Objectives

In this reading assignment you should be able to learn,

- Edge detection (What and why)
- Types of edge detection
- Algorithm
- Applications

Prerequisites

- Basic calculus
- Features detection
- Basic image properties

Introduction

Let's consider the following image. Do you recognize the image? Probably, it won't take much time for you to recognize this image as a maze. This is how powerful our recognition capabilities are.

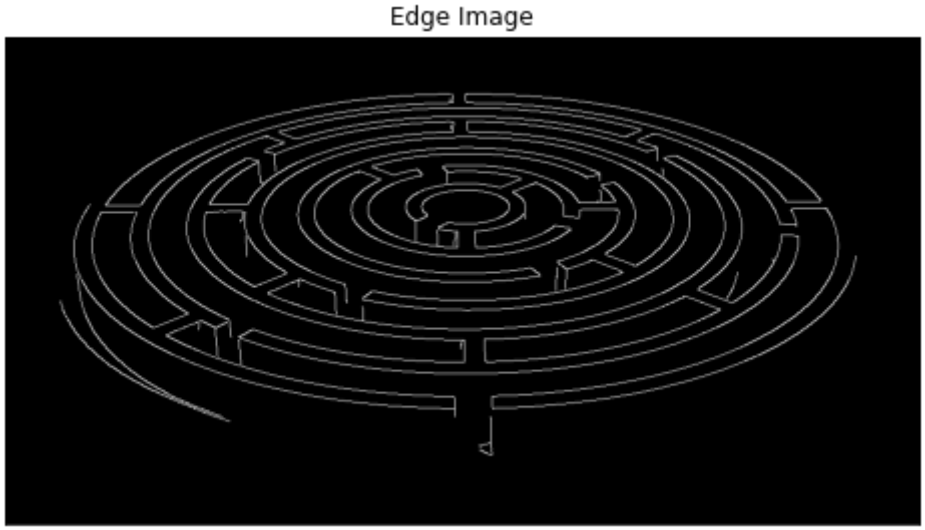

Fig. 1: Output image after edge detection

A computer cannot visualize an image the way a human does. Each image is interpreted as an array of pixel values. Each pixel value can range from 0 to 255, representing black to white pixels, respectively. On this basis, the computer interprets an image in various ways.

The feature detection technique is one of the major approaches used for recognition that detects the most important features of the image. Some of the features are interest points, edges, lines, etc.

Among the many essential features of an image, as mentioned above, detecting an edge is crucial for various recognition tasks.

Edges are the area where intensity between the pixels changes sharply. Although interest points carry an essential feature of an image, detecting interest points is not always a good strategy.

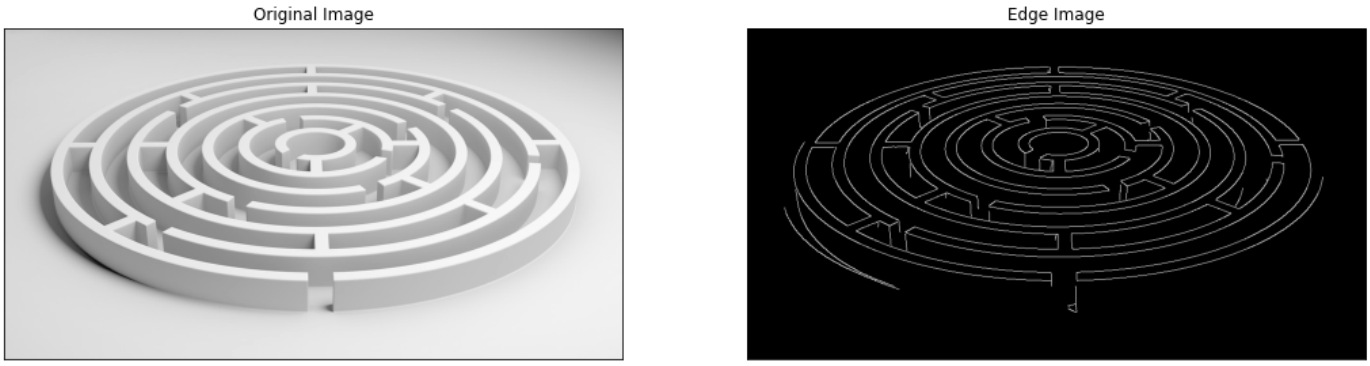

Fig. 1: (a) Original image (left) (b) Output image after edge detection (right)

Why to detect edges?

