A GRAPH-BASED APPROACH FOR LATENCY MODELING AND OPTIMIZATION IN MULTIVIEW VIDEO ENCODING

Pablo Carballeira,[†] *Julián Cabrera,*[†] *Antonio Ortega,*[§] *Fernando Jaureguizar,*[†] *and Narciso García*[†]

[†] Grupo de Tratamiento de Imágenes, ETSI de Telecomunicación,
Universidad Politécnica de Madrid, Ciudad Universitaria, 28040, Madrid, Spain.
[§] Signal and Image Processing Institute, Department of Electrical Engineering,
University of Southern California, Los Angeles, CA 90089-2564, USA.

ABSTRACT

We present a novel framework for encoding latency analysis of arbitrary multiview video coding prediction structures. This framework avoids the need to consider an specific encoder architecture for encoding latency analysis by assuming an unlimited processing capacity on the multiview encoder. Under this assumption, only the influence of the prediction structure and the processing times have to be considered, and the encoding latency is solved systematically by means of a graph model. The results obtained with this model are valid for a multiview encoder with sufficient processing capacity and serve as a lower bound otherwise. Furthermore, with the objective of low latency encoder design with low penalty on rate-distortion performance, the graph model allows us to identify the prediction relationships that add higher encoding latency to the encoder. Experimental results for JMVM prediction structures illustrate how low latency prediction structures with a low rate-distortion penalty can be derived in a systematic manner using the new model.

Index Terms— 3D Video, video-conference, Multiview Video Coding, prediction structure, low latency, graph theory.

1. INTRODUCTION

3D Video (3DV) and Free Viewpoint Video (FVV) are new types of visual media that expand the user's experience beyond what is offered by 2D video [1]. A common element of 3DV and FVV systems is the transmission of multiple views of the same scene to the user. Multiview Video Coding (MVC) provides efficient compression of multiple video inputs that capture different views of the same content. In MVC, a very flexible design of temporal and interview prediction dependencies is introduced, which leads to considerably different coding performances on the selected coding structure [2].

Application-driven requirements also play an important role in the design of MVC systems. In our previous work [3], we addressed a multiview video-conferencing scenario with strict constraints on end-to-end delay. It was argued that using solely rate-distortion (RD) performance on prediction structure design ignores important differences on latency behavior, which may be critical for such systems. Therefore, we proposed a framework for a systematic encoding latency analysis of arbitrary multiview prediction structures. This framework firstly captures the temporal relationships among frames due to the MVC GOP prediction structure which are independent of the specific hardware architecture of the multi-processor encoder. However, the rest of the latency model assumes specific architectural features of the encoder, such as number of processors, single/multi task encoding within each processor, policies to control the frame-to-processor assignment, etc. Therefore, the proposed framework has to be customized for different encoder implementations [4]. This motivates us to extend the latency analysis framework in order to make it as independent as possible of specific features of the encoder architecture.

In this paper, we present a general framework for encoding latency analysis that assumes the use of a multiprocessor encoder with an unbounded processing capacity. We demonstrate that under this assumption, most of the restrictions imposed by the encoder architecture are avoided, so the framework customization according to the specific encoder architecture is not necessary. To the best of our knowledge, this is the first time that such a general and systematic encoding latency analysis framework is presented, and it may be used to simplify multiview prediction structure design for low-delay applications. Given a specific multiview encoder architecture (number of cameras, processors, etc), latency analysis can be performed using a specific encoder architecture model [3][4]. Instead, this general model can be used to find a prediction structure that meets a target encoding latency, under the assumption that the number of processors is essentially unbounded. For that structure we can then easily identify the minimum number of processors required to provide the target latency. In problems where with targets in terms of both latency and number of processors, we can iteratively simplify the coding structure until both targets are met. The results obtained within this model are accurate for real multiview encoder implementations where the number of processors is above the required minimum, and provide a lower bound for encoding latency otherwise.

