① Translation: Moving the point from one place to another by t_x, t_y

$$A' = \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

where : $x, y \Rightarrow$ og coordinates of point A

$x', y' \Rightarrow$ after performing translation

② Scaling: Enlarging or Compressing an image by a factor S_x, S_y

$$A' = \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

③ Rotation :

Anticlockwise

$$A' = \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$\Rightarrow$ Rotation about origin

If we wanna perform rotation about another point q $\Rightarrow$ we first have to translate & get that point (and ur Δ or whatever) to the origin, then perform the rotation & translate back

$$A' = \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 1 \\ -\sin\theta & \cos\theta & 1 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

④ Reflection :

$$M_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad M_y = \begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

~~Rotation~~ Reflection about x-axis, only includes changes in Y-coordinates

" " y-axis

" " " " x-coordinates

① Transformation General Formula:

$$\boxed{\text{Transform} = \begin{bmatrix}\text{Transformation}\\ \text{Matrix}\end{bmatrix} \times f(x,y) \times \begin{bmatrix}\text{Transformation}\\ \text{Matrix}\end{bmatrix}^T}$$

where: (i) $f(x,y) \Rightarrow$ Image

(ii) Transformation Matrix $\Rightarrow$ Changes w/ diff shiz

↳ DFT, DCT, Haar, KL (IDK)

↳ can be 2x2 or 4x4 depending on what size matrix is $f(x,y)$

↳ learn 2x2, 4x2 for all three above

② DFT:

~~Matrix~~ 2x2 Matrix: $\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$

4x4 Matrix = $\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}$

③ ~~Bcqes~~ Haar :

2x2 Matrix : $\frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$

4x4 Matrix : $\frac{1}{2} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ +\sqrt{2} & -\sqrt{2} & 0 & 0 \\ 0 & 0 & \sqrt{2} & -\sqrt{2} \end{bmatrix}$

④ DCT :

4x4 Matrix: $\begin{bmatrix} 0.5 & 0.5 & 0.5 & 0.5 \\ 0.65 & 0.27 & -0.27 & -0.65 \\ 0.5 & -0.5 & -0.5 & 0.5 \\ 0.27 & -0.65 & 0.65 & -0.27 \end{bmatrix}$

① Haar Wavelet:

Transformed Matrix : Row wise ⇒

$$\begin{bmatrix} \frac{a_{00}+a_{01}}{2} & \frac{a_{02}+a_{03}}{2} & \frac{a_{00}-a_{01}}{2} & \frac{a_{02}-a_{03}}{2} \\ \frac{a_{10}+a_{11}}{2} & \vdots & \vdots & \vdots \\ \frac{a_{20}+a_{21}}{2} & \vdots & \vdots & \vdots \\ \frac{a_{30}+a_{31}}{2} & & & \end{bmatrix}$$

Transformed Matrix Column :

$$\begin{bmatrix} \frac{a_{00}+a_{01}}{2} + \frac{a_{10}+a_{11}}{2} \\ \frac{a_{20}+a_{21}}{2} + \frac{a_{30}+a_{31}}{2} \\ \frac{a_{00}+a_{01}}{2} - \frac{a_{10}+a_{11}}{2} \\ \frac{a_{20}+a_{21}}{2} - \frac{a_{30}+a_{31}}{2} \end{bmatrix}$$

① KL-Transform:

(i) Find vector x: Find the coordinates of the object (1s if binary image)

(ii) Find Mean μ_x: Sum of all values divided by N for row1 & row2

(iii) Find Covarience:

$$C = \exp\left[(x_i - \mu_i)(x_i - \mu_i)^T\right]$$

(iv) Find "C" matrix:

C = ~~sum~~ mean of all values for each element

(v) Find Eigen Values:

$$|C_x - \lambda I| = 0$$

↳ find values of λ

(vi) Find Eigen Vectors:

Put value of λ_1 & λ_2, find

$x_1 = _ x_2 \Rightarrow$ Eigen Vector

(VII) Find Transformation Matrix:

$$A = \begin{bmatrix} e_0^T \\ e_1^T \\ \vdots \\ e_n^T \end{bmatrix}$$

$\Rightarrow$ 1st Eigen ~~value~~ vector should correspond to greatest λ value & should decrease down

(VIII) The output after transformation:

$$\boxed{Y = A\left[X - \mu_x\right]}$$

① Steps for Histogram Equalization:

(i) Write highest pixel value, $2^n \geq$ highest pixel value, n = no. of bit to represent that value. n = no. of gray levels

(ii) Write pixels corresponding to no. of occurences of that pixel
Gray Level VS Occurence of that gray level

(iii) Draw corresponding Histogram for i/p image

(iv) Make Table: Gray level, n_k = no. of pixels, PDF = $\frac{n_k}{Sum}$, CDF = Sum till previous pixel $\rightarrow S_k$

$S_k \times$ (highest gray level $= 2^n$), Equalization (take approx of $(S_k \times 2^n)$)

(a) Gray Level
(b) n_k = no. of pixels for each Gray level
(c) PDF = n_k / Sum, Sum = Σn_k

(d) CDF = S_k = First as it is, rest gets added on to it
(e) $S_k \times 2^n$ =
(f) Equalization level = Take approx of $(S_k \times 2^n)$

(v) Draw Histogram of o/p

(vi) Make corresponding image matrix, basically compare i/p & o/p gray levels & replace those values from the og i/p matrix

① Histogram Matching :

(i) Step 1 : Make "Equilization" table for both Histo 1 & Target Histo

(ii) Step 2 : For Histo 1, add a column "New n_k", write corresponding n_k for distinct numbers while if the numbers repeat, write sum of all n_k for that number (equalization number).

Imp. (iii) Step 3 : Take 1st & last column from histo 2 (target) (Gray, Equi) and last two histo 1 (og histo) (Equi, New n_k)

(iv) Step 4 : Match the value for gray level from checking the equi value in table 2 (from histo 1), if there exists a value, new n_k is the final value, if it doesn't take nearest lowest value from table 2 (histo 1) as final value for it.

(v) Step 5 : Draw final "gray level" & "no. of pixels" table and Histogram.

Smoothing Spatial Filters in digital image processing
To exit full screen, press esc

Smoothing Spatial filters

- Linear filters
  - Mean/Box filter
  - Weighted average filter
  - Gaussian filter
- Non-linear filters (Order-Statistic filter)
  - Median filter
  - Max filter
  - Min filter

1:09 / 10:18

Smoothing Spatial Filters in digital image processing
To exit full screen, press esc

* Non-linear (Order-Statistic) filters

Their response is based on ordering (ranking) the pixels contained in the image area encompassed by the filter, and then replacing the value of the center pixel with the value determined by the ranking result.

1) Median filter - Find the median of all the pixel values.
2) Min filter - Find the minimum of all the pixel values.
3) Max filter - Find the maximum of all the pixel values.

5:34 / 10:18

(1) ~~Spari~~ Spatial Filtering: Rejecting / Accepting certain freq.

if it passes low ~~pass~~ freq. ⇒ low pass filter
" " " high freq ⇒ high " "

Spatial Filters: Masks, Kernels, Templates & windows

(2)

Spatial Filters

- Smoothing (Low Pass)
- Sharpening (High Pass)

(3)

Smoothing

- Linear
  - Mean / Box
  - Weighted Avg
  - Gaussian
- Non-Linear (Order Static)
  - Median
  - Max
  - Min

(A) Linear Filters:

(4) Mean / Box : All coefficients are same

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

$\frac{1}{9}$ → sum of all pixels

② Weighted Average: More value to 4-adjacent pixels from center pixel & lesser to diagonal

$$\frac{1}{\underbrace{16}_{\text{Sum}}} \times \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \Rightarrow \text{Mask}$$

③ Gaussian Filter: a mathematical f^n gives weights

$$G_\sigma(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$

$\Rightarrow$ 2D - Gaussian Function

where: σ = std. deviation

$$\frac{1}{16} \times \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \Rightarrow \text{Mask}$$

(B) Non-Linear Filters:

(i) Median Filter : Find median of all pixels by writing in ascending order & finding middle value

(ii) Min " : " min " " "

(iii) Max " : " max " " "

fifty

① Sharpening Filters:

(i) First order Derivative for 1-D f^n:

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

(ii) Second order derivative for 1-D f^n:

$$\boxed{\frac{\partial^2 f}{\partial x^2} = f(x+1) + f(x-1) - 2f(x)}$$

(3) Laplacian Filter:

$$\boxed{\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}}$$

where: (i) $\frac{\partial^2}{\partial x^2} = f(x+1,y) + f(x-1,y) - 2f(x,y)$

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

$$\therefore \boxed{\nabla^2 f = f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y)}$$

obtaining Laplacian Mask now:

for i/p image:

$$\begin{matrix} f(x-1,y-1) & f(x,y-1) & f(x+1,y-1) \\ f(x-1,y) & f(x,y) & f(x+1,y) \\ f(x-1,y+1) & f(x,y+1) & f(x+1,y+1) \end{matrix}$$

Mask: just substitute coefficients from above eqn:

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \Rightarrow \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix} \Rightarrow \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

imp. Sum of all pixels = 0

(i) Contrast Stretching :

(i) Plot graph of "r VS S", find slope of all lines as α, β, γ

(ii) Write f^n:

$$S = \begin{cases} \alpha r & 0 \le r < r_1 \\ \beta(r - r_1) + S_1 & r_1 \le r < r_2 \\ \gamma(r - r_2) + S_2 & r_2 \le r \le \text{max value} \end{cases}$$

(iii) Write values :

r	S
0	from above f^n & round-off
1	
2	

(iv) Make new image, old value (r) gets replaced w/ new value (S) in the image.

(5) Estimation of Degradation Models:

(i) By Observation: Looking at the degraded image, we can figure out the degradation function

(ii) By Experimentation: Some experimental setup.

(iii) By Modeling: Apply Maths

(6) Restoration: (i) Inverse filtering

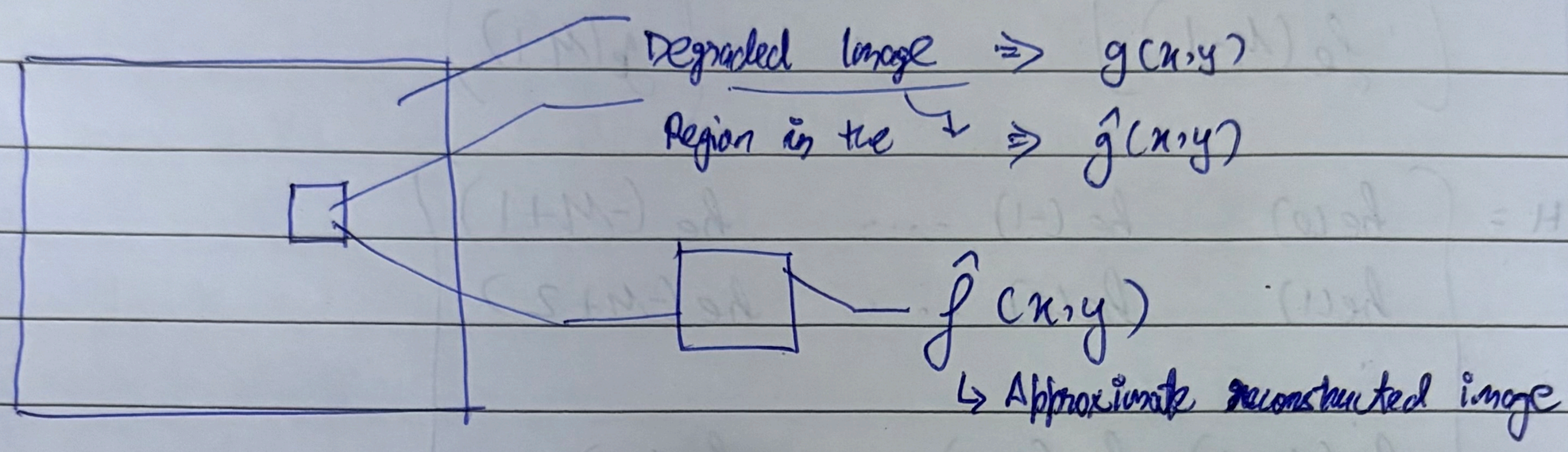
(ii) Weiner "

(iii) Constrained Least Square Filtering

① By Observation : We'll look at the degraded image to try to figure out the degradation.

(i) We identify a region in the image, region that contains a simple structure.

(ii) We try to estimate an og image that should've been degraded to get that degraded region.



$\hat{g}(x,y) = g_s(x,y)$ (cuz it's a subimage, just call it $g_s(x,y)$)

$g_s(x,y) \xrightarrow{F.T} G_s(u,v)$ ---- ①

$\hat{f}(x,y) = f_s(x,y)$ (og image that we have reconstructed from the region)

$f_s(x,y) \xrightarrow{F.T} F_s(u,v)$ — ②

$$\boxed{\text{Degradation } f^n \text{ Estimation} = H = \frac{G_s(u,v)}{F_s(u,v)}}$$

↳ $H_s(u,v)$

↳ We haven't considered noise

↳ Thus choosing a better region

(2) By Experimentation:

$F(u,v) = A$ → const. mag of impulse response

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

$$\boxed{H(u,v) = \frac{G(u,v)}{A}}$$

↳ Somehow related to impulse response

We try to get an imaging setup same as the one used to obtain the degraded image in the first place.

We get the impulse response of the imaging setup

↳ We simulate by striking it w/ a bright spot of light.

(3) Modeling: Assumes a theoretical model for degredation based on system behaviour

for ex: Gaussian Blur

Motion Blur

Atmospheric Turbulence : $H(u,v) = e^{-k(u^2+v^2)^{5/6}}$

(4) Motion Blurring:

$f(x,y)$ ⇒ undergoes motion

$x_0(t), y_0(t)$ ⇒ time-varying components in x & y direction

∴

$$\boxed{g(x,y) = \text{blurred image} = \int_0^T f(x - x_0(t), y - y_0(t))\,dt}$$

where: T ⇒ is time when shutter of camera is open

Obtaining the F.T we use:

$$G(u,v) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(x,y)\, e^{-2\pi(ux+vy)}\,dx\,dy \quad \text{--- (1)}$$

$$\boxed{{}^{*}H(u,v) = \frac{T}{\pi(ua+vb)} \sin[\pi(ua+vb)]\, \exp[-j\pi(\;)]}$$

after maths we get:

$$\boxed{G(u,v) = F(u,v)\int_0^T \exp\left[-j2\pi\left(u x_0(t) + v y_0(t)\right)\right] dt}$$

$$\boxed{{}^{*}H(u,v) = \frac{G(u,v)}{F(u,v)} = \int_0^T \exp\left[-j2\pi\left(u x_0(t) + v y_0(t)\right)\right] dt}$$

① Restoration : (i) Inverse Filtering

② Inverse Filtering :

We know that: $G(u,v) = H(u,v)\,F(u,v)$

$$\therefore \boxed{\hat{F}(u,v) = \frac{G(u,v)}{H(u,v)}}$$

where: (i) $G(u,v) = F.T[g(x,y)]$

(ii) $H(u,v)$ = Degradation in approximated terms

(iii) $\hat{F}(u,v)$ = reconstructed image

but if

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

$$\boxed{\hat{F}(u,v) = F(u,v) + \frac{N(u,v)}{H(u,v)}}$$

↳ if $H(u,v)$ ↓, Noise is very high

Imp. In practicality, it is not used

Date ___/___/_____

(3) Weiner Filtering: Tries to minimize the error $f^{\wedge}(x)$ to reconstruct the og image.

$$\boxed{\hat{F}(u,v) = \left[\frac{1}{H(u,v)} \cdot \frac{|H(u,v)|^2}{|H(u,v)|^2 + K}\right] G(u,v)}$$

where: K is to chosen manually to get the best visual image

Date ___/___/_____

Saathi

① Image segmentation : Extract attributes from an Image.

Types

- Discontinuity Based
  - Point
  - Line
  - Edge
- Similarity Based
  - Thresholding
    - Global
    - Variable
      - Local or Regional
      - Dynamic OR Adaptive
  - Region Growing
  - Region Splitting & Merging

② Point Detection :

We use Laplacian Mask :

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad \text{OR} \quad \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

OR multiplication of -1 w/ both above

$$g(x,y) = \begin{cases} 1, & \text{if } f(x,y) > T \\ 0, & \text{o/w} \end{cases}$$

If after multiplying, the value is greater than Threshold, then its found/detected

Imp. centre pixel of kernel/mask coincides w/ the operating ~~pixel~~ pixel

(2) Line Detection:

Horizontal ⇒ $\begin{bmatrix} -1 & -1 & -1 \\ 2 & 2 & 2 \\ -1 & -1 & -1 \end{bmatrix}$ Vertical ⇒ $\begin{bmatrix} -1 & 2 & -1 \\ -1 & 2 & -1 \\ -1 & 2 & -1 \end{bmatrix}$

$45° \Rightarrow \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix}$ $-45° \Rightarrow \begin{bmatrix} -1 & -1 & 2 \\ -1 & 2 & -1 \\ 2 & -1 & -1 \end{bmatrix}$

Imp. Just multiply w/ the mask, if ans is above Threshold, then it is detected.

(3) Edge Detection:

(i) Roberts:

Horizontal = $G_x = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$ $G_y = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$ = Vertical

$Mag = |G| = \sqrt{G_x^2 + G_y^2}$

(ii) Sobel:

Horizontal = $G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$ $G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$

$Mag = \sqrt{G_x^2 + G_y^2}$ $Angle = \tan^{-1}\left(\frac{G_y}{G_x}\right)$

(iii) Prewitt:

Horizontal: $\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} = G_x$ G_y = Vertical = $\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$

$Mag = G = \sqrt{G_x^2 + G_y^2}$ $Angle = \tan^{-1} \frac{G_y}{G_x}$

(4) Region Growing :

Steps: (i) You're given the coordinates of seed point (x,y) & threshold T and which connectivity to use.

(ii) From that seed point, based on the connectivity given, you compute: absolute difference b/w the (intensity values or gray level values) & seed value

Imp. [if abs diff <= Threshold ⇒ Region Grows
if abs diff > Threshold ⇒ Nothing happens

(iii) Now you repeat the same for the pixels that you obtained when your region grew

(iv) If you're not given the coordinates of seed pixel, you'll be given value of pixel, just apply the same for all pixels that pixels exists in the image.

(v) The resultant segmented image would be 0 ⇒ abs diff ≥ T
1 ⇒ abs diff < T

(basically "1" for where the region grew, "0" o/w)

(5) Region Splitting & Merging :

Steps: (i) Consider full image

(ii) Compute : abs diff (max value - min value)

Imp. [(iii) if abs diff <= T ⇒ don't split
if abs diff > T ⇒ split

(iv) Once you've split evenly find the same thing again for subregions

(v) For merging, we compare the adjacent sub-regions & find max & min values

Imp. [(vi) If abs diff (max val - min val) = T ⇒ we merge the region into a bigger one.

(vii) Repeat the same for other adjacent subregions.

Date ___/___/___

⑥ Thresholding : Difference in intensities b/w objects & b/g.

Steps : (i) T is selected

(ii) if $f(x,y) > T = 1$

olw $= 0$

(iii) $g(x,y) = \begin{cases} 1 & , f(x,y) > T \\ 0 & , \text{olw} \end{cases}$ = segmented Image

Types : (i) Global

(ii) Variable = (i) Local or Regional

(ii) Adaptive or Dynamic

(i) Global :

Steps : (i) If T isn't given, compute it by taking avg of all pixels

(ii) G_1 = pixels w/ value $> T$

G_2 = " " " $< T$

(iii) μ_1 = Avg of all values in G_1

μ_2 = " " " " " G_2

(iv) $T' = \frac{1}{2}(\mu_1 + \mu_2)$

(v) Repeat the process until you get a value of T' closer to T.

Imp. All on this page is a bit dicey

(1) Hough Transform :

Steps : (i) You'll be given ~~an~~ ~~a~~ bunch of points

(ii) Eq^n of line : $y = mx + c$

$$\Rightarrow c = -mx + y \quad \text{--- (1)}$$

(iii) Put the points in the above eq^n & construct lines in the M-C plane

(iv) ~~If~~ If the lines intersect at a point, it means they're collinear.

(v) You can cross check by putting that point in $y = mx + c$ & get all the points that were given in the question.

Frequency domain filtering in image processing - Low pass and High pass filters

1) Multiply the input image by $(-1)^{x+y}$ to center the transform.
2) Compute $F(u,v)$, the DFT of the image from (1).
3) Multiply $F(u,v)$ by a filter function $H(u,v)$.
4) Compute the inverse DFT of the result in (3).
5) Obtain the real part of the result in (4).
6) Multiply the result in (5) by $(-1)^{x+y}$.

6:25 / 24:15

① Spatial to Freq Domain :

$$\longrightarrow \boxed{\text{DFT}} \rightarrow F(u,v) \longrightarrow \boxed{H(u,v)} \longrightarrow G(u,v) \longrightarrow \boxed{\text{IDFT}}$$

(Pre-processing)

$f(x,y)$ $\qquad\qquad$ $g(x,y)$

② Different types of filters : $H(u,v)$

2.1 Smoothening Filters : (a) Ideal Low Pass

(b) Butterworth Low Pass

(c) Gaussian Low Pass

Ideal

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

where : D_0 is cutoff freq

$D(u,v)$ is Euclidean distance

(b) Butterworth:

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

where: D_0 = cutoff freq

n = order of Butter filter

$D(u,v)$ = Euclidean distance

(c) Gaussian:

$$H(u,v) = e^{-D^2(u,v)/2D_0^2} = \exp\left[\frac{-D^2(u,v)}{2D_0^2}\right]$$

where: D_0 = cutoff freq

(2.2) Sharpening Filters: (AKA High Pass Filters)

(a) Ideal High Pass

(b) Butterworth High Pass

(c) Gaussian High Pass

(a) Ideal:

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

(b) Butterworth:

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

(c) Gaussian:

$$H(u,v) = 1 - e^{-D^2(u,v)/2D_0^2} = 1 - \exp\left[\frac{-D^2(u,v)}{2D_0^2}\right]$$

Imp.

$$D(u,v) = \sqrt{\left(u - \frac{M}{2}\right)^2 + \left(v - \frac{N}{2}\right)^2}$$

where: M, N is the order of the matrix

↳ Dis b/w (u,v) & centre pixel

① Spatial to Freq :

Steps: (i) Multiply the i/p image with $(-1)^{x+y}$ matrix pixel by pixel

Imp. (ii) Take DFT w/ help of kernel of matrix obtained in (i).

DFT = $F(u,v) = f(x,y) =$ kernel x image x $(\text{kernel})^T$

(iii) Figure out $H(u,v)$ a/c to the ques by figuring $D(u,v)$

(iv) For that, find $D(u,v)$ matrix by:

$$D(u,v) = \sqrt{\left(u-\frac{M}{2}\right)^2 + \left(v-\frac{N}{2}\right)^2}$$ where: M, N is the order of matrix

(v) Multiply $F(u,v)\,H(u,v)$ pixel by pixel

Imp. (vi) Take Inverse DFT :

$$\text{IDFT} = g(x,y) = G^{-1}(u,v) = \frac{1}{4}\,\text{kernel} \times \text{image} \times \frac{1}{4}(\text{kernel})^T$$

(vi) Multiply w/ $(-1)^{x+y}$ pixel by pixel again to get the final image

① DFT : kernel x image x $(\text{kernel})^T$

$$\text{kernel} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}$$

② IDFT : $\frac{1}{4}$ kernel x image x $\frac{1}{4}(\text{kernel})^T$

$$\text{kernel} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & j & -1 & -j \\ 1 & -1 & 1 & -1 \\ 1 & -j & -1 & j \end{bmatrix}$$

(5) Spatial to freq & Ideal Low pass filtering:

(i) Multiply $(-1)^{x+y}$:

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & -1 & 1 & -1 \\ -1 & 1 & -1 & 1 \\ 1 & -1 & 1 & -1 \\ -1 & 1 & -1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & -1 & 1 & -1 \\ 0 & 0 & 0 & 0 \\ 1 & -1 & 1 & -1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

(ii) Take DFT: kernel × image × (kernel)^T

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 1 & -1 & 1 & -1 \\ 0 & 0 & 0 & 0 \\ 1 & -1 & 1 & -1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} = \begin{bmatrix} 2 & -2 & 2 & -2 \\ 0 & 0 & 0 & 0 \\ 2 & -2 & 2 & -2 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}$$

$$\begin{bmatrix} 0 & 0 & 8 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 8 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} = F(u,v)$$

(iii) $H(u,v)$:

Ideal LPF: $\begin{cases} 1, & D(u,v) \le D_0 \\ 0, & o/w \end{cases}$

finding $D(u,v)$: $D(u,v) = \sqrt{\left(u-\frac{M}{2}\right)^2 + \left(v-\frac{N}{2}\right)^2}$

$= \sqrt{(u-2)^2 + (v-2)^2}$

$D(0,0) = \sqrt{8} \approx 3$

$D(0,1) = \sqrt{5} \approx 2$

$D(0,2) = 2$

$D(0,3) = \sqrt{5} \approx 2$

$D(1,0) = \sqrt{5} \approx 2$

$D(1,1) = \sqrt{2} \approx 1$

$D(1,2) = 1$

$D(1,3) = \sqrt{2} \approx 1$

$D(2,0) = 2$

$D(2,1) = 1$

$D(2,2) = 0$

$D(2,3) = 1$

$D(3,0) = \sqrt{5} \approx 2$

$D(3,1) = \sqrt{2} \approx 1$

$D(3,2) = 1$

$D(3,3) = \sqrt{2} \approx 1$

$H(u,v) = \begin{cases} 1, & D(u,v) \le D_0 \\ 0/w \end{cases}$

Imp. Do Not approx $D(u,v)$, let it be points

↳ Take decimal value

$$D(u,v) = \begin{bmatrix} \sqrt{8} & \sqrt{5} & 2 & \sqrt{5} \\ \sqrt{5} & \sqrt{2} & 1 & \sqrt{2} \\ 2 & 1 & 0 & 1 \\ \sqrt{5} & \sqrt{2} & 1 & \sqrt{2} \end{bmatrix} \qquad H(u,v) = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

(iv) Multiply $H(u,v)\,F(u,v)$:

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & 8 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 8 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 8 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

(v) Take inverse transform: $IDFT = \frac{1}{4}\,\text{kernel} \times \text{image} \times \frac{1}{4}(\text{kernel})^T$

$$\frac{1}{4}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & j & -1 & -j \\ 1 & -1 & 1 & -1 \\ 1 & -j & -1 & j \end{bmatrix} \begin{bmatrix} 0 & 0 & 8 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \frac{1}{4}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & j & -1 & -j \\ 1 & -1 & 1 & -1 \\ 1 & -j & -1 & j \end{bmatrix}$$

$$= \frac{1}{16}\begin{bmatrix} 0 & 0 & 8 & 0 \\ 0 & 0 & 8 & 0 \\ 0 & 0 & 8 & 0 \\ 0 & 0 & 8 & 0 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & j & -1 & -j \\ 1 & -1 & 1 & -1 \\ 1 & -j & -1 & j \end{bmatrix} = \frac{1}{16}\begin{bmatrix} 8 & -8 & 8 & -8 \\ 8 & -8 & 8 & -8 \\ 8 & -8 & 8 & -8 \\ 8 & -8 & 8 & -8 \end{bmatrix}$$

(vi) Multiply the result w/ $(-1)^{x+y}$

$$g(x,y) = \frac{8}{16}\begin{bmatrix} 1 & -1 & 1 & -1 \\ 1 & -1 & 1 & -1 \\ 1 & -1 & 1 & -1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \begin{bmatrix} 1 & -1 & 1 & -1 \\ -1 & 1 & -1 & 1 \\ 1 & -1 & 1 & -1 \\ -1 & 1 & -1 & 1 \end{bmatrix} = \frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & 1 \\ -1 & -1 & -1 & -1 \\ 1 & 1 & 1 & 1 \\ -1 & -1 & -1 & -1 \end{bmatrix}$$

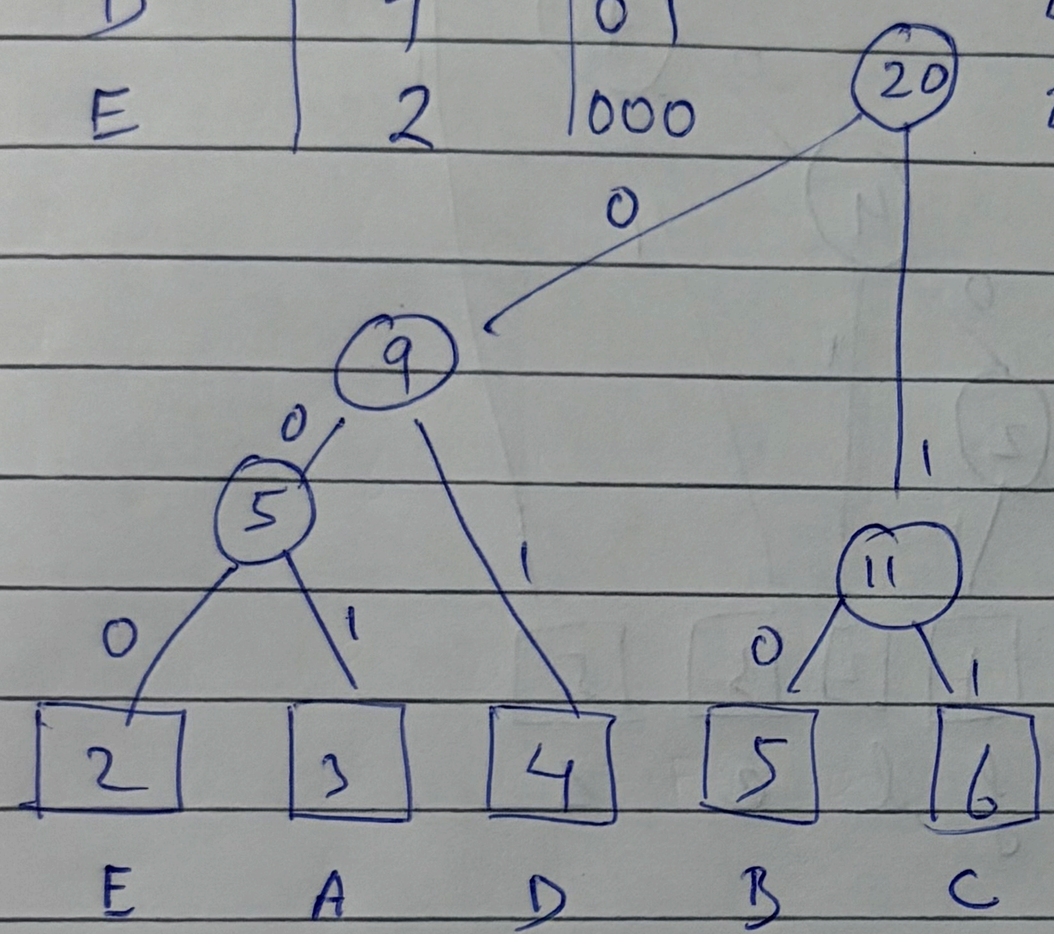
Ans

① Huffman Coding:

Message : BCCABBDDAECCBBAEDDCC

char	count	code	
A	3	001	3×3
B	5	10	5×2
C	6	11	6×2
D	4	01	4×2
E	2	000	2×3

⇒ $\boxed{\text{Size} = 45 \text{ bits}}$



→ write in increasing freq.

⇒ Merge the min nodes out of the nodes left

⇒ Mark left nodes as 0
right nodes as 1

⇒ Just follow pattern on path from root to the letter to get the code

Encoding : 10 11 11 001 .... so on ⇒ just write each corresponding code

Steps: (i) Make table char, count, code & write char & count

(ii) Write letters in increasing order of freq & above them write their frequency

(iii) Merge min nodes every time out of the nodes left

(iv) Mark left as "0" & right as "1"

(v) Follow path for each letter ⇒ code (Huffman)

(vi) Avg code word length = $\frac{\text{Total bits}}{\text{Total characters}}$

(vii) Total bits = (freq × length of code (Huffman) in bits) for all letters

① Message : selfnessless

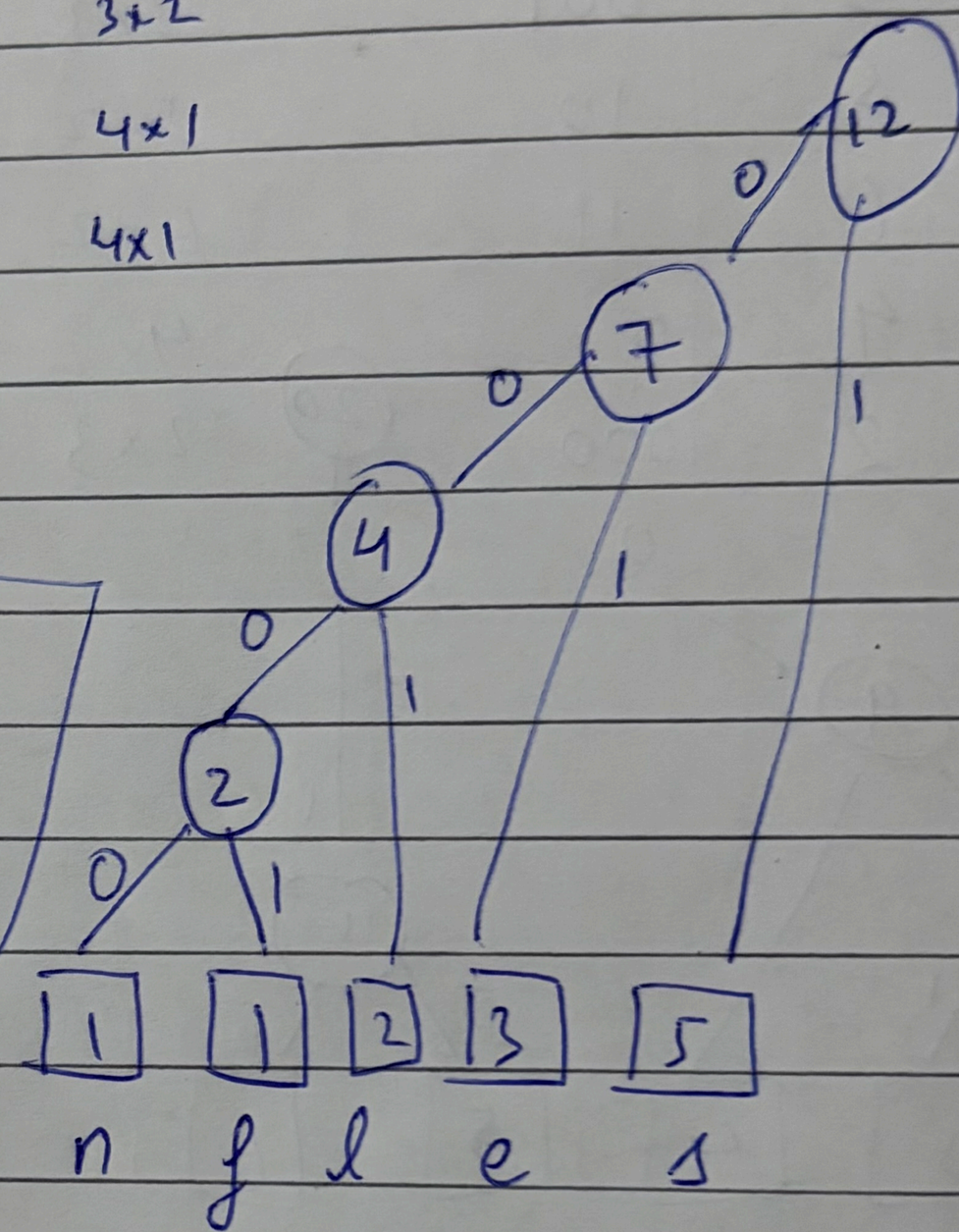
char	count	code	
s	5	1	1×5
e	3	01	2×3
l	2	001	3×2
f	1	0001	4×1
n	1	0000	4×1
		total = 12	

$5+6+6+4+4 = 25$ bits

Length = $\frac{\text{total bits}}{\text{total characters}}$

$= \frac{25}{12}$ bits/character

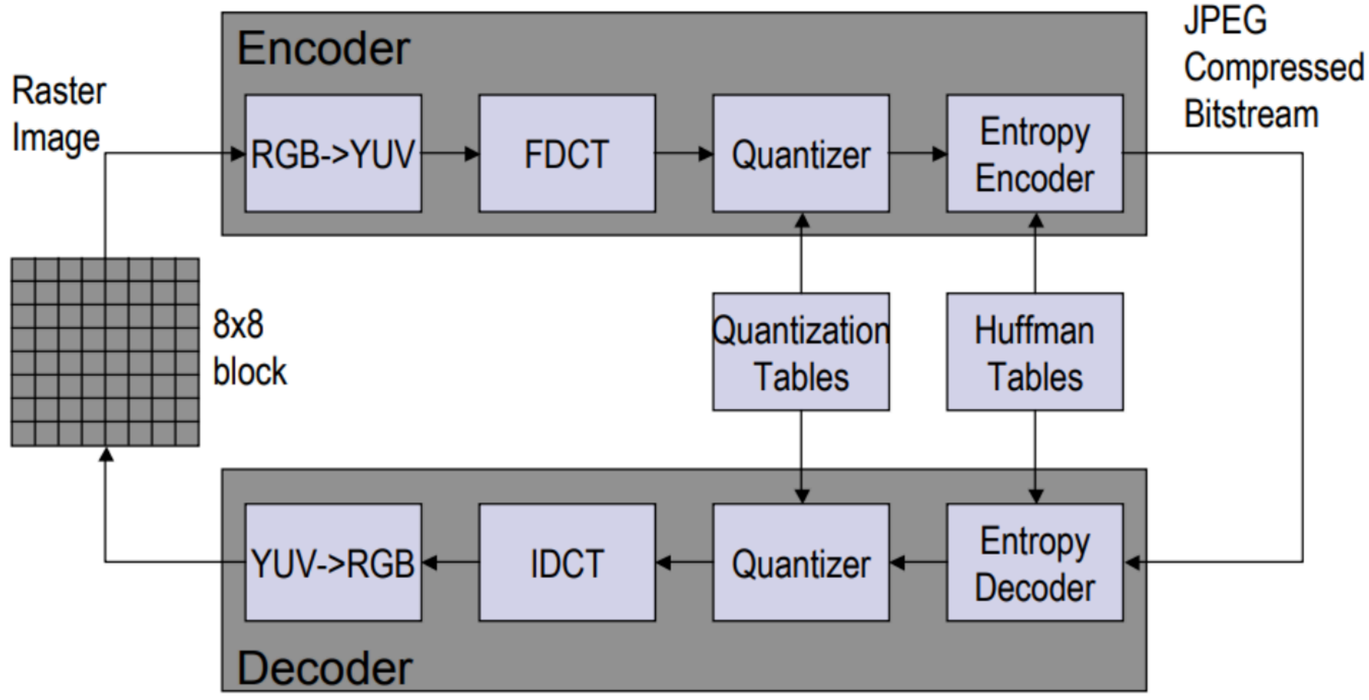
$\neq l$



Classwork for DIVP EALL
Image
classroom.google.com/u/0/w/NzA3NDY5NjM4NzIy/t/all
Drive Study Material 10 CGPA 4th Sem Docs C++ Compiler EMFT EMFT All Lectures DSA Subject DSA Subject 2 Sem-5
Video Processing and compression.pdf
Open with Google Docs

Huffman coding in a zig zag manner

JPEG (Joint Photographic Experts Group)

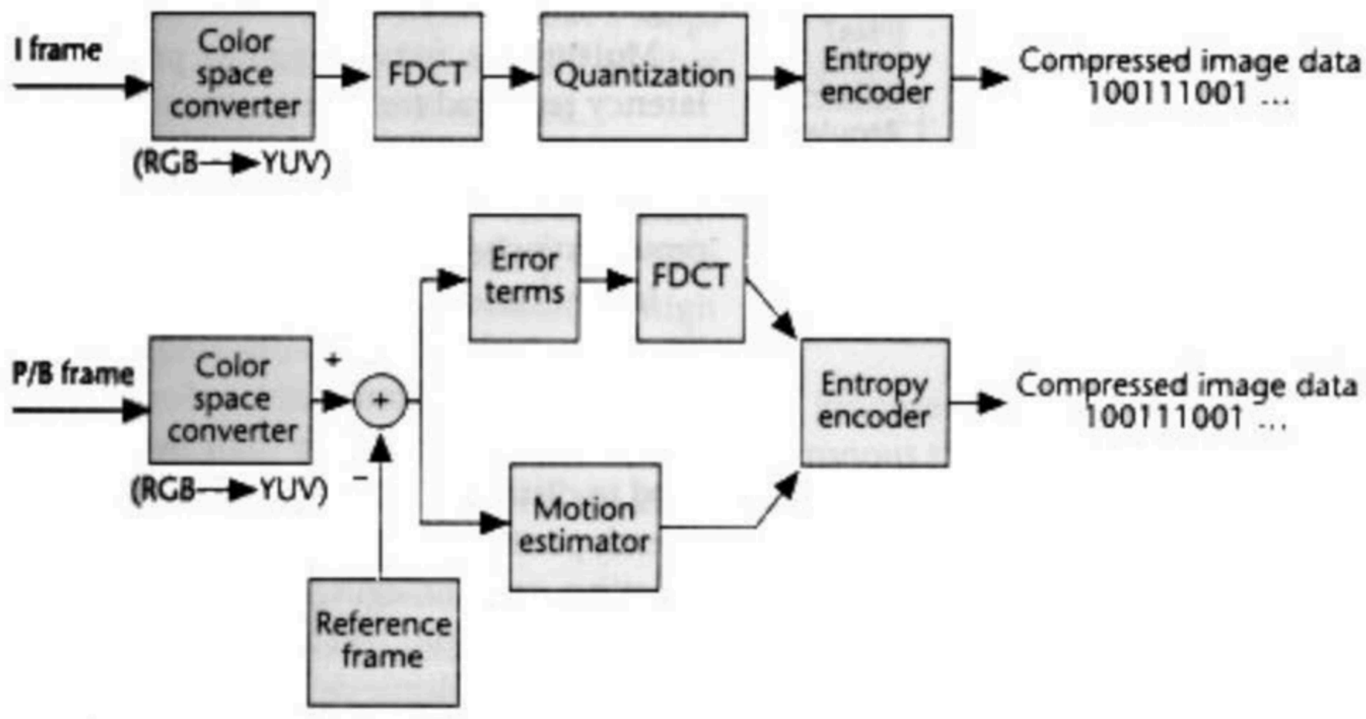


Stream
View material
Page 27 / 57

RGB-YUV(YC_bC_r)

Classwork for DIVP EALL
Image
classroom.google.com/u/0/w/NzA3NDY5NjM4NzIy/t/all
Drive Study Material 10 CGPA 4th Sem Docs C++ Compiler EMFT EMFT All Lectures DSA Subject DSA Subject 2 Sem-5
Video Processing and compression.pdf
Open with Google Docs

MPEG encoder:



Page 45 / 57
Video