Edge and Contour detection

Tushar B. Kute,
http://tusharkute.com

Edges

- Edges are significant local changes of intensity in a digital image.
- An edge can be defined as a set of connected pixels that forms a boundary between two disjoint regions.
- There are three types of edges:
  - Horizontal edges
  - Vertical edges
  - Diagonal edges

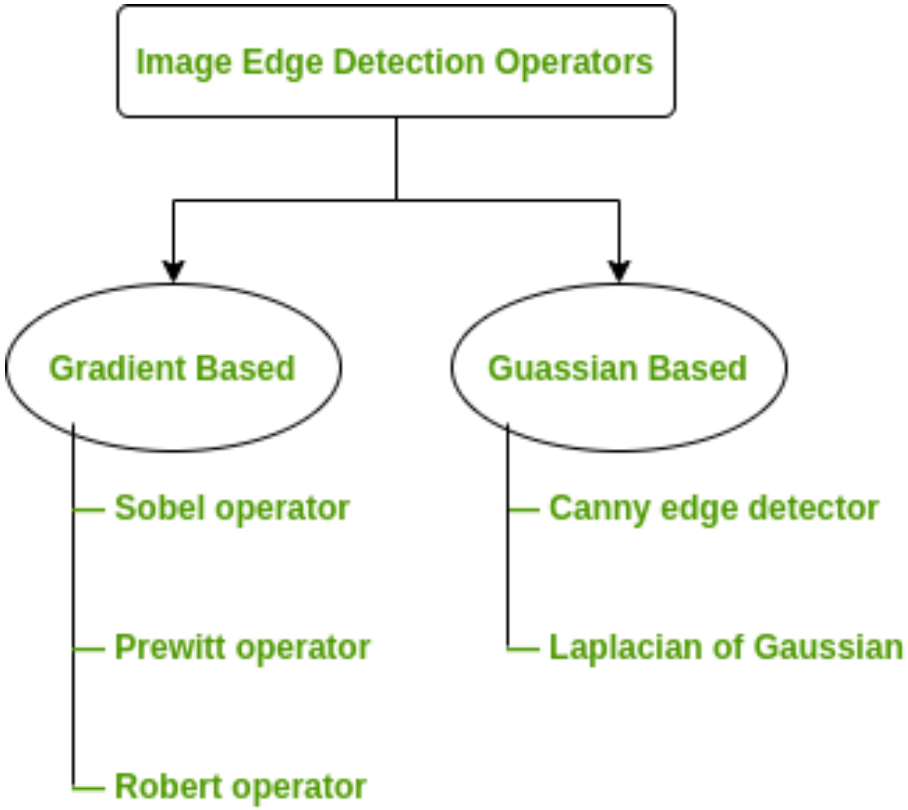
Edge Detection

- Edge Detection is a method of segmenting an image into regions of discontinuity. It is a widely used technique in digital image processing like
  - pattern recognition
  - image morphology
  - feature extraction
- Edge detection allows users to observe the features of an image for a significant change in the gray level. This texture indicating the end of one region in the image and the beginning of another.
- It reduces the amount of data in an image and preserves the structural properties of an image.

Edge Detection

- Edge Detection Operators are of two types:
  - Gradient –
    - based operator which computes first-order derivations in a digital image like, Sobel operator, Prewitt operator, Robert operator
  - Gaussian –
    - based operator which computes second-order derivations in a digital image like, Canny edge detector, Laplacian of Gaussian

Edge Detection



Edge Detection: How?

- Image smoothing: The image is first smoothed to reduce noise and enhance edges. This is often done using filters like Gaussian blur.
- Gradient calculation: The gradient of the image is calculated, which measures the change in intensity between neighboring pixels. Edges are typically located where the gradient magnitude is high.
- Edge thresholding: A threshold is applied to the gradient magnitude to identify pixels that are likely to be part of an edge.
- Edge linking: The identified edge pixels are connected to form continuous edge lines.

Edge Detection?

