iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 11 **Issue:** V **Month of publication:** May 2023

DOI: https://doi.org/10.22214/ijraset.2023.52343

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 11 Issue V May 2023- Available at www.ijraset.com

Imperceptible Technique to Secure Digital Audio Signals while Attaining Trade-Off among the Parameter of Magic Triangle

Asheesh Kumar Dwivedi[1], Mr. Neeraj Kumar[2]
[1] *M.Tech Scholar, Shri Ramswaroop Memorial University, Lucknow*
[2] *Assistant Professor, Shri Ramswaroop Memorial University, Lucknow*

Abstract: ***This paper outlines an audio encryption method that is used during audio compression. Security for the multimedia data is provided by encryption methods. Here, audio encryption is a method used to communicate secret information by subtly altering an audio stream. It is a technique that guarantees secure data transfer between parties that often interact online. Here, we offer a fresh method for dealing with issues raised by audio encryption's substitution mechanism. We employ an enhanced RSA encryption technique, which is incredibly difficult to crack, to encrypt messages in the first level of protection. The encrypted message bits are embedded into random LSB layers to boost robustness against purposeful assaults, in which the hackers constantly attempt to reveal the hidden message, as well as some inadvertent attacks, like noise addition. Here, GA operators are utilised to lessen distortion. The major goal of this work is to propose a good, effective way for hiding data from hackers and sending it to the target in a safer manner by maintaining randomization in message bit insertion into audio data.***
Keywords: RSA; MP3; GA Operators; LSB layers

I. INTRODUCTION

The Internet's current rapid advancement and the revolution in digital information have had a significant impact on culture as a whole. In today's communication systems, Data Hiding is crucial for issues with Network Security. Numerous audio applications, including voice search and assistance on the internet, voice-activated websites, and internet telephony, have also been aided by the internet. Another global interest is music, which can be heard by listeners through audio files or online radio broadcasts. (apart from conventional radio and television). The amount of audio traffic on the internet is rapidly rising. Thus, using audio as a cover material for data concealing is pretty obvious. WAV (Windows Audio Visual), AIFF (Audio Interchange File Format), log scale 8bit mlaw, and MP3 are common audio formats used online. (Motion Picture Experts Group Layer III). The supported data rates range from 8 kbps to 44.1 kbps. Techniques for masking audio rely on the limitations of the human auditory system. (HAS).

Today, it's common practise to cloak information in audio. Anytime you wish to hide data, you can utilise audio data hiding. There are several reasons to hide data, but the main one is to stop people from finding out that a message even exists. One of the simplest algorithms with a very fast data rate of additional information is data hiding in the least significant bits (LSBs) of audio samples in the temporal domain. We will discuss broad data hiding principles, fundamental terminology, an overview of applications, and methodologies in this project.

We will focus on data hiding inside audio signals, fundamental needs, and cutting-edge methods. We'll outline a cutting-edge method with numerous versions. Perfect transparency, resilience, high bit rate, minimal processing load, and, most importantly, great security are all characteristics of the suggested solution.

Cryptography is the study of information concealment. Information must always be protected when communicating over an unreliable medium like the internet, and cryptography is crucial to this process. Today, cryptography draws on concepts from a variety of academic fields, including computer science and mathematics. Steganography is the art of creating communications that are hidden from everyone but the sender and recipient. The communication does not draw unwanted attention because only the sender and the recipient are aware of its existence. When communicating across an unreliable channel, such as the internet, where information needs to be secured from other third parties, cryptography is utilised. Cryptography is the study of concealing information. Steganography is the art of creating communications that are hidden from everyone but the sender and recipient. The communication does not draw unwanted attention because only the sender and the recipient are aware of its existence. Steganography was utilised in the past, and these earlier techniques are known as Physical Steganography.

The following formula provides a very generic description of the pieces of the steganographic process:
Cover medium + hidden data + stego key = stego medium
To carry secret communications, a variety of cover objects (signals) might be utilised. Audio masking, a flaw in the human auditory system, is used to hide data in audio communications. Depending on the spectral and temporal properties of both the masked signal and the masker, a weaker signal may not be detectable in the presence of a louder signal (the masker). In-depth research has been done on masking models for perceptual compression of audio streams. In a data hiding application, the encoded signal is concealed there, whereas in perceptual compression, the quantization noise is hidden below the masking threshold.
Due to the human auditory system's vast dynamic range, data hiding in audio signals is particularly difficult. The human hearing system can distinguish sounds with powers greater than 1 billion to 1 and frequencies greater than 1000 to 1.
The parity coding method is one of the earlier works in audio data concealment. The parity coding technique divides a signal into regions of samples as opposed to individual samples, and each bit from the secret message is encoded in a sample region's parity bit. The method flips the LSB of one of the samples in the region if the parity bit of a chosen region does not match the secret bit to be encoded. As a result, the sender has more control over how to encode the secret bit, and the signal can be altered in a less obvious way. The process for parity coding is depicted in Figure 2.3.1.

