Volume 3, No. 5, May 2012
Journal of Global Research in Computer Science
ISSN-2229-371X
TECHNICAL NOTE
Available Online at www.jgrcs.info

A JAVA BASED TOOL FOR BASIC IMAGE PROCESSING AND EDGE DETECTION

Tirtharaj Dash[*1] and Tanistha Nayak[2]
[*1, 2]Department of Information Technology
National Institute of Science and Technology
Berhampur-761008, India
tirtharajnist446@gmail.com
tanisthanist213@gmail.com

***Abstract*—** Edge detection is of fundamental importance of image processing. It is the foundation of image comparison, forgery detection, image detection etc. In this work a tool has been developed for basic image processing viz. scaling, histogram analysis and edge detection. The tool is developed in Java language. To reduce the noise in edge detection, an Enhanced method has been proposed and corresponding results is shown. We basically focused on Sobel edge detection technique. The results showed that the enhanced version is better than the ordinary method.

***Keywords*-**Edge Detection; scaling; histogram; Sobel; Java; noise reduction;

INTRODUCTION

Edge Detection is a very important area in the field of Computer Vision, Image Processing viz. feature detection and feature extraction in which the image brightness changes sharply [1]. The purpose of detecting sharp changes in image brightness is to capture important events and changes in properties of the world. Edges define the boundaries between regions in an image, which helps with segmentation and object recognition. They can show where shadows fall in an image or any other distinct change in the intensity of an image. Edge detection is a fundamental of low-level image processing and good edges are necessary for highest level processing. In typical images, edges characterize object boundaries and are therefore useful for segmentation, registration, and identification of objects in a scene [2]. The quality of edge detection is highly dependent on lighting conditions, the presence of objects of similar intensities, density of edges in the scene, and noise. The problem is that, general edge detectors behave very poorly. While their behavior may fall within tolerances in specific situations, in general edge detectors have difficulty adapting to different situations. In ideal cases, the output of an edge detector should be a set of connected curves that indicate the boundaries of objects, as well as the curves that correspond to discontinuities in surface orientation. But practically, this does not happen. The curves we get are distorted at some places and along with the curve there is a lot of noise. After implementing *enhanced edge detection* techniques, we are able to get a smooth edge of the object of our interest. Moreover we are able to suppress the noise to a large extent.

There are different types of edge detectors. Different edge detectors work better under different condition. A survey on some of the Edge Detection techniques has been discussed below.

SURVEY OF SOME TECHNIQUES

The Canny Edge Detector:

The Canny edge detector is considered to be the standard edge detection algorithm. It was developed by John Canny. It performs better many of the newer algorithms that have been developed. Canny saw the edge detection problem as a signal processing optimization problem, and he developed an objective function to be optimized [3]. The solution to this problem was a complex exponential function, but Canny solved this problem in several ways to approximate and optimize the edge-searching problem. The steps in the Canny edge detector are as follows:

a. ***Smooth the image with a two dimensional Gaussian*:** In most cases the computation of a two dimensional Gaussian is costly, so it is approximated by two one dimensional Gaussians, one in the x direction and the other in the y direction.
b. ***Take the gradient of the image:*** This shows changes in intensity, which indicates the presence of edges. This actually gives two results, the gradient in the x direction and the gradient in the y direction.
c. ***Non-maximal suppression.*** Edges will occur at points the where the gradient is at a maximum. Therefore, all points not at a maximum should be suppressed. In order to do this, the magnitude and direction of the gradient is computed at each pixel. Then for each pixel check if the magnitude of the gradient is greater at one pixel's distance away in either the positive or the negative direction perpendicular to the gradient. If the pixel is not greater than both, suppress it.
d. ***Edge Thresholding*:** The method of thresholding used by the Canny Edge Detector is referred to as "hysteresis". It makes use of both a high threshold and a low threshold. If a pixel has a value above the high threshold, it is set as an edge pixel. If a pixel has a value above the low threshold and is the neighbor of an edge pixel, it is set as

an edge pixel as well. If a pixel has a value above the low threshold but is not the neighbor of an edge pixel, it is not set as an edge pixel. If a pixel has a value below the low threshold, it is never set as an edge pixel.

The Marr-Hildreth Edge Detector:

The Marr-Hildreth edge detector was a very popular edge operator. It is a gradient based operator which uses the Laplacian to take the second derivative of an image. The idea is that if there is a step difference in the intensity of the image, it will be represented by in the second derivative by a zero crossing.

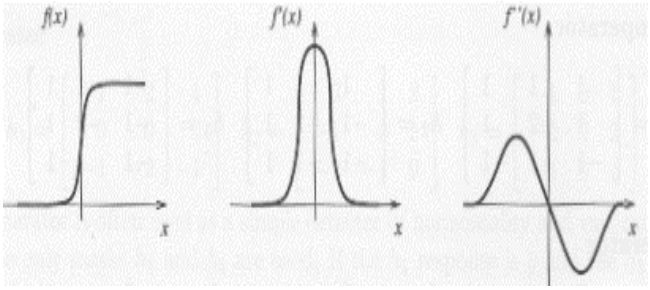


Figure 1. A figure showing a Function and its 2nd derivative

So the general algorithm for the Marr-Hildreth edge detector is as follows:

a. Smooth the image using a Gaussian. This smoothing reduces the amount of error found due to noise.
b. Apply a two dimensional Laplacian to the image: $r^2 f = \partial^2 f/\partial x^2 + \partial^2 f/\partial y^2$.

This Laplacian will be rotation invariant and is often called the "Mexican Hat operator" because of its shape:

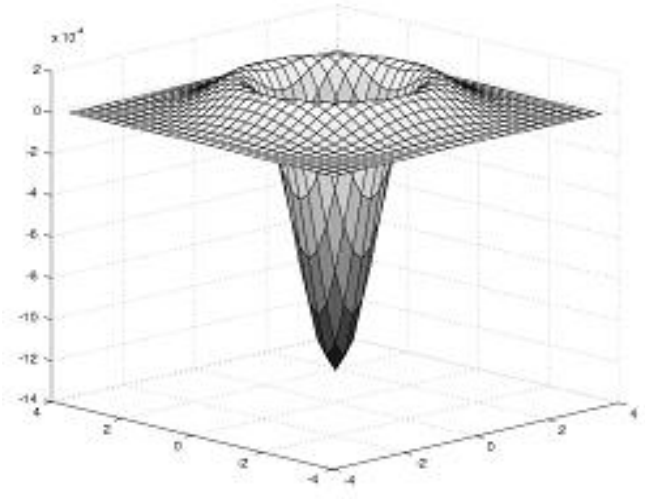


Figure 2. 3D shape of the second derivative

This operation is the equivalent of taking the second derivative of the image.

c. Loop through every pixel in the Laplacian of the smoothed image and look for sign changes. If there is a sign change and the slope across this sign change is greater than some threshold, Mark this pixel as an edge. Alternatively, you can run these changes in slope through a hysteresis (described in the Canny edge detector) rather than using a simple threshold.

Local Threshold and Boolean Function Based Edge Detection:

This edge detector is fundamentally different than many of the modern edge detectors derived from Canny's original. It does not rely on the gradient or Gaussian smoothing. It takes advantage of both local and global thresholding to find edges. Unlike other edge detectors, it converts a window of pixels into a binary pattern based on a local threshold, and then applies masks to determine if an edge exists at a certain point or not. By calculating the threshold on a per pixel basis, the edge detector should be less sensitive to variations in lighting throughout the picture [4]. It does not rely on blurring to reduce noise in the image. It instead looks at the variance on a local level. The algorithm is as follows:

a. Apply a local threshold to a 3x3 window of the image. Because this is a local threshold, it is recalculated each time the window is moved. The threshold value is calculated as the mean of the 9 intensity values of the pixels in the window minus some small tolerance value. If a pixel has an intensity value greater than this threshold, it is set to a 1. If a pixel has an intensity value less than this threshold, it is set to a 0. This gives a binary pattern of the 3x3 window.
b. Compare the binary pattern to the edge masks. There are sixteen possible edge-like patterns that can arise in a 3x3 indow, as shown in Figure 3 below.

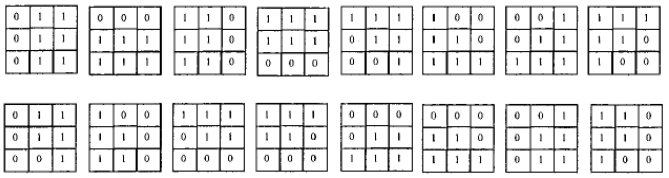


Figure 3. Binary pattern of 3×3 window

If the binary window obtained in step 1 matches any of these sixteen masks, the center pixel of the window is set to be an edge pixel.

c. Repeat steps 1 and 2 for each pixel in the image as the center pixel of the window. This will give all edges, but it will also give some false edges as a result of noise.
d. Use a global threshold to remove false edges. The variance for each 3x3 window is calculated, which will have a maximum at an edge. This value is then compared with a global threshold based on the level of noise in the image. If the value is greater than the threshold, it is kept as an edge. If it is not greater than the threshold, it is removed.

METHODOLOGY

A stepwise method is being followed for development of this tool. This tool is developed using Java language [5]. These are as described below.

Read the Image:

The user is allowed to open an image file for which the edge detection method will be followed. However, we have tested

the tool for BMP images. But this tool can work for any file types.

Color Scaling:

The image is then converted to different color scales. Mostly used color scale is the Gray Scale. However, Red, Green, Blue scales conversions are also being done.

Histogram Analysis:

The image histogram can be displayed and analyzed by the user.

Edge Detection:

Now, two methods for edge detections are implemented. Out of these two, one is Enhanced edge detection tool which follows our methodology.
The section IV shows the developed tool and a sample result.

RESULT

The windows for the developed tool are given below. Figure 4 shows the welcome window of the developed tool.

Figure 4. Welcome window

The user will be asked to select an image file for processing. The windows are given figure 5(a-b).

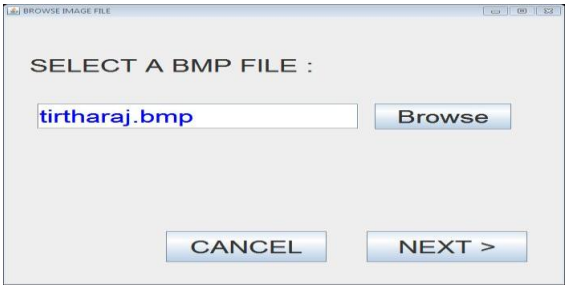


Figure 5 (a). Image Selection window

Figure 5 (b). Loaded Image Window for tirtharaj.bmp image

Now the user will be asked for different operations like scaling, histogram analysis and edge detection. The windows are shown below.

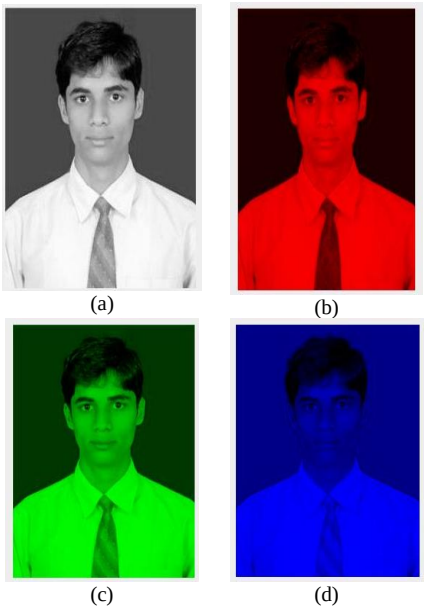


Figure 6. Image Scaling (a) Gray (b) Red (c) Green (d) Blue

A Histogram analysis window is given in figure 7.

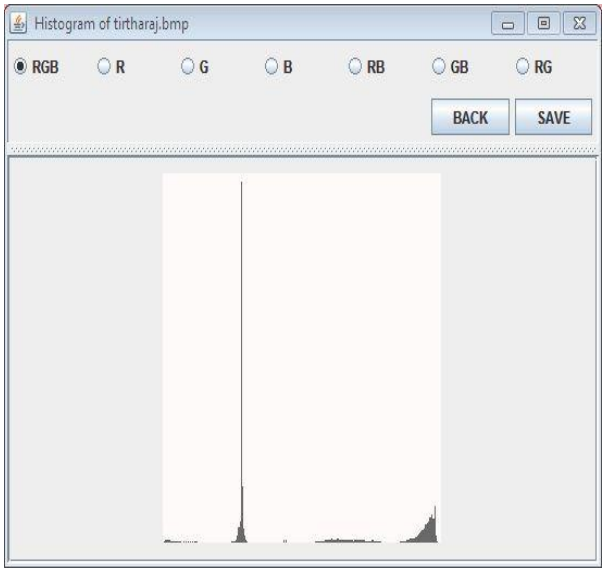
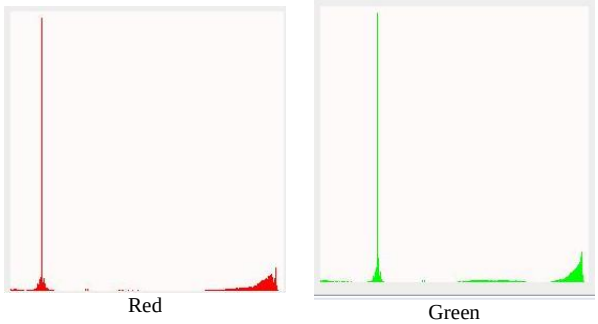


Figure 7. Histogram analysis window



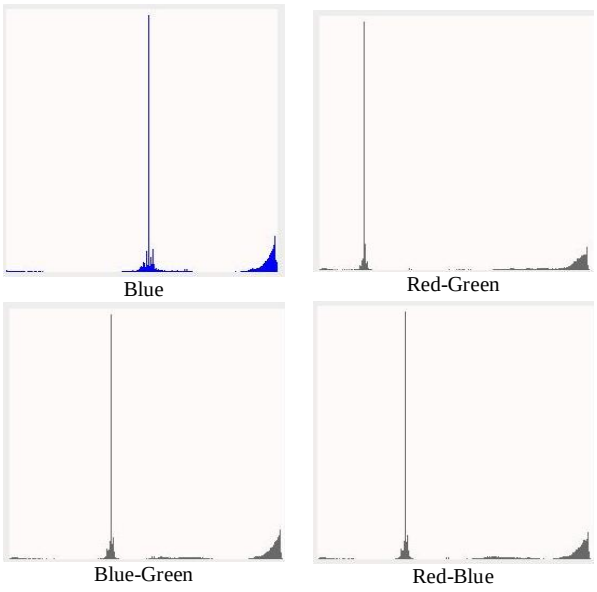


Figure 8. Different Histograms of the image

For our method, we have used Sobel operator for edge detection. This operator performs a 2 dimensional spatial gradient measurement of the considered image. This operator uses two 3×3 convolution mask matrices; one X-gradient and another as Y-gradient. A matrix is given below to show these two matrices.

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

x-gradient

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

Y-gradient

For edge detection, these matrices will be slide over an area of image pixels row-wise. In each manipulation one column will be considered until the end is reached. Then next row will be allowed to slide. The magnitude is given in Equation-1 below.

$$|G| = \sqrt{G_x{}^2 + G_y{}^2} \qquad (1)$$

The windows showing edge detection method are given in figure 9.

Figure 9. Windows showing output of Sobel Edge Detection

We have also proposed an enhanced edge detection method to reduce any kind of noise in the image. The corresponding output after noise reduction is given in figure 10.

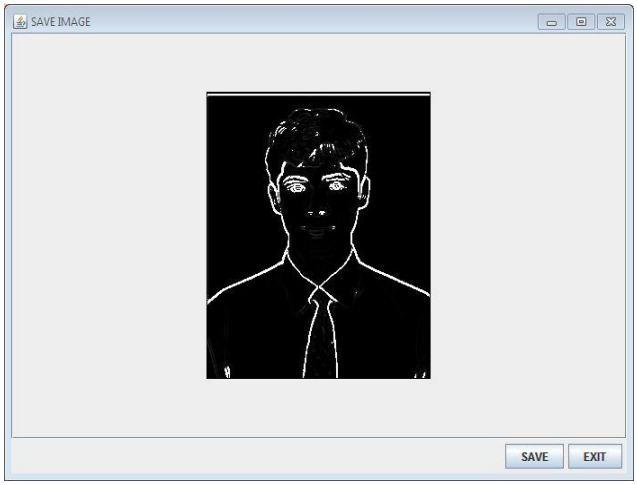


Figure 10. Enhanced Edge detected image

CONCLUSION

This work proposed a tool for basic image processing and edge detection. Edge detection is a major module of this tool. For image noise reduction, enhanced edge detection method was proposed. Enhancing the edge means sharpening the edge of image and filtering with higher accuracy. The result showed that the proposed enhanced method is better as compared to ordinary used techniques.

As future work, it will be interesting to enhance the image using Fuzzy rule bases and the tool will be focusing on medical images. The authors are currently working on it.

REFERENCES

[1] Philips D., "Image Processing in C", 2nd Edition, ISBN: 0-13-104548-2 (2000).

[2] Muhammad Bilal Ahmad and Tae-Sun Choi, "Local Threshold and Boolean Function Based Edge Detection", IEEE Transactions on Consumer Electronics, Vol. 45, No 3. August 1999.

[3] Robyn Owens, "Lecture 6", Computer Vision IT412, 10/29/1997. http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/OWENS/LECT6/node2.html.

[4] Sarah Price, "Edges: The Canny Edge Detector", July 4, 1996. http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/MARBLE/low/edges/canny.html.

[5] Robinson M., Vorobiev P., "Swing", 2nd Edition, ISBN: 1930110-88-X (2003).