The model is based on a directed acyclic graph (DAG) extracted from the multiview prediction structure. We will refer to it as the *Directed Acyclic Graph Encoding Latency* (DAGEL) model. It can be seen as a task scheduling model [5] in which the objective is not the minimization of the schedule length, but the computation of the encoding latency. This problem is solved by finding the critical path on the DAG. We show how the DAGEL model properties can be used for low latency encoder design with low penalty on RD performance: low latency prediction structures can be derived from an initial prediction structure by cutting the dependency links that introduce a higher encoding delay in the original structure, and those can be systematically found by using the DAGEL model. Results show evidence of completely different latency values for structures with comparable RD performance, e.g. for two structures with close RD performance the latency difference can be more than 40%.

This paper is organized as follows: in Section 2 we discuss the conditions of the DAGEL model and we present it. In Section 3 we show how the DAGEL model can be used to reduce the encod-

978-1-61284-162-5/11/$26.00 ©

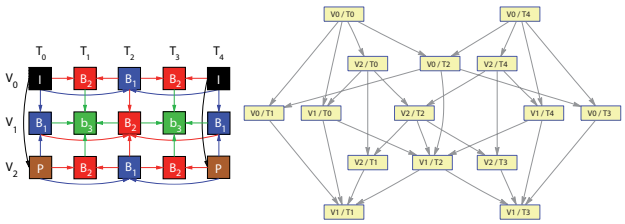


Fig. 1. Example of a JMVM prediction structure with three views and a GOP size of four frames and the DAG extracted from it. V_i/T_j represents frame j of view i as signaled on GOP structure

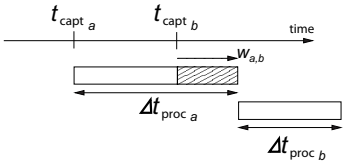


Fig. 2. Graphic significance of the edge cost $w_{a,b}$ in the graph. Delay introduced by one frame.

ing latency of arbitrary prediction structures. In Section 4 we present latency and RD results for latency reduction applied to JMVM structures. In Section 5 we present the conclusions.

2. LATENCY ANALYSIS USING THE DAGEL MODEL

As defined in [3], the encoding latency for a multiview sequence is:

$$Lat = \max(t_{\text{cod}}{}_j^i - t_{\text{capt}}{}_j^i)\ i = 0 \dots N-1,\ j = 0 \dots M-1, \quad (1)$$

where N is the number of views, M the number of frames per view, $t_{\text{cod}}{}_j^i$ is the instant when x_j^i (frame j of view i) is completely coded and $t_{\text{capt}}{}_j^i$ is the capture time of x_j^i.

For any frame in the multiview sequence, $t_{\text{cod}}{}_j^i$ can be computed as:

$$t_{\text{cod}}{}_j^i = t_{\text{start}}{}_j^i + \Delta t\text{proc}_j^i, \quad (2)$$

where $t_{\text{start}}{}_j^i$ is the instant when encoding of x_j^i starts and $\Delta t\text{proc}_j^i$ is the corresponding processing time for this frame.

Besides, $t_{\text{ready}}{}_j^i$ is defined as the instant when x_j^i is ready to be coded, and is computed as follows:

$$t_{\text{ready}}{}_j^i = \max(t_{\text{capt}}{}_j^i, \max_{l \in L(j,i)}(t_{\text{cod}_l})), \quad (3)$$

where $L(j,i)$ is the set of reference frames for x_j^i.

2.1. Encoder architecture with unlimited processing capacity

Without loss of generality, if we assume an encoder architecture model in which each processor can only encode one single frame at a time, we will have that:

$$t_{\text{start}}{}_j^i \geq t_{\text{ready}}{}_j^i, \quad (4)$$

that is we cannot start coding x_j^i until all frames used to predict it have been encoded.

