Low Complexity H.264 Video Encoding

Paula Carrillo[†], Hari Kalva[†], and Tao Pin[‡].
[‡]Dept. of Computer Science and Technology,Tsinghua University, Beijing, China
[†]Dept. of Computer Science and Engineering, Florida Atlantic University, Boca Raton, FL, USA

ABSTRACT

H.264/AVC encoder complexity is mainly due to variable block size in Intra and Inter frames. This makes H.264/AVC very difficult to implement, especially for real time applications and mobile devices. The current technological challenge is to conserve the compression capacity and quality that H.264 offers but reduce the encoding time and, therefore, the processing complexity. This paper applies machine learning technique for video encoding mode decisions and investigates ways to improve the process of generating more general low complexity H.264/AVC video encoders. The proposed H.264 encoding method decreases the complexity in the mode decision inside the Inter frames. Results show, on average, a 67.36% reduction in encoding time, a 0.2 dB decrease in PSNR, and an average bit rate increase of 0.05% for different kinds of videos and formats.

KEYWORD: Low complexity H.264, Machine learning, Data mining, Inter prediction.

1 INTRODUCTION

H.264 video coding standard is the latest block-oriented motion-compensation-based codec standard developed by the ITU-T Video Coding Experts Group (VCEG) together with the ISO/IEC Moving Picture Experts Group (MPEG). H.264 can achieve considerably higher coding efficiency than previous standards. Unfortunately, this comes at a cost in considerably increased complexity at the encoder mainly due to motion estimation and mode decision. The high-computational complexity of H.264 and requirements of real-time video systems represent the main challenge to overcome in the development of efficient encoder solutions.

Different techniques to reduce complexity in H.264 have been proposed in the literature and most of them are focused on limiting the subset of modes evaluated. This paper is focused on reducing the complexity of the H.264 encoder using machine learning techniques. The ideas developed are implemented and evaluated in a highly optimized encoder implementation available in the Intel IPP SDK. The idea behind using machine learning is to exploit structural similarities in video in order to make optimal prediction modes through the use of fast if-else statements instead of the usual cumbersome Sum of Absolute Differences (SAD) and cost evaluations. In this paper, we also use commpnly applied concepts for speeding up the encoding process in H.264 such as fast searching algorithms, window refinement and search window reduction. Also, residual information, Macro Block (MB) AC energy (MB mean, MB variance) and edge detection were implemented as part of the attribute set used for training and classification of homogenous/stationary regions with a corresponding MB mode.

The rest of the paper is organized as follows. Section 2 reviews the principles of operation of the prediction of inter-coded MB in P-slices in the H.264 encoding standard. Section 3 gives background of the proposed approach. Section 4 introduces the implemented decision tree structure. Results are presented in Section 5 and conlcusions drawn in section 6.

2 PREDICTION OF INTER-CODED MACROBLOCKS IN P-SLICES IN AN H.264 CODEC

In the H.264 standard, the MB mode decision in Inter frames is the most computationally expensive process due to the use of the features such as variable block-size, motion estimation, and quarter-pixel motion compensation. Inter prediction creates a prediction model from one or more previously encoded video frames or fields using block based motion compensation. H.264 uses block-based motion compensation, the same principle adopted by every major coding standard since H.261. Important differences from earlier standards include the support for a range of block sizes (down to 4x4) and fine sub-pixel motion vectors (1/4 pixel in the luminance component). H.264 supports motion compensation block sizes ranging from 16x16 to 4x4 luminance samples with many options between the two. The luminance component of each MB (16x16 samples) may be split up in four ways: one 16x16 MB partition, two 16x8 partitions, two 8x16 partitions or four 8x8 partitions (see Figure 1). Each of the sub-divided regions is a MB partition. If the 8x8 mode is chosen, each of the four 8x8 MB partitions within the MB may be further split into four ways: one 8x8 partition, two 8x4 partitions, two 4x8 partitions or four 4x4 partitions. These new shapes are

known as sub-MB partitions, (see Figure 2). These partitions and sub-partitions give rise to a large number of possible combinations within each MB. This method of partitioning MBs into motion compensated sub-blocks of varying size is known as Tree structured motion compensation. The resolution of each chrominance component in a MB (Cr and Cb) is half that of the luminance component. Each chrominance block is partitioned in the same way as the luminance component, except that the partition sizes have exactly half the horizontal and vertical resolution.

