AUDIO STEGANOGRAPHY SYSTEM FOR MUSIC COPYRIGHT PROTECTION

By

RAMANAN B R	23BLC1199
SUBRAMANIAM B B	23BLC1175
DHARHSHINI V J	23BLC1080
LAYASHREE L	23BLC1245

A project report submitted to

Dr. A. ANNIS FATHIMA

SCHOOL OF ELECTRONICS ENGINEERING

In partial fulfilment of the requirements for the course of

BECM301L – SIGNAL PROCESSING

In

B.Tech. ELECTRONICS AND COMPUTER

ENGINEERING

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

Vellore Institute of Technology, Chennai

Vandalur-Kelambakkam Road, Chennai – 600127

NOVEMBER 2024

BONAFIDE CERTIFICATE

Certified that this project report entitled “**Audio Steganography System for Music Copyright Protection”** is a bonafide work of **Ramanan B R (23BLC1199), Subramaniam B B (23BLC1175), Dharhshini V J (23BLC1080), Layashree L (23BLC1245)** who carried out the Project work under my supervision and guidance.

Dr .ANNIS FATHIMA

Associate Professor

School of Electronics Engineering (SENSE),

VIT University, Chennai

Chennai-600127.

ABSTRACT

This report presents the design and implementation of an audio steganography system aimed at enhancing music copyright protection. The system allows for embedding a secret text message within an audio signal, leveraging four distinct steganography techniques: Least Significant Bit (LSB), Direct Sequence Spread Spectrum (DSSS), Phase Spectrum, and Echo Hiding. Each method is analysed in terms of embedding capacity, perceptual transparency, robustness against attacks, and decoding accuracy. A comparative study of the results provides insight into the advantages and limitations of each technique. The report concludes with a recommendation on the most effective method for secure and efficient audio-based copyright protection, based on experimental findings.

ACKNOWLEDGEMENT

We wish to express our sincere thanks and deep sense of gratitude to our project guide, **Dr. Annis Fathima,** Professor, School of Electronics Engineering, for her consistent encouragement and valuable guidance offered to us in a pleasant manner throughout the course of the project work.

We are extremely grateful to **Dr. Susan Elias,** Dean of the School of Electronics Engineering, VIT Chennai, for extending the facilities of the School towards our project and for her unstinting support.

We express our thanks to our Programme Chair **Dr. Mohanaprasad K** for their support throughout the course of this project.

We also take this opportunity to thank all the faculty of the School for their support and their wisdom imparted to us throughout the course.

We thank our parents, family, and friends for bearing with us throughout the course of our project and for the opportunity they provided us in undergoing this course in such a prestigious institution.

TABLE OF CONTENTS

1. INTRODUCTION

1.1 OBJECTIVES and GOALS

Design and Implement an Audio Steganography System:

To create a system that allows embedding secret text messages into audio signals, providing a means of secure communication and copyright protection.

Explore Four Steganographic Techniques:

To implement and analyze four distinct methods - Least Significant Bit (LSB), Direct Sequence Spread Spectrum (DSSS), Phase Spectrum, and Echo Hiding - in terms of their ability to conceal information within audio files.

Evaluate Performance:

To evaluate each method based on critical parameters such as Embedding Capacity, Perceptual Transparency (imperceptibility), Robustness against Attacks, and Decoding Accuracy.

Conduct a Comparative Analysis:

To compare the strengths and weaknesses of each technique through objective metrics like MSE (Mean Squared Error), PSNR (Peak Signal-to-Noise Ratio), SNR (Signal-to-Noise Ratio), and Embedding Capacity.

Goals:

Imperceptibility and Quality Preservation: Ensure that the embedded message does not significantly affect the quality of the audio, maintaining its original perceptual characteristics.

Maximize Embedding Capacity: Maximize the amount of data that can be embedded in the audio without affecting its quality or robustness.

Robustness to Attacks: Ensure the system can withstand common attacks such as compression, noise addition, or filtering, preserving the embedded message's integrity.

Decoding Accuracy: Achieve high accuracy in the extraction of the hidden message from the audio signal, ensuring reliable recovery without errors.

Recommend the Best Method: Based on the comparative analysis, provide a recommendation on the most effective steganography technique for audio-based copyright protection, balancing security, capacity, and audio quality.

2. THEORY

Audio steganography embeds secret information within an audio signal while preserving its perceptual quality. The process involves preprocessing the message and audio for compatibility, followed by embedding the data using methods like LSB, echo hiding, or DSSS. The stego-audio is then transmitted or stored, ensuring robustness against distortions. Finally, the extraction phase retrieves the hidden message securely using the reverse embedding process.

Parameters used to measure audio quality:

MSE (Mean Squared Error)

$MSE = (1/N) * \Sigma (x_i - y_i)^2$

Measures the distortion between original and modified audio. Lower MSE indicates less distortion.

PSNR (Peak Signal-to-Noise Ratio)

$PSNR = 10 * \log_{10}(MAX^2 / MSE)$

Assesses the quality of the modified audio. Higher PSNR means better fidelity.

SNR (Signal-to-Noise Ratio)

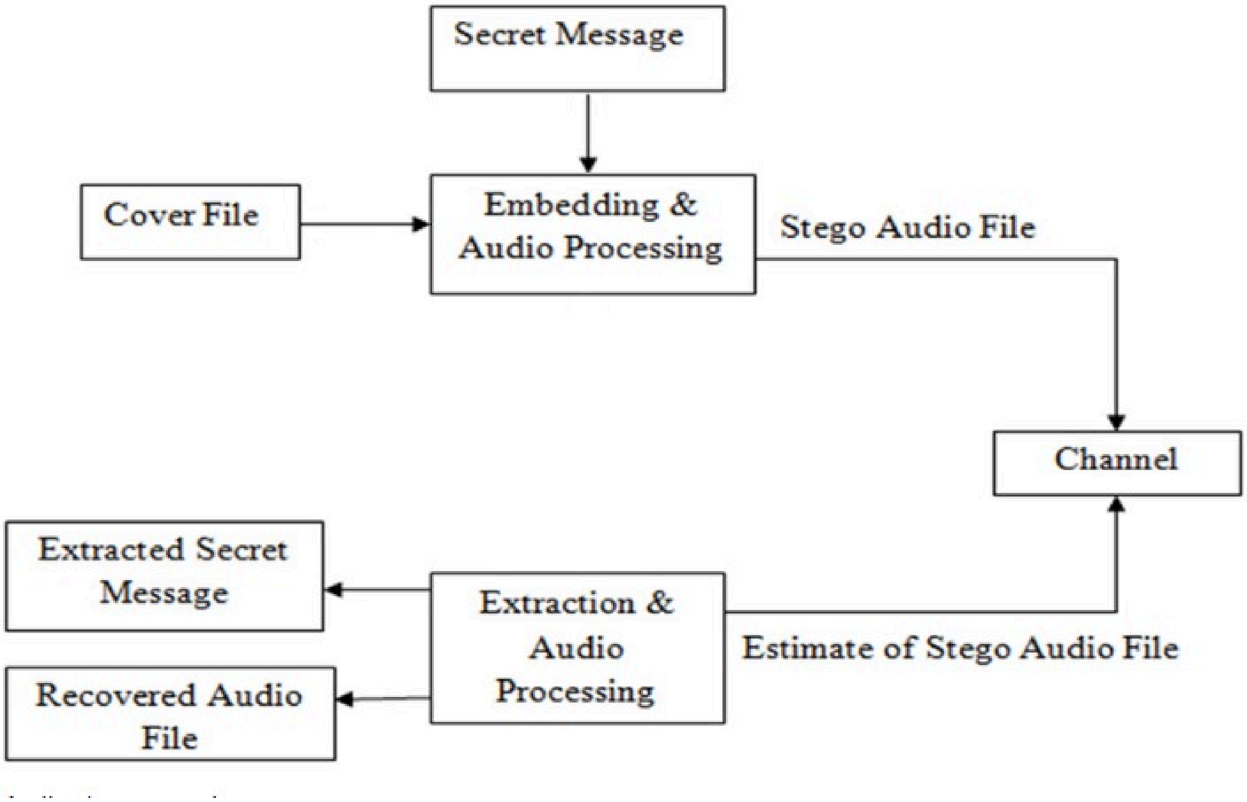
$SNR = 10 * \log_{10}(Signal\ Power / Noise\ Power)$

Evaluates robustness against noise. Higher SNR ensures stronger resistance to interference.

Purpose

These metrics quantify distortion, fidelity, and robustness, helping compare methods like LSB, Echo Hiding, Phase Encoding, and DSSS for audio data embedding.

Block Diagram



Algorithm:
Software Used: MATLAB

1. DSSS (Direct Sequence Spread Spectrum) Encoding Algorithm
Encoding (Transmitter Side)

1. Input:
   Secret message (e.g., text or binary data).
   Audio file (e.g., WAV file) containing the host signal.
2. Convert the Message:
   - Convert the secret message into binary form (e.g., ASCII or binary string).

3. Generate Spread Sequence:
   - Create a pseudo-random noise (PN) sequence that will be used to spread the message.
4. Modulate the Message:

- XOR each bit of the secret message with the corresponding bit in the PN sequence to create the spread message.

5. Embed the Message in Audio:
   - Modify the audio samples to encode the spread message.
   - Embed the encoded message into the host audio signal by adjusting its amplitude based on the modulated data.
6. Save the Stego Audio:
   - Save the modified audio as a new audio file (stego audio).

Decoding (Receiver Side)

1. Input:
   - Stego audio file (audio containing the embedded message).
2. Extract Audio Data:
   - Extract the audio samples from the stego audio.
3. Generate the PN Sequence:
   - Recreate the same PN sequence that was used during encoding.
4. Demodulate the Message:
   - Perform an XOR operation between the PN sequence and the extracted audio samples to retrieve the original message.
5. Convert the Message:
   - Convert the binary data back to the original message format (text, for instance).

2. Echo Encoding Algorithm

Encoding (Transmitter Side)

1. Input:
   - Secret message (text or binary).
   - Audio file (host signal).
2. Convert the Message:
   - Convert the secret message into binary form.
3. Generate Echo Pattern:
   - For each bit of the secret message, decide on the time delay and amplitude to add to the audio signal to encode each bit.
   - A bit of 0 might result in a short delay, while 1 results in a longer delay.
4. Embed Echoes in Audio:
   - Modify the host audio by inserting short or long echoes at certain points in the audio based on the binary message.
5. Save the Stego Audio:
   - Save the modified audio as a new audio file with echoes embedded.

Decoding (Receiver Side)

1. Input:
   - Stego audio file (audio with embedded echoes).
2. Extract Audio Data:
   - Extract the audio samples from the stego audio.
3. Identify Echoes:
   - Analyze the audio to identify where echoes are present. The delay between consecutive audio samples indicates the binary message.
4. Retrieve the Message:
   - Determine the binary message based on the timing of the echoes (e.g., short delays for 0 and long delays for 1).
5. Convert the Message:
   - Convert the binary data back to the original message format.

3. Phase Encoding Algorithm

Encoding (Transmitter Side)

1. Input:
   - Secret message (binary or text).
   - Audio file (host signal).
2. Convert the Message:
   - Convert the secret message into binary form.
3. Generate Phase Shift Pattern:
   - For each bit of the message, generate a phase shift (e.g., 0 might correspond to a phase shift of 0°, while 1 corresponds to 180°).
4. Embed the Phase Shift in Audio:
   - Alter the phase of the host audio signal based on the binary message.
   - Apply the phase shifts at specific points in the audio.
5. Save the Stego Audio:
   - Save the modified audio with phase-shifted samples as the stego audio.

Decoding (Receiver Side)

1. Input:
   - Stego audio file (audio with embedded phase shifts).
2. Extract Audio Data:
   - Extract the audio samples from the stego audio.
3. Analyze Phase Shifts:

- Analyze the phase shifts in the audio to detect where changes occurred.

4. Retrieve the Message:
   - Determine the binary message by comparing the phase shifts (e.g., 0° for 0, 180° for 1).
5. Convert the Message:
   - Convert the binary data back to the original message format.

4. LSB (Least Significant Bit) Encoding Algorithm

Encoding (Transmitter Side)

1. Input:
   - Secret message (binary or text).
   - Audio file (host signal).
2. Convert the Message:
   - Convert the secret message into binary form.
3. Embed Message in Audio:
   - Replace the least significant bits of the audio samples with the bits of the secret message.
   - Ensure that the message is embedded in such a way that the original audio remains relatively unchanged.
4. Save the Stego Audio:
   - Save the modified audio with the message embedded as the stego audio.

Decoding (Receiver Side)

1. Input:
   - Stego audio file (audio with embedded message in the LSB).
2. Extract Audio Data:
   - Extract the audio samples from the stego audio.
3. Extract the Message:
   - Retrieve the least significant bits from the audio samples to recover the binary message.
4. Convert the Message:
   - Convert the binary data back to the original message format (text).

GUI Algorithm

Initialization Phase (Setup)

1. Create GUI Components:
   - Create the main window for the GUI application.
   - Add labels, buttons, text areas for inputs, and actions.
   - Add a dropdown menu to select encoding technique (DSSS, Echo, Phase, LSB).

- Add an area to display spectrograms and other audio-related information.

Transmitter Side (Encoding)

2. Select Message:
   - Let the user input a custom message or generate a random one using a button.
3. Select Audio File:
   - Let the user select an audio file (host signal).
4. Select Encoding Technique:
   - Allow the user to choose one of the four encoding techniques from the dropdown: DSSS, Echo, Phase, LSB.
5. Encode Message:
   - Based on the selected encoding technique:
     - Use the corresponding encoding algorithm (DSSS, Echo, Phase, or LSB).
   - Display results such as encoding status, metrics (MSE, PSNR, etc.), and capacity.
6. Save Stego Audio:
   - Allow the user to save the encoded (stego) audio file.
7. Display Visual Feedback:
   - Show the spectrogram of the original and stego audio files.

Receiver Side (Decoding)

8. Select Stego Audio File:
   - Allow the user to select a stego audio file.
9. Select Decoding Technique:
   - Based on the encoding technique chosen earlier, decode the message from the stego audio using the appropriate algorithm.
10. Display Decoded Message:
    - Show the decoded message in the text area.
11. Save Decoded Message:
    - Allow the user to save the decoded message into a text file.

Error Handling and User Feedback

12. Handle Errors:
    - Display alerts for missing inputs or encoding/decoding failures.
13. Play Audio:
    - Provide buttons to play both the original and stego audio files for comparison.

