



UNIVERSIDADE FEDERAL DO AMAZONAS
FACULDADE DE TECNOLOGIA
ENGENHARIA DA COMPUTAÇÃO

CLASSIFICAÇÃO DE SÉRIES TEMPORAIS COM APRENDIZAGEM PROFUNDA

CLEMILTON VASCONCELOS PEREIRA

MANAUS-AM
2018

CLEMILTON VASCONCELOS PEREIRA

CLASSIFICAÇÃO DE SÉRIES TEMPORAIS COM
APRENDIZAGEM PROFUNDA

Monografia apresentada à Coordenação do
Curso de Engenharia da Computação da
Universidade Federal do Amazonas, como
parte dos requisitos necessários à obtenção
do título de Engenheiro de Computação.

Orientador: RAFAEL GIUSTI

MANAUS-AM

2018

Ficha Catalográfica

Ficha catalográfica elaborada automaticamente de acordo com os dados fornecidos pelo(a) autor(a).

C957s Cruz Junior, António César Vieira da
Sistema de detecção de frequência fundamental para música /
António César Vieira da Cruz Junior. 2018
66 f.: il. color; 31 cm.

Orientador: Waldir Sabino da Silva Júnior
TCC de Graduação (Engenharia da Computação) - Universidade
Federal do Amazonas.

1. Frequência fundamental. 2. Processamento digital de áudio. 3.
Transcrição musical. 4. Detecção automática. I. Silva Júnior, Waldir
Sabino da II. Universidade Federal do Amazonas III. Título

CLASSIFICAÇÃO DE SÉRIES TEMPORAIS COM APRENDIZAGEM PROFUNDA

Clemilton Vasconcelos Pereira

**MONOGRAFIA SUBMETIDA AO CORPO DOCENTE DO CURSO DE
ENGENHARIA DA COMPUTAÇÃO DA UNIVERSIDADE FEDERAL DO
AMAZONAS COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A
OBTENÇÃO DO GRAU DE ENGENHEIRO.**

Aprovada por:

Prof. Rafael Giusti, D. Sc.

Prof., M. Sc.

Prof. , D. Sc.

Manaus

Julho de 2018

*Dedico este trabalho à minha mãe
Lúcia, "In memoriam", também aos meus
pais, Antonio e Elenir, e às minhas avós
Jacyra e Maria, meus exemplos de vida.*

Agradecimentos

Antes de tudo, a Deus, autor da vida, cujo a bondade e a misericórdia me seguem a cada dia. Tu és fiel, Senhor!

Aos meus pais, Antonio e Elenir, que sempre me apoiaram de todas as maneiras possíveis, à minha amada esposa Yasmin, sempre ao meu lado, e aos meus familiares, que sempre me incentivaram a persistir.

Ao meu orientador, prof. Dr. Waldir Sabino, pela paciência, disposição e direcionamento durante a execução deste projeto, e também pelos conselhos que levo para minha vida profissional.

A todos os professores da UFAM, em especial o prof. Dr. César Melo, aos amigos de turma Rosmael, Helton, João Victor, Gabriel, Jordan, Kabashima, Eduardo, Luiz Henrique, Micael e Clemilton, e aos demais amigos e colegas da UFAM pelo convívio, aprendizado e cooperação durante todos esses anos. Vocês foram essenciais nessa caminhada.

À família Machado, por todo o cuidado no momento em que mais precisei, e aos irmãos da Segunda Igreja Batista de Manaus, que me adotaram assim que cheguei nesta cidade. Também à Keyseane, que me ajudou em momentos importantes na escrita deste trabalho, e aos meus amigos José Eduardo, Vanessa e Débora.

A todos que, de algum modo, colaboraram para que este momento se concretizasse:

Muito obrigado!

“Se algum de vocês tem falta de sabedoria, peça-a a Deus, que a todos dá livremente, de boa vontade; e lhe será concedida.”

(*Tiago 1.5*)

Resumo

Existem diferentes caminhos para o reconhecimento automático de uma notas musical, e um deles é por meio da frequência fundamental (f_0). Muitas soluções e métodos de detecção automática de f_0 já foram desenvolvidas e apresentadas, obtendo resultados positivos. Todavia, é difícil que uma única solução de detecção seja de fato eficaz em ambientes muito diferentes daqueles para o qual a solução foi proposta. Diante disso, o presente projeto aborda o desenvolvimento de um sistema de detecção de f_0 para áudios musicais monofônicos, através de processamento digital de sinais, com o intuito de mapear os áudios, fornecendo as frequências fundamentais soadas em cada instante de tempo. Este trabalho também avaliou o sistema desenvolvido, por meio de experimentação a partir de uma base de dados construída para este fim. Percebeu-se que o sistema proposto apresenta boas respostas, sendo necessário melhorias em relação ao tratamento do efeito de ressonância.

Palavras-chave: Detecção automática de frequência fundamental, Processamento digital de áudio, Transcrição musical.

Abstract

There are different ways for a automatic recognition of musical note, one of them is through fundamental frequency (f_0). Many solutions and methods of automatic detection of f_0 already developed and presented, getting positive results. However, it's hard that one only solution has been effective in environments much different from these that it was proposed. Therefore, the present project approaches the development of a f_0 detector system for monophonic musical audios through digital signal processing, with the intention of mapping it, recording the fundamental frequencies sounded at every instant time. This work also evaluated the system developed, through experiments using a database built for it. It was perceived that the proposed system presents good answers, being necessary improvements in the treatment of the resonance effect.

Keywords: Automatic Detection of fundamental frequency, Digital Audio Processing, Musical transcription.

Listas de Figuras

2.1	Codificação One-hot	6
2.2	Princípio dos k-vizinhos mais próximos	7
2.3	Representação simplificada de um neurônio	8
2.4	Modelo matemático de um neurônio	9
2.5	Multilayer Perceptron. Cada círculo representa um neurônio mostrado anteriormente	10
2.6	Amostragem e Quantização	16
2.7	Espectrogramas no MATLAB [®]	21
3.1	Estrutura do sistema desenvolvido	22
3.2	Diagrama do bloco de pré-processamento	23
3.3	Diagrama do algoritmo de deslocamento da janela	26
3.4	Interface GUIDE	29
4.1	Clarinete e sua forma de onda	34
4.2	Interface de experimentação para áudio ”Escala Bb Maior” com clarinete .	35
4.3	Escala de Bb Maior com clarinete	36
4.4	Interface de experimentação para áudio da música “ <i>Agnus Dei</i> ” com clarinete	37
4.5	Música “ <i>Agnus Dei</i> ” com clarinete	37
4.6	Interface de experimentação para áudio da Música “Super Mário Brós” com clarinete	38
4.7	Música “Super Mário Brós” com clarinete	39
4.8	Sax alto e sua forma de onda	40
4.9	Interface de experimentação para áudio ”Escala Eb Maior” com sax alto .	40
4.10	Escala Eb Maior com sax alto	41

4.11 Interface de experimentação para áudio da Música “Deus Cuida de Mim” com sax alto	42
4.12 Música “Deus Cuida de Mim” no sax alto	43
4.13 Interface de experimentação para áudio da música “ <i>Summertime</i> ” com sax alto	44
4.14 Audio da música “ <i>Summertime</i> ” no sax alto	45
4.15 Piano e sua forma de onda	46
4.16 Interface de experimentação para áudio da escala C Maior com piano eletrônico	46
4.17 Escala de C Maior no piano eletrônico	47
4.18 Interface de experimentação para áudio da música “Eu Navegarei” com piano eletrônico	48
4.19 Áudio da música “Eu Navegarei” tocado com piano eletrônico	49
4.20 Interface de experimentação para áudio da música “ <i>Skyfall</i> ” com piano	49
4.21 Música “ <i>Skyfall</i> ” tocada no piano	50
4.22 Violão e sua forma de onda	51
4.23 Interface de experimentação para áudio da escala de C Maior com violão	51
4.24 Escala de C Maior no violão	52
4.25 Interface de experimentação para áudio da música “Nona Sinfonia” com violão	53
4.26 Música “Nona Sinfonia” tocada no violão.	54
4.27 Interface de experimentação para áudio da música “Deus de Abraão” com violão	54
4.28 Música “Deus de Abraão” tocada no violão.	55

Lista de Tabelas

2.1	Representação da base de dados	5
2.2	Notas musicais da 4 ^a oitava e suas frequências	17
3.1	Lista de Gravações da Base de Dados	30
4.1	Comparativo entre os áudios	56

Lista de Abreviaturas e Siglas

PDS	Processamento Digital de Sinais
MIDI	Interface Digital de Instrumentos Musicais – do inglês <i>Musical Instrument Digital Interface</i>
DTFT	Transformada de Fourier em Tempo Discreto – do inglês <i>Discret-Time Fourier Transform</i>
DFT	Transformada Discreta de Fourier – do inglês <i>Discrete Fourier Transform</i>
FFT	Transformada Rápida de Fourier – do inglês <i>Fast Fourier Transform</i>
STFT	Transformada de Fourier de Tempo Curto – do inglês <i>Short-Time Fourier Transform</i>

Lista de Símbolos

Símbolos Matemáticos

f_s	Frequência de amostragem
f_0	Frequência fundamental
Hz	Hertz - Unidade do Sistema Internacional (SI) para frequência
bpm	Batidas por minutos

Sumário

1	Introdução	1
1.1	Organização do Trabalho	2
1.2	Objetivo	3
2	Fundamentação Teórica	4
2.1	Aprendizado de Máquina	4
2.1.1	Aprendizado Supervisionado	5
2.1.2	Normalização e One-Hot Encoding	6
2.1.3	K-Nearest Neighbors (KNN)	7
2.2	Redes Neurais Artificiais	8
2.2.1	Inspiração Biológica e Perceptron	8
2.2.2	Perceptron multicamadas	9
2.2.3	Camada Softmax	11
2.2.4	Hiperparâmetros de uma Rede Neural	11
2.2.4.1	Mini-batches	11
2.2.4.2	Taxa de aprendizado	11
2.2.4.3	Função de Perda(Loss Function)	12
2.2.4.4	Otimizadores	12
2.3	Séries Temporais	12
2.3.1	Aplicações de Séries Temporais	13
2.4	Classificação de Séries Temporais	14
2.5	Frequência Fundamental	14
2.6	Pitch	14
2.7	Detecção Automática de f_0	14
2.8	Digitalização de um Sinal de Som	15

2.8.1	Amostragem	15
2.8.2	Quantização	15
2.8.3	Codificação	16
2.9	Notas Musicais	16
2.10	Ressonância	17
2.11	Domínios	17
2.12	Transformada de Fourier	18
2.13	Transformada Discreta de Fourier	18
2.14	Transformada Rápida de Fourier	19
2.15	Transformada de Fourier de Tempo Curto	20
2.16	Espectrograma	20
3	Sistema Proposto	22
3.1	Visão geral	22
3.2	Bloco de Pré-Processamento	23
3.3	Bloco de Detecção de f_0	24
3.4	Interface GUIDE	27
3.5	Construção da Base de Dados	29
4	Experimentos	32
4.1	Métricas	32
4.2	Objetivos	33
4.3	Resultados Obtidos	33
4.3.1	Clarinete	34
4.3.1.1	Escala: Bb Maior	34
4.3.1.2	Música: <i>Agnus Dei</i>	35
4.3.1.3	Música: Super Mário Brós	35
4.3.2	Sax Alto	38
4.3.2.1	Escala: Eb Maior	40
4.3.2.2	Música: Deus Cuida de Mim	42
4.3.2.3	Música: <i>Summertime</i>	42
4.3.3	Piano Eletrônico	44
4.3.3.1	Escala: C Maior	44

4.3.3.2	Música: Eu Navegarei	48
4.3.3.3	Música: <i>Skyfall</i>	48
4.3.4	Violão	50
4.3.4.1	Escala: C Maior	51
4.3.4.2	Música: Nona Sinfonia	52
4.3.4.3	Música: Deus de Abraão	53
4.4	Discussões	56
5	Conclusão	58
	Referências Bibliográficas	60

Capítulo 1

Introdução

Como em todas as outras áreas, a música também vem progredindo com a tecnologia, tornando-se com maior praticidade, acessibilidade e inovações. A praticidade pode ser vista através das inúmeras ferramentas musicais que estão disponíveis no mercado, como os afinadores digitais, que podem ser encontrados até mesmo em aplicativos para *smartphone*. A evolução do poder computacional também reduziu o custo de desenvolvimento nesse setor, e ampliou os horizontes para uma infinidade de novas possibilidades para as ferramentas musicais. Atualmente, a transcrição de fala já é uma realidade em processamento digital de sinais, e tem-se empreendido grande esforço na tentativa de transcrever notas musicais^[1] ^[2] ^[3].

Existem diferentes caminhos para o reconhecimento de notas musicais, e um deles é por meio da frequência fundamental (f_0). Muitas soluções e métodos de detecção automática de f_0 já foram desenvolvidas e apresentadas, obtendo resultados positivos. Todavia, é difícil que uma única solução de detecção de f_0 seja de fato eficaz em todos os contextos possíveis. Deste modo, cada solução de detecção tem sua aplicação para a qual é ideal, e não há garantias de bons resultados em alguns outros contextos, como, por exemplo, ambientes ruidosos. Por isso, cada contexto demanda que as soluções propostas sejam analisadas, tendo em vista os fenômenos ao qual o sistema será submetido, para só então eleger-se o ideal para tal situação.

Dentro dessa realidade, o presente projeto aborda o desenvolvimento de um sistema de detecção de f_0 para áudios musicais monofônicos, através de processamento digital de sinais, com o intuito de mapear os áudios, fornecendo as frequências fundamentais soadas em cada instante de tempo e identificando os fenômenos aos quais um sistema nesse

contexto é submetido. A detecção de f_0 será realizada por meio da Transformada de Fourier de Curto Tempo (STFT), considerando f_0 como a frequência de maior amplitude dentro de cada janela. Foi construída uma base de dados com áudios de 4 tipos de instrumentos musicais para a realização de experimentos, visando validar as detecções realizadas pelo sistema. Este trabalho propõe-se ainda a avaliar a metodologia adotada por meio desses experimentos realizados.

1.1 Organização do Trabalho

Este trabalho escrito está organizado 3 capítulos, onde discorrem-se os fundamentos teóricos, os detalhes do sistema proposto e os experimentos realizados.

No Capítulo 2, Fundamentação Teórica, são abordados, de modo sucinto, os termos e conceitos utilizados na execução deste projeto. Serão tratado os conceitos de sinal de som e suas características, frequência fundamental e sua detecção, *pitch*, digitalização de um sinal, mencionando os processos de amostragem, quantização e codificação, conceitos de notas musicais, ressonância, domínios e transformadas, transformada discreta de fourier, transformada rápida de fourier e transformada de curto-tempo, além dos espectrogramas e suas formas.

O Capítulo 3, Sistema Proposto, detalha o desenvolvimento do sistema, especificando a metodologia adotada para o projeto. É apresentado, inicialmente, uma visão geral do sistema, com uso de diagrama de blocos. Os blocos de pré-processamento e de detecção da frequência fundamental são abordados de forma mais detalhada e expõe-se também o algoritmo usado para a detecção da f_0 . Explica-se ainda como foi implementada a análise por janelas de curto-tempo, bem como a forma de visualização dos resultados obtidos. O capítulo aborda ainda como foi desenvolvida a interface de experimentação e como construiu-se a base de dados.

Por fim, o Capítulo 4, Experimentos, explica as métricas utilizadas para a avaliação do sistema, os objetivos do experimento, os resultados obtidos e as discussões acerca desses resultados. Os resultados são apresentados de forma detalhada, para cada áudio da base de dados, separados por instrumento. Os resultados vão sendo analisados à medida em que são apresentados, sendo feito, ao fim de todos os experimentos, uma discussão mais detalhada acerca das informações obtidas por meio da experimentação.

1.2 Objetivo

Este projeto tem por objetivo desenvolver um sistema de detecção de frequência fundamental para áudios musicais monofônicos, através de processamento digital de sinais, com o intuito de mapear os áudios, fornecendo as fundamentais soadas em cada instante de tempo. Para isso, faz-se necessário:

1. Implementar um sistema de processamento digital de áudio que, por meio da STFT, possibilite uma análise do sinal no domínio da frequência.
2. Construir uma base de dados contendo áudios monofônicos para diferentes instrumentos, e seus respectivos arquivos de registro dos tempos de notas soadas e suas frequências fundamentais.
3. Desenvolver uma interface gráfica de experimentação para avaliar o sistema desenvolvido.

Capítulo 2

Fundamentação Teórica

Este capítulo aborda, de forma sucinta, os fundamentos teóricos que foram aplicados no planejamento e implementação deste projeto.

2.1 Aprendizado de Máquina

Aprendizado de máquina(AM) é uma área da inteligência artificial cujo objetivo é o desenvolvimento de técnicas computacionais sobre o aprendizado, bem como a construção de sistemas capazes de adquirir conhecimento de forma automática. Um sistema baseado em aprendizado trabalha por meio de experiências acumuladas e de soluções bem-sucedidas de problemas anteriores [4]. Normalmente, algoritmos de aprendizado utilizam experiências anteriores, denominadas conjunto de treino, para auxiliar no processo de tomada de decisão.

Existem três diferentes tipos de aprendizado: supervisionado, não-supervisionado e semi-supervisionado. A diferença entre esses tipos de aprendizado é se o método utiliza ou não utiliza o rótulo de treino. No aprendizado supervisionado, esse rótulo é conhecido, enquanto que no aprendizado não-supervisionado os exemplos não vistos. Já no aprendizado semi-supervisionado, o conjunto de treinamento consiste de uns poucos exemplos rotulados e muitos não rotulados[5]. O escopo deste trabalho se encontra no aprendizado supervisionado.

	A_1	A_2	\dots	A_M	Classe(Y)
E_1	x_{11}	x_{12}	\vdots	x_{1M}	y_1
E_2	x_{21}	x_{22}	\vdots	x_{2M}	y_2
\vdots	\vdots	\vdots	\ddots	\vdots	\vdots
E_N	x_{N1}	x_{N2}	\vdots	x_{NM}	y_N

Tabela 2.1: Representação da base de dados

2.1.1 Aprendizado Supervisionado

O objetivo do aprendizado supervisionado é construir um modelo que consegue fazer predição através de instâncias de uma base de dados rotuladas. Cada instância é representada por um conjunto de características. Na Tabela 2.1 é mostrada a estrutura de uma base. Neste exemplo cada vetor $E_i = [x_{i1}, \dots, x_{iM}]$ se refere a classe y_i .

A idéia da aprendizagem supervisionada é o conseguir encontrar uma função desconhecida f (função conceito) tal que $y = f(\mathbf{x})$, onde o vetor \mathbf{x} são os atributos de uma instância específica. Na prática, a função f deve conseguir prever o valor correto y_i de uma instância E_i não vista. Normalmente, o número de exemplos da base de dados não é suficiente para descrever a função conceito. Nesse caso, o classificador é visto como uma hipótese h que aproxima f , ou seja, $h(x) \approx f(x)$. Caso os valores dos rótulos y_1, y_2, \dots, y_N sejam numéricos o problema é denominado de *regressão*, caso sejam valores categóricos o problema é denominado de *classificação*.

De maneira geral, a base de dados é dividida em dois conjuntos: conjunto de *treino* e conjunto de *teste*. O conjunto de treinamento é utilizado para ajustar o classificador. Como dito anteriormente, o classificador é uma hipótese da função conceito f , logo, é fundamental que o conjunto de treinamento tenha uma distribuição o mais semelhante possível do conjunto original. O conjunto de teste é utilizado para avaliar o modelo construído. Idealmente, esse conjunto não deve ter exemplos em comum com o conjunto de treinamento.

Em alguns casos, pode ser necessário utilizar um conjunto de *validação*, extraído do conjunto de treinamento, para realizar ajustes no modelo construído pelo algoritmo de aprendizado. Logo tem-se três conjuntos: *treino*, *validação* e *teste*. O treino é utilizado para aprendizagem do algoritmo. O modelo é avaliado através do conjunto de validação. É feita uma alteração dos parâmetros do classificador e outro treinamento é realizado.

O intuito é melhorar o desempenho do modelo através desses ”ajustes”. Dessa maneira os exemplos de validação são indiretamente ”vistos” durante o processo de aprendizado, o que obriga que esses exemplos sejam diferentes dos exemplos de testes.

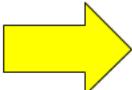
2.1.2 Normalização e One-Hot Encoding

Os algoritmos de aprendizagem de máquina aprendem através dos dados. Dados do mundo real apresentam valores que estão em distintas faixas. A fim de evitar que algum atributo predomine sobre outro ou que inclua alguma ponderação indesejada ao induzir um modelo de AM, é comum fazer uma normalização dos valores de cada atributo. Um forma de normalizar os dados é mostrada na Equação 2.1:

$$x_{ij} = \frac{x_{ij} - \bar{x}}{\sigma_j} \quad (2.1)$$

onde \bar{x} representa a média do atributo e σ_j representa o desvio padrão.

Os algoritmos de AM geralmente possuem como entrada e saída valores numéricos, portanto é necessário converter os valores categóricos da base de dados para valores numéricos. A codificação one-hot é um processo que converte rótulos em vetores binários como mostrado na Figura 2.1. Uma vantagem dessa codificação é que não cria uma ”ordem” numérica nos dados. Essa ordem poderia interferir na indução do classificador, podendo dar maior importância para valores maiores, o que não faz sentido para variáveis categóricas.



Color	Red	Yellow	Green
Red	1	0	0
Red	1	0	0
Yellow	0	1	0
Green	0	0	1
Yellow	0	0	1

Figura 2.1: Codificação One-hot

2.1.3 K-Nearest Neighbors (KNN)

Um classificador bastante popular é o K-Nearest Neighbors(K-Vizinhos mais próximos). O KNN utiliza os próprios dados de treinamento como modelo de classificação. Para classificar uma instância de teste, procura-se entre os dados de treinamento os K mais próximos da instância de teste. Por fim, verifica-se qual a classe predominante desses K dados de treinamento, e instância de teste é classificada com essa mesma classe. A cada nova exemplo a ser classificado faz-se uma varredura nos dados de treinamento, o que provoca um grande esforço computacional

O princípio do classificador k-NN é a “regra dos vizinhos mais próximos”. A hipótese é que, dado um conjunto de exemplos distribuídos sobre o espaço de dados X , a “vizinhança” de um exemplo $x \in X$ estabelecida por uma função de distância apropriada tende a ser ocupada por exemplos que pertencem à mesma classes que x [6] , como ilustrado na Figura 2.2. Desse modo a informação fornecida pelos exemplos conhecidos que são mais similares a x .

