19CSE367 Digital Image Processing

SARATH TV

Last lecture

- Histograms Equalization.

Spatial Filtering

- Image enhancement – filtering principles.
- Filtering –concept from frequency domain processing.
- Passing/rejecting a specified component.
- Eg- low/high frequency filter
- On an image-effect → smoothen the image → blurring.
- Spatial filtering modifies an image by replacing the value of each pixel by a function of values of the pixel & its neighbors.
- Linear and non linear spatial filters based on the operation performed on the image pixels.

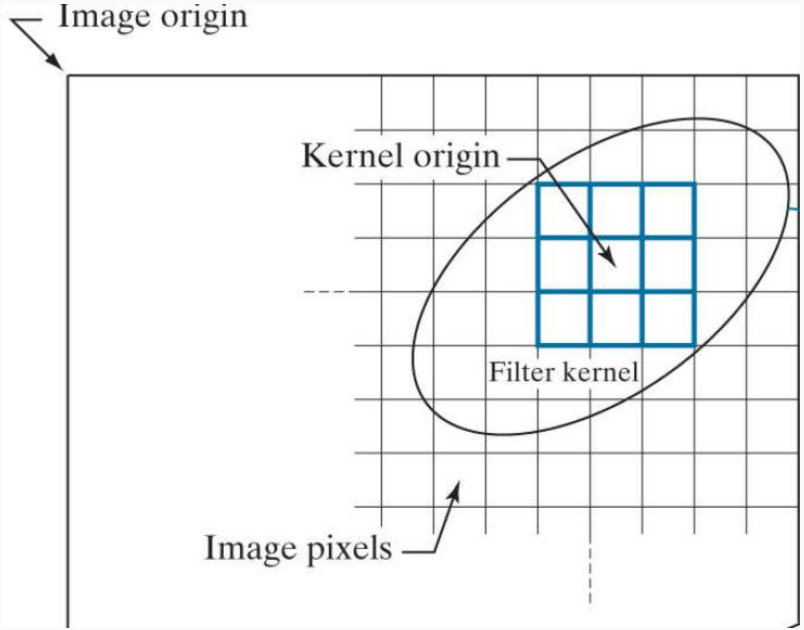
Linear spatial filtering mechanism

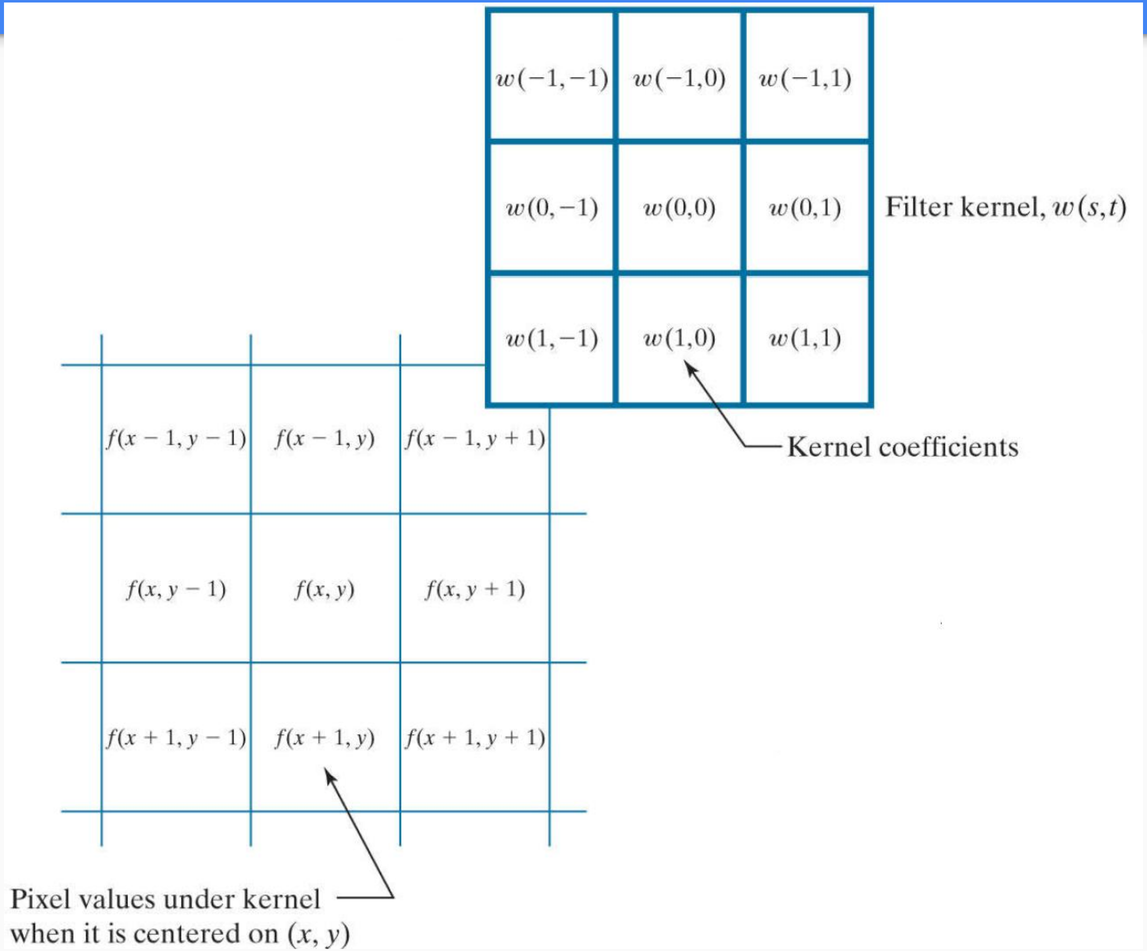
1	1	1
1	1	1
1	1	1

- Sum of products operations.
- Image f and image kernel w
- Kernel- An array which defines the neighborhood of operation
- Other names→ *Mask, template, window or filter kernel*
- Image – f(x,y)
- 3x3 kernel
- Response image
  - $g(x,y) = w(-1,-1)f(x-1,y-1) + w(-1,0)f(x-1,y) + \cdots . w(0,0)f(x,y) + \cdots w(1,1)f(x+1,y+1)$
- Centre of kernel moves from pixel to pixel.
- **Centre coefficient w(0,0)** aligns with the pixel at location (x,y)
- Linear spatial filtering is given as
- $g(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) f(x+s, y+t)$
- Centre of kernel visits every pixel of the image once.

Spatial correlation & convolution

- Correlation -moving the center of the kernel over an image and computing the sum of products at each location.
- Convolution – same thing but the kernel is rotated by 180^0.
- Convolution and correlation will yield the same result if the values of kernel are symmetric about its center.

