Computer Vision

Assignment no: 2

Kinjal Kachi
CWID:
Course no - CS-512
fall 2020.

Note: ~~Q2b~~ & Q6e solved at the end.

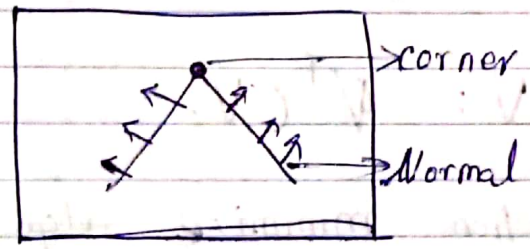
Q5 Q1

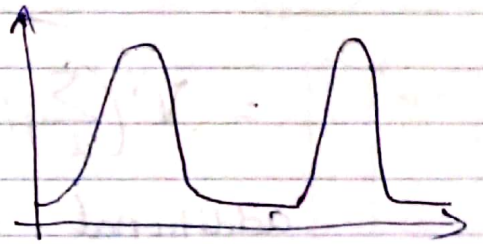
Sequence: Q1 all, Q2 all, Q3 all, Q5 all, Q6 all, Q4 all

Q1. Principle of Corner detection. How is number of principal directions assessed?

Principal directions assessment: Principle of corner detection:

Ans - If there is more than one direction of normal in orientation histogram then we can find corner
- Corners are more concise than edges.
- Corner is well localized.





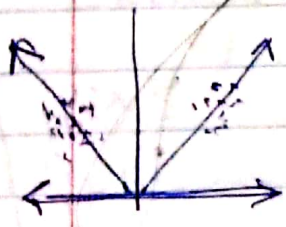
Normals are in all direcn.

Two peaks is because of two direcn of normal.

→ Corner Detection Summary:

1) Scan Image T-B, L-R, at each pixel select a local neighborhood
2) Build correction matrix :- $C = \sum_i g_i g_i^T$
3) Compute Eigenvalue of C.
4) Detect corner if $\lambda_1 \cdot \lambda_2 > T$.

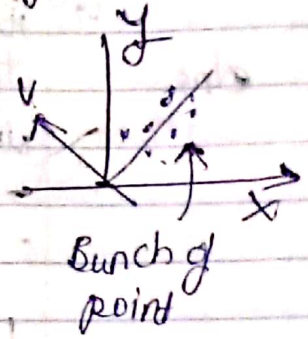
T = threshold. λ_1, λ_2 - principal directions.



Q2) PCA used to find principal directions of gradient orientations in a local patch.

→ Find direction V s.t. element projection of $\{g_i\}$ onto V is minimized.

$$E(V) = \sum_i (g_i \cdot V)^2 = \sum_i (g_i^T V)(g_i^T V)$$



$$= \sum_i (V^T g_i)(g_i^T V) = \sum_i V^T g_i g_i^T V$$

$$= V^T \left(\sum_i g_i g_i^T\right) V = V^T C V$$

additional directions minimize ~~algo~~ projection statement being orthogonal to previous directions.

$E(V)$ → objective function; V → direction

g_i → points

C → correlation matrix $= \sum_i g_i g_i^T$ and

$g_i = (x_i, y_i)$.

C → Correlation matrix.

C can be written as 2x2 matrix :-

$$C = \sum_i g_i g_i^T = \begin{bmatrix} \sum x_i^2 & \sum x_i y_i \\ \sum x_i y_i & \sum y_i^2 \end{bmatrix}$$

We can minimize $E(V)$ by taking gradient of $E(V)$ to zero

$$\nabla E(V) = 0$$

q3) Gradient vectors {(0,0), (0,1), (0,2), (0,3), (0,4), (1,0), (1,1), (1,2), (1,3)} compute correlation matrix for corner detect.

$g_i = \{(0,0); (0,1); (0,2); (0,3); (0,4), (1,0); (1,1); (1;2); (1,3)\}$

$g_i^T = \begin{bmatrix} 0,0 \\ 0,1 \\ 0,2 \\ 0,3 \\ 0,4 \\ 1,0 \\ 1,1 \\ 1,2 \\ 1,3 \end{bmatrix}$ $C = \sum_i g_i g_i^T \begin{bmatrix} \Sigma x_i^2 & \Sigma x_i y_i \\ \Sigma x_i y_i & \Sigma y_i^2 \end{bmatrix}$ $\begin{bmatrix} 0,0 \\ 0,1 \\ 0,2 \\ 0,3 \\ 0,4 \\ 1,0 \\ 1,1 \\ 1,2 \\ 1,3 \end{bmatrix}$

