J Inf Process Syst, Vol.9, No.4, pp.575-591, December 2013
http://dx.doi.org/10.3745/JIPS.2013.9.4.575

pISSN 1976-913X
eISSN 2092-805X

Simple Fuzzy Rule Based Edge Detection

O.P. Verma*, Veni Jain*, and Rajni Gumber*

Abstract—Most of the edge detection methods available in literature are gradient based, which further apply thresholding, to find the final edge map in an image. In this paper, we propose a novel method that is based on fuzzy logic for edge detection in gray images without using the gradient and thresholding. Fuzzy logic is a mathematical logic that attempts to solve problems by assigning values to an imprecise spectrum of data in order to arrive at the most accurate conclusion possible. Here, the fuzzy logic is used to conclude whether a pixel is an edge pixel or not. The proposed technique begins by fuzzifying the gray values of a pixel into two fuzzy variables, namely the black and the white. Fuzzy rules are defined to find the edge pixels in the fuzzified image. The resultant edge map may contain some extraneous edges, which are further removed from the edge map by separately examining the intermediate intensity range pixels. Finally, the edge map is improved by finding some left out edge pixels by defining a new membership function for the pixels that have their entire 8-neighbourhood pixels classified as white. We have compared our proposed method with some of the existing standard edge detector operators that are available in the literature on image processing. The quantitative analysis of the proposed method is given in terms of entropy value.

Keywords—Edge detection, Edge improvement, Fuzzy rules, Membership function.

1. INTRODUCTION

Edge detection plays a vital role in many of the applications of image processing such as pattern recognition and image segmentation. Edges correspond to sharp variations of image intensity and convey vital information in an image. Edges are formed from pixels with derivative values that exceed a pre-set threshold [1]. Edge detection not only extracts the edges of the interested objects from an image, but it also forms the basis for image fusion, shape extraction, image matching, image tracking, etc.

Normally, the edge detection methods use the gradient of images and arithmetic operators. The most popular edge detection methods, such as Sobel, Prewitt, Roberts, [1] etc. detect edges using a first-order derivative of intensity since they consider edges to be a set of pixels where there is an abrupt change in the intensity of the gray level. However, the Canny edge detector [2] searches for the partial maximum value of the image gradient. The gradient is counted by the derivative of the Gauss filter. The Canny operator uses two thresholds to detect strong edges and weak edges, respectively. These edge detection methods do not consider the neighborhood of the pixel, while in our proposed method the neighborhood plays an important role in edge de-

Manuscript received February 13, 2013; first revision May 28, 2013; second revision September 6, 2013; accepted; October 22, 2013.
Corresponding Author : Rajni Gumber (rajni_gumber23@yahoo.com)
* Delhi Technological University, Delhi (opverma.dce@gmail.com, vjain1735@gmail.com, rajni_gumber23@yahoo.com)

tection. These classical operators [1-2] work well in circumstances where the area of the image that is being studied is of a high contrast. These edge detection techniques have fixed results or some parameters like a threshold for finding good results. On the other hand, fuzzy logic employs simple if-then rules, which do not require any thresholding or complex gradient based calculations. Thus, an edge detection method that is based on fuzzy logic is being proposed in this paper.

The theory of fuzzy sets was first introduced by Zadeh [3], and has been successfully applied to many image processing and pattern recognition problems.

Many edge detection techniques that are based on fuzzy sets and rules have been proposed. For instance, Pal *et al.*[4] proposed a fuzzy edge detection method for X-ray images. Zhang *et al.*[5] proposed a fuzzy edge detection method, which overcame the drawbacks of the Pal *et al.* [5] method, by using a new enhancement operator that was based on the Guassian function. Law *et al.* [6]used fuzzy reasoning for edge detection. This included three stages: filtering, detection, and tracing. Alshennawy *et al.* [7] proposed a fuzzy inference system for finding the edge maps of simple binary images without using the complex derivatives and thresholding, but the method only works well for simple black and white images and not for complex images. Begol *et al.* [8] improved this drawback by considering each pixel to be a fuzzy input and by examining fuzzy rules in its vicinity, the edge pixel is specified more clearly as compared to Alshennawy *et al.* [7]. Kiranpreet Kaur *et al.* [9] proposed a similar fuzzy logic based image edge detection algorithm that had a 2×2 window instead of the 3×3 window used in Alshennawy *et al.* [7]. The drawback of this method is the need to process the output again for better results.

Another excellent work proposed by Liang *et al.* [10] defines a competitive fuzzy edge detector by using fuzzification for classification and then competitive rules are applied to find the edges. This method gives significant results but requires users to set the edge sensitivity level.

There are other edge detection methods that use a hybrid of fuzzy technique with other techniques like neural networks, arithmetic operators, etc. For instance, Wang *et al.* [11] proposed an edge detection method, which is a hybrid of fuzzy technique and neural networks. This gives better results than using individual methods. Another Fuzzy edge detection method that was proposed bySinaie *et al.* [12] is a hybrid edge detection method that is based on fuzzy sets and cellular learning automata (CLA). It uses fuzzy sets to find edges and CLA to enhance the edges, and hence, is a bit time-consuming. Tizhoosh, H.R. [13] proposed fast fuzzy edge detectors based on heuristic membership functions, simple fuzzy rules, and complement are simple to implement and are fast to compute. These methods give rough edge maps, which are useful in some cases where results are urgently required.

There are some other methods that also use a combination of fuzzy rules and evolutionary techniques. For instance, C.Naga Raju *et al.* [14] proposed a fuzzy logic based ant colony system that is used for image classification and analysis. It requires extensive computation and does not produce good results for images that contain small and overlapped objects. Khalid *et al.*[15] combine the fuzzy heuristic edge detection technique with the particle swarm optimization algorithm. Another excellent approach for edge detection that uses a combination of the bacterial foraging algorithm (BFA) and the probabilistic derivative technique, which is derived from the ant colony system, is presented by O.P. Verma *et al.* [16].

In this paper, we modified Alshennawy *et al.* [7] to get better results for simple, as well as for the complex, gray scale images. In Alshennawy *et al.* [7], there is a fuzzy technique for edge detection, which only works well for simple black and white images and gives poor results for complex images. Here, we used the previously defined rules along with newly defined fuzzy rules and modified the membership function. When applied to simple and complex images, this gives improved results, as compared to the Alshennawy *et al.* [7] method. The two extra rules are also defined to encounter the noise problem.

Our proposed edge detection method uses the defined fuzzy rules to find the candidate edge pixels, by applying the rules to each pixel along with its 8-neighbourhood. Before applying the rules, each pixel is categorized into a "white" and "black" pixel using the triangular membership function.

This paper is organized as follows: Section 2 explains our proposed edge detection method. Section 3 describes the algorithm and pseudo-code for the proposed method. Section 4 compares the results of the proposed method with the sobel operator and the method proposed in [7] for some standard images. The conclusions are presented in Section 5.

2. THE PROPOSED FUZZY EDGE DETECTION METHOD

There are a lot of ways to detect edges using fuzzy image processing. But the simplest way is to fuzzify the image. This involves finding the membership value of each pixel for a particular set and then applying the defined rules to the fuzzified image to find the edge map.

2.1 Fuzzification

If the information in a database is inexact, incomplete, or not entirely certain, then the systematic use of fuzzy logic becomes practically indispensable. In many image processing applications, the image information that is to be processed, is uncertain and imprecise. In the proposed approach, the question of whether a pixel is darker or brighter comes under the realm of fuzzification. The darker pixels are placed in the black class, whereas, the brighter ones are put in the white class. In order to fuzzify the image, the membership of each pixel is found by using the triangular membership function shown in Fig. 1. The membership function of an element defines the degree to which that element belongs to the fuzzy set. The value of the membership function always lies between [0…1].

The image pixels are fuzzified into two sets, viz. the black and the white, by using the **Triangular** membership function.

Membership for the black fuzzy set is defined as:

$$\mu_{black}(x) = \begin{cases} 0 & x < 0 \\ \dfrac{255-x}{255} & 0 \le x \le 255 \end{cases} \tag{1}$$

Here, x is the intensity of a pixel.

After calculating $\mu_{black}(x)$ and $\mu_{white}(x)$, the membership value of both the sets is compared and the pixel is assigned the fuzzy variable whose membership value is high. The example of the fuzzification step for a 3×3 window is shown in Fig 2. Fig. 2(a) represents the intensity values of the pixels in a 3×3 window. Fig. 2(b) represents the maximum of the $\mu_{black}(x)$ and $\mu_{white}(x)$ where x represents the intensity value of that pixel and finally, Fig. 2(c) represents the fuzzy set whose

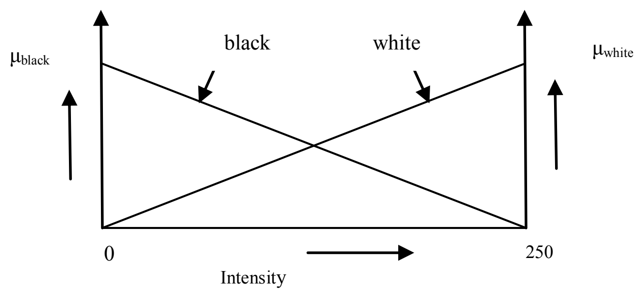


Fig. 1. Triangular membership function

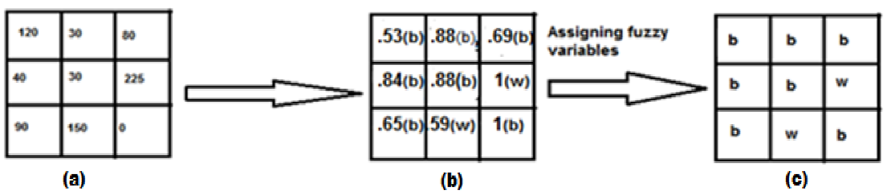


Fig. 2. Fuzzification step of the proposed edge detection method (a) The intensity value of pixels in an image (b) The final membership of each pixel =max(μblack,μwhite) (c) The final fuzzy variable assigned to each pixel.

membership value is shown in Fig. 2(b).

Membership for the white fuzzy set is defined as:

$$\mu_{white}(x) = \begin{Bmatrix} 0 & x < 0 \\ \dfrac{x}{255} & 0 \le x \le 255 \end{Bmatrix} \tag{2}$$

2.2 Fuzzy Rules

Human beings make decisions based on rules. The decision and the means of choosing that decision are replaced by fuzzy sets and the rules are replaced by fuzzy rules. A fuzzy rule is defined as a conditional statement in the form of:

IF x is A
THEN y is B

Where x and y are linguistic variables, A and B are linguistic values that are determined by fuzzy sets on the universe of the discourse X and Y, respectively. The fuzzy rules used in the proposed edge detection approach take into account the linguistic values of the 8-neighborhood of the pixel that is under consideration. Here, the linguistic values can be white or black.

The fuzzy rule is a control system that is used to infer decisions from a knowledge base. The knowledge base to infer the edge pixel in an image is the pixel with its 8-neighborhood. The decision whether each pixel is an edge pixel or not is made by using the fuzzy rules that are applied to the 8-neighborhood. The pixels in the 8-neighborhood of a pixel may be black or white Let I be the fuzzified image and I(i,j) represent the fuzzy value assigned to the pixel (i,j), where i & j represent the coordinates of the decision pixel.

The coordinates of decision pixel's 8-neighborhood are defined as show in Fig.3.

(i-1,j-1)	(i-1,j)	(i-1,j+1)
(i,j+1)	(i,j)	(i,j+1)
(i+1,j-1)	(i+1,j)	(i+1,j+1)

Fig. 3. The 8-neighbourhood of pixel (i, j)

On the basis of the number of white and black pixels in the neighborhood of decision pixel, a total of 30 rules are classified into 5 subclasses as follows:

Class 1: Rules with 3 black & 5 white pixels in neighbourhood shown in figure 4(a)

Rule 1: If I (i-1,j-1) AND I(i-1,j) AND I(i-1,j+1) AND I(i,j-1) AND I(i,j+1) are white and I(i+1,j-1) AND I(i+1,j) AND I(i+1,j+1) are black then (i,j) is the edge pixel

Rule 2: If I(i-1,j-1) AND I(i-1,j) AND I(i-1,j+1) are black and I(i,j-1) AND I(i,j+1) AND I(i+1,j-1) AND I(i+1,j) AND I(i+1,j+1) are white then (i,j) is the edge pixel

Rule 3: If I(i-1,j-1) AND I(i-1,j) AND I(i-1,j+1) are black and I(i,j-1) AND I(i,j+1) AND I(i+1,j-1) AND I(i+1,j) AND I(i+1,j+1) are black then (i,j) is the edge pixel

Rule 4: If I(i-1,j-1) AND I(i,j-1) AND I(i+1,j-1) are black and I(i-1,j) AND I(i-1,j+1) AND I(i,j+1) AND I(i+1,j) AND I(i+1,j+1) are white then (i,j) is the edge pixel

w	w	w
w	E	w
b	b	b

Rule 1

w	w	b
w	E	b
w	w	b

Rule 2

b	b	b
w	E	w
w	w	w

Rule 3

b	w	w
b	E	w
b	w	w

Rule 4

Fig. 4(a). Fuzzy Rules (Class 1 rules)

Class 2: Rules with 4 black & 4 white pixels in the neighborhood shown in Figure 4(b)

Rule 5: If I(i-1,j-1) AND I(i,j-1) AND I(i+1,j-1) AND I(i+1,j) are white and I(i-1,j) AND I(i-1,j+1) AND I(i,j+1) AND I(i+1,j+1) are black then (i,j) is the edge pixel

Rule 6: If I(i-1,j-1) AND I(i,j-1) AND I(i+1,j-1) AND I(i-1,j) are white and I(i-1,j+1) AND I(i,j+1) AND I(i+1,j) AND I(i+1,j+1) are black then (i,j) is the edge pixel

Rule 7: If I(i-1,j-1) AND I(i,j-1) AND I(i+1,j-1) AND I(i-1,j) are black and I(i-1,j+1) AND I(i,j+1) AND I(i+1,j) AND I(i+1,j+1) are white then (i,j) is the edge pixel

Rule 8: If I(i-1,j-1) AND I(i,j-1) AND I(i+1,j-1) AND I(i+1,j) are black and I(i-1,j) AND I(i-1,j+1) AND I(i,j+1) AND I(i+1,j+1) are white then (i,j) is edge pixel

Rule 9: If I(i-1,j-1) AND I(i-1,j) AND I(i-1,j+1) AND I(i,j+1) are white and I(i,j-1) AND I(i+1,j-1) AND I(i+1,j) AND I(i+1,j+1) are black then (i,j) is the edge pixel

Rule 10: If I(i-1,j-1) AND I(i-1,j) AND I(i-1,j+1) AND I(i,j-1) are white and I(i,j+1) AND I(i+1,j-1) AND I(i+1,j) AND I(i+1,j+1) are black then (i,j) is the edge pixel

Rule 11: If I(i-1,j-1) AND I(i-1,j) AND I(i-1,j+1) AND I(i,j+1) are black and I(i,j-1) AND I(i+1,j-1) AND I(i+1,j) AND I(i+1,j+1) are white then (i,j) is the edge pixel

Rule 12: If I(i-1,j-1) AND I(i-1,j) AND I(i-1,j+1) AND I(i,j-1) are black and I(i,j+1) AND I(i+1,j-1) AND I(i+1,j) AND I(i+1,j+1) are white then (i,j) is the edge pixel

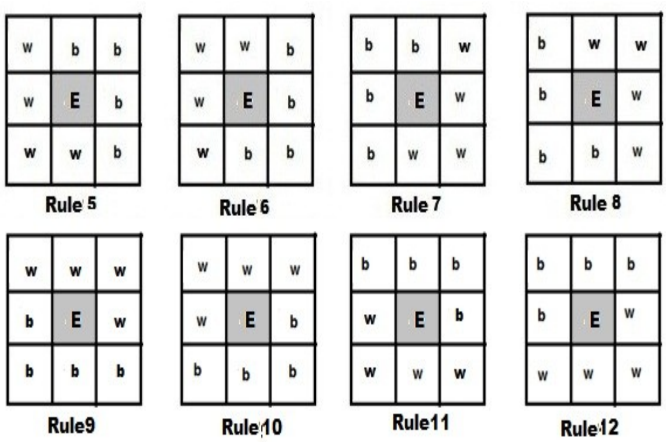


Fig. 4(b). Fuzzy Rules (Class 2 rules)

<u>Class 3: Rules with 5 black & 3 white pixels in the neighborhood shown in Figure 4(c)</u>

Rule 13: If I(i-1,j-1) AND I(i,j-1) AND I(i+1,j-1) AND I(i-1,j) AND I(i+1,j) are black and I(i-1,j+1) AND I(i,j+1) AND I(i+1,j+1) are white then (i,j) is the edge pixel

Rule 14: If I(i-1,j-1) AND I(i-1,j) AND I(i-1,j+1) are white and I(i,j-1) AND I(i+1,j-1) AND I(i+1,j) AND I(i+1,j+1) AND I(i,j+1) are black then (i,j) is the edge pixel

Rule 15: If I(i-1,j-1) AND I(i,j-1) AND I(i+1,j-1) are white and I(i-1,j) AND I(i-1,j+1) AND I(i,j+1) AND I(i+1,j+1) AND I(i+1,j) are black then (i,j) is the edge pixel

Rule 16: If I(i,j-1) AND I(i-1,j-1) AND I(i-1,j) AND I(i-1,j+1) AND I(i,j+1) are black and I(i+1,j-1) AND I(i+1,j) AND I(i+1,j+1) are white then (i,j) is the edge pixel

Rule 17: If I(i-1,j-1) AND I(i-1,j) AND I(i-1,j+1) AND I(i,j-1) AND I(i+1,j-1) are black and I(i,j+1) AND I(i+1,j) AND I(i+1,j+1) are white then (i,j) is the edge pixel

Rule 18: If I(i-1,j-1) AND I(i,j-1) AND I(i+1,j-1) AND I(i+1,j) AND I(i+1,j+1) are black and I(i-1,j) AND I(i-1,j+1) AND I(i,j+1) are white then (i,j) is the edge pixel

Rule 19: If I(i-1,j-1) AND I(i-1,j) AND I(i,j-1) are white and I(i,j+1) AND I(i+1,j-1) AND I(i+1,j) AND I(i+1,j+1) AND I(i-1,j+1) are black then (i,j) is the edge pixel

Rule 20: If I(i,j-1) AND I(i+1,j-1) AND I(i+1,j) are white and I(i-1,j-1) AND I(i-1,j) AND I(i-1,j+1) AND I(i,j+1) AND I(i+1,j+1) are black then (i,j) is the edge pixel