Sobel Edge Detection

- It is a discrete differentiation operator. It computes the gradient approximation of image intensity function for image edge detection.
- At the pixels of an image, the Sobel operator produces either the normal to a vector or the corresponding gradient vector.
- It uses two 3 x 3 kernels or masks which are convolved with the input image to calculate the vertical and horizontal derivative approximations respectively

Sobel Edge Detection

- It uses two 3 x 3 kernels or masks which are convolved with the input image to calculate the vertical and horizontal derivative approximations respectively –

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

Sobel Edge Detection: Steps

- Convolution with kernels:
  - The Sobel algorithm uses two 3x3 kernels, one for horizontal edges and one for vertical edges. Each kernel element has a specific weight assigned to it.
  - These kernels are convolved with the grayscale image, meaning they are slid across the image, and the weighted sum of the corresponding pixel values under the kernel mask is calculated at each pixel location.
  - The horizontal kernel emphasizes intensity changes in the horizontal direction, while the vertical kernel emphasizes changes in the vertical direction.

Sobel Edge Detection: Steps

- Gradient calculation:
  - The convolution with each kernel results in two gradient images, one representing the horizontal gradient and another representing the vertical gradient.
  - Each pixel in these gradient images holds the magnitude of the intensity change in the corresponding direction.

Sobel Edge Detection: Steps

- Combining gradients:
  - The final edge strength at each pixel is calculated by combining the horizontal and vertical gradients. This can be done using different approaches:
  - Magnitude: Square root of the sum of squared horizontal and vertical gradients.
  - Gradient direction: arctangent of the vertical gradient divided by the horizontal gradient.

Sobel Edge Detection: Steps

- Thresholding:
  - A threshold is applied to the combined gradient image to classify pixels as edge pixels or non-edge pixels.
  - Pixels with gradient values exceeding the threshold are considered edges.
  - The threshold value can be chosen manually or determined automatically using techniques like Otsu's method.

Sobel Edge Detection: Steps

- Post-processing (optional):
  - Additional steps like non-maximum suppression can be applied to refine the detected edges by removing single-pixel edge segments and ensuring only the strongest edges remain.

Sobel Edge Detection:

- Practical

Scharr Edge Detection

- Scharr edge detection is an image processing technique used to identify and highlight edges or boundaries of objects within an image.
- It is a type of gradient-based edge detection, which means it calculates the magnitude and direction of the gradient at each pixel in the image.
- Edges are then identified as pixels with high gradient magnitudes.

Scharr Edge Detection

- The Scharr operator is a 3x3 filter that is applied to the image to calculate the gradient.
- The filter kernel for the Scharr operator in the x-direction is:

[-3 0 3]

[-10 0 10]

[-3 0 3]

Scharr Edge Detection

- The filter kernel for the Scharr operator in the y-direction is:

  [-3 -10 -3]

  [0 0 0]

  [3 10 3]

Scharr Edge Detection

- The Scharr operator has several advantages over other gradient-based edge detectors, such as the Sobel operator.
  - First, it is more rotationally symmetric, which means that it is less sensitive to the orientation of edges in the image.
  - Second, it has a slightly higher response to edges, which can improve the accuracy of edge detection.

Scharr Edge Detection: Summary

- Convolution with Scharr filters:
  - Applies 3x3 Scharr filters to the image to compute gradients in two directions:
  - Horizontal filter (dx = 1, dy = 0) emphasizes vertical edges.
  - Vertical filter (dx = 0, dy = 1) emphasizes horizontal edges.
- Gradient magnitude calculation:
  - Combines the horizontal and vertical gradients using the following formula: gradient_magnitude = sqrt(dx^2 + dy^2)

