Date: __/__/__ Page No.__

HOMEWORK-2

NetId: iv447
University Id: N17385760

A) Theoretical Questions:

A1) Linear Shift Invariant System (LSI)

Given the LSI criteria:

- Shift invariant: if $g(x) = w(x) * f(x)$, then $w(x) * f(x - x_0) = g(x - x_0)$, where $w(x)$ is the filter and $f(x)$ the original signal.
- Linearity would require $Op(A+B) = Op(A) + Op(B)$, where Op is the filter operation (Smoothing or median) and A and B are sets of pixels to be processed.

Discuss/Show that Smoothing filter with a $\frac{1}{3}[1, 1, 1]$ filter kernel is linear & Shift invariant.

Now apply a Median filter using 3 neighborhood. Discuss "Why" or "Why not" Median filtering is an LSI system. You can make a numerical example of Set A and B to demonstrate.

Solution: Kernel: $\frac{1}{3}[1, 1, 1]$

We have one-D filter.

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

Where $w(s)$ is filter and $f(x)$ is original

Date: / / Page No.

image.

Input Image

50	60	100	80
87	95	103	110
90	150	36	104
50	47	205	77

Filter

$\frac{1}{3}[1 \quad 1 \quad 1]$

Output Image

69.3	79.2	.
94.05	101.64	.
91.08	95.7	.
99.66	108.57	.

Applying linear operation of pixel with intensity 95

$= 87 * 0.33 + 95 * 0.33 + 103 * 0.33$

$= 28.71 + 31.35 + 33.99$

$= 94.05$

Similarly, we can calculate for 103, 150 and 36 pixel.

* A system is shift invariant when
  $y(n)$ is response to $x(n)$ and
  $y(n+k)$ is response to $x(n+k)$.

* A system is linear when
  $w_0(\alpha e + \beta f) = w_0 \alpha e + w_0 \beta f$

* Here, we can see that we can apply filter to the sum of signals or we can apply filter to each one of them and then add the result.

For eg: Take pixel with intensity value 103.

Date: / / Page No.

$= (95+103+110) * 0.33 = 101.64$

$= 95 * 0.33 + 103 * 0.33 + 110 * 0.33 = 101.64$

Therefore, Smoothing filter with a $\frac{1}{3}[1,1,1]$ filter kernel is linear and Shift invariant.

(*) Median filter Using 3 neighborhood.

In median filter, it considers each pixel in the image in turn and looks at its nearby neighbors. Instead of simply replacing pixel value with the mean of neighboring pixel values, it replaces it with the median of those values. Calculation of median:

1. Sort all the pixels values from the surrounding neighborhood.
2. Calculate the median.

Eg:

Input Image:

122	124	126	127	135
118	120	150	125	134
119	115	119	123	133
111	116	110	120	130

Output Image:

0	0	0	0	0
0	120	124	127	0
0	118	120	125	0
0	0	0	0	0

Calculating median value for pixel value 150:

Neighborhood values in sorted order

115, 119, 120, 123, 124, 125, 126, 127, 150.

Median Value: 124

Median filtering is Non-Linear.

Median [A(x) + B(x)] ≠ median [A(x)] + median [B(x)].

⊛ Sorting is done differently in every neighborhood because we have a new set of pixel values from scratch. Therefore, it is not linear Shift invariant.

A2) 1-D Convolution

Convolve a [1, 1, 1] filter with a [0, 0, 0, 1, 1, 1, 0, 0, 0] Signal and plot/graph the result.

