

Design of Improved Canny Edge Detection Algorithm

Deepa Krushnappa Maladakara; H R Vanamala

M.Tech 4th SEM Student; Associate Professor

PESIT Bengaluru; PESIT Bengaluru

d.k.maladkar@gmail.com; vanamalahr@pes.edu

ABSTRACT— *An improved Canny edge detection algorithm is implemented to detect the edges with complexity reduction and without any performance degradation. The existing algorithms for edge detection are Sobel operator, Robert Operator, Prewitt Operator, Laplacian of Gaussian Operator, have various drawbacks in edge detection methodologies with respect to noise, performance, etc. Original Canny edge detection algorithm based on frame level statistics overcomes the drawbacks giving high accuracy but is computationally more intensive to implement in hardware. Hence a novel method is suggested to decrease the complexity without any performance degradation compared to the original algorithm. Further enhancements to the proposed algorithm can be made by parallel processing of the blocks to improve the speed of edge detection and the integration of this block-based algorithm with other block-based image codec's is easy.*

Keywords: Canny edge detection, reduced complexity, block classification, Fuzzy logic, non-uniform quantizer.

1. INTRODUCTION

One of the most important and fundamental step in any digital image processing system is Edge Detection. It is because of the fact that, detection of edges simplifies the analysis of images by significantly reducing the amount of data to be processed by filtering unwanted information, mean while preserving the important structural information of an image.

Edge, in simple words can be defined as a significant variation in gray level or intensity level of an image. This is achieved by calculating the intensity gradient of the image under consideration. Various edge detection techniques are available such as Robert edge detector, Prewitt edge detector, Sobel edge detector, Canny edge

detector, Laplacian of Gaussian, etc. Among these, Canny edge detection algorithm is most extensively used in industries, as it is highly performance oriented and provides low error bit rate.

The Canny edge detection operator was developed by John F. Canny in 1986, an approach to derive an optimal edge detector to deal with step edges corrupted by a white Gaussian noise.

An optimal edge detector should have the following criteria¹.

- 1) Good Detection: Must have a minimum error detection rate, i.e. there are no significant edges to be missed and no false edges are detected.
- 2) Good Localization: The space between actual and located edges should be minimal.
- 3) Response should be minimal: Unique response to each edge.

The Canny edge detector has a superior performance, so that it was widely used in various applications. Compared to other edge detection algorithms, the complexity of the computation involved and the latency is high. This is because the Canny edge algorithm is based on frame-level statistics, making it unsuitable for real time applications.

Many implementations of the Canny algorithm have been proposed on a wide list of hardware platforms. The optimal edge detector should satisfy three criteria's. First two criterions are detections and localization and third one is detector has unique response to each edge [1]. Paper [1] shows that step edge detector performance was improved by extending the operator point spread function along the edge. Determination of gradient direction in the direction of 45 degree and 135 degree along with X and Y direction in [2] improves the accuracy of gradient values. Self-adapt threshold calculation method in [3] calculates high and low thresholds by make use of gradient histogram of an

image reduces implementation time by employing the pipelining of the design modules on FPGA.

The utilization of reconfigurable hardware's in the image processing algorithms, reduces TTM (Time to Market), facilitates quick model for the compound algorithms and simplifies the verification and debugging [4]. The iterative post-processing algorithm is introduced in [5] to reduce the blocking and ringing effects in block transform-coded images. A novel distributed Canny edge detection algorithm [6]-[7] applies on block level image instead of whole image. In [6]-[7] the computational cost is reduced by applying 8-bin non-uniform gradient magnitude histogram for Hysteresis threshold calculation and shows that it can be employed in high speed, real time edge detection of images and videos. The Fuzzy Inference System (FIS) is used for edge detection based on three 3×3 linear spatial filters such as low-pass filter, high-pass filter and edge enhancement (Sobel) filter through the spatial convolution process to detect the edges [8]. It has very less computational cost and improved edge detection performance. The high efficiency is achieved in [9] by using efficient techniques such as entropy-based thresholding, wavelet detail and the Sobel operator for FIS.

