EDGE DETECTION IN DIGITAL IMAGES USING SOFT COMPUTING TECHNIQUES

A THESIS

SUBMITTED TO THE DELHI TECHNOLOGICAL UNIVERSITY
FOR THE AWARD OF THE DEGREE OF

DOCTOR OF PHILOSOPHY

Sahil Raheja

DEPARTMENT OF INFORMATION TECHNOLOGY
DELHI TECHNOLOGICAL UNIVERSITY
DELHI, INDIA

2024

EDGE DETECTION IN DIGITAL IMAGES USING SOFT COMPUTING TECHNIQUES

A THESIS

Submitted to Delhi Technological University
for the award of the degree of

DOCTOR OF PHILOSOPHY

By

SAHIL RAHEJA
(2K14/PHD/IT/03)

Under the Joint Supervision of

Supervisor
Dr. Kapil Sharma
Professor
Department of IT
Delhi Technological University (DTU),
New Delhi -110042

Co-Supervisor
Dr. Akshi Kumar
Senior Lecturer & Director-Post Graduate Research (PGR)
Deptt. Of Computing
Goldsmiths, University of London,
United Kingdom

DEPARTMENT OF INFORMATION TECHNOLOGY
DELHI TECHNOLOGICAL UNIVERSITY
DELHI, INDIA

2024

DECLARATION

I, Sahil Raheja, Ph.D. student (Roll No. 2K14/Ph.D/IT/03), hereby declare that the thesis entitled "Edge Detection In Digital Images Using Soft Computing Techniques" which is being submitted for the award of the degree of Doctor of Philosophy in Information Technology, is a record of bonafide research work carried out by me in the Department of Information Technology, Delhi Technological University. I further declare that this work is based on original research and has not been submitted to any university or institution for any degree or diploma.

Place: DTU, Delhi **Sahil Raheja**

Date: (2K14/PHD/IT/03)

CERTIFICATE

This is to certify that the work embodied in the thesis entitled **"Edge Detection In Digital Images Using Soft Computing Techniques"** being submitted by Sahil Raheja to the Department of Information Technology, Delhi Technological University, Delhi, for the award of the degree of Doctor of Philosophy is a record of bonafide research work carried out by him under our guidance and supervision. In our opinion, the thesis has reached the standards fulfilling the requirements of the regulations relating to the degree. The results contained in this thesis have not been submitted to any other university or institute for the award of any degree or diploma.

Supervisor
Dr. Kapil Sharma
Professor
Department of IT
Delhi Technological University (DTU),
New Delhi -110042

Co-Supervisor
Dr. Akshi Kumar
Senior Lecturer & Director-Post Graduate Research (PGR)
Deptt. Of Computing
Goldsmiths, University of London, United Kingdom

Dedicated

To

My Father……

ACKNOWLEDGEMENT

First and foremost, praises and thanks to GOD, the almighty, for Divine showers of blessingson me for successful completion of my research work.

This thesis is the end of the journey in obtaining my Ph.D and at the end of my thesis, it is a pleasant task to express thanks to all those who contributed in many ways to the success of this study and made it an unforgettable experience for me.

I express my greatest thanks to the GOD for the blessings at every step of my life.

I would like to express my heartfelt gratitude to my supervisors, Prof. (Dr.) Kapil Sharma and Dr. Akshi Kumar. I am greatly thankful to them for their continuous motivation, support and guidance. They both always encouraged and helped me during ups and downs in my PhD journey. Their expertise in taking out the best out of a student is beyond appreciation and their inspiration can lift any student to excellence. I would also like to thank them for being very patient with me and for having faith in me.

I consider myself very fortunate to have Dr. Akshi Kumar as my supervisor. I have learned a lot from her over the past few years. The time spent with her and the conversation we had really boosted my spirits and ignited the passion of learning. I could not have imagined having a better guide and mentor for my Ph.D.

I am grateful to the members of my Department Research Committee for reviewing my work and for their valuable comments. I would also like to thank to the faculty and staff of the department, for their assistance and encouragement during my research work.

It is very difficult to thank enough to my entire friends including Nitin Jain, Gulshan Srivastava and Shiv Naresh Shivhare. They always rendered their unconditional help to me in all aspects including research and otherwise. I also want to thank, Rajni Sethi, Himanshu Sharma and Gopal Katala. Their support in my personal and professional life made me complete this work.

Thanks to Delhi Technological University, Delhi, for providing the facilities and scholarship during the period of my research work, which was very helpful in the successful completion of this research work.

To my two sweet mothers, Mrs. Shanta Raheja for always being there as a pillar of strength and support to manage my personal and professional life and my mother in law, Mrs. Anita Rani for their blessings.To my two fathers, Mr. Subhash Raheja who always wanted me to be a doctor and supported me in my every decision and my father-in-law, Mr. Ashok Kumar for always motivating me during this journey. I am forever indebted to my parents for their unfettered love, invaluable blessings and unwavering belief in me. I dedicate this thesis to my parents. Thanks to almighty for being there as my parents. This thesis is your dream come true. I owe them everything.

To my sisters Neetika and Mahak, brother-in-law Sumit and Vikas for cheering me and motivating me. I feel myself very lucky to have a great family who have supported me throughout the journey of my research work.

Last but not the least, without whom I could not imagine being enrolled in PhD, my wife Mrs. Tejasvi Raheja, for her unwavering encouragement, undeterred faith in me and being the biggest pillar of strength who supported me all the way till the end. She was always there to motivate me in my tough time. A special thanks to my children Divit and Pavika, for bearing my absence and always loving me. It would not have been possible to complete my research work without their love, encouragement and support.

TABLE OF CONTENTS

Title *Page No.*

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

ELA	:	Edge Localization Accuracy
NMS	:	Non-Maximum Supression
SNR	:	Signal to Noise Ratio
pdf	:	Probability Density Function
SNN	:	Symmetrical Nearest Neighbor
MHN	:	Maximum Homogeneity Neighbor
CAF	:	Conditional Averaging Filter
BF	:	Bilateral Filter
WGIF	:	Weighted Guided Image Filter
ABF	:	Adaptive Bilateral Filter
AGIF	:	Adaptive Guided Image Filter
GBF	:	Guided Bilateral Filter
ACO	:	Ant Colony Optimization
BSD	:	Barkley Segmentation Database
ANN	:	Artificial Neural Network
PSO	:	Particle Swarm Optimization
SVM	:	Support Vector Machine
GA	:	Genetic Algorithm
QWAF	:	Quaternion weighted average filter
COM-SOBEL	:	Center of Mass with Sobel Operator
FCD	:	Fractional-order Charef differentiator
DOG	:	Difference of Gaussian
LOG	:	Laplacian of Gaussian
SUSAN	:	Smallest Univalue Segment Assimilating Nucleus
AD	:	Anisotropic Diffusion
GAP	:	Gradient-Adjusted Predictor
DNN	:	Deep Neural Network
EA	:	Evolutionary Algorithms

MSE	:	Mean Square Error
CNN	:	Convolution Neural Networks
ABC	:	Artificial Bee Colony
CSO	:	CAT Swarm Optimization
HED	:	Holistically-Nested Edge Detection
BEL	:	Boosted Edge Learning
gt	:	Ground Truth
ed	:	Edge Detected Image
TP	:	True Positive
FP	:	False Positive
TN		True Negative
FN		False Negative
FoM	:	Figure of Merit
SSIM	:	Structure Similarity Image Metrics
MSE	:	Mean-Squared Error
PSNR	:	Peak Signal to Noise Ratio
HoD	:	Hausdoff Distance
E_D	:	Euclidiean Distance
BDM	:	Baddeley's Delta metric
P_r	:	Precision
R_c	:	Recall
USC	:	University of Southern California
SIPI	:	Signal and Image Processing Institute
BFOA	:	Bacteria Foraging Optimization Algorithm

ABSTRACT

Analyzing the contents of a real-world view is an essential task which needs to be carried out in machine vision and image processing, that has gained a lot of interest from the researchers in the past four decades. Edge detection algorithm calculates the contours of an object, separates it from its background and helps in analyzing the contents of the image. This represents the significance of edge detection in the area of machine vision. The method of edge detection involves in locating the points in a digital image where abrupt changes in image intensity or discontinuities occurs.

There are different conventional algorithms useful for edge detection in various points of view. Some of the popular algorithms are Sobel, Canny, and Roberts etc. All of these traditional methods are based on a gradient estimation for the pixels. These techniques are less complex but fail in the presence noise. These techniques are highly dependent on the threshold which results in missing, false and spurious edges. Several other techniques are also proposed in the literature. For edge detection, soft-computing based techniques had recently gained much popularity. In these methods, artificial neural networks (ANN), Fuzzy logic, deep neural network (DNN) and evolutionary algorithms (EA) based on swarm's behaviors' are the most common. The ANN, and DNN based methods are complex and very susceptible to noise perturbations. In some of soft-computing methods, weak edges are also not detected properly. To overcome these limitations, a two step process for edge detection is considered. In the first step, edge refinement is done, and then edge detection is done using soft computing technique.

In our first work, an edge detection method based on Ant Colony Optimization (ACO) is described. A novel intensity mapping function is utilized to record intensity change among neighbouring pixels which guides the movement of ant. Finally, the Peak-Signal-to-Noise Ratio (PSNR), accuracy, and F-Score are used to evaluate and compare performance.

In the second work, fuzzy logic-based edge detection approach using a sharpening guided filter is proposed. A Gaussian filter is also used to deal with noise caused by sharpening. A range of statistical indicators are used to assess the method's accuracy.

It has been discovered that by properly setting the smoothening parameters, a significant improvement can be achieved in the identified edges. Simulation results are presented on various images, statistical results are evaluated and compared with other latest techniques.

In the third work, images are smoothed at varying degrees using a guided L_0 smoothen filter and then a fuzzy logic-based edge detection algorithm is used to detect edges. Simulation results for Canny, Sobel, fuzzy logic-based edge detection, and lastly fuzzy logic edge detection with L_0smoothen filter are shown. Results are contrasted with both traditional and contemporary methods. More than 100 images are taken into account from the Berkley Segmentation Database (BSD) and USC-SIPI Image Database. The measured F-value reaches a maximum of 0.848.

Finally in fourth work, the edge detection considering guided image filtering and ACO is discussed. Simulation results are presented on various images and statistical results will also be evaluated and comparisons with latest techniques are also done.

In this research, an effort is made to propose an edge detection algorithm which can work well in presence of noise. The performance of proposed edge detection techniques is analyzed with other notable techniques. The simulation is performed on BSD (Berkeley Segmentation Dataset) and USC-SIPI Image Database using computer simulation in MATLAB. Summarized table for result using different parameters achieved by proposed method and other edge detection techniques shows that the proposed method improves the performance of edge detection. Due to better performance of proposed method; it can be used in different field of science and engineering. In addition, possible directions for further developments are outlined.

Chapter One
Introduction

Machine vision is getting more & more significance every day because of its importance in many applications like the production and testing of various industrial parts, medical image analysis, robotics, etc. It is the process of designing machines to behave like human beings in terms of seeing and interpreting real-world scenes. Analyzing the contents of a real-world view is an essential task that needs to be carried out in machine vision and image processing, that has received a great interest from the researchers in the past four decades.

An image represents a real-world scene with objects of different sizes, shapes, orientations, and colours. The first step towards analyzing the contents of the picture isto separate all the image's content from the background. To accomplish this, one must calculate the object's contour present in the image. For calculating it, edge detection is to be carried out to detect all the edges forming that object which represents the significance of edge detection in the area of machine vision. Edges are used to estimate the object's boundaries, which helps in segmentation and scene interpretation. For example, human facial features, fingerprints and the body shape of an object can be defined by edges. (Ziou & Tabbone, 1998)

Edge detection has utility in several image processing applications, including image morphing, enhancement, restoration, recognition, compression, retrieval, watermarking, etc. An edge detection method detects the existence and location of edges in the image. Applying the edge detection algorithm on an image may be termed mapping a 2-D image into a set of lines or curves so that the output gives a more compact representation of it yet helpful for understanding. (M, 2022).

1.1 Need of Edge Detection

The need for edge detection arises from its crucial contribution in various image processing and machine vision tasks. Here's why edge detection is essential:

1. **Feature Extraction:** Edges represent significant changes in intensity or color within an image, serving as key features for subsequent analysis. By accurately identifying edges, important information such as object boundaries, shapes, and textures can be extracted, facilitating higher-level image understanding and interpretation.
2. **Object Recognition and Segmentation:** Edge detection forms the basis for object recognition and segmentation algorithms. Detecting and delineating object boundaries allow for precise localization and extraction of objects from complex backgrounds, enabling tasks such as object tracking, classification, and scene understanding.
3. **Image Enhancement:** Edges often correspond to salient image structures and details. Enhancing edges through techniques like edge sharpening or contrast enhancement can improve image visual quality, making it easier for humans or automated systems to perceive and analyze important image content.
4. **Medical Imaging:** In medical imaging applications such as X-rays, MRI, and CT scans, accurate edge detection is crucial for identifying anatomical structures, lesions, and abnormalities. It aids in diagnosis, treatment planning, and monitoring disease progression.
5. **Industrial Inspection:** Edge detection has a signigicant place in quality control and defect detection across various industries. From manufacturing to automotive and electronics, precise identification of edges helps detect flaws, measure dimensions, and ensure product consistency and reliability.
6. **Robotics and Autonomous Systems:** For robots and autonomous systems to navigate and interact with their environment effectively, they need to perceive and understand their surroundings. Edge detection provides essential spatial information for obstacle avoidance, path planning, and object manipulation tasks.

In essence, edge detection acts as a fundamental building block in numerous image analysis and machine vision applications, enabling tasks ranging from basic image enhancement to advanced scene understanding and autonomous decision-making. Its importance lies in its ability to extract meaningful visual information critical for further processing and interpretation.

1.2 Edge Detection and Basic concepts

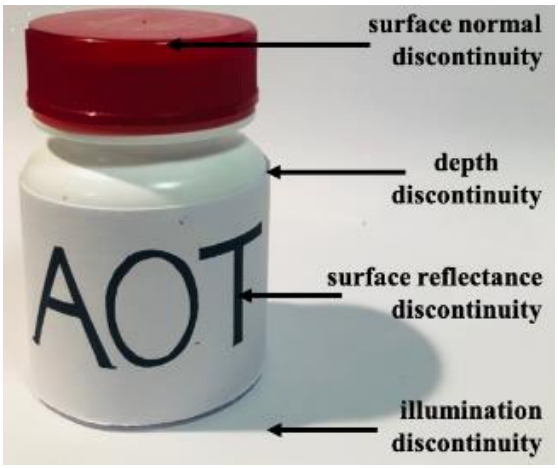
1.2.1 Edge

A quick and noticeable change in visual brightness is referred to as an edge. Since pixel intensity value represents image brightness, an edge point can be considered a place with a significant shift in intensity value. The boundaries of the image's objects are typically where these modifications take place.

1.2.2 Origin of Edges

Edges correspond to significant photometrical, physical and geometrical variations of the objects in a scene, providing vital visual information. According to (Marr & Hildreth, 1980), the change in intensity value occurs mainly because of the geometry, viewpoint, illumination and reflectance. Edges are the place where there is discontinuity occurs in image intensity. These discontinuities may occur because of the following reasons, which are explained in the forthcoming section:

a. **Surface Normal discontinuity:** This type of discontinuities occurs because of the surface normal. For example, as shown in Figure 1.1, the bottle cover is cylindrical, and as you move around this cover, the surface's normal changes. After a certain point, the discontinuity occurs in the surface's normal direction. Although the surface remains continuous but the discontinuity appears as an edge to the human eye.



Source: https://www.cs.toronto.edu/~guerzhoy/320/lec/edgedetection.pdf

Figure 1.1: Origin of Edges

b. **Depth discontinuity:** The gap in depth also leads to a discontinuity, which may be perceived as an edge to the human eye.

c. **Surface colour discontinuity:** occurs because of variations in the surface's colour or appearance.

d. **Illumination discontinuity:**This type of discontinuity occurs because of light/shadows reflecting on the images.

The discontinuities because of Surface Normal and Depth reflects the significant changes in the object's geometry, while surface colour and illumination discontinuities are primarily because of photometric reasons.

1.2.3 Type of Edges

Different types of edges may be present in the image. It can be one out of four major types (Ruslau, Pratama, & Meirista, 2019). These are important because only given edge information is sufficient for object recognition.

a. **Step Edge:** A step edge in an image usually occurs between the background and foreground of the picture. If we identify a region different from its surrounding, a step edge exists between two neighbouring pixels; if one belongs to region one and the other belongs to the surrounding area. The transition in intensity should be instantaneous to result in a step edge.

b. **Ramp Edge:**It is another common type of edge in real-world images. In this, the transition in intensity between a region and its surroundings spans over the number of pixels. The change is not instantaneous as in the case of step edge; instead, it occurs over many pixels.

c. **Roof Edge:** It occurs if the intensity profile of the image gradually increases and, after a certain point, it starts decreasing gradually. This type of edge does not happen because of object boundaries. Instead, it happens because of a change in surface orientation regarding the illumination source.

d. **Spike Edge:** It is composed of two-step edges of different signs and occurs over a short interval.

All four types of edges are depicted in Figure 1.2:

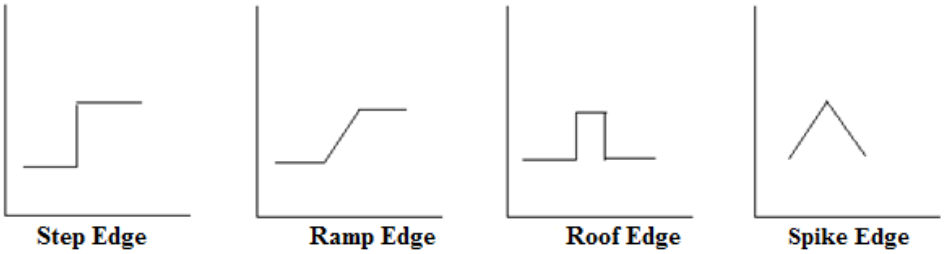


Figure 1.2: Types of Edges

1.2.4 The Motivation behind Edge Detection

Image processing applications have matured in the past few decades. The need for more advanced applications is necessitated as computing power increases. Research in image processing aims to provide accurate and effective methods. Many such applications, including face recognition, content-based image retrieval, image processing-based security and medical image processing, depends greatly on the success of edge detection. Therefore, it is the main focus of this research. Step edges are easy to locate since they correspond to sharp changes in image intensity, usually occurring at object boundaries, orientation etc. Edge profiles are not as smooth as described since noise which often exists in images, degrades the image's visual quality, making the edge profiles smoother and deflecting it from its actual path. The aim of edge detection is to produce an accurate connected edge line with a finer orientation, which is to be done without changing the image's meaning. In the ideal case, the edge detection algorithm should return a connected set of lines and curves, usually the image's boundary information. The successful execution of edge detection simplifies the subsequent task of image analysis, classification, and other image processing applications may therefore be substantially simplified. However, it is not always possible to get accurate edge detection from real-world images as they are often very complex and usually corrupted by noise.

Imperfections in lens optics, sampling, and image collection systems can all contribute to the blurry edge (Gongalez, Melin, Castro, Mendoza, & Castillo, 2016). Edges extracted from real-world images often suffer from the problem of fragmentation, i.e., broken edge curves, false negatives, misplaced edge segments,

incorrect edges, etc., which complicates the further image processing tasks of interpreting the image's data. An edge map not only decreases the amount of the data to be processed, but also it preserves crucial structural details like shapes, object orientation, etc. This image format is simple to utilize as an input for other sophisticated image processing tools in machine vision. The desired output of an edge detection algorithm should be well-connected accurate edge lines and curves with finer orientation, and much work has been done on this in the past few decades.

1.2.5 Criteria for Good Edge Detection

Different edge detection operators are available, but not a single edge detection operator is efficient in detecting all kinds of edges. Each one is developed to be more sensitive to a specific edge type. However, no standard performance measure can calculate the image edges' quality. An edge detection algorithm's performance is typically assessed subjectively and uniquely depending upon the application. However, the output with a thin edge line and few speckles can be considered good edge detection.

An edge detection algorithm's performance can be evaluated objectively, for which some criterias had been discussed in the literature, out of which few can be expressed mathematically. At the same time, others depend on a particular application's requirements. Quantitative evaluation of performance is only possible when the ground truth is available. The following are a few cases that can be considered to measure performance quantitatively. (Chawla & Khokhar, 2015)

Good detection: The edge detection operator should result in fewer fake edges. The edge pixels are usually calculated after thresholding operation. The choice of a threshold should be optimum since there is always a trade-off. If a low value is selected, more pixels are detected, leading to many false edges; similarly, if a high threshold is chosen, false edges can be avoided at the cost of missing true edges. Therefore, fine-tuning the threshold value is crucial and can not be set arbitrarily.

Noise sensitivity: Noise usually corrupts real-world images which degrades the quality of the picture. The edge detection algorithm should be more immune to noise.

It should be able to work in a noisy (Gaussian, Salt & pepper noise etc.) environment. Since edges and noise are high-frequency components, the noise gets amplified using an edge detection operator. So, noise suppression becomes essential before applying any edge detection operator. Some pre-processing tasks, like image denoising, image smoothening, image sharpening, etc., may be performed to reduce the effect of noise or make actual edges sharp, thus creating a clear distinction between edge pixels and noisy pixels. Some post-processing steps like non-maximum suppression and a thinning procedure may also be used to get an approximate better output.

Good localization: The effect of blurring and smoothening may result in displacing an edge pixel from its actual location, which is not desired. If edge pixels are not appropriately localized, it may result in incorrect output. The edge localization should be as accurate as close to actual position, also called edge-localization accuracy (ELA).

Orientation sensitivity: The purpose of an edge detection operator is not just to calculate gradient magnitude, it should also calculate its orientation correctly. The finding of the accurate direction of an edge point is also helpful in connecting edge segments, accurately detecting noisy pixels, and accurately performing thinning procedures like Non-Maximum suppression (NMS).

Speed and efficiency: An edge detection algorithm has many applications like medical image processing, image segmentation etc. It may be used in real-time applications like fire detection, vehicle motion analysis etc. The speed should be fast enough to be used in a particular application.

The five given criteria are practical when choosing between different edge detection algorithms. The task of detecting accurate edges is both vital and crucial for further high-level processing tasks. Ideally, a real edge point should be identified, and a non-edge pixel should not be detected as an edge. A real optimum threshold point must be determined before any edge detector can be used. The higher threshold value helps reject noise as edges but leads to the false rejection of genuine edge pixels. The signal-to-noise ratio (SNR) improves when real edge pixels are recognized, and spurious edges are suppressed.

1.3 Edge Detection Techniques

An edge detection algorithm receives an image as input and produce the binary image as output, called an edge map. The output of some algorithms may also contain explicit information to maintain other attributes of the edges, like position, strength, and orientation. From a technical perspective, the edge detection algorithms can be classified under two different categories which are gradient based and zero-crossing based (Kumar, Upadhyay, Dubey, & Varshney, 2021), explained as follows:

1.3.1 Gradient Based

These methods use first-order derivative methods to compute the gradients in the image (Gonzalez & Woods, 1992). Then, the search looks for strong gradient magnitude, the local maximum in the direction that matches the edge profile. These methods depend heavily on the threshold; thus, accuracy is limited.

1.3.2 Zero-crossing based/Laplacian

The second-order derivative method, such as the Laplacian method, is used to calculate the 2^{nd}-derivative of the image. Then a search is made to look for zero-crossings to find the edges.

In noisy images, the edges exhibit a ramp-like profile (Sridhar, 2016). If the intensity profile is like a ramp, then the first derivative is positive, and for constant intensity first derivative is zero (Gupta & Mazumdar, 2013). The 2^{nd} derivative is+ve along the darker side of the edge and -ve along the lighter side, similar to delta functions. It's also 0 outside the ramp and along the ramp. So, regarding the edge's derivatives, it can be concluded that the first derivative indicates whether the edge pixel is present at a point.

In contrast, the second derivative provides information about whether the edge pixel is present on the brighter or darker side (Nalwa & Binford, 1986). To overcome the difficulties of the first derivative, 2^{nd} derivative can be considered. The 2^{nd} derivative can be used to derive two more properties: The first, it provides several values for an edge. Secondly, it also shows a zero-crossing property. Specifically, it displays the

zero in the middle of the edge. As a result, it aids in detecting the thick edge's centres (Gupta & Mazumdar, 2013).

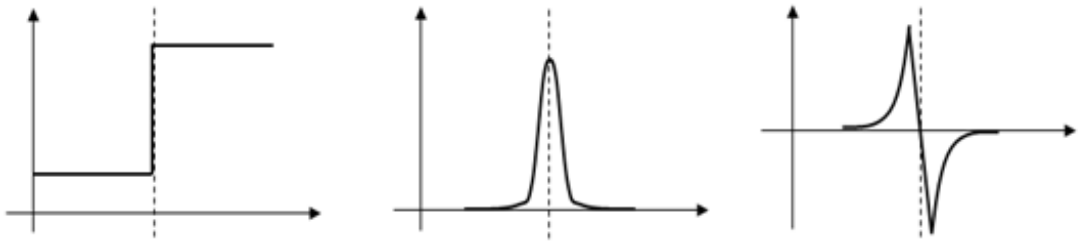


Figure 1.3: (a) Step Function (b) First Derivative (c) Second Derivative

The derivative in Figure 1.3(b) displays a maximum in the original signal, which is positioned at centre of the edge.The Sobel edge detection technique also comes under the gradient filter,"a family of edge detection filters". As previously mentioned, edge pixel's intensity are greater than the intensity value of surrounding pixels, therefore, when the gradient value of a particular pixel exceeds a particular threshold, it is identified to be an edge pixel. Also the second derivative is 0, when the first derivative is at its maximum. As a result, finding the zeros in the second derivative is another method for locating an edge. The Laplacian approach is used, and Figure 1.3(c) shows the second derivative of the signal.

1.4 Variables of an edge detection operator

Many edge detection operators are available, each sensitive to a particular edge. The following are a few variables that are involved in choosing a specific edge detection operator (Pinho & Luis, 1997):

- **Orientation of Edges**: The operator's form determines which direction it responds to edges the most strongly. To find vertical, horizontal, or diagonal edges more effectively, operators can be employed.
- **Noise environment**: Noise distorts the quality of the image. In noisy images, detecting accurate edges is very difficult because the noise and edge pixels include high-frequency components. Noise reduction leads to distorted and

dislocated edges. On noisy images, operators are often wider in scope, allowing them to average enough data to disregard localized noisy pixels. As a result, the detected edges are less accurately localized.

- **Edge structure:** A step shift in intensity is not present at all edges. Objects with borders with slow variation in intensity might result from effects like refraction or inadequate focus. In certain instances, the operator must be selected to respond to such a slow change.

1.5 Steps for Edge Detection

Structurally, edge detection is usually performed in three steps: image pre-processing, differentiation and localization (Oskoei & Hu, 2010). Directly implementing an edge detection algorithm on the source image may not result in a good result because of noise, so the images need to be pre-processed. In pre-processing, image smoothing is used to decrease the impact of noise. This step is usually performed by filtering through a low-pass filter. Since noise and edges are high-frequency components, the noise reduction technique can remove both of them. A parameter that balances noise reduction and preservation of edge information is generally used.

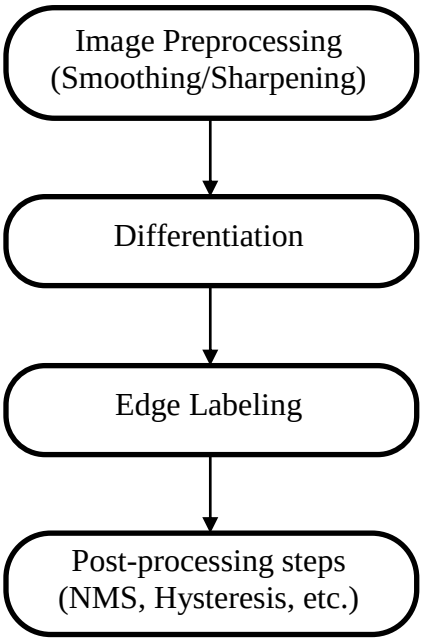


Figure 1.4: Edge Detection Notable Steps

This process reduces noise as much as possible while maintaining the essential edges. Image sharpening may also be used to pre-process the images. In this step, actual edges are made sharper to make them more distinct than false edges. Then, the first/second derivative of the image is calculated using image differentiation. Localization means localizing edges accurately. This step is used to improve the signal-to-noise ratio (SNR) of the edge-detected image by eliminating the number of false edges. This step is usually calculated to find the true edges, which are also differentiated from the pixels having same response because of noise (Yu & Chang, 2006). In this step, the locations of the edge points are estimated with sub-pixel resolution. The orientation of the edges is also evaluated. Some post-processing steps like Non maximum suppression(NMS), hysteresis, and double thresholding may also be used to enhance the edge detecting procedure even further. Each of the edge detection steps is detailed in the following sub-sections.

1.5.1 Image Differentiation

An image's first and second-order derivatives typically need to be computed to calculate edge points. For example, Step edges are detected by looking at the maxima in 1st-order derivative or zero crossings in 2nd order derivative of the image.

First Derivative: Consider $f(x, y)$ is a function that denotes the image's intensity values. Then 1st order derivative of *f* can be defined using two partial derivatives of *f* along the main axes, *x* and *y*.

$$g_x = \frac{\partial f}{\partial x}\ g_y = \frac{\partial f}{\partial y} \quad \text{.....(1.1)}$$

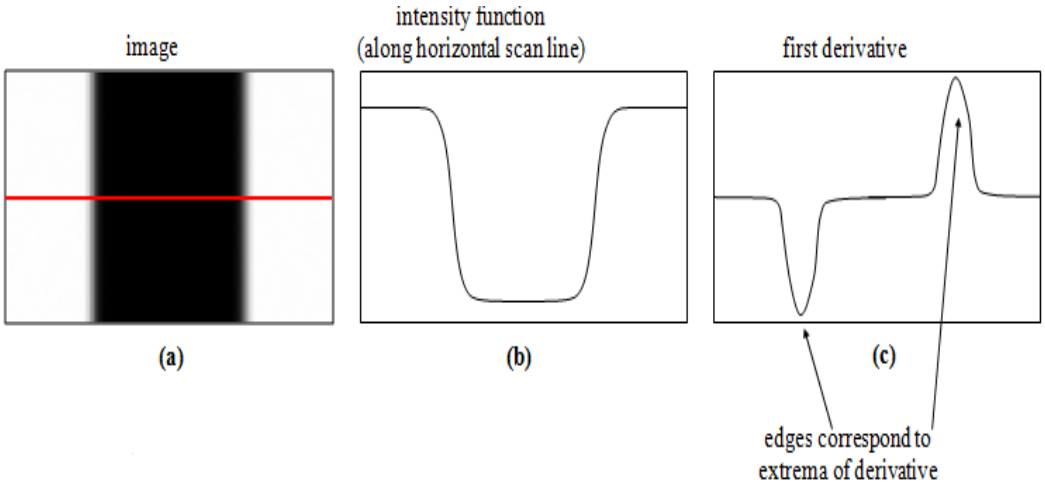
g_x and g_y represents the partial derivatives in both the *x* and *y* directions of image *f*. The gradient of *f* is a vector of magnitude and direction as defined in Eq. (1.2) and Eq. (1.3).

$$|\nabla g| = \sqrt{g_x^2 \ + \ g_y^2} \quad \text{.....(1.2)}$$

$$\theta = tan^{-1}\frac{g_y}{g_x} \quad \text{..... (1.3)}$$

The edge locations are found by observing the maxima in the gradient. The gradient always has a perpendicular direction to the edge.

The following Figure 1.5 (a) depicts a image's horizontal scan line, the intensity profile of the horizontal line is depicted in Figure 1.5 (b), and the corresponding 1st derivative is depicted in Figure 1.5 (c). It can be easily seen that edges correspond to extreme points of 1st derivative.



Source: http://www.cs.toronto.edu/~fidler/slides/2015/CSC420/lecture3.pdf

Figure 1.5: a) A horizontal scan line of an image (b) intensity function of scan line (c) first derivative

Second Derivative: The 2nd order derivative of an image *f* may be calculated in two ways: the first is by calculating the 2nd derivative along with the gradient direction, and the other is by using the Laplacian operator. The 2nd derivative of *f* along the gradient direction is related to the derivatives of *f* with respect to the main axes *x* and *y* as follows:

$$g_{xx} = \frac{\partial^2 f}{\partial x^2} \quad g_{yy} = \frac{\partial^2 f}{\partial y^2} \quad g_{xy} = \frac{\partial^2 f}{\partial x \partial y} \qquad \text{.....(1.4)}$$

The Laplacian of g, defined in Eq. (1.5) , is the most commonly used operator to approximate the 2nd order derivative along the gradient direction. It is a good approximation of the 2nd order derivative, provided that the line curvature is of constant intensity which crosses the pixel under examination is small. Laplacian is

very less efficient in locating high curvature points such as junction edges.

$$\nabla^2 g = g_{xx} + g_{yy} \quad \text{.....(1.5)}$$

The Laplacian operator has several advantages in relation to calculating the 2nd derivative along the gradient direction.

1. This operator is quite easy to use.
2. The Laplacian operator is linear, while the 2nd derivative is non-linear.
3. This operator is directional independent. It eliminates the need to calculate the operator's most appropriate direction.

Discrete Differentiation of 1st Derivative: Digital images are quantified versions of a real scene and stored as an array. So there is a need to determine the discrete approximation of the differential operator. To approximate the first-order derivative, the simplest way is to calculate it through two main dimensions, as given in Eq. (1.6).

$$g_x(x,y) = f(x,y) - f(x+1,y),\ g_y(x,y) = f(x,y) - f(x,y+1) \quad \text{.....(1.6)}$$

where $g_x(x,y)$ and $g_y(x,y)$ is an approximation of the gradient along the x direction and y direction of a pixel $f_x(x,y)$. These calculations can be done using a mask given as follows:

$$H_x = [1 \quad -1]\ H_y = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

The above-given mask has a problem of being not symmetric with respect to the point under consideration $f(x,y)$ that further introduces a partiality according to the position. One approach of solving this issue is to include the odd number of elements in the mask as explained by following Eq. (1.7):

$$g_x(x,y) = f(x+1,y) - f(x-1,y), g_y(x,y) = f(x,y+1) - f(x,y-1) \quad \text{.....(1.7)}$$

The mask which can be used to solve Eq. (1.7) of g_x and g_y can be given as following:

$$H_x = [-1 \quad 0 \quad 1] \; H_y = \begin{bmatrix} +1 \\ 0 \\ -1 \end{bmatrix}$$

There exist many masks to approximate first-order derivative along both directions, x and y, some of which are most properly-known are Prewitt, Sobel, Robert and Robinson.

Discrete Differentiation of 2nd Derivative

The discrete approximation of the second-order derivative may be defined as follows:

$$g_{xx}(x,y) = g_x(x-1,y) - g_x(x,y), g_{yy}(x,y) = g_y(x,y-1) - g_y(x,y). \quad ...(1.8)$$

Substituting the values of g_c and g_r from Eq. (1.6), we get the following:

$$g_{xx}(x,y) = f_x(x-1,y) - 2f_x(x,y) + f_x(x+1,y) \quad .....(1.9)$$

$$g_{yy}(x,y) = f_y(x,y-1) - 2f_y(x,y) + f_x(x,y+1)$$

The values of Eq. (1.9) can be represented using the following operator:

$$H_{xx} = \begin{bmatrix} 0 & 0 & 0 \\ +1 & -2 & +1 \\ 0 & 0 & 0 \end{bmatrix}$$

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

Using the mask given above, the Laplacian operator is calculated as follows:

$$H_{xx} + H_{yy} = \begin{bmatrix} 0 & 0 & 0 \\ +1 & -2 & +1 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 1 & 0 \\ 0 & -2 & 0 \\ 0 & 1 & 0 \end{bmatrix} \quad H = \begin{bmatrix} 0 & +1 & 0 \\ +1 & -4 & +1 \\ 0 & +1 & 0 \end{bmatrix}$$

Calculating Gradients using Convolution

To calculate the gradients, a product is performed between the corresponding mask and a small window of image equal to the size of the mask.

$$g_a(x,y) = \sum_i \sum_j f(x+i, y+j).(H_\alpha)_{i,j} \quad ..... (1.9)$$

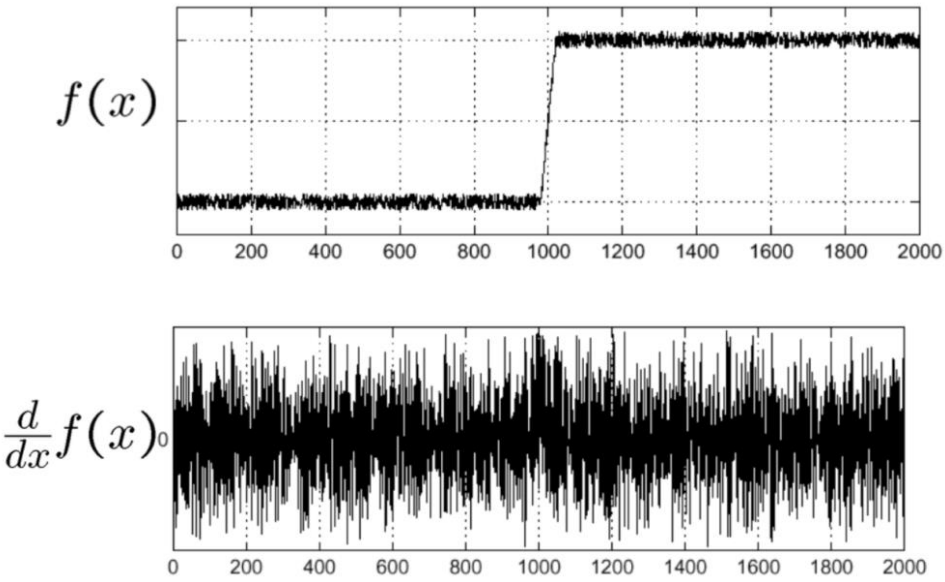
In mathematics, a convolution is an operation on two signals that produces a third signal, which is treated as a different version of one of the original signals. Suppose there are two signals $f\ and\ g$, the convolution written as *f*g* is defined as:

$$f * g = \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} f(i,j)g(x-i, y-j) \text{.....(1.10)}$$

Convolution works as a filter that produces several effects using different convolution masks. It calculates a new intensity value for the centre pixel by calculating the weighted average value by the neighbourhood pixel values. The weights applied are given by the convolution mask/kernel, a 2-D array. Let's compare Eq. (1.10) and Eq. (1.11). It can be agreed that convolution operation can be used to estimate the discrete derivative of an image by applying an appropriate convolution mask. Then by looking at the local maximum, edges can be localized.

