1. Explain the various stages involved in edge detection process. 10M CO5 L2

The idea is to detect the sharp changes in image brightness, which can capture the important events and properties. This is done in three stages. The edge detection process is shown in Fig. 9.8

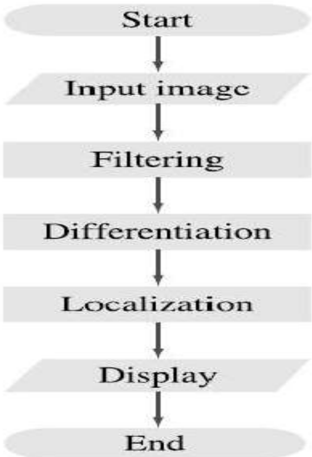


Fig. 9.8 Edge detection process

1. Filtering: The first step in edge detection is often to smooth the image to reduce the amount of noise. This is important because edge detection techniques are sensitive to noise and can mistake noise for edges. Smoothing can be achieved using various filters, such as Gaussian filters, which reduce the influence of noise without blurring the edges too much.

2. Differentiation: After smoothing, the next step is to calculate the gradient of the image. This phase distinguishes the edge pixels from other pixels. The idea of edge detection is to find the difference between two neighbourhood pixels. If the pixels have the same value, the difference is 0. This means that there is no transition between the pixels. The non- zero difference indicates the presence of an edge point.

The gradient is a measure of the change in brightness (or intensity) at each pixel in the image. It has two components: **the gradient magnitude** (strength of the edge) and the **gradient direction** (orientation of the edge).

Images are two-dimensional. Hence, the gradient vector of f(x, y) is also two-dimensional. The gradient of an image f(x, y) at location (x,y) is a vector that consists of the partial derivatives of f(x,y) as follows:

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

or simply

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

Where

$$g_x = \left[\frac{\partial f(x, y)}{\partial x}\right] \text{ and } g_y = \left[\frac{\partial f(x, y)}{\partial y}\right]$$

The magnitude of this vector, generally referred to as the gradient ∇f, is

$$\nabla f(x, y) = \text{mag}(\nabla f(x, y)) = \left[(g_x)^2 + (g_y)^2\right]^{1/2}$$

Edge strength is indicated by the edge magnitude. The **direction of the gradient vector** is useful in detecting a sudden change in image intensity. The common practice is to approximate the gradient with absolute values that are simpler to implement, as follows:

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

or

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

The gradient direction can be given as:

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

3. Localization: In this stage the detected edges are localized. The localization process involves determining the exact location of the edge. In addition, this stage involves edge thinning and edge linking steps to ensure that the edge is sharp and connected. The sharp and connected edges are then displayed.

The prerequisite for the localization stage is normalization of the gradient magnitude. The calculated gradient can be scaled to a specific range say, 0-K by performing this operation. For example, the value of constant K may be an integer, say, 100. N(x, y) is called the normalized edge image and is given as

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

The normalized magnitude can be compared with a threshold value T to generate the edge map. The edge map is given as

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

The edge map is then displayed or stored for further image processing operations.

2. How edge detection is performed in digital images using Roberts operator. (8M) CO5 L3

The Roberts operator is an edge detection algorithm that was one of the first methods used in digital image processing to detect edges in digital images. It works by approximating the gradient of the image intensity function. The Roberts operator uses two 2x2 kernels (also known as masks) to calculate the differences in diagonal directions.

Let $f(x,y)$ and $f(x+1, y)$ be neighbouring pixels. The difference between the adjacent pixels is obtained by applying the mask $[1\ -1]$ directly to the image to get the difference between the pixels. This is defined mathematically as

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

Roberts kernels are derivatives with respect to the diagonal elements. Hence they are called cross-gradient operators. They are based on the cross diagonal differences. The approximation of Roberts operator can be mathematically given as

$$g_x=\frac{\partial f}{\partial x}$$

$$g_y=\frac{\partial f}{\partial y}$$

Roberts masks of the for the given cross difference is

$$g_x=\begin{pmatrix}-1 & 0\\ 0 & 1\end{pmatrix} \text{ and } g_y=\begin{pmatrix}0 & -1\\ 1 & 0\end{pmatrix}$$

The Roberts operator is simple and computationally efficient, but it has some limitations.

- It is sensitive to noise because it uses a small 2x2 kernel, which means it may detect noise as edges.
- Additionally, it only considers two diagonal directions, which can lead to missing some edges that are not aligned with these directions.

Despite these limitations, the Roberts operator is historically significant and can still be useful in certain applications where simplicity and speed are important.

The generic gradient-based algorithm can be given as

1. Read the image and smooth it.
2. Convolve the image f with g_x. Let $\hat{f}(x) = f * g_x$.
3. Convolve the image with g_y. Let $\hat{f}(y) = f * g_y$.
4. Compute the edge magnitude and edge orientation.
5. Compare the edge magnitude with a threshold value. If the edge magnitude is higher, assign it as a possible edge point.

This generic algorithm can be applied to other masks also.

3. How edge detection is performed in digital images using Sobel operator. (8M) CO5 L3

The Sobel operator is a discrete differentiation operator used primarily for edge detection in digital images. It works on the principle of gradient detection, where the edges are identified by looking for the maximum local changes in intensity levels.

The Sobel operator also relies on central differences. This can be viewed as an approximation of the first Gaussian derivative. This is equivalent to the first derivative of the Gaussian blurring image obtained by applying a 3×3 mask to the image. Convolution is both commutative and associative, and is given as

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

A 3×3 digital approximation of the Sobel operator is given as

$$\nabla f \cong \left|(z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)\right| + \left|(z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)\right|$$

The Sobel operator consists of two separate kernels (matrices or masks) to detect edges in the horizontal and vertical directions, respectively.

The masks are as follows:

$$M_x = \begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix} \text{ and } M_y = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}$$

An additional mask can be used to detect the edges in the diagonal direction.

$$M_x = \begin{pmatrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ -2 & -1 & 0 \end{pmatrix} \text{ and } M_y = \begin{pmatrix} -2 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 2 \end{pmatrix}$$

The edge mask can be extended to 5 x 5, 7 x 7, etc. An extended mask always gives a better performance.

4. How edge detection is performed in digital images using Prewitt operator. (8M) CO5 L3

The Prewitt operator is a gradient–based edge detection operator used to detect boundaries of objects within a digital image. To detect and highlight the edges, the Prewitt operator approximates the gradient of the image intensity at each pixel.
The Prewitt operator utilizes two 3×3 convolution kernels to detect edges. One is used to detect the horizontal edges while the other one is to detect the vertical edges of objects within the image.

The Prewitt method takes the central difference of the neighbouring pixels; this difference can be represented mathematically as

$$\frac{\partial f}{\partial x} = f(x+1) - f(x-1)/2$$

For two dimensions, this is

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

The central difference can be obtained using the mask $[-1\ 0\ +1]$. This method is very sensitive to noise. Hence to avoid noise, the Prewitt method does some averaging. The Prewitt approximation using a 3×3 mask is as follows:

$$\nabla f \cong |(z_7 + z_8 + z_9) - (z_1 + z_2 + z_3)| + |(z_3 + z_6 + z_9) - (z_1 + z_4 + z_7)|$$

This approximation is known as the Prewitt operator. Its masks are as follows:

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

5. How edge detection is performed in digital images using First-order Edge Detection operator. (8M) CO5 L3

Edge detection using first-order edge detection operators involves calculating the gradient of the image intensity function. The gradient measures the rate of change in intensity at each pixel, which helps identify regions with significant changes, indicative of edges.

Steps for Edge Detection Using First-order Operators

1. **Smoothing (Optional):**
   - Apply a smoothing filter (e.g., Gaussian) to the image to reduce noise, which can cause false edges.
2. **Gradient Calculation:**
   - Compute the gradient of the image intensity. The gradient is a vector that has both magnitude and direction, indicating the strength and direction of the maximum rate of change of intensity.
3. **Gradient Magnitude and Direction:**
   - The gradient magnitude represents the strength of the edge, while the gradient direction indicates the orientation of the edge.