$\Sigma g_i g_i^T = [(0,0); (0,1); (0,2); (0,3); (0,4); (1,0); (1,1)(1,2)(1,3)]$

$\Sigma x_i^2 = 0^2+0^2+0^2+0^2+0^2+1^2+1^2+1^2+1^2 = 4$

$\Sigma x_i y_i = 0+0+0+0+0+0+1+2+3 = 6$

$\Sigma y_i^2 = 0^2+1^2+2^2+3^2+4^2+0^2+1^2+2^2+3^2$

$= 1+4+9+16+1+4+9$

$= 44$

$$C = \begin{bmatrix} 4 & 6 \\ 6 & 44 \end{bmatrix}$$

d) Condition on Eigenvalues of the gradient correla^n matrix that is used for corner detection.-

→ If $\lambda_1 \cdot \lambda_2 > T$
Which has two principal directions
→ If the eigenvalues are comparitively larger

Condition on Eigenvalues:

$E(v) = v^T C v$

$v^* = \underset{v}{\operatorname{argmin}} E(v)$

$\frac{\partial}{\partial v}(cv^2) = 2cv.$

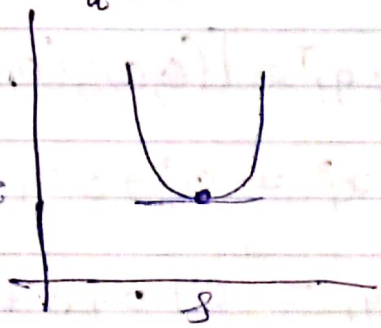
where
C = correlation matrix

$C = \sum_i g_i g_i^T$

$\nabla E(v) = 0$

$2cv = 0.$

↳ Solution is eigenvector belonging to smallest eigenvalue.



Eigenvalue = variance in corresponding principal direction

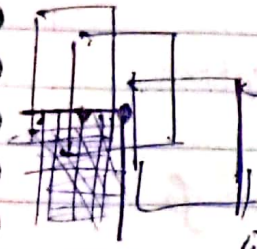
$\lambda_1 \cdot \lambda_2 > T$

e) Non-maximum Suppression for Corner detection.

→ To avoid detecting corners for multiple times non-maximum suppression is used. It is used to find threshold value as well.

→ Steps for Non-maximum Suppression

1) Compute $\lambda_1 . \lambda_2$ for all windows.
2) Select windows with $\lambda_1 \lambda_2 > T$ and sort in decreasing order.
3) Select the top of the list as corner, and delete all other corner in its neighborhood from the list.
4) Stop once detecting x% of the points as corners.
5) Overlapping windows are needed to account for corners between windows.
6) The analysis need to be done at multiple scales.



f) Explain how Harris corner detection avoids computing the eigen values of gradient correlation matrix.

⇒ 1) Compute correlation matrix C for window
2) Compute cornerness measure:

$$G(c) = \frac{\det(c)}{\lambda_1 . \lambda_2} - \frac{k\, tr^2(c)}{k . (\lambda_1 + \lambda_2)^2}$$

3) Detect corners where $G(c)$ is high.

$k \in [0, 0.5]$ is a user parameter.
Eg:- 0.04 - 0.15.
if $k = 0$ $G(c)$ detects corners
if $k = 0.5$ $G(c)$ detects edges.

$$C(C) = \det(C) - k\,tr^2(C)$$
$$= \lambda_1 \lambda_2 - k(\lambda_1 + \lambda_2)^2$$
$$= (1-2k)\lambda_1 \cdot \lambda_2 - k(\lambda_1^2 + \lambda_2^2)$$

$= 0$ if $k = 0.5$ Corner detection

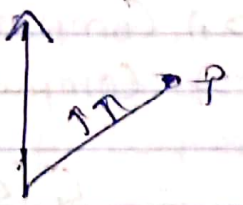
$= 0$ if $k = 0$ edge detection.

Here det(C) is product of eigen vectors & trace is sum of sum of eigenvectors Hence in this way Haveis corner detection avoids using eigen values & instead it uses determinant & trace of correlaⁿ matrix

g) Formula for computing letter localizaⁿ of a corner. ~~letter~~ Condition for the solution in the formula to exist?

⇒ Let's assume there is a window and we are trying to deted corner.

$x_i \rightarrow$ point
$n_i \rightarrow$ normal vector
$p \rightarrow$ divide point



If $\sum n_i \cdot (x_i - p) = 0$, it will be best localizaⁿ at corner.

To determine if p is the corner, connect each point x_i to p and project the gradient of x_i onto $(x_i - p)$.

$$E(p) = \sum_i \left(\nabla I(x_i) \cdot (x_i - p)\right)^2$$

$$= \sum (x_i - p)^T \nabla I(x_i) \nabla I(x_i)^T (x_i - p)$$

$$= \sum_i (x_i - p)^T \left(\nabla I(x_i) \nabla I(x_i)^T\right)(x_i - p)$$

$(x_i - p)$ should be $\perp$ n_i

$\nabla E(p) = 0$ gradient is zero.

$$p^* = \underset{p}{\operatorname{argmin}}\ E(p)$$

$$\sum_i \left(\nabla I(x_i) \nabla I(x_i)^T\right)(x - p) = 0$$

$$\sum_i \nabla I(x_i) \nabla I(x_i)^T p = \sum_i \nabla I(x_i) \nabla I(x_i)^T$$

$$p^* = \left(\sum_i \nabla I(x_i) \nabla I(x_i)^T\right)^{-1}$$

$$= \sum_i \nabla I(x_i) \nabla I(x_i)^T x_i$$

$$= C^{-1} \sum_i \nabla I(x_i) \nabla I(x_i)^T x_i$$

p^* → localization at corner.

C → correlaⁿ matrix

→ Condition for formula :-

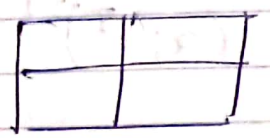
→ Equaⁿ should be linear and gradient should be equal to zero. If the equaⁿ is non-linear equation then we will not get the solution.

h) Ex

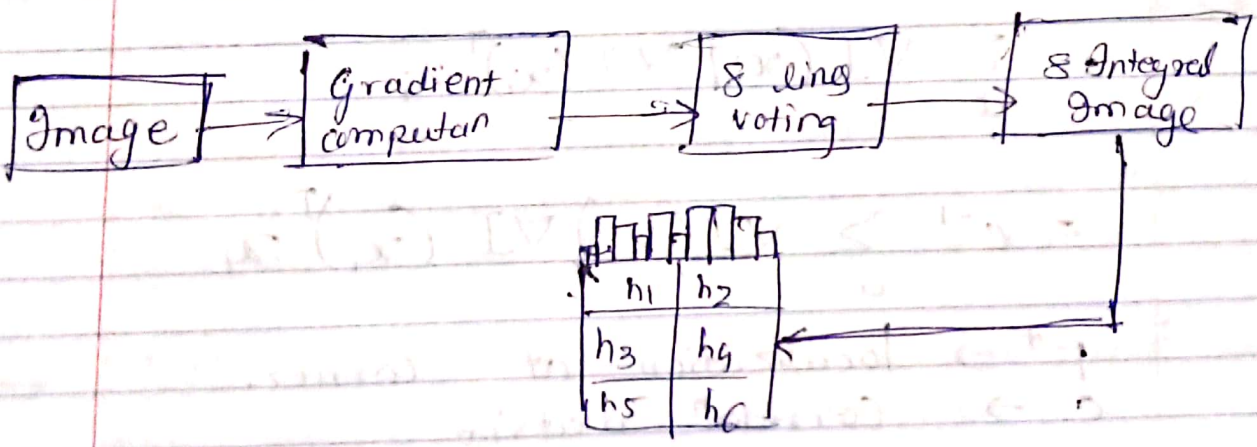
h) Explain how fast feature points can be characterized using HOG. What are requirements from a good characteriza" of feature points?

→ Part a) 1) Split each patch into cells, which may be overlapping.
2) Create orientation histogram in each cell by using either gradient directions or edge directions. Possibly weighted by distance from ~~corner~~ centre or gradient magnitude.
3) ~~Conte~~ Create orientation histogram

~~Eg~~



2X2 cell block of a patch.



* Requirements of good characteriza" of feature points

Local wind → Transla" in variance
Histograms → Rota" in variance
Pyramid → Scale in variance
Gradients → Illumination in variance

i) SIFT features

1) Use weighted sum to create orientation histograms in cells.
1) Create internal representations of the original image to ensure scale invariance.
2) Use LOG to find keypoints.
3) Eliminate bad keypoints (edges, low contrast regions,)
4) Compute an orienta^n for each keypoint
5) finally scale and rotation in place.

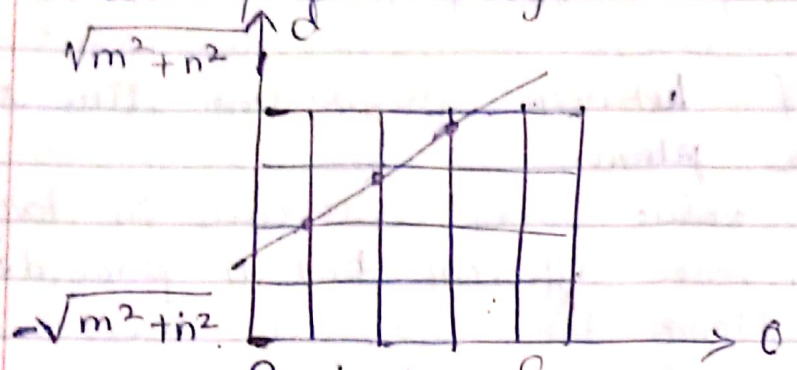
Q2) Line detection:-

a) Explain the problem of using slope and y-intercept as line parameters when using Hough x'form

⇒ $y = ax + b$

Here, is this equation the range of slope 'a' is a problem. We need to get value of 'a' to determine 'x'.
The value of 'a' can lie from anywhere between 0 to ∞. The maximum value of 'a' is infinity as we approach 90°. Thus, representing vertical lines is also a problem. The value of 'y' depends on value of 'a' which means an infinity value of 'a' will disrupt the value of 'y' intercept.

