Complexity-Constrained Video Bitstream Shaping

Yiannis Andreopoulos, *Member, IEEE*, and Mihaela van der Schaar

***Abstract*—In scalable or layered video coding, bitstreams that fulfill specific rate or distortion profiles can be derived (shaped) postencoding, i.e., at transmission time. This can also be a very effective method for real-time adjustment of the decoding complexity to bounds specified by the decoding processor or operating system. In this paper, a new type of bitstream shaping is discussed, which is able to fulfill various instantaneous complexity profiles/constraints imposed by the decoder. Complexity is estimated by modeling generic metrics that can be translated into receiver-specific execution-time or energy-consumption estimates. For various bitstream-shaping alternatives, we associate their corresponding complexity estimates with the decoder modules via a decomposition into complexity coefficients and complexity functions. This process drives bitstream shaping according to joint distortion and complexity constraints, while the proposed model incurs limited online computational and storage overhead. In addition, our experiments demonstrate that the proposed method that is based on offline training can outperform various online training methods commonly used for resource prediction in the literature. For each group of pictures, our bitstream-shaping experiments indicate that the accuracy of the proposed method permits adaptation to real-time imposed constraints on the complexity metrics within a 10% error margin.**

***Index Terms*—Bitstream adaptation, complexity modeling, video streaming.**

I. Introduction

Low-power reconfigurable processors with multitasking capabilities are already used in wireless and portable devices. The user experience during video communications through these devices is affected by the capability of the decoding platform to adjust the video characteristics (e.g., quality, frame rate) not only based on capacity of the transmission medium, but also on the decoding platform's dynamic complexity profile. This motivated both theoretical research [1] for the definition of a complexity-distortion framework, as well as more applied frameworks where basic signal processing algorithms [2], [3] or video coding schemes [4]–[6] are analyzed based on the frequency of occurrence of the highly complex operations under certain assumptions for the realization platform.

In this paper, we are considering selective shaping and transmission of bitstreams such that certain complexity constraints are met at the decoder. In our framework, the bitstream-shaping process is defined as the postencoding adjustment of quality levels, temporal decomposition levels (for scalable coders [7], [8]), or parts of the motion information (using motion-scalability functionalities [7]) under certain rate, distortion, and complexity constraints. Any postencoding modification of the bitstream implicitly alters the decoding realization as well, as the decoder modules are invoked a different number of times and they process compressed bitstreams that require different computational and memory resources. Given real-time requirements of a particular decoder, the appropriate shaping can be performed such that the decoder is expected to seamlessly process the received bitstream.

Manuscript received May 7, 2005; revised June 16, 2006. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Shuvra S. Bhattacharyya. The authors would like to acknowledge the kind support of the NSF: CCF 0541867 (Career Award), CCF 0541453, and CNS 0509522.

Y. Andreopoulos was with Department of Electrical Engineering, University of California, Los Angeles (UCLA), Los Angeles, CA 90095-1594 USA. He is now with the Department of Electronic Engineering, Queen Mary University of London, London, U.K. E1 4NS (e-mail: yiannis.a@elec.qmul.ac.uk).

M. van der Schaar is with the Department of Electrical Engineering, University of California, Los Angeles (UCLA), Los Angeles, CA 90095-1594 USA (e-mail: mihaela@ee.ucla.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSP.2006.890867

A. Basic Principles and Link to Previous Work

This paper presents an extension of our previously proposed complexity-modeling framework [9] that is general and can fit to numerous video decoding schemes. The derived models estimate the expected (rather than worst-case) complexity. For each bitstream configuration, this framework predicts the complexity associated with the various modules of the video decoder based on complexity variables, which represent the rate of change of nonzero transform coefficients or nonzero motion parameters across each group of pictures (GOP). The measured complexity variables for each video frame depend on the bit rate, the content characteristics, as well as on the decoding algorithm (entropy coding method used, motion model in the codec, etc.). Given a certain set of the complexity variables for the current GOP, we calculate complexity decomposition functions based on the error frames and the motion vectors. In this way, these functions capture the content characteristics. Finally, for each instantiation of the complexity variables and decomposition functions, we calculate complexity decomposition coefficients based on an offline training process. These consist of a mapping of each frame's complexity decomposition functions to the measured complexity metrics. As a result, they depend on the decoding algorithm and its implementation. Notice that for the same algorithm specification, a particular implementation will influence the complexity metrics (and hence the decomposition coefficients) but not the decomposition functions, since the resulting representation will be identical. For example, the discrete wavelet transform can be implemented via lifting or convolution in literature; however, for every instantiation of the complexity variables, the resulting transform representation of the error frame (and, hence, the corresponding decomposition functions of our framework) will be identical in both cases. Nevertheless, the decomposition coefficients related to the inverse discrete wavelet transform operations at the decoder will differ, due to the implementation differences between lifting and convolution.

B. Contributions and Outline of This Paper

In this paper, we build on our prior work [9] and establish bitstream truncation points using the corresponding metadata (complexity variables and decomposition functions). We propose a new framework for bitstream shaping based on complexity constraints, which, unlike previous work [9], adapts several decoding parameters in real time in order to comply with rapidly changing complexity bounds. We select module-specific complexity metrics that can be easily mapped into real complexity such as execution time or energy consumption, as shown in our recent experiments [10]. Moreover, unlike our initial work [9], the proposed training process is performed without any modifications or parameter enforcement to the decoder. To substantiate our complexity-estimation method experimentally, we present an extensive set of results as well as a comparison with relevant methods from the literature for dynamic resource prediction.

Section II explains the chosen complexity metrics and analyzes our modeling approach. Section III presents experimental validations of the modeling accuracy. Bitstream shaping under decoder complexity and distortion constraints is presented in Section IV. Finally, Section V presents our conclusions.

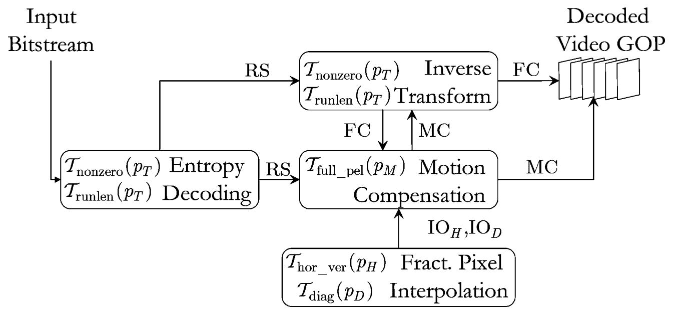

Input Bitstream
Decoded Video GOP
RS
FC
FC
MC
RS
MC
$\mathcal{T}_{\text{nonzero}}(p_T)$ Inverse
$\mathcal{T}_{\text{runlen}}(p_T)$ Transform
$\mathcal{T}_{\text{nonzero}}(p_T)$ Entropy
$\mathcal{T}_{\text{runlen}}(p_T)$ Decoding
$\mathcal{T}_{\text{full\_pel}}(p_M)$ Motion Compensation
IO_H,IO_D
$\mathcal{T}_{\text{hor\_ver}}(p_H)$ Fract. Pixel
$\mathcal{T}_{\text{diag}}(p_D)$ Interpolation

Fig. 1. Basic modules of the proposed generic video decoding complexity modeling framework. The complexity functions (with the corresponding complexity variables) are indicated in each module; the generic complexity metrics of each module are indicated at the connecting arrows.

II. Complexity Modeling Based on Generic Metrics

The basic operational framework for the majority of transform-based motion-compensated video decoders can be summarized by the modules illustrated in Fig. 1. The decoder modules, depicted as nodes connected by edges, communicate via decoded data that are determined by the received shaped bitstream. The functions within each module of Fig. 1 consist of the complexity decomposition functions associated to this module. The schematic of an entire video transmission framework incorporating bitstream-shaping is illustrated in Fig. 2. As shown in the figure, bitstream shaping occurs postencoding and it may be performed in various places in the network, i.e., at the encoder side, at an intermediate location (server or proxy), or at the decoder side, prior to the actual decoding. In the following subsection we define the metrics quantifying the generic complexity cost for the data flow between modules. In addition, Section II-B presents the complexity variables with the corresponding complexity functions and formalizes the proposed estimation framework.

A. Complexity Metrics

For a particular realization of a video decoding algorithm (i.e., using a programmable processor, or an application-specific media processor, or custom hardware), the decoding complexity associated with the information exchange between the modules of Fig. 1 may be expressed in execution cycles, processor functional-unit utilization, energy consumption, etc. Based on experiments that analyzed several such costs [10], we assume generic operations for each module that are easily mapped to real decoding complexity and are indicated by the connecting arrows.

- *Entropy decoding*: The number of iterations of the "read symbol" (RS) function is considered. This function encapsulates the (context-based) entropy decoding of a symbol from the compressed bitstream.
- *Inverse transform*: The number of multiply-accumulate (MAC) operations with nonzero entries is considered, by counting the number of times a MAC operation occurs between a filter coefficient (FC) and a nonzero decoded pixel or transform coefficient.
- *Motion compensation*: The basic motion compensation (MC) operation per pixel or transform coefficient is the dominant factor in this module's complexity.
- *Fractional-pixel interpolation*: The MAC operations corresponding to horizontal or vertical interpolation (IO_H), and also the MAC operations corresponding to diagonal interpolation (IO_D), can characterize the complexity of this module. Notice that diagonal interpolation is more complex since it includes both horizontal and vertical interpolation.

B. Complexity Variables, Complexity Decomposition Functions, and the Proposed Estimation Framework

As proposed in our recent work [9], we are modeling complexity based on compressed-source statistics that are easy to determine at encoding time. For this purpose, we use the concept of a video adaptation unit (AU), which can be a video frame, a group of macroblocks, a transform subband, etc. [9]. A compressed AU consists of the smallest bitstream component taken into account in the complexity model. Typically, a GOP consists of N AUs, with N being a parameter of the video decoding algorithm. It is important to notice that, in the proposed framework, complexity modeling and bitstream shaping are always performed with a GOP granularity. This complexity-modeling granularity is chosen such that the model equations express an inner product (i.e., decomposition) of the GOP complexity into complexity coefficients and complexity functions that obtain different values for each AU depending on the individual bitstream component characteristics. Both our current experimental findings (Sections III and IV), as well as our previous work [9], indicate that the averaging performed during the calculation of the model-based GOP complexity decreases the estimation error in comparison to the AU-level complexity estimation. Moreover, the derived estimates are more relevant for practical bitstream-shaping algorithms that operate on a GOP level.

We define the following complexity variables: the percentage of decoded nonzero transform coefficients p_T; the percentage of decoded nonzero motion vectors p_M out of the maximum possible motion vectors per video frame; and, finally, the percentage of nonzero horizontally or vertically interpolated fractional-pixel positions p_H and diagonally interpolated fractional-pixel positions p_D. These can be perceived as the rate of utilization of each module per decoded pixel.

1) Definition of Complexity Decomposition Functions: For each AU i, with $1 \leq i \leq N$ within each GOP, we define complexity decomposition functions with respect to these variables. Starting from the entropy decoding, we define $\mathcal{T}_{\text{nonzero}}(p_T(i))$ [9], [11] as the sum of magnitudes of the nonzero coefficients

$$\mathcal{T}_{\text{nonzero}}\left(p_T(i)\right) = \sum_{k=1, c_k(p_T(i)) \neq 0}^{X_l \cdot Y_l} \left(\lfloor \log_2 \left| w_k\left(p_T(i)\right)\right| \rfloor + 2\right). \quad (1)$$

where each AU reconstructs $X_l \cdot Y_l$ pixels (l indicates the resolution level, where $l = 0$ is the full-resolution) and w_k represents the kth nonzero coefficient, after reordering all the transform coefficients in a one-dimensional array.[1] In embedded or layered-coding schemes, the value of each coefficient is a function of the percentage of decoded nonzero coefficients $p_T(i)$ [9]. We additionally define $\mathcal{T}_{\text{runlen}}(p_T(i))$ [11] as the sum of the run lengths of zero coefficients, after the conversion of the decoded coefficients of each AU in a one-dimensional array. First, we count the run lengths across the coefficients of each AU i as follows. For every $k \in [1, X_l \cdot Y_l]$ with $w_k(p_T(i)) = 0$ and $w_{k-1}(p_T(i)) \neq 0$, we have

$$Q_k\left(p_T(i)\right) = \begin{cases} j, & \text{if } \exists j \geq 1 : \left\{w_{k+m}\left(p_T(i)\right) = 0\right\}_{1 \leq m \leq j} \\ 0, & \text{otherwise} \end{cases} \quad (2)$$

otherwise $Q_k(p_T(i)) = 0$. Then, we define the corresponding complexity decomposition function as

$$\mathcal{T}_{\text{runlen}}\left(p_T(i)\right) = \sum_{k=1}^{X_l \cdot Y_l} Q_k\left(p_T(i)\right). \quad (3)$$

The complexity decomposition function related to motion-compensation is defined as follows [9]:

$$\mathcal{T}_{\text{full\_pel}}\left(p_M(i)\right) = v \cdot p_M(i) \cdot X_l \cdot Y_l \quad (4)$$

[1]Following similar definitions [13], the log-magnitude of each coefficient in (1) is incremented by 2, and not by 1, in order to include the sign information.

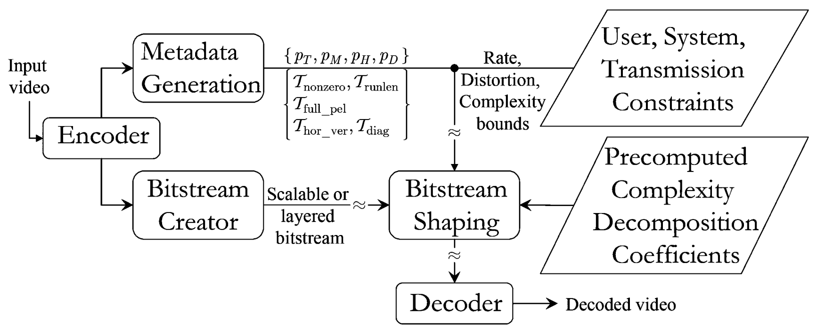

Input video
Encoder
Metadata Generation
$\{p_T, p_M, p_H, p_D\}$
$\mathcal{T}_{\text{nonzero}}, \mathcal{T}_{\text{runlen}}$
$\mathcal{T}_{\text{full\_pel}}$
$\mathcal{T}_{\text{hor\_ver}}, \mathcal{T}_{\text{diag}}$
Rate, Distortion, Complexity bounds
User, System, Transmission Constraints
Bitstream Creator
Scalable or layered bitstream
Bitstream Shaping
Precomputed Complexity Decomposition Coefficients
Decoder
Decoded video

Fig. 2. Schematic of the proposed bitstream-shaping video coding system. The symbol $\approx$ indicates the potential existence of a transmission channel.

with v indicating the maximum number of motion vectors assigned for each pixel of AU i. Additional decomposition functions can be defined for modeling the complexity of deblocking operations, as in our previous work [9]. Finally, for each AU i, the interpolation complexity decomposition functions are defined in a similar manner as

$$\mathcal{T}_{\text{hor\_ver}}(p_H(i)) = p_H(i) \cdot X_l \cdot Y_l, \quad \mathcal{T}_{\text{diag}}(p_D(i)) = p_D(i) \cdot X_l \cdot Y_l. \tag{5}$$

2) Definition of the Complexity Modeling Framework: We can now use the following set of equations for the complexity modeling of each module of Fig. 1, assuming that every GOP consists of N AUs. Starting from the modeling of the complexity metrics for entropy decoding and inverse transformation, we have[2]

$$\mathbf{cm}^N = \mathbf{C}^N_{\text{nonzero}} \cdot \boldsymbol{\mathcal{T}}^N_{\text{nonzero}} + \mathbf{C}^N_{\text{runlen}} \cdot \boldsymbol{\mathcal{T}}^N_{\text{runlen}} + \mathbf{C}^N_{\text{dec\_const}} \cdot \mathbf{1} \tag{6}$$

with $\mathbf{cm}^N = [\text{RS}^N \ \text{FC}^N]^T$ the generic complexity metrics for entropy decoding and inverse transformation of N AUs; the complexity decomposition coefficients are the $2 \times N$ matrices

$$\mathbf{C}^N_{\text{dec\_op}} = \begin{bmatrix} C_{\text{RS,dec\_op}}(p_T(1)) & \cdots & C_{\text{RS,dec\_op}}(p_T(N)) \\ C_{\text{FC,dec\_op}}(p_T(1)) & \cdots & C_{\text{FC,dec\_op}}(p_T(N)) \end{bmatrix},$$
$$\text{where dec\_op} = \{\text{nonzero}, \text{runlen}, \text{dec\_const}\}.$$

The elements of these matrices form inner products with the corresponding complexity decomposition functions $\boldsymbol{\mathcal{T}}^N_{\text{dec\_op}} = [\mathcal{T}_{\text{dec\_op}}(p_T(1)) \cdots \mathcal{T}_{\text{dec\_op}}(p_T(N))]^T$ defined by (1) and (3); we additionally define $\boldsymbol{\mathcal{T}}^N_{\text{dec\_const}} \equiv \mathbf{1}$.

Similarly, for the motion compensation and interpolation modules, we have

$$\mathbf{cm}^N_{\text{pred\_op}} = \mathbf{c}^N_{\text{pred\_op,func}} \cdot \boldsymbol{\mathcal{T}}^N_{\text{pred\_op}} + \mathbf{c}^N_{\text{pred\_op,const}} \cdot \mathbf{1} \tag{7}$$

where $\mathbf{cm}^N_{\text{pred\_op}} = \{\text{MC}^N, \text{IO}^N_H, \text{IO}^N_D\}$, with, respectively $\text{pred\_op} = \{\text{full\_pel}, \text{hor\_ver}, \text{diag}\}$, the generic complexity metrics for the prediction modules (motion compensation and interpolation) utilized for N AUs; in this case, the vectors of complexity decomposition coefficients, are defined as $\mathbf{c}^N_{\text{pred\_op,term}} = [C_{\text{pred\_op,term}}(p_U(1)) \cdots C_{\text{pred\_op,term}}(p_U(N))]$, $\text{term} = \{\text{func}, \text{const}\}$, and the elements of these vectors form inner products with the motion-related complexity decomposition functions $\boldsymbol{\mathcal{T}}^N_{\text{pred\_op}} = [\mathcal{T}_{\text{pred\_op}}(p_U(1)) \cdots \mathcal{T}_{\text{pred\_op}}(p_U(N))]^T$ defined by (4) and (5), where $U = \{M, H, D\}$.

[2]All superscripts indicate the number of AUs included in the complexity modeling of a GOP, while subscripts indicate the related operation or operation type. Moreover, vectors and matrices are respectively indicated in lower and upper case boldface font, with the exception of the vectors of decomposition functions which are indicated in uppercase, boldface calligraphic font. Finally, $\mathbf{1}$ indicates the column vector of ones, whose length is identified from the context.

3) Estimation of the Complexity Decomposition Coefficients—Off-Line Training Process: The estimation of these coefficients is performed based on an offline training process, which is described in detail in the following steps.

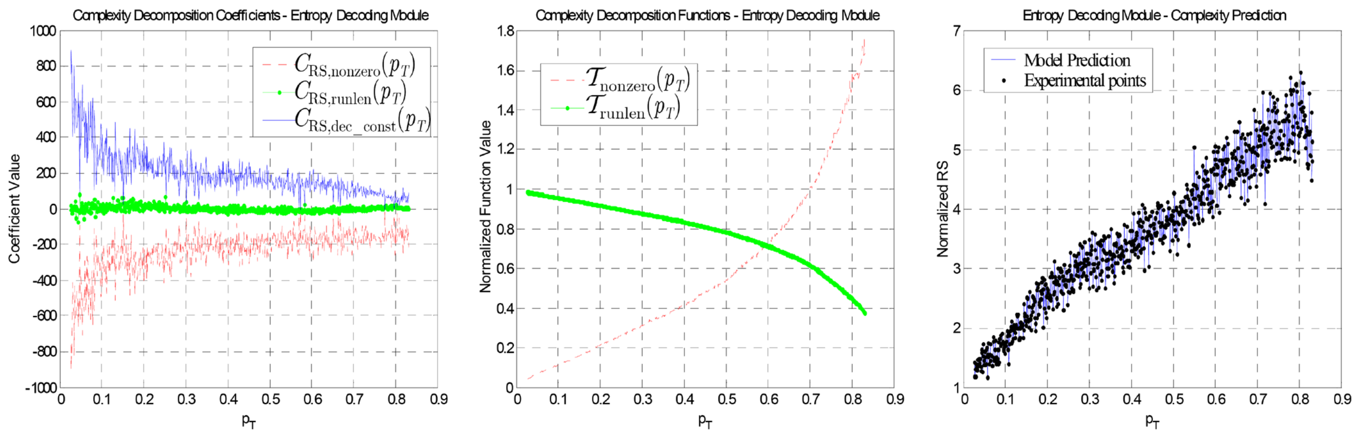
1) We selected a representative set of standard CIF[3] test video sequences and target bit rates. The spatial-domain version of the MCTF-based[4] video coder used in our experiments [8] encoded the first 256 frames of nine sequences: "Stefan," "Silent," "Sailormen," "City," "Raven," "Football," "Coastguard," "Paris," "Harbour." Subsequently, 13 bitstreams were extracted at equidistant bit rates selected between 200 kb/s and 1.5 Mb/s.
2) During the decoding of each bitstream, the complexity decomposition functions and variables mentioned before for the various modules are experimentally measured for the processing of each frame. Hence, we accumulate a pool of $J = 256 \times 9 \times 13$ experimental points for each complexity variable (p_T, p_M, p_H, p_D) and function $(\mathcal{T}_{\text{nonzero}}, \mathcal{T}_{\text{runlen}}, \mathcal{T}_{\text{full\_pel}}, \mathcal{T}_{\text{hor\_ver}}, \mathcal{T}_{\text{diag}})$.
3) By profiling the software realization of the selected decoder, J experimental measurements are generated for each of the metrics derived in (6), (7). We sort the experimental data of all the sequences and bit rates of interest, by grouping the experimental values of each complexity variable in G sets of P consecutive values (with $G = \lceil J/P \rceil$). In our training, after experimentation with several choices, we selected $P = 25$, thereby $G \simeq 1200$.
4) We estimate the complexity decomposition coefficients using linear regression for each of the complexity modeling (6), (7) within each group $g = \{1, \ldots, G\}$. For example, for the entropy decoding part of (6)

$$\mathbf{rs}^g = \text{diag}\left(\mathbf{c}^g_{\text{nonzero}}\left(\overline{p^g_T}\right)\right) \cdot \boldsymbol{\mathcal{T}}^g_{\text{nonzero}}\left(\overline{p^g_T}\right)$$
$$+ \text{diag}\left(\mathbf{c}^g_{\text{runlen}}\left(\overline{p^g_T}\right)\right) \cdot \boldsymbol{\mathcal{T}}^g_{\text{runlen}}\left(\overline{p^g_T}\right) + \mathbf{c}^g_{\text{dec\_const}}\left(\overline{p^g_T}\right)^T \tag{8}$$

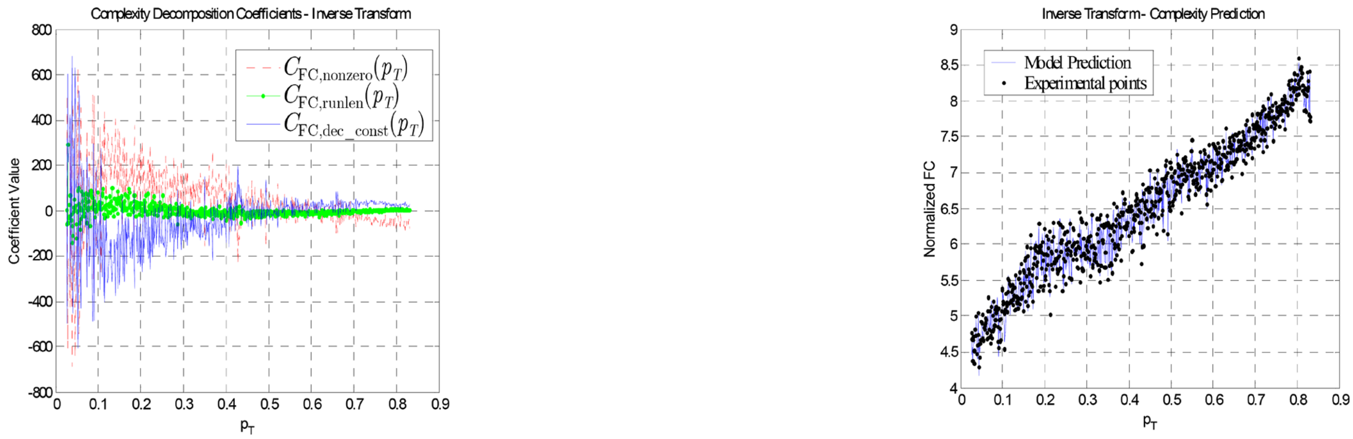
where the superscripts indicate the current group (g) of values, $\overline{p^g_T} = (1/P)\sum_{i=1}^P p^g_T(i)$, $\mathbf{rs}^g = [RS^g_1 \cdots RS^g_P]^T$, $\boldsymbol{\mathcal{T}}^g_{\text{dec\_op}}(\overline{p^g_T}) = [\mathcal{T}^g_{\text{dec\_op}}(p^g_T(1)) \cdots \mathcal{T}^g_{\text{dec\_op}}(p^g_T(P))]^T$, and $\mathbf{c}^g_{\text{dec\_op}}(\overline{p^g_T}) = [C^g_{\text{dec\_op}}(p^g_T(1)) \cdots C^g_{\text{dec\_op}}(p^g_T(P))]$. Hence, P represents the size of the linear system of (8). Larger values lead to estimation of complexity coefficients over a larger span of data, which is generally preferable since it reduces the effect of outlier points. However, this also leads to smaller granularity in the intervals of complexity variables since each group $g = \{1, \ldots, G\}$ contains more experimental points. The complexity decomposition coefficients derived with $P = 25$ were experimentally found to produce the best prediction, over all our simulations.

[3]CIF: Common Interchange Format.

[4]MCTF: Motion Compensated Temporal Filtering.



(a)



(b)

Fig. 3. From left to right, for all the range of the percentages of nonzero coefficients (p_T) collected from the training data set: (a) complexity decomposition coefficients corresponding to (9), complexity decomposition functions, and the resulting model-based prediction (lines) versus the experimental data (points) and (b) complexity decomposition coefficients for the inverse transform and the resulting model-based prediction (lines) versus the experimental data (points). The normalization was performed by dividing with the number of pixels of each decoded frame (AU).

For each group, the exact derivation of these coefficients based on linear regression is

$$\begin{aligned}&\left[C^{g,*}_{\text{nonzero}}\left(\overline{p^g_T}\right)\; C^{g,*}_{\text{runlen}}\left(\overline{p^g_T}\right)\; C^{g,*}_{\text{dec\_const}}\left(\overline{p^g_T}\right)\right]^T\\ &= \left(\left[\boldsymbol{\mathcal{T}}^g_{\text{nonzero}}\left(\overline{p^g_T}\right)\; \boldsymbol{\mathcal{T}}^g_{\text{runlen}}\left(\overline{p^g_T}\right)\; \mathbf{1}\right]^T\right.\\ &\qquad \left.\left[\boldsymbol{\mathcal{T}}^g_{\text{nonzero}}\left(\overline{p^g_T}\right)\; \boldsymbol{\mathcal{T}}^g_{\text{runlen}}\left(\overline{p^g_T}\right)\; \mathbf{1}\right]\right)^{-1}\\ &\quad \cdot \left[\boldsymbol{\mathcal{T}}^g_{\text{nonzero}}\left(\overline{p^g_T}\right)\; \boldsymbol{\mathcal{T}}^g_{\text{runlen}}\left(\overline{p^g_T}\right)\; \mathbf{1}\right]^T \mathbf{rs}^g \end{aligned} \tag{9}$$

and we set $\mathbf{c}^{g,*}_{\text{dec\_op}}(\overline{p^g_T}) \equiv C^{g,*}_{\text{dec\_op}}(\overline{p^g_T}) \cdot \mathbf{1}^T$ for every $\text{dec\_op} = \{\text{nonzero}, \text{runlen}, \text{dec\_const}\}$. The process is performed similarly for the remaining complexity decomposition coefficients of other modules.

Once they have been obtained for each group and each module, these coefficients are kept in lookup tables to be used during the complexity estimation process. In Fig. 3(a), we illustrate the complexity decomposition coefficients corresponding to (9) for every group g, along with the complexity decomposition functions calculated for each interval, and the resulting model-based complexity estimate. Similarly, Fig. 3(b) illustrates the complexity decomposition coefficients corresponding to the inverse transform, and the resulting model-based complexity estimate. Notice that the complexity functions and decomposition coefficients shown in Fig. 3 exhibit certain trends. When the decoded transform-domain representation of the video frame consists of a sparse representation (p_T is small), then $\mathcal{T}_{\text{runlen}}(p_T \to 0)$ is large and $\mathcal{T}_{\text{nonzero}}(p_T \to 0)$ is small, since the probability of zero run lengths in the source data approaches one and the probability of significant coefficients approaches zero. However, the decoding complexity is not significantly affected by the zero run lengths (since they are always treated by low-cost operations—such as the decoding of a "nonsignificance" symbol), but rather from the (sparse) nonzero coefficients in this case, since $C_{\{\text{RS,FC}\},\text{runlen}}(p_T \to 0)$ remain small and $C_{\{\text{RS,FC}\},\text{nonzero}}(p_T \to 0)$ have large magnitude. Equivalent observations can be made for large values of p_T. In general, depending on the variation of the compressed data across each GOP (as captured by the complexity variables p_T, p_M, p_H, p_D of each video frame), the decomposition functions encapsulate the source statistics, while the decomposition coefficients consist of a mapping of the source statistics to the decoding algorithm and its particular implementation. Finally, although $C_{\{\text{RS,FC}\},\text{nonzero}}(p_T)$ and $C_{\{\text{RS,FC}\},\text{dec\_const}}(p_T)$ appear to be, respectively, antisymmetric in certain intervals of p_T, their effect is not cancelled out, since (6) shows that the contributions of $C_{\{\text{RS,FC}\},\text{nonzero}}(p_T)$ are weighted by $\mathcal{T}_{\text{nonzero}}(p_T)$.

For our experiments, each GOP contains $N = 16$ video frames (AUs). Since we utilized a fairly representative set of sequences and bit rates for the training process, the complexity-coefficients versus complexity-variables graphs [such as the one of left part of Fig. 3(a) and (b)] represent the complexity "decomposition" of each decoding module and depend on the chosen decoding algorithm and its realization. Since they are estimated by measuring the number of times specific functions

TABLE I

AVERAGE PREDICTION ERROR OF THE PROPOSED METHOD FOR ENTROPY DECODING AND INVERSE TRANSFORM IN FOUR CIF VIDEO SEQUENCES ("TEMPETE," "FOREMAN," "MOBILE," AND "NEWS") FOR VARIOUS BIT RATES. THE PREDICTION ERROR FOR THE MOTION COMPENSATION, HORIZONTAL–VERTICAL I NTERPOLATION, AND DIAGONAL INTERPOLATION IS ALSO REPORTED AT THE BOTTOM OF THE TABLE (SAME FOR ALL BIT RATES)

Bitrate	Tempete		Foreman		Mobile		News	
(kbps)	RS (%)	FC (%)	RS (%)	FC (%)	RS (%)	FC (%)	RS (%)	FC (%)
384	20.19	4.95	13.97	11.62	5.11	8.27	13.71	5.95
512	18.12	3.51	10.89	3.69	9.84	7.57	19.66	5.49
896	5.90	4.82	13.09	3.56	7.71	5.37	7.98	6.15
1024	8.84	3.58	11.80	4.52	10.73	6.25	9.92	6.28
1280	5.55	5.51	7.91	1.78	10.42	11.81	6.15	6.61
1536	13.35	6.75	6.61	3.01	7.94	10.01	8.57	5.44
Average:	**11.99**	**4.85**	**10.71**	**4.70**	**8.63**	**8.21**	**11.00**	**5.99**

Motion	MC	IO_H	IO_D	MC	IO_H	IO_D	MC	IO_H	IO_D	MC	IO_H	IO_D
Parameters	3.16	3.77	4.10	1.39	3.92	1.17	6.00	5.00	10.91	13.88	3.77	7.76

of the decoder are invoked for a large set of representative inputs, the complexity decomposition coefficients exhibit considerable variance. However, the right part of Fig. 3(a) and (b) shows that, when these coefficients are multiplied ("modulated") by the complexity decomposition functions (as shown in (8)), an accurate prediction of the experimental measurements (points) is obtained for each group g of p_T values. To quantify this, for each module of the decoding system we have calculated the residual variance around the regression line based on the Pearson's coefficients. In particular, for the entire training set of video sequences, the residual variance (in percentage of the original variance of the complexity measurements) for the texture decoding was 49.98%. The inverse transform percentile residual variance was found to be 27.17%. Concerning motion compensation, the percentile residual variability was 3.22%, while for horizontal or vertical interpolation and for the diagonal interpolation it was 5.60% and 5.41%, respectively.

These results justify the linearity of the proposed complexity modeling with respect to the proposed complexity decomposition functions for each AU, since the error variance around the regression line is significantly reduced. For the particular case of the entropy decoding that exhibits high residual variance, additional decomposition functions that are particular to the chosen decoding scheme, or the exclusion of outlier measurement points that correspond to extreme cases, can further decrease the residual variance. Finally, it is important to notice that, due to the fact that complexity modeling is performed with a GOP granularity (shown in (6) and (7)), the averaging of several AUs decreases the error variance for our simulations with new sequences. As a result, the error margin observed in practice for complexity modeling with a GOP granularity was reasonably small, as it will be shown by the experimental results of Sections III and IV.