Detecting an edge plays very important role in various computer vision applications. Some of the purposes behind edge detection are,

- To create a line drawing of an image scene.
- To extract essential features of edges from an image (such as corners, curves, boundaries, lines, etc.).
- To generate features for various computer vision algorithms (such as object recognition)

In the above figure, the right image is produced by applying edge detection technique in the left image. You can see that we can still recognize the image on the right as a maze from its outlines.

Since there are continuous intensity changes along the edges, it will be easier to apply gradient methods along the edge to detect it. Hence, detecting an edge is preferred than interest points in these conditions.

Edges intensity modeling

Now, let's see how continuous intensity changes along the edges. Edges can be divided into different types based on their intensity, shown in the figure below.

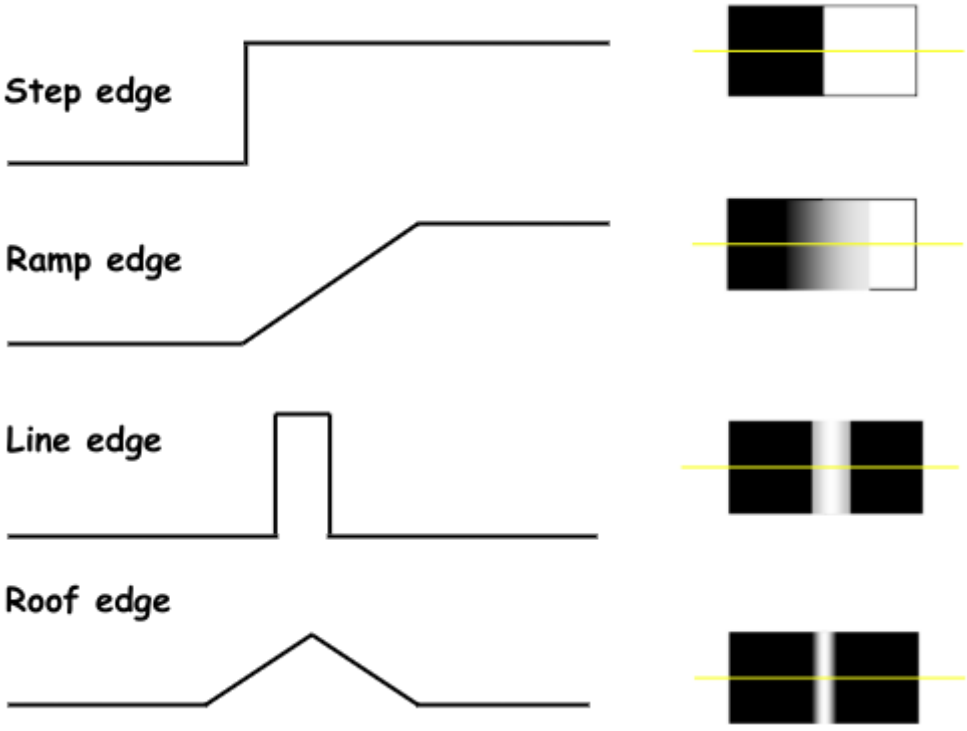

Fig. 2: Edge intensity profiles

1. Step edge: The image intensity suddenly changes from one side to another side of an image.

2. Ramp edge: It is a kind of step edge where image intensity doesn't change suddenly but occur over a finite distance.
3. Line edge: The image intensity suddenly changes and return to previous value with some short distance.
4. Roof edge: It is a kind of Line edge where image intensity doesn't change suddenly and occur very short distance.

Why intensity changes?

So, you got the idea behind how intensity changes along the edges and now let's explore why intensity changes. There are mainly two events for intensity changes. They are,

