Journal of Big Data

RESEARCH Open Access

Reversible data hiding with segmented secrets and smoothed samples in various audio genres

Tohari Ahmad* and Yoga Samudra

*Correspondence:
tohari@if.its.ac.id
Department of Informatics, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia

Abstract
In this age, information technology has grown significantly. Computer networks, which connect a device to others, have made it easier for people to transfer data than before. Moreover, smart devices have the capability of supporting this data transmission, including those in the cloud that may contain massive data. However, the security factor can be a severe issue if sensitive big data, such as military and medical data, do not have enough protection. Furthermore, an attacker may be able to disclose such data. Some algorithms have been introduced to solve that problem, one of which is the data hiding method. Nevertheless, some factors are still challenging, concerning the capacity of the secret data and the quality of the generated data, which are represented by bit and Peak Signal-to-Noise Ratio (PSNR), respectively. Besides, some techniques are not reversible, which means that they cannot reconstruct the carrier (cover). In this research, we investigate those problems by taking audio as the carrier. It is done by sampling the audio file before being interpolated to present spaces for accommodating the secret. Meanwhile, the secret is segmented before the embedding. Later, the embedded audio is smoothed according to the required level. The experimental result is obtained by using a public data set containing various audio genres and instruments, and 11 secret sizes, from 1 to 100 kb. It shows that the proposed method outperforms the others. This higher PSNR value means that the proposed method can generate more similar stego data; it also implies that at a certain quality level, the number of bits that can be hidden in the audio cover is higher than that of others.

Keywords: Audio processing, Confidentiality, Data hiding, Information security, Secret data

Introduction

Security has been an essential factor in this cyber age. It is because the internet connection, which initially may be designed without much worrying about bad users, has been suffering from various types of attacks [1, 2]. Among the security properties that should be protected, confidentiality is often the primary target. Once these confidential data have been compromised, the attacker may disclose them, breaching the users privacy [2–4]. Nevertheless, it does not mean that other properties like integrity, availability, or non-repudiation are safe from disruption. It is even possible that an attack is launched to compromise some properties at once, such as in [5], where several security properties are illegally taken over. Moreover, advanced technology has made limited devices more popular than ever [6], which may attract attackers to access it [1].

In many cases, a system comprising big data has been targeted for attack, which may devastate their availability and integrity [2, 5, 7]. Various methods have been introduced to protect those security properties, specifically the confidentiality. For example, cryptography and steganography, which is also called data hiding, are widely implemented. In this paper, the terms steganography and data hiding are used interchangeably. Although the purpose of cryptography and steganography are the same, those two schemes have different characteristics. In the application level, they can be implemented in a system, such as in [8–13]. However, this combination affects its performance [14]. On its progress, it is also possible to use data hiding for forensics [15] or secret sharing [16]. In general, the concept of data or information hiding can be depicted in Fig. 1, where the secret (payload) is hidden in a cover (carrier) to produce a stego file.

The types of cover vary. It can be an image, video, or audio, where the last is less popular than those two other types [17]. Recently, the text has also been investigated [18] to complement the previous media. In fact, audio has unique characteristics to explore further. The algorithms being applied can be extensions and variations of either image- or video-based methods, such as Echo Data Hiding (EDH) [19] and Histogram Shifting (HS) [20, 21] which is then extended in [22] by applying a statistical histogram. Other examples are Prediction Error Expansion (PEE) [23], Noncausal Prediction (NP) [24], and Difference Expansion (DE) along with its variations, such as Reduced Difference

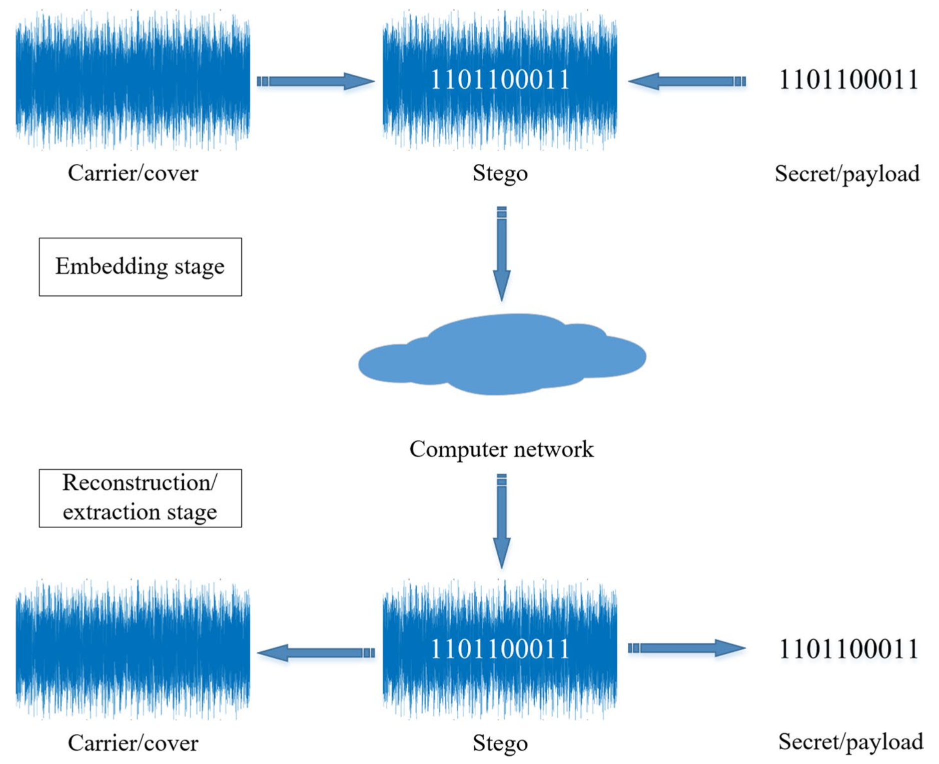

Fig. 1 The overall concept of audio-based data hiding

Expansion (RDE) [25, 26]. An extended method has been developed by implementing machine learning, which is claimed to be suitable for smart devices [27].

Some recent algorithms have been introduced to get effective results. For example, Jung and Yoo [28, 29] propose the use of an interpolation algorithm for the embedding space. Next, Ahmad and Fiqar [30] explore the quality of the stego audio, which is then further refined in [31]. Previously, Andra et al. [32] focused on the protection of medical data, also by using audio, while Bobeica et al. [33] extended the concept of PEE.

Nevertheless, those existing methods still have challenging problems: the capacity of the secret, the quality of the stego, and the reversibility of both the secret and the cover. Capacity refers to the size of the data or the number of bits that can be accommodated by the cover. Data hiding methods may have different capacities when they are implemented to the same cover. Quality refers to the similarity level between the cover and the stego data. Like the capacity factor, the higher the quality, the better the method. As predicted, however, those two factors are typically inversely proportional. In this research, we work on these issues by designing an extended embedding and smoothing approaches in the audio environment. Here, an audio signal is first sampled and normalized before being embedded by the segmented secret. Next, the stego is smoothed to make it as similar to the cover as possible.

The remaining parts of the paper are organized as follows. In "Audio-based data hiding techniques" section, we present the earlier research on data hiding techniques. "The design of reversible data hiding" section describes how the proposed reversible data hiding method works. The experimental results and the corresponding analysis are provided in "Experimental results" section. Moreover, this proposed method is compared with related methods to evaluate its performance. Next, the discussion is presented in "Discussion" section. Lastly, the conclusion of this research is drawn in "Conclusions" section.

Audio-based data hiding techniques

As previously described, audio is not as popular as other media for carrying the secret [17]. Nevertheless, it has unique characteristics, which have the potential to be explored. Some state-of-the-art audio-based research can be described as below.

