IMPROVING H.264/AVC INTER COMPRESSION WITH MULTISCALE RECURRENT PATTERNS

Nuno M. M. Rodrigues[1,2], *Eduardo A. B. da Silva*[3], *Murilo B. de Carvalho*[4], *Sérgio M. M. de Faria*[1,2], *Vitor M. M. da Silva*[1,5]

[1]Instituto de Telecomunicações, Portugal; [2]Instituto Politécnico Leiria - ESTG, Portugal; [3]PEE/COPPE/DEL/Poli, Univ. Fed. Rio de Janeiro, Brazil; [4]TET/CTC, Univ. Fed. Fluminense, Brazil; [5]DEEC, Universidade de Coimbra, Portugal.

e-mails: nuno.rodrigues@co.it.pt, eduardo@lps.ufrj.br, murilo@telecom.uff.br, sergio.faria@co.it.pt, vitor.silva@co.it.pt

ABSTRACT

In this paper we describe the ongoing work on a new paradigm for compressing the motion predicted error in a video coder, referred to as MMP-Video. This new coding algorithm uses the Multidimensional Multiscale Parser image coding algorithm to encode the residue error, in a H.264/AVC based video coder.

MMP has shown to perform very well as a universal still image coding method, particularly when it is combined with Intra prediction schemes. In addition, previously published preliminary results have also presented MMP as a promising video coding method.

In this paper, we propose new dictionary updating techniques for MMP-video. Along with other functional optimizations, these techniques allow for a significant improvement in the encoder performance. Thus, we were able to achieve considerable gains over H.264/AVC for B slices, specially for medium and high bit-rates, while maintaining equivalent performance for the P slices.

Index Terms— Video coding, approximate pattern matching, Multidimensional Multiscale Parser

1. INTRODUCTION

Compression methods based on the transform and quantization paradigm have been dominant for both Intra and Inter residue coding in hybrid video compression. Despite the development of highly effective prediction techniques, the efficient compression of the predicted residues remains an important factor for the overall efficiency of the coder. This fact has motivated the development of an adaptive block size, integer version of the DCT, that is used as the residual encoding method for the most recent video coding standard, the H.264/AVC [1] (hereby refered to as H.264).

In previous works [2, 3] it has been shown that, due to its universal character, MMP is a good alternative to transform-quantization based state-of-the-art image coding methods. For image coding, MMP is able to achieve a very good performance not only for smooth images, but also for other types of images, like text, graphics and compound images, where it outperforms state-of-the-art still image coders, like JPEG2000 and H.264-Intra.

In [3], the MMP's coding efficiency for intra prediction residuals was demonstrated. Such results have motivated the development of MMP-Video, that uses MMP for the compression of the motion estimated residual data, in an H.264 video coding framework. The initial results obtained were encouraging [4], but they have also shown that MMP-video still presented minor quality losses when compared with H.264.

In this paper we present some results and conclusions of the most recent studies on MMP-Video, that focused in the use of new dictionary updating techniques, along with the optimization of some practical aspects of MMP for video coding, mainly for the B slices. Such dictionary updating techniques resulted from an adaptation of some techniques, proposed in [5], to encode video signals.

Experimental results, presented in this paper, show a clear performance improvement when compared with the method described in [4]. With the proposed modifications, MMP-Video is able to achieve better results than H.264's high profile for B slices, while maintaining an equivalent performance for P slices.

Section 2 summarizes the main aspects of the original MMP algorithm, while section 3 introduces the MMP-Video encoder and describes the improvements that led to the new version of MMP-Video. Section 4 compares the performance of the new MMP-Video coder with that obtained by H.264 and by using the method described in [4]. Finally, section 5 presents some conclusions of this work and highlights the foreseen steps for further developments.

2. THE MMP ALGORITHM

MMP uses *pattern matching with scales* to encode each original image block with a vector from an *adaptive dictionary* $\mathcal{D}$. The use of scales means that blocks of different dimensions can be approximated by this procedure.

