

# Serial parallel dataflow-pipelined processing architecture based accelerator for 2D transform-quantization in video coder and decoder

**Sumalatha Shivarudraiah, Rajeswari**

Department of Electronics and Communication Engineering, Acharya Institute of Technology, Bangalore, India

## Article Info

### Article history:

Received Apr 9, 2024

Revised Oct 17, 2024

Accepted Oct 21, 2024

### Keywords:

Contrast sensitivity function  
Discrete cosine transforms  
Field programmable gate array  
High efficiency video coding  
Human visual system  
Modulation transfer function  
Versatile video coding

## ABSTRACT

The video coder and decoder (CODEC) standards from MPEG-4 to the recent versatile video codec (VVC), adopted lossy compression methodologies, which involves transformation, quantization and entropy coding. The growing usage of video data in all means of communication demands more bandwidth and storage requirements. While compression with redundancy removal by transform coefficient coding, the focal point is the crucial sequential data flow and data processing structures. Handling the block wise data near to the processing unit prior and after computation will reduce the data waiting time of the processing unit, hence accelerating the targeted functionality. The proposed serial parallel data-flow pipelined processing architecture (SPDPA) accelerates the speed of processing unit by on chip data availability and parallel data accessing options and also with the pipeline operations of transformation, data transpose and quantization. The post implementation results of the architecture targeted to 16 nm and 28 nm field programmable gate array (FPGA) shows that there is a trade-off between power and frequency of operations for various block sizes. The design targeted to 16 nm works for higher frequencies with an average power consumption 0.64 w as compared to 28 nm FPGA which consumes less average power of 0.15 w.

*This is an open access article under the [CC BY-SA](#) license.*



## Corresponding Author:

Sumalatha Shivarudraiah

Research Scholar, Department of Electronics and Communication Engineering

Acharya Institute of Technology

Bangalore, India

Email: sumalatha.disha@gmail.com

## 1. INTRODUCTION

Video coder and decoder (CODEC) standards starting with the latest versatile video codec (VVC) 2020 down to any previous standard high efficiency video coding (HEVC) 2013 and advanced video coding (AVC) 2003 had a goal of achieving high video quality, reducing the bandwidth and storage requirements [1], [2]. Reducing bit rate in the order of 30% to 50% over previous standards and maintaining high video quality is possible only by using sophisticated coding tools and algorithms. By adopting the newer coding tools like increasing the number of intra and inter prediction modes, including primary transformation techniques like discrete cosine transform (DCT)-type-II/V/VIII, discrete sine transform (DST)-type-I/VII, secondary low frequency non separable transformation (LFNST) types with rectangular transformation, and frequency dependent and rate dependent perceptual quantization, increases the computational complexity [3]. In order to perform all these computationally intensive transformation and quantization of video frames efficiently, the hardware architecture implementation reported a performance gain over software only solutions, especially for a real time processing on an embedded platform. As the new era of silicon-on-chip (SoC) field programmable

gate array (FPGA) ranging from low end to high end comes with hybrid processing elements like digital signal processors (DSPs), GPUS and CPU which supports hardware and software co-design, where an architect can partition the complex video CODEC'S implementation on both CPU-which can handle more sequential data-flow and control intensive part while allowing FPGA to handle reconfigurable transform and quantization acceleration tasks [4].

## 2. LITERATURE REVIEW ON TRANSFORMATION AND QUANTIZATION

Video CODEC is responsible to satisfy all the needs of consumer electronic requirements like data security, internet bandwidth and storage which is eventually possible by encoding of the video frames after transformation and quantization. The transformation and quantization of video frames involves crucial sequential data flow, data access and tight data dependent processing structures. The HD video frames are initially converted to blocks of size varying from  $4 \times 4$  to  $128 \times 128$  and then submitted for transformation and quantization. This block wise residual pixels of the video frame after intra and inter predictions were transformed from spatial to frequency domain using cosine/sine transformation technique, in-order to de-correlate an important information from redundant within the block. Transformation followed by quantization, helps in defining the finer levels for encoding the transform coefficients and removing perceptual redundant data, hence able to achieve first level of data compression.

The well-known transformation architecture proposals for HEVC and VVC rely on two 1D processors to transform rows and columns by taking advantage of separable properties, connected through a transposition memory. Reconfigurable architecture for HEVC 2D-DCT, supporting block sizes from  $4 \times 4$  to  $32 \times 32$  was proposed by [5]. The design targets logical elements like multipliers and DSP blocks with local storage memory elements on FPGA. The synthesized design reported in the result sustains 4Kp30 encoding. The processing technique used in [6]–[8] is an even-odd decomposition-based transformation by shift and add units which is more suitable for the reconfigurable FPGA platform than the matrix multiplication method. To reduce the computational complexity of DCT-II in HEVC, Meher *et al.* [6] proposed different integer approximated architectures of folded, full parallel structures with pruning. The additional adder tree and muxes used in the design increase data path complexity and latency. The buffer used between 1D and 2D transformation for matrix transpose operation reported in the literature is either a combination of register array and multiplexers [6] or based on RAMs [7]. A unified adaptive multiple transform (AMT) architecture performing transformation of all square and asymmetric size combinations from 4, 8, 16, and 32 using multiplier IP cores and DSPs suitable for VVC were presented in [9] can render 2 K resolution video coding at 50 fps, but this design [9] is proposed and tested for encoder path only. In [10], [11] a general multiplier-based pipelined 2D-transformation with dual port SRAM in matrix form is used as a transpose memory. This approach of transpose memory utilizes more area on the targeted platform. The primary DCT-II and secondary transforms like DCT-VIII and DST-VII hardware implementations [12]–[14] reported good performance on SoC FPGA. The new integer DCT coefficients derived in [15] had a goal of similar performance compared to the original DCT, with a trade-off between resource cost and compression. An architecture supporting all transform sizes of HEVC by recursive processing was proposed by [16] involves identifying the number of pipelined registers to be included in the critical path to obtain all the outputs in a single clock cycle. The recent VLSI implementation of integer architecture based on obfuscation technique and systolic array structure [17] with minimal computational overhead reported good speed and low power consumption.