Depending on its specific purpose, data hiding is designed by considering the basic requirements of security, including its reversibility. In case only the secret can be extracted from the stego, the method is called irreversible; on the other hand, if both the secret and the cover can be reconstructed the same as their initial data, then the method is reversible data hiding (RDH). In most cases, RDH is crucial for fulfilling confidentiality and authentication requirements [34].

The size of the secret can be embedded in a cover is one of the challenges faced by RDH and also represents the possible types of data to protect. Therefore, most RDH methods take it as an essential consideration. To meet this requirement, [28, 29] implement Neighbor Mean Interpolation (NMI) to expand the space. On the other hand, [29] takes various numbers of the bit to hide, while [28] sets it as a fixed number. Next, this feature is adopted in [30] and [31], where respectively Newton's Divided Difference Interpolation (NDDI) and linear interpolation methods are implemented.

The research in [28, 29] pays more attention to the capacity of the secret. Therefore, the method does not do much to improve quality. Nevertheless, its stego quality is relatively good enough that ordinary users may not be able to distinguish noise from the original file. Andra et al. [32] refine this possible quality problem by modifying the intelligent partitioning method before the embedding stage. In this case, a 2D array of data is developed based on the sampled signal. It is done by arranging the audio samples in a particular order. This approach is different from the others, where the audio signal is treated as 1D. For the embedding, [32] employs improved RDE. Their experiment is performed by embedding several sizes of secret medical data into various audio covers. It is shown that their scheme works well.

Similar to [32], the research in [33] does not implement interpolation. Instead, it reserves some audio samples to carry the specific data, where the cross and dot areas are defined at the beginning. It needs to find appropriate threshold values, which are used for the embedding. This stage, however, may not be simple since those values depend on some other factors. Furthermore, as [32], the number of bits to conceal in each sample is static. The experiment shows that their stego quality is relatively stable.

Considering the stego quality, Ahmad and Fiqar [30] introduce the smoothing stage to reduce the difference between the stego and the cover. It is claimed that this scheme is able to work on it, which is proven in their experimental results. Nevertheless, its performance relies on the genre of the audio cover. It can also be an advantage, considering that a user may have a unique preference. In the next research, the noise in the stego audio is decreased by applying multiple-smoothing [31]. Furthermore, an additional step is also implemented, called a reducing step. In most conditions, this method generates a better stego audio.

Generally, this previous research is to be the base of our proposed method. Some stages are taken and refined to get better results. Initially, the NMI and NDDI interpolation algorithms, which are implemented in [28, 29, 30], respectively, have inspired us to use them. Nevertheless, the linear method employed in [31] potentially delivers an equivalent result. Therefore, we take this simpler algorithm in our design. Moreover, [32, 33], which do not take the interpolation at all, may have a relatively lower capacity. Each sample's payload size in [32, 33] is static, which is different from that of [30, 31]. So, the payload's dynamic (non-static) size and the smoothing step have also been our focus in this research. Unlike [31, 32], we integrate it with the reducing step. Briefly, we take a simple interpolation algorithm, improve each sample's dynamic capacity, and design the smoothing to include the reducing step. Accordingly, the extraction and reconstruction follow this embedding. More details of the proposed algorithm and its comparison with the baseline methods are given in "The design of reversible data hiding" section.

The design of reversible data hiding

As shown in Fig. 1, the secret data protection is carried out in two stages: embedding and reconstruction / extraction. These processes can be described as follows.

Embedding stage

In this study, the pre-processing of both the audio cover and the secret is carried out in parallel. Once those data are ready, the embedding stage is performed, followed by

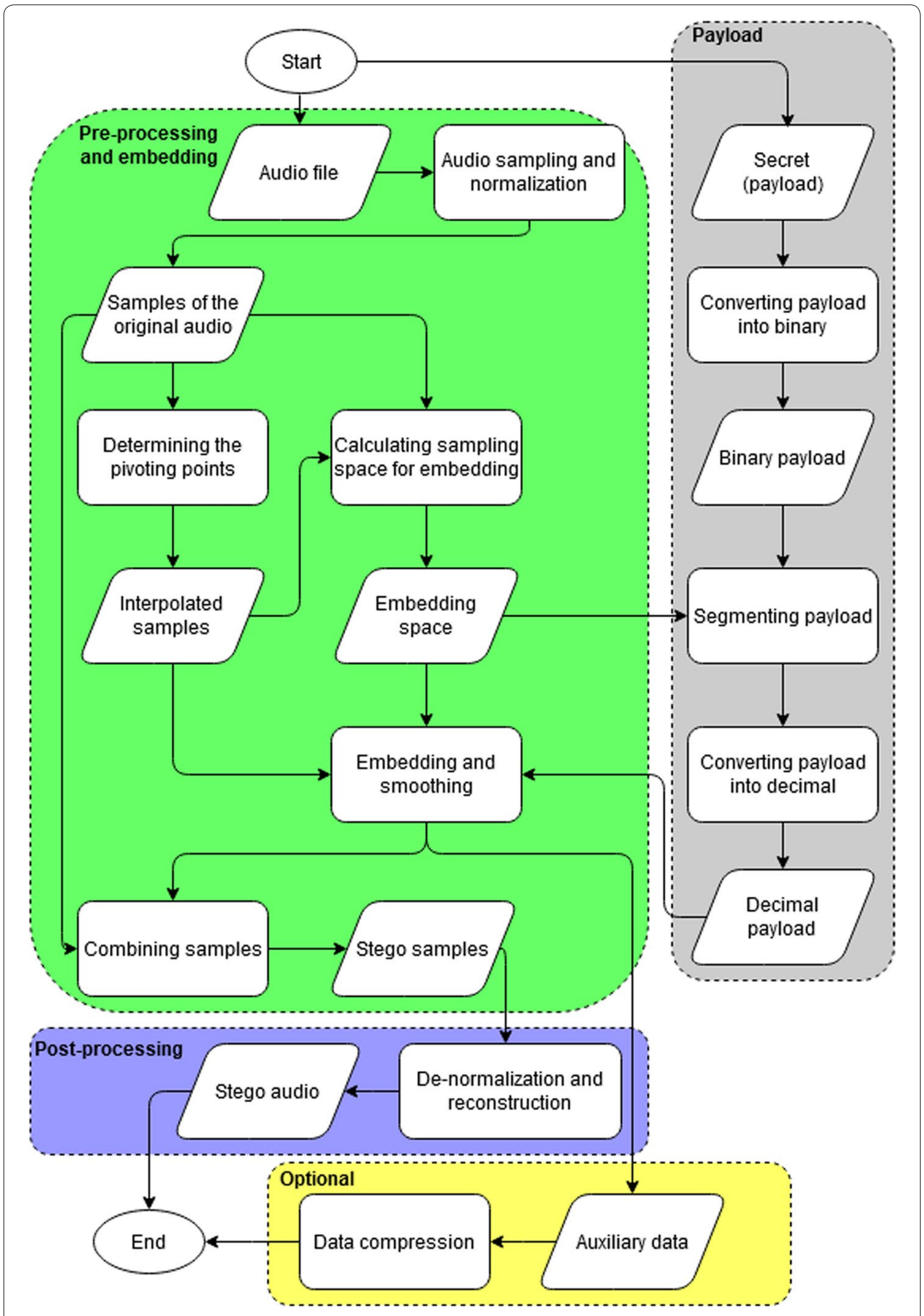
Start
Pre-processing and embedding
Audio file
Audio sampling and normalization
Samples of the original audio
Determining the pivoting points
Calculating sampling space for embedding
Interpolated samples
Embedding space
Embedding and smoothing
Combining samples
Stego samples
Payload
Secret (payload)
Converting payload into binary
Binary payload
Segmenting payload
Converting payload into decimal
Decimal payload
Post-processing
Stego audio
De-normalization and reconstruction
Optional
End
Data compression
Auxiliary data

Fig. 2 Process of embedding the secret in the audio cover

post-processing. This pre-processing step includes binarization and segmentation of the secret and providing an embedding space in each sample, while post-processing includes smoothing and combining samples for preparing the stego audio. Optionally, data compression can be applied to reduce its size. In more detail, this embedding process is depicted in Fig. 2, where its stages can be described as follows.

