applied sciences

MDPI

Article

Quality Assessment of Dual-Parallel Edge Deblocking Filter Architecture for HEVC/H.265

Prayline Rajabai Christopher * and Sivanantham Sathasivam *

School of Electronics Engineering, Vellore Institute of Technology, Vellore 632014, Tamil Nadu, India
* Correspondence: prayline.c@vit.ac.in (P.R.C.); ssivanantham@vit.ac.in (S.S.)

Abstract: Preserving the visual quality is a major constraint for any algorithm in image and video processing applications. AVC and HEVC are the extensively used video coding standards for various video processing applications in recent days. These coding standards use filters to preserve the visual quality of the processed video. To retain the quality of the reconstructed video, AVC uses an in-loop filter, called the deblocking filter, while HEVC uses two in-loop filters, the sampling adaptive offset filter and the deblocking filter. These filters are implemented in hardware by adopting various optimization techniques such as reduction of power utilization, reduction of algorithm complexity, and consuming lesser area. The quality of the reconstructed video should not be impacted by these optimization measures. For the HEVC/H.265 coding standard, a parallel edge deblocking filter architecture is designed, and the effectiveness of the parallel edge filter architecture is evaluated using various quantization values for various resolutions. The quality of the parallel edge filter architecture is on par with the HEVC reference model.

Keywords: deblocking filter; quality assessment; parallel edge filter; HEVC/H.265; sample adaptive offset filter

Citation: Christopher, P.R.; Sathasivam, S. Quality Assessment of Dual-Parallel Edge Deblocking Filter Architecture for HEVC/H.265. *Appl. Sci.* **2022**, *12*, 12952. https://doi.org/10.3390/app122412952

Academic Editors: Zbigniew Lubniewski, Tadeus Uhl and Przemysław Falkowski-Gilski

Received: 14 October 2022
Accepted: 12 December 2022
Published: 16 December 2022

1. Introduction

Video compression is ineluctable in recent days owing to the rapid advancements in digital electronics. Several video coding techniques are widely used to compress video data, save storage space, and reduce channel bandwidth during transmission. HEVC is the last released video coding standard developed by the Joint Collaborative Team on Video Coding (JCT-VC) which has been used for more than half a decade in many multimedia applications. This video coding standard splits the raw input video frames into rectangular blocks which are transformed to the frequency domain and then predicted based on the former decoded video either by motion compensated prediction, interprediction, or intraprediction [1]. Due to this block-based transform coding followed by coarse quantization, there may not be uniformity in the intensity of the pixel transition within the two adjacent blocks. This nonuniformity in the transition of the intensity of the pixel values generates a visible discontinuity in the reconstructed video and, thus, degrades the quality of the decoded video.

Nonsmooth block boundaries commonly observed in earlier video coding standards performed at low and medium bit rates, such as visible block boundaries, color biases, and blurring effects [2], still exist in H.264 and H.265. Nonsmooth block boundaries are known as the blocking effect which is one of the most perceivable and objectionable artifacts of block-based compression methods [3,4]. Figure 1 shows the existence of blocking artifacts at the block boundary. It shows the variation of the pixel intensities between the edges of two 4 × 4 blocks. DBFs are used to reduce these blocking artifacts. Figure 2 shows the elimination of the blocking artifacts by smoothening the intensities of the pixels at the edges of a block. The computational complexity of the DBF algorithm is one-third of the H.264 video decoder [3] and one-fifth of the H.265 video decoder [5].

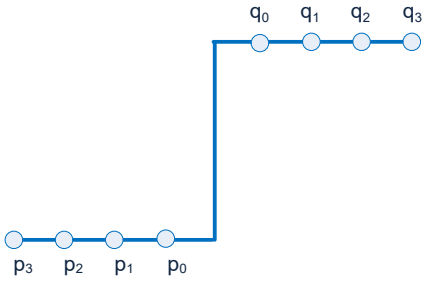

Figure 1. Variation of pixel intensities between block boundaries.

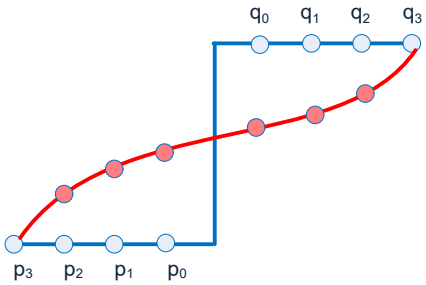

Figure 2. Smoothing of pixel intensities between block boundaries.

The discontinuities are perceivable for the human visual system (HVS) as blocking effects in the region with lower activity in the video frame [6]. Preserving the same visual quality as the original visual scene is a significant challenge. The AVC and HEVC coding standards utilize lossy compression algorithms such as DCT/DWT, and, hence, the originality of the pixel intensities are lost when the video data are reconstructed. However, measures are taken to upgrade the quality of the reconstructed video in the codec. AVC has an in-loop filter called the deblocking filter (DBF) to reduce visible discontinuities. HEVC standard reduces this visible discontinuity in the main profile by the use of two in-loop filters applied to the reconstructed video in succession. These two in-loop filters are the DBF and the sample adaptive offset (SAO) filter. Although these filters are employed to improve the reconstructed video's quality, these enhancement methods do not produce a perfect replica of the original image.

The DBF and SAO filters used in HEVC are optimized by modifying the filtering algorithms or by implementing the algorithms efficiently in hardware concerning cost and energy saving. Even though these optimization techniques enhance the quality of the video, we need to measure the quality to compare with different optimization techniques.

2. Related Work on Deblocking Filters

The deblocking filter and the sample adaptive offset (SAO) filter are the two filters used within the codec in the HEVC coding standard. To enhance the visual quality of the reconstructed frames, these filters are applied in two phases, the first stage applying the deblocking filter and the second stage applying the SAO filter. Blocking artifacts are eliminated by the deblocking filter, and the SAO filter enhances the visual quality by adding the offset values to the first-stage filtered pixel samples [7]. Edge offset or band offset are two possible offset values. The two filters for the HEVC coding standard are implemented in hardware adopting several architectural optimization techniques.

The SAO filter and deblocking filter are combined and implemented in hardware, or these filters are implemented separately as deblocking filter [2,8–14] and SAO filter [15]. Parallel and pipeline-based architectures are utilized to implement the area and throughput optimized deblocking filter. Few deblocking filter architectures use novel ordering for the filtering process to improve the performance. Several filter architectures were implemented by [8,16–23] to realize the H.265 deblocking filter. The deblocking filter architecture of H.264 is more complex compared to the deblocking filter architecture of H.265/HEVC deblocking filter [24]. SAO and deblocking filters

were combined and are implemented by [17,23,25–27]. In [22,28], a graphics processing unit (GPU) was used to implement the in-loop filter using parallelism. A multicore coprocessor was used to implement the HEVC in-loop filtering in [19]. Convolutional neural network (CNN) was used by [29,30] to create an in-loop deblocking filter with coding unit categorization.

3. Quality Assessment Metrics

The HEVC standard is known to be advantageous for higher video resolutions such as HD and UHD videos with lower bit rates. The features of HEVC improve the compression ratio by 50% with an increase in the complexity by 150% compared to its former video coding standard AVC [31]. Research is ongoing to reduce the overall complexity of the coding standard without affecting the compression ratio. Overall complexity can be reduced if there is a reduction in the complexity of the various modules used in these coding standards, but reduction in computational complexity may deteriorate the quality of the encoded video stream.

Mean square error (*MSE*) and *PSNR* are the metrics used to evaluate the objective perceptional video quality [32,33]. Structural similarity (SSIM) is also employed to evaluate the video quality by [34,35], but none of the metrics correlate precisely with the perceptional quality of the HVS [36]. However, SSIM provides better results with respect to the perceptional quality of the HVS using the assessment of three components, viz., perceptional impact of changes in the luminance, contrast, and structure [37]. Despite the complexity of measuring SSIM, due to the assessment of three different components, it is more reliable compared to other measures [38]. Several block-edge impairment metrics were proposed and a generalized block-edge impairment metric (DBIM) was proposed by [39], which shows the difference in the perceptual quality. Equations (1)–(3) show the quality assessment of the reconstructed video using *PSNR*, *MSE*, and *SSIM*, respectively.

$$MSE(i,j) = \frac{(\sum_{i=1}^{M}\sum_{j=1}^{N}[f(i,j) - F(i,j)]^2)}{M.N} \tag{1}$$

$$PSNR = 20log_{10}\frac{255}{\sqrt{MSE}} \tag{2}$$

where $f(i,j)$ is the pixel value at ith row and jth column of the original video frame, $F(i,j)$ is the pixel value at ith row and jth column of the reconstructed/modified video frame, and M and N represent the width and the height of the video frame. The value of *PSNR* ranges from 30 dB to 40 dB as the quality of the modified video ranges from medium to high, respectively.

$$SSIM = \frac{(2\bar{x}\bar{y} + C_1)(2\sigma_{xy} + C_2)}{[\bar{x}^2 + \bar{y}^2 + C_1](\sigma_x^2 + \sigma_y^2 + C_2)} \tag{3}$$

where $\bar{x}$, $\bar{y}$ are the mean of x and y, respectively; σ_x ,σ_y, and σ_{xy} are the variance of x, the variance of y, and the covariance of x and y, respectively; and C_1 and C_2 are constants. *SSIM* values range between -1 to 1, and the quality is said to be the best when the value is 1 [38].

