COMPLEXITY-SCALABLE H.264/AVC IN AN IPP-BASED VIDEO ENCODER

Tiago A. da Fonseca, Ricardo L. de Queiroz

Universidade de Brasilia
Department of Electrical Engineering
Brasilia, DF, Brazil

Debargha Mukherjee

Hewlett Packard Laboratories
Palo Alto, CA, USA

ABSTRACT

Real-time high-definition video encoding is a computation-hungry task that challenges software-based solutions. For that, in this work we adopted an Intel software implementation of an H.264 video encoder and optimized its prediction stage in the complexity sense (C). Thus, besides looking for the coding options which lead to the best coded representation in terms of rate and distortion, we constrain the process to fit within a certain time budget. We present an RDC-optimized framework which allows for real-time HD video compression.

Index Terms— H.264, mode decision, complexity scalability, high-definition, real-time video coding.

1. INTRODUCTION

In H.264/AVC [1], the many improvements over previous encoding methods led to enhanced code efficiency for a wide range of applications including video telephony, video conferencing, digital TV, streaming video etc. The H.264/AVC coder has been well decribed in the literature [2, 3, 4, 5]. As a hybrid DPCM encoder [6], the H.264 searches for the best possible prediction of the encoding signal in order to reduce the prediction residue and to spend less bits in its compressed representation.

When compressing digital video, temporal (Inter) and spatial (Intra) redundancies are explored. "Inter" prediction generates prediction from one or more previously encoded video frames using block-based motion compensation. A prediction block can also be formed based on planar extrapolation of previously encoded and reconstructed neighbouring pixels ("Intra" prediction). The encoder may either select the prediction mode for each block that minimizes the difference between the predicted block and the block to be encoded, or to use rate-distortion (RD) optimization [7].

The search for the best possible prediction is one of the most time consuming stages of a software-based video encoder [8]. When encoding video sequences in a low latency communications scenario, e.g. video conferencing, the time spent compressing the signal is an issue and real-time coding of high-definition (HD) content is challenging. As encoders take most of time doing predictions, reducing the complexity of this stage seems to be the natural way to reduce the overall complexity.

There are many works aimed at reducing AVC complexity. Some explore prediction techniques for reducing computations with small RD penalties [9, 10, 11, 12]. Other common approach is to implement the AVC recommended framework in an complexity optimized way by using faster algorithms, high-performance libraries and platform dependent resources [13, 14].

In order to perform real-time software HD video encoding, we present some modifications to non-normative aspects of the H.264/AVC prediction stage and we evaluate them using high performance libraries provided by Intel [14].

2. PREDICTIONS IN H.264/AVC

2.1. Prediction stage in AVC

The AVC prediction stage can be divided into temporal ("Inter") and spatial ("Intra") models. In Inter prediction, we observe important advances from earlier video standards, which include the support for a wide range of block sizes (16×16-pixels and down, as in Fig. 1, according to a quadtree-like rule), multiple reference frames and refined motion vectors (quarter-sample resolution for the luminance component). In Intra prediction, the predicted block can have different sizes (besides 16×16-pixel size macroblock, blocks of 8×8 and 4×4-pixel size are also allowed) and is formed based on planar extrapolation of previously encoded blocks which belong to the same frame. The prediction residue is transformed and quantized through the use of integer transforms [7].

The data set composed by block size and Intra (extrapolation choice) or Inter (motion vectors and reference frames) parameters forms the "prediction **mode**" of a block. The encoder tipically selects the prediction mode for each block that minimizes the difference between the predicted block and the block to be encoded, constrained to a given bitrate.

2.2. Aspects of Intel's® AVC implementation

The Intel® H.264/AVC compliant implementation, delivered through IPP Libraries [14], is a computational efficient soft-

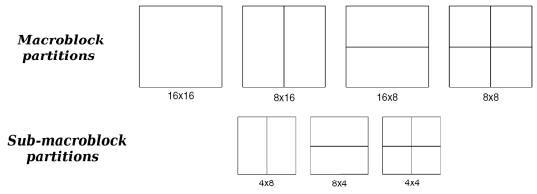


Fig. 1. Macroblock and submacroblock partitions for motion compensation in Inter Prediction.

ware implementation. Significant reduction in the time spent encoding digital video is reached by using good programming techniques, as well as optimized compiler and libraries built to explore processor resources. Using processors shipped in regular personal computers, the IPP AVC implementation allows for real-time AVC coding of standard resolution sequences. Besides the regular parameters used to control the compression process, the IPP AVC implementation provides an extra parameter set to scale the encoding computational complexity and, thus, the distortion of the encoded video sequence.