(c) Parameter space is an array. We can cast a vote inside cells. Everywhere the line would fall, the value increased by 1. Different points on the parameter space can be noticed. These cells are called as bins. The lines should pass through these line.

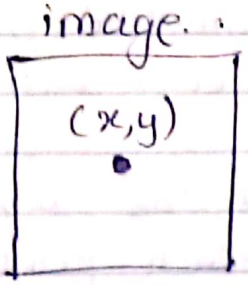


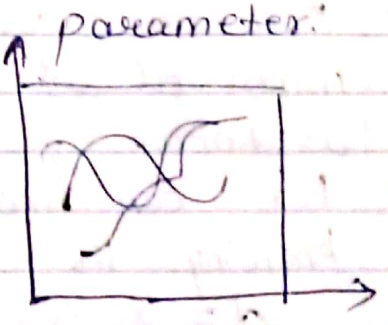
Parameter Space

The parameter vote for a point (x, y)

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

scan - $\theta \in [0, 360]$ and compute d





It scans all the θ ranges, and comes to different lines.

d) How are lines detected by checking parameter plane?

→ When different lines which represented points in image has a intersection point the distance and angle of that point, represent the line. Lines are detected in the parameter plane by taking the point where all the votes

intersed. The point defines the parameters; we can construct the lines are $ax+by+c=0$

Hence 1) Detect edges

2) Rap edge points to Hough space and store it.

3) Yield store points in lines of infinite length.

4) Convert infinite lines & find interception.

e) Trade off between regarding line size in parameter plane.

→ Fewer value votes are casts in larger bin so it is more efficient but it provides less localiza^. Hence the accuracy level is very low.

On otherhand, smaller bins provide more accuracy of vote but efficiency is less. This is known as trade off regarding bin size.

f) How can voting in the parameter plane be improved if the normal at each voting point is known.

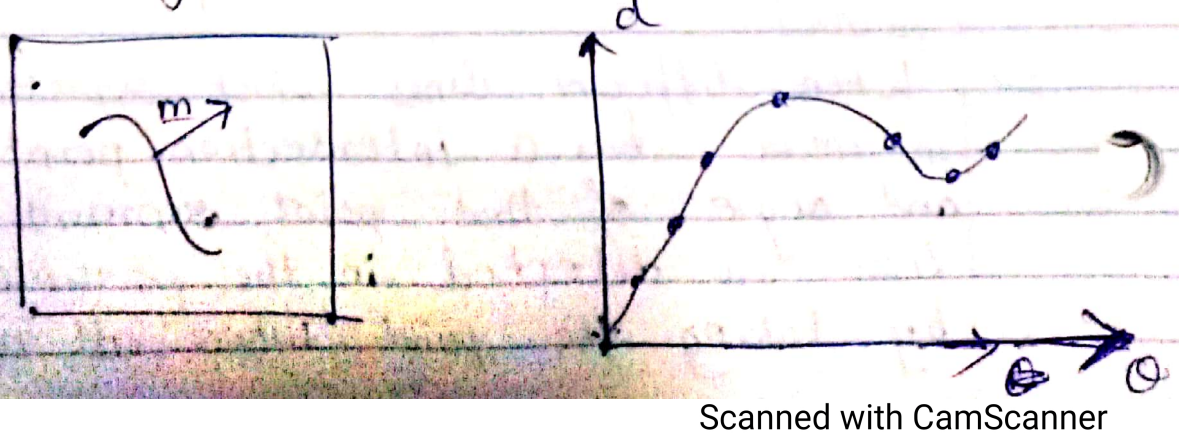
→ We can find the value of the angle θ by knowing the normal at each point.

$d = x\cos(\theta + \alpha\Delta\theta) + y\sin(\theta + \alpha\Delta\theta)$

where $\alpha \in [-1, 1]$

→ We cannow cost a single vote instead of multiple votes as we know θ.

image



We can get 'θ' by knowing the normal with means. We can get single vote for single point. In this way, voting is parameter space improved. We don't cost multiple votes of single point in parameter space. It saves time and increase efficiency as well.

If the normal ~~act~~ at each voting points is known to be more efficient, instead of $\theta \in [0, 180]$, take $\theta \in [\theta_{min}, \theta_{max}]$

9) When using Hough Transform for circles, what is the no. of dimensions of the parameter space?

→ ~~We can find the Nature of the circle by knowing the normal at each point.~~

→ The number of dimension of the parameter space is there while using Hough Transform

Circle is represented by

$(x-a)^2 + (y-b)^2 = r^2$

The parameters are a, b, r

It's 3D in parameter space.

(Q3) a) Disadvantages of $y = ax + b$ for line fitting (What kind of lines cannot be fitted accurately to this equation)

→ Disadvantage of using $y = ax + b$ for line fitting is that it is not suitable for vertical lines.