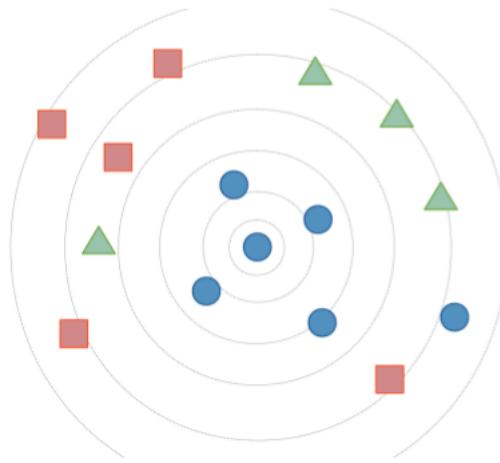


Figura 2.2: Princípio dos k-vizinhos mais próximos

Para encontrar os vizinhos mais próximos é necessário definir uma medida de similaridade entre dois exemplos. Uma medida de similaridade bastante popular é a *distância euclidiana*. Tal medida calcula a raiz quadrada da norma do vetor diferença entre os vetores x e y :

$$d(x, y) = \sum_{i=1}^K (x_i - y_i)^2 \quad (2.2)$$

2.2 Redes Neurais Artificiais

Redes Neurais Artificiais são modelos computacionais que buscam simular o processamento de informação pelo cérebro humano. Elas são compostas por unidades simples de processamento, os neurônios, que se unem por meio de conexões sinápticas [7]. Cada conexão, além de ser altamente especializada, é responsável pelo envio de sinais de um neurônio para outro. Segundo (Haykin 2009[8]), os neurônios e suas conexões podem ser implementados utilizando-se de componentes eletrônicos ou via simulação programada em computador.

2.2.1 Inspiração Biológica e Perceptron

Um bloco básico de uma rede neural tem algumas semelhanças com um neurônio biológico. O neurônio biológico é uma célula formada por três seções com funções específicas e complementares: *corpo*, *dendritos* e *axônio*. Os dendritos captam os estímulos recebidos em um determinado período de tempo e os transmitem ao corpo do neurônio, onde são processados. Quando tais estímulos atingirem determinado limite, o corpo da célula envia um novo impulso que se propaga pelo axônio e é transmitido às células vizinhas por meio de sinapses. Este processo pode se repetir em várias camadas de neurônios. Como resultado, a informação de entrada é processada, podendo levar o cérebro a comandar reações físicas. [9]. A figura 2.3 ilustra de forma simplificada as partes de um neurônio.

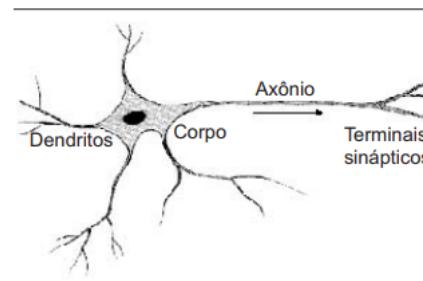


Figura 2.3: Representação simplificada de um neurônio

O modelo de um neurônio artificial é apresentado na Figura 3.4. Este modelo é composto por três elementos:

- Um conjunto de entradas (x_1, x_2, \dots, x_n) que são multiplicadas por um conjunto de pesos (p_1, p_2, \dots, p_n);
- Um somador (\sum) para acumular o sinais de entrada;

- Uma função de ativação que (φ) limita o intervalo permissível de amplitude do sinal de saída (y) a um valor fixo.

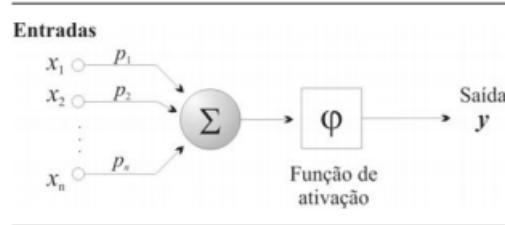


Figura 2.4: Modelo matemático de um neurônio

Esse modelo foi proposto por *McCulloch and Pitts* em 1943 [10] e é conhecido como *perceptron*. A função de ativação (φ) tem a seguinte definição:

$$\varphi = \begin{cases} 1 & \text{if } \sum_{i=1}^n p_i x_i^k \geq T, \\ 0 & \text{if } \sum_{i=1}^n p_i x_i^k < T \end{cases} \quad (2.3)$$

Os valores dos pesos podem ser positivos ou negativos e eles refletem se aquela conexão é inibitória ou excitatória. Um valor positivo ou negativo reflete a importância da respectiva entrada para o processamento. Frequentemente é adicionado um *viés* b na entrada da função de ativação. A forma geral do modelo é descrito como:

$$y(k) = \varphi\left(\sum_{i=1}^n p_i(k)x_i(k) + b(k)\right) \quad (2.4)$$

Os pesos sinápticos do Perceptron podem ser adaptados empregando um processo de aprendizado com um número finito de iterações. A aprendizagem é conduzida pela regra de correção de erro conhecida como algoritmo de convergência do Perceptron. Esse algoritmo visa encontrar um vetor de pesos w tal que as duas igualdades da função degrau sejam satisfeitas(*Lippmann*, 1987 [11])

2.2.2 Perceptron multicamadas

O Perceptron multicamadas(*Multi-Layer Perceptron - MLP*) é uma generalização da rede perceptron. Novas camadas são adicionadas o que possibilita a solução de problemas que não sejam linearmente separáveis. O vetor de entradas \mathbf{x} passa pela camada inicial,

cujos valores de saída são ligados a camada seguinte. Esse processo se repete até chegar na última camada.(Figura 2.5) Pode-se arranjar a rede em várias camadas, tornando-a profunda e capaz de aprender relações cada vez mais complexas.

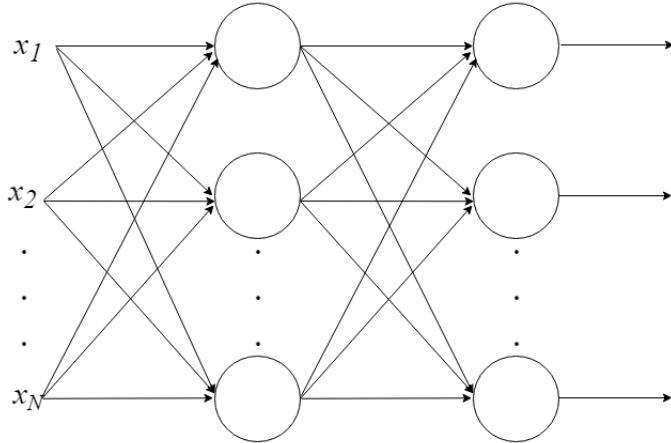


Figura 2.5: Multilayer Perceptron. Cada círculo representa um neurônio mostrado anteriormente

Em 1986, *Rumelhart, Hinton e Williams* [12] desenvolveram o algoritmo *backpropagation*, que utiliza o gradiente descendente para treinar uma MLP. Este método é composto pelas fases *forward* e *backward*. O objetivo do backpropagation é otimizar os pesos para que a rede neural possa aprender a mapear corretamente as entradas para as saídas. A primeira fase é a “propragação adiante” (forward), onde as entradas inseridas na rede se propagam entre as camadas, uma a uma, até a produção das respectivas saídas, portanto a função dessa fase é gerar uma resposta considerando as entradas e os respectivos pesos sinápticos, os quais permanecem inalterados.

Na fase backward é onde a aprendizagem dos pesos é realizada. Esse aprendizado se dá através da otimização (minimização) da função loss, a qual determina a qualidade da classificação do dado de entrada. Essa otimização é realizada através de um método chamado *Gradiente Descendente* que busca a minimização da função *loss* ao alterar os pesos na direção de maior declive do gradiente. O gradiente é calculado na última camada e então é retro-propagado para as camadas intermediárias anteriores que contribuem diretamente para a formação da saída. Cada elemento da camada intermediária recebe é responsável apenas por uma porção do gradiente total. Este processo se repete, camada por camada, até que cada elemento tenha a sua parcela de gradiente para o gradiente total. Baseado no gradiente, é feita uma alteração dos valores dos pesos e bias de modo que a rede aprenda os padrões do conjunto de treinamento.

2.2.3 Camada Softmax

Como explicado em *GoodFellow, Bengio, e Courville(2016 [13])*, a camada softmax é utilizada como um classificador na camada de saída e tem como objetivo representar a probabilidade de cada classe para cada valor de entrada.

Pela Equação 3.5 nota-se que os valores de saída da camada softmax estão entre 0 e 1, e que a soma de todas as saídas é igual a 1. Desta forma, cada neurônio de saída representa a probabilidade da entrada pertencer a uma determinada classe.

$$\text{softmax}(z_i) = \frac{\exp(z_i)}{\sum_j \exp(z_j)} \quad (2.5)$$

2.2.4 Hiperparâmetros de uma Rede Neural

A maioria dos algoritmos de AM envolvem ”hiperparâmetros”, que são variáveis definidas para o algoritmo antes do treinamento com o objetivo de otimizá-lo. Definir os valores de hiperparâmetros pode ser visto como uma seleção de modelos, ou seja, escolher qual modelo usar do conjunto hipotético de modelos possíveis. Em redes neurais os hiperparâmetros determinam a estrutura da rede (Ex: Número de neurônios em uma camada) ou como a rede será treinada(Ex: Taxa de aprendizado)

2.2.4.1 Mini-batches

O treinamento geralmente é realizado em *batches* também conhecidos como *mini-batches* que são subconjuntos do treino. As razões de utilizar batches e não o treino inteiro são: diminuição do tempo de treinamento; caso o conjunto de treino seja suficientemente grande, uma amostra desse conjunto pode representar de forma *boa* o conjunto original. A tamanho dessa amostra é um hiperparâmetro da rede, sendo geralmente aceito que o treinamento com *batches* maiores resulta numa melhor performance.

2.2.4.2 Taxa de aprendizado

A taxa de aprendizado é um parâmetro constante no intervalo de [0,1] que interfere na convergência do aprendizado. Ela determina o quanto “rápido” as atualizações dos pesos irão em direção ao gradiente. Se a taxa de aprendizado for muito pequena, o modelo convergirá muito lentamente; Se a taxa de aprendizado for muito grande, o modelo irá

divergir.

2.2.4.3 Função de Perda(Loss Function)

A função de perda compara a saída da rede para um exemplo de treinamento com o rótulo verdadeiro. Uma função de perda comum é o erro médio quadrático dado pela Equação 2.6

$$MSE = \frac{\sum_{i=1}^N (y_i - \hat{y}_i)^2}{N} \quad (2.6)$$

Quando a saída da rede neural está sendo tratada como uma distribuição de probabilidade (Ex: Camada softmax na saída) é comum usar a *entropia cruzada* que indica o grau de perda entre duas distribuições de probabilidades, uma sendo referente a predição e a outra referente ao esperado , definida por:

$$L = - \sum_i y_i \log(z_i) \quad (2.7)$$

2.2.4.4 Otimizadores

A atualização dos parâmetros da rede ocorre através dos otimizadores. Existem diversos métodos de otimização e alguns deles utilizam hiperparâmetros para calibrar a forma com que os pesos devem ser atualizados . Eles são variações do *gradiente descendente*. Dentre os mais utilizados se destacam:

- Stochastic Gradient Descent:
- Momentum
- RMSprop
- AdaDelta
- Adam

2.3 Séries Temporais

Série temporal é uma sequência de observações de um fenômeno ao longo do tempo. Geralmente essas medições são feitas em um intervalo de tempo regular. A ordem das

amostras é crucial, pois há um dependência entre os dados e uma alteração da ordem pode modificar o significado dos dados. Uma série temporal pode ser definida como:

$$X(t) = (x_1, x_2, \dots, x_n) \quad (2.8)$$

onde x_n representa uma observação no instante t , n o número de observações e $X(t)$ a função que descreve a série temporal em termos de t . Caso a série seja constituída de uma observação em cada instante de tempo ela é chamada de *univariada*. Caso a série foi obtida por uma coleta simultânea de dois ou mais fenômenos ela é chamada de *multivariada*.

As séries temporais estão em diversas áreas do conhecimento como Economia (preços diáários de ações, taxa mensal de desemprego, produção industrial), Medicina (electrocardiograma, eletroencefalograma), Epidemiologia (número mensal de novos casos de meningite), Meteorologia(precipitação pluviométrica, temperatura diária, velocidade do vento).

2.3.1 Aplicações de Séries Temporais

A análise de séries temporais tem atraído muitos pesquisadores em aprendizado de máquina ao redor do mundo. As principais tarefas envolvendo séries temporais na qual se utiliza aprendizagem de máquina são as seguintes:

- *Classificação*: cada série temporal representa uma classe distinta de objetos. Dada uma série temporal, o objetivo é descobrir qual é a classe de objetos ela representa;
- *Agrupamento*: Dado um conjunto de séries temporais, o objetivo é encontrar uma estrutura natural que permita distribuir as séries em grupos;
- *Detecção de Motivos*: também conhecido como detecção de *motifs*; o objetivo é encontrar uma ou mais subsequências que aparecem frequentemente na séries;
- *Detecção de Anomalias*: encontrar subsequências ou séries que são inesperadas em algum contexto.

Este trabalho está inserido na tarefa de classificações de séries temporais.

2.4 Classificação de Séries Temporais

2.5 Frequência Fundamental

A frequência fundamental f_0 é a menor frequência de uma vibração, ou também, a primeira frequência de uma série harmônica. Uma série harmônica é o conjunto de ondas composto pela f_0 e suas múltiplas inteiras. Normalmente, todo estímulo sonoro é formado por uma série harmônica. A frequência fundamental está diretamente ligada à altura de um som e, por meio dela, distingue-se grave e agudo.

2.6 Pitch

O *pitch* é um conceito musical relacionado à frequência fundamental, entretanto voltado para a percepção humana do som. Ele é definido em [14] como “*o atributo pelo qual a audição pode ordenar os sons em uma escala de grave para agudo, dependendo principalmente do conteúdo de frequência do estímulo sonoro, mas também da forma de onda desse estímulo*”. Embora pareça confusa, esta definição abrange a relação do *pitch* com a altura e com o timbre de um som, necessário para que o conceito seja válido em todos os casos, tanto em aplicações musicais como em aplicações de reconhecimento de fala. Em [15], fica evidente que o *pitch* de um som refere-se a frequência em que uma onda senoidal é reconhecida por um ouvido humano como correspondente ao som em questão.

2.7 Detecção Automática de f_0

Embora muitos trabalhos tratem frequência fundamental e *pitch* como sinônimos, [16] explica de modo claro que existem diferenças consideráveis entre os dois conceitos, de modo que um sistema de detecção de *pitch* objetiva resultados diferentes de um sistema de detecção de f_0 . Como exemplo disso, pode-se observar que um sinal com frequências mais puras e poucas informações de timbre são ideais para a detecção de f_0 , entretanto, a detecção de *pitch* depende ainda da intensidade, duração e timbre.

Muitas soluções e métodos de detecção automática de f_0 já foram desenvolvidas e apresentadas, obtendo resultados positivos. Entretanto, [16] também afirma que é difícil que uma mesma solução de detecção de f_0 tenha bom desempenho tanto aplicações de música

quanto em aplicações de fala. Deste modo, primeiro torna-se necessário definir o tipo de aplicação do sistema de detecção para, só então, desenvolver e avaliar o desempenho desse sistema no contexto escolhido.

2.8 Digitalização de um Sinal de Som

Para a digitalização de um sinal sonoro, são necessários a amostragem, a quantização e a codificação.

2.8.1 Amostragem

A amostragem é o processo de conversão de um sinal contínuo no tempo em um sinal discreto. Se a discretização no tempo ocorrer em uma frequência de amostragem (f_s) muito baixa, as altas frequências não serão amostradas, gerando o efeito *aliasing*, que é a distorção de um sinal causado pela perda das informações de alta frequência.

Segundo o Teorema da Amostragem[17], dado um sinal contínuo amostrado com frequência f_s , se $f_s > 2f_M$, onde f_M é a frequência máxima do sinal, então será possível recuperar toda a informação do sinal a partir dos valores amostrados. O intervalo entre a coleta de cada amostra é denominado *período de amostragem*, representado pelo símbolo T_s . Pela relação de período e frequência, temos que $T_s = \frac{1}{f_s}$.

A Figura 2.6a demonstra graficamente a amostragem de um sinal.

2.8.2 Quantização

A quantização, por sua vez, consiste em selecionar um conjunto de valores finitos, espaçados linearmente ou não, para os quais as amostras terão seus valores de amplitude arredondados. A diferença entre o valor real e o valor quantizado da amostra é chamado de erro de quantização. Na digitalização de sinais, deve-se adotar um conjunto de quantização com tamanho suficientemente grande para que as perdas de informação por erros nesse procedimento não impeçam uma posterior recuperação do sinal.

A Figura 2.6b demonstra graficamente a quantização de um sinal.

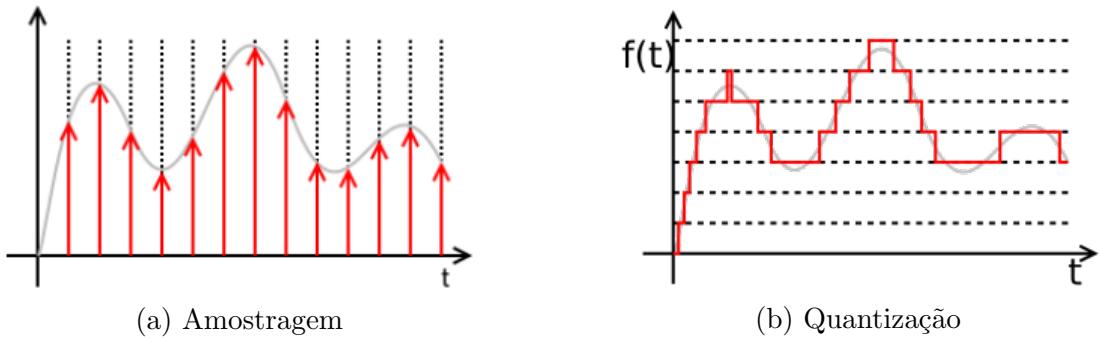


Figura 2.6: Amostragem e Quantização

2.8.3 Codificação

Por fim, a codificação é o procedimento no qual se define o formato de armazenamento das informações do sinal. Após esse passo, essa representação digital passa a ser denominada áudio. Existem diferentes codificadores para áudio, cada um deles com suas particularidades. Alguns codificadores usam técnicas de compressão, que podem gerar pequenas perdas de informação, enquanto outros buscam preservar a integridade da representação, sendo, consequentemente, maiores em tamanho de arquivo. O Matlab® trabalha nativamente com 8 tipos de codificadores de áudio: *WAVE*, *OGG*, *FLAC*, *AU*, *AIFF*, *AIFC*, *MP3* e *MPEG-4 AAC*[18].

2.9 Notas Musicais

Sob a ótica da música, os *pitches* são representados pelas notas musicais. A distância entre a frequência de uma nota e metade ou dobro dessa mesma frequência é chamada de oitava. O menor intervalo entre duas notas é chamado de “semitom”, enquanto um “tom” são dois semitonos. A Tabela 2.2 exibe todas as notas da 4^a oitava. A nota Lá (A) nessa oitava equivale a frequência de 440Hz, isso significa que essa mesma nota na 3^a oitava tem metade dessa frequência, ou seja, 220Hz. Por padrão, a nota lá da 4^a oitava, ou A4, é usada como base para a definição de todas as outras notas. A maioria dos músicos utiliza A4 em 440Hz, mas há casos em que são adotadas frequências diferentes como 436Hz e 444Hz.

Nota	Abreviação	Frequência (em Hz)
Dó	C	261.6
Dó Sustenido ou Ré Bemol	C# ou Db	277.2
Ré	D	293.7
Ré Sustenido ou Mi Bemol	D# ou Eb	311.1
Mi	E	329.6
Fá	F	349.2
Fá Sustenido ou Sol Bemol	F# ou Gb	370.0
Sol	G	392.0
Sol Sustenido ou Lá Bemol	G# ou Ab	415.3
Lá	A	440.0
Lá Sustenido ou Si Bemol	A# ou Bb	466.2
Si	B	439.9

Tabela 2.2: Notas musicais da 4^a oitava e suas frequências

2.10 Ressonância

A ressonância é um fenômeno físico em que uma vibração dá origem a outras vibrações com maior amplitude em um conjunto de frequências específicas. Essas frequências geradas são chamadas de frequências ressonantes. Os instrumentos musicais também podem apresentar esse fenômeno, como, por exemplo, alguns instrumentos de corda (violão, piano acústico, violino, entre outros) que utilizam uma caixa de ressonância para amplificar o som gerado pela vibração de suas cordas.

2.11 Domínios

Em PDS, os sinais são representados em dois diferentes domínios: tempo e frequência. No domínio do tempo, ou domínio temporal, um sinal é representado como amostragens de sua amplitude em intervalos de tempo T . Já no domínio da frequência, ou domínio espectral, um sinal é representado pela amplitude das oscilações que ocorrem em cada frequência. Desse modo, ter o mesmo sinal em diferentes domínios é tê-lo em diferentes perspectivas.

Os domínios do tempo e da frequência são permutáveis entre si, ou seja, dado um sinal $x(t)$ temporal, é possível transportar esse sinal para a representação espectral na forma $X(j\omega)$ sem que haja perda da informação. Do mesmo modo, é possível fazer o caminho inverso e retornar um sinal na frequência para a sua forma temporal inicial. Para transitar entre os domínios, utiliza-se as transformadas de Fourier.

2.12 Transformada de Fourier

A transformada de Fourier é uma ferramenta de decomposição de uma função em partes de base senoidal, ou seja, o sinal temporal passa a ser expresso em frequências. Inicialmente proposta pelo francês Jean Baptiste Joseph Fourier[19], as séries de Fourier deram origem a diferentes versões da transformada de Fourier, cada uma voltada para um tipo de sinal de origem.