If no idle processor is available at $t_{\text{ready}}{}_j^i$, the start of the encoding process of frame x_j^i is delayed, despite all of its reference frames have been already coded. For different encoder architecture models, the difference between $t_{\text{ready}}{}_j^i$ and $t_{\text{start}}{}_j^i$ is variable. However, the relationship in Equation (4) can be simplified under certain conditions on the multiview encoder. Consider a scenario with an encoder architecture of K independent processors that can communicate to exchange interview references, and that they encode their assigned frames sequentially. It can be proven that for a finite number of views and a finite frame rate, a value K_{min} exists so that it if $K \geq K_{min}$ there will always be at least one idle processor at any time in which a frame of the sequence is ready for encoding. Formally:

$$t_{\text{start}}{}_j^i = t_{\text{ready}}{}_j^i, \text{ and therefore:} \quad (5)$$

$$t_{\text{cod}}{}_j^i = \max(t_{\text{capt}}{}_j^i, \max_{l \in L(j,i)}(t_{\text{cod}_l})) + \Delta t\text{proc}_j^i. \quad (6)$$

If this second condition holds, the GOP latency is the same for all the GOPs of the sequence, as the encoding process of the previous GOPs does not add any delay to the current GOP. This value is also equal to the encoding latency of the whole sequence, and therefore Equations (1)-(3) only have to be computed for the first GOP.

2.2. Definition of the DAGEL model

Under the conditions of Equation (5) we can define the DAGEL model, which allows us to systematically solve Equations (1)-(3) for any multiview prediction structure. The frames on the prediction structure can be seen as nodes of a directed graph and the prediction dependencies as the edges. Each directed edge links a reference frame to the frame that is predicted from it. A path is a sequence of nodes linked by directed edges. Figure 1 shows an example of a prediction structure and the directed graph derived from it.

A DAG is a directed graph with no directed cycles, i.e. such that there is no path starting at some node A that eventually loops back to A again.

Result 1 *Any dependency graph extracted from a feasible MVC prediction structure is necessarily a DAG.*

Sketch of proof *Assume there is a directed cycle in the dependency graph starting and ending at frame (node) x_a. The last edge of this path links frame x_b with frame x_a, i.e., frame x_a is predicted from frame x_b. On the other hand, frame x_a is linked in a series of steps to frame x_b, i.e., frame x_b is non-directly predicted from frame x_a. If frame x_a is predicted from frame x_b and viceversa, none of them can be encoded, so that the prediction structure is not feasible. Therefore, a feasible structure must be a DAG.*

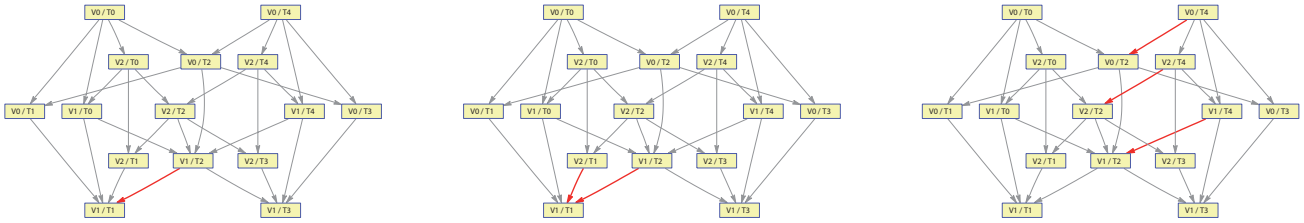
Each edge of the DAG has a cost that indicates the delay added by a parent node to the encoding process of its child node. The cost $w_{a,b}$ of the edge that links node x_a with node x_b is:

$$w_{a,b} = \max(0, (t_{\text{capt}_a} + \Delta t\text{proc}_a) - t_{\text{capt}_b}). \quad (7)$$

where t_{capt_a} and t_{capt_b} are the capture times of frames x_a and x_b respectively, and $\Delta t\text{proc}_a$ is the processing time of frame x_a. Figure 2 shows graphically the computation of $w_{a,b}$. It shows a time chronogram in which the encoding process of parent frame x_a delays the encoding start of child frame x_b. As only positive delay values have a realistic meaning, $w_{a,b}$ is restricted to positive values.

