# Module 5

## Image segmentation

**Segmentation :-**

→ Process of partioning a digital image into multiple regions and extracting meeningful region known as region of interest.

→ Image segmentation algorithms are based on similarity or discontinuity principle.

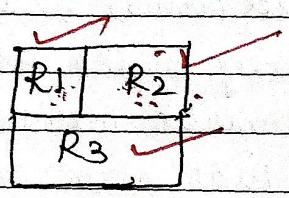
i) Discontinuity principle - Here we extract regions that are different in properties like intensity, colors and textures.

ii) Similarity principle - Grouping pixels based on a common property.

Formal Defination:

| | | | | |
|---|---|---|---|---|
| 20 | 20 | 20 | 10 | 10 |
| 200 | 20 | 20 | 10 | 10 |
| 20 | 20 | 20 | 10 | 10 |
| 15 | 15 | 15 | 15 | 15 |

→




Characteristics :→

i) If subregions are combined the orginal image can be obtained

ii) Subregions should be connected they should not be open during tracing process

iii) Regions $R_1$, $R_2$ should not share any common property

iv) Each region satisfies a predicate which satisfies gray scale value, texture or any image stastics.

Classification of image segmentation:-

1) Based on user interaction - Manual, semiAutomatic, Automatic

2) Based on pixel relationships - Contextual, Non contextual

Date
Page

## Based on user Interaction

i) Manual – It involves an expert observing and tracing the Region of Interest boundaries of an object using software. The software makes segmentation decisions. control points and splines help maintaining boundary integrity, the process is subjective, time consuming and susceptable to human errors.

ii) Automatic – They segment the structure of object without any human intervention. Generally carried out for a large number of images.

iii. Semi Automatic – They start with a manual input (seed points) and then they proceed automatically. They strike a balance b/w mannual and fully automatic methods, utilising human expertise for initial step and leveraging automated process for efficiency and consistency
Also called ~~Assisted~~ ~~as~~ Assisted manual segmen$^n$ algorithm.

## Based On Pixel Relationships:

i) Contextual (Region based Algorithms).
→ Groups pixels on shared properties and relationships
→ Aim to form homogenous regions by exploiting pixel similarities
→ Eg region growing, clustering

ii) Non Textual (Pixel based algorithms)
– Focus on individual pixels and local features without considering pixel relationships
– Detect and group local discontinuities like edges and isolated points
– Eg – Intensity based threshold, edge detection algorithms

@mypadhai

## Detection and Discontinuities:

~~i) Point~~

i) **Point Detection:-** Technique to identify isolated points in an image where the grey level is significantly different from the surrounding background in a homogenous area.

Generic 3x3 spatial mask:—

| | | |
|---|---|---|
| $z_1$ | $z_2$ | $z_3$ |
| $z_4$ | $z_5$ | $z_6$ |
| $z_7$ | $z_8$ | $z_9$ |

This mask is superimposed on an image and convulation process is applied.

$$\boxed{R = \sum_{k=1}^{9} z_k f_k}$$

- $f_k$ are gray level values of pixels associated with the image.
- Threshold T is used to identify the points
- A point is said to be detected at the location on which the mask is centered where T is non negative integer

| | | |
|---|---|---|
| 1 | 1 | 1 |
| 1 | -8 | 1 |
| 1 | 1 | 1 |

ii) **Line Detection:** 4 masks are used to get the responses, i.e $R_1, R_2, R_3, R_4$ for directions horizontal and vertical $+45^\circ$ and $-45^\circ$

Final maximum response is defined by.

$$M_1 = \begin{bmatrix} -1 & -1 & -1 \\ 2 & 2 & 2 \\ -1 & -1 & -1 \end{bmatrix} \qquad M_2 = \begin{bmatrix} -1 & 2 & -1 \\ -1 & 2 & -1 \\ -1 & 2 & -1 \end{bmatrix}$$

$$M_3 = \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix} \qquad M_4 = \begin{bmatrix} -1 & -1 & 2 \\ -1 & 2 & -1 \\ 2 & -1 & -1 \end{bmatrix}$$

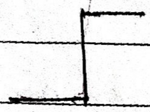
ii) **Edge Detection:** An edge is a set of connected pixels that lies b/w the boundary b/w 2 regions that differ in grey value.

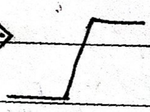
- An edge is extracted by computing the derivative of the image function. This consists of:-

i) Magnitude of derivative - indicates strength / contrast of the edge.

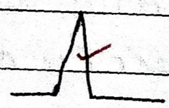
ii) Direction of derivative vector - measure of edge orientation

**Different edges are:-**

a) Step edge →  (A abrupt intensity change)

b) Ramp edge →  (Gradual change in intensity)

c) Spike edge →  (Quick change and immediately back to original intensity)

d) Roof edge -  (Not instantenous over a short distance)

**Stages in Edge Detection**

Start
↓
Input image
↓
Filtering
↓
Differentiation
↓
Localization
↓
display
↓
End

Filtering, Differentiation, Localization — 3 stages.

Date
Page

1. Filtering – Filter the input image to get maximum performance for the edge detectors. It involves:-
smoothing :- Reducing noise while preserving Important edges.
Gaussian filter – Applies gaussian function to blur the image and reduce the noise
It enhances the quality of edges.

2) Differentiation – Helps to distinguish edge pixels from non edge pixels. It is essential in edge detection to:-
i) Identify transitions in pixel intensity by calculating first derivative
ii) confirm edges by detecting zero crossings in the second derivative

First Derivative – It measures the rate of change of intensity
- If difference b/w 2 neighbouring pixel is 0, it indicates no transition and hence no edge and vice versa
Let f(x) is a continuous function:-

$$\frac{\partial f}{\partial x} = \lim_{x \to 0} \frac{f(x) - f(x - \Delta x)}{\Delta x}$$

For a discrete image, change of $\Delta x$ is 1. The discrete first derivative is measured by:-

$$\nabla x = f(x) - f(x-1)$$

Second Derivative (zero crossings)
- It checks for changes in the first derivative
- Significant changes in the value of first order derivatives indicate presence of edges.

Q) Consider an 1D image f(x) = 60 60 60 100 100 100
What are first and second derivatives?

First order derivative : ~~f(x)~~ $f(x+1) - f(x)$.

0 0 40 0 0

first order derivative: 0 0 40 0 0

2nd order derivative: find difference in values of 1st order derivative

$= 0 \quad 40 \quad -40 \quad 0$

edge        zero crossing

---

Gradient of an image:

$$\nabla f(x,y) = \begin{bmatrix} \dfrac{\partial f(x,y)}{\partial x} \\ \dfrac{\partial f(x,y)}{\partial y} \end{bmatrix} \begin{matrix} \rightarrow g_x \\ \rightarrow g_y \end{matrix}$$

or

$$\nabla f(x,y) = \begin{bmatrix} g_x \\ g_y \end{bmatrix}$$

where, $g_x = \dfrac{\partial f(x,y)}{\partial x}$   $g_y = \dfrac{\partial f(x,y)}{\partial y}$

Magnitude of above vector: $\nabla f(x,y) = \text{mag}(\nabla f(x,y))$

$$= |g(x)^2 + (g_y)^2|^{1/2}$$

Gradient approximation with absolute values:

$$\nabla f(x,y) \cong |g_x| + |g_y|$$

or $\nabla f(x,y) \cong \max(g_x, g_y)$

Gradient direction:

$$\theta = \tan^{-1}\left(\frac{g_y}{g_x}\right)$$

3) Localisation: Final stage of edge detection aimed at accurately identifying the exact position of edges in an image and ensures whether they are sharp well defined and continuous.

i) Normalisation – scales the gradient value to a range of 0 to K which helps to standardize the gradient magnitude across different image.

- Localisation also help in edge thinning, edge linking and storing the edge map.

$$N(x,y) = \frac{G(x,y)}{\max_{i=1,2,\dots n} G(i,j)} \times K$$

K is constant, set to value like 100

$$\text{Edgemap } E(x,y) = \begin{cases} 1 & \text{if } N(x,y) > T \\ 0 & \text{otherwise} \end{cases}$$

## Edge Detection Algorithms

- Derivative types
- Template Matching
- Gaussian Derivatives
- Pattern fit approach

i) First-order Edge Detection operators:

→ Identifies edges by measuring intensity gradients

→ The operators work on differential geometry and vector calculus where gradient determines rate of change of intensity of image.

$$\nabla = \begin{pmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \end{pmatrix}$$ when applied to image f

$$\nabla_f = \begin{pmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{pmatrix}$$

This result in the gradient vector providing both magnitude and direction of greatest intensity change of image resulting in an edge.

$$\text{Magnitude} = \nabla f(x,y) = \max(\nabla f(x,y)) = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

$$\text{Direction } \theta = \tan^{-1}\left(\frac{\partial f / \partial y}{\partial x / \partial y}\right)$$

Discrete approximations:

- Backward difference = $\boxed{\frac{f(x) - f(x-\Delta x)}{\Delta x}}$

- ~~Forward difference~~

With $\Delta x = 1$, the mask is $[1, -1]$

- Forward difference: $\boxed{\dfrac{f(x+\Delta x) - f(x)}{\Delta x}}$

When $\Delta x = 1$, the mask is $[-1, 1]$

- Central difference $= \boxed{\dfrac{f(x+\Delta x) - f(x-\Delta x)}{2\Delta x}}$

When $\Delta x = 1$, the mask is $\frac{1}{2}[1, 0, -1]$

Extending to 2D, $\boxed{g_x = \dfrac{\partial f}{\partial x}}$ $\boxed{g_y = \dfrac{\partial f}{\partial y}}$

ii) Roberts operation - Calculate the gradient by computing the difference between adjacent pixels.

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

Robert Kernel - Compute gradient w.r.t diagonal elements

Gradient components are:-

$$\boxed{g_x = \frac{\partial f}{\partial x} = z_9 - z_5}$$

$$\boxed{g_y = \frac{\partial f}{\partial y} = z_8 - z_6}$$

| | | |
|---|---|---|
| $z_1$ | $z_2$ | $z_3$ |
| $z_4$ | $z_5$ | $z_6$ |
| $z_7$ | $z_8$ | $z_9$ |

Masks = $g_x = \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix}$ $g_y = \begin{pmatrix} 0 & -1 \\ +1 & 0 \end{pmatrix}$

$$\nabla f(x,y) = mag(\nabla f(x,y)) = \sqrt{g(x)^2 + g(y)^2}$$

$$\theta = \tan^{-1}\left(\frac{g_y}{g_x}\right), \quad \nabla f(x,y) = |g_x| + |g_y|$$

Generic Gradient Based Algorithm

i) Read and smooth the image.

ii) Convolve with $g_x$ : $f_x = f * g_x$

iii) Convolve with $g_y$ $f_y = f * g_y$.

iv) Compute edge magnitude and orientations

v) Compare edge magnitude with threshold.

iii) **Prewitt operator** - It computes the gradient by taking the central difference of neighbouring pixels and applies averaging to reduce noise.

For 1D image. $\boxed{\frac{\partial f}{\partial x} = \frac{f(x+1) - f(x-1)}{2}}$

For 2D image. $\boxed{\frac{\partial f}{\partial x} = \frac{f(x+1, y) - f(x-1, y)}{2}}$

Central difference - represented by mask. $[-1, 0, +1]$

Prewitt Approximation - uses 3X3 mask to reduce noise.

$$\nabla f \approx |(Z_7 + Z_8 + Z_9) - (Z_1 + Z_2 + Z_3)| + |(Z_3 + Z_6 + Z_9) - (Z_1 + Z_4 + Z_7)|$$

edges in x direction : $M_x = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$

$M_y = \begin{pmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{pmatrix}$

iv) **Sobel operator** - Improvement over prewitt operation.

It incorporates to reduce noise and uses central dependence for gradient approximation.

$$\nabla f \approx |(Z_7 + 2Z_8 + Z_9) - (Z_1 + 2Z_2 + Z_3)| +$$
$$|(Z_3 + 2Z_6 + Z_9) - (Z_1 + 2Z_4 + Z_7)|$$

Sobel masks: X direction $M_x = \begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix}$

Y direction $M_y \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}$

Additional masks can be used in diagonal direction

$$M_x = \begin{bmatrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ -2 & 1 & 0 \end{bmatrix} \quad M_y = \begin{bmatrix} -2 & 1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 2 \end{bmatrix}$$

(iv) Template Matching Masks - Direction sensitive filter helps in detecting edges in specific directions

@vtupadhai

a) Kirsch masks - Detect edges by applying eight different masks oriented in compass direction north, south, east, west, northeast, north west, south east, south west

$$K_0 = \begin{bmatrix} -3 & -3 & 5 \\ -3 & 0 & 5 \\ -3 & -3 & 5 \end{bmatrix} \quad K_1 \text{ (North West)} = \begin{bmatrix} -3 & 5 & 5 \\ -3 & 0 & 5 \\ -3 & -3 & -3 \end{bmatrix}$$

$$K_2 \text{ (West)} = \begin{bmatrix} 5 & 5 & 5 \\ -3 & 0 & -3 \\ -3 & -3 & -3 \end{bmatrix} \quad K_3 \text{ (South West)} = \begin{bmatrix} 5 & 5 & -3 \\ 5 & 0 & -3 \\ -3 & -3 & -3 \end{bmatrix}$$

$$K_4 \text{ (South)} = \begin{bmatrix} 5 & -3 & -3 \\ 5 & 0 & -3 \\ 5 & -3 & -3 \end{bmatrix} \quad K_5 \text{ (South east)} = \begin{bmatrix} -3 & -3 & -3 \\ 5 & 0 & -3 \\ 5 & 5 & -3 \end{bmatrix} \quad K_6 \text{ (east)} = \begin{bmatrix} -3 & -3 & -3 \\ -3 & 0 & -3 \\ 5 & 5 & 5 \end{bmatrix}$$

$$K_7 \text{ (northeast)} = \begin{bmatrix} -3 & -3 & -3 \\ -3 & 0 & 5 \\ -3 & 5 & 5 \end{bmatrix}$$

b) Robinson compass masks - Same as Kirsch masks

$$R_0 = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

c) Frei Chen Masks - Decompose an image into different components, edges and averages.

Edge Detection Masks:

- $F_1 = \frac{1}{2\sqrt{2}}\begin{bmatrix} 1 & \sqrt{2} & 1 \\ 0 & 0 & 0 \\ -1 & -\sqrt{2} & -1 \end{bmatrix}$

- $F_2 = \frac{1}{2\sqrt{2}}\begin{bmatrix} 1 & 0 & -1 \\ \sqrt{2} & 0 & -\sqrt{2} \\ 1 & 0 & -1 \end{bmatrix}$

- $F_3 = \frac{1}{2\sqrt{2}}\begin{bmatrix} 0 & -1 & \sqrt{2} \\ 1 & 0 & -1 \\ \sqrt{2} & 1 & 0 \end{bmatrix}$

$F_4 : \frac{1}{2\sqrt{2}}\begin{pmatrix} \sqrt{2} & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & \sqrt{2} \end{pmatrix}$

Line Detection Masks:-

$F_5 : \frac{1}{2}\begin{pmatrix} 0 & 1 & 0 \\ -1 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix}$ $F_6 = \frac{1}{2}\begin{pmatrix} -1 & 0 & -1 \\ 0 & 0 & 0 \\ 1 & 0 & -1 \end{pmatrix}$

$F_7 : \frac{1}{6}\begin{pmatrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{pmatrix}$ $F_8 = \frac{1}{6}\begin{pmatrix} -2 & 1 & -2 \\ 1 & 4 & 1 \\ -2 & 1 & -2 \end{pmatrix}$

Average Detection Mask:

$F_9 : \frac{1}{3}\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$

@vrupadhai

* Second Order Derivative filters:

$$\nabla \cdot \nabla = \begin{bmatrix} \partial/\partial x \\ \partial/\partial y \end{bmatrix}\begin{bmatrix} \partial/\partial x \\ \partial/\partial y \end{bmatrix}$$

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

sensitive to noise and produce double edges.

$\nabla^2$ is laplacian operator. The laplacian of 2D function is defined by:-

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

- Gradient is vector and 2 orthogonal filter is required
- Laplacian operator is scaler, a single mask sufficient for edge detection process,

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

$$= \frac{\partial}{\partial x}\left[\frac{\delta}{\delta x} f(x+1,y) - \frac{\delta}{\delta x} f(x,y)\right]$$

$$= f(x+1,y) - f(x,y) - [f(x,y) - f(x-1,y)]$$

$$= f(x+1,y) - f(x,y) - f(x,y) + f(x-1,y)$$

Similarly: $\frac{\partial^2 f(x,y)}{\partial y^2} = \cancel{f(x+1,y}$

$$= f(x,y+1) - 2f(x,y) + f(x,y-1)$$

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

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

Algorithm: –

i) Generate the mask.

ii) Apply the mask.

iii) Detect zero crossing: A situation where neighbourhood pixels differ from each other in sign

| | | |
|---|---|---|
| 0 | 1 | 0 |
| 1 | −4 | 1 |
| 0 | 1 | 0 |

Laplacian mask

| | | |
|---|---|---|
| 1 | 0 | 1 |
| 0 | −4 | 0 |
| 1 | 0 | 1 |

Rotated by 45°

| | | |
|---|---|---|
| 1 | 1 | 1 |
| 1 | −8 | 1 |
| 1 | 1 | 1 |

Variant 1

| | | |
|---|---|---|
| 1 | −2 | 1 |
| −2 | 4 | −2 |
| 1 | −2 | 1 |

Variant 2

## Laplacian of Gaussian (Marr Hildrith) operator

→ To minimise noise susceptibility of laplacian operator, Laplacian of Gaussian is preferred. At first, the image is blurred using Gaussian operator and then Laplacian operator is used.

- Gaussian function reduce the noise and Laplacian minimises the detection of false edges.

For 1D, $\nabla^2(f * G) = f * \nabla^2 g = f * LoG$.

Let 2D Gaussian function be:

$$G_\sigma(x,y) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$$

$$\nabla(G_\sigma(x,y) * f(x,y)] = [\nabla G_\sigma(x,y)] * f(x,y) = LoG$$

The LoG can be derived as:-

$$\frac{\partial}{\partial x} G_\sigma(x,y) = \frac{\partial}{\partial x} e^{\left(\frac{-x^2+y^2}{2\sigma^2}\right)} = -\frac{x}{\sigma^2} e^{\left(\frac{-x^2+y^2}{2\sigma^2}\right)}$$

Similarly,

$$\frac{\partial^2}{\partial x^2} G_\sigma(x,y) = \frac{x^2}{\sigma^4} e^{\left(\frac{-x^2+y^2}{2\sigma^2}\right)} - \frac{1}{\sigma^2} e^{\left(\frac{-x^2+y^2}{2\sigma^2}\right)} = \frac{x^2-\sigma^2}{\sigma^4} e^{\left(\frac{-x^2+y^2}{2\sigma^2}\right)}$$

By ignoring normalisation constant $\frac{1}{\sqrt{2\pi\sigma^2}}$

$$\frac{\partial^2}{\partial y^2} G_\sigma(x,y) = \frac{y^2}{\sigma^4} e^{\left(\frac{-x^2+y^2}{2\sigma^2}\right)} - \frac{1}{\sigma^2} e^{\left(\frac{-x^2+y^2}{2\sigma^2}\right)} = \frac{y^2-\sigma^2}{\sigma^4} e^{\left(-\frac{x^2+y^2}{2\sigma^2}\right)}$$

The LoG kernel is described as:-

$$LoG \triangleq \frac{\partial^2}{\partial x^2} G_\sigma(x,y) + \frac{\partial^2}{\partial y^2} G_\sigma(x,y) = \frac{x^2+y^2-2\sigma^2}{\sigma^4} e^{\left(\frac{-x^2+y^2}{2\sigma^2}\right)}$$

LoG Algorithm:-

i) Generate the mask and apply LoG to the image.

ii) Detect zero crossing.

## Difference of Gaussian filter

→ LoG filter can be approximated by taking 2 differently sized Gaussians.

$$G_{\sigma_1}(x,y) = \frac{1}{\sqrt{2\pi\sigma_1^2}} \exp\left(-\frac{x^2+y^2}{2\sigma_1^2}\right)$$

$$G_{\sigma_2}(x,y) = \frac{1}{\sqrt{2\pi\sigma_2^2}} \exp\left(\frac{-x^2+y^2}{2\sigma_2^2}\right)$$

DoG is the difference b/w these 2 kernels:-

$$DoG = G_{\sigma_1} - G_{\sigma_2} = \frac{1}{\sqrt{2\pi}}\left[\frac{1}{\sigma_1} e^{\left(\frac{-x^2+y^2}{2\sigma_1^2}\right)} - \frac{1}{\sigma_2} e^{-\left(\frac{x^2+y^2}{2\sigma_2^2}\right)}\right]$$

### DoG Algorithm:

i) Generate the mask and apply DoG to the image.
ii) Detect zero crossing and apply the threshold to suppress the weak zero crossings.
iii) Display and exit.

## Canny Edge Detection

→ This algorithm balances edge detection accuracy, localisation and response consistency.
→ It combines Gaussian smoothing, gradient computation, non maxima suppression and hysteresis thresholding to achieve optimal detection results.
→ Maintains edge continuity and reduces noise sensitivity.

### Performance criteria: →

i) Good edge Detection – It should accurately detect real edge points while discarding false ones.

ii) Good edge localisation – The detected edge points should be as close as possible to the true edges

iii) Single response to each edge – The algorithm should avoid multiple responses to a single edge, eliminating false, double and spurious edges.

Algorithm:–

i) Gaussian filtering –

→ Convolve the image with Gaussian filter to smooth and reduce the noise

→ Compute gradient of the smoothed image to detect edge strength and orientation.

→ Store the edge magnitude in array $M(x,y)$ and edge orientation in array $\alpha(x,y)$

ii) Non Maxima suppression

→ Thin the edges to produce sharp edges by examining the edge magnitude and orientation

→ Approximate the gradient direction to one of 4 sectors reducing 0–360° range into four sectors.

→ For each point $M(x,y)$, compare its magnitude with neighbouring pixels $M(x_1,y_1)$ and $M(x_2,y_2)$ along the gradient direction

→ If $M(x,y)$ is less than $M(x_1,y_1)$ or $M(x_2,y_2)$ suppress $M(x,y)$ by setting its value to 0

iii) Hysteresis Thresholding :–

→ Apply 2 thresholds $t_0$ (low) and $t_1$ (high) to differentiate strong and weak edges.

- If gradient magnitude is greater than $t_1$, it is accepted as a definite edge point.
- If gradient magnitude is less than $t_0$, it is rejected as a weak edge point
- For gradient magnitude between $t_0$ and $t_1$, determine if they are part of an edge based on their context.
- Create 2 images using threshold $t_0$ and $t_1$

→ The high threshold image will have definitive edges but may contain gaps

→ The low threshold image might include noisy edge points

- Bridge the gaps in the high threshold image by consulting the low threshold image, examining the 8 neighbours of each pixel to link the edges and form a continuous contour.

@vtupadhai