# Reduction of Frame Memory Accesses and Motion Estimation Computations in MPEG Video Encoder

Wei-Cheng Lin and Chung-Ho Chen
Department of Electrical Engineering and
Institute of Computer and Communication Engineering, National Cheng-Kung University
No.1 University Road, Tainan, Taiwan 70101, R.O.C.
kevin@casmail.ee.ncku.edu.tw chchen@mail.ncku.edu.tw

*Abstract*—**This paper presents an approach that reuses data stored in the frame memory and in the motion estimation (ME) internal buffer to avoid unnecessary memory accesses and redundant ME computations for MPEG video encoders. This work employs a macroblock bitmap table, which can be easily maintained, to locate the reusable data. The experimental results show that the proposed scheme is particularly efficient in low motion video sequences, approximately saving 18% of the frame memory accesses as well as about 16% of the ME computations without any sacrifice in the image quality.**

***Keywords: data reuse, frame memory, MPEG video encoder, motion estimation, stationary macroblock***

## I. INTRODUCTION

The MPEG video encoder system puts heavy demands on data transfer. For example, in MPEG-2 video encoding, around 112 Mbyte data transfers per second are required for encoding a 30 fps video sequence in 4:2:0 format with a frame size of 720×576 pixels and search range of −16 to 15 pixels. The frame memory, which is typically too large to fit into the embedded memory and usually located in external memory, stores reconstructed anchor frames (I- or P-frame) for reference; it dominates total memory usage in the encoder due to high video compression rate. For instance, frame memory accesses are responsible for approximately 98% of the total memory accesses with a video compression rate of 30. Such a large data transfer becomes the dominant part in the power consumption of a video encoder. On the other hand, motion estimation is also a major consumer of overall system energy, consuming about 50% of the total computational resources available to the encoder [1].

When low activity videos (e.g., video conference and video surveillance) which contain highly correlated images and large static video objects are encoded, there exist many stationary macroblocks. This work exploits the stationary macroblock characteristic to avoid unnecessary frame memory accesses and redundant ME computations in the following ways:

- Writing the reconstructed macroblock of an anchor frame to the frame memory is redundant if the macroblock to be overwritten has the same content as that.
- Transferring the macroblock of the future reference frame from the frame memory to a ME internal buffer for bi-directional prediction is unnecessary if the co-located macroblock in the pervious reference frame with same content has been transferred to the buffer.
- Backward ME operations can be omitted if the search area data of the previous and future frames are the same.

The performance of the proposed method greatly depends on the percentage of the stationary macroblocks. Some video sequences show more than 85% stationary macroblocks in a frame, resulting in greater opportunity to improve the performance of the frame memory and the ME. The proposed scheme is evaluated with various video benchmarks. This approach is very effective for low motion video sequences. On the average, this work has obtained about 18.4% saving in frame memory accesses and reduced 16.1% in ME computations for low motion video without any picture quality loss.

The remaining of the paper is organized as follows. Section II discusses related work. Section III presents the methodology that reduces memory accesses and ME computations. Simulation results are discussed in Section IV. Finally, Section V concludes this paper.

## II. RELATED WORK

Reducing memory access in power-aware codec designs has been explored in many different ways [2-9]. Several works focus on the external memory interface design such as [2, 3]. Kim and Park proposed an address translation technique [2] to reduce the number of row-activations for external DRAM access in MPEG-2 video decoder applications. In [3], the impulse memory controller remaps the non-sequential data to unused physical address to increase bus utilization.

A number of recent researches investigate memory hierarchies that make use of the data locality to reduce external references for multimedia applications [4-9]. The processing core accesses faster and smaller local memories [4] including caches [6] for heavily used data. Some works propose the use of software controlled memory, such as the streaming memory [7]

1-4244-1251-X/07/$25.00 2007 IEEE. 

and scratch-pad memory [8] to reduce data transfers between the on-chip and off-chip memory.

Some proposals attempt to explore reusable data at the algorithmic level by increasing data reuse for data-dominated operations such as the motion estimators [10-12] or the deblocking filter [13]. These works mainly focus on the reuse of macroblocks by avoiding repeated access from the memory. In [10], Migita and et al. proposed an architecture that reduces computations of bi-directional motion estimation by exploiting the stationary macroblocks in the future reference frame but at the expense of degrading image quality.

