



**International Journal of Nanotechnology**

ISSN online: 1741-8151 - ISSN print: 1475-7435

<https://www.inderscience.com/ijnt>

---

**Low area FPGA implementation of modified histogram estimation architecture with CSAC-DPROM-OBC for medical image enhancement application**

Koteswar Rao Bonagiri, Giri Babu Kande, P. Chandrasekhar Reddy

**DOI:** [10.1504/IJNT.2022.10050245](https://doi.org/10.1504/IJNT.2022.10050245)

**Article History:**

Received:	02 November 2021
Last revised:	15 December 2021
Accepted:	31 December 2021
Published online:	31 May 2023

---

## Low area FPGA implementation of modified histogram estimation architecture with CSAC-DPROM-OBC for medical image enhancement application

---

Koteswar Rao Bonagiri\*

Department of Electronics and Communication Engineering,  
Jawaharlal Nehru Technological University,  
500085 Hyderabad, India

and

Marrilaxman Reddy Institute of Technology and Management,  
500043 Telangana, India

Email: bonagirikoteswarrao@yahoo.com

Email: bonagirikoteswarrao@gmail.com

\*Corresponding author

Giri Babu Kande

Department of Electronics and Communication Engineering,  
Vasireddy Venkatadri Institute of Technology,  
522508 Andhrapradesh, India  
Email: giribabukande@gmail.com

P. Chandrasekhar Reddy

Department of Electronics and Communication Engineering,  
Jawaharlal Nehru Technological University,  
500085 Hyderabad, India  
Email: drpcsreddy@gmail.com

**Abstract:** In this work, modified histogram estimation (MHE) architecture is proposed to verify the histogram count in the FPGA platform, and the Basic HE (BHE) architecture is also implemented for comparative purpose. The entire proposed MHE architecture is developed newly so as to reduce the logical elements involved in the HE process. In MHE architecture, dual port read only memory (DPROM), carry select adder based counter (CSAC), and Optimal Bin Counter (OBC) are used to evaluate the HE count with effective accuracy. The amount of percentage reduced by the 256 sample MHE is 17.62%, 15.41% and 23.01% for area, power and delay respectively. Additionally, the performance of the proposed MHE is compared with four existing methods HOG, HBS, MBPA and DMH. The number of flip flops utilised by the MHE architecture is 2177 for Vertex 6 device, which is less compared to the HOG and MBPA.

**Keywords:** area; basic histogram estimation; CSAC; carry select adder based counter; delay; DPROM; dual port read only memory; field programmable gate array; medical image enhancement; MHE; modified histogram estimation; optimal bin counter; power.

**Reference** to this paper should be made as follows: Bonagiri, K.R., Kande, G.B. and Chandrasekhar Reddy, P. (2023) 'Low area FPGA implementation of modified histogram estimation architecture with CSAC-DPROM-OBC for medical image enhancement application', *Int. J. Nanotechnol.*, Vol. 20, Nos. 1/2/3/4, pp.259–280.

**Biographical notes:** B. Koteswar Rao received BTech (ECE) from Nalla Malla Reddy Engineering College, Hyderabad, Telangana in 2006 and MTech (Electronics and Communication Engineering ) from Madhira Institute of Technology and Science , Kodad, Telangana in 2010 and now pursuing PhD in VLSI from Jawaharlal Nehru Technological University Hyderabad since 2016, Currently working as an Associate Professor in the Department of ECE in Marri laxman Reddy Institute of Technology and Management, Telangana. He has published many Research papers in National/International Journals.

Kande Giri Babu presently working as a Professor (ECE Dept) and Dean of Studies in Vasireddy Venkatadri Institute of Technology, Guntur, AP. He received BTech Degree from Nagarjuna University from 1992–1996 and ME (Electronic instrumentation) from Andhra University . He did his Ph.D from JNTU Hyderabad in 2010. He has published many Research papers in National/International Journals/Conferences and guiding six research scholars.

P. Chandrasekhar Reddy, presently he is working as a Professor and BOS chairperson ECE dept , in Jawaharlal Nehru Technological university Hyderabad. He received B.Tech Degree from JNTU from 1983–1987 and M.Tech Applied Electronics from Baratiar University He did his PhD from JNTU. He is also worked as Head of the Department for the Electronics and Communication Engineering and Computer Science &Engineering in JNTU Anantapur. He has published many Research papers in National /International Journal/Conferences and guiding six research scholars.

---

## 1 Introduction

Generally, histogram provides an approximate visual interpretation of discrete or continuous data [1]. Histogram of a grey scale image contains 256 grey levels and its hardware implementation makes it suitable in medical and military applications. histograms are used in a wide range of applications. In digital image processing it plays a vital role in colour image segmentation and also provide better brightness and contrast to an image [2]. Technical calculation and higher area are required when the applications of image processing are integrated into the portable devices. Therefore, the low complexity design is required because of the restricted battery capacity and less size of portable devices [3]. Adaptive Fuzzy Filter based on Histogram Estimation (AFHE) was developed to remove the salt and pepper noise from the colour image [4]. Histograms estimation for RGB model is computed faster and requires a little memory. If the image size is  $512 \times 512$  pixels, it required more memory to store histogram. So in Wei and Lin [5], the fast and memory efficient multivariate histogram computation method is introduced and later pipelining concept were implemented [6].

Denisova and Sergeyevev [7] introduced an efficient colour-based method that depends on the histogram feature for person re-identification issue in video surveillance due to large illumination variations, low resolution and insufficient viewing [8]. Further, Segmented Histogram Object Recognition Technique (SHORT) was implemented to create impact in real-time applications and it helped to identify the similar objects in database [9]. The FPGA hardware implementation proves that SHORT is 28X faster and suitable for real-time. An exploratory synthesis revealed that in CLAHE [10] process the image sequence from infrared cameras with high dynamic range and some other artefacts are introduced in Dutton et al. [11]. The feature like single shot exposure time is impossible due to jitter in CMOS sensors which are reconstructed with pixel wise histogram streaming. But, it does not improve the depth accuracy due to large amount of uncorrelated noise [12]. Shifted inter frame histogram [13] produced a remarkable robustness for low SNR. Histogram also relies on TDC implementations, which significantly increase the conversion rate but lags in providing accuracy that may be improved with more advanced FPGA [14]. Basically, a histogram is a density estimator where PDF is estimated by rescaling histogram data. Even though the performance loss is minimised in PDF, there was no proper design about bin count to estimate the histogram count in FPGA platform [15]. The major contributions of the research work are given as follows:

- The entire architecture is designed newly to evaluate the histogram of the input images. The BHE and MHE architectures are implemented in this research work.
- In this work, 9 samples and 256 sample input values are undergone in the HE process. So, two different architectures are designed to perform histogram and the estimated values are shown on the Modelsim waveform window.
- In MHE, DPROM, CSAC, and OBC are used to design the HE with more accuracy in the estimation window.
- Due to the usage of proposed logical elements architecture, the ASIC and FPGA performances are reduced in MHE architecture.
- Moreover, the MHE is used to remove the noises which are present in the medical images. The image enhancement process is implemented for the Brain MRI, Lungs CT, and Mammogram.

