View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Universidad Carlos III de Madrid e-Archivo

This is a postprint version of the following published document:

González-de-Suso, José Luis, Martínez-Enríquez, Eduardo, Díaz-de-María, Fernando. (2017). Adaptive Lagrange Multiplier Estimation Algorithm in HEVC. *Signal Processing: Image Communication*, v. 56, pp.: 40-51.

DOI: https://doi.org/10.1016/j.image.2017.04.010

© 2017 Elsevier Ltd. All rights reserved.

This work is licensed under a Creative Commons Attribution-NonCommercialNoDerivatives 4.0 International License.

Elsevier Editorial System(tm) for Signal Processing: Image Communication
Manuscript Draft

Manuscript Number: image4467

Title: Adaptive Lagrange Multiplier Estimation Algorithm in HEVC

Article Type: Full Length Article (FLA)

Keywords: HEVC; motion estimation; rate-distortion optimization; source coding; video coding.

Corresponding Author: Dr. José Luis González-de-Suso, Ph.D.

Corresponding Author's Institution:

First Author: José Luis González-de-Suso, Ph.D.

Order of Authors: José Luis González-de-Suso, Ph.D.; Eduardo Martínez-Enríquez, Ph.D.; Fernando Díaz-de-María, Ph.D.

Abstract: The latest High Efficiency Video Coding (HEVC) standard relies on a large number of coding tools from which the encoder should choose for every coding unit. This optimization process is based on the minimization of a Lagrangian cost function that evaluates the distortion produced and the bit-rate needed to encode each coding unit. The value of the Lagrangian parameter $\lambda$, which balances the weight of the rate and distortion terms, is related to the quantization parameter through a model that has been implemented in the HEVC reference software. Nevertheless, in this paper we show that this model can be refined, especially for static background sequences, so that the coding performance of HEVC can be improved by adaptively modifying the relation between $\lambda$ and the quantization parameter.

Specifically, the proposed method (i) determines whether the background of a sequence is static or not by means of a simple classifier; and (ii) when static, it evaluates an exponential regression function to estimate a proper value of the $\lambda$ parameter. In so doing, the proposed method becomes content-aware, being able to dynamically act on the $\lambda$ parameter.

Experiments conducted over a large set of static and dynamic background video sequences prove that the proposed method achieves an average bit-rate saving of $-6.72\%$ ($-11.07\%$ for static background video sequences) compared with the reference HM16.0 software, notably outperforming the results of a state-of-the-art method.

Manuscript

Adaptive Lagrange Multiplier Estimation Algorithm in HEVC[☆]

José Luis González-de-Suso[a], Eduardo Martínez-Enríquez[b], Fernando Díaz-de-María[a]

[a] *Signal Theory and Communications Department, Carlos III University, Avenida de la Universidad 30, 28911, Leganés, Madrid, Spain*
[b] *Instituto de Óptica "Daza de Valdés", Consejo Superior de Investigaciones Científicas, 28006, Madrid, Spain*

Abstract

The latest High Efficiency Video Coding (HEVC) standard relies on a large number of coding tools from which the encoder should choose for every coding unit. This optimization process is based on the minimization of a Lagrangian cost function that evaluates the distortion produced and the bit-rate needed to encode each coding unit. The value of the Lagrangian parameter λ, which balances the weight of the rate and distortion terms, is related to the quantization parameter through a model that has been implemented in the HEVC reference software. Nevertheless, in this paper we show that this model can be refined, especially for static background sequences, so that the coding performance of HEVC can be improved by adaptively modifying the relation between λ and the quantization parameter.

Specifically, the proposed method (i) determines whether the background of a sequence is static or not by means of a simple classifier; and (ii) when static, it evaluates an exponential regression function to estimate a proper value of the λ parameter. In so doing, the proposed method becomes content-aware, being able to dynamically act on the λ parameter.

Experiments conducted over a large set of static and dynamic background

[☆]This work has been partially supported by the National Grant TEC2014-53390-P of the Spanish Ministry of Economy and Competitiveness.

Email addresses: `jlmolinero@tsc.uc3m.es` (José Luis González-de-Suso), `emenriquez@tsc.uc3m.es` (Eduardo Martínez-Enríquez), `fernando.diaz@uc3m.es` (Fernando Díaz-de-María)

URL: `http://tsc.uc3m.es/~jlmolinero` (José Luis González-de-Suso)

video sequences prove that the proposed method achieves an average bit-rate saving of -6.72% (-11.07% for static background video sequences) compared with the reference HM16.0 software, notably outperforming the results of a state-of-the-art method.

Keywords:
HEVC, motion estimation, rate-distortion optimization, source coding, video coding.

1. Introduction

The most recent video coding standard developed by the Joint Collaborative Team on Video Coding is the High Efficiency Video Coding (HEVC) standard [1, 2]. The current popularity of high definition and beyond-high definition video signals, video-on-demand services, stereo and multiview capture and display and video streaming over the internet and mobile networks have motivated the need for video coding standards with higher coding efficiency. These needs have been addressed by the HEVC standard, which is a block-based hybrid coding technique that achieves notable bit-rate savings with respect to the previous H.264/AVC standard [3] by relying on the ability to evaluate a large number of different coding tools to dynamically adapt to any video content. In the HEVC standard, this set of tools includes different coding block (CB) sizes, prediction block (PB) sizes, transform block sizes (TB), intra prediction modes, reference (Ref) frames and motion vectors (MV), quantization step (QP) sizes and Sample Adaptive Offset (SAO).

Therefore, it becomes necessary for the video encoder to choose among these coding options from a rate-distortion ($R-D$) perspective in order to reach a suitable coding solution for every coding unit. This task is performed through a so-called rate-distortion optimization (RDO) procedure, which aims to minimize a distortion measure while satisfying a rate constraint, i.e.:

$$\min_{\boldsymbol{\theta}} \{D(\boldsymbol{\theta})\} \text{ subject to } R(\boldsymbol{\theta}) \leq R_c, \tag{1}$$

where $\boldsymbol{\theta}$ represents a certain combination of coding options (CB, PB, Ref, MV, etc.); $D(\boldsymbol{\theta})$ is the distortion associated with this combination; $R(\boldsymbol{\theta})$ is the number of bits generated by the encoder with this combination including the bits used for headers, side-information and the residual transform coefficients; and R_c is the maximum rate allowed.

For practical reasons, this constrained problem is turned into an unconstrained one by using the Lagrange formulation [4]:

$$\min_{\boldsymbol{\theta}} \{J\} \text{ with } J(\boldsymbol{\theta}) = D(\boldsymbol{\theta}) + \lambda R(\boldsymbol{\theta}), \tag{2}$$

where $J(\boldsymbol{\theta})$ is a cost function that considers both terms $D(\boldsymbol{\theta})$ and $R(\boldsymbol{\theta})$, and the corresponding tradeoff between them through the Lagrange multiplier λ. For a specific value of λ, the optimal solution for the unconstrained problem (2), $\boldsymbol{\theta}^*(\lambda)$, turns out to be the optimal solution to the original problem (1) for $R_c = R(\boldsymbol{\theta}^*)$.

According to this formulation of the optimization problem, an HEVC coder should evaluate all the possible combinations of coding options ($\boldsymbol{\theta}$) to choose the best one. Although this problem has been solved through dynamic programming [5], in practice it becomes unfeasible because the computational complexity grows exponentially with the number of coding units. Therefore, some assumptions have been made in order to reduce its complexity.

First, decisions regarding coding a specific coding tree unit (CTU) are considered independent of decisions concerning previously coded CTUs. This hypothesis does actually not hold since choosing a specific set of coding options $\boldsymbol{\theta}^*$ for a CTU obviously affects the coding process of the subsequent ones. Nevertheless, this assumption notably contributes to make this process feasible. Second, decisions made at different levels (PB size, TB size, etc.) are also considered independently. And third, in order to avoid a simultaneous optimization of λ and QP, a relationship between these two parameters has been obtained (either theoretically or experimentally). For instance, in the case of the HEVC reference software HM16.0 [6] an experimental relationship was derived through a procedure similar to the one described in [7]; in particular:

$$\lambda = \alpha \cdot W_k \cdot 2^{(QP-12)/3}, \tag{3}$$

where α depends on the frame coding type (intra (I) or inter (P or B)) and the reference level of the frame, and W_k depends on the encoder configuration. This relationship is further supported by a theoretical derivation that assumes a Sum of Squared Differences (SSD)-based model for D and a high-rate approximation-based model for R [7]. Thus, given a QP value (provided by a rate-control algorithm in order to meet a certain rate constraint R_c), the Lagrange multiplier can be computed by using (3) and then the optimal solution for $\boldsymbol{\theta}$ can be obtained by minimizing $J(\boldsymbol{\theta})$ in (2).

As an alternative to this $\lambda(QP)$ model, other $R-D$ models have been proposed in the literature. For instance, Schuster et al. [8] modeled D as the maximum distortion value, instead of considering the SSD, under the same optimization framework. Some others are based on modeling D and R as parametric distributions. Li et al. [9] proposed a relationship between R and λ based on a hyperbolic function which also depends on the QP value. Si et al. [10] modeled the non-zero coefficients of the residue as a Laplacian distribution, yielding a relationship between λ, QP, and the distribution parameter. Li et al. [11] also proposed a Laplacian distribution for modeling the non-zero coefficients, leading to an elegant model to derive λ. Finally, Biatek et al. [12] modeled D and R based on the percentage of non-zero coefficients.

Other works have found improvements in coding efficiency from other points of view. For instance, Rehman et al. [13] proposed a D model based on perceptual measures with the purpose of improving the subjective coding performance. Deng et al. [14] provided models that considered the specific content of a video sequence. Xiong et al. [15] proposed a novel so-called motion compensation $R-D$ cost, which is exponentially related to the cost function in (2) to alleviate the number of cost functions computation. Liu et al. [16] proposed D and R-allocation models to perform $R-D$ optimization under the emerging compressed sensing technology. Z. Liu et al. [17] proposed an adaptive λ parameter according to a $R-D$ model that accounts for the correlation between residues belonging to successive frames in H.264/AVC (also applicable to HEVC) and Zeng et al. [18] proposed a modification of λ based on perceptual characteristics of the video sequence, making it adaptive on a CTU basis.