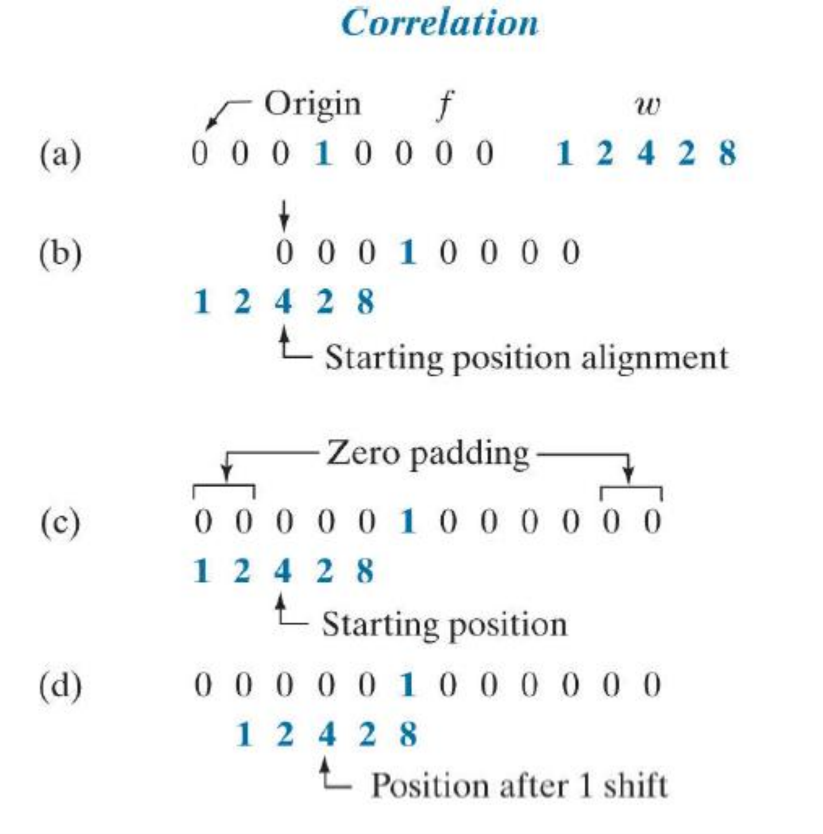




w(−1,−1)
w(−1,0)
w(−1,1)
w(0,−1)
w(0,0)
w(0,1)
w(1,−1)
w(1,0)
w(1,1)
Filter kernel, w(s,t)
Kernel coefficients
f(x − 1, y − 1)
f(x − 1, y)
f(x − 1, y + 1)
f(x, y − 1)
f(x, y)
f(x, y + 1)
f(x + 1, y − 1)
f(x + 1, y)
f(x + 1, y + 1)
Pixel values under kernel
when it is centered on (x, y)