Shih and et al. proposed a mechanism to reduce memory accesses and the size of memory requirements for MPEG-2 video decoder by using a variable-length codec which compresses the reconstructed frame data before writing into the memory, and de-compresses the frame data when read from the memory for motion compensation or display [14].

## III. THE PROPOSED METHODOLOGY

The MPEG-2 standard defines three types of frames (i.e., I, P and B). The I-frame is encoded independently without reference to other frames. The P-frame is encoded by taking reference frame information from the last encoded anchor frame while the B-frame is based on the last two encoded anchor frames: previous and future reference frames. Since the B-frame uses backward prediction, the future reference frame must be encoded prior to encoding the B-frame. Therefore, the encoding order of frames is different from the sampling order as illustrated in Figure 1, where the arrows indicate the prediction directions and anchor frames are shown in bold.

Figure 2 depicts the architecture of an MPEG-2 video encoder showing the typical processing flow. The shaded blocks in the figure represent the frame memory, which is used to store reference frames and input original frames for reordering. The reconstructed frame buffer stores two anchor frames for bi-directional prediction; two search window buffers are employed to reduce the number of read accesses to the reconstructed frame buffer. The encoder accesses the frame memory through five paths:

Path1: Writing the original video frame.
Path2: Reading the original video frame for encoding.
Path3: Writing the reconstructed anchor frame for future reference.
Path4: Reading the luminance data of reference frames for motion estimation.
Path5: Reading the chrominance data of reference frames according to motion vector(s) for motion compensation.

This paper employs three methods shown as follows to reduce the ME computations and frame memory accesses in path3 and path4, which are responsible for around 61% of the total frame memory access when the search range is −16 to 15 pixels.

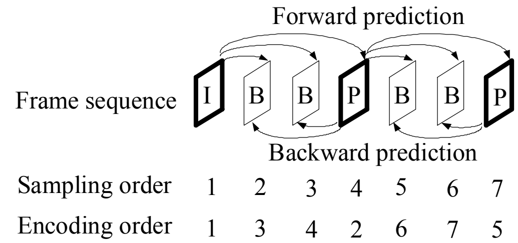

Figure 1. Sampling and encoding orders

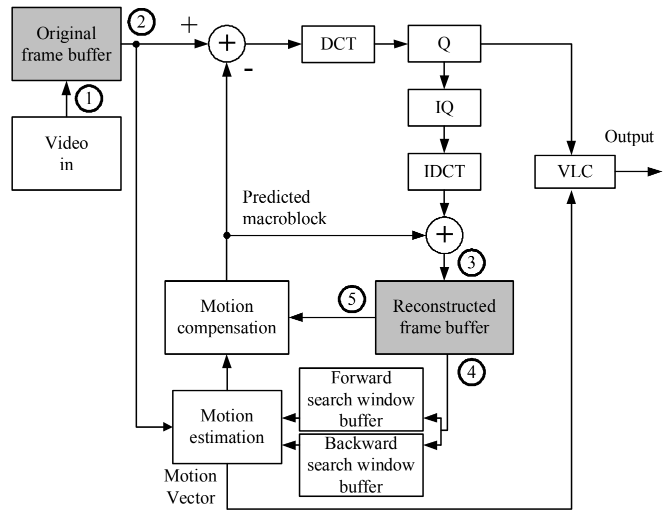

Figure 2. MPEG-2 encoding architecture

*1) Avoiding unnecessary writing in pathe3:* The reconstructed frame buffer is recycled to store a new anchor frame. Consequently, the new reconstructed macroblock of an anchor frame always overwrites a macroblock (i.e., victim macroblock) belonging to a previous anchor frame (i.e., victim frame). The victim macroblock in the buffer is completely reusable when these two macroblocks are exactly the same. This means that it is unnecessary to write the macroblock into the buffer.

Figure 3 gives an example illustrating how to identify the reusable victim macroblock. The four frames ($I_0$, $B_1$ ,$B_2$ and $P_3$) and the upper part of the current frame $P_6$ are assumed to be completely encoded. The frame memory stores two anchor frames: the victim frames $I_0$ and the reference frame $P_3$. The current macroblock $m_c$ will overwrite the victim macroblock $m_v$. If $m_c$ and co-located macroblock $m_f$ in $P_3$ are both stationary ($m_c = m_f = m_v$), then $m_v$ in the frame buffer is reusable.

A macroblock is stationary if its motion vector and the luminance and chrominance residual are all equal to zero. The "stationary-information" of the current macroblock $m_c$ is obtained from the ME and Q (quantization) units without extra computations while the "stationary-information" of the macroblock $m_f$ is obtained from a macroblock bitmap (MBB)

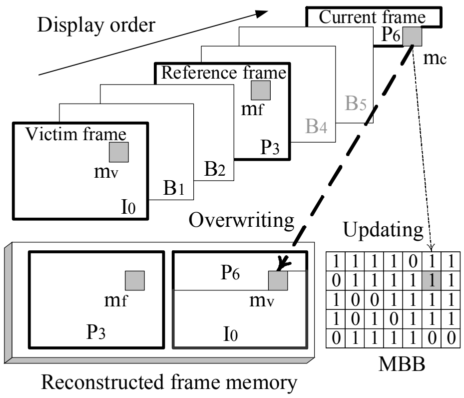


Figure 3. Detection of reusable macroblocks through the use of a macroblock bitmap.

which is used to keep track of the status of macroblocks in the last reconstructed anchor frame. An MBB has as many cells as the number of macroblocks in a frame. The cell is set to one if the corresponding macroblock is stationary; otherwise, it is set to zero. When the macroblock of an anchor frame is reconstructed, the corresponding cell is looked up to determine whether the victim macroblock is reusable or not and then the MBB is updated according to the state of the current macroblock.

The occurrence of stationary macroblocks not only depends on the inherent video sequence, but also on the encoder-defined quantization parameter (QP). The QP controls quantization (Q) step size and inverse quantization (IQ). Using a larger QP to encode a video sequence will result in lower bit-rate, poorer image quality, but more stationary macroblocks.

*2) Avoiding unnecessary reading in path4:* Motion estimation eliminates the temporal redundancy in successive video frames. Block-matching algorithm, such as the three-step-search algorithm [15] and diamond search algorithm [16], is a popular motion estimation method due to its simplicity. Block matching requires internal buffer to store the current macroblock and reference macroblocks; forward and backward search window buffers, as shown in Figure 2, are employed to hold the search area data in the previous and future reference frames, respectively. The search range is assumed to be −16 to 15 pixels in this paper and each search window buffer size is 3 Kbyte (16 × 16 × 12) [17].

To encode a macroblock in a B-frame, the motion estimation unit needs to transfer nine luminance macroblocks of the previous reference frame to the forward search window buffer for forward prediction. Because two adjacent macroblocks share six reference macroblocks, as shown in Figure 4, only three luminance macroblocks are transferred. Then for backward prediction, another three luminance macroblocks of the future reference frame are transferred to the backward search window buffer. This transfer, however, is

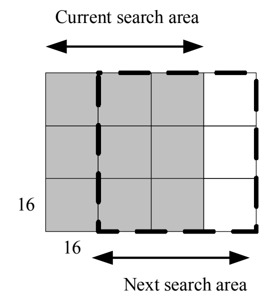


Figure 4. Search area overlapping in a search window buffer (search range: −16 ~ +15).

redundant if these three luminance macroblocks are stationary. This is because the three co-located macroblocks stored in the forward search window buffer can be reused for backward prediction.

To avoid unnecessary data transfers in path4, when the reference macroblock of the previous reference frame is read from the frame memory, the encoder looks up the MBB which records the stationary information of the future reference frame, i.e., the last reconstructed anchor frame. The transferred macroblock (of the previous reference frame) to the forward search window buffer is broadcast to the backward search window buffer if the co-located macroblock in the future reference frame is stationary; namely stationary macroblocks in the future reference frame will not be transferred.

*3) Avoiding redundant ME computations:* Computations for backward motion estimation are redundant if the nine reference macroblocks in the future reference frame are all stationary. These redundant computations can be reduced by looking up the MBB.

## IV. SIMULATION RESULTS

This work uses an MPEG-2 main profile at low level (MP @ LL) video coding environment to evaluate the performance of the proposed approach. The three-step-search method with search range [−16, +15] is adopted in this paper. Experiments run on six video sequences with a frame size of 352×288 pixels in 4:2:0 format, setting the frame pattern in IBBPBBP. Performance comparisons are done on the first 300 frames. The Intra-period is 30 frames.

Table I and II show the percentage of reduced frame memory access and ME computation with different QP values. Obviously, the higher the QP value is, the more the reduced frame memory accesses and ME computations are. For video sequences *Akiyo*, *Hall monitor*, *Container ship* and *News*, on the average, an 18.4% reduction in frame memory accesses and a 16.1% saving in ME computations have been achieved when the QP value is equal to sixteen. The previous four sequences have much lower spatial frequency and movement, leading to larger amount of stationary macroblocks in the P-frame. Thus, our approach is especially effective for low motion videos.

1-4244-1251-X/07/$25.00 2007 IEEE.

Table I. The percentage of reduced frame memory accesses.

| Benchmark | % of reduced frame memory acceses | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | In path3 | | | In path4 | | | In five paths | | |
| | QP=8 | QP=16 | QP=24 | QP=8 | QP=16 | QP=24 | QP=8 | QP=16 | QP=24 |
| Akiyo | 57.3 | 62.6 | 63.6 | 27.3 | 29.8 | 30.5 | 18.4 | 20.1 | 20.5 |
| Hall monitor | 17.6 | 58.2 | 64.7 | 12.7 | 28.7 | 30.9 | 8.1 | 19.2 | 20.8 |
| Container ship | 25.2 | 50.0 | 58.7 | 15.5 | 25.9 | 28.8 | 10.0 | 17.2 | 19.3 |
| News | 44.5 | 51.4 | 53.9 | 21.8 | 25.4 | 26.6 | 14.6 | 17.0 | 17.8 |
| Foreman | 2.0 | 7.2 | 8.9 | 2.1 | 5.7 | 7.0 | 1.3 | 3.6 | 4.4 |
| Coastguard | 0 | 1.4 | 4.6 | 0.1 | 1.7 | 4.2 | 0.1 | 1.0 | 2.6 |

Table II. The percentage of reduced ME computations.

| Benchmark | % of reduced ME computations | | | | | |
|---|---|---|---|---|---|---|
| | For B frames | | | For all frames | | |
| | QP=8 | QP=16 | QP=24 | QP=8 | QP=16 | QP=24 |
| Akiyo | 24.3 | 27.4 | 27.3 | 19.4 | 21.9 | 21.8 |
| Hall monitor | 0.3 | 18.2 | 24.1 | 0.2 | 14.6 | 19.3 |
| Container ship | 5.2 | 15.5 | 20.8 | 4.1 | 12.4 | 16.7 |
| News | 15.7 | 19.2 | 20.0 | 12.6 | 15.3 | 16.0 |
| Foreman | 0.1 | 0.4 | 0.8 | 0 | 0.3 | 0.6 |
| Coastguard | 0 | 0 | 0.2 | 0 | 0 | 0.1 |

## V. CONCLUSION

In this paper, we have proposed a scheme to reduce unnecessary frame memory accesses and redundant ME computations for the MPEG-2 video encoders. Simulations of the proposed method have shown that the approach can eliminate around one fifth of the frame memory accesses as well as 16% of the ME computations for low motion videos, without affecting the image quality. Through the experiments, we found that the inherent video features have made the proposed scheme quite attractive to use in power-aware video encoders for embedded applications.

## ACKNOWLEDGMENT

The work in this paper is in part supported by the National Science Council, Taiwan, under NSC 94-2220-E-006-004.

## REFERENCES

[1] D. Wu, Y. T. Hou, and Y.-Q. Zhang, "Transporting Real-Time Video over the Internet: Challenges and. Approaches," *Proceedings of the IEEE*, Vol. 88, No. 12, pp.1855-1874, December 2000.