Para sinais discretos no tempo utiliza-se a Transformada de Fourier em Tempo Discreto (DTFT, do inglês *Discrete-Time Fourier Transform*), dada pela equação (2.9):

$$X(\omega) = \sum_{n=-\infty}^{+\infty} x(n)e^{-j\omega n} \quad (2.9)$$

onde ω é a frequência angular, em radianos por segundos, e n é a posição da enésima amostra do sinal.

A DTFT é aplicável a sinais discretos no tempo, entretanto, é computacionalmente inviável, visto que sua saída é contínua na frequência. Deste modo, torna-se necessário discretizar essa saída para valores específicos de ω . Para este fim, desenvolveu-se a Transformada Discreta de Fourier (DFT, do inglês *Discrete Fourier Transform*).

2.13 Transformada Discreta de Fourier

A DFT é dada pela equação (2.10):

$$X(k) = \sum_{n=0}^{N-1} x(n)e^{-j2\pi \frac{kn}{N}} \quad (2.10)$$

onde $x(n)$ é um sinal finito de tamanho N , e k assume valores inteiros de 0 a $N-1$.

Nesse contexto, $X(k)$, DFT do sinal de entrada, é uma saída periódica. Essa discretização da saída aproxima a transformação de Fourier para a viabilidade computacional. Entretanto, executar uma soma do tamanho do sinal sob análise para cada valor específico da DFT, ainda demanda um grande trabalho computacional, pois à medida em que aumenta-se o sinal de entrada, também aumenta-se quadraticamente a quantidade de operações. Por notação assintótica, pode-se afirmar que a DFT exige um esforço computacional $\Omega(N^2)$.

Por esse motivo, tornou-se necessário obter um algoritmo computacionalmente mais

eficiente para calcular a DFT: A Transformada Rápida de Fourier (FFT, do inglês *Fast Fourier Transform*).

2.14 Transformada Rápida de Fourier

Designa-se como FFT uma série de algoritmos propostos que visam realizar a DFT de maneira mais eficiente. A Transformada Rápida de Fourier reduz as operações computacionais de $\Omega(N^2)$ para $\Omega(N \log_2 N)$, tornando viável tal implementação. Para o presente projeto, utilizou-se a função nativa *fft* do MATLAB[®], cujo a documentação fornece também casos básicos de uso[20]. Essa implementação baseia-se na equação:

$$X(k) = \sum_{j=1}^N x(j)W(N, j, k) \quad (2.11)$$

onde $x(n)$ é o sinal de entrada, N o tamanho do sinal, e W é dado como:

$$W(N, j, k) = \cos\left(\frac{2\pi i(j, k)}{N}\right) + j \sin\left(\frac{2\pi i(j, k)}{N}\right) \quad (2.12)$$

onde,

$$i(j, k) = (j - 1)(k - 1) \quad (2.13)$$

O tempo de execução dessa função depende do tamanho da entrada. Sinais com comprimentos que sejam potências de 2 são mais rápidos para o processamento, seguidos pelos sinais cujo o comprimento contém apenas pequenos fatores primos. As entradas que demoram mais para serem processadas são aquelas em que seu comprimento contém grandes fatores primos.

O tamanho e a precisão na frequência da saída de um algoritmo FFT também está diretamente relacionado ao tamanho do sinal de entrada. Em casos onde o sinal a ser transformado não possui um tamanho ideal para a precisão no domínio da frequência e tempo de execução desejados, pode-se adotar a técnica de *padding*, acrescentando zeros ao final do sinal.

2.15 Transformada de Fourier de Tempo Curto

A Transformada de Fourier de Tempo Curto (STFT, do inglês *Short-time Fourier Transform*), é uma técnica de aplicação da transformada de Fourier utilizada para a análise de sinais não periódicos. Ela consiste na utilização de uma janela de tempo curto, que é colocada sobre o início do sinal. O segmento interno da janela é considerado como periódico e é submetido à transformada de Fourier. Após, desloca-se a janela de tempo curto e repete-se o processo até que a janela alcance o fim do sinal. Desse modo, é possível entender como o espectro de um sinal se comporta a cada instante de tempo.

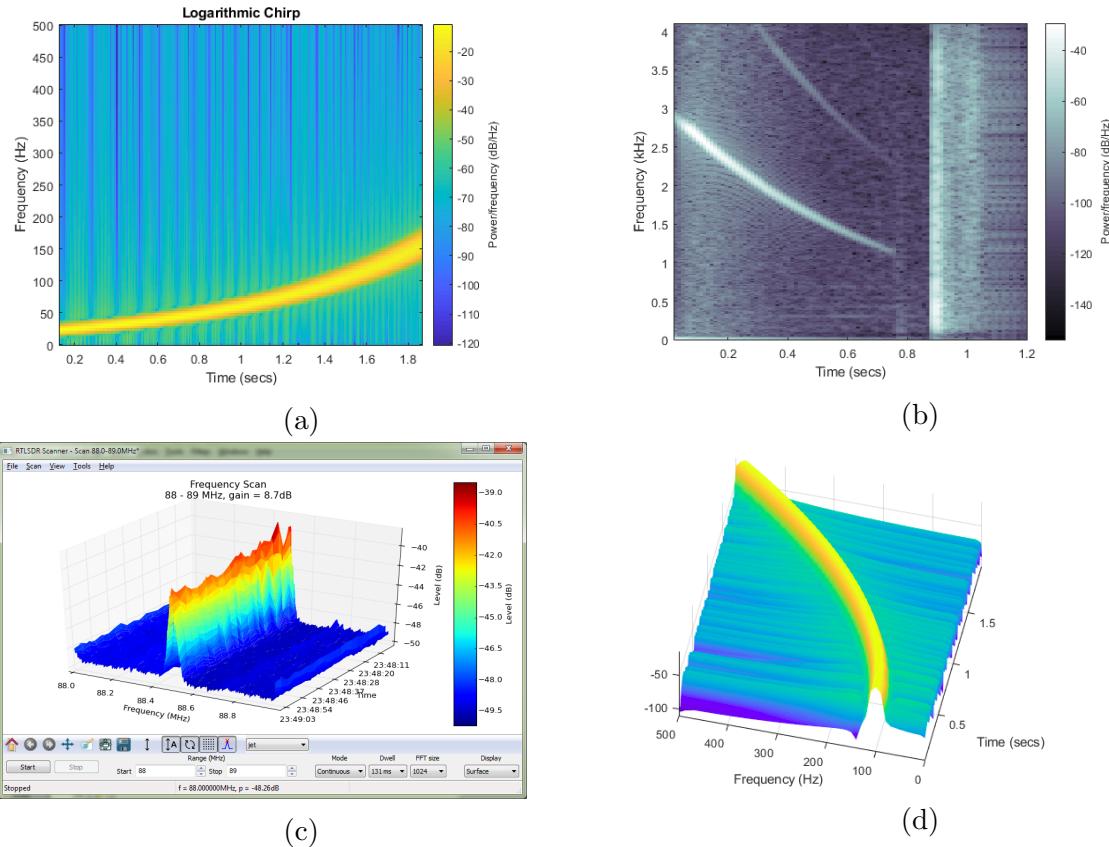
A STFT discreta pode ser implementada utilizando a FFT. O comprimento das janelas de tempo curto precisa ser definido conforme o tipo de aplicação desejada, pois deve ser compatível com o tamanho do fenômeno que deseja-se observar. Outro fator importante a ser definido é a forma de deslocamento da janela. Quando o tamanho da janela é igual ao valor de deslocamento, todas as amostras são analisadas uma única vez. Se o tamanho da janela for menor que o deslocamento, algumas amostras não serão analisadas, e se o deslocamento for maior, cada amostra será analisada em mais de uma posição da janela.

Em PDS, mais precisamente em sistemas de reconhecimento e detecção por áudio, essa técnica se torna imprescindível, pois permite uma visão clara da informação contida no sinal.

2.16 Espectrograma

O espectrograma é uma forma de representar sinais. Ele mostra como o sinal evolui na frequência conforme o tempo. Existem diferentes formatos de gráfico para um espectrograma. A Figura 2.7 demonstra dois desses formatos: (a) e (b) Gráfico bidimensional com escala de cores, onde o eixo do tempo está na horizontal, o eixo da frequência na vertical e a amplitude é representada pela escala de cores; (c) e (d) Gráfico tridimensional clássico $z = f(x, y)$, onde o tempo está no eixo x, y é a frequência e z é a amplitude;

O método mais usual para a geração do espectrograma de um sinal é por meio da STFT. Aplicando a técnica, calcula-se a FFT de cada janela do sinal de entrada. Como a saída da FFT pode conter números complexos, computa-se o módulo dos valores de saída da transformada, obtendo-se vários planos de frequências e amplitudes. Por fim, os planos de espectro de cada janela de tempo são colocados lado a lado, formando assim o

Figura 2.7: Espectrogramas no MATLAB[®]

espectrograma completo.

A equação (2.14) descreve esse processo com o uso da STFT:

$$\text{spectrogram}(x(n), \Delta) = |\text{STFT}(x(n), \Delta)|^2 \quad (2.14)$$

onde $x(n)$ é o sinal sob análise e Δ é o tamanho da janela para a STFT.

O MATLAB[®] fornece o método *spectrogram* para obter, por meio da STFT, o espectrograma de um sinal[21]. Este método utiliza por padrão o gráfico bidimensional com escala de cores.

Capítulo 3

Sistema Proposto

Neste capítulo é abordado o sistema de detecção de f_0 proposto neste projeto, com a descrição da metodologia e dos materiais utilizados no desenvolvimento.

3.1 Visão geral

O sistema proposto é composto pelo bloco de detecção de f_0 envolto no ambiente de experimentação, desenvolvido com a finalidade de facilitar a análise de um arquivo de áudio, de modo que tal tarefa possa ser executada apenas por meio de uma interface gráfica. A Figura 3.1 mostra uma visão geral do projeto. Dado um arquivo de áudio como entrada, o bloco de pré-processamento é responsável pelo preparo da mídia para a análise, definindo o tamanho da janela a ser analisada e executando qualquer processamento que seja necessário para que a detecção seja feita. Após isso, o áudio é submetido ao bloco de detecção de f_0 , que fará a detecção para cada deslocamento da janela de tempo. Por fim, a saída do bloco de detecção de f_0 é plotada na janela principal da aplicação, de modo a ser comparada visualmente com os registros originais de notas e tempo do áudio.

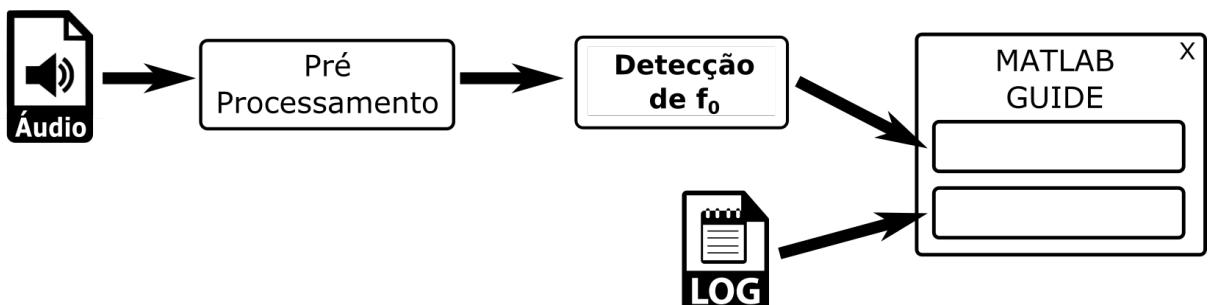


Figura 3.1: Estrutura do sistema desenvolvido

O sistema foi desenvolvido em linguagem MATLAB[®], utilizando a ferramenta de interface gráfica GUIDE para interação com o usuário, fornecendo a lista de áudios disponíveis na base de dados para a execução do algoritmo de detecção e retornando, ao término da execução, os gráficos necessários para a análise visual do áudio. Os registros originais de notas e tempo dos áudios estão em formato texto, em arquivos com extensão *.log*. Os áudios de entrada devem estar em formato reconhecido pelo MATLAB[®]. Para a execução de um experimento, um usuário precisará fornecer o áudio, seu respectivo arquivo de registros de tempo de notas, a velocidade da música, em batidas por minutos, e a referência de menor nota para a detecção, que é um valor referente a menor nota executada no áudio a ser analizado.

3.2 Bloco de Pré-Processamento

O bloco de pré-processamento é responsável pelo preparo do áudio para a detecção de f_0 . Esse bloco recebe como entrada o caminho para o arquivo de áudio, seu valor de batidas por minutos (bpm) e um número real positivo equivalente a nota de menor tempo de duração desejada na detecção. O diagrama da figura 3.2 demonstra o funcionamento dessa parte inicial do sistema. Primeiro, lê-se o arquivo de áudio a partir do caminho fornecido. Para isso, a função *audioread*, nativa do MATLAB[®], foi utilizada e retorna as amostras do áudio e sua f_s . Após, obtém-se as informações de metadados presentes no arquivo. A função *audioinfo*, também nativa da linguagem utilizada, busca as informações do áudio como duração e número de canais.

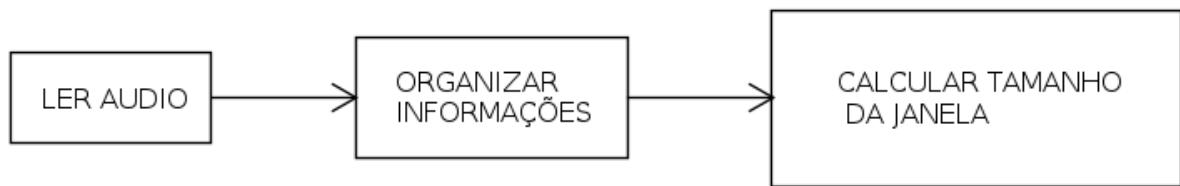


Figura 3.2: Diagrama do bloco de pré-processamento

Após a leitura do áudio e de suas informações, o sistema determina o tamanho da janela para análise de tempo curto. Tal comprimento de janela não pode ser muito pequeno, ao ponto de não ter amostras suficientes para análise, e também não pode ser tão grande, para não misturar em uma mesma posição de janela um intervalo com duas f_0 diferentes. Deste modo, optou-se por um tamanho em que a duração do intervalo não

fosse maior do que o tempo de referência de nota mínima. Utilizando essa referência, é possível garantir que o sistema não fará análise de uma janela com duas f_0 's. Essa entrada é dada em função da velocidade de execução da música (em bpm), sendo que uma nota mínima de referência 1 equivale a uma nota por batida. Para definir o tamanho da janela em quantidade de amostras, foi utilizado:

1. Converter o valor de bpm para batidas por segundos, multiplicando por 60;
2. Dividir o valor de duração da menor nota por dois, de modo a analisar 2 vezes a menor nota;
3. Multiplicar batidas por segundos por metade da menor nota e a frequência de amostragem, obtendo a quantidade de amostras em uma janela

Os pontos acima são expressos na equação (3.1):

$$L = \frac{60}{bpm} \frac{t_{ref}}{2} F_s \quad (3.1)$$

onde t_{ref} é o tempo de referência de duração da menor nota a ser detectada e bpm é a velocidade de execução da música no áudio, em batidas por minutos.

Simplificando, obtém-se a equação (3.2):

$$L = \frac{30t_{ref}F_s}{bpm} \quad (3.2)$$

Em linha de código, foi necessário ainda arredondar o valor de L , visto que posições de amostras devem ser valores inteiros. Para isso, utilizou-se a função nativa *round* do MATLAB® .

Com o tamanho da janela definido, o áudio segue para o bloco de detecção de f_0 .

3.3 Bloco de Detecção de f_0

A detecção de f_0 é feita através da STFT, utilizando a FFT. O método de detecção consiste em transportar a janela sob análise do áudio para o domínio da frequência, de modo que seja possível localizar a frequência de maior amplitude nessa janela. Essa frequência é registrada como a f_0 do respectivo intervalo. O tamanho do deslocamento da janela é igual ao comprimento da mesma, de modo que não se repitam amostras entre um

deslocamento e outro. Outros métodos de deslocamento podem ser adotados de acordo com o tipo de problema que deseja-se solucionar, por exemplo, para a detecção exata do tempo de início das notas, um deslocamento menor que o tamanho da janela permitiria verificar com mais precisão o ponto de início de cada nota. O algoritmo desse bloco pode ser dividido em três partes: A pré-análise, o deslocamento da janela e a análise de detecção de f_0 . Após esses passos, o bloco retorna a f_0 detectada.

A parte de pré-análise consiste basicamente na inicialização das variáveis que serão utilizadas no momento da análise. A função *zeros*, nativa do MATLAB[®], é utilizada para gerar vetores de zeros do tamanho esperado para a saída da detecção. Para o tamanho de janela igual ao deslocamento, o comprimento da saída pode ser calculado pela divisão do tamanho total do áudio pelo comprimento da janela. Durante a execução do código, o vetor de saída, inicialmente preenchido com zeros, receberá cada uma das detecções, e será retornado finalmente com todas as frequências fundamentais obtidas em cada posição da janela.

Na parte de deslocamento da janela, um laço é utilizado, conforme o diagrama da Figura 3.3. A variável *CONTADOR* é utilizada para marcar o posicionamento das saídas das detecções, sendo incrementado a cada execução do laço, enquanto o iterador i do laço é usado para marcar a posição da janela sobre o áudio. O tamanho do deslocamento da janela é definido como a variável *windowSize*, que é o comprimento dessa janela, de modo a garantir que cada amostra será analizada apenas uma vez. A parte do áudio no intervalo do deslocamento da janela é retirado para a variável *window*, esse será o sinal analizado. A análise é invocada com *DETECTAR_F0(window)*, que é a função desenvolvida para a detecção das frequências fundamentais, retornando para cada janela uma f_0 detectada. Por fim, essa frequência retornada é armazenada no vetor de saída, conforme a posição guardada na variável *CONTADOR*, e se repete todo o procedimento até que a janela tenha se deslocado por todo o áudio.

A terceira e última parte do bloco de detecção de f_0 é o método responsável pela execução da FFT e identificação da fundamental. A função foi desenvolvida tendo como base as aplicações exemplo da *fft* do MATLAB[®], de modo que todo o tratamento pós-transformada é feito conforme as recomendações obtidas na documentação. A implementação adotada é capaz de fazer a detecção da f_0 em um tempo consideravelmente rápido, de modo que a janela possa ser deslocada rapidamente e, consequentemente, fi-

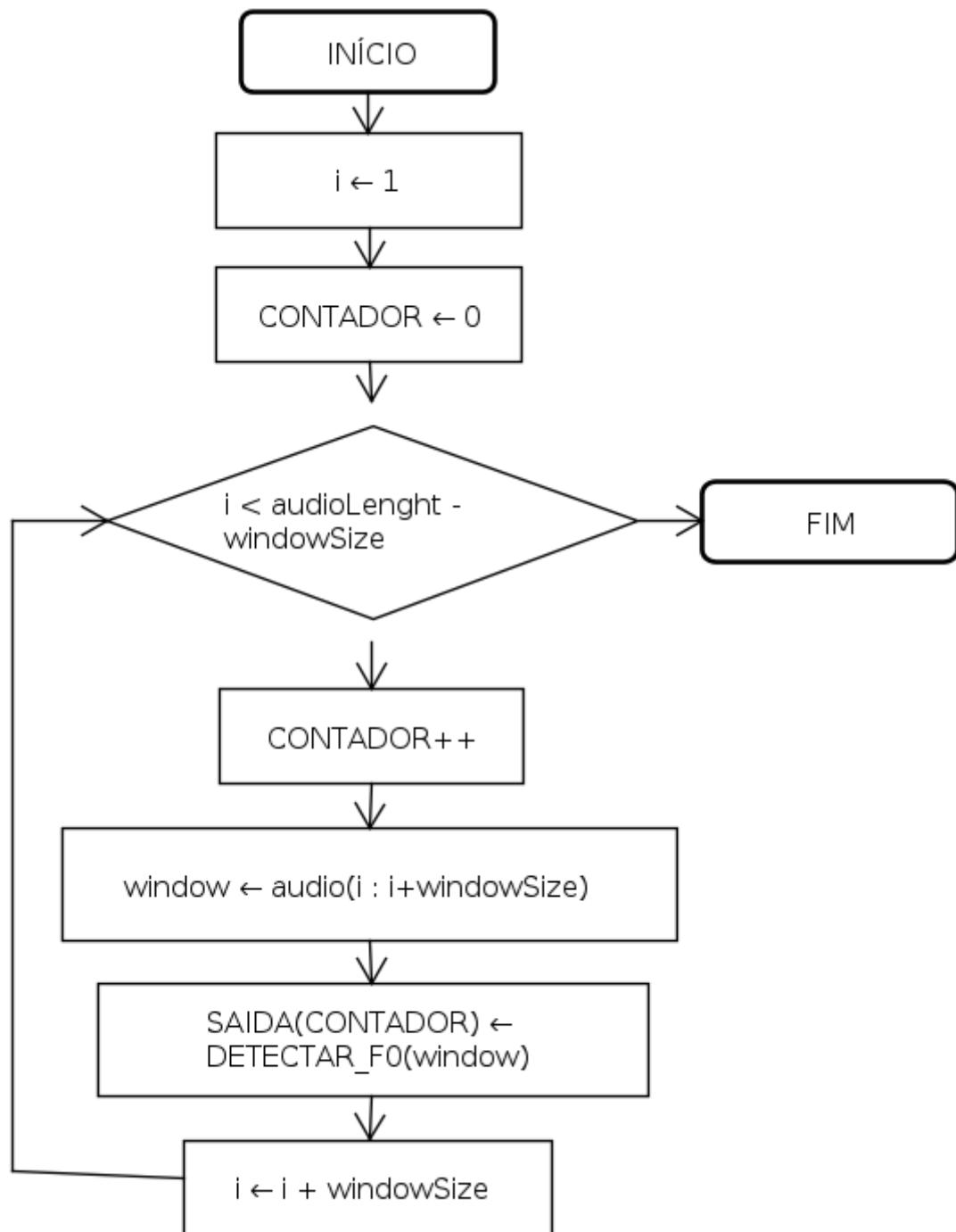


Figura 3.3: Diagrama do algorítmo de deslocamento da janela

nalizar a análise do áudio em poucos segundos. O redimensionamento da quantidade de amostras a serem transformadas também cooperou para a boa performance do código.

A função, exibida no Algorítmo 1, recebe como entrada a janela de áudio, na variável $x(t)$, e a frequência de amostragem. Sua saída é a f_0 detectada. Da linha 2 à 5, o sinal sob análise é preenchido com *padding* de zeros de modo a satisfazer o tamanho de 2^{16} amostras. Como visto na fundamentação teórica, este passo é importante para a

performance do algoritmo. Na linha 6, a FFT do MATLAB[®] é executada, transportando a janela de áudio para o domínio da frequência. É importante ressaltar que a saída da FFT tem o mesmo comprimento que a entrada. Uma das características da FFT está em que sua saída é multiplicada pela quantidade de amostras. Desse modo, na linha 7, foi necessário dividir pelo comprimento da janela ao fim da transformação. Na mesma linha também calcula-se o módulo do sinal complexo. Na linha 8, devido ao espelhamento na frequência, retira-se metade do vetor do espectro e, na linha 9, dobram-se as amplitudes como tratamento padrão para os efeitos da transformação pela FFT. Após todos esses passos, as linhas de 11 à 13 identificam a frequência fundamental da janela sob análise buscando a frequência com amplitude máxima do vetor do espectro.

Algorithm 1 Função de detecção de f_0

```

1: função DETECTAR_F0( $x(t)$ ,  $f_s$ )
2:    $\Delta = COMPRIMENTO(sinal)$ 
3:    $\Delta_{ideal} = 2^{16};$ 
4:    $x(t) = CONCATENA(x(t), zeros(\Delta_{ideal} - \Delta))$ 
5:    $\Delta = \Delta_{ideal}$ 
6:    $X(w) = FFT(x(t))$ 
7:    $P2 = ABSOLUTO(X(w)/\Delta)$ 
8:    $P1 = P2(1 : \frac{\Delta}{2} + 1)$ 
9:    $P1(2 : fim - 1) = 2 * P1(2 : fim - 1)$ 
10:   $f = f_s * (0 : \frac{\Delta}{2})/\Delta$ 
11:   $A_{max} = MAX(P1(:))$ 
12:   $i_{max} = POSIÇÃO(A_{max}, P1)$ 
13:   $f_0 = f(i_{max})$ 
14:  devolve  $f_0$ 
15: fim função

```

Conforme visto no diagrama da Figura 3.3, o retorno da função de detecção de f_0 é armazenado em um vetor de saída do bloco, que será fornecido à interface GUIDE para a plotagem do gráfico dos resultados da detecção em cada deslocamento da janela.

3.4 Interface GUIDE

Com o propósito de facilitar o uso do sistema, foi desenvolvida, através da ferramenta GUIDE[22] do MATLAB[®] , uma interface gráfica para o usuário, que é exibida na Figura 3.4. Na janela, é disponibilizado ao usuário um menu *popup* com a lista dos áudios

disponíveis no diretório do projeto e um botão *PLAY*. Ao selecionar um arquivo e executar, a interface inicia o algoritmo de detecção de pitch com os parâmetros necessários para a análise. Janelas desenvolvidas em GUIDE utilizam *callbacks* para seus componentes, de modo que a chamada para a detecção está na *callback* do botão. A interface foi desenvolvida com o objetivo de ser utilizada para testes de algoritmos de detecção de f_0 e *pitches*, de modo que é possível incorporar outros algoritmos sempre que necessário, e de maneira rápida.

Ao término da detecção, são exibidas na interface 4 gráficos: (i) O sinal de áudio no tempo, que permite ter noção da localização das notas e de cada trecho de uma música, (ii) o espectrograma do áudio, que permite uma localização visual das frequências fundamentais, (iii) as saídas do sistema de detecção de pitch no tempo, saída a ser analisada e comparada com o espectrograma e os registros, e (iv) as saídas do sistema de detecção de pitch no tempo sobrepostas nas saídas esperadas, obtidas do arquivo de registro original de notas e tempos do áudio. O último gráfico permite que se faça uma avaliação visual rápida sobre as detecções, sendo possível identificar os trechos onde o sistema encontrou maiores dificuldades de detectar a f_0 .

O projeto foi organizado de modo que a aplicação é lançada a partir do arquivo GUIDE. Demais funções estão disponíveis no mesmo diretório, em arquivos com extensão *.m*, sendo chamadas pela interface quando necessário. Os arquivos da base de dados (Áudios e registros de logs) são armazenados em subdiretórios específicos para cada tipo: Os áudios ficam no subdiretório *samples*, enquanto os registros ficam no subdiretório *logs*. O menu *popup* com a lista de áudios da base de dados é criado por meio da *callback* de criação do próprio menu, que é chamada pela interface logo na inicialização. Para o preenchimento da lista desse menu, obtém-se a lista de áudios do subdiretório “*samples*” por meio da função “*getFiles*”, que mapeia o diretório e retorna uma lista com todos os arquivos internos no subdiretório.

Os gráficos foram gerados de modo nativo pelo matlab, através da função *plot*, com exceção do espectrograma que possui função específica para isso. Após a geração dos gráficos, todos foram setados na mesma escala, de modo a exibir o áudio completo e as frequências máximas detectadas. Verificou-se que, em alguns casos, o espectrograma não aparece por completo, devido ao seu tamanho limitado. Para melhor visualização de todos os gráficos, também é possível plota-los de maneira isolada da janela, através do comando

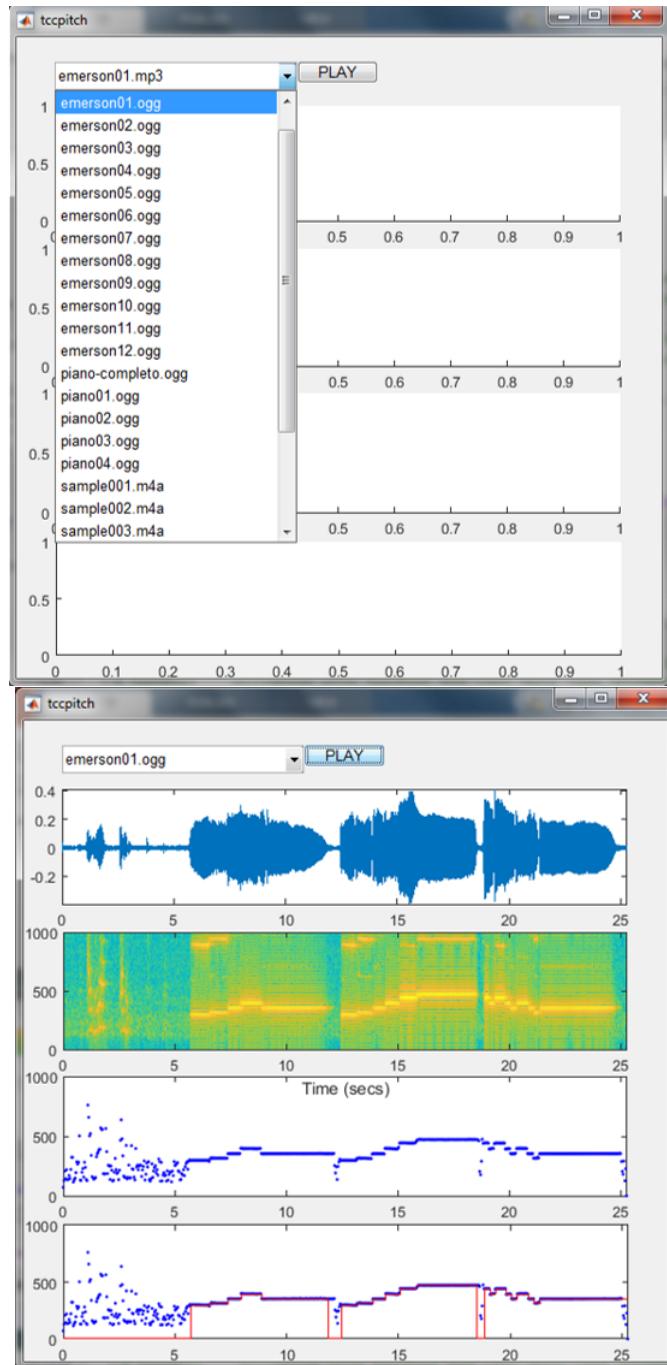


Figura 3.4: Interface GUIDE

figure, nativo do MATLAB[®].

3.5 Construção da Base de Dados

A base de dados construída para este projeto consiste em dois diretórios, onde um contém os arquivos de áudio para experimentos e o outro contém os registros de tempos de notas para os áudios. Foram feitas gravações com os instrumentos sax alto, clarinete,

piano eletrônico e violão acústico. Ao término do projeto, a base foi disponibilizada de modo público por meio de um repositório GIT[23].

As gravações dos áudios da base de dados foram feitas com um *smartphone*. Foi utilizado a aplicação de gravador do sistema Android®, que já conta com uma filtragem básica. As gravações foram realizadas em ambientes de pouco ruído, de modo a se obter um áudio com uma qualidade mínima necessária para a identificação das notas. Cada instrumento foi gravado pelo menos 3 vezes, executando 2 músicas em diferentes velocidades e, por fim, executando uma escala maior com velocidade de 2 tempos para cada nota. O violão e o piano executaram uma escala maior em Dó (C), enquanto o sax alto executou em Mi Bemol (Eb), e o clarinete em Si Bemol(Bb). Isto ocorreu devido às próprias características desses instrumentos, que possuem afinação em tons diferentes.

Tabela 3.1: Lista de Gravações da Base de Dados

INSTRUMENTO	ÁUDIO
CLARINETE	Escala - Bb Maior
	Música - <i>Agnus Dei</i>
	Música - Super Mário Brós
SAX ALTO	Escala - Eb Maior
	Música - Deus Cuida de Mim
	Música - <i>Summertime</i>
PIANO ELETRÔNICO	Escala - C Maior
	Música - Eu Navegarei
	Música - <i>Skyfall</i>
VIOLÃO	Escala - C Maior
	Música - Nona Sinfonia
	Música - Ao Deus de Abraão

Após as gravações, foram selecionados 2 áudios de música e o áudio da escala maior de cada instrumento, os quais foram inseridos no subdiretório “samples”, onde reunem-se todos os áudios da base. Todas as músicas gravadas para este projeto foram escolhidas pelos próprios instrumentistas, de modo a popular a base de dados com melodias mais comuns entre eles. Estipulou-se apenas 3 critérios:

- Tocar apenas a melodia, sem acompanhamento;
- Tocar apenas uma nota por vez, sem notas simultâneas;
- Duração entre 10 e 25 segundos;

A Tabela 3.1 lista as músicas que foram gravadas para a construção da base de dados.

Foi realizado um mapeamento manual dos áudios com o auxílio do software Audacity[24], de onde obteve-se informações precisas de tempo e duração de cada nota. Os registros ficam armazenados no subdiretório “*logs*”, e possuem a mesma nomenclatura do respectivo áudio que registram, acrescentados da extensão “*.log*”. Os registros são feitos em formato texto, cada nota soada por linha. Cada linha possui a sigla da respectiva nota, sua frequência fundamental e seu tempo inicial, em segundos, separados por tabulação.

Um exemplo do registro é mostrado a seguir:

1	XX	0	0
2	D4	293.66	5.725
3	D#4	311.13	6.56
4	F4	349.23	7.4

No exemplo, pode-se identificar que a primeira nota, um ré na 4^a oitava, é soada no tempo 5.725 segundos. A nota seguinte inicia em 6.56 segundos, um ré sustenido, e por fim, um fá se inicia em 7.4 segundos.

Capítulo 4

Experimentos

Para avaliar o sistema proposto, organizou-se uma bateria de experimentos utilizando a base de dados construída. Os resultados obtidos durante esses experimentos serão comparados com as informações disponíveis a fim de analisar a eficácia do algoritmo de detecção de f_0 . Neste capítulo, serão relatadas as métricas utilizadas, os objetivos propostos para a experimentação, os resultados obtidos para cada um dos áudios da base de dados e as discussões acerca desses resultados.

4.1 Métricas

Os experimentos consistiram, de modo geral, em submeter os áudios ao sistema desenvolvido, e comparar os resultados obtidos com os arquivos de registros criados manualmente. Para realizar um experimento, deve-se primeiro abrir a interface GUIDE e selecionar, no menu *popup*, o áudio que se deseja analisar. Ao clicar para iniciar o experimento, a interface submete o áudio ao sistema e, com o resultado, plota 4 diferentes gráficos: (i) Gráfico do áudio no tempo, (ii) Espectrograma do áudio, (iii) Saídas do sistema - f_0 detectada - no tempo, e (iv) Saídas esperadas - conforme registro manual - sobreposta nas saídas do sistema, ou seja, f_0 real sobreposta na f_0 detectada. A principal métrica adotada para a avaliação dos experimentos é a comparação visual no último gráfico, que sobrepõe o valor esperado com o valor detectado. Também é possível analisar o desempenho do algoritmo com base no espectrograma. Para um funcionamento ideal do algoritmo, espera-se que os valores detectados estejam sempre juntos com os valores reais.

Outra métrica utilizada para este projeto é o percentual de frequências fundamentais detectadas com sucesso. Para essa métrica, momentos de pausas e silêncios iniciais e finais deveriam ser ignorados, sendo considerado apenas os momentos em que uma nota musical realmente estivesse soando. Outro problema são os arquivos de registro, que não fornecem uma f_0 precisa, sendo apenas uma referência para comparação visual. Devido a essa complexidade para se obter a métrica de forma automática, adotou-se um método manual de cálculo desse parâmetro: as detecções erradas serão identificadas e contadas por meio da análise visual do gráfico, sendo considerada errada todas as detecções que estiverem visualmente diferentes da f_0 referente à nota soada em cada instante de tempo. Após contabilizar esses erros, faz-se o cálculo percentual de erros em relação ao total de detecções. A métrica de percentual de f_0 's detectadas é o inverso da métrica de erro percentual.

Os experimentos foram realizados para cada um dos 4 instrumentos musicais adotados neste projeto, sendo usado para todos eles a referência 196*bpm* para uma semicolcheia, nota com duração de $\frac{1}{4}$ de tempo de nota.

4.2 Objetivos

Esta bateria de experimentos tem por objetivo avaliar o desempenho do algoritmo proposto em detectar as frequências fundamentais nos áudios de 4 diferentes instrumentos musicais. Para essa avaliação, espera-se:

1. Verificar o percentual de acerto de frequência fundamental, com base no gráfico.
2. Identificar eventuais problemas de detecção e suas causas.
3. Discutir acerca da metodologia adotada no desenvolvimento.

4.3 Resultados Obtidos

Nesta seção, os resultados obtidos são expostos a partir dos gráficos gerados e descrição textual.

4.3.1 Clarinete

O clarinete, ou também clarino e clarinete soprano, é um instrumento de sopro. Ele integra o grupo de instrumentos transpositores, ou seja, sua nota soada é diferente da nota escrita na partitura. Isso ocorre pela sua afinação padrão em Bb. As principais características de seu timbre é o som aveludado, encorpado e penetrante, tendo muito brilho nas notas mais agudas. A Figura 4.1 mostra o clarinete ao lado de sua forma de onda, responsável pelo timbre do instrumento. Foi realizado experimento com os 3 áudios desse instrumento na base de dados: A escala de Bb, a música “*Agnus Dei*” e a música tema do game “Super Mário Brós”.

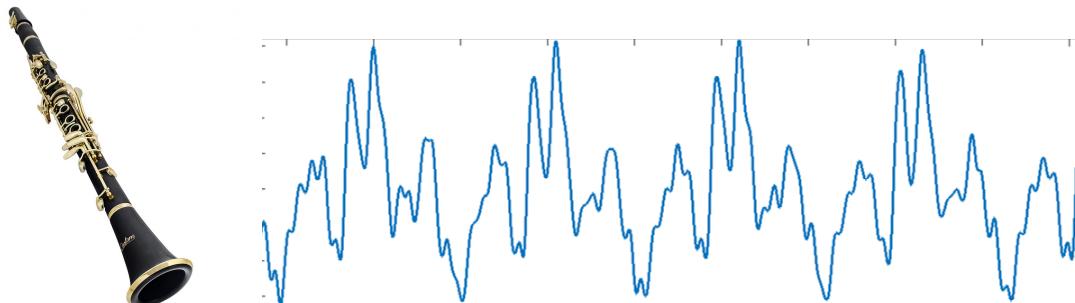


Figura 4.1: Clarinete e sua forma de onda

4.3.1.1 Escala: Bb Maior

O áudio com a escala de Bb Maior no clarinete tem duração de 20 segundos, contando com o tempo inicial sem notas. A Figura 4.2 exibe a interface gráfica para o experimento com o áudio da escala de Bb Maior.

Os gráficos obtidos no experimento são colocados em separado para melhor visualização das informações. A Figura 4.3a mostra o espectrograma do áudio, a Figura 4.3b exibe as saídas do sistema de detecção de f_0 , enquanto a Figura 4.3c compara as saídas do sistema de detecção com as frequências esperadas, conforme o arquivo de registros.

Os gráficos demonstram que o sistema obteve sucesso nas detecções de f_0 , ocorrendo erros em momentos específicos de apenas duas notas, onde detectou-se um harmônico devido ao efeito de ressonância da gravação. Foi calculado um sucesso de 93.4% nas detecções realizadas para este áudio.

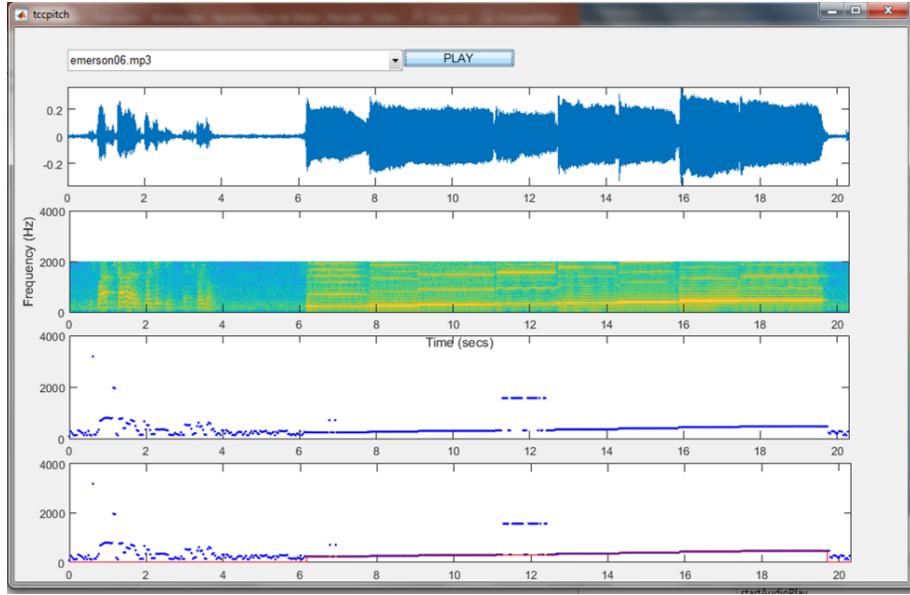


Figura 4.2: Interface de experimentação para áudio "Escala Bb Maior" com clarinete

4.3.1.2 Música: *Agnus Dei*

O áudio com a música “*Agnus Dei*” no clarinete tem duração de 21 segundos. A Figura 4.6 exibe a interface gráfica para o experimento com esse áudio.

Os gráficos obtidos no experimento são colocados em separado para melhor visualização das informações. A Figura 4.7a mostra o espectrograma do áudio, a Figura 4.7b exibe as saídas do sistema de detecção de f_0 , enquanto a Figura 4.7c compara as saídas do sistema de detecção com as frequências esperadas, conforme o arquivo de registros.

Para esta música, o sistema conseguiu detectar todas as frequências fundamentais, cometendo erro em apenas duas janelas, localizadas em pontos de mudança de nota. O percentual de sucesso na detecção foi de 99.6%, sendo que os erros podem ser desconsiderados, visto que no instante de alternância entre notas, a janela pode conter mistura de fundamentais da nota anterior com a nova nota que será soada.

4.3.1.3 Música: Super Mário Brós

O áudio com a música tema do game “Super Mário Brós” no clarinete tem duração de 21 segundos, contando com o tempo inicial sem notas. A Figura 4.6 exibe a interface gráfica para o experimento com esse áudio.