Common First-order Edge Detection Operators

1. **Sobel Operator:**
   - The Sobel operator uses convolution masks to approximate the gradient in the x and y directions. It emphasizes edges running horizontally and vertically.
   - Sobel masks for the x and y directions:

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

2. **Prewitt Operator:**
   - Similar to the Sobel operator but with different convolution masks.
   - Prewitt masks for the x and y directions:

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

3. **Roberts Cross Operator:**
   - Computes the gradient using a pair of 2x2 convolution masks, which are simpler and faster but more sensitive to noise.
   - Roberts masks:

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

6. Describe about the canny edge detector with necessary equation and also write its algorithm. 12M CO5 L2

The Canny edge detector is a multi-step algorithm used for detecting edges in an image while minimizing noise and ensuring thin, well-connected edges. The process involves several stages: smoothing, gradient computation, non-maximum suppression, double thresholding, and edge tracking by hysteresis.

Stages of the Canny Edge Detector

1. Smoothing

First, the image is smoothed using a Gaussian filter to reduce noise.

$$G(x,y) = I(x,y) * \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

where $I(x,y)$ is the original image, $*$ denotes convolution, and σ is the standard deviation of the Gaussian filter.

2. Gradient Computation

The gradient of the smoothed image is computed to find the intensity changes. This is done using Sobel operators to approximate the derivatives in the x and y directions.

$$G_x = \frac{\partial G}{\partial x} \approx \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} * G$$

$$G_y = \frac{\partial G}{\partial y} \approx \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} * G$$

The gradient magnitude and direction are then calculated as:

$$\text{Magnitude} = |\nabla G| = \sqrt{G_x^2 + G_y^2}$$

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

3. Non-Maximum Suppression

To thin the edges, non-maximum suppression is applied. The gradient magnitude is compared to the magnitudes of its neighbors along the gradient direction. If the gradient magnitude at a pixel is not the maximum compared to its neighbors, it is set to zero.

4. Double Thresholding

Two thresholds are applied to identify strong and weak edges:

- **Strong Edges:** Pixels with gradient magnitude above the high threshold.
- **Weak Edges:** Pixels with gradient magnitude between the low and high thresholds.

5. Edge Tracking by Hysteresis

Weak edges are retained if they are connected to strong edges, ensuring edge continuity. Otherwise, weak edges are suppressed.

Canny Edge Detection Algorithm

1. **Input:** Image *I(x, y),* low threshold T_L, high threshold T_H.
2. **Smooth the image:** Convolve the image *I* with a Gaussian filter to obtain *G*.
$$G(x,y) = I(x,y) * \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$
3. **Compute gradients:** Use Sobel operators to find gradients G_x and G_y.
$$G_x = \frac{\partial G}{\partial x}$$
$$G_y = \frac{\partial G}{\partial y}$$
4. **Calculate gradient magnitude and direction:**
$$\text{Magnitude} = |\nabla G| = \sqrt{G_x^2 + G_y^2}$$
$$\text{Direction} = \theta = \arctan\left(\frac{G_y}{G_x}\right)$$
5. **Apply non-maximum suppression:** Thin the edges by setting non-maximum pixels to zero.
6. **Apply double thresholding:** Classify pixels as strong edges, weak edges, or non-relevant based on T_L, and T_H.
7. **Edge tracking by hysteresis:** Retain weak edges if they are connected to strong edges.

7. Describe about the Marr-Hilldreth edge detector used in image segmentation with necessary equations. 10M CO5 L3

The Marr-Hildreth edge detector is an edge detection technique based on the theory of visual processing in the human visual system, developed by David Marr and Ellen Hildreth. It utilizes the Laplacian of Gaussian (LoG) to identify edges in an image. The main idea is to first smooth the image with a Gaussian filter to reduce noise and then apply the Laplacian operator to find regions of rapid intensity change.

Stages of the Marr-Hildreth Edge Detector

1. Smoothing with Gaussian Filter

To reduce noise and small variations in the image, the image is first smoothed using a Gaussian filter. The Gaussian smoothing function is given by:

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

where σ is the standard deviation of the Gaussian filter, controlling the amount of smoothing.

2. Apply the Laplacian of Gaussian (LoG)

The Laplacian of Gaussian combines the smoothing and differentiation steps into a single operation. The LoG function is defined as:

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

where ∇^2 denotes the Laplacian operator. The LoG can be computed by convolving the image with the LoG filter:

$$L(x, y) = I(x, y) * \nabla^2 G(x, y)$$

where $I(x, y)$ is the original image and $*$ denotes convolution.

3. Zero-Crossing Detection

After applying the LoG, the edges are located at points where the Laplacian changes sign, i.e., zero-crossings. These zero-crossings indicate the positions of edges. The algorithm detects zero-crossings by examining the sign changes in the LoG response:

- An edge is identified where the response changes from positive to negative or from negative to positive.

Marr-Hildreth Edge Detection Algorithm

1. **Input:** Image $I(x, y)$ and standard deviation σ for the Gaussian filter.
2. **Smooth the image**: Convolve the image I with a Gaussian filter to obtain the smoothed image S.

   $$S(x, y) = I(x, y) * \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

3. **Compute the Laplacian of Gaussian (LoG)**: Convolve the smoothed image S with the LoG filter to obtain the LoG response L.
   $L(x, y) = S(x, y) * \left(\frac{x^2+y^2-2\sigma^2}{\sigma^4}\right) e^{-\frac{x^2+y^2}{2\sigma^2}}$
4. **Detect zero-crossings**: Identify edges by finding zero-crossings in the LoG response L. A zero-crossing occurs when the sign of L changes between adjacent pixels.

8. Define image segmentation formally and describe the characteristics of the segmentation process. 10M CO5 L2

Image segmentation has emerged as an important phase in image-based applications. Segmentation is the process of partitioning a digital image into multiple regions and extracting a meaningful region known as the region of interest (ROI). Regions of interest vary with applications. For example, if the goal of a doctor is to analyse the tumour in a computer tomography (CT) image, then the tumour in the image is the ROI.

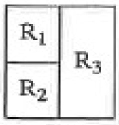
An image can be partitioned into many regions $R_1, R_2, R_3, \ldots, R_n$. For example, the image R in Fig. 9.1(a) is divided into three subregions R_1, R_2, and R_3 as shown in Figs 9.1(b) and 9.1(c). A subregion or sub-image is a portion of the whole region R. The identified subregions should exhibit characteristics such as uniformity and homogeneity with respect to colour, texture, intensity, or any other statistical property. In addition, the boundaries that separate the regions should be simple and clear.

20	20	10	10	10
20	20	10	10	10
20	20	10	10	10
15	15	10	10	10
15	15	10	10	10

(a)

20	20	10	10	10
20	20	10	10	10
20	20	10	10	10
15	15	10	10	10
15	15	10	10	10

(b)



(c)

Fig. 9.1 Image segmentation (a) Original image (b) Pixels that form a region (c) Image with three regions

The characteristics of the segmentation process are the following:

1. If the subregions are combined, the original region can be obtained. Mathematically, it can be stated that $\bigcup R_i = R$ for $i = 1, 2, \ldots, n$. For example, if the three regions of Fig. 9.1(c) R_1, R_2, and R_3 are combined, the whole region R is obtained.
2. The subregions R_i should be connected. In other words, the region cannot be open-ended during the tracing process.
3. The regions $R_1, R_2, \ldots, R_n$ do not share any common property. Mathematically, it can be stated as $R_i \bigcap R_j = \varphi$ for all i and j where $i \neq j$. Otherwise, there is no justification for the region to exist separately.
4. Each region satisfies a predicate or a set of predicates such as intensity or other image statistics, that is, the predicate (P) can be colour, grey scale value, texture, or any other image statistic. Mathematically, this is stated as $P(R_i) = \text{True}$.

9. Explain the classification of various image segmentation algorithms and delineate their distinct types. 10M CO5 L2

There are different ways of classifying the segmentation algorithms. Figure 9.2 illustrates these ways. One way is to classify the algorithms based on user interaction required for extracting the ROI. Another way is to classify them based on the pixel relationships.

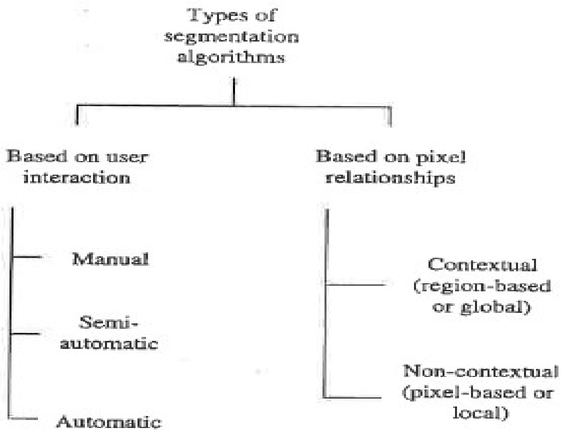


Fig. 9.2 Classification of segmentation algorithms

1. Based on user interaction, the segmentation algorithms can be classified into the following three categories:

a) Manual
b) Semi-automatic
c) Automatic

a) Manual method(algorithm),

- The object of interest is observed by an expert who traces its ROI boundaries as well, with the help of software. Hence, the decisions related to segmentation are made by the human observers. Many software systems assist experts in tracing the boundaries and extracting them. By using the software systems, the experts outline the object. The outline can be either an open or closed contour. The software provides help to the user in extracting the closed regions.
- Manual method of extraction is time consuming, highly subjective, prone to human error, and has poor intra-observer reproducibility. However, manual methods are still used commonly by experts to verify and validate the results of automatic segmentation algorithms.

b) Automatic segmentation algorithms

- Automatic segmentation algorithms are a preferred choice as they segment the structures of the objects without any human intervention. They are preferred if the tasks need to be carried out for a large number of images.

c) Semi-automatic algorithms

- Semi-automatic algorithms are a combination of automatic and manual algorithms. In semi-automatic algorithms, human intervention is required in the initial stages.

Normally, the human observer is supposed to provide the initial seed points indicating the ROI. Then the extraction process is carried out automatically as dictated by the logic of the segmentation algorithm.

- Region-growing techniques are semi-automatic algorithms where the initial seeds are given by the human observer in the region that needs to be segmented. However, the program process is automatic. These algorithms can be called assisted manual segmentation algorithms.

2. Based on pixel similarity relationships with neighbouring pixels. The similarity relationships can be based on colour, texture, brightness, or any other image statistic. On this basis, segmentation algorithms can be classified as follows:

a) Contextual (region-based or global) algorithms
b) Non-contextual (pixel-based or local) algorithms

a) Contextual algorithms

- Contextual algorithms group pixels together based on common properties by exploiting the relationships that exist among the pixels.
- These are also known as **region-based or global algorithms**.
- In region-based algorithms, the pixels are grouped based on some sort of similarity that exists between them.

b) Non-contextual algorithms

- Non-contextual algorithms are also known as **pixel- based or local algorithms.**
- These algorithms ignore the relationship that exists between the pixels or features. Instead, the idea is to identify the discontinuities that are present in the image such as isolated lines and edges. These are then simply grouped into a region based on some global-level property. Intensity-based thresholding is a good example of this method.

10. Explain the three fundamental types of gray-level discontinuities in digital images. 12M CO5 L2

In digital image processing, gray-level discontinuities refer to abrupt changes in intensity (gray level) in an image. These discontinuities are critical for edge detection and other image analysis tasks. There are three fundamental types of gray-level discontinuities:

1. **Point Discontinuities**:
   - These occur when the intensity of a single pixel differs significantly from its neighboring pixels. This kind of discontinuity is often referred to as an isolated point or an impulse noise.
   - For example, in an otherwise uniform region, if one pixel has a much higher or lower intensity than the surrounding pixels, it forms a point discontinuity.

   Steps for Point Detection

   i. **Mask Definition**:
   - A mask or kernel is defined, usually a small matrix, that slides over the image to examine each pixel and its neighbors. Commonly used masks for point detection include the Laplacian and other high-pass filters.

   ii. **Convolution**:
   - The mask is convolved with the image. This involves placing the center of the mask over each pixel and computing a weighted sum of the pixel values covered by the mask.

   iii. **Thresholding**:
   - The result of the convolution is compared against a threshold. Pixels where the convolution result exceeds the threshold are marked as points of interest (potential point discontinuities).

2. **Line Discontinuities**:
   - These happen along a line of pixels where the intensity changes abruptly, forming a ridge or trough. This can be seen as a sudden transition in intensity along a specific direction.
   - Line discontinuities are useful in identifying thin lines or edges in images, such as the boundaries of objects.

3. **Edge Discontinuities**:
   - Edges are the most common type of gray-level discontinuity. An edge represents a boundary between two regions with different intensities.
   - There are different types of edges, including:
     - **Step Edge**: A sudden transition between two different intensity levels. It looks like a sharp step in the intensity profile.
     - **Ramp Edge**: A more gradual transition between two intensity levels, creating a sloped appearance.
     - **Roof Edge**: A rapid but not instantaneous transition that creates a peak-like shape in the intensity profile.
     - **Ridge Edge**: Similar to a roof edge but with a more pronounced ridge-like shape.

These discontinuities are fundamental for various image processing tasks such as edge detection, feature extraction, and image segmentation. Identifying and analyzing these discontinuities help in understanding the structure and features within an image.

11. Describe the concept of an "edge" in image processing, and how does it contribute to the understanding and analysis of digital images? Classify the types of edges in the digital image. 10M CO5 L3

An edge in image processing refers to the boundary or transition between two regions with different intensity levels in a digital image. Edges are significant as they represent important local changes in the image and are often associated with the boundaries of objects within the scene.

Importance of Edges:

1. **Feature Extraction:** Edges help in identifying and extracting key features from an image, such as shapes and object boundaries.
2. **Segmentation:** They assist in dividing an image into meaningful regions for analysis.
3. **Object Recognition:** Edges provide important information for recognizing and classifying objects within an image.
4. **Image Enhancement:** Enhancing edges can improve the visual quality of an image and make specific features more prominent.

Types of Edges in Digital Images

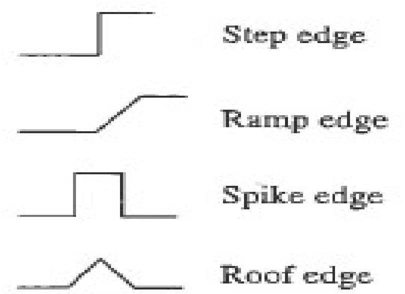


Fig. 9.7 Types of edges

Edges can be classified based on the nature of the intensity transition:

1. **Step Edge**:
   - A sudden, abrupt change in intensity between two regions.
   - Represented by a sharp step in the intensity profile.
   - Idealized as a perfect vertical or horizontal step in a one-dimensional signal.
2. **Ramp Edge**:
   - A more gradual transition between two regions of different intensities.
   - The intensity changes linearly over a certain distance, forming a slope.
   - Common in real images where transitions are not perfectly sharp.
3. **Ridge(Spike) Edge**:
   - Characterized by a rapid change in intensity that creates a peak-like shape.
   - Similar to roof edges but more pronounced and sharp.
4. **Roof Edge**:
   - A rapid, but not instantaneous, transition in intensity that forms a ridge or a peak.
   - The intensity increases to a maximum point and then decreases, creating a roof-like shape.
   - Common in images where objects have smooth and rounded features.

12. Write a python program to read an image and extract and display low-level features such as edges, textures using filtering techniques. 8M CO5 L4

```
import cv2
from matplotlib import pyplot as plt
import numpy as np

# Load the image

img = cv2.imread("D:\\puppy.jpg")
img=img[:,:,::-1]

# Convert the image to grayscale
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

# Edge detection
edges = cv2.Canny(gray, 100, 200)  # Use Canny edge detector

# Texture extraction
kernel = np.ones((5, 5), np.float32) / 25  # Define a 5x5 averaging kernel
texture = cv2.filter2D(gray, -1, kernel)  # Apply the averaging filter for texture extraction

# Display the original image, edges, and texture

plt.figure(figsize=(7, 8))
plt.subplot(2, 2, 1)
# showing image
plt.imshow(img)
plt.axis('off')
plt.title("Original Image")
plt.subplot(2, 2, 2)
# showing image
plt.imshow(edges)
plt.axis('off')
plt.title("Edges")
plt.subplot(2, 2,3)
# showing image
plt.imshow(texture)
plt.axis('off')
plt.title("Texture")
```