To improve the efficiency of image processing, any enhancement to edge detection algorithm would prove to be beneficial. Our focus is employing Canny edge detection algorithm in real time real-time processing applications by reducing the latency and computational cost without affecting the detection performance. A novel method proposed for canny edge detection algorithm at block level without loss of any edge information compared to original frame level canny algorithm. More edges in smooth regions and loss of significant edges in high detailed region occurs if canny edge detection applied on block of image because it calculates thresholds based on frame level statistics. It can be solved by calculating threshold values based on block type and local distribution of gradients in the image block. Here, block classification is done with the help of fuzzy logic and a non-uniform gradient magnitude histogram is used to compute block-based hysteresis thresholds, and optimization is applied for calculation of gradient to reduce the computational cost.

II. EDGE DETECTION TECHNIQUES

The process of detecting edges in an image is called edge detection. The edge detector is a mathematical

operator that segments an image into regions of discontinuity.

There are many edge detection methods are explained in this section.

1) Robert Operator¹¹: The Roberts edge detection was introduced by Lawrence Roberts in 1965. The 2-D spatial gradient measurement of Roberts's operator on an image is quick and simple. The directional kernels of the Roberts edge detector are given by

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

This operator provides an accurate edge location with less computation but sensitive to the noise. However there is no embedded smoothing in the Roberts operator.

2) Prewitt Edge Detector¹¹: The Prewitt edge detection was proposed by Prewitt in 1970. The Prewitt operator is a discrete differentiation operator, which computes approximation of the gradient intensity of an image.

The directional kernels used by the Prewitt edge detector are shown below.

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

Prewitt edge detection obtains the direction directly from the kernel with the highest response. Prewitt's operator detects strong edges but there are discontinuities and missing edges in the curved regions.

Comparably, Prewitt's operator is more sensitivity to the noise. However it doesn't contain weighted values in the kernel.

3) Sobel Edge Detector^{11, 12}: The Sobel edge detection method was introduced by Sobel in 1970. The sobel operator is average, non-directional edge detectors. The directional kernel of sobel operator is as follows

$$G_x = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix} \quad G_y = \begin{bmatrix} -1 & 0 & +1 \\ 2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix}$$

The smoothing is achieved by weighted kernels shown above by giving more importance to certain

locations. The Sobel operator detects strong edge points, but still notices some missing edges in the edge map.

4) Laplacian Of Gaussian¹²: Laplacian of Gaussian was introduced by Marr and Hildreth (1980), who combined Gaussian filtering with the Laplacian. The Laplacian of an image highlights the regions of rapid change in the intensity and de-emphasizes regions with slowly varying intensity levels; hence it is used in the edge detection.

The LOG operator normally takes a single gray level image as input and produces another gray level image as output. The Laplacian $L(x,y)$ of an image with pixel intensity values $I(x,y)$ is given by

$$L(x,y) = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2} \quad (1)$$

Since the input image is a set of discrete pixels, these discrete convolution kernels are used in order to approximate the second derivatives in the definition of the Laplacian. Commonly used 3×3 small discrete convolution kernels are given below:

| | | | | | | | | |
|---|----|---|----|----|----|---|----|---|
| 0 | 1 | 0 | -1 | 2 | -1 | 1 | 1 | 1 |
| 1 | -4 | 1 | 2 | -4 | 2 | 1 | -8 | 1 |
| 0 | 1 | 0 | -1 | 2 | -1 | 1 | 1 | 1 |

The 2-D LoG function centered on zero and with Gaussian standard deviation (σ) has the form:

$$L(x,y) = \frac{1}{\pi\sigma^4} \left(1 - \frac{x^2+y^2}{2\sigma^2}\right) e^{-\frac{x^2+y^2}{2\sigma^2}} \quad (2)$$

LOG operator has two advantages:

1. It requires fewer arithmetic operations because it uses the kernel size of Gaussian and the Laplacian are much smaller than the image.

2. The LOG (Laplacian of Gaussian) kernel can be pre calculated in advance so only one convolution needs to be performed at run-time on the image.

Disadvantages of LOG operator:

1. Malfunctioning at corners, curves and at the edges.
2. Cannot find the orientation of edge as a consequence of using the Laplacian filter.

