Detection of Image Edges: A gradient based approach

Submitted as a partial fulfillment of Bachelor of Technology in Computer Science & Engineering
of
Maulana Abul Kalam Azad University of Technology
(Formerly known as West Bengal University of Technology)

Project Report

Submitted by

Name of Students	**University Roll No.**
Hriday Saha	**11600114029**
Baljeet Singh	**11600114023**
Bharat Bhushan Gupta	**11600114025**

Under the supervision of

Mr. Puspen Lahiri

Assistant Professor, Department of CSE

Department of Computer Science & Engineering,
MCKV Institute of Engineering
243, G.T.Road(N)
Liluah, Howrah - 711204

Department of Computer Science & Engineering
MCKV Institute of Engineering
243, G. T.Road (N),
Liluah, Howrah-711204

CERTIFICATE OF RECOMMENDATION

I hereby recommend that the thesis prepared under my supervision by Hriday Saha, Baljeet Singh, and Bharat Bhushan Gupta entitled Detection of Image Edges: A gradient based approach, be accepted in partial fulfillment of the requirements for the degree of Bachelor of Technology in Computer Science & Engineering Department.

Prof. S. S.Thakur
Head of the Department,
Computer Science & Engineering Department.
MCKV Institute of Engineering,Howrah

Project guide
Mr. Puspen Lahiri
Assistant Professor
Department of CSE, MCKVIE

MCKV Institute of Engineering
243, G. T. Road (N), Liluah
Howrah-711204

Affiliated to
Maulana Abul Kalam Azad University of Technology
(Formerly known as West Bengal University of Technology)

CERTIFICATE

This is to certify that the project entitled Detection of Image Edges: A gradient based approach and submitted by

Name of students	University Roll No.
Hriday Saha	11600114029
Baljeet Singh	11600114023
Bharat Bhushan Gupta	11600114025

has been carried out under the guidance of myself following the rules and regulations of the degree of Bachelor of Technology in Computer Science & Engineering of **Maulana Abul Kalam Azad University of Technology (**Formerly West Bengal University of Technology).

_____

(Signature of the project guide)
Mr. Puspen Lahiri
Assistant Professor
Department of CSE

1. _____
2. _____
3. _____
4. _____
5. _____

MCKV Institute of Engineering
243, G. T.Road (N), Liluah
Howrah-711204

Affiliated to
Maulana Abul Kalam Azad University of Technology
(Formerly known as West Bengal University of Technology)

CERTIFICATE OF APPROVAL

(B.Tech Degree in Computer Science & Engineering)

This project report is hereby approved as a creditable study of an engineering subject carried out and presented in a manner satisfactory to warrant its acceptance as a prerequisite to the degree for which it has been submitted. It is to be understood that by this approval, the undersigned do not necessarily endorse or approve any statement made, opinion expressed and conclusion drawn therein but approve the project report only for the purpose for which it has been submitted

COMMITTEE ON FINAL
EXAMINATION FOR
EVALUATION OF
PROJECT REPORT

1. _____
2. _____
3. _____
4. _____
5. _____

Acknowledgement

We take this opportunity to express our profound gratitude and deep regards to our faculty Mr Puspen Lahiri, Assistant Professor, CSE Department, for his exemplary guidance, monitoring and constant encouragement throughout the course of this project. The blessing, help and guidance given by him time to time shall carry us a long way in the journey of this project. We are obliged to our project team members for the valuable cooperation, coordination and information provided by them in their respective fields. We are grateful for their cooperation during the period of our assignment.

Our sincere thanks to our project coordinator Mr. Abhisek Saha, Assistant Professor, CSE Department, and Prof. (Dr.) S. S. Thakur (HOD, Department of Computer Science & Engineering, MCKV Institute of Engineering) for their guidance and encouragement that helped us to bag an opportunity of working at a research work oriented project.

CONTENTS

1. Abstract

Edge detection includes a variety of mathematical methods that aim at identifying points in a digital image at which the image brightness changes sharply, or more formally, has discontinuities (edges). A traditional approach to do this is to find the gradient of intensity levels, in X and Y directions, find its direction, and finally find the edge direction viz. orthogonal to the gradient vector. Operators involved in the approach are Prewitt and SOBEL operators. Another approach, called Canny edge detection, requires smoothing of an image (Gaussian filter), which is an improvement of the traditional. Considering traditional gradient based approach to detect image edges, we present different but simple edge detection algorithms some of which uses local maxima or applies mathematical statistics. The proposed algorithms applies suitable threshold, using which the pixels may be considered as either edge pixels or not. Some of these algorithms determine a fixed threshold, while others determine adaptive threshold. The best edge detection method is selected based on the comparison made between our proposed methods.

2. Introduction

2.1 Edge Detection

In a grey-level picture containing homogeneous (i.e., un-textured) objects, an edge is the boundary between two regions of different constant grey level [1]. Edges are significant local changes of intensity in an image. Edges typically occur on the boundary between two different regions in an image.

Edge detection is an image processing technique for finding the boundaries of objects within images [2]. It works by detecting discontinuities in brightness. Edge detection helps in optimizing network bandwidth and it is needed to keep track of data flowing in and out of the network.

