AN ENHANCED BLOCK BASED EDGE DETECTION TECHNIQUE USING HYSTERESIS THRESHOLDING

Jayasree M[1] , N K Narayanan[2], Kabeer V[3] and Arun C R[4]

[1]Dept. of Computer Science & Engineering, Govt. Engineering College, Thrissur, India
[2]College of Engineering, Vadakara, India
[3]Department of Computer Science, Farook College, Kozhikode, India
[4]Dept. of Electronics & Comm. Engineering, Model Engineering College, Kochi, India

ABSTRACT

Edge detection is a crucial step in various image processing systems like computer vision , pattern recognition and feature extraction. The Canny edge detection algorithm even though exhibits high accuracy, is computationally more complex compared to other edge detection techniques. A block based distributed edge detection technique is presented in this paper, which adaptively finds the thresholds for edge detection depending on block type and the distribution of gradients in each block. A novel method of computation of high threshold has been proposed in this paper. Block-based hysteresis thresholds are computed using a non uniform gradient magnitude histogram. The algorithm exhibits remarkably high edge detection accuracy, scalability and significantly reduced computational time. Pratt's Figure of Merit quantifies the accuracy of the edge detector, which showed better values than that of original Canny and distributed Canny edge detector for benchmark dataset. The method detected all visually prominent edges for diverse block size.

KEYWORDS

Canny edge detection, edge detection, gradient magnitude histogram, hysteresis threshold.

1. INTRODUCTION

Edge detection is an elementary step in pattern recognition, computer vision and image processing in general and in the fields of feature extraction in particular. Important elements like corners, lines and curves can be obtained from the edges of a figure. Further, these features could be utilized in computer vision algorithms for object tracking and recognition. An efficient edge detection technique should exhibit good detection, localization and single response. Good detection factor shows that the algorithm notes as many genuine edges as possible. Localization indicates how close the marked edges are to those in the real image. Single response means that there should be only one marked edge to a true edge and image noise should not create non-existent edges [1]. Conceptually, edge detection involves three operations, namely smoothing, differentiation and labelling [2]. Smoothing an image consist of removing the noises and standardizing the numerical differentiation. Differentiation comprises of assessing the required image derivatives. Labelling refers to classifying a pixel as an edge or a non-edge pixel.

A novel technique for implementing hexagonal image processing operators, utilizing text functions and linear basis was introduced by Gardiner B et al [3]. The edge maps are constructed

DOI : 10.5121/sipij.2018.9202

at various scales by reuse of seven point linear operators and detailed analysis results were presented. Piotr Dollar and C. Lawrence Zitnick [4] has presented a method in which the image patch structure is utilized to train the accurate and efficient edge detector. This method maps the structured labels with a discrete space which forms the basis for the measure standard information gain for evaluation. The method showed competing results compared to the existing approaches. Patrica Merlin et al. has introduced an edge detection technique using the morphological gradient method, generalized type-2 fuzzy logic and theory of alpha planes [5]. The heights and approximation techniques are used for defuzzification, and the method exhibited comparable results. The problem caused due to uneven illumination in edge detection from digital images was solved by a method introduced by Anand Swaminathan and Santha Ramapachiyam[6], incorporating weighted least square regularization with the wavelet filter directional responses, and the method performed better than many existing techniques.

A novel technique for edge detection from digital images based on adjacent dispersion is proposed by Qindong Sun et al., in which adjacent dispersions are made use of for detecting edges [7]. A novel instantaneous technique for edge detection and denoising called Rided-2D, which is invariant to the geometric transformations, was proposed by Jian Wang et al. [8]. A parameter-free real-time edge detection technique was proposed by Cuneyt Akinlar, which works based on the edge drawing algorithm and delivers only meaningful edge segments after execution [9].

Among the various edge detection algorithms, Canny edge detection technique is accepted as the standard one [10], since it shows excellent performance and resistance to streaking [11]. Streaking is the breakup of edge contours occurring due to the operator output fluctuating above and below the threshold due to noise, along the length of the contour. To solve this problem, canny algorithm has implemented thresholding with hysteresis, in which the two thresholds are computed depending on the whole image information [11]. If any portion of an edge falls above the threshold, those points are immediately marked as edge points. The points in between low and high thresholds are selected only if they are part of the edge already marked above the high threshold. The chance of streaking is very much lowered now since for an edge to be broken, it should fluctuate above the upper threshold and below the lower threshold.

Numerous implementations of Canny algorithm have been realized on various hardware platforms. Typical studies were carried out on Deriche edge detectors which are based on Canny's criteria [12] – [14]. Extensive research has been carried out on Canny edge detection algorithm and its extensions [15] – [18]. In [14], a self-adaptive threshold based Canny edge detection technique is introduced to a mobile robotic system. The findings of the studies of general-purpose graphics processing unit (GPGPU) based Canny edge detection had been published in [10], [19], [20]. However, most of these existing implementations (e.g., [15] – [18], [10], [19], [20]) makes use of the same values for high and low thresholds for the whole input image frame. Other implementations discussed earlier (example. [18]) have used other thresholding techniques like adaptive thresholding.

Since Canny edge detection technique finds upper and lower thresholds considering the whole image information, its computational complexity is more compared to other edge detection techniques and necessitates added pre-processing steps to be performed on the whole image. Also, since Canny edge detection technique works at the frame level of an image, it takes more computational time, which makes real-time applications more time consuming. To solve these problems, Qian Xu et al., has proposed a technique in which the image is divided into numerous subblocks [21]. Applying the original canny algorithm directly to each block fails since it causes excessive edges in smooth regions where edges need not be marked, and dropping of essential edges in highly dense areas. This is because Canny edge detection method calculates the upper and lower thresholds depending on the entire image data. In order to resolve this problem, Qian

Xu et al. has introduced a distributed approach to edge detection in which the image is subdivided into blocks and the edge detection thresholds are dynamically computed depending on the block category and the distribution of gradients in each block. To compute block based hysteresis thresholds, a non-uniform gradient magnitude histogram is used. To further improve the edge detection accuracy, a new method has been proposed in this article, which computes high threshold in each block as the average of the maximum gradient magnitude and the threshold factor. This method supports quick edge detection from high resolution images and video frames more accurately. Quantitative performance evaluations proved that the proposed edge detection technique exhibits excellent performance over conventional frame based method and the distributed Canny algorithm.

The remaining part of this paper is organized as follows. The detailed description of the classical Canny Edge Detection Technique is presented in section 2. The proposed Edge Detection method is elaborated in section 3 and the results of simulation experiments are presented in section 4. Finally, conclusions are drawn in section 5.

2. CLASSICAL CANNY EDGE DETECTION TECHNIQUE

John F. Canny developed the Classical Canny Edge Detection Technique in 1986 [11]. Canny aimed at finding an ideal edge detection algorithm that exhibits good detection and minimal response to noise. Canny edge detection algorithm consists of the following five steps. (1) Noise removal (2) Intensity gradient Computation (3) Non-maximal suppression (4) Computation of thresholds and (5) Tracing edges through hysteresis thresholding.

2.1 Noise removal

Since raw, unprocessed image data contain noises, Canny edge detector utilizes a Gaussian filter to remove noise. The filtered image is a little hazier than the original image, but is liberated from noises.

2.2 Intensity Gradient Computation

An edge that occurs in an image may be in any orientation or direction. So the Canny algorithm uses four filter masks to identify vertical, horizontal and the two diagonal edges in the filtered image. The Sobel operator for edge detection returns values for the first order differentials in the horizontal and vertical direction. From these derivatives, the horizontal and vertical gradients along with direction could be computed. The gradient direction angle is approximated to any one of the four angles which correspond to horizontal, vertical, and two diagonals (which are 0, 90, 45 and 135 respectively). The gradient magnitude |G| and gradient angle Θ are calculated as

$$|G| = \sqrt{{G_x}^2 + {G_y}^2} \quad \text{and} \quad \Theta = \arctan\, G_y / G_x$$

$$\text{Where } G_x = \frac{\partial f}{\partial x} \quad ; \quad G_y = \frac{\partial f}{\partial y}$$

G_x and G_y represent the horizontal and vertical gradients respectively, and f = f(x,y) represents the image pixel. The gradient magnitude can be obtained by convolving the image with the horizontal and vertical gradient mask. Figure 1 shows the convolution mask using the Sobel operator to find the gradient magnitude.

