Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 4, Issue. 12, December 2015, pg.58 – 66

Review on Determination of Edges by Automatic Threshold Value Generation

Pardeshi Ragini Sanjay[1], Prof. Naoghare M.M.[2]

[1]Department of Computer Engineering (Savitribai Phule Pune University), Sir Visvesvaraya Institute Technology, Chincholi, Tal-Sinner, Dist- Nasik, India

[2]Department of Computer Engineering (Savitribai Phule Pune University), Sir Visvesvaraya Institute Technology, Chincholi, Tal-Sinner, Dist- Nasik, India

[1] raginipardeshi03@gmail.com; [2] manisha.naoghare@gmail.com

Abstract: ***As the performance of canny edge detector is good it is widely use edge detection algorithm. Because of its frame level statistics, it is not only more intensive but also has a high latency. We proposed a system using canny edge algorithm at block level images without losing edge detection compared to original frame level canny algorithm. Applying canny edge detection algorithm on input image may cause excessive edges in smooth regions and to loss of significant edges in high-detailed regions. To solve this problem, we present a distributed Canny edge detection algorithm that adaptively computes the edge detection thresholds based on the block type and the local distribution of the gradients in the image block.***

In this system we have implemented the new canny algorithm at the block level without any loss in edge detection performance compared with the original frame-level Canny algorithm. The methodology employed here is distributed canny edge detection algorithm on the basis of threshold segmentation as the segmentation has good advantage in noise restraining with wide applicability and computes the thresholds based on the block statistics. It is capable of supporting fast edge detection of images and videos with high resolutions, including full-HD since the latency is now a function of the block size instead of the frame size.

The main objective of the proposed system includes significant reduction in latency, better edge detection performance and parallel implementation of image blocks. The proposed system is implemented using MATLAB version R2012a which simulates the required results with improved edge detection algorithm than existing. The quantitative results include the effect of mask size on the latency and effect of PSNR on quality of edges which quantify the overall system performance.

Keywords: Canny Edge Detector, Threshold, Latency, Codec's, Hysteresis

I. INTRODUCTION

One of the most important tasks in any vision system is the detection of edges in digitized images. The requirements of a good edge detector are that it responds only to true edge structure and is relatively insensitive to noise. Computationally efficient realization is also required. Several methods have been proposed and most of them suggest considering local operations on the elements of the input image to extract edges. Examples are the gradient operators and second-derivative operators generally followed respectively by peak and zero-crossing detections. Thresholding and thinning operations are applied to localize boundaries and remove edge segments that arise from noise in the imaging process [1]

To select the geometrical information related to the structure of the objects present in an image, a key step consists of extracting the edges of the considered objects. Several accurate numerical algorithms have been proposed in this field (Roberts, Prewitt, Sobel, Canny, Deriche,...) and have already been implemented in software or with dedicated hardware in several applications . One of the main differences between these algorithms is the calculation complexity of differential operators between neighbour pixels : from 2x2 to more than 9x9 kernels. The choice of an edge detection algorithm depends mainly on the signal to noise ratio (robustness), the localization and the quality of the detection and the implementation efficiency. One of the main problem is the necessary CPU time required to perform the edge extraction [2].

The edge detection process serves to simplify the analysis of images by drastically reducing the amount of data to be processed, while at the same time preserving useful structural information about object boundaries. There is certainly a great deal of diversity in the applications of edge detection, but it is felt that many applications share a common set of requirements [3-4].

The basic edge-detection operator is a matrix area gradient operation that determines the level of variance between different pixels. The edge-detection operator is calculated by forming a matrix centered on a pixel chosen as the centre of the matrix area. If the value of this matrix area is above a given threshold, then the middle pixel is classified as an edge. Examples of gradient-based edge detectors are Roberts, Prewitt, and Sobel operators. All the gradient-based algorithms have kernel operators that calculate the strength of the slope in directions which are orthogonal to each other, commonly vertical and horizontal. Later, the contributions of the different components of the slopes are combined to give the total value of the edge strength [5].An important issue in edge detection is the scale of detection filter. Small-scaled filters are sensitive to edge signals but also prone to noise, whereas large-scaled filters are robust to noise but could filter out fine details. As suggested by Marr and Hildreth, multiple scales could be employed to describe and synthesize the varieties of edge structures [6].The canny edge detection algorithm includes four tasks: image smoothing, vertical and horizontal gradient calculation, directional non maximum suppression and threshold. Classically, edge detection algorithms are implemented on software [7].