Os gráficos obtidos no experimento são colocados em separado para melhor visualização das informações. A Figura 4.7a mostra o espectrograma do áudio, a Figura 4.7b exibe as saídas do sistema de detecção de f_0 , enquanto a Figura 4.7c compara as saídas

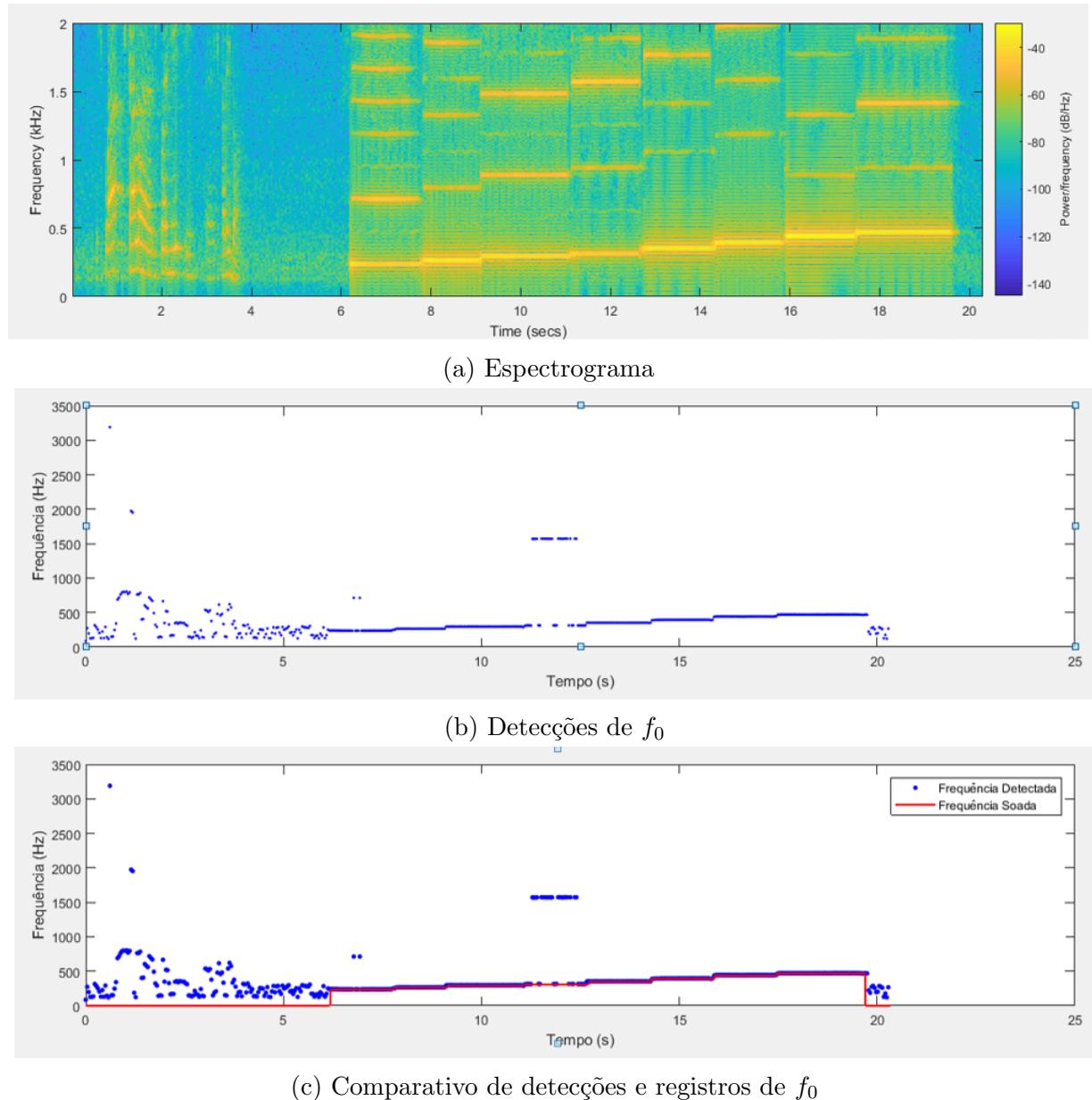


Figura 4.3: Escala de Bb Maior com clarinete

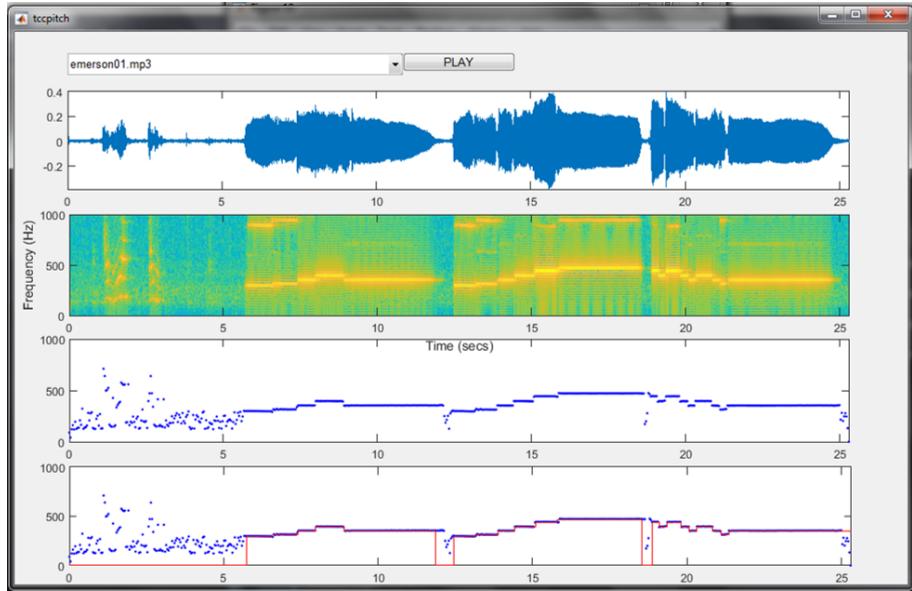


Figura 4.4: Interface de experimentação para áudio da música “Agnus Dei” com clarinete

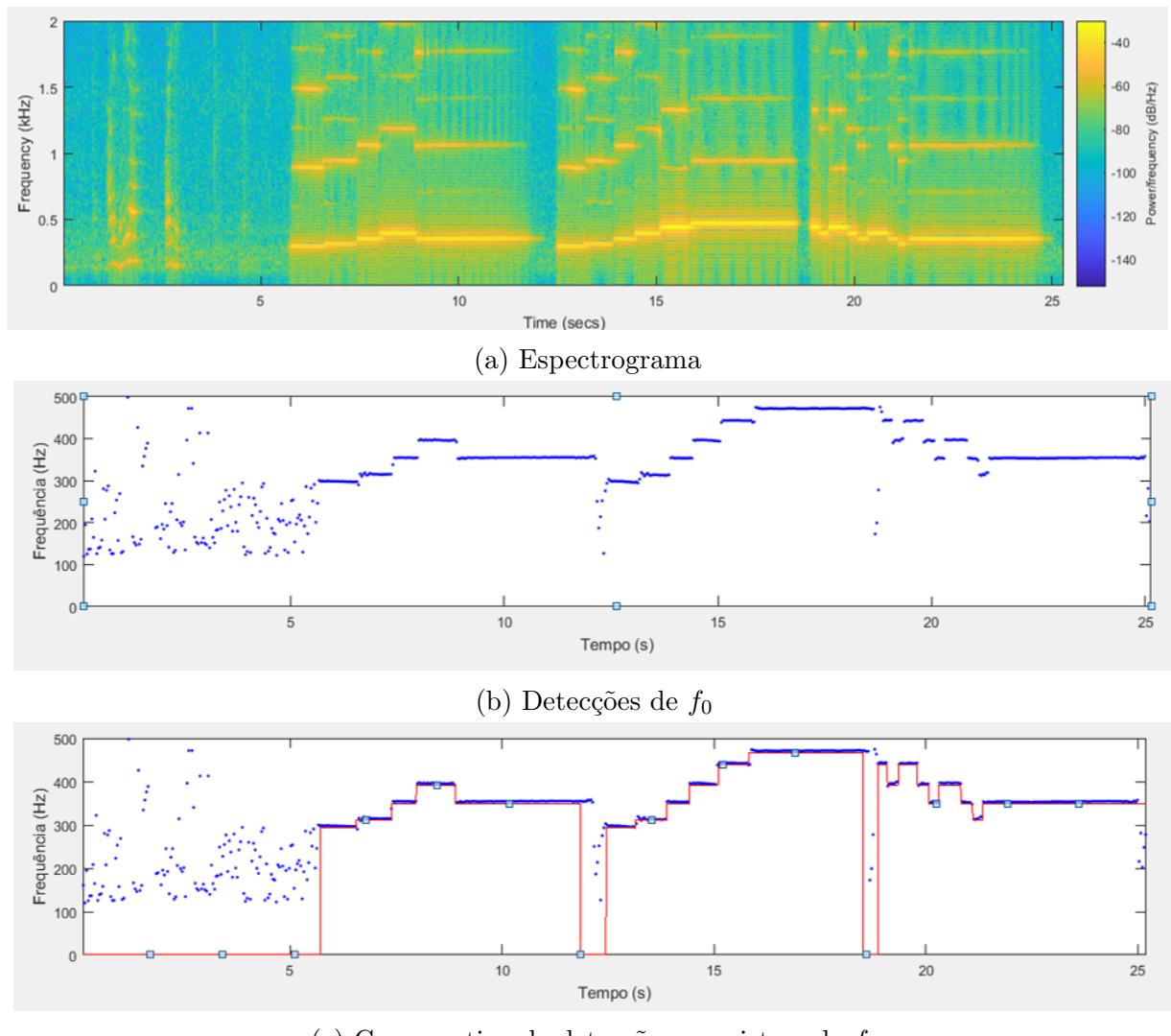


Figura 4.5: Música “Agnus Dei” com clarinete

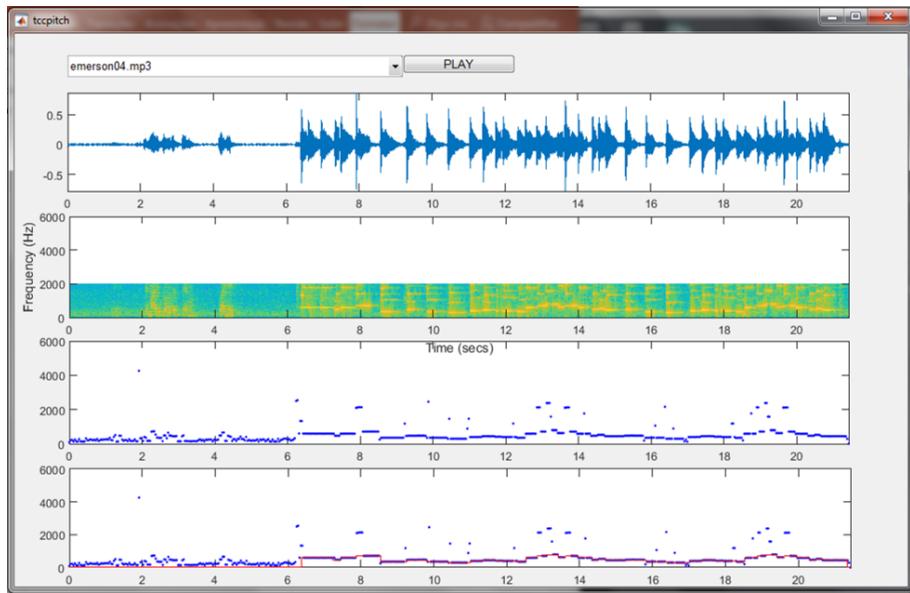


Figura 4.6: Interface de experimentação para áudio da Música “Super Mário Brós” com clarinete

do sistema de detecção com as frequências esperadas, conforme o arquivo de registros.

Para este caso, o sistema obteve um bom resultado nas detecções, entretanto, errou em 46 janelas, obtendo um percentual de sucesso nas detecções de 88.2%. Os erros ocorreram em sua maioria no tempo de subida das notas, e de forma harmônica. Uma característica interessante do clarinete a ser considerada é que, ao soar uma nova nota, se as mãos e a boca do instrumentista não estiverem bem posicionadas no instrumento, percebe-se um leve assobio em um harmônico da nota desejada. Esse efeito ocorre claramente na primeira nota da música tema do “Super Mário Brós”, e pode ser percebido em mais outra nota durante a execução.

4.3.2 Sax Alto

O Sax Alto também é um instrumento de sopro e, do mesmo modo que o clarinete, está incluso grupo de instrumentos transpositores, ou seja, sua nota soada é diferente da nota escrita na partitura. No caso do sax alto, sua afinação padrão é em Eb. As principais características de seu timbre é o som aveludado, avivado e encorpado. A Figura 4.8 mostra o sax alto ao lado de sua forma de onda. Foi realizado experimento com os 3 áudios desse instrumento na base de dados: A escala de Eb, e as músicas “Deus Cuida de Mim” e “Summertime”.

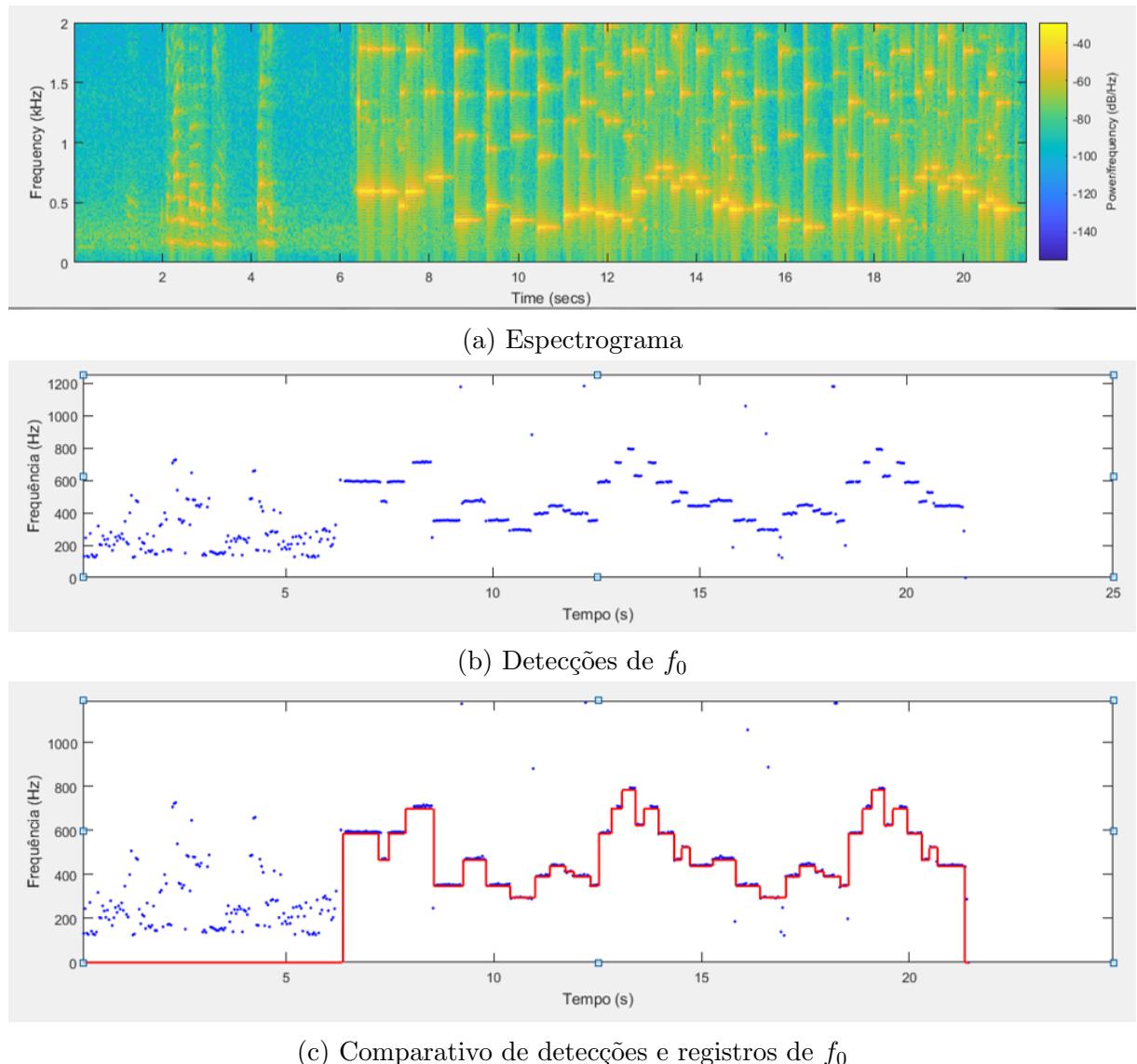


Figura 4.7: Música “Super Mário Brós” com clarinete

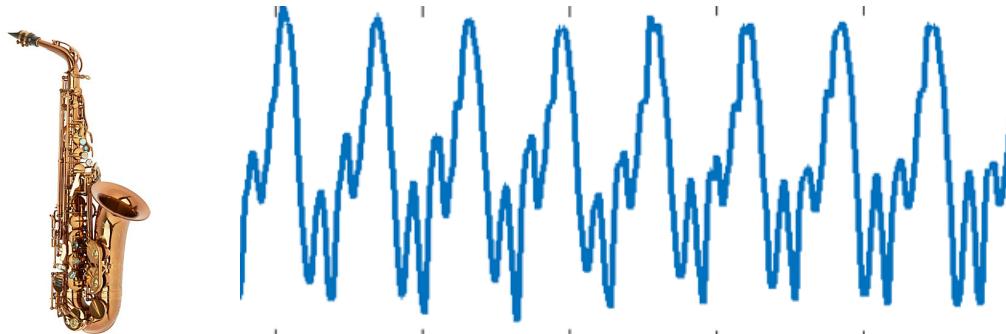


Figura 4.8: Sax alto e sua forma de onda

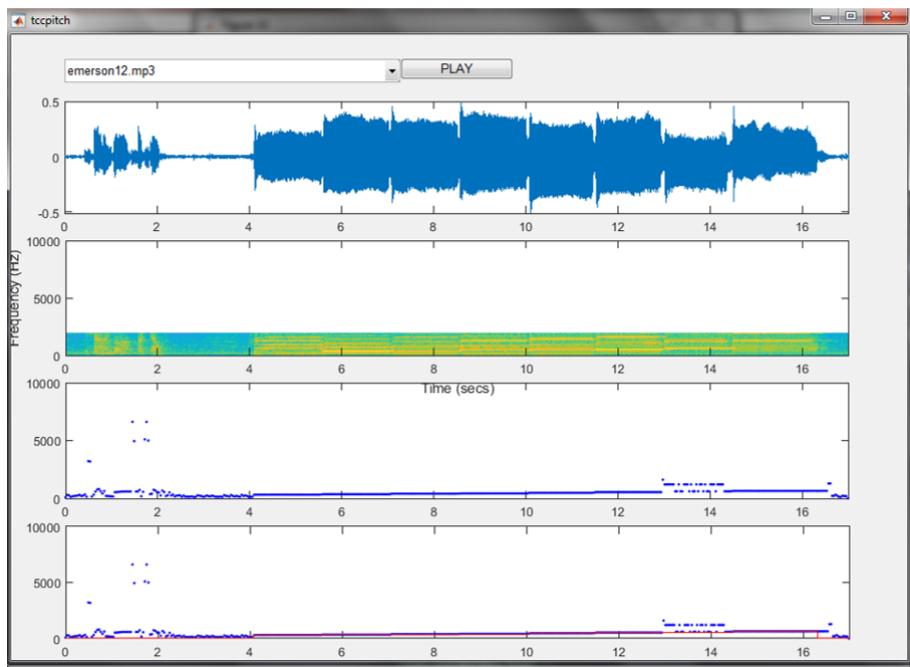


Figura 4.9: Interface de experimentação para áudio ”Escala Eb Maior” com sax alto

4.3.2.1 Escala: Eb Maior

O áudio com a escala de Eb Maior no sax tem duração de 18 segundos. A Figura 4.9 exibe a interface gráfica para o experimento com o áudio da escala de Eb Maior.

Os gráficos obtidos no experimento são colocados em separado para melhor visualização das informações. A Figura 4.10a mostra o espectrograma do áudio, a Figura 4.10b exibe as saídas do sistema de detecção de f_0 , enquanto a Figura 4.10c compara as saídas do sistema de detecção com as frequências esperadas, conforme o arquivo de registros.

Para este áudio, o sistema errou a detecção para a nota D5, capturando em mais de 50% da duração desta nota o harmônico com o dobro da frequência fundamental. Entretanto, não houve nenhum erro nas outras notas. Com isso, o percentual de sucesso na detecção para este áudio foi de 92.8%, um resultado satisfatório.

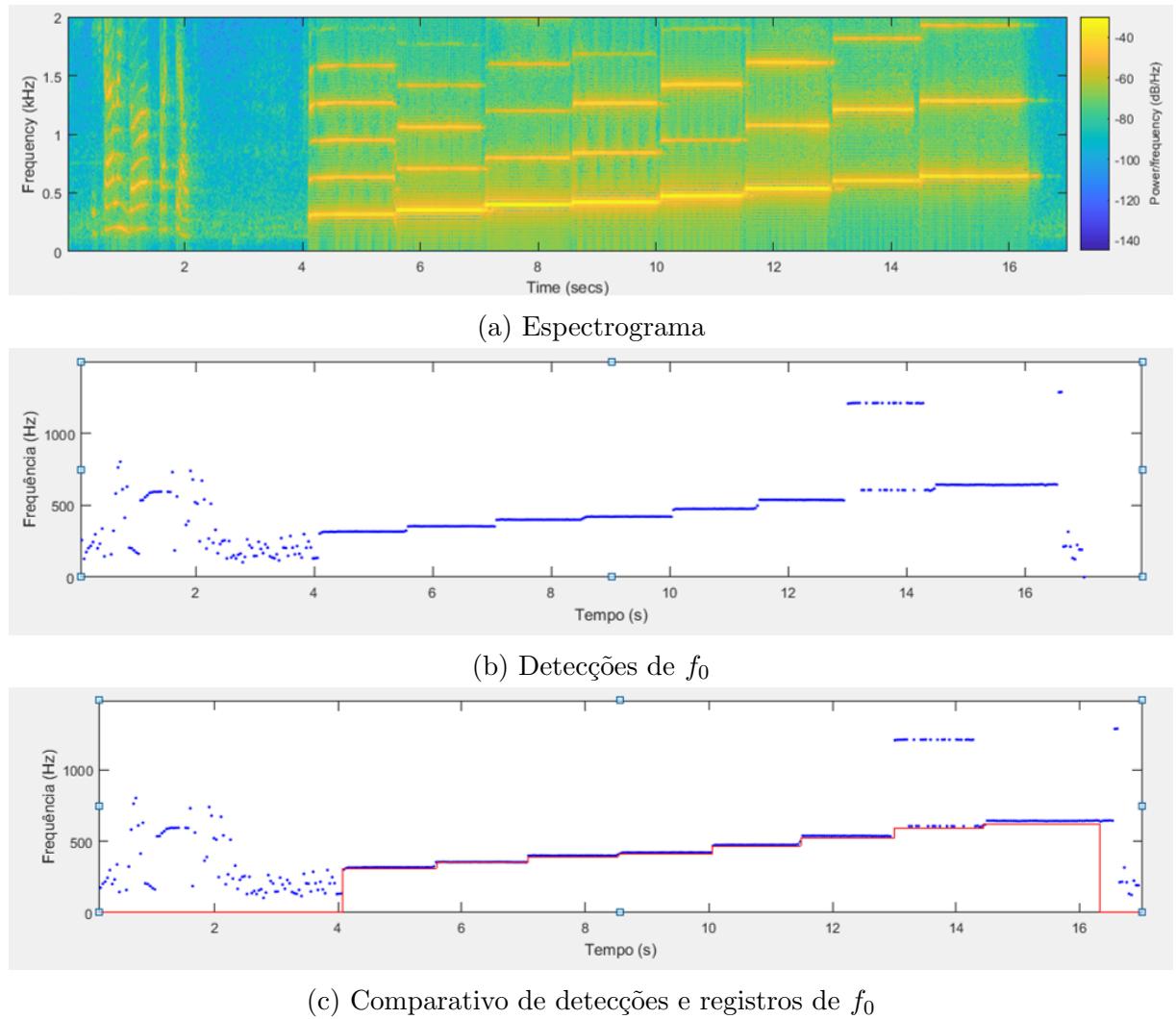


Figura 4.10: Escala Eb Maior com sax alto

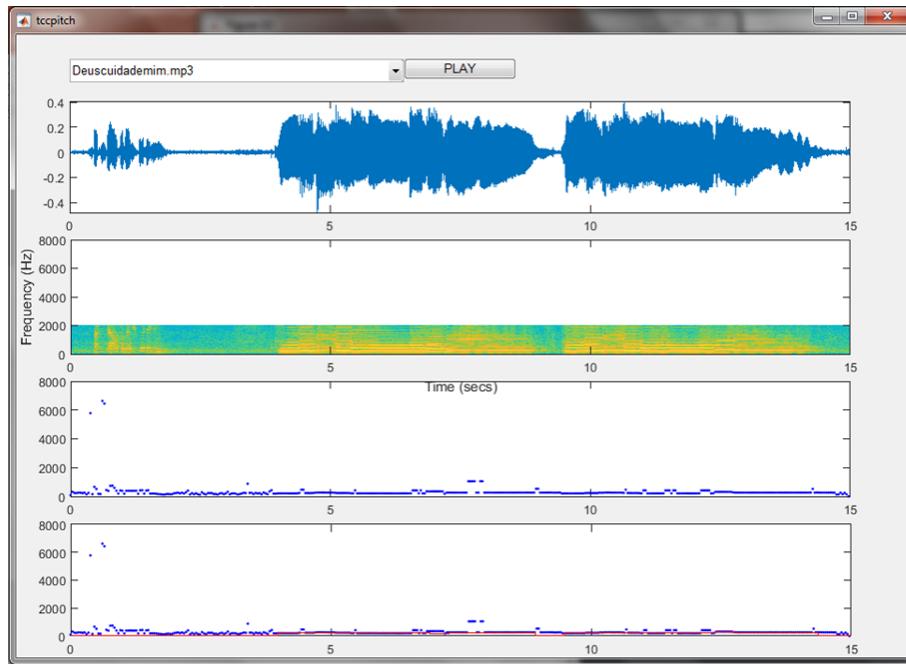


Figura 4.11: Interface de experimentação para áudio da Música “Deus Cuida de Mim” com sax alto

4.3.2.2 Música: Deus Cuida de Mim

O áudio com a música “Deus Cuida de Mim” no sax alto tem duração de 16 segundos. A Figura 4.11 exibe a interface gráfica para o experimento com esse áudio.

