

New methodology for breaking steganographic techniques for JPEGs

Jessica Fridrich^{*a}, Miroslav Goljan^a, Dorin Hoge^b

^aDept. of Electrical and Computer Engineering, SUNY Binghamton, Binghamton, NY 13902-6000

^bDept. of Computer Science, SUNY Binghamton, Binghamton, NY 13902-6000

ABSTRACT

In this paper, we present general methodology for developing attacks on steganographic systems for the JPEG image format. The detection first starts by decompressing the JPEG stego image, geometrically distorting it (e.g., by cropping), and recompressing. Because the geometrical distortion breaks the quantized structure of DCT coefficients during recompression, the distorted/recompressed image will have many macroscopic statistics approximately equal to those of the cover image. We choose such macroscopic statistic S that also predictably changes with the embedded message length. By doing so, we estimate the unknown message length by comparing the values of S for the stego image and the cropped/recompressed stego image. The details of this detection methodology are explained on the F5 algorithm and OutGuess. The accuracy of the message length estimate is demonstrated on test images for both algorithms. Finally, we identify two limitations of the proposed approach and show how they can be overcome to obtain accurate detection in every case. The paper is closed with outlining a condition that must be satisfied by all secure high-capacity steganographic algorithms for JPEGs.

Keywords: Steganalysis, steganography, attacks, JPEG

1. STEGANOGRAPHY FOR JPEG IMAGES

The JPEG format is currently the most common format for storing image data. It is also supported by virtually all software applications that allow viewing and working with digital images. Recently, several steganographic techniques for data hiding in JPEGs have been developed: J-Steg¹², JP Hide&Seek¹², F5¹⁴, and OutGuess¹⁰. In all programs, message bits are embedded by manipulating the quantized DCT coefficients. J-Steg and OutGuess embed message bits into the LSBs of quantized DCT coefficients, while F5 always decrements the values in order to modify the LSBs.

J-Steg with sequential message embedding is detectable using the chi-square attack¹⁵. J-Steg with random straddling as well as JP Hide&Seek are detectable using the generalized chi-square attack^{11,13}. The chi-square attacks are not effective for F5 (because F5 does not flip LSBs) and for OutGuess (OutGuess preserves first-order statistics). The universal blind detector pioneered by Farid⁶ seems to be able to detect most steganographic methods after appropriate training on a database of stego and cover images. However, at the time of writing this paper, the blind detector does not naturally allow accurate estimation of the length of the embedded messages and it is not clear how its performance scales to more diverse databases. Also, this detector cannot detect messages embedded using F5 in grayscale images. The authors of this paper also believe that detection methods targeted to a specific steganographic technique will necessarily give better accuracy and detection reliability than blind approaches. Another important advantage of the approach proposed in this paper compared to previously introduced detection schemes is that one can obtain an accurate estimate for the length of the embedded secret message.

In this paper, we will position ourselves into the role of a passive warden who inspects digital images and tries to identify those that contain secret messages. In particular, our goal is to estimate the number of embedding modifications (and thus the secret message length). We start the next section by introducing the concept of steganographic capacity (maximal number of bits that are “safe” to embed) in an informal manner. Then, in Section 3 we outline our strategy for design of steganalytic techniques capable of estimating the secret message length. In Section 4, we describe the steganalytic algorithm for F5 and in Section 5 the detection algorithm for OutGuess. The paper is concluded in Section 6 by outlining how the proposed methods can be used for estimating steganographic capacity. In the same section, we also state a necessary condition for any high-capacity JPEG steganographic method to be secure against attacks of the kind described in this paper.

2. STEGANOGRAPHIC CAPACITY

Each steganographic communication system consists of an embedding algorithm and an extraction algorithm. To accommodate a secret message, the original image (cover image) is slightly modified by the embedding algorithm. As a result, the stego image is obtained. The steganographic method will be called secure if the stego images do not contain any detectable artifacts due to message embedding. In other words, the set of stego images should have the same statistical properties as the set of cover images. If there exists an algorithm that can guess whether or not a given image contains a secret message with a success rate better than random guessing, the steganographic system is considered broken. Several definitions of steganographic security were proposed in the literature^{1,2,9}.

Obviously, the less information we embed into the cover image, the smaller the probability of introducing detectable artifacts by the embedding process. Each steganographic method seems to have an upper bound on the maximal safe message length (or the bit-rate expressed in bits per pixel or sample) that tells us how many pseudo-random bits can be safely embedded in a given image without introducing any statistically detectable artifacts. This limit is called the steganographic capacity. The absolute steganographic capacity C_A is a function of the cover image I and the embedding method Σ . $C_A(I, \Sigma)$ is the expected value of the maximal number of bits that can be safely embedded in the cover image I using the method Σ , the expected value being taken over all stego-keys K uniformly distributed in the key space and all pseudo-random messages. By safely, we understand that no detection algorithm can perform better for distinguishing cover and stego images than random guessing,

While for some special cases it is possible to establish C_A exactly, in general, its determination is a very difficult and still unsolved problem even for the simplest schemes. One can *partially* remove the dependency of C_A on the cover image by defining the relative steganographic capacity, C_R , which is the ratio between the steganographic capacity and the largest message, with length m_{max} , that can be embedded in the cover image

$$C_R = C_A / m_{max} .$$

An example of techniques for which $C_A = 0$, are embedding methods for palette images (GIFs) that preprocess the image palette by reducing the palette to 128 colors or less and then expanding the palette by adding to each color other colors that are close to it. The resulting palette will have a quite unusual structure (clusters of close colors) that is practically never created using color quantization and dithering algorithms. Thus, no matter what message is later embedded in the image, just analyzing the palette itself can reveal the fact that the stego image has been manipulated. As another, less trivial example, we cite the JPEG compatibility analysis⁸. It can be shown that for virtually all steganographic techniques that embed bits in the spatial domain, $C_A = 0$ for covers that are JPEG images decompressed to the spatial domain.