The goals of Edge Detection are as follows:

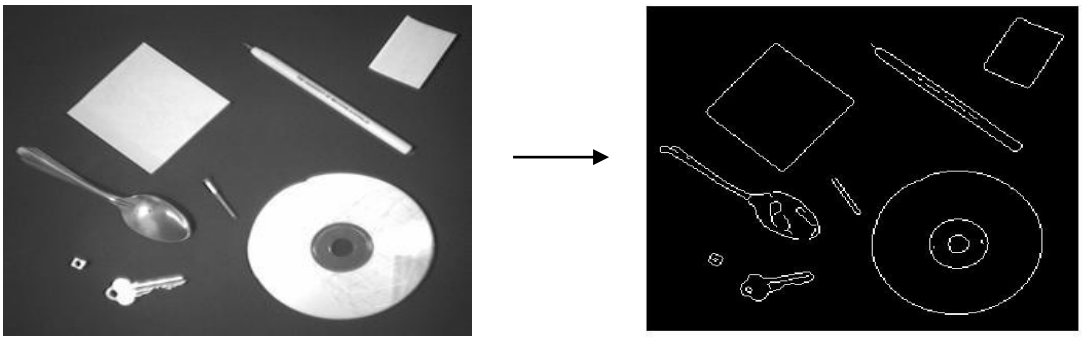
a. Produce a line drawing of a scene from an image of that scene.
b. Important features can be extracted from the edges of an image (e.g., corners, lines, curves).
c. These features, mentioned in **(b)** are used by higher-level computer vision algorithms (e.g., recognition).

Edge detection is widely used in the following area of image processing:

a. Image Segmentation
b. Feature Extraction
c. Computer Vision
d. Machine Vision
e. Image Matching etc.

So, we can conclude as mentioning Edge detection is a process of locating an edge of an image and detection of edges in an image is a very important step towards understanding image features. Edges consist of meaningful features and contain significant information. It significantly reduces the image size and filters out information that may be regarded as less relevant, thus preserving the important structural properties of an image. Most images contain some amount of redundancies that can sometimes be removed when edges are detected and replaced during reconstruction. This is where edge detection comes into play. Also, edge

detection is one of the ways of making images not take up too much space in the computer memory. Since edges often occur at image locations representing object boundaries, edge detection is extensively used in image segmentation when images are divided into areas corresponding to different objects.

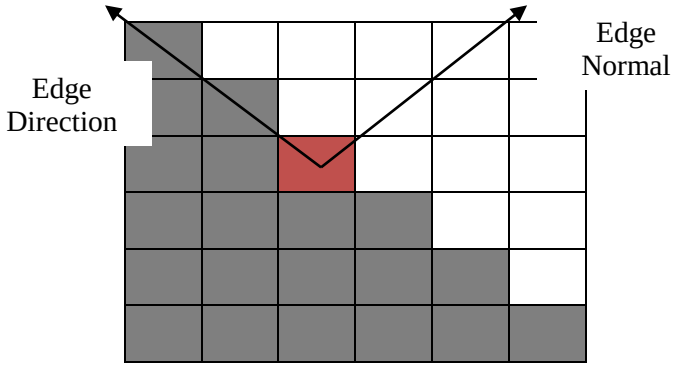


Input Image Image Edges

2.1.1 Example: **Edge Detection**

2.2 Edge Descriptors

a. **Edge normal**: unit vector in the direction of maximum intensity change.
b. **Edge direction**: unit vector to perpendicular to the edge normal.
c. **Edge position or center**: the image position at which the edge is located.
d. **Edge strength**: related to the local image contrast along the normal.



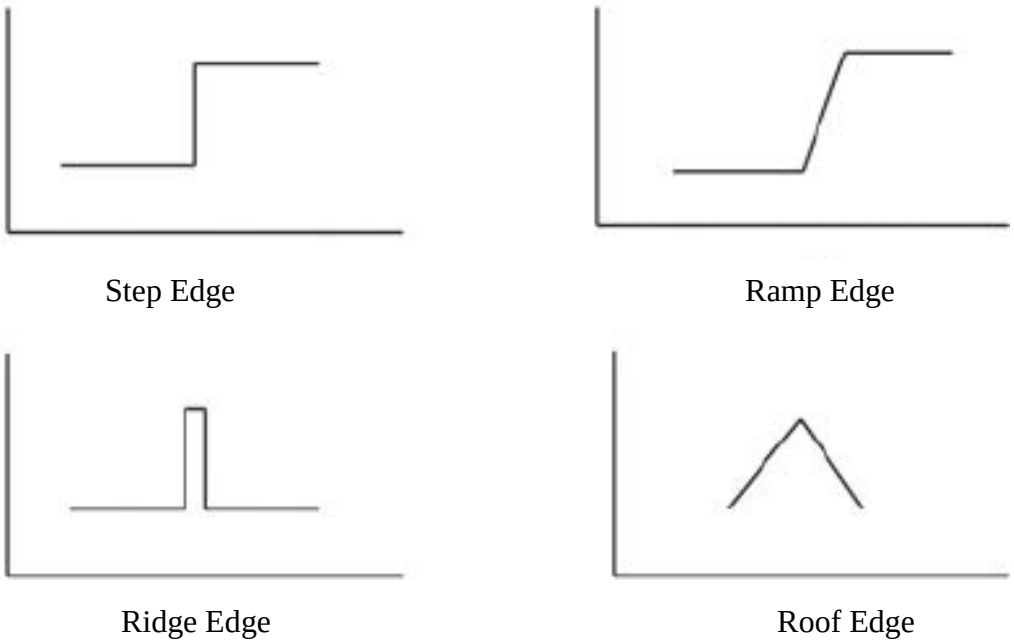
2.2.1 Edge Descriptors

2.3 Types of Edges

Edges can be modeled according to their intensity profiles.