The usual coding tool applied after transformation is the quantization to remove perceptual redundancy. Quantization process in video codec standard plays a crucial role in achieving high compression efficiency without significant loss in visual quality. The isotropic human visual system (HVS) model proposed by Daly is adopted for DCT based JPEG image compression to derive a perceptually adaptive quantization table [18], which is used as a default QM<sub>intra</sub> matrix in HEVC. Keeping HVS-contrast sensitivity function (CSF) model in mind, the frequency based quantization matrix (QM) [19] is suggested to scale low frequency coefficients by finer values than high frequency within the transformed block. The default frequency dependent QM based on intra and inter predicted type transform blocks with transform size is defined [19] only for  $4 \times 4$  and  $8 \times 8$  size. For higher block size the  $8 \times 8$  size matrix values are repeated one to  $2 \times 2$  pattern for  $16 \times 16$  size matrix and one to  $4 \times 4$  pattern for  $32 \times 32$  size matrix respectively. The commonly applied block based lossy video compression tools to meet network requirement of ultra high definition television (UHDTV), has to handle problems like blocking, ringing and blurring artifacts. The contouring artifacts are most commonly noticeable in ultra high definition (UHD) displays, because of coarsely quantized high frequency values by scalar quantization. To avoid this contouring problem, an adaptive quantize values to be considered [20] to avoid zeroing of dead zone values and false edges. The improvement in HVS-CSF model for high resolution displays suggested by [21] and developed adaptive QM for scalable HEVC, where high frequency coefficients were quantized with lower weight values and hence had to pay a bit more budget. The perceptual redundancies are exploited by combining the lossless transform step with quantization in all the video CODEC standards.

So the position importance of the transformed pixels by Euclidean distance measurement from DC coefficient to all other AC coefficients in Luma and Chroma Cb and Cr transform blocks and normalized display resolution hypotenuse for QM derivation is considered [21], [22]. Fitting the complex Barten's CSF model to Daly model, suitable for high dynamic range (HDR)-ultra high definition video, developed [23] for both Luma and Chroma CSF tuned frequency weighting matrices (FWM) for  $8 \times 8$  transform unit (TU) size. Then this matrix can be up and down sampled to derive QM for other sizes. The first QM for Luma and Chroma coding was investigated by considering the CSF of DCT subbands for RGB videos [24]. The R, G, and B channels were combined with 1:1:4 ratios with high priority assigned to G-channel for QM derivation. The visual quality metric analysis and the corresponding experimental works are reviewed detail in [25] which are more on HVS. They also suggested learning based adaptive quantization can advance the performance and will also be suitable for machine vision applications.

From the literature review, it is revealed that, most researchers suggested different architectures for 2D transformation of square and rectangular block sizes. Also proposed integer approximation of transformation kernel coefficients suitable for VLSI implementations and to reduce complexity of handling real values. Next defined various methods for transposing the intermediate 1D DCT result suitable for 2D transformation. Then applying quantization for perceptual redundancy removal and to support entropy coding. In all this work, handling multiple data values from the external source or storage to processing unit and processing data in parallel within a pipeline architecture still remained challenging. Also quantizing transform coefficients to give a better balance between redundancy removal and improving visual quality on a HD display is a domain of research interest. The main contribution of the proposed work to address the gaps include i) the size of data selection based on block size and image width to perform transformation operation of multiple data in parallel. Also, this approach allows flexible transformation of square and rectangular data sizes specified in new CODEC standards; ii) having data near to the processing unit loaded to line buffers and new technique of 1D-DCT result transpose using demux and linebuffers accelerates the processing of transformation and quantization operations; and iii) adaptive quantization method based on display resolution to have a trade-off between visual quality and number of encoding bits per pixel during entropy coding.

The rest of the section is organized as follows. Section 3 gives the outline of our proposed method and detailed mathematical model with an architectural framework of integer approximated 2D-DCT transformation and perceptually optimized adaptive quantization modules. In section 4, the data flow accelerations and pipeline operations of proposed architecture are described with simulation and implementation results on the targeted FPGA evaluation boards. Finally, the conclusion is covered in section 5.

### 3. PROPOSED METHODOLOGY

The architectural framework in Figure 1 gives the outline of the processing sub-modules and data flow acceleration. The test input will be loaded serially into the on-chip line-buffers for 1D transformation along the row. The number of line buffers instantiated will be based on transform size selection and depth of each line buffer is equal to image width shown in Figure 1.

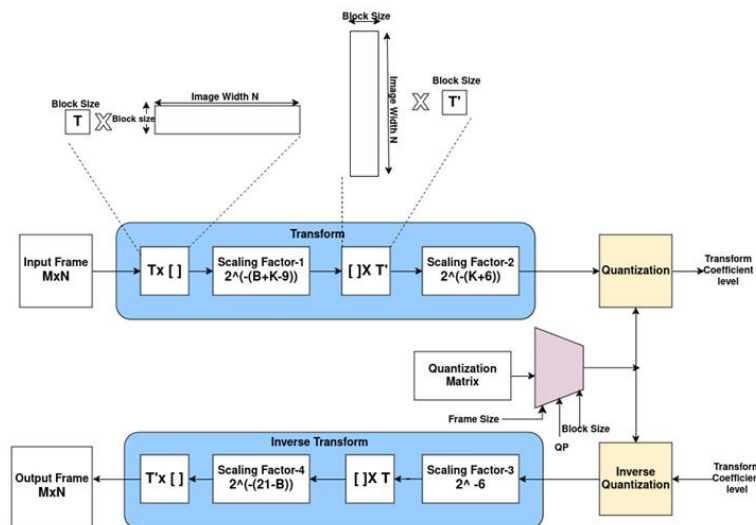


Figure 1. Proposed serial parallel data-flow pipelined processing architecture-hardware accelerator framework

The main objective of required data availability near the processing unit is achieved by these line buffers. The transform kernel coefficients -T are loaded into a read only memory (ROM) based on transform size selection. The first 1D-transformation along row-wise and second 2D-transformation along column-wise will be performed by general matrix multiplication method. After every multiplication usual data truncation is performed by scaling to maintain data size of 16-bit after every step. The scaling factor depends on pixel depth 'B' and 'K' which is a logarithmic value of the selected block size 'N'. The 1D-transformation output will be transposed by DEMUX and line buffers, which will be fed to the second column-wise transformation by multiplication with 'T'. This way of transformation supports both square and rectangular transformation of various sizes as shown in Figure 1. The 2D-transformation output will be then quantized based on quantization parameter, block size, and frame size configured by control unit to obtain transform coefficient level. As the proposed architecture is a unified structure, the process of forward transformation is performed in reverse order for inverse transform and inverse quantization. The serial and parallel dataflow through sub modules of the architecture with four stage pipelined operations are described in detail in the next sections.