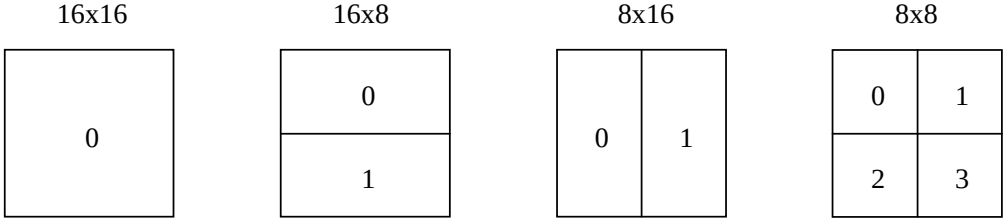

Figure 1 MB partitions: 16x16, 16x8, 8x16, 8x8

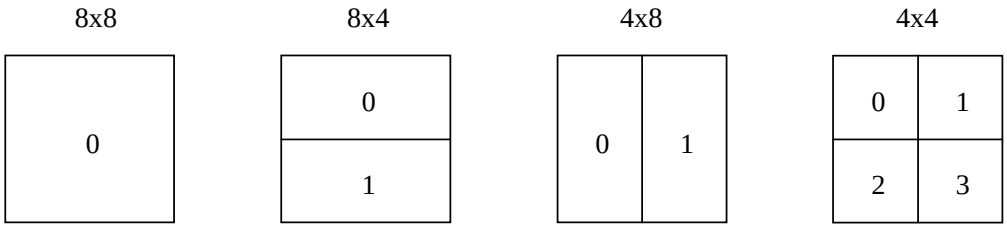

Figure 2 Sub-MB partitions: 8x8, 8x4, 4x8 and 4x4.

For evaluating the motion vectors, each partition in an inter-coded MB is predicted from an area of the same size in a reference picture. The offset between the two areas (the motion vector) has ¼-pixel resolution (for the luminance component). The luminance and chrominance samples at sub-pixel positions do not exist in the reference picture and so it is necessary to create them using interpolation from nearby image samples. If components of the motion vectors are integers, the relevant samples in the reference block actually exist. If one or both vectors components are fractional values, the prediction samples are generated by interpolation between adjacent samples in the reference frame. Sub-pixel motion compensation can provide significantly better compression performance than integer-pixel compensation, at the expense of increased complexity. To summarize, new features in H.264 not only increase compression rate performance, but also increase the complexity. However, among H.264 compression modules, it is important to emphasize that the most computational expensive process is Motion Estimation (ME). For example, assuming Full Search (FS) and Z block types, N reference frames and a search range for each reference frame and block type equal to +/- W, we need to examine N x Z x $(2W + 1)^2$ positions for a single reference/block type. After the first FS, a sup-pixel search could also be performed. Moreover, H.264 supports skip mode and intra mode evaluation for Inter slices. Intra mode has two block types: 4x4 and 16x16. Intra 4x4 supports 9 modes and intra 16xl6 supports 4 modes. At the end, the union of all mode evaluations, cost comparisons and exhaustive search inside ME cause a great amount of time spent by the encoder. In other words, complex and exhaustive ME evaluation is the key to good performance achieved by H.264, but the cost is in the encoding time. For this reason, this paper will focus on reducing the processor operations inside the ME module, more specifically inside of the MB mode decision, therefore reduce the H.264 complexity and encoding time.

3 RELATED WORK

In the H.264 standard, the MB mode decision in Inter frames is the most computationally expensive process due to the use of the variable block-size (16x16, 16x8, 8x16, 8x8, 8x4, 4x8, 4x4) motion estimation. Different techniques on fast inter mode decisions in H.264 encoding [1 to 6] have been proposed. All these approaches try to reduce the computational cost of making MB mode predictions by skipping all/some of the mode calculations and comparisons that are necessary for optimal coding. Previous works made fast MB mode type classification by experimentally setting some thresholds of a metric. The metric used is based on the observation of the structural similarities present in videos. Some of the metrics used are: frame residual information [1], amplitude of edge vectors [2], correlation of collocated MB mode type [3], MB index complexity classification through energy variance [4] among others. Our approach is summarized in Figure 1. The key idea behind this approach is to determine encoder decisions such as MB coding mode decisions that are computationally expensive using easily

computable features derived from uncompressed video. A machine learning algorithm is used to deduce the classifier /decision tree based on such features. Once a tree is trained, the encoder coding mode decisions that are normally done using cost-based models that evaluate all possible coding options are replaced with a decision tree. Decision trees are in effect if-else statements in software and require negligible computing resources. As in the case of previous works our proposed approach also uses MB metrics in order to exploit the video structure, but with the difference that we train classifiers with a set of metrics instead of trying to manually find the correct threshold for each one at the time. This difference permits us to generate more general and accurate MB mode type classifications as data mining algorithms can detect correlations more effectively.