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

b. **Ramp edge**: a step edge where the intensity change is not instantaneous but occur over a finite distance.
c. **Ridge edge**: the image intensity abruptly changes value but then returns to the starting value within some short distance (generated usually by lines).
d. **Roof edge**: a ridge edge where the intensity change is not instantaneous but occur over a finite distance (generated usually by the intersection of surfaces).



2.3.1 Edge Types

2.4 Steps of Edge Detection

The following steps are generally followed for detection of image edges:

a. **Smoothing**: suppress as much noise as possible, without destroying the true edges.
b. **Enhancement**: apply a filter to enhance the quality of the edges in the image (sharpening).
c. **Detection**: determine which edge pixels should be discarded as noise and which should be retained (usually, thresholding provides the criterion used for detection).
d. **Localization**: determine the exact location of an edge (*sub-pixel* resolution might be required for some applications, that is, estimate the location of an edge to better than the spacing between pixels). Edge thinning and linking are usually required in this step.

3. Hardware and Software Requirement

- No. of PCs required: 01PC
- Configuration: 2 GB (or more) RAM, Intel i3 (or above) processor
- Any other device(s) required: NIL
- OS: Windows 7/8/8.1/10
- Software required: MATLAB R2017

4. Edge Detection using derivatives and gradients:

4.1 Detection using derivatives

Introduction:

Calculus describes changes of continuous functions using *derivatives*. An image is a 2D function, so operators describing edges are expressed using *partial derivatives*. Points which lie on an edge can be detected by:

a. Detecting local maxima or minima of the first derivative
b. Detecting the zero-crossing of the second derivative

Process:

To compute the derivative of a signal, we approximate the derivative by finite differences:

Computing the 1st derivative:

$$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h} \approx f(x+1) - f(x)(h=1)$$

mask: [-1 1]

Computing the 2nd derivative:

$$f''(x) = \lim_{h \to 0} \frac{f'^{(x+h)} - f'(x)}{h} \approx f'^{(x+1)} - f'^{(x)} = f(x+2) - 2f(x+1) + f(x)(h=1)$$

This approximation is **centered** about $x + 1$; by replacing $x + 1$ by x we obtain:

$$f''(x) \approx f(x+1) - 2f(x) + f(x-1)$$

mask: [1 -2 1]

4.2.1 Detection using gradients

Introduction: Gradient with magnitude and directions are the main concept here.

Process:

The gradient is a vector which has certain magnitude and direction:

$$\nabla f = \begin{pmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{pmatrix}$$

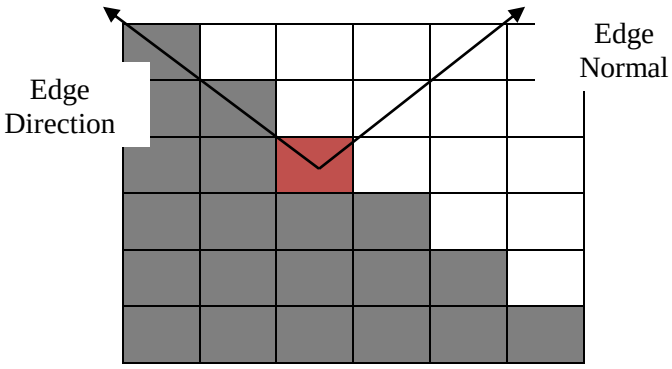
$$magn\,(\nabla f) = \sqrt{(\frac{\partial f}{\partial x})^2 + (\frac{\partial f}{\partial y})^2} \qquad dir\,(\nabla f) = tan^{-}(\frac{M_y}{M_x})$$

To save computations, the magnitude of gradient is usually approximated by:

$magn\,(\nabla f) = |M_x| + |M_y|$

Properties of the gradient

a. The magnitude of gradient provides information about the strength of the edge.
b. The direction of gradient is always perpendicular to the direction of the edge (the edge direction is rotated with respect to the gradient direction by -90 degrees).



4.2.1.1 Gradient Properties

4.2.2 Estimating the gradient with finite differences

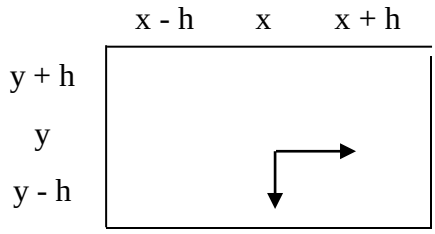
Process:

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

The gradient can be approximated by *finite differences*:

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

Using pixel-coordinate notation (**remember:** *j* corresponds to the *x* direction and *i* to the negative *y* direction):



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

$$\frac{\partial f}{\partial y} = f(i-1, j) - f(i,j), or \frac{\partial f}{\partial y} = f(i,j) - f(i+1,j)$$

4.3 Detection using Roberts , Prewiit and Sobel Operator

4. 3. 1. Detection using Roberts Operator

Description: The Roberts operator performs a simple, quick to compute, 2-D spatial gradient measurement on an image. It thus highlights regions of high spatial gradient which often correspond to edges. In its most common usage, the input to the operator is a grayscale image, as is the output. Pixel values at each point in the output represent the estimated absolute magnitude of the spatial gradient of the input image at that point.

Working Principle: In theory, the operator consists of a pair of 2×2 convolution masks as shown in Figure below. One mask is simply the other rotated by 90°.

+1	0
0	-1

G_x

0	+1
-1	0

G_y

4.3.1.1 Roberts Cross convolution masks

These masks are designed to respond maximally to edges running at 45° to the pixel grid, one mask for each of the two perpendicular orientations. The masks can be applied separately to the input image, to produce separate measurements of the gradient component in each orientation (call these G_x and G_y). These can then be combined together to find the absolute magnitude of the gradient at each point and the orientation of that gradient. The gradient magnitude is given by:

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

Although typically, an approximate magnitude is computed using:

$|G| = |G_x| + |G_y|$, which is much faster to compute.

The angle of orientation of the edge giving rise to the spatial gradient (relative to the pixel grid orientation) is given by:

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

In this case, orientation 0 is taken to mean that the direction of maximum contrast from black to white runs from left to right on the image, and other angles are measured anticlockwise from this.

Often, the absolute magnitude is the only output the user sees --- the two components of the gradient are conveniently computed and added in a single pass over the input image using the pseudo-convolution operator shown in Figure below:

P_1	P_2
P_3	P_4

4.3.1.2 Pseudo-Convolution masks used to quickly compute approximate gradient magnitude

Using this mask the approximate magnitude is given by: $|G| = |P_1 - P_4| + |P_2 - P_3|$

4. 3. 2. Detection using Prewiit Operator

Description: The Prewitt operator is used in image processing, particularly within edge detection algorithms. Technically, it is a discrete differentiation operator, computing an approximation of the gradient of the image intensity function. At each point in the image, the result of the Prewitt operator is either the corresponding gradient vector or the norm of this vector. The Prewitt operator is based on convolving the image with a small, separable, and integer valued filter in horizontal and vertical directions and is therefore relatively inexpensive in terms of computations. On the other hand, the gradient approximation which it produces is relatively crude, in particular for high frequency variations in the image. The Prewitt operator was developed by Judith M. S. Prewitt.

Working Principle: Mathematically, the operator uses two 3×3 kernels which are convolved with the original image to calculate approximations of the derivatives - one for horizontal changes, and one for vertical. If we define $\boldsymbol{A}$ as the source image, and $G_x$and G_y are two images which at each point contain the horizontal and vertical derivative approximations, the latter are computed as:

$$G_x = \begin{bmatrix} -1 & 0 & +1 \\ -1 & 0 & +1 \\ -1 & 0 & +1 \end{bmatrix} * A \qquad \text{and} \qquad G_y = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ +1 & +1 & +1 \end{bmatrix} * A$$

In the above * denotes the 1-dimensional convolution operation.

Since the Prewitt kernels can be decomposed as the products of an averaging and a differentiation kernel, they compute the gradient with smoothing. Therefore, it is a separable filter. For example, G_x can be written as:

$$\begin{bmatrix} -1 & 0 & +1 \\ -1 & 0 & +1 \\ -1 & 0 & +1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} [-1 \quad 0 \quad +1]$$

The *x*-coordinate is defined here as increasing in the "right"-direction, and the *y*-coordinate is defined as increasing in the "down"-direction. At each point in the image, the resulting gradient approximations can be combined to give the gradient magnitude, using: $G = \sqrt{{G_x}^2 + {G_y}^2}$

Using this information, we can also calculate the gradient's direction: $\theta = atan2(G_x, G_y)$, where θ , for example, is 0 for a vertical edge is darker on the right side.

4. 3. 3. Detection using Sobel Operator

Description: The Sobel operator performs a 2-D spatial gradient measurement on an image and so emphasizes regions of high spatial gradient that correspond to edges. Typically it is used to find the approximate absolute gradient magnitude at each point in an input grayscale image. Technically, it is a discrete differentiation operator, computing an approximation of the gradient of the image intensity function. At each point in the image, the result of the Sobel operator is either the corresponding gradient vector or the norm of this vector. The Sobel operator is based on convolving the image with a small, separable, and integer valued filter in horizontal and vertical direction and is therefore relatively inexpensive in terms of computations. On the other

hand, the gradient approximation that it produces is relatively crude, in particular for high frequency variations in the image.

Working Principle: In theory at least, the operator consists of a pair of 3×3 convolution masks as shown in Figure below. One mask is simply the other rotated by 90°. This is very similar to the Roberts Cross operator.