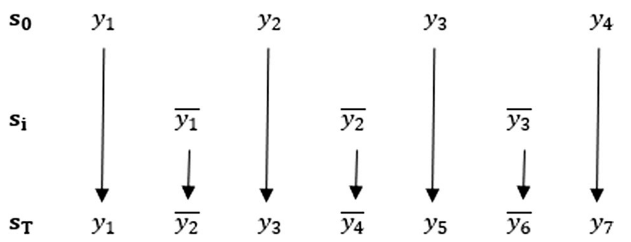

Fig. 3 Illustration of developing an audio vector comprising original and interpolating samples

1. Sampling and normalization. A continue audio signal is discretized to have audio samples, represented by a 16-bit signed integer. Each of these frames is then normalized to an unsigned integer.
2. Interpolation. Let the vector of samples be $\mathbf{s_o} = (y_1, y_2, ..., y_n)$, where y_n is the number of samples. The linear interpolation is applied to $\mathbf{s_o}$ to get a vector of interpolated samples $\mathbf{s_i}$ by using (1), where $\overline{y_j}$ is the interpolating sample resulted from ith and $(i+1)$th samples, where i and j are the index of original and interpolating samples, respectively. The $\overline{y_j}$ is to be a pivoting point where the embedding in the next step will be done. After the interpolation, the vector of samples is $\mathbf{s_T} = (y_1, \overline{y_2}, y_3, \overline{y_4}, ..., y_{n-2}, \overline{y_{n-1}}, y_n)$, whose illustration is depicted in Fig. 3. It is shown that the index of both the original and the interpolating audio samples are shifted.

$$\overline{y_j} = \left\lfloor \frac{y_i + y_{(i+1)}}{2} \right\rfloor \tag{1}$$

3. Calculating sampling space. The number of bits that can be protected by each sample may not be the same. It depends on the characteristics of the cover as well as the sample itself. Based on the vector $\mathbf{s_T}$, firstly, we need to find the average of the difference between two consecutive samples of the original cover ($\mathbf{s_o}$) before being interpolated by using (2). It is worth noting that now there is a sample $\overline{y_i}$ between $y_{(i-1)}$ and $y_{(i+1)}$. Next, the multiplication factor, m, is determined by using (3), where b is the bit-depth of the sample, obtaining from the sampling process in step 1, which in this research is 16. Finally, the sampling space of each sample is found by designing (4), where c_i is the space (maximum capacity) of the interpolating sample $\overline{y_i}$, generated based on the original samples $y_{(i-1)}$ and $y_{(i+1)}$.

$$\bar{x} = \left\lfloor \frac{\sum_{i=1}^{n-1} |y_{(2i-1)} - y_{(2i+1)}|}{n-1} \right\rfloor \tag{2}$$

$$m = \frac{2^b}{(\bar{x})} \tag{3}$$

$$c_i = m \times |y_{(i-1)} - y_{(i+1)}| \tag{4}$$

To find the position of where the embedding should be carried out, it needs to locate the position of the sampling space of each corresponding sample, whether it is above or below the magnitude of the pivoting point. For this purpose, we define p_i and recalculate c_i whenever needed, by designing Algorithm 1.

4. Segmenting payload. Each sample may not fully take the maximum number of payloads. Some factors should be considered, such as the distribution across those interpolating samples, which affects the quality of the generated stego audio file [35]. For this reason, c_i is further processed using (5), where N_i is the number of bits hidden in the corresponding sample.

$$N_i = \begin{cases} \lfloor log_2(c_i) \rfloor \quad if \quad c_i \neq 0 \\ 0 \quad if \quad c_i = 0 \end{cases} \tag{5}$$

At this stage, the binary payload is segmented according to N_i of ith sample. In case it is bigger than the payload to hide, then it is likely that the remaining interpolating samples are not used. Thus, each sample may hold different numbers of bits, which are then converted to decimal (d_i).

5. Embedding. The embedding is performed by adding or subtracting the payload from the interpolating sample, relying on the embedding space's position p_i. This process is depicted in (6). It can be seen that there is a difference between $\overline{y_i}$ and $\overline{y_i'}$, which is resulted from this embedding process. In case the value of d_i is relatively large, the quality of the audio dramatically drops.

$$\overline{y_i'} = \begin{cases} \overline{y_i} + d_i \quad if \quad p_i = 1 \\ \overline{y_i} - d_i \quad if \quad p_i = 0 \end{cases} \tag{6}$$

6. Smoothing. This step aims to reduce the difference caused by the embedding. To make the difference in the sample value between before and after embedding as small as possible, we design a smoothing step which can be formulated in (7), where $\overline{y''_{(i,j)}}$ is the interpolating sample after the jth smoothing is applied to the embedded sample. The smoothing coefficient (e) and the maximum number of smoothing instances (h) should be defined. A larger e means that less h is required to achieve the specified quality; consequently, less computation should be done, which is good. However, it needs more numbers to store for extraction purposes.

$$\overline{y''_{(i,j)}} = \begin{cases} \left\lfloor \frac{(2^e-1)\times\overline{y''_{(i,j-1)}}+\overline{y_i'}}{2^e} \right\rfloor \quad if \quad p_i = 1 \\ \left\lceil \frac{(2^e-1)\times\overline{y''_{(i,j-1)}}+\overline{y_i'}}{2^e} \right\rceil \quad if \quad p_i = 0 \end{cases} \tag{7}$$

Along with reducing of the noise, a remainder (r) is taken by using (8). This value is then required for extraction. It is shown that its value relies on e. It needs to find its appropriate value, which is neither too low nor too high.

$$r_{(i,j)} = \left((2^e - 1) \times \overline{y''_{(i,j)}} + \overline{y_i'}\right) \quad mod \quad 2^e \tag{8}$$

7. Combining samples. The smoothed-embedded sample $\overline{y''_{(i,h)}}$ is then put in $\mathbf{s''_T}$. Therefore, the vector of samples is $\mathbf{s''_T} = (y_1, \overline{y''_{(2,h)}}, y_3, \overline{y''_{(4,h)}}, ..., y_{n-2}, \overline{y''_{(n-1,h)}}, y_n)$, which is then de-normalized to make it in the original range value. Its frame rate is now 88.2kHz.

```
Algorithm 1: Finding the position of the sampling space and updating its capacity
   Input: $\overline{y_i}, c_i$
   Output: $p_i, c_i$
 1 MAX=65535 /* The maximum value of samples */
 2 MIN=0 /* The minimum value of samples */
 3 if $(\overline{y_i} + c_i) \leq MAX$ then
 4     $p_i = 1$
 5 else if $(\overline{y_i} - c_i) \geq MIN$ then
 6     $p_i = 0$
 7 else if $(MAX - \overline{y_i}) \geq (\overline{y_i} - MIN)$ then
 8     $p_i = 1$
 9     $c_i = MAX - \overline{y_i}$
10 else
11     $p_i = 0$
12     $c_i = \overline{y_i} - MIN$
```

Let the binary secret be 1010000111. An example of this embedding stage can be given as follows. Here, the smoothing is only performed once for simplicity ($h = 1$); to achieve better quality, this smoothing step can be done several times. The embedding is applied to the first interpolating sample because its maximum capacity is higher than the number of bits.