The first extra parameter, **Sub-block Split**, is responsible to control how do the encoder follow the quadtree-like macroblock partition scheme for Inter prediction. It has three options: (i) to allow only 16×16-pixel block motion compensation; (ii) to allow for motion compensation up to 8×8-pixel blocks and, (iii) an unconstrained block-size motion compensation. Another parameter is **Speed/Quality Grade**, which is responsible to enable/disable optimization tools in such a way that one exchanges between performance and coding speed. It basically controls the RD-optimization method, how precise should be the motion estimation (sub-pel motion estimation) and the reference buffer management. The last extra parameter, **Optimal Quantization**, is almost self explanatory: to turn on or off an optimal quantization process when dealing with residues.

Those parameters focus on the prediction stage performance in order to achieve some complexity scalability. The present work explores adapting these and other complexity related encoder parameters to more finely constrain the encoder complexity.

3. THE APPROACH

Our framework for complexity-constrained real-time HD video coding consists in controlling the amount of complexity spent while encoding a video sequence by adjusting encoding parameters in such a way that the RD penalties, compared to a full-featured case, remain as low as possible. Here, the computational complexity is measured as the time spent to encode a particular video sequence on a given system setup.

Initially, we modified the **Sub-block Split** parameter, which is originally responsible to control how does the IPP AVC encoder follow the quadtree-like macroblock partition scheme for Inter prediction. Such a modification is intended to account for intermediate values allowing for a finer granularity in complexity scaling.

The approach is to extend an RD-optimization [15] strategy, adding another dimension (C, which stands for complexity) to the regular 2-D problem of optimizing a particular codec to spend the smallest bitrate (R) necessary to represent a encoded video signal at a particular distortion (D). So, for each particular parameters selection (see 2.2), we compute the total bitrate (R), the average PSNR (D) and the ratio between the time spent to encode a training sequence and the time spent by the full-featured encoder to do the same job (C). The RDC points are used to populate an initial search space from which we look for the points that lie on its lower convex hull, as illustrated in Fig. 2 for constant rates. After finding the setups that belong to the Pareto front, i.e, the ones which, besides imposing the lesser RD penalties, allows for the fastest encoding, we build a lookup table from the performance numbers in order to provide optimal starting RDC points. Any intermediate complexity point not found in the table can be easily achieved by interpolation, using linear combinations of neighbouring setups to encode different parts of the sequence, in such a way that the global complexity is very close to the required complexity "budget".

As a result of optimization process, was inserted an extra parameter to the IPP AVC encoder, **QPC**, which stands for Quantization Parameter for Complexity and tunes the encoder complexity according to the user demand by adjusting jointly the extra parameters described in Section 2.2. The values **QPC** can assume stand between $(0, 100)$ and each value represents the amount of complexity (in percents) that will be used to encode the video sequence.

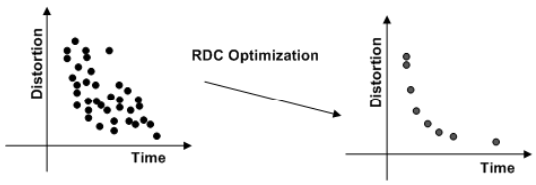


Fig. 2. Finding the set of points that compose the Pareto Front. For the illustration, rate was kept constant.

4. RESULTS

The proposed method was implemented in the H.264 IPP encoder, using *UMHexS* motion estimation and one reference frame. Complexity was taken as the ratio of the net encoding time for the reduced complexity setup to the full-featured coder net encoding time. Initially, we evaluated the performance of the modified AVC IPP implementation by coding a 720p (progressive, 1280×720-pixel frame size) digital video sequence composed by the concatenation of standard sequences ("Shields", "Parkrun" and "Mobile and Calendar"

sequences), forming a training sequence. We tested different encoding parameters like: quantization parameter (QP), motion estimation range, rate-distortion optimization analysis (different efficiency techniques), optimal quantization of residues (turned on and off). We also tested the full set of macroblock partitions.

To each parameter set choice, we compared its performance to the full-featured case (100%-complexity, which corresponds to 2.7fps in our system for the training sequence) by evaluating average bitrate and PSNR differences between two RD curves, as described in [16]. Then, we found the optimal RDC points which belong to the Pareto front, as in Fig. 3. The general behaviour suggests that, as we reduce the complexity used to encode a video sequence, the penalties in performance increase. There is a region where the modified encoder can even outperform the original one, basically for complexity reductions from 0% up to 40%. This is mainly due the fact that constraining the encoding to certain syntax elements can better better explore RD costs through more efficient entropy coding. It also provides significative reductions in the time spent to encode the sequence.