Furthermore, a revision of the literature suggests that the $\lambda(QP)$ relationship (3) used in the HM16.0 reference software could be improved for some types of video sequences. In this sense, Zhao et al. [19] proposed a λ modification based on the percentage of static background in the image for surveillance video sequences. Specifically, they classify each CTU into static background percentage bins and then, they find a relationship between this percentage of static background and the optimal λ parameter, which is parametrized specifically for each video sequence in a training stage carried out at the beginning of the encoding process. This proposal yielded interesting results; however, although this approach performed well for static and continuous video contents as those coming from video surveillance sequences, it did not work well for general varying-content video sequences, as the parameter training stage for the λ model is performed only once at the beginning of the encoding process. Finally, Zhang et al. [20] proposed a different approach to improve the coding performance of video sequences with static background. They proposed the use of a so-called G-reference frame that intends to model the background and that is used as a long-term reference. However, again, this method is specifically designed for videoconference and surveillance videos.

Thus, the aim of this paper is to propose a method for the optimization of λ that addresses the inefficiencies of the reference model on static background video sequences and continues to be effective in a general scenario (i.e., static and dynamic video content). To this end, an in-depth analysis of the behavior of the $R-D$ optimization process in an HEVC encoder is conducted for video sequences exhibiting both static and dynamic backgrounds and, as a result of this analysis, an adaptive method is proposed which yields significant savings in bit-rate (-6.72% compared with the reference software) and encoder complexity (-6.76% of encoding time savings compared with the reference software). Our proposal is very effective for those sequences in which the $\lambda(QP)$ relationship (3) is far from being optimal, and transparent for those cases in which (3) is actually optimal.

The remainder of this paper is organized as follows. In Section 2 we discuss the motivation of the work by analyzing the performance of the HEVC encoder for static and dynamic background video sequences. Then, in Section 3 we describe in detail the proposed method. In Section 4 we describe the experiments conducted to assess the proposed approach and discuss the achieved results. Finally, in Section 5 we draw conclusions and outline future lines of research.

2. Motivation

To motivate our work, we first analyzed the efficacy of the relation between λ and QP proposed for HEVC looking for leads which allowed us to come up with an improved $\lambda(QP)$ relation. Specifically, to test the baseline $\lambda(QP)$ relation in (3), we relied on a parametric re-definition of it (which we previously used in H.264/AVC [21] for similar purposes):

$$\lambda = F \cdot \alpha \cdot W_t \cdot 2^{(QP-12)/3}, \tag{4}$$

where F is the parameter which allows us to dynamically adapt the relation between λ and QP. Hereafter, we will refer to F as the F *multiplier*.

Experiments were carried out over a set of 10 CIF and HD video sequences for a *low-delay-P* profile using several values of F. A *low-delay-P* profile is suitable for static background sequences, as motion estimation is performed based on previous reference frames (as shown in Fig. 1) [20]. In order to draw reliable conclusions we have created *toy-examples* of short video segments of only 20 frames, so that they can be considered stationary (i.e., 20 frames of purely static or dynamic background). Moreover, a balanced number of static and dynamic background sequences has been chosen[1].

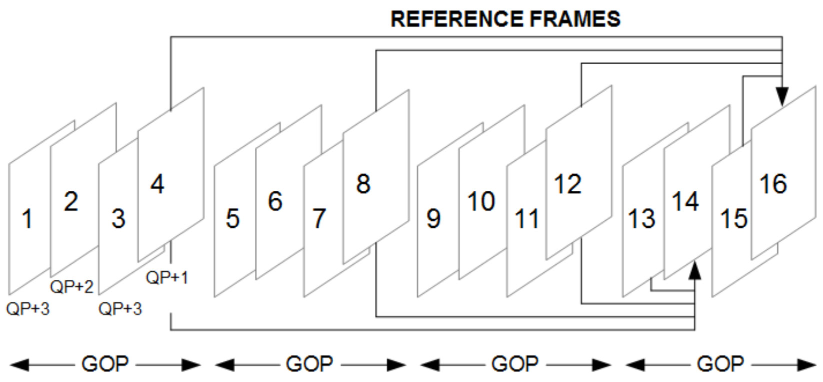


Figure 1: Group of pictures (GOP) structure for prediction under a *low-delay-P* profile. References for frames 14 and 16 are shown.

The encoder configuration used for these experiments is shown in Table 1. The QP Cascading parameter refers to the frame-to-frame QP adaptation illustrated in Fig. 1 (see frames #1 to #4); IP stands for Intra Period, which is the number of frames between Intra frames; *F range* establishes the possible range of F values; and *F step* determines the step forward used to cover that range. To facilitate the study of the $\lambda(QP)$ relation, the QP Cascading scheme was switched off, i.e., all the experiments were conducted at constant QP, and the IP was set to -1, which means that only the first frame is coded as Intra.

The Bjöntegaard Differences BD-Rate (BDR (%)) and BD-YPSNR (BDP-SNR_Y (dB)) (as defined in [22] and calculated following the procedure in [23]) were used for assessing the coding performance in terms of bit-rate savings and quality improvement on the luma component, while the time

[1]Hereafter, the terms *static* and *dynamic* will be used referring to static background video sequences and dynamic background sequences, respectively.

Table 1: Encoder configuration

Parameter	**Value**
#Frames	20
QP	22, 27, 32, 37
Profile	Low-delay-P
QP Cascading	Off
IP	-1
F Range	[0.2, 9.0]
F Step	0.4

increment ΔTI (%) was used to evaluate the computational cost:

$$\Delta TI = \frac{T_{method} - T_{HM16.0}}{T_{HM16.0}} \cdot 100, \tag{5}$$

where T_{method} is the encoding time associated with the method under evaluation and $T_{HM16.0}$ is the encoding time of the reference encoder.

2.1. Influence of the Lagrange multiplier on coding performance

Results obtained for a representative subset of the evaluated *F multipliers* in terms of BDR (%) and BDPSNR_Y (dB) are shown in Table 2 (in fact, a wider range of F values has been tested, but for brevity reasons only the more relevant subset will be analyzed in this section). As can be observed, for some video sequences the coding performance improves with larger values of F, achieving bit-rate savings of up to -17.21% (or BDPSNR_Y increments of 0.77 dBs) for *Snow Mountain*. Specifically, we observe that the coding performance improvement happens for those video sequences with static background. For instance, *Akiyo* and *News* show a news broadcast where the anchors move slightly while the background remains static.

From an optimization point of view, this improvement is due to the fact that the notable reductions in the R term for high F values exceed the corresponding small increments in the D term. Fig. 2 illustrates the explanation of it. Let A and B be two operating points of the $R - D$ space. Given a λ value, the best coding option is that of the $R - D$ space which first hits the straight line with slope λ. Consequently, when incrementing λ ($\lambda > \lambda_{ref}$)

Table 2: Coding performance for several F values in terms of BDR (%) and BDPSNR_Y (dBs) with respect to the reference HM16.0 software.

		$F = 0.2$		$F = 1.8$		$F = 3.4$		$F = 5.0$	
	Tag	**BDR(%)**	**BDPSNR_Y(dB)**	**BDR(%)**	**BDPSNR_Y(dB)**	**BDR(%)**	**BDPSNR_Y(dB)**	**BDR(%)**	**BDPSNR_Y(dB)**
Akiyo (CIF)	*static*	42.06	-1.56	-8.93	0.45	-13.38	0.72	-13.94	0.77
Foreman (CIF)	*static*	21.15	-0.76	-3.36	0.14	-4.40	0.19	-3.13	0.13
Ice Age (CIF)	*static*	48.83	-2.52	-1.42	0.09	-1.67	0.10	-1.54	0.09
News (CIF)	*static*	25.39	-1.26	-5.01	0.28	-7.97	0.47	-8.51	0.50
Controlled Burn (HD)	*static*	41.20	-1.44	-8.97	0.40	-14.21	0.67	-15.14	0.74
Snow Mountain (HD)	*static*	36.05	-1.25	-9.65	0.38	-15.71	0.67	-17.21	0.77
Average	***static***	**35.78**	**-1.46**	**-6.22**	**0.29**	**-9.56**	**0.47**	**-9.91**	**0.50**
Coastguard (CIF)	*dynamic*	12.18	-0.56	2.22	-0.09	4.42	-0.16	7.79	-0.24
Pedestrian (HD)	*dynamic*	11.16	-0.43	0.29	-0.01	3.49	-0.16	5.92	-0.27
Park Run (HD)	*dynamic*	12.34	-0.60	0.92	-0.05	3.31	-0.15	9.19	-0.36
Speed Bag (HD)	*dynamic*	6.02	-0.15	1.67	-0.07	6.71	-0.29	11.06	-0.47
Average	***dynamic***	**10.42**	**-0.43**	**1.27**	**-0.05**	**4.48**	**-0.19**	**8.49**	**-0.33**

a different coding option is selected (B instead of A). In particular, since increasing λ emphasizes the weight of R in (2), the operating point B accounts for a lower R and a higher D. In the particular case of *static* video

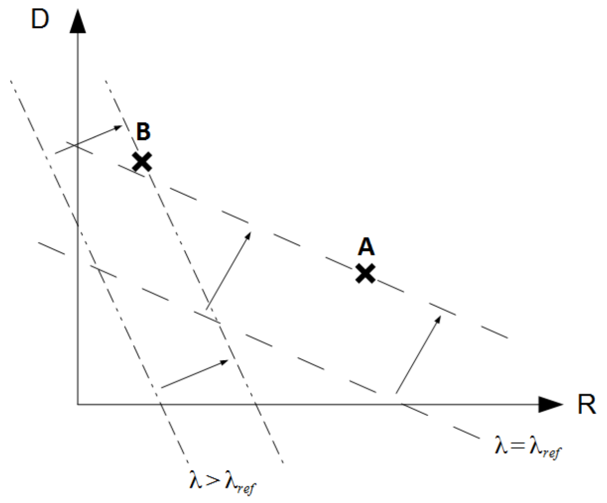


Figure 2: Selection of different $R - D$ points by using different λ values.

sequences, these notable reductions in the R term happen because the temporal prediction is more accurate, and coding with larger CB sizes saves lots of bits in terms of headers, indexes, etc. On the contrary, when considering *dynamic* video sequences, higher F values produce losses in video coding performance (11.06% bit-rate loss for *Speed Bag* is the worst case). Furthermore, also lower F values produce worse results than the reference value (all the F values produce positive bit-rate increments with respect to $F = 1$). As a result, we decided to code *dynamic* sequences using the baseline model. Hence, there is a need to determine in advance the type of background we are dealing with to either using large F values in case of static backgrounds or keep the baseline $\lambda(QP)$ relation ($F = 1$) in case of dynamic backgrounds.

