

**ÇANKAYA UNIVERSITY  
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES  
COMPUTER ENGINEERING**

**MASTER THESIS**

**EFFECT OF NOISE ON EDGE DETECTION TECHNIQUES**

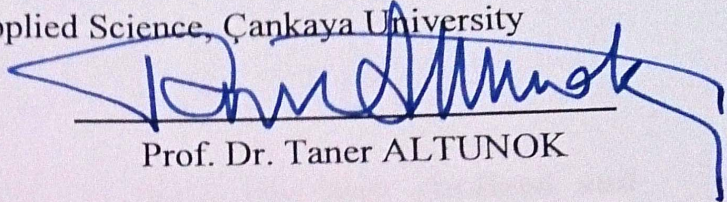
**ALAA MAHMOOD**

**FEBRUARY 2014**

Title of the Thesis: **Effect of Noise on Edge Detection Techniques**

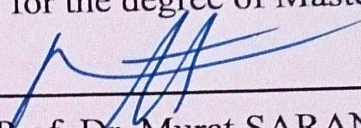
Submitted by: **Alaa MAHMOOD**

Approval of Graduate School of Natural and Applied Science, Çankaya University

  
Prof. Dr. Taner ALTUNOK

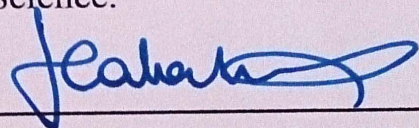
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science

  
Assist. Prof. Dr. Murat SARAN

Acting Head of Department

This is to certify that I have read this thesis and that in my opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

  
Assoc. Prof. Dr. H.Hakan MARAŞ

Supervisor

Examination Date:

21.02.2014

**Examining Committee Members**

Assoc.Prof.Dr.Hadi Hakan MARAŞ

(Çankaya Univ.)

Assist.Prof.Dr.Reza ZARE HASSANPOUR

(Çankaya Univ.)

Assoc.Prof.Dr.Ersin ELBAŞI

(TÜBİTAK)

## STATE OF NON-PLAGIARISM

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: Alaa MAHMOOD

Signature: 

Date: 21.2.2014

## **ABSTRACT**

### **EFFECT OF NOISE ON EDGE DETECTION TECHNIQUES**

MAHMOOD, Alaa

M.Sc., Department of Computer Engineering

Supervisor: Assoc. Prof. Dr. H.Hakan Maraş

February 2014, 108 Pages

The subject of identification edge in images has a wide application in various fields for that it's considered one of the important topics in a digital image processing. There are many algorithms to detect the edge in images, but the performance of these algorithms depends on the type of image, the environment of the image and the threshold value of the edge algorithm. In this thesis five edge detection algorithms were evaluated by using several types of original images, these images were placed in multi-environments (clean, noisy, and de-noised). According to this evaluation results, the best edge detection algorithm and the best threshold value were found in each environment.

**Keywords:** edge detection, noise, noise removal

## ÖZ

### GÜRÜLTÜNÜN KENAR BELİRLEME TEKNİKLERİNE ETKİSİ

Mahmood, Alaa

Yükseklisans, Bilgisayar Mühendisliği Anabilim Dalı

Tez Yöneticisi : Doç. Dr. H.Hakan Maraş

Şubat 2014, 108 sayfa

Görüntülerde kenarların belirlenmesi, değişik çalışma alanlarında geniş yer bulması nedeniyle, sayısal görüntü işlemenin önemli konularından birini oluşturmaktadır. Görüntülerdeki kenarların belirlenmesi için birçok algoritma bulunmaktadır, fakat algoritmanın performansı, görüntünün tipine, bulunduğu ortama ve algoritmada kullanılan eşik değere bağlıdır. Bu tez çalışmasında, orijinal görüntülerin değişik etkilere (temiz, gürültülü, gürültüden arındırılmış) büründürülerek oluşturulan çeşitli formları kullanılarak beş kenar belirleme algoritması değerlendirilmiştir. Bu değerlendirme sonucunda, her ortamda en iyi kenar belirleme algoritması ve en iyi eşik değeri belirlenmiştir.

Anahtar Kelimeler: kenar belirleme, gürültü, gürültüden arındırma

## **ACKNOWLEDGEMENTS**

I would like to express my thanks and appreciation to my thesis advisor Assoc. Prof. Dr. H. Hakan Maraş for his advice and guidance during this study, also special thanks to Assist. Prof. Dr. Reza ZARE HASSANPOUR, who give me a bright spots and special thanks to Assist.Prof. Dr. Emre SERMUTLU Who was always collaborator with me gently.

## LIST OF CONTENTS

<b>ABSTRACT.....</b>	<b>III</b>
<b>ÖZ .....</b>	<b>IV</b>
<b>ACKNOWLEDGEMENTS .....</b>	<b>V</b>
<b>LIST OF CONTENTS .....</b>	<b>VI</b>
<b>LIST OF FIGURES .....</b>	<b>IX</b>
<b>1 INTRODUCTION .....</b>	<b>1</b>
1.1 Background .....	1
1.2 Aim of the Study .....	2
1.3 Literature Review .....	2
1.4 Thesis Overview .....	4
<b>2 EDGE DETECTION TECHNIQUES .....</b>	<b>5</b>
2.1 Edge in an Image .....	5
2.2 Types of Edge.....	5
2.3 Edge Detection in an Image .....	6
2.4 Applications of Edge Detection.....	7
2.5 Introduction to an Edge Detection Methods.....	9
2.5.1 Gradient .....	9
2.5.2 Laplacian.....	11
2.6 Edge Detection Algorithms .....	12
2.6.1 Roberts operator.....	13
2.6.2 Sobel operator .....	14
2.6.3 Prewitt operator.....	15
2.6.4 Laplacian of gaussian operator .....	15
2.6.5 Canny operator.....	17
2.7 Noise.....	20

2.7.1	Gaussian noise .....	20
2.7.2	Salt and pepper noise .....	21
2.7.3	Speckle noise .....	22
2.8	Noisy Image .....	23
2.9	Image Restoration.....	23
2.9.1	Degradation and restoration process in an image .....	24
2.9.2	Image restoration in the presence of noise only spatial filtering .....	25
2.9.2.1	Arithmetic mean filter .....	25
2.9.2.2	Median filter .....	25
2.9.2.3	Alpha trimmed mean filter .....	26
2.9.2.4	Minimum mean square error filtering .....	26
2.10	Algorithms Evaluation .....	27
2.10.1	Normalized correlation.....	27
2.10.2	Peak signal to noise ratio.....	28
<b>3</b>	<b>TOOLS AND METHODS .....</b>	<b>30</b>
3.1	Tools Used in Thesis .....	30
3.1.1	Matlab .....	30
3.1.2	Original images.....	33
3.2	Methodology .....	34
3.2.1	Clean environment.....	34
3.2.2	Noisy environment.....	34
3.2.3	De-nosing environment.....	35
<b>4</b>	<b>PRACTICAL RESULTS .....</b>	<b>39</b>
4.1	Clean Environment Results .....	39
4.2	Noisy Environment Results.....	43
4.2.1	Noisy environment results-gaussian noisy images .....	43
4.2.2	Noisy environment results-salt and pepper noisy images.....	48
4.2.3	Noisy environment results-speckle noisy images.....	53
4.3	De-noising Environment .....	58
4.3.1	Results of noise removal algorithms.....	58



4.3.1.1	Results of noise removal algorithms-gaussian noise.....	58
4.3.1.2	Results of noise removal algorithms-salt and pepper noise .....	62
4.3.1.3	Results of noise removal algorithms-speckle noise .....	66
4.3.2	De-noising environment results .....	70
4.3.2.1	De-nosing environment results-gaussian de-noised images.....	70
4.3.2.2	De-nosing environment results-salt and pepper de-noised images .....	75
4.3.2.3	De-nosing environment results-speckle de-noised image .....	80
<b>5</b>	<b>DISCUSSIONS AND CONCLUSIONS.....</b>	<b>85</b>
5.1	Discussions .....	85
5.2	Conclusions .....	88
5.3	Future Work .....	89
<b>6</b>	<b>REFERENCES .....</b>	<b>90</b>
	<b>Curriculum vitae .....</b>	<b>95</b>

## LIST OF FIGURES

Figure 2.1: Types of Edge.....	6
Figure 2.2: Applications of Edge Detection System.....	7
Figure 2.3: $F(X)$ and $F'(X)$ .....	9
Figure 2.4: $f'x$ and $f''x$ .....	12
Figure 2.5: Robert's Kernels.....	13
Figure 2.6: Sobel Operator.....	14
Figure 2.7: Prewitt Operator .....	15
Figure 2.8: Laplacian Kernels.....	15
Figure 2.9: Approximating Kernel of LOG .....	16
Figure 2.10: 5x5 Sub-Image.....	19
Figure 2.11: Possible Directions for a Pixel in an Image.....	19
Figure 2.12: PDF of the Gaussian Noise.....	21
Figure 2.13: PDF of the Salt and Pepper Noise .....	22
Figure 2.14: Degradation / Restoration Process.....	24
Figure 3.1: Matlab Structure .....	30
Figure 3.2: Original Images .....	33
Figure 3.3 : Clean Environment Scheme .....	35
Figure 3.4: Noisy Environment Scheme .....	36
Figure 3.5: Noise Removal Scheme.....	37
Figure 3.6: De-Noising Environment Scheme .....	38
Figure 4.1: Threshold Values of Edge Detection Algorithms in Clean Environment .....	40
Figure 4.2: Edge Detection Images of Binary Image in Clean Environment .....	40
Figure 4.3: Edge Detection Images of Graphic Image in Clean Environment .....	41
Figure 4.4: Edge Detection Images of High-Freq Image in Clean Environment.....	41

Figure 4.5: Edge Detection Images of Low-Freq Image in Clean Environment .....	42
Figure 4.6: Edge Detection Images of Median-Freq Image in Clean Environment .....	42
Figure 4.7: Edge Detection Images of Texture Image in Clean Environment.....	43
Figure 4.8: Threshold Values of Edge Detection Algorithms in Noisy Environment- Gaussian Noise .....	44
Figure 4.9: Correlation Values of Noisy Environment-Gaussian Noise.....	44
Figure 4.10: Edge Detection Images of Binary Image in Noisy Environment-Gaussian Noise .....	45
Figure 4.11: Edge Detection Images of Graphic Image in Noisy Environment-Gaussian Noise .....	46
Figure 4.12: Edge Detection Images of High-Freq Image in Noisy Environment- Gaussian Noise .....	46
Figure 4.13: Edge Detection Images of Low-Freq Image in Noisy Environment- Gaussian Noise .....	47
Figure 4.14: Edge Detection Images of Median-Freq Image in Noisy Environment- Gaussian Noise .....	47
Figure 4.15: Edge Detection Images of Texture Image in Noisy Environment-Gaussian Noise .....	48
Figure 4.16: Threshold Values of Edge Algorithms in Noisy Environment-Salt and Pepper Noise .....	49
Figure 4.17: Correlation Values of Noisy Environment-Salt and Pepper Noise .....	49
Figure 4.18: Edge Detection Images of Binary Image in Noisy Environment-Salt and Pepper Noise .....	50
Figure 4.19: Edge Detection Images of Graphic Image in Noisy Environment-Salt and Pepper Noise .....	50
Figure 4.20: Edge Detection Images of High-Freq Image in Noisy Environment-Salt and Pepper Noise .....	51
Figure 4.21: Edge Detection Images of Low-Freq Image in Noisy Environment-Salt and Pepper Noise .....	51
Figure 4.22: Edge Detection Images of Median-Freq Image in Noisy Environment-Salt and Pepper Noise .....	52
Figure 4.23: Edge Detection Images of Texture-Freq Image in Noisy Environment-Salt and Pepper Noise .....	52
Figure 4.24: Threshold Values of Edge Detection Algorithms in Noisy Environment- Speckle Noise .....	53

Figure 4.25: Correlation Values of Noisy Environment-Speckle Noise.....	54
Figure 4.26: Edge Detection Images of Binary Image in Noisy Environment-Speckle Noise .....	55
Figure 4.27: Edge Detection Images of Graphic Image in Noisy Environment-Speckle Noise .....	55
Figure 4.28: Edge Detection Images of High-Freq Image in Noisy Environment-Speckle Noise .....	56
Figure 4.29: Edge Detection Images of Low-Freq Image in Noisy Environment-Speckle Noise .....	56
Figure 4.30: Edge Detection Images of Median-Freq Image in Noisy Environment- Speckle Noise .....	57
Figure 4.31: Edge Detection Images of Texture Image in Noisy Environment-Speckle Noise .....	57
Figure 4.32: PSNR Values between the Original Images and the Noisy Images .....	58
Figure 4.33: PSNR values between the original images and the de-noising images- Gaussian noise .....	59
Figure 4.34: De-Noising Images of Binary Image-Gaussian Noise .....	59
Figure 4.35: De-Noising Images of Graphic Image-Gaussian Noise .....	60
Figure 4.36: De-Noising Images of High-Freq Image-Gaussian Noise.....	60
Figure 4.37: De-Noising Images of Low-Freq Image-Gaussian Noise .....	61
Figure 4.38: De-Noising Images of Median-Freq Image-Gaussian Noise .....	61
Figure 4.39: De-Noising Images of Texture Image-Gaussian Noise .....	62
Figure 4.40: PSNR between the Original Images and the De-Noising Images-Salt and Pepper Noise .....	63
Figure 4.41: De-Noising Images of Binary Image-Salt and Pepper Noise .....	63
Figure 4.42: De-Noising Images of Graphic Image-Salt and Pepper Noise .....	64
Figure 4.43: De-Noising Images of High-Freq Image-Salt and Pepper Noise .....	64
Figure 4.44: De-Noising Images of Low-Freq Image-Salt and Pepper Noise .....	65
Figure 4.45: De-Noising Images of Median-Freq Image-Salt and Pepper Noise .....	65
Figure 4.46: De-Noising Images of Texture Image-Salt and Pepper Noise .....	66
Figure 4.47: PSNR Values between the Original Images and the De-Noising Images- Speckle Noise .....	67
Figure 4.48: De-Noising Images of Binary Image-Speckle Noise .....	67

Figure 4.49: De-Noising Images of Graphic Image-Speckle Noise .....	68
Figure 4.50: De-Noising Images of High-Freq Image-Speckle Noise.....	68
Figure 4.51: De-Noising Images of Low-Freq Image-Speckle Noise .....	69
Figure 4.52: De-Noising Images of Median-Freq Image-Speckle Noise .....	69
Figure 4.53: De-Noising Images of Texture Image-Speckle Noise.....	70
Figure 4.54: Threshold Values of Edge Detection Algorithms in De-Noising Environment-Gaussian Noise .....	71
Figure 4.55: Correlation Values of De-Nosing Environment-Gaussian Noise.....	71
Figure 4.56: Edge Detection Images of Binary Image in De-Noising Environment- Gaussian Noise .....	72
Figure 4.57: Edge Detection Images of Graphic Image in De-Noising Environment- Gaussian Noise .....	73
Figure 4.58: Edge Detection Images of High-Freq Image in De-Noising Environment- Gaussian Noise .....	73
Figure 4.59: Edge Detection Images of Low-Freq Image in De-Noising Environment- Gaussian Noise .....	74
Figure 4.60: Edge Detection Images of Median-Freq Image in De-Noising Environment- Gaussian Noise .....	74
Figure 4.61: Edge Detection Images of Texture Image in De-Noising Environment- Gaussian Noise .....	75
Figure 4.62: Threshold Values of Edge Algorithms in De-Noising Environment-Salt and Pepper Noise .....	76
Figure 4.63: Correlation Values of De-Nosing Environment-Salt and Pepper Noise .....	76
Figure 4.64: Edge Detection Images of Binary Image in De-Noising Environment-Salt and Pepper Noise .....	77
Figure 4.65: Edge Detection Images of Graphic Image in De-Noising Environment-Salt and Pepper Noise .....	78
Figure 4.66: Edge Images of High-Freq Image in De-Noising Environment-Salt and Pepper Noise .....	78
Figure 4.67: Edge Images of Low-Freq Image in De-Noising Environment-Salt and Pepper Noise .....	79
Figure 4.68: Edge Images of Median-Freq Image in De-Noising Environment-Salt and Pepper Noise .....	79

Figure 4.69: Edge Detection Images of Texture Image in De-Noising Environment-Salt and Pepper Noise .....80

Figure 4.70: Threshold Values of Edge Detection Algorithms in De-Noising Environment-Speckle Noise .....80

Figure 4.71: Correlation Values of De-Nosing Environment-Speckle Noise.....81

Figure 4.72: Edge Detection Images of Binary Image in De-Noising Environment-Speckle Noise .....81

Figure 4.73: Edge Detection Images of Graphic Image in De-Noising Environment-Speckle Noise .....82

Figure 4.74: Edge Detection Images of High-Freq Image in De-Noising Environment-Speckle Noise .....82

Figure 4.75: Edge Detection Images of Low-Freq Image in De-Noising Environment-Speckle Noise .....83

Figure 4.76: Edge Detection Images of Median-Freq Image in De-Noising Environment-Speckle Noise .....83

Figure 4.77: Edge Detection Images of Texture Image in De-Noising Environment-Speckle Noise .....84

## CHAPTER 1

### INTRODUCTION

#### 1.1 Background

The edges of the image can be defined as the pixels that occur in the boundary and connect two different areas [1, 2, 3], the edges of an image always refer to a high frequency in the image. Edge detection is for checking the edges in images, for that it is considered a kind of image segmentation techniques [3]. The edge detection system goal is to convert the data of the image to reduce the size of data which is going to be handled [4], this detection helps for compression of data, segmentation of images, and images matching, like image reconstruction. Generally, the identification of edge in images is considered as an important topic in image processing, because of its great importance in various applications like pattern recognition and scene analysis. Too many researches were made in this area to achieve the ability to identify edges with high accuracy, especially in aerial images that are captured from the satellites [5]. The accuracy of the detection depends on the efficiency of the algorithm used, so studies and researches have been conducted to discover new methods in identifying the edges and to develop the existing methods to get a convenient way and good results in various applications. The reason for the determination of the edge is because of the representation of the components by the edge is more understandable than an ordinary image, furthermore the human ability to distinguish between components is better by using lines [6]. There are many methods to find the edges in images. Calculating the differentiation of an image is considered one of the most common methods of edge detection system. The first- and second-order derivative of an image are obtained using the gradient and the Laplacian respectively. It is known that each edge algorithm has a threshold value, choosing the best algorithm and the best threshold value depends on the type of image and the

environment of the image, for that reason it becomes necessary to find evaluation for these Algorithms by using several types of images in multi environments.

## **1.2 Aim of the Study**

The aim of this study, is to evaluate five algorithms of edge detection which are Roberts, Sobel, Prewitt, LOG, and Canny in multi environments (clean, noisy and de-noising) by using several types of original images (binary image, graphic image, high frequency image, low frequency image, median frequency image, and texture image) and then determine the best algorithm. In noisy environment the following noises was used Gaussian, salt and pepper and speckle, and the following noise removal was used in the de-noising environment mean, median, Wiener and Alpha trimmed mean filters. It's known that for every edge detection algorithm has a threshold value, if the current pixel value is less than the defined threshold in strength, it will be considered an edge pixel. The change rate of the threshold value in all environments is also explained through this study.

## **1.3 Literature Review**

The subject of identifying the edge is considered one of the topics that received a big attention from researchers. Hence, many algorithms in this area are appeared. The first algorithm in edge detection “Roberts” is submitted in 1965. This algorithm depended on the point and its neighboring points. Then there was an evolution in the concepts of the determination of the edge in 1970. Two methods are proposed by Prewitt and Sobel [7]. The methods depended on the edges of the points that have high values in covariance or regression. In 1980, a new method appeared (Marr & Hildreth), which actually makes smoothing to the image before edge detection. In 1983 Canny method appeared, Canny used smoothing operation and he was interested in treatment of the weak edges [4]. After that, many methods appeared which focused on the studying of the edge detection methods and comparing them such as the research presented by Vliet, Young & Beckers. They have shown that the Laplace operator is effective and flexible in the detection of



one pixel thick edges [8]. Then Ziou & Tabbone explained the mutual influence between edges and the existing edge detectors [9]. Heath, Sarkar, Sanocki & Bowyer, have chosen four edge detection methods and applied on eight images and they also conducted a statistical study on the performance of each method [10]. Min C. Shin, Dmitry B. Goldgof, Kevin W. Bowyer and Savvas Nikiforou have shown that the Canny algorithm had a good performance and reliability with all image sequences for motion [11]. Ruman and Soahel calculated the execution of the Prewitt algorithm for noisy images Gaussian, salt and pepper, speckle and they concluded that this algorithm didn't work with the noisy images [12]. Ruman and Himanshy worked on many classical edge detection algorithms and they were able to show the advantages and disadvantages of these classical edge detection algorithms and They concluded that the Canny algorithm is more complex to compute and perform compared to other operators, and this algorithm works better than all algorithms under almost all environments [13]. Nandheta and Etal worked on wavelet and they showed that these algorithms gives better results if the noise was removed from the source images [14]. Wnag luo worked on the colony images and he reached a result that the classical algorithms did not operate properly with the colony images, but the Canny edge detector worked best both visually and quantitatively [15]. Sara, Ehsan and Hamid have shown that the Boolean edge detector performs surprisingly similarly to the Canny edge detector even though they both take drastically different approaches [16]. Shashidhar and Roshan worked on the Haar based on the Prewitt algorithm and they concluded that the canny algorithm is better than the Haar based on the Prewitt algorithm and also the canny algorithm that depends on parameters, that are adjustable, operate in a better way in noisy and blur images [17]. LiBin and Mehdi Samiei have shown that Canny algorithm is not liable to interfering noise and it has the power to detect weak edges that are true. And this algorithm can be called an optimal algorithm [56]. Shrivakshan and Chandrasekar have shown that Canny edge algorithm works well in all environments by changing its parameters and this algorithm is most costly in comparing to Sobel, Prewitt and Roberts [57]. Additionally books [18, 19, 20, 21, 22] are also emphasized the important resources in the edge detection subject.

## **1.4 Thesis Overview**

This thesis is organized as follows. Chapter two presents, the edge detection techniques. In Chapter three, the tools and methods are discussed. Chapter four shows the practical results conducted in this study. Finally chapter five contains the discussions and conclusions.

## CHAPTER 2

### EDGE DETECTION TECHNIQUES

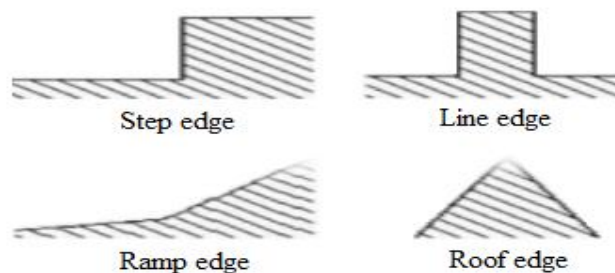
This chapter contains an explanation in detail about the edge, types of edge, edge applications, edge detection, edge detection methods, edge detection algorithms, noise, types of noise, noisy image, image restoration and image filters.

#### 2.1 Edge in an Image

An edge is generally defined as the boundary where a considerable change happens in some sides of the digital image. Changes in the physical aspect manifest themselves in a variety of ways, such as changes in intensity, color or texture. The variations in physical aspect can be caused by interruption in depth, direction of the surface and differences in lighting [23]. Edge is an information in an image that can be found by looking at the relationship between a pixel and its neighbors. If a pixel's gray-level value is similar to those around it, probably there is no edge at that point. However, if a pixel has neighbors with widely varying gray levels, it may represent an edge point. The edge is considered as a high frequency in the image, therefore, it is a discontinuity in the gray-level values.

#### 2.2 Types of Edge

- Step edge
- Ramp edge
- Line edge
- Roof edge



**Figure 2.1:** Types of Edge

Step edge: the image intensity abruptly changes from one value to one side of the discontinuity to a different value on the opposite side.

Ramp edge: a step edge where the intensity change is not instantaneous, but occurs over a finite distance.

Line edge: the image intensity abruptly changes value, but then returns to the starting value within some short distance.

Roof edge: a ridge edge where the intensity change is not instantaneous, but occur over a finite distance (usually generated by the intersection of surfaces).

### **2.3 Edge Detection in an Image**

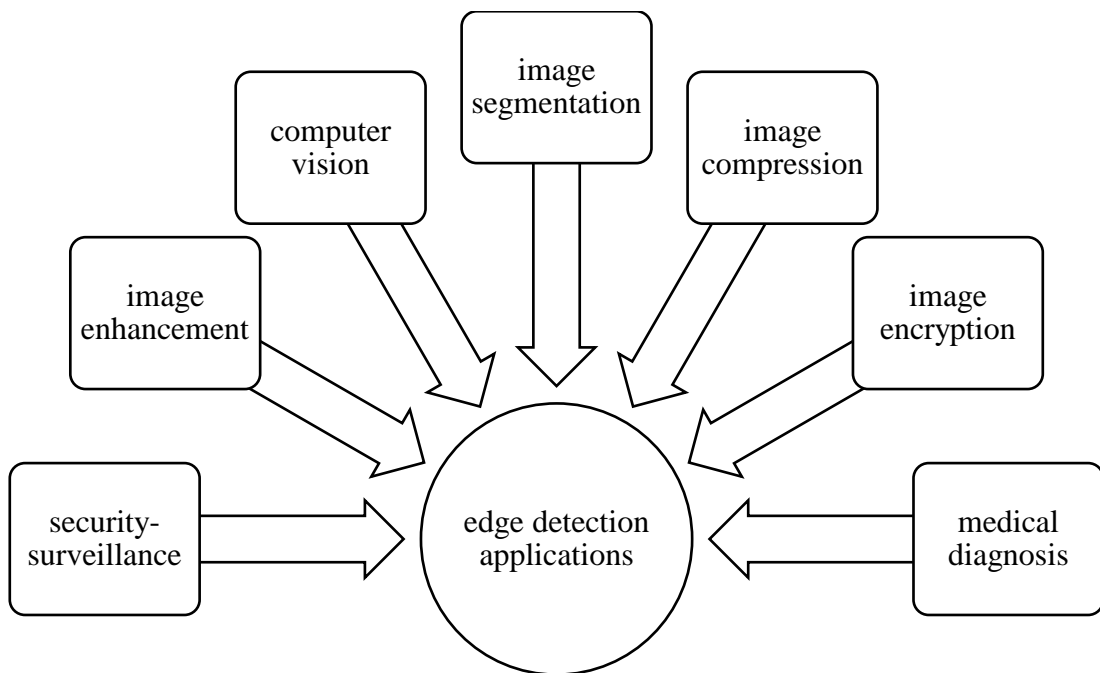
Edge detection is a type of image segmentation methods which decides the presence of an edge in an image [24]. It is the process of characterizing the intensity changes in an image in terms of the physical processes that have originated them. Edge detection is considered a high pass filter that can be applied to extract the edge points in an image. An edge detection system is dependent on a variety of factors like the application or context in which it is used, noise in the source image and level of edge details required. The edge detection system goal is to convert the data of the image to reduce the size of data which is going to be handled [25].

There are several steps that must be characterized by an edge detection system to get a successful result:

- Smoothing: Removing noise from the source image, without effecting on the true edges.
- Enhancement: Emphasizing pixels which have an important change in the local intensity.
- Detection: identify true edges.
- Localization: Locate the edge accurately or determine the exact location of an edge, estimate edge orientation.

## 2.4 Applications of Edge Detection

An edge detection system has a wide variety of applications, as shown below in the following figure:



**Figure 2.2:** Applications of Edge Detection System

Segmentation: it refers to the procedure of dividing an image into several parts such that pixels in each segment share certain visual characteristics. The result of image segmentation is a set of contours or segments that collectively cover the entire image. Image segmentation is used in a variety of fields like satellite imagery, environmental studies, face recognition, and medical imaging [26].

Compression: in order to reduce the amount of important data that are required to be stored while retaining most of the image information, an image must be represented by its edges. Transmitting of the edge pixels in an image or multimedia would result in a great deal of compression and there exist very reliable algorithms to reconstruct the entire image (or sequence of images) based on the edge map. Therefore, edge detection is applicable in image compression techniques.

Enhancement: An edge detection filter can also be used to improve the appearance of blurred or anti-aliased video streams. The fusing of edges to a blurry image would give perception of an enhanced (sharper) output and this concept is the basis of some image enhancement algorithms that utilize edge detectors.

Encryption: Edge detection techniques are applied in the field of image encryption and multimedia communication as edge information can be manipulated to carry hidden data in it. This data would be secure compared to encryption methods were less significant portions of the image store the secret data and they are lost when the image (multimedia) is stored in common compressed forms [58].

Computer Vision: Edge detection is helpful in various applications of computer vision like in simple object counting in a speeding production lane. An edge detection module forms the front end of most of these vision systems. Here it is important that the edge detection module operates in real time and has tolerance to noise [58].

Security: In the field of security and surveillance several methods exist to identify and verify humans based on their biometrics. Face and its features form one such important biometric based on which humans can be identified and verified. Face recognition based on edge maps is one of the popular methods [58].

## 2.5 Introduction to an Edge Detection Methods

One of the most common techniques for edge detection is to find the differentiation of an image. The first-order-derivative in an image is obtained using the gradient, and the second-order-derivative is produced using the Laplacian. Before talking about the edge detection algorithms, a brief explanation must be given about the gradient and Laplacian, which are based on the principle of differentiation.

### 2.5.1 Gradient

The gradient detects the edges by looking for the maximum and minimum in the first derivative of the image. From figure 2.3, it can be seen that, at the point of greatest slope in  $f(x)$  the first derivative  $f'(x)$  has a maximum value.

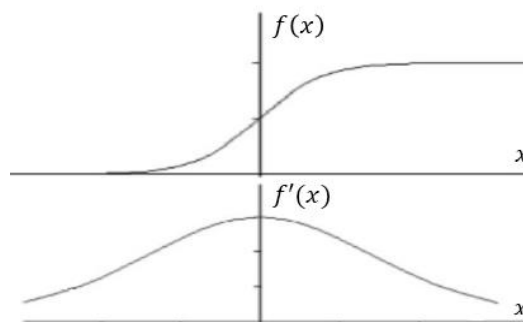


Figure 2.3:  $F(X)$  and  $F'(X)$

The variation in space of any quantity can be represented (e.g. graphically) by a slope. The gradient represents the steepness and direction of that slope. Mathematically, the gradient of a two-variable function (here the image intensity function) at each image point is a two dimensional vector with the components given by the derivatives in the

horizontal and vertical directions as it is shown below:

Gradient:

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

(2.1)

Gradient magnitude:

$$|\nabla f| = \sqrt{\left(\frac{\partial f(x,y)}{\partial x}\right)^2 + \left(\frac{\partial f(x,y)}{\partial y}\right)^2} \quad (2.2)$$

Gradient direction:

$$\theta = \tan^{-1}\left(\frac{\partial f(x,y)}{\partial x} / \frac{\partial f(x,y)}{\partial y}\right) \quad (2.3)$$

Now let's review the partial derivatives. When taking the partial derivative of  $f$  that is related to  $x$ , it will determine how rapidly the image intensity changes as  $x$  changes. For a continuous function,  $f(x, y)$  the partial derivative of this function that is related to  $x$  will be as in equation 2.4.

$$\frac{\partial f(x,y)}{\partial x} = \lim_{\Delta x \rightarrow 0} \frac{f(x+\Delta x, y) - f(x, y)}{\Delta x}$$

(2.4)

In the discrete case, only the differences can be taken at one pixel intervals. So the difference can be taken either between  $f(x, y)$  and the pixel before it, or the pixel after it. Therefore it will be as in equation 2.5.

$$\frac{\partial f(x,y)}{\partial x} \approx \frac{f(x+1,y) - f(x,y)}{2} \quad (2.5)$$

By similar reasoning,  $\frac{\partial f(x,y)}{\partial y}$  can be computed in equation 2.6.



$$\frac{\partial f(x,y)}{\partial y} \approx \frac{f(x,y+1)-f(x,y)}{2} \quad (2.6)$$

By combining the equations (2.5) and (2.6) a complete method for computing the image gradient can be obtained. The approximation of the gradient to a discrete two dimensional function for vertical (equation 2.8) and horizontal (equation 2.7) direction is given below:

Horizontal - differentiation approximation

$$G_x = \frac{\partial f(x,y)}{\partial x} = f(x+1, y) - f(x, y) \quad (2.7)$$

Vertical - differentiation approximation

$$G_y = \frac{\partial f(x,y)}{\partial y} = f(x, y+1) - f(x, y) \quad (2.8)$$

Depending on the above equations (2.7) and (2.8) the approximation of the gradient, gradient magnitude and the gradient direction to a discrete two dimensional function will be shown in the following equations, 2.9, 2.10 and 2.11.

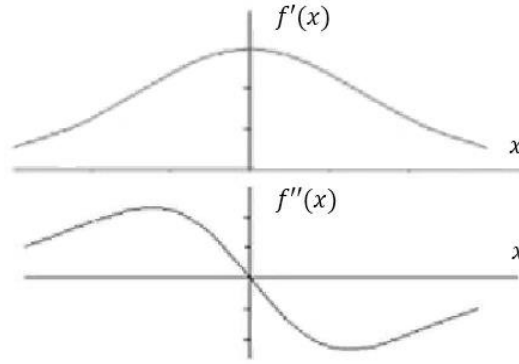
$$G = \begin{bmatrix} G_x \\ G_y \end{bmatrix} \quad (2.9)$$

$$|G| = |G_x| + |G_y| \quad (2.10)$$

$$\theta = \tan^{-1} \left( \frac{G_y}{G_x} \right) \quad (2.11)$$

### 2.5.2 Laplacian

This method looks for zero in the second derivative of the image in order to locate the edge of the input image. From figure 2.4, it can be seen that, when the first derivative  $f'(x)$  has a maximum value, the second derivative  $f''(x)$  will be having a zero crossing.



