Steganography System to Hide a Sound File in a Color Image

Hameed Abdul-Kareem Younis  Aliaa’ Jaber Jalil  Zainab Ali Abbood  
(hameedalkinani2004@yahoo.com  sadnj83@yahoo.com  zainab.maki@yahoo.com)  

Dept. of Computer Science - College of Science - University of Basrah

Abstract:  
Data hiding represents a class of process used to embed data into various forms of media, such as image, audio, or text. The proposed voice in image steganography system is new approach used to embed audio into color image. Instead of using the least-significant-bit-plane (LSB 1) of the cover for embedding the voice, third-least-significant-bit-plane (LSB 3) has been used to increase the robustness. First, and second-least-significant-bit-plane (LSB 1 and LSB 2) may be modified according to bits of the voice, to minimize the difference between the cover and the stego-cover. For more protection to the voice signals, a stego-key has been used to permute the voice signals before embedding it. Experimental results of the modified method shows that Peak-Signal-to-Noise-Ratio (PSNR) is greater than the conventional method of LSBs replacement. Thus, the proposed system gives good results and can be applied in several cases in life.  
Keywords: Steganography, cover-image, LSBs, PSNR, voice signal.
1. Introduction:

1.1 Motivation

In conventional cryptography, even if the information contents are protected by encryption, the existence of encrypted communications is known. In view of this, digital steganography provides an alternative approach in which it conceals even the evidence of encrypted messaging. Generally, steganography is defined as the art and science of communicating in a covert fashion [1]. It utilizes the typical digital media such as text, image, audio, video, and multimedia as a carrier (called a host signal) for hiding private information in such a way that the third parties (unauthorized person) cannot detect or even notice the presence of the communication. In this way, steganography allows for authentication, copyright protection, and embedding of messages in the image or in transmission of the image [1, 2].

A typical digital steganographic encoder is shown in Figure (1). The message is the data that the sender wishes to remain confidential and can be text, images, audio, video, or any other data that can be represented by a stream of bits. The cover or host is the medium in which the message is embedded and serves to hide the presence of the message. This is also referred to as the message wrapper. The message embedding technique is strongly dependent on the structure of the cover, and in this paper covers are restricted to being digital images. It is not required that the cover and the message have homogeneous structure. The image with the secretly embedded message produced by the encoder is the stego-image. The stego image should resemble the cover image under casual inspection and analysis. In addition, the encoder usually employs a stego-key which ensures that only recipients who know the corresponding decoding key will be able to extract the message from a stego-image.

Recovering the message from a stego-image requires the stego-image itself and a corresponding decoding key if a stego-key was used during the encoding process. The original cover image may or may not be required; in most applications it is desirable that the cover image not be needed to extract the message. It requires the cryptographic decoding key to decipher the encrypted message.
1.2 Applications

There are many applications for digital steganography of images, including copyright protection, feature tagging, and secret communications [3, 4]:

**Copyright Protection:** A secret copyright notice or watermark can be embedded inside an image to identify it as intellectual property. This is the watermarking scenario where the message is the watermark [5, 6]. The “watermark” can be a relatively complicated structure. In addition, when an image is sold or distributed an identification of the recipient and time stamp can be embedded to identify potential pirates. A watermark can also serve to detect whether the image has been subsequently modified [7]. Detection of an embedded watermark is performed by a statistical, correlation, or similarity test, or by measuring other quantity characteristic to the watermark in a stego-image. The insertion and analysis of watermarks to protect copyrighted material is responsible for the recent surge of interest in digital steganography and data embedding.

**Feature Tagging:** Captions, annotations, time stamps, and other descriptive elements can be embedded inside an image, such as the names of individuals in a photo or locations in a map. Copying the stego-image also copies all of the embedded features and only parties who possess the decoding stego-key will be able to extract and view the features. In an image database, keywords can be embedded to facilitate search engines. If the image is a frame of a video sequence, timing markers can be embedded in the image for synchronization with audio. The number of times an image has been viewed can be embedded for “pay-perview” applications.

**Secret Communications:** In many situations, transmitting a cryptographic message draws unwanted attention. The use of cryptographic technology may be restricted or forbidden by law. However, the use of steganography does not advertise covert communication and therefore avoids scrutiny of the sender, message, and recipient. A trade secret, blueprint, or other sensitive information can be transmitted without alerting potential attackers or eavesdroppers. It is important that the embedding occur without significant degradation or loss of perceptual quality of the cover. In a secret communications application, if an attacker notices some distortion that arouses suspicion of the presence of hidden data in a stego-image, the steganographic encoding has failed even if the attacker is unable to extract the message. Preserving perceptual transparency in an embedded watermark for copyright protection is also of paramount importance because the integrity of the original work must be maintained [7]. For applications where the perceptual transparency of embedded data is not critical, allowing more distortion in the stego-image can increase hiding capacity, robustness, or both.

**Robustness:** Robustness refers to the ability of embedded data to remain intact if the stego-image undergoes transformations, such as linear and non-linear filtering, addition of random noise, sharpening or blurring, scaling and rotations, cropping or decimation, lossy compression, and conversion from digital to analog form and then reconversion back to digital form (such as in the case when a hard copy of a stego-image is printed and then a digital image is formed by subsequently scanning the hardcopy). Robustness is critical in copyright protection watermarks because pirates will attempt to filter and destroy any watermarks embedded in images [5, 6]. Anti-watermarking software is already available on the Internet and have been shown effective in removing some watermarks [8, 9]. These techniques can also be used to destroy the message in a stego-image.

**Tamper Resistance:** Beyond robustness to destruction, tamper resistance refers to the difficulty for an attacker to alter or forge a message once it has been embedded in a stego-image, such as a pirate replacing a copyright mark with one claiming legal ownership. Applications that demand high robustness usually also demand a strong degree of tamper resistance [4]. In a copyright protection application, achieving good tamper resistance can be difficult because a copyright is effective for many years and a watermark must remain resistant to tampering even when a pirate attempts to modify it using computing technology decades in the future.
1.3 Least Significant Bit (LSB 1) Replacement

This is the simplest of the steganography methods based in the use of LSB, and therefore the most vulnerable. The embedding process consists of the sequential substitution of each Least Significant Bit (LSB 1) of the image pixel for the bit message. For its simplicity, this method can camouflage a great volume of information [10, 11, 12, 13]. This technique is quite simpleton and it presents a safety fault. It is necessary only a sequential LSB reading, starting from the first image pixel, to extract the secret message. These methods also generate a unbalanced distribution of the changed pixels, because the message is embedded at the top of the image. In the next section, a modified method will be proposed.

2. The Proposed Method (LSB 3):

In this paper, a $256 \times 256$ color image has been used as a cover. So, a message (voice) up to 65536 bits (8192 bytes) can be hidden. The message is embedded in the LSB 3 of the cover to increase the robustness of the system and protect the message against the external influences such as noise, filter, and compression … etc.

Let's have the message bits set $M=\{m_0, m_1, m_2, ..., m_{L-1}\}$, where $1 \leq L \leq 65536$, $L$ is the length of the message that is embedded, and $m_i = \{0, 1\}$, for $i = 0, ..., L-1$. Let's have the cover image $= \{\text{pixel}_0, \text{pixel}_1, ..., \text{pixel}_{65535}\}$. Suppose that LSB 3 of the cover image is $\text{LSB } 3 = \{c_0, c_1, c_2, ..., c_{65535}\}$, where $c_j = \{0, 1\}$ for each $j = 0, ..., 65535$. To protect the message, a stego-Key is used, which is employed as a seed for pseudo-random number generator (PRNG). This creates a sequence of indexes used to permute the message bits. Figure (2) shows the block diagram of message signals permutation.

![Block Diagram of Message Signals Permutation](image)

The embedding process is very easy, which only replaces the permuted voice signals of the message ($P$) by the LSB 3 set of the cover to obtain the new stego-image $Z = \{\text{newpixel}_0, \text{newpixel}_1, ..., \text{newpixel}_{65535}\}$.

To minimize the difference between the old value (pixel) in the cover and the new value (newpixel) in the stego-image, we suggest the following embedding algorithm:

2.1 Embedding Algorithm
Step 1: Extract LSB 1 set of the color cover image (red color space only), $\text{LSB } 1 = \{a_0, a_1, ..., a_{65535}\}$. //first-bit-plane
Step 2: Extract LSB 2 set of the color cover image (red color space only), LSB 2 = \{b_0, b_1, \ldots, b_{65535}\}.// second-bit-plane

Step 3: For I = 1 to L do
   If \( p_i = c_i \) Then do nothing
   Else
   \{ If \( p_i = 1 \) and \( c_i = 0 \) Then
     \{ a_i = 0; \\
     \quad b_i = 0; \\
     \}
   \}
   Else If \( p_i = 0 \) and \( c_i = 1 \) Then
   \{ a_i = 1; \\
   \quad b_i = 1; \\
   \}
   \}
   c_i = p_i;  // embed message bit in the least-third-bit of the cover

To explain the above algorithm, let's have the following pixel in the cover image, pixel = \((3)_{10} = (00000011)_{2}\). Suppose we need to embed \( p = 1 \) in the LSB 3, so the new pixel will be, newpixel = \((00000011)_{2} = (7)_{10}\). Notice that the difference is \( 7 - 3 = 4 \). In our algorithm, we will set LSB 1 and LSB 2 to 0 when \( p = 1 \) and \( c = 0 \). So newpixel = \((00000010)_{2} = (4)_{10}\). As you see the deference becomes \( 4 - 3 = 1 \). On the other hand, suppose that pixel = \((4)_{10} = (00000100)_{2}\), and \( p = 0 \). The newpixel = \((00000000)_{2} = (0)_{10}\). The difference is \( 4 - 0 = 4 \). In our algorithm, in this case, we will set LSB 1 and LSB 2 to 1, so newpixel = \((00000011)_{2} = (3)_{10}\). As you see the difference becomes \( 4 - 3 = 1 \). Thus, the difference in LSB 3 replacement less or equal one as in LSB 1 but in more robust as the following:

<table>
<thead>
<tr>
<th>Cover</th>
<th>Voice Signal (p)</th>
<th>LSB_1 LSB_2 LSB_3 LSB_4 LSB_5 LSB_6 LSBl</th>
</tr>
</thead>
<tbody>
<tr>
<td>. . .</td>
<td>0</td>
<td>. . . . . . 0 . . .</td>
</tr>
<tr>
<td>Cover</td>
<td>Voice Signal (p)</td>
<td>LSB_1 LSB_2 LSB_3 LSB_4 LSB_5 LSB_6 LSBl</td>
</tr>
<tr>
<td>. . .</td>
<td>1</td>
<td>. . . . . . 1 . . .</td>
</tr>
<tr>
<td>Cover</td>
<td>Voice Signal (p)</td>
<td>LSB_1 LSB_2 LSB_3 LSB_4 LSB_5 LSB_6 LSBl</td>
</tr>
<tr>
<td>. . .</td>
<td>0</td>
<td>. . . . . . 1 0 0</td>
</tr>
<tr>
<td>Cover</td>
<td>Voice Signal (p)</td>
<td>LSB_1 LSB_2 LSB_3 LSB_4 LSB_5 LSB_6 LSBl</td>
</tr>
<tr>
<td>. . .</td>
<td>1</td>
<td>. . . . . . 0 1 1</td>
</tr>
</tbody>
</table>

3. Experimental Results:

In this section, a number of experiments which are used to investigate the effectiveness of our proposed algorithm will be performed. The algorithm is programmed in MATLAB version 6.5 on Pentium IV PC (3 GHz) using three color images of size \((256 \times 256)\).

In our experiments, we use the following voice signal as a message \((M)\). Figure (3) explains the audio we need to hide it. The size of embedded voice signal is 6816 bits.
Before we hide the message we permute the voice using stego-Key to compute (P) from M. Figure (4) explains the message after permutation.
Boys, Lena, and map images of size 256*256 are applied as cover images for comparison. Figure (5) shows the three cover images.

To measure the difference between the original cover and stego-image we use the Peak signal-to-noise ratio (PSNR), which is expressed as the following equation [10]:

\[
PSNR = 10 \log_{10} \frac{255^2}{MSE} \quad \ldots(1)
\]

and Mean-Square Error (MSE) is defined as [10]:

\[
MSE = \left( \frac{1}{H \times W} \right) \sum_{i}^{H} \sum_{j}^{W} (x_{ij} - x'_{ij})^2. \quad \ldots(2)
\]

where \(H, W\) are the size of the cover image (\(H = 256, W = 256\) in this paper), \(x_{ij}\): is the original cover image, and \(x'_{ij}\): is the stego-image.

For color images, the reconstruction of all three color spaces must be considered in the PSNR calculation. The MSE is calculated for the reconstruction of each color space. The average of these three MSEs is used to generate the PSNR of the reconstructed RGB image. The color PSNR equations are as follows [10]:

\[
PSNR = 10 \log_{10} \frac{255^2}{MSE_{RGB}} \quad \ldots(3)
\]

\[
MSE_{RGB} = \frac{MSE_{red} + MSE_{green} + MSE_{blue}}{3} \quad \ldots(4)
\]

where \(MSE_{red}\) (or green or blue) is similar to Equation (2) for each color space.

We use three experiments which listed here.

**Experiment 1**

In this experiment, the LSB 3 method is used to embed the message (P) in the three covers separately without any modification to LSB 1 and LSB 2 of the color cover images. We obtain the following results, as shown in Table (1).
Table (1): Results of experiment 1

<table>
<thead>
<tr>
<th>Cover image</th>
<th>PSNR (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boys</td>
<td>48.9446</td>
</tr>
<tr>
<td>Lena</td>
<td>48.9742</td>
</tr>
<tr>
<td>Map</td>
<td>48.9563</td>
</tr>
</tbody>
</table>

**Experiment 2**

In this experiment, LSB 3 is used to embed the message (P) in the three color covers separately, but with modifying the LSB 1 and LSB 2 of the cover image as seen in Section (2.1). We obtain the following results, as shown in Table (2). Figure (6) explains the three stego-images after embedding the message.

Table (2): Results of experiment 2

<table>
<thead>
<tr>
<th>Cover image</th>
<th>PSNR (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boys</td>
<td>54.9640</td>
</tr>
<tr>
<td>Lena</td>
<td>54.9704</td>
</tr>
<tr>
<td>Map</td>
<td>54.8956</td>
</tr>
</tbody>
</table>

Figure (6): Stego-images with LSB 3

**Experiment 3**

To evaluate the transparency of embedded data of our method, we are increased in the amount of bits of voice which are different of whose embedded, according to the quality of the cover.
4. Conclusions:

In this paper, we have presented a suggested technique for data hiding. This work presents a new spatial domain data hiding method used for steganography applications. Our method of embedding message in the LSB 3 color image cover, and modifying LSB 1 and LSB 2 of the color cover, minimized the difference between the old values of the cover pixels and the stego-images. This minimization (increasing PSNR) leads to provide high secret communications, so the attacker cannot notice the difference between the stego-image and the original cover.

Based on the results obtained and detailed in the previous sections, we include the following:

- Steganography is the practice of encoding secret information in a manner such that the very existence of the information is concealed. Moreover, to detect the message's existence will be very hard for those stego-images.
- The proposed technique can be defined as a secret key steganography since it shares a secret key (stego-Key) between sender and receiver, in this technique there is no need for the knowledge of original cover in the extraction process.
- In Figure (7), both the PSNR values of the stego-image and the original cover image result in a trade-off problem. That is, we find that as the size of embedded text is increased, the PSNR is decreased and vice versa. Figure (8) shows PSNR versus the number of embedded bits.

![Figure (7): PSNR versus the number of the embedded bits for color boys image](image)

(a) 904 bits, PSNR = 63.9202 dB.
(b) 2840 bits, PSNR = 58.6793 dB.
(c) 6816 bits, PSNR = 54.9640 dB.

![Figure (8): PSNR versus the number of embedded bits for color boys image](image)
However, the amount of information can actually be hidden in an image that depends upon the composition of the image. An image that contains high frequency areas can be manipulated (not make sense from the viewpoint of human eyes) more than an image containing primarily low frequency areas (high noticeable) of the stego-image as shows in Table (2).

The proposed model has proved to be easy to use and efficient in terms of security.

5. References: