



# **An Automatic Method for Edge Detection Evaluation Based on Semi-Optimal Edge Detector**

**طريقة مؤتمتة لتقييم اكتشاف الحواف بالاعتماد على مكتشف  
الحواف شبه المثالي**

**Student Name:**

Ammar N. A. Alnajjar

**Supervisor:**

Dr. Mohammed Otair

A Thesis Submitted in Partial Fulfillment of the Requirements for Degree  
of Master in Computer Science

Amman Arab University  
College of Computer Sciences and Informatics  
2014

## AUTHORIZATION

I, the under signed "Ammar N. A. Alnajjar" authorize hereby Amman Arab University to provide copies of this thesis to libraries, institutions and any other parties upon their request.

Name: Ammar N. A. Alnajjar

Signature:



Date: 11 / 6 / 2014

## RESOLUTION OF THE EXAMINING COMMITTEE

This thesis titled "An Automatic Edge Detection Evaluation Based on Semi-Optimal Edge Detector". Has been defended and approved on \_\_  
11 / 6 / 2014.

<u>Examining Committee</u>	<u>Title</u>	<u>Signature</u>
Prof. Dr. Ala'a Al-hamami	Chair	
Dr. Mohammed Otair	Member and Supervisor	
Dr. Osama Musleh	Member	

### **Declaration**

I am Ammar Alnajjar; hereby declare that this work entitled "An Automatic Method for Edge Detection Evaluation Based on Semi-Optimal Edge Detector" was independently carried out by me under the guidance and supervision of Dr. Mohammed Otair, Jordan.

This work is submitted to Amman Arab University in partial fulfillment of the requirement for the award of the Master's Degree in computer science during the academic year 2014.

Place: Jordan

Date: 11 / 6 / 2014

**Ammar Alnajjar**

## الإهداء

يا من أحمل اسمك بكل فخر  
يا من أفتقدك منذ الصغر  
يا من يرتعش قلبي لذكرك  
يا من أودعتني لله أهديك هذا البحث

والذي العزيز

إلى من أرضعتني الحب والحنان  
إلى رمز الحب وبلسم الشفاء  
إلى القلب الناصع بالبياض

والدتي الحبيبة

إلى القلوب الطاهرة الرقيقة والنفوس البريئة إلى رياحين حياتي

زوجتي

ابنتي

إخوتي

إلى من زرعوا التفاؤل في درينا وقدموا لنا المساعدات والتسهيلات والأفكار  
والمعلومات، ربما دون أن يشعروا بدورهم

فلهم منا كل الشكر

الدكتور محمد عطيير

المهندس أنس سمير

إلى كل من ساهم في إنجاز هذا العمل  
فلهم منا كل الشكر

## ACKNOWLEDGEMENTS

I have taken efforts in this thesis. However, it would not have been possible without the support and help of many individuals. I would like to extend my sincere thanks to all of them. I am highly indebted to Dr. Mohammed Otair, Eng. Anas Sameer and Dr. Firas A. Jassim, for their guidance and constant supervision as well as for providing necessary information regarding the thesis and also for their support in completing the thesis. I would like to express my gratitude towards my parents and member of Orphans Fund Development Corporation and especially to Dr. Faysal Alhyari, for their kind co-operation and encouragement which help me in completion of this paper. My thanks and appreciations also go to my colleague in developing the thesis and people who have willingly helped me out with their abilities.

# Table of Contents

الإهداء .....	V
<b>ACKNOWLEDGEMENTS</b> .....	<b>VII</b>
<b>Table of Contents</b> .....	<b>VIII</b>
<b>ABBREVIATIONS</b> .....	<b>X</b>
<b>LIST OF FIGURES</b> .....	<b>XI</b>
<b>Abstract</b> .....	<b>XV</b>
المُلخَص .....	<b>XVI</b>
<b>CHAPTER ONE:(INTRODUCTION)</b> .....	<b>1</b>
1.1 Introduction .....	2
1.2 Edge Detection .....	3
1.3 Need for edge detection.....	5
1.4 Criteria for optimal edge detection process.....	5
1.5 Statement of the problem .....	6
1.6 Objectives of this work.....	7
1.7 Methodology.....	7
1.8 Thesis contribution .....	8
1.9 Thesis organization .....	8
<b>CHAPTER TWO:(BACKGROUND &amp; LITERATURE REVIEWS)</b> .....	<b>10</b>
2.1 Background .....	11
2.2 Literature Review .....	14
2.3 Conclusion of Literature Reviews .....	23
<b>CHAPTER THREE:(MEASURING THE EDGE DETECTION METHODS)</b> .....	<b>24</b>
3.1 The Steps of the main Edge Detection .....	25
3.2 Methods of edge detection techniques.....	27
3.3 Filtering .....	37
3.4 Pratt Measure .....	38
<b>CHAPTER FOUR:A-EDV (THE PROPOSED ALGORITHM)</b> .....	<b>41</b>
4.1 An Automatic Edge Detection Evaluation .....	42
4.2 A-EDV How Does It Work? .....	44
<b>CHAPTER FIVE:(THE EXPERIMENTS &amp; RESULTS)</b> .....	<b>51</b>
5.1 Experiments & Results .....	52
6.1 RECOMMENDATIONS.....	57



6.2 CONCLUSION.....	57
<b>REFERENCES .....</b>	<b>59</b>
<b>APPENDIX.....</b>	<b>63</b>

## ABBREVIATIONS

Abbreviations	Meaning
A-EDV	An Automatic Edge Detection Evaluation
ED	Edge Detection
GUI	Graphical User Interface
LOG	Laplacian of Gaussian
MF	Median Filter
SD	Standard Deviation

## LIST OF FIGURES

Figure No.	Figure Name	Page
1	Sobel operator mask	
2	Robert's cross operator masks	
3	Prewitt gradient edge detector masks	
4	Three commonly used discrete approximations to the Laplacian filter	
5	The 2-D LOG function, the x and y axes are marked in standard deviations ( $\sigma$ )	
6	LOG function has closing discrete with Gaussian $\sigma= 1.4$	
7	Pair of 3X3 convolution masks	
8	Pixels of a 5X5 image	
9	A half circle with 5 regions	
10	Lena.png	
11	The final GUI Results	
12	Allah.jpg	
13	Flamingo.bmp	
14	Lena.png	
15	Edge Detection Methods Ratio Chart	
16	Allah.jpg	
17	Mohammed.bmp	
18	Panda.bmp	
19	Bird.jpg	
20	Bird1.bmp	

21	Golden.jpg	
22	AAU.bmp	
23	Rice.bmp	
24	Lena.png	
25	Tala.jpg	
26	Poly.bmp	
27	Tools.jpg	
28	Screws.jpg	
29	Kitchen.jpg	
30	Pout.tif	
31	Zag.jpg	
33	Shadow.jpg	
34	Flamingo.jpg	

## LIST OF BLOCK DIAGRAMS

Table Number	Table Name	Page
1	Level Processing of an Image	
2	A-EDV	

## LIST OF TABLES

Table Number	Table Name	Page
1	The resulted ratio values	

## Abstract

The methods of edge detection play an important role in many image processing applications as edge detection is regarded as an important stage in image processing and the extraction of certain information from it. The evaluation method of various edge detection techniques is the perfect way to choose the best technique within these multiple methods for certain image.

This study determined a new process based on the most dynamic techniques for new automatic edge detection evaluation based on semi-optimal edge detector. The main advantages of the proposed method are the evaluation of edge detection methods to know which technique is the best of edge detection.

The study showed that the results of evaluation for each experimental automated technique of edge detection through applying an algorithm on several preferable edge detection techniques, such as Sobel, Roberts, Prewitt, Canny and Laplacian of Gaussian (LOG) in according to reach the objects that must be clear to human vision.

Moreover, the results showed that applying standard deviation with median filter ensure smooth image and help getting rid of the noisy pixel to perform ideal images. In addition to applying Pratt measure, each method of edge detection was used separately to get the final results of the evaluation algorithm in terms of an automatic method for edge detection evaluation based on semi-optimal edge detector to reach the most appropriate methods through using the best ratio of the value one.

## الملخص

تؤدي طرائق اكتشاف الحواف دوراً مهماً في العديد من تطبيقات المعالجة التصويرية، إذ يعد ضمن مرحلة مهمة في عملية تحليل الصور واستخلاص معلومات معينة منها، كما أن أساليب تقييم تقنيات اكتشاف الحواف المختلفة تعد من أفضل الأساليب التي تساعد في اختيار أفضل تقنية ممكن استخدامها بعد تقييم الأساليب المتعددة.

تهدف هذه الدراسة إلى تحديد عملية جديدة تقوم على واحدة من التقنيات الحديثة الأكثر ديناميكية لتقييم اكتشاف الحواف بشكل تلقائي على أساس مكتشف الحواف شبه المثالي. وتتمثل المزايا الرئيسية من الأساليب المقترحة في تقييم أساليب اكتشاف الحواف لمعرفة تقنية الأنسب لاكتشاف الحواف.

أظهرت الدراسة أن النتائج المتعلقة بالتقييم التجريبي المؤتمت لكل تقنية من خلال استخدام نتائج الخوارزمية المطبقة على الطرق المتعددة للكشف عن أساليب اكتشاف الحواف، مثل تقنية سوبل، روبرتس، بريويت، كاني و لابلاسيان للحصول على المجسمات التي يمكن تحليلها لتكوين رؤية واضحة.

النتائج أن تطبيق الانحراف المعياري مع فلتر من نوع ميديان يضمن سلاسة الصورة ويساعد في التخلص من البكسل المؤثر لإظهار الصور المثالية. بالإضافة إلى أن تطبيق مقياس برات لكل أسلوب من أساليب لاكتشاف الحواف واستخدامها بشكل منفصل للحصول على النتائج النهائية لتقييم الخوارزمية للأسلوب المؤتمت لاكتشاف الحواف بالاعتماد على مكتشف الحواف شبه المثالي لاستخراج أنسب طريقة عن طريق استخدام أفضل نسبة هي أقرب للعدد واحد.



## CHAPTER ONE:(INTRODUCTION)

## 1.1 Introduction

Edge detection refers to the process of locating and sharp discontinuities in an image. Interruptions and sudden changes in pixel density represent the boundaries of objects in the scene. Edge points are those points in an image which have a different intensity from the other neighboring points. The points at which intensity changes forms the edge points. There are many of edge detection techniques have been used in example of discrete gradients, Laplacian of Gaussian. Gradient methods work by finding maximum and minimum in the first derivative of the image. Laplacian method searches for zero crossings in second derivative of the image to find edges.

Operators can be optimized to search for the edges of the horizontal and vertical, or diagonal. Edge detection is difficult in noisy images, because both the noise and the edges contain a high frequency. So, it could attempts to reduce the noise result in blurred and distorted edges by using appropriate filter.

Operators used in noisy images are usually larger in scale, so they can average enough data to discount localized noisy pixels. This result is less accurate in localization of the detected edges. Not all edges involve a step change in the intensity. Effects such as refraction or poor concentration can lead to objects with the limits specified by the gradual change in density [4]. The operator needs to be chosen to be in response to such a gradual change in those cases.

Therefore, edge detection process was the focus of many studies performed by many authors. Many new techniques of edge detection which search into the discontinuity in color intensity of the image leading to the features of the image components were suggested. In despite of many methods of edge detection which proved their efficiency in particular fields and concludes good results on application. The performance of methods has different application from each others, so there was a reason to evaluate of the performance of each method to show the efficiency. This thesis evaluates the performance of edge detection on a certain image by choosing five methods known as (Canny, Laplacian of Gaussian, Prewitt, Sobel, Roberts) and each application method with grayscale to detect the performance of each of them.

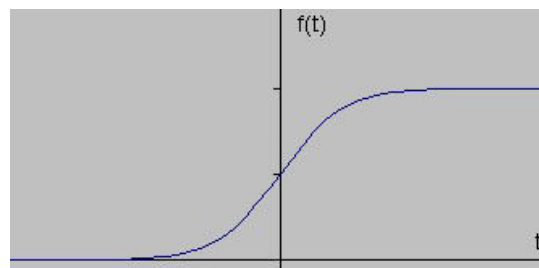
An Automatic evaluation that compares the performance of these five methods using Pratt Figure of Merit, computing the increment percentage in the detected edges, the decrement percentage in the edge points with the correct location of the edge in every method based on semi-optimal edge detector [8].

## **1.2 Edge Detection**

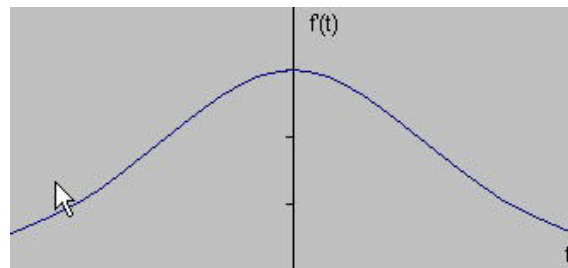
The methods of edge detection depend on the majority of based edge detection and it can be grouped into two different methods. The first one is the gradient based edge detection method that detects the edges of the search for the maximum and minimum in the first derivative of the image.

The second one is the Laplacian based edge detection, which is to find edges, must look at the way of the Laplacian zero crossings in the second derivative of the image. Feature has a one-dimensional form of the slope, and can calculate the derivative of the image to highlight its location.

According to the following signal, with the advantage shown by the jump in intensity ( $t$ ):

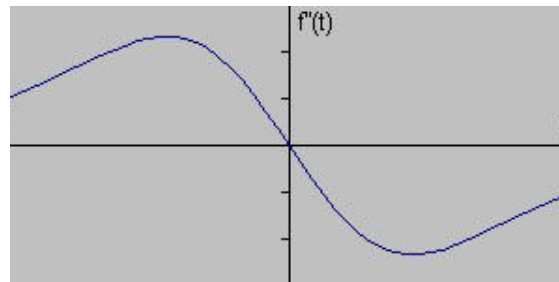


Then, taking the gradient of this signal (which, in one dimension, is just the first derivative with respect to  $t$ ) will get the following:



Distinctly, the derivative shows a maximum located in the center of the edge in the original signal. This method of locating the edge is the characteristic of the family "candidate gradient" of edge detection filters, said method comprising Sobel. Sobel announced pixel location somewhere edge if the value of the gradient exceeds some threshold. It will be the top edges of the pixel density values than those surrounding it. So once you put the threshold, you can compare the value of the gradient of the value of the threshold and edge detection whenever the threshold is exceeded [6].

Moreover, when the first derivative is a maximum and the second derivative is zero. As a result, another alternative to find a site has the advantage is to locate the zeros in the second derivative. This method is known as the Laplacian and shows the second derivative of the signal:



### 1.3 Need for edge detection

The applications of edge detection are in the enhancement of noisy images, satellite images, x-rays, medical images like cat scans, text detection, mapping of roads, video surveillance, etc.

According to the measure for edge detection evaluation, some of the operators should be examined to get the optimal edge detection in considering that of extracting important features which can be extracted from the edges of an image (e.g., corners, lines, curves) and these features are used by higher-level computer vision algorithms (e.g., recognition) [31].

### 1.4 Criteria for optimal edge detection process

The edge detection results must be with optimal criteria in order to optimize edge detection techniques, and judge the effectiveness of the edge detection process and the percentage of success reached. Criteria must be taken into account in any edge detection process so that the availability of these criteria means optimal edge detection through following:

- 1- Good detection that must minimize the probability of false positives in detecting spurious edges in addition, minimize the probability of false negatives in detecting missing real edges.
- 2- Good localization that detected edges must be as close as possible to the true edges.
- 3- Single response that minimizes the number of local maxima around true edge.

### **1.5 Statement of the problem**

As in literature review shows that there are significant efforts that have been made in the study and analysis of methods for detecting edges in order to reach the best edge detection method for a certain image according to other techniques.

The research shows that there is a constant need to develop or establish a method of edge detection evaluation to be highly efficient in detecting the edges, so that at the end of any process to detect edges must be met the needed criteria to pick out good results extracting objects that is needed.