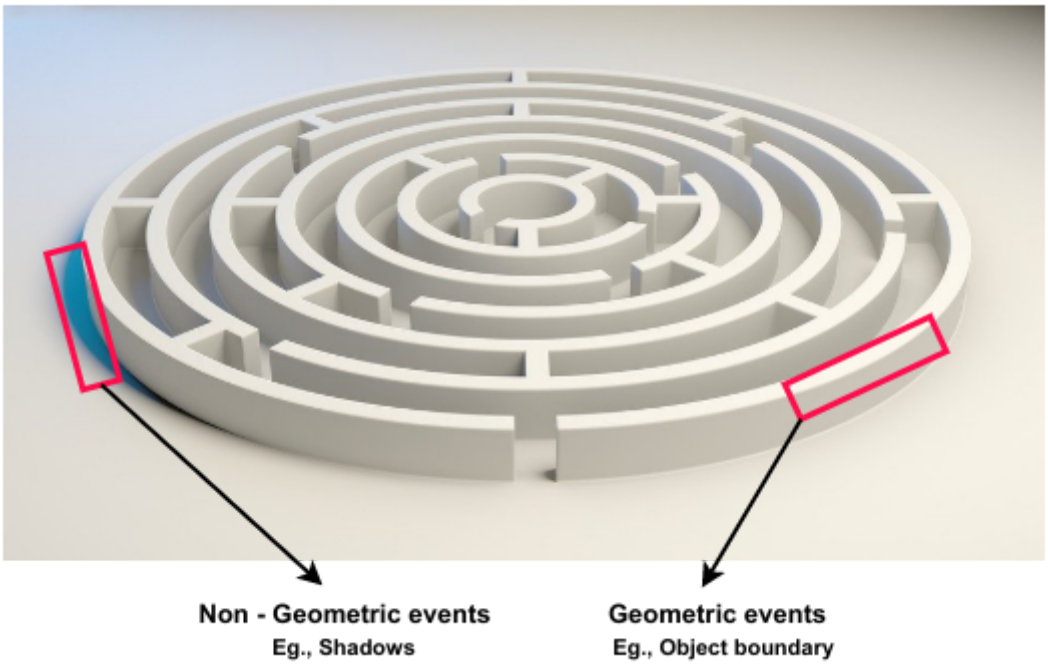

Fig. 3: Edge intensity profiles

- Geometric events

> These events create sharp intensity changes due to the physical properties of an object and surface.

  - Object boundary: Boundary around the objects define the shape of the object as well as create sharp intensity changes.
  - Surface boundary: In the same way as object boundary, the surface boundary also produces sharp intensity changes.
- Non-geometric events

> These events occurred due to the effect of light on objects and surfaces to produce various intensity changes.

  - Shadows: shadows are created when objects block the light and produce sharp intensity changes behind the object.
  - Inter-reflections: inter-reflections also create sharp intensity changes due to object surface reflection.

You can visualize all these events and their effect on intensity changes in the above figure.

Edge descriptors

We know each image is interpreted as an array of pixel values by computer. The following figure shows how a small portion of an image is interpreted by a computer. Each box is represented as a pixel. Now, let's understand the edge descriptors in terms of pixels.

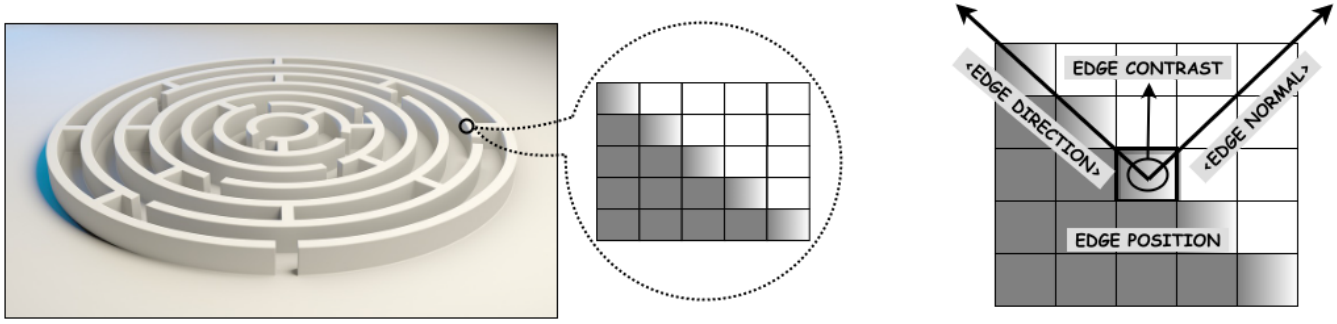

Fig. 4: Edge descriptors

Edge descriptors are those which describe properties of an edge. There are generally four edge descriptors. They are,

- Edge normal: It is a unit vector in the direction of maximum intensity changes.
- Edge direction: It is a unit vector in the direction of the same intensity and perpendicular to the edge normal vector.
- Edge position: It is the coordinate or position of the pixel at which edge is located.
- Edge contrast: It is the local image contrast along the normal, which value typically ranges between 0 to 255.

All these descriptors can be visualized through the above figure.

Algorithm for edge detection

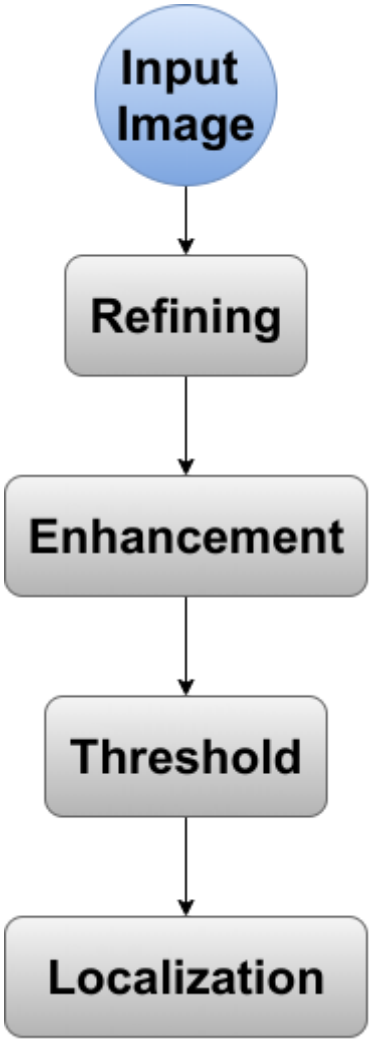

Fig. 5: Edge detection algorithm

1. Take an image: First of all, the colored image should be taken for detection.
2. Refining: The purpose of refining is to remove the noises from the image without eliminating true edges.
3. Enhancement: In this approach, various filters or differentiation techniques are applied to enhance the edges' quality.
4. Threshold: Here, threshold determines which edge pixels to be discarded as noise and which one to be retained.
5. Localization: The exact location of an edge is determined in this process. For eg., Edge thinning, Edge linking, etc.

Criteria for Optimality

There are generally 3 primary criteria for checking the optimality of an edge detection algorithm. They are,

- Good detection

> To satisfy good detection criteria, the algorithm should minimize the probability of false positives (eg., spurious edges) as well as false negatives (eg., missing real edges).

- Good localization

> The detected edges of an image by an algorithm should be as close as possible to the original edges.

- Single response

> The algorithm should minimize the number of local maxima around the original edges.

The following figure illustrates failing of these three criteria respectively.

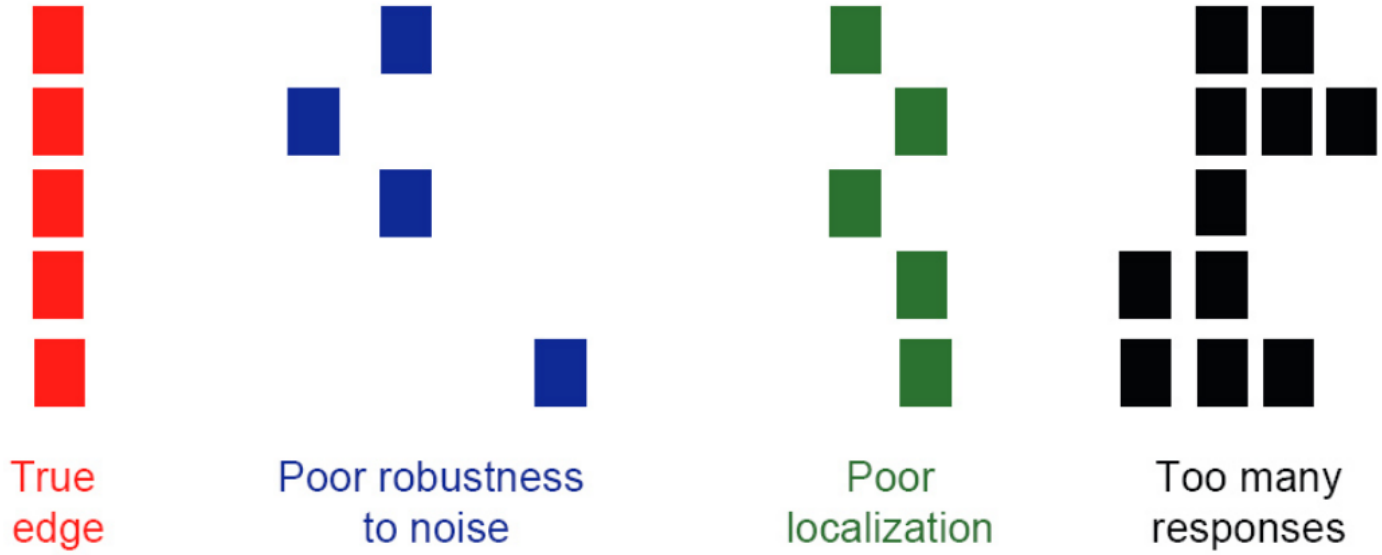

Fig. 6: Edge detection algorithm

Edge detection using derivatives

In calculus, gradient or derivative describes the changes of continuous functions. With the help of derivative or gradient approach, the edge can be detected in the following two ways,

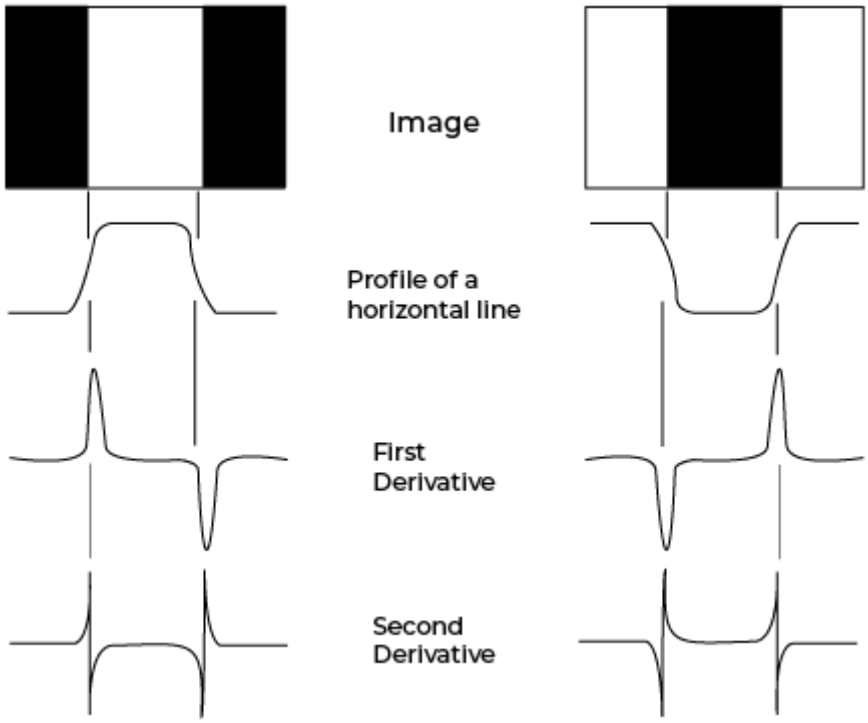

Fig. 7: Edge detection using derivatives

- Using 1st derivatives

> The point or pixel lies on edge, can be identified by detecting local minima or maxima of the first derivatives.

- Using 2nd derivatives

> The point or pixel lies on edge, can be identified by detecting zero crossings of the second derivatives.

