DIGITAL IMAGE PROCESSING-1

Unit 3: Lecture 35

Dr. Shikha Tripathi
Department of Electronics &
Communication Engineering

DIGITAL IMAGE PROCESSING-1

Unit 3: Image Enhancement

Dr. Shikha Tripathi
Department of Electronics & Communication Engineering

Last Session

- Spatial / Neighborhood processing Cont…
  - Convolution
  - Correlation
  - Averaging
  - Filtering

Today's Session

- Spatial / Neighborhood processing Cont…
  - Highpass filtering

Types of Enhancement Techniques

Image Enhancement

Point Operations

- Contrast Stretching
- Noise Clipping
- Window Slicing
- Histogram Modeling

Spatial Operations

- Noise Smoothing
- Median Filtering
- Unsharp Masking
- LP,HP,BP filtering

Transform Operations

- Linear filtering
-
- Homomorphic Filtering

Pseudocoloring

- False Coloring
- Pseudocoloring

Spatial Domain

Types of Spatial Filters

- **Spatial Filters**

  ✓Convolution based filters

  ✓Order-statistics(rank) filters

  - Hybrid filters

Types of Local Operations

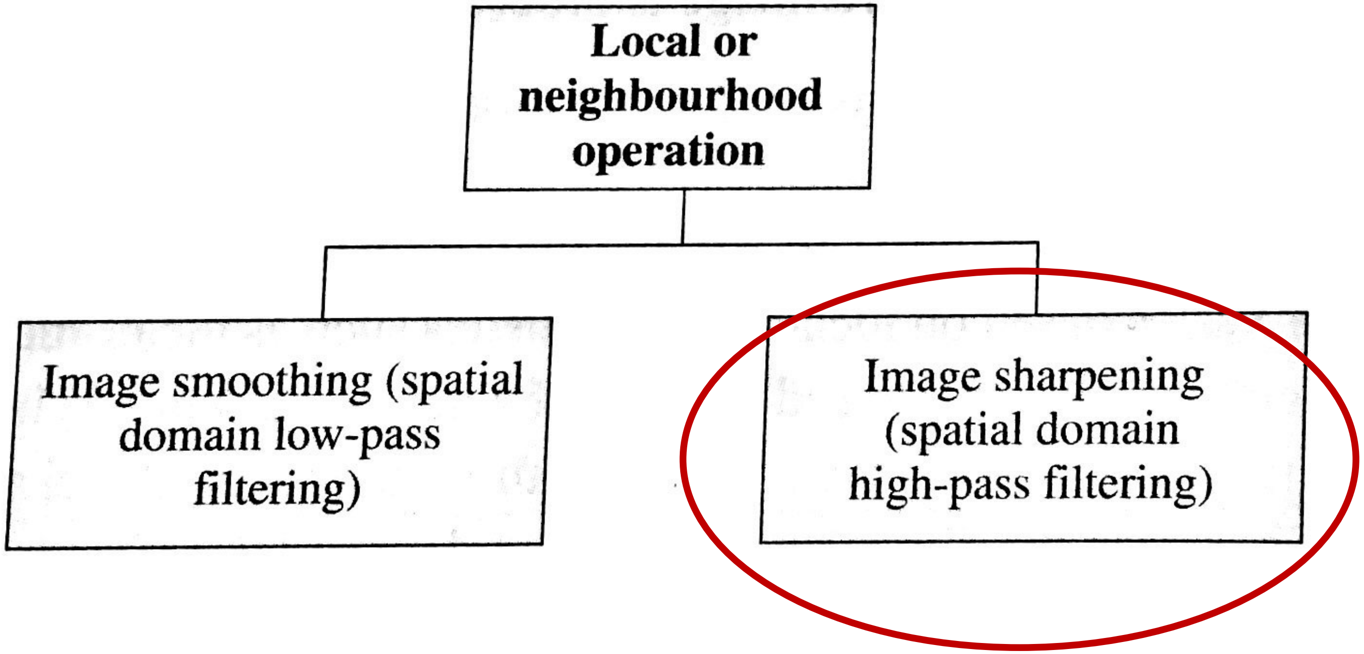


Image Sharpening (Highpass Filtering)

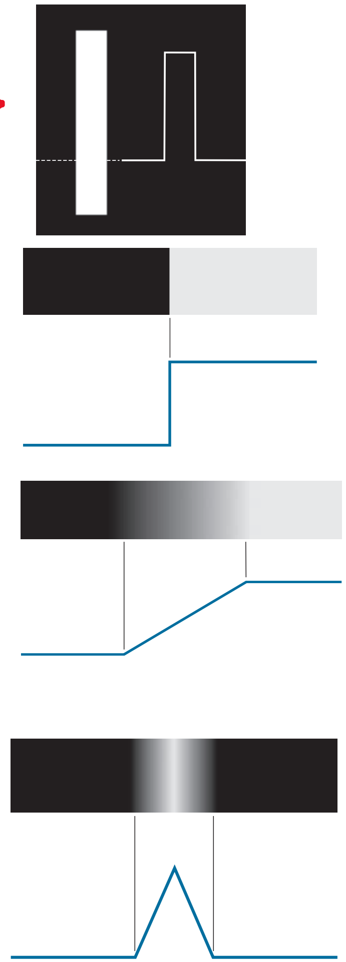
- Sharpening highlights transitions in intensity (highlights the fine details and edges in an image
- High frequency spatial components have detailed information
- In edges and boundaries
  - Edges are significant intensity variations that exist between two different regions
  - Provide outline of objects
- Image sharpening algorithms are used to separate object outlines
  - Edge enhancement or edge crispening algorithms