Solution: Convolution:

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

0	0	0	1	1	1	0	0	0

Filter:

1	1	1

1	1	1

1	1	1

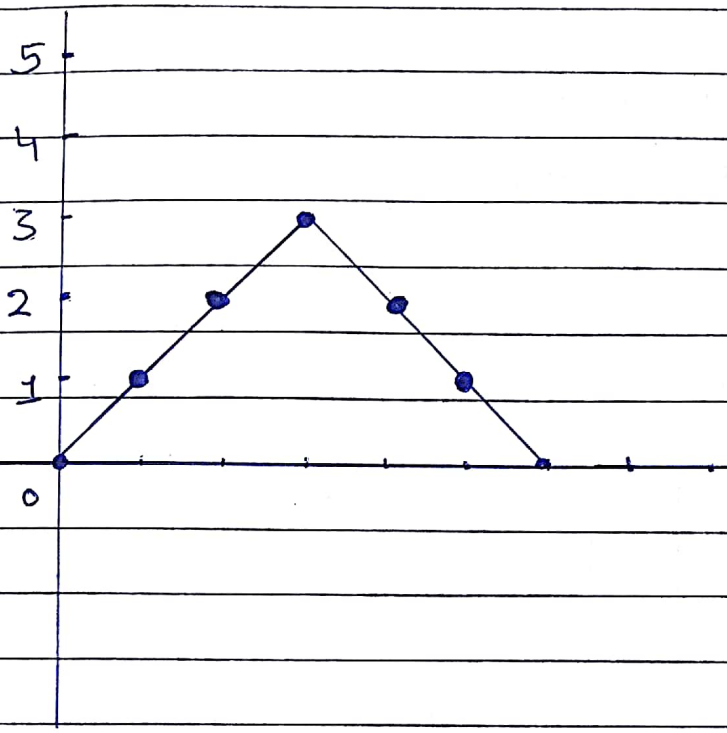
1	1	1

1	1	1

1	1	1

Output :

0	1	2	3	2	1	0

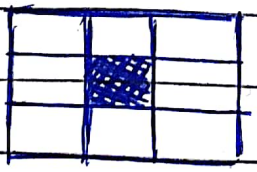


When these two functions are convolved, the result (plot/graph) is triangle.

A3) Median Filtering

See the following Synthetic images, where empty pixels are 0 and shaded pixels are 1. Apply 3x3 median filter to these 3 images, using Sliding window filtering operation as discussed in course. Note that the border can not be filtered, so that you can move the window only within the outer, dark boundary as shown.

3x3 Median Filter



Date: / / Page No.

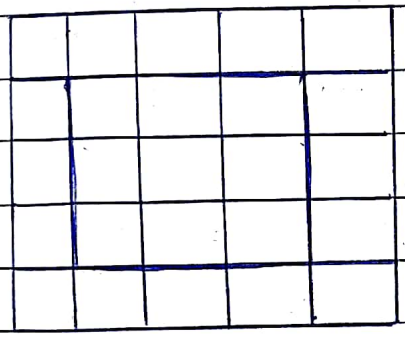
① Original Image : Bar

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

→

	0	0	0	
	0	0	0	
	0	0	0	

Final Result



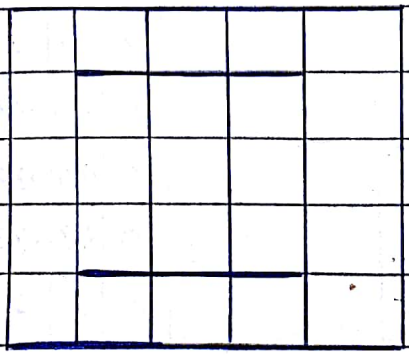
② Original Image : a point

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	

Final Result



Date: / / Page No.

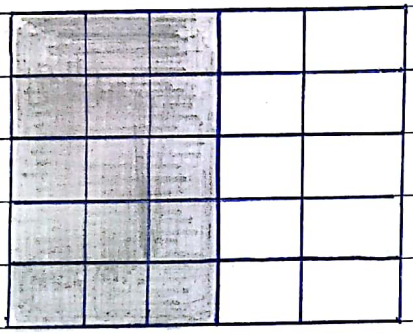
③ Original Image: intensity edge

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

→

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

Final Result



Median filter eliminated bar and point but the edge structure was preserved.

