Low Power Motion Estimation Algorithm and Architecture of HEVC/H.265 for Consumer Applications

Karam Singh, *Member, IEEE*, and Shaik Rafi Ahamed, *Member, IEEE*

***Abstract*—High-quality videos like high-definition (HD) and ultra HD became an essential requirement in recent applications such as security surveillance, television system, etc. However, due to increase in resolution of the videos, the volume of visual information data increases significantly, which became a challenge for storage, transmission and processing the HD video data. The new video compression standard, high efficient video coding (HEVC), achieved two-fold video efficiency improvements as compared to H.264/AVC using efficient compression techniques. Motion estimation (ME) is one of the computationally intensive blocks in video CODEC. In HEVC, the complexity of ME further increases due to a large processing unit and flexible partitioning of the prediction unit (PU). In this paper, we proposed a low power ME algorithm and architecture of the HEVC for consumer applications. The proposed algorithm and architecture utilizes sub-sampling, data reuse, pixel truncation and adaptive search range techniques for reducing the computational power. Simulations result shows that the proposed ME algorithm requires an average of 53.82% fewer search points as compared to the reference software HM with a small degradation in PSNR and little increment in bit-rate. The proposed architecture is simulated and synthesized using standard 90 nm technology. The proposed ME architecture can process 3840 × 2160 @ 30 fps video sequences with only 4.5193 mm^2 of the area and 8.192 KB of SRAM. The operating frequency of the proposed architecture is 250 MHz with 151.7619 mW of power.**

***Index Terms*—CTU, HEVC, motion estimation, motion vector, PU, SAD.**

I. Introduction

In THE past decade, with the progress in technology, the quality of digital video has improved tremendously. Now a day's high-definition (HD) video has become very popular and several researchers are thinking about the ultra HD video for next-generation applications. It includes the resolutions of 4*K* (3840 × 2160) and 8*K* (7680 × 4320), which delivers 4× to 16× the number of pixels per frame compared with today's HD. To store, transmit and process such a huge amount of video data, efficient and real-time video compression is essential. Keeping this in mind, ISO/IEC Moving Picture Experts Group (MPEG) and ITU-T Video Coding Experts Group (VCEG) formed a Joint Collaborative Team on Video Coding (JCT-VC) for developing the next generation video coding standard in 2010. In 2013 JCT-VC finalized the first draft of the HEVC. It is also known as ISO/IEC 23008-2 MPEG-H part 2 and ITU-T H.265 [1]. As the HEVC is capable of processing huge amount of video data, the hardware implementation of HEVC is very complex and requires enormous power for compression. It becomes almost impractical to adopt HEVC in portable devices such as mobile phone and digital cameras due to their limited power sources. To overcome this problem a power efficient algorithm and architecture is very much required.

Like the previous video coding standards, HEVC also works on the hybrid block-based video coding technique [2], [3]. In hybrid block-based video coding, first each picture is partitioned into square size blocks of samples and then each block of samples is predicted with the help of the intra-picture and inter-picture prediction. In intra prediction, spatial redundancy is removed and only previously encoded block of the same picture are used as a reference block. Since inter prediction is used to remove temporal redundancy or in other word it tracks the motion of the real-world objects between the pictures of video sequences. Thus, previously encoded pictures are used as a reference. In both the cases, residual is calculated by taking the difference between the original picture and predicted picture. The residual picture is then sent to transform unit (TU), where it will be further divided into square size blocks depending on the type of transform used. HEVC supports 4 × 4, 8 × 8, 16 × 16 and 32 × 32 sizes of TUs. TU converts residual into transform coefficient. In the next step, the transform coefficient is quantized with the help of quantization parameter (QP) which lies in the range of 0-51. The entropy encoder converts quantized transform into a bit stream. In HEVC, Context-Based Adaptive Binary Arithmetic Coding (CABAC) is used, which is a lossless compression scheme that uses the statistical property to compress data. For improving the picture quality, the HEVC standard specifies two in-loop filters, a deblocking filter, and a sample adaptive offset (SAO) filter. These filters are applied in encoding and decoding loops, after the inverse quantization and before storing the picture in the decoded picture buffer. The deblocking filter firstly eliminates discontinuities at the prediction and transforms block boundaries. Then, SAO is applied to the output of the deblocking filter to improve the quality of the decoded picture by attenuating ringing artifacts and changes in the sample intensity of some area of a picture.

In HEVC, every picture may be divided into a set of the slice, which is part of frames that can be encoded/decoded

Manuscript received February 2, 2018; revised June 23, 2018 and August 17, 2018; accepted August 17, 2018. Date of publication August 29, 2018; date of current version September 24, 2018. *(Corresponding author: Karam Singh.)*

The authors are with the Department of Electronic and Electrical Engineering, Indian Institute of Technology Guwahati, Guwahati 781039, India (e-mail: karam@iitg.ernet.in; rafiahamed@iitg.ernet.in).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCE.2018.2867823

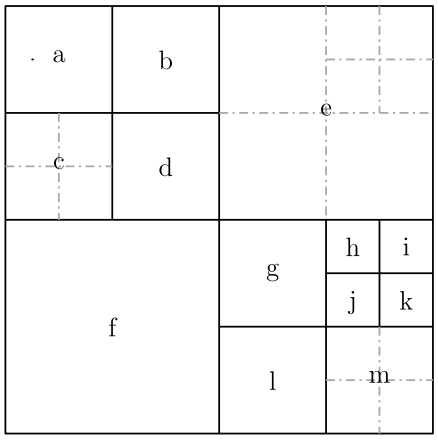

Fig. 1. Quadtree block partitioning scheme.

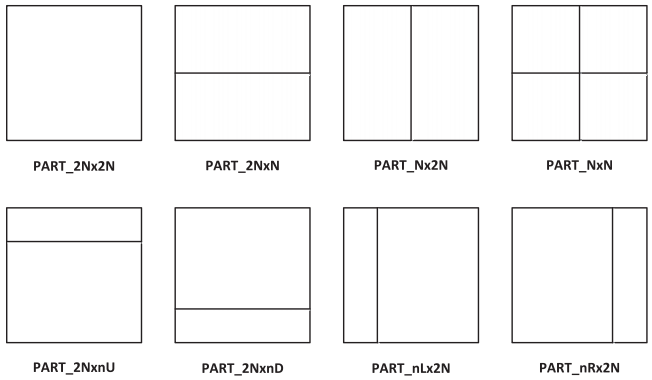

Fig. 2. Prediction unit size and partitioning.