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

G_x

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

G_y

4.3.3.1 Sobel convolution masks

These masks are designed to respond maximally to edges running vertically and horizontally relative to the pixel grid, one mask for each of the two perpendicular orientations. The masks can be applied separately to the input image, to produce separate measurements of the gradient component in each orientation (call these G_x and G_y). These can then be combined together to find the absolute magnitude of the gradient at each point and the orientation of that gradient. The gradient magnitude is given by:

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

Although typically, an approximate magnitude is computed using:

$|G| = |G_x| + |G_y|$, which is much faster to compute

The angle of orientation of the edge giving rise to the spatial gradient (relative to the pixel grid orientation) is given by:

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

In this case, orientation 0 is taken to mean that the direction of maximum contrast from black to white runs from left to right on the image, and other angles are measured anticlockwise from this. Often, this absolute magnitude is the only output the user sees --- the two components of the gradient are conveniently computed and added in a single pass over the input image using the pseudo-convolution operator shown in Figure below.

P_1	P_2	P_3
P_4	P_5	P_6
P_7	P_8	P_9

4.3.3.2 Pseudo-Convolution masks used to quickly compute approximate gradient magnitude

Using this mask the approximate magnitude is given by: $|G| = |(P_1 + 2P_2 + P_3) - (P_7 + 2P_8 + P_9)| + |(P_3 + 2P_6 + P_9) - (P_1 + 2P_4 + P_7)|$

5. Literature Survey

We have considerd the following literature survey for our work, especially for Thresholding:

Title, Author/s, and Publishing Information	Details
Title: "A Survey on Image Segmentation" Author/s: K.S. Fu, J.K. Mui Published at: Pattern Recognition - Vol 13. pp 3 16 / Pergamon Press Ltd, 1981 Printed in Great Britain / © Pattern Recognition Society.	Survey of the techniques to detect edges, which include edge element extraction methods like high-emphasis spatial filtering , in which the image is passed through high-pass filter by Fourier transformation and linear spatial filtering and inverse transform, and gradient operators as proposed by Roberts, SOBEL, Prewitt, Kirsch etc. , and edge element combination methods like heuristic search, proposed by Martelli in which a state space search is performed on the pixels, and dynamic programming, proposed by Montanari in which the best path of the edges is selected after all the enumerations. Heuristic search and dynamic programming has been considered a very complex, yet efficient, method used in artificial intelligence.
Title: "A Spatial Thresholding method for Image Segmentation" Author/s: K.V. Mardia, T.J. Hainsworth Published at: IEEE Transactions on Pattern Analysis and Machine Learning, Vol. 10, 6 November, 1988.	Suggested thresholding for image segmentation, which is defined as the selection of a certain value, beyond which the pixels are accepted and plotted. Various thresholding techniques have been proposed, including non-spatial Bayes' thresholding, which assigns a gray level to the population of maximum likelihood, different iterative selection methods suggested by Ridler and

	Calvard, in which the averages of mean gray levels of two image segments until convergence is reached, and Lloyd, which modifies the previous concept by adding a logarithmic function of ration of number of pixels of two segments, and spatial thresholding methods like local mean thresholding, in which the threshold is calculated as the sum of average of mean of two segments and the logarithmic function of ratio of probability of occurrence.
Title: "A Survey of Thresholding Techniques" Author/s: P.K. Sahoo, S.Sooltani, A.K.C. Wong Published at: Computer Vision, Graphics, And Image Processing 41, 233-260 (1988).	Suggested numerous thresholding techniques for edge detection and/or image segmentation, which include global and local thresholding techniques. Global techniques can be point-dependent like p-Tile method, a traditional method of finding the threshold as the highest gray level which maps pixels of minimum occurring probability, mode method, which calculates mode from a histogram of gray levels, Ostu method, which divides an image into two classes, find their respective variances and minimizing criterion functions, histogram concavity analysis method, which finds the threshold at the "shoulder" of the histogram, entropy method, which calculates the threshold by observing the regions with minimum randomness, moment preserving method, in which values are calculated such that the moment of pixels of the image are preserved in the output image and minimum error method, in which the gray-level values are viewed as

	probabilistic density functions of mixture population comprising of object and background pixels, and solving the proposed quadratic equation, as well as region-dependent like histogram transformation method, which first transforms the histogram into one with deeper valleys and sharper peaks and then applying the mode method, gray level statistics and relaxation method, in which the “light probabilities” become very high for pixels of light region as compared to those of dark region, and then finding the threshold from the light region. Chow and Kaneko suggested a 7X7 window for local thresholding.

6. Proposals

6.1 Proposal 1: (Global Gradient Threshold)

An arbitrary value is selected as the global threshold values. This threshold is compared with the gradient magnitude values of the image pixels. If the gradient exceeds the thresholds, the pixels are plotted. Further, to improve the output we apply the median filter.

Algorithm/ Pseudo Code before filtering:

Let 'Gmag' store the gradient values of a grayscale image I. Let 'wid' be the image width and 'ht' be the image height. Let the experimental threshold value be 100.

```
For i=1 to wid
    For j=1 to ht
        If Gmag(i,j) >= 100 then plot the pixel.
    End j-loop
End i-loop
```

6.2 Proposal 2: (Effective Average Gradient)

The effective average gradient is selected as the threshold, which is calculated as the ratio of total gradient magnitude to the number of pixels with nonzero gradient values. To improve the output we apply the median filter.

Algorithm/ Pseudo Code:

```
S=sum of gradient values
count=number of pixels with non-zero gradient values
avg=S/count
ht = Height of the image
    For i=1 to wid
        For j=1 to ht
            If Gmag(i,j) >= avg then plot the pixel
        End j-loop
    End i-loop
```

6.3 Proposal 3: (Local Maxima and Difference)

Firstly, the local maximum grey level is selected. Secondly, the difference between the selected grey level and other grey levels of pixels in the same window is observed. If any of the differences exceeds an arbitrary value, the selected pixel has a high chance of being an edge point. We use a 3X3 sliding window as a local sub-image. To improve the result we apply the median filter.

Algorithm/ Pseudo Code:

Let ‘max’ store the maximum gray value, and ‘pos’ store its location. Let a 3X3 sub-image window be chosen for implementation of the algorithm.

```
For i=1 to wid-2
    For j=1 to ht-2
        max=maximum( I(i to i+2,j to j+2) )
        pos=position of pixel of maximum value
        For k=i to i+2
            For l=j to j+2
                If max - I(k,l) >=40 then plot pixel of pos coordinates
            End l-loop
        End k-loop
    End j-loop
End i-loop
```

Here, the preferred cutoff value is 40 (experimentally set), although it can be adaptive.

6.4 Proposal 4: (Global Mean and Standard Deviation)

The mean and standard deviation of the grey level of all the pixels are calculated. If their sum or difference does not exceed the gradient magnitude of the pixel, then the pixel is plotted. We also apply a special case in which we use the difference between mean and half of the standard deviation, where we try to cancel out the unwanted pixels.

Algorithm/ Pseudo Code:

Me=mean of gray values in image I

S=standard deviation

wid = Width of the image

ht = Height of the image

```
For i=1 to wid
    For j=1 to ht
        If Gmag(i,j) >= Me + S then plot pixel
    End j-loop
End i-loop
```

6.5 Proposal 5: (Local Mean and Standard Deviation)

We apply the same concept as described in Proposal 4, except that we find the mean and standard deviation for a 3X3 sliding sub-image. We check for the sum as well as the difference of mean and standard deviation. To improve the result we apply the median filter.

Algorithm/ Pseudo Code:

Let 'Gmag' store the gradient magnitude values of image I.

```
For i=1 to wid-2
    For j=1 to ht-2
        sub = sub-image of I from rows i to i+2 and columns from j to j+2
        M = mean of values in sub
        S = standard deviation of values in sub
        For k=i to i+2
            For l=j to j+2
                If Gmag(k, l) >= M+S then plot pixel at (k, l)
            End l-loop
        End k-loop
    End j-loop
End i-loop
```

7. Tools Used

MATLAB Version R2017

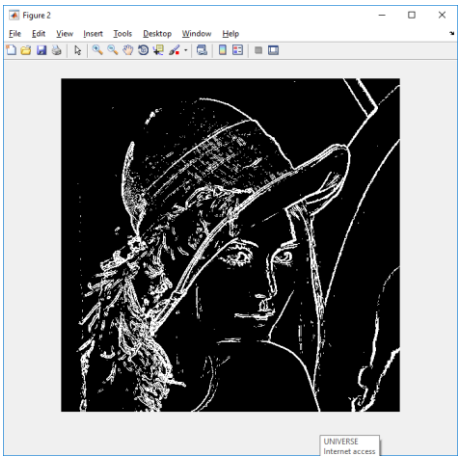
Edge detection is a process of locating an edge of an image and detection of edges in an image is a very important step towards understanding image features. Edges consist of meaningful features and contain significant information. It significantly reduces the image size and filters out information that may be regarded as less relevant, thus preserving the important structural properties of an image.

Most images contain some amount of redundancies that can sometimes be removed when edges are detected and replaced during reconstruction. This is where edge detection comes into play. Also, edge detection is one of the ways of making images not take up too much space in the computer memory. Since edges often occur at image locations representing object boundaries, edge detection is extensively used in image segmentation when images are divided into areas corresponding to different objects. We can use various techniques to perform edge detection and associated algorithms can be implemented in MATLAB.

MATLAB is a convenient tool for us to apply the logic and concepts that can be used for analysing, processing and working on images and videos after converting them into 2D and 3D matrices. Here we have some well defined functions as well. But if we want to implement our own algorithms at the core level, there is provision to do so. Hence it’s a reliable tool for our project.

8. Result and Discussion

Results of Proposal 1 to 5:



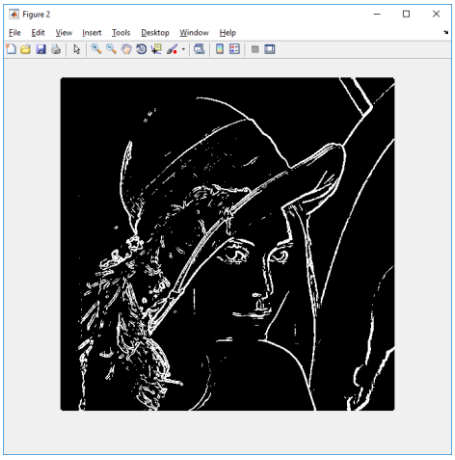
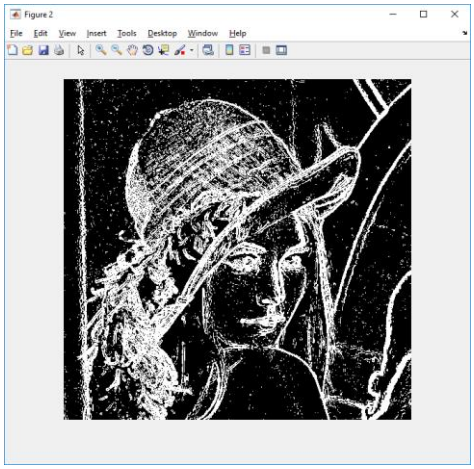


Image before Filtering

Detection of Edges after Filtering

8.1 Results of Proposal 1: (Global Gradient Threshold)



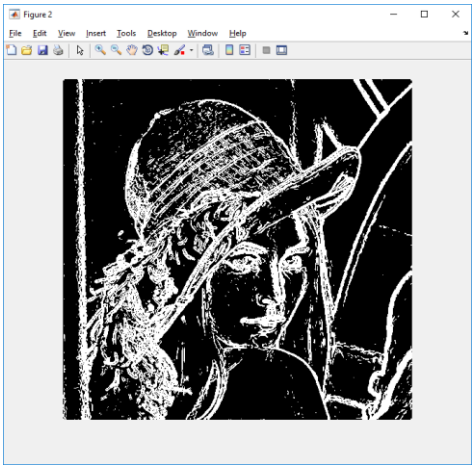


Image before Filtering

Detection of Edges after Filtering

8.2 Results of Proposal 2: (Effective Average Gradient)

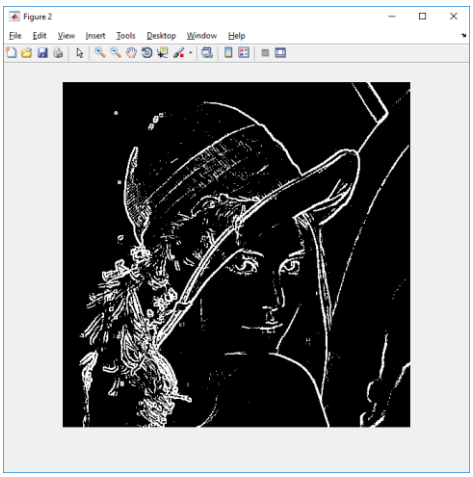
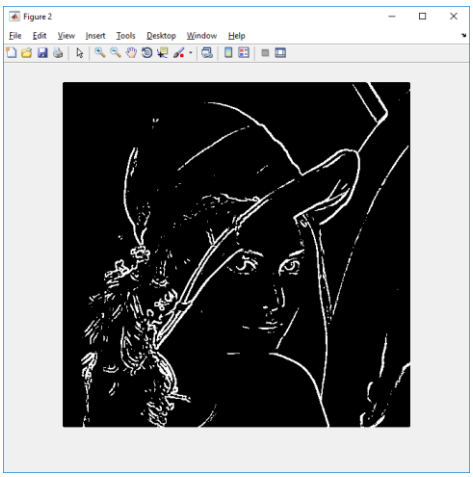


Image before Filtering



Detection of Edges after Filtering

8.3 Results of Proposal 3: (Local Maxima and Difference)

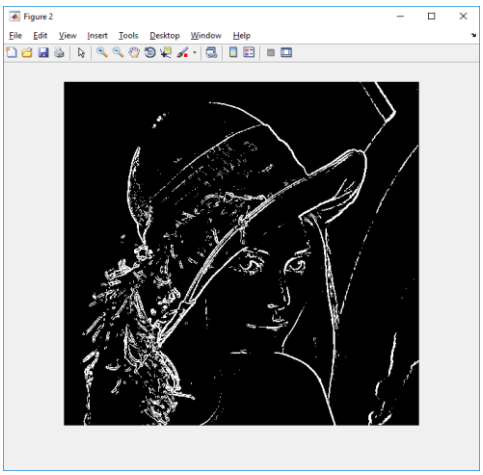


Image before Filtering (Me + S)

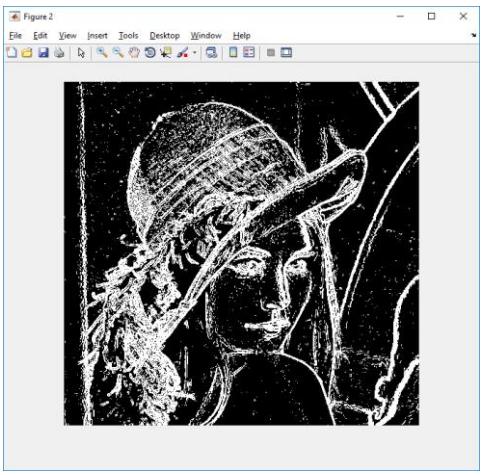


Image before Filtering (Me - S)

8.4 Results of Proposal 4: (Global Mean and Standard Deviation)

Note: a digital image has a very high chance of being contaminated with unwanted signals, removal of which is expected to produce an output of better quality. Of all the filters, median filter is very efficient and thus used on the image. The filtered outputs provided in all cases are with better quality.

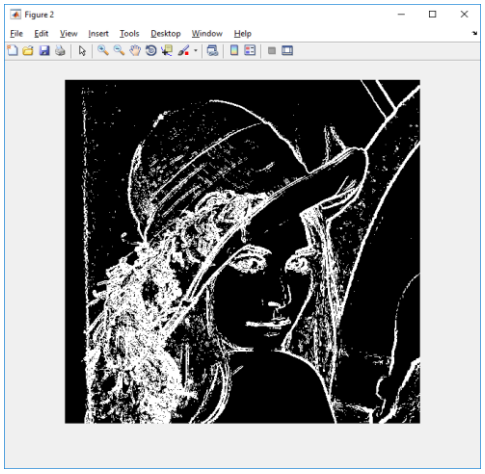


Image before Filtering (M - S)

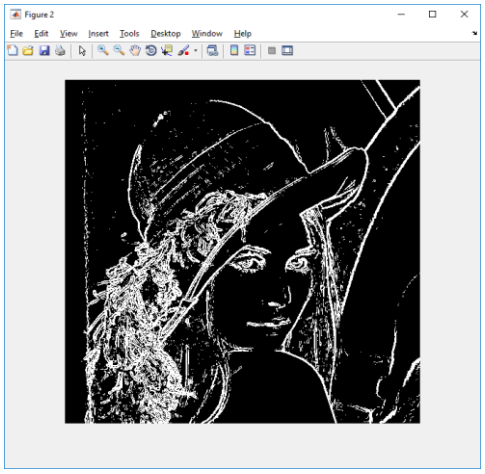
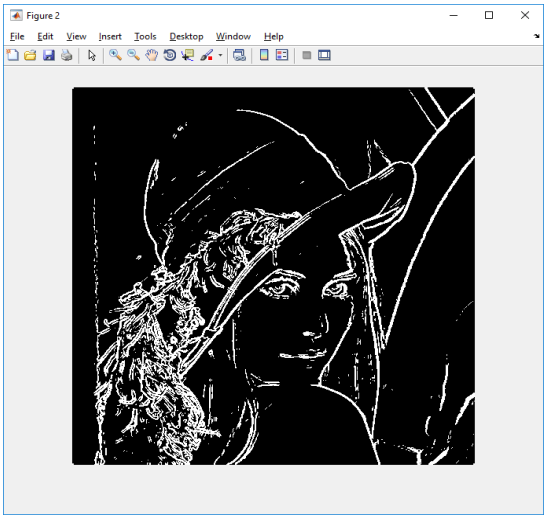


Image before Filtering (M + S)



Detection of Edges after Filtering

8.5 Results of Proposal 5: (Local Mean and Standard Deviation)

Note: After comparing the filtered outputs of all 5 proposals, it has been concluded that Proposal 5 is providing best performance than Proposals 1 and 3 but rest two proposals are not providing satisfactory results.

Discussion

In this project, we have attempted to put forth our proposed edge-detection algorithms, which are expected to produce satisfactory result, is an improvement of the traditional methods, but not optimum like other commercially used techniques. We have also tried to take a rather different, if not ground-breaking, and simple approach in one of the proposals, in which the algorithm works satisfactorily on gray-levels only, rather than gradient magnitude. We even mixed the concepts of image gradient with mathematical statistics, and produce successful results. To check the relations between edge-detecting efficiency and noise, we intended to pass the image through a filter and check whether the output shows an improvement or not. All the proposals have been implemented and tested on MATLAB.

9. Conclusion

Edge detection helps to extract useful features for pattern recognition. The algorithms needed for detecting edges work primarily on finding a threshold value, beyond which the pixels are expected to be edge points. But, our proposed algorithms work in different complexities and mechanisms. In this busy era, people expect to use a device which not only helps them solve problems, but in a short span of time. A good algorithm is one which qualifies in problem-solving and time complexity. In our proposals, some take less time and produce less satisfying results, while others take more time to produce a rather better result. But it is up-to the user whichever algorithm he desires to apply depending upon the Database. Some of our proposals have proven to be different from those ever found.

10. Future Scope of the Work

We have completed literature surveys for our project and identified merits and demerits of the available methods and finally applied our own simple idea to detect image edges and tested our proposals upon popular Lena Image. Actually finishing the job, in full, with proper result by designing an alternative edge detection algorithm of different types of images is really a challenging task if anyone consider pose variant image database. So, detection of image edges using Pose Variant Data Set shall be considered as Future Scope of work.

11. References

[1] Larry S. Davis, Computer Graphics and Image Processing, Academic Press, Inc. (1975) 4, (248--270): Face Recognition: A Survey of Edge Detection Techniques

[2] MathWorks/ Edge Detection // in.mathworks.com/discovery/edge-detection.

[3] Isabelle Guyon, Steve Gunn, Masoud Nikravesh, Lotfi A. Zadeh (Eds.): Feature Extraction: Foundations and Applications, Springer

[4] Rafael C. Gonzalez, Richard E. Woods, Digital Image Processing, 3rd Edition, Person Education

[5] K.S. Fu, J.K. Mui, Pattern Recognition - Vol 13. pp 3 16, Pergamon Press Ltd, 1981: A Survey on Image Segmentation.

[6] K.V. Mardia, T.J. Hainsworth, IEEE Transactions on Pattern Analysis and Machine Learning, Vol. 10, 6 November, 1988: A Spatial Thresholding method for Image Segmentation.

[7] P.K. Sahoo, S.Sooltani, A.K.C. Wong, Computer Vision, Graphics, And Image Processing 41, 233-260 (1988): A Survey of Thresholding Techniques