1(a) Signal to Noise Ratio (SNR) - is defined as energy of signal divided by energy of noise.

$$SNR = \frac{E_s}{E_N} = \frac{\sigma_s^2}{\sigma_N^2}$$

For an image,

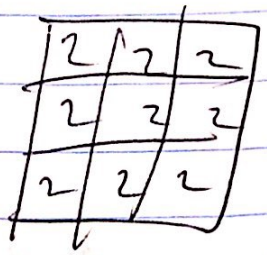
$$SNR = \frac{\text{variance of pixels in a sequence of images}}{\text{variance of in a uniform area of the image.}}$$

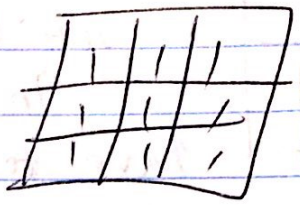
↓ measured in dB

1(b)

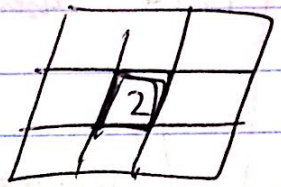
Gaussian noise	Impulsive noise.
→ noise take on gaussian distribution giving peaks on cross-profile of an image	→ The impulsive noise also called pepper-& salt noise gives small white & black spots on image & some pixels are flipped in cross-profile

Impulsive noise uses median filtering for reducing it.

1c) $I =$ 

Filter used will be $= \frac{1}{9}$ 

So middle entry value be 2.



1 d) For taking derivative.

$$\frac{d}{dx}(f * g) = \frac{d}{dx}(f) * g = f * \frac{d}{dx}(g)$$

So what we can do is take derivative of filter & apply this resultant filter to the image to get the convoluted image (new). Both this will have same effect as the taking derivative of $(f * g)$.

Also, or we can do is take filter & apply to derivative of image → to get convolute image.

1(c) To handle boundaries during convolution.
we use:

→ Zero padding - when the boundary value is assigned '0'.

→ Replication - when the boundary value is replicated of the centre value of the convoluted.

→ Here we donot care about the boundary value this is used when filter is very small in comparison to image size

1f) Basic 3x3 Smoothing filter.

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

→ sum = 1.

The sum of entries of this filter is 1 because if the values increases the intensity will increase or if sum is lesser the intensity will decrease.

1(g) 2D convolution can be implemented using Gaussian using two 1D convolution filters as :-

$$G(x,y) = e^{\frac{-x^2+y^2}{2\sigma^2}}$$

$$= e^{\frac{-x^2}{2\sigma^2}} \cdot e^{\frac{-y^2}{2\sigma^2}}$$

$$= G(x) \cdot G(y)$$

So Now if we do convolution of Image I :-

$$I * G(x,y) = G(y) * (I * G(x))$$

This is more efficient way as this will more computationally efficient taking only $2 M \times N \times m$ rather than normal convolution will take $M \times N \times m^2$

1(h) $\sigma = 2$.

accordingly

$$\sigma \leq \frac{m}{5}$$

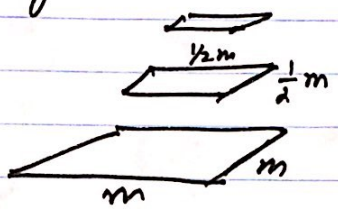
so $m \geq 10$

so size of filter should be 10 or more.

1(i) Gaussian Image pyramids →

- Gaussian Image pyramids are produced by taking the image at level 'j', then performing gaussian smoothing on it, & then down sampling it. This is repeated to get a pyramids of images., where no. of levels = log (size of original image 'm')

$(j) = \log(m)$



- While doing multiple scale analysis, we get a problem of choosing the size of window, so what we do is. we try to choose same size of the window but down scale & smooth the image iteratively.

- total computations we go through = $m^2 + \frac{1}{4}m^2 + \cdots < \frac{4}{3}m^2$
(per pixel)

So to the total we just adding $\frac{1}{3}$ times. approx.

1(j) For Laplacian pyramid, is produced in following step by step :-

a) Take the image at level "j" where $j = \log(m)$

m :- size of the image matrix

b) Then Gaussian smooth it.
c) Then down sample it.
d) Again up-sample it (doubling the size of pixel)
e) smooth it using gaussian
f) to produce image at level $(j-1)$

Repeat above step (a - f) to produce each level of Laplacian pyramid.