- Original samples: $\mathbf{s_o} = (80, 86, 90, 90, 85, 90)$
- Interpolating samples: $\mathbf{s_i} = (83, 88, 90, 87, 87)$
- Combined samples: $\mathbf{s_T} = (80, 83, 86, 88, 90, 90, 90, 87, 85, 87, 90)$
- Multiplication factor: $m = 16384$
- Initial embedding spaces: $c_2 = 98304 \quad c_4 = 65536, \quad c_6 = 0 \quad c_8 = 81920, \quad c_{10} = 81920$
- Updated embedding spaces: $c_2 = 65452 \quad c_4 = 65447, \quad c_6 = 16384 \quad c_8 = 65448, \quad c_{10} = 65448$
- Position: $p_2 = 1, \quad p_4 = 1, \quad p_6 = 1, \quad p_8 = 1, \quad p_{10} = 1$
- Number of bits: $N_2 = 15, \quad N_4 = 15, \quad N_6 = 13, \quad N_8 = 15, \quad p_{10} = 15$
- Segmented payload in decimal: $d_2 = 647, \quad c_4 = 0, \quad c_6 = 0, \quad c_8 = 0, \quad c_{10} = 0$
- Embedded samples: $\overline{y'_2} = 730, \quad \overline{y'_4} = 88, \quad \overline{y'_6} = 90, \quad \overline{y'_8} = 87, \quad \overline{y'_{10}} = 87$
- Resulted vector (before smoothing): $\mathbf{s'_T} = (80, 730, 86, 88, 90, 90, 90, 87, 85, 87, 90)$
- Smoothing embedded samples (with $h = 1) : \overline{y''_2} = 163, \quad \overline{y''_4} = 88, \quad \overline{y''_6} = 90, \quad \overline{y''_8} = 87, \quad \overline{y''_{10}} = 87$
- Remainder: $r_2 = 7, \quad r_4 = -, \quad r_6 = -, \quad r_8 = -, \quad r_{10} = -$
- Final stego samples (after smoothing): $\mathbf{s''_T} = (80, 163, 86, 88, 90, 90, 90, 87, 85, 87, 90)$

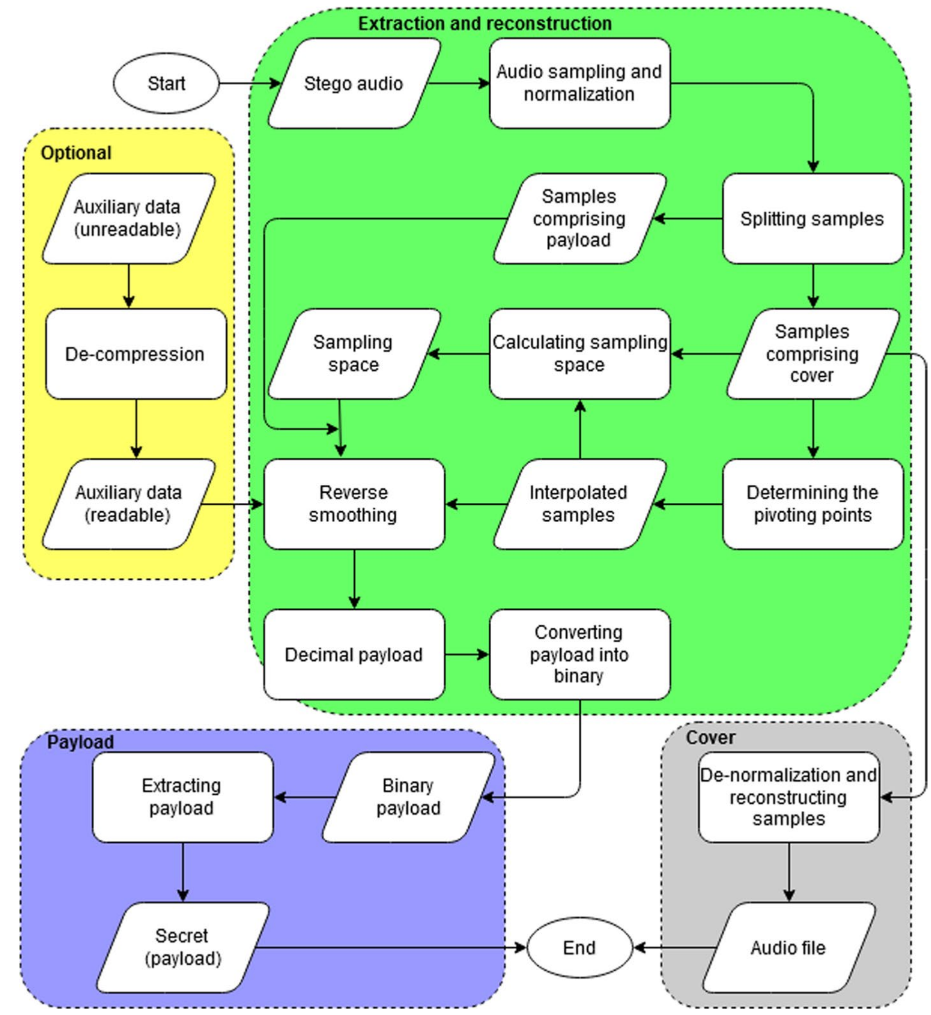
Extraction and reconstruction
Start
Stego audio
Audio sampling and normalization
Optional
Auxiliary data (unreadable)
Samples comprising payload
Splitting samples
De-compression
Sampling space
Calculating sampling space
Samples comprising cover
Auxiliary data (readable)
Reverse smoothing
Interpolated samples
Determining the pivoting points
Decimal payload
Converting payload into binary
Payload
Extracting payload
Binary payload
Cover
De-normalization and reconstructing samples
Secret (payload)
End
Audio file

Fig. 4 Process of extracting the secret and reconstructing the cover from the stego audio

Extraction stage

This stage aims to extract the payload and to reconstruct the cover as the proposed method is designed to be reversible. Parameter values are directly generated from the stego file, such as the sampling space obtained after the original samples have been split from the interpolating ones. This general process is presented in Fig. 4.

Based on the received values, the reverse smoothing is performed by using (9). This process is done as many times as the number of smoothing instances in the embedding stage. Therefore, in the last iteration we have $\overline{y''_{(i,j)}} = \overline{y'_i}$, which is a non-smoothing embedded sample.

$$\overline{y''_{(i,j-1)}} = \begin{cases} (2^e \times \overline{y''_{(i,j)}}) - \left((2^e - 1) \times \overline{y'_i}\right) + r_{(i,j)} & if \quad p_i = 1 \\ (2^e \times \overline{y''_{(i,j)}}) - \left((2^e - 1) \times \overline{y'_i}\right) - r_{(i,j)} & if \quad p_i = 0 \end{cases} \tag{9}$$

The hidden payload of ith sample is obtained by using (10). Along with other payloads, this decimal value is then converted into binary, the number of digits of which is

Table 1 Summary of the data hiding techniques in [28–33] and the proposed method (inspired by [30, 31])

Evaluation factor	Research in [28, 29]	Research in [30]	Research in [32]	Research in [33]	Method of [31]	Proposed method
Cover (carrier)	Image	Audio	Audio	Audio	Audio	Audio
Interpolation	NMI	NDDI	None	None	Linear	Linear
Payload size	Dynamic static	Dynamic	Static	Static	Dynamic	Dynamic
Embedding	$\overline{y_{ij}}' = y_{ij} + b$ $\overline{y_{ij}}' = y_j^T - y_{ij}^T \; mod \; 2^k + b_l'$	$\overline{y_n}' = \begin{cases} \overline{y_n} - b_i', & if y_{n-1} > \overline{y_n} \\ \overline{y_n} + b_i', & if y_{n-1} < \overline{y_n} \\ \overline{y_n}, & if y_{n-1} = \overline{y_n} \end{cases}$	Improved RDE	$y_i' = y_i \; e_i + b$	$\overline{y_j'} = \overline{y_j} + p_{n\_i}$	See (6)
Reducing	No	No	No	No	Yes	No
Smoothing	No	Yes	No	No	Yes	See (7)

determined by the respective N_i. In the previous embedding step, it has been shown that this number may differ for each sample.

$$d_i = |\overline{y_i'} - \overline{y}_i| \tag{10}$$

Reconstruction of the audio cover is performed after the odd-indexed samples have been de-normalized back to the original range values. The reconstructed audio is in *.WAV file whose sampling rate is 44.1 kHz. In general, the comparison between the proposed and other methods is summarized in Table 1.