Vertical lines cannot be fitted properly by using $ax + b = y$. We get poor fitting when the angle approaches to 90° i.e $a = \infty$.

b) ~~Given a line with a~~

b) Normal (1, 2) and distance of 2 from the origin:-

$ax + by + d = 0$ → Line equations

(a, b) → normal co-ordinates

d → -ve distance from centre

$l = [1, 2, -2]$

c) How to fit a line using explicit line equa^n Write equa^n that has to be solved for the unknown line parameters.

→ Use explicit line equa^n to minimize geometric distance.

$l^T x = 0$, where all points 'x' on line 'l' should satisfy their objective function to get $E(l)$ minimized

$$E(l) = \sum_{i=1}^{n} (l^T x_i)^2$$

$$= l^T \left(\sum_{i=1}^{m} (x_i x_i^T) \right) l$$

$$E(l) = l^T C l$$

Here $l^* = \underset{l}{\operatorname{argmin}}\ E(l) = \nabla E(l) = 0$

$= 2Cl = 0$ ... ①

It will solve for eigen vectors for corresponding zero eigen values.

d) $\{(0,1); (1,3); (2,6)\}$. → Given points.

$C = \sum P_i P_i^T$ $\qquad P_i = (x, y, 1)$

$$D = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 3 & 1 \\ 2 & 6 & 1 \end{bmatrix}$$

$$\therefore D^T = \begin{bmatrix} 0 & 1 & 2 \\ 1 & 3 & 6 \\ 1 & 1 & 1 \end{bmatrix}$$

$$D^T D = \begin{bmatrix} 0 & 1 & 2 \\ 1 & 3 & 6 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 & 1 \\ 1 & 3 & 1 \\ 2 & 6 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 0+1+4 & 0+3+12 & 0+1+2 \\ 0+3+12 & 1+9+36 & 1+3+6 \\ 0+1+12 & 1+3+6 & 1+1+1 \end{bmatrix}$$

$$= \begin{bmatrix} 15 & 15 & 3 \\ 15 & 46 & 10 \\ 3 & 10 & 3 \end{bmatrix}$$

e) Write the explicit equations for conic curves. What is the constraint on the parameters a, b, c, d, e, f that guarantees that the model will be an ellipse?

→ Explicit axis aligned equation →

$$\left(\frac{x - x_0}{a}\right) + \left(\frac{y - y_0}{b}\right) = 1$$

For conic curves,

$$ax^2 + bxy + cy^2 + dx + ey + f = 0$$

$$l^T p = 0 \quad \text{where} \quad l = \begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \end{bmatrix}$$

$$p = (x^2, y^2, xy, x, y, 1)$$

for $b^2 - 4ac < 0$, the model will be an ellipse

f) Write equation that needs to be solved for fitting an ellipse using algebraic distance. Explain which points on the ellipse affect more the fitting (points close to the long axis or short axis of the ellipse.)

→ $E(l) = \sum_{i=1}^{n} (l^T p_i)^2$, where $p_i(x_i^2, x_i y_i, y_i^2, x_i, y_i, 1)$

To fit an ellipse using algebraic distance $\Rightarrow SL = 0$;
where $S \equiv \sum_{i=1}^{n} p_i p_i^T$.

Given the solution, it is eigenvector ~~corresponding~~ correspond to zero eigenvalue. The point which is closer to the short axis of ellipse have more effect on fittings as these point get more weight considering the algebraic distance of these are lesser than those closer to the long axis of the ellipse.

g) Geometric Distcn Objective: $E(l) = \sum_i \frac{|f(P_i - l)|}{|\nabla f(P_i - l)|}$

This formula is not a linear.
Formula, where $|f(P_i - l)| = (l^T P_i)^2$ and
$P_i (x_i^2, x_i y_i, y_i^2, x_i, y_i, 1)$.
Complication which is involved is, this doesnot result in a quadratic equation, so we don't get an explicit solution

h) Objective function of active contours:-

$$E[\phi(s)] = \int (\alpha(s) E_{continuity} + \beta(s) E_{curvature} + \gamma(s) E_{image}) ds.$$

$$E_{continuity} = \left|\frac{\partial \phi}{\partial s}\right|^2 ; \quad E_{curvature} = \left|\frac{\partial^2 \phi}{\partial s^2}\right|^2$$

$$E_{image} = -|\nabla I|^2$$

$\alpha(s)$, $\beta(s)$; $\gamma(s)$ are coefficients (variable).

$E_{continuity}$, $E_{curvature}$ and E_{image} are energy terms.
$\alpha(s) E_{continuity} + \beta(s) E_{curvature}$ is internal part.
$\gamma(s) E_{image}$ is external part.
We want $E_{continuity}$ & $E_{curvature}$ do be smaller & E_{image} to be high.

i) When the curve is discrete and using active contours:-

Econtinuity is estimated as distance between neighbouring points i.e $E_{continuity} = \sum_i |P_i - P_{i-1}|$

Ecurvature is estimated as difference of tangents at neighbouring points

$$E_{curvature} = \sum |(P_{i+1} - P_i) - (P_i - P_{i-1})|^2$$

$$= \sum |P_{i+1} - 2P_i + P_{i-1}|^2$$

ii) The continuity of active contours may be relaxed or to allow discontinuity, we find high curvature points as let $\beta_i = 0$ i.e if $|P_{i+1} - 2P_i + P_{i-1}| > T$ then $\beta_i = 0$

Q5] Robust Estimation :

a) Explain what are outliers, and is the fundamental problem associations with them where when fitting a model.

→ Outliers are pixels which values doesn't follow the ones from the rest of the pixels. It can be due to noise, occlusion or alignment error for example:-
The problem is that a single outliers can modify the estimation and make it worse for the rest of the point.

which is more sensitive to outliers.

c) Geman-Mcclure function: $\rho_\sigma(x) = \frac{x^2}{x^2+\sigma^2}$.

It advantage is that it puts a limit of 1 to the error for outliers.
If we choose a large σ we way include outliers but if it is too small we way not include enough points so we can estimate σ as $\sigma = 1.5 \text{ med}(d(x_i; \theta_n))$.

e) RANSAC is an iterative method to estimate the parameters of a mathematical model from a dataset that contains outliers. It provides a reasonable result only with a certain probability that increases with the number of iterations

b) The parameters of the RANSAC are:-

n= minimum number of points at each evaluation

d= minimum number of points needed.

k= number of trails.

t = distance to determine outliers,

w = probeblity that a point is a inlier

b) $E(\theta) = \sum_{i=1}^{n} \rho_\sigma(d(x_i(\theta))$

In robust estimation $\rho_\sigma(x) = \frac{x^2}{x^2 + \sigma^2}$

In the standard least squares objective function, outliers will have higher value and ~~out~~ influence mode(~~#~~ However, in robust estimation $\rho_\sigma(x) = \frac{x^2}{x^2 + \sigma^2}$ will lower the influence of outlier.

Q6] a) Objective of image segmentation is to separate the background and foreground of the image. find contours of object and label each pixel in the image with class label.

b) Difference between Agglomerative (merge) and divise (split) approaches :-

↳ Agglomerative Segmentation:

→ Start with each pixel in a separate cluster.
→ Merge clusters with small distance.
→ Repeat while clusters are not satisfactory

↳ bivisive segmentation:-

→ Start with all pixels in the cluster
→ Split clusters to produce large distance between them.
→ Repeat while clusters are not satisfactory.

c) K-means algorithm for clustering pixels with spatial context.

→ Select k. k = no. of clusters.
- Select initial guess of k-means.
- Repeat: take each pixels, :-

$$l_i = \underset{j \in [1,k]}{\operatorname{argmin}} \|f_i - m_j\|^2$$

$$S_j = \{i \mid l_i = j\}$$

$$m_j = \frac{\sum_{i \in S_j} f_i}{\# S_j}$$

mean of points in cluster m_j

S_j => collection of points in cluster.

d) Difference between Graph cut and Normalized cut? Why is normalized cut necessary

⇒ In Graph cuts :- images are represented using graphs. Pixels are nodes. Each pixel is connected to all pixels. The weight on the link between nodes p and q represent the similarity between them (intensity + location)

$w_{pq} = \exp(-\|f(p) - f(q)\|)$

$f(p)$ similar to $f(q) \rightarrow w_{pq} = 1$

$f(p)$ not similar to $f(q) = w_{pq} = 0$

Cost of choosing cut :-

Choose cut where cost is smaller :-

$$\text{cut}(A,B) = \sum_{p \in A, q \in B} w_{p,q}$$

↳ delete links between nodes with weak links which is the price of the cuts.

Problem with Graph cuts:

↳ ~~cut~~ creates min cuts, where weight of cut $\propto$ to no. of edges into cut.

↳ Also it tends to produce small isolated components

Hence we go for Normalized cuts

* Normalized cuts fixes bias to isolate components :-

$$N\text{cut}(A,B) = \frac{\text{cut}(A,B)}{\text{assoc}(A,V)} + \frac{\text{cut}(A,B)}{\text{assoc}(B,V)}$$

normalizing elements

assoc (A, V) = Sum of weights of all edges that touch A.

Approximate solution for minimizing N cut :-
Generalized Eigenvalue Problem

⇒ W/P ⇒ adjacency matrix
D ⇒ Dic

The weighted degree d_i of a node is sum of all edges connected to it :-

$$d_i = \sum_{j=1}^{n} w_{ij}$$

w_{ij} = similarity of node i to j

$$W = \begin{bmatrix} w_{11} & \cdots & w_{1n} \\ \vdots & & \vdots \\ w_{n1} & & w_{nn} \end{bmatrix}$$ similarity matrix

$$D = \begin{bmatrix} d_1 & & \\ & \ddots & \\ & & d_n \end{bmatrix}$$ weighted degree at each node.

$$N\text{cut} = \frac{\text{cut}(A,B)}{\text{vol}(A)} + \frac{\text{cut}(A,B)}{\text{vol}(B)}$$

convert to ⇒ $\min_y \frac{y^T(D-W)y}{y^TDy}$ s.t $y^TD\mathbb{1} = 0$

$$(D-W)y = \lambda Dy$$

⇒ Solution :- take second smallest eigen values, because 1st eigen value will be zero

f) How can we specify a cut using a vector whose length is number of nodes

→

$$\text{cut}(A,B) = \sum_{p \in A,\, q \in B} w_{p,q}$$

where $w_{p,q}$ → weight between nodes

g) Optimization problem that has to be solved to determined a minimum cut

→ Minimum cuts are cut with lowest cost to delete the weak links compared to other links thus separating the clusters

→ Problem here is minimizing the cost of the cut will yield cuts of a single node.

h) Solution to Normalized cuts using continuous variables.

$$N\text{cut}(A,B) = \frac{\text{cut}(A,B)}{\text{vol}(A)} + \frac{\text{cut}(A,B)}{\text{vol}(B)}$$

$$\text{vol}(A) = \sum_{p \in A,\, q \in B} w_{p,q}$$

Weighted degree d_i of a node is the sum of all edges connected to it:-

$$d_i = \sum_{j=1}^{n} w_{ij}$$

$$W = \begin{bmatrix} w_{11} & w_{1n} \\ w_{n1} & w_{nn} \end{bmatrix}$$

↳ similarity matrix

$D = \begin{bmatrix} d_1 & & \\ & \ddots & \\ & & d_n \end{bmatrix}$ → weighted degree at each node

Defining a cut :-

$$N_{cut} = \frac{\sum_{\substack{x_i>0 \\ x_j<0}} (-w_{ij} x_i x_j)}{\sum_{x_i>0} d_i} + \frac{\sum_{\substack{x_i>0 \\ x_j<0}} (-w_{ij} x_i x_j)}{\sum_{x_j<0} d_j}$$

$$k = \frac{\sum_{x_i>0} d_i}{\sum_i d_i} \qquad b = \frac{k}{1-k} = \frac{\sum_{x_i>0} d_i}{\sum_{x_i<0} d_i}$$

$$y = (1+x) - b(1-x)$$

Convert :- $N_{cut}(x) = \min_y \frac{y^T(D-W)y}{y^T D y}$

where y is discrete → move to continuous.

:- Solving $\min_y \frac{y^T(D-W)y}{y^T D y}$ s.t $y^T D \mathbb{1} = 0$.

$\Rightarrow (D-W)y = \lambda D y$

$\Rightarrow$ The first eigenvector ($\lambda = 0$) is $\mathbb{1} = [1, \ldots 1]$

Hence second smaller eigenvector is solution

i) Eigenfaces faces approach & limitation

→ map image (eg: 100 x 100) to lower dimensional vectors using PCA

→ measure similarity to templates in lower dimensional space

Limitation:- less sensitive to small variations.

ii) Bag of words for object recognition & limitation

→ Extract features. (Use SIFT of HOG

→ Cluster features to create a codebook (dictionary)

- Compute a distribution of codewords in each class
- Classify using distribution of code words

Limitation :- The classifier looks at different local neighborhood but doesnot look at the relations between them

Q4) Model fitting :

(a) Given gradient vectors are (1,2) & (3,4)

Correlation matrix is given by

$$C = \begin{bmatrix} \Sigma x_i^2 & \Sigma x_i y_i \\ \Sigma x_i y_i & \Sigma y_i^2 \end{bmatrix}$$

$$C = \begin{bmatrix} 1+9 & 2+12 \\ 2+12 & 4+16 \end{bmatrix} = \begin{bmatrix} 10 & 14 \\ 14 & 20 \end{bmatrix}$$

b) Slope $= \frac{y_2 - y_1}{x_2 - x_1} = \frac{10}{10} = 1$

Normal = (1, 1)

Normalized normal = $\left(\frac{-1}{\sqrt{2}}, \frac{-1}{\sqrt{2}}\right)$

$d = (n_x, n_y)(x, y)$

$= \frac{-10}{\sqrt{2}}, \mathcal{F}\left[\left(\frac{-1}{\sqrt{2}}\right)\left(\frac{1}{\sqrt{2}}\right)\right] (10, 10)$

$= f$

Implicit line eqn :-

$n_x x + n_y y - d = 0$

$\therefore \frac{-x}{\sqrt{2}} + \frac{y}{\sqrt{2}} = 0$

$-x + y = 0 \quad \therefore x = y$

c) line coefficients = (1, 2, 3)

$a = 1, \quad b = 2, \quad c = 3$

line equations:- $ax + by + c = 0$

$x + 2y + 3 = 0$

$x = 2$

$2y = -5 \quad \therefore y = \frac{5}{2}$

d) point $(1, 1)$ $\theta = 0$.

$\therefore$ value $d = x\cos\theta + y\sin\theta$

$= 1\cos(0) + y(\sin(0))$

$= 1$

e) $P_1 = (1, 2)$; $P_2 = (3, 4)$

Homogeneous pts : $(1, 2, 1) : (3, 4, 1)$

dimension $n = 2$

$$S = \begin{bmatrix} 1+9 & 2+2 & 1+3 \\ 2+12 & 4+16 & 2+4 \\ 1+13 & 2+4 & 2 \end{bmatrix} = \begin{bmatrix} 10 & 14 & 4 \\ 14 & 20 & 6 \\ 4 & 6 & 2 \end{bmatrix}$$

f) For an implicit curve f & point p, value of f at point p is 1

Gradient of f closest point to p is 2

$|f(p)| = 1$ & $|\nabla f(x^*)| = 2$

Distance

$d(p, f) = |p - x^*|$

$= \frac{f(p)}{\nabla(f(x^*))} = \frac{1}{2}$

g) for same assumption we have, above eg; approximated algebraic distance of point P as :-

$$d(p,f) = \frac{|f(p)|}{|\nabla (f(x)^*|} = \frac{1}{2}$$

h) $P_1 = (1,2)$
$P_2 = (2,3)$
$P_3 = (3,4)$

Continuity $= |P_3 - P_2|^2$

$= |(3,4) - (2,3)|^2$

$= |(1,1)|^2$

$= (\sqrt{2})^2 = 2$

Curvature $= |P_3 - 2P_2 + P_1|^2$

$= |(3,4) - 2(2,3) + (1+2)|^2$

$= 0$

ii) To guarantee high fitting of an active contour the β coefficient is given as follow :- $\underline{\beta = 0}$

(Q2) $\theta = 45$; intercept $= 10$

Slope $= \tan\theta = \tan 45 = 1$

Slope $= \tan\theta = \tan(45) = 1$

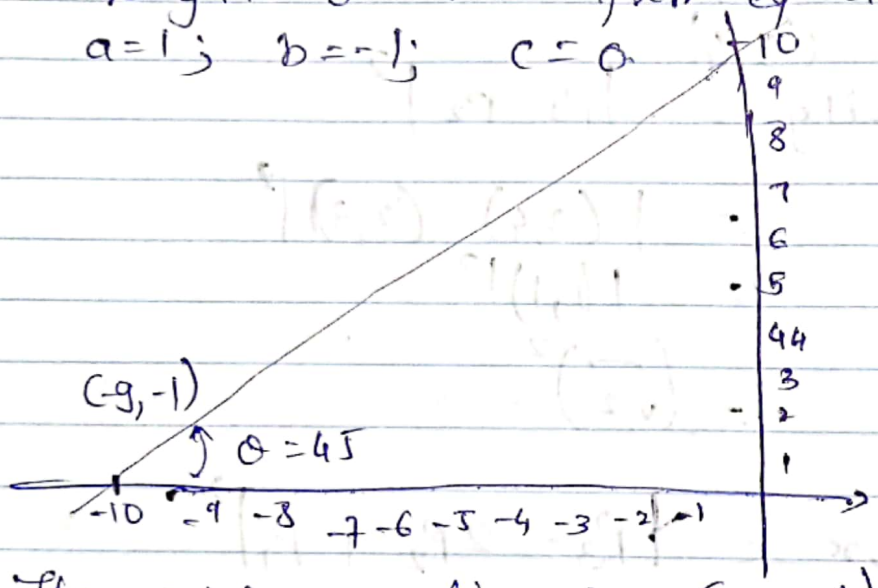
Slope point for : $y = m + b$

$y = x + c$.

line equation : $ax + by + c = 0$

$x - y + 10 = 0$ ... from eq^n (1)

$a = 1$; $b = -1$; $c = 0$



The point on line is $(-9, -1)$. Using it in eq^n :-

$(-9) - 1 + 10 = 0$

$0 = 0$

Satisfies eq^n

$a = 1$, $b = 1$ $c = 10$.

Q6e) A graph with 100 nodes have following:

Similarity matrix

$$\begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1,100} \\ w_{21} & w_{22} & & \vdots \\ \vdots & \vdots & & \\ w_{100,1} & w_{100,2} & & w_{100,100} \end{bmatrix}_{100 \times 100}$$

w_{ij} is similarity btw^n node i & j

- Degree matrix:

$$\begin{bmatrix} d_1 & & & \\ & d_2 & & \\ & & d_3 & \\ & & & \ddots \\ & & & d_{100} \end{bmatrix}$$

d_i = weighted degree of node i in graph.

- Laplacian Matrix

It is given as $L = D - W_{100 \times 100}$

where,

D = degree matrix

W = similarity matrix.