CSE6706:

Advanced Digital Image Processing

Dr. Md. Monirul Islam

CSE-BUET

Topics

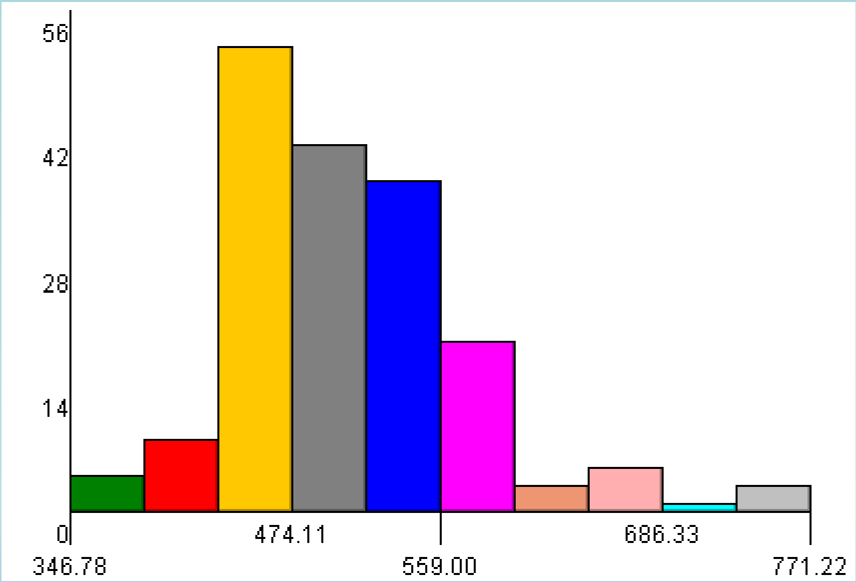
- Image Processing Basics
- Data Structure
- Image Enhancement

CSE-BUET

Image Enhancement

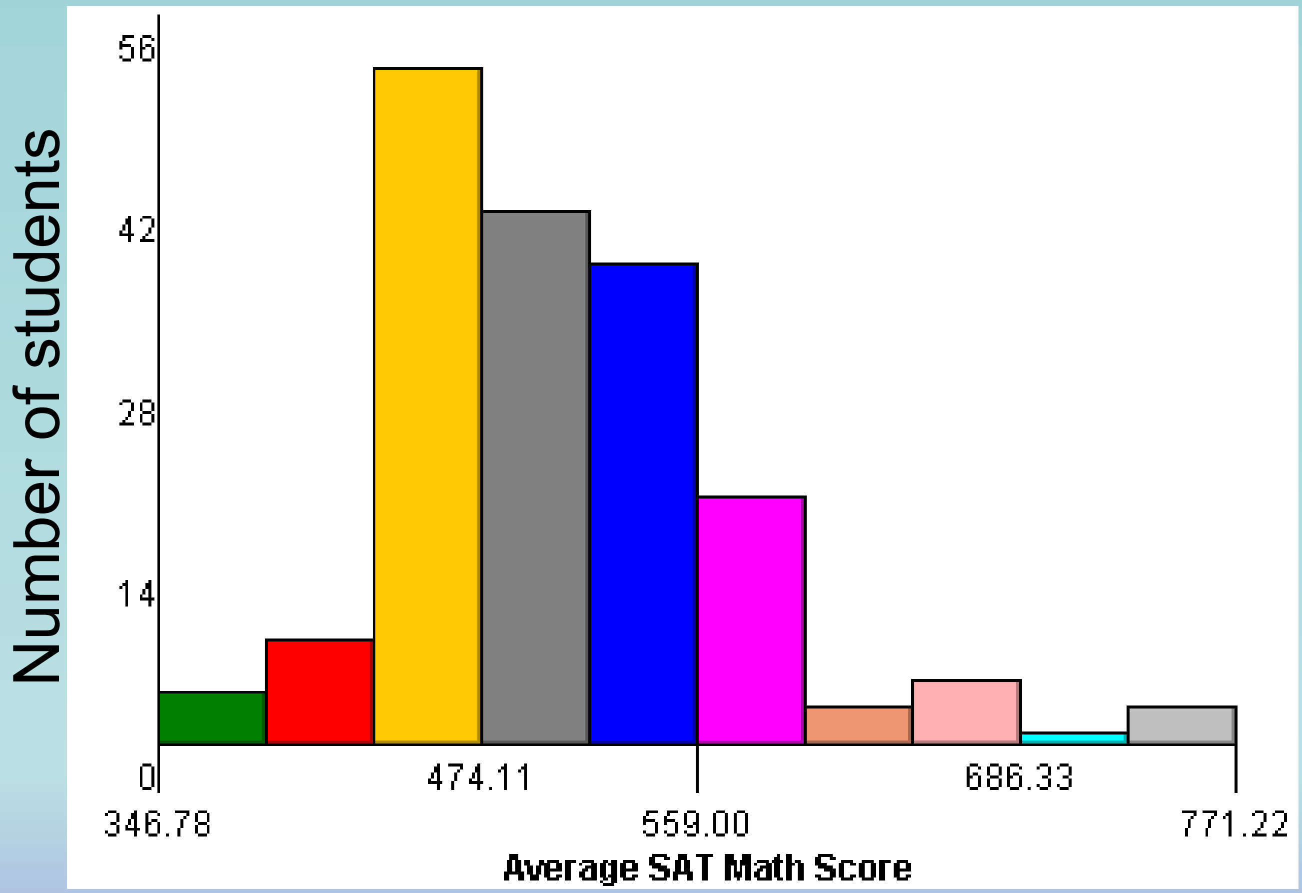
CSE-BUET

Image Enhancement using Histogram Processing



CSE-BUET

Histogram



CSE-BUET

Image Histogram

Assuming image has *L* gray levels,

a histogram is an array, *h:*

bin(0)	*bin*(1)	. . .	*bin*(*L*-2)	*bin*(*L*-1)
		…		
h(0)	*h*(1)	. . .	*h*(*L*-2)	*h*(*L*-1)

bin(*i*) or *h*(*i*) contains the number of pixels with gray level *i*

$$h(r_k) = n_k$$

CSE-BUET

Image Histogram

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

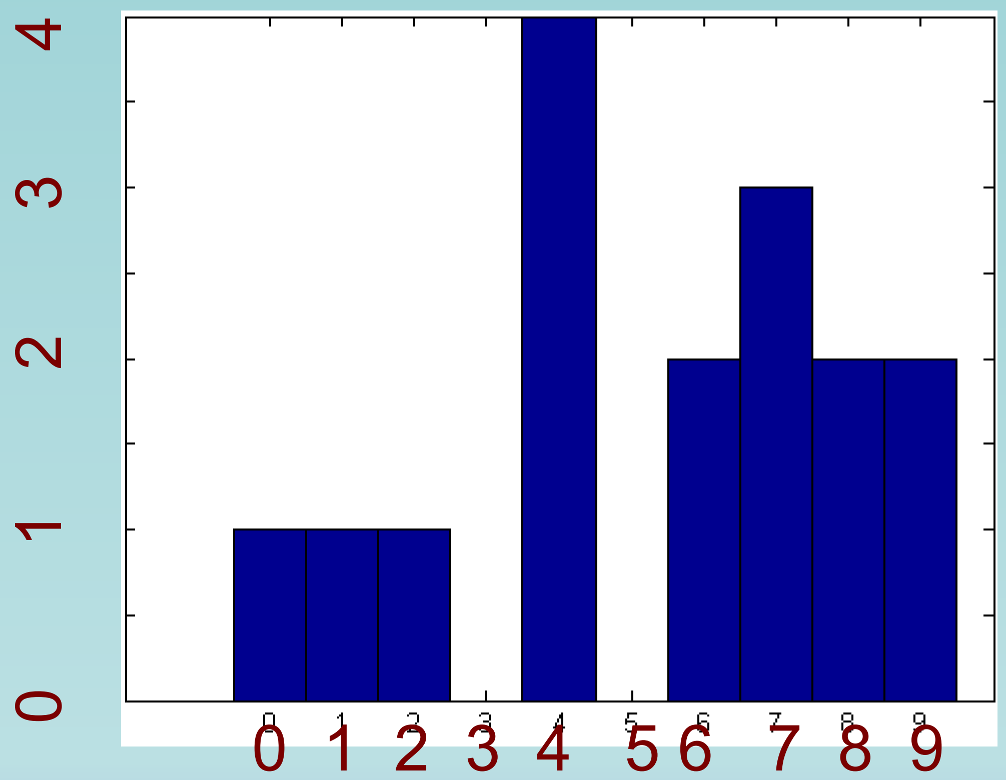
An image

Histogram:

Bin pos.:	0	1	2	3	4	5	6	7	8	9
h:	1	1	1	0	4	0	2	3	2	2

CSE-BUET

Image Histogram



CSE-BUET

Normalized Image Histogram

a histogram is an array, *h:*

bin(0)	*bin*(1)	. . .	*bin*(*L*-2)	*bin*(*L*-1)
		…		
h(0)	*h*(1)	. . .	*h*(*L*-2)	*h*(*L*-1)

Bins are divided by total count

$$p(r_k) = n_k / n$$

where, $n = \sum_k n_k$

CSE-BUET

Normalized Image Histogram

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

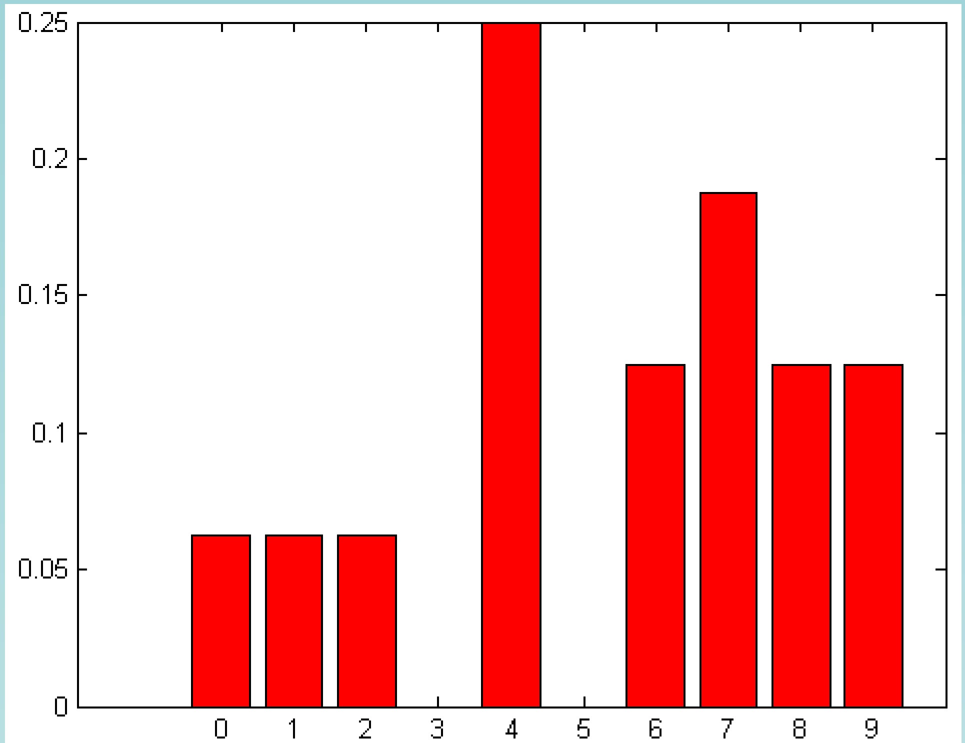
An image

Histogram:

Bin pos:	0	1	2	3	4	5	6	7	8	9
h:	1/16	1/16	1/16	0	1/4	0	1/8	3/16	1/8	1/8

CSE-BUET

Normalized Image Histogram



0 1 2 3 4 5 6 7 8 9

CSE-BUET

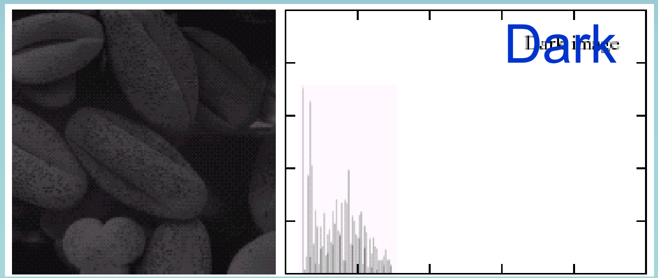
Normalized Image Histogram

$$p(r_k) = n_k / n$$

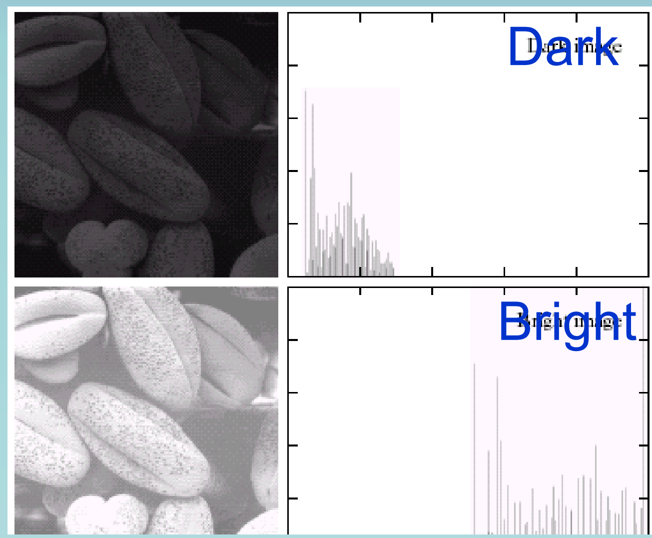
Normalized histogram is more like probabilities