5) Canny Edge Detector^{11, 12}: The Canny edge detector algorithm was proposed by John Canny in 1986. Canny edge detector has low error rate optimal edge detector. Canny proposed three criteria to improve edge detection method¹.

Canny edge detector involves the following steps:

1. *Smoothing:* It uses linear filtering with a Gaussian kernel to smooth the image.

2. *Gradient and direction calculation:* It finds the image gradient in order to highlight regions with high spatial derivatives.

3. *Non-maximal suppression (NMS):* Edges will occur at points where the gradient is at the maximum. Therefore, all points that are not at the maximum should be suppressed.

4. *Threshold calculation:* The high and low thresholds are calculated with the help of histogram of an image.

5. *Hysteresis Thresholding:* The pixels whose gradient magnitudes are below the low threshold are set as zero (non-edge). The pixels whose gradient magnitudes are above the high threshold are set as one (edge). The pixels whose gradient magnitudes falls between the two thresholds is set to zero unless it is connected to the pixel having magnitude more than high threshold.

III. DESIGN AND IMPLEMENTATION

In the improved Canny edge detection method, the whole image is divided into fixed size blocks. The improved Canny algorithm is same as original Canny algorithm except that this algorithm is applied at the block level. Each block is classified as - smooth, texture, hybrid, strong to minimize the computation. These block classifications done, by making use of fuzzy logic for good performance. The hysteresis, high and low thresholds calculation is modified to enable pipelining in a block-level processing without degrading the edge detection performance.

First input image is divided into $m \times m$ overlapping blocks, with $m = n + L + 1$, where L is size of gradient mask and n is size of non-overlapping block. In other words the whole image is divided into $n \times n$ non-overlapping blocks, the $m \times m$ overlapping blocks are obtained by simply extending each non-overlapping block by $(L + 1)/2$ pixels along the left, right, top, and bottom boundaries, respectively.

The Fig. 1 shows $m \times m$ overlapping block structure. This prevents the edge artifacts and loss of edges at block boundaries while computing the gradients and due to the fact that the NMS operation at boundary pixels requires the gradient values of the neighboring pixels of the considered boundary pixels in a block.

In order to perform NMS for the border pixel (i, j) , the gradient information of the adjacent pixels $(i-1, j-1)$, $(i, j-1)$, $(i+1, j-1)$, $(i-1, j+1)$, $(i-1, j)$ are needed. In order to compute the gradient of the adjacent pixels $(i, j-1)$, $(i+1, j-1)$, $(i-1, j)$, $(i-1, j-1)$, $(i-1, j+1)$ for the 3×3 gradient mask, the block has to be extended by 2 (where $(L-1)/2 + 1 = 2$) pixels on all sides in order to generate a block of size $(n+4) \times (n+4)$. Thus, m equals to $n+4$.

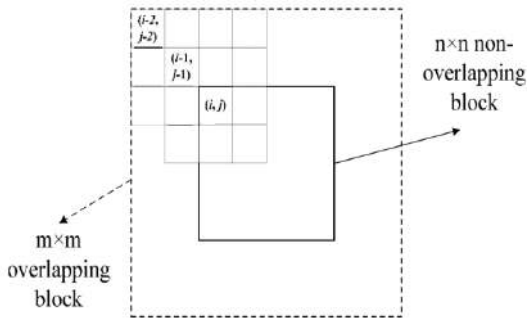


Fig. 1: Example structure of an $m \times m$ overlapping block (courtesy of Qian Xu).

Note that, the final edge map of each block contains the edges of $n \times n$ non-overlapping block, where $n = m - L - 1$.

The $m \times m$ overlapping image block is given as input to proposed block diagram and it produces edges of an $n \times n$ non-overlapping block.

The proposed block diagram is shown in the Fig. 2, consists of 7 processing units:

1. Gaussian Filtering,
2. Horizontal and Vertical Gradient Calculation Unit,
3. Direction and Magnitude calculation Unit,
4. Non-maximal Suppression Unit,
5. Threshold Calculation Unit,
6. Fuzzy logic Unit,
7. Hysteresis Thresholding.