Fig. 4 presents a perfomance drop curve for the 720p test sequences for the range of complexity with starts with 40% of complexity reduction. This is because the RDC-optimization process delivered a new best operating point, which is around 40% of complexity savings, since the encoder may outperform the 100% in terms of RD and also complexity (C). In general, there are small performance losses (average rate increase below 6%, for fixed quality) when encoding the test video sequence in a reduced complexity scenario. The curve's shape is similar to the ones obtained for the training set. An exception is the performance penalty curve for "Rushhour" sequence, whose values are a bit higher that the expected from the optimization process. The full-featured encoder (100%-complexity reference) compressed the test sequences in 2.3fps ("Stockholm"), 4.0fps ("Sunflower") and 3.0fps ("Rushhour") in our platform, a notebook with a Centrino® 2 processor (2.4 GHz of clock speed) and 4GB of RAM.

The main result is a real-time software-based AVC encoder. Using our optimization framework, we found a parameter set which allows for the fastest encoding by spending the lesser bitrate for each tested QP (D). Therefore, the results may be slightly distinct from those shown in Fig. 3, when we were concerned on the average behaviour within the QP range [16]. The real-time encoder RD-curves (H.264-IPP RT) are shown and compared to the best performance achieved (H.264-IPP Best) in Fig. 5, which is not necessarily the 100%-complexity setup as discussed above. Average rate increasings are below 5% for "Stockholm" and "Sunflower" sequences, and around 11% for Rushhour. A mean encoding frame rate around 15fps, with the encoder reaching 20fps for lower rates, was the performance provided by our test platform. Faster encoding for lower rates is somewhat expected as the best macroblock coding modes become biased around

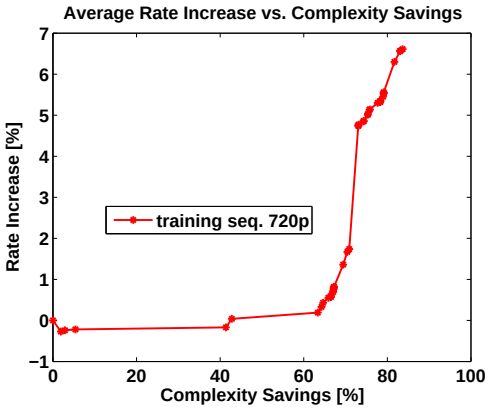


Fig. 3. Average performance penalties *vs.* complexity reduction for the training video sequences. Here, 100% of complexity corresponds to 2.7fps.

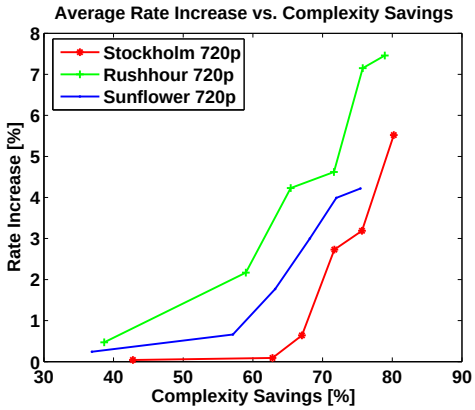
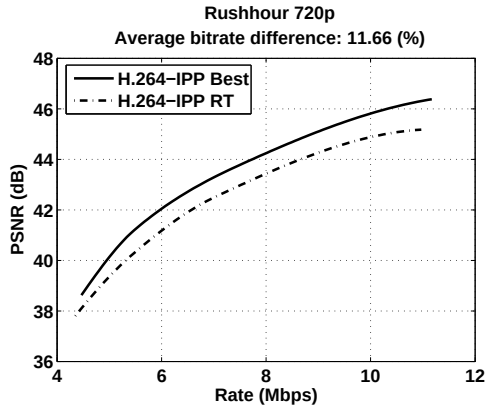


Fig. 4. Average rate increase vs. complexity savings for 720p video sequence.

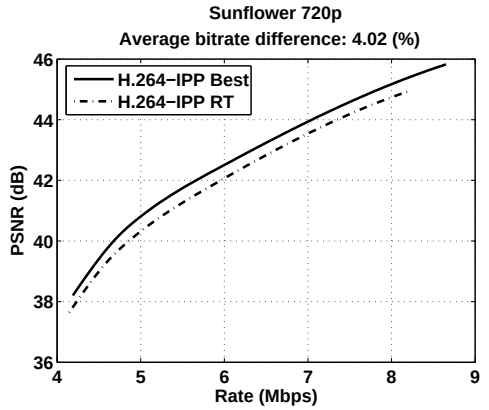
bigger partitions for HD sequences at higher QPs [17]. Nevertheless, these frame rates, besides low for applications like digital video broadcast, are suitable for video conferencing using a notebooks with embedded digital cameras where we can tradeoff frame rate and resolution for performance.

