IPCV Practical No : 5

18 September 2025 08:23

Aim

Detect edges in an image using various edge detection techniques to identify object boundaries.

Requirements

- **Libraries:** OpenCV (cv2), NumPy (numpy)
- **Sample Images:** jpg/png images

Theory

1. What is an Edge?
   - Edge is a location where there is a **sudden change in image intensity**.
   - The **discontinuity in image brightness** represents object boundaries.
   - Edge detection converts a 2D image into a set of points where intensity changes rapidly.
2. Edge Detection Techniques

◇ Gradient-Based (First Derivative) Methods:

- Detect edges by measuring intensity changes using first derivative (gradient).
- Operators:
  - **Sobel Operator** – Uses derivatives to detect edges in x and y directions.
  - **Prewitt Operator** – Similar to Sobel with different kernel values.
  - **Roberts Operator** – Computes gradients diagonally.

◇ Second Derivative-Based Methods:

- Detect edges by finding zero-crossings using second derivative (Laplacian).
- Operators:
  - **Laplacian Operator** – Uses second derivative to highlight regions of rapid intensity change.
  - **LoG (Laplacian of Gaussian)** – Smooths image first, then applies Laplacian.
  - **Canny Edge Detection** – Advanced multi-step method: Gaussian smoothing → Gradient calculation → Non-maximum suppression → Thresholding.

3. First derivative :-

Important Edge operators

- Prewitt Operator

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad G_y = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$

- Sobel Operator

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

- Robert Cross Operator

$$G_x = \begin{bmatrix} +1 & 0 \\ 0 & -1 \end{bmatrix} \quad G_y = \begin{bmatrix} 0 & +1 \\ -1 & 0 \end{bmatrix}$$

- Resultant gradient magnitude

$$G = \sqrt{G_x^2 + G_y^2}$$

- Resultant gradient direction

$$\theta = \tan^{-1}\left[\frac{G_y}{G_x}\right]$$

- An approximate magnitude is computed using

$$G = |G_x| + |G_y|$$

1. Sobel Edge detector :

   - The Sobel filter is a **gradient-based edge detection operator** that computes the **first derivative** of an image.
   1. It detects edges by highlighting regions where **pixel intensity changes rapidly**.
   2. The image is processed separately in **X and Y directions**. The final edge image is usually the **combination (sum) of X and Y edges**

- Step to solve

  1. Define sobel Problem
  2. Compute Gx and Gy for Each 3×3 Region
  3. Compute Gradient Magnitude G
  4. Interept the result

- Given the **5×5 grayscale image**:

$$I = \begin{bmatrix} 10 & 10 & 80 & 80 & 80 \\ 10 & 10 & 80 & 80 & 80 \\ 10 & 10 & 80 & 80 & 80 \\ 10 & 10 & 80 & 80 & 80 \\ 10 & 10 & 80 & 80 & 80 \end{bmatrix}$$

- We will apply the **Sobel X (Gx) and Sobel Y (Gy) kernels** to all possible 3×3 regions.

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad G_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

$$G = \begin{bmatrix} 280 & 280 & 280 \\ 280 & 280 & 280 \\ 280 & 280 & 280 \end{bmatrix}$$

2 . Prewitt operator

- The Prewitt operator was developed by Judith M. S. Prewitt. Prewitt operator is used for edge detection in an image. Prewitt operator detects both types of edges, these are:

Vertical Edges or along the y-axis.

Horizontal edges or along the x-axis,

Step to Solve:

1. **Define Prewitt Problem** → Choose the image and prepare it for processing.
2. **Compute Gx and Gy for Each 3×3 Region** → Use Prewitt kernels:
3. **Compute Gradient Magnitude G** → G = sqrt(Gx^2 + Gy^2)
4. **Interpret the Result** → Higher gradient → stronger edge

- Given the **5×5 grayscale image**:

$$I = \begin{bmatrix} 10 & 10 & 80 & 80 & 80 \\ 10 & 10 & 80 & 80 & 80 \\ 10 & 10 & 80 & 80 & 80 \\ 10 & 10 & 80 & 80 & 80 \\ 10 & 10 & 80 & 80 & 80 \end{bmatrix}$$

- We will apply the **Sobel X (Gx) and Sobel Y (Gy) kernels** to all possible 3×3 regions.

If We will apply **Prewitt X (Gx) and Prewitt Y (Gy) filters** to each valid pixel for the previous image given in previous (sobel) example, we will get

$$G_x = \begin{bmatrix} 210 & 210 & 210 \\ 210 & 210 & 210 \\ 210 & 210 & 210 \end{bmatrix}$$

$$G_y = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

- The edge magnitude is given by $G = \sqrt{210^2} = 210$

Interpretation of Results

•**The edge strength is uniform (210) across all valid pixels** (same as in Sobel but with a different magnitude).