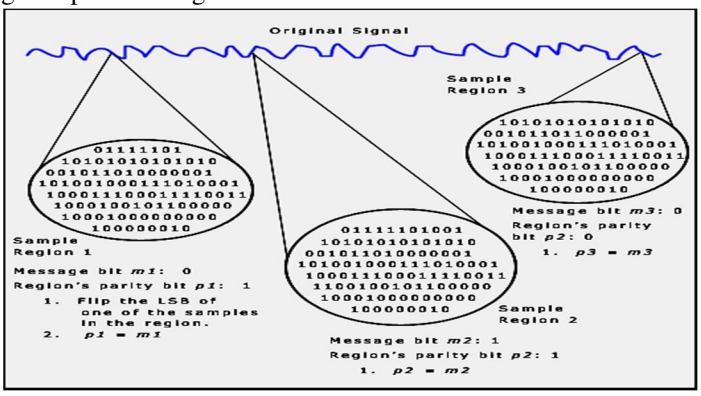


Figure2.1: Parity coding procedure

The phase of an initial audio segment is replaced with a reference phase that represents the data in the phase coding method. The relative phase between segments is maintained by adjusting the phase of succeeding segments. When it is possible, phase coding is one of the coding techniques with the highest signal-to-perceived noise ratios. There will be audible phase dispersion if the phase relationship between each frequency component is drastically altered. However, an inaudible coding can be achieved as long as the phase modification is sufficiently small (sufficiently small depends on the observer; experts in broadcast radio can discover adjustments that are unperceptible to an average observer). Phase coding is based on the idea that noise is more detectable to the human ear than the phase components of sound. The method achieves an inaudible encoding in terms of signal-to-perceived noise ratio by encoding the message bits as phase changes in the phase spectrum of a digital signal as opposed to adding disturbances.

$$phase\_new = \begin{cases} \pi/2 & if \quad message \quad bit = 0 \\ -\pi/2 & if \quad message \quad bit = 1 \end{cases}$$

The receiver needs to be aware of the segment length in order to decode the hidden message from the sound file. The receiver can then obtain the phases and extract the information using DFT. (consider Figure 2.3.2 for phase coding procedure).

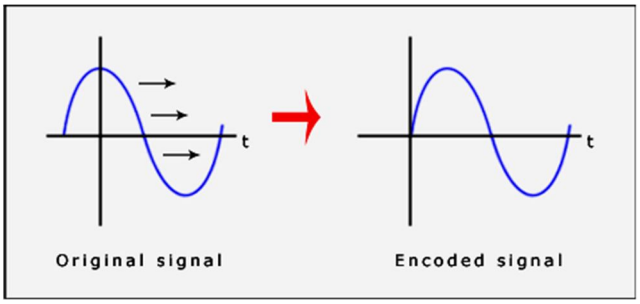


Figure 2.2: The signals before and after Phase coding procedure

II. REVIEW OF LITERATURE

Dutta, Hrishikesh et al. (2019) Steganography is used to conceal messages over digital media, such as speech, image, or video. Audio signals are used as covers due to their larger size and higher redundancy. This work reviews different techniques, algorithms and principles, and provides a compendium to digital audio steganography.

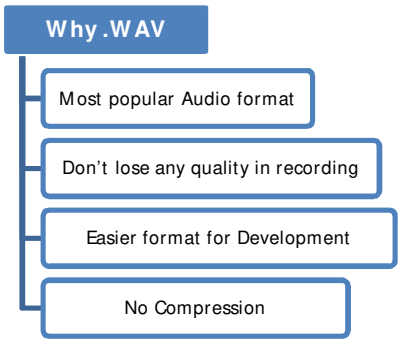
Nour Alaa (2021) Digital image watermarking is used for copyright protection and authentication, but this paper proposes an LSB-based digital watermarking scheme for fact-checking and fake news inquiry.

Gupta Sumeet, Dhanda Dr. Namrata (2019) Steganography is an art and science of communicating in a way that hides the existence of the communication, using a "cover" of a message to form a stego-signal. The main goal is to communicate securely in a completely undetectable manner and avoid suspicion. If a Steganography method causes someone to suspect there is secret information, then the method has failed.

Haleem, Alyaa & Mohammed, Nadia. (2019) PixInWav proposes a novel residual architecture for hiding images in audio, allowing independent encoding of the hidden image from the host audio without compromising quality.

Zhang et al. (2021) AEOLUS is a security overlay that proactively embeds a dynamic acoustic nonce at the time of user interaction and detects the presence of the embedded nonce in the recorded speech to ensure freshness. Experimental results show that AEOLUS yields 0.5% FRR at 0% FAR for speech re-use prevention upto a distance of 4 meters, and a user study with 120 participants shows that the acoustic nonce does not degrade overall user experience for 94.16% of speech samples.

