Computer Vision

Unit I

Image-based Feature Extraction

Introduction to Computer Vision and the challenges; Image-based feature extraction: Thresholding techniques - region growing methods - Thresholding - Adaptive Thresholding - Approaches to threshold selection - global valley approach to thresholding; Edge detection: Differential gradient operator - hysteresis thresholding - canny operator - Laplacian operator - active contours - level set - Graph cut approach; Corner and Interest point detection: Second order derivative schemes - Harris Interest Point operators - local invariant feature detectors and descriptors; texture.

* Introduction to Computer Vision and the Challenges

→ Computer vision is the science and technology of machines that see.

→ It deals with the theory for building artificial systems that obtain information from images.

→ The image data can take many forms such as video sequences, depth images, views from multiple cameras, or multi-dimensional data from a medical scanner.

* Components of a CV System

(i) Camera (iii) Scene

(ii) Lighting (iv) Computer for Scene Interpretation

* Applications of Computer Vision

(i) Optical character recognition (OCR)

(ii) Face detection

(iii) Smile detection

(iv) Vision - based biometrics

(v) Special effects - shape and motion capture

(vi) Industrial robots & mobile robots

* Vision as a source of semantic information

→ At a high level, the objects in the image can be identified, and the scene and context can be understood.

→ However, at a lower level, only simpler patterns like slant, horizontal/vertical lines, presence of some object can be identified.

* Challenges in Computer Vision

1. Viewpoint Variation
2. Illumination
3. Scale
4. Object Deformation
5. Occlusion
6. Background Clutter
7. Object intra-class variation
8. Local ambiguity

* Image-based feature extraction - Thresholding Techniques

Thresholding – The demarcation of objects in digital images (called segmentation) can be approximated by the process of thresholding. This involves separating the dark and light regions of the image.

* Region - Growing Methods

→ Pixels of like intensity are successively grouped together to form larger and larger regions, until the whole image has been segmented.

→ Rules have to be made about not combining adjacent pixels that differ too much in intensity, but at the same time should permit combinations for which intensity changes gradually due to variations in background illumination.

→ The technique has to include the facility to not only merge regions but also to split them if they become too large and inhomogenous.

→ Region growing methods are iterative in nature, gradually refining hypotheses about which pixels belong to which regions — involves local and global operations

@ Challenges

① Noise and sharp edges and lines form disconnected boundaries — and it is difficult to decide whether they form true region boundaries.

② Each pixel intensity has to examined multiple times, and the process is thus computationally intensive.

* Single-Value (Simple) Thresholding

→ Mathematically represented as:

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

If too low — loss of edges, lines, useful info. is lost

If too high — presence of noise, unnecessary artifacts

* Global Thresholding

→ based on the histogram of an image.

→ Analyze the histogram of intensity levels in the digitized image - if a significant minimum is found, it is interpreted as the required threshold value.

Algorithm

The basic global threshold, T, is calculated as follows:

1. Select an initial estimate for T (typically the average grey level in the image)

2. Segment the image using T to produce two groups of pixels:

   G_1 consisting of pixels w/ grey levels $> T$

   G_2 consisting of pixels w/ grey levels $\leq T$

3. Compute the average grey levels of pixels in G_1 to μ_1 and G_2 to give μ_2.

4. Compute the new threshold value : $T = \frac{\mu_1 + \mu_2}{2}$

5. Repeat steps 2-4 until the difference in T in successive iterations is less than a predefined limit T_∞.

Challenges - (i) works only with bimodal histograms

(ii) the valley may be so broad that it is difficult to locate a significant minimum

(iii) noise within the valley may inhibit location of the optimum position

(iv) may not have any clear visible valley because of excessive noise or because of varying background lighting

(v) major peak in the histogram may be larger than the other - and this will then bias the position of the minimum

(vi) histogram may be inherently multimodal - making it difficult to determine which the relevant thresholding level is.

* Tackling the Problem of Bias in Threshold Selection

→ Bias in the selection of thresholds arises when one peak in the histogram is larger than the other.

→ Bias can be tackled in the following ways:

(i) weight down extreme values, and weight up the intermediate values to achieve a better intensity distribution.