$p(r_k)$: how likely a pixel will have gray level r_k

Histograms of Some Real Images

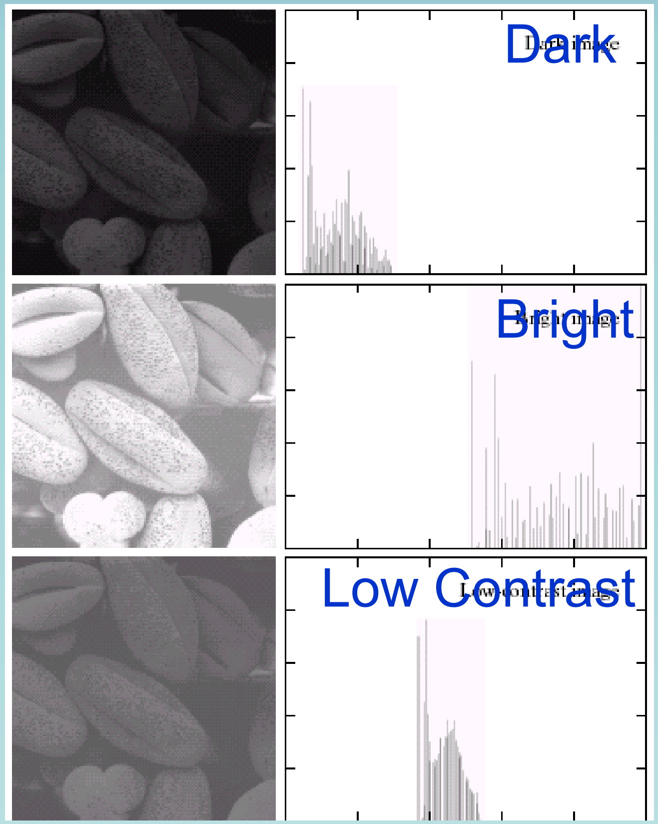


Histograms of Some Real Images

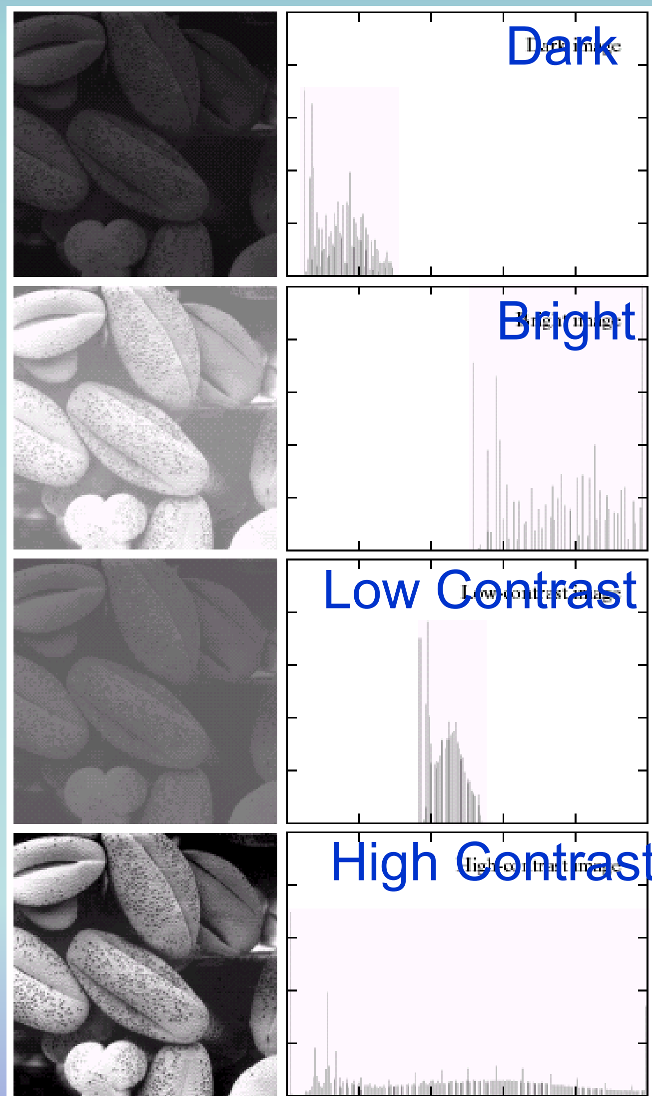


CSE-BUET

Histograms of Some Real Images

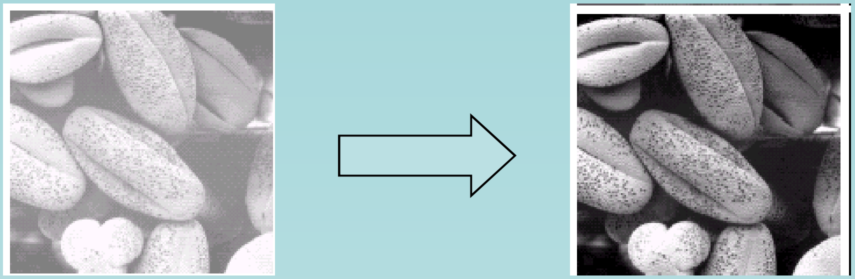


Histograms of Some Real Images



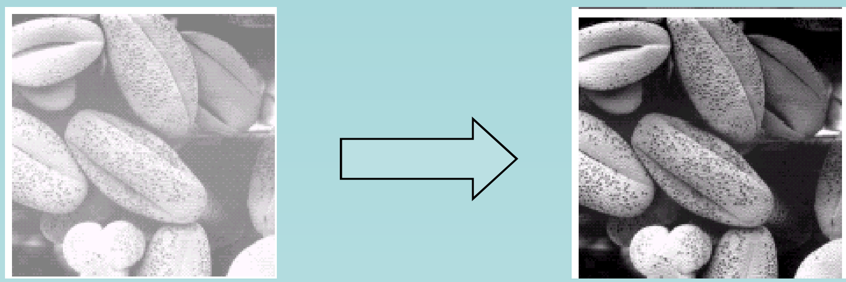
- Dark images: bins in the low end
- Bright images: bins in the high end
- Low contrast images: bins in a narrow range
- High contrast images: bins are in the *entire range* and *uniform*

Histogram Equalization: Automatic Enhancement of Images



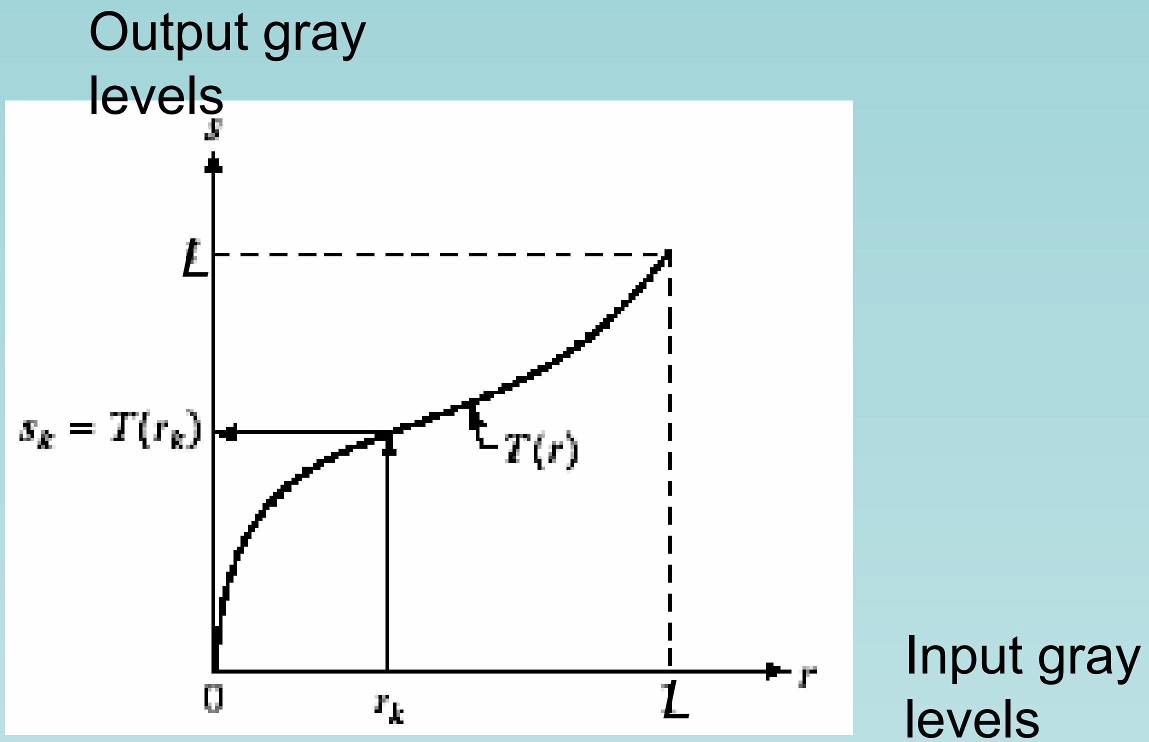
CSE-BUET

Histogram Equalization: Automatic Enhancement of Images



CSE-BUET

Histogram Equalization: Automatic Enhancement of Images



CSE-BUET

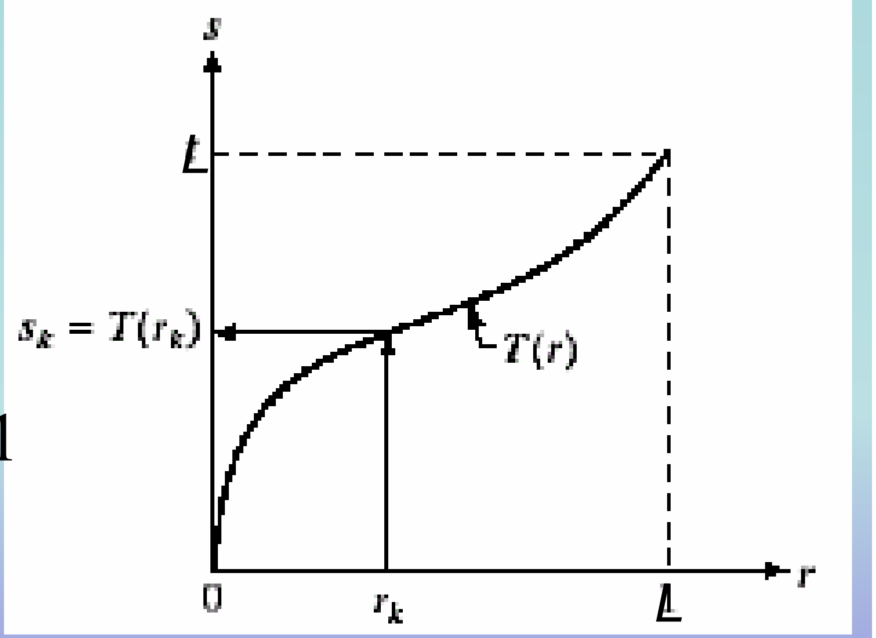
Histogram Equalization: Automatic Enhancement of Images

- Let

  s, r: are all discrete and in [0, *L*-1]

$$s = T(r) \qquad 0 \leq r \leq \mathrm{L} \text{-} 1$$

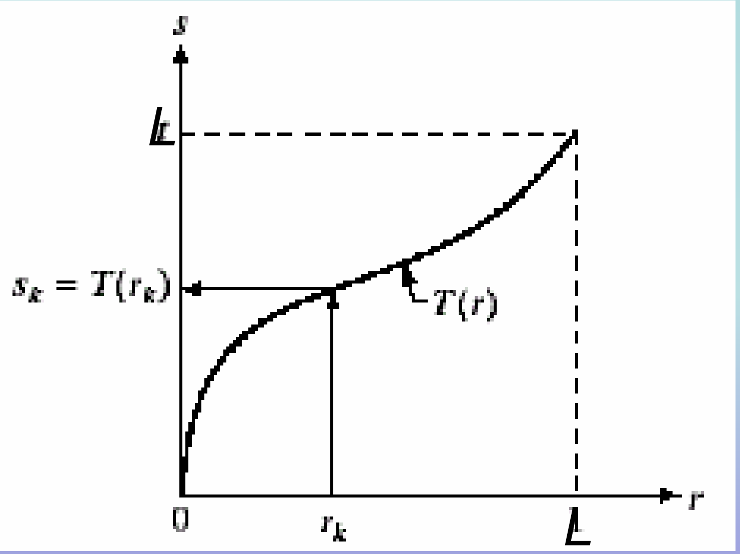
- Let, 2 conditions hold:
- *T(r): single valued and monotonous in* $0 \leq r \leq \mathrm{L} \text{-} 1$
- $0 \leq T(r) \leq L-1 \quad \text{for } 0 \leq r \leq L-1$



CSE-BUET

Histogram Equalization: Automatic Enhancement of Images

- Let,
  - $p_r(r)$: the normalized histogram of the given image
  - $p_s(s)$: the normalized histogram of the output image

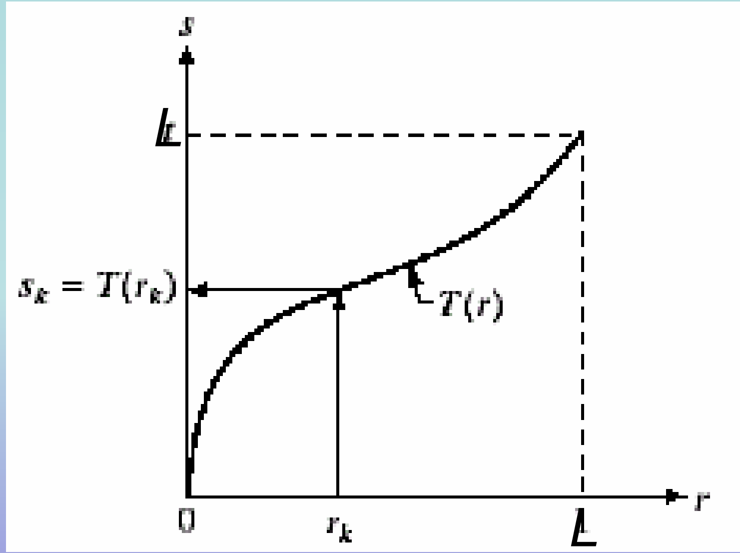


CSE-BUET

Histogram Equalization: Automatic Enhancement of Images

- Let,
  - $p_r(r)$: the normalized histogram of the given image
  - $p_s(s)$: the normalized histogram of the output image

- We have to find the transformed image so that $p_s(s)$ of the transformed image is uniform

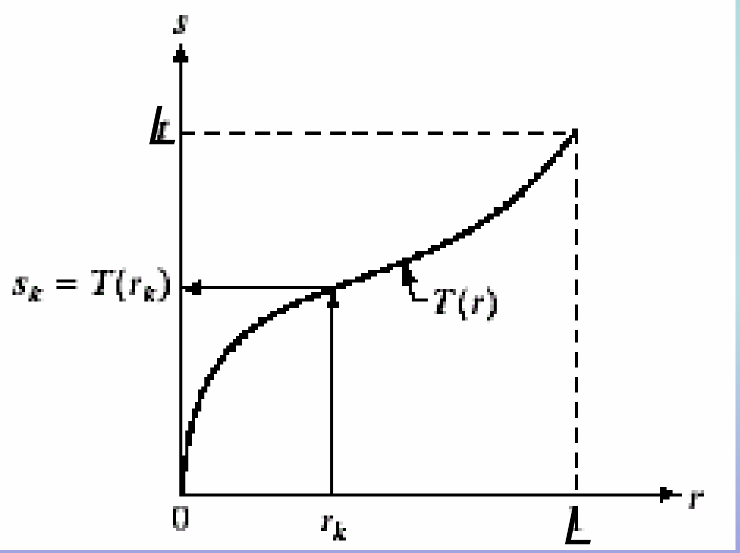


CSE-BUET

Histogram Equalization: Automatic Enhancement of Images

- If $p_r(r)$ and $T(r)$ are known and $T^{-1}(s)$ increases monotonically,

$$p_s(s) = p_r(r)\left|\frac{dr}{ds}\right|$$



CSE-BUET

Histogram Equalization: Automatic Enhancement of Images

- Let us try with this transformation function

$$s = T(r) = (L-1)\int_0^r p_r(w)dw$$

CSE-BUET

Histogram Equalization: Automatic Enhancement of Images

- Let us try with this transformation function

$$s = T(r) = (L-1)\int_0^r p_r(w)\,dw$$

Looks like a CDF

CSE-BUET

Histogram Equalization: Automatic Enhancement of Images

- Let us try with this transformation function

$$s = T(r) = (L-1)\int_{0}^{r} p_r(w)\,dw$$

- It also holds conditions:
  - *T(r): single valued and monotonous in* $0 \le r \le L-1$

$-\,0 \le T(r) \le L-1 \quad \text{for } 0 \le r \le L-1$

Histogram Equalization: Automatic Enhancement of Images

- Let us try with this transformation function

$$s = T(r) = (L-1)\int_0^r p_r(w)dw$$

$$\frac{ds}{dr} = \frac{dT(r)}{dr} = (L-1)\frac{d}{dr}\left[\int_0^r p_r(w)dw\right]$$

$$= (L-1)p_r(r)$$

Histogram Equalization: Automatic Enhancement of Images

$$\frac{ds}{dr} = (L-1)\,p_r(r)$$

$$p_s(s) = p_r(r)\left|\frac{dr}{ds}\right| = p_r(r) \times \frac{1}{(L-1)\,p_r(r)} = \frac{1}{L-1}$$