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

Gx

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

Gy

Figure 1. Sobel operator used to find the horizontal and vertical gradients

2.3 Non-maximal suppression

Once the gradients of the image are obtained, the next step is to learn whether the gradient magnitude reaches local maxima along the direction of the gradient. If the approximated gradient angle is zero degrees, the pixel can be treated as an edge pixel in the horizontal axis, if its gradient magnitude falls above the magnitudes of pixels in the north – south directions (since the gradient angle is normal to the edge) [22]. Similarly, if the approximated gradient angle is 90 degrees, the pixel will be treated as an edge pixel if its gradient magnitude falls above the magnitudes of pixels in the east-west directions. If the approximated gradient angle is 135 degrees, the pixel will be treated as an edge pixel its gradient magnitude falls above the magnitudes of pixels in the south west and north east directions. If the approximated gradient angle is 45 degrees, the pixel will be treated as an edge pixel if its gradient magnitude falls above the magnitudes of pixels in the north west and south east directions. At this stage, a set of binary edge points called thin edges are obtained. The range of angles of the edge normal for the edge directions of four types in a 3 x 3 neighborhood is shown in Figure 2 [22].

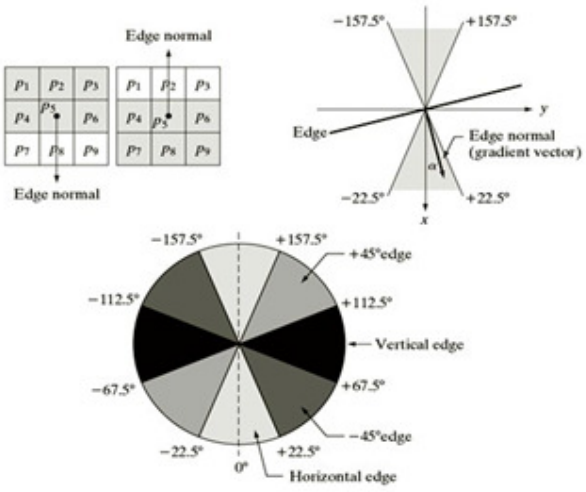

Figure. 2. The angle ranges of the edge normals for four edge directions (Reproduced as in [22])

2.4 Computation of Thresholds

Larger intensity gradients have greater probability to correspond to an edge than a smaller intensity gradient. Generally, it is not possible to fix a threshold value at which a given intensity gradient change from one that corresponds to an edge to that of a non-edge. Therefore Canny edge detector uses thresholding with hysteresis which needs two thresholds (low and high). These thresholds are calculated based on the histogram of the gradient magnitude of the whole image. The higher threshold is computed such that a percentage P1 of the total pixels in the image would be categorized as strong edges, which can be added directly to the final edge image. A high threshold is the point at which the value of the gradient magnitude cumulative distribution function (CDF) equals to (1-P1). Low threshold is calculated as the percentage P2 of the high threshold. P1 and P2 are set at 0. 2 and 0.4 respectively, for optimal results [11], [23].

2.5 Tracing edges through hysteresis thresholding

A pixel is categorized as a strong edge pixel, if its gradient magnitude is larger than upper threshold. A pixel is labeled as weak edge, if the value of gradient magnitude of the pixel falls between the lower and higher threshold. Strong edges are the ones which can be included immediately as edges in the final edge image. Weak edges can be marked only if they are linked to the strong edges. The block diagram in Figure 3. shows the stages of canny edge detection algorithm.

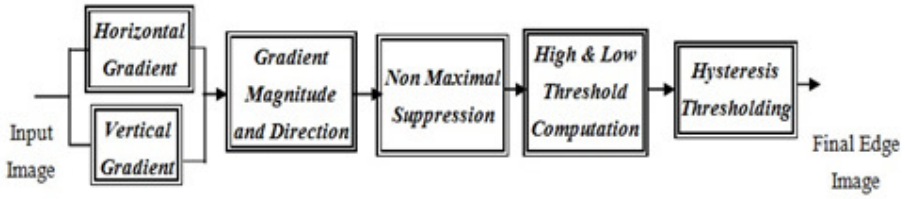

Figure 3. Block Diagram of Original Canny Edge Detection Technique

3. PROPOSED EDGE DETECTION ALGORITHM

The Proposed Edge Detection Algorithm introduces a novel technique to compute the high threshold. In the proposed method, the input image is being subdivided into a number of overlapping blocks of size m × m, where the blocks are processed without dependence with each other. The local variance of each pixel within a 3 × 3 window, centered at the current pixel is used to categorize each pixel as uniform, edge, or texture pixel. Then the blocks are categorized into five types viz. smooth, texture, edge/texture, medium edge, and strong edge block, using the block classification method, which considers the total number of uniform, edge and texture pixels in the particular block [24].