One Dimensional example

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



(e) 0 0 0 0 0 **1** 0 0 0 0 0 0

1 2 4 2 8

Position after 3 shifts

(f) 0 0 0 0 0 **1** 0 0 0 0 0 0

1 2 4 2 8

Final position

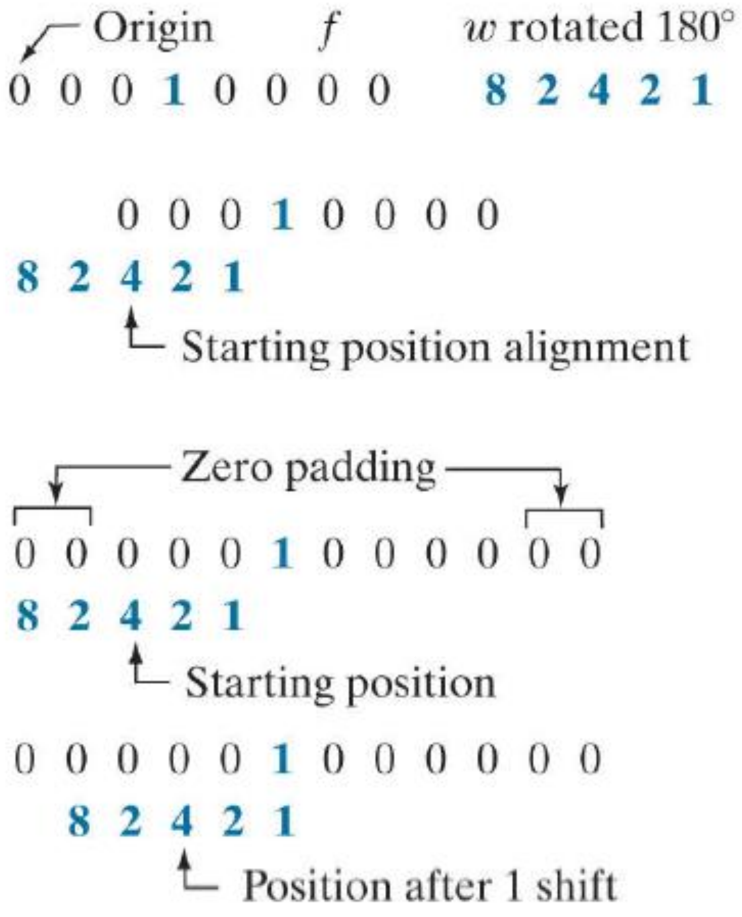
Correlation result

(g) 0 8 2 4 2 1 0 0

Extended (full) correlation result

(h) 0 0 0 8 2 4 2 1 0 0 0 0

Convolution



0 0 0 0 0 **1** 0 0 0 0 0 0

8 2 4 2 1

Position after 3 shifts

0 0 0 0 0 **1** 0 0 0 0 0 0

8 2 4 2 1

Final position

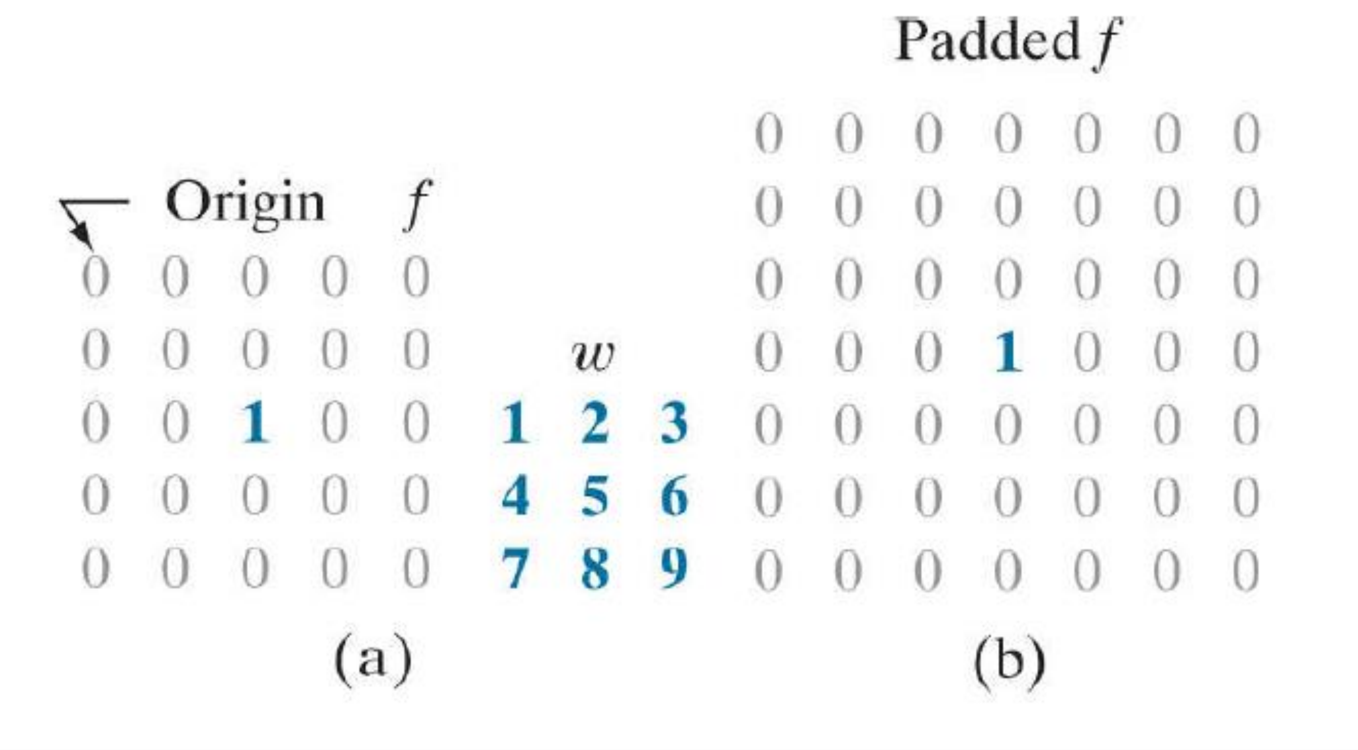
Convolution result

0 1 2 4 2 8 0 0

Extended (full) convolution result

0 0 0 1 2 4 2 8 0 0 0 0

For images


Origin
f
w
Padded 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
1 2 3
4 5 6
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 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
(a)
(b)

