Assignment 2

1) a) $$SNR = \frac{E_s}{E_n} = \frac{\sigma_s^2}{\sigma_n^2} = \frac{\frac{1}{n}\sum_{ij}\left(I(i,j) - \bar{I}\right)}{\sigma_n^2}$$

where σ_s^2 = variance of signal
σ_n^2 = variance of noise

σ_n^2 can be calculated by the variance for multiple frames of a static scene or by the variance in a uniform image region

b) Gaussian noise has a probability density function equal to that of the normal distribution.

Impulsive noise is caused by sharp disturbances in the image signal.

Median filter is used to handle impulsive noise

c) Image =

2	2	2	2
2	2	2	2
2	2	2	2
2	2	2	2

Filter =

1	1	1
1	1	1
1	1	1

Image after applying filter =

8	8	8	8
8	18	18	18
8	18	18	18
8	18	18	18

d) The filter can be split into 2 smaller filters A and B, whose result will be the same as the larger filter A * B and the image as G.
The resultant is A * B * G.

e d) Three different ways to handle boundaries during convolution:

(i) Zero padding : Assuming all cells that are ~~outside~~ outside of the image to be filled with 0s.

(ii) Mirror / Replicate : Filling the cells outside of the image with the same values as the one adjacent to it (which is part of the image)

(iii) Ignore : Ignoring the cells at the ~~boarder~~ boundary during convolution.

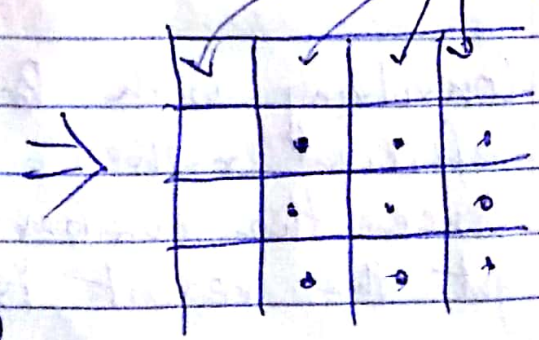
0	0	0	0
0	5	6	7
0	8	9	10
0	11	12	13

(i)

5	5	6	7
5	5	6	7
8	8	9	10
11	11	12	13

(ii)

5	6	7
8	9	10
11	12	13



(iii)

f) Basic smoothing filter

$$\frac{1}{9} \times \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

The sum of all the entries will be 1. The sum is selected this way so that the output has the same intensity as that of the input.

g) $$I_G = I * G = \sum_i \sum_j I(i,j) e^{-\frac{i^2+j^2}{2\sigma^2}}$$

$$= \sum_i e^{\frac{-i^2}{2\sigma^2}} \sum_j I(i,j) e^{\frac{-j^2}{2\sigma^2}}$$

$$= (I * G_y) * G_x = I * G_x * G_y$$

where I is the image and G_x and G_y are the 1D Gaussians in the direction of x and y respectively.

Convolving with two 1D filter is more efficient rather than using one 2D filter since the number operations required to get the result is less.

No. It is only possible for the filters are seperable.

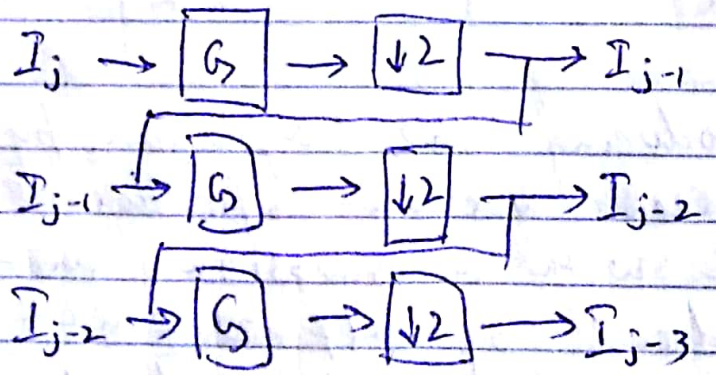
h) We know that $\sigma \leq m/5$

If $\sigma = 2$, the size of the filter should be more than or equal to 10.

i) Gaussian image pyramid is formed by creating multiple versions of a single image, and each version having a dimension half of the previous version.