This thesis will be presents the results an algorithm of five known methods of detecting edges (Prewitt, LOG, Sobel, Roberts, Canny) for a certain chosen image and considered as real edge points and evaluate between all techniques based on the semi-optimal edge detector [8] that it will be considered as ideal edge points. At the end of the algorithm.

the equation of Pratt measure compares the result of an edge detection method with semi-optimal edge detector (ideal with real edge points) to returns a number between 0 and 1 based upon the quality of the edge detection, with 1 being the best.

The ratio and results of the algorithm for each edge detection method will be shown as an GUI and will give the best ratio of edge detection method with high speed and in an automatic way for the results that are extracted.

### **1.6 Objectives of this work**

The objective of the thesis is to develop or establish a novel method algorithm of evaluation for various images used with many edge detection techniques to evaluates the quality of the modalities of the determinants of the edges of five methods techniques which are widely used (Canny, LOG, Prewitt, Sobel, Roberts) to get knowledge of the quality of each method towards An Automatic Method for Edge Detection Evaluation Based on Semi-Optimal Edge Detector that will be in extracting the best technique can be used for edge detection method according to each other methods, in an automatic algorithm evaluation form and with GUI results.

### **1.7 Methodology**

The methodology that will be done in this thesis will be as following:

- New algorithm will be proposed in order to extract the best edge detection method from all of five edge detection techniques through an automatic evaluation algorithm.
- The experiments will be achieved more than one of images in order to be sure that the results are accurate and precise.

- The result of the experiments will be listed and analyzed that the chosen ratio and the name of edge detection method is the best one that can be used for edge detection.

## **1.8 Thesis contribution**

The main contribution of the thesis that was published in International Journal of Computer Information Systems, Vol. 8, No.3, March 2014, [1]. This thesis aims to produce a novel automatic evaluation algorithm to show the main disadvantages and advantages of using edge detection technique with a certain image chosen according to each method of edge detection technique. The produced algorithm solution enhanced the choice of edge detection technique quickly according to five known edge detection techniques with a certain image by selecting of the best method with best ratio that is close to the number of one through an automatic evaluation using of the Pratt measure, thus this method is more effective and easy to get results in an excellent GUI.

## **1.9 Thesis organization**

The thesis consists of six chapters; each one handles a certain topic as follows:

Chapter 1 gives an introduction about the study, it transacts the basic concepts in the study which are: edge detection, need for edge detection, criteria for optimal edge detection technique, statement of the problem, objective of the this work, in addition to, methodology and thesis contribution.

Chapter 2 presents the background and literature review.



Chapter 3 describes the steps of the main edge detection, accompanied methods of edge detection techniques (Sobel, Prewitt, Roberts, Canny and LOG), filtering and Pratt measure.

Chapter 4 describes the proposed algorithm of A-EDV and how does it work.

Chapter 5 represents the experiments and results of the algorithm used in this study.

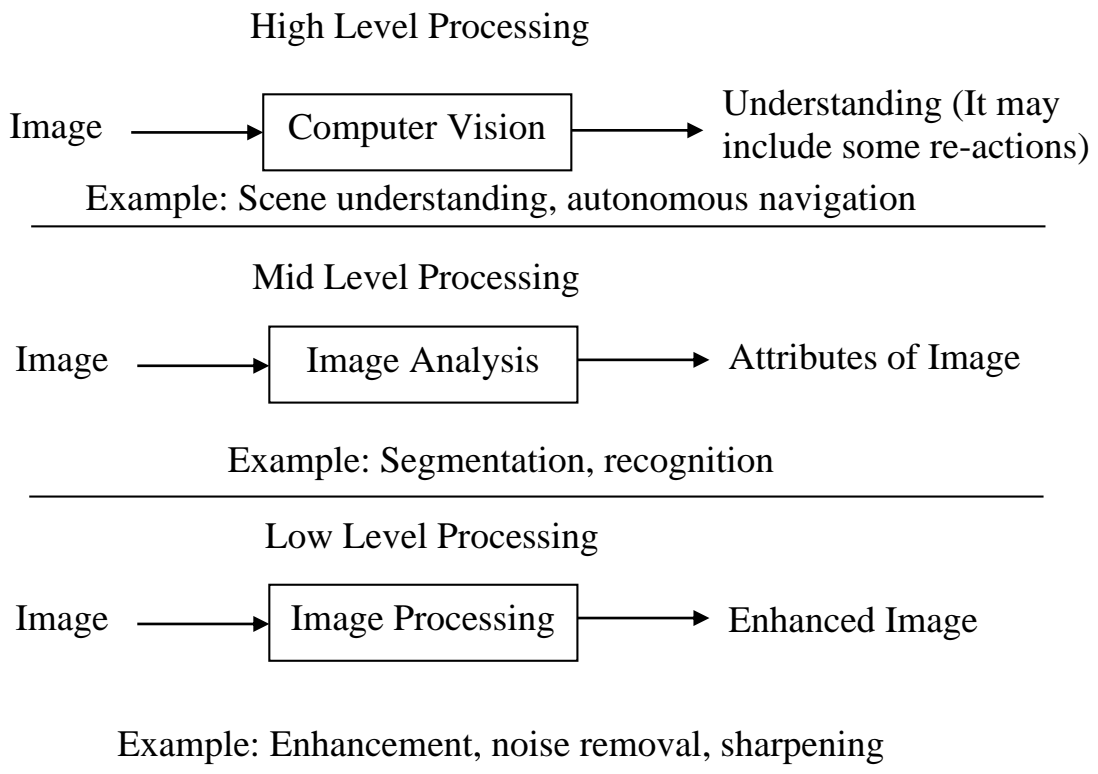
Chapter 6 describes the recommendations and the conclusion of this study.

Finally, the appendix will follow to view the algorithm related to the study, and a set of images that have been applied by the experiment in form of GUI.

## **CHAPTER TWO:(BACKGROUND & LITERATURE REVIEWS)**

## 2.1 Background

Edge detection considers as a critical area in image processing, because it has a huge role in higher level in processing as shown in the Block Diagram 1. Edge detection algorithm helps in detecting the location and existence of the edges, which formalized in images through sharp the discontinuities like changes in intensity (brightness) and color of image. Edges are detected based on the relationship with its neighbor pixel by pixel. It is called an edge if a pixel changes rapidly with its surrounding gray level values according to neighboring pixel.



Block Diagram 1, Level Processing of an image

The accurate edge line without changing can in the structural properties of the image that is the main objective in edge detecting process. So far, there is no global edge detection method that works well for all situations. So there are a large numbers of edge detection operators are obtainable, every contagious to be sensible to definite type of edges and thus upon distinct situations the impression of various algorithms differs.

A lot of edge detection algorithms such as Sobel, Prewitt and Roberts are called first order derivative based methods and applied for detecting edges which rely on gradient value. In these methods they are very sensitive to noise and fail some edges out of edge detection procedure [8].

Edges can be also detected by using Laplacian of Gaussian methods in second order derivative methods like Laplacian. LOG needs for large calculation for a large edge detector mask. Probabilities of false and failing edges still remain; cause of it is very sensitive to noise. So the noise, low contrast and another factor that edge detection methods cannot give pleasant results.

There are an important edge detector called Canny algorithm that in most of images gives in mostly times good edge detection according to another methods of edge detection. This method can keep better balancing between edge detection and noise because it is not easily disturbed by latter. So, As compared to other edge detectors [30], it can detect the true weak edges. In order to achieve the results of the amount of reliable algorithms in the areas of computer vision and image processing [18, 21], there is a greater concern in recent years with making experimental performance evaluation.

Numerous studies showed the problem of determining the performance of edge detectors in digital images [13, 20]. The majority of this work is based on the use of real or synthetic images. Evaluation can depend on comparing edges according to ground truth by using synthetic images because they give accurate and easy specification of the ground truth edge locations. However, synthetic images usually used implicate only simple geometric patterns. The core of the complexity of a real image is that it usually implicates edges of many different types, scales, and curvatures. Thus, synthetic images alone are often far too simple to give confidence in the value of the results.

A measure is taken from semi-optimal edge detector to specify ideal edge points that combines together simple standard deviation, the median filter [8], and in another measure for image processing select five known methods of edge detection known as (Canny, LOG, Prewitt, Sobel and Roberts) to specify real edge points. An automatic algorithm for each edge detection method to evaluate the methods by Pratt Figure of Merit [13] based on semi-optimal edge detector, computing the increment percentage in the detected edges, the decrement percentage in the edge points and the correct location of the edge in every method to get best proportion of the best technique can be used in the result of the best ratio printed out with form of GUI.

Automatic edge detection evaluation algorithm which is simple and very effective will become one of the important evaluation algorithm methods to get best edge detection technique to use in the image which is chosen in order to enhance the time of the judgment of choosing best method of edge detection.

Pratt proposed a render measure called the "figure of merit" [22]. This measure is between the numbers of 0 to 1, which on the number of 1 gives a perfect result in the evaluation and if approaching to the number of 0 that it increases of lost edges or lost locations.

## **2.2 Literature Review**

**Firas A. Jassim. [8]**, He introduced a novel method which adds both median filter and simple standard deviation to perform a best edge detector for image processing. De-noising process must be done on the grey scale image using median filter to correspond pixels which are likely to be become corrupted by noise.

The improvement of this stage is to smooth the image and eliminate the noisy pixels. At the second, it could be computed for each 2x2 window size by the simple statistical standard deviation. If the 2x2 window size of the value of the standard deviation is greater than a predefined threshold, then the upper left pixel in the 2x2 window performs an edge.

**Mohamed Elhabiby [5]**. In this paper, the relatively new multi-resolution technique, curve let transform, is assessed and introduced to overcome the wavelet transform limitation in directionality and scaling.

In this research paper, the assessment of second generation curve let transforms as an edge detection tool will be introduced and compared to traditional edge detectors such as wavelet transform and Canny Edge detector. Second generation curve let transform provides optimally sparse representations of objects, which display smoothness except for discontinuity along the curve with bounded curvature.

Preliminary results show the power of curve let transform over the wavelet transform through the detection of non-vertical oriented edges, with detailed detection of curves and circular boundaries, such as non-straight roads and shores.

The proposed algorithm was applied with curve let transform on high-resolution satellite imagery data, and repeated using wavelet and canny operators, the results were promising. The total number of coefficients used to reconstruct the edge map using curve let transforms was 180456, representing 6.9% of the total coefficients, while in case of wavelet, the coefficients' number was 360000, 100% of the total coefficients. Curve let transforms give close or even improved delineation to edges compared to Canny and a far better than the wavelet transforms.

This method can be used as an important layer for a further classification step as it gives a closed boundary for almost all features in the input image. Although Curve let transforms is promising and efficient for edge detection, there is one drawback must be regarded in the future, which one is that the quality of the edge detection is a function of the pre-processing steps (the high-pass filter to enhance edges), as any edge detector will suffer from the great deal of heterogeneity of the images, especially when using very high resolution imagery.

**G.T. Shrivakshan and Dr. C. Chandrasekar [11].** They presented in their paper that the significant problem is to know the primary ideas of different filters and apply these filters in defining a shark fish type which is taken as a case study. The techniques of edge detection are taken for consideration. The using of MATLAB is the software which is implemented. Gradient and Laplacian operators are the main two operators in image processing. The case study treats with notification of Shark Fish Classification using the various filters through Image Processing which depends on gradient. In their study the advantages and disadvantages of these filters are deeply studied.

**Shammi Sharma [29].** They presented an algorithm to detect and accurately locate ellipse objects in digital images. The algorithm consists of two steps. In the first step, the edge pixels are extracted using canny edge detection algorithm and then a noise removal process is run to remove the non-ellipse edge points. In the second step, Freeman Chain code is applied to detect ellipse from the digital images. At each pixel they determined the position of next pixel and so on the outline of the whole object can be obtained. Hence, given a binary image, the boundary of the shape can be determined efficiently. They implemented this algorithm to detect shapes in digital images. Chain code is the best method for boundary detection.



**Mohamed A. El-Sayed [6].** Edge detection is important field in image processing. Edges characterize object boundaries and are therefore useful for segmentation, registration, feature extraction, and identification of objects in a scene. In this paper proposed method can reduce the CPU time required for the edge detection process and the quality of the edge detector of the output images is robust. In order to validate the results in this paper, the run time of the proposed method are presented and observed that the proposed edge detector works effectively for different gray scale digital images.

When performance evaluation Experimental results demonstrate that the proposed method achieves better result than the relevant classic method. In this paper, an attempt is made to develop a new technique for edge detection. Experiment results have demonstrated that the proposed scheme for edge detection works satisfactorily for different gray level digital images. The proposed approach solves the problem of run time algorithm and the quality of the edge detector of the output images.

**Wenshuo Gao [9].** This paper proposes a method which combines Sobel edge detection operator and soft-threshold wavelet de-noising to do edge detection on images which include White Gaussian noises. In this paper, they firstly use soft-threshold wavelet to remove noise, and then use Sobel to edge detection on the image. This method is mainly used on the images which includes White Gaussian noises.

The core idea in this paper is summarized in the following points: do wavelet decomposition to the image matrix and get the wavelet coefficients with noises, process the wavelet coefficients HL, LH and HH obtained by the decomposition and keep the low frequency coefficients unchanging, Select an appropriate threshold to remove Gaussian white noise signals, do inverse wavelet transformation to the image matrix and get the image matrix after de-noising, after given Sobel edge detection operator template, convolute on every pixel of the image using this template, get the gradient of this point, and the gradient amplitude is the output of this point. At the end, get the edge detection image.

**N. Senthilkumaran [28].** This paper mainly focuses on the study of soft computing approach to edge detection for image segmentation. When process of partitioning a digital image into multiple regions or sets of pixels then called image segmentation.

Edge is a boundary between two homogeneous regions. Edge detection refers to the process of identifying and locating sharp discontinuities in an image. In this paper, the main aim is to survey the theory of edge detection for image segmentation using soft computing approach based on the Fuzzy logic, Genetic Algorithm and Neural Network.

The soft computing approaches namely, fuzzy based approach, Genetic algorithm based approach and Neural network based approach is applied on a real life example image of nature scene and the results show the efficiency of image segmentation.

**D. P. Argialas [2].** Photo interpretation of geologic lineaments is a subjective process. Therefore there is a need for automation of lineament mapping using optimal edge detection techniques. This paper includes the application of Sobel and Prewitt operators or directional detectors followed by edge linking techniques (e.g. the HOUGH Transform). The objective of this work was the implementation, evaluation and comparison of selected optimal edge detectors and the HOUGH transform algorithm towards automated geologic feature mapping in a volcanic geotectonic environment.

The resulted edge maps were quantitatively assessed with the use of evaluation metrics. Finally, the performance and behavior of each algorithm for geologic feature extraction on the specific geotectonic terrain was investigated.

One main aspect from applying the edge detection algorithms is the capability of extracting segments that really follow the terrain topography. This leads to the extraction of the exact shape of the geomorphologic features, as the caldera in this case. Taking also into consideration that this algorithm perform well in terms of coherence, edge localization and high edge response, the implementation of edge detection algorithms provides useful means towards automated lineament mapping.

Finally, the HOUGH Transform is quite useful for line extraction, but it requires a proper parameter setting and adjustments to be applicable in different terrain and illumination conditions of natural scenes when applied to satellite data. Its performance to the Digital Elevation Model could be further investigated.

**Qixiang Ye et al. [23]** have proposed to find main edges meantime filter edges within texture regions. They have calculated pixel similarity degree around a pixel, have calculated a new gradient, and enforced a Canny like operator to detect and locate edges. In this way in addition gives very fine results.

**Jiang and Bunke [18]** have produced a novel method of edge detection based on scan line approximation. It gives edge strength measures that have directly geometric interpretation. Their algorithm is the strongest to many region based algorithms whence segmentation quality and computational efficiency.

**Hou and Kuo [12]** have introduced a new edge detection method that gives good edge detection accuracy than 4-connected, 8-connected and Sobel techniques. It is based upon simple arithmetic and logic operations, takes care with three procedures: image binarization, contraction and subtraction image. The algorithm works in automatic visual inspection. It depends on use of no threshold. Both of binary and grey scale images It can be used. Consequently, the first converted into binary images are the grey scale images. This way can be removed for a binary image. Then, the image is treated to get the contents of the inspected region. At the end, the contents are subtracted from the inspected region to produce the boundary.

