Efficient Mobile-to-Mobile Video Communications Using a Multicore Transcoder

Alberto Corrales-Garcia, José Luis Martínez, and Gerardo Fernández-Escribano

Instituto de Investigación en Informática de Albacete (I3A)
Universidad de Castilla-La Mancha
02071 Albacete, Spain
{albertocorrales,joseluismm,gerardo}@dsi.uclm.es

Abstract. Video transcoding from Distributed Video Coding (DVC) to H.264/AVC provides a suitable framework for Mobile-to-Mobile video communications because both the transmitter and the receptor keep low complexity constraints during video compression and decompression processes. DVC is formed by simpler encoders jointly with a high complex decoder, a scheme counterposed to H.264/AVC. This framework thus moves the high complex processes to the transcoder, which has more resources. For this reason, this work is focused on reducing the complexity of the transcoder. With this aim, on the one hand the first transcoding stage is improved through a multicore processor, which executes the DVC decoding in a parallel way. On the other hand, the second stage uses MVs generated during the DVC decoding to reduce the motion estimation complexity of H.264/AVC encoding. As a result, an efficient paradigm is provided to support mobile-to-mobile video communications which reduces the time spent by up to 77% without significant rate-distortion loss.

Keywords: Mobile-to-Mobile Video Communications, Transcoding, DVC, H.264/AVC, Parallel Computing.

1 Introduction

Nowadays, mobile devices demand multimedia services such as video communications due to the advances in mobile communications systems (such us 4G) and the integration of video cameras in mobile devices. However, these devices have some limitations of computing power, resources and complexity constraints for performing complex algorithms. For this reason, in order to establish a video communications between mobile devices it is necessary to use low complex encoding techniques. In traditional video codecs (such as H.264/AVC [1]) these low complexity requirements have not been met because H.264/AVC is more complex at the encoder side. Then, mobile video communications based on H.264/AVC imply a high loss of Rate – Distortion (RD) to reduce the complexity of the encoder. However, DVC [2] provides a novel video paradigm where the complexity of the encoder is reduced by shifting the complexity of the encoder to the decoder [6]. Taking into account the

J. Rodriguez, R. Tafazolli, C. Verikoukis (Eds.): MOBIMEDIA 2010, LNICST 77, pp. 74–87, 2012.

benefits of both paradigms, recently DVC to H.26X transcoders have been proposed in the multimedia community to support mobile-to-mobile video communications. As is shown in Figure 1, this framework provides a scheme where transmitter and receiver execute lower complexity algorithms and the majority of the computation is moved to the network. This complexity is assumed by a transcoder, which has more resources and no battery limitations. Nevertheless, for real time communications it is necessary to perform this conversion from DVC to H.264/AVC with a short delay, and then the transcoding process must be executed as efficiently as possible.

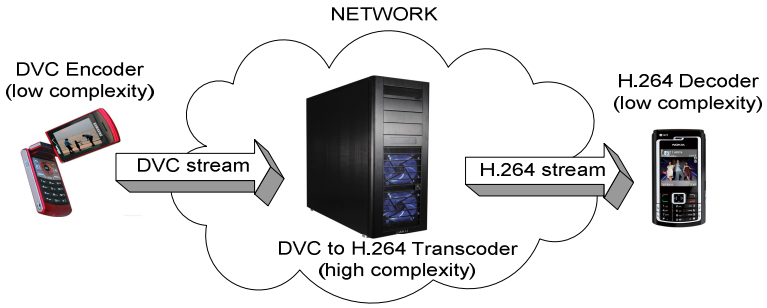

Fig. 1. Scheme of DVC to H.264/AVC transcoding framework to support Mobile-to-Mobile video communications

At this point, this work presents two proposals in order to create a more efficient transcoding process. On the one hand, due to today's parallel programming it is becoming more important to solve high complexity computation tasks and the computing market is full of multicore systems, an approach is proposed to execute DVC decoding in a parallel way. On the other hand, at the same time DVC is decoding, some MVs generated during Side Information (SI) generation process are sent to the H.264/AVC encoder in order to reduce the search area of the Motion Estimation (ME) process. In this way, the complexity of the two most complex tasks of this framework (DVC decoding and H.264/AVC encoding) are largely reduced making the transcoding process more efficient.

2 Technical Background

2.1 Wyner-Ziv Video Coding

Wyner-Ziv (WZ) [2] video coding is a particular case of DVC. WZ video codingwhich departs from the Wyner-Ziv theorem [3] on source coding with SI available only at the decoder, which is known to be a technique used to reduce the processing complexity of the encoder, leading to low-cost implementation, while the majority of the computations are taken over by the decoder. Most of the research done in WZ video coding is carried out based on the architecture first proposed by Stanford

[4]; this architecture is mainly characterized by turbo-code based Slepian-Wolf (SW) coding and a feedback channel to perform rate control at the decoder. Departing from this architecture, many improvements have been introduced in the literature [6][7] but, recently, a WZ video coding architecture has been established by the VISNET-II project [5]. The VISNET-II WZ video codec is the most advanced codec available in the literature which is based on the architecture depicted in Figure 2.

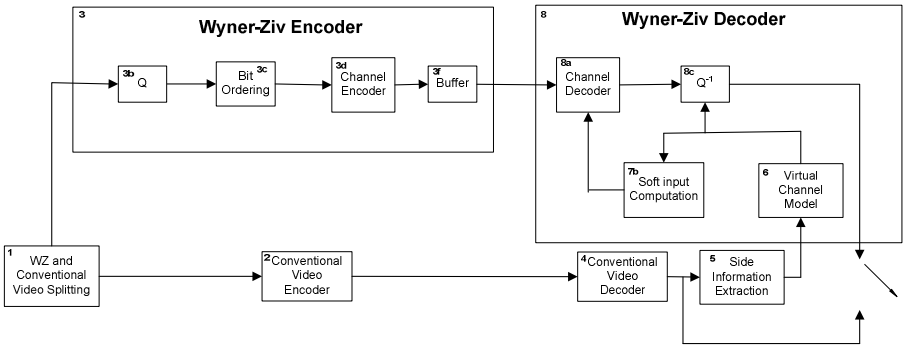

Fig. 2. Block diagram of the DVC reference architecture

In a nutshell, the video sequence is divided into Key (K) frames and WZ frames in the splitting module (1). At the encoder, the K frames are coded using Intra H.264/AVC video codec [1] (2). On the other hand, pixels of WZ frames are firstly quantized (3b). Over the resulting quantized symbol stream, bitplane (BP) extraction is performed per BPs (3c). Each BP is then independently channel encoded, starting with the most significant BP (3d). The parity bits produced by the channel encoder are stored in the buffer and transmitted in small amounts upon decoder request via the feedback channel; the systematic bits are discarded (3f).

At the decoder side, firstly the K frames are decoded using Intra H.264/AVC video codec [1] (4) and theses frames are used in the SI generation process (5). The frame interpolation module is used to generate the SI frame, an estimate of the WZ frame, based on previously decoded frames [6]. The SI can be seen as a corrupted version of the original information, the difference between the original WZ frame and its corresponding SI is considered as correlation noise in a virtual channel. A Laplacian model is used to obtain an approximation of the residual distribution in a Virtual Channel Model (VCM) (6) [7]. The SI is used by an iterative decoding algorithm to obtain the decoded quantized symbol. The parity bits and statistical modeling of the virtual noise are the input of the channel decoders (8a). The success of the channel decoding algorithm is determined by the module 8b, if the decoding algorithm does not converge, more parity bits are requested using the feedback channel. This iterative procedure is repeated until successful decoding is obtained (normally without errors) and another BP starts being decoded. After that, the reconstructed pixels are obtained using the decoded quantized pixels, the virtual channel model estimated in (6) and the quantized SI pixels.

2.2 H.264/AVC

H.264/AVC [1] or MPEG-4 part 10 Advanced Video Coding (AVC) [8] is a compression video standard developed by the ITU-T Video Coding Experts Group (ITU-T VCEG) together with the ISO/IEC Moving Picture Experts Group (MPEG). In fact, both standards are technically identical.

The main purpose of H.264/AVC is to offer a good quality standard able to considerably reduce the output bit rate of the encoded sequences, compared with previous standards, while exhibiting a substantially increasing definition of quality and image. H.264/AVC promises a significant advance compared with the commercial standards currently most in use. (MPEG-2 and MPEG-4) [9]. For this reason H.264/AVC contains a large amount of compression techniques and innovations compared to previous standards; it allows more compressed video sequences to be obtained and provides greater flexibility for implementing the encoder. Figure 3 shows the block diagram of the H.264/AVC encoder.