This is mostly use for compression of image, as the images produced at each further levels are having errors, which can be compressed easily & also we can reproduce the image again by iterating back.

2. EDGE DETECTION

2(a) Edge Detection - It is used to detect the boundaries of different objects in the image. Different types of discontinuities that we can detect using edge detection are :-

- → discontinuity of Normal
- → illumination discontinuity
- → texture discontinuity
- → depth discontinuity

2(b) Properties of edge detection are:-

→ Correspond to scene element where the feature of the scene is highlighted.

→ Reliable and consistant ~~detection~~ detection

→ Invariant → example, the result should be same if a person's photo is taken from side and front.

2b) STEPS for edge detection include:-

→ Smoothing → It is used because derivatives are ~~noisy~~ sensitive to noise, so we smooth before taking derivative.

$$I * \left(G * \frac{\partial}{\partial x}\right)$$

→ Enhancement - It is used for enhancing the edges, we calculate gradient magnitude.

→ Localization - The location of the edges are estimated. Here when we enhance we get a band of edges (expected area to get a edge) so we do localization to detect exact location.

2c) Two ways to compute the gradient matrix are:-

→ Forward Difference [consider forward neighbour]

→ Central difference [take diff b/w current & previous neighbour]

(i) Forward difference :-

$$fx = \frac{f(x,y)}{f(x)} = \lim_{x \to 0} \frac{I(x+h, y) - I(x)}{h}$$

(put h=1 $= I(x+1, y) - I(x, y) \rightarrow$ in x

$I(x, y+1) - I(x, y) \rightarrow$ in y

$I_x =$ | -1 | 1 |

↓

-1	1
-1	1

$I_y =$ | -1 |
|---|
| 1 |

→

-1	1
-1	1

→ convoluting the image with these matrix will also give gradient.

(ii) Central difference:-

$I_x = I(x+1, y) - I(x-1, y)$ →

-1	0	1
-1	0	1
-1	0	1

$I_y = I(x, y+1) - I(x, y-1)$ →

-1	-1	-1
0	0	0
1	1	1

2(a) During edge detection, we need to smoom first before doing / taking derivative so,

$$= I * \left[G * \frac{\partial}{\partial x} \right]$$

so as we know, Gaussian Filter =

1	1
1	1

Derivative filter =

-1	1
-1	1

so

1	1
1	1

$*$

-1	1
-1	1

$=$

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

→ for x derivative

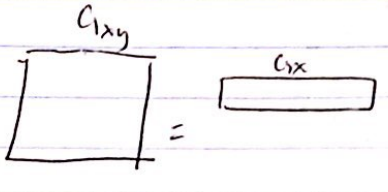
this filter is called sobel filter, which is applied on image

similary for 'y' derivative;

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

So we also observe that, takes derivative in one direction & also doing smoothing in other direction.

2(e) In, more accurate filter.,
we try approximate $h(n)$ with gaussian $G(x)$.
& the derivative can be taken for $G(x)$ (∵ it is continuous)
$f[n] * G'[n]$ → to get edges.
↳ more accurate derivate.



$I_x = I * G_y * G'_x$ → along rows (derivative of G)

$I_y = I * G_x * G'_y$

where ⇒ $G_x = e^{\frac{-x^2}{2\sigma^2}}$, $G(y) = e^{\frac{-y^2}{2\sigma^2}}$

$G'x = \frac{-x}{\sigma^2} e^{\frac{-x^2}{\sigma^2}}$, $G'y = \frac{-y}{\sigma^2} e^{\frac{-y^2}{\sigma^2}}$

So for a filter -
we know $\sigma \leq \frac{m}{5}$

$m \geq 10$ [for $\sigma = 2$].

so
filter size will be 10.

