LOW ENERGY MOTION ESTIMATION VIA SELECTIVE APROXIMATIONS

Yunus Emre and Chaitali Chakrabarti
School of Electrical, Computer and Energy Engineering
Arizona State University, Tempe, AZ 85287
{yemre,chaitali}@asu.edu

***Abstract*—This paper presents a novel sum of absolute difference (SAD) scheme that significantly reduces the energy consumption of the motion estimation kernel in video coders. The proposed scheme exploits the facts that most of the absolute difference (AD) calculations result in small values, and most of the large AD values do not contribute to the SAD values of the blocks that are selected. Thus the large AD values can be approximated, resulting in lower critical path delay and lower energy consumption of the SAD unit. In addition, the proposed scheme truncates one lower order bit and variants of this scheme implement sub-sampling to further reduce the energy consumption. Performance of the proposed technique is evaluated using the H.264 video encoding framework. Simulation results show 37.5% reduction in energy consumption at nominal voltage and 68% energy reduction for iso-throughput compared to the conventional implementation with only 0.08% drop in PSNR and 1.8% increase in compressed data rate. With an additional ¼ sub-sampling, the proposed scheme achieves 90% reduction in energy consumption with 0.6% reduction in PSNR and 6.5% increase in compressed data rate.**

***Index Terms*—Motion Estimation, Error Resiliency, Low Energy Implementation**

I. INTRODUCTION

The number of portable multimedia devices have dramatically increased in the last two decades. Many of these devices support video codecs which consume significant amount of energy. To increase the lifetime of these devices, the energy consumption has to be reduced, and yet the quality of service has to be maintained. In this paper we focus on reducing the energy consumption of motion estimation with very small degradation in performance. We focus on motion estimation since it is the most energy greedy part of a video codec and consumes approximately 50% to 70% of the overall energy of codecs such as H.263, MPEG-4 and H.264 [1]-[3].

Motion estimation (ME) is used to remove temporal redundancy across consecutive frames. Each block in the current frame is compared with candidate blocks from previous and future frames, and the block with the best match is selected. One of the widely used criteria to find the best match is the sum of absolute difference (SAD) in which absolute differences (AD) of pixel values in the current and reference blocks are accumulated over all pixel positions in a block. Of the different strategies to find the candidate reference blocks, three-step search (TSS) [4] and diamond search (DS) [5] are popular and have been used in this paper.

Various architectures have been proposed to improve throughput and reduce energy consumption and latency of ME computation. While some truncate the lower order bits [1], [3] and/or subsample the block [1], [9] to reduce the hardware complexity, others minimize the number of memory accesses [7]. In [8], a motion estimator based on significance driven computation has been proposed, where only significant computations are operated in error free manner and errors due to voltage overscaling in other computations are tolerated. In [9], voltage overscaling is employed to reduce power consumption of the ME part and error compensation is done using a sub-sampled version of the original computation. This system suffers from approximately 25% extra computations and result in 2% reduction in PSNR when ¼ sub-sampling is used in the replica block.

In this paper, we present a novel SAD computation scheme that achieves low energy consumption by judiciously approximating the computations in the SAD unit. The proposed scheme uses the statistics of AD and SAD computations to approximate the computations. Specifically, it exploits the fact that: i) most of the AD values are small, due to locality of current and reference blocks, and that ii) most of the large AD values are for blocks that are likely not to be selected, and thus these values can be approximated. In the proposed scheme, large AD values are detected using special logic and the corresponding SAD value updated with a correction factor. In addition, one lower order bit is truncated to further reduce the computational load. We also propose variants of this scheme based on sub-sampling which reduce the energy consumption even more. The corresponding architecture has a lower critical path delay compared to the baseline SAD architecture and achieves 37.5% energy reduction at nominal voltage. For iso-throughput, i.e., when the throughput is the same as the baseline architecture, it achieves 68% energy reduction while incurring 1.8% increase in compressed data rate and 0.08% reduction in PSNR. If larger increase in compressed data rate is tolerable, the proposed scheme with ½ sub-sampling has approximately 90% reduction in energy consumption with 6.5% increase in compressed data rate and only 0.6% reduction in PSNR.

This work was funded in part by NSF CSR0910699.

The rest of the paper is organized as follows. In Section 2 we briefly describe ME followed by the simulation set-up. Section 3 describes the low power techniques used to reduce complexity of SAD computation followed by the proposed algorithm. The corresponding SAD architecture and its evaluation are given in Section 4. Simulation results for quality performance of the proposed scheme as well as energy-quality comparison with other candidate schemes are presented in Section 5. Section 6 concludes the paper.

II. BACKGROUND

A. Motion Estimation

In a video codec, frames are partitioned into smaller non-overlapping blocks typically of size 8x8 or 16x16. For each block in the current frame, we find a block in the reference frame with which it has the best match within a search window. To find the best match, the sum of absolute difference (SAD) is widely used. Here the absolute difference (AD) of pixels corresponding to the same location in the reference and current blocks are computed and then summed over all the pixels in the block. SAD can be expressed as:

$$SAD(P_{cur}, P_{ref}) = \sum_{m=0}^{B-1}\sum_{n=0}^{B-1}\left|P_{cur}(m,n) - P_{ref}(m,n)\right| \qquad (1)$$

where the current block is of size BxB, and P_{cur} and P_{ref} refer to current and reference blocks, respectively. The block that achieves the lowest SAD value is the block that is selected for motion compensation.

There are several search strategies to find the candidate blocks in a search area. Full search, where all reference block positions are searched, is computationally intensive and thus rarely used. Instead, several heuristic algorithms with lower complexity but sub-optimal performance are used. A comprehensive list of search strategies has been described in [6]. We use two popular strategies, namely, three step search (TSS) [4] and diamond search (DS) [5] in this paper. The proposed techniques are essentially independent of the search algorithm and can be used in conjunction with other search methods.