[2] H. Kim and I.-C. Park, "High-performance and low-power memory-interface architecture for video processing applications," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 11, No. 11, pp. 1160-1170, November 2001.

[3] L. Zhang, Z. Fang, M. Parker, B. K. Mathew, L. Schaelicke, J. B. Carter, W. C. Hsieh, and S. A. McKee, "The Impulse Memory Controller," *IEEE Transactions on Computers*, Vol. 50, No. 11, pp. 1117-1132, November 2001.

[4] E. Brockmeyer, L. Nachtergaele, F. Catthoor, J. Bormans, and H. J. De Man, "Low Power Memory Storage and Transfer Organization for the MPEG-4 Full Pel Motion Estimation on a Multimedia Processor, " *IEEE Transactions on Multimedia*, Vol. 1, No. 2, pp. 202-216, June 1999.

[5] N. Kavvadias, A. Chatzigeorgiou, N. Zervas, and S. Nikolaidis, "Memory Hierarchy Exploration for Low Power Architectures in Embedded Multimedia Applications," *International Conference on Image Processing (ICIP)*, pp. 326-329, October 2001.

[6] Z. Xu , S. Sohoni , R. Min, and Y. Hu, "An Analysis of Cache Performance of Multimedia Applications," *IEEE Transactions on Computers,* Vol. 53, No.1, pp.20-38, January 2004.

[7] A. Ramachandran and M. F. Jacome, "Energy-Delay Efficient Data Memory Subsystems," *IEEE Signal Processing Magazine*, pp. 23-37, May 2005.

[8] M. Kandemir, J. Ramanujam, M. J. Irwin, V. Narayanan, I. Kadayif, and A. Parikh, "A Compiler-Based Approach for Dynamically Managing Scratch-Pad Memories in Embedded Systems," *IEEE Transactions on Computer-Aided Design,* Vol. 23, No. 2, pp. 243-260, February 2004.

[9] I. Issenin, E. Brockmeyer, M. Miranda, and N. Dutt, "Data Reuse Analysis Technique for Software-Controlled Memory Hierarchies," *Proceedings of the Design, Automation and Test in Europe Conference and Exhibition (DATE)*, Vol. I, pp. 202-207, February 2004.

[10] T. Migita and V. G. Moshnyaga, "An Algorithmic Enhancement for Reducing Computations of Bidirectional Motion Estimation," *IEEE International Workshop on Signal Processing Systems*, pp. 669-672, November 2005.

[11] J.-C. Tuan, T.-S. Chang, and C.-W. Jen "On the Data Reuse and Memory Bandwidth Analysis for Full Search Block Matching VLSI Architecture," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol.12, No.1, pp.61-72, January 2002.

[12] Y.-K. Lai, Y.-L. Lai, Y.-C. Liu, P.-C. Wu, and L.-G. Chen, "VLSI Implementation of the Motion Estimator with Two-Dimensional Data-Reuse," *IEEE Transactions on Consumer Electronics,* Vol. 44, No.3, pp. 623-629, August 1998.

[13] C.-M. Chen and C.-H. Chen, "A Memory Efficient Architecture for Deblocking Filter in H.264 Using Vertical Processing Order," *IEEE International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP),* Melbourne, Australia, pp. 361-366, December 5-8, 2005.

[14] C.-W. J. Shih, N. Ling, and T. Ogunfunmi, "Memory Reduction by Haar Wavelet Transform for MPEG Decoder," *IEEE Transactions on Consumer Electronics*, Vol. 45, No. 3, pp. 867-873, August 1999.

[15] R. Li, B. Zeng, and M. L. Liou, "A New Three-Step Search Algorithm for Block Motion Estimation," *IEEE Transaction on Circuits and Systems for Video Technology,* Vol. 4, pp. 438-442, August 1994.

[16] S. Zhu and K.K. Ma, "A New Diamond Search Algorithm for Fast Block-Matching Motion Estimation," *IEEE Transactions on Image Processing,* Vol. 9, pp. 287-290, February 2000.

[17] H. M. Jong, L. G. Chen, and T. D. Chiueh, "Parallel Architectures for 3-Step Hierarchical Search Block-Matching Algorithm", *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 4, No. 4, pp. 4077-416, August 1994.