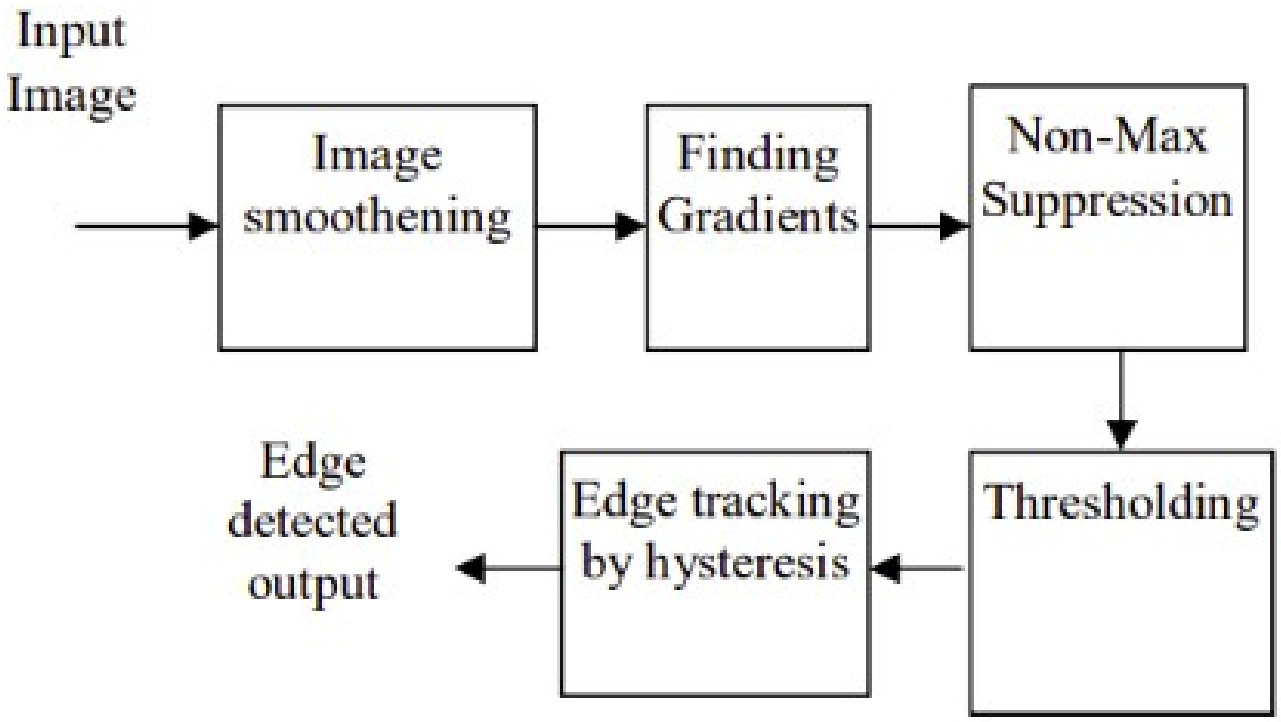
Scharr Edge Detection

- Practical

Canny Edge Detection

- A multi-stage edge detection algorithm developed by John F. Canny in 1986.
- Aims to find wide range of edges while considering factors like:
  - Good detection: Finding real edges accurately.
  - Localization: Placing detected edges precisely.
  - Minimizing single responses: Not detecting the same edge multiple times.
  - Minimizing noise: Not detecting noise as edges.

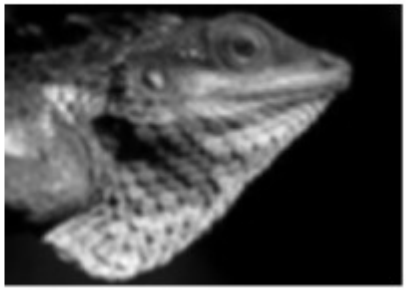
Canny Edge Detection



Canny Edge Detection: Example

Canny Edge Detection: Example

- Image Smoothening
  - In this step, we convert the image to grayscale as edge detection does not dependent on colors.
  - Then we remove the noise in the image with a Gaussian filter as edge detection is prone to noise.