B. Algorithm Evaluation Set-Up

To evaluate the performance of the proposed SAD computation method, we embed it into the ME module of the H.264 encoder[11]. We also implement the DCT module of [12] and CAVLC module of [13] to see the end to end performance. Our baseline scheme uses half pixel accuracy with 8x8 block size, 4 different configurations of quantizer Q=10, 20, 30 and 40, and the TSS and DS search strategies. We study 13 different video sequences with different degrees of motion. These include Claire, Carphone, Foreman, Football, Hall, Miss America, Salesman, Suzie, Bus (QCIF), Coastguard, Mobile, Soccer (CIF) [10]. In each case, we consider the first 60 frames.

We use change in peak to noise ratio ($\Delta PSNR$) and change in compressed data rate (ΔCR) to evaluate the performance of the modified scheme. If C is the original frame and R is the reconstructed frame, then for frame of size MxN, the PSNR is given by:

$$\text{PSNR} = 10\log\left(\frac{\max(\text{C}^2)}{\text{MSE}}\right), \qquad \text{where}$$

$$MSE = \frac{1}{MN}\sum_{i=0}^{N-1}\sum_{j=0}^{M-1}\left(C_{ij} - R_{ij}\right)^2$$

For the baseline scheme, PSNR ranges from 25 dB to 50 dB. $\Delta PSNR$ is defined as the percentage increase in the PSNR value of the modified scheme compared to the baseline scheme averaged over different values of Q.

Compressed data rate (CR) is defined as the size of the encoded data for 1 second of video and expressed in terms of bits per second (bps). For the baseline scheme, CR varies from 50 kbps to 800 kbps. ΔCR is defined as the percentage increase in compared to the baseline scheme. So both reduction in ΔCR and increase in $\Delta PSNR$ should be as small as possible.

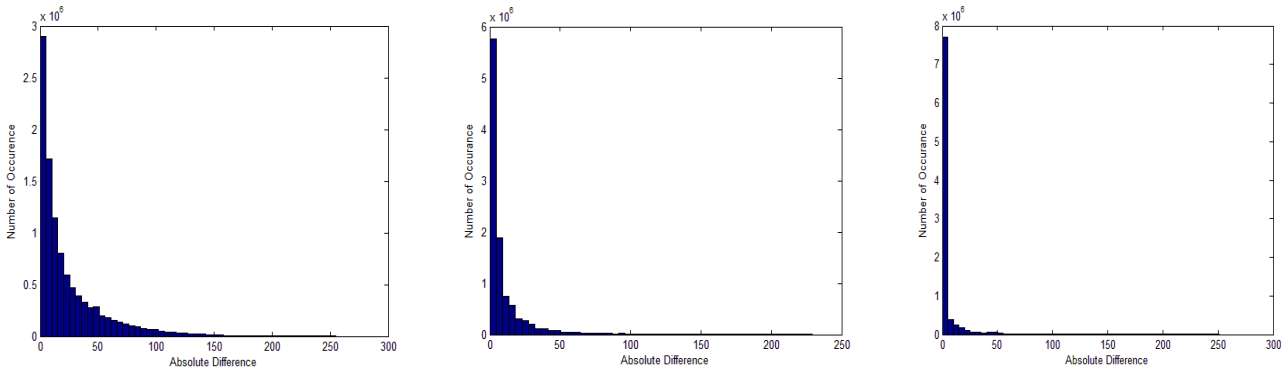
III. PROPOSED LOW POWER TECHNIQUES

In this section, we describe the proposed scheme based on clipping of higher order bit computation (HOC) and two supportive techniques that can be used in conjunction with the HOC scheme to reduce energy consumption.

A. High Order Clipped Computation

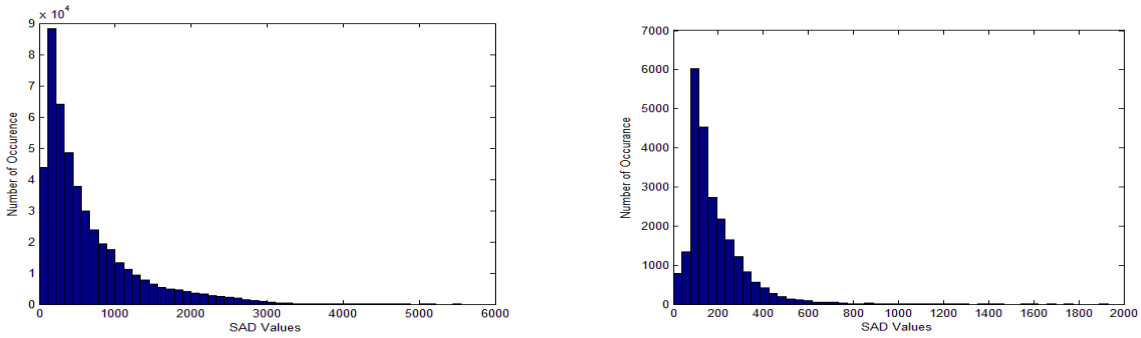
To motivate the HOC scheme, we first study the statistics of AD and SAD computations of motion blocks for video sequences with different amount of motion. Figure 1 illustrates the distribution of AD values for Football, Foreman and Claire video sequences for quantizer level $Q = 20$. Most of the AD calculations result in very small values due to locality of current and reference blocks. We see that, on average, 82% of the AD results of Football sequence is smaller than 32, which can be represented by 5-bits. This ratio increases up to 93% for Foreman and 97% for Claire video sequences. Tables I and II give the percentages of AD values that can be represented by 4 to 8 bits for Football and Foreman video sequences for different values of Q. We see that these ratios are not affected by Q. Similar results have been obtained for other test video sequences.