**Kumar and Kathane [30]** have introduced that these edge detection operators can have better edge effect under the circumstances of obvious edge and low noise. But the actual collected image has lots of noises. So many noises may be considered as edge to be detected. In order to solve the problem, wavelet transformation is used to de noise in the paper. Yet its effect will be better if those simulation images processed above are again processed through edge thinning and tracking.

Although there are various edge detection methods in the domain of image edge detection certain disadvantages always exist. For example, restraining noise and keeping detail can't achieve optimal effect simultaneously. Hence acquire satisfactory result if choosing suitable edge detection operator according to specific situation in practice.

**Maini and Aggarwa [25]** have introduced that since edge detection is the initial step in object recognition, it is important to know the differences between edge detection techniques. In this paper we studied the most commonly used edge detection techniques of Gradient-based and Laplacian based Edge Detection. The software is developed using MATLAB 7.0. Gradient-based algorithms such as the Prewitt filter have a major drawback of being very sensitive to noise. The size of the kernel filter and coefficients are fixed and cannot be adapted to a given image.

An adaptive edge detection algorithm is necessary to provide a robust solution that is adaptable to the varying noise levels of these images to help distinguish valid image contents from visual artifacts introduced by noise. The performance of the Canny algorithm depends heavily on the adjustable parameters,  $\sigma$ , which is the standard deviation for the Gaussian filter, and the threshold values, 'T1' and 'T2'.  $\sigma$  also controls the size of the Gaussian filter.

The bigger the value for  $\sigma$ , the larger the size of the Gaussian filter becomes. This implies more blurring, necessary for noisy images, as well as detecting larger edges. As expected, however, the larger the scale of the Gaussian, the less accurate is the localization of the edge. Smaller values of  $\sigma$  imply a smaller Gaussian filter which limits the amount of blurring, maintaining finer edges in the image. The user can tailor the algorithm by adjusting these parameters to adapt to different environments.

Canny's edge detection algorithm is computationally more expensive compared to Sobel, Prewitt and Robert's operator. However, the Canny's edge detection algorithm performs better than all these operators under almost all scenarios. Evaluation of the images showed that under noisy conditions, Canny, LoG, Sobel, Prewitt, Roberts's exhibit better performance, respectively.

### **2.3 Conclusion of Literature Reviews**

It can be summarize how the previous studies are developed or modified methods of edge detection or in study and analysis of all edge detection techniques, in order to enhance time consumption of the best chosen of edge detection ratio and to get quality edge detection that it is needed.

Most of the previous studies in the edge detection methods evaluation field are not intended to directly use an automatic regular technique of edge detection evaluation algorithm.

The main idea of the proposed method in this paper is to enhance the time and the quality of choosing and selecting the best edge detection method for a certain image.

## **CHAPTER THREE:(MEASURING THE EDGE DETECTION METHODS)**



### **3.1 The Steps of the main Edge Detection**

The main steps in edge detection process are to determine the edges of any image must perform several operations, namely:

#### **Step 1: Smoothing**

The first step is filtering the noise in the original image before trying to locate and detect any edges. In order to blur filter, it should be computed using a simple mask, It is used obsessive in the algorithm wise. Once the account has been appropriate mask, filter and homogeneity can be completed using standard convolution methods. Convolution mask is commonly much smaller than the original image. Cause a result of this dropped a mask on the image, manipulating the square of pixels in time. The largest show of filter mask, and low sensitivity to noise is the detector. Localization error in the detected edges too increases a bit as the Gaussian width is increased.

Smoothing is used for two purposes essential: first in order to give a special effect to the components of the image and the second in order to get rid of the confusion in the picture, that's the positive side [29]. On the negative side, it may occur during treatment for the loss of some of the information and the picture here make trade-off between the losses of information or get rid of the confusion [34]. Accomplished smoothing the spatial information of the image required a certain point and a number of neighborhoods to this point is then applied a smoothing methods.

Most of the methods of treatment depend on the use of jamming digital filters to get rid of the negative impact of interference. The choice of methods of treatment depends on the quality of the added confusion to the picture [11].

## Step 2: Enhancement

It is the simplest technique used to detect edges and find the amount of variation in the value of the intensity chromaticity of the image data and the direction of covariance [26].

The value is calculated covariance between the point at the site (i, j) and adjacent points in both directions (horizontal and vertical) and as shown Site equations (1) and (2) respectively:

$$1. \nabla_x G(i, j) = G(i, j) - G(i - 1, j)$$

$$2. \nabla_y G(i, j) = G(i, j) - G(i, j - 1)$$

The resulting value may be positive or negative depending on the direction of covariance (negative direction or positive direction), It can make the result of the positive always, using the absolute value [26]. In equations (3) and (4) respectively:

$$3. |G| = \sqrt{G_x^2 + G_y^2} \approx |G_x| + |G_y|$$

$$4. \theta = \tan^{-1} \left( \frac{G_y}{G_x} \right)$$

## Step 3 Thresholding:-

After applying the process of discrimination and showing the variation in the values, marking points that can be part of the edge is then connecting these points to form a border that characterize the components and separating it from the background. This is after selecting a specific value called the threshold value. In general, the process can be defined as the process of rating threshold elements of the image into two zones represent the entity and the background, and have a mathematical representation as follows:

$$5. E(x, y) = \begin{cases} 1, & \text{if } \|\nabla f(x, y)\| > T \text{ for some threshold } T \\ 0, & \text{otherwise} \end{cases}$$

There are many algorithms used to select a threshold value appropriate but noticeable mostly because they are useful in the application to a specific image, and can be difficult to represent a number of different images, as well as the some of the values you choose choice intuitively based on the values of image points and the components required to appear in the picture.

Many of the efforts are still being made to provide a way to access all the requirements and all the conditions required [27]. It is the easiest methods used to determine the threshold value and that have been applied in this research to calculate the average values of the image data as follows:

$$T = \frac{1}{H * W} \sum_{i=0}^H \sum_{j=0}^w f(i, j)$$

Where H, represents height of image and W, represents width of image.

### 3.2 Methods of edge detection techniques

#### 3.2.1 Sobel Operator:

Operator consists of a pair of 3x3 convolution kernels as shown in Figure1.

One kernel is simply the other rotated by 90°.

-1	0	+1
-2	0	+2
-1	0	+1

Gx

-1	0	+1
-2	0	+2
-1	0	+1

Gy

Figure 1: Sobel operator mask

The kernels which are designed relative to the pixel grid according to respond to the maximum edges that be run vertically and horizontally, for every two vertical orientations there are one kernel. Separately, the kernels can be implemented to the input image until to get separate measurements of the gradient component in every direction (name these  $G_x$  and  $G_y$ ). Then, these can be integrated together to extract the absolute magnitude of the gradient at every point and the orientation of the gradient [17].

Given the magnitude of the gradient by:

$$|G| = \sqrt{G_x^2 + G_y^2} \approx |G_x| + |G_y|$$

The orientation angle of the edge (relative to the pixel grid), which led to the spatial gradient by:

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

### 3.2.2 Robert's Cross Operator:

Roberts Cross operator which performs spatial gradient measurement is a simple and quick to compute with 2-D image. The estimated absolute size of the spatial gradient of the input image at the point represents pixel values at each point in the output represent. Operator consists of a pair of  $2 \times 2$  convolution kernels as shown in Figure 2. One kernel is simply the other rotated by  $90^\circ$  [12]. This is very similar to the operator Sobel.

+1	0
0	-1

0	+1
-1	-1

Figure 2: Robert's cross operator masks.

These kernels are equipped for respond to the maximum edges, respectively, in 45° to the pixel grid, one kernel for every two vertical orientations. The kernels can be applied separately to the input image then, to output separate measurements of the gradient component in every direction (name these Gx and Gy). The absolute magnitude of the gradient at each point and the orientation of the gradient can be found that are be integrated together.

The magnitude of the gradient is given by:

$$|G| = \sqrt{G_x^2 + G_y^2} \approx |G_x| + |G_y|$$

Given orientation angle of the edge of rise to the spatial gradient (relative to the pixel grid orientation) and is set by:

$$\theta = \tan^{-1} \left( \frac{G_y}{G_x} \right) - \frac{3\pi}{4}$$

### 3.2.3 Prewitt's operator:

Prewitt operator [25] likes to the Sobel operator and it is used to detect vertical and horizontal edges in images.

-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 3: Prewitt gradient edge detector masks

### 3.2.4 Laplacian of Gaussian:

The Laplacian is a measure of properties second spatial derivative of the image. The Laplacian of an image focuses on the areas of quick intensity change and therefore it is often used to detect the edge. The Laplacian that is applied to the image at the first must be smoothed with something approaching a candidate smoothing filter in order to reduce its sensitivity to noise. This operator usually takes one image gray level image as input and yields as output level of the other gray.

Given the Laplacian  $L(x, y)$  of the image with pixel intensity values  $I(x, y)$  by this equation:

$$L(x, y) = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2}$$

The input image that is symbolized as a set of discrete pixels has to find a discrete convolution kernel that is closing to the second derivatives in the definition of The Laplacian [26]. There are contained three kernels commonly used in Figure 4.

0	1	0	1	1	1	-1	2	-1
1	-4	1	1	-8	1	2	-4	2
0	1	0	1	1	1	-1	2	-1

Figure 4: Three commonly used discrete approximations to the Laplacian filter.

These kernels are closed to second derivative measurement on the image because they are very sensitive to noise. To resist this thing, the picture is often smoothed by Gaussian filter is applied to the Laplacian.

In this pre-processing step it decreases even if little, the high frequency noise components prior to the step of differentiation. Actually, since the convolution operation is coherent, first of all can convolve the Laplacian filter with the Gaussian smoothing filter, and then convolve the image with hybrid filter to get the required result. With this previous working on this way has two advantages: as both the Gaussian and the Laplacian kernels are commonly much smaller than the image, this method commonly requires fewer mathematical operations.

The LOG ('Laplacian of Gaussian') [32] kernels can be calculated in advance so only one convolution needs to be completed at run-time on the image. The 2-D LOG function [5] centered on zero and with Gaussian standard deviation  $\sigma$  has the form:

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

And it is shown in the Figure of 5:

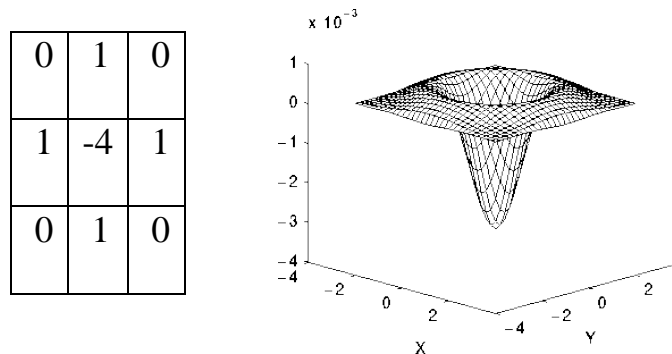


Figure 5: The 2-D LOG function, the x and y axes are marked in standard deviations ( $\sigma$ ).

A discrete kernel that approximate this function (for a Gaussian  $\sigma = 1.4$ ) as it is shown in Figure of 6.

$$\begin{bmatrix} 0 & 0 & 3 & 2 & 2 & 2 & 3 & 0 & 0 \\ 0 & 2 & 3 & 5 & 5 & 5 & 3 & 2 & 0 \\ 3 & 3 & 5 & 3 & 0 & 3 & 5 & 3 & 3 \\ 2 & 5 & 3 & -12 & -23 & -12 & 3 & 5 & 2 \\ 2 & 5 & 3 & -23 & -40 & -23 & 0 & 5 & 2 \\ 2 & 5 & 3 & -12 & -23 & -12 & 3 & 5 & 2 \\ 3 & 3 & 5 & 3 & 0 & 3 & 5 & 3 & 3 \\ 0 & 2 & 3 & 5 & 5 & 5 & 3 & 2 & 0 \\ 0 & 0 & 3 & 2 & 2 & 2 & 3 & 0 & 0 \end{bmatrix}$$

Figure 6: LOG function has closing discrete with Gaussian  $\sigma= 1.4$ .

It is noticed that as the Gaussian is made increasingly narrow, the LOG kernel turns into the same as the simple Laplacian kernels shown in figure 4, and that is because smoothing with a very narrow Gaussian ( $\sigma < 0.5$  pixels) on a discrete grid has no effect with it. Hence on a discrete grid, the simple Laplacian can be looked as a limiting condition of the LOG for narrow Gaussians [26, 32, and 34].

### 3.2.5 Canny edge detection algorithm

The Canny edge detection algorithm is known to many techniques as the optimal edge detector. The objectives of Canny to enhance a lot of edge detectors already out at the time he began his work. He was succeeded in obtaining his objective by his thoughts and methods as in his works that been founded in his paper of, "A Computational Approach to Edge Detection" [16]. In his paper, the researcher has gone ahead of a list of criteria to progress methods of edge detection. The first thing of that to work is considered in decreasing of error rate. The important edges which are happening in images should not be missed and also that there be no missed to non-edges.



The second criterion is that the edge points be well localized as much as it can. The concept of another, the distance between the edge pixels as established by the detector and the actual edge is to be at a minimum. The third criterion is to have only one response to a single edge. This was executed because of the first two criteria were not fundamental enough to fully eliminate the possibility of various responses to an edge.

Based on these criteria, the canny edge detector in its first step, smoothes the image to eliminate the noise, then finds the image gradient to highlight regions with high spatial derivatives. The algorithm of edge detection after that is route along these regions and suppresses any pixel that isn't at the maximum (non-maximum suppression). The gradient array is furthermore reduced by hysteresis which is used to track over the lasting pixels that have not been suppressed. Hysteresis in its working uses two thresholds and if the magnitude is under the first threshold, it is set to zero (put a non-edge). If the magnitude is over the high threshold, it is put an edge. And if it is the magnitude between the two thresholds, then it is set to zero except if there is a track from this pixel to a pixel with a gradient over the second threshold.

The steps progress of using canny Algorithm as following:

### **Progress 1:-**

In order to perform the algorithm for detecting the edge of the wise, the user must follow a series of steps. The first step is filtering either noise in the original image before trying to locate and detect any edges. In order to blur filter can be computed using a simple mask, it is used obsessive in the algorithm wise. Once the account has been appropriate mask, filter and homogeneity can be completed using standard convolution methods.

Convolution mask is commonly much smaller than the original image. Cause a result of this dropped a mask on the image, manipulating the square of pixels in time. The largest show of filter mask, and low sensitivity to noise is the detector. Localization error in the detected edges too increases a bit as the Gaussian width is increased.

**Progress 2:-**

Next smoothing the image and clear the noise, then the next step is to detect the edge strength by picking the gradient of the image. Sobel operator performs 2-D spatial gradient measurement on the image. Then, the near size of the absolute regression (edge strength) at every point can be found. Sobel operator [19] uses a pair of 3X3 convolution masks, one evaluating the gradient in the direction Q (columns) and the other evaluating the gradient in the direction of Y (rows). They are shown:

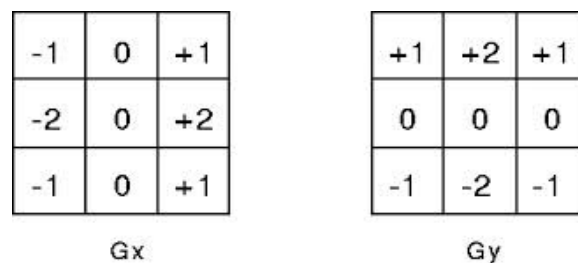


Figure 7, pair of 3X3 convolution masks

The magnitude, or edge strength, of the gradient is then nearest using the formula:

$$|G| = \sqrt{Gx^2 + Gy^2} \approx |Gx| + |Gy|$$

### Progress 3:-

The direction of the edge is calculated using the gradient in the x and y directions. But, an error will be produced when sum X is equal to zero. So in the code there has to be a limitation set whenever this takes place. Whatsoever the gradient in the x-direction is equal to zero, the edge direction has to be equal to 90° or 0°, based on what the value of the gradient in the y-direction is equal to. If Gy was a value of zero, the edge direction will equal 0°. On the other hand the edge direction will equal 90°. The formula for locating the edge direction is just:

$$\theta = \tan^{-1}\left(\frac{Gy}{Gx}\right)$$

### Progress 4:-

The moment that the edge direction is known, the next step is to bind the edge direction to a direction that can be traced in the image. Thus if the pixels of a 5x5 image are aligned 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 8, pixels of a 5x5 image

Then, it can be seen by seeing at pixel "a", there are only four directions when describing possible at the surrounding pixels - 0° (within horizontal direction), 45° (over positive diagonal), 90° (within vertical direction), or 135° (over negative diagonal).

Thus now the edge orientation has to be determined into one of these four directions based on which direction it is nearest to (e.g. if the orientation angle is found to be  $3^\circ$ , make it  $0^\circ$ ). Think of this as taking a half of circle and dividing it into 5 regions.

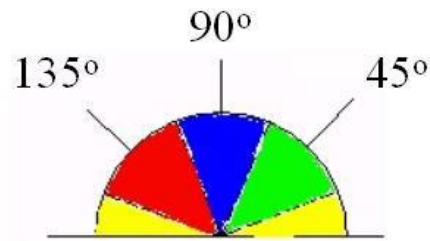


Figure 9, a half circle with 5 regions

Thus, any edge direction drop in the yellow range ( $0^\circ$  to  $22.5^\circ$  and  $157.5^\circ$  to  $180^\circ$ ) is set to  $0^\circ$ . Any edge direction drop in the green range ( $22.5^\circ$  to  $67.5^\circ$ ) is set to  $45^\circ$ . Any edge direction drop in the blue range ( $67.5^\circ$  to  $112.5^\circ$ ) is set to  $90^\circ$ . At last, any edge direction drop in the red range ( $112.5^\circ$  to  $157.5^\circ$ ) is set to  $135^\circ$ .

#### **Progress 5:-**

Next the edge directions are known, non-maximum suppression now has to be utilized. Non-maximum suppression is used to trace over the edge in the edge tendency and suppress each pixel value (sets it equal to 0) that is not looked to be an edge. This will give a thin line in the output of the image.

### **Progress 6:-**

At the end, hysteresis [15] is used as a means of removing streaking. Streaking is the breaking up of an edge contour resulting from the operator output fluctuating up and down the threshold. If a single threshold,  $T_1$  is applied to the image, and an edge has an average strength equal to  $T_1$ , then due to noise, there will be examples where the edge dips down the threshold. Evenly it will also expand up the threshold making an edge resemble as a dashed line. To bypass this, hysteresis uses two thresholds, a low and a high. Any pixel in the image that has a value greater than  $T_1$  is assumed to be an edge pixel, and remarkable as such immediately.

Then, any pixels that are linked to this edge pixel and that have a value greater than  $T_2$  are as well selected as edge pixels. If you think of following an edge, you need a gradient of  $T_2$  to begin but you don't stop till you hit a gradient down  $T_1$ .

### **3.3 Filtering**

In normal cases to remove noise from images. It must to use filters and there are two types of filters which are linear and non-linear filters. Linear filters work with blurring sharp edges with destroying lines and other fine image details, in addition implement poorly in the presence of signal-dependent noise. However, with non-linear filters, the noise is eliminated without any attempt to identify correctly.

Perform median filtering at a point in an image to sort the values of the pixel in the neighborhood, determine their median, and assign that value to the corresponding pixel in the filtered image. The median filter is appropriate for images that are affected by Salt & Pepper noise [33].

The median filter has proved in it is working, which is very useful in many image processing applications.

Median filter is commonly used in methods of removal impulse noise in view of its de-noising ability and computational efficiency [10]. Due to implementing of traditional median filter for removal of such type of noise, the acceptable results gives relatively which are shown in restoring of brightness drops, objects edges and local peaks in noise corrupted images [3]. From here, the process of edge detection maybe by some de-noises process to smooth the image. These two processes should be appropriate with each other due to perform a better result. When numbering the standard deviation it will be a sensitive situation because of the noisy pixels were remedied as noise-free pixels [8].

The semi-optimal edge detector was executed by, at the first, eliminating the noisy pixels to get rid of the high dispersion between 2x2 window pixels. This step would highly enhance the proposed edge detector by preserving the edges as possible. As a result, the compatibility between the median filter and the proposed edge detector is needed to achieve the proposed edge detection process [8].

### **3.4 Pratt Measure**

The researchers note that to assess the performance of certain technical modalities, there are series of scheduled scientific experiments determined by objective and clear performance efficiency.

The actual evaluation in the case of the determinants of the edge of the performance is based on the vision system of the human being (i.e., looking at the results) [22], Since there is no efficient computer vision system to lead this purpose, so it may be appropriate in some cases to represent the results of visual identifiers for each along with some of them and leave the evaluation of the user's personal estimate [7]. And appreciation here depends on the experience of the person and vision system has.

The thesis here may be ambiguous and cannot be used to measure the performance of the specified but is only used to prove the success or failure specified in the correct identification of the components [24]. As formula following:

$$R = \frac{1}{I_N} \cdot \sum_{i=0}^{I_A} \frac{1}{1 + \alpha \cdot d^2}$$

Where,  $I_1$ = Number of points edge in the ideal image,  $I_A$ = the real number of points which are calculated by a specific edge. And the largest value between  $i_1$ ,  $I_A$ , And  $\alpha$ = Scale constant is chosen to balance the site point and it is equal to 1/9. Also  $d$ = is the distance between the edge of the real points and edge points idealism. If the value resulting from the application of the scale of the one approaching [22].

The interests of the plants are hard edge settlement points that appear in the twisted sites for its correct location that can be considered an efficient method of specifically

this. Standard is applied Pratt on each method of selection of the five to get an automatic objective evaluation of the performance of each method, and the values are adjusted dealing each way to get the best possible performance of the method, and is used the same image with the modalities five gradations of gray scale (0, 255).

Way of the process set real locations of the edges by using the methods of selection, thereafter selects the edges after that applies the ideal equation of the scale. The 'figure of merit' measure of Pratt is beneficial for assessment the execution of edge detectors. This measure uses the distance between all pairs of points due to the determination with accuracy, the difference between the contours [13]. The 'figure of merit' of Pratt, which rate the similarity between two contours, is called as:

$$R = \frac{1}{\text{Max}(I_N, I_1)} \cdot \sum_{i=0}^{I_A} \frac{1}{1 + \alpha \cdot d^2}$$

Where respectively  $I_N$  and  $I_1$  are the points of edges in the foreground image and background image,  $d_i$  is the distance between the edge pixel and the nearest edge pixel of the background and  $\alpha$  is an empirical calibration constant and was used  $\alpha = 1/9$  as mentioned before, optimal value established by Pratt [13].

The figure of merit of Pratt is an indicator of the quality of edge, and reflects the overall behavior of the distances between the edges, being a relative measure which varies in the range [0, 1], where 1 represents the optimal value, i.e., the edges detected coincide with the ground truth [14].

$$d = \sqrt{(x_I - x_r)^2 + (y_I - y_r)^2}$$



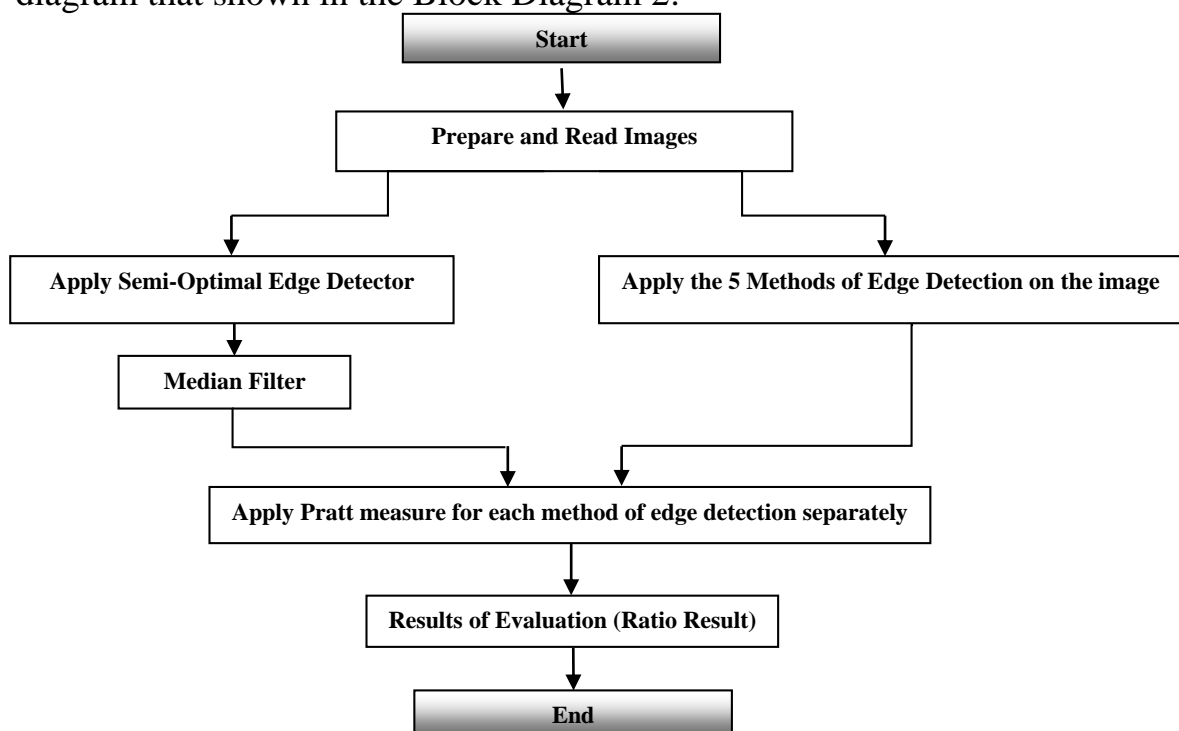
## **CHAPTER FOUR:A-EDV (THE PROPOSED ALGORITHM)**

## 4.1 An Automatic Edge Detection Evaluation

This thesis will be evaluate the performance of edge detection by choosing five famous methods known as (Canny, Laplacian of Gaussian, Prewitt, Sobel, Roberts) and the images of each application method with grayscale to detect the performance of each of them and by using a software of MATLAB R2009a of version 7.8.0.347, software is executed for each operator with specification of computer device by a core processor of i3 and RAM of 2GB.

To complete the evaluation process in software to use an automatic code that to compare the performance of these five methods using Pratt Figure of Merit equation, that computing the increment percentage in the detected edges, the decrement percentage in the edge points and the correct location of the edge in every method to get the required ratio of each edge detection method separately.

The proposed A-EDV method can be described by the following block diagram that shown in the Block Diagram 2:



Block Diagram 2, the Block Diagram of A-EDV.

In this thesis could try to get the edge detection results with GUI through evaluation algorithm in order to pick out the best detection, according to the following algorithm steps:

- 1- Reading the image.
- 2- Converting the image to gray scale.
- 3- Identifying the image Size.
- 4- Calculating the standard deviation to each four pixels array to determine the edges that exceeds 7 SD.
- 5- Applying median filter after SD.
- 6- Display the SD Image.
- 7- Applying the 5 methods of edge detection.
- 8- Apply Pratt measure for each method of edge detection separately, by finding the maximum number of edge points between the ideal and real images
  - 8.1- Number of edges points of the ideal image.
  - 8.2- Number of edges points of the real image.
  - 8.3- Maximum number edges points.
  - 8.4- Finding the coordinates of edges points.
  - 8.5- Calculating the distances between the edge point in the real image and the corresponding one at the ideal.
  - 8.6- Testing the edge point in both images, if match then, the distance is zero else we have find  $D(i1)=0$ .
  - 8.7- The nearest point to calculate the distance between it in the ideal image and the point at the real one.
  - 8.8- Loop to find the nearest edge point in the ideal image and have to make sure the points are within the image boundary.
  - 8.9- Take a part of the ideal image to find the edges points within it.

8.10- Find the edges points and if it founds then, calculating the distances to find the nearest edge point.

9- Completing the Pratt equation calculation for each method of edge detection.

10- Results of evaluation through displaying images which are original, standard deviation, Prewitt, Sobel, Canny, Roberts and LOG image in graphical user interface with displaying also the results of best operator with its ratio value.

#### **4.2 A-EDV How Does It Work?**

A new automatic edge detection evaluation method of coding comes to evaluate the original gray scale image (A-EDV) has been developed and executed in MATLAB R2009a of version 7.8.0.347 platform. A-EDV code presents the work step by step; code has been implemented on the grayscale image, and then shows the results after each step until closing to the final result.

Step1 in A-EDV method is to read the original grayscale image that shown in Figure 10 on to the workspace of the MATLAB R2009a.



Figure 10, Lena.png.

**Step 1:** Reading the image, converting the image to gray scale and preparing the image Size.

```
im1 = evalin('base','imagein'); %reading the image
im1g = rgb2gray(im1); %Converting the image to gray scale
[ym,xm] = size(im1g); %Preparing the image size
im1bw = im2bw(im1g);
edg1 = 0&im1bw;
```

**Step 2:** Calculating the standard deviation to each four pixels array to determine the edges that exceeds 7 SD. Then applying median filter and display of SD Image.

```
for i = 1:xm-1
%calculating the SD to each four pixels array to determine the edges that
exceeds 7 SD
    for j = 1:ym-1
        val = std2([im1g(j,i) im1g(j,i+1);im1g(j+1,i) im1g(j+1,i+1)]);
        if val >= 7
            edg1(j,i) = 1;
        else
            edg1(j,i) = 0;
        end
    end
end
edg1f = medfilt2(edg1); %applying median filter
axes(handles.axes2);imshow(edg1f)
```

**Step 3:** Applying the 5 methods of edge detection and displaying the resulted images.

`%applying the methods of edge detection`

```
[BW_sobel,thresh_sobel] = edge(im1g,'sobel');  
[BW_roberts,thresh_roberts] = edge(im1g,'roberts');  
[BW_prewitt,thresh_prewitt] = edge(im1g,'prewitt');  
[BW_canny,thresh_canny] = edge(im1g,'canny');  
[BW_log,thresh_log] = edge(im1g,'log');
```

`%displayng the resulted images`

```
axes(handles.axes3); imshow(BW_sobel);  
axes(handles.axes4); imshow(BW_roberts);  
axes(handles.axes5); imshow(BW_prewitt);  
axes(handles.axes6); imshow(BW_canny);  
axes(handles.axes7); imshow(BW_log);
```

**Step 4:** Apply Pratt measure for each method of edge detection separately, by finding the maximum number of edge points between the ideal and real images, in example Pratt of Sobel by depends on these procedures:-

- Number of edges points of the ideal image.
- Number of edges points of the real image.
- Maximum number edges points.
- Finding the coordinates of edges points.
- Calculating the distances between the edge point in the real image and the corresponding one at the ideal.
- Testing the edge point in both images, if match then, the distance is zero else we have find  $D(i1)=0$ .
- The nearest point to calculate the distance between it in the ideal image and the point at the real one.

- Loop to find the nearest edge point in the ideal image and have to make sure the points are within the image boundary.
- take a part of the ideal image to find the edges points within it.
- find the edges points and if it finds then, calculating the distances to find the nearest edge point.