If the original image has dimensions $M \times M$ the next versions will have dimensions $\frac{M}{2} \times \frac{M}{2}$ $M/4 \times M/4$ and so on.

For each version of the image before shrinking the dimensions the image is convolved with a Gaussian as below:



The reason for producing such pyramids is that when we analyze the lower resolution images we use a bigger window and analyze multiple scales instead of a single scale.

Number of computations in a single image = M^2

Number of computations in an image pyramid

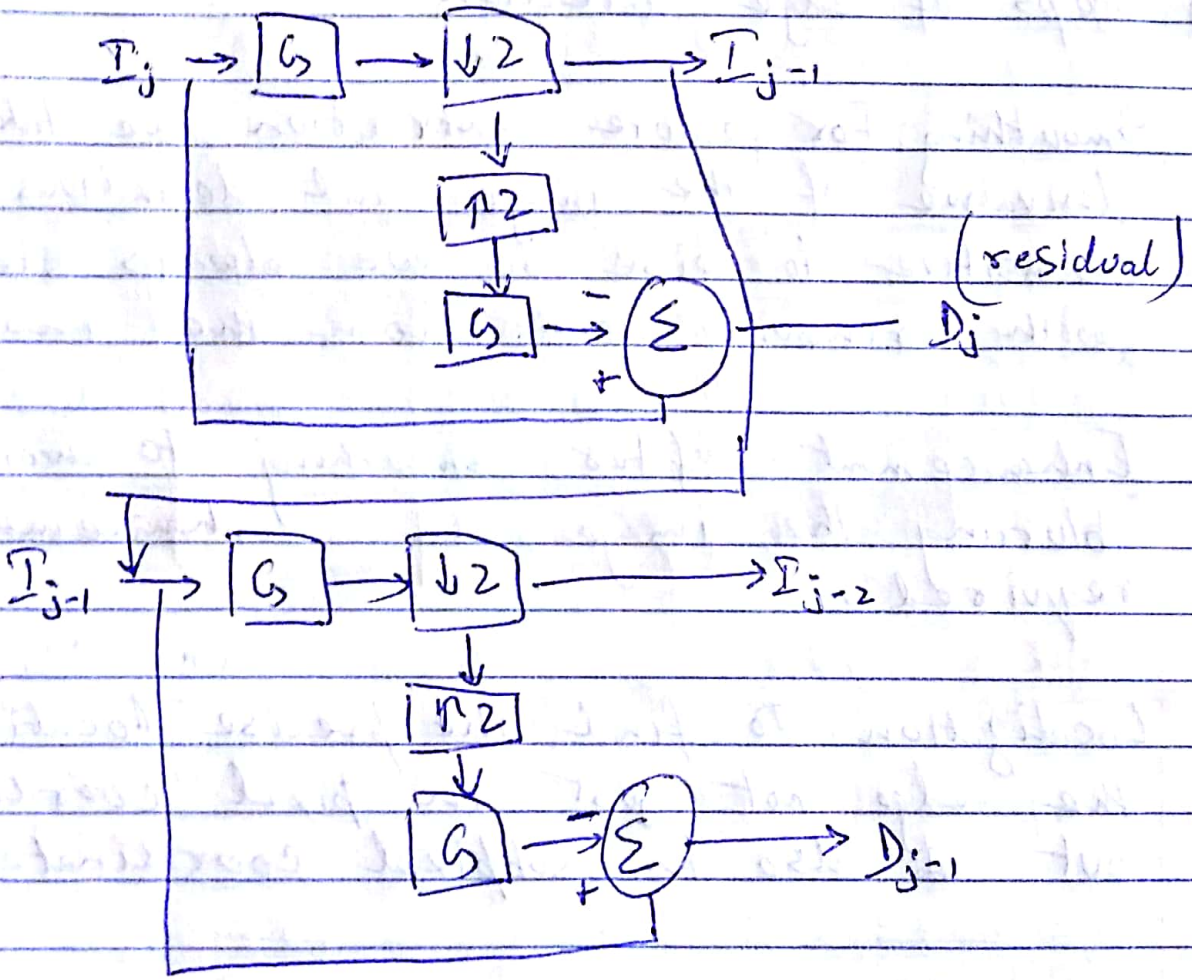
$$= M^2 + \frac{M^2}{4} + \frac{M^2}{16} + \cdots < \frac{4}{3} M^2$$

which is equal to a 30% of additional processing as compared to a single image.