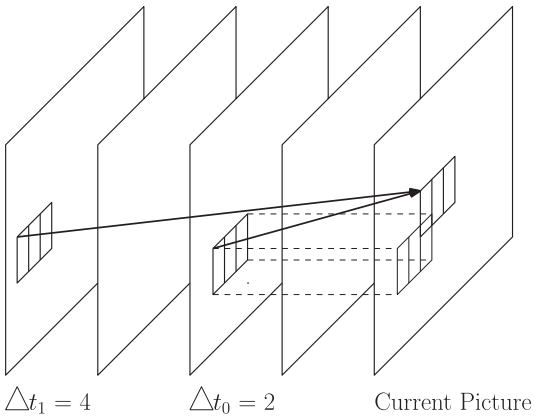

Fig. 3. Motion estimation principle.

independently. A slice is composed of the equal square size of coding tree units (CTUs). Each CTU consist of one coding tree block (CTB) of Luma samples, two corresponding square CTBs for chroma samples and associated syntax elements. CTU is the basic processing unit of HEVC video coding standard, which is conceptually same as a macroblock in H.264/AVC. The size of CTB is $2N_{max} \times 2N_{max}$, where N_{max} values are 8, 16 and 32 [1]. Each CTB can be partitioned into smaller coding blocks (CB) in the form of the quadtree structure as shown in Fig. 1. Each leaf node of the quadtree is called CB. The size of CB is $2N \times 2N$, where $N \leq N_{max}$. In HEVC, N can be 4, 8, 16 and 32. A CB defines the region that shares the same predication mode (Intra, Inter, SKIP, or Merge). Hence, it acts as the root of predication tree as well as transform tree. A prediction unit (PU) represents a picture region that shares the identical prediction information. That is, the same prediction process is applied for its luma and chroma prediction blocks (PBs). In prediction tree, each luma/chroma CBs can be further partitioned into one, two or four rectangular shape PBs as shown in Fig. 2. These PBs shapes are divided into two categories: symmetrical and asymmetrical. The symmetrical partitions contain $2N \times 2N$, $2N \times N$ and $N \times 2N$ sizes and asymmetrical contains $2N \times nU$, $2N \times nD$, $nL \times 2N$ and $nR \times 2N$ sizes [4]. HEVC set a certain limitation on PBs. Firstly, in inter-coded CBs $N \times N$ partition is possible only when the depth is max (depth $= 4$). Secondly, in inter-coded CBs 4×4 parts are completely disabled due to worst case memory bandwidth. Thirdly, AMP is disabled when N $= 4$ to avoid the dimension smaller than four. Finally, always square sizes are used in intra-coded CUs. The size of skip mode is always $2N \times 2N$.

HEVC improves the compression ratio by around 59% compared with H.264/Advanced Video Coding (AVC) [5] maintaining the same video quality or at the same bit-rate HEVC provide more visual quality than H.264/AVC. The coding efficiency/bit-rate compression of HEVC results due to various improvements as compared with H.264/AVC, such as quadtree block partitioning (Fig. 1), a new PU partition scheme (Fig. 2), larger CU size, advanced motion vector prediction (AMVP), new block merging techniques, more accurate interpolation filter for fractional motion estimation (FME), variable size of the Hadamard transform (HT), a large number of intra and inter prediction modes, larger TU size and new technique used in entropy encoding. Due to the above mentioned techniques, computational complexity is increased. The total software complexity of HEVC CODEC is 1.5 times as compared to H.264/AVC video coding standard [6]. The average time distribution of inter prediction in encoder is 69% [7]. Thus, it is the most computationally intensive block in HEVC CODEC. Hence, in order to reduce the total power consumption of CODEC, power efficient inter prediction algorithm and architecture is proposed.

In inter prediction, motion parameters are obtained through the ME, the heart of all existing video compression standards. ME is basically used to remove the temporal redundancy in video compression standards [8]. It tries to find the best possible matched block in the previously encoded frames and generates the motion vector (MV) as shown in the Fig. 3. MV is the distance between best-matched candidate block in a search window of the reference frame and the position of a current block in the current frame. ME involves a two-step process: - Integer ME (IME) and FME. In IME, the best-matched block is found at the integer position in the search area. In HEVC, IME is carried out for 593 (13 + 52 + 208 + 320) sub-blocks, which is a very large value as compared to the number of sub-blocks (41 sub-blocks) used in H.264/AVC [9]. In IME, distortion is calculated using the sum of the absolute difference/distortion (SAD) as shown in (1). FME refines luminance IMVs to 1/4-pixel accuracy and chrominance IMVs to 1/8-pixel accuracy [10]. HEVC uses 8-tap and 7-tap interpolation filter for calculating the pixel value at 1/2 and 1/4-pixel positions respectively for luminance samples and 4-tap interpolation filter for calculating the pixel value at 1/8-pixel positions for chrominance samples [10]. Both filters are upgraded as compared to AVC interpolation filter. In FME, distortion is calculated as a transform sum of the

absolute difference/distortion (SATD) as expressed in (2). For complex/fast moving objects, FME becomes more important and complex part because there is a very high probability that the best-matched blocks are smaller in size. FME is searched at half and quarter position for luminance samples. The mathematical expression for the SAD and SATD can be represented as follows:

$$SAD = \sum_{i,j} |BlockA(i,j) - BlockB(i,j)| \tag{1}$$

$$SATD = \left(\sum_{i,j} |Y(i,j)| \right) /2 \tag{2}$$

where (i, j) denotes the index point of two-dimensional block, *BlockA* and *BlockB* denote the current and reference block respectively. *Y* is the transformed matrix of distortion and calculates as

$$Y(i,j) = H_m X(i,j) H_m^T \tag{3}$$

Here H_m is the Hadamard matrix and X is distortion matrix given by

$$X(i,j) = (BlockA(i,j) - BlockB(i,j)) \tag{4}$$

Due to the large block size, asymmetrical PU partition, quadtree partition for CTU and computational complex interpolation filters, ME becomes a more computational intensive process in HEVC. Therefore, for consumer applications, there is a great demand to reduce the computational power of ME process.

II. Review Of Related Works

The best possible algorithm for finding the best-matched candidate is full search (FS). However, FS involves huge computational complexity. For reducing the complexity involved in ME, several efficient algorithms and architectures are reported in the literature. For example, the works in [11]–[14] are based on reducing the number of search points. The pixel truncation technique was employed in [15]. Zhou *et al.* [16] used sub-sampling of the images. Ismail *et al.* [17], uses early termination technique.

Tsai *et al.* [18] proposed the first architecture for HEVC encoder. The proposed architecture was implemented for ultra HD specification in 28 nm technology. It dissipates 708 mW at 312 MHz for 8192×4320p at 30 fps encoding. Byun *et al.* [19] proposed an architecture for IME in 64 nm technology. It supports all the PU sizes and operating frequency is 250 MHz. Biswas *et al.* [20] proposed an architecture for IME. The proposed architecture is implemented in Virtex 6 FPGA and supports adaptive root search ME algorithm. In [21], ME algorithm and corresponding architecture are presented. For IME, a predictive search algorithm is proposed based on the statistical analysis; it selects the most probable direction for search. PU size dependent FME algorithm that adopts interpolation free FME for 0/1, the full search for depth 2 and skips FME for depth 3 are also presented. Architecture for proposed algorithm is implemented with a TSMC 90-nm CMOS process, which supports the real-time encoding of 4096 × 2048p @ 60 fps operated at 270 MHz with 778.7 K logic gates and 17.4 KB SRAM / on-chip memory.

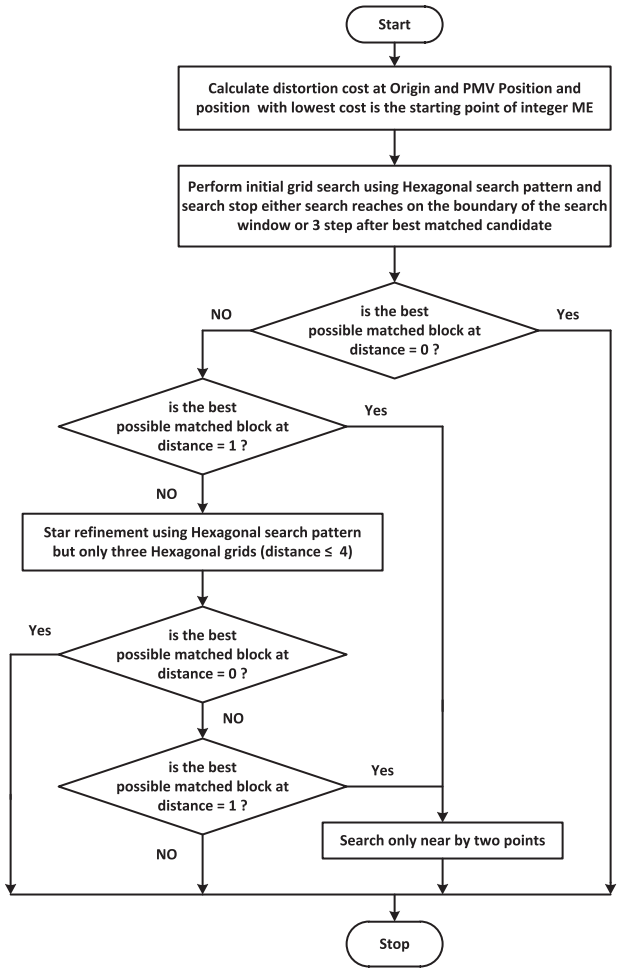

Fig. 4. Flow chart of the Computational Efficient IME Algorithm [22].

In [23], a fast ME algorithm based on rotating hexagonal search pattern for coarse search and an adaptive threshold for early termination is presented. Yang *et al.* [24] proposed a fast directional search ME algorithm based on the descent search and cross-search pattern. Both the search patterns are executed in parallel. A two-stage ME algorithm and architecture is presented in [25]. In the first stage, coarse ME is performed on truncated pixel and in the second stage, refinement ME is performed with full pixel resolution with an adaptive search pattern. The algorithm presented in [26], first the frames are filtered with the help of multiband-pass filter kernel and then constructs one bit-plane of the image. ME is performed on one bit-plane, which saves computational complexity in the calculation of the SAD. Ismail *et al.* [17] proposed a computationally efficient accurate skipping model to speed up any block-based ME algorithm. The reduction in computational complexity for the ME process has been achieved in four phases namely: - initial search center (ISC), dynamic early stop search termination (DESST), dynamic padding window size (DPWS) techniques and dynamic internal and external stop search technique.

To the best of our knowledge, very few works so far has been reported to reduce the complexity of ME involved in HEVC. Moreover, the ME algorithms and architectures developed for HEVC are power hungry and are not suitable for portable devices. In this paper, we proposed a power-efficient

IME algorithm and architecture which can be employed in portable devices.

In [22], we proposed IME algorithm which uses a hexagonal search pattern with a fixed number of the search points at each grid (except the first grid). We labelled it as hexagonal grid search (HGS) algorithm. This algorithm utilizes pixel truncation, sub-sampling and adaptive search range techniques for reduction of computational complexity. Pixel truncation is one of the most famous techniques to reduce the hardware cost. In [22], we found that truncating up to four LSBs of the pixels results in a 1% increment in bit-rate and 0.01 dB decrement of average PSNR. However, reducing the pixel size of four bits saves 50% hardware cost in the calculation of the SAD and SATD units involved in ME. In natural video sequences, the correlation between successive pixels is very high. So, sub-sampling the picture during the calculation of SAD does not affect the quality too much as presented in [16]. In [22], experimentally we found that the best matched PBs found at depth 0, 1, 2 and 3 are 55.52, 23.96, 14.78 and 5.74% respectively and total 95% of the best matched PBs is of size $2N \times 2N$, $2N \times N$ and $N \times 2N$ only. We also performed experiments for MVs behaviour and found that 48% of the MVs are concentrated at origin (0,0) of the search window and 93, 96.1, 97.3 and 99.31% of MVs lies in ($\pm 8, \pm 8$), ($\pm 12, \pm 12$), ($\pm 16, \pm 16$) and ($\pm 32, \pm 32$) ranges of search window respectively. From the above observations, it is clear that depth 3 is used only 5.74% and most of the MVs lies nearer to the origin. That means only for the complex section of the video depth 3 and the larger search window is required. So, instead of using a simple search window we can use adaptive search window, i.e., the small search area for larger PBs and the larger area for small PBs. We have also noted that if the depth of previously encoding CU is 0, the chance of the current CU partitioned up to depth 3 is only 0.9%. Therefore, based on the depth of the previously encoded CU, we can skip "depth 3" partitioning of the current CU. Based on the experimental results; we have chosen search range 2, 4, 8, and 16 for PUs of size 64×64, 32×32, 16×16 and 8×8 respectively. The flow of the proposed energy efficient IME algorithm (Fig. 4) is explained by the following steps:

- *Step 1 (Motion Vector Prediction):* - This step calculates the distortion at origin and PMV position to choose the best one as the seed.
- *Step 2 (Initial Grid Search):* - Unlike the TZS algorithm, the proposed algorithm uses only hexagonal search pattern with seven search points for all distances > 1. Depending on the distance of the best-matched candidate, the search process will either be terminated or perform one of the following steps namely star refinement or two-point search.
- *Step 3 (Star Refinement):* - This step is performed only when the distance of the best-matched candidate in the previous step is greater than one. This step also uses hexagonal search pattern like TZS but the major difference in the proposed algorithm is that this algorithm tests only three hexagonal grids (distances ≤ 4). Again, depending on the distance of the best-matched candidate block, the search process will either stop or go to two point search (Step 4).
- *Step 4 (Two Point Search):* - This step is executed only when the distance of the best-matched block in the previous step is equal to one. It executes only once and gives the final best possible matched candidate in the search window.

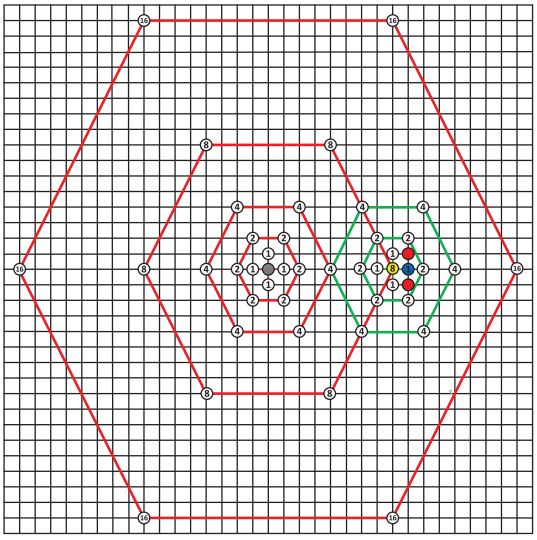

Fig. 5. Illustration of the IME Algorithm [22].