First three steps and step 5 of the proposed algorithm are same as that of the original frame based Canny algorithm and the only difference is that they are performed at the block level. Figure. 4 show the schematic block diagram of the proposed edge detection system. The block division method is detailed in section II.A and the pixel classification procedure is given in section 2.2. The block classification method is elaborated in section 2.3. The threshold calculation method proposed in this article computes the high threshold value based on gradient magnitude features. 40 % of the high threshold is set as low threshold.

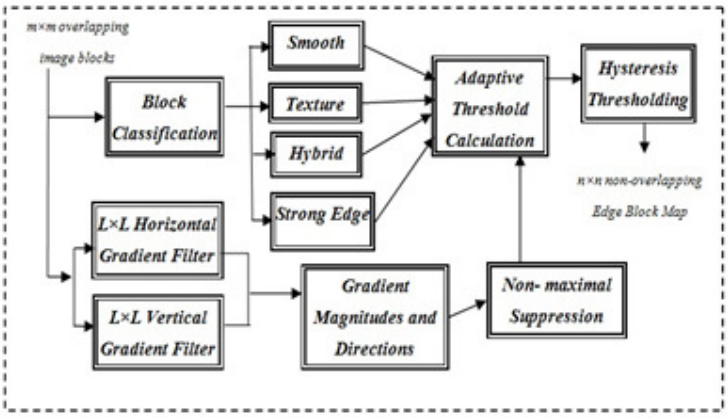

Figure 4. Proposed distributed edge detection technique

3.1 Block Division

In the proposed technique, the input image needs to be subdivided into m×m overlapping blocks where these blocks are handled without dependence with each other. Consider a gradient mask of size L×L. The m×m overlapping blocks are created by subdividing the input image into n × n non-intersecting blocks first and then by padding every block by (L+1)/2 pixels through the left, right, top, and bottom boundaries. The result is m×m overlapping blocks, where m = (n+L+1). The n × n blocks which are non-overlapping is padded with zeros to get rid of edge artifacts and failure to detect edges at block boundaries while computing the gradients. Furthermore the non-maximal suppression action performed on boundary pixels requires gradients of the neighboring pixels in a block. Here, we have considered n=64, L=3 so, m=68. Figure 5(b) shows the block division result of the cameraman image in Figure. 5 (a)

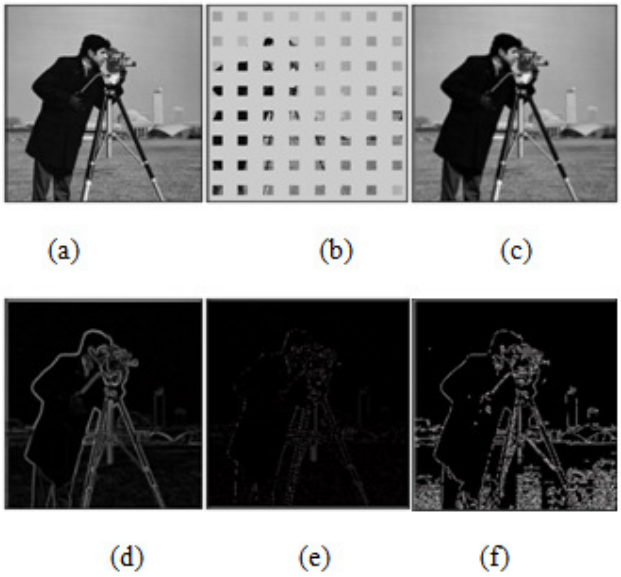

Figure. 5. The result of applying the proposed algorithm (a) Cameraman image as input (b) After Block Division (c) After smoothing each block individually (d) Gradient Magnitude of the image (e) After Non-Maximal Suppression (f) After Hysteresis Thresholding

3.2 Pixel Classification

Now, classify each pixel in a block into three types which are uniform, texture and edge pixels using the pixel classification method [24]. The local variance of every pixel within a 3 × 3 window, centered on the pixel under consideration is utilized so as to classify each pixel as edge, texture or uniform pixel type. The pixel classification strategy is given below.

$$\text{Pixel type} = \begin{cases} \text{Uniform}, & \text{var}(x, y) \leq T_u \\ \text{Texture} & T_u < \text{var}(x, y) \leq T_e \\ \text{Edge} & T_e < \text{var}(x, y) \end{cases}$$

Here var(x, y) of a pixel at location (x,y) is the local variance of the pixel in the (3x3) window centered at the pixel at (x,y). T_u & T_e are two thresholds, which are set to 100 and 900 respectively, for optimum result [21] [24].