**Figure 2.4:**  $f'(x)$  and  $f''(x)$

The Laplace operator in mathematics is a 2<sup>nd</sup> differential operator given by the divergence ( $\nabla \cdot$ ) of the gradient ( $\nabla f$ ). The symbols  $\nabla \cdot \nabla$ ,  $\nabla^2$  or  $\Delta$  stand for the Laplacian. So for a function  $f$  that can be derived twice, the Laplacian of this function is presented by equation 2.12

$$\Delta f = \nabla^2 f = \nabla \cdot \nabla f \quad (2.12)$$

The Laplacian of a continuous two dimensional function will be as follows in equation 2.13 below:

$$\Delta f(x, y) = \frac{\partial^2 f(x, y)}{\partial x^2} + \frac{\partial^2 f(x, y)}{\partial y^2} \quad (2.13)$$

The approximation of a Laplacian operator for a discrete two dimensional function is given by equation 2.14 below:

$$\Delta f(x, y) \approx f(x + 1, y) + f(x - 1, y) + f(x, y + 1) + f(x, y - 1) - 4f(x, y) \quad (2.14)$$

## 2.6 Edge Detection Algorithms

Edge detection is an active area of research where several algorithms are being developed to suit different purposes and applications. It is a proven fact that not all edge detection algorithms are suitable for all types of applications [27]. Still, there are popular

edge detection algorithms that form a broad basis for developing algorithms suitable for specific purposes [28]. This thesis considers the following five popular edge detection algorithms:

1. Roberts.
2. Sobel.
3. Prewitt.
4. Laplacian of Gaussian (LOG).
5. Canny.

Roberts, Sobels and Prewitts algorithms used the first derivative, LOG algorithm used the second order derivative, Canny algorithm is an optimal edge detection.

### 2.6.1 Roberts operator

It is a nonlinear and first-order operator [29], in which the edge is found by using the partial derivative. This operator depends in its operation on finding an approximation between two neighboring diagonally pixels, of the gradient amplitude to detect edges. This algorithm is simple and quick to be performed and computed. The operator consists of two kernels and the size of each one is  $2 \times 2$ , both kernels are completely similar but the only difference between them is  $90^\circ$  [29]. Robert's edge detector is based on a small diagonal convolution kernel which is shown in Figure 2.5. It is based on the principle that the edges extracted should be well defined with minimal background noise and the intensity of the edges should correspond as closely as possible to what humans would perceive.

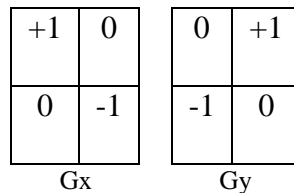


Figure 2.5: Robert's Kernels

The Robert's convolution kernel is designed to be more sensitive towards diagonal edges. The Robert's kernel is based on equations 2.15 and 2.16. Each kernel when applied on the image detects the edges along corresponding diagonal directions.

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

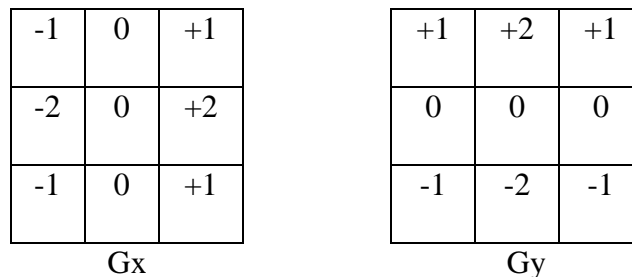
(2.15)

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

(2.16)

### 2.6.2 Sobel operator

This operator is nonlinear and consisting of 3x3 two convolution kernels as it is shown in Figure 2.6. The first one gives maximum response of the vertical edge, and the second one gives maximum response of the horizontal edge [30]. Both kernels have the same values but the only difference between them is the 90° rotation. This operator is easy to obtain in space and it is somehow sensitive to noise. It gives truer information about edge direction, however, the disadvantages of this operator is that it extracts false edges that have rough edge width.



**Figure 2.6:** Sobel Operator

As long as the two kernels are separate, so it is possible to pass each kernel on the source image to get separately results of the gradient in the vertical and horizontal edges. The two kernels Gx and Gy can be merged at the same time to compute the gradient magnitude or the gradient direction [31], as it is predefined in equations (2.9) and (2.10).

### 2.6.3 Prewitt operator

Prewitt operator [32], it is a nonlinear operator consists of a pair of 3x3, it is similar to the Sobel operator and it is used for detecting vertical and horizontal edges in images. The difference between Sobel and Prewitt is the kernel's values. Figure 2.7 shows the Prewitt kernels.

-1	0	+1
-1	0	+1
-1	0	+1

G<sub>x</sub>

+1	+1	+1
0	0	0
-1	-1	-1

G<sub>y</sub>

**Figure 2.7:** Prewitt Operator

### 2.6.4 Laplacian of gaussian operator

The Laplace operator or Laplacian is considered a measure of the second derivative of an image (two dimensional function). This operator is often applied to an image that has first been smoothed with Gaussian approximating smoothing filter in order to decrease its sensitivity to noise. The operator normally takes a single gray level image as input and produces another gray level image as output. Since the input image is represented as a set of discrete pixels, a discrete convolution kernel that can give an approximation to the second derivative in the definition of the Laplacian must be found [32]. Three commonly used small kernels are shown in following Figure.

0	1	0
1	-4	1
0	1	0

-1	2	-1
2	-4	2
-1	2	-1

1	1	1
1	-8	1
1	1	1

**Figure 2.8:** Laplacian Kernels

These laplacian operators can be affected by noise very easily because they make an approximation of the second derivative on the image. Before applying the Laplacian operator, the image is often smoothed by Gaussian filter in order to avoid this sensitivity. This pre-processing step reduces noise before the step of image differentiation. In order to achieve the required result the Gaussian smoothing filter must be convolved with the Laplacian filter, and then convolve this hybrid filter with the image.

It is known that the Laplacian and Gaussian kernels are normally smaller than the source image, and by recalculating in advance then the result would be one kernel only which means that one convolution is required to be applied on an image [33]. The continuous two dimensional function of the Laplacian of Gaussian that have zero mean value, with  $\sigma$  (Gaussian standard deviation) has a equation as follows [34]:

$$Log(x, y) = -1/\pi\sigma^4 \left[ 1 - \left( \frac{x^2+y^2}{2\sigma^2} \right) \right] e^{-\frac{x^2+y^2}{2\sigma^2}} \quad (2.17)$$

The following Figure shows the discrete approximation kernel of the LOG function, at  $\sigma = 1.4$ .

0	1	1	2	2	2	1	1	0
1	2	4	5	5	5	4	2	1
1	4	5	3	0	3	5	4	1
2	5	3	-12	-24	-12	3	5	2
2	5	0	-24	-40	-24	0	5	2
2	5	3	-12	-24	-12	3	5	2
1	4	5	3	0	3	5	4	1
1	2	4	5	5	5	4	2	1
0	1	1	2	2	2	1	1	0

**Figure 2.9:** Approximating Kernel of LOG

Note for  $\sigma < 0.5$  pixels, the LOG kernel becomes same as the simple Laplacian kernels that shown in figure 2.8 [35, 36].

### 2.6.5 Canny operator

This algorithm is considered as an optimal edge detector. Canny was a successful engineer and he implemented his goals which are presented in his published work [37]. Canny followed many criteria to enhance the current algorithms of edge detection. The first criterion is the error with the low rate which means the detection is a full detection and the reaction to a non edge pixels is zero. The second criterion is that the edge points are extracted in an appropriate way, which means that the similarity between the pixels of the edge that is found by this algorithm and the ideal edge is high similarity. In the third criterion there is only one reaction to a single edge. The benefit from the third criterion is the single response whereas the first and the second criteria couldn't remove the occurrence of multiple reactions to an edge. According to the above mentioned criteria, this edge detection algorithm firstly makes smoothing to the image to get rid of the noise. After that it computes the gradient of the input image of the highlighted areas, then goes along these areas and eliminates any pixel which doesn't reach the maximum. The gradient is now decreased by hysteresis which is used to go to the remaining pixels that weren't eliminated. Two thresholds are used by the hysteresis, if the gradient magnitude of a pixel value is below the first threshold, it will be considered as a non edge whereas if it is between the two thresholds it will be considered an edge and if it is greater than the high threshold it will be an edge if there is a connection between this pixel and the pixel that its value is between the two thresholds. Canny algorithm can be implemented by following the next six steps:

#### **Step 1:-**

Before the edge extraction operation, the noise of the original image must be removed, Gaussian filter can be used in this step and it can be computed by applying a simple mask through the convolution method. A convolution kernel is normally smaller than the input image. Then the kernel is passed on the source image, handling a group of pixels at the same time. When the size of kernel is big, the localization error of the detected edges a little increase.

### Step 2:-

This step is to compute the edge strength depending on gradient magnitude that can be obtained by using Sobel operator as in Figure 2.6. The Sobel operator [31] uses a pair of 3x3 convolution masks, one estimating the gradient in the x-direction (columns) and the other estimating the gradient in the y-direction (rows). Then the magnitude of the gradient is approximated, as it is predefined in equation (2.10).

### Step 3:-

In this step by finding the gradient in the  $x$  and  $y$  directions, the direction of an edge can be computed. And when the sum of  $(x)$  is equal to zero an error will be generated. So it must be taken into consideration that whenever this error is generated, a restriction must be set in the code. If the gradient in the  $x$ -direction and  $y$ -direction are equal to zero then the edge direction will be  $0^\circ$ , whereas if the gradient in the direction of  $(x)$  is zero but the gradient in the direction of  $(y)$  is not zero then the edge direction will be  $90^\circ$ . The equation (2.11) shows the edge direction.

### Step 4:-

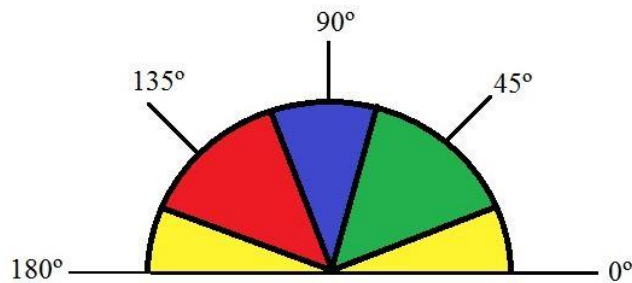
The direction of the edges is known in this step, now the direction of each edge will be connected to a direction which can be examined in the source image. Consider the pixel “a” in sub-image with size 5x5 as follows:

X	x	x	x	x
X	x	x	x	x
X	x	a	x	x
X	x	x	x	x
X	x	x	x	x



**Figure 2.10:** 5x5 Sub-Image

The pixel “a” has four probable directions when depicting the pixels that surround the pixel “a”:  $0^\circ$  in horizontal,  $45^\circ$  in the diagonal that is along the positive side,  $90^\circ$  in vertical, or  $135^\circ$  in the diagonal that is along the negative side. Therefore the direction of the edge will be one of these probable directions according to the closest direction. By looking at the following semicircle that is divided into five regions, it is clear that when the edge direction is between  $0^\circ$  and  $22.5^\circ$  or between  $157.5^\circ$  and  $180^\circ$  (yellow rang), it will be set to  $0^\circ$ ; and when the edge direction is between  $22.5^\circ$  and  $67.5^\circ$  (green range), it will be set to  $45^\circ$ ; whereas if the edge direction is between  $67.5^\circ$  and  $112.5^\circ$  (blue range), it will be set to  $90^\circ$ ; finally when the edge direction is between  $112.5^\circ$  and  $157.5^\circ$  (red range), it will be set to  $135^\circ$ .



**Figure 2.11:** Possible Directions for a Pixel in an Image

#### **Step 5:-**

In this step non-maximum suppression will be applied. This method is used to trace along the edge in the edge direction and block any pixel value that is not an edge pixel. By doing this a thin line will be given in the output image.

#### **Step 6:-**

In this final step hysteresis [38] is used to eliminate streaking which is trying to breaking up the edge contour that is fluctuating by the edge operator. Hysteresis uses two thresholds, one of them is high and the other is low. And if the pixel value is between

the two thresholds it will be an edge pixel but if this value is less than the low threshold it will be blocked and if it has a value greater than the high threshold and has a connection with a pixel that has value between the two thresholds also it will be considered an edge pixel.

## 2.7 Noise

Photographic technologies often is not ideal, each image will suffer from distortion. Each Photographic system has the accuracy of discrimination and speed. The limited accuracy of discrimination and limited speed in the Photographic system often lead to the loss of image information. One of the important factors which affect the quality of an edge is the amount of noise present in the original image before edge extraction. Generally, noise degrades the nature of the source image, and affects the output of the edge detection algorithms, because the noise and the edges contain high frequencies. Not all edge extraction algorithms perform equally well in all kinds of noises [39].

Certain algorithms perform better than others under particular types of noises. In general the noise can be defined as the disturbance tends to intervene in the natural processes of the device or system [40].

### 2.7.1 Gaussian noise

It is also called amplifier noise or normal noise, which is statistical noise that has its probability density function equal to that of the normal distribution. In other words, noise has a normal distribution. The probability density function of this noise is given by the following equation (2.18).

$$p(z) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(z-\mu)^2/2\sigma^2}$$

(2.18)

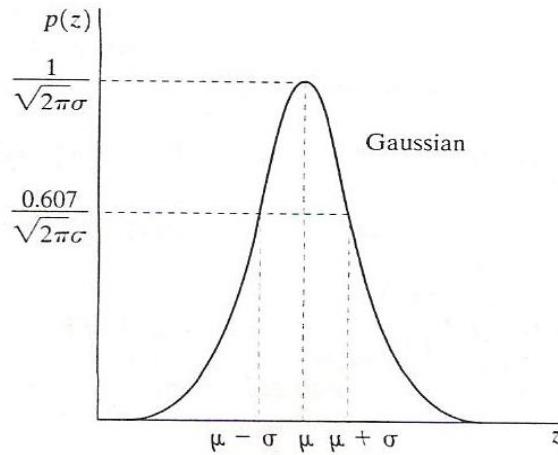
$\mu$ : represent the mean value.

$Z$ : represent the gray level.

$\sigma$ : represent the standard deviation.

$\sigma^2$ : represent the standard square or what is known as the coefficient of variation.

Figure 2.12, shows the PDF of this noise. Note that 70% of this function value in the range of  $[(\mu-\sigma), (\mu+\sigma)]$  and 95% from it is value in the range  $[(\mu-2\sigma), (\mu+2\sigma)]$  the Gaussian noise appears in the image either because of poor lighting or high temperature [41, 42].



**Figure 2.12:** PDF of the Gaussian Noise

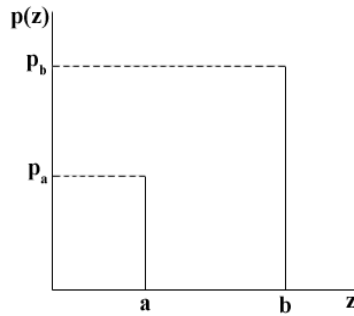
### 2.7.2 Salt and pepper noise

It is also called spike or impulse noise [41]. This noise appear as dark pixels in bright regions of the image and bright pixels in dark regions of the image [42]. This noise appear due to bit errors in transmission and analog to digital converter errors, etc. The probability density function of this noise is shown below in equation 2.19.

$$p(z) = \begin{cases} pa & \text{for } z = a \\ pb & \text{for } z = b \\ 0 & \text{otherwise} \end{cases} \quad (2.19)$$

If  $pa$  or  $pb$  is 0, then it becomes a unipolar impulse noise. If both are nonzero and almost equal, it is called salt-and-pepper noise. This noise can either be positive or

negative. It appears in the image as white and black dots or it can also be like saturated peaks. The following Figure shows the PDF. The impulse noise appears in places, where there is a fast data transfer, such as (faulty switching) that occurs during photography operation [41, 42].



**Figure 2.13:** PDF of the Salt and Pepper Noise

### 2.7.3 Speckle noise

Speckle noise is a granular noise that inherently exists in the SAR images and it degrades the quality of them. Speckle has a negative impact on ultrasound imaging. It increases the gray level mean of a current area. Speckle noise is one of the more complex noise models. It is signal dependent, multiplicative and non-Gaussian. When applying this noise to a brighter area of an image, it presents a magnified view and also a higher random variation can be observed in pixel intensity. And when applying this noise to a darker area in the image, the random variation will be less observed than when it is applied to a brighter areas. Therefore, this type of noise is considered signal dependent and it makes a distortion to the image in large magnitude. The noise introduced by multiplicative effect in an image signal can be represented by equation 2.20.

$$Y_i(x, y) = X_i(x, y) f_i(x, y) \quad (2.20)$$

Where  $X_i(x, y)$  is the input image signal,  $f_i(x, y)$  is the multiplicative noise component and  $y_i(x, y)$  is the output image signal corrupted with noise. This noise can be analyzed with multiplicative or non-linear models. The probability density function

of the multiplicative speckle noise with Rayleigh distributions [43] is given in equation 2.21.

$$p(z) = \begin{cases} \frac{2}{b}(z-a)e^{-\frac{(z-a)^2}{b}} & \text{for } z \geq a \\ 0 & \text{for } z < a \end{cases} \quad (2.21)$$

When  $a > 0$  and  $b$  is a positive number. The variance  $\sigma^2$  and the mean value  $\mu$  of the above pdf is given by equations 2.22 and 2.23 respectively.

$$\mu = a + \sqrt{\frac{\pi b}{4}} \quad (2.22)$$

$$\sigma^2 = \frac{b(4-\pi)}{4} \quad (2.23)$$

## 2.8 Noisy Image

It is a random and undesirable image contains undesirable information in the image intensity color, and can be created by the scanner device or the digital camera [41]. Digital images exposure for different types of noise as a result of errors in the image acquisition or transmitting operation to make changes in the pixel value which do not reflect the true density of the real scene [44, 45]. The spatial component of the noise is based on the statistical behavior of the density values. This may be considered as a random variable characterized by a probability density function (PDF). Probability density function is a random change description of the probability density at each point in the sample space [45]. This thesis considered the following types of common noises: Gaussian, Salt-and-Pepper and Speckle.

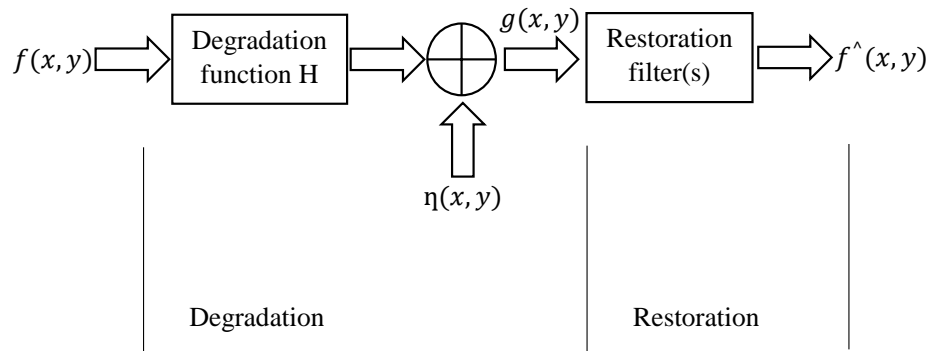
## 2.9 Image Restoration

The ultimate goal of restoration techniques is to improve an image [45, 46]. Restoration attempts to reconstruct or recover an image that has been degraded by using a priori knowledge of the degradation phenomenon. Thus, restoration techniques are oriented

toward modeling the degradation and applying the inverse process in order to recover the original image. Some restoration techniques are best formulated in the spatial domain, while others are better suited for the frequency domain. For example, spatial processing is applicable when the only degradation is additive noise. On the other hand, degradations such as image blur are difficult to approach in the spatial domain using small masks. In this case, frequency domain filters based on various criteria of optimality are the choice of approaches. These filters also take the presence of noise into account.

### 2.9.1 Degradation and restoration process in an image

The following figure shows that the degradation process is symbolized as the function  $H$ . The additive noise  $\eta(x, y)$  with this function operate on an input image  $f(x, y)$  in order to produce a degraded image  $g(x, y)$ . If  $g(x, y)$  is available and some knowledge about  $H$  and  $\eta$ .  $f^{\wedge}(x, y)$  is the estimated image and it must be as close as possible to the input image [37]. More knowledge about  $H$  and  $\eta$  will give an estimated image which is closer to the input image [46].



**Figure 2.14:** Degradation / Restoration Process

The degraded image  $g(x, y)$  is given in the spatial domain by, equation 2.24.

$$g(x, y) = h(x, y) * f(x, y) + \eta(x, y) \quad (2.24)$$

$h(x, y)$  is the spatial representation of the degradation function and, the symbol "\*" stands for convolution. Whereas  $H$  is the Fourier transforms of  $h(x, y)$ .

## 2.9.2 Image restoration in the presence of noise only spatial filtering

When the function  $h(x, y)$  is missing, equation (2.24) become as in equation 2.25.

$$g(x, y) = f(x, y) + \eta(x, y) \quad (2.25)$$

Spatial filtering is used to handle the images that contain only the additive noise. This thesis considered the following types of common spatial filters: arithmetic mean filter, median filter, Alfa trimmed mean filter, Wiener filter.

### 2.9.2.1 Arithmetic mean filter

This filter is one of the easiest types of the mean filters [46]. Where  $S_{xy}$  represents the set of pixels in a sub image window with size  $m \times n$  which is fixed at point  $(x, y)$ . This filter calculates the average value of the degraded image in  $S_{xy}$  area. The value of the enhanced image  $f^{\wedge}(x, y)$  at each point in the image is simply the arithmetic mean computed using the pixels that is defined in the area  $S_{xy}$  as follows:

$$f^{\wedge}(x, y) = \frac{1}{mn} \sum_{(s,t) \in S_{xy}} g(s, t) \quad (2.26)$$

A convolution mask can be used to implement this operation and all its coefficients have value  $1/mn$ . This filter gives good results for the Gaussian noise because the average operation, but adds blurring effect, especially when the size of the mask is big.

### 2.9.2.2 Median filter

The famous order statistics filter is the current filter, and its name refers to the replacement of the current image pixel by the median gray level pixel in the neighborhood of that pixel [46], as it is shown in the equation 2.27.

$$\hat{f}(x, y) = \text{median}_{(s,t) \in S_{xy}} \{g(s, t)\} \quad (2.27)$$

The current image pixel is entered in the calculation of the median gray level. Median filters are famous because they provide best noise removal. This filter gives less blurring results comparison with the linear smoothing filters despite the fact that both of them have the same size. This filter gives good results in Salt & pepper noise because the sort operation, but it rounds the corners.

### 2.9.2.3 Alpha trimmed mean filter

By clipping half value of  $d$  gray level values from both sides of the degraded image, in the block of  $S_{xy}$ . Now let  $g_r(s, t)$  stand for the residual pixels  $mn-d$ . Alpha-trimmed mean filter finds the averaging to the residual pixels as shown in the equation below:

$$\hat{f}(x, y) = \frac{1}{mn-d} \sum_{(s,t) \in S_{xy}} g_r(s, t) \quad (2.28)$$

$d$  takes values from 0 to  $mn-1$ . For  $d=0$ , this filter becomes an arithmetic mean filter, and when  $d=(mn-1)/2$ , the filter becomes a median filter. The alpha-trimmed filter is useful in situations involving multiple types of noise, such as a combination of salt-and-pepper and Gaussian noise for other values of  $d$ . This filter gives good results in salt-and-pepper noise and Gaussian noise because of the sorting and average operations.

### 2.9.2.4 Minimum mean square error filtering

This method considers images and noise as a random process. The aim of this method is to find an estimate  $\hat{f}$  of the uncorrupted image  $f$  until the mean square error between them is minimized [46]. This error is shown below in equation 2.29.

$$e^2 = E\{(f - \hat{f})^2\} \quad (2.29)$$

Where  $E\{.\}$  it is the expected value of the argument. The noise and the image are assumed to be not correlated or one of them has zero mean; and that the gray levels in



the estimate are a linear function of the levels in the degraded image. The frequency domain expression in equation 2.30 below, of the minimum error function that is dependent on the above criteria:

$$f^{\wedge}(u, v) = \left[ \frac{1}{H(u, v)} * \frac{|H(u, v)|^2}{|H(u, v)|^2 + \frac{S_{\eta}(u, v)}{S_f(u, v)}} \right] G(u, v) \quad (2.30)$$

As it is known that the product of a complex quantity multiplied by its conjugate, is the squared magnitude of the complex quantity.

$S_f(u, v) = |f(u, v)|^2 =$  Power spectral of the ungraded image.

$S_{\eta}(u, v) = |N(u, v)|^2 =$  Power spectral of the noise.

$H(u, v) =$  Degradation function.

$H^*(u, v) =$  Complex conjugate of  $H(u, v)$ .

$|H(u, v)|^2 = H^*(u, v) H(u, v)$ .

So the wiener filter minimizes the overall mean square error also it is easy to see from equation (2.30) that the Wiener filter has two separate parts, an inverse filtering part and a noise smoothing part.

## 2.10 Algorithms Evaluation

In this thesis the normalized correlation (NC) and the PSNR were used to evaluate the edge detection algorithms and noise removal algorithms respectively.

### 2.10.1 Normalized correlation

It is used to find the relation degree between two dependence variables x and y, and its result is a number between +1 and -1 [47], +1 means full positive correlated signals, 0 means uncorrelated signals, whereas -1 means negative correlated signals. The

correlation is called weak correlation if its value is less than 0.5 and it is called strong correlation if its value greater than 0.5. Karl Pearson developed this correlation from Francis Galton's related idea produced in the 1880s [48,49]. The equation of the normalized correlation is given by equation 2.31.

$$r = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2} \sqrt{\sum_{i=1}^n (y_i - \bar{y})^2}} \quad (2.31)$$

$r$  : the correlation factor.

$\bar{x}$  : mean value of the function ( $x$ ).

$\bar{y}$  : mean value of the function ( $y$ ).

$x_i$ : one value from the input signal ( $x$ ).

$y_i$ : one value from the input signal ( $y$ ).

### 2.10.2 Peak signal to noise ratio

PSNR defined as the proportion between the noise power and the maximum power of the input signal. PSNR is often represented in logarithmic decibel scale because a lot of signals have a range which is dynamic. For example, PSNR is used to measure the quality of lossy compression codecs, here the signal is the original data, and the noise is the error that is generated by compression. The higher PSNR value represents higher quality reconstruction, but sometimes it may not. PSNR is defined via the mean squared error (MSE) very easily as follows in equation 2.32 [50].

$$MSE = \frac{1}{m \cdot n} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [I(i, j) - In(i, j)]^2 \quad (2.32)$$

The PSNR is defined as in equation 2.33 below:

$$PSNR = 10. \log_{10} \left( \frac{MaxI^2}{MSE} \right)$$

(2.33)

Here, MAXI is the maximum possible pixel value of the image. When the pixels are represented using 8 bits per sample, this is 255. For color images with three RGB values per pixel, the definition of PSNR is the same except the MSE is the sum over all squared value differences divided by image size and by three. Alternately, Naturally PSNR values of video compression and lossy image are between (30 and 50 dB) in 8 bit depth, whereas for 16 bit the PSNR values are between (60 and 80 dB). For wireless transmission quality loss, the 20 dB to 25 dB are acceptable values [51,52]. The PSNR is undefined if there is no noise, and the MSE is zero because the images  $I$  and  $K$  are identical [53, 54].

## CHAPTER 3

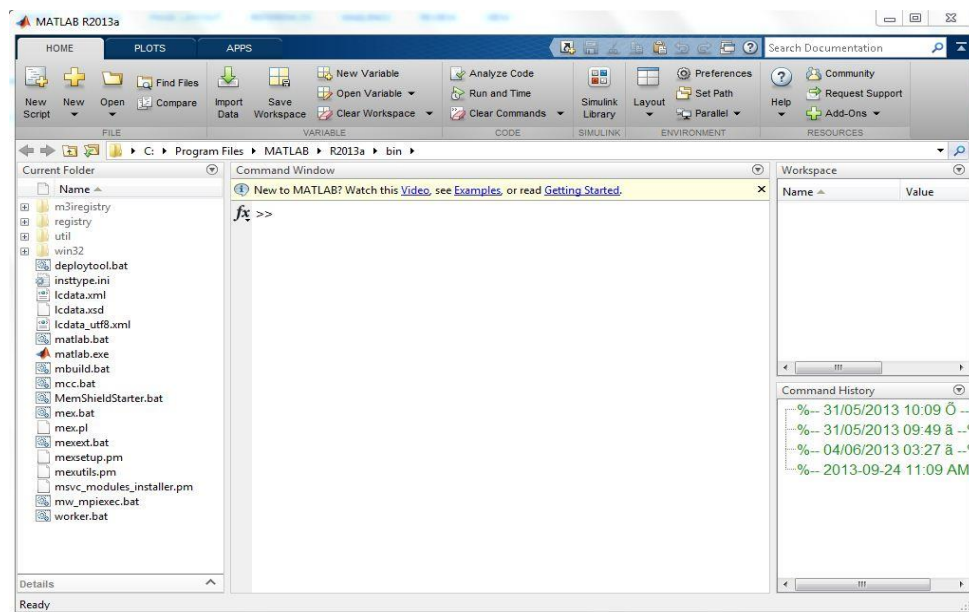
### TOOLS AND METHODS

#### 3.1 Tools Used in Thesis

This section is divided into two parts, the first part is talking about the MATLAB language that is used to implement the algorithms. The second part dealt with the original images that are used in this thesis.

##### 3.1.1 Matlab

In this thesis, the MATLAB library, version R2013a is used to implement the algorithms. MATLAB is an abbreviation for Matrix Laboratory. MATLAB is a high-level language and interactive environment that enables you to perform computationally intensive tasks. Figure 3.1 shows the MATLAB structure.



**Figure 3.1:**  
Matlab

Structure

From the previous Figure 3.1 the MATLAB windows component consists of:

- Workspace
  - Displays all the defined variables.
- Command Window
  - To execute commands in the MATLAB environment.
- Command History
  - Displays record of the commands used.
- File Editor Window
  - Define your functions.

The important MATLAB instructions that are used in this study will be given in general form and explained. More information about MATLAB instructions, can be found in [55].

#### MATLAB instruction to read the input images

*image read = imread('image path\image name .image extension')*

(3.1)

This instruction was used to read the original images, these images are grayscale images that have the same size of 256X256 and the same extension “PNG”.

#### MATLAB noise instruction

*Noisy image = imnoise(source image, 'type of noise', variance value)*

(3.2)

This instruction was used to add the following noises: Gaussian noise, Salt and Paper noise, and Speckle noise, to the original images. The variance that is used in all types of noise  $\sigma=0.01$ .

#### MATLAB noise removal instruction

*denoising image = filter type(noisy image, [mask size])*

(3.3)

This instruction was used to apply the following spatial filters: Arithmetic Mean Filter, Median Filter, Alfa trimmed Mean Filter, and Wiener Filter on the noisy images. 3x3 mask size was used in all filters.

#### MATLAB edge instruction

$$\textit{edge image} = \textit{edge}(\textit{source image}, \textit{edge algorithm}, \textit{threshold value})$$

(3.4)

The MATLAB edge instruction mostly depends on the threshold value, by this value edge ignores all edges that are not stronger than the given threshold value. Changing this threshold value, the output edge image will be changed. The best threshold value is the threshold value that gives us an optimal edge detection (i.e. an image contains all edge details). All edge algorithms use one threshold value except Canny algorithm threshold is a two-element vector in which the first element is the low threshold, and the second element is the high threshold. In this thesis the low threshold of the Canny algorithm is set to zero and the change just will be in the high threshold.

#### MATLAB correlation instruction

$$\textit{correlation} = \textit{corr2}(\textit{image1}, \textit{image2})$$

(3.5)

This instruction was used to make an evaluation to the edge detection algorithms.

#### MATLAB PSNR instruction

$$\textit{psnr value} = \textit{step}(\textit{hpsnr}, \textit{image1}, \textit{image2})$$

(3.6)

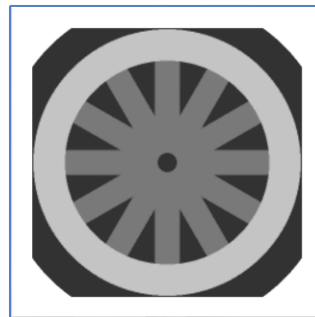
This instruction was used to make an evaluation to the noise removal algorithms (filters).

### 3.1.2 Original images

The original images that are used in this thesis are divided into six categories as follows: binary image, graphic image, high frequency image, low frequency image, median frequency image, and texture image. The binary image is an image that consists of a two gray level value. The graphic image is an image that every region in it almost contains the same values of the gray level. If the most frequencies in the image are high frequencies this image is a high frequency image, if the most frequencies in the image are low frequencies this image is a low frequency image and if the most frequencies in the image are median frequencies this image is a median frequency image. Texture image contains a variety of intensities which form certain repeated patterns [38]. The original images that are used in this thesis are shown below in the figure 3.2.



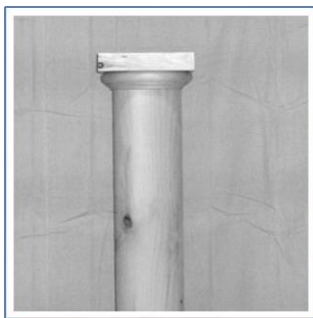
binary image



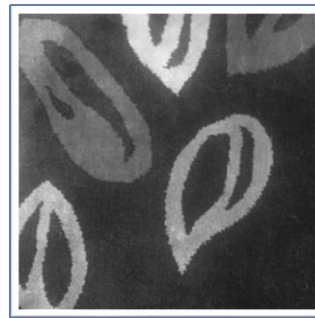
graphic image



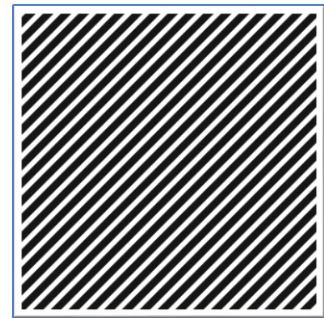
high frequency image



low frequency image



median frequency image



texture image

**Figure 3.2:** Original Images

## 3.2 Methodology

The following five edge detection algorithms: Roberts, Sobel, Prewitt, Laplacian of Gaussian, and Canny are implemented in three environments clean environment, noisy environment and de-noising environment, by using the original images that are shown in the above Figure 3.2. In noise environment, the following types of noise are used: Gaussian noise, salt-and-pepper noise, and speckle noise, equal variance was used in all types of noise ( $\sigma = 0.01$ ). In de-noising environment the following types of noise removal (filters) are used: arithmetic mean filter, median filter, Alpha trimmed mean filter “ATMF” and Wiener filter, all filters have the same mask size 3x3.

### 3.2.1 Clean environment

In this environment the source images are the original images, the edge detection images of each original image are obtained by changing the threshold value in MATLAB edge instruction for each one of the five edge detection algorithms using *trial-and-error* method until the output is an optimal edge detection image (image containing all edge details without losing any edge information and without the presence of any fake edge).

### 3.2.2 Noisy environment

In this environment the source images are noisy images resulted from adding the noises that are mentioned previously to the original images. Accordingly the source images are noisy images (Gaussian, salt and pepper and speckle). The edge detection images for each noisy image is obtained by using a software which continuously changes the threshold value for each one of the five edge detection algorithms, until the correlation factor between the noisy edge detection image and the clean edge detection image reaches its maximum value. It is important to say that the clean edge detection images came from the clean environment.



