K. Ayyappa Swamy, G. Hemanth Kumar, M. Balaji / International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622 www.ijera.com
Vol. 3, Issue 1, January -February 2013, pp.1522-1526

Low Bandwidth and Low Power Video Encoding

K. Ayyappa Swamy, G. Hemanth Kumar, M. Balaji
(Assistant Professor of EIE Department, Sree Vidyanikethan Engineering College, Tirupati,)
(Assistant Professor of EIE Department, Sree Vidyanikethan Engineering College, Tirupati,)
(Assistant Professor of EIE Department, Sree Vidyanikethan Engineering College, Tirupati,)

ABSRACT
In video compression, using reference frames next frames are represented with less number of bits. For this the reference frame has to be stored in eternal memory. Accessing reference frame while ME and MC, Large external memory bandwidth required this leads to increased system power dissipation and cost. By compressing the reference frame before storing to external memory can significantly reduces the memory bandwidth and power with minimal impact on quality. For this MMSQ-EC algorithm is used.

Keywords — Chroma pixels, DDR SDRAM, luma pixels, macroblock (MB), motion compensation (MC), motion estimation (ME), scalar quantization.

I. INTRODUCTION
In video coding, one or more decoded frames are used as reference frames. Reference frames are stored in an external memory. Accessing them results in large external memory bandwidth and power consumption. The energy spent in accessing a unit of data from the DDR can be as much as 50 times larger than accessing same amount of data from on-chip static random access memories [1]. In multimedia applications, up to 50% of the total system power budget can be attributed to external memory accesses [2].

In the past, focus has been on reducing DDR bandwidth in a video decoder system [3]. Compressing reference frames before storing to DDR can significantly reduce DDR traffic and power. While lossless compression would provide best quality, the compression ratio that can be achieved is limited [4]. Further, variable length coding associated with lossless compression increases the complexity of data access. Lossy compression can achieve higher compression ratio even if it results in small loss in quality [7], [8]. However, the traditional lossy compression approach can cause drift since encoder and decoder use nonidentical reference data. Recently, video encoding has become an important feature of handheld devices such as mobile phones. In [5], a wavelet transform-based reference frame compression scheme was proposed. Either lossy or lossless compression could be used. However, the complexity of being able to efficiently access the compressed data was not considered. In [6] memory bandwidth optimizations based on data reuse and on the fly padding were proposed, but no reference frame compression was used. Lossy scalar quantization-based compression scheme [min-max scalar quantization (MMSQ)] was proposed to reduce encoder bandwidth. Compressed reference was stored in DDR and was used for motion estimation (ME) and motion compensation (MC). As MC used lossy reference, the scheme is not compliant to the existing standards.

We propose a modification to this technique which is compliant to the existing video standards and, hence, avoids drift in a standard decoder. While we use the lossy reference for ME, MC is performed in bit-accurate fashion. This is achieved by separately storing the quantization error. The impact of the compression scheme on the ME quality is negligible. Also, the bit-accurate MC helps significantly in reducing quality loss due to lossy compression at same compression ratio.

II. DDR Bandwidth Requirement in Video Encode Application
A. Nature of DDR Traffic in Encoder
During ME, luma data from reference frames is fetched from the DDR. Most ME algorithms typically use a rectangular region called "search window" within the reference frame to search for a good match with current macroblock (MB). Search windows for neighboring MBs have a large overlap. Since the MBs are coded in raster-scan fashion, horizontal overlap can be easily exploited. For example, Fig. 1 shows a "sliding search window" that can be created in on-chip memory. DDR bandwidth can be reduced further by exploiting search window overlap in vertical direction. However, this requires large on-chip memory and hence is costly. We assume a sliding search window for analysis. With sliding search window, the traffic due to ME is proportional to the vertical search range.

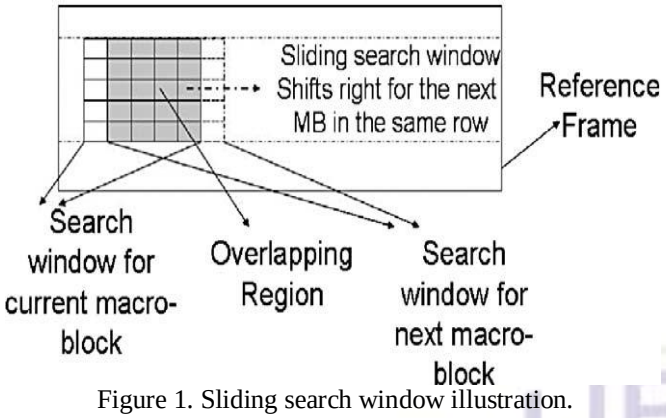


Figure 1. Sliding search window illustration.