5. CONCLUSIONS

We proposed a fully compliant complexity optimized framework for a H.264/AVC software implementation that allows real-time software coding of HD video. Rather than testing all prediction modes available, we search for a subset of prediction modes whose size and composition are constrained by the amount of available complexity. Our main contribution is the insertion of an encoding parameter capable of controlling the encoding complexity in an IPP H.264/AVC implementation: **QPC** - Quantization Parameter for Complexity. Our tests have shown that the RD performance is only moderately affected by skipping prediction modes, which means the skipped modes are not always very relevant in terms of RD, even though there are many mismatches. Nevertheless, we



(a)



(b)

Fig. 5. Performance comparison between a full-featured codec and its real-time capable version for the test video sequences: (a) "Rushhour" and (b) "Sunflower".

achieved significant complexity reduction. Our framework can benefit from the availability of more powerful computational platforms and also be used in PC-based video encoder appliances. We plan to further work on making an encoder aware of environmental and commutications conditions, capable of adjusting itself to meet channel, quality and energy constraints.

6. REFERENCES

[1] JVT of ISO/IEC MPEG and ITU-T VCEG, "Advanced video coding for generic audiovisual services," Tech. Rep. 14496-10:2005, March 2005.

[2] I. E. G. Richardson, *H.264 and MPEG-4 Video Compression*, John Wiley & Sons Ltd, 2003.

[3] T. Wiegand, G. J. Sullivan, G. Bjontergaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 560–576, July 2003.

[4] G. J. Sullivan, P. Topiwala, and A. Luthra, "The H.264/AVC Advanced Video Coding Standard: Overview and Introduction to the Fidelity Range Extensions," *Proc. SPIE Conference on Applications of Digital Image Processing XXVII*, Aug. 2004.

[5] R. L. de Queiroz, R. S. Ortis, A. Zaghetto, and T. A. Fonseca, "Fringe benefits of the H.264/AVC," in *Proc. International Telecommunication Symposium*, 2006, pp. 208–212.

[6] K. Sayood, *Introduction to Data Compression*, Morgan Kuffmann Publishers, 2000.

[7] J. Ostermann, J. Bormans, P. List, D. Marpe, M. Narroschke, F. Pereira, T. Stockhammer, and T. Wedi, "Video coding with H.264/AVC: tools, performance, and complexity," *IEEE Circuits and Systems Magazine*, pp. 7–28, Jan-Mar 2004.

[8] Y.-Y. Huang, B.-Y. Hsieh, S.-Y. Chien, S.-Y. Ma, and L.-G. Chen, "Analysis and complexity reduction of multiple reference frames motion estimation in H.264/AVC," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 4, pp. 507–522, Apr. 2006.

[9] Z. Chen, P. Zhou, and Y. He, "Hybrid unsymmetrical-cross multi-hexagon-grid search strategy for integer pel motion estimation in H.264," *Proc. Picture Coding Symposium*, Apr. 2003.

[10] B.G. Kim, S.-K. Song, and C.-S. Cho, "Efficient inter-mode decision based on contextual prediction for the P-slice in H.264/AVC video coding," *Proc. IEEE International Conference on Image Processing*, pp. 1333–1336, September 2006.

[11] C. S. Kannangara, Y. Zhao, I. E. Richardson, and M. Bystrom, "Complexity control of H.264 based on a bayesian framework," *Proc. Picture Coding Symposium*, November 2007.

[12] Z. He, Y. Liang, L. Chen, I. Ahmad, and D. Wu, "Power-rate-distortion analysis for wireless video communication under energy constraints," *IEEE Transactions on Circuits and Systems for Video Technology*, pp. 645–658, May 2005.

[13] Kyu Park, Nitin Singhal, Man Hee Lee, , and Sungdae Cho, "Efficient Design and Implementation of Visual Computing Algorithms on the GPU," *Proc. IEEE Intl. Conf. on Image Processing*, November 2009.

[14] Intel, "Intel Integrated Performance Primitives," *http://software.intel.com/en-us/intel-ipp/*.

[15] Gary J. Sullivan and Thomas Wiegand, "Rate-Distortion Optimization for Video Compression," *IEEE Signal Processing Magazine*, vol. 15, no. 6, pp. 74–90, November 1998.

[16] G. Bjontegaard, "Calculation of average PSNR differences between RD-curves," *VCEG-M33*, April 2001.

[17] T. A. da Fonseca and R. L. de Queiroz, "Complexity reduction techniques applied to the compression of high definition sequences in digital TV," *Proc. XXVI Simposio Brasileiro de Telecomunicacoes*, pp. 43–47, September 2008.