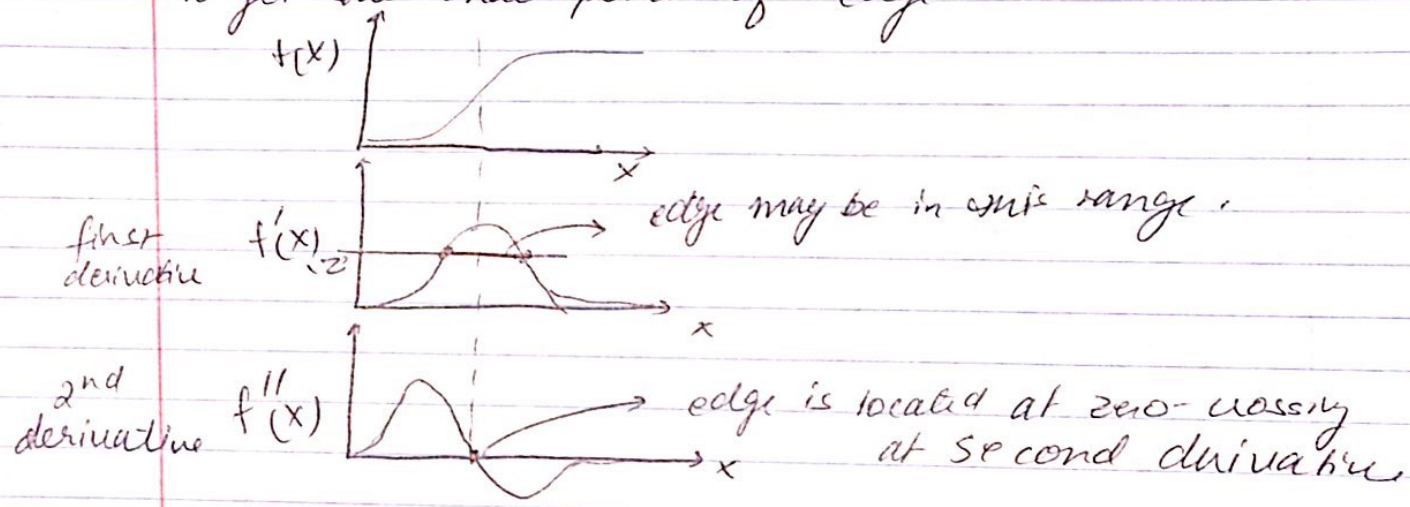
$I_x = I * e^{\frac{-x^2}{8}} * \frac{-x}{4} e^{\frac{-x^2}{4}}$

$I_y = I * e^{\frac{-x^2}{8}} * \frac{-y}{4} e^{-y^2/4}$

2(f) Localization is used to detect edge's most accurate position,

So we take a derivative and see if the magnitude is above some threshold, 'Z', we have a edge in that range. (so we get a thick edge)

So, now, we look at 2nd derivative to get the exact point of edge.



2g) LOG filter → ΔG

ΔG Laplacian Gaussian filter is = Δ (I * G)

= I * Δ(G)

where $G = e^{-r^2/2\sigma^2}$ where $r = \sqrt{x^2+y^2}$

$= \Delta G \frac{r^2-2\sigma^2}{\sigma^4} e^{\frac{-r^2}{2\sigma^2}}$

For $\sigma = 1$

$\Delta G = (r^2-2) \cdot e^{-r^2/2}$.

Edge detection with LOG:-

→ Compute LOG convolve with LOG.

→ Set a threshold as $= \begin{cases} 1 & \text{when } I * LOG > 0 \\ 0 & \text{where } I * LOG \leq 0 \end{cases}$

→ On the resultant image, mark edges at transition (0 → 1 or 1 → 0).

2(h) The main difference between canny edge detection and other is that in canny we take derivative in the direction of the normal of edges.

Whereas, in other we take in the direction of x & y.

if $n = \nabla I = \nabla (I * G)$ where n → normal.

if $|n| => 2$

detect edges as max of $\frac{\partial I}{\partial n}$ or $\frac{\partial}{\partial n}[I * G]$

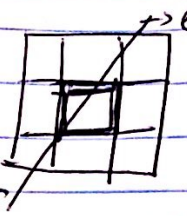
(or zero crossing of $\frac{\partial^2}{\partial n^2}[I * G]$)

$$\frac{\partial}{\partial n}(I * G) = n \cdot \nabla (I * G)$$

↳ product gradient vector onto the direction 'n' you

~~2(b)~~

2(i) Non-Maximum Suspension.



Let 'e' be the gradient computed. So in non-maximum suspension we have to find any box in the, where the gradient is maximum, so that we take box other boxes & supress it.

Hysteresis threshold

One of the condition for edge-detection in canny is :-

$$|\nabla I| > Z$$

where if

→ Z is high value - we will miss some of important edges.

→ Z is low value → we will wrongly detect edges.

so we select 2 value of Z, (Z_H & Z_L)

where

$$Z_H > Z_L$$

Till we get $|\Delta I| > Z_H$, we will be treaty Z_L.