### 3.1. Integer approximated 2D-transform architecture

The most widely used transformation type in image processing and video compression standard is the DCT-II. The unified separable 2D transform of an input image/frame of size MxN is computed in (1).

$$T_{i,j} = \begin{cases} \frac{1}{\sqrt{N}} & \text{if } i = 0 \\ \sqrt{\frac{2}{N}} \cos \frac{(2j+1)i\pi}{2N} & \text{if } i > 0 \end{cases} \quad (1)$$

In (1),  $T_{i,j}$  represents the element value of the transformation coefficient matrix in real; i is the row index, j is the column index, N is the transform size; and  $i, j = 0, 1, \dots, N-1$ . In order to process the transformation in integer, the integer approximation of real coefficients in (1) can be obtained by (2) defined as:

$$T_{i,j} = \text{round}[2^n * t_{i,j}] \quad (2)$$

where  $n=6 + \frac{\log_2 N}{2} = 6 + \frac{K}{2}$  and N= transformation size.

The two-dimensional transform of an MxN block of residual matrix can be achieved by (3), first applying a one-dimensional transform to each row of the block and then applying another one-dimensional transform to each column of the row-transformed result.

$$Y = TXT' \quad (3)$$

The matrix form of the 4-point one-dimensional transform is given by (4).

$$P = T * X = \begin{bmatrix} P_{00} & P_{01}P_{02} & P_{03} \\ P_{10} & P_{11}P_{12} & P_{13} \\ P_{20} & P_{21}P_{22} & P_{23} \\ P_{30} & P_{31}P_{32} & P_{33} \end{bmatrix} = \begin{bmatrix} T_{00} & T_{01}T_{02} & T_{03} \\ T_{10} & T_{11}T_{12} & T_{13} \\ T_{20} & T_{21}T_{22} & T_{23} \\ T_{30} & T_{31}T_{32} & T_{33} \end{bmatrix} * \begin{bmatrix} X_{00} & X_{01}X_{02} & X_{03} \\ X_{10} & X_{11}X_{12} & X_{13} \\ X_{20} & X_{21}X_{22} & X_{23} \\ X_{30} & X_{31}X_{32} & X_{33} \end{bmatrix} \quad (4)$$

As shown in (4), X represents the pixel residual matrix, T is the 4-point transform kernel matrix, and P is the 1D-transformed resultant matrix. The calculation formula for the second level of transformation to get 2D-transform output can be expressed as in (5).

$$Y = P * T' = \begin{bmatrix} Y_{00} & Y_{01}Y_{02} & Y_{03} \\ Y_{10} & Y_{11}Y_{12} & Y_{13} \\ Y_{20} & Y_{21}Y_{22} & Y_{23} \\ Y_{30} & Y_{31}Y_{32} & Y_{33} \end{bmatrix} = \begin{bmatrix} P_{00} & P_{01}P_{02} & P_{03} \\ P_{10} & P_{11}P_{12} & P_{13} \\ P_{20} & P_{21}P_{22} & P_{23} \\ P_{30} & P_{31}P_{32} & P_{33} \end{bmatrix} * \begin{bmatrix} T_{00} & T_{10}T_{20} & T_{30} \\ T_{01} & T_{11}T_{21} & T_{31} \\ T_{02} & T_{12}T_{22} & T_{32} \\ T_{03} & T_{13}T_{23} & T_{33} \end{bmatrix} \quad (5)$$

Based on the above steps of computation the architecture is designed as shown in Figure 2. The complete pipeline architecture has mainly four important modules: i) input pixel control unit, ii) line-buffers, iii) processing unit, and iv) output buffer. The transform accelerator module has a very regular structure for both 1D and 2D forward/inverse transform. Hence gives a more efficient pipelining of the sub module as well as maximum frequency of operation.