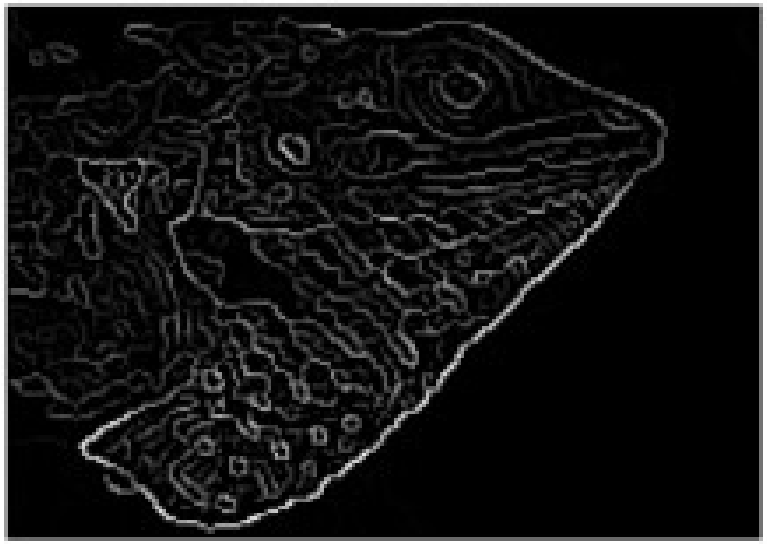
Canny Edge Detection: Example

- Finding Intensity Gradients of the Image
  - We then apply the Sobel kernel in horizontal and vertical directions to get the first derivative in the horizontal direction (Gx) and vertical direction (Gy) on the smoothened image. We then calculate the edge gradient(G) and Angle(θ) as given below,

    $Edge\_Gradient(G) = \sqrt{(Gx^2+Gy^2)}$

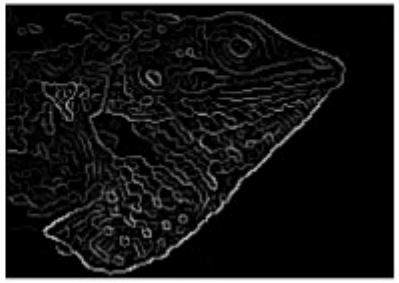
    $Angle(\theta)=\tan^{-1}(Gy/Gx)$
- We know that the gradient direction is perpendicular to the edge. We round the angle to one of four angles representing vertical, horizontal, and two diagonal directions.

Canny Edge Detection: Example



Canny Edge Detection: Example

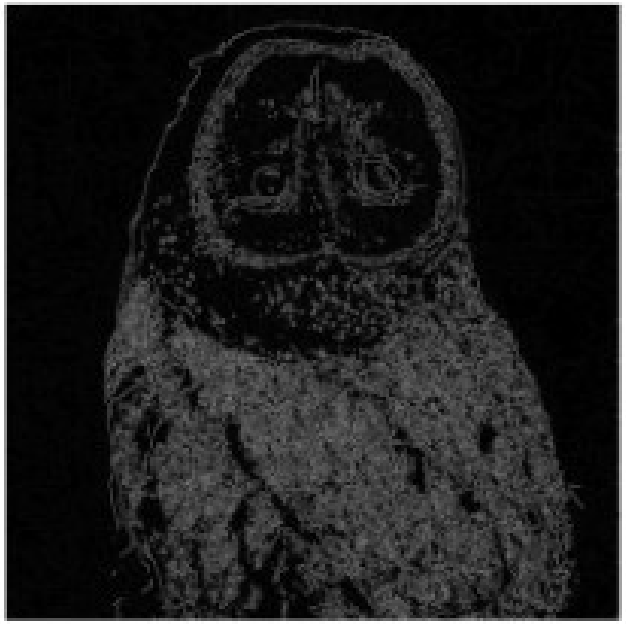
- Non-Max Suppression
  - Now we remove all the unwanted pixels which may not constitute the edge.
  - For this, every pixel is checked in the direction of the gradient if it is a local maximum in its neighbourhood.
  - If it is a local maximum, it is considered for the next stage, otherwise, it is darkened with 0. This will give a thin line in the output image.