4) Application of the Complexity Modeling for Decoder-Complexity Estimation: The real-time complexity modeling is performed as follows. During encoding, the complexity variables and functions are calculated for each potential bitstream truncation point. In particular, if we assume a typical scenario of three truncation points for the texture bitstream of each temporal level (with a total of four temporal decomposition levels), we precalculate 12 values for each of the p_T, $\mathcal{T}_{\text{nonzero}}$, and $\mathcal{T}_{\text{runlen}}$ per AU. Similarly, with two truncation points for the motion bitstream of each temporal level (under motion scalability), we precalculate eight values for each of the p_M, p_H, p_D, $\mathcal{T}_{\text{hor\_ver}}$, and $\mathcal{T}_{\text{diag}}$. The computational complexity for these calculations is minimal versus the overall encoding complexity. Similarly, the compression overhead due to the presence of these values in the bitstream is insignificant, i.e., less than 10 kb/s in the above example for the entire set of truncation points corresponding to the highest bit rate.

During the decoding of each GOP, the complexity variables and functions of every frame (AU) are used in (6), (7) in conjunction with the precalculated values of the corresponding complexity decomposition coefficients. In this way, the model prediction for each complexity metric is accomplished. The required computational overhead involves only a small number of MAC operations per GOP for the calculation of the model and is negligible in comparison to the decoding complexity. Notice that, instead of complexity estimation with a GOP granularity, other granularities could be considered as well, depending on the selected coding scheme and the desired bitstream-shaping intervals.

III. MODEL VALIDATION

Four sequences not belonging to the training set were selected for the purpose of validating the proposed method. For each sequence, the average model prediction error over ten GOPs is presented in Table I for a variety of decoding bit rates, while Table II presents the corresponding results with two other online complexity estimation methods, adapted from the relevant literature on system resource prediction [12]–[14]. In particular, the first comparison is performed with the autoregression with mean adaptation (ARM) algorithm from Liang *et al.* [12, Sec. 4], which was shown to outperform various alternative autoregressive methods for system resource prediction [12], [14]. Per temporal decomposition level, this method predicts the complexity metrics of each GOP based on the mean of observed values of the corresponding level of the previous GOP.

The results of Table II indicate that this method is less accurate than the proposed method by a significant margin. Consequently, even though this method has the advantage of observing the dynamic behavior of the previous GOP's decoding, the lack of a source-adaptive driving mechanism for the modification of the prediction appears to be important. For this reason, as a second comparison, we opted for a widely used adaptive linear prediction method, where the knowledge of the transmitted complexity variables can be exploited for the adaptive prediction of the complexity metrics of each GOP. In this case, for each temporal decomposition level, we perform a linear mapping between the input complexity variables (p_T, p_M, p_H, p_D) and the observed complexity metrics (RS, FC, MC, IO_H, and IO_D) of the previous GOP, e.g., $RS(i) = \alpha \cdot p_T(i) + \beta$ for each decoded frame i within a temporal level. The parameters of the mapping are estimated based on linear regression and are used for the prediction of the complexity metrics of the current GOP in conjunction with the complexity variables of the current GOP. This method performs better than ARM and also outperforms the proposed method in certain cases where the complexity variation is small (e.g., low motion sequences or higher bit rates). Notice, however, that both ARM and the adaptive linear prediction use online measurements, which may be difficult to obtain in a real decoding system, since modifications at the decoder software would be required. In addition, the adaptive linear regression is computationally demanding since it performs online parameter estimation for the linear mapping for each decoded GOP.

TABLE II
RESULTS CORRESPONDING TO TABLE I WITH THE USE OF THE ARM MODEL [12], [14], AND WITH ADAPTIVE LINEAR REGRESSION (IN PARENTHESES) [13]

Bitrate (kbps)	Tempete		Foreman		Mobile		News	
	RS (%)	FC (%)	RS (%)	FC (%)	RS (%)	FC (%)	RS (%)	FC (%)
384	16.27 (13.48)	12.39 (8.49)	16.76 (11.73)	19.77 (16.51)	13.95 (5.49)	17.86 (12.49)	14.38 (19.29)	19.26 (11.91)
512	16.40 (8.96)	12.47 (6.75)	17.03 (12.82)	17.33 (14.62)	13.63 (8.58)	17.76 (13.27)	14.50 (9.12)	19.22 (14.29)
896	15.05 (8.22)	10.74 (0.95)	15.49 (22.75)	17.34 (7.97)	12.33 (5.10)	13.61 (6.00)	14.17 (7.23)	19.55 (6.99)
1024	14.71 (8.38)	10.78 (1.14)	15.61 (18.11)	17.45 (11.58)	12.35 (6.55)	13.59 (5.92)	14.27 (7.75)	19.57 (6.94)
1280	14.64 (13.17)	11.02 (2.27)	15.22 (8.16)	12.95 (4.51)	11.68 (5.48)	10.79 (4.85)	14.17 (8.98)	18.07 (5.52)
1536	14.71 (5.86)	11.15 (1.02)	14.99 (12.33)	13.03 (3.01)	11.74 (5.78)	10.83 (4.40)	13.68 (4.71)	16.64 (4.52)
Average:	**15.29 (9.68)**	**11.42 (3.44)**	**15.85 (14.32)**	**16.31 (9.70)**	**12.61 (6.16)**	**14.07 (7.82)**	**14.20 (9.51)**	**18.72 (8.36)**

	Tempete			Foreman			Mobile			News		
Motion Parameters	MC	IO_H	IO_D	MC	IO_H	IO_D	MC	IO_H	IO_D	MC	IO_H	IO_D
	13.67 (4.26)	16.52 (5.38)	17.26 (6.75)	15.66 (5.18)	19.58 (5.42)	16.47 (8.21)	12.17 (5.38)	17.35 (12.42)	20.12 (12.70)	15.76 (6.77)	44.35 (35.62)	52.76 (62.08)

TABLE III
AVERAGE PREDICTION ERROR FOR MOTION COMPENSATION AND FRACTIONAL-PIXEL INTERPOLATION IN FOUR CIF VIDEO SEQUENCES (SEEN IN TABLE I) FOR VARIOUS BITSTREAM-SHAPING CONFIGURATIONS, WHERE SEVERAL TEMPORAL LEVELS OR MOTION PARAMETERS (INTERPOLATION DATA) ARE OMITTED

Skipped Temporal levels	MC (%)	IO_H (%)	IO_D (%)
0	6.10	4.12	5.98
1	5.74	10.37	3.89
2	5.80	9.77	3.23
2, skipped motion data	5.80	8.67	3.12
Average:	**5.86**	**10.50**	**3.87**