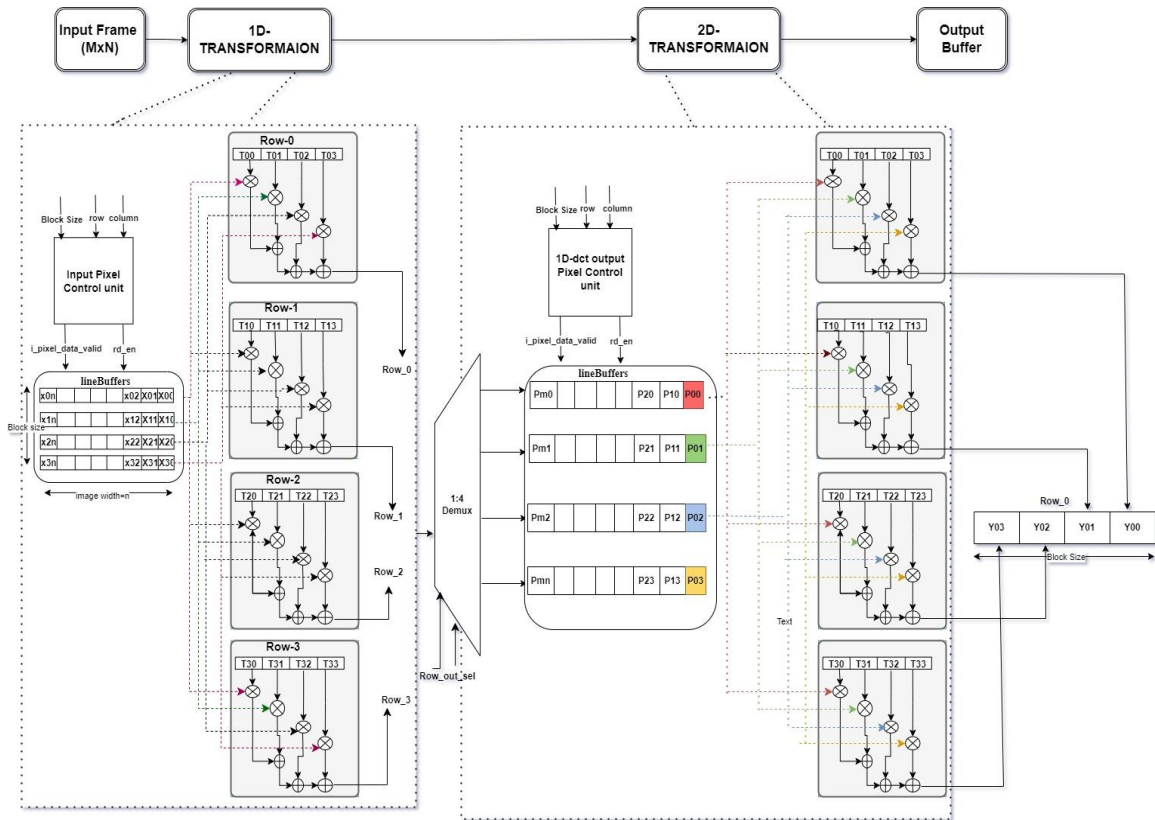


Figure 2. Pipeline architecture of 2D-transformation

### 3.1.1. Input pixel control unit

The main function of this control unit is to handle the data movement from external sources to line-buffers i.e. data writing operations, and feed to processing elements from line-buffers i.e. data reading operation. The number of rows to be accessed from the input frame depends on the block size considered for operation. The block size selection for transformation and quantization is parameterized, so the corresponding number of line-buffers and other functional units can be instantiated accordingly. The state diagram in Figure 3 illustrates the read and write control of line-buffers. The depth of the line-buffer is decided by the number of columns in the input frame and size of each location is equal to input pixel size.

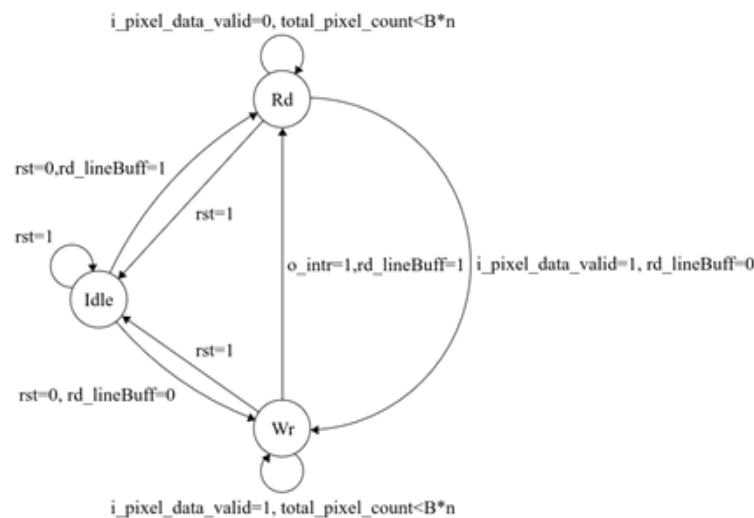


Figure 3. Finite state machine for input control unit

### 3.1.2. Line-buffers

As shown in the hardware architecture framework in Figure 2, the line-buffers are used at the input stage of 1D and 2D transforms. The total depth of the line-buffers used at the input of both 1D and 2D transforms is equal to the column value of the input frame. The data width of line-buffer used before 1D transformation is 8-bit, as it stores input pixel values and the data width of line-buffer used at the input of 2D transformation is a 1D-DCT output of 16-bit. The register transfer level (RTL) elaborated line buffer module is shown in Figure 4.

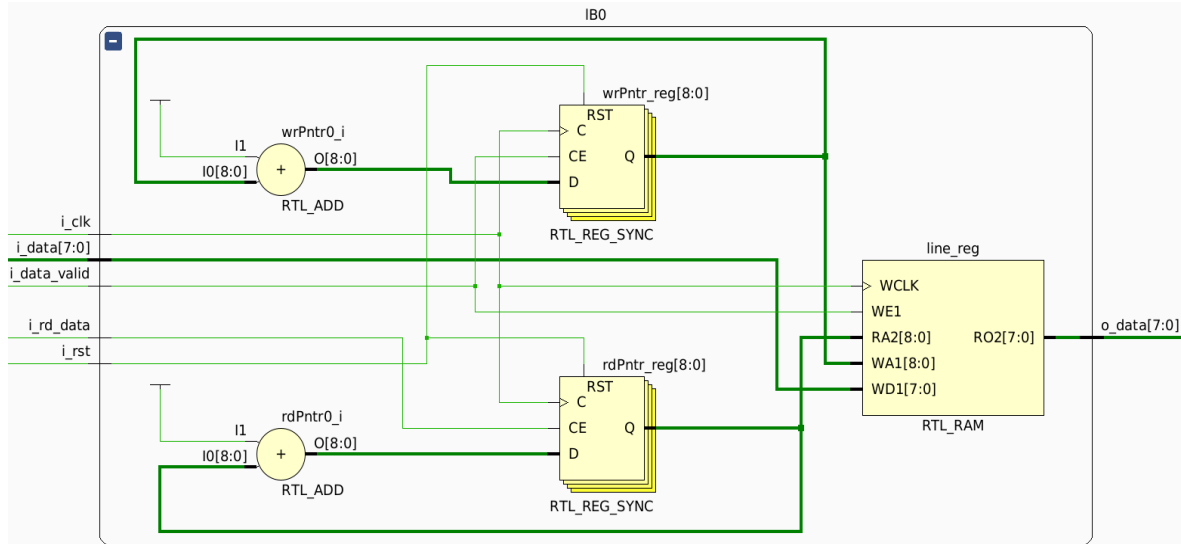


Figure 4. RTL elaborated line-buffer module before 1D-transformation

### 3.1.3. Processing unit

If the block size selected for transformation is  $4 \times 4$ , then based on (4), the four row elements of the transform matrix coefficients and four column elements of the data matrix are fed to the computing unit. This is basically an element wise multiplier and adder. During the first 1D transformation step, first four transformation coefficients get multiplied with the four column data elements, which are continuously streamed by input line-buffers, thereby resulting in the transformed row-0 output. Then the next row four transform coefficients are selected and multiplied with the previously selected column data values to generate row-1 output values of a 1D-transformation. For deciding the 1D-row wise transformation operation the  $4:1$  mux is used. The obtained 1D-transformed values are transposed using  $1:4$  demux and the second set of line-buffers. These 1D-transformed values are applied to the column-wise 2D transformation module as depicted in (5). The result of this module is the row-wise final output of the transformed frame.

### 3.1.4. Output buffer

The transformed results are sent sequentially to this output buffer, which is basically an AXI-stream based single clock enabled FIFO. The depth of FIFO is kept 16 bytes and width is 16-bit. Through this buffer module, resultant data can be sent to the DRAM memory and to the quantization module or to any transform dependent processing module, i.e. even for inverse transformation without quantization.

## 3.2. Perceptually adaptive frequency dependent quantization

The architecture mainly considers the adaptive QM derived based on display resolution parameter 'w', the perceptual important weight value consideration on distance measurement 'Eud' between DC and AC coefficients within TU block size and finally modifying 2D FWM,  $H(u,v)$ . In this entire process the methodology proposed in [21], [22] is followed and the parameters listed were modified in equations, explained in detail as follows.

### 3.2.1. Quantization matrix based on display resolution

Based on display screen size, the normalized hypotenuse value parameter 'h', in pixels can be modeled as in (6):

$$h = \frac{h_{actual}}{h_{theoretical}} \in [0,1] \quad (6)$$

where,  $h_{theoretical}$ = theoretical maximum hypotenuse value, in pixels and  $h_{actual}$ = actual maximum hypotenuse value, in pixels.

Calculating 'h' based on resolution of display unit, depends on  $h_{theoretical}$  and  $h_{actual}$  defined by using (7) and (8):

$$h_{theoretical} = \sqrt{x_{max}^2 + y_{max}^2} \quad (7)$$

$$h_{actual} = \sqrt{x^2 + y^2} \quad (8)$$

The theoretical maximum pixel values  $x_{max}$  and  $y_{max}$  on the maximum possible image size, in pixels, permitted in the JPEG standard is 65535×65535 [14]. Therefore, substituting  $x_{max}=65535$  and  $y_{max}=65535$  into (7) gives  $h_{theoretical}=92680.4858$ . Then the values x and y for 2 k and 4 k resolution display, as per standard HD size is considered in work is 1920×1080 and 3840×2160 respectively. Table 1 shows the  $h_{actual}$  and h values for both 2 k and 4 k resolution.

Table 1. Display resolution and corresponding hypotenuse

Resolution	x	y	$h_{actual}$	h
2 K	1920	1080	2202.9072	0.02377
4 K	3840	2160	4405.8153	0.04754

The resolution parameter w in-terms of normalized hypotenuse 'h' is quantified in (9). From (9) it is clear that w is totally controlled by appropriate normalized distribution of  $h_{theoretical}$  and h values.

$$w = h_{theoretical}^h \in [0,1] \quad (9)$$

### 3.2.2. Quantization matrix based on positional importance of transform coefficient

The block wise transformed coefficients are energy compacted pixel values, where the dc and low frequency AC components have to be retained with care and high frequency AC components have to be scaled. The pixel position based on this requirement can be calculated by normalized Euclidean distance parameter  $Eud_{(i,j)}$  given by (10).

$$Eud_{(i,j)} = \sqrt{\frac{(i_1-i_2)^2+(j_1-j_2)^2}{(i_1-i_{max})^2+(j_1-j_{max})^2}} \in [0,1] \quad (10)$$

Where,  $(i_1, j_1)$ = position of the DC coefficient,  $(i_2, j_2)$ = position of the current AC coefficient.  $(i_{max}, j_{max})$ = position of the last AC coefficient from DC coefficient for the considered block size.

Finally, a parameter  $M_{i,j}$  that relates both display resolution and perceptually important transform coefficients position in the considered block size is given by (11).

$$M_{i,j}(Eud_{i,j}, W) = e^{-\left[\frac{Eud_{i,j}}{1+W}\right]} \in [0,1] \quad (11)$$

Now this  $M_{i,j}$  is applied to each element of matrix  $H(x, y)$  located at position  $(i, j)$ , denoted as  $H_{i,j}$  defined in [21] to produce an adaptive 2D FWM  $H'(i, j)$ .

$$H'_{i,j} = H_{i,j}^{M_{i,j}} \quad (12)$$

$$QM = \frac{QP}{H(x,y)} \quad (13)$$

Finally, the adaptive QM in (13) can be obtained from (12) and the quantization parameter which controls quantization step size. This matrix will be used to quantize the transform coefficients by right shift operations instead of regular element-wise division operations. The same QM will be shared between forward



and inverse path as shown in Figure 1. Hence during inverse quantization, each transform coefficient level will be shifted left by the corresponding value specified in the QM instead of regular multiplications.

#### 4. RESULTS AND DISCUSSION

In this section, first the details of the experimental setup, like the tool used to code the proposed design with input test data, followed by functional verification of the 2D transform and quantization, are outlined. Once the functional correctness of individual modules and integrated system level is tested by simulation, the design is synthesized and implemented on the targeted SoC FPGA hardware. The implementation results on the hardware are tabulated in terms of resource utilization and operating frequency. Also shown are comparison results of the proposed design with the research works of others.

##### 4.1. Experimental setup and functional verification by simulation

The proposed architecture of 2D transformation and quantization accelerator modules and the corresponding testbench is coded in Verilog using Xilinx Vivado 2022.2. At the system level, all these modules are instantiated in the top module and compiled to fit the targeted hardware. The elaborated complete architecture is shown below with two split stages of 1D-transformation in Figure 5 and 2D-transformation in Figure 6.

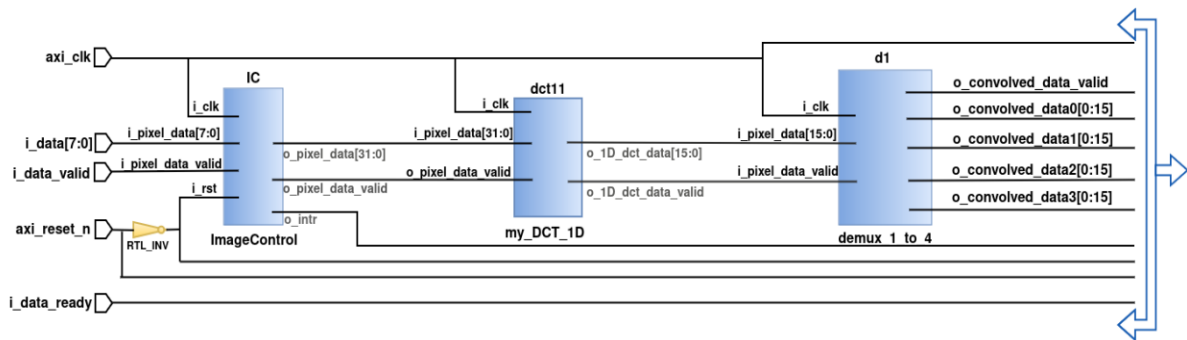


Figure 5. Elaborated schematic of 1D transformation stage

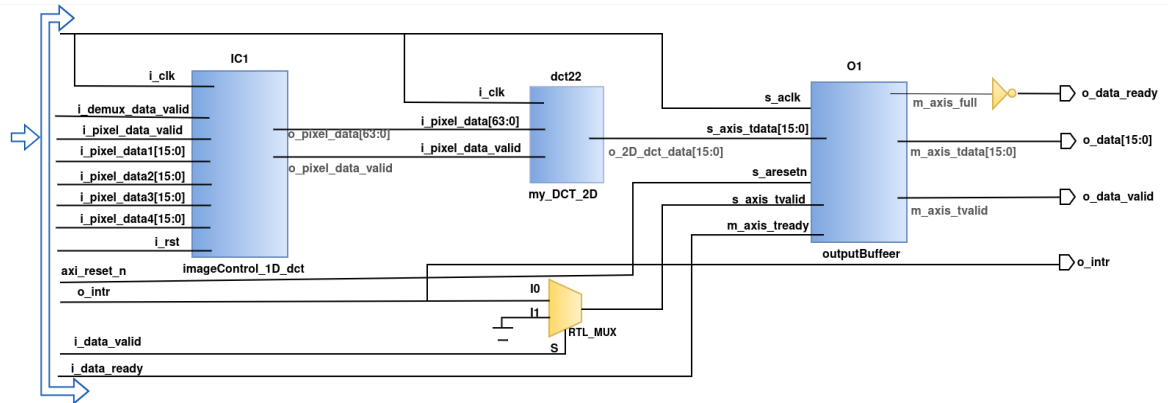


Figure 6. Elaborated schematic of 2D transformation stage

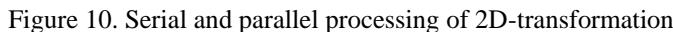
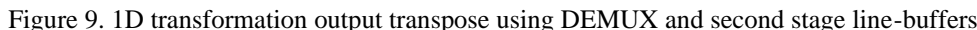
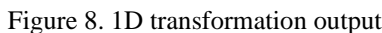
The pixel values of the image are read as a hexadecimal character using file operations supported in Verilog HDL included in test bench code. The simulation waveforms obtained from Vivado Simulator 2022.2 are shown starting from data read into line buffers then feeding to the transformation and quantization module followed by writing to the output buffer. In the following simulation waveform the stage wise operation of the accelerator modules are illustrated considering the block size four and image size 512×512 with pixel size of eight bit. The line buffers are loaded with input pixels, row wise sequentially one after the other shown in Figure 7(a) and reading four-pixel values column wise for feeding to transformation operation is shown in Figure 7(b).





(b)

As the input image has 512 columns, total  $512 \times 4 = 2048$  pixels are written first into the line buffers and it will be processed by the 1D transform module as per equation (4). As shown in Figure 8 the 1D transformation of the first 4 pixels takes three clock cycles, where multiplication and accumulation of four partial products happens in one clock cycle and then moving data through pipeline registers takes two clock cycles. So to get one set of four transformed row DCT output, it takes 7 clock cycles. Now each of these 1D-DCT row values are transposed using de-multiplexer and second stage line buffers as shown in Figure 9. Based on de-multiplexer select line input, the de-multiplexer output will be fed to line buffers at every clock cycle, which will be accessed by 2D-transformation module for column-wise transformation as per (5) and shown in architecture Figure 2. At every clock cycle the 1D transformed pixel gets multiplied with transposed kernel coefficients to produce four partial product columns, shown in Figure 10. All these partial products are accumulated row wise at fourth clock cycles to output four 2D- transformed results.



Hence in the proposed architecture, based on the selected block size, the multiple pixel data, made available near the processing module within line buffers and then this data will be processed to have multiple outputs in parallel. The QM obtained from (13) for the selected quantization parameter will be used to quantize every 2D-transformed output pixel by right shift operation. Hence the proposed design has four pipeline stages like 1D transformation, data transpose, 2D transformation and quantization. In the proposed pipeline architecture, for the selected  $4 \times 4$  block size processing shown in Figure 11 has a latency of 15 clock cycles from input to output. The Table 2 shows the total clock cycles required to process transformation and quantization of different block sizes from  $4 \times 4$  to  $32 \times 32$  with pipeline latency of each stage.

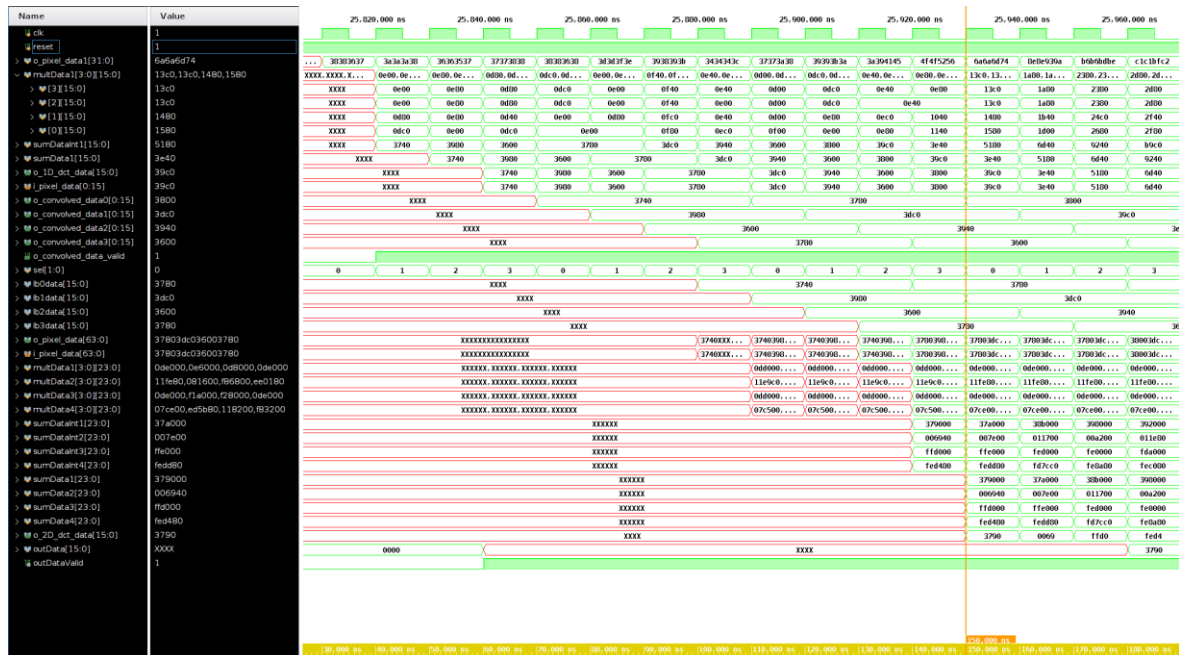


Figure 11. 2D-transformation and quantization pipelined operation output for block size  $4 \times 4$

