Audio Steganography Using Modified LSB and PVD

R. Darsana and Asha Vijayan

Center For Cyber Security
Amrita Vishwa Vidyapeetham
Coimbatore, India
{darsanaraj1,asha03vijayan}@gmail.com

Abstract. In Audio Steganography we find a way so that an audio file can be used as a host media to hide textual message without affecting the file structure and content of the audio file. In this system a novel high bit rate LSB audio data hiding and another method known as Pixel value differencing is proposed. This scheme reduces embedding distortion of the host audio. The hidden bits are embedded into the higher LSB layers resulting in increased robustness against noise addition. To avoid major differences from the cover audio and the embedded audio this algorithm helps in modifying the rest of the bits. To enlarge the capacity of the hidden secret information and to provide an imperceptible stego-audio for human perception, a pixel-value differencing (PVD) is used for embedding. The difference value of audio samples is replaced by a new value to embed the value of a sub-stream of the secret message. The method is designed in such a way that the modification is never out of the range interval. This method provides an easy way to produce a more imperceptible result than those yielded by simple least-significant-bit replacement methods. The SNR value is good for LSB scheme and the capacity is high for PVD scheme.

Keywords: Audio Steganography, LSB, PVD, substitution Techniques, SNR.

1 Introduction

In Audio Steganography we find a way so that an audio file can be used as a host media to hide textual message without affecting the file structure and content of the audio file. Because of degradation in the perceptual quality of the cover object may leads to a noticeable change in the cover object, may leads to the failure of objective of steganography. The two primary criteria for embedding the covert message are that the stego signal resulting from embedding is indistinguishable from the host audio signal and the message should be correctly received at the receiver side. Audio data hiding method provides the most effective way to protect privacy.

In the past few years, several algorithms for the embedding and extraction of message in audio sequences have been presented.[3] All of the developed algorithms take advantage of the perceptual properties of the human auditory system (HAS) in order to add a message into a host signal in a perceptually transparent manner. Embedding additional information into audio sequences is a more tedious task than that of images, due to dynamic supremacy of the HAS over human visual system. On the other hand, many attacks that are malicious against image steganography

D.C. Wyld et al. (Eds.): NeCoM/WeST/WiMoN 2011, CCIS 197, pp. 11–20, 2011.
© Springer-Verlag Berlin Heidelberg 2011

algorithms cannot be implemented against audio steganography schemes. Audio Steganography should guarantee Undetectability, Capacity, Robustness, Perceptual transparency, Security and Accurate Extraction[1].

The rest of this paper is organized as follows. Section 2 reviews previous works related to Audio steganographic methods. Section 3 summarizes the proposed scheme and has 2 effective methods to embed message in audio. Section 4 presents performance analysis. Section 5 concludes this paper with a summary of the main contributions of this work and future works.

2 Related Work

In audio steganography, secret message is embedded into digitized audio signal which result slight altering of binary sequence of the corresponding audio file. There are several methods are available for audio steganography.

LSB Coding:-Least significant bit (LSB) coding is the simplest way to embed secret message in a digital audio file. This is done by replacing the LSB of each sample with a binary message. This coding helps in embedding large amount of data to be encoded.

Parity Coding:-The parity coding method breaks a signal down into separate regions of samples and encodes each bit from the secret message in a sample region's parity bit. If the parity bit of a selected region does not match the secret bit to be encoded, the process flips the LSB of one of the samples in the region.

Phase Coding:- Phase components of sound are not clearly perceptible to the human ear. Rather than introducing disturbances, the technique encodes the message bits as phase shifts in the phase spectrum of a digital signal, achieving an inaudible encoding in terms of signal-to-perceived noise ratio.

Echo Hiding:-In this information is embedded in a sound file by giving an echo into the discrete signal. It allows for a high data transmission rate and provides superior robustness. To hide the data three parameters of the echo are varied: Amplitude, decay rate, and offset (delay time) from the original signal[11].