The cost of a path is the sum of the costs of the edges that link the nodes in the path. Among the set of paths ending on the same node, we call *delay path* the one having the highest total cost. For any frame of the multiview sequence:

$$t_{\text{start}}{}_j^i = t_{\text{capt}}{}_j^i + p_{del}{}_j^i \quad (8)$$



(a) 1 cut (b) 2 cuts (c) 3 cuts

Fig. 3. Latency reduction by edge pruning in the DAG. The edges in red are the ones selected to cut.

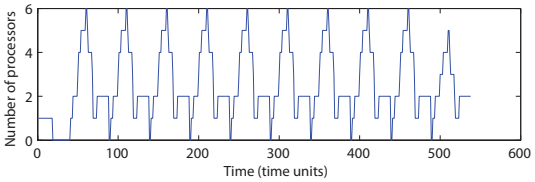


Fig. 4. Chronogram of the number of processors needed for the encoding process of the prediction structure in Figure 1.

where ${p_{del}}_j^i$ is the cost of the delay path of frame x_j^i. From Equations (1), (2) and (8) it can be derived that the encoding latency value is:

$$Lat = \max({p_{del}}_j^i + \Delta t\text{proc}_j^i), i = 1 \dots N. j = 1 \dots M, \quad (9)$$

where the delay path that maximizes Equation (9) is the *critical path*.

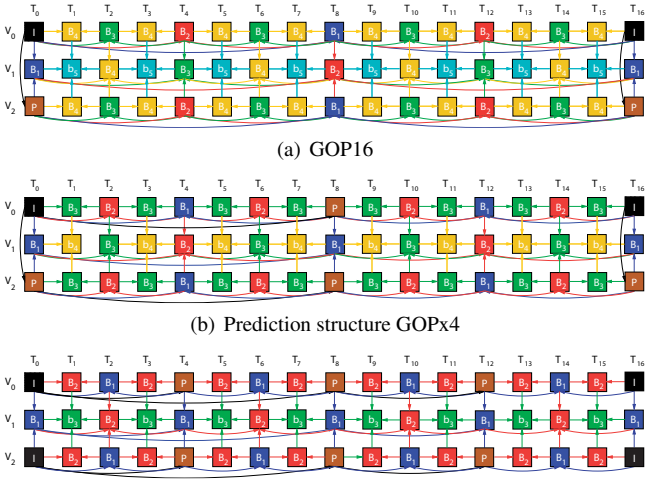
The latency value obtained using the DAGEL model considers the effects of both the multiview prediction structure and the individual processing time of each frame if a sufficient number of processors is available, and it is a lower bound of the encoding latency otherwise. A multiview encoder with a lower processing capacity available will result in a latency greater than or equal to the one obtained with the DAGEL model. Intuitively, it can be seen that by delaying the start of the encoding process of certain frames, due to certain periods of time in which all the processors are busy, the encoding latency can be incremented.

2.3. Minimum number of processors on the DAGEL model

The minimum number of processors K_{min} that ensures that the condition of Equation (5) holds, can be computed by analyzing the number of frames that are processed concurrently. Figure 4 shows an example of the chronogram of processor usage for the prediction structure in Figure 1. The chronogram shows the results for the encoding of multiples GOPs. The maximum value of this chronogram equals to K_{min}, a value that depends on the frame processing times.

3. ENCODING LATENCY REDUCTION USING THE DAGEL MODEL

In MVC encoders for immersive videoconferencing applications, low encoding delay is a strict requirement. Assuming that the number of dependency relationships in a prediction structure is directly proportional to its RD performance (the more references a frame has the more efficiently it can be encoded) it is desirable to reduce the encoding latency by pruning the prediction structure links, with the minimum number of cuts. The DAGEL model can be used to identify, in a systematic manner, the dependency links that introduce a higher encoding delay, and therefore the best ones to cut in order to reduce the encoding latency.