Experimental results

As in other research, such as [29–33], the proposed method is evaluated based on the following factors. The first is the level of similarity between the cover and the generated stego files. As previously described, this is to be the quality of the produced stego audio measured by the Peak Signal-to-Noise Ratio (PSNR) that is represented in dB. It is calculated by using (11), whose Mean Squared Error (MSE) value is taken from (12). The constant *MAX* is specified by the maximum possible value of the sample. It is worth to note that audio samples are 1D, which is different from either image or video. So, the variables *m* and *n* are adjusted.

Overall, this PSNR value is likely to be affected by the second evaluation factor (i.e., the size of the secret message represented in bits). Therefore, those two factors should be calculated concurrently.

$$\text{PSNR} = 10 \times \log_{10}\left(\frac{\text{MAX}^2}{\text{MSE}}\right) \tag{11}$$

$$\text{MSE} = \frac{1}{m \times n} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [\overline{y}(i,j) - \overline{y''}(i,j)]^2 \tag{12}$$

Table 2 Various instruments and genres of the audio cover for evaluating the methods [36, 37]

Cover (carrier)	Genre_Instrument
A-1	Country-Folk_Cello
A-2	Classical_Cello
A-3	Pop-Rock_Cello
A-4	Country-Folk_Acoustic-guitar
A-5	Classical_Acoustic-guitar
A-6	Pop-Rock_Acoustic-guitar
A-7	Country-Folk_Piano
A-8	Classical_Piano
A-9	Pop-Rock_Piano
A-10	Country-Folk_Saxophone
A-11	Classical_Saxophone
A-12	Pop-Rock_Saxophone
A-13	Country-Folk_Voice
A-14	Classical_Voice
A-15	Pop-Rock_Voice

Table 3 Size of secret (payload) for the experiment

Payload	Size (kbits)
1	1
2	10
3	20
4	30
5	40
6	50
7	60
8	70
9	80
10	90
11	100

For this measurement, we obtain a public data set of audio files from [36, 37]. It comprises 15 files, consisting of 3 genres, each of which is made up of 5 instruments, as shown in Table 2. For the secret, we generate 11 files with various sizes, starting from 1 to 100 kb, whose detail is provided in Table 3. We believe that these numbers of bits reflect the actual need in the real environment. Furthermore, we implement those experimental data to other research [29–33], such that the comparison can be performed as fairly as possible.

Multiplier

As previously described in "Embedding stage" section, the embedding result is influenced by the multiplier m, whose value is dynamically determined by (3). According to the experimental results, we find that smaller m means a higher quality of the stego data. Therefore, in the next experiment, we set m statically. For this second

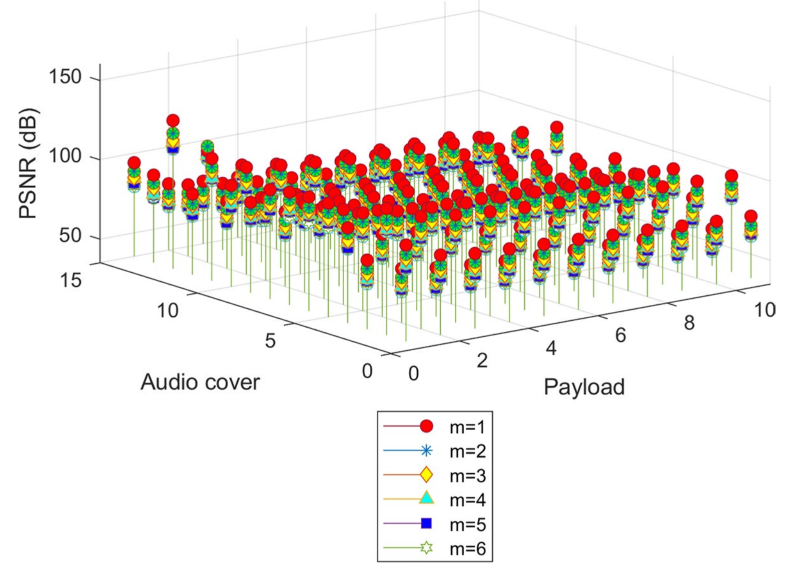

Fig. 5 The effect of multiplier (m) on the quality of the stego files, where $m \in \{1, 2, 3, 4, 5, 6\}$, $e = 3$ and $h = 1$

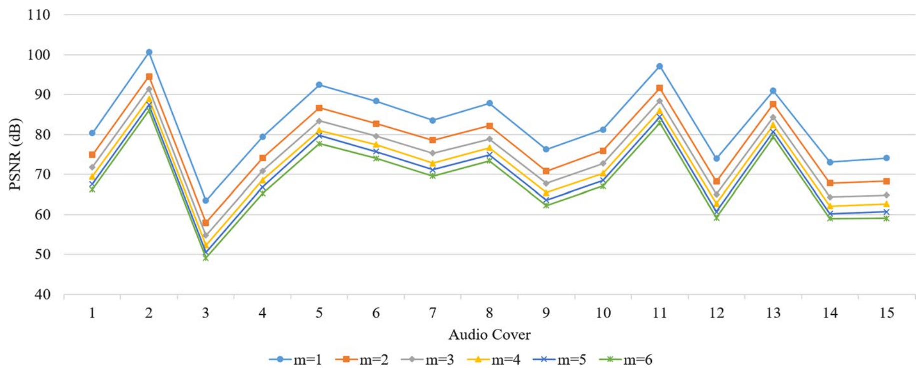

Fig. 6 The average of stego quality, where $m \in \{1, 2, 3, 4, 5, 6\}$, $e = 3$ and $h = 1$

type of m, we take $m \in \{1, 2, 3, 4, 5, 6\}$, whose experimental result is given in Fig. 5 where $e = 3$ and $h = 1$. In Fig. 5, it is depicted that by setting $m = 1$, the generated stego data are the best; on the contrary, $m = 6$ produces the lowest PSNR. On the other hand, the capacity of the secret that can be embedded is lower. This pattern also works on other values of e and h. In evaluating the effect of this multiplier, we take $e = 3$ and $h = 1$ because those values produce the minimum quality of the stego. It means that it is very likely to obtain a better stego quality than what has been achieved in this subsection (see the next analysis in "Smoothing coefficient and smoothing level" section).

On average, the PSNR is provided in Fig. 6. It describes that each audio genre has the same effects on the quality of the stego. The Audio3 (pop-rock_cello) should not

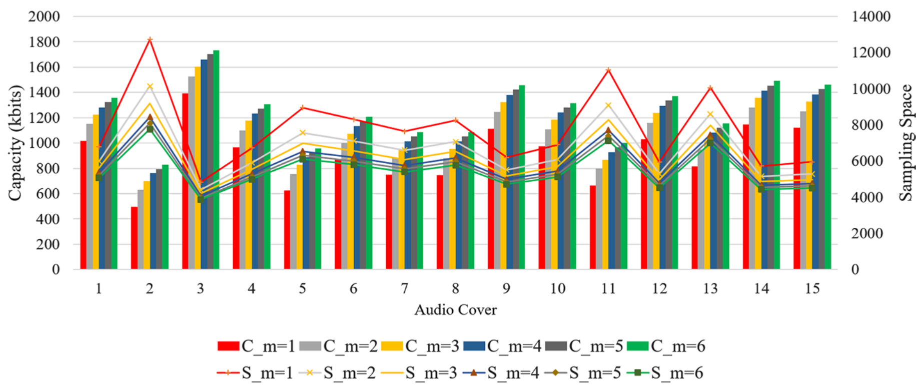

Fig. 7 The average of the maximum number of bits that can be held by the audio cover along with its average of sampling space. Here, *C_m* and *S_m* represent respectively the capacity and the number of sampling space with multiplier *m*, where $m \in \{1, 2, 3, 4, 5, 6\}$, $e = 3$ and $h = 1$

be used if the quality is to be the concern, while Audio2 (classical_cello), Audio11 (classical_saxophone), Audio5 (classical_acoustic-guitar), Audio13 (country-folk_voice) and Audio8 (classical_piano) are more appropriate to use. It is found that, in general, the classical genre produces better stego quality than the others. Nevertheless, as previously predicted, there is a trade-off between the quality and the capacity. That is, Audio3 is the best choice if relatively bigger data are to be protected. This trend also works on Audio12, Audio15, and Audio9, whose genre is pop-rock. The other genres may be applicable if a balance between the quality and the capacity is required.

In more detail, the experiment also shows that lower m causes more space to hold the secret in each sample. Consequently, embedding mainly occurs in fewer samples. It is different from higher m, which provides fewer spaces in each sample; this condition has caused the secret to be widely spread amongst samples. It can be inferred that the number of embedded samples affects the quality of the stego file. An example of this condition is illustrated in Fig. 7, which plots the maximum number of bits that can be embedded and the respective number of sampling spaces. This figure shows this characteristic that the capacity is inversely proportional to the number of spaces. On the contrary, the quality is proportional to the number of spaces.

Smoothing coefficient and smoothing level

The smoothing coefficient (e) is designed to be used in (7) in the embedding process. From the experiment, we find that the higher the e, the higher the PSNR. Nevertheless, as described in the previous section, it is inversely proportional to the capacity. For this reason, in the experiment, we determine the smoothing coefficient of $e \in \{1, 2, 3, 4, 5\}$, whose result is plotted in Fig. 8, where $m = 1$ and $h = 1$.

It is also depicted in that figure that for $e = 4$ and $e = 5$, the PSNR of stego file generated from Audio13 is infinite when the payload is 1 kb. It means that it is the same as its original cover, which is an ideal condition. When the smoothing is performed three times ($h = 3$) with $e = 5$, the PSNR value of all stego files is infinite. It is shown that, at this point, increasing the e is useless because the best quality

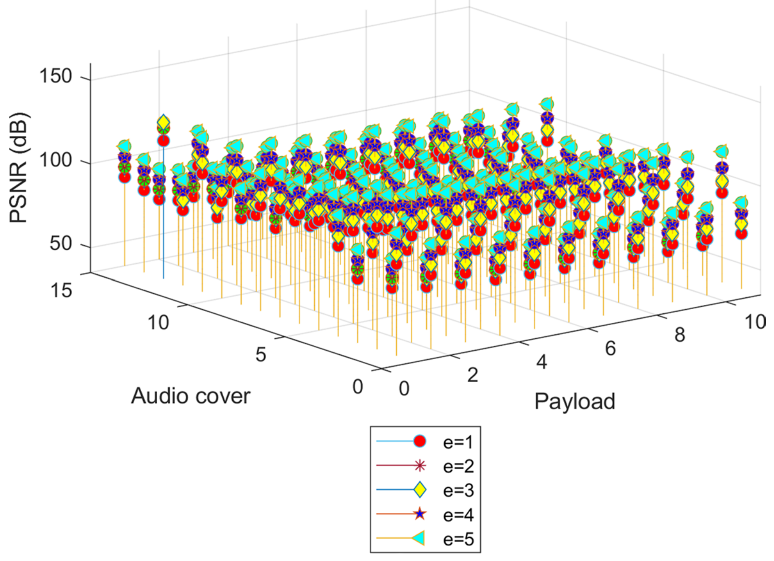

Fig. 8 The quality of the stego file generated by the proposed technique by using various smoothing coefficients, where $e \in \{1, 2, 3, 4, 5\}$, $m = 1$ and $h = 1$

has been achieved. This ideal condition is also reached by $e = 3$ and $e = 4$ when the smoothing is implemented five and four times, respectively. For the number of smoothing less than those, only specific covers obtain the infinite value: Audio2, Audio11, Audio5, Audio13. It is consistent with the results, which have been discussed in "Multiplier" section.

It can be inferred that higher e takes fewer steps to reach the best condition, which is good. At a certain level, raising that value does not affect the quality. Therefore, finding an appropriate e is essential. It is also shown that increasing h instead of e can also improve quality. Some considerations of which parameter should be taken among them have been discussed in "The design of reversible data hiding" section. Next, the capacity of the embedding should also be a consideration. Furthermore, as predicted, the value of h is proportional to the PSNR, regardless of m and e, where a smoothing step may increase around 20–30 dB of PSNR.

Research comparison

For comparing with other research, we take some methods proposed by Bobeica et al. [33], Ahmad and Fiqar [30], Andra et al. [32], Ahmad et al. [31], and Jung and Yoo [29]. To make the comparison as fair as possible, we implement them such that we can evaluate the methods by using the same data, whose results are provided in Figs. 9, 10, and 11. In those figures, the smoothing is applied 1, 3, and 4 times, respectively, depending on their characteristics. Furthermore, in each graph, we show the performance of the proposed technique, which is evaluated without using the smoothing step, considering that not all compared algorithms are designed with the smoothing step.

It is provided in Fig. 9 that without the smoothing, the proposed method is lower than [29], even though it is still higher than the rest of the methods. By implementing one-time smoothing ($h = 1$), the proposed method is excellent. In this environment, [29] is

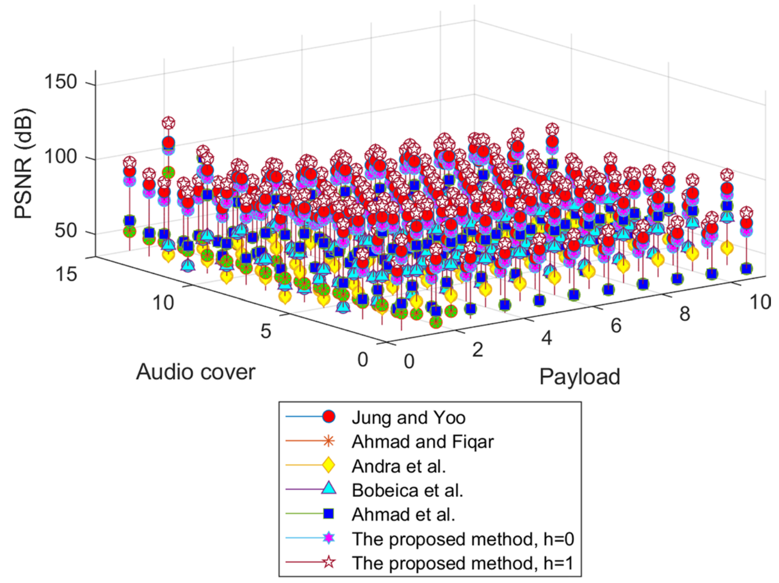

Fig. 9 The quality of the stego file generated by Jung and Yoo [29], Ahmad and Fiqar [30], Andra et al. [32], Bobeica et al. [33], Ahmad et al. [31], and the proposed technique with $e = 3, m = 1, h = 0$ and $h = 1$

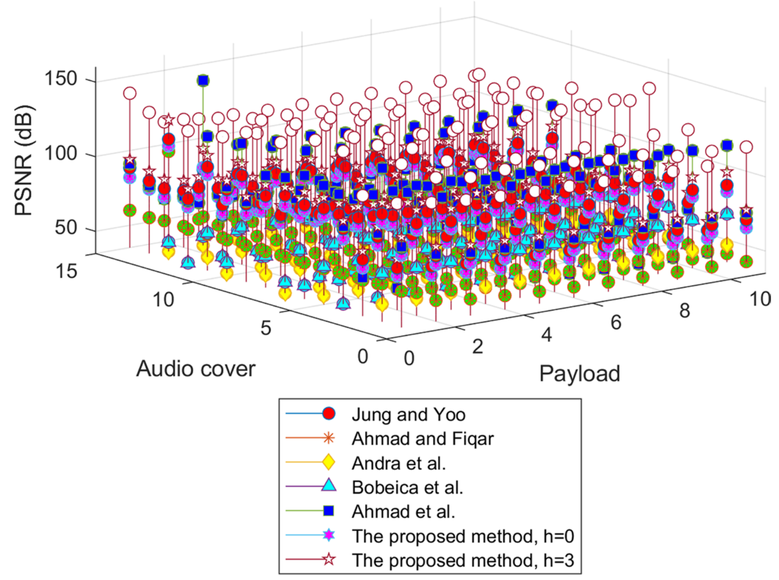

Fig. 10 The quality of the stego file generated by Jung and Yoo [29], Ahmad and Fiqar [30], Andra et al. [32], Bobeica et al. [33], Ahmad et al. [31], and the proposed technique with $e = 3, m = 1, h = 0$ and $h = 3$

gently lower than the proposed algorithm. When the smoothing step is carried out three times (see Fig. 10), the proposed method is still higher than the others. Furthermore, the algorithms in [29, 32, 33] do not have the smoothing step; therefore, their PSNR does not change. It is different from this proposed method, which has flexibility in achieving the required level of the stego file. Finally, by doing smoothing four times (see Fig. 11), almost all PSNR values of the proposed method are infinite. This condition is shown in

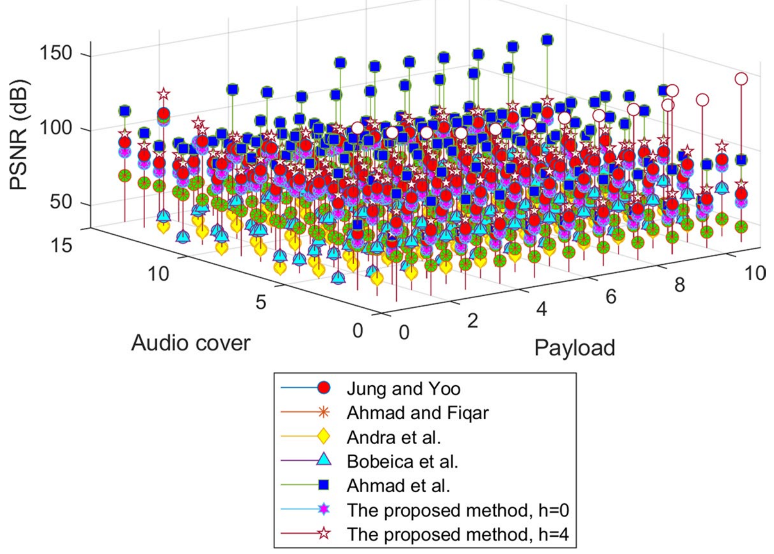

Fig. 11 The quality of the stego file generated by Jung and Yoo [29], Ahmad and Fiqar [30], Andra et al. [32], Bobeica et al. [33], Ahmad et al. [31], and the proposed technique with $e = 3, m = 1, h = 0$ and $h = 4$

Figs. 10 and 11, where the proposed technique is mostly out of those figures since its value is infinite. Please note that here, $e = 3$ and $m = 1$ are the standard in the evaluation. As described in Section 4.2, the performance of the proposed technique is even better than that of others for a higher e.

Discussion

According to the experimental results that have been provided in the previous section, we can determine that the proposed technique outdoes the others. One of the advantages is its flexibility, which can be used to specify the level of the stego file's quality and the size of the secret to protect. Moreover, the reversibility requirement has also been met.

Despite this superiority, it has factors to refine. For the smoothing purpose, the method produces an auxiliary file containing the required information. Considering the security, we may just implement an existing cryptographic algorithm, such as Advanced Encryption Standard (AES), which is out of this data hiding research. Another factor to consider is the size of this auxiliary file, which should be as small as possible. In the case that compression is implemented, it can be performed according to Figs. 2 and 4 for the embedding and extraction, respectively. It is depicted that those two processes are independent of the main data hiding research. Therefore, any algorithm may be suitable to implement.

Considering that research on data hiding is growing fast, any stage of this proposed method may be combined with other research to produce better results. For this purpose, the stages can be grouped in blocks independently, as depicted in Figs. 2 and 4, for embedding and extraction, respectively. On the other hand, attackers may try to detect the existence of hidden information in the stego audio [38, 39]. Although it is out of the scope of this investigation, this possible attack should be anticipated.

Conclusions

In this research, we have proposed a reversible data hiding technique to protect sensitive data in the audio environment. It is carried out by interpolating the audio samples, which have been obtained from the sampling process. The proposed method is flexible in that it can be applied to meet the required performance. Regarding the quality, classical audio is more appropriate to use, combining it with either cello, saxophone, acoustic guitar, or piano. Besides, the quality is proportional to the number of spaces in each sample. If the capacity is the concern, then pop-rock is better to use than others. The implementation itself can focus on either quality or capacity. Moreover, this proposed method has satisfied the reversibility requirement.

The experimental results also signify that the proposed technique is of higher quality than the others in almost all conditions. By applying a one-time smoothing, the method achieves the best results. Moreover, most existing methods even do not have this smoothing stage, which makes their quality static.

In the future, we would like to work on additional factors that do not directly relate to the performance of the proposed technique, which includes reducing the size of the auxiliary file. We believe that it should be as small as possible. An algorithm to protect this file can also be considered, although existing cryptographic algorithms are applicable. Furthermore, the multiplier and the coefficient should be dynamically defined to find a possible value that can deliver better results. This value can be adaptive, considering the characteristics of each cover.

Abbreviations

AES: Advanced Encryption Standard; DE: Difference Expansion; EDH: Echo Data Hiding; HS: Histogram Shifting; MSE: Mean Squared Error; NDDI: Newtons Divided Difference Interpolation; NMI: Neighbor Mean Interpolation; PEE: Prediction Error Expansion; PSNR: Peak Signal to Noise Ratio; RDE: Reduced Difference Expansion; RDH: Reversible data hiding.

Acknowledgements

All research team members, who have given an advice to make the paper better.

Author' contributions

TA-design, analysis, writing draft, funding acquisition, administration; YS-design, software, testing, analysis. Both authors read and approved the final manuscript.

Funding:

The research was supported by Ministry of Research and Technology (Grant Number: 1223/PKS/ITS/2020), the Republic of Indonesia.

Availability data

To be published.

Competing interests

The authors declare that they have no competing interests.

Received: 17 June 2020 Accepted: 14 September 2020
Published online: 22 September 2020

References

1. Hadlington AL. Human factors in cybersecurity; examining the link between Internet addiction, impulsivity, attitudes towards cybersecurity, and risky cybersecurity behaviours. Heliyon. 2017;. https://doi.org/10.1016/j.heliyon.2017.e00346.