`%pratt for sobel`

`Applying the Pratt equation by finding the maximum number of edge points between the:- %Ideal and real images`

`N_Ideal = sum(sum(BW_sobel));%number of edges points of the ideal image`

`N_real = sum(sum(edg1f)); %number of edges points of the real image`

`N_max = max([N_Ideal N_real]); %maximum number edges points`

`[y1,x1] = find(edg1f); %finding the coordinates of edges points`

`D = 0;`

`%calculating the distances between the edge point in the real image and the corresponding one at the ideal`

`for i1 = 1:length(x1)`

`test1 = BW_sobel(y1(i1),x1(i1));`

`if test1 == 1`

`%testing the edge point in both images, if match then the distance is zero else we have find`

`D(i1) = 0;`

`%the nearest point to calculate the distance between it in the ideal image and the point at the real one`

`else`

`for n = 1:min(size(BW_sobel))`

%this loop to find the nearest edge point in the ideal image

```
xp_Im1 = x1(i1);  
yp_Im1 = y1(i1);  
yt1 = yp_Im1-n;  
xt1 = xp_Im1-n;  
yt2 = yp_Im1+n;  
xt2 = xp_Im1+n;  
size1 = size(BW_sobel);  
if yt2 > size1(1)
```

%to make sure the points are within the image boundary

```
    yt2 = size1(1);    %  
end  
if yt1 < 1            %  
    yt1 = 1;          %  
end  
if xt2 > size1(2)    %  
    xt2 = size1(2);  %  
end  
if xt1 < 1  
    xt1 = 1;  
end  
part1 = BW_sobel(yt1:yt2,xt1:xt2);
```



```

%take a part of the ideal image to find the edges points within it
    [yp_Ref1,xp_Ref1] = find(part1);
%find the edges points
    if yp_Ref1 > 0          %edges points found
        Dt = 0;
        for m = 1:length(xp_Ref1)
            %calculating the distances to find the nearest edge point
                Dt(m) = (xp_Ref1(m) - xp_Im1)^2 + (yp_Ref1(m) -
yp_Im1)^2;
            end
            D(i1) = min(Dt);
            %taking the one with the minimum distance
                break;
            end
        end
    end
end
end

```

**Step 5:** Finally, completing the Pratt equation calculation for each method of edge detection. And the results of evaluation through displaying images which are original, standard deviation, Prewitt, Sobel, Canny, Roberts and LOG image in graphical user interface with displaying also the results of best operator with its ratio value as shown in Figure 11.

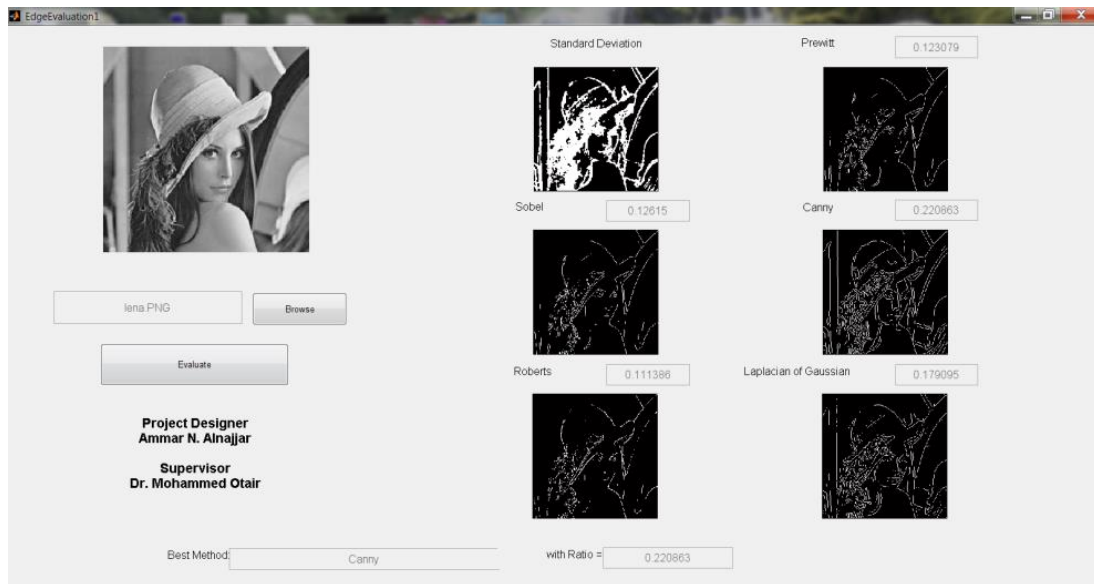


Figure 11, the final GUI Results.

$D_{total} = (D/9) + 1;$       %completing the Pratt equation calculation

$D_{totaln} = 1./D_{total};$

$Acc1 = \text{sum}(D_{totaln});$

$R_{sobel} = Acc1/N_{max};$

$\text{set}(\text{handles.edit2}, 'String', R_{sobel});$

Last step the GUI that shown in figure of 9 displays the last results of A-EDV implementation on the original image and gives best method of edge detection with the ratio value of all results.

## **CHAPTER FIVE:(THE EXPERIMENTS & RESULTS)**

## 5.1 Experiments & Results

The algorithm which is used in automatic evaluation is MATLAB R2009a of version 7.8.0.347, the software will be executed for each operator with specification of computer of i3 processor and 2GB RAM.

The Experiments implemented on a set of different images ranging with types of bmp, jpg, and png. The Figure of 12 explains the results algorithm of A-EDV implementation after applying on the chosen image that its type of jpg.

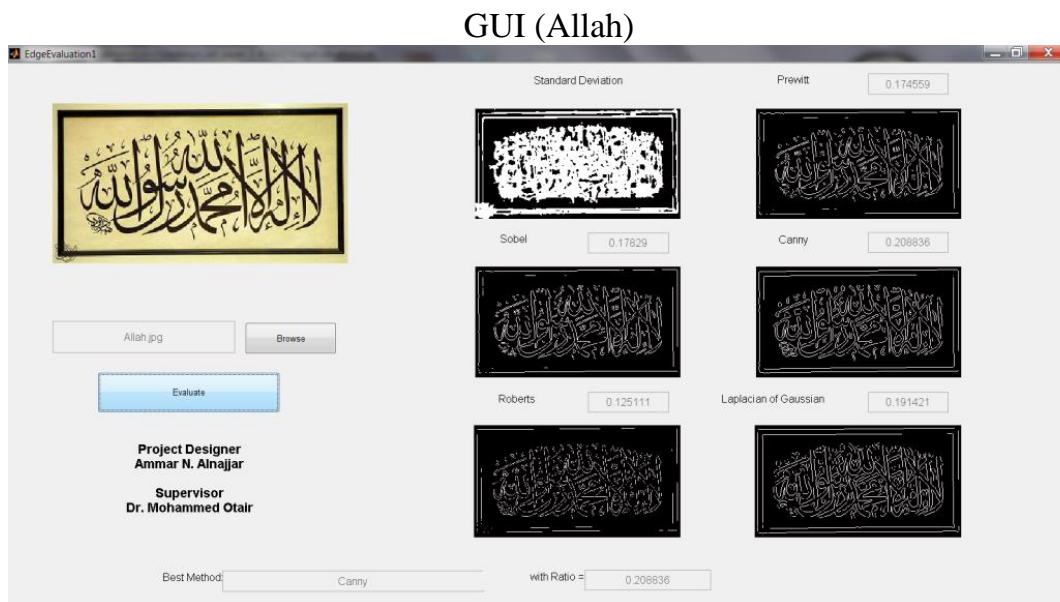
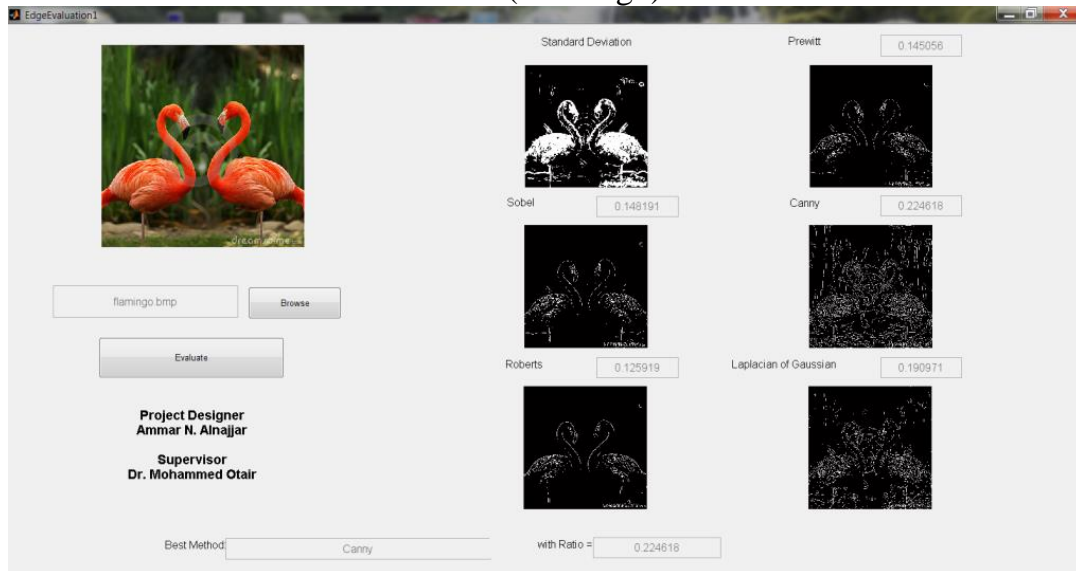


Figure 12, Allah.jpg

The Figure of 13 explains the results of A-EDV implementation after applying on the chosen image that its type of .bmp.

GUI (Flamingo)



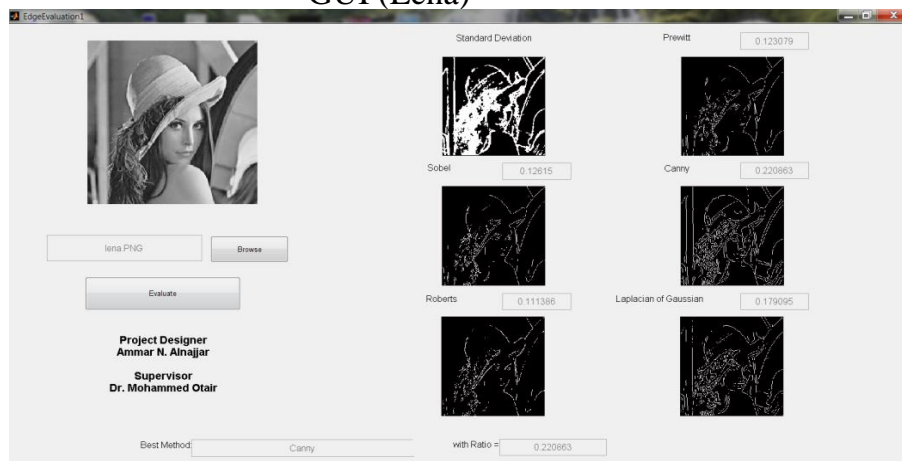
BMP Image SD Prewitt Sobel Canny Roberts LOG



Figure 13, Flamingo.bmp.

The Figure of 14 explains the results of A-EDV implementation after applying on the chosen image that its type of png.

GUI (Lena)



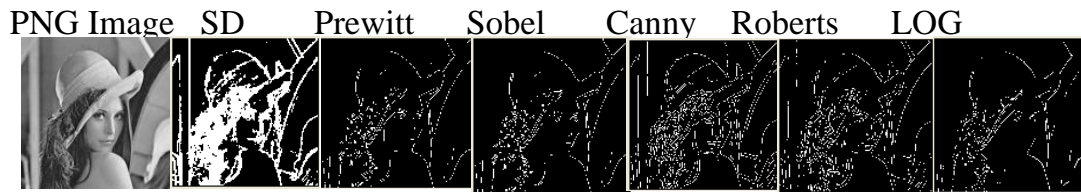


Figure 14, Lena.png.

The table 1 concludes the results of the application or implementation A-EDV Program with many of pictures in different formats.

No.	Image Name & Extension	Size in Bits	Pratt Ratio of Prewitt	Pratt Ratio of Sobel	Pratt Ratio of Canny	Pratt Ratio of Roberts	Pratt Ratio of LOG	The Best (A-EDV) Ratio Nearest To The Value of ONE
1	Allah.jpg	41842	0.174559	0.17829	0.208836	0.125111	0.191421	Canny
2	Flamingo.bmp	480054	0.145056	0.148191	0.224618	0.125919	0.190971	Canny
3	Lena.png	76574	0.123079	0.12615	0.220863	0.111386	0.179095	Canny
4	Rice.bmp	196662	0.239956	0.240658	0.237294	0.210888	0.218322	Sobel
5	Bird.jpg	200855	0.093749	0.0947141	0.193892	0.0603244	0.156281	Canny
6	Bird1.bmp	1042038	0.10865	0.109591	0.235137	0.0914627	0.185275	Canny
7	Golden.jpg	68503	0.0690545	0.0697473	0.166487	0.0392227	0.12695	Canny
8	Kitchen.jpg	69191	0.142345	0.142322	0.238387	0.130281	0.178761	Canny
9	Mohammed.bmp	270054	0.15021	0.153394	0.190245	0.148064	0.172359	Canny
10	Panda.bmp	370502	0.0667919	0.0679751	0.174274	0.0664341	0.121644	Canny
11	Poly.bmp	344502	0.440634	0.450253	0.328718	0.573183	0.0650003	Roberts
12	Pout.tif	65364	0.118773	0.124369	0.107381	0.0603416	0.227264	LOG
13	Screws.jpg	29981	0.186316	0.1864	0.26541	0.171126	0.227509	Canny
14	Shadow.jpg	741794	0.188586	0.189466	0.268881	0.13647	0.207059	Canny
15	Tala.jpg	223213	0.146547	0.146269	0.0642777	0.176838	0.0665598	Roberts
16	Tools.jpg	22196	0.152555	0.153142	0.200829	0.160906	0.177041	Canny
17	Zag.jpg	44010	0.0532636	0.0600204	0.154081	0.0367407	0.148387	Canny
18	AAU.bmp	102174	0.0940041	0.0969855	0.282843	0.0393011	0.18488	Canny

Table 1, the resulted ratio values

As the results in the Tables 1, it is observed that the proposed A-EDV evaluates the five edge detection operators with choosing the best operator in edge detection. The experiment results show that canny operator in mostly give the best value which is nearest to the ratio of one according to the Pratt measure that explained with A-EDV program.

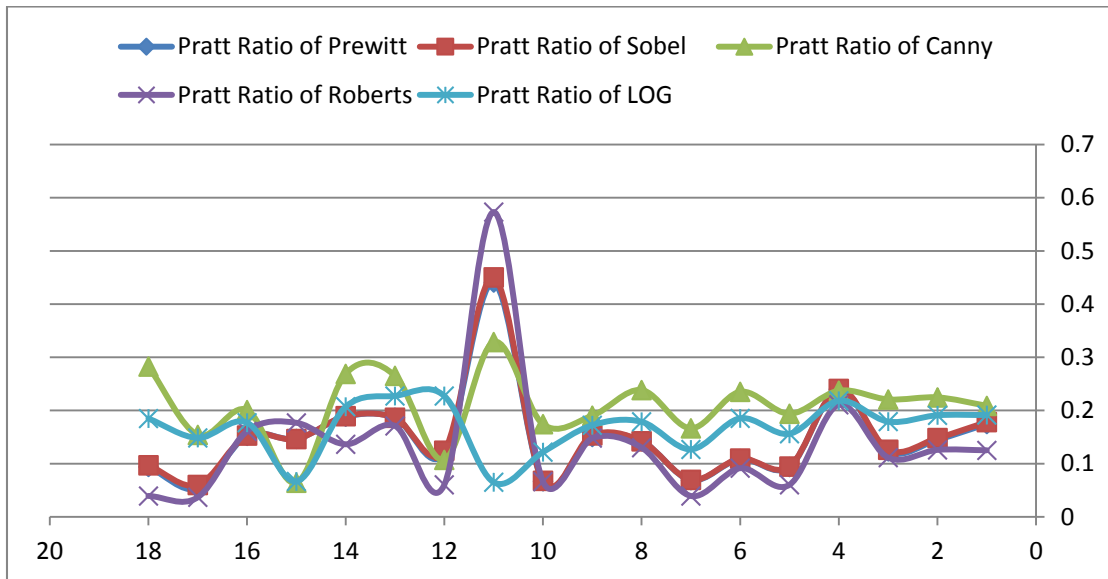


Figure 15 Edge Detection Methods Ratio Chart

Through Figure 15 chart can be comparing the performance of the techniques with A-EDV performance, regarding the Pratt Measure ratio, and been reached as a result that in all conditions the ratios for all images that have been compared using A-EDV method for the five edge detection methods of techniques, regardless of image type or size, and by using A-EDV method there is always the best choice technique which is the best in edge detection through the ratio that is nearest to the value of one.

