Spatial domain operations on images

# Sl. No.	3
Topics	Point processing and neighbourhood processing

Here co-ordinates do not change, but the pixel values will change. There are two types:

- Point processing
- Neighbourhood processing

Point Processing

$$\text{Transform } \; T\{0,1,...,(L-1)\} \rightarrow \{0,1,...,(L-1)\} \; \text{ such that } \; T(r) = s$$

Identity Transformation

$$T(r) = s$$

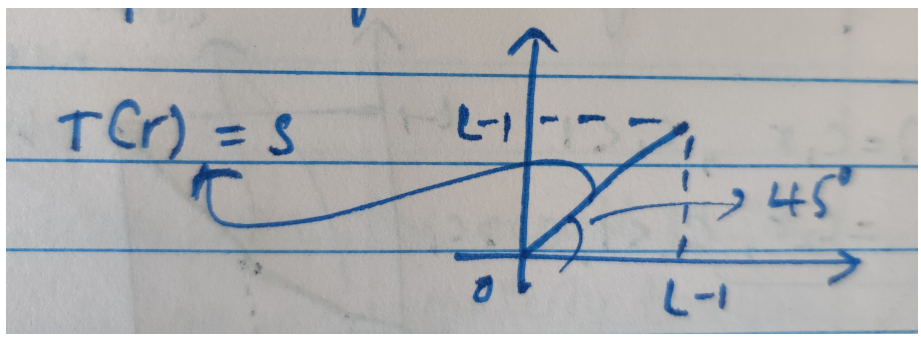
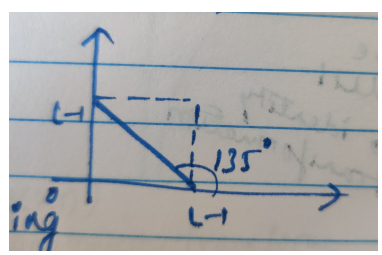


Image Negation / Complementation

$$s = T(r) = (L-1) - r$$



- Used in digital x-ray processing

Thresholding

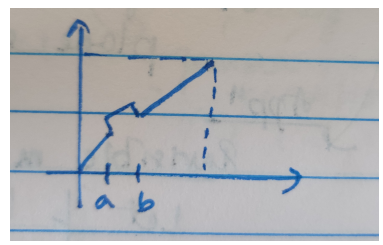
$$
\begin{aligned}
s = T(r) &= 1 \text{ if } r \geq T_o \\
&= 0 \text{ otherwise} \\
\text{where } T_o &\text{ is some threshold value}
\end{aligned}
$$

- Edging (To get edges of particular object in an image)
- To get binary image from gray

Contrast Enhancement

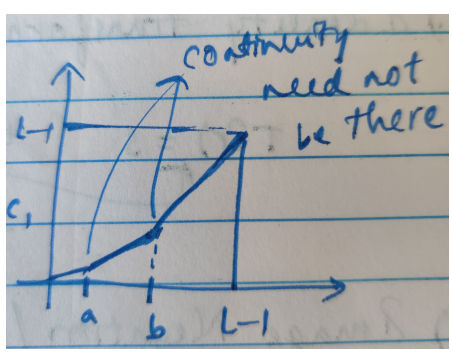
> ✍🏻 **Contrast:** How background is different from foreground or we can consider that as a line of difference between two objects or two environments.

$$
\begin{aligned}
s = T(r) &= cr \text{ if } a \leq r \leq b \\
&= r \text{ otherwise}
\end{aligned}
$$



Contrast Stretching

$$
\begin{aligned}
s = T(r) &= c_1 r, \text{ where } c_1 < 1 \ \& \ r < a \\
&= c_2 r, \text{ where } c_2 < 1, c_2 > c_1 \ \& \ a \leq r \leq b \\
&= c_3 r, \text{ where } c_3 > 1 \ \& \ r > b
\end{aligned}
$$



- c_2 not necessary to be parallel to identity transformation.

Bit plane slicing

Instead of visible `water marking` we can use `invisible water marking` using binary, consider each pixel value as a eight bit binary value i.e., $p = (b_7b_6b_5b_4b_3b_2b_1)b_o$ where $(b_7b_6b_5b_4b_3b_2b_1)$ is original bit values of original pixel image and b_o[least significant bit] is binary value of logo. Since only one bit plane is varied as the the binary pixel values of logo in the original image pixel values it is known as `bit plane slicing`.

Let $f(x, y)$ be an 8-bit image, i.e.,

$$f(x, y) = (b_7(x, y), b_6(x, y), ...b_0(x, y))$$

$$f_i(x, y) = b_i(x, y) * 2^i, \text{where } f_i \text{ is called bit plane slicing of } f$$

Application:

Invisible water marking, Let f be the image and h be the water mark logo (in binary),

