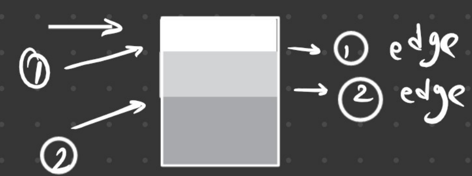
KRISHAI TECHNOLOGIES

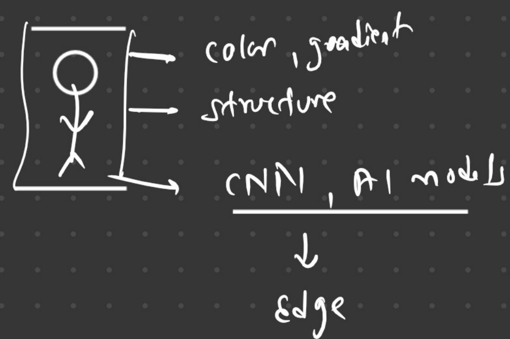
Edge Detection Using Sobel, Canny & Laplacian

www.krishnaik.in

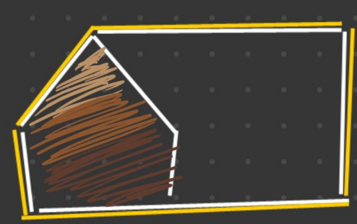
Edges represent sharp changes in intensity (or color) in an image. These changes often correspond to object boundaries, textures, or other key features.

Edge Detection
- Feature extraction
- Shape analysis
  - Detect Shapes
  - Contours
  - Boundaries
- Preprocessing





Edge Detection
- Sobel Filter
- Laplacian Filter
- Canny Edge Detector



Sobel Filter

The Sobel filter detects edges in horizontal and vertical directions by calculating gradients.
Below filters mathematically mirrors the derivate operation.

Horizontal kernel (G_x)

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

Vertical kernel (G_y)

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

Convolution

IG_x , IG_y

Combining Gradients

$G = \sqrt{G_x^2 + G_y^2}$ → Calculate magnitude of gradient of each pixel

$$G = \sqrt{IG_x[0][0]^2 + IG_y[0][0]^2} = \text{value}$$

Magnitude

$G =$

10	15	3	8
6	10	10	2
10	18	5	12
13	17	10	18

→ Avg → Ex. $\boxed{10}$ ← Threshold

0	1	0	0
0	0	0	0
0	1	0	1
0	1	0	1

→ $\boxed{\text{value if} > \text{threshold else } 0}$

Laplacian Filter

Laplacian detects edges by finding regions of rapid intensity change. Unlike Sobel, it combines x- and y-direction derivatives in a single operation.

The Laplacian kernel below is a discrete approximation of this continuous mathematical derivative. Below filters mathematically mirrors the derivate operation.

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

Kernel 1   Kernel 2

Both kernels are performing 2nd order Derivative

Laplacian detects sudden change in intensity, highlighting the edge.

Canny Edge

The Canny algorithm detects edges using multi-step processing, combining Sobel and thresholding techniques.

Step 1 : Convert to Grayscale because edges are present in color change information not in colors.

Step 2 : Apply Gaussian Blur which adds smoothness to image making it less sensitive to small color change variations.

Step 3.1 : Compute Intensity Gradients using the Sobel operator to approximate the first derivatives in x and y-directions

Step 3.2 : Step 3.1 calculate magnitude but we also need angle of the edge and we can get using :

$$\theta = \text{arctangent}\left(\frac{G_y}{G_x}\right)$$

Step 4 : Non-Maximum Suppression.

→ For each pixel, compare the gradient magnitude to its two neighbors along the gradient direction

→ If the pixel's gradient is not the largest, suppress (set to 0).

Example:

Gradient Direction : 45°

compare $G(x,y) \longrightarrow G(x+1, y+1)$

$\longrightarrow G(x-1, y-1)$

This keeps only the strongest edge points.

Step 5 : Double Thresholding

→ High Threshold: Pixels above this are strong edges.

→ Low Threshold: Pixels below this are suppressed entirely.

→ Pixels between the two thresholds are classified as weak edges.

→ High threshold = 100

→ Low threshold = 50

→ G > 100 : Strong edge
→ 50 < G <= 100 : Weak edge
→ G <= 50 Suppressed (Not and edge)

Step 6 : Edge Tracking by Hysteresis

→ In this final step, weak edges are either kept or discarded based on their connectivity to strong edges.

→ If a weak edge is connected to a strong edge, it is Retained else Discarded.

Canny combines several techniques (Gaussian blur, Sobel, thresholds) to ensure edges are clean & noise-free.