The different block dimensions correspond to successive binary segmentations of an original square block, first in the vertical, then in the horizontal direction, that are represented by a binary tree $\mathcal{T}$. A block X^l, belonging to $level\ l$ of the segmentation tree, has dimensions $(2^{\lfloor \frac{l+1}{2} \rfloor} \times 2^{\lfloor \frac{l}{2} \rfloor})$.

The MMP algorithm can be summarized as described below. For each block X^l of the original image:

1. find the dictionary element S_i^l that minimizes the Lagrangian cost function of the approximation, given by: $J(\mathcal{T}) = D(X^l, S_i^l) + \lambda R(S_i^l)$, where $D()$ is the sum of square differences (SSD) function and $R()$ is the rate needed to encode the approximation;
2. parse the original block into two blocks, X_1^{l-1} and X_2^{l-1}, with half the pixels of the original block;
3. apply the algorithm recursively to X_1^{l-1} and X_2^{l-1}, until level 0 (1×1 block) is reached;
4. based on the values of the cost functions determined in the previous steps, decide whether to segment the original block or not;
5. if the block should not be segmented, use vector S_i^l of the dictionary to approximate X^l;
6. else
   (a) create a new vector S_{new}^l from the *concatenation* of the vectors used to approximate each half of the original block: X_1^{l-1} and X_2^{l-1};
   (b) use S_{new}^l to approximate X^l;
   (c) use S_{new}^l to *update* the dictionary, making it available to encode future blocks of the image.

The binary tree is encoded as a string of binary symbols: '0' for the tree nodes, or block segmentations, and '1' for tree leafs, or unsegmented blocks, using a top-bottom approach.

The final MMP bit-stream is composed by the segmentation tree symbols plus the indexes of the vectors of the dictionary that should be used for each sub-block. These items are encoded using an adaptive arithmetic encoder.

As can be seen in step 6 of the previous algorithm, every block of level l that is segmented originates the concatenation of two dictionary blocks of level $l-1$. The resulting block is used to update the dictionary, becoming available to encode future blocks of the image, independently of their size. This process uses only information that is also available at the decoder, where the dictionary update procedure is replicated.

MMP uses a separable scale transformation T_N^M to adjust the vectors' sizes before attempting to match them. This allows the matching of vectors of different lengths: to approximate an original block X^l using one block S^k of a different scale of the dictionary, MMP first determines $S^l = T_k^l[S]$.

Detailed information on these and other aspects of MMP can be found in [2].

3. VIDEO ENCODING WITH MMP

In this section we first present a brief description of the main aspects of MMP-Video algorithm and compare them with the H.264 standard. After this we describe the dictionary design techniques and the other relevant improvements to the method that were developed in our recent study.

The MMP-Video implementation is based on the JM9.3 H.264 reference software [6]. It shares the same structure, but it uses MMP to encode the motion compensated residue data of the Inter macroblocks (MBs), instead of the integer version of the DCT (hereby referred to as DCT) defined in [1].

In the MMP-Video version proposed in this paper, all Intra MBs are encoded using exactly the same procedure has in H.264, including the DCT for Intra residue data coding. This model was chosen so that we could effectively compare the performances of H.264 and MMP-Video for coding motion compensated residual errors (this would not be the case if the Intra reference frames used for the two encoders were different). A detailed comparison between MMP and H.264 for Intra frame coding can be found in [3].

For Inter MB coding, MMP-Video uses the same seven modes defined for H.264, related to the segmentation of a 16×16 luma MB. For these MBs, the encoder first has to optimize the motion vectors (MVs) for each specific mode M_i, determine the motion compensated residuals for that mode and then evaluate the corresponding Lagrangean cost function, J_{M_i}. When this process is completed for all available modes, the encoder chooses the mode with the smaller cost.

After this the encoder determines once more the motion compensated residue block, using the best MVs and the best mode. It then encodes a set of data related to the MB type and prediction (as the used coding mode, the reference frames and the MV for each partition) followed by the encoding of the motion compensated residual error. In this stage lies the main difference between the MMP-Video and the H.264 encoders: H.264 uses a adaptive block size (ABS) version of the DCT transform while MMP-Video uses the MMP algorithm, previously described.

As was previously referred, the existence of several partition modes allows the motion compensation to be performed using a block size that optimizes the RD cost function. In the motion estimation step, H.264 estimates the cost J for each MV by evaluating the distortion of the transform coding of the residues, either by using the sum of absolute differences (SAD) or the sum of absolute transformed differences (SATD).

MMP-Video inherited this procedure, i.e., the MVs are chosen based on the energy of the motion compensated error. This is sub-optimal from the MMP coding point of view, because the residue block with minimal SAD or SATD is not necessarily the block that is more efficiently encoded by MMP. Nevertheless, our simulations have demonstrated that the use of these error measures still allows MMP-Video to perform efficiently, with a significant reduction in computational complexity. This also has the advantage of allowing for a fair comparison between the encoding efficiency of MMP and the DCT, because the residue patterns, generated by the motion compensation process, tend to be approximately the same for both encoders.

In H.264 the block size for transform coding depends on the partition size that was used for the motion estimation. This enables significant gains when compared with the fixed size blocks (either 4×4 or 8×8) used by its predecessors. Such scale adaptability allows saving bits by the use of large blocks where the residue is mostly uniform, while providing good coding accuracy through the use of small blocks where the residue data is more detailed. Nevertheless, MMP-Video does not use this ABS feature explicitly. Instead, it considers the entire 16×16 residue block (8×8 for the chroma residues) built from the concatenation of the motion compensated sub-block residues. Notice, however, that this is not a problem for MMP due to its strong inherent scale adaptability.

Apart from of the motion compensated residual data, all information transmitted by MMP-Video is encoded by the same techniques used in H.264. The only exception is the MB's coded block pattern (CBP) that is not used by MMP, and consequently is not transmitted by MMP-Video for Inter MBs.

3.1. Improvements in Dictionary Updating and Coding

In its initial version [4], MMP-Video coder used six independent dictionaries to encode the MBs of each of the YUV components of the P and B slices. This has the potential advantage of allowing each dictionary to be adapted to the specific prediction patterns of each type of source data.

However, a detailed analysis of the algorithm's behaviour demonstrated that this was not the case for the chroma's dictionaries. Since chroma residue blocks tend to be very smooth, the number of segmentations performed by MMP for these blocks is small. This has a significant impact in the growing rate of the MMP dictionaries and therefore in the adaptability of the method.

The proposed method only uses two independent dictionaries: one for all three components of the P slice's MBs and another for all components of the B slices' MBs. Thus, when a chroma MB is encoded, MMP uses a richer dictionary, that resulted from the coding of the luma MBs, which improves the coding efficiency of the chroma components.

Although in this case the luma MBs have to "share" the dictionary with the chroma patterns, causing an efficiency loss for the luma component, this configuration achieved the best overall results among all those tested; the loss for the luma components is compensated by the gains achieved for the chroma components.

Our recent work on the dictionary updating aspects for the MMP image coder demonstrated that the use of distortion controlled techniques in the dictionary update stage of the algorithm allows for an overall performance gain. MMP initializes each dictionary with a few blocks with constant value, but the updating procedure quickly adapts the dictionary, by inserting the patterns used by the encoder.

For image coding, practical tests have shown that the final dictionary size tends to grow linearly with the final bit rate. For higher rates, the number of blocks that are actually used to encode a segment is much smaller than the final size of each dictionary. Since each new block generates a new index to encode, dictionaries over populated with blocks that are not useful, tend to decrease the overall performance.