1) Gaussian Filtering: Gaussian filter is a convolution operator, which blur the images and removes the high frequency components (noise). The characteristics equation of Gaussian filter is given below.

$$G(x, y) = \frac{1}{2\pi\sigma^2} * e^{-(x^2+y^2)/(2\sigma^2)} \quad (3)$$

Here σ is the standard deviation of Gaussian distribution.

Advantages of Gaussian Filter:

1. The Gaussian filter can be personalized in noisy environment, by just varying σ value.
2. Since it is a low pass filter, it removes the high frequency components of an image.

2) Horizontal and Vertical Gradient Calculation Unit¹⁴:

The gradient of an image is basically used for edge detection in image processing. The calculation of the gradient involves, determining the edges where there are significant changes in the intensity of the grey image.

The Horizontal and Vertical gradient calculated as shown in Fig. 3.

| | | |
|----------|----------|----------|
| A_{00} | A_{01} | A_{02} |
| A_{10} | A_{11} | A_{12} |
| A_{20} | A_{21} | A_{22} |

$$* \frac{1}{4}$$

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

(a)

| | | |
|----------|----------|----------|
| A_{00} | A_{01} | A_{02} |
| A_{10} | A_{11} | A_{12} |
| A_{20} | A_{21} | A_{22} |

$$* \frac{1}{4}$$

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

(b)

Fig. 3: (a) Gaussian discrete kernel for G_x ; (b) Gaussian discrete kernel for G_y .

3) Direction and Magnitude calculation

Unit: The gradient magnitude and the gradient direction are calculated by following equations:

$$\nabla A = \sqrt{G_x^2 + G_y^2} \quad (4)$$

$$\theta = \tan^{-1}(G_x/G_y) \quad (5)$$

The gradient magnitude is often approximated by:

$$|\nabla A| = |G_x| + |G_y| \quad (6)$$

Considering the pixel "a", the four possible directions to describe the surrounding pixels are –

1. Horizontal direction (0 degrees)
2. Positive diagonal direction (45 degrees)
3. Vertical direction (90 degrees)
4. Negative diagonal direction (135 degrees)¹⁴.

One of the four directions is chosen depending on the closest direction of the pixel to the gradient direction.

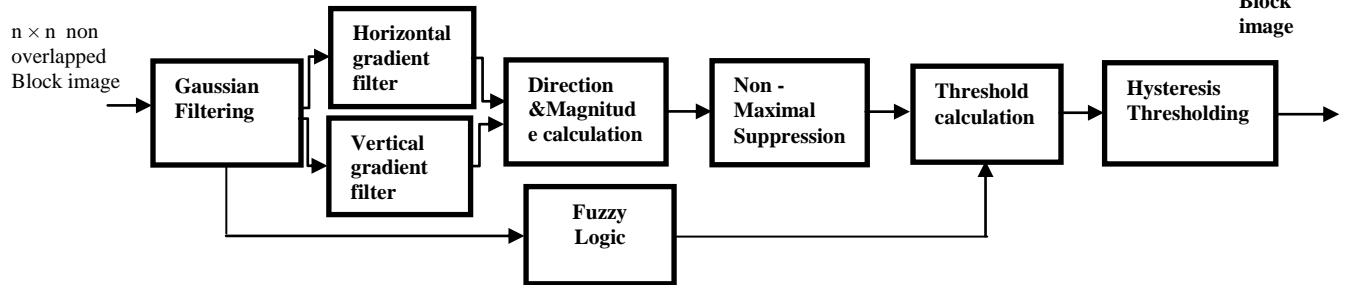


Fig. 2: Proposed Block Diagram of Canny Edge Detection Algorithm.