DIGITAL IMAGE PROCESSING-1

Edge Detection using Spatial Filters

Different edge profiles:

- Line Edge
- Step edge
- Ramp Edge
- Roof Edge



Gradient Filters

- Edges can be extracted by taking gradient(difference between pixels) of the image
- The derivation/differentiation operation enhances the degree of discontinuity
  - If neighboring pixels have same intensity, gradient will be zero and hence no edge
  - Edge exists only at points where there is a significant local intensity variation
  - By comparing this with a threshold we can identify if it is a significant edge point or not

Sharpening Filters

- Based on first and second order derivatives
    - Derivatives of a digital function is defined in terms of differences
- For 1-D Signals

1st Order derivative is $\dfrac{\partial f}{\partial x}$

2nd Order derivative is $\dfrac{\partial^2 f}{\partial x^2}$

- First derivative:
  - Must be zero in in areas of constant intensity (flat segments)
  - Must be non zero at onset of an intensity step or ramp
  - Must be non zero along intensity ramps

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

- Second Derivative:
  - Must be zero in in areas of constant intensity (flat segments)
  - Must be non zero at onset ***and*** end of a gray level step or ramp
  - Must be zero along ramps of constant slopes

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

$$= f(x+1) + f(x-1) - 2f(x)$$

1. Determine the first and second order derivatives of:

a) f(x)=[4 3 2 5 9]

b) f(x)=[6 6 6 6 5 4 3 2 1 1 1 1 6 1 1 1]

Ans:

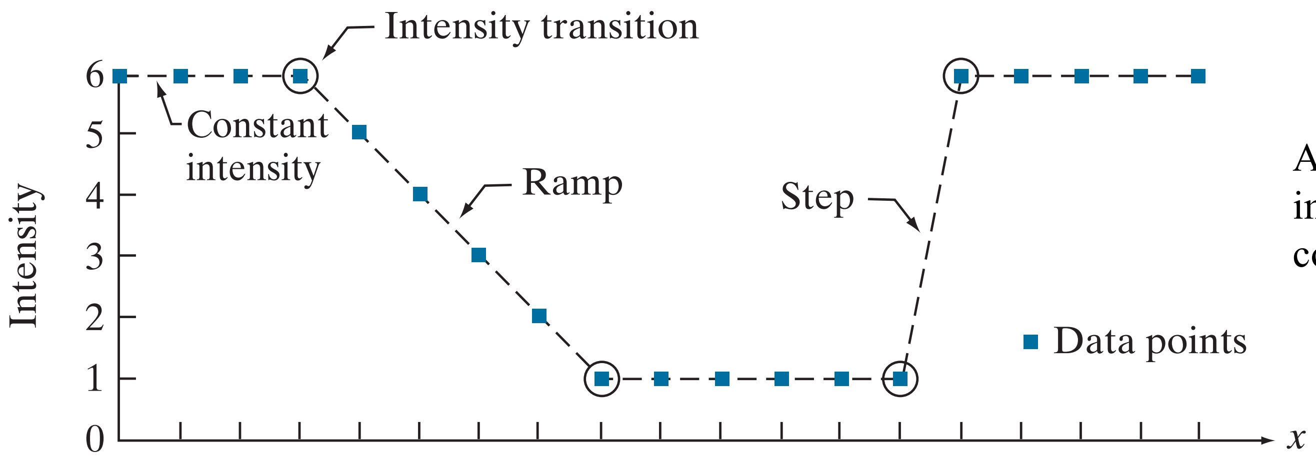
a). $\frac{\partial f}{\partial x}$ = [-1 -1 3 4].

$\frac{\partial^2 f}{\partial x^2}$ = [0 4 1]

b)). $\frac{\partial f}{\partial x}$ = [0 0 0 -1 -1 -1 -1 -1 0 0 0 5 -5 0 0]

$\frac{\partial^2 f}{\partial x^2}$ = [0 0 -1 0 0 0 0 1 0 0 5 -10 5 0]

