A Natural Steganography Embedding Scheme Dedicated to Color Sensors in the JPEG Domain

Théo Taburet, Patrick Bas, Wadih Sawaya, Jessica Fridrich

▶ **To cite this version:**

Théo Taburet, Patrick Bas, Wadih Sawaya, Jessica Fridrich. A Natural Steganography Embedding Scheme Dedicated to Color Sensors in the JPEG Domain. Electronic Imaging 2019, Jan 2019, Burlingame, United States. hal-02005407v2

HAL Id: hal-02005407
https://hal.science/hal-02005407v2

Submitted on 26 Feb 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

A Natural Steganography Embedding Scheme Dedicated to Color Sensors in the JPEG Domain

***Théo Taburet*[×], *Patrick Bas*[×], *Wadih Sawaya*[♯], *and Jessica Fridrich*[+]**
[×] ***CNRS, Ecole Centrale de Lille, CRIStAL Lab, 59651 Villeneuve d'Ascq Cedex, France, {theo.taburet,patrick.bas}@centralelille.fr***
[♯] ***IMT Lille-Douais, Univ. Lille, CNRS, Centrale Lille, UMR 9189, France, wadih.sawaya@imt-lille-douai.fr***
[+] ***Department of ECE, SUNY Binghamton, NY, USA, fridrich@binghamton.edu***

Abstract

Using Natural Steganography (NS), a cover raw image acquired at sensitivity ISO_1 is transformed into a stego image whose statistical distribution is similar to a cover image acquired at sensitivity $ISO_2 > ISO_1$. This paper proposes such an embedding scheme for color sensors in the JPEG domain, extending thus the prior art proposed for the pixel domain and the JPEG domain for monochrome sensors. We first show that color sensors generate strong intra-block and inter-block dependencies between DCT coefficients and that theses dependencies are due to the demosaicking step in the development process. Capturing theses dependencies using an empirical covariance matrix, we propose a pseudo-embedding algorithm on greyscale JPEG images which uses up to four sub-lattices and 64 lattices to embed information while preserving the estimated correlations among DCT coefficients. We then compute an approximation of the average embedding rate w.r.t. the JPEG quality factor and evaluate the empirical security of the proposed scheme for linear and non-linear demosaicking schemes. Our experiments show that we can achieve high capacity (around 2 bit per nzAC) with a high empirical security ($P_{\mathrm{E}} \simeq 30\%$ using DCTR at QF 95).

Introduction

The class of model-based (MB) steganographic schemes, such as [13, 14], is a rational source of inspiration for secure steganography since the goal is to embed a payload while matching a statistical model. In the MB method introduced by Sallee [13], the embedding is done in order to preserve the underlying generalized Cauchy distribution of each DCT coefficient. In MiPOD [14], the goal is to minimize the statistical distortion of residuals modeled in the pixel domain using a Gaussian distribution.

Natural Steganography (NS) is a another class of model-based embedding schemes which embeds a stego signal that mimics the statistical properties of the camera shot noise. NS embeds a payload by adding noise (a stego signal) to an image acquired at ISO_1 to make it look like the image was acquired at a larger ISO sensitivity $ISO_2 > ISO_1$. 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 noise. The high security of NS schemes is also due to the fact that NS uses side-information at the embedder [2]. However, contrary to other schemes relying on side information such as SI-Uniward [9] or other side-informed implementations [6], the embedding capacity of NS is only limited by the gap between the two ISO sensitivities.[1]

In the spatial domain, implementations of NS have been proposed for monochrome sensors, which do not perform demosaicking, with development processes that apply quantization, gamma correction [2], and downsampling [3]. In the JPEG domain, our recent work [5] highlighted that models that only consider first-order marginal statistics (histograms) work well for monochrome sensors but the embedding is very detectable for color sensors.

This problem is addressed in this paper by first modeling dependencies among $3 \times 3 \times 64$ DCT coefficients from 3 blocks $\times$ 3 blocks by estimating the covariance matrix of DCT coefficients of shot noise of a given power. Then using four different sublattices and computing conditional probabilities by modeling the dependencies with a multivariate Gaussian distribution, we are able to compute the embedding probability associated with each DCT coefficient. This probability will be later transformed into costs and used with syndrome-trellis codes (STCs) [7] to perform practical embedding in the greyscale domain.

Preliminaries

Throughout this paper, we use capital letters for random variables and the corresponding lower-case symbols for their realizations. Matrices are typed in upper-case and vectors in lower-case boldface font. Matrix transpose will be denoted with a superscript "t".

We distinguish between three forms of steganographic embedding: (classical) *embedding*, *simulated-embedding* and *pseudo-embedding*. By an *embedding* scheme, we understand a practical embedding algorithm that embeds a given payload in a cover.

In *simulated-embedding*, the embedding changes are simulated according to a given selection channel – the probability $\pi(k)$ of modifying the sample in the cover at location k. For simulated embedding, practical embedding can be realized using multilayered STCs [7] based on costs directly computed from the set of embedding probabilities $\pi(k)$, but is often not implemented to reduce the implementation time. We call this output *simulated-stego* or

[1] It should be pointed out that NS needs the RAW, undeveloped image while previous side-informed steganography only uses a higher quality version of the cover, the so-called precover.

simu-stego for brevity.

Finally, *pseudo-embedding* means that the practical embedding is not possible with the proposed implementation, but that the output of the algorithm, that we call a *pseudo-stego*, is statistically distributed as the stego image.

Pseudo-embedding at the photo-site level

We first recall how to generate a pseudo-stego signal and how to perform pseudo-embedding on RAW image photo-sites.

The embedding relies on the principle that the shot noise values perturbing the photo-sites of a CCD or CMOS sensor are assumed to be independent realizations of random variables $N_{i,j}$ that follow the heteroscedastic noise model

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

where $\mu_{i,j}$ is the noiseless photo-site value at photo-site i,j, while (a_i, b_1) are a pair of constants that only depend on the ISO_1 sensitivity and the specific sensor. The acquired photo-site sample $x_{i,j}^{(1)}$ is thus a realization

$$x_{i,j}^{(1)} = \mu_{i,j} + n_{i,j}^{(1)} \tag{2}$$

of a Gaussian variable

$$X_{i,j}^{(1)} \sim \mathcal{N}(\mu_{i,j}, a_1\mu_{i,j} + b_1). \tag{3}$$

Similarly, at ISO_2 the realization is distributed as $X_{i,j}^{(2)} \sim \mathcal{N}(\mu_{i,j}, a_2\mu_{i,j} + b_2)$. However, we can also write the photo-site value of a stego image acquired at ISO_2 as

$$y_{i,j} = x_{i,j}^{(1)} + s_{i,j}, \tag{4}$$

where $S_{i,j}$ is a random variable independent of the cover and representing the stego signal necessary to mimic the image captured at ISO_2. It is distributed as

$$S_{i,j} \sim \mathcal{N}(0, (a_2 - a_1)\mu_{i,j} + b_2 - b_1). \tag{5}$$

The photo-site of the stego image is distributed as

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

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

$$Y_{i,j} \sim X_{i,j}^{(2)}. \tag{6}$$

Equation (6) shows that the distribution of a stego image pixel is the same as the distribution of a cover pixel acquired at ISO_2. Equation (4) 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 [2]. The work presented in [3, 2] 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. We next study the properties of the acquisition noise after the Discrete Cosine Transform (DCT).

Pseudo-embedding at the DCT level

We now explain the basic development pipeline shown in Figure 1 that enables us to generate pseudo-stego images in the JPEG domain. It consists of the following steps:

- Using (4), a pseudo-stego image $\mathbf{y}$ is generated in the RAW domain.
- $\mathbf{y}$ first goes through the demosaicking algorithm to generate a vector $\mathbf{y}_d$. Here, we use popular algorithms implemented by LibRaw [1], such as the bilinear interpolation, VNG, DCB, and AAHD. To preserve information, the resulting image is sampled at 16 bits per channel and stored, for example, in the TIFF format.
- Because we target embedding in grayscale images, we then convert the color image into a grayscale image by computing the luminance:

$$\mathbf{y}_l = 0.299\,\mathbf{y}_r + 0.587\,\mathbf{y}_g + 0.114\,\mathbf{y}_b, \tag{7}$$

  where $\mathbf{y}_r$, $\mathbf{y}_g$, and $\mathbf{y}_b$ denote, respectively, the red, green, and blue components (without loss of generality we assume here that there is no color correction).
- The DCT transform is then computed on 8×8 blocks:

$$\mathbf{y}_t = \mathrm{DCT}(\mathbf{y}_l). \tag{8}$$

- Finally, the DCT coefficients are quantized using quantization tables to generate a set of JPEG coefficients $\mathbf{y}_q$. We use standard quantization tables parametrized by the JPEG Quality Factor (QF).
- Quantized coefficients $\mathbf{y}_q$ are stored in a JPEG container to generate a pseudo-stego image in the JPEG format.

Modeling DCT coefficients

Rationale

In [5], the stego signal in the JPEG domain was modeled as a set of independent modifications of each DCT mode in the stego image. Practically, we used Monte-Carlo simulations and 300 pseudo-stego images to blindly estimate the empirical histogram of the stego signal for each mode, and showed that the empirical security was high for monochrome sensors but low for color sensors (particularly for high JPEG QF), which was due to the fact that we did not take into account dependencies between DCT coefficients during embedding.

Consequently, in the current paper we first analyze the dependencies between DCT coefficients within one block but also between blocks. In the development pipeline proposed in Figure 1, dependencies between DCT modes can only be introduced during the demosaicking process since

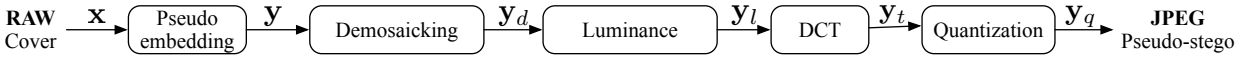


Figure 1: Development of a pseudo-stego in the JPEG domain.

pseudo-embedding at the photo-site level and luminance computation are both pixel-wise operations. Since most demosaicking algorithms only involve a small window,[2] we consider only the dependencies between neighboring blocks.

In order to perform NS in the JPEG domain, we also need to specify the joint distribution of the stego signal in the DCT domain.

If we assume that the demosaicking operation is linear, and since the stego signal at the photo-site level is Gaussian, the stego-signal $\mathbf{s}_t$ at the DCT level is distributed as a Multivariate Gaussian Distribution (MGD) $\mathbf{S}_t \sim \mathcal{N}(\mathbf{m}, \mathbf{\Sigma})$, where $\mathbf{m} \in \mathbb{R}^{9\times 64}$ and $\mathbf{\Sigma} \in \mathbb{R}^{(9\times 64)\times(9\times 64)}$ represent the mean and the covariance matrix of the MGD that models the dependencies among all 9×64 DCT coefficients from a local 3 block $\times$ 3 block neighborhood.

If the demosaicking operation is non-linear, we hope that the distribution will be close enough to a Gaussian to guarantee practical security. This issue is investigated in the Results section of this paper.

Covariance matrix estimation

In order to estimate the empirical mean $\mathbf{m}$ and the covariance matrix $\mathbf{\Sigma}$ without explicitly knowing the development pipeline, the following experiment has been conducted:

- A constant RAW image with all photo-site values 2^{12} is first generated. Note that the maximum value of each photo-site for the sensors considered in this paper is 2^{14}. The values are multiplied by 4 to be encoded with two bytes.
- A stego signal $\mathbf{s}$ at the photo-site level associated for a given pair of parameters (a, b) is added to the RAW image to simulate embedding. The resulting RAW image is stationary and can be used to estimate statistics in the developed domain.
- We develop the image using the development pipeline shown in Figure 1 to compute a vector of DCT coefficients $\mathbf{y}_t$.
- A set of N_o observations are generated by extracting 24×24 non-overlapping patches from the same developed image in order to gather 3×3 JPEG blocks. Note that since the image is stationary, can do not need generate several pseudo-stego images but instead we gather observations from the same developed image.
- The covariance matrix Σ of dimension 576×576 is finally computed from these observations. In order to get an accurate estimation of the covariance matrix, we used $N_o = 6 \times 10^4$ observations.

[2] Usually a 3×3 window.

Covariance matrix analysis

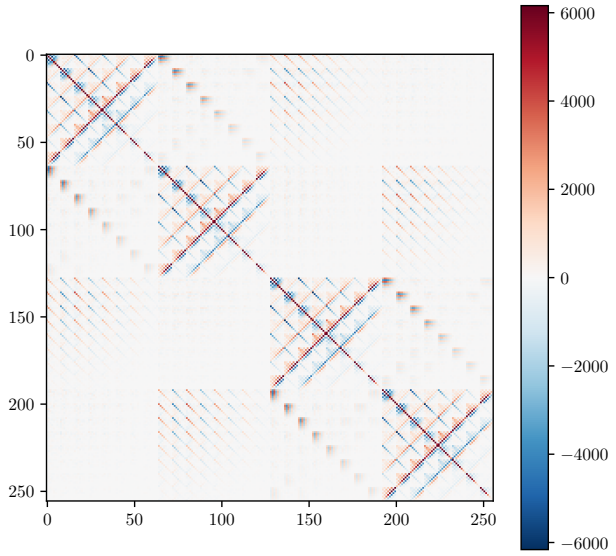
The covariance matrix $\mathbf{\Sigma}$ and the mean $\mathbf{m}$ have been computed for one color sensor (E1 action cam camera, CMOS) and one monochrome sensor (Leica M Monochrome Type 230 camera). A subset of the covariance matrix, computed on only four adjacent blocks, is illustrated for both sensors in Figure (2a) and Figure (2b), respectively. The scan order for the four 8×8 DCT blocks consists of a scan by rows within each block and a blockwise scan across the four blocks as shown in Figure (2d).

Note that while the covariance matrix of a monochrome sensor is diagonal, it exhibits many correlations between coefficients for the color sensor. This explains the negative result reported in [5] and the need to take into account dependencies between DCT coefficients for color sensors.

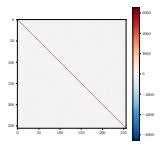
By observing Figure (2a) together with the scan order depicted in Figure (2c) and the decomposition of the matrix into different types in Figure (2d), we can decompose the entire covariance matrix into four types of matrices of size 64×64: one intra-block matrix and three inter-block matrices:

- Intra-block 8×8 covariance matrices $\mathbf{\Sigma}_I$ capture the correlations between DCT coefficients of the same block. They are located on the diagonal of the covariance matrix $\mathbf{\Sigma}$. Note that DCT coefficients can be both positively and negatively correlated.
- Horizontal inter-block covariance matrices of type $\mathbf{\Sigma}_{\rightarrow}$ or $\mathbf{\Sigma}_{\leftarrow}$ hold correlations between horizontal blocks.
- Vertical inter-block covariance matrices capture correlations between vertical blocks. They can be of type $\mathbf{\Sigma}_{\uparrow}$ or $\mathbf{\Sigma}_{\downarrow}$.
- Diagonal inter-block covariance matrices capture correlations between diagonal blocks. They can be of type $\mathbf{\Sigma}_{\nearrow}$, $\mathbf{\Sigma}_{\swarrow}$,$\mathbf{\Sigma}_{\searrow}$, or $\mathbf{\Sigma}_{\nwarrow}$.

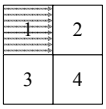
Note that with a crude examination of the covariance matrix, one might expect that $\mathbf{\Sigma}_{\nearrow} = \mathbf{\Sigma}_{\swarrow} = \mathbf{\Sigma}_{\searrow} = \mathbf{\Sigma}_{\nwarrow} = \mathbf{0}$, since theses blocks share only a few pixels in the corners, and that these matrices exhibit symmetries, such as $\mathbf{\Sigma}^t_{\nearrow} = \mathbf{\Sigma}_{\swarrow}$. Both conclusions are, however, generally incorrect, which can be explained using, for example, bilinear demosaicking. For this case, inspect Figure (3) showing the locations of photo-sites that are used to predict pixel values within one block for bilinear demosaicking. Notice, for example, that the photo-sites involved in the interpolation process are different between for the right neighboring block and the left. Additionally, the correlations involved on the NorthEast and SouthWest neighboring blocks are more important than on the NorthWest and SouthEast neighboring blocks.



(a)



(b)



(c)

Σ_I	$\Sigma_\rightarrow$	$\Sigma_\downarrow$	$\Sigma_\searrow$
$\Sigma^t_\rightarrow$	Σ_I	$\Sigma_\swarrow$	$\Sigma_\downarrow$
$\Sigma^t_\downarrow$	$\Sigma^t_\swarrow$	Σ_I	$\Sigma_\rightarrow$
$\Sigma^t_\searrow$	$\Sigma^t_\downarrow$	$\Sigma^t_\rightarrow$	Σ_I

(d)

Figure 2: (a) 256×256 covariance matrix of DCT coefficients of a color sensor and bilinear demosaicking (different sub-matrices are presented), (b) of a monochrome sensor. The values have been thresholded to the range $[-6000, 6000]$ for visualization purposes. (c): scan order by block and coefficients. (d): types of sub-matrices representing the nine covariance matrices introduced in the text.

Why these correlations happen?

As seen in the previous section, inter-block correlations between DCT coefficients are caused by demosaicking, which averages adjacent photo-site values to interpolate the missing color values and thus creates correlations between neighboring pixels and neighboring JPEG blocks.

The intra covariance matrices $\mathbf{\Sigma}_I$ exhibit patterns induced by the demosaicking algorithm itself (see Figure (2)). Next, we analyze those inter-blocks correlations that appeared to be consistent across various demosaicking methods. Figure (4) depicts for the DCT mode (1,0) the locations and correlations of the DCT modes in neighboring blocks. Based on this observation, we can draw several important remarks linked to the notions of *frequential consistency* and *spatial continuity*:

- The most significant correlations correspond to the surrounding vertical and horizontal blocks. As explained in the previous section, this is due to the number of neighboring photo-sites involved in the interpolation process.
- The largest correlations are for the same vertical or horizontal frequency due to *frequential 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 modes. 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.