Table 2. Number of clock cycles required for 2D-DCT

Latency of pipeline stages	Block size $4 \times 4$	Block size $8 \times 8$	Block size $16 \times 16$	Block size $32 \times 32$
DCT-1 I/p to DCT-1 o/p	3	3	3	3
Transpose 1D-DCT o/p	4	4	4	4
DCT-2 i/p to DCT-2 o/p	4+1=5	8+1=9	16+1=17	32+1=33
Quantization and final o/p through output buffer	3	3	3	3
Total clock cycles	15	19	27	43

#### 4.2. Synthesis results and discussion

The Xilinx Vivado 2022.2 synthesis tool is used to synthesize the proposed scalable transformation and quantization pipeline architecture which does the unified forward and inverse operations of input data in terms of block sizes  $4 \times 4$  to  $32 \times 32$ . The synthesis and implementation of the proposed architecture is targeted to two different SoC FPGA devices Zynq ZC702 and Zynq Ultrascale+ ZCU-104 respectively. The Table 3 shows that the proposed pipeline architecture has a trade-off between power consumption and speed, where on the low end FPGA ZC702 consumes less energy than high end ZCU-104 FPGA of average power consumption 0.15 W. The performance of the architecture on high end FPGA is almost twice that of low end FPGA. Also noticed that the operating frequency of the architecture decreases as the block size increases.

The proposed pipelined architecture is compared with other hardware implementations shown in Table 4. The resource utilization and number of clock cycles required to process the maximum block size of  $32 \times 32$  in the proposed architecture is compared with others work. The hardware implementation of [5] computes 2D-DCT operations by even-odd decomposition based butterfly structure and uses almost 8-times more DSP elements than our implementation. Also it takes [5], 500 clock cycles to process  $32 \times 32$  block size DCT, where our architecture requires only 43 clock cycles. The reduced number of clock cycles in the proposed architecture is because of two important techniques, first one the required amount of data for processing is

made available in advance near to the transformation module by having sufficient on chip line buffers and second one will be the processing of multiple data in parallel by four stage pipelined architecture. The number of flip flops and BRAM resource utilization in Table 4 for the proposed architecture is due to line buffers at every input and output stages of transformation and quantization operations. The synthesis result of 2D unified architecture to support VVC multiple transform cores by [14], shows more resource utilization and clock cycles to process maximum block size of  $32 \times 32$  on targeted Arria-10 SoC FPGA.

Table 3. 2D DCT synthesized results on 16 nm and 28 nm FPGA technology

		Design targeted FPGA EVA board							
		Zynq ultrascale+ MPSoC (16 nm) Zcu104 (xczu7ev-fvc1156-2e)				Zynq-7000 (28 nm) ZC702 (xc7z020-clg484-1)			
FPGA resources	Block size	4×4	8×8	16×16	32×32	4×4	8×8	16×16	32×32
	LUT	1,388	2,669	5,443	8,758	1,883	2,426	4,920	9,020
	FF	349	528	1,069	1,584	614	534	1,427	2,973
	BRAM	2.5	0.5	29	57.5	2.5	8	32.5	64.5
	URAM	Nil	1	Nil	Nil	Nil	Nil	Nil	Nil
Total power (W)	DSP	4	8	16	32	8	8	16	0
	On-chip (dynamic +static)	0.623	0.645	0.670	0.624	0.136	0.145	0.159	0.164
Freq (MHz)	1/(T-WNS)	144	140	87	58	101	79	38	25

Table 4. Performance comparison of 2D DCT-II

Hardware implementation	[5]	[14]	Proposed	Proposed
FPGA technology	Xilinx zynq-28 nm	Arria-10 SoC	Xilinx zynq-28 nm	Xilinx Zynq Ultrascale+ 16 nm
LUT/ ALM'S	5.8 K	26.1 K	9 K	8.7 K
FF	--	62.1 K	2.9 K	1.5 K
DSP	128	328	16	32
BRAM	--	64 K	64.5	57.5
Clock cycles	500	175	43	43
Frequency (MHz)	222	225	101	107
Transform size	4×4 to 32×32	4×4 to 32×32	4×4 to 32×32	4×4 to 32×32

## 5. CONCLUSION

In the proposed pipelined architecture, the integer approximated 2D-transformation with integration of perceptual models into the quantization process to enhance the visual quality of compressed videos is presented. The reconfigurable architecture supports various block sizes for unified transformation and quantization using only limited hardware resources on targeted FPGA. The overall latency from input to output of state-of-the-art architecture is less for processing maximum block size due to the novel approach of data acceleration technique. In future implementation, the goal is to increase the performance by enhancing possible data accessing and data processing methodology to support real time processing of 2K and 4K videos with run time selection of the processing parameters. Also, the high-level implementation of transformation and quantization accelerators and its testing on a heterogeneous platform with performance analysis is our future work.




## REFERENCES

- [1] G. J. Sullivan, J. R. Ohm, W. J. Han, and T. Wiegand, "Overview of the high efficiency video coding (hevc) standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1649–1668, 2012, doi: 10.1109/TCSVT.2012.2221191.
- [2] B. Bross *et al.*, "Overview of the versatile video coding (vvc) standard and its applications," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 10, pp. 3736–3764, 2021, doi: 10.1109/TCSVT.2021.3101953.
- [3] F. Bossen, K. Suhring, A. Wiecekowsky, and S. Liu, "VVC complexity and software implementation analysis," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 10, pp. 3765–3778, 2021, doi: 10.1109/TCSVT.2021.3072204.
- [4] Xilinx, "Zynq ultrascale+ mpsoC: embedded design tutorial," *xilinx*, vol. 1209, 2018.
- [5] M. Chen, Y. Zhang, and C. Lu, "Efficient architecture of variable size HEVC 2D-DCT for FPGA platforms," *AEU - International Journal of Electronics and Communications*, vol. 73, pp. 1–8, 2017, doi: 10.1016/j.aeue.2016.12.024.
- [6] P. K. Meher, S. Y. Park, B. K. Mohanty, K. S. Lim, and C. Yeo, "Efficient integer DCT architectures for HEVC," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 24, no. 1, pp. 168–178, 2014, doi: 10.1109/TCSVT.2013.2276862.
- [7] W. Zhao, T. Onoye, and T. Song, "High-performance multiplierless transform architecture for HEVC," in *2013 IEEE International Symposium on Circuits and Systems (ISCAS2013)*, IEEE, 2013, pp. 1668–1671, doi: 10.1109/ISCAS.2013.6572184.
- [8] R. Conceição, J. C. D. Souza, R. Jeske, B. Zatt, M. Porto, and L. Agostini, "Low-cost and high-throughput hardware design for the hevc 16x16 2-D DCT transform," *Journal of Integrated Circuits and Systems*, vol. 9, no. 1, pp. 25–35, 2014, doi: 10.29292/jics.v9i1.386.