### 3.2.3 De-noising environment

In this environment, the work is divided in two steps:

1. In this step, a suitable filter for each type of noise must be chosen, by de-noising the noisy images using the following spatial filters: mean filter, median filter, Alfa trimmed mean filter (ATMF), and Wiener filter, all with the same mask size 3x3, after that an evaluation of the noise removal algorithms is used to choose the suitable filter for each type noise.
2. The source images are the de-noised images from the previous step. The edge detection images for each de-noised image are obtained by using a software which continuously changes the threshold value for each one of the five edge detection algorithms, until the correlation factor between the de-noised edge detection image and the clean edge detection image reaches the maximum value. Hence, this step like section 3.2.2, just the source images are the de-noised images. Figures 3.3, 3.4, 3.5 and 3.6, describes the whole idea of this study.

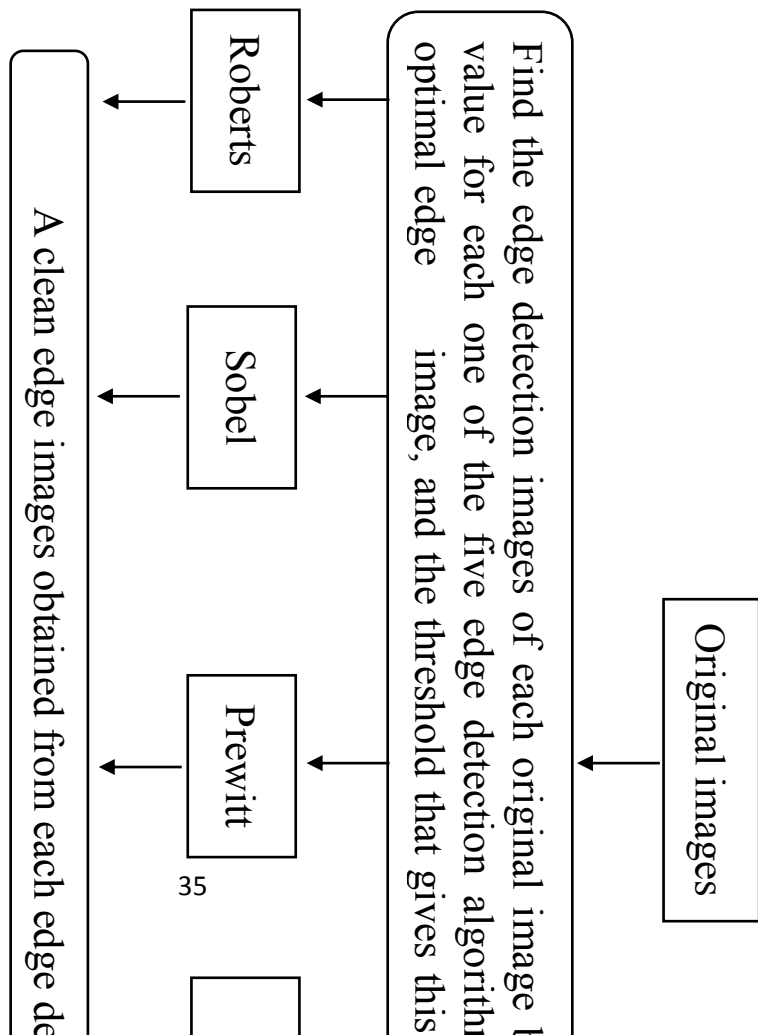


Figure 3.3 : Clean Environment Scheme

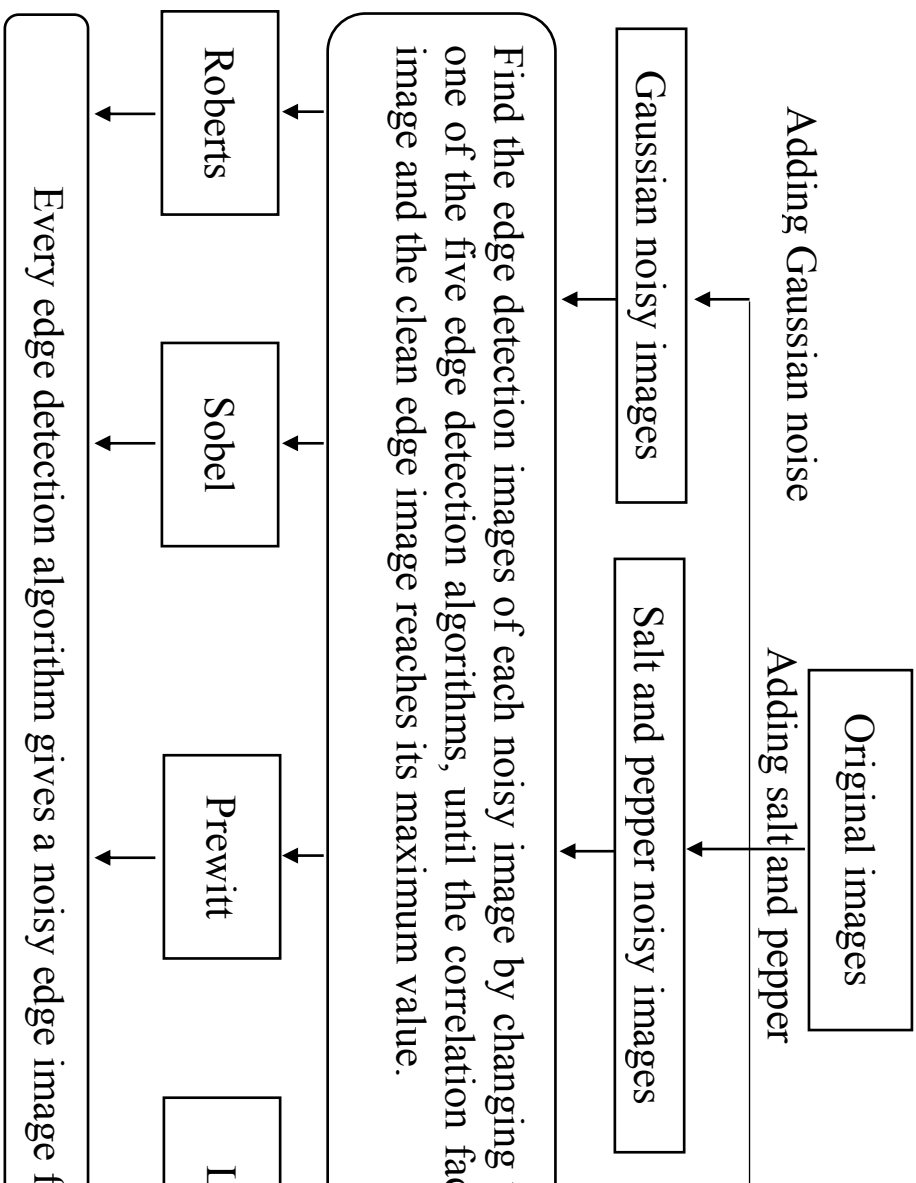


Figure 3.4: Noisy Environment Scheme

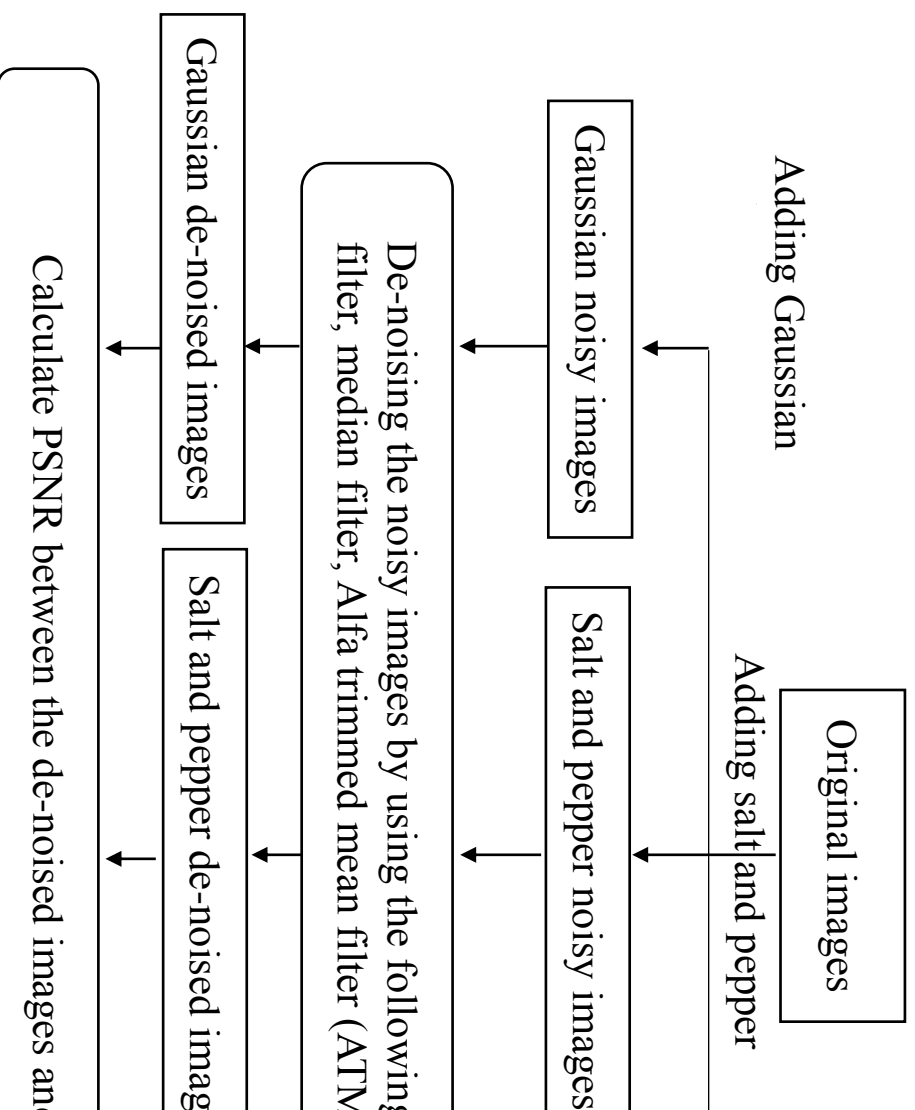


Figure 3.5: Noise Removal Scheme

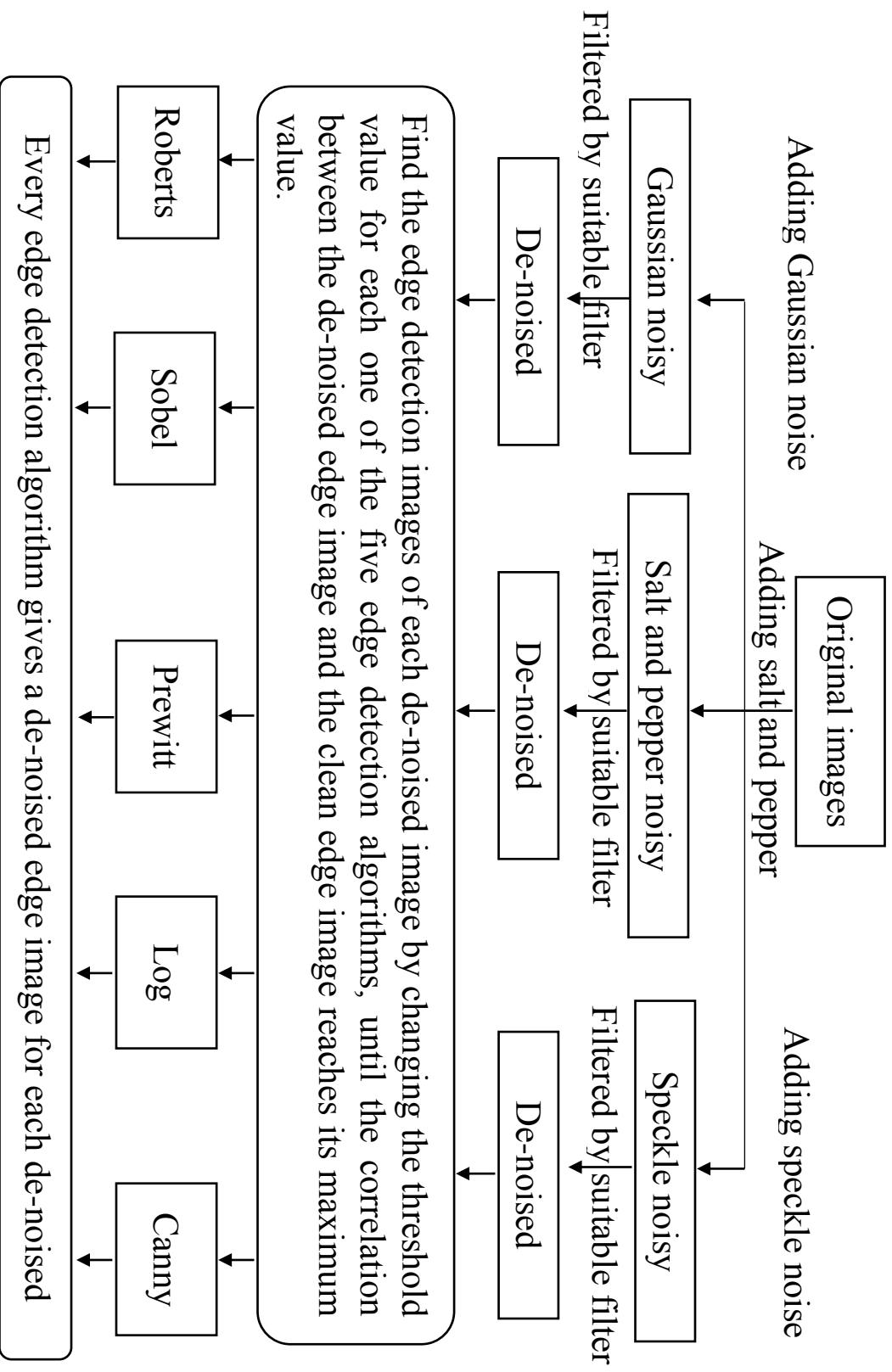


Figure 3.6: De-Noising Environment Scheme

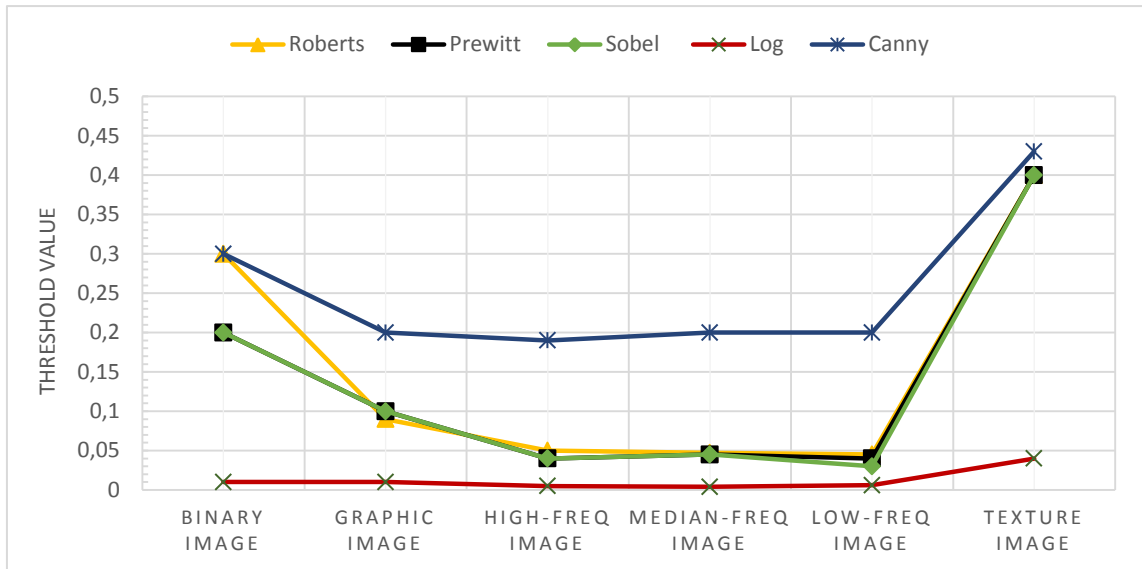
## CHAPTER 4

### PRACTICAL RESULTS

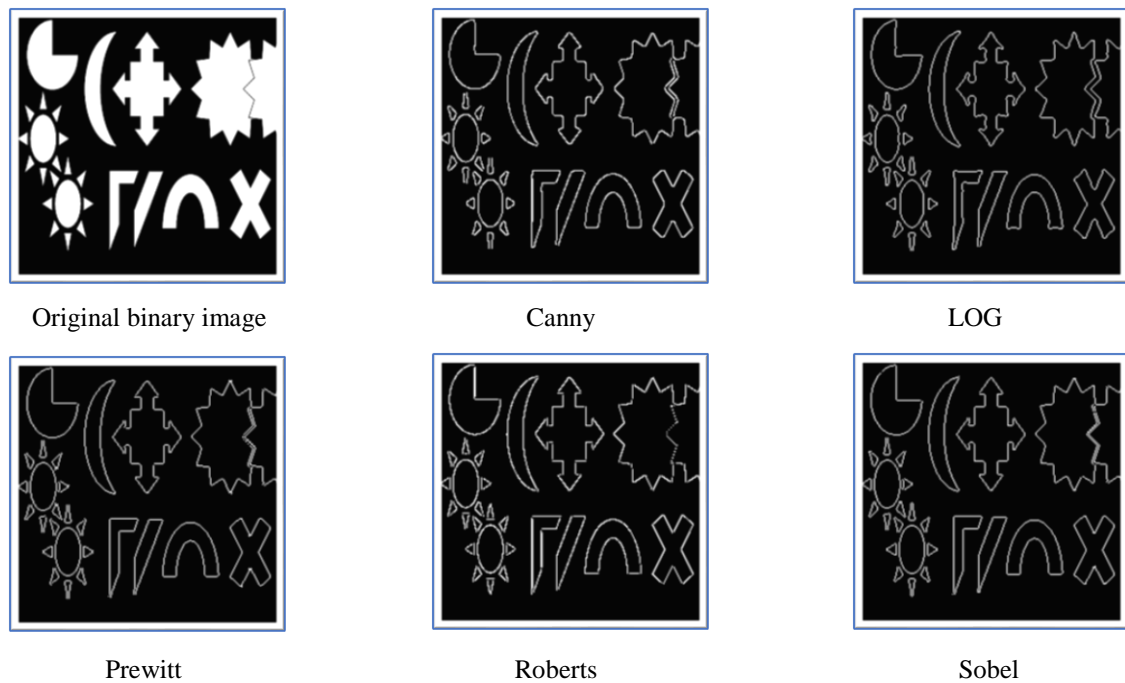
In this chapter the results of the edge detection algorithms are shown in multi-environments. Therefore, this chapter is divided into three sections as follows: the first section contains the results of clean environment, the second section contains the results of noisy environment, and the third section contains the result of de-noising environment.

#### 4.1 Clean Environment Results

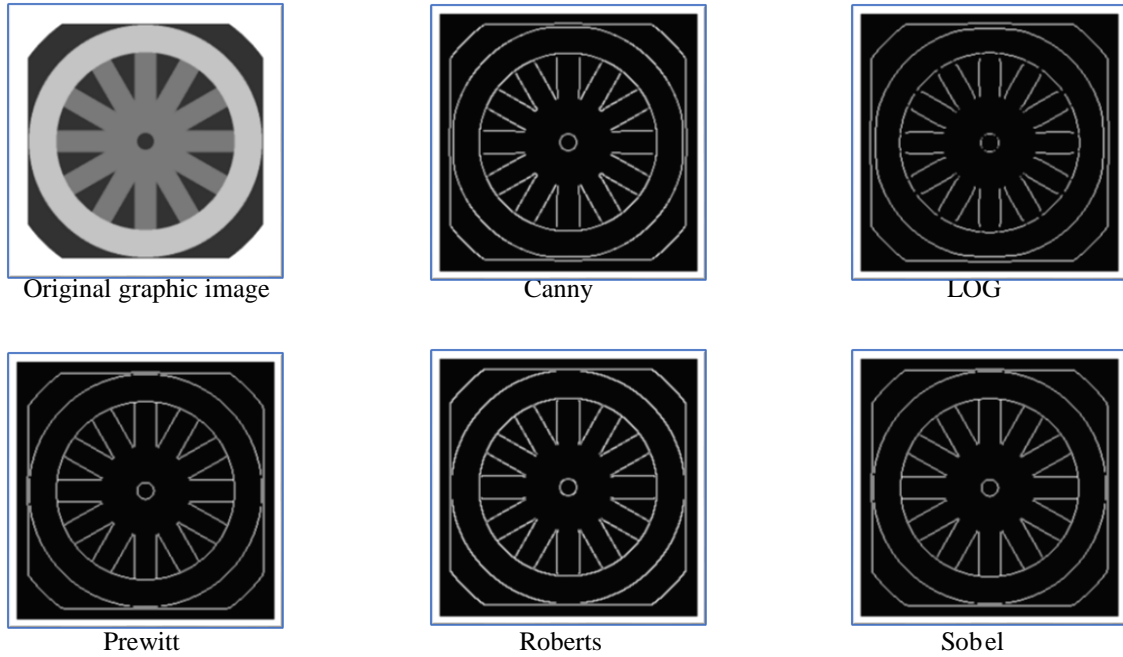
Figure 4.1 is threshold values of the edge detection algorithms in the clean environment and in this figure with all types of original images, it can be seen that the Canny takes the highest threshold values, LOG takes the lowest threshold values, Sobel and Prewitt take an identical threshold values and Roberts takes threshold values that is not so different from the Prewitt and Sobel threshold values. The figures from 4.2 to 4.7 are the edge detection images of the original images and from these figures it can be seen that all algorithms give a good detection if the threshold value of each detection algorithm is chosen in an appropriate way as it is explained in section 3.2.1, and figure 3.3.



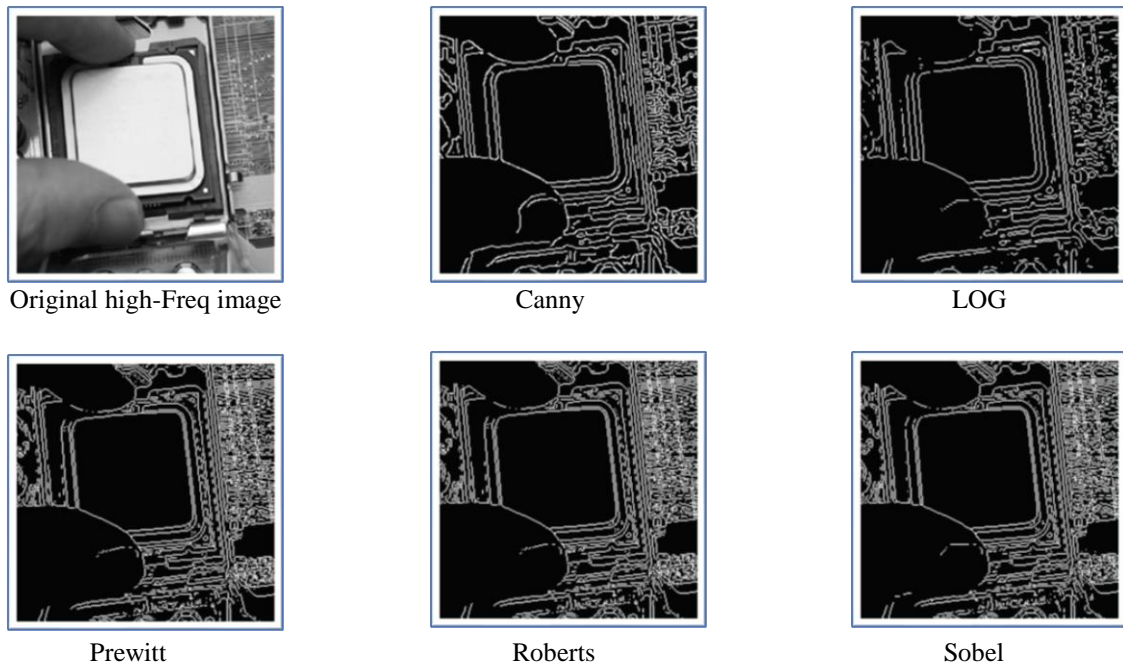
**Figure 4.1:** Threshold Values of Edge Detection Algorithms in Clean Environment



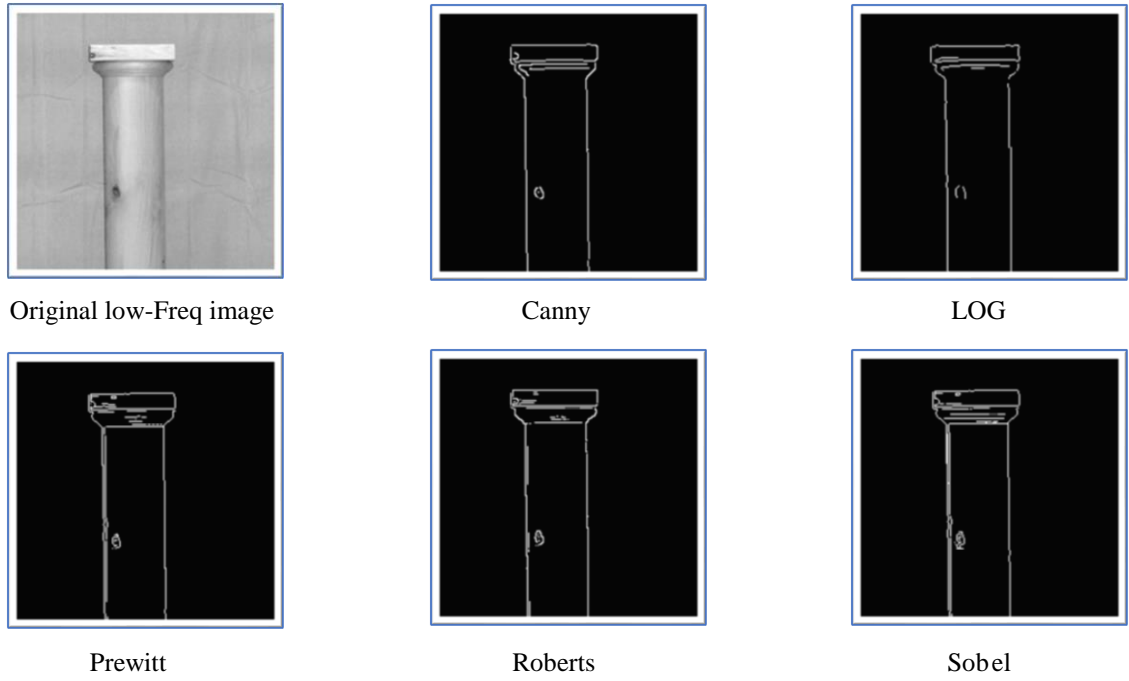
**Figure 4.2:** Edge Detection Images of Binary Image in Clean Environment



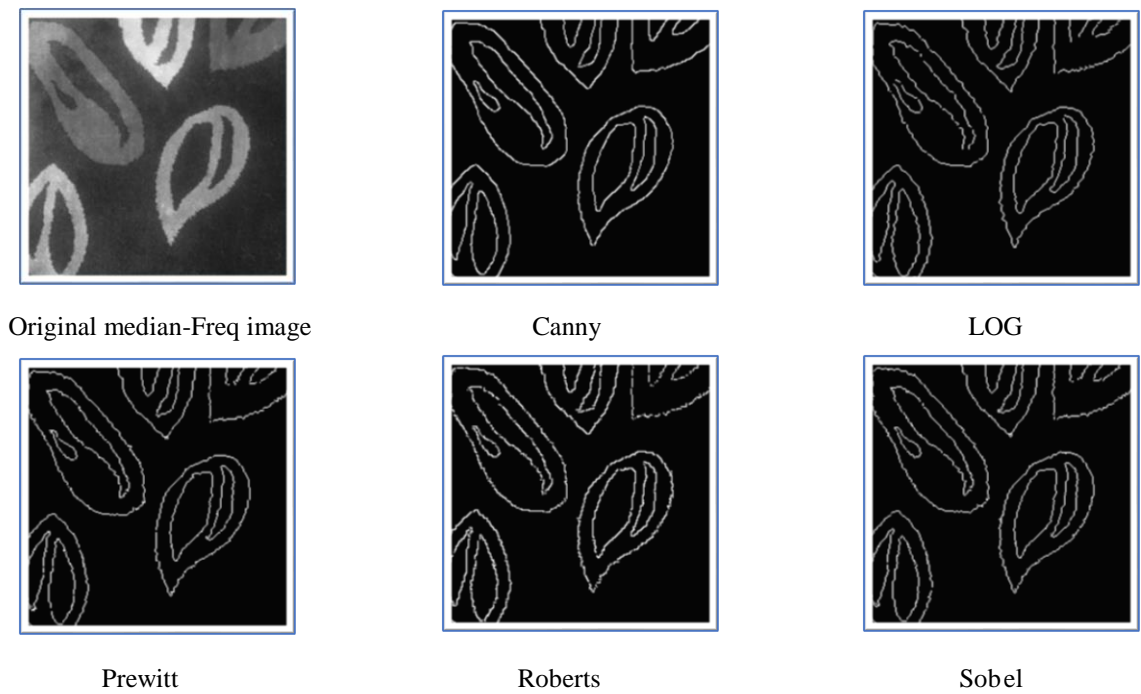
**Figure 4.3:** Edge Detection Images of Graphic Image in Clean Environment



**Figure 4.4:** Edge Detection Images of High-Freq Image in Clean Environment

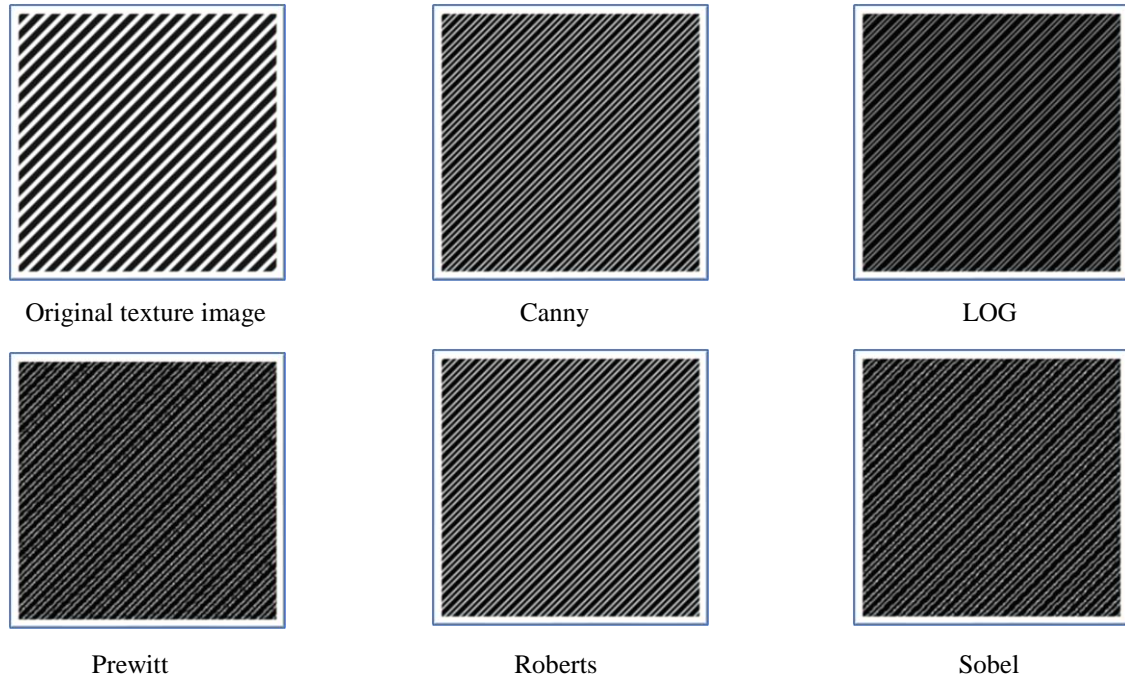


**Figure 4.5:** Edge Detection Images of Low-Freq Image in Clean Environment



**Figure 4.6:** Edge Detection Images of Median-Freq Image in Clean Environment





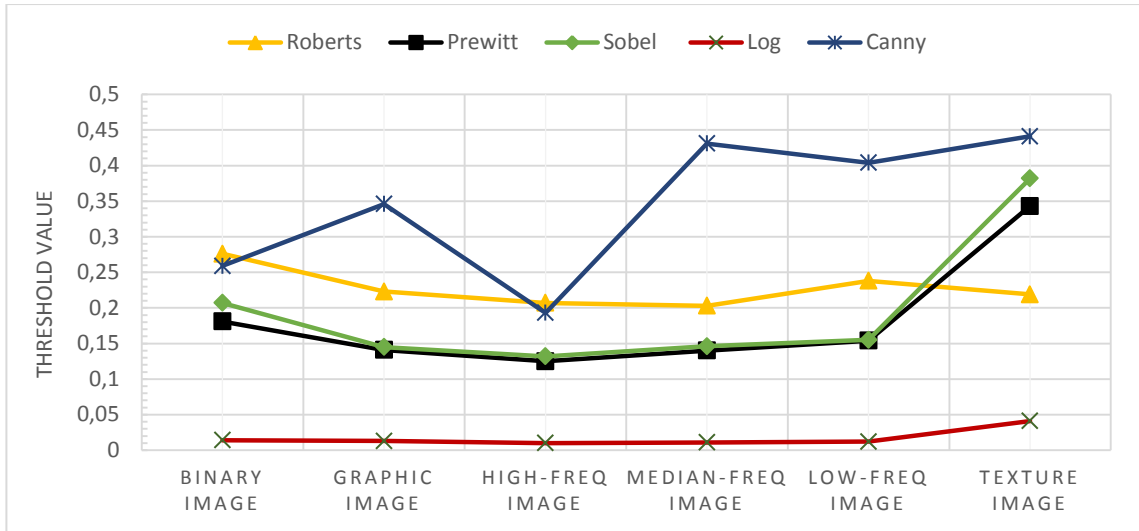
**Figure 4.7:** Edge Detection Images of Texture Image in Clean Environment

## 4.2 Noisy Environment Results

In this environment the source images are noisy images. Depending on the type of noise in the original images, this section is divided into three parts.