Among the various metrics for video quality assessment, *PSNR* is the most desired quality assessment metric owing to its lucidity. Though *PSNR* is used to check the quality of the video to a great extent, it does not provide the actual perceptual quality as perceived by the HVS.

4. Parallel Edge Deblocking Filter Architecture (PEDBF)

The dual-parallel edge DBF architecture [40], employing five pipeline stages (V-DPEDBF) used to assess the quality, is shown in Figure 3. The (i) control unit, (ii) boundary Strength (BS) calculation unit, and (iii) filter unit are the three main modules in this architecture. The various operations of the filter architecture are administered by the control unit. An enable

signal is used as a primary input from the external world to enable the control unit. The control unit oversees and coordinates a number of processes, including data fetch from the external storage, data fetch from the memory within PEDBF, data write to the PEDBF memory, data write to the external storage device, activation of the boundary strength calculator, and activation of the filter unit. The BS unit determines the values of the BS, which range from 0 to 2. The filter unit performs the filtering procedure in accordance with the computed BS value.

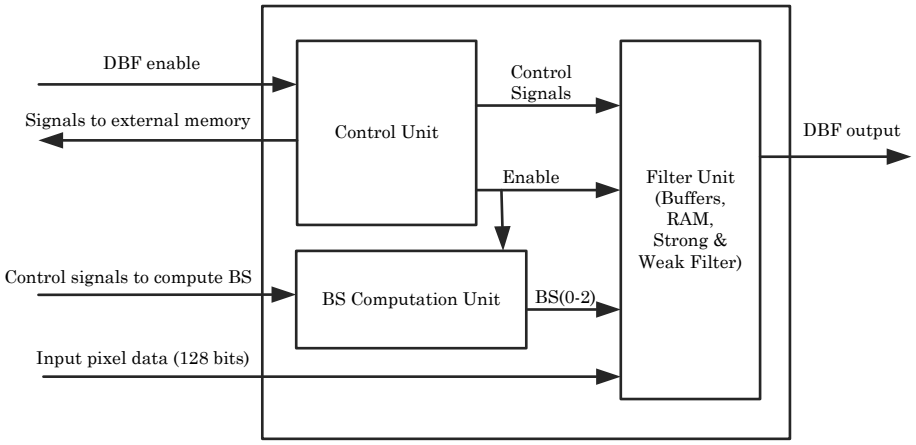

Figure 3. Architecture of V-DPEDBF for HEVC [40].

4.1. Control Unit

The control unit is activated by the control signal, which is one of the primary inputs to the filter architecture. The control unit of the HEVC DBF has a finite state machine to administer all the operations to handle the deblocking filtering process. When the control signal is not active, all the filtering process is deactivated and, thus, the modules are turned off. Hence, the DBF architecture consumes less power. When the enable signal is active, the control unit triggers the state machine and activates the filtering process in five stages. Five stages—memory read, parameter computation, filter determination, filtering, and memory write—are managed by the state machine during the filtering process. The pixel data are fetched from the external memory, which is outside the filter architecture, after the finite state machine is enabled. Then, 4 × 4 blocks (128 bits) of pixel data are fetched for each clock cycle from the external memory. Figure 4a depicts the sequencing of data fetch from the external memory for a largest coding unit (LCU), which is a 64 × 64 block, whereas Figure 4b,c depict the sequencing of data fetch from the external storage for a 16 × 16 block. The state machine activates the filter unit and the BS calculation unit by generating control signals once four 4 × 4 blocks of pixel data are in place.

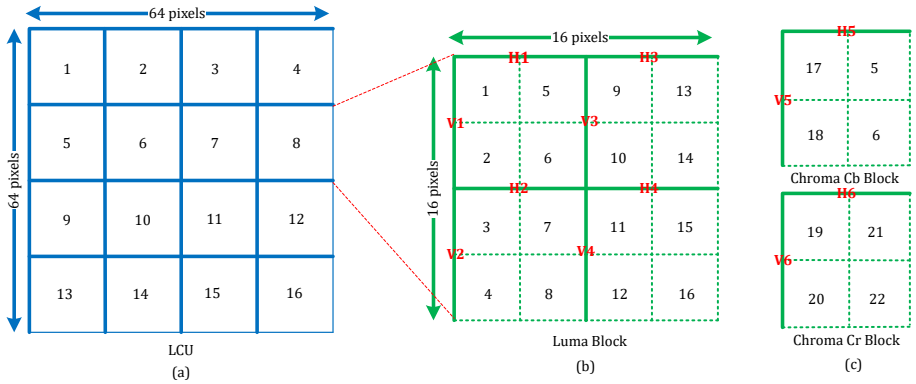

Figure 4. (**a**) Sequence of read for an LCU; (**b**,**c**) sequence of read for a 16 × 16 block.

The 16 × 16 luma block's vertical edges V1 and V2 are filtered in parallel in accordance with the determined BS value. Blocks 1 through 8 of the filtered pixel data are then transposed and stored in the internal memory. The filter unit is triggered once more to filter the edges V3 and V4 once the twelfth block of data is accessible. Blocks 9 through 16 of the vertically filtered data are eventually transposed and stored within the PEDBF architecture. After vertical filtering, the horizontal filter is applied by reading the pixel data that are stored within the PEDBF. The internal RAM receives the appropriate signals from the control unit to perform the required operation. H1 and H2 horizontal edges are parallel-filtered and then the filtered data are transposed. The frame memory, which is located external to the architecture, is subsequently written with the filtered data. The horizontal edges H3 and H4 are parallel filtered in a similar manner. After that, they are written to the external frame memory after being transposed. The Chroma Cb and Cr blocks filter the vertical edges (V5 and V6) first, followed by the horizontal edges (H5 and H6) using the same process. The filtered pixel data are subsequently saved in the external storage as 4 × 4 blocks, i.e., 128 bits for each clock cycle.

4.2. Boundary Strength Computation Unit

The boundary strength computation unit computes the BS value, as shown in Figure 5. The BS value range for the HEVC coding standard is between 0 and 2. The BS processing unit receives control signals regarding the pixel block received from the external buffer, such as whether the received pixel block is the left, right, top, or bottom edge of the frame, if it is inter/intracoded, and if its transform coefficients are not zero. The BS value is 0 if the block of pixel data read from the external buffer is a part of the left or top edge of a frame. In addition, the BS value is 0 if the data are not a part of the left or top edge of a frame, the two neighboring 8 × 8 blocks are not intracoded, if the block has nonzero transform coefficients, and if the motion vector is less than 4. The BS value is 1 if the transform coefficients of the block are nonzero and the adjacent blocks are not intracoded. The BS value is also 1 if the adjacent blocks are not intracoded and if the block does not have nonzero transform coefficients, and if the motion vector is higher than or equal to 4. The BS value is 2 if any of the above conditions are not met. DBF is triggered based on the calculated BS value. If the generated value of BS is 0, no filtering is performed; if the determined value is 1, a weak/normal filter is used; and if the stipulated value is 2, a strong filter is used.

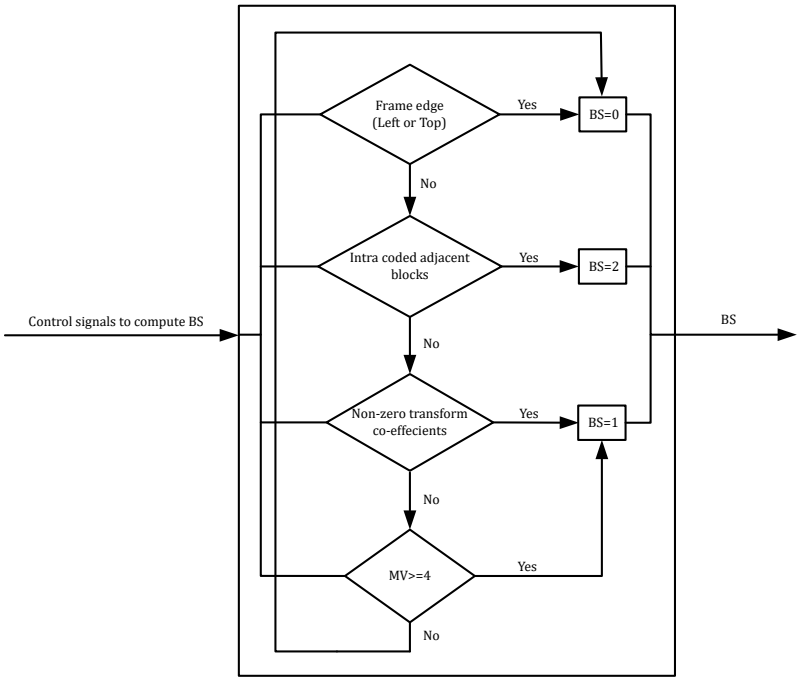

Figure 5. Boundary strength computation unit.