The burst accesses from DDR are much more efficient when accessed in a regular fashion such as a rectangle. Also, the latency of access from DDR is large. For efficient ME hardware utilization, the search region for a MB is fetched from the DDR, and kept in local memory in advance. Hence, the rectangular search window as described above is commonly used irrespective of the ME search pattern. For example, full search method performs exhaustive search, whereas algorithms, such as UMHexagonS and EPZS search, optimize the search by intelligently choosing positions in the search window, thereby reducing computations and local memory accesses. However, all these algorithms will still typically implement a rectangular search window in the local memory.

B. Constraints on the Compression Scheme

Reference frames are stored to and retrieved from DDR in different order. MBs in current frame are processed in rasterscan fashion, while ME accesses reference data as vertically aligned MBs. Also, for MC, nondeterministic blocks of data that can span across multiple MBs need to be read. The quality loss resulting due to lossy compression should be small. In the next section, we describe the proposed scheme that satisfies all these constraints.

III. DESCRIPTION OF PROPOSED SCHEME

First reference frame is decomposed into blocks which are known as MBs. Reconstructed MBs undergo in-loop lossy compression using MMSQ (MMSQ algorithm is described in Section III-A). The scheme provides fixed compression ratio of reference frame data with small quality loss. We refer to this scheme as MMSQ-IL. In our proposed scheme (Figure 2), the reference luma frame data is compressed using MMSQ. Compressed reference and quantization error due to compression are stored to the DDR. For ME, compressed reference is used, while for MC, only the error data is fetched from DDR, and is combined with the decompressed reference data already fetched for ME, to recreate bit-accurate reference pixels. This ensures standard compliance. This proposed method is referred as MMSQ-EC.

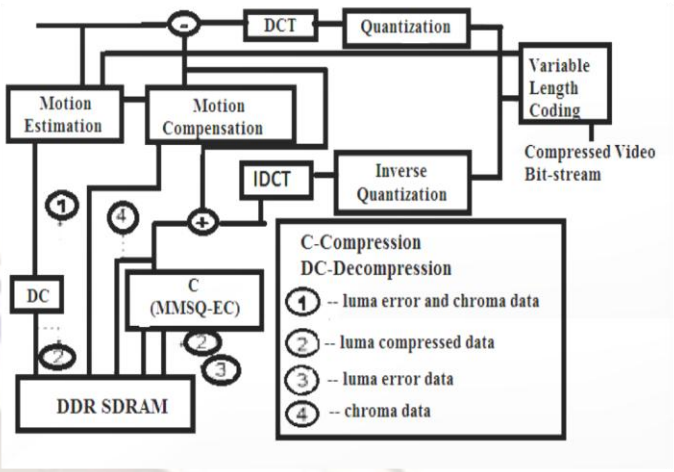


Figure 3. Proposed ME bandwidth reduction using luma compression and error storage (MMSQ-EC). Only error data is fetched during MC.

VI. ALGORITHM IMPLEMENTATION

Reference frame is taken and it is divided into MBs of mxm size using block extractor. For each block, max and min values are found and stored as part of compressed data (8 bit each). For each pixel, difference between the pixel value and minimum value is quantized to *n* bits. The quantized pixels with the max and min values form the compressed data block. For example, $n = 3$ gives a compression ratio of 2. We also calculate the quantization error for each pixel. We show that the quantization error can be represented using fixed number

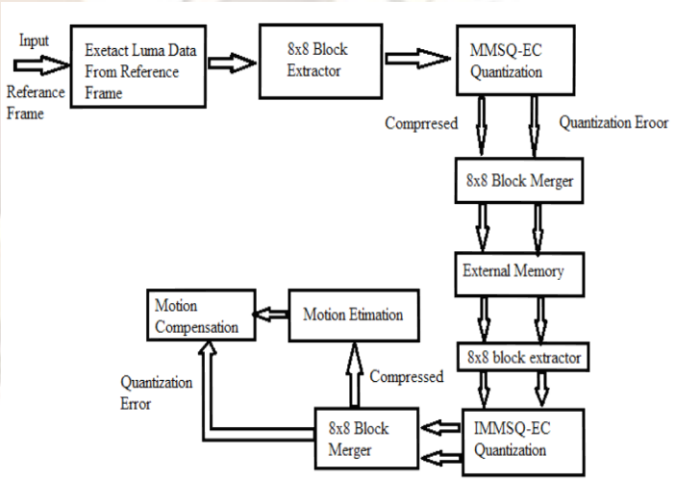


Figure 3. block diagram for implementation of MMSQ-EC algorithm

of bits, which is smaller than a full pixel. This keeps the DDR bandwidth needed for MC luma error accesses small and also enables random access. Bound on quantization error is calculated. In reality,

the error is slightly smaller than this bound. For example, with the maximum range of 256, and n = 3, the error is bounded as $-16 \leq error \leq 15$.

