A novel steganography approach for audio files

ABDULRAZZAQ, Sazeen T, SIDDEQ, Mohammed M and RODRIGUES, Marcos <http://orcid.org/0000-0002-6083-1303>

A Novel Steganography Approach for Audio Files

Sazeen T. Abdulrazzaq[1], *Mohammed M. Siddeq[1], Marcos A. Rodrigues[2]
sazeentaha4@gmail.com, mamadmmx76@gmail.com, M.Rodrigues@shu.ac.uk
[1]*Computer Engineering Dept., Technical College/Kirkuk,*
Northern Technical University, IRAQ

[2]*GMPR-Geometric Modelling and Pattern Recognition Research Group,*
Sheffield Hallam University, Sheffield, UK

Abstract
We present a novel robust and secure steganography technique to hide images into audio files aiming at increasing the carrier medium capacity. The audio files are in the standard WAV format, which is based on the LSB algorithm while images are compressed by the GMPR technique which is based on the Discrete Cosine Transform (DCT) and high frequency minimization encoding algorithm. The method involves compression-encryption of an image file by the GMPR technique followed by hiding it into audio data by appropriate bit substitution. The maximum number of bits without significant effect on audio signal for LSB audio steganography is 6 LSBs. The encrypted image bits are hidden into variable and multiple LSB layers in the proposed method. Experimental results from observed listening tests show that there is no significant difference between the stego audio reconstructed from the novel technique and the original signal. A performance evaluation has been carried out according to quality measurement criteria of Signal-to-Noise Ratio (SNR) and Peak Signal-to-Noise Ratio (PSNR).

Keywords: audio steganography, LSB, 2D Image compression, DCT, Matrix Minimization, Binary Search Algorithm.

1. Introduction

Steganography is the skilful masking of data in a coating media such as text, image and video. The term steganography originates from Greek which means "Covered Writing". Steganography is a widely used technique in the area of information technology [1]. Throughout history, steganography methods included methods such as hiding messages in the belly of a hare using invisible ink or shaving of a courier's head to write a message or a tattoo, or an image on the messenger's head [2]. Steganography thus, supplies techniques to cover the existence of a secondary message inside a primitive message. The primitive message is delegate to as the carrier signal which could be text, audio, image, video, etc., while the secondary (hidden) message is assigned to as the loaded message [3]. A signal is being hidden in such a way to be unrecognized to the onlooker, while the primary signal (carrier) is modified in an imperceptible form [4]. Therefore, in steganography the original message is not changed and the presence of a hidden message is not apparent to the observer as it is embedded in the selected medium [5].

Audio steganography is defined as the Cover Audio + Secret data = Stego audio signal [6,7,8]. The essential requirements for audio steganography are: 1) the secret data should not be perceptible to humans [6,7]; 2) it should maximize the hiding capacity; and 3) preferably, the hidden data should be encrypted. In the digital world, two types of objects are used: either audio or image files as stego-objects for masking the furtive message [9]. By performing a bit change in the binary sequence of a sound file, the secret data is embedded into the original file. In the last few years, different approaches have been developed for the hiding and extraction of a data from audio signals. Most of the developed approaches are based on the cognitive characteristics of the Human Auditory System (HAS) in order to add a message/data to a host signal in a cognitive transparent way. Hiding data/message into audio signals is an interesting endeavour but at the same time more difficult, as the HAS is more sensitive to small variations in sound than the Human Visual System (HVS) is to small changes in image intensities [10].

The objective is thus, to merge the secret data into an audio data such that there are insignificant differences between the original audio data and the embedded file. The embedded file contains a header, with essential information about the audio. This information is untouchable; in a WAVE file for instance, the first 44 bytes contain information about the file and any changes to this header will corrupt the WAVE file [3]. A general audio steganography method is illustrated in Figure 1.

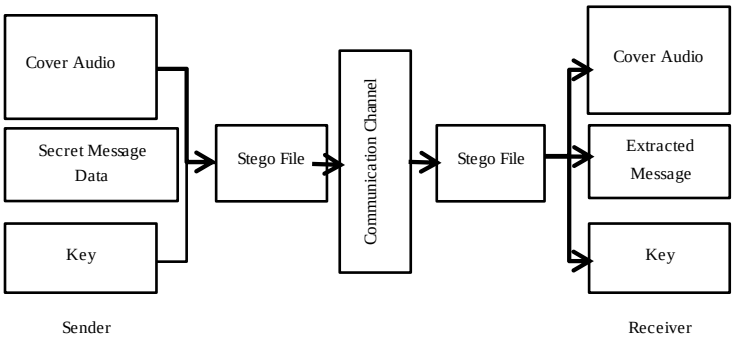


Figure 1: General audio steganography method

The masking of the confidential data into the secret medium should not make any loathsome changes to the secret medium so that the authenticity of the file should not be disturbed [11]. The audio steganography view is to merge precious/undercover encrypted data into an audio data in such a way that the human auditory system (HAS) cannot expose the changes which happened through merger of the encrypted data into the audio data. In audio steganography, the Least Significant Bit (LSB), Spread Spectrum, and Echo hiding approaches have been developed in recent years [12]. The main properties of audio steganography [2] being exploited in different steganography applications are: confidentiality, imperceptibility, high capacity, difficult to detect ability, accuracy, survivability and visibility.

2. State-of-the-Art in Audio Steganography

In [17] an approach is described for resolving the problem related to the substitution technique of audio steganography. In the first level of security, the RSA algorithm is used to encrypt the message and, in the next level, the encrypted message is encoded into audio data. A genetic algorithm-based substitution method is used to encode the data. The basic idea behind the method is to enhance both security and robustness of audio steganography. In [18] a novel approach is proposed where a duel encryption methodology is implemented. In the first level of encryption, a pattern matching algorithm has been employed to encrypt the text message in terms of their positional value. In the second level, the conventional LSB method has been used to embed the positional value in the cover file. Such a duel encryption method ensures data security in an efficient manner. In [19] a method for concealing data is proposed through an amalgamation of text encryption, audio steganography and audio encryption. In the first step, the original text message is encrypted using a modified Vigenère cipher algorithm. The cipher text gets embedded into the cover audio using LSB encoding in the second step. Finally, the audio file is then subjected to transposition making use of Blum Blum Shub pseudorandom number generator. The authors recommend more secure encryption algorithms to be utilized for text encryption, so that data is not easily decoded by an unauthorized party. The work in [20] gives an overview of two primitive techniques to highlight how steganography in audio file works and what are their main desirable characteristics. Both LSB modification and phase encoding techniques are extensively exploited. The main characteristics of an effective audio steganographic scheme are emphasized as inaudibility of distortion, data rate, and robustness. These characteristics are referred to as the magic triangle of data hiding.

