

IMPROVING EDGE DETECTION OF DARK IMAGES USING FUZZY METHOD

Kalyan Kumar Jena¹, Sonali Routray², Anisur Rahman³

¹*Computer Science Engineering & Application, Indira Gandhi Institute of Technology,
Sarang, Odisha, (India)*

²*Computer Science & Engineering, College of Engineering & Technology,
Bhubaneswar, Odisha, (India)*

³*School of Computer Science & Engineering, National Institute of Science & Technology,
Berhampur, (India)*

ABSTRACT

Fuzzy Image processing is a widely used technique used in image processing and often gives better result in many such occasions as compared to the conventional image processing. Gray-scale images can be processed based on fuzzy topology and color images can be processed by regarding the color image as multi-dimensional gray-scale images. In this paper a new construction method for interval-valued fuzzy relations (interval-valued fuzzy images) from fuzzy relations (fuzzy images) by vicinity is presented. This construction method is based on the concepts of triangular norm (t-norm). We analyze the effect of using different t-norms on darker images. This method would be helpful in examining various darker images like x-ray or diagnostic sonography images. To compare the results, we also examine the result of some traditional methods of edge detection. Finally, the construction method is applied to images and the results are compared.

Keywords: *Image Processing, Fuzzy Mathematics*

I. INTRODUCTION

Edge detection is a fundamental tool in image processing. The current image edge detection methods are mainly differential operator technique and high-pass filtration. The widely used operators such as Sobel, Prewitt, Roberts and Laplacian are sensitive to noises and their anti-noise performances are poor. The Log and Canny edge detection operators which have been proposed use Gaussian function to smooth or do convolution are computationally very large. Fuzzy techniques have always been utilized as one of the modern methods in different processes. This paper mainly used sobel, prewitt operators to do edge detection on bitmap images and compare the results with the results of lower constructor laplacian of Gaussian method which is based on fuzzy systems[5,9]. It has been proved that the effect by using this fuzzy method to do edge detection is very good and it can highlight almost all the edges associated with an image[8]. Also this method detects the edges in low illumination areas.

II. THE PRINCIPLE OF EDGE DETECTION

The goal of edge detection is to mark the points in a digital image at which the luminous intensity changes sharply. Sharp changes in image properties usually reflect important events and changes in properties of the world. These include

- (i) discontinuities in depth,
- (ii) discontinuities in surface orientation,
- (iii) changes in material properties and
- (iv) Variations in scene illumination.

2.1 Edge Properties

Edges may be viewpoint dependent - these are edges that may change as the viewpoint changes, and typically reflect the geometry of the scene, objects occluding one another and so on, or may be viewpoint independent - these generally reflect properties of the viewed objects such as surface markings and surface shape. In two dimensions, and higher, the concept of perspective projection has to be considered.

A typical edge might be (for instance) the border between a block of red color and a block of yellow; in contrast a line can be a small number of pixels of a different color on an otherwise unchanging background. There will be one edge on each side of the line. Edges play quite an important role in many applications of image processing. During recent years, however, substantial (and successful) research has also been made on computer vision methods that do not explicitly rely on edge detection as a pre-processing.

2.2 Detecting an Edge

Taking an edge to be a change in intensity taking place over a number of pixels, edge detection algorithms generally compute a derivative of this intensity change. To simplify matters, we can consider the detection of an edge in one dimension. In this instance, our data can be a single line of pixel intensities. For instance, we can intuitively say that there should be an edge between the 4th and 5th pixels in the following 1-dimensional data:

5	7	6	4	152	148	149
---	---	---	---	-----	-----	-----

To firmly state a specific threshold on how large the intensity change between two neighboring pixels must be for us to say that there should be an edge between these pixels is, however, not always an easy problem. Indeed, this is one of the reasons why edge detection may be a non-trivial problem unless the objects in the scene are particularly simple and the illumination conditions can be well controlled.

2.3 Approaches to Edge Detection

There are many methods for edge detection [10], but most of them can be grouped into two categories, search-based and zero-crossing based. The search-based methods detect edges by first computing a measure of edge strength, usually a first-order derivative expression such as the gradient magnitude, and then searching for local directional maxima of the gradient magnitude using a computed estimate of the local orientation of the edge, usually the gradient direction. The zero-crossing based methods search for zero crossings in a second-order derivative expression computed from the image in order to find edges, usually the zero-crossings of the Laplacian or the zero-crossings of a non-linear differential expression, as will be described in the section on differential edge detection following below. As a pre-processing step to edge detection, a smoothing stage, typically Gaussian smoothing [7], is almost always applied.

The edge detection methods that have been published mainly differ in the types of smoothing filters that are applied and the way the measures of edge strength are computed [3]. As many edge detection methods rely on the computation of image gradients, they also differ in the types of filters used for computing gradient estimates in the x- and y-directions.

III. AN EDGE DETECTION MODEL BASED ON SOBEL OPERATOR

The Sobel operator [4] performs a 2-D spatial gradient measurement on an image and so emphasizes regions of high spatial frequency that correspond to edges. Typically it is used to find the approximate absolute gradient magnitude at each point in an input grayscale image [9,10].

In theory at least, the operator consists of a pair of 3×3 convolution kernels as shown in Figure 1. One kernel is simply the other rotated by 90°. This is very similar to the Roberts Cross operator.

$$\begin{matrix}
 \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} & & \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} \\
 \mathbf{G}_x & & \mathbf{G}_y
 \end{matrix}$$

Fig. 1: Sobel Edge Operator

These kernels are designed to respond maximally to edges running vertically and horizontally relative to the pixel grid, one kernel for each of the two perpendicular orientations. The kernels 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:

$$\sqrt{G_x^2 + G_y^2}$$

Typically, an approximate magnitude is computed using:

$$G = |G_x| + |G_y|$$

This is much faster to compute.

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

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

In this case, orientation θ 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 anti-clockwise 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 2.

P_1	P_2	P_3
P_4	P_5	P_6
P_7	P_8	P_9

Fig.2:Pseudo-Convolution Kernels Used To Quickly Compute Approximate Gradient Magnitude

Using this kernel the approximate magnitude is given by:

$$|G| = |(P_1 + 2XP_2 + P_3) - (P_7 + 2XP_6 + P_4)| + |(P_3 + 2XP_6 + P_9) - (P_1 + 2XP_4 + P_7)|$$

IV. AN EDGE DETECTION MODEL BASED ON PREWITT OPERATOR

The Prewitt operator [4, 10] 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 direction 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 is named for Judith Prewitt.

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 **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:

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

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

.Since the Prewitt kernels can be decomposed as the products of an averaging and a differentiation kernel, they compute the gradient with smoothing. For example, **G_x** can be written as

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

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:

$$\sqrt{G_x^2 + G_y^2}$$

Using this information, we can also calculate the gradient's direction:

$$\Theta = \text{atan2}(G_y, G_x)$$

where, for example, Θ is 0 for a vertical edge which is darker on the right side.

V. PRINCIPLE OF MIN CONSTRUCTOR WITH LAPLACIAN OF GAUSSIAN METHOD

5.1 Min Construction Method

The *lower constructor* is a generalization of the *t*-processing

A *t*-norm $T: [0, 1]^2 \rightarrow [0, 1]$ is an associative, commutative, increasing function, such that, $T(1, x) = x$ for all $x \in [0, 1]$. A *t*-norm T is called idempotent if $T(x, x) = x$ for all $x \in [0, 1]$.

The four basic *t*-norms are as follows

1. The minimum $T_M(x,y)=\min(x,y)$.
2. The product $T_P(x, y) = x \cdot y$.
3. The Łukasiewicz *t*-norm

$$T_L(x, y) = \max(x + y - 1, 0).$$

4. The nilpotent minimum t-norm

$$T_{nM}(x, y) = \min(x, y), \text{ if } x + y > 1$$

$$0, \text{ otherwise.}$$

- Let $R \in F(X \times Y)$ be an FR. Consider two t-norms T_1 and T_2 and two values $n, m \in \mathbb{N}$ so that $n \leq P - 1/2$, and $m \leq Q - 1/2$. We define the lower constructor associated with T_1, T_2, n , and m in the following way:

- $L^{n,m}T_1, T_2 : F(X \times Y) \rightarrow$ given by

$$L^{n,m}T_1, T_2 [R](x, y) =$$

$$T_1^{m,n}(T_2 (R(x - i, y - j), R(x, y)))$$

$$i = -n$$

$$j = -m$$

$$\text{for all } (x, y) \in (X, Y)$$

The Algorithm begins with reading an $M \times N$ image. The first set of nine pixels of a 3×3 window are chosen with central pixel having values (2,2) i.e for each pixel (i, j) we are taking the 8 neighbourhood of (i, j) . After the initialization, the pixel values are initially marked as edge pixel after an observation to the 8 neighbourhood. After the subsection of the pixel values the algorithm generates an intermediate image using a construction method stated below. It is checked whether all pixels have been checked or now, if not then first the horizontal coordinate pixels are checked. If all horizontal pixels have been checked the vertical pixels are checked else the horizontal pixel is incremented to retrieve the next set of pixels of a window. In this manner the window shifts and checks all the pixels in one horizontal line then increments to check the next vertical location.

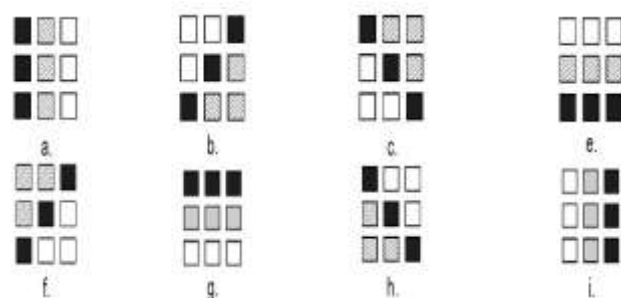


Figure3: Initial Conditions to Check 8 Neighborhood

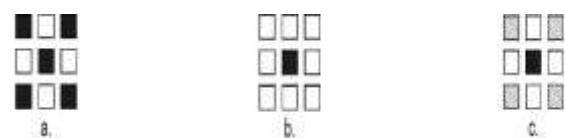



Fig.4: (A,B) Type Of Unwanted Edge Pixels (C) Condition For Removal Of Unwanted Edge Pixels.

 Pixel checked

 Edge pixel

 Unchecked
pixel

After edge highlighting image is subjected to another set of condition with the help of which the unwanted parts of the output image of type shown in Fig.(a-b) are removed to generate an image which contains only the edges associated with the input image. Let us now consider the case of the fuzzy condition displayed in Fig. (g). For an input image A and an output image B of size M x N pixels respectively we have the following set of conditions that are implemented to detect the edges pixel values.

Input: An image A of M x N pixels (Phase 1)

Output: An image B of M x N pixels

Initial Edge Detection (A, B) using Min Construction

For I←2 to M-1

For J←2to N-1

If A (I-1, J)>A (I-1, J+1)

Then If A (I-1, J-1)>A (I, J)

Then If A (I, J-1)>A (I+1, J-1)

Then

B (I-1, J+1) ←0

B (I, J) ←0

B (I+1, J-1) ←0

End For

End For

For I←2 to M-1

For J←2to N-1

If B(I-1,J)=255& B(I,J)=0& B(I+1,J)=255& B(I,J-1)=255

Then B (I, J) is minimum and highlighted as edge initially.

End For

End For

In the above algorithm Min construction[1] is used but not after fuzzification as after fuzzification the membership values would become fractions that can't be stored in unsigned char. Hence the same technique of min construction is used but on true picture and taking into consideration 8-nbd of a pixel (i,j).

We can observe in the above algorithm written for a particular fuzzy condition that the nesting of statements is done in a manner that only the edge associated pixels are granted black pixel values and initially min valued edge pixels are given white value. These pixels are initially marked as edge.

Phase 2. Input: An image B (256 color true bmp image) of size MxN

Output: Edge image of size MxN

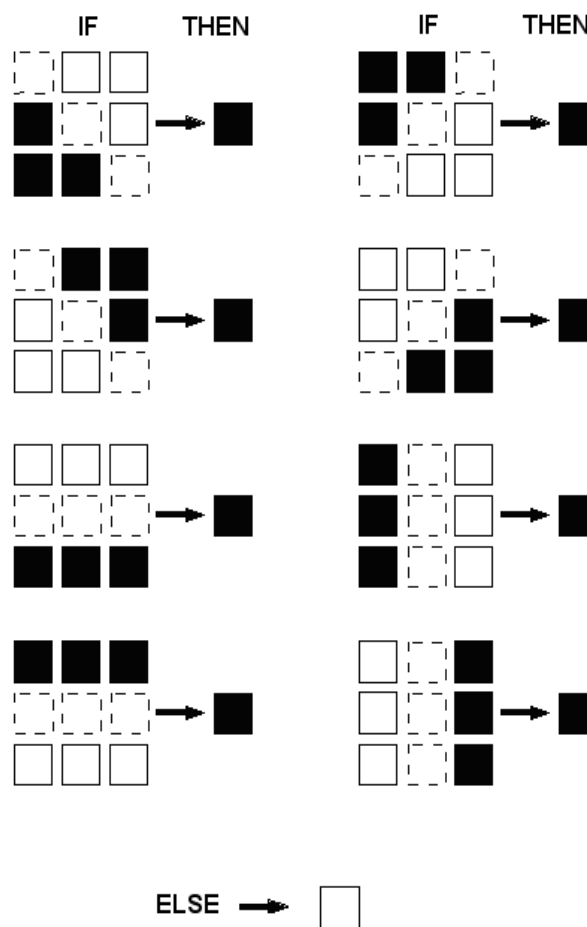
We now use Laplacian of Gaussian (log) operator[7] on the intermediate image to get the edge image. And In this way whatever image is being constructed is compared with edges found on same image by other existing techniques.

