```
else if texture block type
        P1 = 0.03;          /* Few edges*/
else if texture/edge block type
        P1 = 0.1;           /* Some edges*/
else if medium edge block type
        P1 = 0.2;           /* Medium edges*/
else
        P1 = 0.4;           /* Many edges*/
```

Step 2: Compute the 8-bin non-uniform gradient magnitude Histogram and the corresponding cumulative distribution function F (G).

Step 3: Compute High threshold as

F (High threshold)= 1-P1

Step 4: Compute Low threshold = 0.4*High threshold

The pixel classified into 3, uniform, texture and edge. The uniform and edge pixel will count using the counters. The result is the $N_{uniform}$ and N_{edge} . According to the $N_{uniform}$ and N_{edge} the blocks will classify. For each type block there will be a P1 value, which is used for calculating the threshold values.

3. *Gradient and Magnitude Calculation*: To find the Gradient and Magnitude the entire image divides into 3×3 overlapping block. To find the x and y direction gradient these 3×3 blocks will multiplied with the matrix KGx and KGy [1].

$$KGx=\begin{pmatrix}-1 & 0 & 1\\ -2 & 0 & 2\\ -1 & 0 & 1\end{pmatrix}$$

$$KGy=\begin{pmatrix}1 & 2 & 1\\ 0 & 0 & 0\\ -1 & -2 & -1\end{pmatrix}$$

The gradient magnitudes (also known as the edge strengths) can then be determined as a Euclidean distance measure by applying the law of Pythagoras Equation.

$$G=\sqrt{G_X^2+G_y^2}$$

$$|G| = |Gx| + |Gy|$$

Where: Gx and Gy are the gradients in the x- and y-directions respectively, G is the magnitude.

4. *Directional Non Maximum Suppression (NMS)*: The horizontal and vertical gradient and the gradient magnitude are fetched from local memory 2, 3 and 1, respectively; and used as input to the Arithmetic unit. According to horizontal gradient G_x and the vertical gradient G_y, two intermediate gradients M1 and M2 will calculate. The M1 and M2 values will give to the Arithmetic unit. This arithmetic unit consists of one divider, two multipliers and one adder.

$$M_1 (x,y) = (1-d).M(x+1,y-1)+d.M(x+1,y);$$

$$M_2 (x,y) = (1-d).M(x-1,y-1)+d.M(x-1,y);$$

Where: $d=G_{y\,(x,y)} /G_{x(x,y)}$

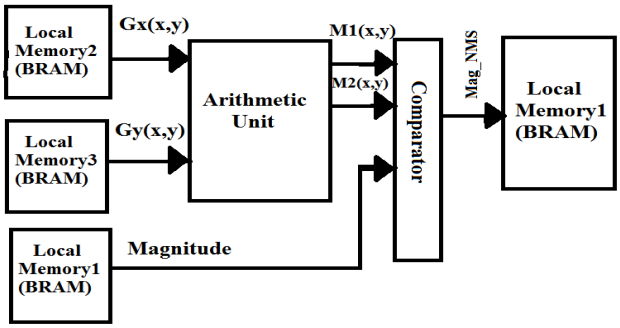

Fig 3: Directional Non Maximum Suppression Unit

Finally, the output of the arithmetic unit is compared with the gradient magnitude of the center pixel. The gradient of the pixel that does not correspond to a local maximum gradient magnitude is set to zero. The latency between the first input and the first output is 20 clock cycles and the total execution time is $m \times m$ +20.

5. *Calculation of Thresholds*: NMS is used for finding the threshold values, this unit can be pipelined with the directional NMS unit. Besides, the P1value, which is determined by the block classification unit, the mag_max, and mag_min, which are determined by the gradient and magnitude calculation unit, are the inputs for this unit.

The arithmetical unit can compute the corresponding R_i which is the high threshold T_H. Finally the low threshold T_L is 40% of the high threshold.

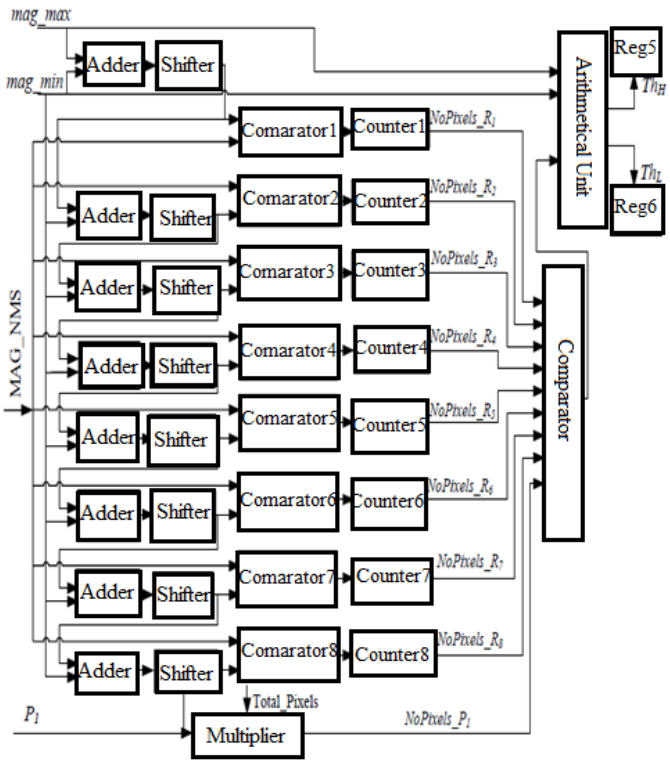

Fig 4: The architecture of thresholds calculation unit.