The organisation of the paper is given as follows: the literature review about the existing histogram estimation architectures developed in the FPGA platform is given in Section 2. The description about the existing BHE architecture with its problem statement is described in Section 3. The clear description about the MHE architecture with its sub modules architecture is provided in Section 4. The ASIC and FPGA performances of the MHE architecture along with existing architecture is evaluated in Section 5. Finally, the conclusion is made in Section 6.

## **2 Related works**

Ghaffari et al. [16] analysed a FPGA based feature extraction algorithm, named Histogram of Oriented Gradients (HOG) with four principles and different improvement techniques in every class. The first group was represented as the optimisation of the

algorithm. The subsequent class was used for information control procedures which incorporate numerical portrayal. The third gathering modified the features for HOG, and the fourth one was the usage of appropriate platform. Based on the concerned applications and scenarios, the system uses the pure FPGA implementations or HW/SW co-designs. The survey helped to separate the algorithm into parts for each platform to get good performance results. Most of the works in HOG algorithm were approximated and simplified using interpolation in bin assignments (smoothing histogram curve) and normalisation steps to gain high speed. However, the proposed method was failed as the normalisation process was the most time consuming and became a hurdle to achieve high speed algorithm with high accuracy.

Hazra et al. [17] discussed the concept of FPGA in the reversible watermarking algorithm by Histogram Bin Shift (HBS). After processing the embedding and decoding procedure in reversible mode process, it extracted both the original image and watermark at the receiver end. In a Xilinx system generator, watermarking scheme was involved and a brief survey of hardware architectures were required for the embedding and extraction processes with maximum operating frequency of 445.330 MHz and 201.824 MHz. The power consumption was found to be low (i.e.) 1.215 W and 0.104 W. The hardware based simulation for the full image was carried out on 2 FPGA platforms, Spartan 3E (for extraction) and Virtex-7 (for embedding). Compared to software based module, it achieved an inherent speed which was incorporated in test equipments like MRI scan, CT scan and X-ray whose test result was given in greyscale format. Thus, this scheme was incorporated only for grey scale images and not applicable for colour images.

Mondal and Banerjee [18] presented a module to compute the histogram by fast VLSI architecture. The Memory-Based Parallel Algorithm (MBPA) was used to perform the histogram computation and it was possible to implement different logic modules. It consequently accelerated the computation of joint histogram which favoured the computation of mutual information for similarity measurements. If the similarity among the neighbourhood data increases, the total number of clock cycles to compute histogram was reduced. The histogram computation for an image of size  $480 \times 640$  was examined; whose critical path delay was reduced to  $34\tau$  when compared to parallel array histograms critical path delay (i.e.)  $101\tau$ . The hardware utilisation of the MBPA architecture was 99.66% which is less compared to latest architecture and it consumed 8.78 mW power. Reconfiguration time span was not tolerable and considered to be bottleneck, because it required more number of clock cycles to re-compute the histogram and increased the critical path delay which directly affected the speed of architecture.

Yang et al. [19] proposed a FPGA implementation of an efficient sliced integral histogram algorithm with two key technologies namely, local integral histogram (LIH) and buffered integral histogram (BIH). The entire image was divided into 'S' non overlapping slices and each slices are independent to each other. So, BIH was used to store the global IH at lower boundaries which saved 21.2% of storage than conventional algorithm. The FPGA platform was considered for hardware implementations where its functionality and system performance evaluated. The performance metrics like speed, storage capacity and power consumption of SIH algorithm were compared with IIC and FII algorithms. In the comparison, high resolution image ( $1280 \times 720$ ) SIH algorithm was 3–6 times faster which saved 1962–3157 count LE's with minimum power consumption of 334.41 mW than the IIC and FII algorithm. A line buffer of length 'W' was introduced to buffer the meantime results which exhausted the hardware resources if the slice number was getting higher. It significantly affected the resource utilisation and

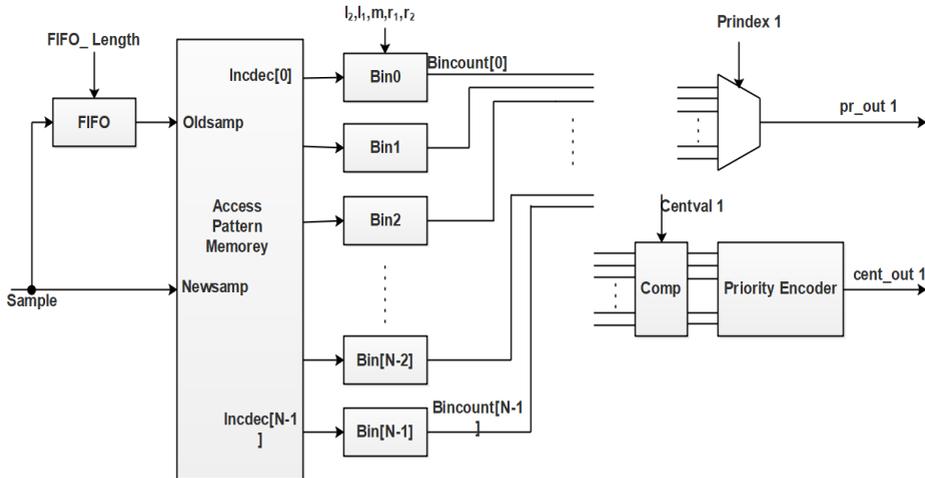
performance. Hence, this approach was inadequate to keep a balance between performance and hardware resources in real-time.

Bannoy et al. [20] explained the concept of face recognition with high speed FPGA implementation and Distance between Multiple Histogram (DMH). In this algorithm, from each row of query and database images, distance metric (i.e.) sum of standard deviations between multiple histograms were calculated. Gamma correction method was used to conclude that standard deviation computation was best option to compare the query and database. The database face matched query with lowest distance score which was independent of illumination and outperforms better than the face recognition algorithm “Eigen faces”. Even if the ambient illumination changed with facial expression, this module identified the face 16 times faster than Eigen face algorithm. The operating frequency of the FPGA Zed-Board is 100 MHz which helped to increase the speed of the entire architecture. This hardware accelerator utilised only 18% of look up tables, 11% of flip flop and worked 10× time faster than software version and 160× time faster than Eigen face. Furthermore, this concept was tested with sidelight (left or right) effects on the face, but it failed to recognise the face with sidelight (left or right) effects due to the asymmetrical change in histogram.

### 3 Conventional architecture

The existing PDF architecture of the histogram estimation is shown in Figure 1 which contains the First In First Out (FIFO), Access Pattern Memory (APM), bin count, and priority encoder [15]. Based on the FIFO length, the input sample is given to the FIFO which generate the output for every clock cycle. As per the architecture design, old sample and the new sample were connected to the APM module. The APM module has designed by the Read Only Memory (ROM) incremented and decremented modules. After performing the incremented/decremented operation, the output has connected to the Bin module.

Figure 1 Conventional histogram estimation architecture



Bin module helped to find the histogram values of the input samples. In Figure 1, the bin count process was evaluated for all the samples. These bin count outputs were connected to the comparator to perform the comparison with the centre value. The comparative results were taken and processed the priority encoder operation [2, 15].