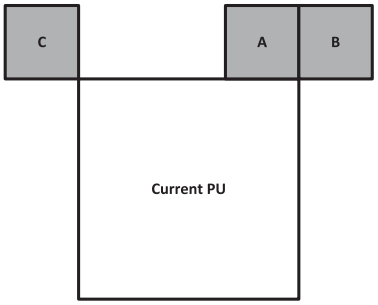

Fig. 6. Motion Vector Prediction Technique Used [27].

The illustration of the HGS ME algorithm is presented in Fig. 5. Traditionally, the motion vector prediction (MVP) in HGS algorithm uses left, top left, top and top right blocks. For reducing the power we have employed pipelining in the proposed architecture. Thus, for the MVP of the current block, the left side block is not available. Therefore, we have to modify the HGS algorithm accordingly. The proposed integer ME algorithm does not support left side blocks in MVP for the starting point. In the MVP scheme, top, top-left and top-right blocks are used as shown in Fig. 6 [27]. We labeled the proposed algorithm as modified hexagonal grid search (MHGS). Expect the change in MVP the flow of the MHGS is same as given in [22]. The effect on BD-PSNR and BD-rate due to change in MVP is discussed in the result section.

III. Proposed Architecture

The block diagram of the proposed IME architecture is shown in Fig. 7. It contains mainly six blocks: 1) SAD unit, 2) rate-distortion unit, 3) memory control unit, 4) memory, 5) comparator and 6) MV decision unit. In the proposed architecture a total six SAD units each of size 32×32 are used. Each of the SAD units is capable of calculating the SAD value of

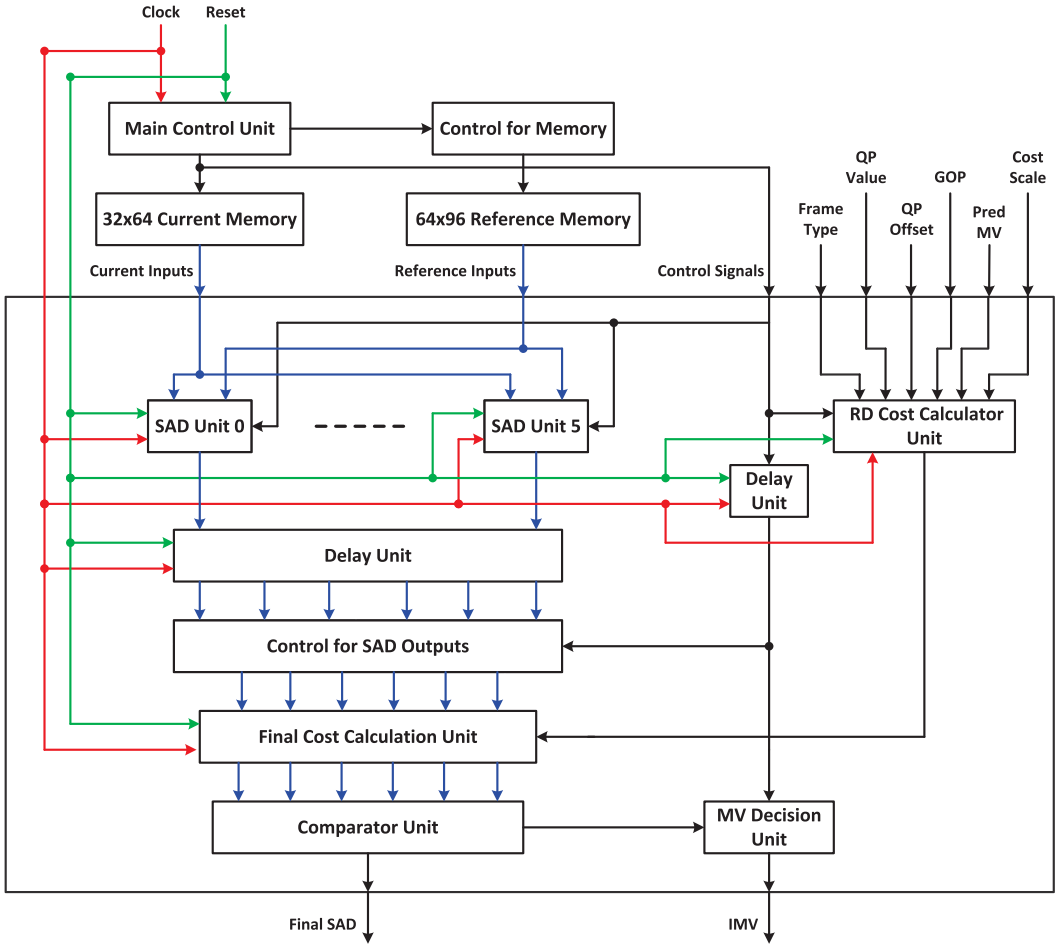

Fig. 7. Block Diagram of the Proposed Integer Motion Estimation Architecture.

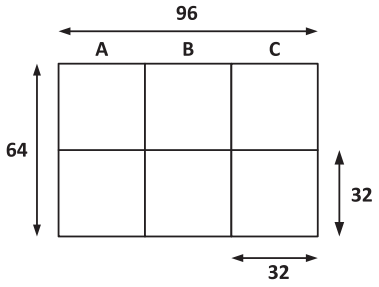

Fig. 8. Reference Memory for IME.

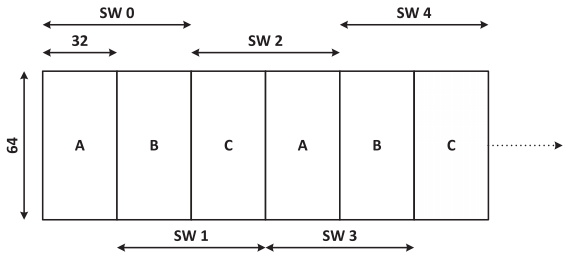

Fig. 9. Flow of the Reference Memory.

the 4×4 to 32×32 sub-blocks in two clock cycles. For reducing the hardware complexity, we used 4:2 compressor circuit, carry increment adder, pixel truncation and sub-sampling.

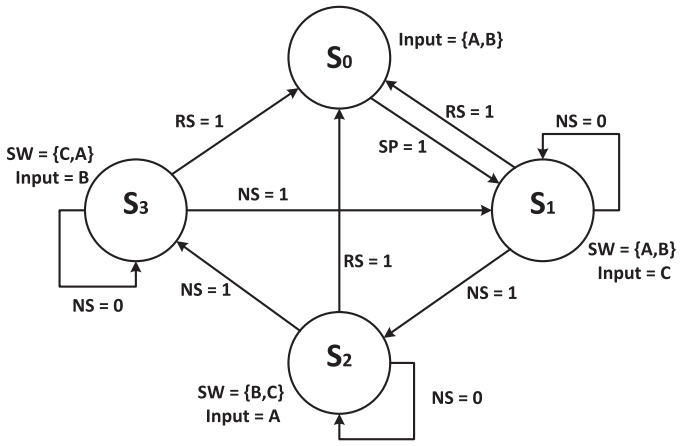

Fig. 10. Proposed Control Unit for Reference Memory.

For storing the reference and current block pixels, we used two memories of size $64 \times 96 \times 8$ and $32 \times 32 \times 8 \times 2$ bits respectively. Reference memory contains 6, 32×32 pixel blocks and these blocks are partitioned into three-part as shown in Fig. 8. Each of the partition takes 256 clock cycles to be filled up. Thus, for the first block, it takes 512 clock cycles. The flow of the reference memory is shown in Fig. 9. The first PU block search window *SW0* (formed by block *A* and *B*) is used for

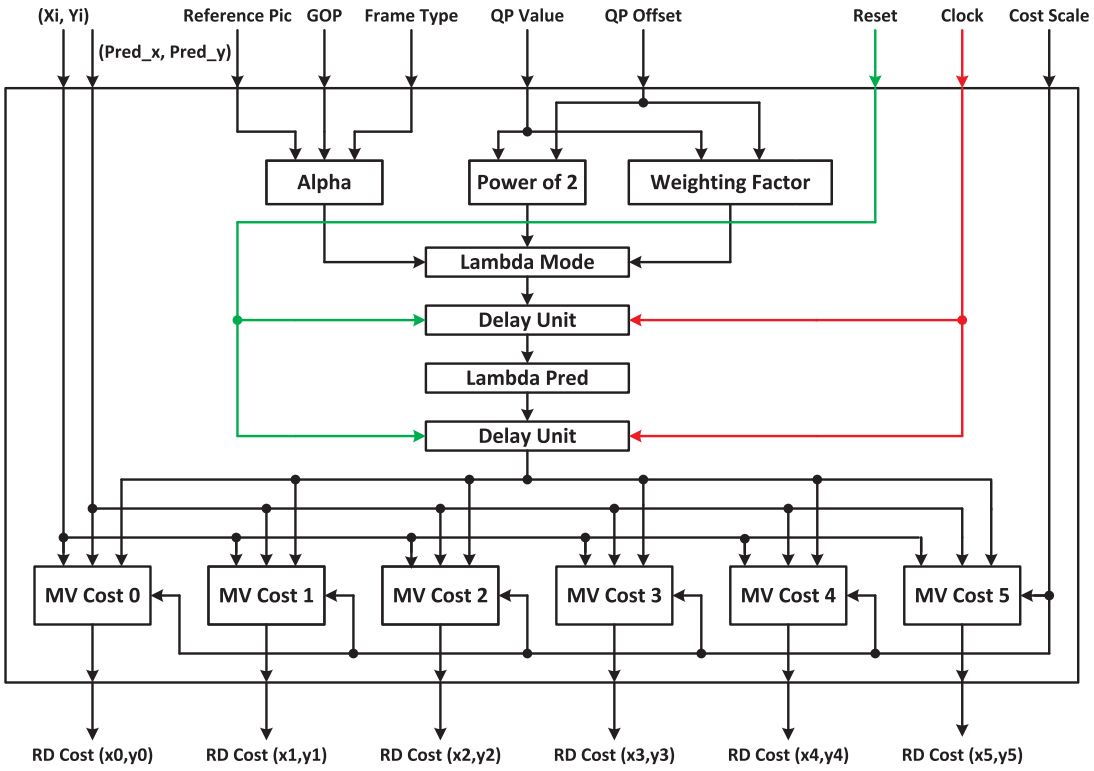

Fig. 11. Rate Distortion Cost Unit.

reference memory, whereas block C is used to store the reference pixels of next PU. For 2nd PU, reference memory which is formed by blocks B and C is used. A similar procedure is followed for the remaining PUs. The control unit of memory is given in Fig. 10. In Fig. 10, *RS*, *SP*, and *NS* are reset state, starting position and next stage signals respectively. There are total four stages: - S_0, S_1, S_2 and S_3. The stage S_0 is the starting or first stage. In this stage, data is stored in reference memory part A and B as shown in Fig. 8. After 512 clock cycles, the second stage, i.e., S_1 is activated. In this stage, data is stored in part C of reference memory and part A and B are used as a search window for first PU. When the S_1 stage output is 1 the control moves to stage S_2. In this stage, data is stored in memory part A and memory part B and C is used as a search window. The same procedure is repeated for the next stages.

In HEVC, for prediction parameter decision, the following expression is used

$$J_{pred,SAD} = SAD + \lambda_{pred} \times R_{pred} \tag{5}$$

The value of the Lagrangian constant (λ) and bit cost (R) are calculated using the rate-distortion (RD) cost unit. The block diagram of the RD cost unit is shown in Fig. 11. The proposed RD cost unit contains six MV cost units that work parallel and calculates the cost for six different positions. The proposed RD cost unit depends on a QP, λ, group of picture (GOP), frame type and the weighting factor. The value of λ is calculated using the relation provided in the standard (i.e., $\lambda = \alpha \times W_k \times 2^{((QP-12)/3)}$). Lookup tables are used to implement a weighting factor (W_k), α, the power of 2 and λ_{pred}. It is pipeline two times as shown in Fig. 11.

The proposed architecture for SAD tree unit and 2×2 SAD unit are presented in Fig. 12 and Fig. 13 respectively. In SAD

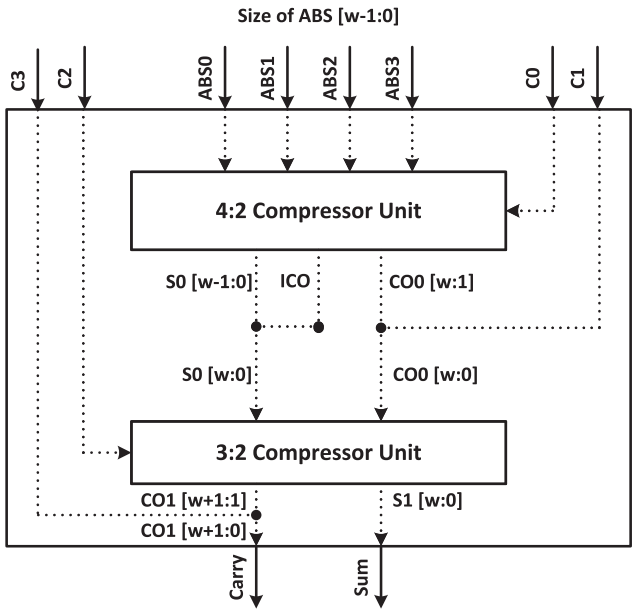

Fig. 12. Proposed SAD Tree Unit.

tree unit, we used 4:2 and 3:2 compressor, which requires less number of gates and smaller critical path as compared to the normal addition. The SAD tree takes a total 8 inputs (4 sums + 4 carries) and generate two outputs namely sum and carry. The proposed 2×2 SAD unit consists absolute difference calculation unit (ABSU), SAD tree unit and 6-bit carry-lookahead adder. The ABSU calculates the absolute difference between current and reference pixels and output of this unit is sent to the SAD tree unit. The 6-bit carry-lookahead adder is used for

TABLE I
BD-PSNR, BD-BITRATE AND ASP COMPARISON WITHOUT USING LEFT SIDE BLOCK IN PREDICTION

Video Sequences	Algorithm	Profile	BD-Bitrate	BD-PSNR	△ASP(%)
BQMall (832x480_60fps)	MTZS	Low Delay	0.2322	-0.0202	2.6661
		Random Access	0.5609	-0.0242	3.2601
	MHGS	Low Delay	2.3662	-0.0551	-58.1684
		Random Access	2.5432	-0.1383	-63.9470
Johnny (1280x720_60)	MTZS	Low Delay	0.1013	-0.0084	1.1076
		Random Access	0.1559	-0.0423	1.5208
	MHGS	Low Delay	1.3743	-0.0013	-32.4039
		Random Access	1.8491	-0.0326	-35.0114
crowd_run (1080p50fps)	MTZS	Low Delay	0.3616	-0.0013	12.7612
		Random Access	0.5160	-0.0347	13.7067
	MHGS	Low Delay	1.1448	-0.0211	-72.5020
		Random Access	1.3327	-0.2250	-73.3427
Traffic (2560x1600_30fps)	MTZS	Low Delay	0.3011	-0.0034	4.1284
		Random Access	0.3886	-0.0092	4.8907
	MHGS	Low Delay	2.3254	-0.0248	-47.0048
		Random Access	2.5256	-0.0410	-48.1423

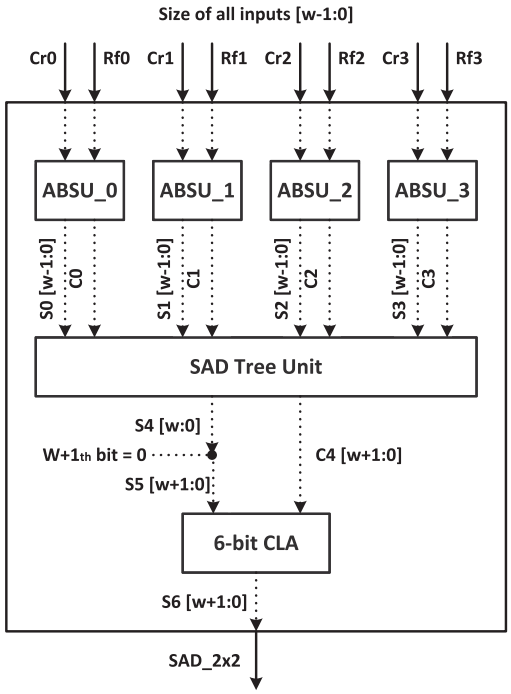

Fig. 13. Proposed 2 × 2 SAD Unit.

the final sum of 2 × 2 pixels. It is a combination of two 3-bit carry-lookahead adders.

IV. SIMULATION RESULTS FOR PROPOSED ALGORITHM AND ARCHITECTURE

The reference software HM is used with default configuration, i.e., the TZS algorithm is used in IME search with a search range of ±64 and CU size is 64. The proposed MHGS ME algorithm is integrated into reference software HM and simulated with the *encoder_lowdelay_P_main* and *encoder_randomaccess_main* profile under different QP values of 22, 27, 32 and 37. The effect on BD-PSNR, BD-bit rate, and average search points (ASP) are presented in Table I. In Table I, MTZS is the TZS without left side MVP blocks and MHGS is the HGS without left side MVP blocks. The degradation in BD-bit rate and BD-PSNR for MTZS is small with little increment in search points by 5.51% in comparison to reference software HM. The proposed MHGS ME algorithm requires an average of 53.82% fewer search points in comparison to reference software HM with a little degradation in BD-PSNR and average 2% increment in BD-bit rate. In a video like *crowd_run 1080p @ 50fps*, the number of search points are saved more than 73% with around 1.3% increment in bit-rate and 0.22 dB decrement in PSNR. So, we can say that for a video like *crowd_run 1080p @ 50fps*, around 3/4th computational power of ME process is saved. As shown in Table I the proposed algorithm is suitable for all types of video sequences.

The rate-distortion results for video sequences *BasketballPass*_416 × 240@50 and *KristenAndSara*_1280 × 720@60 using *encoder_lowdelay_P_main* profile under different QP values of 22, 27, 32 and 37 is shown in Fig. 14. The BD-PSNR [29] loss as compared to the reference software HM in *BasketballPass*_416 × 240@50 and *KristenAndSara*_1280 × 720@60 video sequences are 0.0621 dB and 0.0610 dB respectively. Fig. 15 shows the results for *encoder_randomaccess_main* profile under different QP values of 22, 27, 32 and 37. The BD-PSNR losses for *encoder_randomaccess_main* profile are 0.0646 dB and 0.1040 dB for *BasketballPass*_416 × 240@50 and *KristenAndSara*_1280×720@60 video sequences respectively. It is clear from Fig. 14 and 15 that, the target bit-rate increases the loss in the BD-PSNR decreases. The proposed HGS IME algorithm requires a fewer number of search points and loss in BD-PSNR is also less for *encoder_lowdelay_P_main* and *encoder_randomaccess_main* profiles.

Simulations have been carried out using Synopsys Design Compiler and Cadence Innovus software, to obtain the performance of the proposed architecture. The proposed architecture is implemented using standard 90 nm technology and the results are presented in Table II. In Table II, the proposed architecture corresponding to the MHGS ME algorithm is compared with other state of the art architectures of ME. For reducing the computational power, we used sub-sampling, pixel truncation, adaptive search range and removing the

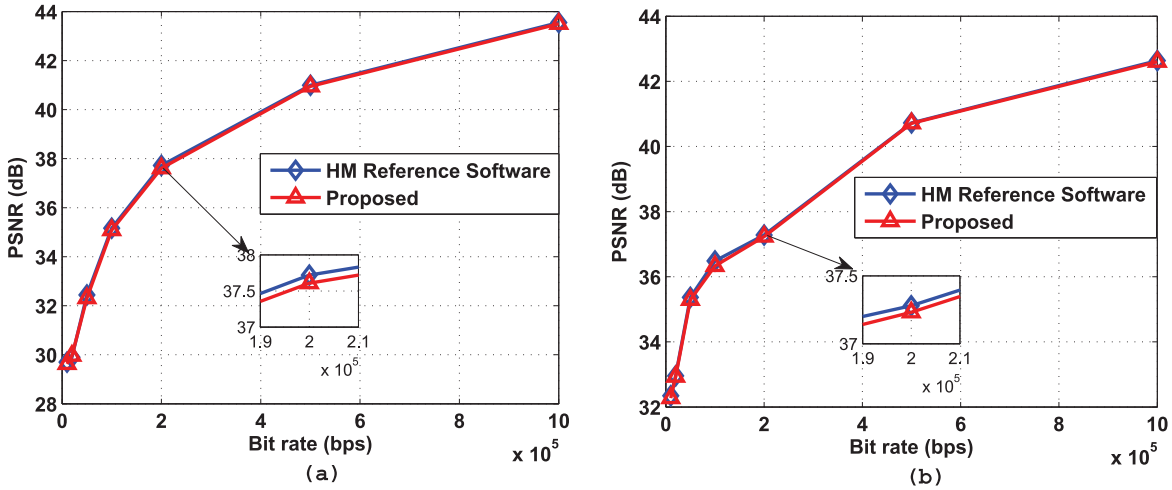

Fig. 14. Rate-distortion results for video sequence (a) *BasketballPass*_416×240@50 and (b) *KristenAndSara*_1280×720@60 using *encoder_lowdelay_P_main* profile under different QP values of 22, 27, 32 and 37.

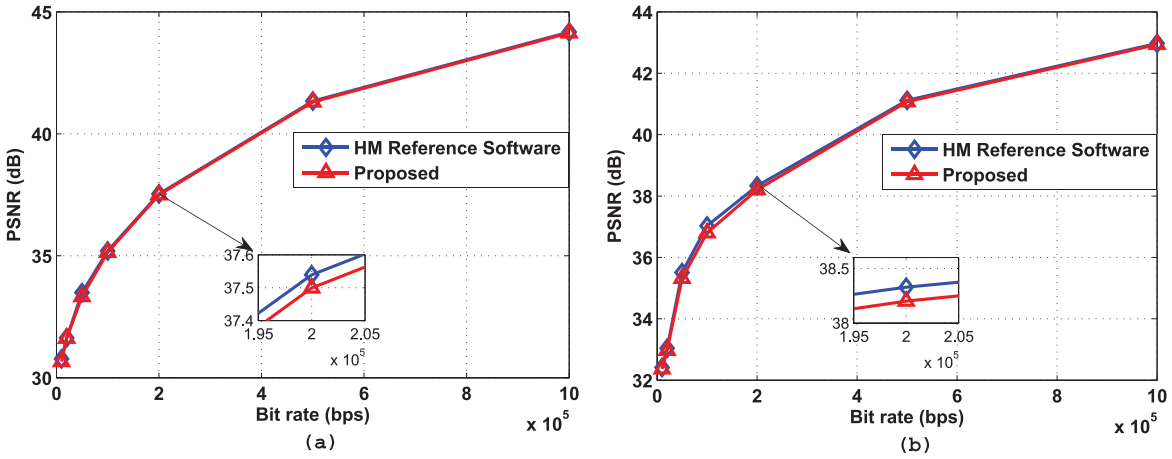

Fig. 15. Rate-distortion results for video sequence (a) *BasketballPass*_416 × 240@50 and (b) *KristenAndSara*_1280 × 720@60 using *encoder_randomaccess_main* profile under different QP values of 22, 27, 32 and 37.

TABLE II
COMPARISON OF THE PROPOSED ARCHITECTURE WITH STATE OF THE ART ARCHITECTURES

	ME2014[16]	ME2013[18]	MEHEVC13[28]	MEHEVC15[21]	Proposed
Technology	40 nm CMOS	TMSC 28 nm	65 nm Low-Power CMOS	90 nm	90 nm
Standard	H.264/AVC	HEVC	HEVC	HEVC	HEVC
Resolution	7680 × 4320 @ 48 fps 3840 × 2160 @ 120 fps	8192 × 4320 @ 30 fps	3840 × 2160 @ 30 fps	4096 × 2048 @ 60 fps	3840 × 2160 @ 30 fps
Search Range	(($\pm$211, $\pm$106) for P ($\pm$107, $\pm$56) for B)	($\pm$512, $\pm$128)	($\pm$64, $\pm$64)	($\pm$64, $\pm$64)	Adaptive
Algorithm	FS + 5T12S	Inter + intra	FS + TSS	Modified TZS	Modified HGS
Memory	552 KB	7.14 MB	0.68 MB	17.4 KB	8.192 KB
Throughput	1.59 Gpixel/s	1062 Mpixels/s	248.8 Mpixel/s	503 Mpixel/s	248.8 Mpixel/s
Block Size	16 × 16 - 8 × 8	64 × 64 - 16 × 16	64 × 64 to 8 × 8	64 × 64 to 8 × 8	32 × 32 to 8 × 8
Area	15.52 mm^2	25 mm^2	3965 K gates	778.7 K gates	4.5193 mm^2/1441 K gates
Power	622 mW	708 mW	NA	NA	151.7619 mW
Frequency	210 MHz	312 MHz	200 MHz	270 MHz	250 MHz
Sub-sampling	4 × 4 : 1	NA	Not Used	Not Used	2 × 2 : 1
Pixel Truncation	6-bit/pixel	NA	NA	NA	4-bit/pixel
Cost Function	Not Used	NA	Not Used	Not Used	Yes

left side block from the motion vector prediction technique. Simulations result shows that the proposed architecture is able to process 3840 × 2160 @ 30 fps video sequences with 4.5193 mm^2 of the area and 8.192 KB of SRAM. The operating frequency of the proposed architecture is obtained as 250 MHz with 151.7619 mW of power. As compared to other state of the art ME architectures, the proposed architecture can be implemented using lesser power, SRAM and area.

V. CONCLUSION

In this paper, we proposed a low power algorithm and architecture of IME which is suitable for portable devices. For

reducing the hardware complexity, we used pixel truncation, sub-sampling, data reuse, adaptive search area, removing some of the PU sizes and fewer search points ME algorithm. In the proposed algorithm, left side blocks are not used in MVP due to the pipeline. Simulation result shows that MHGS ME algorithm required 53.82% fewer search points as compared to the number of search points required using reference software HM results with a little degradation in PSNR and small increment in bit-rate. The proposed architecture is simulated and synthesized using standard 90 nm technology. The proposed IME architecture can process 3840×2160 @ 30 fps video sequences with 4.5193 mm^2 of the area and 8.192 KB of SRAM. The ME architecture consumes 151.7619 mW of power when it operates on 250 MHz.

ACKNOWLEDGMENT

The authors would like to thank Department of Electronics and Information Technology (DeitY), Government of India, for providing the software resources under the Special Manpower Development Programme (Phase III and C2SD).

REFERENCES

[1] G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1649–1668, Dec. 2012.

[2] M. Chavarrias, F. Pescador, M. J. Garrido, E. Juárez, and M. Raulet, "A DSP-based HEVC decoder implementation using an actor language dataflow model," *IEEE Trans. Consum. Electron.*, vol. 59, no. 4, pp. 839–847, Nov. 2013.

[3] F. Pescador, M. J. Garrido, E. Juarez, and C. Sanz, "On an implementation of HEVC video decoders with DSP technology," in *Proc. IEEE Int. Conf. Consum. Electron. (ICCE)*, Jan. 2013, pp. 121–122.

[4] J. Vanne, M. Viitanen, and T. D. Hämäläinen, "Efficient mode decision schemes for HEVC inter prediction," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 9, pp. 1579–1593, Sep. 2014.

[5] Y. Ye, Y. He, and X. Xiu, "Manipulating ultra-high definition video traffic," *IEEE Multimedia*, vol. 22, no. 3, pp. 73–81, Jul. 2015.

[6] J. Vanne, M. Viitanen, T. D. Hamalainen, and A. Hallapuro, "Comparative rate-distortion-complexity analysis of HEVC and AVC video codecs," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1885–1898, Dec. 2012.

[7] A. Lee, D. Jun, and J. S. Choi, "Fast motion estimation using priority-based inter-prediction mode decision method in high efficiency video coding," *J. Real Time Image Process.*, vol. 12, no. 2, pp. 433–441, Aug. 2016, doi: 10.1007/s11554-015-0493-7.

[8] I. Richardson and J. Wiley, *Video Codec Design: Developing Image and Video Compression Systems*. Chichester, U.K.: Wiley, 2002.

[9] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 560–576, Jul. 2003.

[10] M. B. V. Sze and G. J. Sullivan, Eds., *High Efficiency Video Coding (HEVC) Algorithms and Architectures*. Cham, Switzerland: Springer, 2014.

[11] T. Koga *et al.*, "Motion-compensated interframe coding for video conferencing," in *Proc. NTC*, 1981, p. 9.

[12] S. Zhu and K.-K. Ma, "A new diamond search algorithm for fast block-matching motion estimation," *IEEE Trans. Image Process.*, vol. 9, no. 2, pp. 287–290, Feb. 2000.

[13] C. Zhu, X. Lin, and L.-P. Chau, "Hexagon-based search pattern for fast block motion estimation," *IEEE Trans. Circuits. Syst. Video Technol.*, vol. 12, no. 5, pp. 349–355, May 2002.

[14] L.-K. Liu and E. Feig, "A block-based gradient descent search algorithm for block motion estimation in video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, no. 4, pp. 419–422, Aug. 1996.

[15] Z. Liu *et al.*, "HDTV1080p H.264/AVC encoder chip design and performance analysis," *IEEE J. Solid-State Circuits*, vol. 44, no. 2, pp. 594–608, Feb. 2009.

[16] D. Zhou, J. Zhou, G. He, and S. Goto, "A 1.59 Gpixel/s motion estimation processor with -211 to +211 search range for UHDTV video encoder," *IEEE J. Solid-State Circuits*, vol. 49, no. 4, pp. 827–837, Apr. 2014.

[17] Y. Ismail, J. B. McNeely, M. Shaaban, H. Mahmoud, and M. A. Bayoumi, "Fast motion estimation system using dynamic models for H.264/AVC video coding," *IEEE Trans. Circuit. Syst. Video Technol.*, vol. 22, no. 1, pp. 28–42, Jan. 2012.

[18] S.-F. Tsai *et al.*, "A 1062Mpixels/s 81924320p high efficiency video coding (H.265) encoder chip," in *Proc. Symp. VLSI Circuits (VLSIC)*, Jun. 2013, pp. C188–C189.

[19] J. Byun, Y. Jung, and J. Kim, "Design of integer motion estimator of HEVC for asymmetric motion-partitioning mode and 4K-UHD," *Electron. Lett.*, vol. 49, no. 18, pp. 1142–1143, Aug. 2013.

[20] B. Biswas, R. Mukherjee, and I. Chakrabarti, "Efficient architecture of adaptive rood pattern search technique for fast motion estimation," *Microprocess. Microsyst.*, vol. 39, no. 3, pp. 200–209, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0141933115000149

[21] S.-Y. Jou, S.-J. Chang, and T.-S. Chang, "Fast motion estimation algorithm and design for real time QFHD high efficiency video coding," *IEEE Trans. Circuit. Syst. Video Technol.*, vol. 25, no. 9, pp. 1533–1544, Sep. 2015.

[22] K. Singh and S. R. Ahamed, "Computationally efficient motion estimation algorithm for HEVC," *J. Signal Process. Syst.*, Dec. 2017. [Online]. Available: https://doi.org/10.1007/s11265-017-1321-z

[23] N. Purnachand, L. Alves, and A. Navarro, "Fast motion estimation algorithm for HEVC," in *Proc. IEEE Int. Conf. Consum. Electron Berlin (ICCE-Berlin)*, Sep. 2012, pp. 34–37.

[24] S.-H. Yang, J.-Z. Jiang, and H.-J. Yang, "Fast motion estimation for HEVC with directional search," *Electron. Lett.*, vol. 50, no. 9, pp. 673–675, Apr. 2014.

[25] S. K. Chatterjee and I. Chakrabarti, "Power efficient motion estimation algorithm and architecture based on pixel truncation," *IEEE Trans. Consum. Electron.*, vol. 57, no. 4, pp. 1782–1790, Nov. 2011.

[26] S. Lee, G. Jeon, and J. Jeong, "Fast motion estimation based on enhanced constrained one-bit transform," *Electron. Lett.*, vol. 50, no. 10, pp. 746–748, May 2014.

[27] T.-C. Chen *et al.*, "Analysis and architecture design of an HDTV720p 30 frames/s H.264/AVC encoder," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 16, no. 6, pp. 673–688, Jun. 2006.

[28] M. E. Sinangil, V. Sze, M. Zhou, and A. P. Chandrakasan, "Cost and coding efficient motion estimation design considerations for high efficiency video coding (HEVC) standard," *IEEE J. Sel. Topics Signal Process.*, vol. 7, no. 6, pp. 1017–1028, Dec. 2013.

[29] G. Bjontegaard, *Calculation of Average PSNR Differences Between RD Curves*, document VCEG-M33, 13th VCEG Meeting, VCEG, Austin, TX, USA, Apr. 2001.

Karam Singh received the B.Tech. degree in electronics and communication engineering from Maharshi Dayanand University, Rohtak, India, in 2010. He is currently pursuing the Ph.D. degree with the Department of Electronics and Electrical Engineering, Indian Institute of Technology Guwahati, Guwahati, India. His research area includes very-large-scale integration architecture and algorithm for video processing and digital very-large-scale integration design for low power applications.

Shaik Rafi Ahamed received the B.Tech. and M.Tech. degrees in electronics and communication engineering from Sri Venkateswara University, Tirupati, India, in 1991 and 1993, respectively, and the Ph.D. degree from the Indian Institute of Technology, Khargpur, in 2008. He is currently a Professor with the Department of Electronics and Electrical Engineering, Indian Institute of Technology Guwahati, Guwahati, India. From 1993 to 1995, he was a Faculty Member with the Deccan College of Engineering and Technology, Hyderabad, India, and from 1995 to 2003 with Bapatla Engineering College, Bapatla, India. His teaching and research interests are in digital and adaptive signal processing, biomedical signal processing, and very-large-scale integration signal processing.