One of the four directions is chosen depending on the closest direction of the pixel to the gradient direction.

| | | |
|---|---|---|
| x | x | x |
| x | a | x |
| x | x | x |

Arctan and the division can be eliminated by simply comparing magnitude of horizontal gradient (G_x) and magnitude of vertical gradient (G_y) values. The final gradient direction as shown in Fig. 4.8 is taken as -

1. 90 degrees, if G_y is 2.5 times greater than G_x
2. 0 degrees, if G_x is 2.5 times greater than G_y
3. 45 degrees, if both G_y and G_x are positive or negative
4. 135 degrees, if G_y and G_x are of different sign

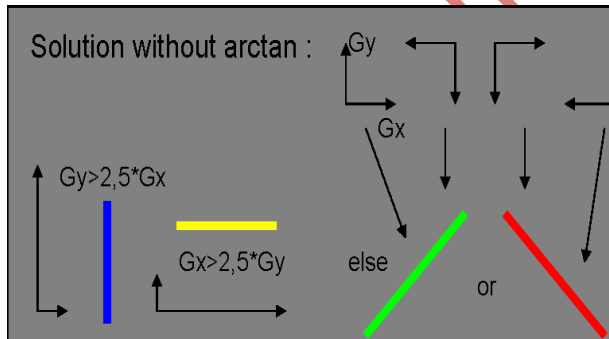


Fig. 4: Simplified phase calculations (courtesy of Lukas Benda).

4) Non-maximal Suppression Unit: Non-maximum suppression (NMS) is basically used in edge detection algorithms. This technique is used to remove the pixels which are not desirable. The pixels with edge strength greater than the neighboring pixels in the gradient direction are retained as it is, otherwise set to zero.

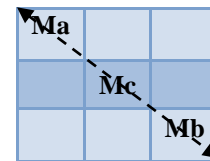


Fig. 5: Example of non-maxima elimination for a particular pixel.

The above Fig. 5 shows NMS principle. The M_c is the magnitude of the current pixel. M_a and M_b are the magnitudes of the neighbor pixels in the gradient direction of the current pixel (M_c). If ($M_c > M_a$) and ($M_c > M_b$), the current pixel (M_c) remains the same, otherwise removed.

5) Fuzzy logic Unit: The concept of fuzzy logic was first introduced in the year 1965, by Dr.Lotfi A. Zadeh of the University of California at Berkeley. Fuzzy logic is one of many-valued logic that attempts to solve problems by assigning values to an imprecise field in order to get most accurate result possible. It has values between 0 and 1 when compared to binary logic.

In the improved canny edge detection, the fuzzy logic is used in the block classification to determine exact block type. The block classification is done using Fuzzy inference system (FIS) as shown in the Fig. 6.

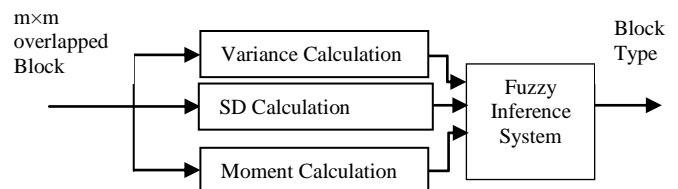


Fig. 6: The block diagram for Block Classification. Fuzzy inference system⁸ (FIS) is a fuzzy modeling approach and a computing framework based on the concepts of fuzzy rule base which contains the selection of fuzzy rules, and membership functions, and the

reasoning mechanism, the fuzzy inference system is designed with three inputs and one output⁹.

Table 1: Fuzzy rules for block classification

| Fuzzy Inputs | | | Fuzzy Outputs |
|--------------|----------------|--------|---------------|
| Variance | Std. deviation | Moment | Block Type |
| Low | Low | Low | Smooth |
| Low | Low | Medium | Smooth |
| Low | Medium | Medium | Texture |
| Medium | Medium | Medium | Texture |
| Medium | High | Medium | Hybrid |
| Medium | High | High | Hybrid |
| High | Medium | Medium | Hybrid |
| High | Medium | High | Strong |
| High | High | High | Strong |

Fig. 7 shows an example where the 512×512 Cameraman image is segregated into blocks, and each block is classified according to this Fuzzy classification technique as shown in Table 1. Fig. 4.12 (b)–(e) provides a respective examples of uniform, texture, hybrid, strong edge block for the 512×512 Cameraman image with a block size of 64×64 .

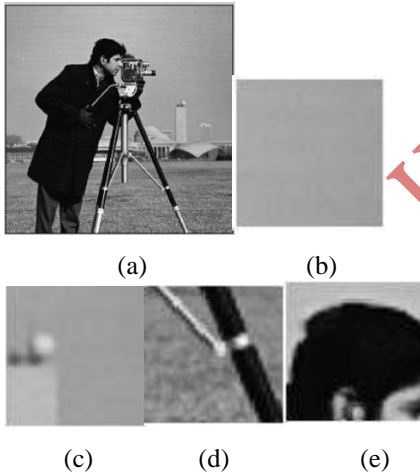


Fig. 7: (a) Original 512×512 Camera man image; (b) uniform block; (c) texture block; (d) hybrid block; (e) strong edge block of the Cameraman image. Shown blocks are of size 64×64 .

Let P1 be the percentage of pixels in a block which are classified as strong edges⁷. The P1 value shown in Table 2, for each block is taken from Qian Xu, et. Al

Table 2: P1 values for each block type of size 64×64

| Block Type | P1 value |
|-------------------|----------|
| Uniform Block | 0 |
| Texture Block | 0.1 |
| Hybrid Block | 0.2 |
| Strong Edge Block | 0.4 |

The corresponding P1 value for each block type is given as input to the threshold calculation unit.

6) Threshold Calculation Unit: In this unit low and high thresholds are calculated based on given P1 value for each block^{6, 7}. If the block is smooth, then there is no need of further processing, it is directly connected to the output. Here 8-step non-uniform quantizer⁷ is used to calculate high and low threshold.

7) Hysteresis Thresholding³: This unit decides the each pixel as edge or non-edge. The pixels whose gradient magnitudes are below the low threshold are set as zero (non-edge). The pixels whose gradient magnitudes are above the high threshold are set as one (strong edge). The pixels whose gradient magnitude falls between the two thresholds are set to zero unless it is connected to strong edge.

IV. SYNTHESIS RESULTS

The proposed Design can process any type of image of size 512×512 . The 512×512 grayscale image with a block size 64×64 is processed using Xilinx Virtex-5 to show the performance of the system. The 8-bit data with maximum of 255 gray values is sufficient to meet the requirements.

The 512×512 grayscale camera man image is processed using proposed improved Canny Edge Detection method and the edge detected 512×512 camera man image is as shown in Fig. 8 (b).

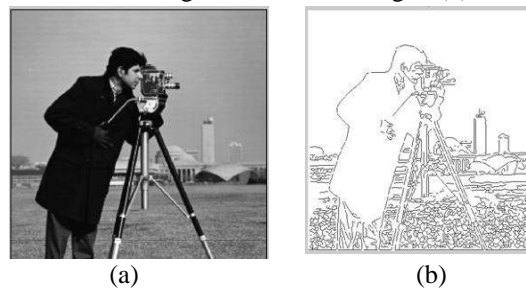


Fig. 8: (a) The 512×512 grayscale camera man image; (b) The edge detected 512×512 camera man image

using the proposed improved Canny edge detector with block size of 64×64 and with a 3×3 gradient mask.

A. Resource Utilization: The resource utilization results of distributed canny edge detection and our Improved Canny edge detection shown in Table 3. In Improved Canny edge detection number of slices used is 6941; this is achieved by using approximation methods in magnitude and direction calculation units.

Table: 3 Resource Utilization of Distributed and Improved Canny Detection Algorithm.

| Method | Image Size | Xilinx device | Used Slices |
|--|------------|-----------------|-------------|
| Distributed Canny Edge Detection (Paper [7]) | 512×512 | Xilinx Vertex-5 | 23904/37440 |
| Improved Canny Edge Detection (our method) | 512×512 | Xilinx Vertex-5 | 6940/32640 |

B. Observation: Bit Error rate (BER), Peak Signal to Noise ratio, Latency for the proposed Canny edge detection method compare to the canny function are tabulated in the following Table. 4.

Table: 4 BER, PSNR, Latency compare to Matlab Image

| | |
|----------------------------|---------|
| Bit Error Rate | 8.96% |
| Peak Signal to Noise Ratio | 34.5 dB |
| Latency | 0.014ms |

Bit Error Rate obtained for proposed method is very low which implies that the number of false positives and false negatives in edge detection will be less.

Peak Signal to Noise Ratio (PSNR) is 34.5 dB, which indicates good image quality and less error introduced in the image.

Latency of the proposed canny edge detection algorithm is 0.014ms, which means system will respond within .014ms. Since latency is the function of the block size instead of the frame size as in original canny algorithm which has a high latency, the resultant latency in the proposed algorithm is low.

V. CONCLUSION AND FUTURE WORK

Conclusion:

The original Canny algorithm relies on frame-level statistics to predict the high and low thresholds and thus has latency proportional to the frame size. In order to reduce the large latency and meet real-time requirements, improved Canny edge detection algorithm is proposed which has the ability to compute edges of multiple blocks at the same time.

The block classification used is based on the fuzzy logic which provides accurate block types. Adaptive thresholding is applied to different images. These procedures results in three benefits: 1) significant reduction in the latency; 2) low error bit rate; 3) better edge detection performance.

Besides, non-uniform quantized histogram for block hysteresis threshold and approximation methods for gradient calculation are results in low complexity. The proposed algorithm is scalable and has a good edge detection performance.

Future work:

Calculation of medium range for variance, Standard deviation, moment is done manually with the help of Matlab. This can be generated using Verilog.

Each block is independent of each other so that blocks can be processed parallel to improve the speed of edge detection and the integration of this block-based algorithm with other block-based image codec's is easy.

Acknowledgement

I would like to say thanks to my college and TEQUIP-II department for their economical support.

REFERENCES

1. J. F. Canny, "A computation approach to edge detection," IEEE Transactions on Pattern Analysis And Machine Intelligence, vol. 8, no. 6, pp. 769–798, Nov. 1986.
2. X. Wang and J. Q. Jin, "An edge detection algorithm based on improved Canny operator," in Seventh International Conference on Intelligent Systems Design and Applications (ISDA 2007), October 2007, pp. 623–628.
3. W. He and K. Yuan, "An improved canny edge detector and its realization on FPGA," in Proc. IEEE 7th WCICA, Jun. 2008, pp. 6561–6564.

4. D. V. Rao and M. Venkatesan, "An efficient reconfigurable architecture and implementation of edge detection algorithm using handle-C," in Proc. IEEE Conf. ITCC, vol. 2. Apr. 2004, pp. 843–847.
5. J. K. Su and R. M. Mersereau, "Post-processing for artifact reduction in JPEG-compressed images," in *Proc. IEEE ICASSP*, vol. 3. May 1995, pp. 2363–2366.
6. Qian Xu, C. Chakrabarti, and L. J. Karam, "A distributed Canny edge detector and its implementation on FPGA," in Proc. DSP/SPE), Jan. 2011, pp. 500–505.
7. Qian Xu, SrenivasVaradarajan, ChaitaliChakrabarti and L. J. Karam, "A Distributed Canny Edge Detector: Algorithm and FPGA Implementation," in Proc. IEEE Transaction Image Processing, July 2014, pp. 2944–2960.
8. Aborisade, D.O, "Fuzzy Logic Based Digital Image Edge Detection," *Global Journal of Computer Science and Technology*, November 2010.
9. HodaFarag, Said E. El-Khamy, "Rigorous Pack Edge Detection Fuzzy System," *International Journal of Engineering and Science*, October 2014, PP 19-30.
10. Rafael C.Gonzalez, Richard E. Woods, "Digital Image Processing", 3rd edition, published in 2009.
11. S. Jansi, P. Subashini, "Optimized Adaptive Thresholding based Edge Detection Method for MRI Brain Images," *International Journal of Computer Applications*, August 2012.
12. Mohammad Shahnoor Islam Khan, "Implementation of Edge & Shape Detection Techniques and their Performance Evaluation," M.S.thesis, Dept. Computer science, Ryerson Univ., Toronto, Canada, Dec 2012.
13. Pedro Albertos and Antonio Sala, "Fuzzy Logic Controllers, Advantages and Drawbacks," PhD thesis, Dept. Ingenieria de Sistemas, Politecnica de Valencia University, Valencia, Sept. 1998.
14. Lukas Benda, Pierree-Andre Mudry, Prof. Auke Ijspeert, "Hardware Acceleration for Image Processing", M.S.thesis, EPFL, I&C, BIRG, Jan. 2008.