**Abstract**

In this paper the comparative analysis of various Image Edge Detection techniques is shown. The software used for this purpose is MATLAB 7.0. It has been shown that the Canny’s edge detection algorithm performs better than all these operators under almost all scenarios. In this paper different edge and noise values are used which ranges from .0 to 1.

### 1 Introduction

Image detection is an attractive for detecting discontinuity in gray level. Its boundary between two uniform regions. Edge detection is the strategy of abrupt discontinuity detection. In case of noiseless images, edge detection is very easy, but it becomes a challenging task when the image is a noisy image.

Noisy image is an image which consists of some undesired information along with desired details. For feature extraction and object segmentation, edge detection is the base for it. For enhancing the appearance of blurred image, edge detection filters plays very important role. Edge detection comes under one of the main image segmentation technique. For distinguishing objects from background details, segmentation is used. Image segmentation is a method which splits the image from its background. Approaches for image segmentation are: threshold techniques, edge detection techniques, region-based techniques, and connectivity preserving relaxation methods.
Out of all above image segmentation techniques, edge detection is very popular and widely used [1]. Edge detection splits an image into multiple regions or pixels. In other words edge is defined as the point in an image where sharp changes occur.

2. Edge Detection Methods

![Figure 2.1 - Methods of edge detection](image)

Above figure shows the tree of various edge detection methods.

### 2.1.1) Gradient based method

In this method we used first order derivative. The gradient based methods for edge detection is categorized as: Robert operator, sobel operator and prewitt operator.

**a) Robert Operator** - Robert operator is based on gradient. It firstly calculates the sum of the squares of the difference between diagonally adjacent pixels through discrete differentiation and then finds approximate gradient of the image. The input which is given to Robert operator is first convolved with the default kernels of operator and gradient magnitude and directions are computed. It uses following 2 x 2 two kernels:

\[
D = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} D_x = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \text{ And } D_y = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}
\]

The main quality this operator is its simplicity but having small kernel it is highly sensitive to noise not and not much compatible with today’s technology [3].

**b) Sobel Operator** - Sobel operator is a tool for computing an approximation of image intensity gradient function. In an image at each pixel value, sobel operator provides either the corresponding gradient vector or normal to the vector. It convolves the input image with kernel and finds the gradient magnitude and direction. It emphasizes on high spatial frequency. It uses following 3x3 two kernels:

\[
D_i = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} \text{ And } D_j = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix}
\]

As compared to Robert operator have slow computation ability but as it has large kernel so it is less sensitive to noise as compared to Robert operator. As having larger mask, errors due to effects of noise are reduced by local averaging within the neighborhood of the mask [3].

**c) Prewitt operator** - The function of Prewitt edge detector is almost same as of sobel detector:[3] except constant =1. It’s a way for approximating edge magnitude and orientation.
2.1.2) Gaussian based method

a) Laplacian of Gaussian (LOG) : It was proposed by Marr(1982). The LOG of an image f(x,y) is a second derivative. Its first step is to smooth the image. It first smooths the image and then computes the Laplacian. This yields in double edge image; hence for finding the edge the zero crossing between the double edges is taken.

The Gaussian filtering is combined with Laplacian to break down the image where the intensity varies to detect the edges effectively [3]. As its name consists of laplacian and then Gaussian. The Laplacian of an image marked the regions of rapid intensity change and is therefore often used for edge detection.

b) Canny edge detector - It is an optimal edge detection technique as provide good detection, clear response and good localization. It is widely used and it follows the procedure of finding edges by judging for local maxima of gradient of original image. For gradient calculation, Gaussian filter is used in it. This method detects the strong and weak edges. For this purpose, it uses two threshold levels. Actually this method detects the true weak edges.

3. Research Methodology

There are many ways to perform the edge detection. However, it may be grouped into two categories, that are gradient and Laplacian. The gradient method detects the edges by looking for the maximum and minimum in the first derivative of the image. The Laplacian method searches for the zero crossings in the second derivative of the image to find edges.[4]

It does not give much importance to those pixels that are closer to center.

\[
D_i = \begin{bmatrix} -1 & 0 & +1 \\ -1 & 0 & +1 \\ -1 & 0 & +1 \end{bmatrix} \quad \text{And} \quad D_j = \begin{bmatrix} +1 & +1 & +1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}
\]
4. Results And Discussion

4.1 Comparison of different edge methods using threshold value 0.4

Figure 4.1.1: shows edge detection with sobel method using intensity 0.4

Figure 4.1.2: shows edge detection with Roberts method using intensity 0.4

Figure 4.1.3: shows edge detection with canny method using intensity 0.4
Figure 4.1.4: shows edge detection with prewitt method using intensity 0.4

Figure 4.1.5: shows edge detection with log method using intensity 0.4

Figure 4.1.6: shows edge detection with zerocross method using intensity 0.4
4.2 Comparison of different methods using threshold value 0.8

Figure 4.2.1: shows edge detection with sobel method using intensity 0.8

Figure 4.2.2: shows edge detection with Roberts method using intensity 0.8

Figure 4.2.3: shows edge detection with Canny method using intensity 0.8
Figure 4.2.4: shows edge detection with Prewitt method using intensity 0.8

Figure 4.2.5: shows edge detection with Log method using intensity 0.8

Figure 4.2.6: shows edge detection with Sobel method using intensity 0.8
4.3 Results
From above figures it is clear that canny edge detection method shows the better results irrespective of any other method even in noisy conditions at intensity both at 0.4 and 0.8. Canny edge detector shows better localization as well as smooth detection. It detects single edges for one response.

Table 4.3.1 : Resultant Response Comparison of Edge Detection methods

<table>
<thead>
<tr>
<th>S. No</th>
<th>Edge methods</th>
<th>For threshold value 0.4</th>
<th>For threshold value 0.8</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Sobel</td>
<td>Dispersive</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>Roberts</td>
<td>Mild Detection</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>Canny</td>
<td>Very Sharp Detection</td>
<td>Sharp Detection</td>
</tr>
<tr>
<td>4</td>
<td>Prewitt</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>5</td>
<td>Log</td>
<td>No Detection</td>
<td>No Detection</td>
</tr>
<tr>
<td>6</td>
<td>ZeroCross</td>
<td>No Detection</td>
<td>No Detection</td>
</tr>
</tbody>
</table>

5 Conclusion
This study of different Edge detection techniques and after their experimental results shows that canny method is best. It shows better results in nose conditions. The relative performance of various edge detection techniques is carried out with an image by using MATLAB software. Canny result is best one when compared to all other images using different methods. Above results depicts that canny edge detection operation performs outstanding even while setting different threshold values i.e., 0.4 and 0.8.

References

Appendix ‘A’
clc
close all;
disp('FOR IMAGES')
disp('1. my image')
disp('2. naina image')
disp('3. football.jpg')
disp('4. eight.tif')
disp('5. snowflakes.png')
disp('6. pout.tif')
disp('7. mandi.tif')
disp('8. rice.png')

n=input('enter the value for image\n')

disp('PRESS')
q=input('1. edge detection\n2.filtering the noise in the image\n')

disp('FOR NOISE')
disp('1. speckle')
disp('2. poisson')
disp('3. sale & pepper')
disp('4. localvar')
h=input('enter the value for noise in original image\n')

disp('FOR EDGE DETECTION METHOD')
disp('1. sobel method')
disp('2. roberts method')
disp('3. canny')
disp('4. prewitt')
disp('5. log')
disp('6. zerocross')
d=input('enter the value for edge detection method and detecting the edges of original image \n')
j=edge3(d,h,n);%main function that detecting the edges of original image

function j=edge3(d,h,n)
i=image2(n)
[m]=imnoise3(h,n)
x=input('enter the value for edge\n')
switch d
    case 1
j=edge(i,'sobel',x)
subplot(2,2,2)
imshow(j)
xlabel('step 2')
title('SOBEL')
j=edge(m,'sobel',x)
subplot(2,2,3)
imshow(j)
xlabel('step 4')
title('SOBEL(with noise)')
    case 2
j=edge(i,'canny',x)
subplot(2,2,2)
imshow(j)
xlabel('step 4...CANNY')
j=edge(m,'canny',x)
subplot(2,2,3)
imshow(j)
title('step 4...CANNY(with noise)')

case 3
j=edge(i,'roberts',x)
subplot(2,2,2)
imshow(j)
title('step 2...ROBERTS')
j=edge(m,'roberts',x)
 subplot(2,2,3)
imshow(j)
title('step 4...ROBERTS(with noise)')
    case 4
j=edge(i,'prewitt',x)
 subplot(2,2,2)
imshow(j)

title('step 2...PREWITT')
j=edge(m,'prewitt',x)
 subplot(2,2,3)
imshow(j)

title('step 4...PREWITT(with noise)')
    case 5
j=edge(i,'log',x);
 subplot(2,2,2)
imshow(j)
title('step 2...LOG')
j=edge(m,'log',x)
 subplot(2,2,3)
imshow(j)

title('step 4...LOG(with noise)')
    case 6
j=edge(i,'zerocross',x)
 subplot(2,2,2)
imshow(j)

title('step 2...ZEROCROSS')
j=edge(m,'zerocross',x)
 subplot(2,2,3)
imshow(j)

title('step 4...ZEROCROSS(with noise)')
end
end