④ 3×3 Linear Filter

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

① Original Image : bar

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

→

0	0	1	0	0
0	.33	.33	.33	0
0	.33	.33	.33	0
0	.33	.33	.33	0
0	0	1	0	0

Date: / / Page No.

② Original Image : a point

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

→

Result

0.11	0.11	0.11
0.11	0.11	0.11
0.11	0.11	0.11

③ Original Image : Intensity edge

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

→

Result

1	1	1	0	0
1	0.99	0.66	0.33	0
1	0.99	0.66	0.33	0
1	0.99	0.66	.33	0
1	1	1	0	0

Linear smoothing blurs all three : bar, point and edge.

A4) Seperability

Show that the 2D Gaussian filter ($G(x, y; \sigma)$) function can be seperated into a product of two 1D components $G(x; \sigma)$ and $G(y; \sigma)$, each only be a function of x or y, respectively.

2-D Gaussian : $\frac{1}{2\pi\sigma^2} e^{\frac{-(x^2+y^2)}{2\sigma^2}}$

a) decompose the 2D Gaussian into the two components.

Date: / / Page No.

Answer) $\Rightarrow \frac{1}{2\pi\sigma^2} \exp^{\frac{-(x^2+y^2)}{2\sigma^2}}$

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

$= \frac{1}{\sqrt{2\pi}\sigma} \exp^{\frac{-x^2}{2\sigma^2}} \times \frac{1}{\sqrt{2\pi}\sigma} \exp^{\frac{-y^2}{2\sigma^2}}$

$G(x;\sigma) = \frac{1}{\sqrt{2\pi}\sigma} \exp^{-x^2/2\sigma^2}$

$G(y;\sigma) = \frac{1}{\sqrt{2\pi}\sigma} \exp^{-y^2/2\sigma^2}$

b) discuss how this helps to speed up the filtering operation by making an example of mxm filter size covering the Guassian, and then use the two 1-D filters.

Answer) Guassian kernel is seperable, which allows fast Computation

For eg:-

1	2	1
2	4	2
1	2	1

$\times \frac{1}{16}$ Can be seperated to two one-D kernels.

Date: / / Page No.

1	2	1
2	4	2
1	2	1

$* \frac{1}{16} =$

1
2
1

$* \frac{1}{4} *$ | 1 | 2 | 1 | $* \frac{1}{4}$

$$W = W_x * W_y.$$

First we can apply horizontal filter (kernel) and then vertical kernel on intermediate result.

original Image

15	20	25	25	15	10
20	15	50	30	20	15
20	50	55	60	30	20
20	15	65	30	15	30
15	20	30	20	25	30
20	25	15	20	10	15

$\frac{1}{4} *$ | 1 | 2 | 1 | $\rightarrow$

(Intermediate) Result

15	20	24	23	16	10
20	25	36	33	21	15
20	44	55	51	35	20
20	29	44	35	22	30
15	21	25	24	25	30
20	21	19	16	14	15

Now apply vertical kernel on intermediate result.

$\frac{1}{4} *$

1
2
1

15	20	24	23	16	10
19	28	38	35	23	15
20	35	48	43	28	21
19	31	42	36	26	28
18	23	28	25	22	21
20	21	19	16	14	15

15	20	25	25	15	10
20	15	50	30	20	15
20	50	55	60	30	20
20	15	65	30	15	30
15	20	30	20	25	30
20	25	15	20	10	15

$\frac{1}{16} *$

1	2	1
2	4	2
1	2	1

48

Calculation for pixel value 55:

$$\frac{1}{16}(1\times15 + 50\times2 + 30\times1 + 50\times2 + 55\times4 + 60\times2 + 1\times15 + 65\times2 + 30\times1)$$

$$= 47.5 \approx 48$$

A5) Linearity and Seperability:

Answer) In 2-D, we apply derivatives of Gaussian in x and y:

Partial Derivatives:

$$\left[\frac{\partial}{\partial x} \circ G(x, y; \sigma)\right] \circ f(x, y) \quad \& \quad \left[\frac{\partial}{\partial y} \circ G(x, y; \sigma)\right] \circ f(x, y).$$