Initial position for w

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

(c)

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

(d)

Full correlation result

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

(e)

Rotated w

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

(f)

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

(g)

Full convolution result

0	0	0	0	0	0	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

(h)

Averaging and Gaussian kernel

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

$$\frac{1}{4.8976} \times \begin{array}{|c|c|c|} \hline 0.3679 & 0.6065 & 0.3679 \\ \hline 0.6065 & 1.0000 & 0.6065 \\ \hline 0.3679 & 0.6065 & 0.3679 \\ \hline \end{array}$$

1. Consider a 5x5 matrix A = $\begin{bmatrix} 1 & 6 & 11 & 16 & 21 \\ 2 & 7 & 12 & 17 & 22 \\ 3 & 8 & 13 & 18 & 23 \\ 4 & 9 & 14 & 19 & 24 \\ 5 & 10 & 15 & 20 & 25 \end{bmatrix}$

2. Define a 3x3 Kernel to perform spatial averaging

$$\text{avg3} = \begin{bmatrix} 0.1111 & 0.1111 & 0.1111 \\ 0.1111 & 0.1111 & 0.1111 \\ 0.1111 & 0.1111 & 0.1111 \end{bmatrix}$$

3. Pad the matrix A with zeros

$$B = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 6 & 11 & 16 & 21 & 0 \\ 0 & 2 & 7 & 12 & 17 & 22 & 0 \\ 0 & 3 & 8 & 13 & 18 & 23 & 0 \\ 0 & 4 & 9 & 14 & 19 & 24 & 0 \\ 0 & 5 & 10 & 15 & 20 & 25 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

4. Place a 3x3 window on B and fetch the data.

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 6 & 11 & 16 & 21 & 0 \\ 0 & 2 & 7 & 12 & 17 & 22 & 0 \\ 0 & 3 & 8 & 13 & 18 & 23 & 0 \\ 0 & 4 & 9 & 14 & 19 & 24 & 0 \\ 0 & 5 & 10 & 15 & 20 & 25 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

5. Multiply the window with the kernel.

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 6 \\ 0 & 2 & 7 \end{bmatrix} \times \begin{bmatrix} 0.1111 & 0.1111 & 0.1111 \\ 0.1111 & 0.1111 & 0.1111 \\ 0.1111 & 0.1111 & 0.1111 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0.111 & 0.667 \\ 0 & 0.222 & 0.778 \end{bmatrix}$$

6. Find the sum of the result obtained in step 5 and update the result.[0+0+0+0+0.1111+0.222+0+0.667+0.778]=1.778
7. Output Matrix (5x5) with updated value.

$$\text{Output}=\begin{bmatrix} 1.778 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

8. Now slide the window to the next position on B and fetch the data.

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 6 & 11 & 16 & 21 & 0 \\ 0 & 2 & 7 & 12 & 17 & 22 & 0 \\ 0 & 3 & 8 & 13 & 18 & 23 & 0 \\ 0 & 4 & 9 & 14 & 19 & 24 & 0 \\ 0 & 5 & 10 & 15 & 20 & 25 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

9. Repeat the process of multiplying it with the kernel (step 5),finding the sum (step 6) and update the result.(Step 7)

$$\text{Output} = \begin{bmatrix} 1.778 & 4.333 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

10. Similarly, perform the steps 5 through 7 by sliding the window on the whole matrix.
11. Final updated matrix, Output =

$$\begin{bmatrix} 1.7778 & 4.3333 & 7.6667 & 11.000 & 8.4444 \\ 3.000 & 7.000 & 12.000 & 17.000 & 13.000 \\ 3.6667 & 8.000 & 13.000 & 18.000 & 13.6667 \\ 4.3333 & 9.000 & 14.000 & 19.000 & 14.3333 \\ 3.1111 & 6.3333 & 9.6667 & 13.000 & 9.7778 \end{bmatrix}$$

- Imfilter in matlab
- Arguments for the function


THANK YOU!