Os gráficos obtidos no experimento são colocados em separado para melhor visualização das informações. A Figura 4.12a mostra o espectrograma do áudio, a Figura 4.12b exibe as saídas do sistema de detecção de f_0 , enquanto a Figura 4.12c compara as saídas do sistema de detecção com as frequências esperadas, conforme o arquivo de registros.

O sistema conseguiu detectar corretamente em 85.9% das janelas, tendo errado todas as detecções na nota F3. Percebe-se uma real dificuldade em detectar tal nota, pois em ambas as ocorrências dela, detecta-se o harmônico.

4.3.2.3 Música: *Summertime*

O áudio com a música “*Summertime*” no sax alto tem duração de 29 segundos. A Figura 4.13 exibe a interface gráfica para o experimento com esse áudio.

Os gráficos obtidos no experimento são colocados em separado para melhor visualização das informações. A Figura 4.14a mostra o espectrograma do áudio, a Figura 4.14b exibe as saídas do sistema de detecção de f_0 , enquanto a Figura 4.14c compara as saídas do sistema de detecção com as frequências esperadas, conforme o arquivo de registros.

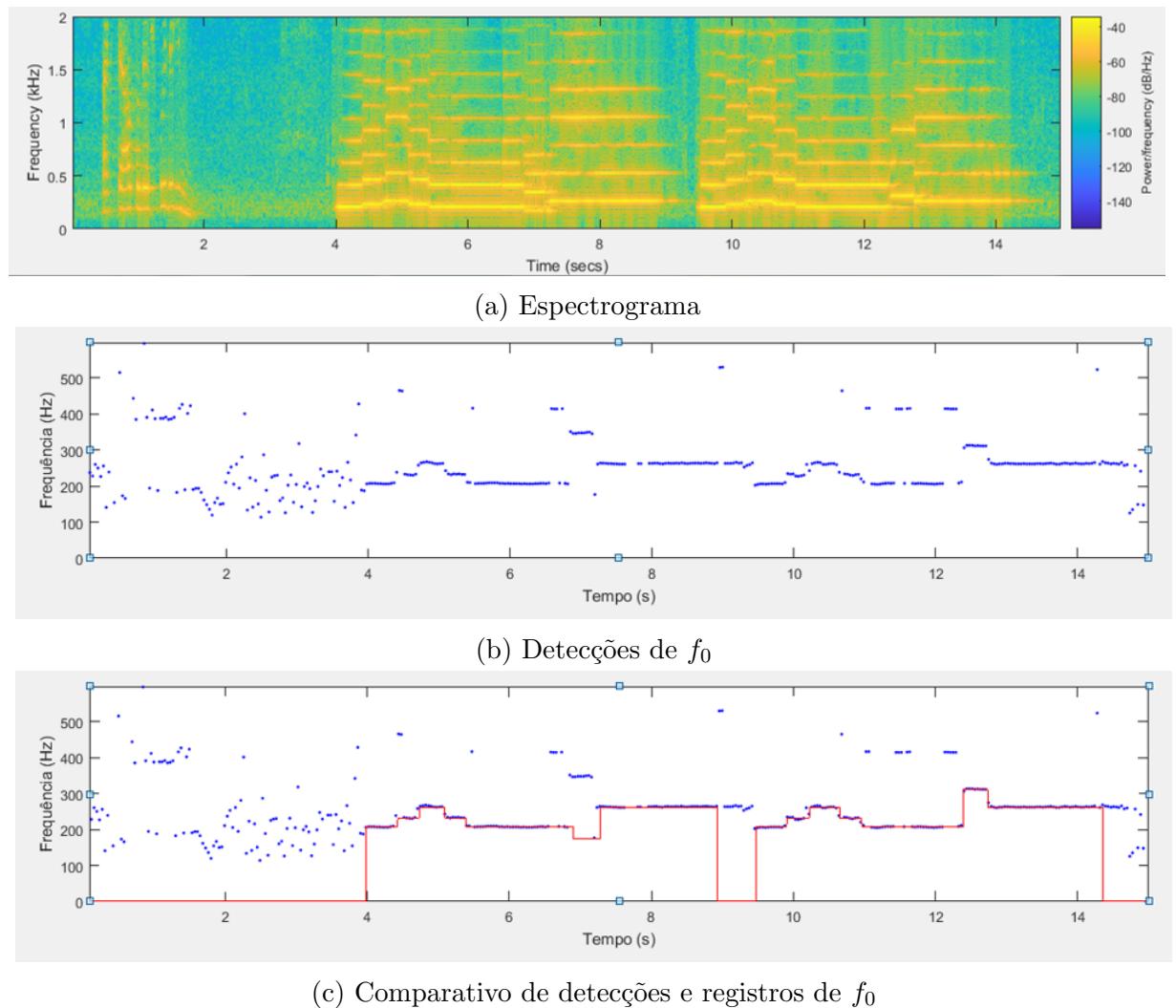


Figura 4.12: Música “Deus Cuida de Mim” no sax alto

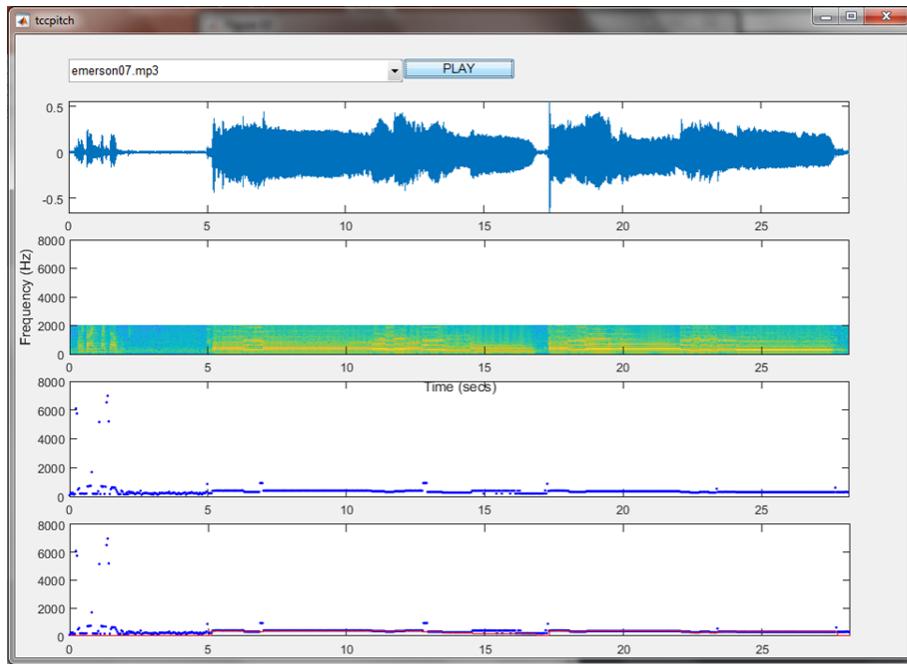


Figura 4.13: Interface de experimentação para áudio da música “*Summertime*” com sax alto

Para este áudio, o sistema também errou em mais de 50% da duração de uma das notas, dessa vez G3, única nota da 3^a escala na música. Entretanto, as detecções nas demais notas, todas da 4^a escala, obtiveram mínimos erros, de modo que a taxa de sucesso nas detecções foi de 91.5%. Com este áudio, ficou evidente que há uma dificuldade para que o sistema consiga detectar frequências fundamentais para notas abaixo de 200Hz no saxofone alto.

4.3.3 Piano Eletrônico

O piano é um instrumento de corda percussivas, entretanto, o piano eletrônico tem o papel de facilitar o uso do piano, de modo que torna-se cada vez mais raro encontrar um piano acústico sendo usado. O piano eletrônico busca imitar o som do piano acústico e, embora não seja perfeitamente igual, já alcança uma boa semelhança. A Figura 4.15 mostra o piano eletrônico ao lado de sua forma de onda, responsável pelo timbre do instrumento.

4.3.3.1 Escala: C Maior

O áudio com a escala de C Maior no piano tem duração de 12 segundos. A Figura 4.16 exibe a interface gráfica para o experimento com o áudio da escala de C Maior no

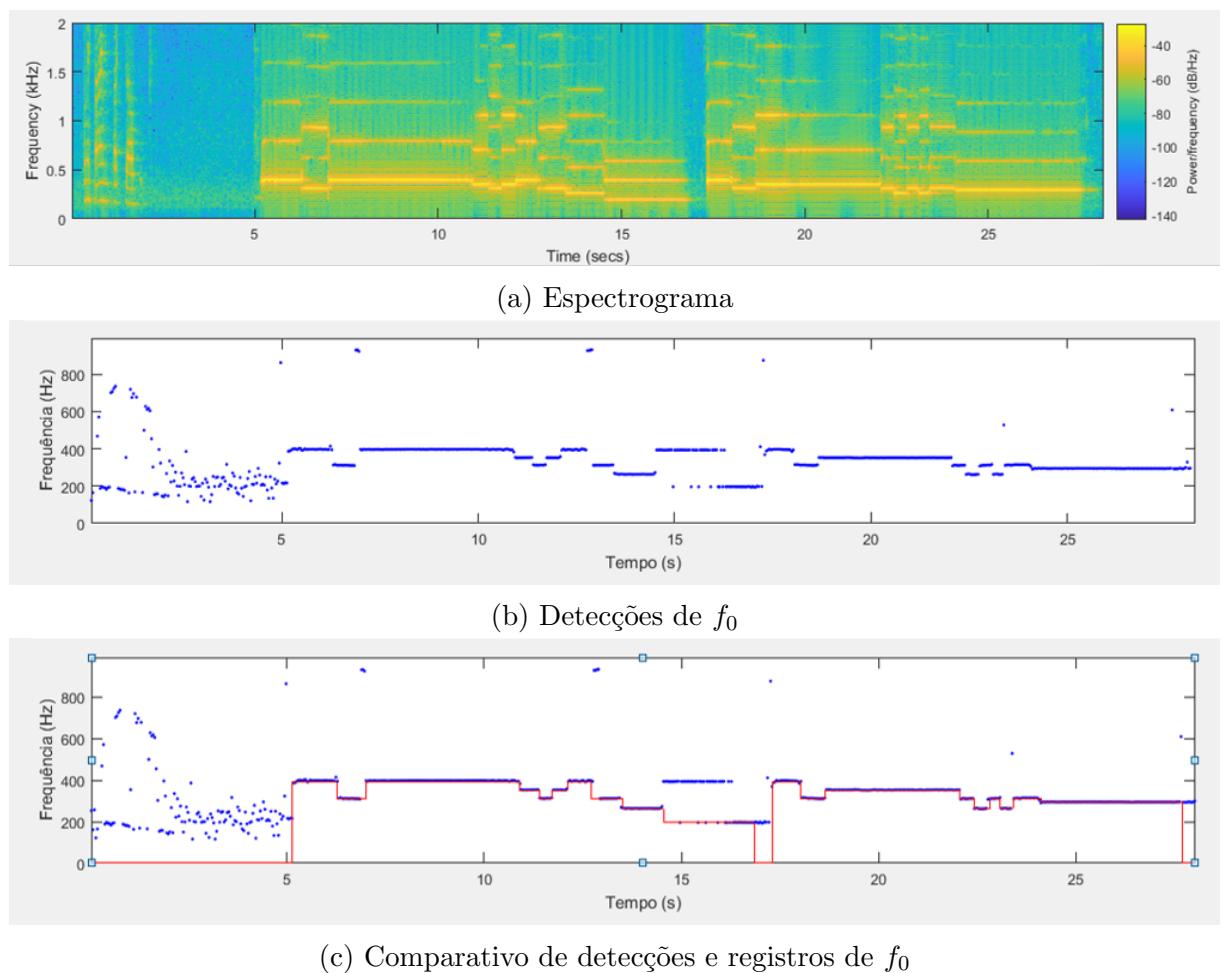


Figura 4.14: Audio da música “Summertime” no sax alto

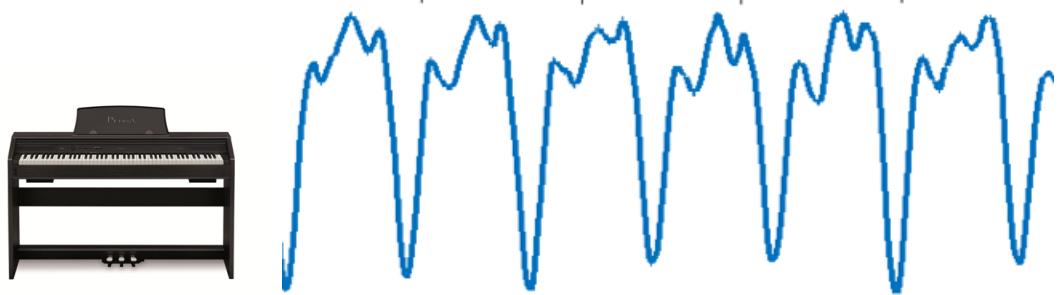


Figura 4.15: Piano e sua forma de onda

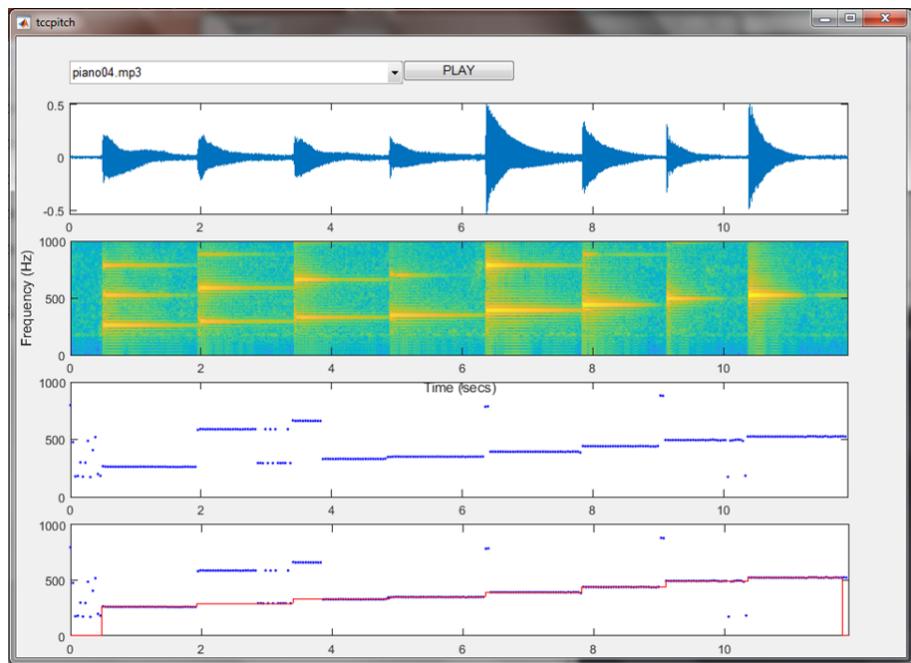


Figura 4.16: Interface de experimentação para áudio da escala C Maior com piano eletrônico

piano eletrônico.

Os gráficos obtidos no experimento são colocados em separado para melhor visualização das informações. A Figura 4.17a mostra o espectrograma do áudio, a Figura 4.17b exibe as saídas do sistema de detecção de f_0 , enquanto a Figura 4.17c compara as saídas do sistema de detecção com as frequências esperadas, conforme o arquivo de registros.

Para a escala de C Maior, o sistema obteve percentual de sucesso nas detecções de 85.1%. As notas D4 e E4 foram as que mais tiveram erros, tendo suas harmônicas detectadas no dobro da f_0 real. Nas demais notas, o sistema conseguiu trabalhar bem.

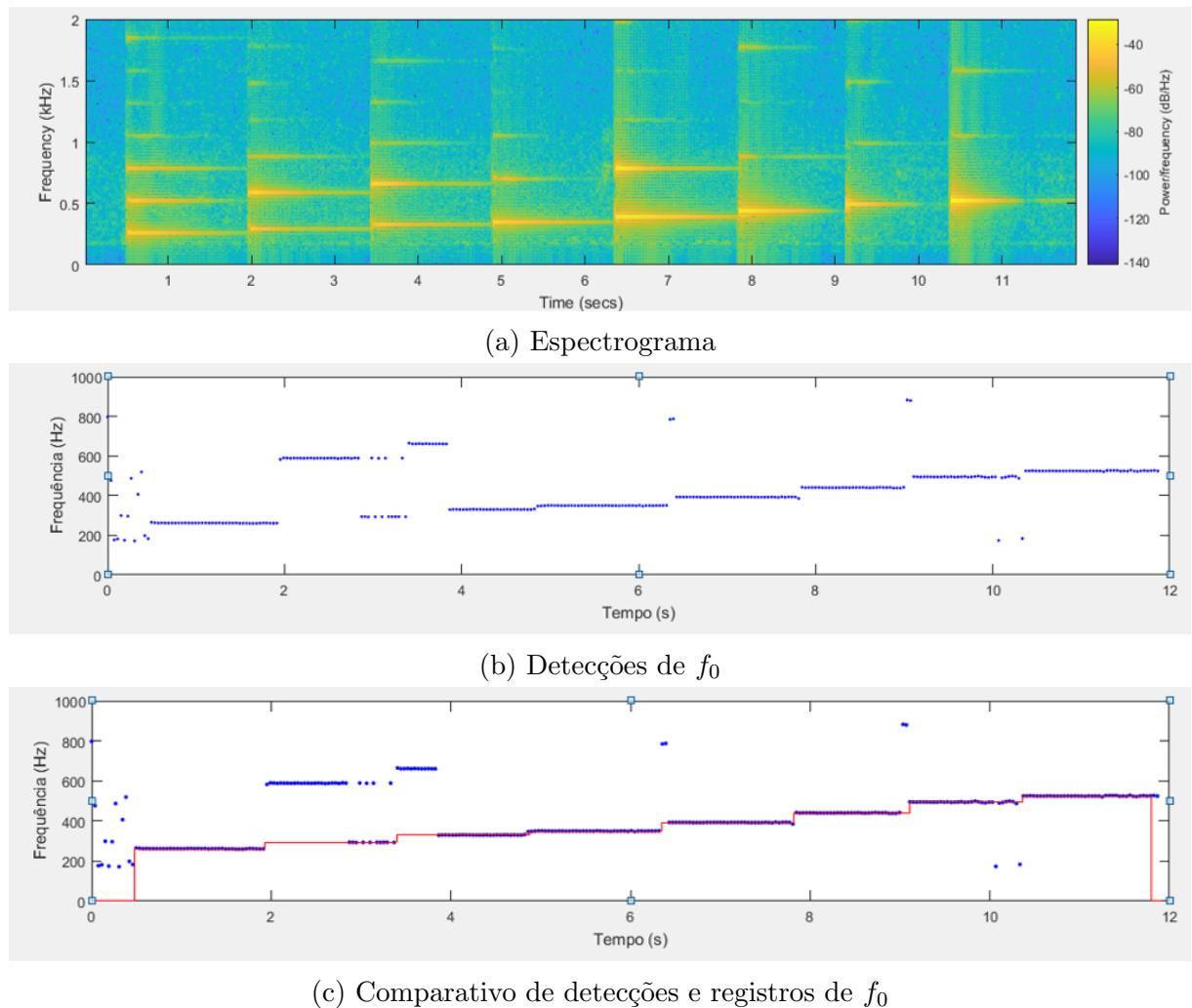


Figura 4.17: Escala de C Maior no piano eletrônico

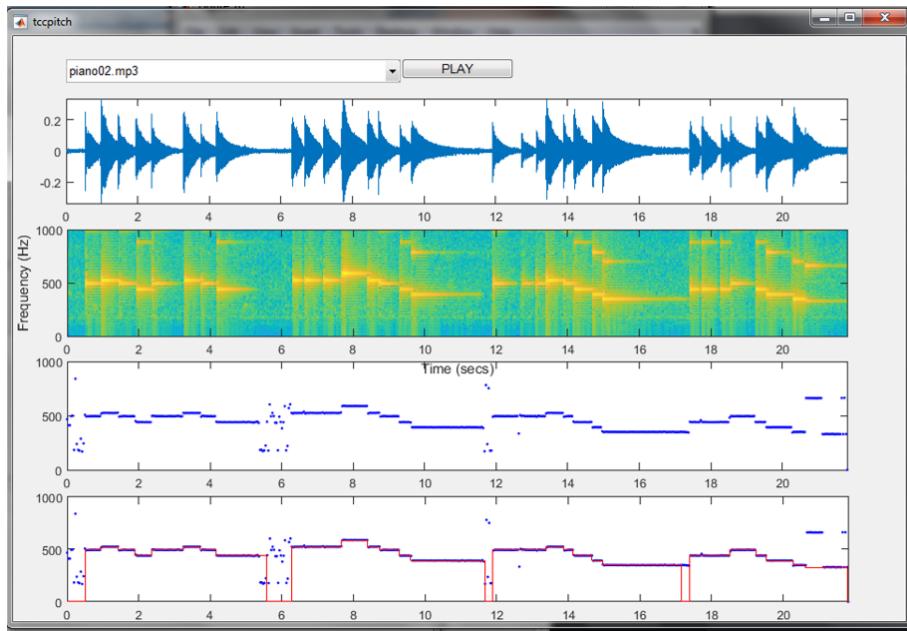


Figura 4.18: Interface de experimentação para áudio da música “Eu Navegarei” com piano eletrônico

4.3.3.2 Música: Eu Navegarei

O áudio com a música “Eu Navegarei” no piano tem duração de 22 segundos. A Figura 4.18 exibe a interface gráfica para o experimento com esse áudio.