Examples



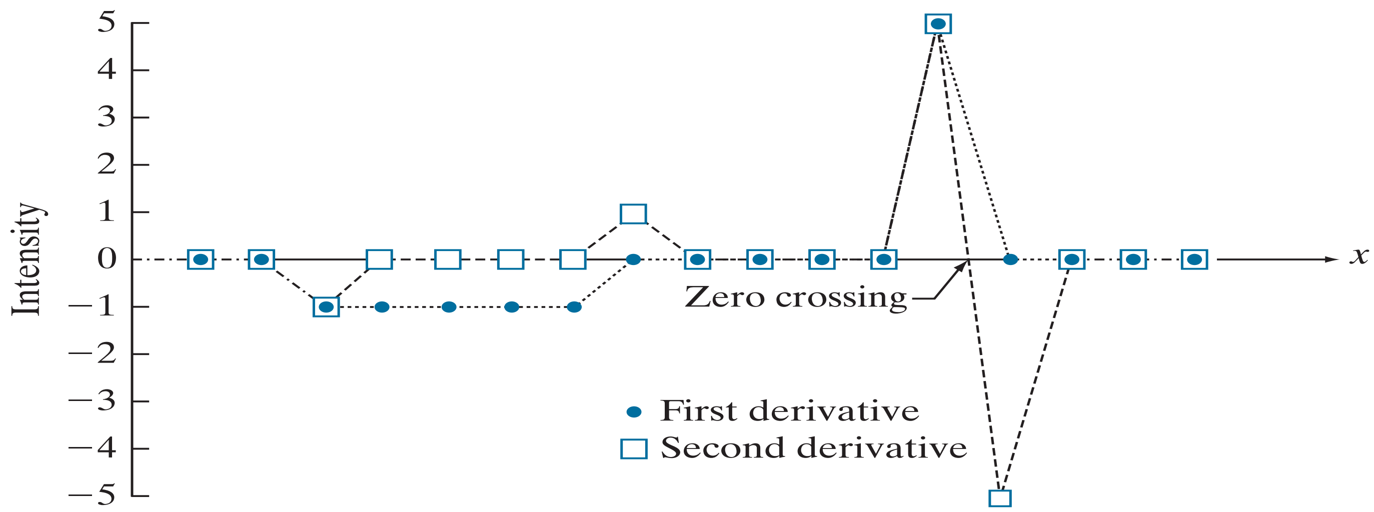
A section of a horizontal scan line from an image, showing ramp and step edges, as well as constant segments.

Values of scan line	6	6	6	6	5	4	3	2	1	1	1	1	1	1	6	6	6	6	6
1st derivative		0	0	−1	−1	−1	−1	−1	0	0	0	0	0	5	0	0	0	0	
2nd derivative		0	0	−1	0	0	0	0	1	0	0	0	0	5	−5	0	0	0	

Values of the scan line and its derivatives.

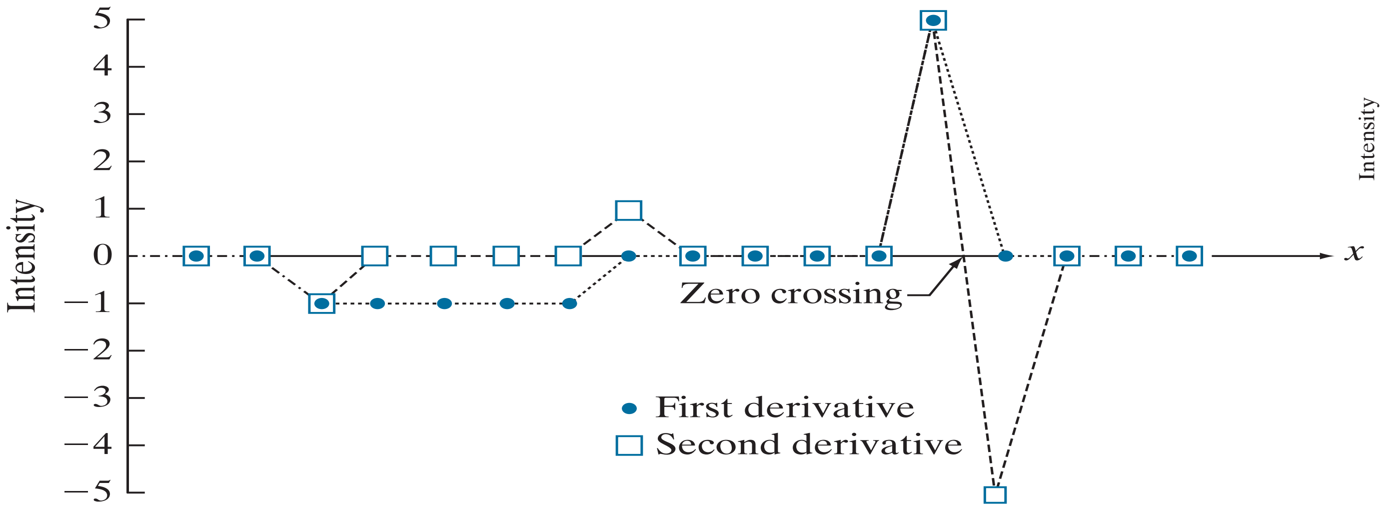
Examples

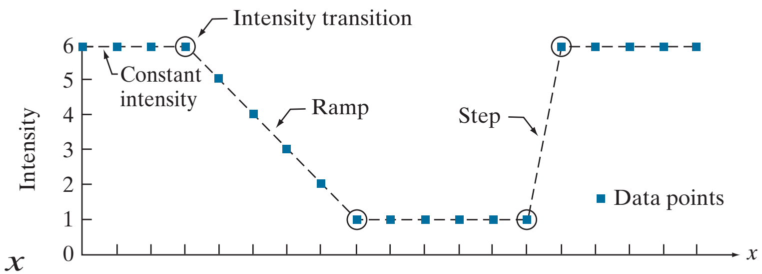
1st derivative	0	0	−1	−1	−1	−1	−1	0	0	0	0	0	5	0	0	0	0
2nd derivative	0	0	−1	0	0	0	0	1	0	0	0	0	5	−5	0	0	0



Plot of the derivatives, showing a zero crossing.

Examples





- As we traverse the profile from left to right we encounter first an area of constant intensity and, both derivatives are zero there, so condition (1) is satisfied by both.
- Next, we encounter an intensity ramp followed by a step, and we note that the first-order derivative is nonzero at the onset of the ramp and the step
- Similarly, the second derivative is nonzero at the onset and end of both the ramp and the step; therefore, property (2) is satisfied by both derivatives.
- Property (3) is satisfied also by both derivatives because the first derivative is nonzero and the second is zero along the ramp.
- **Note that the sign of the second derivative changes at the onset and end of a step or ramp.**

Observations

- **In a step transition a line joining these two values crosses the horizontal axis midway between the two extremes. This *zero crossing* property is quite useful for locating edges,**
- For a noisy pixel, 1st derivative produces one zero crossing, whereas 2nd derivative produces 2 zero crossings, making it easier to identify
- Edges in digital images often are ramp-like transitions in intensity, in which case the first derivative of the image would result in thick edges because the derivative is nonzero along a ramp.
- On the other hand 2nd derivative would produce double edge one pixel thick, separated by zeros.
  - **2nd derivative enhances fine detail(edge) better than 1st derivative**

Sharpening Filters

- As images are 2 Dimensional,

  gradient vector of an image f(x,y) is also 2 dimensional
- The gradient of an image f(x,y) at a location (x,y) is a vector that consists of partial derivatives of f(x,y)

First Derivative (Gradient)

- For a 2-D function f(x,y), the gradient (1st derivative) is defined as

$$\nabla f(x,y) = \begin{bmatrix} \dfrac{\partial f(x,y)}{\partial x} \\ \\ \dfrac{\partial f(x,y)}{\partial y} \end{bmatrix} \qquad \nabla f(x,y) = \begin{bmatrix} g_x \\ \\ g_y \end{bmatrix}$$

First Derivative (Gradient)

- Magnitude of this vector (or length of gradient or norm) is given by

$$\nabla f(x,y) = \text{mag}(\nabla f(x,y)) = \left[(g_x)^2 + (g_y)^2\right]^{1/2}$$

- Gradient direction can be given by

$$\theta = \tan^{-1}\left(\frac{g_y}{g_x}\right)$$

Laplacian for Image Sharpening

- 2nd Order derivative for image sharpening :

  The Laplacian

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

Laplacian for Image Sharpening

$f(x-1, y-1)$	$f(x-1, y)$	$f(x-1, y+1)$
$f(x, y-1)$	$f(x, y)$	$f(x, y+1)$
$f(x+1, y-1)$	$f(x+1, y)$	$f(x+1, y+1)$

In the *x*-direction, we have

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

similarly, in the *y*-direction, we have

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

The discrete Laplacian of two variables is

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

$$= f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y)$$

Highpass Masks

This equation can be implemented using convolution with the following kernel

0	1	0
1	−4	1
0	1	0

This kernel is isotropic for rotations in increments of 90° with respect to the *x*- and *y*-axes.

4-neighborhood Laplacian Mask

The diagonal directions can be incorporated in the definition of the digital Laplacian by adding four more terms

1	1	1
1	−8	1
1	1	1

Because each diagonal term would contains a $-2 f(x, y)$ term, the total subtracted from the difference terms now would be $-8 f(x,y)$

This kernel yields isotropic results in increments of 45°

8-neighborhood Laplacian Mask

Highpass Masks (Alternate Versions)

0	−1	0
−1	4	−1
0	−1	0

4-neighborhood Laplacian Mask

−1	−1	−1
−1	8	−1
−1	−1	−1

8-neighborhood Laplacian Mask

-1	-2	-1
-2	12	-2
-1	-2	-1

8-neighborhood weighted Laplacian Mask

- These kernels are also used to compute the Laplacian.
- They are obtained from definitions of the second derivatives that are the negatives of the ones used earlier.
- They yield equivalent results, but the difference in sign must be kept in mind when combining a Laplacian-filtered image with another image.

Note: Addition of all the coefficients of the mask = 0

Image Sharpening using Laplacian Mask

- The basic way in which we use the Laplacian for image sharpening is

$$g(x,y) = f(x,y) + c\left[\nabla^2 f(x,y)\right]$$

where $f(x, y)$ and $g(x, y)$ are the input and sharpened images, respectively.

- **It is important to keep in mind which definition of the Laplacian is used.**

  We let $c = -1$ if the Laplacian kernels with negative center is used, and $c = 1$ if either of the other two kernels is used.

- If the definition used has a negative center coefficient, then we *subtract* the Laplacian image from the original to obtain a sharpened result.
- **Because the Laplacian is a derivative operator, it highlights sharp intensity transitions in an image and de-emphasizes regions of slowly varying intensities.**

Image Sharpening using Laplacian Mask

Neighborhood Pixels Processing

Ex 5) 8x8 Pseudo image with a single edge (High Frequency) of 10 & 100. Sharpen the image 3x3 size High pass filter mask.

10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
100	100	100	100	100	100	100	100
100	100	100	100	100	100	100	100
100	100	100	100	100	100	100	100
100	100	100	100	100	100	100	100

Image Sharpening using Laplacian Mask

Neighborhood Pixels Processing

10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
100	100	100	100	100	100	100	100
100	100	100	100	100	100	100	100
100	100	100	100	100	100	100	100
100	100	100	100	100	100	100	100

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

-10-10-10-10-10-10-10-10+80 = 0

Image Sharpening using Laplacian Mask

Neighborhood Pixels Processing

10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
100	100	100	100	100	100	100	100
100	100	100	100	100	100	100	100
100	100	100	100	100	100	100	100
100	100	100	100	100	100	100	100

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

-10-10-10-10-10-10-10-10+80 = 0

Image Sharpening using Laplacian Mask

Neighborhood Pixels Processing

10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
100	100	100	100	100	100	100	100
100	100	100	100	100	100	100	100
100	100	100	100	100	100	100	100
100	100	100	100	100	100	100	100

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

-10-10-10-10-10-100-100-100+80 = -270

Image Sharpening using Laplacian Mask

Neighborhood Pixels Processing

10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
100	100	100	100	100	100	100	100
100	100	100	100	100	100	100	100
100	100	100	100	100	100	100	100
100	100	100	100	100	100	100	100

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

-10-10-10-100-100-100-100-100+800 = +270

Image Sharpening using Laplacian Mask

Neighborhood Pixels Processing

10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
100	100	100	100	100	100	100	100
100	100	100	100	100	100	100	100
100	100	100	100	100	100	100	100
100	100	100	100	100	100	100	100

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

-100-100-100-100-100-100-100-100+800 = 0

Image Sharpening using Laplacian Mask

Neighborhood Pixels Processing

10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10
100	100	100	100	100	100	100	100
100	100	100	100	100	100	100	100
100	100	100	100	100	100	100	100
100	100	100	100	100	100	100	100

0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
-270	-270	-270	-270	-270	-270	-270	-270
270	270	270	270	270	270	270	270
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

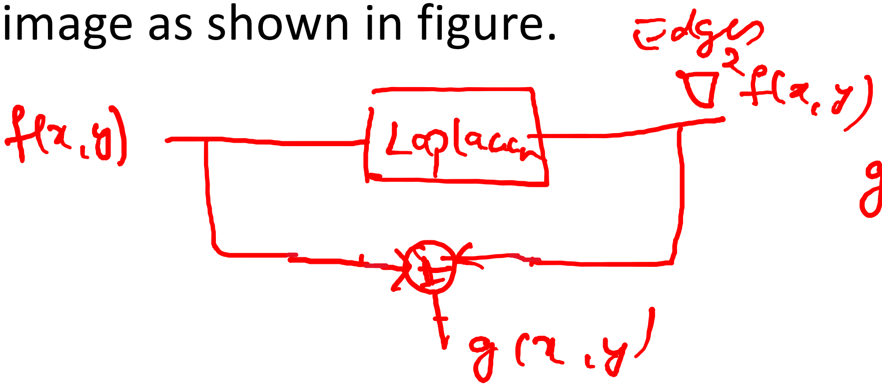
$$g(x,y) = f(x,y) + \left[\nabla^2 f(x,y)\right]$$

Note: -270 is replaced by 0.

PES UNIVERSITY

Unsharp Masking

- The unsharp masking techniques is used commonly in printing industry since 1930s for crispening the edges.
- It is applied by subtracting an unsharp or smoothed or low-pass filtered version of an image from the original image.
- A sharpened image is thus generated by subtracting blurred (lowpass filtered/smoothed) image from original image
- It is equivalent to adding the gradient, or high-pass signal to the blurred image as shown in figure.



$g(x,y) = f(x,y)$

$\pm \nabla^2 f(x,y)$

Unsharp Masking

It consists of following steps

1. Blur the original image.
2. Subtract the blurred image from the original (the resulting difference is called he *mask*.)

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

3. Then add a weighted portion of the mask back to the original image to give a generalized expression

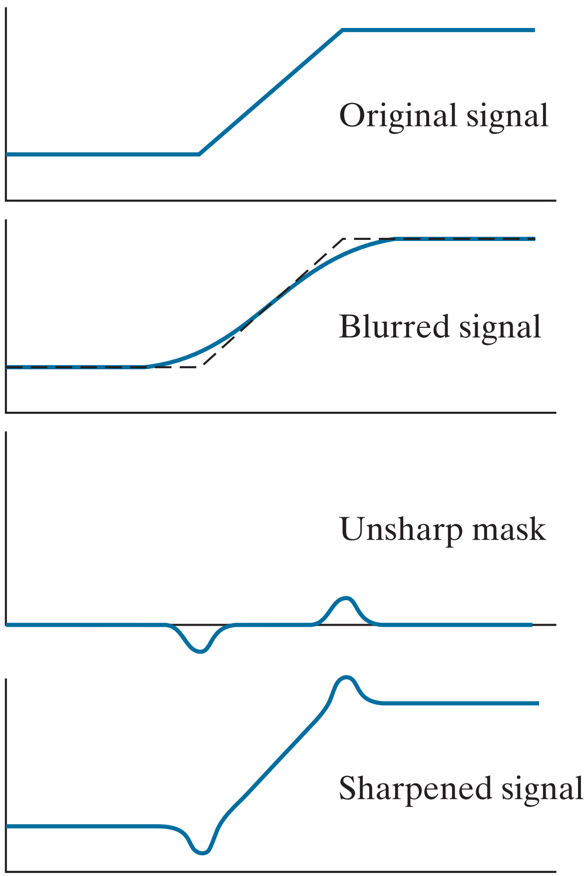
$$g(x, y) = f(x, y) + k\, g_{\text{mask}}(x, y) \quad k \geq 0$$

K=1 is used in unsharp masking technique

- When $k > 1$, the process is referred to **as *highboost filtering*.**
- Choosing $k < 1$ reduces the contribution of the unsharp mask

Unsharp Masking

1-D illustration of the mechanics of unsharp masking.



(a) Original signal.(a horizontal intensity profile across a vertical ramp edge that transitions from dark to light

(b) Blurred signal with original shown dashed for reference.

(c) Unsharp mask.

(d) Sharpened signal, obtained by adding (c) to (a).

Unsharp Masking

- By comparing this result corresponding to the ramp we note that the unsharp mask is similar to what we would obtain using a second-order derivative.
- The final sharpened result is obtained by adding the mask to the original signal.
- The points at which a change of slope occurs in the signal are now emphasized (sharpened).
- Observe that negative values were added to the original.
  - Thus, it is possible for the final result to have negative intensities if the original image has any zero values, or if the value of *k* is chosen large enough to emphasize the peaks of the mask to a level larger than the minimum value in the original signal.
  - Negative values cause dark halos around edges that can become objectionable if *k* is too large.

Unsharp Masking using Lapacian

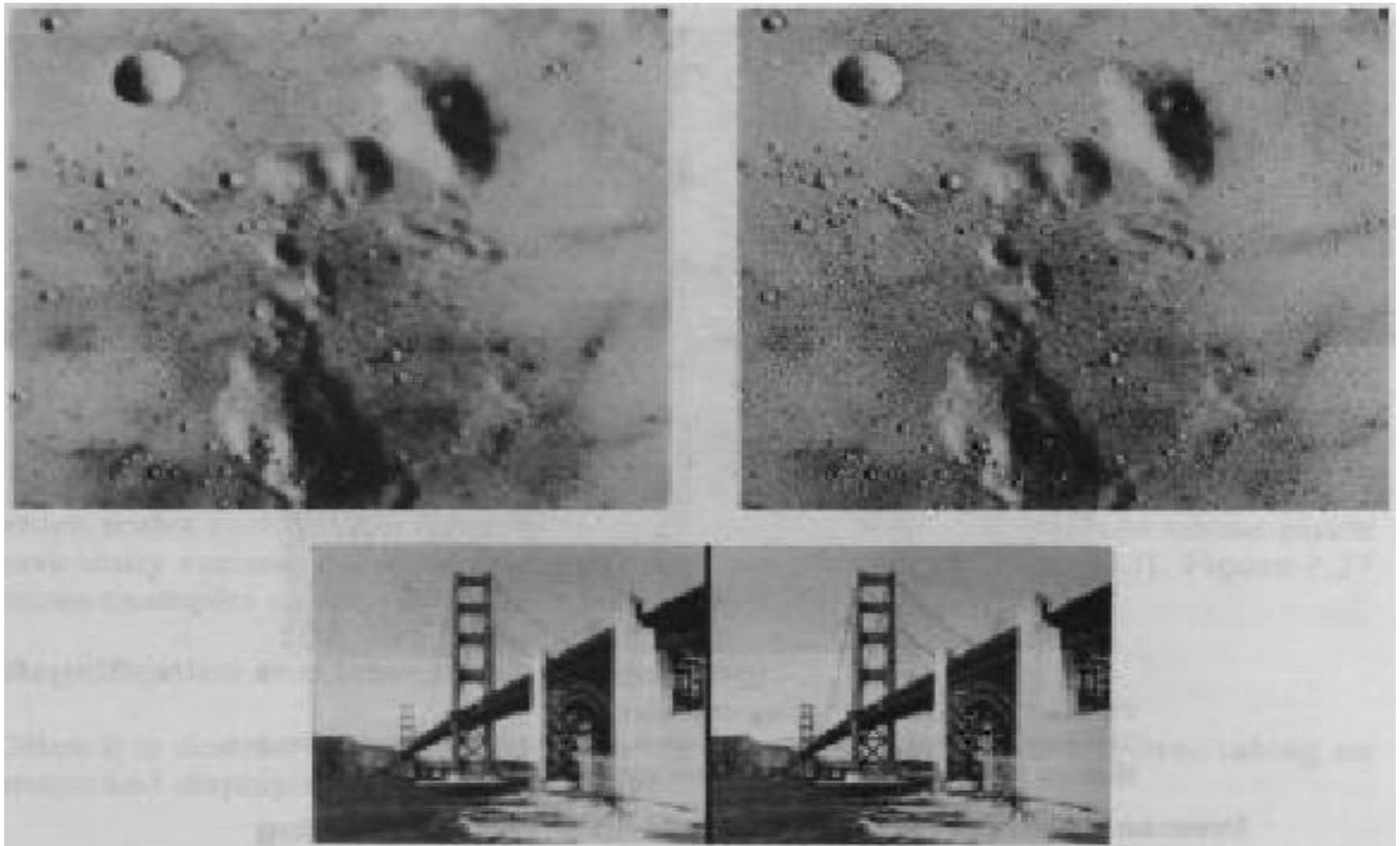


Figure 7.24 Unsharp masking. Original (left), enhanced (right).

Highboost Filtering

- Sharpened image generated by unsharp masking and Laplacian has average background intensity near black

- The result can be improved by increasing the scaling factor ‘k’ of the sharpening mask in

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

Highboost Filtering

- Consider the following slightly blurred image of white text on a dark gray background of size 600 × 259 pixels

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

- Now blur the image using a Gaussian smoothing filter of size 31 × 31 with σ = 5

Highboost Filtering

- The unsharp mask, obtained using. $g_{\text{mask}}(x, y) = f(x, y) - \bar{f}(x, y)$ is

- Now using unsharp masking expression. $g(x, y) = f(x, y) + k\, g_{\text{mask}}(x, y)$ with k=1, the sharpened image obtained is

Original

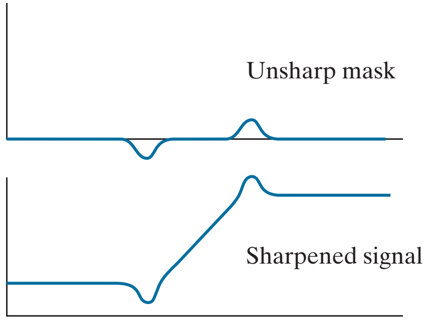
Highboost Filtering

- Now applying highboost filtering techniques with k=4.5 in $g(x,y) = f(x,y) + k\,g_{\text{mask}}(x,y)$ results in the sharpened image

Highboost

Unsharp masking

Original



- K = 4.5 is almost at the extreme of what we can use without introducing some serious artifacts in the image.
- The artifacts are dark, almost black, halos around the border of the characters.
- This is caused by the lower “blip” in figure becoming negative
- When scaling the image so that it only has positive values for display, the negative values are either clipped at 0, or scaled so that the most negative values become 0, depending on the scaling method used.
- In either case, the blips will be the darkest values in the image

Highboost Filtering

a b c
d e

FIGURE 3.49 (a) Original image of size 600×259 pixels. (b) Image blurred using a 31×31 Gaussian lowpass filter with $\sigma = 5$. (c) Mask. (d) Result of unsharp masking using Eq. (3-56) with $k = 1$. (e) Result of highboost filtering with $k = 4.5$.

Using First Order Derivative for Sharpening

- An edge is the boundary between two regions with distinct gray level properties. Hence derivative is used for most edge detection techniques
- First derivatives in image processing are implemented using the magnitude of the gradient.
- The *gradient* of an image *f* at coordinates (*x*, *y*) is defined as the two- dimensional column vector

$$\nabla f \equiv \text{grad}(f) = \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \dfrac{\partial f}{\partial x} \\ \dfrac{\partial f}{\partial y} \end{bmatrix}$$

- This vector has the important geometrical property that it points in the direction of the greatest rate of change of *f* at location (*x*, *y*).

First Order Derivative for Sharpening

- The *magnitude* (*length*) of vector ∇f, denoted as *M*(*x*, *y*) (the vector norm notation $\|\nabla f\|$ is also used frequently), where

$$M(x, y) = \|\nabla f\| = \text{mag}(\nabla f) = \sqrt{g_x^2 + g_y^2}$$

is the *value* at (*x*, *y*) of the *rate of change* in the direction of the gradient vector.

- Note that *M*(*x*, *y*) is an image of the same size as the original, created when *x* and *y* are allowed to vary over all pixel locations in *f*.
- It is common practice to refer to this image as the *gradient image* (or simply as the *gradient*).
- In some implementations, it is more suitable computationally to approximate the squares and square root operations by absolute values:

$$M(x, y) \approx |g_x| + |g_y|$$

First Order Derivative for Sharpening

- Defining discrete approximations to these equations results in first order kernels
- Consider the following notation for intensities of pixels in 3 x 3 region

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

- The value of the center point, z_5, denotes the value of $f(x, y)$ at an arbitrary location, (x, y).
- z_1 denotes the value of $f(x - 1, y - 1)$; and so on
- The simplest approximations to a first-order derivative are

  $g_x = (z_8 - z_5)$ and $g_y = (z_6 - z_5)$

-

First Derivative for Sharpening

Other first order derivative kernels

- Roberts Operator
- Prewitt Operator
- Sobel Operator

First Order Derivative: Roberts Operator

- Roberts operator [proposed in 1965] in the early development of digital image processing, uses cross differences:

$$g_x = (z_9 - z_5) \quad \text{and} \quad g_y = (z_8 - z_6)$$

- The gradient image is

$$M(x,y) = \left[(z_9 - z_5)^2 + (z_8 - z_6)^2\right]^{1/2}$$

Or

$$M(x,y) \approx |z_9 - z_5| + |z_8 - z_6|$$

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

-1	0
0	1

0	-1
1	0

First Order Derivative: Roberts Operator

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

- Second variant of Roberts operator

$$g_x = (z_5 - z_8) \text{ and } g_y = (z_5 - z_6)$$

1	0
-1	0

1	−1
0	0

The gradient image = $|z_5 - z_8| + |z_5 - z_6|$

- Third variant of Roberts operator

$$g_x = (z_5 - z_9) \text{ and } g_y = (z_6 - z_8)$$

1	0
0	-1

0	1
-1	0

First Order Derivative: Prewitt Operator

- Roberts operators use kernels of size 2 x 2.
- We prefer to use kernels of odd sizes because they have a unique center of spatial symmetry.
- The smallest kernels in which we are interested are of size 3 × 3.
- Another approximation to gradient using 3 x 3 mask is Prewitts operator given by