(ii) find positions in the image where there is a significant intensity gradient, corresponding to pixels in the regions of edges and analyze the intensity values only along these locations, while ignoring the other points in the image (make scatter grams - intensity variation vs. intensity gradient magnitude variation)

* Adaptive Thresholding

→ Adaptive thresholding permits the threshold to vary dynamically over the whole image. The different ways of achieving this include:

(i) model the background within an image

(ii) work out a local threshold value for each pixel by examining the range of intensities in its neighborhood

(iii) split the images into sub-images and deal with them independently?.

* Approaches to Threshold Selection

→ Global threshold selection can be done using 3 approaches

A. Variance-based Thresholding

B. Entropy-based Thresholding

C. Maximum-likelihood Thresholding

A. Variance-based Thresholding?

→ The image intensity histogram is analyzed to find where it can best be partitioned to optimize criteria based on the ratios of the within-class, between-class and total variance.

→ Otsu's Thresholding to calculate the between-class variance is as follows:

1. Let the image have a grayscale resolution of L gray levels.
2. The no. of pixels with gray level i is written as n_i
3. The total no. of pixels in the image is:

$$N = n_1 + n_2 + \dots n_L$$

4. Thus, the probability of a pixel having a gray level i is:

$$p_i = \frac{n_i}{N}$$

5. The between-class variance is:

$$\sigma^2_B = \pi_0 (\mu_0 - \mu_T)^2 + \pi_i (\mu_1 - \mu_T)^2$$

where $\pi_0 = \sum_{i=1}^{k} p_i$ $\pi_i = \sum_{i=k+1}^{L} p_i = 1 - \pi_0$

$$\mu_0 = \sum_{i=1}^{k} \frac{i p_i}{\pi_0} \qquad \mu_1 = \sum_{i=k+1}^{L} \frac{i p_i}{\pi_i}$$

$$\mu_T = \sum_{i=1}^{L} i p_i$$

6. The total variance is:

$$\sigma^2_T = \sum_{i=1}^{L} (i - \mu_T)^2 p_i$$

→ For a single threshold, the ratio to be maximized is the ratio of the between-class variance to the total variance:

$$\eta = \frac{\sigma^2_B}{\sigma^2_T}$$

B. Entropy-based Thresholding

→ The entropy statistic is high if a variable is well distributed over the available range - and low if it is well-ordered & narrowly distributed.

→ Entropy is a measure of disorder - and is zero for a perfectly ordered system.

→ Entropy thresholding involves finding the threshold at an intensity for which the sum of the entropies of the 2 intensity probability distributions is maximized.

→ The most appropriate threshold level is the one that imposes the greatest order on the system.

Method : The intensity probability distribution is again divided into 2 classes - those with ~~interest~~ gray levels upto the value k, and those w/ intensity values above k.

→ The 2 probability distributions A & B are:

→ capital

$$A : \frac{p_1}{P_k} \quad \frac{p_2}{P_k} \quad \cdots \quad \frac{p_k}{P_k} \qquad \text{or} = P_k$$

$$B : \frac{p_{k+1}}{1 - P_k} \quad \frac{p_{k+2}}{1 - P_k} \quad \cdots \quad \frac{p_L}{1 - P_k}$$

$$P_k = \sum_{i=1}^{k} p_i \qquad 1 - P_k = \sum_{i=k+1}^{L} p_i$$

→ The entropies for each class are given by;

$$H(A) = -\sum_{i=1}^{k} \frac{p_i}{P_k} \ln \frac{p_i}{P_k}$$

$$H(B) = -\sum_{i=k+1}^{L} \frac{p_i}{1 - P_k} \ln \frac{p_i}{1 - P_k}$$

The total entropy is:

$$H(k) = H(A) + H(B)$$

c. **Maximum Likelihood Thresholding**

→ The threshold value is determined by maximizing the likelihood that the observed pixel intensities belong to either the foreground or background.

→ This method assumes that the pixel intensities within each class follow a specific probability distribution, typically assumed to be a Gaussian distribution.

→ The algorithm iterates through the possible threshold values and computes the likelihood of observing the pixel intensities given the assumed probability distribution of each class. The threshold that maximizes the likelihood is the optimal threshold.

* Global Valley Approach to Thresholding

Objective: To look for the most significant valley in an intensity distribution, corresponding to an optimum discriminating point between dark objects & a light background in the original image

→ To judge the global valley deepness, consider any potential global valley point (called point).

→ Look at all points i to the left of it to find the highest peak to the left and all the points k on the right of it to find

the highest peak to the right.

→ Hence the maximum over all the points over i and max. of over all points k must be taken.

→ This must be done for all potential valley points j and only the points $i < j$ and $k > j$ need to be considered.

→ To avoid complications from -ve heights, use the sign function

$s(\cdot)$ s.t $s(u) = u \rightarrow u > 0$

$s(u) = 0 \rightarrow u \leq 0$

∴ The fn is:

$$F_j = \max_{i,k} \left\{ \frac{1}{2} \left[s(h_i - h_j) + s(h_k - h_j) \right] \right\}$$

→ This method might be complicated If there are many peaks and valley.. ∴ In the global valley method (GVM), the geometric mean can be applied:

$$\kappa_j = \max_{i,k} \left\{ \left[s(h_i - h_j)\, s(h_k - h_j) \right]^{1/2} \right\}$$

* Edge Detection

* Edges - edges are those places in an image that correspond to object boundaries

- edges are pixels where image brightness changes abruptly
- It is a vector variable (magnitude of the gradient, direction of an edge)

* Intuition behind edge detection

→ Edge information in an image is found by looking at the relationship a pixel has with its neighbors.

→ If a pixel's gray-level value is similar to those around it, there is probably not an edge at that point

→ If a pixel has neighbors with widely varying gray levels, there is probably an edge at that point.

* Use of Edge Detection

→ Important features can be extracted from the edges of an image - like corners, lines and curves.

→ These features are used by higher-level computer vision algorithms - eg. for recognition

* Edge Detection Methods

(i) can be implemented using a convolution mask

(ii) Differential operations measure the rate of change in the image brightness function.

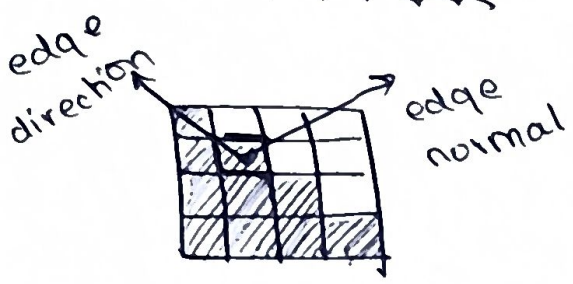
(iii) can also be based on orientation information.

* Edge Descriptors

A. Edge Direction - ⊥r to the direction of maximum intensity change

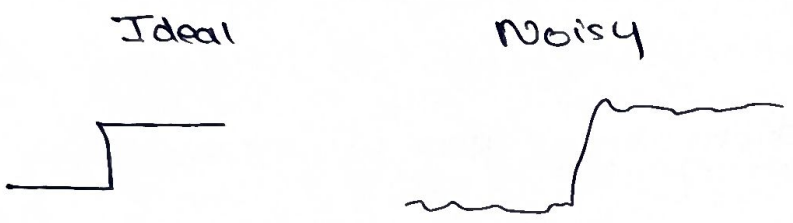
B. Edge Strength - related to the local image contrast along the normal

c. Edge position - the image position at which the edge is located.

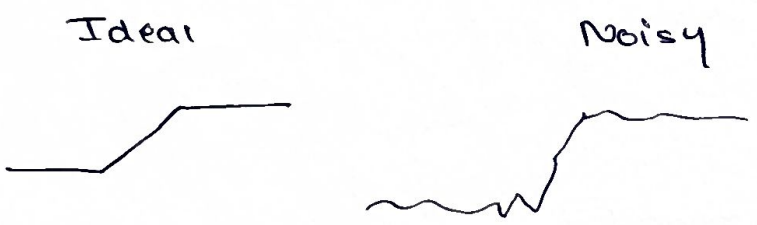


* Modeling Intensity Changes – Types of Edges

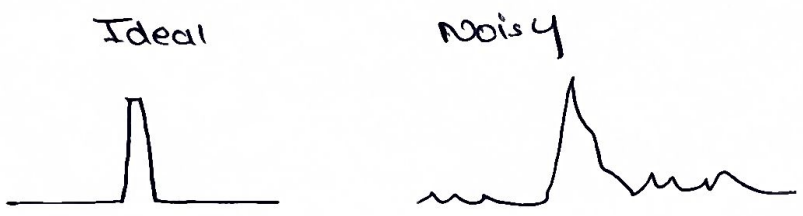
A. **Step Edge** - the image intensity abruptly changes from one value on one side of the discontinuity to a different value on the opposite side.



B. **Ramp Edge** - a step edge where the intensity change is not instantaneous but occurs over a finite distance.



C. **Ridge Edge** - the image intensity abruptly changes value but then returns to the starting value within some short distance. (usually generated by lines)



D. **Roof Edge** – a ridge edge where the intensity change is not instantaneous but occurs over a finite distance (usually generated by the intersection of two surfaces)

* Main Steps in Edge Detection

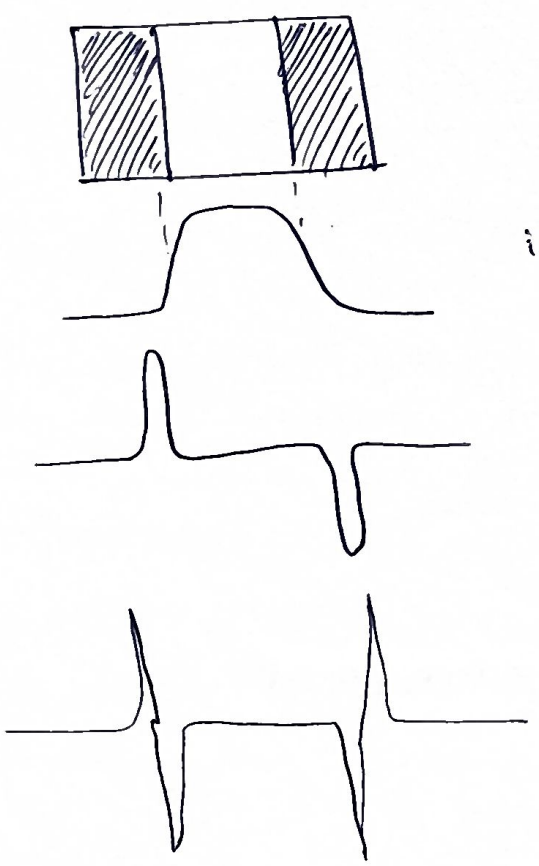
1. Smoothing - suppress as much noise as possible, without destroying true edges.

2. Enhancement - apply differentiation to enhance the quality of edges (i.e sharpening)

3. Thresholding - determine which edge pixels should be discarded as noise and which should be retained (i.e find the threshold edge magnitude)

4. Localization - determine the exact edge location

* Edge Detection using Derivatives



→ Points that lie on an edge are detected by:

(i) Detecting the local maxima or minima of the first derivative

(ii) detecting the zero-crossings of the second derivatives

* Edge Detection using the First Direction (Gradient)

→ The first derivative of an image can be computed using the gradient

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

→ The gradient is a vector which has magnitude & direction.

$$\text{magnitude}(\text{grad}(f)) = \sqrt{\frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}}$$ = edge strength

↳ isotropic in nature (detects edges in all directions)

$$\text{direction}(\text{grad}(f)) = \tan^{-1}\left(\frac{\partial f/\partial y}{\partial f/\partial x}\right)$$ = indicates edge direction (⊥r to the edge direction)

* Edge Detection Steps using the Gradient

1. Smooth the input image : $\hat{f}(x,y) = f(x,y) * G(x,y)$

2. $\hat{f}_x = \hat{f}(x,y) * M_x(x,y) \rightarrow \frac{\partial f}{\partial x}$

3. $\hat{f}_y = \hat{f}(x,y) * M_y(x,y) \rightarrow \frac{\partial f}{\partial y}$

4. $\text{magn}(x,y) = |\hat{f}_x| + |\hat{f}_y|$

5. $\text{dir}(x,y) = \tan^{-1}(\hat{f}_y / \hat{f}_x)$

6. If $\text{magn}(x,y) > T$, then it is a possible edge point

* Issues with the Edge Detection using Gradient

(1) Noise suppression - localization tradeoff

→ smoothing depends on the mask size

→ a larger mask size reduces noise, but they worsen localization (i.e they add uncertainty to the location of the edge) and vice versa.

(2) choice of threshold

(3) must link and thin edges appropriately

* Criteria for Optimal Edge Detection

A. Good detection - (i) minimize the probability of false +ves (spurious edges)

(ii) minimize the probability of false negatives (i.e missing real edges)

B. Good localization - detected edges must be as close as possible to the true edges

C. Single Response - minimize the number of local maxima around the true edge.

* Canny Edge Detector

→ The Canny edge detector carefully specifies the spatial bandwidth and excludes unnecessary thresholds, while permitting thin line structures to emerge and ensuring that they are connected together.

→ Canny has shown that the first derivative of the Gaussian closely approximates the operator that optimizes the product of the signal-to-noise ratio and localization.

Steps in Canny Edge Detection

Step1 : Compute f_x and f_y

$$f_x = \frac{\partial}{\partial x}(f * G) = f * \frac{\partial(G)}{\partial x} = f * G_x$$

$$f_y = \frac{\partial}{\partial y}(f * G) = f * \frac{\partial}{\partial y}(G) = f * G_y$$

$G(x,y)$ = Gaussian function.

$$G_x = \frac{-x}{\sigma^2} G(x,y) \qquad G_y = \frac{-y}{\sigma^2} G(x,y)$$

Step2 : Compute the gradient magnitude

$$\text{magn}(x,y) = |\hat{f}_x| + |\hat{f}_y|$$

$$\text{dir}(x,y) = \tan^{-1}\left(\hat{f}_y / \hat{f}_x\right)$$

Step3 : Apply non maxima suppression.

In NMS, the intensity of each local maximum is compared to its neighbors, and if it is significantly larger, it is retained as a feature, else it is suppressed.

Step4 : Apply hysteris thresholding / edge linking

Hysteresis thresholding uses two thresholds:

- a low threshold t_l
- a high threshold t_h (usually $t_h = 2t_l$)

If $\|\nabla f(x,y)\| \geq t_h \Rightarrow$ definitely an edge

$t_l \leq \|\nabla f(x,y)\| \leq t_h \Rightarrow$ maybe an edge, depends on context

$\|\nabla f(x,y)\| \leq t_l \Rightarrow$ definitely not an edge.

Algorithm for Hysteresis Thresholding / Edge Linking

① Produce 2 thresholded images $I_1(i,j)$ and $I_2(i,j)$ using t_l and t_h (I_2 is formed with $t_h \Rightarrow$ fewer false edges but more gaps in the contours)

② Link the edges in $I_2(i,j)$ into contours

2.1 Look in $I_1(i,j)$ when a gap is found

2.2 By examining the 8 neighbors in $I_1(i,j)$ gather edge points from $I_1(i,j)$ until the gap has been bridged to an edge in $I_2(i,j)$

* Edge Detection using the Laplacian Operator

→ The Laplacian is a second derivative operator, and it is sensitive only to changes in the intensity gradient.

→ Mathematically, it is

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

→ Localized masks for computing the Laplacian output can be derived by taking the difference of Gaussian (DoG) kernels using two Gaussians of different bandwidths . eg. $\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$

Properties of Laplacian Operator

(i) It is an isotropic operator

(ii) Cheaper to implement than the gradient (i.e one mask only)

(iii) Does not provide information about the edge direction

(iv) More sensitive to noise (since differentiation is done twice)

* Active Contours

→ Active contours/ snakes are used to find the boundary of objects or regions of interest within an image by minimizing an energy function.

→ Consider a contour consisting of discrete 2D point positions

$v_i = (x_i, y_i)$

→ The current contour has to be adjusted to form the new contour at each iteration.

→ This is done by:

(i) Define a cost / energy function that says how good a candidate configuration is.

(ii) Seek the next configuration that minimizes the cost function

→ The total energy of the snake is defined as:

$$E_{total} = E_{internal} + E_{external}$$

Internal Energy: encourages prior shape preferences – like smoothness, elasticity.

External energy – encourages contour to fit on places where image structures exist eg. edge.

→ Consider gradient images $G_x(x,y)$ and $G_y(x,y)$ at a point

$$E_{external}(v) = -\left(|G_x(v)|^2 + |G_y(v)|^2 \right)$$

For the whole curve

$$E_{external} = -\sum_{i=0}^{n-1} |G_x(x_i,y_i)|^2 + |G_y(x_i,y_i)|^2$$

$$E_{internal}(v(s)) = \alpha \underbrace{\left| \frac{dv}{ds} \right|^2}_{\text{tension, elasticity}} + \beta \underbrace{\left| \frac{d^2v}{ds^2} \right|^2}_{\text{stiffness, curvature}}$$

$$= \alpha \| v_{i+1} - v_i \|^2 + \beta \| v_{i+1} - 2v_i + v_{i-1} \|^2$$