The Eurasia Proceedings of Science, Technology, Engineering & Mathematics (EPSTEM)
ISSN: 2602-3199

The Eurasia Proceedings of Science, Technology, Engineering & Mathematics (EPSTEM)

Volume 1, Pages 397-403

ICONTES2017: International Conference on Technology, Engineering and Science

A COMPARISON OF ECHO HIDING METHODS

Kadir Tekeli
Adnan Menderes University

Rifat Asliyan
Adnan Menderes University

Abstract: Echo-based data hiding algorithms are known to be very popular in audio watermarking research. In this study, several echo kernels such as single, bipolar, backward-forward, bipolar backward-forward and time-spread, which are coded in Matlab, are compared in terms of robustness against mp3 compression, imperceptibility and capacity. Mixer signals with sinusoidal smoothing are used to improve imperceptibility. All kernels are tested on the same audio database for a fair comparison. Importance of parameters is discussed during comparison. Results show that bipolar backward-forward echo kernel is more robust against compression and gives a higher SNR (Signal-to-noise ratio).

Keywords: Audio steganography, watermarking, echo hiding

Introduction

Steganography is the science of hiding information within cover objects in order to transfer data securely. Study of Anderson & Petitcolas (1998) is a good source about what steganography is with its history and limits. Digital steganography can be used for secret communication or encoding copyright information (watermark) into digital media files such as image, audio and video to prove ownership. Pure steganography is interested in payload capacity and imperceptibility while watermarking focuses on robustness.

Audio steganography is harder than image steganography since Human Auditory System (HAS) is more sensitive than Human Visual System (HVS). There are many methods with different approaches proposed to hide secret data in audio files. Least Significant Bit (LSB) Coding, Phase Coding (Bender et al., 1996), Echo Hiding (Gruhl et al., 1996), Quantization Index Modulation (Chen & Wornell, 2001), Spread Spectrum (Kirovski & Malvar, 2001), Patchwork (Yeo & Kim, 2003) are some of these audio steganography (watermarking) methods.

In this study, various echo hiding algorithms have been studied and compared in terms of robustness and imperceptibility. Since echo hiding is proposed in 1996 for the first time, many approaches came out in order to improve conventional echo hiding method. Bipolar echo kernel (also known as negative-positive echo kernel) has been proposed by Oh et al. (2001) to improve robustness where two bits are represented in the same kernel with opposite signs. Backward-forward echo kernel has been proposed by Kim & Choi (2003) with symmetrical echo impulses to add backward and forward echoes at the same time which improved robustness much more. In order to improve security of echo hiding, time-spread echo hiding (Ko et al., 2005) is used where many more echoes are generated by a pseudorandom sequence to represent data bits. Bipolar backward-forward echo kernel (also known as negative-positive and backward-forward echo kernel or mirrored echo kernel) is used in analysis-by-synthesis approach (Wu & Chen, 2006).

- Selection and peer-review under responsibility of the Organizing Committee of the conference

*Corresponding author: Kadir Tekeli-E-mail: kadir.tekeli@outlook.com

Methods

Single echo hiding (SEH), bipolar echo hiding (BEH), backward-forward echo hiding (BFEH), bipolar backward-forward echo hiding (BBFEH) and time-spread echo hiding (TSEH) methods are described below. Note that notation and explanation of methods can be slightly different from original papers. We intend to give basics of each variation, so same notation is used for each method to be able to provide a common expression.

Single Echo Hiding

Echo hiding (Gruhl et al., 1996) is the first echo-based audio watermarking method where data is embedded into cover audio by adding up delayed versions of audio signal on itself. In conventional method, data bits are represented by single echoes with known delays for each bit, so it is also known as single echo hiding (SEH) in literature.