4.3. Filter Module

When the pixel data are ready to carry out the filtering operation, the control unit turns on the filter unit. This unit is the sophisticated computational unit. The architecture of the filter unit is shown in Figure 6. It has a parameter computation unit, buffers to store the pixel data, a filter decision block, internal memories, a strong filter, and a weak filter.

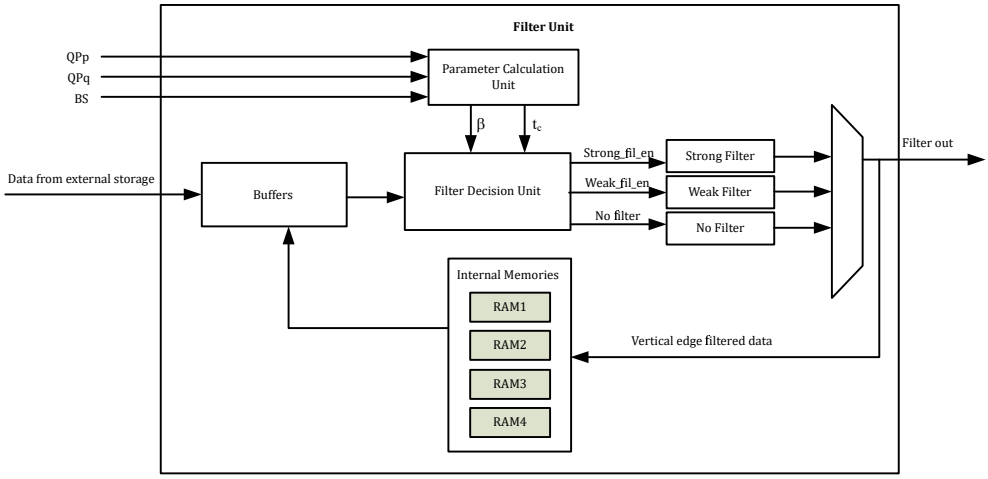

Figure 6. Filter unit of dual-edge deblocking filter architecture.

4.3.1. Parameter Calculation Unit

This unit calculates the filtering parameters such as β and t_c according to Tables 8–12 of [41]. Based on the quantization parameter values and the BS of the neighboring P block and Q block, referred to QP_p and QP_q, respectively, these filtering parameters are calculated. The LUT used to implement the parameter calculation unit has the outputs t_c and β. These output values are relative to the inputs BS, QP_p and QP_q.

4.3.2. Buffers

The filter unit of the dual-edge deblocking filter architecture uses eight buffers, as shown in Figure 7, each of which can store a 4×4 block of pixels (128 bits). These buffers are initialized with all zeroes. Before the control unit begins the filtering process, the block of pixels 1–4 indicated in Figure 4b from the external memory are stored into the corresponding buffers Q1 BUF, Q2 BUF, Q3 BUF, and Q4 BUF. The filtering operation is carried out along the vertical edges V1 and V2; at the same time, the block of pixels 5–8 is sent to the buffers P1 BUF, P2 BUF, P3 BUF, and P4 BUF, respectively. Figure 8 indicates the relative buffer for both the luma and chroma blocks, along with the outline of each 4×4 pixel block. The filtered data are saved to the internal RAM after the edges V1 and V2 have been filtered. Similar techniques are used to filter V3 and V4 vertical edges. For horizontal filtering, the pixel data kept in the internal memory were subsequently transferred to these storage units. The same process is utilized for horizontal filtering as for vertical filtering.

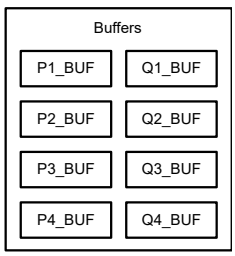

Figure 7. Buffers to store the pixel block.

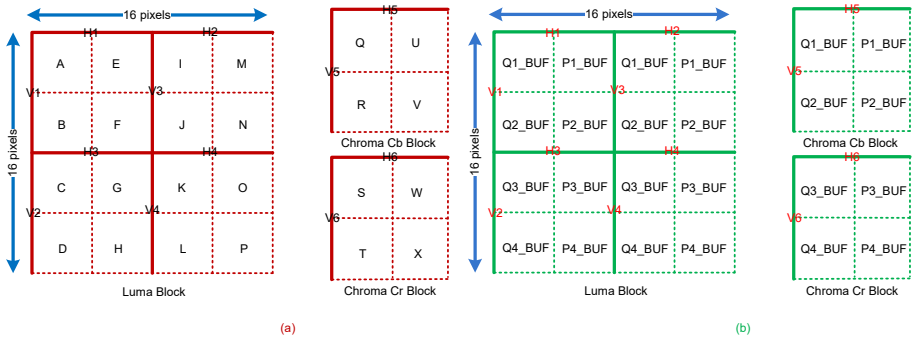

Figure 8. Mapping of pixel blocks to buffers: (**a**) pixel blocks; (**b**) buffers.

4.3.3. Filter Decision Unit

Based on the values of the parameters β and t_c, the strength of the filtering procedure to be used for a 4 × 4 pixel block is determined. The pixel threshold values for the two neighboring blocks are decided by the filter decision unit.

4.3.4. Internal Memories

The memories utilized within the filter architecture employ four dual-port RAM to store the pixel data filtered vertically. A 4 × 4 block of pixel data can be stored in each of the four segments of the 64-byte RAM (16 bytes or 128 bits). Figures 9 and 10 depict the transfer of the pixel data for the luma and the chroma Cb, Cr blocks from the RAM to the buffers. The amount of external memory access cycles is reduced by this method of data storage. It also avoids the utilization of transpose buffers. The first two memory regions are used exclusively during the chroma block filtering procedure, leaving the other regions unused.

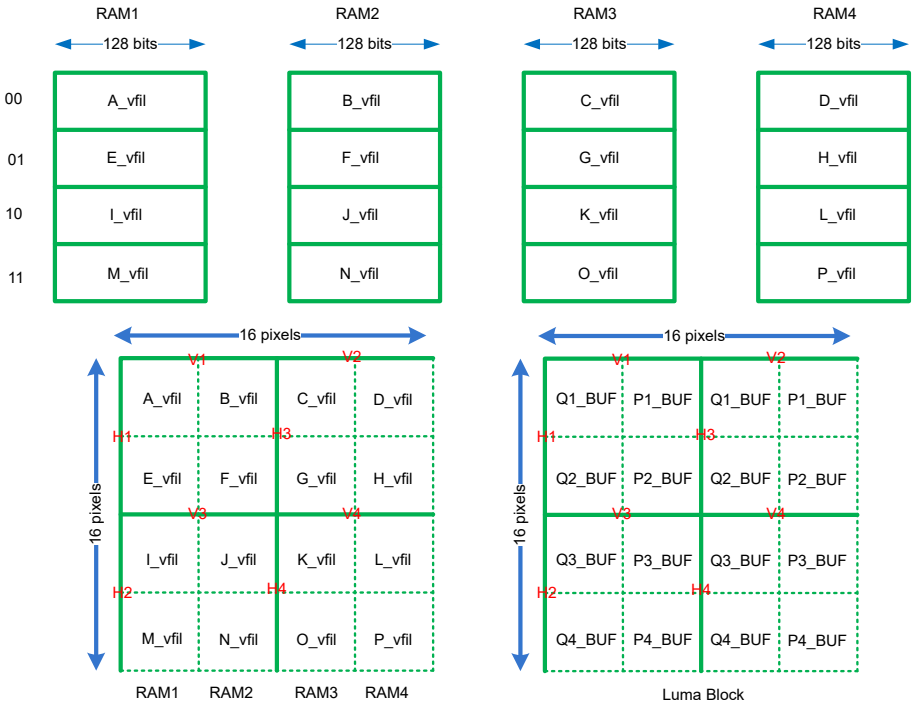

Figure 9. Mapping of internal memory to buffer before horizontal filtering for luma block.