6. *Thresholding with Hysteresis:* The NMS value will compare with the high and low threshold values T_H and T_L. The NMS value below the T_H will change to 0 and the value above the T_L value wills change to 0. The remaining values will change to the maximum pixel value, typically 255.

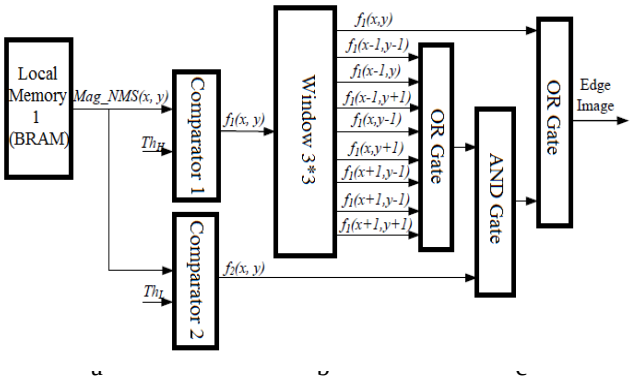

Fig 6: (a) image with noise, (b) histogram output, (c)edge images

V SIMULATION OUTPUT

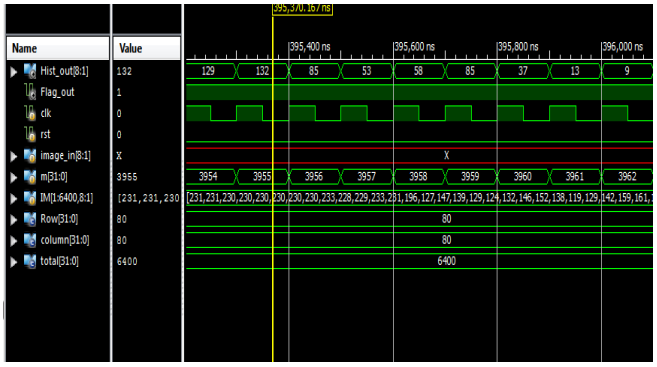

Fig 7: Output waveform of Histogram equalization

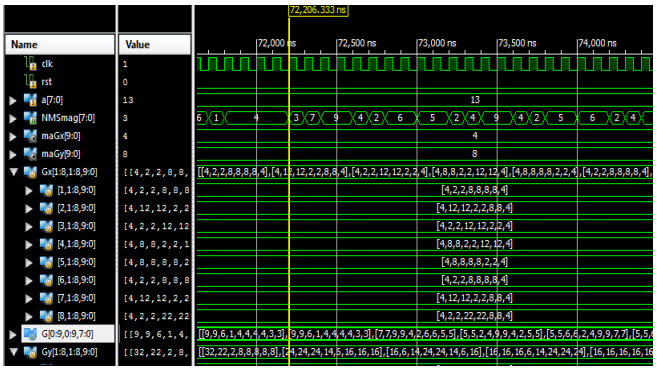

Fig 8: Vertical gradient magnitude

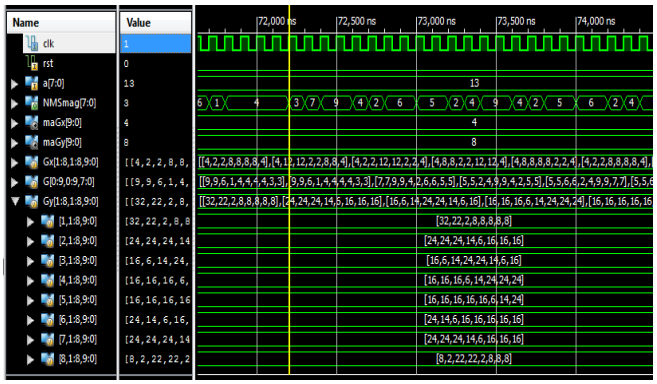

Fig 9: Horizontal gradient magnitude

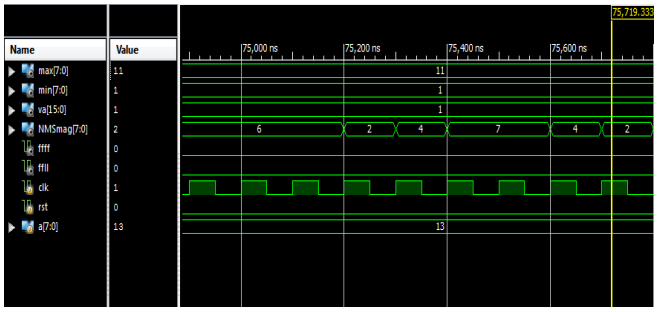

Fig 10: Output waveform of Directional NMS unit

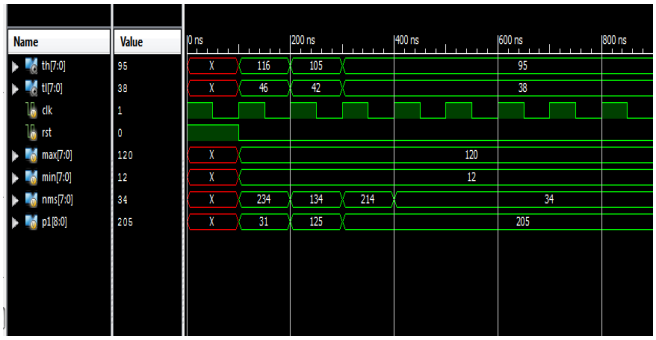

Fig 10: Output waveform of Threshold unit

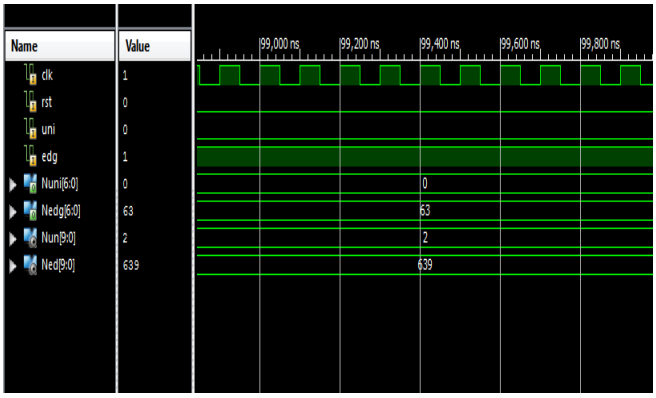

Fig 11: Output waveform of Block classification

Fig 12: Output waveform of Hysteris unit

(a) (b) (c)

Fig 13(a) Low clarity input image (b) histogram output (c) Edge of the input image

VI CONCLUSION

The original canny algorithm is a frame level processing to detect the high and low threshold value. In the proposed canny edge detection algorithm, it's a block level processing. In this Histogram equalization is applied for getting good clarity image. The histogram equalization will remove the noise in the image. In the future the shake image edges can be detected. The proposed system can applied for finger print detection, face reorganization etc.

REFERENCES

[1]. Qian Xu, Srenivas Varadarajan, Chaitali Chakrabarti Distributed Canny Edge Detector: Algorithm and FPGA Implementation, *Fellow, IEEE*, and Lina J. Karam, *Fellow, IEEE*

[2]. D. V. Rao and M. Venkatesan, "An Efficient Reconfigurable Architecture and Implementation of Edge Detection Algorithm using Handle-C," *IEEE Conference on Information Technology: Coding and Computing* (*ITCC*), vol. 2, pp. 843 – 847, Apr. 2004.