### 3.1 Problem identification

In recent years, most of the HE research works were developed in the MATLAB platform and only few research works have been developed in the topic of HE in the FPGA platform. So, it is quite difficult to implement the HE in Verilog language. While designing the HE in FPGA platform, following problems have been encountered.

- Bin count architecture module was not given properly to estimate the histogram count. The process of each and every step did not explain clearly. Due to the unavailability of explanation of the bin count, the researchers have been suffered a lot to find the HE.
- For designing the APM, the depended modules are not perfectly matched to evaluate the histogram count. More logical elements have been used to design the APM module.
- Normally, most of the HE research works have been shown in the MATLAB platform only. No other research works have been developed in FPGA platform to visualise the histogram count in the simulation waveform.
- For storing the old sample and new sample operation in APM, two ROM have been used in the architecture.
- The hardware utilisation of the HE architecture is high due to the usage of more logical elements, ROM, and bin count.

#### *Solution:*

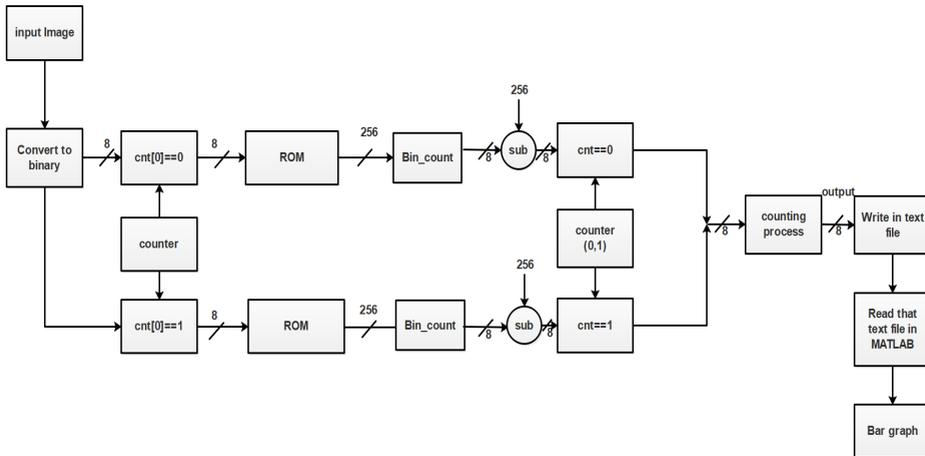
To overcome the aforementioned drawbacks, MHE is introduced in this research work. Along with the MHE architecture, BHE architecture is also implemented to evaluate the histogram count for analysis purpose. Both the architectures were implemented using Verilog language and the FPGA and the pattern values are stored in the DPROM helps to analyses the pattern process for old and new sample values. CSLA based counter is used to count the values for every clock cycle with less hardware utilisation. Optimal bin counter is used to analyse the number of 1's present in the DPROM output.

## 4 Basic histogram estimation architecture (BHE)

- The architecture of BHE is shown in Figure 2. The Input pixel is read and converted to binary value with the help of MATLAB.
- If counter LSB is equal to 0 means, the input value goes to upper ROM.
- If counter LSB is equal to 1 means, the input value goes to lower ROM.
- Normal adder is used to design the 8-bit binary counter

- Upper and lower 256 bits are going to perform the bin count operation.
- In bin count, one’s count operation has been performed. Example. If input is 1011011 means output is 5. This bin count has less efficiency and uses more resources.
- Bin count results are subtracted with 256 decimal value. Normally, 1-bit counter is used in output side.
- If counter is 0, above value is delivered to the output side. If counter is 1, below value is move to output side.
- This result is given to the counting process. This results stored in text file is used to get histogram bar. The bar graph is generated with the help of ‘bar’ function in the MATLAB. The detail explanation of the MHE architecture with its equation is explained in 4.1.

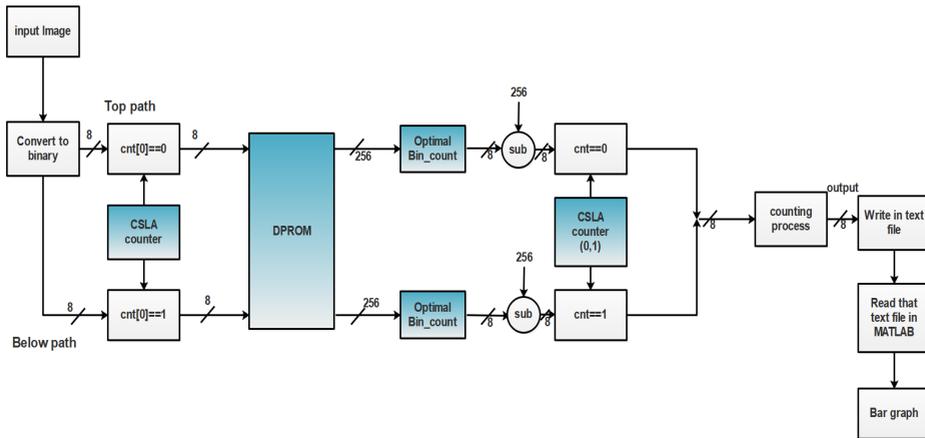
**Figure 2** BHE architecture process



*4.1 Modified histogram estimation architecture (MHE)*

The MHE architecture is shown in Figure 3 which contains the CSLA counter, DPROM, OBC, counting process modules. In this section, 9 samples HE process and 256 samples HE process is explained. Initially, 3x3 size of input image is read in MATLAB to calculate the HE process in FPGA platform. The 3x3 image contains 9 samples as well as 16x16 size image contains the 256 samples. With the help of “dec2bin” function in MATLAB, the pixel values are converted into the binary values and each pixel contains 8 bit of binary data which is stored in text file. This stored text file is given to the VLSI platform to evaluate the HE count. The 9 sample pixel values are shown in Figure 4. Here, 9s ( 0 ) memory is stored “00000001” binary values as well as 9s ( 7 ) memory is stored “00000001”.

**Figure 3** MHE architecture process (see online version for colours)



**Figure 4** Input values of 3×3 image

00000001  
 00000011  
 00000011  
 00000010  
 00000010  
 00000001  
 00000010  
 00000001

where

9s ( 0 ): Initial sample stored in 0th memory

9s ( 7 ): Last sample stored in 8th memory.

From the 9 samples, the histogram count evaluation process is performed in FPGA platform. Initially, the counter is designed to count the values from 0 to 8 based on the pixel values. If the image contains the 9 sample, the counter is incremented from 0 to 8. If the image contains 256 samples, the counter is incremented from 0 to 255. So, 8 bit counters have designed in this research work. The 9 sample counter values and 256 samples counter values are given in equations (1) and (2).

$$\text{counter\_9\_sample\_out} = \{0,1,2,3,4,5,6,7,8\} \tag{1}$$

$$\text{counter\_256\_sample\_out} = \{0,1,2,\dots\dots\dots255\} \tag{2}$$

Based on the counter least significant bit (LSB), the input samples are crossed the counting process, which are given equations (3) and (4).

$$\begin{aligned} &\text{if}(\text{cnt}_{\text{out}}[0] == 4'd0) \\ &D1 = X \end{aligned} \tag{3}$$

$$\begin{aligned} &\text{if}(\text{cnt}_{\text{out}}[0] == 4'd1) \\ &D2 = X \end{aligned} \tag{4}$$

where

$\text{cnt}_{\text{out}}[0]$ : LSB bit of counter output

X: input samples

D1: current sample

D2: previous sample.