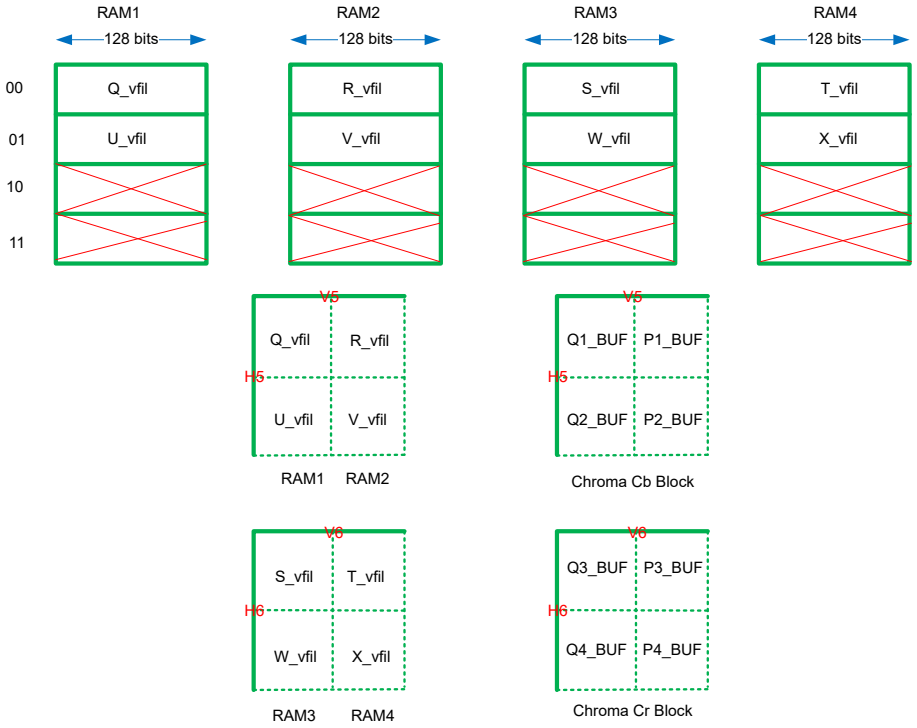

Figure 10. Mapping of internal memory to buffer before horizontal filtering for chroma block.

4.3.5. Filter Modules

The architecture includes filter modules that make use of both strong and weak/normal filters. Any of these filters are activated according to the filtering decisions determined by the filter decision unit. Therefore, any one of the weak filter or the strong filter is activated to execute the filtering process. The filtering process is omitted if the filtering decision unit determines for no filtering. The weak or strong filtering is executed according to the filtering equations specified in [41]. The similarities in these equations allow for the creation and implementation of a resource-sharing architecture for the filter module, which optimizes the area. The data from the horizontally filtered pixel block are then saved in the external memory, while the data from the vertically filtered pixel block are stored in the internal memory.

4.4. Resource Sharing Architecture

Depending on the deblocking filtering technique used in [9], which follows the HEVC standard, the weak and strong filter units are implemented. The filter architecture is constructed in such a way that it shares the common resources that are used in these equations by taking advantage of the similarity in the filtering equations. This technique thenceforward brings down the area and the power utilization. The pixels of the two neighboring blocks are updated for the strong filter using the third parameter of the clip3 function, which is an equation involving two or more of the pixel values of the adjacent P and Q blocks. Adders and shifters are utilized to implement these equations in the hardware. It is identified that $p_0 + q_0$ is employed in all the expressions and $p_0 + q_0 + 2$ is used in most of the expressions. In addition, $p_1 + 2$ and $q_1 + 2$ are used twice. Hence, to add $p_0 + q_0$, an eight-bit binary adder is employed and the output is fanned out to all the equations. This adder's output is also used as an input for another binary adder, whose other input is 2, producing the result $p_0 + q_0 + 2$, which is then utilized in the appropriate expressions. The same mechanism is used to implement the addition operations $p_1 + 2$ and $q_1 + 2$, and the output of these adders is reused wherever necessary. As a result, the shared resources are distributed over several equations, and an area-optimized filter is used. Figure 11 depicts the resource sharing architecture used for the third argument of the clip3 function for the strong filter.

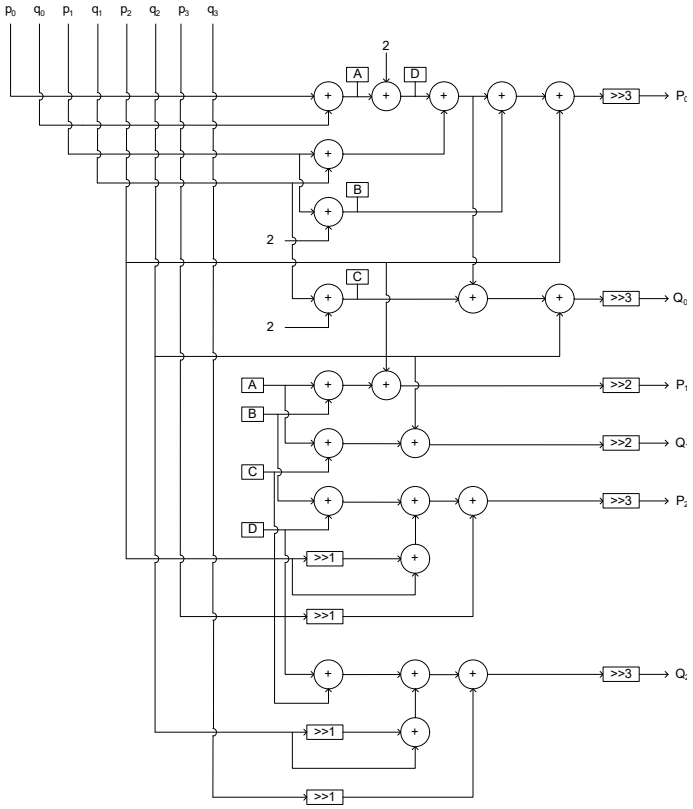

Figure 11. Resource sharing architecture of strong filter—partial view.

5. Quality of PEDBF for HEVC/H.265

The optimization techniques employed in the hardware architectures of the DBF filter should not deteriorate the quality of the reconstructed video for any reason. Degradation in the quality of the filtered video will lead to poor performance and coding inefficiency. The DBF algorithm is highly adaptive and data-dependent, as the current edge to be filtered depends on the previously filtered pixel blocks. Since different filter ordering is followed, these changes in the ordering should not affect the quality of the filtered video. Hence, the quality assessment of the dual-edge DBF for H.265 coding standard is measured using *PSNR*.

Quality Assessment Procedure

To measure the quality of the V-DPEDBF architecture, the following steps are executed sequentially.

1. Original raw video is given as input to the HEVC Test Model to obtain the reconstructed video by disabling the in-loop filter
2. Reconstructed video data from the HM is given as input to the dual-parallel edge deblocking filter (PEDBF) architecture.
3. The input video to PEDBF architecture is split into luminance (Y), chrominance Cb (U) and, chrominance Cr (V) components.
4. Each video component is segregated into image frames and stored into frame buffers.
5. Each image frame of the Y, U, and V components is fetched from the frame buffers.
6. Each image frame is split into uniform pixel blocks of size 16×16.
7. Each 16×16 block is again split into four 8×8 blocks.
8. DBF is applied at the boundary of two 8×8 blocks simultaneously for vertical filtering.
9. The vertically filtered output data are transposed and moved into pixel buffers to perform horizontal edge filtering.

10. Horizontal edge filtering is performed after the vertical edge filtering is performed for the entire 16 × 16 block.
11. The horizontally filtered output is transposed and stored into the frame buffer.
12. The filtered video of PEDBF is reconstructed from the frame buffer.
13. The *PSNR* of the filtered video is computed.
14. Original raw video is given as input to the HEVC Test Model to obtain the reconstructed video by enabling the in-loop filter.
15. The *PSNR* of the filtered video from the HEVC reference software is computed and the results are compared.

The flow diagram to perform the quality assessment is shown in Figure 12.

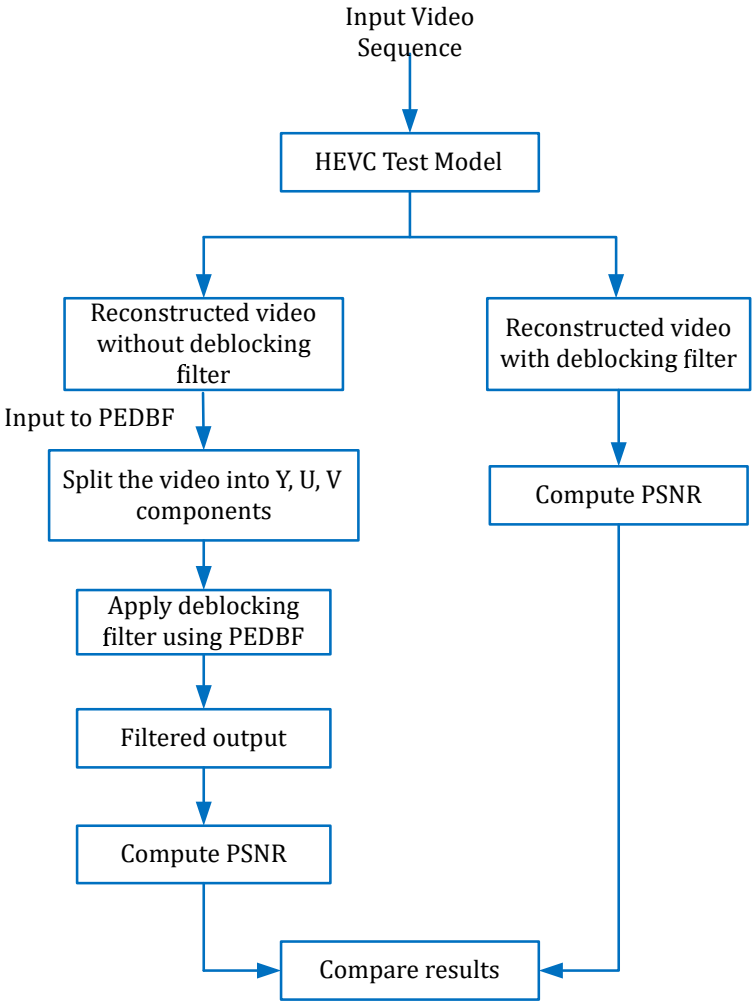

Figure 12. Flow diagram to perform quality assessment.