**CHAPTER SIX**  
**(RECOMMENDATIONS & CONCLUSION)**



## 6.1 RECOMMENDATIONS

The focus has been on making A-EDV application work well together with the existing MATLAB R2009a of version 7.8.0.347 Software. The most important issue has been to get it all work; consequently there are improvements to make.

It would have been desirable if the run of the file be on the directory to the MATLAB path instead of coming message to put the directory of file to take it from the path for execution. In the present situation it is impossible to do execution of new image and there is another image running or busy with execution, it is preferable to end the execution of an image or to wait until the results of the before image chosen so to choose a new image for a new execution.

The version of the MATLAB mostly working with all new updated versions of MATLAB R2008a, MATLAB R2009a and MATLAB R2009 $\beta$

## 6.2 CONCLUSION

The conclusion that it must be required of a better measure for edge detection evaluation, and that some of famous operators should be tested to get the optimal edge detection. In advance, the principle purpose for edge detection is to extracting significant features can be get from the edges of an image (e.g., corners, lines, curves). These features are used by higher-level computer vision algorithms (e.g., recognition) [31].

In Addition, the objective this research in analyzing various image edge detection techniques to evaluates the performance of the modalities of the determinants of the edges by choosing the five methods and the application of all (Canny, Laplacian of Gaussian, Prewitt, Sobel, Roberts) well-known methods of image grayscale to get knowledge of the performance of each method towards an automatic method for edge detection evaluation based on semi-optimal edge detector (A-EDV).

Furthermore, The objective of this work is to detect the advantages and disadvantages for every operator and see what the appropriate application to be useful also to help in the future researches who depends on edge detection methods to help in finding the appropriate operator that must be chosen automatically the an algorithm evaluation. Therefore, these measures may be useful in consuming a less of time and to evaluate new algorithms for detecting edges in a better quality. This should be done to facilitate the visualization of the performance of detectors, allowing a more careful interpretation on performance of edge detectors and hence validation of the proposed method.

## REFERENCES

- [1] Ammar N. A., and Mohammed O. O., "An Automatic Method for Edge Detection Base on Semi-Optimal Edge Detector", International Journal of Computer Information Systems, Vol. 8, No.3, March 2014. ISSN: [2229-5208].
- [2] Argialas, D. P., Mavrantza, O. D., "COMPARISON OF EDGE DETECTION AND HOUGH TRANSFORM TECHNIQUES FOR THE EXTRACTION OF GEOLOGIC FEATURES", The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Vol. 34, 2004.
- [3] Bovik A. C., Huang T. S., and Munson D. C. 1985. Edge-sensitive image restoration using order constrained least squares methods. IEEE Trans. Acoust., Speech, Signal Processing, vol. 33, pp. 1253–1263.
- [4] E. Argyle. "Techniques for edge detection", Proc. IEEE, vol. 59, pp. 285-286, 1971.
- [5] Elhabiby, M., Elsharkawy,A., El-Sheimy, N., "Second Generation Curvelet Transforms Vs Wavelet transforms and Canny Edge Detector for Edge Detection from WorldView-2 data", International Journal of Computer Science & Engineering Survey (IJCSES) Vol.3, No.4, August 2012.
- [6] El-Sayed. M., Abd-El Hafeez.T., "New Edge Detection Technique based on the Shannon Entropy in Gray Level Images", International Journal on Computer Science and Engineering. Vol. 3 No. 6 June 2011.
- [7] F. Bergholm. "Edge focusing," in Proc. 8th Int. Conf. Pattern Recognition, Paris, France, pp. 597- 600, 1986.

- [8] Firas A. Jassim., "Semi-Optimal Edge Detector based on Simple Standard Deviation with Adjusted Thresholding", International Journal of Computer Applications (0975 – 8887) Volume 68– No.2, April 2013.
- [9] Gao, W., Yang, L., Zhang, X., Liu, H., "An Improved Sobel Edge Detection", 2010.
- [10] Gonzalez R. C. and Woods R. E. 2001. Digital Image Processing. Upper Saddle River, NJ: Prentice-Hall.
- [11] G.T. Shrivakshan And Dr. C. 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.
- [12] Hou, T. H. and Kuo, W.L., 1997. "A New Edge Detection method for Automatic Visual Inspection", international Journal of Advanced Manufacturing Technology, (1997) 13:407412 © 1997 Springer-Verlag London Limited.
- [13] I. A. Abdou and W. Pratt, "Quantitative design and evaluation of enhancement/thresholding edge detectors," in Proceedings of the IEEE, vol. 67, no. 5, 1979, pp. 753–766.
- [14] Inês Aparecida Gasparotto Boaventura and Adilson Gonzaga, "Method to Evaluate the Performance of Edge Detector".
- [15] J. Canny. "Finding edges and lines in image". Master's thesis, MIT, 1983.
- [16] Jensen, J. R. (1986) Introductory Digital Image Processing, A Remote Sensing Perspective, Prentice - Hall.
- [17] J. F. Canny. "A computational approach to edge detection". IEEE Trans. Pattern Anal. Machine Intell., vol. PAMI-8, no. 6, pp. 679-697, 1986.

- [18] Jiang, X. and Bunke, H. 1998. "Edge Detection in Range Images based on Scan Line Approximation", Computer Vision and Image Understanding, Vol. 73, No. 2, February, pp. 183–199, 1999 Article ID cviu.1998.0715, available online at <http://www.idealibrary.com>.
- [19] J. Matthews. "An introduction to edge detection: The Sobel edge detector" at <http://www.generation5.org/content/2002/im01.asp>, 2002.
- [20] K. Bowyer, C. Kranenburg, and S. Dougherty, "Edge detector evaluation using empirical roc curves," Computer Vision and Image Understanding, vol. 84, pp. 77– 103, 2001.
- [21] M. Viergever, H. Stiehl, R. Klette, and K. Vincken, Performance Characterization and Evaluation of Computer Vision Algorithms. Kluwer Academic, 2000.
- [22] Pratt W.K. Digital Image Processing. Second Edition, J. Wiley & Sons, New York, 1991.
- [23] Qixiang, Ye, Wen, G., Weiquiang, W. 2003. "A New Texture Insensitive Edge Detection Method", Institute of Computing Technology, Chinese Academy of Sciences ,China, ICICS-PCM 2003, 15-18 Dec 2003, Singapore.
- [24] R. A. Kirsch. "Computer determination of the constituent structure of biomedical images". Comput. Biomed. Res., vol. 4, pp. 315-328, 1971.
- [25] Raman Maini & Dr. Himanshu Aggarwal. "Study and comparison of various Image Detection Techniques" Proc. IJIP, vol. 3, Issue (1). 2009.
- [26] R. C. Gonzalez and R. E. Woods. "Digital Image Processing". 2nd ed. Prentice Hall, 2002.

- [27] Rosenfeld, A. and A. C. Kak , (1976) Digital Picture Processing, ACADEMIC PRESS.
- [28] Senthilkumaran, N., Rajesh, R.,” Edge Detection Techniques for Image Segmentation – A Survey of Soft Computing Approaches”, International Journal of Recent Trends in Engineering, Vol. 1, No. 2, 2009.
- [29] Sharma,s., Nitasha, Rani, u.,” Ellipse Detection Using Canny Edge Detection Algorithm”, International Journal of Computer Applications & Information Technology Vol. I, Issue II, September 2012.
- [30] Sushil Kumar Singh, Aruna Kathane “Various Methods for Edge Detection in Digital Image Processing” International Journal of Computer Science and Technology (IJCSST) Vol. 2, Issue 2, June 2011.
- [31] Trucco, Chapt 4 AND Jain et al., Chapt 5.
- [32] Umbaugh, S. E. (1998) Computer Vision and Image Processing, A practical Approach Using CVIP tools, Prentice Hall PTR.
- [33] V. Thirilogasundari, V. Suresh babu, S. Agatha Janet, “Fuzzy Based Salt and Pepper Noise Removal Using Adaptive Switching Median Filter”, Procedia Engineering, Volume 38, 2012, Pages 2858-2865.
- [35] 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.

# APPENDIX

## APPENDIX-A (A-EDV CODE)

% THIS PROGRAM IS TO AN AUTOMATIC METHOD FOR EDGE DETECTION EVALUATION  
BASED ON SEMI-OPTIMAL EDGE DETECTOR

% THIS PROGRAM IS DONE BY AMMAR A. N. ALNAJJAR

**function** varargout = EdgeEvaluation1(varargin)

% EDGEEVALUATION1 M-file for EdgeEvaluation1.fig

% EDGEEVALUATION1, by itself, creates a new EDGEEVALUATION1 or raises the existing  
% singleton\*.

%

% H = EDGEEVALUATION1 returns the handle to a new EDGEEVALUATION1 or the handle to  
% the existing singleton\*.

%

% EDGEEVALUATION1('CALLBACK',hObject,eventData,handles,...) calls the local  
% function named CALLBACK in EDGEEVALUATION1.M with the given input arguments.

%

% EDGEEVALUATION1('Property','Value',...) creates a new EDGEEVALUATION1 or raises the  
% existing singleton\*. Starting from the left, property value pairs are  
% applied to the GUI before EdgeEvaluation1\_OpeningFcn gets called. An  
% unrecognized property name or invalid value makes property application  
% stop. All inputs are passed to EdgeEvaluation1\_OpeningFcn via varargin.

%

% \*See GUI Options on GUIDE's Tools menu. Choose "GUI allows only one  
% instance to run (singleton)".

%

% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help EdgeEvaluation1

% Last Modified by GUIDE v2.5 19-Mar-2014 12:20:16

% Begin initialization code - DO NOT EDIT

gui\_Singleton = 1;

```

gui_State = struct('gui_Name',    mfilename, ...
                  'gui_Singleton', gui_Singleton, ...
                  'gui_OpeningFcn', @EdgeEvaluation1_OpeningFcn, ...
                  'gui_OutputFcn', @EdgeEvaluation1_OutputFcn, ...
                  'gui_LayoutFcn', [] , ...
                  'gui_Callback', []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end
if nargin
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT
% --- Executes just before EdgeEvaluation1 is made visible.
function EdgeEvaluation1_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to EdgeEvaluation1 (see VARARGIN)
% Choose default command line output for EdgeEvaluation1
handles.output = hObject;
% Update handles structure
guidata(hObject, handles);
evalin('base','clc;');
axes(handles.axes1);
imshow('BG1.jpg')
axes(handles.axes2);
imshow('BG2.jpg')

```



```

axes(handles.axes3);
imshow('BG3.jpg')
axes(handles.axes4);
imshow('BG4.jpg')
axes(handles.axes5);
imshow('BG5.jpg')
axes(handles.axes6);
imshow('BG6.jpg')
axes(handles.axes7);
imshow('BG7.jpg')

% UIWAIT makes EdgeEvaluation1 wait for user response (see UIRESUME)
% uiwait(handles.figure1);
% --- Outputs from this function are returned to the command line.
function varargout = EdgeEvaluation1_OutputFcn(hObject, eventdata, handles)
% varargout cell array for returning output args (see VARARGOUT);
% hObject handle to figure
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
% Get default command line output from handles structure
varargout{1} = handles.output;

function edit1_Callback(hObject, eventdata, handles)
% hObject handle to edit1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit1 as text
% str2double(get(hObject,'String')) returns contents of edit1 as a double
% --- Executes during object creation, after setting all properties.
function edit1_CreateFcn(hObject, eventdata, handles)
% hObject handle to edit1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB

```

```

% handles empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in browse1.
function browse1_Callback(hObject, eventdata, handles)
% hObject handle to browse1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
[file1,path1] = uigetfile({'*.jpg';*.bmp';*.png'},'Select Image');
set(handles.edit1,'String',file1);

str1 = path1;
str2 = file1;
str3 = [str1 str2];
image1 = imread(str3);
axes(handles.axes1);imshow(image1)
evalin('base','imagein = 0;');
assignin('base','imagein',image1);

% --- Executes on button press in eval1.
function eval1_Callback(hObject, eventdata, handles)
% hObject handle to eval1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
im1 = evalin('base','imagein'); %reading the image
im1g = rgb2gray(im1); %Converting the image to gray scale
[ym,xm] = size(im1g); %Preparing the image size
im1bw = im2bw(im1g);
edg1 = 0&im1bw;
for i = 1:xm-1 %calculating the SD to each four pixels array to determine the edges that exceeds 7 SD

```

```

for j = 1:ym-1
    val = std2([im1g(j,i) im1g(j,i+1);im1g(j+1,i) im1g(j+1,i+1)]);
    if val >= 7
        edg1(j,i) = 1;
    else
        edg1(j,i) = 0;
    end
end
end

edg1f = medfilt2(edg1);           %applying median filter
axes(handles.axes2);imshow(edg1f)
[BW_sobel,thresh_sobel] = edge(im1g,'sobel');
%applying the methods of edge detection
[BW_roberts,thresh_roberts] = edge(im1g,'roberts');
[BW_prewitt,thresh_prewitt] = edge(im1g,'prewitt');
[BW_canny,thresh_canny] = edge(im1g,'canny');
[BW_log,thresh_log] = edge(im1g,'log');
axes(handles.axes3); imshow(BW_sobel);           %displayng the resulted images
axes(handles.axes4); imshow(BW_roberts);
axes(handles.axes5); imshow(BW_prewitt);
axes(handles.axes6); imshow(BW_canny);
axes(handles.axes7); imshow(BW_log);

%pratt for sobel           Applying the Pratt equation by finding the maximum number of edge points
between the

%Ideal and real images
N_Ideal = sum(sum(BW_sobel));           %number of edges points of the ideal image
N_real = sum(sum(edg1f));           %number of edges points of the real image
N_max = max([N_Ideal N_real]);           %maximum number edges points
[y1,x1] = find(edg1f);           %finding the coordinates of edges points
D = 0;

```

```

for i1 = 1:length(x1)          %Calculating the distances between the edge point in the real image
and the corresponding one at the ideal
    test1 = BW_sobel(y1(i1),x1(i1));

    if test1 == 1              %testing the edge point in both images, if match then the distance is
zero else we have find
        D(i1) = 0;            %the nearest point to calculate the distance between it in the ideal image
and the point at the real one
    else
        for n = 1:min(size(BW_sobel)) %this loop to find the nearest edge point in the ideal image
            xp_Im1 = x1(i1);
            yp_Im1 = y1(i1);
            yt1 = yp_Im1-n;
            xt1 = xp_Im1-n;
            yt2 = yp_Im1+n;
            xt2 = xp_Im1+n;
            size1 = size(BW_sobel);
            if yt2 > size1(1)      %to make sure the points are within the image boundary
                yt2 = size1(1);  %
            end
            if yt1 < 1            %
                yt1 = 1;         %
            end
            if xt2 > size1(2)     %
                xt2 = size1(2);  %
            end
            if xt1 < 1
                xt1 = 1;
            end
            part1 = BW_sobel(yt1:yt2,xt1:xt2); %take a part of the ideal image to find the edges points
within it
            [yp_Ref1,xp_Ref1] = find(part1); %find the edges points
            if yp_Ref1 > 0        %edges points found
                Dt = 0;
            end
        end
    end
end

```

```

    for m = 1:length(xp_Ref1)    %calculating the distances to find the nearest edge point
        Dt(m) = (xp_Ref1(m) - xp_Im1)^2 + (yp_Ref1(m) - yp_Im1)^2;
    end

    D(i1) = min(Dt);           %taking the one with the minimum distance
    break;

end

end

end

end

D_total = (D/9) + 1;         %completing the Pratt equation calculation
D_totaln = 1./D_total;
Acc1 = sum(D_totaln);
R_sobel = Acc1/N_max;
set(handles.edit2,'String',R_sobel);

%pratt for roberts
N_Ideal = sum(sum(BW_roberts));
N_real = sum(sum(edg1f));
N_max = max([N_Ideal N_real]);
[y1,x1] = find(edg1f);
D = 0;
for i1 = 1:length(x1)
    test1 = BW_roberts(y1(i1),x1(i1));
    if test1 == 1
        D(i1) = 0;
    else
        for n = 1:min(size(BW_roberts))
            xp_Im1 = x1(i1);
            yp_Im1 = y1(i1);
            yt1 = yp_Im1-n;
            xt1 = xp_Im1-n;
            yt2 = yp_Im1+n;