2. Tawalbeh LA, Saldamli G. Reconsidering big data security and privacy in cloud and mobile cloud systems. J King Saud Univ Comput Inf Sci. 2019;. https://doi.org/10.1016/j.jksuci.2019.05.007.
3. Azeez NA, der Vyver CV. Security and privacy issues in e-health cloud-based system: a comprehensive content analysis. Egyp Inf J. 2019;20(2):97–108. https://doi.org/10.1016/j.eij.2018.12.001.
4. Tran H-Y, Hu J. Privacy-preserving big data analytics a comprehensive survey. J Parallel Distrib Comput. 2019;134:207–18. https://doi.org/10.1016/j.jpdc.2019.08.007.
5. Calvert CL, Khoshgoftaar TM. Impact of class distribution on the detection of slow HTTP DoS attacks using Big Data. J. Big Data. 2019;. https://doi.org/10.1186/s40537-019-0230-3.
6. Haraty RA, Bitar G. Associating learning technology to sustain the environment through green mobile applications. Heliyon. 2019;5(1):e01141. https://doi.org/10.1016/j.heliyon.2019.e01141.
7. Jallad KA, Aljnidi M, Desouki MS. Big data analysis and distributed deep learning for next-generation intrusion detection system optimization. J Big Data. 2019;. https://doi.org/10.1186/s40537-019-0248-6.
8. Chen K, Chang C. High-capacity reversible data hiding in encrypted images based on extended run-length coding and block-based MSB plane rearrangement. J Vis Commun Image Represent. 2019;58:334–44. https://doi.org/10.1016/j.jvcir.2018.12.023.
9. Qin C, He Z, Luo X, Dong J. Reversible data hiding in encrypted image with separable capability and high embedding capacity. Inf Sci. 2018;465:285–304. https://doi.org/10.1016/j.ins.2018.07.021.
10. Yang Y, Xiao X, Cai X, Zhang W. A secure and high visual-quality framework for medical images by contrast-enhancement reversible data hiding and homomorphic encryption. IEEE Access. 2019;7:96900–11. https://doi.org/10.1109/access.2019.2929298.
11. Fu Y, Kong P, Yao H, Tang Z, Qin C. Effective reversible data hiding in encrypted image with adaptive encoding strategy. Inf Sci. 2019;494:21–36. https://doi.org/10.1016/j.ins.2019.04.043.
12. Xiong L, Dong D. Reversible data hiding in encrypted images with somewhat homomorphic encryption based on sorting block-level prediction-error expansion. J Inf Secur Appl. 2019;47:78–85. https://doi.org/10.1016/j.jisa.2019.04.005.
13. Xiang S, Luo X. Reversible data hiding in homomorphic encrypted domain by mirroring ciphertext group. IEEE Trans Circ Syst Video Technol. 2018;28(11):3099–110. https://doi.org/10.1109/tcsvt.2017.2742023.
14. Al-Juaid N, Gutub A. Combining RSA and audio steganography on personal computers for enhancing security. SN Appl Sci. 2019;. https://doi.org/10.1007/s42452-019-0875-8.
15. Renza D, Dora M, Ballesteros L, Lemus C. Authenticity verification of audio signals based on fragile watermarking for audio forensics. Expert Syst Appl. 2018;91:211–22. https://doi.org/10.1016/j.eswa.2017.09.003.
16. Gutub A, Al-Juaid N, Khan E. Counting-based secret sharing technique for multimedia applications. Multimedia Tools Appl. 2019;78(5):5591–619. https://doi.org/10.1007/s11042-017-5293-6.
17. Shi Y, Li X, Zhang X, Wu H, Ma B. Reversible data hiding: advances in the past two decades. IEEE Access. 2016;4:3210–37. https://doi.org/10.1109/access.2016.2573308.
18. Gutub AA, Alaseri KA. Refining Arabic text stego-techniques for shares memorization of counting-based secret sharing. J King Saud Univ Comput Inf Sci. 2019;. https://doi.org/10.1016/j.jksuci.2019.06.014.
19. Wang S, Yuan W, Wang J, Unoki M. Inaudible Speech Watermarking Based on Self-compensated Echo-hiding and Sparse Subspace Clustering. In: Proceedings of IEEE international conference on acoustics, speech and signal processing (ICASSP), 2019, Brighton, UK. https://doi.org/10.1109/ICASSP.2019.8682352
20. Zhang R, Lu C, Liu J. A high capacity reversible data hiding scheme for encrypted covers based on histogram shifting. J Inf Secur Appl. 2019;47:199–207. https://doi.org/10.1016/j.jisa.2019.05.005.
21. Peng F, Zhao Y, Zhang X, Long M, Pan W. Reversible data hiding based on RSBEMD coding and adaptive multi-segment left and right histogram shifting. Signal Process Image Commun. 2020;. https://doi.org/10.1016/j.image.2019.115715.
22. Liang X, Xiang S. Robust reversible audio watermarking based on high-order difference statistics. Signal Process. 2020;. https://doi.org/10.1016/j.sigpro.2020.107584.
23. Caciula I, Coanda HG, Coltuc D. Multiple moduli prediction error expansion reversible data hiding. Signal Process Image Commun. 2019;71:120–7. https://doi.org/10.1016/j.image.2018.11.005.
24. Xiang S, Li Z. Reversible audio data hiding algorithm using noncausal prediction of alterable orders, EURASIP. J Audio Speech Music Process. 2017;2017(1):4. https://doi.org/10.1186/s13636-017-0101-9.
25. Arham A, Nugroho HA, Adji TB. Multiple layer data hiding scheme based on difference expansion of quad. Signal Process. 2017;137:52–62. https://doi.org/10.1016/j.sigpro.2017.02.001.
26. Choi K-C, Pun C-M, Chen CL. Application of a generalized difference expansion based reversible audio data hiding algorithm. Multimedia Tools Appl. 2015;74(6):1961–82. https://doi.org/10.1007/s11042-013-1732-1.
27. Jiang S, Ye D, Huang J, Shang Y, Zheng Z. SmartSteganogaphy: Light-weight generative audio steganography model for smart embedding application. J Netw Comput Appl. 2020;. https://doi.org/10.1016/j.jnca.2020.102689.
28. Jung K-H, Yoo K. Steganographic method based on interpolation and LSB substitution of digital images. Multimedia Tools Appl. 2015;74(6):2143–55. https://doi.org/10.1007/s11042-013-1832-y.
29. Jung K, Yoo K. Data hiding method using image interpolation. Comput Std Interfaces. 2009;31(2):465–70. https://doi.org/10.1016/j.csi.2008.06.001.
30. Ahmad T, Fiqar TP. Enhancing the performance of audio data hiding method by smoothing interpolated samples. Int J Innov Comput Inf Control. 2018;14(3):767–79.
31. Ahmad T, Amrizal MH, Wibisono W, Ijtihadie RM. Hiding data in audio files: a smoothing-based approach to improve the quality of the Stego Audio. Heliyon. 2020;. https://doi.org/10.1016/j.heliyon.2020.e03464.
32. Andra MB, Ahmad T, Usagawa T. Medical record protection with improved GRDE data hiding method on audio files. Eng Lett. 2017;25(2):112–24.
33. Bobeica A, Dragoi IC, Caciula I, Coltuc D, Albu F, Yang F. Capacity control for prediction error expansion based audio reversible data hiding. In: Proceedings of 22nd international conference on system theory, control and computing (ICSTCC), Sinaia, Romania, 2018, p. 810–815. https://doi.org/10.1109/icstcc.2018.8540672

34. Peng Y, Niu X, Fu L, Yin Z. Image authentication scheme based on reversible fragile watermarking with two images. J Inf Secur Appl. 2018;40:236–46. https://doi.org/10.1016/j.jisa.2018.04.007.
35. Ahmad T, Faruki JN, Ijtihadie RM, Wibisono W. Analyzing the effect of block size on the quality of the stego audio. In: Proceedings of the 5th international conference on science and technology, Yogyakarta, Indonesia, 2019.
36. Bosch JJ, Janer J, Fuhrmann F, Herrera P. A comparison of sound segregation techniques for predominant instrument recognition in musical audio signals. In: Proceedings of ISMIR, p. 559–564; 2012.
37. IRMAS: a dataset for instrument recognition in musical audio signals, accessed on August 2017. [Online]. Available: https://www.upf.edu/web/mtg/irmas/.
38. Gong C, Zhang J, Yang Y, Yi X, Zhao X, Ma Y. Detecting fingerprints of audio steganography software. For Sci Int Rep. 2020;. https://doi.org/10.1016/j.fsir.2020.100075.
39. Wang Y, Zhao X, Cao Y. Detecting the fingerprint of video data hiding tool OpenPuff. For Sci Int Rep. 2020;. https://doi.org/10.1016/j.fsir.2020.100088.