[3]. H. Neoh, A. Hazanchuck, "Adaptive Edge Detection for Real-Time Video Processing using FPGAs," *Application notes*, Altera Corporation, 2005. Online at http://www.altera.com/

[4]. C. Gentsos, C. Sotiropoulou, S. Nikolaidis, N. Vassiliadis, "Real-Time Canny Edge Detection Parallel Implementation for FPGAs," *IEEE International Conference on Electronics, Circuits and Systems* (*ICECS*), pp. 499-502, Dec. 2010

[5]. Y. Luo and R. Duraiswami, "Canny edge detection on nvidia cuda," *Computer Vision and Pattern Recognition Workshop*, vol. 0, pp. 1–8, 2008.

[6]. R. Palomar, J. M. Palomares, J. M. Castillo, J. Olivares, and J. G´omez- Luna, "Parallelizing and optimizing lip-canny using nvidia cuda," *ser. IEA/AIE'10*, Berlin, Heidelberg: Springer-Verlag, pp. 389–398, 2010.

[7]. L.H.A. Lourenco, "Efficient Implementation of Canny Edge Detection Filter for ITK Using CUDA ," *13th Symposium on Computer Systems*, pp 33-40, 2012.

[8]. J.F. Canny, "A Computation Approach to Edge Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, no 6, pp. 769-798, November 1986.

[9]. J.K. Su and R.M. Mersereau, "Post-processing for Artifact Reduction in JPEG-Compressed Images," *IEEE International Conference on Acoustics, Speech, and Signal Processing* (*ICASSP*), vol. 3, pp. 2363-2366, 1995.

Rinjo A J received the B.Tech in Electronics and Communication Engineering Degree from Thejus Engineering college, Vellarakkad, under the University of Calicut, Kerala, India in 2013, and doing M.Tech degree (2014-16) in VLSI Design Department of ECE, from Nehru College Of Engineering and Research Centre, Pampady, under the University of Calicut, Kerala, India

Co-author:

Remya K P Assistant Professor Nehru College Of Engineering and Research Centre, Pampady, Thrissur, Kerala, India. B.Tech in CUSAT. M.Tech in college of engineering Chegannur