A Low-Complexity Cross-Layer Optimization Algorithm for Video Communication Over Wireless Networks

Simone Milani and Giancarlo Calvagno

Abstract—Recent years have witnessed a rapid increment in video applications over wireless networks including on-demand video streaming and videophoning. This growth has also brought the need to find a good compromise in the conflict between resource limitations affecting mobile devices and the desire for high quality multimedia services. It is possible to face this problem adopting a cross-layer strategy that jointly tunes the parameters of each layer in the network protocol stack. In this optimization strategy, complexity is one of the most significant issues because of the limited computational resources and power supply. The paper presents a low-complexity cross-layer algorithm that is able to jointly tune the parameters of different protocol layers by adopting simple but effective models. The quality of the reconstructed video sequence, the produced bit rate, and the service class associated to each packet are seen as functions of the percentage of null DCT coefficients. This modelling permits to find a closed-form solution to the joint optimization problem that can be computed with a limited number of operations and grants, at the same time, a good visual quality in the reconstructed sequence.

Index Terms—cross-layer optimization, bit allocation, cross-packet codes, packet classification, video coding, H.264/AVC, IEEE 802.11, DiffServ, video transmission.

I. INTRODUCTION

The advent of wireless multimedia communications has brought to evidence that the traditional configurations of network protocol stacks are not adequate for video delivery over time-varying channels. The modularization of layered architectures permits great flexibility and interoperability among heterogeneous networks and devices, but at the same time, it could lead to significant inefficiencies considering the issues raised by the transmission of video sequences over wireless communication systems [1].

During the last years a considerable research effort has been made to investigate efficient cross-layer solutions that grant a high Quality-of-Service (QoS) level to the end user by allowing a synergetic interaction between different protocol layers. The main goal of these architectures is to improve the performance of the transmission by jointly tuning the parameters of each layer according to holistic algorithms.

In the traditional network protocol stacks the design of a joint optimization strategy, which is able to compute the best parameters setting for each layer and maximize the quality of the sequence reconstructed at the decoder, is not a trivial task, as it is depicted in [2].

Let $S_L$, $i = 1, \ldots, 7$, denote the set of transmitting and coding parameters related to layer $L_i$ of the OSI protocol stack. The set of joint possible configurations is identified by the Cartesian product $S = S_{L_1} \times S_{L_2} \times \cdots \times S_{L_7}$.

Then, given the vector $x$ of the current channel conditions, the cross layer optimization problem consists in finding the optimal set

$$S_{\text{opt}}(x) = \arg \max_{S \in S} \text{PSNR}(S(x))$$

such that

$$\text{rate}(S(x)) \leq R_b \quad \text{and} \quad \text{delay}(S(x)) \leq D_{\text{max}}$$

where $\text{PSNR}(\cdot)$ is an estimated Peak Signal-to-Noise Ratio. The value $R_b$ is the available transmission bit rate while the bound $D_{\text{max}}$ denotes the maximum accepted delay. Beyond bandwidth and delay limitations, power and fairness constraints are commonly considered too.

Denoting with $N_L$, the number of strategies available at layer $L_i$, the cardinality $N_{CL}$ of the parameters space (including different strategies and parameter values) for cross-layer techniques is

$$N_{CL} = \prod_{L_i \in \mathbf{L}} N_{L_i} > N_L = \sum_{L_i \in \mathbf{L}} N_{L_i}$$

where $\mathbf{L}$ is the set of layers involved in the optimization and $N_L$ is the cardinality of the parameters space for the layered approach. The cross-layer optimization has to search the optimal configuration among an extended set of parameter settings, and this implies a high computational cost. As [3] reports, the problem is further complicated by:

- the difficulty to derive tractable analytical expressions for PSNR, delay, power, fairness, etc.;
- the computational complexity of the proposed solving methods;
- the continuous changing of the channel conditions $x$, requiring a continuous update of the system parameters.

Different analytic techniques have been proposed to solve this cross-layer optimization problem (convex programming, Lagrange duality, stochastic stability and many others [3]), together with heuristic approaches [4], learning and classification methods [5], bargaining theory [6].

This paper shows how it is possible to design an efficient optimization algorithm by adequately modelling the parameters of the video coder as functions of the percentage $\rho$ of null
quantized DCT coefficients (called zeros), which has already proved to be an accurate modelling parameter for the coding bit rate [7], [8]. These functions are used to design a low complexity cross-layer bit allocation algorithm that adaptively tunes the video source coder parameters, the FEC channel coder rate, and the transmission priority of each packet at MAC level in order to maximize the video quality.

In Section II we provide an overview of different cross-layer optimization techniques, which have been presented in literature. Section III will present how the percentage of zeros can be used to model the bit rate and the distortion produced by a video coder. Section IV will show the relation existing between the distortion introduced in the sequence by the loss of a packet and the corresponding value of \( \rho \) for that packet. This relation is used to define an efficient strategy to tune the FEC channel coder. In Section V a similar rule will be applied to assign different priorities to video packets in a DiffServ-enabled network and in an IEEE802.11e-like network. Section VI will present a low-cost cross-layer strategy that includes all these solutions and maximizes the estimated PSNR at the receiver. The simulation results reported in Section VII will show that on a simulated IEEE 802.11 network the proposed solution permits to improve the PSNR value up to 2 dB with respect to a blind approach where no interaction among different layers is performed. Finally conclusions and future improvements are reported in Section VIII.

II. Overview of Some Cross-Layer Optimization Solutions

During the last year, the study of effective cross-layer transmission strategies that enhance the quality of the video sequence reconstructed at the decoder has lead to a wide panorama of solutions involving different layers and optimization algorithm.

Some of the proposed techniques aim at providing Unequal Error Protection (UEP) to each packet according to the characteristics of the coded signal (see [9]). In [10] Zhai et al. propose a joint optimization algorithm that tunes the coding parameters of the video source coder and the channel coder at different layers.

Other solutions rely on the possibility of retransmitting the lost packets at network layer according to a feedback channel [11]. In [12] Zhai et al. present a retransmission policy that can adaptively choose whether to retransmit or to protect the current packet by adding redundant information in the packet stream.

In case the transmitting network specifies different classes of service in data delivery, it is possible to assign to each packet the most appropriate QoS in order to vary the corresponding loss probabilities and time delays [13]. Different classes of service can also be defined at the lowest levels. The algorithm presented in [14] relies on the EDCA algorithm of the IEEE 802.11e protocol and distributes H.264/AVC video packets among different transmission queues at MAC level.

