QUALITY-TEMPORAL TRANSCODER DRIVEN BY THE JERKINESS

G. Iacovoni, S. Morsa

Ericsson Lab Italy / CoRiTel
via Anagnina 203, 00040 Rome, Italy
giovanni.iacovoni@ericsson.com
salvatore.morsa@ericsson.com

R. Felice

CoRiTel
via Anagnina 203, 00040 Rome, Italy
felice@coritel.it

ABSTRACT

We propose a new video homogeneous transcoding architecture DCT-based which relies on both quality and temporal reduction techniques. The frame layer control is driven by a new indicator, the *jerkiness*, which represents the user perception of the movement which affects a video stream. The proposed transcoder can meet the constraints of a real-time communication and it has been extensively tested under different conditions.

1. INTRODUCTION

In this paper[1] we deal with real-time video transcoding DCT-based, which is an emerging field of research due to the coexistence of different video standards and the growth of heterogeneous networks. Constraints requirements of real-time communications call for fast transcoding architectures. Here we first review the basic transcoding schemes proposed so far, together with their limitations. Then we outline a new approach by proposing a transcoding scheme based on a new indicator, the *jerkiness*. We tested our proposed architecture in a real-time environment. It resulted to be very promising.

There are two main homogeneous video transcoding schemes[2]: the coarse requantization transcoding (also known as quality transcoding) and the temporal transcoding.

Since most of the coding time (roughly 80%) is spent in the motion estimation process, both the above mentioned transcoding schemes try to reuse the incoming motion vectors, possibly refining them (see e.g.[3]).

In the **quality transcoding**, the encoding process is accomplished by using the incoming motion vectors and quantizing the DCT coefficients by means of a bigger quantization step size. Open-loop schemes suffer from drift errors. To overcome this problem DCT residual coefficients must be recomputed. To achieve a further compression, motion vectors can be previously refined, so that less bits are spent to code DCT residual coefficients.

Quality transcoder architecture might be simplified by collapsing DCT/IDCT operations ([10]) and might be speed-up if motion compensation is performed in the DCT domain ([2]).

The main disadvantage of the quality transcoder is that if the bit-rate reduction happens to be too high, even the roughest quantizer factor might not be able to guarantee the output target bit-rate.

Temporal transcoding, on the other hand, provides bit-rate reduction by skipping frames. Vector composition methods are needed in order to recompute the motion vectors for the output stream [1]. An update review of frame skipping methods can be found in [4]. However, in mere frame skipping architectures, only the transcoded frame rate can be modified since a macroblock layer rate control is missing (and therefore the bit budget for each frame cannot be changed). As a result, buffer constraints could not be met in real-time transmissions unless frame skipping is done at a constant frame rate, which can be ineffective from a visual point of view.

From the above discussion it is quite evident that the research trend is focusing on more flexible architectures based on a mix of quality and temporal transcoding. However, the criterion for real-time selection of the frames to drop is still an open problem. For instance the skipping decision could be driven by the motion activity: frames with relatively small motion activity are selected to be dropped ([5],[6]) [3].

But motion activity by itself does not take into account the temporal distance between two consecutive frames, as it will be shown in detail in the next section. Thus the visual impact can still suffer from a jerky effect.

In [7] an improvement is proposed. To reduce the jerky effect, frames to be skipped are selected on the basis of an

[1]This work was carried out in the context of the research activities of CoRiTeL Consortium, Via Anagnina 203, 00040 Rome, Italy

[2]It should be also mentioned the spatial resolution reduction, but we do not deal with it in this paper

[3]Moreover, these two papers propose schemes which do not work in real-time

indicator which is the derivative of the motion activity. Unfortunately this algorithm doesn't work in a real-time communication since it relies on the presence of a too big buffer and needs the motion vectors of the next frame to transcode the current frame.

On the contrary, our transcoding architecture solves the above mentioned problems by using a new indicator, the *jerkiness*, defined as the product of the motion activity by the temporal distance between two consecutive frames. This indicator is computed real-time and used in a mixed quality-temporal transcoding architecture to drive the frame layer control.

The *jerkiness*, on the basis of the motion content in the incoming video stream, predicts the outgoing frame rate and selects as a consequence the frames to drop. Real-time experimental tests confirm that our transcoder, driven by the *jerkiness*, is superior from a perceptual point of view to any mixed quality-temporal transcoder architecture working real-time proposed in the literature so far. Even if we applied this approach to the homogeneous H.263+ transcoding, the proposed architecture is still working for any DCT-based codecs.

Section 2 describes our proposed architecture starting from the definition of *jerkiness* and then going into the details of the mechanism. Section 3 shows some results obtained in a test-bed. The paper ends with our conclusions in section 4.

2. PROPOSED VIDEO TRANSCODER

The basic principle of our transcoder is to make the most of both temporal and spatial transcoding: when the motion content is "high", spatial transcoding is used keeping the original frame rate (at expense of a reduced frame quality) so not to penalize the perceived temporal smoothness. On the other hand, when the motion content is "low", temporal transcoding is performed so that skipping is active but the quality of each frame will not be reduced. In the following, first we define the *jerkiness* indicator, then we present our transcoder architecture which use the *jerkiness* indicator to dynamically adjust temporal and spatial quality.

2.1. Jerkiness indicator

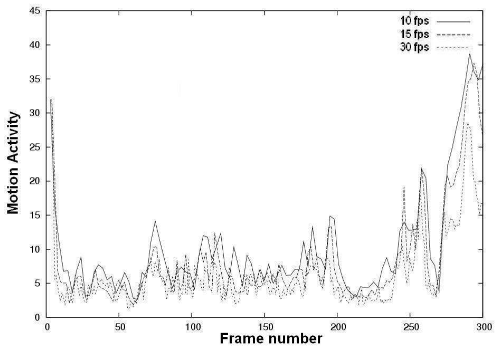
The key concept is that the temporal smoothness depends on **both the transcoding frame rate and the motion content of the scene**. As the motion content by itself, it is captured by the local Motion Activity (MA_n^l) defined as follows for the n-th frame:

$$MA_n^l = \frac{1}{N_I + N_{nz}} \left(\sum_{i=1}^{N_{nz}} |mvx_i| + |mvy_i| + N_I \cdot P_I \right)$$

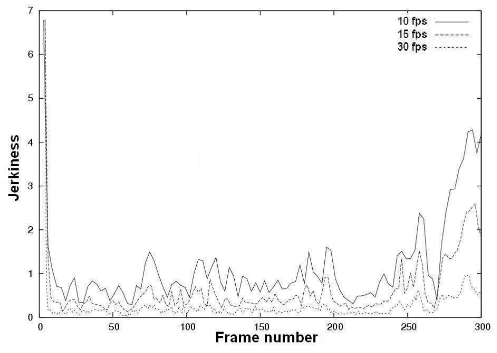
where N_I and N_{nz} are the number of INTRA Macroblocks and the number of non zero motion vectors Macroblocks in a frame respectively, P_I is a weight for INTRA Macroblocks that we set to 32 (the maximum of motion vector for a 16x16 integer motion search window) and mvx_i and mvy_i are the horizontal and vertical components of the i-th motion vector. (MA_n^l) just defined allows us to capture also local movement (e.g a talking head). Fig. 1a shows the motion activity of the well-known sequence *Foreman* encoded at three different frame rates. It is evident from the figure that the difference between them is almost unnoticeable, even if form a visual point of view the jerky effect is clearly appreciable when the frame rate is low. This is true for all the sequences where we measured the MA_n^l. This led us to define the *jerkiness* J_n as follows:

$$J_n = K \cdot MA_n^l \cdot (t_n - t_{n-1}) \quad (1)$$

where t_n and t_{n-1} are the current and previous frame transcoding time. (K is just a constant suitable chosen so to make the plots easier to read). J_n can capture the annoying effect which may arise even if MA_n^l is low. This is further shown in fig. 1b, where now the three curves are clearly distinguishable.



(a) Motion Activity



(b) Jerkiness

Fig. 1. Motion Activity and *jerkiness* of "Foreman" sequence

2.2. Transcoder Architecture

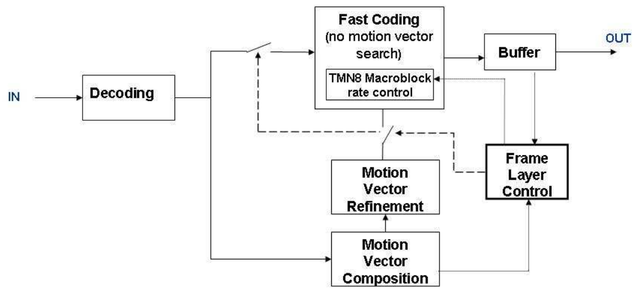

IN
Decoding
Fast Coding
(no motion vector search)
TMN8 Macroblock rate control
Buffer
OUT
Frame Layer Control
Motion Vector Refinement
Motion Vector Composition

Fig. 2. Proposed transcoder scheme

With reference to figure 2, in the proposed architecture, full decoding of the input stream provides motion vectors to be used in re-encoding. Forward Dominant Vector Search (FDVS) is used for motion vector composition because of its high performances ([1]). Motion vector refinement with a 5x5 search window is then performed and the estimated motion vectors are used for fast encoding of residual DCT coefficients. The rate control of the proposed transcoder at the Macroblock Layer follows the TMN8 algorithm ([9]). Following we explain how our Frame Layer Control (FLC) works. First we define $\Delta B = \frac{R_{Out}}{F_{In}}$ the buffer threshold, where R_{Out} is the target output rate and F_{In} is the frame rate of the input stream. The input frame rate can be easily computed using two successive RTP Timestamps. ΔB is the minimum size of an output frame to prevent buffer underflow.

The FLC drives the skipping decision according to the buffer constraint: if the buffer level is greater than ΔB, the current incoming frame is not transcoded and the FLC enables the FDVS algorithm (i.e. Motion Vector values are stored).

If the test on the buffer doesn't fail, the current frame will be transcoded. In this way we minimize the delay introduced by the rate control buffer. The FLC computes now the bit budget B_n according to a threshold Th for J_n. The value of Th reflects the degree of smoothness which is set according to the user preference.

To compute B_n, let us refer to figure 3 where t_{lt} is the time of the the last transcoded frame, t_n is the current frame time, and therefore the current frame rate is $\frac{1}{t_n - t_{lt}}$. There are two possible cases according to the value of Th:

1. $J_n < Th$: the jerky effect is less than the maximum allowed, therefore the current frame rate is higher than the frame rate needed to guarantee the given temporal quality

2. $J_n \geq Th$: otherwise

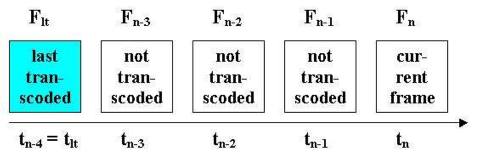


Fig. 3. Bit budget computation

In the **first case**, since the frame rate is too high, the bit budget B_n will be:

$$B_n = R_{Out} \cdot (t_n - t_{lt}) + \Delta B \qquad (2)$$

Therefore, after having skipped k frames (k=3 in fig. 3), next frame transcoding will be done after skipping $k+1$ frames, thus lowering the frame rate so that the bit gain is exactly spent to improve the quality of the n^{th} frame.

In the **second case**, the frame rate is too low to achieve the desired Th. Therefore the bit budget will be:

$$B_n = R_{Out} \cdot (t_j^* - t_{lt}) \qquad (3)$$

where t_j^* is

$$t_j^* = t_j - \frac{t_j - t_{j-1}}{J_j - J_{j-1}} \cdot (J_j - Th)$$

and t_j is such that $j = \min_{lc+1,\dots,n}(k : J_k \geq Th)$.

In other words, we look into the paste in order to identify the frame time j where the threshold was crossed for the first time but there was no enough buffer space to transcode frame j. So now frame n can be transcoded in such a way that the target J_n can be met. If second condition is always verified, our transcoder behaves as if it were a TMN8-based transcoder with the target frame rate set equal to the input one[4].

3. EXPERIMENTAL RESULTS

The proposed Transcoder has been implemented in C language and its performances measured in a test-bed. The test-bed is composed of two IP terminals equipped with VIC (VIdeo Conferencing tool, see [8]); the video codec is the ITU H.263+. The RTP video packets are transmitted from *terminal 1* to the Transcoder at rate R_{In}. The transcoder adapts the input bitrate in real-time and sends RTP packets to *terminal 2* at rate R_{Out} (smaller than R_{In}). Actually, video capture can be done in real-time from a Webcam, but we used some well-known QCIF video sequences (300 frames, 10 seconds long) to test the Transcoder. We report the results for the ”Carphone” and ”Akiyo” sequences,

[4]There is a minor difference in the buffer dimension, which however doesn't worth further comments

transcoded from 256 kbit/sec to 32 kbit/sec. In figure 4 the PSNR of Carphone for temporal, TMN8, and the proposed transcoder with $Th = 3$ and $Th = 11$ are shown. The temporal transcoder is a plain one, i.e. it uses the same quantization factors of the input frame. The PSNR of the proposed transcoder lies between the PSNR of a quality transcoder TMN8-based (where a big deal of the frames are transcoded) and that of a plain temporal transcoder (where the number of transcoded frames is directly dependent on the ratio of the two bit rates). An higher value of Th corresponds to an higher tolerance to the jerky effect. Therefore as the Th is increased the behaviour of the proposed transcoder converges towards that of the temporal transcoder.

As an additional comment, from frame 180 to frame 220, the behaviour of the proposed architecture follows closer that of the TMN8. What happens here is that J_{180} exceeds both the thresholds, because of the high motion content [5]. Therefore the transcoder suddenly behaves as predicted by eq. (3), at least as soon as the buffer content allows it. The output frame rate is then approximately the same as the input frame rate, at least till frame 220, beyond which the motion content slows down. For the same sequence and simulation settings, Table 1 shows the mean PSNR and the number of transcoded frames. Our architecture, for the two given values of Th, turns out to have an average PSNR which is only slightly better than a plain quality transcoder TMN8-based. Nevertheless we checked that from a visual point of view its performance are better since our transcoder always aims at meeting the *jerkiness* constraint.

4. CONCLUSIONS

We have proposed a new transcoding architecture which takes profit from both quality and temporal transcoding by using a suitable Frame Layer Control driven by a new indicator, the *jerkiness*. This indicator has proven to be very useful since it possesses the right degree of flexibility to satisfy any given target level of smoothness, which can be actually preset by the user. The transcoder has been extensively tested using a wide range of well-known sequences and live experimentations. Besides it can work regardless of the values of all the parameters. Furthermore the transcoder doesn't assume any constance in the environment where it works: for instance the incoming frame rate or the bit rate can vary over the time, as well the threshold of *jerkiness* and all the other parameters used in the system.

Further studies may investigate if and how it would be possible to control automatically this threshold on the basis of some features real-time extracted from the incoming video stream.

[5]In Carphone video the camera is placed in a car, filming a close-up talking head; at frame 180 the camera zooms out, and there aree trees entering the background yielding the effect of movement

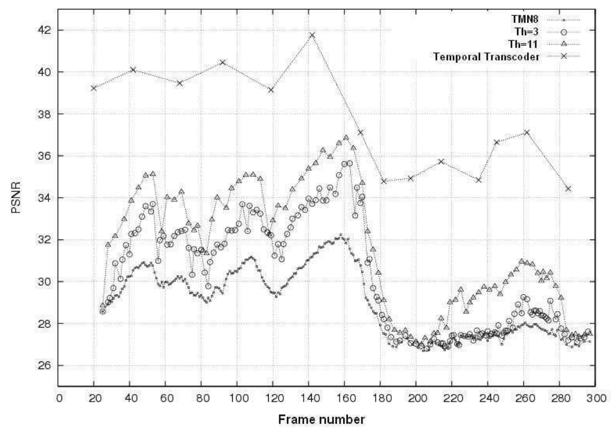


Fig. 4. PSNR of the Carphone sequence transcoded from 256 kbit/s to 32 kbit/s

Transcoder architecture	Frame transcoded	PSNR
TMN8	232	29.30
Th=3	152	30.11
Th=11	86	31.28
Temporal	14	37.79

Table 1. Mean PSNR of Carphone sequence transcoded from 256 kbit/sec to 32 kbit/sec

5. REFERENCES

[1] M.-T Sun, J. Youn, "Motion Vector Refinement for High-Performance Transcoding", *IEEE Trans. On Multimedia*, 1(1), March 1999.

[2] S.F. Chang, D.G. Messerschmidt,"Manipulation and compositing of MC-DCT compressed video", *IEEE J. Select. Areas Commun.*, 13 (1), Jan. 1995

[3] A. Vetro, C. Cristopoulos, Huifang Sun, "Video Transcoding Architectures and Techniques: an Overview", *IEEE Signal Processing Magazine*, 20 (2), March 2003

[4] Shan Liu, C. -C. J. Kuo, "Joint Temporal-Spatial Rate Control For Adaptive Video Transcoding", *IEEE Proc. of ICME 2003*, pages 225-8, July 2003

[5] Jenq-Nenq Hwang, Tzong-Der Wu, Chia-Wen Lin, "Dynamic Frame-Skipping in Video Transcoding", *IEEE Workshop on Multimedia Signal Processing* , pages 616-621, Dec. 1998

[6] Kai-Tat Fung, Yui-Lam Chan, Wan-Chi Siu, "New Architecture for Dynamic Frame-Skipping Transcoder", *IEEE Trans. On Image Processing*, 11 (8), Aug. 2002

[7] Haiyan Shu, Lap-Pui Chau, "Frame Skipping Transcoding with Motion Change Consideration", *IEEE Proc. of ISCAS 2004*, pages 773-776, May 2004

[8] http://www-nrg.ee.lbl.gov/vic/

[9] J. Ribas-Corbera, S. Lei, "Rate Control in DCT Video Coding for Low-Delay Communications ", *IEEE Trans. On Circuits and Systems for Video Technology*, 9 (2), Feb. 1999

[10] J. Youn, Ming-Ting Sun, Yun Xin, "Video Transcoder architectures for Bit Rate Scaling of H.263 Bit Streams", *ACM Multimedia*, Oct. 1999