Os gráficos obtidos no experimento são colocados em separado para melhor visualização das informações. A Figura 4.19a mostra o espectrograma do áudio, a Figura 4.19b exibe as saídas do sistema de detecção de f_0 , enquanto a Figura 4.19c compara as saídas do sistema de detecção com as frequências esperadas, conforme o arquivo de registros.

Para este áudio, o sistema obteve sucesso em 97% das janelas analisadas. A maior parte dos 3% de erros foram durante a execução de uma nota E4, repetindo o problema detectado no áudio da escala de C Maior. Mesmo assim, o sistema alcançou um resultado muito satisfatório para esse experimento

4.3.3.3 Música: *Skyfall*

O áudio com a música “*Skyfall*” no piano tem duração de 12 segundos. A Figura 4.20 exibe a interface gráfica para o experimento com esse áudio.

Os gráficos obtidos no experimento são colocados em separado para melhor visualização das informações. A Figura 4.21a mostra o espectrograma do áudio, a Figura 4.21b exibe as saídas do sistema de detecção de f_0 , enquanto a Figura 4.21c compara as saídas

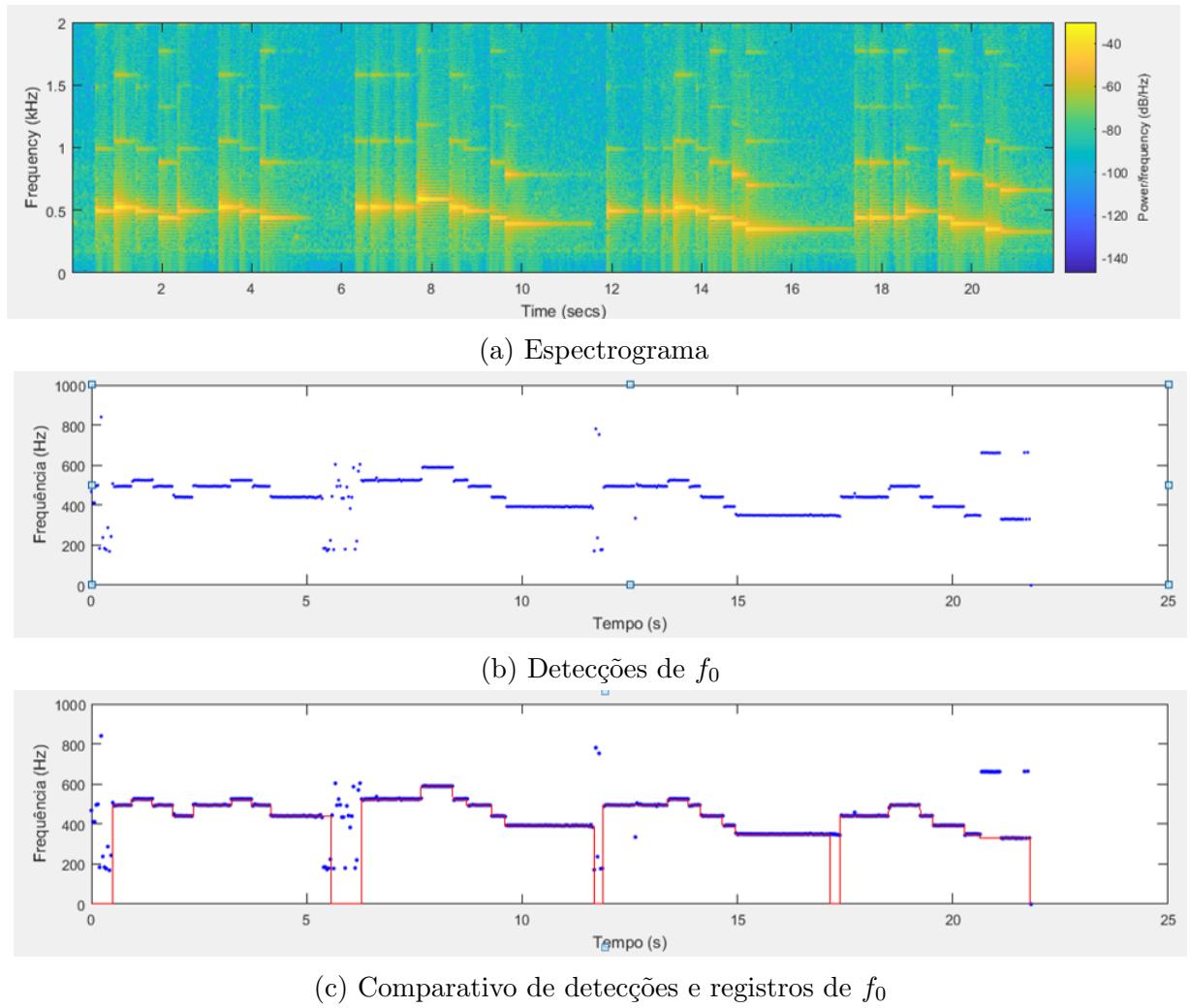


Figura 4.19: Áudio da música “Eu Navegarei” tocado com piano eletrônico

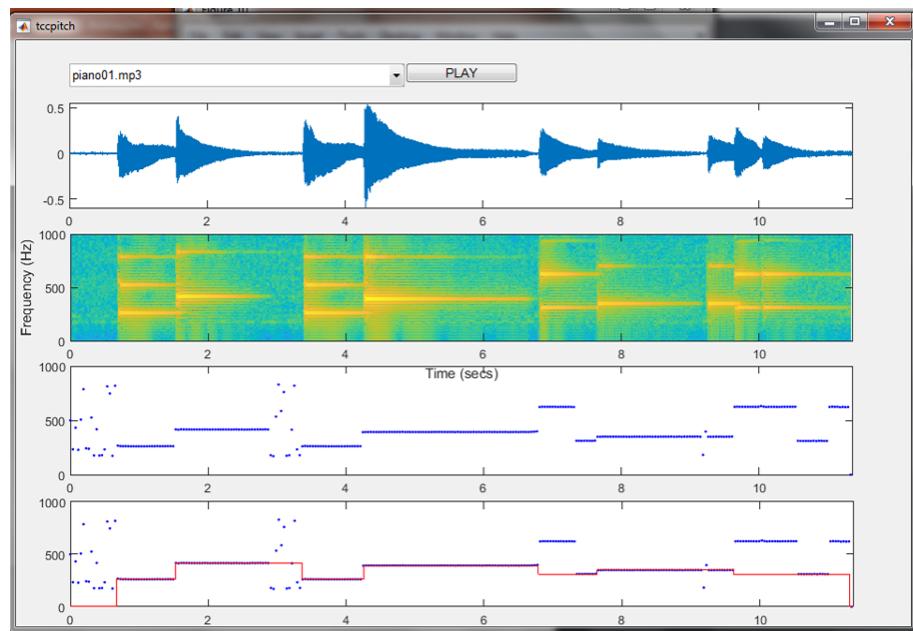
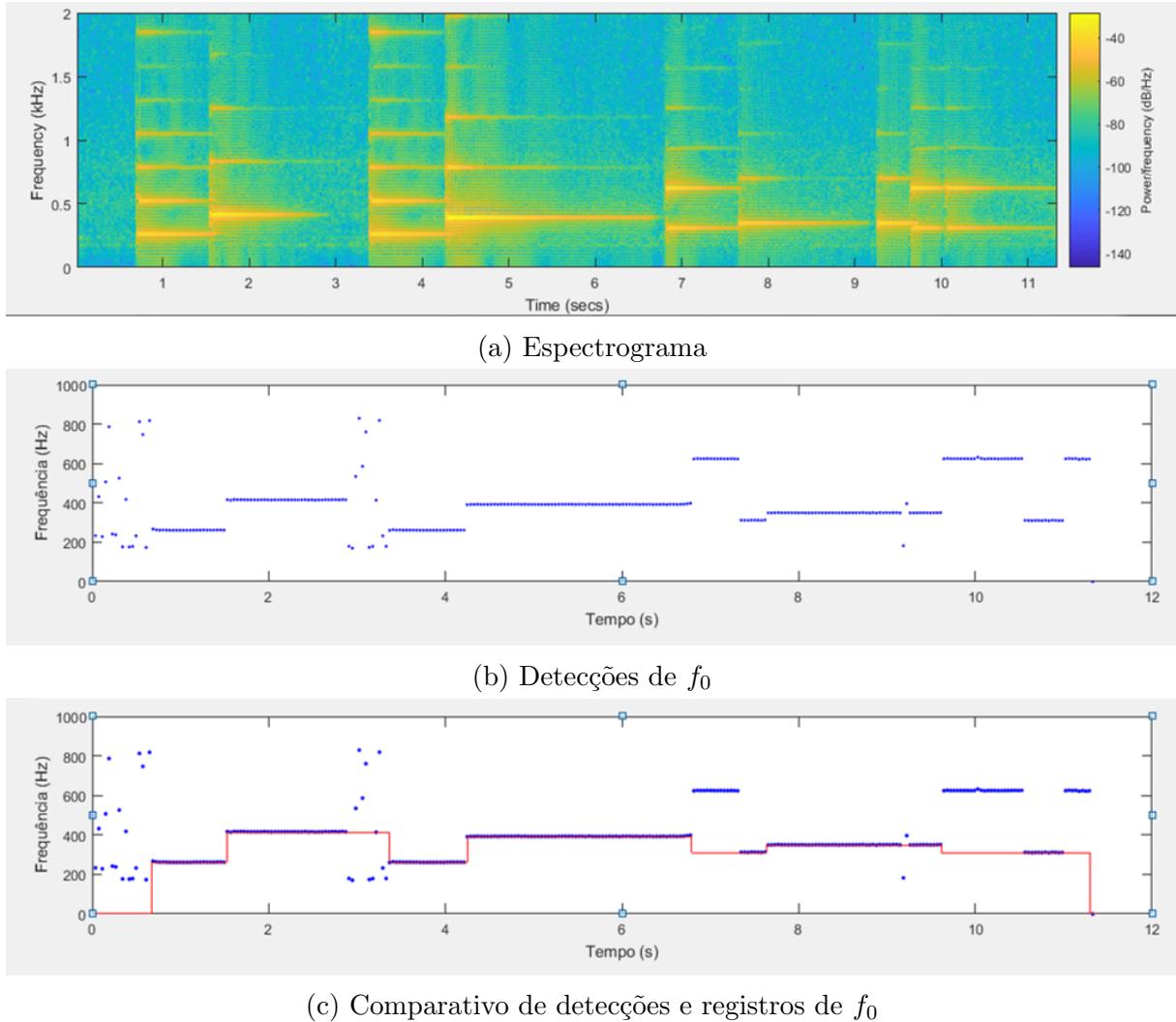


Figura 4.20: Interface de experimentação para áudio da música “Skyfall” com piano

Figura 4.21: Música “*Skyfall*” tocada no piano

do sistema de detecção com as frequências esperadas, conforme o arquivo de registros.

Para o áudio “*Skyfall*”, o sistema teve dois tipos de erros principais: O primeiro refere-se ao problema para detectar Eb4, pois em todos os experimentos com piano eletrônico, o sistema demonstrou uma grande dificuldade em detectar as fundamentais relativas as notas D4, Eb4 e E4, entre 290Hz e 330Hz. O segundo problema esteve relacionado ao decaimento da nota. Neste áudio, o sistema obteve 79.5% de taxa de sucesso nas detecções, um valor bem abaixo da média alcançada entre os instrumentos de sopro.

4.3.4 Violão

O violão é um instrumento de corda. A Figura 4.22 mostra o violão ao lado de sua forma de onda.

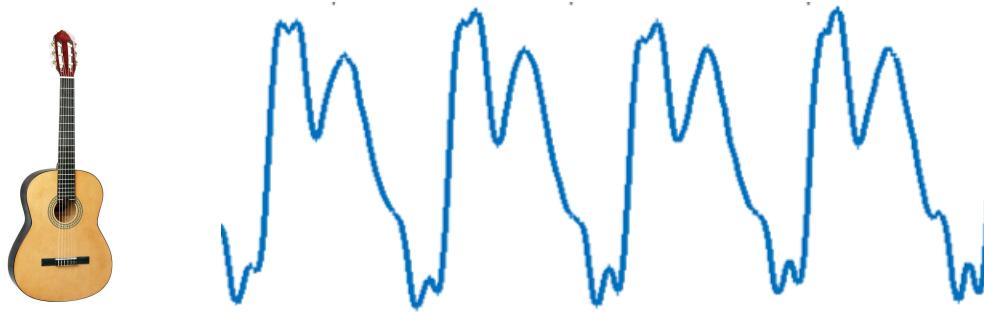


Figura 4.22: Violão e sua forma de onda

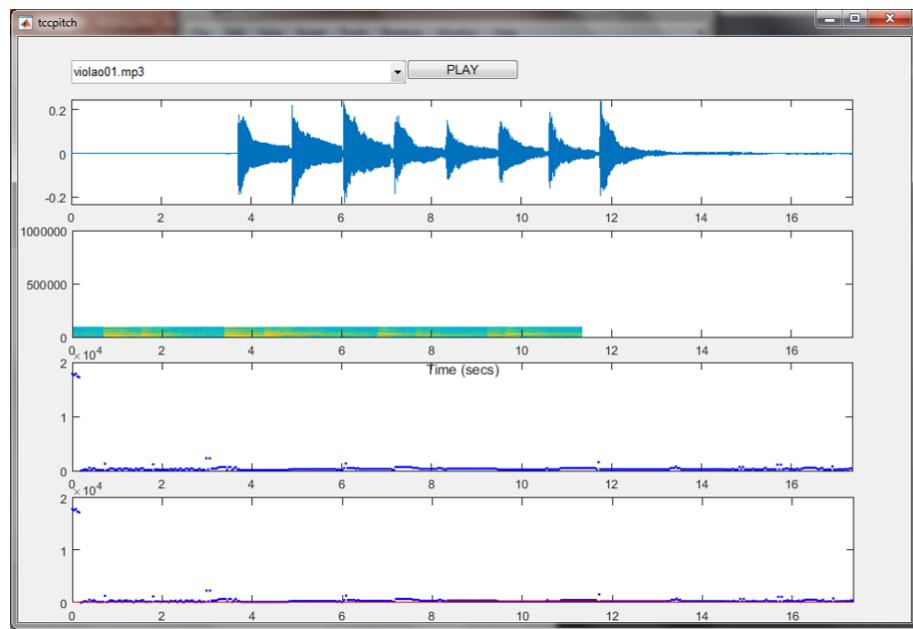


Figura 4.23: Interface de experimentação para áudio da escala de C Maior com violão

4.3.4.1 Escala: C Maior

O áudio com a escala de C Maior no violão tem duração de 17 segundos, contando com o tempo inicial de silêncio. A Figura 4.23 exibe a interface gráfica para o experimento com o áudio da escala de C Maior tocada pelo violão.

Os gráficos obtidos no experimento são colocados em separado para melhor visualização das informações. A Figura 4.24a mostra o espectrograma do áudio, a Figura 4.24b exibe as saídas do sistema de detecção de f_0 , enquanto a Figura 4.24c compara as saídas do sistema de detecção com as frequências esperadas, conforme o arquivo de registros.

Para este áudio, o sistema apresentou o pior resultado, tendo 46.3% de sucesso nas detecções. Somente as notas C3, A3 e C4 foram detectadas corretamente, enquanto todas as outras foram detectadas no harmônico com dobro de f_0 . A principal característica a ser considerada nessa questão é o funcionamento físico do violão, que amplia o som da

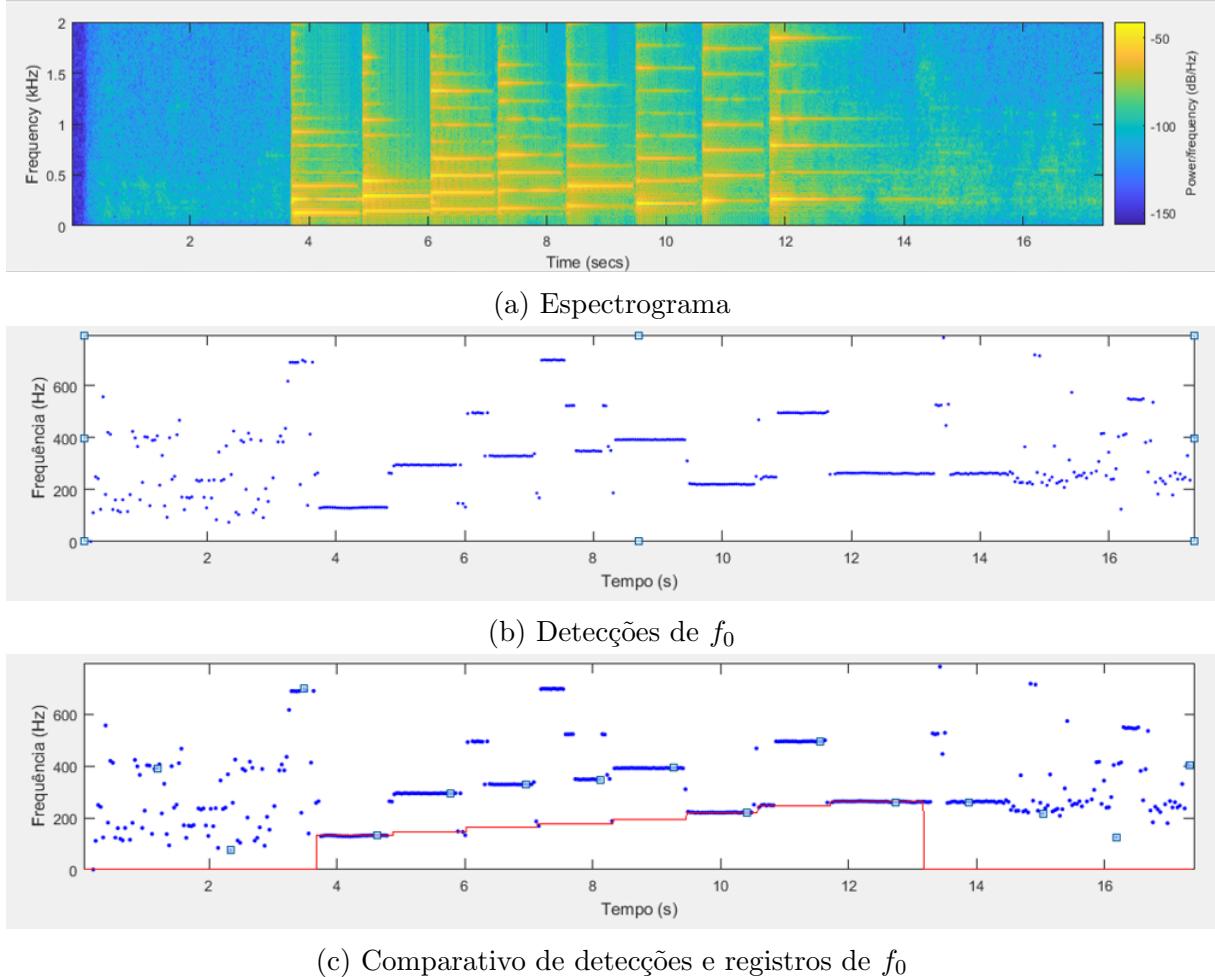


Figura 4.24: Escala de C Maior no violão

vibração de suas cordas através de sua caixa de ressonância. Sendo assim, é comum que o fenômeno da ressonância aconteça, o que atrapalha a performance do sistema.

4.3.4.2 Música: Nona Sinfonia

O áudio com a música “Nona Sinfonia” no violão tem duração de 15 segundos, contando com o tempo inicial sem notas. A Figura 4.25 exibe a interface gráfica para o experimento com esse áudio.

Os gráficos obtidos no experimento são colocados em separado para melhor visualização das informações. A Figura 4.26a mostra o espectrograma do áudio, a Figura 4.26b exibe as saídas do sistema de detecção de f_0 , enquanto a Figura 4.26c compara as saídas do sistema de detecção com as frequências esperadas, conforme o arquivo de registros.

Para este áudio, a detecção obteve taxa de 72.4% de sucesso nas detecções. Somente a nota G3 que não foi detectada, tendo sido apontado o harmônico da f_0 . Esse resultado

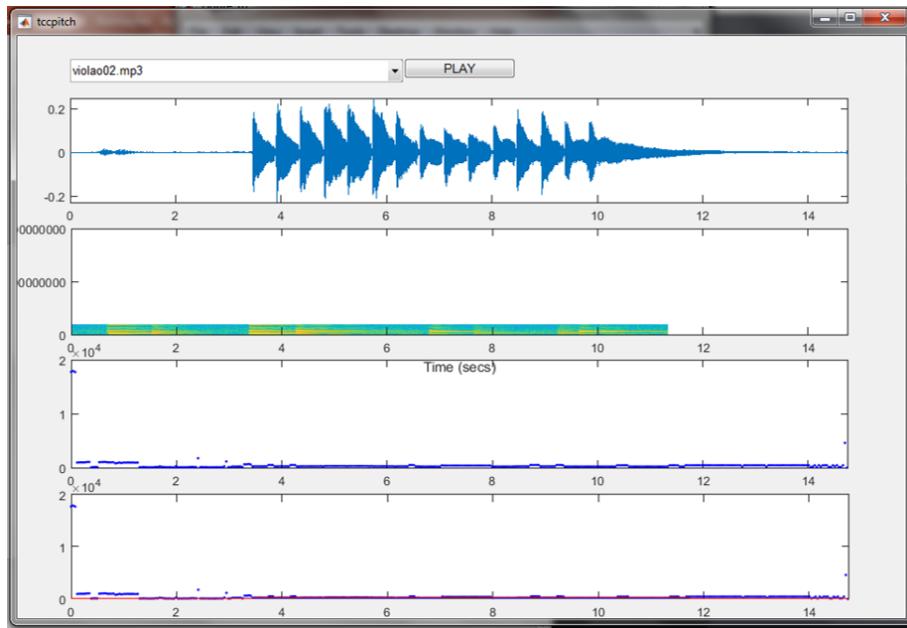


Figura 4.25: Interface de experimentação para áudio da música “Nona Sinfonia” com violão

evidencia que a dificuldade com sons mais graves acontece em todos os instrumentos, sendo difícil diferenciar qual o harmônico fundamental.

4.3.4.3 Música: Deus de Abraão

Por fim, o último áudio a ser analisado foi a música “Deus de Abraão” no violão, que tem duração de 22 segundos, contando com o tempo inicial de silêncio. A Figura 4.27 exibe a interface gráfica para o experimento com esse áudio.

Os gráficos obtidos no experimento são colocados em separado para melhor visualização das informações. A Figura 4.28a mostra o espectrograma do áudio, a Figura 4.28b exibe as saídas do sistema de detecção de f_0 , enquanto a Figura 4.28c compara as saídas do sistema de detecção com as frequências esperadas, conforme o arquivo de registros.

Os resultados para esse áudio foram bem acima do esperado, alcançando na detecção de f_0 , para notas de violão, a taxa de sucesso de 92.6%. O sucesso se deve principalmente ao uso de notas mais agudas, em maioria da 4^a oitava. Nesse experimento, os erros ocorreram em sua maioria nas 2 notas mais graves, pertencentes a 3^a oitava (F#3 e B3). Deste modo, fica evidente que o sistema apresenta bons resultados para sons mais agudos, e tem maior dificuldade para discernir sons mais graves.

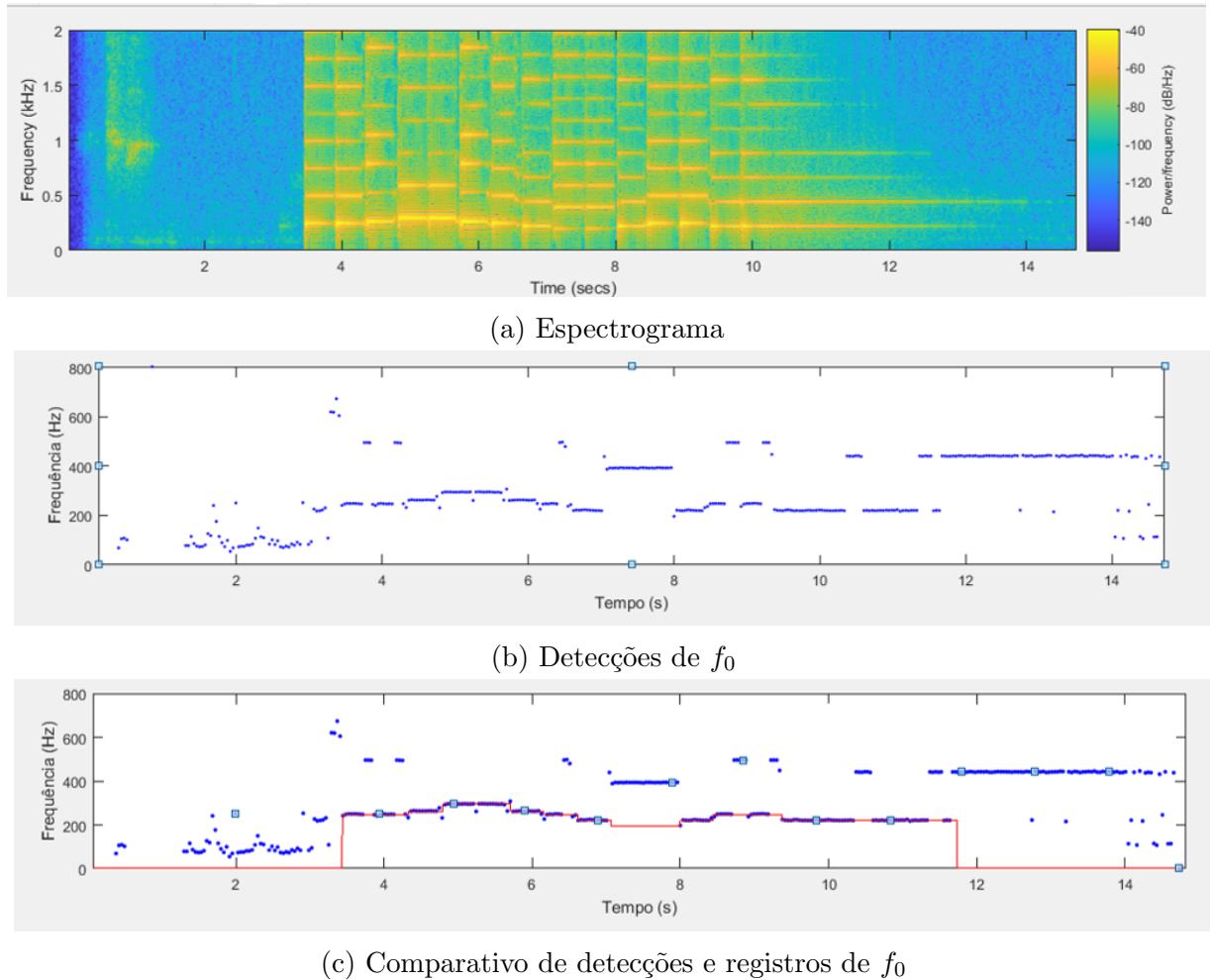


Figura 4.26: Música “Nona Sinfonia” tocada no violão.

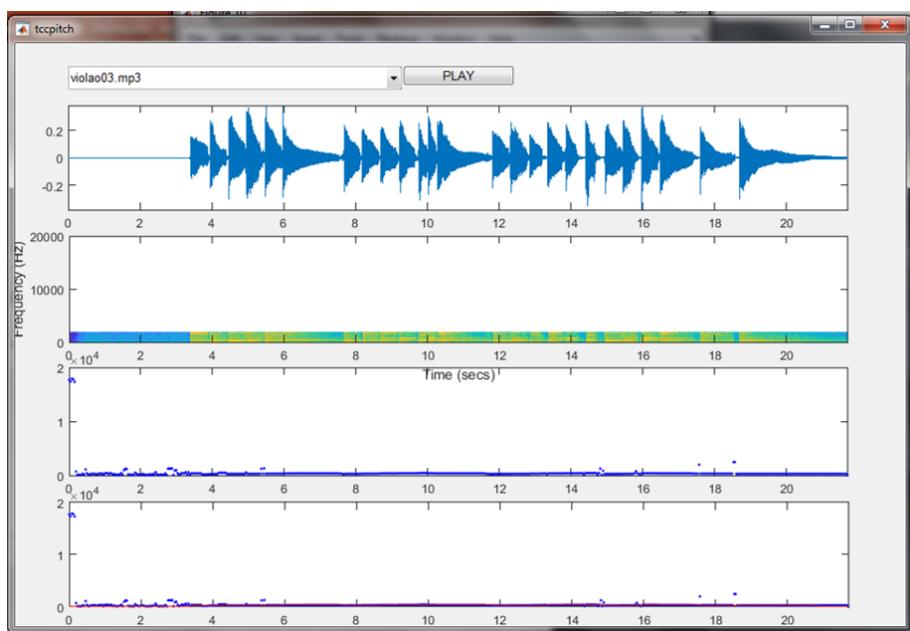


Figura 4.27: Interface de experimentação para áudio da música “Deus de Abraão” com violão

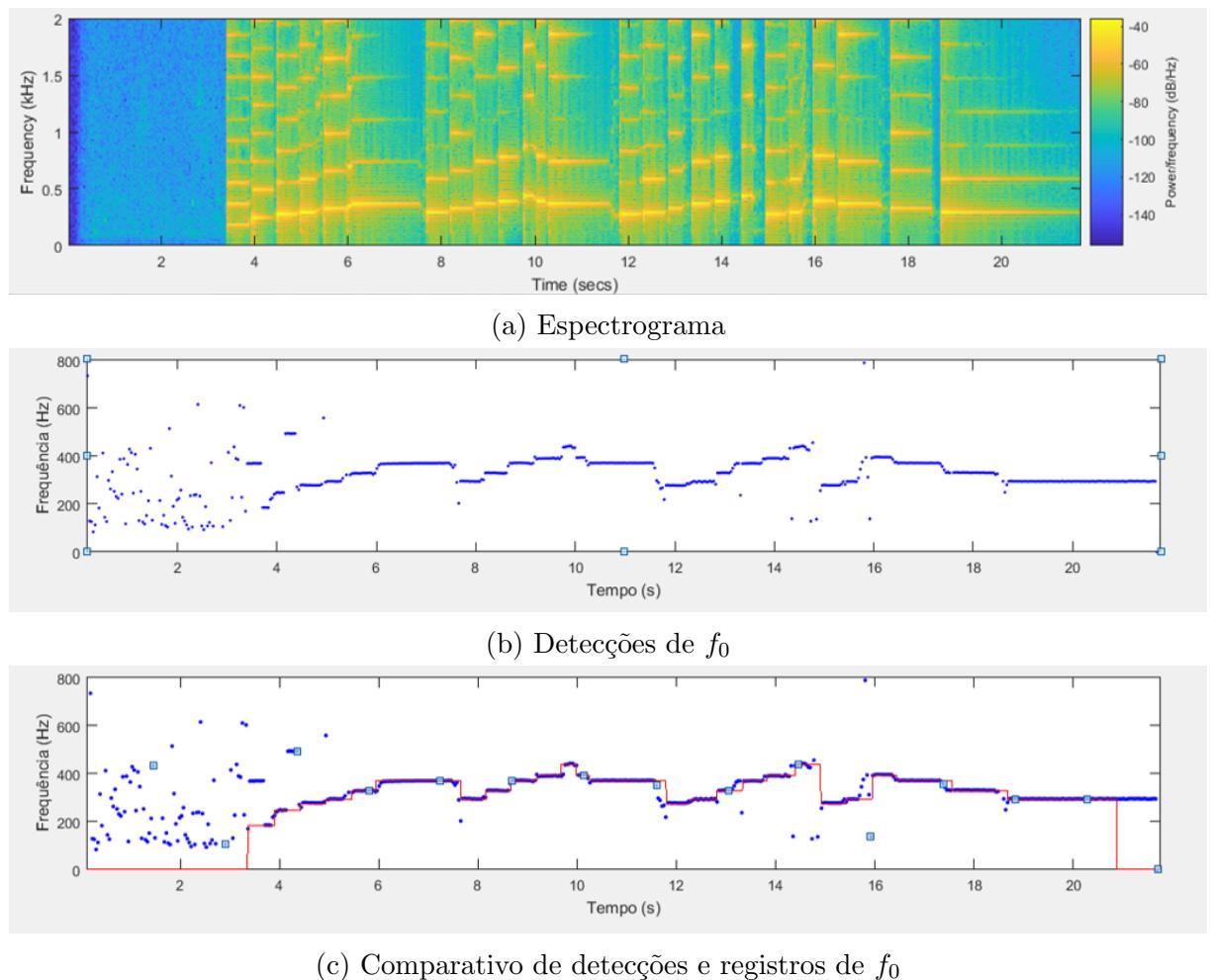


Figura 4.28: Música “Deus de Abraão” tocada no violão.

4.4 Discussões

O sistema alcançou resultados satisfatórios, visto que detectou com sucesso um percentual de cerca de 85% das frequências fundamentais. Entre os instrumentos de sopro, o percentual médio de sucesso ultrapassou os 90%. Em compensação, percebeu-se que o sistema ainda é deficiente em melodias graves, principalmente para instrumentos de corda. A Tabela 4.1 exibe as taxas de sucesso obtidas em cada áudio. Os erros nas detecções estiveram em sua maioria relacionados a dificuldade do sistema em detectar fundamentais em notas muito graves, e em outros casos, os erros eram em função de não sustentação da nota ou ruídos de execução. Muitos desses erros podem ser evitados se o sistema for combinado a algum método estatístico de tratamento que identifique a continuidade de uma nota. Entretanto, torna-se um verdadeiro desafio garantir a detecção para notas com efeito de ressonância.

Tabela 4.1: Comparativo entre os áudios

INSTRUMENTO	ÁUDIO	SUCESSO NAS DETECÇÕES
CLARINETE	Escala - Bb Maior	93.4%
	Música - <i>Agnus Dei</i>	99.6%
	Música - Super Mário Brós	88.2%
SAX ALTO	Escala - Bb Maior	92.8%
	Música - Deus Cuida de Mim	85.9%
	Música - <i>Summertime</i>	91.5%
PIANO ELETRÔNICO	Escala - C Maior	85.1%
	Música - Eu Navegarei	97.0%
	Música - <i>Skyfall</i>	79.5%
VIOLÃO	Escala - C Maior	46.4%
	Música - Nona Sinfonia	72.4%
	Música - Ao Deus de Abraão	92.6%

O método de detecção de f_0 buscando a frequência de maior amplitude, por meio da STFT, é trivialmente falho na ocorrência de ressonância, visto que o efeito de ressonância consiste na superação da amplitude da fundamental por um harmônico em frequência diferente. Diante desse problema, torna-se necessário combinar o método proposto com outros métodos que busquem identificar o efeito de ressonância, de modo a tratar os erros causados por esse efeito. Neste sentido, o sistema implementado apenas da forma como foi proposto não está apto para a detecção de fundamentais em contextos que apresentem o efeito de ressonância. De todo modo, as fundamentais detectadas em 99% das vezes eram

suficientes para fornecer informações da nota soada, divergindo apenas para identificar a oitava a qual essa nota pertence.

A metodologia adotada para a implementação desse projeto mostrou-se de grande potencial, podendo ser evoluída para um sistema mais robusto e com menor sensibilidade ao efeito de ressonância. O ambiente montado para a experimentação também demonstrou ser prático e de fácil manutenção, permitindo que o ritmo de experimentação seja satisfatório. A base de dados foi utilizada com sucesso, e permitiu a verificação de diversos incidentes possíveis na execução de uma melodia no cotidiano do músico, desde os erros humanos até mesmo os efeitos sonoros gerados por cada instrumento, devido às suas particularidades.

Capítulo 5

Conclusão

O presente projeto implementou um sistema de detecção de frequências fundamentais para áudios musicais, baseado na análise por janelas de curto tempo. A fim de avaliar o sistema desenvolvido, foi construído uma base de dados contendo áudios gravados com 4 instrumentos musicais diferentes e seus respectivos registros temporais de nota, e foi desenvolvido um ambiente de experimentação com interface gráfica para a seleção dos áudios da base, e geração dos gráficos para a avaliação de desempenho da detecção. Por fim, foram executados os experimentos e calculado a métrica de percentual de sucesso nas detecções, avaliou-se os resultados obtidos identificando também os eventuais problemas encontrados pelo sistema no momento da detecção. Discutiu-se também a metodologia adotada, buscando evidenciar os aspectos positivos e negativos da mesma, bem como as melhorias que poderiam ser incorporadas a fim de tornar o sistema mais robusto.

Os conceitos abordados neste trabalho, relacionados ao processamento e análise de sinais digitais, colaboraram para uma melhor compreensão do tema, e permitindo um melhor desenvolvimento do projeto. O desenvolvimento deste proporcionou um aprofundamento prático em relação à detecção de f_0 no ambiente musical e pode-se observar as diferentes características do som de cada instrumento musical quando na análise pelo domínio da frequência. Também verificou-se que métodos de detecção de f_0 baseados na obtenção da frequência de maior amplitude estão vulneráveis aos efeitos de ressonância, sendo necessário um tratamento específico para este problema, entretanto, o método apresentou bons resultados quando em sons não muito graves. Outra importante contribuição deste projeto é a interface gráfica de experimentação, que mostrou-se prática e eficaz para a avaliação do método de detecção adotado, e pode ser aproveitada em outros trabalhos

neste tema.

Espera-se que este trabalho seja um incentivo para novos projetos em Processamento Digital de Sinais voltados para a área musical. Muitos outros métodos de detecção de frequência fundamental tem sido apresentados e desenvolvidos, sendo necessário o trabalho de comparar esses diferentes métodos para obter-se o mais apropriado para a detecção desejada. Outro esforço também pode ser empreendido na intenção de reforçar o método adotado neste projeto, para que se alcance bons resultados mesmo em análise de sons graves.

Referências Bibliográficas

- [1] VASS, J. Automatic Transcription of Audio Signals. *Master of Science Thesis, Czech Technical University in Prague*, Prague, 2004.
- [2] AMADO, R. G.; FILHO, J. V. Pitch Detection Algorithms Based on Zero-Cross Rate and Autocorrelation Function for Musical Notes. *Sao Paulo State University*, São Paulo, 2008.
- [3] AOQUI, C. J. et al. Methods to Musical Notes Recognition from Sheet Music: Preliminary Results. *X Workshop de Visão Computacional - Universidade Estadual de São Paulo*, São Paulo, 2014.
- [4] MONARD, M.; BARANAUSKAS, J. *Conceitos sobre aprendizado de máquinas. Em Sistemas Inteligentes: Fundamentos e Aplicações*. 1º. ed. [S.l.]: Editora Manole, 2003.
- [5] CHAPELLE, O.; SCHÖLKOPF, B.; ZIEN, A. *Semi-Supervised Learning*. 1º. ed. [S.l.]: MIT Press, Cambridge, 2006. 12 – 13 p.
- [6] COVER, T.; HART, P. Nearest neighbor pattern classification. *Information Theory, IEEE Transaction on*, 1967.
- [7] ZHANG, G.; PATUWO, B. E.; HU, M. Y. Forecasting with artificial neural networks: The state of art. *International Journal of Forecasting*, 1998.
- [8] HAYKIN, S. S. *Neural networks and learning machines*. 3º. ed. Upper Saddle River, United States of America: Prentice Hall.
- [9] FERNEDA, E. Redes neurais e sua aplicação em sistemas de recuperação de informação. *Ciência da Informação*, v. 35, n. 1, 2006. ISSN 1518-8353. Disponível em: <<http://revista.ibict.br/ciinf/article/view/1149>>.

- [10] MCCULLLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 1943. ISSN 1522-9602. Disponível em: <<https://doi.org/10.1007/BF02478259>>.
- [11] LIPPMANN, R. An introduction to computing with neural nets. *IEEE ASSP Magazine*, v. 4, n. 2, 1987.
- [12] RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Learning representations by back-propagating errors. *Nature*, Nature Publishing Group, v. 323, p. 533–, out. 1986. Disponível em: <<http://dx.doi.org/10.1038/323533a0>>.
- [13] GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning*. [S.l.]: MIT Press, 2016. <http://www.deeplearningbook.org>.
- [14] HARTMANN, W. Pitch, periodicity, and auditory organization. *Journal of the Acoustical Society of America*, n. 100(6), p. 3491 – 3502, 1996.
- [15] KLAPURI, A.; DAVY, M. *Signal Processing Methods for Music Transcription*. 1^a. ed. US: New York: Springer Science, 2006.
- [16] GERHARD, D. Pitch Extraction and Fundamental Frequency: History and Current Techniques. *Technical Report TR-CS*, n. 2003-6, 2003.
- [17] LATHI, B. *Sinais e Sistemas Lineares*. 2^a. ed. Porto Alegre: Bookman, 2007.
- [18] MATHWORKS. Documentation: audioread - Read audio file. 2018. Disponível em: <<https://www.mathworks.com/help/matlab/ref/audioread.html>>. Acesso em: 13 jul. 2018.
- [19] OPPENNHEIM, A. V.; SCHAFER, R. W.; BUCK, J. R. *Discrete-Time Signal Processing*. 2^a. ed. New Jersey: Prentice Hall, 1999.
- [20] MATHWORKS. Documentation: fft - Fast Fourier Transform. 2018. Disponível em: <<https://www.mathworks.com/help/matlab/ref/fft.html>>. Acesso em: 11 jul. 2018.
- [21] MATHWORKS. Documentation: spectrogram - Spectrogram using Short-Time Fourier Transform. 2018. Disponível em: <<https://www.mathworks.com/help/signal/ref/spectrogram.html>>. Acesso em: 11 jul. 2018.

- [22] MATHWORKS. Documentation: guide - Open GUIDE. 2018. Disponível em: <<https://www.mathworks.com/help/matlab/ref/guide.html>>. Acesso em: 21 jul. 2018.
- [23] CRUZ JR, A. C. V. F0Detection - Bitbucket Repository. 2018. Disponível em: <<https://bitbucket.org/juninhocruz/f0detection>>. Acesso em: 21 jul. 2018.
- [24] AUDACITY. Audacity: Free, open source, cross-platform audio software. 2018. Disponível em: <<https://www.audacityteam.org>>. Acesso em: 21 jul. 2018.