Digital image steganography using stochastic modulation

Jessica Fridrich* and Miroslav Goljan
Department of Electrical and Computer Engineering, SUNY Binghamton, Binghamton, NY 13902-6000, USA

ABSTRACT

In this paper, we present a new steganographic paradigm for digital images in raster formats. Message bits are embedded in the cover image by adding a weak noise signal with a specified but arbitrary probabilistic distribution. This embedding mechanism provides the user with the flexibility to mask the embedding distortion as noise generated by a particular image acquisition device. This type of embedding will lead to more secure schemes because now the attacker must distinguish statistical anomalies that might be created by the embedding process from those introduced during the image acquisition itself. Unlike previously proposed schemes, this new approach, that we call stochastic modulation, achieves oblivious data transfer without using noise extraction algorithms or error correction. This leads to higher capacity (up to 0.8 bits per pixel) and a convenient and simple implementation with low embedding and extraction complexity. But most importantly, because the embedding noise can have arbitrary properties that approximate a given device noise, the new method offers better security than existing methods. At the end of this paper, we extend stochastic modulation to a content-dependent device noise and we also discuss possible attacks on this scheme based on the most recent advances in steganalysis.

Keywords: Steganography, steganalysis, stochastic modulation, device noise

1. INTRODUCTION

The purpose of steganography is to hide the very presence of communication by embedding messages into innocuous-looking cover objects, such as digital images. To accommodate a secret message, the original cover image is slightly modified by the embedding algorithm to obtain the stego image. The embedding process usually incorporates a secret stego-key that governs the embedding process and it is also needed for the extraction of the hidden message.

In contrast to watermarking when the embedded message has a close relationship to the cover image supplying data, such as sender or receiver information, authentications codes, etc., in steganography, the cover image is a mere decoy and has no relationship to the hidden data. The most important requirement for a steganographic system is undetectability: stego images should be statistically indistinguishable from cover images. In other words, there should be no artifacts in the stego image that could be detected by an attacker with probability better than random guessing, given the full knowledge of the embedding algorithm except for the stego-key (Kerckhoff's principle).

The early steganographic schemes focused on introducing as little distortion in the cover image as possible utilizing the seemingly intuitive heuristics that the smaller the embedding distortion is, the more secure the steganographic scheme becomes. However, recent advances in steganalysis clearly showed that this is not the case. The Least Significant Bit embedding (LSB) with sequential or random message spread has been successfully attacked even for very short messages[2,3,11]. In essence, the LSB embedding is so easily detectable because it introduces distortion that never naturally occurs to images and creates an imbalance between appropriately defined statistical quantities. A better approach is to replace the operation of flipping the LSBs by randomly adding 1 or –1 to pixels (+–1 embedding) and extracting the message bits from LSBs as in the classical LSB embedding. This is the embedding algorithm of Hide[8] and it has also been accepted (in a slightly different version) for steganography in the JPEG format[10]. It turns out that this simple modification of the LSB embedding paradigm is, in fact, much more difficult to detect[4,11].

* fridrich@binghamton.edu; phone: 1 607 777-2577; fax: 1 607 777-4464; http://www.ssie.binghamton.edu/fridrich; SUNY Binghamton; T. J. Watson School of Engineering, Dept. of Electrical Engineering, Binghamton, NY USA 13902-6000

The +–1 embedding is a special case of our stochastic modulation when the noise η added to the cover image has the following probability distribution P: $P(\eta = -1) = p/2$, $P(\eta = 1) = p/2$, $P(\eta = 0) = 1-p$ (assuming the message is a random bit-stream and $100p$ % of pixels were used for embedding). Notice that in +–1 embedding, the message bits are still encoded and extracted as LSBs of pixels. In this paper, we show how to extend this embedding archetype to a noise with an arbitrary probabilistic distribution P defined on an arbitrary set of integers. The algorithm that achieves this is called stochastic modulation.

The embedding party (Alice) can use stochastic modulation, for example, in the following way. Alice will carry out experiments on her source of cover images and estimate the properties of the noise present in them. If Alice's acquisition device is a digital camera, the noise depends on the exposition time, the amount and type of ambient light at the scene, usage of a flash, the specific CCD sensor and camera circuitry, interpolation algorithms in camera's hardware, etc. The sensor and hardware noise are known to be well modeled by an i.i.d. Gaussian noise[5]. Because there is in general a great variation in the amount of noise in the images due to the multitude of contributing effects mentioned above, one could slightly increase the amount of noise without introducing any easily detectable statistical artifacts. This idea is at the base of our stochastic modulation presented in this paper.

Determining the actual security of stochastic modulation, however, is not an easy task due to the fact that we are adding a quantized noise to an already quantized (and processed) signal rather than at the point of acquisition when the light hits the CCD sensor. This issue is also discussed in the paper.

Before we close this introduction, we note that, similar to stochastic modulation, DSSS (Direct Sequence Spread Spectrum) embedding, that is widely used for robust watermarking, also superimposes message-modulated noise on the image. However, DSSS cannot be simply turned into a high-capacity non-robust embedding needed for steganography due to the correlation-based message extraction.

The paper is organized as follows. In the next section, we give a brief overview of related methods proposed in the past. Then, in Section 3, we describe the main ideas behind stochastic modulation and in Section 4 we discuss some important implementation issues that need to be considered to establish practical communication. In Section 5, we investigate the security of the proposed algorithm from the point of view of recent advances in steganalysis. Stochastic modulation is extended to a content-dependent noise in Section 6. Finally, in Section 7 we conclude the paper and outline possible future research directions.