6. Filter Selection

The filter architecture uses four resource sharing edge filters that operate simultaneously to filter the edges of two 8 × 8 blocks of pixels. Each 8 × 8 block's filtering decision is made in accordance with the β, t_c parameters and in accordance with the threshold values of the two adjacent pixel blocks. The following criteria are employed for selecting the type of filter:

1. If the BS value is zero, then no filtering is performed.
2. If the BS value is nonzero, then Equations (4)–(8) are computed; and if Equation (8) is not satisfied, then no filtering is performed.

$$dp0 = abs(p_{2,0} - 2p_{1,0} + p_{0,0}) \quad (4)$$

$$dp3 = abs(p_{2,3} - 2p_{1,3} + p_{0,3}) \tag{5}$$

$$dq0 = abs(q_{2,0} - 2q_{1,0} + q_{0,0}) \tag{6}$$

$$dq3 = abs(q_{2,3} - 2q_{1,3} + q_{0,3}) \tag{7}$$

$$dp0 + dp3 + dq0 + dq3 < \beta \tag{8}$$

3. If Equation (8) is satisfied then Equations (9)–(16) are computed to decide a strong or weak filter. If Equations (11)–(16) are satisfied, then a strong filter is used.

$$dpq0 = dp0 + dq0 \tag{9}$$

$$dpq3 = dp3 + dq3 \tag{10}$$

$$dpq0 < \beta/8 \tag{11}$$

$$dpq3 < \beta/8 \tag{12}$$

$$abs(p_{3,0} - p_{0,0}) + abs(q_{0,0} - q_{3,0}) < \beta/8 \tag{13}$$

$$abs(p_{3,3} - p_{0,3}) + abs(q_{0,3} - q_{3,3}) < \beta/8 \tag{14}$$

$$abs(p_{0,0} - q_{0,0}) < 2.5t_c \tag{15}$$

$$abs(p_{0,3} - q_{0,3}) < 2.5t_c \tag{16}$$

4. If any one of Equations (11)–(16) are not satisfied, then Equations (17)–(20) are computed and if Equation (20) is satisfied, a weak filter is used.

$$dp0 + dp3 < \beta/16 \tag{17}$$

$$dq0 + dq3 < \beta/16 \tag{18}$$

$$\delta_0 = 9(q_{0,0} - p_{0,0}) - 3(q_{0,1} - p_{0,1}) \gg 4 \tag{19}$$

$$abs(\delta_0) < 10t_c \tag{20}$$

5. If Equation (20) is not satisfied then the filtering process is skipped (no filter).

7. Filtering Operation

The deblocking filtering operations are performed on the edges of the luma and the chroma Cb and Cr pixel blocks based on the computed boundary strength value. Boundary strength is computed based on the coding information given by the codec.

7.1. Luma Block

A strong filter is used, and up to three pixels on either side of the block edges are adjusted as in Equation (21) to Equation (26) for a luma block if the computed BS value is 2.

$$p_0 = Clip3(p_0 - 2t_c,\ p_0 + 2t_c\ , \\ (p_2 + 2p_1 + 2p_0 + 2q_0 + q_1 + 4) \gg 3) \tag{21}$$

$$p_1 = Clip3(p_1 - 2t_c,\ p_1 + 2t_c\ , \\ (p_2 + p_1 + p_0 + q_0 + 2) \gg 2) \tag{22}$$

$$p_2 = Clip3(p_2 - 2t_c,\ p_2 + 2t_c\ , \\ (2p_3 + 3p_2 + p_1 + p_0 + q_0 + 4) \gg 3) \tag{23}$$

$$q_0 = Clip3(q_0 - 2t_c,\ q_0 + 2t_c\ , \\ (p_1 + 2p_0 + 2q_0 + 2q_1 + q_2 + 4) \gg 3) \tag{24}$$

$$q_1 = Clip3(q_1 - 2t_c,\ q_1 + 2t_c\ , \\ (q_2 + q_1 + q_0 + p_0 + 2) \gg 2) \tag{25}$$

$$q_2 = Clip3(q_2 - 2t_c,\ q_2 + 2t_c\ , \\ (2q_3 + 3q_2 + q_1 + q_0 + p_0 + 4) \gg 3) \tag{26}$$

If BS = 1, then a weak filter is applied and the pixels on either side of the block edges are modified as in Equations (27)–(32).

$$\Delta = 9(q_0 - p_0) - 3(q_1 - p_1) + 8 \gg 4 \tag{27}$$

If Abs(Δ)<10t_c then

$$\Delta = Clip3(-t_c,\ t_c, \Delta) \tag{28}$$

$$p_0 = Clip1_Y(p_0 + \Delta) \tag{29}$$

$$q_0 = Clip1_Y(q_0 - \Delta) \tag{30}$$

when dEp = 1, then

$$p_1 = Clip1_Y(p_1 + \Delta_p) \tag{31}$$

where $\Delta_p = Clip3(-\frac{t_c}{2}, \frac{t_c}{2}, (((p_2 + p_0 + 1) \gg 1) - p_1 + \Delta) \gg 1)$.

When dEq = 1, then

$$q_1 = Clip1_Y(q_1 + \Delta_q) \tag{32}$$

where $\Delta_q = Clip3(-\frac{t_c}{2}, \frac{t_c}{2}, (((q_2 + q_0 + 1) \gg 1) - q_1 + \Delta) \gg 1)$

7.2. Chroma Block—Cb and Cr

Chroma blocks are filtered only if BS = 2, and no filtering is performed when BS = 1 or 0. When BS = 2, the strong filter is applied, and the pixels p_0 and q_0 alone on either sides of the block edges are modified as in Equations (33)–(35).

$$\Delta_c = Clip3(-t_c, t_c, (((p_0 - q_0) \ll 2) + p_1 - q_1 + 4) \gg 3) \tag{33}$$

$$p_0 = ClipY(p_0 + \Delta_c) \tag{34}$$

$$q_0 = ClipY(q_0 - \Delta_c) \tag{35}$$

8. Results and Discussion

The quality assessment of the five-stage pipelined dual-edge deblocking filter architecture implemented for the HEVC standard [40] is performed using Matlab. Different test video sequences are used to obtain the *PSNR* values of the PEDBF architecture using different QP values. The typical QP value is 32. Hence, the quality assessment is performed with the typical QP value (32), the QP value lesser than the typical value (27), and QP value greater than the typical QP (37). Figure 13 shows an image frame of the original video sequence. This original/raw video sequence is given as input to the HM software to obtain two different types of encoded video. One type of encoded video is obtained from the HM software with the deblocking filter turned off (without DBF), and another type of encoded video is obtained from the HM software with the deblocking filter turned on (with DBF). Figures 14 and 15 show a sample frame of the output video sequence from the HM software without DBF and with DBF, respectively. The encoded output video from the HM software without DBF is used to perform the quality analysis of the PEDBF architecture. This video encoded without DBF is given as the input video sequence to the PEDBF implemented in Matlab to undergo deblocking filtering. Figure 16 shows a sample frame of the output video obtained from the PEDBF. The *PSNR* value of the output video obtained from the PEBDF is computed.

Figure 13. Frame of the original input video.

Figure 14. Frame of output video from HM with DBF disabled.

Figure 15. Frame of output video from HM with DBF enabled.

Figure 16. Frame of output video from dual PEDBF.

The quality of this architecture is assessed by comparing the *PSNR* value of the output video of the PEDBF architecture and the *PSNR* value of the video obtained using the HM software with DBF turned on for different QP values. The results for QCIF video sequences are tabulated in Table 1 and the comparison graph is shown in Figure 17 with QP = 32. The results for CIF video sequences are tabulated in Table 2 and the comparison graph is shown in Figure 18 with QP = 32. It is noted that the implemented architecture shows a slight improvement in quality concerning the *PSNR* metrics. The execution time for QCIF and CIF video sequences are tabulated in Tables 3 and 4, respectively, and the comparison graphs are shown in Figures 19 and 20. It is seen that, as the two edges are filtered in

parallel, the filtering time is reduced by 50%, and thus the throughput of the architecture is improved by 50% with the increase in quality.

Table 1. *PSNR* of QCIF video sequences with QP = 32.

Video Sequence	HM (dB)	PEDBF (dB)
Akiyo	37.3833	37.59186
Coastguard	34.4138	36.40767
Foreman	35.4274	36.58643
Mobile Calendar	32.2366	32.85071
Hall	36.4334	37.235
Carphone	36.1972	36.4323
Miss America	38.8972	39.12775

Table 2. *PSNR* of CIF video sequences with QP = 32.