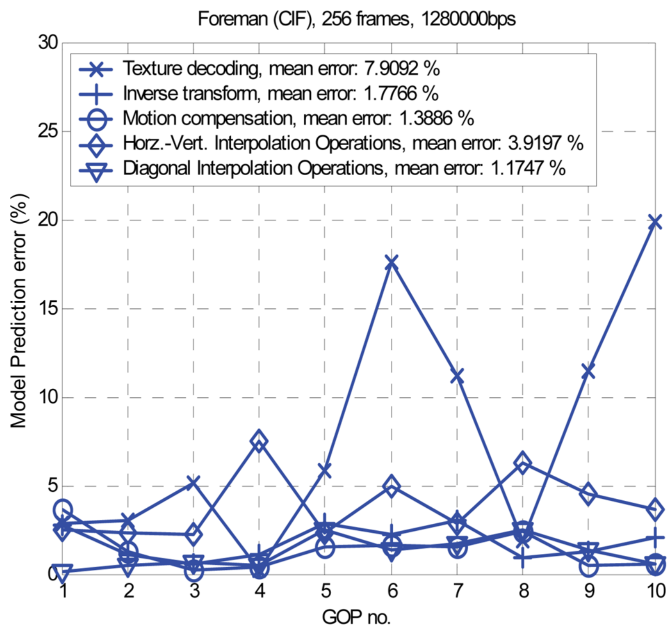


Fig. 4. Example of the prediction error for various GOPs of the "Foreman" sequence.

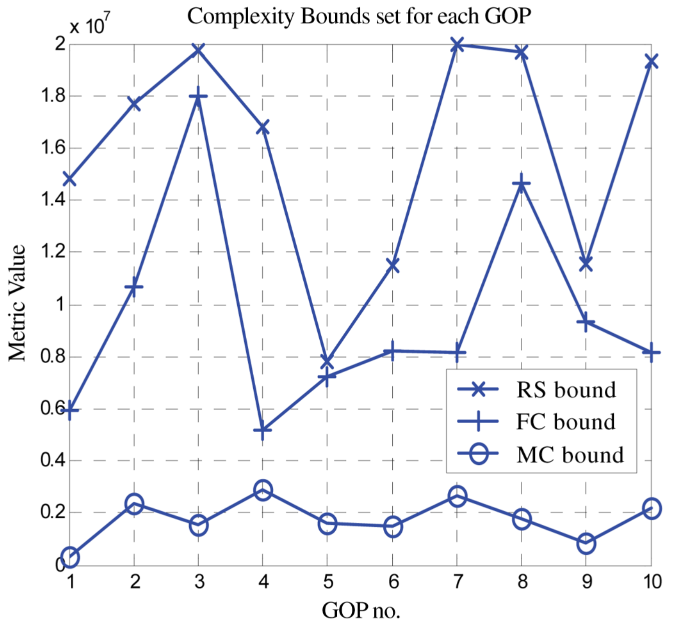


Fig. 5. Complexity upper bounds set for each GOP.

Overall, the proposed algorithm provides accurate predictions under varying decoding configurations without considering the specific video decoding algorithm characteristics such as the particular MCTF process, the number of decoded temporal levels, the chosen motion information, etc. To the contrary, this is achieved in a generic manner based only on the derived complexity decomposition functions and the complexity decomposition coefficients for each AU of each GOP. This is further illustrated by the results of Table III, where various decoding configurations were considered, while Fig. 4 illustrates a typical example of the prediction error for each metric across various GOPs. Finally, the required training process is performed offline, and the complexity prediction can be achieved with minimal complexity overhead and without the need for online decoder feedback. As a result, the complexity adaptation via bitstream shaping can be performed at the server side and the decoder only needs to provide bounds for each complexity metric. This is further elaborated in the next section.

IV. BITSTREAM SHAPING BASED ON COMPLEXITY CONSTRAINTS

We present an example of bitstream shaping for each video GOP under complexity and distortion constraints. In particular, for each of the test sequences used in Section III, we set random complexity bounds for each GOP in terms of upper bounds for each complexity metric. This is done as a simulation of the time-varying resources available at the receiver side for the decoding of each GOP. In a video decoding scenario for a real multitasking multimedia platform these complexity bounds would be determined in real-time based on the system-resource utilization, the energy consumption and the current configuration of the processor [10]. We consider bounds for the entropy decoding, the inverse transform and the motion compensation metrics, since they capture the whole range of different operations (symbol decoding, motion compensation and filtering) performed

```
1.  For each video GOP
2.    Receive $\mathrm{RS}_{\max}, \mathrm{FC}_{\max}, \mathrm{MC}_{\max}$ for the current GOP to be transmitted to the decoder
3.    For each bitstream truncation point $k$ out of $K$ total truncation points with admissible distortion
4.      For each AU $i$ of the current GOP
5.        Extract $p_T(i,k)$, $p_M(i,k)$, $p_H(i,k)$, $p_D(i,k)$ and $\mathcal{T}_{\text{nonzero}}(p_T(i,k))$, $\mathcal{T}_{\text{runlen}}(p_T(i,k))$, $\mathcal{T}_{\text{full\_pell}}(p_M(i,k))$, $\mathcal{T}_{\text{hor\_ver}}(p_H(i,k))$,
          $\mathcal{T}_{\text{diag}}(p_D(i,k))$
6.      Calculate $\mathrm{RS}(k)$, $\mathrm{FC}(k)$, $\mathrm{MC}(k)$, $\mathrm{IO}_H(k)$, $\mathrm{IO}_D(k)$ with (6),(7) in conjunction with the precalculated values of
        the corresponding complexity-decomposition coefficients
7.      Set $\text{diff\_C}(k) = \kappa \cdot |\mathrm{MC}_{\max} - \mathrm{MC}(k)| + \lambda \cdot |\mathrm{FC}_{\max} - (\mathrm{FC}(k) + \mathrm{IO}_H(k) + \mathrm{IO}_D(k))| + \mu \cdot |\mathrm{RS}_{\max} - \mathrm{RS}(k)|$
8.    Select $k^* = \arg\min_{1 \le k \le K} [\text{diff\_C}(k)]$
```

Fig. 6. Pseudocode for the proposed bitstream-shaping process. Notation $p_U(i,k), U = \{T, M, H, D\}$, indicates the complexity variable of the kth truncation point of the ith AU. Parameters κ, λ, μ indicate the importance of each metric in the overall complexity estimate $\text{diff\_C}(k)$ and they are typically system dependent. In our simulations, for simplicity we set $\kappa = \lambda = \mu = 1$.

TABLE IV
SELECTED OPERATING POINT FOR EACH GOP. THE SEQUENCE "FOREMAN" WAS USED. THE AVERAGE COMPLEXITY ERROR IS THE AVERAGE (OVER ALL THREE METRICS SHOWN IN FIG. 5) OF THE ABSOLUTE ERROR BETWEEN THE UPPER BOUND SET FOR EACH GOP AND THE OBTAINED COMPLEXITY

GOP no.	Dec. Temporal levels	Dec. Bitrate (kbps)	Av. Complexity error (%)	Obtained PSNR (dB)
1	2	896	4.25	36.19
2	3	896	7.12	34.20
3	4	512	2.21	33.26
4	2	512	4.23	35.81
5	2	384	1.19	34.22
6	3	512	6.79	34.72
7	4	1280	5.31	38.64
8	3	896	3.29	36.44
9	2	896	5.19	37.17
10	3	512	2.52	33.95

at the decoder side.[5] Moreover, we set a lower bound in terms of peak signal-to-noise ratio (PSNR) distortion for the decoded video at 33 dB; notice that, for each operational point, the chosen PSNR measurement quantifies in an objective manner the distortion between the selected operating point and the sequence decoded at a very high bit rate under the shaped bitstream (which corresponds to the "best" quality that this bitstream can obtain if more bit rate is provided) [16]. This can be a user or Quality-of-Service (QoS) specification and it is application-dependent. One representative result for a video sequence is presented in Table IV, under the dynamic conditions of Fig. 5. Similar results were obtained for the remaining sequences and for a large variation in the complexity bounds for each GOP.

Bitstream shaping in Table IV was performed by selecting the truncation point that best matched the complexity bounds set in real time for each GOP, as explained in the pseudocode of Fig. 6. Since each truncation point in our scalable coder corresponds to a particular bit rate and frame rate, the results of Table IV indicate the practical bit rate and temporal-level selections corresponding to the optimal truncation of each GOP. In our simulations, the distortion is experimentally measured versus the decoder-side reference video sequence since the focus on this paper is on the complexity aspects of bitstream shaping. However, model-based distortion estimates [15] could be used instead of the measured distortion. From all the potential truncation points, only the ones above the lowest permissible distortion are considered (line 3 of Fig. 6).

The results of Table IV indicate that this simple algorithm generates accurate complexity adaptation to the dynamically changing requirements that simultaneously satisfies the distortion bound. The relatively large PSNR fluctuation for various GOPs is due to the large complexity variations imposed by the randomly selected constraints of Fig. 5, as well as the attempt of the bitstream shaping to match the complexity bounds accurately, rather than provide a constant distortion. Similar to rate shaping under varying channel conditions, an algorithm that performs bitstream shaping with quasi-constant distortion can be easily derived. However, such an algorithm can potentially result in an unnecessarily conservative system-resource usage.

[5]Bounds on the interpolation process can be set separately, or incorporated with the inverse transform bounds, since both are realized by MAC operations.

V. CONCLUSION

A complexity modeling framework and its utilization for bitstream shaping in video decoders has been discussed. For each GOP of several decoded video sequences, the derived models produce accurate complexity predictions with respect to certain generic metrics that can be mapped into real complexity measurements such as timing results and energy consumption. The proposed complexity prediction is very low complex since the required training is performed offline. Nevertheless, our experiments indicate that it can outperform autoregressive and linear-mapping prediction methods from the literature, even though these methods adapt based on online training. Using the model-based complexity prediction, the proposed bitstream shaping determines admissible bitstreams with respect to upper bounds on complexity and a lower bound on distortion. Our future work will attempt to investigate additional metrics for several decoder modules, as well as their combination and mapping process for the production of bitstream shaping based on energy or performance bounds.

REFERENCES

[1] D. Sow and A. Eleftheriadis, "Complexity distortion theory," *IEEE Trans. Inf. Theory*, vol. 49, no. 3, pp. 604–608, Mar. 2003.

[2] J. Winograd and S. Nawab, "Probabilistic complexity analysis of incremental DFT algorithms," in *Proc. IEEE Int. Conf. Accoust., Speech, Signal Process. (ICASSP)*, Apr. 1997, vol. 3, pp. 1985–1988.

[3] J. Reichel, "Predicting the complexity of signal processing algorithms," in *Proc. IEEE Int. Conf. Image Processing (ICIP)*, Thessaloniki, Greece, Oct. 2001, vol. 3, pp. 318–321.

[4] D. G. Sachs, S. V. Adve, and D. L. Jones, "Cross-layer adaptive video coding to reduce energy on general purpose processors," in *Proc. IEEE Int. Conf. Image Processing (ICIP)*, Sep. 2003, pp. 25–28.

[5] M. Horowitz, A. Joch, F. Kossentini, and A. Hallapuro, "H.264/AVC baseline profile decoder complexity analysis," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 704–716, Jul. 2003.

[6] Z. He, Y. Liang, L. Chen, I. Ahmad, and D. Wu, "Power-rate-distortion analysis for wireless video communication under energy constraints," *IEEE Trans. Circuits Syst. Video Technol*, vol. 15, no. 5, pp. 645–658, May 2005.

[7] H. Schwarz, T. Hinz, H. Kirchhoffer, D. Marpe, and T. Wiegand, Technical Description of the HHI Proposal for SVC CE1 , ISO/IEC JTC1/SC29/WG11 (MPEG), Oct. 2004, m11244.

[8] Y. Andreopoulos, A. Munteanu, J. Barbarien, M. van der Schaar, J. Cornelis, and P. Schelkens, "In-band motion compensated temporal filtering," *Signal Processing: Image Commun.*, vol. 19, no. 7, pp. 653–673, Aug. 2004.

[9] M. van der Schaar and Y. Andreopoulos, "Rate-distortion-complexity modeling for network and receiver aware adaptation," *IEEE Trans. Multimedia*, vol. 7, no. 3, pp. 471–479, Jun. 2005.

[10] Y. Andreopoulos and M. van der Schaar, "Adaptive linear prediction for resource estimation of video decoding," *IEEE Trans. Circuits Syst. Video Technol.*, to be published.

[11] Z. He and S. K. Mitra, "A unified rate-distortion analysis framework for transform coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, no. 12, pp. 1221–1236, Dec. 2001.

[12] J. Liang, K. Nahrstedt, and Y. Zhou, "Adaptive multi-resource prediction in distributed resource sharing environment," in *Proc. IEEE Int. Symp. Cluster Comput. and the Grid (CCGrid)*, Apr. 2004, pp. 293–300.

[13] D. G. Sachs, W. Yuan, W. Yuan, C. J. Hughes, A. Harris, S. V. Adve, D. L. Jones, R. H. Kravets, and K. Nahrstedt, "GRACE: A hierarchical adaptation framework for saving energy," Comp. Sci. Dept., Univ. of Illinois, Urbana-Champaign, Tech. Rep. UIUCDCS-R-2004-2409, Feb. 2004.

[14] P. A. Dinda and D. R. O'Hallaron, "An evaluation of linear models for host load prediction," in *Proc. IEEE Int. Symp. High Perf. Distrib. Comput.*, Aug. 1999, pp. 87–96.

[15] M. Wang and M. van der Schaar, "Operational rate-distortion modeling for wavelet video coders," *IEEE Trans. Signal Processing*, vol. 54, no. 9, pp. 3505–3517, Sep. 2006.

[16] Y. Wang, M. van der Schaar, S.-F. Chang, and A. C. Loui, "Classification-based multidimensional adaptation prediction for scalable video coding using subjective quality evaluation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 15, no. 10, pp. 1270–1279, Oct. 2005.