Histogram Equalization: Automatic Enhancement of Images

$$p_s(s) = \frac{1}{L-1}$$

This means transformation function

$$s = T(r) = (L-1)\int_0^r p_r(w)\,dw$$

can lead to an equalized image

Histogram Equalization: Automatic Enhancement of Images

- Next Objective: find the digitized version of

$$s = T(r) = (L-1)\int_0^r p_r(w)\,dw$$

Histogram Equalization: Automatic Enhancement of Images

We know: $p_r(r_k) = \dfrac{n_k}{n}$

Histogram Equalization: Automatic Enhancement of Images

We know: $p_r(r_k) = \dfrac{n_k}{n}$

Then, the digitized version of

$$s = T(r) = (L-1)\int_0^r p_r(w)dw$$

is

$$s_k = T(r_k) = (L-1)\sum_{j=0}^{j=k} p_r(r_j)$$

CSE-BUET

Histogram Equalization: Automatic Enhancement of Images

We know: $p_r(r_k) = \dfrac{n_k}{n}$

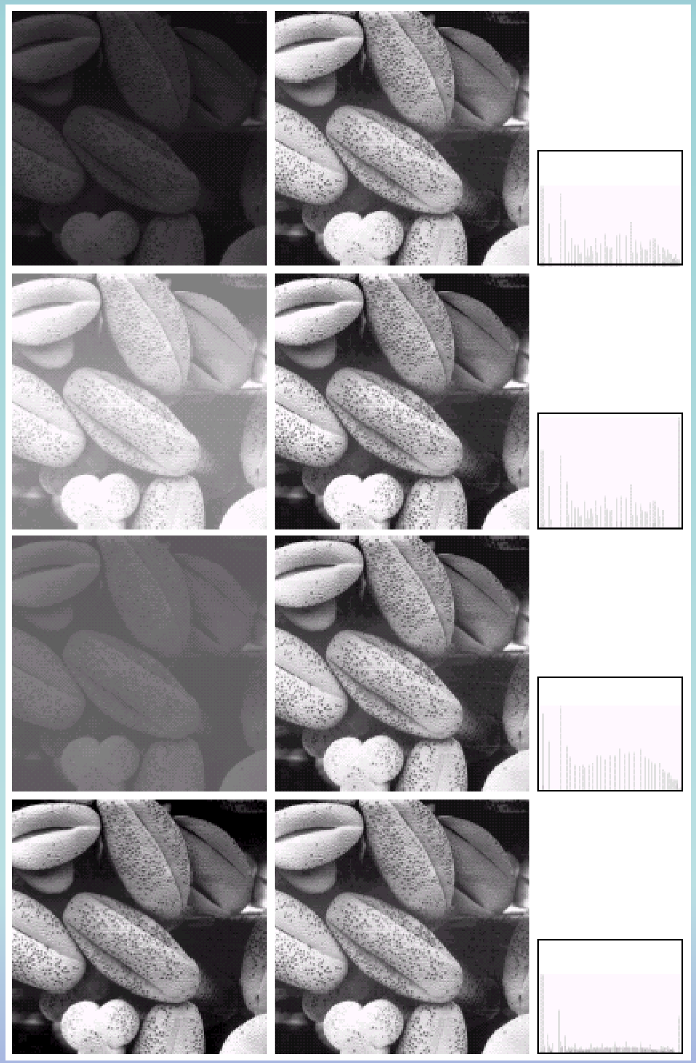
Then, the digitized version of

$$s = T(r) = (L-1)\int_0^r p_r(w)dw$$

is

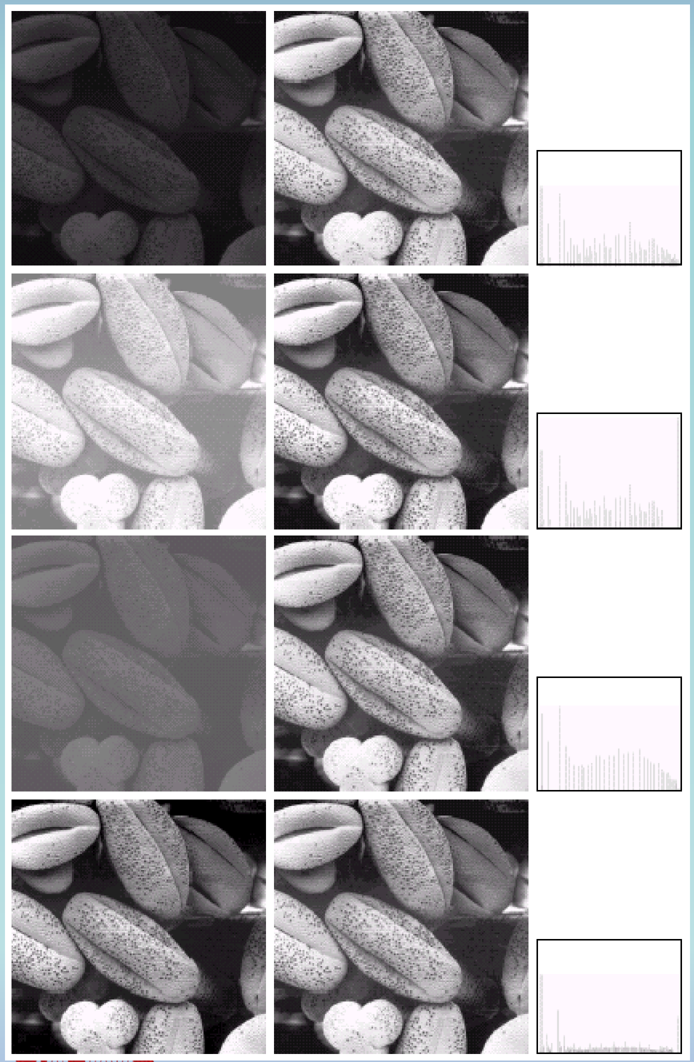
$$s_k = T(r_k) = (L-1)\sum_{j=0}^{j=k} p_r(r_j) = (L-1)\sum_{j=0}^{j=k} \frac{n_j}{n}$$

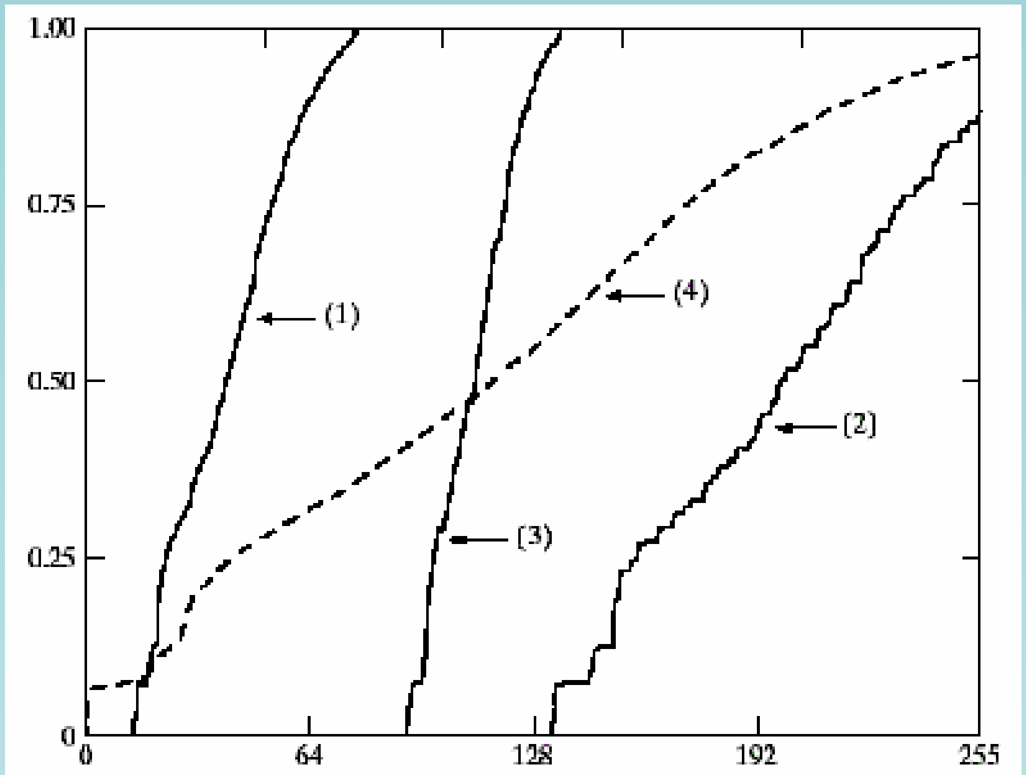
Histogram Equalization



Original Enhanced

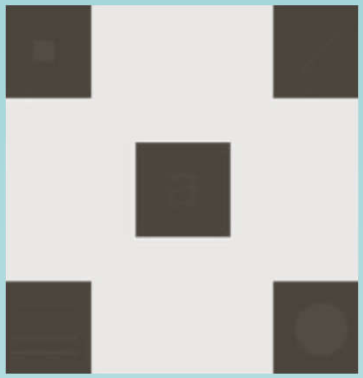
Histogram Equalization



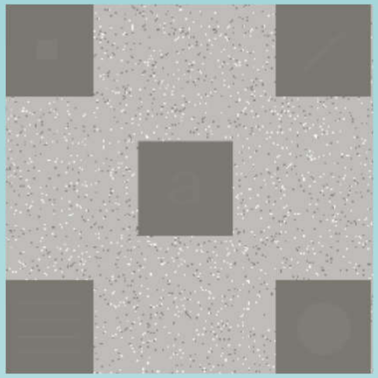


Transfer Functions

Local Histogram Processing



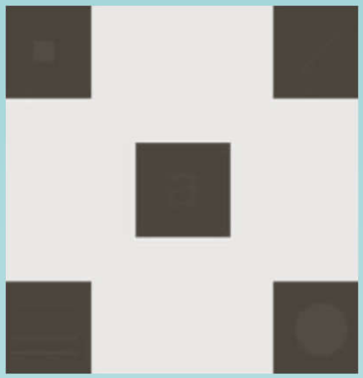
original



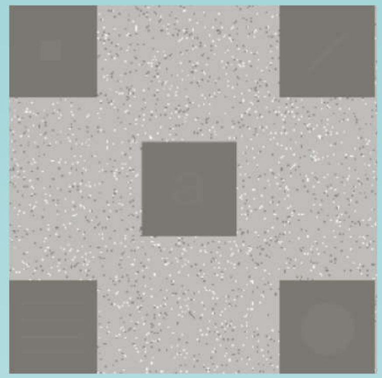
Equalized
with global
histogram

CSE-BUET

Local Histogram Processing



original



Equalized with global histogram

Equalized with 3X3 local histogram

CSE-BUET

Arithmetic/Logic operations for Image Enhancement

- Pixel by pixel operation
- Output pixel at (x, y) is calculated from input pixels of the same location, e.g., (x, y)

$$g(x, y) = f(x, y) + h(x, y)$$
$$g(x, y) = f(x, y) - h(x, y)$$
$$g(x, y) = f(x, y) \text{ and } h(x, y)$$
$$g(x, y) = f(x, y) \text{ or } h(x, y)$$

CSE-BUET

Logic operations for Image Enhancement

$$g(x, y) = f(x, y) \,\mathrm{and}\, h(x, y)$$
$$g(x, y) = f(x, y) \,\mathrm{or}\, h(x, y)$$

- Used for
  - masking
  - selecting a sub-image or *region of interest*

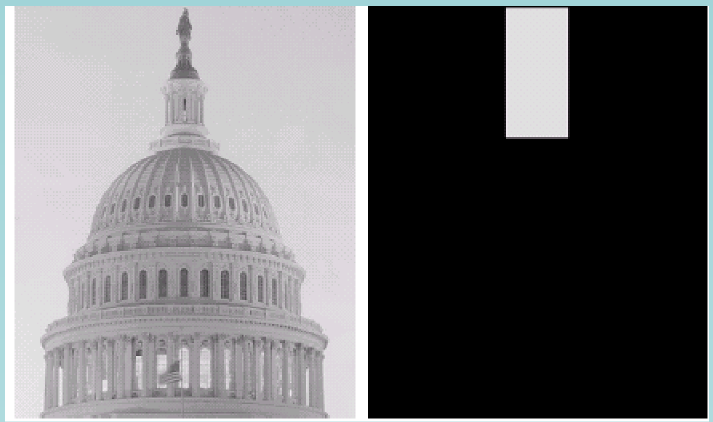
Logic operations for Image Enhancement

$$g(x, y) = f(x, y) \text{ and } h(x, y)$$
$$g(x, y) = f(x, y) \text{ or } h(x, y)$$

- Used for
  - masking
  - selecting a sub-image or *region of interest*

CSE-BUET

Logic operations for Image Enhancement

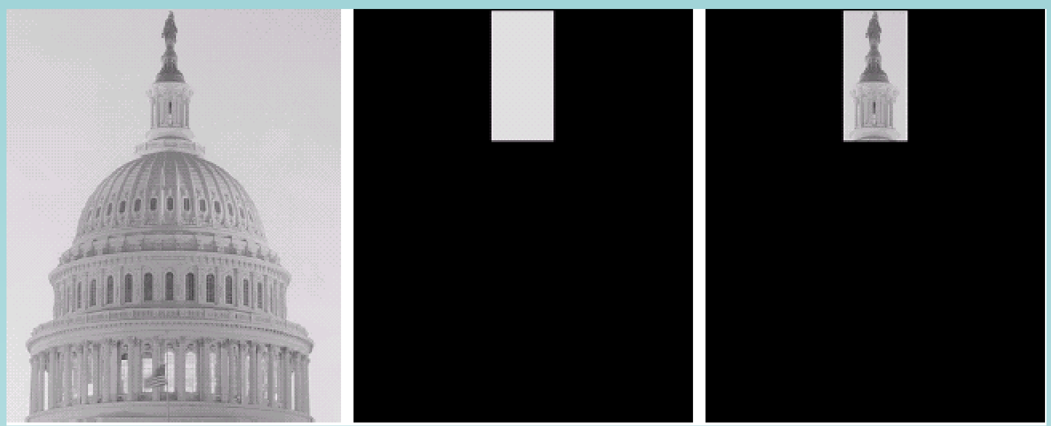


mask

- AND masking

CSE-BUET

Logic operations for Image Enhancement

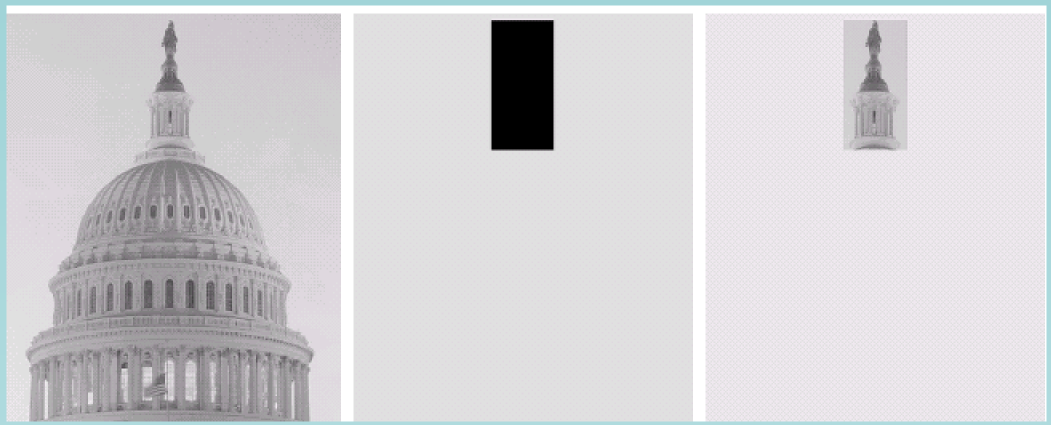


mask

- AND masking

CSE-BUET

Logic operations for Image Enhancement



mask

- OR masking

CSE-BUET

Image Enhancement using Subtraction

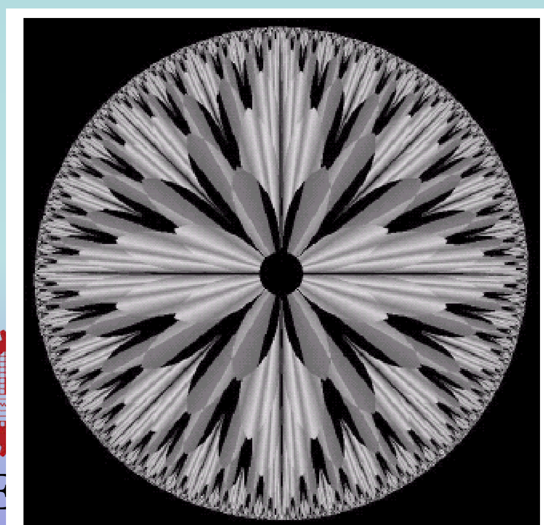
$$g(x, y) = f(x, y) - h(x, y)$$

- Used to enhance the differences in detail
- Have many commercial applications

Image Enhancement using Subtraction

- Used to enhance the differences in detail

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



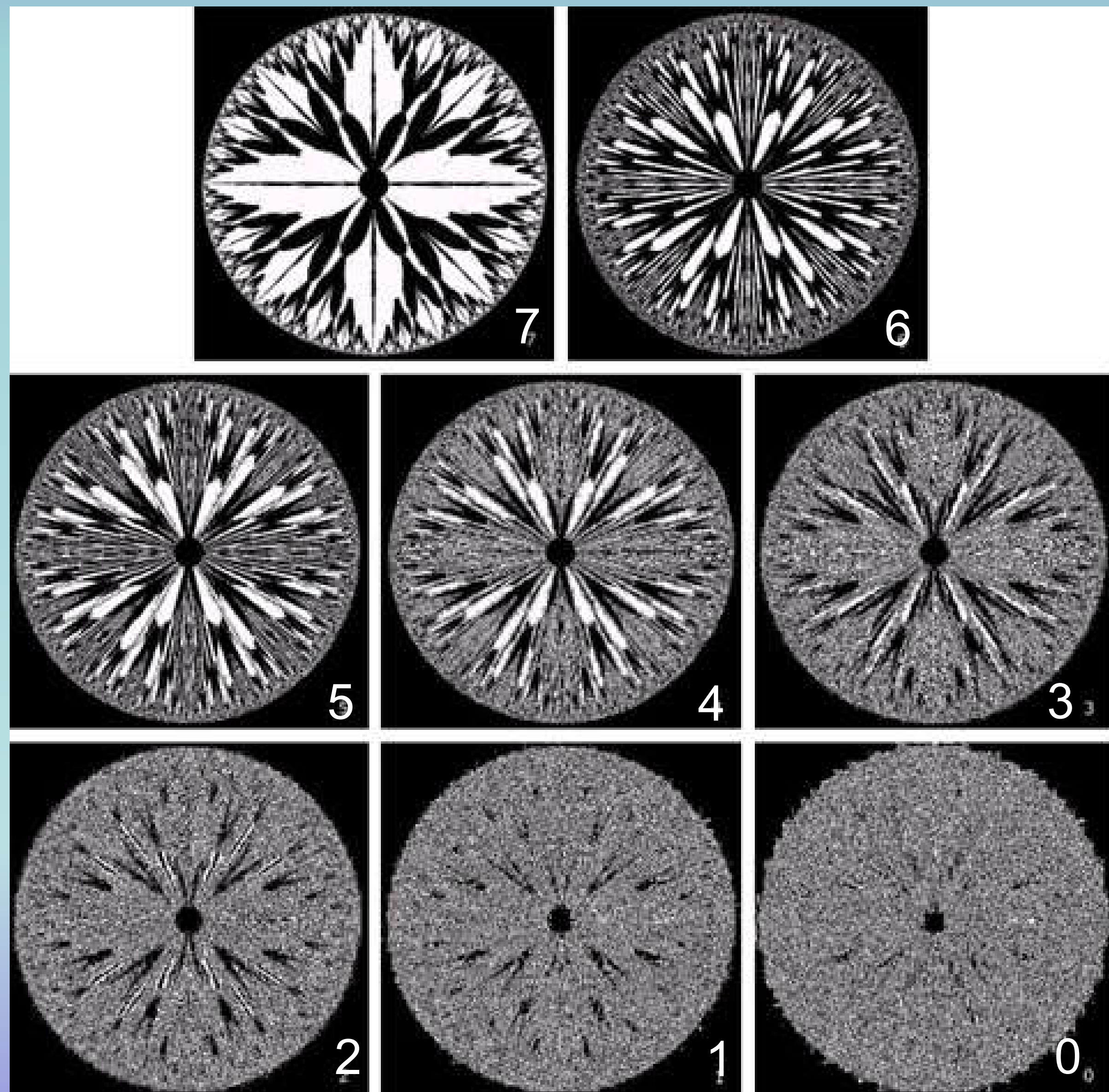
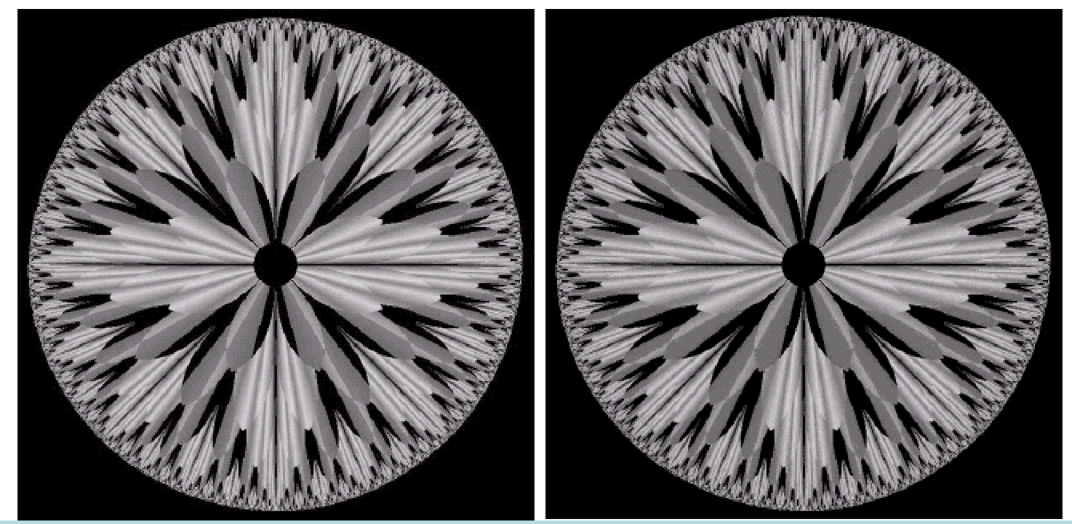


Image Enhancement using Subtraction

- Used to enhance the differences in detail

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



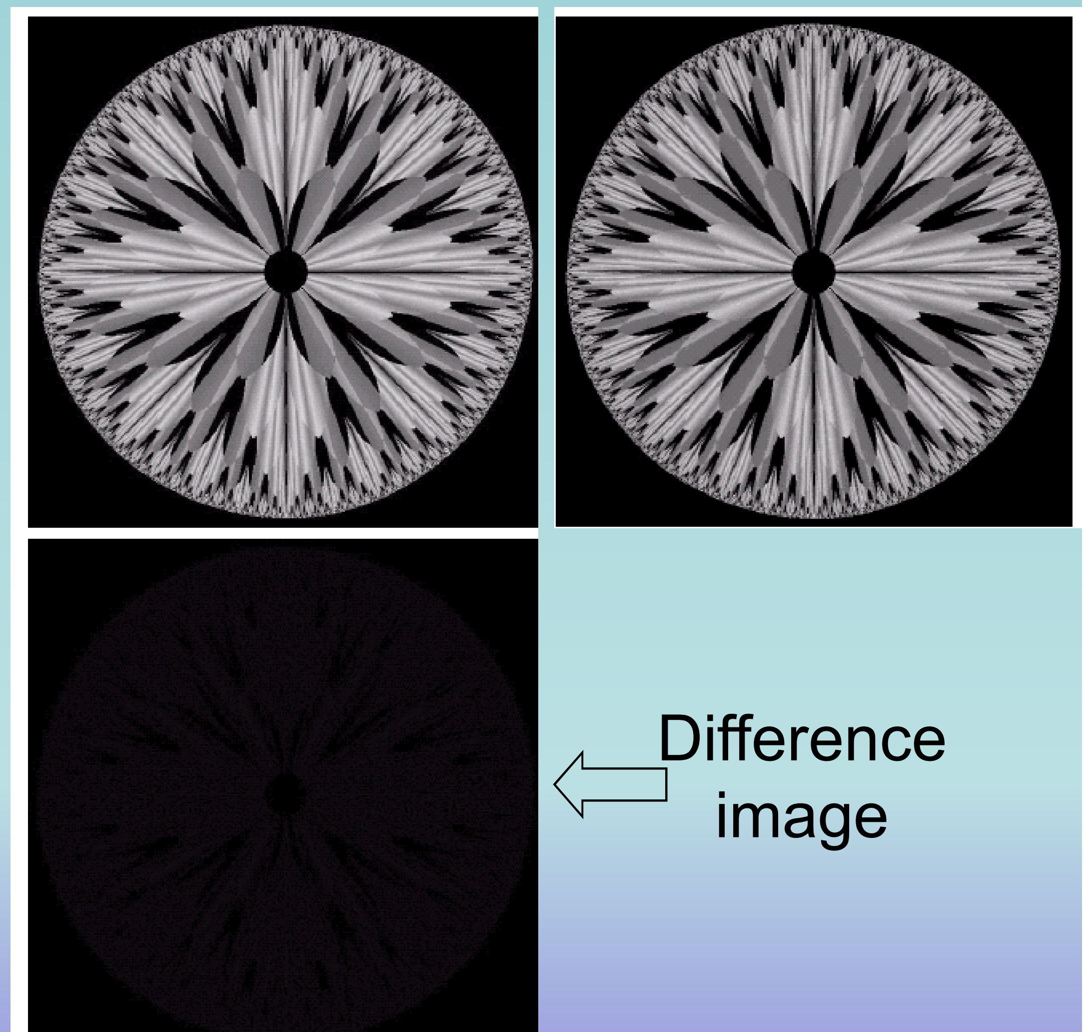
Least 4 bits set to 0

CSE-BUET

Image Enhancement using Subtraction

- Used to enhance the differences in detail

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

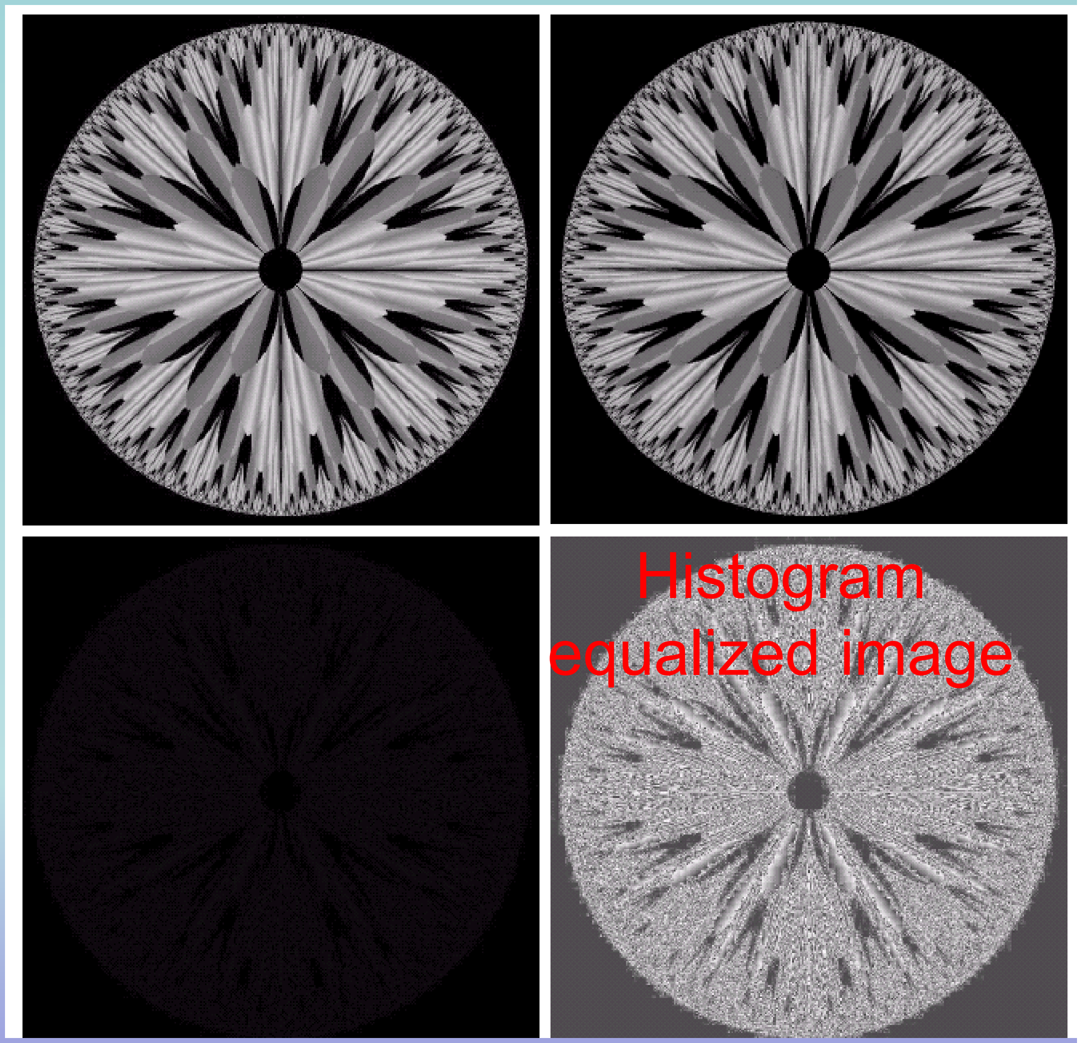


Difference image

Image Enhancement using Subtraction

- Used to enhance the differences in detail

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



CSE-BUET

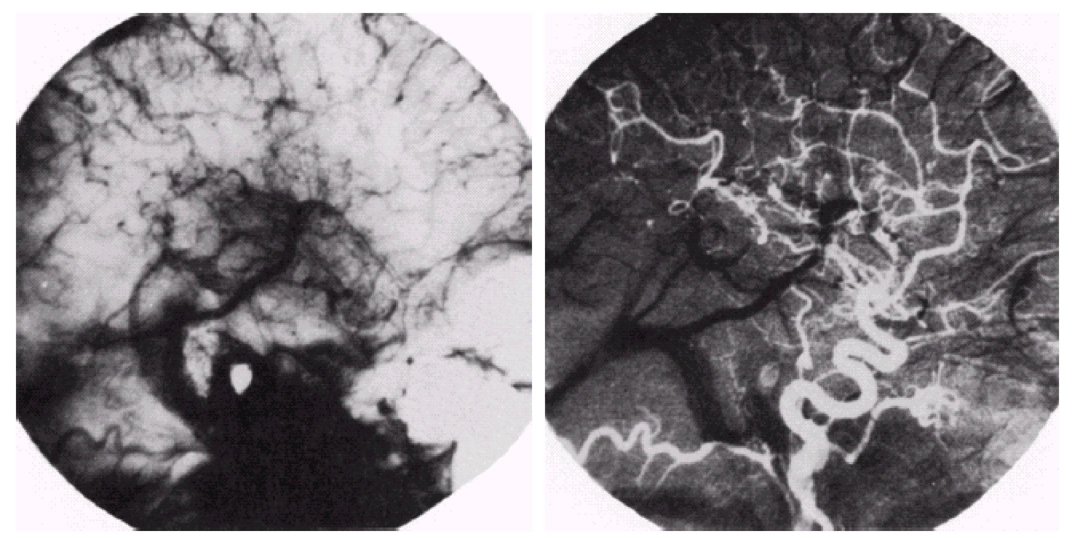
Image Enhancement using Subtraction

- commercial applications
  - *Mask mode radiography*
  - Used to closely inspect an area under investigation

Image Enhancement using Subtraction

- *Mask mode radiography*

- contrast medium goes through blood vessels
- The mask is subtracted from each subsequent photograph



A single diff. image but together with other shots it appears as a video

Issues in Image Enhancement using Subtraction

- Subtraction can result a pixel-value, p, in [-255 255]

$$0 - 255 = -255$$

$$255 - 0 = 255$$

Issues in Image Enhancement using Subtraction

- Subtraction can result a pixel-value, p, in [-255 255]
- Two ways to handle
  - Replace p with $(p + 255)/2$
    - Fast implementation
    - But entire range is not utilized

Issues in Image Enhancement using Subtraction

- Subtraction can result a pixel-value, p, in [-255 255]
- Two ways to handle (*method 2*)
  - Find the minimum, m, of all pixel values of the diff image
  - Add $-m$ to all pixel values
  - Find the maximum, M, of the modified pixel values
  - The modified pixels in the range [0 M]
  - Multiply all values by 255/ M