ii) The procedures for ~~creating~~ producing a Laplacian pyramid is similar to that of a Gaussian pyramid but has a few extra processes ~~of~~ before getting the output image.

i) For producing the Laplacian pyramid ~~each~~ each version of the image is convolved with a Gaussian and the dimensions are reduced by half ~~and~~ which is similar to Gaussian pyramid and this becomes the input for the next version of the

image. This image is then brought back to its original resolution and convolved with a Gaussian. The difference between this image and the input image for this level gives the output image of that level which is the residual.



The ~~xx~~ pyramid formed using these residuals is called Laplacian pyramid.

Laplacian
~~Residual~~ pyramids are ~~xx~~ used for compressions, since the residuals are small it is easy to compress.

2)

a) It can detect change of image which is helpful in detecting features of the image.

b) Steps of edge detection:

Smoothing: For feature extraction, we take derivative of the image and derivatives are sensitive to noise so we always start with removing noise from the image.

Enhancemant: After smoothing to avoid blurring the image an enhancement is required.

Localization: To find the precise location of the edge not just in pixel coordinates but also in subpixel coordinates.

c) Forward difference filter and central difference filter or Sobel and LOG.

A directional change in the intensity in an image is called an image gradient.

It is used for edge detection, corner detection.

d)

1	1
1	1

Smoothing filter

$*$

-1	1
-1	1

x-direction derivative filter

$=$

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

x direction of Sobel filter

e) If we have $f(x)$, we need to calculate $f'(x)$, which is the derivative we reconstruct it to $f(x) = f[x] * h^{\pi}(x)$ and calculate the derivative as $f[x] * h'(x)$ and then sample. $h'(x) = G_{x'}$

$G_x = G_y =$

$\frac{1}{4.98} * |0.04|0.14|0.32|0.61|0.88|1|0.88|0.61|0.32|0.14|0.04|$

$G_{x'} = G_{y'} =$

|0.055|0.135|0.243|0.303|0.221|0|-0.221|-0.303|-0.243|-0.135|-0.055|

f) The first order derivative will capture a significant rise or fall in pixel values. A threshold can be used to identify that an edge exists somewhere in the vicinity.
The second order derivative captures peaks of value change. This is a reliable way of detecting exactly where the edge exists.

g)

0.110	0.246	0.271	0.246	0.110
0.246	0	-0.607	0	0.246
0.271	-0.607	-2	-0.607	0.271
0.246	0	-0.607	0	0.246
0.110	0.246	0.271	0.246	0.110

Detect edges using LOG:

1) Compute the LOG

2) Value = 1 when 1 * LOG > 0 or
Value = 0 when 1 * LOG ≤ 0

3) Mask edge at transitions from 0 to 1 and 1 to 0.

h) Canny edge detection uses directional derivative whereas the standard edge detection uses gradient of image.

If $|n| > \tau$ then detect edges as maximum of $\frac{\partial(I * G)}{\partial n}$

where $n = \nabla(I * G)$,
τ = threshold

i) Non maximum supression:
The image is scanned along the direction of the ~~gradi~~ gradient and if the pixels are not part of the local maxima they are set to zero.

Hysteresis Thresholding:

Two thresholds are set τ_H and τ_L where $\tau_H > \tau_L$

1) Make array of pixel $V(i,j) = 0$

2) Scan image in all directions

If $V(i,j) == 0$ and $\nabla(I * G) > \tau_H$
track edge using τ_L and set $V(i,j) = 1$