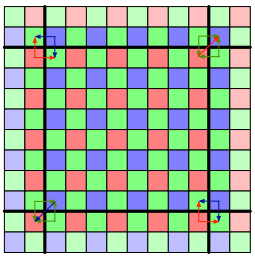


Figure 3: Locations of photo-sites (dark colors) used to predict pixel values within one block using bilinear demosaicking. Diagonal blocks are correlated either through common photosites belonging to horizontal or vertical blocks (red and blue photosite on the upper left and lower right corners) or through photosites belonging to the central block (red and blue photosite on the upper right and lower left corners or green photosite on the upper left and lower right corners).

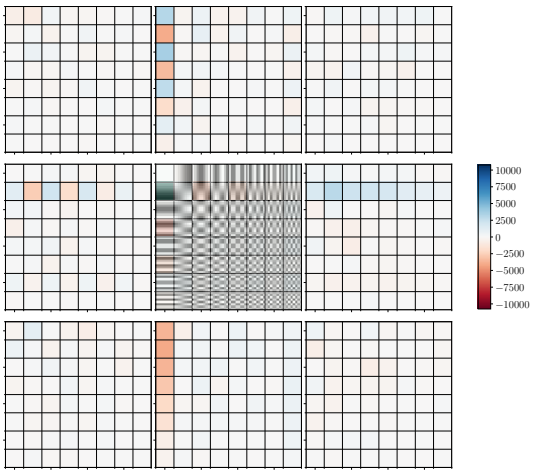
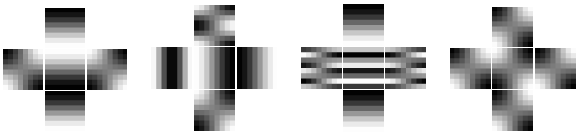


Figure 4: Analysis of the covariance matrix after bilinear demosaicking: Intra and Inter-block dependencies for mode (1,0) of the central block (in blue): positively correlated blocks are in blue and negatively correlated blocks in red.

Figure (5) shows the arrangement of the most correlated blocks based on the estimated covariance matrix after bilinear demosaicking. We can see that these arrangements preserve both frequential consistency and continuity.



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

Figure 5: Different arrangements of most correlated modes in the 4-connected pixels for different modes for bilinear demosaicking.

Relationship with previous art

It is worth pointing out that using joint or conditional probability distributions among frequency domain pairs of coefficients during embedding has recently been proposed to improve the empirical security of JPEG steganography. Li *et al.* [11] define a joint distortion function for JPEG steganography to improve the empirical security of UERD and J-UNIWARD against GFR and DCTR by using block boundary constraints during embedding. Their idea relies on the principle of block boundary continuity (BBC) in frequency domain (DCT).

In order to preserve pair block realizations, the authors used simultaneous modifications on inter-block neighbors. A $+1$ on the DCT mode would imply a $+1$ on the horizontally connected neighboring blocks and a -1 on the vertically connected neighboring blocks. This allowed them to be more consistent with block boundary dependencies. By using the dependencies exhibited in the covariances matrices, we can show that BBC assumptions are a specific case of how to use our estimated covariance matrix for embedding.

Regarding steganalysis, in JRM [10] the authors proposed features built using "unions of submodels formed as joint distributions of DCT coefficients from their frequency and spatial neighborhoods" and thus make a classifier able to take into account a wide range of statistical dependencies. The authors also showed that the strongest dependencies among DCT coefficients are those within the same block and those between the closest neighboring blocks, confirming our analysis of the covariance matrix.

Embedding algorithm (simulated)

This section presents the whole embedding algorithm, which relies on lattice decomposition, scaling the covariance matrix $\mathbf{\Sigma}$, and sampling from MGD conditioned by the already sampled neighborhood. Each step is detailed below.

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 block, its four horizontal and diagonal neighbors and possibly its four diagonal neighbors.
3. Independence of blocks that are not neighbors.

Argument (1) means that we practically have to use 64 sublattices to perform embedding in one DCT block and (2) and (3) mean that we need a maximum of four lattices $\{\Lambda_1, \Lambda_2, \Lambda_3, \Lambda_4\}$ to perform embedding on each DCT block while respecting the correlations exhibited by the estimated covariance matrix. The different lattices are illustrated in Figure 7 together with the neighboring blocks that are used to generate the stego signal. If we consider that diagonal blocks can be generated independently, then it is possible to use only three lattices. We are going to describe the implementation for four lattices but an implementation using three lattices can be proposed by considering, respectively, Λ_2 and Λ_3 as Λ_3 and Λ_4 of the four-lattice implementation.

Covariance scaling

The sampling mechanism uses the estimated covariance matrix $\mathbf{\Sigma}$ described in the previous section, which is computed for a stationary image with $\mu = 2^{12}$.

For blocks of arbitrary photo-site values, we assume that Σ is scaled w.r.t. these values, and, consequently, we avoid estimating $\mathbf{\Sigma}$ for each block. This assumption is, to some extent, valid as we shall see in the the "Results" section. It is motivated by the fact that our development pipeline is mostly linear. The covariance matrix associated with each block is consequently $\mathbf{\Sigma}' = \gamma\mathbf{\Sigma}$, where γ represents the scaling factor given by:

$$\gamma = \frac{0.299^2(a\bar{x}_r + b) + 0.587^2(a\bar{x}_g + b) + 0.114^2(a\bar{x}_b + b)}{(0.299^2 + 0.587^2 + 0.114^2)(a2^{12} + b)}, \tag{9}$$

where $\bar{x}_r$, $\bar{x}_g$, and $\bar{x}_b$ represent, respectively, the average photo-site value of the red, green, and blue component of the block that is sampled.