The new dictionary design algorithm is based on redundancy reduction: before inserting a new block in the dictionary, MMP determines the smallest quadratic distortion between the new block and the ones already available in the dictionary. If this distortion (normalized by the number of pixels of the block) is below a given threshold, the dictionary is not updated, guaranteeing a "minimum distance condition" between all the blocks in the dictionary. The value for the best distortion threshold depends on the target rate (and therefore can be related to the value of λ). A detailed description of this procedure is presented in [5].

An analysis of the MMP-Video coding data revealed that a large number of null 16×16 and 8×8 blocks were used for for encoding B slices. This is a source of inefficiency, since for such null blocks MMP has to transmit one no-segmentation flag followed by the index that corresponds to the null block, for each of the three components. The solution to this problem is the use of a binary flag, for Inter MBs on B slices, that signals the existence on non-zero residue blocks for the YUV components. If this is not the case, than the encoder simply does not transmit any MMP information for the current MB, thus saving bits for the same reconstruction quality.

4. EXPERIMENTAL RESULTS

The new MMP-Video encoder was compared with the H.264 high profile video and with the method in [4]. Figures 1 to 3 show the results for the P and B slices of the first 99 frames of the CIF Foreman sequence, 4:2:0. The encoded sequences used a IBPBP pattern with only one I frame. The variable bit rate mode was used and the encoders were tested for several quality levels of the reconstructed video sequence, by fixing the QP parameter for the I/P and B slices.

In figure 1 it is clear that the proposed method represents an improvement over the method in [4]. This is especially evident for the chroma components: up to 1 dB for B frames and 0.5 dB for P frames (not presented due to space limitations).

When we compare the new MMP-Video coder with JM9.3 H.264 high profile (figures 2 and 3) we observe a clear improvement of the reconstructed frame quality, specially for B slices at medium to high bit rates. For P slices, MMP-Video achieves an equivalent performance to H.264.

The good results for the B-slices are an indication that the coding of motion compensated residuals takes good advantage of the adaptability of MMP, that is able to "learn" the typical patterns of motion compensated residue blocks and store them in the adaptive dictionary.

The pattern matching procedure, used by MMP-Video, is, however, more complex than the integer transform used by H.264, which causes an increase in the computacional complexity. Procedures to reduce this complexity are been studied and will be evaluated after the method has been optimised for encoding efficiency.

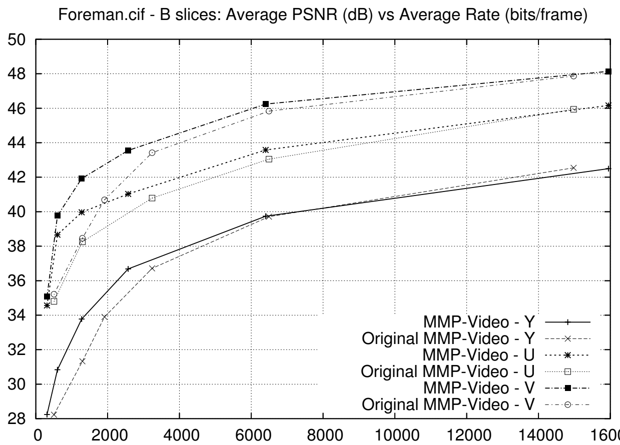

Fig. 1. Original vs Current MMP-Video RD plots for B slices of Foreman CIF sequence.

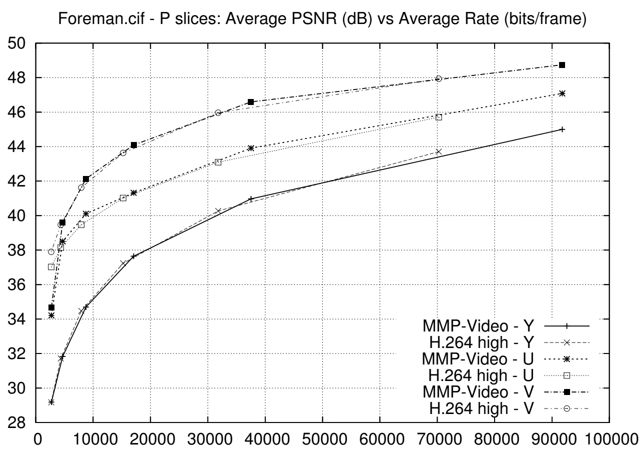

Fig. 2. MMP-Video vs H.264 RD plots for P slices of Foreman CIF sequence.