At link layer, cross-layer strategies try to optimize the transmission power rising or decreasing the SNR level according to the characteristics of the video sequence (see [15], [16]).

All these solutions can be jointly combined to optimize the final performance [17], [18], but the computational complexity becomes critical because of the large number of parameters that are involved in the optimization process.

Moreover, the algorithms adopted to parameterize and evaluate the expected video quality of the sequence at the decoder play a significant role too. For example, all the joint source-channel coding schemes based on the ROPE algorithm (see [19], [20]) require a significant amount of operations to compute moments and distortions.

In addition, the joint optimization algorithm has also to deal with the reliability of the estimated channel and network conditions, like loss probabilities. These methods could lead to a severely degraded performance in case the estimated channel parameters do not correspond to the actual ones.

Many algorithms that were proposed in literature are not suitable for wireless systems since the required computational complexity violates the severe power limitations of mobile terminals and an accurate estimation of channel and network parameters is needed. In order to overcome these problems, some authors have tried to simplify their approaches introducing some approximations and reducing the computational complexity. In our algorithm we are able to reduce the parameter space (and as a consequence, the computational complexity as well) by characterizing accurately the functions in eq. (1) with simple models and by introducing relations between different parameters that improve the performance of the optimization algorithm without significantly increasing the computational complexity. The following sections will present how the proposed modelization is applied at different levels of the protocol stack.

III. Rate-Distortion Modelling at Application Layer

Most of the complexity issues related to a joint cross-layer optimization concern the number of parameters that have to be optimized for each layer and the difficulty of characterizing the function in eq. (1) with a simple analytical expression. At the application level (i.e. at the source coder) this problem leads to the investigation of an efficient rate-distortion model [21] that accurately characterizes the behavior of the video coder and permits the estimation of an optimal coding setting at a reasonable computational cost.

In the technical literature, most rate control and optimization algorithms are based on hyperbolic R-D models, where bit rate and distortion are functions of the quantization step (e.g. [22]). In [7] He et al. present a better solution to parameterize the number of bits produced by a video encoder. The coding rate \( R \) is well represented by a linear function of the percentage \( \rho \) of null quantized transform coefficients (also called zeros)

\[
R(\rho) = m\rho + q, \tag{3}
\]

where \( q \) is the estimated number of overhead bits that are not related to DCT coefficients, and \( m \) is the ratio between the number of bits that code the transform coefficients and \( \rho \). Experimental results (see [7], [8]) show that the “zeros” parameterization also suits well when applied to the H.264/AVC protocol stack.
coder since the linear relation of eq. (3) proves to be satisfied for different kinds of video sequences, independently of the nature of the prediction (spatial, temporal or temporal bi-directional). Moreover, the number of zeros for a single transform block can be easily obtained since it is one of the configuring parameters for the entropy coder defined in the H.264/AVC standard (see [23]).

Empirical data also reveal that there is a linear relation between $\rho$ and the relative PSNR decrement $\delta$PSNR for the $i$-th frame which can be expressed as

$$\delta$PSNR$ = \frac{PSNR_{i-1} - PSNR_i}{PSNR_{i-1}}$ (4)

where PSNR$_i$ is the PSNR value for the $i$-th frame. This relation can be analytically derived from the equations that parameterize the source coding distortions reported in [24]. Figure 1 shows the relative quality loss $\delta$PSNR for frame $i = 1$ of the test sequence foreman vs. the percentage $\rho_i$ for different PSNR$_{i-1}$ values. The different values of PSNR$_{i-1}$ for the $(i-1)$-th frame were obtained varying the Quantization Parameter $QP_{i-1}$, which controls the quantization step value in the H.264/AVC coder (for more details on the relation between the quantization step value and $QP$ see [23]).

![Fig. 1. PSNR vs. $\rho_i$ for frame 1 of the sequences foreman and mobile (QCIF format). Frame 0 is coded in Intra mode using the quantization parameter $QP_{i-1}$.](image)

In case the average performance of Motion Estimation (ME) on a frame is stationary (i.e. the energy of the residual signal after ME does not change abruptly in successive frames), it is possible to assume that the probability mass function (pmf) of the transform coefficients approximately remains the same, and therefore, the relative PSNR decrement $\delta$PSNR$_i$ is approximately zero whenever $\rho_i = \rho_{i-1}$. Then, it is possible to approximate the PSNR value for the current frame with the equation

$$PSNR_i = \frac{1 - k_1 (\rho_i - \rho_{i-1})}{1 + k_2 (\rho_i - \rho_{i-1})} PSNR_{i-1}$ (5)

where $k_1$ depends on the characteristics of the coded sequence and can be estimated from the residual signal after the temporal prediction (more specifically in our approach we adopted a low-complexity linear regression to compute $k_1$ given the $\rho$ and the PSNR values available at the encoder). The parameter $k_2$ in eq. (5) is equal to $k_1 \rho_{i-1}$, but it can be appropriately tuned in order to model the distortion related to Motion Compensation whenever the performance of ME significantly varies for different frames (e.g. in presence of a scene change).

The following section will show how the percentage $\rho$ can be used to characterize the impact of packet losses on the visual quality.

IV. CHANNEL DISTORTION MODELLING AT TRANSPORT LAYER

In video coding the intensive use of temporal prediction permits obtaining high coding gains, but at the same time, it makes the coded packet stream more vulnerable to information losses. Since the lost information is needed to predict the following coded frames, the receiver has to estimate the missing data via error concealment algorithms [25]. Unfortunately, error concealment introduces an additional distortion that propagates in the reconstructed sequence until a complete refresh is performed. The amount of channel distortion depends on the accuracy of the error concealment algorithm, on the characteristics of the video sequence, and on the RD-optimization choices of the video source coder. Therefore, an effective cross-layer strategy must accurately estimate the expected channel distortion produced by the loss of a packet according to the characteristics of the coded data.

The energy of the residual signal after the temporal prediction is an effective parameter to characterize channel distortion since it reveals whether the displayed picture can be easily predicted or not with respect to the other frames. The joint source-channel optimization strategy reported in [26] approximates the energy value using the activity of the residual signal and increases the channel code rate as the activity level rises up. In this way the algorithm reduces the loss probability for those frames that are the most crucial for temporal prediction. The paper also proposes an alternative strategy that increases the channel code rate as the percentage $\rho$ decreases, since for a given $QP$ this implies an increment in the energy value under the assumption that the coded residual signal has a low-pass characteristic. In fact, the solution in [24] uses the parameter $\rho$ to model the source coding rate-distortion function, while it adopts the activity to model channel distortion. Instead, in our approach the parameter $\rho$ is employed to characterize the quality decrement due to packet losses. Figure 2 shows the relative quality decrement $\delta$PSNR$_{i,i}$ with respect to the previous frame produced by the loss of the $i$-th frame as a function of the percentage of zeros $\rho_i$. Experimental results show that there is a linear relation between $\delta$PSNR$_{i,i}$ and $\rho_i$ that can be approximated by the equation

$$\delta$PSNR$_{i,i} = k_{l,1} \rho_i + k_{l,2}$ (6)

where $k_{l,1}$ and $k_{l,2}$ are estimated at the encoder using a linear regression and are strictly related to the characteristics of the coded video sequence. It is possible to relate equation (6) to equation (5) assuming that in the error concealment process motion vectors can be estimated with sufficiently-accurate approximation while the prediction error is completely lost. Therefore, the loss of a single packet can be associated to coding the packet with null residual information (i.e. $\rho = 1$).

In case the assumption about the correctness of estimated
Motion Vectors (MV) is unattended, the parameters $k_{l,1}$ and $k_{l,2}$ of equation (6) can be adequately modified in order to suit the characteristics of the actual relative quality decrement $\delta_{\text{PSNR}_l,i}$. The final expected average PSNR value at the decoder can be expressed as

$$\hat{\text{PSNR}}_i = \left(1 - p_{\text{loss}}\right) \text{PSNR}_i + p_{\text{loss}} \left[1 - (k_{l,1} \rho + k_{l,2})\right] \text{PSNR}_{i-1}. \quad (7)$$

The average PSNR value $\text{PSNR}_i$ is computed as a weighted average of the PSNR in case no packet is lost ($\text{PSNR}_i$) and of the PSNR in case a packet with percentage $\rho$ of zeros is missing ($\left[1 - (k_{l,1} \rho + k_{l,2})\right] \text{PSNR}_{i-1}$). The weights are computed considering the loss probability $p_{\text{loss}}$, which can be approximated by the packet loss ratio.

In order to reduce the expected distortion on the sequence reconstructed at the decoder, it is possible to protect the transmitted information introducing some redundant packets in the stream. In our approach, we considered the 3GPP coding setting reported in [27], which sequentially includes RTP video packets into a coding matrix in a columnwise order and computes the additional redundant bytes applying a Reed Solomon code $RS(n, s)$ along the rows. The paper [26] presents some possible optimization strategies for the coding matrix. One of the proposed solutions linearly increases the channel coder rate according to the percentage of zeros in order to reduce the probability that packets carrying information associated to a low $\rho$ value get lost. Moreover, channel coder rate must take into consideration the position in the Group of Picture (GOP) of the video information that is included in each RTP packets. In case packets are related to a frame which occurs at the beginning of the GOP, the channel coder rate must increase since the distortion produced by their loss could propagate and affect all the following frames. In case packets are related to a frame placed at the end of the GOP, the propagation of the distortion produced by their loss is blocked by the following Intra frame. Therefore, the tuning of the channel coder rate $r_i$ for the current frame must take into consideration both criteria as follows

$$r_i = \frac{\bar{r} - w_1}{\sigma_\rho} (\rho_i - \bar{\rho}) + \left[ w_2 - w_2 \frac{(i \mod N_{\text{GOP}})}{N_{\text{GOP}}} \right]$$

$$= h_{r,i} + h_1 (\rho_i - \bar{\rho}) \quad (8)$$

where $\bar{r}$ is the average coder rate, $N_{\text{GOP}}$ is the number of frames in a GOP, and the variables

$$h_1 = \frac{w_1}{\sigma_\rho}$$

$$h_{r,i} = \bar{r} + w_2 - w_2 \frac{(i \mod N_{\text{GOP}})}{N_{\text{GOP}}} \quad (9)$$

are introduced here for the sake of simplification. The average percentage of zeros $\bar{\rho}$ is computed from coding results of previous frames coded with the same mode of the current frame (Intra, P, or B), $\sigma_\rho$ represents the standard deviation value, and constants $w_1$ and $w_2$ are found from a set of simulations. The channel coder rate $r_i$ is then converted into an integer number $c_i$ of channel code columns through the approximation $c_i = \lfloor s_i r_i + 0.5 \rfloor$, where $s_i$ is the number of source code columns.

The effectiveness of this approach will be discussed in Section VII-A, where simulation results for some coded sequences are reported. The next section will show how it is possible to apply the same policy in assigning different priority classes to packets.

V. PRIORITY CLASSIFICATION AT NETWORK AND MAC LEVEL

At network level, transmission protocols permit an efficient delivery of multimedia data thanks to congestion control algorithms and loss control algorithms that rule the retransmission and the discarding of packets [28]. In the same way, at MAC level different strategies have been designed in order to partition the stream of frames into different access categories for a shared transmission link [29]. Both approaches rely on a classification algorithm that associates each packet to a service class which transmits and handles the data with different priorities.

The previous section has shown how the percentage $\rho$ of zeros permits the detection of those packets that are more critical in the temporal prediction loop. Therefore, it is possible to adapt eq. (8) in order to compute the most appropriate packet priority level $\pi_i$ for the $i$-th frame, which can be expressed as

$$\pi_i = d_0 - d_1 \rho_i + d_2 N_{\text{IntraMB},i} \quad (10)$$

where $N_{\text{IntraMB},i}$ is the number of Intra macroblocks. Constants $d_0, d_1, d_2$ are greater than zero and were optimized according to the simulation results obtained from an extensive set of coded sequences. According to eq. (10), packet priority increases as the number of Intra MB and the percentage of non-null coefficients increase (since in this case the coded information results more difficult to estimate), while the priority is reduced whenever the percentage of zeros increases.

Then, packet priority is adapted to a specific service according to the characteristics of the network and of the MAC layer protocols. Assuming that there are $N_C$ different classes available, the priority parameter $\pi_i$ makes possible to assign a service class $z_i$ to the $i$-th packet according to the rule

$$z_i = j \text{ if and only if } \pi_i \in [\tau_j, \tau_{j+1}) \text{ with } j = 0, \ldots, N_C - 1 \quad (11)$$

where $\tau_j$ are fixed thresholds estimated from an extensive set of simulations.
Section VII will present the decoding results of two possible implementations.

VI. CROSS-LAYER OPTIMIZATION BASED ON ρ MODELLING

The strategies proposed in the previous sections rely on the common consideration that the percentage of zeros ρ is an effective and efficient indicator of the significance of each frame in the decoding process. At the same time the parameter ρ permits the adoption of simple rate-distortion models which, at a limited computational cost, characterize both the quality of the reconstructed sequence and the amount of information that has to be transmitted. It is possible to combine all these techniques in order to design an efficient joint optimization strategy.

According to equation (7), the expected PSNR value for the reconstructed i-th frame is

$$\text{PSNR}_i = (1 - p_{loss,i}) \text{PSNR}_i + p_{loss,i} [1 - (k_{1,1} \rho_i + k_{1,2})] \text{PSNR}_{i-1}$$

$$= (1 - p_{loss,i}) (1 + k_{2} - k_{1} \rho_i) \text{PSNR}_{i-1} + p_{loss,i} [1 - (k_{1,1} \rho_i + k_{1,2})] \text{PSNR}_{i-1}$$

which can be expanded considering the loss probability $p_{loss,i-1}$ of the $(i-1)$-th frame into

$$\text{PSNR}_i = (1 - p_{loss,i}) (1 - p_{loss,i-1}) \text{PSNR}_i + p_{loss,i} (1 - p_{loss,i-1}) \text{PSNR}_{i-1}$$

$$+ (1 - p_{loss,i}) p_{loss,i-1} \text{PSNR}_{i-1}$$

where $\text{PSNR}_i$ is the objective quality measure for the i-th frame in case no losses have occurred, and $\text{PSNR}_{i-1}$ is the PSNR value of the i-th frame estimated by the error concealment algorithm assuming the previous frame has correctly arrived. Note that the term related to the case when both frames are lost is omitted since we assume that the contribution of the distortion produced by the loss of both the i-th and $(i-1)$-th frame is negligible. In fact, its influence in the target function $\text{PSNR}_i$ is weighted by the probability $p_{loss,i} p_{loss,i-1}$ which is smaller than the probability values of the other terms. According to equations (6) and (5), it is possible to derive the relations

$$\text{PSNR}_i = (1 + k_{2} - k_{1} \rho_i) \text{PSNR}_i$$

$$\text{PSNR}_{i-1} = [1 - (k_{1,1} \rho_i + k_{1,2})] \text{PSNR}_{i-1}$$

$$\text{PSNR}_{i-1} = [1 - (k_{1,1} \rho_i + k_{1,2})] \text{PSNR}_{i-2}$$

Note that the term expressed by eq. (14c) is equal to the PSNR of the $(i-1)$-th frame after error concealment despite the amount of introduced distortion progressively decreases through the GOP. The distortion leakage in the prediction step (see [30]) is produced by spatial filtering which can be explicitly related to loop filtering or implicitly related to the side-effects of fractional-pel motion compensation and spatial prediction. The overall effect of distortion propagation in the GOP is ruled by a decreasing function as [11] reports, but the quality improvement after one frame is negligible with respect to the other terms. Therefore, we assume that the value $\text{PSNR}_{i-1}$ after loosing the $(i-1)$-th frame remains the same for the following i-th frame despite the fact that the i-th frame has been correctly received.

From these assumptions, equation (13) can be expressed as follows

$$\text{PSNR}_i = A_1 (1 - p_{loss,i}) (1 + k_{2} - k_{1} \rho_i) + A_1 p_{loss,i} [1 - (k_{1,1} \rho_i + k_{1,2})] + (1 - p_{loss,i}) A_2$$

where

$$A_1 = (1 - p_{loss,i-1}) \text{PSNR}_{i-1}$$

$$A_2 = p_{loss,i-1} \text{PSNR}_{i-1} [1 - (k_{1,1} \rho_i + k_{1,2})] \text{PSNR}_{i-1}$$

Including equations (15) and (3) in a per-frame optimization problem derived from eq. (1), we obtain

$$\max_{\rho_i, r_i \in [0, 1]} (S_1 + S_2 \rho_i) + (1 - p_{loss,i}) (S_3 + S_4 \rho_i)$$

such that

$$(1 + r_i)(m \rho_i + q) \leq T_i$$

where $T_i$ is the target number of bits assigned for the i-th frame, $r_i$ is the channel coder rate, and the variables $S_i$, $i = 1, 2, 3, 4$, are set as follows

$$S_1 = 1 - k_{1,2}$$

$$S_3 = A_1 (k_{2} + k_{1,2}) + A_2$$

$$S_4 = A_1 (k_{1,1} - k_{1})$$

Note also that the delay term of equation (1) is missing in equation (17) since it has been included in the loss probability assuming that the late arrival of a packet equals a loss (more details will be provided later in this section). In this way, the rate control routine assigns a target bit budget $T_i$ for the current frame, and the cross-layer optimization algorithm finds the best parameters setting that maximizes the expected PSNR value.

The probability of receiving correctly the i-th frame depends on the channel coder rate $r_i$ and the assigned priority $\pi_i$, which modifies the probability of loosing the current frame. More specifically, it is possible to approximate $p_{loss,i}$ through the linear relation

$$(1 - p_{loss,i}) \approx \sigma_1 \pi_i + \sigma_2$$

where the parameters $\sigma_1$ and $\sigma_2$ depends on the channel loss probability related to class $\pi_i$ (see Appendix A for more details).

Moreover, we want to constrain the channel coder redundancy according to the adaptation rule expressed in eq. (8), and therefore, we add an extra constraint in the optimization problem (17) and obtain

$$\max_{\rho_i, r_i \in [0, 1], i = 1, 2, 3, 4} (S_1 + S_2 \rho_i) + (\sigma_1 \pi_i + \sigma_2) (S_3 + S_4 \rho_i)$$

such that

$$(1 + r_i)(m \rho_i + q) \leq T_i$$

$$(h_{r,i + 1} (\rho_i - \overline{\rho}) - r_{i}) \leq 0$$

(20)

The parameters $m$ and $q$ are computed at the source encoder according to the coding results of the previous frame (see [8]). As for the parameters $k_{1,1}, k_{2}$ and $k_{1,1}, k_{1,2}$ related to distortion models in eq. (4) and (6), the optimization algorithm performs a linear regression considering the percentages of zeros, the PSNR values, and the energies of the residual signals.
for the previously coded frames. Further details about this estimation are reported in Appendix B. For every possible class of service \( z_i \), the optimal \( \rho_{\text{opt}, i} \) and \( r_{\text{opt}, i} \) are found solving the constrained maximization problem in eq. (20) (see Appendix B), and the corresponding target function is computed in order to evaluate the resulting final configuration \([\rho_{\text{opt}, i}, r_{\text{opt}, i}, z_{\text{opt}, i}]\) that maximizes the target function. In case the resulting parameters do not lie within the range of allowed values (specified by the constraints in eq. (20) with \( \rho_{\text{opt}, i}, r_{\text{opt}, i} \in [0, 1] \)), the parameter \( \rho_i \) is computed solving the equations

\[
\begin{align*}
(1 + r_i)(m \rho_i + q) &= T_i \\
\frac{1}{m} &= h_{T,1} - k_1 \rho_i
\end{align*}
\tag{21}
\]

where all the available bits are used to code the current frame. The overall procedure is summarized by the pseudocode in the Appendix B. Simulation results will show that this technique permits a significant improvement of the PSNR value obtained at the receiver.

As regards the computational complexity, the proposed approach proves to be a competitive solution to perform a cross-layer optimization of the different transmission parameters. Table I summarizes the overall amount of operations required by the CL optimization algorithm. The reported values also include the operations required by the rate control at macroblock level. Once again it is possible to notice that the proposed approach proves to be quite cheap in terms of computational complexity since the heaviest computation is performed at frame level and does not require a significant number of operations. This fact is mainly due to the adoption of linear models that permit solving the optimization problem in eq. (1) with a limited computational effort. Note also that the parameter \( \rho \) can be straightforwardly obtained from the H.264/AVC syntax and requires a small extra amount of computation.

VII. SIMULATION RESULTS

Different simulations were performed in order to evaluate the performance of each algorithm described in this paper. In our tests we adopted the JM60a implementation of the H.264/AVC codec, where the decoder has been enabled with the error concealment techniques reported in [31]. Sequences were coded using GOPs of 15 frames with structure IPP...P and with either fixed QP or fixed transmission bit rate (when using the joint cross-layer optimization algorithm described in Section VI). In order to evaluate the performance of the different optimizations, packet losses were simulated using both a simple Gilbert model (see [32]) and the NS2v2.29 simulator.

<table>
<thead>
<tr>
<th>Level</th>
<th>Sums</th>
<th>Mult/Div.</th>
<th>Shifts</th>
<th>Comp.</th>
<th>Fetch/Store</th>
</tr>
</thead>
<tbody>
<tr>
<td>GOP</td>
<td>11</td>
<td>7</td>
<td>0</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>Frame</td>
<td>87</td>
<td>144</td>
<td>8</td>
<td>64</td>
<td>106</td>
</tr>
<tr>
<td>MB</td>
<td>11</td>
<td>4</td>
<td>1</td>
<td>5</td>
<td>23</td>
</tr>
</tbody>
</table>

TABLE I
COMPUTATIONAL COMPLEXITY FOR THE CL ALGORITHM

<table>
<thead>
<tr>
<th>QP</th>
<th>Sequences</th>
<th>P_L</th>
<th>( \rho )-based</th>
<th>PSNR (dB)</th>
<th>FEC fixed</th>
<th>No FEC</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>foreman</td>
<td>0.01</td>
<td>45.21</td>
<td>44.22</td>
<td>44.06</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.05</td>
<td>43.04</td>
<td>39.09</td>
<td>36.81</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.10</td>
<td>39.78</td>
<td>36.89</td>
<td>32.58</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.15</td>
<td>38.78</td>
<td>31.42</td>
<td>29.95</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>foreman</td>
<td>0.01</td>
<td>37.61</td>
<td>36.93</td>
<td>36.08</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.05</td>
<td>36.70</td>
<td>34.19</td>
<td>32.43</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.10</td>
<td>34.74</td>
<td>30.68</td>
<td>29.19</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.15</td>
<td>32.69</td>
<td>30.29</td>
<td>27.95</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>news</td>
<td>0.01</td>
<td>38.95</td>
<td>38.00</td>
<td>36.37</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.05</td>
<td>37.89</td>
<td>35.03</td>
<td>32.60</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.10</td>
<td>36.29</td>
<td>31.27</td>
<td>29.49</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.15</td>
<td>34.71</td>
<td>30.59</td>
<td>27.21</td>
<td></td>
</tr>
</tbody>
</table>

TABLE II
COMPARISON BETWEEN \( \rho \)-BASED ADAPTIVE AND FIXED FEC METHOD

The following subsections will report the results obtained for different cross-layer settings and for the final joint bit allocation strategy.

A. Optimization of the FEC coder at transport layer

The first set of simulations was performed to test the effectiveness of the adaptation rule expressed in eq. (8). Table II reports the average PSNR values obtained corrupting the packet stream with 100 loss patterns generated from a two-states Gilbert model where

\[
P[\text{bad} \mid \text{good}] = \frac{1}{L_B} \quad P[\text{good} \mid \text{bad}] = \frac{P_L}{L_B (1 - P_L)} \tag{22}
\]

with average burst length \( L_B \) equal to 4 and varying the loss probability \( P_L \) as specified in [32]. Using a pseudo-random engine, for each packet we evaluate whether the state of the channel changes or remains the same (starting from the “good” state), and in case the resulting state is “bad” the packet is lost. The reported data compare the \( \rho \)-based adaptive and fixed FEC method presented in Section IV for the sequences foreman and news (coded at 30 frame/s and fixed QP=15, 25) using a target RS(13, 9) channel code). The constant values \( u_1 \) and \( u_2 \) in equations (8) and (9) were obtained empirically from an extensive set of simulations using different kinds of sequences with average channel code rate \( \tau \) = 0.1. More specifically, \( u_1 \) and \( u_2 \) are set to 0.0055 and 0.083 respectively, which proved to be an effective choice for different channel conditions too.

It is possible to notice that the average PSNR value increases up to 4 - 5 dB for the sequence foreman since the designed strategy is able to detect those frames that are the most significant in the decoding process and those that can be easily approximated from the previously-decoded ones. In this way it is possible to reduce the number of additional FEC packets for those frames that can be easily estimated by the error-concealment routine and to increase the number of FEC packets for those frames that crucially affect the distortion propagation in case they are lost. Note that here the coding matrix for both the FEC-fixed and the \( \rho \)-based approach is dimensioned in order to allow a maximum decoding delay equal to a frame constraining the channel coding rate to be equal to \((13 - 9)/9 = 0.44\).
The simulation results in Table II show that at medium quality the PSNR gain for the $\pi$-based adaptive algorithm is higher for sequences characterized by a greater amount of motion since the error concealment algorithms provide coarse estimates of the lost frames and the channel distortion is more evident. With $QP = 25$ the PSNR gain for the sequence foreman is about 4 dB while it is approximately 1 dB for the sequence news. In the latter case, the video sequence presents a low amount of motion, and therefore, the performance of the error concealment is quite good whenever the FEC decoder can not recover the lost information. It is also possible to notice that whenever the loss probability $P_L$ is not accurately estimated, the proposed optimization strategy proves to be adequately robust since those frames that are crucial in the decoding process result sufficiently protected. As an evidence for this, when $P_L = 0.15$ the failure probability for the $RS(13, 9)$ FEC coder in recovering the lost packets belonging to a frame rises up to 3%. Nevertheless, the $\rho$-based approach still performs well while the performance of the non-adaptive scheme collapses. Figure 3 reports the average PSNR vs. different average burst lengths $L_B$ with $P_L = 0.1$ for the sequence foreman (coded with $QP = 15$). It is possible to notice that the $\rho$-based adaptive approach improves the quality of the reconstructed sequence for all the configurations since an unequal allocation of redundancy permits recovering the most significant lost information even in case of bursts of losses.

B. Performance of packet classification at network level in a DiffServ environment

In a second set of simulations, we evaluated the performance of the labelling algorithm reported in Section V. Tests were performed on a simulated DiffServ (DS) wired network [28], which is depicted in Figure 4 and was implemented using the NSv2.29 simulator. We simulated different congestion levels reducing the available bandwidth in the link between edge nodes $e_1, e_2$ and core node $c$. The node $e_1$ provides network access to three nodes $s_1, s_2, s_3$, where $s_1$ transmits the video sequence that we want to monitor and $s_2, s_3$ transmit Constant Bit Rate (CBR) traffic at the same bit rate. In our tests we adopted the packet labelling convention used in [33], and we compared the proposed solution, which is called $\pi$-based since it operates according to the packet priority $\pi$ of eq. (10), with a more traditional approach, the Two-Rates-

Three-Colors-Marking (TRTCM) [34]. The labelling algorithm for video packets produced by $s_1$ assigns a service class to the $i$-th packet according to the variable $\pi_i$ specified in eq. (10) and the thresholds $\tau_j$ found from an extensive training set of simulations (see equation (11)). Table III reports the simulation results for the two algorithms obtained coding the sequences foreman and news at 30 frame/s with fixed $QP$. The table reports the average PSNR value for the luminance component (over 10 trials), together with the percentage of lost packets $P_L$. The percentage $P_L$ varies according to the congestion level which is obtained by changing the overall available transmission rate $R_w$ of the links between the core and the edge nodes. In Table III we refer to the different congestion levels using the ratio between the bit rate $R_0$ of the traffic generated by the three nodes and the parameter $R_w$. The proposed strategy proves to be quite effective whenever the average bandwidth reduction inhibits a correct transmission of all the packets. In this case the packet labels assigned by the $\pi$-based approach are able to drive effectively the DS policer in the network by discriminating those packets that are more significant in the decoding process.

The simulation data reported in Table III show that the proposed algorithm could increase the PSNR value up to 1 dB for the foreman sequence with respect to the traditional TRTCM approach. It is also worth underlying that the percentage of lost RTP packets could result higher for the $\pi$-based algorithm with respect to that for the TRTCM protocol since the $\pi$-based approach increases the number of packets assigned to the lowest priority service classes. In this way

![Image](image-url)
the proposed algorithm reduces the probability of congestion for the most important packets by discarding a higher number of packets which can be easily estimated. As it was noticed for the $\rho$-based tuning of the FEC channel coder, in case of sequences with a low amount of motion (like news) the quality improvement is less significant with respect to the traditional TRTCM at low $R_w$ values. The error-concealment performs quite well without the help of the FEC decoder, and due to the small variance of the parameter $\rho$ the $\pi$-based algorithm assigns most of the packets to the same class.

C. Packet priority differentiation at MAC level for an IEEE802.11e-like network

The priority assignment strategy presented in the previous subsection was employed to rule the channel access for different packets in the IEEE802.11 environment depicted in Fig. 5. In our tests we considered an IEEE802.11e-like network characterized by $2 + N_I$ wireless nodes communicating with a wired node through a base station (see Figure 5), where $N_I$ is the number of interfering nodes with the video transmission. The first node downloads a video sequence from the access point, while the second node (which implements the CL classification strategy) transmits a QCIF video sequence. The other nodes generate CBR traffic characterized by equally-timed packets with the same size. The bit rate of CBR traffic (i.e. the length of the packets), together with the number $N_I$, is varied in order to test the proposed approach under different congestion levels. The IEEE802.11e-like nature is related to the fact that the MAC layer unit includes frames coming from UDP packets into 4 possible queues that are characterized by different priorities (see [29]), i.e. each packet gains access to the channels with different probabilities, while the other parameters that characterize each single queue are the same. Whenever one node gets access to the medium, it starts polling all the queues in priority order. Whenever a packet is popped up, its timestamp is checked in order to compute whether it is obsolete or not. In case the expected transmission delay is too high (i.e. larger than $D_{\text{max}}$), the packet is discarded and it is considered lost. So, we avoid transmitting obsolete packets that are going to be discarded anyway, thus improving the final throughput. Since in our work we address interactive communication, we set $D_{\text{max}} = 0.2$ sec. The medium access is ruled by the Distributed Coordination Function (DCF) [29], which relies on a CSMA/CA algorithm to avoid collisions. The data rate is 11 Mbit/s and lost frames are retransmitted up to 4 possible trials provided that the packet has not become obsolete and it is no more useful for the decoder. In this way we reduce the extra overhead of retransmitted packets in the network and limit the transmission delay. Table IV reports a summary of the parameters setting for the considered wireless network.

In our tests we compare a Single Queue (SQ) approach, which buffers all the packets in a single transmission queue, with the $\pi$-based Multi Queue (MQ) algorithm, where each packet is assigned to a different transmission queue corresponding to a different access probability (i.e. a different QoS level). The assignment is performed comparing the priority level $\pi_i$ of the $i$-th packet with a fixed set of thresholds $\tau_j$, which are estimated from a set of simulation data. Table V reports the average PSNR values for the SQ and the $\pi$-based MQ approaches relative to the coded sequences foreman, news, and silent (coded at 30 frames/s and fixed QP) over 40 trials varying the transmission rate of an interfering node. The value $R_I/R_b$ reports the ratio between the transmission bit rate of the interfering node ($R_I$) and the bit rate of the video source ($R_b$).

Once again it is possible to notice that the algorithm affected by the lowest percentage of lost packets does not provide the higher PSNR value at the decoder. The $\pi$-based approach limits the congestion probability reducing the transmission priority of the least significant packets with respect to the Single Queue (SQ) strategy, where no differentiation in the transmission priority of the packets is performed. This choice proves to be quite effective in case the bit rate of the CBR source becomes heavy and the network is not able to support all the sources at the same time. Simulation results shows that it is possible to gain up to 4 dB for the sequence foreman with respect to the SQ approach whenever the bandwidth contention starts growing significant. At lower packet loss percentages,
the algorithm proves to be effective since the PSNR value of
the \( \pi \)-based MQ solution is higher or equal to the SQ approach
for different sequences (Table V reports the simulation results
for the test sequences foreman and silent).

D. Comparison between the joint cross-layer strategy and an
independent-layer optimization

In the last set of simulations we tested the optimization
strategy proposed in Section VI with the same network and
simulation setting adopted for the \( \pi \)-based MQ algorithm (see
the previous subsection). The target number of bits for each
frame was assigned using the same strategy that was presented
in [8], [26], but the computation of the service class, \( \rho_i \) and \( r_i \)
was made using the equations (30) reported in the Appendix B.
Figure 6 and Figure 7 report the simulation results obtained
using the proposed cross-layer strategy (CL) compared with
the algorithm proves to be effective since the PSNR value of
the \( \pi \)-based MQ solution is higher or equal to the SQ approach
for different sequences (Table V reports the simulation results
for the test sequences foreman and silent).

D. Comparison between the joint cross-layer strategy and an
independent-layer optimization

In the last set of simulations we tested the optimization
strategy proposed in Section VI with the same network and
simulation setting adopted for the \( \pi \)-based MQ algorithm (see
the previous subsection). The target number of bits for each
frame was assigned using the same strategy that was presented
in [8], [26], but the computation of the service class, \( \rho_i \) and \( r_i \)
was made using the equations (30) reported in the Appendix B.
Figure 6 and Figure 7 report the simulation results obtained
using the proposed cross-layer strategy (CL) compared with
the cross-layer architecture. The performance of the cross-layer architecture
presents the same advantages underlined in Sections VII-A,
VII-B and VII-C. The proposed strategy improves the PSNR
value with respect to its counterpart where independent layers
(IL) are optimized separately. More specifically, the PSNR
gain is about 2 dB for the sequence foreman with \( N_I = 1 \)
and \( P_{lp} = 0.06 \). Note that a similar gain can be obtained
with \( N_I = 3 \). In case of sequences with a lower amount of
motion (like suzie), the gain of the CL over the IL approach
decreases since the performances of the previous algorithms,
which are encapsulated by the CL approach, decrease as well.

VIII. CONCLUSIONS

This paper faces the problem of finding an efficient low-cost
cross-layer algorithm that tunes the parameters of each layer
in the protocol stack in order to maximize the video quality
at the decoder. According to the percentage \( \rho \) of null DCT
coefficients associated to the video information carried by a
packet, it is possible to tune both the amount of additional
redundant information, which is included in the packet stream
to protect packets against losses, and the service class, which
is assigned to each packet. At the same time, the percentage \( \rho \)
is used to model the quality parameter that characterizes the
reconstructed sequence in order to find a joint optimization
strategy that requires a limited amount of computation. The
proposed solution permits the discrimination between parts
of information that are essential in the decoding process and
parts that can be easily estimated in case they are lost, and it
applies a QoS differentiation in order to maximize the expected
PSNR of the reconstructed sequence. Simulation results show
that this cross-layer strategy can increase the PSNR value of
the reconstructed sequence up to 2 dB. Future work will be
concerned with the application of the proposed solution to
Multiple Description coding schemes.

ACKNOWLEDGEMENT

The authors would like to acknowledge the Ph.D. student
Simone Merlin of the University of Padova, Italy, for his support
in the definition of network scenarios.

APPENDIX A

APPROXIMATION OF THE PACKET LOSS PROBABILITY

Given the optimization problem expressed in eq. (17),
we have derived a closed form solution approximating the
probability of receiving correctly the \( i \)-th frame \( (1 - p_{loss,i}) \),
which depends on the channel coder rate \( r_i \) and the assigned
priority \( \pi_i \).

Given the probability \( p_{col} \) that a column in the channel coder
matrix is lost, the probability of receiving correctly the \( i \)-th
frame equals the probability of recovering the lost columns,
which for a matrix with \( s \) source columns and \( c = n - s \)
channel code columns is equal to

\[
(1 - p_{loss,i}) = \sum_{k=0}^{c} \binom{n}{k} p_{col}^k (1 - p_{col})^{n-k}, \quad (23)
\]

and can be approximated by a sigmoid function

\[
(1 - p_{loss,i}) \simeq C + \frac{1}{1 + \exp \frac{-\pi}{\sigma_c}}, \quad (24)
\]

where \( \pi \) and \( \sigma_c \) are respectively the mean and the standard
deviation of the random variable expressed in eq. (23) while
\( C \) is a constant. Their values can be estimated as follows

\[
\begin{align*}
\pi &= n \ p_{col} + 0.5 \\
\sigma_c &= n \ p_{col} (1 - p_{col}) \\
C &= (1 - p_{loss,i})^n - \frac{1}{1 + \exp [1/(1 - p_{col})]}.
\end{align*}
\]
Unfortunately the adoption of a sigmoid function in the optimization increases the computational complexity of the solving algorithm for the problem expressed in eq. (17), and we substituted the sigmoid with a piecewise-linear function since it provides a sufficiently-accurate approximation for the range of $r_i$ values we are considering. As a consequence, it is possible to approximate the probability $\left(1 - p_{\text{loss},i}\right)$ of eq. (24) as expressed in equation (19), where $\sigma_i$, $i = 1, 2$, are computed from $\tilde{\sigma}$ and $\sigma_c$. Note that the probability of loosing one column depends on the service class assigned to the packets contained in the row. Each service class $z$ is characterized by a different waiting time $t_j$ before transmitting the current $j$-th RTP packet. Therefore, the probability of losing the $j$-th packet equals

$$p_{\text{RTP},j} = P[t_j > D_{\text{max}}] + (1 - P[t_j > D_{\text{max}}]) \cdot p_{\text{chan}}$$

(26)

where $p_{\text{chan}}$ denotes the loss probability related to the transmission channel (depending on the medium access probability, noise level, etc...). Assuming that the waiting time is exponentially distributed, the probability of losing the packet because of its obsolescence (i.e. it arrives too late) can be expressed as follows

$$P[t_j > D_{\text{max}}] = e^{-D_{\text{max}}/\overline{D}_z}$$

(27)

where $\overline{D}_z$ is the average waiting time for packets assigned to the $z$-th class. The probability $p_{\text{RTP},j}$ is then mapped into $p_{\text{col}}$ through a linear relation (no extra indices are added to $p_{\text{col}}$ for the sake of simplicity).

**APPENDIX B**

**DERIVATION OF THE CROSS-LAYER OPTIMIZATION ALGORITHM**

Given the optimization problem of equation (20), the corresponding Lagrangian function can be written as

$$L(\rho_i, r_i, z, \lambda_1, \lambda_2) = (S_1 + S_2 \rho_i) + (\sigma_1 r_i + \sigma_2)(S_3 + S_4 \rho_i) + \lambda_1 \left[ T_i - (1 + r_i)(m \rho_i + q) \right] + \lambda_2 \left[ r_i - h_{r,i} - h_1 (\rho_i - \overline{\rho}) \right]$$

and can be maximized solving the equation system

$$\begin{align*}
\frac{\partial L}{\partial \rho_i} &= 0 \\
\frac{\partial L}{\partial r_i} &= 0 \\
\frac{\partial L}{\partial z_i} &= 0 \\
\frac{\partial L}{\partial \lambda_1} &= 0 \\
\frac{\partial L}{\partial \lambda_2} &= 0.
\end{align*}$$

(29)

From eq. (29) it is possible to derive a closed form solution for a given class $z_i$, which can be expressed by the following equations

$$\lambda_{\text{opt},1} = \frac{b (1 + h_{r,i} - k_1 \overline{\rho}) - c - k_1 \sigma_1 S_3}{m(1 + h_{r,i} - k_1 \overline{\rho}) - k_r q}$$

(30a)

$$\lambda_{\text{opt},2} = \frac{0.5 h_1 a - mc}{-2k_3 m} \pm \sqrt{(0.5ah_1 - mc)^2 + 4h_1 m [0.5ac + (b - \lambda_{\text{opt},1} m)^2T_i]}$$

(30b)

$$\rho_{\text{opt},i} = \frac{-\sigma_1 S_3 + \lambda_{\text{opt},1} q - \lambda_{\text{opt},2}}{\sigma_1 S_4 - \lambda_{\text{opt},1} m}$$

(30c)

$$r_{\text{opt},i} = h_{r,i} + h_1 (\rho_{\text{opt},i} - \overline{\rho})$$

(30d)

where

$$a = 2\sigma_1 (S_3 m - S_4 q)$$

$$b = \sigma_1 S_4$$

$$c = S_4 (\sigma_1 - \sigma_2) - S_2.$$
the equation
\[ k_1 = \frac{3 \varphi_{\text{PSNR}} - \varphi_p \varphi_{\text{PSNR}}}{3\varphi_{\text{p,p}} - \varphi_p^2} \quad k_2 = \frac{\varphi_{\text{PSNR}} \varphi_{\text{p,p}} - \varphi_p \varphi_{\text{PSNR}}}{3 \varphi_{\text{p,p}} - \varphi_p^2} \]

with
\[ \varphi_p = \sum_{w=0}^{2} \rho_{i-w} \quad \varphi_{\text{PSNR}} = \sum_{w=0}^{2} \delta\text{PSNR}_{i-w} \]
\[ \varphi_{\text{p,p}} = \sum_{w=0}^{2} \rho_{i-w}^2 \varphi_{\text{PSNR}} = \sum_{w=0}^{2} \delta\text{PSNR}_{i-w} \rho_{i-w} \]

(32)

while \( k_{1,1}, k_{1,2} \) are computed in the same way made exception for the fact that \( \delta\text{PSNR} \) is evaluated considering the concealed image.

The overall cross-layer optimization algorithm can be summarized by the following steps.

1. Let PSNR\(_{\text{best,i}}\), \( r_{\text{best,i}} \), and \( c_{\text{best,i}} \) respectively the percentage of zeros, channel code rate, and QoS class for the current frame.

2. Compute the target bit rate \( T_i \) for the current \( i \)-th frame using a proportional rate control equation (see [26] for more details)

\[ T_i = \frac{K_t G}{K_I N_I + K_P N_P + K_B N_B} \]

where \( G \) is the number of bits available for the current GOP and \( t \) is the coding type for the \( i \)-th frame. The parameters \( N_t, t = I, P, B \), count the number of \( t \)-type frames in the current GOP that remain to be coded, while \( K_I \) is the complexity ratio between \( t \)-type and \( B \)-type frames with \( K_B = 1 \) (see [8] for more details).

3. Compute \( k_1, k_2, k_{1,1}, k_{1,2} \) from previously coded frames (with a regression). Update also \( m \) and \( q \) as previously described in this appendix.

4. for \( z \) in available QoS classes do

5. Compute \( \sigma \) and \( \sigma_c \) related to the current QoS class (see eq. (25)).

6. Compute \( S_1, S_2, S_3, S_4, \sigma_1 \) and \( \sigma_2 \) from previous parameters (see eq. (18) and eq. (19)).

7. Compute the optimal parameter \( \rho_{\text{opt,i}} \) and \( r_{\text{opt,i}} \) via the equations (30c) and (30d).

8. Compute the associated PSNR value PSNR\(_z\) using the target function of the problem in eq. (20).

9. if PSNR\(_z\) > PSNR\(_{\text{best,i}}\) then

10. PSNR\(_{\text{best,i}}\) ← PSNR\(_z\)

11. \( z_{\text{best,i}} = z \)

12. \( r_{\text{best,i}} = r_{\text{opt,i}} \)

13. end if

14. end for

15. Code the current frame.

16. Update previously-used parameters.

REFERENCES