Types of Edge Detection

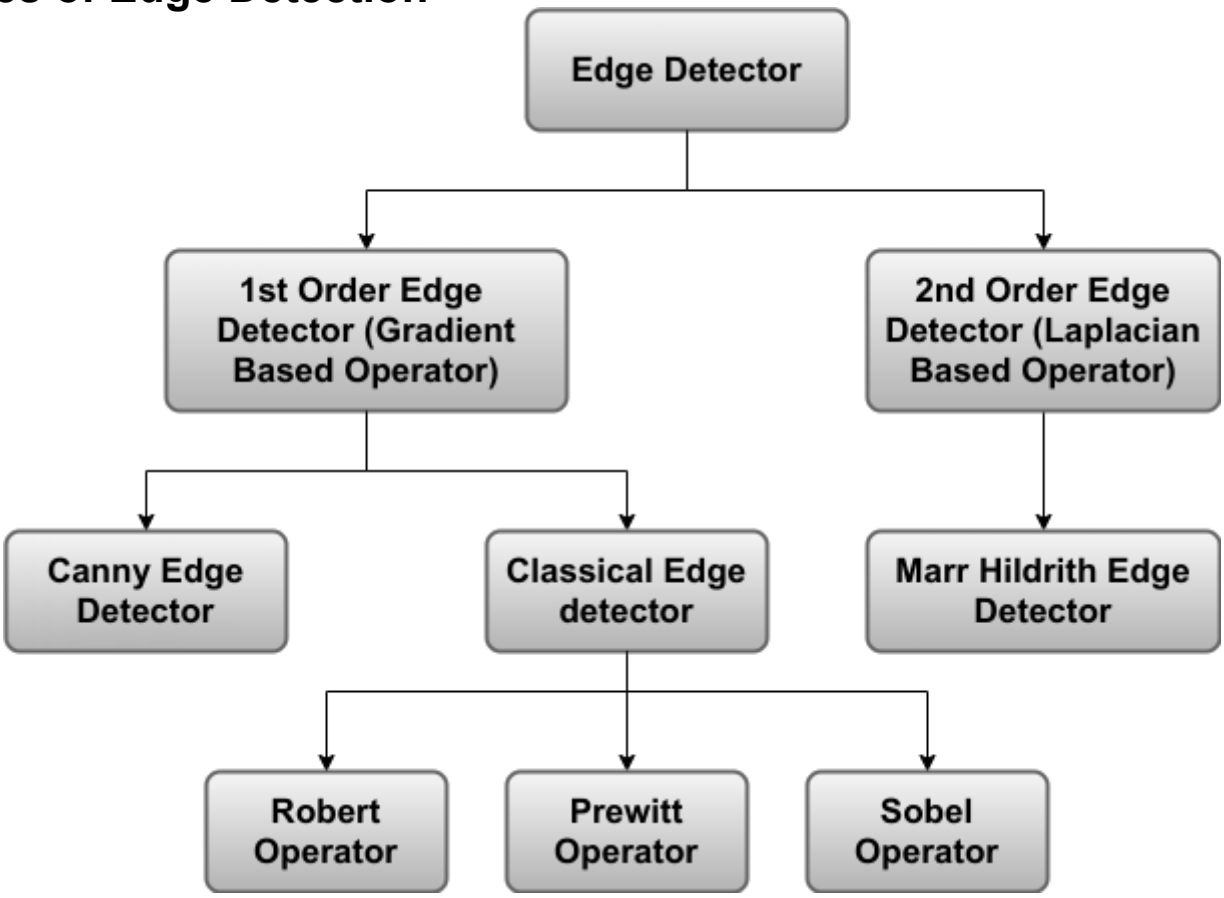

Fig. 8: Edge Detection Types

Types of edge detection can be broadly divided into two major categories.

- First-order edge detector (Gradient-based operator)
- Second-order edge detector (Laplacian based operator)

In this module, we will talk about only a first-order edge detector.

Classical Edge Detector

Classical edge detector falls under the first-order edge detector category, which is also known as a gradient-based operator. The following three operators, Robert, Prewitt, and Sobel, are classified as classical operators. Although they are easy to operate, they are sensitive to the noise.

- Sobel Operator

> This is a discrete differentiation operator used to compute an approximation of the gradient of image intensity function. The kernels used by this operator are,

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

Now, let's understand how these kernels used.