Our next observation is that most of the large AD values make the corresponding SAD values large, and at the end, the blocks with the large SAD values are not selected. Figure 2-a shows the distribution of SAD values of all blocks and Figure 2-b shows the distribution of SAD values of only the selected blocks for the Foreman video sequence. We see that SAD values of the selected blocks are not greater than a couple of hundred and large AD values could not have contributed to these SAD values. Table III lists the maximum, mean and standard deviation of SAD values of selected and unselected blocks for 4 different quantization levels. The SAD values of selected blocks have approximately 4 times smaller values for maximum, mean and variance. Thus, when the AD values are large, we do not need to update SAD value with the exact AD



(a) (b) (c)

Figure 1: Histogram plots of AD computations of a) Football, b) Foreman and c) Claire videosequences



(a) (b)

Figure 2: a) Histogram of SAD values of all blocks, b) Histogram of SAD values of selected block for Football video sequence

value. In fact, we can ignore the result and replace it with a correction factor which is an approximate of the actual AD value. This scheme results in significant energy reduction and little performance penalty as will be demonstrated in the following sections.

TABLE I
PERCENTAGE OF AD VALUES THAT CAN BE REPRESENTED BY 4 TO 8 BITS FOR FOOTBALL VIDEO SEQUENCE

# of bits	Q=10	Q=20	Q=30	Q=40
4	59%	60%	60%	60%
5	81%	82%	82%	82%
6	92%	92%	93%	93%
7	98%	99%	99%	99%
8	100%	100%	100%	100%

TABLE II
PERCENTAGE OF AD VALUES THAT CAN BE REPRESENTED BY 4 TO 8 BITS FOR FOREMAN VIDEO SEQUENCE

# of bits	Q=10	Q=20	Q=30	Q=40
4	85%	86%	86%	84%
5	93%	93%	94%	94%
6	98%	98%	98%	98%
7	~100%	~100%	~100%	~100%
8	100%	100%	100%	100%

TABLE III
STATISTICS OF SELECTED AND UNSELECTED VALUES FOR FOREMAN VIDEO SEQUENCE

	Max of selected blocks (max of all blocks)	Mean of selected blocks (mean of all blocks)	σ of selected blocks (σ of all blocks)
Q=10	1595 (4810)	161 (593)	126 (578)
Q=20	1576 (4810)	178 (579)	124 (578)
Q=30	1261 (4879)	233 (599)	125 (568)
Q=40	1242 (4866)	364 (624)	182 (553)

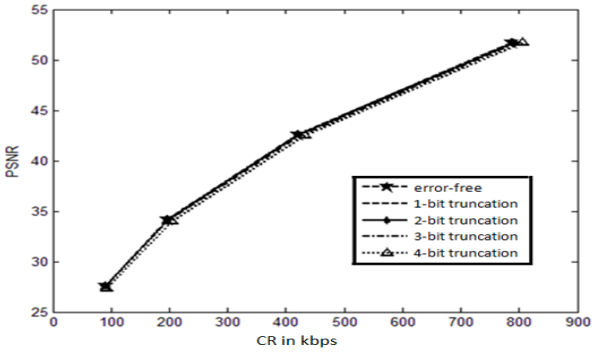
B. Supportive Technique 1: Low Order Bit Truncation

In order to reduce the computation burden of the SAD unit, bit truncation methods that remove low order bits have been proposed in [1], [3]. For instance, in [3], instead of using 8-bits, only 4 or 5 of the higher order bits are kept. Such a scheme results in degradation in performance especially when a large number of low order bits is dropped.

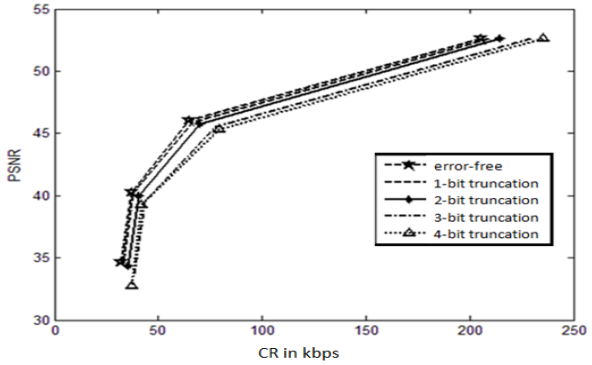
We study the performance degradation and increase in compressed data rate for low order bit (LOB) truncation using the 13 different video sequences. Figure 3 illustrates the PSNR and bit rate performance of bit-truncation techniques for Football and Claire video sequences. For Football video, maximum increase in the size of compressed data rate is 3% for 4-bit truncation at 42dB PSNR. The Claire video sequence shows larger degradation. For instance, 4-bit and 3-bit

truncations increase the compressed data rate rate by 21% and 18%, respectively, at 40 dB PSNR.

Figure 4 illustrates the average degradation over 13 video sequences for LOB truncation ranging from 1-bit to 4-bits. The degradation is similar for diamond (DS) and three step search (TSS) strategies. 1-bit truncation generates less than 0.8% increase in CR and 0.05% reduction in PSNR performance. For truncation of higher number of bits, the performance degradation increases significantly, whereas the reduction in complexity is only a linear function of the truncation size [14]. Based on this analysis, we choose only 1-bit LOB truncation in the proposed scheme. The overhead of truncating larger number of bits is further elaborated in Section 5.