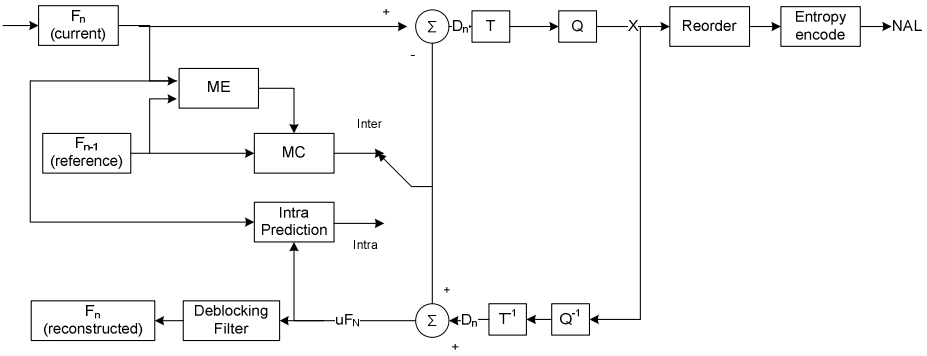

Fig. 3. H.264/AVC encoder diagram

The ME is the most time-consuming task in the H.264/AVC encoder. It is a process which removes the temporal redundancy between images, comparing the current one with previous or later images in terms of time (reference images), looking for a pattern that indicates how the movement is produced inside the sequence.

To improve the encoding efficiency, H.264/AVC allows the use of partitions resulting from dividing the MB in different ways. Greater flexibility for the ME and Motion Compensated (MC) processes and greater motion vector precision give greater reliability to the H.264/AVC encoding process. The ME process is thus carried out many times per each partition and sub-partition. This feature is known as variable block size for the ME.

2.3 Transcoding for Mobile to Mobile Communications

One of the most referenced overviews of DVC [4], mentioned as one of the benefits of this new video coding paradigm, is the support of mobile-to-mobile communications

using a transcoder device. The first WZ-based video transcoder was presented by Peixoto et al in [10] 2008, and is based on WZ/H.263 to support this kind of communications. The authors propose a mapping between the WZ Group of Pictures (GOP) and the traditional GOP and, moreover, for the P or B slices in H.263 some ME refinement is also proposed. However, they did not exploit the correlation between the WZ MVs and the traditional ME successfully and only used them to determine the starting centre of the ME process. Our previous work [11] presents the first WZ to H.264/AVC based transcoder which is another step forward in the framework of WZ-based video transcoders. This work improves the H.264/AVC inter prediction in two ways: i) by using the Motion Vectors (MV) calculated in the SI to reduce the H.264/AVC searching area and, ii) by using a decision tree to reduce the MB codec partition algorithm. The approach presented in [11] focuses only on the H.264/AVC encoding part of the whole transcoder but, the WZ decoding part also consumes even more time than an H.264/AVC one [6]. The WZ time decoding also has an impact on the overall delay of the transcoder. Nevertheless, the previous approach only employs a WZ GOP size of 2 to be transcoded to a H.264/AVC GOP size of 2. In the present approach this GOP size has been adapted and generalized for longer GOP as I11P which is the most suitable pattern for mobile-to-mobile video communications [17]. And, finally, the WZ decoding part used in the present approach marks an improvement [11] by means of implementing lossy key frame coding, on-line correlation noisy modeling and not using the ideal procedure call at decoder for the stopping criterion, which makes the proposed transcoder more practical.

3 Proposed Architecture

In this section, a brief introduction will be given in section 3.1 in order to provide the motivation for this work. Subsequently, both proposals are described in sections 3.2 and 3.3.

3.1 Introduction

It is well known that traditional codecs (such as H.264/AVC) provide more complex encoders than decoders while DVC inverts the complexity providing decoders which are less complex than encoders. As a consequence, the Mobile-to-Mobile video communications framework based on DVC to H.264/AVC transcoding unites the most complex processes of two paradigms in the same system: the transcoder. Therefore, the complexity of both parts must be reduced.

On one hand, during the transcoding process some information provided by DVC decoding could be used to save time on the second stage. Particularly, in the SI generation process some MVs could be obtained. As is shown in figure 4, the first step of SI generation consists of matching each MarcoBlock (MB) of forward frame (k+n) with a MB of the backward frame (k). This matching is done by searching inside the search area for the MB which produces the lowest residual. Through this

process a MV is obtained for each MB which quantifies the displacement between both MBs, and the middle of this MV represents the displacement for the MB interpolated. The complete SI estimation procedure is detailed in [12]. In this way, MVs generated during the DVC decoding will be reused to reduce the complexity of the H.264/AVC encoding as will be explained in section 3.3.

