ABSTRACT

Though H.264 is highly efficient compared to MPEG-2, the wide and deep penetration of MPEG-2 creates a need for co-existence of these technologies and hence creates a need for transcoding. In this paper, we introduce and evaluate a novel macroblock partition mode decision algorithm for inter-frame prediction to be used as part of a high-efficient MPEG-2 to H.264 transcoder. The proposed tools are used to compute an optimal MB coding mode decision with significantly reduced computational complexity. Specifically, we achieve the computational savings by using the following MB information coming from MPEG-2: the MB coding modes in MPEG-2 and the mean and variance of the 16 4x4 sub blocks of the MPEG-2 residual MBs. We use data mining algorithms to develop a decision tree for obtaining the H.264 coding decision modes. Our results show that the proposed algorithm is able to maintain a good picture quality while considerably reducing the number of operations to be performed.

Index Terms— MPEG-2, H.264, Transcoding

1. INTRODUCTION

The H.264 video coding standard, also known as MPEG-4 AVC, is expected to take a dominant position in the digital video market over the next several years [1]. Though H.264 is highly efficient compared to MPEG-2, the wide and deep penetration of MPEG-2 creates a need for co-existence of these technologies and hence creates a need for transcoding. However, given the significant differences between the MPEG-2 and the H.264 coding algorithms, transcoding is a much more complex task compared to the task involved in other heterogeneous video transcoding architectures.

The main elements that require to be addressed in the design of an efficient heterogeneous MPEG-2/H.264 transcoder are: the inter-frame prediction, the transform coding and the intra-frame prediction. Each one of these elements requires to be examined and various research efforts are underway [2-7]. In this paper, we focus our attention on a part of the inter-frame prediction: the macroblock partition mode decision, one of the most computationally intensive tasks involved in the encoding process.

The H.264 video coding standard uses the notion of macroblock partition to refer to the group of pixels in a macroblock that share a common prediction. The encoder selects the coding-modes for the macroblock, including the best macroblock partition and mode of prediction for each macroblock partition, such that the video coding performance is optimized. In this paper, we introduce and evaluate a novel macroblock partition mode decision algorithm for speeding-up the inter-frame prediction in a MPEG-2 to H.264 transcoder. We achieve the computational savings by using the following MB information coming from MPEG-2: the MB coding modes in MPEG-2 and the mean and variance of the 16 4x4 sub blocks of the MPEG-2 residual MBs. From an exhaustive analysis of this information, we derive our tree decision suitable for their integration into our algorithm. Our results show that the proposed algorithm is able to maintain a good picture quality while considerably reducing the computational complexity.

The rest of the paper is organized as follows. Section 2 reviews the principles of operation of the prediction of inter-coded macroblocks in P-slices used by the H.264 encoding standard. Section 3 introduces our macroblock partition mode decision algorithm specifically designed for MPEG-2 to H.264 transcoders. In Section 4, we carry out a performance evaluation of the proposed algorithm in terms of its computational complexity and rate-distortion results. We compare the performance of our proposal to the SAE-cost method proposed by the H.264 standard. Finally, Section 5 draws our conclusions and outlines our future research plans.

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

The H.264/AVC video coding standard 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 luma component). H.264/AVC supports motion compensation block sizes ranging from 16x16 to 4x4 luminance samples with many options between the two. The luminance component of each macroblock (16x16 samples) may be split up in 4 ways: 16x16, 16x8, 8x16 or 8x8. Each of the sub-divided regions is a macroblock partition. If the 8x8 mode is chosen, each of the four 8x8 macroblock partitions within the macroblock may be further split in 4 ways: 8x8, 8x4, 4x8 or 4x4 (known as macroblock sub-partitions). These partitions and sub-partitions give rise to a large number of possible combinations within each macroblock. This method of partitioning macroblocks into motion compensated sub-blocks of varying size is known as tree structured motion compensation.

A separate motion vector is required for each partition or sub-partition. Each motion vector must be coded and transmitted; in addition, the size of the partition(s) must be encoded in the compressed bitstream. Choosing a large partition size (e.g. 16x16, 16x8, 8x16) means that a small number of bits are required to signal the choice of motion vector(s) and the type of partition; however, the motion compensated residual may contain a significant amount of energy in areas with high detail. Choosing a small partition size (e.g. 8x4, 4x4, etc.) may give a lower-energy residual after motion compensation but requires a larger number of bits to signal the motion vectors and choice of partition(s). The partition size therefore has a significant impact on compression performance. In general, a large partition size is appropriate for homogeneous areas of the frame and a small partition size may be beneficial for areas with high detail.