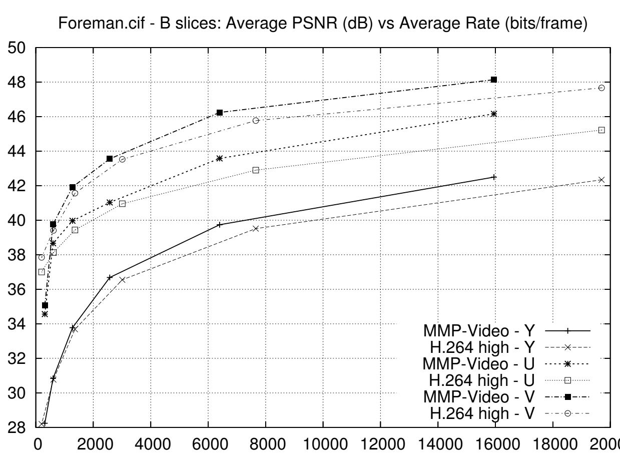

Fig. 3. MMP-Video vs H.264 RD plots for B slices of Foreman CIF sequence.

5. CONCLUSIONS

This paper presents the results of our most recent studies in the development of a new video encoding algorithm, that uses MMP to encode motion predicted residual data, that we refer to as MMP-Video.

We have presented new dictionary updating techniques and some functional improvements, which result in a significant gain in coding performance of the chroma components (up to 1 dB), when comparing the experimental results of the current encoder with those of the initial method.

When compared to H.264 high profile, coding results for MMP-Video showed a consistent improvement for the coding of B slices. For medium to high bit rates, this improvement is more than 1 dB in final average PSNR for some components of the B slices. For P slices, MMP-Video consistently matches the coding performance of H.264 high profile.

Our results show that MMP is a new and exciting paradigm for video coding, whose results top the ones of state-of-the-art encoder H.264. Nevertheless, this method still allows a lot of room for improvement. Further research is undergoing on topics like coding efficiency for P slices' MBs and further MMP optimizations for encoding video sequences.

6. REFERENCES

[1] Joint Video Team (JVT) of ISO/IEC MPEG & ITU-T VCEG (ISO/IEC JTC1/SC29/WG11 and ITU-T SG16 Q.6), *Draft of Version 4 of H.264/AVC (ITU-T Recommendation H.264 and ISO/IEC 14496-10 (MPEG-4 part 10) Advanced Video Coding)*, March 2005.

[2] M. de Carvalho, E. da Silva, and W. Finamore, "Multidimensional signal compression using multiscale recurrent patterns," *Elsevier Signal Processing*, no. 82, pp. 1559–1580, November 2002.

[3] N. M. M. Rodrigues, E. A. B. da Silva, M. B. de Carvalho, S. M. M. de Faria, and Vitor M. M. Silva, "Universal image coding using multiscale recurrent patterns and prediction," *IEEE International Conference on Image Processing*, September 2005.

[4] N. M. M. Rodrigues, E. A. B. da Silva, M. B. de Carvalho, S. M. M. de Faria, and Vitor M. M. Silva, "H.264/AVC based video coding using multiscale recurrent patterns: First results," *Lecture Notes on Computer Science - VLBV 05*, vol. 3893, pp. 107–114, February 2006.

[5] N. M. M. Rodrigues, E. A. B. da Silva, M. B. de Carvalho, S. M. M. de Faria, Vitor M. M. Silva, and Frederico Pinagé, "Efficient dictionary design for multiscale recurrent patterns image coding," *ISCAS 2006 IEEE International Symposium on Circuits and Systems*, May 2006.

[6] http://iphome.hhi.de/suehring/tml/download/.