(a)



(b)

Figure 3: Performance of bit truncation for a) Football and b) Claire sequences

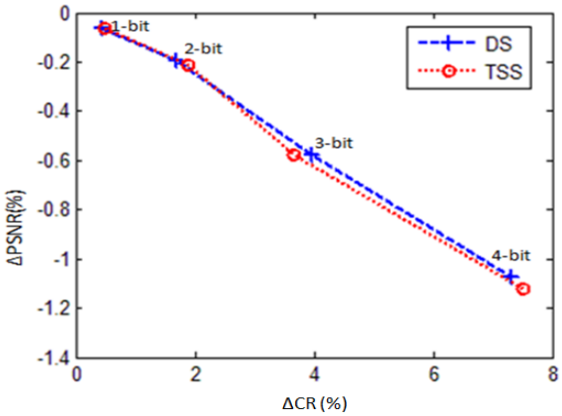


Figure 4: Average performance loss due to bit truncation using DS and TSS

C. *Supportive Technique 2: Sub-sampling*

Sub-sampling schemes have been also studied in the context of motion estimation to reduce number of AD computations per SAD calculation [1], [9]. These result in higher performance degradation but have significantly lower energy consumption. Two sub-sampling schemes are investigated here: s=2, s=4, where 1/s is the sub-sampling ratio. Based on our simulations, we choose uniform sampling for both the sub-sampling schemes. The patterns for ½ and ¼ sub-sampling schemes for 8x8 blocks are illustrated in Figure 5. These schemes reduce the number of AD computations by 1/s, and for iso-throughput, they also allow us to scale the voltage of the system, resulting in significant energy reduction.

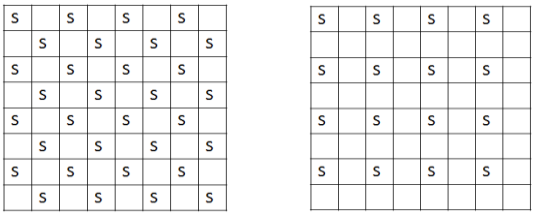


Figure 5: Sub-sampling patterns for ½ and ¼ schemes

D. *Proposed SAD Scheme*

The proposed scheme uses HOC computation and 1-bit LOB truncation. As described earlier, most of the AD computations result in small values and the large values of AD generally contribute to SAD computations of unselected blocks. Thus, we update the SAD value with an approximate value when AD is larger than a threshold value (Thr). The pseudo code for the algorithm is given in Algorithm-1.

In order to avoid 8-bit AD computation before comparing with Thr, we split the computation into two separate parts. Let BBIT be the boundary bit which is determined by the threshold value, BBIT=$log_2 Thr$. Then the lower order bits are LSB to BBIT-1 and higher order bits are BBIT to MSB. LSB is bit 1 since bit 0 has been truncated. We compute the difference of the lower order bits and higher order bits in parallel and combine these results to determine whether AD $>$ Thr and the SAD value should be updated by the correction factor or not.

Algorithm 1. Proposed HOC Computation

Input: Pixels from reference and search blocks.
Output: SAD value of the given blocks
Initialize ***Thr***, SAD=0 and BBIT=$\log_2 Thr$
    for each pixel location in the block
        AD_Low = AD of bits LSB to BBIT-1
        if (AD of pixels> ***Thr***) **then**
            SAD = SAD + correction factor
        else
            SAD = SAD + AD_Low
    end

The threshold value *Thr* has to be selected carefully. A larger value of *Thr* gives better PSNR and CR performance

but has larger circuit overhead including larger critical path delay. Table IV illustrates the performance degradation of three threshold values (*Thr*=16, 32, 64) for two sample video sequences. We see that *Thr*=32 provides very small increase in $\Delta PSNR$ and a small decrease in ΔCR for most of the video sequences. One added advantage of choosing *Thr*=32 is that it splits the 7-bit AD computations into two parts with comparable delay (a 3-bit higher order computation part and a 4-bit lower order computation part). In contrast, *Thr*=64 would split the computations into a 2-bit higher order computation part and a 5-bit lower order computation part. The implementation would have higher delay because of the longer critical path through the 5-bit computation part.

The rest of the paper assumes *Thr*=32. If, however, the application requires higher algorithm performance, then *Thr*=64 should be chosen at the expense of larger circuit overhead.

TABLE IV
AVERAGE INCREASE IN COMPRESSED DATA RATE AND REDUCTION IN PSNR FOR THR = 16, 32, and 64

		Thr=16	**Thr=32**	Thr=64
Coastguard	ΔCR (%)	4.2%	**1.2%**	0.2%
	ΔPSNR (%)	0.1%	**0.03%**	<0.01%
Foreman	ΔCR (%)	4.2%	**1%**	0.2%
	ΔPSNR (%)	0.1%	**0.02%**	<0.01%

IV. Proposed System Architecture

In this section, we describe a new SAD architecture based on the proposed scheme for Thr=32. Figure 6 illustrates the conventional SAD architecture that computes the difference of two 8-bit numbers in parallel (A-B and B-A), picks the positive one and updates the SAD value.

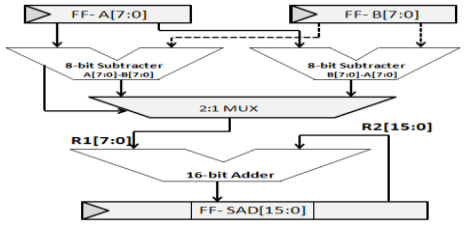


Figure 6: Baseline SAD calculation unit

The proposed architecture is illustrated in Figure 7 for $Thr = 32$. The two LOB unit (LOB1 and LOB2) compute A[4:1]-B[4:1] and B[4:1]-A[4:1], respectively. When B[4:1]>A[4:1], $Cout_1$ of LOB2 is 1. Similarly, when B[7:5]>A[7:5], $Cout_2$ of HOC is 1.

The HOC unit computes the difference B[7:5]-A[7:5] in parallel with the two LOB units. When the result of HOC part is 001, it means that B[7:5]-A[7:5]=32. Now, if $Cout_1$=0 then B[7:1]-A[7:1]<32, and the SAD value is updated using LOB1 if $Cout_2$=0 and LOB2 otherwise; if $Cout_1$=1, then SAD value is updated with the correction factor which is 32. Similarly, when the HOC part is 111, it means that B[7:5]-A[7:5]=-32, and if $Cout_1$=1 then A[7:1]-B[7:1]<32 and the SAD value is updated using LOB1 if $Cout_2$=1 and LOB2 otherwise; if $Cout_1$=0 it is updated with the correction factor. When the HOC part is 000, the SAD is updated using LOB1 if $Cout_1$=0 and LOB2 otherwise. For all the other HOC results, SAD is updated using the correction factor. The *HOC logic* unit has inputs $Cout_1$, $Cout_2$ and the 3-bit HOC output and generates 2 control signals to select either LOB1output or LOB2 output or 0, and a third signal corresponding to bit 5 of R1. Note that bit 5 corresponds to the correction factor 32. The *HOC logic* has a critical path of 2 NAND and 1 NOR gates. Figure 8 gives the flow chart of the proposed scheme.

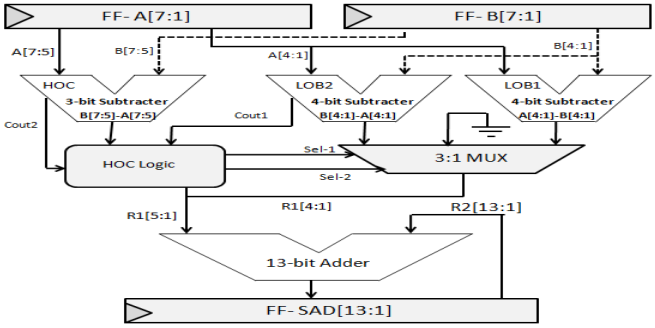


Figure 7: Proposed SAD calculation unit

The proposed architecture has smaller critical path delay compared to the baseline architecture. First, the LOB and HOC units compute in parallel and the delay of the HOC logic unit is quite small. Next, the bit-widths of R1 and R2 are reduced. The bit-width of R1 is reduced from 8 bits to 5 bits. As a result, the maximum SAD value is reduced from 16 bits to 13 bits. The last stage now has a 13 bit adder which adds the first 5 bits using full adders and the last 8 bits using half adders. The new adder has smaller critical path and energy consumption compared to the general 16 bit adder. Furthermore, to reduce the critical path delay of the half adder chain, we merge two half adder modules as illustrated in

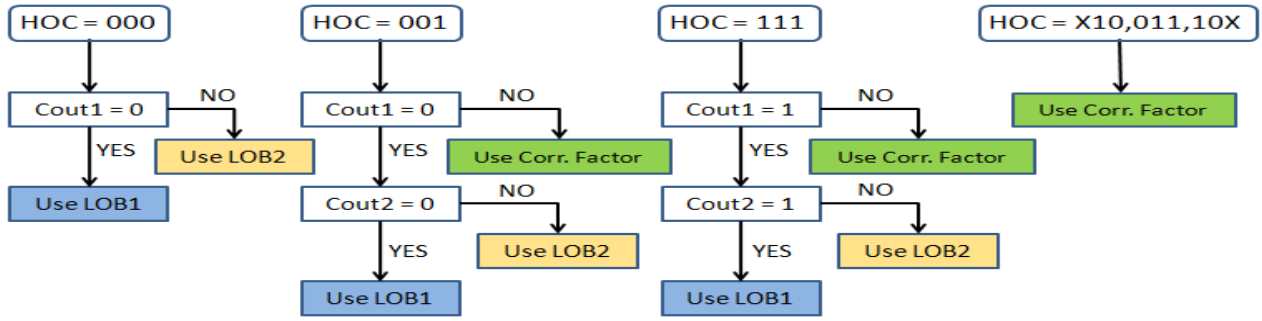


Figure 8: Flow Chart of the proposed SAD calculation scheme

Figure 9. The carry-ripple path for a single 2 bit HA chain is now only one NAND and one NOR gate instead of two AND gates.

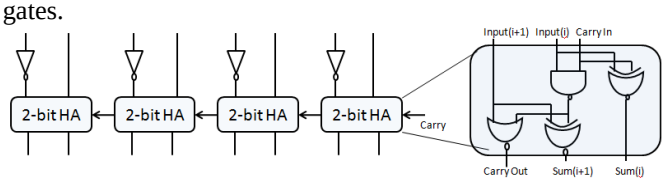


Figure 9: 8-bit Half Adder Chain

We use high performance 45nm PTM models [16] to implement our system. The nominal voltage is equal to 1V. Delay and energy values of the basic gates are generated using HPSICE and PTM models. Then, delay and energy for the SAD architectures are computed for different input values using Verilog on ModelSim simulator [17]. For the addition and subtraction units, ripple carry adder (RCA) structure is used, because, for small bit widths, the energy delay product of RCA outperforms fast adder structures.

We compare critical path delays vs average energy consumption of the proposed SAD unit with HOC (Thr=32) & LOB (1-bit) (described in Figure 7) and the baseline architecture (described in Figure 6). The performance curves of the proposed and baseline schemes for different voltage levels are illustrated in Figure 10. At nominal voltage, the proposed scheme can reduce critical path delay by 36.7% and average energy consumption by 37.5%. For iso-throughput, we scale the supply voltage of the proposed architecture to 0.7V, resulting in a 68% drop in energy as shown in Figure 10.

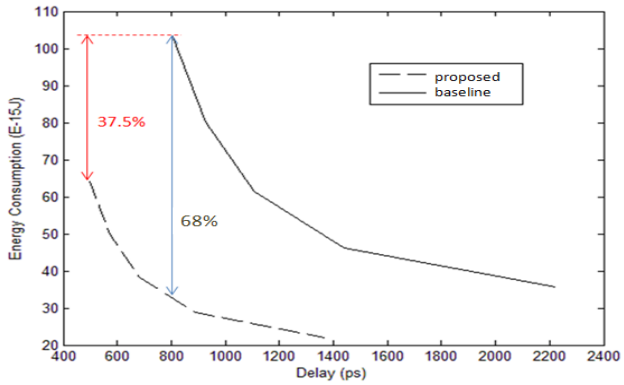


Figure 10: Delay vs Energy Consumption of the baseline and proposed schemes

We also implemented the delay predictor circuitry of [8], [15] in the proposed SAD unit. Even though it can reduce the critical path delay by an additional 7.5%, on average, it requires 5% additional cycles, and additional memory to hold the clock gated data. Furthermore, for an 8x8 block, delay prediction unit improves the throughput by 7.9% but increases the total energy consumption by approximately 10% for our proposed scheme. So we did not consider it for further investigation.

V. Performance Analysis

A. *Algorithm-Quality Performance*

Figure 11 illustrates the performance of the proposed technique and its variants for 3 different video sequences (Football, Foreman and Claire). In all cases, we find that the performance degradation is fairly low. There are several factors that help to achieve low degradation. The proposed scheme i) retains computations with low order bits, which contribute more effectively to the SAD computation of the selected block, ii) approximates the computation of higher order bits which does not affect the performance much since higher AD values typically correspond to blocks that are likely not to be selected. We also observe that ½ and ¼ sub-sampling schemes provide decent quality performance, which makes these schemes attractive especially since they do not require any change in the overall SAD architecture.

Next we present the performance results for the 13 video sequences. Figure 12 illustrates the cost of using the proposed schemes in terms of percentage increase in compressed data rate and reduction in PSNR. We see that the proposed scheme without sub-sampling provides very good performance results. On average, it only increases ΔCR by 1.8% and reduces $\Delta PSNR$ by 0.08%. The proposed scheme with ½ sub-sampling increases ΔCR by 6.5%, on average, and reduces $\Delta PSNR$ by less than 1%. Table-V summarizes average performance degradations of the proposed scheme and its variants for diamond search and three step search. Note that the type of search has little effect on the ΔCR and $\Delta PSNR$ performance.

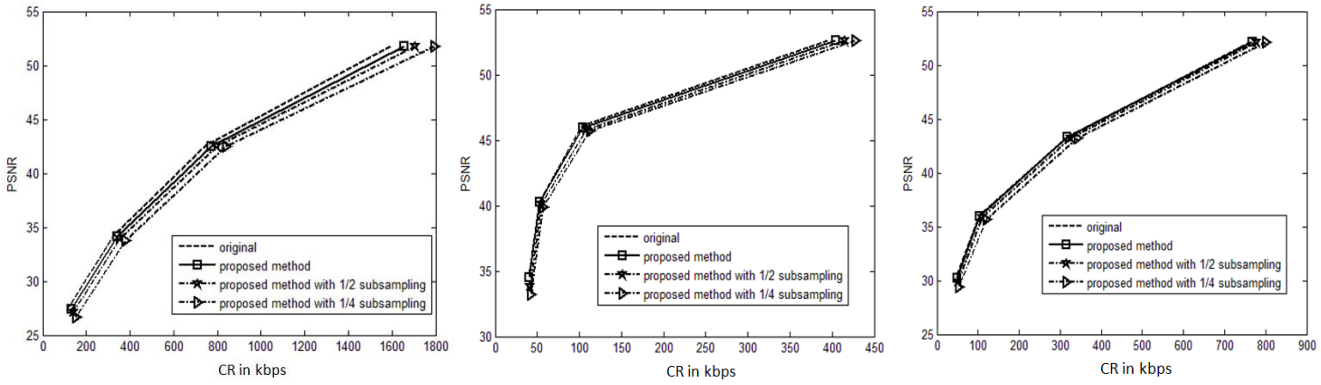


Figure 11: PSNR vs Compressed Data Rate of a) Football, b) Claire and c) Foreman sequences under different configurations

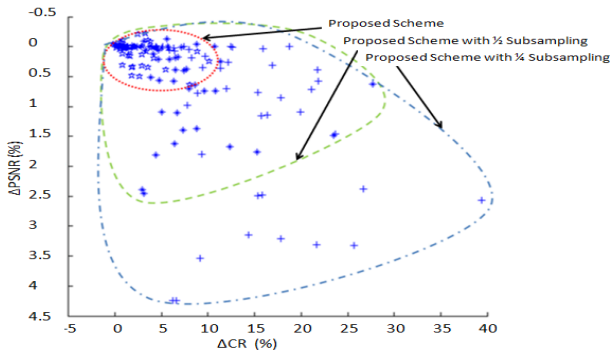


Figure 12: Cost of using proposed scheme in terms of percentage increase in compressed data size and PSNR reduction.

B. *Energy-Quality Trade-offs*

Next, we compare the energy and algorithm quality of the proposed scheme and its variants with several candidate schemes. The candidate schemes are as follows:

A) LOB truncation. A.1: 1-bit truncation, A.2: 2-bit truncation, A.3: 3-bit truncation and A.4: 4-bit truncation. In each case, we modify the sizes of the internal computation blocks of the baseline SAD unit in Figure 6.

B) Subsampling schemes. B.1: ½ sub-sampling, and B.2: ¼ sub-sampling. There is no modification in the baseline SAD unit.

C) Proposed scheme. C.1: HOC (Thr=32) & LOB (1-bit), C.2: HOC (Thr=32) & LOB (1-bit) & ½ sub-sampling and C.3: HOC (Thr=32) & LOB (1-bit) & ¼ sub-sampling. The proposed SAD unit of Figure 7 is used.

Table-VI illustrates the critical path delays, energy consumption per AD computation, total energy per SAD computation of 8x8 block, $\Delta PSNR$ and ΔCR of all the candidate schemes. As expected, increasing LOB truncation reduces energy and delay while incurring higher performance degradation. On the other hand, sub-sampling schemes reduce the total number of AD computations for a single search resulting in significant reduction in energy. However, they result in quite high performance degradation compared to other techniques. The proposed schemes achieve a balance between quality degradation and energy consumption. For instance, although proposed scheme C.1 has approximately the same performance degradation as A.2, its critical path delay is 21.5% smaller and thus can be operated at lower voltage. Furthermore, proposed schemes C.2 and C.3 require 37.5% less energy compared to their counter parts B.1 and B.2 at the

TABLE-V
AVERAGE INCREASE IN COMPRESSED DATA RATE AND REDUCTION IN PSNR

	HOC (Thr=32) with 1-bit LOB truncation		HOC (Thr=32) with 1-bit LOB truncation and ½ sub-sampling		HOC (Thr=32) with 1-bit LOB truncation and ¼ sub-sampling	
	DS	TSS	DS	TSS	DS	TSS
CR increase (%)	1.8%	1.9%	6.5%	6.8%	12.4%	12.2%
PSNR Reduction(%)	0.08%	0.08%	0.5%	0.6%	0.9%	1%

TABLE-VI
COMPARISION OF QUALITY AND ENERGY CONSUMPTION AT NOMINAL VOLTAGE OF ALL CANDIDATE SCHEMES

	Baseline	A.1	A.2	A.3	A.4	B.1	B.2	**C.1**	**C.2**	**C.3**
Delay (ps)	807	724	651	576	507	807	807	**511**	**511**	**511**
Energy (E-15J) per AD computation	103	82.4	64.9	49.4	39.1	103	103	**64.4**	**64.4**	**64.4**
Total Energy(E-15J) per SAD computation	6592	5273	4153	3161	2502	3296	1648	**4121**	**2060**	**1030**
$\Delta PSNR$	0	0.05	0.21	0.6	1.16	0.5	0.9	**0.08**	**0.6**	**1**
ΔCR	0	0.5	1.8	4	7.8	5.1	10.4	**1.8**	**6.5**	**12.3**

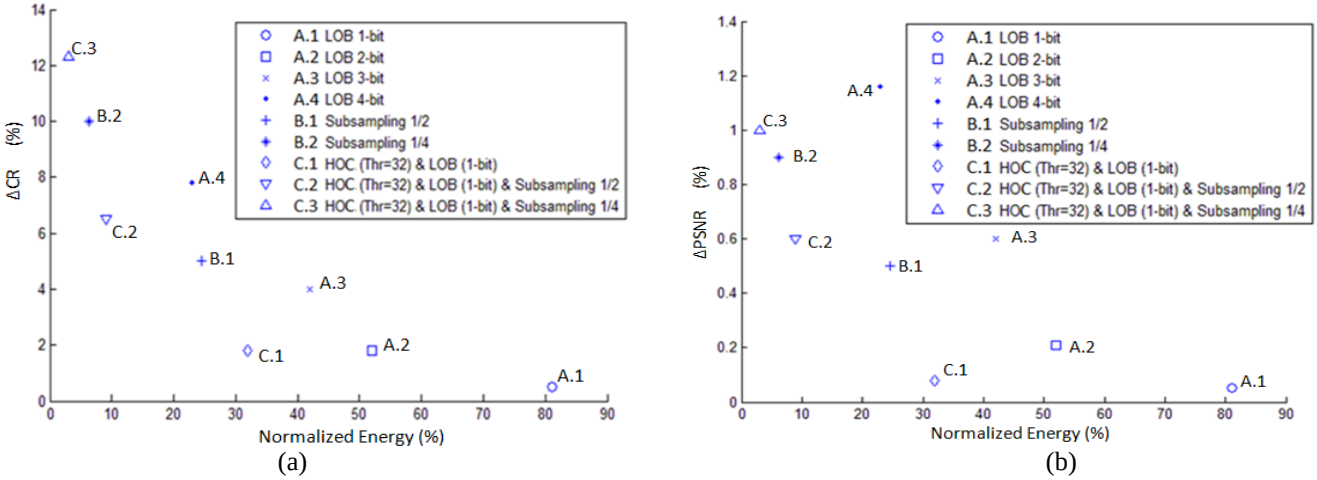


Figure 13 a) Compressed Data Rate Increase (ΔCR) vs Normalized Energy, b) PSNR degradation ($\Delta PSNR$) vs Normalized Energy

expense of slightly higher performance degradation..

Figure 13 compares the PSNR drop, CR increase and normalized energy consumption under iso-throughput conditions. To compute energy consumption for iso-throughput, we apply voltage scaling so that the delay is comparable to the baseline scheme. We compute the energy consumption and normalize it to that of the baseline scheme. From Figure 13, we see that the 4-bit LOB truncation scheme achieves 77% reduction in normalized energy. For this energy reduction, sub-sampling by a factor of 2 (B.1) incurs 2.7% less increase in ΔCR and 0.6% less reduction in $\Delta PSNR$, and is thus more competitive. The proposed schemes achieve better energy and quality performance compared to single LOB and sub-sampling schemes. For instance, normalized energy consumption of C.2 is approximately 15% lower than B.1 for a comparable ΔCR. The normalized energy consumption of B.1 and C.1 are comparable but the ΔCR of C.1 is approximately 3.3% lower compared to B.1. Also, C.1 achieves approximately 20% less normalized energy consumption compared to A.2 for comparable ΔCR. Of the other two proposed schemes, C.2 has 90% lower energy consumption with 6.5% increase in ΔCR and C.3 has 97% lower energy consumption with 12.5% increase in ΔCR. Figure 13-b compares the performance with respect to $\Delta PSNR$. While the trends are similar, the $\Delta PSNR$ values are very small for all the schemes.

Related work: In [3], adaptive bit truncation has been proposed to reduce energy consumption. It has been shown that 58% average energy saving is possible with 4-bit LOB truncation without voltage scaling while incurring PSNR degradation up to 1% compared to a baseline video sequence. For iso-throughput, proposed scheme C.1 has ~5% higher energy consumption but 6% lower ΔCR and 1% higher PSNR. In [9], voltage overscaling is employed to reduce power consumption of the ME part. A sub-sampled replica of the original computation is used to detect any error due to delay violation for scaled voltage levels. At nominal voltage, the scheme provides error free performance and it is shown that with the help of voltage scaling 60% energy saving is possible with 2% reduction in PSNR of the video sequence. In contrast, our proposed scheme achieves only 0.08% degradation in PSNR for 68% of energy reduction.

VI. Conclusion

This paper presents a novel SAD computation scheme that achieves significant reduction in energy consumption while incurring slight quality degradation. It exploits the features of AD and SAD distributions to derive a scheme that carefully approximates the the computations in the SAD unit. If the AD value is larger than a threshold, it is likely to contribute to the SAD value of an unselected block. Thus large AD values can be approximated and the corresponding SAD value updated with a correction term. Furthermore, one lower order bit is also truncated to reduce the computational load. The resulting architecture has a lower critical path delay compared to the baseline architecture and significantly lower energy consumption. It achieves 37.5% energy reduction at nominal voltage and 68% reduction for iso-throughput while incurring 1.8% increase in compressed data size and 0.08% reduction in PSNR. Two variants of this scheme based on sub-sampling further reduce the energy consumption. If higher compressed data sizes are tolerable, the proposed scheme with ½ sub-sampling can achieve 90% reduction in energy consumption with 6.5% increase in compressed data size and 0.6% reduction in PSNR.

References

[1] C. Chen, S. Chien, Y. Huang, T. Chen, T. Wang, and L. Chen, "Analysis and Architecture Design of Variable Block-Size Motion Estimation for H.264/AVC", IEEE Trans. on Circuit and System-I, vol. 53, no. 2, pp. 578-593, Feb. 2006.

[2] P. Kuhn, "Algorithms, Complexity Analysis and VLSI Architectures for MPEG-4 Motion Estimation", Boston, MA, Kluwer, 1999.

[3] Z. He, C. Tsui, K. Chan, and M.L. Liou, "Low-Power VLSI Design for Motion Estimation Using Adaptive Pixel Truncation", IEEE Trans. on Circuits and Systems For Video Technology, vol. 10, no 5, pp. 669-678, August 2000.

[4] R. Li, B. Zeng, and M. L. Liou, "A New Three-Step Search Algorithm for Block Motion Estimation", IEEE Trans. on Circuits and Systems For Video Technology, vol 4., no. 4, pp. 438-442, August 1994.

[5] S. Zhu, and K. Ma, "A New Diamond Search Algorithm for Fast Block-Matching Motion Estimation", IEEE Trans. on Image Processing, vol 9, no. 2, pp. 287-290, February 2000.

[6] S. Yang, W. Wolf and N. Vijaykrishnan, "Power and Performance Analysis of Motion Estimation Based on Hardware and Software Realization", IEEE Trans. on Computers, vol 54, no 6, pp. 714-726, June 2006.

[7] Y. Huang, B. Hsieh, S. Chien, S. Ma, and L. Chen "Analysis and Complexity Reduction of Multiple Reference Frames Motion Estimation in H.264/AVC", IEEE Trans. on Circuits and Systems For Video Technology, vol. 16, no 4, pp. 507-522, April 2006.

[8] D. Mohapatra, G. Karakonstantis and K. Roy, "Significance Driven Computation: A Voltage-Scalable, Variation-Aware, Quality-Tuning Motion Estimator", International Symposium on Low Power Electronics and Design, pp.195-200, August 2009.

[9] G. V. Varatkar and N. Shanbhag, "Error-Resilient Motion Estimation Architecture", IEEE Trans. on VLSI, vol 16, no 10, pp. 1399-1412, Oct 2008.

[10] http://trace.eas.asu.edu/yuv/

[11] I. E Richardson, "H.264 and MPEG-4 Video Compression", John Wiley & Sons, September 2003.

[12] H. S. Malvar, A. Hallapuro, M. Karczewicz, and L. Kerofsky, "Low-Complexity Transform and Quantization in H.264/AVC", IEEE Trans. on Circuits and Systems For Video Technology, vol. 13, no 7, pp. 598-603, July 2003.

[13] JVT Document JVT-C028, G.Bjontegaard and K. Lillevold, "Context-Adaptive VLC Coding of Coefficients", Fairfax, VA, May 2002.

[14] A. Sinha, A. Wang and A. Chandrakasan, "Energy Scalable System Design", IEEE Trans. on VLSI, vol 10, no 2, pp. 135-145, Apr. 2002.

[15] S. Ghosh and K. Roy, "Exploring High-Speed Low-Power Hybrid Arithmetic Units at Scaled Supply and Adaptive Clock-Stretching", Asia and South Pacific Design Automation Conference, pp. 635-640, 2008.

[16] http://ptm.asu.edu/

[17] http://model.com/content/modelsim-pe-student-edition-hdl-simulation