A ROBUST AUDIO WATERMARKING IN CEPSTRUM DOMAIN COMPOSED OF SAMPLE'S RELATION DEPENDENT EMBEDDING AND COMPUTATIONALLY SIMPLE EXTRACTION PHASE

Alok Kumar Chowdhury[1], Md. Ibrahim Khan[2] and Kaushik Deb[3]

[1]Department of Computer Science & Engineering,
Premier University, Chittagong, Bangladesh
[2,3]Department of Computer Science & Engineering, Chittagong University of Engineering & Technology, Chittagong, Bangladesh

ABSTRACT

Watermark bits embedded in audio signals considering the sample's relative state in a frame may strengthen the attack-invariant features of audio watermarking algorithm. In this work, we propose to embed watermarks in an audio signal considering the relation between the mean values of consecutive groups of samples which shows robustness by overcoming common watermarking challenges. Here, we divide the host audio signal into equal-sized non-overlapping frames which in turn is split into four equal-sized non-overlapping sub-frames. After, transforming these sub-frames in cepstrum domain we finally use the relation between the differences of first two sub-frames and last two sub-frames to embed watermarks. Depending on the watermark bit (either 0 or 1) to be embed, our embedding technique either interchange or update the differences between these groups of samples by distorting the sample values in sub-frames selectively. Thus, watermarks are embedded by making a little or no distortion of the sub-frames which helps our scheme to be imperceptible in nature. Moreover, use of such embedding technique lead our watermarking scheme to a computationally less complex extraction method. Simulation results also justify our claim of the proposed scheme to be both robust and imperceptible.

KEYWORDS

Audio watermarking, copyright protection, cepstrum domain, differences of mean, frame, group of samples, mean, relative states, sub-frames, transform domain

1. INTRODUCTION

We, the mass people, are enjoying the blessings of modern human-network through the diversified use of internet which ensures high availability, easy sharing, distribution and storing of digital content specially the audio, etc. Besides, the availability, popularity, high-end GUI of smart devices like smart phone, tab, notebook, etc, are adding more people in this network very rapidly. But, the heart of this digital mega world i.e. the contents owner or providers are facing more newer and complex challenges in protecting the copyright of their contents. As always, under the light there is always a shadow, the blessings of internet help unethical or bad people to reproduce unauthorized copies and illegally distribute these copies without the consent of the owner. The monetary loss or the extra efforts needed to protect the originality are frustrating content owners or proprietors of the music. Internet based multimedia systems and widespread deployments of computers are adding more security threats to the copyright protection. As a consequence, some newer techniques (Watermarking) are developing to protect the copyright of the owner by injecting the owner's signature or symbol (i.e., a watermark) into the multimedia.

DOI : 10.5121/ijma.2014.6207

Digital watermarking has many applications such as copyright notification and enforcement [1, 2], authentication [3], hybrid transmission applications [4], and covert communication [5].

An audio converted to or represented in digital format usually stored in Waveform Audio File (WAV) system. The common digital audio watermarking system, shown in Figure 1, try to hide some logical information i.e. watermark in a digitally represented audio signal and then disclose or uncover the watermark at the receiving end. A watermark, which uniquely identifies the content owner, should represent the presence of the content owner in terms of characters, a logo, a signature, an image, etc.

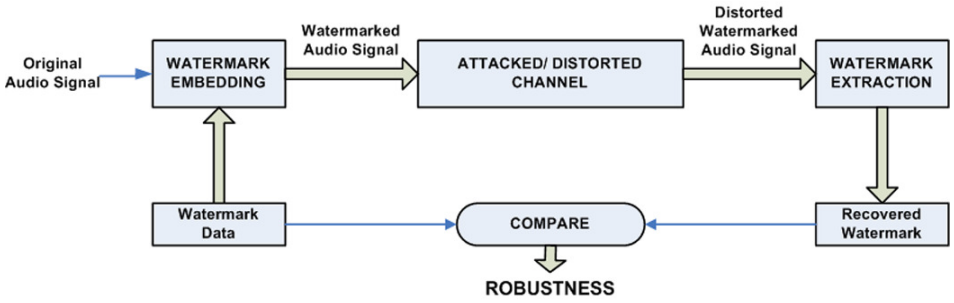

Figure 1. The common digital audio watermarking system

One of the main concerns of the content owner or distributors is imperceptibility, i.e. the presence or adding of watermark should not disfigure the original audio significantly. Robustness, another most important requirement needs that the hidden information (watermark) should be extractable at the receiving end despite of various common audio signal processing operations (re-sampling , re-quantization, MP3 compression, cropping, time stretching, etc.) and various other intentional attacks. In addition to these thumb requirements, the International Federation of Phonographic Industry (IFPI) [6] imposes other audio watermarking requirements are capacity and security. The former refers to the number of bits that can be embedded into the audio signal within a unit of time and the later implies that only legitimate receivers can detect or extract the watermark data by statistical analysis.

If the amount of distortion added to the audio samples is noteworthy then the rate of extraction of the watermark at the receiving end increases. But, at the same time this strategy annoys the audience more by making the watermark perceptible to them. Thus, satisfying these contradictory requirements is the main challenge of digital audio watermarking and hence a trade-off between them is needed for good quality audio watermarking.

In this work, we propose to divide the host audio signal into equal-sized non-overlapping frames which in turn is split into four equal-sized non-overlapping sub-frames. After being converted to their corresponding cepstrum representation, these sub-frames are acted upon by our embedding algorithm to find the differences between the mean values of first two and last two consecutive groups of samples i.e. sub-frames. Depending on the watermark bit (either 0 or 1) to be embed, our embedding technique either interchange or update the differences between these first and last groups of samples by distorting the sample values in each sub-frame selectively or left the sub-frames unchanged. Thus, watermarks are embedded by making a little or no distortion of the sub-frames which helps our scheme to be highly imperceptible in nature. Moreover, the relations between the sub-frames or consecutive groups of samples are not affected by most of the common signal processing attacks hence making the algorithm robust. Simulation results also justify our claim of the proposed scheme to be both robust and imperceptible.

2. RELATED WORKS

Several techniques [1-34] currently exist for embedding in digital watermarking. Research interests for watermarking in audio [8-34] are new compared to that of video [2] and image [7]. The embedding schemes for audio watermarking can be classified either based on domain transformation (time domain–based scheme [8-12] and transform domain–based schemes [16-32]) or based on the necessity of host signal during extraction (blind schemes and non-blind schemes).

Blind schemes do not require the original watermark data during extraction; on the other hand, non-blind schemes require the original watermark to extract correct watermarks. Though non-blind schemes are theoretically interesting but not useful in practical applications because they require double bandwidth and storage during detection.

Time domain–based watermarking is easy to implement and most of them can retrieve watermark without referencing the original audio. The main disadvantage of time-domain based schemes is its immunity to manipulations. The simplest time-domain based watermarking is least significant bit [8] method which simply uses the least significant bits of the host audio signal to embed watermark data. But this scheme fails and cannot tolerate most signal processing attacks. Some more audio watermarking schemes in time-domain were reported in the literature. Moderately robust time domain based spread spectrum technique was proposed by Bassia et al. [9]. Foote et al. [10], proposed a non-blind watermarking scheme using time-based modulation. Lie et al. [11] proposed another time-domain based watermarking scheme where they used differential average-of-absolute-amplitude relations within each group of samples to represent one-bit information. To preserve the time-domain waveform envelope they employed low-frequency amplitude modification to scale amplitudes in a group manner in selected sections of samples. Lemma et al. [12] proposed modified audio signal keying (MASK) to modify the short-time envelope of the audio signal in an imperceptible approach. According to them, their algorithm is robust and retains good audibility.

Echo hiding based watermarking techniques embed the watermark by introducing an echo [13]. However, echo hiding can effectively embed watermarks in imperceptible way [14, 15] but in such schemes watermark can be easily detected by anyone.

Transform domain–based schemes are currently considered a hot topic in audio watermarking. Audio watermarking in several transform domain like discrete cosine transform (DCT), discrete Fourier transform (DFT), modified discrete cosine transform, cepstrum transformation, etc. can spread watermark by embedding energy to several samples. Therefore, transform domain-based watermarking schemes can preserve most of the watermarks than that of time domain-based schemes. Guo et al. [16] proposed DCT domain–based audio signal watermarking that shows very good robustness, especially against resampling and MP3 compression, and high perceptual quality having signal-to-noise ratio (SNR) of more than 20 dB. Additionally, they also encoded watermarking information with BCH coding before embedding it into the double DCT transform of the host audio signal. They reported the resistance of their scheme against A/D and D/A conversions. Yeo et al. [17] presented an enhanced version of conventional patchwork algorithm, modified patchwork algorithm (MPA), for DCT, DFT and DWT domains. MPA performs well for MP3 and AAC compressions but it still needs refinements and requires to be integrated with other algorithms.

BCH code–based audio watermarking in the cepstrum domain was proposed by Liu et al. [19]. By employing both the attack-invariant feature of the cepstrum domain and the self–error-correction capability of BCH code they exhibit robustness against most asynchronous attacks compared to other well-known techniques. But using this scheme, the receiver should know the statistical

mean of the host signal during extraction which makes their scheme non-blind. Zhang et al. [20] also presented cepstrum domain–based audio watermarking algorithm using block encoding to improve recovery rate at the expense of lowering the maximum communication rate. Vivekananda et al. [21] decomposed the frame in the cepstrum domain to a two-level wavelet transform to access the low frequencies in which to embed the watermark. Their scheme utilizes error correcting code and cepstrum transform to lower the BER of the extracted watermark. Li et al. [22] presented an embedding method by updating the statistical mean of selected cepstrum coefficients. Their scheme was not very sensitive to audio asynchronism.

Brian et al. [29] introduced three new embedding methods: quantization index modulation (QIM), distortion-compensated QIM, and dither modulation. Compared to spread-spectrum and low-bit(s) modulation methods, QIM achieves better rate distortion–robustness trade-offs and is also good against arbitrary and fully informed attacks. A blind audio watermarking in DWT domain using QIM was proposed by Meng et al. [30]. Vivekananda et al. [31] also presented a similar blind mean quantization method in the cepstrum domain. Both algorithms proposed in [30, 31], need to set optimum value of quantization parameter to make the scheme robust and imperceptible. Another computationally less complex DFT domain–based audio watermarking scheme using quantization was proposed by Tan W. et al. [32]. Though, they reduced the mean square error to minimize embedding distortion, they did not provide any criteria to determine the optimal quantization step.

The previous work we proposed in [34] shows that watermark bits embedded in audio signals considering the sample's relative state in a frame are robust against noise or attacks. Though embedding was done in the cepstrum domain but any domain could have been used for embedding watermarks considering the relation between the groups of samples. However, embedding in cepstrum domain using such strategy seems more stable after attacks.

Most schemes proposed either in the time domain or the transform domain need to determine the amount of distortion (e.g. quantization step size [30, 31]) and the receiver to know the amount of distortion added during the embedding period. This legacy technique is one of the major causes for preventing watermark extraction schemes from being easy and faster. But, we have designed our proposed algorithm in such a way where the embedding algorithm takes the responsibility of all computations thereby leaving the extraction algorithm simple and hence making the proposed watermarking procedure efficient. Moreover, our embedding algorithm sets the distortion amount automatically from the relation among groups of samples and receiver need not know the amount of distortion during the extraction.

3. CEPSTRUM DOMAIN

Cepstrum transform, shown in Figure 2, involves three consecutive linear steps; take the fast Fourier transform first followed by the logarithm and finally take the inverse fast Fourier transform of the host signals.

The same time domain signals can be recovered from the cepstrum represented signals by inverse cepstrum transform, shown in Figure 3, which consists of exact reverse steps of embedding i.e. take the inverse fast Fourier transform first, later take the exponential followed by the fast Fourier transform.

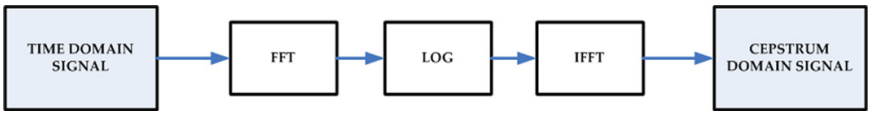

Figure 2. Cepstrum Domain Transformation

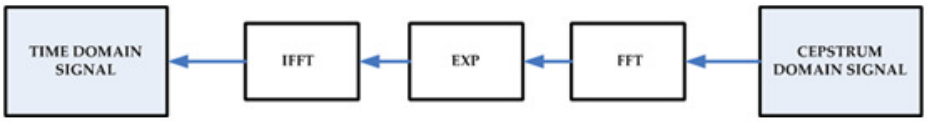

Figure 3. Inverse Cepstrum Domain Transformation

Cepstrum domain transform shows less variance against most common signal processing attacks and due to this attack-invariant feature cepstrum domain is one of major transform domains for audio watermarking chosen by the researcher in these days [23].

