A New Steganographic Method for Palette-Based Images

Jiri Fridrich
Center for Intelligent Systems, SUNY Binghamton, Binghamton, New York

Abstract

In this paper, we present a new steganographic technique for embedding messages in palette-based images, such as GIF files. The new technique embeds one message bit into one pixel (its pointer to the palette). The pixels for message embedding are chosen randomly using a pseudo-random number generator seeded with a secret key. For each pixel at which one message bit is to be embedded, the palette is searched for closest colors. The closest color with the same parity as the message bit is then used instead of the original color. This has the advantage that both the overall change due to message embedding and the maximal change in colors of pixels is smaller than in methods that perturb the least significant bit of indices to a luminance-sorted palette, such as EZ Stego.[1] Indeed, numerical experiments indicate that the new technique introduces approximately four times less distortion to the carrier image than EZ Stego. The maximal color change is 4–5 times smaller for the new technique than that of EZ Stego. A technique that introduces less distortion to the carrier image will generally cause changes that are more difficult to detect, and will therefore provide more security.

Introduction

The techniques for secret hiding of messages in an otherwise innocent looking carrier message belong to the field of steganography. The purpose of steganography is to conceal the very presence of secret information. To make the communication more secure, the secret information can be compressed and encrypted before it is hidden in the carrier. This is important because in this way we minimize the amount of information that is to be sent, and it is also easier to hide a random looking message into the carrier than to hide a message with a high degree of regularity. Encrypting the compressed message before hiding is recommended and provides double protection.

The field of steganography is very old. The most popular steganographic methods used by spies include invisible ink and microdots.[2] Microdots are blocks of text or images scaled down to the size of a regular dot in a text. Shifting words and changing spacing between lines in text documents or in images containing text can also be used for hiding bits of secret messages. Today, it seems natural to use digital images, digital video, or audio for hiding secret messages. The gaps in human visual and audio systems can be used for information hiding. In the case of images, the human eye is relatively insensitive to high frequencies. This fact has been utilized in many steganographic algorithms, which modify the least significant bits of gray levels in digital images or digital sound tracks. Additional bits of information can also be inserted into coefficients of image transforms, such as discrete cosine transform, Fourier transform, etc. Transform techniques are typically more robust with respect to common image processing operations and lossy compression.

The steganographer's job is to make the secretly hidden information difficult to detect given the complete knowledge of the algorithm used to embed the information except the secret embedding key.* This so called Kerckhoff's principle is the golden rule of cryptography and is often accepted for steganography as well. In steganography, the source from which the carrier images are drawn is of paramount importance. The image source is essentially a part of the algorithm, and therefore, according to the Kerckhoff's principle, it should be assumed to be public knowledge. In real world, however, it may be rather difficult if not impossible to obtain this knowledge. In one typical scenario, Alice and Bob exchange messages (images) and Eve wants to find out if they use steganography. Alice and Bob draw images from a certain source. Eve may find out, for example, that Alice uses a specific scanner, digital camera, a camcorder with a TV card, or some other imaging device as a source of images. Eve could purchase the same type of imaging device and generate a large database of images. From that database, she can derive a set of statistical measures satisfied by all images. Based on the steganographic algorithm, Eve may be able to derive some statistical fingerprints caused by the presence of secret messages. When Alice sends an image to Bob, Eve can check the consistency of that image with her statistical evidence. Eve can make two types of error: (I) false alarm (detecting steganography when no secret message is present in the image) and (II) missed detection (not detecting image with secret message). Eve can optimize the decision threshold used for consistency checking that would minimize both errors. There is a possibility that Eve overdoes her job and builds an overly detailed noise model of the imaging device. Then, she would actually detect

* For theoretical aspects of steganography, see [3].

differences between the two pieces of hardware rather than secret messages.

In another scenario, Eve does not know the image source, and her task is much more difficult. This could be the situation for a monitoring device connected to a node in a global computer network checking all images for evidence of steganography. Eve can again build a large database from images and specify a set of statistical measures satisfied by the images, but the variety of images and the fact that Eve does not necessarily know the steganographic algorithm will diminish Eve's ability to discern images with messages from those without messages.

The ability to discern images with secret messages is directly influenced by the size of the secret message and the format and content of the carrier image. Obviously, the longer the message, the larger the modification of the carrier image and the higher the probability that the modifications can be statistically detected. Given the complete knowledge of the algorithm including the image source, there is obviously an upper limit on the maximal length of messages that can be transmitted in a secure manner. The choice of the carrier image is also crucial. Natural photographs with 24 bits per pixel provide the best environment for message hiding. The redundancy of the data helps to conceal the presence of secret messages. Compressed images, such as JPEG files, are more sensitive to small perturbations of the image data and pose a challenge for creating a secure steganographic technique with reasonable capacity. Palette-based images, although abundant over the Internet, also provide a hostile environment for the steganographer. The limitation on the available colors imposed by the finite palette makes the process of message hiding a difficult challenge. In the next section, we discuss possible approaches, review current techniques, and describe a new method for message hiding in palette images. The new method is explained in detail and compared with EZ Stego in Section 3. Finally in Section 4, we summarize the new technique and conclude the paper by outlining further possible security improvements.

Steganography using Palette-Based Images

A large portion of images on the Internet is available in palette-based formats, such as GIF or PNG. There are two approaches to message hiding in palette-based images:

Embedding messages into the palette;
Embedding into the image data.

The advantage of the first method is that it will probably be easier to design a secure method under some assumptions about the noise properties of the image source (a scanner, a CCD camera, etc.). The obvious disadvantage is that the capacity does not depend on the image and is limited by the palette size.

Methods from the second group have higher capacity, but it is generally harder to design a secure scheme.

In order to prove security of an embedding scheme, we need to understand the details of algorithms for creating palette-based images. Virtually all algorithms consist of two steps: color quantization (also called vector quantization) and dithering. Color quantization selects the palette of the image by truncating all colors of the original raw, 24-bit image to a finite number of colors (256 for GIF images, and 216 for Netscape version of GIFs, 2 for black and white images, etc.). Dithering is used for apparent increasing of color depth. It uses the integrating properties of the human visual system and creates the illusion of additional colors by trading space resolution for color depth. The best results are obtained using dithering algorithms based on error diffusion.

There are several algorithms for color quantization. The two most frequently used are based on iterative dividing of the three-dimensional color cube into two boxes with approximately the same number of colors. The half with the largest dimensions is selected for the next iteration till the desired number of boxes (colors) is obtained. The centers of gravity of each box are then rounded to integer colors representing the colors of the palette. If the largest dimension is replaced by the largest standard deviation, another, slightly better algorithm is obtained.

The most common algorithm for dithering is based on error diffusion. The image is scanned in some regular manner, for example by rows, columns, or diagonally. As the color in one pixel is rounded to its closest color in the palette, an error is produced (it is negative if the rounding decreases the pixel value and it is positive otherwise). The error is multiplied by weights and added to surrounding pixels that have not yet been visited. This way, the rounding error is spread to neighboring pixels, and a visually pleasing image free of contouring artifacts is obtained.

Principles of Steganographic Methods

It has been suggested in the past that secure message hiding in palette-based images can be obtained by permuting the image palette rather than changing the colors in the image.[4] While this method does not change the appearance of the image, which is certainly an advantage, its security is questionable because many image processing software products order the palette according to luminance, frequency of occurrence, or some other scalar factor. A randomized palette will raise suspicion. Also, displaying the image and resaving it may erase the information because the software routine may rewrite (and reorder) the palette. Another disadvantage is a rather limited capacity.

A better approach may be to hide encrypted (random) messages in the least significant bits of the palette colors. One would need to guarantee that the perturbed palette is still consistent with the noise model of the original 24-bit image. This, however, could be established in each particular case by studying the sensitivity of the color quantization process to perturbations. A possible difficulty is the recovery of the message from the least significant bit (LSB) of the palette entries even after the palette has been reordered. The problem with this algorithm is that the order of the palette will change after embedding. It is therefore not simple for the decoder to synchronize with the stego-image after the message has been encoded even if the image palette has not been reordered. To overcome this difficulty, we may have to analyze the palette before message

embedding can begin to find out which palette entries may be safely changed without disturbing the palette order. For example, we may attempt to embed just a single bit into each palette entry (a triple of R, G, and B) as the parity bit of that color. Obviously, this will further decrease the already limited capacity of palette embedding techniques to one third.

Practical methods should have capacity proportional to the image size, or the number of pixels. Many currently available software tools first decrease the color depth of the GIF image to 128, 64, or 32. This way, when the LSBs (two LSBs, or three LSBs) of the colors are perturbed, the total number of newly created colors will be at most 256. Thus, it will be possible to embed one, two, or three bits per pixel without introducing visible artifacts into the carrier image. However, as pointed out by Johnson,[5,6] the new palettes will have easily detectable groups of similar colors. It is thus relatively easy to distinguish images with and without secret messages.

One of the most popular message hiding schemes for palette-based images (GIF files) has been proposed by Machado.[1] In her method called EZ Stego, the palette is first sorted by luminance. In the reordered palette, neighboring palette entries are typically near to each other in the color space, as well. EZ Stego embeds the message in a binary form into the LSB of indices (pixels) pointing to the palette colors. Here are the steps:

1. Find the index of the pixel's RGB color in the sorted palette.
2. Get one bit from the binary message and replace the LSB of the index.
3. Find the new RGB color that the index now points to in the sorted palette.
4. Find the index of the new RGB color in the original palette.
5. Change the pixel to the index of the new RGB color.

Message recovery is simply achieved by collecting the LSBs of all indices in the image file. Of course, the method could be improved by injecting message bits into randomly selected pixels based on a pseudo-random number generator (PRNG) seeded with a secret key.

The algorithm is based on the premise that close colors in the luminance-ordered palette are close in the color space. However, since luminance is a linear combination of three colors *R*, *G*, and *B*, occasionally colors with similar luminance values may be relatively far from each other (e.g., colors [6,98,233] and [233,6,98] have the same luminance but represent two completely different colors). To avoid this problem, we propose to hide message bits into the parity bit of close colors. For the color of each pixel, into which we embed message bits, we search the closest colors in the palette till we find a palette entry with the desired parity bit (parity bit of the color *R*, *G*, *B* is *R*+*G*+*B* mod 2). Since the parity bits of palette entries corresponding to real images are more or less randomly distributed, this will guarantee that we will never have to depart from the original color too much. This way, we avoid the problem of occasionally making large changes in color, which will certainly contribute to the undetectability of the message. In this paper, we compare the performance of EZ Stego and the new technique using two parameters: (1) the RMS distance between the original image and the stego-image, and (2) the maximal change over pixel colors. Both measures indicate that the new technique produces significantly better results.

New Steganographic Algorithm

The secret message *m* is first converted into a binary stream of length *M*. Then, a user-defined seed is used to randomly select *M* pixels in the image.

For each pixel, the set of the closest colors (in Euclidean norm) is calculated (this is done by calculating the distance between the color of the pixel from each palette entry and sorting the result). The distance between colors $(R_1G_1B_1)$ and $(R_2G_2B_2)$ is

$$\sqrt{(R_1-R_2)^2+(G_1-G_2)^2+(B_1-B_2)^2}$$

Starting with the closest color (note that the closest color is the one corresponding to the same pixel), we proceed to the next closest color till we find a match between the bit to be encoded and the parity of the color. The parity of a color is defined as *R*+*G*+*B* mod 2.

Once the color is found, the index for the pixel is changed to point to the new color. This way, we guarantee that we never replace a pixel color by a completely different color, which could occasionally happen in EZ Stego because ordering of the palette by luminance may introduce discontinuities in neighboring colors.

To extract the secret message, *M* pixels are selected using a PRNG seeded with user-defined seed. The secret message is simply read by extracting the parity bits of the colors of selected pixels.

Limitations:

The only limitation on message length is that message length in bits should be smaller than or equal to the number of pixels in the image. However, embedding a large message comparable to the image size increases the likelihood of making detectable changes inconsistent with the dithering algorithm. Detailed analysis of detectability of hidden messages as a function of message length will be part of a future research.

On average, the amount by which the image is modified is smaller than with EZ Stego. This statement has been quantified using a numerical simulation on two images. We used two test mages in our study. The image "Fox" with 240×320 pixels has been truncated to 256 colors using the routine rgb2ind.m in Image Processing Toolbox in Matlab. The image and its luminance-ordered palette are shown in Figure 1 and 2. The second test image, "Mandrill", with 512×512 pixels truncated to 256 colors is shown in Figure 3. Its luminance-ordered palette is shown in Figure 4. We can clearly see that the luminance-ordered palette of "Fox" contains visibly fewer and less severe discontinuities in color than that of "Mandrill".

Figure 1 Test image "Fox" (320 × 240).

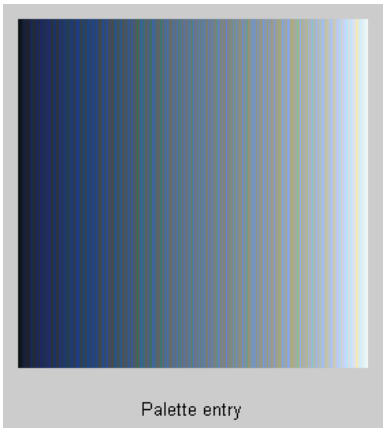

Figure 2 Luminance-ordered palette.

Figure 3 Test image "Mandrill" (512 × 512).

Figures 5–6 illustrate the superior performance of the new technique. Figure 5 is the difference between the stego-image and the original carrier image measured as the Euclidean distance between two vectors (matrices). The independent variable is the length of the embedded secret message in bits. The results for both test images are in the same diagram. The distance between the original and stego images is more than four times smaller for the new method. A method that introduces less distortion into an image stands a better chance of going through statistical tests than a method that introduces larger distortion. Figure 6 shows the maximal change in color (again, Euclidean distance between colors was used) for both methods and both test images.

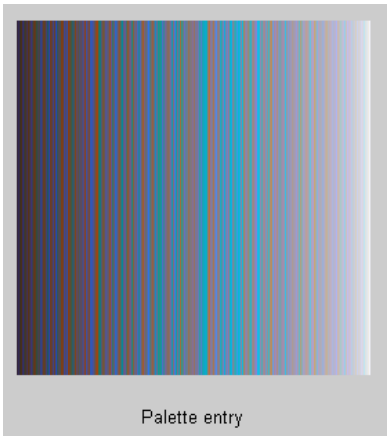

Figure 4 Luminance-ordered palette.

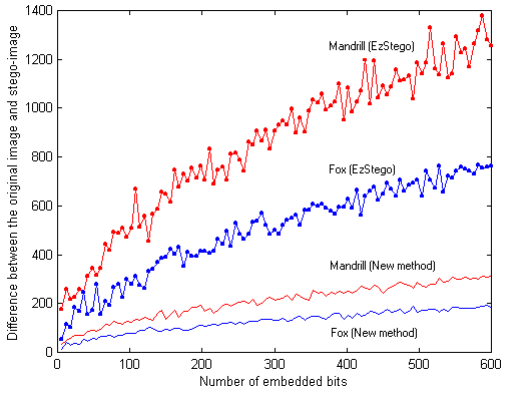

Figure 5 Comparison of distortion introduced by EZ Stego and the new technique.

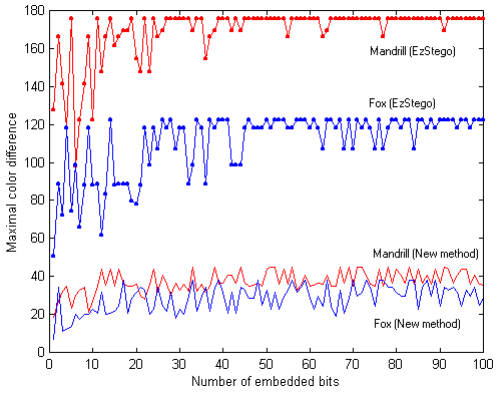

Figure 6 Maximal color change caused by EZ Stego and the new technique.

Clearly, the new method introduces far less severe changes into the pixel colors than EZ Stego. The maximal

color change is about 40 or less for the new method, while EZ Stego can modify the color by a large amount (almost 180 for "Mandrill" and 120 for "Fox"). Again, the maximal color change is at least 4 times smaller for the new method when compared to EZ Stego. This confirms our heuristic argument presented in Section 2. Both techniques introduce more distortion for the test image "Mandrill" than "Fox" because Mandrill's palette exhibits less regularity than that of "Fox". This can clearly be seen in the luminance-ordered palette of Figure 4, which contains numerous abrupt changes in colors even though the luminance changes smoothly. This property of Mandrill's palette is due to rich, detailed texture present in the image.

Summary and Future Research

In this paper, we have introduced a new steganographic technique for embedding messages in palette-based images, supported for example by GIF or PNG image formats. The new technique embeds one message bit into one pixel (its pointer to the palette). The pixels for message embedding are chosen randomly using a PRNG seeded with a secret key. For each pixel at which one message bit is to be embedded, the palette is searched for closest colors. The closest color with the same parity is then used instead of the original color. This has the advantage that both the overall change and the maximal change in colors of pixels is smaller than in methods that perturb the LSB of indices to a luminance-sorted palette, such as EZ Stego. Indeed, numerical experiments done with two test images indicate that the new technique causes approximately four times less distortion to the carrier image than EZ Stego. The maximal color change is 4–5 times smaller for the new technique than that of EZ Stego. A technique that introduces less distortion to the carrier image will generally cause changes that are more difficult to detect, and will therefore provide more security. Because the new technique does not change the image palette, the only artifacts due to embedding of large messages that may be possibly introduced are local inconsistencies with error diffusion algorithms. Detailed investigation of this issue will be part of our future effort.

The security of the new scheme can be further improved by a more careful selection of the pixels that carry the secret information. Each pixel can be assigned a weight between zero and one according to how easily these pixels can be modified for message bit embedding. For example, pixels with close colors of different parities will be assigned values close to one, while pixels isolated in the color space will be assigned values close to zero. Instead of randomly choosing the pixels that will carry the secret message, one can select the pixels with probabilities proportional to their weights. Additional factors, such as a local variance in pixel neighborhood can be introduced to avoid making changes to pixels in areas of uniform color. These issues will be part of the future research.

Acknowlegements

The work on this paper was supported by Air Force Research Laboratory, Air Force Material Command, USAF, under a grant number F30602-98-C-0009. 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. Machado, EZ Stego, [http://www.stego.com]
2. D. Kahn, "The history of steganography", 1st *Information Hiding Workshop, Lecture Notes in Computer Science*, R. Anderson, ed., vol. 1174, pp. 1–5, Springer-Verlag, 1996.
3. R. J. Anderson and F. A. P. Petitcolas, "On the limits of steganography", *IEEE Journal of Selected Areas in Communications (J-SAC) – Special Issue on Copyright & Privacy Protection*, **16**(4), May 1998. [http://www.cl.cam.ac.uk/~fapp2/steganography/].
4. Matthew Kwan, mkwan@darkside.com.au, [http: //www.darkside.com.au/gifshuffle/]
5. N. Johnson and S. Jajodia, "Steganalysis of Images Created Using Current Steganography Software", *Proc. of the 2nd Information Hiding Workshop*, Portland, Oregon, April 15–17, 1998.
6. N. Johnson and S. Jajodia, "Steganalysis: The Investigation of Hidden Information", *Proc. of the 1998 IEEE Information Technology Conference*, Syracuse, New York, September 1–3, 1998.