The Progressive Multilevel Embedding Method for Audio Steganography

Ahmed A. Alsabhany¹, Farida Ridzuan¹,² & A. H. Azni¹,²

¹Faculty of Science & Technology, Universiti Sains Islam Malaysia (USIM), Nilai, 71800, Malaysia
²CyberSecurity and System Research Unit, Islamic Science Institute (ISI), Universiti Sains Islam Malaysia (USIM), Nilai, 71800, Malaysia

farida@usim.edu.my

Abstract. Audio steganography refers to hiding a secret message inside a cover audio file. The existing methods follow almost a sequential embedding which leads to poor utilization of the cover. The proposed method aims to achieve the maximum feasible capacity of the audio file without lowering the quality. The proposed method is based on seven levels of thresholds and six LSB layers, while the embedding is carried out progressively. The method allocates the message bits over the cover signal horizontally to ensure fair payload distribution over the signal. Hence, the selected samples might be visited multiple times till the message is totally embedded. The experimental results show that the method is able to reach a hiding capacity of 40% of the file size or 266.6 kbps while maintaining an SNR and PSNR values of 40 and 58 dB respectively. It is concluded that the proposed progressive method is highly adaptive to the message size and capable of providing high-quality stego even when big payloads are used. The comparative study shows that the progressive method provides the best message distribution based on the nature of the cover.

1. Introduction

Steganography is a method of data security through obscurity. Steganography is the art and science of hiding data inside a cover file such that only the sender and the receiver can detect the existence of the secret communication. Several studies proposed methods based on two main components which are cryptography and steganography. Cryptography is defined as the science of secret writing with the goal of hiding the meaning of a message. Image, audio, and video files can be used as cover files. Audio steganography is concealing a secret message in an audio file.

An audio steganography method is evaluated based on three main parameters which are (1) capacity, (2) imperceptibility, (3) and robustness [1]–[6]. Capacity refers to the amount of secret data that could be embedded inside a cover file, imperceptibility refers to the degree of secrecy that the embedded message retains, (i.e. the difference between the audio file before embedding the secret message and after should be negligible), and robustness refers to the ability of the embedded message to resist the attacks [3], [5], [7], [8]. An inverse relationships exist among these parameters [9]–[11], where an increment of the bit rate (capacity) may lower the quality of the stego (transparency) and decrease the robustness level while an enhancement of the qualitative parameters may decrease the capacity. These tradeoffs render any static design of the steganography method inefficient. On the other hand, an adaptive design that capitalizes on the dynamic embedding
parameters is more efficient. An interesting issue for steganography is to determine the highest load of data that can be embedded in an audio file while preserving the quality and maintaining a high level of robustness for the embedded message.

In recent years, many ideas were proposed to embed a message in an audio file. Several methods such as those proposed in [12]–[14] were designed based on the idea of embedding the message in the audio samples that obtain high amplitude values (loud) and to avoid embedding in the silent or near to silent intervals of the audio signal. The idea of avoiding embedding in the silent intervals originally aims to improve the imperceptibility by removing the added noise from the silent intervals in the audio, and therefore minimizing detectability. A crucial factor for those methods is the selected embedding threshold. Lowering the embedding threshold may lead to some noticeable noise in the intervals that is near to silent in the audio. While high thresholds lower the capacity since it eliminates more samples of embedding. The problems of the existing methods are the lack of adaptivity and balance, which will be explained next.

2. Lack of Adaptivity
The adaptivity is related to dynamic embedding variables of message size and cover length, and the ability of the method to achieve high transparency dynamically while maintaining high overall capacity in all embedding scenarios. The problem occurs when the method statically assign a fixed number of bits in one sample for embedding. For example, Ahmed et al. (2010) allocates 8 bits of each selected sample for embedding, while Wakiyama (2010) embeds only 1 or 2 bits per sample. High LSB number increases the capacity and lowers the transparency, while low LSB number decreases the capacity and improves the transparency. This issue introduces the degree of modification as a significant variable that affects the transparency, where it equals the decimal amount of error incurred on the sample for each message bit embedding. For example, if a method uses only the 4th LSB for embedding, then the degree of modification equals \(2^3 = 8\). Following up on our examples, the average Degree of Modification (DOM) for the method proposed by Ahmed et al. (2010) equals:

\[\text{Average } \text{DOM}_{[11]} = \frac{1 \times 2^0 + 1 \times 2^1 + 1 \times 2^2 + 1 \times 2^3 + 1 \times 2^4 + 1 \times 2^5 + 1 \times 2^6 + 1 \times 2^7}{8} = 31.8\]

While the ADOM for the method proposed by the method in Wakiyama (2010) equals:

\[\text{DOM}_{[13]} = \frac{1 \times 2^0 + 1 \times 2^1}{2} = 1.5\]

In terms of capacity, to embed 16 bits using the first method uses 2 samples, while the second method will use at least 8 samples. As demonstrated, the high degree of modification causes higher damage to the signal as in the first example, while allocating low LSB for allocation improves the transparency and decreases the capacity.

3. Lack of Balance.
The balance issue is related to the distribution of error over the cover signal. The security risk here is dependent on steganalysis methods’ ability to separate noised (embedded) and clean intervals of the stego, which has been covered in more details by Fridrich & Goljan (2002) and Westfeld (2001). The imbalance load distribution occurs at its best in the cover underloading cases; when the message size is lower than the maximum capacity of the method for a certain cover signal. The overloading and underloading issues have been discussed by Kaur et al. (2017). Many existing methods are based on sequential embedding such as in the method by Cvejic & Seppanen (2002), which operates by embedding 4 LSBs in each sample sequentially. Threshold based methods operate by selecting audio samples for embedding based on the appointed threshold. The method by Ahmed et al. needs only 2 samples to embed 16-bits which leaves the error condensed in a short interval. While, in the method by Wakiyama 8 samples are needed to embed 16-bits, which achieves more spread error and therefore better transparency. In this context, thresholding methods achieve better spreading of error than fully sequential methods. However, as noticed, even exiting
thresholding methods suffer from imbalance error distribution, which is aimed to be improved in this research.

To summarize, existing thresholding methods are based on (1) one or two fixed thresholds and (2) fixed degree of modification, which are critical to the capacity and transparency. As a result, if the threshold is high:

(a) many samples will be eliminated thus lowering the capacity, and improving the transparency, and
(b) more spread embedding and therefore more balanced distribution.

While if the threshold is low:

(a) more samples will be included thus increasing the capacity, but it will lower the transparency, and
(b) more condensed embedding and therefore less balanced distribution.

A fixed degree of modification renders an efficient embedding in terms of transparency. High DOM causes unnecessary damage on the cover in underloading cases where the error can be distributed in more balanced fashion. While, low DOM lowers the capacity. These issues are caused by the fixed and un-adaptive selection of thresholds and DOMs. Better adaptive and balanced methods use the message size and the cover length dynamically to achieve the highest performance in terms of capacity-transparency tradeoff, and achieve better masking where the message is distributed fairly over the cover signal.

In the following paragraphs, some related methods are reviewed.

A method proposed by Cvejic & Seppanen (2002) operates by substituting 4 message bits at 4 LSBs in each sample until the whole message is embedded. The method is fully sequential and causes error condensing at the first intervals of the signal.

A method proposed by Ahmed et al. (2010) avoids embedding in the silent or near to silent periods of the host audio signal. The method uses a noise logic gate to control the embedding process. The embedding occurs only if the sample amplitude is equal to or higher than the selected threshold. The method uses eight layers of LSB’s for embedding, which significantly reduce the quality of the stego. The method uses only one threshold which negatively affects the capacity.

A method was proposed by Srivastava & Rafiq (2011) based on two ideas, which are: (1) the amplitude thresholding and (2) pattern matching embedding. The samples that have a value below the designated threshold will be considered as silent samples, and therefore are excluded from the embedding. Only the samples above that threshold are used for embedding. Three message bits are compared with the sample MSBs, and based on the matching results the LSBs are modified. The disadvantage here is a high rate of mismatch cases which incurs many un-necessary embeddings. As a result, the total number of modified bits in the cover is much higher than the actual message, and therefore capacity is reduced.

Two methods were proposed by Wakiyama et al.(2010), which are based on the amplitude value of the sample. The first method selects two thresholds to avoid embedding in the silent samples. If the amplitude is lower than the lowest threshold, no data will be embedded. If the amplitude is between the thresholds, one bit is used for embedding. If the amplitude is above the highest threshold, two bits are used for embedding. The second method uses the average amplitude of the samples before and after the selected sample. If the amplitude of that sample is higher than the average amplitude, two bits are used for embedding. However, no encryption method was implemented to enhance security. The method lacks an encryption method, which lowers the robustness level.
A method proposed by Vimal & Alex (2014) consists of three stages; (1) Huffman encoding, (2) RSA encryption, and (3) dual randomness embedding technique. In the dual randomness technique, the sample and LSB index are determined randomly based on the MSB in the audio samples. One bit is embedding in one sample and the gap of \((1 - 8)\) samples are left untouched between each embedding. As a result, more balanced distribution is achieved. Due to the high elimination rate and low degree of modification, high transparency and low capacity are achieved.

### Table 1. A Summary of the Related Works

| Author (Year) | Method                                        | Disadvantage                                      |
|---------------|-----------------------------------------------|--------------------------------------------------|
| [15]          | Embeds 4 bits in the 4 LSB of each sample     | Low balance due to the fully sequential embedding |
| [12]          | Uses a noise logic gate to control data embedding. | Low balance and Low adaptivity (low transparancy) due to the severe and abrupt DOM |
| [14]          | The variable low bit coding                   | Low adaptivity (low capacity) caused by the low DOM. |
| [13]          | Thresholding and pattern matching             | Low adaptivity (low capacity) caused by the mismatch cases embedding |
| [7]           | Huffman encoding, RSA, and dual randomness to select sample number and bit index | Low adaptivity (low capacity) due to the DOM and the high sample elimination rate |

### 4. The Proposed Method

The proposed method is developed based on the idea of embedding in the samples of high amplitude and the progressive embedding. To achieve the maximum capacity while maintaining the highest level of secrecy, a multilevel method is proposed to utilize the concept. This method comprises three main phases which are (1) Huffman encoding, (2) AES encryption, and (3) the embedding method. The method is capable of embedding any digital file format inside an audio file. The proposed progressive method uses uncompressed audio files (.wav) with CD-quality (44100 samples/second; each is encoded in 16 bits). Figure 1 shows the encoding process comprises of three phases which are Huffman Encoding, AES encryption, and the multilevel embedding method.

**Fig. 1. The Encoding Process.**

**Huffman Encoding**

In this phase, the data are grouped as a stream of bytes. Then the ASCII representation of each byte is retrieved. Next, the Huffman encoding method is performed on the ASCII characters. Huffman encoding provides lossless compression, which improves the capacity of the overall method. Before forwarding the result to the next phase, the method regroups the results in a stream of bytes.
Advanced Encryption Standard (AES)

In this phase, AES is performed on the message based on a 256-bit key, which is agreed on among the sender and recipient. AES provides a high level of security and speed compared with other encryption methods [16]. The resulted cipher is then converted back into an array of bits $T_i$. Finally, the total number of bits (the array length) is calculated and added to 50 bits. A number of 50 bits size is large enough to represent the message length. Then the array size is incremented by 50 locations, the array is shifted by 50, and then the message length is inserted without being encrypted in the first 50 bits of the array. This process is included to insert the message length at the beginning of the hidden message to enable a smooth extraction process.

The Progressive Multilevel Embedding Method

The unique idea in this method is that the embedding is carried out progressively. To maintain quality, low degrees of embedding are prioritized, namely, the method starts from 1st LSB up to the 6th LSB layer. Moreover, in each LSB layer, seven thresholds are used to select the embedding samples. However, again the method prioritizes the highest amplitude samples based on the amplitude. As a result, 42 embedding cases are generated. This method reduces the effect of the sequential embedding behavior and provides better utilization of the horizontal space in the cover signal. Moreover, the method aims to achieve more adaptive and dynamic DOM, and more balanced and spread payload over the cover signal.

The levels in the methods are the amplitude threshold levels. The method embeds 6 bits maximum into an audio sample of 16 bits. The position on which the embedding occurs is denoted by $q$. In this phase, seven thresholds are considered for embedding. A key aspect in this method is that it prioritizes the samples of highest amplitude for embedding before the samples with lower amplitude values. This process is carried out to ensure that small messages will secure the most camouflaged positions in the audio signal. At each level $k$, the lower threshold is calculated by setting the $k$th position to 1 and all other bits to 0. The resulted amplitude thresholds are 32768, 16384, 8192, 4096, 2048, 1024, 512 in amplitude. Figure 2 shows examples of threshold calculation when $k$ equals 15, 11, and 9. The lower threshold $LT$ equals $2^k$, where the upper threshold $UT$ equals $2^{k+1}$. The maximum lower threshold is the 15 and at each step, the counter is decreased by 1, till $k = 9$ where the lower threshold equals 512 amplitude. Samples with amplitude values below 512 are considered silent or near to silent samples, therefore they are excluded from embedding. For security reasons, a secret pre-shared constant $c$ is added to all thresholds values to confuse attackers.

![Fig. 2. Threshold Calculation.](image-url)

The decimal representation of a sample $j$ is the amplitude value $AV_j$. At each level of the seven levels, the method compares $AV_j$ with $LT$ and $UT$, and only if $AV_j$ is more than or equal to $LT$ and less than $UT$ the method substitute one LSB at a time, in the position $q$, with one bit of the encrypted message. The method is adaptive to message size if all the levels are satisfied and the message is not fully embedded yet, the method increases the value of $q$ (the depth of embedding) by one. In more details, the method first attempts to embed the message in the first LSB ($q = 0$) of the audio samples, which obtain decimal values between 32768 to 65536, then if the message was not fully embedded,
the level is decreased from 16384 to 32768. The method keeps decreasing the level until \( k = 9 \) (512), the minimum range is (512-1024). If the message is still not fully embedded, the method resets the levels and increase the \( q \) value by 1, then the method starts embedding in the second LSB of \( q = 1 \) of the first level (32768-65536), the method keeps increasing the \( q \) value until it is equal to 5 (six LSB). If the message is still not fully embedded, the embedding process fails because the message is too big. The behavior of the method provides a unique scrambling mechanism, which increases the security of the embedded message. Table 2 shows the set of the parameters used in the method.

**Table 2. The Method Parameters**

| Symbols | Description |
|---------|-------------|
| \( q \) | the layer of embedding |
| \( T \) | the bit array of the encrypted message |
| \( m \) | the length of \( T \) |
| \( i \) | the index of an entry in the bit array \( T \) |
| \( n \) | the total number of samples, |
| \( k \) | the level of amplitude, |
| \( j \) | the audio sample number, |
| \( c \) | secret pre-shared constant |
| \( AV_j \) | the decimal representation of \( j \), |
| \( LT \) | the lower threshold, where \( LT = 2^k + c \) |
| \( UT \) | the upper threshold, where \( UT = 2^{k+1} + c \) |

The pseudo code for the method is shown in Figure 3. The outer loop is responsible for the embedding layer or embedding bit index. First, the method embeds in the first LSB where \( q = 1 \), then at each increment, the layer is increased until the 6th LSB where \( q = 6 \). In the intermediate loop 7 levels are used to embed the message, in this loop the threshold values are calculated. Finally, the inner loop visits all audio samples to evaluate the embedding condition. If the value of the sample satisfies the condition the embedding will occur and the counter is set to point to the next bit of the message array.

```plaintext
i = 1
for \( q = 1; q < 7; q++ \) {
    if (i > m) {break;}
    for \( k = 15; k > 8; k-- \) {
        if (i > m) {break;}
        LT = \( 2^k \) + c;
        UT = \( 2^{k+1} \) + c;
        for \( j = 1; j < n; j++ \) {
            if (i > m) {break;}
            if \( AV_j >= LT \) \&\& \( AV_j < UT \) {
                \( T[i] = \sum \)
                \( i++ \);
            }
        }
    }
}
/*Initial counter value*/
/*Outer Loop for bit index*/
/*Breaking point of the Outer Loop*/
/*Intermediate Loop for amplitude level*/
/*Breaking point of the Intermediate Loop*/
/*Upper threshold calculation*/
/*Inner Loop to visit samples*/
/*Breaking Point of the Inner Loop*/
/*Embedding condition*/
/*Embedding by substitution*/
/*Counter increment*/
/*report q value and prompt the user to continue or not.*/
```

**Fig. 3. The Embedding Pseudo Code**

The progressive embedding achieves more adaptive DOM since the message size and the cover signal length dynamically decide the number of bits to replace in the selected samples. Moreover,
the progressive embedding achieves more spread and balanced masking than sequential embedding methods. For example, the first bit of the binary stream of the letter A is embedded in the sample number 1000, and the next sample in the same level is sample number 3000 where the second bit is embedded. Moreover, the lower amplitude levels may embed data between those samples. In addition to that, the same samples that have been used to embed in the LSB $q = 1$ are revisited and reused to embed up to the 6th LSB $q = 6$. A successful attacker should be able to determine the exact values of each threshold in order to recover the message.

The extraction operation is the exact inverse, instead of writing into the audio sample, the method reads and reconstructs the array of bits. The only difference is that after extracting the first 50 bits which indicate the length of $T$, the method stops and modify the value of $m$ to the remaining bits by subtracting 50 from the decimal value represented in the first 50 bits. Figure 4 shows the priority of embedding and based on the LSB layer and the amplitude levels order.

5. Results and Discussion

In order to measure the produced capacity and audio quality PSNR, SNR, and MSE are used to report the experimental results of the proposed progressive method. The environment used is Matlab R2014b. The audio samples used are in 44.1 kHz mono, with 16-bit depth. The audio samples are
selected randomly and cover a variety of genres. The samples include music, sound effects, and speeches. The samples are a maximum of 12 seconds to help capture the performance of the method precisely. Commonly, PSNR is used to assess the quality of the stego. If the value of the PSNR is more than 36 dB, then the produced stego file is indistinguishable from the original audio by the human ear [17]. Kaur et al. used a fixed payload of size 640 bits per second. However, in this analysis, three message loads are used to capture the potential, the performance, and the adaptivity of the proposed progressive method under low and high loads of the method. The message loads used are of size 5000, 25000, and 100000 bytes. Moreover, the message is of type text and randomly generated (Lorem ipsum text).

Figure 5 shows an audio signal before and after the embedding. The size of the used message is 5000 bytes. The length of the used sample is one second. The signals show that there is no clear distortion occurred after embedding, which indicates high transparency.

In the next tables, experimental results are carried out on 10 cover signals of both music and speech types. Moreover, basic cover details and the results in terms of SNR, PSNR, and MSE are featured in the tables. Table 3 shows the experimental results when the message size is 5000 bytes. The average SNR and PSNR for the selected samples is 80.8 dB and 101.8 dB, respectively, which indicates high-quality stego signal with approximately no distortion. It can be observed that when the message size is increased, SNR and PSNR values will start to drop gradually.

![Audio Signal](image)

**Table 3.** 5000-Byte Result

| File | Length (S) | Size (KB) | SNR (dB) | PSNR (dB) | MSE          |
|------|------------|-----------|----------|-----------|--------------|
| #1   | 5          | 475       | 78.91    | 101.20    | 7.5888e-11   |
| #2   | 9          | 778       | 86.29    | 103.31    | 4.6609e-11   |

Fig. 5. The Audio Sample Before and After Embedding.
Table 4 shows the experimental results when the message size is 25000 bytes. It is noticeable that the average SNR and PSNR have decreased to 72 dB and 93 dB, respectively. However, these values are still very high and indicate excellent stego quality.

Table 4. 25000-Byte Results

| File | Length (S) | Size (KB) | SNR (dB) | PSNR (dB) | MSE          |
|------|------------|-----------|----------|-----------|--------------|
| #1   | 5          | 475       | 71.27    | 93.55     | 4.41E-10     |
| #2   | 9          | 778       | 79.28    | 96.31     | 2.34E-10     |
| #3   | 3          | 249       | 69.14    | 87.94     | 1.61E-09     |
| #4   | 6          | 516       | 76.49    | 94.53     | 3.53E-10     |
| #5   | 4          | 358       | 69.12    | 90.65     | 8.62E-10     |
| #6   | 10         | 861       | 79.40    | 96.75     | 2.11E-10     |
| #7   | 3          | 319       | 67.94    | 89.12     | 1.22E-09     |
| #8   | 12         | 1000      | 73.55    | 97.48     | 1.78E-10     |
| #9   | 8          | 689       | 69.54    | 95.77     | 2.65E-10     |
| #10  | 5          | 485       | 75.87    | 94.259    | 3.76E-10     |
| Average | 6.5      | 573       | 72.88    | 93.63     | 5.75182E-10  |