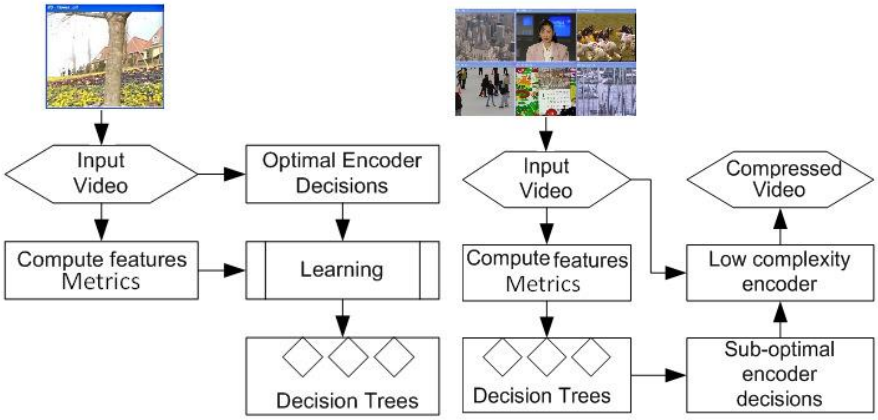

Figure 3 Applying machine learning to video coding

The process of obtaining data for training is done offline. In this supervised learning approach, we used the data of the first four frames of the flower.yuv video in CIF format for training and with a fixed QP of 28. An offline process is used to extract attributes for the current MB and the residual MB; a detailed description of attributes and attribute selection can be found in [9]. The attributes extracted from the MC residual and the current MB and MB mode chosen by the full complexity Intel® IPP H.264 are saved in an ARRF file. Next is the training stage where using WEKA tool, mode decision Trees are discovered through C4.5 (J48) classifier algorithm. Then, these Trees are implemented as if-else statements in the Intel® H.264 encoder. The purpose of these Trees is to replace the original complex Inter mode decision. During the encoder operation, for each MB, MB type is determined using the decision trees and a reduced motion search is computed only for a single MB size. Reducing the motion estimation to motion estimation of a single block reduces the encoder complexity significantly.

4. MB DECISION TREE STRUCTURE

In order to improve the accuracy and to reduce a bit-rate penalty in the case of an erroneous selection of the MB mode, the proposed evaluated topology Tree was selected as shown in Figure 4.

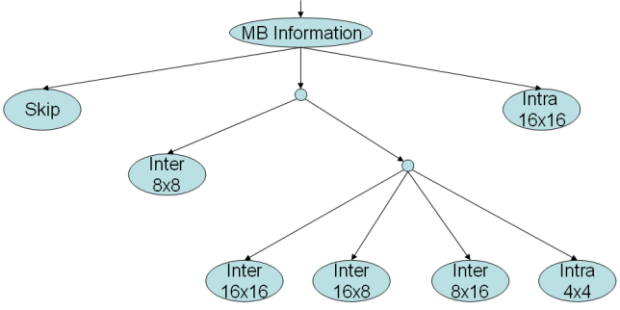

Figure 4 **Implemented Inter Tree mode decision**

This paper proposes a three level topology Tree for Inter mode decision. In the first level, an improvement in speed up is achieved with an early Skip decision. Also, after observation of the residual and metrics information, a temporary mode sub-classification for the first level was chosen as {Skip, Intra 16x16 and the rest of the modes}. Skip shows reduced residual information, while Intra 16x16 shows uniform residual information and the rest of the modes show a more diverse residual and metrics values. In the second level, there is a consequent division between

Inter 8x8 and sub modes against Inter 16x16 and sub modes. The implementation is not going inside the 8x8 modes so everything that is an element of the possible outcomes {Inter 8x8, 8x4, 4x8, 4x4} is marketed and treated as Inter 8x8. On the other hand, if the other leaf is selected, a third classification between Inter 16x16 sub modes {16x16, 16x8, 8x16} and the Intra 4x4 is evaluated. Observations of the metrics detected a high co-occurrence between 16x8, 8x16 and Intra 4x4, and for that reason, these modes were at the same Tree level, in an attempt to reduce a negative impact of a bad mode selection.