(a) GOP16

(b) Prediction structure GOPx4

(c) Prediction structure GOPx10

Fig. 5. Latency reduction by edge pruning in the DAG.

Result 2 *Given a DAG extracted from a prediction structure, the best cut in terms of encoding latency reduction is necessarily on its critical path.*

Sketch of proof *Consider a given DAG and its critical path, which is formed by the set of edges l_e. Assume that one cut in the edges is performed, and that it does not belong to the set l_e. It is trivial to note that the path l_e is still present in the resulting graph, so the encoding latency of the new prediction structure is equal to the parent structure. Therefore, the best cut in terms of latency performance must be on the critical path l_e.*

In the case of encoding latency reduction by multiple edge pruning, a greedy solution that iteratively cuts an edge in the critical paths may normally be sub-optimum. Instead, to obtain the optimum solution for a given number of cuts, an exhaustive search of all the possible cut combinations is needed. Figure 3 depicts an example of this phenomenon by showing optimum edge selection for an increasing number of cuts in the initial JMVM prediction structure depicted in Figure 1. The DAGs in Figure 3 correspond to the results for one, two, and three cuts respectively. The results show that the selected edge in the case of one cut is maintained for the case of two cuts, while in the case three cuts three different edges of the DAG are selected.

4. EXPERIMENTAL RESULTS

To demonstrate the capabilities of our DAGEL model, we show how it can be used to reduce the latency of a given prediction structure to a target encoding latency value, with a low penalty on RD performance. This is done by performing the minimum number of cuts

Table 1. Time parameter values.

Time parameter	Δt_{basic}	Δt_{ref}	Capture Period
Value (ms)	20	10	40 (25 fps)

Table 2. Encoding latency values.

	GOP16	GOP8	GOP4	GOPx4	GOPx10
Encoding latency (ms)	930	550	330	550	330

in the prediction relationships to achieve the target latency. In this set of experiments the time parameter values of the frame processing time model [3] have been estimated using the X264 platform in a general purpose PC, working at 2.40 GHz, with 3.25 GB of RAM memory in a QuadCore processor. The time parameter values are shown in Table 1.

Starting from an initial prediction structure with three views and a GOP size of 16 frames (GOP16, Figure 5 (a)), we have iteratively pruned its associated DAG to reduce its encoding latency to the level of analogous JMVM structures with GOP sizes of 8 (GOP8) and 4 frames (GOP4). This has been done by an increasing number of cuts in the DAG, using an exhaustive search of the possible cut combinations. The encoding latencies of these prediction structures are shown in Table 2.

The structures obtained for four (GOPx4) and ten (GOPx10) cuts are shown in Figures 5(b) and 5(c) respectively. As shown in Table 2 these structures have the same encoding latency value that structures GOP8 and GOP4 respectively. We have evaluated the RD performance of the different prediction structures using the JMVM software version 2.1 [6] and the MVC common conditions on sequences Ballroom and Race1 [7]. The results (Figure 6) show that for the tested sequences our structures outperform the JMVM prediction structures with the same encoding latency value. Table 3 shows the average RD differences [8] of GOPx4 and GOPx10 structures compared to GOP8 and GOP 4 respectively.

Table 3. RD-Bjontegaard results.

	GOPx4/GOP8		GOPx10/GOP4	
	Δ PSNR(dB)	Δ bitrate(%)	Δ PSNR(dB)	Δ bitrate(%)
Ballroom	0.20	-5.33	0.71	-17.11
Race1	-0.04	1.05	0.29	-6.77

5. CONCLUSIONS

We have presented an encoding latency analysis model based on a DAG which allows a systematic computation of the encoding latency of arbitrary prediction structures by finding the critical path on the graph. This model assumes an ideal encoder multi-processor model with a sufficient processing capacity, making it independent of the multi-processor architecture. The obtained latency value is valid for a system with the assumed processing capacity and a lower bound otherwise. This encoding latency model may be used for multiview prediction structure design in a system with strict end-to-end latency requirements. We show how the DAGEL model can be used to reduce the encoding latency of a given prediction structure to a target value by the minimum number of cuts in the dependency relationships. To demonstrate this we have applied this method to JMVM prediction structures. Results show that the prediction structures obtained with our method outperform JMVM structures with same encoding latency value in terms of RD performance.