If the counter LSB is equal to zero, the input is considered as current sample and it choose the top path as shown in Figure 3. Similarly, if the counter LSB is equal to one, the input is considered as previous sample and it choose the below path in Figure 3. In every clock cycle the counter LSB is changed to 0 and 1. Based on the clock cycle, the previous and current sample values are connected to the DPROM to perform the pattern operation. In DPROM module, the 8-bit new sample and 8-bit old sample data are stored in DPROM and the main advantage of this method is that the old and new samples are stored in single DPROM. Each and every memory (M) like M [0], M [1], up-to M [255] can be stored in 256-bits. The pattern values are given in Table 1.

**Table 1** Patter values of 255 memory

Address	Pattern
0	111111.....1111111
1	011111.....1111111
2	001111.....1111111
3	000111.....1111111
4	000011.....1111111
0	0
0	0
0	0
254	000000.....0000011
255	000000.....0000001

At initial clock cycle, 9s [0] and 9s [1] samples are given to the DPROM to perform the pattern process. 9s [0] memory and 9s [1] memory contains the decimal value of 1 and 3 respectively. Initially, these two values are connected to the DPROM and the output is given in equations (5) and (6).

$$9s[0]_{DPROM\_out} = 111111.....1111111 \tag{5}$$

$$9s[1]_{DROM_{out}} = 011111\dots\dots\dots111111 \tag{6}$$

Each samples are performed the pattern process with respect to the clock signal. Based on the sample size, the output count of the DPRROM is evaluated. In this research work, 9 and 256 samples undergo the pattern process. So, the DPRROM output count are 9 and 256 respectively for the proposed architecture. This 255 bit outputs are connected to the optimal bin count value module which helps to count the number of 1’s present in the 256 bit values.

The BHE and MHE bin count process is shown in Figures 5 and 6. For both the process, DPRROM output (256 bit) is given to the input of bin count process. Among the 256 bit, each and every bit is checking whether “1” is present in the respective bit or not. In both the processes, “FOR loop” is used to fetch the entire 256 bits. In BHE architecture, every bit checks the “If statement” to confirm which output is stored in “ones” variable. But, there is no usage of “if statement” in MHE architecture. In MHE, the bit values are directly undergone the one count process. Due to this, the MHE architecture occupied less resources. The BHE and MHE bin count output generation process is given in equations (7) and (8).

$$BHE_{BC} = ones + 1 \tag{7}$$

$$MHE_{OBC} = ones + DO [i] \tag{8}$$

where

$BHE_{BC}$ : bin count output of BHE architecture

$MHE_{OBC}$ : optimal bin count output of MHE architecture.

The output of the bin count is considered as 8 bit which performs the subtraction operation with 256 maximum value. The subtraction output is given in equations (9) and (10).

$$9s[0]sub_{out} = 256 - MHE_{OBC} \tag{9}$$

$$9s[1]sub_{out} = 256 - MHE_{OBC} \tag{10}$$

After performing the subtraction, carry select adder based counter (CSAC) is designed to count 0 and 1 respectively as shown in equations (11) and (12)

$$\begin{aligned} &\text{if}(cnt_{out} == 4'd0) \\ &F\_out\_1 = 9s[1]sub_{out} \end{aligned} \tag{11}$$

$$\begin{aligned} &\text{if}(cnt_{out} == 4'd1) \\ &F\_out\_2 = 9s[0]sub_{out} \end{aligned} \tag{12}$$

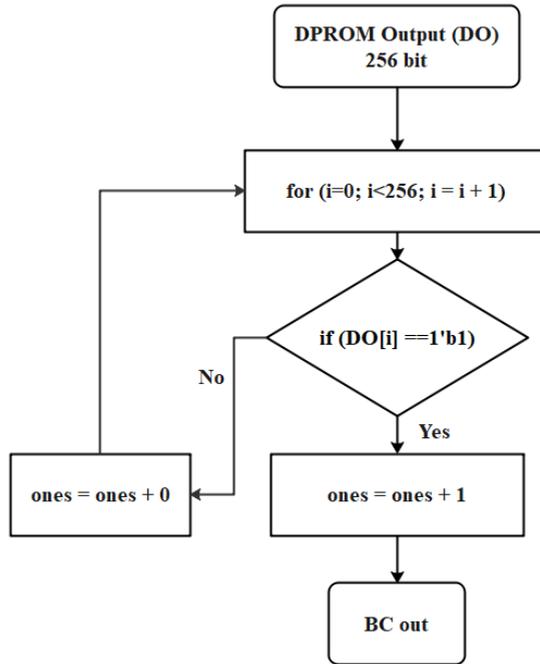
where

$F\_out\_1$ : 1st memory subtraction output

$F\_out\_2$ : 0th memory subtraction output.

Based on the counter value, the output of the subtraction performs the counting process operation.