3.3 Block Classification

After classifying each pixel, the next step is block classification. Blocks are categorized depending on the aggregate number of edge, texture, and uniform pixels in the block [24]. The block classification strategy is given in Table 1 where $N_{uniform}$ is the total number of uniform pixels in the block, N_{edge} is the total number of edge pixels in the block and Total_pix is the total number of pixels in the block.

Table 1. Block Classification Strategy

Block Type	**No of pixels of uniform pixel type $N_{uniform}$**	**No of pixels of Edge Pixel Type N_{edge}**
Smooth	>=(0.30 x Total_pix)	0
Texture	<(0.30 x Total_pix)	0
Edge/Texture	<0.65 x (Total_pix - N_{edge})	(>0) & (<0.30 x Total_pix)
Medium	>=0.65 x (Total_pix-N_{edge})	(>0) & (< 0.30 x Total_pix)
Strong	<=(0.7 x Total_pix)	>=0.30 x Total_pix

Now, the first three steps of Canny edge detection technique are performed on each block sequentially. Figure. 5(c) to (e) presents the visual results of these steps performed on the cameraman image. After performing these steps, the proposed edge detection technique computes the thresholds, using a finely tuned non-uniform quantiser, the details of which are explained in the next section.

3.4 Threshold Computation using non-uniform quantiser

Since hysteresis thresholding is adopted for edge detection, two thresholds (high and low) need to be computed. A finely quantized gradient magnitude histogram is used to find the high and low hysteresis thresholds. For calculating the high threshold, a non-uniform quantizer is implemented [20] which is shown in Figure 6. For the medium edge, strong edge and edge/texture block, the greatest peak in the gradient magnitude histogram of the input image falls near origin, which shows that the low intensity pixels form the maximum in number, whereas edge pixels form a series of smaller peaks, in which each peak represents edges having similar gradient magnitudes [18], [25]. So, the high threshold needs to be selected between the highest and the second highest peak, which would be an edge peak. The quantizer should have more quantization levels in the area between the highest peak and the second highest peak and fewer levels in other parts. The highest and lowest values of the gradient magnitudes in the block considered are averaged to get the first reconstruction level (R_1). The second reconstruction level (R_2) is obtained as the mean of the first reconstruction level and the least value of the gradient magnitude. Similarly, n reconstruction levels are computed. The corresponding equations are given below.

$R_1 = (least + most) /2$

$R_{i+1} = (least + R_i) /2$ for (i = 1,2, 3, . . . 7)

Where least and most are the lowest and highest values of the gradient magnitudes, respectively. Here the histogram is divided into 8 reconstruction level (R_1 to R8). Now, compare total number of pixels in each reconstruction level Ri (NoPixels_Ri) with the product of P1 value and the total number of pixels in the block (NoPixels_P1) to choose the Reconstruction level i, where NoPixels_Ri is nearest to the NoPixels_P1. Using the selected level Ri and highest gradient magnitude, mag_most, the high threshold ThHigh is computed. The low threshold ThLow is calculated as 40% of the high threshold [21]. Here the selected reconstruction level Ri is

considered as threshold factor Thresh_Factor. The proposed novel computation of high threshold is given below. Low threshold calculation has been adopted from [25].

ThHigh= ((mag_most + Thresh_Factor)/2)

ThLow = ThHigh x 0.4

Finally, hysteresis thresholding is performed on each block using these calculated high and low thresholds, ThHigh and ThLow respectively to obtain the required edge image. Figure 5(f) shows the final image after performing hysteresis thresholding.

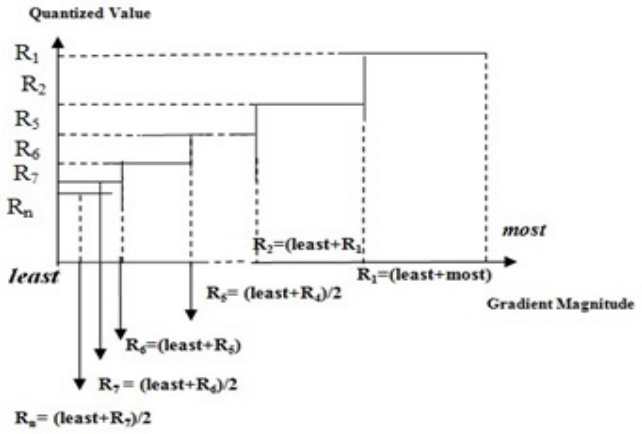

Figure 6. Reconstruction values and quantization levels

4. RESULTS

To evaluate the performance of the proposed edge detection technique, experimental results with test images from standard dataset are presented. Datasets used are Berkeley Segmentation Dataset [26], Kodak Dataset [27] and a dataset created by authors using mobile camera for testing purpose.

The performance of an edge detection technique is reflected by the degree of coincidence between the detected boundary and the ground-truth boundary. There are three common errors that can occur with edge detectors: (1) Missing of valid edge points (2) Failure to localize edge points and (3) classification of noise fluctuations as edge points. Pratt introduced a Figure of merit that balances these types of error. Pratt's Figure of Merit quantifies the accuracy of edge detectors, which is determined as follows [1].

$$F = \frac{1}{\text{Max}\{N_I, N_A\}} \sum_{K=1}^{N_A} \frac{1}{1 + \alpha d^2(K)}$$

Where, N_I is the number of ideal edge pixels, N_A is the number of detected edge pixels, and d(K) is the distance from the K^{th} ideal edge to the corresponding detected edge. α is a scaling constant, which is set to 1/9 [1]. Visual results of original Canny, distributed Canny and the proposed edge detectors are presented in Figure.7 for the images from the standard Berkeley Segmentation Dataset.

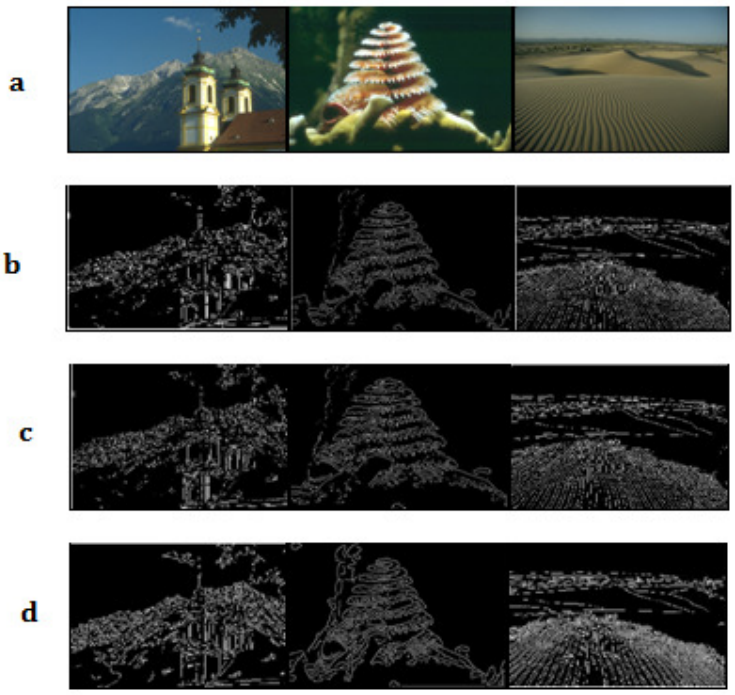

Figure7. (a) Images from Berkeley Segmentation dataset with 5 %Gaussian noises added (b) Images after applying Canny Edge Detector (c) Images after applying Distributed Edge Detector (d) Images after applying the Proposed Edge Detector

Values of Pratt's Figure of Merit for the cameraman image for different variances of Gaussain noise, for performance analysis are given in Table 2. From Table 2, it is clear that the proposed algorithm provides improvement over the other two approaches

Table 2. Comparison of Pratt's Figure Of Merit (FOM) for the Cameraman Image

Gaussian Noise Density (σ)	PRATT'S FOM for Canny edge detector	PRATT'S FOM for Distributed Canny edge detector	PRATT'S FOM for the Proposed Method
0.01	0.5388	0.6223	0.6879
0.02	0.5369	0.6220	0.6861
0.03	0.5351	0.6219	0.6744
0.04	0.5317	0.6210	0.6721
0.05	0.5302	0.6190	0.6701
0.06	0.5299	0.6110	0.6662
0.07	0.5291	0.6101	0.6643
0.08	0.5288	0.6093	0.6610
0.09	0.5211	0.6081	0.6587
0.10	0.5159	0.5989	0.6526

Figure 8 shows the graphical analysis of performance in terms of PRATT'S FOM for the Canny edge detection algorithm, distributed Canny method and the proposed edge detection technique. From Fig 8, it is obvious that the proposed method outperforms Canny edge detection technique and distributed Canny edge detection algorithm.

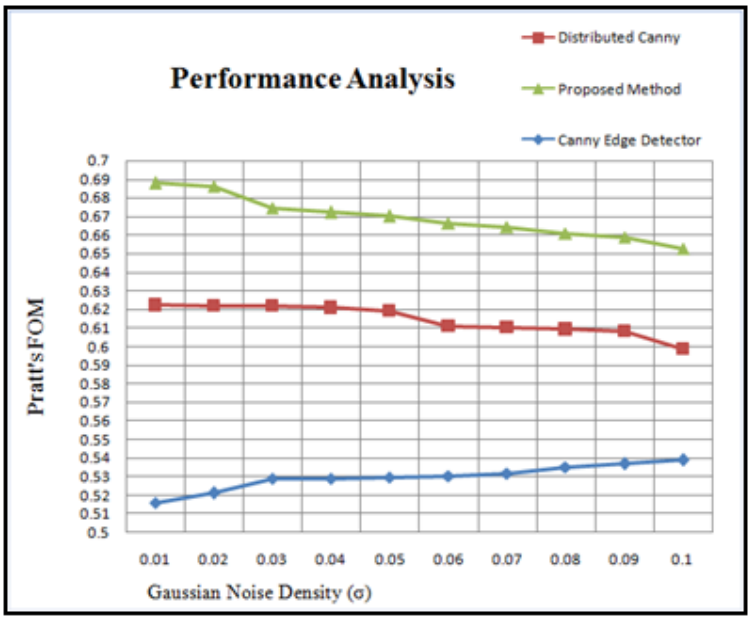

Figure 8. Performance analysis in terms of Pratt's FOM for the cameraman image corrupted by various densities of Gaussian Noise

5. CONCLUSIONS

For reducing the huge computational time of Canny edge detection technique and meet the real-time needs, a novel distributed edge detection technique is presented, which computes edges of multiple blocks effectively. The thresholds are computed adaptively using a finely tuned non-uniform quantiser. The algorithm exhibits better edge detection performance, a significant reduction in the computational time and scalability. The algorithm detected all psycho-visually prominent edges in the image for diverse block sizes.

REFERENCES

[1] Pratt, W. K . Digital Image Processing, 3rd ed.; Publisher: Wiley Interscience New York, USA, 2001; pp. 459–498

[2] Ziou, Djemel.; Salvatore, Tabbone. Edge detection techniques-an overview. Pattern Recognition and Image Analysis C/C of Raspoznavaniye Obrazov I Analiz Izobrazhenii 8, 1998, pp 537-559

[3] Gardiner, B., Coleman, S.A. and Scotney, B.W., 2016. Multiscale Edge Detection Using a Finite Element Framework for Hexagonal Pixel-Based Images. IEEE Transactions on Image Processing, 25(4), pp.1849-1861.

[4] Dollár, Piotr, and C. Lawrence Zitnick. "Fast edge detection using structured forests." IEEE transactions on pattern analysis and machine intelligence 37.8 (2015): 1558-1570.

[5] Melin, P., Gonzalez, C.I., Castro, J.R., Mendoza, O. and Castillo, O., 2014. Edge-detection method for image processing based on generalized type-2 fuzzy logic. IEEE Transactions on Fuzzy Systems, 22(6), pp.1515-1525.

[6] Swaminathan, A. and Ramapackiyam, S.S.K., 2014. Edge detection for illumination varying images using wavelet similarity. IET Image Processing, 8(5), pp.261-268.

[7] Sun, Q., Qiao, Y., Wu, H. and Wang, J., 2016. An Edge Detection Method Based on Adjacent Dispersion. International Journal of Pattern Recognition and Artificial Intelligence, 30 (10), pp 1-17

[8] Wang, Jian, Zhen-Qiang Yao, Quan-Zhang An, Yao-Jie Zhu, Xue-Ping Zhang, Wei-Bin Gu, Xin-Guang Liang et al. "RIDED-2D: a rule-based instantaneous denoising and edge detection method for 2D range scan line." International Journal of Pattern Recognition and Artificial Intelligence 25, no. 06 (2011): 807-833.

[9] Akinlar, C. and Topal, C., 2012. EDPF: a real-time parameter-free edge segment detector with a false detection control. International Journal of Pattern Recognition and Artificial Intelligence, 26(01), p.1255002.

[10] Luo, Y., Duraiswami, R. Canny edge detection on NVIDIA CUDA, Proceedings of the IEEE CVPRW, Alaska, USA ,Jun. 23-28, 2008, pp. 1–8.

[11] Canny, J. F. A computation approach to edge detection. IEEE Trans. Pattern Anal. Mach. Intell. 1986, volume. 8, no. 6, pp. 679–698, DOI: 10.1109/TPAMI.1986.4767851

[12] R, Deriche. Using canny criteria to derive a recursively implemented optimal edge detector. Int. J. Comput. Vis., 1987,volume. 1, no. 2, pp. 167–187

[13] L. Torres.; M, Robert, E, Bourennane.; M, Paindavoine. Implementation of a recursive real time edge detector using retiming technique, Proc. Asia South Pacific IFIP Int. Conf. Very Large Scale Integr., pp. 811–816

[14] F. G, Lorca.; L, Kessal.; D, Demigny. Efficient ASIC and FPGA implementation of IIR filters for real time edge detection, in Proc. IEEE ICIP, 1997, vol. 2, pp. 406–409

[15] D. V, Rao.; M, Venkatesan. An efficient reconfigurable architecture and implementation of edge detection algorithm using handle-C, in Proc. IEEE Conf. ITCC, 2004, vol. 2, pp. 843–847, DOI

[16] H, Neoh.; A, Hazanchuck. Adaptive edge detection for real-time video processing using FPGAs, Altera Corp., San Jose, CA, USA, Application Note, 2005. Available online: https://www.cse.unsw.edu.au/~cs4601/10s1/slides/neoh-edge-detection.pdf (Accessed on 24th Nov 2016)

[17] C, Gentsos.; C, Sotiropoulou.; S, Nikolaidis.; N, Vassiliadis.; Real time canny edge detection parallel implementation for FPGAs, in Proc.IEEE ICECS, 2010, pp. 499–502

[18] W, He.; K, Yuan.; An improved canny edge detector and its realization on FPGA, in Proc. IEEE 7th WCICA, 2008, pp. 6561–6564.

[19] R, Palomar, J. M, Palomares.; J. M, Castillo.; J, Olivares.; J, Gómez-Luna. Parallelizing and optimizing lip-canny using NVIDIA CUDA, in Proc. IEA/AIE, Berlin, Germany, 2010, pp. 389–398.

[20] L. H. A, Lourenco. Efficient implementation of canny edge detection filter for ITK using CUDA, in Proc. 13th Symp. Comput. Syst., 2012, pp. 33–40.

[21] Qian, Xu.; Srenivas, Varadarajan.; Chaitali,Chakrabarti.; Lina J, Karam.; A Distributed Canny Edge Detector: Algorithm and FPGA Implementation, IEEE transactions on image processing, 2014, vol. 23, no. 7, pp. 2944 - 2960

[22] Rafael C, Gonzalez.; Richard E, Woods.; Digital Image Processing , 3rd ed.; Publisher: Prentice Hall New Jersey, United States, 2006, pp. 689-763

[23] S, Nercessian.; A new class of edge detection algorithms with performance measure, M.S. thesis, Dept. Electr. Eng., Tufts Univ., Medford, MA, USA, 2009.

[24] J. K, Su.; R. M, Mersereau.; Post-processing for artifact reduction in JPEG-compressed images, in Proc. IEEE ICASSP, 1995, vol. 4, pp. 2363-2366

[25] Q, Xu.; C, Chakrabarti.; L J, Karam. A distributed Canny edge detector and its implementation on FPGA, in Proc. DSP/SPE, 2011, pp. 500–505.

[26] Berkeley Segmentation Dataset. Available online: https://www2.eecs.berkeley.edu/Research/Projects/CS/vision/bsds/ (Accessed on 21 Nov 2016)

[27] Kodak Dataset. Available online: http://r0k.us/graphics/kodak/ (Accessed on 21 Nov 2016)