III. SYSTEM DESIGN



A. Wave File Format Specifications

File Description: WAVE or RIFF WAVE sound file

File Extension: Commonly .wav, sometimes .wave

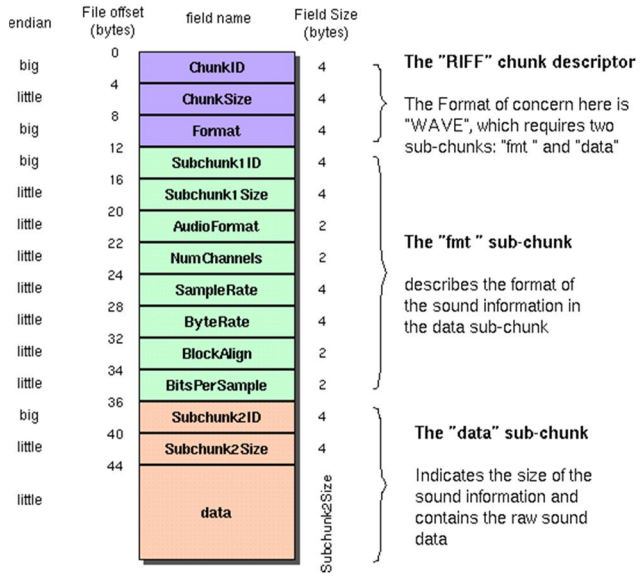


Figure 3.1: .WAV File Format

B. LSB Coding

An audio file with the ".wav" extension has been chosen as the host file for the current project. It is assumed that the least important parts of that file can be changed without the sound quality suffering.

To do it, one must first be familiar with the audio file's file structure. The header and the data are the two main components of all files, including WAV files. The header is found in the first 44 bytes of standard wav files. The file's remaining bytes, with the exception of the first 44, are entirely made up of data. The information is basically a single, enormous collection of audio samples. The header portion cannot be handled while embedding data. This is so that an audio file can become corrupted with only a small alteration to the header section.

The audio file can be read bit by bit by a program, which then stores the information in a new file. The first 44 bytes shouldn't be altered in any way because they include the header section's data.

Start by modifying the remaining data fields so that text can be embedded. For instance, to embed the word "Audio" inside an audio file, one must include the word's binary values in the audio data field.

C. Consider the Following Table

One can conclude from the table that in order to embed the term "Audio" into the host audio file, the appropriate eight bit binary values must really be embedded into the audio file's data field.

Each sample of the file has had numerous bits altered or updated to include text data in order to construct this technique. Additionally, it has been noted that the host audio file degrades after bits are changed. Different methods were used to modify the bits, such as changing 1, 2, 3, and 4 bits sequentially. But after undergoing all the modifications, it was found that a single bit adjustment in the LSB produced the best results.

TABLE I

Letters with ASCII Values and Corresponding Binary Values

Letter	ASCII Value	Corresponding Binary Value
A	065	01000001
U	117	01110101
D	100	01100100
I	105	01101001
O	111	01101111

D. Steps (For Embedding of Data)

Step 1. Don't change the audio file's header section.

Step 2. Begin with a suitable data byte place. (For the experiment purpose the present start byte was the 45th byte). Add the info that must be incorporated to the least important bit.

Step 3. To embed the entire message, modify the least important part in each alternative sample. The receiver's end data retrieval algorithm uses the same logic as the embedding algorithm.

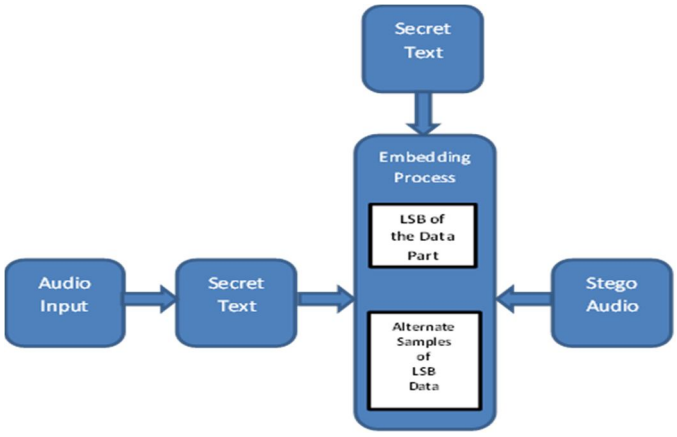


Figure 3.2: Block Diagram for Embedding

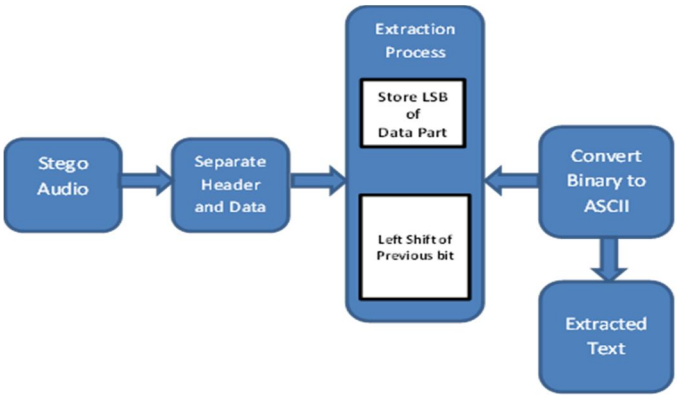


Figure 3.3: Block Diagram for Extracting

For this test, the audio file "audio.wav" has been chosen. The first 44 samples were left unchanged after each sample's binary values were checked. After the header section, the data embedding with LSB alteration has begun. If the 45th sample is used as the starting point for the data embedding procedure, the 45th sample's LSB value needs to be changed. If the associated sample's binary value is "01110100," "1" needs to be changed. Table I shows that the sender must embed the binary value "01000001" in order to embed the letter "A". Because of this, "A" should be embedded in accordance with Table II, according to the embedding method.

TABLE II

Samples of Audio File with Binary Values before and after Embedding

Sample No.	Binary values ofcorresponding sample	Binary valueto be embedded	Binary valuesafter modification
46	01110100	0	01110100
48	01011110	1	01011111
50	10001011	0	10001010
52	01111011	0	01111010
54	10100010	0	10100010
56	00110010	0	00110010
58	11101110	0	11101110
60	01011100	1	01011101

Take the 46th bit as a starting point, examine the least important bit, and then queue it up. Verify each alternative sample to get all of the messages. similar to 48th, 50th, 52nd, and so forth. Alternate samples' least significant bits are stored in the queue with the previous bit's value shifted to the left. To get the ASCII from which the text can be extracted, convert the binary numbers to decimal. The following table provides a more detailed representation of the entire retrieval procedure.

TABLE III

Extraction of Data from Audio File

Sample No	Binary values with embedded secret data	Bits that are stored in thequeue
46	01110100	0
48	01011111	01
50	10001010	010
52	01111010	0100
54	10100010	01000
56	00110010	010000
58	11101110	0100000
60	01011101	01000001

Because the letter "A"'s" embedding process was described in Table II, its retrieval process is shown in Table III. Every alternate sample has been examined since the 46th sample, and the least significant bit has been queued up with the prior bit's left shift. Once all the bits have been received, begin at the left side, take the first eight, convert them to decimal equivalents to obtain ASCII, and then pull the embedded text message from ASCII. The chart makes it apparent that after receiving the number 01000001 in the queue, it is translated into its equivalent decimal, which is 60, the ASCII character for "A." So, "A" is found.
The next letters were likewise located in the same manner, resulting in the whole word "Audio."

IV. IMPLEMENTATION

A. Algorithm for Encrypting

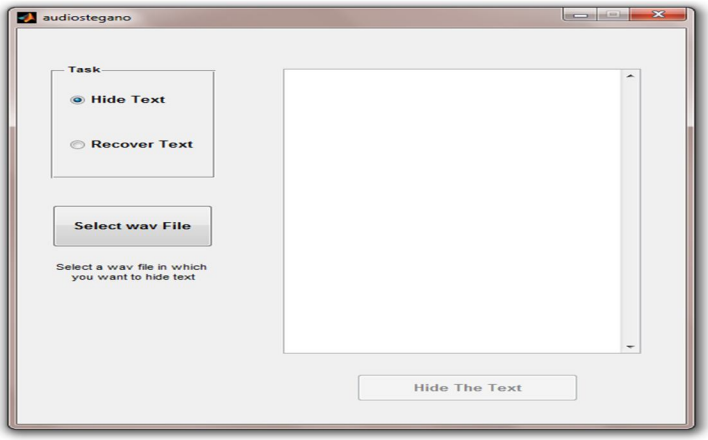
Step 1. Select a .wav file as a carrier to hide the text as a message.
Step 2. Check if the selected .wav file is >100KB.
Step 3. If the condition is satisfied then open the selected .wav file.
Step 4. Prepare message text as a binary column vector of 8.
Step 5. If text length is more than the no. of samples present in the selected .wav file then display “Message too big, select small message”
Step 6. Skip the first 44 byte of carrier which is address part of .wav file.
Step 7. Replace least significant bit of message carrier matrix with corresponding element of message vector by converting them from ASCII to decimal and then to equivalent binary.
Step 8. Create a new file to store the modified LSB.
Step 9. Get the stegano audio file as output.