Significance of Convolution

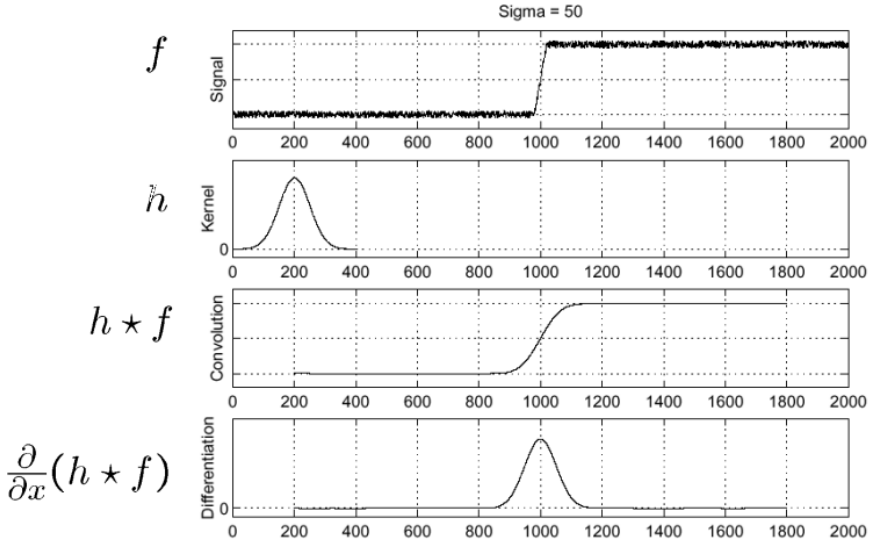
Since real-world images usually have noise, it is not always possible to detect step change by calculating the derivative of it and then locating the maximum; therefore, we need a different pre-processed version of the original image. It can be understood using the following Figure 1.6.



Source: https://ai.stanford.edu/~syyeung/cvweb/tutorial1.html

Figure 1.6: Noisy signal and its 1st Derivative

It can be seen from Figure 1.6, that calculating the first derivative is not enough to calculate the maxima point in noisy images. First, the original signal needs to be filtered out to reduce the noise to overcome this problem. It can be understood in Figure 1.7. The noisy signal is first convoluted by a Gaussian filter, and then the first derivative is calculated. The step change is now easily localized in this derivative output.

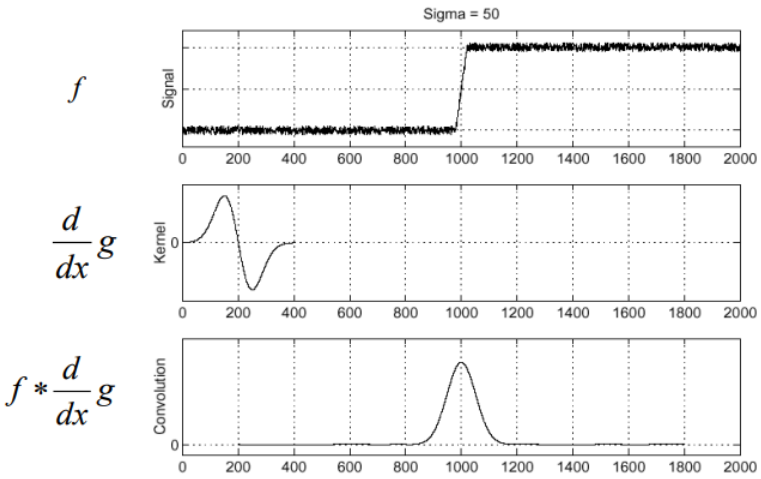


Source: https://pdfs.semanticscholar.org/0c61/2ec78f3dadf1b7bed2ce9ecadf536382ab2c.pdf

Figure 1.7: a) Original Signal *f* b) Convolution kernel *h* c) convolution output *h*f* d) derivative of convoluted output *h*f*

Since the convolution is a linear operator, we can use a pre-computed derivative of the Gaussian kernel to minimize the computation time.

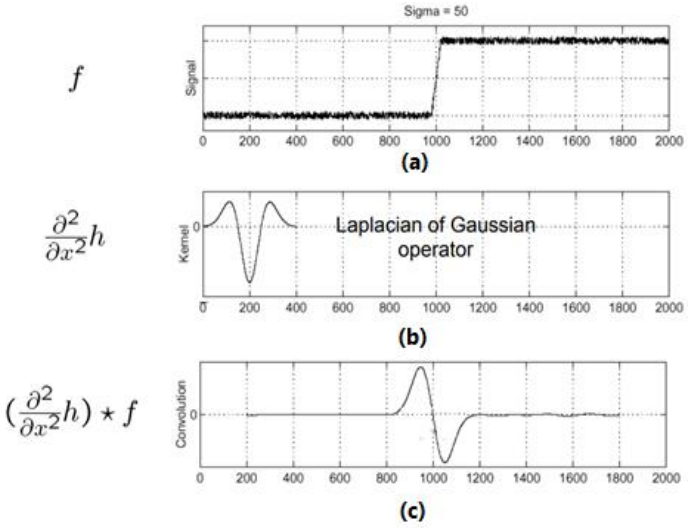
$$\frac{\partial}{\partial x}(h * f) = \frac{\partial}{\partial x} h * f \quad \text{.....(1.11)}$$



Source: https://pdfs.semanticscholar.org/0c61/2ec78f3dadf1b7bed2ce9ecadf536382ab2c.pdf

Figure 1.8: a) Noisy signal f b) the First derivative of Gaussian kernel c) convoluted output of f and $\frac{\partial}{\partial x}g$

Similarly, to locate the step edges, one can also use the 2nd derivative of the Gaussian kernel. In the following figure, if we look at the zero crossing of the modified version of the signal, we can easily find the step.



Source: https://medium.com/jun94-devpblog/cv-3-gradient-and-laplacian-filter-difference-of-gaussians-dog-7c22e4a9d6cc

Figure 1.9: a) Original image f b) Thesecond derivative of Gaussian c) The convoluted output of signal (a) and signal (b)

From this, we can conclude that convolution is important to realize the differentiation and calculate a modified version of the original noisy signal.

1.5.2 Edge Labeling

It is a process which involves locating the edges and increasing the SNR ratio by eliminating false edges. The procedure of localizing the edges is specific to the differentiation operator used. The thresholding method is generally used in gradient-based methods. Edges are detected by using a threshold value on gradient magnitude. These methods require further post-processing, like non-maximum suppression, to get uniform edges. While in zero-crossing methods, it is done by comparing the value of the 2^{nd} order operator of neighbourhood pixels. Instead of using a threshold, the value of the 2^{nd} order operator of the current pixel is calculated. This value is compared with the pixel left and below it. If all these three values are of different signs, it means zero-crossing. The results proved that using both vertical and horizontal directions improves localization, especially in localizing junction edges.

1.5.3 Non-maximum suppression

Non-maximum suppression is a post-processing technique generally used in threshold-based edge detection algorithms to improve performance. The image gradients result in a lot of thick edges. The ideal edges should be thin. An extra step called non-maxima suppression can be used as a thinning procedure to get thinned edges. NMS means to suppress, which is not maximum. In NMS, the image is looked along the direction of the gradient, and the gradient which is not local maximum is suppressed by setting the value equal to 0(black) and the local maximum set to 1(white) (Hosang, Benenson, & Schiele, 2017). For example: if the rounded angle of the gradient direction is 0^0, then the pixel of interest is decided as a pixel if the gradient value is larger than the pixels in south and north directions; similarly, if the rounded angle is 45^0, then the pixel is checked for the maximum gradient with the gradient value of pixels in the north-west and south-east directions, if the rounded angle is 90^0, then the pixel is checked for the maximum gradient with the gradient value of pixels west and east directions and finally, if the rounded angle is 135^0, then the pixel is checked for the maximum gradient with the gradient value of pixels in north-east and south-west directions This procedure is repeated from each pixel. In this way, the suppression procedure has thinned thick edges that are wider than a pixel.

1.5.4 Hysteresis Algorithm

False edges often occur because of noise in the image; however, this is not the only reason for false edges' occurrence. Despite having strong smoothing and differentiation methods, edge detectors may result in false edges. The significant cause is the use of a single fixed thresholding scheme. The search-based methods use a threshold to classify a particular pixel into the border or non-border pixels. The threshold is set as the minimum acceptable plausibility value (Oskoei & Hu, 2010). Due to continuous variation in the plausibility value of the image, the process of thresholding may result in false edges. In Hysteresis Algorithm, the problem of a single fixed threshold is consideredusing double thresholds T_1 and T_2. Any intensity gradient greater than T_1 is surely regarded as TRUE EDGE, and any intensity smaller than threshold T_2 is considered NOT EDGE. The intensity gradients between these two thresholds are classified as EDGE or NOT EDGE based on their connectivity with TRUE edges. In most cases, the value of two thresholds satisfies $T_2 \approx 1.5T_1$.

In zero-crossing methods, eliminating false edges is very difficult because a zero-crossing may occur because of a low gradient magnitude called a saddle point. The zero crossings may be falsely occurred because of noise or staircase edges. The false zero crossing because of low gradient magnitude can be removed using Hysteresis thresholding. The other false edges generated by certain models of edges called phantom edges, which have higher gradient magnitude than those of original true edges, can also be identified using their gradient sign.

1.5.5 Sub-pixel accuracy

To improve the localization accuracy sub-pixel approach is used. Each picture element comprises a sub-picture individual component with greater detail. As discussed, edges are localized using either maximum of local pixels in search-based methods or by zero-crossings; however, this may be further improved by applying interpolating neighbourhood pixels (Oskoei & Hu, 2010). This process is used the increase the visual resolution of an image. The following Figure 1.10, depicts the different steps of the sub-pixel edge detection method. In the first part of Figure 1.10, the intensity profile of a particular line of an image is shown by blue sample points. The first step of the edge

detection algorithm is to calculate the first-order derivative of this intensity profile. The second picture with red sample points illustrates the same. The maximum and minimum of this derivative function localize the edges. The second derivative is also calculated to determine these positions, shown as green sample points. The edges are then determined by zero-crossings of this second derivative. The function is interpolated to detect it more accurately, as seen in the fourth part of the figure. Finally, the blue arrow points to the correct position of the +ve and –ve edges.

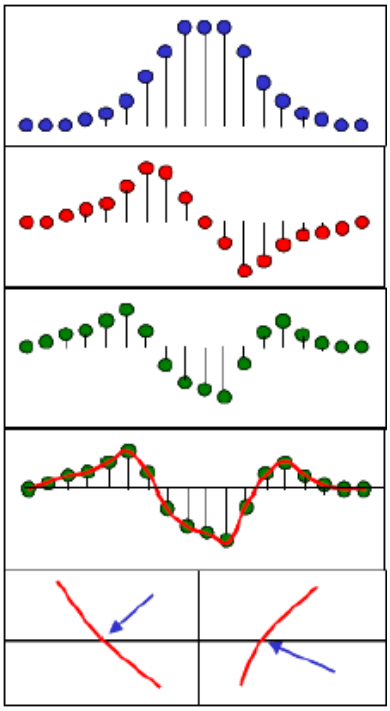


Figure 1.10: Interpolation on sub-pixel methods

1.5.6 Image Pre-processing

Image pre-processing is one of the main steps of edge detection. Before calculating edge points, images are pre-processed to remove unwanted noisy pixels and enhance the image's true edges. Different methods may be used to pre-process the image, e.g., image smoothing, sharpening, etc.

Various types of noises can corrupt an image. Substitute and additive noises are the two common types of noise. Substitutive noise (impulse noise) is a type of noise that occurs when something is substituted for something else. These types include salt and pepper noises, random valued impulse noise, etc. (Lendave, 2021) White Gaussian

noise, for example, is an example of additive noise. It has been discovered that noise in digital images is additive, has a Gaussian probability distribution, and has a power that is constant throughout the whole bandwidth. Additive white Gaussian noise refers to this type of noise. Different types of noise are explained here as under:

a. Gaussian Noise: The probability density function (pdf) of Gaussian noise is a normal distribution. A significant portion of an image sensor's "read noise" is present at a consistent level in dark portions of the image. Gaussian noise samples are independent, indicating that the noise's time correlation or spectral densities are both Gaussian. White noise, often known as Gaussian noise, defines the correlation of white Gaussian noise. Figure 1.11 shows an image that has been damaged by Gaussian noise.

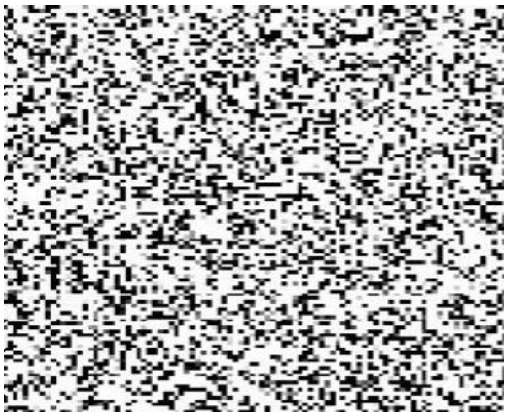


Figure 1.11: Image corrupted with Gaussian noise

b. Salt and Pepper Noise: In the images having salt-pepper noise, dark pixels appear in bright areas, and bright pixels appear in dark areas (Figure 1.12). It manifests itself as a jumble of white and black pixels. There are just two potential values in the Salt and Pepper noise model: "a" and "b." Each of these has a probability of less than 0.1. The intensity value for pepper noise is often found nearer to 0 in an 8-bit/pixel image, while the intensity value for salt noise is around 255 (Marr & Hildreth, 1980). Salt and pepper noise is commonly encountered in photographs. A median and morphological filters are used in an effective noise reduction technique for Salt and Pepper noise.

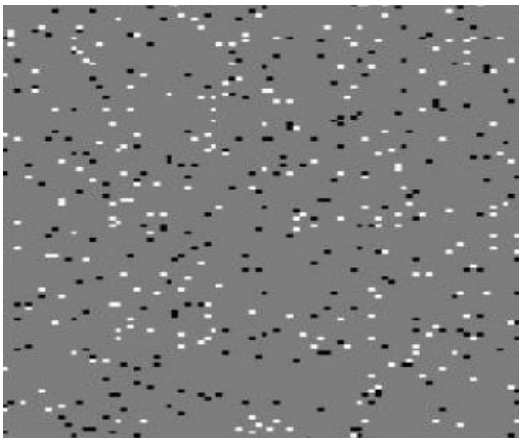


Figure 1.12: Image Corrupted with Salt and Pepper Noise

- **Image Smoothing**

As we have already discussed, real-world images are often corrupted by noise, and it is not easy to directly use the edge detection operator and find the edges. The noise should be reduced first, and then the edge detection operator should work efficiently. The main aim of image smoothing is to minimize noise. Noise in digital images may occur because of several reasons, such as the nature of the scene, image's acquisition system, sampling, and quantization. Sampling and quantization noise occurs as we limit the size and colour of a real-world scene. The noise induces several problems in edge detection, including false edges, missing real edges, localization errors, and broken edges. Image smoothing reduces the noise, but some edge information may also be lost. So there is always a trade-off between noise reduction and loss of important information. The goal is to make an optimal choice that makes the best balance between noise reduction and edge preservation. The main aim of smoothing is to regularize the numerical computation. The problems of edge detection are often ill-posed since the uniqueness, reliability, and stability of a solution cannot be ensured. The solution is not reliable and stable in case of high-frequency noises. Consider an original signal *f(x)*, and the signal is corrupted by small amplitude noise $\in . \sin(\omega t)$. The difference between *f(x)* and $f(x) + \in .\sin(\omega t)$ can be made fairly small if the value $\in$ is made small. But if we calculate the difference of their derivatives which may be quite large for large values of ω. It means the differentiation operation did not hold the stability principle of well-posed problems. However, by adding extra constraints,

an ill-posed problem can be converted into a well-posed one. This action is known as regularization. Regularization formalises the process of searching for an optimal filter that finds the necessary additional constraints. With the help of these extra constraints, the best compromise between noise reduction and edge preservation can also be made. Some of the notable smoothing techniques are discussed below:

a) Gaussian Smoothing: Gaussian smoothing causes a Gaussian function to blur an image. It is extensively used in graphics applications to minimize noise and enhance image details. Gaussian Smoothing is mathematically equivalent to convoluting an image with a Gaussian function, a low-pass filter frequently employed in image edge detection. The procedure is divided into two stages. A one-dimensional kernel is employed in the first phase to blur the image in only vertical or horizontal directions, while the remaining direction is blurred in the second phase using a one-dimensional kernel. The primary flaw in Gaussian smoothing is that point sampling the Gaussian function with very few samples results in a high in accuracy. So, in this scenario, precision is maintained by integrating the Gaussian function across each pixel's area dependent on computing cost (Wink & Roerdink, 2004).

b) Edge-Preserving Smoothing Techniques: Edge Preserving Smoothing techniques are used to smooth away textures while retaining sharp edges. Some examples of Edge-Preserving Smoothing Filters are:

1. Median filter (Brownrigg, 1984)
2. Symmetrical Nearest Neighbor Filter (SNN) (Harwood, Muralidhara, Hannu, & Larry, 1987)
3. Maximum Homogeneity Neighbor Filter (MHN) (Seo & HunJoo, 2015)
4. Conditional Averaging Filter (CAF) (Bauer, Jianchun, Steven, & Douglas, 1998)

Non-linear filtering algorithms are used in these filters. The edges, however, are not preserved by these four filters. In homogeneous environments, minor grey value variations are highlighted rather than eliminated.

Edge-preserving smoothing techniques are divided into two categories.

1. **Global optimization-based filtering:** optimized performance is measured based on data and a regularization term. It produces excellent quality with high computational costs.
2. **Local filters,** such as (Bilateral Filters/Guided Image Filter) are an extension in the gradient domain.

When comparing global optimization-based filters with local filters, the latter is simpler than the former, but it cannot preserve the sharp edge like in global optimization. The halo artefacts cannot be removed or avoided in the local filter, whereas in global optimization-based filter reduce the halo artefacts present in an image.

i) Bilateral Filter (BF): A BF is a non-linear filter employed in image edge preservation and noise reduction smoothing filters. The weighted average of the intensities values of the surrounding pixels replaces the intensity value for the pixel under examination. The weights are distributed according to a Gaussian distribution. Image artefacts are produced by a bilateral filter that is based on a local filter. Limitations in bilateral filters are:

1. Staircase effect: The image appears like a cartoon
2. Gradient reversal: It produces false edges in the image to remove these artefacts. Guided filters are an efficient alternative without these limitations (Michael, 2002).

ii) Weighted Guided Image Filter (WGIF): WGIF is a tool for removing halo artefacts from GIF or local filters. The WGIF, like the global optimization filters (Zhengguo, Jinghong, Zijian, Wei, & Shiqian, 2014), can preserve sharp edges due to the presence of weighting. It's a good filter that solves all problems with smoothing approaches, but its biggest flaw is that it's not as robust as WGIF. Training is necessary for the Adaptive Bilateral Filter (ABF) or Adaptive Guided Image Filter (AGIF); however, no such training is required for the Weighted Guided Image Filter (WGIF).

iii) Guided Bilateral Filter (GBF): GBF is a general, iterative filter that solves the shortcoming of bilateral filters while also inheriting their resilience qualities. Therefore if the noise level is higher in an image, then Guided Image filters which handle the situation more strongly than Joint/Cross Bilateral Filter but at the expense

of high peak signal-to-noise ratios (Caraffa, Tarel, & Charbonnie, 2015).

The bilateral filter causes artefacts in the image. The bilateral filter's main idea is to add a photometric weight to a conventional Gaussian filter. So this new approach is used in bilateral filters to make it perform well in preserving the edge without any artefacts. The main advantages of bilateral filters are:

1. The bilateral filter is used to estimate the intensity average in a neighbourhood with precision.
2. Using a grid or distributive histogram, the computational time improves in the bilateral filter.
3. The guided image is combined with the original image to produce a target-filtered image.
4. Adding photometric weight to an image leads to a joint/cross-bilateral filter.

a. Edge Sharpening

Human perception is delicate to an image's edges and minute features. Since high-frequency components make up pictures, their elimination or attenuation will lower the image's visual quality. Image sharpening is any enhancement technique that highlights a picture's edges and fine details. In various applications, images cannot be used straightway because it has varieties of intensities, varieties in brightness and may have uneven contrast. This irregular variety of intensity may be regarded as noise; therefore, image sharpening is needed to upgradethe image's visual quality. In doing this, high pass filtering plays a major role. Image sharpening is a technique to enhance the edges and other fine features of the image. It is achieved by the addition of the original images with a signal equivalent to a high-pass filtered version of the original image. The high-pass filtered version of any image can be achieved by subtracting its smoothed version from the original image (Archana & Aishwarya, 2016). Any smoothing operator can be applied to achieve the smoothed image. Then, the original image is combined with a scaled version of the high pass filtered original image to create a sharpened image.Using the Eq. (1.12), unsharp masking generates image detail from an input image.

$$g(x,y) = f(x,y) - f_{smooth}(x,y) \quad \text{.....(1.12)}$$

where $g(x,y)$ represents image detail, $f(x,y)$ represents original input image and $f_{smooth}(x,y)$ represents the smoothed version of the image f. The smoothed image version results from applying any image smoothing filter, such as the mean filter, on the input image.

After getting $g(x,y)$, the sharpened version of the original image can be expressed as:

$$f_{sharp}(x,y) = f(x,y) + \Upsilon * g(x,y) \quad \text{.....(1.13)}$$

Where 'γ' is the scaling factor whose values generally vary between 0.2 to 0.7.The figure demonstrating the process of image sharpening is shown in Figure 1.13 and Figure 1.14.

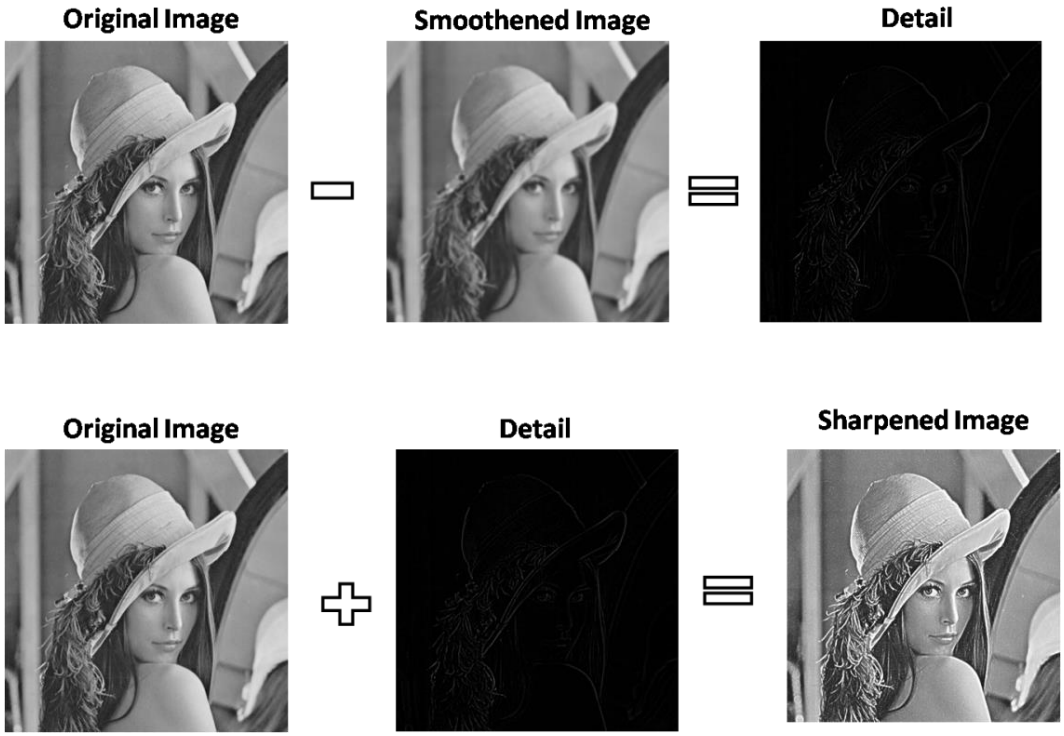


Figure 1.13:Demonstration of image sharpening (Lena)*h'

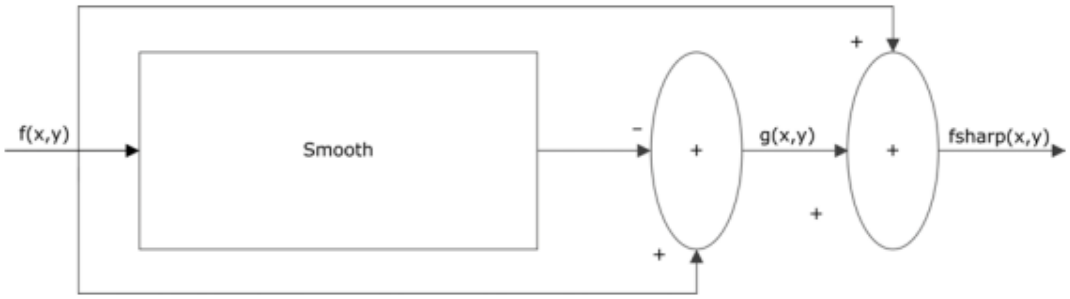


Figure 1.14: Image Sharpening process block diagram

1.6 Canny Edge Detection

The Canny's edge detection operatorgiven by John F. Cannyin 1986 is one of the most popular and widely used operators in the field of edge detection (Canny, 1986). It works as a multi-stage process and detects different type of edges from the image. Canny edge detection comprises five steps (Ziqi, Xiaoqiang, Meijiao, & Xiaobing, 2021):

1. Noise suppression
2. Calculation of gradients
3. Non-maximum suppression (NMS)
4. Double Thresholding
5. Edge tracking using Hysteresis

1. Use a Gaussian filter to reduce noise and other redundant information.

   $f(i,j) = H(i,j) * f(i,j)$

   Where $H(i,j) = \frac{1}{\sqrt{2\pi\sigma^2}}\exp(-\frac{i^2+j^2}{2\sigma^2})$ and * is the convolution operator.

2. Calculate the gradient of $f(i,j)$by applying any of the gradient operator to get:

   $M(i,j) = \sqrt{f_i^2(i,j) + f_j^2(i,j)}$ and $\theta = \tan^{-1}\frac{f_j(i,j)}{f_i(i,j)}$

3. Threshold M:

$$M_T(i,j) = \begin{cases} M(i,j) & \text{if } M(i,j) > \gamma \\ 0 & \text{otherwise} \end{cases}$$

   Where γ is a threshold chosen carefully so that all edges are preserved while the

major portion of the noise is suppressed.

4. Use non-maximal suppression to $M_T(i,j)$ to thin the edge ridges.

5. Obtain two binary images, I_1 and I_2, using two unique thresholds, τ_1 and τ_2 ($\tau_1 < \tau_2$). It is evident that the resulting image I_2 using higher threshold value, has less noise and erroneous edges compared to I_1, but it also has larger gaps between edge segments.

6. If and only if at least one strong pixel is present in the vicinity of the pixel being analyzed, the hysteresis turns weak pixels into strong ones.In I_2, connect edge segments to produce continuous edges.

1.7 Thesis Motivation

The pursuit of efficient edge detection techniques is integral to various fields, including image processing, computer vision, and pattern recognition. Traditional methods often struggle with balancing accuracy and computational efficiency. Swarm intelligence, inspired by the collective behavior of social insects, presents a promising avenue for addressing this challenge. By leveraging the principles of self-organization and decentralized control, swarm algorithms offer innovative solutions to complex optimization problems. Integrating swarm intelligence with guided image filtering techniques can enhance edge detection by efficiently extracting meaningful features while preserving important details. This thesis aims to explore the synergy between swarm intelligence and guided image filtering to advance the state-of-the-art in edge detection, fostering applications in medical imaging, autonomous navigation, and beyond.

Key Points:

- Traditional edge detection methods face challenges in balancing accuracy and computational efficiency.
- Swarm intelligence, inspired by social insects' collective behavior, offers novel solutions through self-organization and decentralized control.

- Integrating swarm intelligence with guided image filtering techniques can enhance edge detection by efficiently extracting meaningful features while preserving important details.
- The proposed research seeks to advance the state-of-the-art in edge detection, with potential applications in medical imaging, autonomous navigation, and more.

1.8 Novelty of the Proposed Work

1. **Integration of Swarm Intelligence, Fuzzy Logic and Guided Image Filtering:** This research pioneers the fusion of swarm intelligence algorithms, fuzzy logic with guided image filtering techniques for edge detection. By combining these two disparate methodologies, the proposed work explores novel avenues for enhancing the efficiency and accuracy of edge detection processes.
2. **Synergistic Optimization:** The proposed approach harnesses the collective intelligence of swarm algorithms to optimize guided image filtering parameters adaptively. Unlike traditional methods that rely on fixed parameter settings, this adaptive optimization scheme ensures dynamic adjustment based on the characteristics of the input image, leading to superior edge detection performance.
3. **Robustness Across Diverse Domains:** The versatility of the proposed framework enables robust edge detection across diverse domains, including medical imaging, remote sensing, and industrial inspection. By leveraging swarm intelligence's inherent adaptability, the system can effectively handle varying image characteristics and environmental conditions, thereby extending its applicability to real-world scenarios.
4. **Efficient Computational Framework:** Through judicious algorithmic design and parallel processing techniques, the proposed framework achieves computational efficiency without compromising on edge detection quality. This aspect is crucial for real-time applications and resource-constrained

environments, where rapid processing and minimal computational overhead are paramount.

5. **Potential for Autonomous Systems:** By leveraging swarm intelligence principles, the proposed work lays the foundation for edge detection systems suitable for autonomous systems, such as drones, robotics, and self-driving vehicles. The ability to accurately perceive and interpret edges in real-time is indispensable for enabling autonomous entities to navigate and interact with their surroundings effectively.

1.9 Thesis Layout

The thesis is organized into seven chapters as follows:

The Chapter 2 of this thesis presents the related work in the field of edge detection along with various methods' pros and cons. Performance measures are also discussed in this chapter. Problem formulation and thesis objectives are also discussed.

In Chapter 3 of this thesis, an edge detection scheme using ACO is presented, which uses a novel intensity mapping function to capture the intensity variations of the image. All the basic concepts employed in ACO-based edge detection have been explained. The chapteris concluded with results, and a comparative analysis is also presented with relevant techniques.

The fourth chapter of this thesis presents a fuzzy-logic-based algorithm, “Type-1 Fuzzy Logic and Guided Smoothing for Edge Detection’ which uses the type-1 fuzzy system and guided smoothing. This work furnishes a fuzzy logic-based edge detection approach that uses a sharpening-guided filter to manage edge quality and a Gaussian filter to limit noise caused by sharpening. The methodology is described, then the results are presented, along with a comparison to other available approaches in the literature. A set of statistical indicators is used to assess the method's accuracy.

In chapter 5, an edge detection method, “Edge Detection in Digital Images Using Guided L_0 Smoothen Filter and Fuzzy Logic” is introduced. The guided L_0 smoothen filter is used to control the degree of smoothness. Simulation is performed on BSD

and USC-SIPI image databases, considering more than 100 images. Some of the simulation results are presented in this chapter and compared with classical and modern methods using different performance metrics.

The chapter 6, discusses edge detection while considering guided image filering and ACO. Here, two sets of results are discussed; in the first work, ACO is combined with guided image filtering, while in the second method, guided image filtering along with modified ACO is considered. Simulation results are presented on various images, statistical results are evaluated, and a comparison with the latest technique is also made.

The work is summarized in the chapter 7. The thesis concludes with the contributions of the work done in this thesis and the discussion of their future scopes in this chapter.

Chapter Two

Literature Review

2.1 Introduction

In the pursuit of solving a scientific problem, examining past literature plays a crucial role in guiding researchers toward effective and informed solutions. A thorough literature survey not only provides insights into existing methodologies and findings but also helps in identifying gaps, trends, and the evolution of scientific understanding. This foundational step is essential for setting a well-defined direction for new research. Given the vast volume of available literature, however, it becomes imperative to establish clear criteria for selecting relevant papers to ensure that the review process is both efficient and effective.

The criteria for paper selection typically involve evaluating the relevance, quality, and impact of the research, as well as its alignment with the current problem being addressed. Researchers often prioritize studies that offer innovative approaches, significant findings, or comprehensive reviews relevant to their specific area of interest. This selective approach helps in distilling the most pertinent information from a sea of publications, thereby enhancing the quality and focus of the research effort.

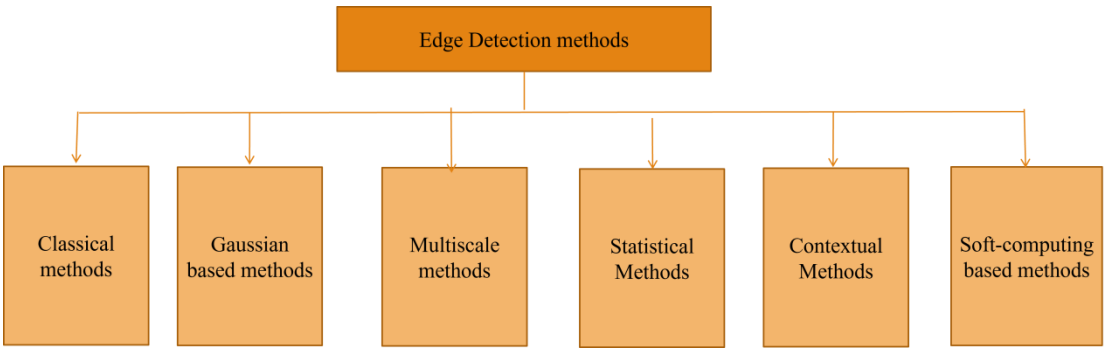


Figure 2.1: Block diagram for Paper Selection Criterion

Figure 2.1 illustrates a visual representation of the edge detection methods that were chosen for detailed examination in this study. The figure categorizes and organizes the various methods based on their principles and applications, providing a clear

overview of the different approaches considered. This block diagram serves as a reference point for understanding the criteria used in selecting edge detection techniques from the extensive body of literature. By systematically analyzing these methods, the research can build upon existing knowledge while contributing new insights or improvements to the field.

In this section, we have examined dominant work in the field of edge detection.Edge detection removes irrelevant data and interestingly protects the essential properties and features of an image. We have classified edge detection methods into seven categories based on the techniques used. The first category is of classical methods: in which traditional methods like Laplacian, Roberts, Sobel and Prewitt (Marr & Hildreth, 1980) (Sobel & Feldman, 1973) (Prewitt, 1970) (Fram & Deutsch, 1975) (Roberts, 1963) (Gupta & Mazumdar, 2013) are discussed. In these traditional methods, edge detection is done by using masks. These masks are convoluted with the image; after that, some method is applied to localize the edges. The edge detection methods using classical and soft-computing methods are described in more detail, while work under other categories is just reviewed with a description of the main contributions. The list of different categories of edge detection methods is detailed in Table 2.1.

Table 2.1: Categories of Edge Detection Techniques

Edge Detection Techniques	Methods
Gradient-based methods	Sobel, Roberts, Prewitt, etc.
Second–order derivative/ Zero crossing	Laplacian of Gaussian(LOG) etc.
Soft Computing	SVM, PSO, ANN, ACO, GA etc.
Deep Neural Networks	Sketch Token, Holistically-Nested, etc.

2.2 Classical Methods

Classical methods do not use smoothing filters but use discrete differential operators. Sobel, Prewitt, Kirsch, Robinson, and Frei-Chen (Lakshmi & Sankaranarayanan, 2010) all contributed to the development of some of these underlying algorithms. All of these methodsare based on a gradient estimation for the pixels. These techniques

are simple and precise but fail in the presence of edge orientation and noise. An edge pixel is the one for whichthe estimated gradient pixel value exceeds a threshold in these techniques. Because the threshold value is generated empirically now and then, we're likely to lose some true edges. While lowering the threshold values enhances the number of detected edges. As a result, there are a lot of spurious edges. Furthermore, the boundaries that have been observed are thick. In Table 2.2, orthogonal differential edge masks are shown as proposed by various authors in the past.

Table 2.2: The orthogonal differential edge masks

Methods	Masks	
Pixel difference	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & -1 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & -1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$
Separated Pixel difference	$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & -1 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$
Roberts	$\begin{bmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$
Prewitt	$\frac{1}{3}\begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}$	$\frac{1}{3}\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$
Sobel	$\frac{1}{4}\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$	$\frac{1}{4}\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$

2.2.1 Kirsch Mask

Although calculating the derivative in two directions is enough to approximate the gradient of the image, some researchers have given methods to calculate the gradient in more than two directions to efficiently suppress the noise (Maini & Aggarwal, 2009). In those cases, the gradient would be approximated by the maximum gradient magnitude of different directions. One of the most famous operators under this

category is the Kirsch operator (Sridhar, 2016), which has the following different masks according to different directions:

$$H_E = \frac{1}{15}\begin{bmatrix} -3 & -3 & 5 \\ -3 & 0 & 5 \\ -3 & -3 & 5 \end{bmatrix} \leftarrow \quad H_{NE} = \frac{1}{15}\begin{bmatrix} -3 & -3 & 5 \\ -3 & 0 & 5 \\ -3 & -3 & 5 \end{bmatrix} \nearrow$$

$$H_N = \frac{1}{15}\begin{bmatrix} 5 & 5 & 5 \\ -3 & 0 & -3 \\ -3 & -3 & -3 \end{bmatrix} \uparrow \quad H_{NW} = \frac{1}{15}\begin{bmatrix} 5 & 5 & -3 \\ 5 & 0 & -3 \\ -3 & -3 & -3 \end{bmatrix} \nwarrow$$

The masks and the directions of derivatives best approximated by the mask are given above. Each of these masks is generated by rotating the other by an angle of 45^o around the central element. The maximum angular rotation achieved by a 3x3 mask is 45^o, which means that with the help of 3x3 masks, we can get edges in four different directions. To distinguish between more directions, larger masks may be needed.

2.2.2 Robinson Mask

This mask is a different type of derivative mask which is used for edge detection. This mask can be called a direction mask as the mask is rotated according to 8-major directions. The mask is not fixed; rather, any mask can be used and turned to find edges according to the direction on the bases of zero columns. These masks are downscale versions of Kirsch masks (Sridhar, 2016).

$$H_N = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad H_S = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \quad H_W = \begin{bmatrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ -2 & -1 & 0 \end{bmatrix} H_E = \begin{bmatrix} 2 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & -2 \end{bmatrix}$$

$$H_{NE} = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ -1 & 0 & 1 \end{bmatrix} H_{NW} = \begin{bmatrix} 0 & -1 & -2 \\ 1 & 0 & -1 \\ 2 & 1 & 0 \end{bmatrix} H_{SE} = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} H_{SW} = \begin{bmatrix} -2 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 2 \end{bmatrix}$$

2.2.3 Frei-Chen Mask

Frei-Chen is a unique type of mask used for edge detection. This operator also uses masks of size 3x3, but it has a total of 9 convolution masks. These masks contain the basis vector,which implies that a sub-image of size 3*3 can be represented using the weighted sum of all nine masks (Rakos, 2011). The masks used in this operator are given as under:

$$H_1 = \frac{1}{2\sqrt{2}}\begin{bmatrix} 1 & \sqrt{2} & 1 \\ 0 & 0 & 0 \\ -1 & -\sqrt{2} & -1 \end{bmatrix} H_2 = \frac{1}{2\sqrt{2}}\begin{bmatrix} 1 & 0 & -1 \\ \sqrt{2} & 0 & -\sqrt{2} \\ 1 & 0 & -1 \end{bmatrix} H_3 = \frac{1}{2\sqrt{2}}\begin{bmatrix} 0 & -1 & \sqrt{2} \\ 1 & 0 & -1 \\ -\sqrt{2} & 1 & 0 \end{bmatrix}$$

$$H_4 = \frac{1}{2\sqrt{2}}\begin{bmatrix} \sqrt{2} & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & -\sqrt{2} \end{bmatrix} H_5 = \frac{1}{2}\begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} H_6 = \frac{1}{2}\begin{bmatrix} -1 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & -1 \end{bmatrix}$$

$$H_7 = \frac{1}{6}\begin{bmatrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{bmatrix} H_8 = \frac{1}{6}\begin{bmatrix} -2 & 1 & -2 \\ 1 & 4 & 1 \\ -2 & 1 & -2 \end{bmatrix} H_9 = \frac{1}{3}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

2.2.4 Laplacian Operator

The Laplacian operator, which uses the second derivative to locate the edge, has become increasingly popular. Laplacian masks are described as: (Sridhar, 2016)

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

2.2.5 Notable Papers for Classical Methods Variants

An approach suggested by (Xin, Chen, & Hu, 2012) to operate on colour images that are an enhanced version of the canny algorithm. The proposed technique employs a quaternion weighted average filter (QWAF) to improve the edge detection mechanism compared to traditional mask-based methods. This method uses QWAF on 9×9 windows, which slide over the complete image. The Sobel operator is used to evaluate the image's gradient in this method. The sliding window's size greatly influences the algorithm's overall performance. As a result, the edges become more blurred and thicker. This approach minimises the outline of broken and false edges but is computationally complex.

