Motion vector memory reduction scheme for scalable motion estimation

Jinha Choi and Jaeseok Kim
Yonsei University, Department of Electrical and Electronics Engineering, Shinchondong Seodaemungu 134, Seoul 120-749, Korea
E-mail: b815@yonsei.ac.kr

Abstract. We propose a motion vector memory reduction scheme for the H.264 spatial scalable motion estimation. The spatial scalable prediction of H.264 scalable video coding requires a significant amount of motion vector bits from the previous layer motion prediction. This motion vector causes the Internal memory size to increase, which may result in increasing hardware cost and power consumption. To reduce the memory size of the motion vector, we propose a motion vector bit-compression scheme of the interlayer motion estimation. The proposed compression scheme uses the difference value of current motion vector and previous motion vector to get a probability for entropy coding. In addition, the proposed scheme modifies a variable length coding table of the H.264 system as a simple entropy coding table. The proposed scheme reduces the motion vector–storing bit by ~66.5% with less changes in hardware size.

[DOI: 10.1117/1.3212689]

Subject terms: motion vector coding; scalable video coding; motion estimation; H.264.

Paper 090353LR received May 20, 2009; revised manuscript received Jul. 4, 2009; accepted for publication Jul. 10, 2009; published online Sep. 2, 2009.

1 Introduction

Scalable video coding (SVC) is a new design that has been standardized as an extension of the H.264 standard. SVC has many scalable skills to adopt many channel types of transmission layers.[1] Temporal scalability, spatial scalability, and quality scalability are the main scalable features of the SVC. Spatial scalability is the key to coded image quality. Spatial scalability is divided into intra and interlayer prediction. Intralayer prediction is based on the H.264 standard prediction, and interlayer prediction is an additional prediction for the spatial scalable prediction. Residual prediction and motion prediction are the most complex predictions in interlayer prediction. These methods need temporal information that contains motion vector information of the lower layer. This information causes an increase of internal data transaction, power consumption, and internal memory size. This paper proposes a motion vector–storing bit-compression scheme for internal memory reduction. The proposed scheme compresses internal motion vector memory for enhancement layer motion prediction without a hardware increment.

2 Spatial Scalable Motion Prediction

H.264 SVC follows the conventional approach of multilayer coding. In each spatial layer coded, motion compensation or intraprediction coding is employed as single-layer coding, which is called the intralayer prediction. However, in order to improve coding efficiency, the additional interlayer prediction is included. In the interlayer motion prediction, motion vector information of the base layer is referred to the enhancement-layer motion estimation when the encoding mode is the base-layer mode or the refinement mode. The estimated motion vector information in the previous layer is stored by bits that depend on the search range. For example, when the search range is +64 to −64, one bit for the signed bit, six bits for integer motion vector bit, and two bits for the fractal motion vector bit. Thus, nine bits for the x-axis motion vector value and another nine bits for the y-axis motion vector value are stored. Up to 16 motion vectors at one macroblock are used when all submacroblocks are in 4×4 block mode. Therefore, motion vector bits are highly increased and make the motion vector–storing memory area increase. For example, there is a maximum of 57,600 motion vectors in the high definition [(HD), 1280×720] resolution, and 20 bits per motion vector are required when the search range is +128 to −128. Therefore, a 1152×10^3−bit memory area is used for the enhancement-layer prediction. These motion vector bits make memory storage area overhead and many memory transactions. Extended to full HD (1920×1080) resolution, motion vector bit information is much higher. To reduce the temporal motion vector bit, the compression method is required.

H.264 adopts the motion vector coding (MVC) with the motion vector prediction (MVP) for MVC bit reduction. This scheme is based on motion vector characteristics. Motion vectors have a tendency that is a similar value to neighbor motion vectors. The current motion vector is compared to three motion vectors that are located at the top, top right, and left, and the smallest variation of these is selected for prediction.[2] The difference of motion vector values goes near to zero, and these values can be compressed with entropy coding. However, this MVC with the MVP method needs additional memory for previous motion vector information.[3] Motion vectors of the upper line and the left motion vector are referred to the current macroblock line. For example, a maximum of 320 motion vectors are needed when the video resolution is the HD for the current macroblock MVC, and they only hold the motion vector at the current macroblock line. For this reason, the MVC is too big for internal motion vector bit reduction. To reduce memory transaction in the MVC, additional memory and additional processing are needed. Therefore, there is less hardware gain for bit reduction. Thus, the motion vector bit compression scheme will be a simple hardware size with good compression quality.

3 Proposed Motion Vector Memory Reduction Scheme

The proposed motion vector bit compression scheme also bases on motion vector tendency. First, the proposed scheme divides motion vectors to integer motion vectors and fractal motion vectors because integer motion vectors and fractal motion vectors have different probabilities. The case of zero motion in the integer motion vector is higher in probability than other motion cases. However, the fractal motion vector is nearly even in distribution probability of

0091-3286/2009/$25.00 © 2009 SPIE

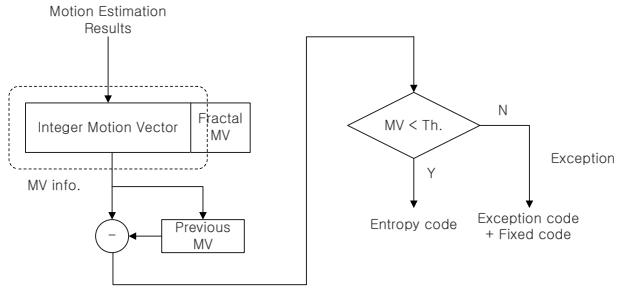

Fig. 1 Detail processing of the proposed scheme.

each fractal motion value. Therefore, the proposed scheme compresses the integer motion vector bit as variable-length bit coding and fixed bit length for the fractal motion vector. In addition, the proposed method compresses the difference of the current motion vector and previous motion vector.

The H.264 system has entropy coding methods. The MVC also uses the signed Ex-Golomb table for motion vector compression.[4] However, these entropy coding methods have long coding table length. For low hardware increments, the proposed compression modifies the signed Ex-Golomb table, which is a simple part of the table, because the unlimited Ex-Golomb table size causes hardware overhead. Thus, the proposed scheme restricts the table size for highly probable motion vector values. When the table codes are bigger than threshold value, the proposed scheme does not follow the end of long Ex-Golomb table value and the proposed coding takes exception code and the fixed code of the original motion vector. The threshold value of the proposed coding is defined by the length of the original motion vector, where it is below the length of fixed code motion vector bits. Figure 1 shows the detail processing of the proposed scheme.

Table 1 is an example of the proposed modified signed Ex-Golomb table when the required motion vector bit is 8. When the motion vector value is 9, a coded motion vector bit "000010000," which is designated to the exception code in the table and the bit "00001001," the fixed coding motion vector bit, is added. However, this exception case is a very low probable case. Most motion vectors are represented in less than half of bit cases. Therefore, using this algorithm, integer motion vector bits can be compressed about 50–60%. The proposed scheme operates only one subtraction with the simplified Ex-Golomb table. Thus, the proposed method has lower hardware size increments for temporal motion vector compression than the MVC.

Table 1 Modified signed Ex-Golomb coding table.

Code number	Code (length)	Syntax element	Modified SE
0	1 (1)	0 (MV)	0 (Integer MV)
1	010 (3)	1	1
2	011 (3)	−1	−1
..	..	..	..
14	0001111 (7)	−7	−7
15	000010000 (9)	9	Exception code

Table 2 Compression ratio of the proposed compression scheme.

Sequence	Original	Proposed	Compression ratio (%) (integer only)
Bridge-close	366k	111k	69.51 (89.51)
Mobile	390k	142k	63.63 (83.63)
Paris	373k	124k	66.81 (86.81)
Average	(Bits)	(Bits)	66.49 (86.49)

4 Performance Analysis and Complex Comparison

To verify the performance of the proposed motion vector compression, we execute the motion estimation process with CIF (352×288) resolution video sequences and process on the +128 to −128 search ranges, where motion vector bits are 20 bit per each pair. Fourteen kinds of the video sequences are simulated that are Bridge-Close, Coast-guard, Container, Flower, Foreman, Hall, Highway, Mobile, Mother & daughter (Mother), News, Paris, Silent, Tempete, and Waterfall. These sequences have variable motion cases that are from low motion cases to high motion cases. One-hundred frames of the sequence are simulated with each sequence where the quantization parameter (QP) is 28. Table 2 shows the bit compression ratio of the simulated sequence. Motion vector bits are the average of 99 interprediction frame motion vector bits.

As shown in Table 2, the proposed motion vector compression algorithm reduces total motion vector bits by ~66.5% and reduces integer motion vector bits by ~86.49%. The low motion sequence is more compressed than the high motion sequence because <5 bits motion vector cases (low motion vector values) occur more frequently in video sequences.

Figure 2 is the example of the bit comparison when various QP cases in Silent (low motion), News, and Tempete (high motion) sequences, where QP is 20 (low QP), 24, 28, 32, and 36 (high QP). Following Fig. 2, the proposed method reduces motion vector bits when QP is higher because high QP has more low motion vector cases than low QP.

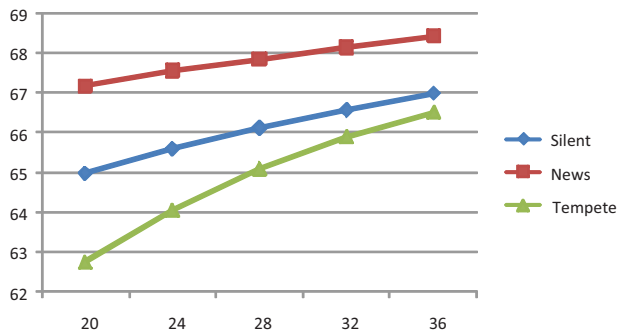

Fig. 2 Compression ratio comparison with 5 QP types.

Table 3 Comparison between the standard MVC and the proposed.

	Standard MVC	Proposed
Compression	High (83.87%)	High (66.49%, 86.49%)
Hardware Increment	Two additional memory	No additional memory
	Full-size signed table	Simplified size table

Table 3 shows the comparison between the proposed method and the MVC of the H.264. The compression ratio of the MVC is simulation results of the same condition. The MVC is also a high-compression scheme. However, the MVC needs two additional memories (one for base-layer encryption, the other for enhancement-layer decryption) and the full-sized Ex-Golomb coding table. Despite the high-compression ratio, the extra memory units make motion vector memory reduction gain low because the MVC mainly considers bit-stream reduction more than hardware complexity. On the other hand, the proposed compression method has no internal memory increment and has the simplified Ex-Golomb table, which is considered for hardware overhead. Therefore, the proposed method is a more attractive compression method for internal motion vector memory reduction.

5 Conclusion

In the interlayer motion prediction, motion vector information of the base layer is referred. Because motion vector information is a large amount of bits to store in memory, motion vector bits are needed to be compressed. The proposed method is based on entropy coding. Difference value of the previous motion vector is coded with the modified signed Ex-Golomb table. The proposed method reduces motion vector bits by $\sim$66.5% without hardware increment. The MVC also highly reduces motion vector bits. However, the MVC needs two additional memory areas for MVP and the full-size Ex-Golomb table, which make motion vector bit-reduction gain low. Therefore, the proposed algorithm is advanced for hardware implementation because of low memory area and low data transaction, which is related to power consumption.

Acknowledgments

This research was supported by the Ministry of Knowledge Economy (MKE), Korea, under the ITRC support program supervised by the IITA and "System IC 2010" project of the MKE, Korea.

References

1. H. Schwarz, D. Marpe, and T. Wiegand, "Overview of the scalable video coding extension of the H.264/AVC Standard," *IEEE Trans. Circuits Syst. Video Technol.* **17**(9), 1103–1120 (Sep. 2007).
2. M. Chen, J. Willson, and A. N. Willson, "A spatial and temporal motion vector coding algorithm for low-bit-rate video coding," in *Int. Conf. on Image Processing*, Vol. **2**, pp. 791–794 (Oct. 1997).
3. R. Wang, J. Li, and C. Huang, "Motion compensation memory access optimization strategies for H.264/AVC Decoder," in *IEEE Int. Conf. on Acoustics, Speech, and Signal Processing*, Vol. **5**, pp. v97–v100 (Mar. 2005).
4. W. Di, G. Wen, H. Mingzeng, and J. Zhenzhou, "An Exp-Golomb encoder and decoder architecture for JVT/AVS," in *Int. Conf. on ASIC*, Vol. **2**, pp. 910–913 (Oct. 2003).