II. LITERATURE REVIEW

The purpose of edge detection in general is to significantly reduce the amount of data in an image, while preserving the structural properties to be used for further image processing. Several algorithms exists, and this worksheet focuses on a particular one developed by John F. Canny (JFC) in 1986. Even though it is quite old, it has become one of the standard edge detection methods and it is still used in research. The aim of JFC was to develop an algorithm that is optimal with regards to the following criteria:

1. Detection: The probability of detecting real edge points should be maximized while the probability of falsely detecting non-edge points should be minimized. This corresponds to maximizin the signal-to-noise ratio.

2. Localization: The detected edges should be as close as possible to the real edges.

3. Number of responses: One real edge should not result in more than one detected edge (one can argue that this is simplicitly included in the first requirement). With Canny's mathematical formulation of these criteria, Canny's Edge Detector is optimal for a certain class of edges.

Edges characterize boundaries and are therefore a problem of fundamental importance in image processing. Edges in images are areas with strong intensity contrasts – a jump in intensity from one pixel to the next. Edge detecting an image significantly reduces the amount of data and filters out useless information, while preserving the important structural properties in an image. This was also stated in my Sobel and Laplace edge detection tutorial, but I just wanted reemphasize the point of why you would want to detect edges. The Canny edge detection algorithm is known to many as the optimal edge detector. Canny's intentions were to enhance the many edge detectors already out at the time he started his work. He was very successful in achieving his goal and his ideas and methods can be found in his paper, "A Computational Approach to Edge Detection". In his paper, he followed a list of criteria to

improve current methods of edge detection. The first and most obvious is low error rate. It is important that edges occuring in images should not be missed and that there be NO responses to non-edges. The second criterion is that the edge points be well localized. In other words, the distance between the edge pixels as found by the detector and the actual edge is to be at a minimum. A third criterion is to have only one response to a single edge. This was implemented because the first 2 were not substantial enough to completely eliminate the possibility of multiple responses to an edge. Based on these criteria, the canny edge detector first smoothes the image to eliminate and noise. It then finds the image gradient to highlight regions with high spatial derivatives. The algorithm then tracks along these regions and suppresses any pixel that is not at the maximum (non maximum suppression). The gradient array is now further reduced by hysteresis. Hysteresis is used to track along the remaining pixels that have not been suppressed. Hysteresis uses two thresholds and if the magnitude is below the first threshold, it is set to zero (made a non edge). If the magnitude is above the high threshold, it is made an edge. And if the magnitude is between the 2 thresholds, then it is set to zero unless there is a path from this pixel to a pixel with a gradient above T2.

III. CANNY EDGE DETECTION METHOD

Canny proposed a new approach to edge detection that is optimal for step edges contaminated by white nose. The optimality of the detector is related to three criteria -

- The detection criterion expresses the fact that important edges should not be missed and that there should be no spurious responses.
- The localization criterion says that the distance between the actual and located position of the edge should be minimal.
- The one response criterion minimizes multiple responses to a single edge[8]

The Canny edge detector is predominantly used in many real-world applications due to its ability to extract significant edges with good detection and good localization performance. Unfortunately, the Canny edge detection algorithm contains extensive pre-processing and post-processing steps and is more computationally complex than other edge detection algorithms, such as Roberts, Prewitt and Sobel algorithms. Furthermore, it performs hysteresis thresholding which requires computing high and low thresholds based on the entire image statistics. This places heavy requirements on memory and results in large latency hindering real-time implementation of the Canny edge detection algorithm [9].

