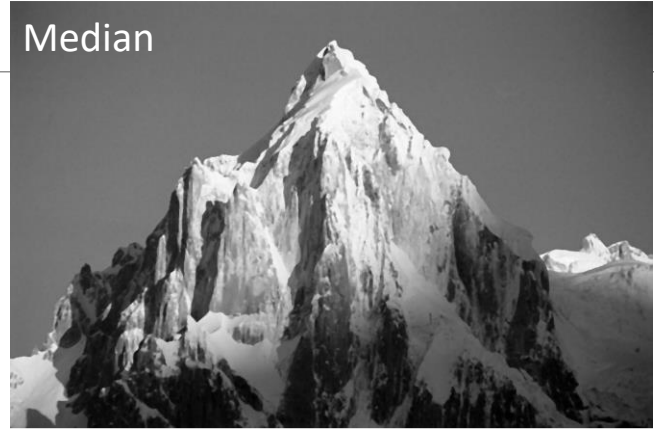


Comparison

Salt & Pepper



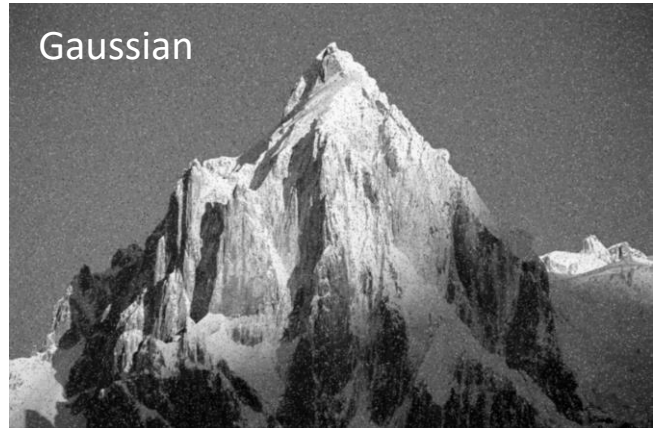
Median



Averaging



Gaussian

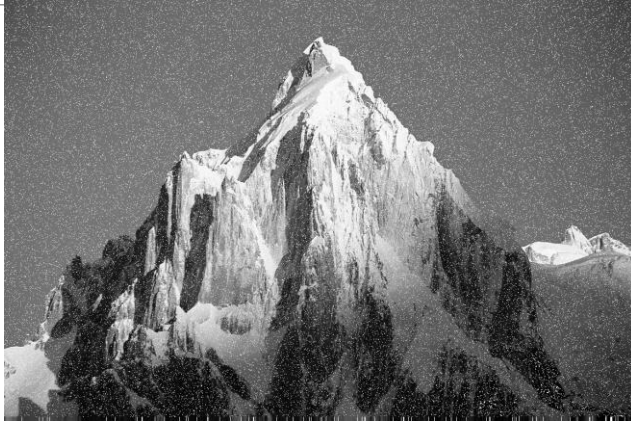


Smoothing Filters: Max / Min

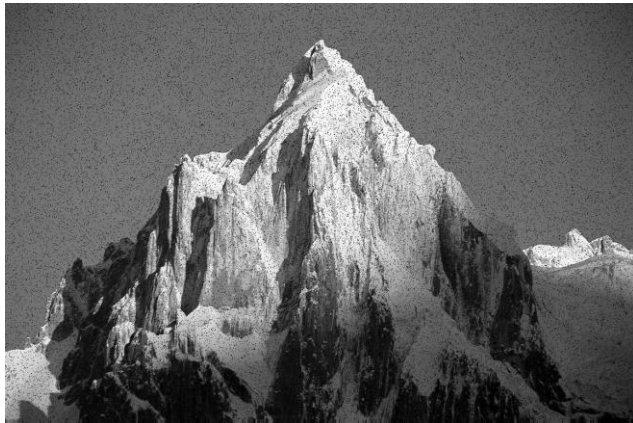
- A max filter or a min filter work similarly as a median filter.
- Non-linear filters
- Max filter
 - takes the max value of a neighbourhood.
 - for images with pepper (0) noise.
- Min filter
 - takes the min value of a neighbourhood.
 - for images with salt (255) noise.

Comparison

Original



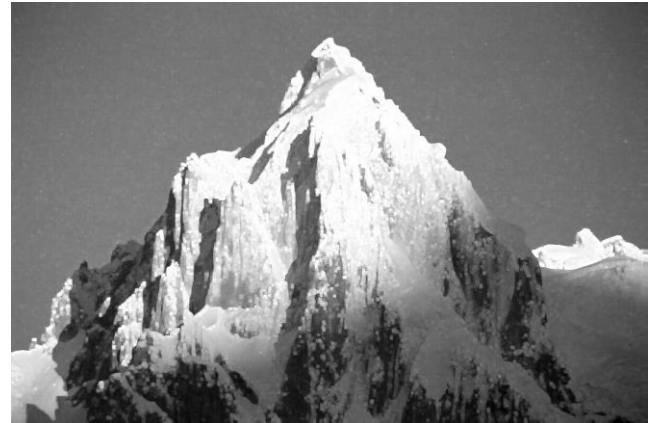
Original

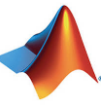


after min filtering



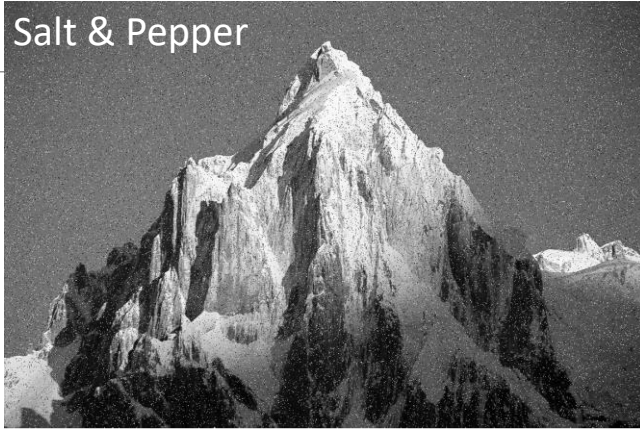
after max filtering



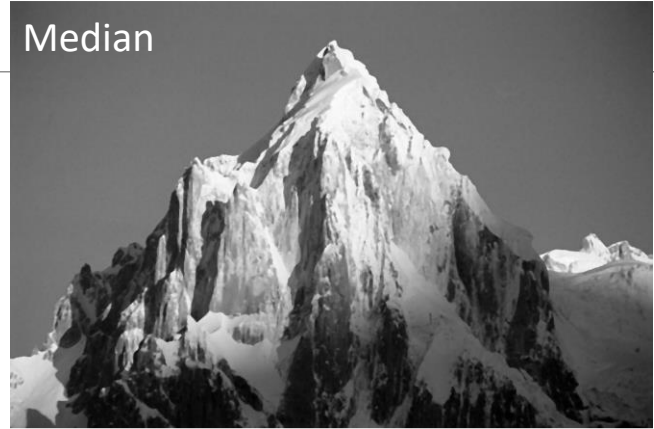


Comparison

Salt & Pepper



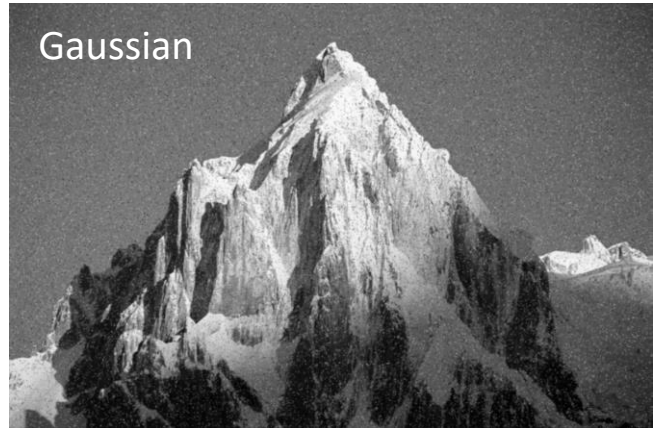
Median



Averaging



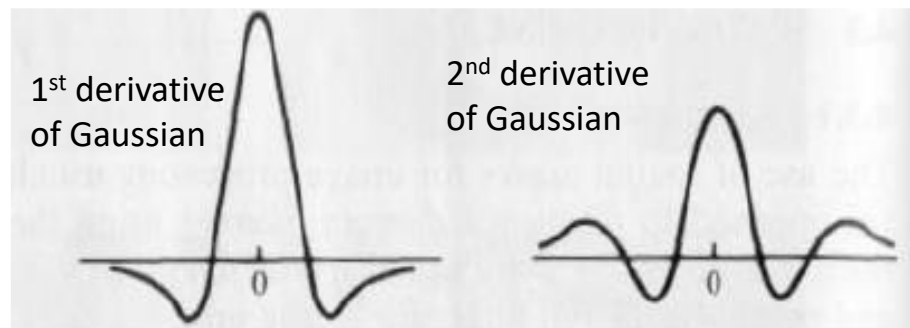
Gaussian



2. Sharpening Filters (High-Pass)

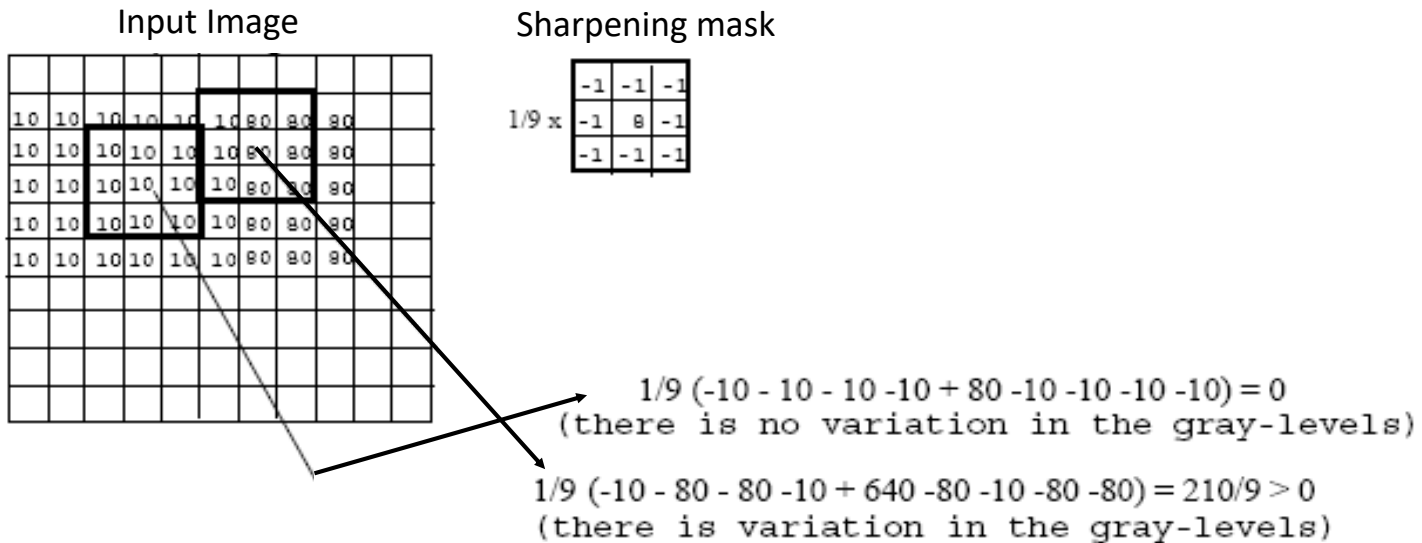
Sharpening (i.e., high-pass filters)

- Highlight fine details or enhance detail that has been blurred.
- The elements of the mask contain both positive and negative weights.
- Sum of the mask weights is 0.



Sharpening Filters

Useful for emphasizing transitions in image intensity (e.g., edges).



