Multimedia Systems (2003)
Digital Object Identifier (DOI) 10.1007/s00530-003-0100-9

Multimedia Systems
© Springer-Verlag 2003

Quantitative steganalysis of digital images: estimating the secret message length

Jessica Fridrich[1], **Miroslav Goljan**[1], **Dorin Hogea**[2], **David Soukal**[2]

[1] Department of Electrical and Computer Engineering, SUNY Binghamton, Binghamton, NY 13902-6000, USA
[2] Department of Computer Science, SUNY Binghamton, Binghamton, NY 13902-6000, USA

Abstract. The objective of steganalysis is to detect messages hidden in cover objects, such as digital images. In practice, the steganalyst is frequently interested in more than whether or not a secret message is present. The ultimate goal is to extract and decipher the secret message. However, in the absence of the knowledge of the stego technique and the stego and cipher keys, this task may be extremely time consuming or completely infeasible. Therefore, any additional information, such as the message length or its approximate placement in image features, could prove very valuable to the analyst. In this paper, we present general principles for developing steganalytic methods that can accurately estimate the number of changes to the cover image imposed during embedding. Using those principles, we show how to estimate the secret message length for the most common embedding archetypes, including the F5 and OutGuess algorithms for JPEG, EzStego algorithm with random straddling for palette images, and the classical LSB embedding with random straddling for uncompressed image formats. The paper concludes with an outline of ideas for future research such as estimating the steganographic capacity of embedding algorithms.

Key words: Steganalysis – Steganography – Capacity – Attacks – Covert

1 Basics of steganography and steganalysis

The purpose of steganography is to communicate information in a stealth manner so that anyone who inspects the messages being exchanged cannot collect enough evidence that the messages hide additional secret data. As opposed to cryptography that makes the communication unintelligible to those who do not know the proper cipher keys, steganography makes the communication inconspicuous or invisible. An old example of steganography is writing messages between the lines of an ordinary letter using invisible ink. Another simple technique is to use tiny markers (pinholes) to mark letters in a text. A very well-written historical perspective on steganography can be found in [20]. In today's digital world, invisible ink and paper have been replaced by much more versatile and practical covers for hiding messages – digital documents, images, video, and audio files. All electronic documents containing perceptually irrelevant or redundant information provide a good environment for steganographic communication. The object that holds the secret information is called the cover object. After a secret message has been hidden in the cover, the cover object becomes the stego object.

The problem of steganography was formulated for the first time by Simmons as the prisoners' problem [18]. In this scenario, Alice and Bob are prisoners locked in separate cells. All communication between them goes through a warden. The warden can read their messages or even modify them at will (reformulate phrases, inject noise, etc.). The goal of Alice and Bob is to come up with a method of communication that would enable them to hatch an escape plan, i.e., to send messages so that the warden will be unaware of their existence. Once the warden finds out about the secret communication, she will cut the communication channel. We note that Alice and Bob may agree on a communication method and on a secret key before imprisonment. The key may, for example, be used for seeding a pseudorandom number generator (PRNG) for selecting the letters or pixels that will carry secret message bits.

We distinguish different steganographic scenarios based on what information is available to the warden and what she can do to the messages:

Pure steganography: The warden may not know the method Alice and Bob are using. This is called pure steganography [13]. However, Alice and Bob should not rely on the fact that the warden does not know the communication method. Historically, this "security through obscurity" principle was never successful in the long term.

Secret-key steganography: The warden has full knowledge of the communication algorithm but does not know the secret key shared by Alice and Bob. This scenario complies with Kerckhoff's principle, which is always used in cryptography. Some researchers argue, however, that in any practical situation, this principle is likely too strong. This is because a complete knowledge of the steganographic system includes a complete knowledge of the source of cover objects, which may be very difficult for the warden to obtain.

Passive warden: If the warden can only observe the communication but not modify the messages, the communication scenario is called the passive warden scenario.

Active warden: If the warden is allowed to make changes to the messages, we talk about an active (or malicious) warden scenario. Obviously, under the active warden scenario, Alice and Bob must resort to robust data hiding methods and quite likely sacrifice the message payload for robustness.

In this paper, we will position ourselves into the role of the passive warden who inspects digital images. In particular, our goal is to estimate the number of embedding modifications (and thus the secret message length). The main contribution of this paper is to put several detection schemes previously proposed by the authors under one umbrella and present a concise, unified approach to quantitative steganalysis for all three major image archetypes – transform, uncompressed, and palette formats. We start the next section by introducing the concepts of steganographic security and steganographic capacity in an informal manner. Then in Sect. 3, we outline our strategy for the design of steganalytic techniques capable of estimating the secret message length. In Sect. 4, we describe steganalytic algorithms for F5 and OutGuess for JPEG images. Uncompressed raw formats are investigated in Sect. 5, while Sect. 6 is devoted to palette images. The paper is concluded in Sect. 7 with an outline of how the proposed methods can be used for estimating steganographic capacity.

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 have been proposed in the literature [1,2]. The problem with most definitions is that they assume observers have unlimited computational power and detailed statistical knowledge of the source of cover images. In practice, these assumptions are rarely satisfied or feasible to obtain. Attempts to define the concept of steganographic security in a relevant but practical way include the recent work of Katzenbeisser and Petitcolas [14].

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 pseudorandom 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 stegokeys K uniformly distributed in the key space and all pseudorandom messages. When we say safely, we mean 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 attempt to 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 a technique for which $C_A = 0$ is that used 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 another color that is 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 can reveal the fact that the stego image has been manipulated. As another, less trivial example, we cite the JPEG compatibility analysis [9]. 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 exact estimates of steganographic capacity. A notable exception is the work of Chandramouli et al. [4], who gave a theoretical analysis of the steganographic capacity for the LSB embedding (least significant bit) in the spatial domain and for a binary-valued embedding distortion [3]. At the end of this paper, we outline how the detection methods described in this paper can be used to determine an upper bound on steganographic capacity for the most common embedding paradigms for digital images in various formats.

3 General detection methodology

In this section, we outline the principles 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 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 in 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 (e.g., 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*.

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 (Sect. 4). This is because the JPEG file is formed by quantized DCT (discrete cosine transform) 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 four 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, macroscopic characteristics will be approximately preserved. For detection of F5, we use the histograms of individual DCT coefficients as the distinguishing statistics because the changes in histograms are proportional to the number of embedding changes. Because OutGuess uses LSB embedding, we measure the degree of LSB randomization in the stego file by embedding additional messages in the stego image (and in the cropped/recompressed image) and derive the unknown message length from the increase in the sum of spatial discontinuities along 8×8 boundaries.

For uncompressed raw formats (e.g., BMP), we cannot use the same approach as for JPEGs because the embedding distortion is too small and there is little hope that we will be able to obtain an approximation to the cover image. In Sect. 5, we describe an attack on LSB embedding (the RS steganalysis), and in Sect. 6 we deal with detection of LSB embedding in GIF images with a preordered palette (pairs analysis). Both methods are based on the fact that LSB embedding creates an imbalance between neighboring grayscales – the values $2i$ and $2i+1$ flip into one another but never to $2i-1$ or $2i+2$. Thus, the proposed distinguishing statistics are sensitive to changes in LSBs but are roughly invariant when 1 is added to all pixels in the cover image.

4 JPEG images

In the past, several steganographic techniques were proposed for the JPEG format – J-Steg, JP Hide&Seek, F5, OutGuess. In this section, we show how the ideas presented in Sect. 3 apply to the recently developed F5 algorithm and to OutGuess. These two algorithms represent the current state of the art in JPEG steganography.

4.1 The F5 algorithm

The F5 steganographic algorithm was introduced by Westfeld [22]. The F5 algorithm 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 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 nonzero 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. A detailed description of F5 can be found in the original paper [22].