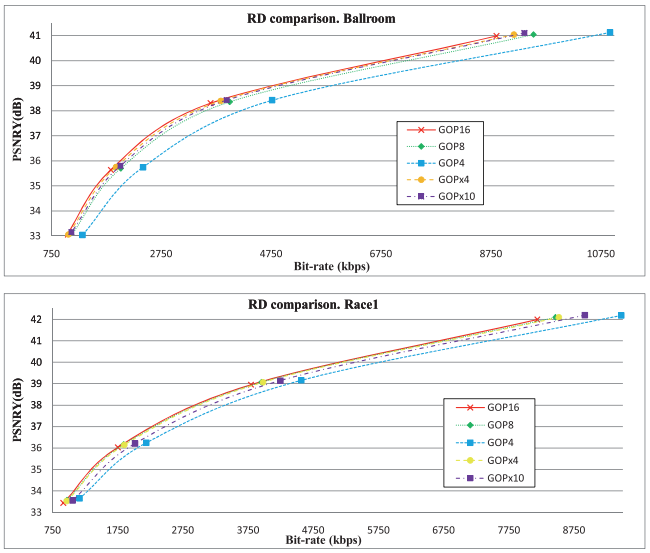


Fig. 6. Rate-Distortion performance. Ballroom and Race1.

6. ACKNOWLEDGEMENTS

This work has been partially supported by SAPEC and the Spanish Administration agency CDTI under project CENIT-VISION 2007-1007, and by the Ministerio de Ciencia e Innovación of the Spanish Government under projects TEC2007-67764 (SmartVision) and TEC2010-20412 (Enhanced 3DTV). Also, P. Carballeira wishes to thank the Comunidad de Madrid for a personal research grant.

7. REFERENCES

[1] A. Smolic, K. Mueller, P. Merkle, C. Fehn, P. Kauff, P. Eisert, and T. Wiegand, "3D Video and Free Viewpoint Video - Technologies, Applications and MPEG Standards," *Proc. IEEE ICME 2006*, pp. 2161–2164, July 2006.

[2] P. Merkle, A. Smolic, K. Mueller, and T. Wiegand, "Efficient Prediction Structures for Multiview Video Coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, no. 11, pp. 1461–1473, Nov. 2007.

[3] P. Carballeira, J. Cabrera, A. Ortega, F. Jaureguizar, and N. García, "Comparative latency analysis for arbitrary multiview video coding prediction structures," *IS&T/SPIE VCIP 2009*, vol. 7257, pp. 72570L–1–12, Jan. 2009.

[4] P. Carballeira, J. Cabrera, A. Ortega, F. Jaureguizar, and N. García, "Latency analysis for a multi-processor multiview video encoder implementation," in *APSIPA ASC 2009*, October 2009, pp. 367–372.

[5] Y. Kwok and I. Ahmad, "Static scheduling algorithms for allocating directed task graphs to multiprocessors," *ACM Comput. Surv.*, vol. 31, pp. 406–471, December 1999.

[6] P. Pandit and A. Vetro, "JMVM 2 software," *Doc. JVT-U208 Hangzhou, China*, October 2006.

[7] Y. Su, A. Vetro, and A. Smolic, "Common Test Conditions for Multiview Video Coding," *Doc. JVT-T207 Klagenfurt, Austria*, July 2006.

[8] G. Bjontegaard, "Calculation of average PSNR differences between RD curves," *ITU-T SG16/Q6, 13th VCEG Meeting, Doc. VCEG-M33*, April 2001.