4. PROPOSED WATERMARKING ALGORITHM

In [34], we proposed a faster and efficient watermarking algorithm considering only the relation between the mean values of two consecutive sample groups. There, we calculated the mean values of each group of samples individually and then interchanged the mean values to embed watermarks. But, the extraction algorithm used at the receiver was very easy for an intruder to literally disclose the embedding algorithm. Here, we propose a revised watermark embedding and extraction algorithm inspired from [34] but using four groups of samples instead of two and considering the relation between the 'differences of mean values' instead of 'mean values' only to embed watermarks. First, we divide each frame of a host audio signal into four equal-sized sub-frames and then individually convert each sub-frame into cepstrum representation. After that, we calculate the differences between the mean values of first two sub-frames (d_1) and last two sub-frames (d_2) respectively. Later, depending on the watermark bit (either 0 or 1), we interchange the values of d_1 and d_2 to reflect the embedded information. We selectively change the sample values of the sub-frames in such a way that the value of d_1 is higher than that of d_2 when the watermark bit is 1, and vice-versa when it is 0. Finally, during extraction, we calculate the values of d_1 and d_2 again and use the relation of their values to extract a watermark bit (i.e., we treat the extracted watermark as 0 when d_1 is lower than d_2, and treat it as 1 in the opposite case). In this section, we describe our proposed watermark embedding algorithm followed by our watermark extraction algorithm.

4.1 Watermark Embedding Algorithm

We consider the bits of a binary image as watermarks and embed the bits into the host audio signal using the following steps.

Step 1) Preparation of Watermark

A binary image B of size $M \times N$ is used as the watermark data. B can be given by $B= \{b(m, n) : 1 \le m \le M, 1 \le n \le N, b(m, n) \in \{0, 1\}\}$. We prepare the watermark to be embedded by converting the two-dimensional binary image to a one-dimensional sequence of $M \times N$ bits. Thus, the watermark is $W= \{w(k) = b(m, n) : 1 \le m \le M, 1 \le n \le N, k = (m-1) \times N + n, 1 \le k \le M \times N, w=\{0, 1\}\}$. W is further encrypted using random permutation to ensure security.

Step 2) Frame

The host audio signal, sampled at 44,100 Hz, is divided into n equal-sized non-overlapping frames. One watermark bit is embedded in each of these n-sized frames unless otherwise mentioned.

Step 3) Sub-frames

Again, each frame is further divided into four equal-sized non-overlapping sub-frames with a sub-frame having a size of $n/4$ samples.

Step 4) Cepstrum Transform

As we observing [34] the cepstrum domain representation of audio signals exhibits more attack- or noise-invariant quality during audio watermarking, here again the sub-frames are individually transformed to their corresponding cepstrum domain.

Step 5) Embedding Technique

Step 5.1 Difference of Mean Values

The mean value of the samples in a sub-frame *i* of a frame is calculated using eq. (1) which in turn is used to find the differences between the mean values of sub-frames. The difference between the mean values of the first two sub-frames is denoted by d_1 and calculated using eq. (2).

$$mean_i = \frac{\sum_{k=LB}^{UB} x_k}{n/4} \qquad (1)$$

Where, x_k is the cepstrum coefficient of the k^{th} sample, *LB* and *UB* are the start and end of a sub-frame respectively and thus $UB - LB = n/4$ and $i = 1, 2, 3, 4$.

$$d_1 = \left| mean_1 - mean_2 \right| \qquad (2)$$

The difference between the mean values of the last two sub-frames of a frame is denoted by d_2 and calculated using eq. (3).

$$d_2 = \left| mean_3 - mean_4 \right| \qquad (3)$$

Step 5.2 Watermark Bit Embedding

The relation between d_1 and d_2 of a frame can be described by one of the following three cases, $d_1 < d_2$, $d_1 > d_2$ or $d_1 = d_2$.

Depending on the type of the cases and the value of the watermark bit to be embedded (either 0 or 1), the following techniques can be used to insert the watermark and change the values of d_1 and d_2 accordingly.

Step 5.2a) CASE: $d_1 < d_2$

i) Watermark Bit 0

In this case, the samples of the four sub-frames of a frame are all left intact. Thus, we have an advantage of embedding watermark 0 in this case without changing any of the sample values in that frame.

ii) Watermark Bit 1

In this case, the values of d_1 and d_2 are interchanged. To set the new value of d_1 to that of d_2, the mean values of the first two sub-frames (sub-frame1 and sub-frame2) of a frame are altered using the rules given in Algorithm 1.

Algorithm 1. increase d_1 to d_2

```
//increase d1 to d2
displacement := (d1-d2)/2
if mean1>mean2 then
   setmean(sub-frame1, sub-frame2, displacement)
else
   setmean(sub-frame2, sub-frame1, displacement)
endif
```

Here, the *setmean* procedure takes first two sub-frames, corresponding displacement parameter (in this case, $(d_1- d_2)/2$) and based on the sign of displacement (in this case negative) the procedure increases the value of d_1 by the amount of absolute value of the displacement.

Again, to set the new value of d_2 to that of d_1, the mean values of the last two sub-frames (*sub-frame3* and *sub-frame4*) are altered using the following rules (Algorithm 2).

Algorithm 2. decrease d_2 to d_1

```
//decrease d2 to d1
displacement := (d2-d1)/2
if mean3>mean4 then
   setmean(sub-frame3, sub-frame4, displacement)
else
   setmean(sub-frame4, sub-frame3, displacement)
endif
```

Here, the *setmean* procedure takes last two sub-frames, corresponding displacement parameter (in this case, $(d_2- d_1)/2$) and based on the sign of displacement (in this case positive) the procedure decreases the value of d_2 by the amount of absolute value of the displacement.

The general form or body of the *setmean* procedure is defined in step 5.3. Therefore, we do not describe or explain the *setmean* procedure for the cases $d_1>d_2$ and $d_1=d_2$.

Step 5.2b) CASE: $d_1>d_2$

i) Watermark Bit 0

In this case, the values of d_1 and d_2 are interchanged. To set the new value of d_1 to that of d_2, the mean values of the first two sub-frames (*sub-frame1* and *sub-frame2*) of a frame are altered using Algorithm 3.

Algorithm 3. decrease d_1 to d_2

```
//decrease d1 to d2
displacement := (d1- d2)/2
if mean1>mean2 then
   setmean(sub-frame1, sub-frame2, displacement)
else
   setmean(sub-frame2, sub-frame1, displacement)
endif
```

And to set the new value of d_2 to that of d_1, the mean values of the last two sub-frames (*sub*-frame3 and sub-frame4) are altered using the rules of Algorithm 4.