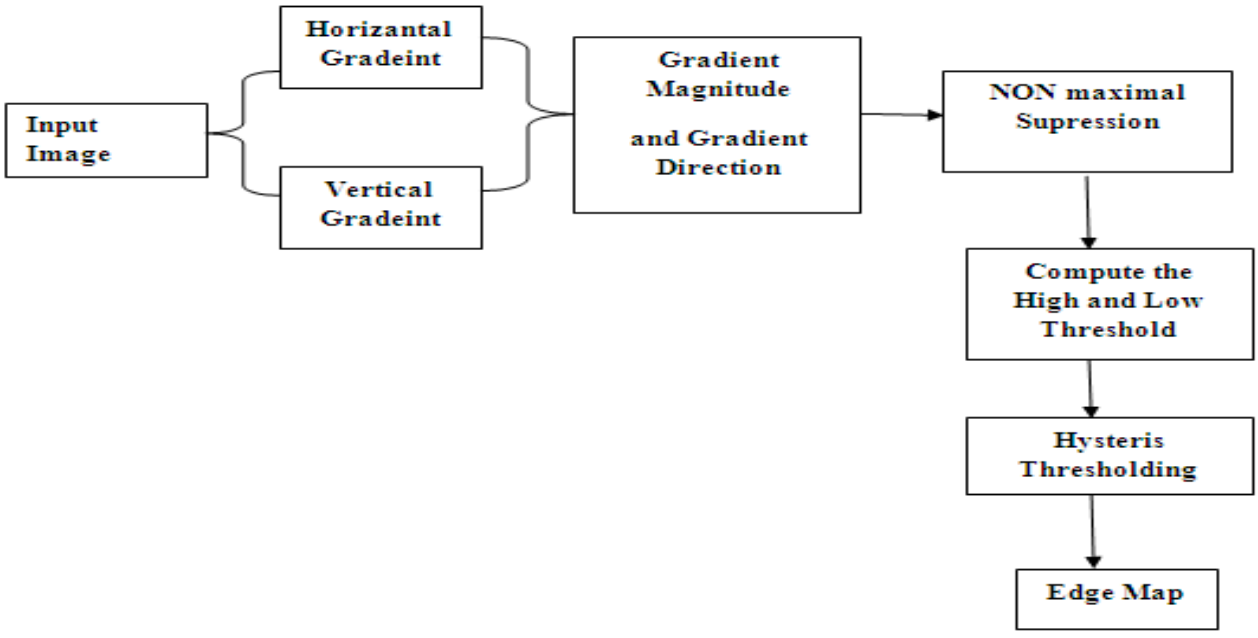

Fig1. Block Diagram of the Canny Edge Detection Algorithm

IV. DISTRIBUTED CANNY EDGE DETECTION METHOD

The Canny edge detection algorithm operates on the whole image and has a latency that is proportional to the size of the image. While performing the original Canny algorithm at the block-level would speed up the operations, it would result in loss of significant edges in high-detailed regions and excessive edges in texture regions. Natural images consist of a mix of smooth regions, texture regions and high-detailed regions and such a mix of regions may not be available locally in every block of the entire image. We proposed a distributed Canny edge detection algorithm, which removes the inherent dependency between the various blocks so that the image can be divided into blocks and each block can be processed in parallel [9].

The methodology used here is distributed canny edge detection algorithm on the basis of threshold segmentation as the segmentation has good advantage in noise restraining with wide applicability and computes the thresholds based on the block statistics. Each pixel in image block has specific threshold and then the structural operator of small size is used to obtain the outline of an edge. The proposed system is robust to change in block size which can affect the processing time of algorithm. This algorithm for edge detection gives the intended results for the different images using threshold segmentation

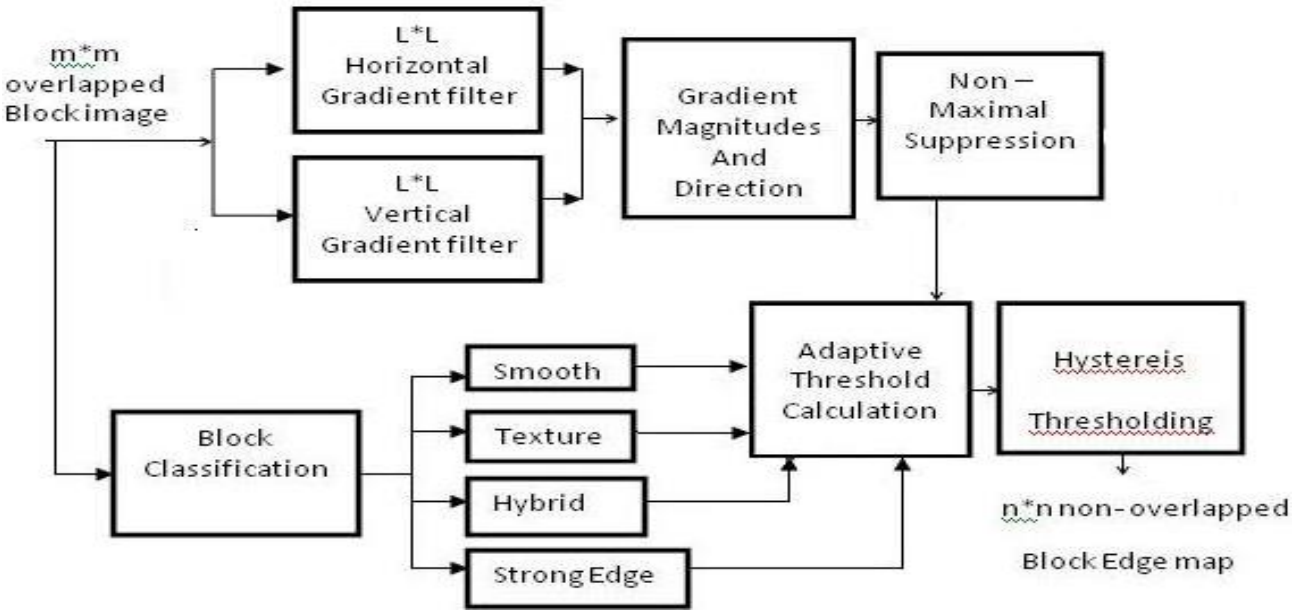

Fig2. Distributed Canny Edge Detection Algorithm Block Diagram

The Canny algorithm uses an optimal edge detector based on a set of criteria which include finding the most edges by minimizing the error rate, marking edges as closely as possible to the actual edges to maximize localization, and marking edges only once when a single edge exists for minimal response [4].

The Canny algorithm consists of the following steps [4] –

IV-A Image Smoothing

The input image by Gaussian mask. In the Canny edge detection algorithm, the gradient calculation (Step 1) is performed by using Finite Impulse Response (FIR) gradient masks designed to approximate the following 2D sampled versions of the partial derivatives of a Gaussian function:

$$F_x(x,y) = -\frac{x}{\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} = \left(-xe^{-\frac{x^2}{\sigma^2}}\right)\left(\frac{1}{\sigma^2} e^{-\frac{y^2}{\sigma^2}}\right)$$

$$F_y(x,y) = -\frac{y}{\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} = \left(-ye^{-\frac{y^2}{\sigma^2}}\right)\left(\frac{1}{\sigma^2} e^{-\frac{x^2}{\sigma^2}}\right)$$

Where σ is the standard deviation of the Gaussian function. The size of the gradient masks used by the Canny edge detector is usually implemented as a function of the chosen σ, with larger values of σ yielding larger masks. However, the best choice of σ is image-dependent and can be selected by the user based on knowledge of the present noise characteristics or the size of desired objects in the image [6]. The parameter σ can also be set by a separate application that estimates the noise and/or scale of objects in the image.

IV-B Gradient calculation

Calculating the horizontal gradient Gx(x, y) and vertical gradient Gy(x, y) at each pixel location by convolving the image I(x, y) with partial derivatives of a 2D Gaussian function

$$G(x,y) = \frac{1}{2\pi\sigma} \exp\left(\frac{-x^2 + y^2}{2\sigma^2}\right)$$

IV-C Gradient magnitude and direction

Computing the gradient magnitude G(x, y) and direction θG(x, y) at each pixel location.

$$M(x,y) = \sqrt{Gx^2(x,y) + Gy^2(x,y)}$$

$$\theta_G(x,y) = \arctan\left(\frac{G_y(x,y)}{G_x(x,y)}\right)$$

If the gradient magnitude of a pixel is greater than the high threshold, this pixel is considered as a strong edge. If the gradient magnitude of a pixel is between the low threshold and high threshold, the pixel is labeled as a weak edge. Strong edges are interpreted as “certain edges” and can be immediately included in the final edge images. Weak edges are included if and only if they are connected to strong edges.

IV-D Non-maximum suppression (NMS)

This step involves computing the gradient direction at each pixel. If the pixel’s gradient direction is one of 8 possible main directions (0°, 45°, 90°, 135°, 180°, 225°, 270°, 315°, the gradient magnitude of this pixel is compared with two of its immediate neighbors along the gradient direction and the gradient magnitude is set to zero if it does not correspond to a local maximum. For the gradient directions that do not coincide with one of the 8 possible main directions, an interpolation is done to compute the neighboring gradients

IV-E Pre-processing

Performing Pre-processing on the input image which includes the classification of image into different blocks on the basis of image contents that is smooth block, uniform block, texture block, strong edge block .Hysteresis Thresholding Performing hysteresis thresholding to find where edges begin and end. The High and low threshold of each block should match with the thresholds for entire image Threshold segmentation This block is intended to use for segmenting the image into segments having appropriate threshold to achieve fine edges of the input image.

Natural images consist of a mix of smooth regions, texture regions and high-detailed regions and such a mix of regions may not be available locally in every block of the entire image. The input image is divided into m×m overlapping blocks. The adjacent blocks overlap by (L − 1)/2 pixels for a L × L gradient mask

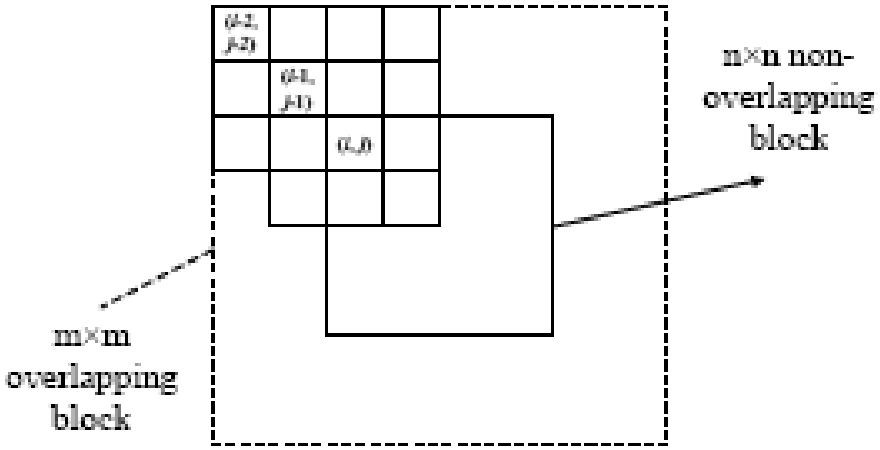

Fig3**.** Structure of m x m overlapping block

V. EXPRIMENT RESULTS

V-A Canny Edge detection

Result of Existing System

Fig4**.** Result of Existing System

V-B Distributed Canny edge detection

The edge detection effectiveness of the proposed distributed approach is analyzed by comparing perceptual significance of its resulting edge map with the existing distributed canny edge detector and original canny edge detector.

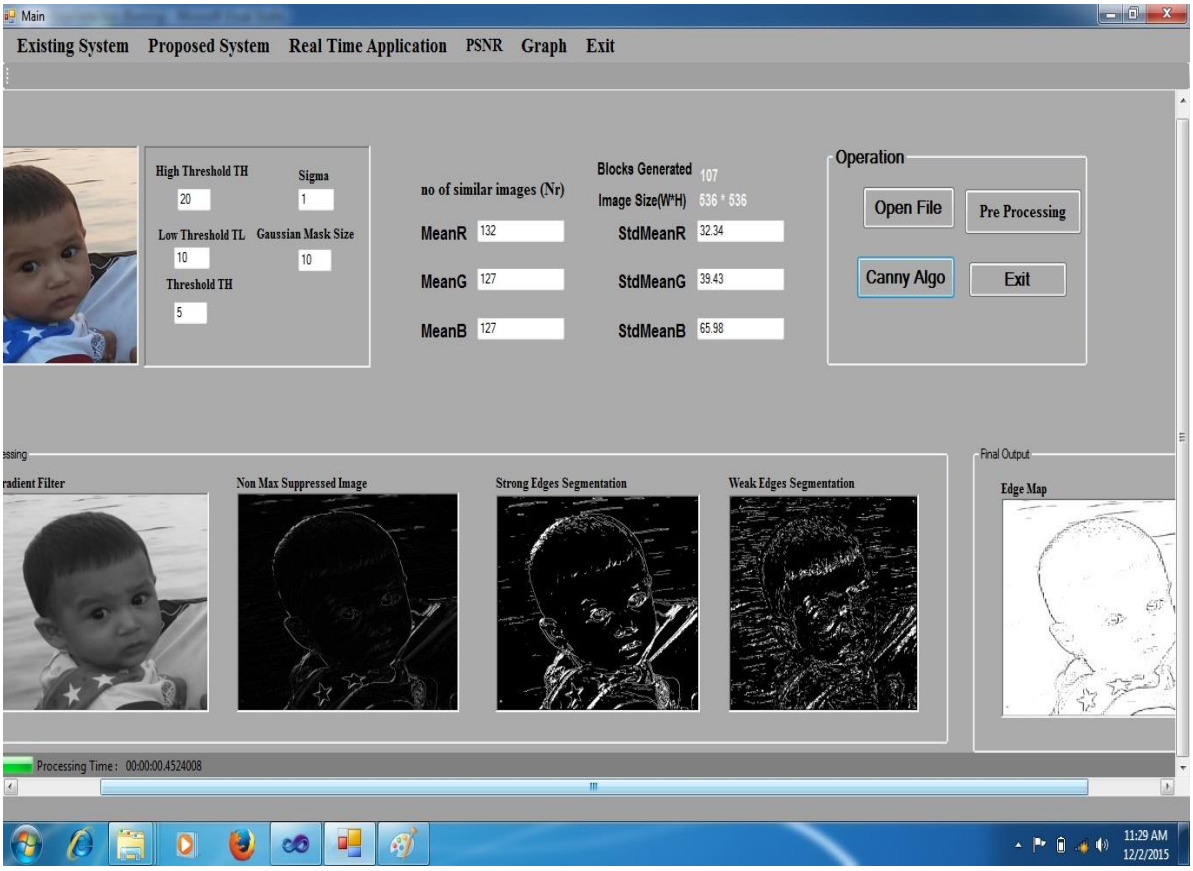

Fig5Distributed Canny edge detection

V-C Comparison result for processing time

The proposed canny algorithm is tested using standard test images. The proposed algorithm is compared with existing canny edge algorithm. To objectively compare the edge detection results for this set of images, the measurement PSNR (Peak Signal to Noise Ratio) and processing time t is used to evaluate the performance quantitatively. The experimental results of the performance evaluation for mask size 5x5 are listed in Table below.

TABLE I

	Existing system	Proposed system
Image with One pixel	00:00:03.4944062	00:00:01.4976026
Image with one face	00:00:21.7932383	00:00:20.4672359
Image with 2 faces	00:00:07.4688828	00:00:05.1268666
Image with 3 faces	00:00:09.0168159	00:00:06.7392118
Image with multiple faces	00:00:10.2648180	00:00:08.3772147

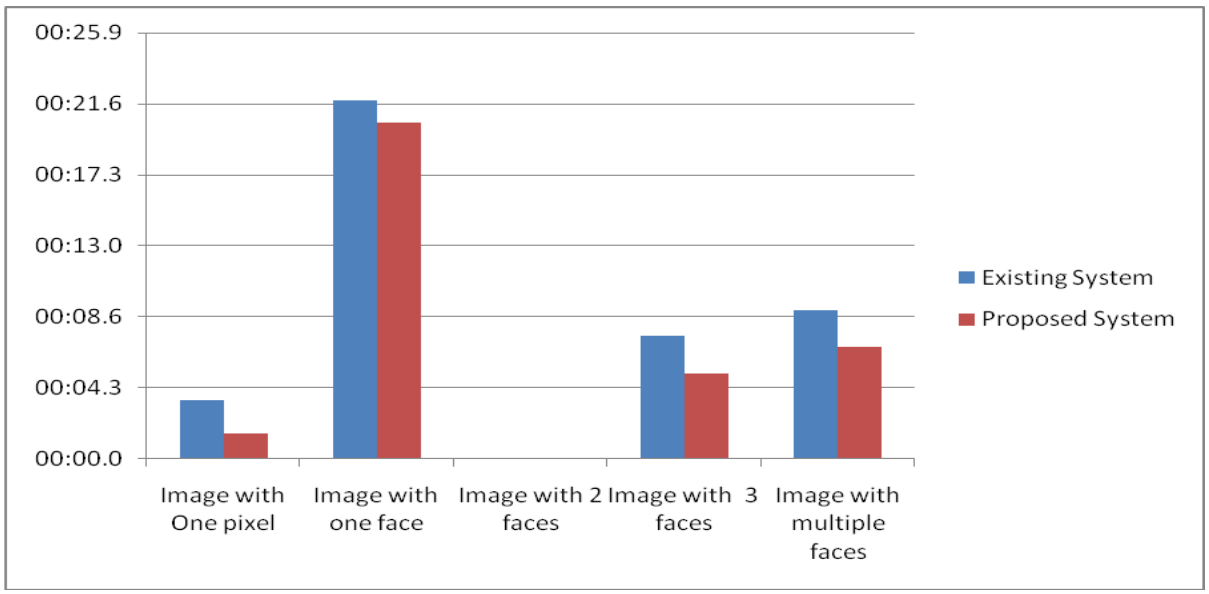

Fig 6: Comparison of Processing Time for existing and proposed system

V-D Effect of mask size on edge performance

The choice of filter mask size depends on the image characteristics. The large size of mask corresponds to higher value of variance, it results in smoothing but degrades the edge performance. Likewise the small sized filter mask best suits for textures and true edges. Also the small sized mask requires less time for block processing. The size of the mask must be chosen as small as possible that able to detect all psycho-visually important edges.

Fig7 Comparison of PSNR for existing and proposed system

VI. 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, we presented a novel distributed Canny edge detection algorithm which has the ability to compute edges of multiple blocks at the same time. To support this, an adaptive threshold selection method is proposed that predicts the high and low thresholds of the entire image while only processing the pixels of an individual block. In this we developed the threshold segmentation based scheme to improve performance of distributed canny edge detector. The theoretical analysis shows that threshold segmentation can improve the edge detection accuracy. The high and low threshold of image can vary in adaptive manner according to the context contained in the image. The use of small size structural operator responses to precise edge structures with reduction in noise level. The comparison of results is shown to analyze effectiveness of the approach

REFERENCES

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

[2] L. Torres, M. Robert, E. Bourennane, and M. Paindavoine, "Implementation of a recursive real time edge detector using retiming technique," *in Proc. Asia South Pacific IFIP Int. Conf. Very Large Scale Integr.*, 1995, pp. 811–816.

[3] F. G. Lorca, L. Kessal, and D. Demigny, "Efficient ASIC and FPGAimplementation of IIR filters for real time edge detection," in *Proc.IEEE ICIP*, vol. 2. Oct. 1997, pp. 406–409.

[4] J. F. Canny, "A computation approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 8, no. 6, pp. 769–798, Nov. 1986.

[5] H. Neoh and A. Hazanchuck, "Adaptive edge detection for real-timevideo processing using FPGAs," Altera Corp., San Jose, CA, USA,Application Note, 2005.

[6] P. Bao, L. Zhang, and X. Wu, "Canny edge detection enhancement by scale multiplication," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 9, pp. 1485–1490, Sep. 2005.

[7] 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.

[8] W. He and K. Yuan, "An improved canny edge detector andits realization on FPGA," in *Proc. IEEE 7th WCICA*, Jun. 2008,pp. 6561–6564.

[9] Q. 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.

[10] Vinod Mall, Anil K. Roy, Suman K. Mitra, ShivanshuShukla, "Detection of Structural Tampering in a DigitalImage Using Canny Edge Detector" 2013 IEEE

[11] QianXu, SrenivasVaradarajan, ChaitaliChakrabarti"A Distributed Canny Edge Detector: Algorithmand FPGA Implementation"*IEEE Trans. on Image Processing.*vol. 23,no.7, Jul. 2014.