A statistical MB mode decision observation shows a leak of Intra 16x16 and Intra 4x4. Therefore, in order to correct possible misclassifications, a double check in the program was implemented. Another important observation was that, due to the heuristic criteria to rank possible tests of C4.5, a minimization of the total entropy of the classification subsets is sought, but results are heavily biased towards tests with numerous outcomes. For different QPs or/and video content, MB mode distribution changes, and for that reason, the classification Trees could start to be inaccurate. In order to have a more general mode decision Tree, it would be necessary to train Trees for different MB type distributions, and then depending on the video content chose the most suitable Tree. In order to gain generality, for this paper, an implementation using two trained Trees was implemented. One with QP =28 and the other with a QP = 40. The implementation selects a decision Tree according to a closer MB QP. Experiments show that Rate Distortion (RD) performance with two Trees has a better performance, especially in low bitrates, comparing with experiments using a single QP trained Tree.

5 RESULTS

Two sets of experiments were conducted to evaluate the performance: 1) constant QP where encoder varies the bitrate and 2) constant BR where QP varies by content.

5.1 Performance Evaluation Metrics

Many low complexity H.264 approaches have been proposed in the literature; most of them have been implemented in JM H.264 reference software. Our results cannot be compared in a direct way with previous studies due to the use of different codecs. The main difference of JM with respect to Intel® IPP H.264 is that JM H.264 reference software is not an optimized code in a computationally sense. JM H.264 reference code is primarily used for evaluation purposes; during many steps of the encoding process a lot of statistical data information is obtained and processed, increasing significantly the encoding time. The results presented compare the Intel IPP H.264 with and without our complexity reduction tools. The following metrics are used for performance evaluation:

- % Increase in FPS: $\frac{FPS_{Tree} - FPS_{Original}}{FPS_{Original}} x100$
- % Increase in Bit rate: $\frac{BR_{Tree} - BR_{Original}}{BR_{Original}} x100$
- Decrease in PSNR: $PSNR_{Original} - PSNR_{Tree}$
- % Reduction in MB mode decisions: $\frac{Time_{MB\_Original} - Time_{MB\_Tree}}{Time_{MB\_Original}} x100$
- %Reduction in encoder time: $\frac{Time_{Enc\_Original} - Time_{enc\_Tree}}{Time_{enc\_Original}} x100$

5.2 Experimental Results

For HD format sequences tested, the first set of experimental results show (constant QP), on average, a time reduction of 67% with a bit rate increase of 0.6%. On the other hand, for the second set of experiments (constant BR), the bit rate range used were between 1.5Gbps and 12Gbps, and the outcome presents an average time reduction of 68% and a PSNR decrease of 0.19dB.

	% Average Increase BitRate	Average decrease PSNR	% Average speed up Encoder	% Average speed up MB

			time	Decision
QPs Fixed Experiments HD				
ParkrunHD.yuv	0,60%	0,04	60,82%	67,30%
ShieldsHD.yuv	1,75%	0,04	66,88%	72,35%
StockholmHD.yuv	-0,01%	0,02	68,35%	73,87%
SuperBowlHD.yuv	0,11%	0,03	72,33%	77,91%
Average	**0,61%**	**0,03**	**67,10%**	**72,86%**
BRs Fixed Experiments HD				
ShieldsHD.yuv	0,15%	0,52	65,40%	70,84%
ParkrunHD.yuv	-0,06%	0,20	67,38%	72,75%
StockholmHD.yuv	-0,01%	0,02	68,35%	73,87%
SuperBowlHD.yuv	0,11%	0,03	72,33%	77,91%
Average	**0,05%**	**0,19**	**68,37%**	**73,84%**

Table 2 Average results HD format

For SD format sequences tested, in the first set of experiments, results show, on average, a time reduction of 65% with a bit rate increase of 4.5%. For the second set of experiments, the bit rate range used were between 350Kbps and 4Gbps, and the outcome presents an average time reduction of 67% and a PSNR decrease of 0.21dB.

	% Average Increase BitRate	**Average decrease PSNR**	**% Average speed up Encoder time**	**% Average speed up MB Decision**
QPs Fixed Experiments TV				
CrewTV.yuv	3,93%	0,02	64,24%	70,15%
HarbourTV.yuv	0,99%	0,07	62,84%	68,57%
IceTV.yuv	8,14%	0,09	71,57%	77,62%
SoccerTV.yuv	5,02%	0,07	63,25%	69,02%
Average	**4,52%**	**0,06**	**65,48%**	**71,34%**
BRs Fixed Experiments TV				
CrewTV.yuv	0,13%	0,07	66,76%	72,73%
HarbourTV.yuv	0,13%	-0,04	66,22%	71,51%
IceTV.yuv	0,01%	0,39	71,84%	77,75%
SoccerTV.yuv	-0,16%	0,43	64,19%	70,11%
Average	**0,03%**	**0,21**	**67,25%**	**73,02%**

Table 3 Average results TV format

Some experiments result in better RD performance compared with the standard H.264Intel® IPP FS encoder algorithm. These results are primarily due to two factors. The first one is because the encoder version used does not support RDO option, then is not possible to guarantee that RD performance is always better with the reference encoder. The second reason is that because of a better MV prediction, achieved through the calculation of the PMV followed by a fast EPZS, makes possible to extend the search range and it is possible to capture a better MB match; hence, sometimes resulting in better RD averages.

5.3 Comparitive Evaluation

Previous works are based on JM reference software. However, this work was implemented over Intel® IPP H.264 encoder. For that reason it is not possible to make a total comparison of the results. However, in order to present an idea of absolute results below we show a resume table of average time saving, PSNR increase of our work comparing with some of the previous presented works in this field. The comparison used the results from the fixed QPs experiments. It also important to point out that since we are using an optimized encoder implementation, a 65% reduction is far more significant than reducing the encoding time in the reference unoptimized code by 80%.

FIXED QPS	**Encoding Time Saving (%)**	**PSNR decrease (dB)**	**JM/Intel®**	**RDO (on/off)**
Yu's method	17% - 31%	0.2	JM6.1	ON
Cheng-Chang and Chung-	60%	0.04	JM9.3	ON

Ping method				
Xuan and Lap-Pui method	36%	-0.04	JM6.0	ON
Wu's method	29.35%	0.09	JM5.0	ON
Proposed	**65,71%**	**0.05**	**Intel®**	**OFF**

Table 5 Result Comparisons previous works

RDO "ON" parameter increases the compression and the precision of the MB mode selection, but at the same time the complexity and hence the encoder time. For that reason, the works which compares themselves with JM RDO "ON", they are comparing time results against a slower version of the encoder. But of course at the same time they are comparing with a most precise encoder, in other words with the optimal encoder outcome.

6 CONCLUSIONS

The following are the main conclusions drawn from this work:

1) Machine learning has a very good potential of reducing the complexity of encoding and make realtime encoding possible on resource constrained decvices such as mobiles.
2) The complexity of a highly optimized encoder is reduced by about 65%
3) The J48 learning algorithm produces a reasonably good classifier. However, there is room for improvement.
4) Training set doesn't have to include a large number of videos but a single sequence that has the right type of block types. This is mainly because attributes used are based on motion compensated residual.
5) Attribute design and evaluation is the most critical and time consuming stage of such encoder design.

REFERENCES

[1] Xuan Jing and Lap-Pui Chua, "An Efficient Inter Mode Decision Approach for H.264 Video Coding" International Conference on Multimedia and Expo (ICME) 2004.

[2] D.Wu, S.Wu, K.P Lim, F. Pan, Z.G. Li, C. C. Ko "Fast Inter Mode Decision for H.264 Encoding". Consumer Electronics, 2004 IEEE International Symposium on Volume , Issue , Sept. 1-3, 2004 Page(s): 406 - 409

[3] Cheng-Chang Lien, Chung-Ping Yu, "A Fast Mode Decision Method for H.264/AVC Using the Spatial-Temporal Prediction Scheme", ICPR 2006

[4] Andy C. Yu, "Efficient block-size selection algorithm for inter-frame coding in H.264/Mpeg-4 AVC", IEEE International Conference on Acoustics, Speech and Signal Processing. 2004.

[5] S. Saponara, M. Casula, F. Rovati, D. Alfonso, L. Fanucci, "Dynamic Control of Motion Estimation Search Parameters for Low Complex H.264 Video Coding", IEEE Transactions on Consumer Electronics, Vol. 52, No. 1, FEBRUARY 2006.

[6] S. Saponara, M. Melani, L. Fanucci, P. Terreni, "Adaptive algorithm for fast motion estimation in H.264/MPEG-4 AVC", Proc. Eusipco2004, pp. 569 – 572, Wien, Sept. 2004

[7] Fernández, Kalva, Cuenca, Orozco, "A first approach to speeding-up the inter mode selection in MPEG-2/H.264 transcoders using machine learning", Multimed Tools Appl (2007) 35:225–240

[8] Intel Integrated Performance Primitives Reference Manual: Volume 2.

[9] Paula Carrillo, " Low complexity H.264 Video Encoder Design Using Machine Learning techniques," MS Thesis, Department of Electrical Engineering, Florida Atlantic University, Boca Raton, FL, USA, November 2008.