CSE-BUET

Issues in Image Enhancement using Subtraction

82	83	-5	3
63	81	25	10
7	31	71	9
30	79	-10	50

Min= -10

92	93	5	13
73	91	35	20
17	41	81	19
40	89	0	60

-(-10) added

Max= 93

252	255	14	36
200	250	96	55
47	112	222	52
110	244	0	165

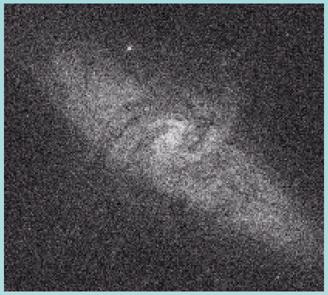
Multiplied by 255/93

All values in [0 255]

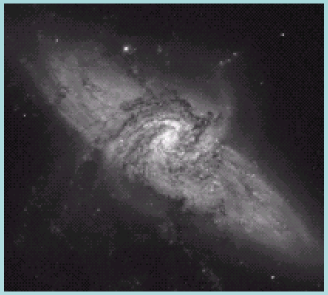
CSE-BUET

Image Enhancement using Addition or Averaging

- used for noise removal



Noisy image

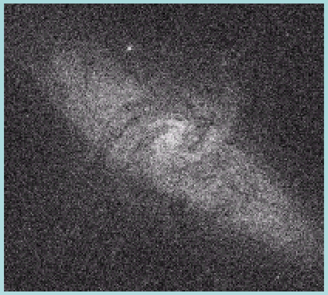


Original image

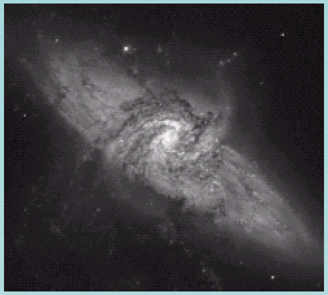
CSE-BUET

Image Enhancement using Addition or Averaging

- used for noise removal



$g(x,y)$



$f(x,y)$

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

noise

CSE-BUET

Image Enhancement using Addition or Averaging

- Let the noise is uncorrelated and has zero average
- We try to approximate $f(x, y)$ from a number of $g(x, y)$

$$\bar{g}(x, y) = \frac{1}{K}\sum_{i=1}^{K} g_i(x, y)$$

Image Enhancement using Addition or Averaging

$$\bar{g}(x, y) = \frac{1}{K}\sum_{i=1}^{K} g_i(x, y)$$

- As *K* increases, it can be proved that

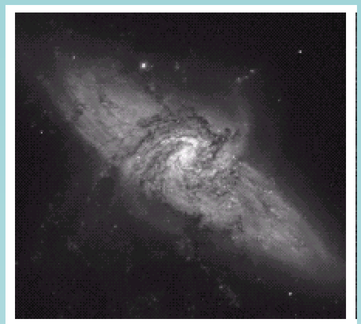
$$E\{\bar{g}(x, y)\} = f(x, y)$$

and

$$\sigma^2_{\bar{g}(x,y)} = \frac{1}{K}\sigma^2_{\eta(x,y)}$$

CSE-BUET

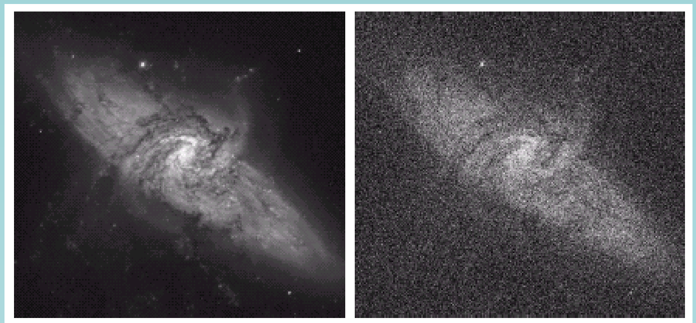
Image Enhancement using Averaging



This image was sent from space craft

CSE-BUET

Image Enhancement using Averaging

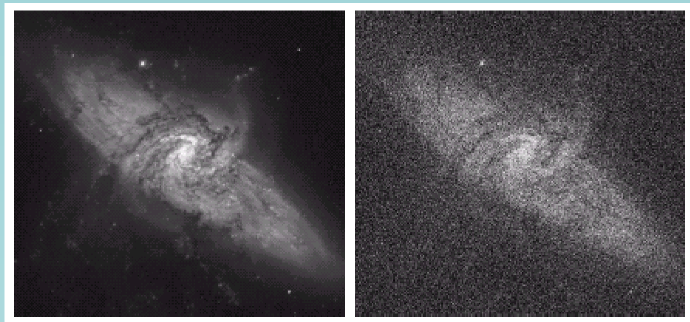


Original Image

Noisy image received

CSE-BUET

Image Enhancement using Averaging



Original Image

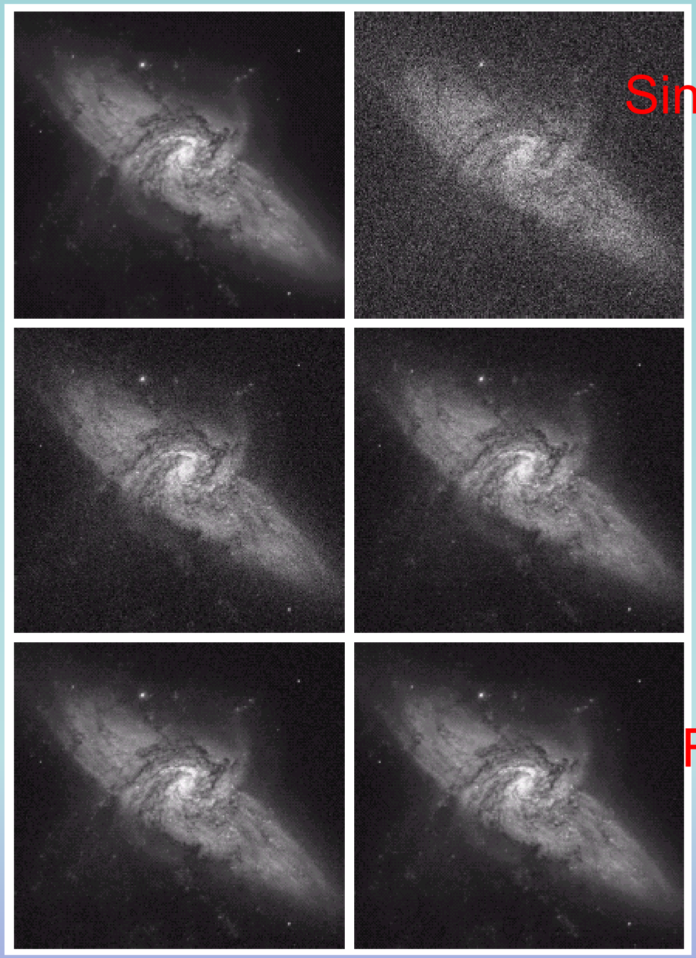
For simulation this noisy image is generated

with μ=0, б=64

CSE-BUET

Image Enhancement using Averaging

Averaged images



Single noisy images

From 8 images

From 16 images

From 64 images

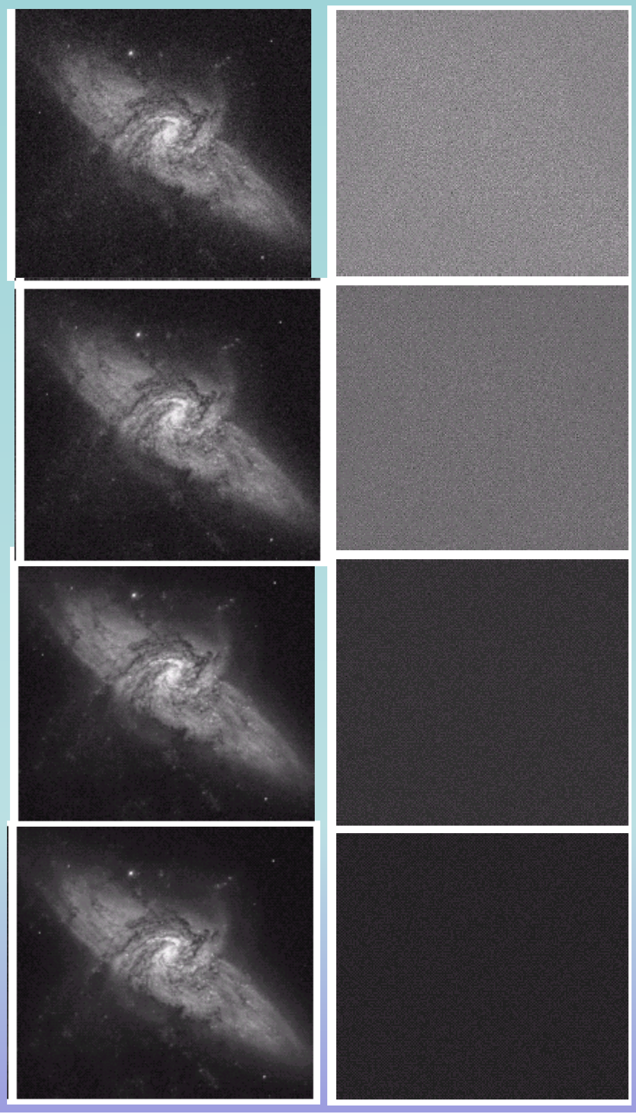
From128

images

CSE-BUET

Image Enhancement using Averaging

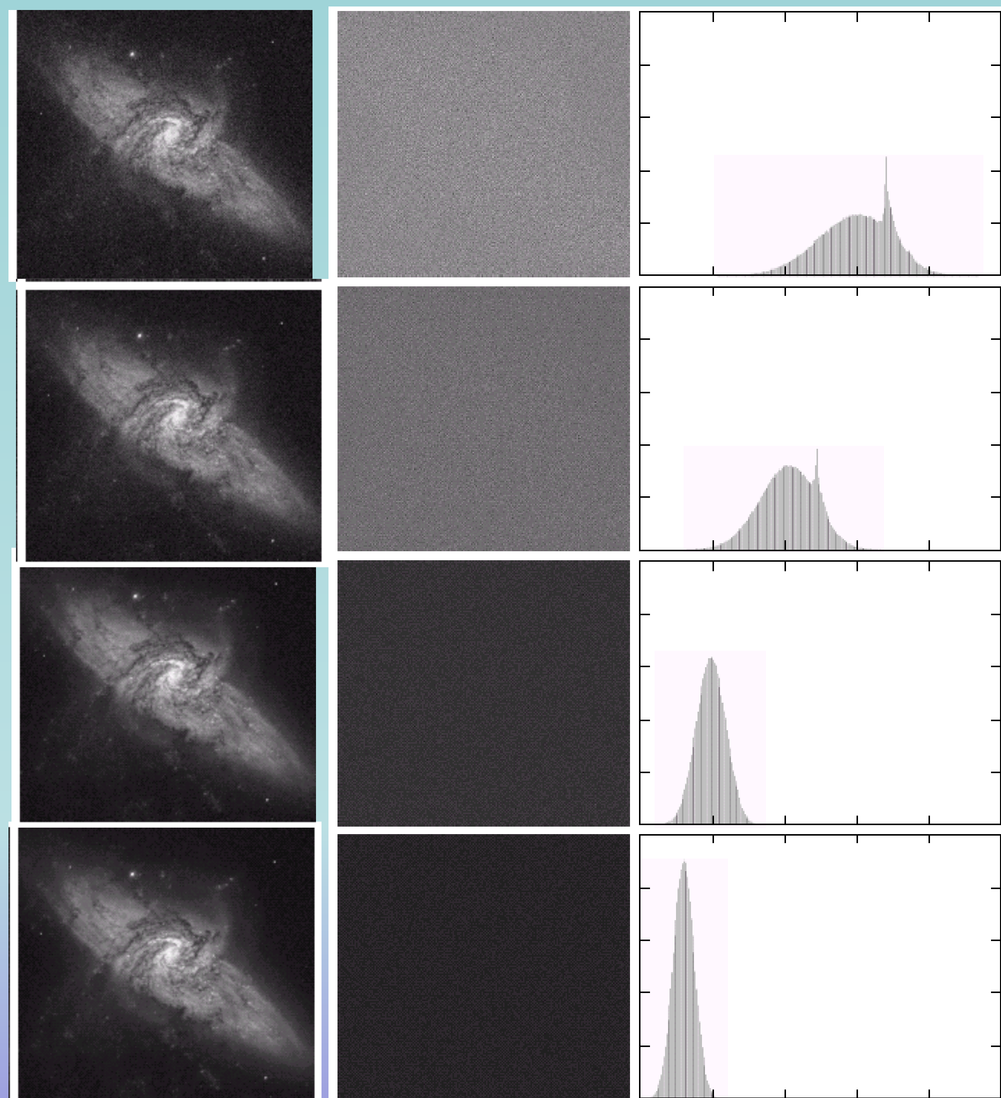
- As *K* increases, both standard deviation and mean decrease



CSE-BUET

Image Enhancement using Averaging

- As *K* increases, both standard deviation and mean decrease



CSE-BUET

Issues in Image Averaging

- The value of pixels will be in [0 255*K]
- Even, can have negative values due to noise
  - E.g., Gaussian r.v. with 0 mean and nonzero variance

CSE-BUET

Image Enhancement using Spatial Filtering

- Similar to neighborhood operation
- A *mask* or *filter* or *template* or *kernel* or *window* defines the neighborhood
- Mask size is usually $m \times n$
  - $m = 2a+1$, $n = 2b+1$
- Output pixel value is determined from the pixels under the mask

Image Enhancement using Spatial Filtering

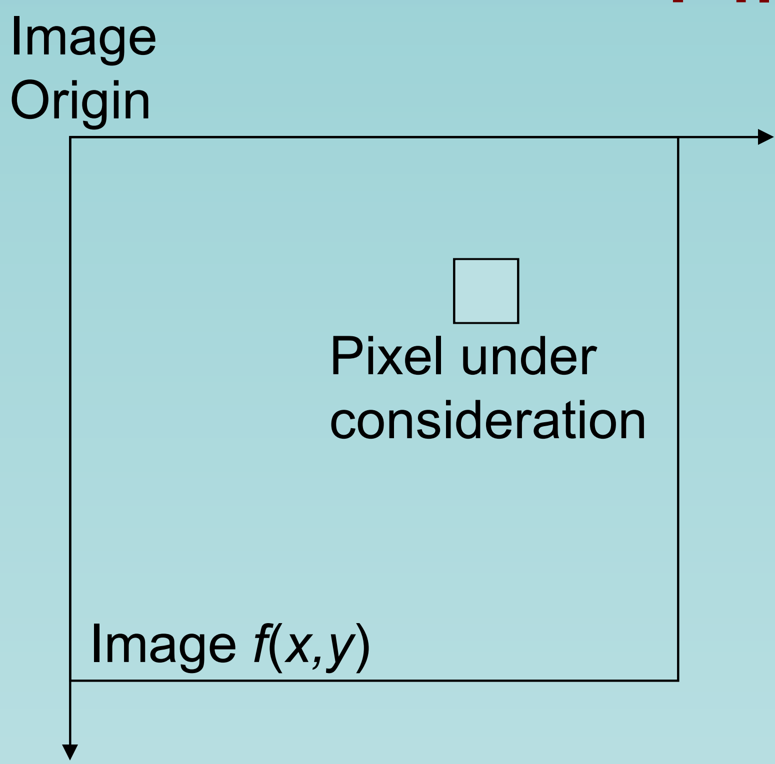
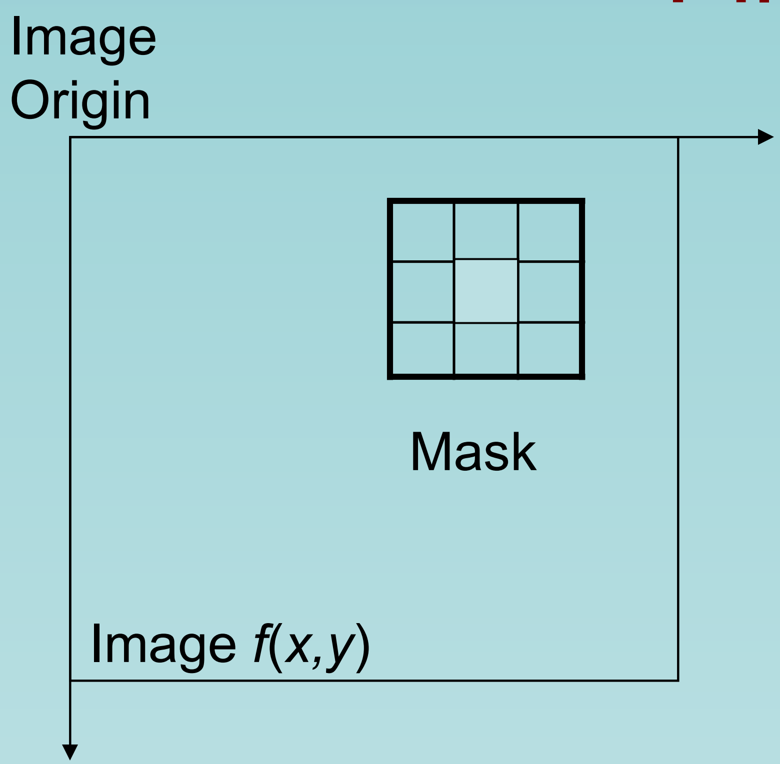
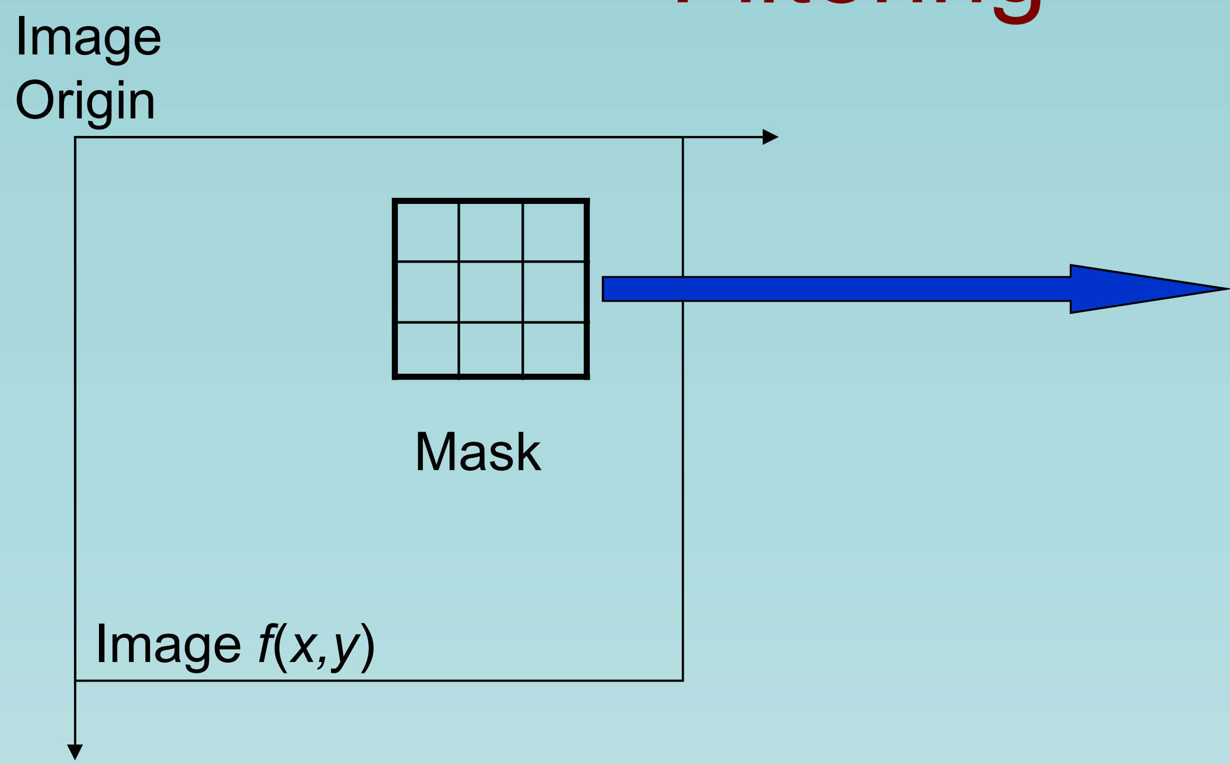


Image Enhancement using Spatial Filtering



CSE-BUET

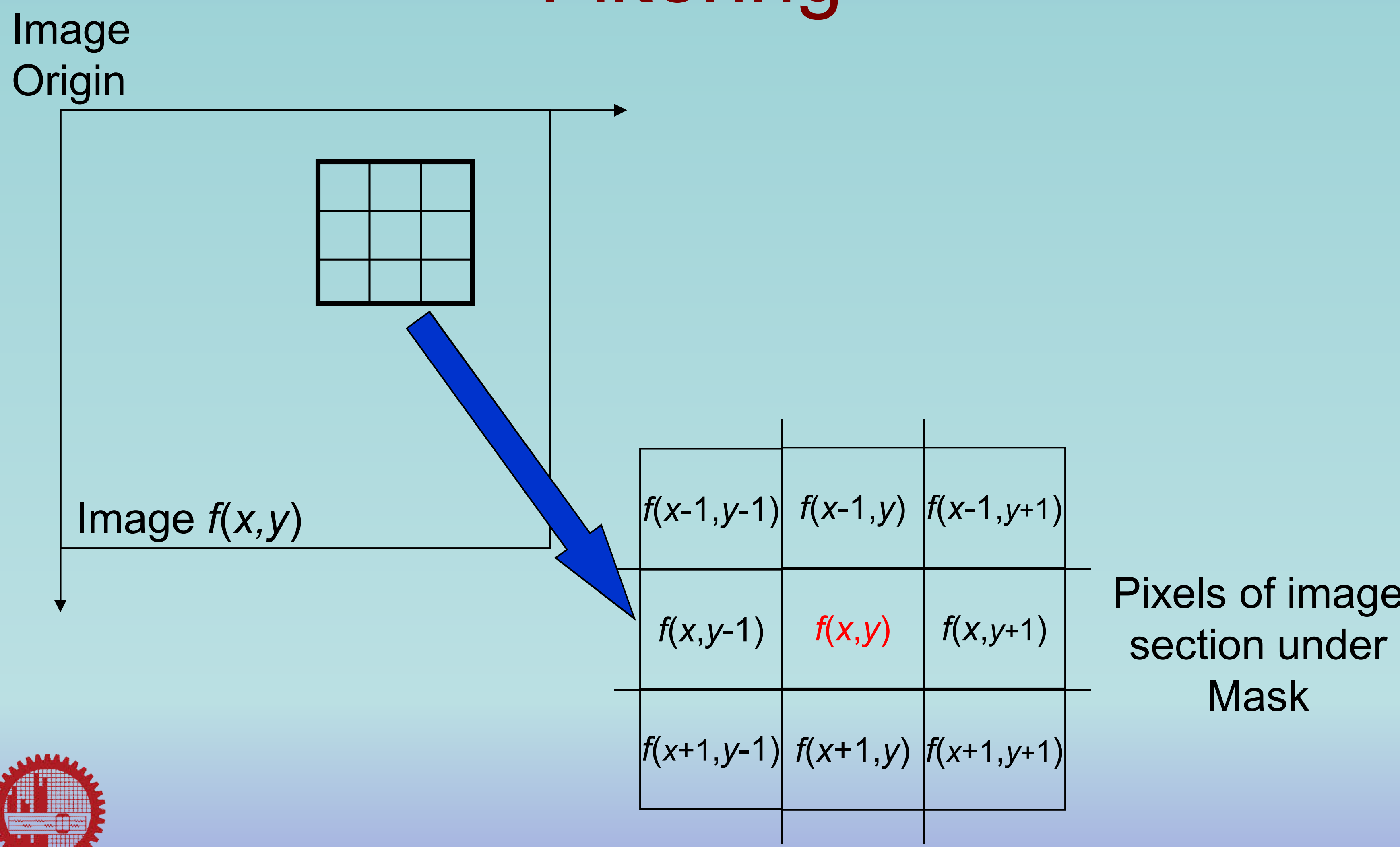
Image Enhancement using Spatial Filtering



w(-1,-1)	w(-1,0)	w(-1,1)
w(0,-1)	w(0,0)	w(0,1)
w(1,-1)	w(1,0)	w(1,1)

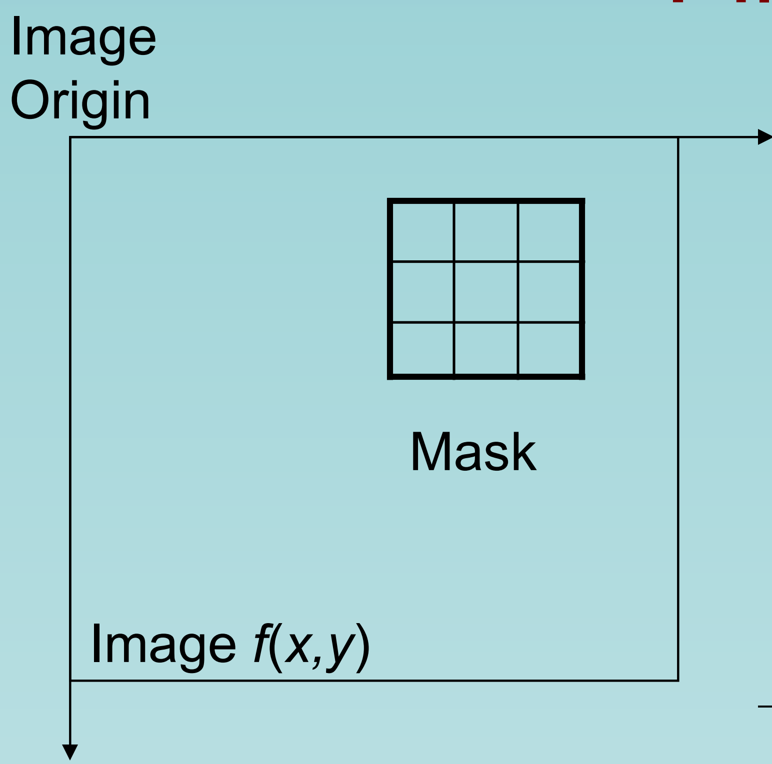
Mask Coefficients showing coordinate arrangement

Image Enhancement using Spatial Filtering



CSE-BUET

Image Enhancement using Spatial Filtering



w(-1,-1)	w(-1,0)	w(-1,1)
w(0,-1)	w(0,0)	w(0,1)
w(1,-1)	w(1,0)	w(1,1)

Mask Coefficients

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

Pixels under Mask

Image Enhancement using Spatial Filtering

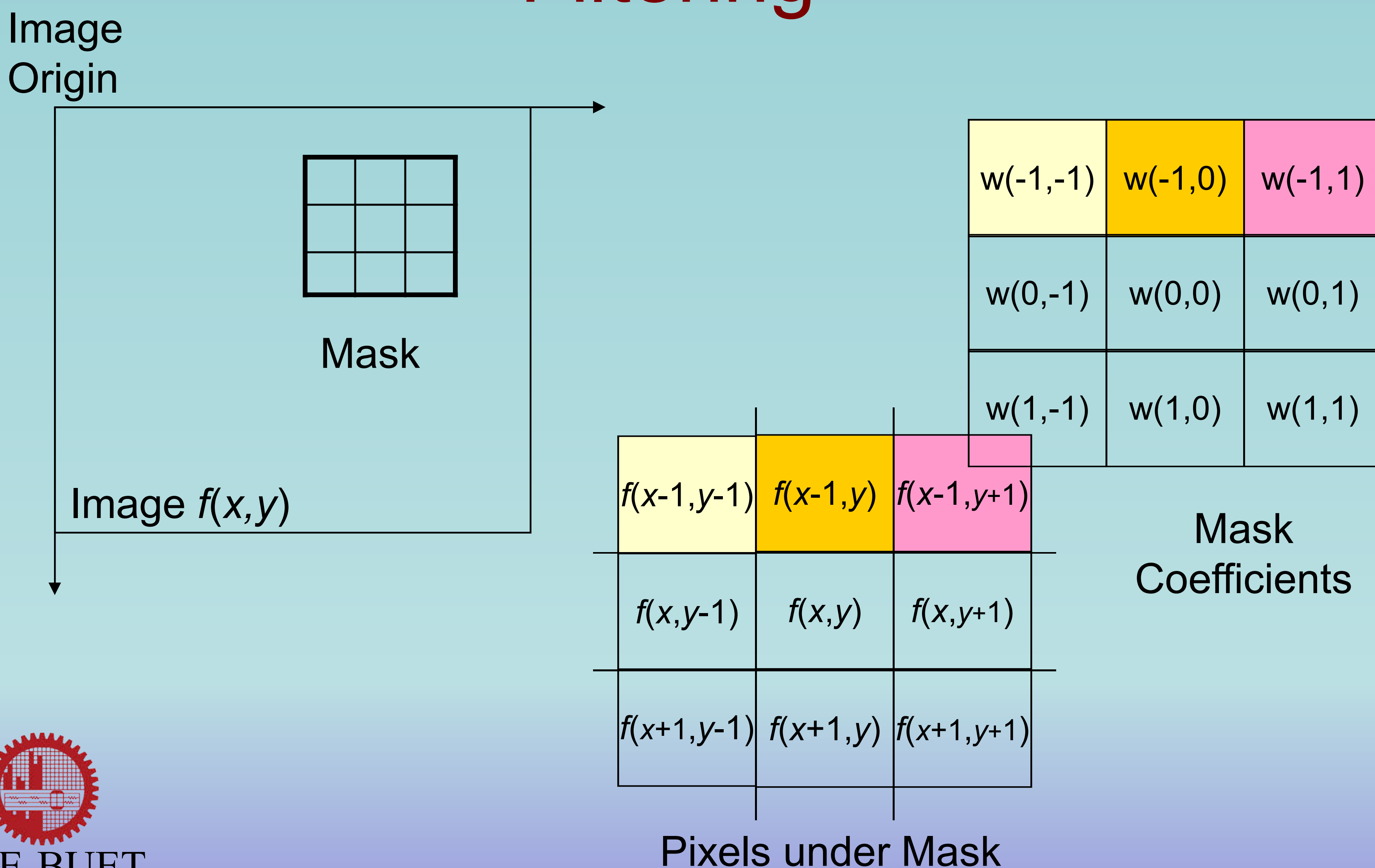


Image Enhancement using Spatial Filtering

Response of the filter at point (x, y):

$$R = w(-1,-1)f(x-1,y-1) + w(-1,0)f(x-1,y) + \cdots$$
$$\cdots + w(1,0)f(x+1,y) + w(1,1)f(x+1,y+1)$$

w(-1,-1)	w(-1,0)	w(-1,1)
w(0,-1)	w(0,0)	w(0,1)
w(1,-1)	w(1,0)	w(1,1)

Mask Coefficients

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

Pixels under Mask

CSE-BUET

Image Enhancement using Spatial Filtering

Response of the filter at point (*x, y*):

$$R = w(-1,-1)f(x-1, y-1) + w(-1,0)f(x-1, y) + \cdots$$
$$\cdots + w(1,0)f(x+1, y) + w(1,1)f(x+1, y+1)$$

w(-1,-1)	w(-1,0)	w(-1,1)
w(0,-1)	w(0,0)	w(0,1)
w(1,-1)	w(1,0)	w(1,1)

Mask Coefficients

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

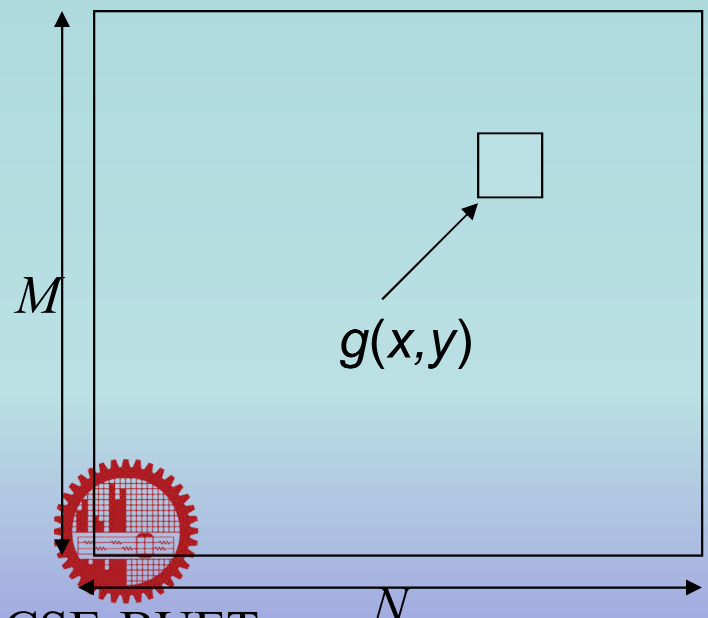
Pixels under Mask

**This type of response is called linear filtering

Image Enhancement using Spatial Filtering

A more general equation for response:

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



$2b+1$

$2a+1$

w(-1,-1)	w(-1,0)	w(-1,1)
w(0,-1)	w(0,0)	w(0,1)
w(1,-1)	w(1,0)	w(1,1)

Mask Coefficients

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

Image Enhancement using Spatial Filtering

A more general equation for response:

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

Looks like a convolution operation

Called convolution mask

$2b+1$

$2a+1$

w(-1,-1)	w(-1,0)	w(-1,1)
w(0,-1)	w(0,0)	w(0,1)
w(1,-1)	w(1,0)	w(1,1)

Mask Coefficients

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

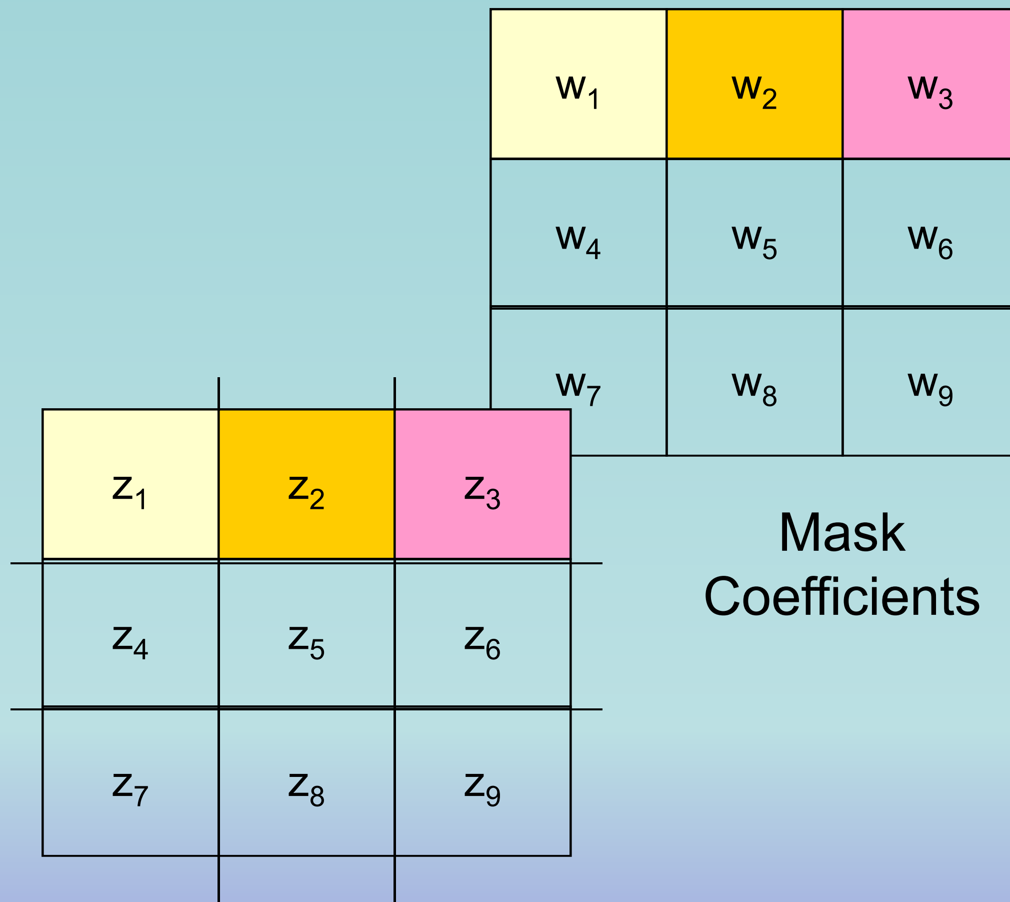
Image Enhancement using Spatial Filtering

Other form of the response:

$$R = \sum_{i=1}^{9} w_i z_i$$

Or, for a general case of mask size mXn:

$$R = \sum_{i=1}^{mn} w_i z_i$$



Mask Coefficients

Pixels under Mask

CSE-BUET

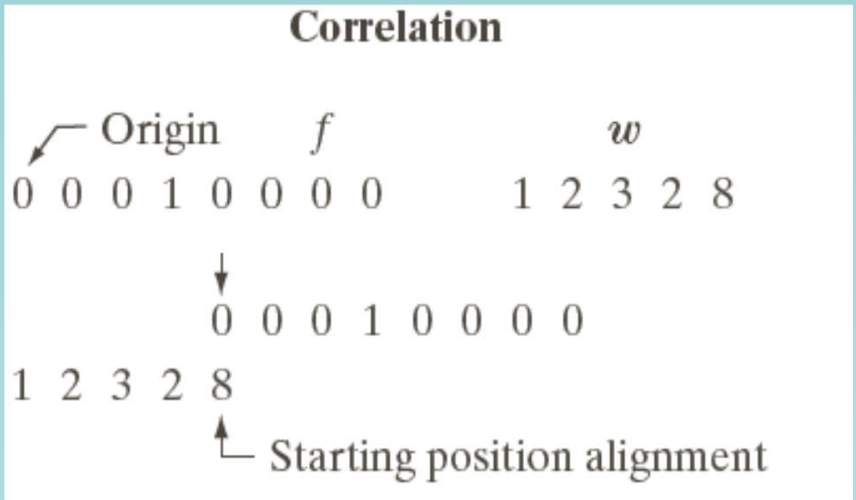
Spatial Filtering Vs. Correlation and Convolution

Correlation

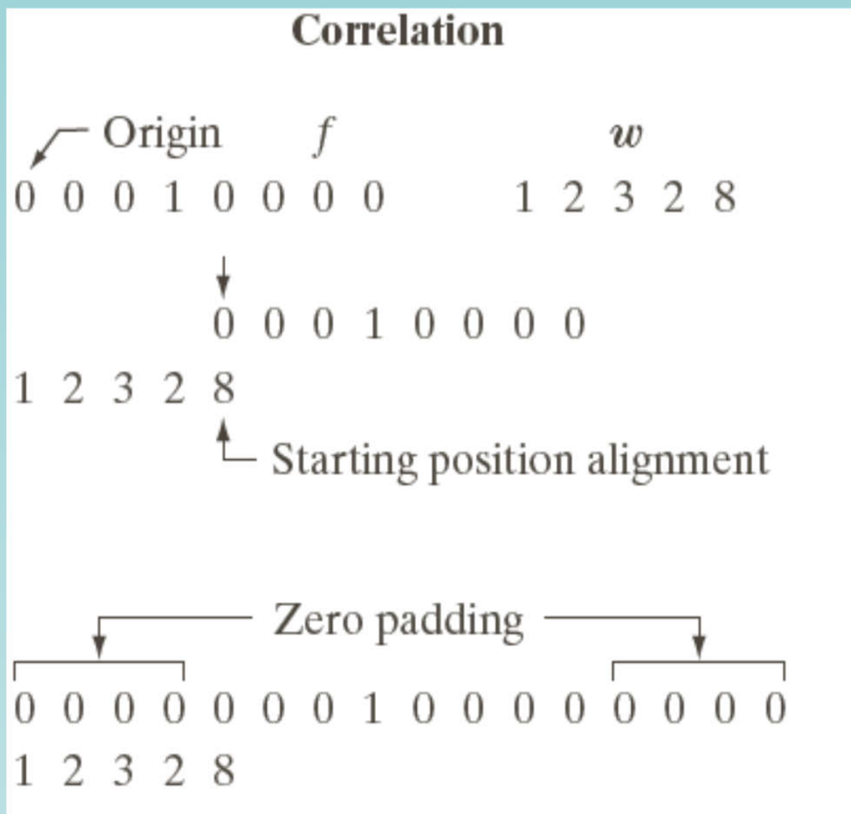
Origin	f	w
	0 0 0 1 0 0 0 0	1 2 3 2 8

CSE-BUET

Spatial Filtering Vs. Correlation and Convolution



Spatial Filtering Vs. Correlation and Convolution



Spatial Filtering Vs. Correlation and Convolution

Correlation

Origin f w

0 0 0 1 0 0 0 0 1 2 3 2 8

0 0 0 1 0 0 0 0

1 2 3 2 8

Starting position alignment

Zero padding

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

1 2 3 2 8

Full correlation result

$\boxed{0}$ 0 0 8 2 3 2 1 0 0 0 0

Spatial Filtering Vs. Correlation and Convolution

Correlation

Origin f w

0 0 0 1 0 0 0 0 1 2 3 2 8

0 0 0 1 0 0 0 0

1 2 3 2 8

Starting position alignment

Zero padding

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

1 2 3 2 8

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

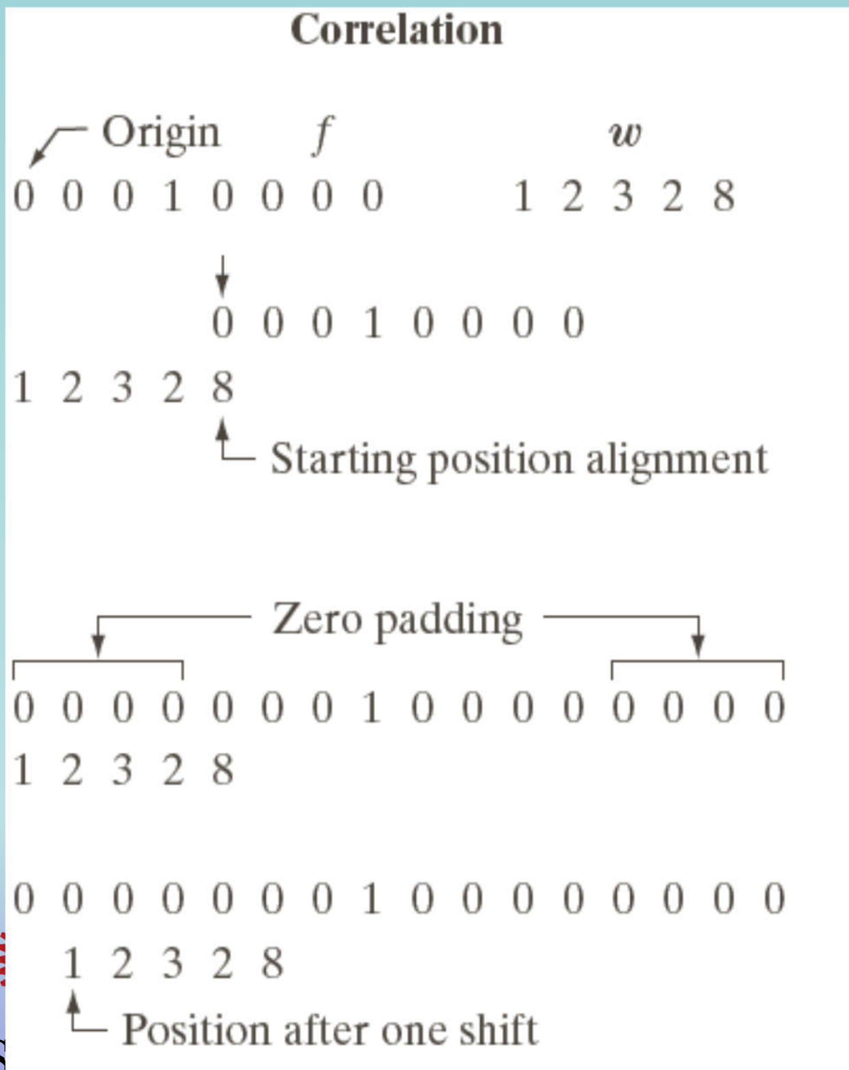
1 2 3 2 8

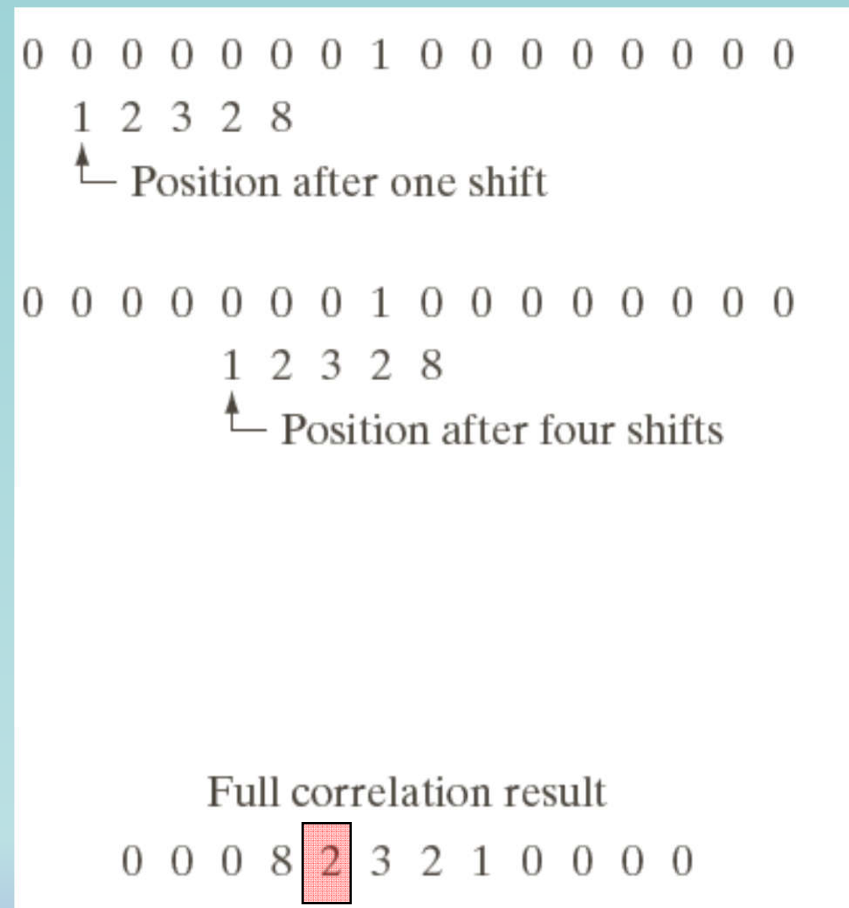
Position after one shift

Full correlation result

0 $\boxed{0}$ 0 8 2 3 2 1 0 0 0 0

Spatial Filtering Vs. Correlation and Convolution





Spatial Filtering Vs. Correlation and Convolution

Correlation

Origin f w

0 0 0 1 0 0 0 0 1 2 3 2 8

0 0 0 1 0 0 0 0

1 2 3 2 8

Starting position alignment

Zero padding

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

1 2 3 2 8

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

1 2 3 2 8

Position after one shift

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

1 2 3 2 8

Position after one shift

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

1 2 3 2 8

Position after four shifts

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

1 2 3 2 8

Final position

Full correlation result

0 0 0 8 2 3 2 1 0 0 0 $\boxed{0}$

CS

Spatial Filtering Vs. Correlation and Convolution

Correlation

Origin f w

0 0 0 1 0 0 0 0 1 2 3 2 8

0 0 0 1 0 0 0 0
1 2 3 2 8
Starting position alignment

Zero padding

0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
1 2 3 2 8

0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
1 2 3 2 8
Position after one shift

0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
1 2 3 2 8
Position after one shift

0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
1 2 3 2 8
Position after four shifts

0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
1 2 3 2 8
Final position

Full correlation result

0 0 0 8 2 3 2 1 0 0 0 0

Cropped correlation result

0 8 2 3 2 1 0 0

Spatial Filtering Vs. Correlation and Convolution

Convolution

Origin f w rotated 180°

0 0 0 1 0 0 0 0   8 2 3 2 1

0 0 0 1 0 0 0 0
8 2 3 2 1

Starting position

0 0 0 0 0 0 0 1 0 0 0 0 0 0 0
8 2 3 2 1

Full padded

0 0 0 0 0 0 0 1 0 0 0 0 0 0 0
8 2 3 2 1

Position after one shift

0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
8 2 3 2 1

Position after one shift

0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
8 2 3 2 1

Position after 4 shifts

0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
8 2 3 2 1

Final position

Full convolution result

0 0 0 1 2 3 2 8 0 0 0 0

Cropped convolution result

0 1 2 3 2 8 0 0

2D Correlation and Convolution

Origin $f(x, y)$

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

$w(x, y)$

1	2	3
4	5	6
7	8	9

(a)

Padded f

0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	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	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0

(b)

Initial position for w

1	2	3	0	0	0	0	0	0
4	5	6	0	0	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	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	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0

Full correlation result

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

Cropped correlation result

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

2D Correlation and Convolution

Rotated w

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

Full convolution result

0	0	0	0	0	0	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	2	3	0	0	0
0	0	0	4	5	6	0	0	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	0	0	0	0	0	0

Cropped convolution result

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

CSE-BUET

Examples of Spatial Filtering: Smoothing or Low Pass Filtering

- Averaging or low pass filtering
- Replace every pixel by the average of its neighbor
- Reduce sharp transitions
- Removes or blurs some of the edges
- Helps to find gross elements, ignoring fine details
- Removes false contouring

Examples of Spatial Filtering: Smoothing or Low Pass Filtering

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

- Easy to implement: $R = \sum_{i=1}^{9} z_i$
- The division is carried out only once

CSE-BUET

Examples of Spatial Filtering: Smoothing or Low Pass Filtering

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

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

- Easy to implement: $$R = \sum_{i=1}^{9} z_i$$
- The division is carried out only once

Examples of Spatial Filtering: Smoothing or Low Pass Filtering

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

- More general formula for smoothing filter:

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

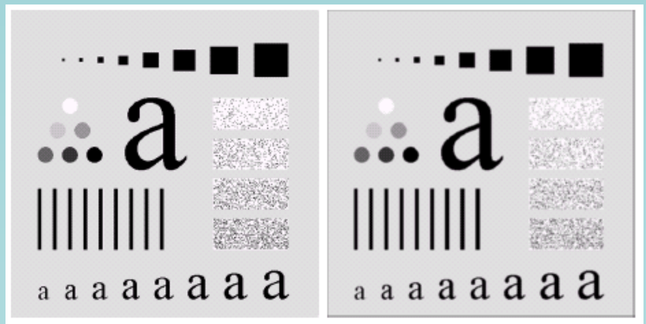
Examples of Spatial Filtering: Smoothing or Low Pass Filtering

- Black square: 3, 5, 9, 15, 25, 35, 45
- Border 25 pixels apart
- Circle: 25 pixels, gray: 0, 20, ..., 100%
- Small a's: 10, 12, 24 pixels
- Large a: 60 pixels
- Bars: 5X100 pixels, Separated by 20 pix
- Noise boxes: 50X120 pixels

Examples of Spatial Filtering: Smoothing or Low Pass Filtering

- Black square: 3, 5, 9, 15, 25, 35, 45
- Border 25 pixels apart
- Circle: 25 pixels, gray: 0, 20, ..., 100%
- Small a's: 10, 12, 24 pixels
- Large a: 60 pixels
- Bars: 5X100 pixels, Separated by 20 pix
- Noise boxes: 50X120 pixels

- Smoothing using mask of size 3X3

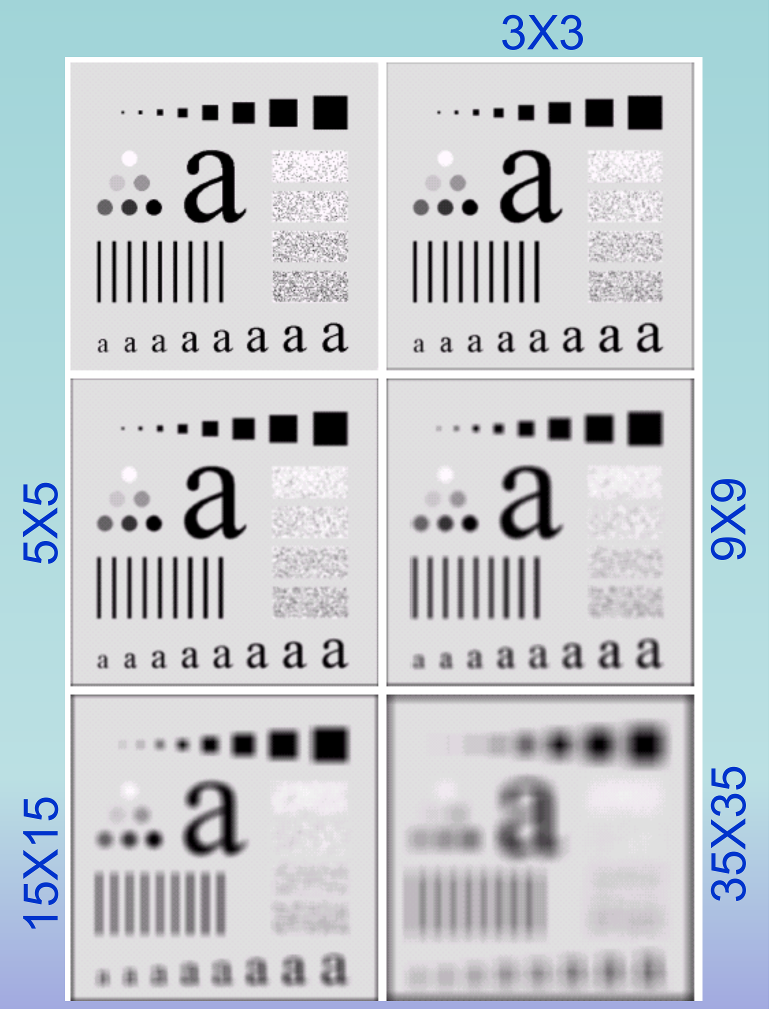


CSE-BUET

Examples of Spatial Filtering: Smoothing or Low Pass Filtering

- Black square: 3, 5, 9, 15, 25, 35, 45
- Border 25 pixels apart
- Circle: 25 pixels, gray: 0, 20, ..., 100%
- Small a’s: 10, 12, 24 pixels
- Large a: 60 pixels
- Bars: 5X100 pixels, Separated by 20 pix
- Noise boxes: 50X120 pixels

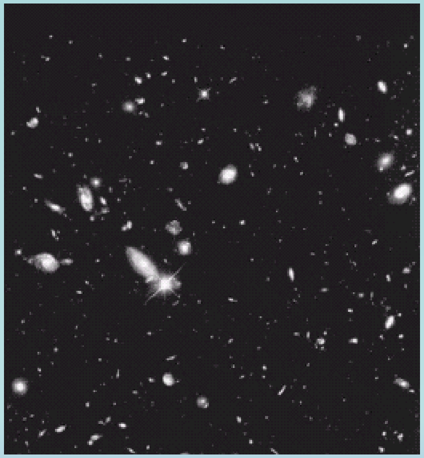
- Smoothing using masks of diff sizes



CSE-BUET

Another Example of Spatial Filtering

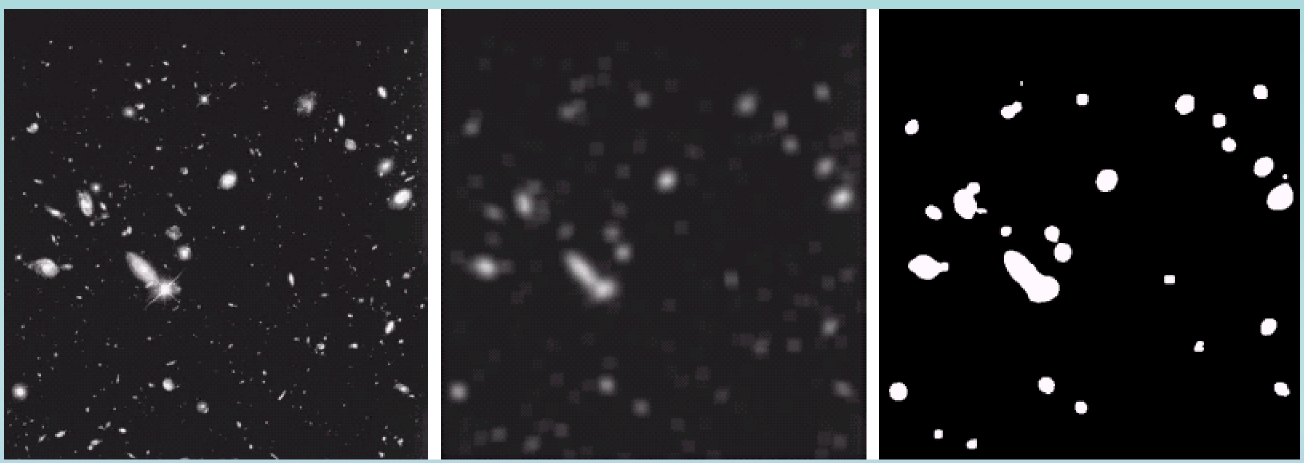
- Blurring helps to find major objects
- Removes fine details
- Original image contains lots of small objects



Original image

Another Example of Spatial Filtering

- Blurring helps to find major objects
- Removes fine details
- Original image contains lots of small objects



Original | Smoothed by 15X15 mask | Thresholding after smoothing

Other Spatial Filtering: Order Statistics filtering

Consider this image segment

- Centre pixel value is significantly different from its neighbors
- Smoothing will reduce the centre, but increase the others

10	20	20
20	100	20
25	20	15

Other Spatial Filtering: Order Statistics filtering

Consider this image segment

- Centre pixel value is significantly different from its neighbors
- Smoothing will reduce the centre, but increase the others
- Solution:
  - Centre will be more or less similar to others

10	20	20
20	100	20
25	20	15

Other Spatial Filtering: Order Statistics filtering

- Nonlinear filtering
- Example: median filtering
- Response based on order or ranking
- Example: *median filtering*
- Adv:
  - *Less blurring effect*
  - *removes random noise*
  - *Force point to be more like their neighbors*
- Policy:
  - *Sort the values of enclosed pixels*
  - *Select the median as output pixel level*

Other Spatial Filtering: Order Statistics filtering

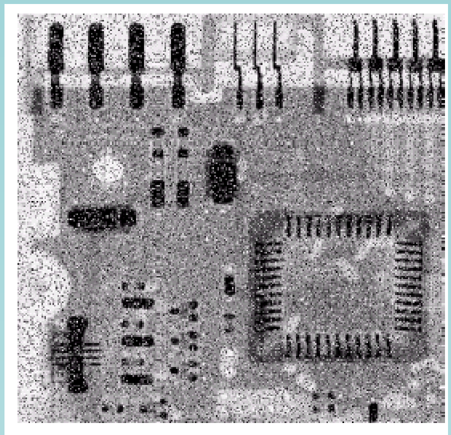
- Policy:
  - *Sort the values of enclosed pixels*
  - *Select the median as output pixel level*
- Example:
  - Let, 3X3 mask size
  - Let Image pixel values under masks
  - Sorted values: 10, 15, 20, 20, 20, 20, 20, 25, 100
  - Median is 20
  - Output pixel value is 20

10	20	20
20	100	20
25	20	15

Image pixel values under masks

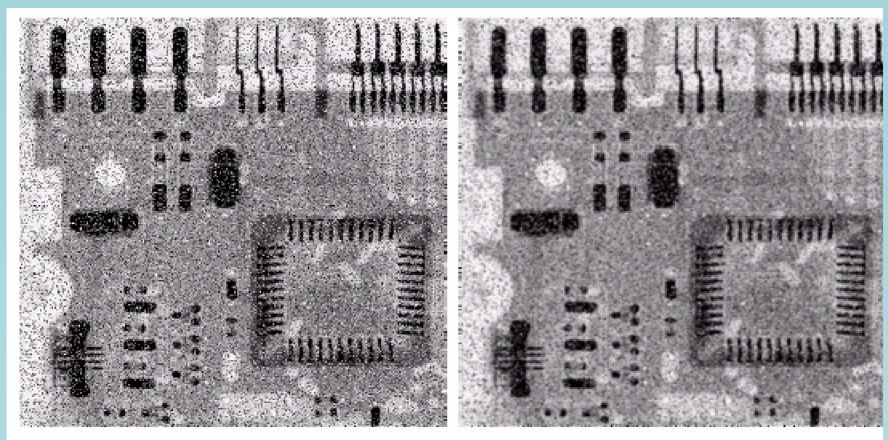
CSE-BUET

Example of Averaging and Median Filters



Noisy image

Example of Averaging and Median Filters

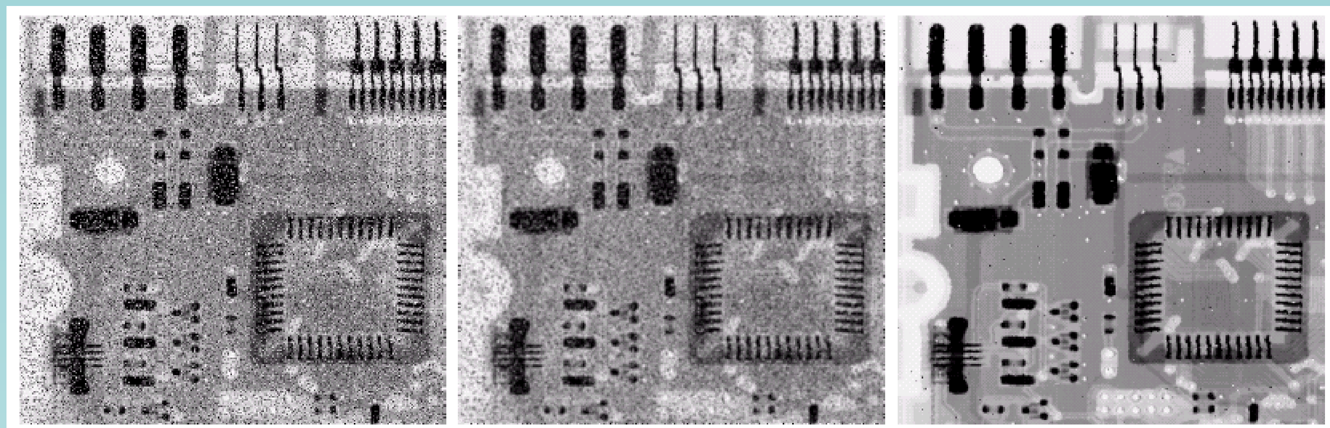


Noisy image

Noise reduction by Avg. filter

CSE-BUET

Example of Averaging and Median Filters



Noisy image | Noise reduction by Avg. filter | Noise reduction by median filter

- Isolated dark or light clusters of size smaller than $n^2/2$ are removed by median filter of size nXn

CSE-BUET

Other Order Statistics filters

- median filter is 50th percentile filter
- Others are:
  - *100^{th} percentile filter or max filter*
    - *Replace with the brightest pixel in neighborhood*
  - *0^{th} percentile filter or min filter*
    - *Replace with the darkest pixel in neighborhood*

Issues in spatial filtering

- When the mask moves closer to the border
  - safe when the center of the mask is at distance (*n*-1)/2 pixels away from the border
  - Closer to that distance: some rows/columns of the mask are out of image
- Way out:
  - *Ignoring the outside columns/rows*
  - *Padding*
    - *Zero padding*
    - *Mirror padding*