Table 5 shows the experimental results when the message size is 100000 bytes. The average SNR and PSNR values are 54.5 dB and 75.2 dB. The PSNR values are still above 55 even when the message size is up to 40% of the cover audio in some cases. The quality is preserved for all used samples. By listening to both samples before and after embedding, a slightly indistinguishable difference or even no difference at all is reported, which indicates high quality and imperceptibility levels.

Table 5. 100000-Byte Results

| File | Length (S) | Size (KB) | SNR (dB) | PSNR (dB) | MSE           |
|------|------------|-----------|----------|-----------|---------------|
| #1   | 5          | 475       | 51.73    | 74.016    | 3.97E-08      |
| #2   | 9          | 778       | 67.34    | 84.37     | 3.66E-09      |
| #3   | 3          | 249       | 39.65    | 58.44     | 1.43E-06      |
| #4   | 6          | 516       | 60.89    | 78.93     | 1.28E-08      |
| #5   | 4          | 358       | 47.4     | 68.93     | 1.28E-07      |
| #6   | 10         | 861       | 69.54    | 86.89     | 2.05E-09      |
| #7   | 3          | 319       | 38.96    | 60.14     | 9.69E-07      |
| #8   | 12         | 1000      | 60.16    | 84.1      | 3.89E-09      |
| #9   | 8          | 689       | 51.64    | 80.63     | 8.65E-09      |
| #10  | 5          | 485       | 58.07    | 76.46     | 2.26E-08      |
Figure 6 shows the PSNR values for the message loads and the selected files. The first observation indicates that for all files used, the proposed progressive method is able to produce high-quality stego files under various message sizes. However, it is important to notice that in file #3 and file #7, in the 10000-byte series, the SNR, and PSNR values are more sensitive than others due to the short length of these signals. These two files highlight how the method prioritizes the horizontal embedding and error-scattering over sequential embedding since the SNR and PSNR values show lower levels only when the number of samples is limited. The justification is that as the depth of embedding increases, the error power will be increased. For example, in a sample of decimal value of 34055, flipping the LSB \((q = 1)\), produces an error value of 1. However, flipping the LSB at \(q = 2, 3,\) or 4 produces a more critical error. Based on Table 4 and Figure 6, the embedding rate in file #3 is 33.3 KBps or 266.6 Kbps, with an SNR and PSNR values of 40 and 58 dB respectively. The progressive method is more cautious regarding modifying deeper layers than existing methods, such as those proposed in [12], [13], [15], [18], [19], as the deepest layer is 6 and the embedding in that layer is left as the last choice. Hence, a conclusion could be made that the proposed progressive method is highly adaptive to the message size and capable of providing high-quality stego audios even when big payloads are used. Moreover, the progressive method achieves more spread and balanced embedding than most existing methods since both embedding locations and the bits order are generated randomly based on the dynamic message and cover. Therefore, it

![Fig. 6. The PSNR Values for Three Message Loads](image)

The performance results of the proposed progressive method are compared with two other existing methods, which are those proposed in [11], [12]. These methods are chosen because both of them include fixed threshold methods that can be recreated and evaluated. The experiment used 1-second audio file to capture the slight differences. The audio file is sampled at 44.1 kHz, 16 bits per sample, and of type music. At all points of the experiment, an identical copy of the message with no compression were used. The SNR and PSNR were used to capture the results. 8 bits per sample are modified and the embedding threshold is set to 512 amplitude to replicate the method proposed in [11]. While for the method proposed in [12], 2 bits per sample are modified and the embedding threshold is set to 2000 amplitude. Figure 7 and 8 show the SNR and PSNR levels, respectively, for the three methods under various payloads.
Fig. 7. The SNR Levels for the Proposed Progressive Method against Related Works for Various Message Loads.

In Figure 7 and 8, the 10000 Bytes and 15000 Bytes payloads were above the capacity limits for the method proposed in [13]. Therefore, the embedding fails and no values were extracted. Here it is observed that eliminating a big portion of audio samples from being used for embedding leads to capacity loss. However, it is clear that the proposed progressive method achieved better results in terms of both SNR and PSNR over the other two methods at all points of the experiment. It is worth to highlight that the method proposed in [13] modifies only 2 bits per selected sample and yet the proposed progressive method achieved better audio quality. This is due to the adaptive DOM in the progressive method; where it only uses the first LSB if the message is small enough while the other method modifies two bits per sample.

To measure the balance of embedding and the error-scattering ability, another experiment is carried out to highlight how each method utilizes the horizontal space of the audio signal. In this experiment, a small payload is used to capture how far, in sample numbers, each method will reach before hiding the message. The audio signal is of 1 second length and 44100 samples. The payload is 1000 byte. Figure 9 shows the number of samples visited for hiding the message in each method. In the method proposed in [12], only 1994 out of 44100 samples were used to hide the message, which means that the error is concentrated in the first 4% segment of the audio. In addition to the lowered transparency, the condensed embedding increases its detection accuracy against statistical steganalysis methods [20]. In the method proposed in [13], better utilization of the horizontal space is achieved. The number of samples visited is almost 8000, which represents 18% of the total signal length, and that is justified by the higher threshold used in the method. However, the proposed progressive method obtained 36%, doubling the ratio of method counted in [13]. This result proves that the proposed method achieves better error-scattering and cover length utilization and therefore more balanced embedding.

Fig. 8. The PSNR Levels for the Proposed Progressive Method against Related Works for Various Message Loads.
To summarize the above discussion, it has been showed that the proposed progressive method is highly adaptive to the message size and capable of providing high-quality stego audios even when big payloads are used. Based on the comparison study, the proposed progressive method provides better efficiency in both low and high payloads, due to its high adaptivity to variable message sizes. Moreover, the proposed progressive method achieves better error-scattering ability and utilization of the horizontal space which improves both the imperceptibility and the robustness against attacks.

6. Conclusions

In this paper, a progressive multilevel embedding method was proposed. The method was designed to achieve adaptivity. The adaptivity objective is related to the ability of the method to dynamically achieve high performance depending on the dynamic embedding variables, such as the message size and the signal length. The progressive method uses an adaptive DOM to counter the fixed designs that uses one DOM and one static embedding threshold. Moreover, the proposed progressive method is more efficient than the related methods, which are based on one or two thresholds in terms of capacity and quality of the audio. The produced method uses seven thresholds of embedding to handle the audio samples properly. The progressive embedding prioritizes lower degrees of modifications and samples with the highest amplitudes to provide the maximum camouflage for the message. Then higher degrees of modifications are triggered when needed. The experimental results show that the method efficiently utilizes the capacity in the audio samples while maintaining high quality even when big payloads are used. Based on the comparison, the proposed progressive method provides better efficiency in both low and high payloads, due to its high adaptivity to variable message sizes. Moreover, the comparison results show that the proposed progressive method achieves better error-scattering ability and utilization of the horizontal space, and therefore more balanced and secure embedding against steganalysis attacks.

Furthermore, the employment of a lossless compression method which is the Huffman-encoding increases the capacity even more. On the security side, the sample selection criteria provide a unique scrambling mechanism that is generated randomly based on the nature of the audio. The scrambling mechanism is reinforced with AES-256 encryption.

Acknowledgment

The authors would like to express their gratitude to Ministry of Education Malaysia, Universiti Sains Islam Malaysia and Islamic Science Institute (ISI), USIM for the support and facilities provided. The research paper is supported by grant number USIM/FRGS/FST/055002/50117.

References

[1] M. L. M. Kiah, B. B. Zaidan, A. A. Zaidan, A. M. Ahmed, and S. H. Al-bakri, 2011, “A review of audio based steganography and digital watermarking,” Int. J. Phys. Sci., vol. 6, no. 16, pp. 3837–3850.
[2] P. S. Sapra and H. Mittal, 2016, “Secured LSB Modification using Dual Randomness,” in 2016 International Conference on Recent Advances and Innovations in Engineering,
ICRAIE 2016, pp. 23–26.

[3] F. Djebbar, B. Ayad, K. A. Meraim, and H. Hamam, 2012, “Comparative study of digital audio steganography techniques,” EURASIP J. Audio, Speech, Music Process., vol. 2012, no. 1, pp. 1–16.

[4] F. Djebbar, B. Ayad, H. Hamam, and K. Abed-Meraim, 2011, “A view on latest audio steganography techniques,” 2011 Int. Conf. Innov. Inf. Technol. IIT 2011, pp. 409–414.

[5] A. Kaur, M. K. Dutta, K. M. Soni, and N. Taneja, 2017, “Localized & self adaptive audio watermarking algorithm in the wavelet domain,” J. Inf. Secur. Appl., vol. 33, pp. 1–15.

[6] A. H. Ali, “A HIGH CAPACITY AUDIO STEGANOGRAPHY MODEL BASED ON FRACTAL CODING.”

[7] J. Vimal and A. M. Alex, 2014, “Audio steganography using dual randomness LSB method,” in 2014 International Conference on Control, Instrumentation, Communication and Computational Technologies, ICCICCT 2014, pp. 941–944.

[8] M. Asad, J. Gilani, and A. Khalid, 2012, “Three layered model for audio steganography,” Proc. - 2012 Int. Conf. Emerg. Technol. ICET 2012, pp. 270–275.

[9] K. Gopalan and Q. Shi, 2010, “Audio steganography using bit modification - A tradeoff on perceptibility and data robustness for large payload audio embedding,” in Proceedings - International Conference on Computer Communications and Networks, ICCCN.

[10] K. Gopalan and J. Fu, 2015, “An imperceptible and robust audio steganography employing bit modification,” Proc. IEEE Int. Conf. Ind. Technol., vol. 2015-June, no. June, pp. 1635–1638.

[11] Z. Mazdak, B. A. M. Azizah, M. A. Shahidan, and S. C. Saman, 2012, “Mazdak Technique for PSNR Estimation in Audio Steganography,” Appl. Mech. Mater., vol. 229–231, pp. 2798–2803.

[12] M. A. Ahmed, M. L. M. Kiah, B. B. Zaidan, and A. A. Zaidan, 2010, “A Novel Embedding Method to Increase Capacity and Robustness of Low-bit Encoding Audio Steganography Technique Using Noise Gate Software Logic Algorithm,” J. Appl. Sci., vol. 10, no. 1, pp. 59–64.

[13] M. Srivastava and M. Q. Rafiq, 2011, “A Novel Approach to Secure Communication Using Audio Steganography,” Adv. Mater. Res., vol. 403–408, pp. 963–969.

[14] M. Wakiyama, Y. Hidaka, and K. Nozaki, 2010, “An audio steganography by a low-bit coding method with wave files,” in Proceedings - 2010 6th International Conference on Intelligent Information Hiding and Multimedia Signal Processing, I IHMSP 2010, pp. 530–533.

[15] N. Cvejic and T. Seppanen, 2002, “Increasing the capacity of LSB-based audio steganography,” Proc. 2002 IEEE Work. Multimed. Signal Process. MMSP 2002, pp. 336–338.

[16] C. Paar and J. Pelzl, 2010, Understanding Cryptography: A Textbook for Students and Practitioners. Springer–Verlag.

[17] S. Saraireh, 2013, “A Secure Data Communication System Using Cryptography and Steganography,” Int. J. Comput. Networks Commun., vol. 5, no. 3, pp. 125–137.

[18] M. Bazyar and R. Sudirman, 2015, “A new method to increase the capacity of audio steganography based on the LSB algorithm,” J. Teknol., vol. 74, no. 6, pp. 49–53.

[19] A. Delforouzi and M. Pooyan, 2009, “Adaptive and efficient audio data hiding method in temporal domain,” ICICS 2009 - Conf. Proc. 7th Int. Conf. Information, Commun. Signal Process., pp. 0–3.

[20] D. M. Ballesteros L and J. M. Moreno A, 2012, “Highly transparent steganography model of speech signals using Efficient Wavelet Masking,” Expert Syst. Appl., vol. 39, no. 10, pp. 9141–9149.