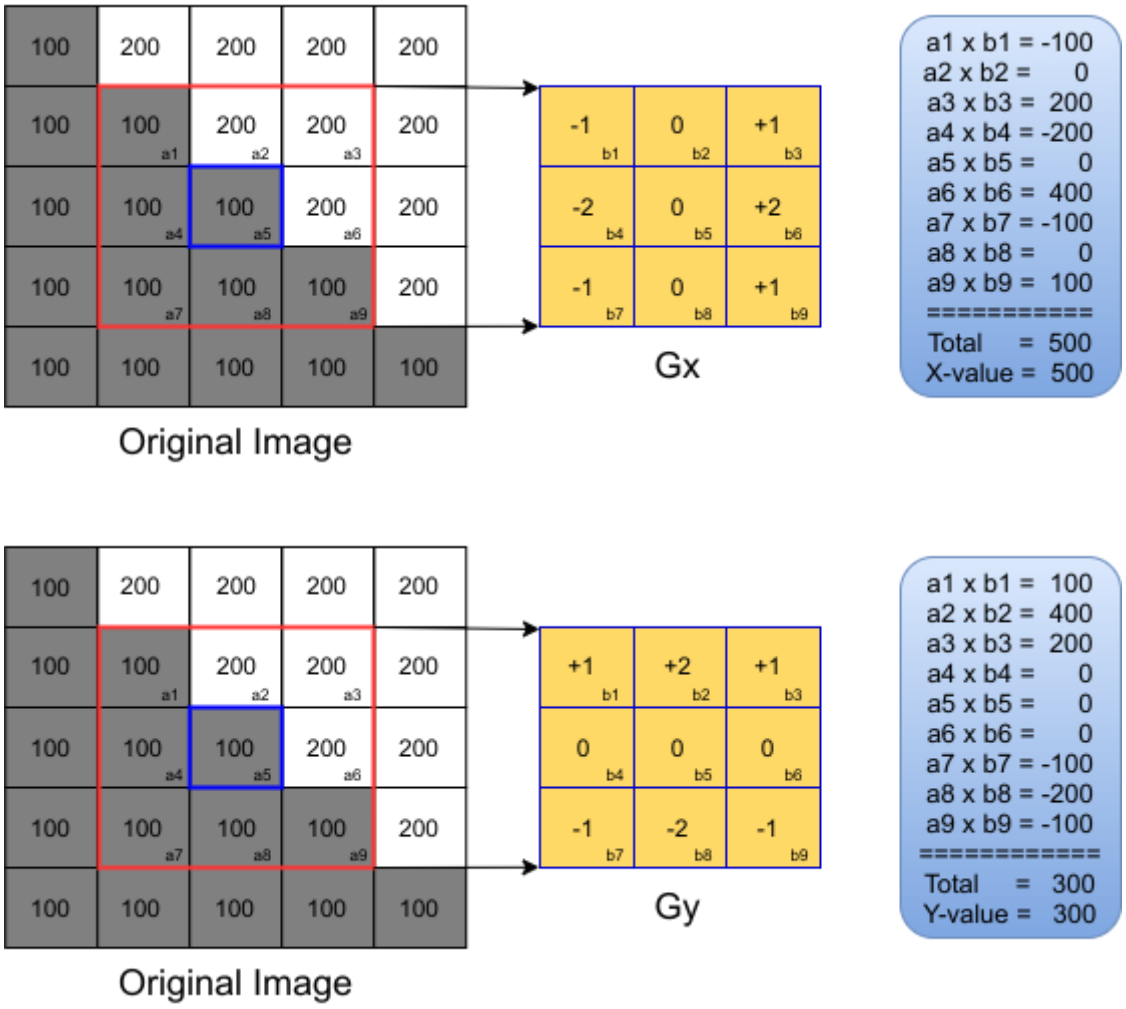

Fig. 9: Sobel operator calculation

- Robert Operator

This is a gradient-based operator. The kernels used by this operator are,

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

- Prewitt Operator

It is the same as the Sobel operator but has different kernels. The kernels used by this operator are,

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

Summary

We understand the basics of edge detection, and it's types. In the next chapter, we will discuss Canny edge detection, which solves the problems of classical edge detectors.

Key takeaways

- Why edges are important?
  - Edges are the areas of maximum intensity changes abruptly.
  - To extract lines, boundary, corners which are useful in many computer vision applications
- How edges forms? Why intensity changes?
  - Geometric events (eg., object boundary, surface bounday)
  - Non-geometric events (eg., shadows, inner-reflections)
- Criteria for optimality
  - Good detection
  - Good localization
  - Single response
- Edge detection using derivaties
  - Using 1st derivatives (local maxima or local minima of 1st derivatives)
  - Using 2nd derivatives (zero crossing of 2nd derivatives)
- Classical edge detector
  - Robert operator
  - Prewitt operator
  - Sobel operator

References

- Papers
  - Rashmi, Mukesh K., Rohini S. (2013), Algorithm and technique on various edge detection: a survey (https://aircconline.com/sipij/V4N3/4313sipij06.pdf), Vol.3, SIPIJ
    - Figure 2 shows algorithm for classical detector.
- Books
  - Richard S. (2010), Computer Vision: Algorithms and Applications (http://szeliski.org/Book/drafts/SzeliskiBook_20100903_draft.pdf), 2010 Draft, Springer
    - Check pages 211 - 213 to understand the foundations and mathematics of edge detection in detail.
- Lectures
  - Trucco, Jain, et al., Edge detection (https://www.cse.unr.edu/~bebis/CS791E/Notes/EdgeDetection.pdf)
    - Check pages 8 - 12 to understand classical operators in detail with examples.