Video Sequence	HM (dB)	PEDBF (dB)
Akiyo	38.307737	39.2812
Coastguard	35.03723	34.9852
Foreman	35.8951	35.99
Mobile Calendar	33.04411	33.0432
Hall	36.93208	37.4513
Tempete	34.09227	34.224

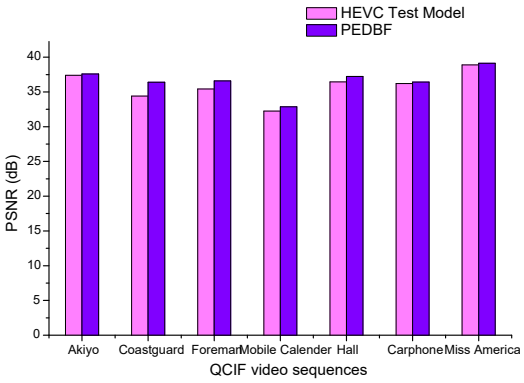

Figure 17. Comparison of *PSNR* with HM for QCIF video sequences using QP = 32.

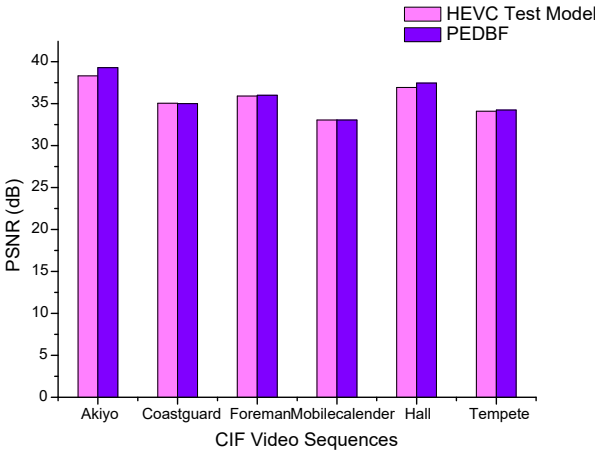

Figure 18. Comparison of *PSNR* with HM for CIF video sequences using QP = 32.

Table 3. Processing time of QCIF video sequences with QP = 32.

Video Sequence	HM (Sec)	PEDBF (Sec)
Akiyo	64.678	39.4553
Coastguard	85.103	44.81924
Foreman	82.368	45.6632
Mobile Calendar	85.673	43.21693
Hall	64.064	37.28175
Carphone	73.358	43.57872
Miss America	23.811	13.27997

Table 4. Processing time of CIF video sequences with QP = 32.

Video Sequence	HM (Sec)	PEDBF (Sec)
Akiyo	264.989	165.49741
Coastguard	376.373	196.2231
Foreman	278.388	176.485
Mobile Calendar	312.898	193.3375
Hall	219.416	112.2503
Tempete	229.776	152.46557

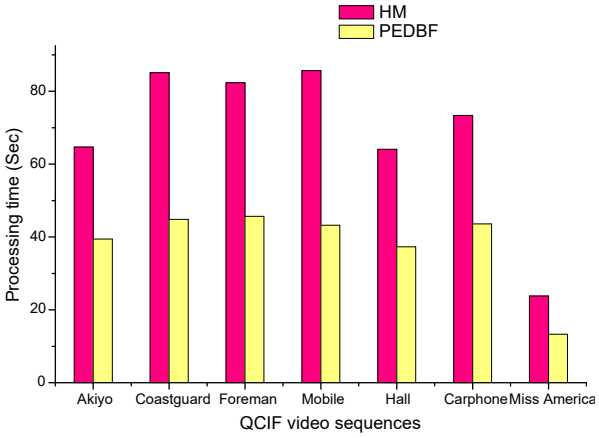

Figure 19. Comparison of processing time with HM for QCIF video sequences using QP = 32.

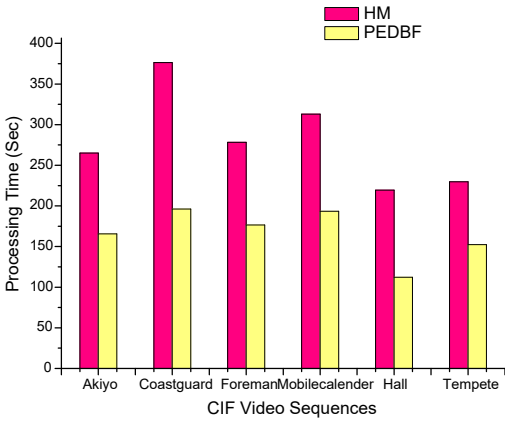

Figure 20. Comparison of processing time with HM for CIF video sequences using QP = 32.

Tables 5 and 6 show the results of QCIF and CIF video sequences, respectively, filtered with QP = 37, and the comparison graphs are shown in Figures 21 and 22. The execution

time for QCIF and CIF video sequences are tabulated in Tables 7 and 8, and the comparison graphs are shown in Figures 23 and 24.

Table 5. *PSNR* of QCIF video sequences with QP = 37.

Video Sequence	HM (dB)	PEDBF (dB)
Akiyo	34.0182	34.893362
Coastguard	31.1392	33.457211
Foreman	32.1997	33.862149
Mobile Calendar	28.3189	29.568311
Hall	32.971	34.430916
Carphone	32.9865	34.303814
Miss America	36.4185	37.267113

Table 6. *PSNR* of CIF video sequences with QP = 37.

Video Sequence	HM (dB)	PEDBF (dB)
Akiyo	36.2985	36.716387
Coastguard	31.8043	33.970701
Foreman	33.0051	34.390624
Mobile Calendar	29.2222	30.454212
Hall	34.5437	35.553274
Tempete	30.6828	32.131876

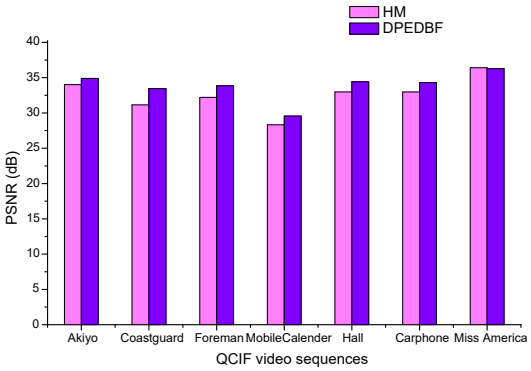

Figure 21. Comparison of *PSNR* with HM for QCIF video sequences using QP = 37.

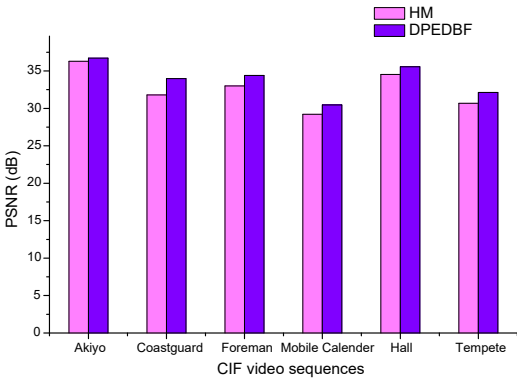

Figure 22. Comparison of *PSNR* with HM for CIF video sequences using QP = 37.

Table 7. Processing time of QCIF video sequences with QP = 37.

Video Sequence	HM (Sec)	PEDBF (Sec)
Akiyo	46.354	23.904816
Coastguard	52.02	27.976072
Foreman	48.861	28.335635
Mobile Calendar	70.905	38.544973
Hall	50.138	28.943229
Carphone	63.938	33.216266
Miss America	23.778	15.006227

Table 8. Processing time of CIF video sequences with QP = 37.

Video Sequence	HM (Sec)	PEDBF (Sec)
Akiyo	361.664	139.577545
Coastguard	323.229	141.569951
Foreman	293.987	161.030763
Mobile Calendar	461.949	225.603688
Hall	280.68	142.62714
Tempete	200.476	112.878238

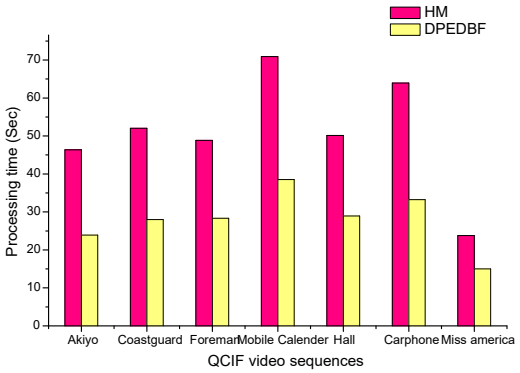

Figure 23. Comparison of processing time with HM for QCIF video sequences using QP = 37.

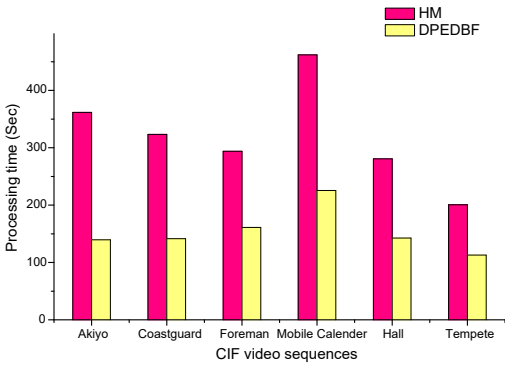

Figure 24. Comparison of processing time with HM for CIF video sequences using QP = 37.