Algorithm 4. increase d_2 to d_1

```
//increase d2 to d1
displacement := (d2- d1)/2
if mean3>mean4 then
   setmean(sub-frame3, sub-frame4, displacement)
else
   setmean(sub-frame4, sub-frame3, displacement)
endif
```

ii) Watermark Bit 1

In this case, the samples of the four sub-frames of a frame are all left unchanged. Thus, here again we have the same advantage of embedding watermark 1 without altering any of the sample values in that frame.

Step 5.2c) CASE: $d_1=d_2$

i) Watermark Bit 0

In this case, with the value of d_1 equal to d_2, it is unnecessary to use the rules to exchange the differences. But, to keep the extraction algorithm straightforward, as generally it is, we increase the value of d_2 and decrease the value of d_1 by introducing a control parameter *c,* shown in Algorithm 5. The selection of the value of *c* has a significant impact on the robustness-imperceptibility tradeoff of our algorithm therefore the value of *c* could be determined during the implementation. Despite of the importance of the control parameter *c* the receiver yet not need to know its value**.**

Algorithm 5. increase d_2 and decrease d_1

```
//decrease d1
 displacement := d1/c
if mean1>mean2 then
   setmean(sub-frame1, sub-frame2, displacement)
else
   setmean(sub-frame2, sub-frame1, displacement)
endif


//increase d2
displacement := - d2/c
if mean3>mean4 then
   setmean(sub-frame3, sub-frame4, displacement)
else
   setmean(sub-frame4, sub-frame3, displacement)
endif
```

ii) Watermark Bit 1

Here, we use the same control parameter *c* but to increase the value of d_1 and decrease the value of d_2. Algorithm 6 shows the rules for this case.

Algorithm 6. increase d_1 and decrease d_2

```
//increase d1
displacement := - d1/c
if mean1>mean2 then
   setmean(sub-frame1, sub-frame2, displacement)
else
   setmean(sub-frame2, sub-frame1, displacement)
endif


//decrease d2
displacement := d2/c
if mean3>mean4 then
   setmean(sub-frame3, sub-frame4, displacement)
else
   setmean(sub-frame4, sub-frame3, displacement)
endif
```

Step 5.3 The setmean Function

In this section, the pseudo code of the general form of the *setmean* function, shown in Algorithm 7, is described. The *setmean* procedure receives two sub-frames, larger one and the smaller one based on their mean values, as the value of its first two arguments (sf_{large}, sf_{small}) respectively and a value as its third argument (displacement). Depending on the value and sign of the displacement, the *setmean* procedure either increases or decreases the difference between the mean values of the two received sub-frames namely sf_{large} and sf_{small}.

When the displacement is positive the *setmean* decreases the difference between the mean values of sf_{large} and sf_{small} by lowering the mean of sf_{large} and raising the mean of sf_{small} by the amount of displacement. And, when the displacement is negative the *setmean* procedure increases the difference between sf_{large} and sf_{small} by raising the mean of sf_{large} and lowering the mean of sf_{small} by the amount of displacement. To make the amount of distortion imperceptible only the positive samples of a sub-frame are altered in case of decreasing the mean of that sub-frame and negative samples in case of increasing the mean value.

Step 6) Inverse Cepstrum and Merge

The sub-frames transformed to their cepstrum representation using step 4 are now transformed back to their original time domain representation using inverse cepstrum domain transform and then merged together to reform the actual frame.

Algorithm 7. The setmean Procedure

setmean (sf_{large}, sf_{small}, displacement)

if displacement>0 then

//decrease the difference between the mean values

for $i = 1$ *to* $length(sf_{large})$

if $(sf_{large,i} > 0)$ *then*

$$sf_{large,i} = sf_{large,i} - \frac{|displacement|}{no\_of\_positive\_sample} * length(sf_{large})$$

endif

endfor

for $i = 1$ *to* $length(sf_{small})$

if $(sf_{small,i} < 0)$ *then*

$$sf_{small,i} = sf_{small,i} + \frac{|displacement|}{no\_of\_negative\_sample} * length(sf_{small})$$

endif

endfor

else

//increase the difference between the mean values

for $i = 1$ *to* $length(sf_{large})$

if $(sf_{large,i} < 0)$ *then*

$$sf_{large,i} = sf_{large,i} + \frac{|displacement|}{no\_of\_negative\_sample} * length(sf_{large})$$

endif

endfor

for $i = 1$ *to* $length(sf_{small})$

if $(sf_{small,i} > 0)$ *then*

$$sf_{small,i} = sf_{small,i} - \frac{|displacement|}{no\_of\_positive\_sample} * length(sf_{small})$$

endif

endfor

endif

return Sf_{large} *and* Sf_{small}

//Here, $Sf_{large,i}$ and $Sf_{small,i}$ are the i^{th} sample of Sf_{large} and //Sf_{small} respectively.

4.2 Watermark Extraction Algorithm

Unlike [34], the proposed watermarking extraction algorithm requires a bit more logical operation to extract a watermark which helps the algorithm to be more secured compared to [34]. The steps involved in the extraction process are described as follows.

Step 1) Frame

The watermarked audio signal is divided into *n* sized non-overlapping frames where each frame is expected to carry a single watermark bit.

Step 2) Sub-frames

In this step, each frame is again split into four equal-sized non-overlapping groups of samples i.e. sub-frames of size *n/4* samples each.

Step 3) Cepstrum Transform

Here, each sub-frame is transformed to their corresponding cepstrum domain.

Step 4) Extraction Technique

The mean values of all four sub-frames are calculated using eq. (1) and then the differences between the first two sub-frames (d_1) and last two sub-frames (d_2) are computed using eq. (2) and (3) respectively.

As the embedding algorithm always make the value d_1 lower than d_2 in case of embedding watermark bit 0 and vice versa for embedding watermark bit 1, our extraction algorithm identify the watermark bit by simply comparing the value of d_1 and d_2. So, a watermark bit, eb_k, can be identified using the following method.

$$eb_k = \begin{cases} 0 & if\ (d_1 < d_2) \\ 1 & otherwise \end{cases}$$

Step 5) Reform Binary Image

The extracted one-dimensional watermark information (*eb*) is decrypted by the use of the same random permutation used in embedding process. And finally, the binary image (*B'*) is reformed again by converting decrypted one-dimensional watermark data to two-dimensional binary sequences. Thus extracted binary image (*B'*) is $B' = \{b'(m, n) = eb_k : 1 \leq k \leq M \times N,\ m = (k-1)/N + 1,\ n = k-(m-1)\times N,\ eb_k = \{0, 1\}\}$.

5. EXPERIMENTAL RESULT

In order to assess the performance of our proposed algorithm we performed various experiments against common signal processing attacks, such as amplitude modification, resampling, re-quantization, noise addition, MP3 conversion, etc. Commonly used digital audio and binary image were used as host audio signals and watermark respectively. Since, this work has greatly been motivated from [34] we have kept all the host signals and types of attacks same as [34] to draw a performance comparison chart to claim the superiority of this work.

5.1 Experimental Data

Same as [34] we use the same five (5) different types of audio signal as the host signals. A Waveform Audio File (WAV) format for two pop songs, one classical piece, a country piece, and a human voice were selected as host signals. Each host audio signal was mono signals with a length of 60 seconds, the sampling frequency was 44.1 KHz and the sampling resolution was 16 bits. Plots of a small portion of the pop audio signal and the corresponding watermarked audio signal are shown in Figure 4. A binary image of size 40 X 20 pixels was used as watermark during the experiment, which is displayed in Figure 5.

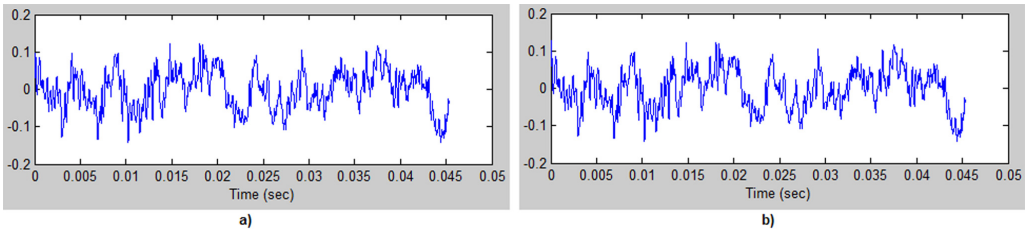

Figure 4. a) A portion of the host audio signal, and b) a portion of the watermarked audio signal.

Figure 5. Watermark binary image

5.2 Experimental Environment and Parameters

Our proposed scheme was simulated in MATLAB environment. During the experiment, we used frame size (n) to 100 samples, where each sub-frame contains 25 samples.

Adobe Audition CS 6 and MATLAB both were engaged to apply attacks on the watermarked audio. For all the simulation scenarios we found eight (8), as the value of ***c,*** was working better.

5.3 Experimental Assessment Standard

The trade-off between the two most important factors, robustness and imperceptibility, is the main challenge of an audio watermarking scheme. We used the performance evaluation metric [35], such as normalized correlation (NC), bit error rate (BER) and perceptual quality to assess the performance of proposed scheme. NC and BER of extracted watermark data i.e. binary images were used to measure the robustness and SNR was selected to evaluate imperceptibility of the watermark.

5.3.1 Normalized Correlation

Normalized Correlation provides the similarities between the extracted binary images, w' and original binary image, w. NC can be defined as

$$NC(w, w') = \frac{\sum_{i=0}^{m-1}\sum_{j=0}^{n-1} w(i,j)\, w'(i,j)}{\sqrt{\sum_{i=0}^{m-1}\sum_{j=0}^{n-1} w^2(i,j)} * \sqrt{\sum_{i=0}^{m-1}\sum_{j=0}^{n-1} w'^2(i,j)}}$$

where m and n are the number of rows and columns of the binary image. A higher NC indicates more similarity between w and w'.

5.3.2 Bit Error Rate

Bit Error Rate (BER) is used to find the percentage of errors between the embedded and extracted watermark i.e. binary images. BER is given by

$$BER = \frac{100 * \sum_{n=1}^{K} \begin{cases} 0, & w(n) = w'(n) \\ 1, & w(n) \neq w'(n) \end{cases}}{K}$$

where, w and w' are the binary data of embedded and extracted watermark images, respectively.

5.3.3 Signal-to-Noise Ratio

The Signal-to-Noise Ratio (SNR) of the watermarked audio, s', versus the host audio signal, s, can be defined as

$$SNR(s, s') = 10 \log_{10} \left\{ \frac{\sum_{n=0}^{N-1} s^2(n)}{\sum_{n=0}^{N-1} [s'(n) - s(n)]^2} \right\}$$

where N is the total number of samples in the audio signal. SNR is used to measure the perceptual quality i.e. inaudibility of the watermark. A higher SNR indicates higher perceptual quality.

5.4 The Results & Analysis

First, we have computed the SNR values of all watermark embedded audio signals to measure the perceptual quality of our scheme. Later, we have applied various signal processing tasks or attacks to the watermarked signals and then computed the NC and BER to test the robustness of our proposed algorithm. Figure 6 shows the NC and BER values for the watermark images extracted under different types of attacks.

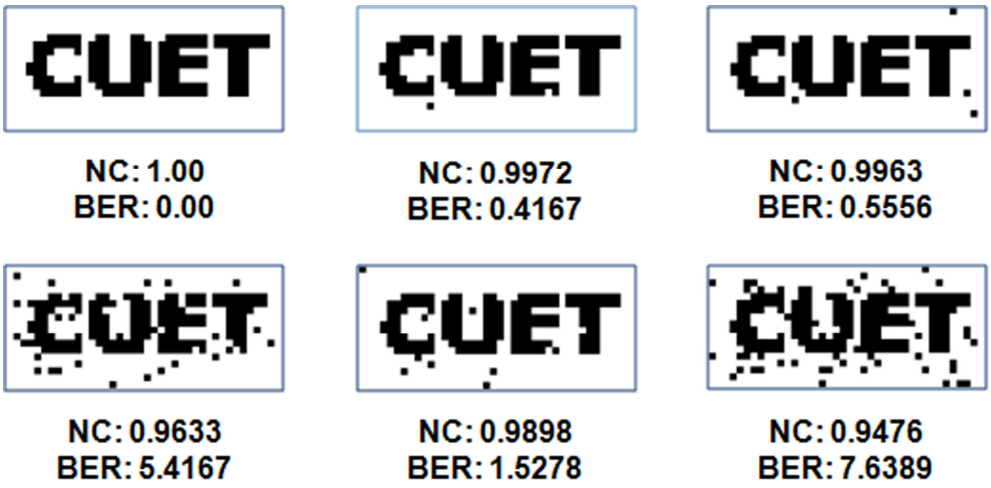

Figure 6. Extracted watermark images with various NC and BER values.

5.4.1 Perceptual Quality of Our Scheme

Subjected test was performed by taking opinion from the same 10 undergraduate student as in [34]. According to their report they didn't found any perceptual deviation from the original host signals. And, the SNRs calculated using the experimental data were greater than 25dB for each song. We got the above results since watermark bits are embedded either by little or no distortion

to the samples of host signals. Thus, both these experimental outcomes conform to the imperceptibility of our scheme.

