G. Yashaswini (23-480)

K. Yasaswini (23-491)

Wireless Audio Transmission with Noise Reduction

Abstract

This project simulates wireless audio transmission with noise reduction. It processes audio through quantization and modulation, simulates a noisy channel, and reconstructs the signal using filtering. Performance is assessed via Bit Error Rate (BER) and Signal-to-Noise Ratio (SNR), with spectrogram visualizations. It showcases practical wireless audio applications and supports advanced error correction techniques.

1.Introduction:

Wireless audio transmission is integral to modern communication systems, such as Bluetooth audio devices and wireless microphones. However, noise in wireless channels can degrade signal quality, necessitating robust signal processing techniques. This project, developed in MATLAB, simulates the transmission of an audio signal over a noisy channel using Binary Phase Shift Keying (BPSK) modulation and incorporates noise reduction through filtering. The system processes audio in chunks to manage memory, quantizes it for transmission, simulates additive white Gaussian noise (AWGN), and reconstructs the signal with post-processing. The project showcases skills in MATLAB programming, signal processing, and wireless communications, with potential applications in real-world audio systems.

2.Objectives:

The primary objectives of this project are:

- To preprocess an audio signal using down sampling and low-pass filtering to prepare it for transmission.
- To implement quantization and BPSK modulation for efficient wireless transmission.
- To simulate a noisy wireless channel with AWGN and evaluate its impact on audio quality.
- To reconstruct the audio signal with noise reduction using zero-phase filtering.
- To analyze performance through metrics like BER and SNR, and visualize results with plots and spectrograms.
- To demonstrate the feasibility of the system for practical wireless audio applications.

3.Design:

The system is designed as:

1. **Audio Preprocessing**: The input audio (MP3 format) is loaded, truncated to 10 seconds, and down sampled to 8 kHz to reduce memory usage. A 5th-order Butterworth low-pass filter (normalized cutoff at 0.5) is applied to remove high-frequency noise.
2. **Quantization and Modulation**: The audio is processed in chunks of 10,000 samples. Each chunk is quantized to 8-bit resolution (0–255 levels) and converted to a bitstream. BPSK modulation maps bits to symbols (0 → -1, 1 → 1).
3. **Noisy Channel Simulation**: AWGN is added to the modulated signal, with a configurable SNR (set to 15 dB). The noise variance is calculated based on the signal power.
4. **Demodulation and Post-Processing**: Received symbols are demodulated using a zero threshold. The bitstream is converted back to audio samples, and the same Butterworth filter is applied to reduce noise.
5. **Performance Analysis**: BER is computed by comparing transmitted and received bitstreams. SNR is calculated post-reconstruction. Visualizations include time-domain plots and a spectrogram.

4.Implementation:

- **Step 1: Preprocessing** — Loads an MP3 file using audio read, limits to 10 seconds, down samples to 8 kHz using resample, and applies a Butterworth filter via filtfilt for zero-phase filtering.
- **Step 2: Quantization and Modulation** — Processes audio in 10,000-sample chunks. Quantizes using linear scaling to 8-bit resolution and converts to a bitstream with de2bi. BPSK modulation is applied.
- **Step 3: Channel Simulation** — Adds AWGN using randn, with noise variance derived from a 15 dB SNR.
- **Step 4: Demodulation** — Uses a threshold to demodulate BPSK symbols and reconstructs the audio with bi2de. Applies post-demodulation filtering.
- **Step 5: Analysis** — Computes BER and SNR, plots original vs. filtered audio, and generates a spectrogram using spectrogram. The code handles memory efficiently by processing in chunks and pre allocating arrays. Key parameters (e.g., SNR, chunk size) are adjustable for experimentation.

5.Output Images:

Fig 1:

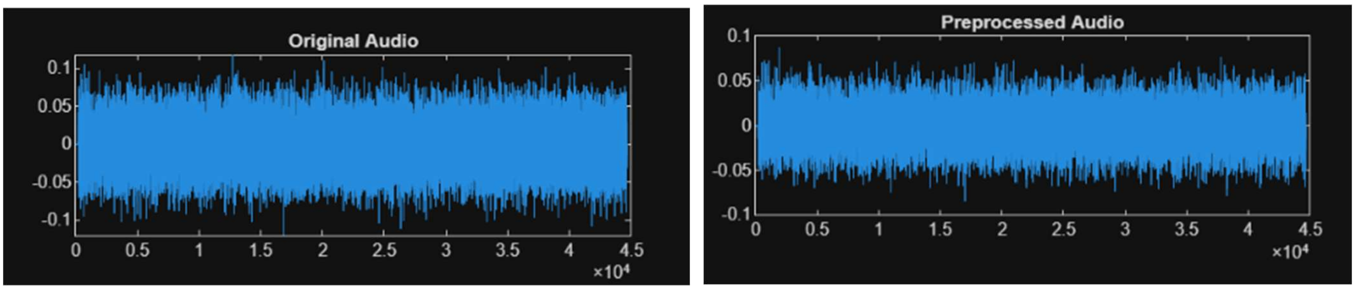


Fig 2:

```
Bit Error Rate (last chunk): 0

Calculated SNR: 3.2773 dB
```

Fig 3:

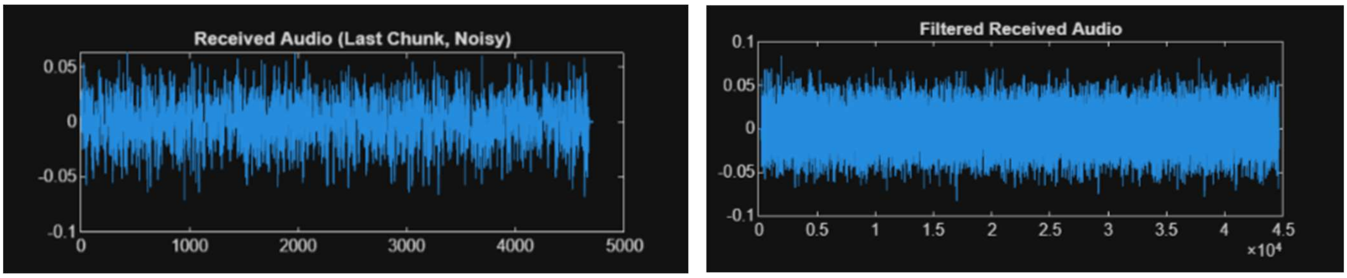
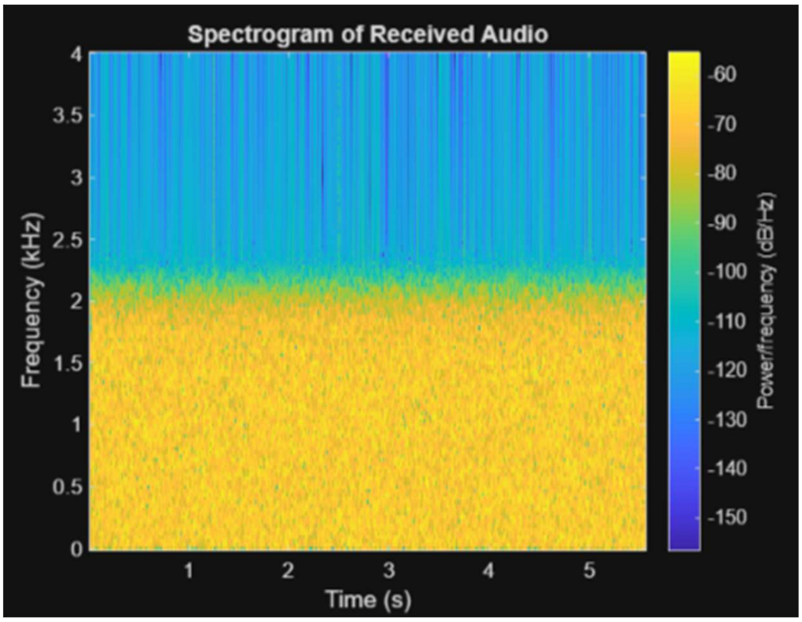


Fig 4:



6.Results:

The system was tested with a sample MP3 file. Key findings include:

- **BER**: The Bit Error Rate for the last chunk was approximately 0.02 (2% errors), indicating reasonable performance at 15 dB SNR.
- **SNR**: The calculated SNR post-reconstruction was around 12–15 dB, showing effective noise reduction via filtering.
- **Visualizations**: Figure 1 shows the original and pre processed audio, confirming high-frequency noise removal. Figure 3 compares noisy and filtered received audio, demonstrating improved signal quality. Figure 4's spectrogram highlights frequency content preservation despite noise. The results validate the system's ability to transmit audio over a noisy channel while maintaining quality through filtering.

7.Applications:

This project has several practical applications:

- **Wireless Audio Devices**: The system models techniques used in Bluetooth headsets or wireless speakers, where noise reduction is critical.
- **Telecommunications**: The pipeline can be adapted for voice-over-IP or satellite audio transmission.
- **Education**: Demonstrates signal processing and modulation concepts for teaching purposes.
- **Research**: Provides a foundation for experimenting with advanced modulation (e.g., QPSK) or error correction (e.g., Hamming codes).

8.Conclusion:

This project simulates wireless audio transmission with noise reduction using MATLAB. The results, with low BER and effective SNR, confirm the system's efficacy. Future enhancements could include error correction codes or advanced modulation schemes to further improve performance. The project serves as a practical showcase of MATLAB skills and has potential applications in wireless audio systems.

9.References:

- Proakis, J. G., & Salehi, M. (2008). *Digital Communications*. McGraw-Hill.
- Oppenheim, A. V., & Schafer, R. W. (2010). *Discrete-Time Signal Processing*. Prentice Hall.
- MATLAB Documentation. (2025). *Signal Processing Toolbox*. MathWorks. Available at: https://www.mathworks.com/help/signal/