DIGITAL IMAGE AND VIDEO PROCESSING

Model compression methods → pruning, foreign? approximation

Methods directly applied on Image

Basic operations (making img more interpretable)

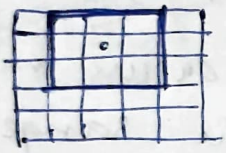
1) Intensity transformations ; eg: gamma correction

→ special case of spatial filtering (convolution)

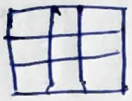
img ⇒ $f(x,y)$

we need impulse-response (filter) of input

3×3 input



of



filter mask

But for intensity transformation we don't need filter mask.

The intensity value has a range $0 \to (L-1)$

for gray scale image → $L = 256$; so the range ⇒ 0 - 255

So the 3×3 neighbourhood get transformed to a 1×1 form.

$$s = T(r)$$

s ↙ output intensity value

r ↓ input intensity value

Transformations:-

1) Image negative transform

$$S = (L-1) - r$$ [∵ as we can't go outside the range $0-(L-1)$]

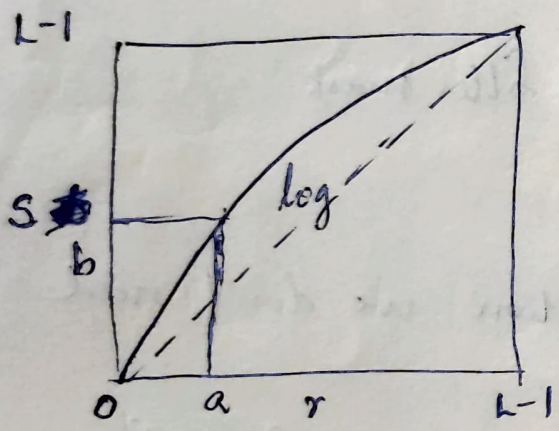
It is used in medical imaging (perception) Detel Detection of a minute anomaly (dark) ~~can be found dete~~ when the whole & surrounding is bright.

2) log transformation

as log 0 not defined

$$S = c \cdot \log(1+r)$$

c → constant to ensure values fall in the range

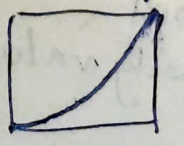


---- → when there is no change in intensity

Narrow ranged intensity values are mapped towards a wider ranged intensity values. → so stretching happens

we do this when the information that we need is lying within the narrow intensity range.

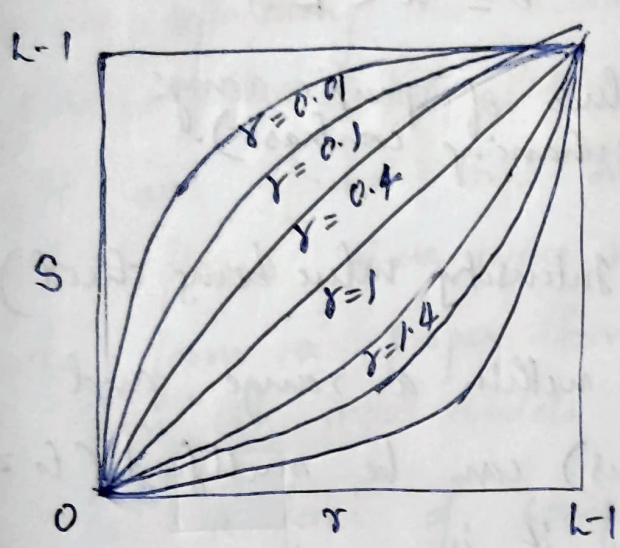
Exponential function gives the inverse mapping



3) Power Law Transformation / Gamma Correction

$$S = c(r+\varepsilon)^{\gamma}$$

This gives a family of curves based on the γ values.

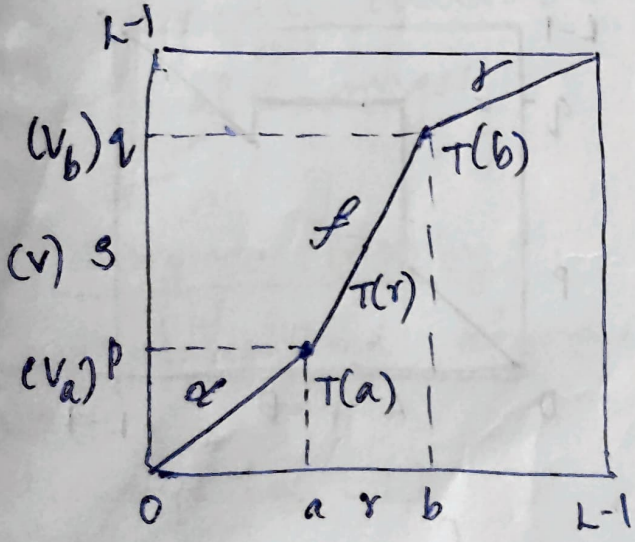


changing the γ value reflects in the resultant image.

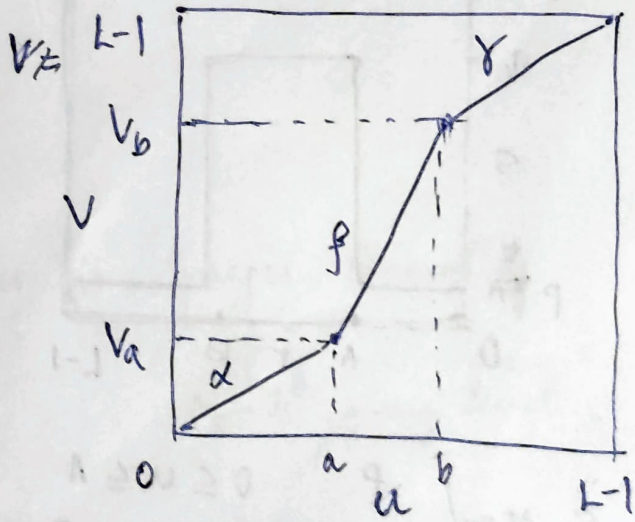
γ lesser than 1 → log

γ more than 1 → exponential

4) Contrast Stretching



(u)



α, β, γ → slopes

u, v → generalized for a, b

$$V = \begin{cases} \alpha u, & 0 \le u \le a \\ \beta(u-a) + V_a, & a \le u < b \\ \gamma(u-b) + V_b, & b \le u < L-1 \end{cases}$$

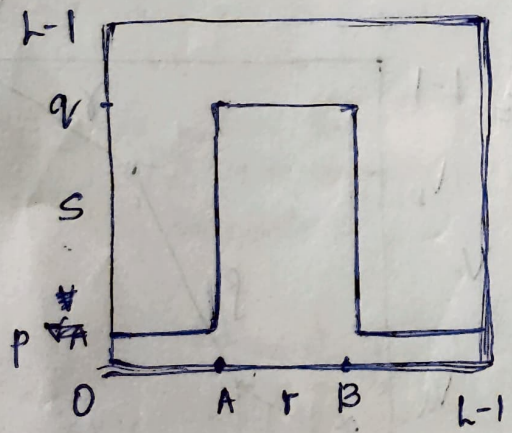
To stretch intensity values of specific range to a specific range (for enhancing contrast).

~~5) Intensity value~~

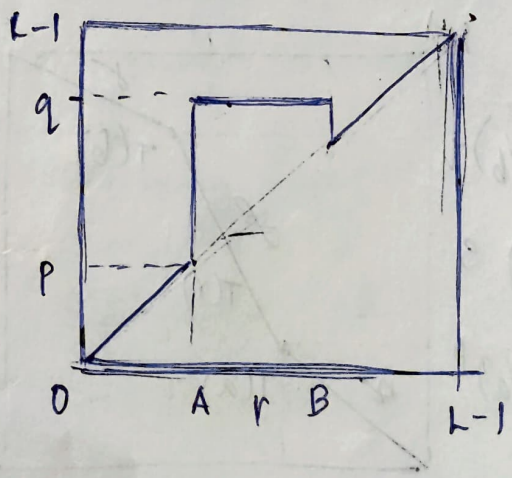
5) Intensity-Level Slicing (Intensity value being sliced)

To boost the values within a range, and the background (other values) can be nullified (to zero) or we can keep it as it is

Without background



With background



$$S = \begin{cases} p & 0 \le u \le A \\ q & A \le u \le B \\ p & B \le u \le L-1 \end{cases}$$

$$S = \begin{cases} \end{cases}$$

6) Bit - Plane Slicing ($B_7, B_6, B_5, B_4, B_3, B_2, B_1, B_0$)

B_0 plane → binary image (just take the B_0 value (it will be 0 or 1) for each pixel)

There are 8 bit planes for a single image.
so separating the image by each of its bit plane is called Bit Plane Slicing.

It was found that only 4 bit planes ~~were~~ were necessary to form a proper form of the image (we can pass it into models for operations)

Its a form of compression → thus reducing size.

Bit Plane Slicing helps identify which all bit planes contribute most towards forming a proper characteristics ~~image~~ form of the original image.

Histogram Processing :-

Understand abnormalities in images, especially medical images

Range $[0, L-1]$ r_k - k^{th} gray level.

Function :- $h(r_k) = n_k$

↳ no: of occurences of r_k in an image intensity

n_k can be from '0' to the no: of pixels in the images

x-axis ⇒ r_k

y-axis ⇒ n_k

Q) Why histogram processing?

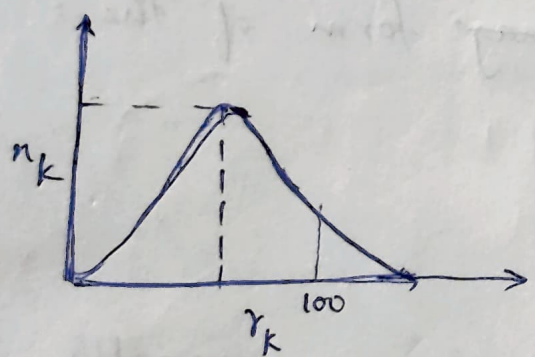
It is a dimensionality reduction technique, because it is a one dimensional of two projection of 2D image. It is the distribution of pixel values in that plane.

Normalized Histogram

Range = [0, 1]

Then $h(r_k) = p(r_k)$ → probability

$\therefore p(r_k) = \frac{n_k}{n}$ → no: of pixels



If pixel values are low, then the values are close to each other → low contrast

Histogram Equalization (uniform distribution)

It is a contrast enhancement technique / intensity transformation function. r is a gray level value.

r → [0, L-1]

0 → white

L-1 → black

$s = T(r)$

Satisfy 2 conditions :-

- T(r) should be single value and should be monotonically increasing.
  inverse transformation exist if single value and black to white progression is proved.
- T(r) should be within 0 to L-1

$$P(r_k) = \frac{n_k}{n} \rightarrow \text{pdf} \qquad s = T(r)$$

$$s_k = T(r_k) = \sum_{j=0}^{k} P(r_j) = \sum_{j=0}^{k} \frac{n_j}{n} \rightarrow \text{cdf}$$

cdf has a Linear distribution (uniform distribution)
↓
because the values are monotonically increasing.

∴ When r_k becomes s_k, it becomes uniform distribution, because cdf is linear distribution.

$$s_k^* = \text{int}\left[\left(\frac{s_k - s_{k\,min}}{1 - s_{k\,min}}\right)(L-1) + 0.5\right]$$

$s_{k\,max}$

Histogram Matching / Histogram Specification

Mapping a given distance to a reference distance
Shape of original pdf mapped to another reference pdf.

$$s_k = T(r_k) = \sum_{j=0}^{k} \frac{n_j}{n}$$

$$V_k = G(z_k) = \sum_{i=0}^{k} \phi(z_i) = s_k$$

$$z_k = G^{-1}(s_k)$$

Satisfy either

→ $(G(z_k) - s_k) = 0$

→ $(G(z_k) - s_k) > 0$

obtain histogram
obtain s_k value
obtain transformation G

compute values of z_k iteratively

1) $r_k \rightarrow s_k \rightarrow z_k$

finally

- Histogram useful in local enhancement, increase results not obtained in global enhancements
- Helps to extract statistical features from histogram. Helps compute n^{th} moment of ~~text~~ pixel r around its mean.

30/07/25

Enhancement

using Arithmetic / Logic Operation

1) ~~AND~~ AND, OR

these are used to perform Masking (to select certain regions)

ROI extraction can also be performed using these operators.

(Region of Interest)

It is also capable of providing colored marking at the specific regions.

2) Multiplication, Addition, Subtraction.

Addition (for reducing average intensity values)

(also additive multiplication – as it is repeated addition)

Subtraction for finding corresponding intensity value differences

$$g(x,y) = \underbrace{f(x,y)}_{\text{image obtained after contrast agent injection}} - \underbrace{h(x,y)}_{\text{anatomical structure}}$$ [Mask mode

(functional)

So $g(x,y)$ gives the blood flow information

3) Averaging

$$g(x,y) = f(x,y) + \eta(x,y)$$

In this equation, there is added ZWGN → White noise

(Zero-mean White Gaussian Noise)

It is an assumed noise distribution as we have no info

We assume it is un-correlated to the pixel values.
(white noise → energy levels of each color from the visible spectrum are the same)

If there are multiple $g(x,y)$

$$\bar{g}(x,y) = \frac{1}{k} \sum_{i=1}^{k} g_i(x,y)$$

then,

$$E\{\bar{g}(x,y)\} = f(x,y)$$

$$\sigma^2_{\bar{g}(x,y)} = \frac{1}{k} \sigma^2_{y}(x,y)$$

So when k increases, the variability of the pixel values decreases.

self-learn.
Fourier Domain filters, Ideal, Gaussian, } previous SEM

Homomorphic Filtering

Based on an important acquisition model → IR model
(Illumination Reflectance)

An image is formed by product of its illumination component and reflectance component.

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

the issue → cannot operate on the components separately as

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

could've been fine if it was addition

the range of $i(x,y)$ is still infinity

$$0 \leq i(x,y) \leq \infty$$

for $r(x,y)$ there will either be total adsorption (i.e. 0) or total reflection (i.e. 1)

So we have to convert it into an additive form

We can take log,

$$z(x,y) = \ln(f(x,y))$$

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

Now taking fourier transform,

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

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

Now we can apply

It will boost certain frequencies. (we can boost it)
components are in reflectance component, while low (attenuate/reduce it)
frequency components are in illumination component

$$S(u,v) = H(u,v) \cdot Z(u,v)$$

we need to come back to the original domain

$$s(x,y) = \mathcal{F}^{-1}[S(u,v)]$$

↳ this is enhanced $z(x,y)$

Now taking exponential

$$g(x,y) = e^{s(x,y)}$$

We assume it is un-correlated to the pixel values (white noise → energy levels of different colors ~~visible lights~~ from visible spectrum ~~have~~ are the same)

if there are multiple ~~no~~ $g(x,y)$

$$\bar{g}(x,y) = \frac{1}{K} \sum_{i=1}^{K} g_i(x,y)$$

then,

$$E\{\bar{g}(x,y)\} = f(x,y)$$

$$\sigma^2_{g(x,y)} = \frac{1}{K} \sigma^2_{\eta}(x,y)$$

so when k increases, ~~noise~~ the variability of the pixels decreases.

12/07/25

Estimating Degradation function

The process of restoring an image by estimating the degradation function. (process called blind ~~deconvolution~~ deconvolution)

$$g(x,y) = f(x,y) \circledcirc h(x,y) + \eta(x,y)$$

$g(x,y)$ — the img we have

$f(x,y)$ — we don't know

$h(x,y)$ — degradation function to be estimated

we need to find $\hat{f}(x,y)$ from $g(x,y)$

3 approaches

1) Estimation by Image Observation

Imp assumption :- the process is shift invariant

writing in terms of fourier domain

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

How do we ~~extract~~ observe?

→ Take a small portion of the given image ~~$G(u,v)$~~ $g(x,y)$ where the signal content is very ~~to~~ strong.

small portion ⇒ $g_s(x,y)$

Then blur this image to make it similar to the ~~the~~ other regions of the image (by smoothening or so)

~~→ Make it unblurred~~ This is to get info about the blur.

→ ~~From the equation~~: Make the sub image as unblurred as possible using sharpening or similar operations, ~~to ass~~ This is to find an ~~asso~~ assumed

$\hat{f}_s(x,y)$

→ From the equation:- Take the unblurred sub image ~~and~~ and find fourier of $g_s(x,y)$ and $\hat{f}_s(x,y)$ to get $G_s(u,v)$ and $F_s(u,v)$, then

$$H_s(u,v) = \frac{G_s(u,v)}{F_s(u,v)}$$

estimate ~~asso~~ ↓

$H(u,v)$

then we can find

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

2) Estimation by Experimentation

The experiment is acquisition.

So the condition is :- The image acquisition equipments should remain same throughout experiment to get some kind of value of h function (degradation)

If we have a bright spot of light, we can extract the strength of impulse (because fourier transform of an impulse is always a constant ∴ ⇒ A)

$$H(U,V) = \frac{G(U,V)}{A}$$

Then $$\hat{F}(U,V) = \frac{G(U,V)}{H(U,V)}$$

3) Estimation by Modeling (pure mathematical model equations)

eg: FOG Model (adds fog to image ∴ can control the severity)

Color Images

It has 3 basic quantity:-

1. Radiance :- Total amount of energy coming from an image.
2. Luminance :- Measure of amount of energy that an observer perceives from a light source. (related to human vision capabilities)
3. Brightness :- Difficult to measure.

There are 2 sets of colours:-

1. Primary colours → RGB
2. Secondary colours → CMY ⇒ when combined → black (CMYK) → black
   - cyan (B+G)
   - majenta (R+B)
   - yellow (R+G)

There are certain features that helps to distinguish the colours when talking in terms of human vision system (HVS)

1st parameter → Hue (true color)

2nd parameter → Saturation (amount of white light added mixed with hue)

The degree of saturation is inversely proportional to the amount of white light added.

Full colors are fully saturated.

3rd parameter → Brightness (only one not associated with color)

it depends on light source, perception.

These parameters help form an achromatic notation of colour.

for controlling and depicting the image according to our on prefrence we can manipulate these 3 parameters, but for that we need to convert RGB to Hue, Saturation, Brightness model (we are converting from chromatic to achromatic notation)

During compression, most manipulations are made on the colour, not on brightness.

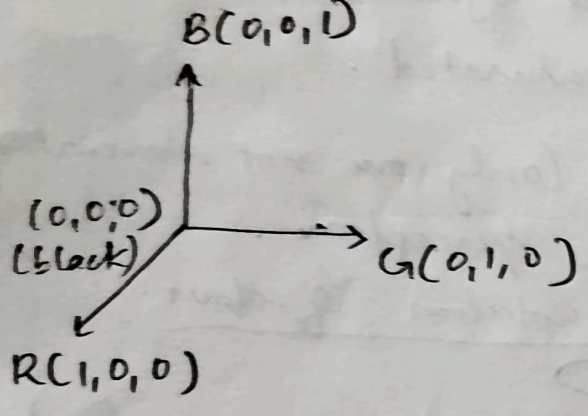
Color Models

RGB
CMY
CMYK

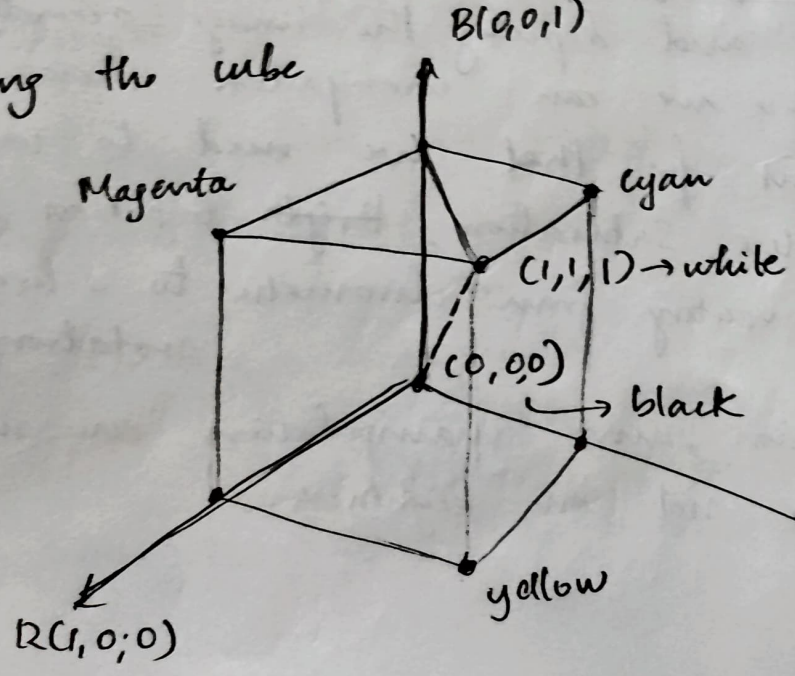
HSI → color components, Intensity
Most popular

In HSI we can apply filters on I and bring change in the whole image while keeping H and S intact.

