MAPPING MOTION VECTORS FOR A WYNER-ZIV VIDEO TRANSCODER

Eduardo Peixoto[1], *Ricardo L. de Queiroz*[1] *and Debargha Mukherjee*[2]

[1] Universidade de Brasilia, Brasilia, Brazil, eduardo@image.unb.br, queiroz@ieee.org
[2] Hewlett Packard Laboratories, Palo Alto, USA, debargha.mukherjee@hp.com.

ABSTRACT

Wyner-Ziv (WZ) coding of video utilizes simple encoders and highly complex decoders. A transcoder from a WZ codec to a traditional codec can potentially increase the range of applications for WZ codecs. We present a transcoder scheme from the most popular WZ codec architecture to a DPCM/DCT codec. As a proof of concept, we implemented this transcoder using a simple pixel domain WZ codec and the standard H.263+. The transcoder design aims at reducing complexity, since the transcoder has to perform both WZ decoding and DPCM/DCT encoding, including motion estimation. New approaches are used to map motion vectors for such a transcoder. Results are presented to demonstrate the transcoder performance.

Index Terms— Wyner-Ziv coding, transcoder, H.263+, distributed video coding.

1. INTRODUCTION

Distributed video coding (DVC) has been the focus of many studies in recent years. DVC codecs are known to present low-complexity encoders and high-complexity decoders, which makes it suitable for encoding video where computational resources are scarce. However, as DVC decoders are not indicated for a constrained-resources environment, usual applications for a DVC codec are applications where decoding can be made on a more resourceful machine or offline. DVC rests on the Slepian-Wolf and Wyner-Ziv (WZ) theorems [1, 2], and a good review on DVC can be found elsewhere [3]. Some solid frameworks for DVC codecs were proposed before: a pixel-domain WZ codec [4], a transform-domain WZ codec [5], the PRISM framework [6] and the DISCOVER codec [7]. In order to enable a better performance, WZ codecs usually have a feedback channel between the decoder and the encoder. An analysis of the feedback channel for pixel-domain WZ codecs can be found in [8], and some alternatives do exist [9, 10].

To enable the use of DVC codecs in a scenario where low complexity is required at both ends, one might use a transcoder from a WZ scheme to a traditional scheme. A transcoder is also needed when content encoded with a WZ codec has to be transmitted in a network that works with another standard. Furthermore, since WZ codecs usually have a lower performance than traditional codecs, a transcoder could be used for storage purposes. Hence, a transcoder might expand the applications for a WZ codec.

This work was supported by HP Brasil and by CNPq under grant 47.3696/2007-0.

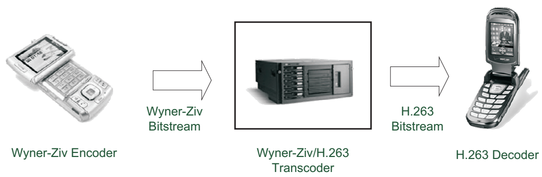


Fig. 1. Mobile video communications architecture.

Fig. 1 illustrates the network process when the transcoder is used to provide low complexity mobile communication. The transmitter uses a WZ encoder, transmitting the data to a server. This acts as a transcoder, changing the sequence to another format, which requires a less complex decoder. The transcoded data is sent to the receiver, the addressee of the video content. As a single server might handle several communications, the complexity of the transcoder is an issue and has to be considered in its design.

Even though the need for a transcoder has been mentioned before [3] and a simple implementation has also been presented [11], we propose a more efficient and universal transcoder from a WZ codec to a traditional codec. Its main idea is to re-map the motion vectors already calculated in the side information (SI) generation step in order to use them to encode video content using a traditional codec. As a proof of concept, the transcoder is implemented using a simple Pixel-Domain WZ codec and H.263+ as the traditional codec.

2. THE CODECS USED

The source WZ codec chosen is a pixel-domain Wyner-Ziv codec [12], for its simplicity and popularity. Many other WZ implementations are very similar to this one, such as the transform-domain WZ codec [5], among others.