**Figure 5** BHE bin count process



**Figure 6** MHE optimal bin count process

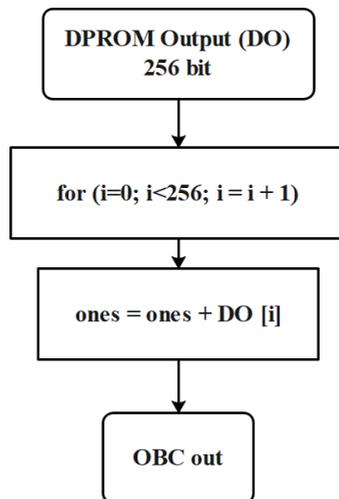
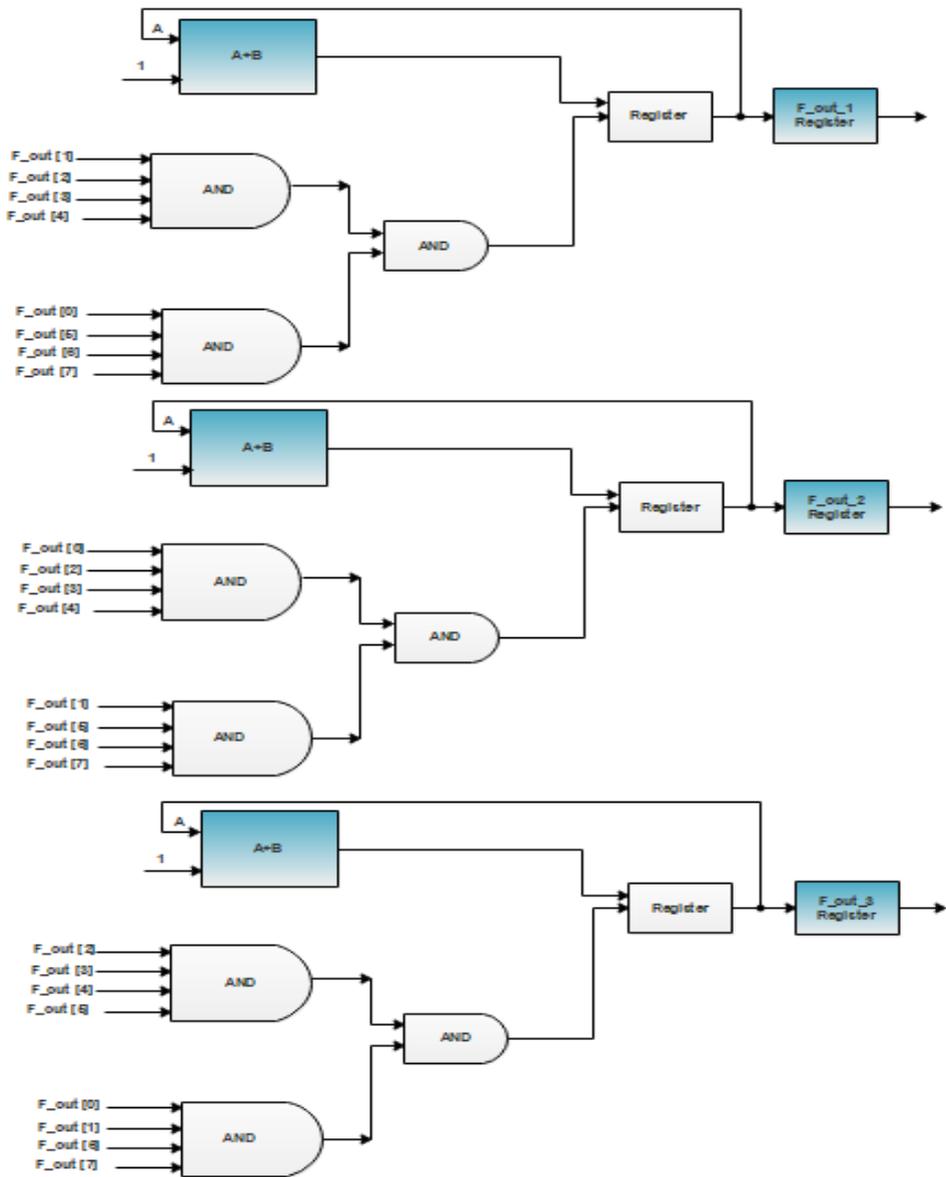


Figure 7 shows the architecture of counting process which is already given as a block in Figure 3. The binary values of image pixels are given as input to the AND gates for counting process. This counting process provides three different outputs such as

$F\_out\_1Register$ ,  $F\_out\_2Register$  and  $F\_out\_3Register$  from three different stages. The pixel density of 1, 2 and 3 are given to the first, second and third stage respectively. Here, the register is initialised to provide pixel density in the respective stages. For example, the operation of the first stage with the input of 00000001 (i.e., 8 bit) is given as follows: The given input is divided into two different inputs as 4bit and it is given to the two AND gates of first stage. In general, AND gate deliver the output of 1, only when all the inputs are 1. So, the output from the two AND gates are 0 as well as the AND operation between the 0 and 0 from 1st AND gate and 2nd gate is 0. Next, this value 0 is stored in the register and it is given as feedback to the operation of  $A+B$ . where,  $A$  is the value stored in the register and  $B$  is a constant value i.e., 1. The operation of  $A+B$  ( $0+1$ ) provides the value of 1 and this value 1 is replaced in the register instead of previous value 0. Subsequently, the value in the register is stored in the  $F\_out\_1Register$ . The same operation is processed up to the 9 samples, when the input image is in the size of  $3 \times 3$ . The operation of first stage is processed, only when the input pixel density is 1. Finally, the counting process delivers the amount of pixels in the image which has a density of 1. According to this 9 sample research work, pixel 1 is coming for 4 times, pixel 2 is coming for 3 times, and pixel number 3 is coming for 2 times which pictorial representation is shown in Figure 12. Similarly, the same counting process is developed for the 256 samples of  $16 \times 16$  image which has 256 stages. Based on the input bit size, the counting stage process need to be designed. For 8 bit input samples, 256 stages of registers are required. Because, the 8-bit combination values are generated from 0 to 255. For counter process, incremental operation need to perform. At that time, CSLA adder is used which is explained as follows.

The architecture of the CSLA is shown in Figure 8 and this CSLA contains a Multiplexer (MUX), Binary-to-Excess Converter (BEC), Ripple Carry Adder (RCA) and 4 Full Adder (FA) design. The operation of the CSLA is explained with the example input  $A=0100$  and  $B=0101$ . At initial stage, the  $A_0=0, B_0=1, A_1=0$  and  $B_1=0$  are given as input to the RCA circuit. The group 2 of CSLA has one 2-b RCA, that contains two FA for  $C_{in}=0$ . The output of the first FA is sum of 1 and carry of 0 as well as this carry value is given as input to the second FA. Subsequently, the 2nd FA provides the sum of 0 and carry of 0. Then the carry from the 2nd FA is given as input to the MUX while the output of RCA is equal to 01. Moreover, the  $A_2=1$  and  $B_2=1$  are given as input to the 1st FA that provides the sum of 0 and carry of 1 during the 2nd stage. The 1st FA's carry is 1,  $A_3=0$  and  $B_3=0$  are provided as input to the 2nd FA that provides sum of 1 and carry of 0. Next, the output of FA is given as input to the BEC as well as the important operation of the BEC is one-bit incremental operation. The output is 11, when the binary input of 10 is given to the BEC. Subsequently, the BEC and 2nd stage FA outputs 10 are given to the input of the MUX. The selection line is 0, when the output of MUX is 10 otherwise MUX delivers the output as 11. Furthermore, the concatenation between the 1st stage output (01) and MUX output (10) provides the value of 1001. The input arrival time is less when compared to the multiplexer selection input arrival time [21].

Figure 7 Counting process for 9 sample architecture (see online version for colours)



## 5 Results and discussion

The experimental results and discussion of the MHE architecture are described in this section. MATLAB R2018a is used to read the image and the image is resized into 3×3 and 16×16. Subsequently, the resized image is converted to binary value which is stored into .txt file. The architecture of MHE is implemented and simulated in the Modelsim 10.5 to verify the histogram estimation. The Verilog code of the MHE is incorporated in



Figure 10 256 samples

```

10100000
10010111
01100111
01110100
10000010
10000100
10000010
10000000
|
|
|
|
|
10111000
01100111
01101111
01011000
01001110
    
```

Tables 2–5 show the ASIC, percentage reduction, FPGA performances and frequency analysis of the MHE and BHE architectures. Tables 2–4 show that the MHE architecture provides better performance than the BHE architecture. The area of the MHE architecture is 96846  $\mu\text{m}^2$  which is less when compared to the existing BHE architecture. The reduction percentage achieved by the MHE in 9 samples are 48.03%, 68.76% and 29.61% for area, power and delay respectively. The RTL code is converted into netlist file to evaluate the area, power, and delay in the TSMC 180 nm technology. With the help of the cadence RTL compiler report menu option, the ASIC performances are calculated. Due to the optimal design of the MHE architecture, the performance values are improved than the BHE architecture. Moreover, the number of slice LUT used by the MHE architecture is 3388 which is less when compared to the BHE architecture. The hardware utilisation of the MHE architecture is minimised using the DPROM and CSLA adder. But, the existing BHE architecture requires two different ROM to store the previous and current sample operation in APM. Additionally, the more number of logical elements, bin count and inappropriate APM used in the BHE architecture increases the hardware utilisation of BHE. From the Table 5, knows that the operating frequency of the MHE architecture is higher than that of the BHE architecture. The operation speed is high and delay is less for the MHE architecture due to the higher operating frequency i.e., 112.335 MHz. Due to the usage of optimal bin count, DPROM and CSLA adder, the MHE frequency has been improved than BHE architecture.