$$f_w(x, y) = \sum_{i=1}^{7} f_i(x, y) + h(x, y)$$

Logarithmic Transformation

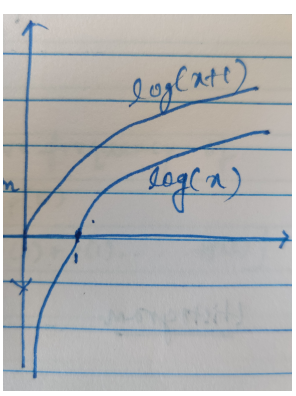
1. Apply logarithmic transformation,

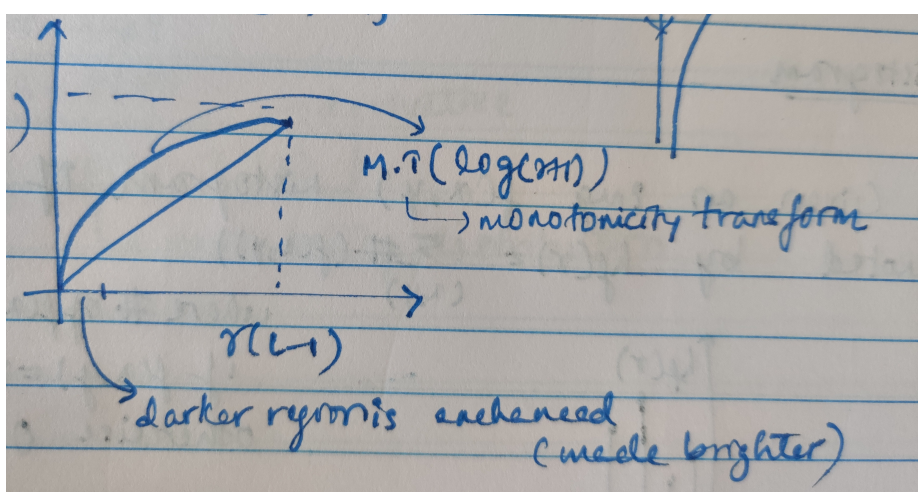
$$s = T(r) = \log(r + 1)$$

2. Transform the result s using monotonicity transform to get the range $[0, L - 1]$

Application:

- Darker region is enhanced or made brighter through logarithmic transformation, so it can be used to get some more clearer representation of darker environment images.





Power Law / Gamma Transformation

$$s = T(r) = c \cdot r^{\gamma} \text{ where } c \text{ and } \gamma \text{ are constants}$$

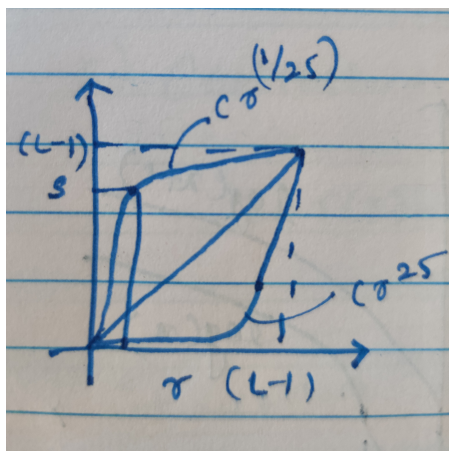
- Voltage to intensity transformation ($r \rightarrow cr^{\gamma}$) which is from range $[0, L-1] \rightarrow [0, k]$ where $k << L-1$
- Normally in standard transformations γ value is used as $\frac{1}{25}$. This γ power transformation is also known as `gamma correction`.

Steps:

1. Apply gamma transformation,

$$s = T(r) = c \cdot r^{\gamma}$$

2. Apply monotonicity transformation on the result of step 1 (s).



Curved parts are monotonically transformed values of gamma transformations

Histogram equalisation transformation

Histogram

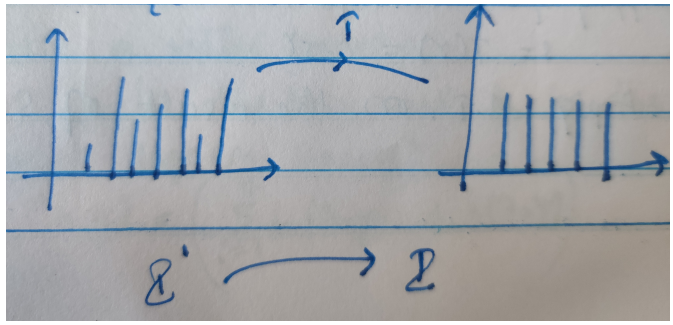
Given an image $f(x, y)$ histogram of f is denoted by,

$$\begin{aligned} h_f(r) &= \sum_{(x,y)} \#(f(x,y)), \\ \text{where } \#(f(x,y)) &= 1 \text{ if } f(x,y) = r \\ &= 0 \text{ otherwise} \end{aligned}$$

Normalisation:

$$h_n(r) = \frac{h_f(r)}{\text{The total number of pixels in the image}}$$

Histogram equalisation:



Histogram equalisation transformation

Objective: manipulating contrast and brightness

$$s = T(r) = \left[\int_0^r p(r)dr\right](L-1)$$

Where, $\int_o^r p(k)dk$ is cumulative frequency of $p(k)$.

Theorem:

Let

$$s = T(r) = \left[\int_0^r p(r)dr\right](L-1)$$

Prove that $p(s) = constant$.

Proof:

We know that, if

$$s = T(r)$$

Then,

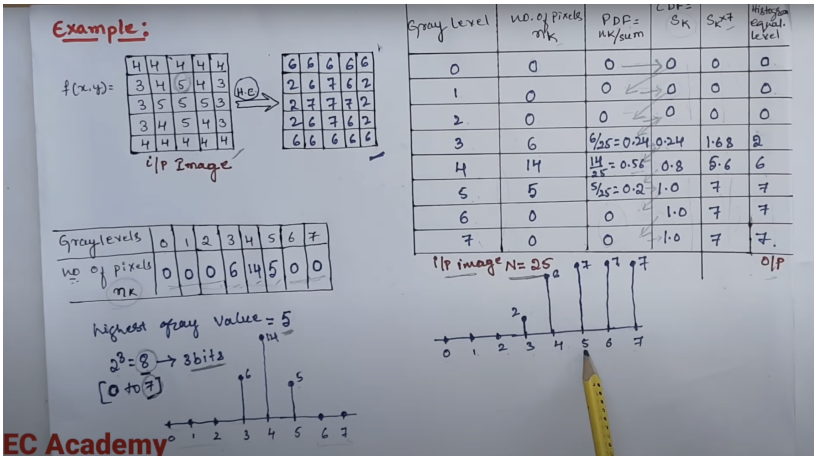
$$p(r) = T'(r) \cdot p(s)$$

So by differentiating,

$$T'(r) = (L-1) \cdot p(r)$$

Now,

$$p(r) = (L-1) \cdot p(r) \cdot p(s)$$
$$\text{Therefore } p(s) = \frac{1}{L-1} \text{ which is a constant.}$$