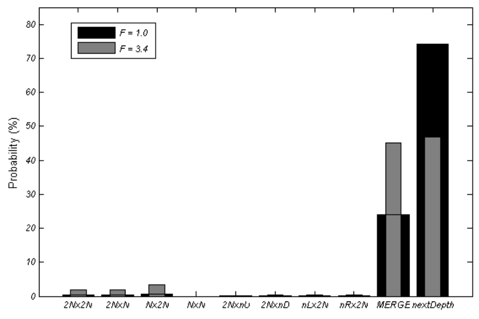
Turning to *static* video sequences again, it should be noted that the optimum value of F is different from one sequence to another. Furthermore, we have considered stationary sequences (20-frame duration) in our experiments, but these conditions do not hold in real videos where scene changes, camera motions, or changes in the background/foreground proportion happen. Hence, an algorithm able to estimate dynamically a proper F value would be desirable.

2.2. Influence of the Lagrange multiplier on complexity

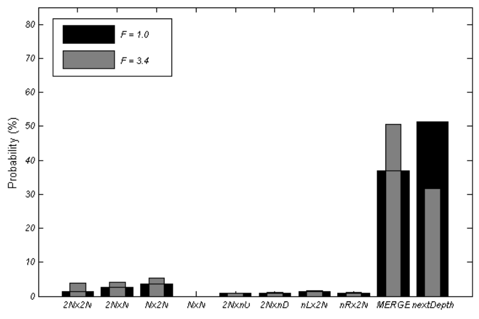
Regarding complexity, considered in Table 3 through the encoding time increment ΔTI (%) defined above, it can be seen that increasing F results in computational cost reductions for all static background video sequences. The reason can be found in Fig. 3, where the probabilities of choosing an specific coding block size[2] when encoding *Controlled Burn* (a static background sequence) at QP27 for $F = 1$ and $F = 3.4$ are shown. Specifically, each graph shows the probability of each CB size for a depth value (from 0 to 3), where *nextDepth* refers to the probability of selecting a size of a deeper depth.

As can be seen, the reference software ($F = 1$) selects higher depths, while lower depths tend to be used for higher Fs. Additionally, the *merge* mode, which is a coding mode where the current CB is coded using the coding options $\boldsymbol{\theta}^*$ of neighboring blocks, is also more likely for higher Fs. These two facts together with the Fast Decision for Merge RD-cost of the HM16.0 reference software generate computation savings for higher Fs. For *dynamic* video sequences, where $F = 1$ has been determined to be the optimal selection, there are no differences with respect to the reference model in terms of CB size selection, and thus, ΔTI values are close to zero.

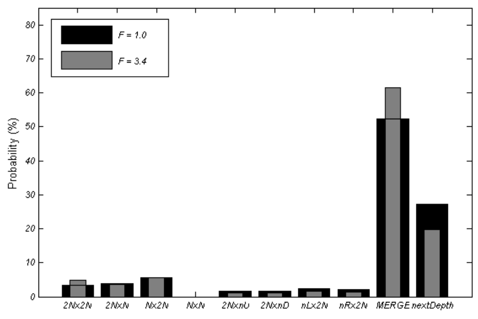
[2]For a review of the CB sizes available in the HEVC standard the reader is referred to [2].



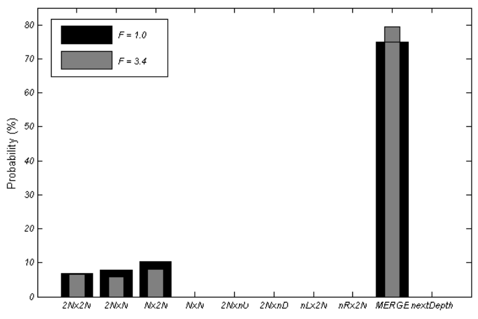
(a) Depth 0.



(b) Depth 1.



(c) Depth 2.



(d) Depth 3.

Figure 3: Comparison between CB size probabilities for several depth values and two values of F ($F = 1$ and $F = 3.4$) for the sequence *ControlledBurn_720p30* at QP27.

3. Proposed Method

3.1. Overview

The proposed method can be described through the following steps: (i) to obtain features that describe the background of the video sequence, (ii) to classify, using these features, between *static* and *dynamic* sequences; and (iii) to find a relation between the features and the optimal F value in order to maximize gains in coding performance. Furthermore, the design of the features, the classifier, and the F estimation method should be done so that the method operates in an adaptive way and does not incur increments in computational complexity.

The flowchart of the proposed method is summarized in Fig. 4. In the *Initialization* stage, F in (4) is set to 1. Then, for each frame, the corresponding features are extracted. Next, a *Classification* stage determines whether

Table 3: Coding performance for several F values in terms of $\Delta TI(\%)$ with respect to the reference HM16.0 software.

		$F = 0.2$	$F = 1.8$	$F = 3.4$	$F = 5.0$
	Tag	$\Delta TI(\%)$	$\Delta TI(\%)$	$\Delta TI(\%)$	$\Delta TI(\%)$
Akiyo (CIF)	*static*	13.03	-1.58	-3.42	-2.32
Foreman (CIF)	*static*	4.50	-1.56	-1.65	-1.60
Ice Age (CIF)	*static*	1.60	-0.61	0.11	0.10
News (CIF)	*static*	3.02	-0.26	-1.90	-2.63
Controlled Burn (HD)	*static*	17.37	-3.44	-6.06	-7.19
Snow Mountain (HD)	*static*	15.43	-3.82	-5.42	-5.35
Average	***static***	**9.16**	**-1.88**	**-3.06**	**-3.16**
Coastguard (CIF)	*dynamic*	5.40	-1.30	1.53	-3.73
Pedestrian (HD)	*dynamic*	1.70	-0.33	-0.24	0.13
Park Run (HD)	*dynamic*	5.56	1.98	4.59	5.96
Speed Bag (HD)	*dynamic*	4.27	-0.91	-2.19	-2.88
Average	***dynamic***	**4.23**	**-0.14**	**0.92**	**-0.13**

the frame is *static* or *dynamic*. If it is classified as *dynamic*, the F multiplier is set to 1; otherwise, a *Regression* stage, which also relies on the previously extracted features, is run to estimate a suitable F multiplier. Then, this F is used to encode the next frame and the process is re-run for each frame until the end of the video sequence.

Two points should be noted: (i) the proposed algorithm makes decisions on a frame basis, starting from the second encoded frame, which makes it adaptive to changes in the video sequence from the very beginning; and (ii) no training is required during the encoding process because the *Classification* and *Regression* stages are defined “off-line”.

3.2. Feature selection

In order to find features that allow classifying each frame as either *static* or *dynamic* frame, all the video sequences used in Section 2 were tagged according to the motion properties of their background as either *static* or *dynamic*. Then, a set of motion-related features such as the number of non-zero residual transformed coefficients, the magnitude of the motion vectors, or the absolute difference between pixels of different frames were tested to check if any allowed us to accurately differentiate between static or dynamic backgrounds.

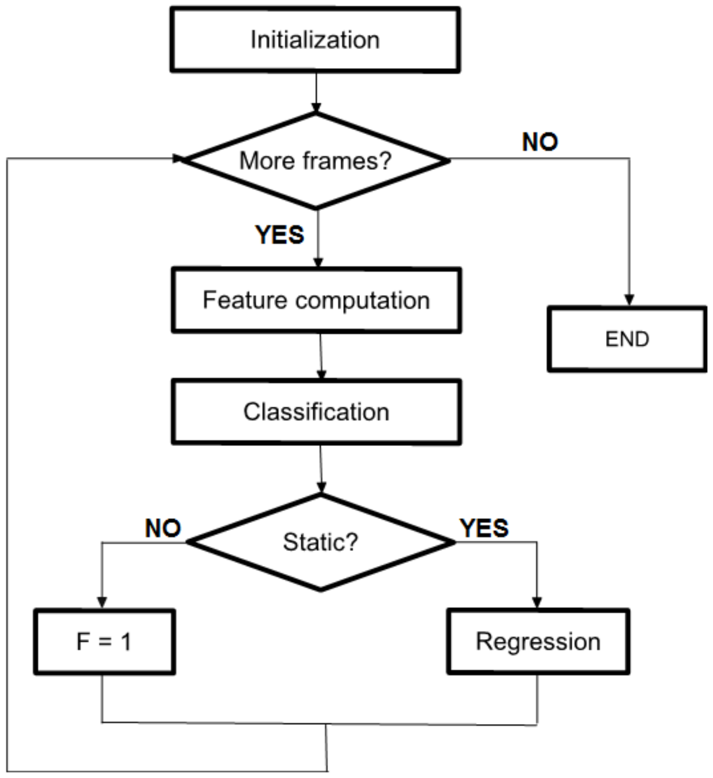
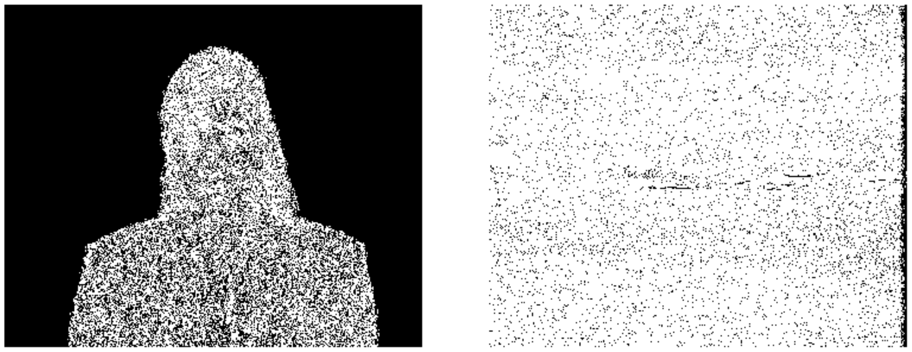


Figure 4: Flowchart of the proposed algorithm.

Among all the analyzed features, the absolute difference between one frame and the previous one was found to be the most useful, as it is sensitive to any relative variation between co-located pixels and it is independent of the encoder configuration parameters (QP, GOP structure, etc.). Fig. 5 shows binarized absolute difference images from *Akiyo* (tagged as *static*) and *Coastguard* (*dynamic*), where white pixels represent high difference values and black ones represent low difference values. As can be seen, the static background of *Akiyo* produces nearly-zero absolute difference values, while higher values are obtained for the anchor. In *Coastguard*, almost the whole frame produces high absolute difference values, as expected from a non-static background.

To be more precise, for practical reasons, we rely on the sum of absolute differences (SAD) between the current 64x64 CB and the co-located one in the previous frame, which has been optimized to be efficiently calculated in



(a) Akiyo (*static* background). (b) Coastguard (*dynamic* background).

Figure 5: Absolute difference images between frames #2 and #3.

the reference encoder and is obtained as follows:

$$SAD = \sum_{x=1}^{S_{CB}} \sum_{y=1}^{S_{CB}} |I_n(x,y) - I_{n-1}(x,y)|, \tag{6}$$

where $I_n(x,y)$ denotes the pixel value at the location (x,y) in the current frame, $I_{n-1}(x,y)$ denotes the same pixel in the previous frame, and S_{CB} is the maximum CB size (which was set to 64).

As our classifier works on a frame-by-frame basis, we define the mean, SAD_m, and the standard deviation, SAD_d, of the SAD values over a frame:

$$SAD_m = \frac{1}{J} \sum_{j=1}^{J} SAD_j, \tag{7}$$

$$SAD_d = \sqrt{\frac{1}{J-1} \sum_{j=1}^{J} (SAD_j - SAD_m)^2}, \tag{8}$$

where J is the number of CTUs in a frame. Additionally, SAD_m and SAD_d are normalized by their mean and standard deviation values μ_{SAD_m} and σ_{SAD_m} (resp. μ_{SAD_d} and σ_{SAD_d}) using:

$$\overline{SAD}_m = \frac{SAD_m - \mu_{SAD_m}}{\sigma_{SAD_m}}, \tag{9}$$

$$\overline{SAD}_d = \frac{SAD_d - \mu_{SAD_d}}{\sigma_{SAD_d}}, \tag{10}$$

where $\overline{SAD}_m$ and $\overline{SAD}_d$ are the normalized versions of SAD_m and SAD_d, respectively.

In order to prove that the previous features are suitable to make a correct classification, we represent in Fig. 6 $\overline{SAD}_d$ versus $\overline{SAD}_m$ for every frame k of the considered video sequences. From data in Fig. 6, two conclusions can be drawn. First, it is feasible to design a classifier that distinguishes between *static* and *dynamic* video sequences on this feature space. Second, both $\overline{SAD}_m$ and $\overline{SAD}_d$ features are useful for classification, since relying only on $\overline{SAD}_m$ or $\overline{SAD}_d$ the classification error increases.

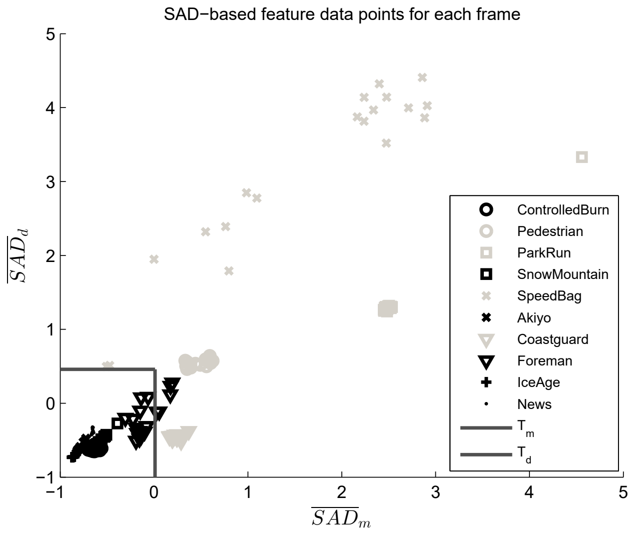


Figure 6: 20 frames of every video sequence are represented in the feature space defined by $\overline{SAD}_m$ and $\overline{SAD}_d$. Black points refer to *static* video sequences and gray points refer to *dynamic* video sequences. T_m and T_d represent the thresholds obtained to perform the classification.

3.3. Classification

The main goal of this stage is to determine whether a frame has a *static* or a *dynamic* background. This decision turns out to be critical for the suitable operation of the proposed method since the λ value should be changed only for *static* frames to avoid losses in coding performance. The classifier, which operates on a frame basis, relies on two thresholds (on $\overline{SAD}_m$ and $\overline{SAD}_d$) to make its decision for each frame k according to this expression:

$$C(\overline{SAD}_m^{(k)}, \overline{SAD}_d^{(k)}) = \\ = \begin{cases} \textit{static} & \text{if } \overline{SAD}_m^{(k)} < T_m \text{ and } \overline{SAD}_d^{(k)} < T_d \\ \textit{dynamic} & \text{otherwise,} \end{cases} \quad (11)$$

where $C(\overline{SAD}_m^{(k)}, \overline{SAD}_d^{(k)})$ represents the classifying function, T_m is the threshold on $\overline{SAD}_m$ and T_d is the threshold on $\overline{SAD}_d$.

These thresholds were obtained by evaluating the likelihood of the estimated probability distributions of both $\overline{SAD}_m$ and $\overline{SAD}_d$, given the tags *static* and *dynamic*. Specifically, for the case of the parameter $\overline{SAD}_m$, the threshold T_m was selected as the value of $\overline{SAD}_m$ which satisfies the following equation:

$$\frac{P(\overline{SAD}_m|\textit{static})}{P(\overline{SAD}_m|\textit{dynamic})} = \frac{P(\textit{dynamic})}{P(\textit{static})}, \quad (12)$$

where $P(\overline{SAD}_m|\textit{static})$ and $P(\overline{SAD}_m|\textit{dynamic})$ are the likelihoods of obtaining $\overline{SAD}_m$ given that the frame is either *static* or *dynamic*, respectively; and $P(\textit{static})$ and $P(\textit{dynamic})$ are the *a priori* probabilities of *static* and *dynamic*. $P(\overline{SAD}_m|\textit{static})$ and $P(\overline{SAD}_m|\textit{dynamic})$ were estimated through normalized histograms [24] and $P(\textit{static})$ and $P(\textit{dynamic})$ were fixed to 0.6 and 0.4, respectively, considering the number of video sequences belonging to each category. This same procedure was used to obtain T_d, finally obtaining $T_m = 0.009$ and $T_d = 0.463$. Fig. 6 shows the classification performance in the training set when the selected T_m and T_d are used.

3.4. Regression

The main goal of this stage is to estimate a suitable value for the F multiplier in (4) to maximize the improvement in terms of coding efficiency.

For this purpose, all the data gathered in Section 2 for *static* sequences and for a wide range of F values were used as training set. First, F_{opt}, which is the value that minimized the BDR with respect to the use of the baseline λ value ($F = 1$), was found for every sequence. An example of the achieved results is shown in Fig. 7 for *News*. As can be seen, the BDR reaches a minimum at $F = 4.2$, which performs notably better than the reference value. Thus, F_{opt} in this case was set as $F_{opt} = 4.2$.

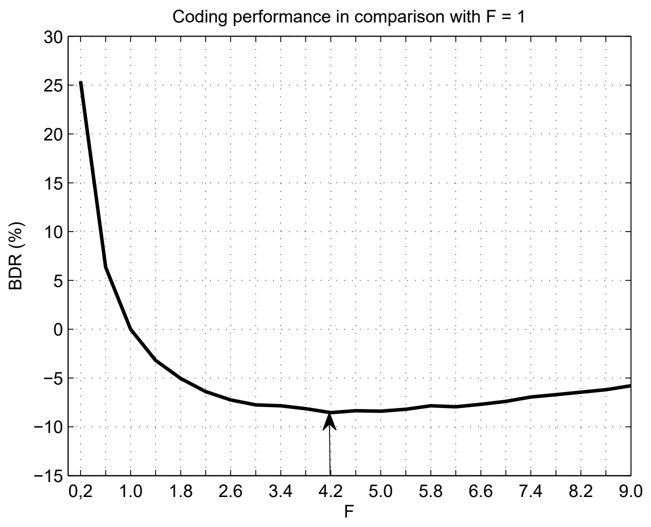


Figure 7: Relative coding performance (in terms of BDR) with respect to the baseline λ ($F = 1$) as a function of F for *News* video sequence. The arrow points out the optimum value of F.

Then, once F_{opt} was obtained for every *static* video sequence, a regression model that predicted this value from the previously described features was found. As shown in Fig. 8, the relation between F_{opt} and $\overline{SAD}_m$ and $\overline{SAD}_d$ can be approximately modeled by means of an exponential function. Thus, the proposed frame basis estimation $\widehat{F}^{(k)}$ of F_{opt} follows the next expression:

$$\widehat{F}^{(k)} = e^{(\alpha\overline{SAD}_m^{(k)} + \beta\overline{SAD}_d^{(k)} + \delta)}, \tag{13}$$

where α, β and δ are regression parameters that are found by converting the previous expression into linear:

$$\ln(\widehat{F}^{(k)}) = \alpha\overline{SAD}_m^{(k)} + \beta\overline{SAD}_d^{(k)} + \delta, \tag{14}$$

and minimizing the mean squared error (MSE) between $\widehat{F}^{(k)}$ and F_{opt} as follows [24]:

$$w^* = \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}s^*, \tag{15}$$

where $w^* = [\alpha, \beta, \delta]$ is the optimal solution; $\mathbf{X} = [1 \quad \overline{SAD}_m^{(1)} \quad \overline{SAD}_d^{(1)}; \cdots ; 1 \quad \overline{SAD}_m^{(K)} \quad \overline{SAD}_d^{(K)}]$ is the extended matrix gathering training data and $s^* = [\ln(F_{opt}^{(1)}); \ \ln(F_{opt}^{(2)}); \ \cdots ; \ \ln(F_{opt}^{(K)})]$ is the vector collecting all the $\ln(F_{opt}^{(k)})$

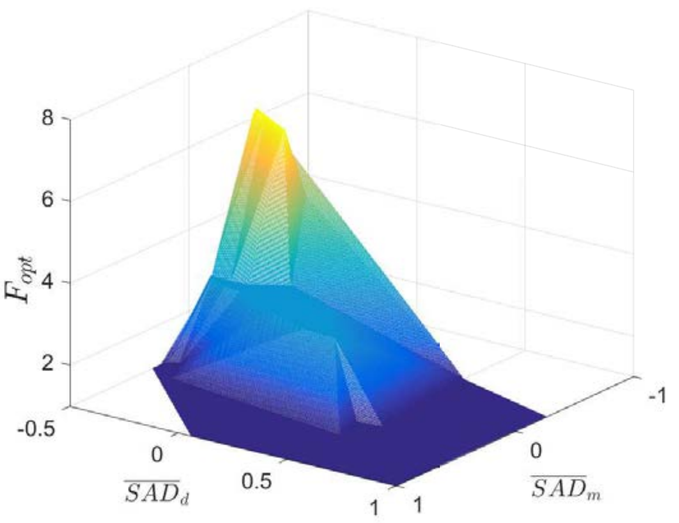


Figure 8: Graphical relationship between F_{opt}, $\overline{SAD}_m$ and $\overline{SAD}_d$.

values corresponding to each pair $\left[\overline{SAD}_m^{(k)}, \overline{SAD}_d^{(k)}\right]$ for $k = [1, 2, \cdots, K]$, being K the number of all frames used for the parameter training process.

Finally, to adjust those estimations producing $\widehat{F}^{(k)} < 1$, which yields bad coding performance results, the final relationship was modified to:

$$\widehat{F}^{(k)} = \max\left\{1.0, e^{(\alpha \overline{SAD}_m^{(k)} + \beta \overline{SAD}_d^{(k)} + \delta)}\right\}. \tag{16}$$

3.5. Additional processing

Some additional processing over $\widehat{F}^{(k)}$ is done to avoid sudden changes of its value from frame to frame, which was experimentally checked to negatively affect the coding performance.

First, $\overline{SAD}_m^{(k)}$ and $\overline{SAD}_d^{(k)}$ features, used in the *Classification* and *Regression* stages, are computed considering N frames instead of just the current one. In particular, each feature is computed as the average value over the $N-1$ previous frames and the current one. This procedure makes the variation of $\widehat{F}^{(k)}$ over k smoother, reducing the likelihood of sudden changes. Regarding the parameter N, a tradeoff should be considered: on the one hand, a high N is desirable because it implies a smooth variation of the $\widehat{F}^{(k)}$ multiplier. On the other, using a low N allows the algorithm to quickly adapt to changes in the video content.

Second, although the features from which F is estimated have been smoothed, a clipping of the frame to frame variation of $\widehat{F}^{(k)}$ has been also added. In particular, $\widehat{F}^{(k)}$ is updated as follows:

$$\widehat{F}^{(k)} = \begin{cases} \max\left\{\widehat{F}^{(k)}, \widehat{F}^{(k-1)} - \mathrm{Th}\right\} & \text{if } \widehat{F}^{(k)} \leq \widehat{F}^{(k-1)} \\ \min\left\{\widehat{F}^{(k)}, \widehat{F}^{(k-1)} + \mathrm{Th}\right\} & \text{if } \widehat{F}^{(k)} > \widehat{F}^{(k-1)}, \end{cases} \tag{17}$$

where $\widehat{F}^{(k-1)}$ denotes the estimation for the previous frame and Th is the clipping threshold, which enables a better control of $\widehat{F}^{(k)}$ on a frame basis.

Proper values for N and Th ($N = 10$ and $Th = 1.5$) were selected using a set of *Train* sequences (Table 4), achieving -1.08% bit-rate reductions in *Test* sequences (Table 4) with respect to a version without additional processing.

3.6. Algorithm

The complete algorithm is summarized in Algorithm 1, where the specific values for the parameters T_m, T_d, Th, N, α, β and δ are given.

4. Experiments and results

In this section, we first assess the two main subsystems of the proposed method, i.e., the classifier and the regressor. Then, we evaluate the coding performance of our proposal in comparison with the HEVC standard and a state of the art method [19]. Then, we test the capability of the proposed method to adapt to varying video content. Finally, we provide an illustration of the subjective quality improvement achieved with the proposed method.

4.1. Classifier and regressor assessment

Before assessing the coding performance of the proposed method, we have checked the efficacy of its two main subsystems separately. To this purpose, we have used as *Train* set the same set of sequences used in Section 2 and Subsection 3.4 and we have added a *Test* set composed of 12 different sequences (see Table 4 for a complete list of sequences). The type of background for all these sequences was manually labeled and the same procedure of Subsection 3.4 was carried out to obtain F_{opt}.

To assess the classifier, the following accuracy measure was used:

$$A(\%) = 100 - \frac{100}{K} \cdot \sum_{n=1}^{K} |T_C - T_{GT}|, \tag{18}$$

Algorithm 1 Proposed coding process.

```
Require: $K$ number of frames.
Require: $J$ number of CTUs.
Require: $N = 10$.
Require: Normalizing parameters $\mu_{SAD_m}$, $\sigma_{SAD_m}$, $\mu_{SAD_d}$, $\sigma_{SAD_d}$.
Require: $T_m = 0.009$, $T_d = 0.463$.
Require: $\alpha = 0.62$, $\beta = 0.01$, $\delta = 1.01$.
Require: Th $= 1.5$.
Require: $\widehat{F}^{(0)} = 1$.
 1: for $\forall\ k \in K$ do
 2:   for $\forall\ j \in J$ do
 3:     Compute $SAD_j$.
 4:     Use $\widehat{F}^{(k-1)}$ in (4) to perform coding.
 5:   end for
 6:   Compute $SAD_m^{(k)}$ and $SAD_d^{(k)}$.
 7:   Compute $\overline{SAD}_m^{(k)}$ and $\overline{SAD}_d^{(k)}$ using (9) and (10).
 8:   if $(\frac{1}{N}\overline{SAD}_m^{(k)} + \frac{1}{N}\sum_{n=1}^{N-1}\overline{SAD}_m^{(n)} < T_m)$
        and $(\frac{1}{N}\overline{SAD}_d^{(k)} + \frac{1}{N}\sum_{n=1}^{N-1}\overline{SAD}_d^{(n)} < T_d)$ then
 9:     Compute $\widehat{F}^{(k)}$ by using (16) and (17).
10:   else
11:     $\widehat{F}^{(k)} = 1$.
12:   end if
13: end for
```

where T_C is the tag provided by the classifier (being 1 for *static* videos, and 0 for *dynamic* videos), T_{GT} is the ground-truth tag listed in the third column of Table 4, and K the number of coded frames, which was set to 20.

The obtained results are shown in Table 4. An average accuracy of 93.33% was obtained on the *Test* set, being almost perfect in 11 of the 12 video sequences.

To properly assess the regressor, the proposed method was compared with an "optimal" encoder using F_{opt} (fourth column of Table 4) for each *static* video sequence (e.g., *Akiyo* was encoded with $F = 5.4$ and *Foreman* with $F = 2.6$).

Results in terms of Y-PSNR and Bit-rate increments, BDPSNR_Y (dB) and BDR (%), respectively, are shown in Table 5 with respect to the "opti-

Table 4: Classification accuracy A(%) of the proposed method for both train and test video sequences.

Sample type	**Sequence**	**Tag**	F_{opt}	**A(%)**
Train	*Akiyo (CIF)*	*static*	5.4	100
	Foreman (CIF)	*static*	2.6	25
	Ice Age (CIF)	*static*	3.4	100
	News (CIF)	*static*	4.2	100
	Controlled Burn (HD)	*static*	5.8	100
	Snow Mountain (HD)	*static*	6.2	100
	Coastguard (CIF)	*dynamic*	1.0	100
	Pedestrian (HD)	*dynamic*	1.0	100
	Park Run (HD)	*dynamic*	1.0	100
	Speed Bag (HD)	*dynamic*	1.0	100
	Average	**-**	**-**	**92.50**
Test	*Bridge Close (CIF)*	*static*	4.6	100
	Bridge Far (CIF)	*static*	2.2	100
	Container (CIF)	*static*	3.8	100
	Hall (CIF)	*static*	5.4	100
	Highway (CIF)	*static*	2.6	100
	Sequence 3 (SD)	*static*	2.6	25
	Tiger & Dragon (SD)	*static*	3.4	100
	Last Samurai (SD)	*static*	3.0	100
	In To Tree (HD)	*static*	3.4	95
	Sequence 10 (SD)	*dynamic*	1.0	100
	Soccer (CIF)	*dynamic*	1.0	100
	Riverbed (HD)	*dynamic*	1.0	100
	Average	**-**	**-**	**93.33**

mal" encoder. As can be seen, the proposed method incurs a bit-rate loss of 0.75% (or a Y-PSNR loss of -0.02 dB) for the *Test* set when compared to the "optimal" encoder. Since this performance is quite close to that of the "optimal" encoder, we consider that the regressor is performing well.

Finally, it is worth noticing that although there is little room for improving the regressor, the classifier seems to be more critical: in the sequence exhibiting the highest losses in terms of BDR (*Foreman*), the classification result is quite poor (see Table 4).

In summary, these results prove that: (i) the proposed classifier allows us to suitably detect static backgrounds for which to modify the $\widehat{F}^{(k)}$ multiplier;

Table 5: Coding performance of the proposed method relative to that of an "optimal" encoder using F_{opt}.

			Proposed	
Sample type	**Sequence**	F_{opt}	**BDPSNR_Y (dB)**	**BDR (%)**
Train	*Akiyo (CIF)*	5.4	-0.02	0.35
	Foreman (CIF)	2.6	-0.15	3.66
	Ice Age (CIF)	3.4	0.00	0.07
	News (CIF)	4.2	-0.02	0.44
	Controlled Burn (HD)	5.8	-0.04	0.70
	Snow Mountain (HD)	6.2	-0.05	0.93
	Average	**-**	**-0.05**	**1.02**
Test	*Bridge Close (CIF)*	4.6	-0.06	2.02
	Bridge Far (CIF)	2.2	-0.02	1.26
	Container (CIF)	3.8	0.00	0.00
	Hall (CIF)	5.4	-0.09	1.95
	Highway (CIF)	2.6	-0.03	1.22
	Sequence 3 (SD)	2.6	-0.01	0.36
	Tiger & Dragon (SD)	3.4	0.00	0.05
	Last Samurai (SD)	3.0	0.01	-0.15
	In To Tree (HD)	3.4	0.01	0.07
	Average	**-**	**-0.02**	**0.75**

and (ii) the proposed regressor can obtain a proper $\widehat{F}^{(k)}$ estimation.

4.2. Coding performance evaluation

The proposed method was implemented in the versions HM12.0 [25] and HM16.0 [6] of the reference software and the encoder configuration was the one shown in Table 6. For the coding performance evaluation, the set of video sequences has been extended with the E Sequences from the HEVC evaluation corpus in [23] (*Four People*, *Kristen and Sara* and *Johnny*). Furthermore, 100 frames of every video sequence were encoded (instead of the 20 frames of previous analyses). It should be noticed that, as long as more frames have been included for these experiments, it would have been more precise to rename the types of sequences as *mainly static* or *mainly dynamic* to account for potential variations over the 100 frames. Also, the BDPSNR for both chroma components BDPSNR_U (dB) and BDPSNR_V (dB), obtained following the procedure described in [23], were computed.

4.2.1. Comparison with State of the Art

A first set of experiments was devoted to perform a comparison between the proposed method and a state-of-the-art method [19], which computes a λ value for each CTU based on the proportion of static background. To that end, we used HM12.0 because the authors of [19] kindly provided us with an executable file of their method implemented in HM12.0 and an encoding configuration file with their coding conditions, which we used. The obtained results are shown in Table 7.

Table 6: Encoder configuration for the HM12.0 and HM16.0 experiments.

Parameter	**Value**
#Frames	100
QP	22, 27, 32, 37
Profile	Low-delay-P
QP Cascading	On
IP (in HM12.0)	-1
IP (in HM16.0)	32

Table 7: Coding performance of the proposed algorithm and [19] relative to the HM12.0 reference software.

		Reference [19]					**Proposed Method**				
Tag		**BDPSNR_Y(dB)**	**BDPSNR_U(dB)**	**BDPSNR_V(dB)**	**BDR(%)**	**ΔTI (%)**	**BDPSNR_Y(dB)**	**BDPSNR_U(dB)**	**BDPSNR_V(dB)**	**BDR(%)**	**ΔTI (%)**
static	*Akiyo (CIF)*	0.14	0.37	0.49	-3.40	-3.73	0.64	1.38	1.50	-14.32	-11.88
	Bridge Close (CIF)	0.03	0.01	0.14	-1.15	-22.83	0.59	0.14	0.23	-22.50	-33.36
	Bridge Far (CIF)	-0.12	0.18	0.12	-17.93	-22.38	0.08	-0.04	-0.03	-22.55	-28.96
	Container (CIF)	0.13	0.61	0.51	-4.05	-10.35	0.37	1.33	1.27	-11.03	-17.54
	Hall (CIF)	0.17	-0.04	0.16	-5.42	-22.40	0.66	-0.01	0.19	-19.36	-28.62
	Highway (CIF)	-0.04	0.06	0.04	1.58	-19.81	0.07	0.00	-0.03	-2.98	-22.23
	Ice Age (CIF)	-0.15	0.05	-0.02	2.81	0.44	0.83	0.69	0.77	-13.89	-7.25
	News (CIF)	-0.07	0.06	0.16	1.32	-0.78	0.47	0.24	0.41	-9.45	-13.46
	Last Samurai (SD)	-0.03	0.04	0.00	0.84	-3.83	1.01	0.59	0.57	-25.88	-4.91
	Tiger & Dragon (SD)	-0.07	0.03	-0.04	2.00	-5.19	-0.07	0.25	0.20	1.97	-8.15
	Controlled Burn (HD)	0.17	0.24	0.27	-5.13	-11.23	0.83	0.88	0.98	-21.45	-19.98
	Four People (HD)	-0.04	0.12	0.15	0.89	-3.80	0.47	0.71	0.70	-11.54	-7.57
	In To Tree (HD)	-0.05	0.03	0.09	2.23	-16.40	0.08	0.12	0.15	-4.60	-22.04
	Kristen and Sara (HD)	0.00	0.15	0.14	0.00	-6.58	0.31	0.68	0.64	-8.60	-9.51
	Johnny (HD)	-0.02	0.23	0.25	1.13	-7.28	0.23	0.70	0.68	-7.96	-8.98
	Snow Mountain (HD)	0.40	0.72	0.62	-15.03	-15.68	0.87	1.42	1.21	-26.64	-23.57
	Average (static)	0.03	0.18	0.19	-2.46	-10.74	0.46	0.57	0.59	-13.80	-16.75
dynamic	*Foreman (CIF)*	-0.17	0.07	0.08	4.27	-6.38	-0.04	0.02	-0.05	0.91	-2.45
	Coastguard (CIF)	-0.27	-0.09	-0.05	7.66	-6.26	0.00	0.00	0.00	0.00	-2.88
	Soccer (CIF)	-0.28	-0.09	-0.03	6.70	-5.87	0.00	0.00	0.00	0.00	-3.22
	Sequence 3 (SD)	-0.24	-0.12	-0.17	6.52	-7.19	0.00	0.00	0.00	0.12	0.20
	Sequence 10 (SD)	-0.18	-0.11	-0.14	5.14	-3.37	0.00	0.00	0.00	0.00	2.26
	Park Run (HD)	-0.20	-0.16	-0.09	4.77	-7.20	0.00	0.00	0.00	0.00	-0.88
	Pedestrian (HD)	-0.31	-0.26	-0.25	7.41	-4.46	0.00	0.00	0.00	0.00	-4.11
	Riverbed (HD)	-0.23	-0.13	-0.06	4.62	-4.01	0.00	0.00	0.00	0.00	-3.51
	Speed Bag (HD)	-0.21	-0.14	-0.05	5.64	-6.39	-0.33	-0.23	-0.22	9.43	-6.29
	Average (dynamic)	-0.24	-0.11	-0.08	5.86	-5.68	-0.04	-0.02	-0.03	1.15	-2.32
	Average (all)	**-0.07**	**0.08**	**0.09**	**0.53**	**-8.92**	**0.28**	**0.36**	**0.37**	**-8.42**	**-11.55**

For *static* video sequences, the proposed method achieves an average gain of -13.80% in terms of the BD-Rate (or 0.46 dBs in terms of BDPSNR_Y) with respect to the reference software; while the method described in [19] achieves an average gain of -2.46% in BD-Rate (0.03 dBs in BDPSNR_Y). Moreover, for *dynamic* video sequences, our proposal applied the reference λ, limiting losses to 1.15% in terms of BD-Rate; while the method described in [19] (not prepared to deal with *dynamic* video sequences) incurred a bit-rate loss of 5.86%.

This notable performance difference can be explained by two main reasons: (i) the proposal in [19] trained the model "on the fly" at the beginning of the encoding process, using M frames of the original video sequence. Thus, during these M frames, [19] did not change the λ value and thus did not achieve improvements comparing with the reference software. This approach works well for video surveillance sequences, but it does not work well for typical video sequences, where the video content changes and thus, the model becomes inefficient (because it was trained for other type of video content). To solve this problem, the model should be re-trained after such changes in the video content, using other M frames in which the algorithm is not enabled; (ii) [19] uses a uni-dimensional space of background percentage bins. Therefore, in some video sequences, one or more of the bins may not have enough data for the training process (as not enough CTUs with a certain percentage of background may be available). Thus, the parametrization of the relationship between background percentage and λ can be inaccurate, leading to poor results in terms of coding performance efficiency. To solve this, the algorithm would need a larger number of frames for training.

We perform both *Classification* and *Regression* parametrizations "off-line" using a bi-dimensional feature space of normalized SAD mean and standard deviations which generalized properly for any video content, as we have shown in Subsection 3.2. Therefore, our approach solves the problems previously described for [19], significantly outperforming its performance.

In terms of the computational efficiency, the proposed method, due to reasons explained in Subsection 2.2, provides on average a time saving of -11.55% compared with the reference software, while the method presented in [19] generates a time saving of -8.92%. However, it should be noted that the computational time required for the training process in [19], which is very complex, is not taken into account in the results. Also, note that the proposed method is fully compatible with many complexity reduction and complexity control approaches in the state of the art (e.g., [26, 27, 28]).

Finally, considering the reference model, it is worth noticing that although the QP Cascading, which also acts on QP on a frame basis, causes a similar effect to that of increasing the λ multiplier, the improvement in coding performance obtained by our proposal is still significantly larger. This improvement comes from the fact that QP Cascading does not take the video content into account, while the proposed method produces a content-aware λ adaptation and, furthermore, it adapts λ in a wider dynamic range than that of the QP cascading.

4.2.2. Comparison with HM16.0 reference software

A second set of experiments was performed to compare the proposed method with a more recent version of the reference software, namely HM16.0, and using a more common encoder configuration (the IP parameter was set to 32 for 30 frames-per-second video sequences, as recommended in [23]) for general purpose video coding.

Table 8: Coding performance of the proposed algorithm relative to the HM16.0 reference software.

		Proposed Method				
Tag		**BDPSNR_Y(dB)**	**BDPSNR_U(dB)**	**BDPSNR_V(dB)**	**BDR(%)**	**ΔTI (%)**
	Akiyo (CIF)	0.51	0.94	0.78	-10.49	-7.59
	Bridge Close (CIF)	0.50	0.08	0.17	-18.06	-17.57
	Bridge Far (CIF)	0.32	0.05	0.04	-25.25	-13.43
	Container (CIF)	0.28	0.59	0.45	-7.23	-9.82
	Hall (CIF)	0.53	-0.03	0.11	-13.90	-17.02
	Highway (CIF)	-0.01	0.00	-0.01	0.60	-15.18
	Ice Age (CIF)	1.38	0.91	0.92	-22.44	-4.75
static	*News (CIF)*	0.27	-0.12	-0.03	-5.27	-8.12
	Last Samurai (SD)	0.87	0.51	0.50	-22.50	-3.19
	Tiger & Dragon (SD)	-0.07	0.20	0.16	1.75	-6.17
	Controlled Burn (HD)	0.55	0.23	0.35	-13.52	-13.08
	Four People (HD)	0.44	0.49	0.39	-10.13	-6.22
	In To Tree (HD)	0.07	-0.04	-0.01	-3.27	-11.91
	Kristen and Sara (HD)	0.30	0.43	0.38	-7.95	-8.30
	Johnny (HD)	0.17	0.41	0.34	-5.75	-6.81
	Snow Mountain (HD)	0.54	0.91	0.79	-13.75	-13.57
	Average (static)	0.42	0.35	0.33	-11.07	-10.17
	Foreman (CIF)	-0.08	-0.09	-0.15	2.04	-2.40
	Coastguard (CIF)	0.00	0.00	0.00	0.00	0.55
	Soccer (CIF)	0.00	0.00	0.00	0.00	-1.88
	Sequence3 (SD)	-0.01	0.00	0.00	0.14	0.25
	Sequence10 (SD)	0.00	0.00	0.00	0.00	0.37
dynamic	*Riverbed (HD)*	0.00	0.00	0.00	0.00	0.10
	Pedestrian (HD)	0.00	0.00	0.00	0.00	0.10
	Park Run (HD)	0.00	0.00	0.00	0.00	0.48
	Speed Bag (HD)	-0.24	-0.22	-0.21	6.84	-3.83
	Average (dynamic)	-0.04	-0.03	-0.04	1.00	-0.70
	Average (all)	**0.25**	**0.21**	**0.20**	**-6.72**	**-6.76**

The improvements of the proposed algorithm over HM16.0 reference software are still quite significant. In particular, an average bit-rate saving of -11.07% was achieved for *static* sequences and quite similar results than those of the reference (a bit-rate increment of 1.00%) were achieved for *dynamic* sequences. Moreover, taking into account all the sequences, an average bit-rate reduction of -6.72% was achieved. The improvements are a little bit lower when compared with those achieved with respect to HM12.0 simply because we have changed the encoder configuration to include an Intra frame every 32 frames, and Intra frames do not benefit as much as Inter frames from adapting the λ parameter.

4.3. Adaptive performance

In this subsection, the adaptation capability of the proposed method is assessed. To that purpose, a simple variation of the proposed method was implemented in the HM16.0 reference software. In particular, the *Classification* and *Regression* stages were only activated in the first frame, keeping the obtained $\widehat{F}^{(1)}$ multiplier constant for the rest of the video sequence. The results obtained by this variation of the proposed method (hereafter referred to as fixed-F) are compared with those of the complete proposal in Table 9.

Table 9: Coding performance comparison of the proposed algorithm and the fixed-F version relative to the HM16.0 reference software.

Tag		**Fixed-F Method**				**Proposed Method**			
		BDPSNR_Y(dB)	**BDPSNR_U(dB)**	**BDPSNR_V(dB)**	**BDR(%)**	**BDPSNR_Y(dB)**	**BDPSNR_U(dB)**	**BDPSNR_V(dB)**	**BDR(%)**
static	*Akiyo (CIF)*	0.33	0.53	0.39	-6.94	0.51	0.94	0.78	-10.49
	Bridge Close (CIF)	0.30	0.06	0.10	-10.82	0.50	0.08	0.17	-18.06
	Bridge Far (CIF)	0.19	0.09	0.07	-19.86	0.32	0.05	0.04	-25.25
	Container (CIF)	0.22	0.31	0.29	-5.41	0.28	0.59	0.45	-7.23
	Hall (CIF)	0.39	-0.01	0.07	-10.43	0.53	-0.03	0.11	-13.90
	Highway (CIF)	0.02	-0.01	-0.01	-0.69	-0.01	0.00	-0.01	0.60
	Ice Age (CIF)	0.77	0.43	0.52	-13.19	1.38	0.91	0.92	-22.44
	News (CIF)	0.21	-0.06	-0.01	-4.01	0.27	-0.12	-0.03	-8.12
	Last Samurai (SD)	0.19	0.01	0.04	-5.28	0.87	0.51	0.50	-22.50
	Tiger & Dragon (SD)	0.02	-0.18	-0.18	-0.57	-0.07	0.20	0.16	1.75
	Controlled Burn (HD)	0.35	0.15	0.19	-8.13	0.55	0.23	0.35	-13.52
	Four People (HD)	0.28	0.29	0.26	-6.79	0.44	0.49	0.39	-10.13
	In To Tree (HD)	0.00	0.00	0.00	0.00	0.07	-0.04	-0.01	-3.27
	Kristen and Sara (HD)	0.21	0.29	0.24	-5.76	0.30	0.43	0.38	-7.95
	Johnny (HD)	0.11	0.22	0.18	-3.95	0.17	0.41	0.34	-5.75
	Snow Mountain (HD)	0.37	0.54	0.45	-9.13	0.54	0.91	0.79	-13.75
	Average (static)	0.25	0.17	0.16	-6.93	0.42	0.35	0.33	-11.07
dynamic	*Foreman (CIF)*	0.00	0.00	0.00	0.00	-0.08	-0.09	-0.15	2.04
	Coastguard (CIF)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	Soccer (CIF)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	Sequence 3 (SD)	0.00	0.00	0.00	0.00	-0.01	0.00	0.00	0.14
	Sequence 10 (SD)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	Park Run (HD)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	Pedestrian (HD)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	Riverbed (HD)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	Speed Bag (HD)	0.00	0.00	0.00	0.00	-0.24	-0.22	-0.21	6.84
	Average (dynamic)	0.00	0.00	0.00	0.00	-0.04	-0.03	-0.04	1.00
	Average (all)	**0.16**	**0.11**	**0.10**	**-4.43**	**0.25**	**0.21**	**0.20**	**-6.72**

For *static* video sequences, an improvement of -4.14% in terms of bit-rate savings was achieved by adapting the λ parameter to the video content. For *dynamic* video sequences, the method incurred reduced losses of 1.00% due to some misclassifications. Taking into account the whole set of sequences, a global improvement of -2.29% was obtained.

Some of the more appealing results happen for the *Last Samurai* sequence, in which the background changes over time. In particular, along the first 100 frames three different scenarios are shown, separated by scene cuts, each exhibiting a different amount of static background. In this case, the fixed-F method achieves a BDR improvement of -5.28% because the first scene exhibits a static background. Nevertheless, by allowing the algorithm to adapt to the content, the proposed algorithm reaches a bit-rate saving of -22.50%, which is significantly higher than that achieved by the fixed-F method. The same behavior can be observed for *Ice Age*, where a cross-fade between two scenes happens at frame #85, where the proposed method is able to properly adapt the λ parameter yielding a significant improvement in coding performance (-22.44% bit-rate saving vs. -13.19% of the fixed-F method). Furthermore, it is worth mentioning the performance improvement for *In To Tree*, which shows a movement towards a tree in a static scene. In this case, the proposed method is able to adapt to those fragments of the video sequence in which the movement is not important, achieving -3.27% coding improvements relative to both the reference HM16.0 and the fixed-F method.

Finally, it is also worth discussing the case of *Speed Bag*, where the fixed-F method achieves a notably better result than that of the proposed method. This happens because one important segment of this sequence shows illumination changes that are not properly managed by the classifier, yielding significant coding losses.

In summary, it can be concluded that the adaptation capability of the proposed method makes it to manage properly realistic situations in which background characteristics change over time.

4.4. Subjective quality assessment

In addition to the objective evaluation relying on BDR (%) and BDPSNR_Y (dB), we provide an illustration of the subjective quality achieved.

The HM16.0 reference software was used to encode 20 frames of the *Controlled Burn* video sequence at QP32, obtaining a target bit-rate for the proposed method. Then, the QP was adjusted for the proposed method to produce a similar bit-rate and a subjective visual comparison was performed.

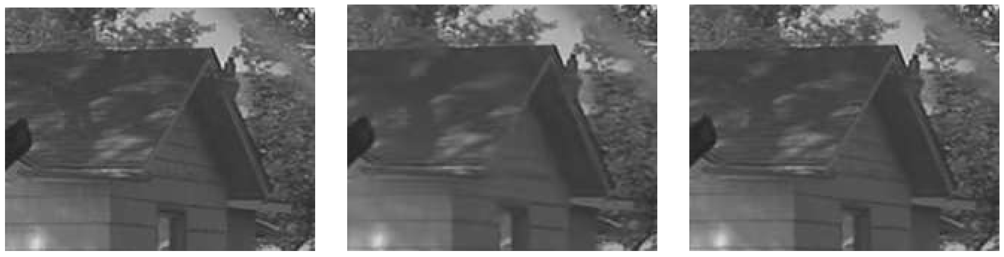

(a) (b) (c)

Figure 9: *Controlled Burn* decoded video fragments belonging to frame #8. (a) Original sequence. (b) Decoded frame using the reference HM16.0 software. (c) Decoded frame using the proposed method.

Fig. 9(a) shows a part of the original frame #8. Fig. 9(b) and Fig. 9(c) show the same frame decoded using the HM16.0 reference software (coded at 35.7 Kbits) and using the proposed method (coded at 35.2 Kbits), respectively. As can be seen, the frame obtained by the proposed method shows more detail than that of the reference software. It is specially noticeable in the right wall of the hut, where the horizontal lines are blurred to the point of almost disappear in the center image.

Thus, as shown by the previous objective evaluations and illustrated through a subjective example, the proposed method saves bits by adapting the $\lambda(QP)$ relationship, allowing the encoder to obtain a better visual quality for the same target bit-rate when compared with the reference HM16.0 software.

5. Conclusions and further work

In this paper a method has been proposed to adaptively select the Lagrangian parameter λ of the cost function associated with the RDO process in the HEVC reference software. This approach has been motivated by means of an experiment that proves that video sequences with static background are more efficiently encoded using higher values of the parameter λ than that of the reference software.

In order to determine whether the background of a sequence is static on a frame basis, some coding-derived features that describe the static or dynamic nature of the background have been found and a classifier has been designed. Furthermore, an exponential regression function has also been proposed to estimate a proper value of the λ parameter. In so doing, the proposed method becomes content-aware, being able to dynamically increase the λ parameter when encoding static background video sequences and keeping it as in the reference software when encoding dynamic background sequences.

The efficacy of both the classifier and the regressor has been experimentally proved. Subsequently, the proposed method has been compared with a state-of-the-art method [19] yielding a significantly better average performance. Moreover, the proposed method has also been assessed in comparison with the HM16.0 version of the reference software, achieving average bit-rate savings of -11.07% (or BDPSNR_Y gains of 0.42 dBs) for *static* video sequences and incurring quite limited losses for *dynamic* video sequences. All these conclusions have been supported by a comprehensive set of experiments over a large set of video sequences. Furthermore, an illustrative example of subjective improvement has been provided.

Finally, the computational efficiency of the proposed method has also been assessed, proving that in average the proposed method turns out to be less demanding than the reference software.

Regarding future lines of research, our experiments have revealed that the classifier performance becomes critical. Therefore, the study of ways to improve the classifier is left as a promising future line of research. In particular, a deeper exploration of potential encoding-derived features might improve the classifier performance.

[1] JVT, High Efficiency Video Coding (HEVC), ITU-T H.265 and ISO/IEC 23008-2, 1 ed. (2013).

[2] G. J. Sullivan, J. Ohm, W.-J. Han, T. Wiegand, Overview of the High Efficiency Video Coding (HEVC) Standard, Circuits and Systems for Video Technology, IEEE Transactions on 22 (12) (2012) 1649–1668.

[3] JVT, Advanced Video Coding (AVC), ITU-T Rec. H.264 and ISO/IEC 14496-10 (MPEG-4 Part 10), 8 ed. (2014).

[4] H. Everett III, Generalized Lagrange Multiplier Method for Solving Problems of Optimum Allocation of Resources, Operations research 11 (3) (1963) 399–417.

[5] A. Ortega, K. Ramchandran, Rate-Distortion Methods for Image and Video Compression, Signal Processing Magazine, IEEE 15 (6) (1998) 23–50.

[6] K. McCann, C. Rosewarne, B. Bross, M. Naccari, K. Sharman, G. Sullivan, High Efficiency Video Coding (HEVC) Test Model 16 (HM 16) Improved Encoder Description (October 2014).

[7] G. J. Sullivan, T. Wiegand, Rate-Distortion Optimization for Video Compression, Signal Processing Magazine, IEEE 15 (6) (1998) 74–90.

[8] G. M. Schuster, G. Melnikov, A. K. Katsaggelos, A Review of the Minimum Maximum Criterion for Optimal Bit Allocation among Dependent Quantizers, IEEE Transactions on Multimedia 1 (1) (1999) 3–17. `doi:10.1109/6046.748167`.

[9] B. Li, H. Li, L. Li, J. Zhang, Rate Control by R-lambda Model for HEVC (October 2012).

[10] J. Si, S. Ma, S. Wang, W. Gao, Laplace Distribution Based CTU Level Rate Control for HEVC, in: Visual Communications and Image Processing (VCIP), 2013, IEEE, 2013, pp. 1–6.

[11] X. Li, N. Oertel, A. Hutter, A. Kaup, Laplace Distribution Based Lagrangian Rate Distortion Optimization for Hybrid Video Coding, Circuits and Systems for Video Technology, IEEE Transactions on 19 (2) (2009) 193–205. `doi:10.1109/TCSVT.2008.2009255`.

[12] T. Biatek, M. Raulet, J.-F. Travers, O. Deforges, Efficient Quantization Parameter Estimation in HEVC Based on ρ-domain, in: Signal Processing Conference (EUSIPCO), 2013 Proceedings of the 22nd European, IEEE, 2014, pp. 296–300.

[13] A. Rehman, Z. Wang, SSIM-inspired Perceptual Video Coding for HEVC, in: Multimedia and Expo (ICME), 2012 IEEE International Conference on, IEEE, 2012, pp. 497–502.

[14] L. Deng, F. Pu, S. Hu, C.-C. J. Kuo, HEVC Encoder Optimization Based on a New RD Model and Pre-Encoding, in: Picture Coding Symposium (PCS 2013), IEEE, 2013.

[15] J. Xiong, H. Li, F. Meng, Q. Wu, K. N. Ngan, Fast HEVC Inter CU Decision Based on Latent SAD Estimation, IEEE Transactions on Multimedia 17 (12) (2015) 2147–2159. doi:10.1109/TMM.2015.2491018.

[16] H. Liu, B. Song, F. Tian, H. Qin, Joint Sampling Rate and Bit-Depth Optimization in Compressive Video Sampling, IEEE Transactions on Multimedia 16 (6) (2014) 1549–1562. doi:10.1109/TMM.2014.2328324.

[17] Z. Liu, D. Wang, J. Zhou, T. Ikenaga, Lagrangian Multiplier Optimization Using Correlations in Residues, in: Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on, 2012, pp. 1185–1188. doi:10.1109/ICASSP.2012.6288099.

[18] H. Zeng, K. N. Ngan, M. Wang, Perceptual Adaptive Lagrangian Multiplier for High Efficiency Video Coding, in: Picture Coding Symposium (PCS), 2013, 2013, pp. 69–72. doi:10.1109/PCS.2013.6737685.

[19] L. Zhao, X. Zhang, Y. Tian, R. Wang, T. Huang, A Background Proportion Adaptive Lagrange Multiplier Selection Method for Surveillance Video on HEVC, in: Multimedia and Expo (ICME), 2013 IEEE International Conference on, IEEE, 2013, pp. 1–6.

[20] X. Zhang, Y. Tian, T. Huang, S. Dong, W. Gao, Optimizing the Hierarchical Prediction and Coding in HEVC for Surveillance and Conference Videos with Background Modeling, IEEE Transactions on Image Processing 23 (10) (2014) 4511–4526.

[21] J. Gonzalez-de Suso, A. Jimenez-Moreno, E. Martinez-Enriquez, F. Diaz-de Maria, Improved Method to Select the Lagrange Multiplier for Rate-Distortion Based Motion Estimation in Video Coding, Circuits and Systems for Video Technology, IEEE Transactions on 24 (3) (2014) 452–464. doi:10.1109/TCSVT.2013.2276857.

[22] G. Bjontegaard, Calculation of Average PSNR Differences between RD-Curves, VCEG-M33, ITU-Telecommunications Standardization Secto (2001) 290–294.

[23] F. Bossen, Common Test Conditions and Software Reference Configurations, jCTVC-L1100.xls (September 2013).

[24] C. M. Bishop, et al., Pattern Recognition and Machine Learning, Vol. 4, Springer New York, 2006.

[25] F. Bossen, D. Flynn, K. Shring, High Efficiency Video Coding (HEVC) Test Model 12 (HM 12) Reference Software.

[26] L. Shen, Z. Liu, X. Zhang, W. Zhao, Z. Zhang, An Effective CU Size Decision Method for HEVC Encoders, IEEE Transactions on Multimedia 15 (2) (2013) 465–470. doi:10.1109/TMM.2012.2231060.

[27] J. Xiong, H. Li, Q. Wu, F. Meng, A Fast HEVC Inter CU Selection Method Based on Pyramid Motion Divergence, IEEE Transactions on Multimedia 16 (2) (2014) 559–564. doi:10.1109/TMM.2013.2291958.

[28] A. J. Moreno, E. Martinez-Enriquez, F. D. de Maria, Complexity Control Based on a Fast Coding Unit Decision Method in the HEVC Video Coding Standard, IEEE Transactions on Multimedia PP (99) (2016) 1–1. doi:10.1109/TMM.2016.2524995.