Atividade Prática 1

Professor Daniel Campos

11 de setembro de 2025

1 Resumo

Esta atividade prática tem como objetivo aplicar conceitos de sinais discretos para a geração de sons e efeitos musicais utilizando MATLAB. Cada grupo de 2 ou 3 alunos deverá implementar funções que gerem notas musicais e efeitos sonoros baseados em operações de modulação e manipulação de sinais.

2 Requisitos do Projeto

- Implementar funções em MATLAB com nomes pré-definidos:
  - `geraNota(f0, fs, duracao, tipo)`: gera uma nota musical com frequência f_0, frequência de amostragem fs, duração e tipo de onda. O parâmetro `tipo` pode ser:
    - `'seno'` – onda senoidal
    - `'quadrada'` – onda quadrada
    - `'serra'` – dente de serra
    - `'triangular'` – onda triangular
    - `'ruido'` – ruído branco (nesse caso a informação de frequência não será usada)
  - `vibrato(x, fs, fv, beta)`: aplica vibrato ao sinal.
  - `tremolo(x, fs, fm, m)`: aplica tremolo ao sinal.
  - `decaimento(x, fs, alpha)`: aplica decaimento exponencial ao sinal.
  - `distorcao(x, L)`: aplica distorção simples (clipping).
  - `eco(x, fs, atraso, ganho)`: aplica eco (delay) ao sinal.
  - `insereSample(musica, sample, t0, fs)`: insere um trecho de áudio em um instante de tempo específico dentro da música.
- Testar cada função individualmente com exemplos.

3 Métodos e Técnicas

Serão utilizadas formas de onda clássicas como sinais básicos (senoidal, quadrada, triangular, serra, ruído), operações de soma, multiplicação e modulação para a criação de efeitos. A implementação será feita em MATLAB.

4 Procedimentos

4.1 Gerador de Notas Musicais

- Criar a função `geraNota(f0, fs, duracao, tipo)`.

- Testar a função para as notas:
  - Lá = 440 Hz
  - Dó = 261.63 Hz
  - Mi = 329.63 Hz
- Tocar cada nota em diferentes formas de onda (`seno`, `quadrada`, `serra`, `triangular`, `ruido`) usando `sound`.

4.2 Formação de Acordes pela Soma de Notas

- Gerar acordes maiores básicos, somando notas criadas com `geraNota`.
- Exemplos de acordes:
  - Dó maior: Dó (261.63 Hz), Mi (329.63 Hz), Sol (392.00 Hz).
  - Lá menor: Lá (440 Hz), Dó (261.63 Hz), Mi (329.63 Hz).
- Reproduzir os acordes e comparar timbres de diferentes formas de onda.

4.3 Efeitos Sonoros

Implementar cada efeito em uma função separada. O efeito pode ser aplicado em uma nota individual por multiplicação ponto a ponto com o sinal de entrada.

1. **Vibrato (modulação em frequência):**

$$x(t) = \sin\left(2\pi f_0 t + \beta \sin(2\pi f_v t)\right)$$

   Função: `vibrato(x, fs, fv, beta)`.

2. **Tremolo (modulação em amplitude):**

$$x(t) = (1 + m\sin(2\pi f_m t))\sin(2\pi f_0 t)$$

   Função: `tremolo(x, fs, fm, m)`.

3. **Decaimento exponencial (envelope):**

$$x(t) = e^{-\alpha t}\sin(2\pi f_0 t)$$

   Função: `decaimento(x, fs, alpha)`.

4. **Distorção simples (clipping):**

$$x_{clip}(t) = \begin{cases} L, & x(t) > L \\ -L, & x(t) < -L \\ x(t), & \text{caso contrário} \end{cases}$$

   Função: `distorcao(x, L)`.

5. **Eco (delay com ganho):**

$$y[n] = x[n] + g \cdot x[n - atraso]$$

   Função: `eco(x, fs, atraso, ganho)`.

4.4 Inserção de Samples

- Implementar a função `insereSample(musica, sample, t0, fs)`.
- Testar inserindo notas ou acordes em diferentes posições do vetor de áudio para criar sequências musicais.
- Dicas de implementação:
  - Calcular o índice inicial da inserção: $inicio = round(t0 \cdot fs) + 1$
  - Somar o sample ao vetor música no intervalo correspondente.
  - Garantir que o tamanho do vetor música seja suficiente para receber o sample.

4.5 Composição Musical

Utilizar todas as funções implementadas para criar uma pequena música com harmonia e melodia.