```

```

xt2 = xp_Im1+n;
size1 = size(BW_roberts);
if yt2 > size1(1)
    yt2 = size1(1);
end
if yt1 < 1
    yt1 = 1;
end
if xt2 > size1(2)
    xt2 = size1(2);
end
if xt1 < 1
    xt1 = 1;
end
part1 = BW_roberts(yt1:yt2,xt1:xt2);
[yp_Ref1,xp_Ref1] = find(part1);
if yp_Ref1 > 0
    Dt = 0;
    for m = 1:length(xp_Ref1)
        Dt(m) = (xp_Ref1(m) - xp_Im1)^2 + (yp_Ref1(m) - yp_Im1)^2;
    end
    D(i1) = min(Dt);
    break;
end
end
end
end
D_total = (D/9) + 1;
D_totaln = 1./D_total;
Acc1 = sum(D_totaln);
R_roberts = Acc1/N_max;

```

```

set(handles.edit3,'String',R_roberts);
%pratt for prewitt
N_Ideal = sum(sum(BW_prewitt));
N_real = sum(sum(edg1f));
N_max = max([N_Ideal N_real]);
[y1,x1] = find(edg1f);
D = 0;
for i1 = 1:length(x1)
    test1 = BW_prewitt(y1(i1),x1(i1));
    if test1 == 1
        D(i1) = 0;
    else
        for n = 1:min(size(BW_prewitt))
            xp_Im1 = x1(i1);
            yp_Im1 = y1(i1);
            yt1 = yp_Im1-n;
            xt1 = xp_Im1-n;
            yt2 = yp_Im1+n;
            xt2 = xp_Im1+n;
            size1 = size(BW_prewitt);
            if yt2 > size1(1)
                yt2 = size1(1);
            end
            if yt1 < 1
                yt1 = 1;
            end
            if xt2 > size1(2)
                xt2 = size1(2);
            end
            if xt1 < 1
                xt1 = 1;
            end
        end
    end
end

```

```

part1 = BW_prewitt(yt1:yt2,xt1:xt2);
[yp_Ref1,xp_Ref1] = find(part1);
if yp_Ref1 > 0
    Dt = 0;
    for m = 1:length(xp_Ref1)
        Dt(m) = (xp_Ref1(m) - xp_Im1)^2 + (yp_Ref1(m) - yp_Im1)^2;
    end
    D(i1) = min(Dt);
    break;
end
end
end
end
D_total = (D/9) + 1;
D_totaln = 1./D_total;
Acc1 = sum(D_totaln);
R_prewitt = Acc1/N_max;
set(handles.edit4,'String',R_prewitt);
%pratt for canny
N_Ideal = sum(sum(BW_canny));
N_real = sum(sum(edg1f));
N_max = max([N_Ideal N_real]);
[y1,x1] = find(edg1f);
D = 0;
for i1 = 1:length(x1)
    test1 = BW_canny(y1(i1),x1(i1));
    if test1 == 1
        D(i1) = 0;
    else
        for n = 1:min(size(BW_canny))

```



```

xp_Im1 = x1(i1);
yp_Im1 = y1(i1);
yt1 = yp_Im1-n;
xt1 = xp_Im1-n;
yt2 = yp_Im1+n;
xt2 = xp_Im1+n;
size1 = size(BW_canny);
if yt2 > size1(1)
    yt2 = size1(1);
end
if yt1 < 1
    yt1 = 1;
end
if xt2 > size1(2)
    xt2 = size1(2);
end
if xt1 < 1
    xt1 = 1;
end
part1 = BW_canny(yt1:yt2,xt1:xt2);
[yp_Ref1,xp_Ref1] = find(part1);
if yp_Ref1 > 0
    Dt = 0;
    for m = 1:length(xp_Ref1)
        Dt(m) = (xp_Ref1(m) - xp_Im1)^2 + (yp_Ref1(m) - yp_Im1)^2;
    end
    D(i1) = min(Dt);

```

```

        break;
    end
end
end
end
end
D_total = (D/9) + 1;
D_totaln = 1./D_total;
Acc1 = sum(D_totaln);
R_canny = Acc1/N_max;
set(handles.edit5,'String',R_canny);

%pratt for Laplacian of Gaussian

N_Ideal = sum(sum(BW_log));
N_real = sum(sum(edg1f));
N_max = max([N_Ideal N_real]);
[y1,x1] = find(edg1f);
D = 0;
for i1 = 1:length(x1)
    test1 = BW_log(y1(i1),x1(i1));
    if test1 == 1
        D(i1) = 0;
    else
        for n = 1:min(size(BW_log))
            xp_Im1 = x1(i1);
            yp_Im1 = y1(i1);
            yt1 = yp_Im1-n;
            xt1 = xp_Im1-n;
            yt2 = yp_Im1+n;
            xt2 = xp_Im1+n;
            size1 = size(BW_log);

```

```

if yt2 > size1(1)
    yt2 = size1(1);
end
if yt1 < 1
    yt1 = 1;
end
if xt2 > size1(2)
    xt2 = size1(2);
end
if xt1 < 1
    xt1 = 1;
end
part1 = BW_log(yt1:yt2,xt1:xt2);
[yp_Ref1,xp_Ref1] = find(part1);
if yp_Ref1 > 0
    Dt = 0;
    for m = 1:length(xp_Ref1)
        Dt(m) = (xp_Ref1(m) - xp_Im1)^2 + (yp_Ref1(m) - yp_Im1)^2;
    end
    D(i1) = min(Dt);
    break;
end
end
end
D_total = (D/9) + 1;
D_totaln = 1./D_total;
Acc1 = sum(D_totaln);
R_log = Acc1/N_max;
set(handles.edit6,'String',R_log);

```

```

function edit2_Callback(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit2 as text
%         str2double(get(hObject,'String')) returns contents of edit2 as a double
% --- Executes during object creation, after setting all properties.

function edit2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit3_Callback(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit3 as text
%         str2double(get(hObject,'String')) returns contents of edit3 as a double
% --- Executes during object creation, after setting all properties.

function edit3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))

```

```

    set(hObject,'BackgroundColor','white');
end
function edit4_Callback(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit4 as text
%    str2double(get(hObject,'String')) returns contents of edit4 as a double
% --- Executes during object creation, after setting all properties.
function edit4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%    See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function edit5_Callback(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit5 as text
%    str2double(get(hObject,'String')) returns contents of edit5 as a double
% --- Executes during object creation, after setting all properties.
function edit5_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%    See ISPC and COMPUTER.