LOG operator is

$$\begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 2 & 1 & 0 \\ 1 & 2 & -16 & 2 & 1 \\ 0 & 1 & 2 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

which is stored in a 5x5 array.

The phase 1 actually performs a check like



Black pixels are having min values.

VI. EXPERIMENTAL RESULTS AND ANALYSIS

This paper will use a BitMap image as the original image. First, use the traditional edge detection operators (include Sobel operator, Prewitt operator, Laplacian operator and Canny operator) to do edge detection. Then use the min constructor laplacian with Gaussian operator[6] which is based on fuzzy systems to do the edge detection.

Use the Bitmap image as the original image. First, use the traditional edge detection operators to do edge detection, and the results are showed in Figure 5,6,7:



Fig.5: The original Bitmap image



Fig. 6: Edge detection using sobel operator



Fig. 7: Edge detection using Prewitt operator

Here we are performed edge detection using two techniques i.e. sobel and prewitt edge detection [5,7,9] methods and outputs are displayed for the corresponding input images which are shown in the figure5 and figure 6 . Though some portions of edges are clearly visible but it is not focusing more to the lower intensity regions. So that the edges of lower intensity areas not so clearly visible.

In order to overcome this defect, the paper combines Laplacian of Gaussian operator and min construction method, see in Figure 8:



Fig. 8 : Edge Detection Using Min Construction With Laplacian Of Gaussian Operator

Figure 8 shows the output of min construction with laplacian of Gaussian operator. The laplacian of Gaussian essentially act as a band pass filter because of its differential and smoothing behavior

Again the Gaussian is separable which make computation very efficient. Therefore, it plays a certain role in smoothing the image and min construction helps in detecting the edges in lower illumination areas.

VII. CONCLUSION

In this paper, the algorithm to find the edges associated with an image had been implemented. Comparison were made amongst the various other edge detection algorithms that have already been developed and displayed the accuracy of the edge detection using Min construction with laplacian of Gaussian method. Min construction with laplacian operator highlights portion of the image with lower contrast. Using sobel operator the edge is clearly visible but slightly differs from output of prewitt operator. The output of this traditional edge detectors shows thin and sharpen edges and ignores many region of interests that is being taken care by min construction with laplacian operator.

REFERENCES

- [1] Edurne Barrenechea and Humberto Bustince, Construction of intervalued fuzzy relation with application to fuzzy edge images, IEEE transactions on fuzzy systems, vol. 19, no. 5, October 2011.
- [2] Rafael C. Gonzalez and Richard E. Woods, Digital Image Processing (2nd Edition, Prentice Hall, ISBN: 0201180758.)
- [3] Zhengquan He and M.Y .Siyal ,An IMAGE detection technique based on morphological edge detection and background differencing for real-time traffic analysis, Elsevier Science Inc., New York, NY, USA.
- [4] Renyan Zhang, Guoling Zhao and Li Su.,New edge detection method in image processing, College of Autom., Harbin Engineering University, China.
- [5] Stamatia Giannarou and ania Stataki.,Edge Detection Using Quantitative Combination, IEEE transactions on signal processing system and implementation,November 2000, 359-364.
- [6] Hamid R. Tizhoosh,Fuzzy Image Processing, University of Waterloo,june 1997.
- [7] M. Basu, Gaussian-based edge-detection methods: A survey, IEEE Trans. Syst.,Man, Cybern. C, Appl. Rev., vol. 32, no. 3, Aug.2002,252–260.
- [8] J. C. Bezdek, J. Keller, R. Krisnapuram, and N. R. Pal, Fuzzy Models and Algorithms for Pattern Recognition and Image Processing. Boston, MA: Kluwer, 1999.
- [9] B. Chanda and D. Dutta Majumdar ,Digital Image Processing and Analysis (Prentice Hall International , 2000)
- [10] Bernd Jähne ,Digital Image Processing,5th revised and extended edition Springer engineering online library.