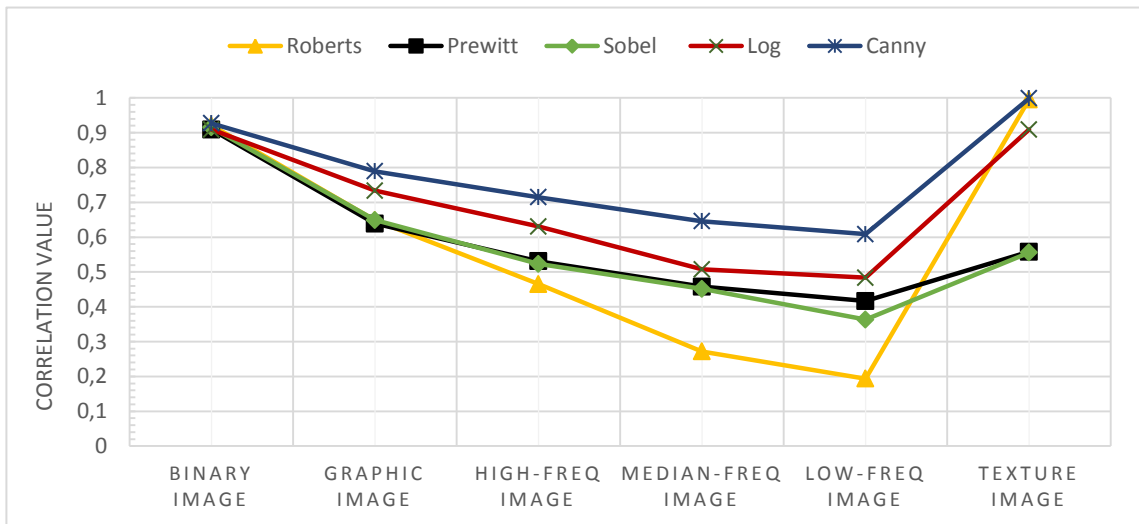
### 4.2.1 Noisy environment results-gaussian noisy images

Figure 4.8 is the threshold values of the edge detection algorithms in the noisy environment-Gaussian noise. In this figure with all types of Gaussian-noisy images, it can be seen that the Canny takes the highest threshold values, LOG takes the lowest threshold values, Sobel and Prewitt almost take an identical threshold values and Roberts takes threshold values that is not so different from the Prewitt and Sobel threshold values.



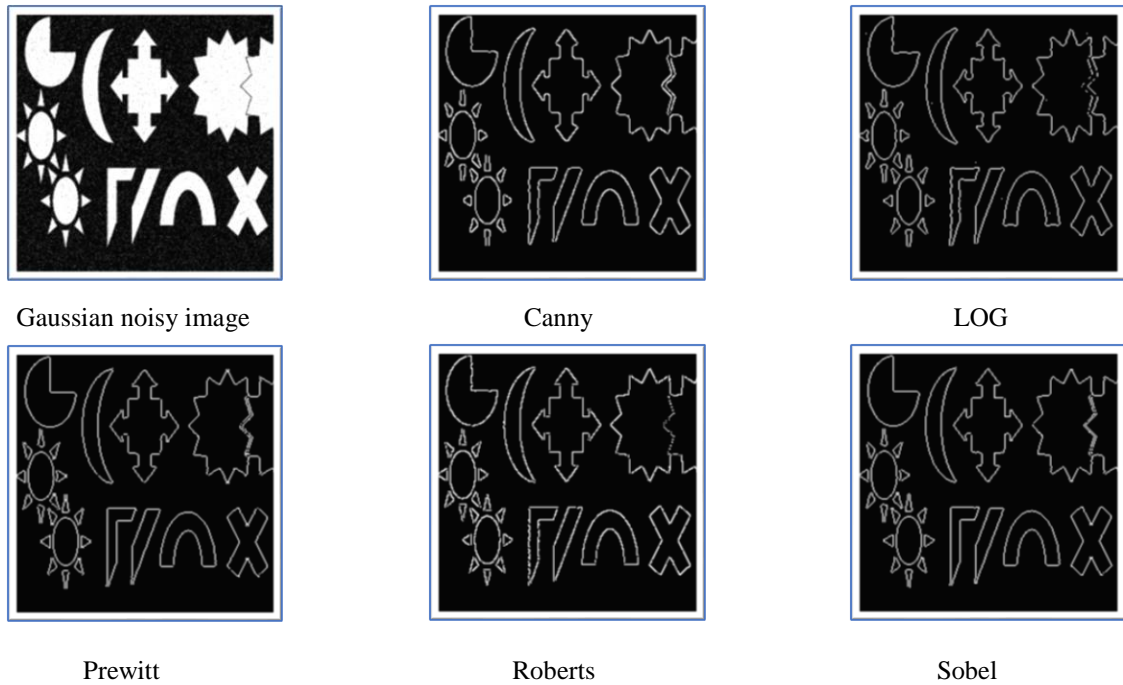
**Figure 4.8:** Threshold Values of Edge Detection Algorithms in Noisy Environment-Gaussian Noise

Figure 4.9 is the correlation value between the Gaussian-noisy edge images and the clean edge images that came from the clean environment. In this figure with all types of Gaussian-noisy images the observation that can be seen that the correlation always is high with Canny algorithm and always is low with Roberts algorithm except when the input is the binary or texture image. All edge algorithms give a good detection when the input is binary image and Roberts gives good detection with the texture image.

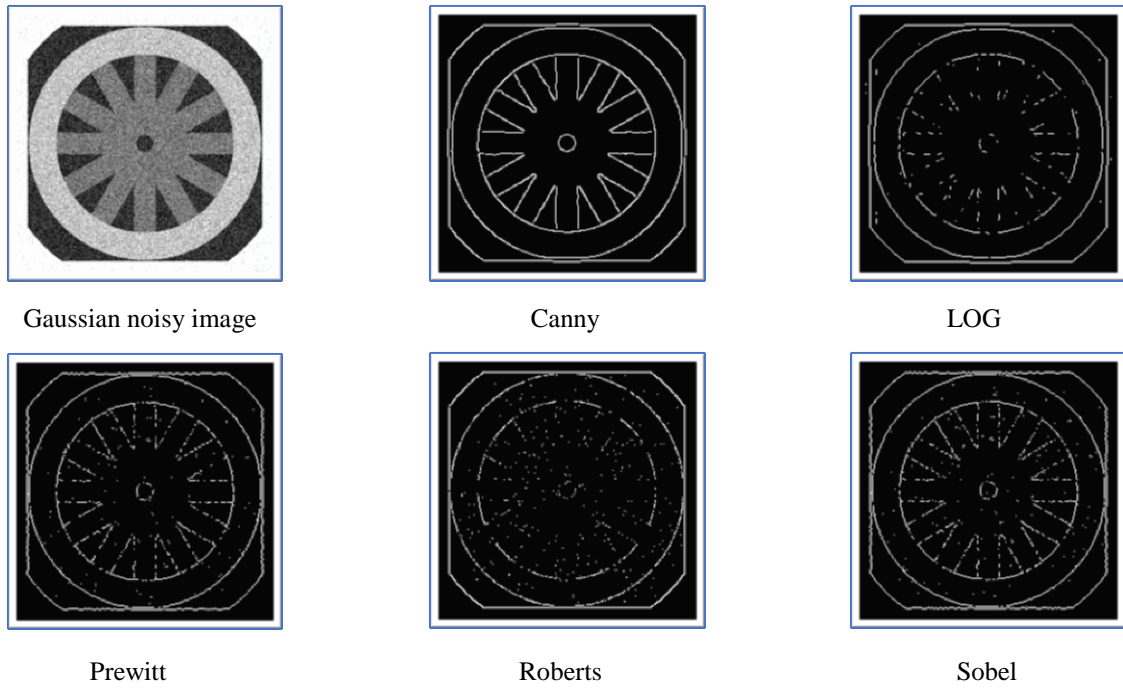


**Figure 4.9:** Correlation Values of Noisy Environment-Gaussian Noise

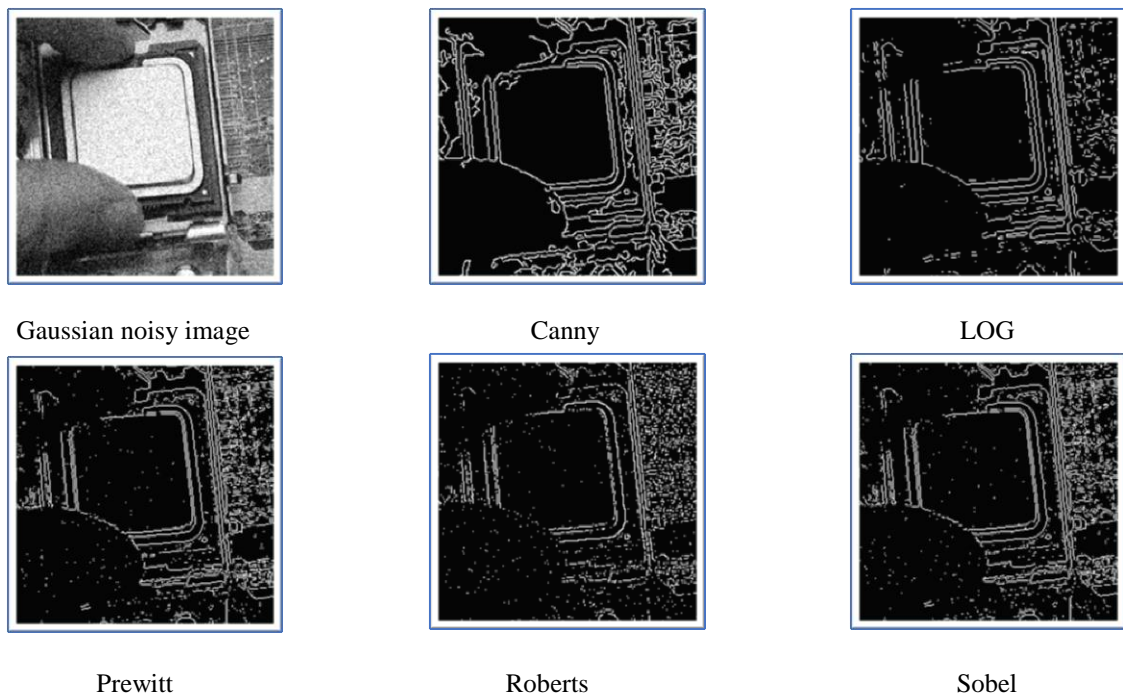
The figures from 4.10 to 4.15 are the edge detection images of the Gaussian noisy images. For the edge images figures 4.11, 4.12, 4.13 and 4.14, it can be seen that the algorithms Roberts, Prewitt and Sobel give bad detection also it is possible to give a sequence to these algorithms from best to worst detection: Canny, LOG, Prewitt, Sobel and Roberts.



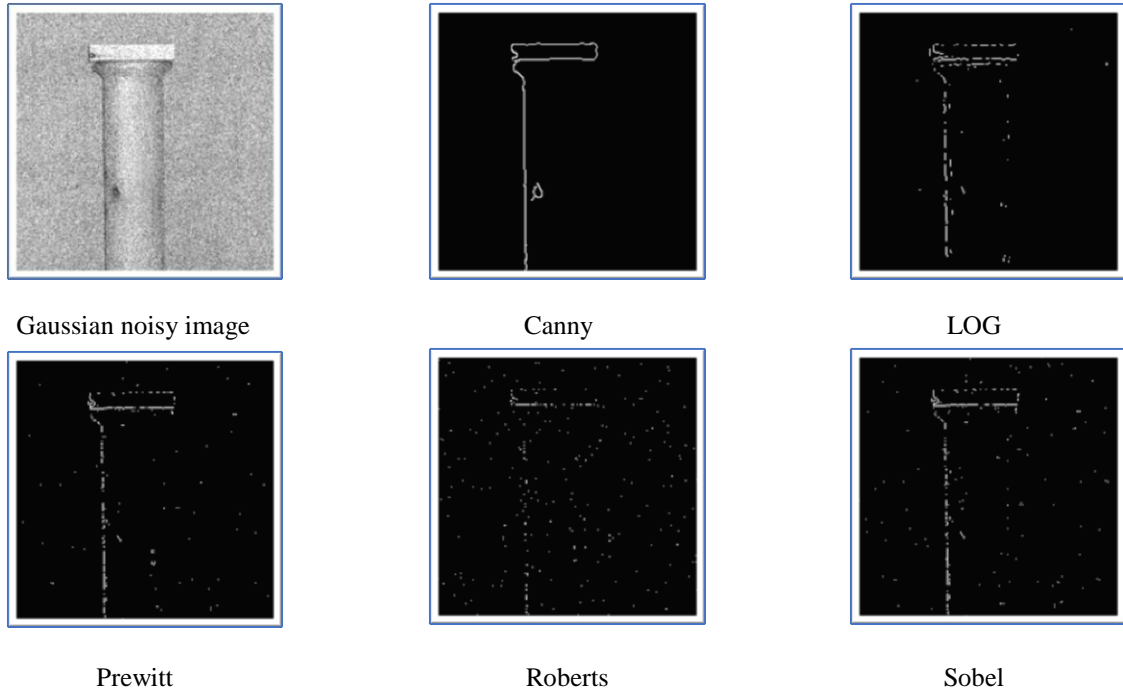
**Figure 4.10:** Edge Detection Images of Binary Image in Noisy Environment-Gaussian Noise



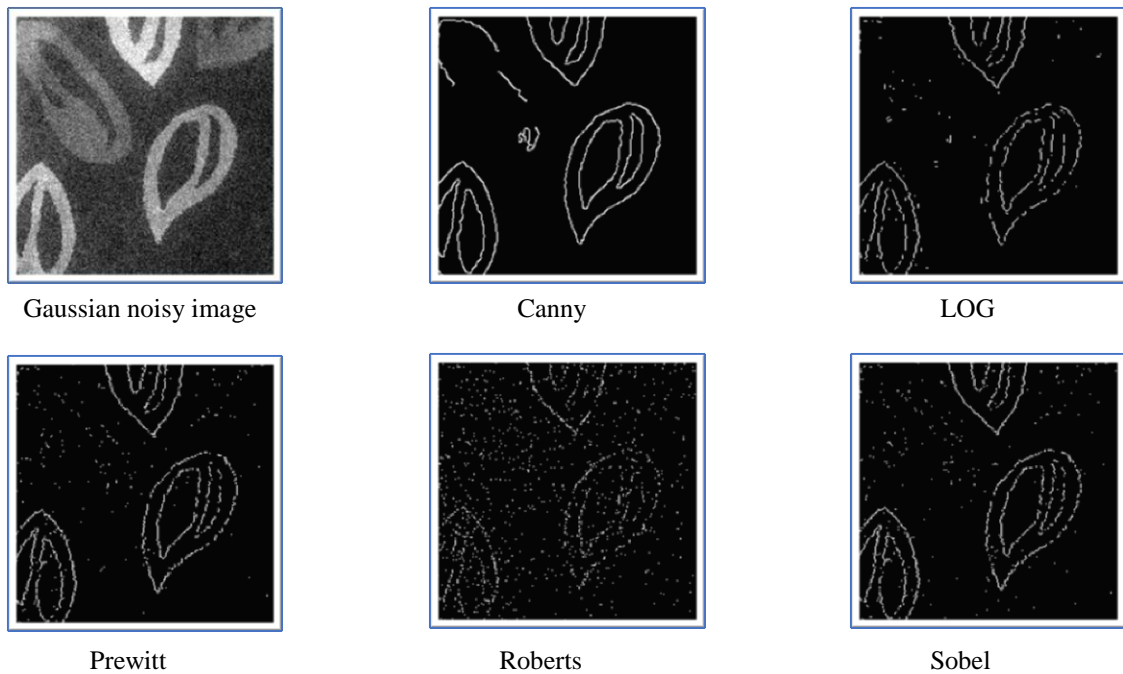
**Figure 4.11:** Edge Detection Images of Graphic Image in Noisy Environment-Gaussian Noise



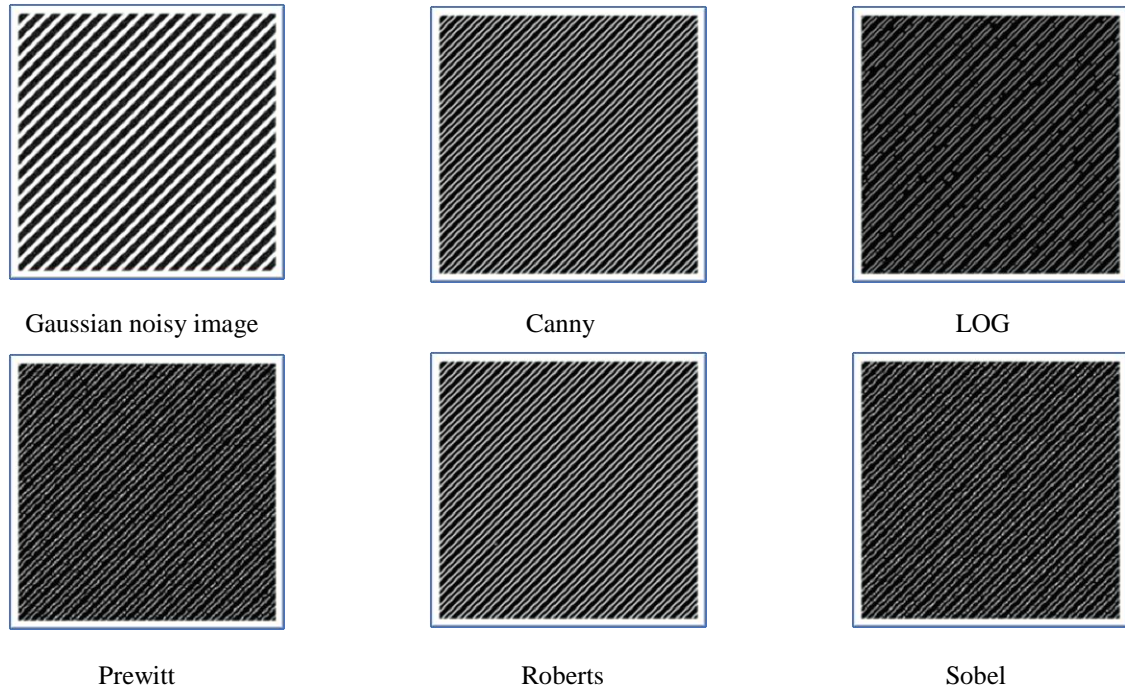
**Figure 4.12:** Edge Detection Images of High-Freq Image in Noisy Environment-Gaussian Noise



**Figure 4.13:** Edge Detection Images of Low-Freq Image in Noisy Environment-Gaussian Noise



**Figure 4.14:** Edge Detection Images of Median-Freq Image in Noisy Environment-Gaussian Noise

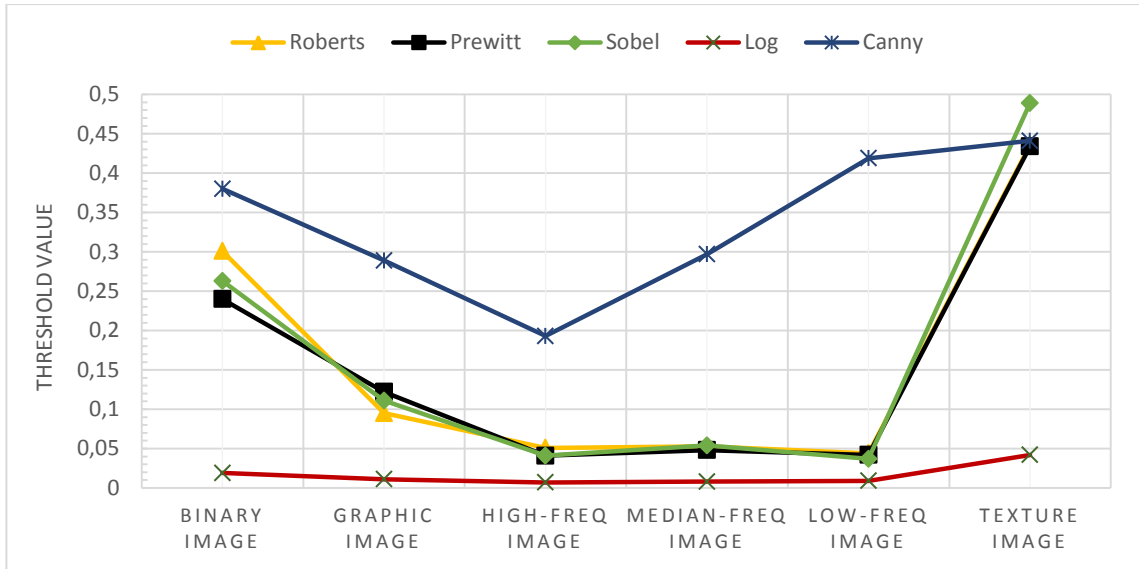


**Figure 4.15:** Edge Detection Images of Texture Image in Noisy Environment-Gaussian Noise

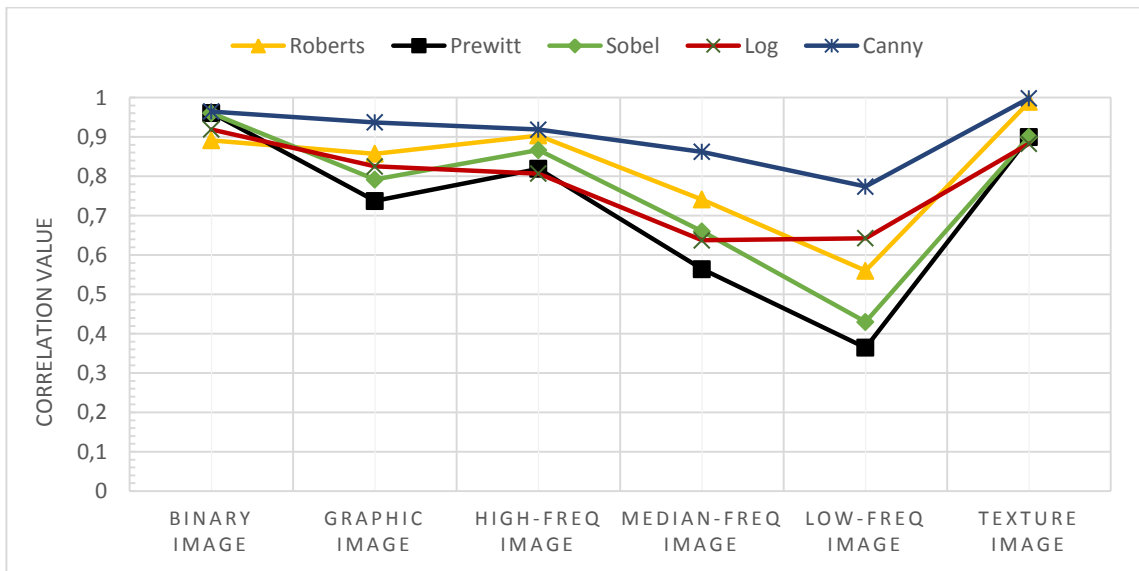
#### 4.2.2 Noisy environment results-salt and pepper noisy images

Figure 4.16 is the threshold values of the edge detection algorithms in the noisy environment-salt and pepper noise. In this figure with all types of salt and pepper-noisy images, it can be seen that the Canny takes the highest threshold values, LOG takes the lowest threshold values, Sobel, Prewitt and Roberts almost take an identical threshold values.

Figure 4.17 is the correlation value between the salt and pepper-noisy edge images and the clean edge images that came from the clean environment. In this figure with all types of salt and pepper-noisy images the observation that can be seen that the correlation always is high with Canny algorithm and always is low with Prewitt algorithm except when the input is the binary. All edge algorithms give a good detection when the input is binary image and Roberts gives good detection with the texture image.



**Figure 4.16:** Threshold Values of Edge Algorithms in Noisy Environment-Salt and Pepper Noise



**Figure 4.17:** Correlation Values of Noisy Environment-Salt and Pepper Noise

The figures from 4.18 to 4.23 are the edge detection images of the salt and pepper noisy images. For the edge images figures 4.19, 4.20, 4.21 and 4.22 it can be seen that the algorithms Roberts, Prewitt and Sobel give the worst detection.



Salt and pepper noisy image



Canny



LOG



Prewitt

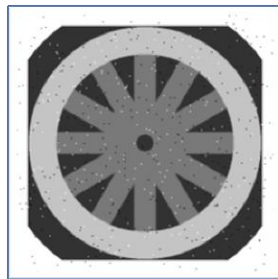


Roberts

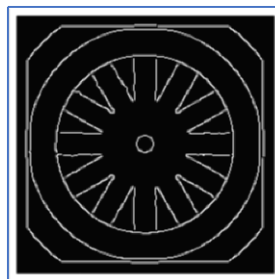


Sobel

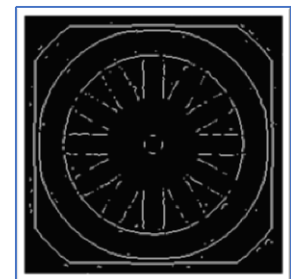
**Figure 4.18:** Edge Detection Images of Binary Image in Noisy Environment-Salt and Pepper Noise



Salt and pepper noisy image



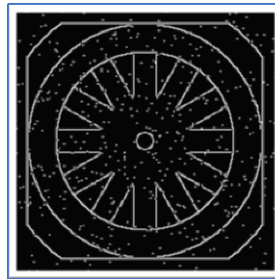
Canny



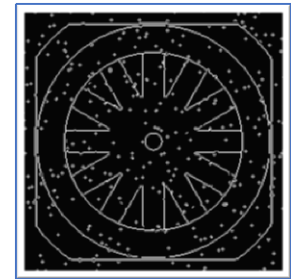
LOG



Prewitt



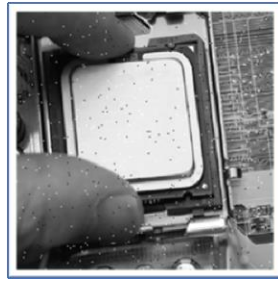
Roberts



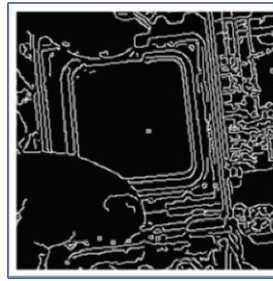
Sobel

**Figure 4.19:** Edge Detection Images of Graphic Image in Noisy Environment-Salt and Pepper Noise

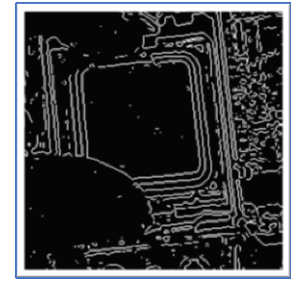




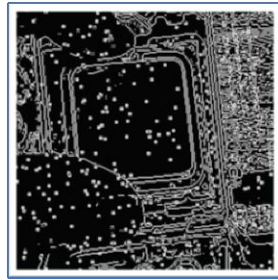
Salt and pepper noisy image



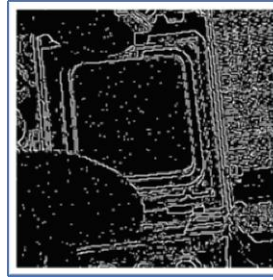
Canny



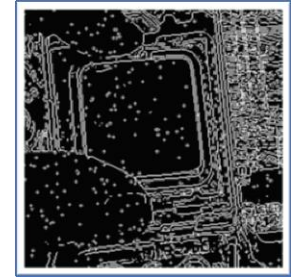
LOG



Prewitt

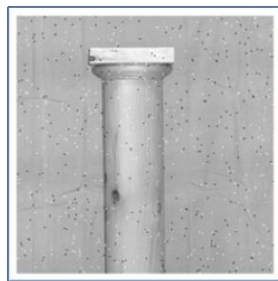


Roberts

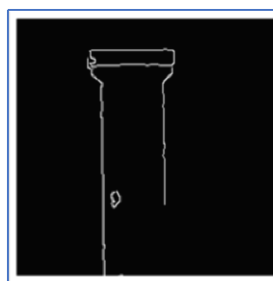


Sobel

**Figure 4.20:** Edge Detection Images of High-Freq Image in Noisy Environment-Salt and Pepper Noise



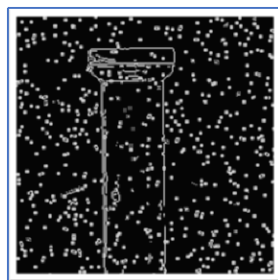
Salt and pepper noisy image



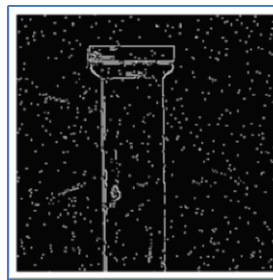
Canny



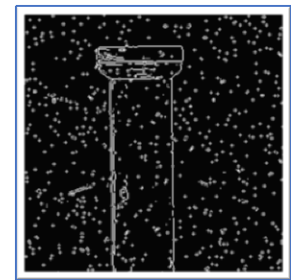
LOG



Prewitt

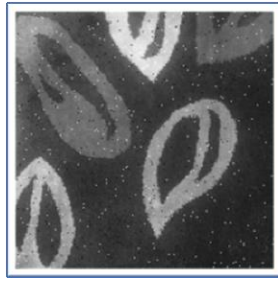


Roberts



Sobel

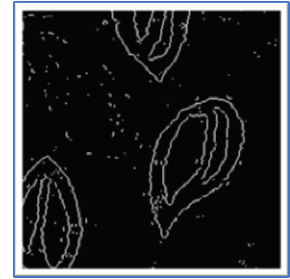
**Figure 4.21:** Edge Detection Images of Low-Freq Image in Noisy Environment-Salt and Pepper Noise



Salt and pepper noisy image



Canny



LOG



Prewitt

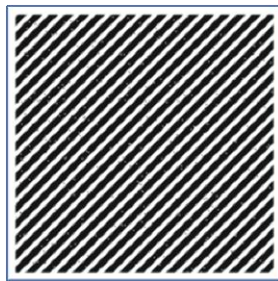


Roberts

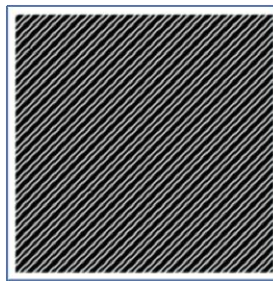


Sobel

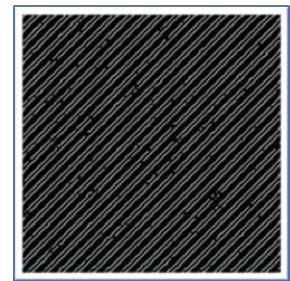
**Figure 4.22:** Edge Detection Images of Median-Freq Image in Noisy Environment-Salt and Pepper Noise



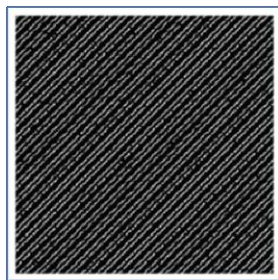
Salt and pepper noisy image



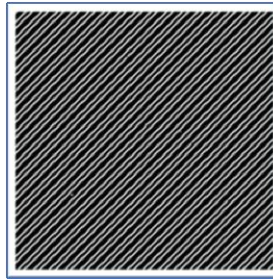
Canny



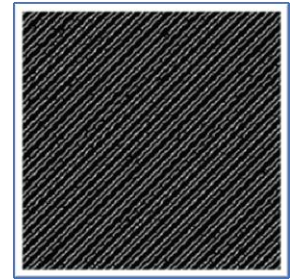
LOG



Prewitt



Roberts

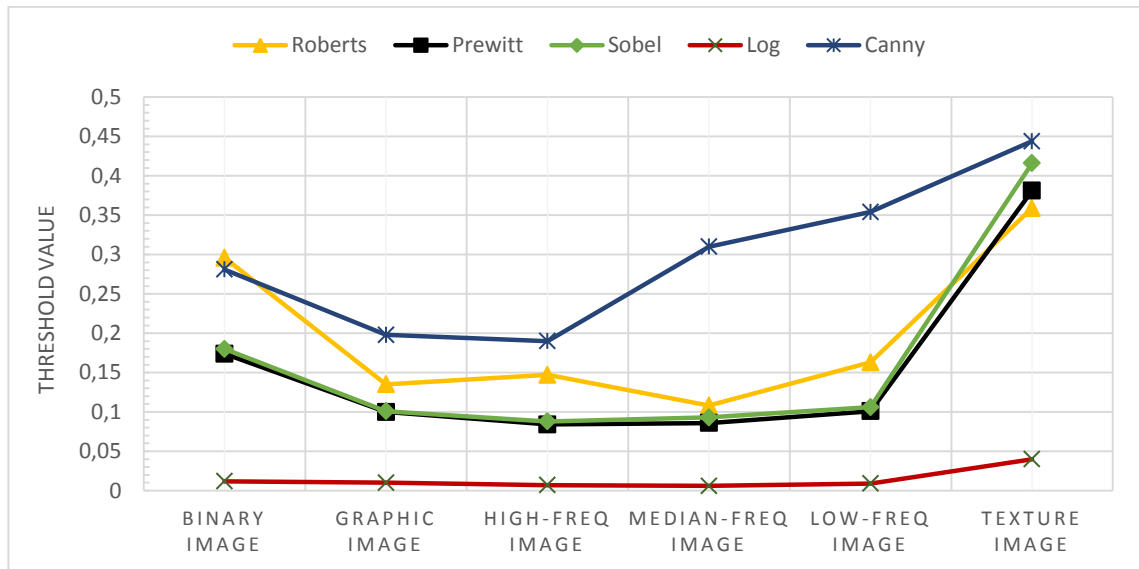


Sobel

**Figure 4.23:** Edge Detection Images of Texture-Freq Image in Noisy Environment-Salt and Pepper Noise

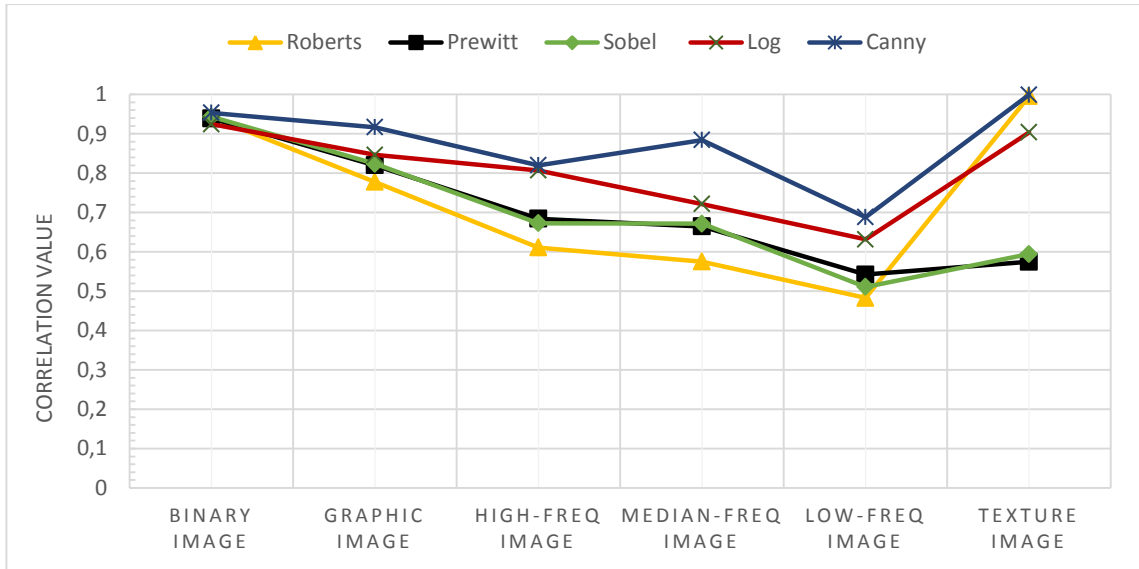
### 4.2.3 Noisy environment results-speckle noisy images

Figure 4.24 is the threshold values of the edge detection algorithms in the noisy environment-speckle noise. In this figure with all types of speckle-noisy images, it can be seen that the Canny takes the highest threshold values, LOG takes the lowest threshold values, Sobel, Prewitt almost take an identical threshold values and Roberts takes threshold values that is not so different from the Prewitt and Sobel threshold values.



