Codec and GOP Identification in Double Compressed Videos

P. Bestagini, *Member, IEEE*, S. Milani, *Member, IEEE*, M. Tagliasacchi, *Member, IEEE*, S. Tubaro, *Senior Member, IEEE*

***Abstract*—Video content is routinely acquired and distributed in digital compressed format. In many cases, the same video content is encoded multiple times. This is the typical scenario that arises when a video, originally encoded directly by the acquisition device, is then re-encoded, either after an editing operation, or when uploaded to a sharing website. The analysis of the bitstream reveals details of the last compression step (i.e., the codec adopted and the corresponding encoding parameters), while masking the previous compression history. Therefore, in this paper we consider a processing chain of two coding steps, and we propose a method that exploits coding-based footprints to identify both the codec and the size of the Group Of Pictures (GOP) used in the first coding step. This sort of analysis is useful in video forensics, when the analyst is interested in determining the characteristics of the originating source device, and in video quality assessment, since quality is determined by the whole compression history. The proposed method relies on the fact that lossy coding is an (almost) idempotent operation. That is, re-encoding a video sequence with the same codec and coding parameters produces a sequence that is similar to the former. As a consequence, if the second codec in the chain does not significantly alter the sequence, it is possible to analyze this sort of similarity to identify the first codec and the adopted GOP size. The method was extensively validated on a very large dataset of video sequences generated by encoding content with a diversity of codecs (MPEG-2, MPEG-4, H.264/AVC, DIRAC) and different encoding parameters. In addition, a proof of concept showing that the proposed method can be used also on videos downloaded from YouTube is reported.**

***Index Terms*—Video forensics, video codec, coding-based footprints, GOP identification**

I. Introduction

Due to the increasing availability of inexpensive digital devices, camcorders are becoming widespread on the market, being embedded in virtually all smartphones. Moreover, thanks to the ubiquitous availability of high-speed Internet connection and the increasing use of video sharing web sites (e.g., YouTube, Vimeo, etc.), many users upload video sequences on the web. At the same time, digital videos might undergo several editing steps during their lifetime. For example, after acquiring a sequence, a user might manipulate it to enhance its quality. Alternatively, after having downloaded a sequence from a website, a user might apply different kinds of transformations, including, e.g., cropping, scaling, color adjustment, text/logo insertion, etc. After any editing operation, video sequences are usually encoded, since uncompressed video would lead to a huge amount of data to be stored or transmitted. Therefore, it is very likely that a video sequence available online has been compressed multiple times.

When a sequence is decoded to the pixel domain and then re-encoded, any information regarding the previous coding step is apparently lost, since the previous compression history cannot be simply obtained by parsing the bitstream of the last coding step. However, video coding is lossy in most cases, since it encompasses the use of quantization, which is a non-invertible operation. Thus, each coding step is bound to leave characteristic traces, or footprints, on the compressed video, which can be leveraged as an asset to reconstruct its compression history. This sort of information is very useful in video forensics [3], since an analyst might be able to estimate the characteristics of the previous coding steps (e.g., the codec adopted and the corresponding encoding parameters, namely, GOP structure, quantization parameters, coding modes, etc.). A forensic analyst can successfully exploit the knowledge of the compression history in many ways: i) it might serve as a potential clue for source device identification, as it might, or might not, be compatible with some acquisition device models; ii) it might be used for video splicing detection, in those cases in which sequences originally encoded separately, are then spliced together (and re-encoded) into the same sequence; iii) it might be adopted to detect removed/inserted frames, since this might alter the original GOP structure. In addition, there are other fields besides video forensics in which the knowledge of the compression history is highly valuable. For example, in video quality assessment, it is interesting to automatically estimate the quality of a sequence in a no-reference set-up, i.e., without the availability of the original sequence. Although many no-reference video quality metrics do exploit directly the information contained in the bitstream of the last coding step, this might not be representative of the actual visual quality, e.g., when a content is first compressed at low rate, and then re-compressed at a higher rate [4].

In this paper we address the problem of reconstructing the compression history of video sequences. We consider the case of double compressed video sequences, and we focus on the identification of the coding-based footprints left by the first encoder. Specifically, we aim at determining the codec adopted by the first step, namely the coding standard, and the corresponding GOP size. Although multiple video compression

P. Bestagini, M. Tagliasacchi and S. Tubaro are with the Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Piazza Leonardo da Vinci 32, 20133, Milan, Italy (e-mail: paolo.bestagini / marco.tagliasacchi / stefano.tubaro@polimi.it).

S. Milani is with the Department of Information Engineering, University of Padova, Via Gradenigo 6/b, 35131, Padova, Italy (e-mail: simone.milani@dei.unipd.it).

This work is an extension of previous work published at ICASSP'12 [1] and EUVIP'13 [2] by the same authors.

can be encountered in some situations, double compression is arguably the most common use case, which arises any time user generated content is acquired (and compressed) by a device and subsequently uploaded to a video sharing web site.

Unlike double image compression, in which the same codec (namely JPEG) is typically assumed to be used in both coding steps, video compression is often performed with codecs defined by different standards, thus complicating the analysis. In our study, we considered widely adopted video coding standards that follow the conventional hybrid DCT/DPCM architecture, namely MPEG-2 [5], MPEG-4 part 4 (MPEG-4) [6] and MPEG-4 part 10 - H.264/AVC (AVC) [7]. In addition, we extended the analysis to include a wavelet-based codec, DIRAC [8].

The proposed method is based on the idea that quantization is an idempotent operation. Indeed, re-quantizing a scalar value with the same quantizer produces exactly the same value. Somewhat similarly, in the specific case of video coding, re-encoding a previously compressed video sequence adopting the same codec and coding parameters produces as output a sequence that is similar to the former. Therefore, our idea is that, in principle, it is possible to re-encode the available (double compressed) video sequence with different codecs, testing for each of them different encoding parameters, and looking for the configuration of codecs/parameters which minimizes the distortion introduced in the re-encoded sequence. As a matter of fact, instead of performing a full search of all coding parameters (i.e., GOP, use of filters, macro-block structure, etc.), we propose an algorithm that is able to loop over just two of them (i.e., the codec and the quantization step). The same method is also used for determine the GOP size of the first coding step, by looking for those frames which were originally encoded using intra-frame coding (I-frames).

As the proposed method is based on re-encoding the video sequence under analysis, the computational time strongly depends on the size of the search space of the re-encoding parameters (i.e., number of tested codec and quantization steps). However, this is not considered to be an issue in most forensic applications where the amount of videos to analyze is limited (e.g., even just one suspect video). Anyway, in this manuscript we also show that it is possible to significantly decrease the size of the parameters' search space strongly reducing the computational complexity of the proposed algorithm without impairing its accuracy. The possibility of reducing the computational complexity of the proposed algorithm enables it to be used also for different applications, where efficiency is a compelling requirement.

Our method was validated on a very large dataset of double compressed sequences, which was generated with a wide range of codecs and encoding parameters. The results reveal the possibility of identifying the coding standard and the GOP size of the first coding step, depending on the strength of the second coding step. Specifically, when the distortion introduced by the second coding step is not significantly stronger than the distortion introduced by the first codec, both the coding standard and the GOP size can be reliably estimated. Moreover, in order to show that the proposed method can be applied in a typical real-world scenario, we tested it on a set of video sequences uploaded and downloaded from YouTube. Also in this case the codec of the first compression step was correctly identified.

The rest of this paper is organized as follows. Section II discusses the related work, focusing on similar ideas recently applied to the case of still images. Section III introduces the problem addressed in this paper and Section IV the idea of exploiting the idempotent property of quantization. Section V illustrates in details the proposed method to identify both the coding standard and the GOP size. A thorough experimental validation is presented in Section VI, and Section VII concludes the paper, providing insights on open problems and future challenges.

II. Related work

The study of the compression history has been widely addressed in the past literature for the case of still images [9]. The problem of determining whether an image was compressed with JPEG was originally studied in [10] and further explored in [11], proposing a method to estimate the quantization matrix by looking at the periodicities in the distribution of the DCT coefficients. In the case of forgeries, the image is often compressed twice, thus stimulating work aimed at determining whether an image (or part of it) is single vs. double compressed [12]. The solution to this problem was recently extended to the challenging case in which the image is cropped [13], resized [14], or contrast enhanced [15] between the first and second compression. Although most of the images are compressed using JPEG, in [16] the authors show how to discriminate between different block-wise transform image codecs (e.g., DCT-based, or DWT-based). A theoretical analysis of the footprint left by transform coding, a key element in most image coding architectures, was presented in [17] and later extended in [18] for the case of double compression.

The study of the compression history of video sequences has been explored only more recently [3]. This has to do with the intrinsic difficulty in dealing with video sequences rather than still images. First, in the case of video, there is no single coding standard that is universally used. Conversely, different standards are being adopted depending on the scenario, so that sequences might be encoded with legacy MPEG-2 and MPEG-4 encoders, as well as with the more recent AVC and HEVC [19] standards. Second, the number of degrees of freedom when configuring a video encoder is significantly larger than in JPEG. These include, for example, the choice of the GOP structure, the coding mode decision rules, the motion estimation algorithm, the rate control algorithm, etc. Third, complex statistical dependencies are created between the quantized coefficient values in different frames, due to the adoption of motion-compensated prediction. For these reasons, early work addressing video compression focused on the analysis of MPEG-2 encoded sequences, and more specifically on I-frames, to detect the quantization parameter [20] or double compression [21], being a straightforward extension of the methods applicable to still images. Conversely, the early works addressing the specific challenges posed by the analysis of compressed video focused on the estimation of the quantization parameter (QP), which determines the distortion

introduced in each frame (and, in some standards, in each coding unit). In [22] and [23], the authors propose a method to estimate the QP parameter in both I- and P-frames for the case of, respectively, MPEG-2 and AVC coded video, by analyzing the histograms computed from DCT coefficients of prediction residuals. The estimation of the GOP structure (i.e., how I- , P- and B-frames alternate in a sequence), or, more simply, the GOP size (i.e., the distance between consecutive I-frames) was addressed in [24] for single-compressed video based on the strength of spatial blocking artifacts. A method suitable for the case of double-compression was proposed in [25], by exploiting the footprints left by the skip coding mode. Anyway, in this manuscript we propose a more general method that works also with codecs based on different kinds of transforms (i.e., DCT and DWT) not exploiting macro-blocks size. A more recent work targeting double-compressed video is [4], where the authors estimate the bitrate adopted in the first compression step. In this case, the first and second codecs are considered to be the same, thus known. Other works address the analysis of motion vectors. In [26], it is shown how to reconstruct motion vectors in AVC-encoded video from decoded pixels, whereas the identification of the motion estimation strategy is proposed in [27]. In case of multiple video compression, a method to estimate the number of compression steps is described in [28], which is based on the analysis of the distribution of the first digits of DCT coefficients, thus extending the previous work that covered the case of JPEG compression [29].

This paper goes beyond the previous literature by addressing for the first time the problem of estimating the video coding standard adopted in the first coding step in the case of double video compression. The proposed method is based on a recompress-and-observe paradigm, inspired by similar methods presented in the literature, which exploit the idempotency property of quantization in order to, e.g., estimate the quality factor in JPEG compressed images [30], exposing forgeries in compressed images [31] and to detect JPEG anti-forensics [32]. However, to the best of the authors' knowledge, it has never been applied to the problem of identifying the video codec.

With respect to our previous conference publications [1], [2], several improvements have been made:

- We present an analytical explanation of the idempotency property of quantizers (i.e., Section IV), which explains the rationale behind the proposed algorithm.
- We address for the first time the problem of the estimation of the GOP size, which was assumed to be given in [1], [2]. To this purpose, we also provide a thorough validation of the proposed GOP estimation algorithm (see Section VI).
- The codec identification algorithm has been modified to increase its robustness to the masking introduced by the second coding step. The new algorithm does not require a linear behavior between the Quantization Parameter (QP) (or its logarithm) and Peak Signal to Noise Ratio (PSNR), which was assumed in [1], [2].
- The dataset used for testing has been widely expanded. We are now considering more videos (also at different resolutions) and more codecs (both DCT- and DWT-based). Moreover we tested the effect of using different implementations of the same codec (for both MPEG4 and H.264). Additionally, we tested the proposed methods on videos downloaded from YouTube to prove the effectiveness of the new algorithm also in a real-world scenario.
- We investigated the possibility of reducing the computational complexity of the presented algorithm, demonstrating the trade-off between accuracy and complexity in our experimental results.

III. Problem Formulation

Conventional video coding is implemented according to an architecture that can be described with a chain of basic processing operators, which is illustrated in Figure 1. The encoder processes a video sequence $\mathbf{X}$ frame-by-frame. Each frame is divided into blocks $\mathbf{x}$, which are then coded according to two main coding modes (note that in some cases, the frame includes a single block, which is as large as the whole frame). In intra-frame coding, each I-frame is considered as a stand-alone image, and inter-pixel correlation within the same block is exploited by means of transform coding. To increase the coding efficiency, in the more recent standards (e.g., AVC and HEVC) intra prediction can be enabled by using a prediction algorithm $\mathcal{P}$, which computes a predictor from the pixels of neighboring blocks and subtracts it to the current block before applying transform coding. In inter-frame coding, for each block of a P- or B-frame, a predictor is computed exploiting the pixels in one (or more) reference frame, by means of Motion Estimation $\mathcal{ME}$ and Motion Compensation $\mathcal{MC}$. Specifically, $\mathcal{ME}$ looks for the best predictor in the reference frame, which is identified by means of a motion vector (MV). $\mathcal{MC}$ subtracts the predictor to the current block to obtain the prediction residual, which is then processed with transform coding.

Regardless of the coding mode, transform coding plays a central role in any video coding architecture. The input block, or the corresponding prediction residual, is transformed (typically using an orthonormal transform) and the transform coefficients $\mathbf{y}$ are quantized with $\mathcal{Q}$ to obtain $\hat{\mathbf{y}}$. The encoded bitstream is generated by entropy coding the quantized transform coefficients. Since entropy coding is perfectly lossless, it does not leave any footprint, and therefore it is omitted from the scheme in Figure 1.

To avoid drift, the encoder embeds the decoder, which reconstructs video frames in the pixel domain, so that they can be used as reference in inter-frame coding and intra prediction. Decoded blocks $\hat{\mathbf{x}}$ are obtained applying the inverse transform $\mathbf{T}^{-1}$ to $\hat{\mathbf{y}}$, and adding back the predictor (if needed). Finally, pixel values are rounded to unsigned integers. Rounding can be considered as a second quantization step, applied in the pixel domain. However, since the distortion introduced in this case is far less pronounced than the one due to the quantizer $\mathcal{Q}$ applied in the transform domain, it is omitted from the scheme in Figure 1.

A decoder is able to reconstruct a video sequence in the pixel domain from the received compressed bitstream. In

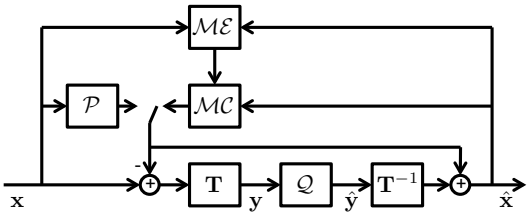


Fig. 1: Simplified block diagram of a conventional video encoder. $\mathbf{T}$ is an orthonormal transform, $\mathcal{Q}$ is a quantizer, $\mathcal{ME}$ and $\mathcal{MC}$ perform, respectively, motion estimation and compensation, and $\mathcal{P}$ computes the spatial prediction in intra-coded frames.

addition, the decoder can read from the bitstream some of the configuration parameters adopted by the encoder. In some cases these parameters can be obtained directly from parsing the bitstream. This is the case, for example, of the GOP structure, the quantization parameters, the coding modes, the motion vectors, etc. Instead, in other cases, the configuration of the encoder can only be determined in an indirect way. For example, it might be possible to infer the adopted motion estimation algorithms from the analysis of the motion vectors [27], or the rate control algorithm from the analysis of the temporal variation of the quantization parameter [23]. However, when the decoded video sequence is re-compressed, most of the information related to the first coding step is seemingly lost, since the analysis of the bitstream reveals details of the second coding step only.

In this paper we consider the set-up illustrated in Figure 2, in which a video sequence $\mathbf{X}$ is encoded twice, namely by codec $\mathtt{c}_1$, obtaining $\hat{\mathbf{X}}_1$, followed by $\mathtt{c}_2$, obtaining $\hat{\mathbf{X}}_2$. As argued in Section I, the first coding step is often performed at the time the sequence $\mathbf{X}$ is acquired. Instead, the second coding step might be applied after manipulating the sequence $\hat{\mathbf{X}}_1$, or when uploading $\hat{\mathbf{X}}_1$ to a video sharing web site. In this paper, we aim at determining the characteristics of the first codec, $\mathtt{c}_1$, given that we observe only the bitstream at the output of the second encoder. Specifically, we assume that $\mathtt{c}_1$ belongs to a group of candidate coding architectures, in which each architecture is the archetypal for a specific video coding standard. In addition, we are particularly interested in determining the GOP size, i.e., the number of frames between two consecutive I-frames, adopted by $\mathtt{c}_1$.

In general, $\mathtt{c}_1$ and $\mathtt{c}_2$ need not to be the same. Indeed: i) they might belong to two different coding architectures; ii) they might follow the same architecture, but represent two different implementations of the same standard; iii) they might be the very same codec, but configured using different encoding parameters, e.g., adopt two different GOP structures, target distortions, etc. In all cases, $\mathtt{c}_2$ acts as a sort of noise source, by masking the footprints left by $\mathtt{c}_1$. The amount of distortion introduced by the second coding step determines to what extent we are able to reveal the traces of $\mathtt{c}_1$. In particular, when the distortion is strong enough, e.g., when the second codec operates at low bitrates, the footprints left by $\mathtt{c}_1$ might not be identified. This will be thoroughly discussed in the experiments presented in Section VI. In the next section, we present an analysis of the footprint left by quantization, which provides the basis for the video coding identification algorithm described in Section V.

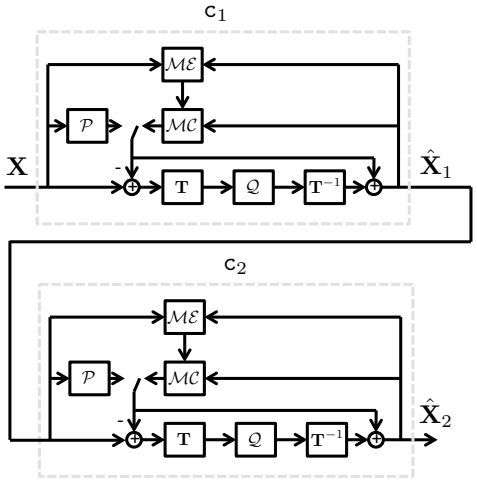


Fig. 2: Simplified block diagram representing a processing chain with two codecs, $\mathtt{c}_1$ and $\mathtt{c}_2$. A video sequence $\mathbf{X}$ is encoded and decoded to $\hat{\mathbf{X}}_1$, then re-encoded and decoded to $\hat{\mathbf{X}}_2$. Each codec (i.e., $\mathtt{c}_1$ and $\mathtt{c}_2$) is represented by a simplified block diagram as in Figure 1.

IV. Analysis of quantization footprints

The proposed method is based on the analysis of coding footprints left by $\mathtt{c}_1$. To this end, we exploit the idempotency property of scalar quantization, and show how this can be somewhat extended to video coding. Scalar quantization is, by construction, an idempotent operator. This means that it is possible to re-iterate quantization multiple times on the same scalar value, and the result will always be the same as the one obtained by applying quantization only once.

More formally, let us consider a scalar value $x \in \mathbb{R}$, which is quantized to $\hat{x}_{q_1}$:

$$\hat{x}_{q_1} = Q_{\Delta_1}(x) = \Delta_1 \left\lfloor \frac{x}{\Delta_1} \right\rfloor, \tag{1}$$

where Δ_1 denotes the quantization step size used, and the subscript q_i denotes a signal quantized i times. If we re-quantize $\hat{x}_{q_1}$, we obtain

$$\hat{x}_{q_2} = Q_{\Delta_2}(\hat{x}_{q_1}) = \Delta_2 \left\lfloor \frac{\Delta_1 \left\lfloor \frac{x}{\Delta_1} \right\rfloor}{\Delta_2} \right\rfloor. \tag{2}$$

When we use the same quantizer and the same step size, i.e., $\Delta_1 = \Delta_2$, then $\hat{x}_{q_2} = \hat{x}_{q_1}$. Hence, we can re-iterate quantization, without affecting the signal after the first quantization.

The idempotent property can be conveniently exploited to identify the quantizer used to produce a set of observed scalar values, when one is given a finite set of candidate quantizers, each one of the form in (1), but characterized by a different step size $\Delta \in \mathcal{S}$. Specifically, let $\{x^1, \ldots, x^P\}$ denote a set of (unobserved) scalar values and $\{\hat{x}^1_{q_1}, \ldots, \hat{x}^P_{q_1}\}$ denote a set of (observed) quantized values according to (1). Then, the quantization footprints can reveal the quantization step size

$$\hat{\Delta}_1 = \max \left\{ \underset{\Delta \in \mathcal{S}}{\operatorname{argmin}} \sum_{j=1}^{P} |\hat{x}^j_{q_1} - Q_\Delta(\hat{x}^j_{q_1})| \right\}, \tag{3}$$

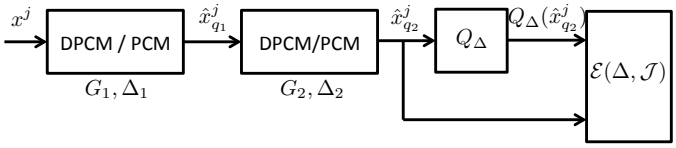


Fig. 3: Block diagram of double DPCM/PCM compression.

where $\bar{\Delta} = \underset{\Delta}{\operatorname{argmin}}\, f(\Delta)$ returns the argument $\bar{\Delta}$ such that $f(\bar{\Delta})$ is the minimum of function $f(\Delta)$. The $\max\{\cdot\}$ operator is needed since its argument might contain more than one quantizer compatible with the observed values. This might be the case, e.g., when $\mathcal{S}$ contains integer multiples or sub-multiples of Δ_1. However, as proved in [17], $\hat{\Delta}_1 = \Delta_1$, provided that we observe a large enough number of values.

In case of double compression, it is not possible to observe the output of the first quantizer, $\{\hat{x}^1_{q_1}, \ldots, \hat{x}^P_{q_1}\}$. However, when $\Delta_2 < \Delta_1$, it is still possible to reveal the first quantization step size. Let us consider a slightly more elaborate model illustrated in Figure 3, which represents a simplified version of double video compression. Let $X = \{x^1, x^2, \ldots, x^P\}$, $x^j \in \mathbb{R}$, denote a 1-dimensional sequence of samples. Differential Pulse Code Modulation (DPCM) is an efficient lossy coding technique that is used to exploit the inter-sample statistical dependency. Instead of quantizing the samples directly as in the case of scalar quantization described in (1) (a.k.a. Pulse Code Modulation (PCM)), the prediction residual is quantized:

$$\hat{e}^j_{q_1} = Q_{\Delta_1}(e^j) = Q_{\Delta_1}(x^j - \mathcal{P}(x^j)), \tag{4}$$

where $\mathcal{P}(x^j)$ denotes a predictor of the sample x^j. In the simplest case, the previously decoded sample is used as predictor, i.e., $\mathcal{P}(x^j) = \hat{x}^{j-1}_{q_1}$. The sample is then reconstructed by adding back the quantized residual to the predictor:

$$\hat{x}^j_{q_1} = \mathcal{P}(x^j) + \hat{e}^j_{q_1}. \tag{5}$$

When using DPCM, it is customary to periodically insert samples which are PCM coded. Let G_1 denote the number of samples between two consecutive PCM-coded samples and $\hat{X}_1 = \{\hat{x}^1_{q_1}, \ldots, \hat{x}^P_{q_1}\}$ the sequence reconstructed after the first coding step. $\hat{X}_1$ is then re-encoded with DPCM to obtain $\hat{X}_2$, with quantization step size Δ_2 and a period of G_2 between PCM-coded samples.

If G_1 is known, it is possible to focus on those samples which were originally PCM-coded in the first coding step, i.e., $\hat{x}^j_{q_2}$, $j \in \mathcal{J}$, $\mathcal{J} = \{\text{mod}(j-1, G_1) = 0\}$. In this case, although the samples produced by the first coding step are not directly observed, it is still possible to estimate the quantization step size. Indeed, the decoded samples after the second coding step can be written as

$$\hat{x}^j_{q_2} = \mathcal{P}(\hat{x}^j_{q_1}) + \hat{e}^j_{q_2} = \mathcal{P}(\hat{x}^j_{q_1}) + Q_{\Delta_2}(\hat{x}^j_{q_1} - \mathcal{P}(\hat{x}^j_{q_1})) \tag{6}$$

Equation (6) states that $\hat{x}^j_{q_2}$ is obtained by shifting $\hat{x}^j_{q_1}$ by an amount equal to the predictor $\mathcal{P}(\hat{x}^j_{q_1})$, quantizing the result according to $Q_{\Delta_2}(\cdot)$, and shifting back the result by $\mathcal{P}(\hat{x}^j_{q_1})$. For any possible input $\hat{x}^j_{q_1}$ to the second coding step, the output $\hat{x}^j_{q_2}$ is equal to the input, plus an offset that depends on the quantization error introduced on the residual. More formally,

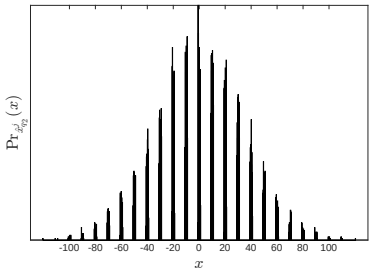


Fig. 4: Histogram of the samples $\hat{x}^j_{q_2}$, $j \in \mathcal{J}$, when $\Delta_1 = 10$ and $\Delta_2 = 4$.

by adding and removing the term $\hat{x}^j_{q_1} - \mathcal{P}(\hat{x}^j_{q_1})$ to equation (6), it simplifies to

$$\begin{aligned} \hat{x}^j_{q_2} &= \mathcal{P}(\hat{x}^j_{q_1}) + Q_{\Delta_2}(\hat{x}^j_{q_1} - \mathcal{P}(\hat{x}^j_{q_1})) + (\hat{x}^j_{q_1} - \mathcal{P}(\hat{x}^j_{q_1})) \\ &\quad -(\hat{x}^j_{q_1} - \mathcal{P}(\hat{x}^j_{q_1})) \\ &= \hat{x}^j_{q_1} + Q_{\Delta_2}(\hat{x}^j_{q_1} - \mathcal{P}(\hat{x}^j_{q_1})) - (\hat{x}^j_{q_1} - \mathcal{P}(\hat{x}^j_{q_1})) \\ &= \hat{x}^j_{q_1} + (Q_{\Delta_2}(e^j_{q_1}) - e^j_{q_1}) \\ &= \hat{x}^j_{q_1} + \eta^j, \end{aligned} \tag{7}$$

where η^j is the quantization error introduced on the residual $e^j_{q_1} = \hat{x}^j_{q_1} - \mathcal{P}(\hat{x}^j_{q_1})$.

It is interesting to determine the statistical distribution of $\hat{x}^j_{q_2}$, which depends on both the distribution of $\hat{x}^j_{q_1}$ and η^j. The form of the probability density function (p.d.f.) of $\hat{x}^j_{q_1}$, $j \in \mathcal{J}$, is determined by the first quantizer:

$$\Pr_{\hat{x}^j_{q_1}}(x) \propto \sum_k w_k \delta(x - k\Delta_1), \tag{8}$$

where $\delta(x) = 1$ if $x = 0$, and equal to zero otherwise. The values $w_k \in \mathbb{R}$ depend on the p.d.f. of the original sequence, and they are not relevant to the present discussion. The residual is represented as a random variable, whose p.d.f. is $\Pr_{e^j_{q_1}}(x)$. If $\Pr_{e^j_{q_1}}(x)$ is smooth and its spread (e.g., measured by its standard deviation) is large when compared to Δ_2, then the p.d.f. of the quantization error on the residual is uniform in the interval $[-\Delta_2/2, \Delta_2/2]$:

$$\Pr_{\eta^j}(x) \propto \text{rect}\left(\frac{x}{\Delta_2}\right), \tag{9}$$

where $\text{rect}(x) = 1$ if $x \in [-1/2, +1/2]$ and zero otherwise. Assuming statistical independence between $\hat{x}^j_{q_1}$ and η^j, the p.d.f. of the output of the second coding step can be written as

$$\Pr_{\hat{x}^j_{q_2}}(x) \propto \sum_k w_k \text{rect}\left(\frac{x - k\Delta_1}{\Delta_2}\right). \tag{10}$$

As an example, Figure 4 shows the empirical distribution $\Pr_{\hat{x}^j_{q_2}}(x)$ obtained by applying double DPCM coding to a 1-dimensional source and setting $\Delta_1 = 10$ and $\Delta_2 = 4$.

Similarly to the case of scalar quantization in (3), when $\Delta_2 < \Delta_1$, it is possible to estimate the quantization step size of the first coding step by re-quantizing the observed samples by varying Δ, and observing the behavior of the cost function

$$\mathcal{E}(\Delta, \mathcal{J}) = \sum_{j \in \mathcal{J}} |\hat{x}^j_{q_2} - Q_\Delta(\hat{x}^j_{q_2})|. \tag{11}$$

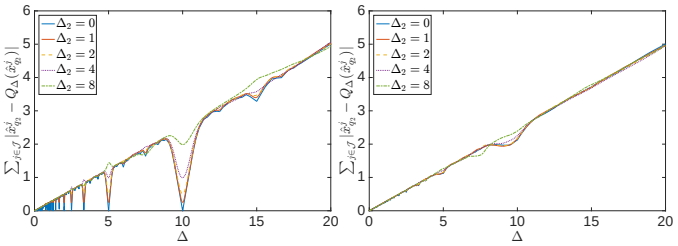


Fig. 5: Behavior of $\mathcal{E}(\Delta, \mathcal{J}_{g_1})$, when $\Delta_1 = 10$. When $g_1 = G_1$ (a) strong traces revealing $\Delta_1 = 10$ are present, while when $g_1 \neq G_1$ (b) these traces are not so evident (best seen in colours).

Figure 5a shows the cost function in (11), when varying Δ, for the case $\Delta_1 = 10$. It is possible to clearly observe the footprints left by the first coding step, as local minima when $\Delta = \Delta_1$ and its sub-multiples.

So far, we have assumed that G_1 is known. In practice, this is not the case, since only the output of the second coding step is observed. However, it is possible to compute $\mathcal{E}(\Delta, \mathcal{J}_{g_1})$ for different hypotheses of the distance between consecutive PCM-coded samples, where $\mathcal{J}_{g_1} = \{\text{mod}(j-1, g_1) = 0 \wedge \text{mod}(j-1, G_2) \neq 0\}$, $g_1 = 2, 3, \ldots$, and select the smallest value of g_1 for which $\mathcal{E}(\Delta, \mathcal{J}_{g_1})$ exhibits the characteristic shape shown in Figure 5a. Indeed, when $g_1 \neq G_1$, the cost function is of the form shown in Figure 5b.

In the next section we show that similar principles can be followed to identify both the video codec and the GOP size used by the first coding step.

V. Video Codec and GOP size identification

The proposed method for the identification of the video codec and of the GOP size follows a recompress-and-observe paradigm, i.e., re-encoding the video under analysis so as to find a coding configuration that matches the one used in the first coding step. More precisely, let us consider a double-compression chain such as the one illustrated in Figure 2. A video sequence $\mathbf{X}$ is encoded by $\mathtt{c}_1$ and decoded in the pixel domain to produce $\hat{\mathbf{X}}_1$. Then, $\hat{\mathbf{X}}_1$ is re-encoded with $\mathtt{c}_2$ and decoded to $\hat{\mathbf{X}}_2$. In the case of DPCM/PCM coding, we showed in Section IV that the quantization footprints left by the first coding step were clearly identifiable when $\Delta_2 < \Delta_1$. Similarly, in the case of video coding, we assume that $\mathtt{c}_2$ is configured to operate in such a way that the distortion introduced in $\hat{\mathbf{X}}_2$ is less than, or equal to, the one introduced by $\mathtt{c}_1$, so as to avoid masking completely the traces left by $\mathtt{c}_1$. This situation commonly arises in real-world scenarios. For example, this is the case of a video that is edited, and then recompressed using coding parameters that retain the same visual quality as the input. For example, when the same codec is used for both the first and the second compression, the target bitrate of the latter step is chosen to be at least as large as the bitrate of the former. This implies the use of a finer quantizer in the second coding step.

The proposed method receives as input the bitstream after the second encoder $\mathtt{c}_2$, decodes it to $\hat{\mathbf{X}}_2$ and re-encodes this

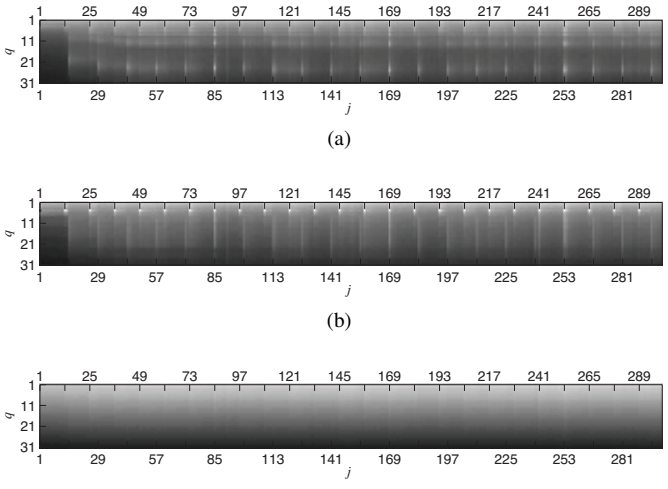


Fig. 6: Example of matrices $\mathbf{P}_{\mathtt{c}_3}(q, j)$ obtained for the sequence NEWS originally encoded with MPEG-2 (GOP size: 14) and re-encoded with MPEG-4 (GOP size: 12). (a) $\mathtt{c}_3$ = MPEG-2; (b) $\mathtt{c}_3$ = MPEG-4; (c) $\mathtt{c}_3$ = AVC.

sequence using a third codec, $\mathtt{c}_3$, obtaining $\hat{\mathbf{X}}_3$. When $\mathtt{c}_3$ matches the same codec and configuration parameters as $\mathtt{c}_1$, we expect that $\hat{\mathbf{X}}_3 \simeq \hat{\mathbf{X}}_2$, i.e., the output of the third coding step is similar to its input, due to the idempotent property. In the case of video, the similarity between sequences is computed by means of the Peak Signal to Noise Ratio (PSNR). We enumerate different codecs $\mathtt{c}_3 \in \mathcal{C}$, each with different encoding parameters, and we look for the one that leads to the highest similarity. Concretely, since the GOP size of the first coding step is unknown, we configure $\mathtt{c}_3$ to operate using intra-frame coding only (I-frames), and repeat the encoding using different quantization parameters (QP) $q \in \mathcal{QP}$. As such, we only need to loop over two parameters (i.e., codec and QP), repeating the encoding $|\mathcal{C}| \times |\mathcal{QP}|$ times and, for each $(\mathtt{c}_3, q)$ pair, we compute the PSNR between $\hat{\mathbf{X}}_2$ and $\hat{\mathbf{X}}_3$ on a frame-by-frame basis. As an example, Figure 6 shows three matrices containing the values of PSNR obtained when analyzing a double-compressed video sequence, which was compressed with MPEG-2 (GOP size 14) and MPEG-4 (GOP size 12) in the first and second coding step, respectively. Each matrix corresponds to a different codec, namely MPEG-2, MPEG-4 and AVC. By inspecting these matrices, we observe the following:

- I-frames in $\hat{\mathbf{X}}_1$ re-encoded as I-frames in $\hat{\mathbf{X}}_3$ result in higher PSNR values with respect to those obtained for P-frames re-encoded as I-frames. As a consequence, a periodic pattern arises, whose period is equal to the GOP size of the first coding step.
- When codec $\mathtt{c}_3$ matches codec $\mathtt{c}_1$, we can analyze how the PSNR varies when changing the QP for those frames which were originally encoded as I-frames in the first coding step. This is equivalent to looking at the columns of the matrix for which $\mathtt{c}_3 = \mathtt{c}_1$ corresponding to the I-frames. This is better illustrated in Figure 7a, in which it is possible to observe that the PSNR vs. QP curve exhibits a local maximum corresponding to the QP value

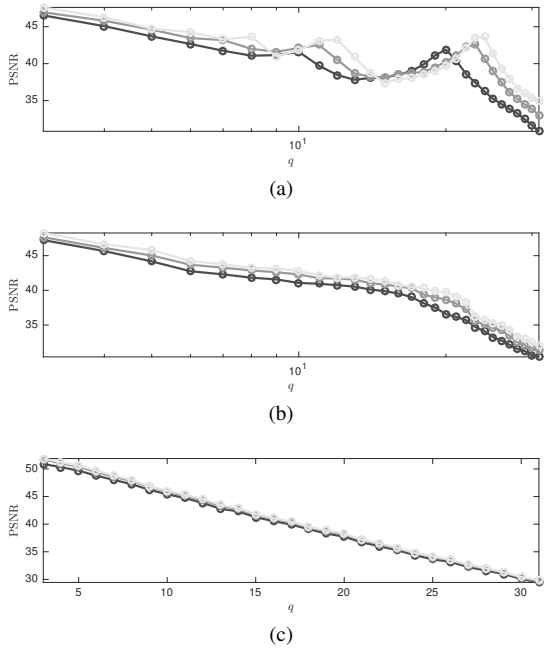


Fig. 7: PSNR vs. q curves for different I-frames of the first GOP. (a) $\mathtt{c}_3$ (MPEG-2) matches $\mathtt{c}_1$; (b) $\mathtt{c}_3$ (MPEG-4) does not match $\mathtt{c}_1$; (c) $\mathtt{c}_3$ (AVC) does not match $\mathtt{c}_1$. Note that the q scale is logarithmic in the top and middle figures to take into account the different mapping between the quantization parameter and the quantization step size in MPEG-2, MPEG-4 and AVC.

originally used by $\mathtt{c}_1$ to encode the frame. Conversely, when codec $\mathtt{c}_3$ does not match codec $\mathtt{c}_1$, the PSNR vs. QP function is smooth, as illustrated in Figure 7b and Figure 7c.

Starting from the above observations, we propose the following identification algorithm in order to detect the codec used by $\mathtt{c}_1$ and its GOP size:

1) *Initialization*: Select the set of candidate codecs $\mathcal{C}$ and the set of quantization parameters $\mathcal{QP} = [QP_{min}, QP_{max}]$. QP_{min} is set equal to the minimum value accepted by the codec that gives the highest quality, while QP_{max} is set to a value that is not greater than the QP value used by $\mathtt{c}_2$ (i.e., the maximum QP value we are able to detect).
2) *Recompress*: For each $\mathtt{c}_3 \in \mathcal{C}$,
   a) Encode $\hat{\mathbf{X}}_2$ with $\mathtt{c}_3$ using all QP values and intra-frame coding mode only. Let $\hat{\mathbf{X}}_3^q$, $q \in \mathcal{QP}$ denote the output of $\mathtt{c}_3$ decoded in the pixel domain.
   b) Compute the matrix $\mathbf{P}_{\mathtt{c}_3}$, whose entries are given by

$$\mathbf{P}_{\mathtt{c}_3}(q,j) = \text{PSNR}(\hat{\mathbf{X}}_2(j), \hat{\mathbf{X}}_3^q(j)), \tag{12}$$

   i.e., the PSNR value computed comparing the j-th frame of $\hat{\mathbf{X}}_2$ and of $\hat{\mathbf{X}}_3^q$.
   c) Since the bitstream used to decode $\hat{\mathbf{X}}_2$ with $\mathtt{c}_2$ is assumed to be available, the matrix $\mathbf{P}_{\mathtt{c}_3}$ is processed column-wise to remove the traces left by $\mathtt{c}_2$, for which the quantization parameter is known. To this end we compute

$$\mathbf{P}'_{\mathtt{c}_3}(q,j) = \begin{cases} \frac{\mathbf{P}_{\mathtt{c}_3}(q-1,j)+\mathbf{P}_{\mathtt{c}_3}(q+1,i)}{2} & q = QP_{\mathtt{c}_2} \\ \mathbf{P}_{\mathtt{c}_3}(q,j) & \text{otherwise,} \end{cases} \tag{13}$$

   In doing so, it is possible to remove potential local maxima obtained if the quantization parameter of $\mathtt{c}_3$ matches the one of $\mathtt{c}_2$ instead of the one of $\mathtt{c}_1$.
   d) To enhance the local maxima due to matching parameters between $\mathtt{c}_3$ and $\mathtt{c}_1$, we compute the matrix

$$\mathbf{P}''_{\mathtt{c}_3} = \mathbf{P}'_{\mathtt{c}_3} - \text{medfilt}(\mathbf{P}'_{\mathtt{c}_3}), \tag{14}$$

   where $\text{medfilt}(\cdot)$ is a median filter applied row-by-row to enhance peaks due to I-frames in $\hat{\mathbf{X}}_1$. An example is shown in Figure 8, which illustrates $\mathbf{P}''_{\mathtt{c}_3}$ and $\mathbf{P}'_{\mathtt{c}_3}$ for comparison.
3) *Estimate the GOP size*: Many methods for estimating the periodicity of a signal have been proposed in the literature. These range from historical frequency estimation algorithms [33], [34] to more modern methods for tempo estimation [35]. However, due to the nature of the analyzed signals, in the following we propose a simple yet effective method for periodicity estimation that enables GOP detection at reduced complexity.
   a) For each $\mathtt{c}_3 \in \mathcal{C}$,
      i) Compute the absolute value of Fourier transform of each row of $\mathbf{P}''_{\mathtt{c}_3}$. Then, average the result along each column to obtain

$$\mathbf{g}_{\mathtt{c}_3} = \frac{1}{|\mathcal{QP}|} \sum_{q \in \mathcal{QP}} |\mathcal{F}\{\mathbf{P}''_{\mathtt{c}_3}(q,\cdot)\}|, \tag{15}$$

      where $\mathcal{F}(\cdot)$ denotes the Fourier transform and $\mathbf{g}_{\mathtt{c}_3}$ is a row vector whose number of elements is equal to the number of frames.
      ii) Use a filterbank to evaluate the periodicity of the peaks in $\mathbf{g}_{\mathtt{c}_3}$, which is an estimate of the GOP size. More specifically we compute the average energy obtained filtering $\mathbf{g}_{\mathtt{c}_3}$ with different filters tuned to different candidate GOP sizes:

$$\hat{G}_{1,\mathtt{c}_3} = \underset{G}{\text{argmax}} \sum_{j=1}^{N} |(\mathbf{g}_{\mathtt{c}_3} * \mathbf{h}_G)(i)|^2, \tag{16}$$

      where $\bar{G} = \underset{G}{\text{argmax}}\, f(G)$ returns the argument $\bar{G}$ such that $f(\bar{G})$ is the maximum of function $f(G)$ and $\mathbf{h}_G$ is a comb-filter whose distance between consecutive peaks is equal to G . That is

$$\mathbf{h}_G(j) = \sum_k \delta(j - kG) \tag{17}$$

      iii) Compute as quality metric of $\mathbf{g}_{\mathtt{c}_3}$ its peakness value defined as

$$p_{\mathtt{c}_3} = \frac{\max(\mathbf{g}_{\mathtt{c}_3})}{\text{mean}(\mathbf{g}_{\mathtt{c}_3})}, \tag{18}$$

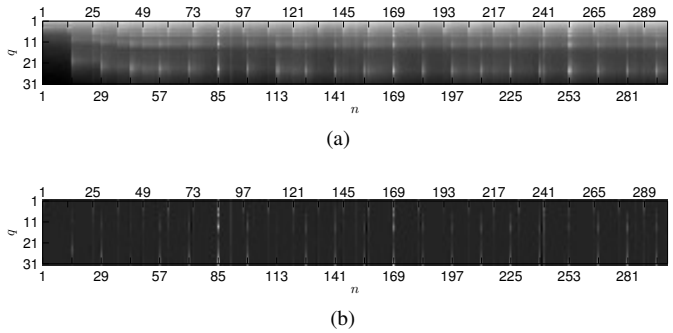


Fig. 8: Effect of median filtering on $\mathbf{P}'_{\mathrm{c}_3}$ realization from Figure 6. (a) $\mathbf{P}'_{\mathrm{c}_3}$; (b) $\mathbf{P}''_{\mathrm{c}_3}$.

where $\max(.)$ computes the maximum value and $\mathrm{mean}(.)$ computes the average value.

b) Estimate the GOP size $\hat{G}_1$ as the $\hat{G}_{1,\mathrm{c}_3}$ with the highest associated p_{c_3}. More formally,

$$\hat{G}_1 = \hat{G}_{1,\bar{\mathrm{c}}_3}, \tag{19}$$

where $\bar{\mathrm{c}}_3 = \underset{\mathrm{c}_3}{\mathrm{argmax}}\,(p_{\mathrm{c}_3})$. Notice that $p_{\bar{\mathrm{c}}_3}$ can be used as a confidence value of the GOP estimation. Indeed, high $p_{\bar{\mathrm{c}}_3}$ values are associated to GOPs estimated from very peaky (thus reliable and easier to analyze) $\mathbf{g}_{\bar{\mathrm{c}}_3}$. GOP estimation associated to $p_{\bar{\mathrm{c}}_3}$ values below a confidence threshold can be discarded (as shown in Section VI).

4) *Identify the codec*:

a) For each $\mathrm{c}_3 \in \mathcal{C}$, compute the cost function

$$J_{\mathrm{c}_3} = \sum_{j \in \mathcal{J}} \sum_{q \in \mathcal{QP}} \mathbf{P}''_{\mathrm{c}_3}(q, j), \tag{20}$$

where $\mathcal{J} = \{j | \mathrm{mod}(j-1, \hat{G}_1) = 0\}$. Equation (20) computes the sum of median-filtered PSNR values for those frames corresponding to the original I-frames.

b) Identify the codec as follows:

$$\hat{\mathrm{c}}_1 = \underset{\mathrm{c}_3 \in \mathcal{C}}{\mathrm{argmax}}\, J_{\mathrm{c}_3}. \tag{21}$$

Notice that the knowledge of the bitstream after c_2 is used only to extract the GOP and QP values of $\hat{\mathbf{X}}_2$. However, the knowledge of the bitstream is not such a strict hypothesis, since videos are usually distributed in compressed format and not already decoded (especially when dealing with user-generated content). Nonetheless, if $\hat{\mathbf{X}}_2$ is available only in the pixel domain, it is still possible to apply the algorithm skipping step 2c with reduced accuracy.

VI. Experimental results

In order to validate the proposed method, in this Section we present a set of results obtained using a dataset composed of a wide set of video sequences encoded with different parameters. More specifically we focus on studying the accuracy of the GOP estimation and the codec detection algorithms for different sequences (seq), coding standards applied during the first coding step (c_1), bitrates of c_1 (R_1), GOP sizes of c_1 (G_1), coding standards applied during the second coding step (c_2), QPs of c_2 (QP_2), and GOP sizes of c_2 (G_2). Notice that, to test the effect of all these parameters on the presented algorithm, we need a complete knowledge of the coding history of each tested sequence. For this reason we performed this validation step on a controlled set of videos we encoded on purpose starting from raw material, rather than on random videos downloaded from the web.

TABLE I: List of parameters names and values used to build the dataset. Number of configurations for each parameter is also provided.

Name	Value							N. conf.
seq	CIF *Foreman*, *Mobile*, *Paris*, *News* 4CIF *Ice*, *Harbour*							6
c_1	MPEG-2 (libavcodec[1]) MPEG-2 MPEG-4 Part 4 (libavcodec) MPEG-4(a) MPEG-4 Part 4 (Microsoft variant[2]) MPEG-4(b) H.264/AVC (libavcodec) w/o deblock AVC(a) H.264/AVC (libavcodec) with deblock AVC(b) H.264/AVC 10 (JM[3]) w/o deblock AVC(c) DIRAC (libschrodinger[4]) DIRAC							7
R_1		R_L	R_M	R_H				3
	CIF	30 dB	33 dB	36 dB				
	4CIF	34 dB	37 dB	40 dB				
G_1	5, 7, 9, 11, 13, 14, 15, 17, 19, 21, 23							11
c_2	MPEG-2 (libavcodec) MPEG-2 MPEG-4 Part 4 (libavcodec) MPEG-4(a) H.264/AVC (libavcodec) AVC(a)							3
QP_2		a	b	c	d	e	f	6
	MPEG-2/4	1	2	4	5	7	10	
	AVC	10	20	23	36	29	32	
G_2	12							1

Table I reports all the used parameters values. We started from six well known uncompressed video sequences of 10 seconds each (i.e., 300 frames approximately), with different spatial and temporal information: four at CIF spatial resolution (352×288), namely *Foreman*, *Mobile*, *Paris*, *News*; two at 4CIF spatial resolution (704×576), namely *Ice* and *Harbour*. For the first coding step we used MPEG-2, MPEG-4 part 4 (MPEG-4), MPEG-4 part 10 H.264/AVC (AVC), and DIRAC. As it regards MPEG-4, we tested two different implementations (MPEG-4(a) and MPEG-4(b)), while for AVC we used two different implementations (AVC(a) and AVC(c)), one of which was tested with the in-loop filter enabled too (AVC(b)). For each codec, we selected three different target bitrates by enabling rate control in order to obtain three sequences at *low*, *medium* and *high* quality, respectively. The mapping between bitrates and the obtained PRNUs is reported in Table I. As for the second coding step, we re-encoded all sequences with either MPEG-2, MPEG-4 or AVC, using a constant QP (i.e., a constant quantization step). In order to unify the notation, the set of possible QP values for c_2 is identified with $\{a, b, c, d, e, f\}$, which corresponds to $\{1, 2, 4, 5, 7, 10\}$ for MPEG-2/4 codecs and to $\{10, 20, 23, 26, 29, 32\}$ for AVC (equalizing the value of quantization steps among the codecs). Concerning the GOP used for the first and second coding steps, we used 10 different combinations reported in Table I. Notice that the algorithm depends on the ratio between the used GOPs (G_1/G_2), rather than on the exact GOP value. This justifies the fact that we fixed G_2 while changing G_1 only. Concerning the third coding step (i.e., the analysis one), we used as c_3 MPEG-2, MPEG-4(a), AVC(a) and DIRAC.

TABLE II: $a_1(QP_2, G_1/G_2)$ fixing $R_1 = R_M$. Bold is used for accuracy values larger than 0.5.

QP_2 \ G_1/G_2	5/12	7/12	9/12	11/12	13/12	15/12	17/12	19/12	21/12	23/12
a	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**0.89**	**0.99**	**0.90**	**0.78**	**0.78**
b	**1.00**	**1.00**	**0.99**	**0.99**	**0.97**	**0.78**	**0.90**	**0.81**	**0.71**	**0.75**
c	**1.00**	**0.99**	**0.94**	**0.85**	**0.79**	**0.64**	**0.64**	**0.58**	**0.57**	0.46
d	**0.99**	**0.96**	**0.78**	**0.72**	**0.64**	0.47	0.49	0.36	0.42	0.33
e	**0.92**	**0.83**	**0.51**	0.46	0.44	0.29	0.28	0.22	0.21	0.21
f	**0.78**	**0.62**	0.25	0.31	0.28	0.21	0.14	0.11	0.12	0.08

GOP Estimation

Let us consider a sequence (seq) double encoded with the parameters $\mathtt{c}_1, R_1, G_1, \mathtt{c}_2, QP_2$ and G_2. We define the detection function as

$$d(\text{seq}, \mathtt{c}_1, R_1, G_1, \mathtt{c}_2, QP_2, G_2) = \begin{cases} 1, & \text{if } \hat{G}_1 = G_1, \\ 0, & \text{otherwise,} \end{cases} \quad (22)$$

whose value is 1 if the estimated GOP $\hat{G}_1$ is correct and 0 otherwise. To analyze the behavior of the GOP estimation algorithm under different conditions and its robustness to various coding parameters, we average the detection function (22) along different dimensions, aggregating results obtained on a subset of the dataset.

To highlight the effect of the distortion introduced by $\mathtt{c}_2$ quantization (QP_2) and different GOP ratios (G_1/G_2) on GOP estimation, we define the accuracy as

$$a_1(QP_2, G_1/G_2) = \underset{\text{seq}, \mathtt{c}_1, R_1, \mathtt{c}_2}{\text{average}} \left[d(seq, \mathtt{c}_1, R_1, G_1, \mathtt{c}_2, QP_2, G_2) \right], \quad (23)$$

where $\underset{x}{\text{average}}\left[d(x)\right]$ computes the average value of d along the x direction and $a_1 \in [0, 1]$ (0 for GOP never correctly detected, 1 for GOP always correctly detected). Table II shows the behavior of $a_1(QP_2, G_1/G_2)$ fixing the average quality of $\mathtt{c}_1$ ($R_1 = R_M$) with $\mathtt{c}_1 \in$ {MPEG-2,MPEG-4(a), AVC(a), DIRAC}, $\mathtt{c}_2 \in$ {MPEG-2, MPEG-4(a), AVC(a)} and $G_2 \in \{5, 7, 9, 11, 13, 15, 17, 19, 21, 23\}$ (i.e., $6 \times 4 \times 1 \times 10 \times 3 \times 6 \times 1 = 4320$ sequences). As expected, when QP_2 is low, the quantization noise introduced by $\mathtt{c}_2$ does not mask the artifacts introduced by $\mathtt{c}_1$, thus allowing an accurate GOP estimation. The same trend can be observed for low values of G_1/G_2. Indeed, since all the sequences share the same number of frames, a low G_1 values determines more peaks in $\mathbf{g}_{\mathtt{c}_3}$ (see (15)), thus making easier to detect the periodicity of the peaks (i.e., G_1). If the quality of $\mathtt{c}_1$ is decreased ($R_1 = R_L$) or increased ($R_1 = R_H$), the artifacts introduced by $\mathtt{c}_1$ become stronger or weaker, respectively. This effect determines an average accuracy increase of 5.8% for $R_1 = R_L$ and an average accuracy decrease of 13.6% for $R_1 = R_H$ with respect to the case $R_1 = R_M$ shown in Table II.

In order to study the behavior of the GOP estimation algorithm for each video sequence, we average the detection function along $\mathtt{c}_1$, R_1, $\mathtt{c}_2$, G_1 and G_2 defining the accuracy as

$$a_2(QP_2, \text{seq}) = \underset{\mathtt{c}_1, R_1, G_1, \mathtt{c}_2, G_2}{\text{average}} \left[d(seq, \mathtt{c}_1, R_1, G_1, \mathtt{c}_2, QP_2, G_2) \right], \quad (24)$$

[1]https://libav.org/

[2]http://ffmpeg.org/~michael/msmpeg4.txt

[3]http://iphome.hhi.de/suehring/tml

[4]http://diracvideo.org/

TABLE III: $a_2(QP_2, \text{seq})$. We use bold and italics for best and worst results for each QP_2, respectively.

QP_2 \ seq	*Foreman*	*Mobile*	*Paris*	*News*	*Ice*	*Harbour*
a	0.95	**1.00**	0.89	0.93	0.96	*0.87*
b	0.92	**0.97**	0.88	*0.86*	0.96	0.87
c	0.77	0.88	0.81	*0.69*	**0.96**	0.75
d	0.62	0.75	0.68	0.55	**0.78**	*0.31*
e	0.42	**0.63**	0.47	0.39	0.53	*0.18*
f	0.27	**0.54**	0.32	0.27	0.31	*0.04*

TABLE IV: $a_3(\mathtt{c}_1, \mathtt{c}_2)$ values. Bold and italics are used values over 0.8 and under 0.6, respectively.

$\mathtt{c}_1$ \ $\mathtt{c}_2$	MPEG-2	MPEG-4(a)	AVC(a)
MPEG-2	0.70	0.66	**0.88**
MPEG-4(a)	0.72	0.61	**0.81**
AVC(a)	*0.56*	*0.51*	0.66
DIRAC	*0.54*	*0.53*	0.66

whose values still ranges between 0 and 1. Table III reports $a_2(QP_2, \text{seq})$ values obtained on the same 4320 sequences dataset of the previous experiment. These results confirm that, despite the accuracy values slightly differ for each sequence, the general trend remains the same, i.e., the accuracy decreases for increasing QP_2 values as expected. It is worth noting that the sequence providing the best results is Mobile. This is due to the presence of complex textures with constant motion that are prone to emphasize visual artifacts left by $\mathtt{c}_1$, thus making GOP footprints stronger and easier to detect. On the other hand, *Harbour* gives overall the worst results. This is due to the lack of motion of textured blocks.

Another interesting aspect to investigate is the effect of different codecs at the first and second coding steps (i.e., $\mathtt{c}_1$ and $\mathtt{c}_2$). To this purpose we define the accuracy as

$$a_3(\mathtt{c}_1, \mathtt{c}_2) = \underset{\text{seq}, R_1, G_1, QP_2, G_2}{\text{average}} \left[d(seq, \mathtt{c}_1, R_1, G_1, \mathtt{c}_2, QP_2, G_2) \right]. \quad (25)$$

Table IV reports $a_3(\mathtt{c}_1, \mathtt{c}_2)$ values computed on the same dataset of the previous experiments, highlighting the highest and lowest values using bold and italics, respectively. Notice that, on average, the highest accuracy (over 80%) is obtained for $\mathtt{c}_1 \in \{\text{MPEG-2}, \text{MPEG-4(a)}\}$ and $\mathtt{c}_2 = \text{AVC(a)}$, while the lowest accuracy (under 60%) is for $\mathtt{c}_1 \in \{\text{AVC(a)}, \text{DIRAC}\}$ and $\mathtt{c}_2 \in \{\text{MPEG-2}, \text{MPEG-4(a)}\}$. This is an expected behavior, as MPEG-2 and MPEG-4 are older standards than AVC and DIRAC. Hence, MPEG-2 and MPEG-4 leave stronger coding artifacts that, on one hand, enable to easily detect their presence, and, on the other hand, increase the masking power over previous coding steps.

Finally, in order to understand whether it is possible to discriminate between correct and wrong GOP estimations, we studied the relationship between GOP estimation and the

TABLE V: Confusion matrix for c_1 identification with different masking c_2. Bold is used to denote the elements on the diagonal, i.e., elements that should be equal to one in the best scenario.

c_1 \ $\hat{c}_1$	MPEG-2			MPEG-4			AVC			DIRAC		
MPEG-2	**0.94**	**0.96**	**0.96**	0.05	0.04	0.03	0	0	0.01	0.01	0	0
MPEG-4 (a)	0	0	0.04	**0.93**	**0.92**	**0.76**	0.02	0.02	0.2	0.06	0.06	0
MPEG-4 (b)	0.02	0.02	0.06	**0.87**	**0.87**	**0.69**	0.06	0.05	0.25	0.06	0.06	0
AVC (a)	0.01	0	0	0.14	0.24	0.06	**0.79**	**0.68**	**0.94**	0.06	0.08	0
AVC (b)	0	0.01	0	0.13	0.2	0.05	**0.81**	**0.69**	**0.94**	0.06	0.09	0.01
AVC (c)	0.06	0.06	0.15	0	0.03	0	**0.92**	**0.87**	**0.81**	0.02	0.05	0.04
DIRAC	0	0.02	0	0.09	0.12	0.01	0.13	0.12	0.21	**0.78**	**0.74**	**0.78**
c_2	MPEG-2	MPEG-4	AVC	MPEG-2	MPEG-4	AVC	MPEG-2	MPEG-4	AVC	MPEG-2	MPEG-4	AVC

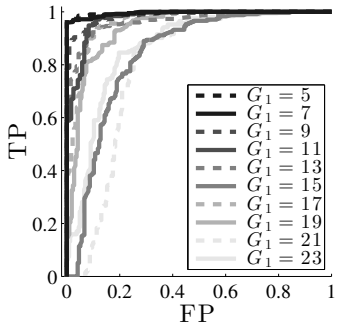


Fig. 9: GOP detection ROC curves for different values of G_1.

peakness value associated to an estimated GOP (18). In other words, we computed the Receiver Operating Characteristic (ROC) curve by thresholding peakness values associated to estimated GOPs. We evaluate as True Positives (TP) the correctly estimated GOPs whose associated peakness is above the threshold, and False Positives (FP) the wrongly estimated GOPs whose associated peakness is above the threshold. Figure 9 shows the ROC curves computed on the same dataset used for the previous experiments, analyzing separately sequences with different G_1 values. Also this experiment confirms that the lower the G_1 value, the easiest is to correctly detect the GOP, as already shown in Table II. Moreover this proves that we can detect with a given probability whether the GOP estimate is to be considered valid or not.

Codec Identification

To analyze the performance of the codec identification algorithm, we consider all the combinations of parameters reported in Table I, fixing the GOP size $G_1 = 14$. The dataset is then composed by 2268 sequences (i.e., six sequences, seven c_1, three R_1, one G_1, three c_2, six QP_2 and one G_2). Since we focus on codec identification, we assume G_1 to be known.

Table V shows the codec identification confusion matrix as a function of the masking codec c_2. The identification method is operated at sequence level by aggregating the observations extracted from all detected intra-coded frames as explained in (21). These results are averaged across all tested sequences. It is interesting to notice that, as highlighted also in the GOP identification procedure, AVC and DIRAC are well masked by MPEG-2 and MPEG-4. Instead, when the masking (c_2) and the masked (c_1) codecs share the same architecture, identification accuracy is increased.

TABLE VI: Codec identification accuracy for different sequences, c_2 and QP_2. Bold is used for values larger than 0.8

(a) c_2 = MPEG-2

seq \ QP_2	a	b	c	d	e	f
Foreman	**1.00**	**0.95**	**0.90**	**0.90**	**0.81**	0.67
Mobile	**1.00**	**1.00**	**0.81**	0.71	0.57	0.62
News	**1.00**	**1.00**	**0.90**	**0.90**	**0.86**	0.76
Paris	**1.00**	**1.00**	**1.00**	**0.90**	**0.86**	0.76
Ice	**0.90**	**0.90**	**0.90**	**0.90**	**0.86**	0.67
Harbour	**0.90**	**0.86**	**0.90**	**0.86**	**0.81**	0.67

(b) c_2 = MPEG-4

seq \ QP_2	a	b	c	d	e	f
Foreman	**0.95**	**0.90**	**0.90**	**0.90**	**0.81**	0.52
Mobile	**1.00**	**0.90**	0.62	0.57	0.71	0.62
News	**1.00**	**0.95**	**0.90**	**0.86**	0.76	0.62
Paris	**1.00**	**1.00**	**0.95**	**0.90**	**0.90**	0.71
Ice	**0.90**	**0.90**	**0.86**	**0.81**	0.76	0.52
Harbour	**0.90**	**0.81**	**0.90**	**0.86**	0.71	0.52

(c) c_2 = AVC

seq \ QP_2	a	b	c	d	e	f
Foreman	**1.00**	**1.00**	**1.00**	**0.95**	**0.90**	**0.81**
Mobile	**1.00**	**1.00**	**0.95**	0.76	0.62	0.57
News	**1.00**	**0.95**	**0.90**	**0.90**	**0.81**	**0.86**
Paris	**1.00**	**1.00**	**1.00**	**0.90**	**0.90**	**0.81**
Ice	**0.90**	**0.86**	0.76	0.67	0.62	0.67
Harbour	**0.90**	**0.86**	**0.86**	0.67	0.48	0.43

In order to analyze the masking effect further, Table VI shows the codec identification accuracy obtained for different c_2 and QP_2 values. In nearly lossless conditions (low QP_2) the proposed method successfully identifies the first codec in almost all cases. Notice that the influence of lossy compression on the effectiveness of the proposed identification algorithm is content-dependent. Indeed, for *Foreman*, *News* or *Paris*, accuracy is large also for high QP_2 values, whereas for *Harbour* the method is prone to fail when QP_2 is moderately increased.

A further test that we performed was to study the codec estimation accuracy for different values of R. To this purpose, Table VII shows the accuracy for different c_2 and R, averaging results over the other parameters. Unlike in the GOP estimation case, codec estimation accuracy increases when the original sequence is encoded at *medium* quality (i.e., $R = R_M$). This is due to the fact that, at *medium* quality, traces left by c_1 are stronger than for $R = R_H$ and the sequence

TABLE VII: Codec identification accuracy for different $\mathtt{c}_2$ and R values. Bold is used for best result for each $\mathtt{c}_2$.

$\mathtt{c}_2$	R_L	R_M	R_H
MPEG-2	0.86	**0.90**	0.82
MPEG-4	0.83	**0.85**	0.78
AVC	0.83	**0.88**	0.82

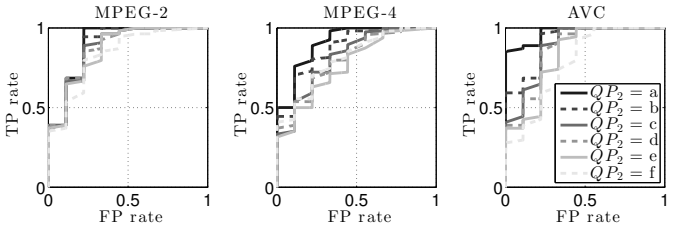


Fig. 10: Detection ROC for each codec $\mathtt{c}_2$. Results are averaged on the other parameters.

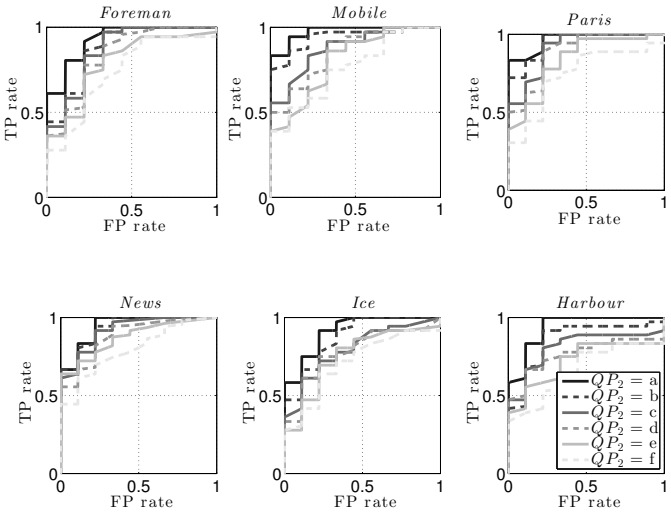


Fig. 11: Detection ROC for each sequence. As shown in Table VI, codec identification is sequence-dependent, although good results can be achieved for low QP values of $\mathtt{c}_2$ since the masking effect is less influential.

suffers less from other noise factor that can be introduced at $R = R_L$.

Finally, we tested the performance of the identification algorithm using a threshold, instead of comparing results between different $\mathtt{c}_3$. To this respect, we show the ROC curves obtained at different values of QP_2 for the second coding step. Let τ denote a threshold value. The proposed method labels a sequence as encoded with $\mathtt{c}_1$ whenever $J_{\mathtt{c}_1} > \tau$. The TP rate is the fraction of sequences originally encoded with $\mathtt{c}_1$ for which $J_{\mathtt{c}_1} > \tau$. Conversely, the FP rate is the fraction of sequences not encoded with $\mathtt{c}_1$ for which $J_{\mathtt{c}_1} > \tau$. ROC curves are traced by varying the value of τ. Figure 10 shows the ROC curves for each masking codec $\mathtt{c}_2$, averaging results across all the other parameters. This allows us to study the impact of the masking codec in terms of identification accuracy. These results highlight that it is possible to detect the codec based on a threshold value on $J_{\mathtt{c}_1}$, thus enabling the algorithm to work also in an open-group scenario.

In order to study the dependency on the video content, Figure 11 shows individual ROC curves for each sequence, always averaging results over all the other parameters. These charts confirm that codec identification is content-dependent, as already observed analyzing the results in Table VI and for the GOP.

A test in a partially uncontrolled scenario

Up to now, we only presented results obtained on synthetic datasets. This is of paramount importance as this is the only feasible solution to study the behavior of our algorithm under many different testing conditions. However, in order to verify that it is possible to apply the proposed method also in a less-controlled scenario, we performed an additional experiment using a media sharing platform as masking codec. More specifically, this proof of concept has been carried out using 9 sequences uploaded to YouTube. The sequences were created by encoding *Foreman* at *low*, *medium*, and *high* quality, with either MPEG-2, MPEG-4, and AVC. YouTube re-encoded all the sequences when uploaded, acting as $\mathtt{c}_2$. We then downloaded the sequences, and tested our method on them, using as $\mathtt{c}_3$ either MPEG-2, MPEG-4, AVC, and DIRAC.

Notice that in this case we only exploited the knowledge of G_2 and not the overall bitstream downloaded from YouTube. This means that we did not remove peaks due to QP_2, skipping step 2c of the algorithm, still achieving good results. More specifically, we were able to always estimate the correct GOP for sequences at *low* and *medium* quality. Conversely, GOP traces were completely lost on sequences at *high* quality, thus making GOP detection not possible. Concerning the codec identification, assuming G_1 as known, we always achieved the correct result.

Computational Complexity

As explained in Section V, the proposed algorithm works by re-encoding each sequence under analysis $|\mathcal{C}|\times|\mathcal{QP}|$ times, which may be time consuming. As a matter of fact, the time spent to analyze a 10 second CIF sequence with our simple implementation using a commercially available laptop (i.e., equipped with 8GB of RAM and a 2.2 GHz Intel Core i7 processor) is approximately 100 seconds.

Even though in many forensic applications accuracy is more valuable than time (e.g., where the number of videos to be analyzed is limited), reducing the computational complexity just slightly decreasing the algorithm accuracy may be interesting in some specific use cases. For this reason, we studied the possibility of reducing the computational complexity of our algorithm by decreasing: i) the size of the search space $\mathcal{QP}$, and; ii) the length of the sequence under analysis. More specifically, we tested the effect of sub-sampling the set $\mathcal{QP}$ to a smaller set with cardinality $|\mathcal{QP}|/\omega_{\text{QP}}$. Concerning time, we selected a portion of the sequence whose length is $1/\omega_{\text{T}}$ of the total length. The total computational time is then decreased by a factor $\omega_{\text{QP}} \times \omega_{\text{T}}$.

Concerning GOP estimation, we restricted the analysis to the cases in which GOP estimation is sufficiently reliable (i.e., values of QP_2 and G_1/G_2 such that $a_1(QP_2, G_1/G_2) > 0.5$ in Table II). Figure 12 shows the average accuracy obtained in

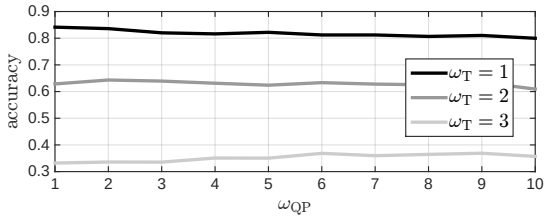


Fig. 12: Effect of reducing computational complexity on GOP estimation.

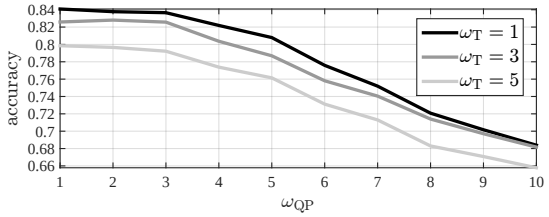


Fig. 13: Effect of reducing computational complexity on codec estimation.

estimating the GOP in this scenario for different values of ω_{QP} and ω_{T}. Notice that reducing the number of tested QP does not affect accuracy. Conversely, reducing the video length highly reduce the algorithm accuracy. This is due to the fact that, the shortest the sequence, the smaller the number of GOPs in it. Nonetheless, it is possible to reduce the computational time by a factor 10 (i.e., $\omega_{\mathrm{QP}} = 10$ and $\omega_{\mathrm{T}} = 1$).

Considering the codec estimation, Figure 13 shows the average accuracy on the whole dataset by changing ω_{QP} and ω_{T}. Notice that in this case it is still possible to have a reliable codec estimation also reducing the temporal resolution (i.e., increasing ω_{T}). As an example it is possible to reduce the computational complexity by a factor 15 by using $\omega_{\mathrm{QP}} = 3$ and $\omega_{\mathrm{T}} = 5$, decreasing the average accuracy by only 4%.

This preliminary analysis shows the actual possibility of scaling the computational complexity of the algorithm still achieving a high detection accuracy. As a matter of fact, it is possible to reduce the analysis time for a sequence of more than an order of magnitude, thus making the analysis time comparable with the sequence length (at CIF resolution).

VII. Discussion

In this paper we presented an algorithm to estimate the codec and GOP size used in the first coding step applied to a double encoded sequence. The algorithm exploits the idempotency property that video codecs inherit from quantizers and it is based on a recompress-and-observe scheme, building upon our previous work presented in [1].

A set of tests on an extended video dataset proves the validity of the proposed method highlighting its working conditions in terms of coding parameters used during the first and second coding steps. More specifically, we showed that it is possible to correctly identify the GOP size and codec (also considering different implementations) no matters which masking codec is used, as long as its quality is sufficiently high to preserve traces left by the first compression. Results using a threshold (i.e., ROC curves) also show that it is possible to discriminate, with a certain probability, which estimates should be considered as valid and which ones should be discarded. An additional proof of concept of our algorithm in a real world scenario (i.e., sequences uploaded on YouTube) highlights the possibility of using it in real applications. Moreover, a preliminary analysis also shows the possibility of reducing the algorithms' computational complexity by only slightly decreasing its accuracy. This possibility will be the topic for our future research.

References

[1] P. Bestagini, A. Allam, S. Milani, M. Tagliasacchi, and S. Tubaro, "Video codec identification," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2012, pp. 2257–2260.

[2] P. Bestagini, S. Milani, M. Tagliasacchi, and S. Tubaro, "Video codec identification extending the idempotency property," in *European Workshop on Visual Information Processing (EUVIP)*, June 2013, pp. 220–225.

[3] S. Milani, M. Fontani, P. Bestagini, M. Barni, A. Piva, M. Tagliasacchi, and S. Tubaro, "An overview on video forensics," *APSIPA Transactions on Signal and Information Processing*, vol. 1, p. e2, August 2012.

[4] S. Bian, W. Luo, and J. Huang, "Exposing fake bit-rate videos and estimating original bit-rates," *IEEE Transactions on Circuits and Systems for Video Technology (TCSVT)*, vol. 24, no. 12, pp. 1–1, December 2014.

[5] B. Haskell, A. Puri, and A. Netravali, *Digital Video: An Introduction to MPEG-2: An Introduction to MPEG-2*, ser. Digital multimedia standards series. Springer, 1997.

[6] I. Richardson, *H.264 and MPEG-4 Video Compression: Video Coding for Next-generation Multimedia*. Wiley, 2003.

[7] T. Wiegand, G. J. Sullivan, G. Bjntegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Transactions on Circuits and Systems for Video Technology (TCSVT)*, vol. 13, no. 7, pp. 560–576, July 2003.

[8] T. Borer and T. Davies, "Dirac video compression using open standards," BBC RD White Paper, Tech. Rep., 2005.

[9] A. Piva, "An overview on image forensics," *ISRN Signal Processing*, vol. 2013, p. 22, November 2013.

[10] Z. Fan and R. de Queiroz, "Maximum likelihood estimation of JPEG quantization table in the identification of bitmap compression history," in *IEEE International Conference on Image Processing (ICIP)*, September 2000, pp. 948–951.

[11] Z. Fan and R. L. de Queiroz, "Identification of bitmap compression history: JPEG detection and quantizer estimation," *IEEE Transactions on Image Processing (TIP)*, vol. 12, no. 2, pp. 230–235, April 2003.

[12] J. Lukáš and J. Fridrich, "Estimation of primary quantization matrix in double compressed JPEG images," in *Digital Forensic Research Workshop (DFRWS)*, August 2003.

[13] T. Bianchi and A. Piva, "Detection of nonaligned double JPEG compression based on integer periodicity maps," *IEEE Transactions on Information Forensics and Security (TIFS)*, vol. 7, no. 2, pp. 842–848, April 2012.

[14] ——, "Reverse engineering of double jpeg compression in the presence of image resizing," in *IEEE International Workshop on Information Forensics and Security (WIFS)*, December 2012, pp. 127–132.

[15] P. Ferrara, T. Bianchi, A. D. Rosa, and A. Piva, "Reverse engineering of double compressed images in the presence of contrast enhancement," in *IEEE International Workshop on Multimedia Signal Processing (MMSP)*, September 2013, pp. 141–146.

[16] W. Lin, S. Tjoa, H. Zhao, and K. Liu, "Digital image source coder forensics via intrinsic fingerprints," *IEEE Transactions on Information Forensics and Security (TIFS)*, vol. 4, no. 3, pp. 460–475, August 2009.

[17] M. Tagliasacchi, M. V. Scarzanella, P. L. Dragotti, and S. Tubaro, "Transform coder identification," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2013, pp. 5785–5789.

[18] ——, "Transform coder identification with double quantized data," in *IEEE International Conference on Image Processing (ICIP)*, September 2013, pp. 1660–1664.

[19] G. J. Sullivan, J.-R. Ohm, W. Han, and T. Wiegand, "Overview of the high efficiency video coding HEVC standard," *IEEE Transactions on Circuits and Systems for Video Technology (TCSVT)*, vol. 22, no. 12, pp. 1649–1668, December 2012.

[20] H. Li and S. Forchhammer, "MPEG2 video parameter and no reference PSNR estimation," in *Picture Coding Symposium (PCS)*, May 2009, pp. 1–4.

[21] W. Wang and H. Farid, "Exposing digital forgeries in video by detecting double quantization," in *ACM Multimedia and Security Workshop*, 2009, pp. 39–48.

[22] Y. Chen, K. Challapali, and M. Balakrishnan, "Extracting coding parameters from pre-coded MPEG-2 video," in *IEEE International Conference on Image Processing (ICIP)*, October 1998, pp. 360–364.

[23] M. Tagliasacchi and S. Tubaro, "Blind estimation of the QP parameter in H.264/AVC decoded video," in *International Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS)*, April 2010, pp. 1–4.

[24] W. Luo, M. Wu, and J. Huang, "MPEG recompression detection based on block artifacts," in *SPIE Conference on Security, Forensics, Steganography, and Watermarking of Multimedia Contents*, February 2008, p. 12.

[25] D. Vazquez-Padin, M. Fontani, T. Bianchi, P. Comesana, A. Piva, and M. Barni, "Detection of video double encoding with GOP size estimation," in *IEEE International Workshop on Information Forensics and Security (WIFS)*, December 2012, pp. 151–156.

[26] G. Valenzise, M. Tagliasacchi, and S. Tubaro, "Estimating QP and motion vectors in H.264/AVC video from decoded pixels," in *ACM Workshop on Multimedia in Forensics, Security and Intelligence (MiFor)*, 2010, pp. 89–92.

[27] S. Milani, M. Tagliasacchi, and S. Tubaro, "Identification of the motion estimation strategy using eigenalgorithms," in *IEEE International Conference on Image Processing (ICIP)*, September 2013, pp. 4477–4481.

[28] S. Milani, P. Bestagini, M. Tagliasacchi, and S. Tubaro, "Multiple compression detection for video sequences," in *IEEE International Workshop on Multimedia Signal Processing (MMSP)*, September 2012, pp. 112–117.

[29] S. Milani, M. Tagliasacchi, and S. Tubaro, "Discriminating multiple JPEG compression using first digit features," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2012, pp. 2253–2256.

[30] W. Luo, Y. Wang, and J. Huang, "Security analysis on spatial 1 steganography for JPEG decompressed images," *IEEE Signal Processing Letters (SPL)*, vol. 18, no. 1, pp. 39 –42, January 2011.

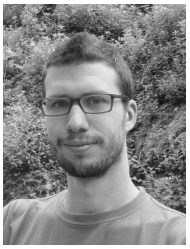
[31] H. Farid, "Exposing digital forgeries from JPEG ghosts," *IEEE Transactions on Information Forensics and Security (TIFS)*, vol. 4, no. 1, pp. 154 –160, March 2009.

[32] G. Valenzise, M. Tagliasacchi, and S. Tubaro, "Revealing the traces of JPEG compression anti-forensics," *IEEE Transactions on Information Forensics and Security (TIFS)*, vol. 8, no. 2, pp. 335–349, February 2013.

[33] V. F. Pisarenko, "The retrieval of harmonics from a covariance function," *Geophysical Journal of the Royal Astronomical Society*, vol. 33, no. 3, pp. 347–366, September 1973.

[34] R. Schmidt, "Multiple emitter location and signal parameter estimation," *IEEE Transactions on Antennas and Propagation (TAP)*, vol. 34, no. 3, pp. 276–280, March 1986.

[35] J. Zapata and E. Gómez, "Comparative evaluation and combination of audio tempo estimation approaches," in *AES Conference on Semantic Audio*, July 2011.



Paolo Bestagini was born in Novara (Italy) on February 22, 1986. He received the M.Sc. in Telecommunications Engineering and the Ph.D in Information Technology at the Politecnico di Milano in 2010 and 2014, respectively. He is currently a Post-Doc researcher at the Image and Sound Processing Group, Politecnico di Milano. His research activity is focused on multimedia forensics and acoustic signal processing for microphone arrays.

Simone Milani was born in Camposampiero, Italy, in 1978. He received the Laurea degree in telecommunication engineering and the Ph.D. degree in electronics and telecommunication engineering from University of Padova, Padova, Italy, in 2002 and 2007, respectively. He was a Visiting Ph.D. Student with University of California at Berkeley, Berkeley, CA, USA, in 2006. He was a Post-Doctoral Researcher with University of Udine, Udine, Italy; University of Padova; and Politecnico di Milano, Milan, Italy, from 2007 to 2013. He has also been with STMicroelectronics, Agrate, Italy. He is currently an Assistant Professor with the Department of Information Engineering, University of Padova. His research interests include digital signal processing, image and video coding, 3-D video processing and compression, joint source-channel coding, robust video transmission, distributed source coding, multiple description coding, and multimedia forensics.

Marco Tagliasacchi is currently Associate Professor at the "Dipartimento di Elettronica, Informazione e Bioingegneria - Politecnico di Milano", Italy. He received the "Laurea" degree (2002, cum Laude) in Computer Engineering and the Ph.D. in Electrical Engineering and Computer Science (2006), both from Politecnico di Milano. He was visiting academic at the Imperial College London (2012) and visiting scholar at the University of California, Berkeley (2004). His research interests include multimedia forensics, multimedia communications (visual sensor networks, coding, quality assessment) and information retrieval. Dr. Tagliasacchi coauthored more than 120 papers in international journals and conferences, including award winning papers at MMSP 2013, MMSP 2012, ICIP 2011, MMSP 2009 and QoMex 2009. He has been actively involved in several EU-funded research projects. He co-coordinated two ICT-FP7 FET-Open projects (GreenEyes and REWIND). Dr. Tagliasacchi is an elected member of the IEEE Information Forensics and Security Technical committee for the term 2014-2016, and served as member of the IEEE MMSP Technical Committee for the term 2009-2012. He is currently Associate Editor for the IEEE Transactions on Circuits and Systems for Video Technologies (2011 best AE award) and APSIPA Transactions on Signal and Information Processing. Dr. Tagliasacchi was General co-Chair of IEEE Workshop on Multimedia Signal Processing (MMSP 2013, Pula, Italy) and Technical Program Coordinator of IEEE International Conference on Multimedia&Expo (ICME 2015, Turin, Italy).

Stefano Tubaro was born in Novara in 1957. He completed his studies in Electronic Engineering at the Politecnico di Milano, Italy, in 1982. He then joined the Dipartimento di Elettronica, Informazione e Bioingegneria of the Politecnico di Milano, first as a researcher of the National Research Council, and then (in November 1991) as an Associate Professor. Since December 2004 he has been appointed as Full Professor of Telecommunication at the Politecnico di Milano. His current research interests are on advanced algorithms for video and sound processing. Stefano Tubaro authored over 180 scientific publications on international journals and congresses and he is co-author of more than 15 patents. In the past few years he has focused his interest on the development of innovative techniques for image and video tampering detection and, in general, for the blind recovery of the "processing history" of multimedia objects. Stefano Tubaro coordinates the research activities of the Image and Sound Processing Group (ISPG) at the Dipartimento di Elettronica, Informazione e Bioingegneria of the Politecnico di Milano. He had the role of Project Coordinator of the European Project ORIGAMI: A new paradigm for high-quality mixing of real and virtual and of the research project ICT-FET-OPEN REWIND: REVerse engineering of audio-VIsual coNtent Data. This last project was aimed at synergistically combining principles of signal processing, machine learning and information theory to answer relevant questions on the past history of such objects. Stefano Tubaro is a member the IEEE Multimedia Signal Processing Technical Committee and of the IEEE SPS Image Video and Multidimensional Signal Technical Commitee. He was in the organization committee of a number of international conferences: IEEE-MMSP-2004/2013, IEEE-ICIP-2005, IEEE-AVSS-2005/2009, IEEE-ICDSC-2009, IEEE-MMSP-2013, IEEE-ICME-2015 to mention a few. From May 2012 to April 2015 he was an Associate Editor of the IEEE Transactions on Image Processing, currently he is an Associate Editor of the IEEE Transactions on Information Forensics and Security.