Let d_0, d_1 and α be delay for bit zero, delay for bit one and echo amplitude respectively. Then echo kernels can be notated as in Equation 1 where $\delta[n]$ is Kronecker delta function given in Equation 2 which represents unit impulse for discrete signals. This echo kernel is shown on Figure 1.

$$h_i[n] = \delta[n] + \alpha\delta[n - d_i], \; i \in \{0,1\} \tag{1}$$

$$\delta[n] = \begin{cases} 1, & \text{if } n = 0 \\ 0, & \text{if } n \neq 0 \end{cases} \tag{2}$$

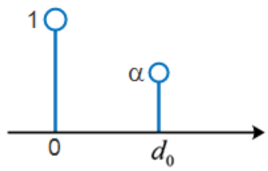

Figure 1. Single echo kernel

Echoed signals are generated by convolving cover audio and echo kernels. Echoing audio signals can be accomplished in various ways below using definition of Kronecker delta function knowing that convolution of cover audio $s[n]$ and $\delta[n-d_i]$ is $s[n-d_i]$.

a) $x[n] = s[n] + \alpha \cdot s[n-d]$
b) $x[n] = s[n] * (\delta[n] + \alpha \cdot \delta[n-d])$
c) $x[n] = s[n] + s[n] * (\alpha \cdot \delta[n-d])$

Echo kernels are usually given with second notation, but it is more practical to use first or third notation in coding which does the same process by using mixer signals to avoid dividing audio signal into segments.

Encoding Process

Cover audio is divided into segments as the number of data bits to be embedded. Then each segment is echoed with the delay corresponding to data bit to be encoded. Let *N* be the number of bits to hide and *L* is the length of segments. Then *L* must be chosen such that $N \cdot L$ is not greater than length of audio signal.

Generating a mixer signal using data bits is very advantageous during embedding since a smoothed mixer signal will improve distortion between adjacent segments. An example mixer signal to embed a bit sequence 01001011 is shown on Figure 2.

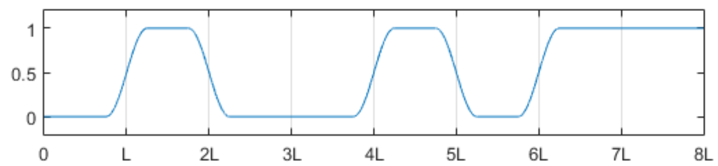

Figure 2. An example mixer signal smoothed with sinusoidal modulation

Echoed signals can be filtered with mixer signal applying dot product before adding up onto cover audio signal. Let k_0 and k_1 be delayed signals with d_0 and d_1 samples of delays, s be cover audio and x be stego signal, then embedding process can be formulized as in Equation 3.

$$x = s + k_1 \cdot \text{mixer} + k_0 \cdot (1 - \text{mixer}) \tag{3}$$

i. Assign $h^*_i = [0, 0, ..., 0, 1]$, $i \in \{0,1\}$ where h^*_0 contains zeros as the number of d_0 at the beginning, and so does h^*_1 as the number of d_1.
ii. Convolve cover audio with h^*_0 and h^*_1 seperately to obtain delayed versions of audio signal. This process can be shortened using filter function in matlab-like programs as; $k_i = \text{filter}\left(h^*_i, 1, s\right)$, $i \in \{0,1\}$.
iii. Generate a smoothed mixer signal using data bits to embed as shown on Figure 2.
iv. Add echoes onto cover audio filtering with mixer signal as in Equation 3.

Decoding Process

Cepstrum analysis is used in echo hiding methods in decoding process. Stego audio is divided into segments as the number of hidden bits with the same segment length as has been used during encoding. Then to retrieve *n*th hidden bit real cepstrum of *n*th segment which is given in Equation 4 is compared on delay points as if $c_n[d_0 + 1] > c_n[d_1 + 1]$ then retrieved bit is zero, else it is one.

$$c_n[i] = \text{ifft}\left(\log\left(\text{abs}\left(\text{fft}\left(s_n[i]\right)\right)\right)\right) \tag{4}$$

Bipolar Echo Hiding

Bipolar echo hiding (BEH), also known as negative-positive echo hiding, is done by having two echo impulses in the kernel where second one is multiplied by negative echo amplitude representing bipolar of the bit to be embedded. This echo kernel can be notated in Equation 5 corresponding to zero bit using same variables as in single echo kernel. This kernel is shown on Figure 3.

$$h_0[n] = \delta[n] + \frac{\alpha}{2}\delta[n - d_0] - \frac{\alpha}{2}\delta[n - d_1] \tag{5}$$

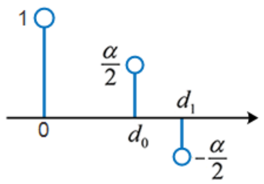

Figure 3. Bipolar echo kernel

Encoding and decoding processes are done very similar as in single echo hiding.

Backward-Forward Echo Hiding

Backward-forward echo hiding (BFEH) is done by having two echo impulses in the kernel where second one is mirrored of the first one with same delay according to bit to be embedded. This echo kernel can be notated as in Equation 6 corresponding to zero bit using same variables as in single echo kernel, and is shown on Figure 4.

$$h_0[n] = \delta[n] + \frac{\alpha}{2}\delta[n-d_0] + \frac{\alpha}{2}\delta[n+d_0] \tag{6}$$

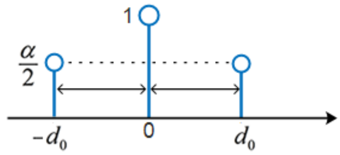

Figure 4. Backward-forward echo kernel

Encoding and decoding processes are done very similar as in single echo hiding.

Bipolar Backward-Forward Echo Hiding

Bipolar backward-forward echo hiding (BBFEH) is a combination of bipolar and backward-forward echo kernels where bipolar kernel is simply mirrored. This echo kernel can be notated in Equation 7 corresponding to zero bit using same variables as in single echo kernel, and is shown on Figure 5.

$$h_0[n] = \delta[n] + \frac{\alpha}{4}\delta[n-d_0] - \frac{\alpha}{4}\delta[n-d_1] + \frac{\alpha}{4}\delta[n+d_0] - \frac{\alpha}{4}\delta[n+d_1] \tag{7}$$

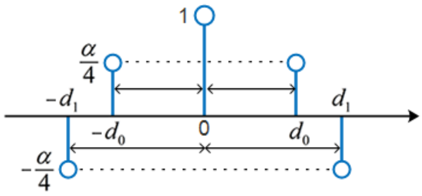

Figure 5. Bipolar backward-forward echo kernel

Encoding and decoding processes are done very similar as in single echo hiding.

Time-Spread Echo Hiding

Decoding process with usual echo kernels is very simple since cepstrum analysis is enough to retrieve data back. This can be advantageous in some cases, but this simplicity causes weakness in security at the same time. Even if data is encrypted before embedding, its existence still can be detected easily. Time-spread echo hiding is an echo-based data hiding method with multiple echoes whose echo amplitudes are generated by a pseudorandom sequence $p[n]$ such that $n \in \{1, 2, ..., L_{PN}\}$ and $p[n] \in \{-\beta, \beta\}$ where L_{PN} is the length of the sequence and $0 < \beta \ll 1$ is echo amplitude. Time-spread echo kernel corresponding to zero bit can be notated as in Equation 8 and it is shown on Figure 2.

$$h_0[n] = \delta[n] + \beta \cdot p[n-d_0] \tag{8}$$

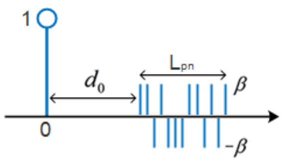

Figure 6. Time-spread echo kernel

M-Sequence or any other pseudorandom sequence, where same sequence can be generated using the same key each time, can be used.

Encoding Process

Encoding process is explained below as simplified.

i. Generate a pseudorandom sequence $p[n]$ with the length of L_{PN}.

ii. Assign $h^*_i = [0,0,...,0, p[1], p[2],..., p[L_{PN}]], i \in \{0,1\}$ where h^*_0 contains zeros as the number of d_0 at the beginning, and so does h^*_1 as the number of d_1.

iii. Convolve cover audio with h^*_0 and h^*_1 to obtain delayed versions of audio signals. This process can be shortened using filter function in matlab-like programs as; $k_i = \text{filter}(h^*_i, 1, s), i \in \{0,1\}$.

iv. Generate a smoothed mixer signal using data bits to embed as shown on Figure 2.

v. Add echoes onto cover audio filtering with mixer signal as in Equation 3.

Decoding Process

Decoding process is done using real cepstrum as in single echo hiding. However, since echoes are spreaded in time using a pseudorandom sequence, it needs to be despreaded with the same sequence having cross correlation of real cepstrum and pseudorandom sequence. Then peaks at d_0 and d_1 can be compared to retrieve data back.

It is impossible to retrieve hidden back without the same pseudorandom sequence which is used in encoding. Existence of hidden data is also not detectable by third party. So, time-spread echo hiding is considered to be more secure comparing to other echo-based hiding algorithms.

Results

Comparison is done over 76 different single channel audio signals chosen from different types such as pop, jazz, rock, classical, country music and speech samples where each one is 30 seconds long and sampled with 44.1 kHz. Robustness of methods against mp3 compression at 64, 96 and 126 kbit/s is compared calculating BER (Bit error rate) given in Equation 9 and NC (Normalized correlation) given in Equation 10 where w is desired bit and w' is retrieved bit. Imperceptibility is compared via calculating SNR (Sample-to-noise ratio) given in Equation 11.

$$BER(w,w') = \frac{100}{N} \cdot \sum_{n=1}^{N} XOR(w(n), w'(n)) \tag{9}$$

$$NC(w,w') = \frac{\sum_{n=1}^{N} w(n) \cdot w'(n)}{\sqrt{\sum_{n=1}^{N} (w(n))^2 \cdot \sum_{n=1}^{N} (w'(n))^2}} \tag{10}$$

$$SNR(s,s') = 10\log_{10}\left(\frac{\sum_{k=1}^{K} (s(k))^2}{\sum_{k=1}^{K} (s'(k) - s(k))^2} \right) \tag{11}$$

Parameters such as L (length of segments), d_0 and d_1 have been changed in order to see importance of parameters. Echo amplitudes are taken fixed as $\alpha = 0.4$ for SEH, $\alpha/2 = 0.2$ for BEH and BFEH, $\alpha/4 = 0.1$ for BBFEH for a fair comparison. Echo amplitude for TSEH needs to be $\beta = \alpha / L_{PN} \approx 0.0008$ theorically for a fair comparison, but it is taken as $\beta = 0.002$ since a great amount of data loss is observed otherwise.

Results are shown on Table 1-4.

Table 1. Comparison of Methods with $L = 1024,\ d_0 = 50,\ d_1 = 75,\ \alpha = 0.4,\ \beta = 0.02,\ L_{PN} = 512$

			SEH	**BEH**	**BFEH**	**BBFEH**	**TSEH**
Robustness	128 kbit	BER	5.637	5.392	5.625	5.099	7.987
		NC	0.943	0.946	0.949	0.949	0.917
	96 kbit	BER	5.678	5.464	5.200	5.159	8.117
		NC	0.943	0.945	0.947	0.948	0.917
	64 kbit	BER	6.161	5.884	5.587	5.548	9.138
		NC	0.938	0.941	0.944	0.945	0.906
	MEAN	BER	5.825	5.580	5.471	**5.269**	8.414
		NC	0.941	0.944	0.947	**0.947**	0.913
Imperceptibility		SNR	8.062	12.50	11.13	**15.48**	8.480

Table 2. Comparison of Methods with $L = 4096,\ d_0 = 50,\ d_1 = 75,\ \alpha = 0.4,\ \beta = 0.02\ L_{PN} = 512$

			SEH	**BEH**	**BFEH**	**BBFEH**	**TSEH**
Robustness	128 kbit	BER	1.834	1.673	1.546	1.551	1.030
		NC	0.982	0.983	0.985	0.984	0.990
	96 kbit	BER	1.858	1.668	1.574	1.552	1.033
		NC	0.981	0.983	0.984	0.985	0.987
	64 kbit	BER	1.870	1.784	1.628	1.618	1.082
		NC	0.981	0.982	0.983	0.984	0.989
	MEAN	BER	1.854	1.708	1.583	1.574	**1.049**
		NC	0.981	0.983	0.984	0.984	**0.989**
Imperceptibility		SNR	7.994	12.30	11.06	**15.29**	8.390

Table 3. Comparison of methods with $L = 4096,\ d_0 = 250,\ d_1 = 350,\ \alpha = 0.4,\ \beta = 0.02\ L_{PN} = 512$

			SEH	**BEH**	**BFEH**	**BBFEH**	**TSEH**
Robustness	128 kbit	BER	1.044	0.930	0.906	0.870	0.876
		NC	0.989	0.991	0.991	0.991	0.991
	96 kbit	BER	1.070	0.930	0.893	0.881	0.877
		NC	0.989	0.991	0.991	0.991	0.991
	64 kbit	BER	1.245	1.020	0.981	0.917	0.973
		NC	0.987	0.990	0.990	0.991	0.990
	MEAN	BER	1.120	0.960	0.927	**0.889**	0.909
		NC	0.989	0.990	0.991	**0.991**	0.991
Imperceptibility		SNR	8.001	11.13	10.92	**14.20**	8.395

Table 4. Comparison of methods with $L = 8192,\ d_0 = 250,\ d_1 = 350,\ \alpha = 0.4,\ \beta = 0.02\ L_{PN} = 512$

			SEH	**BEH**	**BFEH**	**BBFEH**	**TSEH**
Robustness	128 kbit	BER	0.363	0.281	0.253	0.238	0.226
		NC	0.996	0.997	0.997	0.998	0.998
	96 kbit	BER	0.387	0.316	0.303	0.283	0.228
		NC	0.996	0.997	0.997	0.997	0.997
	64 kbit	BER	0.375	0.294	0.349	0.346	0.222
		NC	0.996	0.997	0.997	0.997	0.998
	MEAN	BER	0.375	0.297	0.302	0.290	**0.225**
		NC	0.996	0.997	0.997	0.997	**0.998**
Imperceptibility		SNR	7.999	11.11	10.82	**14.18**	8.387

Conclusion

Results show that robustness is improved when L is being increased. Amount of delays are also important at robustness according to experiments. BBFEH is more robust within other methods considering that echo amplitude for TSEH is taken greater than it should to be.

It is observed that SNR is being improved proportionally as the number of echoes are increased in the kernel. According to our subjective listening tests, multiple echoes give a better acoustic and using smoothed mixer signal improves imperceptibility that hidden data is almost inaudible.

Echo hiding methods are so robust that hidden data is not lost even when extra information is embedded with LSB Coding or Spread Spectrum. However, this costs as low payload capacity. When L is chosen as 1024, payload capacity is around 43 bps for a signal sampled at 44.1 kHz, and it drops when L is chosen greater.

Data losses are observed especially in segments with silent points. Silent segments can be forced by increasing their energy or these segments can be skipped by detecting before embedding in order to drop error rate.

Recommendations

It is very helpful to study why and how cepstrum analysis is used for decoding to understand the main idea of echo hiding method. See appendix of paper by Gruhl et al. (1996). A reference of github repository with some source codes written in Matlab is shared with the readers (Tekeli, 2017).

References

Anderson, R., & Petitcolas, F. (1998). On the limits of steganography. IEEE Journal on Selected Areas in Communications, 16(4), 474-481.

Bender, W., Gruhl, D., & Morimoto, N. (1996). Techniques for data hiding. IBM Systems Journal, 35(3), 313-336.

Chen, B., & Wornell, G.W. (2001). Quantization index modulation: a class of provably good methods for digital watermarking and information embedding. IEEE Transactions on Information Theory, 47(4), 1423-1443.

Gruhl, D., Lu, A., & Bender, W. (1996). Echo hiding. Proceeding from the Information Hiding Workshop, 295-315.

Kim, H.J., & Choi, Y.H. (2003). A novel echo-hiding scheme with backward and forward kernels. IEEE Transactions on Circuits and Systems for Video Technology, 13(8), 885-889.

Kirovski, D., & Malvar, H.S. (2001). Robust spread spectrum audio watermarking. Proceeding from the IEEE International Conference on Acoustics, Speech, and Signal Processing, 3, 1345-1348.

Ko, B., Nishimura, R., & Suzuki, Y. (2005). Time-spread echo method for digital audio watermarking. IEEE Transactions on Multimedia, 7(2), 212-221.

Oh, O.H., Seok, W.J., Hong, W.J., & Youn, H.D. (2001). New echo embedding technique for robust and imperceptible audio watermarking. Proceeding from ICASSP, 3, 1341-1344.

Tekeli, K. (2017). Matlab Codes for Several Basic Audio Steganography Algorithms. Retrieved from https://github.com/ktekeli/audio-steganography-algorithms

Wu, W.-C., & Chen, O. T.-C., (2006). Analysis-by-synthesis echo hiding scheme using mirrored kernels. Proceeding from IEEE International Conf on Acoustic, Speech & Signal Processing, II 325-II 328.

Yeo, I.K., & Kim, H.J. (2003). Modified Patchwork Algorithm: A novel audio watermarking scheme. IEEE Transactions on Speech and Audio Processing, 11, 381-386.