Spread Spectrum:-In the context of audio steganography, the basic spread spectrum (SS) method attempts to spread secret information across the audio signal's frequency spectrum as much as possible. This method spreads the secret message over the sound file's frequency spectrum, using a code that is independent of the actual signal. So the final signal occupies a bandwidth in excess of what is actually required for transmission.

3 Proposed Scheme

In this paper we introduce 2 methods for effective embedding in audio files. Here substituting the least significant bit of each sampling point with a binary representation of message. First method uses substitution in appropriate bit positions then reducing the amount of distortion using a modified LSB algorithm. Second method uses another algorithm where the secret message that can be embed is more that also with less distortion. The embedded secret message can be extracted from the resulting stego-audio without referencing the original cover audio.

3.1 Reduced Distortion Bit Embedding (Modified LSB Scheme)

In this method it is able to shift the limit for transparent data hiding in audio from the lower LSB layer to the higher LSB layers such as fourth to sixth layer, using a two-step approach. In the first step, a message bit is embedded into the i^{th} LSB layer of the host audio using a novel LSB coding method. In the second step, the impulse noise caused by embedding is shaped in order to change its white noise properties. The standard LSB coding method simply replaces the original host audio bit in the i^{th} layer (i=1,…,16) with the bit from the message bit stream. In the case when the original and watermark bit are different and ith LSB layer is used for embedding the error caused by embedding is 2^{i-1} (amplitude range is [-32768, 32767]). The embedding error is positive if the original bit is 0 and message bit is 1 and vice versa.

A key idea of the proposed LSB algorithm is message bit embedding that causes minimal embedding distortion of the host audio. It is clear that, if only one of 16 bits in a sample is fixed and equal to the message bit, the other bits can be flipped in order to minimize the embedding error. For example, if the original sample value was $0\ldots01000_2=8_{10}$, and the watermark bit was zero is to be embedded into 4^{th} LSB layer, instead of value $0\ldots00000_2=0_{10}$ that the standard algorithm would produce, the proposed algorithm produces a sample that has value $0\ldots00111_2=7_{10}$, which is far closer to the original one[3]. However, the extraction algorithm remains the same; it simply retrieves the message bit by reading the bit value from the predefined LSB layer in the embedded audio sample.

In the embedding algorithm, the $(i+1)^{th}$ LSB layer (bit a_i) is first modified by insertion of the present message bit. Then the algorithm given below is run. In a case where the bit a_i need not be modified at all due to already being at correct value, no action is taken with the signal sample. The proposed embedding algorithm is implemented 4.1.1.In addition to decreasing objective measure as signal to noise ratio (SNR) value, in the second step of embedding the proposed method introduces noise shaping in order to increase perceptually transparency of the overall method. LSB watermark embedding in a silent or non-dynamic part of the audio sequence causes perceptible hissing noise as significant audio values are introduced where they did not exist in the host audio signal. In order to decrease these perceptual artifacts, the second part of the algorithm is executed. In our algorithm, embedding error is spread to the four consecutive samples, as samples that are predecessors of the current sample cannot be altered because information bits have already been embedded into their LSBs. Let e(n) denote the embedding error of the sample a(n), For the case of embedding into the 4^{th} LSB layer, the next four consecutive samples of the host audio are modified according to these expressions:

a(n+1)=a(n+1)+ l_e(n)_l
a(n+2)=a(n+2)+ l_e(n)/2_l
a(n+3)=a(n+3)+ l_e(n)/3_l
a(n+4)=a(n+4)+ l_e(n)/4_l

where l_A_l denotes floor operation that rounds A to the nearest integer less than or equal to A. Error diffusion method shapes input impulse noise, introduced by LSB embedding, by smearing it. The effect is most emphasized during silent periods of the audio signal and in fragments with low dynamics e.g. broad minimums or maximums. In these cases, there are several hundreds of samples with the same value (e.g. all sixteen bits in a sample are zeros) and error diffusion method shifts the sample levels towards the mean value of expected additive noise. Therefore, the perceptual distortion is not as high as it would be without this step [3]. Both the steps jointly increase the subjective quality of stego object as noise made by LSB embedding has perceptually better-tuned distribution. The proposed LSB scheme thus tries to avoid large modification in the cover and robustness of embedding increases with the increase of the LSB depth used for hiding.

3.1.1 Algorithm

```
If bit 0 is to be embedded
                    if $a_{i-1}$=0 then $a_{i-1}$ai-2…a0=11…1
                    if $a_{i-1}$=1 then $a_{i-1}a_{i-2}$…a0=00…0 and
                              if$a_{i+1}$=0 then $a_{i+1}$=1
                              else if $a_{i+2}$=0 then $a_{i+2}$=1
                              …
                              else if a15=0 then a15=1
          else if bit 1 is to be embedded
                    if $a_{i-1}$=1 then $a_{i-1}a_{i-2}$…a0=00…0
                    if $a_{i-1}$=0 then $a_{i-1}a_{i-2}$…a0=11…1 and

                    if$a_{i+1}$=1 then $a_{i+1}$=0
                              else if $a_{i+2}$=1 then $a_{i+2}$=0
                              …
                              else if $a_{15}$=1 then $a_{15}$=0
```

3.2 Embedding Using Pixel Value Differencing Algorithm

Hiding data in the LSBs of the samples of an audio is a common information hiding method that utilizes the characteristic of the human Auditory System to small changes in the audio. This simple LSB embedding approach is easy for computation, and a large amount of data can be embedded without great quality loss. The more LSBs are used for embedding, the more distorted result will be produced. Not all samples in an audio can tolerate equal amounts of changes without causing notice to a listener. In the PVD embedding method, the cover audio is simply divided into a number of samples. A flowchart of the proposed embedding method is sketched in Fig. 1.

3.2.1 Quantization of Sample Differences

A difference value d is computed from every two consecutive samples, say s_i and $s_{.i1}$, of a given cover audio. Assume that the values of si and $s_{.i1}$ are vi and v $_{.i1}$ respectively, and then d is computed as v $_{i\,.}$- v_{i+1} which may be in the range from 0 to 255 if 8 bit quantization is used. A block with d close to 0 is considered to be an extremely smooth block, whereas a block with d close to -255 or 255 is considered as a sharply edged block. By symmetry, we only consider the possible absolute values of d (0 through 255) and classify them into a number of contiguous ranges, say R_i where i =1,2,...n. These ranges are assigned indices 1 though n. The lower and upper bound values are l_i and R_i respectively, where l_i is 0 and u_i is 255. The width of R_i is the selected range intervals are based on the human visual capability mentioned previously.

The widths of the ranges which represent the difference values of smooth samples are chosen to be smaller while those which represent the difference values of highly varying samples are chosen to be larger. That is, we create ranges with smaller widths when d is close to 0 and ones with larger widths when d is far away from 0 for the purpose of yielding better undistorted results. A difference value which falls in a range with index k is said to have index k. All the values in a certain range (i.e., all the values with an identical index) are considered as close enough. That is, if a difference value in a range is replaced by another in the same range, the change presumably cannot be easily noticed by human ears. Here some bits of the secret message is embedded into a audio samples by replacing the difference value of the block with one with an identical index, i.e., we change a difference value in one range into any of the difference values in the same range. In other words, in the proposed data embedding process, we adjust the sample values in each two sample pair by two new ones whose difference value causes changes unnoticeable to a listener of the stego-audio.

3.3 Data Embedding and Extraction

We consider the secret message as a long bit stream. We want to embed every bit in the bit stream into the sample pair of the cover audio. The number of bits which can be embedded in each block varies and is decided by the width of the range to which the difference value of the two samples belongs[2]. Given a sample pair B with index k and value difference d, the number of bits, say n, which can be embedded in this block, is calculated by Since the width of each range is selected to be a power of 2,the value of $n=\log_2 (u_k - l_k + 1)$ is an integer.

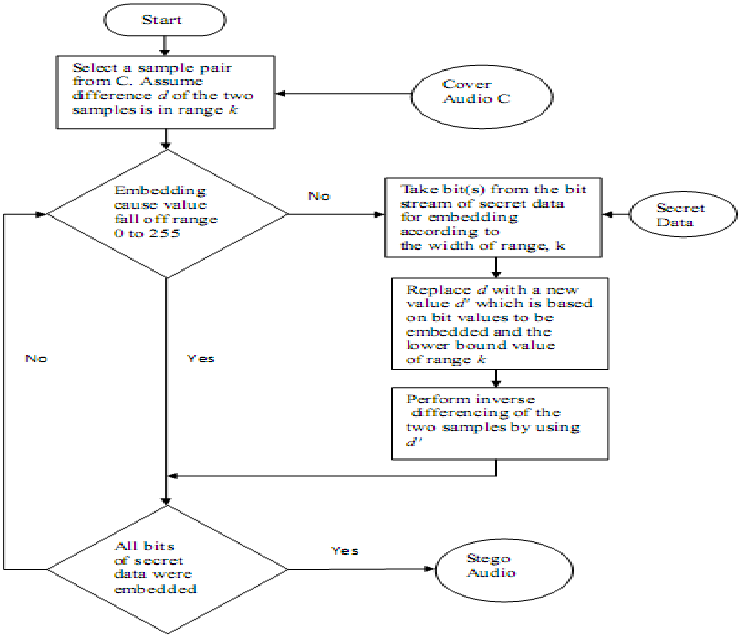

Start
Select a sample pair from C. Assume difference d of the two samples is in range k
Cover Audio C
Embedding cause value fall off range 0 to 255
No
Take bit(s) from the bit stream of secret data for embedding according to the width of range, k
Secret Data
Replace d with a new value d' which is based on bit values to be embedded and the lower bound value of range k
Perform inverse differencing of the two samples by using d'
No
Yes
All bits of secret data were embedded
Yes
Stego Audio

Fig. 1. The data embedding process in PVD

3.3.1 Procedure

1) Calculate the difference value di between two consecutive samples pi and p_{i+1} for each block in the cover audio. The value is given by $d_i=v_{i+1}-v_i$

2) Using d_i to locate a suitable R_k in the designed range table, to compute $j=\min(u_k-|d_i|)$ where $u_k >= d_i$ for all $1<=k<=n$ is the located range.

3) Compute the amount of secret data bits *t* that can be embedded in each pair of two consecutive samples by R_i. The value *t* can be estimated from the width *w* of R_j this can be defined by $t=\log 2w_j$

4) Read *t* bits from the binary secret data and transform the bit sequence into a decimal value *b*. For instance, if bit sequence = 110 , then the converted value *b* = 6.

5) Calculate the new difference value d to replace the original difference

$$di' = \begin{cases} lj+b & if\ di \geq 0 \\ -(lj+b) & ifdi < 0 \end{cases}$$

6) Modify the values of p_i and p_{i+1}1 by the following formula

$(v_i', v'_{i+1}) = (v_i - ceil(m), v_{i+1} + floor(m))$ if *d* is odd

$(v_i', v'_{i+1}) = (v_i - floor(m), v + ceil(m))$ if d is even where $m = (d'-d)/2_{i+1}$

Repeat Step 1-6 until all secret data are embedded into the cover audio, then the stego-audio is obtained.During the phase of secret extraction, the original designed range table is required. In the beginning, the same method in the embedding phase is used to partition the stego-audio into sample pairs. Then the difference value d for each pair of two consecutive samples pi*and p^* the stego-audio is calculated. Next, $d_i{}^*$ is used to locate the suitable R_{i+1} in Step 2 during the embedding phase. Therefore, b^* is obtained by subtracting l_j from $d_i{}^*$. If the stego-audio is not altered, b^* is equal to b. Finally, b^* is transformed from a decimal value into a binary sequence with t bits, where $t=\log 2w_j$.

The above equations satisfy the requirement that the difference between v'_i and v'_{i+1} is *d'*. It is noted that a distortion reduction policy has been employed in designing for producing v'_i andv'_{i+1} from v_i and v_{i+1}, so that the distortion caused by changing v_i and v_{i+1} is nearly equally distributed over the two samples. The effect is that the resulting change is less perceptible. An illustration of the data embedding process is shown in Figure 2. In the inverse calculation, a smaller value of d' produces a smaller range interval between v'_i and v'_{i+1} while a larger d' produces a larger interval. Some of the calculation may cause the resulting (v'_i , v'_{i+1}) to fall off the boundaries of the range [0,255] Although we may re-adjust the two new values into the valid range of [0, 255] by forcing a falling off boundary value to be one of the boundary values of 0 and 255, and adjusting the other to a proper value to satisfy the difference d', yet this might produce some distortions. To solve this problem, a checking process is employed to detect such falling off boundary cases, and abandon the samples which yield such cases for data embedding[2]. The sample values of the abandoned blocks are left intact in the stego-audio. This strategy helps us to distinguish easily samples with embedded data from abandoned blocks in the process of recovering data from a stego audio. The proposed falling-off-boundary checking proceeds by producing a pair (v^*_i and v^*_{i+1}) by replacing m as m=(uk-d)/2. Since u_k is the maximum value in the range l_k to u_k the resulting pair of v$^*_i$,v$^*_{i+1}$) produced by the use of u_k will yield the maximum difference That is, this maximum range interval (v$^*_{i}$ - v^*_{i+1})covers all over the ranges yielded by the other (v$^*_i$,v$^*_{i+1}$). So the falling off boundary checking for the block can proceed by only examining the values of (v$^*_i$,v$^*_{i+1}$) which are produced by the case of using u_k If either v$^*_i$ or v$^*_{i+1}$ falls off the boundary of 0 or 255, we regard the block to have the possibility of falling-off, and abandon the block for embedding data. In addition, the inverse calculation in is designed in such a way that the inverse calculation can proceed directly or progressively. This property is useful for judging the existence of embedded data in each block in the data recovering process. Assume that blocks in stego audio has the values (v$_i$,v$_{i+1}$) and that the difference d' of the two values is with index k. we apply the falling off boundary the two values is with index k.

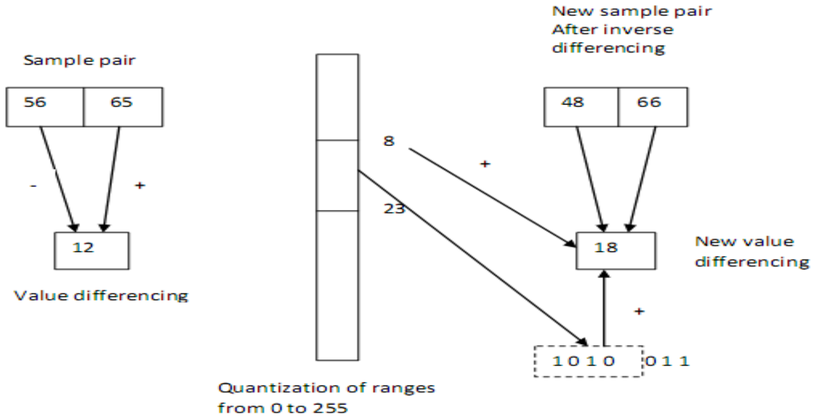


Fig. 2. Illustration of the data embedding process

We apply the falling-off-boundary checking process to (v'$_i$, v'_{i+1}) by using $m=(u_k-d')/2$ and (v''$_i$, v''_{i+1}) is the resulting values We now want to prove that the resulting (v''$_i$, v''_{i+1}) are identical to the values (v$^*_i$,v$^*_{i+1}$)which were computed by $m=(u_k-d')/2$ in the embedding process The proof is as follows

$$(v'_{i,}\ v'_{i+1}) = f((v'_{i,}\ v'_{i+1}),u_k-d) = f((v_{i,}v_{i+1}),d'-d+u_k-d')$$

Also, the inverse calculation is designed in such a way that it satisfies the following property: $f((v_i\ ,v_{i+1),}m) = f(f((v_{i,}v_{i+1})m')m'')$ for m=m'+m''

The above result can be transformed further to be

$$f(v_{i,}v_{i+1}),d'-d+u_k-d') = f(f(v_{i,}v_{i+1}),d'-d)u_k-d')$$

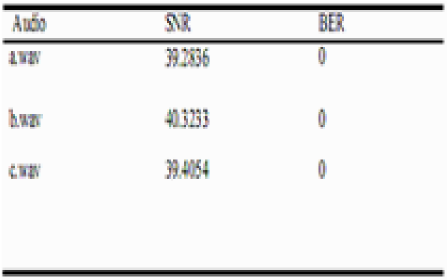
$$= f((v_{i,}v_{i+1}),u_k-d') = (v''_i\ ,v''_{i+1})$$

This completes the proof. The above property shows that the results of both of the falling-off-boundary checking processes, one in data embedding and the other in data recovery, are identical. Note that in the recovery of the secret message from the stego-audio using the previously described extraction process, there is no need of referencing the cover audio.

4 Performance Analysis

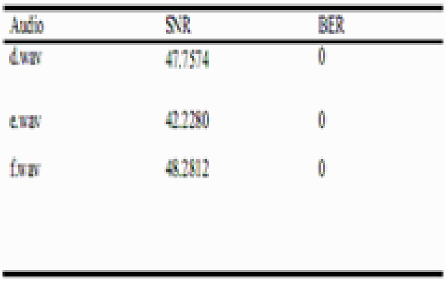
The goal of research is to develop quantitative measures that can automatically improve audio quality. The simplest and most widely used full-reference quality metric is the Signal to Noise ratio (SNR). Steganography capacity is the maximum message size that can be embedded subject to certain constraints. Tables 1 & 2 shows SNR values of different audios. We have randomly taken audios after embedding in those audios the SNR values are still around 40 db. Figures 3 to 6 shows the histogram analysis of original and embedded audios using two methods. Results show that quality is preserved in both methods. In audios 20 % embedding has done. Compared to LSB method PVD method can hold more secret data but LSB method shows high SNR value.

Table 1. SNR value of different audios using PVD method



Audio	SNR	BER
a.wav	39.2836	0
b.wav	40.3233	0
c.wav	39.4054	0

Table 2. SNR value of different audios using LSB method



Audio	SNR	BER
d.wav	47.7574	0
e.wav	42.2280	0
f.wav	48.2812	0

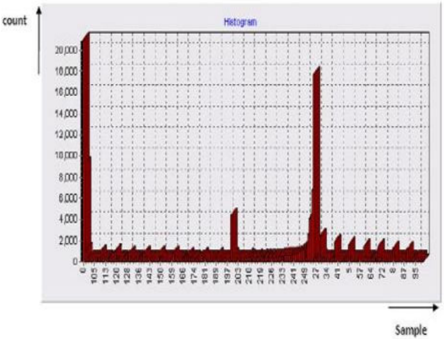


Fig. 3. Histogram of original audio using PVD

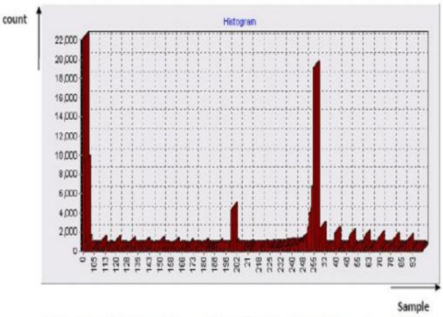


Fig. 4. Histogram of embedded audio using PVD

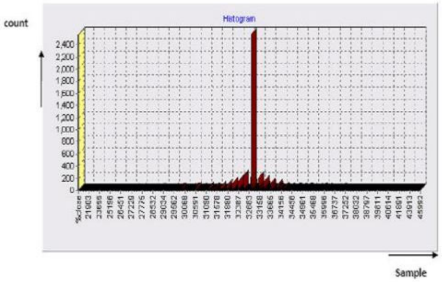


Fig. 5. Original audio using Modified LSB

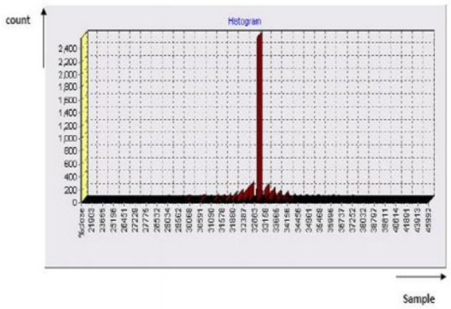


Fig. 6. Embedded audio using Modified LSB

5 Conclusion and Future Work

A new approach is proposed to resolve two problems of substitution technique of audio steganography. First problem is having low robustness against attacks which try to reveal the hidden message and second one is having low robustness against distortions with high average power. Substitution algorithm will try to embed the message bits in the deeper layers of samples and other bits are altered to decrease the error and if alteration is not possible for any samples it will ignore them. Using the proposed scheme, message bits could be embedded into vague and deeper layers to achieve higher robustness. We tested the proposed algorithms with 5 music clips. The clips are sampled at 44.1 KHz with the length of about 1 minute and quantized by 8 bits. The performance of the proposed scheme in terms of SNR (Signal to NoiseRatio)and histogram is analyzed and listed in Table 1 & 2 and Figures 3 to 6. In modified LSB method the distortion are reduced when compared with standard method and in PVD method the capacity of secret data that can be embedded is more. In the future, more effective methods should be taken into account to further increase the embedding capacity and embedding should be done in live audio.

References

1. Zamani, M., Manaf, A.B.A., Ahmad, R.B., Jaryani, F., Hamed, Taherdoost, Zeki, A.M.: A Secure Audio Steganography Approach. IEEE Xplore (2009)
2. Wu, D.C., Tsai, W.H.: A Steganographic Method for Images By Pixel Value Differencing. Pattern Recognition Letters 24, 1613–1626 (2003)
3. Cvejic, N., Seppanen, T.: Increasing Robustness of LSB Audio Steganography Using a Novel Embedding method. In: Proceedings of the International Conference on Information Technology: Coding and Computing, ITCC 2004 (2004)
4. Basu, P.N., Bhowmik, T.: On Embedding of Text in Audio – A case of Steganography. In: 2010 International Conference on Recent Trends in Information, Telecommunication and Computing (2010)
5. Zamani, M., Manaf, A.B.A., et al.: An Approach to Improve the Robustness of Substitution Techniques of Audio Steganography. IEEE Xplore (2009)

6. Gopalan, K.: Audio Steganography Using Bit Modification. In: Proceedings of the 2003 IEEE lntematianal Conference (2003)
7. Katzenbeisser, S., Petitcolas, F.A.P.: Information Hiding Techniques for Steganography and Digital Watermarking. Artech House, Norwood (2000)
8. Cvejic, N.: Algorithms for audio watermarking and steganography. In: Information Processing Laboratory. University of Oulu, Oulu (2004)
9. Wang, H., Wang, S.: Cyber warfare: Steganography vs. Steganalysis. Communications of the ACM 47(10) (October 2004)
10. Artz, D.: Digital Steganography: Hiding Data within Data. IEEE Internet Computing (May-June 2001)
11. Geetha, K., Vanitha Muthu, P.: Implementation of ETAS (Embedding Text in Audio Signal) Model to Ensure Secrecy. (IJCSE) International Journal on Computer Science and Engineering 02(04), 1308–1313 (2010)