- [9] A. Kammoun, W. Hamidouche, F. Belghith, J. F. Nezan, and N. Masmoudi, "Hardware design and implementation of adaptive multiple transforms for the versatile video coding standard," *IEEE Transactions on Consumer Electronics*, vol. 64, no. 4, pp. 424–432, 2018, doi: 10.1109/TCE.2018.2875528.
- [10] M. J. Garrido, F. Pescador, M. Chavarrias, P. J. Lobo, and C. Sanz, "A 2-D multiple transform processor for the versatile video coding standard," *IEEE Transactions on Consumer Electronics*, vol. 65, no. 3, pp. 274–283, 2019, doi: 10.1109/TCE.2019.2913327.
- [11] M. J. Garrido, F. Pescador, M. Chavarrias, P. J. Lobo, C. Sanz, and P. Paz, "An FPGA-based architecture for the versatile video coding multiple transform selection core," *IEEE Access*, vol. 8, pp. 81887–81903, 2020, doi: 10.1109/ACCESS.2020.2991299.
- [12] A. C. Mert, E. Kalali, and I. Hamzaoglu, "High performance 2D transform hardware for future video coding," *IEEE Transactions on Consumer Electronics*, vol. 63, no. 2, pp. 117–125, 2017, doi: 10.1109/TCE.2017.014862.
- [13] Y. Fan, Y. Zeng, H. Sun, J. Katto, and X. Zeng, "A pipelined 2D transform architecture supporting mixed block sizes for the VVC standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 9, pp. 3289–3295, 2020, doi: 10.1109/TCSVT.2019.2934752.
- [14] A. Kammoun *et al.*, "Forward-inverse 2D hardware implementation of approximate transform core for the VVC standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 11, pp. 4340–4354, 2020, doi: 10.1109/TCSVT.2019.2954749.
- [15] J. Chen, S. Liu, G. Deng, and S. Rahardja, "Hardware efficient integer discrete cosine transform for efficient image/video compression," *IEEE Access*, vol. 7, pp. 152635–152645, 2019, doi: 10.1109/ACCESS.2019.2947269.
- [16] P. K. Meher, S. K. Lam, T. Srikanthan, D. H. Kim, and S. Y. Park, "Area-time efficient two-dimensional reconfigurable integer DCT architecture for HEVC," *Electronics*, vol. 10, no. 5, pp. 1–11, 2021, doi: 10.3390/electronics10050603.
- [17] D. F. Chipper and A. Cracan, "An efficient algorithm and architecture for the VLSI implementation of integer DCT that allows an efficient incorporation of the hardware security with a low overhead," *Applied Sciences*, vol. 13, no. 12, 2023, doi: 10.3390/app13126927.
- [18] C. Y. Wang, S. M. Lee, and L. W. Chang, "Designing jpeg quantization tables based on human visual system," *Signal Processing: Image Communication*, vol. 16, no. 5, pp. 501–506, 2001, doi: 10.1016/S0923-5965(00)00012-6.
- [19] M. Budagavi, A. Fuldseth, and G. Bjøntegaard, "HEVC transform and quantization," in *High Efficiency Video Coding (HEVC): Algorithms and Architectures*, 2014, pp. 141–169, doi: 10.1007/978-3-319-06895-4\_6.
- [20] N. Casali, M. Naccari, M. Mrak, and R. Leonardi, "Adaptive quantisation in HEVC for contouring artefacts removal in UHD content," in *2015 IEEE International Conference on Image Processing (ICIP)*, IEEE, 2015, pp. 2577–2581, doi: 10.1109/ICIP.2015.7351268.
- [21] L. Prangnell and V. Sanchez, "Adaptive quantization matrices for HD and UHD resolutions in scalable HEVC," in *2016 Data Compression Conference (DCC)*, IEEE, 2016, pp. 626–626, doi: 10.1109/DCC.2016.47.
- [22] L. Prangnell, "Frequency-dependent perceptual quantization for visually lossless compression applications," *Arxiv-Computer Science*, pp. 1–26, 2019.
- [23] D. Grois and A. Giladi, "Perceptual quantization matrices for high dynamic range h.265/MPEG-HEVC video coding," in *Applications of Digital Image Processing XLII*, SPIE, 2020, p. 24, doi: 10.1117/12.2525406.
- [24] X. Shang, G. Wang, X. Zhao, Y. Zuo, J. Liang, and I. V. Bajic, "Weighting quantization matrices for HEVC/H.265-CODED RGB videos," *IEEE Access*, vol. 7, pp. 36019–36032, 2019, doi: 10.1109/ACCESS.2019.2902173.
- [25] Y. Zhang, L. Zhu, G. Jiang, S. Kwong, and C. C. J. Kuo, "A survey on perceptually optimized video coding," *ACM Computing Surveys*, vol. 55, no. 12, 2023, doi: 10.1145/3571727.

## BIOGRAPHIES OF AUTHORS



**Sumalatha Shivarudraiah**    worked as Assistant Professor in the Department of Electronics and Communication Engineering at Acharya Institute of Technology, Bangalore, Karnataka, India. She has 14 years of academic experience. Currently she is a research scholar and pursuing her Ph.D. from VTU, Belagavi. Her research interests include image processing, analog and digital circuits, VLSI and embedded systems. She is life time member of ISTE. She can be contacted at email: sumalatha.disha@gmail.com.



**Dr. Rajeswari**    is presently working as Professor and Head, Department of Electronics and Communication Engineering at Acharya Institute of Technology, Bangalore, India. She has completed her Ph.D. in the field of speech processing. Her areas of interests include speech processing, AI, computer vision and applications in the field of healthcare and agritech. She can be contacted at email: rajeswari@acharya.ac.in.