Tables 9 and 10 show the results of QCIF and CIF video sequences, respectively, filtered with QP = 27, and the comparison graphs are shown in Figures 25 and 26. The execution time for QCIF and CIF video sequences are tabulated in Tables 11 and 12, and the comparison graphs are shown in Figures 27 and 28.

Table 9. *PSNR* of QCIF video sequences with QP = 27.

Video Sequence	HM (dB)	PEDBF (dB)
Akiyo	40.8999	40.891213
Coastguard	38.1673	39.927605
Foreman	38.9294	39.793245
Mobile Calendar	36.6292	36.744202
Hall	39.8733	40.164219
Carphone	39.6617	39.917189
Miss America	41.5719	41.766507

Table 10. *PSNR* of CIF video sequences with QP = 27.

Video Sequence	HM (dB)	PEDBF (dB)
Akiyo	42.2861	42.332459
Coastguard	38.6598	40.411503
Foreman	39.2529	40.130872
Mobile Calendar	37.1378	37.29197
Hall	40.1375	40.525791
Tempete	38.0572	38.495716

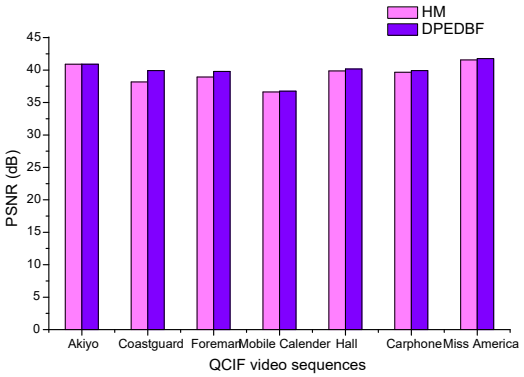

Figure 25. Comparison of *PSNR* with HM for QCIF video sequences using QP = 27.

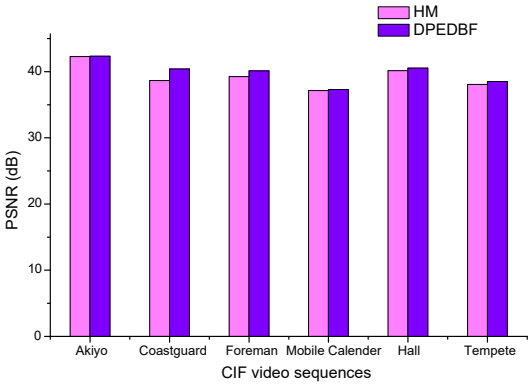

Figure 26. Comparison of *PSNR* with HM for CIF video sequences using QP = 27.

Table 11. Processing time of QCIF video sequences with QP = 27.

Video Sequence	HM (Sec)	PEDBF (Sec)
Akiyo	66.61	32.722607
Coastguard	72.627	39.74846
Foreman	81.681	41.068033
Mobile Calendar	120.716	58.555709
Hall	67.687	35.016408
Carphone	95.34	55.400211
Miss America	24.741	13.174376

Table 12. Processing time of CIF video sequences with QP = 27.

Video Sequence	HM (Sec)	PEDBF (Sec)
Akiyo	205.041	100.410937
Coastguard	277.788	150.710927
Foreman	250.539	126.669947
Mobile Calendar	352.151	208.336164
Hall	228.818	117.526938
Tempete	263.123	156.781587

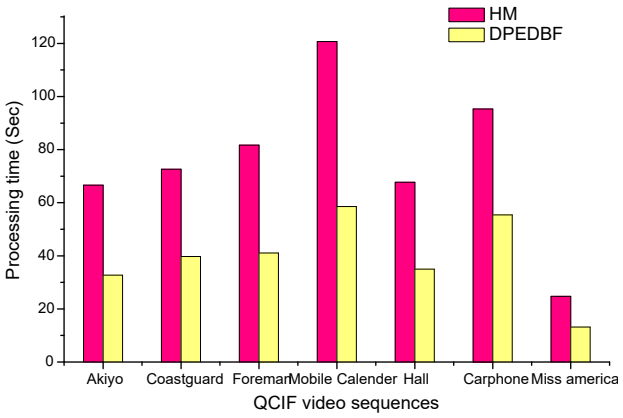

Figure 27. Comparison of processing time with HM for QCIF video sequences using QP = 27.

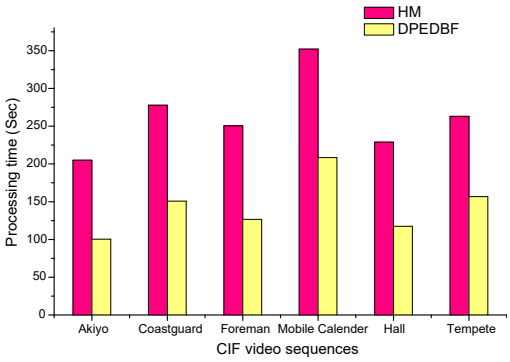

Figure 28. Comparison of processing time with HM for CIF video sequences using QP = 27.

Based on the experiments carried out with different video sequences of different resolutions for different QP values, it is identified that as the QP value increases, the quality decreases and results in lower *PSNR* values. In addition, the execution time decreases as

the QP value increases. This behavior is as expected with any video coding algorithm. Hence, the PEDBF architecture complies with the coding standards. The execution time of the PEDBF architecture is reduced to almost half of the execution time of the HM, owing to the fact of filtering two edges in parallel.

9. Conclusions

The quality assessment of any algorithm implemented in hardware is strongly required as the optimization techniques do not affect the quality of the reconstructed video data in any of the processing steps. Among the various quality assessment metrics, we use the metric *PSNR* to assess the quality of the video owing to its simplicity. Raw video data of two different resolutions (QCIF and CIF) are taken and the quality of the filtered video is checked with the quality of the video obtained from the HEVC Test Model. It is noted that the quality of the parallel edge filter architecture does not affect the quality of the reconstructed video, and it shows slight improvements compared to the HEVC Test Model. In one second, ten to twelve separate images can be processed by the human visual system by perceiving each image discretely. One image is held in the visual cortex for around one of fifteen parts in a second. Therefore, as the frame rate is higher, perception of the moving picture will be smooth. Hence, the processing time decreases in the dual-parallel edge DBF, the frame rate will increase, and, thus, the perceptual quality is increased.

Author Contributions: Conceptualization, P.R.C. and S.S.; methodology, P.R.C. and S.S.; validation, P.R.C.; formal analysis, P.R.C.; investigation, P.R.C. and S.S.; resources, P.R.C.; data curation, P.R.C.; writing—original draft preparation, P.R.C.; writing—review and editing, P.R.C. and S.S.; supervision, S.S. All authors have read and agreed to the published version of the manuscript.

Funding: The APC is funded by Vellore Institute of India, Vellore, Tamil Nadu, India.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The reference software for HEVC called HM (HEVC Test Model) is from HM software repository https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware (accessed on 1 October 2022), and the test video sequences used for quality assessment are from the Video Trace Library of Arizona State University http://trace.eas.asu.edu/yuv/index.html (accessed on 1 October 2022) and Video Information Processing Lab of National Chiao Tung University http://vip.cs.nctu.edu.tw/resource_seq.html (accessed on 1 October 2022).

Acknowledgments: The authors would like to acknowledge Vellore Institute of Technology, Vellore, Tamil Nadu, India, for providing all the necessary facilities for the research.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Wedi, T.; Musmann, H.G. Motion-and aliasing-compensated prediction for hybrid video coding. *IEEE Trans. Circuits Syst. Video Technol.* **2003**, *13*, 577–586. [CrossRef]
2. Zhou, W.; Zhang, J.; Zhou, X.; Liu, Z.; Liu, X. A high-throughput and multi-parallel VLSI architecture for HEVC deblocking filter. *IEEE Trans. Multimed.* **2016**, *18*, 1034–1047. [CrossRef]
3. Pourazad, M.T.; Doutre, C.; Azimi, M.; Nasiopoulos, P. HEVC: The new gold standard for video compression: How does HEVC compare with H. 264/AVC? *IEEE Consum. Electron. Mag.* **2012**, *1*, 36–46. [CrossRef]
4. Singh, J.; Singh, S.; Singh, D.; Uddin, M. Blocking artefact detection in block-based DCT compressed images. *Int. J. Signal Imaging Syst. Eng.* **2011**, *4*, 181–188. [CrossRef]
5. Vanne, J.; Viitanen, M.; Hamalainen, T.D.; Hallapuro, A. Comparative rate-distortion-complexity analysis of HEVC and AVC video codecs. *IEEE Trans. Circuits Syst. Video Technol.* **2012**, *22*, 1885–1898. [CrossRef]
6. Tai, S.C.; Chen, Y.Y.; Sheu, S.F. Deblocking filter for low bit rate MPEG-4 video. *IEEE Trans. Circuits Syst. Video Technol.* **2005**, *15*, 733–741.
7. Wang, Y.; Guo, X.; Lu, Y.; Fan, X.; Zhao, D. GPU-based optimization for sample adaptive offset in HEVC. In Proceedings of the 2016 IEEE International Conference on Image Processing (ICIP), Phoenix, AZ, USA, 25–28 September 2016; pp. 829–833.