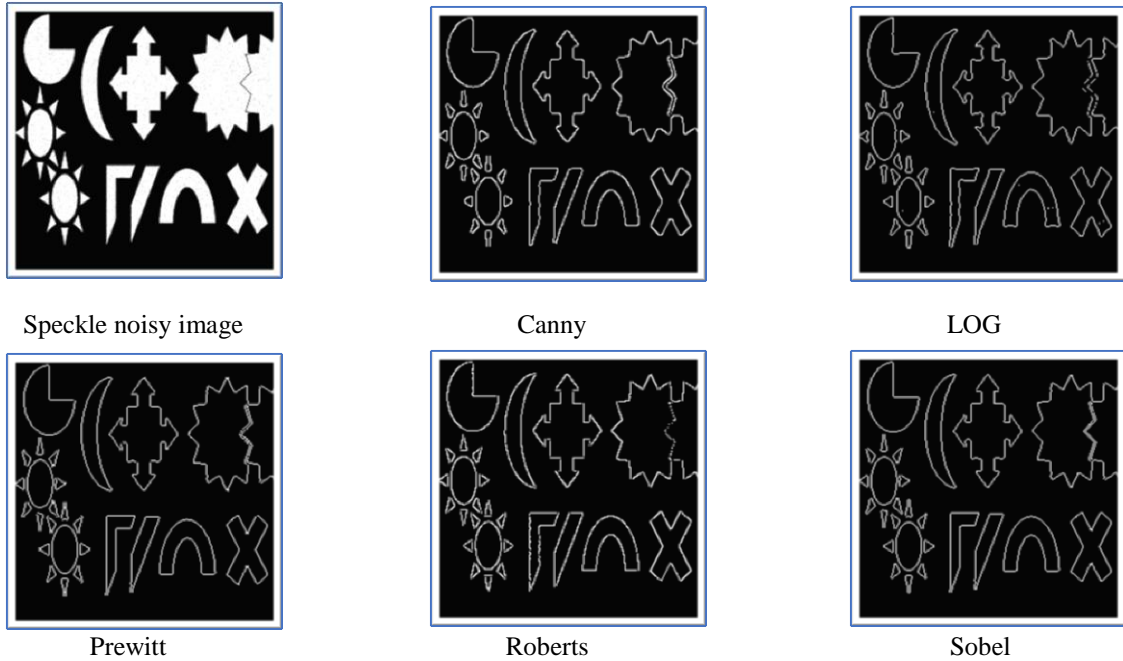
**Figure 4.24:** Threshold Values of Edge Detection Algorithms in Noisy Environment-Speckle Noise

Figure 4.25 is the correlation value between the speckle-noisy edge images and the clean edge images that came from the clean environment. In this figure with all types of speckle-noisy images the observation that can be seen that the correlation always is high with Canny algorithm and always is low with Roberts algorithm except when the input is the binary or texture image. All edge algorithms give a good detection when the input is binary image and Roberts gives good detection with the texture image.

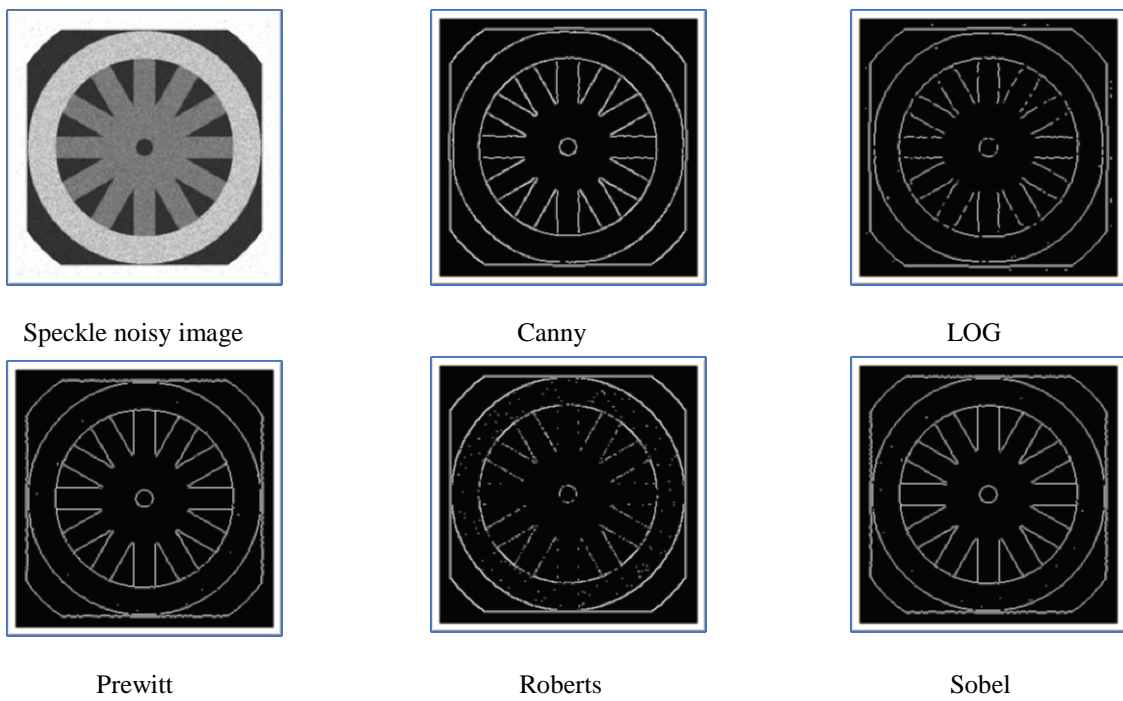


**Figure 4.25:** Correlation Values of Noisy Environment-Speckle Noise

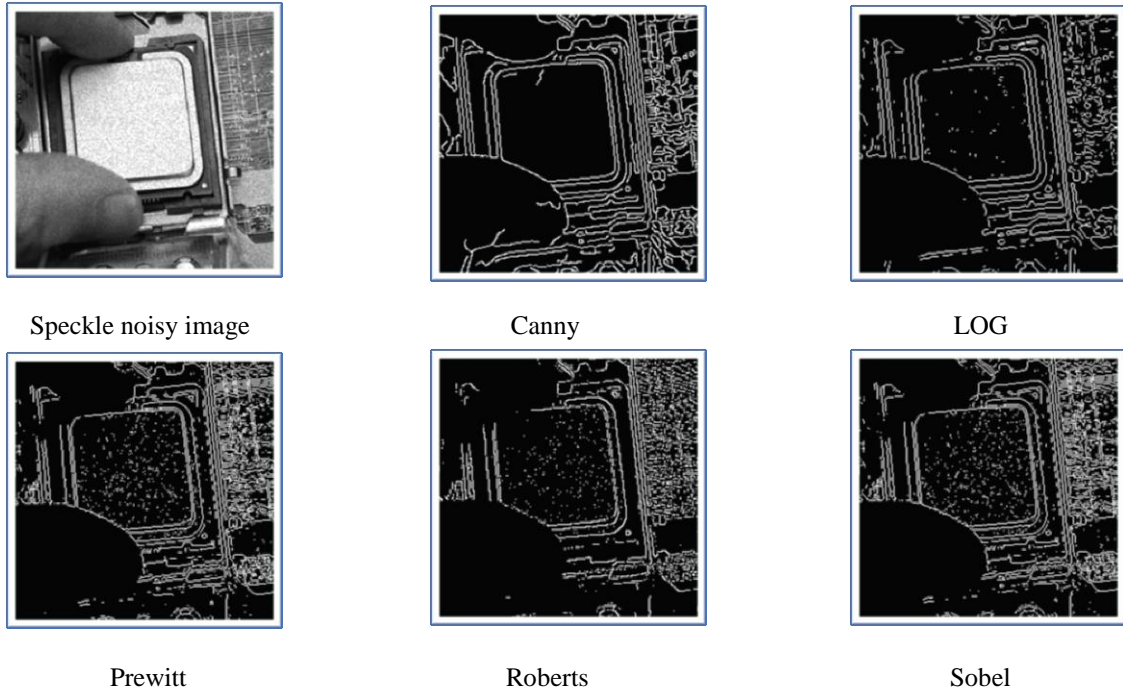
The figures from 4.26 to 4.31 are the edge detection images of the speckle noisy images. For the edge images figures 4.28, 4.29 and 4.30, it can be seen that the algorithms Roberts, Prewitt and Sobel give bad detection also it is possible to give a sequence to these algorithms from best to worst detection: Canny, LOG, Prewitt, Sobel and Roberts.



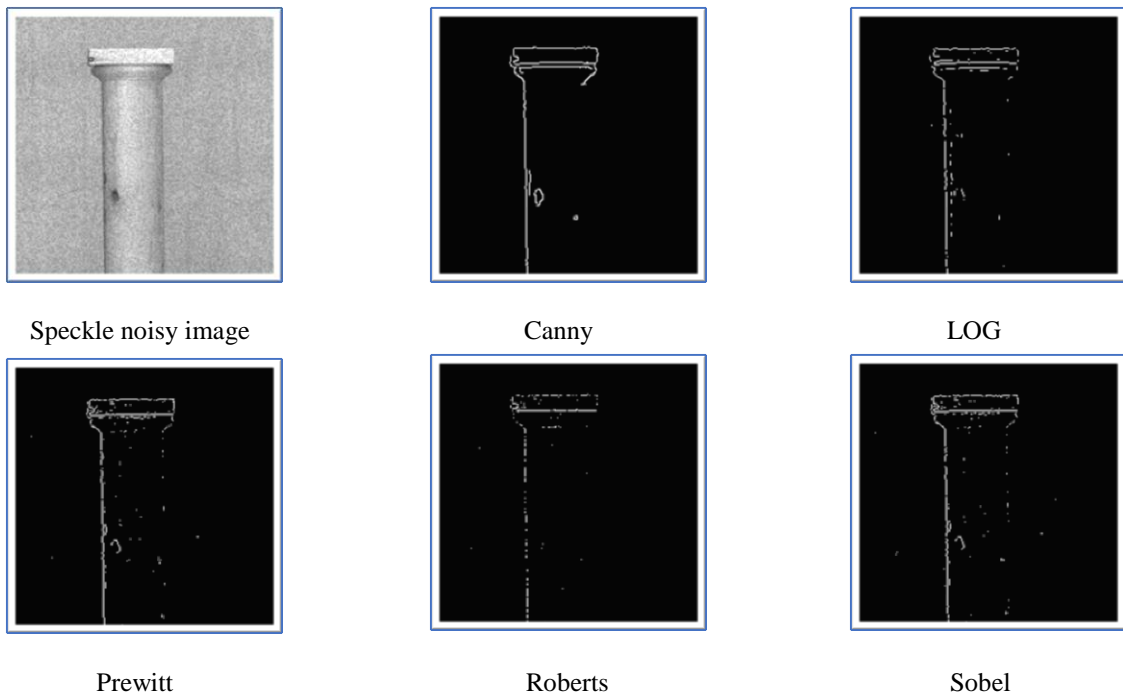
**Figure 4.26:** Edge Detection Images of Binary Image in Noisy Environment-Speckle Noise



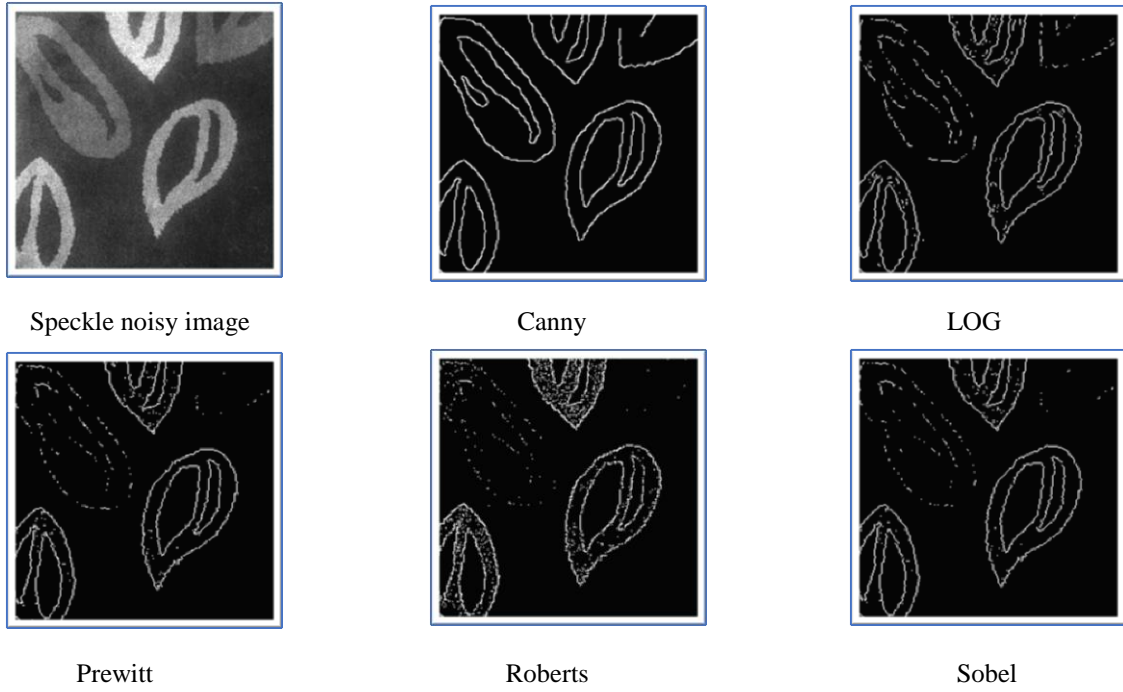
**Figure 4.27:** Edge Detection Images of Graphic Image in Noisy Environment-Speckle Noise



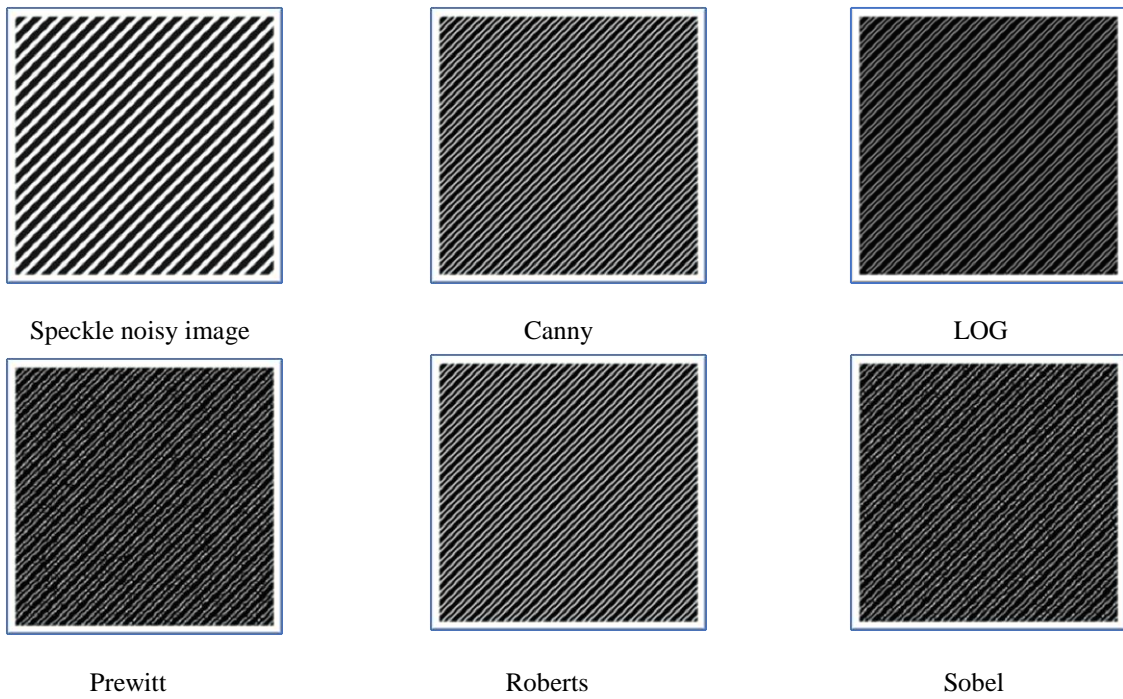
**Figure 4.28:** Edge Detection Images of High-Freq Image in Noisy Environment-Speckle Noise



**Figure 4.29:** Edge Detection Images of Low-Freq Image in Noisy Environment-Speckle Noise



**Figure 4.30:** Edge Detection Images of Median-Freq Image in Noisy Environment-Speckle Noise



**Figure 4.31:** Edge Detection Images of Texture Image in Noisy Environment-Speckle Noise

### 4.3 De-noising Environment

This section is divided into two sub-sections the first sub-section contains the results of noise removal algorithms and the second sub-section contains the results of de-noising environment.

#### 4.3.1 Results of noise removal algorithms

Before showing the results of the de-nosing environment, a suitable filter for each type of noise must be chosen. Figure 4.32 shows the PSNR values between the original images and the noisy images.

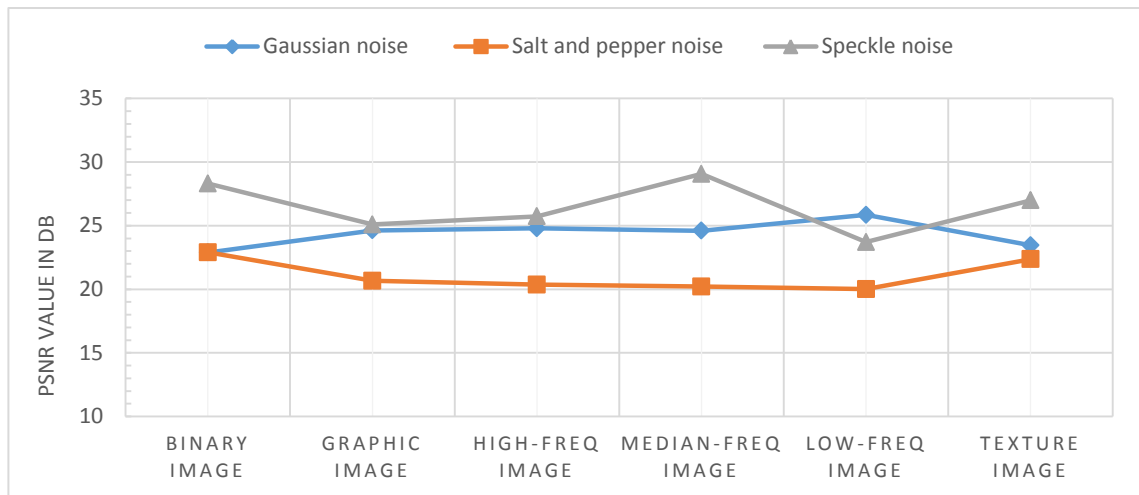
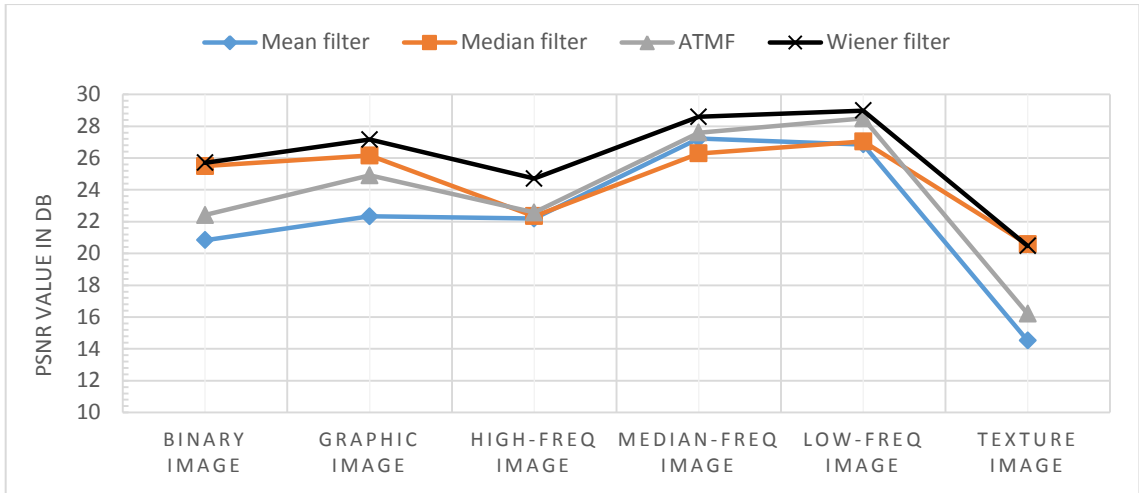


Figure 4.32: PSNR Values between the Original Images and the Noisy Images

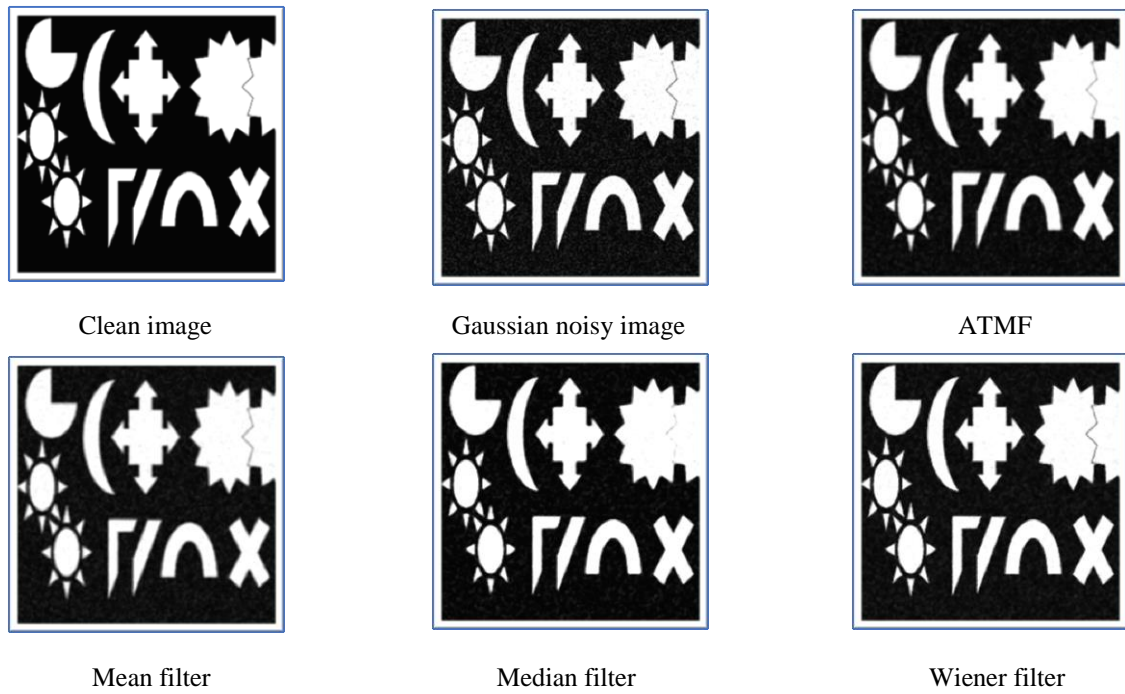
##### 4.3.1.1 Results of noise removal algorithms-gaussian noise

Figure 4.33 shows the PSNR between the original images and the de-noising images-Gaussian noise and it is clear in all types of images the Wiener filter always gives a high PSNR. The figures from 4.34 to 4.39 are the de-nosing images-Gaussian noise and the blurring effect is clear on it.

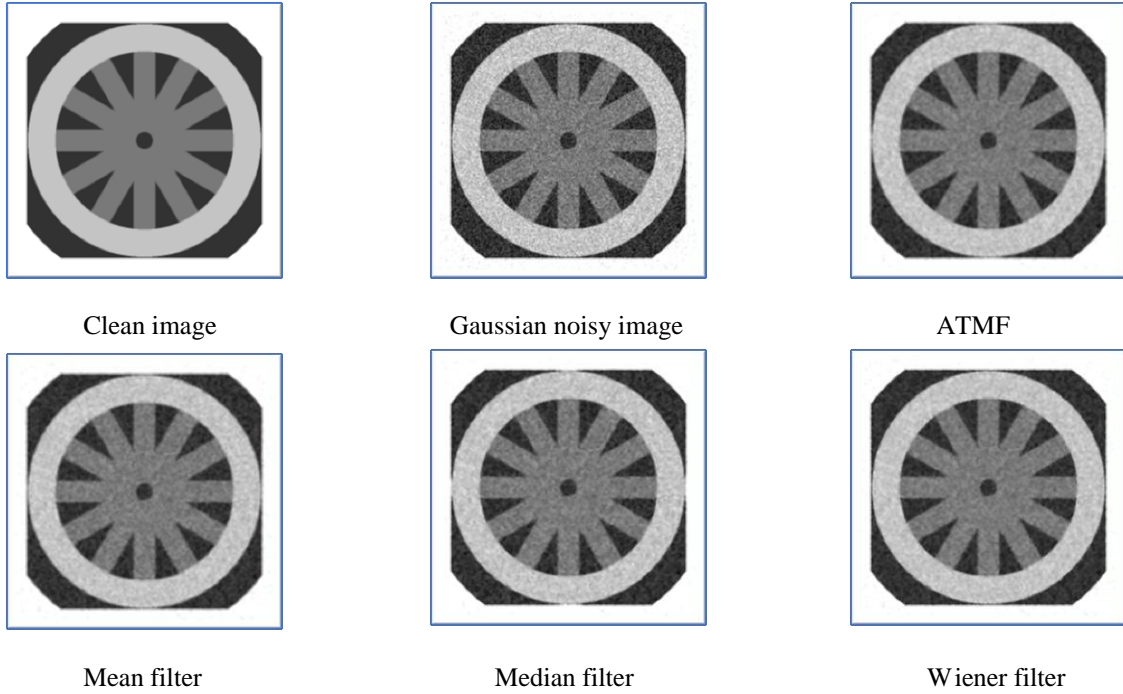




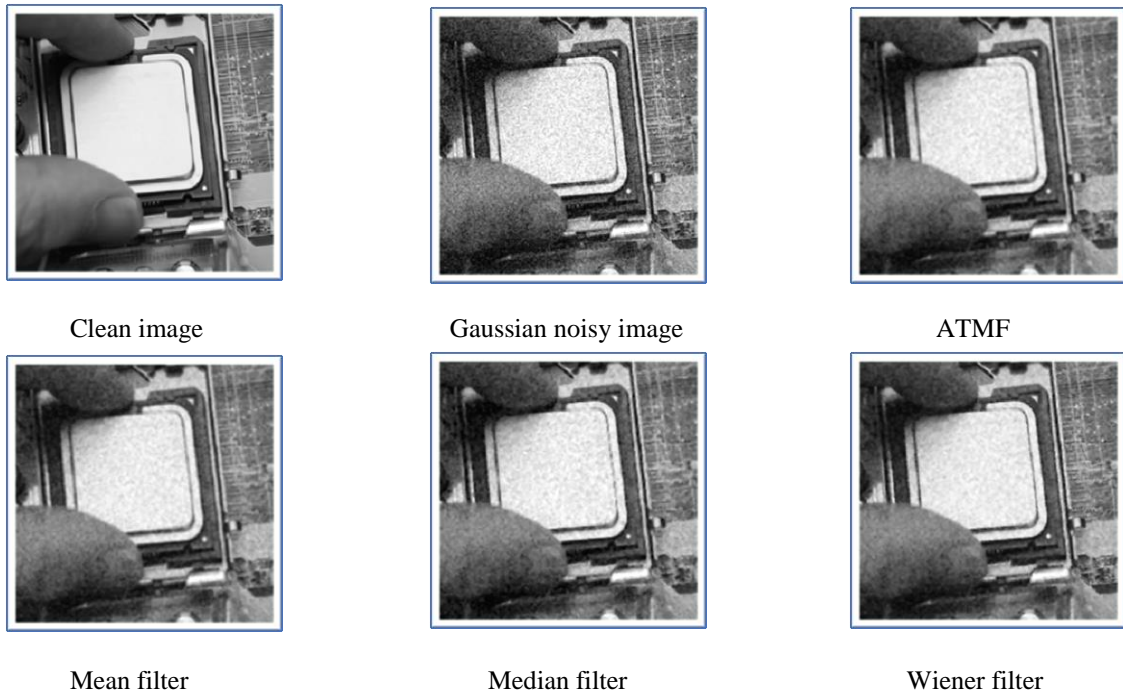
**Figure 4.33:** PSNR values between the original images and the de-noising images-Gaussian noise



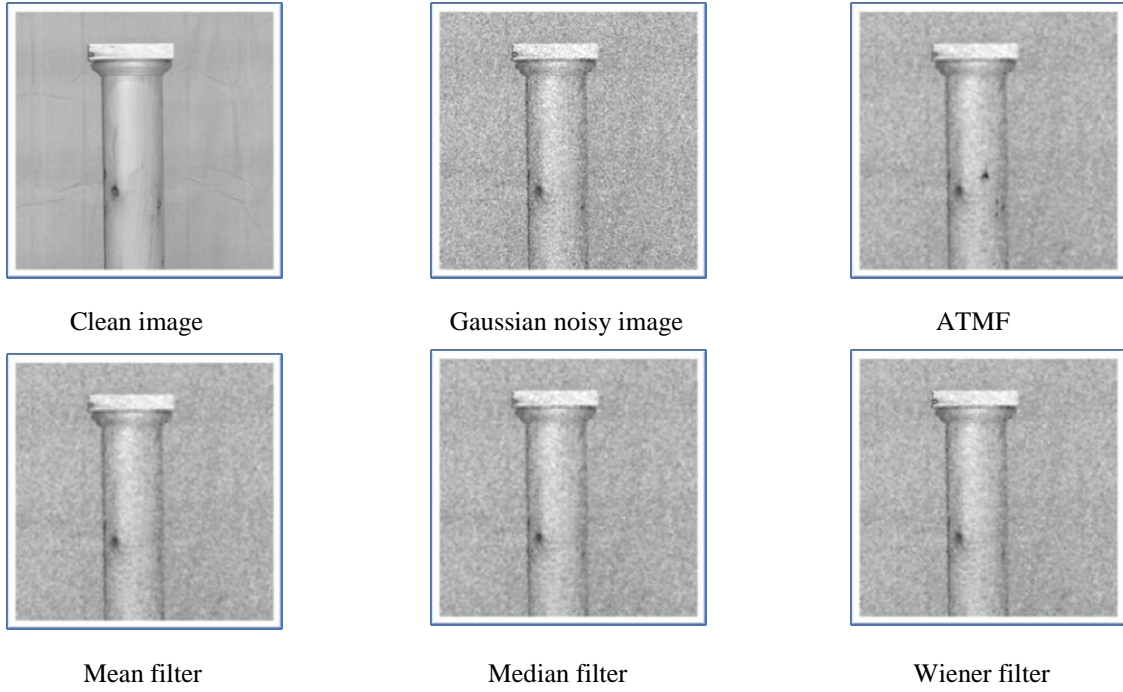
**Figure 4.34:** De-Noising Images of Binary Image-Gaussian Noise



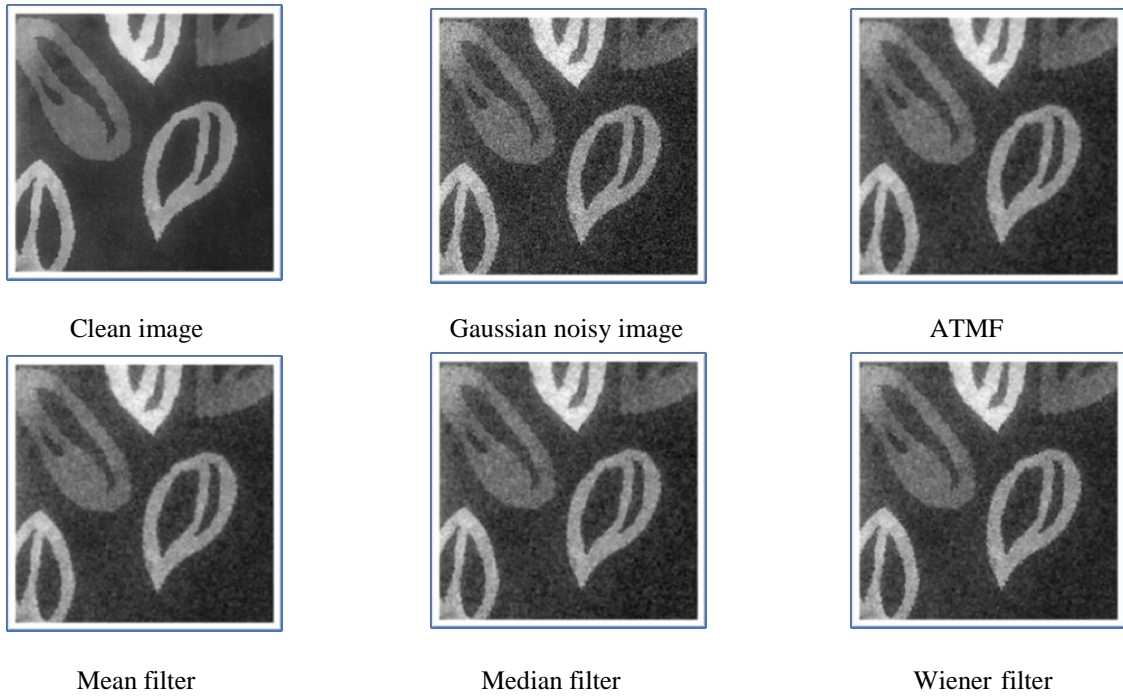
**Figure 4.35:** De-Noising Images of Graphic Image-Gaussian Noise