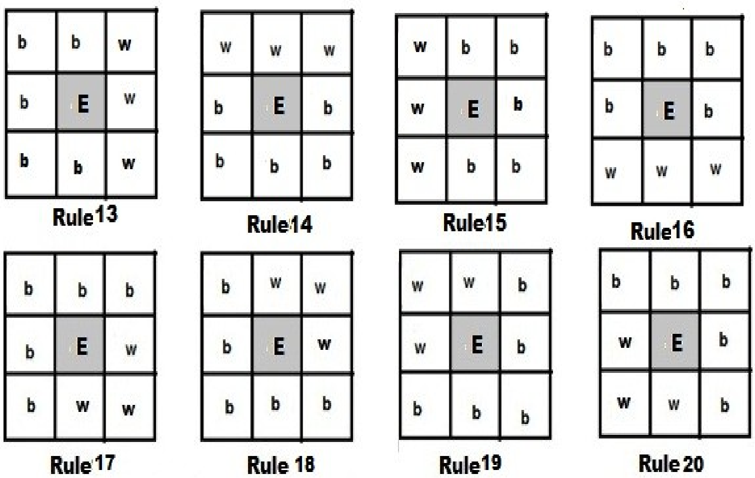


Fig. 4(c). Fuzzy Rule (Class 3 rules)

Class 4: Rules with 6 black & 2 white pixels in the neighborhood shown in Figure 4(d)

Rule 21: If I(i-1,j-1) AND I(i,j-1) AND I(i+1,j-1) AND I(i+1,j) AND I(i+1,j+1) AND I(i,j+1) are black and I(i-1,j) AND I(i-1,j+1) are white then (i,j) is the edge pixel

Rule 22: If I(i-1,j-1) AND I(i,j-1) are white and I(i-1,j) AND I(i,j+1) AND I(i+1,j-1) AND I(i+1,j) AND I(i+1,j+1) AND I(i-1,j+1) are black then (i,j) is the edge pixel

Rule 23: If I(i,j-1) AND I(i+1,j-1) are white and I(i-1,j-1) AND I(i-1,j) AND I(i-1,j+1) AND I(i,j+1) AND I(i+1,j+1) AND I(i+1,j) are black then (i,j) is the edge pixel

Rule 24: If I(i-1,j-1) AND I(i-1,j) AND I(i-1,j+1) AND I(i,j-1) AND I(i+1,j) AND I(i+1,j-1) are black and I(i,j+1) AND I(i+1,j+1) are white then (i,j) is the edge pixel

Rule 25: If I(i-1,j-1) AND I(i,j-1) AND I(i+1,j-1) AND I(i+1,j) AND I(i+1,j+1) AND I(i,j+1) are black and I(i-1,j) AND I(i-1,j+1) are white then (i,j) is the edge pixel

Rule 26: If I(i-1,j-1) AND I(i-1,j) are white and I(i,j+1) AND I(i,j-1) AND I(i+1,j-1) AND I(i+1,j) AND I(i+1,j+1) AND I(i-1,j+1) are black then (i,j) is the edge pixel

Rule 27: If I(i+1,j-1) AND I(i+1,j) are white and I(i-1,j-1) AND I(i-1,j) AND I(i,j-1) AND I(i-1,j+1) AND I(i,j+1) AND I(i+1,j+1) are black then (i,j) is the edge pixel

Rule 28: If I(i-1,j-1) AND I(i-1,j) AND I(i-1,j+1) AND I(i,j-1) AND I(i+1,j-1) are black and I(i+1,j) AND I(i+1,j+1) are white then (i,j) is the edge pixel

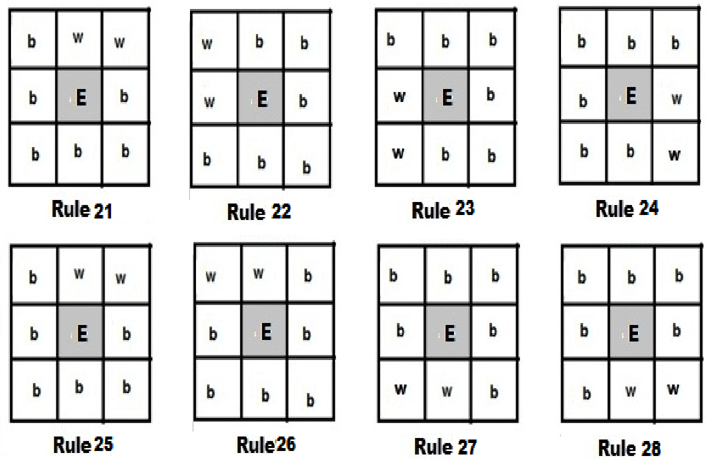


Fig. 4(d). Fuzzy Rule (Class 4 rules)

Class 5: Rules with all of the black and all white pixels in the neighborhood (to remove noise) shown in Figure 4(e)

Rule 29: If I(i,j-1) AND I(i+1,j-1) AND I(i-1,j-1) AND I(i,j+1) AND I(i-1,j) AND I(i-1,j+1) AND I(i,j+1) AND I(i+1,j+1) AND I(i+1,j) are black then (i,j) is non edge pixel

Rule 30: If I(i-1,j-1) AND I(i-1,j) AND I(i-1,j+1) AND I(i,j-1) AND I(i+1,j) AND I(i+1,j-1) AND I(i,j+1) AND I(i+1,j+1) are white then (i,j) is the non edge pixel

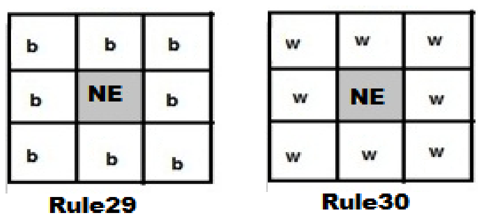


Fig. 4(e). Fuzzy Rule (Class 5 rules)

2.3 False Edge Removal

In our proposed method some false edges also emerge out along with the true edges. The reason for these false edges is explained as follows: there may be a case where there are six or more than six pixels having an intermediate range of intensity values in the 8-neighborhood of a pixel. Although the difference in the intensity of these pixels is very low, these pixels are still classified as being black or white because of the definition of membership functions, which gives rise to the false edges according to the defined fuzzy rules. Hence, to remove these false edges, we simply consider the pixel as being a non-edge pixel if six or more than six pixels in its neighborhood are in the intermediate intensity range. The chosen intermediate range is [117...137]. This range has been figured out via the conducting of experiments and is providing the best results.

2.4 Edge Improvement

Our proposed method is still not able to detect some of the edge pixels, due to the reasons listed below.

1) The pixels with intensity above 128 are classified as white pixels. The pixels with an intensity approaching 128 are also white and pixels approaching 255 are also white. For example, if the intensities of the 8-neighborhood pixels are 129, 137,131,205,128,211,230, and 245 where the intensity difference is very high still, they are all considered to be white pixels.

2) The pixels with intensity below 128 are classified as black pixels. The pixels with an intensity approaching 128 are also black and pixels approaching 0 are also black. For example, if the intensities of the 8-neighborhood pixels are 127, 97, 101, 5, 1, 11, 120, and124 where the intensity difference is very high still, they are all considered to be black pixels.

In this situation, the edge pixels remain undetected by the defined approach. When the 8-neighborhood of a pixel is satisfying any of the above conditions, a new membership function is defined and applied to determine the black and white membership for the pixel.

For the black set, the membership function is given as:

$$\mu'_{black}(x) = \begin{cases} 1 & x < 141 \\ \dfrac{255-x}{255-141} & 141 \le x \le 250 \end{cases} \tag{3}$$

For the white set, the membership function is given as:

$$\mu'_{white}(x) = \begin{cases} 0 & x < 141 \\ \dfrac{x-141}{250-141} & 141 \le x \le 250 \end{cases} \tag{4}$$

We once again locate the fuzzy set to which the pixel belongs according to the new membership function. The previously defined fuzzy rules are applied to find whether the pixel is an edge pixel or not. The improved results are shown Figures 6-11(f).

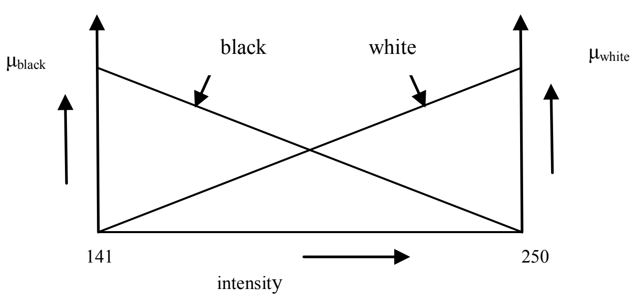


Fig. 5. The membership function for the intensity range [141…250]

3. THE ALGORITHM AND PSEUDOCODE

3.1 The Algorithm

The algorithm is comprised of the following steps:

Step 1: Input the gray scale image of size m x n.
Step 2: Scan the image pixel by pixel considering its 8-neighborhood.
Step 3: Count the number of the pixels in the 8-neighborhood of the scanned pixel in the range of 117-137, as denoted by Nint (the number of pixels in the intermediate range) and the number of the pixels in the 8-neighborhood of the scanned pixel in the range of 141-250, as denoted by Nwhite (the number of pixels in the white range).
Step 4: If Nint〈 6, the scanned pixel is a non-edge and go to Step 2.
Step 5: If Nwhite 〈8, find the membership of entire 8-neighboorhood pixel of the scanned pixel's membership by steps (1) and (2) for the black and white fuzzy set.
Step 6: Else, find the membership of the entire 8-neighboorhood pixel of the scanned pixel's membership with Equation (3) and (4) for the black and white set.
Step 7: Compare the black and white membership for each pixel and assign the fuzzy variable to the pixel whose membership is high.
Step 8: Apply the fuzzy rules to find whether the pixel is an edge pixel or not.
Step 9: If all of the pixels are not scanned, go to Step 2.
Step 10: End.

3.2 Pseudo-Code

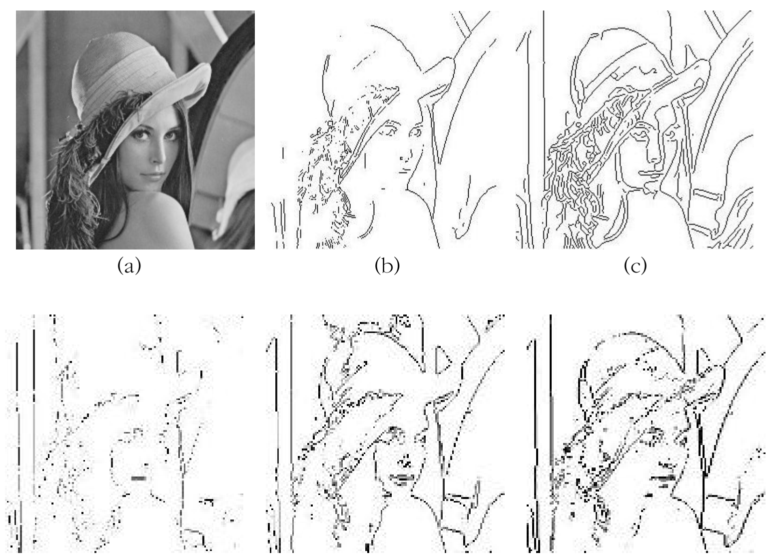
```
FuzzyEdge (Image Img)
// m & n are dimensions of image Img
For   i=1 to m;
For   j=1 to   n;
             For (x,y) in 8-Neighbourhood of   (x,y)
                    If (Img(x,y) 〉= 117   AND Img(x,y) =〈 137)
            Nint = Nint + 1;
                    End
Else If (Img(x,y) 〉= 140   AND Img(x,y) =〈250)
            Nwhite = Nwhite + 1;
                    End
             End for
             If ( Nwhite == 8)
                    For (x,y) in 8-neighbourhood of (i,j)
                             Compute μblack(x,y) and μwhite(x,y) using Equation 1.3 and 1.4, respectively.
                        End for
             Else
                    For (x,y) in 8-neighbourhood of (i,j)
                                    Compute μblack(x,y) and μwhite(x,y) using Equation 1.1 and 1.2, re-
                                    spectively.
                    End for
             End
             If Nint 〉= 6, pixel is non-edge
```

```
        Break;
    Else
        Apply the fuzzy rules to find whether (i, j) is an edge pixel or not
    End
End for
End for
```

4. Comparison with Other Techniques

We implemented our proposed method in MATLAB 9.7.0.471 and ran it on a Core 2 Duo, 2 GHz processor with 1.96 GB RAM for detecting the edge map in gray scale images. No pre-processing is required prior to the application of this algorithm.

To demonstrate the efficiency of our proposed approach, we carried out computer experiments on gray-level images. We selected a few standard images, which are "The Lena," "Baboon," "Cameraman," "Peppers," and "PillSet." The resolution of all the images was 8-bits per pixel. Along with the standard images, one of the simple test images used by A.Alshennawy[7] was also selected.We tested our proposed system with the above-mentioned images, and compared its performance to that of the classical operators (Sobel, Canny) and our proposed method. The edge maps of the images using the Sobel operator and Canny operator were found using the image processing toolbox in Matlab. For the images in Figs. 6–11, the captions are as follows: (a) The original image, (b) The results of the Sobel Edge Detector, (c) The results of the Canny Edge Detector, (d) The results of the A.Alshennawy *et al.*[7] Edge Detector, (e) The results of the proposed Edge Detector (f) The results with the modification mentioned in sections 2.3 and 2.4 in the proposed Edge Detector.



(a) (b) (c) (d) (e) (f)

Fig. 6. (a) Lena image, the edge map of the Lena image by (b) the Sobel operator (c) the Canny operator (d) A Alshennawy et al.[7] the Edge Detector(e) Proposed approach (f) and the modification in the proposed approach

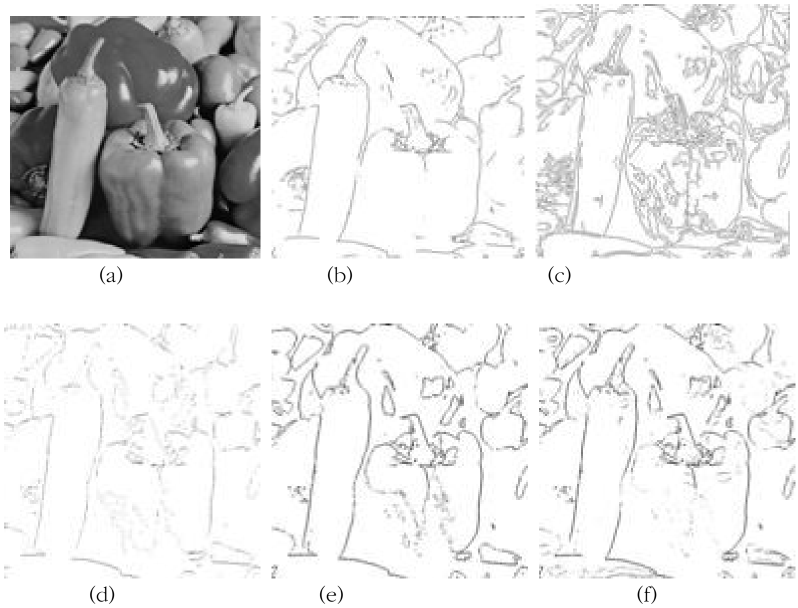


Fig. 7. (a) The Peppers image, the edge map of the peppers image by (b) the Sobel operator (c) the Canny operator (d) A Alshennawy et al.[7] the Edge Detector(e) Proposed approach (f) and the modification in the proposed approach

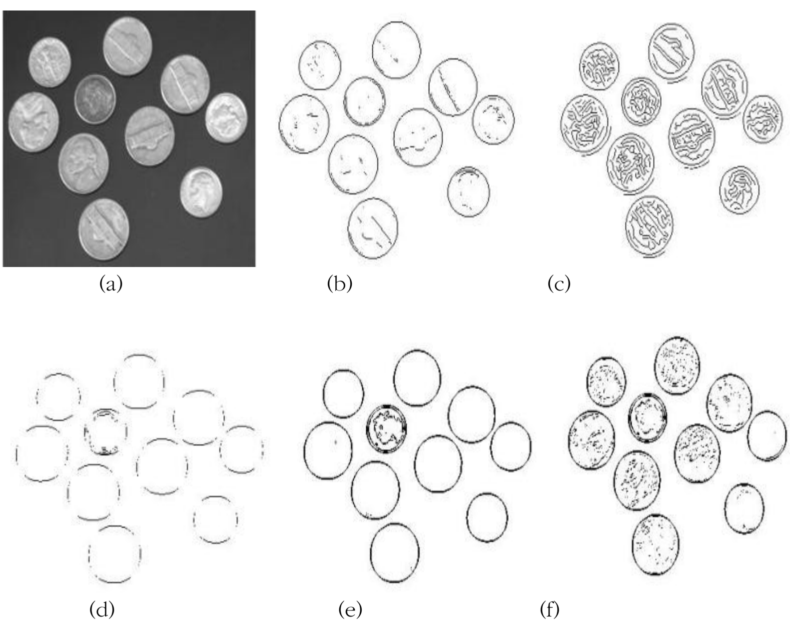


Fig. 8. (a) Coins image, the edge map of the Coins image by (b) the Sobel operator (c) the Canny operator (d) A Alshennawy et al.[7] the Edge Detector(e) Proposed approach (f) and the modification in the proposed approach

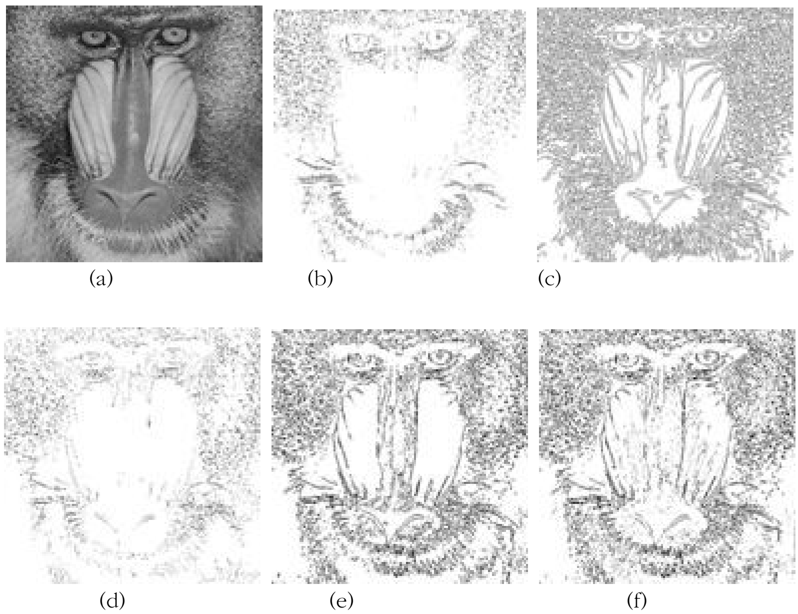


Fig. 9. (a) Baboon image, the edge map of the Baboon image by(b) the Sobel operator (c) the Canny operator (d) A Alshennawy et al.[7] the Edge Detector(e) Proposed approach (f) and the modification in the proposed approach

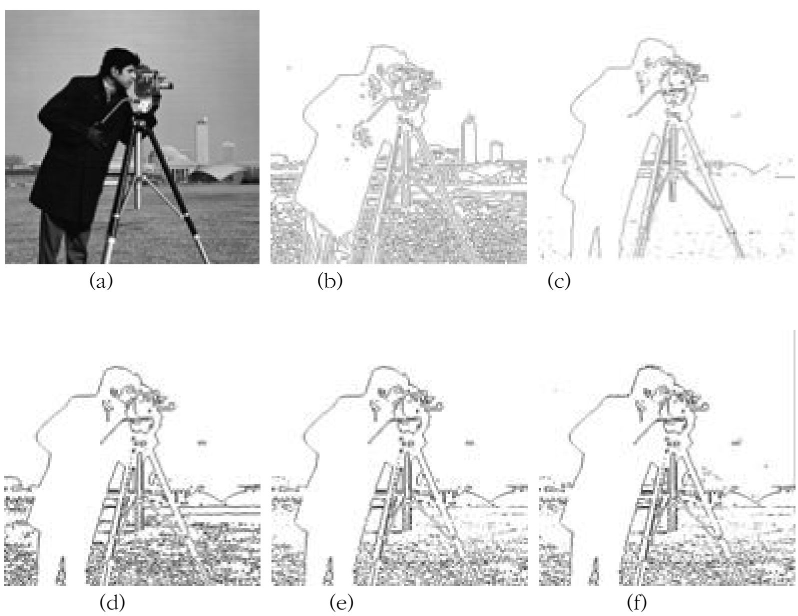


Fig. 10.(a) Cameraman image, the edge map of the Cameraman image by (b) the Sobel operator (c) the Canny operator (d) A Alshennawy et al.[7] the Edge Detector(e) Proposed approach (f) and the modification in the proposed approach

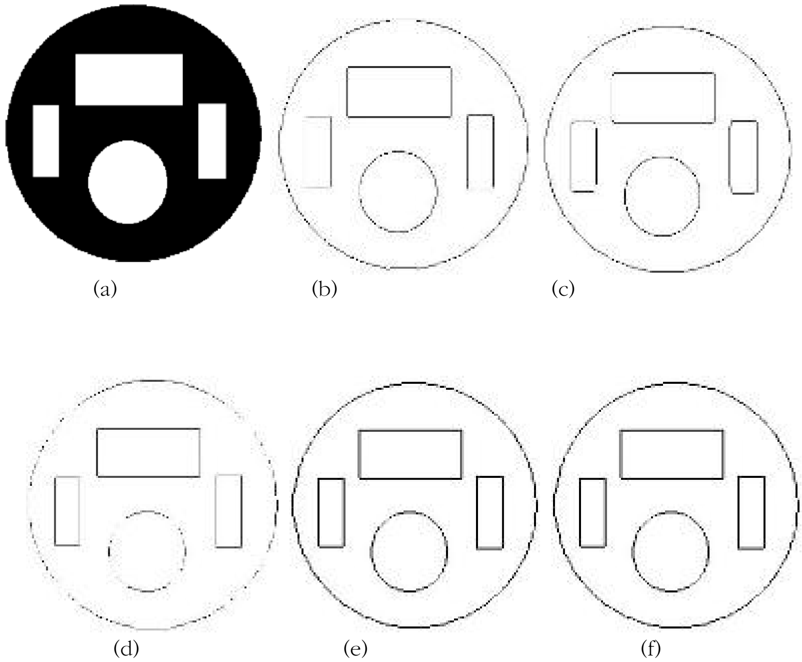


Fig. 11.(a) Test image, the edge map of the test image by (b) the Sobel operator (c) the Canny operator (d) A Alshennawy et al.[7] the Edge Detector(e) Proposed approach (f) and the modification in the proposed approach

To evaluate the performance of the method, the Shannon entropy value was calculated for each of the images. Shannon's entropy gives the indefiniteness in an image and is calculated as:

$$H(I) = -\sum_{i=0}^{L-1} p_i \log p_i$$

Where I stands for an image whose entropy is to be measured, p_i is the frequency of the occurrence pixel with intensity i, and L is the number of intensity levels in an image.

Table 1 shows the entropy values for the various test images by applying several edge detectors. The higher the value of the entropy, the higher the information content is. However, a very large entropy value reflects high noise content or double edges. The Canny edge detector produces double edges and therefore, the entropy values with this method are higher than those obtained with the proposed method for 3 out of 6 images. The higher entropy value with the proposed approach is due to the noise, but the proposed method gives significant results without the use of derivatives and thresholding and is also computationally simple. However, the other edge detectors, namely Sobel and A.Alshennawy *et al.* [7] give very less edge information. Therefore, the entropy values that use these methods are less than those for our method.

Table 1. The entropy values of different edge detectors for multiple images

Image	Sobel	Canny	A. Alshennawy *et al.*[7]	Proposed
Lena	0.6327	0.9730	0.3756	0.9629
Peppers	0.4365	0.8596	0.1461	0.6506
Coins	0.4821	0.9201	0.1239	0.7935
Baboon	0.7866	1.4009	0.1434	1.5563
Cameraman	0.4026	0.9245	0.5057	0.9551
Test Image	0.3300	0.3258	0.3474	0.5278

5. Conclusion

This paper provides a new methodology to detect edges. It is quite simple as compared to other edge detection methodology that is available so far. With the addition of more rules, our proposed method has significantly improved edge detection, as compared to the technique present in [7]. Our method not only detects edges in simple images but it also detects the edges in complex gray scale images.

In order to obtain more edges, the modification that we have suggested is also helpful. The range of intensity that the membership is to be redefined to can be discovered via experimentation, so as to get the maximum edges. But while adding more edges, there are chances of ending up with noise. Here, the intensity range [141…250] gives better results with more edges and lesser noise. This range has been discovered via the conducting of experiments.

Our proposed method does not use any derivative or complex calculation and is robust in finding edges by the use of simple fuzzy rules. However, it did detect some false edges too. The designed fuzzy rules are an attractive solution for improving the quality of the edges, as much as possible. Clearly, our proposed method provides better edge detection results than the previous methods do for illustrated images in a relatively less amount of time.

References

[1] R.C. Gonzalez, R.E. Woods, Digital Image Processing, Addison-Wesley, New York, 1992.

[2] J.F. Canny, 'A computational approach to edge detection', IEEE Trans. Pattern Analysis Mach. Intell. 8 (1986), pp 679–698.

[3] L. A. Zadeh, "Fuzzy Sets", Information and Control", 1965, pp. 338-353.

[4] Pal S K, King R A., 'On Edge detection of X-ray images using fuzzy sets', IEEE Trans. Patt. Anal. and Machine Intell. 1983, 5(1):69-77.

[5] Zhang Jinping; Lian Yongxiang; Dong Linfu; Zhao Xueguang, Liu Jie; "A New Method of Fuzzy Edge Detection Based On Gauss Function", Proceedings of 2nd International Conference on Computer and Automation Engineering, Singapore, Feb 26-28, 2010, pp. 559 – 562.

[6] T. Law, H. Itoh, H. Seki, Image filtering, edge detection, and edge tracing using fuzzy reasoning, IEEE Trans. Pattern Analysis Mach. Intell. 18 (1996) 481-491.

[7] Abdallah A. Alshennawy, and Ayman A. Aly 'Edge Detection in Digital Images Using Fuzzy Logic Technique', World Academy of Science, Engineering and Technology 51 2009, pp 178-186

[8] Begol, Moslem and Maghooli, Keivan, 'Improving Digital Image Edge Detection by Fuzzy Systems'

World Academy of Science, Engineering and Technology 81 2011, pp 76-79

[9] Er Kiranpreet Kaur, Er Vikram Mutenja, Er Inderjeet Singh Gill 'Fuzzy Logic Based Image Edge Detection Algorithm in MATLAB' 2010 International Journal of Computer Applications (0975 - 8887) Volume 1 – No. 22

[10] Lily Rui Liang, Carl G. Looney, 'Competitive fuzzy edge detection',Applied Soft Computing, Volume 3, Issue 2, September 2003, pp 123–137

[11] R. Wang, L. Gao, S. Yang, and Y. Liu, "An Edge detection method by combining fuzzy logic and neural networks," Machine Learning and Cybernetics, 7(2005), Guangzhou, China, 18-21 Aug 2005, pp 4539-4543.

[12] S. Sinaie, A. Ghanizadeh , and S.M. Shamsuddin , E.M. Majd 'A Hybrid Edge Detection Method Based on Fuzzy Set Theory and Cellular Learning Automata'2009 International Conference on Computational Science and Its Application, june 2009,Suwon ,Korea.

[13] Tizhoosh, H.R.; "Fast fuzzy edge detection," Fuzzy Information Processing Society, 2002. Proceedings. NAFIPS. 2002 Annual Meeting of the North American, vol., no., pp. 239- 242, 2002

[14] C.Naga Raju, O.Rama Devi, Sharada Mani, Sanam Nagendram, 'An Improved Ant Colony Optimization Technique by using Fuzzy Inference Rules for Image Classification and Analysis' International Journal of Advanced Engineering & Application, Jan. 2010.

[15] Khalid, N.E.A.; Manaf, M.; Aziz, M.E.," Fusion of Fuzzy Heuristic and Particle Swarm Optimization as an edge detector", Proceedings of International Conference on Information Retrieval & Knowledge Management, Shah Alam, Selangor, March 17-18, 2010, pp. 250 – 254.

[16] Om Prakash Verma, Madasu Hanmandlu, Puneet Kumar, Sidharth Chhabra, Akhil Jindal, A novel bacterial foraging technique for edge detection, Pattern Recognition Letters, Volume 32, Issue 8, 1 June 2011, pp 1187-1196.

Om Prakash Verma

He received his B.E. degree in Electronics and Communication Engineering from Malaviya National Institute of Technology, Jaipur, India, M. Tech. degree in Communication and Radar Engineering from Indian Institute of Technology (IIT), Delhi, India and Ph.D. degree from Delhi University. From 1992 to 1998 he was Assistant Professor in Department of ECE at Malaviya National Institute of Technology, Jaipur, India. He joined Department of Electronics & Communication Engineering, Delhi Technological University (formerly Delhi College of Engineering) as Associate Professor in 1998. Currently, he is Professor and Head, Department of Information Technology Delhi Technological University, Delhi India. He is also the author of more than 30 publications in both referred journals and international conferences. He has guided more than 35 M. Tech. students for their thesis and presently 5 research scholars are working under his supervision for their Ph.D. He has authored a book on Digital Signal Processing in 2003. His present research interest includes: Applied soft computing, Artificial intelligent, Evolutionary computing, Image Processing, Digital signal processing. He is also a Principal investigator of an Information Security Education Awareness project, sponsored by Department of Information Technology, Government of India. He has a started online admission process for B. Tech admissions at DTU in 2011. He is a regular reviewer of many International Journals like IEEE transaction, Elsevier, Springer etc. He has acted as program committee member and chaired session for many conferences.

Veni Jain

She received her B.Tech. degree in Computer Science from T.I.T.S Bhiwani, India in 2009 and M.Tech degree in Information Systems from Delhi Technological University, Delhi, India in 2012. Currently she is working as an Assistant Professor in Computer Engineering Department of Galgotias University, Greater Noida , U.P. ,India. Her research interests are in the area of Image Processing, color image enhancement and soft computing.

Rajni Gumber

She received her B.Tech. degree in Information Technology from Y.M.C,A, Institute of Engineering, Faridabad, India in 2009 and M.Tech degree in Information Systems from Delhi Technological University, Delhi, India in 2012. Her research interests are in the area of Image Processing, color image enhancement and soft computing.