8. Li, M.; Zhou, J.; Zhou, D.; Peng, X.; Goto, S. De-blocking Filter Design for HEVC and H. 264/AVC. In *Lecture Notes in Computer Science, Proceedings of the 13th Pacific-Rim Conference on Multimedia, Singapore, Singapore, 4–6 December 2012*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 273–284.
9. Hsu, P.K.; Shen, C.A. The VLSI Architecture of a Highly Efficient Deblocking Filter for HEVC Systems. *IEEE Trans. Circuits Syst. Video Technol.* **2017**, *27*, 1091–1103. [CrossRef]
10. Ye, X.; Ding, D.; Yu, L. A cost-efficient hardware architecture of deblocking filter in HEVC. In Proceedings of the Visual Communications and Image Processing Conference, Valletta, Malta, 7–10 December 2014; pp. 209–212.
11. Bae, J. Register array-based VLSI architecture of H. 265/HEVC loop filter. *IEICE Electron. Express* **2013**, *10*, 20130161. [CrossRef]
12. Tang, G.; Zeng, X.; Fan, Y. An SRAM-free HEVC Deblocking Filter VLSI Architecture for 8K Application. In Proceedings of the 2018 14th IEEE International Conference on Solid-State and Integrated Circuit Technology (ICSICT), Qingdao, China, 31 October–3 November 2018; pp. 1–3. [CrossRef]
13. Peesapati, R.; Das, S.; Baldev, S.; Ahamed, S.R. Design of streaming deblocking filter for HEVC decoder. *IEEE Trans. Consum. Electron.* **2017**, *63*, 1–9. [CrossRef]
14. Jiang, L.; Yang, Q.; Zhu, Y.; Deng, J. A parallel implementation of deblocking filter based on video array architecture for HEVC. In Proceedings of the 2016 Seventh International Green and Sustainable Computing Conference (IGSC), Hangzhou, China, 7–9 November 2016; pp. 1–7. [CrossRef]
15. Park, S.; Ryoo, K. Hardware design of HEVC in-loop filter for ultra-HD video encoding. *Lect. Notes Electr. Eng.* **2019**, *518*, 405–409. [CrossRef]
16. Shen, W.W.; Shang, Q.; Shen, S.; Fan, Y.; Zeng, X. A high-throughput VLSI architecture for deblocking filter in HEVC. In Proceedings of the IEEE International Symposium on Circuits and Systems, Beijing, China, 19–23 May 2013; pp. 673–676.
17. Shen, S.; Shen, W.; Fan, Y.; Zeng, X. A pipelined VLSI architecture for Sample Adaptive Offset (SAO) filter and deblocking filter of HEVC. *IEICE Electron. Express* **2013**, *10*, 20130272. [CrossRef]
18. Ozcan, E.; Adibelli, Y.; Hamzaoglu, I. A high performance deblocking filter hardware for high efficiency video coding. *IEEE Trans. Consum. Electron.* **2013**, *59*, 714–720. [CrossRef]
19. Hautala, I.; Boutellier, J.; Hannuksela, J.; Silvén, O. Programmable low-power multicore coprocessor architecture for HEVC/H. 265 in-loop filtering. *IEEE Trans. Circuits Syst. Video Technol.* **2014**, *25*, 1217–1230. [CrossRef]
20. Yan, C.; Zhang, Y.; Dai, F.; Wang, X.; Li, L.; Dai, Q. Parallel deblocking filter for HEVC on many-core processor. *Electron. Lett.* **2014**, *50*, 367–368. [CrossRef]
21. Mody, M.; Nandan, N.; Hideo, T. High throughput VLSI architecture supporting HEVC loop filter for Ultra HDTV. In Proceedings of the 2013 IEEE Third International Conference on Consumer Electronics—Berlin (ICCE-Berlin), Berlin, Germany, 9–11 September 2013; pp. 54–57.
22. Wang, Y.; Guo, X.; Fan, X.; Lu, Y.; Zhao, D.; Gao, W. Parallel In-Loop Filtering in HEVC Encoder on GPU. *IEEE Trans. Consum. Electron.* **2018**, *64*, 276–284. [CrossRef]
23. Kim, H.; Ko, J.; Park, S. An Efficient Architecture of In-Loop Filters for Multicore Scalable HEVC Hardware Decoders. *IEEE Trans. Multimed.* **2017**, *20*, 810–824. [CrossRef]
24. Kotra, A.M.; Raulet, M.; Deforges, O. Comparison of different parallel implementations for deblocking filter of HEVC. In Proceedings of the 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, BC, Canada, 26–31 May 2013; pp. 2721–2725.
25. Liu, L.; Chen, Y.; Deng, C.; Yin, S.; Wei, S. Implementation of in-loop filter for HEVC decoder on reconfigurable processor. *IET Image Process.* **2017**, *11*, 685–692. [CrossRef]
26. Shen, W.; Fan, Y.; Bai, Y.; Huang, L.; Shang, Q.; Liu, C.; Zeng, X. A Combined Deblocking Filter and SAO Hardware Architecture for HEVC. *IEEE Trans. Multimed.* **2016**, *18*, 1022–1033. [CrossRef]
27. Baldev, S.; Shukla, K.; Gogoi, S.; Rathore, P.K.; Peesapati, R. Design and Implementation of Efficient Streaming Deblocking and SAO Filter for HEVC Decoder. *IEEE Trans. Consum. Electron.* **2018**, *64*, 127–135. [CrossRef]
28. Jiang, W.; Mei, H.; Lu, F.; Jin, H.; Yang, L.T.; Luo, B.; Chi, Y. A novel parallel deblocking filtering strategy for HEVC/H.265 based on GPU. *Concurr. Comput. Pract. Exp.* **2016**, *28*, 4264–4276. [CrossRef]
29. Dai, Y.; Liu, D.; Zha, Z.J.; Wu, F. A CNN-Based In-Loop Filter with CU Classification for HEVC. In Proceedings of the 2018 IEEE Visual Communications and Image Processing (VCIP), Taichung, Taiwan, 9–12 December 2018; pp. 1–4.
30. Park, W.S.; Kim, M. CNN-based in-loop filtering for coding efficiency improvement. In Proceedings of the 2016 IEEE 12th Image, Video, and Multidimensional Signal Processing Workshop (IVMSP), Bordeaux, France, 11–12 July 2016; pp. 1–5. [CrossRef]
31. Zuo-Cheng, Z.; Ke-Bin, J. Key technologies and new developments of next generation video coding standard HEVC. In Proceedings of the 2013 Ninth International Conference on Intelligent Information Hiding and Multimedia Signal Processing, Beijing, China, 16–18 October 2013; pp. 125–128.
32. Na, T.; Kim, M. A novel no-reference PSNR estimation method with regard to deblocking filtering effect in H. 264/AVC bitstreams. *IEEE Trans. Circuits Syst. Video Technol.* **2013**, *24*, 320–330.
33. Tanchenko, A. Visual-PSNR measure of image quality. *J. Vis. Commun. Image Represent.* **2014**, *25*, 874–878. [CrossRef]
34. Ou, T.S.; Huang, Y.H.; Chen, H.H. SSIM-based perceptual rate control for video coding. *IEEE Trans. Circuits Syst. Video Technol.* **2011**, *21*, 682–691.

35. Wang, S.; Rehman, A.; Wang, Z.; Ma, S.; Gao, W. Perceptual video coding based on SSIM-inspired divisive normalization. *IEEE Trans. Image Process.* **2012**, *22*, 1418–1429. [CrossRef] [PubMed]
36. Winkler, S. *Digital Video Quality: Vision Models and Metrics*; John Wiley and Sons: Hoboken, NJ, USA, 2005.
37. Dosselmann, R.; Yang, X.D. A comprehensive assessment of the structural similarity index. *Signal Image Video Process.* **2011**, *5*, 81–91. [CrossRef]
38. Wang, Y. Survey of Objective Video Quality Measurements. 2006. Available online: https://digitalcommons.wpi.edu/computerscience-pubs/42 (accessed on 10 September 2019).
39. Wei, W.Y. *Deblocking Algorithms in Video and Image Compression Coding*; National Taiwan University: Taipei, Taiwan, 2009.
40. Christopher, P.R.; Sathasivam, S. Five-stage pipelined dual-edge deblocking filter architecture for H.265 video codec. *IEICE Electron. Express* **2019**, *16*, 20190500. [CrossRef]
41. *H.265-ITU-T*; SERIES H: Audiovisual and Multimedia Systems—Infrastructure of Audiovisual Services—Coding of Moving Video, High Efficiency Video Coding. Telecommunication Standardization Sector of ITU: Paris, France, 2018; Recommendation ITU-T H.265.