Table 2 ASIC performance analysis using 180 nm technology

Architecture	Area ( $\mu\text{m}^2$ )	Power (W)	Delay (ps)	APP ( $\mu\text{m}^2 \times \text{W}$ )	ADP ( $\mu\text{m}^2 \times \text{ps}$ )
9 sample-BHE	186357	4.984	233	928803.2	43421181
9 sample-MHE	96846	1.557	164	150789.2	15882744
256 sample-BHE	5889206	8.245	265	48556503.4	1560639590
256 sample-MHE	4851472	6.974	204	33834165.7	989700288

**Table 3** Percentage of ASIC performance reduction by MHE

Parameters	Percentage of reduction by MHE	
	9 sample	256 sample
Area	48.03	17.62
Power	68.76	15.41
Delay	29.61	23.01

**Table 4** FPGA performance analysis using Virtex 6 device

FPGA device, Speed grade and package	FPGA performances	Total resources	Occupied resources		% of utilisation	
			BHE	MHE	BHE	MHE
XC6VCX75T, -2 and FF484	Number of slice registers	160000	4120	4116	2%	2%
	Number of slice LUTs	80000	13500	3388	16%	4%
	Number of fully used LUT-FF pairs	15554	2066	2127	13%	13%
	Block RAM	264	8	8	3%	3%

**Table 5** Frequency analysis of BHE and MHE

Architecture	Operating frequency
BHE	41.39 MHz
MHE	112.335 MHz

The simulation waveform output for 9 samples ( $3 \times 3$  image) is shown in Figure 11. The control signals for the MHE architecture are *clk*, *en*, and *rst*. The binary input given to the MHE architecture is specified as *x* which is the binary value of image pixel. The output from the DPRAM are represented as *Curr\_samp\_ROM out* and *Prev\_samp\_ROM out*. Here, *F\_out* is the output from the counting process. In *F\_out*, the pixel with density of 1, 2 and 3 are obtained for 4, 3 and 2 clock cycles respectively. Then the pixel density of 1, 2 and 3 are obtained from the *Final\_out\_1*, *Final\_out\_2*, and *Final\_out\_3* respectively. This simulation waveform of Figure 11 shows that the histogram is precisely estimated for the all 9 samples of  $3 \times 3$  image. Similarly, the MHE architecture is also analysed for the 256 samples of  $16 \times 16$  image. Moreover, the *F\_out* from the Verilog is given as input to the MATLAB to analyse the number of pixels with respect to the pixel density as shown in Figure 12. Similarly, the number of pixels with respect to the pixel density for 256 samples is shown in Figure 13. Figures 12 and 13 shows that the proposed MHE architecture precisely estimates the histogram of the pixels. The complexity of the model is increased when the input image size is increased. Because, the number of pixel values are increased and it required more registers to store the values. So, the complexity in depends on the input image size.

Figure 11 9 samples simulation waveform output (see online version for colours)

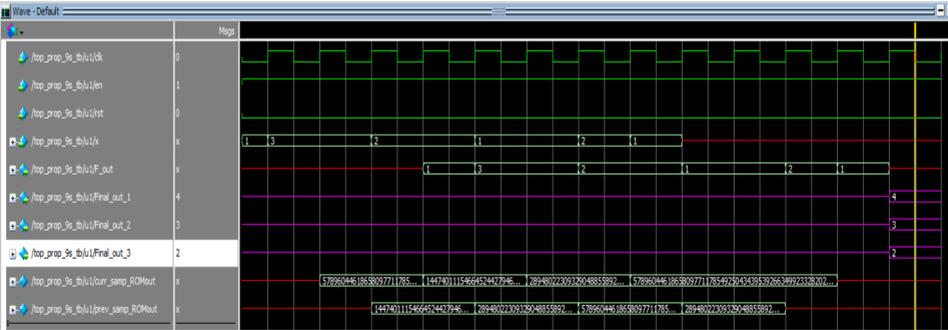


Figure 12 9 sample output (see online version for colours)

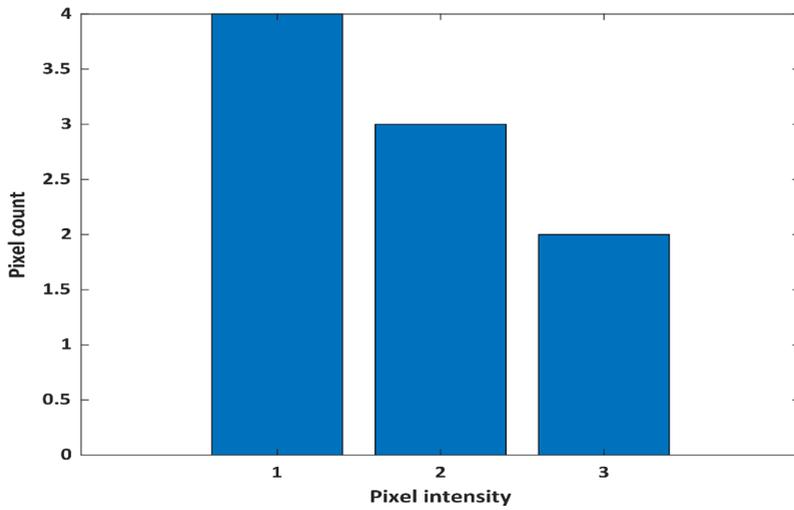
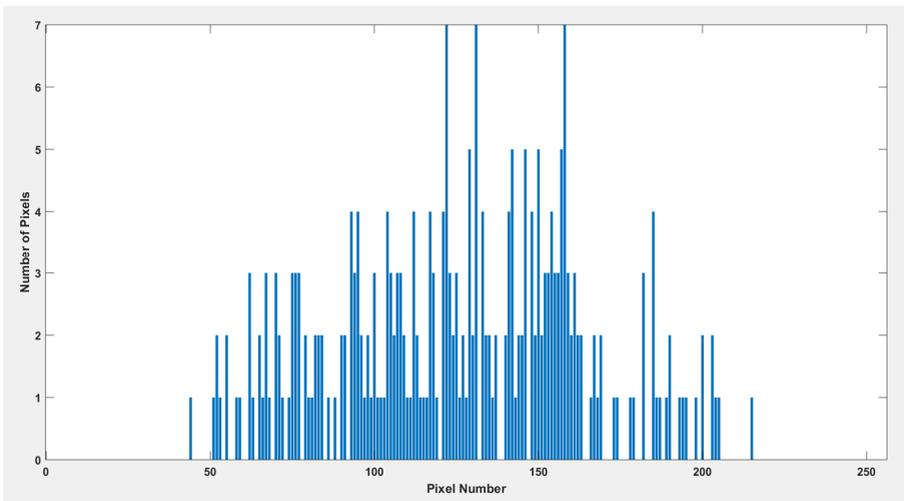


Figure 13 256 sample output (see online version for colours)



## 5.2 *Application of MHE in the medical field*

In the medical field, image enhancement plays a vital role to identify the each pixel clearly for monitoring the disease conditions [22, 23]. Normally, the generated medical images are contains the noises which does not provide the good quality. So, it is mandatory to de-noise the image for obtaining better quality of the image. In this case, histogram estimation is very important stage to identify the pixel value count which helps to process the further operation. After histogram estimation, the probability needs to be calculated for the pixel count. The probability helps to identify the number of occurrence pixel among the overall pixel count [24, 25]. Once the probability is done, the cumulative distribution operation is performed to get the optimal values of the pixel. The combination of two pixel probability values are added and produced the cumulative distribution values. Similarly, the cumulative distribution is performed for all the probability pixel values. From the results, the grey level value (255) is multiplied and produced the enhanced output values. From this, we have obtained enhanced quality of the medical image. This process is applied for the Brain MRI, Lungs CT, Mammogram images which results are shown in Table 10.

In Table 10, the different input images are taken and performed the image enhancement operation. The original image and its histogram as well as the enhanced image and its histograms are clearly included in the table. From this it is clears that the input medical images are perfectly enhanced and produced the output. This noise free medical images are helped the doctors to identify the problem of the patients clearly. The hardware utilisation of the various images results are the same only. Because, there is no variations in the FPGA module.

## 5.3 *Comparative analysis*

The comparative analysis of the proposed MHE architecture with existing architecture is described in this section. The MHE architecture is compared with four existing architectures such as HOG [16], HBS [17], MBPA [18] and DMH [20]. This comparative analysis is taken in terms of slices, LUT, flip flops, frequency, power.

Tables 6–9 shows the performance comparison of the MHE architecture for Virtex 6, Virtex 6 (MBPA), Virtex 7 and Zynq Z 7020 devices. Table 6 shows the comparison of MHE architecture with HOG [16] and Table 7 shows the comparison of MHE with MBPA [18] method. In Virtex 6, the number of BRAM in MHE architecture is 8 that is less when compared to the HOG [16] and the number of slices in MHE is 1478 which is less than MBPA [18] method. The power consumed by the MHE architecture is 1.041 W for Virtex 7 device, it is less when compared to the HBS [17]. Moreover, the number of LUT for Zynq Z 7020 device is 3456 which is less when compared to the DMH [20]. Hence, the aforementioned results prove that the MHE architecture provides better performance than the HOG [16], HBS [17], MBPA [18] and DMH [20]. The MHE architecture requires only one DPRROM to store the current and previous operation values.

The CSLA adder used in the MHE provides faster arithmetic operations during histogram estimation that leads to the minimisation of delays. The hardware utilisation of the MHE architecture is less due to the usage of fewer logical elements.

**Table 6** Comparison of MHE architecture with HOG for Virtex- 6 device

<i>Virtex 6</i>		
<i>Parameters</i>	<i>HOG [16]</i>	<i>MHE</i>
BRAM	3906	8
4 input LUT	77623	3388
Flip flops	5874	2177
Frequency (MHz)	104.25	112.335

**Table 7** Comparison of MHE architecture with MBPA for Virtex- 6 device

<i>Virtex 6</i>		
<i>Parameters</i>	<i>MBPA [18]</i>	<i>MHE</i>
Slices	2920	1478
4 input LUT	2684	3388
Flip flops	2465	2177
Frequency (MHz)	116.24	112.335

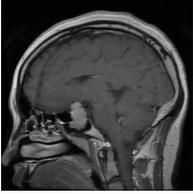
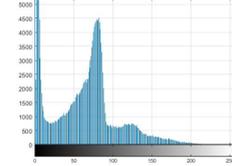
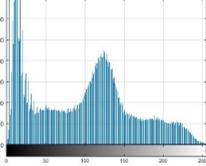
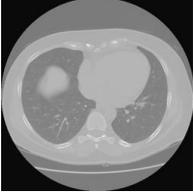
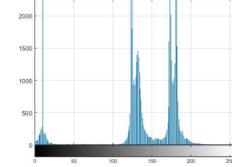
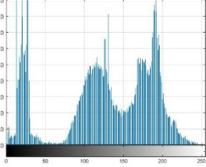
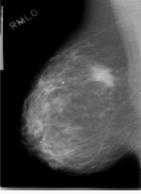
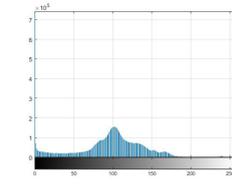
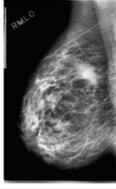
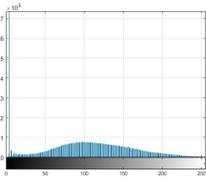
**Table 8** Comparison of MHE architecture for Virtex-7 device

<i>Virtex-7</i>		
<i>Parameters</i>	<i>HBS [17]</i>	<i>MHE</i>
No. of slice registers	38774	4116
No. of slice LUTs	50124	3451
No. of occupied slices	12850	3062
Maximum frequency (MHz)	445.330	134.21
Power (W)	1.215	1.041

**Table 9** Comparison of HE architecture for Zynq Z-7020 FPGA device

<i>Zynq Z-7020 FPGA</i>		
<i>Parameters</i>	<i>DMH [20]</i>	<i>MHE</i>
FF	11850	2345
LUT	9594	3456

**Table 10** Medical image denoising with histogram (see online version for colours)

<i>Input image</i>	<i>Input histogram</i>	<i>Enhanced image</i>	<i>Enhanced histogram</i>
			
			
			

## 6 Conclusion

In this paper, the MHE architecture is proposed to effectively estimate the histogram of the  $3 \times 3$  and  $16 \times 16$  size images. The sub modules used in the MHE are the DPROM, CSAC and OBC, which leads to improve the performance of the histogram estimation. The DPROM used in the MHE stores the current and previous sample operation. But the BHE architecture requires two different ROM to store the samples of current and previous operation. The utilisation of the DPROM, CSAC and OBC leads to minimise the amount of hardware used in the MHE architecture. The operating frequency of the 112.335 MHz is high when compared to the BHE architecture. The higher operating frequency of MHE architecture improved the speed of histogram estimation process. The area, power and delay are reduced by the MHE in 256 sample by 17.62%, 15.41% and 23.01%. Additionally, the MHE architecture provides better performance than the BHE, HOG, HBS, MBPA and DMH. The number of flip flops used in the MHE architecture is 2177 for Virtex 6 device, and it is less than the HOG and MBPA. Moreover, the image enhancement application was also implemented in this work to enhance the medical image quality. In the future, different optimal architecture will be designed to perform the histogram equalisation with less hardware utilisation.

## References

- 1 Yin, S., Ouyang, P., Chen, T., Liu, L. and Wei, S. (2015) 'A configurable parallel hardware architecture for efficient integral histogram image computing', *IEEE Trans. Very Large Scale Integr. VLSI Syst.*, Vol. 24, No. 4, pp.1305–1318.
- 2 Fahmy, S.A. and Mohan, A.R. (2012) 'Architecture for real-time nonparametric probability density function estimation', *IEEE Trans. Very Large Scale Integr. VLSI Syst.*, Vol. 21, No. 5, pp.910–920.
- 3 Bhai, G. and Agrawal, S. (2017) 'A novel low complexity histogram algorithm for high performance image processing applications', *Int. J. Eng. Technol.*, Vol. 04, No. 01, pp.1398–1403.
- 4 Kumar, S., Pant, M., Kumar, M. and Dutt, A. (2018) 'Colour image segmentation with histogram and homogeneity histogram difference using evolutionary algorithms', *Int. J. Mach. Learn. Cybern.*, Vol. 9, No. 1, pp.163–183.
- 5 Wei, Y.L. and Lin, C.H. (2018) 'Efficient weighted histogram features for single-shot person re-identification', *J. Signal Process. Syst.*, Vol. 90, No. 4, pp.477–491.
- 6 Roy, A., Manam, L. and Laskar, R.H. (2020) 'Removal of 'Salt and Pepper' noise from color images using adaptive fuzzy technique based on histogram estimation', *Multimed. Tools. Appl.*, pp.1–23.
- 7 Denisova, A.Y. and Sergeev, V.V. (2017) 'The algorithms of hierarchical histogram computation for multichannel images', *Proceedings of the 2017 International Conference on Computer Graphics and Digital Image Processing*, July, pp.1–6. [https://dl.acm.org/doi/abs/10.1145/3110224.3110238?casa\\_token=M-cxSrsbAEAAAAA:3jk98xdfCOZXO11UwYhb\\_LXC\\_\\_xLHCUAxpGTAxlmw8hk3dZq8cZ-pbh-AHNcemtKxbub8ly6Pw6Q](https://dl.acm.org/doi/abs/10.1145/3110224.3110238?casa_token=M-cxSrsbAEAAAAA:3jk98xdfCOZXO11UwYhb_LXC__xLHCUAxpGTAxlmw8hk3dZq8cZ-pbh-AHNcemtKxbub8ly6Pw6Q)
- 8 Xu, J., Zhang, Z., Xiao, X., Yang, Y., Yu, G. and Winslett, M. (2013) 'Differentially private histogram publication', *VLDB J.*, Vol. 22, No. 6, pp.797–822.
- 9 Vornicu, I., Darie, A., Carmona-Galán, R. and Rodríguez-Vázquez, Á. (2018) 'Compact real-time inter-frame histogram builder for 15-bits high-speed ToF-imagers based on single-photon detection', *IEEE Sens. J.*, Vol. 19, No. 6, pp.2181–2190.
- 10 Hazra, S., Ghosh, S., Maity, S.P. and Rahaman, H. (2016) 'A new FPGA and programmable SOC based VLSI architecture for histogram generation of grayscale images for image processing applications', *Procedia Comput. Sci.*, Vol. 93, pp.139–145.
- 11 Dutton, N., Vergote, J., Gneccchi, S., Grant, L., Lee, D., Pellegrini, S., Rae, B. and Henderson, R. (2014) 'Multiple-event direct to histogram TDC in 65nm FPGA technology', *2014 10th Conference on PhD Research in Microelectronics and Electronics (PRIME)*, June, IEEE, Grenoble, France, pp.1–5.
- 12 Kansal, S., Purwar, S. and Tripathi, R.K. (2018) 'Image contrast enhancement using unsharp masking and histogram equalization', *Multimed. Tools. Appl.*, Vol. 77, No. 20, pp.26919–26938.
- 13 Schatz, V. (2013) 'Low-latency histogram equalization for infrared image sequences: a hardware implementation', *J. Real-Time Image Process.*, Vol. 8, No. 2, pp.193–206.
- 14 Bonny, T., Rabie, T., Baziyad, M. and Balid, W. (2019) 'SHORT: segmented histogram technique for robust real-time object recognition', *Multimed. Tools. Appl.*, Vol. 78, No. 18, pp.25781–25806.
- 15 Vornicu, I., Carmona-Galán, R. and Rodríguez-Vázquez, Á. (2017) 'Real-time inter-frame histogram builder for SPAD image sensors', *IEEE Sens. J.*, Vol. 18, No. 4, pp.1576–1584.
- 16 Ghaffari, S., Soleimani, P., Li, K.F. and Capson, D.W. (2020) 'Analysis and comparison of FPGA-based histogram of oriented gradients implementations', *IEEE Access*, Vol. 8, pp.79920–79934.

- 17 Hazra, S., Ghosh, S., De, S. and Rahaman, H. (2018) 'FPGA implementation of semi-fragile reversible watermarking by histogram bin shifting in real time', *J. Real-Time Image Process.*, Vol. 14, No. 1, pp.193–221.
- 18 Mondal, P. and Banerjee, S. (2019) 'A reconfigurable memory-based fast VLSI architecture for computation of the histogram', *IEEE Trans. Consum. Electron.*, Vol. 65, No. 2, pp.128–133.
- 19 Yang, Y., Liu, Y.X. and Dong, Q.F. (2017) 'Sliced integral histogram: an efficient histogram computing algorithm and its FPGA implementation', *Multimed. Tools. Appl.*, Vol. 76, No. 12, pp.14327–14344.
- 20 Bonny, T., Rabie, T. and Hafez, A.A. (2018) 'Multiple histogram-based face recognition with high speed FPGA implementation', *Multimed. Tools. Appl.*, Vol. 77, No. 18, pp.24269–24288.
- 21 Nareshkumar, S. and Bikshalu, K. (2019) 'Adaptive absolute SCORE algorithm for spectrum sensing in cognitive radio', *Microprocess Microsyst.*, Vol. 69, pp.43–53.
- 22 Kishor, A., Chakraborty, C. and Jeberson, W. (2021) 'Reinforcement learning for medical information processing over heterogeneous networks', *Multimed. Tools. Appl.*, pp.1–22.
- 23 Kishor, A. and Chakraborty, C. (2021) 'Artificial intelligence and internet of things based healthcare 4.0 monitoring system', *Wirel. Pers. Commun.*, pp.1–17.
- 24 Kishor, A., Chakraborty, C. and Jeberson, W. (2020) 'A novel fog computing approach for minimization of latency in healthcare using machine learning', *Int. J. Interact. Multimed. Artif. Intell.*, Vol. 1, No. 1, pp.7–17.
- 25 Kishor, A., Chakraborty, C. and Jeberson, W. (2021) 'Intelligent healthcare data segregation using fog computing with internet of things and machine learning', *Int. J. Eng. Syst. Model. Simul.*, Vol. 12, Nos. 2–3, pp.188–194.