Projeto 1 - Processamento de Sinais multimídia

Amélia O. F. da S

1 Objetivos

O projeto tem como objetivos:

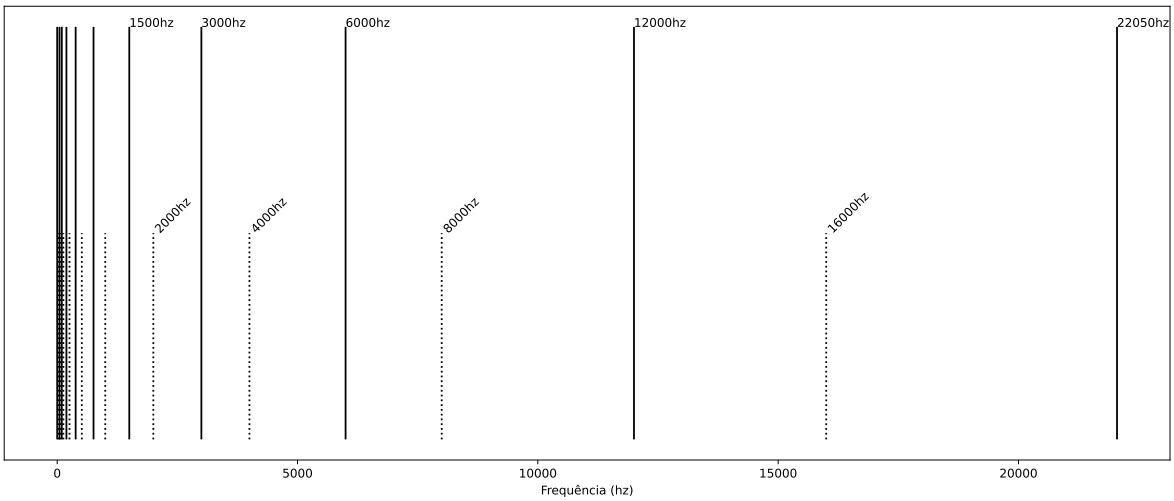
- Demonstração de domínio básico prático do conhecimento de filtros SLIT FIR via:
  - Implementação de um equalizador de bandas;
  - Implementação de um visualizador de energia de bandas.
- Demonstração de domínio básico teórico do conhecimento de filtros SLIT FIR pela descrição e justificativas teóricas das implementações.

2 Planejamento

2.1 Arquitetura de DSP

A fim cumprir os objetivos listados acima, proponho a arquitetura descrita na figura 3.

A aplicação processará sequencialmente as amostras de entrada e aplicará um filtro simplificado calculado a partir de 10 filtros passa-faixa dividindo o espectro entre as frequências requeridas no projeto. Para o cálculo dos valores do gráfico de barras, aplicaremos periodicamente uma FFT, somando as energias das faixas especificadas.



Pontilhadas: frequências "centrais"das bandas
Sólidas: frequências-limite das bandas

Figura 1: Bandas de filtragem

2.2 Detalhes e justificativa de implementação

Inicialmente propus a arquitetura ilustrada na figura 2, que consistia em aplicar separadamente os 10 filtros e guardar seus resultados em acumuladores para calcular a energia de cada banda.

Mesmo tendo uma complexidade maior que a FFT, intuitivamente imaginei que para a pequena quantidade de filtros que temos e para uma largura reduzida de filtro, este método poderia custar menos operações.

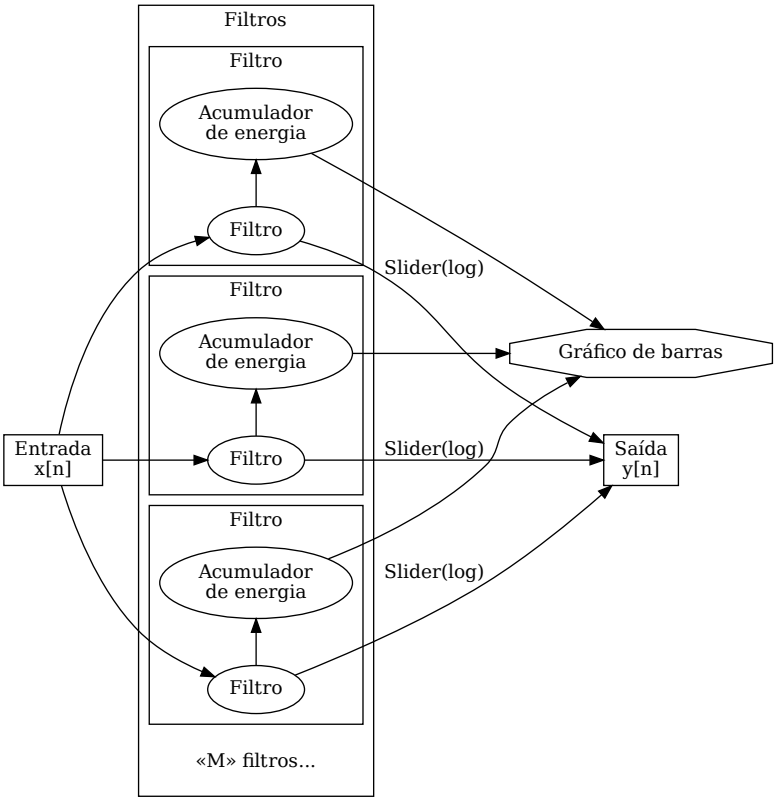


Figura 2: Má arquitetura proposta

Buscando uma comparação mais concreta entre essa proposta e a da figura 3, fiz uma análise mais detalhada.

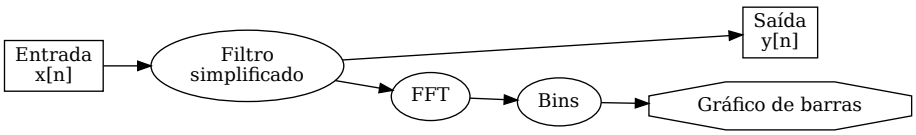


Figura 3: Nova arquitetura proposta

Na arquitetura sem FFT, para uma quantidade de amostras S e M filtros FIR de largura W, o número de operações a serem feitas para calcular todos os valores necessários será:

- Multiplicações: $S(WM)$
  - WM para multiplicação das amostras pelos coeficientes em $(\sum_{i}^{W} a_i x[n-i])$ e ponderação dos sliders;

    (Para um peso w, $w\sum_{i}^{W} a_i x[n-i] = \sum_{i}^{W}(a_i w)x[n-i]$, então pré-calculamos $a_i w$)
- Somas/Subtrações: $S(WM + M + 2M)$
  - $WM + M$ para os filtros:
    - WM para o somatório $\sum_{i}^{W} a_i x[n-i]$;
    - M para somar todos.
  - $2M$ para o acumulador de energia:

    - 1 soma para adicionar a amostra atual;
    - 1 subtração para remover a última amostra.

Para a arquitetura com FFT, teríamos, presumindo $S = 2^p$ e um número de amostragens do gráfico de energia ao longo do *chunk* de memória $G = 2^k, k < p$, gerando "pacotes"de $L = S/G$:

- Multiplicações: $S(W) + G\frac{L}{2}log_2 L$
  - W para multiplicação das amostras pelos coeficientes;
  - $\frac{L}{2}log_2 L$ para a FFT.
- Somas/subtrações: $S(W) + G(Llog_2 L + L)$
  - W para o somatório de $y[n]$.
  - $Llog_2 L$ para a FFT;
  - L para aglomeração das "bins"da FFT.

Presumindo 4096 amostras, podemos explorar as complexidades das duas implementações para diversas combinações dos parâmetros W e G.

Para **todas** as combinações exploradas (todos W entre 1 e 500, todos G entre 1 e 11), o método com FFT foi superior (vide `misc/operacoes.py`).

N_{fft}/N_{naive}	**Multiplicações**	**Somas**
Média	9.68	9.64
Máxima	9.98	10
Mínima	1.53	3.07

Tabela 1: Razão entre o número de operações da abordagem com FFT e sem (N_{fft}/N_{naive})

2.3 Arquitetura da aplicação

A aplicação se dividirá em três componentes principais:

- Provedor de informação dos sliders;
  - Codificado em Python;
  - Apresentará a interface de alteração dos filtros para o usuário;
  - Proverá os coeficientes do filtro para o processador de sinais via socket UNIX.
- Host de filtragem;
  - Codificado em C;
  - Receberá coeficientes de filtragem via socket UNIX;
  - Receberá um feed de som do PulseAudio, aplicará o filtro FIR caracterizado pelos parâmetros guardados e enviará um feed de som de volta ao PulseAudio;
- Visualizador de gráfico de barras.
  - Codificado em Python;
  - Receberá um feed de som do PulseAudio e gerará um gráfico de barras de intensidade de cada banda.

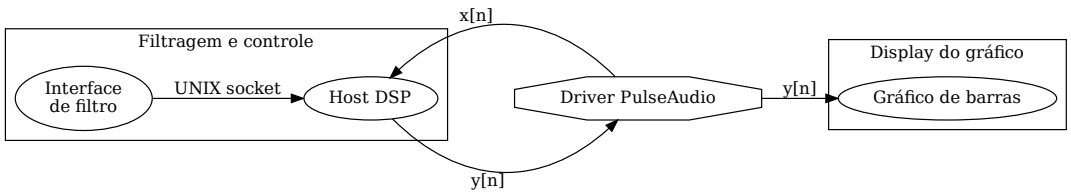


Figura 4: Processos e conexões relevantes

Destes processos, dois grupos são basicamente independentes:

- Processos de controle e filtragem
  - Lidam com a filtragem do áudio e configuração do filtro;
  - Lidam com a interação com o usuário.
- Processo de display do gráfico
  - Lida com o cálculo das intensidades de cada *bin* de frequências.
  - Lida com a ilustração das intensidades com um gráfico de barras.

O isolamento dos dois grupos, bem como o desenho do host de DSP de forma "reativa"(funciona continuamente até que alguma alteração venha da interface, com mínima interrupção) permite que o feed de som seja contínuo, sem falhas provindas de gargalos de processamento, e que o display visual, para o qual continuidade não é tão importante, possa lidar com o feed de som em seu próprio "ritmo".