Simulated embedding on DCT blocks

We derive an embedding mechanism performing simulated embedding in the DCT domain and not directly in the JPEG domain (i.e., on quantized DCT coefficients). This due to computational reasons since it is faster to sample a joint-distribution on one DCT block than to sample individually on each JPEG coefficient. Indeed, the latter necessitates computing 64 conditional distributions and discretizing them. However, the approximation of the embedded payload presented in the next section shows that practical embedding is possible. Figure 6 shows the differences between our implementation and practical and simulated embedding, the simu-stego image is generated by modifying DCT coefficients of the cover image.

Consequently, the embedding scheme directly generates the stego-signal on DCT blocks, i.e., on vectors of 64 DCT coefficients, by sampling according to a MGD and by taking into account the already sampled coefficients on the neighboring blocks. The computation of the conditional MGD is done using the covariance matrix computed for each block $\mathbf{\Sigma}'$ and the Schur complement of the covariance matrix $\mathbf{\Sigma}'$ (see [12]).

In our experiments, we have directly sampled a whole 8×8 DCT block $\mathbf{B}_i$ from the conditional distribution

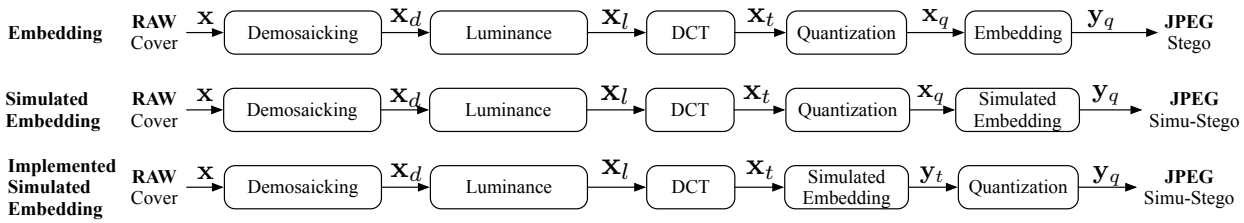


Figure 6: Differences between classical embedding, simulated embedding, and our implementation.

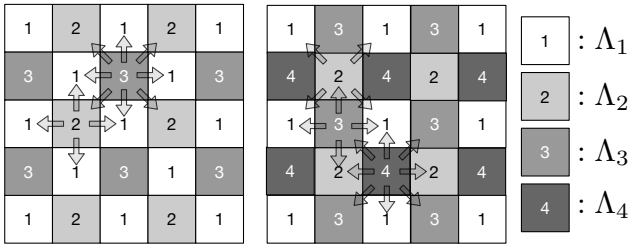


Figure 7: Two possible arrangements using either three lattices (left) or four lattices (right). Arrows indicate the neighborhood used to compute conditional probabilities.

$\mathcal{N}(0,\gamma\mathbf{\Sigma}_k)$ where γ is the scaling factor computed in (9) and $\mathbf{\Sigma}_k$, $k \in \{1,2,3,4\}$, are 64×64 covariance matrices computed differently for each lattice Λ_k.

For Λ_1, $\mathbf{\Sigma}_1$ is the covariance matrix of a DCT block ($\mathbf{B}_i^{(1)} \sim \mathcal{N}(0,\gamma\mathbf{\Sigma}_1)$) where $\mathbf{\Sigma}_1$ are the first 64×64 elements of $\mathbf{\Sigma}$.

For Λ_2, Λ_3, and Λ_4, respectively, we compute the conditional joint distributions conditioned w.r.t. the previously sampled blocks in the neighborhood (see also Figure 7):

$$\mathbf{B}_i^{(2)}|\mathbf{B}_i^{\nwarrow},\mathbf{B}_i^{\nearrow},\mathbf{B}_i^{\searrow},\mathbf{B}_i^{\swarrow} \quad\sim\quad \mathcal{N}(\mathbf{m}_i,\gamma_i\mathbf{\Sigma}_2), \tag{10}$$

$$\mathbf{B}_i^{(3)}|\mathbf{B}_i^{\leftarrow},\mathbf{B}_i^{\rightarrow},\mathbf{B}_i^{\downarrow},\mathbf{B}_i^{\uparrow} \quad\sim\quad \mathcal{N}(\mathbf{m}_i,\gamma_i\mathbf{\Sigma}_3), \tag{11}$$

$$\begin{array}{l}\mathbf{B}_i^{(4)}|\mathbf{B}_i^{\leftarrow},\mathbf{B}_i^{\rightarrow},\mathbf{B}_i^{\downarrow},\mathbf{B}_i^{\uparrow},\\ \quad\mathbf{B}_i^{\nwarrow},\mathbf{B}_i^{\nearrow},\mathbf{B}_i^{\searrow},\mathbf{B}_i^{\swarrow} \quad\sim\quad \mathcal{N}(\mathbf{m}_i,\gamma_i\mathbf{\Sigma}_4),\end{array} \tag{12}$$

where $\mathbf{m}_i$ denotes the expectation computed w.r.t. the previoulsy sampled blocks. Note that a practical embedding will also rely on computing the conditional probabilities on a fine grid of 64 lattices (one for each DCT coefficient of the JPEG block) with STCs using q-ary embeddings whose costs are computed from the quantized probability distributions as executed in [2].

Embedding capacity

The overall purpose of this section is to estimate the average payload size after embedding for different JPEG quality factors. Since the variance of the stego signal relies on the photo-site value (cf (5)), the embedded message size is consequently different for each cover. Using optimal coding, or close to optimal codes such as the STCs [7], we can show that the payload size is close to the entropy related to the embedding change probabilities of each JPEG coefficient.

The different elements involved in the computation of the entropy are summarized in Figure 8, and are listed below :

1. Computation of the marginal conditional distribution $p(C_i \mid C_{i-1},\ldots,C_1)$ w.r.t. the already sampled coefficients $[c_{i-1},\ldots,c_1]$ and the covariance matrix $\mathbf{\Sigma}$ associated with each lattice.
2. Discretization of the conditional distribution to convert the pdf into a pmf $\pi_i(k) = \Pr[S_i = k]$.
3. Computation of the discrete entropy $H_{\pi_i} = -\sum_k \pi_i(k)\log_2\pi_i(k)$.
4. Summation of the entropies computed for each coefficient.

We now detail each of these steps.

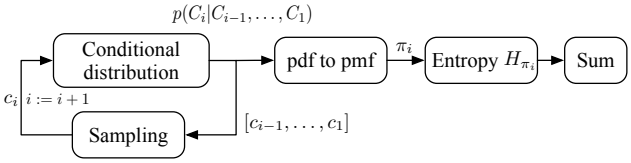


Figure 8: Elements needed to compute the average capacity.

Conditional distribution

With a MGD on i random variables of expectation $\boldsymbol{\mu} = [\mu_1,\ldots,\mu_i]^T$ and covariance matrix

$$\mathbf{\Sigma} = \begin{bmatrix} \mathbf{\Sigma}_{(1:i-1)(1:i-1)} & \mathbf{\Sigma}_{(1:i-1)i} \\ \mathbf{\Sigma}_{i(1:i-1)} & \mathbf{\Sigma}_{ii} \end{bmatrix},$$

the conditional probability $p(C_i \mid C_{i-1},\ldots,C_1)$ is Gaussian and distributed as $\mathcal{N}(\bar{\mu_i},\bar{\Sigma}_i)$ with :

$$\bar{\mu_i} = \mu_i + \mathbf{\Sigma}_{i(1:i-1)}\mathbf{\Sigma}^{-1}_{(1:i-1)(1:i-1)}(\mathbf{c}-\mathbf{m}), \tag{13}$$

and,

$$\bar{\Sigma}_i = \mathbf{\Sigma}_{ii} - \mathbf{\Sigma}_{i(1:i-1)}\mathbf{\Sigma}^{-1}_{(1:i-1)(1:i-1)}\mathbf{\Sigma}_{(1:i-1)i}, \tag{14}$$

for $\mathbf{c} = [c_1,\ldots,c_{i-1}]^T$ and $\mathbf{m} = [\mu_1,\ldots,\mu_{i-1}]^T$.

It is important to notice that in order to compute $(\bar{\mu_i},\bar{\Sigma}_i)$ we need to have sampled already coefficients from

other lattices or from the same block. For example, to compute the distribution of the last DCT coefficient of a block belonging to the 4^{th} lattice, the dependencies between blocks of different lattices imply to have already sampled 12 blocks of Λ_1, 6 blocks of Λ_2 and 4 blocks of Λ_3 together with 63 coefficients of the current block. Figure 9 shows the locations of the different blocks.

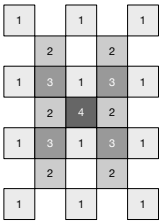


Figure 9: JPEG blocks, associated with their lattice, involved in the computation of the capacity for a block belonging to Λ_4.

Computation of the probability mass function

The stego signal is quantized by the JPEG coding algorithm, thus the conditioned probability density functions must be converted into a probability mass function that takes into account the quantization table associated with a given quality factor. We denote here q_i the discrete quantized version the real valued random variable $(C_i \mid C_{i-1},\dots,C_1)$, q_i is a discretized Gaussian random variable with probability mass function :

$$\begin{aligned}\Pr(q_i=[\mu_i]+k) &= \Pr(u_k < C_i/Q \le u_{k+1}),\\ &= \int_{u_k}^{u_{k+1}} \frac{1}{\sqrt{2\pi\sigma_i^2}} \exp - \frac{(x-\mu_i)^2}{2\sigma_i^2} dx,\\ &= \tfrac{1}{2}\mathrm{erf}\left(\tfrac{u_{k+1}-\mu_i}{\sqrt{2}\sigma_i}\right)\\ &\quad -\tfrac{1}{2}\mathrm{erf}\left(\tfrac{u_k-\mu_i}{\sqrt{2}\sigma_i}\right),\end{aligned} \tag{15}$$

where $\mu=\mu'/Q$ and $\sigma=\sigma'/Q$ for parameters μ' and σ' before quantization associated with a quantization step Q, and $u_k=[\mu_i]+k-0.5$.

Computation of the entropy

Given the alphabet $A=[-K,\dots,0,\dots K]$, the entropy is directly related to the $(2K+1)$-ary alphabet. We perform a truncation of the pmf in order to match the length of the $(2K+1)$-ary alphabet and denote $\pi_i(k)$ the probability for the i^{th} coefficient to add k to the DCT coefficient. According to the number of symbols and to the probability mass function, the probability that for the i^{th} coefficient the stego signal produce $+k$ on the coefficient is normalized as :

$$\pi_i(k)=\frac{\Pr(q_i=[\mu_i]+k)}{\sum_{l\in A}\Pr(q_l=[\mu_i]+l)}. \tag{16}$$

Thus the entropy associated with the steganographic signal for the i^{th} coefficient can be written as follows :

$$H(K,i)=-\sum_{k=-K}^{K}\pi_i(k)\log_2\pi_i(k). \tag{17}$$

Results

We evaluate in this section the capacity of the proposed scheme, its practical security, and the impact of the demosaicking algorithms and the alphabet size.

We used a switch from sensitivity ISO 100 to ISO 200 from one CMOS sensor from the Z CAM E1 action camera as in [5]. If the demosaicking scheme is not explicitly mentioned, bilinear interpolation is used.

Capacity

The tests presented have been conducted for an average photo-site value of 2^{12} for a 14 bit sampling. We first study the impact of the conditioning on the entropy and the lattice index.

Figure 10 shows the evolution of the capacity in bits/nzAC for each lattice and different alphabet size at QF 95. It is important to notice that there is almost no gain beyond pentary embedding ($K=2$) and that conditioning w.r.t. the 8 neighboring blocks (Λ_4) decreases the entropy by about 20% w.r.t. considering only intra-block dependencies (Λ_1). If the impact of using four lattices on the embedding rate is small, we shall notice in the next section that the impact of dealing with inter-block dependencies in terms of practical security is important.

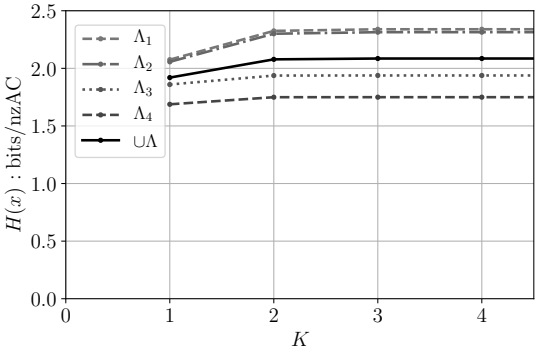


Figure 10: Embedding capacity as a function of K, QF=95, bilinear demosaicking.

Figure 11 presents the impact of the JPEG QF on the theoretical embedding rate for a range of QFs between 75 and 100, we also compare independent embedding which, the approach proposed in [5], which is equivalent to use a diagonal covariance matrix, with the 4 lattices implementation. The embedding rate stays large even for low quality factors because we measure it in bits per non-zero AC DCT coefficient. It is also interesting to notice that taking into account correlations between coefficients reduces the capacity from 0.8 bits/nzAC at QF 95 to 2 bits/nzAC at QF 75 while increasing also the practical security (see next section).

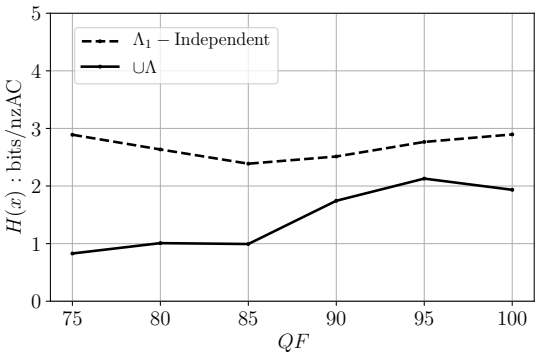


Figure 11: Embedding capacity as a function of QF for $K = 3$, bilinear demosaicking.

Practical security

To assess the empirical security of the proposed scheme, the stego images were generated from cover images captured at $ISO_1 = 100$ while our embedding strives to mimic the statistical properties of images captured at $ISO_2 = 200$. Steganalysis was performed using the DCTR features [8] and the low complexity linear classifier [4] with the threshold set to minimize the total classification error probability under equal priors, $P_{\text{E}} = \min_{P_{\text{FA}}} \frac{1}{2}(P_{\text{FA}} + P_{\text{MD}})$, with P_{FA} and P_{MD} standing for the false-alarm and missed-detection rates, respectively.

For SI-UNIWARD, the embedding rate is set to its maximum, i.e., 1 bit per nzAC coefficient.

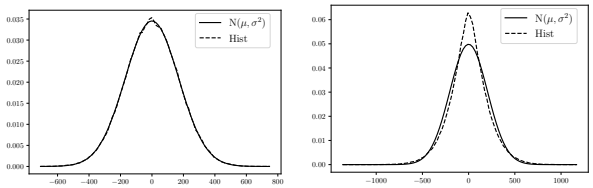
The results are shown in Table 1. Our implementation using lattice embedding increases the practical security w.r.t. embedding where the dependencies between blocks and between coefficients (respectively denoted "Intro only" and "Independent embedding" in the table) are not taken into account. The difference between using either 3 or 4 lattices, i.e. considering or not that diagonal blocks can be processed independently, also appears to be important.

Additionally, we show that the practical security of the scheme decreases with increasing JPEG quality factor. This is due to the fact that the generation of the stego signal is still distinguishable from the sensor noise as high quality JPEG compression preserves more complex properties of covers. The adopted scaling of the covariance matrix w.r.t. the RGB components (see (9)) also probably contributes to increased statistical detectability. The comparison with SI-UNIWARD shows that this classical embedding strategy is more detectable and has a smaller embedding rate.

Non-linear demosaicking algorithms

One advantage of the covariance matrix estimation over explicit calculus is that we can blindly extract the covariance matrices for any demosaicking scheme, including development processes that are not publicly known. In this experiment, we evaluate the practical security for three demosaicking processes that are not linear (VNG, DCB, and AAHD). Note that the stego signal is generally not following a MGD for non-linear demosaicking. Table 2 presents the impact of the demosaicking step on detectability. Depending on the process, the detectability varies quite significantly, e.g., compare the classifier errors for AAHD and VNG.

Interestingly, when we compare the detectability with the statistical distribution of DCT modes (see Figure (12)), we can see that for demosaicking schemes with high detectability, such as AAHD, the empirical histogram and the Gaussian distribution with same mean and variance have very different distributions. On the contrary, VNG, which increases the detectability only slightly over Bilinear demosaicking, is associated with an empirical histogram thatis close to the Gaussian distribution. Consequently, one drawback of the proposed approach is that it cannot be used with demosaicking algorithms that have strong nonlinearities with the exception of low QFs.



(a) VNG (b) AAHD

Figure 12: Practical histogram of mode $(7, 0)$ vs. Gaussian distribution for two different demosaicking algorithms.

Impact of the alphabet size

The impact of the alphabet size K is now measured in terms of practical security. Table (3) presents the results w.r.t the JPEG QF for the proposed implementation. Notice that using only ternary embedding makes the scheme very detectable for QF 95. In fact, this setup required heptary embedding to offer detectability comparable to that of an infinite alphabet. On the contrary, ternary embedding is already sufficient for QF 75.

Conclusions and perspectives

To conclude, we first have to mention that even if NS in the JPEG domain may sound attractive from the the capacity and the security point of view, its practical implementation is not straightforward. As described in this paper, in order to deal with the dependencies between DCT coefficients, we need to use 4×64 lattices. Furthermore, embedding in each lattice requires computing costs from the embedding change probabilities given by (15). Note that these probabilities are also computed considering embedding changes performed on other lattices in order to take into account the MGD. This combination of mechanisms is of course implementable, but the computational complexity may be excessive since the complexity of computing the conditional distribution increases as the complexity of the Cholesky decomposition of the covariance matrix, i.e., as $O(n^3)$ where $n \leq 8 \times 64$.

However, we claim that there are interesting lessons to learn from this study. First, we have shown that embedding in images originating from a color sensor needs more

P_E (%) / JPEG QF	Pseudo embed.	4 Lattices embedding	3 Lattices embedding	Intra only	Independent embedding	SI-Uniward, [5]
95	41.1	30.3	17.6	0.8	0.5	9.57
85	41.4	39.8	36.6	13.1	10.8	7.52
75	43.0	40.4	40.2	30.2	27.0	0.032

Table 1: Empirical security (P_{E} in %) for different quality factors and embedding strategies on E1Base with bilinear demoisaicking. DCTR features combined with regularized linear classifier are used for steganalysis.

QF / P_E in %	Bilinear	VNG	DCB	AAHD
95	30.3	22.7	4.5	3.4
85	39.8	36.9	32.6	25.4
75	40.4	40.9	39.8	35.7

Table 2: Practical security for different demosaicking algorithms (MGD model, 4 lattices).

QF / P_E in %	$K=1$	$K=2$	$K=3$
95	2.8 / 3.5	19.3 / 23.6	29.5 / 39.3
85	39.8	39.8	39.8
75	40.4	40.4	40.4

Table 3: Practical security w.r.t. alphabet size K.Right and left values are respectively for 4-lattice embedding and pseudo-embedding when values differ. For QF 85 and QF 75, the 2 implementation give identical results.

sophisticated strategies than for a monochrome sensor. Additionally, we have shown here that if the correlations between DCT coefficients are not taken into account, this can have an important impact on practical detectability. The need to take into account dependencies has also been reported by others both in the pixel and JPEG domains. Thus, we believe that our analysis will be important for development more realistic synchronization strategies in the future.

Another interesting conclusion is that the demosaicking algorithm creates a statistical fingerprint that needs to be modeled correctly in order to preserve undetectability. If the demosaicking does not preserve the Gaussianity of the stego signal in the DCT domain, our approach becomes more detectable since the embedding scheme is unable to fit the correct distribution.

Finally, we wish to stress for practical use that NS in JPEG domain for low JPEG quality factors enjoys higher security independently of the alphabet size (ternary is sufficient) or of the demosaicking algorithm.

Acknowledgments

The work on this paper was supported by NSF grant No. 1561446 and by DARPA under agreement number FA8750-16-2-0173. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation there on. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied of DARPA or the U.S. Government.

This work was also partially supported by the French ANR DEFALS program (ANR-16-DEFA-0003) and the French ANR ASTRID program (ANR-18-ASTR-0009).

Author Biography

Théo Taburet received the Digital Signal and Image Processing Master degree from Cranfield University - UK, in 2017 and the Engineering diploma from the École supérieure des technologies industrielles avancées - France, in 2017. He is currently in his second 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 first edition of the BOSS contest on steganalysis in 2010. From 2013 to 2016 Patrick Bas was associate editor of IEEE Transactions of Information Forensics and Security (IEEE TIFS). Patrick Bas is the current group leader of the team working on Signal and Images in the CRISTAL Lab.

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 $11 mil that lead to more than 180 papers and 7 US patents.

References

[1] Libraw raw image decoder. http://www.libraw.org, 2010.

[2] Patrick Bas. Steganography via Cover-Source Switching. 2016. IEEE Workshop on Information Forensics and Security (WIFS).

[3] Patrick Bas. An embedding mechanism for Natural Steganography after down-sampling. 2017. IEEE ICASSP.

[4] Rémi Cogranne, Vahid Sedighi, Jessica Fridrich, and Tomáš Pevnỳ. Is ensemble classifier needed for steganalysis in high-dimensional feature spaces? In *Information Forensics and Security (WIFS), 2015 IEEE International Workshop on*, pages 1–6. IEEE, 2015.

[5] Tomáš Denemark, Patrick Bas, and Jessica Fridrich. Natural Steganography in JPEG Compressed Images. In *Electronic Imaging*, San Francisco, United States, 2018.

[6] Tomas Denemark and Jessica Fridrich. Side-informed steganography with additive distortion. In *Information Forensics and Security (WIFS), 2015 IEEE International Workshop on*, pages 1–6. IEEE, 2015.

[7] Tomas Filler, Jan Judas, and Jessica Fridrich. Minimizing additive distortion in steganography using syndrome-trellis codes. *Information Forensics and Security, IEEE Transactions on*, 6(3):920–935, 2011.

[8] Vojtěch Holub and Jessica Fridrich. Low-complexity features for jpeg steganalysis using undecimated dct. *IEEE Transactions on Information Forensics and Security*, 10(2):219–228, 2015.

[9] Vojtěch Holub, Jessica Fridrich, and Tomáš Denemark. Universal distortion function for steganography in an arbitrary domain. *EURASIP Journal on Information Security*, 2014(1):1–13, 2014.

[10] Jan Kodovskỳ and Jessica Fridrich. Steganalysis of jpeg images using rich models. In *IS&T/SPIE Electronic Imaging*, pages 83030A–83030A. International Society for Optics and Photonics, 2012.

[11] Weixiang Li, Weiming Zhang, Kejiang Chen, Wenbo Zhou, and Nenghai Yu. Defining joint distortion for jpeg steganography. In *Proceedings of the 6th ACM Workshop on Information Hiding and Multimedia Security*, pages 5–16. ACM, 2018.

[12] A. Papoulis and U. Pillai. *Probability, Random Variables and Stochastic Processes*. Mac Graw Hill, 2002.

[13] P. Sallee. Model-based steganography. In *International Workshop on Digital Watermarking (IWDW), LNCS*, volume 2, 2003.

[14] Vahid Sedighi, Rémi Cogranne, and Jessica Fridrich. Content-adaptive steganography by minimizing statistical detectability. *Information Forensics and Security, IEEE Transactions on*, 11(2):221–234, 2016.