5.4.2 Robustness without Attack

Our proposed scheme showed 100% success in watermark detection when no attack is applied. Table 1 shows the NC and BER of our proposed scheme and our previous scheme proposed in [34] when no process is applied. In every cases, both scheme extracted watermark without any dissimilarities compared to original watermark.

Table 1. NC and BER of extracted watermark without any attack

Audio	Parameters	[34]		Proposed Algorithm	
		NC	BER	NC	BER
Pop 1	-	1	0	1	0
Pop 2	-	1	0	1	0
Classic	-	1	0	1	0
Countr	-	1	0	1	0
Human	-	1	0	1	0

5.4.3 Robustness against Amplitude Modification Attacks

We computed NC and BER of extracted watermark for each audio signal after applying various amplitude modification attacks such as +3 dB boost, -3 dB cut, +10 dB boost, -10 dB cut, +5% increases in amplitude and +10% increases in amplitude. Table 2 shows that both proposed algorithm and the algorithm proposed in [34] performed extraordinarily well and against amplitude modification attacks.

Table 2. NC and BER of extracted watermark after applying amplitude modification

Audio	Parameters	[34]		Proposed Algorithm	
		NC	BER	NC	BER
Pop 1	-3 dB	0.99	0.6944	0.99	0.416
	+3 dB	0.99	1.1111	0.98	2.778
	-10 dB	0.99	0.8333	0.99	0.277
	+10 dB	0.98	2.6389	0.94	7.638
	+5%	0.99	1.25	0.98	2.222
	+10%	0.99	1.3889	0.98	2.083
Pop 2	-3 dB	0.99	0.8333	0.99	0.138
	+3 dB	0.99	0.9722	0.98	2.638
	-10 dB	0.99	0.8333	0.99	0.138
	+10 dB	0.99	0.8333	0.95	6.666
	+5%	0.99	1.1111	0.99	1.25
	+10%	0.99	1.1111	0.99	1.25
Classic	-3 dB	0.99	0.5556	1	0
	+3 dB	0.99	0.4167	0.99	0.555
	-10 dB	0.99	0.8333	0.99	0.138

	+10 dB	0.99	0.6944	0.98	1.666
	+5%	0.99	1.25	0.99	0.138
	+10%	0.99	1.25	0.99	0.138
Countr	-3 dB	0.98	1.9444	1	0
	+3 dB	0.99	1.1111	0.99	1.388
	-10 dB	0.98	1.9444	1	0
	+10 dB	0.98	1.9444	0.95	6.25
	+5%	0.98	2.0833	0.99	0.416
	+10%	0.98	2.0833	0.99	0.555
Human	-3 dB	0.98	1.9444	0.99	0.416
	+3 dB	0.99	1.1111	0.98	1.666
	-10 dB	0.99	1.3889	0.99	0.277
	+10 dB	0.99	1.3889	0.95	6.111
	+5%	0.98	1.9444	0.99	1.25
	+10%	0.98	1.9444	0.99	0.972

5.4.4 Robustness against Resampling

The watermarked audio signal, sampled at 44.1 KHz, was resampled to 22.50/32/96 KHz, and again resampled back to 44.1 KHz. Table 3 shows that our proposed scheme is robust against resampling and shows similar characteristics (NC and BER) like [34].

Table 3. NC and BER of extracted watermark after resampling

Audio	**Parameters (KHz)**	**[34]**		**Proposed Algorithm**	
		NC	**BER**	**NC**	**BER**
Pop 1	44100 – 22050- 44100	0.97	3.4722	0.97	3.472
	44100 – 32000- 44100	0.98	2.2222	0.98	1.805
	44100 – 96000- 44100	1	0	0.99	0.277
Pop 2	44100 – 22050- 44100	0.98	1.9444	0.97	3.611
	44100 – 32000- 44100	0.99	1.3889	0.99	1.25
	44100 – 96000- 44100	1	0	0.99	0.138
Classic	44100 – 22050- 44100	0.99	1.3889	0.97	3.888
	44100 – 32000- 44100	0.98	1.5278	0.98	1.944
	44100 – 96000- 44100	1	0	0.99	0.138
Countr	44100 – 22050- 44100	0.99	0.6944	0.98	2.916
	44100 – 32000- 44100	0.99	0.5556	0.98	1.527
	44100 – 96000- 44100	1	0	1	0
Human	44100 – 22050- 44100	0.97	4.0278	0.96	4.861
	44100 – 32000- 44100	0.97	3.3333	0.98	2.083
	44100 – 96000- 44100	1	0	0.99	0.555

5.4.5 Robustness against Re-quantization

We re-quantized 16-bit watermarked audio signal to 8 bit/32 bit, and then again converted to 16-bit. Table 4 provides the NC and BER of the extracted image. Re-quantization results in loss of information, but our scheme is capable of preserving most of the embedded watermark data and

outperform the result of watermarking algorithm proposed in [34].

Table 4. NC and BER of extracted watermark after re-quantization

Audio	Parameters	[34]		Proposed Algorithm	
		NC	BER	NC	BER
Pop 1	16 bits - 8 bits -16 bits	0.95	7.0833	0.97	3.3333
	16 bits - 32 bits -16 bits	0.99	0.4167	0.99	0.2778
Pop 2	16 bits - 8 bits -16 bits	0.77	30.417	0.96	4.7222
	16 bits - 32 bits -16 bits	0.99	0.8333	0.99	0.4167
Classic	16 bits - 8 bits -16 bits	0.82	23.472	0.95	6.3889
	16 bits - 32 bits -16 bits	0.99	0.5556	1	0
Countr	16 bits - 8 bits -16 bits	0.85	19.861	0.97	3.0556
	16 bits - 32 bits -16 bits	0.99	0.6944	1	0
Human	16 bits - 8 bits -16 bits	0.77	30.139	0.93	9.1667
	16 bits - 32 bits -16 bits	0.99	1.25	0.99	0.4167

5.4.6 Robustness against MP3 Compression

We applied MP3 compression, a most popular audio compression technique, to the WAV format of the watermarked audio signals at a rate of 64 kbps, 128 kbps and 256 kbps. Finally, we converted the MP3 format back to WAV format. Table 5 shows that proposed algorithm has high robustness against MP3 conversion.