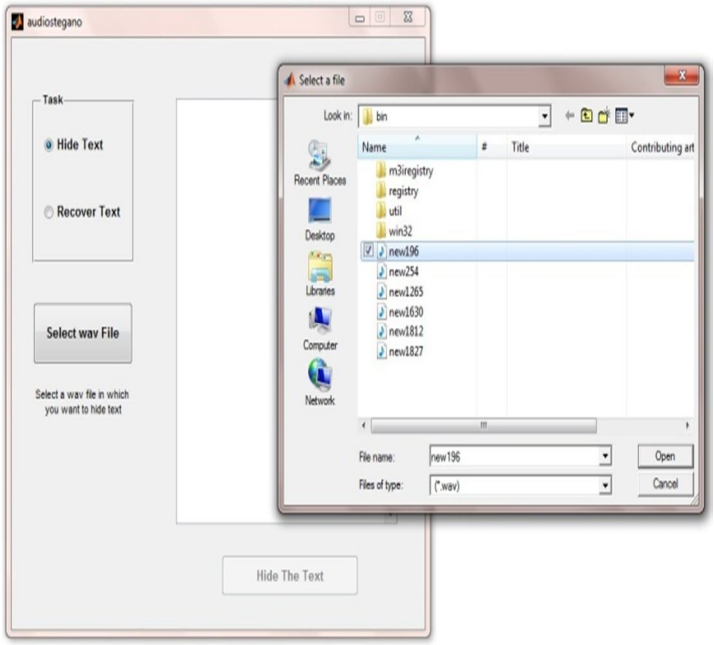
B. Algorithm for Decryption

Step 1. Define a register or variable to store the LSB. 2.Select a file containing the stegano audio.
Step 2. 3.Leave the first 44 bytes of .wav file untouched as it contains belongs to header. 4.Read LSB from next byte.
Step 3. Store the LSB in the variable defined earlier.
Step 4. Read the next LSB from selected .wav (stegano audio) file.
Step 5. Left shift the stored LSB in the variable and store the newly read LSB. 8.Check if all the LSBs of stegano audio are stored in the pre-defined variable. 9.Then convert the binary values stored in this variable to equivalent decimal. 10.Then after convert these decimal values to ASCII.
Step 6. 11.Check if all the values from the stegano audio file have been converted to ASCII. 12.Display the decrypted message on to the message box.

C. Development on MATLAB

Step:1
Guide: Graphical User Interface Tools.






```
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject                    handle to pushbutton1 (see GCBO)

% eventdata  reserved - to be defined in a future version of MATLAB
% handles                   structure with handles and user data (see GUIDATA)value1=get(handles.radiobutton1,'value');
value2=get(handles.radiobutton2,'value');
if value1==1
[handles.fname, handles.pname] = uigetfile('*.wav','Select a file');set(handles.pushbutton2,'enable','on');
guidata(hObject, handles); %update handle structure
end

if value2==1
set(handles.pushbutton2,'enable','off');
[filename, pathname] = uigetfile('*.wav','Select a file');[y,fs,nbits,opts]=wavread([pathname filename],[1 2]);
%open the file with hidden text fid1=fopen([pathname filename],'r');header=fread(fid1,40,'uint8=>char');
data_size=fread(fid1,1,'uint32');
%read the wave data samples [dta,count]=fread(fid1,inf,'uint16');
%close the file,only wav data samples are sufficient for extracting the textfclose(fid1);
```

D. Explanation of the above set of code

Step 1. The above set of code would be executed whenever the “Select a .wav file button” is pressed.

Step 2. Next when the “Hide Text button” is selected and the “Select a .wav file button” is pressed then thecode would guide us through path and then file name to select the .wav file.

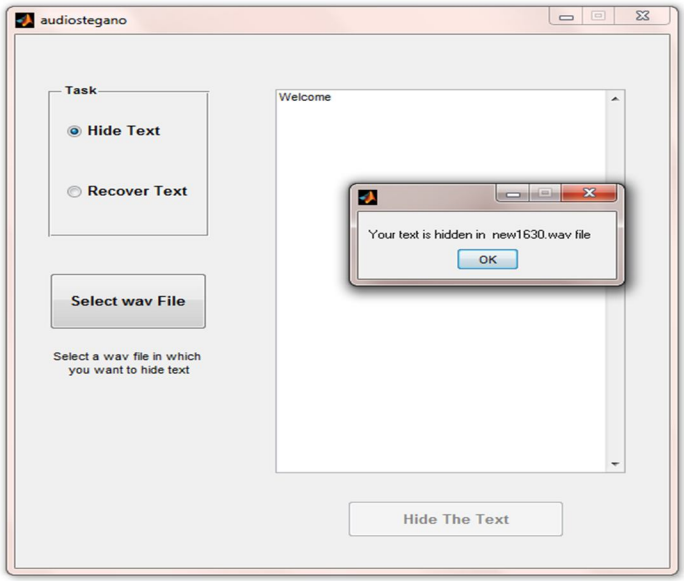
Step 3. As the above sets of steps are executed it would make the “Hide the Text button” go enable.

Step 4. Once the “Hide the Text button” is enabled it reads the selected .wav file and stores first 40 bytes inthe variable named “header”.

Step 5. The next 41st to 44th bytes are clubbed together to make a single element and is stored in the variablenamed “data_size”.

Step 6. Then close the file, only wav data samples are sufficient for extracting the text.

Step: 3
Encryption:



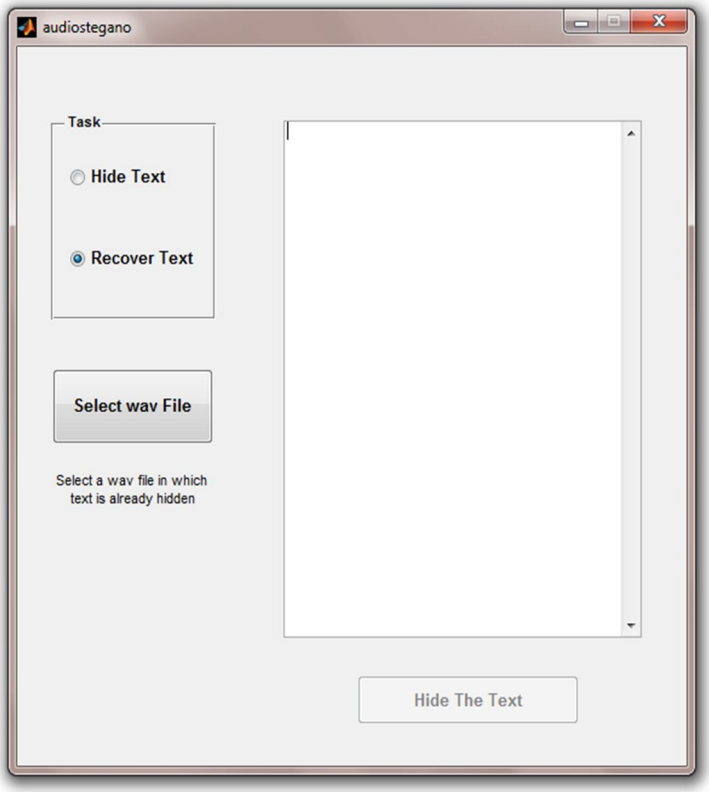
lsb=1;

```
msg=get(handles.edit1,'string'); %get text message from editboxmsg
[ro,co]=size(msg);ro
co
if ( (ro*co*8+28) > count )
msgbox('Message too big, select small message','Empty');else
[m_msg,n_msg]=size(msg);n_msg
m_msg msg_double=double(msg);
msg_double%convert it to double
msg_bin_re m_bin=de2bi(m_msg,10)';m_bin %
n_bin=de2bi(n_msg,10)';                    %n_bin
len=length(msg_bin_re);                    %length of message binarylen

len_bin=de2bi(len,20)';                    %convert the length to binarylen_bin

%hide identity in first 8 wav data samples. identity=[1 0 1 0 1 0 1 0]'; dta(1:8)=bitset(dta(1:8),lsb,identity(1:8));

%hide binary length of message from 9th to 28 th sampledta(9:18)=bitset(dta(9:18),lsb,m_bin(1:10));
dta(19:28)=bitset(dta(19:28),lsb,n_bin(1:10));
%hide the message binary starting from 29th position of wave data samples
dta(29:28+len)=bitset(dta(29:28+len),lsb,msg_bin(1:len)');

randname=num2str(randint(1,1,[1 2000]));

%open a new wav file in write mode fid2=fopen(['new' randname '.wav'],'w');

%copy the header of original wave filefwrite(fid2,header,'uint8'); fwrite(fid2,data_size,'uint32');
%copy the wav data samples with hidden text in new filefwrite(fid2,dta,'uint16');
fclose(fid2);

msgbox(['Your text is hidden in new' randname '.wav file'],'');set(hObject,'enable','off');
```

E. Explanation of the above set of code

Step 1. Enter text in the edit box.
Step 2. Get text message from edit box.
Step 3. Then convert message to binary.
Step 4. Reshape the message binary in a column.
Step 5. Calculate length of message binary.
Step 6. Convert the length to binary.
Step 7. Hide identity in first 8 .wav data samples.
Step 8. Hide binary length of message from 9th to 28th sample
Step 9. Hide the message binary starting from 29th position of wave data samples.
Step 10. Open a new .wav file in write mode
Step 11. Copy the header of original wave file
Step 12. Copy the wav data samples with hidden text in new file
Step 13. Your text is hidden in new ' randname '.wav file

Step: 5
Initialize Recover Process



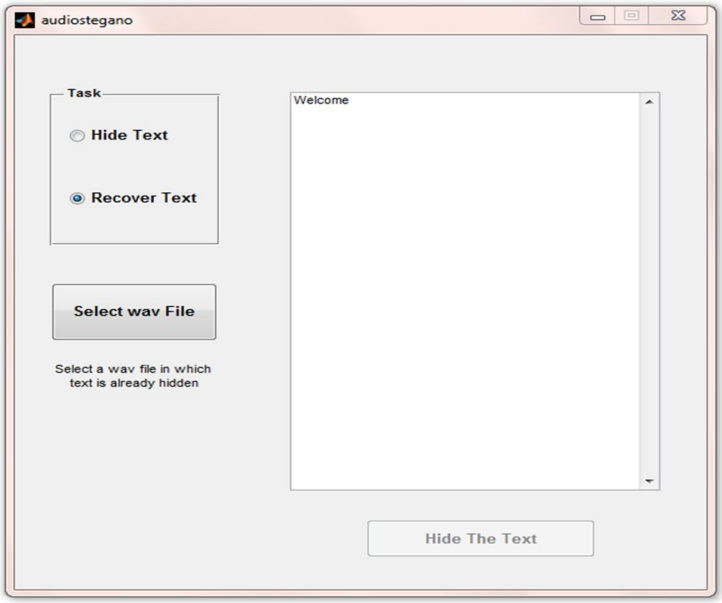

```
% --- Executes on button press in radiobutton2.
function radiobutton2_Callback(hObject, eventdata, handles)
% hObject handle to radiobutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
% Hint: get(hObject,'Value') returns toggle state of radiobutton2set(hObject,'value',1);
set(handles.text2,'string','Select a wav file in which text is already hidden');set(handles.pushbutton2,'enable','off');
```

F. Explanation of the above set of code

"Recover Text button" when enabled select a.wav file in which text is already hidden

'Step: 6

Decryption Process




```
lsb=1;

identity=bitget(dta(1:8),lsb)';if identity==[1 0 1 0 1 0 1 0]
%extract the length of text from first 9th to 28th wav data sampleslen_bin=zeros(20,1);
m_bin=zeros(10,1);n_bin=zeros(10,1);

m_bin(1:10)=bitget(dta(9:18),lsb); n_bin(1:10)=bitget(dta(19:28),lsb);
%convert the length to decimal
%len=bi2de((len_bin)');m=bi2de(m_bin'); n=bi2de(n_bin'); len=m*n*8;

secmsg_bin=zeros(len,1);
%extract the lsb from wave data sample secmsg_bin(1:len)=bitget(dta(29:28+len),lsb);
secmsg_bin_re=reshape(secmsg_bin,len/8,8); secmsg_double=bi2de(secmsg_bin_re); %convert it to binary
secmsg=char(reshape(secmsg_double,m,n));   %convert to char(ASCII)
%size(secmsg)
%secmsg=reshape(secmsg,m,n/8); set(handles.edit1,'string',secmsg);

else
msgbox('File has no hidden text','Empty');end
```

G. Explanation of the above set of code

Step 1. Extract the length of text from first 9th to 28th wav data samples
Step 2. Convert the length to decimal
Step 3. Extract the LSB from wave data sample
Step 4. Convert it to binary
Step 5. Convert to character (ASCII)
Step 6. If message is empty then display File has no hidden text

H. Advantages

Audio based Steganography has the potential to conceal more information:

- Audio files are generally larger than images
- Our hearing can be easily fooled
- Slight changes in amplitude can store vast amounts of information

The flexibility of audio Steganography is makes it very potentially powerful :

- The methods discussed provide users with a large amount of choice and makes the technology more accessible to everyone. A party that wishes to communicate can rank the importance of factors such as data transmission rate, bandwidth, robustness, and noise audibility and then select the method that best fits their specifications.
- For example, two individuals who just want to send the occasional secret message back and forth might use the LSB coding method that is easily implemented. On the other hand, a large corporation wishing to protect its intellectual property from "digital pirates" may consider a more sophisticated method such as phase coding, SS, or echo hiding.

Another aspect of audio Steganography that makes it so attractive is its ability to combine with existing cryptography technologies.

- Users no longer have to rely on one method alone. Not only can information be encrypted, it can be hidden altogether.

Many sources and types makes statistical analysis more difficult :

- Greater amounts of information can be embedded without audible degradation

I. Disadvantages

1) 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.

2) Robustness: Copyright marks hidden in audio samples using substitution could be easily manipulated or destroyed if a miscreant comes to know that information is hidden this way.

3) Commercialized audio Steganography have disadvantages that the existence of hidden messages can be easily recognized visually and only certain sized data can be hidden.

4) Compressing an audio file with lossy compression will result in loss of the hidden message as it will change the whole structure of a file. Also, several lossy compression schemes use the limits of the human ear to their advantage by removing all frequencies that cannot be heard. This will also remove any frequencies that are used by a Steganography system which hides information in that part of the spectrum.

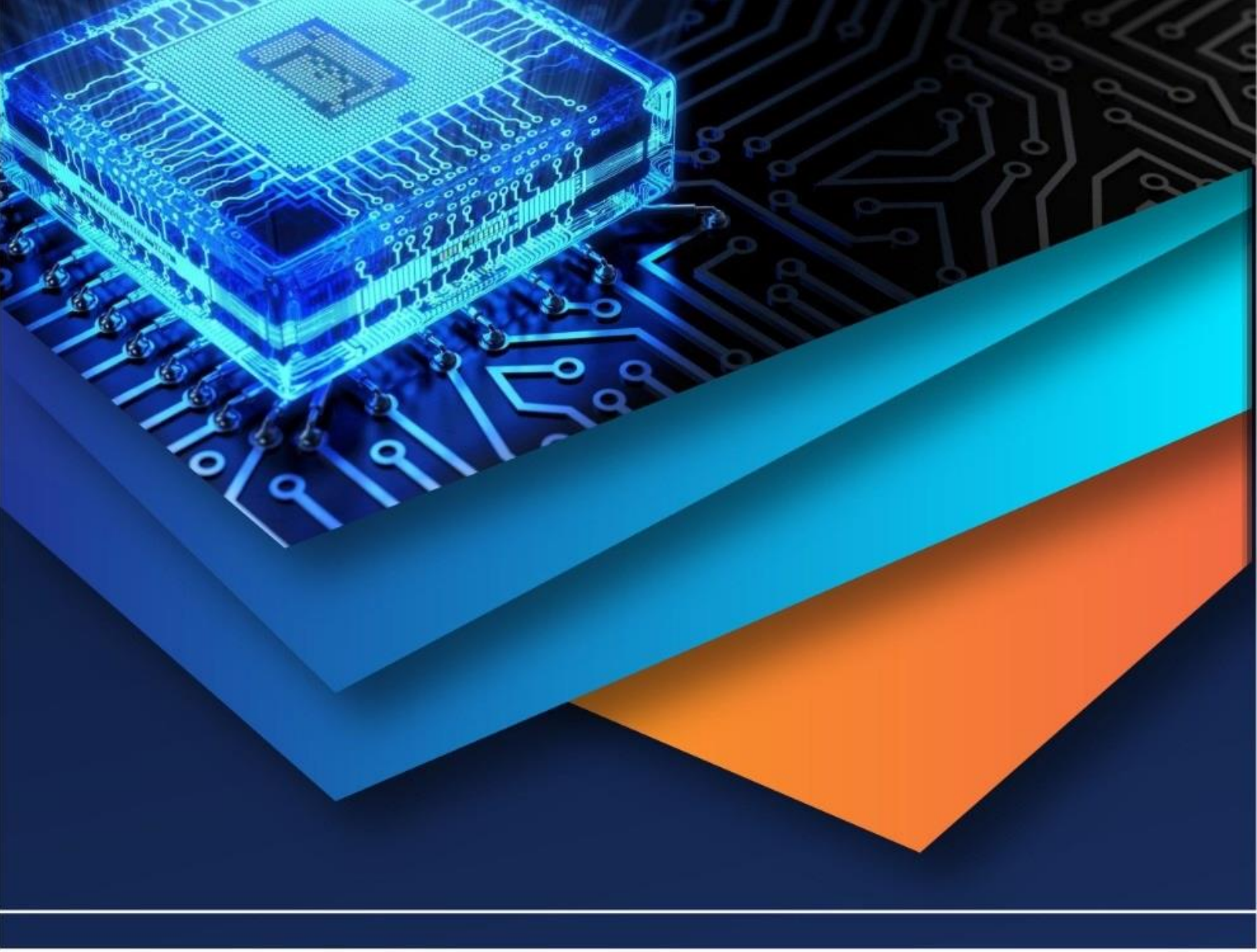
V. CONCLUSION & FUTURE SCOPE

Phase coding, spread spectrum, and echo hiding are a few of the common methods for data concealing in audio files that are examined in this research. Additionally, it assesses the viability of audio steganography, which presents promising opportunities for commercial data concealment. Steganographic elements could be incorporated into camera firmware by digital camera manufacturers to annotate images with the photographer's copyright information. Steganography can help with copyright marking, which can help to mitigate piracy. The goal of current steganography research is to develop other platforms via which information can be concealed. One intriguing concept is to use a distinct steganographic channel in a network to deliver messages. However, it can be challenging to identify the one platform that is optimal for sending covert communications.

REFERENCES

[1] LOU D.C., LIU J.L., LI C.-T.: 'Digital SignatureBased Image Authentication', in LU C.S. (EDS.): 'Multimedia security: steganography and digital Watermarking techniques for protection of intellectual property' (Idea Group Inc., 2003).
[2] SEITZ J.: 'Digital watermarking for digital media' (Idea Group Publishing, 2005), Ch. 2.
[3] SCHNEIDER M., CHANG S.-F.: 'A content based digital signature for image authentication'. Proc. IEEE Int. Conf.Image Processing (ICIP'96), 1996, pp. 227–230.
[4] FRIDRISH J., BALDOZA A.C., SIMARED R.J.: 'Robust digital watermarking based on key dependent basis functions'. Proc. Int. Conf. LNCS:IH, Portland, OR, USA, April 1998, vol. 1525, pp. 143–157.
[5] LU C.S.: 'On the security of structural information extraction/embedding for image authentication'. Proc. IEEE ISCAS'04, 2004, pp. 169–172.
[6] ANTHONY T., HO S., YONG L.G.: 'Image content authentication using pinned sine transform', EURASIP J. Appl. Signal Process., 2004, 14, pp. 2174–2184.
[7] SUN Q., HE D., YE S.: 'Feature selection for semi fragile signature based authentication systems'. Proc. IEEE Workshop on Image Signal Processing, 2003, pp. 99–103.
[8] SUN Q., YE S., LIN C.-Y.: 'A crypto signature scheme for image authentication over wireless channel', Int. J. Image Graph., 2005, 5, (1), pp. 1–14.

[9] PETER M., UHL M.: 'Watermark security via wavelet filter parametrization'. Proc. Int. Conf. ICASSP, USA, 2000.
[10] Dutta, Hrishikesh & Das, Rohan & Nandi, Sukumar & Prasanna, S.. (2019). An Overview of Digital Audio Steganography. IETE Technical Review. 37. 1-19. 10.1080/02564602.2019.1699454.
[11] Nour Alaa (2021) "Viral 'fake news' lists and the limitations of labeling and fact-checking" Fake News [Preprint]. Available at: https://doi.org/10.7551/mitpress/11807.003.0039.
[12] Gupta Sumeet, Dhanda Dr. Namrata (2019). Audio Steganography Using Discrete Wavelet Transformation (DWT) & Discrete Cosine Transformation (DCT). IOSR Journal of Computer Engineering (IOSR-JCE), Volume 17, Issue 2, PP 32-44 Haleem, Alyaa & Mohammed, Nadia. (2019). Digital Information Hiding Techniques . 10.33899/csmj.2010.163916.
[13] Zhang, Yangyong & Shirvanian, Maliheh & Arora, Sunpreet & Huang, Jianwei & Gu, Guofei. (2021). Practical Speech Re-use Prevention in Voice-driven Services.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY