Fast Block-Size Partitioning Using Empirical Rate-Distortion Models for MPEG-2 to H.264/AVC Transcoding

Qiang Tang, Panos Nasiopoulos, Rabab Ward
University of British Columbia, Vancouver, BC, Canada
{qiangt, panosn, rababw}@ece.ubc.ca

***Abstract*—We present an efficient H.264/AVC block-size partitioning prediction method, which is based on our proposed empirical rate and distortion models. Compared to other state-of-the-art transcoding methods, and for the same rate-distortion performance, our proposed algorithm requires the least computational complexity, reaching a 73% reduction in variable block-size motion estimation for SDTV sequences, and 71% reduction for CIF sequences.**

I. INTRODUCTION

MPEG-2 is the video coding standard widely used in digital television (DTV) broadcasting industry and digital versatile disc (DVD) applications. Nevertheless, H.264/AVC, the latest video coding standard, is quickly gaining ground in the video industry due to its higher compression efficiency. Compared to previous standards, H.264/AVC increases the video compression efficiency by about 50% while maintaining the same picture quality [1]. Since the two standards (MPEG-2 and H.264/AVC) are destined to coexist for some time, providing universal multimedia access between them is becoming a hot research area.

Video transcoding provides universal multimedia access among different standards by converting videos from one format to another [2]-[3]. The main objective of a video transcoder is to utilize the existing coding information in one video format to improve the encoding efficiency of creating another video format.

Most existing transcoding schemes for MPEG-2 to H.264/AVC address the acceleration of the motion re-estimation process for H.264/AVC encoding by using some coding information existing in MPEG-2 videos [4]-[6]. The computational complexity of the motion re-estimation process during transcoding could be significantly high due to some new features introduced by H.264/AVC. One of them known as variable block-size motion estimation (VBSME), allows one 16x16 macroblock to have different block-size partitions and each partition to have its own motion vector [7]. Searching the motion vector for each partition in VBSME takes a large amount of time. The *MotionMapping* scheme proposed in [4] reduces the search range of finding the motion vector for each partition. The schemes in [5] and [6] propose algorithms for making fast block-size partitioning decisions. The approach in [5] focuses on 16x16 and 8x8 block sizes, while [6] is designed to decide whether or not an MPEG-2 inter-coded macroblock should be encoded as an Intra macroblock in H.264/AVC. From all the above, *MotionMapping* achieves the best balance between picture quality and computational complexity. However, this method fails to consider the use of a fast block-size partitioning prediction method which will further reduce the overall complexity.

In this paper we propose an efficient H.264/AVC block-size partitioning algorithm for transcoding from MPEG-2 to H.264/AVC. Our algorithm uses our proposed empirical rate/distortion models as well as *MotionMapping* to predict a block-size partitioning choice for H.264/AVC. The result is, a significant reduction in the computational complexity of the VBSME process. Experimental results show that, for the same picture quality as that of the *MotionMapping* scheme, our proposed algorithm reduces the computation of the VBSME process by 73% for SDTV sequences and 71% for CIF sequences. Compared to the full-search scheme, as a reference point of picture quality, our algorithm reduces the computational complexity by about 99.47% for SDTV sequences and 98.66% for CIF sequences.

The rest of the paper is structured as follows. Section II describes our proposed block-size partitioning prediction algorithm. Section III presents the experimental results and the conclusion is drawn in Section IV.

II. PROPOSED ALGORITHM FOR BLOCK-SIZE PARTITIONING PREDICTION

The block size partitioning choice in H.264/AVC selects the best block size for motion compensation from a set of 7 block sizes (i.e., 16x16, 16x8, 8x16, 8x8, 8x4, 4x8, and 4x4). When the chosen block size is smaller than 16x16, more than one partition (block) is created for the 16x16 MB, each of which has its independent motion vector. Choosing the best block size could be a computationally expensive process.

In MPEG-2 to H.264/AVC transcoding, the block size partitioning selection process can be accelerated by using

This work was supported by grants from the Natural Sciences and Engineering Research Council of Canada (NSERC)

some coding information existing in the pre-encoded MPEG-2 videos. For instance, using the MPEG-2 motion vectors (MVs) as initial motion vectors for H.264/AVC encoding, significantly reduces the search range for estimating the final motion vectors. Using these initial motion vectors, an accurate block size partitioning choice can be achieved through rate-distortion optimization techniques. In our work, we propose to predict the block size partitioning choice for MBs in P/B frames using *Lagrangian* techniques. In this case, the block size partitioning choice aims at minimizing the following *Lagrangian* cost $J = D + \lambda \times R$ (with D measuring the picture quality of the reconstructed video obtained after encoding and decoding the original video, R measuring the bits needed to encode the corresponding MB, and λ being the *Lagrangian* multiplier which is related to the quantization parameter). Our proposed algorithm consists of two steps which are explained in the following two sub-sections.

A. Determining the Initial Motion Vector

The first step is to find the initial motion vector which will be used as the initial search point to perform the motion-vector refinement in encoding process. The *MotionMapping* scheme proposed in [4] derives the initial search point by using the motion vectors of the surrounding MPEG-2 MBs. Note that in our method, if the MPEG-2 video streams are interlaced, the average of motion vectors from the top and bottom fields is used as the motion vector for the current MB. In addition to those vectors, we also consider the H.264/AVC predicted motion vector as another candidate for the initial search point. Furthermore, for the B frames, we propose to initialize forward-predicted and backward-predicted MVs for every MPEG-2 MB with the invalid MV set to zero and the valid MV equal to the original MPEG-2 MV. This initialization procedure enables us to derive an initial search point for each prediction mode (forward, backward or bi-directional prediction) using surrounding MPEG-2 MVs.

Since in our algorithm there is more than one candidate to be considered as the initial search point, we use the *Lagrangian* cost (J_{MOTION}) to determine which one to choose as follows:

$$\min\{J_{\mathrm{MOTION}}(S_k,V_k|QP)\} = \min\{SATD(S_k,V_k) + \lambda_{\mathrm{MOTION}} R_{\mathrm{MOTION}}(S_k,V_k)\} \quad (1)$$

where S_k represents the index of MBs in a video frame, and V_k represents the set of initial search points which include the ones derived using surrounding MPEG-2 MVs, and the ones equal to the H.264/AVC predicted MVs. *SATD* is the sum of absolute transformed differences between the original MB and the predicted MB. R_{MOTION} stands for the bits of encoding the corresponding motion vectors. The initial search point which yields the least cost is chosen as the best initial motion vector.

Four block-size partitions are used in our transcoding scheme. Each partition will find its own initial search point in this step. After that, a small search range suffices to find the final motion vector for each partition [4]. The next step is to decide which partition should be chosen as the final macroblock encoding mode.

B. Proposed Block-Size Partitioning Prediction Using Empirical Rate and Distortion Models

In our proposed algorithm, only four block-size partitions are used, i.e., 16x16, 16x8, 8x16 and 8x8. Our performance evaluation shows that the block-size partitions below 8x8 do not give justified improvement in terms of compression performance compared to the additional computational complexity they add to the overall transcoding system.

Theoretically, the *Lagrangian* cost J obtained using accurate D_{REC} and R_{REC} should be used to perform the block-size partitioning choice (or mode decision) in H.264/AVC. However, calculating the accurate D_{REC} and R_{REC} requires significant computational efforts. For instance, calculating the accurate R_{REC} requires a complete encoding process, including transform, quantization and entropy coding. This means we have to completely encode each MB four times (one for each partition), although only one partition is finally chosen. This method, thus, is computationally expensive. To reduce computational complexity, we propose to predict the block-size partitioning choice using our new rate-distortion model.

This rate-distortion model is based on the rate and distortion models proposed in [8], with some necessary modifications for adapting the general cases to the specific needs of transcoding from MPEG-2 to H.264/AVC.

The updated distortion model which measures the distortion in terms of PSNR in our algorithm is given by:

$$D_{\mathrm{REC}}(S_k,V_{\mathrm{INIT}}|QP)=b_1\times\log_{10}((SATD(S_k,V_{\mathrm{INIT}}))/256+1)\times\mathrm{QP}+b_2 \quad (2)$$

where b_1, b_2 are model parameters. V_{INIT} stands for the initial motion vector and $SATD(S_k,V_{\mathrm{INIT}})$ is the sum of absolute transformed differences between the original video blocks and predicted video blocks. This *SATD* is obtained right after the motion compensation. The computational complexity is reduced compared to computing the *SATD* between the original and reconstructed videos, a common practice used in other fast rate-distortion optimization methods. The reason of using *SATD* instead of *SAD* is because *SATD* usually yields better results in H.264/AVC rate-distortion optimization.

Using *SATD* also makes the updated distortion model different from the original distortion model. In the original model, the $SATD(S_k,V_{\mathrm{INIT}})/256$ is replaced by the mean of absolute difference (MAD) which is estimated using the MADs in the previous frames. One drawback in this case is that when the current frame has local scene changes, the estimation may become inaccurate. For this reason, we use the initial motion vectors of the current MB to calculate the *SATD* value.

Theoretically, the *SATD* calculation should be based on the final motion vector of each block-size partition. However, in MPEG-2 to H.264/AVC transcoding, since the final motion vector of H.264 could be obtained through a small range of search ([-1.75 1.75]) starting from the initial search point, the probability of the initial motion vector being very close to the final motion vector is very high. For this reason, our distortion model that uses the initial motion vectors to estimate the *SATD* of different block-size partitions ends up yielding very good results.

For this updated distortion model, the updated rate model of our proposed algorithm is given by the following equation:

$$R_{REC}(S_k, V_{INIT}|QP) = c_2 \times (SATD(S_k, V_{INIT}))/256 \times 2^{-QP/6} \quad (3)$$

where c_2 is the model parameter and is video-content dependent (the choice of c_2 is explained later in this section). As in the distortion model, instead of estimating the MAD from previous frames, we calculate $SATD(S_k,V_{INIT})/256$ by using the initial motion vectors of the current MB.

In [7], the rate model does not take into account the bits needed for encoding motion vectors. However, in low bit-rate scenarios, the number of these bits could be a significant portion of total number of bits needed to encode the current MB. We estimate the bits needed for encoding the motion vectors of the different partitions as follows:

$$R_{MV}(S_k) = \sum_{i=1}^{N} \left[\, |\max(MVX_i(S_k), MVY_i(S_k))| + a \right] \quad (4)$$

where N is the number of motion vectors in each partition (N = 1 for 16x16 and N = 4 for 8x8), a is a constant equal to 1 which compensates for zero-value motion vectors. MVX_i and MVY_i represent the differences between the original motion vector and the predicted motion vector in the x and y axis, respectively.

Based on equations (2), (3) and (4), the new cost function for block-size partitioning choice is formed as below:

$$J_{MODE}(S_k,V_{INIT}|QP) = -D_{REC}(S_k,V_{INIT}|QP) + \lambda_{MODE} \times [R_{REC}(S_k,V_{INIT}|QP) + R_{MV}(S_k)] \quad (5)$$

The best block partition is chosen to be the one that yields the least J_{MODE}. Note that since D_{REC} measures the distortion in PSNR, we use the negative value of $D_{REC}(S_k,V_{INIT}|QP)$ to form the minimization problem (lower $-D_{REC}$ corresponds to lower distortion). Therefore, minimizing J_{MODE} results in a combination which minimizes the distortion and the bit-rate together. In Equation (5), the original λ_{MODE} is designed when the distortion is measured using the sum of squared error (SSE). Since our distortion is measured in PSNR, the λ_{MODE} in our model should be different from the original one. Since $\lambda = dD/dR$ (D refers to distortion and R refers to rate), the λ_{MODE} in our model is simplified as the following equation:

$$\lambda_{MODE} = \frac{6 \times b_1 \times 2^{QP/6}}{\log(2) \times c_2} \quad (6)$$

In order to calculate J_{MODE}, besides knowing $SATD(S_k, V_{INIT})$ and QP, we also need to know the model parameters b_1, b_2, and c_2. In the previous model, the values of these parameters are video-content dependent. However, we propose to fix the values of these parameters in our models. When we calculate J_{MODE} for different block-size partitions, the calculations take place within the same macroblock. In this scenario, b_1, b_2, and c_2 are constants. As a result, by knowing the values of $SATD(S_k, V_{INIT})$ and QP, we can determine which block-size partition gives the least J_{MODE}. In other words, there is no need to calculate the absolute value of J_{MODE} since the relative value suffices finding the least cost. Suggested practical values of b_1, b_2 and c_2 are –0.52, 47 and 64, respectively. Note that c_2 cannot be too small, otherwise the value of $R_{REC}(S_k, V_{INIT}|QP)$ will always be a decimal number, which cannot give a correct value for J_{MODE}.

TABLE I H.264/AVC ENCODING SETTINGS.

Experimental settings of H.264/AVC encoding	
Search Range	±32 (CIF), ±64 (SDTV)
No. of Frames	300
Profile	Main
Level	2.0 (CIF), 3.0 (SDTV)
Reference B	Disabled
Deblocking Filter	Enabled
Entropy Coding	Context-Adaptive Binary Arithmetic Coding (CABAC)

In summary, after applying our proposed algorithm, the motion re-estimation process contains three steps. The first step involves determining the best initial search point for finding the final H.264 motion vectors (Equation (1)). The second step is to predict the best block-size partition by minimizing J_{MODE} in Equation (5). After the best partition is chosen, the last step is to use the best initial search point to perform a [-1.75 +1.75] motion-vector refinement process to find the best H.264 motion vector.

III. EXPERIMENTAL RESULTS

The JM14.2 H.264/AVC reference software codec was used in our implementation [9]. As for the MPEG-2 video sequences, we used several real-life digital TV broadcasting video streams (standard DTV with resolution equal to 720x576 or 704x480), which were encoded using a commercial MPEG-2 encoder. These DTV streams were compressed at high bit-rates, ranging from 3Mbits/sec to 8 Mbits/sec, to ensure high picture quality. These streams represented content that was widely used in real-life applications, such as movies, commercials, music videos and sports scenes. The SDTV streams are interlaced and, thus, the interlace-to-progressive conversion is applied before the streams go through the H.264 encoding process.

Besides the SDTV video streams, several CIF sequences were also tested in our experiments. These CIF sequences were encoded into MPEG-2 video streams by using the official MPEG-2 reference software (TM5). The group of pictures (GOP) structure is set as IPPP. The length of GOP is 15. The bit-rate of the CIF MPEG-2 videos streams is 2 Mbits/sec. The encoding settings of H.264/AVC are listed in Table I.

We compare our proposed MPEG-2 to H.264/AVC transcoding scheme with two other schemes: 1) The *MotionMapping* scheme which uses the motion-mapping algorithm proposed in [4] and also takes advantage of rate distortion optimization (RDO). In our experiments, four block size partitions {16x16, 16x8, 8x16 and 8x8} are used. 2) The *Full-Search* scheme, which is the exhaustive full-search motion estimation scheme in the H.264/AVC encoder using all 7 block sizes and RDO. This scheme yields the best compression performance but also has the highest computational complexity. The reason for this comparison is that although *MotionMapping* has been compared with the

TABLE II DIFFERENCES BETWEEN OUR PROPOSED TRANSCODING SCHEMES AND OTHER SCHEMES.

Transcoding schemes	Search range of motion vectors	Available block size partitioning candidates	Motion estimation strategy
Our Proposed	±1.75	16x16 16x8 8x16 8x8	Full search for integer pixel refinement and *UMHexagonS* for sub-pixel refinement
Full-Search	±32 (CIF) ±64 (SDTV)	16x16 16x8 8x16 8x8 8x4 4x8 4x4	Exhaustive full search
MotionMapping	±1.75	16x16 16x8 8x16 8x8	Full search for integer pixel refinement and *UMHexagonS* for sub-pixel refinement

full-search scheme in [4], the rate-distortion optimization was disabled in those experiments. Therefore, such comparison is necessary for achieving a more accurate performance evaluation of our method.

Table II shows the main differences between our proposed transcoding scheme and the other two schemes in terms of coding parameters related to motion estimation. The extra coding parameters follow the common conditions proposed by the JVT group in [10]. The set of quantization parameters suggested in [10] is {22, 27, 32, 37} for I frames, {23, 28, 33, 38} for P frames, and {24, 29, 34, 39} for B frames.

Table III shows two comparisons: 1) the average PSNR differences between our proposed scheme and the other two schemes; 2) the execution time of the motion estimation process between the proposed and the other two transcoding schemes.

We observe that the proposed method achieves almost the same picture quality as that of the *MotionMapping* scheme. For the SDTV sequences, the differences are negligible (average 0.03 dB in BD PSNR values [11]). Regarding the CIF sequences, the difference is 0.06 dB on average. Compared to the exhaustive *Full-Search* scheme, the drop in picture quality of our transcoding scheme is about 0.28 dB for the SDTV sequences and 0.22 dB for the CIF sequences, a performance similar to that achieved by *MotionMapping*.

Regarding the execution time of the motion estimation process, we observe that for the SDTV sequences, and the search range equal to 64, our proposed algorithm can reduce the computational complexity by 99.47% compared to the *Full-Search* scheme. Compared to the *MotionMapping* transcoding scheme, our proposed scheme achieves almost the same rate-distortion performance, with the reduction of computational complexity reaching 71%. For the CIF sequences, and the search range equal to 32, our proposed algorithm reduces the computational complexity by 98.66% compared to the *Full-Search* scheme and 73% compared to *MotionMapping*.

IV. CONCLUSIONS

We presented an algorithm for transcoding from MPEG-2 to H.264/AVC which efficiently predicts H.264/AVC block-size partitioning. Our proposed transcoding scheme needs only one-time search within a 1.75 pixels window to find the final H.264/AVC motion vectors. We demonstrated that our proposed transcoding scheme requires the least computation efforts for VBSME compared to best performing existing technique (*Motion Mapping*) while achieving almost the same rate-distortion performance.

TABLE III AVERAGE PSNR DIFFERENCES AND COMPARISON OF MOTION ESTIMATION EXECUTION TIME BETWEEN DIFFERENT SCHEMES

	PSNR Differences		Comparison Of Motion Estimation Execution Time	
Test Sequences	*Proposed* vs. *MotionMapping*	*Proposed* vs. *Full-Search*	*Proposed* v.s. *MotionMapping*	*Proposed* v.s. *Full-Search*
Music Video (SDTV)	-0.03 dB	-0.16 dB	70.93%	99.57%
Commercial (SDTV)	-0.02 dB	-0.43 dB	70.24%	99.22%
Sports Scenes (SDTV)	0 dB	-0.17 dB	71.67%	99.66%
Movies (SDTV)	-0.06 dB	-0.36 dB	69.58%	99.43%
Football (CIF)	-0.07 dB	-0.26 dB	72.32%	98.80 %
Foreman (CIF)	-0.07 dB	-0.27 dB	72.98%	98.56 %
Mobile (CIF)	-0.04 dB	-0.13 dB	73.06%	99.03%
SignIrene (CIF)	-0.07 dB	-0.22 dB	75.92%	98.24%
Average of SDTV Seq.	-0.03 dB	-0.28 dB	70.61%	99.47%
Average of CIF Seq.	-0.06 dB	-0.22 dB	73.57%	98.66%

ACKNOWLEDGMENT

The authors would like to thank Dr. Hassan Mansour for the insightful advice on this work.

REFERENCES

[1] T. Wiegand, H. Schwarz, A. Joch, F. Kossentini, G. Sullivan, "Rate-Constrained Coder Control and Comparison of Video Coding Standards," *IEEE Trans. CSVT.,* vol. 13, no. 7, pp. 688-703, Jul. 2003.

[2] J. Xin, C. W. Lin, and M. T. Sun, "Digital Video Transcoding," *Proceedings of the IEEE*, vol. 93, no. 1, pp. 84-97, Jan. 2005.

[3] I. Ahmad, X. Wei, Y. Sun, Y-Q. Zhang, "Video Transcoding: An Overview of Various Techniques and Research Issues," *IEEE Trans. on Multimedia*, vol. 7, no.5, pp. 793-804, Oct. 2005.

[4] J. Xin, J. Li, A. Vetro, H. Sun, S. Sekiguchi, "Motion Mapping for MPEG-2 to H.264/AVC Transcoding, " in proc. *IEEE ISCAS 2007*, May 2007, pp. 1991-1994.

[5] G Fernández, H Kalva, P Cuenca, LO Barbosa, "Speeding-up the Macroblock Partition Mode Decision in MPEG-2/H.264 Transcoding," in proc. *IEEE ICIP 2006*, Oct. 2006, pp.869-872.

[6] H. Kato, A. Yoneyama, Y. Takishima, Y. Kaji, "Coding Mode Decision for High Quality MPEG-2 to H.264 Transcoding," in proc. *IEEE ICIP 2007*, Oct. 2007, pp.IV77-IV80.

[7] M. Wien, "Variable Block-Size Transforms for H.264/AVC," *IEEE Tran. CSVT*, vol. 13, no. 7, pp. 604-613, Jul. 2003.

[8] H. Mansour, P. Nasiopoulos, V. Krishnamurphy, "Real-Time Joint Rate and Protection Allocation for Multi-User Scalable Video Streaming," in proc. *IEEE PIMRC 2008*, Sep. 2008, pp.1-5.

[9] Joint Video Team, H.264/AVC reference software codec (JM), version 14.2, [online], Available: http://iphome.hhi.de/suehring/tml/download/.

[10] TK Tan, G. Sullivan, and T. Weidi, "Recommended Simulation Common Conditions for Coding Efficiency Experiments Revision 1," ITU-T SC16/Q6, Doc. VCEG-AE010, Jan. 2007.

[11] Gisle Bjontegaard, "Calculation of Average PSNR Differences between RD curves", ITU-T SC16/Q6, Doc. VCEG-M33, Apr. 2001.