- A música deve conter:
  - Uma sequência de acordes como acompanhamento (harmonia).
  - Uma sequência de notas individuais formando a melodia principal.
  - Aplicação de pelo menos dois efeitos em momentos diferentes (exemplo: vibrato na melodia, tremolo na harmonia, eco em um acorde longo, etc.).
- Sugestão de progressão harmônica típica (duração 1 s por acorde):

$$C \text{ (Dó maior)} \rightarrow G \text{ (Sol maior)} \rightarrow Am \text{ (Lá menor)} \rightarrow F \text{ (Fá maior)}$$

- Sugestão de melodia (cada nota com 0.5 s de duração):
  - Compasso 1 (C): C – C
  - Compasso 2 (G): D – C
  - Compasso 3 (Am): F – E
  - Compasso 4 (F): G – F
- Variações a serem testadas:
  - Alterar a forma de onda de cada nota ou acorde.
  - Repetir, inverter ou transpor trechos da melodia.
  - Ajustar a duração das notas e dos acordes.
- Combinar os efeitos para enriquecer a sonoridade, por exemplo:
  - Aplicar vibrato na melodia.
  - Tremolo na harmonia.
  - Eco em notas finais de cada compasso.
  - Distorção ou decaimento para criar texturas diferentes.
- Plotar trechos curtos da música usando `plot` para visualizar as ondas antes de reproduzir.

5 Observações

- Cada função deve ser implementada em um arquivo `.m` separado, com o mesmo nome da função.
- Entregar todos os arquivos `.m` e um script principal que demonstre a execução das funções e a música final.
- O script principal deve mostrar exemplos de:
  - Notas simples em diferentes formas de onda.
  - Formação de acordes pela soma de notas.
  - Aplicação de cada efeito separadamente.
  - Inserção de samples para formar a música.
  - Reprodução da música final com harmonia, melodia e efeitos.

6 Efeitos por Convolução

Nesta parte da prática, o objetivo é explorar a criação de efeitos sonoros utilizando convolução. A convolução permite simular sistemas LTI (Lineares e Invariantes no Tempo), aplicando a um sinal qualquer uma resposta ao impulso definida. Isso pode ser feito de forma matemática (com filtros FIR e IIR clássicos) ou de forma experimental (gravando sons que funcionam como respostas ao impulso).

Além da melodia criada na Parte 1, nesta etapa também serão fornecidos dois áudios: `guitar.wav`, contendo uma gravação de guitarra limpa e `voz.wav` contendo uma voz limpa.

6.1 Filtros clássicos simulados por convolução

Nesta etapa, vamos implementar dois efeitos sonoros utilizando a técnica de convolução com respostas ao impulso simuladas. O objetivo é aplicar os efeitos sobre esse sinal de áudio por meio de convolução:

Reverb curto com ruído: Crie uma resposta ao impulso exponencial modulada por ruído branco:

$$h[n] = e^{-\alpha n} \cdot w[n], \quad n = 0, 1, \ldots, N$$

onde $w[n]$ é ruído branco com média zero e variância controlada. O parâmetro α define a taxa de decaimento e N a duração total da resposta. Esse modelo simula melhor a difusão de uma sala, pois cada reflexão é levemente aleatória. A convolução é feita por

$$y[n] = x[n] * h[n],$$

onde $x[n]$ é o sinal.

Delay/Eco: Crie uma resposta ao impulso contendo um ou mais ecos espaçados no tempo:

$$h[n] = \delta[n] + g \cdot \delta[n - D] + g^2 \cdot \delta[n - 2D] + \ldots$$

onde D é o atraso em amostras e g é o fator de ganho dos ecos sucessivos ($|g| < 1$ para evitar instabilidade). Este filtro cria repetições periódicas do sinal original. A convolução é novamente aplicada:

$$y[n] = x[n] * h[n].$$

Tarefas:

- Implementar os dois filtros no MATLAB, gerando as respostas ao impulso $h[n]$ explicitamente como vetores.
- Aplicar a convolução com os sinais fornecidos.
- Plotar a forma de onda do sinal original e dos sinais com efeito aplicado.
- Discutir como os parâmetros influenciam o resultado sonoro.

6.2 Efeitos Experimentais com Convolução

Agora cada grupo deverá criar três efeitos diferentes baseados em respostas ao impulso experimental. Essas respostas podem ser obtidas de diferentes formas.

Gravar um som rápido e curto como palma, estalo, estouro em um ambiente fechado como banheiro, corredor, sala fechada. Também é possível colocar o microfone ou celular em um ambiente ressonante como dentro de um armário, tubo, lata ou caixa. Também pode ser gravado o som de objetos percussivos como uma bolinha de ping-pong quicando, batida em um metal, vidro ou outro objeto com ressonância. Outra opção é gravar objetos ruidosos, como chuva, chocalho, chaves.

Os efeitos também podem ser combinados entre si ou combinados com os sinais e efeitos da parte anterior. Seja criativo.

Guia para criar o efeito:

1. Grave um áudio e importe como vetor $h[n]$ que represente a resposta ao impulso desejada.
2. Normalizar $h[n]$ para evitar distorção (máximo valor entre -1 e 1).
3. Convoluir o sinal musical com $h[n]$:

$$y[n] = x[n] * h[n]$$

4. Ou, em MATLAB:

```
y = conv(x, h, 'same');
```

6.3 Teste

Todos os efeitos devem ser aplicados tanto:

1. Sobre a melodia criada na Parte 1.
2. Sobre o áudio de guitarra limpa (`guitar.wav`) fornecido.
3. Sobre um áudio de voz limpa (`voz.wav`) fornecido.

Em seguida, discuta o efeito obtido em cada caso, comparando os resultados no sinal sintético e no instrumento real.