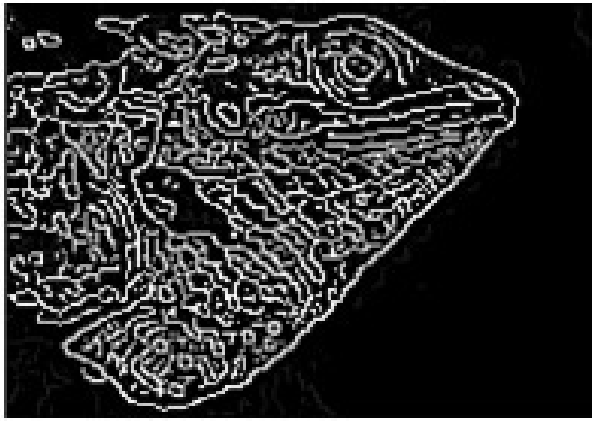


Canny Edge Detection: Example

- Double Threshold
  - Pixels due to noise and color variation would persist in the image. So, to remove this, we get two thresholds from the user, lowerVal and upperVal.
  - We filter out edge pixels with a weak gradient(lowerVal) value and preserve edge pixels with a high gradient value(upperVal).
  - Edges with an intensity gradient more than upperVal are sure to edge, and those below lowerVal are sure to be non-edges, so discarded.
  - The pixels that have pixel value lesser than the upperVal and greater than the lowerVal are considered part of the edge if it is connected to a “sure-edge”. Otherwise, they are also discarded.

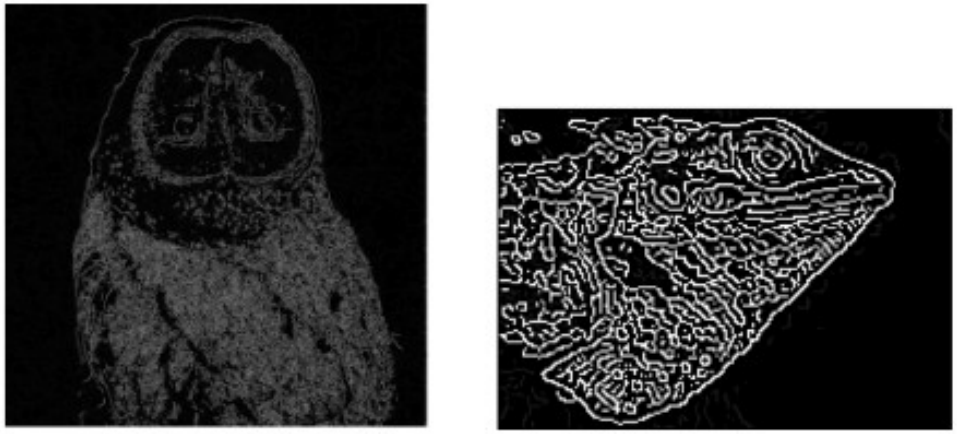
Canny Edge Detection: Example





Canny Edge Detection: Example

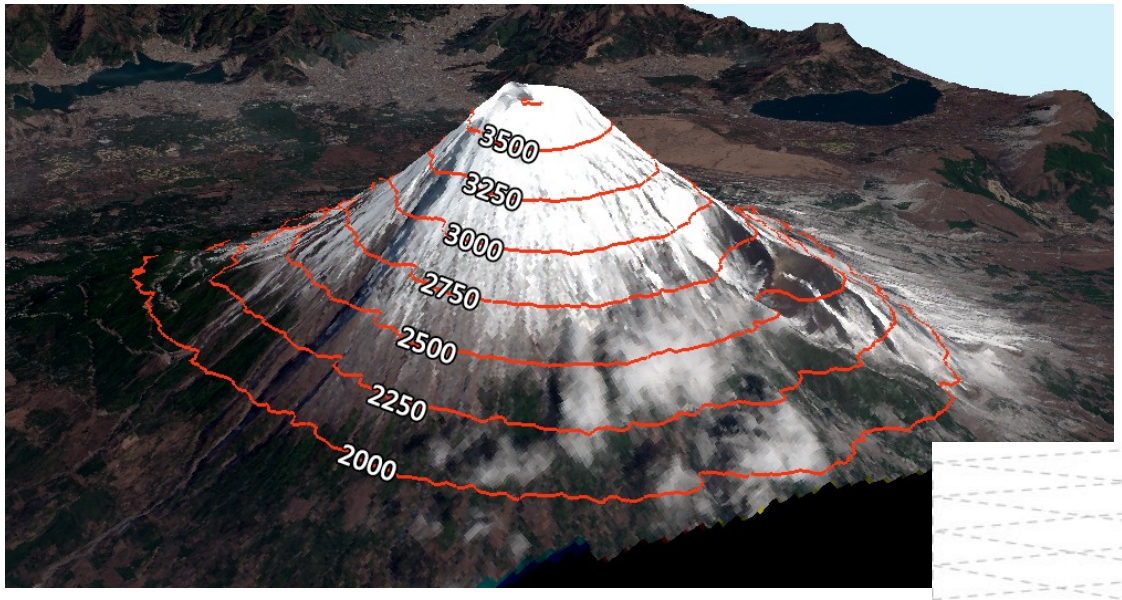
- A pixel is made as a strong pixel if either of the 8 pixels around it is strong(pixel value=255) else it is made as 0.
- That’s pretty much about Canny Edge Detection. As you can see here, the edges are detected from an image.

