Natural Steganography in JPEG Domain With a Linear Development Pipeline

Théo Taburet, Patrick Bas, Wadih Sawaya, Jessica Fridrich

Natural Steganography in JPEG Domain with a Linear Development Pipeline

Théo Taburet *Student Member, IEEE,* Patrick Bas, *Senior Member, IEEE,*
Wadih Sawaya *Member, IEEE,* and Jessica Fridrich, *Fellow Member, IEEE*

***Index Terms*—Steganography, JPEG, Development pipeline**

***Abstract*—In order to achieve high practical security, Natural Steganography (NS) uses cover images captured at ISO sensitivity ISO_1 and generates stego images mimicking ISO sensitivity $ISO_2 > ISO_1$. This is achieved by adding a stego signal to the cover that mimics the sensor photonic noise. This paper proposes an embedding mechanism to perform NS in the JPEG domain after linear developments by explicitly computing the correlations between DCT coefficients before quantization. In order to compute the covariance matrix of the photonic noise in the DCT domain, we first develop the matrix representation of demosaicking, luminance averaging, pixel section, and 2D-DCT. A detailed analysis of the resulting covariance matrix is done in order to explain the origins of the correlations between the coefficients of 3×3 DCT blocks. An embedding scheme is then presented that takes into account all the correlations. It employs 4 sub-lattices and 64 lattices per sub-lattices. The modification probabilities of each DCT coefficient are then derived by computing conditional probabilities computed from a multivariate Gaussian distribution using the Cholesky decomposition of the covariance matrix. This derivation is also used to compute the embedding capacity of each image. Using a specific database called *E1Base*, we show that in the JPEG domain NS (J-Cov-NS) enables to achieve high capacity (more than 2 bits per non-zero AC DCT) and with high practical security ($P_{\mathrm{E}} \simeq 40\%$ using DCTR and $P_{\mathrm{E}} \simeq 32\%$ using SRNet) from QF 75 to QF 100).**

I. Introduction

In 1998, Cachin [1] defined the theoretical security of a steganographic embedding scheme as $D_{KL}(P_X, P_Y)$, the Kullback–Leibler divergence between the distributions of the cover contents P_X and stego contents P_Y. Using this definition, a scheme providing $D_{KL}(P_X, P_Y) = 0$ should be theoretically perfectly secure.

Interestingly, only few exceptions, such as Model-Based Steganography (MBS) [2], HUGO [3], and MiPOD [4], are based on Cachin's rationale, while the majority of embedding schemes, such as UNIWARD [5], HILL [6], and UERD [7] minimize the sum of empirically defined costs based on the local complexity of each pixel/DCT coefficient. In MBS, the embedding preserves the underlying generalized Cauchy distribution fit to each DCT mode. In HUGO, the cost is computed from the difference between the SPAM features set [8] used for steganalysis. MiPOD minimizes the deflection coefficient, i.e., the normalized difference between the expectations of the likelihood ratio under the two hypotheses in the weak signal and large data sample asymptotics, as a "cost."

Natural Steganography (NS) [9], [10], [11], [12] is based on the same principle as model based steganography since it embeds message whose associated stego signal tries to mimic the statistical properties of the camera photonic noise, a.k.a. camera shot noise. Starting with a cover image acquired at ISO_1, the embedding is designed in such a way that the stego image looks like an image acquired at a larger ISO sensitivity $ISO_2 > ISO_1$. This strategy is named "cover-source switching" since it relies on changing the model of the cover-source during the embedding process. In the pixel domain or for monochrome sensors [9], [10], [11], [12], this approach has been shown to achieve both high capacity and statistical undetectability as long as the embedder is able to correctly model the added signal. The high security of NS schemes is also due to the fact that NS uses a pre-cover at the embedder [9]. In contrast to other schemes relying on side information, such as SI-UNIWARD [5] or other side-informed implementations [13], the embedding capacity of NS is only limited by the gap between the two ISO sensitivities.

In the spatial domain, implementations of NS have been proposed for monochrome sensors, which do not perform demosaicking, with a development processes that includes only quantization, gamma correction [9], and downsampling [10]. In the JPEG domain, previous works [11], [12] have shown that models that only consider first-order marginal statistics (histograms) work well for monochrome sensors but the embedding is very detectable for color sensors since the embedding does not take into account dependencies due to demosaicking.

Note that like side-informed embedding methods [14], [5], NS uses a pre-cover image, being here the RAW image. This application scenario can be practically motivated by the fact that it is nowadays possible for a user to record his acquisition in RAW format, even on smart-phones [15].

The goal of this paper is to extend Natural Steganography in the JPEG domain to color sensors. The paper is organized as follows. Section II introduces notation, and describes the considered development pipeline and the principle of embedding using NS. Section III derives the statistical distribution of the stego signal in the DCT domain by computing the covariance matrix of its associated joint distribution. Section IV provides a deep analysis of different components of the resulting covariance matrix. Finally,

Théo Taburet and Patrick Bas are with CNRS, Ecole Centrale de Lille, CRIStAL Lab 59651 Villeneuve d'Ascq Cedex, France
Wadih Sawaya is with IMT Lille-Douai
Univ. Lille, CNRS, Centrale Lille Cité Scientifique UMR 9189, France
Jessica Fridrich is with Department of ECE, SUNY Binghamton, NY, USA

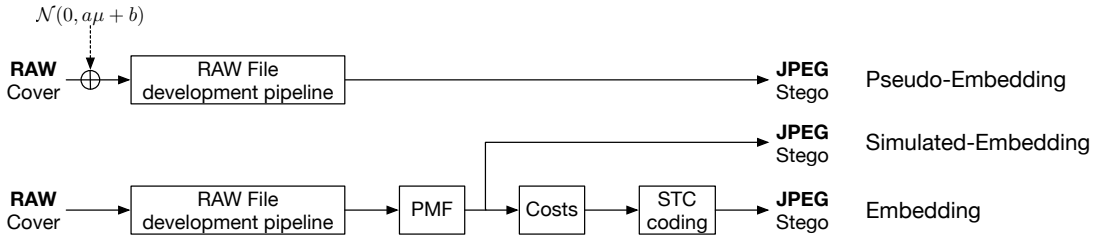


Fig. 1: Differences between embedding, simulated embedding, and pseudo embedding.

Section V presents the embedding scheme. The new scheme is benchmarked in Section VI and compared with relevant state-of-the-art steganographic schemes.

This paper is an important extension of the method presented in [12], where the statistical properties of the photonic noise are obtained by empirically estimating the noise covariance matrix. The obtained estimation error leads to a higher detectability, especially for high JPEG quality factors. In this paper, we instead compute the covariance matrix exactly as presented in [16]. We add an extensive analysis of the properties of this matrix and a detailed description of the embedding scheme. We also propose a large variety of results at different JPEG quality factors and for different alphabet sizes.

II. Preliminaries

A. Notations

Throughout this article, we use capital letters for random variables X and their corresponding lowercase symbols for their realizations x. Matrices are written in uppercase $\mathbf{A}$ and vectors (of scalar or random variables) in lowercase boldface font $\mathbf{a}$. Matrix transposition is denoted with a superscript $\mathbf{A}^t$. The subscripts $\square_p$ and $\square_d$ will be respectively associated to the photo-site domain and the developed domain.

In this article, matrix vectorization of matrices according to the rows or columns are used. For a $m \times n$ matrix $\mathbf{A}$, the respective vectorization by rows and columns is defined as follows:

For:

$$\mathbf{A} \in \mathbb{R}^{m \times n} / \mathbf{A} = \begin{bmatrix} a_{1,1} & \dots & a_{1,n} \\ \vdots & & \vdots \\ a_{m,1} & \dots & a_{m,n} \end{bmatrix} \tag{1}$$

the respective vectorization by columns (C) and rows (R) is defined as follows:

$$\mathrm{vec_C}(\mathbf{A}) = [a_{1,1}, \dots, a_{m,1}, \dots, a_{1,n}, \dots a_{m,n}]^t \in \mathbb{R}^{mn \times 1} \tag{2}$$

$$\mathrm{vec_R}(\mathbf{A}) = [a_{1,1}, \dots, a_{1,n}, \dots, a_{m,1}, \dots a_{m,n}]^t \in \mathbb{R}^{mn \times 1} \tag{3}$$

B. Pseudo-embedding, simulated embedding and embedding

We distinguish between three forms of steganographic embedding that are illustrated in Figure 1: *pseudo-embedding*, *simulated embedding,* and (true) *embedding*.

Pseudo-embedding means that practical embedding is not possible with the proposed implementation. It acts as a generic mathematical operation (a reference) which outputs the so-called *pseudo-stego* image should be statistically distributed like the stego image.

In *simulated embedding*, the embedding changes are simulated according to a given selection channel using the probability $\pi_i(k)$ of modifying the i^{th} cover sample by magnitude $k \in \{-K, ..., K\}$.

(True) embedding can be realized using multilayered STCs [17] based on costs $\rho_i(k)$ directly computed from the set of embedding probabilities $\pi_i(k)$, with $\rho_i(k) = \log\left(\pi_i(0)/\pi_i(k)\right)$. The STC algorithm minimizes the sum of embedding costs while embedding the payload using a Viterbi algorithm.

C. Principles of Natural Steganography

We first review the principles of Natural Steganography when pseudo embedding is performed at the photo-site level, and then introduce the technical goals of this paper.

1) Pseudo-embedding at the photo-site level: Modifying the photo-sites directly leads to pseudo-embedding. However, as mentioned in [9], it can also be directly used for simulated embedding or true embedding in the spatial domain for monochrome sensors.

The key idea here is to add a stego signal S that mimics the statistical properties of the photonic noise. For a CCD or CMOS sensor, the photonic noise N at photo-site i, j due to the error of photonics count during acquisition is assumed to be independent across photo-sites with a widely adopted heteroscedastic model [18]:

$$N_{i,j}^{(1)} \sim \mathcal{N}\left(0, a_1 \mu_{i,j} + b_1\right), \tag{4}$$

where $\mu_{i,j}$ is the noiseless photo-site value at photo-site i, j, and (a_1, b_1) a pair of parameters depending only on the ISO_1 sensitivity and the specific sensor. The acquired photo-site sample $x_{i,j}^{(1)}$ is thus a realization $x_{i,j}^{(1)} = \mu_{i,j} + n_{i,j}^{(1)}$ of a Gaussian variable distributed as $X_{i,j}^{(1)} \sim \mathcal{N}\left(\mu_{i,j}, a_1 \mu_{i,j} + b_1\right)$.

In the same way, for sensitivity ISO_2: $X_{i,j}^{(2)} \sim \mathcal{N}\left(\mu_{i,j}, a_2 \mu_{i,j} + b_2\right)$. Thus, we can generate a

stego image mimicking a cover captured at ISO_2 such that for each photo-site i, j we have:

$$y_{i,j} = x_{i,j}^{(1)} + s_{i,j}, \tag{5}$$

with $S_{i,j}$ the random variable representing the stego signal:

$$S_{i,j} \sim \mathcal{N}\left(0, \left(a_2 - a_1\right) x_{i,j} + b_2 - b_1\right). \tag{6}$$

The photo-site of the stego image is then distributed as:

$$Y_{i,j} \sim \mathcal{N}\left(\mu_{i,j}, a_1 \mu_{i,j} + b_1 + \left(a_2 - a_1\right) x_{i,j} + b_2 - b_1\right). \tag{7}$$

Assuming that the value of the observed photo-site is close to its expectation, i.e., $\mu_{i,j} \approx x_{i,j}^{(1)}$, we obtain

$$Y_{i,j} \overset{d}{=} X_{i,j}^{(2)}, \tag{8}$$

where $\overset{d}{=}$ represents the equality in distribution of two random variables. Equation (8) highlights that the distribution of a stego image photo-site is the same as the distribution of a cover photo-site acquired at ISO_2. Equation (5) is the pseudo-embedding operation, which enables us to generate pseudo-stego content at the photo-site level. Practically, the distribution of the stego signal in the continuous domain takes into account the statistical model of the shot noise estimated for two ISO settings, ISO_1 and ISO_2, using the procedure described in [9], [19]. The work presented in [9], [10] shows that for monochrome sensors, this model in the spatial domain can be used to derive the distribution of the stego signal in the spatial domain after quantization, gamma correction, and image downsampling using bilinear kernels.

2) Simulated embedding in JPEG domain: The main purpose of this paper is to detail how to perform modifications on quantized DCT coefficients in order to perform simulated embedding. The modeling of the stego signal and its dependencies in the DCT domain are crucial for the embedding to be secure. We thus focus on modeling the image development process in order to firstly derive the statistical characteristics of the stego signal in the DCT domain, then compute the modification probabilities for each DCT coefficient, and finally perform simulated embedding.

The next section, we explain how we reach the first goal and in Section V we detail the algorithm used to perform simulated embedding.

III. Modeling dependencies in the DCT domain

A. The development pipeline

In this paper, we use a linear development pipeline. Since the distribution at the photo-site level of the random vector of components $S_{i,j}$ is multivariate Gaussian (with diagonal covariance matrix), and because the pipeline up to the DCT transform is a succession of linear operations, one main result of statistical signal processing [20] it that its distribution in the DCT domain is also a multivariate Gaussian distribution, but with arbitrary covariance matrix. The linear development allows us also to derive the covariance matrix of this distribution. We can write that $\mathbf{y}_p = \mathbf{x}_p + \mathbf{s}_p$, where $\mathbf{x}_p$ is the vectorized version of a block of photo-site values of the cover image, and $\mathbf{s}_p$ the vectorized values of the added stego signal in the photo-site domain.

The goal of this section is to model the development pipeline as a linear equation in the form of:

$$\mathbf{y}_d = \mathbf{M}\mathbf{y}_p \Leftrightarrow \mathbf{s}_d = \mathbf{M}\mathbf{s}_p, \tag{9}$$

where $\mathbf{y}_d$ and $\mathbf{s}_d$ represent the vectors of respectively the stego content and the stego signal in the developed domain. Since the only random component is the stego signal $\mathbf{s}_p$, the covariance matrix $\mathbf{\Sigma}_d = \mathrm{Cov}(\mathbf{s}_d)$ of the multivariate distribution in the DCT domain will then be given by:

$$\mathbf{\Sigma}_d \quad = \mathbf{M}\,\mathbf{\Sigma}_p\,\mathbf{M}^t, \tag{10}$$

where $\mathbf{\Sigma}_p = \mathrm{Cov}(\mathbf{s}_p)$ is the covariance matrix of the considered block of the stego signal in the photo-site domain given the cover $\mathbf{x}$.

Denoting now i the index of one photo-site in $\mathbf{x}_p$, using (6) and the hypothesis $\mu_i \approx x_i$, the covariance matrix $\mathbf{\Sigma}_p$ of the stego-signal is a diagonal matrix with diagonal terms equal to $\left(a_2 - a_1\right) x_i + b_2 - b_1$.

In order to compute $\mathbf{M}$, we consider the different steps of the pipeline and decompose the computation of $\mathbf{M}$ into the following steps (see Figure 2):

1) Demosaicking: this step predicts for each photo-site the two missing colors that are not recorded by the sensor. We use bilinear filtering as a linear interpolation process.
2) Luminance averaging: (we only consider embedding in grayscale JPEG image) the demosaicked vector undergoes luminance averaging following the ITU-R BT 601 standard.
3) 2D-DCT transform is computed independently on each block of 8×8 pixels.
4) Quantization: the DCT coefficients are quantized using the quantization table matching a selected JPEG quality factor (QF) to generate a set of JPEG coefficients. Note that since this operation is non-linear, it is not captured by equation (9).

We now detail the different linear operations which are detailed on content vectors $\mathbf{y}_f$ (the subscript f denoting the operation), but can also be written w.r.t. $\mathbf{s}_f$ thanks to the linear formulation by switching $\mathbf{y}_f$ by $\mathbf{s}_f$.

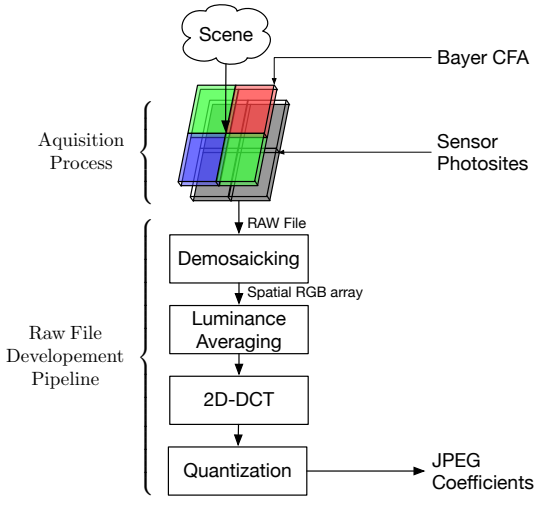


Fig. 2: Development pipeline: From a scene captured by a color sensor to luminance JPEG coefficients.

B. Considered photo-sites

Since the color interpolation step uses the neighboring photo-sites to interpolate colors, this creates correlations between adjacent 8-connected blocks of 8×8 photo-sites.

These correlations between blocks can be very weak, especially between diagonal blocks. On the contrary, it is important to note that two blocks which are not 8-connected represent independent realizations of the sensor-noise after demosaicking. This property will be used in Section (V) to design the embedding scheme. Both correlation between adjacent blocks and uncorrelated blocks are illustrated in Figure 3. On this figure we can see that two diagonal blocks can share only two correlated photo-sites, and the correlations can either come from three photo-site values coming from vertical, horizontal, and diagonal blocks (this is the case between NE and SW neighbors), or two photo-site values coming from horizontal and vertical blocks only (this is the case for NW or SE neighbors). On the contrary two blocks that are disconnected are associated to uncorrelated stego signals.

In order to capture all the correlations between DCT coefficients, we consequently need to consider a matrix $\mathbf{Y}_p$ of $(3 \times 8 + 2) \times (3 \times 8 + 2)$ photo-sites, which gives after vectorization $\mathrm{vec_R}(\mathbf{Y}_p)$ a vector $\mathbf{y}_p$ of 676 photo-sites as an input of our linear system as illustrated in Figure 4.

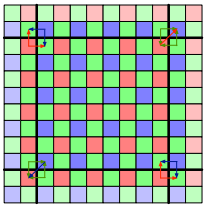


Fig. 3: Locations of photo-sites (dark colors) used to interpolate pixel values within one block using bilinear demosaicking. Diagonal blocks are involved in the computation on two pixels for the blue channel (up right) and the red channel (bottom, left).

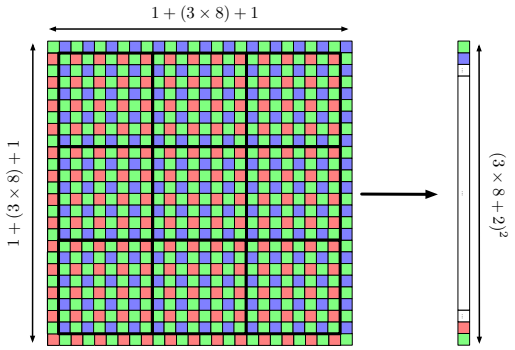


Fig. 4: RAW photo-sites and its outer border.

C. Demosaicking

It is possible to write the demosaicking operations as matrix multiplications. For each component R, G and B, we define the matrices $\mathbf{D}_r$, $\mathbf{D}_g$, $\mathbf{D}_b$ of size $(24+2)^2 \times (24+2)^2$, such that the result of the matrix multiplication of $\mathbf{y}_p$ with one of these matrices is the vectorized version of the corresponding color channel after demosaicking:

$$\mathbf{y}_r = \mathbf{D}_r \mathbf{y}_p,\ \mathbf{y}_g = \mathbf{D}_g \mathbf{y}_p,\ \mathbf{y}_b = \mathbf{D}_b \mathbf{y}_p. \tag{11}$$

Denoting i the index of one photo-site in $\mathbf{y}_i$, one row i of $\mathbf{D}_k$, $k \in \{r, g, b\}$ is obtained by vectorization of a $(24+2) \times (24+2)$ matrix with zeros entries except a specific kernel specific kernel $\mathbf{K}_i$ centered on (i, i). This kernel models any kind of interpolation between neighboring photo-sites and y_i:

$$\mathrm{row}_i(\mathbf{D}_k) = \mathrm{vec}_R \begin{bmatrix} \mathbf{0} & \cdots & & & & \\ \vdots & \ddots & & & & \\ & & \ddots & & & \\ & & & \mathbf{K}_i & \mathbf{0} & \cdots \\ & & & \mathbf{0} & \mathbf{0} & \\ & & & \vdots & & \ddots \end{bmatrix}. \tag{12}$$

Without loss of generality we now focus on the computation of $\mathbf{D}_g$, we consequently have two possibilities in this case:

- If index i corresponds to a green photo-site on the Bayer CFA, this photo-site does not need color interpolation, i.e.:
$$\mathbf{K}_i = [\mathbf{1}]. \tag{13}$$
- If index i corresponds to a pixel which needs to be interpolated, then:
$$\mathbf{K}_i = \begin{bmatrix} 0.25 & 0 & 0.25 \\ 0 & \mathbf{0} & 0 \\ 0.25 & 0 & 0.25 \end{bmatrix}. \tag{14}$$

The kernel coefficient in bold representing the location (i, i). For the red and blue channels, we use four different convolution kernels $\mathbf{K}_i$ to build $\mathbf{D}_r$ and $\mathbf{D}_b$, which are:

$$[\mathbf{1}], \begin{bmatrix} 0.5 \\ \mathbf{0} \\ 0.5 \end{bmatrix}, \begin{bmatrix} 0.5 & \mathbf{0} & 0.5 \end{bmatrix} \text{and} \begin{bmatrix} 0.25 & 0 & 0.25 \\ 0 & \mathbf{0} & 0 \\ 0.25 & 0 & 0.25 \end{bmatrix},$$

and are respectively used for duplication, interpolation between vertical or horizontal photo-sites and interpolation between four diagonal photo-sites.

D. Luminance averaging

To perform luminance averaging, we can define the matrix $\mathbf{L}$ following the ITU-R BT 601 standard as:

$$\mathbf{y}_l = \underbrace{(0.2126 \cdot \mathbf{D}_r + 0.7152 \cdot \mathbf{D}_g + 0.0722 \cdot \mathbf{D}_b)}_{\mathbf{L}} \mathbf{y}_p, \tag{15}$$

with $\mathbf{y}_l \in \mathbb{R}^{(24+2)^2 \times 1}$.

E. Pixel selection

As stated above, the surrounding edges of 3×3 blocks of samples are included in order to take into account the convolution window during demosaicking. Once the demosaicking operations have been carried out, the photo-sites not present in the DCT blocks can then be discarded. Let us

denote $\mathbf{Y}_l$ the $(24+2)\times(24+2)$ photo-sites matrix with its outer border, and $\mathbf{Y}_s$ without it as depicted in Figure 5. The selection matrix $\mathbf{S}\in\mathbb{R}^{(24)^2\times(24+2)^2}$ can then be defined such that:

$$\mathbf{y}_s = \mathrm{vec_R}(\mathbf{Y}_s) = \mathbf{S}\,\mathrm{vec_R}(\mathbf{Y}_l) = \mathbf{S}\mathbf{y}_l. \tag{16}$$

and we also can write: $\mathbf{y}_s = \mathbf{S}\,\mathbf{L}\mathbf{y}_p$.

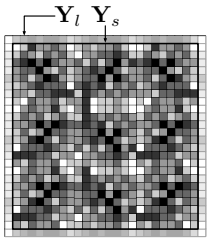


Fig. 5: Block representation of the pixel selection operation.

F. Blocks permutation and block selection

Blocks permutation and block selection are not mandatory, but they are useful to compute conditional probabilities while limiting the computational load (see section V). Depending on the lattice considered during the embedding (see again section V), the correlation matrix can be computed for DCT coefficients belonging to one, five or nine adjacent blocks.

In order to mathematically express a block permutation and selection as the matrix multiplication

$$\mathbf{y}_{pe} = \mathbf{P}\,\mathbf{y}_s, \tag{17}$$

we define $\mathbf{Y}_s\in\mathbb{R}^{24\times24}$ as an array composed of the 3×3 blocks of pixels, such that the vector $\mathbf{y}_s = \mathrm{vec_R}(\mathbf{Y_s})$, with:

$$\mathbf{Y_s} = \left[\begin{array}{|c|c|c|} \hline \mathbf{B}_{0,0} & \mathbf{B}_{0,1} & \mathbf{B}_{0,2} \\ \hline \mathbf{B}_{1,0} & \mathbf{B}_{1,1} & \mathbf{B}_{1,2} \\ \hline \mathbf{B}_{2,0} & \mathbf{B}_{2,1} & \mathbf{B}_{2,2} \\ \hline \end{array}\right],$$

where $\mathbf{B}_{i,j}\in\mathbb{R}^{8\times8}$ are blocks of 8×8 pixels, $0\le i,j\le2$. We recall that DCT is performed independently on each of these blocks. We need then to extract from $\mathbf{y}_s$ the vector corresponding to the required sequence of each block.

For each block to extract, we define a $8^2\times24^2$ block selection matrix $\mathbf{P}_{i,j}$ composed of 3 sub-matrices $\left[\begin{array}{ccc}\tilde{\mathbf{P}}_0 & \tilde{\mathbf{P}}_1 & \tilde{\mathbf{P}}_2\end{array}\right]$, where the size of $\tilde{\mathbf{P}}_\mathbf{i}$ is $64\times(3\cdot64)$, $0\le i\le2$. When extracting $\mathrm{vec_R}(\mathbf{B}_{i,j})$, all $\tilde{\mathbf{P}}_\mathbf{k}$, $k\neq i$ are set to zero and $\tilde{\mathbf{P}}_i$ takes the following entries:

$$\tilde{\mathbf{P}}_i = \begin{bmatrix} \mathbf{F}_j & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{F}_j & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{F}_j & \cdots & \mathbf{0} \\ \vdots & \vdots & \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{F}_j \end{bmatrix},$$

where $\mathbf{F}_j$ is a 8×24 sub-matrix consisting of 3 sub-matrices $\left[\begin{array}{ccc}\tilde{\mathbf{F}}_0 & \tilde{\mathbf{F}}_1 & \tilde{\mathbf{F}}_2\end{array}\right]$, each of size 8×8. When extracting $\mathrm{vec_R}(\mathbf{B}_{i,j})$, all $\tilde{\mathbf{F}}_k$, $k\neq j$, are set to zero and $\tilde{\mathbf{F}}_j = \mathbf{I}_8$, the identity 8×8 matrix.

We illustrate this with two examples.

Example 1: Suppose we need to extract the vectorized form of the central block $\mathbf{B}_{1,1}$, i.e., $i=1$ and $j=1$. We then have:

$$\mathbf{F}_1 = \left[\begin{array}{ccc}\mathbf{0} & \mathbf{I}_8 & \mathbf{0}\end{array}\right],$$

and

$$\mathbf{P}_{1,1} = \begin{bmatrix} \mathbf{0} & \cdots & \mathbf{0} & \mathbf{F}_1 & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \ldots & \mathbf{0} & \mathbf{0} & \mathbf{F}_1 & & \mathbf{0} & \mathbf{0} & \ldots & \mathbf{0} \\ \vdots & \ddots & \mathbf{0} & \vdots & \mathbf{0} & \ddots & \mathbf{0} & \vdots & \ddots & \mathbf{0} \\ \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{F}_1 & \mathbf{0} & \cdots & \mathbf{0} \end{bmatrix}.$$

The corresponding vector permutation matrix is then

$$\mathbf{P} = \mathbf{P}_{1,1}.$$

Example 2: This additional example is useful for the remaining of the paper (see Section V-A). Let us extract from $\mathbf{y}_s$ the vector resulting from the concatenation of the vectorized version of five 8×8 blocks of pixels in a given order,

$$\mathbf{y}_B = [\mathrm{vec_R}(\mathbf{B}_{1,1}), \mathrm{vec_R}(\mathbf{B}_{0,0}), \mathrm{vec_R}(\mathbf{B}_{0,2}), \mathrm{vec_R}(\mathbf{B}_{2,0}), \mathrm{vec_R}(\mathbf{B}_{2,2})]^t$$

The corresponding matrix operation will be:

$$\mathbf{P} = \left[\begin{array}{ccccc}\mathbf{P}_{1,1} & \mathbf{P}_{0,0} & \mathbf{P}_{0,2} & \mathbf{P}_{2,0} & \mathbf{P}_{2,2}\end{array}\right]^t.$$

G. 2D-DCT Transform

For a 8×8 block in the spatial domain, $\mathbf{B}$, its 2D-DCT block version written here as $\mathbf{B}_d$ can be expressed by the following matrix multiplication:

$$\mathrm{DCT}(\mathbf{B}) = \mathbf{A}\cdot\mathbf{B}\cdot\mathbf{A}^t = \mathbf{A}\cdot(\mathbf{A}\cdot\mathbf{B}^t)^t, \tag{18}$$

with:

$$\mathbf{A} = \begin{bmatrix} a & a & a & a & a & a & a & a \\ b & d & e & g & -g & -e & -d & -b \\ c & f & -f & -c & -c & -f & f & c \\ d & -g & -b & -e & e & b & g & -d \\ a & -a & -a & a & a & -a & -a & a \\ e & -b & g & d & -d & -g & b & -e \\ f & -c & c & -f & -f & c & -c & f \\ g & -e & d & -b & b & -d & e & -g \end{bmatrix}, \tag{19}$$

and :

$$\begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \\ g \end{bmatrix} = \frac{1}{2}\begin{bmatrix} \cos(\frac{\pi}{4}) \\ \cos(\frac{\pi}{16}) \\ \cos(\frac{\pi}{8}) \\ \cos(\frac{3\pi}{16}) \\ \cos(\frac{5\pi}{16}) \\ \cos(\frac{3\pi}{8}) \\ \cos(\frac{7\pi}{16}) \end{bmatrix}. \tag{20}$$

It should be observed that the multiplication by $\mathbf{A}$ and $\mathbf{A}^t$ is due to the fact that the DCT transform is separable and processes the columns and rows independently. In order to compute the covariance matrix of the spatial signal $\mathbf{B}$, we use vector notation by transforming the matrix $\mathbf{B}\in\mathbb{R}^{8\times8}$ into a vector $\mathbf{b}\in\mathbb{R}^{64}$ by concatenating the columns. As a result, the

8×8 matrix $\mathbf{A}$ is transformed into a 64×64 matrix $\mathbf{A}_v$ given by :

$$\mathbf{A}_v = \begin{bmatrix} \mathbf{A} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{A} & \mathbf{0} & \vdots \\ \vdots & \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{0} & \cdots & \mathbf{0} & \mathbf{A} \end{bmatrix}. \tag{21}$$

We also define a transpose operator $\mathbf{T}_r \in \mathbb{R}^{64\times64}$ such as $\text{vec}_c(\mathbf{X_S^t}) = \mathbf{T}_r \cdot \text{vec}_c(\mathbf{X_S}) = \mathbf{T}_r \cdot \mathbf{x_S}$, with :

$$\mathbf{T}_r = (\delta_{r(i),\, c(j)})_{\substack{0\leq i<64,\\ 0\leq j<64}}$$

and,

$$\begin{aligned} r(i) &= 8\,\lfloor i/8 \rfloor + (i \bmod (8)), \\ c(j) &= 8(j \bmod (8) + \lfloor j/8 \rfloor, \end{aligned}$$

$\delta_{r(i),\, c(j)}$ being the Kronecker function applied to row $r(i)$ and column $c(j)$.

The transpose operation $\mathbf{B}^t$ is then equivalent to the multiplication $\mathbf{T}_r \cdot \mathbf{b}$, and the vector form of the DCT 8×8 block $DCT(\mathbf{B})$ finally becomes:

$$\text{DCT}(\mathbf{b}) = \underbrace{\mathbf{A}_v\, \mathbf{T}_r\, \mathbf{A}_v\, \mathbf{T}_r}_{\mathbf{T}_b}\, \mathbf{b} \tag{22}$$

In order to compute the DCT of n blocks of size 8×8 ($n \in \{1, 5, 9\}$), we now define :

$$\mathbf{T} = \begin{pmatrix} \mathbf{T}_b & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \mathbf{T}_b \end{pmatrix}.$$

With $\mathbf{T}$ a block diagonal matrix with n matrices $\mathbf{T}_b$ on its diagonal.

H. Whole covariance matrix

The development pipeline can be then explicitly formulated as

$$\mathbf{s}_d = \mathbf{M}\mathbf{s}_p = \underbrace{\mathbf{T}\,\mathbf{P}\,\mathbf{S}\,\mathbf{L}}_{\mathbf{M}}\, \mathbf{s}_p, \tag{23}$$

and the covariance matrix is computed as:

$$\mathbf{\Sigma}_d \quad = \mathbf{M}\, \mathbb{E}\left[\mathbf{s}_p\, \mathbf{s}_p^t\right]\, \mathbf{M}^t. \tag{24}$$

Note that for a uniform constant RAW image defined by $\mu = \text{const.}$ (i.e., $\mathbb{E}\left[\mathbf{s}_d \cdot \mathbf{s}_d^t\right] \propto \mathbf{I}$), we obtain $\mathbf{\Sigma}_d \propto \mathbf{M}\mathbf{M^t}$. Depending of the number of blocks n considered in the neighborhood ($n \in \{1, 5, 9\}$, see V-A), the size of $\mathbf{\Sigma}_d$ is $(n \times 64, n \times 64)$.

IV. Analysis of the Covariance Matrix

In this section, we analyze the properties of the derived covariance matrix and interpret its different components. We show that the inter-block correlations are due to the signal continuity between blocks and that intra-block correlations highlight both artifacts due to demosaicking and due to low-pass filtering.

Note that this analysis is beneficial in order to understand the causes of the observed covariances. This understanding enables to decompose the embedding scheme into independent lattices (see section V) but also to pave the road for other synchronization strategies applied to other development pipelines. For example, in [21], the covariance matrix is limited to the effect of averaging and can be used to synchronize DCT coefficients of classical schemes such as UERD or J-Uniward. In [22], relationships between DCT coefficients to preserve continuities are in line with the presented analysis of the inter-block correlations (see IV-B).

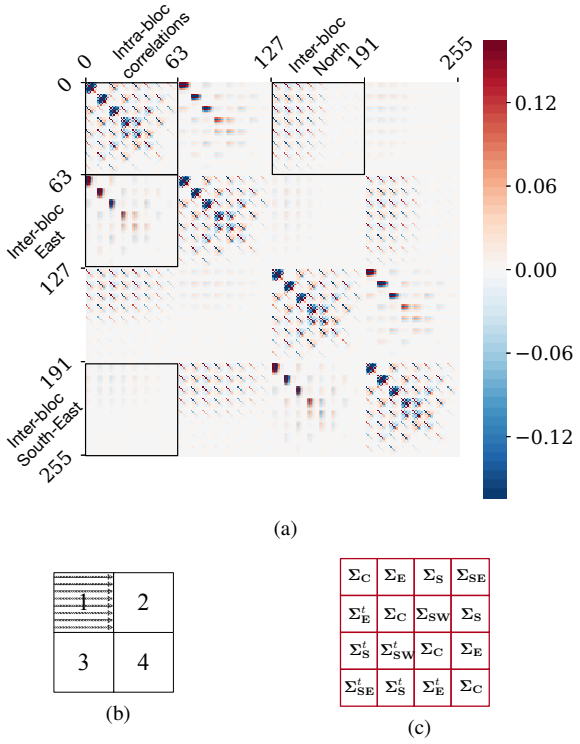


Fig. 6: (a) 256×256 covariance matrix of DCT coefficients of a color sensor with bilinear demosaicking for an i.i.d signal (the correlation values are thresholded for visualization purposes). (b): scan order by blocks and coefficients. (c): types of sub-matrices representing the 9 covariance matrices.

As non connected blocks are uncorrelated, we focus here on only four adjacent 8×8 blocks of unquantized DCT coefficients, as depicted in Figure (6b). This selection enables us to analyze correlations within a block, but also correlations between horizontal, vertical and diagonal neighboring blocks. By observing Figure (6a) together with the scan order depicted in Figure (6b), we can decompose the entire covariance matrix into four types of matrices of size 64×64 as illustrated in Figure (6c):

- Intra-block 8×8 covariance matrices of type $\mathbf{\Sigma}_C$ capture the correlations between DCT coefficients in the same block. They are located on the diagonal of the covariance matrix $\mathbf{\Sigma}_d$. Note that DCT coefficients can be positively or negatively correlated.
- Horizontal inter-block covariance matrices of type $\mathbf{\Sigma}_E$ or $\mathbf{\Sigma}_W$. They hold correlations between horizontal blocks.
- Vertical inter-block covariance matrices capture

correlations between vertical blocks. They can be of type $\mathbf{\Sigma}_N$ or $\mathbf{\Sigma}_S$.
- Diagonal inter-block covariance matrices capture correlations between diagonal blocks. They can be of type $\mathbf{\Sigma}_{NE}$, $\mathbf{\Sigma}_{SW}$,$\mathbf{\Sigma}_{SE}$, or $\mathbf{\Sigma}_{NW}$.

It is worth noting that the stationary behavior that appears here in $\mathbf{\Sigma}_d$ is not true for real images where the input signal is not identically distributed. Being aware of this, we do not consider stationarity for the embedding procedure (see Section (V)) but we use it only for analysis purposes. We give now an accurate analysis of the structure of the above defined covariances matrices.

A. Intra-block correlations

The coefficients of the covariance matrix for intra-block correlations are of two types: they are either due to demosaicking artifacts (see Section IV-A1), or the consequence of low-pass filtering (see Section IV-A2).

1) Effect of demosaicking: In order to emphasize the effect of demosaicking, we select only one color channel, the red one, and we investigate the intra-block correlations when the luminance computation operation is not taken into account. The demosaicking operation introduces dependencies within the same block and this is both due to the structure of the CFA itself and the color interpolation algorithm. For a given waveform of the DCT mode , i.e. its representation in the spatial domain[1], the demosaicking operation, which can be seen as a succession of sub-sampling and linear interpolation, introduces artifacts coming from interpolation errors, such that the final result is a linear combination of the other 63 DCT modes. The initial mode is encoded with a larger magnitude than the others as summed up in the following expression:

$$\mathrm{DCT}\left(\mathrm{Dem}\left(\mathrm{mode_i}\right)\right) = A_i \cdot \mathrm{mode_i} + \underbrace{\sum_{i \neq j} A_j \cdot \mathrm{mode_j}}_{\text{DCT artifacts}},$$

here mode_i represents the spatial representation of DCT mode i after demosaicking (the $\mathrm{Dem}()$ function). The appearance of the A_j terms is due to small interpolation errors of mode i. These artifacts are illustrated in Figure 7. This figure can be explained as follows: in order to encode continuous waveforms that are interpolated during the demosaicking process, the interpolation process has to deal with missing values (see Figure 7a), which encode other frequencies in the DCT domain (see Figure 7c). So, instead of encoding one component (see Figure 7b), it also encodes other DCT components (see Figure 7d).

In Figure 7d, we also compare the covariance matrix computed by interpolating only the red channel on continuous DCT waveforms and the DCT of the interpolated waveform. Note that the fourth line of the covariance matrix is very similar with the components depicted in Figure 7d.

In the 2D spatial domain, for a single mode applied to a 8×8 photo-sites array, the demosaicking algorithm creates artifacts such that the resulting image in the DCT domain is a linear mixture of the different DCT modes.

[1]a.k.a. the pixel domain.

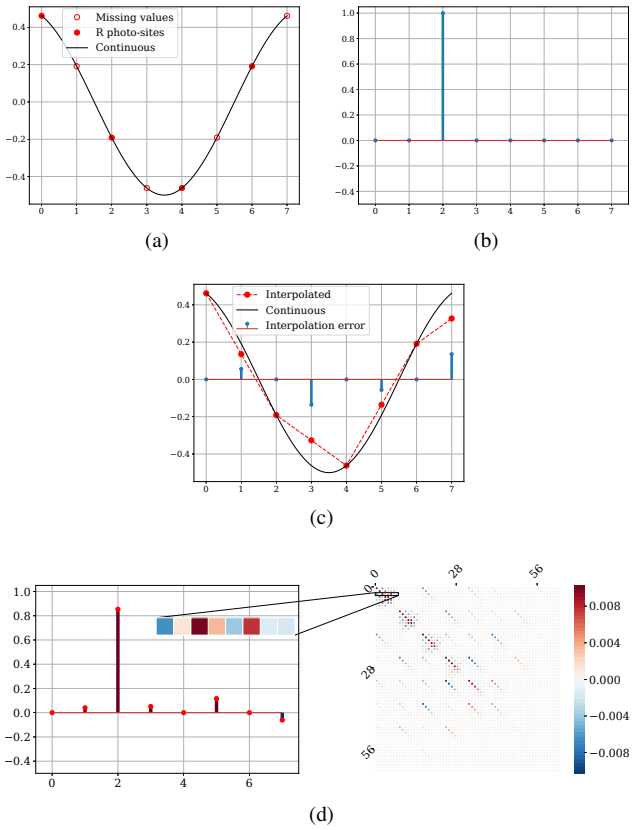
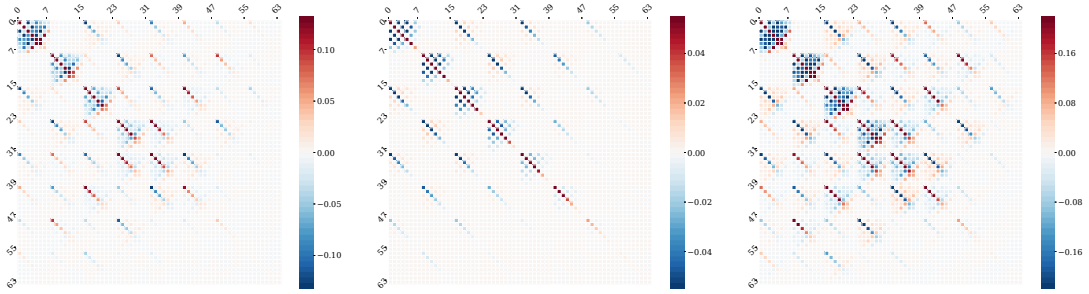


Fig. 7: Impact of demosaicking on correlation between intra-block DCT coefficients: (a) visualization of one line $\mathbf{b}_c$ of the $(0, 2)$ mode in the spatial domain. (b) $DCT(\mathbf{b}_c)$. (c) Continuous signal, interpolated signal $\mathbf{b}_i$ and interpolation error. (d) comparison between the DCT transform of the interpolated waveform (left) and the covariance matrix obtained from interpolated pure DCT modes (right).

2) Effect of low pass filtering: The second category of artifacts is due to a low-pass filter, which can be related to the conversion from RGB to luminance or to any downsampling operation. In order to simulate the effect of low pass filtering, we use a random independent noise as a RAW image and convolve this input with a standard low pass filter, such as:

$$L = \frac{1}{12} \cdot \begin{bmatrix} 1 & 1 & 1 \\ 1 & 4 & 1 \\ 1 & 1 & 1 \end{bmatrix}.$$

The covariance matrix obtained by incorporating the low-pass filter in the development process is complementary to the covariance matrix obtained considering only the demosaicking artifacts. Figure 8 shows these relationships: the total intra-covariance matrix (Figure 8c) can be approximated as the superposition of the covariance matrix of signals representing the demosaicking artifacts (Figure 8a) and the covariance matrix of the independent signal at the photo-site level undergoing low-pass filtering (Figure 8b).



(a) (b) (c)

Fig. 8: (a): Covariance matrix computed after randomly generating DCT continuous modes that are interpolated using bilinear filtering. (b): Intra correlations within a block after low-pass filtering using filter L. (c): Intra-block covariance matrix for $\mu = \text{const}$.The correlation values are thresholded for visualization purposes.

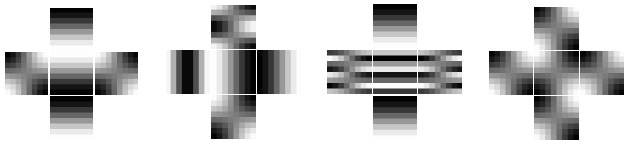
B. Inter-block correlations

Inter-block correlations between DCT coefficients are also caused by demosaicking, which averages adjacent photo-site values to interpolate the missing color values. It creates correlations between neighboring pixels, including pixels belonging to two different DCT blocks. This interpolation process highlights the low-pass component of the sensor noise, and this is consistent across different demosaicking methods (see [12]). This phenomenon is illustrated in Figure 9, which shows for different DCT modes in the spatial domain, the arrangements of blocks that are the most correlated for the horizontal and vertical neighbors. For each arrangement, we can notice that the continuity from one block to its neighbor is preserved.

The most significant correlations correspond to the surrounding vertical and horizontal blocks. This is due to the large number of neighboring photo-sites involved in the interpolation process. Note that the largest correlations are for the same vertical or horizontal frequency due to frequencies consistency between adjacent blocks.

The sign of the correlations represents the preservation of continuity between blocks in order to guarantee spatial continuity. For example, alternating signs are due to the topology of the waveforms. For example for mode $(1, 0)$, all modes $(i, 0)$ have a white top line but the bottom line alternates between white and black w.r.t. i.

It is interesting to connect this analysis with the recent steganographic scheme proposed by Li *et al.* [22] which synchronizes embedding changes between several DCT modes by empirically adjusting costs in order to favor continuities between blocks. This practical rationale is now theoretically justified by our analysis.



(a) $(1, 0)$ (b) $(0, 1)$ (c) $(0, 5)$ (d) $(1, 1)$

Fig. 9: Different DCT modes (central blocks) and their most correlated modes (represented by horizontal or vertical blocks). The presented locations of the blocks correspond to the spatial locations of the blocks. We can notice that the most correlated blocks preserve continuities between neighboring blocks.

V. Simulated Embedding

In order to perform simulated embedding, we first need to compute the probability mass function (pmf) of the embedding changes for each coefficient of the cover JPEG before performing embedding changes. We then sample according to this pmf in order to generate the quantized stego signal $\tilde{\mathbf{s}}_d$ and consequently the JPEG stego image. We recall that true embedding may also be performed by computing the costs associated with each embedding probability change, and by running a multilayer STC (see Section II-B).

In Section (III), we saw that in the DCT domain, the of the coefficients resulting from a stego signal follows a zero-mean multivariate Gaussian distribution. Its covariance matrix computed for 3×3 blocks (each block containing 8×8 DCT coefficients) is given by (24). Moreover 8-connected blocks are correlated, but two not connected blocks can be drawn independently.

In order to sample according to the joint distribution, we need to compute conditional pmfs for each quantized DCT coefficient using the four following technical developments:

1) The decomposition of the image in the DCT domain into four disjoint macro lattices (see (V-A)).
2) The use of the chain rule of conditional sampling (see (V-B)) combined with an embedding over 4×64 lattices.

3) The computation of the associated probability mass functions and associated sampling operations in the continuous and quantized domain (see V-C).
4) The computation of the embedding capacity (see V-D).

A. Decomposition into lattices

The embedding has to take into account three facts:

1) Intra-block dependencies within each 8×8 block.
2) Inter-block dependencies between one central block and its horizontal, vertical and diagonal neighbors.
3) Independence of blocks that are not neighbors.

Argument (1) means that we practically have to use 64 lattices (one per DCT mode) to perform embedding in one DCT block and (2) and (3) mean that we need a maximum of four macro-lattices $\{\Lambda_1, \Lambda_2, \Lambda_3, \Lambda_4\}$ to perform embedding in each DCT block while respecting the correlations exhibited by the computed covariance matrix.

The different macro-lattices are illustrated in Figure 10 together with the neighboring blocks that are involved.

Condider a vector of 3×3 blocks of the stego signal in the DCT domain. Let $\mathbf{s}_d^C$ be the central block and $\mathbf{s}_d^{NW}, \mathbf{s}_d^N, \mathbf{s}_d^{NE}, \mathbf{s}_d^W, \mathbf{s}_d^E, \mathbf{s}_d^{SW}, \mathbf{s}_d^S, \mathbf{s}_d^{SE}$ be respectively the north-west, north, north-east, west, east, south-west, south, and south-east blocks w.r.t. the central one.

We can build the vector of interest $\mathbf{s}^\star$, used to compute conditional probabilities (see next sub-section), as follows:

- For Λ_1, only the intra-block covariance matrix is necessary, computed w.r.t. $\mathbf{s}^\star = \mathbf{s}_d^C$,
- For Λ_2, $\mathbf{s}^\star = [\mathbf{s}_d^C, \mathbf{s}_d^{NW}, \mathbf{s}_d^{NE}, \mathbf{s}_d^{SW}, \mathbf{s}_d^{SE}]$,
- For Λ_3, $\mathbf{s}^\star = [\mathbf{s}_d^C, \mathbf{s}_d^N, \mathbf{s}_d^W, \mathbf{s}_d^E, \mathbf{s}_d^S]$,
- For Λ_4, $\mathbf{s}^\star = [\mathbf{s}_d^C, \mathbf{s}_d^{NW}, \mathbf{s}_d^N, \mathbf{s}_d^{NE}, \mathbf{s}_d^W, \mathbf{s}_d^E, \mathbf{s}_d^{SW}, \mathbf{s}_d^S, \mathbf{s}_d^{SE}]$.

We end up with a decomposition of the image into $4 \times 64 = 256$ lattices (four macro lattices and one lattice per DCT mode). In each lattice, the covariance matrix may be expressed as:

$$\mathbf{\Sigma}_d = \begin{bmatrix} \mathbf{\Sigma}_{[0:64][0:64]} & \mathbf{\Sigma}_{[0:64][64:n\times 64]} \\ \mathbf{\Sigma}_{[64:n\times 64][0:64]} & \mathbf{\Sigma}_{[64:n\times 64][64:n\times 64]} \end{bmatrix}, \tag{25}$$

with n denoting the number of blocks in $\mathbf{s}^\star$ (see footnote [2]) and $n = 1$ for Λ_1, $n = 5$ for Λ_2 and Λ_3 and $n = 9$ for Λ_4, see Figure (10).

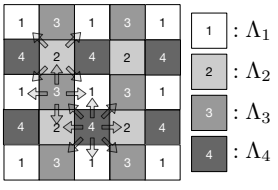


Fig. 10: The four macro lattices used for embedding. Arrows indicate the neighborhood used to compute conditional probabilities.

[2]The pythonic notation $[i : j]$ means that all indexes from the interval$[i, j-1]$ are considered.

B. Conditional sampling

Using the lattice decomposition, changes can be drawn independently according to the pmf π_i for simulated embedding in each lattice, or using a STC based on costs ρ_i (see (II-B)). In order to derive the pmf $\pi_i(k)$ for each sample i and the modification magnitude k, we need to use conditional sampling, a variation of Gibbs sampling, which enables to sample from a multivariate distribution using only conditional distributions.

Without loss of generality, if we focus on the set of 4 macro lattices defined in (V-A) (but this can be applied on any number of lattices that are conditionally independent), the chain rule of conditional probabilities gives

$$\begin{aligned} P(\mathbf{s}_d) &= P(\mathbf{s}_{\Lambda_1}, \mathbf{s}_{\Lambda_2}, \mathbf{s}_{\Lambda_3}, \mathbf{s}_{\Lambda_4}), \\ &= P(\mathbf{s}_{\Lambda_1}) P(\mathbf{s}_{\Lambda_2}|\mathbf{s}_{\Lambda_1}) P(\mathbf{s}_{\Lambda_3}|\mathbf{s}_{\Lambda_1}, \mathbf{s}_{\Lambda_2}) P(\mathbf{s}_{\Lambda_4}|\mathbf{s}_{\Lambda_1}, \mathbf{s}_{\Lambda_2}, \mathbf{s}_{\Lambda_3}). \end{aligned}$$

where $\mathbf{s}$ is a random vector representing the whole set of DCT coefficients related to the stego signal in the DCT domain, and $\mathbf{s}_{\Lambda_i}$ represents the DCT coefficients belonging to lattice Λ_i.

This means that we can perform (simulated) embedding first in lattice Λ_1 by sampling according to $P(\mathbf{s}_{\Lambda_1})$, then embed in the second lattice by sampling according to $P(\mathbf{s}_{\Lambda_2}|\mathbf{s}_{\Lambda_1})$ and so on until embedding in lattice Λ_4 by sampling according to $P(\mathbf{s}_{\Lambda_4}|\mathbf{s}_{\Lambda_1}, \mathbf{s}_{\Lambda_2}, \mathbf{s}_{\Lambda_3})$.

Conditional distribution in the continuous domain:

We explain now how we can compute the conditional probability related to a particular DCT coefficient.
For each macro lattice Λ_k, $k \in 1, .., 4$ and block ℓ, the random vector of stego signal components conditioned by the previous embeddings follows a Multivariate Gaussian Distribution: $\mathcal{N}(\mathbf{m}_{k,\ell}, \mathbf{\Sigma}_{k,\ell})$, where $\mathbf{m}_{k,\ell}$ and $\mathbf{\Sigma}_{k,\ell}$ can be computed using the Schur complement of the full covariance matrix(25 [23]). For example, if we perform the embedding in block ℓ from lattice Λ_4, the mean vector $\mathbf{m}_{4,\ell}$ and the covariance matrix $\mathbf{\Sigma}_{4,\ell}$ are computed conditionally to the embedding performed in $\{\Lambda_1, \Lambda_2, \Lambda_3\}$ (recall that the mean of $\mathbf{s}_d$ is 0):

$$\mathbf{m}_{4,\ell} = \mathbf{\Sigma}_{[0:64][64:n\times 64]} \mathbf{\Sigma}^{-1}_{[64:n\times 64][64:n\times 64]} \mathbf{s}_{\Lambda_1,\Lambda_2,\Lambda_3}, \tag{26}$$

and the Schur complement is given by:

$$\mathbf{\Sigma}_{4,\ell} = \mathbf{\Sigma}_{[0:64][0:64]} \mathbf{\Sigma}_{[0:64][64:n\times 64]} \mathbf{\Sigma}^{-1}_{[64:n\times 64][64:n\times 64]} \mathbf{\Sigma}_{[64:n\times 64][0:64]} \tag{27}$$

for the stego-signal $\mathbf{s}_{\Lambda_1,\Lambda_2,\Lambda_3}$ defined by the surrounding blocks belonging to the three first lattices (see Figure 10).

At this stage of the study, it is possible to generate the 64 stego signal values $\mathbf{s}_{k,\ell} = (c_0, \ldots, c_{63})^t_{k,\ell}$ in the DCT domain.

For each of the 64 lattices in each macro lattice, we sample by using the Cholesky decomposition of the corresponding covariance matrix $\mathbf{\Sigma}_{k,\ell}$, denoted $\mathbf{L}_{k,\ell}$, which is a lower triangular matrix such that $\mathbf{\Sigma}_{k,\ell} = \mathbf{L}_{k,\ell} \cdot \mathbf{L}^t_{k,\ell}$.

Let $(N_1, N_2, \cdots, N_{63}) \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{64})$ a standard multivariate Gaussian distribution, and $\mathbf{n} = (n_0, \ldots, n_{63})$ an outcome of it. Then $\mathbf{s}_{k,\ell} \sim \mathcal{N}(\mathbf{m}_{k,\ell}, \mathbf{\Sigma}_{k,\ell})$ can be sampled by computing $\mathbf{s}_{k,\ell} = \boldsymbol{m}_{k,\ell} + \mathbf{L}_{k,\ell}\mathbf{n}$. More precisely, because we

need to generate $\mathbf{s}_{k,\ell}$ iteratively, omitting here indexes (k,ℓ) for writing convenience, we have:

$$\begin{cases} s_0 = m_0 + L(0,0)\cdot n_0 \\ s_{1|0} = \underbrace{m_1 + L(1,0)\cdot n_0}_{m_{1|0}} + \underbrace{L(1,1)}_{\sigma^2_{1|0}}\cdot n_1 \\ \vdots \end{cases},$$

and

$$S_{i|i-1\ldots,0} \sim \mathcal{N}\left(m'_i, \sigma'^2_i\right) \quad 1 \leq i \leq 63, \tag{28}$$

with $m'_i = m_i + \sum_{l=0}^{i-1} L(i,l)n_l$, and $\sigma'^2_i = L^2(i,i)$, $i \geq 1$, $m'_0 = m_0$, $\sigma'^2{}_i = L^2(0,0)$.

Equation (28) gives consequently the conditional distribution of each sample of the stego signal in the continuous domain.

C. Computation of the probability mass functions and sampling

Using the JPEG quantization matrix, the stego signal undergoes a quantization and the conditioned probability density function has to be to converted into a probability mass function which takes into account the associated quantization table for the chosen quality factor QF. To compute $\pi_i(k) = \Pr[\bar{S}_i = k]$, the probability that the stego signal produces a change of magnitude $k \in \mathbb{Z}$ at a coefficient $i \in \mathbb{N}$ for a given block, we compute the quantized version of the real valued random variable S_i. This probability mass function is given by:

$$\begin{aligned} \pi_i(k) &= \Pr\Big[u_k < \frac{S_i}{Q_i} \leq u_{k+1}\Big], \\ &= \int_{u_k}^{u_{k+1}} \frac{1}{\sqrt{2\pi\hat{\sigma}_i^2}} \exp\Big(-\frac{(x-\hat{m}_i)^2}{2\hat{\sigma}_i^2}\Big)\mathrm{d}x, \\ &= \frac{1}{2}\Big[\mathrm{erf}\Big(\frac{u_{k+1}-\hat{m}_i}{\sqrt{2}\hat{\sigma}_i}\Big) - \mathrm{erf}\Big(\frac{u_k-\hat{m}_i}{\sqrt{2}\hat{\sigma}_i}\Big)\Big], \end{aligned} \tag{29}$$

where $u_k = [\hat{m}_i] - 0.5 + k$, $\hat{m}_i = m'_i/Q_i$, $\hat{\sigma}_i = \sigma'_i/Q_i$ for parameters m'_i and σ'_i before quantization associated with a quantization step Q_i. At each step i, the parameters m'_i and σ'_i have to be generated in the continuous domain with the knowledge of values drawn at steps $0 \leq l \leq i-1$. All the previous continuous samples are then needed to compute m'_i and σ'_i. Once a sample has been generated in the discrete domain, we need then to obtain a candidate in the continuous domain which could have led to the sampled discrete value. This could be done for example by using rejection sampling, where we can obtain for each discrete sample its continuous candidate $S_{i|\bar{c}_i}$.

Rejection sampling works in the following way: for each discrete sampled value, we sample according to the continuous distribution until we find the appropriate candidate $S_i|\bar{s}_i$ such that:

$$u_k < S_i|\bar{s}_i < u_{k+1}. \tag{30}$$

where $\bar{s}_i = k$, $u_k = [\hat{m}_i] - 0.5 + k$, and $k \in \mathbb{Z}$ the symbol sampled as a modification in the discrete domain.

Note that during this step, we need to both to embed/sample on JPEG coefficients, and to sample in the continuous domain in order to be able to compute the conditional distribution using (28), this is illustrated on Figure (11).

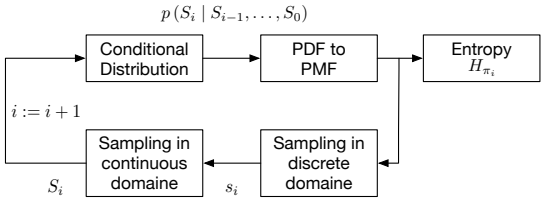


Fig. 11: Sequential computation of the PMF needed to perform simulated embedding.

D. Entropy estimation

Finally, from the probability mass function obtained in the previous section, the binary entropy associated to the steganographic signal for the i^{th} coefficient can be computed. Given the alphabet $\mathcal{A} = (-K, \ldots, 0, \ldots, K)$, $k \in \mathbb{N}^+*$, it is defined as:

$$H(\mathcal{A}, i) = -\sum_{k \in A} \pi'_i(k)\log_2\pi'_i(k), \tag{31}$$

where $\pi'_i(k) = \pi_i(k)$ for $i \in \{-K-1, \ldots, K-1\}$, $\pi'_i(-K) = -\sum_{i=-\infty}^{i=-K}\pi_i(k)$ and $\pi'_i(K) = -\sum_{i=K}^{i=+\infty}\pi_i(k)$.

E. Final embedding algorithm

Algorithm 1 J-Cov-NS embedding scheme.

- **Inputs**: the cover RAW image $\mathbf{X}_p$, the payload, a secret key
- **Develop** $\mathbf{X}_p$ in the DCT domain, before quantization to obtain $\mathbf{X}_d$ and in the JPEG domain to obtain $\mathbf{X}_j$;
- **Divide** $\mathbf{X}_p$ into 4 macro-lattices Λ_1, Λ_2, Λ_3, Λ_4;
- **For** each macro-lattice Λ_i **do**:
  - **For** each DCT block of Λ_i **do**:
    - Compute the covariance matrix for each set of DCT blocks (Eq. (24);
    - Compute the conditional mean vector (Eq. (26)) and covariance matrix (Eq. (27)) w.r.t. the embeddings done on the previous lattices;
      - ∗ **For** each DCT coefficient of $\mathbf{X}_d$ **do**:
        - · Compute the conditional distribution Eq. (28) given the previous embedding changes;
        - · Compute the PMF $\pi_i(k)$, Eq. (29);
        - · Perform the modification on $\mathbf{X}_j$ by sampling according to $\pi_i(k)$;
        - · Sample the continuous variable related to the modification, Eq (30);
- **Return** the JPEG stego image $\mathbf{Y}_j$.

The resulting embedding algorithm (named J-Cov-NS) can be decomposed into the following steps, summed up in the pseudo code presented in Algorithm 1. The use of the key

in not explicit, but it can be used to shuffle the coefficients withing each lattice. The embedded payload is such that its size matches Eq. (31).

VI. Results

This section presents a detailed benchmark of the embedding scheme on JPEG images, in the cover-source switching scenario, i.e., a scenario where the cover image comes from a higher ISO sensitivity than the image used to generate the stego image, and where the embedding mimics the ISO change.

A. Generation of E1Base

We evaluate the proposed embedding scheme to test on images taken by the Micro 4/3 16 MP CMOS sensor from the Z CAM E1 action camera.

Note that this steganalysis setup is relatively unconventional compared to the state of the art (see Figure 12). This is due to the fact that the goal of the classifier here is to distinguish between cover images captured at ISO_2 from stego images coming from cover images captured at ISO_1 but emulating sensor noise captured at ISO_2.

Raw images coming from the E1 sensor are acquired with two ISO settings (ISO 100 and ISO 200) and constitute *E1Base*. This database can be downloaded at https://gitlab.cristal.univ-lille.fr/ttaburet/e1base and is built according to the following requirements:

- It contains an equal number of images of equivalent scenes captured at both $ISO_1 = 100$ and $ISO_2 = 200$. The training and testing sets have been generated from 200 Raw images (DNG format, with a 12 bits dynamic range) that have been developed and cropped without overlapping in order to provide $10,800$ images of size 512×512. This dataset has already been used under similar circumstances in [11], [12], [16].

- A particular care has been taken in order to ensure that the only important difference between the database acquired at ISO_1 and the database acquired at ISO_2 is the sensor noise. In the same way as the MonoBase was acquired by a monochrome sensor [9], the average focus and average luminance are both similar between the two databases. This step is mandatory in order to guarantee that the steganalyzer is not using semantic information to distinguish between the cover and stego datasets. This requirement is specific to the benchmarking process of Natural Steganography since the cover and stego images do not come from the same source in this case.

For this given database, the value used to compute the variance of the sensor noise at the photo-site level are $(a_2 - a_1) = 1.15$ and $(b_2 - b_1) = -1150$ (the variance is set to zero whenever it is negative). A python notebook used to generate both the cover and the stego images is also downloadable here: https://gitlab.cristal.univ-lille.fr/ttaburet/tifs-ns/.

Classically, *E1 Base* is split into two halves, 5400 pairs of images are used for training and 5400 pairs for testing.

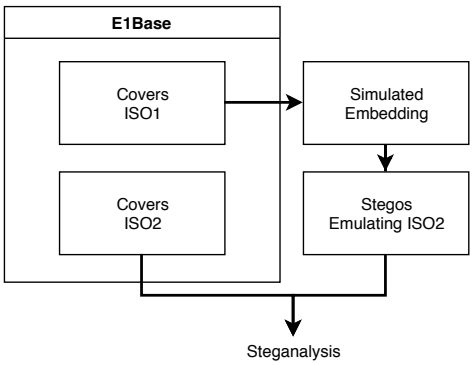


Fig. 12: Steganalysis setup when benchmarking NS.

B. Benchmark settings

We adopt the DCTR features set [24] combined with a low complexity linear classifier [25] to perform the steganalysis with the threshold set in order to minimize the total classification error probability under equal priors, $P_{\mathrm{E}} = \min_{P_{\mathrm{FA}}} \frac{1}{2}(P_{\mathrm{FA}} + P_{\mathrm{MD}})$, with P_{FA} and P_{MD} standing for the false-alarm and missed-detection rates, respectively.

For comparison with the current state of the art (of side informed schemes in the JPEG domain), we embedded all images also with SI-UNIWARD with an embedding rate of 1 bit per nzAC coefficient. In this case, the steganalysis task is the classic one: try to distinguish stegos images (produced by SI-UNIWARD) from covers acquired at ISO_2.

C. Comparison with other embedding strategies

Table I compares the proposed embedding scheme for different JPEG QF with other embedding strategies which are:

- Pseudo embedding in the photo-site domain, i.e. using Eq. (5), and applying the process depicted in the top raw of Figure 1,
- Estimating the empirical covariance matrix from a stationary signal and scaling it according to the average RGB values of the raw image, which is one solution to circumvent the explicit calculus of the covariance matrix [12],
- Embedding without taking into account correlations between DCT coefficients, this is performed by computing an empirical histogram of each DCT mode estimated after multiple embeddings and Monte-Carlo simulations [11],
- Embedding taking into account only intra-block correlations, this is performed by using only the computation of the intra-block covariance matrix, no inter-block correlations are consequently considered here,
- SI-UNIWARD [5], one state of the art embedding scheme in the JPEG domain which use side-informed embedding from the RAW image.

We can notice that computing the covariance matrix for each DCT block enables us to achieve about the same practical security than pseudo-embedding. Contrary to the previous scheme proposed in [12], which relies on an approximation

P_E (%) / JPEG QF	H (bpnzAC)	**J-Cov-NS**	Pseudo embedding (7)	Covariance scaling [12]	Independent embedding [11]	Intra-block correlations only	SI-Uniward [5] 1 bpnzAC
100	2.0	**42.9**	40.2	13.9	0.0	0.0	0.0
95	2.2	**41.2**	40.9	30.3	0.5	0.2	0.4
85	2.4	**41.2**	41.9	39.8	10.8	15.8	12.3
75	7.0	**41.6**	41.3	40.4	27.0	25.2	24.8

TABLE I: Empirical security (P_{E} in %) and average embedding capacity (H) for different quality factors and embedding strategies on E1Base. DCTR features combined with regularized linear classifier are used for steganalysis.

of the covariance matrix using a scaling factor dependent on the RGB values of each block, J-Cov-NS does not exhibit any security loss for high QFs. The comparison with independent embedding, which offers good practical security for monochrome sensors, highlights the fact that the latter scheme is not adapted to color sensors, and that it is extremely important to take into account correlations between DCT coefficients, especially for high QFs. Note also that if only the intra-block correlations are taken into account, the embedding scheme still remains highly detectable. Finally, the comparison with SI-UNIWARD shows that this state-of the art scheme is not secure for very high embedding rates (1 bit pnzAC coefficient here). This is not surprising since SI-UNIWARD does not rely on cover-source switching and does not use all the information provided by the development pipeline.

D. Evaluation for other steganalysis strategies

We also evaluated J-Cov-NS w.r.t to other steganalysis strategies dedicated to JPEG images. To this end, we performed steganalysis using another JPEG feature sets based on resudials extracted using Gabor filters (GFR, see [26]) and also using the non-linear ensemble classifier [27] for different JPEG QF. Results are presented in Table II and shows that both strategies are equivalent with the former one, with a slight advantage on GFR over DCTR features (-1% to -3%). Note however that GRF features have higher dimensionality (17.10^3 vs 8.10^3) and are longer to extract. The use of the ensemble classifier enables also gain reduce the detectability, but by a small margin of maximum 1%, together with a computational cost of about one order of magnitude.

Since steganalysis based on deep neural network offer the opportunity to automatically extract relevant features regardless of the embedding scheme, we also benchmark J-Cov-Net w.r.t. SRNet, one state of the art network in spatial or JPEG steganalysis [28]. The network was trained using mini-batches of 32 512×512 images (16 covers and 16 stego) using Nvidia GPU Quadro P6000 (24 GB of memory), the learning rate was is initially set to 10^{-3} and decreases by 10% each 5000 iterations. The sizes of the training set is 4000 pairs (augmented using rotations and flipping transforms), 1000 pairs are used for validation in order to select the best trained network, and the rest for testing. The results presented in table III are obtained after convergence is reached, i.e. after 100 000 iterations. We can notice that DNN based steganalysis enables to increase the performances in detectability by about 10% w.r.t. to DCTR combined with the low complexity linear classifier. With more than 30% of average error rate, this does not jeopardize the detectability of the presented scheme though. Note also that this improved detectability can be due to the fact that the automatic feature extraction provided by the convolutional layers of SRNet succeeds to catch possible slight general content discrepancies between images of E1Base acquired at ISO 100 and 200.

QF / P_E (%)	Linear Classifier		Ensemble Classifier	
	DCTR	GFR	DCTR	GFR
100	42.9	40.3	40.8	39.6
95	41.2	39.2	41.3	38.4
85	41.2	39.1	41.0	38.1
75	41.6	40.3	41.4	39.1

TABLE II: Practical security of J-Cov-NS for other steganalysis strategies: DCTR and GFR features sets using the Linear Classifier and the Ensemble Classifier.

QF	100	95	75
P_E (%)	37.4	31.2	35.0

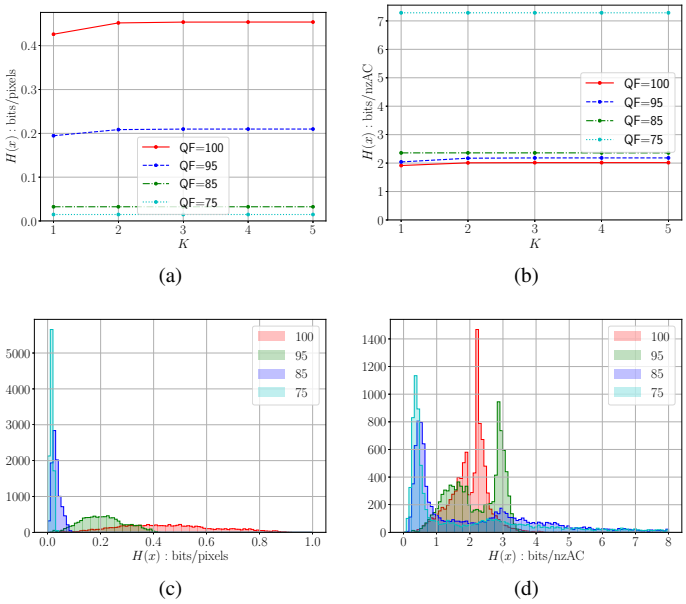
TABLE III: Practical security of J-Cov-NS against SRNet.

E. Embedding capacity

In this section, we investigate the distribution of the embedding capacity through the whole E1Base database, and compute its average value for JPEG QFs 75, 85, 95, and 100 and for different alphabet sizes. Thus, we estimate the entropy for each 512×512 image, compute the proportion of nzAC and obtain $H_{bits/pixels}$ and $H_{bits/nzAC}$ as a function of the of the chosen alphabet size for each QF. Figure 13a and Figure 13b illustrate, respectively, the evolution of $H_{bits/pixels}$ and $H_{bits/nzAC}$ when the size of the alphabet for insertion increases from $\left[\begin{array}{ccc} -1 & 0 & +1 \end{array}\right]$ to $\left[\begin{array}{ccc} -5 & \dots & +5 \end{array}\right]$.

The average embedding capacity in bits per nzAC is relatively high, around 2 bits pnzAC for JPEG QF$\in\{95,100\}$ and over 7 bits pnzAC for $QF\in\{75,85\}$. The alphabet size has a minor impact on the capacity. However, QF $\in\{75,85\}$ highlights an exotic case, since on the one hand the embedding is concentrated on the DC coefficients, and on the other hand there are only few nzAC coefficients at $QF\in\{75,85\}$. For example, given a 512×512 image with an average embedding rate of 1 bit per DC coefficient and having only 100 non-zero AC coefficients, this image has a total embedding rate of 40.96 bits per nzAC!

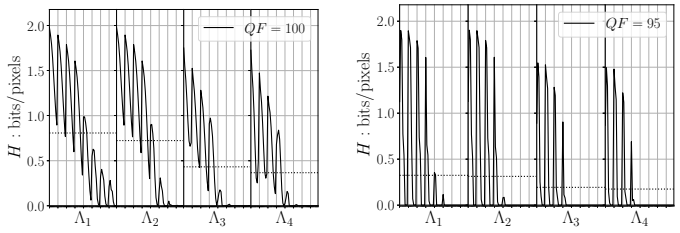
Figure (14) shows the embedding capacity computed on a synthetic constant cover RAW image for each DCT coefficient on the four lattices Λ_1, Λ_2, Λ_3, and Λ_4 at QF $=100$ and QF $=95$. Within each block, row scan is used. Two remarks can be drawn:



(a) (b) (c) (d)

Fig. 13: Average entropy H (bits) of J-Cov-NS over the database (a) per pixel, (b) per nzAC as a function of K for different JPEG QFs. Histograms of H (bits) across images for different QFs in (c) per pixel, (d) per nzAC.

1) the capacity decreases w.r.t. the coefficient frequency, this is due to demosaicking and the fact that the stego signal is mainly encoded by low frequency components. For QF $=$ 95, this is also due to the fact that the quantization steps are larger for high frequencies.
2) the capacity decreases w.r.t. the lattice index, with an average value at QF $=$ 100 of 0.8 bpp for Λ_1 to 0.4 bpp for Λ_4. This is because conditioning reduces the entropy of a random variable [29]. At QF $=$ 100, where the quantization is the same for each DCT mode, this is particularly noticeable by examining the entropy of the last 8 coefficients of each block, which are up to 0.3 bpp for Λ_1 but, due to conditioning, are reduced to zero for Λ_4.



(a) $QF = 100$ (b) $QF = 95$

Fig. 14: Evolution of the embedding rates computed from an i.i.d. Gaussian RAW image for each DCT mode and each sub-lattice for different JPEG QFs. Row scan is used within each sub-lattice. Dotted lines denote the average embedding rate within each sub-lattice.

F. Impact of the alphabet size

The impact of the alphabet size $(2K + 1)$ on the implementation of J-Cov-NS is presented in Table (IV) for different JPEG QF. We can notice that ternary embedding $(K = 1)$ is associated with a very detectable implementation for QF $=$ 95 and QF $=$ 100. This is due to the fact that the truncation of the modification changes alters considerably the distribution of the stego signal which cannot mimic anymore the ISO switch for small quantization steps. On the other hand, heptary embedding offers detectability comparable to that of an infinite alphabet for QF $=$ 95 and should be used for true embedding combined with multi-layer STC in this case. We can also notice that for QF $\leq$ 85 ternary embedding offers already the same practical security than pentary embedding.

QF / P_E in %	$K = 1$	$K = 2$	$K = 3$	$K = 5$
100	1.0	12.9	28.7	40.4
95	3.5	23.6	39.3	40.9
85	39.8	39.8	39.8	41.8
75	40.4	40.4	40.4	41.2

TABLE IV: Practical security of J-Cov-NS w.r.t. alphabet size and different QF.

G. Complexity consideration

This embedding algorithm is computationally expensive since the complexity of computing the conditional distribution increases as the complexity of the Cholesky decomposition of the covariance matrix, i.e., as $\mathcal{O}(n^3)$ where $n \leq i \times 64$, where $i = 1$ for Λ_1, $i = 5$ for Λ_2 and Λ_3, and $i = 9$ for Λ_4 (see Figure 10). On a 3.5 GHz Intel Core i7, our python implementation of simulated embedding is executed at 4000 block/s for blocks belonging to Λ_1, 30 blocks/s for Λ_2, 30 blocks/s for Λ_3 and 10 blocks/s for Λ_4. A 512×512 stego image is generated in approximately 171s without using hyper-threading.

VII. Conclusions and Perspectives

This paper draws important conclusions both in image processing and image steganography.

By deriving the covariance matrix of the random vector of stego signal components in the DCT domain, we have shown that for this basic development pipeline there are medium range correlations between DCT coefficients, and that for a given coefficient, it is correlated with the coefficients belonging to the same blocks, but also with the coefficients belonging to 8-connected blocks. Previous works on the estimation of the covariance matrix were conducted for denoising applications using non-local Bayesian estimation [30], but to the best of our knowledge, it is the first time that an analytical expression of the covariance matrix is derived in the DCT domain (i.e. Eq. (23) and (24)), exhibiting intra-block and inter-block correlations.

The derivation of the covariance matrix enables to generate a stego signal that mimics the photonic noise in the DCT domain and consequently to achieve high practical security ($P_E \geq 40\%$ for DCTR features set) while reaching high

capacity ($>$ 2 bpnzAC). In order to preserve the joint Gaussian distribution after embedding in the quantized DCT domain, the J-Cov-NS embedding scheme needs to use a large number of lattices (4×64) where conditional probability mass functions are derived for each lattice. Our experimental analysis shows that for high JPEG QF, being able to perform conditioning is essential to achieve high practical security. A similar synchronization strategy was also adopted for adaptive schemes using empirical costs in [22] and [21].

In order to bridge the gap between our proposed implementation and operational steganography, our future works will focus on different point, such that (i) the impact of non-linear developments, which may decrease the security of the scheme for important non-linearities (see [12]), (ii) decreasing the complexity by reducing the number of lattices, and (iii) designing a similar scheme for color stego images, which means that we will need to model correlations between color channels.

ACKNOWLEDGMENTS

The authors would like to thank Solène Bernard (from CRIStAL Research Center, Lille) for running the experiments using SRNet. This work was granted access to the HPC resources of IDRIS under the allocation 2020-[AP010611583] made by GENCI, and the HPC resources of University of Lille. This work has been funded in part by the French National Research Agency (ANR-18-ASTR-0009), ALASKA project: https:// alaska.utt.fr, by the French ANR DEFALS program (ANR-16- DEFA-0003).

REFERENCES

[1] C. Cachin, "An information-theoretic model for steganography," in *Information Hiding: Second International Workshop IHW'98*, Portland, Oregon, USA, April 1998.
[2] P. Sallee, "Model-based steganography," in *International Workshop on Digital Watermarking (IWDW), LNCS*, vol. 2, 2003.
[3] T. F. Tomas Pevny and P. Bas, "Using high-dimensional image models to perform highly undetectable steganography," in *Information Hiding 2010*, 2010.
[4] V. Sedighi, R. Cogranne, and J. Fridrich, "Content-adaptive steganography by minimizing statistical detectability," *Information Forensics and Security, IEEE Transactions on*, vol. 11, no. 2, pp. 221–234, 2016.
[5] V. Holub, J. Fridrich, and T. Denemark, "Universal distortion function for steganography in an arbitrary domain," *EURASIP Journal on Information Security*, vol. 2014, no. 1, pp. 1–13, 2014.
[6] B. Li, M. Wang, J. Huang, and X. Li, "A new cost function for spatial image steganography," in *Image Processing (ICIP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 4206–4210.
[7] L. Guo, J. Ni, W. Su, C. Tang, and Y.-Q. Shi, "Using statistical image model for jpeg steganography: Uniform embedding revisited," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 12, pp. 2669–2680, 2015.
[11] T. Denemark, P. Bas, and J. Fridrich, "Natural Steganography in JPEG Compressed Images," in *Electronic Imaging*, San Francisco, United States, 2018. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01687194
[8] T. Pevny, P. Bas, and J. Fridrich, "Steganalysis by subtractive pixel adjacency matrix," *Information Forensics and Security, IEEE Transactions on*, vol. 5, no. 2, pp. 215 –224, june 2010.
[9] P. Bas, "Steganography via Cover-Source Switching," 2016, IEEE Workshop on Information Forensics and Security (WIFS). [Online]. Available: https://hal.archives-ouvertes.fr/hal-01360024
[10] ——, "An embedding mechanism for Natural Steganography after down-sampling," 2017, IEEE ICASSP.

Théo Taburet received the Digital Signal and Image Processing Master degree from Cranfield University - UK, in 2017 and the Engineering diploma from the Ecole supérieure des technologies industrielles avancées (ESTIA)- France, in 2017. He is currently in his third year of PhD under the supervision of Patrick Bas and Wadih Sawaya. The subject of his thesis focuses on steganography.

Patrick Bas received the Electrical Engineering degree from the Institut National Polytechnique de Grenoble, France, in 1997, and then the Ph.D. degree in signal and image processing from Institut National Polytechnique de Grenoble, France, in 2000. He has co-organized the 2nd Edition of the BOWS-2 contest on watermarking in 2007, and the BOSS and Alaska contests on steganalysis respectively in 2010 and 2019.

Wadih Sawaya received the Engineering diploma from the Ecole Supérieure d'Ingénieur de Beyrouth (ESIB), Lebanon, and the PhD degree from the Ecole Nationale Supérieure des Télécommunications - Paris, France. He is currently an associate professor at IMT Lille Douai. His research area concerns communication security and specifically the statistical and information theoretic topics.

Jessica Fridrich is Distinguished Professor of Electrical and Computer Engineering at Binghamton University. She received her PhD in Systems Science from Binghamton University in 1995 and MS in Applied Mathematics from Czech Technical University in Prague in 1987. Her main interests are in steganography, steganalysis, and digital image forensics. Since 1995, she has received 20 research grants totaling over $12 mil that lead to more than 200 papers and 7 US patents.