```

```

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit6_Callback(hObject, eventdata, handles)
% hObject    handle to edit6 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of edit6 as text
%         str2double(get(hObject,'String')) returns contents of edit6 as a double
% --- Executes during object creation, after setting all properties.

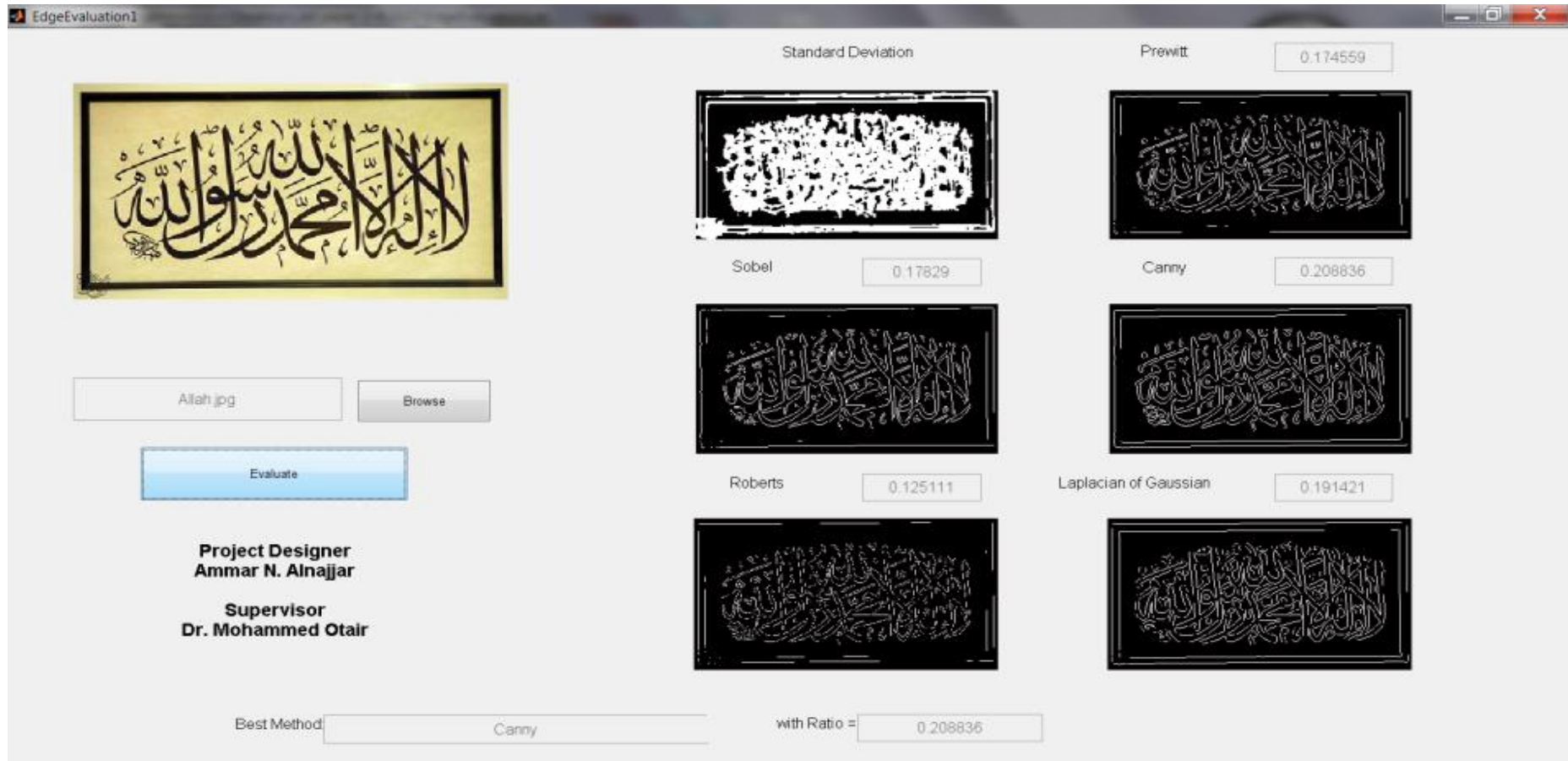
function edit6_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit6 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```


## APPENDIX-B (GUI RESULTED WORKS)

Set of images that have been applied by the experience.



Allah.jp

EdgeEvaluation1




mohammed.bmp    Browse

Evaluate


Project Designer  
Ammar N. Alnajjar

Supervisor  
Dr. Mohammed Otair


Standard Deviation




Sobel    0.153394




Roberts    0.148064




Prewitt    0.15021



Canny    0.190245



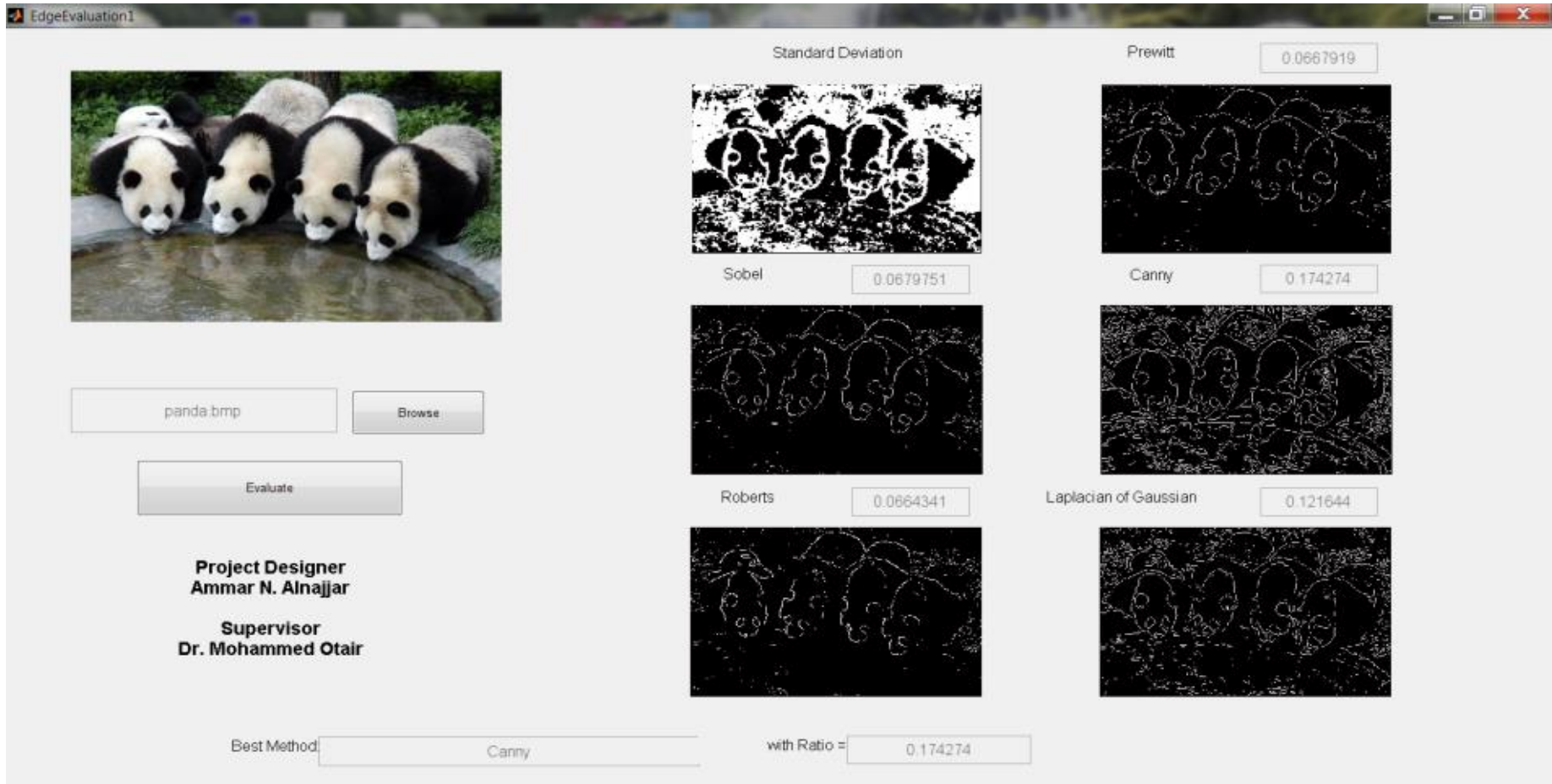
Laplacian of Gaussian    0.172359



Best Method: Canny    with Ratio = 0.190245


Mohammed.bmp





Panda.bmp

EdgeEvaluation1



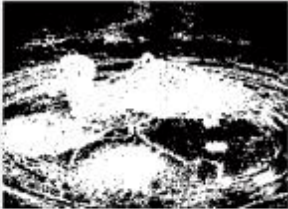
bird.jpg    Browse

Evaluate


**Project Designer**  
Ammar N. Alnajjar

**Supervisor**  
Dr. Mohammed Otair


Standard Deviation




Sobel    0.0947141



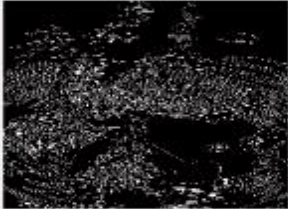
Roberts    0.0603244




Prewitt    0.093749



Canny    0.193892




Laplacian of Gaussian    0.156261



Best Method: Canny    with Ratio = 0.193892

Bird.jpg

EdgeEvaluation1



bird1.bmp    Browse

Evaluate

**Project Designer**  
Ammar N. Alnajjar

**Supervisor**  
Dr. Mohammed Otair

Standard Deviation

Sobel    0.109591

Roberts    0.0914627

Prewitt    0.10865


Canny    0.235137

Laplacian of Gaussian    0.185275

Best Method: Canny    with Ratio = 0.235137

Bird1.bmp

EdgeEvaluation1




golden.jpg    Browse

Evaluate


**Project Designer**  
Ammar N. Alnajjar

**Supervisor**  
Dr. Mohammed Otair


Standard Deviation




Sobel    0.0697473



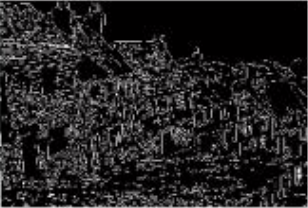
Roberts    0.0392227




Prewitt    0.0690545



Canny    0.166487




Laplacian of Gaussian    0.12695



Best Method:     with Ratio =

Golden.jpg

EdgeEvaluation1




AAU.bmp


**Project Designer**  
Ammar N. Alnajjar

**Supervisor**  
Dr. Mohammed Otair


Standard Deviation




Sobel




Roberts




Prewitt



Canny

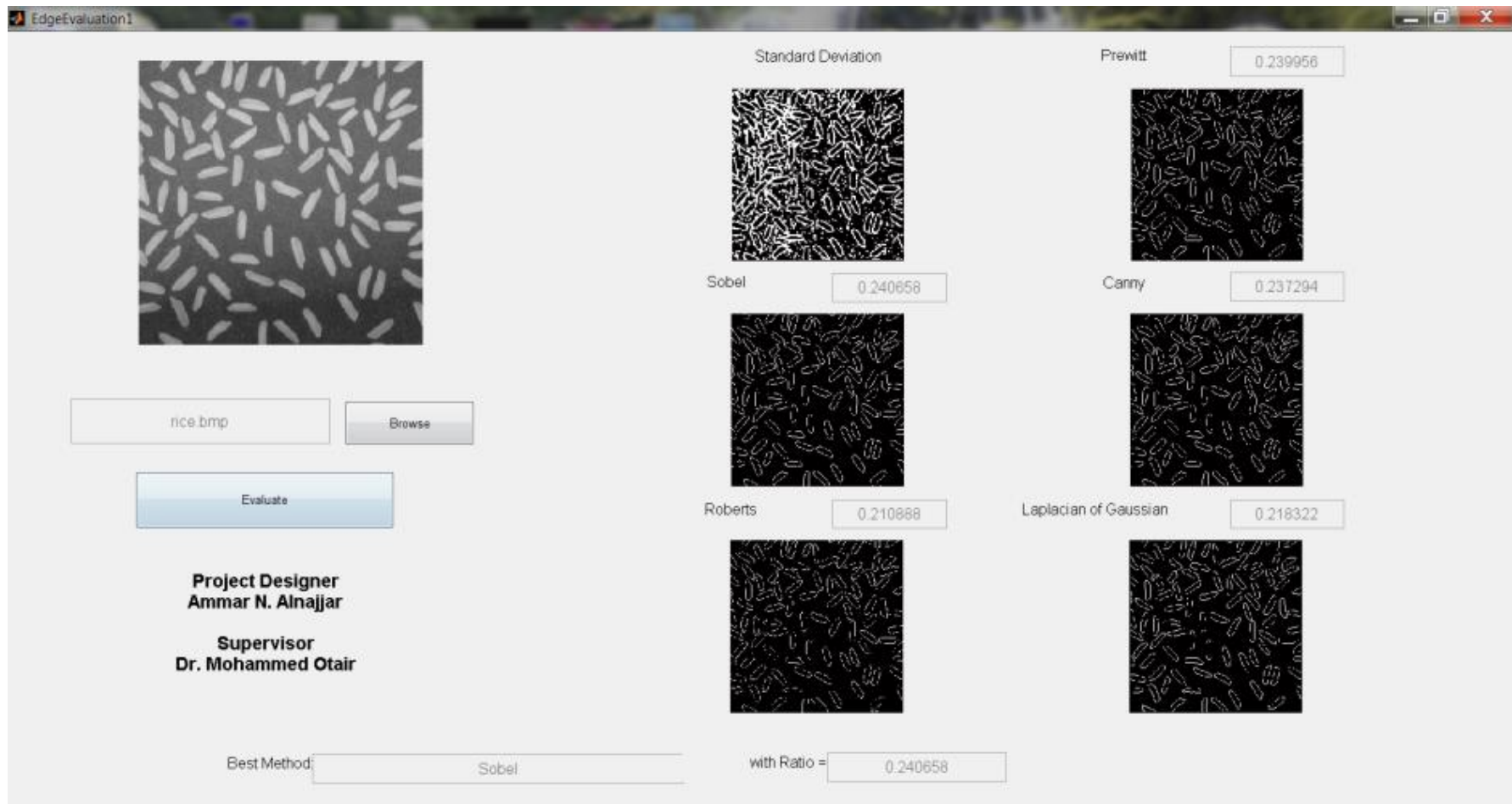


Laplacian of Gaussian

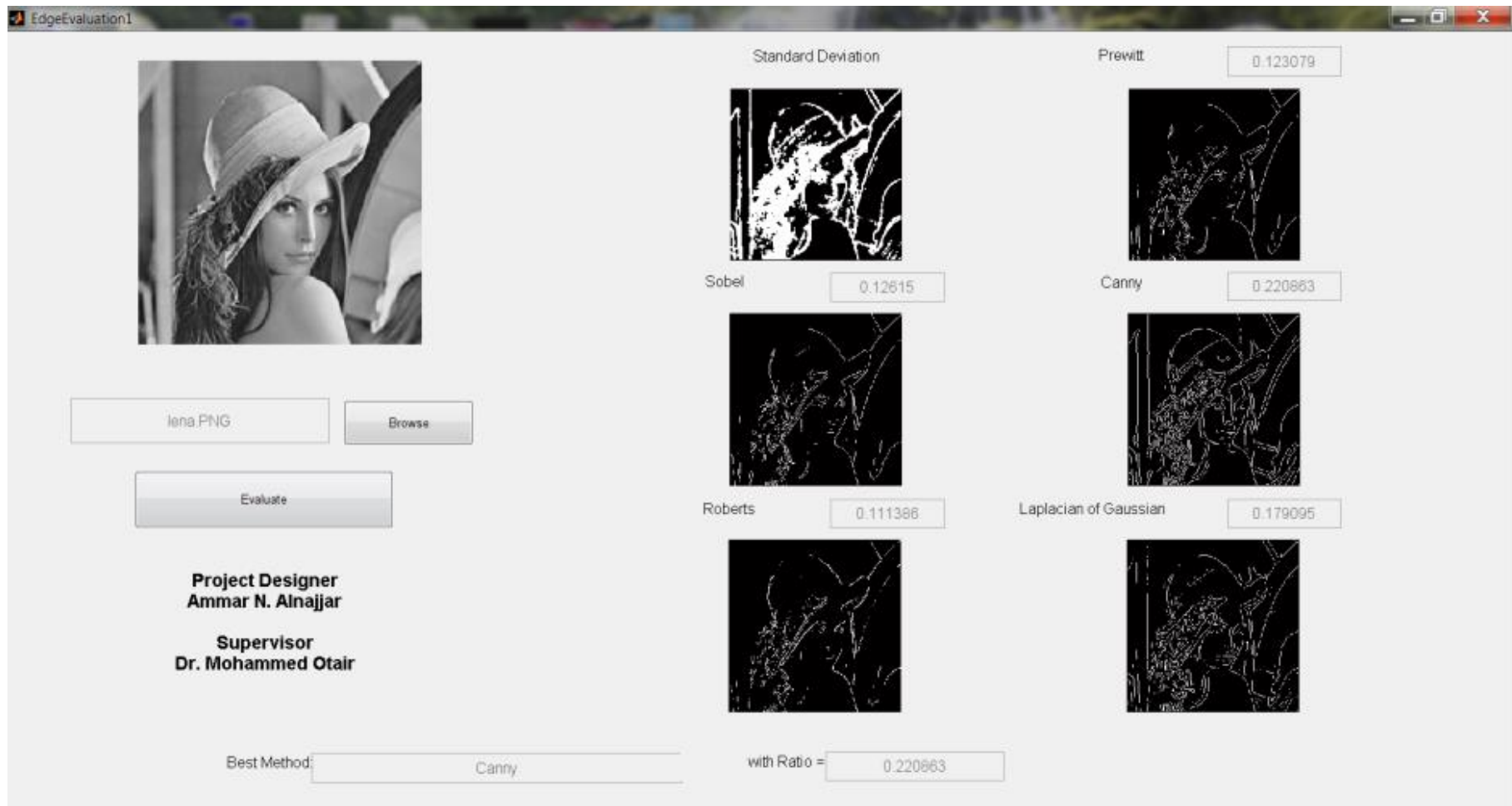


Best Method:  with Ratio =

AAU.bmp




Rice.bmp



Lena.png

EdgeEvaluation1



Tala.jpg    Browse

Evaluate

**Project Designer**  
Ammar N. Alnajjar

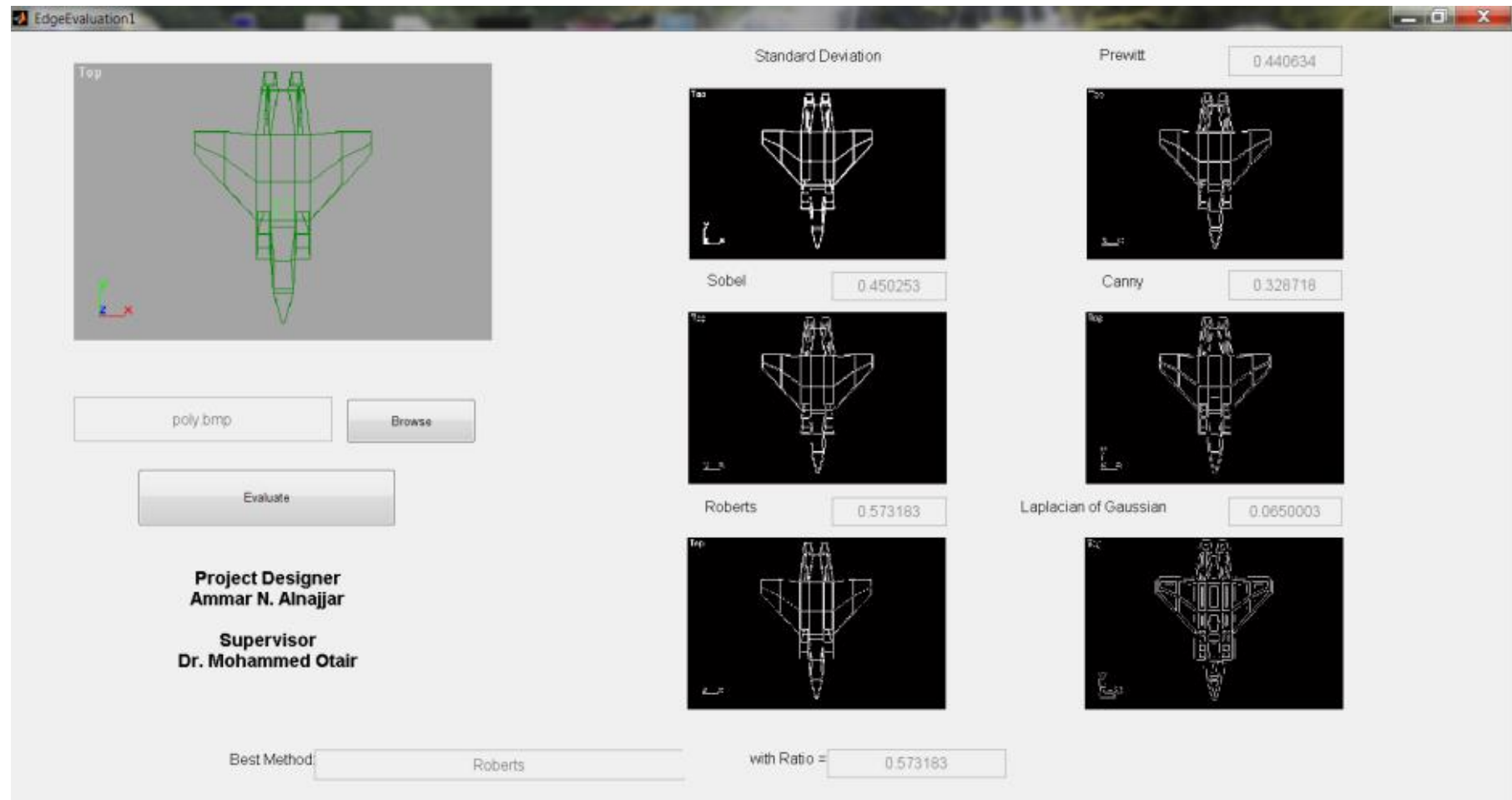
**Supervisor**  
Dr. Mohammed Otair

Method	Ratio
Standard Deviation	
Sobel	0.146269
Roberts	0.176838
Prewitt	0.146547
Canny	0.0642777
Laplacian of Gaussian	0.0665598

Best Method:  with Ratio =


Tala.jpg





Poly.bmp

EdgeEvaluation1



tools.jpg    Browse


Evaluate

**Project Designer**  
Ammar N. Alhajjar


**Supervisor**  
Dr. Mohammed Otair

Standard Deviation


Prewitt    0.152555




Sobel    0.153142




Canny    0.200829



Roberts    0.180906



Laplacian of Gaussian    0.177041



Best Method: Canny    with Ratio = 0.200829

Tools.jpg

EdgeEvaluation1

Screws.jpg    Browse

Evaluate

Project Designer  
Ammar N. Alnajjar

Supervisor  
Dr. Mohammed Otair

Standard Deviation    Prewitt    0.186316

Sobel    0.1864    Canny    0.26541

Roberts    0.171126    Laplacian of Gaussian    0.227509

Best Method: Canny    with Ratio = 0.26541

Screws.jpg

EdgeEvaluation1

Standard Deviation: 0.142345

Prewitt: 0.142345

Sobel: 0.142322

Canny: 0.238367

Roberts: 0.130281

Laplacian of Gaussian: 0.178761

Best Method: Canny with Ratio = 0.238367

Project Designer  
Ammar N. Alnajjar

Supervisor  
Dr. Mohammed Otair

Kitchen.jpg

EdgeEvaluation1

Project Designer  
Ammar N. Alnajjar


Supervisor  
Dr. Mohammed Otair

Best Method: Laplacian of Gaussian with Ratio = 0.227264

Method	Ratio
Standard Deviation	0.124369
Prewitt	0.118773
Sobel	0.124369
Canny	0.107381
Roberts	0.0603416
Laplacian of Gaussian	0.227264

Pout.tif

EdgeEvaluation1




zag.jpg    Browse

Evaluate


**Project Designer**  
Ammar N. Alnajjar

**Supervisor**  
Dr. Mohammed Otair


Standard Deviation




Sobel    0.0600204




Roberts    0.0367407




Prewitt    0.0532636



Canny    0.154081




Laplacian of Gaussian    0.148387



Best Method:     with Ratio =

Zag.jpg

EdgeEvaluation1




Shadow.jpg    Browse

Evaluate


**Project Designer**  
Ammar N. Alnajjar

**Supervisor**  
Dr. Mohammed Otair


Standard Deviation



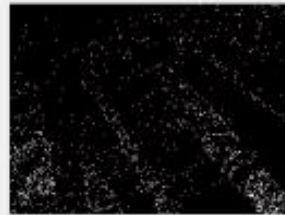
Sobel    0.189466



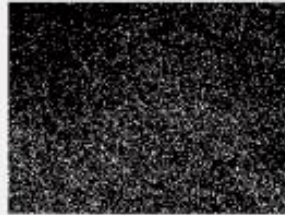
Roberts    0.13647




Prewitt    0.188588



Canny    0.268881




Laplacian of Gaussian    0.207059



Best Method:  with Ratio =

Shadow.jpg

EdgeEvaluation1




flamingo.bmp    Browse

Evaluate


**Project Designer**  
Ammar N. Alnajjar

**Supervisor**  
Dr. Mohammed Otair


Standard Deviation




Sobel    0.148191




Roberts    0.125919




Prewitt    0.145058



Canny    0.224618



Laplacian of Gaussian    0.190971



Best Method:  with Ratio =

Flamingo.jpg