The novel contributions of this paper are best described in the context of our previous work on image compression [13]. In that work, we used double discrete transforms (i.e. DCT-Discrete Cosine Transform and DWT-Discrete Wavelet Transform) combined with Matrix Minimization algorithm yielding 94% compression ratios for grey scale conventional images. The main disadvantage of the previous method was excessive compression and decompression execution times that could run from seconds to minutes. The proposed method described here is based on an enhanced Matrix Minimization algorithm (see Section 4). The method is based on the DCT and is time-wise more efficient than previous methods. A further contribution of the proposed method is higher compression ratios compared with both previous work [13] and the JPEG technique. In this paper, the proposed method (based on DCT with enhanced Matrix Minimization algorithm) is compared with the DCT-based JPEG technique, as this is the closest technique to the proposed method.

3. Method

The proposed algorithm aims at reducing noise, increasing robustness and increasing compressed-encrypted image embedding capacity by including negative audio bytes in the encoding process. In this paper, the compressed-encrypted image can be hidden into the audio file using the following two major steps:

1. Compress-encrypt the image using the GMPR method [13,14]
2. Hide the image into audio files using LSB based encoding

In the first step, the secret image file is encrypted and compressed using the GMPR algorithm (see Section 4 for more details). The sender generates a compression-encryption key. This key is transferred to the receiver through secure means. The encrypted image is converted into a binary string array list.

In the second step, the encrypted image is converted into a stream of bits to encode using the proposed LSB Algorithm. The proposed steganography algorithm embeds the 6-bits (encrypted image) randomly into the audio byte in the higher LSB positions (1-10) for audio byte (the audio byte can be processed as 16-bit sample). This operation is done 100 times every sample by calculating the difference between the value of original audio sample and the value of the 100 stego audio samples. The best embedding position is the sample that has the least difference between it and the original sample.

The following steps provide a detailed explanation on how to encode the message data into the given audio file:

- Read the image file and compress-encrypt it using the GMPR method which will generate the key for decompression. The key is transferred to the receiver through secure means (not part of the algorithm).
- The encrypted image file is converted into a binary string array list.
- Read the cover audio file and converted to binary (16- bits).
- To embed the encrypted image bits, instead of using the least significant bits, higher (LSB) are used. In order to increase the quality of audio, the secret data bits are randomly embedded into audio samples. In each sample 6 bits are embedded randomly in (1-10) positions. To find the best embedding position the following steps are performed:
    1. Substitute audio bits with encrypted image bits.
    2. For every sample repeat this substitution randomly 100 times and evaluate the difference between the original audio sample and the generated (substituted) audio sample bytes.
    3. To reduce the amount of error, choose the sample with smaller difference. Eventually the best embedding position is the minimum value of difference found in the sample. Record the sample positions.
    4. The next step after finding the best embedding position is to place the encrypted image bits sequentially inside the audio samples at the specified locations.
- Repeat steps 1 to 4 for all encrypted image bytes.
- Mark the end of message encoding.
- The last step is writing the audio bytes into the output audio file (stego file). This is done sample by sample.

The workflow of embedding operation for the proposed algorithm is depicts in Figure 2.

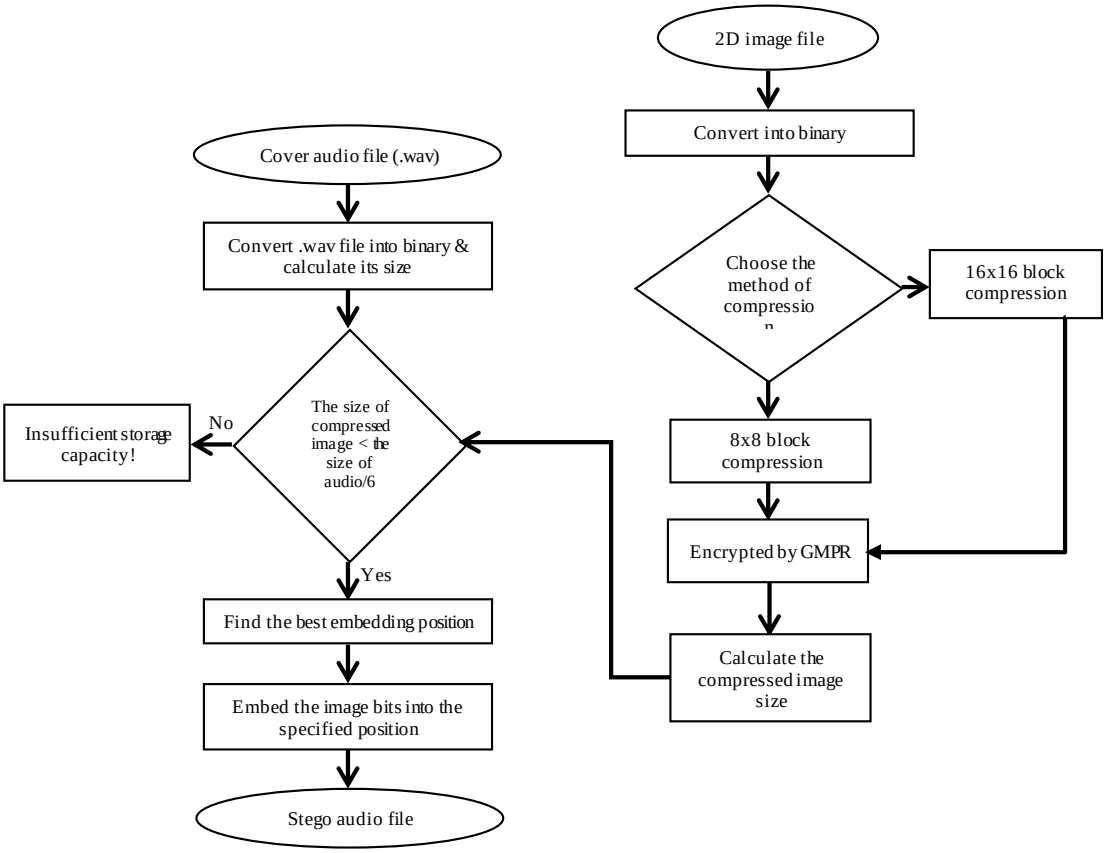


Figure 2: The workflow of embedding operation for the proposed method.

4. The GMPR Compression-Encryption Algorithm

In this section, the image compression-encryption is described. The method has been developed by Siddeq and Rodrigues at Sheffield Hallam University [13,14]. First, the image is compressed by a combination of either DWT-Discrete Wavelet Transform and DCT-Discrete Cosine Transform. In this paper we use the DCT. Second, the Matrix Minimization algorithm produces a stream of encoded data. Third, header information with compression keys are concatenated to encoded data representing the compressed-encrypted image to be used in the steganography step.

The compression algorithm involves applying the DCT with quantization leading to two matrices: the DC-Array and the AC-Coefficients [4,5]. Figure 3 shows a layout of the compression-encryption method. The portion of this paper related to the compression of the matrix of AC-Coefficients which involves reducing the number of zeros in the matrix by the matrix minimization approach yielding a minimized array. The DC-Array can be represented in a few bytes by computing the differential between two adjacent coefficients in the array, increasing thus, the compression ratio for the array [13]. Both encoded DC and AC coefficients are then subject to arithmetic coding.

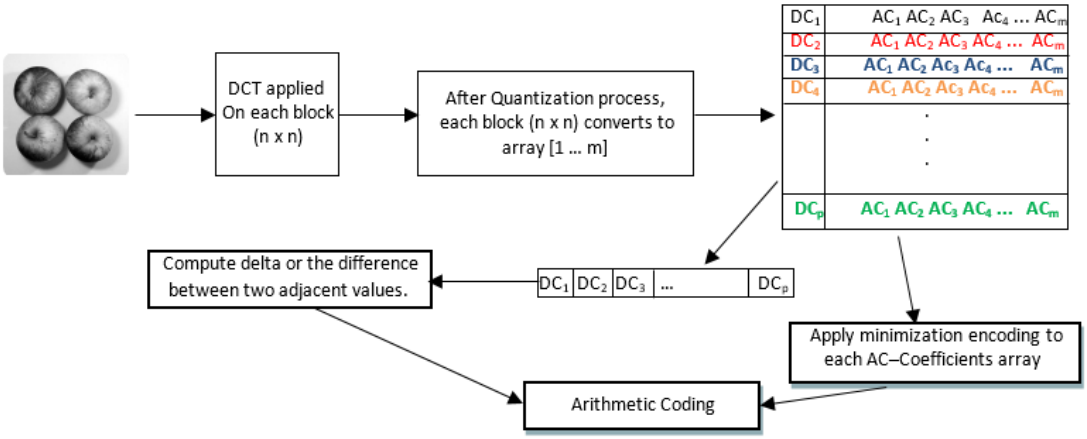


Figure 3: Illustration of the GMPR image compression-encryption algorithm

4.1 GMPR Compression Step 1: Discrete Cosine Transform (DCT)

Divide the image into blocks $n \times n$, followed by DCT which is applied to each block independently. The output from DCT is a set of de-correlated coefficients, and it means that each block consists of a DC value (the first position in the block) while all other values are called the AC coefficients. The DCT is defined by Eq. (1) [15]:

$$C(i,v) = a(u)a(v)\sum_{x=0}^{n-1}\sum_{y=0}^{n-1} f(x,y)cos\left[\frac{(2x+1)u\pi}{2n}\right]\left[\frac{(2y+1)v\pi}{2n}\right] \quad (1)$$

Where $a(u) = \sqrt{\frac{1}{n}}, for\ u = 0, a(u) = \sqrt{\frac{2}{n}}, for\ u \neq 0.$

Where $i, j = 1,2,\ldots,n$

The quantization of each block $n \times n$ can be represented as follows:

$$Q_{(i,j)} = L * (i+j) \quad (2)$$

The transformation is followed by quantization by rounding off floating point values to integers. Each $n \times n$ block is quantized by Eq. (2) using dot-division-matrix. The main reason to use quantization is to remove insignificant coefficients (from AC coefficients) leading to an increased number of zeroes in each block, which then yields increased compression ratios.

The parameter L is used to change the quantization values in matrix Q. Therefore, image quality is decreased or increased according to the value of L. The range of L depends on the maximum value of the DCT coefficients. Thus, the minimum value is L=1 and the maximum value is L=maximum value of DCT coefficients.

Next, the GMPR algorithm separates the DC-components from each block by saving them into a new array called DC-Array. After that, the differential between two adjacent values in the DC-Array are computed. The reason behind this process is to increase compression ratios and facilitate lossless data compression (by arithmetic coding). For information, values in the DC-Array are generally similar. So, the differences between these values are small and many of these data may be repeated [14].

$$D_i = D_i - D_{(i+1)} \quad (3)$$

Where $i = 1, 2 \ldots p\text{-}1$ and p is the size of D.

The remaining AC coefficients (for example, for an 8x8 block there are 63 remaining AC coefficients) of each block are converted into an array and concatenated together. This means that we generate multiple arrays (each block is converted to an array by column-scan). The new array is called AC-Matrix. Now

this matrix is ready for coding by the matrix minimization algorithm which is described in the next section.

4.2 GMPR Compression Step2: Matrix Minimization Algorithm

The output from GMPR Step 1 is an AC-Matrix and a DC-Array. Here the AC-Matrix is encoded by the matrix minimization method, which focus on reduction of zeros and squeezing each triplet data into a value, and then the output is subjected to arithmetic coding. Normally, the AC-Matrix has a large number of zeros with some data scattered randomly in the matrix. A technique to eliminate blocks of zeros and save blocks of nonzero data into an array starts by dividing the AC-Matrix into non-overlapping blocks n×n (n≥8). Each block has been checked for nonzero data, if nonzero data are found, then the block is saved as a nonzero array. Meanwhile, the location of that block is recorded which means that all missing blocks contain just zeros. The algorithm is illustrated below [13].

Algorithm 1: Eliminating Zeros from the AC-Matrix

```
1.  Set block size n
2.  For-loop every column in AC-Matrix do {
3.      For-loop every row in AC-Matrix do{
4.          B=Read block( n, n)
5.          If it is a non-zero block of data {
6.              Save the location of the block in a temporary array
7.              Save block data as one-dimension in new array "R-Array "
8.          end; // if-statement
9.      end; // Loop 2
10. end; // Loop 1
```

Once only nonzero data are saved into the R-Array, The Matrix minimization algorithm is applied to encrypt and compress at 3:1 ratio (that is, every 3 values are coded into a single value). This process uses three key values and multiplies these keys by three adjacent entries in the R-Array which are then summed over and saved in an encoded array called E-Array. It is important to note that the compression keys K1, K2, K3 are data dependent and generated by a key generator algorithm as follows [14]:

$$M = 0.5 * \max(\boldsymbol{R\ Array}) \quad (4)$$

$$F = \text{user defined integer scale factor} \geq 1 \quad (5)$$

$$K1 = \text{random}(0,1) \quad (6)$$

$$K2 = K1 + M + F \quad (7)$$

$$K3 = F * M * (K1 + K2) \quad (8)$$

Assuming that N is the length of R-Array, $i = 1,2,\ldots,N-3$ is the index of data in R-Array, and j is the index of E-Array (encrypted array), the following transformation defines the high frequency minimization encoding:

$$\boldsymbol{E\ Array}_j = K1 * (\boldsymbol{R\ Array}_i) + K2 * (\boldsymbol{R\ Array}_{i+1}) + K3 * \boldsymbol{R}_{i+2} \quad (9)$$

Each value of the encoded E-Array is from triplet summation of Eq. (9), and the reduced R-Array can later be reconstructed through search for values for that block. However, this problem requires extra information which is kept in the header of the compressed file as an array of unique-data as shown in Figure 4.

Each image has its own AC-Matrix coefficients. The GMPR algorithm computes the Unique-Data for the AC-Matrix so that this Unique-Data cannot be used to decode another image's AC-Matrix. The final step of GMPR compression is to apply arithmetic coding to the E-Array yielding a stream of lossless data. Arithmetic coding computes the probability of all data and then set a range to each data (low and high) used by two special equations to produce a single floating-point value; this floating-point value is converted to a stream of bits and saved as compressed file. The main reason we use arithmetic coding instead of Huffman coding is because it yields higher compression ratios although the performance of Huffman coding is higher [14].

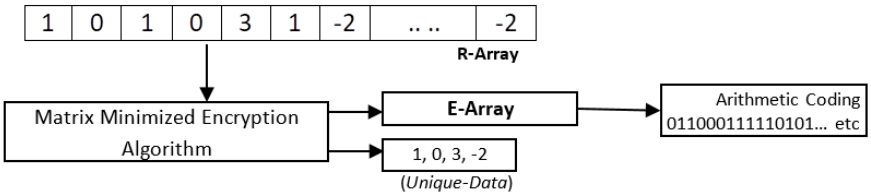


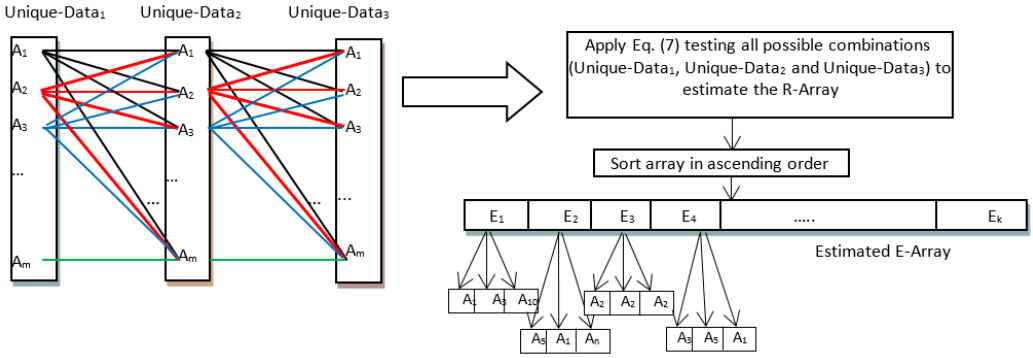
Figure4: Unique-Data is computed through the Matrix Minimization algorithm, and this information is kept in the header file that will later be used as necessary key information for recovery [13,14].

5. The GMPR Decompression-Decryption Steps

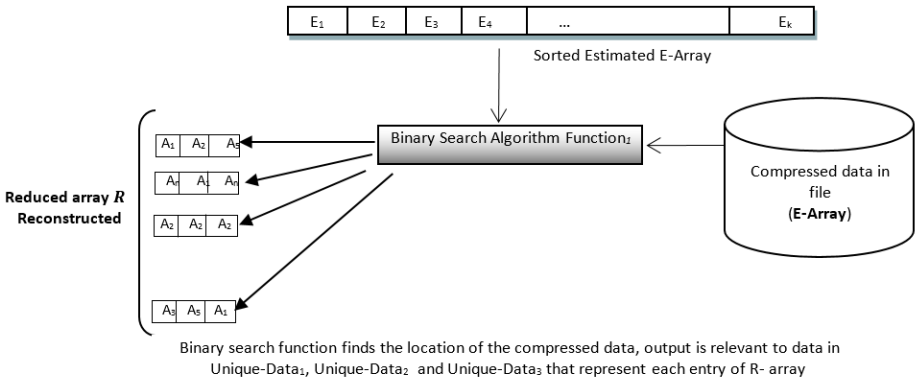
In the first step in the GMPR decompression-decryption method the DC-Array can be decoded in a very simple way, by an addition process (i.e. the inverse of Eq. (3)) as described by Eq. (10) below.

$$D_{(p-1)} = D_{(p-1)} + D_{(p)} \tag{10}$$

Where p represents the last location in the DC-Array. The addition process takes the last value at position p, and adds it to the previous adjacent value, and then the total adds to the next previous adjacent value and so on. This process will continue until $p = 1$ (first position).



(a) Compute all possibilities for keys with Unique-Data to reconstruct the reduced ***R***-Array



(b) The Binary Search algorithms works to find decompressed data [13].

Figure 5: The BS-Algorithm to reconstruct the R-Array.

In the second step, we focus on recovering the R-Array that has been compressed into the E-Array. For this purpose, Siddeq and Rodrigues proposed a Binary Search Algorithm (BS-Algorithm) [14]. The header compressed file contains information about the three keys (i.e. defined in Eqs. (6,7,8) – K1, K2 and K3), the unique data and compressed streams of data (E-Array) respectively. The BS-Algorithm role is to fetch each compressed value from the E-Array and reconstruct the original triplet R-Array of data. This is described in steps A—D [13,14] below where Unique-Data array is sorted ascending:

A) Estimate 3 coefficient values from the Unique-Data array as Unique-Data1=Unique-Data2=Unique-Data3. The search algorithm computes all possible outputs from Eq. (9), which based on K1, K2 and K3. Therefore, the output is saved in an temporary array called estimated E-Array. As a means of an example consider that [Unique-Data1, Unique-Data2, Unique-Data3]=[A1, A2, A3] (that is, the first 3 values in the Unique-Data array). Then, according to Eq. (9) these represent the R-Array values summation respectively, and the equation is executed multiple times, to build the R-Array, as described in Figure 5(a). A match indicates that the estimated Unique-Data1, Unique-Data2 and Unique-Data3 represent the recovered data.

B) A Binary Search algorithm [16] is used to recover the data and their keys. Our design consists of single binary search algorithm to reconstruct the triplets of original data in the R-Array, as shown in Figure 5(b). At each step, the decompressed data is compared with the middle element of the estimated E-Array. If the values match, then a matching element has been found and its relevant (Unique-Data1, Unique-Data2 and Unique-Data3) returned. Otherwise, if the search is less than the middle element the algorithm is repeated to the left of the middle element or, if the value is greater, to the right until a match is found.

C) Once the R-Array is reconstructed, the nonzero blocks are relocated in an empty matrix according to locations saved by Algorithm 1 (Section 1), these will be used to reconstruct the AC-Matrix.

D) The DC-Array values are combined with the AC-coefficients and the high frequency AC-Matrix is re-built. Each row from the AC-Matrix is converted to a block n × n, and then followed by inverse quantization (dot multiplication with Eq. (2)) and the inverse DCT is applied to each n × n block according to Eq. (11) defined below, to reconstruct the original image. Figure 6 illustrates the decompression-decryption steps [15].

$$f(x,y) = \sum_{u=0}^{Block-1} \sum_{v=0}^{Block-1} a(u)a(v)C(u,v)\cos\left[\frac{(2X+1)u\pi}{2Block}\right]\cos\left[\frac{(2y+1)v\pi}{2Block}\right] \quad (11)$$

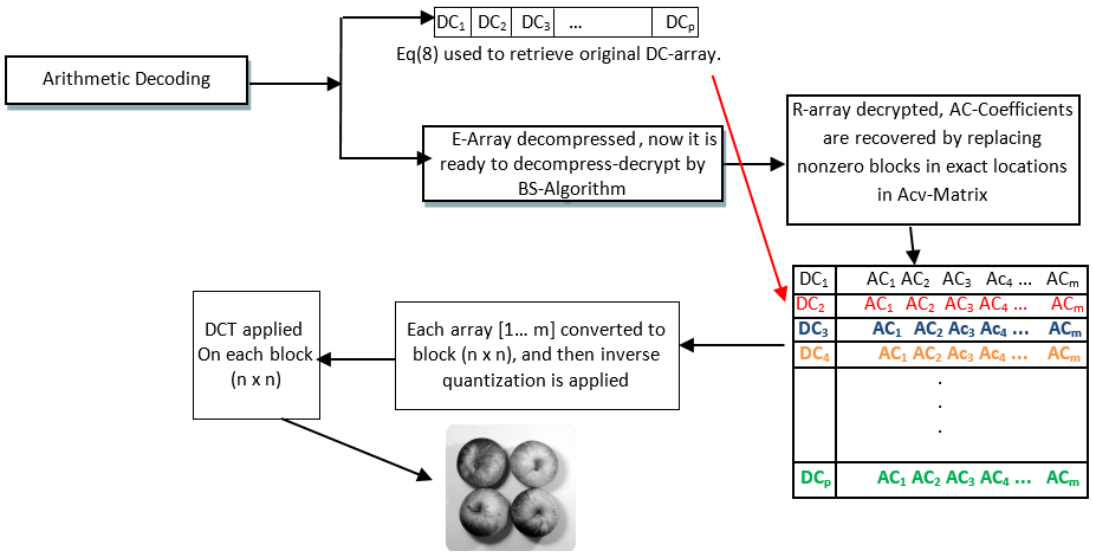


Figure 6: The layout of the GMPR decompression-decryption.

6. Results and Analysis

The proposed method is demonstrated by 1) comparing the SNR and PSNR values obtained from 8x8 and 16x16 block sizes; 2) plotting the resulting stego audio file and comparing it with original audio file; and 3) listening to the original and stego files for perceptual assessment. To test the performance of the proposed method, the used audio files frequency is 44.1 KHz and its duration is 7 sec. Different image file sizes are used (as shown

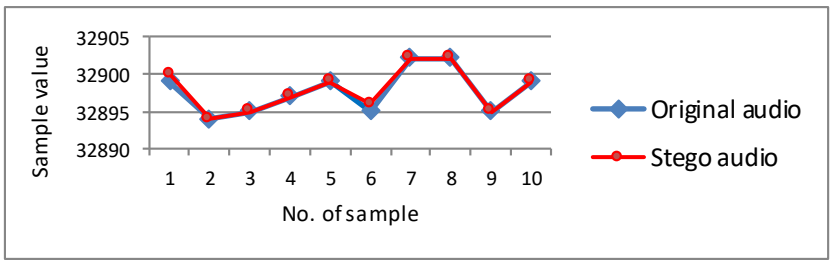
in Table 1). First, the SNR and PSNR are computed for the original audio file and stego audio files using proposed method. Table 1 shows results of SNR and PSNR for the original and stego audio files after embedding the secret image divided into blocks of 8x8 and 16x16.

Table 1: Compressed image by GMPR used in the Audio

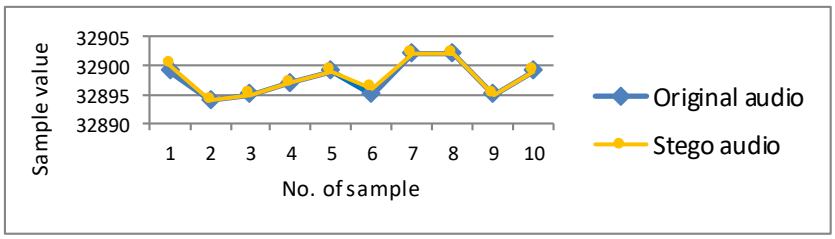
Image name	Original Image Size (MB)	Block Size	Compressed Image Size (KB)	Audio SNR	Audio PSNR
Apples	1.37 MB	8x8	152 KB	61.1895	74.0832
Lena	1.0 MB		105 KB		
Card	1.37 MB	16x16	14.4 KB	61.5147	74.4083
Lena	1.0 MB		23 KB		
Apples	1.37 MB		20.3 KB		
Guitar	1.37 MB		14.1 KB		

As shown in Table 1 block sizes of 8x8 (light grey) yields larger image sizes than block sizes of 16x16 (dark grey). The result is that we can embed only two images of 8x8 block size into the audio file while we can embed four 16x16 block images into the same audio file. The perceptual assessment of the stego-audio file of the proposed work is almost the same when compared with the original audio file, it is hard to notice any difference. As presented in Table 1 the SNR and PSNR values of proposed work are high. High values mean that the hidden message remains undetected or it is hard to notice. By using 6 LSB, the information embedding capacity is high.

Figure 7 depicts the difference between the original audio file and the stego-audio file using the proposed method. The stego-audio file after embedding 2 images with 8x8 block compression is shown in Figure 7(a) and the stego-audio file after embedding 4 images with 16x16 block compression is shown in Figure 7(b). Both are almost identical as compared with the original audio file.



(a) Deviation between the original and stego audio file using 8x8 block for image encryption

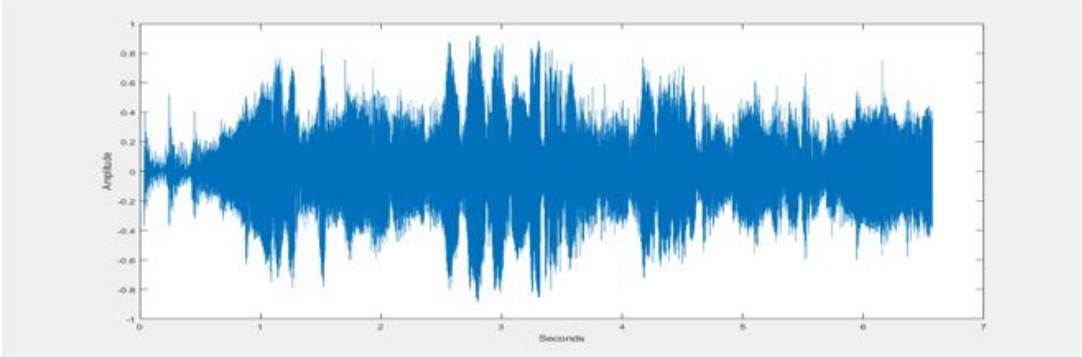


(b) Deviation between the original and stego audio file using 16x16 block for image encryption

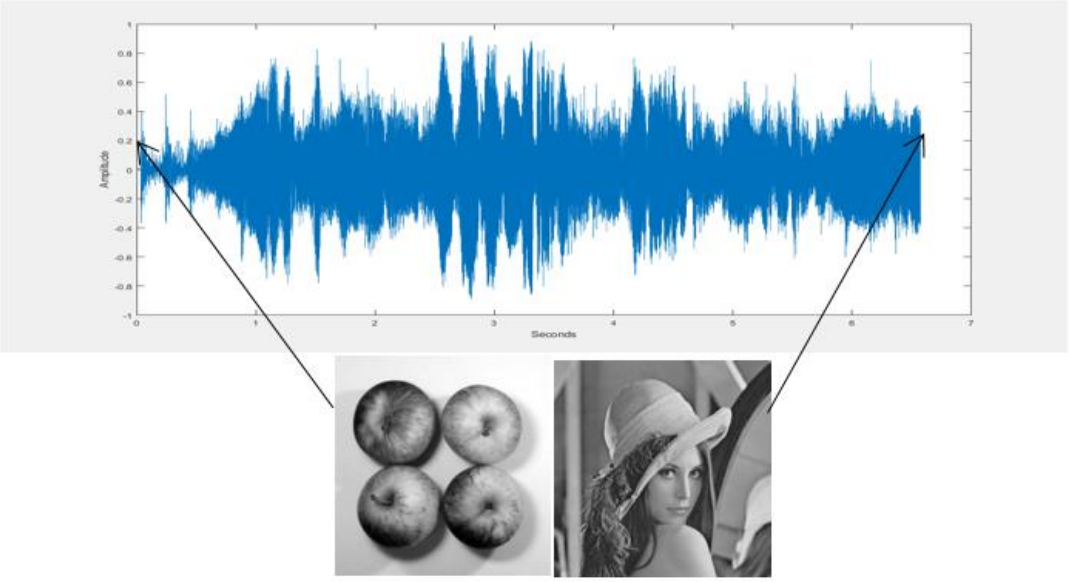
Figure 7: Deviation between the original and stego audio files

Figure 8 shows that before and after embedding, the overall size of the audio file remains the same. Thus, the audio steganography is successful as there are undetectable or non-audible differences between the original and the stego files. The original cover audio sample is shown in Figure 8(a), the stego wav file

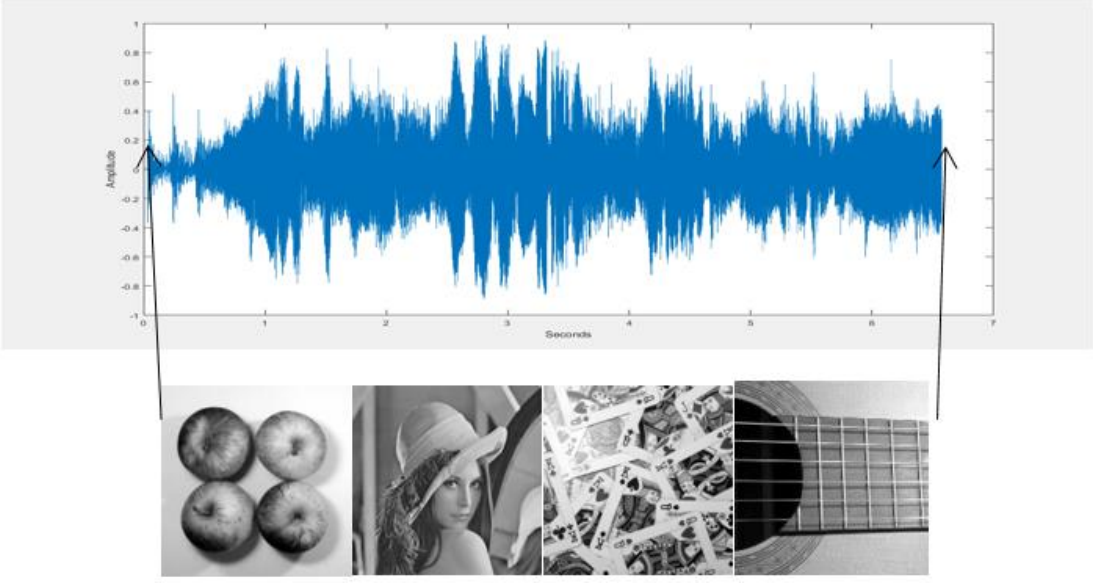
after embedding 2 images (8x8) is shown in Figure 8(b) and the stego wav file after embedding 4 images (16x16) is shown in Figure 8(c).



(a) The original wav file



(b) The stego wav file after embedding 2 images (8x8)



(c) The stego wav file after embedding 4 images (16x16)

Figure 8: The waveform of original and stego-audio files using the proposed methods

7. Comparison with JPEG

JPEG is a main technique used in many digital image processing applications and digital video processing, for this reason we chose this technique to compare with our approach [15]. Another reason is that JPEG is based on the DCT and, by necessity, there are a few steps in the JPEG technique that are similar to our compression-encryption method [13]. The assessment presented here focuses on image quality (Root Mean Square Error – RMSE) [13,14] and compression ratios, before engaging with the steganography method. Table 2 shows our compression-encryption method results, while Table 3 shows the comparison between our method and JPEG technique. Figure 9 and Figure 10 show the decompressed images by our method and JPEG technique respectively.

Table 2: Compression method results

Image name	Original Image Size	Block Size	Quantization Factor	Compressed Image Size	RMSE
Card	1.37 MB	8x8	1	360 KB	3.5
		16x16	10	14.4 KB	8.6
Apples	1.37 MB	8x8	1	152 KB	1.06
		16x16	10	20.3 KB	3.2
Lena	1.0 MB	8x8	2	105 KB	1.7
		16x16	10	23 KB	4.3
Guitar	1.37 MB	8x8	2	95 KB	1.3
		16x16	10	12.8 KB	4.9

Original Image Decompressed Image, RMSE=3.5 Decompressed Image, RMSE=8.6

(a) Card image with dimensions (1200 x 1200)

Original Image Decompressed Image, RMSE=1.06 Decompressed Image, RMSE=3.2

(b) Apples image with dimensions (1200 x 1200)

Original Image Decompressed Image, RMSE=1.7 Decompressed Image, RMSE=4.3

(c) Lena image with dimensions (1024 x 1024)

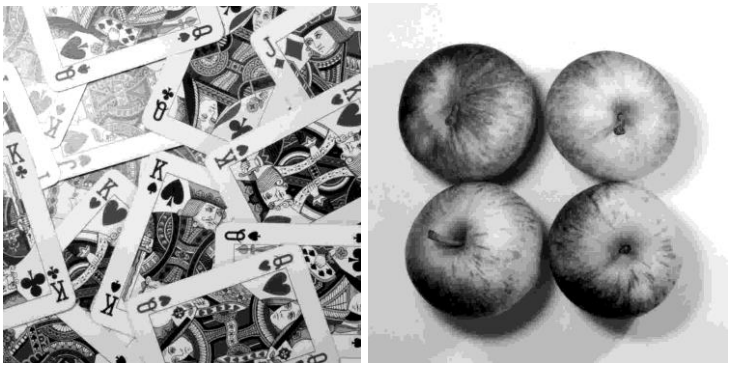
Original Image Decompressed Image, RMSE=1.3 Decompressed Image, RMSE=4.9

(d) Guitar image with dimensions (1200 x 1200)

Figure 9: (a-d) (Left): Original 2D images with different dimensions. (Middle and Right), decompressed 2D images by the GMPR compression-encryption method representing high- and low-resolution images respectively.

Table 3: Comparison of our method results with JPEG technique

Image name	Original Image Size	Compression Method	Compressed Image Size	RMSE
Card	1.37 MB	JPEG	50.5 KB	11.68
		Our Method	14.4 KB	8.6
Apples	1.37 MB	JPEG	24.2 KB	6.4
		Our Method	20.3 KB	3.2
Lena	1.0 MB	JPEG	28 KB	5.0
		Our Method	23 KB	4.3
Guitar	1.37 MB	JPEG	25 KB	7.3
		Our Method	12.8 KB	4.9



(a) Decompressed Image, RMSE=11.68 (b) Decompressed Image, RMSE=6.4

(c) Decompressed Image, RMSE=5.0 (d) Decompressed Image, RMSE=7.

Figure 10: (a-d): Decompressed 2D images by JPEG technique at higher compression ratios.

8. Conclusions

A new method for embedding image data into audio files using random and higher LSB audio steganography has been successfully demonstrated in this paper. The proposed technique offers large capacity for secret message embedding, it is robust, and the quality of embedded images is not reduced by noise or other artefact distortions. The proposed technique is robust to stego analysis attacks because in the cover audio file more than one bit has been changed to produce the stego file. The resulting stego audio file has high PSNR and SNR ratios indicating small noise distortions. Perceptually, the resulting stego-audio file is almost identical to the cover audio file, so the embedding is imperceptible.

The analysis of the proposed algorithms indicates that the most important aspects of the method and their role in providing robust embedding and recovering of high-quality image with high compression ratios are as follows:

1. As a first step, DCT applied to block sizes 8x8 and 16x16, after that the DC-components and AC-coefficients are split into two different matrices, and each of these matrices are coded separately.
2. Normally, the AC-coefficients contain a large number of zeros. The elimination of zero data and the keeping of significant information by the Matrix Minimization algorithm reduces the AC-Matrix contents by more than 80%.
3. The Matrix Minimization algorithm replaces each three coefficients from the AC-coefficients by a single floating-point value leading to increased compression ratios.
4. At decompression stage, the BS-Algorithm is the engine for estimating the original data from the E-Array and relies on the key values and the availability of a set of unique data.
5. The keys used in the compression-encryption method are data-dependent and, without the keys, images cannot be decoded. For this reason, the method is referred to as image compression-encryption and suitable for secure transmission and storage of data.
6. The compression-encryption image algorithm was tested on grey images at high compression ratios and compared with the JPEG technique. Results demonstrate that our method is more suitable than JPEG for steganography yielding higher compression ratios as shown in Table 3.

Finally, Table 1 has shown that the proposed technique yields SNR over 61% which is higher than work reported in the literature. In addition, the proposed method offers a level of security that is unmatched by current work. We understand that developing steganography methods that are at the same time highly secure and highly robust are difficult to achieve. We demonstrated methods that make a significant contribution towards such requirements so that future work will be focused on improving on those requirements.

Acknowledgments: We grateful acknowledge the Computing, Communication and Cultural Research Institute (C3RI) and the Research and Innovation Office at Sheffield Hallam University for their support.

Conflicts of Interest: The authors declare that there are no conflicts of interest regarding the publication of this paper.

References

1. Bangera, K.N., Reddy, N.S., Paddambail, Y. and Shivaprasad, G., 2017, May. Multilayer security using RSA cryptography and dual audio steganography. In *2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT)* (pp. 492-495). IEEE.
2. El-Khamy, S.E., Korany, N.O. and El-Sherif, M.H., 2017. A security enhanced robust audio steganography algorithm for image hiding using sample comparison in discrete wavelet transform domain and RSA encryption. Multimedia Tools and Applications, 76(22), pp.24091-24106.
3. PRASAD, L.C. and RAO, V.S.R., 2017. MATLAB Implementation of Audio Steganography for Secure Data Transmission.
4. Hashim, J., Hameed, A., Abbas, M.J., Awais, M., Qazi, H.A. and Abbas, S., 2018, November. LSB Modification based Audio Steganography using Advanced Encryption Standard (AES-256) Technique. In 2018 12th International Conference on Mathematics, Actuarial Science, Computer Science and Statistics (MACS) (pp. 1-6). IEEE.
5. Kundu, N. and Kaur, A., 2017. Audio Steganography for Secure Data Transmission. International Journal of Computer Sciences and Engineering, 5(2), pp.124-129.
6. Atoum, M.S., Alnabhan, M.M. and Habboush, A., 2017. ADVANCED LSB TECHNIQUE FOR AUDIO STENOGRAPHY. CoSIT, SIGL, AIAPP, CYBI, CRIS, SEC, DMA, pp.79-86.
7. Din, R., Mahmuddin, M. and Qasim, A.J., 2019. Review on Steganography Methods in Multi-Media Domain. International Journal of Engineering & Technology, 8(1.7), pp.288-292.
8. Chen, K., Yan, F., Iliyasu, A.M. and Zhao, J., 2018. Exploring the implementation of steganography protocols on quantum audio signals. International Journal of Theoretical Physics, 57(2), pp.476-494.
9. Ali, A.H., George, L.E., Zaidan, A.A. and Mokhtar, M.R., 2018. High capacity, transparent and secure audio steganography model based on fractal coding and chaotic map in temporal domain. Multimedia Tools and Applications, 77(23), pp.31487-31516.
10. Mohamad, F.S. and Yasin, N.S.M., 2018. Information Hiding Based on Audio Steganography using Least Significant Bit. International Journal of Engineering & Technology, 7(4.15), pp.536-538.
11. Mohajon, J., Ahammed, Z. and Talukder, K.H., 2018, December. An Improved Approach in Audio Steganography Using Genetic Algorithm with K-Bit Symmetric Security Key. In 2018 21st International Conference of Computer and Information Technology (ICCIT) (pp. 1-6). IEEE.
12. Tan, D., Lu, Y., Yan, X. and Wang, X., 2019, March. A Simple Review of Audio Steganography. In 2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC) (pp. 1409-1413). IEEE.
13. SIDDEQ, M and RODRIGUES, Marcos (2015). A novel 2D image compression algorithm based on two levels DWT and DCT transforms with enhanced minimize-matrix-size algorithm for high resolution structured light 3D surface reconstruction. 3D Research, 6 (3), p. 26.
14. SIDDEQ, Mohammed and RODRIGUES, Marcos (2017). A Novel High Frequency Encoding Algorithm for Image Compression. EURASIP Journal on Advances in Signal Processing, 26. DOI: 10.1186/s13634-017-0461-4.
15. C. Christopoulos, A. Skodras, T. Ebrahimi, "The JPEG sill image coding system: an overview," IEEE Trans. Cons. Elect., vol. 46, pp.1103-1127, 2000.
16. Knuth, Donald (1997). Sorting and Searching: Section 6.2.1: Searching an Ordered Table, The Art of Computer Programming 3 (3rd Ed.), Addison-Wesley. pp. 409–426. ISBN 0-201-89685-0.
17. Singh, G., Tiwari, K. & Singh, S. (2014). Audio steganography using RSA algorithm and genetic based substitution method to enhance security. International Journal of Scientific and Engineering Research, 5(5), 703-707.
18. Chowdhury, R., Bhattacharyya, D., Bandyopadhyay, S. K., & Kim, T. H. (2016). A view on LSB based audio steganography. International Journal of Security and Its Applications, 10(2), 51-62.
19. Sinha, N., Bhowmick, A., & Kishore, B. (2015). Encrypted information hiding using audio steganography and audio cryptography. International Journal of Computer Applications, 112(5), 49-53.
20. Bandyopadhyay, S. K. & Banik, B. G. (2012). Multi-level steganographic algorithm for audio steganography using LSB modification and parity encoding technique. International Journal of Emerging Trends and Technology in Computer Science, 1(1), 71-74.