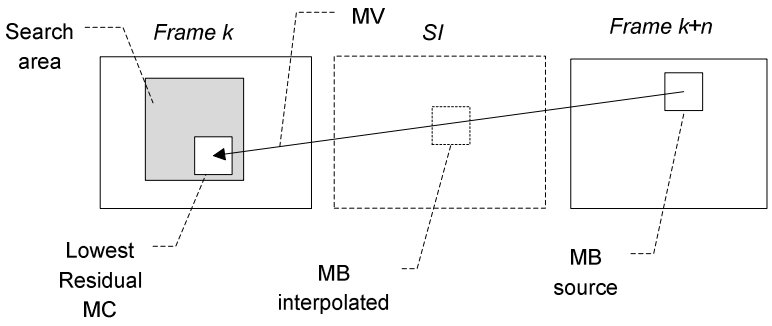

Fig. 4. First step of SI generation process

On the other hand, the first stage must be improved because if only the second stage of the DVC to H.264/AVC transcoding process is improved, the first stage is still a bottleneck for the complete transcoding process. In the particular case of WZ decoding, a high complexity is concentrated in the turbo decoder module. Table 1 shows the analysis of WZ decoding complexity in PD for the Foreman QCIF at 15fps sequence organized by most relevant modules. As may be observed, most of the complexity between relevant modules (Turbo Decoder, VCM, SI generation, Reconstruction and Other) corresponds to the Turbo Decoder module. In addition, the time increment, as the time spent by each BP compared with the time taken by the decoding of 1 BP, increases when more BPs are decoded. As is displayed, this time grows exponentially when more bitplanes are decoded. For this reason, when more BPs are decoded the necessity of improving the decoding process increases.

Table 1. Analysis of WZ decoding complexity organized by modules for different BPs

BP	Turbo Decoder (%)	VCM (%)	SI generation (%)	Reconstruction (%)	Other (%)	Time Increment
1	97.63	1.27	1.08	0.01	0.01	1.00
2	98.65	0.76	0.56	0.01	0.01	1.94
3	99.26	0.43	0.29	0.01	0.01	3.69
4	99.50	0.29	0.19	0.01	0.01	5.68
5	99.67	0.19	0.12	0.01	0.00	8.89

However, parallel programming is becoming more and more important and parallel systems are a growing market. In fact, most commercial computers include a Multicore Processor. This kind of processor is composed by independent cores sharing the same processor. This allows programs to be executed in a parallel way increasing the performance of a high complexity task, like video coding. Nevertheless, most of programs follow a sequential execution order due to the sequential dependencies. Then, new ways for breaking these dependencies should be investigated. In the case of DVC decoding, the VCM is updated whenever a BP decoding finishes. In section 3.2, a proposal is explained to break the sequential dependence and then execute the DVC decoding by a multicore processor in a parallel way.

3.2 Parallelization of DVC Decoding

As was observed in Table 1, the Turbo Decoder is the most complex module and this complexity increases when the number of BPs decoded increases. For this reason, in this section a parallel DVC decoding execution at BP level is proposed. To carry out this task a multicore Intel i7-940 was selected, which is based on Intel Nehalem Micro-architecture [13]. The most important features of this processor are the following: four cores, clock speed of 2.93 GHz, 45nm manufacturing process, new point-to-point processor interconnect Intel QuickPath Interconnect (QIP), Simultaneous Multi-Threading (SMT) by multiple cores which enables the simultaneous execution of two threads per core (hyper-threading), and three levels of cache (32 KB L1 instruction and 32 KB L1 data cache per core, 256 KB L2 cache per core and 8 MB L3 cache shared by all cores). Figure 5 shows an architecture scheme for this multicore processor.

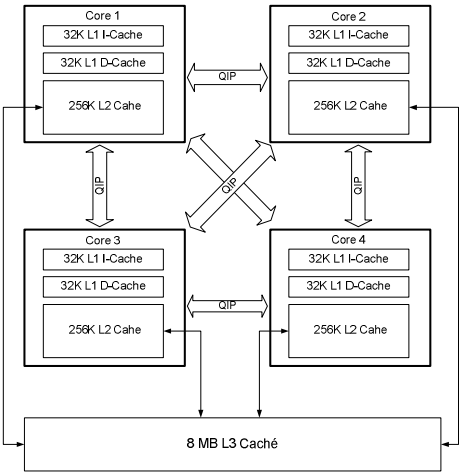

Fig. 5. Architecture scheme of the multicore processor Intel i7-940

Regarding DVC decoding algorithm, in order to use the SI in terms of coding efficiency, the turbo decoder needs to have reliable information on the statistics between the SI frame and the original frame. This is modeled by the VCM as a corrupted version of the original information with an error pattern. The VCM has two limits of integration for each bit of each BP and they are updated whenever a BP is decoded. For example, for the BP 0 a pixel could obtain a value from 0 to 255. However, if a bit of the decoded BP 0 is equal to 1; the pixel will obtain a value from 128 to 255, so integration limits must be updated with these values. In this way, each BP offers more accuracy for the final pixel value. This means that the time line is sequential and every BP is dependent on the previous one. Figure 6 shows the sequential time line for the decoding of a frame with 4 BPs. On the other hand, taking into account SI as an approach of the original frame, we propose making an initial update of the integration limits based on the SI information available before the decoding starts. Accordingly, each BP could be decoded in a parallel way. However, the use of approximated integrations limits implies an increase in the number of bit parity chunks necessary to correct each BP. For this reason, as Figure 7 shows, we propose a dynamic update of the integration limits when each BP decoding finishes. In addition, the ending order is not always sequential. For example, in Figure 8 BP1 will finish at X1 and B 4 at X4. If this happens, BP 1 and BP 4 will finish the decoding without updating the integration limits and this could imply PSNR losses. Therefore, an improvement is added in order to wait for the previous update before finishing. For example, BP 1 waits from X1 to X2 and then, with the model updated, finishes.

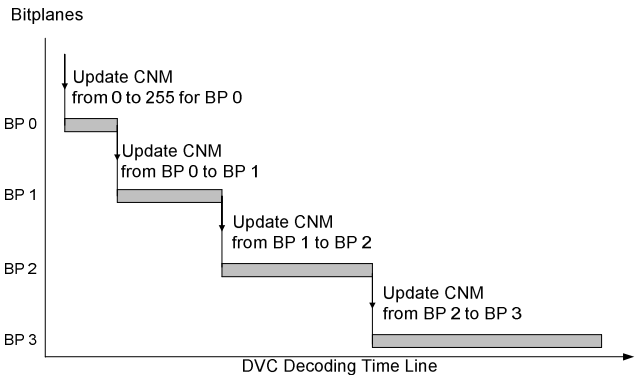

Fig. 6. Timeline for sequential decoding 4 BPs in PD updating VCM

3.3 H.264/AVC Transcoding Approach

As was explained in section 3.1, the SI generation process provides some MVs which could give us an approximation of the quantity of movement. In order to reduce the time spent by H.264/AVC encoding, we propose using these MVs to reduce the ME

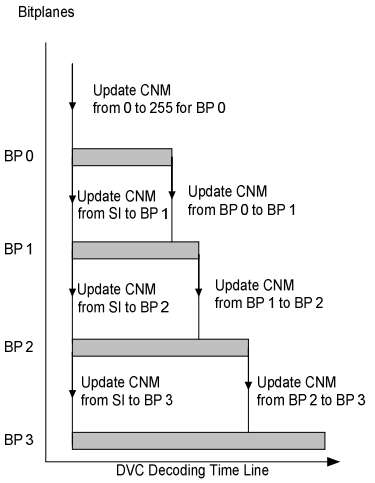

Fig. 7. Timeline for parallel decoding of 4 BPs in PD updating the VCM from the SI at the beginning and re-updating the VCM when the previous BPs finish

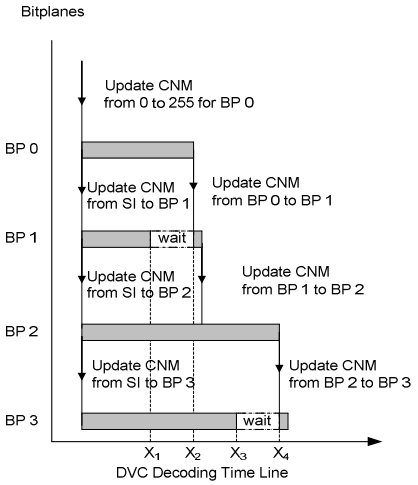

Fig. 8. Timeline for parallel decoding of 4 BPs in PD updating the VCM from the SI at the beginning and re-updating the VCM when the previous BP finishes and waiting until the previous BPs finish

search area in H.264/AVC. Firstly, the mapping issue should be solved. In DVC decoding, the SI is estimated between two frames. Although DVC encodes larger GOPs, in the last step the SI is generated between two frames with distance 2, so this is the most accurate SI and only these MVs will be taken into account to reduce the H.264/AVC complexity. For example, Figure 9 shows the decoding for a GOP 4. In step 1 MVs generated are ignored while the MVs generated in step 2 are sent to H.264/AVC. In the pattern I11P inter frames have references with distance 1, so MVs are divided into two halves.

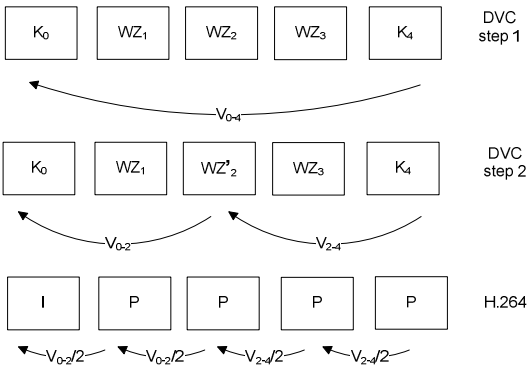

Fig. 9. Mapping from DVC GOP 4 to H.264/AVC GOP I11P

In the H.264/AVC process, a big part of this complexity depends largely on the search range used in the H.264/AVC ME process to carry out the checking. However, this range could be reduced, thus avoiding unnecessary checking without significant impact on quality and bit rate. To achieve this aim, we propose to reuse the MVs generated in DVC to define a smaller search range for each MB of H.264/AVC including every sub MB partition. In this way, the checking area is limited by the area S defined in the expression (1).

$$S = \{(x,y)/(x,y) \in (A \cap C)\} \tag{1}$$

$$A = d * d \tag{2}$$

Where (x,y) are the coordinates to check. A is the search range used by H.264/AVC or the square of the search range *d,* equation (2). Finally, C is a circumference which restricts the search with centre on the upper left corner of the MB. C is defined by the equation (3).

$$C^2 = r_x^2 + r_y^2 \tag{3}$$

$$r_x = \max\left(\frac{MV_x}{2}, \frac{d}{4}\right) \tag{4}$$

$$r_y = \max\left(\frac{MV_y}{2}, \frac{d}{4}\right) \tag{5}$$

Where r_x and r_y are calculated from equations (3) and (4) depending of the MVs halves ($MV_x/2$ and $MV_y/2$) provided by DVC or a minimum defined by a quart of the search range (labeled *d*). This minimum area is considered to avoid applying too small search ranges.

4 Experimental Results

In order to evaluate the performance of the proposed transcoder for Mobile-to-Mobile video communications we have taken into account two halves and also global results are presented. Firstly, on the DVC stage, 150 frames of three QCIF sequences with different features (Foreman, Hall, CoastGuard) were encoded using a codec based on PD VISNET-II codec with GOP of 2 and 4 at 15fps. A QP of 4 was used because it provides a suitable quality without too high a bitrate increment [6]. During the DVC decoding values of PSNR, bitrate (in kbps) and decoding time for sequential and parallel version were measured. As table 2 shows, the PSNR obtained is the same for both versions as a consequence of sequential and parallel versions iterating until a stopping condition is reached [18]. However, the parallel version presents a non-significant bitrate increment because some BPs need a few iterations more to complete the decoding. On the other hand, the time reduction (TR) obtained is defined as $TR = 100 * \left(Time_{reference} - Time_{porposal}\right)/Time_{reference}$. With proposed parallel DVC decoding up to 81.77% of TR is reached. Comparing GOPs 2 and 4 similar conclusions are observed.

Table 2. Performance of the proposed DVC parallel execution for 15fps QCIF sequences

GOP 2					
Sequence	**Sequential**		**Parallel**		**TR (%)**
	PSNR	**Bitrate**	**PSNR**	**Bitrate**	
Foreman	33.53	420.38	33.53	427.27	81.77
Hall	35.71	287.29	35.71	294.15	77.77
CoastGuard	33.14	412.75	33.14	413.99	80.27
GOP 4					
Sequence	**Sequential**		**Parallel**		**TR (%)**
	PSNR	**Bitrate**	**PSNR**	**Bitrate**	
Foreman	33.56	622.5	33.56	643.15	81.73
Hall	36.38	313.28	36.38	325.93	77.63
CoastGuard	33.17	529.34	33.17	532.62	80.36

In the second stage, sequences are encoded by a H.264/AVC codec, which makes a I11P pattern at 15 fps. The simulations were run by using the version JM 17.0 of H.264/AVC [14] and the baseline profile with the configuration file by default and search area of length 32. The baseline profile was selected because it is the profile most used in real-time applications due to its low complexity. For the same reason, RDOptimization was turned off. To encode QPs = 28, 32, 36, 40 were used, as specified in *Bjøntegaard and Sullivan´s common test rule* [15][16]. Table 3 shows the results for the H.264/AVC encoding of WZ GOP 2 and GOP 4 sequences. As may be observed, the proposed H.264/AVC encoder saves up to 74.14% of TR without negligible RD losses.

Table 3. Performance of the proposed H.246 improved execution for 15fps QCIF sequences

GOP 2			
Sequence	**ΔBitrate (%)**	**ΔPSNR**	**TR (%)**
Foreman	-2.35	-0.10	72.02
Hall	0.02	0.00	66.54
CoastGuard	-0.42	-0.02	74.42
GOP 4			
Sequence	**ΔBitrate (%)**	**ΔPSNR**	**TR (%)**
Foreman	-3.26	-0.13	72.20
Hall	-0.06	0.00	66.19
CoastGuard	-0.38	-0.02	74.14

Finally, table 4 collects the results for the complete transcoding process. In this case the ΔPSNR was measured comparing sequences before transcoding starts and after transcoding finishes. On the other hand ΔBitrate was calculated by adding the bitrate generated by the DVC decoder and bitrate generated by the H.264/AVC

decoder and then, applying the average. In a similar way, TR was calculated as the average of both TR (DVC decoding and H.264/AVC encoding). As may be observed, there are no negligible RD losses and the time spent by the transcoding process is reduced by up to 77.25%.

Table 4. Performance of the proposed DVC to H.246 transcoder for 15fps QCIF sequences

GOP 2			
Sequence	**ΔBitrate (%)**	**ΔPSNR**	**TR (%)**
Foreman	-1.86	-0.06	76.89
Hall	-2.13	-0.15	72.15
CoastGuard	-0.33	-0.01	77.34
GOP 4			
Sequence	**ΔBitrate (%)**	**ΔPSNR**	**TR (%)**
Foreman	-3.37	-0.36	76.97
Hall	-3.66	-0.12	71.91
CoastGuard	-0.60	-0.05	77.25

Figures 10 and 11 show the four RD performance points (QP = 28, 32, 36 and 40) for the complete transcoding process. As may be observed, the quality level is maintained but sometimes there is a little displacement of the bitrate due to the increment of the DVC decoding stage. For GOP 2 and 4 conclusions are similar.

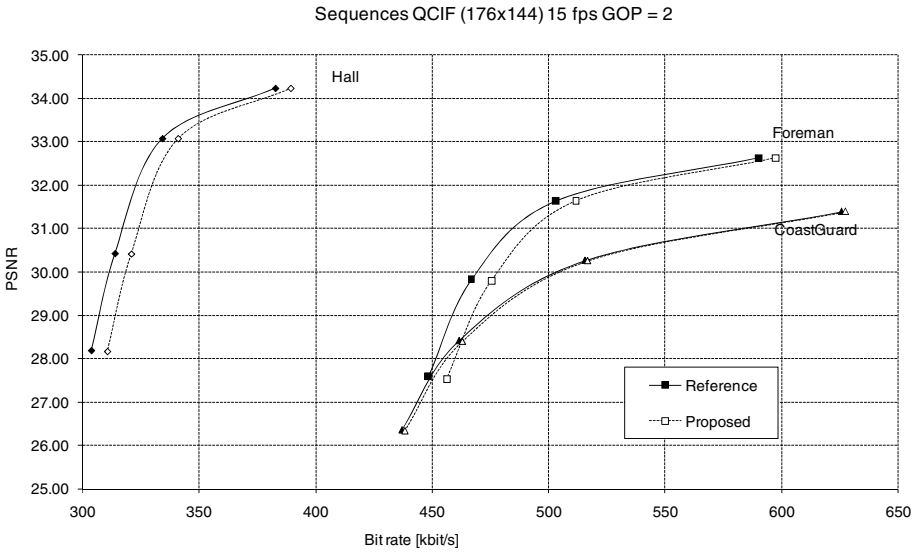

Fig. 10. Transcoding PSNR/bitrate results using QP=28, 32, 36 and 40 for 15fps sequences with GOP = 2. Reference symbols: ■Foreman, ♦Hall and ▲CoastGuard.

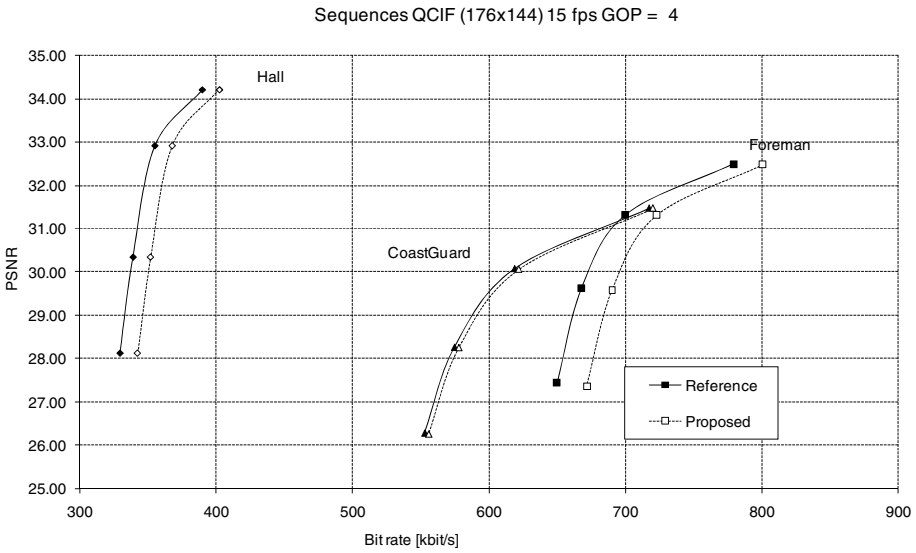

Fig. 11. Transcoding PSNR/bitrate results using QP=28, 32, 36 and 40 for 15fps sequences with GOP = 4. Reference symbols: ■Foreman, ♦Hall and ▲CoastGuard.

5 Conclusions

This work presents a DVC to H.264/AVC transcoder designed to support Mobile-to-Mobile video communications. Since the transcoder device accumulates the highest complexity of both video coders, reducing the time spent in this process is a major task. With this aim, in this work two approaches are proposed to improve the DVC decoding and the H.264/AVC encoding. With both approaches a time reduction of up to 77.34% is achieved for the complete transcoding process without negligible RD losses. In a nutshell, this work offers a straightforward step in the framework of efficient DVC to H.264/AVC video transcoders.

Acknowledgment. This work was supported by the Spanish MEC and MICINN, as well as European Commission FEDER funds, under Grants CSD2006-00046, TIN2009-14475-C04. It was also partly supported by JCCM funds under grant PEII09-0037-2328 and PII2I09-0045-9916, and the University of Castilla-La Mancha under Project AT20101802.The work presented was developed by using the VISNET2-WZ-IST software developed in the framework of the VISNET II project.

References

1. ITU-T and ISO/IEC JTC 1: Advanced Video Coding for Generic Audiovisual Services. In: ITU-T Rec. H.264/AVC and ISO/IEC 14496-10 Version 8 (2007)
2. Aaron, A., Zhang, R., Girod, B.: Wyner-Ziv Coding of Motion Video. In: Proceeding of Asilomar Conference on Signals and Systems, Pacific Grove, CA (November 2002)

3. Wyner, A.D., Ziv, J.: The Rate-Distortion Function for Source Coding with Side Information at the Decoder. IEEE Transactions on Information Theory IT-22, 1–10 (1976)
4. Girod, B., Aaron, A., Rane, S., Monedero, D.R.: Distributed Video Coding. Proceeding of IEEE Special Issue on Advances in Video Coding and Delivery 93(1), 1–12 (2005)
5. VISNET II project, http://www.visnet-noe.org/ (last visited May 2010)
6. Brites, C., Ascenso, J., Pedro, J.Q., Pereira, F.: Evaluating a feedback channel based transform domain Wyner-Ziv video codec. Signal Processing: Image Communication 23(4), 269–297 (2008)
7. Brites, C., Pereira, F.: Correlation noise Modeling for efficient pixel and transform domain Wyner-Ziv video coding. IEEE Transactions on Circuits and Systems for Video Technology 18(9), 1177–1190 (2008)
8. ISO/IEC 14496-10:2005, – Information technology – Coding of audio-visual objects – Part 10: Advanced Video Coding
9. ISO/IEC 14486-2 PDAM1:1999, Infomation Technology-Generic Coding of Audio-Visual Objects- Part 2: Visual
10. Peixoto, E., Queiroz, R.L., Mukherjee, D.: A Wyner-Ziv Video Transcoder. IEEE Transactions on Circuits and Systems for Video Technology 20(2), 189–200 (2010)
11. Martínez, J.L., Kalva, H., Fernández-Escribano, G., Fernando, W.A.C., Cuenca, P.: Wyner-Ziv to H.264 video transcoder. In: 16th IEEE International Conference on Image Processing (ICIP), Cairo, Egypt, pp. 2941–2944 (2009)
12. Ascenso, J., Brites, C., Pereira, F.: Improving frame interpolation with spatial motion smoothing for pixel domain distributed video coding. In: 5th EURASIP Conference on Speech and Image Processing, Multimedia Communications and Services, Smolenice, Slovak Republic (2005)
13. Intel Processor Core family, http://www.intel.com/ (last visited May 2010)
14. Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, Reference Software to Committee Draft. JVT-F100 JM15.1 (2009)
15. Sullivan, G., Bjøntegaard, G.: Recommended Simulation Common Conditions for H.26L Coding Efficiency Experiments on Low-Resolution Progressive-Scan Source Material. ITU-T VCEG, Doc. VCEG-N81 (2001)
16. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: From error visibility to structural similarity. IEEE Transactions on Image Processing 13(4), 600–612 (2004)
17. Wiegand, T., Sullivan, G.J., Bjøntegaard, G., Luthra, A.: Overview of the H.264/AVC video coding standard. IEEE Trans. Circuits Syst. Video Technol. 13, 560–576 (2003)
18. Tagliasacchi, M., Pedro, J., Pereira, F., Tubaro, S.: An efficient request stopping method at the turbo decoder in distributed video coding. In: EURASIP European Signal Processing Conf. (September 2007)