A method similar to the Sobel method is presented by (Gupta & Mazumdar, 2013). However, the size of the kernel mask has increased to 5×5. This method produces better results as compared to 3×3 masks. However, because the gradient approximation it provides is imprecise and the resultant image contains false, broken, and thick edges.

A comparison of common edge detection algorithms is presented by (Katiyar & Arun, 2012). They showed that the performance of the canny edge detection algorithm outperformed earlier approaches, with a decreased number of incorrect edges detected. On the other hand, the problem of false and broken edges is dealt with using Non-maxima suppression and hysteresis.

An edge detection approach that calculates the edge's intheimage using the concept of Center of Mass with Sobel Operator (COM-SOBEL) was proposed by (Jena, 2015). They also showed that their proposed method works better than the classical Sobel operator. The obtained results are better than Sobel's; still, accuracy is very limited.

A work published by the (Peter, Justice, & Isaac, 2016) in which they examined the image smoothing algorithms. This method fails in the presence of noise because it was based on pre-conceived assumptions and employed only a Gaussian function as its smoothing function. As Gaussian smoothing introduces blur in the image thus, correct edge detection can be difficult in the case of weak edges.

Feature extraction is a well-known problem in machine vision and image processing (Hacini, Akram, Herman, & Fella, 2017). To locate the edges, integer-order differential operators are typically used. A multi-directional operator is selected in this study to expand 1-dimensional digital fractional order Charef differentiator(1D-FCD) into the 2-dimensional version. 2D-fractional differentiation is a new edge detection method (2D-FCD). The generated multi-directional mask coefficients are used to recognise and retain image information.

Abrief description of different edge detection techniques, that are categorized according to the fundamental principles was presented by (Hagara & Kubinec, 2018) The database containing different type of images isused to detect edges. The latest developments in field of edge detection techniques are finally discussed. The images from Berkeley segmentation database (BSD) namely "Lena" and two additional photos are used to compare different edge detection techniques (BSDS500).

A novelkind of fuzzy masks that are useful for detecting edges in images was discussed by (Seng, Samad, & Nor., 2019). It is possible to identify edges in four

directions by using a series of 3-pixel masks that are generated. When the obtained findings are contrasted with conventional 3x3 fuzzy masks, it is discovered that the edge detection method of the suggested masks is superior. Results can be further improved by applying the thinning algorithms.

Existing edge recognition systems have problems with noise sensitivity, ineffective edge localization, and a limited ability to recognise edges automatically. Zheng et al. (Zheng, Zha, Yuan, Xuchen, Gao, & Zhang, 2020) address these issues. They recommended an algorithm to enhance the edge detection process. This research created an autonomous edge recognition method depending on grey prediction model to overcome these difficulities. In their work, they presented a mask to handle all 24 directions for improved and more precise edge identification.

A technique utiliizing fractional calculus was proposed by (Aboutabit, 2021). Fractional calculus allows for the development of the derivative of fractional orders; thus, various gradients are conceivable. Both noise-free and noisy images are used to test the suggested fractional order mask. The acquired findings confirmed the suggested edge detector's improved performance in comparison to both fractional and classical edge detectors.

2.3 Gaussian Based Methods

The most popular filters used in image processing are Gaussian filters. Marr and Hildreth (Marr & Hildreth, 1980) introduced an edge detector based on a Gaussian filter. They implied the smoothing filter to minimize the effect of noise. They proposed the 2-D Gaussian function as the smoothing filter. Gaussian-based methods have advantages compared to classical methods that minimise the impact of noise but have some disadvantages too. The Gaussian smoothing dislocates edges from their original positions and results in the number of false edges.

A related method applies the Difference of the Gaussian (DOG) operator on an image. Edges using DoG can be computed with the help of using two Gaussian operators having different values of σ for an image and developing the difference between the resulting two smoothed images. Zero-crossing detections are done in the DoG image

(Kovesi, 2010). However, the problem remain the same. Another problem with LoG and DoG is that the edges determined by zero crossings form numerous closed loops called the spaghetti effect, which is one of the major drawbacks of this method. The issue of missing edges is also present in the LOG-filtered images.

A method introduced by Canny was accepted worldwide as the finest edge detection algorithm. Canny took three important features needed for any good edge detector. These three features are good localization and detection and oneunique response to each edge.

1. Thresholds are calculated according to the noise densityof the image, so prior estimation of the noise is required.
2. Because to the Gaussian smoothing, edge pixels dislocate from their true positions.
3. Computational complexity is high.
4. The Gaussian smoothing blurs corners and junctions, and they become hard to detect.

This last problem is addressed by the SUSAN method (Rezai-Rad & Aghababaie, 2006), which connects edges better and results in nice junctions. The SUSAN method uses circular masks in contrast with a normal window kernel. The circular mask is placed at each pixel, and the brightness of each pixel is compared with the nucleus (centre point). The SUSAN method (Smith & Brady, 1997) works well for all types of edges except roof edges. The anti-noise ability of the SUSAN method is weak, and the SUSAN detector uses a fixed global threshold that is unsuitable for the general situation.

Some of the important publications under this category are described below:

A multi-scale edge detection algorithm that works on the basis of enhancing Gaussian Smoothing was proposed by Lopez-Molina et al. (Lopez-Molina, Baets, Bustince, Sanz, & Barrenechea, 2013). After increasing Gaussian smoothing, the Sobel technique was used on each of these images and then a new method was given to track edges from the coarser to the finer scale. The algorithm can improve the single-scale methods as far as noise avoidance and edge location is concerned. But the improvement we get at the cost of increased complexity and the addition of certain

additional parameters make it more costly.

An information fusion algorithm for a multi-scale multi-expert edge detection algorithm is proposed by (Ozkan & Sahin, 2015). In this method, different Gaussian smoothening functions are applied at various parts of the image.

A new technique to calculate edges is proposed by Kenan et al. (Kenan, Hui, Zhao, & Prehofer, 2016) using multi-scale edge fusion. Initially, they used DoG pyramid decomposition to get different multi-scale images. In the second step, edge maps are computed in each different-scale version, and then multiple edge maps are fused. They proposed a multi-scale edge fusion technique to achieve this. They also considered the edge displacement between the adjacent edge maps to calculate the final edge map. They took the scale of 1 result as a candidate edge map and then looked in all other edge maps for the pixel where edge displacement is not more than 1. This method has decreased both the computational complexity and time consumption of the process. The problem with this method is that although they had provided experimental values, the sensitivity of the parameters was not analyzed.

2.4 Non-Linear Methods

In the case of linear methods, images are convolved using a Gaussian filter for smoothing, which decreases noise but blurs the edges due to isotropic smoothing. To deal with this issue, an anisotropic diffusion(AD)-based scale space representation of an image was proposed (Perona & Malik, 1990). The main aim is to allow for space variation blurring. The idea is to keep boundaries crisp while smoothing within a zone. The desired effect can be achieved by setting the diffusion constant to a high value within the area and a very low value (perhaps 0) at the boundary. The image gradient is used to determine the diffusion coefficient, which might vary throughout the image plane; this results in adaptive smoothing of the image with good localization of edges.

2.5 Statistical Methods

A new method of edge detection based on the statistical model and data driven is

proposed by (Konishi, Yuille, Coughlan, & Zhu, 2003). This method highlights the need to create a model of the image background (the off-edges). They used non-parametric representations to demonstrate the conditional probability distributions on two data sets of images.

A statistical edge detection approach is proposed by (Santis & Sinisgalli, 1999). A sharp local intensity fluctuation of the grey-level mean value was used to model the presence of an edge. The statistical model parameters in each pixel were calculated using a Bayesian method. For the hypothesis, testing likelihood ratio statistics were then utilised to determine if a pixel was an edge point. This strategy has the benefit of using the estimated local signal characteristics while obviating the need for an overall thresholding procedure.

2.6 Contextual Methods

An edge detection method based on context analysis was proposed by (Yu & Chang, 2006) (Yu Y. C., 2006). To detect the edges, they suggested method to employ information from predictive error value generated by the gradient adjusted predictor (GAP). These methods are well used in data hiding techniques, and various interpolation-based methods are developed to estimate GAP.

2.7 Soft Computing Methods

Soft computing techniques are continuously gaining popularity because oftheircontrolling mechanisms in edge detection. In these methods, artificial neural networks (ANN), Fuzzy logic, deep neural network (DNN) and evolutionary algorithms (EA) based on swarm's behaviours are the most common.

2.7.1 ANN-based edge detection

In this method, ANN is made learnt to judge a pixel as an edge and non-edge pixel. The basic concept of ANN is shown in Figure 2.2, where a 3×3 mask with pixels values varied from g_1 to g_8 with the centre pixel is under investigation and marked as '×'. The pixel values are mapped to input neurons, and ANN is learnt to obtain output pixels as edge and non-edge pixels; for better classification, a hidden layer of

interconnected neurons is added. An activation function (*f*) is used for the classification (Zheng & He, 2004).

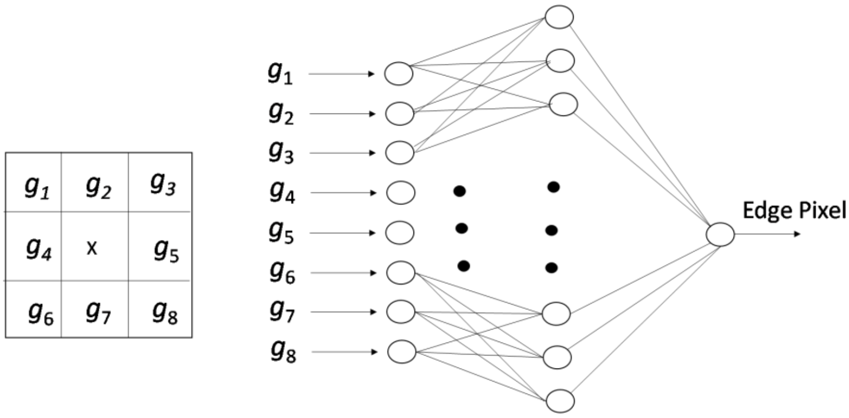


Figure 2.2: Schematic of the ANN structure for pixel prediction (edge/non-edge) (Zheng & He, 2004)

The calculated output pixel value after getting passed through the ANN system is defined by Eq. (2.1)

$$\bar{y}_j = f\left[\sum_i g_{ij} w_{ij} + b_{i,j}\right] \ldots \qquad (2.1)$$

Where 'w_i' are weights, and 'b' is fixed biased. Let the original pixel value is 'y', the mean-square error (MSE) is defined bythe following Eq. (2.2):

$$E = \frac{1}{2}(y_j - \bar{y}_j)^2 \ldots \qquad (2.2)$$

The error may bereduced by proper manipulating the weights. The weight modifications is proportional to the error as:

$$\Delta w_{ij} = -\frac{\partial E}{\partial w_{ij}} \ldots \qquad (2.3)$$

The updated weights can be written as follows:

$$w_{ij}(t+1) = w_{ij}(t) \pm \alpha \Delta w_{ij} \ldots \qquad (2.4)$$

'α' is a weight-controlling parameter.

An edge detection scheme based on the concept of a Back Propagation Neural

Network is proposed by Hamed Mehrara et al. which is very similar to the above-detailed method (Mehrara, Zahedinejad, & Pourmohammad, 2009).

2.7.2 Fuzzy Set Based Edge Detection

Fuzzy sets are an important part of the fuzzy theory, which can be applied in edge detection. Here, the intensity of the pixels is represented in terms of membership functions. The membership functions are derived for both inputs and outputs. For pixels neighbourhood, fuzzy rules are developed, and a fuzzy inference engine is used for the output prediction.

Table 2.3: Fuzzy based notable Techniques

Authors	**Techniques**
Alshennawyet al. (Alshennawy & Aly, 2009)	Fuzzy Logic
Kaur et al. (Kiranpreet, Mutenja, & Gill, 2010)	Fuzzy Logic
Aborisade (Aborisade, 2011)	Fuzzy Logic
Moslem et al. (Moslem & Maghooli, 2011)	Fuzzy Logic
Zhang et al. (Zhang, Xiao, Ma, & Song, 2009; Zhang, Xiao, Ma, & Song, 2009)	Adaptive Neuro-Fuzzy

The theory of fuzzy sets has also been used for the process of edge detection. Kim et al. (Kim, Lee, & Kweon, 2004) proposed an algorithm using a 3×3 kernel and a look-up table. Kaur et al. (Kiranpreet, Mutenja, & Gill, 2010) discussed a method based on fuzzy rules; here, 16 fuzzy rules were devised to characterize edge and non-edge. The results were obtained with good accuracy for images without any noise. But this method fails in the presence of noise. More experiments have been performed on higher types of fuzzy logic, particularly fuzzy type-2, to oblige more noteworthy vulnerabilities (Chen, Chang, & Pan, 2013) (Hsu & Juang, 2011).

Mathur et al. introduced the latest algorithm based on fuzzy relative pixel value (Mathur & Ahlawat, June-July 2008). It finds and features each one of the edges related to an image. In this methodology, the relative pixel values are examined and subsequently given a calculation to shorten the image processing by the application of Artificial intelligence. Table 2.3 presents some of the edge-detection techniques based

on Fuzzy logic.

2.7.3 Deep Learning Based Edge Detection

Recently, Convolutional Neural Network based methods have gained popularity in edge detection,and some of the notable techniques are Deep-Contour (Bertasius, Shi, & Torresani, 2015), Deep Edge (Xie & Tu, "Holistically-nested edge detection", 2017), and CSCNN (Wang, Zhao, & Huang, 2017). Holistically-nested methods have automatic learning capabilities based on deep learning phenomena. The other recent notable mechanisms in edge detection are deep convolutional neural networks (Liu, Cheng, Hu, Wang, Zhang, & Bai, 2017), Fuzzy cellular automata convolution neural networks (CNNs) (Hwang & Liu, 2015). Edge detection based on single-pixel imaging was proposed by (Zhang, Zhao, Breckon, & Chen, 2017). In recently published work, detailed artificial intelligence and CNN-based edge detection methods have shown that these methods fail in the presence of small perturbations (Farbod, Akbarizadeh, Kosarian, & Rangzan, 2018).

2.7.4 Evolutionary Method-Based Edge Detection

Edge detection is employed in segmentation registration and object identification since it is necessary for determining object boundaries in images. Several strategies for improving edge detection have been developed. To detect the edges, some techniques used evolutionary-based algorithms. The edge detection problem has lately been subjected to various evolutionary optimization strategies. Evolutionary algorithms examples can be found in detail in (Dorigo, Mauro, & Stutzle, 2006) (Caponetti, Abbattista, & Carapella, 1994) (Wang, Dapei, & Lei, 2018) (Joshi, Kulkarni, Kakandikar, & Nandedkar, 2017) (Tian, Weiyu, & Shengli, 2008) (Ari, Ghosh, & Mohanty, 2014) (Banharnsakun, 2019) (Yigitbasi & Baykan, 2013) (Verma, Agrawal, & Sharma, "An optimal edge detection using modified artificial bee colony algorithm", 2016) (Setayesh, Mengjie, & Johnston, 2009) (Chen, Ting, & Xiaosheng, 2012) (El-Khamy, Lotfy, & El-Yamany, 2000) (Gongalez, Castro, Patricia, & Oscar, 2015). Table 2.4 details some of these methods presented by various researchers.

Table 2.4: Swarm-based notable edge detection methods

Authors	Techniques
Dorigo et. al. (Dorigo, Mauro, & Stutzle, 2006)	**ACO**
Tian et. al. (Tian, Weiyu, & Shengli, 2008)	**ACO**
Samit et. al. (Ari, Ghosh, & Mohanty, 2014)	**ACO**
Banharnsakun (Banharnsakun, 2019)	**ABC**
Yigitbasi et al. (Yigitbasi & Baykan, 2013)	**ABC**
Verma et al. (Verma, Agrawal, & Sharma, "An optimal edge detection using modified artificial bee colony algorithm", 2016)	**ABC**
Setayesh et al. (Setayesh, Mengjie, & Johnston, 2009)	**PSO**
Chen et. al. (Chen, Ting, & Xiaosheng, 2012)	**PSO**
Gonzalez et. al. (Gongalez, Castro, Patricia, & Oscar, 2015)	**CSO**
Gonzalez et. al. (Gongalez, Melin, Castro, Mendoza, & Castillo, 2016)	**CSO**

In the edge detection technique using ACO, numerous artificial ants be randomly distributed and then make random moves on the image. This movement of ants is directed by the other ant'sactions and simultaneously builds a pheromone matrix. The probability of edge is denoted by the values of the pheromone matrix. Then the binary decision is made to classify the current pixel as an edge pixel or a non-edge pixel. This method is proposed by De-Sian (Lu & Chen, 2008).

A method to calculate edges by applying the Ant Colony Optimization(ACO) and the fuzzy derivative technique is proposed by Verma et al. (Verma, Hanmandlu, Kumar, & Srivastava, A novel approach for edge detection using Ant colony optimization and Fuzzy Derivative Technique, 2009). The authors claimed that the computational complexity of their algorithm is low. Also, the proposed method is immune to noise. The only problem with their approach is that it results in few edge pixels contrasted with other standard edge detection techniques. Also, the resulting edges are broken and thick.

A method using bacteria foraging to do edge detection is proposed by Verma et al. (Verma, Hanmandlu, Kumar, Chhabra, & Jindal, A novel bacterial foraging technique for edge detection, 2011). In their technique, they allowed the bacterias to traverse randomly across the image's pixel and then derivative are calculated in every direction. The direction probability matrix is constructed using computed derivatives. It is stated that the directions for which both pairs of derivatives are high, e.g. North-south

derivative, west-east derivative etc., must have a higher value in the direction probability matrix. They calculated the performance of their method using the Kappa and Shannon entropy function to calculate information content in the output edge maps. The results of the presented technique is superior than traditional kernel-based edge detectors, favouring this method a better option for edge detection. The algorithm has some problems of double and broken edges. They also suggested how the problem could be dealt with.

An edge detection method based on the law of universal gravity by using the concept that gravitational force attracts the body is devised by Sun et al. (Sun, Liu, Liu, Ji, & Li, 2007). In the same way, it is assumed that each pixel is a body with mass equal to its intensity value, which puts gravitational pull on its neighbourhood pixels and experiences force from their surroundings. The experience force larger than some pre-defined threshold is distinguished as an edge pixel.

There were some problems in the method stated above of Sun et al. (Sun, Liu, Liu, Ji, & Li, 2007). These problems were addressed by C. Lopez-Molina et al. (Lopez-Molina, Bustince, Fernandez, Couto, & De Baets, 2010) and they gave a modified method for edge detection. They solved the problem of zero-mass pixels by adding some small constant to make it non-zero. Also, they compensated for the effect of brightness dependence by adjusting the multiplicative parameter for each pixel according to the pixel intensity. The problem with this method is that several variables need to be tuned to get better results.

Another edge detection method using gravitiational search algorithm (GSA) is proposed by Om Prakash Verma et al. (Verma, Sharma, Kumar, & Agrawal, May 2013). The local differences in the pixel'sintensity value is utilized to calculate the edges. The technique reduces the amount of image's data significantly to be further used in other sophisticated image processing alogirthms, making it more time-efficient in comparison to other methods. Although, their methods have a problem resulting in fewer edge pixels compared to the other methods.

2.8 Notable Methods

In this section, some other state-of-the-art methods are discussed.

2.8.1 gPb Method

For perceptual organisation and shape recognition, contours and junctions are critical clues. However the picture intensity surface is confused in the vicinity of a junction, detecting junctions locally has proven to be difficult. Near intersections, edge detectors are similarly ineffective. This paper develops a proper strategy for junction detection, while considering the contours that occur at a junction; contours can be recognised by algorithms that employ more global approaches.

In addition to intensity, taking texture and colour gradients into account, more contemporary Pb boundary detectors greatly outperform the classical approaches. The ability of the Pb algorithm to suppress false positive edges compared to classical approaches in textured regions make it a better approach. In (Arbelaez, Maire, Fowlkes, & Malik, 2011), a new "global" Pb (*gPb*) approach, that improves boundary identification by reasoning about longer-range interactions between contours has been proposed. This method is still very close to becoming state-of-the-art in terms of performance. Results are shown in Figure 2.3.

Source: http://cs.brown.edu/courses/csci1430/2011/proj2/

Figure 2.3: (a) Input image (b) Canny edge detection (c) Pb-lite (4) *gPb*

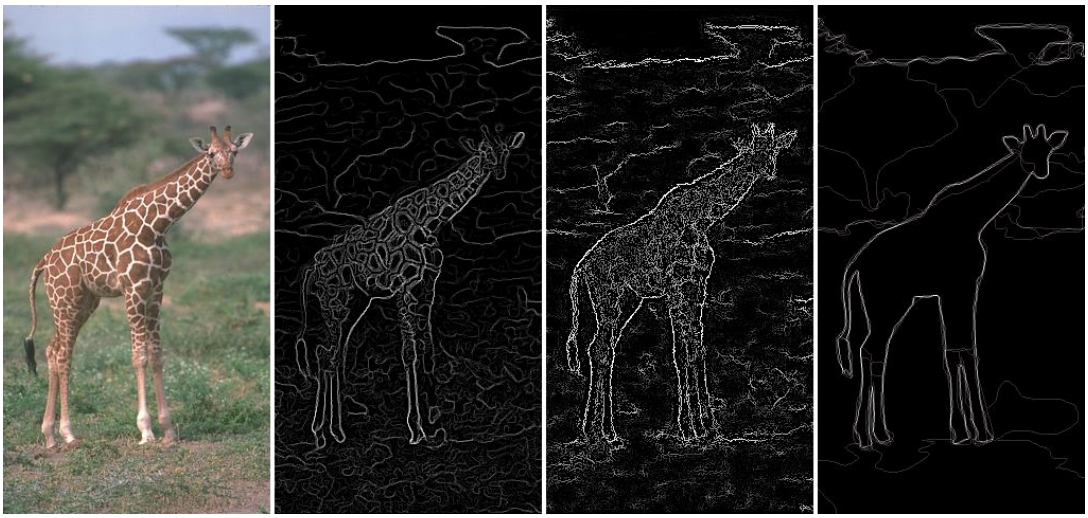
2.8.2 Sketch Tokens

The Sketch Tokens boundary detector was introduced by Lim etal. (Lim, Zitnick, & Dollar, 2013). In various aspects, the Sketch Tokens technique differs from the *gPb*

algorithm.

1. Sketch Tokens is a one-of-a-kind algorithm that works only locally. Without any global or long-range thinking, each border choice is based on local patch statistics.
2. Instead of Pb's very sophisticated, hand-designed texture half-disc descriptor, Sketch Tokens uses comparatively simple gradient and colour attributes.
3. To establish the mapping from image structure to boundary scores, Sketch Tokens depends extensively on machine learning, whereas Pb does not.

The Sketch Tokens algorithm outperforms the local Pb algorithm and is on par with the more complex "global" Pb algorithm. Results are shown in Figure 2.4.



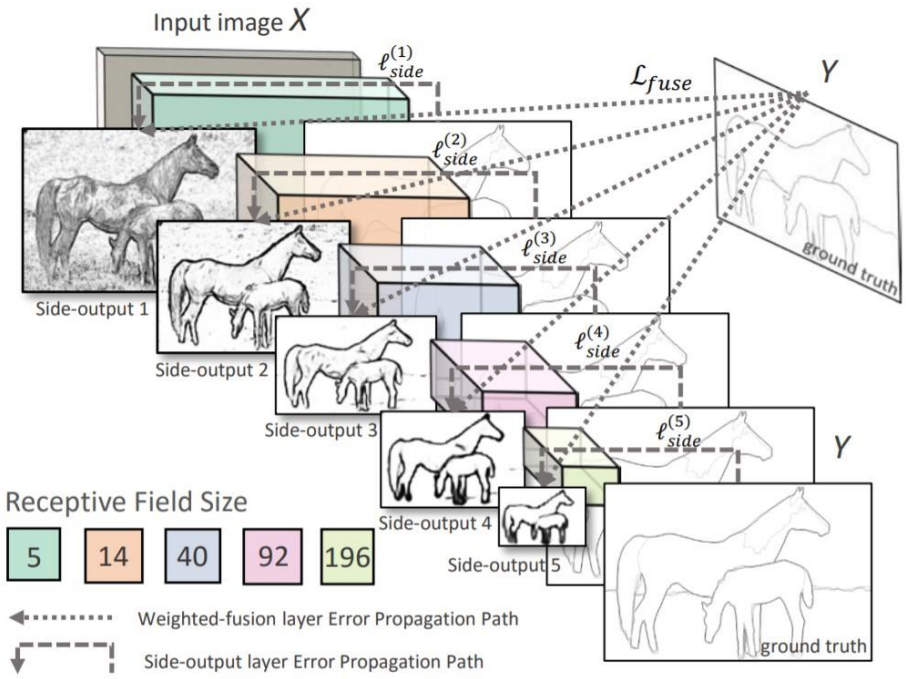
Source:http://cs.brown.edu/courses/cs143/2013/proj5/

Figure 2.4: (a) Input image (b) Canny edge detection (c) Sketch Token (d) Ground truth

2.8.3 Holistically-Nested Edge Detection (HED)

HED is one of the first CNN-based edge detection methods (Xie & Tu, "Holistically-nested edge detection", 2015). According to the authors, the model contains two distinguishing characteristics that give it its name. The model is 'holistic' in the sense that it accepts an image as input and outputs another image (edge map). The model does not require hand-crafted features as inputs; its architecture allows it to construct these features internally in the hidden layers. The fully-convolutional networks pass on this trait to the fully-convolutional networks.

Second, the hierarchical model uses deep supervised learning to learn at different scales. It is accomplished by altering the depths of side outputs (Xie & Tu, "Holistically-nested edge detection", 2017). The model is built on a VGGNet architecture that has been reduced from its previous pooling layer. Each of the five convolution blocks has a side output that aids in learning features at various scales. These side outputs are joined to form a fusion layer whose weights can also be learned. The fusion layer that results produces a unified output. Results are depicted in Figure 2.5.



Source: https://developers.arcgis.com/python/guide/edge-detection-with-arcgis-learn/

Figure 2.5: Illustration of HED architecture

The sum of the losses calculated at both the fusion layer and the side outputs is the total loss. The cross-entropy loss function is utilised in the model, along with a class-balancing weight for the side outputs.

2.8.4 Boosted Edge Learning (BEL)

BEL is a revolutionary supervised learning technique for object and edge boundary identification (Dollar, Tu, & Belongie, Supervised learning of edges and object boundaries, 2006). At each position in the image, an edge point decision is determined

independently using a probabilistic boosting tree classification algorithm. There are no settings to modify as in the learning-based system, which makes BEL a good technique for edge and object detection. Results for BEL are shown in Figure 2.6.

Source https://www2.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/bench/gray/BEL/157055.html

Figure 2.6: (a) Original Image (b) Ground Truth (c) Detected edges

2.9 Edge Detection Performance Measures

It is not easy to define the general-purpose evaluation for edge detection because of the following issues: Missing edge segments, falsely accepted edges and dislocated edges etc. Before going into detail about edge detection performance measures, we start with the basics. Consider the ground truth image as (I_{gt}) and the edge detected image as (I_{ed}). Let the quality measure is denoted by Q and property as (P); then the following properties must be satisfied:

- Symmetry (P_1):$Q(I_{gt}, I_{ed}) = Q(I_{ed}, I_{gt})$
- Ideal Solution (P_2): I_{gt}=I_{ed}
- Sensitivity to noise (P_3): if any pixel (p) does not belong to ground truth (I_{gt}) or edge detected image (I_{ed}), i.e., $p \notin (I_{gt} \cup I_{ed})$, then $Q(I_{gt}, I_{ed}) < Q(I_{ed} \cup \{p\}, I_{gt})$
- Sensitivity to improvement (P_4): if $p \in I_{gt}$ and $p \in I_{ed}$, then $Q(I_{gt}, I_{ed}) < Q(I_{ed} \cup \{p\}, I_{gt})$

The first property is self-explanatory, second property state that there is only one optimal solution. The third and fourth properties indicate that including correct and incorrect pixels decreases or increases error. However, these four properties are not good enough to describe edge detection,including the falsely excluded important edge. Sometimes, extra pixels are accepted. Considering $p \notin I_{gt}$ and having p in I_{ed} increase unavoidable errors. In edge detection, the main requirements are both the

correct identification and correct location of edge pixels.

We classify the positive pixels as edges when the classification is binary. We can classify pixels edge into four unique classes with the condition that ground truth is present. These four classes are True Positive (TP), False Positive (FP), True Negative (TN) and False Negative (FN). Since not a large portion of the pixels are edges, there is an issue of imbalanced binary classification (Chawla, Japkowicz, & Kołcz, 2004), where the dominating class is the negative class.

The issues related to the position of the pixel can be solved with the help of the spatial tolerance during the matching of the edge pixels as the factor, which decides whether the pixel classification is accurate or not, is the least alteration in the position of the pixel. Normally, an edge pixel cannot be assumedcorrect, relying upon the way it is found t or t+1 pixel from the true edge. According to Liuand Haralick, a quality measure is exact in the event of small changes in value reflecting small variations in the detection (Liu & Haralick, 2002). The measurements such as F-score and test are precise when pixel positions are exact but fall flat when pixel position changes. These issues prompted penalizing an edge pixel relying upon its separation to a true edge, encouraging the concept of distance-based EMs.

Correlation Coefficient

The correlation coefficient is obtained using the equation (Heel, 1987)

$$|\rho| = \frac{cov[I_{gt}, I_{ed}]}{\sqrt{Var[I_{gt}]Var[I_{ed}]}} \quad \text{.....(2.5)}$$

Under perfect matching $|\rho|$ is one, and its lowest value is zero. This metric satisfies properties P_1 and P_2.

Pratt's Figure of Merit (FoM)

Pratt's FoM evaluates edge location exactness in edge detected image compared to ground truth image by measuring the displacement of edge points detected from an ideal edge. The FoM is characterized by (Wesolkowski, Jernigan, & Dony, 2000).

$$FoM = \frac{1}{\max(I_{gt}, I_{ed})} \sum_{i=1}^{I_{ed}} \frac{1}{1+\mu d^2(p, I_{gt})} \quad \text{.....(2.6)}$$

Here,

I_{gt} = ideal edge points (ground truth)

I_{ed} = edge points detected

d = displacement of detected edges from ideal edges

μ = scaling constant.

It is essential to note that these measurements binarize information before assessing images; this implies that assessment is done over images that have lost data. The metrics mentioned above return a value from 0 to 1, where 0 means no similarity between the detected image and the reference image, and 1 implies high closeness was seen. In other words, each edge pixel detected in one image is also recognized at the same place in another.

Structure Similarity Image Metrics (SSIM)

SSIM completes a greatly improved activity at measuring subjective image quality compared to MSE or PSNR. At a high state, SSIM endeavours to estimate a picture's luminance, contrast, and structural adjustment. The SSIM is given by: (Hore & Ziou, 2010).

$$SSIM(I_{gt}, I_{ed}) = \frac{(2\mu_{gt}\mu_{ed}+k_1)(2\sigma_{gted}+k_2)}{(\mu_{gt}^2+\mu_{ed}^2+k_1)(\sigma_{gt}^2+\sigma_{ed}^2+k_2)} \quad \text{.....(2.7)}$$

Where μ is mean, σ is the cross-correlation parameter, σ^2 is variance, and the remaining parameters are fixed constants.

Hausdorff Distance (HoD)

Considering two images I_{gt} ={$a_1, \ldots, a_n$} and I_{ed}={$b_1, \ldots, b_n$}, the Hausdorff distance calculated as (Ma & Grimson, 2005):

$$H(I_{gt}, I_{ed}) = \max(d(I_{gt}, I_{ed}), d(I_{ed}, I_{gt})) \quad \text{.....(2.8)}$$

$$d(I_{gt}, I_{ed}) = \max_{a \in I_{gt}} \max_{b \in I_{ed}} ||a - b||$$

The function $d(I_{gt}, I_{ed})$is the directed Hausdorff distance from I_{gt} to I_{ed}. This method is based on the distance among the points; a lesser distance means more closeness between the images.

Euclidean Distance (E_D)

The Euclidian distance E_D between two images is evaluated as: (Nadernejad, Sharifzadeh, & Hassanpour, 2008)

$$E_D(I_{gt}, I_{ed}) = \frac{1}{ij}\sum_{m=0}^{i-1}\sum_{n=0}^{j-1}|I_{gt}(m,n) - I_{ed}(m,n)|^2 \quad (2.9)$$

Average point-to-set distances (D_K)

The average distance between the image's edge pixels and those in the ground truth is calculated as follows (Peli & Malah, 1982):

$$D_K = \frac{1}{I_{ed}}\sqrt[K]{\sum_{p\in I_{ed}} d^K(p, I_{gt})} \quad (2.10)$$

Baddeley's Delta metric (BDM)

BDM is a modified form of the Hausdorff distance (Ma & Grimson, 2005). It's based on the distance between each set's elements, and it's written as (Lopez-Molina, Ayala-Martini, Lopez-Maestresalas, & Bustince, 2017):

$$B_f^K = \left[\frac{1}{|P|}\sum_{p\in P}|fd(p, I_{gt}) - fd(p, I_{ed})|^K\right]^{1/K} \quad (2.11)$$

f is a function which is concave and modulates the point-to-point distance.

F-Score

F-measure is a metric used to determine a test's degree of accuracy for binary classification. The test score is based on both precession and recollection. F has a maximum value of 1 and a minimum value of 0. It is the harmonic mean of precession and recall in equal weighting (Tariq, Hamzah, NG, Wang, & Ibrahim, 2021).

		Predicted	
		Positive	Negative
Ground-Truth	Positive	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

Figure 2.7: Characteristic Matrix

The important parameters in

Figure 2.7 are defined as following:

Precision$(P_r) = \frac{TP}{(TP+FP)}$ Recall $(R_c) = \frac{TP}{TP+FN}$

F-Score = $\frac{\mathbf{2}}{(\frac{\mathbf{1}}{\boldsymbol{P_r}}+\frac{\mathbf{1}}{\boldsymbol{R_c}})}$

2.10 Research Gaps

2.10.1 Masking-Based Techniques

Dependency on Threshold: Masking-based techniques, such as the popular Sobel or Canny edge detectors, rely heavily on predefined thresholds to distinguish between edges and non-edges. Setting these thresholds can be subjective and challenging, leading to suboptimal results. Moreover, the optimal threshold values may vary significantly across different images and application scenarios.

Prone to Errors: Due to their sensitivity to threshold selection, masking-based methods are prone to errors, including false positives and false negatives. In regions with low contrast or noisy backgrounds, setting appropriate thresholds becomes particularly challenging, often resulting in incomplete or inaccurate edge detection.

2.10.2 Soft Computing Methods

Inability to Detect Weak Edges: Soft computing methods, such as fuzzy logic and genetic algorithms, may struggle to detect weak edges effectively. These methods typically rely on fuzzy membership functions or evolutionary optimization techniques, which may not adequately capture subtle edge information present in noisy or low-contrast regions of an image.

Sensitivity to Parameter Tuning: Soft computing methods often involve tuning several parameters, such as membership function parameters or genetic algorithm parameters, to achieve optimal edge detection performance. However, finding the

right parameter settings can be time-consuming and may not always guarantee robust edge detection across diverse image datasets.

2.10.3 ANN and DNN-Based Methods

Complexity: Edge detection using artificial neural networks (ANNs) or deep neural networks (DNNs) can be computationally intensive and complex. DNN architectures, such as convolutional neural networks (CNNs), often require large amounts of training data and extensive computational resources for model training and inference. Many light network designs have been developed recently, such as Fined (Wibisono & Hang, 2020), EDTER (Mengyang Pu, 2022), and PiDiNet (Su, 2021), with the express purpose of detecting edge contours from images. These architectures do not necessitate a large number of datasets or pre-trained models.

Susceptibility to Noise Perturbations: ANN and DNN-based methods are highly susceptible to noise perturbations in input images. Even small amounts of noise can significantly affect the performance of these models, leading to degraded edge detection accuracy. Preprocessing steps such as denoising may be necessary, adding to the overall computational complexity and processing time.

In summary, earlier edge detection methods suffer from various limitations, including sensitivity to threshold selection, inability to detect weak edges, and susceptibility to noise perturbations. Addressing these limitations is crucial for developing more robust and reliable edge detection techniques applicable across diverse image datasets and real-world scenarios.

2.11 Simulation Details

In the realm of computer vision and image processing, the detection of edges plays a pivotal role in extracting important visual cues and features from digital images. To achieve accurate edge detection, sophisticated algorithms are often employed, which necessitates the use of computer simulations to assess their performance under various conditions. These simulations involve the systematic manipulation of a set of

parameters that influence the edge detection process, allowing for a comprehensive evaluation of algorithmic robustness, sensitivity, and overall effectiveness.

The process of edge detection via computer simulation involves the following key steps:

1. **Selection of Edge Detection Algorithm:** Before conducting simulations, an appropriate edge detection algorithm is chosen based on its suitability for the given application and the characteristics of the input images. Common algorithms include Canny edge detector, Sobel operator, and Laplacian of Gaussian (LoG) method, among others.
2. **Definition of Simulation Parameters:** A set of parameters is defined, each of which affects various aspects of the edge detection process. These parameters may include:
    - Threshold values: Parameters that determine the intensity or gradient threshold for identifying potential edge pixels.
    - Kernel size: Parameters related to the size and shape of convolutional kernels used in filtering operations.
    - Noise reduction techniques: Parameters specifying the type and strength of noise reduction methods applied prior to edge detection.
    - Post-processing operations: Parameters governing additional processing steps, such as thinning, connecting, or refining detected edges.
3. **Generation of Synthetic Test Images:** Synthetic test images are generated or selected to serve as inputs for the edge detection simulations. These images may vary in complexity, containing features such as lines, curves, corners, textures, and noise to mimic real-world scenarios.
4. **Parameter Sweep and Optimization:** The defined parameters are systematically varied across a range of values, and the edge detection algorithm is applied to the test images using each parameter configuration. This parameter sweep allows for the exploration of different algorithm settings and their effects on edge detection performance.

5. **Evaluation and Analysis:** Quantitative metrics, such as the F-measure, precision-recall curves, or receiver operating characteristic (ROC) curves, are calculated to evaluate the accuracy and robustness of the edge detection results. Comparative analysis with ground truth data or benchmarks helps assess the algorithm's performance relative to other methods.
6. **Iterative Refinement:** Based on the simulation results and analysis, adjustments may be made to the algorithm parameters or the algorithm itself to improve performance. This iterative refinement process aims to enhance the algorithm's capability to accurately detect edges across a wide range of image types and conditions.

By systematically manipulating and analyzing the set of parameters detailed in later sections, researchers and practitioners can gain valuable insights into the behavior and performance of edge detection algorithms. This approach facilitates the development of more robust and adaptive algorithms capable of effectively extracting edge information from digital images in diverse real-world scenarios.

2.12 Research Objectives

The critical issue in edge detection is to minimize the loss of edge information. In noisy images, this problem is even more significant. Although several edge detection approaches are described in the literature, there is still an urge to improve noise management. This thesis attempts to address some of the major drawbacks of existing methods used for edge detection.

Algorithms are proposed for efficient edge detection and noise removal, which outperform state-of-the-art. The following problems are addressed in the thesis:

While managing noise, many edge detection algorithms do not successfully preserve the small details present in an image. The actual image pixel data is modified due to the filtering process, which results in various side effects on an image, such as blurring, loss of edges, lack of required sharpness, etc. The quality of image data should not be compromised with the filtering process used in edge detection.

Although there has been a massive advancement in the technology used for noise suppression in the last decade, such as the incorporation of neural networks, soft computing, and fuzzy logic, the formulation of such methods is so complex that despitepromising results, their practical use becomes limited. The structure of edge detection algorithms, along with noise management should be easy to implement, and the computation time must range in applicable limits.

Lastly, even if all the parameters discussed above for a good edge detection operator are catered, most of these fail when there is a rise in noise density, resulting in poor localization, false edges, broken edges, etc. The operator should be robust enough to handle even high-density noise levels with consistent performance.

The schematic diagram of the proposed framework is shown in Figure . The proposed system is a two steps process. In the first step, edge refinement is done, and edge detection is done using a soft computing technique in the second step.

Step 1: In this step,the image edges are refined using smoothening and sharpening methods. In noisy images, unwanted edges which occur because of noiseare suppressed using image smoothening. In contrast,edges can be enhanced in the case of blurred images using the image sharpening method to make them easily detectable. In this work, both approaches are considered.

Step 2: In this step, edge detection is performed on the edge-refined images. For edge detection, soft computing techniques, ant colony optimization (ACO), and Fuzzy logic are considered.ACO is inspired by the movement of ants they used for food searching. In ACO-based edge detection, ants move on a complete image and decide whether a chosen pixel is an edge or non-edge pixel. The movement and memory of ants are considered for determining the edge or non-edge pixel. The ACO parameters are linked with intensity differences of the pixels. It is also important to note that a chosen pixel is an edge pixel if most ants decide in their favour.

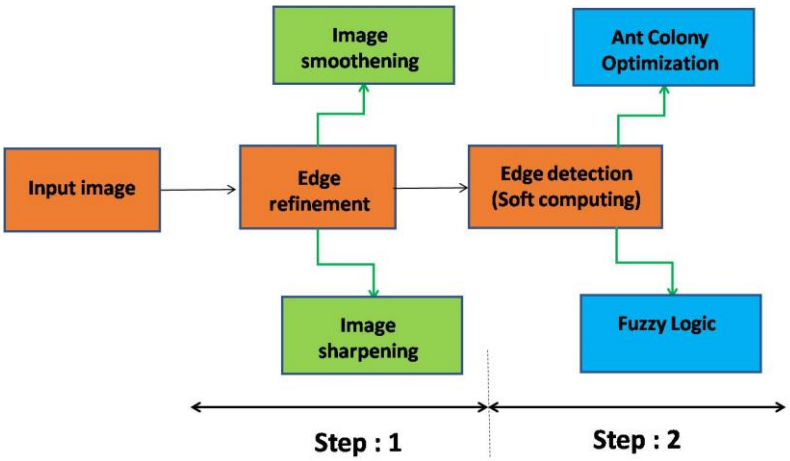


Figure 2.6 Framework of proposed work

In fuzzy logic method of edge detection, fuzzy rules are developed to identify whether a pixel should be deemed an edge. In fuzzy based system, the correctness of the algorithm is heavily dependent on how the rules are devised. The selected input/output membership functions are also important in the accuracy of the results. In this work, we have carefully designed fuzzy rules along with input and output membership functions to get improved results.

Literature abounds on edge detection operators. Many operators filter images that produce other artefacts in the image in the form of a blur, missing true edges, loss of original pixel information, edge distortion, and many more. An operator performing better edge refinement also makes the best balance between noise removal and preserving the image's fine details, which should be the primary choice before extracting the edges. This research aims to develop a new image edge detection scheme that provides more accurate and efficient detection of edges. The objectives of this thesis, after the broad analysis of techniques present in the literature, are asfollows:

1. To devise a reliable edge detection schemethat first refine the edges to efficiently remove noisy pixels while preserving and enhancing the image's fine details.
2. To work towards improved edge detectors using fuzzy logic,which aids in better computational complexity and is easy to implement.
3. To explore the utility of soft computing methods in edge detection to improve the

qualitative and quantitative results.

4. To create a robust edge detection operator whose performance is not degraded with a rise in noise density.

Chapter Three

Edge Detection using ACO under Novel intensity mapping function

3.1 Introduction

Image edge detection is a prime problem in image segmentation. The traditional methods are based on the design of a kernel. Thus, various kernel-based methods like Canny, Sobel, Robert, Laplacian, and Prewitt have been proposed. These methods are simple and successful in finding the edges from different images. However, they are very sensitive to noise and not perfect because they identify false boundaries alongwith genuine ones. Also, the number of false edges are more than true edges. These problems occurs in the methods which use fixed set of thresholds. A technique that dynamically alters its threshold would be preferable to eliminate incorrectly identified edges. Methods based on artificial and swarm intelligence are able to manage minute details. Therefore, several techniques based on swarm intelligence have been proposed in the literature to tackle the problem of edge detection.

One such technique is Ant Colony Optimization (ACO). This methodology is fundamentally based on the perception of real-ant colonies. In the early 1990s, this algorithm was presented by M. Dorigo et al. (Dorigo, Mauro, & Stutzle, 2006). In the past, various ACO-based approaches for edge detection in images and video frames have been proposed (Tian, Weiyu, & Shengli, 2008) (Ari, Ghosh, & Mohanty, 2014). These papers present some mechanisms to enhance the detected edges further. However, none of these papers uses any statistical measure to observe the quality of edge detection. This chapter proposes a method of edge detection using ACO under a novel intensity mapping function, along-with results that are contrasted with recently proposed alternative methods using statistical measures.

3.2 Novelty of the Proposed Method

Edge detection using Ant Colony Optimization (ACO) under a novel intensity mapping function introduces a unique approach to enhancing the detection of edges in digital images. This innovative method leverages the principles of swarm intelligence

inspired by the foraging behavior of ants to efficiently identify edges while mitigating common challenges encountered in traditional edge detection techniques.

3.2.1 Ant Colony Optimization (ACO):

- ACO is a metaheuristic optimization algorithm inspired by the foraging behavior of ants. It mimics the pheromone-based communication and decentralized decision-making observed in ant colonies to find optimal solutions to complex problems.
- In the context of edge detection, ACO can be employed to explore the image space and identify regions with significant intensity gradients indicative of edges.

3.2.2 Novel Intensity Mapping Function:

- The intensity mapping function plays a crucial role in preprocessing the image data to enhance edge visibility and distinguish edges from the background noise effectively.
- The novel intensity mapping function proposed in this approach is designed to amplify the contrast between edge pixels and non-edge pixels, thereby improving the discriminative power of edge detection algorithms.

3.2.3 Integration of ACO and Intensity Mapping:

- ACO is utilized to search for optimal edge locations within the image space, guided by the intensity gradients introduced by the mapping function.
- By incorporating information from the intensity mapping, ACO can efficiently navigate the image space and converge towards regions with pronounced edge features, reducing the computational complexity and enhancing the accuracy of edge detection.

3.2.4 Benefits and Advantages

- Robustness: The synergy between ACO and the intensity mapping function enhances the robustness of edge detection by mitigating the effects of noise and variations in illumination.

- Adaptive: The intensity mapping function can be adaptively adjusted based on the characteristics of the input image, ensuring optimal edge enhancement across diverse datasets and application scenarios.
- Computational Efficiency: By leveraging the collective intelligence of the ant colony, the proposed method achieves efficient exploration of the image space, minimizing the computational overhead associated with exhaustive search strategies.

3.2.5 Potential Applications

- Medical Imaging: Improved edge detection can aid in the detection and segmentation of anatomical structures in medical images, facilitating diagnosis and treatment planning.
- Object Recognition: Enhanced edge detection can contribute to more accurate object recognition and classification in computer vision applications.
- Autonomous Systems: Reliable edge detection is essential for enabling autonomous systems such as drones and robots to perceive and navigate their environments effectively.

In summary, edge detection using ACO under a novel intensity mapping function offers a promising approach to address the challenges of traditional edge detection methods, providing enhanced accuracy, robustness, and computational efficiency across a wide range of applications.

3.3 Image Edge Detection Using ACO

In ACO, the behaviour of ants in searching for food is utilized for the problem of edge detection.The herd of ants works cooperatively to find the best path in the search for food. These ants have memory and trust in each other, and the optimal path is made based onmost ants' path decisions. The procedure of image edge detection (Ari, Ghosh, & Mohanty, 2014) contains the accompanying steps:

3.3.1 Initialization phase

In this procedure, a picture I_{MN} (where M and N represent in size) is input information on which ants travel to find solutions. Each pixel has its unique location in the image,

represented by (m, n) and intensity value $I_{m,n}$. In the initialization phase, a fixed number of ants K is decided by the size of an image randomly distributed onthe image with the end goal, where each pixel in the image is treated as a node. The pheromone matrix $\tau^{(0)}$ is also initialized with a constant value which is small but non-zero.

3.3.2 Construction phase

Initially, the pheromone matrix $\tau^{(0)}$ is initialized with some pre-defined constant values. One out of K ants is randomly selected at the n^{th} construction step, and this ant continuously moves S number of steps on the image. The ant's movements to its neighbouring node (x, y) depend on transition probability and are defined as:

$$p^{n}_{(l,m)(x,y)} = \frac{\left(\tau^{n-1}_{x,y}\right)^{\alpha}(\eta_{x,y})^{\beta}}{\sum_{x,y\in\Omega_{(l,m)}}\left(\tau^{(n-1)}_{x,y}\right)^{\alpha}(\eta_{x,y})^{\beta}}, \quad y\varepsilon\Omega_{x} \text{.....(3.1)}$$

In the above equation, $\tau^{n-1}_{x,y}$ is the pheromone value. Parameter $\Omega_{(l,m)}$ denotes all possible neighbourhood nodes of the node (*l*,*m*) and $\eta_{x,y}$ is a heuristic parameter at a particular node (x,y). The parameters α and β control the effect of the pheromone and heuristic matrix, respectively.

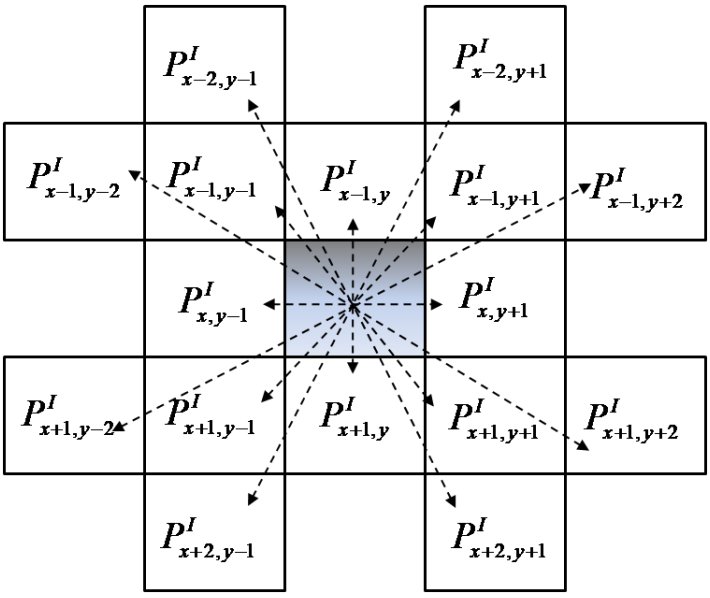


Figure 3.1: Pictorial representation of clique

The procedure consists of two important issues:

Using the inner clique determines the heuristic data:

$$\eta_{x,y} = \frac{G_c(x,y)}{\sum_{x=1:M}\sum_{y=1:N} G_c(x,y)} \quad \text{.....(3.2)}$$

The function $G_c(x,y)$ is a function of the local group of pixels defined as a clique; its value depends on the difference in intensity values of pixels in the clique. The graphical representation of the clique is shown in Figure 3.1. For the pixel, $P^I_{(x,y)}$, the value of function $G_c(x,y)$ is:

$$G_c(P^I_{x,y}) = F\begin{pmatrix} \left|P^I_{x-2,y-1} - P^I_{x+2,y+1}\right| + \left|P^I_{x-2,y+1} - P^I_{x+2,y-1}\right| + \left|P^I_{x-1,y-2} - P^I_{x+1,y+2}\right| \\ +\left|P^I_{x-1,y-1} - P^I_{x+1,y+1}\right| + \left|P^I_{x-1,y} - P^I_{x+1,y}\right| + \left|P^I_{x-1,y+1} - P^I_{x-1,y-1}\right| \\ +\left|P^I_{x-1,y+2} - P^I_{x-1,y-2}\right| + \left|P^I_{x,y-1} - P^I_{x,y+1}\right| \end{pmatrix} \quad \text{.....(3.3)}$$

The function $F(\cdot)$ is proposed as a new mapping function to capture the intensity variations better. This function $F(.)$ is defined as in Eq. (3.4). In this mapping function, the first term scales the pixel difference defined by Eq. (3.3), and the second term is the perturbation term included to tackle small variations in intensities.

$$F(x) = \mu x + sin\left(\frac{\pi x}{2\mu}\right) \text{ for } 0 \leq x \leq \mu \quad \text{.....(3.4)}$$

To completely map pixel values of a particular image, $F(x)$ is unique for each image. The developed function is considered two types of variations, one linear and the other sinusoidal, which fits on a large number of images, as shown in the result section. The parameter μ is a scaling constant. The ants' movement (i.e., $\Omega_{(i,j)}$ as in Eq. (3.1) is considered the 8-connectivity neighbourhood, as demonstrated in Figure 3.2.

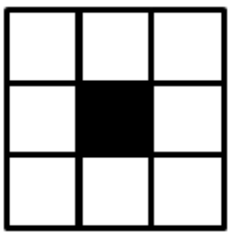


Figure 3.2: Schematic of 8-connectivity neighbourhood

3.3.3 Update phase

In the update process, the pheromone matrix is updated twice. Every time an ant visits a pixel, it updates the corresponding pheromone value. The local update, which denotes the updated pheromone matrix of each ant, is given by:

$$\tau_{x,y}^{(n-1)} = \begin{cases} (1-\rho)\tau_{x,y}^{(n-1)} + \rho\Delta_{i,j}^{k} & if\ (x,y) \in vca \\ \tau_{x,y}^{(n-1)} & otherwise \end{cases} \quad \text{.....(3.5)}$$

Where *'vca'* means 'visited current ant', ρ is the rate of evaporation of pheromone.The second update, called the global update of the pheromone matrix, is done when all the ants complete one construction step.

$$\tau^{(n)} = (1-\psi)\tau^{(n-1)} + \psi\tau^{(0)} \quad \text{.....(3.6)}$$

ψ denotes the decay of pheromones (Tian, Weiyu, & Shengli, 2008).

3.3.4 Decision Phase

We chose the initial threshold $Th^{(0)}$ as the mean value of the pheromone matrix. Then the following steps are performed:

Step 1: Initialize $Th^{(0)}$ as:

$$Th^{(0)} = \frac{\sum_{x=1:M}\sum_{y=1:N}\tau_{x,y}^{(n)}}{MN} \quad \text{.....(3.7)}$$

And fix the iteration index as q= 0.

Step 2: Now pheromone matrix $\tau^{(n)}$ is divided into two classes as below:

$$m_L^{(q)} = \frac{\sum_{x=1:M}\sum_{y=1:N} c\tau_{x,y}^{(n)}}{\sum_{x=1:M}\sum_{y=1:N}\tau_{x,y}^{(n)}} \quad for\ c < Th^{(q)} \quad \text{.....(3.8)}$$

$$m_U^{(q)} = \frac{\sum_{x=1:M}\sum_{y=1:N} c\tau_{x,y}^{(n)}}{\sum_{x=1:M}\sum_{y=1:N}\tau_{x,y}^{(n)}} \quad for\ c \geq Th^{(q)} \quad \text{.....(3.9)}$$

Step 3: Fix the index of iteration $q = q+ 1$, and we update the threshold as given below:

$$Th^{(q)} = \frac{m_L^{(q)} + m_U^{(q)}}{2} \quad \text{.....(3.10)}$$

Step 4: In the case of $\left|Th^{q} - Th^{(q-1)}\right| > \epsilon$, after this, move on to *2nd Step*; if not, then the iteration method is discontinued, and a decision is made on all pixel's locations (x, y) to find out the edge using:

$$E_{x,y}^{d} = \begin{cases} 1 & \tau_{x,y}^{(n)} \geq Th^{(q)} \\ 0 & elsewhere \end{cases} \quad \text{.....(3.11)}$$

3.3.5 Proposed Modifications

In the first suggested modification, step-3 is modified as follows:

Step 3: Fix the index of iteration $q = q+ 1$, and we update the threshold as given below:

$$Th^{(q)} = \frac{w_1 m_L^{(q)} + w_2 m_U^{(q)}}{2} \quad \text{.....(3.12)}$$

Where w_1 and w_2 are the weights given to both thresholds satisfying $w_1 + w_2 = 1$.

In the second modification, step-4 is defined as:

Step 4: In continuation of step 4, the following condition is also included. For each pair of the values w_1 and w_2, threshold calculation is done. Then using this threshold, output images and F-score is also calculated, and finally, the value of w_1 and w_2 is selected for which F-score is maximum.

3.4 Simulation and Results

The performance of ACO based edge detection method with the novel intensity method is done using computer simulation in MATLAB(R) 2015a. For simulation, we utilized a computer system equipped with an Intel Core i7-11700K processor, which features 8 cores and 16 threads, with a base clock speed of 3.6 GHz and a turbo boost up to 5.0 GHz. The system is supported by 32 GB of DDR4 RAM running at 3200 MHz, ensuring sufficient memory bandwidth and capacity for handling extensive computational tasks. For graphics processing, we employed an NVIDIA GeForce RTX 3080 GPU with 10 GB of GDDR6X VRAM, which facilitated accelerated computations and improved performance. The storage requirements were met with a 1 TB SSD, providing fast read/write speeds essential for efficient data handling. The operating system used was Windows 10 Pro 64-bit, which is compatible with the software tools employed in the simulation. The simulation is performed on BSD (Berkeley Segmentation Data set).

In total, we have considered six images, numbered 1-6 (Figure 3.3), as well as ground truth images (g) with ideal edges.

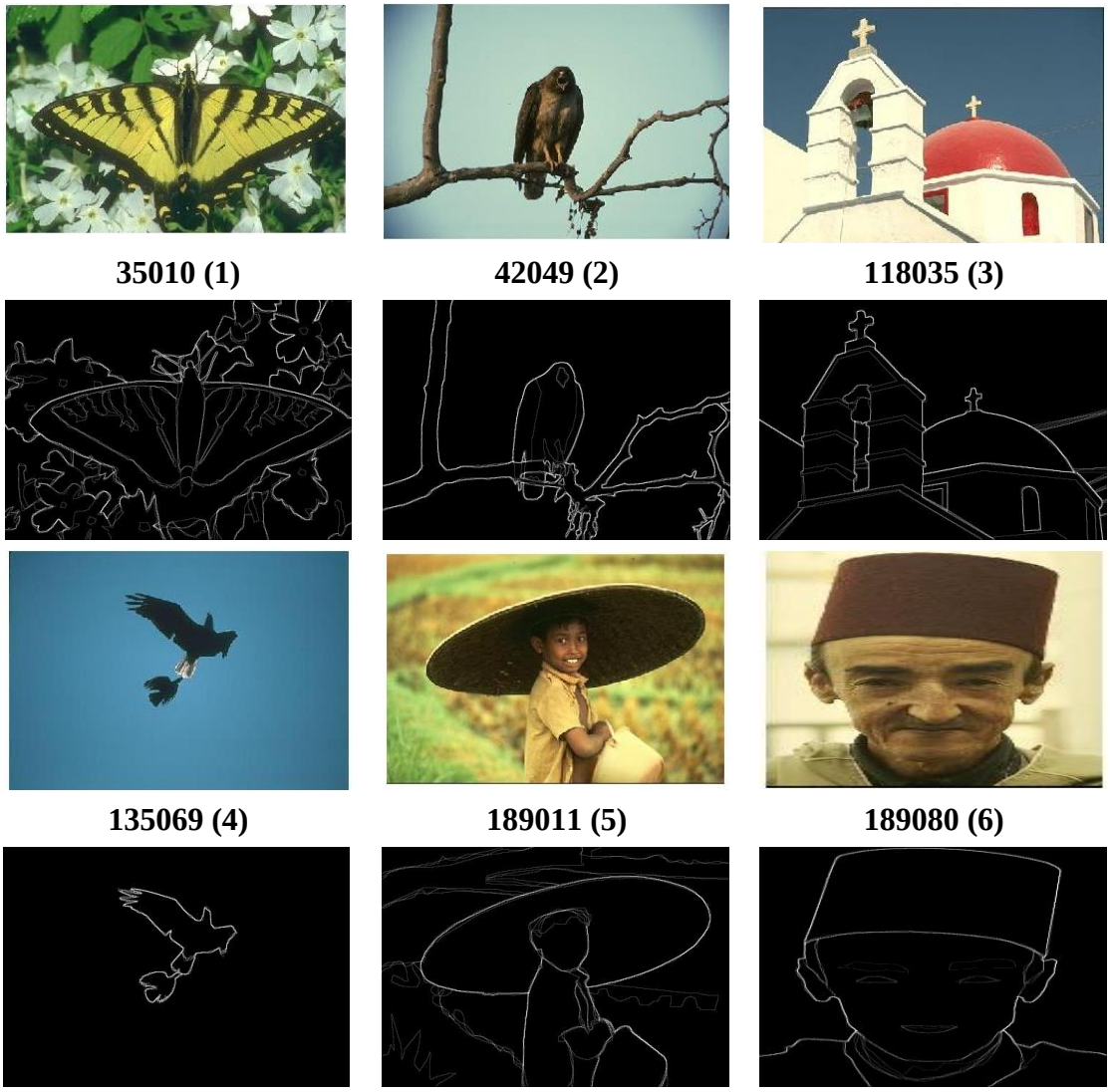


Figure 3.3: Sample images from the BSD image database and their ground truths

In Figure 3.4, intensity profiles for all six images are shown. In all six profiles, linear and sinusoidal variations can be seen, except in image 6, where a narrow spike can be seen. Thus, images where abrupt profile changes are seen sinc function will be a better option, but still chosen function fit most of the images with fair accuracy. In particular, images 3 and 4 can be approximated by a linear function; similarly, images 1 and 5 can be approximated with a sinusoidal function, while image 2 and 6 is well approximated by the addition of both linear and sinusoidal function. The difference between exact mapping and approximated function leads to a difference of nearly 1.5 to 2.0 dB in PSNR, while accuracy is affected by 0.5 to 2.5%. We calculate performance in terms of PSNR, Accuracy and F-measure.

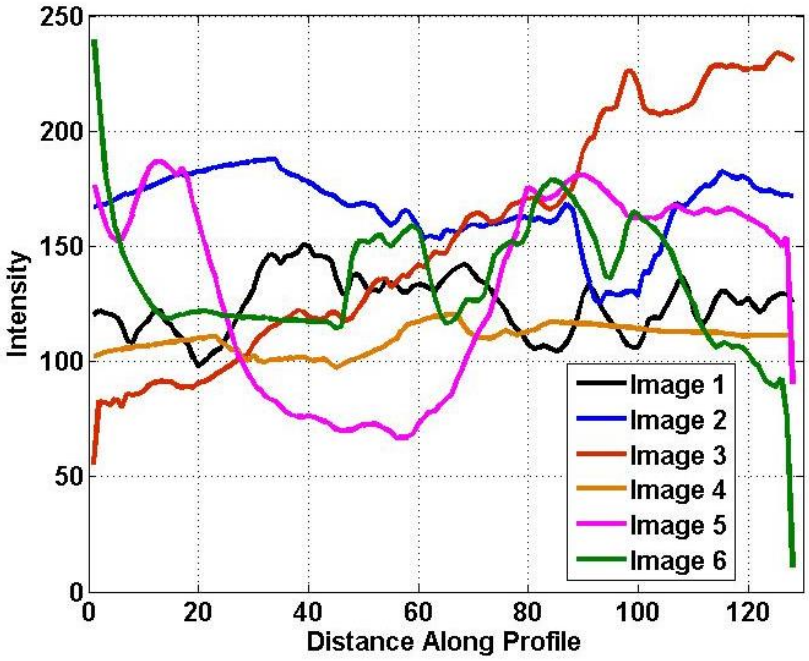


Figure 3.4: Intensity profiles for all six images

The (PSNR) is given by:

$$PSNR = \frac{[255]^2}{MSE} \quad \text{.....(3.13)}$$

Where:

$$MSE = \frac{[I_g(i,j) - I_o(i,j)]^2}{mn} \quad \text{.....(3.14)}$$

In above $I_g(i,j)$ denotes ground-truth images and $I_o(i,j)$ is edge-detected output image.

The accuracy is defined as:

$$Accuracy = \frac{TE}{TE+FE} \quad \text{.....(3.15)}$$

where, TE=True Edges and FE=False Edges

Accuracy is an important factor in measuring the performance of an edge detection algorithm, and its ideal value is 1. Because of the problem of false edge detection, accuracy goes down, and in many traditional methods, it is very low as more false edges are detected than true edges.

		Actual	
		Positive	Negative
Predicted	Positive	True Positive(TP)	False Positive(FP)
	Negative	False Negative(FN)	True Negative(TN)

Figure 3.5: Characteristic matrix

The important parameters are defined as follows:

$$P_r = \frac{TP}{TP+FP}, \quad R_c = \frac{TP}{TP+FN} \quad \text{....(3.16)}$$

where,

P_r=Precision, R_c=Recall

Finally, the *F-Measure* is given by:

$$F - Measure = \frac{2P_r R_c}{P_r + R_c} \quad \text{....(3.17)}$$

F-measure is a test of accuracy in binary classification. It depends on both precession and recall to get a test score. The maximum value of *F* is 1 with a minimum of 0.

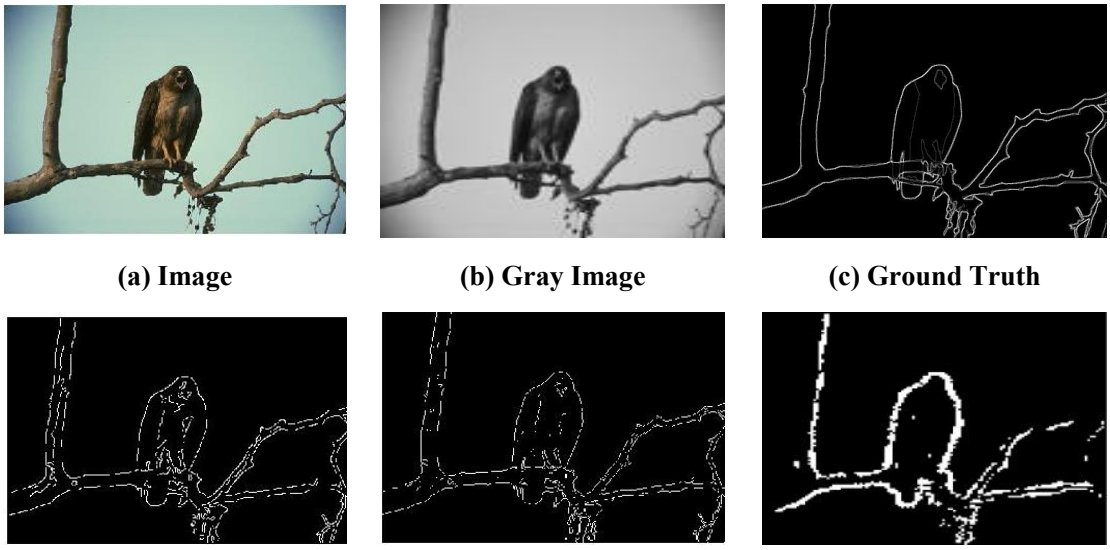
The simulation parameters are detailed in Table 3.1. The total number of ants needed to be taken depends on image size. If the image is under consideration of size (m×n), then the number of ants is $\sqrt{mn}$.

Table 3.1: Simulations Parameters

Parameter	**Value**
Total number of construction steps	8
pheromone matrix, τ_{init} (Initial values)	0.0001
Pheromone information, α (Weighting factor)	1
Heuristic information, β (Weighting factor)	0.1
Connectivity neighbourhood, Ω	8
Functions adjusting parameter, μ	10
evaporation rate, ρ	0.1
Total number of ants K	Vary
Total number of ant's movementsteps, *S*	40

Pheromone decay coefficient, ψ	0.05
Tolerance value, ε	0.1
Threshold, *Th*	Adaptive

In Figure 3.6, six images (a-f) are shown. Images description is also given. If we carefully examine image (d), we observe that it detects most of the true edges but may also detect many false edges. The Sobel method tries to discard false edges, but in doing so, it also discards true edges (e).



(a) Image **(b) Gray Image** **(c) Ground Truth**

(d) Canny **(e)Sobel** **(f) ACO**

Figure 3.6: Results comparison of different algorithms

However, ACO detects large numbers of true edges with few false edges (f). It is very difficult to judge the quality of the image by using the human visual system; therefore, performance measures, as discussed above, are used for comparisons of methods. In our work, we have shown a comparison with Sobel and Canny methods, which are still used in edge detection methods. The main aim of choosing these two methods is to show the effectiveness of ACO methods over currently used edge detection methods.

In Figure 3.7, PSNR (dB) is plotted for all six images under consideration. In terms of PSNR, a widely used metric for assessing the fidelity of image processing algorithms, the performance of the Sobel and Canny edge detection methods appears to be

comparable. However, it's crucial to note that the PSNR values obtained for both Sobel and Canny edge detection fall below the threshold of 20 dB in most cases, indicating poor quality in terms of preserving image information.

Typically, in image processing, a good quality image is expected to have a PSNR greater than 30 dB. PSNR measures the ratio between the maximum possible power of a signal and the power of corrupting noise that affects the fidelity of its representation, with higher values indicating better preservation of image details and lower levels of distortion.

The evaluation reveals that the edge-detected images produced by Sobel or Canny detection methods are deemed unusable due to their low PSNR values. Despite their effectiveness in detecting edges, the resulting images suffer from significant loss of image quality and fidelity, making them unsuitable for practical applications where preserving image details is paramount.

On the contrary, the ACO method demonstrates remarkable performance in terms of PSNR. Across all cases evaluated, the PSNR values obtained with ACO edge detection consistently exceed the threshold of 30 dB, indicating excellent image quality and fidelity preservation. Particularly noteworthy is the PSNR value of nearly 44 dB obtained for image 4, highlighting the exceptional quality achieved by the ACO method in edge detection tasks.

The superior PSNR values attained by the ACO method underscore its effectiveness in accurately preserving image details while detecting edges, even under challenging conditions. These results suggest that the ACO-based edge detection approach not only outperforms Sobel and Canny methods in terms of image quality but also offers greater reliability and usability in practical image processing applications.

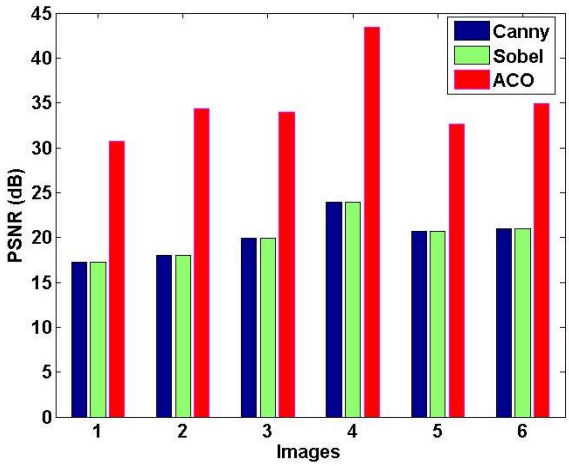


Figure 3.7: PSNR comparison for different algorithms

In edge detection, accuracy is an important factor, as most kernel-based methods successfully identify the edges, but these methods also detect false edges. When compared to true edges, the number of false edges is higher. In Figure 3.8, the accuracy of different methods is shown; the performance of the Canny method is better in comparison to the Sobel method, but still, the accuracy is below 20%. In the context of Ant Colony Optimization (ACO) for edge detection, the minimum accuracy achieved is notably high, reaching 87%. This impressive level of accuracy is attributed to the inherent characteristics of the ACO algorithm, which effectively minimizes the occurrence of falsely accepted edges while maximizing the detection of true edges.

One key factor contributing to the high accuracy of ACO-based edge detection is the stringent criteria employed by the algorithm for edge identification. ACO algorithms typically incorporate sophisticated mechanisms for evaluating and selecting edge candidates based on multiple criteria, such as edge strength, gradient magnitude, and contextual information. As a result, the likelihood of falsely accepting noise or spurious features as edges is significantly reduced.

Moreover, while ACO algorithms may occasionally encounter states of confusion or uncertainty, where true edges are erroneously rejected, such instances are relatively rare. The robustness of ACO-based edge detection stems from its ability to effectively

navigate complex image landscapes and discern genuine edge structures from background noise or artifacts.

Furthermore, ACO algorithms are designed to prioritize the detection of true edges over the acceptance of false edges. This strategic emphasis ensures that the number of detected true edges significantly outweighs the instances of falsely accepted edges, leading to a higher overall accuracy rate.

In summary, the high accuracy achieved with ACO-based edge detection can be attributed to several factors:

- Rigorous criteria for edge identification, minimizing false positives.
- Robustness in navigating image complexities and distinguishing true edges from noise.
- Strategic prioritization of true edge detection, mitigating the impact of false acceptances.

By leveraging these strengths, ACO-based edge detection algorithms demonstrate exceptional performance in accurately delineating edge structures within digital images, making them a valuable tool for various image processing and computer vision applications.

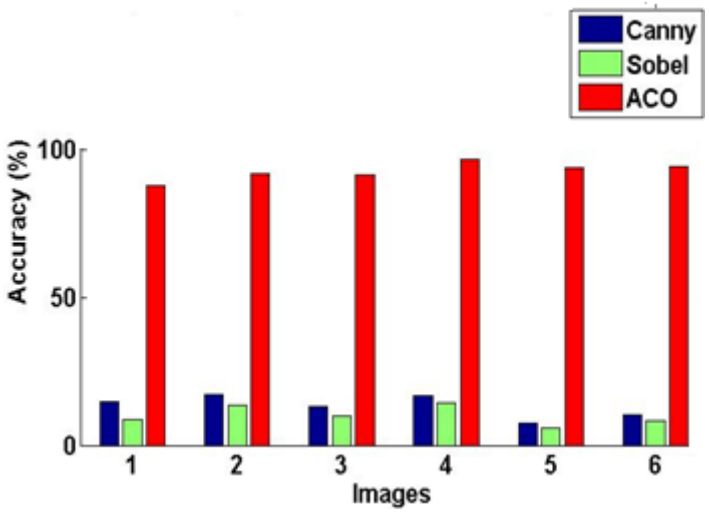


Figure 3.8: Accuracy comparison for different algorithms

In Figure 3.9, the F-score is depicted for all six images, serving as a comprehensive measure of the accuracy and effectiveness of the edge detection methods employed. Each ground truth image achieves an F-score of 1, indicating perfect alignment between the detected edges and the actual edges present in the image. However, when utilizing basic edge detection methods such as Sobel and Canny without incorporating any morphological operations, significant errors are observed, leading to F-measure values falling below one.

The performance shortcomings of these basic methods are attributed to their inherent limitations in accurately identifying and delineating edge structures within the images. Without the refinement provided by morphological operations, these methods are prone to errors such as false positives, missed edges, and inaccuracy in edge localization. As a result, the F-measure, which considers both precision and recall, is adversely affected, resulting in values below unity.

To address these deficiencies and enhance the F-score, further improvements are implemented, as detailed in Table 3.2. These improvements likely encompass additional processing steps, parameter adjustments, or algorithmic enhancements aimed at refining the edge detection results and reducing the occurrence of errors.

Notably, the F-scores obtained with the Ant Colony Optimization (ACO) method demonstrate substantial improvements over the basic Sobel and Canny methods. Despite starting with F-scores below unity, the application of ACO yields F-scores ranging from 0.67 to 0.97, indicating a significant enhancement in edge detection accuracy and alignment with ground truth edges.

The achieved F-scores with ACO reflect the effectiveness of the algorithm in overcoming the limitations of traditional edge detection methods and producing more reliable and precise edge maps. By leveraging sophisticated optimization techniques and robust criteria for edge identification, ACO achieves remarkable performance improvements, leading to F-scores that approach perfection across various image datasets.

In summary, while basic edge detection methods such as Sobel and Canny may initially yield suboptimal results, further enhancements and the adoption of advanced techniques such as ACO can significantly improve the F-score, resulting in edge detection outputs of excellent quality with enhanced accuracy and fidelity.

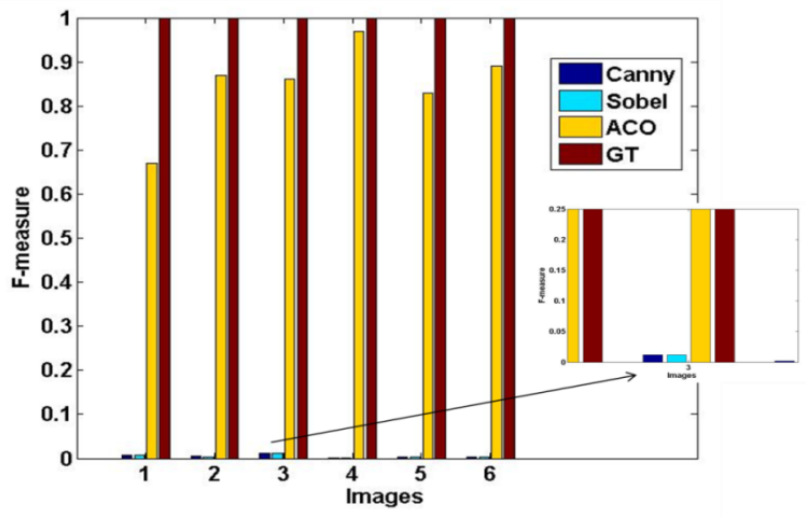


Figure 3.9: F-measure comparison for different algorithms

Table 3.2: F-Score and Weights

Image	Weight pair (w_1 ,w_2)	F-Score (old, new)
1	[(0,1)	(0.67153, 0.67282)
2	(0.5,0.5)	(0.87325, 0.87503)
3	[(0.9, 0.1)	(0.85650, 0.85850)
4	(0,1)	(0.97108, 0.97153)
5	(0.9, 0.1)	(0.82623, 0.82854)
6	(0,1)	(0.88510, 0.88711)

Figure 3.10 to Figure 3.15 shows F-score variations for all six images. The dot marked on each figure is the value obtained from previous methods. It is clear from the figures that ups and downs are seen in results, but by tuning the parameters, the result could be better in comparison to old methods. It is also noticeable that the number of edges in an image is very large; therefore, a small increment in F-score significantly improves edge detection. Table 3.2 shows the maximum value of the F-

score and corresponding values of w_1 and w_2 for all six images.

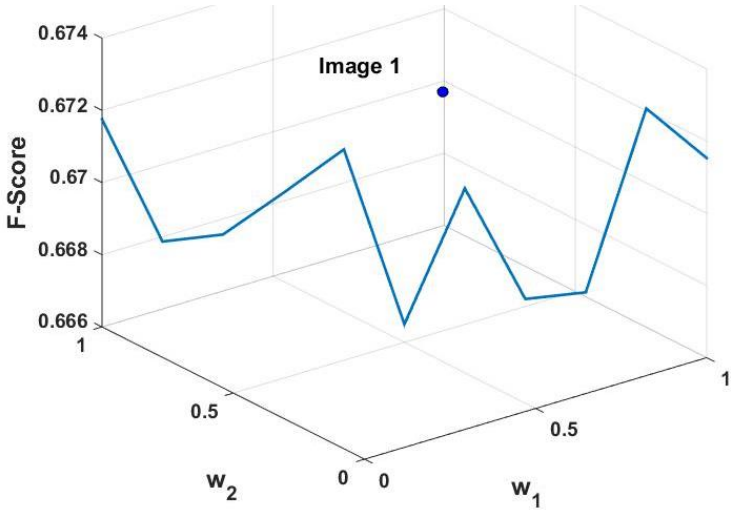


Figure 3.10: F-measure variation for image 1

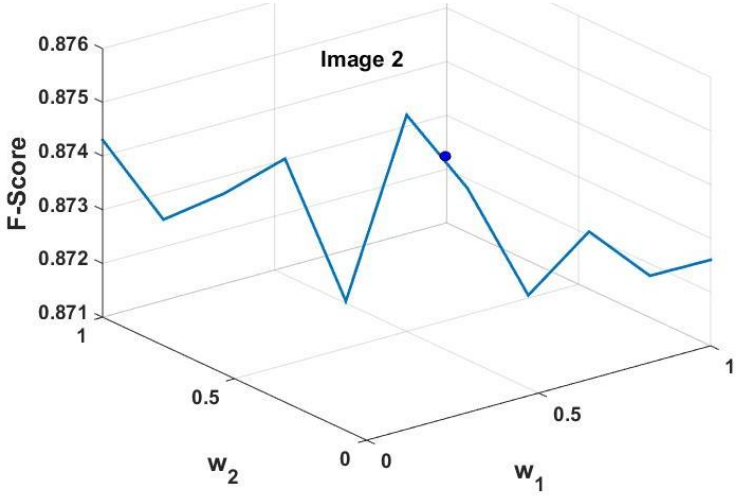


Figure 3.11: F-measure variation for image 2

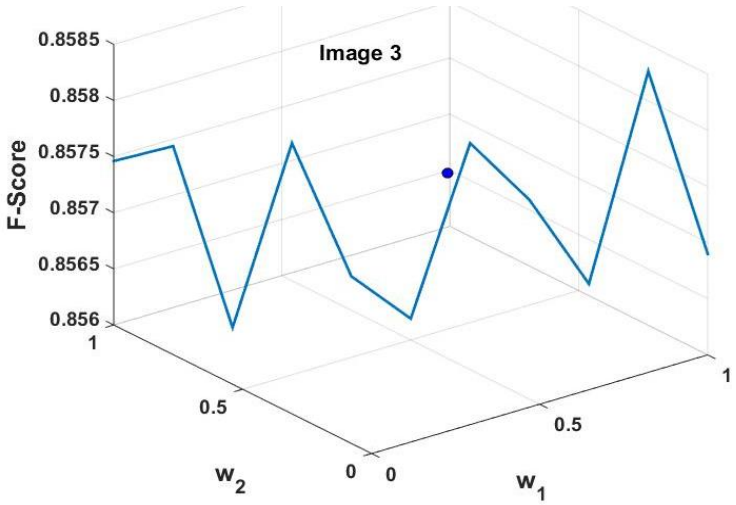


Figure 3.12: F-measure variation for image 3

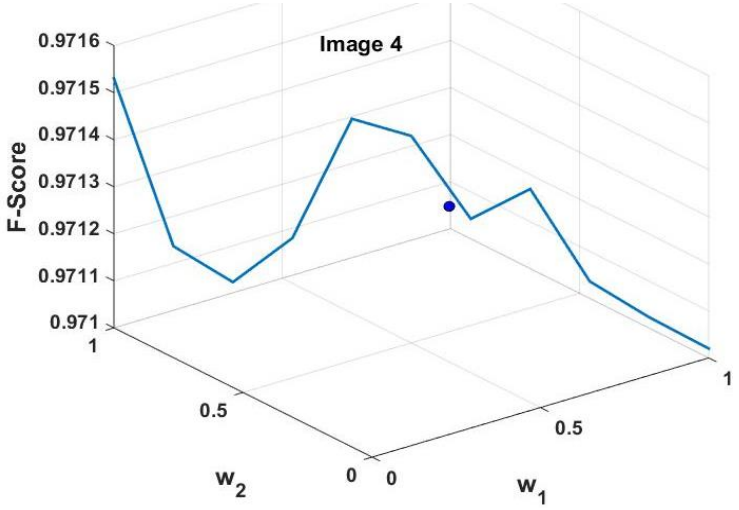


Figure 3.13: F-measure variation for image 4

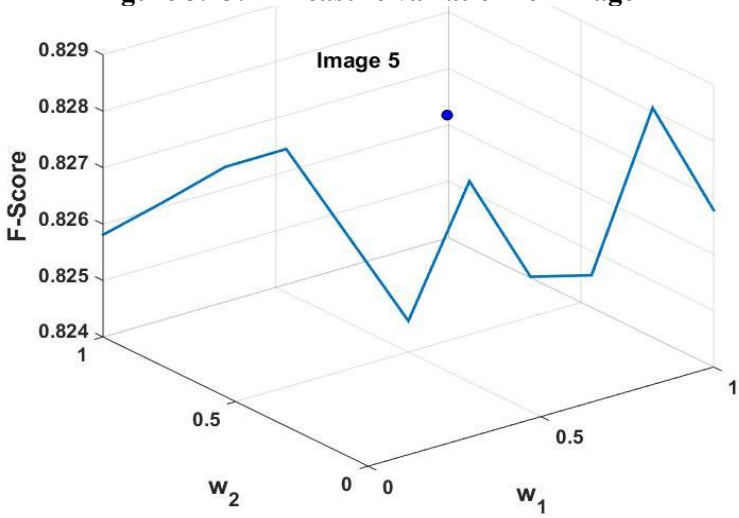


Figure 3.14: F-measure variation for image 5

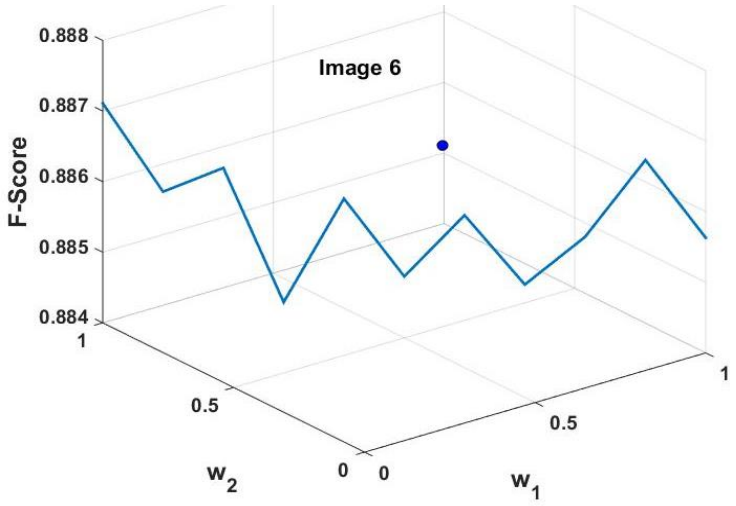


Figure 3.15: F-measure variation for image 6

Thus, edge detection using the ACO technique is a good choice. The obtained F-score using other methods proposed recently is shown in Table 3.3. In the table, F-score is presented after applying morphological operations. The F-measure for the Canny and the Sobel methods are 0.49 and 0.40, respectively. For the BEL method, it is 0.63, while for gPb and structure forest, it is 0.71. For the sketch token F-score is 0.73. However, in our case, F-score varies from 0.67 to 0.97. It is also noticeable that our method has not used any morphological operations for contour generation, edge joining, etc.

Table 3.3: Comparison with Notable Works

Method	Year	F-measure
Canny (Canny, 1986)	[1996]	0.49
Sobel (Vincent & Folorunso, 2009)	[2009]	0.40
BEL (Dollar, Tu, & Belongie, Supervised learning of edges and object boundaries, 2006)	[2006]	0.63
gPb (Arbelaez, Maire, Fowlkes, & Malik, 2011)	[2011]	0.71
Sketch Token (Lim, Zitnick, & Dollar, 2013)	[2013]	0.73
Structured Forest (Dollar & Zitnick, Structured forests for fast edge detection, 2013)	[2013]	0.71
Proposed		0.83

3.5 Summary

In this chapter, an ACO-based edge detection method is detailed and obtained results are compared with recently proposed methods. In this work, we have come up with a novel pixel mapping function, and it has been found that the ACO method is very efficient, with an average detection accuracy of nearly 87%. The obtained F-score is very good and outperforms the recently proposed methods. The PSNR value is of very good quality.The weighted method is effective in maximizing F-scores. Sketch Token provides the best F-score of 0.73, and with the proposed work, the best F-score is 0.97, while average taken score is 0.83; therefore, a percentage improvement of 32.80% is observed with the proposed method.

Chapter Four

Type-1 Fuzzy Logic and Guided Smoothing for Edge Detection

4.1 Introduction

In this chapter, another soft computing technique in conjunction with guided smoothing has been experimented with for edge detection,intending to get better results. As discussed in Chapter Two fuzzy-based edge detection can be effectively used in edge detection. However, only using fuzzy methods is insufficient as they may suffer from the problem of inaccurate edge detection because of image noise. In this work, the noise problem is first dealt with, and then edges are calculated using Fuzzy edge detection. This chapter describes a fuzzy logic-based edge detection method where noise because of sharpening is managed by a Gaussian filter, and the quality of the edges is controlled by a sharpening-guided filter. The accuracy of the method is judged using a variety of statistical measures.

4.2 Novelty Of the Proposed Method

4.2.1 Type-1 Fuzzy Logic

- Type-1 Fuzzy Logic extends traditional binary logic to handle uncertainty by allowing for degrees of truth between 0 and 1. It enables the representation of vague or imprecise information and provides a framework for making decisions based on fuzzy rules.
- In the context of edge detection, Type-1 Fuzzy Logic can be employed to model the ambiguity inherent in edge boundaries, where pixels may belong to both edge and non-edge regions to varying degrees.

4.2.2 Guided Smoothing

- Guided smoothing techniques leverage additional guidance information, such as an edge map or a reference image, to adaptively adjust the smoothing process while preserving edge details. Unlike traditional smoothing filters, guided

smoothing methods prioritize retaining sharp transitions, making them well-suited for edge-aware image processing tasks.

- By incorporating guidance information, guided smoothing can effectively suppress noise and unwanted texture while preserving important edge structures in the image.

4.2.3 Integration of Type-1 Fuzzy Logic and Guided Smoothing:

- Type-1 Fuzzy Logic is utilized to formulate fuzzy rules that capture the uncertainty associated with edge detection. These rules define the relationship between input image characteristics (e.g., pixel intensity gradients, local image statistics) and the likelihood of a pixel belonging to an edge.
- Guided smoothing acts as a complementary technique by providing additional context and spatial coherence during the edge detection process. It helps refine the fuzzy membership assignments by adaptively smoothing regions that are less likely to contain true edges while preserving important edge features.

4.2.4 Benefits and Advantages

- Robustness to Noise: The integration of Type-1 Fuzzy Logic and guided smoothing enables edge detection algorithms to handle noisy input images more effectively. Fuzzy logic can accommodate uncertainty and variability in image content, while guided smoothing suppresses noise without compromising edge integrity.
- Adaptive Edge Enhancement: By combining fuzzy reasoning with guided smoothing, the proposed approach can adaptively enhance edges based on local image characteristics and contextual information. This adaptability ensures that edge detection performance remains consistent across diverse image datasets and scenarios.
- Edge Preservation: Guided smoothing techniques prioritize the preservation of edge structures, even in the presence of strong noise or complex textures. This property is essential for maintaining the fidelity of edge features while suppressing unwanted artifacts introduced during preprocessing.

4.2.5 Potential Applications

- Medical Imaging: Accurate edge detection is critical for medical image analysis tasks such as tumor detection, organ segmentation, and anatomical landmark localization.
- Remote Sensing: Edge detection plays a vital role in extracting features from satellite or aerial imagery for applications such as land cover classification, urban planning, and environmental monitoring.
- Industrial Inspection: Edge detection techniques are employed in quality control processes to detect defects, measure dimensions, and identify anomalies in manufactured products.

In summary, integrating Type-1 Fuzzy Logic and guided smoothing for edge detection offers a powerful framework for robust, adaptive, and noise-resilient edge detection algorithms with broad applicability across various domains.

4.3 Guided Image Smoothing

The guided filter(GF) performs a filtering operation by examining the contents of the guidance image. The guidance image may be taken as the same input image or some other identical image. In guided image filtering, the output image is a linear transformation of guided image '*G*' and is expressed as (Kaming, Sun, & Tang, 2013)

$$\hat{I}_n = a_i G_n + b_i \qquad \forall n\ belongs\ to\ w_i \text{.....(4.1)}$$

a_i and b_i are the coefficients in window w_i. The guided image filtering process can be devised as the minimization of difference in input and output images as expressed in (4.2, where ε is the smoothness parameter and decides the degree of smoothness.

$$E(a_k, b_k) = \sum_{i \epsilon w_k}((a_k I_i + b_k - p_i)^2 + \epsilon\ a_k^2) \text{.....(4.2)}$$

The coefficients a_k and b_k are evaluated through linear regression as follows:

$$a_k = \frac{\frac{1}{|w|}\sum_{i \epsilon w_k} I_i p_i - \mu_k \overline{p_k}}{\sigma_k^2 + \epsilon} \text{..... ....(4.3)}$$

$$b_k = \overline{p_k} - a_k \mu_k$$

Where $\mu_k$and σ_k^2 are mean and variance of guidance image I in w_k. $\overline{p_k}$ is the mean of input image p in the window w_k.$|w|$ denotes the total number of pixels w_i. This local linear model is applied to the entire overlapping windows of the image. Since a pixel i may exist in many windows, As a result, when q_i is calculated in different windows, its value does not remain the same. A straightforward method is to take the average of all potential q_i values. So after getting computed a_k, b_k for all the windows w_k in the image, the filter output is calculated as follows:

$$q_i = \frac{1}{|w|}\sum_{k:i\in w_k}(a_k I_i + b_k) \quad \text{.....(4.4)}$$

$$q_i = \overline{a_\iota} I_i + \overline{b_\iota}$$

Where $\overline{a_\iota} = \frac{1}{|w|}\sum_{k\in w_i} a_k$and $\overline{b_\iota} = \frac{1}{|w|}\sum_{k\in w_i} b_k$

Guided Filtering performs edge-preserving filtering. To understand this, we take the help of the following two cases:

Consider the case in which the guidance image is the same as the input image as in our research work, $I = P$.

- If $\epsilon = 0$, then the solution to Eq. (4.2) is $a_k = 1$ and $b_k = 0$.
- If $\epsilon > 0$, the following two cases may be considered:

Case 1: Flat patch: if the image segment is flat in w_k, then Eq. (4.2) is solved by a_k = 0 and $b_k = \overline{p_k}$

Case 2: "High variance" If the image is significantly changing in the patch w_k, then we let a_k be close to 1 and letting b_k close to 0.

In the case of flat patches, the guided smoothing approximates the value with the average value of the pixels in the window, and in the case of high variance region, it preserves edges by keeping the values of a_k large and b_k close to zero.

4.4 Edge detection based on type-1 Fuzzy Logic and Guided Smoothing

In this section, the proposed work is detailed. The various steps and utility of each one are discussed.

4.4.1 Fuzzy Expert System

Figure 4.1 depicts the basic architecture layout of a fuzzy expert system. In this work, three M_1, $M_2$and M_3 methods are presented. In method M_1, the edge refinement step is not applied. The fuzzy system is directly applied to the input image to detect the edges.

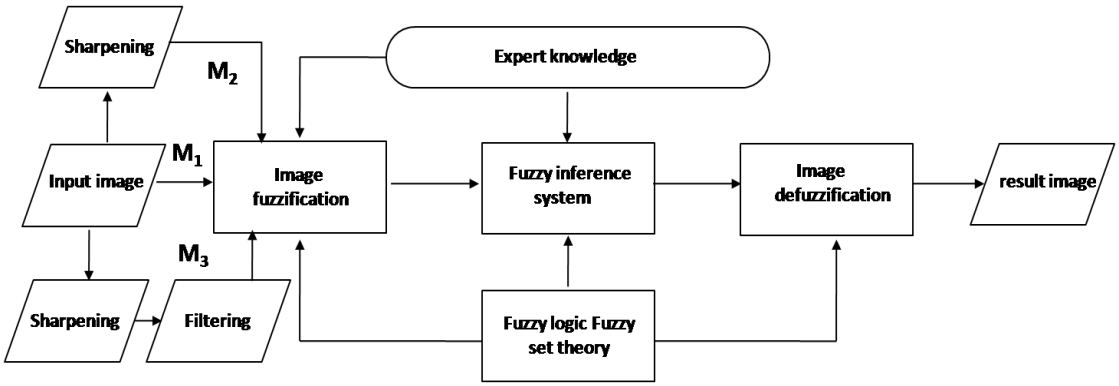


Figure 4.1: Schematic diagram of Proposed Edge Detection mechanism (M_1: Edge detection using fuzzy logic only, M_2: Edge detection using fuzzy logic and sharpening filter and M_3: Edge detection using fuzzy logic and sharpening filter)

In method M_2, the input image is first passed through the sharpening filter, and after this, fuzzy logic is used to detect edges. In method M_3, the input image is first passed through the sharpening filter and then through the Gaussian filter, and finally fuzzy logic is applied to detect edges. While applying fuzzy logic in edge detection, the given image is first fuzzified using the fuzzy input and output membership function. Then IF-ELSE rules are fired where the strength of each rule is calculated using Mamdani Fuzzy-Inference-System (MFIS). At the output, defuzzification is performed to get crisp values and required results. A detailed description of the fuzzy-based system, image sharpening using a guided filter and the Gaussian kernel for noise removal is discussed as follows. The graphical representation of fuzzy logic-based edge detection is shown in Figure 4.2.

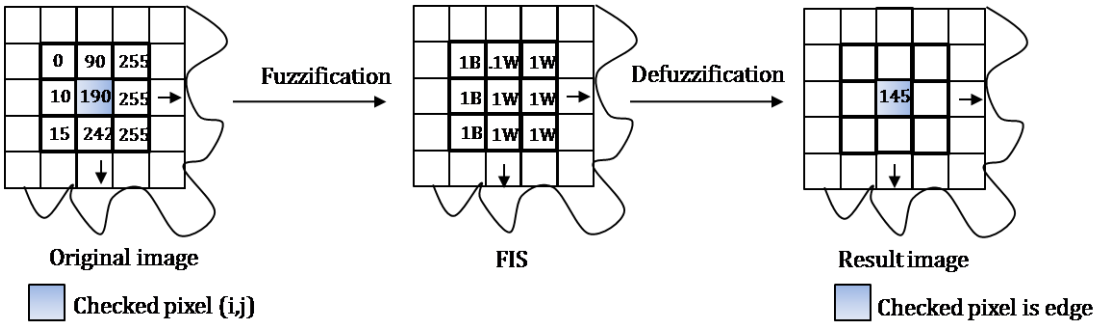


Figure 4.2: Schematic view of fuzzy logic-based edge detection

In this work, the Triangular membership function has been used for input and output,which is described as follows:

$$triangle(P;A,B,C) = \max\left(\min\left(\frac{P-A}{B-A},\frac{C-P}{C-B}\right),0\right)$$

The graphical representation of input and output membership functions is shown in Figure 4.3 and Figure 4.4.

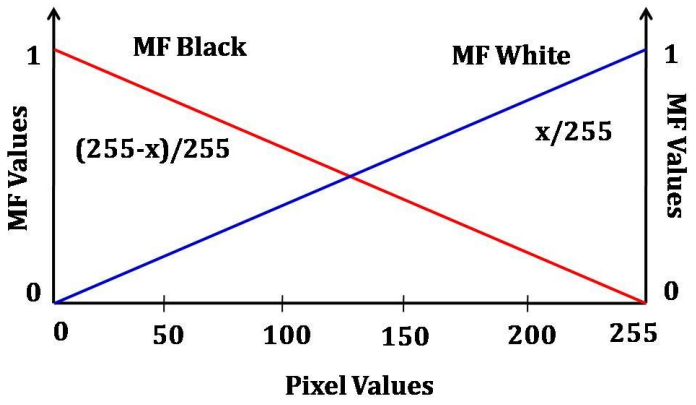


Figure 4.3: Membership functions for both white and black pixels

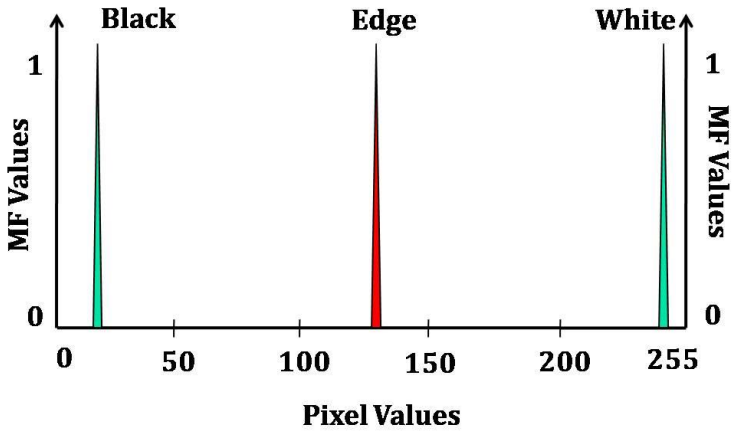
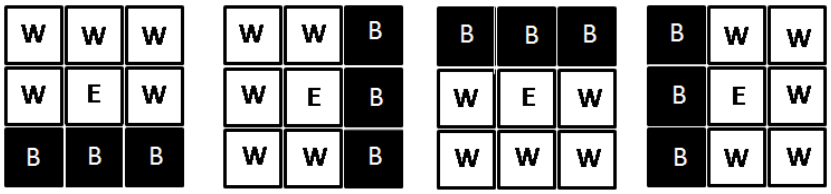


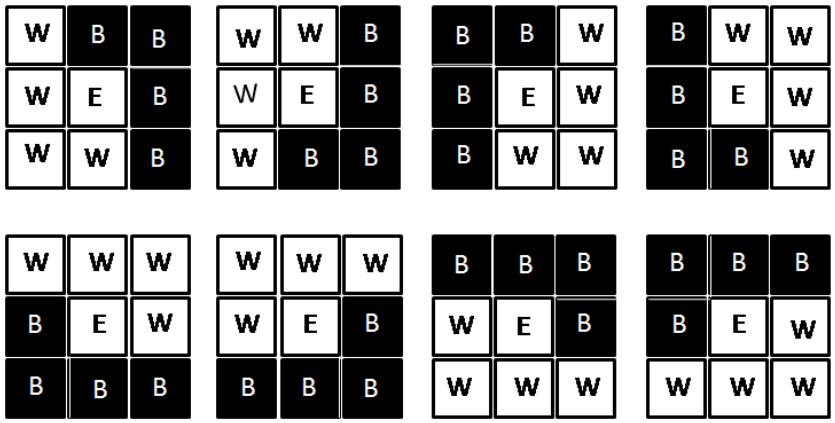
Figure 4.4: Output membership function for black, white and edge

4.4.2 Fuzzy Rules

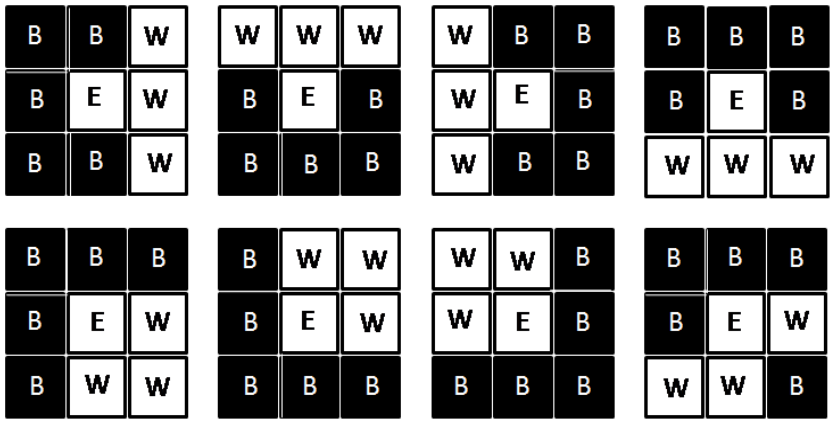
Fuzzy rules are developed using a human expert system, and edge and non-edge pixels are decided based on neighbourhood pixels. For edge detection, a total of 30 rules are defined. The rules are developed according to a window of 3×3, as shown in Figure 4.5. In the rule designing, img(*x*, *y*) represents the pixel position under consideration, '1' represents a white pixel, and '0' represents a black pixel value. Rules are designed to cover each possible scenario. To design fuzzy rules, we look for the number of black and white pixels in the neighbourhood. Four rules are developed for five white and three black pixels. Eight rules are designed for four white and four black pixels in the neighbourhood. Another eight rules are designed for three white and five black pixels and six black and two white pixels. Finally, two rules are designed for all white or black pixels. The graphical representation of the rules designed is given below:



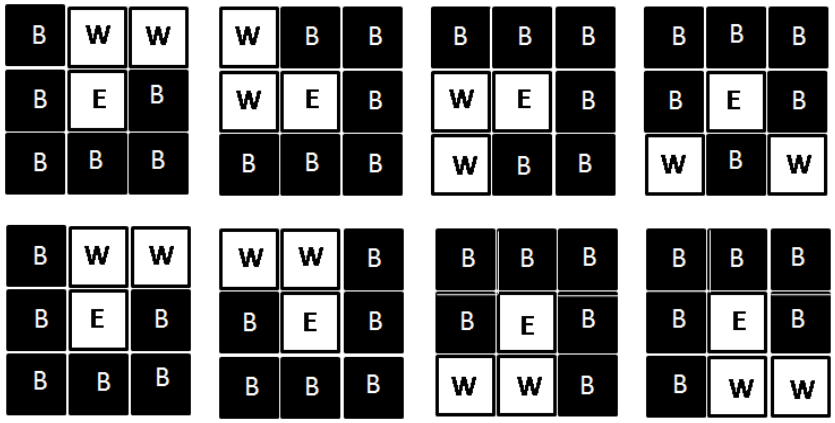
a) Rules with five white and three black pixels in the neighbourhood



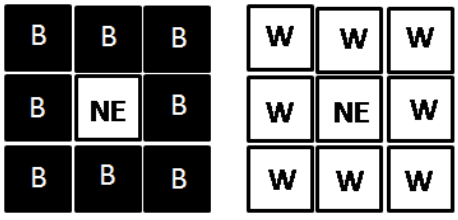
b) Rules with fourwhite and fourblack pixels in the neighbourhood



c) Rules with three white and five black pixels in the neighbourhood



d) Rules with two white and six black pixels in the neighbourhood



e) Rules with all white or black pixels in the neighbourhood

Figure 4.5: Fuzzy rulesforwhite (W), black (B), Edge(E) and non-edge(NE) pixels

4.4.3 Guided Image Filtering and Sharpening

In (Kaming, Sun, & Tang, 2013), it is proved that Guided Filtering(GF) can also be expressed as:

$$W_{nm}^{GF}(G) = \frac{1}{|w|^2}\sum_{i(n,m)\in w_i}\left(1 + \frac{(G_n - \overline{G_i})(G_m - \overline{G_i})}{\sigma_i^2 + \epsilon}\right) \text{.....(4.5)}$$

In general, we have:

$$\hat{I}_n = \sum_{m\in w_n} W_{nm}^{GF}(G) I_m \text{..... ....(4.6)}$$

The enhanced image can be written as:

$$\hat{I}_s = \gamma\left(\hat{I}_n - G_n\right) + G_n \text{.....(4.7)}$$

Plugging Eq. (4.1) into Eq. (4.7), we get,

$$\hat{I}_s = \gamma[a_i G_n + b_i - G_n] + G_n \text{..... .....(4.8)}$$

After simplification, we get,

$$\hat{I}_s = [\gamma(a_i - 1) + 1]G_n + kb_i \text{.....(4.9)}$$

The Gaussian 5×5 kernel is applied as a filter for removing noise, with the kernel as:

$$\frac{1}{159}\begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix}.$$

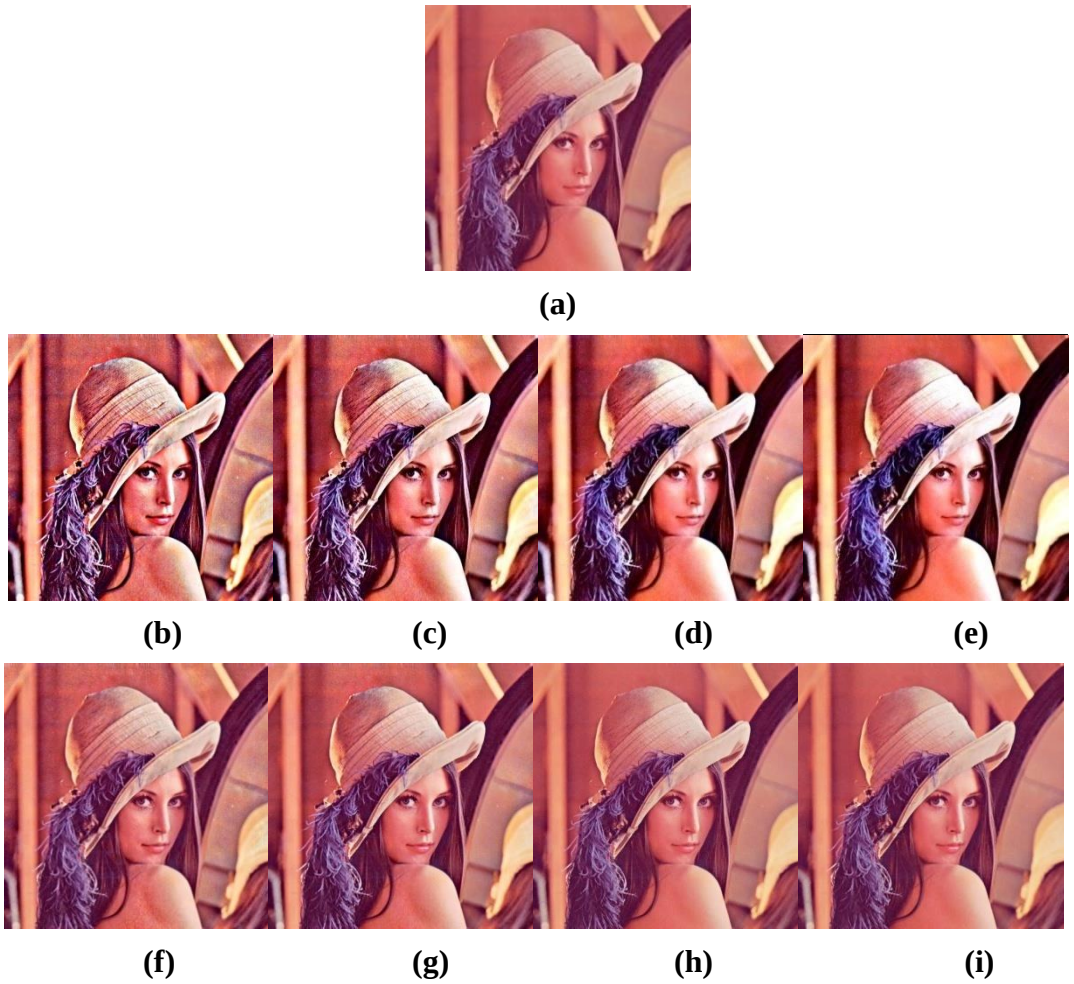
The Gaussian kernel is used in method M_3 only.

4.4.4 Results and Discussions

This section presents the edge detection results obtained using the proposed method.

In Figure 4.6(a), the original Lena image is shown while varying the patch radius from r= 16, 32, 64 and 128 while keeping ε to a fixed level of 0.01 and γ equals five. The obtained results are shown in Figure 4.6(b) to Figure 4.6(e). The local patch size is considered to be $(2r+1)\times(2r+1)$. For smaller radius, the block size is smaller, and sharpening takes place in local patches; therefore, edges become sharper, but

discontinuity points increase; thus, sharpness quality is not good.



(a)

(b) **(c)** **(d)** **(e)**

(f) **(g)** **(h)** **(i)**

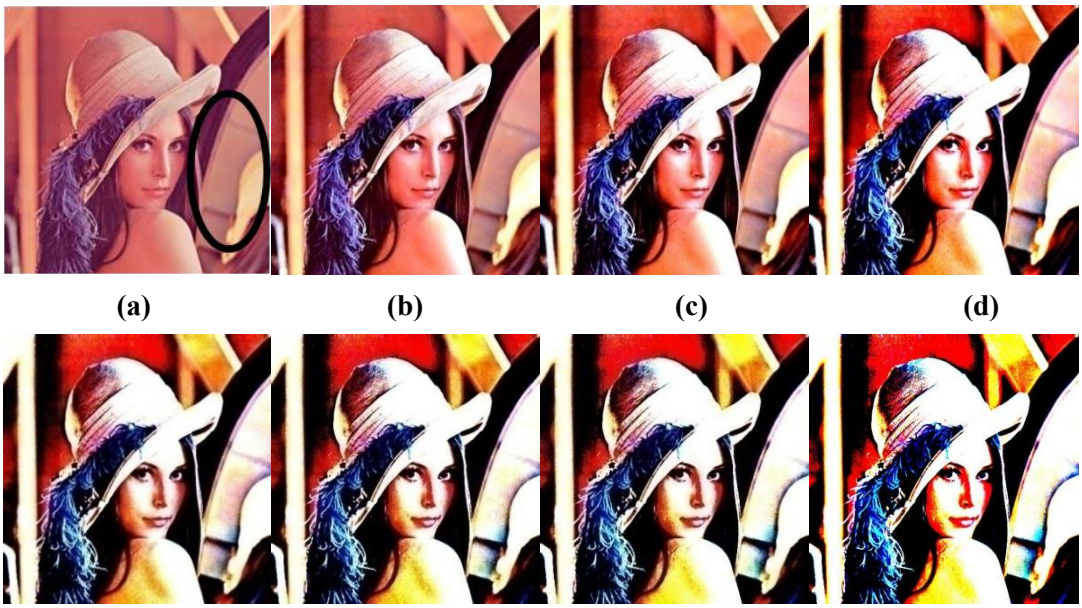
(a) Original Image **(b)** r=16,ε=0.01 **(c)**r=32,ε=0.01 (**d**)r=64,ε=0.01 **(e)**r=128,ε=0.01 **(f)**r=16,ε=0.001 **(g)**r=32,ε=0.001 **(h)**r=64,ε=0.001 **(i)**r=128,ε=0.001

Figure 4.6: Image Sharpening using Guided Filtering (γ=5)

As we increase the radius, the sharpness in the image is relatively more uniform, and for r=64 and r=128, the sharpness in images is of good quality. The above experiment is repeated with ε=0.001 while keeping other parameters fixed; the effect of the regularization parameter can be observed. The mean-variance of the Lena image under consideration is 0.0256; therefore, the impact of the regularization parameter (ε) is negligible, and image quality does not alter significantly.

In Figure 4.7, image enhancement for various values of γ is shown; here, as γ increases, the sharpness increases, but for larger values of γ, the image quality changes considerably compared to the original image. Moreover, a significant

variation in the colours is also observed. It can be observed (a marked region in the original image) from the figures that the edge becomes sharper with the rise in γ. After that, they start to diminish. Therefore it can also be concluded that both ε and γ significantly affect the overall quality of the sharpened images.



(a) Original Lena Image **(b)** r=64,ε=0.01, γ =5 **(c)** r=64,ε=0.01, γ =10 **(d)** r=64,ε=0.01, γ =20
(e) r=64,ε=0.01, γ =30 **(f)** r=64,ε=0.01, γ =50 **(g)** r=64,ε=0.01, γ =75 **(h)** r=64,ε=0.01, γ =100

Figure 4.7: Image Sharpening using Guided Filtering (γ varying)

In Figure 4.8, results are presented for methods M_1 and M_2. Figure 4.8(a) edge detected image using Fuzzy logic is shown for the Lena image. In Figure 4.8(b) to Figure 4.8(h), results are shown for various values of γ, as in Figure 4.10. Considering the mark region in Figure 4.7(a), it is clear from Figure 4.8(a) that the Fuzzy logic method fails to capture edges in this region. Similarly, around the hat area, noise is also added. As we increase the value of γ from 5 to 100, the noise around the hat area diminishes, but the noise rises just above the marked area. These changes are shown in Figure 4.9.

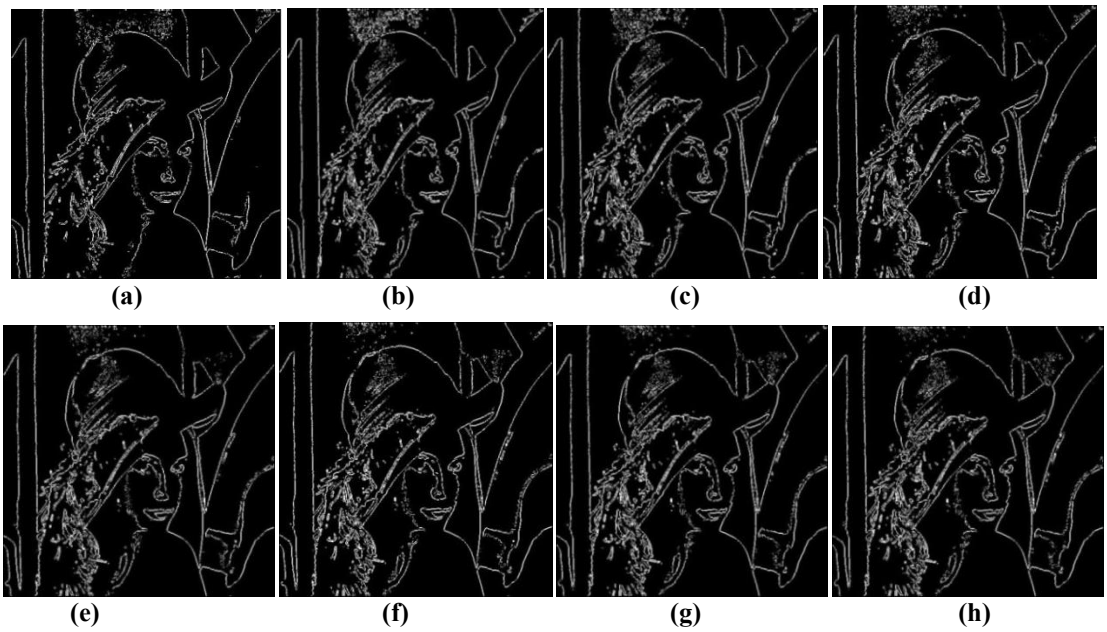


Figure 4.8: Edge detected images (a) Fuzzy logic (b) M_2, γ=5 (c) M_2, γ=10 (d) M_2, γ=20 (e) M_2, γ=30 (f) M_2, γ=50 (g) M_2, γ=75 (h) M_2, γ=100

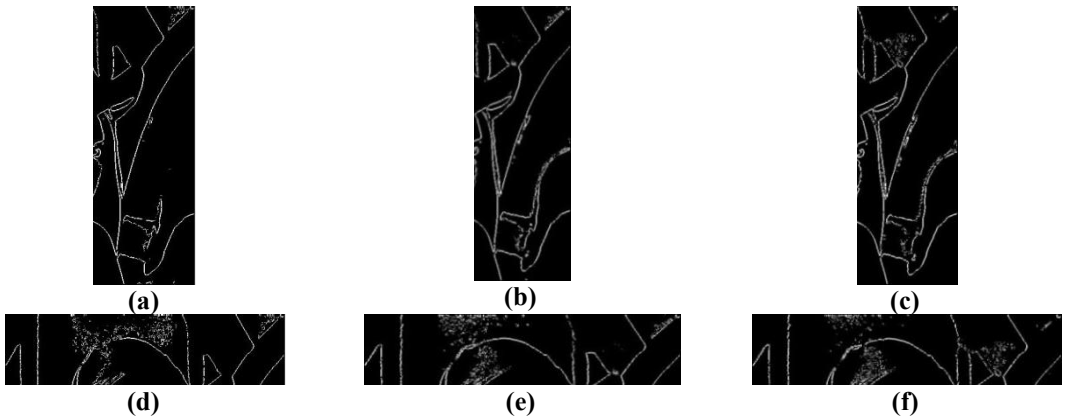


Figure 4.9: Image segment with visible variations

In Figure 4.10, results are presented for method M_3. It is clear from the figure that filtering noise is suppressed significantly, and detected edges are much better than shown in Figure 4.8.

(a) **(b)** **(c)** **(d)**

Figure 4.10: Edge detected images (a) M_3, γ=5 (b) M_3, γ=10 (c) M_3, γ=50 (d) M_3, γ=100

The discussion made above is based on human visualization. However, shifting in pixel positions cannot be identified using a human vision system. Moreover, the regularization term also leads to the shifting of pixels. Therefore, distance-based parameters for quality measurements provide the correctness of the chosen methods. In our results, we have not done any further processing on the detected image to visualize the effect of each method.

In Table 4.1, the listing of parameters results is shown. FoM is least for the Canny edge detection with the value of 0.297, and it is best for γ=5, which equals 0.483. SSIM for the Canny method is 0.451, and for γ=20, it equals 0.560, while HoD is best for γ=50 and equals 4.44/7.

Table 4.1: Comparison of parameters for methods M_1 and M_2(Bold values shows the best result obtained for a performance metric)

Methods	Performance Metrics						
	FoM	**SSIM**	**HoD (Avg/Max)**	E_D	**BDM**	D_K	**P**
Canny	0.297	0.451	6.69/9.21	**0.0048**	10.74	**0.06**	0.225
Fuzzy	0.398	0.519	5.31/8.12	0.0059	12.24	0.08	0.146
γ=5	**0.483**	0.559	4.67/7	0.0056	**9.40**	0.111	0.21
γ=10	0.472	0.557	4.59/6.78	0.0056	9.48	0.113	0.218
γ=20	0.469	**0.560**	4.47/6.86	0.0055	10.03	0.11	0.229
γ=30	0.467	0.559	4.45/7.42	0.0055	10.02	0.112	**0.232**
γ=50	0.471	0.556	**4.44/7**	0.0055	9.98	0.102	**0.232**
γ=75	0.467	0.557	4.46/7.42	0.0055	9.85	0.102	0.231
γ=100	0.469	0.555	4.47/7.21	0.0055	9.93	0.106	0.231

The Euclidian distance is minimum for the Canny method, which is an expected result as a large number of edges are detected by the Canny method. BDM is least for γ=5, while it is at maximum for the Fuzzy method. D_K is best for the Canny method. Finally, the co-relation co-efficient is best for γ=30, 50. As a result of the table, it can be deduced that the Fuzzy method's performance alone is the worst and among the chosen parameters for γ=5; the obtained results are better than other considered methods.

In Table 4.2, results are compared for methods M_1 and M_3. Obtained results have shown similar trend but obtain results are much better in comparison to the results in Table 4.1. In Table 4.2, FoM is maximum for γ = 20, SSIM is maximum for γ = 30, 50. HoD is maximum for γ = 50. Still these performance measures are indicative, and do not provide clear information about exact edge detection mechanism. Therefore, in most of the recent research (Begol & Maghooli, 2011) (Mehrara, Zahedinejad, & Pourmohammad, 2009) (Zhang, Xiao, Ma, & Song, 2009) (Mathur & Ahlawat, June-July 2008) (Marmanis, Schindler, Wegner, Galliani, Datcu, & Stilla, 2018) (Farbod, Akbarizadeh, Kosarian, & Rangzan, 2018) (Setayesh, Mengjie, & Johnston, 2009) (Yirenkyi & Appati, 2016) human visual system (HVS) is used to characterize edge and non-edge pixels.

Table 4.2: Comparison of parameters for methods M_1 and M_3 (Bold values shows the best result obtained for a performance metric)

Methods	Performance Metrics						
	FoM	**SSIM**	**HoD (Avg/Max)**	E_D	**BDM**	D_K	**P**
Canny	0.297	0.451	6.69/9.21	**0.0048**	10.74	**0.06**	**0.225**
Fuzzy	0.398	0.519	5.31/8.12	0.0059	12.24	0.08	0.146
γ=5	0.470	0.581	4.15/10.44	0.0056	9.92	0.120	0.186
γ=10	0.475	0.582	4.17/9.64	0.0056	9.69	0.110	0.198
γ=20	**0.497**	0.590	4.13/8.43	0.0055	9.05	0.093	0.211
γ=30	0.496	**0.595**	4.09/7.09	0.0055	8.81	0.088	0.217
γ=50	0.495	**0.595**	**4..09/6.86**	0.0057	8.59	0.079	0.216
γ=75	0.489	0.593	4.11/6.63	0.0057	8.65	0.087	0.217
γ=100	0.493	0.593	4.12/6.56	0.0057	**8.57**	0.081	0.219

The findings, as mentioned above, were produced using the Lena image, and the identified edges were compared to the ground truth image, with the results being compared using a

variety of performance metrics. However, to prove the usefulness of the proposed method, results are obtained from other database images. The image shown in Figure 4.11 is taken from Berkley Segmentation Database (Berkeley Segmentation Dataset: Images, 2003), while in image in Figure 4.12 and is from USC-SIPI Image Database (The USC-SIPI Image Database). Considering, Figure 4.11, the fuzzy logic-based method fails to detect the edge on the west-south corner of the image. However, from (b) to (e) in Figure 4.11, it is clear that the edges can be detected more correctly as we increase sharpness. From Figure 4.12, it can be visualized that as sharpness increases, image detail is more visible. The same effect is also obtained in detected edges.

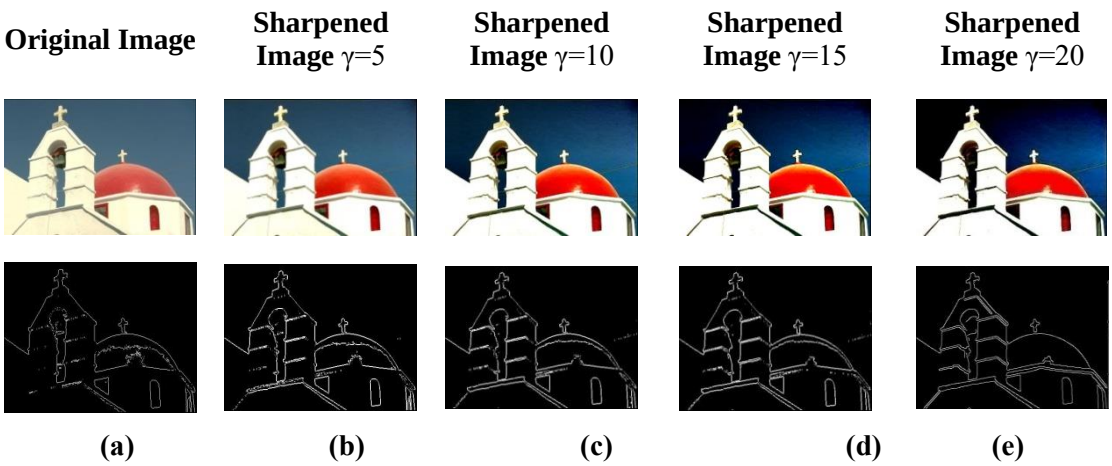


Figure 4.11: BSD edge detected images (a) Fuzzy logic (b) M_3, γ=5 (c) M_3, γ=10 (d) M_3, γ=15 (e) M_3, γ=20

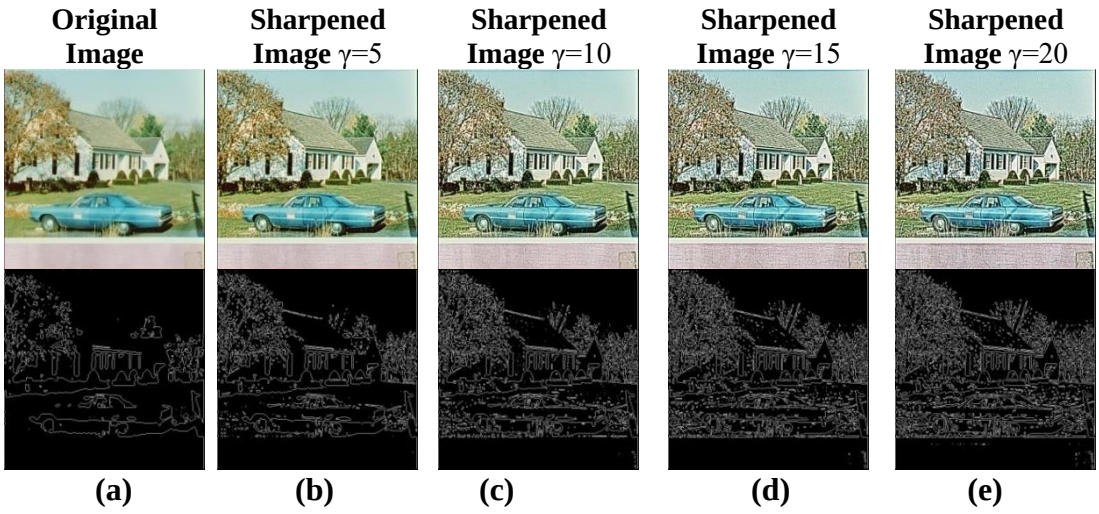


Figure 4.12: USC-SIPI edge detected images (a) Fuzzy logic (b) M_3, γ=5 (c) M_3, γ=10 (d) M_3, γ=15 (e) M_3, γ=20

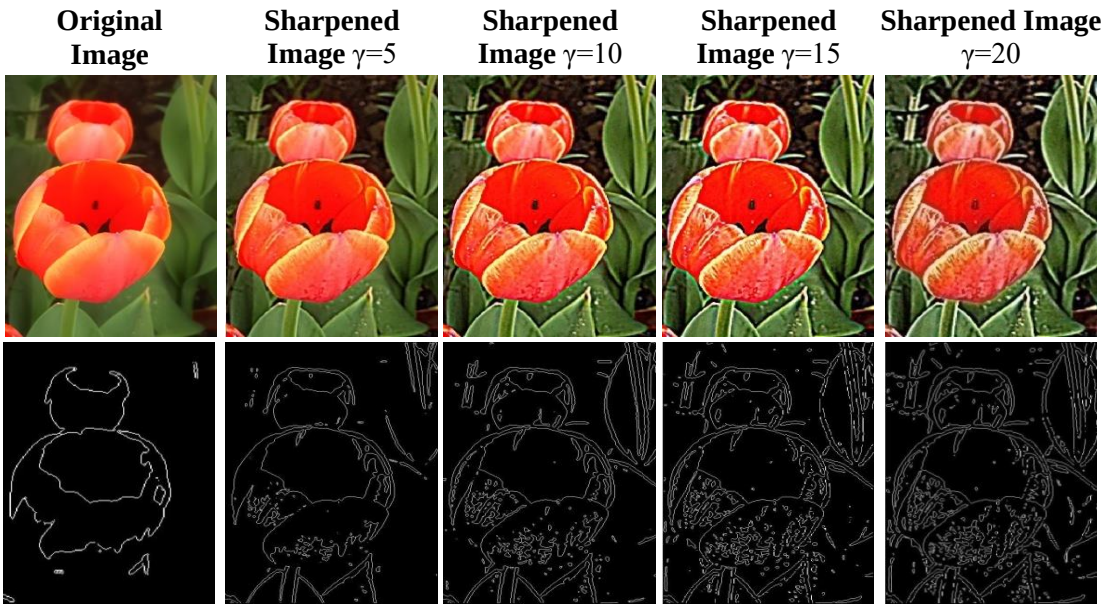


Figure 4.13: Tulip edge detected images (a) Fuzzy logic (b) M_3, γ=5 (c) M_3, γ=10 (d) M_3, γ=15 (e) M_3, γ=20

In Figure 4.13, results for tulip images and obtained response show a similar trend as in Figure 4.11 and Figure 4.12. It is also clear from Figure 4.12 and Figure 4.13 sometimes more sharpness leads to the generation of noise; therefore image should be sharpened to a level where the effect of noise is minimal. However, if this noise is dominant, then this additional noise can be suppressed using filters as used in other edge detection methods (Maini & Aggarwal, 2009).

In the above-discussed results, it is found that the Fuzzy Logic based method fails to detect some of the edges. Therefore, to check the validity of the considered fuzzy logic structure, in Figure 4.14, results are generated for two images, Figure 4.14(a) and Figure 4.14(c), where edges are visible, and it is found that the considered fuzzy logic structure correctly detect all the edges in both the images. Thus, in the case of clear edges images, fuzzy logic structure gives accurate results.

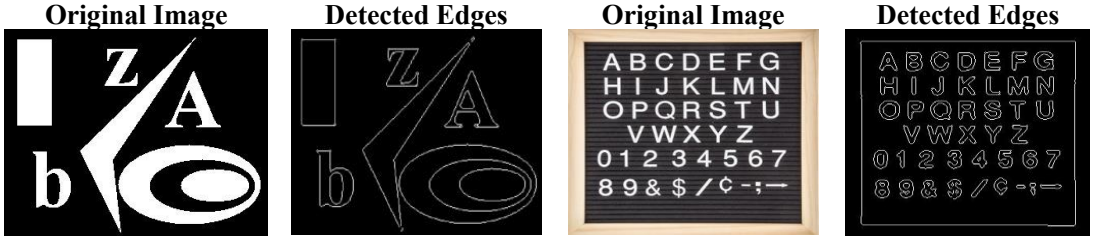


Figure 4.14: Edge Detection using Fuzzy Logic (a) Original Image (b) Detected Edges (c) Original Image (d) Detected Edges

In Figure 4.15, an animal alphabet image is taken, which has more complex edges than those considered in Figure 4.14. The fuzzy logic-based method fails to detect 'Tiger' and 'Orangutan' shapes, and the animals' letter marks are not detected. As we increase sharpness, edges and letters are detected more clearly.

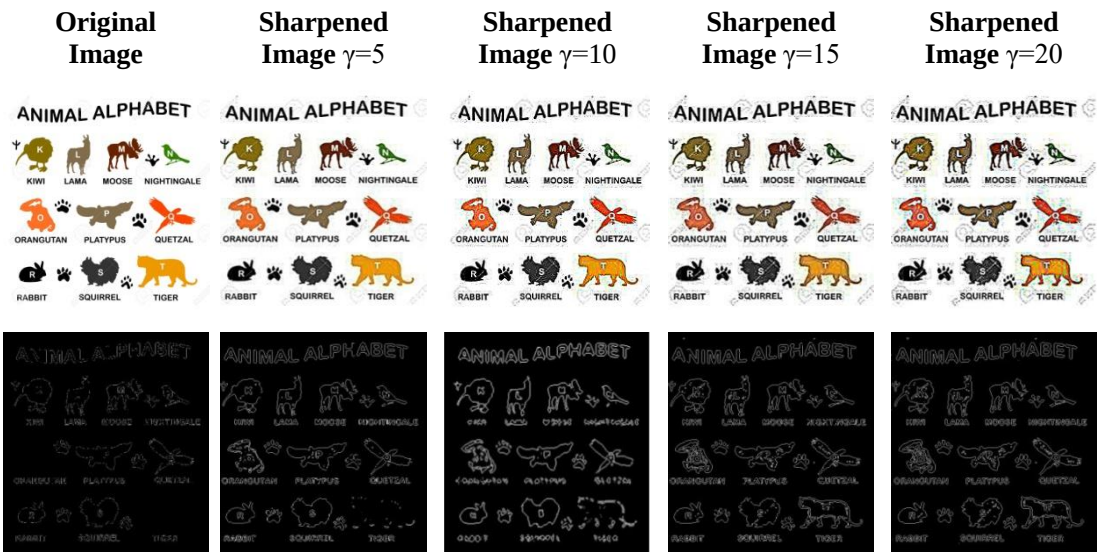


Figure 4.15: Animal alphabet image (a) Fuzzy logic (b) M_3, γ=5 (c) M_3, γ=10 (d) M_3, γ=15 (e) M_3, γ=20

In Figure 4.16, zoomed version of fuzzy logic and sharpened image γ=20 is shown; the animal name is not detected with the Fuzzy design, while with the proposed method, the edges of both the animal name and letter mark on the animals are detected. The claws and paws edges are also clearly visible.

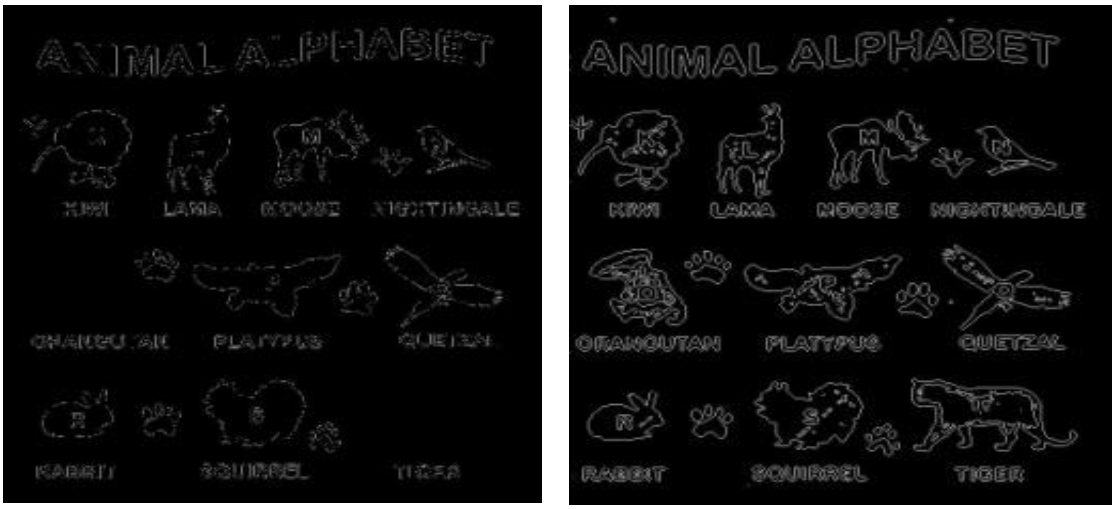


Figure 4.16: Zoomed version of (a) Figure 4.15(a) **(b)** Figure 4.15(e)

Based on the comprehensive analysis and evaluation conducted in this study, it can be concluded that the proposed edge detection method, denoted as M_3, exhibits superiority over the fuzzy logic-based method. Despite leveraging the same membership function as previous works, M_3 surpasses the fuzzy logic approach and effectively addresses its limitations.

The superiority of method M_3 is attributed to several key factors:

1. **Enhanced Performance:** Method M_3 demonstrates superior performance in terms of accuracy, robustness, and consistency compared to the fuzzy logic-based method. It effectively overcomes the shortcomings of fuzzy logic structures, such as ambiguity in rule interpretation and limited adaptability to complex image scenarios.
2. **Innovative Approach:** M_3 introduces innovative techniques or algorithms that leverage advanced methodologies beyond traditional fuzzy logic. These innovations may include guided image filtering, ant colony optimization, or other novel strategies tailored to the specific challenges of edge detection tasks.
3. **Efficient Utilization of Resources:** Method M_3 optimally utilizes computational resources, ensuring efficient processing and reduced computational overhead compared to the fuzzy logic-based approach. This efficiency translates into faster execution times and improved scalability for large-scale image processing tasks.
4. **Robustness to Variability:** M_3 demonstrates robustness to variations in image content, lighting conditions, and noise levels, making it suitable for diverse real-world applications. Its adaptive nature allows it to maintain high performance across a wide range of imaging scenarios without requiring extensive parameter tuning.

While method M_3 emerges as the preferred choice for edge detection in this study, it is acknowledged that further research is warranted to continually improve the effectiveness of fuzzy logic-based methods. Future investigations could focus on developing more efficient fuzzy rule sets based on human expert knowledge and

refining the selection of membership functions to better capture the underlying relationships in the image data.

Additionally, advancements in machine learning techniques, such as fuzzy logic systems combined with neural networks or evolutionary algorithms, may offer promising avenues for enhancing the capabilities of fuzzy logic-based edge detection methods. By integrating these complementary approaches, researchers can strive to achieve even greater accuracy, flexibility, and adaptability in edge detection tasks, ultimately advancing the state-of-the-art in image processing and computer vision.

4.5 Summary

This work proposed a fuzzy logic, sharpening, and filtering-based edge detection approach.The main aim of thework is to design an edge detection method where edgescan be controlled without deteriorating the considered image. It has been found that using the fuzzy logic method alone is not suitable for edge detection as it incorrectly rejects some of the true edges, and some noise pixels may also be classified as edge. We demonstrated that image sharpening might be done to overcome this problem, which considerably improves the results. It is also shown that sharpening depends on parameters r, ε and γ, and these parameters should be chosen efficiently to get desired results. It is also notable that the regularization parameter (ε) should be kept within the sub-range of image variance as the regularization parameter may shift the pixel positions of edge pixels. It is also shown that noise generated due to the fuzzy process can be significantly brought down by using a Gaussian filter. The obtained results are compared using various statistical measures, and it has been found that proposed methods M_2 and M_3 perform better in comparison to method M_1.

Chapter Five

Edge Detection in Digital Images Using Guided L_0 Smoothen Filter and Fuzzy Logic

5.1 Introduction

Smoothing an image may change the number of pixels detected as edges, as smoothing reduces amplitude variation; therefore, true edges can also be diminished, which is not actually desired. Therefore edge-preservation alongwith smoothing is required. In the spirit of edge-preserving smoothening, various methods have been proposed over the period. In a similar context, L_0 smoothing filter is proposed. In this method, prominent edges are preserved by increasing the steepness of transition and diminishing the other low-magnitude edges, still maintaining the overall structure of an image. After smoothing the image,a fuzzy logic-based edge detection method is applied. To better understand the Guided L_0 smoothen filter, first the L_0 smoothing filter is discussed; after that, L_0 gradient minimization is explained, which is used to sharpen the dominant edges. Finally, Guided L_0 smoothen filter is discussed, which has the edge-preserving smoothing property. Therefore, Guided L_0 smoothen filter takes advantage of L_0 gradient minimization and guided filtering.

5.2 Novelty of the Proposed Method

5.2.1 Guided L_0 Smoothen Filter: Unlike conventional smoothing filters, the guided L_0 smoothen filter preserves edges and fine details while effectively reducing noise. Its adaptive nature ensures that the filtering strength is adjusted based on local image structures, resulting in improved edge preservation.

- Preserves edges and fine details while reducing noise.
- Adaptive nature adjusts filtering strength based on local image structures.
- Improved edge preservation compared to conventional smoothing filters.

5.2.2 Integration of Fuzzy Logic: The incorporation of fuzzy logic enables the algorithm to handle uncertainties and imprecisions inherent in image data. By modeling the ambiguity of edge boundaries, fuzzy logic enhances the robustness of edge detection, particularly in challenging conditions such as low contrast or high noise.

- Handles uncertainties and imprecisions in image data.
- Models ambiguity of edge boundaries.
- Enhances robustness of edge detection, especially in challenging conditions like low contrast or high noise.

5.2.3 Benefits and Advantages

1. **Enhanced Edge Preservation:** The guided L_0 smoothen filter maintains the sharpness of edges while effectively reducing noise, resulting in edge maps with improved clarity and accuracy.
2. **Robustness to Variations:** By leveraging fuzzy logic, the method can adapt to variations in contrast, lighting conditions, and noise levels, ensuring consistent performance across diverse imaging scenarios.
3. **Reduced False Positives:** The combined approach of guided filtering and fuzzy logic helps minimize false detections of edges, leading to more reliable results suitable for downstream analysis tasks.
4. **Computational Efficiency:** Despite its sophistication, the proposed method remains computationally efficient, making it suitable for real-time applications and large-scale image processing tasks.

5.2.4 Applications

1. **Medical Imaging:** The precise delineation of anatomical structures is critical in medical imaging for diagnosis and treatment planning. The proposed method can aid in segmenting organs and tissues from medical images with high accuracy.

2. **Computer Vision:** Edge detection serves as a fundamental step in various computer vision tasks, including object recognition, scene understanding, and motion analysis. The proposed method can improve the quality of edge maps, thereby enhancing the performance of downstream vision algorithms.
3. **Remote Sensing:** In satellite and aerial imagery analysis, accurate edge detection is essential for mapping land cover, monitoring environmental changes, and detecting anomalies. The proposed method can contribute to more reliable interpretation of remote sensing data.
4. **Industrial Inspection:** Edge detection plays a vital role in quality control and defect detection in manufacturing processes. The proposed method can help identify product defects and irregularities with greater precision, improving overall production efficiency.

In summary, the integration of guided L_0 smoothen filter and fuzzy logic in edge detection offers a novel approach with significant benefits in terms of accuracy, robustness, and efficiency, making it well-suited for a wide range of applications across various domains.

5.3 L_0 smoothing filter

This section discusses the basic concept of the L_0 smoothing filter (Xu, Lu, Xu, & Jia, 2011). In the gradient-based method,intensity changes is measured between neighbouring pixels. While in L_0 smoothing filter, L_0 norm is calculated, which may be defined as the number of non-zero differences in intensity values and can be mathematically defined as:

$$c(I) = N\{i, j | |I_{i,j} - I_{i-1,j}| \neq 0\} \quad (5.1)$$

Where *i, j*and $(i - 1)$, *j* represents neighbouring samples (or pixels) indices. $|I_{i,j} - I_{i-1,j}|$ is the forward difference of the intensity, also known as gradient w.r.t. *i*. The parameter $N\{\}$ is used to represent the counting operator, which counts the number of *i* that satisfies $|I_{i,j} - I_{i-1,j}| \neq 0$. This is the L_0 norm of the gradient. c(I) does not consider gradient magnitude; it calculates the number of pixels with non-zero

differences. After calculating L_0, we define the objective function as input image g and output image *I* should be structurally similar. The objective function is defined as follows:

$$\min_I \sum_i (I_i - g_i)^2 \quad such\ that\ c(I) = \alpha \text{.....(5.2)}$$

The number of the non-zero gradient is given by $c(I) = \alpha$, the discrete input signal is denoted by variable *g*, and its smoothed version is represented by the variable *I*. This equation can be minimized through an exhaustive search. The output image will flat out low amplitude edges and increase the steepness of the transition. The number of significant edges in the output image will be given by *c(I)*. The term $(I_i - g_i)^2$ will not allow pixels to drastically change their colour. No edge blurriness will occur in this method as local averaging and filtering operations are not performed. In practice, the value of α may vary from hundreds to thousands in 2-D images depending upon their resolutions. To make a balance between result similarity and structure flattening, a more appropriate objective function which represents the constraint optimization would be:

$$\min_I \sum_i (I_i - g_i)^2 + \lambda c(I) \text{..... ...(5.3)}$$

This optimization is necessary to maintain the image structure as the value of α may be very large. The parameter λ will control the significance of *c(I)*. The parameter λ is very important, as it exploits the sparsity of the image gradient and, consequently, the smoothness of the output image. The parameter λ can be seen as a smoothing parameter.

5.4 L_0 gradient minimization

The L_0 norm could be optimised using L_0 gradient minimization to produce a piecewise steady output image (Xu, Lu, Xu, & Jia, 2011). It's great for boosting the steepness of transitions and sharpening strong edges. As a result, the following minimization problem must be resolved.

$$\min_I \|I - I^*\|_2^2 + \lambda \|\nabla I\|_0 \text{.....(5.4)}$$

Where *I* is the output image ∇I denotes the gradients of the image *I*, the notation I^*

represents the observed smoothed image, and λ is a scaling factor to control the gradient. With the end purpose of overcoming the issue of the discontinuity of the term $\|\nabla I\|_0$, auxiliary variable Ω is introduced to handle ∇I. Therefore Eq. (5.4) can be converted into the following minimization problem:

$$\min_I\{\|I - I^*\|_2^2 + \beta\|\Omega - \nabla I\|_0\} + \lambda\|\Omega\|_0 \quad \text{.....(5.5)}$$

Where β controls the similarity between ΔI and Ω, the degree of smoothness is handled by λ.

5.5 Guided L_0 smoothing filter

L_0 gradient minimization was introduced by Xu et al., which sharpens the image while keeping dominating edges (Xu, Lu, Xu, & Jia, 2011). X. Ding et al. later propose a guided L_0 smoothing filter. It takes benefits of the properties of both L_0 gradient minimization and guided filter. This method is known as a guided L_0 smoothing filter (Ding, Chen, Zheng, Huang, & Zeng, 2016).

To begin, Using, I^k, the parameter Ω^k is optimized using:

$$\min_{\delta^k} \beta^k\|\nabla I - \Omega^k\|_2^2 + \lambda\|\Omega^k\|_0 \qquad (k = 1,2,3 \dots\dots\dots\dots\dots) \quad \text{.....(5.6)}$$

The given Eq. 5.6 can be analyzed by following the steps outlined in (Seng, Samad, & Nor., 2019).

$$\Omega^k = \begin{cases} 0 & \nabla I^k \le \frac{\lambda}{\beta} \\ \nabla I^k & others \end{cases} \quad \text{.....(5.7)}$$

Now both Ω^k and I^k are known, we now evaluate I^{k+1} using Eq. 5.5 and Eq. 5.7:

$$\min_{I^{k+1}}\|I^{k+1} - I^*\|_2^2 + \beta^k\|\Omega^k - \nabla I^{k+1}\|_2^2 \quad \text{.....(5.8)}$$

Considering Eq. .....(5.7), Eq. .....(5.8) can be re-written as Eq. (5.9):

$$\left\{ \begin{matrix} min_{I^{k+1}}\|I^{k+1} - I^*\|_2^2 + \beta^k\|\nabla I^{k+1} - H.*\nabla I^k\|_2^2 \\ H = \begin{cases} 0 & \Omega^k = 0 \\ 1 & \Omega^k \neq 0 \end{cases} \qquad (k = 1,2,3 \dots\dots) \end{matrix} \right\} \quad \text{.....(5.9)}$$

The objective function in Eq. (5.9) is quadratic. Therefore, it has a convex optimization issue. Thus, the least square technique and Fourier transformation are used to solve it (Ding, Chen, Zheng, Huang, & Zeng, 2016). The solution of Eq. (5.9) is:

$$I^{k+1} = ifft\left(\frac{fft(I^*)+\beta(fft(\partial_x^T)fft(\Omega_x^t)+fft(\partial_y^T)fft(\Omega_y^k))}{fft(1)+\beta(fft(\partial_x^T)fft(\partial_x)+fft(\partial_y^T)fft(\partial_y))}\right) \quad (5.10)$$

The Fast Fourier transform operator is represented by the parameter *fft*, while its inverse is represented by the value *ifft*. The terms ∂x and ∂y denote horizontal and vertical difference operators, respectively.

Defining new variable 's' for smoothed image, then using, Ω^k and s^k, we obtain s^{k+1} as:

$$\begin{cases} \min_{s^{k+1}} \|s^{k+1} - s^*\|_2^2 + \beta^k \|\nabla s^{k+1} - H.* \nabla s^k\|_2^2 \\ H = \begin{cases} 0 & \Omega^k = 0 \\ 1 & \Omega^k \neq 0 \end{cases} \qquad (k = 1,2,3 \dots \dots) \end{cases} \quad (5.11)$$

The solution of Eq. (5.11) is:

$$s^{k+1} = ifft\left(\frac{fft(s^*)+\beta(fft(\partial_x^T)fft(H.\nabla s_x^k)+fft(\partial_y^T)fft(H.\nabla s_y^k))}{fft(1)+\beta(fft(\partial_x^T)fft(\partial_x)+fft(\partial_y^T)fft(\partial_y))}\right) \quad (5.12)$$

Algorithm

Input: Images s^*, guided image I^*, parameters λ, β_0, β_{max} rate κ

Initialization: $I^1 \leftarrow I^*, s^1 \leftarrow s^*, \beta^1 \leftarrow \beta_0, k \leftarrow 1,$

repeat:

with I^k, solve Ω^k for in (13);

with I^k and Ω^k, solve for I^{k+1} in (14);

with s^k and Ω^k, solve for s^{k+1} in (16);

$\beta \leftarrow \kappa\beta, k++;$

Until $\beta > \beta_{\max}$

Output: *s*.

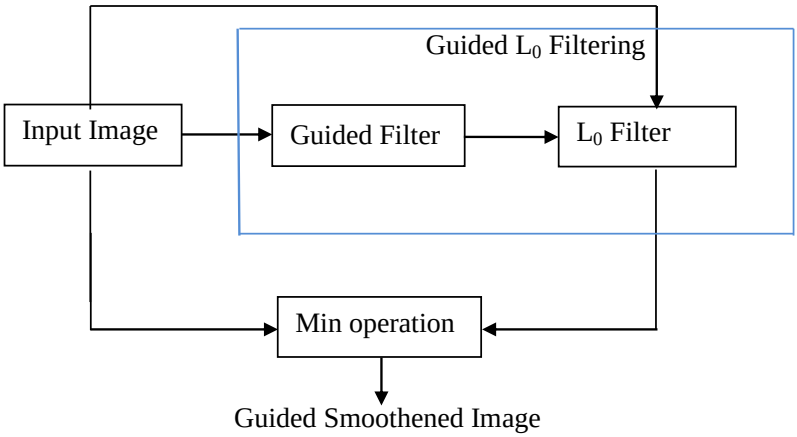


Figure 5.1: Guided L_0 smoothen filter

5.6 Proposed Edge Detection Structure

The proposed architecture is depicted in Figure 5.2, the basic architecture layout of the proposed fuzzy expert system for edge detection. In this framework, three methods, defined as M_1, M_2 and M_3 are presented. In method M_1, fuzzy logic is directly applied to detect edges on the input image. In method M_2, the input image is first passed through the L_0 smoothen filter, and after this, fuzzy logic is used to detect edges. In method M_3, the input image is first passed through the guided L_0 smoothen filter, and then fuzzy logic is applied to detect edges. In Fuzzy logic-based edge detection,the first input image is fuzzified using fuzzy input and output membership function and then IF-THEN-ELSE rules are fired using Mamdani Fuzzy-Inference-System(MFIS). At the output, defuzzification is performed using the centroid method to get crisp values and to achieve desired results further.

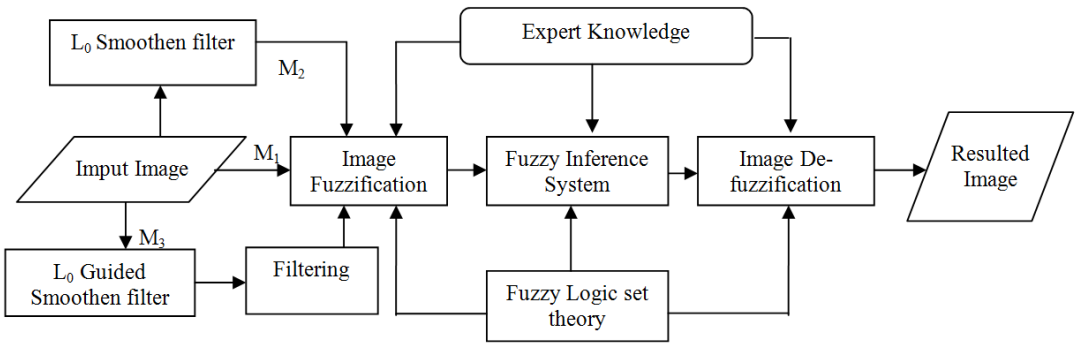


Figure 5.2: Schematic of the proposed edge detection mechanism (M_1: Edge detection using fuzzy logic only, M_2: Edge detection using fuzzy logic and L_0smoothen filter and M_3: Edge detection using fuzzy logic and Guided L_0 smoothen filter)

The fuzzy structure used for edge detection is the same as described in sections 4.4.1 and 4.4.2.

5.7 Results

In this section, the usefulness of the proposed method is discussed.To cover a wide variety of experiments, results are presented using single image considered from Berkley Segmentation Database (BSD) [101]. Finally, the comparative analysis results are presented using the BSD and USC-SIPI Image Database.

Results M_1 method

In Figure 5.3, results for edge detection are shown using fuzzy logic only. It is clear from the figure that in edge-detected images, most edges are correctly detected, with some more edges being falsely detected, especially in Figure 5.3(c), where at corners, falsely detected edges can be easily seen.