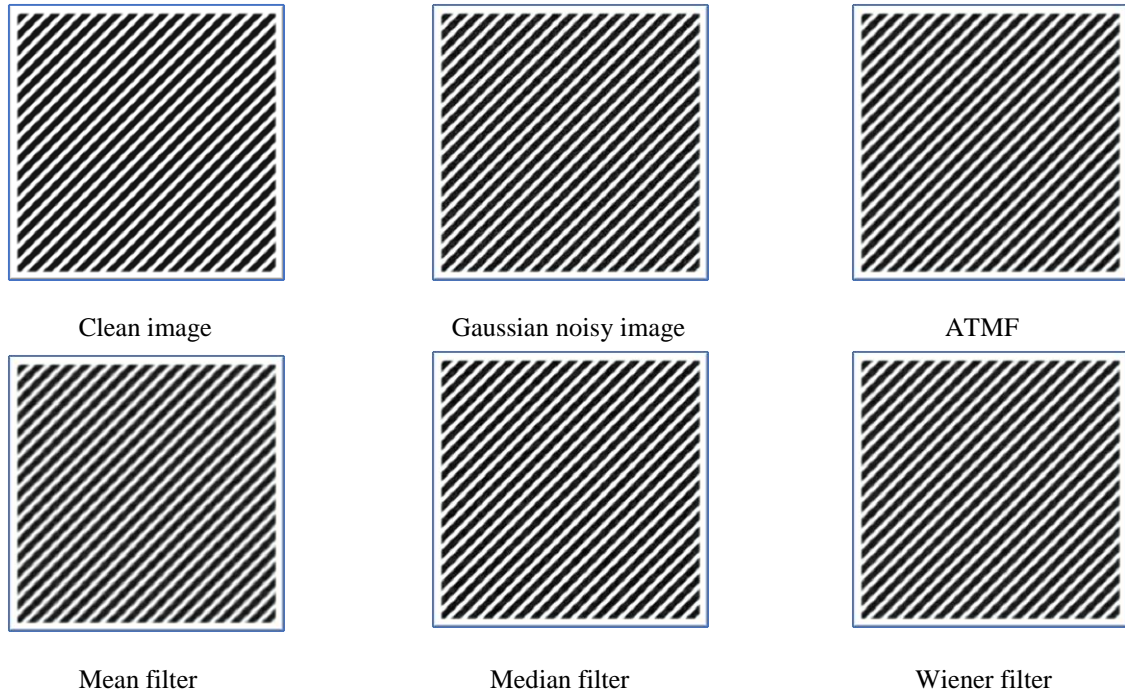
**Figure 4.36:** De-Noising Images of High-Freq Image-Gaussian Noise



**Figure 4.37:** De-Noising Images of Low-Freq Image-Gaussian Noise



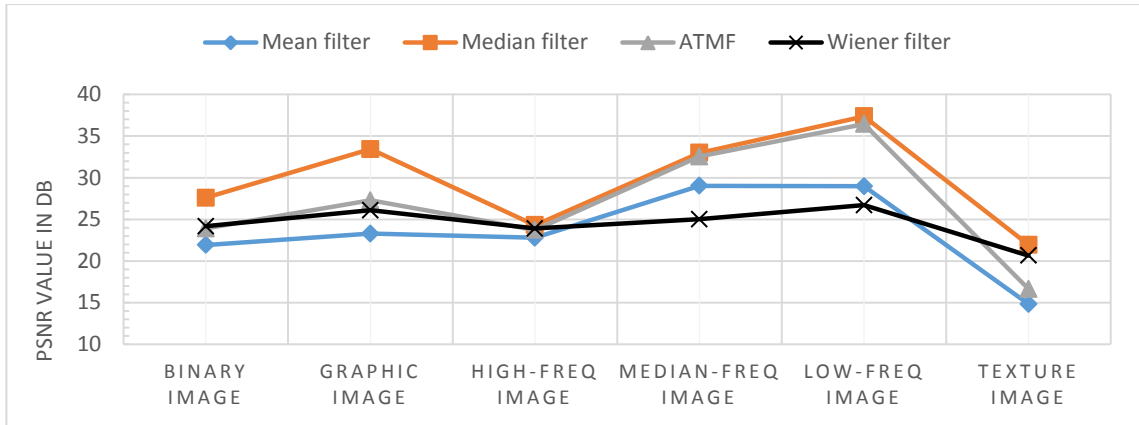
**Figure 4.38:** De-Noising Images of Median-Freq Image-Gaussian Noise



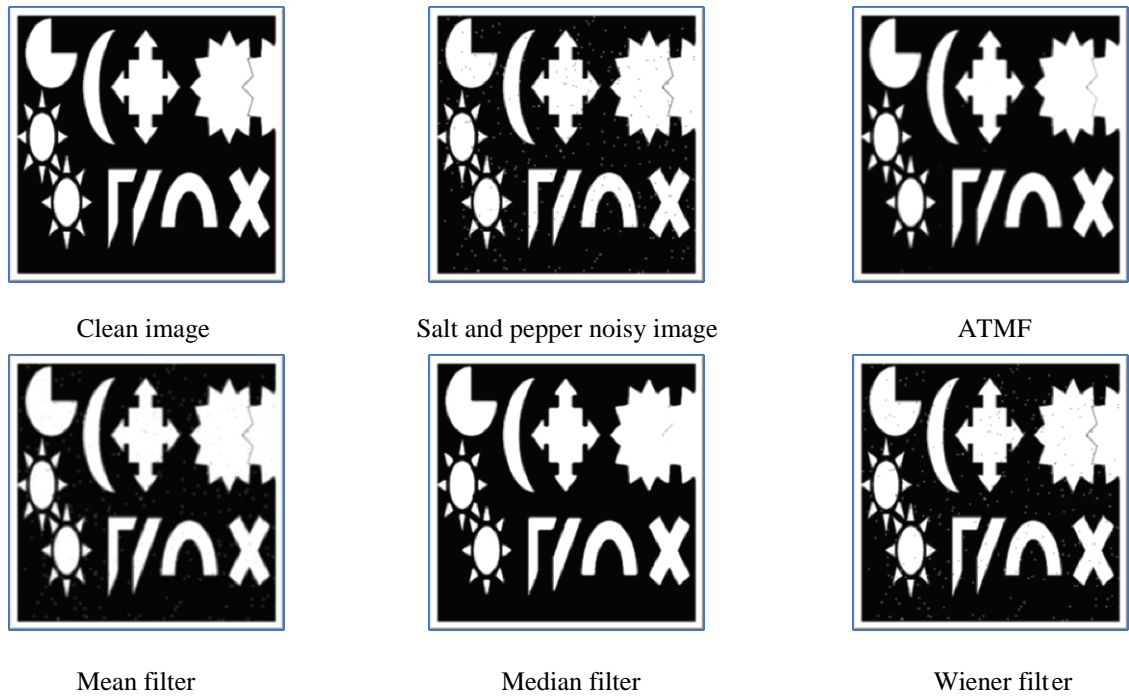
**Figure 4.39:** De-Noising Images of Texture Image-Gaussian Noise

#### 4.3.1.2 Results of noise removal algorithms-salt and pepper noise

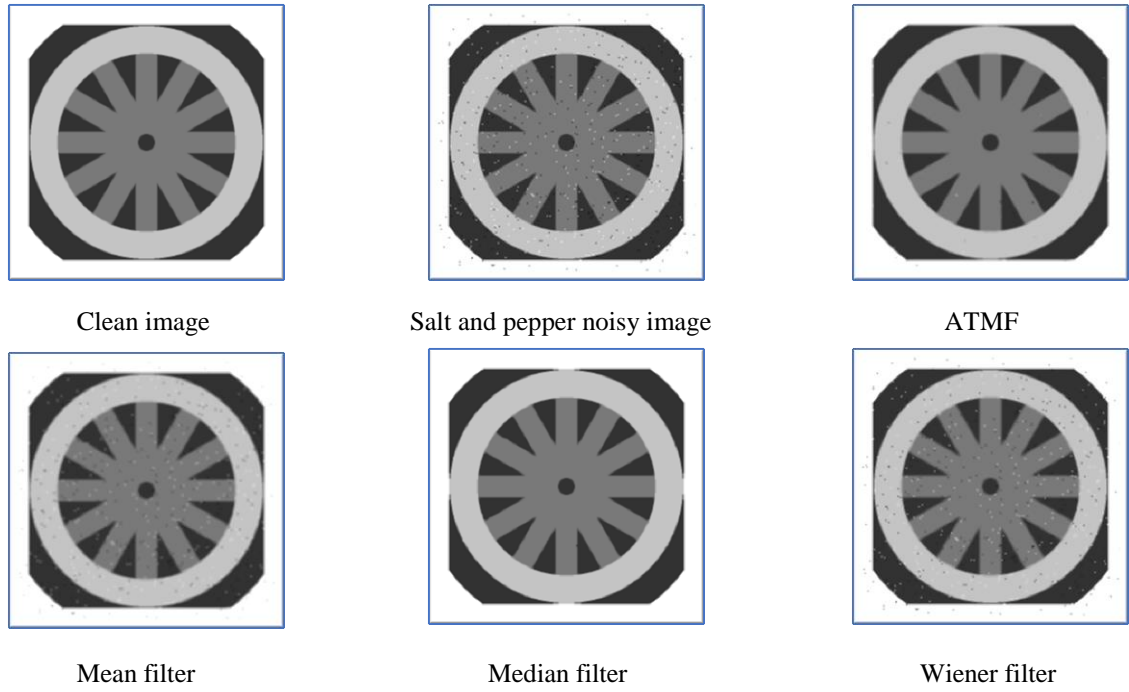
Figure 4.40 shows the PSNR between the original images and the de-noising images-salt and pepper noise and it is clear in all types of images the median filter always gives a high PSNR. The figures from 4.41 to 4.46 are the de-nosing images-salt and pepper noise from these figures it is clear that the de-noising images are enhanced too much and less blurring compared with Wiener filter.



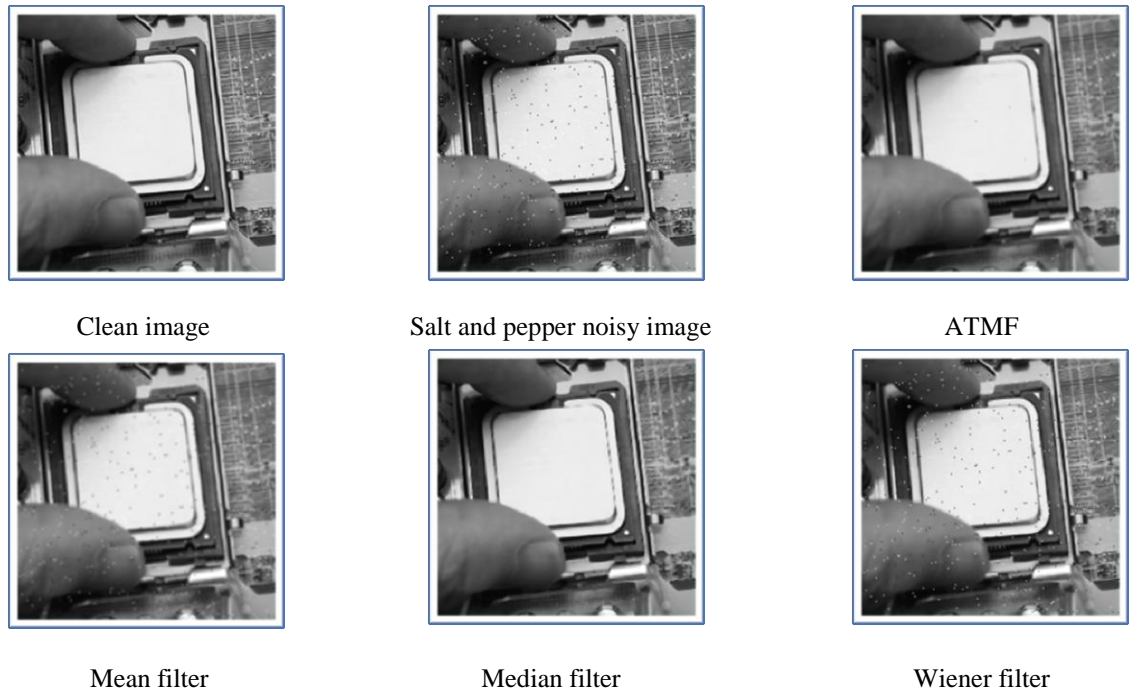
**Figure 4.40:** PSNR between the Original Images and the De-Noising Images-Salt and Pepper Noise



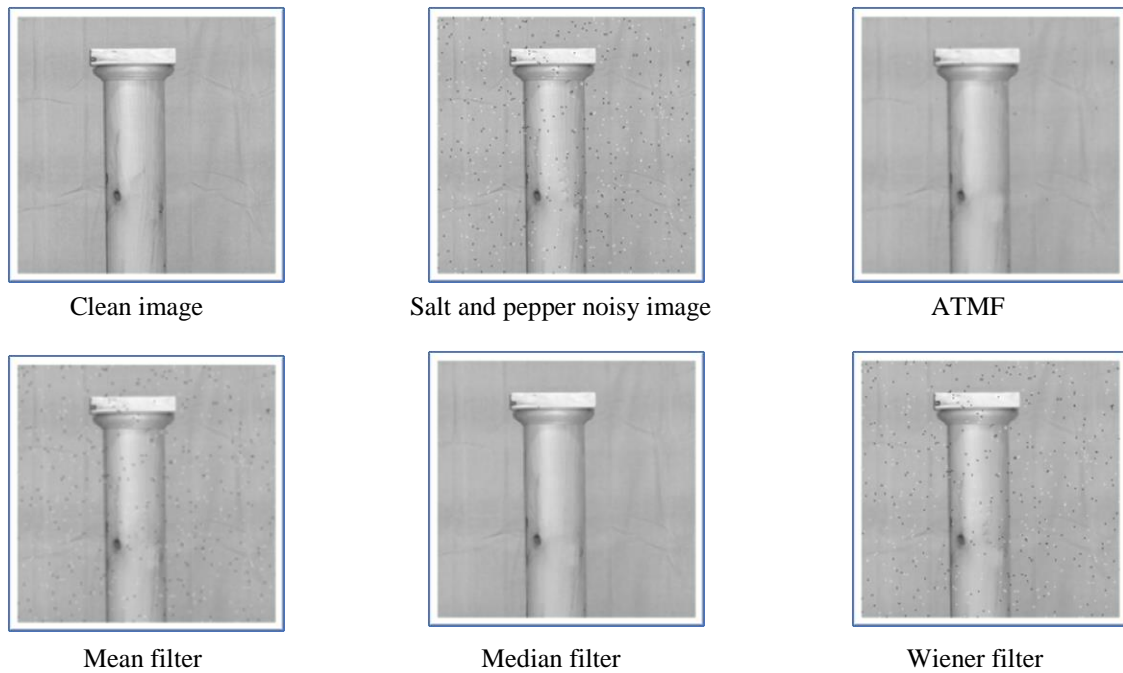
**Figure 4.41:** De-Noising Images of Binary Image-Salt and Pepper Noise



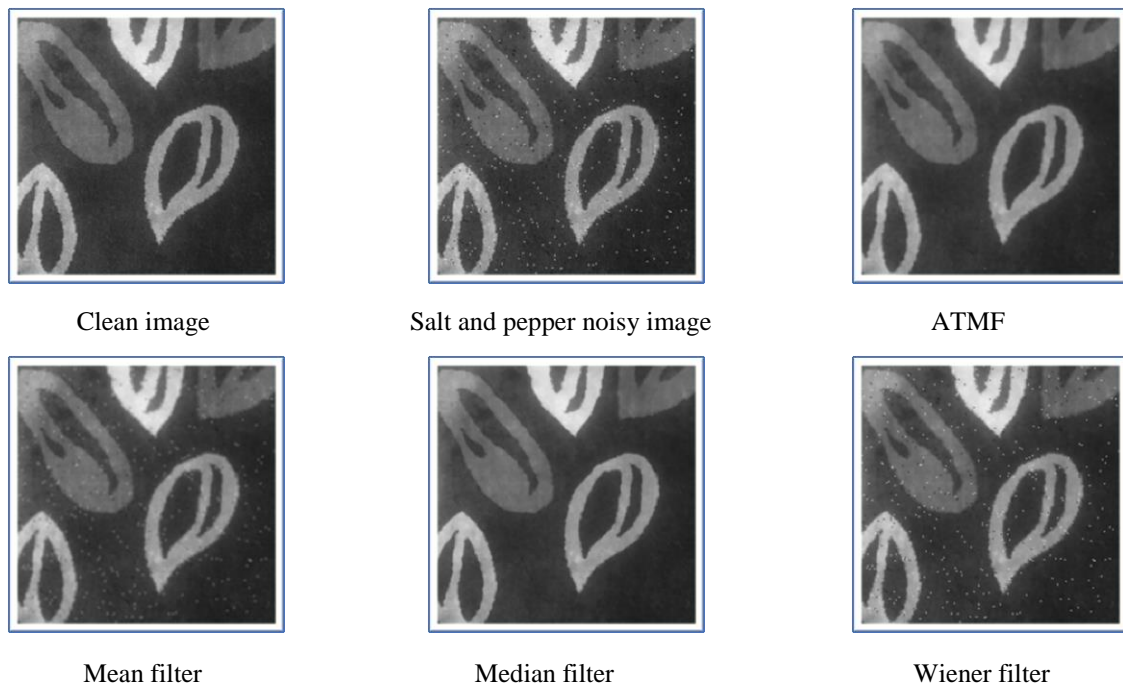
**Figure 4.42:** De-Noising Images of Graphic Image-Salt and Pepper Noise



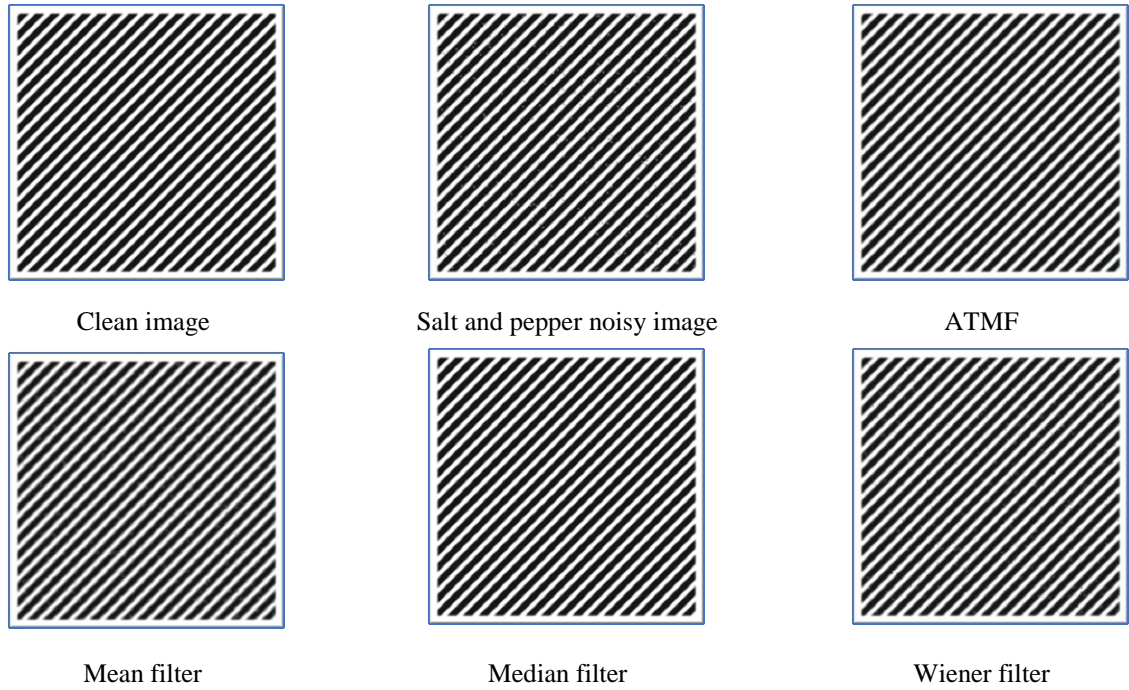
**Figure 4.43:** De-Noising Images of High-Freq Image-Salt and Pepper Noise



**Figure 4.44:** De-Noising Images of Low-Freq Image-Salt and Pepper Noise



**Figure 4.45:** De-Noising Images of Median-Freq Image-Salt and Pepper Noise

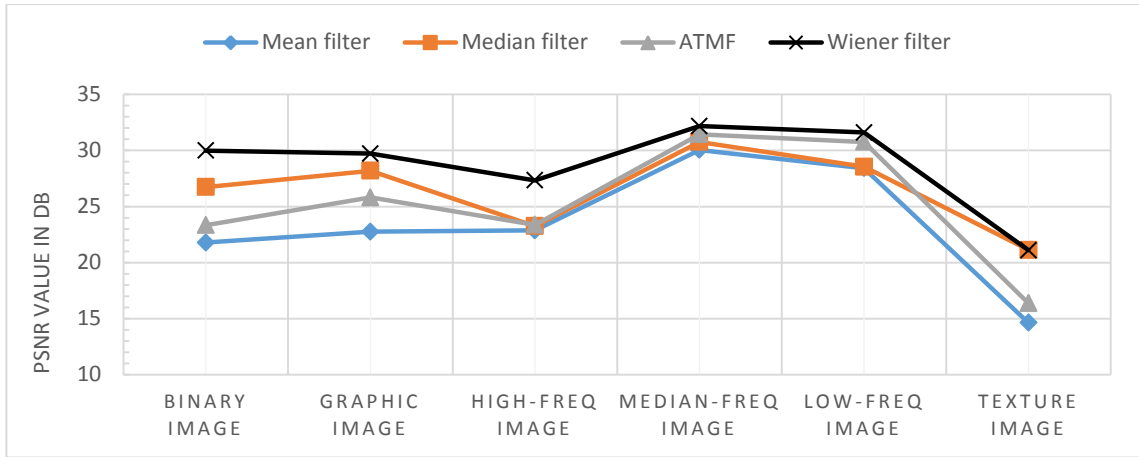


**Figure 4.46:** De-Noising Images of Texture Image-Salt and Pepper Noise

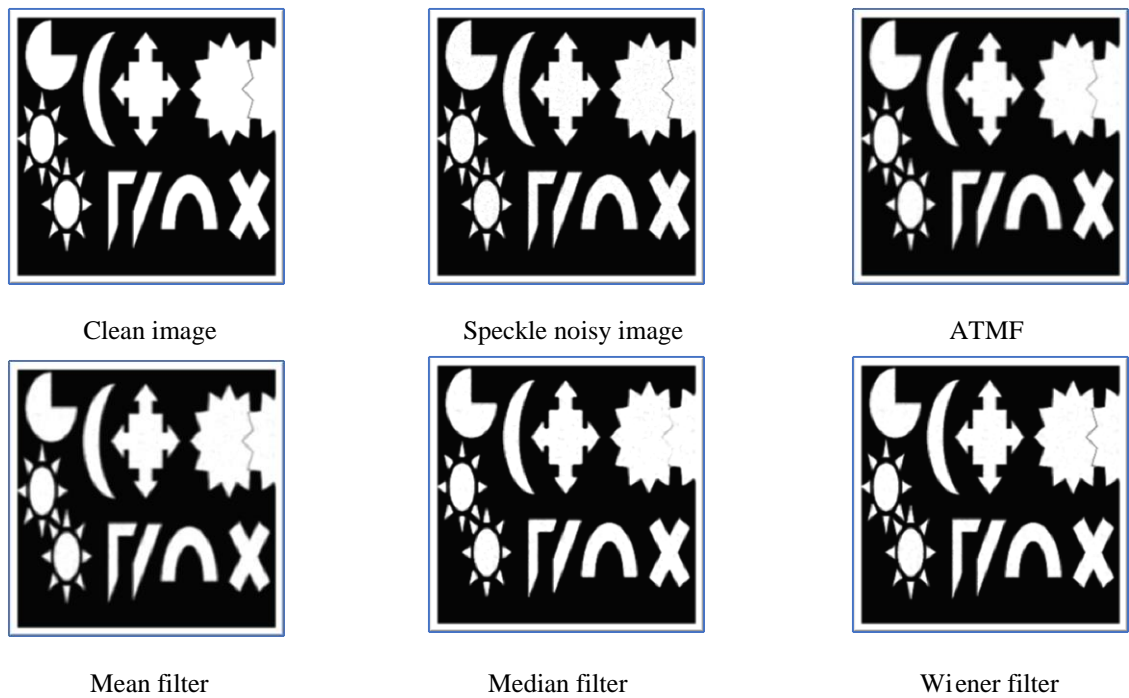
### 4.3.1.3 Results of noise removal algorithms-speckle noise

Figure 4.47 shows the PSNR between the original images and the de-noising images-speckle noise and it is clear in all types of images the Wiener filter always gives a high PSNR. The figures from 4.48 to 4.53 are the de-nosing images-speckle noise the blurring effect is clear on these images.

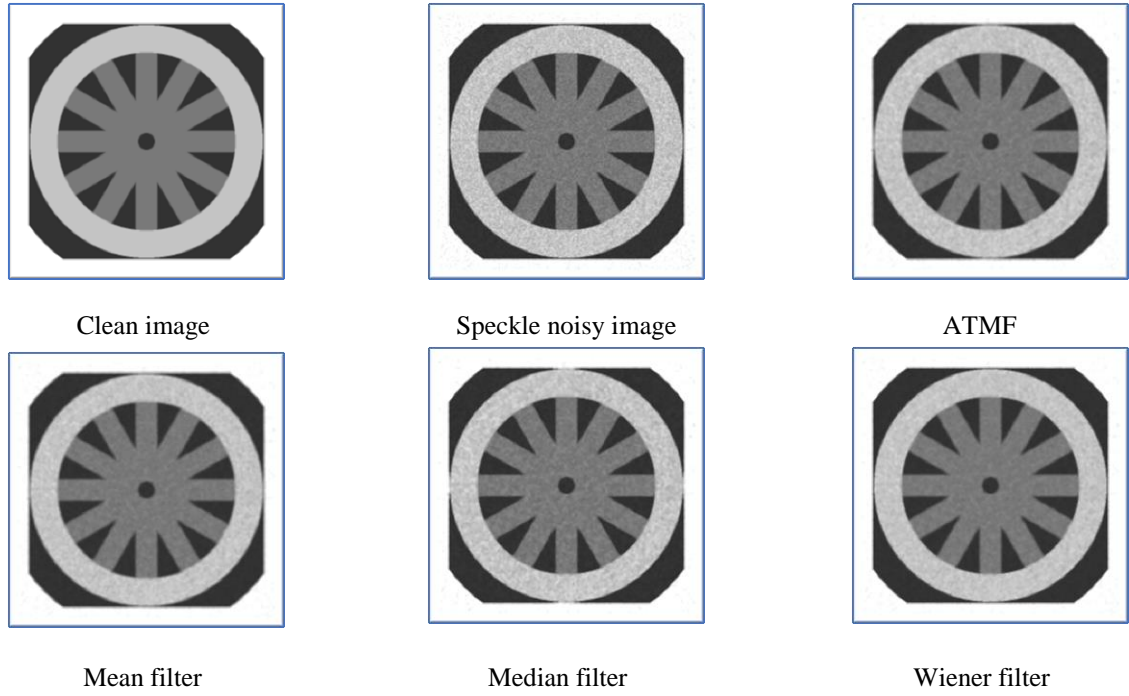




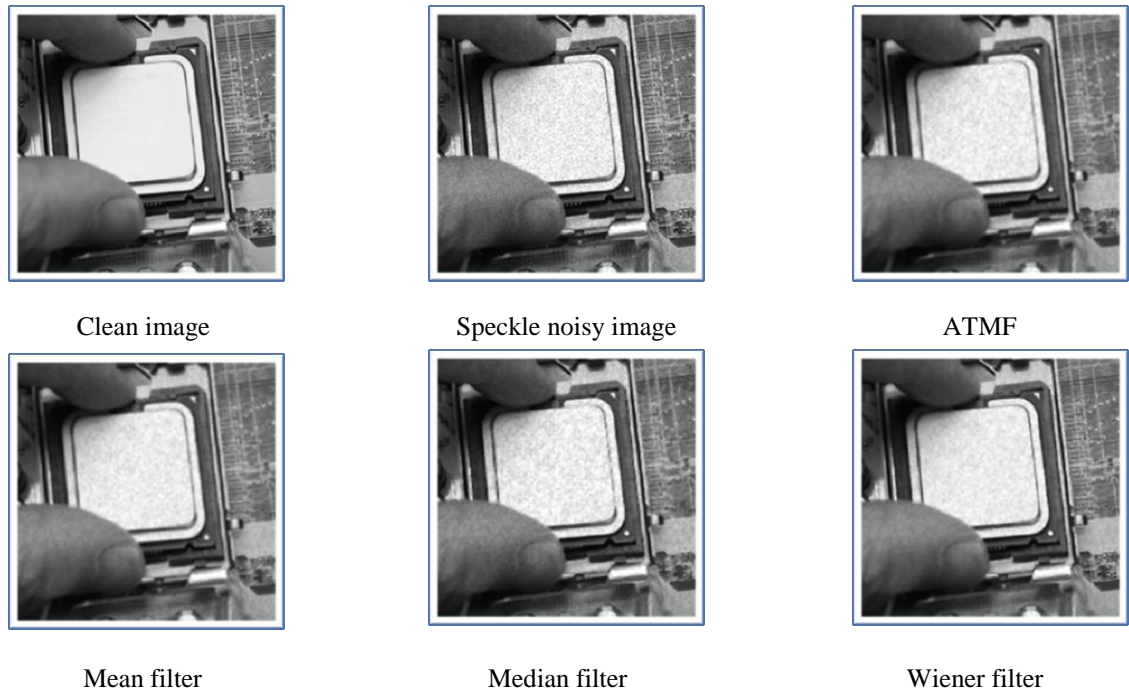
**Figure 4.47:** PSNR Values between the Original Images and the De-Noising Images-Speckle Noise



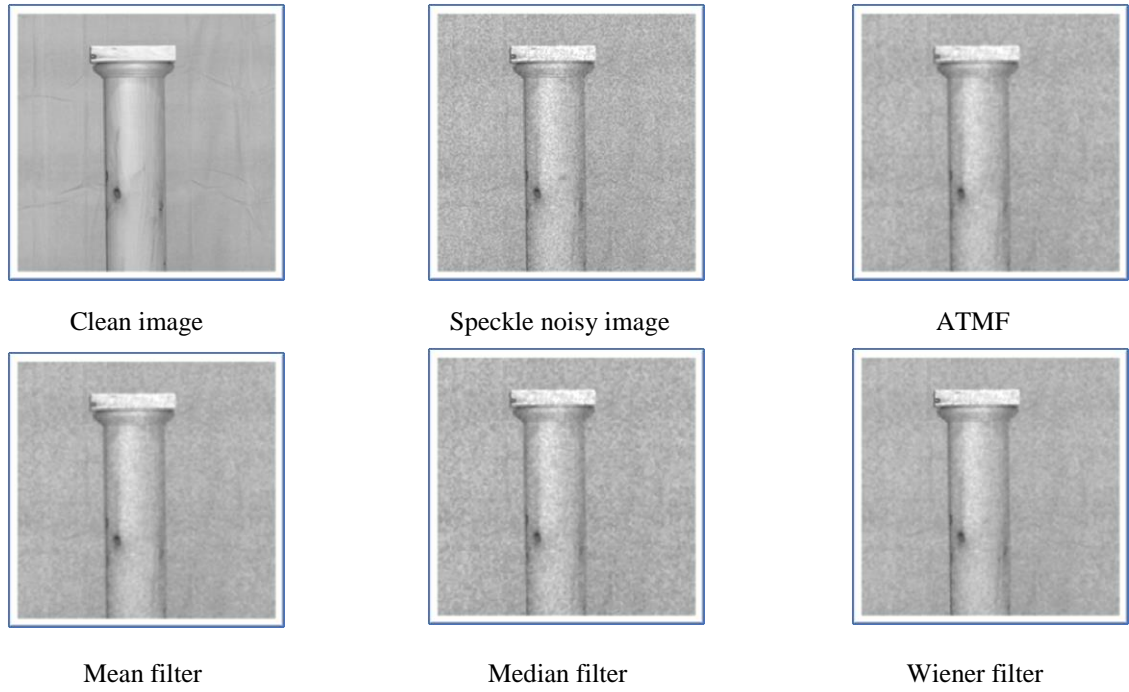
**Figure 4.48:** De-Noising Images of Binary Image-Speckle Noise



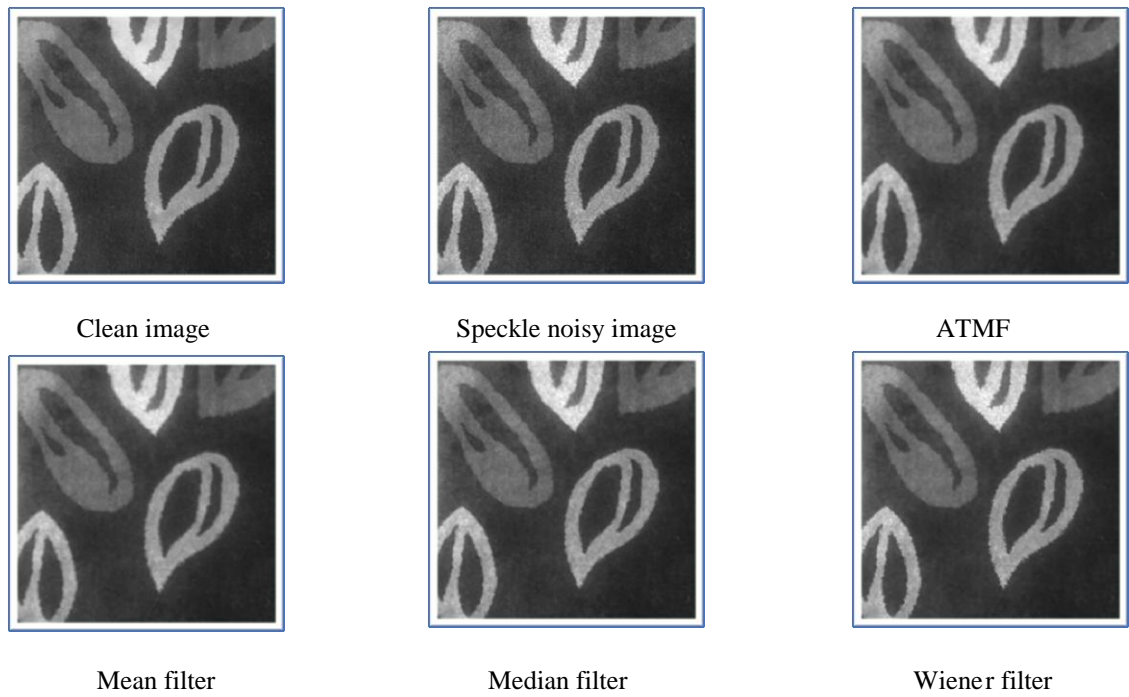
**Figure 4.49:** De-Noising Images of Graphic Image-Speckle Noise