There are two options in which we can proceed for this calculation. For Eg. let us consider x-partial derivative. It can be done in following two ways:

① $$\left[\frac{\partial}{\partial x} \circ G(x, y; \sigma)\right] \circ f(x, y)$$

or

$$\frac{\partial}{\partial x} \circ \left[G(x, y; \sigma) \circ f(x, y)\right]$$

We should go with option 2, i.e first apply guassian filter and then find the derivative. Because, calculation of derivative is an expensive step. It would be more efficient rather than again and again Calculating derivative of guassian filter and apply it to the image pixel values.

A6) Canny - Non - maximum Suppression

Answer) Canny edge detection is used for edge-detection, it uses multi-stage algorithm to detect wide range of edges in images.

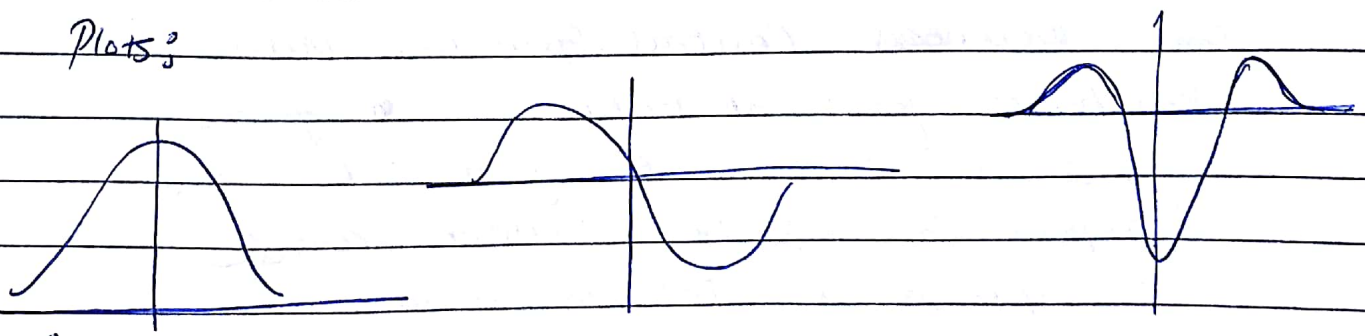
Guassian function in 1-D:

$$G(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-x^2/2\sigma^2}$$

1st Derivative $\Rightarrow$ $G'(x) = \frac{-x}{\sqrt{2\pi}\sigma^3} e^{-x^2/2\sigma^2}$

2nd derivative $\Rightarrow$ $G''(x) = \frac{-1}{\sqrt{2\pi}\sigma^3} e^{-x^2/2\sigma^2} \left[1 - \frac{x^2}{\sigma^2}\right]$

Plots:

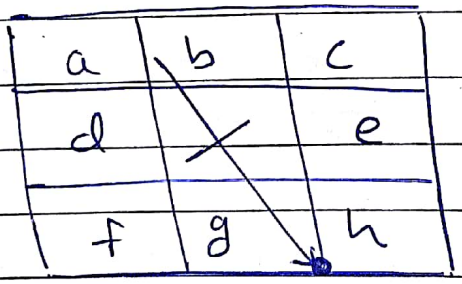


Guassian | 1st Derivative | 2nd Derivative.

Canny works in four Stages : ① Image Smoothing, ② Differentiation and

③ Non-maximum Suppression.

In above two steps, we can find the rate of intensity change at each point in the image, edges must be placed at the point of maxima, or rather non-maxima must be Suppressed. A local maximum occurs at a peak in the gradient function, or alternatively where the derivative of the gradient is set to zero. We Suppress non-maximum perpendicular to edge direction, Since we expect Continuity of edge Strength along an extended contour.



Non-Maximum Suppression.

From Central gradient value interpolate gradient value at ● from gradient value at e, g and h. We repeat in opposite direction and Suppress if non-maximum.