Each steganographic technique involves different signal processing steps, which are integrated into a cohesive GUI where the user can select techniques, input messages, view results, and save both the encoded audio and decoded message.

3. RESULT AND ANALYSIS

Dataset:

Our dataset consists of a collection of audio files sourced from the *No Copyright Sound* YouTube channel (link in references). These audio tracks have been carefully modified by adjusting their frequencies and durations. The durations of the audio files vary, ranging from as short as 5 seconds to as long as 1 minute and 30 seconds. This diverse set of audio clips provides a versatile foundation for experimentation and analysis.

Tools used: Matlab Simulink

Result and analysis:

The secret message was successfully encoded into the audio, and the decoding process was carried out to retrieve the hidden information. Error metrics such as Mean Squared Error (MSE), Peak Signal-to-Noise Ratio (PSNR), and Signal-to-Noise Ratio (SNR) were calculated to analyze the impact of the embedding on the audio quality. The analysis showed that the embedding caused minimal distortion, with a low MSE and high PSNR, indicating that the audio quality remained largely intact. A comparison of the results highlighted the trade-offs between data capacity, robustness, and audio fidelity across different methods.

Comparison/analysis:

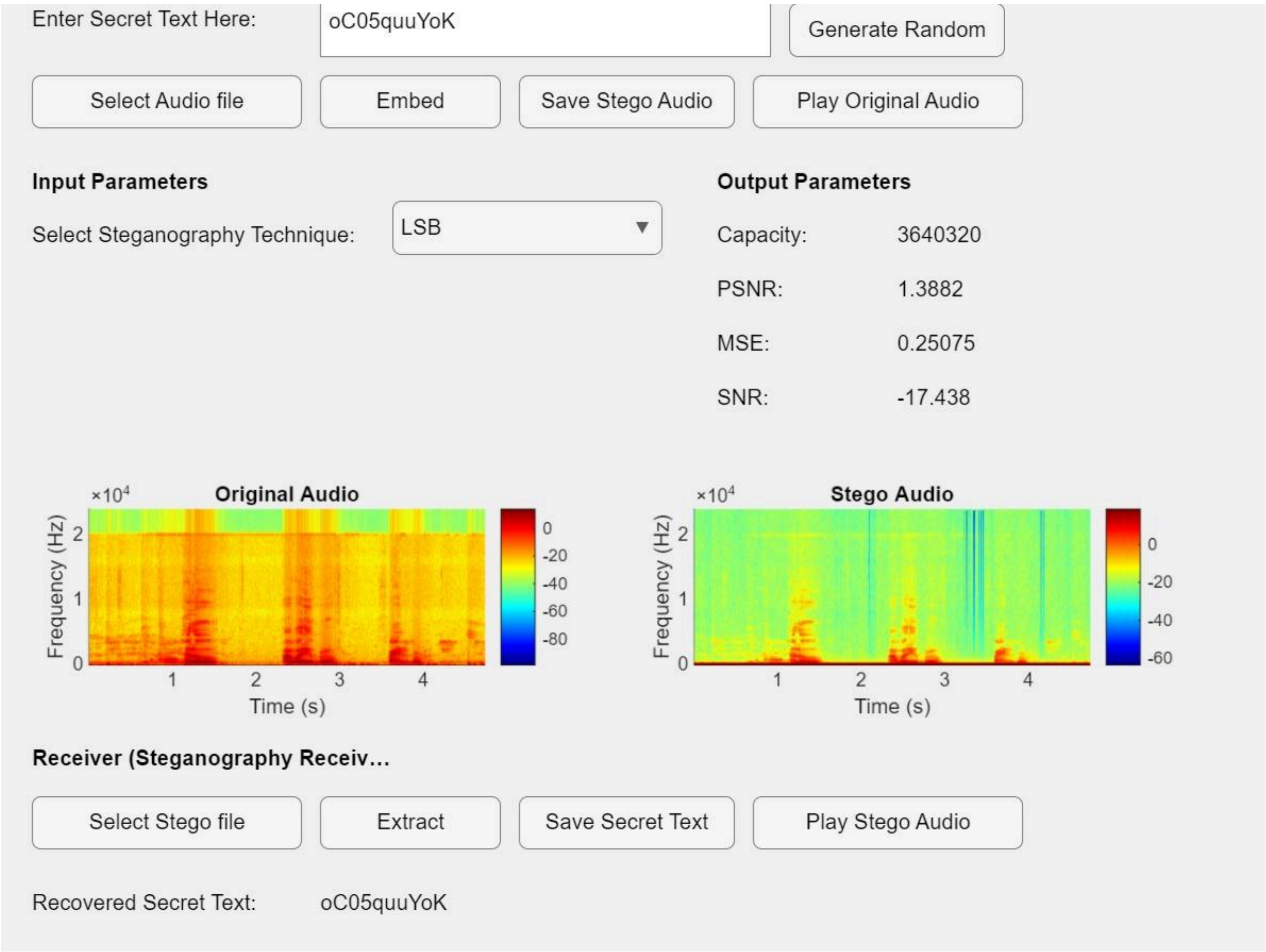
1. Least Significant Bit (LSB) Method

Observation/Result:

- High MSE (0.25075) and low PSNR (1.3882): Indicate significant perceptual changes to the original audio, leading to noticeable distortion.
- Moderate capacity (3,640,320): Allows for moderate data embedding, but it sacrifices audio quality.
- Low SNR (-17.438): Indicates high susceptibility to noise, especially at higher embedding capacities.

Analysis:

The LSB method is not ideal for high-fidelity applications due to noticeable distortion. It may only be suitable for applications where audio quality is not a priority and where simple data hiding is required, as it provides high capacity at the expense of audio quality.



Enter Secret Text Here: oC05quuYoK

Input Parameters

Select Steganography Technique: LSB

Output Parameters

Capacity: 3640320

PSNR: 1.3882

MSE: 0.25075

SNR: -17.438

Receiver (Steganography Receiv...

Recovered Secret Text: oC05quuYoK

2. Echo Hiding

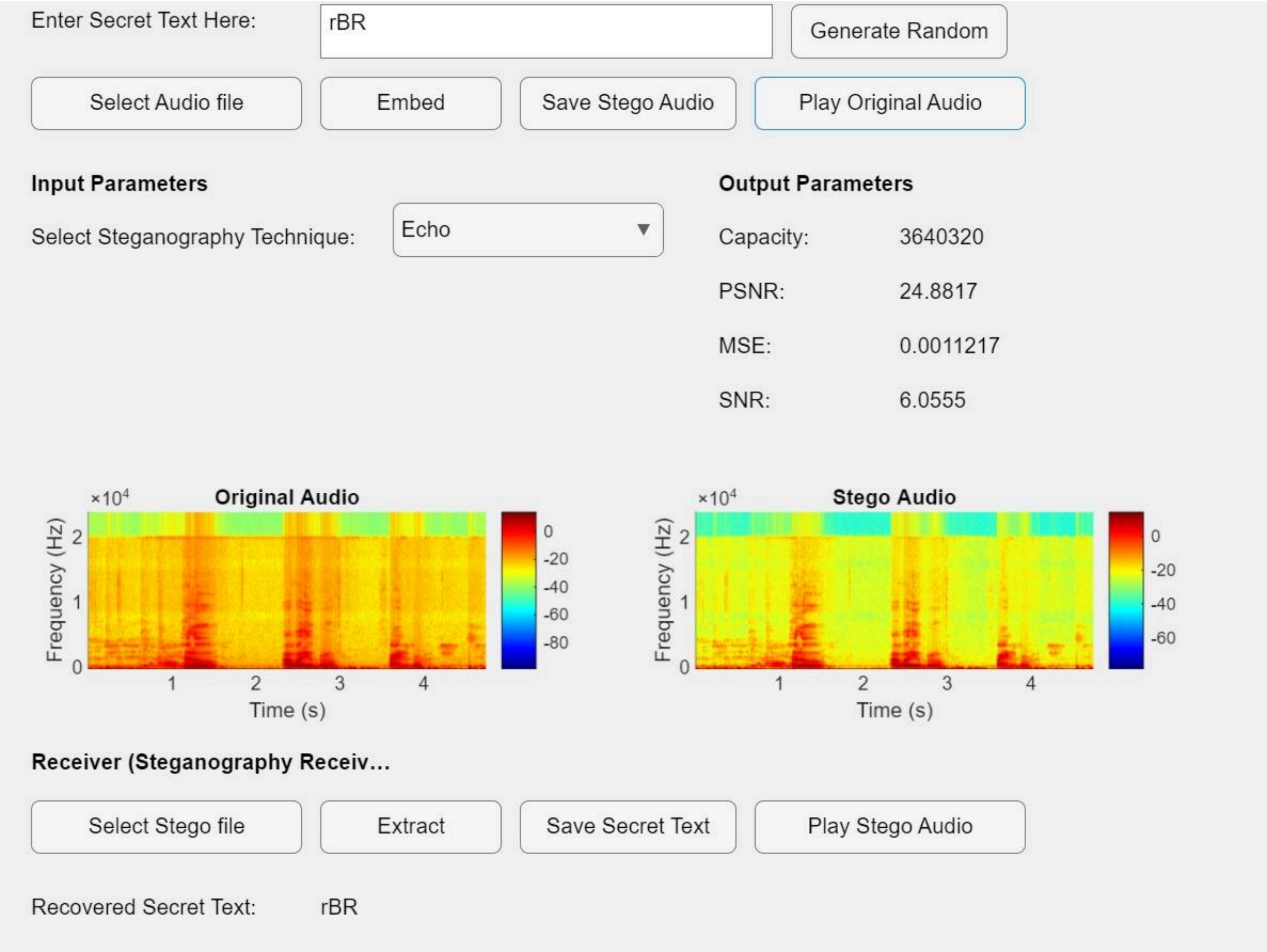
Observation/Result:

- Moderate MSE (0.001217) and moderate PSNR (24.8817): Indicate moderate perceptual changes to the audio, resulting in some distortion but retaining reasonable audio quality.
- Higher capacity (3,640,320): Supports embedding of a large amount of data.

- Moderate SNR (6.0555): Indicates some robustness against noise, though not as high as other techniques.

Analysis:

- Echo hiding is suitable for applications that require high capacity and some level of robustness but are less sensitive to minor audio quality reductions. It could work well for digital watermarking where data embedding is prioritized over audio fidelity.



Enter Secret Text Here: rBR

Input Parameters

Select Steganography Technique: Echo

Output Parameters

Capacity:	3640320
PSNR:	24.8817
MSE:	0.0011217
SNR:	6.0555

Receiver (Steganography Receiv...

Recovered Secret Text: rBR

3. Phase Encoding

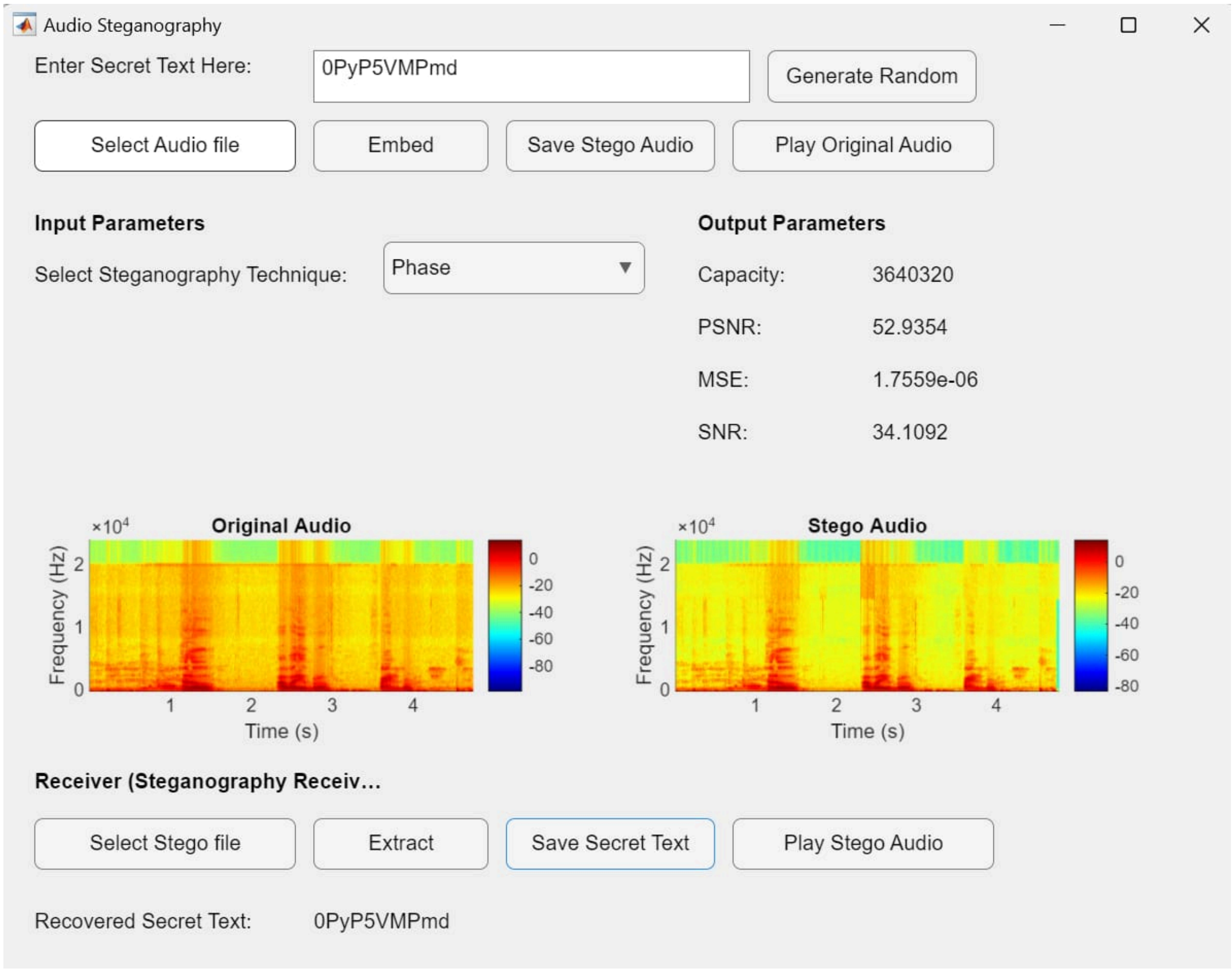
Observation/Result:

- Very low MSE (1.7559e-06) and very high PSNR (52.9354): Ensure minimal perceptual distortion, maintaining excellent audio quality.

- Moderate capacity (3,640,320): Balances data embedding needs with preservation of audio integrity.
- High SNR (34.1092): Provides strong noise resistance, ensuring high signal integrity.

Analysis:

- The Phase encoding technique is ideal for high-fidelity applications where audio quality is critical, such as watermarking in security systems. It allows data embedding with minimal distortion and excellent robustness, making it suitable for scenarios requiring minimal perceptual changes.



Enter Secret Text Here: 0PyP5VMPmd

Input Parameters

Select Steganography Technique: Phase

Output Parameters

Capacity: 3640320

PSNR: 52.9354

MSE: 1.7559e-06

SNR: 34.1092

Receiver (Steganography Receiv...

Recovered Secret Text: 0PyP5VMPmd

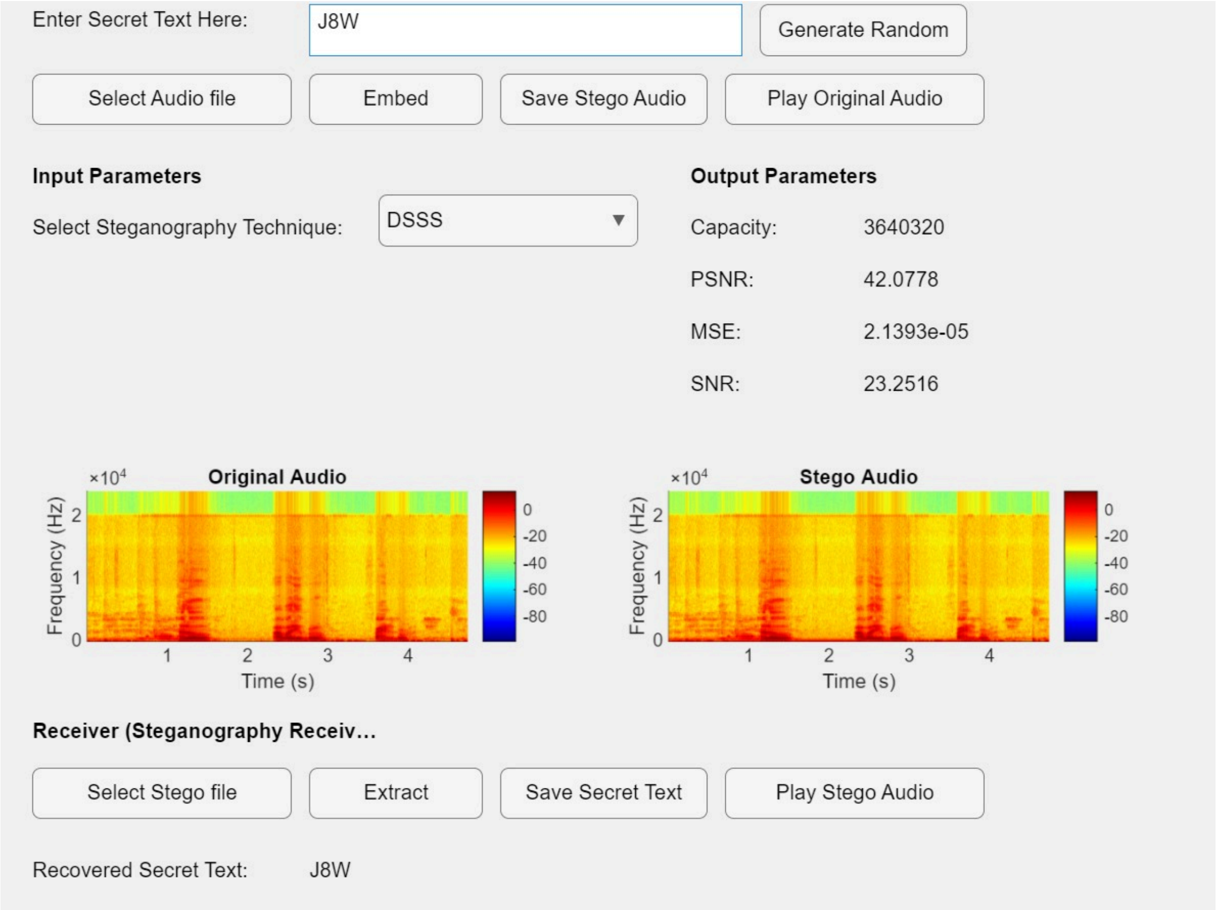
4. DSSS (Direct Sequence Spread Spectrum)

Observation/Result:

- Very low MSE (2.1393e-05) and high PSNR (42.0778): Indicate high audio quality with minimal distortion.
- Moderate capacity (3,640,320): Supports secure data embedding while preserving audio quality.
- High SNR (23.2516): Ensures robustness against noise, especially suitable for noisy environments.

Analysis:

DSSS is best suited for secure communication applications where both audio fidelity and robustness are critical. Its low distortion, high noise resistance, and secure data embedding capabilities make it ideal for secure and high-quality audio transmission.



Enter Secret Text Here: J8W

Input Parameters

Select Steganography Technique: DSSS

Output Parameters

Capacity: 3640320

PSNR: 42.0778

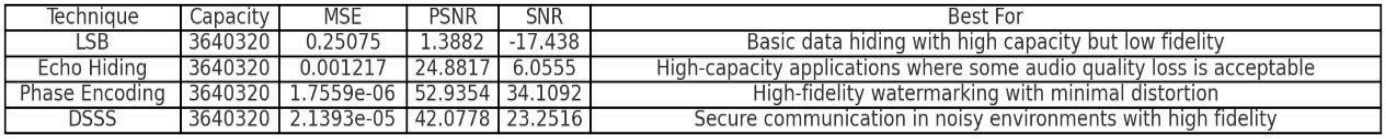
MSE: 2.1393e-05

SNR: 23.2516

Receiver (Steganography Receiv...

Recovered Secret Text: J8W

4. CONCLUSION:



Technique	Capacity	MSE	PSNR	SNR	Best For
LSB	3640320	0.25075	1.3882	-17.438	Basic data hiding with high capacity but low fidelity
Echo Hiding	3640320	0.001217	24.8817	6.0555	High-capacity applications where some audio quality loss is acceptable
Phase Encoding	3640320	1.7559e-06	52.9354	34.1092	High-fidelity watermarking with minimal distortion
DSSS	3640320	2.1393e-05	42.0778	23.2516	Secure communication in noisy environments with high fidelity

Low bit encoding

- **Embedding Techniques**: LSB of each sample in the audio is replaced by one bit of hidden information
- **Advantages**: Simple and easy way of hiding information with high bit rate
- **Drawbacks**: Easy to extract and destroy
- **Hiding Rate**: 16 kbps

Echo hiding

- **Embedding Techniques**: Embeds data by introducing an echo in the cover signal
- **Advantages**: Resilient to lossy data compression algorithms
- **Drawbacks**: Low security and capacity
- **Hiding Rate**: 50 bps

Phase spectrum

- **Embedding Techniques**: Modulate the phase of the cover signal
- **Advantages**: Robust against signal processing manipulation and data retrieval needs the original signal
- **Drawbacks**: Low capacity
- **Hiding Rate**: 333 bps

Spread spectrum

- **Embedding Techniques**: Spread the data overall signal frequencies
- **Advantages**: Provide better robustness
- **Drawbacks**: Vulnerable to time scale modification
- **Hiding Rate**: 20 bps

Best Choice for Music Copyright Protection:

Based on the results and analysis, the **Phase Encoding technique emerges as the best choice for music copyright protection**. Here’s why:

High PSNR (52.9354) and Low MSE (1.7559e-06): Phase encoding preserves excellent audio quality with minimal perceptual distortion, essential for copyrighted music.

High SNR (34.1092): High noise resistance ensures the embedded copyright data is protected across various playback and transmission conditions.

Moderate Capacity (3,640,320): It allows sufficient data embedding for copyright information without compromising audio quality.

Conclusion:

The Phase Encoding technique is ideal for embedding copyright information in music, offering a balanced approach with minimal distortion, high fidelity, and strong protection. This makes it highly suitable for applications where maintaining original audio quality is essential while embedding copyright marks securely.

Future scope:

1. **Efficient embedding:** Develop methods for higher data capacity without affecting audio quality.
2. **Real-time processing:** Enable embedding and extraction in live-streaming or communication systems.
3. **Enhanced security:** Combine encryption with steganography for stronger data protection.
4. **Compression resistance:** Improve resistance to audio compression algorithms.
5. **Multi-modal steganography**: Combine audio with video or images for greater capacity.
6. **Blockchain for copyright**: Use blockchain to secure and track audio files.
7. **Machine learning:** Apply AI for better detection and optimization of methods.
8. **Error correction:** Implement advanced error correction for improved data integrity.

These developments would boost the effectiveness and security of audio steganography.

5. REFERENCES

PAPERS:

P. P. Balgurgi and S. K. Jagtap, "Intelligent processing: An approach of audio steganography," 2012 International Conference on Communication, Information & Computing Technology (ICCICT), Mumbai, India, 2012, pp. 1-6, doi: 10.1109/ICCICT.2012.6398182.

K. Ashwini, M. Keerthana, Maria Catherine, 2014, CONCEALING DATA USING AUDIO STEGANOGRAPHY, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) IFET – 2014 (Volume 2 – Issue 01).

Djebbar, F., Ayad, B., Meraim, K.A. et al. Comparative study of digital audio steganography techniques. J AUDIO SPEECH MUSIC PROC. 2012, 25 (2012). https://doi.org/10.1186/1687-4722-2012-25.

Website:

Link to the YouTube channel (dataset)

www.youtube.com/@NoCopyrightSounds

APPENDIX

The source code can be accessed here:

GitHub Repository: https://github.com/RamananBaskar/Audio-Steganography