Problem: Given the probability distribution of image as follows, find the histogram equalisation

Note: $p(s)$ is constant only in continuous domain but not in discrete since some values are lost cause of quantisation

Histogram matching / specification

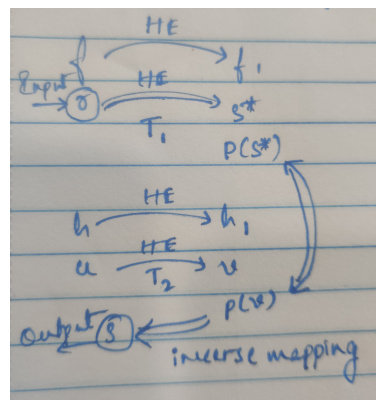
Consider, input image: f which is a bad quality image and reference image: h which is a good quality image.

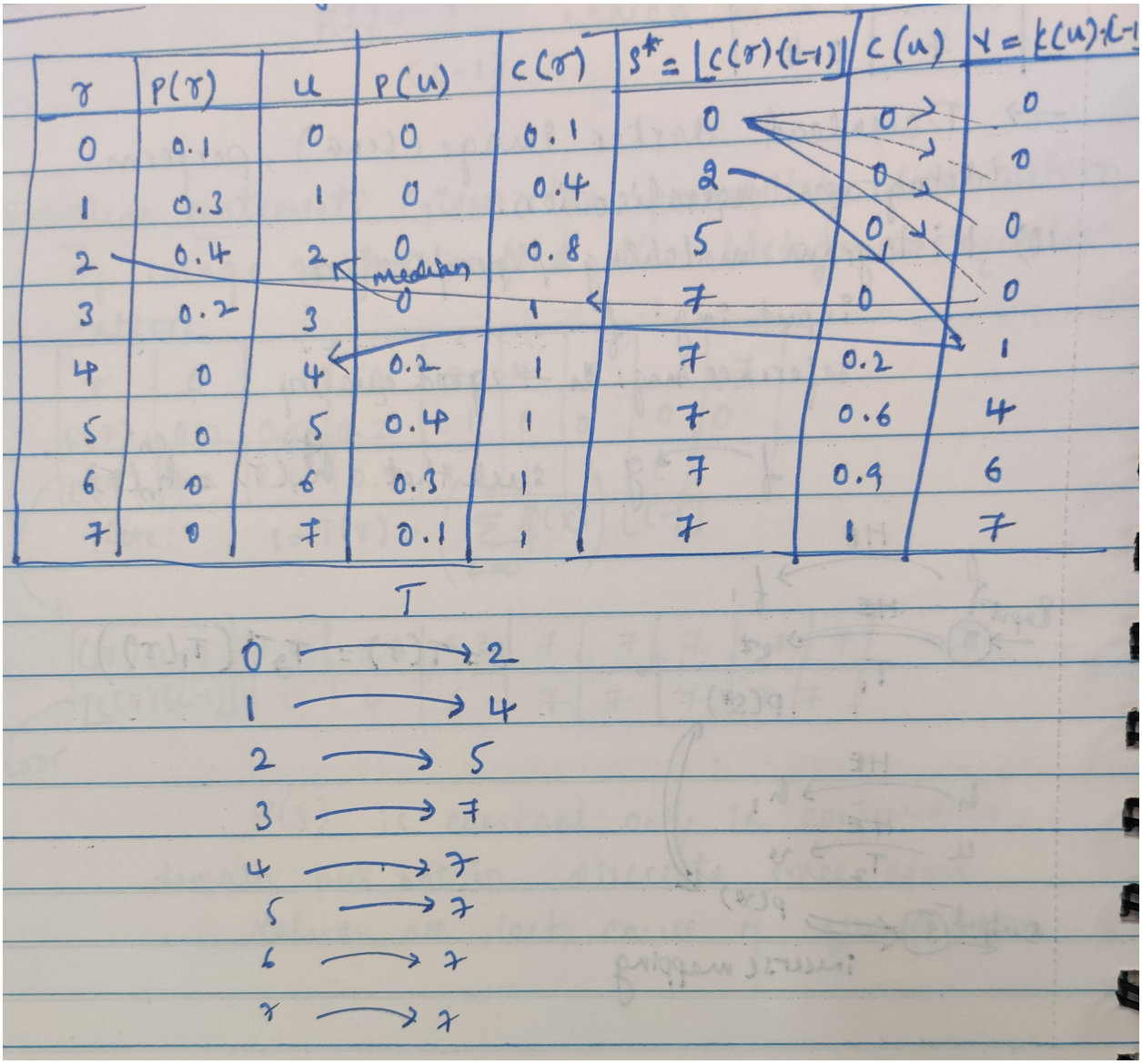
We need to transform,

$$f \to g \text{ such that } h_n^{(g)}(r) = h_n^{(h)}(r)$$

where h_n represents normalised histogram and such transformation T is given by,

$$s = T(r) = T_2^{-1}(T_1(r))$$





r	$P(r)$	u	$P(u)$	$c(r)$	$s^* = \lfloor c(r)(L-1) \rfloor$	$c(u)$	$v = \lfloor c(u)(L-1) \rfloor$
0	0.1	0	0	0.1	0	0	0
1	0.3	1	0	0.4	2	0	0
2	0.4	2	0	0.8	5	0	0
3	0.2	3	0	1	7	0	0
4	0	4	0.2	1	7	0.2	1
5	0	5	0.4	1	7	0.6	4
6	0	6	0.3	1	7	0.9	6
7	0	7	0.1	1	7	1	7

median

T

0 → 2
1 → 4
2 → 5
3 → 7
4 → 7
5 → 7
6 → 7
7 → 7

Neighbourhood processing / operations

Average filter

$$g(x,y) = \sum_{\substack{(a,b)\in N(x,y)\cup\{(x,y)\} \\ 0\le x\le m-1,\ 0\le y\le n-1}} f(a,b) \times \frac{1}{|N(a,b)|+1}$$