2. RELATED METHODS

In the past, several researchers attempted to design steganographic schemes that embed messages by adding Gaussian noise to the image. Marvel et al.[7] describe a high-capacity method for embedding message bits in uncompressed raw image formats. A special non-linear transformation together with the message bits is used to generate a Gaussian signal that is added to the cover image. The purpose of the transformation is to maximize the separation between two samples of a random Gaussian variable that encode a 0 and a 1. The message detector first applies an adaptive Wiener filter to the image to estimate the noise component. The noise component then determines the embedded bit via the inverse embedding function. The amplitude of the added Gaussian signal, however, must be large enough to minimize the errors during bit extraction. This forces the user to increase the amplitude of the added noise, which in turn decreases the security of the steganographic method. Error correction is also a necessity to guarantee error-free bit extraction even when no distortion is present. This further decreases the capacity of the method. We acknowledge that this work was focused on the capacity-robustness issue rather than security.

Alturki's[1] approach is a simple bit-replacement of quantized DCT coefficients calculated from a randomly permuted image. The key-dependent permutation serves as a pre-whitening and distributes the image energy evenly over the whole spectrum. Consequently, the quantization noise appears to be Gaussian although no formal proof of this is given. In Alturki's method, truncation of grayscales at 0 or 255 may introduce read-out errors during the decoding. The author mentions that the problem can be dealt with by applying error-correction to his scheme. However, the error-correcting scheme will further decrease the capacity of the method. Because the maximal bit error rate highly depends on the image, it is not easy to find fixed error bounds. Also, the method cannot be easily generalized to make the distortion follow an arbitrary probability distribution that would approximate a non-Gaussian device noise.

According to the best knowledge of the authors of this paper, no steganographic method has so far been proposed that would provide a high embedding rate (e.g., above 0.5 bpp) and could be interpreted as adding noise of predefined properties, including the proof that the distortion really has the required statistical properties. In the next section, we describe a simple high-capacity steganographic method that embeds message bits by adding a noise signal with an arbitrary distribution or even a content-dependent noise. The method does not need any error-correction scheme or any computationally expensive image processing or transformations.

3. STOCHASTIC MODULATION

In Subsection 3.1, we describe a high-capacity steganographic method that embeds message bits into individual pixels by adding to the cover image a noise signal with a probabilistic distribution that is symmetrical about zero. Generalization to an arbitrary noise distribution is presented in Subsection 3.3. Throughout this text, we assume that the cover image is an 8-bit grayscale image.

First, note that if $\{s_i\}$ is a normally distributed Gaussian sequence $N(0,\sigma)$ and if z_i is a random variable uniformly distributed in $\{-1, 1\}$, then $\{z_i s_i\}$ is also $N(0,\sigma)$. In other words, a Gaussian sequence with randomized signs stays Gaussian. This statement is true for any random variable with a distribution symmetrical about zero.

Suppose the message m_i consists of a random sequence of 1's and –1's (m_i has zero mean). Consider a naïve steganographic scheme in which we add the signal $\{m_i s_i\}$ to the image. Unfortunately, in order to recover the message, the original image or at least its approximation (e.g., using low-pass filtering) is necessary. Errors in estimating the original image necessitate employment of error-correction schemes, which in turn may dramatically decrease the steganographic capacity. Below, we show a simple idea how a class of parametrized parity functions can be used to make this scheme oblivious.

We define a parity function P on pixel values, $P(x, s) \in \{-1,1\}$, for $x \in \{0, \ldots, 255\}$ and $s > 0$, where s is an integer parameter, and $P(x, s) = 0$ for $s = 0$. This function applied to the stego image pixel values will produce message bits. The parity function is required to satisfy the following "anti-symmetric" property for all x

$$P(x+s, s) = -P(x-s, s) \text{ for } s \neq 0 . \quad (1)$$

For example, for $s = 1$, we can define $P(x, 1)$, $x = 0, 1, 2, \ldots$ as $P(x, 1) = 1, 1, -1, -1, 1, 1, -1, -1, \ldots$. In general, for $s > 0$, the first segment of $2s$ parities can be arbitrary, but every next segment of $2s$ values must be the negative copy of the previous segment. Thus, it is enough to define P on the set $[1, 2s]$. A good choice for the parity function is

$$P(x, s) = (-1)^{x+s}, x \in [1, 2s] .$$

This parity function ensures that P changes its sign as often as possible. We will find this property useful when $x+s$ or $x-s$ should get outside of their dynamic range during embedding.

Notice that besides the pixel value x, the parity function depends on the second parameter s. This is important because otherwise we could not find a function $P(x)$ satisfying $P(x+s) = -P(x-s)$ for all pixel values x and all positive integers s.

3.1 Embedding method

Having defined the parity function, we can now continue with the description of the embedding method. The image pixels can be visited either sequentially or along a pseudo-random walk generated from the stego-key. A pseudo-random number generator (PRNG) is seeded with a secret seed derived from the stego-key. The PRNG should produce numbers with a distribution that matches the distribution of the noise that will be superimposed on the cover image during embedding. We will call the noise generated by the PRNG the *stego noise*.