The SI generation method used here is fully described in [13]. It performs motion estimation to find the best motion vectors (MV) between two key frames. Then, for each macroblock, it keeps the reference block and displaces it by half the MV to create the SI frame. Hence, the SI frame has overlapped and uncovered areas. In order to minimize the uncovered areas, backwards motion estimation is performed between the neighbour key frames and a new SI frame is generated. Simple averaging is used to combine the two SI frames. After this, there are still uncovered areas in the SI frame. The SI frame is then divided into macroblocks and, if a given macroblock contains uncovered areas, bi-directional motion estimation is performed for this macroblock. The uncovered area is not considered when calculating the SAD. Once the new reference block is found, it is used to replace the macroblock with the gaps. However, only the uncovered area is replaced. The remaining macroblock pixels are left untouched.

3. THE TRANSCODER

Instead of re-encoding the sequence, the transcoder uses information that was calculated in the WZ decoding process to speed up the transcoding. The main ideas are to copy the bitstreams of the intra frames that will remain I-frames in the transcoded sequence, and to reuse the motion estimation performed in the SI generation process instead of performing a new motion estimation.

There are two main degrees of freedom for the transcoder: the GOP structure and the motion vector mapping.

3.1. Defining the GOP for the transcoder

The GOP size of the transcoded sequence does not need to be of the same size as the original WZ sequence. Instead, it can be changed to a wide range of GOPs. Any GOP size which is an integer multiple of the original GOP size is possible. It is also possible to use *B*-frames in the transcoded sequence.

Some of these options are depicted in Fig. 2. In the top row, we illustrate a WZ coded sequence. The key frames were encoded as H.263+ intra frames by the WZ encoder, and their bitstreams are to be passed through virtually untouched. The motion vectors are mapped using a function $g(\cdot)$ which is explained in Sec. 3.2.

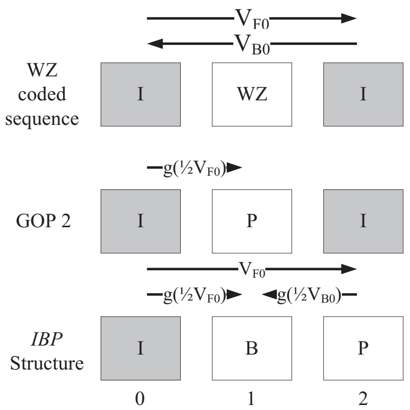


Fig. 2. Options of the GOP for the transcoded sequence starting with a WZ GOP of 2 frames. The motion vectors are mapped using a function $g(\cdot)$, defined in Eq. 1.

If the transcoded sequence has a GOP length of 2, it has the form of the second row of Fig. 2. The *I*-frames have the same structure of its pairs in the WZ coded sequence and are just passed through. The last row in Fig. 2 refers to the use of B-frames. As the SI generation process calculates forward and backward motion vectors, it is easy to relate them to B-frames. For a GOP length of 2 in the WZ coded sequence, the motion vectors can be used to generate a stream of *IBPBP...*-frames. For such a structure, the motion vectors for the *P*-frames are readily available from the SI generation process. The *B*-frames, however, require a more sophisticated approach, which is described in Sec. 3.2.

3.2. Mapping Motion Vectors

H.263+ performs motion estimation and uses the motion vectors to generate the compensated frame, which is used as a prediction to encode the current frame. In the transcoder, there is motion information, generated by the WZ decoding process. However, this motion information were not calculated for the current WZ frame. Mapping the motion vectors available to use for the WZ frame avoids the computation of new motion estimations. If more WZ frames are added to the sequence, the complexity of the transcoder is further reduced, compared to the trivial approach.

In order to explain the mapping process for the motion vectors, we have to introduce some notation. Let x be the sequence of decoded key frames and $x(\mathbf{p}, k)$ be a pixel in it, where $\mathbf{p} = [p_i\ ,\ p_j]$ is the spatial location and k is the temporal location (i.e., the frame number). Each frame can be divided into macroblocks of 16×16 pixels. Thus, $x(16\mathbf{m} + \mathbf{n}, k)$ represents each macroblock within the *k*-th frame, where $\mathbf{m} = [m_i\ ,\ m_j]$ is the macroblock index and $\mathbf{n} = [n_i\ ,\ n_j]$ is the pixel index within the block.

We want to create a grid for the SI frame and to try to decide, for each macroblock, the best motion vector among those available within it. Note that the spatial position $16\mathbf{m} + \mathbf{n} - \left\lfloor \frac{\mathbf{v_m}}{2} \right\rfloor$ (i.e., the position pointed by the motion vector halved) does not necessarily fit the macroblock grid for the SI frame y. So, the same pixel can be pointed by two or more motion vectors. Also, different regions of the same macroblock might be filled by different motion vectors, making it difficult to decide which motion vector to use.

Thus, for a given macroblock, there are pixels with zero, one or more motion vectors. Yet, we have to decide for a single motion vector $\hat{\mathbf{v}}$ for the whole macroblock. One solution would be to simply average the motion vectors for the uniquely defined pixels, denoted as $\mathbf{v}_{mean}$. However, it is possible that $\mathbf{v}_{mean}$ does not represent any group of motion vectors. So, we propose a convex solution, in the form of:

$$\hat{\mathbf{v}} = g(\mathbf{v}_l) = \{\mathbf{v}_l | l = min(\|\mathbf{v}_{mean} - \mathbf{v}_l\|)\} \qquad (1)$$

in other words, $\hat{\mathbf{v}}$ is the motion vector in the group that is closest to $\mathbf{v}_{mean}$.

Note that $g(\cdot)$ operates on the motion vector set scaled to the represent the motion vector for a given frame. For example, when used in a GOP length of 2, the motion vectors $\mathbf{v}_l$ were calculated using the frames in positions $2k$ and $2k+2$. Thus, what is used for the WZ frame in position $2k+1$ is the motion vector $g(1/2 * \mathbf{v}_l)$. (as it can be seen in Fig. 2).

An advantage of this process is that it assigns motion vectors based on pixels, not blocks. So, we can group the motion vectors in different ways. For example, we can group the motion vectors in 8×8 blocks even though they were originally calculated in 16×16 blocks. The 4 motion vectors for 8×8 blocks inside a 16×16 block are not necessarily equal to the single motion vector for the 16×16 block, since overlaps may have occurred in this area. Hence, we can have an 8×8 motion vector set with a search performed within 16×16 blocks. This is depicted in Fig. 3.

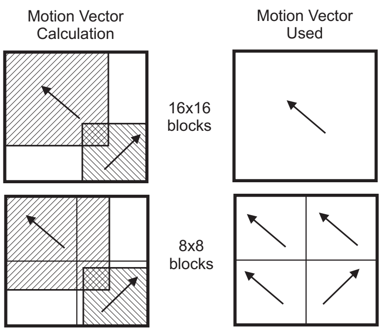


Fig. 3. Calculating the MV for the transcoder. When there are overlapping areas, the case of 8×8 blocks allows for extra flexibility when choosing the output MV.

If the WZ frame is transcoded to a *B*-frame, both the forward and backward motion vectors are chosen using (1). In all cases, the transcoder uses the best motion vector between $\hat{\mathbf{v}}$ and the null motion vector (i.e., the $(0,0)$ one).

For some macroblocks, there is still another motion vector to consider. Back to the SI generation process, in order to fill the blank areas, a new motion estimation is carried for the macroblocks in which they appear. Then, for these macroblocks, we did carry motion vector search. This motion vector might be a better choice than the motion vector given by (1), provided this information has already been calculated in the SI generation process. Thus, for those macroblocks, said motion vector is used instead of $\hat{\mathbf{v}}$.

3.3. Refining the Motion Vectors

A good way to improve the motion vectors inherited from the WZ decoding process is to use them as seeds for a fast motion estimation method, such as a diamond or hexagonal search. Instead of always starting at motion vector $(0,0)$, the refinement would begin at the motion vector calculated in Sec. 3.2. This seed is compared (in terms of SAD) with the null motion vector, and the search starts at the best of these two. This approach has proven to lead to the testing of less candidates.

4. EXPERIMENTAL RESULTS

We evaluated the proposed transcoder using the WZ codec [12] and the standard H.263+. For all tests, the PSNR is shown as an average of the PSNR of the luminance component only, while the rate also considers the chrominance components. The sequences used are in CIF resolution at 30 fps and QCIF resolution at 15 fps and the first 299 and 149 frames were used, respectively. The test sequences at 15 fps were obtained by discarding every even frame of the original sequences at 30 fps.

The tests made to evaluate the transcoder effectiveness compare three options: (i) the trivial transcoder (i.e., the tandem connection of a decoder with a full encoder) with full motion estimation, (ii) the proposed transcoder, and (iii) the proposed transcoder with motion vectors refinement. For each case, the motion vectors for the H.263+ stream are chosen as follows. (i) The integer precision motion vectors are chosen with a full search with search range $[-16, 15]$ and half-pixel refinement on the 8 neighbors of the chosen motion vector. (ii) The transcoder tests two motion vectors (the null motion vector and $\hat{\mathbf{v}}$) and performs the half-pixel refinement on the 8 neighbors of the best of these two. (iii) The transcoder performs a diamond refinement starting at the null motion vector or $\hat{\mathbf{v}}$ (wichever has a lower SAD), and then performs the same half-pixel refinement.

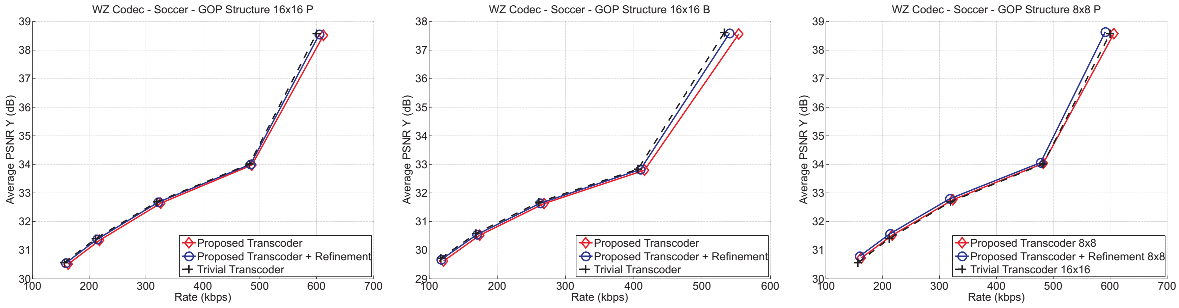
The results for the transcoder are shown in Fig. 4 and 5. As it can be seen, the proposed transcoder closely matches the performance of the trivial transcoder for all sequences and GOP structures tested.

The proposed transcoder is also tested using the optional 8×8 motion vectors in a GOP structure of *IPIP*. Since the H.263+ standard does not allow 8×8 MV for *B*-frames, this option was not used in the *IBPBP.*. GOP structure. The results are shown in Fig. 4(c) through 5(c). In this case, the proposed transcoder has a performance very close to the trivial transcoder. For some sequences, the latter is even outperformed by the former. As no search is made for 8×8 blocks, the transcoder complexity does not significantly change when using this mode. Therefore, when the 8×8 mode is available, it should be used instead of the 16×16 one.

5. CONCLUSIONS

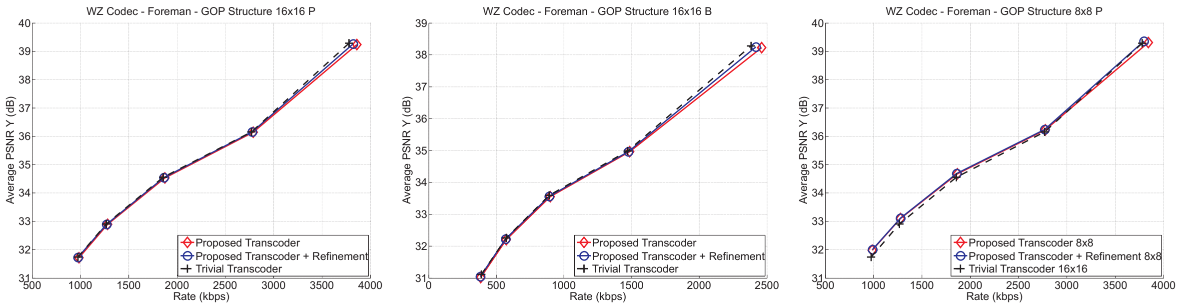
We proposed a transcoder from streams generated by WZ codecs to those generated by traditional DPCM/DCT video codecs. As a proof of concept, the transcoder was implemented and tested using a simple pixel-domain WZ codec and H.263+ as the regular codec. However, the same transcoder architecture can be used by other popular WZ codecs, such as the DISCOVER codec, and can also be modified to work with other standards, such as MPEG-2 and H.264, since the proposed transcoder is based on reusing motion information.

The performance of the proposed transcoder is virtually the same as decoding and re-encoding the whole sequence,



(a) (b) (c)

Fig. 4. Results of the transcoder for the *Soccer* sequence at QCIF resolution and 15 fps using different GOP structures. The original WZ coded sequence has a GOP length of 2. (a) GOP: *IPIP...*; (b) GOP: *IBPBP...* ; (c) GOP: *IPIP...* using 8×8 MV.



(a) (b) (c)

Fig. 5. Results of the transcoder for the *Foreman* sequence at CIF resolution and 30 fps using different GOP structures. The original WZ coded sequence has a GOP length of 2. (a) GOP: *IPIP...*; (b) GOP: *IBPBP...* ; (c) GOP: *IPIP...* using 8×8 MV.

at a much lower computational cost. For a GOP structure of *IPIP*, the 8×8 transcoding mode even outperforms the regular 16×16 full search mode, without additional complexity cost. The transcoder is also adjustable for different complexity requirements, being able to reduce the bitrate of the transcoded sequence by changing the transcoded frame structure. Further details can be found in the full version of this paper [14].

6. REFERENCES

[1] J. D. Slepian and J. K. Wolf, "Noiseless Coding of Correlated Information Sources", in *IEEE Trans. on Information Theory*, vol.19, pp. 471-480, Jul. 1973.

[2] A. D. Wyner and J. Ziv, "The Rate-Distortion Function for Source Coding with Side Information at the Decoder", in *IEEE Trans. on Information Theory*, vol.22, no.1, pp. 110, Jan. 1976.

[3] B. Girod, A. Aaron, S. Rane, and D. Rebollo-Monedero,, "Distributed Video Coding", in *Proc. IEEE Special Issue on Advances in Video Coding and Delivery*, 2005, pp. 112.

[4] A. Aaron, R. Zhang and B. Girod,, "Wyner-Ziv Coding of Motion Video", in *Proc. Asilomar Conf. on Signals and Systems*, Nov. 2002.

[5] A. Aaron, S. Rane, E. Setton and B. Girod,, "Transform-Domain Wyner-Ziv Codec for Video", in *Proc. SPIE Visual Communication and Image Processing*, vol.5308, pp. 520528, Jan. 2004.

[6] R. Puri, A. Majumdar and K. Ramchandran,, "PRISM: A Video Coding Paradigm with Motion Estimation at the Decoder", in *IEEE Trans. on Image Processing*, vol.16, no.10, pp. 24362448, Oct. 2007.

[7] X. Artigas, J. Ascenso, M. Dalai, S. Klomp, D. Kubasov and M. Ouaret,, "The Discover Codec: Architecture, Techniques and Evaluation", in *Proc. of Picture Coding Symposium*, vol.17, no.9, pp. 11031120, Nov. 2007.

[8] C. Brites, J. Ascenso and F. Pereira,, "Feedback Channel in Pixel Domain Wyner-Ziv Video Coding: Myths and Realities", in *Proc. of the 14th European Signal Processing Conf. (EUSIPCO '06)*, Sep. 2006.

[9] M. Morbee, J. Prades-Nebot, A. Pizurica and W. Philips,, "Rate Allocation Algorithm for Pixel-Domain Distributed Video Coding without Feedback Channel", in *Proc. of IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, vol.1, pp. 521524, Apr. 2007.

[10] T. Sheng, G. Hua, H. Guo, J. Zhou and C. W. Chen,, "Rate Allocation for Transform Domain Wyner Ziv Video Coding without Feedback", in *Proc. of the 16th ACM Int. Conf. on Multimedia*, pp. 701704, 2008.

[11] E. Peixoto, R. L. Queiroz and D. Mukherjee,, "Mobile Video Communications using a Wyner-Ziv Transcoder", in *Proc. SPIE Visual Communication and Image Processing*, Jan. 2008.

[12] A. Aaron, D. Varodayan and B. Girod,, "Wyner-Ziv Residual Coding of Video", in *Picture Coding Symposium*, Apr. 2006.

[13] B. Machiavello, F. Brandi, E. Peixoto, R. L. de Queiroz and D. Mukherjee,, "Side-information Generaton for Temporal and Spatial Scalable Wyner-Ziv codecs", *EURASIP Journal of Image and Video Processing, Special Issue on Distributed Video Coding*, vol. 2009, Article ID 171257, 11 pages, 2009.

[14] E. Peixoto and R. L. Queiroz and D. Mukherjee,, "A Wyner-Ziv Video Transcoder", *to appear, IEEE Trans. Circuits and Systems for Video Technology*, 2009.