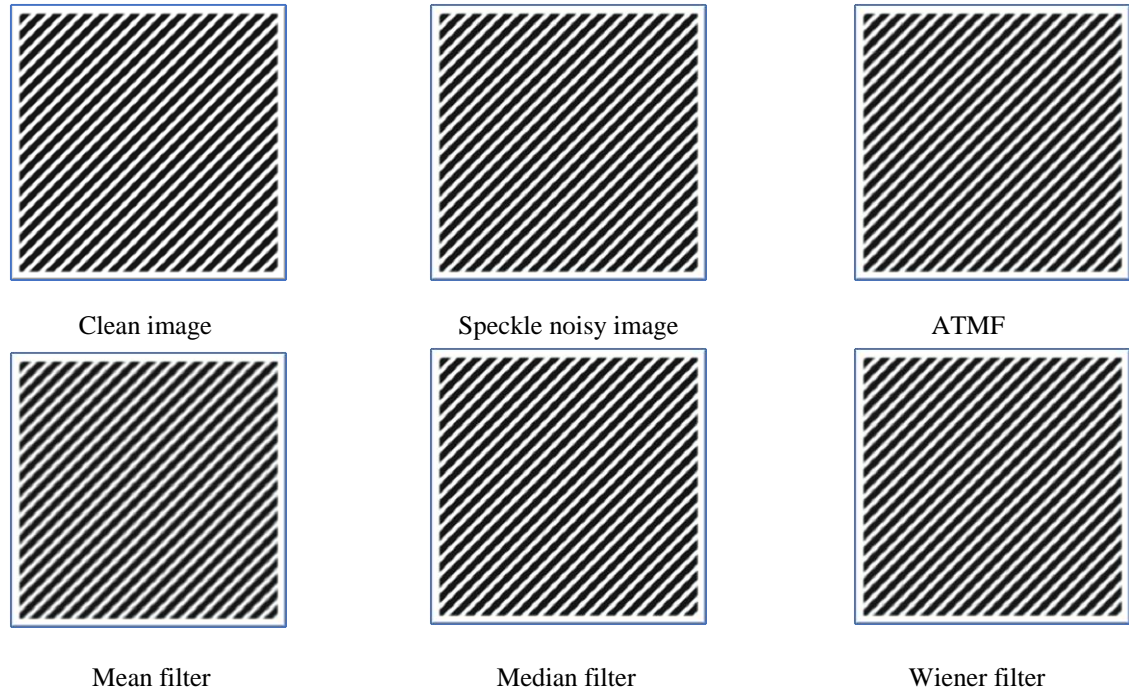
**Figure 4.50:** De-Noising Images of High-Freq Image-Speckle Noise



**Figure 4.51:** De-Noising Images of Low-Freq Image-Speckle Noise



**Figure 4.52:** De-Noising Images of Median-Freq Image-Speckle Noise



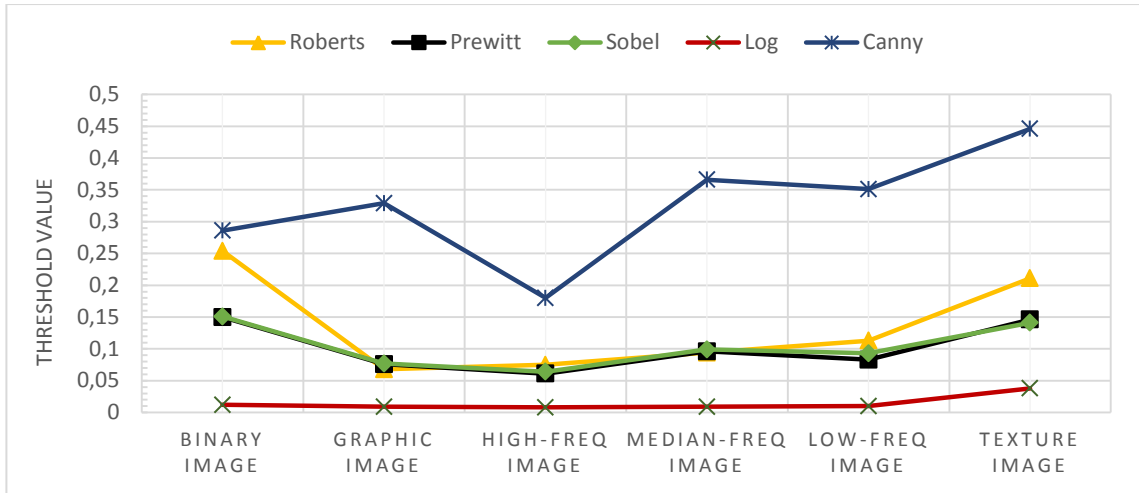
**Figure 4.53:** De-Noising Images of Texture Image-Speckle Noise

### 4.3.2 De-noising environment results

In this environment the source images are de-noised images, Depending on the results of the noise removal algorithms, the suitable filter for a Gaussian-noisy image or Speckle-noisy image is the Wiener filter and the median filter for salt-and-pepper noisy image. This section is divided into three parts depending on the type of the de-noised images.

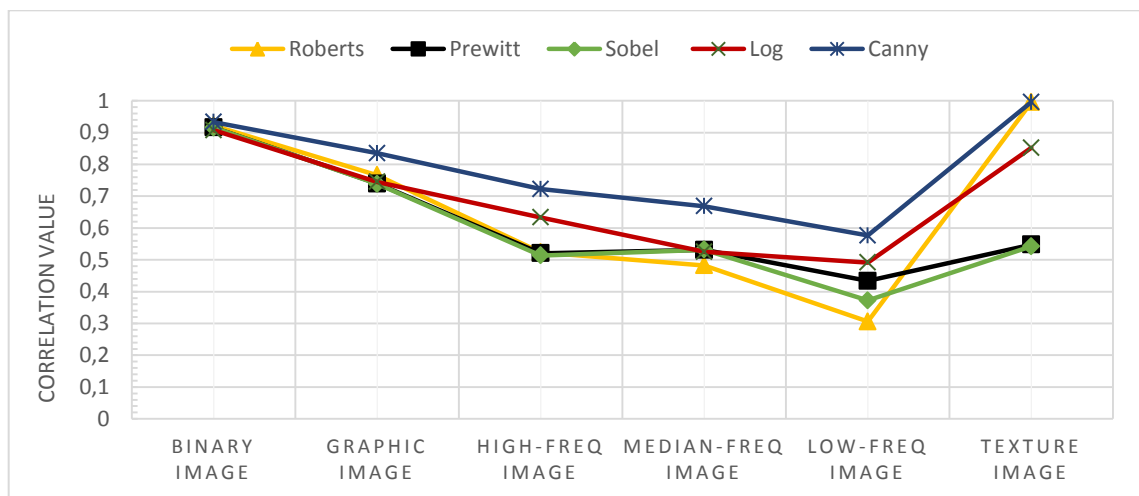
#### 4.3.2.1 De-nosing environment results-gaussian de-noised images

Figure 4.54 is the threshold values of the edge detection algorithms in the de-noising environment-Gaussian de-noised images. In this figure with all types of Gaussian de-noised images, it can be seen that the Canny takes the highest threshold values, LOG takes the lowest threshold values, Sobel, Prewitt almost take an identical threshold values and Roberts takes threshold values that is not so different from the Prewitt and Sobel threshold values.



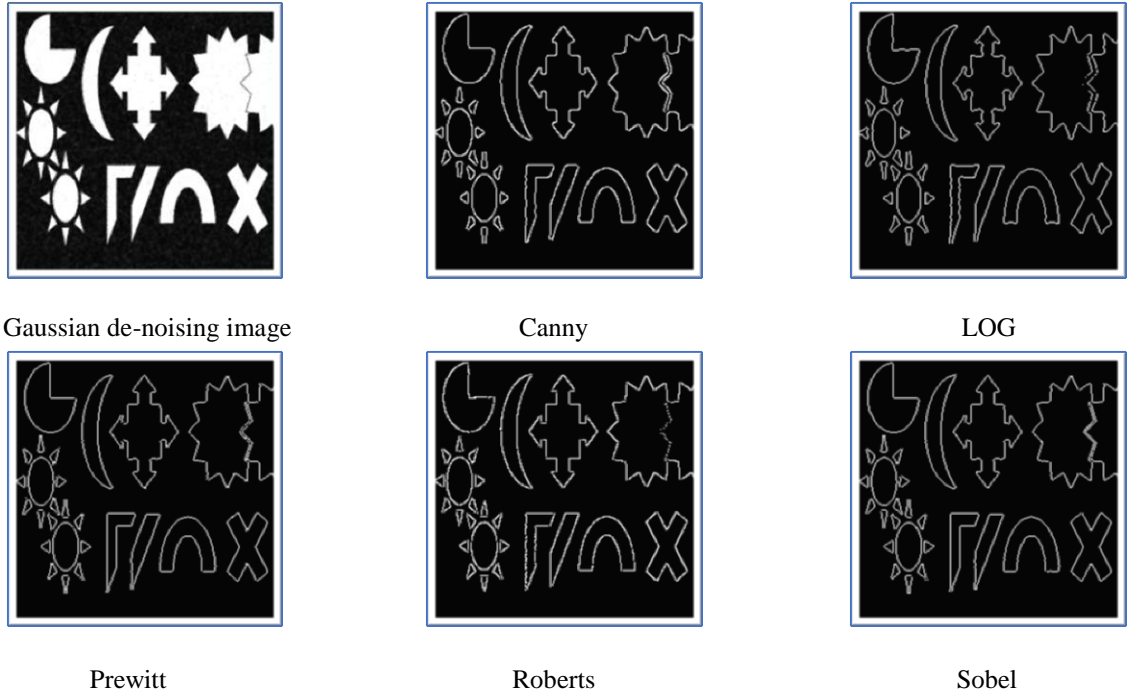
**Figure 4.54:** Threshold Values of Edge Detection Algorithms in De-Noising Environment-Gaussian Noise

Figure 4.55 is the correlation values between the edge images of Gaussian de-noised images and the clean edge images that came from the clean environment. In this figure with all types of Gaussian-de-noised images the observation that can be seen that the correlation always is high with Canny algorithm and always is low with Roberts algorithm except when the input is the binary or texture image. All edge algorithms give a good detection when the input is binary image and Roberts gives good detection with the texture image.

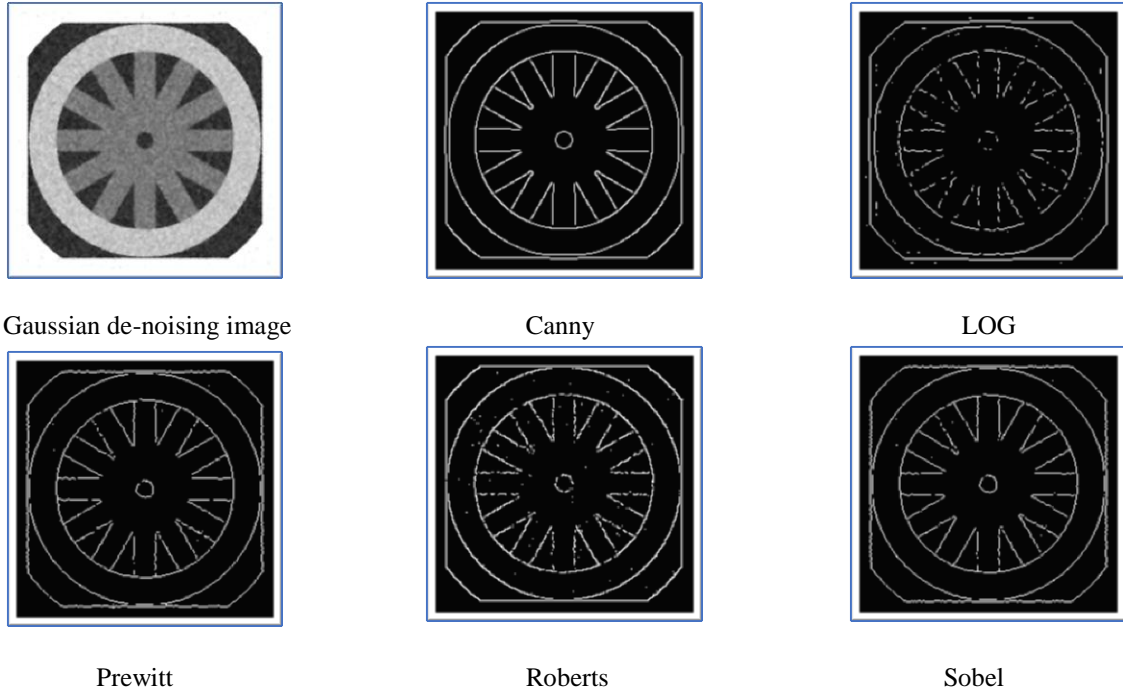


**Figure 4.55:** Correlation Values of De-Nosing Environment-Gaussian Noise

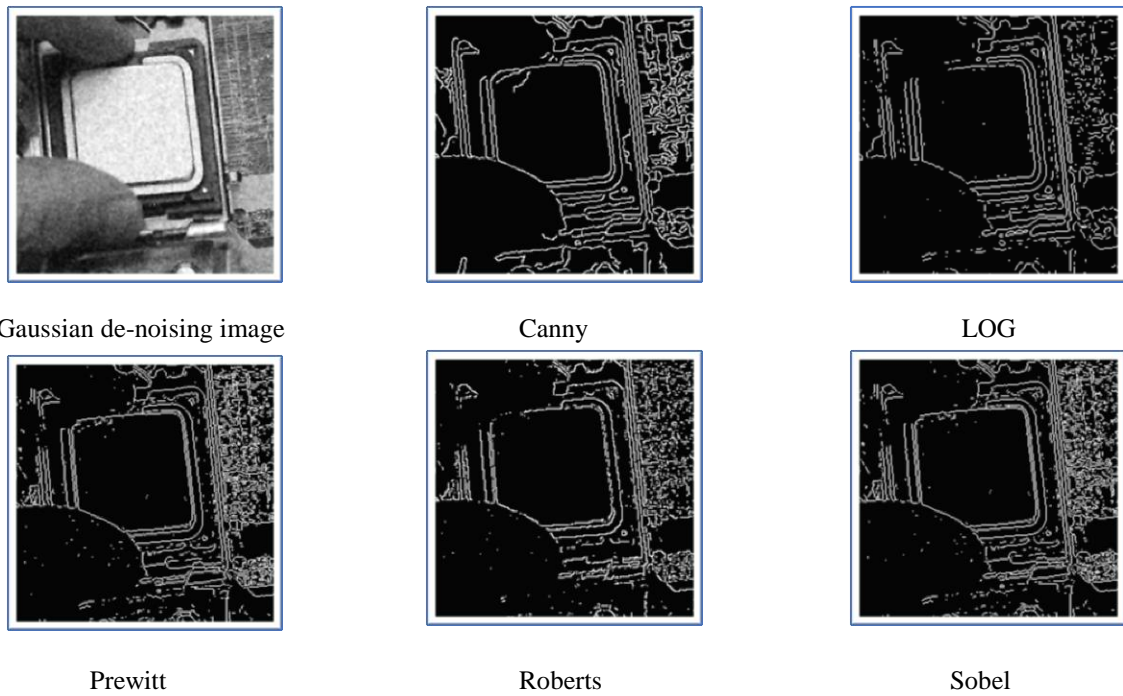
The figures from 4.56 to 4.61 are the edge detection images of the Gaussian de-noised images and these images are enhanced compared with the edge images of the noisy environment. For the edge images figures 4.11, 4.12, 4.13 and 4.14, it can be seen that the algorithms Roberts, Prewitt and Sobel still give not good detection although the images was enhanced, also it is possible to give a sequence to these algorithms from best to worst detection: Canny, LOG, Prewitt, Sobel and Roberts.



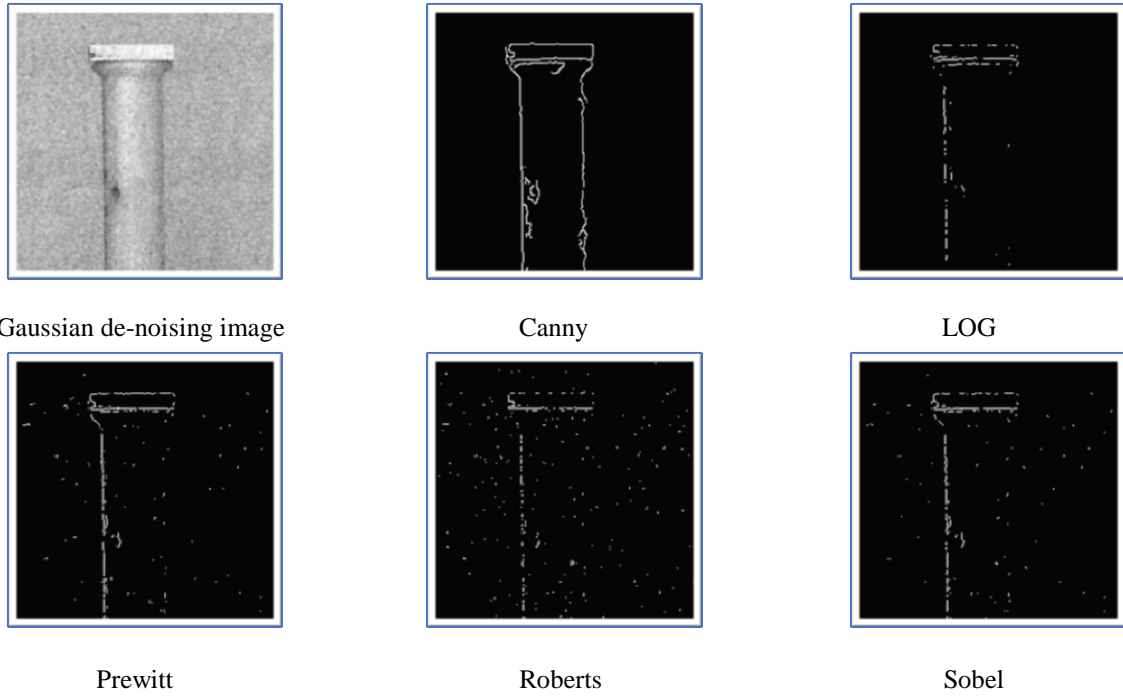
**Figure 4.56:** Edge Detection Images of Binary Image in De-Noising Environment-Gaussian Noise



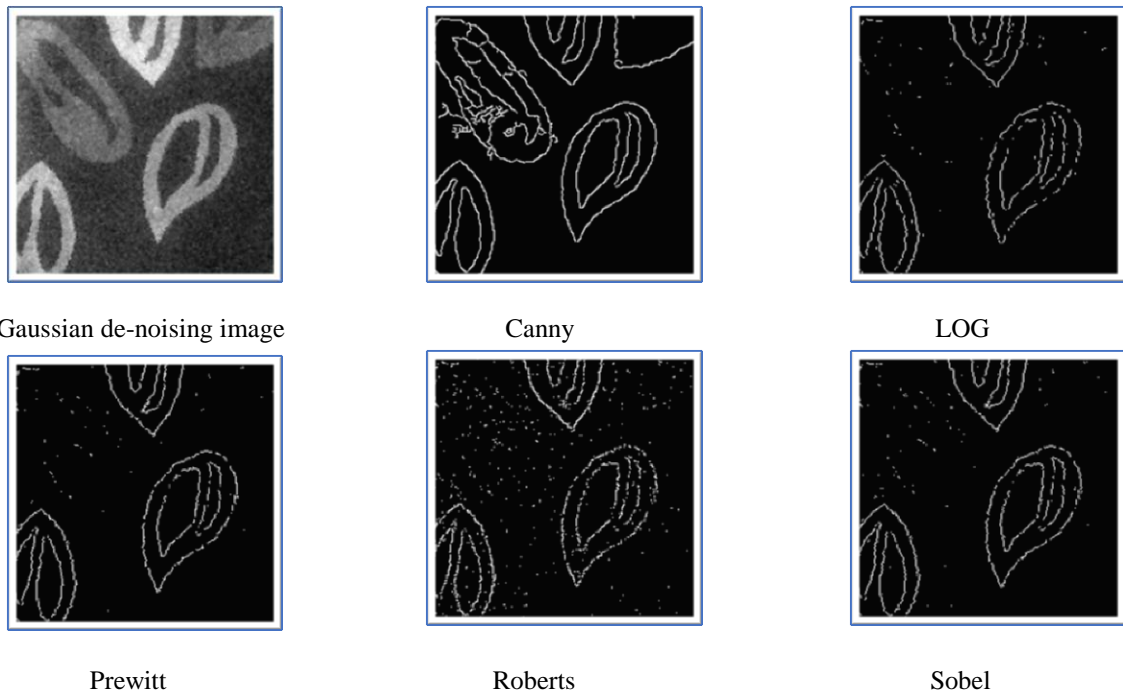
**Figure 4.57:** Edge Detection Images of Graphic Image in De-Noising Environment-Gaussian Noise



**Figure 4.58:** Edge Detection Images of High-Freq Image in De-Noising Environment-Gaussian Noise

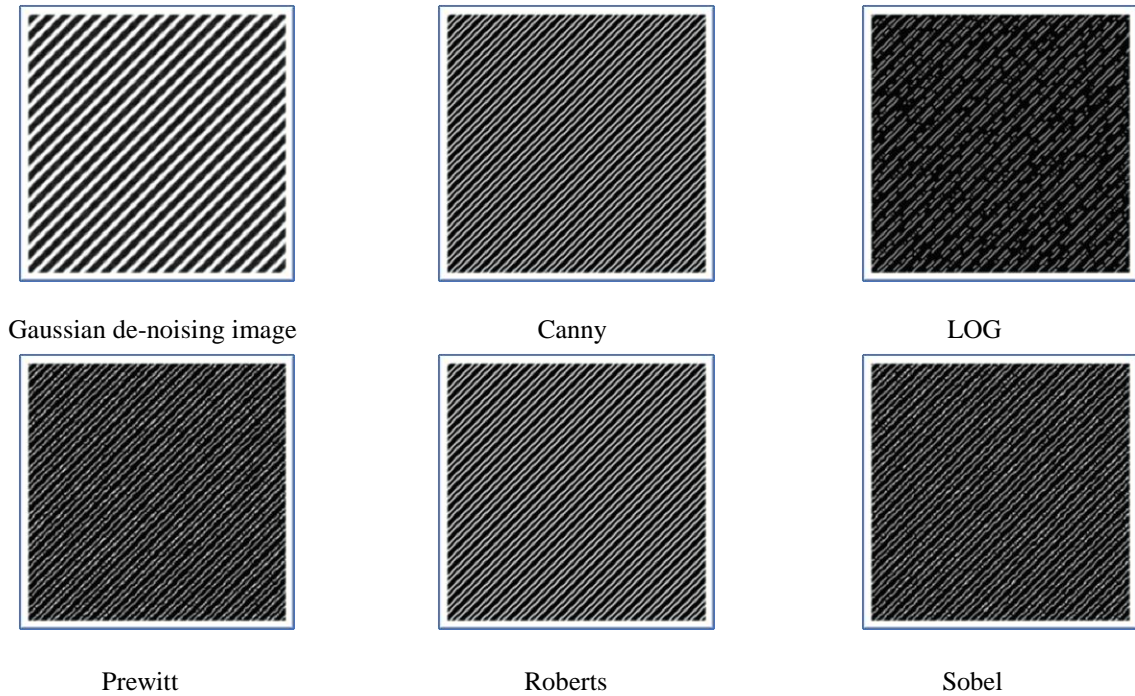


**Figure 4.59:** Edge Detection Images of Low-Freq Image in De-Noising Environment-Gaussian Noise



**Figure 4.60:** Edge Detection Images of Median-Freq Image in De-Noising Environment-Gaussian Noise





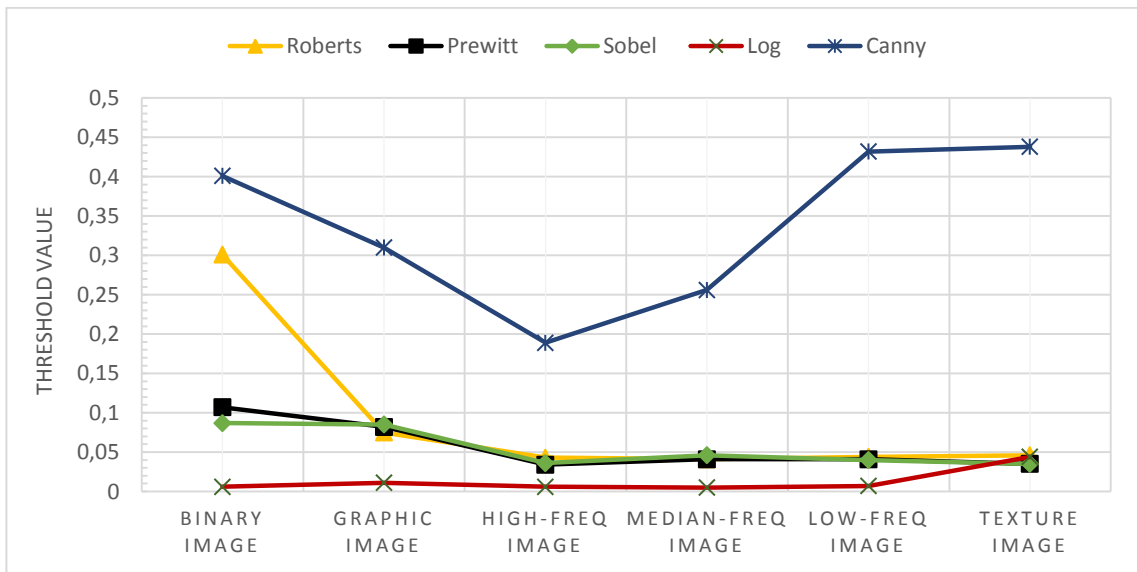
**Figure 4.61:** Edge Detection Images of Texture Image in De-Noising Environment-Gaussian Noise

#### 4.3.2.2 De-nosing environment results-salt and pepper de-noised images

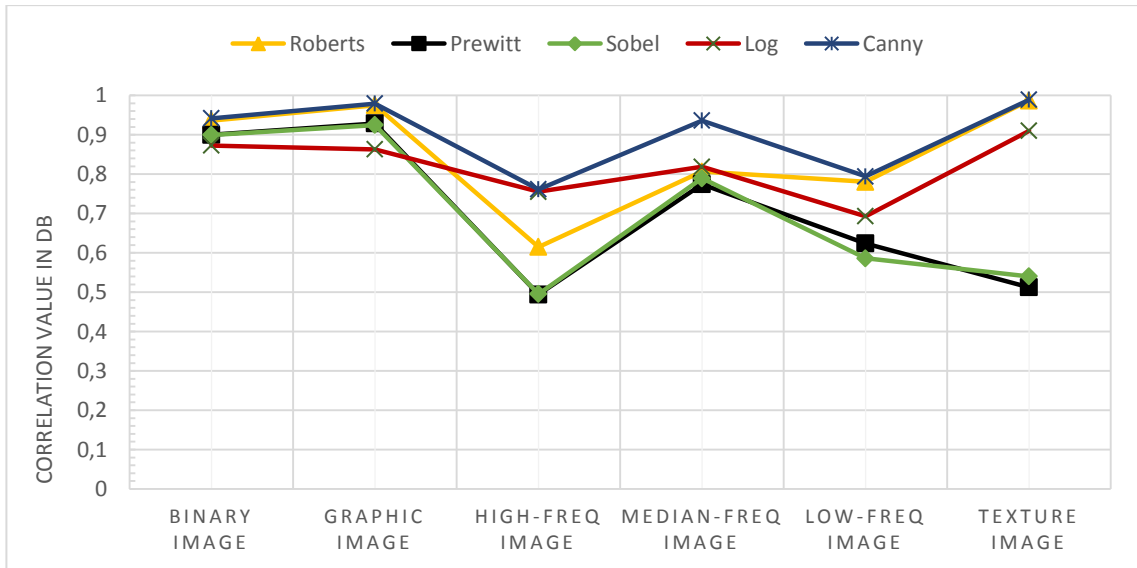
Figure 4.62 is the threshold values of the edge detection algorithms in the de-noising environment-salt and pepper de-noised images. In this figure with all types of salt and pepper de-noised images, it can be seen that the Canny takes the highest threshold values, LOG takes the lowest threshold values, Sobel, Prewitt almost take an identical threshold values and Roberts takes threshold values that is not so different from the Prewitt and Sobel threshold values.

Figure 4.63 is the correlation values between the edge images of salt and pepper de-noised images and the clean edge images that came from the clean environment. In this figure with all types of salt and pepper-de-noised images the observation that can be seen that the correlation always is high with Canny algorithm and always is low with Sobel algorithm except when the input is the binary. All edge algorithms give a good

detection when the input is binary image and Roberts gives good detection with the texture image.



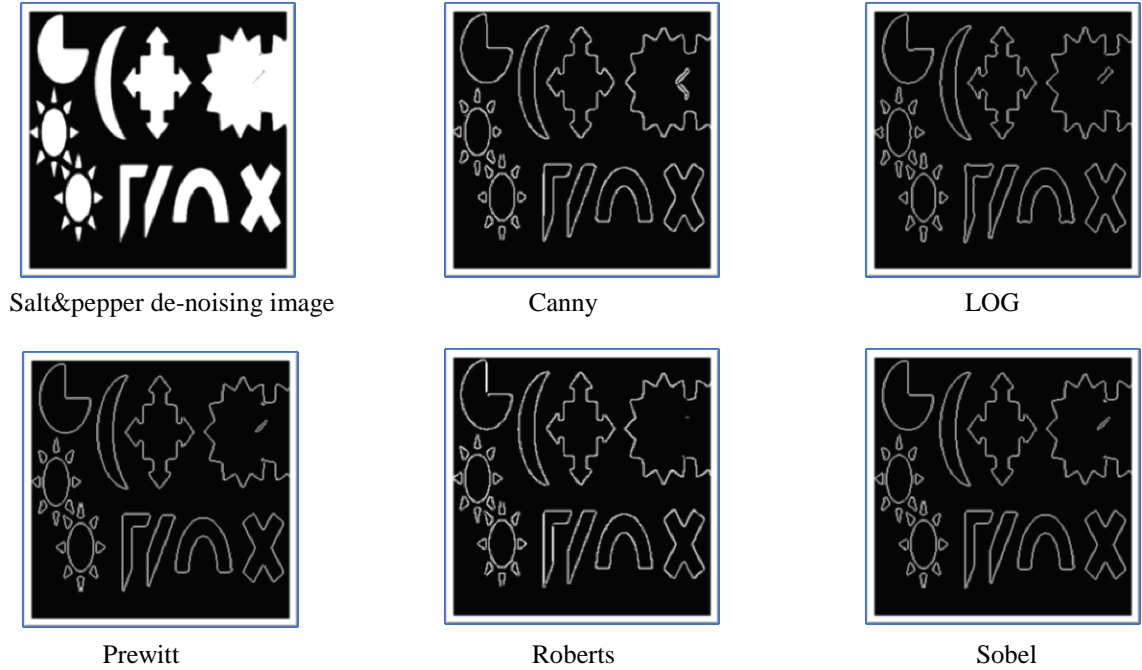
**Figure 4.62:** Threshold Values of Edge Algorithms in De-Noising Environment-Salt and Pepper Noise



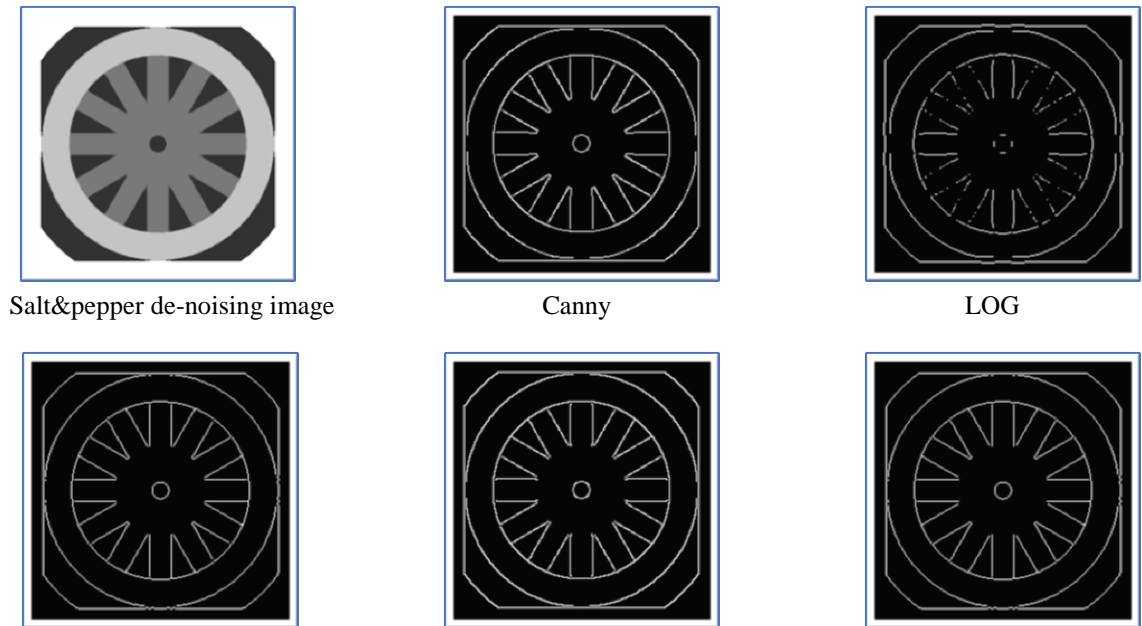
**Figure 4.63:** Correlation Values of De-Noising Environment-Salt and Pepper Noise

The figures from 4.64 to 4.69 are the edge detection images of the salt and pepper de-noised images and these images are enhanced compare with the edge images of the

noisy environment. For the edge images figures 4.12, 4.13 and 4.14, it can be seen that the algorithms Roberts, Prewitt and Sobel still give not good detection although the images was enhanced.



**Figure 4.64:** Edge Detection Images of Binary Image in De-Noising Environment-Salt and Pepper Noise

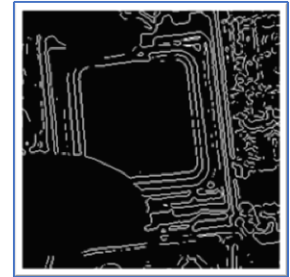
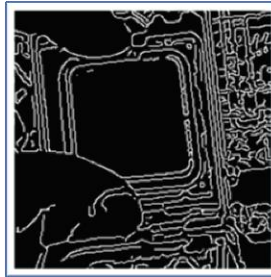


Prewitt

Roberts

Sobel

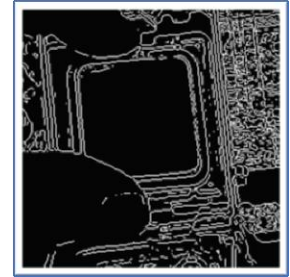
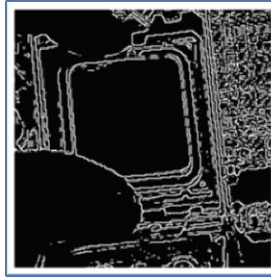
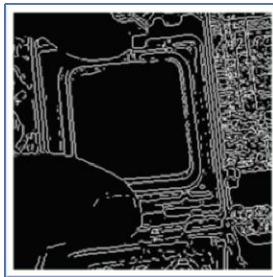
**Figure 4.65:** Edge Detection Images of Graphic Image in De-Noising Environment-Salt and Pepper Noise



Salt&pepper de-noising image

Canny

LOG

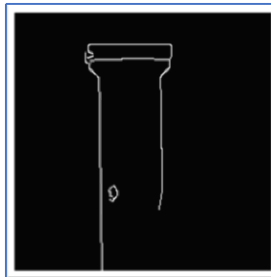
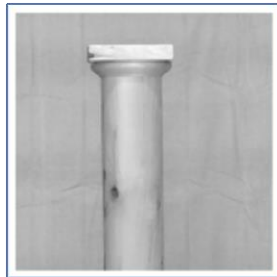


Prewitt

Roberts

Sobel

**Figure 4.66:** Edge Images of High-Freq Image in De-Noising Environment-Salt and Pepper Noise



Salt&pepper de-noising image

Canny

LOG



Prewitt

Roberts

Sobel

**Figure 4.67:** Edge Images of Low-Freq Image in De-Noising Environment-Salt and Pepper Noise



Salt&pepper de-noising image

Canny

LOG

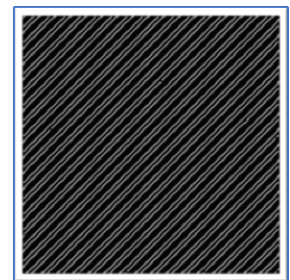
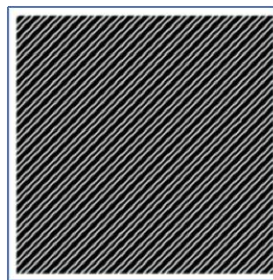
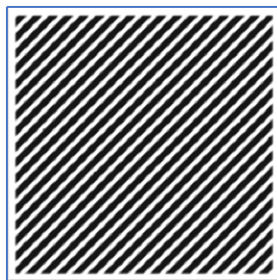


Prewitt

Roberts

Sobel

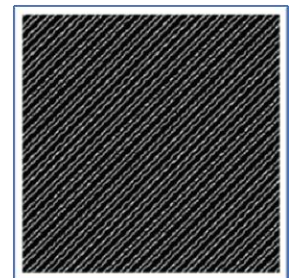
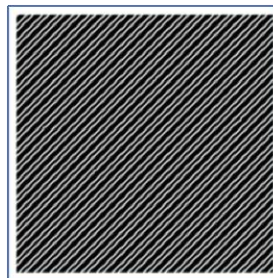
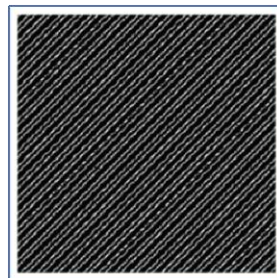
**Figure 4.68:** Edge Images of Median-Freq Image in De-Noising Environment-Salt and Pepper Noise



Salt&pepper de-noising image

Canny

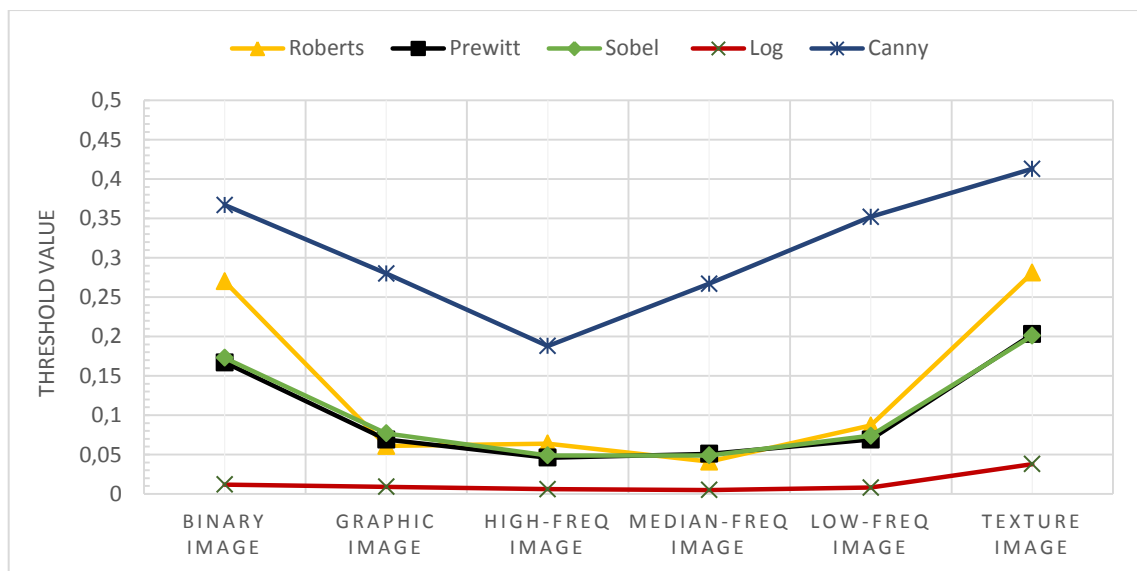
LOG



**Figure 4.69:** Edge Detection Images of Texture Image in De-Noising Environment-Salt and Pepper Noise

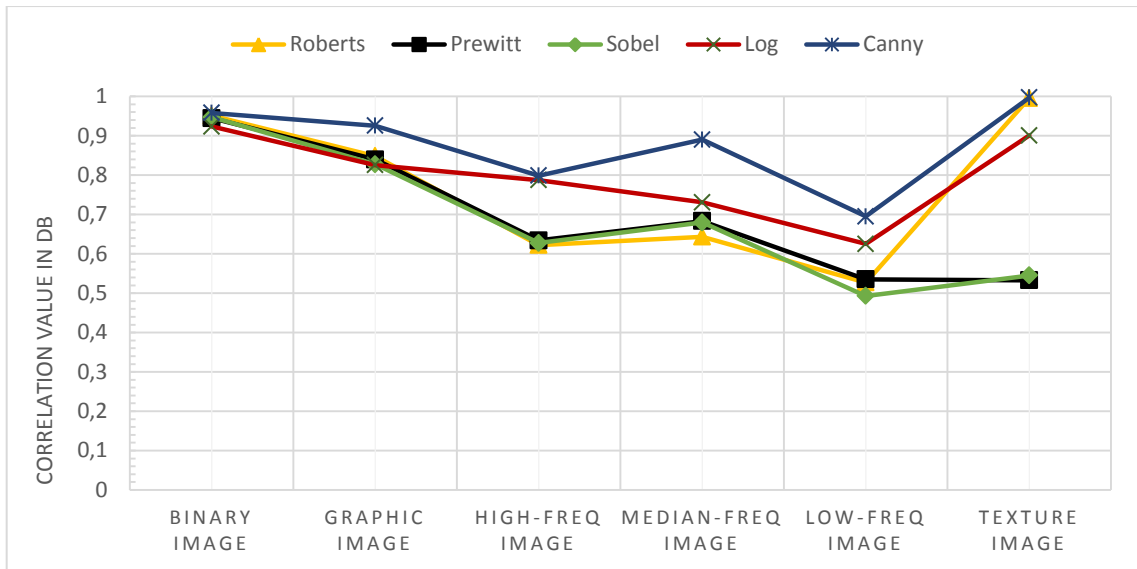
### 4.3.2.3 De-nosing environment results-speckle de-noised image

Figure 4.70 is the threshold values of the edge detection algorithms in the de-noising environment-speckle de-noised images. In this figure with all types of speckle de-noised images, it can be seen that the Canny takes the highest threshold values, LOG takes the lowest threshold values, Sobel, Prewitt take an identical threshold values and Roberts takes threshold values that is not so different from the Prewitt and Sobel threshold values.



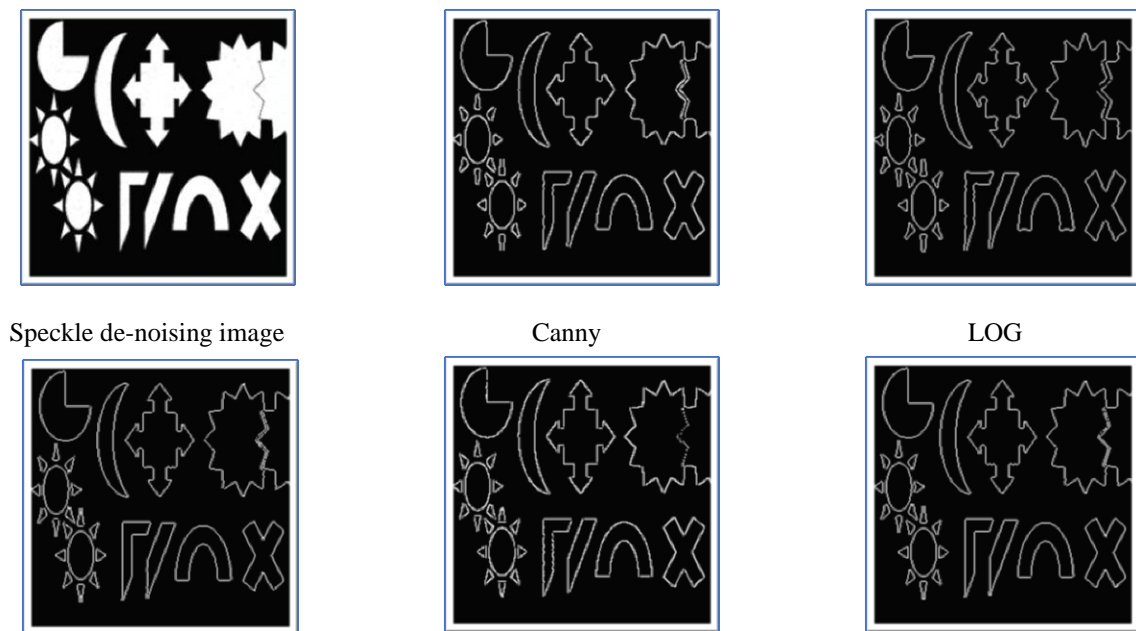
**Figure 4.70:** Threshold Values of Edge Detection Algorithms in De-Noising Environment-Speckle Noise

Figure 4.71 is the correlation values between the edge images of speckle de-noised images and the clean edge images that came from the clean environment. In this figure with all types of speckle-de-noised images the observation that can be seen that the correlation always is high with Canny algorithm and always is low with Sobel algorithm except when the input is the binary. All edge algorithms give a good detection when the input is binary image and Roberts gives good detection with the texture image.

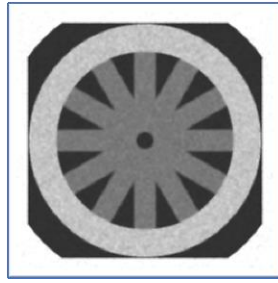


**Figure 4.71:** Correlation Values of De-Nosing Environment-Speckle Noise

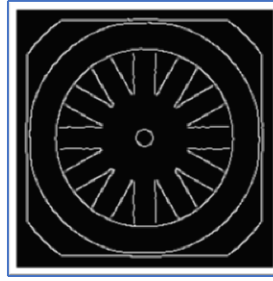
The figures from 4.72 to 4.77 are the edge detection images of the speckle de-noised images and these images are enhanced compare with the edge images of the noisy environment. For the edge images figures 4.74, 4.75 and 4.76, it can be seen that the algorithms Roberts, Prewitt and Sobel still give not good detection although the images was enhanced.



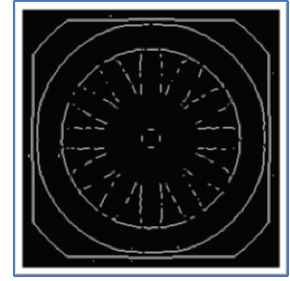
**Figure 4.72:** Edge Detection Images of Binary Image in De-Nosing Environment-Speckle Noise



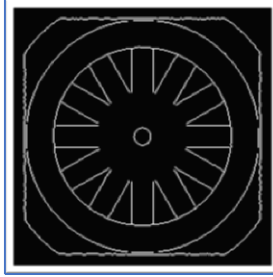
Speckle de-noising image



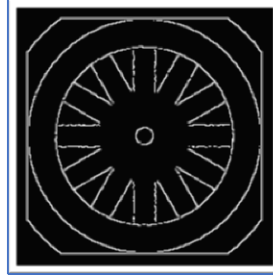
Canny



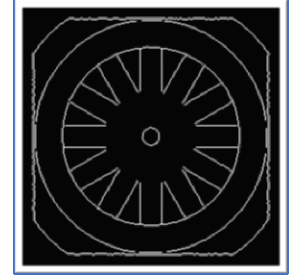
LOG



Prewitt



Roberts

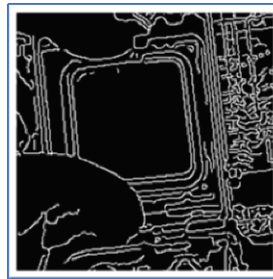


Sobel

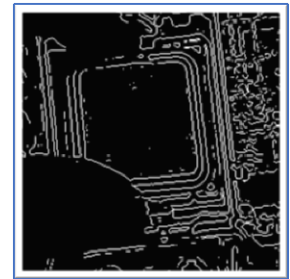
**Figure 4.73:** Edge Detection Images of Graphic Image in De-Noising Environment-Speckle Noise



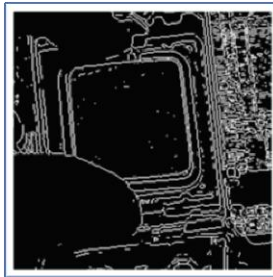
Speckle de-noising image



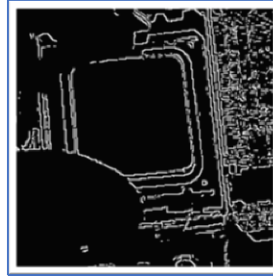
Canny



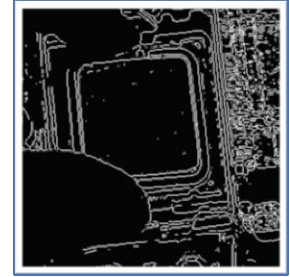
LOG



Prewitt



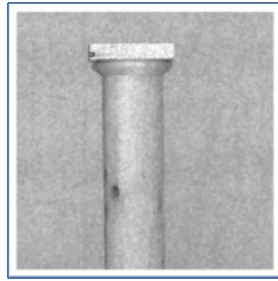
Roberts



Sobel

**Figure 4.74:** Edge Detection Images of High-Freq Image in De-Noising Environment-Speckle Noise

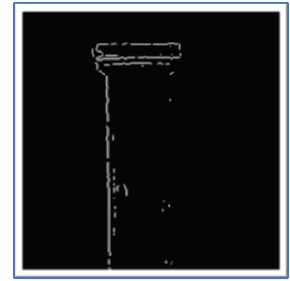




Speckle de-noising image



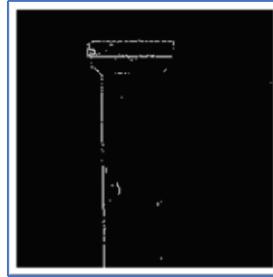
Canny



LOG



Prewitt

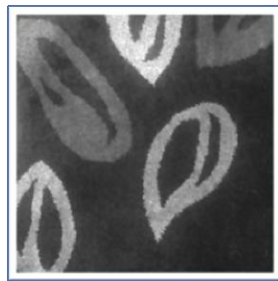


Roberts



Sobel

**Figure 4.75:** Edge Detection Images of Low-Freq Image in De-Noising Environment-Speckle Noise



Speckle de-noising image



Canny



LOG



Prewitt

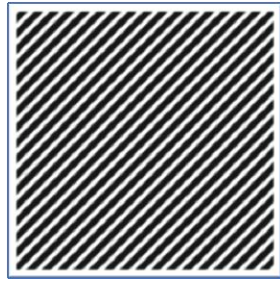


Roberts

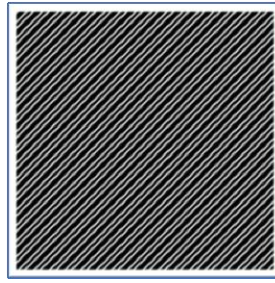


Sobel

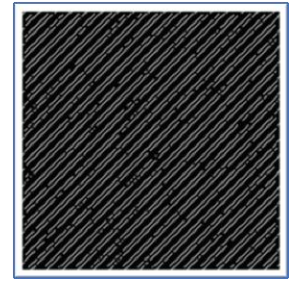
**Figure 4.76:** Edge Detection Images of Median-Freq Image in De-Noising Environment-Speckle Noise



Speckle de-noising image



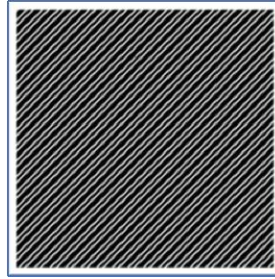
Canny



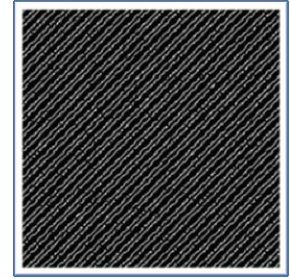
LOG



Prewitt



Roberts



Sobel

**Figure 4.77:** Edge Detection Images of Texture Image in De-Noising Environment-Speckle Noise

## CHAPTER 5

### DISCUSSIONS AND CONCLUSIONS

#### 5.1 Discussions

- In clean environment the edge images, figures (4.2, 4.3, 4.4, 4.5, 4.6, 4.7) show that all edge detection algorithms can give a good detection in all types of images if the threshold value for each edge detection algorithm is chosen in an appropriate way.
- In noise removal algorithms, the highest PSNR values are obtained with Wiener filter in all types of Gaussian-noisy images and speckle-noisy images, figures (4.33, 4.47) also the median filter with all types of salt and pepper-noisy images gives the highest PSNR values figure 4.40.
- In both noisy and de-noised environments, it is possible to infer the following:
  - The edge detection algorithms Sobel, Prewitt and Roberts give the worst detection (low correlation values) with Gaussian noise and speckle noise images as in the edge images in figures (4.11, 4.12, 4.13, 4.14, 4.27, 4.28, 4.30) and the correlation figures (4.9, 4.25), the edge detection images of these algorithms in salt-and-pepper noisy images are even worse than the previous ones, as in the edge images in figures (4.19, 4.20, 4.21, 4.22, 4.23) and the correlation figure 4.17. In de-noised images (Gaussian, salt and pepper and speckle) the correlation values of Sobel, Prewitt and Roberts algorithms increased as shown in correlation figures (4.55, 4.63, 4.71) and the edge images of these algorithms are enhanced as in figures

(4.57, 4.58, 4.59, 4.60, 4.65, 4.66, 4.67, 4.68, 4.73, 4.74, 4.75, 4.76) but still didn't reach the desired detection compared with other edge detection like LOG and Canny.

- Canny edge detection algorithm gave the best detection (the correlation is always high) in all types of noisy and de-noised images see the correlation figures (4.9, 4.17, 4.25, 4.55, 4.63, 4.71) but this algorithm is computationally more expensive compared with the other edge detection algorithms.
  - All edge detection algorithms give a good detection in all types of noise and de-noised binary images see the edge images, figures (4.10, 4.18, 4.26, 4.56, 4.64, 4.72).
  - The Roberts edge detection algorithm gives a good detection in all types of noise and de-noised texture images as in the edge images in figures (4.15, 4.23, 4.31, 4.61, 4.69, 4.77).
  - The order of these algorithms that were used in this study from best to worst detection Canny, LOG, Sobel, Prewitt, Roberts respectively.
  - All algorithms are facing difficulties with the high-frequency image compared with the other images like binary image, graphic image and low frequency image.
- In all types of images of all environments the threshold figures (4.8, 4.16, 4.24, 4.54, 4.62 and 4.70) show the following:
    - The threshold value for each edge detection algorithm isn't fixed value, this value will be changed if the image type or the image environment is changed but this change is a slight change.

- The threshold values of LOG edge detection algorithm taken values between 0.01 and 0.05 and these values represent a lower threshold values of all the practical results.
- The threshold values of Roberts, Sobel and Prewitt edge detection algorithms were between 0.05 and 0.45. Furthermore Sobel and Prewitt edge detection algorithms have closet threshold values.
- The high threshold values of the Canny edge detection algorithm were between 0.2 and 0.45.

Compression with some works in the literature:

Clear environment		
#	Work	Result
1	This work	(Roberts, Sobel, Prewitt, LOG and Canny) give complete edge detection if the suitable threshold is selected
2	[11]	Canny is the best edge detector
3	[15]	Canny is the best edge detector
4	[16]	Canny is the best edge detector

Noisy environment		
#	Work	Result
1	This work	Canny is the best edge detector
2	[13]	Canny is the best edge detector
3	[17]	Canny is the best edge detector
4	[56]	Canny is the best edge detector

De-noising environment		
#	Work	Result
1	This work	Canny is the best edge detector
2	[12]	Smoothing before detection give good results
3	[57]	Canny is the best edge detector

## 5.2 Conclusions

- In clean environments all algorithms give a good detection if the threshold value of each algorithm is chosen in appropriate way.
- Canny is the best algorithm that can be used in all environments with all types of images and the best high threshold range is from 0.2 to 0.45. If it is used in the de-nosing environment, it is preferable to use the Wiener filter, if the source noise in the image is Gaussian or speckle noise but it is preferable to use the median filter, if the source noise in the image is the salt and pepper noise, before extracting the edge by using this algorithm.
- In all environments, all edge algorithms give a good detection, if the source image is the binary image.
- In all environments, Roberts's algorithm gives a good detection, if the source image is the texture image.
- All edge detection facing difficulties with the high-freq image because this image contains huge details.

### **5.3 Future Work**

This study is considered the following types of noises: Gaussian noise, Salt and Pepper noise and Speckle noise and the following types of filters: mean filter, Alpha trimmed mean filter, median filter and Wiener filter, so it can be extended to include more types of noises like quantization noise, Poisson noise or a random noise, also it can be extended with more types of noise removal like geometric mean filter, Gaussian low pass filter or by making an estimation to the noise parameters if the noise in the noisy image is unknown noise.

## REFERENCES

- [1] **R. C. GONZALEZ AND R. E. WOODS**, Digital Image Processing. Upper Saddle River, NJ: Prentice-Hall, 2001, pp. 572-585.
- [2] **W. K. PRATT**, Digital Image Processing. New York, NY: Wiley-Interscience, 1991, pp. 491-556.
- [3] **W. FREI AND C. CHEN**, "Fast Boundary Detection: A Generalization and New Algorithm," IEEE Trans. Computers, vol. C-26, no. 10, pp. 988-998, Oct. 1977.
- [4] **J. CANNY**, "A computational approach to edge detection," IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. 8, No. 6, pp. 679-698, Nov. 1986.
- [5] **EKSTROM, M. P.** (1984) Digital Image Processing Techniques, ACADEMIC PRESS, Inc.
- [6] **TANG, S.** (2000) "Survey of Edge Detection", tang-s@cs.unr.edu.
- [7] **LINDEBERG, T.** (1996) "Edge Detection and Ridge Detection with Automatic Scale Selection", <http://www.bion.kth.se/tony>.
- [8] **VLIET, L. J.; T. YOUNG, AND G. L. BECKER'S**, (1988) " An Edge Detection model Based on Non-Linear Laplace Filtering" , Pattern Recognition and Artificial Intelligence, Edited By E. S. Gelsema and L. N. Kanal, Elsevier Science Publishers B. V. , 63-73.4 .
- [9] **ZIOU, D. AND S. TABBONE**, (1998) "Edge detection Techniques-An Overview", <http://www.ziou.com>.
- [10] **HEATH, M.; S. SARKAR; T. SANOCKI, AND K. BOWYER**, (1999) "Comparison of edge detectors: A Methodology and Initial Study", <http://figment.csee.usf.edu/>.
- [11] **MIN C. SHIN, DMITRY B. GOLDFOF, KEVIN W. BOWYER AND SAVVAS NIKIFOROU**, "Comparison of Edge Detection Algorithms Using a Structure from Motion Task" IEEE transactions on systems, man, and cybernetics- part b: cybernetics August (2001).



- [12] **RAMAN MAINI, J.S SOHAL**, 2006, "Performance evaluation of Prewitt edge detector for noisy images", GVIP journal volume 6, issue 3.
- [13] **RAMAN RAMAN MAINI AND DR. HIMANSHU AGGARWAL**, 2006, "Study and comparison of various edge detection techniques", IJIP volume 3 issue.
- [14] **N.M. N.M. NANDHITHA AND ET AL**, 2006, "A comparative study on the performance of the classical and wavelet based edge detection for image de-noising on defective weld thermographs", Asia pacific conference on NDT.
- [15] **WANG LUO**, 2006, "Comparison for edge detection of colony images", International journal of computer science and network security volume 6 number 9A.
- [16] **EHSAN NADERNEJAD, SARA SHARIFZADEH AND HAMID HASSANPOUR**, "Edge Detection Techniques Evaluations and Comparisons", Mathematical Sciences, Vol. 2, 2008, no. 31, 1507 – 1520.
- [17] **SHASHIDHAR RAM JOSHI AND ROSHAN KOJU**, "Study and Comparison of Edge Detection Algorithms", IEEE Internet (AH-ICI), 2012 Third Asian Himalayas International Conference 23-25 Nov. 2012.
- [18] **W.R. GONZALES, R.**, Digital Image Processing, Prentice-Hall (2002).
- [19] **JENSEN, J. R.** (1986) Introductory Digital Image Processing, A Remote Sensing Perspective, Prentice - Hall.
- [20] **PRATT, W. K.** (1978) Digital Image Processing, a Wiley-Interscience Publication by John Wiley & Sons, Inc.
- [21] **UMBAUGH, S. E.** (1998) Computer Vision and Image Processing, A practical Approach Using CVIP tools, Prentice Hall PTR.
- [22] **FISHER, R.; S. W. A. PERKIN, AND E. WOLF ART**, (2000) Image Processing Learning Resources, HIPR2, Explore with JAVA.
- [23] **J. E. LIM.** Two-Dimensional Signal and Image Processing. Prentice Hall, Inc., 1990.
- [24] **W. FREI AND C. CHEN**, "Fast Boundary Detection: A Generalization and New Algorithm," IEEE Trans. Computers, vol. C-26, no. 10, pp. 988-998, Oct. 1977.
- [25] **J. CANNY**, "A computational approach to edge detection," IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. 8, No. 6, pp. 679-698, Nov. 1986.

- [26] **R. A. SCHOWENGERDT**. Techniques for image processing and classifications in remote sensing. Remote Sensing, Queens College, 2004.
- [27] **D. MARR AND E. HILDRETH**. Theory of edge detection. Proc. Royal Society, London, pages 187–217, 1980.
- [28] **M. SHARIFI, M. FATHY, AND M. TAYEFEH MAHMOUD**. A classified and comparative study of edge detection algorithms. Proceedings of the International Conference on Information Technology: Coding and Computing, ITCC, 2002.
- [29] **L. G. ROBERTS**. “Machine perception of 3-D solids” ser. Optical and Electro-Optical Information Processing. MIT Press, 1965.
- [30] **ABBASI TA, ABBASI MU**, A novel FPGA-based architecture for Sobel edge detection operator, International Journal of Electronics, 13(9), 2007, 889-896.
- [31] **J. MATTHEWS**. “An introduction to edge detection: The sobel edge detector,” Available at <http://www.generation5.org/content/2002/im01.asp>, 2002.
- [32] **R. C. GONZALEZ AND R. E. Woods**. “Digital Image Processing”. 2nd ed. Prentice Hall, 2002.
- [33] **V. TORRE AND T. A. POGGIO**. “On edge detection”. IEEE Trans. Pattern Anal. Machine Intell, vol. PAMI-8, no. 2, pp. 187-163, Mar. 1986.
- [34] **E. R. DAVIES**. “Constraints on the design of template masks for edge detection”. Pattern Recognition Lett. vol. 4, pp. 111-120, Apr. 1986.
- [35] **W. FREI AND C.-C. Chen**. “Fast boundary detection: A generalization and a new algorithm”. IEEE Trans. Comput., vol. C-26, no. 10, pp. 988-998, 1977.
- [36] **R. M. HARALICK**. “Digital step edges from zero crossing of the second directional derivatives,” IEEE Trans. Pattern Anal. Machine Intell. Vol. PAMI-6, no. 1, pp. 58-68, Jan. 1984.
- [37] **J. F. CANNY**. “A computational approach to edge detection”. IEEE Trans. Pattern Anal. Machine Intell. Vol. PAMI-8, no. 6, pp. 679-697, 1986.
- [38] **F. V. DER HEYDEN**. Evaluation of edge detection algorithms. University of Twente, Netherlands, 1989.
- [39] **L. BEAUREPAIRE, K. CHEHDI, AND B. VOZEL**. Identification of the nature of the noise and estimation of its statistical parameters by analysis of local histograms, Proc. 1997 IEEE International Conference on Acoustics, Speech and Signal Processing, Vol. 4, pp. 2805-2808.

- [40] **ALOKE DATTA**, 2009, "Removal of Random Valued Impulsive Noise", Master thesis, Computer Science and Engineering National Institute of Technology Rourkela.
- [41] **RAFAEL C. GONZALES, RICHARD E. WOODS, AND STEVEN L. EDDINS**, 2003, "Digital Image Processing Using Matlab", John Wiley & Sons.
- [42] **RAFAEL C. GONZALES AND RICHARD E. WOODS**, 2002, "Digital Image Processing", by Prentice-Hall, Inc. 2nd edition, pp. 220-277.
- [43] **E. ARGYLE**. Techniques for edge detection. Proc. IEEE, 59:285–286, 1971.
- [44] **STANLEY J. REEVES**, 1990, "A Cross-Validation Approach to Image Restoration and Blur Identification", Ph.D. thesis in Electrical Engineering, Georgia Institute of Technology.
- [45] **STUART W. PER**, 2006, "Adaptive Image Restoration: Perception Based Neural Network Models and Algorithms", Ph.D. thesis, Electrical and Information Engineering, University of Sydney, AUSTRALIA.
- [46] **GONZALEZ, R. C. AND WOODS, R. E.** [2002]. Digital Image Processing, 2nd ed., Prentice Hall, Upper Saddle River, NJ.
- [47] **ALBERT BARABBAS ET AL**, "The human disease network", Plos.org.
- [48] **J. L. RODGERS AND W. A. NICE WANDER**. Thirteen ways to look at the correlation coefficient. The American Statistician, 42(1):59–66, February 1988.
- [49] **STIGLER, STEPHEN M.** (1989). "Francis Galton's Account of the Invention of Correlation". Statistical Science 4 (2): 73–79. doi:10.1214/ss/1177012580. JSTOR 2245329.
- [50] **HUYNH-THU, Q.; GHANBARI, M.** (2008). "Scope of validity of PSNR in image/video quality assessment". Electronics Letters 44 (13).
- [51] **WELSTEAD, STEPHEN T.** (1999). Fractal and wavelet image compression techniques. SPIE Publication. pp. 155–156 ISBN 978-0-8194-3503-3.
- [52] **BARNI, MAURO, ED.** (May 2006). "Fractal Image Compression". Document and image compression (CRC Press) 968: 168–169. ISBN 9780849335563. Retrieved 5 April 2011.
- [53] **THOMOS, N., BOULGOURIS, N. V., & STRINTZIS, M. G.** (2006, January). Optimized Transmission of JPEG2000 Streams Over Wireless Channels. IEEE Transactions on Image Processing.

- [54] **XIANGJUN, L., & JIANFEI, C.** Robust transmission of JPEG2000 encoded images over packet loss channels. ICME 2007 (pp. 947-950). School of Computer Engineering, Nanyang Technological University.
- [55] MATLAB help website, <http://www.mathworks.com/help/matlab/>.
- [56] **LI BIN AND MEHDI SAMIEI YEGANEH**, “ Comparison for Image Edge Detection Algorithms”. IOSR Journal of Computer Engineering, 2278-0661 Volume 2, Issue 6 (July-Aug. 2012).
- [57] **SHRIVAKSHAN AND CHANDRASEKAR**, “A Comparison of various Edge Detection Techniques used in Image Processing”, IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 5, No 1, September 2012.
- [58] **A. K. SAO AND B. YEGNANARAYANA**. Face recognition using correlation filters and auto aussociative neural networks. Proc. International conference on Intelligent Sensors and Information Processing, 2004.

## CURRICULUM VITAE

Surname, Name: Mahmood, Alaa  
Nationality: Iraqi  
Date and Place of Birth: 14 /11/1984, Mosul  
Marital Status: Married  
Phone: +9647702944867  
Email: [amm.eng84@yahoo.com](mailto:amm.eng84@yahoo.com)

### EDUCATION

Degree	Institution	Year of Graduation
BS	Mosul university	2007
High School	Alresla	2003

### WORK EXPERIENCE

Year	Place	Enrollment
2009-Present	Al-Hadbaa University College / Tech. Comp. Eng. Dept.	Lab Instructor
2007	Al mehrab Company	communication

### FOREIGN LANGUAGES

English, Arabic.