For each pixel x along the random walk, we generate one sample of the stego noise rounded to an integer s. If $s = 0$, we do not modify x and move to the next pixel. If $s \neq 0$, we check if $P(x+s, s) = m$, where m is the message bit to be embedded. In this case, we modify x to $x + s$ and move to the next pixel and embed the next message bit. If $P(x+s, s) = -m$, we modify x to $x - s$. Denoting the pixel values of the stego image as x_i', the embedding process can be expressed using the formula

$$x_i' = x_i + m_i - P(x_i + s_i , s_i)\, s_i .$$

In this formula, the message bits m_i are duplicated as necessary to account for the cases when $s_i = 0$. We can say that instead of adding the signal $\{m_i s_i\}$ to the cover image as we did in the beginning of this section, we add $\{v_i s_i\}$, where $v_i = m_i P(x_i + s_i, s_i)$. According to our assumption, the message bits m_i form a pseudo-random sequence of 1's and –1's. Because the image and the stego noise sequence s_i are independent of the message, the variable v_i is also a pseudo-random sequence of 1's and –1's. Thus, the signal v_i has the same statistical properties as the stego noise.

There is a slight complication at the boundaries of the pixels' dynamic range at 0 and 255. The amplitude of the noise that is added to the image should be truncated as it would happen during the image acquisition process. Whenever $x_i + s_i > 255$ the x_i' will be the nearest value less or equal to 255 with the desired parity m_i. A similar measure is applied when $x_i + s_i < 0$.

3.2 Message extraction

In the decoding process, we generate the same stego noise sequence $\{s_i\}$ from the stego-key as was done during message embedding, follow the same pseudo-random path in the stego image, and apply the parity function P to the pixel values. The non-zero parity values form the secret message

$$m_i = P(x_i, s_i).$$

We note that the stego-key can determine both the random embedding walk and the sequence s_i.

If the device noise is Gaussian, the embedding distortion can be expressed using the Peak-Signal-to-Noise-Ratio (PSNR) as

$$PSNR = 10\log_{10}\left(\frac{255^2}{\sigma^2}\right).$$

To obtain the expected capacity C measured as the number of bits-per-pixel, bpp, we have to subtract the probability of occurrence of '0' in the stego noise sequence s_i from the maximum bit rate of 1 bit per pixel (bpp):

$$C = 1 - erf\left(\frac{1}{2\sqrt{2}\sigma}\right), \tag{2}$$

where *erf* is the statistical error function $erf(x) = \frac{2}{\sqrt{\pi}}\int_0^x e^{-t^2} dt$.

On the other hand, if the user specifies the capacity C in bpp, the standard deviation σ of the noise that needs to be added to the image to carry such payload must be greater than

$$\sigma = \frac{\sqrt{2}}{4\, erf^{-1}(1 - C)},$$

where erf^{-1} is the inverse error function.

3.3 Improved stochastic modulation

The steganographic method as described in the previous subsection works with *one* stego noise sequence s_i that is either *added* or *subtracted* from the pixel value based on the match between the message bit and the parity function. It is possible to obtain a higher embedding capacity with the same distortion by considering *two* stego noise sequences rather than one and always *add* one or the other, again based on the match between the message bit and a parity function. Furthermore, the improved technique works for stego noise with an arbitrary probability distribution. In particular, the distribution now does not have to be symmetrical about zero.

First, we generate two independent stego noise sequences rounded to integers r_i, s_i (for example, we can seed the PRNG with two different seeds derived from the stego-key). For each pixel x_i, if $r_i - s_i = 0$, we do not embed a message bit but we do modify x_i to $x_i + r_i$ and embed the same message bit in the next pixel. If $r_i - s_i \neq 0$, we verify whether $P_2(x_i + s_i, r_i - s_i) = m_i$, where m_i is the message bit to be embedded, and P_2 is a parity function defined so that $P_2(x + k, k) = -P_2(x, k)$ for all $x \in [0, 255]$ and $k \neq 0$ (see the definition below). In this case, we modify x_i to $x_i + s_i$ and

move to the next pixel. If $P_2(x_i+s_i, r_i-s_i) \neq m_i$, we modify x_i to x_i+r_i. According to the definition of P_2, we now have to obtain a match because

$$P_2(x+r, r-s) = -P_2(x+r-(r-s), r-s) = -P_2(x+s, r-s) \neq 0, \text{ whenever } r-s \neq 0.$$

Denoting the pixel values of the stego image as x_i', the embedding process can be expressed using the formula

$$x_i' = \begin{cases} x_i + s_i, & \text{if } P_2(x_i + s_i, r_i - s_i) = m_i \\ x_i + r_i, & \text{if } P_2(x_i + r_i, r_i - s_i) = m_i \end{cases}$$

Thus, the pixel value is modified by adding the value of one of the stego noise sequences to it. Because the selection of the values r_i vs. s_i is governed by the match between two uncorrelated quantities – the parity function $P_2(x_i+s_i, r_i-s_i)$ and the message bit m_i – the noise added to the image has the same characteristics as the stego noise.

The parity function P_2: $([0, 255], \boldsymbol{Z}) \rightarrow \{-1,1\}$ is periodic with the period $2|k|$ and is defined similarly to the previous parity function P requiring

$$P_2(x+k, k) = -P_2(x, k), \text{ for all } x \in [0, 255] \text{ and } k \neq 0.$$

Again, it will become advantageous to define P_2 so that it frequently changes sign to minimize deviations from the stego noise signal at 255 and 0:

$$P_2(x, k) = (-1)^{x+k}, x \in [1, k], k \neq 0 \text{ is an integer,}$$
$$P_2(x, 0) = 0.$$

The embedding distortion for the new method is still the same as when just one stego noise sequence was used because we are adding a signal selected at random from two stego noise sequences of the same distribution. But now the probability of not embedding a message, which occurs when $r - s = 0$, is smaller. This is because in general the distribution of the sum of two random variables is the convolution of both and is thus "flatter" than for a single variable. In the case of a Gaussian stego noise, the probability that $r - s = 0$ can be calculated as follows. The random variables r and s are quantized Gaussian variables with the distribution $P_s=P_r$:

$$P_r(r=n) = \int_{n-1/2}^{n+1/2} \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-t^2}{2\sigma^2}} dt \, .$$

Thus the probability that $r - s = 0$ is $(P_r * P_s)(0)$, where $*$ denotes the convolution. Finally, the capacity of the modified stochastic modulation algorithm is

$$C = 1 - (P_r * P_s)(0). \tag{3}$$

The improvement in capacity is quite obvious from Figures 1 and 2 (the broken line corresponds to the embedding method with one Gaussian sequence and the continuous line to the new two-sequence method).

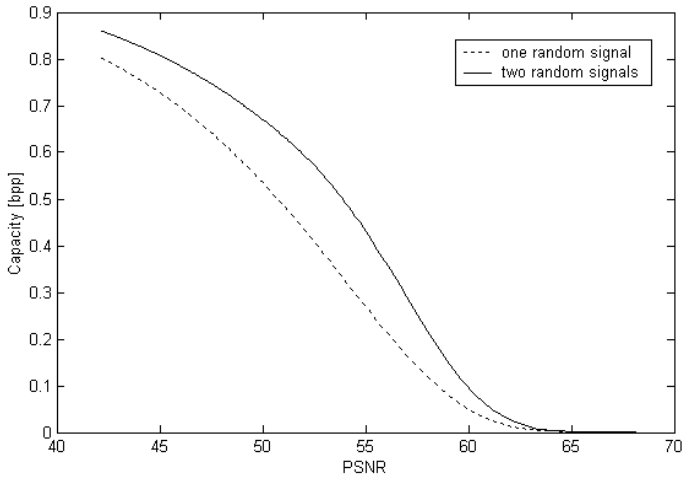


Figure 1: Capacity as a function of the PSNR.

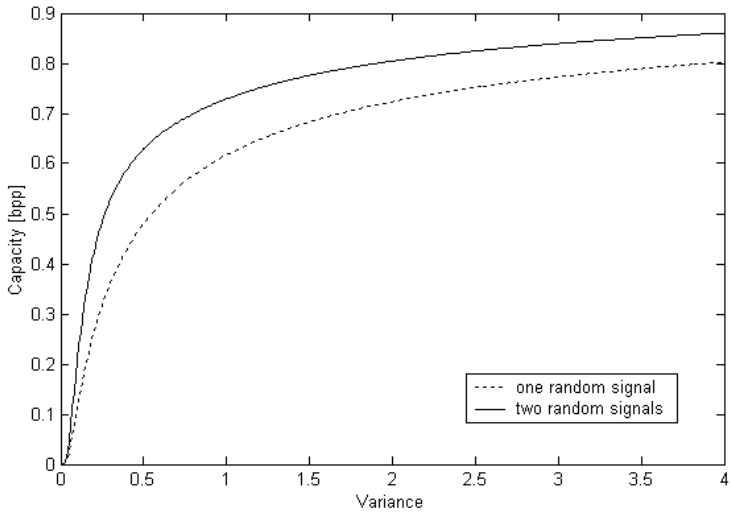


Figure 2: Capacity as a function of the Gaussian variance σ^2.

4. PRACTICAL IMPLEMENTATION ISSUES

In this section, we want to address two important implementation issues that must be considered in any practical communication scheme that uses the paradigm of stochastic modulation:

(1) Communication of the stego noise parameters (e.g., the variance σ for Gaussian noise) to the receiver,
(2) The stego noise sequences r_i and s_i should be independent for different images, i.e., it is not sufficient if they only depend on the stego-key.

Let us first inspect the issue of the stego noise parameters. Because stochastic modulation masks embedding by adding stego noise to the cover image, it is important to always add the noise signal to *all* image pixels rather than a proper subset of pixels. However, the embedding capacity is a function of the stego noise probability distribution. Thus, either we choose one fixed probability distribution (i.e., a fixed variance sigma for Gaussian noise) and pad the messages with dummy bits in order to cover the whole image or we adjust the noise parameters according to the message length. The second alternative does not introduce unnecessary distortion and is thus preferable. However, in this case we need to communicate the stego noise parameters. It is not clear how to achieve this because one needs the parameters in order to begin with the extraction. As one possible choice, one could choose a small set of pixels

and encode the stego noise parameters, for example, into their LSBs. Although this set of pixels can be generated from the stego-key, for cover images of the same size the pixels carrying the stego noise parameters will be the same unless one changes the stego-key. Why this arrangement can introduce a weakness and a possible place to attack the scheme, will become apparent in the paragraph below where we discuss the second implementation issue.

Let us assume that an attacker obtains N different stego images that are of the same size and were embedded with different messages of the same length with one fixed stego-key. Each pixel $x_i(t)$ in the t-th cover image can be modified during embedding either to $x_i(t)+r_i$ or to $x_i(t)+s_i$, where r_i and s_i are two stego-noise sequences generated from the secret stego-key. An attacker could apply denoising algorithms to each stego image and extract the noise signal $\eta(t)$. The extracted noise can be decomposed into three components $\eta(t) = \eta_{noise}(t)+\eta_{content}(t)+\eta_{stego\text{-}noise}(t)$, where η_{noise} is the component due to noise already present in the image, $\eta_{content}$ depends on image content, and $\eta_{stego\ noise}$ is either r_i or s_i. Because the first two components are independent from image to image, the expected value $E(\eta(t)) = r_iP(\eta = r_i)+s_iP(\eta = s_i) \neq 0$ in general. This observation could be used to collect enough statistical evidence to distinguish stego and cover images, given sufficiently large N. Also, for the same reason, we might identify the set of pixels that encode the parameters (see the previous paragraph) unless we change the stego-key sufficiently often. This attack is similar in principle to the idea proposed by Memon et al.[6] for attacking image-independent robust spread-spectrum watermarking schemes.

The second issue could be solved by changing the stego key sufficiently often, for example, by communicating it using a side channel or by sending future keys as part of the messages. However, sending the next key as part of the message brings up synchronization issues, should some of the images get corrupted or lost on their way to the recipient. Ideally, we would like to have a scheme that avoids any potentially complicated issues with key management and does not rely on any side channel. As a result of these requirements, we propose the following system for embedding and extraction that uses a session key and a clever communication of the stego noise parameters.

Emb 0: In the beginning, both communicating parties agree on the stego-key K. This key will be kept secret and will not be changed during the course of communication. Alice and Bob will also agree on how many bits they will be using to code the stego noise parameters (n_p bits) and on the length l_s of the session key S, $|S| = s$.

Emb 1: To embed m bits in the cover image, Alice first calculates the values of the stego noise parameters that will enable her to embed with high probability at least m bits in the cover image.

Emb 2: As the next step, she uses the stego-key K to divide the pixels of the cover image I into $n_p + l_s$ random subsets $I_1, I_2, \ldots, I_{n_p+l_s}$, $I_1 \cup I_2 \cup \ldots \cup I_{n_p+l_s} = I$. The algorithm for this part of the stego system is public and will be known to the opponent.

Emb 3: Then, she generates a random session key S. She uses both the stego key K and the session key S to generate the random stego noise sequences r_i and s_i with parameters determined in step Emb 1 and uses them as in Subsection 3.3 to embed the message (she can embed the bits sequentially in the cover image, for example).

Emb 4: After the message is embedded (she may need to pad the message to make sure that the whole image has been affected), Alice encodes the $n_p + l_s$ bits (the stego noise parameters and the session key) as parities of subsets I_k defined in step Emb 2. Alice can, for example, encode one bit per each I_k as the XOR the LSBs of all pixels in I_k. If she needs to modify this XOR, she can randomly choose one pixel y_k in I_k that was not used for embedding in step Emb 3 due to the fact that $r = s$ (see Subsection 3.3) and add or subtract 1 from it.

The extraction of the secret message proceeds as follows.

Ext 0: Bob uses his stego key K and divides the stego image into $n_p + l_s$ subsets I_k.

Ext 1: Then, he calculates the XOR of all LSBs in each I_k and recovers the stego noise parameters and the session key S.

Ext 2: Bob then uses K and S to generate the same stego noise sequences r_i and s_i as Alice.

Ext 2: Bob reads the header and then m message bits using the algorithm of Subsection 3.3.

First of all, notice that this communication system solves the problem of having to communicate the stego noise parameters before Bob can apply the extraction algorithm of stochastic modulation. The session key also brings enough diversity to the stego noise, so that the attack mentioned above is no longer plausible.

In practice, if we used Gaussian noise as the stego noise, we would need, say, 10 bits to represent the variance σ^2 and, say, 10 bits for the session key. This length of the session key may seem too short, but will likely be sufficient for any practical purposes. Even if the attacker had 1 million of stego images, there would be about 1000 images with the same stego noise but with different modulation (assuming all messages have the same length) that he would have to somehow find among the 1 million of images. Considering the fact that the stego noise variance is less than 1.38 (for less than 0.8 bpp messages), which is far below typical noise levels in images, this task does not seem plausible. In any case, paranoid users could reserve 20 or more bits for the session key to obtain their peace of mind.

Because the stego noise sequence depends on both the stego and session keys, a brute force search for the message presence will not be feasible. We reiterate that the purpose of the stego key is just to create sufficient variety among the noise sequences to prevent the attack mentioned above.

The only deviation from the desired statistics of the stego noise occurs in step Emb 4 when we modify on average $(n_p + l_s)/2$ pixels by 1 (10 pixels if we use the parameters from the previous paragraph). However, because now those pixels are selected at random and are different for each image (due to the session key), the attacker cannot identify them by averaging the stego images.

In step Emb 4, it could theoretically happen that all pixels in one set I_k have been modified in Step 3 and Alice would not be able to embed any bit in that group. For a small 128×128 image and $n_p + l_s = 20$, on average $|I_k| = 128^2/20 = 819$. The number of zeros in the stego sequence decreases with message length. In the least favorable case for us, when the message is large, such as 0.8 bpp, the probability that all pixels in I_k will be used for embedding is $0.8^{819} \cong 10^{-80}$, which is already astronomically small. In either case, to make the embedding 100% reliable, the embedder could go back to Emb 3 and generate a new session key *S*. Embedding rates higher than 0.8 bpp are not recommended for this steganographic method due to a sharp increase in distortion above this rate and a negligible gain in capacity (see Figure 2).

5. SECURITY CONSIDERATIONS

In this section, we try to answer the following question: "How detectable is the act of adding stego noise to the cover image?" Westfeld[11] recently described an attack on Hide[8]. The +–1 embedding mechanism of Hide creates a large (and anomalous) number of close colors (i.e., colors that differ by at most 1 in each channel). There are between 0 and 26 close neighbors for each color in the RGB cube. The attack starts by calculating for each color present in the image the number of its closest neighbors. Then, we calculate the histogram h_i, $i = 0, 1, \ldots, 26$, h_i standing for the number of colors with exactly i closest neighbors. Non-embedded cover images typically have $h_i = 0$ for $i > 10$, while images embedded using Hide show a tail of gradually falling non-zero values h_i for $i > 10$ even for relatively small messages. This difference could be used to identify stego images created by Hide and the method would also work for detection of stochastic modulation because most of the changes in stochastic modulation are also –1, 0, and 1. However, this attack appears to work for a relatively narrow class of images that include cover images that are decompressed JPEGs or, in general, images that do not have a large number of unique colors, such as JPEGs due to the low-pass character of JPEG compression, or filtered images with a high SNR. However, there is a large class of naturally occurring images that exhibit a non-zero tail in h_i reminiscent of the one caused by embedding. This class includes scans of photographs or decompressed JPEG images resampled to a smaller size. The scanning process or the resampling algorithm often introduce many new close colors in the image and thus appear as stego images produced by Hide or the stochastic modulation algorithm. Such images would be incorrectly identified as false positives.

Westfeld[11] mentions in his paper that his technique could in principle be extended to grayscale images after transforming them to color images by constructing colors from three neighboring pixels. However, the attack is not further explored and is quite likely to encounter similar problems for scans and resampled images.

Another related, but different attack has been proposed by Harmsen[4]. Harmsen based his attack on the fact that noise adding in the spatial domain corresponds to low-pass filtering of the histogram. Thus, the histogram of stego images has less power in high frequencies than the same histogram for cover images. The attack starts with calculating the 3D histogram of color images $h(r, g, b)$ standing for the number of pixels with the color (r, g, b). Then, h is transformed using the 3D Fourier transform to obtain $\mathcal{F}(h)$. Finally, he calculates the center of gravity (which is a 3D vector for color images and a scalar for grayscale images) of $|\mathcal{F}(h)|$ and uses this quantity as the distinguishing statistics to differentiate between cover and stego images. As reported by Harmsen, it is indeed possible to find a

fixed threshold (!) that will distinguish between cover images and embedded stego images for the Greenspun database (www.greenspun.com). The attack also works for images with a low number of colors, such as decompressed JPEGs or in general images with high SNR.

Our experiments indicate that, similar to the method by Westfeld, images with a high number of unique colors, such as scans, cannot be thresholded using this method. Equivalently, such images will introduce many false positives. One possibility to fix this problem would be to use an image adaptive threshold, but at this point, it is not clear how to choose such a threshold. Also, in our tests, Harmsen's method could not distinguish between cover and stego grayscale images produced by stochastic modulation.

To summarize this section, we point out that we are currently unaware of any reliable method for detection of stochastic modulation. Partial success has been reported for detection of noise adding in color images, but the detection appears to be unreliable for scans and resampled images. Stochastic modulation for grayscale images appears as a relatively safe method and is also recommended as the method of choice by the authors.

6. CONTENT-DEPENDENT STOCHASTIC MODULATION

Some digital image acquisition devices may generate noise that depends on the image content. For example, the raw image data obtained from an image sensor is usually post-processed using filters that may introduce a noise component that is spatially non-uniform. The question that naturally arises is whether it is possible to develop stochastic modulation steganography for some sufficiently general content-dependent device noise model. Indeed, this topic is the subject of this section.

In what follows, we will assume the following model for a content-dependent device noise:

$$x' = x + \eta f_a(H(x)) ,$$

where x is the original pixel value without noise, x' is the noisy pixel value, $H(x)$ is a small neighborhood of x (such as 3×3 or 5×5 neighborhood), η is a random variable, and $|f_a(H(x))| \le 1$ is a scaling factor capturing the dependence of the noise amplitude on the image content. The scaling factor may depend on a parameter a that indicates the "degree of content-dependency". Note that $f_a(H(x_i)) \equiv 1$ for a device noise that is independent of the image content. Before we proceed further, we would like to stress that the device noise is a quite different concept from the concept of just-noticeable noise that is related to perception rather than the acquisition device.

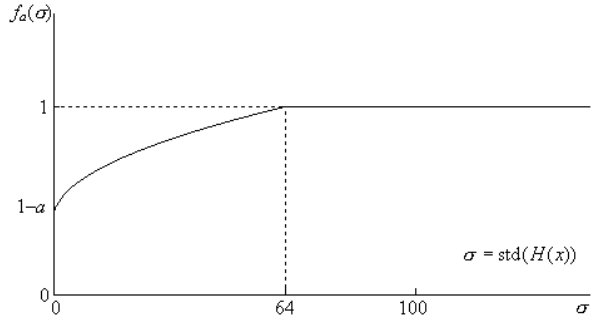


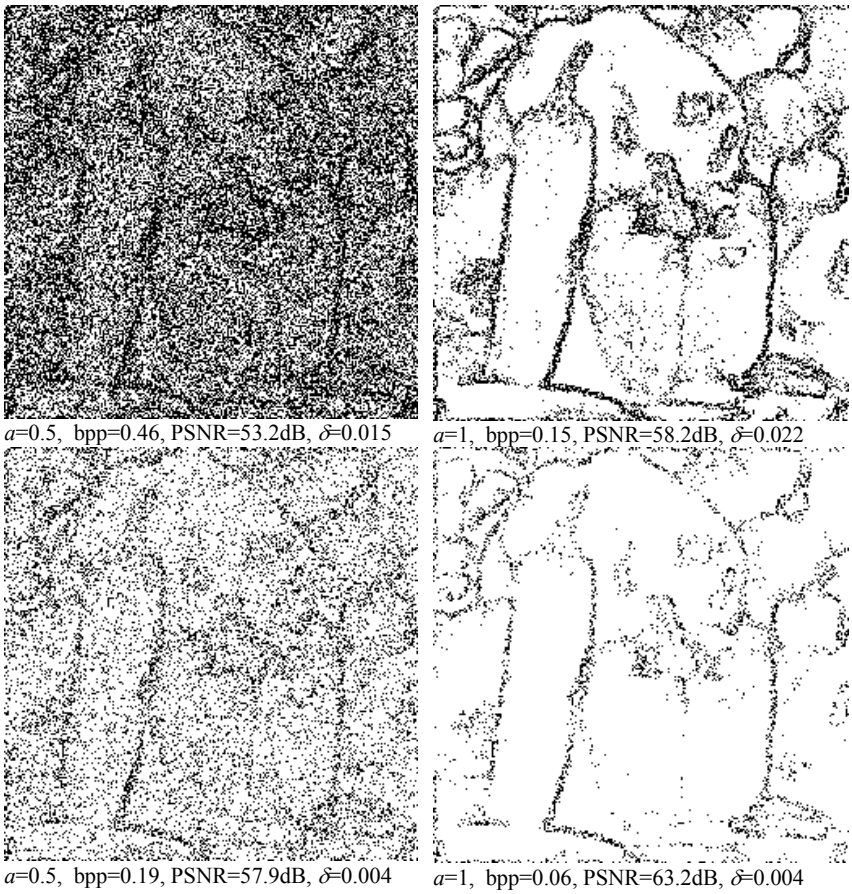
Figure 3: Scaling function $f_a(x)$ as a function of the standard deviation σ of the neighborhood $H(x)$.

To follow the logic of stochastic modulation, the stego noise should be modulated as

$$r_i' = [R_i f_a(H(x_i)] , \; s_i' = [S_i f_a(H(x_i)], \tag{3}$$

where $[x]$ is the rounding operation and R_i, S_i are two independent random sequences with the same distribution as η. Because the noise is now modulated by the cover image values, that will not be available to the recipient, during embedding we have to calculate the content-dependent component only from those pixels that will not be modified by the steganographic process. Thus, the message embedding proceeds as follows:

1. Follow either a sequential, $i = 1, \ldots, M \times N$, or a pseudo-random path, $i = P(1), \ldots, P(M \times N)$, through the image, where P is a pseudo-random permutation of the set $\{1, \ldots, M \times N\}$
2. Initialize $r_i' = [R_i]$, $s_i' = [S_i]$ for all i
3. If X is the set of all pixels in the image, we identify the set Z of pixels that is guaranteed to not be modified independently of the message: $Z = \{x_i \in X \mid r_i' = s_i' = 0\} - \{y_k \mid k = 1, \ldots, n_p + l_s\}$. We will use the pixels from the set Z for calculating the actual scaling factor f_a during message embedding. Let us further denote $H_Z(x) = H(x) \cap Z$ for all pixels x. Let us note that the location of pixels y_k must be known to the decoder and thus cannot be completely random as originally proposed in Section 4, but they can depend on the session key S and the stego key K.
4. for $i = 1, \ldots, M \times N$
   if $x_i \in X - Z$ then
   $r_i' = [R_i f_a(H_Z(x_i))]$
   $s_i' = [S_i f_a(H_Z(x_i))]$
   if $r_i' = s_i' = 0$ then update the set Z, $Z = Z \cup \{x_i\}$
   end (if)
   end (for)
5. Use r_i' and s_i' as in the content-independent version of stochastic modulation from Subsection 3.3.



a=0.5, bpp=0.46, PSNR=53.2dB, δ=0.015 a=1, bpp=0.15, PSNR=58.2dB, δ=0.022

a=0.5, bpp=0.19, PSNR=57.9dB, δ=0.004 a=1, bpp=0.06, PSNR=63.2dB, δ=0.004

Figure 4: Examples of modified pixels for the test image "Peppers" and two different device noise types (a = 0.5 and a = 1). The message size is given in bits per pixel (bpp) and the embedding distortion is measured using the PSNR in dB. The quantity δ measures the discrepancy between the stego-noise and the actual noise added to the cover-image (see the explanation in the text). Black dots correspond to modified pixels.

The message extraction proceeds in exactly the same way. First we generate the sequences R_i and S_i, and the rounded sequences $r_i' = [R_i]$ and $s_i' = [S_i]$. Then, we follow the Steps 3 and 4 from the previous paragraph. The final

sequences r_i' and s_i' obtained from the stego image will be the same as during embedding because they are modulated by those pixels in the image that do not change by message embedding.

To test the plausibility of the approach and test its performance, we briefly present some results we obtained with the following device noise model (see Figure 3):

$$f_a(\sigma) = \begin{cases} 1 - a + a\sqrt{\sigma}/8 & \sigma < 64 \\ 1 & \sigma \geq 64 \end{cases}, \quad \text{where } a \in [0,1],\ x \in [0,255],\ \sigma = \text{std}(H(x)) .$$

This model corresponds to an i.i.d. Gaussian noise of amplitude $1-a$ superimposed with another Gaussian component proportional to a that is modulated by the image content.

Figure 4 shows the pixels that are modified in the test image "Peppers". The results are for two different device noise types corresponding to a = 0.5 and a = 1 and two message sizes expressed in bpp. The noise is clearly concentrated in busy areas containing edges while the smooth areas contain noise of a smaller amplitude.

The only issue that needs to be clarified is how much the stego noise of Equation (3) calculated from the neighborhood H is different from the stego noise calculated only from the subset H_Z (see Step 4 above). In fact, if the discrepancy was too big, it would be a weakness that could possibly be exploited for steganalysis. Fortunately, our experiments indicate that at least for the device noise model from Figure 3, the number of stego noise samples δ that differ when calculated from H and H_Z is less than 2.5%.

In this section, we have shown that it is possible to use the concept of stochastic modulation for secure steganography even when the device noise contains a content-dependent component. This modification makes the new steganographic principle more flexible and increases its applicability in practice.

7. CONCLUSIONS AND FUTURE DIRECTIONS

In this paper, we describe a steganographic technique that embeds high payloads (up to 0.8 bpp for grayscale images) by adding a small-amplitude noise of specified properties to the image pixels. Because the probability distribution of the noise can be arbitrary, the communicating parties have the flexibility to mask the embedding distortion as superposition of a particular device noise. Thus, this embedding paradigm will provide better security than embedding that uses somewhat arbitrary operations, such as flipping the LSBs or adding a fixed-amplitude noise to the image (+–1 embedding).

Compared to previously proposed methods that embed messages by adding Gaussian noise[1,7], stochastic modulation can embed bigger payloads without using error correction schemes or denoising algorithms at the receiving end, which makes the algorithm significantly simpler and faster. Because the noise distribution can be arbitrarily adjusted, the new method provides much greater flexibility than previous methods.

This paper also pays close attention to practical implementation issues, such as the issue of having to communicate the parameters of the added stego noise and the necessity to use different noise signals for each image due to security. In Section 6, we show that the concept of stochastic modulation can be extended to stego noise that depends on the image content.

The security of data hiding by adding noise to the cover image has been recently addressed[4,11]. These attacks, however, do not work well for grayscale images and may produce a significant rate of false positives for color scans of natural photographs and resampled images (see Section 5).

Future directions include steganalysis of the proposed steganographic paradigm. In particular, it appears that distinguishing stego images embedded using stochastic modulation from color scans or resampled images is of paramount importance. Analyzing the difference between these image classes appears to be the key part in building reliable detection algorithms.

ACKNOWLEDGEMENT

The work on this paper was supported by Air Force Research Laboratory, Air Force Material Command, USAF, under a research grant number F30602-02-2-0093. 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 Air Force Research Laboratory, or the U. S. Government. Special thanks belong to David Soukal for useful discussions during the preparation of this manuscript.

REFERENCES

1. F. Alturki and R. Mersereau, A Novel Approach for Increasing Security and Data Embedding Capacity in Images for Data Hiding Applications. *Proc. of ITCC*, Las Vegas, Nevada, 2001, pp. 228–233.
2. S. Dumitrescu, Wu Xiaolin, and Z. Wang, "Detection of LSB Steganography via Sample Pair Analysis. *Preproceedings 5th Information Hiding Workshop*, Noordwijkerhout, Netherlands, Oct. 7–9, 2002.
3. J. Fridrich, M. Goljan, and R. Du, "Detecting LSB Steganography in Color and Gray-Scale Images", *Magazine of IEEE Multimedia*, *Special Issue on Security*, October-November issue, 2001, pp. 22–28.
4. J.J. Harmsen and W. A. Pearlman, "Steganalysis of Additive Noise Modelable Information Hiding", *Proc. SPIE Electronic Imaging*, Santa Clara, January 21–24, 2003.
5. G.E. Healey and R. Kondepudy, Radiometric CCD Camera Calibration and Noise Estimation. IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. **16**(3), March 1994, pp. 267–276.
6. M. Holliman, N. Memon, and M. M. Yeung, "On the Need for Image Dependent Keys for Watermarking", *Proc. Content Security and Data Hiding in Digital Media*, Newark, NJ, May 14, 1999.
7. L.M. Marvel, C.G. Boncelet, and C.T. Retter, Reliable Blind Information Hiding for Images. In: D. Aucsmith (eds.): *Information Hiding: 2nd International Workshop*, LNCS, Vol. 1525. Springer-Verlag, New York, 1998, pp. 48–61.
8. T. Sharp, "An Implementation of Key-Based Digital Signal Steganography", In: I. S. Moskowitz (eds.): *4th International Workshop on Information Hiding*, LNCS 2137, Springer-Verlag, New York, 2001, pp. 13–26.
9. A. Westfeld and A. Pfitzmann, Attacks on Steganographic Systems. In: A. Pfitzmann (eds.): *3rd International Workshop on Information Hiding*. LNCS, Vol.1768. Springer-Verlag, New York, 2000, pp. 61–75.
10. A. Westfeld, High Capacity Despite Better Steganalysis (F5–A Steganographic Algorithm). In: Moskowitz, I.S. (eds.): *4th International Workshop on Information Hiding*, LNCS, Vol. 2137. Springer-Verlag, New York, 2001, pp. 289–302.
11. A. Westfeld, Detecting Low Embedding rates. In: Petitcolas et al. (eds.): *Preproceedings 5th Information Hiding Workshop*. Noordwijkerhout, Netherlands, Oct. 7–9, 2002.