13. Write a python program to contour an image. 8M CO5 L4

```
import cv2
from matplotlib import pyplot as plt
import numpy as np
fig = plt.figure(figsize=(20, 10))
rows = 3
columns = 2
# Reading an image in default mode
src = cv2.imread('D:\\puppy.jpg')
gray = cv2.cvtColor(src, cv2.COLOR_BGR2GRAY)

# Find Canny edges
edged = cv2.Canny(gray, 30, 200)

contours, hierarchy = cv2.findContours(edged,cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_NONE)

fig.add_subplot(rows, columns, 1)

plt.imshow(edged)
plt.axis('off')
plt.title("Canny Edges After Contouring")

print("Number of Contours found = " + str(len(contours)))

# Draw all contours
# -1 signifies drawing all contours
cv2.drawContours(src, contours, -1, (0, 255, 0), 3)


fig.add_subplot(rows, columns, 2)

# showing image
img11= src[:,:,::-1]
plt.imshow(img11)
plt.axis('off')
plt.title("Contours")
```

14. Explain the different types of edge detectors. 10M CO5 L4

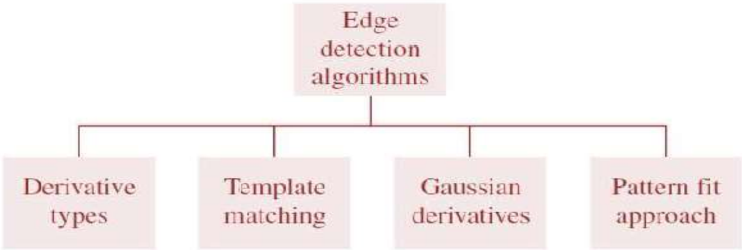


Fig. 9.9 Types of edge detectors

Edge detection is a fundamental tool in image processing, computer vision, and machine vision, particularly in the areas of feature detection and feature extraction, which aim at identifying points in a digital image at which the image brightness changes sharply or more formally has discontinuities. These points are often associated with the boundaries of objects in a scene. There are 4 types of edge detection algorithms (shown in figure 9.9); each with its own approach is used to identifying edges in an image:

1. Gradient (or Derivative) Filters:

- Gradient filters are based on the principle that edges in an image can be detected by looking for local maxima and minima of the image's intensity gradient.
- The gradient is a multi-variable generalization of the derivative, which measures how much the image's intensity changes in a given direction.
- Common gradient filters include the **Sobel operator, Prewitt operator, and Roberts Cross operator.**
- These filters typically use a pair of convolution kernels to approximate the gradient in the horizontal and vertical directions, and then combine these to calculate the gradient magnitude and direction.

2. Template Matching Filters:

- Template matching filters work by comparing a small region of the image (the "template") with a set of predefined templates that represent different edge orientations.
- The template that best matches the local image region is selected, and the corresponding edge orientation and strength are determined.
- This method is computationally intensive because it involves matching the image region with multiple templates.
- An example of a template matching filter is the **Kirsch operator**, which uses a set of eight templates to detect edges at different orientations.

3. Gaussian Derivatives:

- Gaussian derivative filters involve convolving the image with derivatives of Gaussian functions.
- This approach is particularly useful for detecting edges at different scales.
- By varying the standard deviation (σ) of the Gaussian function, the filter can be tuned to detect edges over a range of scales.
- The Gaussian derivative approach is robust to noise because the Gaussian function smoothes the image while enhancing edges.

- The **Laplacian of Gaussian (LoG) and Difference of Gaussians (DoG)** are examples of edge detectors that use Gaussian derivatives.

4. Pattern Fit Approach:

- The pattern fit approach to edge detection involves fitting a predefined pattern or model to the local image data.
- This method assumes that edges have a certain shape or profile, and it attempts to fit this profile to the image data.
- The goodness of fit determines the presence and strength of an edge. This approach can be more complex than other methods because it requires a model of what the edges should look like.
- An example of a pattern fit approach is the use of deformable models or active contours, which are curves that move within the image to lock onto edges or boundaries.