Surprisingly little theoretical work has been published regarding estimates of steganographic capacity. A notable exception is the work of Chandramouli et al.^{3,4} who gave a theoretical analysis of the steganographic capacity for a binary-valued embedding distortion in the spatial domain.

The current paper helps establish the upper bound for steganographic capacity for images in the JPEG format.

3. GENERAL DETECTION METHODOLOGY

In this section, we describe a general methodology that will be applied in the following sections for development of steganalytic techniques that can estimate the length of the secret message. We can only provide general guidelines and principles in this paper because it appears that no simple straightforward recipe can be formulated for all possible steganographic techniques. Consequently, each steganographic embedding archetype needs to be treated individually. Having said this, the authors believe that this text contains enough constructive examples and tools to assist researchers at the creative process of designing steganalytic techniques that can accurately estimate the secret message length.

For most steganographic techniques, it is usually not too difficult to identify a macroscopic quantity $S(m)$ that predictably changes (for example, monotonically increases) with the length of the embedded secret message m . Let us assume that the functional form of S is known or can be guessed from experiments. For example, S may be linear, quadratic, exponential, etc. In general, the function S will depend on several undetermined parameters. We can attempt to determine those parameters by estimating some extreme values of S , such as $S(0)$ (S for the cover image) or $S(m_{max})$ (for the stego image with maximal message). Once the parameters have been determined, one can calculate an estimate of the unknown message length m by solving the equation $S(m) = S_{stego}$ for m , where S_{stego} is the value of S for the stego image under investigation. We call $S(m)$ the *distinguishing statistics*. In this paper, we show what distinguishing statistics are useful for detection of two steganographic archetypes for JPEG images.

For JPEG images, it is actually possible to construct from the stego image a new JPEG image that will have many macroscopic properties very close to the cover JPEG image. This is because the JPEG file is formed by quantized DCT coefficients, which are “robust” to small distortion, such as the one due to message embedding and previous JPEG compression. By cropping the (decompressed) stego image by 4 pixels and recompressing it using the quantization table of the stego image, we obtain a JPEG file with macroscopic properties that well approximate the properties of the cover image. Because of the cropping, the newly calculated DCT coefficients will not exhibit clusters due to quantization. Also, because the cropped stego image is visually similar to the cover image, many macroscopic characteristics will be approximately preserved. For detection of F5, we use the histograms of individual DCT coefficients as the distinguishing statistics, while for OutGuess we use the increase in the sum of spatial discontinuities along 8×8 boundaries.

4. F5 ALGORITHM

The F5 steganographic algorithm was introduced by Westfeld¹⁴ in 2001. F5 embeds message bits as the LSBs of coefficients along a key-dependent random walk through all DCT coefficients of the cover image while skipping the DC coefficients and all coefficients that are zeros. If the coefficient’s LSB does not match the message bit, the absolute value of the coefficient is always *decremented*. If the subtraction leads to a zero coefficient (we say that the so-called *shrinkage* occurred), the same message bit must be embedded at the next coefficient, because at the receiving end, the message is extracted only from non-zero coefficients. As a special feature, the F5 algorithm employs matrix embedding to minimize the necessary number of changes to embed a message of certain length. Detailed description of F5 can be found in the original paper¹⁴.

Because the embedding is not based on bit-replacement or exchanging any fixed Pairs of Values, the F5 algorithm cannot be detected using the chi-square attack¹⁵ or its generalized versions^{11,13}. On the other hand, the F5 algorithm does modify a macroscopic quantity of the JPEG file – the histogram of DCT coefficients – in a predictable manner. Thus, we use it as the distinguishing quantity S . The number of zeros in the histogram increases due to shrinkage, while the histogram values for other coefficients decrease with embedding. In the next section, we give mathematical formulas that express the values of the stego image histogram as a function of the total number of embedding modifications and the histogram of the cover image. In order to calculate the number of modifications, we need to estimate the cover image histogram. This is achieved using a trick that is very effective for design of detection methods for most current steganographic methods for JPEGs. We crop the stego image by 4 columns and recompress the cropped image using the same quantization table as that of the stego image. The spatial shift by 4 pixels breaks the quantized structure of block DCT coefficients and, as a result, a good approximation of the cover JPEG file results. Finally, the number of changes is obtained by minimizing the L_2 norm between the stego image histogram and the histogram obtained from the estimated cover image histogram after certain number of changes.

4.1 Distinguishing statistics

Let $h(d)$, $d = 0, 1, \dots$ be the total number of AC DCT coefficients in the cover image with absolute value equal to d before the actual embedding starts. In a similar manner, we denote $h_{kl}(d)$ the total number of AC coefficients corresponding to the frequency (k, l) , $1 \leq k, l \leq 8$, whose absolute value is equal to d . The corresponding histogram values for the stego image will be denoted using the capital letters H and H_{kl} .

Let us suppose that the F5 embedding process changes n AC coefficients. The probability that a non-zero AC coefficient will be modified is $\beta = n/P$, where P is the total number of non-zero AC coefficients ($P = h(1) + h(2) + \dots$). Because the selection of the coefficients is random in F5, the expected values of the histograms H_{kl} of the stego image are

$$\begin{aligned} H_{kl}(d) &= (1 - \beta)h_{kl}(d) + \beta h_{kl}(d + 1), & \text{for } d > 0, \\ H_{kl}(0) &= h_{kl}(0) + \beta h_{kl}(1), & \text{for } d = 0. \end{aligned} \quad (1)$$

Equation (1) expresses the distinguishing statistics as a function of the number of modifications and the original image histogram. If we had an estimate $\hat{h}_{kl}(d)$ of the cover image histogram (the unknown parameters of the distinguishing statistics), we could use this estimate to calculate the expected values $H_{kl}(d)$ using Equation (1) and estimate β as the value that gives us the best agreement with the cover image histogram. We have experimented with different formulas for β and the best performance was obtained using the least square approximation. Because the first two values in the histogram ($d=0$ and $d=1$) experience the largest change during embedding (see Figure 1), we calculate β as the value that minimizes the square error between the stego image histogram H_{kl} , and the expected values $\hat{H}_{kl}(d)$ calculated from the estimated histogram \hat{h}_{kl} using Equation (1):

$$\beta_{kl} = \arg \min_{\beta} [H_{kl}(0) - \hat{h}_{kl}(0) - \beta \hat{h}_{kl}(1)]^2 + [H_{kl}(1) - (1 - \beta)\hat{h}_{kl}(1) - \beta \hat{h}_{kl}(2)]^2. \quad (2)$$

The least square approximation in Equation (2) leads to the following formula for β

$$\beta_{kl} = \frac{\hat{h}_{kl}(1)[H_{kl}(0) - \hat{h}_{kl}(0)] + [H_{kl}(1) - \hat{h}_{kl}(1)][\hat{h}_{kl}(2) - \hat{h}_{kl}(1)]}{\hat{h}_{kl}^2(1) + [\hat{h}_{kl}(2) - \hat{h}_{kl}(1)]^2}. \quad (3)$$

The final value of the parameter β is calculated as an average over selected low-frequency DCT coefficients ($k, l \in \{(1,2),(2,1),(2,2)\}$). We decided to not include the higher frequency coefficients due to problems with potential insufficient statistics especially for small images.

The reasons why we opted to work with histograms of individual low-frequency DCT coefficients rather than the global histogram will become apparent in the next section after we introduce the method for obtaining an approximation to the cover image histogram.

4.2 Parameters estimation

Accurate estimation of the baseline value (the cover image histogram h) is absolutely crucial for our detection method to work. The accuracy of this estimate is the main factor that determines the accuracy for the estimate of the relative number of modifications β (and the estimated message length). We first decompress the stego image to the spatial domain, then crop the image by 4 columns, and recompress the cropped image using the same quantization matrix as that of the stego image. The resulting DCT coefficients provide the estimates $\hat{h}_{kl}(d)$ for our analysis. We note that a similar trick will be used for attacking OutGuess in Section 5.

According to our experiments, the estimated histogram is quite close to the histogram of the cover image. We give a simple heuristic explanation of why the method for obtaining the baseline histogram values is indeed plausible. In fact, unless the quality factor of the JPEG compression is too low (e.g., lower than 60), the stego image produced by F5 is still very close to the cover image both visually and using measures, such as the PSNR. The spatial shift by 4 pixels effectively breaks the structure of quantized DCT coefficients. Thus, it is not surprising that macroscopic properties, such as the statistics of DCT coefficients, are similar to those of the cover image.

In Figure 1, we show a typical example of how good the histogram estimate is when compared to the histogram of the cover image. The graph shows the cover image histogram values $h_{21}(d)$ (crosses), histogram values after applying the F5 algorithm with maximal possible message, or $\beta = 0.5$ (stars), and the estimate of the cover image histogram (circles).

The main reason why we decided to use histograms of individual low-frequency DCT coefficients rather than the global image histogram is as follows. Even with the low-pass pre-filtering, the spatial shift by 4 pixels introduces some non-zero coefficients in high frequencies due to the discontinuities at block boundaries. And the values that are most influenced are 0, 1, and -1 , which are the most influential in our calculations. Individual histograms of low frequency coefficients are much less susceptible to this onset of spurious non-zero DCTs.

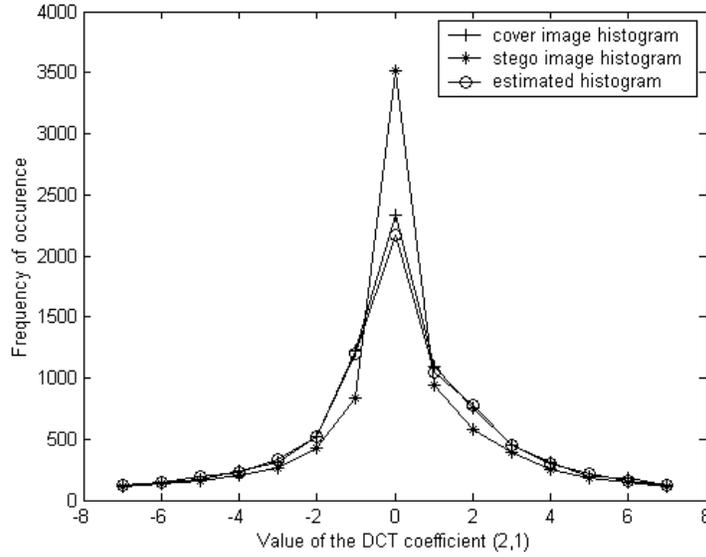


Figure 1: The effect of F5 embedding on the histogram of the DCT coefficient (2,1).

4.3 Estimating the true message length

Once the relative number of changes β has been estimated, we may attempt to further estimate the total message length. Both the shrinkage and matrix embedding must be taken into account. Let n be the total number of changes in quantized DCT coefficients introduced by the F5 algorithm. We can write n as $n = s + m$, where s is the shrinkage (modifications that did not lead to message bits embedding), and m is the number of changes due to actual message bit embedding. The probability of selecting a coefficient that may lead to shrinkage is $P_S = h(1)/P$. Since the coefficients are selected at random, the expected value of s is nP_S . Thus, we obtain the following formula:

$$m + nP_S = n,$$

which gives $m = n(1 - P_S)$ for the number of changes due to message embedding. Assuming the $(1, 2^k - 1, k)$ matrix embedding¹⁴, the expected number of bits per change $W(k)$ is

$$W(k) = \frac{2^k}{2^k - 1} k.$$

Thus, the unknown message length M can be calculated as

$$M = W(k)m = \frac{2^k}{2^k - 1} kn(1 - P_S) = \frac{2^k}{2^k - 1} k\beta P(1 - h(1)/P) = \frac{2^k}{2^k - 1} k\beta(P - h(1)),$$

where

$$P = \sum_{i \geq 0} h(i) \approx \sum_{i \geq 0} \sum_{\substack{k, l=1 \\ k+l > 2}}^8 \hat{h}_{kl}(i).$$

The parameter k can be derived from the knowledge of $n = \beta P$ and m and the estimated cover image histogram by following the algorithm of determining the optimal matrix embedding as implemented in F5.

4.4 Correcting for double compression for F5

When the cover image is stored in the JPEG format, both the F5 and OutGuess decompress it first and then recompress with a user-specified quality factor. After that, the message is embedded in the quantized DCT coefficients. This means that the stego image has been double compressed before embedding. The double compression can have a profound effect on the distinguishing statistics S and it complicates the detection.

The process of obtaining the baseline value $S(0)$ from the cropped image as described in Section 4.2 will produce a histogram similar to the broken line in Figure 2 instead of the solid line from which the F5 started its embedding. Consequently, the estimated relative number of changes β may be quite different from the actual value. To address the problems with inaccurate detection when the cover image is stored in the JPEG format, we proposed the following modification of our detection.

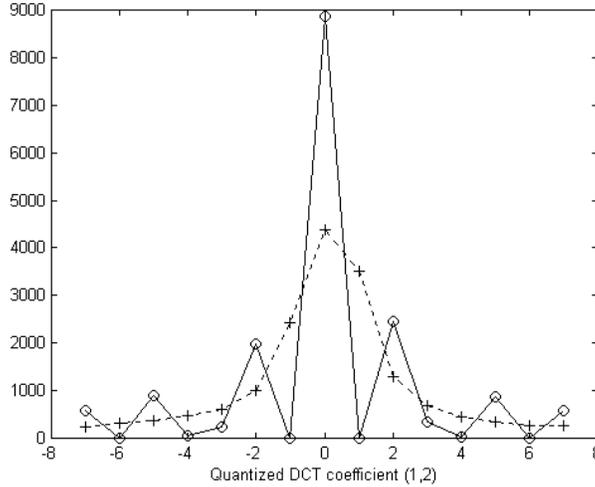


Figure 2: Effect of double compression on the histogram of quantized DCT coefficients. The broken line is the image histogram with a single compression, the solid line after double compression with a lower quality factor being the first one. The histogram corresponds to the DCT coefficient (1,2).

We calculate the ratio β for a fixed set of quantization tables, $\{Q_1, Q_2, \dots, Q_r\}$. For each quantization table, we run our detection scheme with one small modification – after cropping the decompressed stego image, it is compressed with the quantization table Q_i and immediately decompressed before proceeding with the rest of the baseline histogram estimation. Then, the estimated ratio β_i , $i = 1, \dots, r$ is calculated in the usual manner. For each i and for each DCT mode kl , we calculate the L_2 distance $E_{kl}^{(i)}$ between the stego image histogram H_{kl} and the histogram obtained using Equation (1) with $\beta = \beta_i$:

$$E_{kl}^{(i)} = [H_{kl}(0) - \hat{h}_{kl}(0) - \beta_i \hat{h}_{kl}(1)]^2 + \sum_{j=0}^J [H_{kl}(j) - (1 - \beta_i) \hat{h}_{kl}(j) - \beta_i \hat{h}_{kl}(j+1)]^2,$$

where in our experiments, we took $J = 10$ histogram values. The final estimated ratio β is obtained as $\beta = \beta_t$, where $t = \arg \min_i \sum_{kl} E_{kl}^{(i)}$, where the sum is being taken over all low-frequency DCT modes that participate in our calculations (see Section 4.1).

The estimated relative number of modifications improves dramatically when the double compression detection is added to the detection routine. The overall accuracy of the estimated ratio β is slightly lower when compared to the results obtained for cover images that were not JPEG compressed.

Another case of test images that may produce large errors in our detection scheme are images that exhibit very different block frequency characteristics after the cropping. This “spatial resonance” may occur when the cover image contains some regular structure with a characteristic length comparable to the block size, such as the metal grid in Figure 3. Fortunately, it is easy to identify such images both visually and algorithmically and take appropriate measures. One possibility is to use those frequency modes that are most stable with respect to cropping and avoid those that exhibit strong resonant behavior. In our tests, we have encountered only two images with spatial resonance among hundreds of images randomly selected from different sources (see Figure 4).

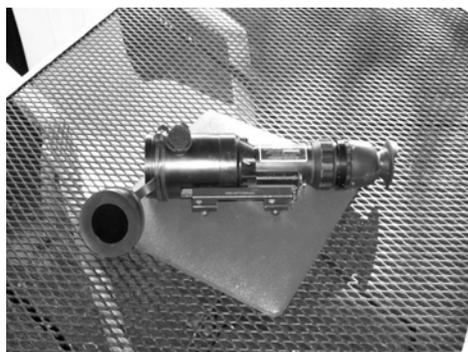


Figure 3: Example of an image with spatial resonance. The same image cropped by 4 and 4 pixels has very different block frequency characteristics than the original image.

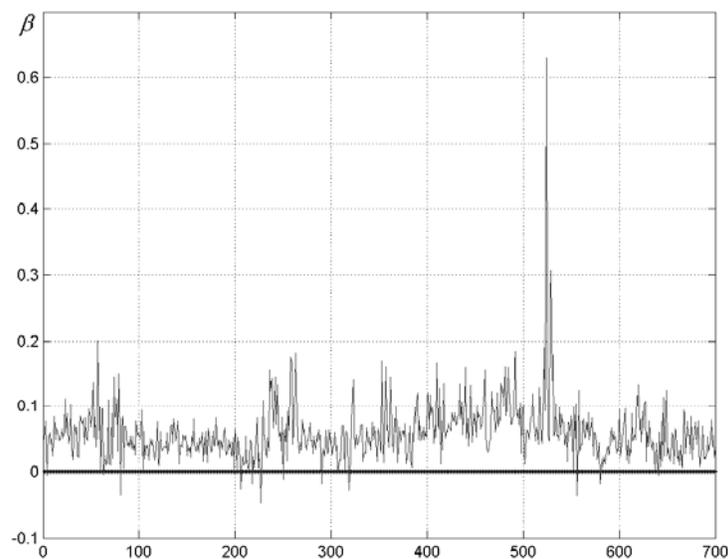


Figure 4: Relative number of modifications β detected in 670 cover images.

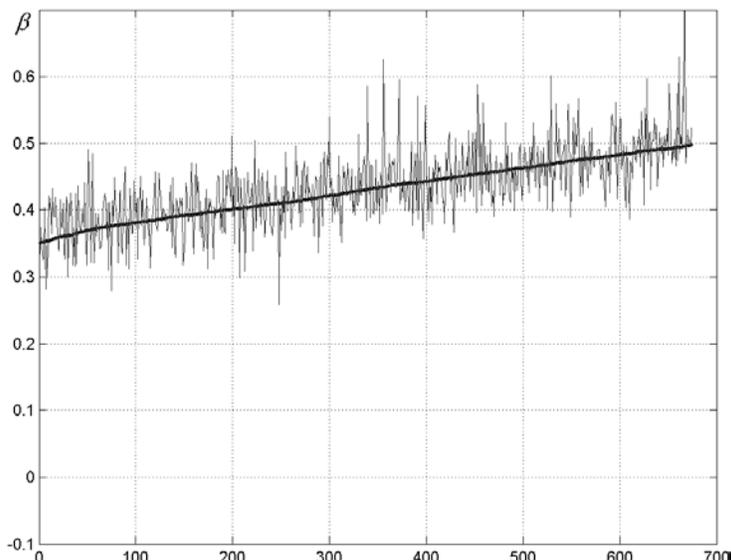


Figure 5: Relative number of modifications β (thick line) and the estimated β for 670 test images embedded with messages corresponding to $\beta \in [0.3, 0.5]$.

4.5 Experimental results

For testing purposes, we have used a database of 900 grayscale images of natural scenes. All images were obtained using a digital camera and originally stored as JPEG images. We have resized them all to 800×600 pixels before processing with F5. Thus, none of the test images showed double compression artifacts. First, we processed all images using F5 with 80% quality factor without embedding any messages and then applied our detection scheme to estimate the number of modifications β . The detection results are shown in Figure 4. The results indicate that the estimated β has a small systematic error. We tend to estimate a small positive number of changes ($\beta_0 = 0.056 \pm 0.037$) in cover images. Also, we can see two outliers around the index 520 that correspond to cases of uncorrected spatial resonance. Then, we have embedded messages of different sizes corresponding to the value of β ranging from 0.3 to 0.5. The images were sorted by β and then processed using our detection algorithm. The results are shown in Figure 5. The error between the true β and the estimated β was 0.034 ± 0.03 .

5. OUTGUESS

The OutGuess steganographic algorithm was proposed by Neils Provos¹⁰ to counter the statistical chi-square attack¹⁵. In the first pass, similar to J-Steg¹², OutGuess embeds message bits along a random walk into the LSBs of coefficients while skipping 0's and 1's. After embedding, the image is processed again using a second pass. This time, corrections are made to the coefficients that were not visited during the first pass to make the stego image histogram match the cover image histogram.

5.1 Analyzing the embedding mechanism

Because OutGuess preserves the first order statistics, the image histogram cannot be used as the distinguishing statistics. On the other hand, even though the global image histogram is preserved, the histograms of individual DCT coefficients are not necessarily preserved. They are only preserved if the shape of the individual histogram is the same as the shape of the global histogram. However, this distinguishing statistics proved unreliable in our initial tests because the differences between the histograms of individual DCT coefficients between the cover and stego images were too small to be useful for detection.

Instead of using histograms, we turned our attention to the fact that the embedding mechanism in OutGuess is overwriting the LSBs. This means that embedding another message into the stego image will partially cancel out and will thus have a different effect on the stego image than on the cover image. The embedding process introduces noise

into the DCT coefficients and thus increases the discontinuities in the spatial domain along the boundaries of 8×8 JPEG blocks. However, due to partial canceling of repeated LSB embedding, this increase in spatial discontinuities will be smaller when we re-embed message bits in the stego image than the increase for the cover image. This increase is a candidate for the distinguishing statistics S . As in Section 4.1, we will use the same trick with cropping and recompressing the stego image to obtain the baseline value for the distinguishing statistics.

In this section, we explain the algorithm for grayscale images. Extension to color images should be obvious. Let $h(d)$, $d = \dots, -2, -1, 0, 1, 2, \dots$ be the histogram of the quantized DCT coefficients from the cover image. Let P be the total number of coefficients different from 0 and 1:

$$P = \sum_{\substack{i \neq 0 \\ i \neq 1}} h(i).$$

We will call those coefficients usable coefficients. OutGuess first calculates the maximal length of a randomly-spread message that can be embedded in the image while making sure that one will be able to make corrections to adjust the histogram to its original values. After embedding m pseudo-random bits in the LSBs of the cover image in randomly selected usable coefficients, the histogram values $h(2i)$, $h(2i+1)$ will be changed to

$$\begin{aligned} h(2i) &\rightarrow h(2i) - \alpha [h(2i) - h(2i+1)], \\ h(2i+1) &\rightarrow h(2i+1) + \alpha [h(2i) - h(2i+1)], \end{aligned}$$

where $2\alpha = m/P$. Let us assume that, for example, $h(2i) > h(2i+1)$. After embedding, there must be enough coefficients with value $2i+1$ to make necessary corrections. Thus, $h(2i+1) - 2\alpha h(2i+1) = \alpha(h(2i) - h(2i+1))$, which gives

$$\alpha_i = \frac{h(2i+1)}{h(2i+1) + h(2i)}.$$

This condition must be satisfied for all histogram pairs $(h(2i), h(2i+1))$. Thus, the maximal number of bits that can be embedded in the image with appropriate corrections is $2aP$, where $a = \min_i \alpha_i$.

After embedding a message of size $2paP$ bits, $0 \leq p \leq 1$, in the cover image (we call such a message a p -percent message), due to the correction step, the number of changes for values $2i$ and $2i+1$ are both $pah(2i)$, assuming $h(2i) > h(2i+1)$. Thus, the total number of changes (both due to embedding and correction) is

$$T_p = 2pa \sum_{i \neq 0} \bar{h}(2i) = paP + pa \sum_{i \neq 0} |\bar{h}(2i) - \underline{h}(2i)|, \quad (4)$$

where $\bar{h}(2i) = \max(h(2i), h(2i+1))$ and $\underline{h}(2i) = \min(h(2i), h(2i+1))$ for each i . The first term in (4) is due to message embedding, the second term due to corrections.

Because OutGuess introduces random changes into the quantized coefficients, the spatial discontinuities at the boundaries of all 8×8 blocks will increase. We will measure the discontinuity using the blockiness measure (6). We take the increase of this blockiness measure after embedding a 100% message again using OutGuess as the distinguishing statistics. This increase will be smaller for the stego image than for the cover image because of the partial cancellation of changes.

To mathematically analyze the proposed idea, we first calculate the number of changes after consecutive embedding of two messages in one image. Given a set of n integers, if we randomly select a subset U consisting of u integers and flip their LSBs and then do the same again with another randomly chosen subset V with v integers, the number of integers with flipped LSBs will be equal to $|U \dot{-} V|$, where the symbol “ $\dot{-}$ ” denotes the symmetric set difference and $|A|$ is the

cardinality of A . This is because the integers in $U \cap V$ will be flipped twice and thus unchanged. Consequently, the total expected number of integers with flipped LSBs will be $u + v - 2uv/n$.

Therefore, if we embed an additional message of size $2qaP$, $0 \leq q \leq 1$, into the image that already holds $2paP$ bits, the expected values of changes for the values $2i$ and $2i+1$ are

$$\begin{aligned} pa\bar{h}(2i) + qa\bar{h}(2i) - 2pqa^2\bar{h}(2i) &= a\bar{h}(2i)(p + q - 2pqa) \text{ and} \\ pa\bar{h}(2i) + qa\bar{h}(2i) - 2pqa^2\bar{h}^2(2i) / \underline{h}(2i+1) &= a\bar{h}(2i)(p + q - 2pqa\bar{h}(2i) / \underline{h}(2i+1)), \end{aligned}$$

respectively. Thus, the total number of expected changes in the cover image after consecutive embedding of two independent randomly-spread messages of size $2paP$ and $2qaP$ bits, $0 \leq p, q \leq 1$, is

$$T_{pq} = 2a \sum_{i \neq 0} \bar{h}(2i) \left(p + q - apq \left(1 + \frac{\bar{h}(2i)}{\underline{h}(2i)} \right) \right). \quad (5)$$

The measure of blockiness at the block boundaries will be calculated using the following formula

$$B = \sum_{i=1}^{\lfloor (M-1)/8 \rfloor} \sum_{j=1}^N |g_{8i,j} - g_{8i+1,j}| + \sum_{j=1}^{\lfloor (N-1)/8 \rfloor} \sum_{i=1}^M |g_{i,8j} - g_{i,8j+1}|, \quad (6)$$

where g_{ij} are pixel values in an $M \times N$ grayscale image and $\lfloor x \rfloor$ denotes the integer part of x .

5.2 Distinguishing statistics and parameter estimation

We have compelling experimental evidence that the blockiness B increases linearly with the number of DCT coefficients with flipped LSBs. The slope of this linear dependency is largest for the cover image and becomes smaller for an image that already contains a message. We use this slope as the distinguishing statistics S to estimate the message length. It is shown below using Equation (5) that, assuming the blockiness B is a linear function of p , the slope is again a linear function of p . Thus, we need to determine two unknown parameters in this linear dependency, which will be done by estimating the blockiness B for the cover image and for the stego image with maximal embedded message.

The detection consists of the following steps:

1. Decompress the stego image, calculate its blockiness and denote $B_s(0)$.
2. Using OutGuess, embed the maximal length message in the stego image ($2aP$ bits), decompress, calculate the blockiness, and denote $B_s(1)$. Calculate the slope $S = B_s(1) - B_s(0)$.
3. Crop the decompressed stego image by 4 columns. This image will be the baseline image that we will use to calibrate the slope. Compress the baseline image using the same JPEG quantization matrix as that of the stego image. Decompress to the spatial domain and calculate its blockiness $B(0)$.
4. Using OutGuess, embed the maximal length message in the cropped image and calculate the blockiness $B(1)$.
5. Use the fully-embedded image from Step 4 and, again, using OutGuess, embed the maximal length message in it denoting its blockiness $B1(1)$.
6. Calculate the secret message length using Equation (7) (see the derivation below)

The slope $S_0 = B(1) - B(0)$ is what we would expect for the original cover image ($p = 0$). The slope $S_1 = B1(1) - B(1)$ is what we would obtain for an image with maximal embedded message ($p = 1$). The slope $S = B_s(1) - B_s(0)$ for the stego image will be somewhere in between these two slopes, $S \in [S_1, S_0]$ corresponding to an unknown message length p . We use linear interpolation to obtain the formula for p , $S = S_0 - p(S_0 - S_1)$, which gives us

$$p = \frac{S_0 - S}{S_0 - S_1}. \quad (7)$$

The linear interpolation and Equation (7) can be justified using Equation (5) for the number of changes. Assuming the blockiness is a linear function of the number of DCT coefficients with flipped LSBs, we can write $B(p) = c + dT_p$, where T_p is the number of coefficients with flipped LSBs after embedding a message of length $2paP$ bits, and c and d are constants. Using (5) we can write

$$\begin{aligned} S_1 &= B1(1) - B(1) = d(T_{11} - T_{10}) = 2ad \sum_{i \neq 0} \bar{h}(2i) \left(1 - a \left(1 + \frac{\bar{h}(2i)}{\underline{h}(2i)} \right) \right) \\ S_0 &= B(1) - B(0) = d(T_{10} - T_{00}) = 2ad \sum_{i \neq 0} \bar{h}(2i) \\ S &= Bs(1) - Bs(1) = d(T_{p1} - T_{p0}) = 2ad \sum_{i \neq 0} \bar{h}(2i) \left(1 - ap \left(1 + \frac{\bar{h}(2i)}{\underline{h}(2i)} \right) \right) \end{aligned}$$

which, after simple algebra, confirms Equation (7).

5.3 Correcting for double compression for OutGuess

Equation (7) generally provides an accurate estimate of the secret message length. The exceptions are when the image sent to OutGuess is already a JPEG file or when the stego image exhibits spatial resonance (both the double compression effect and spatial resonance are commented upon in Section 4.4). Fortunately, OutGuess preserves the histogram and this enables us to recover Q_c using a simpler method than for F5. To incorporate detection of double compression and correct for it, Step 3 is replaced with:

3'. Detect the primary quality factor Q_c . Crop the decompressed stego image by 4 columns. Compress the cropped image using Q_c , decompress, and recompress using Q_s – a process that effectively simulates what happens during the embedding. Decompress to the spatial domain and calculate its blockiness $B(0)$.

We opted for the following simple algorithm to detect Q_c . Let $h_{ij}(d)$ is the histogram of values of the (i, j) -th DCT mode for the stego image and let $h_{ij}(d, Q)$ is the same for the cropped stego image that has been compressed using the quality factor Q , decompressed, and recompressed using the stego image quality factor Q_s . We calculate Q_c as the quality factor that minimizes the difference between $h_{ij}(d, Q)$ and $h_{ij}(d)$ for those DCT modes (i, j) that correspond to the lowest-frequency DCTs (1,2), (2,1), (2,2):

$$Q_c = \arg \min_Q \sum_{(i,j)} \sum_d |h_{ij}(d) - h_{ij}(d, Q)|^2.$$

We have tested this algorithm on 70 test grayscale 600×800 JPEG images with both quality factors ranging from 70 to 90. In all but four cases we estimated the cover image quality factor correctly.

5.4 Experimental results

The same database of 70 images was used for evaluation of the performance of our detection method. Among the 70 test images, 24 of them were processed using OutGuess with message sizes ranging from the maximal capacity to zero. Because all test images were originally stored in the JPEG format, we ran our double compression correction algorithm in all cases. Since the detection algorithm contains randomization, we have repeated the detection 10 times for each image and averaged the p values (7). The results are shown in Figure 6. On the y axis is the relative number of changes due to embedding T_p/aP (see Equation (4)) and on the x axis is the image number. Assuming the distribution of the difference between the estimated and actual values is Gaussian, the estimation error is -0.0032 ± 0.0406 . From our experiments with Equation (1) on test images, we determined that the number of changes due to the correction step is about 1/3 of the changes due to message embedding. Thus, on average the total number of changes due to embedding m bits is $T_p = m/2 (1+1/3)$. This means that the error for the estimated message length m is $-0.48 \pm 6\%$ of total capacity.

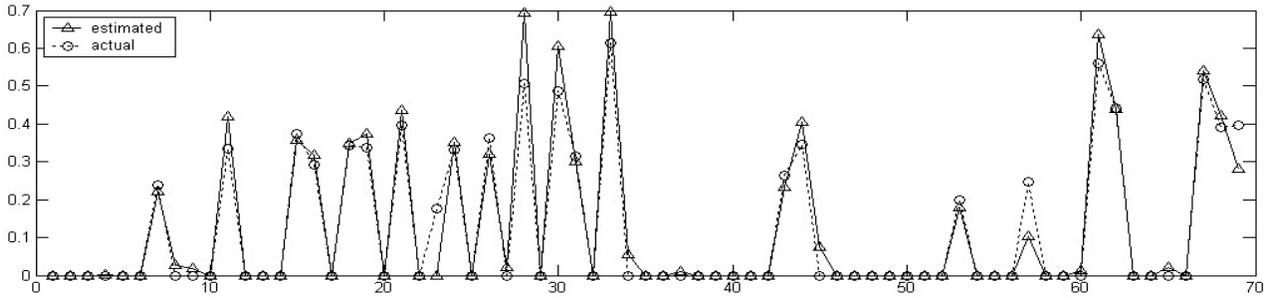


Figure 6: The actual relative number of changes T_p/aP (circles) compared to the calculated number of changes (triangles) for 70 test JPEG images resized to 600×800 pixels obtained using a digital camera Kodak DC 290. The quality factors for the stego and cover image ranged between 70 and 90. The results include correction for double compression.

6. CONCLUSIONS

In this paper, we describe a new methodology for detecting hidden messages and estimating their length in JPEG images. This approach does not use thresholds or a training database to infer the message presence or its length. It is based on identifying a macroscopic quantity S that sensitively changes with the number of embedded bits m , estimating or deriving the functional form of $S(m)$ as a function of m , and deriving the parameters of S either analytically or from experiments. Finally, we calculate the unknown message length q by solving the equation $S(q) = S_{stego}$, where S_{stego} is the value of S evaluated for the stego image. The quantity S is called the distinguishing statistics. We show what quantities S are useful for the F5 algorithm and for OutGuess. The performance of each detection method is evaluated on a test database of images.

One of the lessons that can be learned from this paper is that in order to develop a high-capacity steganographic method for JPEGs, one needs to avoid making predictable changes to all macroscopic characteristics of the JPEG file that are approximately invariant with respect to recompression on a geometrically deformed stego images. However, this task seems to be quite difficult if we insist on embedding one bit in each non-zero DCT coefficient. Also, another lesson is that one should abandon the concept of LSB flipping for embedding and instead use incrementing/decrementing the coefficient values as already pointed out by Westfeld¹⁴.

In the future, we intend to use the proposed detection paradigm for determination of upper bounds on steganographic capacity. As elaborated on in Section 2, steganographic capacity is the maximal number of bits that can be embedded in a given image using a specific method that cannot be detected with a better algorithm than random guessing.

To derive an estimate for the relative steganographic capacity C_R , we use a large test database of images and apply the proposed detection methods to all cover images obtaining the distribution P_0 of relative message length p detected in cover images. Then, we embed messages with a fixed relative message size q into all database images and calculate its distribution P_q . The next step is to test the statistical hypothesis that both distributions are equal. If we model the distribution P_0 as Gaussian, it is then possible to test whether the samples from P_q fit P_0 using the Kolmogorov-Smirnov test⁵. The smallest value of q that does not pass this test determines an *upper bound* for steganographic capacity of the test database. If the database is large containing a wide diversity of images, the estimate for the steganographic capacity will likely be valid for other databases as well.

In another interpretation of our results, we can use the distribution P_0 to estimate the threshold T_a for the message length that would limit the probability of false positives below certain specified limit a . This threshold T_a can then be used to determine the probability of missed detections assuming a message of relative length of q has been embedded.

ACKNOWLEDGEMENTS

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.

REFERENCES

1. R.J. Anderson and F.A.P. Petitcolas, "On the Limits of Steganography". *IEEE Journal of Selected Areas in Communications*, Special Issue on Copyright and Privacy Protection, vol. **16**(4), pp. 474–481, 1998.
2. C. Cachin, "An Information-Theoretic Model for Steganography", In: *LNCS vol. 1525*, Springer-Verlag, New York, pp. 306–318, 1998.
3. R. Chandramouli and N. Memon, "Analysis of LSB Based Image Steganography Techniques". *Proc. of ICIP 2001*, Thessaloniki, Greece, October 7–10, 2001.
4. R. Chandramouli and N. Memon, "A Distributed Detection Framework for Watermark Analysis". *Proc. ACM Multimedia Workshop on Multimedia and Security*, Los Angeles, California, 2000.
5. R. D'Agostino and M. Stephens, *Goodness-of-Fit Techniques*. Marcel Dekker, Inc., New York, 1986.
6. H. Farid and L. Siwei, "Detecting Hidden Messages Using Higher-Order Statistics and Support Vector Machines". *Pre-proceedings 5th Information Hiding Workshop*, Netherlands, Oct. 7–9, 2002.
7. J. Fridrich, M. Goljan, and D. Hoge, "Steganalysis of JPEG Images: Breaking the F5 Algorithm". *Pre-proceedings of 5th Information Hiding Workshop*, Netherlands, Oct. 7–9, 2002.
8. J. Fridrich, M. Goljan, and R. Du, "Steganalysis based on JPEG compatibility". *Proc. SPIE Multimedia Systems and Applications IV*, Denver, Colorado, pp. 275–280, 2001.
9. Katzenbeisser and F.A.P. Petitcolas, "On Defining Security in Steganographic Systems". *Proc. Electronic Imaging, Photonics West, SPIE 2002*, San Jose, California, 2002.
10. N. Provos, "Defending Against Statistical Steganalysis", 10th USENIX Security Symposium, Washington, DC, 2001.
11. N. Provos and P. Honeyman, "Detecting Steganographic Content on the Internet", CITI Technical Report 01-11, 2001.
12. Steganography software for Windows, <http://members.tripod.com/steganography/stego/software.html>, 2002.
13. A. Westfeld, "Detecting Low Embedding Rates". *Pre-proceedings 5th Information Hiding Workshop*. Noordwijkerhout, Netherlands, Oct. 7–9, 2000.
14. A. Westfeld, "High Capacity Despite Better Steganalysis (F5–A Steganographic Algorithm)", In: *LNCS vol.2137*, Springer-Verlag, New York Heidelberg Berlin, pp. 289–302, 2001.
15. A. Westfeld and A. Pfitzmann, "Attacks on Steganographic Systems", In: *LNCS vol.1768*, Springer-Verlag, Berlin, pp. 61–75, 2000.