A further optimization is made to limit the size of the error data compared to that. In case the error is too large, MMSQ-EC is replaced by truncation. For example, for n=3, if *range* > 127, then error size is 5 bits per pixel (b/p). This can be limited to 4 bits with truncation. This not only reduces the MC bandwidth but also helps in efficient data packing in DDR memory.

Each nxn block of compressed reference frame has to be converted back to full frame. This is to get back the compressed reference frame. This compressed frame is used for motion estimation. After separating the compressed reference frame and quantization error from reference frame they are stored in memory. For ME, compressed reference is used, while for MC, only the error data is fetched from DDR, and is combined with the decompressed reference data already fetched for ME, to recreate bit-accurate reference pixels. In this the each pixel in the block are added to the min of that block which is given at the time quantization. And quantization error is added and then in verse quantized. This process is lossy but the loss is negligible. And finally the output of inverse quantization is used for motion compensation.

V. RESULTS

The table below gives the total number of bits requires to represent the reference frame (I) with and without MMSQ-EC algorithm and the bit per second have to access from external memory with and without MMSQ-EC algorithm. The experiment conducted on three videos 'news.avi'. The number of bits to represent a pixel is 8 bits per pixel. And the total number pixels per frame are 16384. So bits require to represent a reference frame are 131072. For motion estimation and motion compensation we need to fetch reference frame from external memory if we consider 30 frames per second then totally 60 times the reference frame has to be fetched from external memory this needs more bandwidth and power. When compared to internal memory, accessing external memory requires 50 times more power. MMSQ-EC algorithm is use full in order to reduce this power and band width.

By varying the block size in MMSQ-EC algorithm that is clear that the PSNR, Time required for computing and Number of bits per second requires being access from external memory are decreased when the block size is increased. The number of bits per second decreases when block size increased because the minimum and quantization values increases with number blocks increases when block size decreased the number of blocks increased. PSNR also decreased with increase in block size because that the quantization error increases with block size because the increase in range of each block. From the above figure 4 it is clear that the PSNR. From the above table it is clear that by using MMSQ-EC algorithm the Number of bits per second requires to be accessed from external memory are reduced by a factor more then 3. So the band width and power require to access the external memory are also reduced. Decreases with increase in block size. This experiment is done on 'news.avi' video in this video first 10 frames are taken as I1, B2, B3, P4, B5, B6, P7, B8, B9, I10 frames. PSNRs are calculated for B2, P4, P7, and B8 frames for different block sizes 4, 8, 16, 32, 64, 128. PSNR must be as high as possible so the block size is as high as possible when we take block size as 4 then the computation time is very high and the number of block to process is also high. When the block size is more 128 then the compression ratio is less. So for better performance the block size should be 8. At block size 8 we will get modified results.

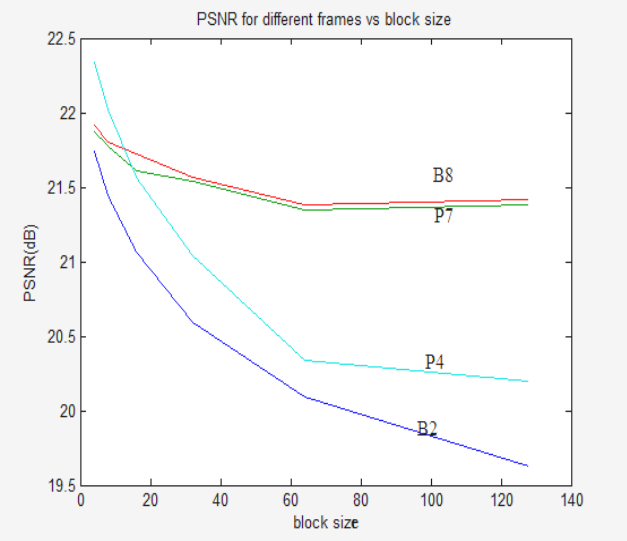


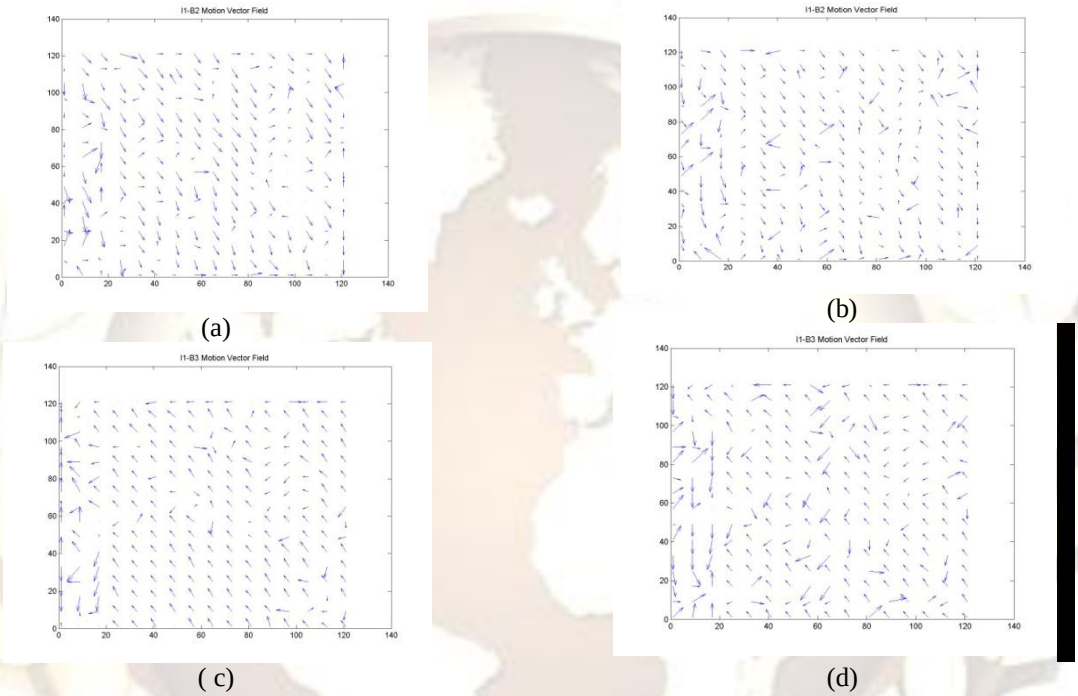
Figure 4. block size vs. PSNR for different frames.

TABLE I. COMPARING ALGORITHMS WITH AND WITHOUT MMSQ-EC ALGORITHM

	Number of bits require to represent reference frame	Number of bits per second require to be accessed from external memory
Without MMSQ-EC algorithm	131072	7864320
With MMSQ-EC algorithm	68864	2065920

For different block sizes number of bits per second and the PSNR for B2, P4, P7, B8 frames and time require for compression.

Block size	frame	Number of bits per second	PSNR of B2 frame (dB)	PSNR of P4 frame (dB)	PSNR of P7 frame (dB)	PSNR of B8 frame (dB)	Time (sec)
4	I1	2365440	21.7452	21.8767	21.9246	22.3412	130.8262
8	I1	2064920	21.4423	21.7819	21.8084	22.0234	17.9737
16	I1	1991040	21.0744	21.6111	21.7228	21.5765	6.4835
32	I1	1960200	20.5923	21.5494	21.5789	21.0476	5.8102
64	I1	1967640	20.0912	21.3527	21.3809	20.3490	5.8762
128	I1	1966470	19.6376	21.3867	21.4201	20.2034	5.9029



(a) (b) (c) (d)

Figure 5. Motion vector of I1-B2 (a) and I1-B3 (b), (c) and(d) are motion vector fields with MMSQ-EC algorithm

The above figures are the motion vectors fields due to motion vectors between first frame and second frame (a) as well as first frame and third frame (b) of the video 'news.avi' and third and forth figures (c), (d) is the motion vectors of the same frames of 'news.avi' video using MMSQ-EC algorithm for video compression. By observation it is clear that bought motion vector fields are almost same, that mean by using MMSQ-EC algorithm there is not much deviation in the motion vectors.

And the images shown below are frames of the video 'news.avi' which is compressed using normal compression algorithm and algorithm with MMSQ-EC algorithm. First image is the first frame that means reference frame of the given video next are B2, B3, reconstructed B2, reconstructed B3, reconstructed using MMSQ-EC B2, and reconstructed using MMSQ-EC B3 respectively. By observation it is clear that visually there is not much difference in the reconstructed frames using normal method and MMSQ-ED algorithm.

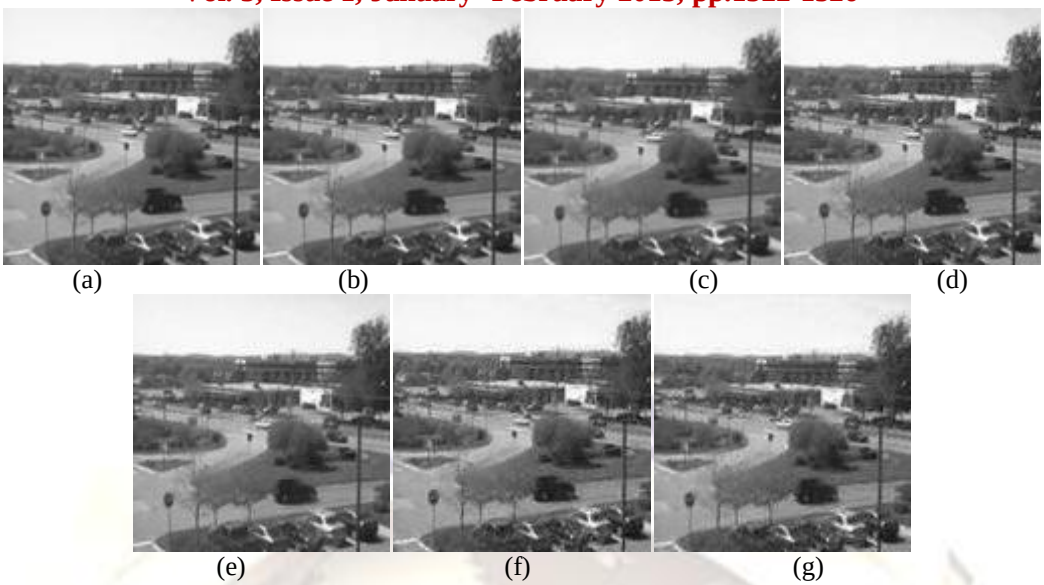


Figure 6. (a) reference frame, (b) B2 frame, (c) B3 frame, (d) reconstructed B2 frame, (e) reconstructed B3 frame, (f) reconstructed B2 frame using MMSQ-ECand (g) reconstructed B2 frame using MMSQ-EC

VI. CONCLUSION

Bandwidth and power reduction technique for video encoder, using lossy reference frame compression based on compression error calculation is presented. The fixed compression ratio, transform free technique is low in complexity and is compliant with existing standards. And the results are shown that by using this algorithm Bandwidth and power are reduced.

ACKNOWLEDGMENT

The authors would like to acknowledge Prof. S. Sengupta Dept. of Electronics & communication Engg. I.I.T Kharagpur for his video material on digital video & picture communication.

REFERENCES

[1] J. Trajkovic, A. V. Veidenbaum, and A. Kejariwal, "Improving SDRAM access energy efficiency for low power embedded systems," *ACM Trans. Embedded Comput. Syst.*, vol. 7, no. 3, article 24, p. 21, Apr. 2008.

[2] F. Catthoor, F. Franssen, S. Wuytack, L. Nachtergaele, and H. D. Man, "Global communication and memory optimizing transformations for low power signal processing systems," in *Proc. Workshop VLSI Signal Process.*, Oct. 1994, pp. 178–187.

[3] T. Takizawa, J. Tajime, and H. Harasaki, "High performance and cost effective memory architecture for an HDTV decoder LSI," in *Proc. Int. Conf. Acou., Speech Signal Process.*, vol. 4. Mar. 1999, pp. 1981–1984.

[4] L. Benini, D. Bruni, A. Macii, and E. Macii, "Memory energy inimization by data compression: Algorithms, architectures and implementation," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 12, no. 3, pp. 255–268, Mar. 2004.

[5] C. Cheng, P. Tseng, C. Huang, and L. Chen, "Multi-mode embedded compression codec engine for power-aware video coding system," in *Proc. IEEE Workshop Signal Process. Syst. Des. Implement.*, Nov. 2005, pp. 532–537.

[6] T.-C. Chen, S.-Y. Chien, Y.-W. Huang, C.-H. Tsai, C.-Y. Chen, T.- W. Chen, and L.-G. Chen, "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.

[7] M. Budagavi and Z. Minhua, "Video coding using compressed reference frames," in *Proc. Int. Conf. Acou., Speech Signal Process.*, Mar. 2008, pp. 1165–1168.

[8] S. Lei, "Compression and decompression of reference frames in a video decoder," U.S. Patent 6272180, Aug. 7, 2001.