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

$g(x) = (z_7 + z_8 + z_9) - (z_1 + z_2 + z_3)$

$g(y) = (z_3 + z_6 + z_9) - (z_1 + z_4 + z_7)$

−1	−1	−1
0	0	0
1	1	1

−1	0	1
−1	0	1
−1	0	1

First Order Derivative: Sobel Operator

- 3 x 3 mask in Sobel operator is given by

$$g_x = \frac{\partial f}{\partial x} = (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)$$

$$g_y = \frac{\partial f}{\partial y} = (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)$$

$$M(x,y) = \left[g_x^2 + g_y^2\right]^{\frac{1}{2}} = \left[\left[(z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)\right]^2 + \left[(z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)\right]^2\right]^{\frac{1}{2}}$$

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

−1	−2	−1
0	0	0
1	2	1

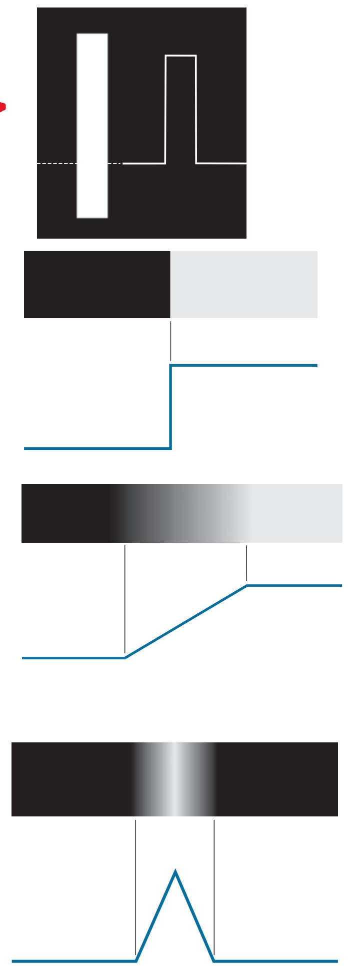
−1	0	1
−2	0	2
−1	0	1

- **The idea behind using a weight value of 2 in the center coefficient is to give more importance to the center point**

Edge Detection using Spatial Filters

Different edge profiles:

- Line Edge
- Step edge
- Ramp Edge
- Roof Edge



Edge detection using Spatial Filters (Sobel Operators)

- Apply Sobel operator and thresholding to extract edges

F(x,y)=

0	**0**	**0**	**0**	**0**	**0**	**2**	**0**	**3**	**3**
0	0	0	1	0	0	0	2	4	2
0	0	2	0	2	4	3	3	2	3
0	0	1	3	3	4	3	3	3	3
0	1	9	4	3	3	2	4	3	2
0	0	1	2	3	3	4	4	4	3

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

−1	−2	−1
0	0	0
1	2	1

−1	0	1
−2	0	2
−1	0	1

$$g_x = \frac{\partial f}{\partial x} = (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)$$

$$g_y = \frac{\partial f}{\partial y} = (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)$$

$$M(x,y) \approx |g_x| + |g_y|$$

DIGITAL IMAGE PROCESSING-1

Edge detection using Spatial Filters (Sobel Operators)

Solution:

$g(0,0) = \overset{g_x}{2} + \overset{g_y}{2} = 4$

F(x,y)=

0	**0**	**0**	**0**	**0**	**0**	**2**	**0**	**3**	**3**
0	$\overset{4}{\cancel{0}}$	$\overset{6}{\cancel{0}}$	$\overset{4}{\cancel{1}}$	$\overset{10}{\cancel{0}}$	$\overset{4}{\cancel{0}}$	$\overset{12}{\cancel{0}}$	2	4	2
0	0	2	0	2	4	3	3	2	3
0	0	1	3	3	4	3	3	3	3
0	1	9	4	3	3	2	4	3	2
0	0	1	2	3	3	4	4	4	3

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

-1	-2	-1
0	0	0
1	2	1

-1	0	1
-2	0	2
-1	0	1

Edge detection using Spatial Filters (Sobel Operators)

Solution:

F(x,y)=

0	**0**	**0**	**0**	**0**	**0**	**2**	**0**	**3**	**3**
0	0	0	1	0	0	0	2	4	2
0	0	2	0	2	4	3	3	2	3
0	0	1	3	3	4	3	3	3	3
0	1	9	4	3	3	2	4	3	2
0	0	1	2	3	3	4	4	4	3

4	6	4	10	14	12	14	4
6	8	10	20	16	12	6	0
4	10	14	10	2	4	2	4
2	12	12	2	2	4	2	4

	4	6	4	10	14	12	14	4
	6	8	10	20	16	12	6	0
F(x,y)=	4	10	14	10	2	4	2	4
	2	12	12	2	2	4	2	4

	0	0	0	0	1	1	1	0
	0	0	0	1	1	1	0	0
F(x,y)=	0	0	1	0	0	0	0	0
	0	1	1	0	0	0	0	0

- If threshold = 12 then

Next Session

- Enhancement using Frequency domain techniques

THANK YOU

Dr. Shikha Tripathi
Department of Electronics &
Communication Engineering

shikha@pes.edu
+91 9482219115