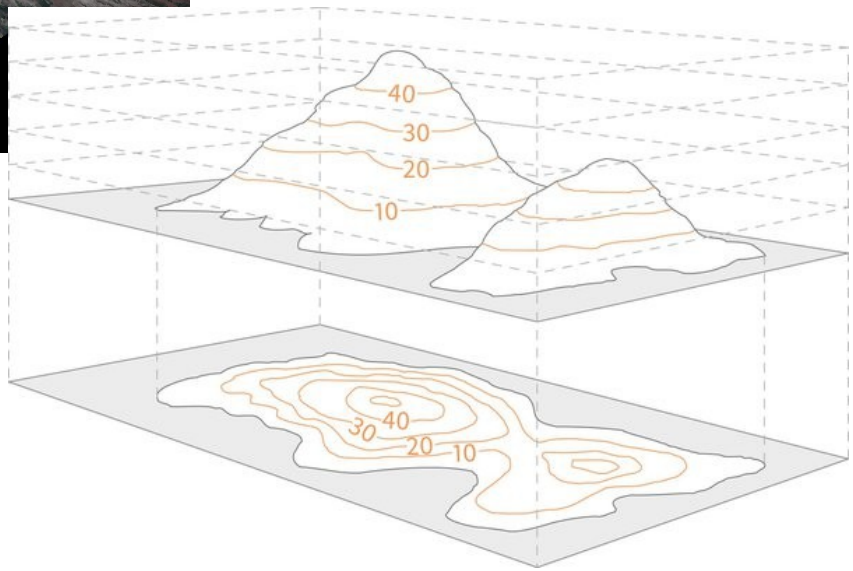


Contours

- Contours can be explained simply as a curve joining all the continuous points (along the boundary), having same color or intensity.
- The contours are a useful tool for shape analysis and object detection and recognition.

Contours





Contours

- Continuous sequences of connected pixels that share similar intensity or color values.
- Think of them as outlines or "shorelines" that separate objects from their background.
- Often represented as a list of coordinates defining the boundary path.

Contours

- How are they detected?
  - Various algorithms exist, typically involving:
    - Edge detection: Identifying pixels with significant intensity changes, likely marking object boundaries.
    - Pixel tracing: Connecting these edge pixels into continuous contours.
    - Filtering and simplification: Removing noise and refining the contour representation.

Contours

- Types of contours:
  - Closed contours: Form a complete loop, enclosing an object fully.
  - Open contours: Start and end at image borders, representing partial object boundaries.

Contours

- Examples of contour usage:
  - Detecting fruits and vegetables in a grocery image.
  - Analyzing cell shapes in microscopy images.
  - Recognizing traffic signs in road scenes.
  - Tracking people in a surveillance video.

Contours Detection

- Practical

Thank you

@mitu_skillologies

@mITuSkillologies

@mitu_group

@mitu-skillologies

@MITUSkillologies

kaggle

@mituskillologies

Web Resources
https://mitu.co.in
http://tusharkute.com

@mituskillologies

contact@mitu.co.in

tushar@tusharkute.com