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 attack [21] or its generalized versions (17,23]. 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 proved very effective for designing detection methods for most current steganographic methods for JPEGs. We crop the stego image by four columns and recompress the cropped image using the same quantization table as that of the stego image. The spatial shift by four pixels breaks the quantized structure of block DCT coefficients, and 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 a certain number of changes.

4.1.1 Distinguishing statistics

Let $h(d)$, $d = 0, 1, \ldots$ 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 by $h_{kl}(d)$ the total number of AC coefficients corresponding to the frequency (k, l), $1 \le k, l \le 8$, 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 nonzero AC coefficient will be modified is $\beta = n/P$, where P is the total number of nonzero AC coefficients ($P = h(1) + h(2) + \ldots$). Because the selection of the coefficients is random in F5, the expected values of the histograms H_{kl} of the stego image are

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

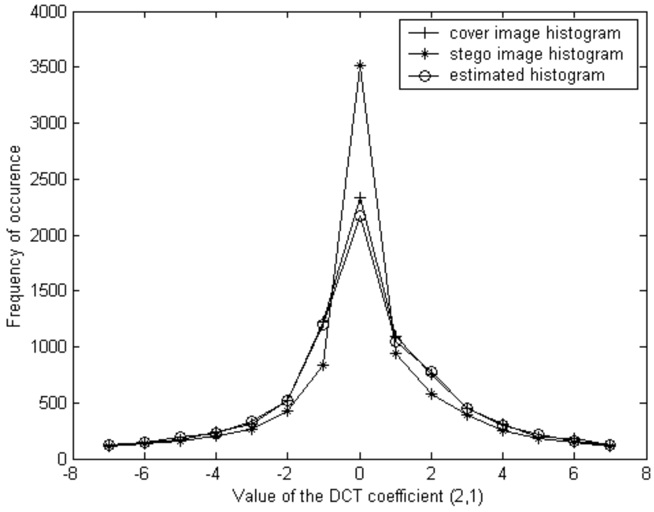

Fig. 1. The effect of F5 embedding on the histogram of the DCT coefficient (2,1)

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 Eq. 1 and estimate β as the value that gives us the best agreement with the cover image histogram. We have experimented with various 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 $d = 1$) undergo the largest change during embedding (see Fig. 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 Eq. 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 Eq. 2 leads to the following formula for β:

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

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 not to 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.

4.1.2 Parameter 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 four columns, and finally 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 the same trick will be used for attacking OutGuess in Sect. 4.2.

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 four 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 Fig. 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 for the low-frequency DCT mode (2,1) (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 prefiltering, the spatial shift by four pixels introduces some nonzero 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 nonzero DCTs.

4.1.3 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 bit 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 $m = n(1{-}_S)$ for the number of changes due to message embedding. Assuming the $(1, 2^k - 1, k)$ matrix embedding [23], 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.

4.1.4 Correcting for double compression for F5

When the cover image is stored in the JPEG format, both 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 Sect. 4.1.2 will produce a histogram similar to the broken line in Fig. 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.

We calculate the ratio β for a fixed set of quantization tables, $\{Q_1, Q_2, \ldots, Q_r\}$. 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, $i = 1, \ldots, r$ 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 Eq. 1 with $\beta = \beta_i$:

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

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 Sect. 4.1.1).

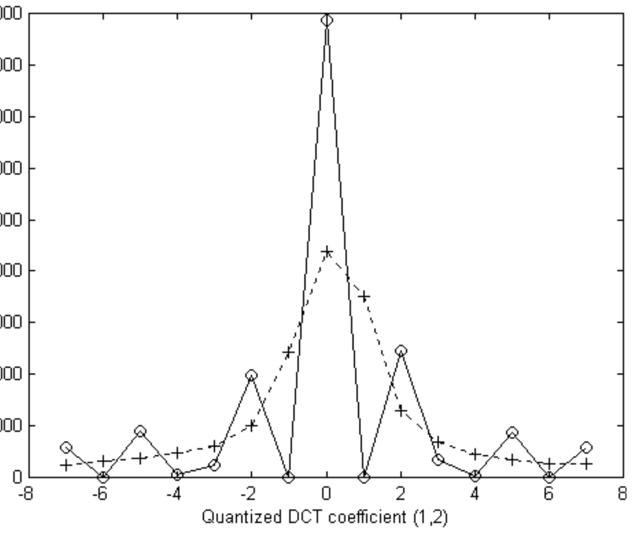

Fig. 2. Effect of double compression on the histogram of quantized DCT coefficients. The *broken line* is the image histogram with a single compression, the *solid line* after double compression with a lower quality factor being the first one. The histogram corresponds to the DCT coefficient (1,2)

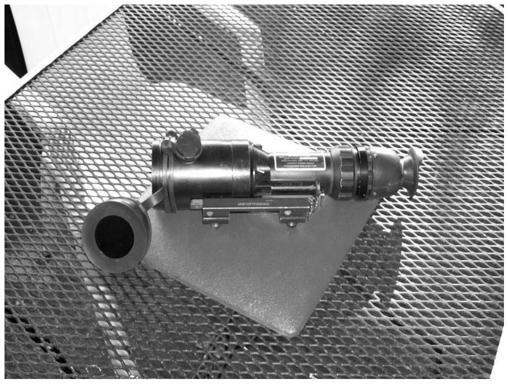

Fig. 3. Example of an image with spatial resonance. The same image cropped by four and four pixels has very different block frequency characteristics than the original image

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 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 Fig. 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 res-

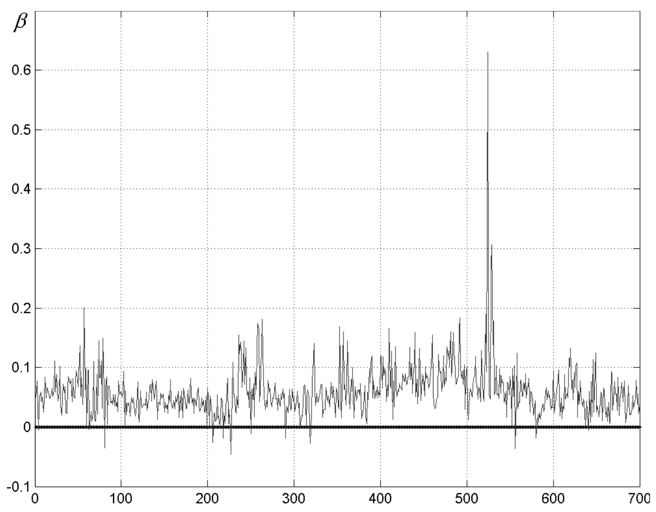

Fig. 4. β detected in 670 cover images

onance among hundreds of images randomly selected from different sources (see Fig. 4).

4.1.5 Experimental results

For testing purposes, we 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 message and then applied our detection scheme to estimate the number of modifications β. The detection results are shown in Fig. 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 Fig. 5. The error between the true β and the estimated β was 0.034 ± 0.03.

4.2 OutGuess

The OutGuess steganographic algorithm was proposed by Neils Provos [17] to counter the statistical chi-square attack [18,21]. In the first pass, like J-Steg, OutGuess 0.2 embeds message bits along a random walk into the LSBs of coefficients while skipping 0s and 1s. 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.

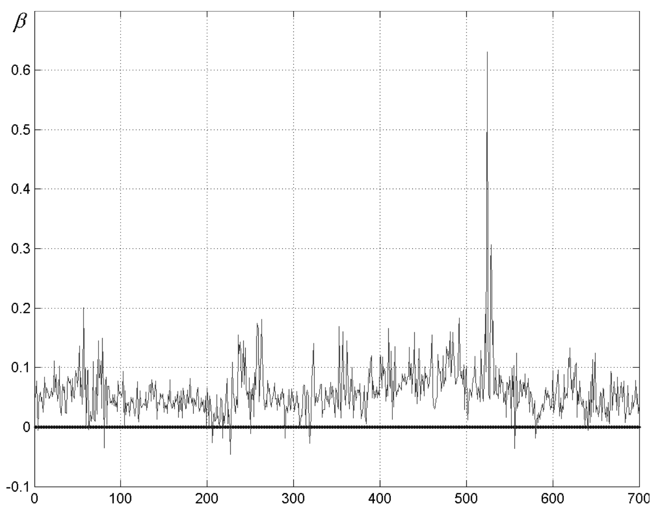

Fig. 5. True value of β (*thick line*) vs. the estimated β for 670 test images embedded with messages corresponding to $\beta \in [0.3, 0.5]$

4.2.1 Analyzing the embedding mechanism

Because OutGuess preserves the first-order statistics, the image histogram cannot be used as the distinguishing statistics. It is true, however, that, even though the global image histogram is preserved, the histograms of individual DCT coefficients are not necessarily preserved. They are preserved only 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 from 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 8×8 JPEG blocks. However, due to partial canceling of repeated LSB embedding, this increase in spatial discontinuities will be smaller when we reembed 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 Sect. 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 will work again with grayscale images. Extension to color images should be obvious. Let $h(d)$, $d = \ldots, -2, -1, 0, 1, 2, \ldots$ 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 pseudorandom 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

$$h(2i) \to h(2i) - \alpha[h(2i) - h(2i+1)]$$
$$h(2i+1) \to h(2i+1) + \alpha[h(2i) - h(2i+1)]$$

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 maximum message size 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, $0 \le p \le 1$, 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 (due to both embedding and correction) is

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

where $\bar{h}(2i) = \max(h(2i), h(2i+1))$ and $\underline{h}(2i+1) = \min(h(2i), h(2i+1))$ for each i. The first term in Eq. 4 is due to message embedding, the second term 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 (Eq. 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 $|U \div V|$, where "$\div$" denotes the symmetric set difference and $|A|$ 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 $2qaP$, $0 \le q \le 1$, into the image that already holds $2paP$ bits, the expected values of changes for the values $2i$ and $2i+1$ are

$$pa\bar{h}(2i) + qa\bar{h}(2i) - 2pqa^2\bar{h}(2i) = a\bar{h}(2i)(p + q - 2pqa)$$

and

$$pa\bar{h}(2i) + qa\bar{h}(2i) - 2pqa^2\bar{h}^2(2i)/\underline{h}(2i+1)$$
$$= a\bar{h}(2i)(p + q - 2pqa\bar{h}(2i)/\underline{h}(2i+1))$$

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, $0 \le p$, $q \le 1$, 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 is calculated at the block boundaries 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.

4.2.2 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 Eq. 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 *Bs*(0).
2. Using OutGuess, embed the maximal length message in the stego image ($2aP$ bits), decompress, calculate the blockiness, and denote *Bs*(1). Calculate the slope $S = Bs(1) - Bs(0)$.
3. Crop the decompressed stego image by four 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 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 Eq. 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 $S = Bs(1) - Bs(0)$ 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

$$p = \frac{S_0 - S}{S_0 - S_1} \tag{7}$$

The linear interpolation and Eq. 7 can be justified using Eq. 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 Eq. 5:

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

which, after simple algebra, confirms Eq. 7.

4.2.3 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 Sect. 4.1.4). 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 four 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 $h_{ij}(d)$ be the histogram of values of the (i, j)-th DCT mode for the stego image, and let $h_{ij}(d,Q)$ be 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):

$$Q_c = \arg\min_Q \sum\nolimits_{(i,j)} \sum\nolimits_d |h_{ij}(d) - h_{ij}(d,Q)|^2$$

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.

4.2.4 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 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 ten times for each image and averaged the p values (Eq. 7). The results are shown in Fig. 6. On the y-axis is the relative number of changes due to embedding T_p/aP (see Eq. 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 Eq. 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). Thus, the error for the estimated message length m is $-0.48 \pm 6\%$ of total capacity.

One of the lessons that can be learned from this Sect. 4 is that in order to develop a high-capacity steganographic method for JPEGs, one needs to avoid making predictable changes to macroscopic characteristics of the JPEG file. However, this task seems to be quite difficult if we insist on embedding one bit in each nonzero 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 in [21].

5 LSB for uncompressed raw formats

The methods for JPEG images were based on the fact that it was possible to define a macroscopic quantity that sensitively reacted to the embedding process and whose values could be estimated for the cover and fully embedded images (histogram for F5 and blockiness increase for OutGuess). In the case of LSB embedding in raw image formats, the modifications are so small that one cannot obtain a similarly good and usable approximation to the LSBs of the cover image as could be done using cropping and recompressing for JPEG images. On the other hand, LSB embedding creates an imbalance between neighboring grayscales – during embedding the grayscale values $2i$ and $2i+1$ are flipped into each other but never to $2i-1$ or $2i+2$. Thus, it would be useful to find a distinguishing statistics that is sensitive to LSBs but roughly invariant when we add 1 to all pixels in the cover image. One sensitive measure that satisfies these requirements is the lossless capacity of the LSB plane as defined in [12]. This quite sensitive quantity captures the delicate (and quite weak) relationship between the LSB plane and the remaining seven image bit planes. The lossless capacity is a function of the number of "regular" and "singular" pixel groups (see the definitions below). The number of regular groups is approximately the same if calculated from the cover image or from the cover image after adding 1 to all pixels (the "shifted" image). However, with embedding, these two values diverge as the LSB plane becomes more randomized. Thus, the difference between the two numbers is chosen as the distinguishing statistics.

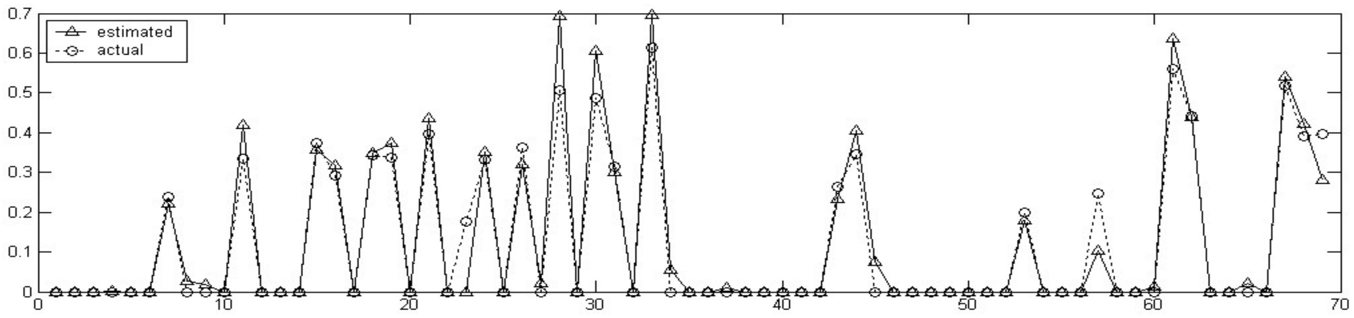

Fig. 6. The actual relative number of changes T_p/aP (*circles*) compared to the calculated number of changes (*triangles*) for 70 test JPEG images resized to 600 × 800 pixels obtained using a digital Kodak DC 290 camera. The quality factors for the stego and cover image ranged between 70 and 90. The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

5.1 Distinguishing statistics

Let us assume that we have a cover image with $M \times N$ pixels and with pixel values from a set P. For example, for an 8-bit grayscale image, $P = \{0, \ldots, 255\}$. We introduce a discrimination function f that assigns a real number $f(x_1, \ldots, x_n) \in \mathbf{R}$ to a group of pixels $G = (x_1, \ldots, x_n)$. As in [12], we use the function f defined as

$$f(x_1, x_2, ..., x_n) = \sum_{i=1}^{n-1} |x_{i+1} - x_i| \tag{8}$$

The function f measures the smoothness of G – the noisier the group G is, the larger the value of the discrimination function becomes.

The LSB embedding increases the noisiness in the image, and thus we expect the value of f to increase after LSB embedding. Let us define the flipping function F_1 as the following permutation on P: $0 \leftrightarrow 1, 2 \leftrightarrow 3, \ldots, 254 \leftrightarrow 255$. Changing the LSB of the gray level x is the same as applying F_1 to x. We also define a dual concept called *shifted* LSB flipping F_{-1} as $-1 \leftrightarrow 0$, $1 \leftrightarrow 2$, $3 \leftrightarrow 4, \ldots, 253 \leftrightarrow 254$, $255 \leftrightarrow 256$, or

$$F_{-1}(x) = F_1(x + 1) - 1, \forall x \in P \tag{9}$$

Finally, for completeness, we also define F_0 as the identity permutation $F(x) = x, \forall x \in P$.

The discrimination function f and the flipping operation F define three types of pixel groups: R, S, and U, depending on how the flipping changes the value of the discrimination function: group G is regular (R) if $f(F(G)) > f(G)$, G is singular (S) if $f(F(G)) < f(G)$, and G is unchanged (U) if $f(F(G)) = f(G)$. Here, $F(G)$ means that we apply the flipping function F to the components of the vector $G = (x_1, \ldots, x_n)$.

If we apply different flipping to different pixels in group G, we capture the assignment of flipping to pixels with a mask M, which is an n-tuple with values -1, 0, and 1. The flipped group $F(G)$ is defined as $(F_{M(1)}(x_1), F_{M(2)}(x_2), \ldots, F_{M(n)}(x_n))$.

In typical images, flipping group G will more frequently lead to an increase in the discrimination function f rather than a decrease. Thus, the total number of regular groups will be larger than the total number of singular groups. Let us denote the relative number of regular groups for a nonnegative mask M as R_M (in percents of all groups) and let S_M be the relative number of singular groups. We have $R_M + S_M \le 1$ and $R_{-M} + S_{-M} \le 1$. The values R_M, S_M, R_{-M}, and S_{-M} will play the role of our distinguishing statistics. An important hypothesis of our steganalytic method is that for typical cover images, the value of R_M is approximately equal to that of R_{-M}, and the same should be true for S_M and S_{-M}:

$$R_M \cong R_{-M} \quad \text{and} \quad S_M \cong \quad S_{-M} \tag{10}$$

We can justify this hypothesis heuristically by inspecting Eq. 9. Using the flipping operation F_{-1} is the same as applying F_1 to an image whose colors have been shifted by one. Because the discrimination function f captures the smoothness, adding the value of 1 to all pixels should not influence the statistics of regular and singular groups in any significant way. Indeed, we have extensive experimental evidence that Eq. 10 holds very accurately for images taken with a digital camera for both JPEG and uncompressed formats. It also holds well for images processed with common image-processing operations and for scanned photographs. The relationship in Eq. 10, however, is violated after randomizing the LSB plane (because of LSB steganography, for example).

Randomization of the LSB plane forces the difference between R_M and S_M to zero as the length m of the embedded message increases. After flipping the LSB of 50% of pixels (which is what would happen after embedding a random message bit into every pixel), we obtain $R_M \cong S_M$. This is just a reformulation of the fact that the lossless capacity of the LSB plane after completely randomizing it must be zero [12]. What may be surprising is that randomizing the LSB plane has the *opposite* effect on R_{-M} and S_{-M}. Their difference *increases* with the length m of the embedded message. The graph that shows R_M, S_M, R_{-M}, and S_{-M} as functions of the number of pixels with flipped LSBs appears in Fig. 7 (the RS diagram). To be precise, the diagram actually shows the expected values of R_M and S_M over the statistical sample of all possible LSB randomizations, but to simplify the notation we use the same symbols for the expected values.

We provide a simple explanation for the peculiar increase in the difference between R_{-M} and S_{-M} for the mask $M = [010]$. Similar arguments could be used for other masks. We define sets $C_i = \{i, 2i + 1\}$, $i = 0, \ldots, 127$, and cliques of groups $C_{rst} = \{G | G \in C_r \times C_s \times C_t\}$. There are 128^3 cliques, each clique consisting of eight groups (triples). The cliques are closed under LSB randomization. For the purpose of our analysis, we recognize four different types of cliques, ignoring horizontally and vertically symmetrical cliques. Ta-

Table 1. Four types of cliques

Clique type	F_1 flipping	F_{-1} flipping
$r = s = t$	$2R$, $2S$, $4U$	$8R$
$r = s > t$	$2R$, $2S$, $4U$	$4R$, $4U$
$r < s > t$	$4R$, $4S$	$4R$, $4S$
$r > s > t$	$8U$	$8U$

ble 1 shows the four clique types and the number of R, S, and U groups under F_1 and F_{-1} after randomization. From the table one can see that while randomization of LSBs has a tendency to equalize the number of R and S groups in each clique under F_1, it will increase the number of R groups and decrease the number of S groups under F_{-1}.

5.2 *Parameter estimation*

As the next step, we estimate the four curves of the RS diagram. Experimental evidence indicates that the R_{-M} and S_{-M} curves are well modeled with straight lines, while second-degree polynomials are appropriate for the "inner" curves R_M and S_M. We can determine the parameters of the curves from the points marked in Fig. 7. We note that Dumitrescu et al. [6] have found a different method of arriving at essentially the same detection method and were able to prove under fairly general assumptions that for the special mask M=[0 1] the curves indeed must be straight lines and parabolas. This result somewhat justifies our assumptions about the functional form of the curves for other masks.

If we have a stego image with a message of an unknown relative length p ($p = 1$ for one bit per pixel) embedded in the LSBs of randomly scattered pixels, our initial measurements of the number of R and S groups correspond to the values R_M(*p*/2), S_M(*p*/2), R_{-M}(*p*/2), and S_{-M}(*p*/2) (Fig. 7). The factor of one half is because, assuming the message is a random bit stream, on average only one half of the pixels will be flipped by message embedding. If we flip the LSBs of *all* pixels in the image and calculate the number of R and S groups, we will obtain the four values R_M(1-*p*/2), S_M(1-*p*/2), R_{-M}(1-*p*/2), and S_{-M}(1-*p*/2) (Fig. 7).

By randomizing the LSB plane of the stego image, we can calculate the values R_M(1/2) and S_M(1/2). Because these two values depend on the particular randomization of the LSBs, we could repeat the process many times and estimate R_M(1/2) and S_M(1/2) from the statistical samples. However, it is possible to avoid this time-consuming statistical estimation and simultaneously make the message length estimation more elegant by accepting two more (natural) assumptions: (1) The point of intersection of the curves R_M and R_{-M} has the same x coordinate as the point of intersection for the curves S_M and S_{-M} (this is essentially Eq. 10). (2) $R_M(1/2) = S_M(1/2)$.

We have experimentally verified the first assumption for a large database of images with unprocessed raw BMPs and JPEGs and processed images. The second assumption essentially says that the lossless capacity in the LSBs of a fully embedded image is zero, which is a well-founded statement [12].

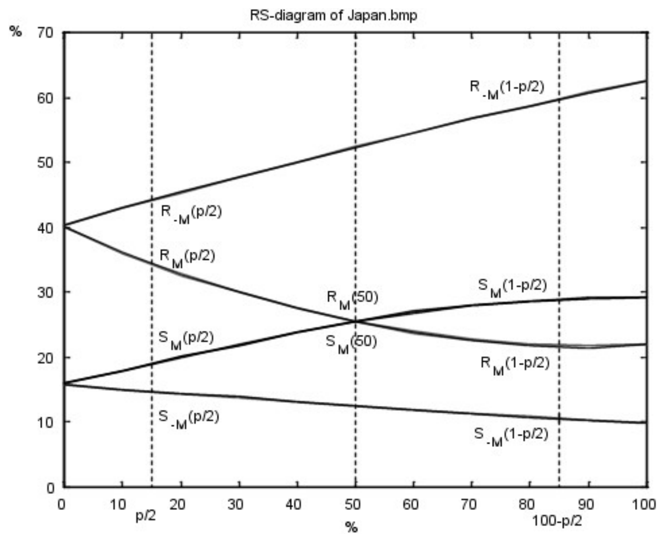

Fig. 7. RS diagram of an image taken by a digital camera. The x-axis is the percentage of pixels with flipped LSBs, and the y-axis is the relative number of regular and singular groups with masks M and $-M$, $M = [0110]$

The number of R and S groups at *p*/2 and 1-*p*/2 define the straight lines and, together with the assumptions (1) and (2) above, also provide enough constraints to uniquely determine the parabolas and their intersections. After rescaling the x-axis so that $p/2$ becomes 0 and $100 - p/2$ becomes 1, which is obtained by the linear substitution $z = (x - p/2)/(1 - p)$, the x-coordinate of the intersection point can be calculated from the root of the following quadratic equation:

$$2(d_1 + d_0)z^2 + (d_{-0} - d_{-1} - d_1 - 3d_0)z + d_0 - d_{-0} = 0$$

where

$$\begin{aligned} d_0 &= R_M(p/2) - S_M(p/2), d_1 \\ &= R_M(1 - p/2) - S_M(1 - p/2), d_{-0} \\ &= R_{-M}(p/2) - S_{-M}(p/2), d_{-1} \\ &= R_{-M}(1 - p/2) - S_{-M}(1 - p/2) \end{aligned}$$

We calculate the message length p from the root z whose absolute value is smaller by

$$p = z/(z - 1/2) \qquad (11)$$

There are three main factors that influence the accuracy of the estimated message length: the initial bias, the noise level or quality of the cover image, and the placement of message bits in the image.

Initial bias: The RS steganalysis may indicate a small nonzero message length due to random variations even for the original cover image. This initial nonzero *bias* could be both positive and negative, and it puts a limit on the achievable accuracy of RS steganalysis. Smaller images tend to have higher variation in the initial bias due to a smaller number of R and S groups. Scans of half-toned images and noisy images exhibit larger variations in the bias as well. On the other hand, the bias is typically very low for JPEG images, uncompressed images obtained by a digital camera, scans of photographs,

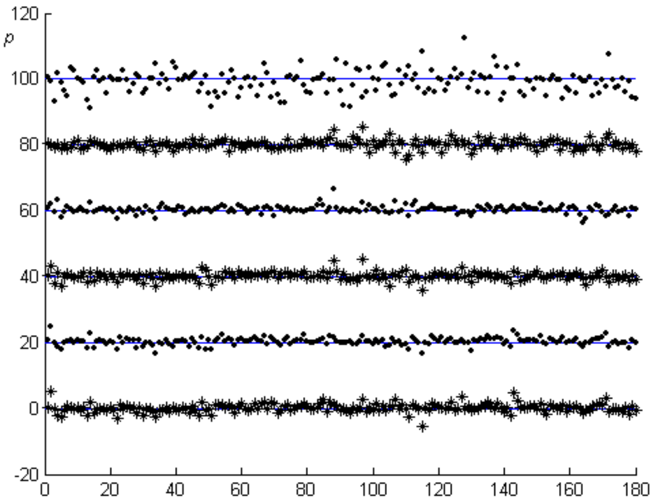

Fig. 8. Estimated message length (in % of maximal message length) for a database of 180 grayscale images

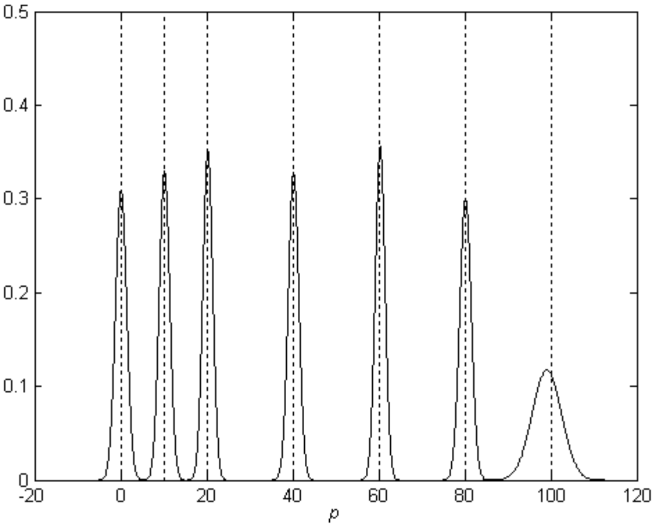

Fig. 9. Distribution of estimated message length compared to the true message length (*vertical lines*) for the same database of 180 grayscale images

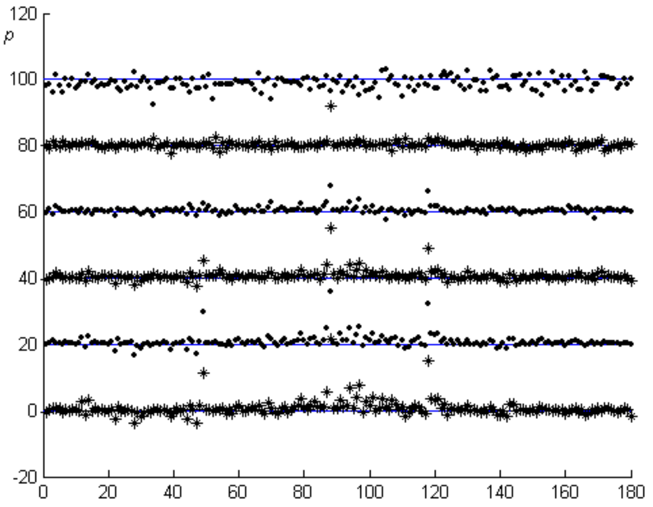

Fig. 10. Estimated message length (in % of maximal message length) for a database of 180 color images

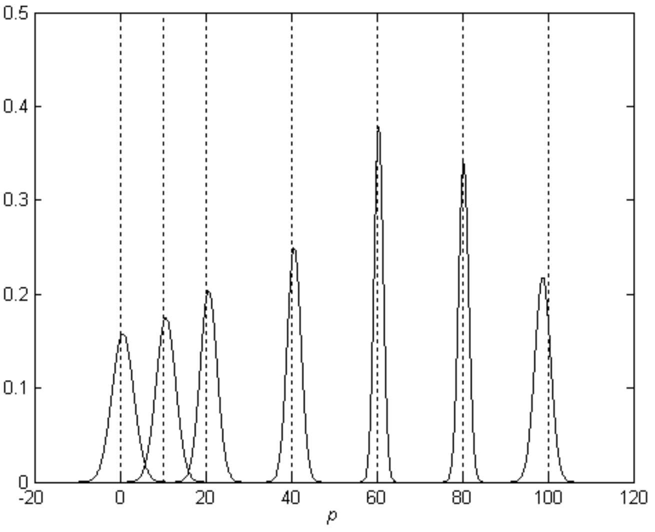

Fig. 11. Distribution of estimated message length compared to the true message length (*vertical lines*) for the same database of 180 color images

and images processed with typical image-processing filters. As another rule of thumb, we state that color images exhibit larger variations in the initial bias than grayscales.

Noise: For very noisy images, the difference between the number of regular and singular pixels in the cover image is small. Consequently, the lines in the RS diagram intersect at a small angle and the accuracy of RS steganalysis decreases. The same is true for low-quality images, overcompressed images, and small images (due to insufficient statistics).

Message placement: The RS steganalysis is more accurate for messages that are randomly scattered in the stego image than for messages concentrated in a localized area of the image. To address this issue, we could apply the same algorithm to a sliding rectangular region of the image. Another possibility is described in the work of Dumitrescu et al. [7], who extend their approach, which originated from RS steganalysis, to messages that are spread nonuniformly in the stego image.

5.3 Experimental results

We have performed tests on a database of 180 grayscale images and another database of 180 24-bit color images. The images were obtained using four different digital cameras and were originally stored as high-quality JPEG images. For our test purposes, we resampled them to 800×600 pixels using Corel PhotoPaint 9 (with the antialias option). All images were embedded with 0%, 10%, 20%, 40%, 60%, 80%, and 100% messages and processed using RS steganalysis (100% corresponds to one bit per pixel for 8-bit images and one bit per channel for 24-bit images). The results are shown in Figs. 8 (grayscales) and 10 (24-bit color). Figures 9 and 11 show the distribution of detected message length around the true message length (assuming the distribution is Gaussian).

Simple visual analysis of Figs. 8–11 tells us that the RS steganalysis estimates the secret message length extremely accurately and reliably. For grayscale images we can easily separate cover images from stego images with a 10% embedded message. The color case is equally good with a few outliers with slightly larger initial bias than the majority of images.

The RS steganalysis is applicable to most commercial steganographic software products. Examples of vulnerable programs include, for example, Steganos, Windstorm, S-Tools, and Hide4PGP (Steganography software for Windows 2002).

6 Palette images

In this section, we apply the principles outlined in Sect. 3 to one general embedding paradigm for palette images – LSB embedding in randomly selected indices to an ordered palette. The EzStego algorithm [15], Steganos, and Hide&Seek (Steganography software for Windows 2002) are examples of this embedding archetype. EzStego first orders the palette to minimize color differences between consecutive colors by finding an approximate solution to the traveling salesman problem. Then the message bits are embedded as the LSBs of color indices to the sorted palette. The original EzStego algorithm embeds bits sequentially, but in our version we proceed along a pseudorandom key-dependent walk to make the detection harder. For a more detailed description of the algorithm, see [15].

Because EzStego preserves the original palette order, one needs to inspect the image data to mount an attack. Before we describe our approach, we briefly cite previous attacks applicable to this embedding algorithm. Westfeld [23] generalized his chi-square attack [21] for bit-exchanging algorithms with random straddling and reports the ability to detect messages as small as 33% of the total image capacity. Provos and Honeyman [17] introduced a different generalized version of the chi-square attack. While this method proves quite powerful for LSB embedding in quantized JPEG coefficients, according to our experiments, this attack does not give as reliable results for palette images. Finally, Farid and Siwei [8] has also reported a successful detection for EzStego. For a false detection rate less than 1%, he detects about 45% of all stego images with a close to 100% message length embedded. With shorter messages, the detection rate quickly falls to zero. Given the fact that this algorithm is blind to the stego method, this is a remarkable result. However, targeted approaches such as the one reported in this section provide significantly higher detection rate while giving an accurate estimate of the secret message length. Before we describe our approach, we give a brief description of the EzStego algorithm and introduce the terminology.

6.1 EzStego algorithm

EzStego first sorts the palette colors $c_0, c_1, \ldots, c_{P-1}$, $P \leq 256$ in a cycle $c_{\pi(0)}, c_{\pi(1)}, \ldots, c_{\pi(P-1)}$, $\pi(P) = \pi(0)$, so that the sum of distances $\sum_{i=0}^{P-1} \left| c_\pi(i) - c_{\pi(i+1)} \right|$ is small. In the last expression, π is the sorting permutation. The set of pairs whose colors will be exchanged for each other during embedding is

$$E = \{(c_{\pi(0)}, c_{\pi(1)}), (c_{\pi(2)}, c_{\pi(3)}), \ldots, (c_{\pi(P-2)}, c_{\pi(P-1)})\} \tag{12}$$

Using the stego key, generate a pseudorandom walk through image pixels. For each pixel along the walk, replace its color $c_{\pi(k)}$ with the color $c_{\pi(j)}$, where j is the index k with its LSB replaced by b, where b is the message bit: $\text{LSB}(j) = b$. Repeat the embedding steps until all message bits are embedded or the end of the image file is reached.

The message extraction algorithm first determines the same palette ordering π and then generates the pseudorandom walk from the stego key. The message bits are read out from LSBs of indices to the sorted palette, $b = \text{LSB}(k)$, where $c_{\pi(k)}$ is the pixel color visited along the random walk.

6.2 Distinguishing statistics

Before we define the distinguishing statistics and estimate its parameters, we analyze the impact of EzStego embedding on the cover image.

Let (c_1, c_2) be a color pair from E. We extract the colors c_1, $c_2$from the whole image, for example, by scanning it by rows or columns. This sequence of colors is then converted to a binary vector by associating c_1 with a "0" and c_2 with a "1." We denote this vector $Z(c_1, c_2)$ and call it the color cut for the pair (c_1, c_2). In palette images, Z will show considerable structure because palette images have a small number of colors. The embedding process will disturb this structure and increase the entropy of Z. Finally, when the maximal length message has been embedded in the cover image (one bit per pixel), the entropy of Z will be maximal, corresponding to a random binary sequence. Now we will look at what happens during embedding to color cuts for "shifted" color pairs from the set E':

$$E' = \{(c_{\pi(1)}, c_{\pi(2)}), (c_{\pi(3)}, c_{\pi(4)}), \ldots, (c_{\pi(P-1)}, c_{\pi(0)})\} \tag{13}$$

Let us look at one fixed color pair $(c_{\pi(2k-1)}, c_{\pi(2k)})$ from E'. During embedding, the colors $c_{\pi(2k-1)}$ and $c_{\pi(2k-2)}$ are exchanged for each other and so are the colors $c_{\pi(2k)}$ and $c_{\pi(2k+1)}$. Even after embedding the maximal message (each pixel modified with probability $1/2$), the color cut $Z(c_{\pi(2k-1)}, c_{\pi(2k)})$ will still show some residual structure. To see this, imagine a binary sequence W formed from the cover image by scanning it by rows and associating a "0" with the colors $c_{\pi(2k-1)}$ and $c_{\pi(2k-2)}$ and a "1" with the colors $c_{\pi(2k)}$ and $c_{\pi(2k+1)}$. Convince yourself that the color cut $Z(c_{\pi(2k-1)}, c_{\pi(2k)})$ after embedding a maximal pseudorandom message in the image is the same as starting with the sequence W and skipping each element of W with probability $1/2$. Now, because W showed structure in the cover image, most likely long runs of 0s and 1s, we see that randomly chosen subsequences of W will show some residual structure as well.

Having presented our arguments in the paragraph above, we describe our new detection method. We concatenate color cuts for all pairs in E into one vector $Z = Z(c_{\pi(0)}, c_{\pi(1)})$ & $Z(c_{\pi(2)}, c_{\pi(3)})$ & …& $Z(c_{\pi(P-2)}, c_{\pi(P-1)})$ and then all color cuts for shifted pairs E' into the vector Z' = $Z(c_{\pi(1)}, c_{\pi(2)})$ & $Z(c_{\pi(3)}, c_{\pi(4)})$ & …& $Z(c_{\pi(P-1)}, c_{\pi(0)})$. Next we define the distinguishing statistics that will be used to measure the structure in the bit streams Z and Z'. Let $E_2(Z)$

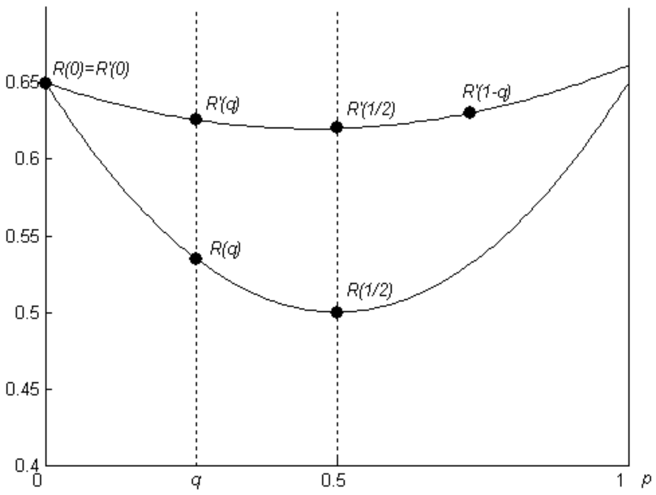

Fig. 12. Typical pairs diagram showing the relative number of homogenous bit pairs in Z and Z' as functions of the number of pixels p with flipped color indices

be the second-order entropy

$$E_2(Z) = \sum_{i=1}^{4} -p_i \log p_i$$

where p_i are the probabilities of occurrence of the bit pairs "00," "01," "10," and "11" in Z. To simplify the calculations, instead of using E_2 directly, we simply count the number of "homogenous" bit pairs ("00," "11") in Z. Let $R(p)$ denote the number of homogeneous bit pairs in Z after flipping the LSBs of indices of $100p$ percent of randomly chosen pixels, $0 \leq p \leq 1$, divided by n – the length of Z. Similarly, let $R'(p)$ be the relative number of homogeneous bit pairs in Z'. For $p < 1/2$, this number of modifications corresponds to embedding a message of length $2pMN$ bits ($2p$ bits per pixel), where M and N are image dimensions.

6.3 Parameter estimation

In Theorem 1, we prove that $R(p)$ is a parabola with its vertex at $p = 1/2$ and $R(1/2) = 1/2$ (Fig. 12). Because we can calculate $R(q)$ from the stego image with an unknown message length q, the points $(q, R(q))$ and (1/2, $R(1/2)$) uniquely determine $R(q)$. Note that $R(q) = R(1-q)$.

It appears that $R'(p)$ is well modeled using a parabola as well, although we have no formal proof of this statement. The value of $R'(1/2)$ can be derived from Z' (see Theorem 2), while the values $R'(q)$ and $R'(1-q)$ can be calculated from the stego image and the stego image with all colors flipped, respectively. Thus, we can fit a second-degree polynomial through the points $(q, R'(q))$, (1/2, $R'(1/2)$), and $(q, R'(1--q))$ to obtain $R'(p)$.

Finally, we accept one additional assumption $R(0) = R'(0)$, which says that the number of homogenous pairs in Z and Z' must be the same if no message has been embedded. This is indeed intuitive because there is no reason why the color cuts of pairs in E and E' should have different structures.

Figure 12 shows $R(p)$ and $R'(p)$ as functions of p – the pairs diagram – for a typical test image. After denoting $D(p) = R(p) - R'(p) = ap^2 + bp + c$, with a, b, and c yet undetermined constants, we can write $D(0) = c = 0$ and $D(1/2) = R(1/2) - R'(1/2) = a/4 + b/2$. Also, $D(q) = aq^2 + bq$ and $D(1-q) = a(1-q)^2 + b(1-q)$. Eliminating the unknown parameters a and b leads after simple algebra to a quadratic equation for q:

$$4D(1/2)q^2 + [D(1-q) - D(q) - 4D(1/2)]q + D(q) = 0 \tag{14}$$

The coefficients of this quadratic equation are known, so we can solve it for the unknown q. The root that is smaller is our approximation to the unknown message length q.

Theorem 1. *The expected value of $R(p)$, $p \in [0,1]$ is a parabola with its minimum at 0.5. In particular, $R(1/2) = (n-1)/2n \approx 1/2$ for large n.*

Proof: We repeat that $R(p)$ is the relative number of homogenous pairs in Z after embedding a message of relative length p. We can write Z as a concatenation of binary segments consisting of consecutive 0s or 1s with lengths of the segments $k_1, k_2, \ldots, k_r, k_1 + k_2 + \ldots + k_r = n, k_i > 0$, where n is the length of Z. For example, for Z = 001110110..., we have $k_1 = 2, k_2 = 3, k_3 = 1, \ldots$. For $p = 0$, we have $nR(0) = \sum_{i=1}^{r}(k_i - 1) = n - r$. After embedding a message of relative length p, in a segment consisting of k bits, the probability that a given pair of consecutive bits will be homogenous is $p^2 + (1-p)^2$ (both are changed or neither is changed). Because we have $k-1$ consecutive pairs, the expected number of homogenous pairs is $[p^2 + (1-p)^2](k-1) + 2p(1-p)$, where the last term comes from the right end of the segment (an additional pair will be formed at the boundary if the last bit in the segment flips and the first bit of the next segment does not flip, or vice versa). Thus, the total number of homogenous pairs is a sum over all segments except for the last one, which lacks the boundary term $2p(1-p)$:

$$\begin{aligned} nR(p) =& \sum_{i=1}^{r}[(k_i - 1)(p^2 + (1-p)^2) \\ &+ 2p(1-p)] - 2p(1-p) \\ =& 2p^2(n-2r) - 2p(n-2r) + n - r - 2p(1-p) \end{aligned}$$

We see that $R(p)$ is a parabola in p with its vertex at $p = 1/2$, $R(0) = R(1) = (n-r)/n$, and $R(1/2) = (n-1)/2$, which concludes the proof. □

Theorem 2. *Let $Z' = \{b_i\}_{i=1}^{n}$ be the binary vector defined in the text above. The expected value of $R'(1/2)$ is $\sum_{k=1}^{n-1} 2^{-k} h_k$, where h_k is the number of homogeneous pairs in the sequence of pairs $b_1b_{1+k}, b_2b_{2+k}, b_3b_{3+k}, \ldots, b_{n-k}b_n$.*

Proof: Let $W = W_1 \& W_2 \& \ldots \& W_{P/2}$ be the concatenation of binary sequences W_j formed from the stego image by scanning it by rows and associating a "0" with the colors $c_{\pi(2j-1)}$ and $c_{\pi(2j-2)}$ and a "1" with the colors $c_{\pi(2j)}$ and $c_{\pi(2j+1)}$. The color cut $Z'(c_{\pi(2j-1)}, c_{\pi(2j)})$ after embedding a maximal message in the cover image is the same as starting with the sequence W_j and skipping each element of W with probability $1/2$. Imagine you are going through Z' while skipping each element with probability $1/2$. Then, the probability of skipping exactly $k-1$ elements in a row is $2^{-k}, k = 1, 2, \ldots$. Because there are h_k homogenous pairs in the sequence of pairs

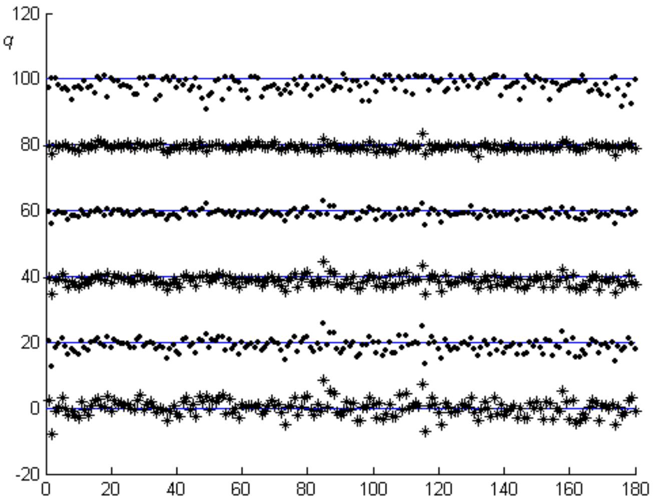

Fig. 13. Estimated message length (in percentage of maximal message length) for a database of 180 GIF images

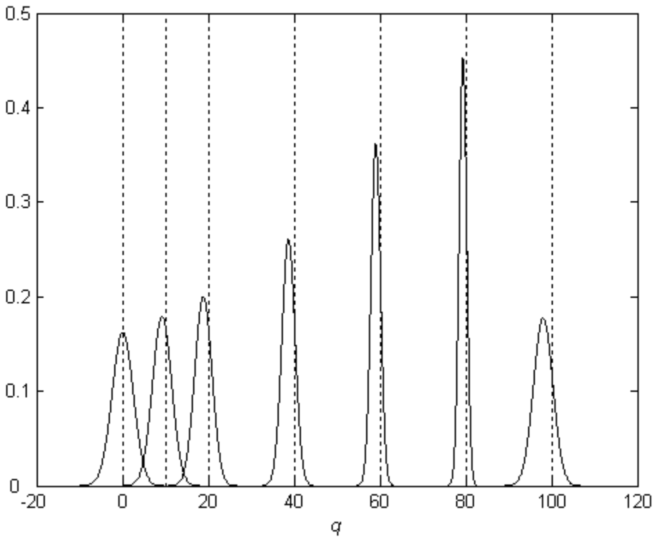

Fig. 14. Distribution of estimated message length q compared to the true message length (*vertical lines*) for the database of 180 GIF images

$b_1b_{1+k}, b_2b_{2+k}, b_3b_{3+k}, \ldots, b_{n-k}b_n$, the expected number of homogenous pairs separated by $k-1$ elements is $2^{-k}h_k$. The formula of Theorem 2 is obtained by summing these contributions from $k = 1$ to the maximal separation $k - 1 = n - 2$. □

6.4 Experimental results

We have performed tests on a database of 180 color GIF images. The images were obtained using four different digital cameras and were originally stored as high quality JPEG images. For our test purposes, we resampled them to 800×600 pixels using Corel PhotoPaint 9 (with antialias option) and converted to palette images using with the following options: optimized palette, ordered dithering. All images were embedded with $0\%, 10\%, 20\%, 40\%, 60\%, 80\%$, and 100% messages (100% message corresponding to one bit per pixel) and processed using our detection method. The results are shown in Fig. 13. Figure 14 shows the distribution of detected message length around the true message length (assuming the distribution is Gaussian).

The accuracy is mostly influenced by the "initial bias," which is the message length (in percent) detected in the cover image. The fact that this initial bias may not be exactly zero reflects the fact that the assumption $R(0) = R'(0)$ in Sect. 6.3 is an approximate experimental fact.

Images with low numbers of unique colors, such as cartoons and computer-generated images, may have some singular structure that could occasionally introduce a large error into our detection. For example, some dithering patterns between two colors (e.g., in a uniform background) may be misinterpreted as a false message. A histogram with a very high number of pixels for one or two colors next to each other may also in some exceptional cases negatively influence the detection accuracy. Fortunately, the method as described in this section can be adjusted to eliminate such outliers. More details are given in [10].

7 Conclusions

In this paper, we describe a new paradigm for steganalysis. The methods developed under this paradigm can estimate the length of the secret embedded message with very good accuracy. 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 depends on 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. Having achieved this, we can calculate the unknown message length q by solving the equation $S(q) = S_{stego}$, where S_{stego} is the value of S measured for the stego image. The quantity S is called the distinguishing statistics.

We show what quantities S are useful for JPEG images, palette images, and uncompressed raw formats. In particular, we apply the detection methodology to F5, OutGuess, EzStego with random straddling, and generic LSB embedding with random straddling. The performance is always interpreted and evaluated on a test database of images.

In the future, we intend to use the proposed detection paradigm for determination of upper bounds on steganographic capacity. As elaborated on in Sect. 2, steganographic capacity is the maximum 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.

One of our future goals is to estimate the relative steganographic capacity C_R that would be *independent* of the cover image. On the one hand, this is a formidable task because "natural" images exhibit such a wide spectrum of statistical properties that developing a mathematical model becomes impossible. On the other hand, it seems plausible to estimate C_R for a given image source Ω, such as images taken with a specific digital camera under specified conditions. In this case, the relative steganographic capacity C_R is the function of the embedding method Σ and the image source Ω: $C_R = C_R(\Omega, \Sigma)$. To derive an estimate for the relative steganographic capacity C_R, we use a large test database of images from the source Ω

and apply the proposed detection methods to all cover images from this database, 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 test [5]. The smallest value of q that does not pass this test determines an estimate for the *upper bound* for steganographic capacity for the image source Ω. If the database is large and contains a wide diversity of images, the estimate for the steganographic capacity will likely be valid for other images from Ω 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 a 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. This paper was supported by Air Force Research Laboratory, Air Force Material Command, USAF, under 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 therein. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either express or implied, of Air Force Research Laboratory or the U.S. government.

References

1. Anderson RJ, Petitcolas FAP (1998) On the limits of steganography. IEEE J Selected Areas Commun (Special issue on copyright and privacy protection) 16(4):474–481
2. Cachin C (1998) An information-theoretic model for steganography. Lecture notes on computer science, vol 1525. Springer, Berlin Heidelberg New York, pp 306–318
3. Chandramouli R, Memon N (2000) A distributed detection framework for watermark analysis. Proceedings of the ACM Multimedia workshop on multimedia and security, Los Angeles, 4 November 2000
4. Chandramouli R, Memon N (2001) Analysis of LSB based image steganography techniques. Proceedings of ICIP 2001, Thessaloniki, Greece, 7–10 October 2001
5. D'Agostino R, Stephens M (1986) Goodness-of-fit techniques. Marcel Dekker, New York
6. Dumitrescu S, Wu Xiaolin, Memon N (2002) On steganalysis of random embedding in continuous-tone images. Proceedings of ICIP 2002, Rochester, NY, 22–25 September 2002
7. Dumitrescu S, Wu Xiaolin, Wang Z (2003) Detection of LSB steganography via sample pair analysis. Lecture notes in computer science, vol 2578, Springer, Berlin Heidelberg New York, pp 355–372
8. Farid H, Siwei L (2003) Detecting hidden messages using higher-order statistics and support vector machines. Lecture notes in computer science, vol 2578. Springer, Berlin Heidelberg New York, pp 340–354
9. Fridrich J, Goljan M, Du R (2001) Steganalysis based on JPEG compatibility. Proceedings of SPIE: multimedia systems and applications IV, Denver, pp 275–280
10. Fridrich J, Goljan M, Soukal D (2003a) Higher-order statistical steganalysis of palette images. Proceedings of SPIE: Electronic Imaging 2003, security and watermarking of multimedia contents, Santa Clara, CA, 21–25 January 2003
11. Fridrich J, Goljan M, Hogea D (2003b) Steganalysis of JPEG images: breaking the F5 algorithm. Lecture notes in computer science, vol 2578. Springer, Berlin Heidelberg New York, pp 310–323
12. Goljan M, Fridrich J, Du R (2001) Distortion-free data embedding. Lecture notes in computer science, vol 2137. Springer, Berlin Heidelberg New York
13. Katzenbeisser S, Petitcolas FAP (2000) Information Hiding techniques for steganography and digital watermarking. Artech House, Boston
14. Katzenbeisser S, Petitcolas FAP (2002) On defining security in steganographic systems. Proceedings of electronic imaging: security and watermarking of multimedia content, SPIE 02, San Jose, 21–24 January 2002, pp 50–56
15. Machado R (2001) EzStego 2.0b4, http://www.stego.com
16. Provos N (2001) Defending against statistical steganalysis, Proceedings of the 10th USENIX security symposium, Washington, DC
17. Provos N, Honeyman P (2001) Detecting steganographic content on the internet, CITI Technical Report 01-11
18. Simmons GJ (1984) Prisoners' problem and the subliminal channel. In: Chaum D (ed) Advances in cryptology. Proceedings of CRYPTO '83, Santa Barbara, CA. Plenum Press, New York, pp 51–67
19. Steganography software for Windows (2002) http://members.tripod.com/steganography/stego/software.html
20. Wayner P (2002) Disappearing cryptography, 2nd edn. Morgan Kaufmann, San Francisco
21. Westfeld A, Pfitzmann A (1999) Attacks on steganographic systems. Lecture notes in computer science, vol 1768. Springer, Berlin Heidelberg New York, pp 61–75
22. Westfeld A (2001) High capacity despite better steganalysis (F5–a steganographic algorithm). Lecture notes in computer science, vol 2137. Springer, Berlin Heidelberg New York, pp 289–302
23. Westfeld A (2003) Detecting low embedding rates. Lecture notes in computer science, vol 2578. Springer, Berlin Heidelberg New York, pp 324–339