Table 5. NC and BER of extracted watermark after applying MP3 attacks

Audio	Parameters	[34]		Proposed Algorithm	
		NC	BER	NC	BER
Pop 1	64 kbps	0.91	12.222	0.91	12.63
	128 kbps	0.96	5.4167	0.95	6.666
	256 kbps	0.98	2.2222	0.96	4.583
Pop 2	64 kbps	0.94	7.3611	0.93	9.444
	128 kbps	0.97	3.4722	0.95	6.25
	256 kbps	0.99	1.3889	0.97	4.444
Classic	64 kbps	0.93	9.1667	0.89	14.72
	128 kbps	0.97	3.0556	0.94	8.194
	256 kbps	0.98	1.6667	0.95	6.111
Countr	64 kbps	0.96	5.6944	0.93	9.166
	128 kbps	0.98	2.2222	0.94	7.638
	256 kbps	0.99	1.25	0.96	4.722
Human	64 kbps	0.88	16.25	0.90	13.61
	128 kbps	0.96	5.1389	0.95	7.361
	256 kbps	0.98	2.3611	0.97	4.305

5.4.7 Robustness against Low-pass Filtering Attacks

The watermarked audio signals were low-pass filtered with cut off frequencies of 4 KHz, 8 KHz and 11.3 KHz. Table 6 shows that our proposed algorithm performs exceptionally well in this case and beats the performance of previous algorithm [34] significantly in almost every case especially for cut off frequency of 4 KHz.

Table 6. NC and BER of extracted watermark after applying low-pass filtering

Audio	Parameters	[34]		Proposed Algorithm	
		NC	BER	NC	BER
Pop 1	4 KHz	0.79	27.917	0.96	5.416
	8 KHz	0.98	2.5	0.98	2.777
	11.3 KHz	0.98	2.5	0.98	2.222
Pop 2	4 KHz	0.74	34.861	0.94	7.777
	8 KHz	0.99	1.1111	0.98	2.777
	11.3 KHz	0.99	1.3889	0.98	1.944
Classic	4 KHz	0.76	31.528	0.94	8.611
	8 KHz	0.99	1.3889	0.97	3.472
	11.3 KHz	0.98	1.6667	0.98	2.500
Countr	4 KHz	0.72	37.083	0.95	6.805
	8 KHz	0.99	0.4167	0.98	2.361
	11.3 KHz	0.99	0.8333	0.98	1.805
Human	4 KHz	0.77	30.972	0.93	10.13
	8 KHz	0.97	3.0556	0.96	4.583
	11.3 KHz	0.97	3.4722	0.97	3.055

5.4.8 Robustness against AWGN

White Gaussian noise is added to the watermarked audio signals so the resulting signals have an SNR of 20 dB, 30 dB and 50 dB. Table 7 shows the result of our scheme against AWGN. Proposed algorithm showed better performance for most of the AWGN attack parameters compared to the previous algorithm [34].

Table 7. NC and BER of extracted watermark after applying AWGN

Audio	Parameters	[34]		Proposed Algorithm	
		NC	BER	NC	BER
Pop 1	20 dB	0.94	8.4722	0.96	5.277
	30 dB	0.98	2.7778	0.98	1.527
	50 dB	0.99	0.6944	0.99	0.555
Pop 2	20 dB	0.88	16.25	0.95	5.972
	30 dB	0.95	6.3889	0.98	2.916
	50 dB	0.99	1.25	0.99	0.416
Classic	20 dB	0.92	10.972	0.93	9.444
	30 dB	0.97	4.0278	0.97	3.333

	50 dB	0.99	0.5556	0.99	0.277
Countr	20 dB	0.91	12.5	0.94	7.638
	30 dB	0.98	2.5	0.98	2.916
	50 dB	0.99	0.1389	0.99	0.277
Human	20 dB	0.88	16.389	0.92	10.83
	30 dB	0.93	9.4444	0.96	5.555
	50 dB	0.98	1.9444	0.99	0.833

6. CONCLUSION

In this work, we proposed an efficient technique like [34] where it is possible to embed watermark bits (either 0 or 1) selectively without distorting or changing the original audio samples and hence increasing the imperceptibility of the technique. Relation between the groups of samples to embed watermark, used in [34], produced some outstanding results which eventually motivated us to work with newer types of relation among these groups of samples. These new relations being more stable than that of the previous one show more resistance despite of attacks during extraction period. First, we propose to divide a frame into four equal-sized non-overlapping sub-frames, second transforming these sub-frames in cepstrum domain, and finally we use the relation between the differences of first two sub-frames and last two sub-frames to embed watermarks. Depending on the watermark bit we either interchange or update the differences by distorting the samples in each sub-frame selectively or left the sub-frames unchanged. Simulation results also justify our claim of the proposed scheme to be both robust and imperceptible.

In spite of framing samples using our straight forward approach we plan to develop more intelligent frame forming methods that will include samples based on their salient features. Applying our proposed algorithm using such frames may lead to a very good quality audio and robust against synchronous attacks as well.

REFERENCES

[1] I.J. Cox, J.-P.M.G. Linnartz (1998) Some General Methods for Tampering with Watermarks. IEEE Journal on Selected Areas in Communications 16:587–593

[2] J.-P. Linnartz, T. Kalker, J. Haitsma (1999) Detecting Electronic Watermarks in Digital Video. Proc. of the 1999 IEEE International Conference on Acoustics, Speech, and Signal Processing 4:2071–52074

[3] D. Kundur, D. Hatzinakos (1999) Digital Watermarking for Telltale Tamper Proofing and Authentication. Proceedings of the IEEE 87:1167–1180

[4] B. Chen (2000) Design and Analysis of Digital Watermarking, Information Embedding, and Data Hiding Systems. Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge, MA

[5] F.A.P. Petitcolas, R.J. Anderson, M.G. Kuhn (1999) Information Hiding—A Survey. Proceedings of the IEEE 87:1062–1078

[6] International Federation of Phonographic Industry (IFPI) [Online]. Available: http://www.ifpi.org/

[7] Lenarczyk P., Piotrowski Z. (2010) Novel Hybrid Blind Digital Image Watermarking in Cepstrum and DCT Domain. International Conference on Multimedia Information Networking and Security (MINES), pp. 356 – 361

[8] Zhang Xiaoming (2009) Audio watermarking algorithm for public information transmission. Journal of Computer Applications 29:2323-2326

[9] Bassia P., Pitas I., Nikolaidis N. (2001) Robust audio watermarking in the time domain. IEEE Transactions on Multimedia 3:232-241

[10] Foote. J., Adcock J. (2003) Time base modulation: A new approach to watermarking audio and images. International Conference on Multimedia and Expo, ICME '03 1:221 – 224

[11] Lie W.N., Chang L.C. (2006) Robust and High-Quality Time-Domain Audio Watermarking Based on Low-Frequency Amplitude Modification. IEEE Transaction on Multimedia 8:46 – 59

[12] Lemma A.N., Aprea J., Oomen W., van de Kerkhof L. (2003) A Temporal Domain Audio Watermarking Technique. IEEE Transactions on Signal Processing 51:1088 – 1097

[13] D. Gruhl, A. Lu, W. Bender (1996) Echo hiding, lecture notes in computer science, Information Hiding 1174:295–315

[14] H.O. Oh, J.W. Seok, J.W. Huang, D.H. Youn (2001) New echo embedding technique for robust and imperceptible audio watermarking. Proceedings of International Conference on Acoustics, Speech, and Signal Processing 3:1341–1344

[15] K. Byeong-Seob, R. Nishimura, Y. Suzuki (2005) Time-spread echo method for digital audio watermarking. IEEE Transaction on Multimedia 7:212–221

[16] Guo Q., Zhao Y., Cheng P., Wang F. (2012) An Audio Digital Watermarking Algorithm Against A/D And D/A Conversions Based on DCT domain. International Conference on Consumer Electronics, Communications and Networks, pp. 871 – 876

[17] Yeo I.K., Kim H.J. (2003) Modified Patchwork Algorithm: A Novel Audio Watermarking Scheme. IEEE Transaction on Speech and Audio Processing 11:381-386,

[18] Ren K., Li H. (2011) Large Capacity Digital Audio Watermarking Algorithm Based on DWT and DCT. International Conference on Mechatronic Science, Electric Engineering and Computer (MEC), pp. 1765 – 1768

[19] Liu, S.C., Lin, S.D. (2006) BCH Code-Based Robust Audio Watermarking in the Cepstrum Do-main. Journal of Information Science and Engineering 22:535 – 543

[20] Zhang L., Huang Z., Cao X. (2011) A Digital Audio Watermarking Algorithm Based On Block Encoding. International Conference on Wireless Communications and Signal Processing, pp. 1-5

[21] Vivekananda B.K., Sengupta I., Das A. (2008) Audio Watermarking Based on BCH Coding using CT and DWT. International Conference on Information Technology, Bhubaneswar, pp. 49 – 50

[22] Li X., Yu H.H. (2000) Transparent and robust audio data hiding in subband domain. International Conference Information Technology: Coding and Computing, pp. 74 – 79

[23] S. Li, L. Cui, J. Choi, X. Cui (2006) An Audio Copyright Protection Schemes Based on SMM In Cepstrum Domain. International Workshops on Structural, Syntactic, and Statistical Pattern Recognition 4109:923–927

[24] Lee S.K., Ho Y.S. (2000) Digital Audio Watermarking in the Cepstrum Domain. IEEE Transactions on Consumer Electronics 46:744 – 750

[25] Boney L., Tewfik A. H., Hamdy K. N. (1996) Digital watermarks for audio signal. International Conference on Multimedia Computing and Systems, pp. 473-480

[26] Cox I. J., Kilian J., Leigton F. T., Shamoon T. (1996) Secure Spread Spectrum Watermarking for Multimedia. IEEE Transaction on Image Processing 6:1673-1687

[27] Cvejic. N., Keskinarkaus A., Seppanen T. (2001) Audio watermarking using m-sequences and temporal masking. IEEE Workshops on Applications of Signal Processing to Audio and Acoustics, pp. 227-230

[28] Seok J., Hong J., Kim, J. (2002) A novel audio watermarking algorithm for copyright protection of digital audio. ETRI Journal 24:181-189

[29] Chen B., Wornell G.W. (2001) Quantization Index Modulation: A Class of Provably Good Methods for Digital Watermarking and Information Embedding. IEEE Transaction on Information Theory 47:1423 – 1443

[30] Lihua M., Shuangyuan Y., Qingshan J. (2009) A New Algorithm for Digital Audio Watermarking Based on DWT. WRI Global Congress on Intelligent Systems 4:229 – 233

[31] Vivekananda B.K., Sengupta I., Das A. (2008) Audio Watermarking Based on Mean Quantization in Cepstrum Domain. International Conference on Advanced Computing and Communications, pp. 73 – 77

[32] Tan W., Yang S., Chen Y., Zhou J. (2009) Research on DFT Domain Digital Audio Watermarking Algorithm Based on Quantization. International Workshop on Education Technology and Computer Science 3:736 – 739

[33] Hyoung Joong Kim, Yong Hee Choi, JW Seok, JW Hong (2004) Audio Watermarking Techniques. Intelligent Watermarking Techniques, pp. 185-218

[34] Alok Kumar Chowdhury, Md. Ibrahim Khan (2013) A Tutorial for Audio Watermarking in the Cepstrum Domain. Smart Computing Review 3:323-335

[35] Gordy, J.D., Bruton, L.T. (2000) Performance evaluation of digital audio watermarking algorithms. Proceedings of the 43rd IEEE Midwest Symposium on Circuits and Systems 1:456-459

AUTHORS

Alok Kumar Chowdhury received his B.Sc. degree in Computer Science & Engineering from Chittagong University of Engineering & Technology in 2009. Currently, he is pursuing M.Sc in Computer Science & Engineering in the same University. Since 2009, he has been serving as a faculty member in the Department of Computer Science & Engineering at Premier University, Chittagong, Bangladesh. His research interest includes Artificial Intelligent, Computer Vision, Digital Image Processing, Digital Watermarking, Audio Signal Processing, Cloud Computing, etc.

Dr. Mohammad Ibrahim Khan received the B.Sc. degree in Electrical and Electronic Engineering from Bangladesh University of Engineering and Technology (BUET), Bangladesh in 1999. He received M.S. degree in Computer Science and Engineering from the same University in 2002. He received his Ph.D. degree in Computer Science and Engineering from Jahangirnagar University in 2010. Since 1999, he has been serving as a faculty member in the Department of Computer Science and Engineering at Chittagong University of Engineering & Technology (CUET), Chittagong, Bangladesh. His research interest includes Digital Image Processing, Graph Theory, Cryptography, Digital Watermarking, Multimedia Systems, and Digital Signal Processing.

Dr. Kaushik Deb received his B.Tech. and M.Tech. degrees from Department of Computer Science and Eng. of Tula State University, Tula, Russia, in 1999 and 2000, respectively. Since 2001, he has been serving as a faculty member in Department of Computer Science and Engg., Chittagong University of Engineering and Technology (CUET), Chittagong, Bangladesh. He received his Ph.D. from the Graduate School of Electrical Eng. and Information Systems, University of Ulsan, Ulsan, Korea in 2010. His research interests include computer vision, pattern recognition, intelligent transportation systems (ITSs) and human-computer interaction