Popular model that uses RGB representation → Color Cube



Drawing the cube



There is a line connecting the black point with the white point of it passes through the cube.

Usually 24 bits (8+8+8) are used ~~to represent~~ in an image.

How to plot ~~the~~ histogram for a colour image?

→ we can either plot 3 different histograms for R, G & B and observe them together.

But how to consider them together?

→ We can do this by using 6 bits, 2 bits for R, 2 bits for G, 2 bits for B. These are all used from the lower register (as at that range the colors don't look much different for our eyes) so the 6 bits basically represent a color combination

→ Or we can convert RGB → HSI and plot histogram using I info

$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

~~to~~

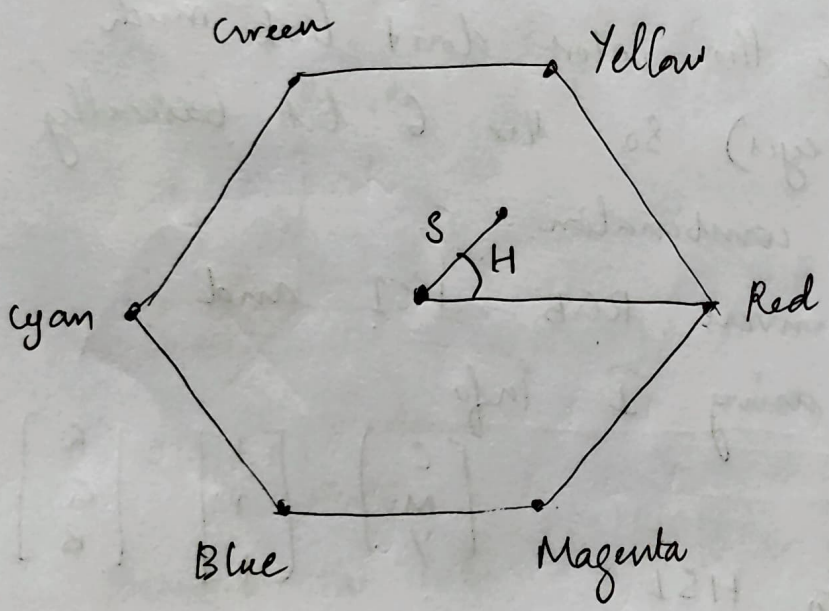
Converting RGB to HSI

We can use Color Cube for this purpose.

Consider a triangle (with one of its side as the dotted line connecting the black & white dot in the color cube) then all points in that triangular will have the same color → hue (saturation may vary). And all points in the dotted line → gray scale

HSI space is represented by a vertical axis (intensity axis?)

If we imagine a plane passing through this vertical axis, then it will interact with the color cube along the way.
This will be causing the formation of cross section areas of different shapes (popular one - Hexagon)



The dot on the center is through where the axis passes.

We then project this dot to a particular color (here Red)

Then mark the point that we want to convert to HSI, connect the point to the dot at center.

The angle formed represents H
The distance between the points represent S
And intensity → I

Full Color Image Processing

→ If we assume all 3 color channels are independent, we can apply on color channels independently → but it creates color artifacts.

→ Convert RGB → HSI.
And then operate on I (intensity component)

→ ~~Lets~~ Apply ~~on~~ directly on color components
But then we would have to deal with vector processing [R G B]. Scalar simplicity will be lost.

Image Segmentation

Image → [Image Segmentation Algorithm] → Segments

The ~~Segments have some simila~~ Segments → areas within the image that have common characteristics.

~~Those segments have certain attrib~~
The images have attributes and the segmentation ~~extract~~ divide accordingly ~~its~~ into the segments.

It the process of dividing an image into certain regions that share a certain common attributes.

Training of a segmentation algorithm stops once the Region of Interest is extracted.

How are Images segmented? on the basis of:

1) Similarity :- The pixels in segments share similar characteristics

2) Discontinuity :- Observed at regi portions where edges occur, or portions where color or intensity changes qui rapidly.

Let R be the spatial region of the image.
We have R divided into n subregions

$R_1, R_2, R_3, \ldots\ldots, R_n$

There are 5 characteristics that any segmentation algorithm must satisfy :-

a) $\bigcup_{i=1}^{n} R_i = R$

After segmentation, all regions when combined should reproduce the original image. (i.e. no single pixel is unlabelled towards any one reg region category.

b) R_i is a connected set. $(i = 1, \ldots, n)$

Connectivity is to be maintained within a given region. Eg: $R_1 + R_2 \neq R_3$. All regions must be defined accordingly.

c) $R_i \cap R_j = \phi$ (for all i,j where $i \neq j$)

The regions must be disjoint.

d) $Q(R_i) = \text{TRUE}$ $(i = 1, \ldots, n)$

↳ predicate we impose on the pixels

All the pixels within a region must respond true to the imposed predicate.

c) ~~Q(R_i ∩ R_j) = False~~

c) $Q(R_i \cup R_j)$ = FALSE

for any 2 adjacent regions R_i and R_j.

If this condition is not met then it means both R_i and R_j can be considered as a single region.

Shape Extraction using Hough - Transform

Straight line Extraction

Hough Transform is capable of extracting shapes even when there is discontinuity; it does so in linear computational time. (Global?)

Suppose there are n points, and we have to ~~extract~~ find all the possible straight lines.

Mathematically,

$\Rightarrow {}^{n}C_{r} = \frac{n(n-1)}{2} \approx n^2$ ~~$\approx n^3$ comparisons to be considered~~

Principle line → line that covers most points

n^3 → comparisons to be considered ~~wh~~ to find all possible principle lines.

Given 2 points i, j, straight line connecting them,

$y_i = ax_i + b$

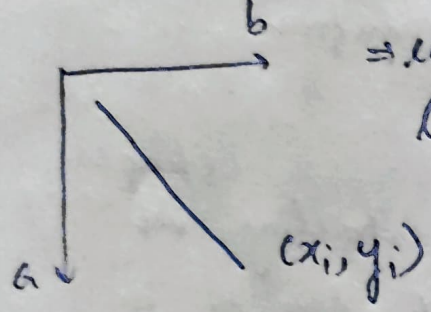
Hough Transform rewrites this equation to

(or space) (coordinate space)

shift the perspective from x-y plane to the a-b plane (parameter space).

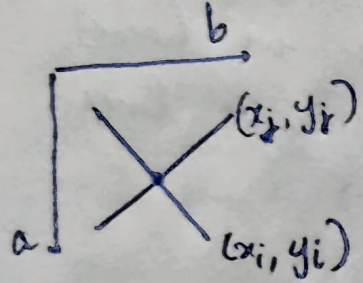
$$b = -x_i a + y_i$$

This would describe the family of lines that ~~could~~ passes through ~~(x_i, y_i)~~ the point (x_i, y_i)



⇒ each point in this line corresponds to a line

Consider another point (x_j, y_j)



this phenomenon can be used to find ~~optimal~~ principle line ~~the a-b~~

So we need to find the (a-b) plane that can ensure the maximum intersections.

But we can't use this equation for Hough Transform, because, when vertical lines occures $\theta = 0$

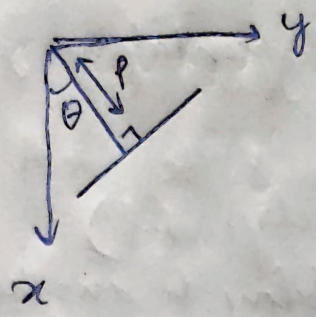
→ slope becomes infinite.

So new equation,

$$x \cos\theta + y \sin\theta = \rho$$

here we use $(\rho - \theta)$ space

How does it solve the issue,



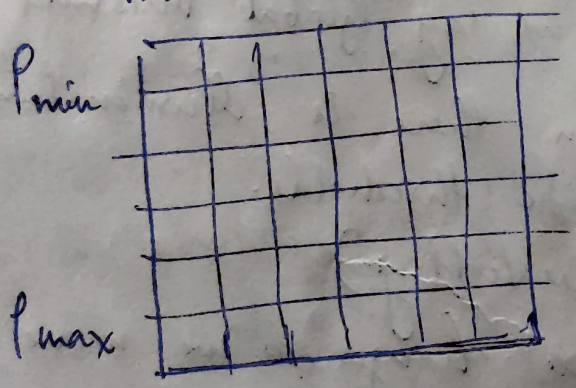
here when a vertical line occurs, θ value becomes 0. (slope)

- we are able to compute this in linear time as we are aware of the range

lines (θ) : $-90°$ to $90°$

distance (ρ) : $-D$ to D

we can form a matrix using this



for each point (x_i, y_i) we check the ~~list~~ through the values of this matrix.

After all points are ~~found accumulated~~ considered the cell that contains max value is the θ & ρ value we need.

Thresholding

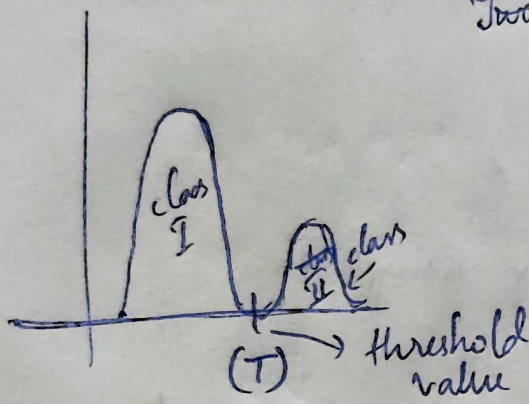
Based on — Also called Region-based segmentation.

based on property of similarity.

pixel
a Values till threshold belong to one class

while the pixel values outside threshold → belong to another class.

Ideal situation (Histogram)



Once we find the threshold value T, further processes are easy:-

$$g(x,y) = \begin{cases} 1, & f(x,y) > T \\ 0, & f(x,y) \le T \end{cases}$$

we have a "global threshold"

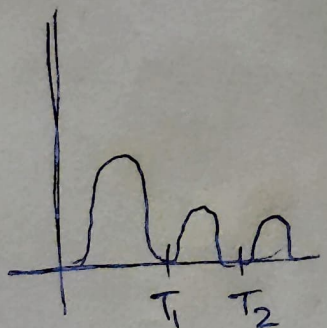
when T is constant, we say

Global threshold fails when there is "illumination", so we need region specific thresholds.

(variable thresholding)

↳ the value of T value changes as we move across the pixels.

another case



Now it takes high computational time, the possible combinations of T_1 and T_2 are huge

So o/p of a segmentation process → binary map

Success of thresholding depends on how well the peaks are separated.

Finding Global Threshold

- Set an initial T_0 value (can be between 0-255)
- Best to take T_0 as average value.
- There are 2 classes ⇒ C_1 and C_2
- C_1 contains all pixels greater than T
- C_2 contains all pixels are equal to or less than T.
- Find average intensity values for both classes:-

  $C_1 \Rightarrow m_1$; $C_2 \Rightarrow m_2$

- ~~Find the~~ Set the next T value as the average of m_1 & m_2. $T = \frac{m_1 + m_2}{2}$
- Repeat the ~~ab~~ steps above until the T value converges.

This is simple method when there is valley between 2 class peaks in our histogram. However this is an ideal case and not realistic, so we follow a statistical approach for that case.

Otsu's Thresholding (1979)

→ Also used in finding Global Threshold.
→ optimal Global Thresholding method.
→ objective:- minimize the avg error in assigning the ~~pix~~ pixels to 2 classes → based on

Bayes Decision Rule

→ If there multiple classes, the pixels classes try to maximize the variance between the class. (they values of different classes must be well separated)

Intensity values :- $0, 1, 2, \ldots, L-1$

Resolution :- $M \times N$

$n_i \rightarrow$ no: of pixels with intensity value i.

so,

$$n_0 + n_1 + n_2 + \ldots + n_{L-1} = M \times N$$

We use normalised histogram for this method.

$$P_i = \frac{n_i}{MN}$$

$$\sum_{i=0}^{L-1} P_i = 1 \quad ; \quad P_i \geq 0$$

Transformation, T,

$$T(k) = k \qquad (\because k \Rightarrow \text{threshold taken})$$

$$(0 \leq k \leq L-1)$$

The classes,

C_1 and C_2

$C_1 \rightarrow [0, k]$

$C_2 \rightarrow [k+1, L-1]$

$P_0(k)$ = probability that a pixel is assigned to class C_1.

$P_1(k) = \sum_{i=0}^{k} P_i$ { probability of class 1 occurring

$$P_2(k) = \sum_{k+1}^{L-1} p_i = 1 - P_1(k)$$

Finding mean intensity values associated with all pixels assigned to class 1.

$$m_1(k) = \sum_{i=0}^{k} i\, p(i|c_1)$$

$$= \sum_{i=0}^{k} i \frac{p(c_1|i)\, p(i)}{p(c_1)}$$

$$= \frac{1}{P_1(k)} \sum_{i=0}^{k} i\, p(c_1|i)\, p(i)$$

(∵ all i values are within k)

$$= \frac{1}{P_1(k)} \sum_{i=0}^{k} i\,(1)\, p(i)$$

$$= \frac{1}{P_1(k)} \sum_{i=0}^{k} i\, p(i)$$

Similarly $$m_2(k) = \frac{1}{P_2(k)} \sum_{i=k+1}^{L-1} i\, p(i)$$

$$m_G = \sum_{i=0}^{L-1} i\, P_i$$

(global mean)

$$P_1(k)\, m_1(k) + P_2(k)\, m_2(k) = m_G$$

We need a measure to find the best threshold value for 2 classes that maximizes the objective function (done by maximizing between class variance)

Between class variance (obj function)

$$\sigma_B^2 = P_1(m_1 - m_G)^2 + P_2(m_2 - m_G)^2$$

Optimal threshold will be,

$$\sigma_B^*(k^*) = \max_{0<k<L-1} \sigma_B^2(k)$$

k^* ↓ threshold

$$g(x,y) = \begin{cases} 1, & f(x,y) > k^* \\ 0, & f(x,y) \le k^* \end{cases}$$

Suppose there are k classes, we need to find multiple threshold.

$$\sigma_B^2 = P_1(m_1 - m_G)^2 + P_2(m_2 - m_G)^2 + \dots + P_k(m_k - m_G)^2$$

$$= \sum_{k=1}^{K} P_k(m_k - m_G)^2$$

If there are 3 classes,

$$\sigma_B^2 = P_1(m_1 - m_G)^2 + P_2(m_2 - m_G)^2 + P_3(m_3 - m_G)^2$$

we need to find k_1^* and k_2^*

$$\sigma_B^*(k_1^*, k_2^*) = \max_{0<k_1<k_2<L-1} \sigma_B^2(k_1, k_2)$$

finally,

$$G(x,y) = \begin{cases} a & f(x,y) > k_2^* \\ b & k_1^* \le f(x,y) \le k_2^* \\ c & f(x,y) \le k_1^* \end{cases}$$

Many evolutionary algorithms have been developed for this same problem. (gives best solutions in minimum time)

Assignment

Water-shed segmentation (hybrid method)

Optical Flow Equation

$$\psi(x+dx, y+dy, t+dt) = \psi(x,y,t) \longrightarrow (1)$$

Taylor Approximation

$$\psi(x+dx, y+dy, t+dt) = \psi(x,y,t) + \frac{\delta\psi(x,y,t)}{\delta x} + \frac{\delta\psi(x,y,t)}{\delta y} + \frac{\delta\psi(x,y,t)}{\delta t}$$

Stereo and Multi-view Sequence Processing

Each of our two eyes form different images of the object in front of us. The brain superimpose these 2 images. The difference between the 2 images decreases as the distance from eye to object ↑ increases.

When 2 cameras are used to capture an image of an object we call it Stereo Sequence Processing.

The difference between the images captured by 2 cameras is called disparity (horizontal difference between pixels)

Distance between cameras and object ⇒ depth

Distance between the cameras ⇒ baseline

focal length

more when zoomed in ⟶ more disparity

less when zoomed out ⟶ less disparity

when baseline decreases ⟶ disparity decreases

(as they are capturing a similar image)

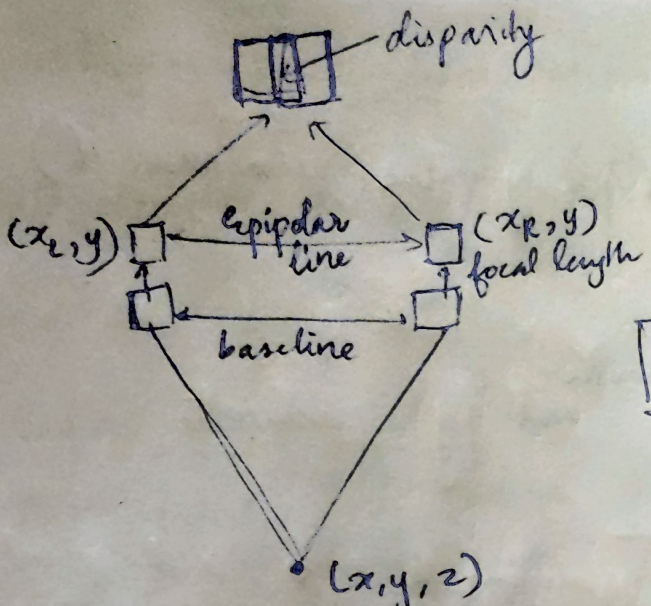
$$d = \frac{f \times B}{z}$$

d ⇒ disparity

f ⇒ focal length

B ⇒ baseline

z ⇒ depth



$$d = x_L - x_R$$

Assumption

If both the cameras are placed parallely, we found an object in image L to find the location of the same object in the image R we just need to search the along the epipolar line to find it. (we find location by searching for similar intensity regions)

Constraints

1) Epipolar Constraint
2) Unidirectional and parallel camera constraint
3) Ordering Constraint

Depth estimation step

- Capture 2 images
- Superimpose them and find the disparity for each pixel
- Find corresponding depth (z) using the equation

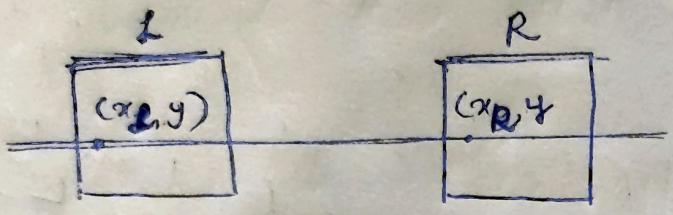
$$d = \frac{f \times B}{z}$$

Disparity Estimation

We need to optimize the disparity value (minimize it)

Error term,

$$E(d) = E_{data} + \lambda E_{smooth}$$



$$E_{data} = \sum_{x,y} \left[I_L(x,y) - I_R(x - d(x,y), y) \right]$$

There is a constraint, given a disparity value for a pixel, the disparity values of the neighbourhood pixels will be the same.

E_{smooth} =

Different Methods to Solve the Objective function

1) Block based Approach

optical flow computation
cov of head & tail
(next frame, prev frame)
Motion estimation
Depth estimation (Disparity)

Region - Based Segmentation

This depends on the predicate Q.
(changes with different cases - modalities, contrast, etc)

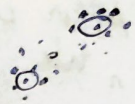
1) Region Growing Algorithm

The point from which the region grows - seed point

Prior knowledge helps in assigning initial location of seed points.

clusters (from each of it, ~~only~~ one point is selected as seed point)

These clusters are not the same as segmented



Stopping condition

There is not pixel that does not satisfy the predicate
(no pixel is to be added to any of segments we found)

~~Location~~ Original image space $\Rightarrow f(x,y)$

Binary images $\Rightarrow S(x,y)$

$Q_E \Rightarrow$ predicates to be applied on each pixel

1) Find all the connected components in $S(x,y)$

2) Use morphological process (erosion) on connected components are reduced to one pixel (labelled 1) and rest of the pixels are labelled 0.

3) We will form an img

such that $f_Q(x,y) = 1$

if the I/p img satisfies the predicate.

4) We get a new Binary image where with pixels with value 1 will be those that satisfy predicate.

5) Let G be an image formed by appending to each seed point, all the one value points of $f_Q(x,y)$

6) After convergence we can label connected components are segmented regions.

2) Region Splitting and Merging Algorithm

Consider R → image region

Q → predicate

1) Divide the region R into different quadrants.
2) If Q(R) is ~~not~~ false, we divide the region into 4 quadrants.
3) Then we find $Q(R_1)$, $Q(R_2)$, $Q(R_3)$, $Q(R_4)$ and apply similarly.
4) Further Adjacent regions where Q(R) is true is merged together.
5) The final output will give the segmented image.