(a) **(b)** **(c)**

Figure 5.3: (a) Original Image (b) Ground Truth (c) Detected edges using Fuzzy Logic

Results M_2 method

The falsely detected edges can be suppressed using L_0 smoothen filtering, but it should be kept in mind that more smoothness may lead to false rejection of edges. Therefore degree of smoothness plays an important role in detected edges. The chosen parameters for smoothening are $\beta_0=2\lambda$, $\beta_{max}=100000$, $k=2$, and λ varies (0.05-0.2). In Figure 5.4, on the top row, smoothened images are shown, using various levels of degree of smoothness. As L_0 is an edge-preserving filter, therefore, even in smoothened, prominent edges are preserved. In the bottom row, results for edge detection using fuzzy logic are shown on the smoothened images. It is clear from the figure that as smoothness increases, the falsely detected edges are reduced while prominent edges are still preserved.

(a) λ=0.05 **(b)** λ=0.1 **(c)** λ=0.2

Figure 5.4: Smoothen images using an L_0 smoothing filter

Results M_3 method

In Figure 5.5, on the top row, smoothened images are shown, using various levels of degree of smoothness while considering guided filtering. As guided, L_0 is an edge-preserving filter, prominent edges are preserved even in smoothened images. On the bottom row, edge-detected images are shown. Therefore, the detected edges can be controlled by varying the degree of smoothness and applied methods (M_2 and M_3). This is the main advantage of the proposed method, where detected edges can be controlled.

(a) λ=0.05 **(b)** λ=0.1 **(c)** λ=0.2

Figure 5.5: Detected Edges in smoothen images using a guided L_0 smoothing filter

In Figure 5.6, a comparison of various edge detection methods is shown, whereas in Figure 5.6(a) ground truth image is shown with marked white areas where notable

changes occur in different methods. In the Sobel method, edges in the circular mark region are not properly detected. In the case of the Canny method, numerous edges that were mistakenly discovered.. In the fuzzy edge detection, obtained results are well in agreement with the ground truth image. Still, variation in the rectangular mark region can be seen, along with several more edges in the image have been mistakenly accepted as true. The findings for the method using fuzzy technique along with L_0 smoothen filter are close to the ground truth image and show little fluctuations. Finally, for guided L_0 smoothen filter and fuzzy method, here again results are very much similar to L_0 smoothen filter and fuzzy method with minor inclusion of false edges in circular region, however oval mark section is best detected.

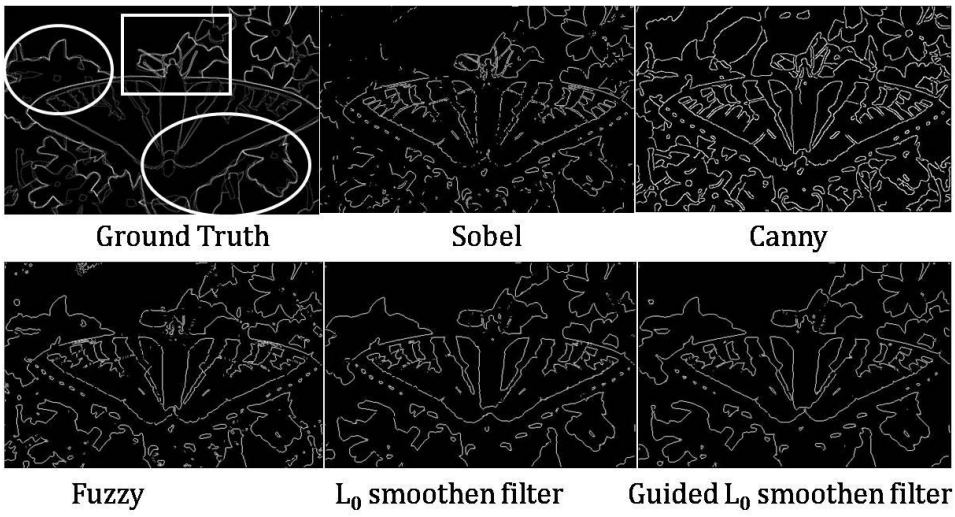


Figure 5.6: Comparison of various edge detection methods

It is clear from the figures that the variation is so small it is difficult to judge from the naked eye; therefore, to evaluate the performance of the method, three performance measures as discussed above are used and obtained results are shown in Figure 5.7 to Figure 5.9.

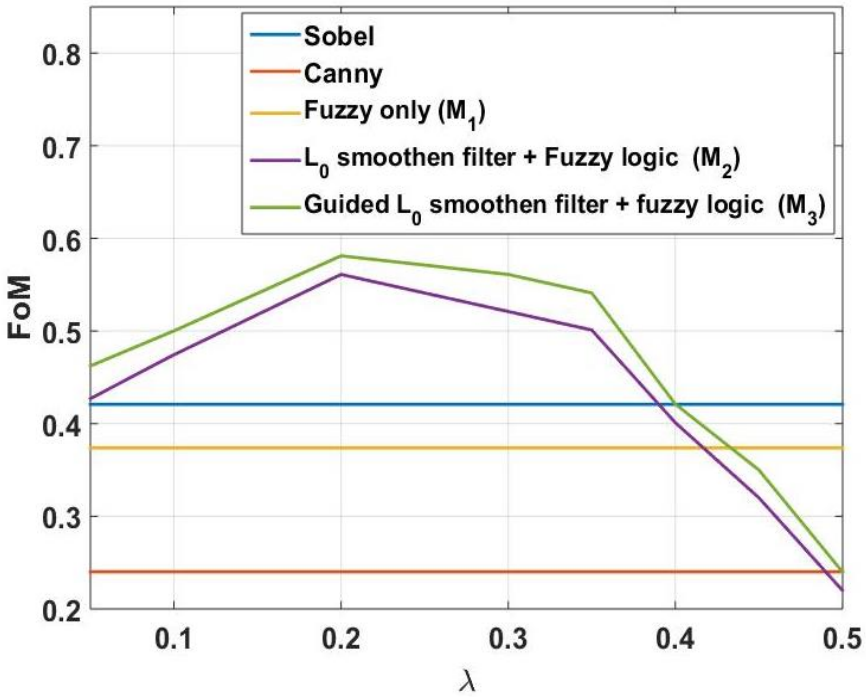


Figure 5.7: Comparison of various edge-detection methods (FoMvs λ)

Figure 5.7 to Figure 5.9 show results for various edge detection techniques under three performance metrics. Ideally, FoM and SSIM should be 1, and HoD should equal 0. It is clear from the tables that FoM is lowest for canny edge detection, i.e., more pixels shift their position compared to other considered techniques. In Figure 5.7, FoM vs λ is shown. It is evident that the best FoM is obtained under the Guided L_0 smoothen filter + fuzzy logic case. It is also observable that as the degree of smoothness increases, FoM increases to a limit; after that, it decreases. The best value of FoM is obtained for λ = 0.2. In Figure 5.8, SSIM vs λ is shown.

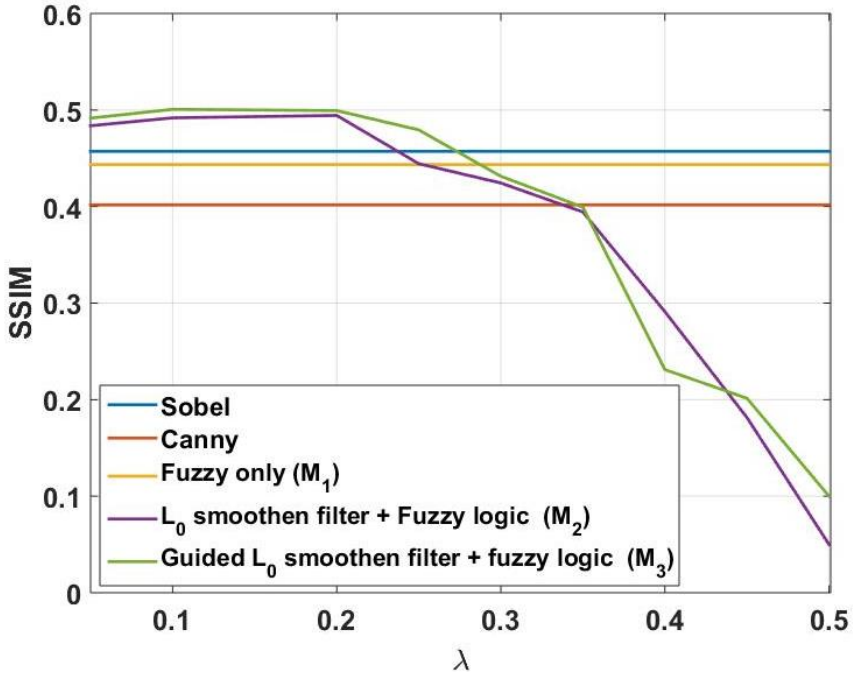


Figure 5.8: Comparison of various edge detection methods (SSIM vs λ)

Again, SSIM is comparatively better for Guided L_0 smoothen filter + fuzzy logic case, but first, it increases with the degree of smoothness; after that, SSIM decreases. This is obvious that more smoothening will lead to structural modifications. In the considered cases, the best SSIM is obtained for the degree of smoothness of 0.1.

In Figure 5.9, Hausdorff Distance is plotted; for the best case, the average minimum distance is 3.30. Again here, with increases in λ, HoD first decreases and then increases. These results clearly reveal that Guided L_0 smoothen filter + fuzzy logic provides better results for edge detection.

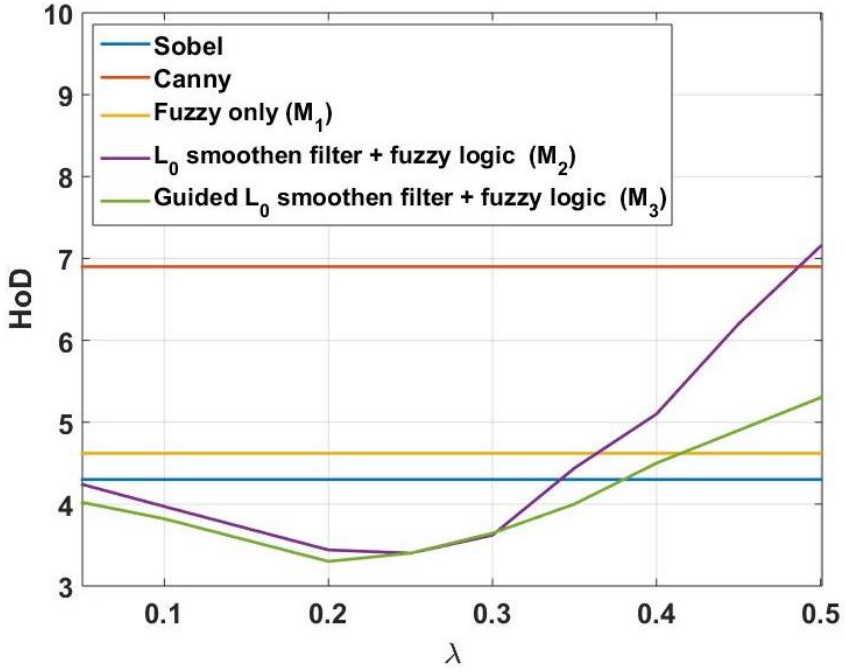


Figure 5.9: Comparison of various edge detection methods (HoDvs λ)

5.8 Comparison with recent methods

To compare our method with recent edge detection techniques, results are compared with Gonzalez, C. et al. (Gongalez, Melin, Castro, Mendoza, & Castillo, 2016) (Gongalez, Patricia, Juan, & Olivia, 2016) work where edges are detected using Sobel and type-2 fuzzy logic method. The results were tested on more than 100 images, and a few results are shown in Figure 5.10.

In Figure 5.10, four rows and four columns are shown, the first column shows the original image, and in the second column, results are shown for canny edge detection. In columns 3 and 4, results are shown for Gonzalez, C. et al. and the proposed method, respectively. In the Canny method, large numbers of false edges are accepted, leading to the lowering of the F-score. It is also observable that when the intensity difference is less Canny method fails to detect edges (fourth-row, second column). Our method is comparable to Gonzalez, C. methods with less number of falsely accepted edges (second row and second, third columns).

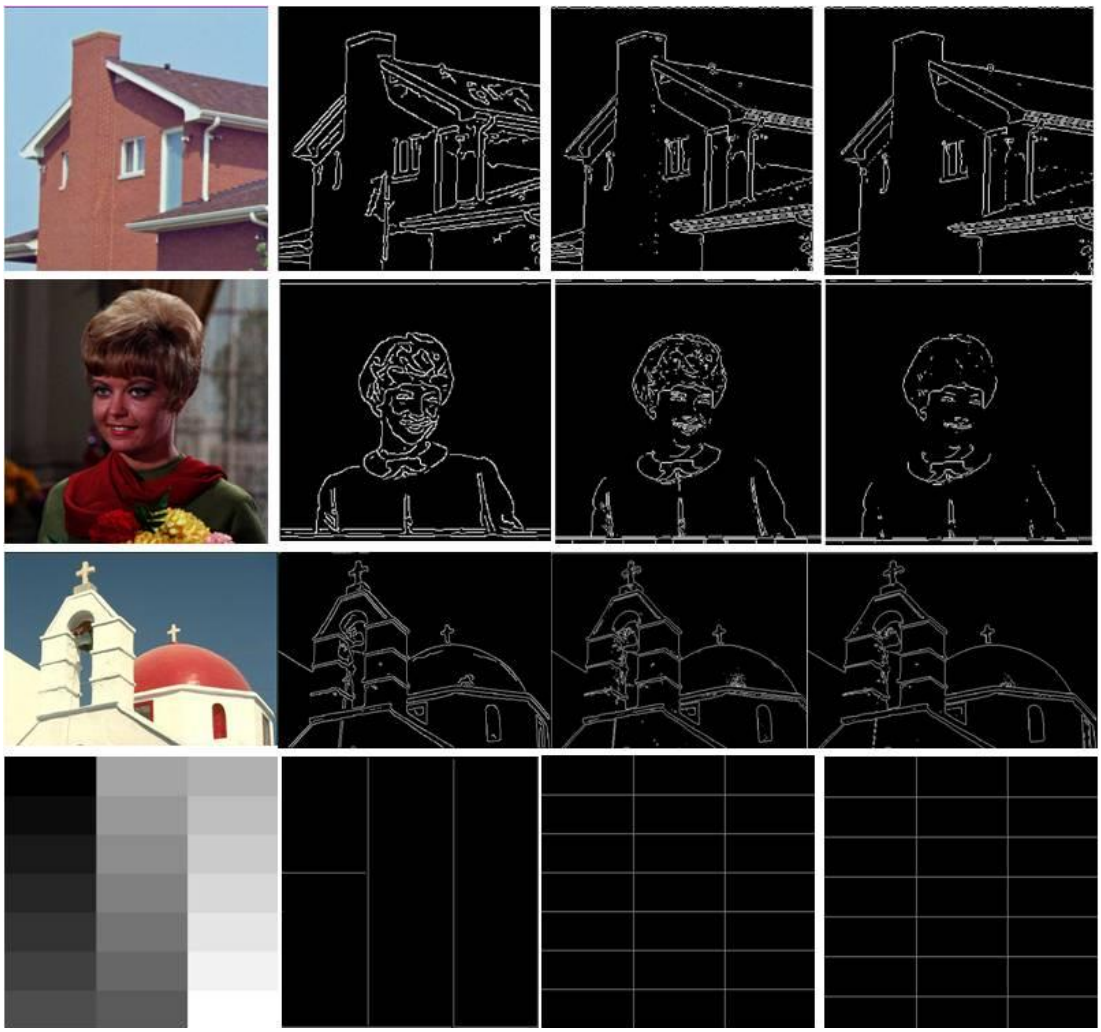


Figure 5.10: Each row left to right: Original, Canny, Gonzalez, C et al., and Proposed

In Table 5.1, notable and recently proposed methods are contrasted in regards of F-score. For classical methods, Canny and Sobel's F-measure is 0.49 and 0.40, respectively. The recently proposed learning-based methods have F-scores ranging from 0.63 to 0.78. A new kernel-based method with singular value decomposition (Avots, Arslan, Valgma, Gorbova, & Anbarjafari, 2018) has F-score as high as 0.83. Our proposed methods, L_0 smoothen filter + fuzzy logic (M_2) and Guided L_0 smoothen filter + fuzzy logic (M_3), attain an F-score of 0.82 and 0.848, respectively.

Table 5.2: Comparing Notable Edge Detection Techniques

Methods	**Year**	**F-measure**
Canny (Canny, 1986)	[1996]	0.49
Sobel (Vincent & Folorunso, 2009)	[2009]	0.40
BEL (Dollar, Tu, & Belongie, Supervised learning of edges and object boundaries, 2006)	[2006]	0.63
gPb (Arbelaez, Maire, Fowlkes, & Malik, 2011)	[2011]	0.71
Sketch Token (Lim, Zitnick, & Dollar, 2013)	[2013]	0.73
Structure Forest (Dollar & Zitnick, Structured forests for fast edge detection, 2013)	[2013]	0.71
Holistically-Nested Edge Detection (Xie & Tu, "Holistically-nested edge detection", 2015)	[2015]	0.78
Gonzalez, C et.al. (Gongalez, Melin, Castro, Mendoza, & Castillo, 2016)	[2016]	0.83
Fuzzy only (M_1)	[2019]	0.77
L_0 smoothen filter + fuzzy logic (M_2)	[2019]	0.82
Guided L_0 smoothen filter + fuzzy logic (M_3)	[2019]	0.848

5.9 Summary

Edge detection has been an important area of research for many years due to its utility in many fields like image segmentation and medical, forensic and defence applications. In past years various methods based on kernels and soft computing based has been proposed, however each one is dependent on a threshold mechanism and is susceptible to misleading acceptance and rejection. To deal with such issues, in this work, a fuzzy-based edge detection mechanism is proposed where edges are controlled using smoothen filters. This work discusses two types of smoothen filters, L_0 smoothen, and guided-L_0 smoothen filters. Using these filters, prominent edges may be preserved and thus enhancing the effectiveness of edge detection. The effectiveness of the proposed edge detection algorithmis compared usingdifferent measures i.e. FoM, SSIM, F-measure and HoD, and it has been found that guided L_0 smoothen filter inconjuction with fuzzy logic produces better results. It is also found that smoothening should be done carefully, and it should be within a limit to obtain better results otherwise, SSIM slips down, and image quality goes down.

Chapter Six

Guided Image Filtering and Ant Colony Optimization for Edge Detection

6.1 Introduction

Edge detection is an important phenomenon in various classes of engineering problems. In general, edge detection comprises three steps.The first step is to remove the unwanted noise from the image. For the removal of noise, high pass filtering is performed. It is also customary to note that edges are also high-frequency components; therefore, during the process of noise removal, some edges may also get removed. Therefore trade-offs exist between edge detection and noise removal mechanism. Hence, an edge preservation mechanism is necessary during the removal of noise. To preserve the true edges, edge enhancement is essential to make them differentiable from noisy pixels to retain them after the filtering process. Keeping this in view, we have considered guided image filtering to enhance edges while reducing noise. Then the differential operator is applied in the second stage to detect the edges, and edge localization is used in the third step to find genuine edges. This paper presents an ant colony optimization-based edge detection process, where guided filtering is used to enhance the edges.

As discussed in Chapter Three the ACO-based edge detection mechanism is a reasonably better technique with good accuracy (Tian, Weiyu, & Shengli, 2008) (Lu & Chen, 2008) (Gupta & Gupta, 2013). However, ACO also fails to detect weak edges, as in the case of other techniques. Before applying ACO, edges should be enhanced so they can be easily identifiable. The objective of the chapter is twofold, in the first step, guided filtering is used for edge enhancement, while in the second step, ACO is used for edge detection. The block diagram of proposed work is shown in Figure 6.1.

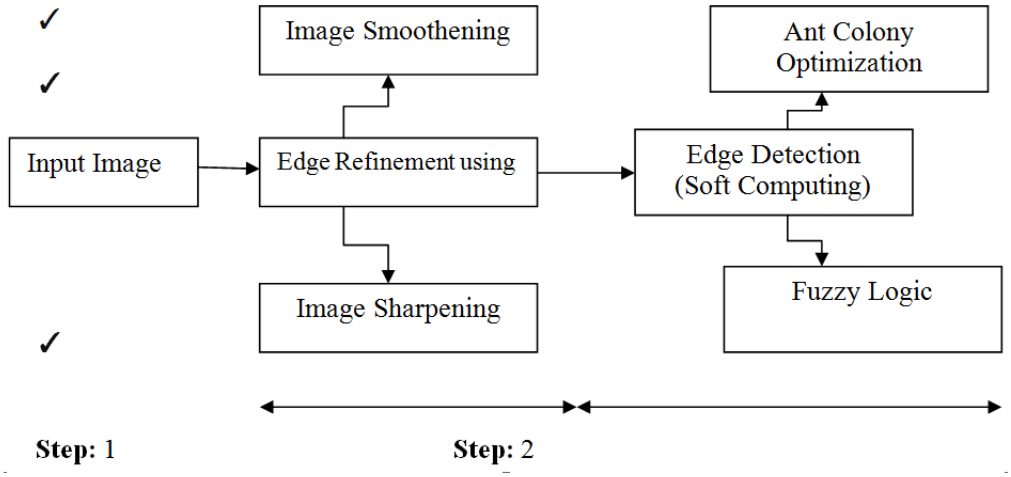


Figure 6.1: Block diagram for the Proposed work

6.2 Novelty of the Proposed Methods

The combination of guided image filtering and ant colony optimization for edge detection presents a novel and powerful approach to accurately identifying boundaries and sharp transitions within digital images.

6.2.1 Guided Image Filtering

- Preservation of Edge Information: Guided image filtering techniques maintain edge information while reducing noise, ensuring that important features are retained in the filtered image.
- Adaptive Filtering Strength: Similar to the guided L_0 smoothen filter, guided image filters adaptively adjust their filtering strength based on local image structures, resulting in enhanced edge preservation.
- Noise Reduction: By selectively smoothing regions of the image while preserving edges, guided image filtering effectively reduces noise without compromising important structural details.

6.2.2 Ant Colony Optimization (ACO) for Edge Detection

- Inspired by Biological Systems: Ant colony optimization is a metaheuristic inspired by the foraging behavior of ants. In the context of edge detection, ACO

algorithms simulate the collective behavior of ants to search for optimal paths along edges.

- Edge Strength Estimation: ACO algorithms can be employed to estimate the strength of edges by iteratively traversing the image and accumulating pheromone trails along potential edge paths.
- Global and Local Optimization: ACO algorithms perform both global and local optimization, allowing them to efficiently explore the entire image space while also focusing on fine details and local edge structures.
- Robustness to Noise and Variations: By leveraging the collective intelligence of ant-like agents, ACO-based edge detection methods are inherently robust to noise and variations in image content, leading to more reliable edge maps.

6.2.3 Advantages and Applications

- High Accuracy: The combination of guided image filtering and ACO-based edge detection offers high accuracy in identifying edge pixels, even in complex and noisy image environments.
- Robustness: The robustness of ACO algorithms to noise and variations in image content ensures consistent performance across diverse datasets and imaging conditions.
- Broad Applicability: The proposed method finds applications in various fields such as medical imaging, robotics, computer vision, and remote sensing, where accurate edge detection is essential for subsequent analysis and decision-making processes.
- Efficiency: Despite its complexity, the proposed approach remains computationally efficient, making it suitable for real-time edge detection tasks in applications requiring rapid image processing.

In summary, the integration of guided image filtering and ant colony optimization for edge detection represents a promising direction in image processing, offering improved accuracy, robustness, and efficiency compared to traditional methods.

6.3 The Proposed Work

In this section, both the proposed works are discussed.

6.3.1 Work I

In the first proposed work, we suggested the use of guided filtering with the original ACO algorithm. In this work, the first edge enhancement is done using guided image filtering. Image sharpening is an effect when applied to images, giving them a much sharper experience. In the sharpened image, the edges are clearly visible to the user. Background and foreground parts become more differentiable in a sharpened image. To detect the weak edges, edges are enhanced to make them sharp so they can be easily detected. Guided image filtering has been used to enhance the edges. The edge enhancement process using Guided Filtering is the same as discussed in 4.4.3, so the details are not discussed again. After enhancing the edges, the original ACO has been applied, and edge detection is done.

6.3.2 Work II

In the second proposed work, after doing edge enhancement using guided image filtering, then instead of using traditional ACO, we applied modified ACO to detect edges on the enhanced image. Some modifications are proposed in the classical-ACO algorithm to achieve better results. The modified ACO-based edge detection method is discussed below:

a. Modified ACO Edge Detection Method

Like other traditional edge detection methods, the intensity values of the input image in the modified ACO technique are transformed into the pheromones values. These transformed values are used to detect the edges of the image. In this method, a grayscale image is taken as input. Now, in the input image, we select m nodes randomly, assuming they as the artificial ants**.** As per the ant moving rule, all selected ants move towards their neighbourhood nodes. We must note here that the movement of each ant is fixed to L steps, where L represents a predefined value. As soon as the movement of each ant is finished, an update in the pheromone matrix is performed. This process is continued till we get a fixed number of iterations. At last, edges are

detected based on the huge amounts of pheromone. This implies that we must have a fixed threshold value of the pheromone to detect an edge.

- **Initialization**

We randomly select m nodes as the artificial ants.We take the amount of pheromone for each one of the pixels as 0.0001.

- **Node transition rule construction**

We select *m* artificial ants randomly and allow them to move from their position to their admissible neighbourhoods as per a probability transition rule. This rule is formulated based on the local intensity and could be defined as:

$$p_{(i,j)(u,v)}^{n} = \frac{\left(\tau_{u,v}^{n-1}\right)^{\alpha}(\eta_{u,v})^{\beta}}{\sum_{u,v\in\Omega_{(i,j)}}\left(\tau_{u,v}^{(n-1)}\right)^{\alpha}(\eta_{u,v})^{\beta}}, \quad v\varepsilon\Omega_{u} \text{.....(6.1)}$$

In the above Eq. (6.1) , *p* defines the movement probability of the selected ant to its neighbourhoods, and *n* represent the number of iteration. The efficacy of pheromone and heuristic informationis controlled by the parameters such as α and β, respectively. $\tau(u, v)$ is the pheromone quantity, and $\eta(a, b)$ is heuristic information for node (u, v). $\Omega(i, j)$ indicates the acceptable neighbourhoods where the selected ant may travel.

While selecting the neighbourhoods for each ant, using the heuristic function η (a, b) produces remarkable results. A 5×5 design (Figure 6.2) is used to formulate the perception of the ant about its neighbourhood nodes at node (a, b).

We can define the heuristic function as:

$$\eta_{u,v} = \max[\eta_1, \eta_2, \eta_3]\frac{1}{I\eta_{max}} \text{..... .....(6.2)}$$

Here,

$$\eta_1 = [In(u-2, v-2) + In(u-1, v-2) + In(u, v-2) + In(u+1, v-2) + In(u+2, v-2)] - [In(u-2, v+2) + In(u-1, v+2) + In(u, v+2) + In(u+1, v+2) + In(u+2, v+2)] \text{..... .....(6.3)}$$

$$\eta_2 = [In(u-2,v-1) + In(u-1,v-1) + In(u,v-1) + In(u+1,v-1) + In(u+2,v-1)] - [In(u-2,v+1) + In(u-1,v+1) + In(u,v+1) + In(u+1,v+1) + In(u+2,v+1)]$$

$$\eta_3 = [In(u-2,v) + In(u-1,v)] - [In(u+1,v) + In(u+2,v)]$$

$I\eta_{(u,v)}$ represent the intensity at a given location, while $I\eta_{max}$ is the maximum intensity. Three representation of the clique is shown in Figure 6.2. We assume the pixels with the same colours as a group (refer to Figure 6.2-I). In addition, Figure 6.2-II and Figure 6.2-III show two famous structures considered (Liu & Fang, 2015). By subtracting the summation of each group's pixel values from the summation values of the corresponding group with the same colour, we may calculate $\eta(a,b)$. Now, we choose the highest value among different pairs.

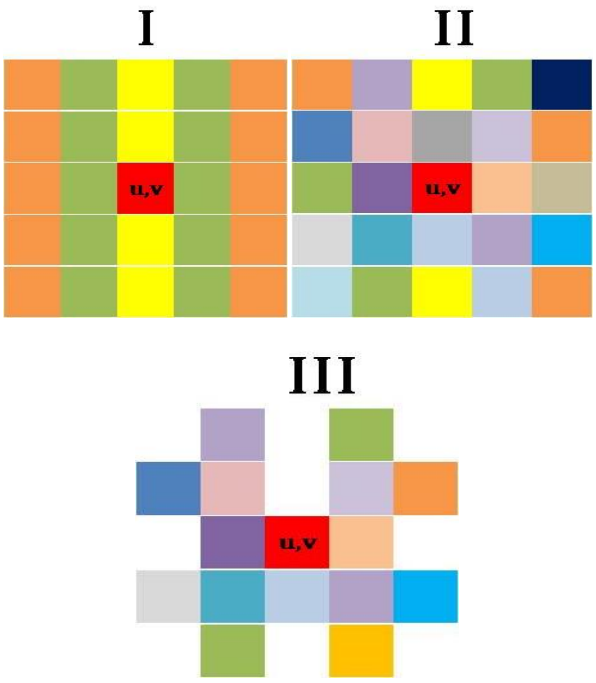


Figure 6.2: Representation of clique

Each selected ant is allowed to move to its eight neighbourhoods based on Eq. (6.1), as shown in Figure 6.3. So, we can say that each ant moves to the node with the highest probability. The above-mentioned process goes on till the completion of the *L* steps. Each of these artificial ants posses pre fixed memory length denoted by *l*.

Therefore, we can say that each ant can memorize only the last *l* nodes visited by it.

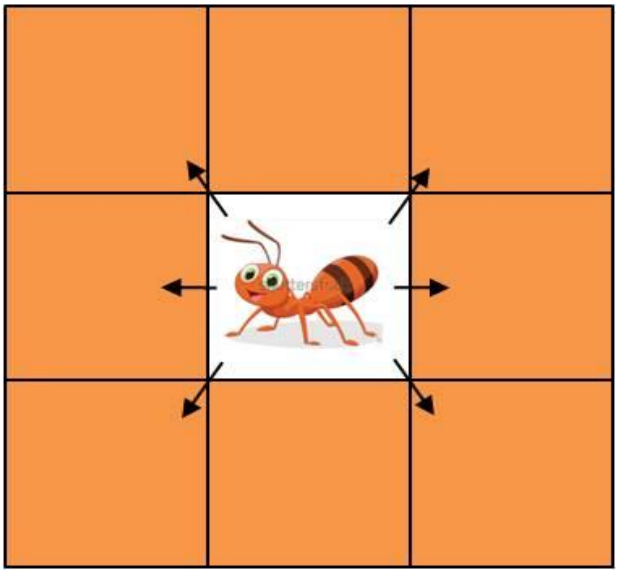


Figure 6.3: Ant movement on image

Pheromone update: We have an update in the pheromone matrix on the proceeding of each artificial ant to *L* steps.This update is done using the following**:**

$$\tau_{(a,b)}(New) = \left[\sum_{s=1}^{S} \Delta\tau_{(u,v)}^{s} + \tau_{(u,v)}(old)(1-\rho)\right]_{d_1=\tau_{min}}^{d_2=\tau_{max}} \text{.....(6.4)}$$

S represents the total number of ants that meet node (*u*,*v*), and $[D]_{d_1}^{d_2}$ is defined as:

$$[D]_{d_1}^{d_2} = \begin{cases} \tau_{min} & if\ D > d_1 \\ \tau_{max} & if\ D > d_2 \\ D & otherwise \end{cases} \text{.....(6.5)}$$

We can estimate $\Delta\tau_{(u,v)}^{s}$ for each node as:

$$\Delta\tau_{(u,v)}^{S} = \begin{cases} \eta_{(u,v)} & if\ s^{th}\ ant\ meets\ node\ (u,v) and\ \eta_{(u,v)} > t \\ 0, & otherwise \end{cases} \text{.....(6.6)}$$

Here, the variable *t* is assumed as a threshold to restrict the amount of the pheromone deposited.

6.4 Results

In the edge detection process, accurately assessing the performance of an algorithm becomes crucial, particularly considering that the positions of pixels may shift from their original locations due to various factors such as noise, blurring, or transformations. To evaluate the effectiveness of our algorithm, we rely on the F-measure metric, which provides a comprehensive measure of a method's precision and recall, thereby offering insights into its overall accuracy.

The F-measure, also known as the F1-score, combines both precision and recall into a single value, making it a suitable metric for comparing edge detection algorithms. Precision measures the proportion of correctly detected edge pixels among all the detected edge pixels, while recall measures the proportion of correctly detected edge pixels among all the ground truth edge pixels. By calculating the harmonic mean of precision and recall, the F-measure provides a balanced assessment of an algorithm's ability to accurately identify edges in an image.

In our evaluation, we compare the F-measure obtained from our algorithm with that of other recent methods to gauge its performance relative to state-of-the-art techniques. This comparison allows us to ascertain whether our algorithm offers improvements in terms of edge detection accuracy compared to existing approaches.

To conduct our evaluation, we utilize a set of simulation parameters, the values of which are detailed in Table 6.1. These parameters encompass various aspects of the edge detection process, including filtering techniques, thresholding strategies, and post-processing methods. By systematically adjusting these parameters and observing their impact on the F-measure, we gain valuable insights into the algorithm's sensitivity to different settings and its overall robustness across diverse image datasets.

Table 6.1: Simulations Parameters

Parameters	**Value**
pheromone matrix, τ_{init} (Initial values)	10^{-4}
Pheromone information, α (Weighting factor)	1
Heuristic information, β (Weighting factor)	0.1
Connectivity neighbourhood, λ	8

Functions adjusting parameter, λ	10
evaporation rate, ρ	0.1
Total number of ants	vary
Total number of ant's movementsteps, *S*	40
Pheromone decay coefficient, ψ	0.05
Tolerance value, ε	0.1
Threshold, *T*	adaptive

The total numbers of ants which need to be taken depend on image size. Considering the image size as (M×N), the numbers of ants are (√MN).

6.4.1 Results Edge Enhancement

In Figure 6.4, results for edge enhancement using guided filtering are shown. Here, four different images are shown in the first column. The first three images are taken from the Berkley image database, while the fourth is the famous Leena image. Each row shows the results for an image under various mechanisms. The second column shows the grayscale version corresponding to its original image in the first column; the edge-enhanced image using guided image filtering is shown in the third column. Finally, a grey-scale version of the enhanced images is shown in the fourth column. It is clear from the results that the edges are sharper and more clear in the enhanced image as compared to the original image. For accurate edge detection, it is desired that the intensity values in grey images should be either 0 or 255. Still, it is not possible to get an exact binary image without using any threshold.

Figure 6.4: First Column; Original image, Second Column; Gray Scale original image, Third Column; Enhanced image, Fourth Column; Gray Scale enhanced image

Still, we expect that the histogram of the image should be more concentrated around 0 and 255. Therefore, in the Figure 6.5, four columns are displayed; in the first column, the original images are shown, and in the second column, a histogram of all the images is shown; in the next column, enhanced images are displayed; and finally, finally in the fourth column histogram of enhanced images are displayed.

It is clear from the Figure 6.5 that the histogram of the greyscale original image intensity is spread from 0 to 255, while in the case of enhanced images, intensities values are more concentrated around 0 and 255.

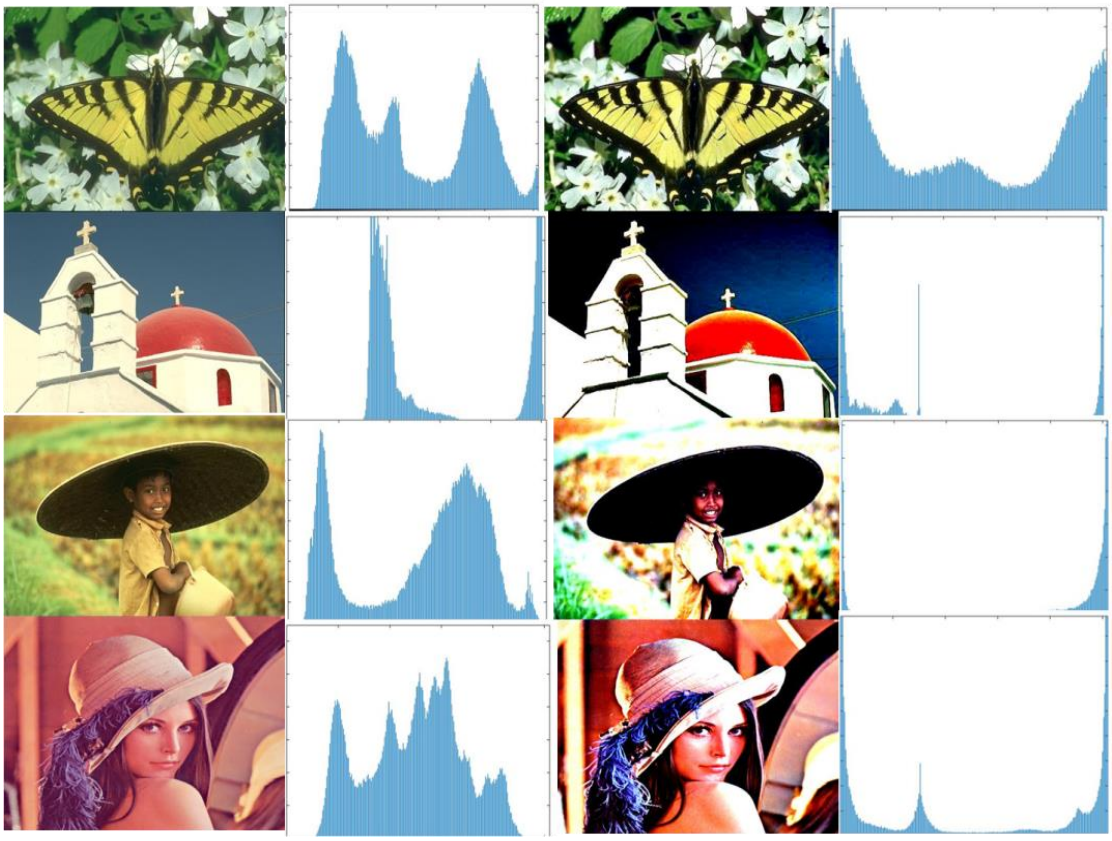


Figure 6.5: First Column; Original image, Second Column; histogram of the original image, Third Column; Enhanced image, Fourth Column; histogram of Scale enhanced image

6.4.2 Results Edge Detection Traditional ACO + Guided Image Sharpening

The results are acquired on a number of images and databases to prove the applicability of the proposed edge detection methodology. Still, we have selected a few of these images from an illustration point of view. In Figure 6.6, the first column displays the original considered images, and correspondingly edge-detected images are displayed in the 2nd column. In the next column, sharpened images are shown, while edge detection on sharpened images is shown in the fourth column. Each row in the figure shows the results for one type of original and sharpened image, along with the corresponding edge detected images. It is also observable that with the proposed method, more edges are detected. The notable areas are marked using circular and oval shapes, where clear differences in detected edges can be observed. However, more edges not always mean that correct edges are detected.

Figure 6.6: First Column; Original image, Second Column; edge detected image, Third Column; Enhanced image, Fourth Column; edge detected enhanced image

6.4.3 Results Edge Detection Modified ACO + Guided Image Sharpening

To prove the applicability of the proposed edge detection methodology, we had tested the method on a wide range of pictures and databases, but for an illustration point of view, we have selected a few of these.