Sharpening Filters: Unsharp Masking

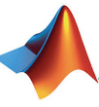
Steps:

1. blur the original image by low-pass filtering.
2. subtract the blurred image from the original (the difference is the Mask).

$$\text{Mask}_{UM} = \text{Original} - \text{Lowpass}$$


3. add the mask to the original.

$$\text{Enhanced image}_{UM} = \text{Original} + \text{Mask}$$

Sharpening Filters: Unsharp Masking

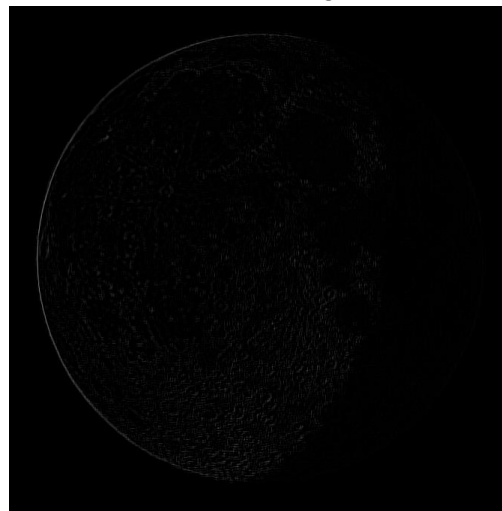
Note that the response of high-pass filtering might be negative.

Values must be re-mapped to $[0, 255]$

Original



Mask_{UM}



Enhanced image_{UM}



Sharpening Filters: High boost

Image sharpening emphasizes edges but details (i.e., low frequency components) might be lost.

High boost filter: amplify input image, then subtract a lowpass image.

$$\mathbf{Mask}_{HB} = A * \mathbf{Original} - \mathbf{Lowpass}$$

$$= (A-1) * \mathbf{Original} + (\mathbf{Original} - \mathbf{Lowpass})$$

$$= (A-1) * \mathbf{Original} + \mathbf{Mask}_{UM}$$


$$\text{TEXT} = (A-1) * \text{TEXT} + \text{TEXT}$$

Sharpening Filters: High boost

If **A=1**, we have unsharp masking.

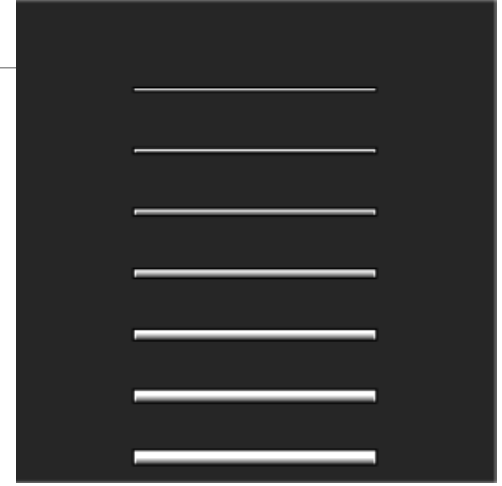
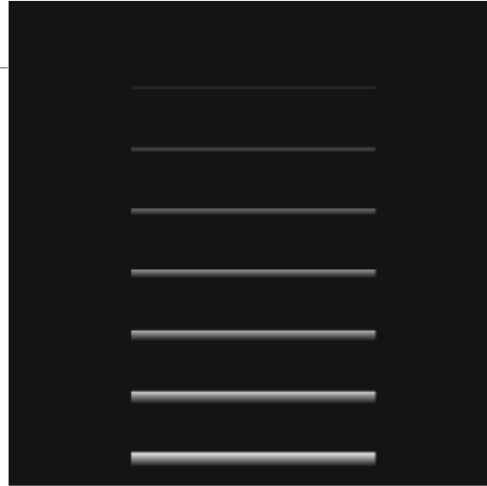
If **A>1**, part of the original image is added back to the high pass filtered image.

If **A<1**, the contribution of the unsharp mask is de-emphasized.

$$W_{\text{highboost}} = \begin{array}{|c|c|c|} \hline -1 & -1 & -1 \\ \hline -1 & 9A-1 & -1 \\ \hline -1 & -1 & -1 \\ \hline \end{array} \times \frac{1}{9}$$

Sharpening Filters: High boost

A=1.9



A=1.4



Summary

- Basics on spatial filtering
 - Border problem
 - Convolution and correlation
- Smoothing filter
 - Averaging
 - Gaussian
 - Median
- Sharpening filter
 - Unsharp masking
 - High-boost

Q&A