The resolution of each chroma component in a macroblock (Cr and Cb) is half that of the luminance (luma) component. Each chroma block is partitioned in the same way as the luma component, except that the partition sizes have exactly half the horizontal and vertical resolution (an 8x16 partition in luma corresponds to a 4x8 partition in chroma; an 8x4 partition in luma corresponds to 4x2 in chroma; and so on). The horizontal and vertical components of each motion vector (one per partition) are halved when applied to the chroma blocks.

3. THE PROPOSED MACROBLOCK PARTITION MODE DECISION ALGORITHM

In this section, we introduce the proposed macroblock partition mode decision algorithm aiming to speed-up the inter-frame prediction. This goal is achieved by making use of the MPEG-2 MB coding mode and the mean and variance of the residual information for this MB calculated for its 4x4 sub-blocks (see Figure 2.a, 2.b and 2.c). MPEG-2 uses 16x16 motion compensation (MC) and does not temporally decorrelate an image in full. The MC residual can thus be exploited to understand the temporal correlation of variable block sizes in H.264. The open source WEKA [8] data mining tool is used to discover a pattern of mean, variance, and MPEG-2 coding modes for H.264 coding mode decisions. The output of the tree is the H.264 MB coding mode (intra, skip, 8x8 and 16x16).

The decision trees were made using the WEKA data mining tool. We use the J.48 algorithm [9] and a training set consisting of over 1000 macroblocks. The macroblocks were obtained from the Flower Garden MPEG-2 sequence encoded at 5 Mbps, with only I and P frames. We found that this training set captures the mean and the variance distribution of the residual in MPEG-2 coded video sequences.

The inputs for this node are skip-mode MBs in the MPEG-2 bitstream (coming from node 1), or new candidates to be skipped as proposed by the node 3. This node evaluates only the H.264 16x16 mode (without the modes 16x8 or 8x16). Then, the node selects the best option, skip or inter 16x16.

We use two WEKA decision trees, shown in the picture using grey balls. The first tree is used to check for the skip mode MBs. If an MB is not skipped, a second decision tree is used for selecting the mode of the MB. At this level, the MB can only be 8x8 or 16x16. The decision tree works as follows:

Node 1. The first level of the decision only depends on the MB mode in MPEG-2. The following rule is applied in this node:

a. If the MPEG-2 MB was coded in intra mode, then the MB will be coded as intra in H.264.
b. If the MPEG-2 MB was coded in skip mode, then the decision will be made in the node 2.
c. For all other MPEG-2 MB coding modes the decision will be made in the node 3.

Node 2. The inputs for this node are skip-mode MBs in the MPEG-2 bitstream (coming from node 1), or new candidates to be skipped as proposed by the node 3. This node evaluates only the H.264 16x16 mode (without the modes 16x8 or 8x16). Then, the node selects the best option, skip or inter 16x16.
Node 3. The inputs for this node are all the other MPEG-2 coded MBs (MBs other than intra and skip modes). In this node we use a tree decision generated with WEKA to decide whether the MB should be coded as a skip MB in H.264. If the answer is no, the MB coding mode decision continues in the node 4.

Node 4. The inputs for this node are the non-skipped MBs coming from the node 3. This tree examines whether the MB has a very high residual or a medium residual. The output of this node is the mode that will be use for coding the MB: 8x8 or 16x16.

The WEKA tool was used to determine two sets of mean and variance thresholds for the MPEG-2 residual, one for skip or non-skip decision in node 3 and, another one for 8x8 or 16x16 decision in node 4. Due to space constraints we cannot show all the rules that are evaluated in the WEKA decision nodes. Each decision node implements a decision tree that examines the mean and variance of MPEG-2 MC residual in 4x4 sub-blocks to arrive at the decision. A decision tree can be made by first creating a file with the 16 mean and variance values for each macroblock, the MPEG-2 MB coding mode, and the corresponding H.264 MB coding mode decision for that MB as determined by the standard reference software. Based on this training file, the WEKA tool is used to create a decision tree with a set of rules for using the mean and variance of the 4x4 sub-blocks. Since the MB mode decision, and hence the thresholds, depend on the quantization parameter (QP), the decision tree thresholds are computed for a mid-QP of 25 and then adjusted for other QPs. Since the quantization step size in H.264 doubles when QP increases by 6, the thresholds are adjusted by 12.5% for a change in QP of 1.

The proposed algorithm only works with the first level of the decision. It selects if the macroblock will be coded as skip, intra, 8x8 or 16x16. Then, for the second level decision, the proposed mechanism uses the reference software method: running all the possibilities and selecting the best sub-partition. Work is ongoing to develop decision trees for sub-partition decisions.

4. PERFORMANCE EVALUATION

In order to evaluate the proposed macroblock partition mode decision algorithm, we have implemented the proposed approach based on the H.264 reference software [10] (version 10.1). The metrics we have been interested in are the computational cost and rate distortion function. Throughout our experiments, we have used various video sequences exhibiting different spatial characteristics and different size formats (CCIR, CIF and QCIF). We use Q factors from QP=0 to QP=50 (corresponding to the full H.264 QP range). The size of the GOP is 12 frames; where the first frame of every GOP was encoded as I-frame, and the rest of the frames of the GOP were encoded as a P-frame. The rate control was disabled for all the simulations.

Figure 2.e and 2.f shows the differences between the inter mode selection made by the H.264 standard, and the proposed algorithm, with a value of 25 for QP.

Figure 3 shows the RD results of applying the H.264 full estimation algorithm and the proposed macroblock partition mode decision algorithm. As seen from the figure, the PSNR obtained when applying our algorithm deviates slightly from the results obtained when applying the considerable more complex full estimation procedure. In same cases we have better results than the standard. This happens because the best solution is obtained by enabling the RD Optimization, and in the experiments reported in the figures we are comparing the faster configuration of a H.264 encoder (SAE cost) with our proposed reduced-complexity encoder. We are working in a real-time scenario and are trying to see how can reduce the time of the faster H.264 encoder (with the SAE cost as evaluation function), without loss in PSNR.

Table 1 shows the computational cost, with respect to H.264 full estimation procedure for encoding the complete sequence of 200 frames (Total encoding time, Motion Estimation time and The Decision Tree time). We used a QP factor between 10 and 30 for different video sequences and
size formats, showing over 30% reduction in the computational complexity characterizing our proposed scheme. The simulations were running in a P4 at 3.0 GHz Intel machine with 512 MB RAM. Our results show that the proposed algorithm is able to maintain a good picture quality while considerably reducing the number of operations to be performed.

Table 1. Computational Cost in seconds.

<table>
<thead>
<tr>
<th>Total encoding time</th>
<th>ME</th>
<th>Mo/Ver/Tree</th>
</tr>
</thead>
<tbody>
<tr>
<td>SQ, QP H.264 Proposed H.264 Proposed H.264 Proposed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hook, 10 416.7 355.3 281.9 259.7 0 0.9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hook, 15 412.1 346.6 301.5 269.0 0 0.9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hook, 20 413.8 349.0 307.6 274.7 0 0.7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hook, 25 413.0 345.3 311.6 271.3 0 0.7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hook, 30 419.0 351.1 319.0 285.7 0 0.7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Football, 10 121.4 82.9 88.6 59.7 0 0.3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Football, 15 122.5 79.4 88.6 58.9 0 0.3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Football, 20 123.0 79.5 92.0 59.2 0 0.3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Football, 25 123.3 73.7 93.7 55.2 0 0.3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Akiyo, 10 28.2 20.4 23.0 18.5 0 0.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Akiyo, 15 28.3 21.0 21.3 17.0 0 0.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Akiyo, 20 28.4 20.9 19.7 16.1 0 0.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Akiyo, 25 28.6 19.8 19.0 15.6 0 0.0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

5. CONCLUSIONS

In this paper, we proposed a novel macroblock partition mode decision algorithm for inter-frame prediction to be used as part of a high-efficient MPEG-2 to H.264 transcoder. The proposed algorithms used data mining techniques to exploit the temporal correlation in the MPEG-2 MC residual. The WEKA tool was used to develop decision trees for H.264 coding mode decision. The proposed algorithm has very low complexity as it only requires the mean and variance of the MPEG-2 residual and a set of rules to compare the mean and variance against a threshold. Our results show that the proposed algorithm is able to maintain a good picture quality while considerably reducing the computational complexity by 30% on average. We are currently working on algorithms, based on the proposed data mining approach, for sub-partition decisions in the H.264 inter MB coding, and we expect to reduce an additional 30% on it. The information of the motion vectors is not being used. We expect to further reduce the transcoding time by reducing the search range.

6. REFERENCES


Figure 3. Rate Distortion Results. (a) CCIR. (b) CIF. (c) QCIF.