In Figure 6.7, the first row and first column show that the original considered Lena image and the corresponding edge detected image using the original ACO as detailed in (Mittal, et al., 2019) shown in the first-row second column. In the second row and first column, the edge-detected images under ACO and guided filtering is shown, while in the second row, the second column under modified ACO and guided filtering is shown. A clear distinction is observed in ACO and guided filtering-based ACO edge detection. However, much difference is not observed with modified ACO and guided filtering.

Figure 6.7: First row and first column; Original image, First row and second Column; edge detection using ACO, Second row and first column; ACO and Guided filtering, Second row and second column; enhanced ACO and Guided filtering

To see the advantages of the proposed method over earlier methods, we have considered another image, Figure 6.8(a), which is full of varieties like text, symbols, contour etc. is considered. In the Figure 6.8(b), detected edges using ACO is shown. Here orangutan and tiger image edges are not detected; moreover, the letter present on the body of the animal is also not recognized. Figure 6.8(c) detected edges using ACO, and guided image filtering is shown; here, orangutan and tiger images edges are now detected, but the letter present on the body of the animal and the name of the animals are not recognized. In the Figure 6.8(d), detected edges using modified ACO and guided image filtering is shown, here orangutan and tiger images edges and letter present on the body of the animal are now detected; moreover, and name of the animals are clearer now.

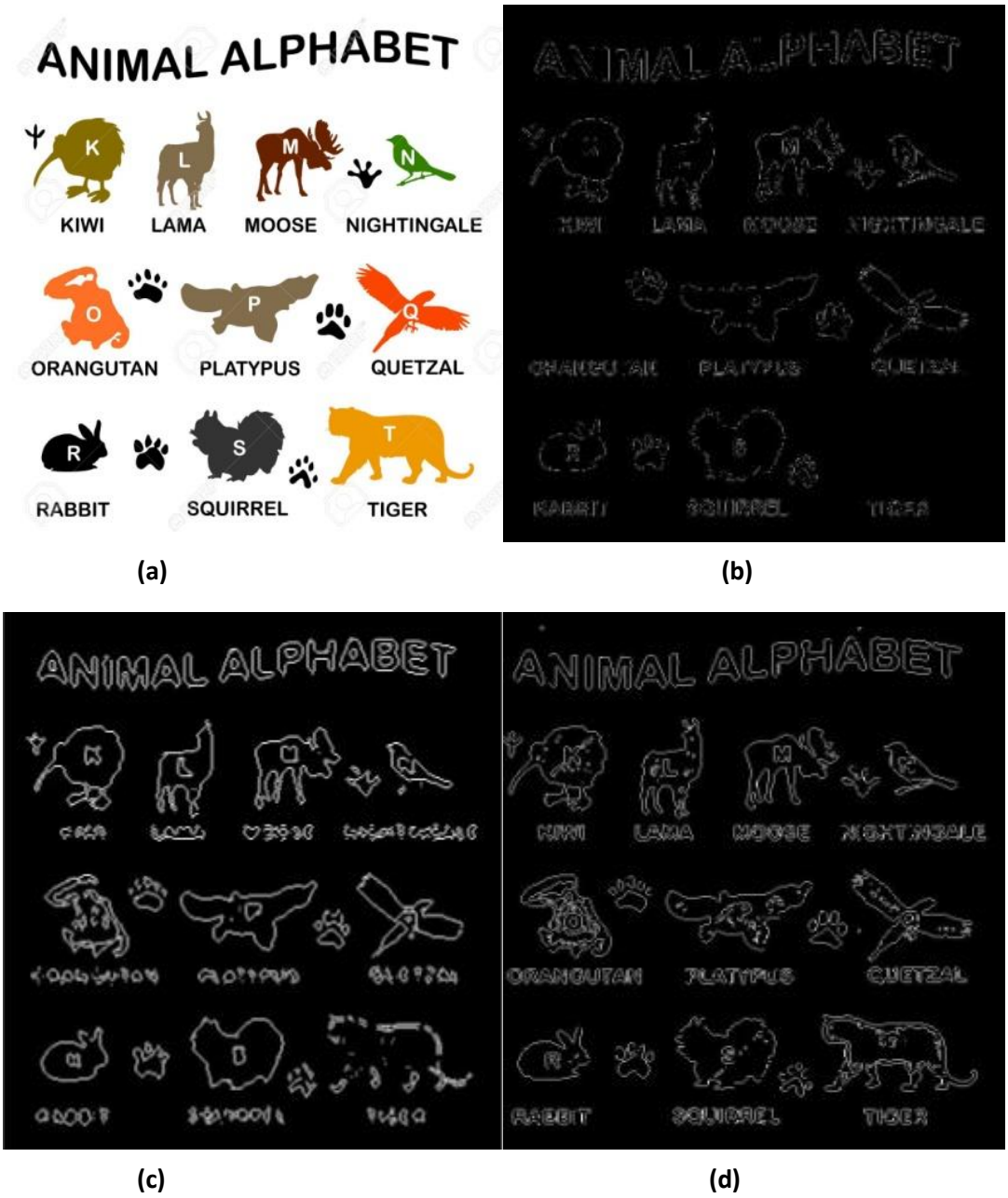


Figure 6.8: (a) Original image (b) Edge detection using ACO (c) ACO and Guided filtering (d) Enhanced ACO and guided filtering

Finally, it is also important to note that more edges do not always mean that correct edges are detected. Therefore, the F-score is obtained for the various images, and the minimum obtained F-score is shown in Table 6.2. From this table, it is clear that the proposed scheme is much better in comparison to notable schemes.

Table 6.2: Comparison with Notable Works

Method	**Year**	**F-measure**
Canny (Canny, 1986)	[1996]	0.49
Sobel (Vincent & Folorunso, 2009)	[2009]	0.40
BEL (Dollar, Tu, & Belongie, Supervised learning of edges and object boundaries, 2006)	[2006]	0.63
gPb (Arbelaez, Maire, Fowlkes, & Malik, 2011)	[2011]	0.71
Sketch Token (Lim, Zitnick, & Dollar, 2013)	[2013]	0.73
Structured Forest (Dollar & Zitnick, Structured forests for fast edge detection, 2013)	[2013]	0.71
ACO (Raheja & Kumar, 2020)	[2020]	0.74
ACO + Guided image filtering (Proposed)		0.79
Modified ACO + Guided image filtering (Proposed)		0.81

6.5 Summary

In this chapter, two sets of work are presented. In the first work, it is shown that the performance of the ACO-based edge detection mechanism can be improved using guided image filtering. In the second method,a modified-ACO algorithm is detailed for edge detection. In the second method for the enhancement of the edges, guided image filtering is performed, and modified ACO is applied for edge detection on the enhanced images. It is found that the proposed scheme can detect minor edge variations with the help of guided filtering and a new heuristic clique function. Obtained results are also compared with recent and notable edge detection techniques in terms of F-score, and it has been found that the proposed scheme performs better edge detection.

Chapter Seven

Conclusions and Future Works

In this thesis, we have formulated algorithms for better edge detection to benefit other image-processing applications. We performed edge detection on images using these algorithms and then compared the values of performance measures. Compared with other state-of-the-art techniques, these algorithms give better results for differentperformance indices for quantitative evaluation like PSNR, MSE, F-measure,FoM, SSIM, HoD, ED, BDM, DE and ρ. The variations in results are also analyzed by changing the values of the parameters. The experiments have been done on the standard dataset Berkeley Segmentation Database (BSD). From work presented in the thesis following conclusions can be made:

Edge detection is an important phenomenon in computer vision. The edge detection process heavily depends on the chosen technique. The traditional edge detection methods that were previously proposed depends heavily on the selected threshold. Soft computing techniques are considered as powerful edge detection methods due to their adaptability. In this thesis, the image edge detection problem is first solved using the modified-ACO method, which generates a desirable result compared to many other algorithms and approaches. This section employs a modified intensity mapping function to capture the intensity variations. Also, the threshold has been chosen adaptively to improve the accuracy of the algorithm. The algorithm is applied to a total of six images of BSD. The results are compared using PSNR and F-score, which are better than traditional approaches of edge detection as well as other recent methods.

In various applications, it has been observed that all the edges in a particular image are not so significant. Therefore, it is necessary to look into a technique that allows for the regulation of the amount of recognised edges. Hence a fuzzy-logic-based edge detection method where the quality of edges is controlled using a sharpening guided filter and noise due to the sharpening is controlled using a Gaussian filter is formulated. The accuracy of the method is judged using a variety of statistical measures. It has been found that a significant improvement in the detected edges can be obtained by properly selecting the smoothening parameters. The results have been

compared with Canny and Fuzzy edge detection without guided filtering, which showed that guided filtering improves the performance of edge detection. The sensitivity of different parameters used is also analyzed.

Fuzzy logic-based edge detection is heavily investigated by changing the number of rules edge detection can be improved. However, due to large colour variations in the images, false edges are detected and even using fuzzy rules, they cannot be reduced significantly. These falsely detected edges can be controlled by using smoothened filter while controlling the degree of smoothness. Then an algorithm is proposed using a fuzzy logic-based edge detection mechanism while using Guided L_0 smoothen filter for the smoothening of the image under various degrees of smoothens. Simulation results for edge detection are provided for the Sobel, Canny, Fuzzy-logic based edge detection and fuzzy-logic edge detection usingan L_0 smoothen filter. The results are compared with classical and modern methods. Simulation is performed on Berkley Segmentation Database (BSD) and USC-SIPI Image Database while considering more than 100 images. The obtained F-measure is as high as 0.848.

In the past, different edge detection methods based on soft computing techniques have been presented. However,these algorithms fails to detect all the real edges. For accurate edge detection, a new two-level technique is proposed, where in the first step, image edges are enhanced using guided image filtering after that an improved ant colony optimization method is applied to these enhanced images for better edge detection.

7.1 Conclusions

The different algorithms proposed in this thesis are very effective and give an improved performance for edge detection. Based onthe work done following conclusions can be made:

1. The modified-ACO-based edge detection mechanism used a novel intensity mapping function to capture both linear and sinusoidal variations of intensity profile, and the use of adaptive thresholding leads to more accurate edge

detection. By choosing various cliques where maximum intensity variations can be captured, the results can be improved in terms of accurate edge detection.

2. We developed a method to tackle all major issues of edge detection: false edges because of noise, and missing true edges, by utilizing guided image filtering where the quality of the edges can be well controlled. Using guided image filtering, both image smoothing and image sharpening can be done precisely. The radius of the guided filters can be used to counter balance the filtered images' colouring effect. Using guided image filtering, edge localization is better in comparison to traditional and state-of-art methods.The compared results clearly show that the recently proposed method, like ANN and DNN, is comparatively more complex, with higher run time and lesser accuracy in terms of F-score.

3. We also utilized the advantages of L_0 smoothing in conjunction with guided filtering to control the degree of smoothing of images so that true edges can not be missed. The proposed algorithms give a much-improved performance where F-measure is as high as 0.848.

7.2 Future Works

1. The obtained results can be improved using further post-image processing operations like Non-maxima suppression as used in the Canny edge detection method. Moreover, research can be made to develop an edge detection mechanism which is free from mapping functions by taking the gradient magnitude and directions into account in ACO-based detection.

2. ACO method can be further improved by using fuzzy rules to detect true edges and to remove false edges.

3. A better version of Guided Image Filters can be developed to enhance the results further.

4. The fuzzy scheme based on intensity difference can be further extended by considering the mean and variance of the intensities of the pixels in the

surrounding overlapped windows.

5. The work can be further analyzed with the introduction of varying level of noise onto different images.

List of Publications

Published (Journals)

1. **Raheja, Sahil**, and Akshi Kumar. "Edge detection based on type-1 fuzzy logic and guided smoothening."*Evolving Systems*(2019) :1-16,doi:https://doi.org/10.1007/s12530-019-09304-6 (**SCIE**)

2. Kumar, Akshi, and **Sahil Raheja,** "Edge Detection in Digital Images Using Guided L0 Smoothen Filter and Fuzzy Logic." *Wireless Personal Communications* 121, no. 4 (2021): 2989-3007, doi:https://doi.org/10.1007/s11277-021-08860-y (**SCIE**)

3. **Raheja, Sahil**, and Akshi Kumar. "Edge detection using ant colony optimization under novel intensity mapping function and weighted adaptive threshold" *International Journal of Integrated Engineering* 12, no. 1 (2020): 13-26,doi:https://dx.doi.org/10.30880/ijie.2020.12.01.002 **(SCOPUS)**

Conferences

1. Kumar, Akshi, and **Sahil Raheja.** "Edge detection using guided image filtering and enhanced ant Colony optimization." *Procedia Computer Science* 173 (2020): 8-17

2. Kumar, Akshi, and **Sahil Raheja.** "Edge Detection Using Guided Image Filtering and Ant Colony Optimization." In *The International Conference on Recent Innovations in Computing*, pp. 319-330. Springer, Singapore, 2020.

Bibliography

Aborisade, D. (2011). Novel fuzzy logic based edge detection technique. International Journal of Advanced Science and Technology , 29 (1), 75-82.

Aboutabit, N. (2021). A new construction of an image edge detection mask based on Caputo–Fabrizio fractional derivative. The Visual Computer , 37 (6), 1545-1557.

Alshennawy, A., & Aly, A. (2009). Edge detection in digital images using fuzzy logic technique. World Academy of science, engineering and technology International Journal of Computer and Information Engineering , 3 (3), 54-548.

Anand, C. S., & Jyotinder, fS. S. (2010). Wavelet domain non-linear filtering for MRI denoising. Magnetic Resonance Imaging , 28 (6), 842-861.

Angelov, P., Sadeghi-Tehran, P., & Ramezani, R. (2011). An approach to automatic real-time novelty detection, object identification, and tracking in video streams based on recursive density estimation and evolving Takagi-Sugeno fuzzy systems. Int J Intell Syst , 26, 189-205.

Arbelaez, P., Maire, M., Fowlkes, C., & Malik, J. (2011). Contour detection and hierarchical image segmentation. IEEE transactions on pattern analysis and machine intelligence , 33 (5), 898-916.

Archana, J., & Aishwarya, P. (2016). A Review on the Image Sharpening Algorithms Using Unsharp Masking. International Journal of Engineering Science and Computing , 6 (7), 8729-8733.

Ari, S., Ghosh, D. K., & Mohanty, P. K. (2014). Edge detection using ACO and F ratio. Signal, Image and Video Processing , 8 (4), 625-634.

Avots, E., Arslan, H., Valgma, L., Gorbova, J., & Anbarjafari, G. (2018). A new kernel development algorithm for edge detection using singular value ratios. Signal Image and Video Processing , 12 (7), 1301-1309.

Banharnsakun, A. (2019). Artificial bee colony algorithm for enhancing image edge detection. Evolving Systems , 10 (4), 679-687.

Bauer, B. O., Jianchun, Y., Steven, L. N., & Douglas, J. S. (1998). Event detection and conditional averaging in unsteady aeolian systems. Journal of Arid Environments , 39 (3), 345-375.

Begol, M., & Maghooli, K. (2011). Improving digital image edge detection by fuzzy systems. International Journal of Computer and Information Engineering , 5 (9), 980-983.

Berkeley Segmentation Dataset: Images. (2003, Oct 31). Retrieved from www2.eecs.berkeley.edu: 101. https://www2.eecs.berkeley.edu/Research/Projects/CS/vision/bsds/BSDS300/html/dataset/images.html

Bertasius, G., Shi, J., & Torresani, L. (2015). Deep edge: A multi-scale bifurcated deep network for top-down contour detection. Proceeding of IEEE conference on computer vision and pattern recognition (pp. 4380-4389). IEEE.

Brownrigg, D. R. (1984). The weighted median filter. Communications of the ACM , 27 (8), 807-818.

Canny, J. (1986). A computational approach to edge detection. IEEE Transactions on pattern analysis and machine intelligence , PAMI-8 (6), 679-698.

Caponetti, L., Abbattista, N., & Carapella, G. (1994). A genetic approach to edge detection. Proceedings of 1st International Conference on Image Processing (pp. 318-322). Austin, TX, USA: IEEE.

Caraffa, L., Tarel, J.-P., & Charbonnie, P. (2015). The guided bilateral filter: When the joint/cross bilateral filter becomes robust. IEEE Transactions on Image Processing , 24 (4), 1199-1208.

Chawla, N. V., Japkowicz, N., & Kołcz, A. (2004). Editorial: Special Issue on Learning from Imbalanced Data Sets. ACM SIGKDD Explorations Newsletter , 6 (1), 1-6.

Chawla, S., & Khokhar, A. (2015). Edge Detection Technique using HSI and Fuzzy Inference System. International Journal of Science and Research (IJSR) , 4 (7), 683-689.

Chen, D., Ting, Z., & Xiaosheng, Y. (2012). A new method of edge detection based on PSO. Conference: Proceedings of the 9th international conference on Advances in Neural Networks - Volume Part II (pp. 239-246). Berlin, Heidelberg: Springer.

Chen, S., Chang, Y., & Pan, J. (2013). Fuzzy rules interpolation for sparse fuzzy rule-based systems based on interval type-2 Gaussian fuzzy sets and genetic algorithms. IEEE transactions on fuzzy systems , 21 (3), 412-425.

Deng, Y., & Manjunath, B. (2001). Unsupervised segmentation of color-texture regions in images and video. IEEE Transaction on Pattern Analysis and Machine Intelligence , 23 (8), 800-810.

Ding, X., Chen, L., Zheng, X., Huang, Y., & Zeng, D. (2016). Single image rain and snow removal via guided L_0 smoothing filter. Multimedia Tools and Applications , 75 (5), 2697-2712.

Dollar, P., & Zitnick, C. (2013). Structured forests for fast edge detection. (pp. 1841-1848). Sydney, NSW, Australia: IEEE.

Dollar, P., Tu, Z., & Belongie, S. (2006). Supervised learning of edges and object boundaries. 2006 Proceeding of Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06) (pp. 1964-1971). New York, NY, USA.

Dorigo, M., Mauro, B., & Stutzle, T. (2006). Ant colony optimization. IEEE computational intelligence magazine , 1 (4), 28-39.

El-Khamy, S., Lotfy, M., & El-Yamany, N. (2000). A modified fuzzy Sobel edge detector. Proceedings of the Seventeenth National Radio Science Conference (pp. C32/1-C32/9). Minufiya, Egypt: IEEE.

Farbod, M., Akbarizadeh, G., Kosarian, A., & Rangzan, K. (2018). Optimized fuzzy cellular automata for synthetic aperture radar image edge detection. Journal of Electronic Imaging , 27 (1), 13-30.

Fram, J., & Deutsch, E. (1975). On the quantitative evaluation of edge detection schemes and their comparison with human performance. IEEE Transactions on Computers , C-24 (6), 616-628.

Gongalez, C., Castro, J., Patricia, M., & Oscar, C. (2015). Cuckoo search algorithm for the optimization of type-2 fuzzy image edge detection systems. 2015 Proceeding of IEEE Congress on Evolutionary Computation (CEC) (pp. 449-455). Sendai, Japan: IEEE.

Gongalez, C., Melin, P., Castro, J., Mendoza, O., & Castillo, O. (2016). An improved sobel edge detection method based on generalized type-2 fuzzy logic. Soft Computing , 20, 773-784.

Gongalez, C., Patricia, M., Juan, R., & Olivia, M. (2016). Optimization of interval type-2 fuzzy systems for image edge detection. Applied Soft Computing , 47, 631-643.

Gonzalez, R. C., & Woods, R. E. (1992). Digital Image Processing. Addison Wesley.

Gu, B., Wujing, L., Minyun, Z., & Minghui, W. (2012). Local edge-preserving multiscale decomposition for high dynamic range image tone mapping. IEEE Transactions on image Processing , 22 (1), 70-79.

Gupta, C., & Gupta, S. (2013). Edge Detection of an Image Based on Ant Colony Optimization Techniques. International Journal of Science and Research (IJSR) , 2 (6), 114-120.

Gupta, S., & Mazumdar, S. G. (2013). Sobel edge detection algorithm. International journal of computer science and management Research , 2 (2), 1578-1583.

Gupta, S., & Mazumdar, S. (2013). Sobel edge detection algorithm. International journal of computer science and management Research , 2 (2), 1578-1583.

Hacini, M., Akram, H., Herman, A., & Fella, H. (2017). A 2D-fractional derivative mask for image feature edge detection. Proceeding of International Conference on Advanced Technologies for Signal and Image Processing (pp. 1-6). IEEE.

Hagara, M., & Kubinec, P. (2018). About Edge Detection in Digital Images. Radio engineering , 27 (4), 919-929.

Harwood, D., Muralidhara, S., Hannu, H., & Larry, S. D. (1987). A new class of edge-preserving smoothing filters. Pattern Recognition Letters , 6 (3), 155-162.

Heel, M. V. (1987). Similarity measures between images. Ultramicroscopy , 21 (1), 95-100.

Hoiem, D., & Collins, R. (n.d.). CSC320: Introduction to Visual Computing. Retrieved 01 08, 2023, from www.cs.toronto.edu: https://www.cs.toronto.edu/~guerzhoy/320/lec/edgedetection.pdf

Hore, A., & Ziou, D. (2010). Image quality metrics: PSNR vs. SSIM. 2010 20th Proceeding of International Conference on Pattern Recognition (pp. 2366-2369). Istanbul, Turkey: IEEE.

Hosang, J., Benenson, R., & Schiele, B. (2017). Learning non-maximum suppression. Computer Vision and Pattern Recognition .

Hsu, C., & Juang, C. (2011). Evolutionary Robot Wall-Following Control Using Type-2 Fuzzy Controller With Species-DE-Activated Continuous ACO. IEEE transactions on fuzzy systems , 21 (1), 100-112.

Hwang, J., & Liu, T. (2015). Pixel-wise deep learning for contour detection. Accepted as a workshop contribution at ICLR 2015 , 1-2.

Jena, K. K. (2015). Application of COM-SOBEL operator for edge detection of images. International Journal of Innovative Science, Engineering & Technology , 2 (4), 48-51.

Jing, T., Yu, W., & Xie, S. (2008). An ant colony optimization algorithm for image edge detection. In: Proceedings of the IEEE International (pp. 751-756). IEEE.

Joshi, A., Kulkarni, O., Kakandikar, G., & Nandedkar, V. (2017). Cuckoo search optimization-a review. Materials Today: Proceedings , 4, pp. 7262-7269.

Kaming, H., Sun, J., & Tang, X. (2013). Guided image filtering. IEEE Transactions on Pattern Analysis and Machine Intelligence , 35 (6), 1397-1409.

Katiyar, S. K., & Arun, P. (2012). Comparative analysis of common edge detection techniques in context of object extraction. IEEE Transactions of Geoscience and Remote Sensing , 50 (11(b)), 68-79.

Kenan, M., Hui, F., Zhao, X., & Prehofer, C. (2016). Multiscale edge fusion for vehicle detection based on difference of Gaussian. Optik , 127 (11), 4794-4798.

Kennedy, J., & Russell, E. (1995). Particle swarm optimization. Proceeding of International conference on neural networks. 4, pp. 1942-1948. Perth, WA, Australia: IEEE.

Kim, D., Lee, W., & Kweon, I. (2004). Automatic edge detection using 3× 3 ideal binary pixel patterns and fuzzy-based edge thresholding. Pattern Recognition Letters , 25 (1), 101-106.

Kiranpreet, K., Mutenja, V., & Gill, I. S. (2010). Fuzzy logic based image edge detection algorithm in MATLAB. International Journal of Computer Applications , 1 (22), 55-58.

Konishi, S., Yuille, A., Coughlan, J., & Zhu, S. (2003). Statistical Edge Detection: Learning and Evaluating Edge Cues. IEEE Transactions On Pattern Analysis And Machine Intelligence , 25 (1), 57-74.

Kovesi, P. (2010). Fast almost-gaussian filtering. Proceeding of International Conference on Digital Image Computing: Techniques and Applications (pp. 121-125). IEEE.

Kumar, A., & Sodhi, S. S. (2020). Comparative analysis of gaussian filter, median filter and denoise autoenocoder. 7th Proceeding of International Conference on Computing for Sustainable Global Development. IEEE.

Kumar, S., Upadhyay, A. K., Dubey, P., & Varshney, S. (2021). Comparative analysis for Edge Detection Techniques. Proceeding of International Conference on Computing, Communication, and Intelligent Systems (ICCCIS). Greater Noida: IEEE.

Kuo, Y. H., Lee, C.-S., & Liu, C.-C. (1997). A new fuzzy edge detection method for image enhancement. Proceedings of 6th International Fuzzy Systems Conference. 2, pp. 1069-1074. Barcelona, Spain: IEEE.

Lakshmi, S., & Sankaranarayanan, V. (2010, August 20). A study of edge detection techniques for segmentation computing approaches. International Journal of Computer Applications , 7-10.

Lendave, V. (2021). A Guide to Different Types of Noises and Image Denoising Methods. DEVELOPERS CORNER.

Lim, J., Zitnick, C. L., & Dollar, P. (2013). Sketch tokens: A learned mid-level representation for contour and object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 3158-3165). Portland, OR, USA: IEEE.

Liu, G., & Haralick, R. (2002). Optimal matching problem in detection and recognition performance evaluation. Pattern Recognition , 35 (10), 2125-2139.

Liu, X., & Fang, S. (2015). A convenient and robust edge detection method based on ant colony optimization. Optics Communications , 353, 147-157.

Liu, Y., Cheng, M., Hu, X., Wang, J., Zhang, L., & Bai, X. (2017). Richer convolutional features for edge detection. IEEE Transactions on Pattern Analysis and Machine Intelligence , 41 (8), 1939-1946.

Lopez-Molina, C., Ayala-Martini, D., Lopez-Maestresalas, A., & Bustince, H. (2017). Baddeley's delta metric for local contrast computation in hyperspectral imagery. Progress in Artificial Intelligence , 6 (2), 121-132.

Lopez-Molina, C., Baets, B. D., Bustince, H., Sanz, J., & Barrenechea, E. (2013). Multiscale edge detection based on Gaussian smoothing and edge tracking. Knowledge-Based Systems , 44, 101-111.

Lopez-Molina, C., Bustince, H., Fernandez, J., Couto, P., & De Baets, B. (2010). A gravitational approach to edge detection based on triangular norms. Pattern Recognition , 43 (11), 3730-3741.

Lu, D., & Chen, C. (2008). Edge detection improvement by ant colony optimization. Pattern Recogition Letters , 29 (4), 416-425.

Lu, D.-S., & Chen, C.-C. (2008). Edge detection improvement by ant colony optimization. Pattern Recognition Letters , 29 (4), 416-425.

M, V. (2022, August). Comprehensive Guide to Edge Detection Algorithms. Retrieved January 2023, from analyticsvidhya.com: https://www.analyticsvidhya.com/blog/2022/08/comprehensive-guide-to-edge-detection-algorithms/

Ma, X., & Grimson, W. (2005). Edge-based rich representation for vehicle classification. Proceeding of Tenth IEEE International Conference on Computer Vision (ICCV'05). 1, pp. 1185-1192. Beijing, China: IEEE.

Maini, R., & Aggarwal, H. (2009). Study and Comparison of Various Image Edge Detection Techniques. International Journal of Image Processing (IJIP) , 3 (1), 1-11.

Marmanis, D., Schindler, K., Wegner, J., Galliani, S., Datcu, M., & Stilla, U. (2018). Classification with an edge: improving semantic image segmentation with boundary detection. ISPRS Journal of Photogrammetry and Remote Sensing , 135, 158-172.

Marr, D., & Hildreth, E. (1980). Theory of edge detection. Proceedings of the Royal Society , 207 (1167), 187-217.

Mathur, S., & Ahlawat, A. (June-July 2008). Application of fuzzy logic on image edge detection. Intelligent Information and Engineering Systems INFOS, (pp. 24-28). Varna, Bulgaria.

Mehrara, H., Zahedinejad, M., & Pourmohammad, A. (2009). Novel edge detection using BP neural network based on threshold binarization. In Computer and Electrical Engineering. 2, pp. 408-412. Dubai, United Arab Emirates: IEEE.

Mengyang Pu, Y. H. (2022). EDTER: Edge Detection with Transformer. Computer Vision and Pattern Recognition .

Michael, E. (2002). On the origin of the bilateral filter and ways to improve it. IEEE Transactions on image processing , 11 (10), 1141-1151.

Mittal, M., Verma, A., Kaur, I., Kaur, B., Sharma, M., Goyal, L. M., et al. (2019). An Efficient Edge Detection Approach to Provide Better Edge Connectivity for Image Analysis. IEEE Access , 7, 33240-33255.

Moslem, B., & Maghooli, K. (2011). Improving digital image edge detection by fuzzy systems. World Academy of Science, Engineering and Technology World Academy of Science, Engineering and Technology , 5 (9), 76-79.

Nadernejad, E., Sharifzadeh, S., & Hassanpour, H. (2008). Edge detection techniques: evaluations and comparisons. Applied Mathematical Sciences , 2 (31), 1507-1520.

Nalwa, V., & Binford, T. (1986). On detecting edges. IEEE Transactions on Pattern Analysis and Machine Intelligence , PAMI-8, 699-714.

Norman, J., & Ehrlich, S. (1987). Spatial frequency filtering and target identification. Vision Research , 27 (1), 87-96.

Oskoei, M. A., & Hu, H. (2010). A Survey on Edge Detection Methods. University of Essex, School of Computer Science & Electronic Engineering. University of Essex.

Ozkan, K., & Sahin, I. (2015). A novel multi-scale and multi-expert edge detector based on common vector approach. AEU-International Journal of Electronics and Communications , 69 (9), 1272-1281.

Peli, T., & Malah, D. (1982). A study of edge detection algorithms. Computer Graphics and Image Processing , 20 (1), 1-21.

Perona, P., & Malik, J. (1990). Scale-space and edge detection using anisotropic diffusion. IEEE Transactions on Pattern Analysis and Machine Intelligence , 12 (7), 629-639.

Peter, A.-Y., Justice, K. A., & Isaac, K. D. (2016). Performance Analysis of Image Smoothing Techniques on a New Fractional Convolution Mask for Image Edge Detection. Open Journal of Applied Sciences , 6 (7), 478-488.

Pinho, A., & Luis, B. A. (1997). A review on edge detection based on filtering and differentiation. Mathematics .

Prewitt, J. M. (1970). Object enhancement and extraction. Picture Processing and Psychopictorics , 75-149.

Raheja, S., & Kumar, A. (2020). Edge Detection using Ant Colony Optimization under Novel Intensity Mapping Function and Weighted. International Journal of Integrated Engineering , 12 (1), 13-26.

Rakos, D. (2011, January 30). Frei-Chen edge detector. Retrieved November 7, 2022, from https://www.rastergrid.com/: https://www.rastergrid.com/blog/2011/01/frei-chen-edge-detector/

Rangarajan, S. Algorithms for edge detection. Stony Book University.

Rezai-Rad, G., & Aghababaie, M. (2006). Comparison of SUSAN and sobel edge detection in MRI images for feature extraction. International Conference on Information & Communication Technologies. 1, pp. 1103-1107.

Roberts, L. G. (1963). Machine perception of three-dimensional solids. PhD diss., Massachusetts Institute of Technology.

Ruslau, M. F., Pratama, R. A., & Meirista, E. (2019). Edge detection of digital image with different edge types. Journal of Physics: Conference Series. 1569, pp. 1-12. Surabaya, Indonesia: IOP Publishing Ltd.

Saining, X., & Zhuowen, T. (2015). Holistically-nested edge detection. Proceedings of the IEEE international conference on computer vision (pp. 1395-1403). Santiago, Chile..

Santis, A., & Sinisgalli, C. (1999). A Bayesian Approach to Edge Detection in Noisy Images. IEEE Transactions On Circuits And Systems—I: Fundamental Theory And Applications , 46 (6), 686-699.

Sargano, A., Angelov, P., & Habib, Z. (2017). A comprehensive review on handcrafted and learning-based action representation approaches for human activity recognition. Applied Science , 7 (1), 1-10.

Seng, N., Samad, Z., & Nor., N. (2019). A 3-Pixel Fuzzy Mask for Edge Detection. In Proceeding of IOP Conference Series: Materials Science and Engineering, 530, pp. 12-23.

Seo, S., & HunJoo, L. (2015). Pixel based stroke generation for painterly effect using maximum homogeneity neighbor filter. Multimedia Tools and Applications , 74 (10), 3317-3328.

Setayesh, M., Mengjie, Z., & Johnston, M. (2009). A new homogeneity-based approach to edge detection using PSO. Proceeding of 24th International Conference Image and Vision Computing (pp. 231-236). Wellington, New Zealand.

Shrivakshan, G., & Chandrasekar, C. (2012). A comparison of various edge detection techniques used in image processing. International Journal of Computer Science , 9 (5), 272-276.

Smith, S., & Brady, J. (1997). SUSAN - A New Approach to Low Level Image Processing. International Journal of Computer Vision , 23, 45-78.

Sobel, I., & Feldman, G. (1973). A 3×3 isotropic gradient operators for image processing. Pattern classification and scene analysis , 271-272.

Sridhar, S. (2016). Digital Image Processing. Oxford University Publication.

Su, Z. L. (2021). Pixel difference networks for efficient edge detection. IEEE International Conference on Computer, (pp. 5117-5127).

Sun, G., Liu, Q., Liu, Q., Ji, C., & Li, X. (2007). A novel approach for edge detection based on the theory of universal gravity. Pattern Recognition , 40 (10), 2766-2775.

Tariq, N., Hamzah, R. A., NG, T. F., Wang, S. L., & Ibrahim, H. (2021). Quality Assessment Methods to Evaluate the Performance of Edge Detection Algorithms for Digital Image: A Systematic Literature Review. IEEE ACCESS , 9, 87763-87776.

The USC-SIPI Image Database. (n.d.). Retrieved from https://www.sipi.usc.edu/: https://sipi.usc.edu/database/database.php

Tian, J., Weiyu, Y., & Shengli, X. (2008). An ant colony optimization algorithm for image edge detection. Proceeding of IEEE Congress on Evolutionary Computation

(IEEE World Congress on Computational Intelligence) (pp. 751-756). Hong Kong, China.

Verma, O. P., Agrawal, N., & Sharma, S. (2016). "An optimal edge detection using modified artificial bee colony algorithm". 86, pp. 157-168. Proceedings of the National Academy of Sciences, India Section A: Physical Sciences.

Verma, O. P., Hanmandlu, M., Kumar, P., & Srivastava, S. (2009). A novel approach for edge detection using Ant colony optimization and Fuzzy Derivative Technique. IEEE International Advance Computing Conference, (pp. 1206-1212).

Verma, O. P., Hanmandlu, M., Kumar, P., Chhabra, S., & Jindal, A. (2011). A novel bacterial foraging technique for edge detection. Pattern Recognition Letters , 32 (8), 1187-1196.

Verma, O. P., Sharma, R., Kumar, M., & Agrawal, N. (May 2013). An optimal edge detection using gravitational search algorithm. Lecture Notes in software engineering , 1 (2), 148-152.

Vincent, O., & Folorunso, O. (2009). A descriptive algorithm for sobel image edge detection. In Proceeding of Informing Science + IT Education Conference, 40, pp. 97-107.

Vincent, O., & Folorunso, O. (2009). A descriptive algorithm for sobel image edge detection. In Proceedings of Informing Science & IT Education Conference (InSITE), 40, pp. 97-107.

Wang, D., Dapei, T., & Lei, L. (2018). Particle swarm optimization algorithm: an overview. Soft Computing , 22 (2), 387-408.

Wang, Y., Zhao, X., & Huang, K. (2017). Deep crisp boundaries. Proceeding of IEEE conference on computer vision and pattern recognition (pp. 1724-1732). Honolulu, HI, USA.

Wesolkowski, S., Jernigan, M., & Dony, R. (2000). Comparison of color image edge detectors in multiple color spaces. In Proceedings 2000 International Conference on Image Processing (pp. 796-799). Vancouver, BC, Canada: IEEE.

Wibisono, J. K., & Hang, H.-M. (2020). FINED: Fast Inference Network for Edge Detection. Computer Vision and Pattern Recognition .

Wink, A. M., & Roerdink, J. B. (2004). Denoising functional MR images: a comparison of wavelet denoising and Gaussian smoothing. IEEE transactions on medical imaging , 23 (3), 374-387.

Xie, S., & Tu, Z. (2015). "Holistically-nested edge detection". In Proceedings of the IEEE international conference on computer vision (pp. 1395-1403). Santiago, Chile: IEEE.

Xie, S., & Tu, Z. (2017). "Holistically-nested edge detection". International Journal of Computer Vision , 125 (1-3), 3-18.

Xin, G., Chen, K., & Hu, X. (2012). An improved Canny edge detection algorithm for color image. In Proceeding of the IEEE 10th International Conference on Industrial Informatics, (pp. 113-117).

Xu, L., Lu, C., Xu, Y., & Jia, J. (2011). Image smoothing via L_0 gradient minimization. ACM Transactions on Graphics , 30 (6), 1-12.

Yigitbasi, E. D., & Baykan, N. A. (2013). Edge detection using artificial bee colony algorithm (ABC). International Journal of Information and Electronics Engineering , 3 (6), 634-638.

Yirenkyi, P. A., & Appati, J. K. (2016). Performance Analysis of Image Smoothing Techniques on a New Fractional Convolution Mask for Image Edge Detection. Open Journal of Applied Sciences , 6 (7), 478-488.

Yu, Y. C. (2006). A new edge detection approach based on image context analysis. Elsevier Journal of Image and Vision Computing , 24 (10), 1090-1102.

Yu, Y., & Chang, C. (2006). A new edge detection approach based on image context analysis. Journal of Image and Vision Computing , 24, 1090-1102.

Zhang, L., Xiao, M., Ma, J., & Song, H. (2009). Edge detection by adaptive neuro-fuzzy inference system. In Proceeding of 2nd International Congress on Image and Signal Processing (pp. 1-4). Tianjin, China.

Zhang, W., Zhao, Y., Breckon, T., & Chen, L. (2017). Noise robust image edge detection based upon the automatic anisotropic Gaussian kernels. Pattern Recognition , 63, 193-205.

Zheng, L., & He, X. (2004). Edge detection based on modified BP algorithm of ANN. VIP '05: Proceedings of the Pan-Sydney area workshop on Visual information processing , 119-122.

Zheng, Z., Zha, B., Yuan, H., Xuchen, Y., Gao, Y., & Zhang, H. (2020). Adaptive edge detection algorithm based on improved grey prediction model. IEEE Access , 8, 102165-102176.

Zhengguo, L., Jinghong, Z., Zijian, Z., Wei, Y., & Shiqian, W. (2014). Weighted guided image filtering. IEEE Transactions on Image processing , 24 (1), 120-129.

Ziou, D., & Tabbone, S. (1998). Edge Detection Techniques: An Overview. International Journal of Pattern Recognition and Image Analysis (8), 537-559.

Ziqi, X., Xiaoqiang, J., Meijiao, W., & Xiaobing, S. (2021). Edge detection algorithm of medical image based on Canny operator. Journal of Physics: Conference Series (pp. 1-6). IOP Publishing.