(36)

Scheme	Academic Year
18	2021-22

NATIONAL EDUCATION SOCIETY (R.)

Jawaharlal Nehru New College of Engineering

Savalanga Road, Navule, Shivamogga -577204

(Approved by AICTE, New Delhi, Recognized by Govt. of Karnataka and Affiliated to VTU, Belagavi)

CONTINUOUS INTERNAL EVALUATION BOOK

Details to be filled by Student:

NAME OF THE STUDENT:	INDHUSHREE L. S			USN:	4JN20EC036		
NAME OF THE PROGRAM :	B.E (ECE)						
SEMESTER:	4th	SECTION:	A	COURSE CODE:	18EC46	COURSE NAME:	Microcontroller

Instruction to Students:

- Do not borrow any instruments/calculators/pen/eraser/sharpener etc. from anybody in the examination hall
- Do not bring any programmable calculators / Cell Phone/Smart Watches to examination hall
- Do not indulge in any malpractice/violate instructions given by the room invigilator
- In case of any violation disciplinary action will be initiated

DETAILS OF CONTINUOUS INTERNAL EVALUATION(CIE) MARKS

TEST	DATE	Q1			Q2			Q3			Q4			Max. Marks	Max. Obtained
		a	b	c	a	b	c	a	b	c	a	b	c		
I	11-7-22	7	8	10				7	8	10				50	50
II		7	8	10				7	8	10					50
III		5	8	10							7	8	10		48
Average/ Total Test Marks														30	30
Assignments														10	10
Activity															
Group Discussion / Seminar / Quiz															
Lab Component															
Final CIE Marks														40	40

Final CIE Marks (in Words)	Signature of the Student
four zero.	Indhushree L. S.

Signature of Faculty

Signature of the HOD

Professor & Head
Electronics & Communication Engg
JNN College of Engineering
SHIVAMOGGA-577 204

IA - II.

(1a) PUSH instruction:

* The push instructions loads the content of address present next to it to the stack register of c.

* Push instruction is with respect to stack register only

* The SP which holds the address of last used location of the stack will increment first and then pushes the value to stack.

* The stack pointer contain 07 by default

i) After PUSH 2, (say R2 = FFH)

Address	
0B	
0A	
09	
08	

SP = 07

0B	
0A	
09	
08	FFH.

SP = 08

ii) After PUSH 6 (say R6 = EEH).

0B	
0A	
09	EEH
08	FFH

SP = 09 H.

POP Instruction : The pop instruction will copy the content of last used location of stack to the address present next to it.

* POP instruction works with respect to the stack

* After POP instruction the SP content will be decremented by one.

Let say the stack have

0F	
0E	A0
0D	E1
0C	3F

SP = 0E

i) After POP 2

0F	
0E	
0D	E1
0C	3F

SP = 0D & R2 = A0 H

ii) After POP 7

0F	
0E	
0D	
0C	3F

SP = 0C & R1 = E1.

ACALL Instruction:

* ACALL is Absolute range call where we can call a subroutine which is in the range of 13 bit add
* After calling instruction, the address the next instruction is stored in the stack by the sta processor
* The Label or the subroutine present after ACALL instruction is then executed.

RET instruction:

* RET means Return

* The RET instruction always present at the end of an subroutine.

* After RET instruction the address which is stored by processor in the stack helps CPU to resume after returning from the subroutine i.e., the instruction just below call. instruction is resumed.

(1b) ALP to Convert 8 bit binary number to BCD.

Algorithm: * The no to be converted is taken to the accumulato

* The no is devided by 64, the quotient obtained is the higher byte of the required BCD

* The remainder obtained is devided by 10 & the quotient is swapped and added with the remainder which will be the lower Byte of BCD.

* The higher byte of BCD is stored in one location & the lower byte also

ALP: Let 30H location has the no to be converted
31H location will have the higher byte of BCD
32H location will have the lower byte of BCD

ALP :

~~MOV A, 30H~~

MOV A, 30H ; Content of 30H location is loaded to A
MOV B, #64H ; 64H value is moved to B
DIV AB ; The no to be converted is devided by 64H
MOV 31H, A ; Quotient is higher byte of BCD stored in 31H loc
MOV A, B ; Remainder is taken to Accumulator
MOV B, #0AH ; 0AH is taken to B reg
DIV AB ; Remainder value is devided by 10
SWAP A ; Quotient is swapped
ADD A, B ; And added with the remainder which will be lower
MOV 32H, A ; lower byte of BCD is stored in 32H loc
END ;

Result : -

i) Before execution

30H : FEH
31H : 00H
32H : 00H

After execution

30H : ~~02H~~ FEH
31H : ~~54H~~ 02H
32H : ~~0~~ 54H

Bin/Hex → BCD
FE → 254.

(ii) Before

30H : FAH
31H : 00H
32H : 00H

After

30H : FAH
31H : 02H
32H : 50H.

FAH → 250H

(1c) $f = \frac{11.0592 \text{ MHz}}{12}$

$T = \frac{12}{f} = 1.085 \mu s.$

$T = 100 \mu s$

$\frac{100\mu}{1.085\mu} = 92.$

$256 - 92 = 164 = A4H.$

The value of TH1 is A4 H

ALP:

```
        MOV TMOD, #20H     ; Timer 1 in mode 2 (8 bit Auto reload)
        MOV TH1, #A4H      ; A4H initial value is loaded.
        ACALL DELAY        ; Calling the delay subroutine
        SJMP NEXT
DELAY : SETB TR1           ; Sets the timer register, to start timer
AGAIN : JNB TF1, AGAIN     ; Checks the Timer 1 flag if it is
        CLR TR1            ; set. Clears the Timer
        CLR TF1            ; register & clears the Timer
NEXT:                        flag & it reloads the
        END.                 content of TH1 to TL1
                             again.
```

0

(3a) i) MOV SP, #70H

Stack

74	
73	
72	
71	

SP = 70

ii) MOV R5, #30H

74	
73	
72	
71	

SP = 70.

R5 = 30H.

iii) MOV A, #44H — A = 44H | R5 = 30H.

iv) ADD A, R5 — A = 44H + 30H = 74H | R5 = 30H.
A = 74H.

v) MOV R4, A — R4 = 74H. | R5 = 30H

vi) PUSH 4

74	
73	
72	
71	74

SP = 71

pushes R4 content to stack (74H).

vii) PUSH 5

74	
73	
72	30H
71	74H.

R5 = 30H | R4 = 74H

SP = 72

viii) POP 4

74	
73	
72	
71	74H

R5 = 30H | R4 = 30H.

SP = 71

(3b) Programming Timer 0 in mode 1

Step I: Since the Timer operation ~~mode~~ mode is to be set ~~it~~ is set by using TMOD

TMOD is set as 00000010 b i.e., 01H to get Timer as Timer 0 in mode 1 (16 bit timer)

~~Step 2.~~

TMOD :

G	C/T	M1	M0	G	C/T	M1	M0
0	0	0	0	0	0	0	1
T1				T0			

Timer 0 is activated by giving 0 to C/T bit and it is an software stop & start timer since gate is 0 & the operation mode is 1 i.e., 16 bit mode.

Step 2:

* The initial value is loaded to the THO and TLO of the timer 0.

* Since the timer is 16 bit & the THO & TLO are 8 bit the higher byte of the initial value will be loaded to THO & the lower byte is loaded to TLO.

e.g :-

FF	F2
THO	TLO

initial value is FFF2.

Step 3:

* After loading the ~~Timer regist~~ TH & TL with the initial value the timer has to be started by making Timer register of timer 0 i.e., TRO ~~to~~ set.

* We can use SETB instruction to make TRO register to set & by making it to start timer.

if the TRO is set the timer starts generating the pulse or delay using the internal frequency and rolls upto FFFF h.

Step 4:

* When the timer rolls upto to FFFF h & rolls over to 0000 H the timer flag of timer 0 i.e., TFO will be set.

* One has to check the bit of TFO content & if that is set stop the timer by making TRO as 0 we can use CLR TRO to stop the timer

* And clear the content of the TFO to start another delay.

FFF2 → FFF3 → FFF4 → FFF5 → FFF6 → FFF7 → FFF8 →
FFF9 → FFFA → FFFB → FFFC → FFFD → FFFE → FFFF
FFFF → 0000, when this happens

1

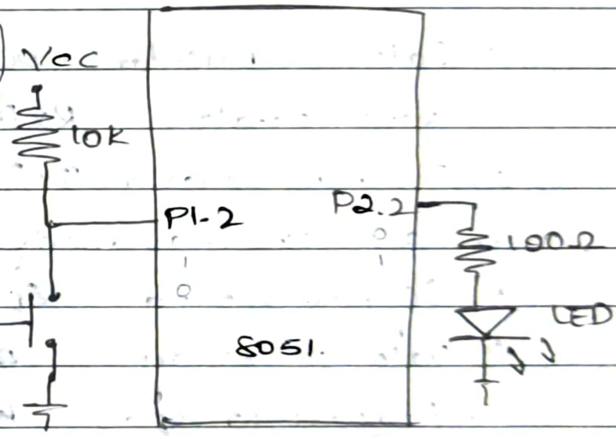
TF0

Step 5 : The user has to reload the content of TH0 and TL0 to start another delay since, the mode is 1.

(3c).

Switch = P1.2
LED = P2.2

(if SW is ON P1.2 is 0, LED is ON)
(SW is OFF P1.2 is 1 LED is OFF)



ALP.

```
        SETB P1.2      ; making P1.2 as i/p pin
        CLR  P2.2      ; making P2.2 as o/p pin
Again:  JNB  P1.2, Ledon  ; if P1.2 is 1
        CLR  P2.2      ; clears P2.2
        SJMP Again
Ledon:  SETB P2.2      ; if P1.2 is 0, sets P2.2
        SJMP Again
        END
```

10

50/50 16-8-22

Roll No.
48

Academic Year
2021 - 22

NATIONAL EDUCATION SOCIETY (R.)

Jawaharlal Nehru National College of Engineering

Savalanga Road, Navule, Shivamogga -577204

(Approved by AICTE, New Delhi, Certified by UGC 2f & 12B, Accredited by NAAC -'B', Recognized by Govt. of Karnataka and Affiliated to VTU, Belagavi)

CONTINUOUS INTERNAL EVALUATION BOOK

(CBCS 2017-18 Scheme Onwards)

DETAILS TO BE FILLED BY STUDENT:

NAME OF THE STUDENT:	Vidyashree. R.		
USN:	4JN18EC112		
SEMESTER:	VII	SECTION:	B
COURSE CODE:	18EC733	PROGRAMME:	B. E
COURSE NAME:	Digital Image Processing.		

DECLARATION:

- I know that, I am not allowed to borrow any instruments/calculators/pen/eraser/sharpener etc. from anybody in the examanation hall.
- I will not be indulging in any malpractice/violation of instructions given by the room invigilators.
- In case, I violate any of these, I know that disciplinary action will be initiated against me.

Vidyashree. R
Signature of Student

Date:

DETAILS OF CONTINUOUS INTERNAL EVALUATION(CIE) MARKS

	DATE	Q1			Q2			Q3			Q4			Max. Marks	Max. Obtained	Sign of Student	Initials of Faculty
		a	b	c	a	b	c	a	b	c	a	b	c				
I	27/4/21	13	12					9	8	8			50	30	30		
II	4/1/22	12	13					12	13		12	13	50	30	30		
III	5/2/22				13	12		13	12				50	30	30		
(a) Average CIE Marks														30	30		
(b) Assignments/Unit Tests/ Quiz														10	10		
Final CIE Marks : (a) + (b)														40	40		

Final CIE Marks (in Words)
Forty Zero

Signature of Faculty

Signature of the HOD

Professor & Head
Dept. of Electronics & Communication Engg.
JNN College of Engineering
SHIVAMOGGA-577 204

1\. a) **Homomorphic filtering:**

It is a frequeny domain process which is used to increase the appearance of the image by
- gray level range compression
- contrast enhancement.

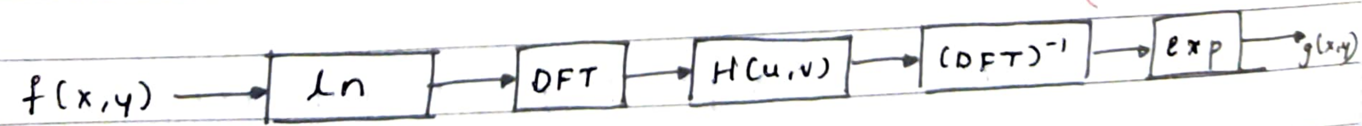
The captured image $f(x,y)$ will be a product of the illumination $(i(x,y))$ & the reflectance $(r(x,y))$.

$$f(x,y) = i(x,y)\, r(x,y) \qquad \text{—— (1)}$$

This cannot be considered the same in frequency domain because Fourier transform of products is not same as the ~~tra~~ product of two transforms.

$$\mathcal{F}[f(x,y)] \neq \mathcal{F}[i(x,y)]\, \mathcal{F}[r(x,y)]$$

These components are separated using the homomorphic filter.



Step (1): For eqn (1), take log on both sides.

$$\ln[f(x,y)] = \ln[i(x,y) \cdot r(x,y)]$$

$$z(x,y) = \ln[i(x,y)] + \ln[r(x,y)]$$

Step ②: Apply Fourier transform on both sides.

$$F[z(x,y)] = F\{\ln[i(x,y)] + \ln[r(x,y)]\}$$

$$Z(u,v) = F\{\ln[i(x,y)]\} + F\{\ln[r(x,y)]\}$$

~~$Z(u,v) = F_i(x,y) + F_r(x,y)$~~

$$Z(u,v) = F_i(u,v) + F_r(u,v)$$

where
$$F_i(u,v) = F\{\ln[i(x,y)]\}$$
$$F_r(u,v) = F\{\ln[r(x,y)]\}$$

Step ③: Multiply with filter function $H(u,v)$ on both sides.

$$S(u,v) = Z(u,v)\,H(u,v)$$
$$= F_i(u,v)\,H(u,v) + F_r(u,v)\,H(u,v)$$

Step ④: Take inverse fourier transform on both sides.

$$F^{-1}\{S(u,v)\} = F^{-1}\{F_i(u,v)\,H(u,v) + F_r(u,v)\,H(u,v)\}$$

$$s(x,y) = F^{-1}\{F_i(u,v)\,H(u,v)\} + F^{-1}\{F_r(u,v)\,H(u,v)\}$$
$$s(x,y) = i'(x,y) + r'(x,y)$$

where
$$i'(x,y) = F^{-1}\{F_i(u,v)\,H(u,v)\}$$
$$r'(x,y) = F^{-1}\{F_r(u,v)\,H(u,v)\}$$

Step 5: Take exponent on both sides.

$$e^{s(x,y)} = e^{i'(x,y)+r'(x,y)}$$
$$= e^{i'(x,y)} \cdot e^{r'(x,y)}$$
$$\therefore \; g(x,y) = i_0(x,y) \cdot r_0(x,y)$$

where $i_0(x,y) = e^{i'(x,y)}$

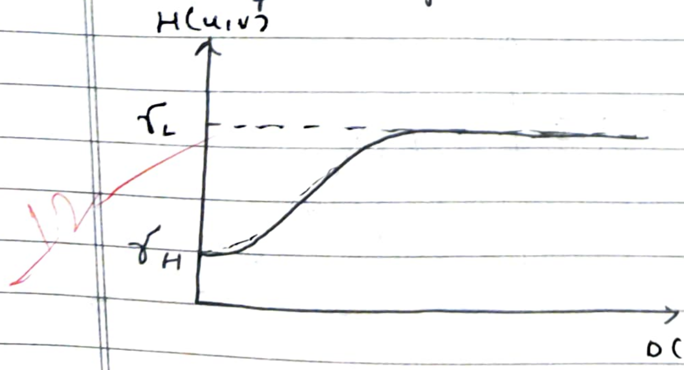
$r_0(x,y) = e^{r'(x,y)}$

Here, $i_0(x,y)$ & $r_0(x,y)$ are the illumination & reflectance components of output image.

Two parameters γ_L & γ_H are considered

- to reduce the contribution of low-frequency (illumination) $\rightarrow \gamma_L < 1$
- to amplify the contribution of high-frequency (reflectance) $\rightarrow \gamma_H > 1$

The illumination is characterized to show slow spatial variations.
The reflectance is characterized to show abrupt transition.



~~$= (\gamma_H - \gamma_L)\left[1 - e^{-c\left[\frac{D(u,v)}{D_0}\right]^2}\right]$~~

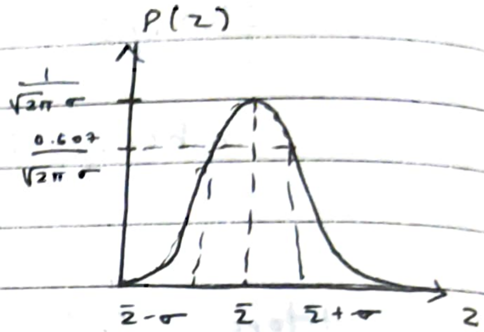
$$H(u,v) = (\gamma_H - \gamma_L)\left[1 - e^{-c\left[\frac{D^2(u,v)}{2D_0^2}\right]}\right] + \gamma_L$$

1. b) **Noise Models:**

i]. **Gaussian Noise:**

The pdf of the Gaussian noise is given by

$$p(z) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(z-\bar{z})^2/2\sigma^2}$$



where z - intensity

$\bar{z}$ - mean of intensities

σ - standard deviation

σ^2 - variance

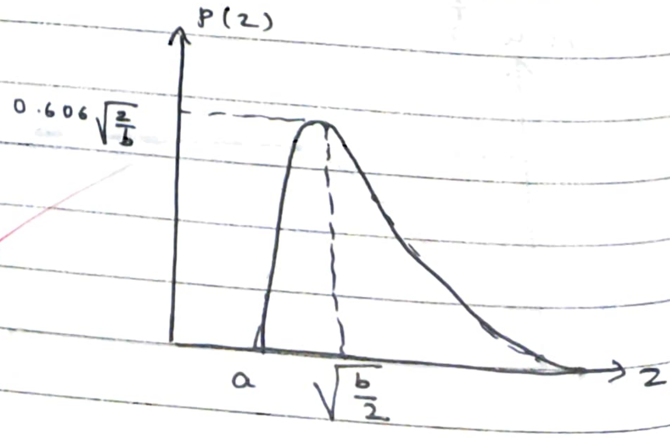
ii] **Rayleigh Noise:**

The pdf of Rayleigh Noise is given by

$$P(z) = \begin{cases} \frac{2}{b}(z-a)e^{-(z-a)^2/b} & \text{if } z \geq a \\ 0 & \text{if } z < a \end{cases}$$

The mean, $\bar{z} = a + \sqrt{\frac{\pi b}{4}}$

Variance, $\sigma^2 = \frac{b(\pi-4)}{4}$



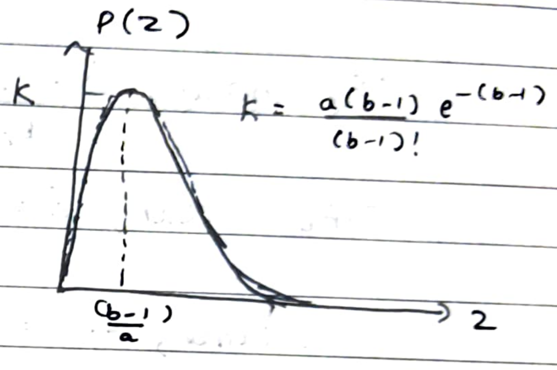
iii] Erlang (Gamma) Noise :

The pdf of Gamma Noise is given by

$$p(z) = \begin{cases} \frac{a^b z^{b-1}}{(b-1)!} e^{-az} & \text{if } z \geq 0 \\ 0 & \text{if } z < 0 \end{cases}$$

The mean, $\bar{z} = \frac{b}{a}$

Variance, $\sigma^2 = \frac{b}{a^2}$



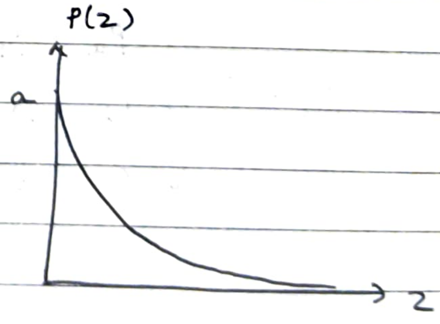
iv] Exponential Noise:

The pdf of exponential noise is given by

$$p(z) \begin{cases} e^{-az} & \text{if } z \geq 0 \\ 0 & \text{if } z < 0 \end{cases}$$

The mean, $\bar{z} = \frac{1}{a}$

Variance, $\sigma^2 = \frac{1}{a^2}$

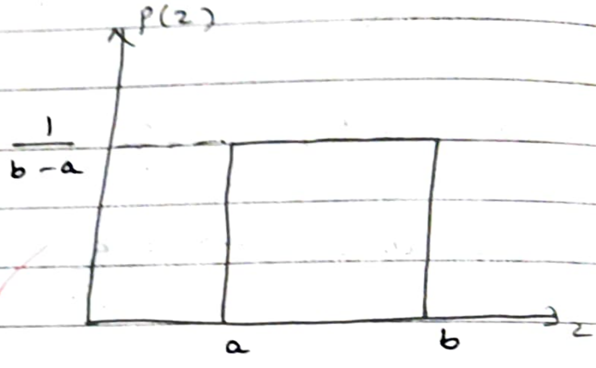


v] Uniform Noise:

The pdf of uniform noise is given by

$$p(z) = \begin{cases} \frac{1}{b-a} & \text{for } b \geq z \geq a \\ 0 & \text{otherwise} \end{cases}$$

~~The mean, $\bar{z} = \frac{1}{b-a}$~~



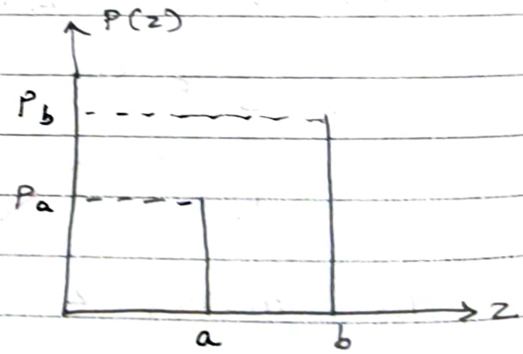
The mean, $\bar{z} = \frac{b+a}{2}$

Variance, $\sigma^2 = \frac{(b-a)^2}{12}$

vi] Salt & Pepper (Impulse) Noise:

The pdf of impulse noise is given by

$$p(z) = \begin{cases} P_a & \text{if } z = a \\ P_b & \text{if } z = b \\ 0 & \text{otherwise} \end{cases}$$



4. a) ## Smoothing images in frequency domain:

Smoothing of images is done using low pass filter.

Low pass filter allows the low frequency components & blocks / attenuates the high frequency components.

$$G(u,v) = H(u,v)\,F(u,v)$$

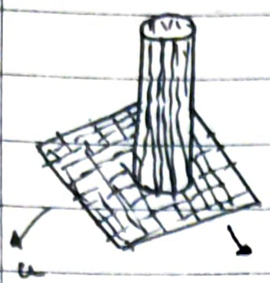
where $G(u,v)$ - F.T of enhanced image
$H(u,v)$ - F.T of filter
$F(u,v)$ - F.T of input image.

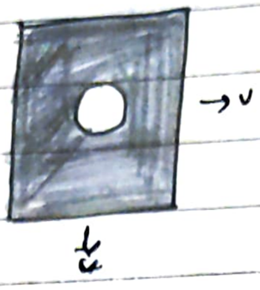
There are 3 smoothing filters in frequency domain
i] Ideal low pass filter [ILPF]
ii] Butterworth lowpass filter [ILPF]
iii] Gaussial lowpass filter [GLPF]

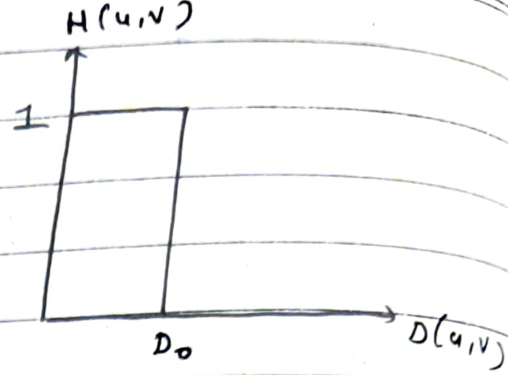
i] Ideal lowpass filter [ILPF]:
It is a filter which allows the components only within the circle of radius D_0 & cuts off " the components outside it.

$$H(u,v) = \begin{cases} 1 & \text{if } D(u,v) \leq D_0 \\ 0 & \text{if } D(u,v) > D_0 \end{cases}$$

where $D(u,v)$ is the distance from origin
D_0 is the cutoff frequency.







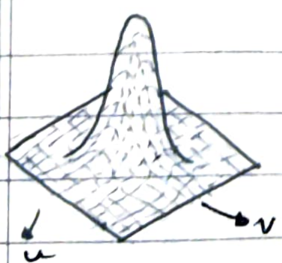
- ILPF reduces the noise but it introduces blurring.
- Ringing is an undesirable effect around the image.
  Ringing artifact is side effect of the ideal lowpass filter.

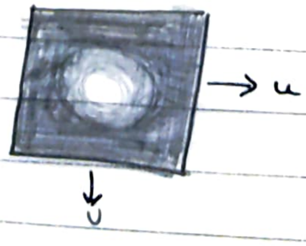
ii] Butterworth Low Pass Filter [BLPF]:

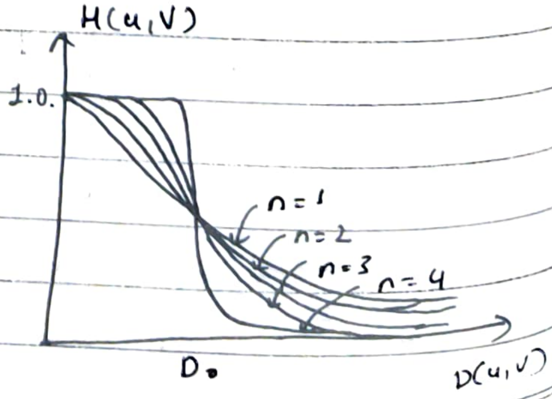
We have

$$H(u,v) = \frac{1}{1 + \left[\frac{D(u,v)}{D_0}\right]^{2n}}$$

where n is the order of filter
$D(u,v)$ is the distance from origin
D_0 is the cutoff frequency.







- In butterworth filter, there is a smooth transition => less ringing artifacts.

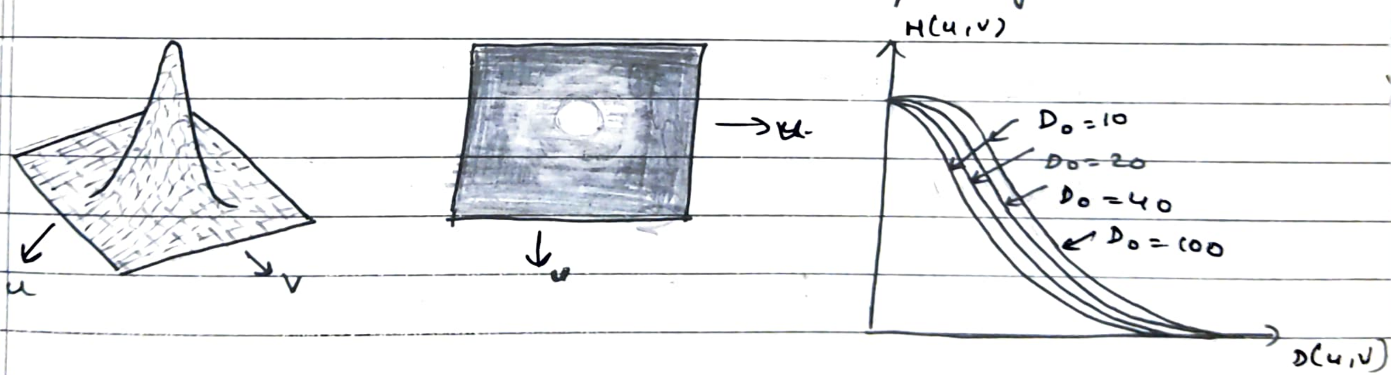
- It reduces noise & introduces blurring.

- For n=1, there is no ringing artifact.
- For n>2, the significance of ringing artifact increases.

iii] Gaussian Low Pass filter (GLPF) :

$$H(u,v) = e^{-D^2(u,v)/2D_0^2}$$

where $D(u,v)$ is the distance from origin
D_0 is the cutoff frequency.



- Gausian low pass filter does not produce ringing artifacts.

- Thus, it is used in medical imaging

4.b) Sharpening - Frequency domain filters:

Sharpening refers to highlighting the detail of the image.

Sharpening can be done using high pass filters.

Highpass filters allows the high-frequency components & attenuates the low-frequency components.

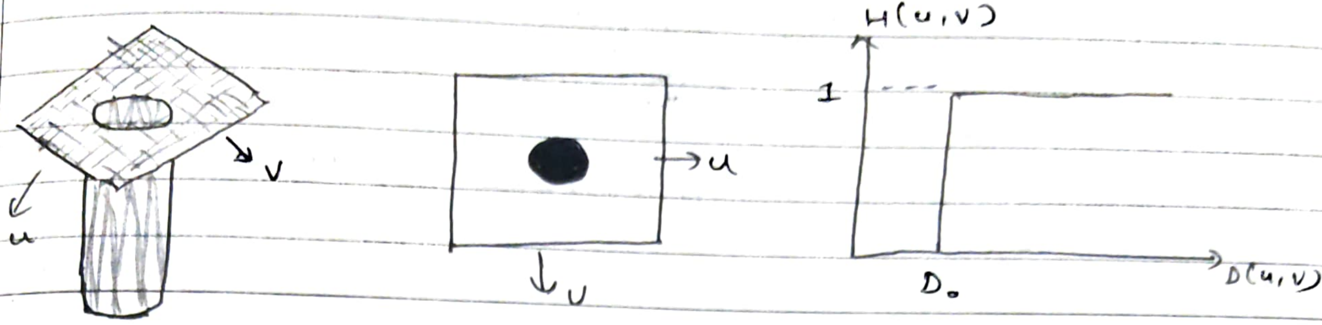
3 types: i] Ideal High pass filter
ii] Butterworth Highpass filter
iii] Gaussian Highpass filter.

i] Ideal High Pass filter[IHPF]:

$$H(u,v) = \begin{cases} 0, & \text{if } D(u,v) \leq D_0 \\ 1, & \text{if } D(u,v) > D_0 \end{cases}$$

where $D(u,v)$ is the distance from origin
D_0 is the cutoff frequency

Ideal high pass filter has ringing artifacts.

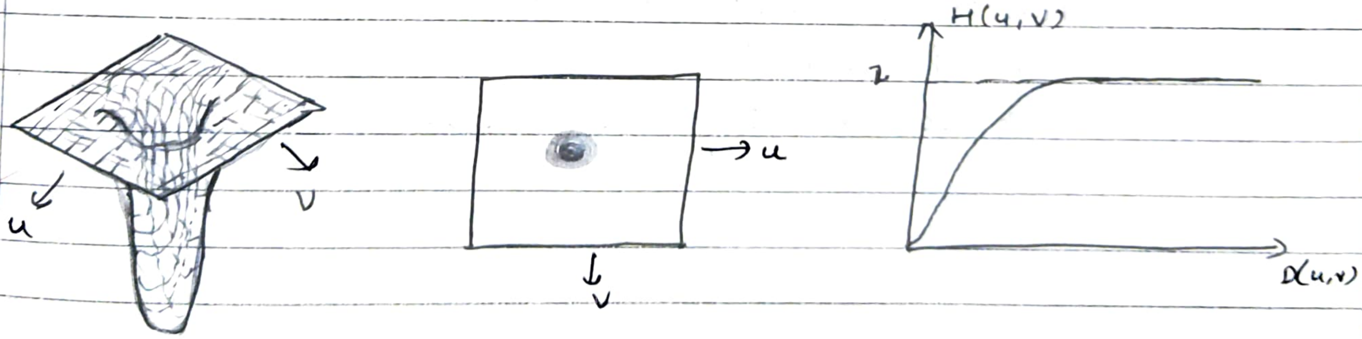


ii] Butterworth High Pass filter:

$$H(u,v) = \frac{1}{1 + \left[\frac{D_0}{D(u,v)}\right]^{2n}}$$

where D_0 - cutoff frequency
$D(u,v)$ is the distance from origin.

It will be same to ideal highpass ~~for~~ small freq
It almost have no ringing effect.



iii] Gaussian Highpass filter:

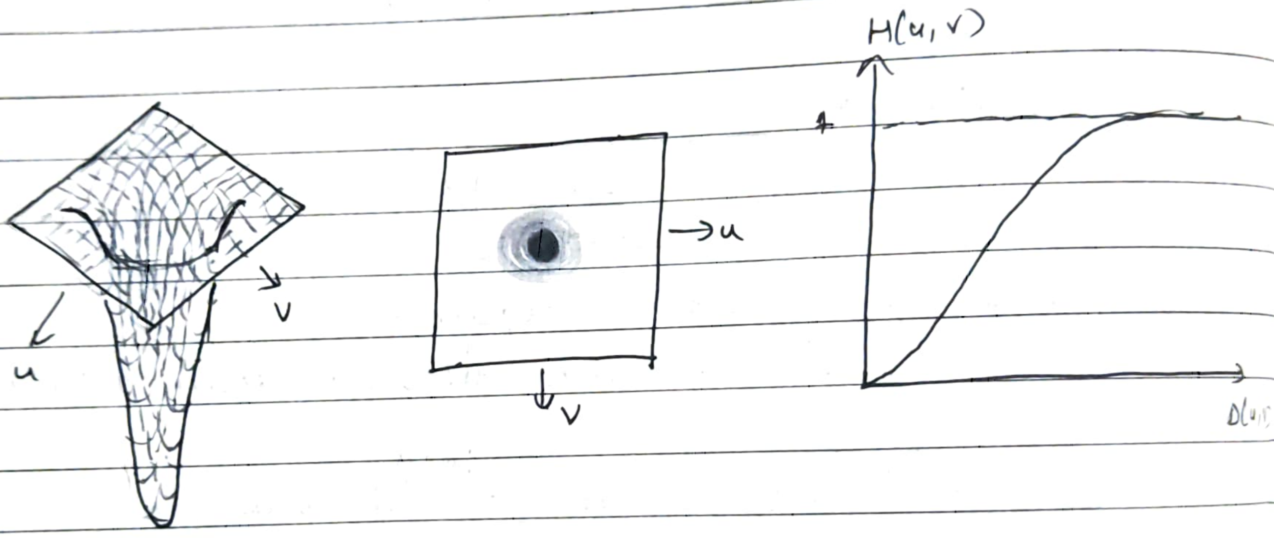
~~$H(u,v) = 1 - e^{-D^2}$~~

$$H(u,v) = 1 - e^{-D^2(u,v)/2D_0^2}$$

where $D(u,v)$ is distance from origin

D_0 - cutoff frequency

The Gaussian Highpass filter has absolutely no ringing artifact.



- Gaussian high pass filter has a smooth transition between stop band & the smooth band.
  → No ringing artifacts.

3. a) **Order - statistics filters:**

They depend on the order of the pixel values in filter.

a) **Median filter:**

It replaces the value by the median of the pixel values in the neighbourhood of the filter.

$$g(x,y) = \text{median} \{ f(x+i, y+i) \mid i \in S(x,y) \}$$

where $S(x,y)$ is a filter function.

0	1	2
9	3	4
6	5	7

→ order = 0, 1, 2, 3, ④, 5, 6, 7, 9.
↓ median

=>

x	x	x
x	4	x
x	x	x

* It reduces the noise by preserving the edges.

b) **Min & Max filter:**

Min filter: Replaces the value by the minimum value in the neighbourhood of filter.

Max filter: Replaces the value by the maximum value in the neighbourhood of the filter.

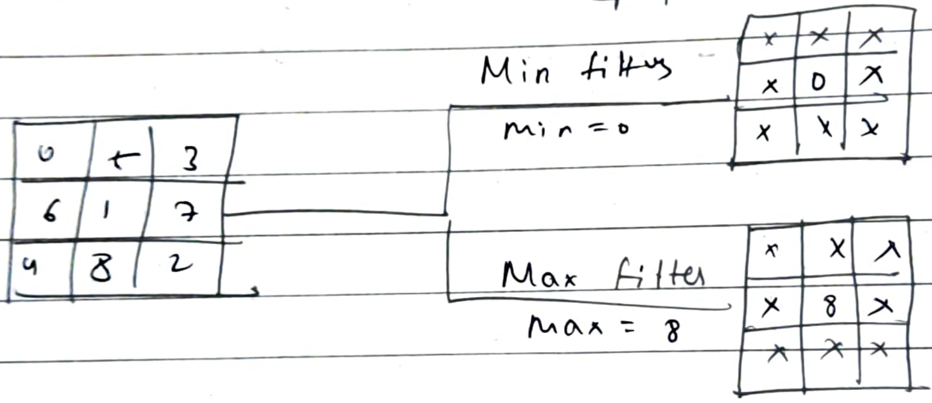
Min filter: $g(x,y) = \min\{f(x+i, y+j) \mid i,j \in S(x,y)\}$

$R = \min\{z_k \mid k = 0, 1, \ldots, 9\}$

Max filter: $g(x,y) = \max\{f(x+i, y+j) \mid i,j \in S(x,y)\}$

$R = \max\{z_k \mid k = 0, 1, \ldots, 9\}$

Min filter reduces salt noise & max filter reduces pepper noise.



3.b] Salt & pepper noise are → 0 & 255

i] $\begin{bmatrix} 24 & 22 & 33 \\ 34 & 255 & 24 \\ 23 & 21 & 32 \end{bmatrix}$ → 21, 22, 23, 24, (24) → median, 32, 33, 34, 255

Median = 24

ii] $\begin{bmatrix} 22 & 33 & 25 \\ 255 & 24 & 0 \\ 21 & 32 & 31 \end{bmatrix}$ → ~~0, 21, 22, 25, 31, 32~~

→ 0, 21, 22, 24, (25), 31, 32, 33, 255

Median = 25

0, 24, 25, 26, (28), 31, 32, 32, 33

iii]

33	25	32
24	0	26
32	31	28

→ 0, 24, 25, 26, (28), 31, 32, 32, 33

Median = 28

iv]

25	32	24
0	26	23
31	28	26

→ 0, 23, 24, 25, (26), 26, 28, 32, 32

Median = 26

⇒

24	22	23	25	32	24
34	24	25	28	26	23
23	21	32	31	28	26

Thus, removing the 255 & 0, the salt & pepper noise are removed.

50/50

60.2.2