Where $N(x,y)$ are the neighbours of (x,y) and $|N(x,y)|$ is cardinality of its neighbours.

- We can observe it as a weighted sum on cardinality of neighbours as weight, $\frac{1}{|N(a,b)+1|}$.
- This is only applied for noisy pixels

Median filter

$$g(x,y) = \text{Median}\left(f(a,b) : (a,b) \in N(x,y) \cup \{(x,y)\}\right)$$

✍🏻 **To prove:** Linearity of average filter

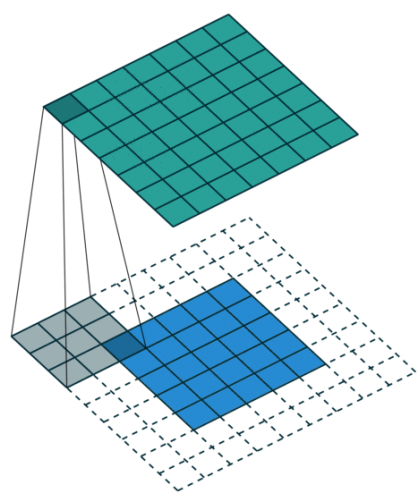
$$Avg(a\cdot f_1(x,y) + b\cdot f_2(x,y)) = a\cdot Avg(f_1(x,y)) + b\cdot Avg(f_2(x,y))$$

expanding LHS,

$$\begin{aligned} Avg(a\cdot f_1(x,y) + b\cdot f_2(x,y)) &= \frac{1}{N}\sum a\cdot f_1(x,y) + b\cdot f_2(x,y) \\ &= \frac{1}{N}\left[a\cdot \sum f_1(x,y) + b\cdot \sum f_2(x,y)\right] \\ &= a\cdot \frac{\sum f_1(x,y)}{N} + b\cdot \frac{\sum f_2(x,y)}{N} \\ &= a\cdot Avg(f_1(x,y)) + b\cdot Avg(f_2(x,y)) \end{aligned}$$

✍🏻 Median filter is not a linear filter.

Template matching



- Correlation: filter is taken as it is.
- Convolution: filter is rotated by 180°

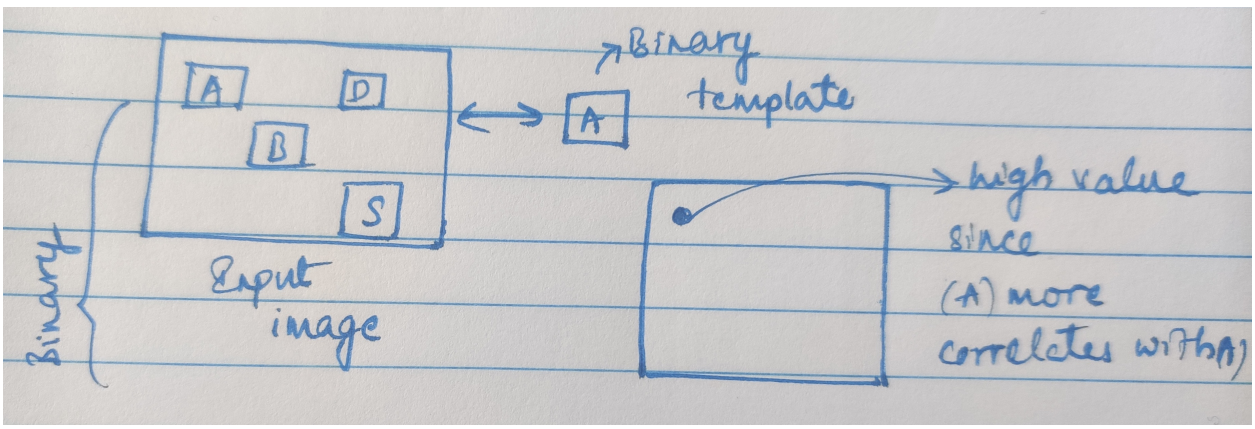
Rotation of window by 180°:

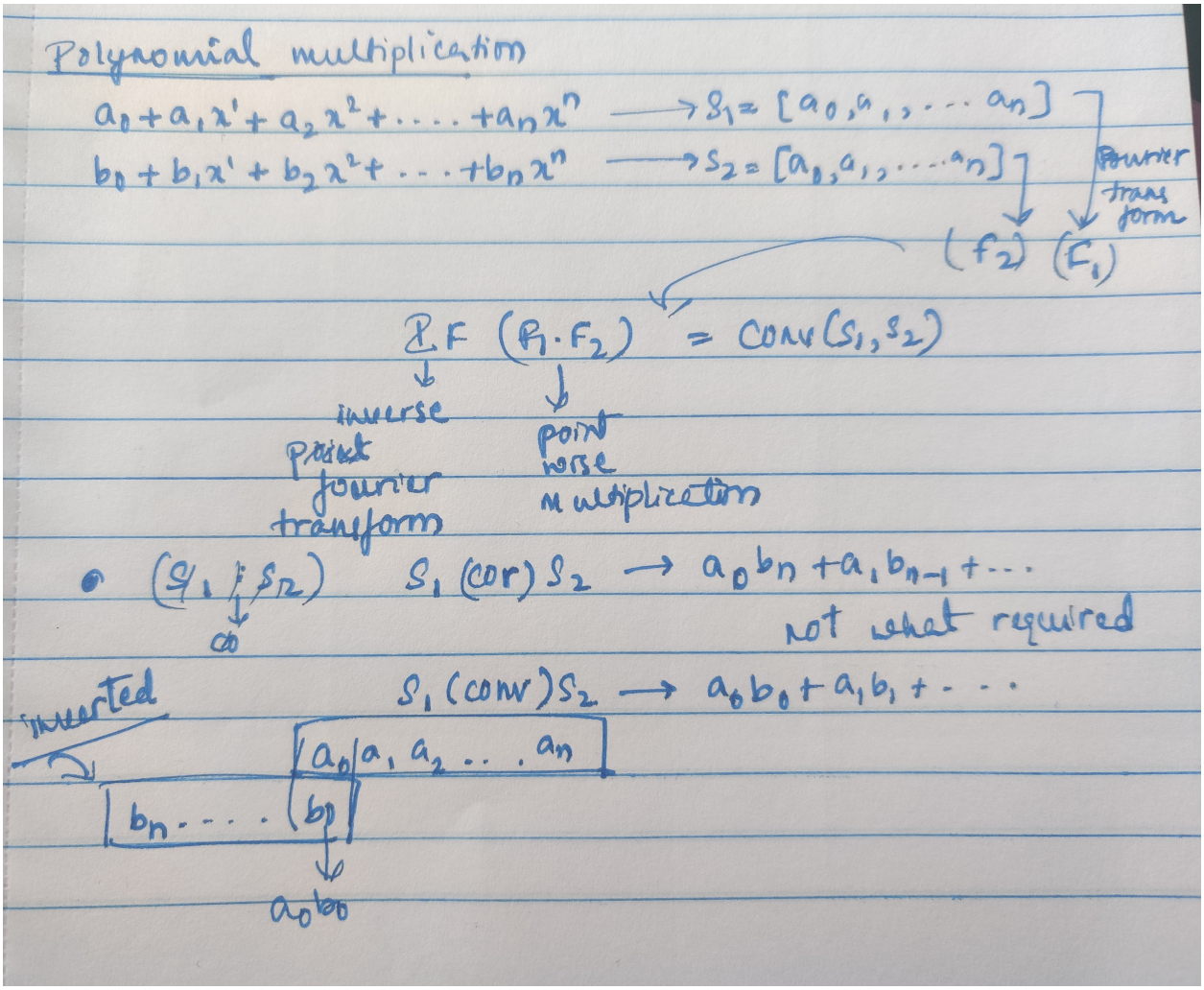
1	2	$\xrightarrow{90^\circ}$	3	3		6	$\xrightarrow{90^\circ}$	9	9		8

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

✍🏻 **Symmetric filter:** A filer w is called as symmetric filter if $\text{rotation}\,(w, 180^\circ) = w$. Hence convolution is same as correlation if filter is symmetric.

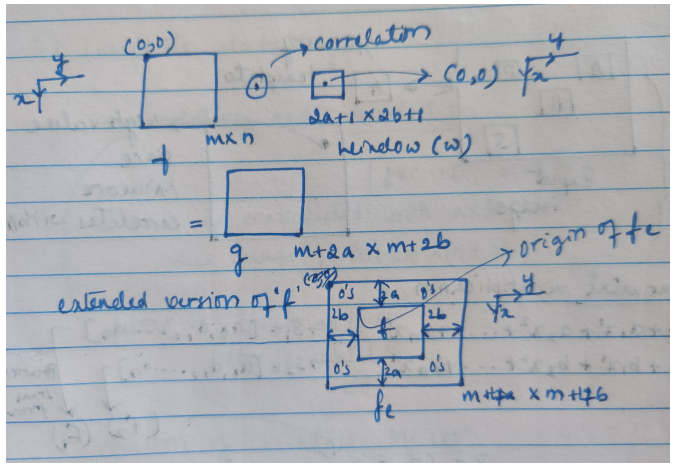
✍🏻 **Note:** For template matching we use normalised correlation





Convolution/Correlation

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



Show that correlation is linear

To prove:

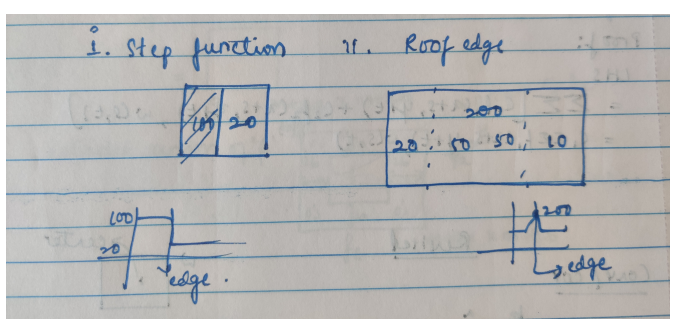
$$\mathrm{Cor}(c_1 \cdot f_1(x,y) + c_2 \cdot f_2(x,y), w(x,y)) = c_1 \cdot \mathrm{Cor}(f_1(x,y), w(x,y)) + c_2 \cdot \mathrm{Cor}(f_2(x,y), w(x,y))$$