•**Gx is large, and Gy is zero**, meaning the edges detected are **purely horizontal**.

3. Robert

- he Roberts filter is a **gradient-based edge detection operator** that detects edges using **diagonal differences**.
- It computes the **first derivative** of an image along diagonals.
- Highlights regions where **pixel intensity changes rapidly**, especially for **diagonal edges**.
- Image is processed using a **2×2 kernel**, and the result gives edge strength at each pixel.

Step to Solve:

a. **Define Roberts Problem** → Select the image for edge detection.
b. **Compute Gx and Gy for Each 2×2 Region** → Use Roberts kernels:
c. **Compute Gradient Magnitude G** → G = sqrt(Gx^2 + Gy^2)
d. **Interpret the Result** → High gradient → edge detected diagonally

Example

- Given 5×5 Image:

$$I = \begin{bmatrix} 10 & 10 & 80 & 80 & 80 \\ 10 & 10 & 80 & 80 & 80 \\ 10 & 10 & 80 & 80 & 80 \\ 10 & 10 & 80 & 80 & 80 \\ 10 & 10 & 80 & 80 & 80 \end{bmatrix}$$

- Since the **Roberts Cross operator** is a **2×2 filter**, we apply it to each valid **2×2 region** in the image.

$$G_x = \begin{bmatrix} 0 & -70 & 0 & 0 \\ 0 & -70 & 0 & 0 \\ 0 & -70 & 0 & 0 \\ 0 & -70 & 0 & 0 \end{bmatrix}$$

$$G_y = \begin{bmatrix} 0 & 70 & 0 & 0 \\ 0 & 70 & 0 & 0 \\ 0 & 70 & 0 & 0 \\ 0 & 70 & 0 & 0 \end{bmatrix}$$

$$G = \begin{bmatrix} 0 & 98.99 & 0 & 0 \\ 0 & 98.99 & 0 & 0 \\ 0 & 98.99 & 0 & 0 \\ 0 & 98.99 & 0 & 0 \end{bmatrix}$$

4. Second derivative :-

A. Laplacian filtre :-

- The Laplacian of an image highlights regions of rapid intensity change. Any feature with a sharp discontinuity will be enhanced by a Laplacian operator.
- The Laplacian filter comes under the derivative filter category.
- It is a second-order filter used in image processing for edge detection and feature extraction.

The Laplacian Operator

The **Laplacian** is a **second-order derivative filter** that combines the second derivative in the x and y directions:

The Laplacian Operator

The **Laplacian** is a **second-order derivative filter** that combines the second derivative in the x and y directions:

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

In discrete 2D form, common Laplacian kernels are:

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \quad \text{or} \quad \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

✅ Properties:

- Coefficients sum to **0** (like edge detectors).
- Highlights regions of **rapid intensity change** (edges, fine details).
- Very **sensitive to noise** (because noise has rapid intensity changes too).

B. LoG(Laplacian of Gaussian):

- The Laplacian of Gaussian (LoG) combines Gaussian smoothing and Laplacian edge detection.
- LoG first smooths the image with a Gaussian before applying the Laplacian:
  It helps in detecting edges while reducing noise

Laplacian of Gaussian

To reduce noise sensitivity, we **smooth the image with a Gaussian** before applying the Laplacian.

This is called **Laplacian of Gaussian (LoG)**:

$$LoG(x, y) = \nabla^2 \left(G(x, y) * I(x, y)\right)$$

where $G(x, y)$ is a 2D Gaussian function.

Example LoG kernel (5×5):

$$\begin{bmatrix} 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & -2 & -1 & 0 \\ -1 & -2 & 16 & -2 & -1 \\ 0 & -1 & -2 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 \end{bmatrix}$$

✅ Properties:

- Smooths first (Gaussian), then finds edges (Laplacian).
- Detects **edges with less noise sensitivity.**

Advantages

1. Highlights object boundaries for further processing.
2. Useful in image segmentation and pattern recognition.
3. Supports multiple edge detection methods for different applications.

Disadvantages

1. Sensitive to noise → may detect false edges.
2. Poor lighting or low contrast reduces accuracy.
3. Requires careful choice of operator and parameters.

Output

- Image with edges highlighted (vertical, horizontal, or diagonal depending on operator).

Conclusion

- Edge detection identifies regions of rapid intensity change, highlighting object boundaries efficiently.