$$\begin{aligned}
\mathrm{RHS} &= \mathrm{Cor}(c_1 \cdot f_1(x,y) + c_2 \cdot f_2(x,y), w(x,y)) \\
&= \sum_{t=-b}^{b} \sum_{s=-a}^{a} c_1 \cdot f(x+s, y+t) \cdot w(s,t) + c_2 \cdot f(x+s, y+t) \cdot w(s,t) \\
&= \sum_{t=-b}^{b} \sum_{s=-a}^{a} c_1 \cdot f(x+s, y+t) \cdot w(s,t) + \sum_{t=-b}^{b} \sum_{s=-a}^{a} c_2 \cdot f(x+s, y+t) \cdot w(s,t) \\
&= c_1 \cdot \sum_{t=-b}^{b} \sum_{s=-a}^{a} f(x+s, y+t) \cdot w(s,t) + c_2 \cdot \sum_{t=-b}^{b} \sum_{s=-a}^{a} f(x+s, y+t) \cdot w(s,t) \\
&= c_1 \cdot \mathrm{Cor}(f_1(x,y), w(x,y)) + c_2 \cdot \mathrm{Cor}(f_2(x,y), w(x,y))
\end{aligned}$$

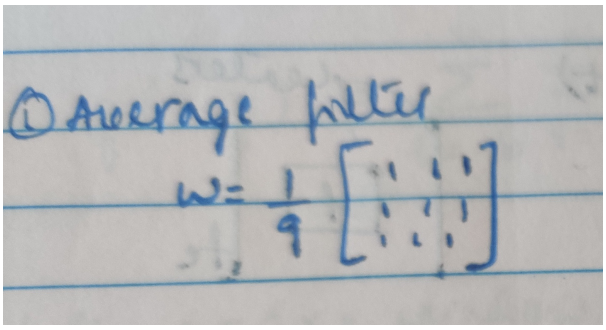
Filter design

Edge is the set of all edge co-ordinate points, a sudden change(can be modelled as step function) pixel values.

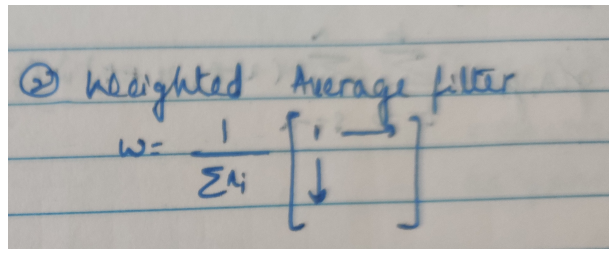
Types:



Average filter



Weighted average filter



Gaussian filter

$$w = \frac{\exp(-\frac{(x^2+y^2)}{2\sigma^2})}{\sum_{y=-b}^{b}\sum_{x=-a}^{a}\exp(-\frac{(x^2+y^2)}{2\sigma^2})} = \frac{h(x,y)}{\sum\sum h(x,y)}$$

Derivative filter

- Derivative in discrete:

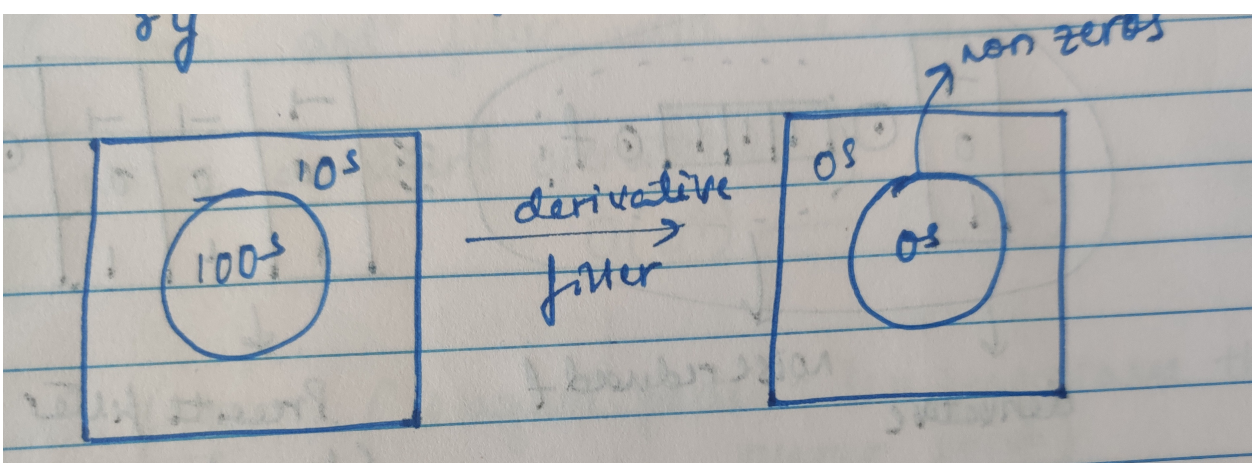
$$f'(x) = f(x+1) - f(x)$$

- Derivative in continuous:

$$\lim_{h\to 1}\frac{f(x+h)-f(x)}{h}$$

- In **2D:**

$$\frac{\partial f(x,y)}{\partial x} = f(x+1,y) - f(x,y)$$
$$\frac{\partial f(x,y)}{\partial y} = f(x,y+1) - f(x,y)$$



✍🏻 **Note:** We can replicate derivative filter using $|-1|+1|$ as window in correlation

- For x

$$\frac{\partial f}{\partial x} = f \odot \begin{array}{|c|}\hline -1 \\ \hline +1 \\ \hline \end{array} \text{(correlation)}$$

$$\frac{\partial f}{\partial x} = f \otimes \begin{array}{|c|}\hline +1 \\ \hline -1 \\ \hline \end{array} \text{(convolution)}$$

- For y

$$\frac{\partial f}{\partial y} = f \odot \begin{array}{|c|c|}\hline -1 & +1 \\ \hline \end{array} \text{(correlation)}$$

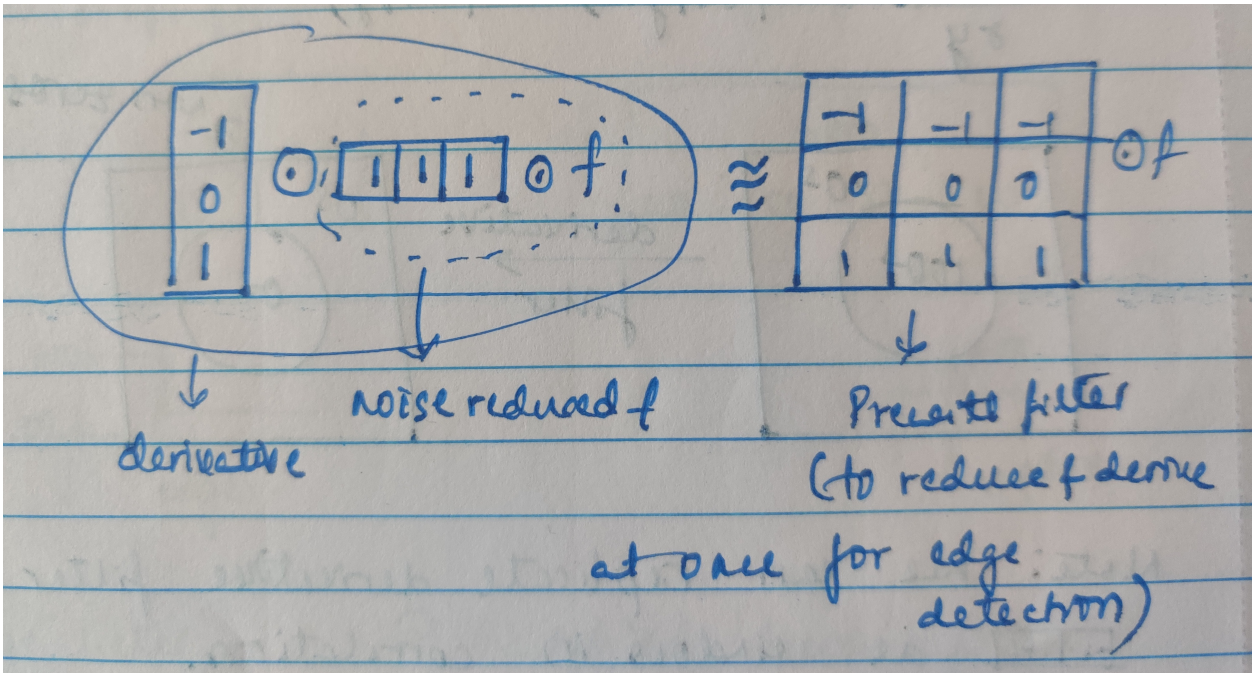
$$\frac{\partial f}{\partial y} = f \otimes \begin{array}{|c|c|}\hline +1 & -1 \\ \hline \end{array} \text{(convolution)}$$

Prewitt filter design

- Derivative in x direction:

$$\underbrace{\begin{array}{|c|}\hline -1 \\ \hline 0 \\ \hline +1 \\ \hline \end{array}}_{\text{Derivative filter}} \odot \begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline \end{array} = \underbrace{\begin{array}{|c|c|c|}\hline -1 & -1 & -1 \\ \hline 0 & 0 & 0 \\ \hline 1 & 1 & 1 \\ \hline \end{array}}_{\text{Prewitt filter}}$$

$$\underbrace{\begin{array}{|c|}\hline -1 \\ \hline 0 \\ \hline +1 \\ \hline \end{array}}_{\text{Derivative filter}} \otimes \begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline \end{array} = \underbrace{\begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline 0 & 0 & 0 \\ \hline -1 & -1 & -1 \\ \hline \end{array}}_{\text{Prewitt filter}}$$



> **Note:** Convolution is commutative.

- Derivative in y direction:

$$\underbrace{\begin{array}{|c|c|c|}\hline -1 & 0 & +1 \\ \hline\end{array}}_{\text{Derivative filter}} \odot \begin{array}{|c|}\hline 1 \\ \hline 1 \\ \hline 1 \\ \hline\end{array} = \underbrace{\begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline -1 & 0 & 1 \\ \hline -1 & 0 & 1 \\ \hline\end{array}}_{\text{Prewitt filter}}$$

$$\underbrace{\begin{array}{|c|c|c|}\hline -1 & 0 & +1 \\ \hline\end{array}}_{\text{Derivative filter}} \otimes \begin{array}{|c|}\hline 1 \\ \hline 1 \\ \hline 1 \\ \hline\end{array} = \underbrace{\begin{array}{|c|c|c|}\hline 1 & 0 & -1 \\ \hline 1 & 0 & -1 \\ \hline 1 & 0 & -1 \\ \hline\end{array}}_{\text{Prewitt filter}}$$

> **Note:** Time complexity(No. of dominant operations) of correlation/convolution for image of size $n \times n$ and filter with size $n \times n$ will be $O(n^4)$ $[O((n^2)^2)$ if n^2 is base$]$

Sobel filter

Intuition: Diagonal points are less important than nearer points

- Derivative in x direction, considering convolution:

$$\begin{array}{|c|c|c|}\hline 1 & 2 & 1 \\ \hline 0 & 0 & 0 \\ \hline -1 & -2 & -1 \\ \hline\end{array}$$

- Derivative in y direction, considering convolution:

1	0	−1
2	0	−2
1	0	−1

Median filter

✍🏻 **Note:** Used for salt and pepper noise removal on image

45 deg

0	−1
1	0

Correlation

0	1
−1	0

Convolution

135 deg

−1	0
0	1

Correlation

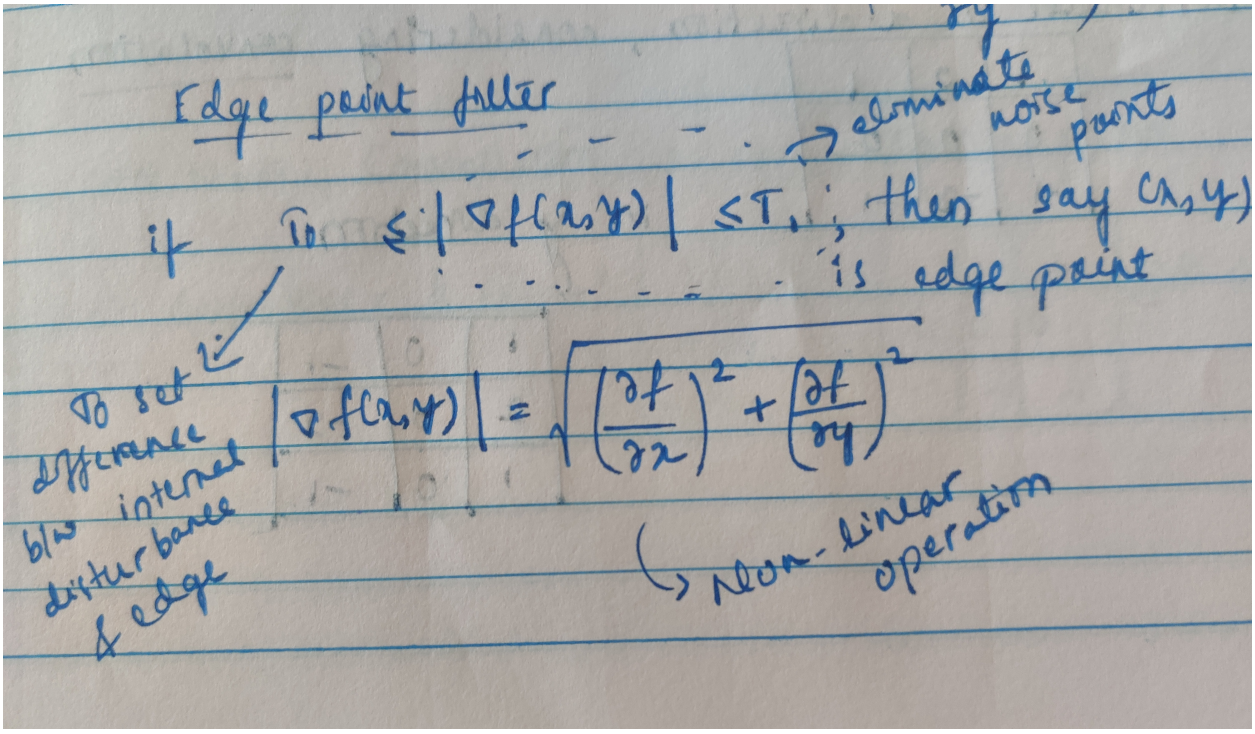
1	0
0	−1

Convolution

Gradient filter

For image $f(x, y)$, the gradient of f is denoted by

$$\nabla f(x, y) = \begin{pmatrix} \frac{\partial f(x,y)}{\partial x} \\ \\ \frac{\partial f(x,y)}{\partial y} \end{pmatrix}$$



Laplace filter

$$\nabla^2 f(x,y) = \frac{\partial^2 f(x,y)}{\partial x^2} + \frac{\partial^2 f(x,y)}{\partial y^2}$$

We know that,

$$\frac{\partial f(x,y)}{\partial x} = f(x+1,y) - f(x,y) \text{ and } \frac{\partial f(x,y)}{\partial x} = f(x,y) - f(x-1,y)$$

Hence,

$$\begin{aligned}
\frac{\partial^2 f(x,y)}{\partial x^2} &= \frac{\partial}{\partial x}\left[\frac{\partial f(x,y)}{\partial x}\right] \\
&= \frac{\partial}{\partial x}[f(x+1,y) - f(x,y)] \\
&= f(x+1,y) - f(x,y) - [f(x,y) - f(x-1,y)] \\
&= f(x+1,y) + f(x-1,y) - 2f(x,y) \text{ .....(1)}
\end{aligned}$$

Similarly,

$$\frac{\partial^2 f(x,y)}{\partial y^2} = f(x,y+1) + f(x,y-1) - 2f(x,y) \text{ .....(2)}$$

Therefore,

$$
\begin{aligned}
\nabla^2 f(x,y) &= \frac{\partial^2 f(x,y)}{\partial x^2} + \frac{\partial^2 f(x,y)}{\partial y^2} \\
&= f(x+1,y) + f(x,y+1) + f(x-1,y) + f(x,y-1) - 4f(x,y)
\end{aligned}
$$

Convolution for Laplace filter

$$
\nabla^2 f = f \otimes \begin{array}{|c|c|c|}\hline 0 & 1 & 0 \\ \hline 1 & -4 & 1 \\ \hline 0 & 1 & 0 \\ \hline \end{array}_{\text{Positive operator}}
$$

Alternatively,

$$
\nabla^2 f = f \otimes \begin{array}{|c|c|c|}\hline 0 & -1 & 0 \\ \hline -1 & 4 & -1 \\ \hline 0 & -1 & 0 \\ \hline \end{array}_{\text{Negative operator}}
$$

Image sharpening

$$
g(x,y) = f(x,y) + c \cdot |\nabla^2 f(x,y)|
$$